Errata

Title & Document Type: 3852A Data Acquisition/Control Unit Mainframe Configuration and Programming Manual

Manual Part Number: 03852-90015

Revision Date: December 1, 1987

HP References in this Manual

This manual may contain references to HP or Hewlett-Packard. Please note that Hewlett-Packard's former test and measurement, semiconductor products and chemical analysis businesses are now part of Agilent Technologies. We have made no changes to this manual copy. The HP XXXX referred to in this document is now the Agilent XXXX. For example, model number HP8648A is now model number Agilent 8648A.

About this Manual

We've added this manual to the Agilent website in an effort to help you support your product. This manual provides the best information we could find. It may be incomplete or contain dated information, and the scan quality may not be ideal. If we find a better copy in the future, we will add it to the Agilent website.

Support for Your Product

Agilent no longer sells or supports this product. You will find any other available product information on the Agilent Test & Measurement website:

www.tm.agilent.com

Search for the model number of this product, and the resulting product page will guide you to any available information. Our service centers may be able to perform calibration if no repair parts are needed, but no other support from Agilent is available.

Agilent Technologies

www.valuetronics.com

HP 3852A Data Acquisition/Control Unit

Mainframe Configuration and Programming Manual

Copyright© Hewlett-Packard Company, 1987

Manual Part Number: 03852-90015
Microfiche Part Number: 03852-99015

Printed: DECEMBER 1987 Edition 2
Printed in U.S.A. E1287

www.valuetronics.com

HEWLETT PACKARD

Printing History

The Printing History shown below lists the printing dates of all Editions and Updates created for this manual. The Edition number changes as the manual undergoes subsequent revisions. Editions are numbered sequentially starting with Edition 1.

Updates, which are issued between Editions, contain individual replacement pages which the customer uses to update the current Edition of the manual. Updates are numbered sequentially starting with Update 1. When a new Edition is created, all Updates associated with the previous Edition are merged into the manual. Each new Edition or Update also includes a revised copy of this printing history page.

Many product updates and revisions 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 product updates and manual updates.

Edition 1 (Part Number 03852-90007) MARCH 1987
Update 1 (Part Number 03852-90090) JULY 1987
Edition 2 (Part Number 03852-90015) DECEMBER 1987

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subdivision (b)(3)(ii) of the Rights in Technical Data and Computer Software clause at 52.227-7013.

Hewlett-Packard Company
3000 Hanover Street
Palo Alto, California 94304

www.valuetronics.com

HEWLETT PACKARD

Herstellerbescheinigung

Hiermit wird bescheinigt, daß das Gerät/System HP 3852A in Übereinstimmung mit den Bestimmungen von Postverfügung 1046/84 funkentstört ist.

Der Deutschen Bundespost wurde das Inverkehrbringen dieses Gerätes/Systems angezeigt und die Berechtigung zur Überprüfung der Serie auf Einhaltung der Bestimmungen eingeräumt.

Zusatzinformation fur Meß- und Testgeräte

Werden Meß- und Testgeräte mit ungeschirmten Kabeln und/oder in offenen Meßaufbauten verwendet, so ist vom Betreiber sicherzustellen, daß die Funk-Entstörbestimmungen unter Betriebsbedingungen an seiner Grundstücksgrenze eingehalten werden.

Manufacturer's declaration

This is to certify that the equipment HP 3852A is in accordance with the Radio Interference Requirements of Directive FTZ 1046/84. The German Bundespost was notified that this equipment was put into circulation, the right to check the series for compliance with the requirements was granted.

Additional Information for Test- and Measurement Equipment

If Test- and Measurement Equipment is operated with unscreened cables and/or used for measurements on open set-ups, the user has to assure that under operating conditions the Radio Interference Limits are still met at the border of his premises.

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

This document contains proprietary information which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced or translated to another language without the prior written consent of Hewlett-Packard Company.

Copyright © 1985 by HEWLETT-PACKARD COMPANY

www.valuetronics.com

HEWLETT PACKARD

CERTIFICATION

Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the United States National Bureau of Standards, to the extent allowed by the Bureau's calibration facility, and to the calibration facilities of other International Standards Organization members.

WARRANTY

This Hewlett-Packard instrument product is warranted against defects in materials and workmanship for a period of one year from date of shipment. During the warranty period, Hewlett-Packard Company will, at its option, either repair or replace products which prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by -hp-. Buyer shall prepay shipping charges to -hp- and -hp- shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to -hp- from another country.

Duration and conditions of warranty for this instrument may be superceded when the instrument is integrated into (becomes a part of) other -hp- instrument products.

Hewlett-Packard warrants that its software and firmware designated by -hp- for use with an instrument will execute its programming instructions when properly installed on that instrument. Hewlett-Packard does not warrant that the operation of the instrument, or software, or firmware will be uninterrupted or error free.

LIMITATION OF WARRANTY

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HEWLETT-PACKARD SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

EXCLUSIVE REMEDIES

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. HEWLETT-PACKARD SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

ASSISTANCE

Product maintenance agreements and other customer assistance agreements are available for Hewlett-Packard products.

For any assistance, contact your nearest Hewlett-Packard Sales and Service Office. Addresses are provided at the back of this manual.

F

www.valuetronics.com

SAFETY SUMMARY

The following general safety precautions must be observed during all phases of operation, service, and repair of this instrument. Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the instrument. Hewlett-Packard Company assumes no liability for the customer's failure to comply with these requirements.

GROUND THE INSTRUMENT

To minimize shock hazard, the instrument chassis and cabinet must be connected to an electrical ground.

DO NOT OPERATE IN AN EXPLOSIVE ATMOSPHERE

Do not operate the instrument in the presence of flammable gases or fumes. Operation of any electrical instrument in such an environment constitutes a definite safety hazard.

KEEP AWAY FROM LIVE CIRCUITS

Operating personnel must not remove instrument covers. Component replacement and internal adjustments must be made by qualified maintenance personnel. Under certain conditions, dangerous voltages may exist even with the instrument switched off. To avoid injuries, always disconnect input voltages and discharge circuits before touching them.

DO NOT SERVICE OR ADJUST ALONE

Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.

DO NOT SUBSTITUTE PARTS OR MODIFY INSTRUMENT

Because of the danger of introducing additional hazards, do not install substitute parts or perform any unauthorized modification to the instrument. Return the instrument to a Hewlett-Packard Sales and Service Office for service and repair to ensure that safety features are maintained.

DO NOT OPERATE A DAMAGED INSTRUMENT

Whenever it is possible that the safety protection features built into this instrument have been impaired, either through physical damage, excessive moisture, or any other reason, REMOVE POWER and do not use the instrument until safe operation can be verified by service-trained personnel. If necessary, return the instrument to a Hewlett-Packard Sales and Service Office for service and repair to ensure that safety features are maintained.

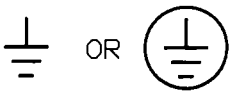
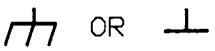
C

www.valuetronics.com

HEWLETT PACKARD

Operating and Safety Symbols

Symbols Used On Products And In Manuals

~ LINE	AC line voltage input receptacle.
	Instruction manual symbol affixed to product. Warns and cautions the user to refer to respective instruction manual procedures to avoid personal injury or possible damage to the product.
	Indicates dangerous voltage – terminals connected to interior voltage exceeding 1000 volts.
	Protective conductor terminal. Indicates the field wiring terminal that must be connected to earth ground before operating equipment – protects against electrical shock in case of fault.
	Clean ground (low-noise). Indicates terminal that must be connected to earth ground before operating equipment – for single common connections and protection against electrical shock in case of fault.
	Frame or chassis ground. Indicates equipment chassis ground terminal – normally connects to equipment frame and all metal parts.
	Affixed to product containing static sensitive devices – use anti-static handling procedures to prevent electrostatic discharge damage to components.

NOTE

Calls attention to a procedure, practice, or condition that requires special attention by the reader.

CAUTION

Calls attention to a procedure, practice, or condition that could possibly cause damage to equipment or permanent loss of data.

WARNING

Calls attention to a procedure, practice, or condition that could possibly cause bodily injury or death.

---

www.valuetronics.com

WARNING, CAUTION, and NOTE Symbols

Some labels on the HP 3852A, HP 3853A, and plug-in accessories include an international warning symbol (triangle with subscripted number) which refers the reader to the manuals for further information. This table shows the warning symbols used for the HP 3852A/3853A and plug-in accessories. Refer to the manual set for specific information on WARNINGS, CAUTIONS, or NOTES referenced with a warning symbol.

HP 3852A WARNING, CAUTION, and NOTE Symbols

Symbol	Meaning	Location
	Shock hazard originating outside the instrument (field wiring)	. Analog Extender Connector on Power Supply Modules . Terminal modules on plug-in accessories . Component module covers on plug-in accessories
	Treat all channels as "one circuit" for safety purposes.	. Inside terminal modules on plug-in accessories . Metal cover on component modules of plug-in accessories
	Maximum number of certain plug-in accessories to be installed into an HP 3852A or HP 3853A.	. HP 44701A, HP 44702A/B, HP 44727A/B/C plug-in accessories
	If High-Speed FET multiplexers are used with the HP 44702A/B, ribbon cable may be connected.	. HP 44711A, 44712A, 44713A (referenced on HP 44702A and HP 44702B)
	The instrument should not be operated at a line frequency of 440Hz with a line voltage of 200 V or greater as the AC leakage current may exceed 3.5mA	. HP 3852A, HP 3853A Power Supply Modules

www.valuetronics.com

TABLE OF CONTENTS

www.valuetronics.com

www.valuetronics.com

CHAPTER 11 - HP 3852A MULTITASKING

www.valuetronics.com

Contents

www.valuetronics.com

Chapter 1

Using the Manuals

Introduction

Information on how to operate the HP 3852A Data Acquisition/Control Unit is contained in several manuals. The purpose of this chapter is to provide an overview of the mainframe manual, identify the other manuals used with the HP 3852A, and describe how the manuals interact.

Mainframe Configuration and Programming Manual

The Mainframe Configuration and Programming Manual can probably be considered as the core of the documentation set. Although as a stand-alone instrument the HP 3852A mainframe has very limited data acquisition and control capabilities, it is the mainframe that enables you to operate the plug-in accessories, manipulate data, and manage your system in general.

The Mainframe Configuration and Programming Manual is divided into 10 chapters. The purpose and content of each chapter is given below.

Manual Content

In addition to this chapter, there are ten other chapters in the Mainframe Configuration and Programming Manual:

Chapter 2 - Meet the HP 3852A

Chapter 3 - Installing the HP 3852A

Chapter 4 - Front Panel Operation

Chapter 5 - HP-IB Communication

Chapter 6 - Managing Data

Chapter 7 - Triggering and Pacing

Chapter 8 - Using Interrupts

Chapter 9 - Using Subroutines

Chapter 10 - Data Processing

Chapter 11 - HP 3852A Multitasking

www.valuetronics.com

Chapter 2 - Meet the HP 3852A

Chapter 2 is intended to introduce you to the HP 3852A. This chapter contains a data acquisition and control overview plus front and rear panel tours of the HP 3852A mainframe and HP 3853A extender which identify many of the instruments' features.

Chapter 3 - Installing the HP 3852A

This chapter describes how the HP 3852A mainframe is prepared for use. The chapter also contains information on how the HP 3853A extender is installed, how the plug-in accessories and extended memory boards are installed, and how an HP-IB cable is connected and how the mainframe's HP-IB address is set.

Chapter 4 - Front Panel Operation

This chapter explains how the HP 3852A mainframe is operated from its front panel. The chapter describes how various characters are accessed on the front panel, how the display is controlled, and how commands are entered.

Chapter 5 - HP-IB Communication

This chapter explains how the HP 3852A is operated from a controller over the Hewlett Packard Interface Bus (HP-IB). The chapter covers how commands are sent to the mainframe and how data from the mainframe/plug-in accessories is entered into the controller. The various remote/local states of the HP 3852A are also addressed.

Chapter 6 - Managing Data

This chapter explains how command and measurement data is directed, stored, and presented in an HP 3852A based system. The chapter describes the HP 3852A addressing convention used to direct commands to specific plug-in accessories, and how to determine the format and destination (e.g. display, output buffer, memory) of data acquired. The chapter explains how to store and retrieve data in the mainframe's internal memory and how to manage data in a packed (high-speed) format.

Chapter 7 - Triggering and Pacing

This chapter explains how measurement and control functions performed by the HP 3852A can be initiated and regulated by the mainframe's triggering, scanning, and pacing signals. This chapter also describes settings and applications of the mainframe's real-time clock.

Chapter 8 - Using Interrupts

This chapter explains how to use the interrupt capabilities of the HP 3852A mainframe and the plug-in accessories to alter or suspend the operation of the mainframe or the controller. The chapter describes how interrupts are serviced by the mainframe and how the mainframe or controller can be specified to respond to the interrupt.

Chapter 9 - Using Subroutines

This chapter explains how HP 3852A subroutines can be used to send commands to the HP 3852A mainframe/plug-in accessories. The chapter identifies the commands used to set up subroutines and covers how subroutines are downloaded into mainframe memory.

www.valuetronics.com

Chapter 10 - Data Processing

This chapter describes how to use the data processing functions of the HP 3852A on data that has been acquired. Data processing functions include channel logging, scaling, statistics, post processing and real time limit checking, and linear interpolation of data using user-defined lookup tables.

Chapter 11 - HP 3852A Multitasking

This chapter covers the multitasking feature of the HP 3852A. The chapter describes how HP 3852A multitasking works and the multitasking commands that are used. The chapter also contains detailed multitasking examples. Note that the multitasking capability is only available with firmware revision 3.0 or greater.

Additional Manuals

Besides the Mainframe Configuration and Programming Manual, the user manuals shipped with the HP 3852A include the Getting Started Guide, Command Reference Manual, Quick Reference Guide, and Plug-In Accessory Configuration and Programming Manuals for the specific accessories you ordered (Figure 1-1). This section provides an overview of the content of those manuals.

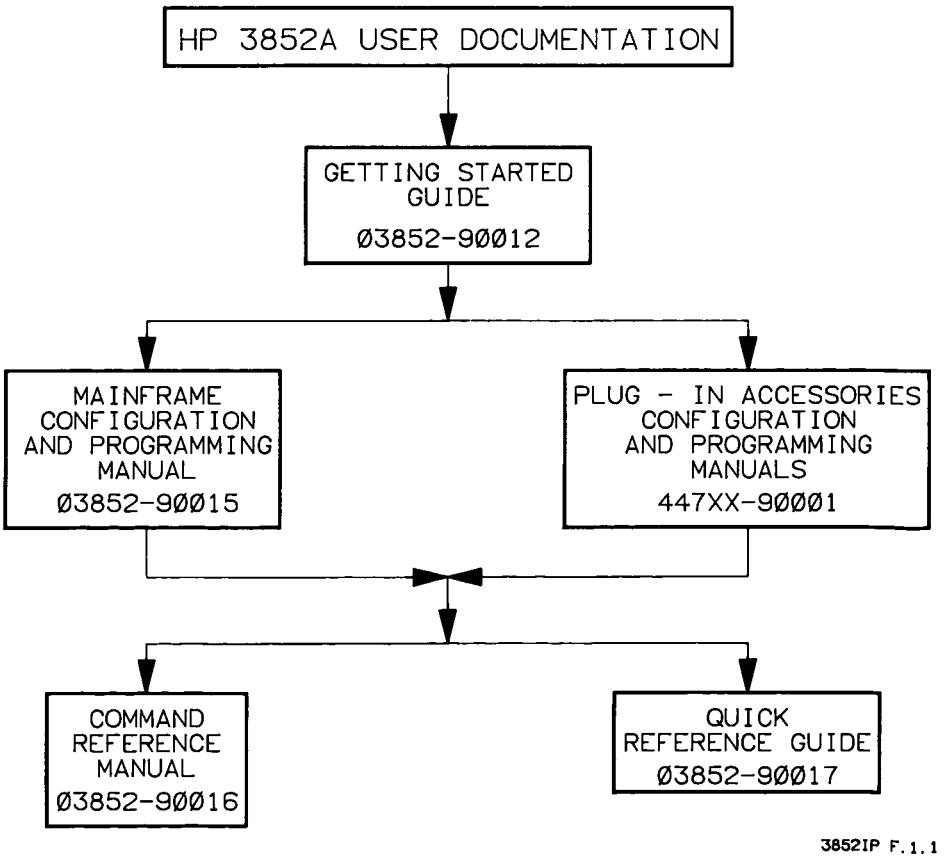


Figure 1-1. HP 3852A User Documentation Map

www.valuetronics.com

Getting Started Guide

When you received your HP 3852A, the first manual used should have been the Getting Started Guide. The purpose of this manual is to quickly familiarize you with the HP 3852A and acquaint you with the capabilies of the instrument with the help of the Getting Started Kit (part number 03852-67901). The manual does not contain detailed reference information on any of the capabilities identified. Reference information of this sort is contained in the mainframe manual and the individual plug-in accessory manuals.

Plug-In Accessory Manuals

A Plug-In Accessory Configuration and Programming Manual is shipped with each plug-in accessory ordered. These manuals contain accessory configuration and wiring information, and information on the commands which program the accessory for its intended functions.

Note that these manuals generally do not contain information on sending commands, data management, etc. You will need to refer to the Mainframe Configuration and Programming Manual for this type of information.

Command Reference Manual

This manual contains an alphabetic listing of all commands associated with the HP 3852A mainframe and all plug-in accessories. Each command entry includes a description of the command, prerequisite commands or configurations, the command's syntax, programming examples, and other relevant information. The manual also contains tables of the HP 3852A addressing convention, the HP 3852A mainframe and plug-in accessory power-on states, data formats, packed data conversions, and error messages.

The Command Reference Manual is intended for the user who has experience in configuring and programming the mainframe and the plug-in accessories. This manual does not contain introductory operating information nor is there tutorial information on any particular data acquisition or control feature. The manual does provide detailed programming information on every mainframe and accessory command in one convenient location.

Quick Reference Guide

The Quick Reference Guide is basically a pocket-sized version of the Command Reference Manual. The guide contains summary information on the HP 3852A's addressing convention, syntax rules, data destinations and formats, and an alphabetical listing of all mainframe and plug-in accessory commands. Each command listed contains a description of the command and where it is used, its syntax, and a description of its parameters. The guide also contains information on using interrupts, plus tables of the mainframe and accessory power-on states, accessory channel ranges, data formats, packed data conversions, and error messages.

www.valuetronics.com

How the Manuals Interact

As stated previously, to set up and execute a data acquisition or control function will initially require the use of more than one HP 3852A manual. To demonstrate how specific manuals interact, assume you want to perform DC voltage measurements under the following conditions:

- The DC voltage measurements are to be made using an HP 44701A voltmeter accessory and an HP 44705A multiplexer accessory
- The HP 3852A is to be operated using a controller
- The measurements are to be stored in HP 3852A mainframe memory

To accomplish this task will require the Mainframe Configuration and Programming Manual and the configuration and programming manuals for the HP 44701A voltmeter and HP 44705A 20-Channel Relay Multiplexer. The interaction of these manuals is shown in Figure 1-2.

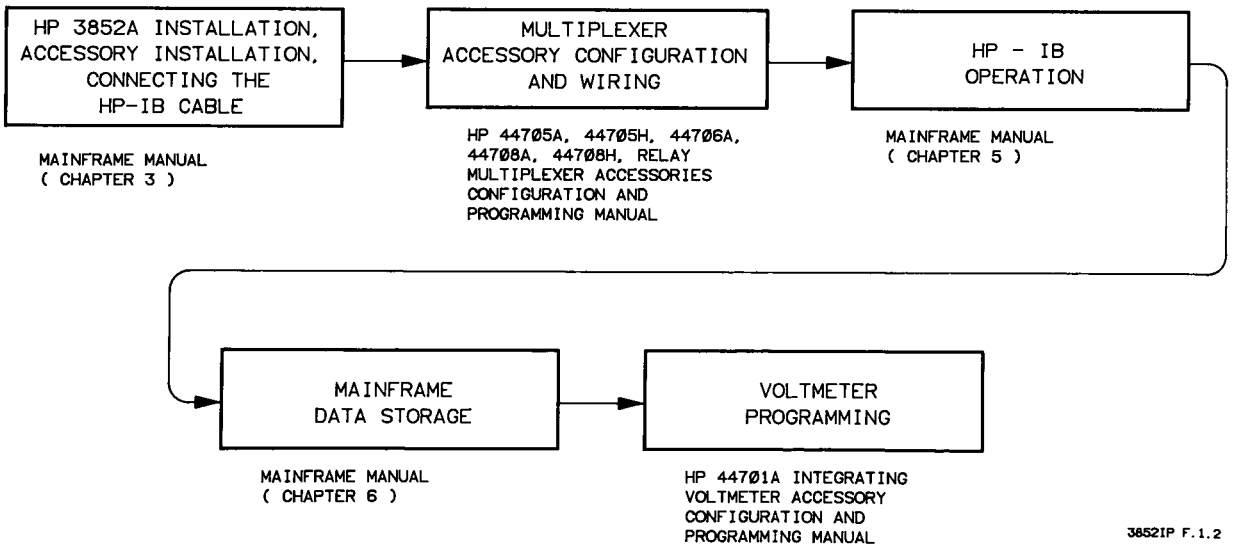


Figure 1-2. HP 3852A User Manual Interaction

www.valuetronics.com

As your tasks become more complex (e.g. subroutines, interrupts, or data processing are involved), references to additional chapters in the mainframe manual and plug-in accessory manuals will be required. As you become more familiar with your application and with the operation of the HP 3852A, usually reference to only the Command Reference Manual or Quick Reference Guide is required.

The sequence in which the manuals are referenced is irrelevant. What is important is knowing that generally more than one manual is required to set up and execute data acquisition and control functions.

www.valuetronics.com

Contents

www.valuetronics.com

Chapter 2

Meet the HP 3852A

Introduction

The purpose of this chapter is to help you understand more about the HP 3852A and its role in a data acquisition and control system. To do this, the chapter has been divided into two sections: Data Acquisition/Control Overview and Equipment Profiles.

- **Data Acquisition/Control Overview** describes general data acquisition and control functions and how they are performed by an HP 3852A based system. Data acquisition and control involving the HP 3852A are not stand-alone functions. Communication is required to initiate and dictate system operation and some form of data processing is usually involved. As a result, both functions are briefly discussed in the overview.

- **Equipment Profiles** contains front and rear panel tours of the HP 3852A and the HP 3853A Extender.

Data Acquisition/Control Overview

A data acquisition/control system contains various types of electronic equipment that measure, process, and analyze data acquired from the user's application. A data acquisition/control system can also control or regulate the user's application or process in response to the data obtained.

Figure 2-1 shows a data acquisition/control system where these functions are performed by the HP 3852A. The HP 3852A as a stand-alone instrument is not capable of acquiring data and has very limited control capabilities. Although not shown in the illustration, it is the plug-in accessories that enable the HP 3852A to function as a data acquisition/control unit. Therefore, references to HP 3852A capabilities assumes the HP 3852A with the appropriate accessories.

www.valuetronics.com

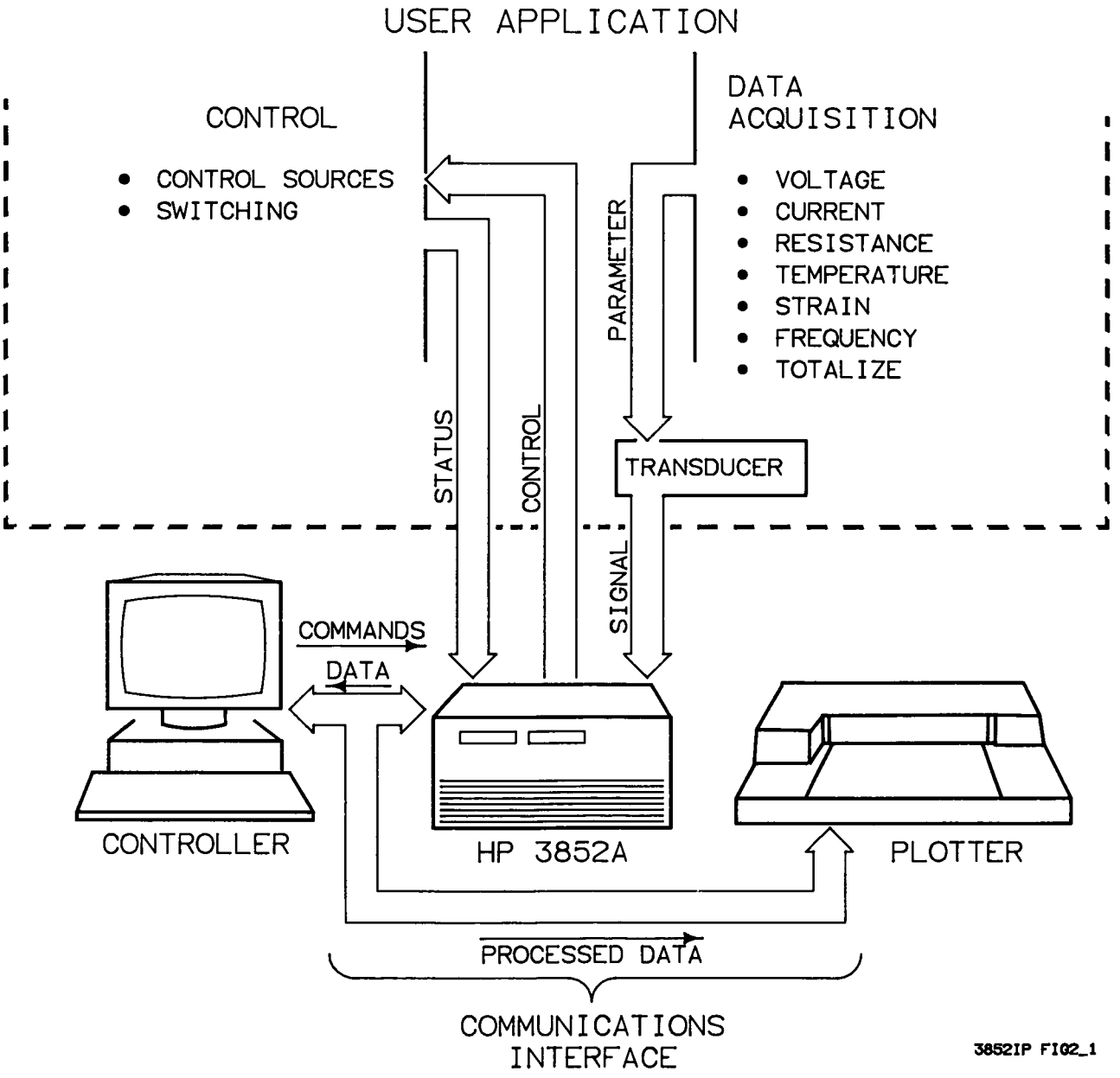


Figure 2-1. Data Acquisition/Control System

www.valuetronics.com

Data Acquisition

The function of the data acquisition task is to measure or count data supplied by the user's application. On command, the HP 3852A receives data from the user's application through the channels of various plug-in accessories. Often a transducer as shown in Figure 2-1 will be connected between an application and the input to the HP 3852A. A transducer is a device that converts a physical parameter (e.g. temperature) into an electrical signal (voltage) the HP 3852A can measure.

Depending on the type of data received, the HP 3852A may route the data to a voltmeter accessory for measurement, display the data directly, or initiate some form of control in response to the data received. In addition to displaying data, the data can also be processed and stored within the HP 3852A or transferred to a controller for further processing or storage.

The HP 3852A acquires data through measurements of voltage, current, resistance, temperature, strain, and frequency. The instrument is also capable of counting and sensing low level AC signals and digital inputs.

Control

The HP 3852A in Figure 2-1 can be instructed by the command set to control the application directly or as a response to a condition sensed or event which occurs.

The HP 3852A is capable of supplying voltages and currents for testing or driving external devices. The instrument can also switch voltage, current, and AC power supplied externally from one channel to another.

The HP 3852A can be used to sense open/closed, on/off, or present/absent conditions and respond by sending the appropriate control signal or by initiating a measurement. If necessary, the HP 3852A can notify the controller when the condition occurs then let the controller respond.

Communication

Communication with the HP 3852A is through the command set. The keys on the HP 3852A front panel are labeled with command headers and parameters. "Shifting" the keyboard re-defines many of the keys to a letter in the alphabet and its position corresponds to the position on a typewriter. With the keyboard shifted, you can enter any command from the instrument's front panel.

The HP 3852A can also be programmed from a controller over the HP-IB.*

*The Hewlett-Packard Interface Bus (HP-IB) is Hewlett-Packard's implementation of IEEE Standard 488-1978 and ANSI MC1.1.

www.valuetronics.com

Subroutines can be downloaded into the mainframe's memory then called on command from the controller or front panel. Subroutines can also be called when an interrupt condition or event occurs. BASIC constructs such as FOR...NEXT, IF...END IF, and WHILE...END WHILE are built into the HP 3852A and can be used to control and synchronize your subroutines.

Data Processing

The HP 3852A is capable of several data processing functions involving the data acquired. The functions include channel logging, post process and real time limit checking, scaling, statistics, and data conversions using user defined lookup tables. With a controller and plotter you can make plots, graphs, and charts of the processed data.

Equipment Profiles

The equipment profiles in this section consist of front and rear panel tours of the HP 3852A mainframe and HP 3853A extender.

The HP 3852A

The following tour of the HP 3852A is intended to help you recognize features and capabilities of the instrument rather than point out an ominous collection of keys, slots, and connectors. The tour highlights key groupings that reflect the instrument's capabilities and identifies not only which ports exist, but why they exist.

www.valuetronics.com

www.valuetronics.com

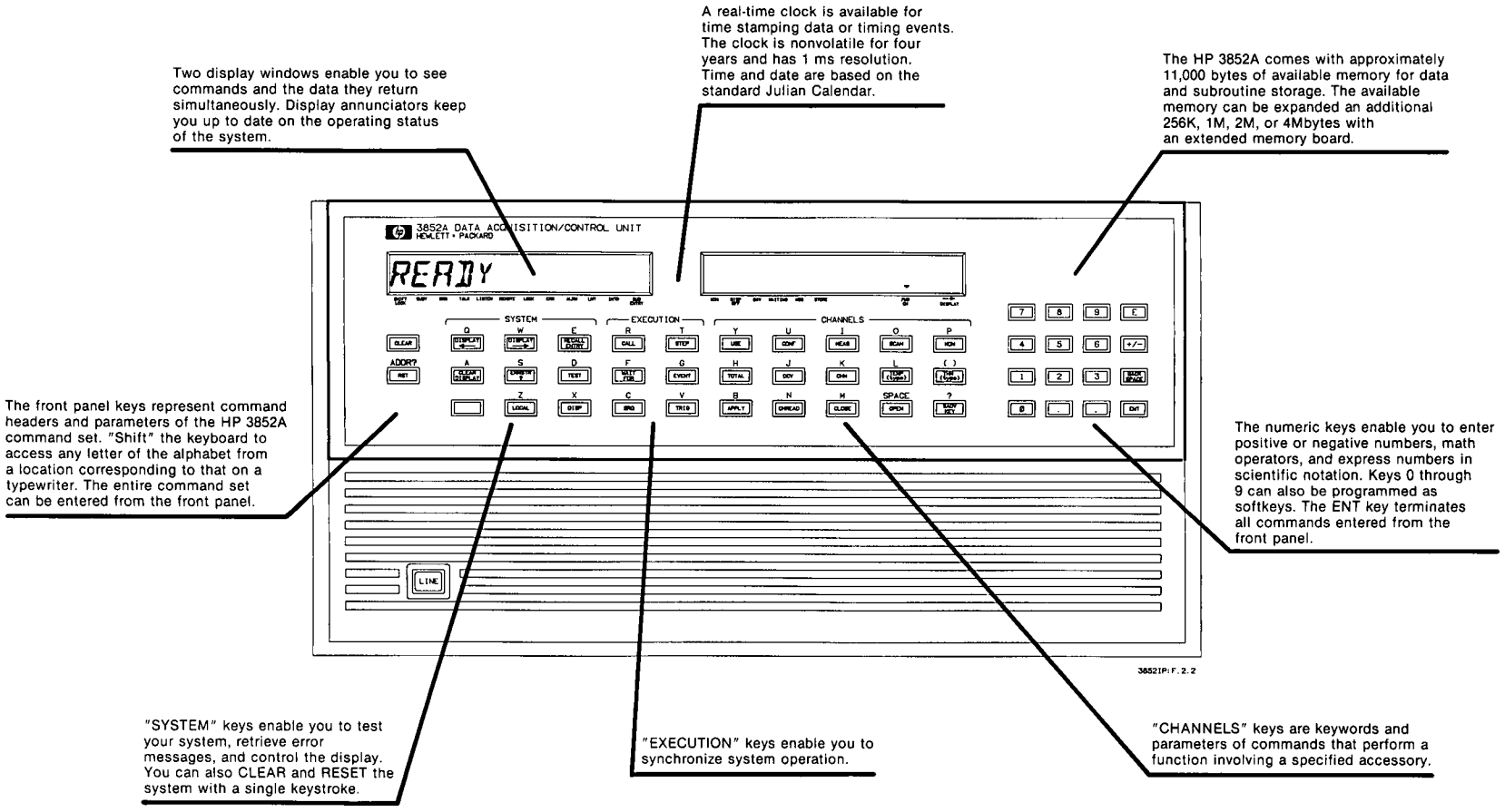


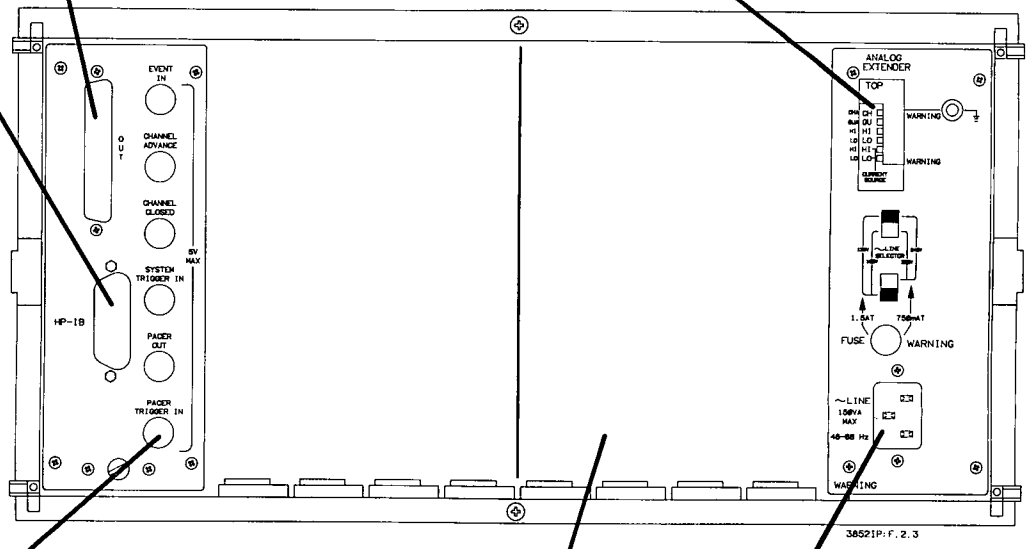
Figure 2-2. HP 3852A Mainframe Front Panel

www.valuetronics.com

Use the Digital Extender Out port to connect up to seven HP 3853A extenders to the HP 3852A mainframe. The digital extender cable routes data and programming information between the mainframe and the accessories installed in the extenders.

The Analog Extender port enables you to extend the HP 3852A's backplane analog bus. Connect the ports of the mainframe and extenders to route analog signals from multiplexer accessories installed in the extenders to voltmeter accessories installed in the mainframe. Multiplexers can also be linked to an external voltmeter via this port. The analog port terminals include chassis ground (CH), guard (GU), high and low sense (HI, LO), and high and low current source (HI, LO)-current source).

Use this port and an HP-IB cable to connect the HP 3852A to a controller. The HP 3852A has a factory set HP-IB address of 9.



Use the BNC ports to trigger and synchronize operation with external devices or use them as the input source for initiating system activities. All BNC input ports are TTL compatible and require pulse widths > 0.5 usec. Output ports can drive CMOS/HCMOS, one TTL load, or two LSTTL loads.

Enable the EVENT IN port with the WAITFOR EVENT command then initiate or suspend system operation using an external device.

Enable the CHANNEL ADVANCE port with the STRIG CHADV command then input a signal to initiate a backplane scan and measurement of a list of channels. Enable the port with the SADV CHADV command then connect the port to an external voltmeter that in turn, will advance the next channel when the measurement is complete.

The CHANNEL CLOSED port outputs a negative-going pulse when a channel is closed. Use the port to trigger an external voltmeter to start a measurement.

Enable the SYSTEM TRIGGER IN port with the TRG EXT command then trigger all accessories with an external device.

The PACER OUT port provides 500 nsec negative-going pulses with a period programmable from 1 usec to 4.19 sec with 250 nsec resolution. Output a continuous stream of pulses or a fixed number from 1 to 65535 to synchronize or control an external event.

Enable the PACER TRIGGER IN port with the PTRIG EXT command then trigger the pacer to output a continuous stream or fixed number of pulses.

Install up to eight accessories with any accessory in any slot. The plastic slot latch prevents the accessory from being pulled out from the backplane edge connector when the terminal module is removed for configuration and wiring. The metal plate provides support should other instruments be stacked on the mainframe.

The HP 3852A detects the line frequency and automatically configures itself for 50 Hz or 60 Hz operation.

Figure 2-3. HP 3852A Mainframe Rear Panel

The HP 3853A

The HP 3853A Extender enables you to expand your system with no loss of functional capability. Up to seven extenders can be used with a single mainframe. With 10 slots per extender, up to 78 slots are available. Plug-in accessories can be installed in any mainframe slot.

Extenders are connected to the HP 3852A by digital and analog extender cables. The digital extender cable carries data and programming information (including interrupts and triggers) between the mainframe and the accessories in the extender. The analog extender cable connects the backplane bus of the mainframe to the backplane bus of the extender. The bus is used to route analog signals from the various multiplexer accessories to the voltmeter accessories. When connected, the analog cable enables a voltmeter accessory to be installed in the mainframe and multplexer accessories to be installed in the extenders.

www.valuetronics.com

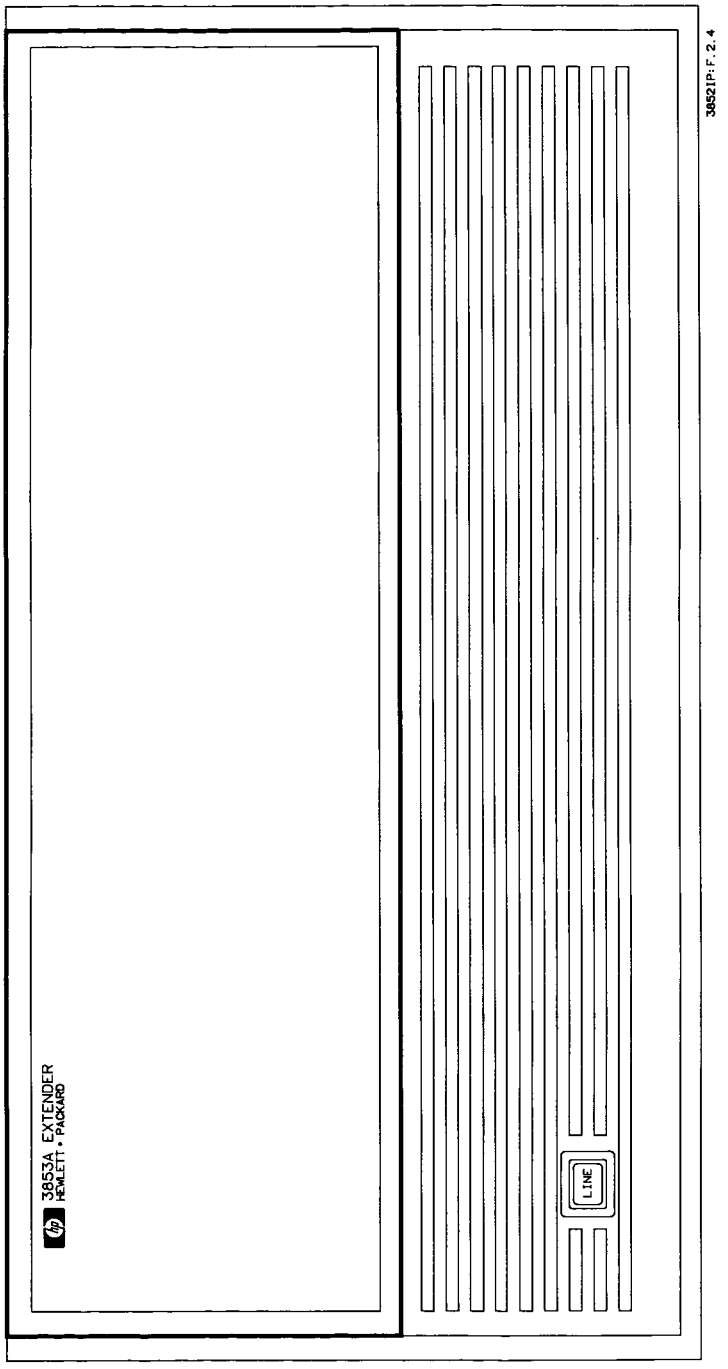


Figure 2-4. HP 3853A Extender Front Panel

www.valuetronics.com

www.valuetronics.com

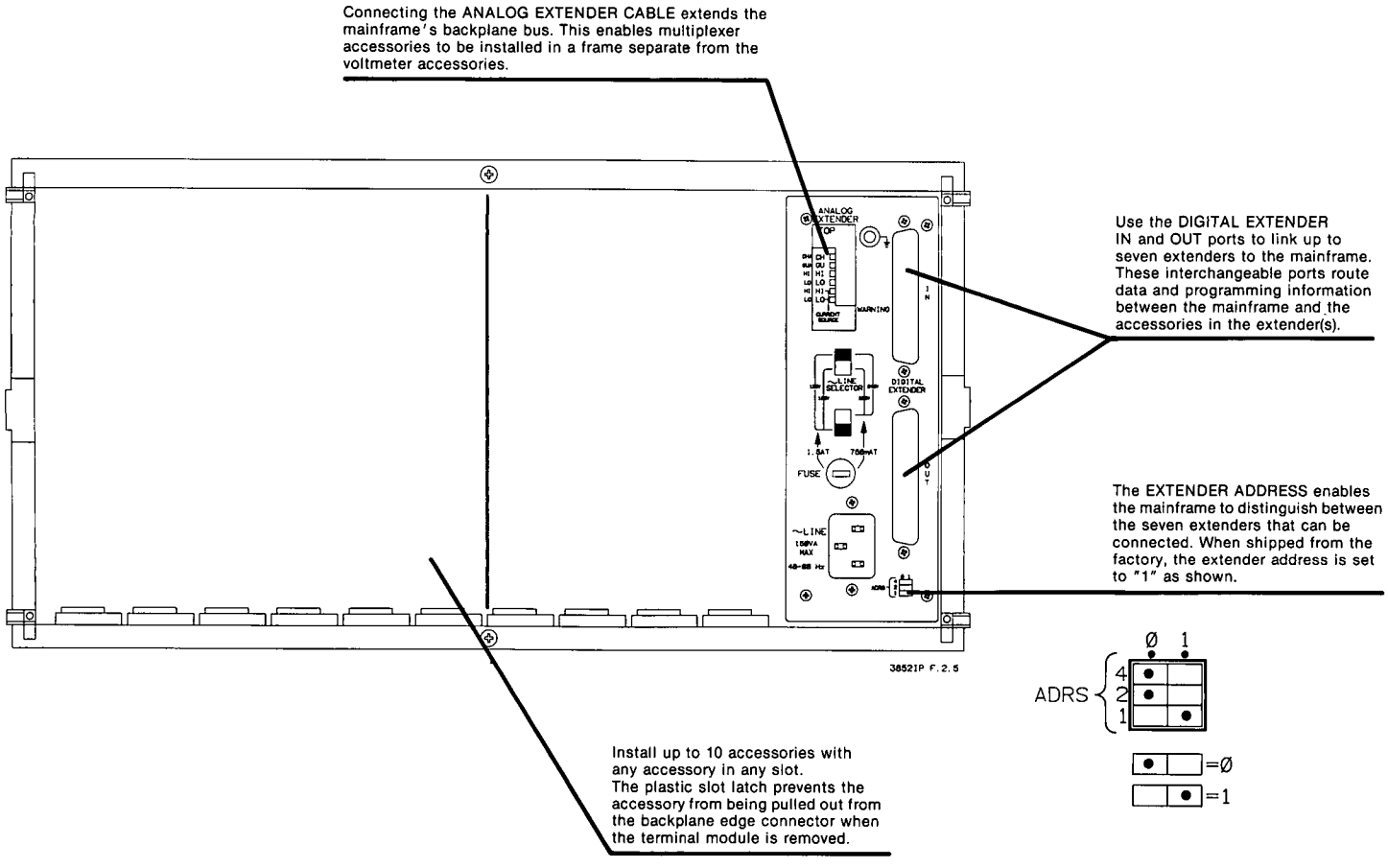


Figure 2-5. HP 3853A Extender Rear Panel

Contents

www.valuetronics.com

Chapter 3

Installing the HP 3852A

Introduction

The purpose of this chapter is to help qualified, service-trained personnel install the HP 3852A Data Acquisition/Control Unit. Since the HP 3852A is not operated as a stand-alone instrument, installation of the HP 3853A Extender and the HP 447XX series of plug-in accessories is also covered here. The chapter concludes by showing you how to connect the HP 3852A to a computer for operation over the HP-IB.*

The installation procedures in this chapter are grouped into five major sections: Introduction, Installing the HP 3852A, Installing the HP 3853A, Installing the Plug-in Accessories, and Connecting the HP-IB. A description of each section is given below.

- **Introduction** reviews the contents of this chapter and describes the operating environment intended for the equipment. The section also identifies several precautions to follow if the equipment is to be re-installed, but is currently operational.

- **Installing the HP 3852A** shows you how to prepare the mainframe for use. Included are procedures for installing the line fuse, setting the line switch, positioning the mainframe, and applying power to the instrument.

- **Installing the HP 3853A** shows you how to install the extender's line fuse, set its line switch and extender number, and connect it to the mainframe.

- **Installing the Plug-in Accessories** shows you how to install the accessories in the mainframe and extender. The section also shows you how to install the extended memory boards.

- **Connecting the HP-IB** provides a brief overview of how the HP-IB is structured and shows you how to set (change) the mainframe's HP-IB address and connect the HP-IB cable.

* The Hewlett-Packard Interface Bus (HP-IB) is Hewlett-Packard's implementation of IEEE Standard 488-1978 and ANSI MC1.1.

www.valuetronics.com

To get your equipment installed and operational as soon as possible, try to "tailor" the installation procedures in this chapter to your particular system (application). It is recommended that you review the entire chapter before beginning any of the procedures. During the review, take note of all Warnings, Cautions, and Notes found in the chapter before omitting any procedures that do not apply.

WARNING

The installation procedures described in this chapter have been written for, and must be performed by, service-trained personnel only. Various procedures within this chapter involve connections that may be made in close proximity to lethal voltages supplied by the user. These voltages can be present on the mainframe and extender backplane even when the instruments are disconnected from AC line power.

WARNING

The installation procedures in this chapter assume that the HP 3852A or any of its associated equipment is not currently installed and operational. If you are re-installing or re-configuring any of the equipment, refer to ***Previous Installation*** *before proceeding.*

NOTE

Plug-in accessory installation in this chapter refers only to the physical placement of the accessories in the HP 3852A mainframe or HP 3853A extender. Accessory wiring, configuration, and programming information can be found in the configuration and programming manual for that particular accessory.

www.valuetronics.com

Previous Installation

As stated, the procedures and guidelines in this chapter assume that the mainframe, plug-in accessories, and any extenders are not currently installed and operational. Before attempting any installation procedures on equipment currently installed, do the following:

1. Turn the mainframe, the controller, and any extenders OFF.
2. Disconnect all external power sources connected to the plug-in accessories or connected to the mainframe or extender backplane analog bus.
3. Disconnect the analog bus extender cable and the digital extender cable between the mainframe and any extenders.
4. Disconnect the HP-IB cable between the HP 3852A and the controller.

Operating Environment

For the equipment to function at its specified level of performance, restrictions are placed on its operating environment. Table 3-1 lists the restrictions regarding operation and storage of the mainframe, extenders, and the plug-in accessories.

Table 3-1. Environmental Restrictions

Parameter	HP 3852A	HP 3853A	All Accessories
Warm up time	1 Hour	1 Hour	1 Hour
Operating Temperature	0° to 55°C	0° to 55°C	0° to 55°C
Storage Temperature	−40° to 75°C	−40° to 75°C	−40° to 75°C
Operating Humidity (Non-Condensing)	28°C @ RH ≤85% 40°C @ RH ≤60% 55°C @ RH ≤25%	28°C @ RH ≤85% 40°C @ RH ≤60% 55°C @ RH ≤25%	28°C @ RH ≤85% 40°C @ RH ≤60% 55°C @ RH ≤25%
Altitude (Operating)	4600m (15000 ft)	4600m (15000 ft)	4600m (15000 ft)
Altitude (Storage)	15300m (50000 ft)	15300m (50000 ft)	15300m (50000 ft)

WARNING

To the Installer

This equipment and software does not provide failsafe control of processes. If loss of control of your application could cause a hazard to personnel and/or property, then additional protection must be provided by the installer.

This equipment is designed for use in temperature, humidity, and pollution controlled environments.

www.valuetronics.com

Installing the HP 3852A

Installation of the HP 3852A mainframe includes the following:

- AC Line Power Requirements
- Power Cord and Grounding Requirements
- Positioning the Mainframe
- Applying Power

AC Line Power Requirements

The mainframe can be set to operate at line voltages of 90 to 132, or 198 to 250 VAC (all values rms), 48 to 440 Hz or 48 to 66 Hz single phase. The power line voltage can vary by ±10% but cannot exceed 250 VAC rms.

CAUTION

Using the wrong fuse value and type for the line voltage present can result in damage to the circuitry inside the instrument.

Connecting the mainframe to the line voltage when the LINE SELECTOR switch is set improperly may blow the fuse.

Installing the Line Fuse

When shipped from the factory, a 1.5 AT (Amperes/time-lag type) fuse, a 3.0 AT fuse, and a fuse cap are packaged together in an anti-static bag and placed on top of the instrument. Packaged with each fuse is a tag identifying the size of the fuse, its part number, and the line voltage for which it is used.

The line fuse holder (FUSE) is located on the rear panel of the mainframe's power module (Figure 3-1). For 90 VAC to 132 VAC operation, the 3.0 AT fuse must be installed. For 198 VAC to 250 VAC operation, the 1.5 AT fuse must be installed.

To insert the fuse, place one end of the fuse into the fuse cap. Insert into the fuse holder and press the cap down until it is even with the rim. Rotate the cap clockwise to lock the fuse in place.

To remove a fuse, use your finger or a small flat blade screwdriver and depress the fuse cap while rotating it counterclockwise. Once removed, the fuse is easily separated from the fuse cap. Replacement fuses and caps are available from Hewlett-Packard for the part numbers shown on the tag or in Figure 3-2.

www.valuetronics.com

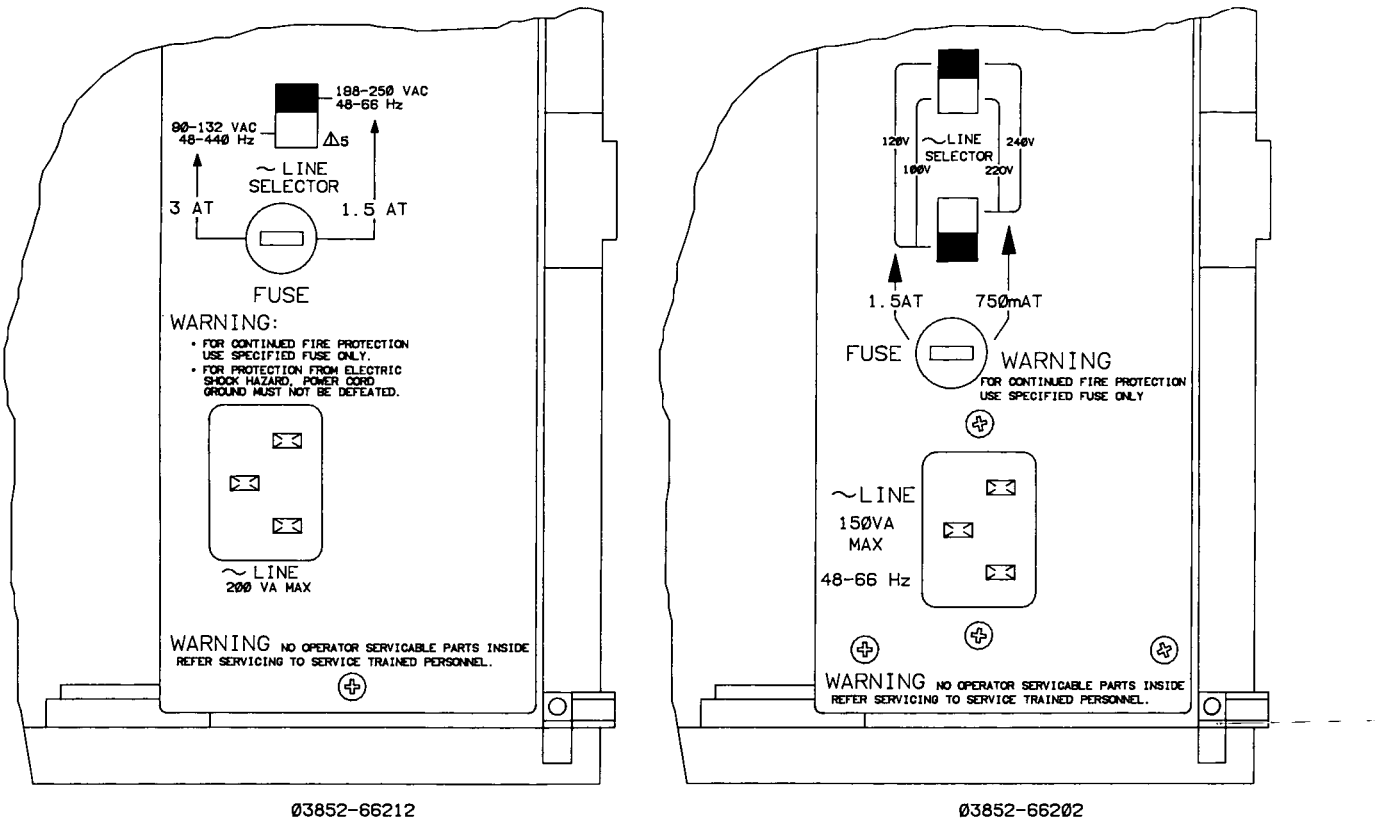


Figure 3-1. FUSE Holder and LINE SELECTOR Switch Location

POWER SUPPLY	LINE VOLTAGE	LINE FUSE	FUSE CAP
03852-66212	90 to 132 VAC (nominal) 198 to 250 VAC (nominal)	3.0 AT - HP P/N 2110-0003 1.5 AT - HP P/N 2110-0043	Gray, HP P/N 2110-0565
03852-66202	100 to 120 VAC (nominal) 220 to 240 VAC (nominal)	1.5 AT - HP P/N 2110-0304 750 mAT - HP P/N 2110-0346	Gray, HP P/N 2110-0565

Figure 3-2. Replacement Line Fuses and Fuse Cap

www.valuetronics.com

Setting the LINE SELECTOR Switch

The LINE SELECTOR switch must be set for the line voltage in your area. The switch is located above the FUSE holder on the rear panel of the mainframe's power module (Figure 3-1). To set the switch, use a small flatblade screwdriver and position the switch as shown in Figure 3-3 for the line voltage present.

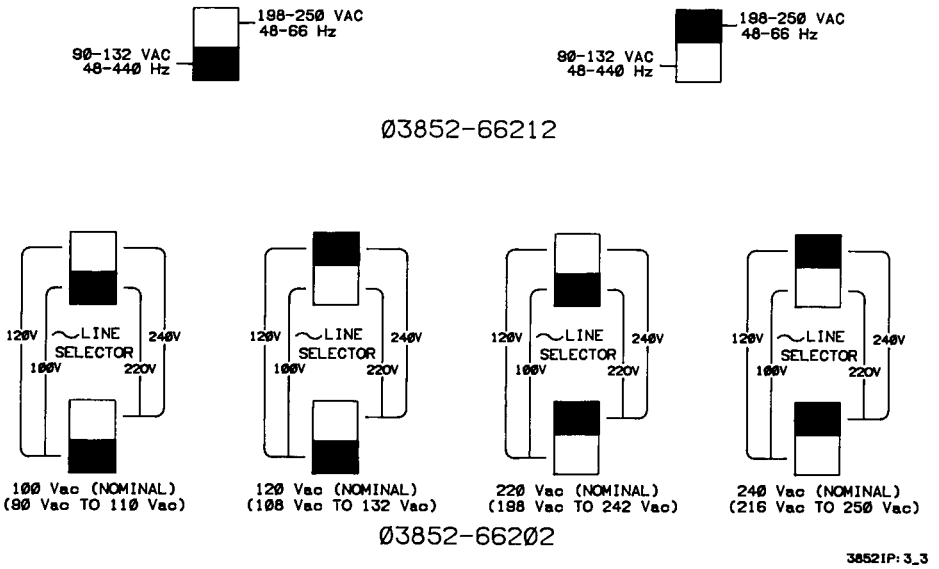


Figure 3-3. LINE SELECTOR Switch Positions

Power Cord and Grounding Requirements

The mainframe is shipped with a three conductor AC power cord which, when plugged into an approved three-contact electrical outlet, grounds the instrument. The type of power cord shipped with the instrument depends on the country of destination. Figure 3-4 shows the power cords that are available. The mainframe's power jack and the supplied power cable meet International Electrotechnical Commission (IEC) safety standards. If the appropriate power cord was not received, notify the Order Administration Department of your nearest HP Sales and Service Office.

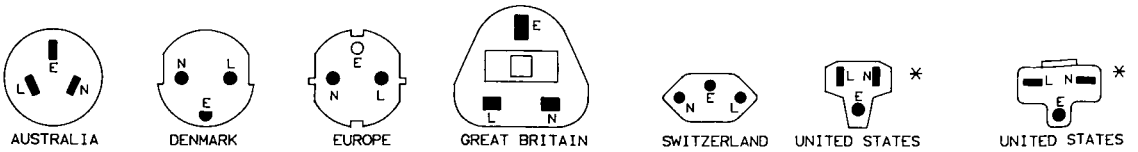
WARNING

If a replacement power cord is needed, be sure to order a Hewlett-Packard power cord that is identical to the original. Otherwise, electrical shock or equipment damage may result.

---

www.valuetronics.com

POWER CORDS



Country	Part Number	Opt.	Voltage
Australia	8120-1369	901	250V 6A
Denmark	8120-2956	912	250V 6A
Europe	8120-1689	902	250V 6A
Great Britain	8120-1351	900	250V 6A
Switzerland	8120-2104	906	250V 6A
*United States	8120-1378	903	120V 10A
*United States	8120-0698	904	240V 10A

Power cords supplied by HP have polarities matched to the power input socket on the instrument:

- L = Line or Active Conductor (also called "live" or "hot").
- N = Neutral or Identified Conductor
- E = Earth or Safety Ground

NOTE: Plugs are viewed from connector end. Shape of molded plug may vary within country.

* CSA certification includes only these Power Plugs

Figure 3-4. Power Cords for the HP 3852A

Positioning the Mainframe

The mainframe can be used as either a bench or rack mounted instrument. This section covers placement of the mainframe in both environments.

Bench Operation

If used as a bench instrument, choose a location that provides at least four inches (100 mm) of clearance at the back of the instrument and one inch (25 mm) of clearance at the front and on each side. The fan located behind the front panel grill draws air through the grill and directs it between the card cage and the bottom cover to cool the instrument. When adequate clearance is not provided, excessive temperatures can be generated inside the mainframe thus reducing mainframe and accessory reliability. The four inches of clearance at the back of the instrument accommodate any cables or wires that are connected.

By extending the tilt stands on the front legs, the front of the HP 3852A can be elevated.

Rack Mounting

Options 908 and 909 (Rack Mounting Kits) enable the mainframe to be mounted in a 30'' or 56'' high x 19'' wide EIA rack. Installation instructions are included with the rack mount kit ordered. For further information on HP rack mounting kits, order the HP SYSTEM II Rack Mounting Kits & Accessories brochure (part number 5952-0095). This brochure is available from your nearest HP Sales and Service Office.

www.valuetronics.com

NOTE

To avoid interference effects between HP 3852A systems and their installation environment, close attention should be paid to electromagnetic compatibility. Wiring should follow good RF practice, such as ensuring that all interconnections are enclosed by shields connected or bypassed to local earth grounds via low impedance paths.

NOTE

If the mainframe is to be installed in an HP 44742A Industrial Cabinet, refer to the 44742-90000 manual for installation procedures.

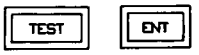
Applying Power

Once the line fuse has been installed and the LINE SELECTOR switch has been set, the mainframe can be turned on. Begin by plugging the power cord into the electrical outlet. Ensure that the front panel LINE switch is in the "0" position then plug the power cord into the mainframe's power jack. Turn the HP 3852A on by depressing the LINE switch. After the switch is is depressed, note the following:

- a "beep" is sounded and the fan begins to run.
- the power-on sequence of Figure 3-5 occurs

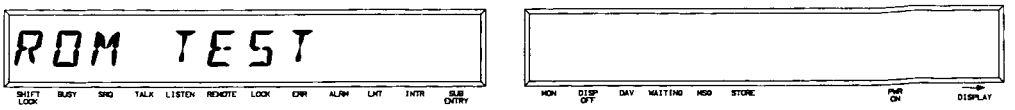
Self Test

The self test which occurs at power-on provides a 90% confidence level that operation of the HP 3852A (mainframe only) is within its specifications if no power-on failures or error messages were displayed. Another self test of the mainframe can be made by sending the TEST command or by pressing:

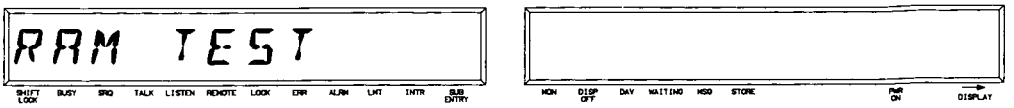


on the front panel. The self test activated by the TEST command (or key), provides a 60% confidence level that the mainframe is operating within specification. This self test is useful when the mainframe is operational, as none of the instrument's currently programmed operating states are affected. Recall that the power-on self test resets all mainframe operating parameters to their power-on conditions. The self test activated by the TEST command has the display sequence shown in Figure 3-6.

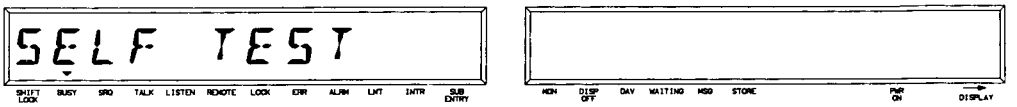
www.valuetronics.com



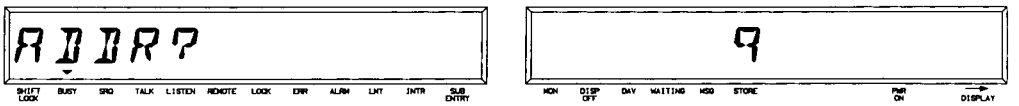
Tests the mainframe's read only memory



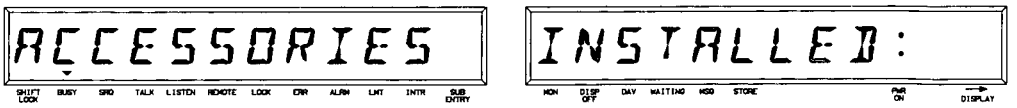
Tests the mainframe's read/write memory



Tests the mainframe's controller module and front panel

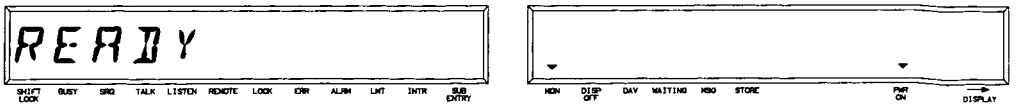


Displays the mainframe's HP-IB address. (The factory set address is 9.)



Indicates the number of accessories installed in the mainframe and extender (s)

Indicates whether or not an extended memory board is installed.

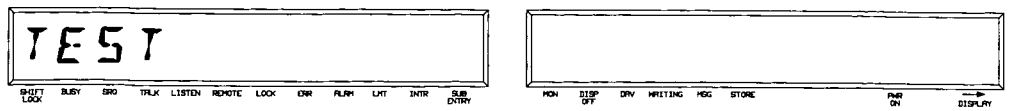


3852650 2_3

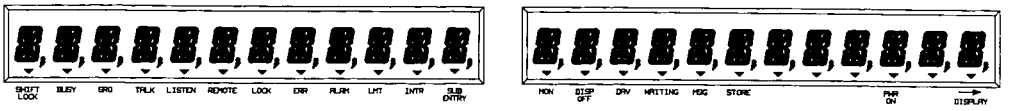
The mainframe completes its power-on sequence and is "READY" to accept commands.

Figure 3-5. HP 3852A Power-on Sequence

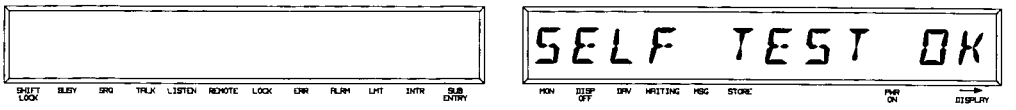
www.valuetronics.com



The TEST command is entered into the mainframe and the self test begins.



The Display and "Beeper" are tested as part of the self test.



Indicates that the self test passed.

Figure 3-6. HP 3852A Self Test Sequence

Power-on and Self Test Failures

If the mainframe failed to turn on as indicated, verify the following:

1. Check that the front panel LINE switch is in the "1" position.
2. Check that the power cord is firmly into the instrument's power jack and into the electrical outlet.
3. Check that power is present at the outlet.
4. Check the line fuse and replace it if necessary. Refer to the **AC Line Power Requirements** section for procedures.
5. Try to turn the instrument on again. If it still fails to turn on, contact the Service Department of your nearest Hewlett-Packard Sales and Service Office.

Power-on and Self Test Error Codes

The following error codes are associated with error conditions that occur during power-on or during a self test.

ERROR 80: If the mainframe displayed "ERROR 80:" followed by a number code when you turned the instrument on, an internal power-on test failed. At this point, the instrument locks up as it cannot function with any degree of confidence. Cycle the power again. If the message is displayed again, record the number code and refer to the HP 3852A Assembly Level Service Manual (HP Part Number 03852-90011) for code definitions. If necessary, contact the Service Department of the nearest Hewlett-Packard Sales and Service Office.

www.valuetronics.com

ERROR 26: If the mainframe displayed "ERROR 26: POWER ON TEST FAILURE" during its power-on sequence, an internal power-on test failed. Under this condition, however, you still have control of the instrument. Cycle the the power again. After the message is displayed and the power-on sequence completes, the ERR annunciator in the display will be on indicating that the error message is stored in a buffer. To retrieve the message, press:

To view the entire message and the number code, it will be necessary to scroll the display to the left by pressing:

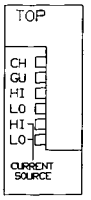
Record the number code and refer to the Service Manual for code definitions. If necessary, contact the Service Department of the nearest Hewlett-Packard Sales and Service Office before attempting to further operate the instrument.

ERROR 27: If the mainframe displayed "ERROR 27: SELF TEST FAILED" when you initiated a self test with the TEST command, send the command again. If the self test still fails, scroll the display to the left by pressing:

and record the number code. Refer to the Service Manual for code definitions. Note that ERROR 27 may occur if an accessory was installed while the mainframe was turned on. The HP 44701A voltmeter may fail if the voltmeter was busy when the TEST command was issued. If either was the case, cycle power and run the test again. If necessary, contact the nearest Hewlett-Packard Sales and Service Office.

WARNING

When using the HP 3852A mainframe with or without an extender, always ensure that the analog extender connector is securely fastened over the extender port.



The HP 3852A and certain plug-in accessories allow up to 350V peak to be applied to the backplane analog bus. Voltages present on the bus are also present on the port, which is covered when the connector is securely installed. These voltages also appear on the connector terminals and terminal screws. These parts are recessed for operator safety and must be accessed only by service-trained personnel **after all field wiring power has been disconnected.**

---

www.valuetronics.com

Mainframe State and Identity

Two commands are available to help you determine the internal configuration of your HP 3852A. The commands are STATE? and IDN? and they have the syntax as shown below:

STATE? [INTO *name*] or [*fmt*]
IDN?

STATE? identifies the presence of extended memory boards, the controller module installed, and the power line frequency. STATE? returns two numbers. The first number is always 1 followed by the sum of the features listed as follows:

Value	Features
1	256 kbyte extended memory board installed
4	1 Mbyte extended memory board installed
8	2 Mbyte extended memory board installed
16	4 Mbyte extended memory board installed
64	Controller module installed is the 03852-66523
128	Power line frequence = 60 Hz

The following program shows how the STATE? command is used.

```
10 INTEGER State(0:1)
20 OUTPUT 709;"STATE?"
30 ENTER 709;State(*)
40 PRINT State(*)
50 END
```

If, for example, the line frequency is 60 Hz (128), the 03852-66523 controller module is installed (64), and a 1 Mbyte extended memory board (4) is installed, the following data is returned:

```
1    196
```

The INTO name and fmt parameters indicate the data returned by STATE? can be stored in mainframe memory or entered/displayed in a particular format.

IDN? is used to determine the firmware revision in the HP 3852A mainframe. When executed, IDN? returns:

HEWLETT PACKARD	(Company name)
3852A	(Model number)
0	(Serial number - unknown)
3.52	(Firmware revision)

When IDN? is executed from the front panel, only the firmware revision (3.52) will be seen unless the display mode is changed from fast to slow with the FASTDISP OFF command. The following program, however, can be used to enter and display the data returned by IDN? on a controller.

```
10 DIM Identity$(0:3) [17]
20 OUTPUT 709;"IDN?"
30 ENTER 709;Identity$(*)
40 PRINT USING "K,/";Identity$(*)
50 END
```


www.valuetronics.com

Installing the HP 3853A Extender

Installation of the extenders includes the following:

- AC Line Power Requirements
- Power Cord and Grounding Requirements
- Positioning the Extender
- Connecting the Extender to the Mainframe
- Setting the Extender Number
- Applying Power
- Verifying Extender Installation

AC Line Power Requirements

The extender can be set to operate at line voltages of 90 to 132, or 198 to 250 VAC (all values rms), 48 to 440 Hz, or 48 to 66 Hz single phase. The power line voltage can vary by ±10% but cannot exceed 250 VAC rms.

CAUTION

Using the wrong fuse value and type for the line voltage present can result in damage to the circuitry inside the instrument.

Connecting the mainframe to the line voltage when the LINE SELECTOR switch is improperly set may blow the fuse.

Installing the Line Fuse

When shipped from the factory a 1.5 AT (Amperes/time-lag type) fuse, a 3.0 AT fuse, and a fuse cap are packaged together in an anti-static bag and placed on top of the instrument. Packaged with each fuse is a tag identifying the size of the fuse, its part number, and the line voltage for which it is used.

The line fuse holder (FUSE) is located on the rear panel of the extender's power module (Figure 3-7). For 90 VAC to 132 VAC operation, the 3.0 AT fuse must be installed. For 198 VAC to 250 VAC operation, the 1.5 AT fuse must be installed.

www.valuetronics.com

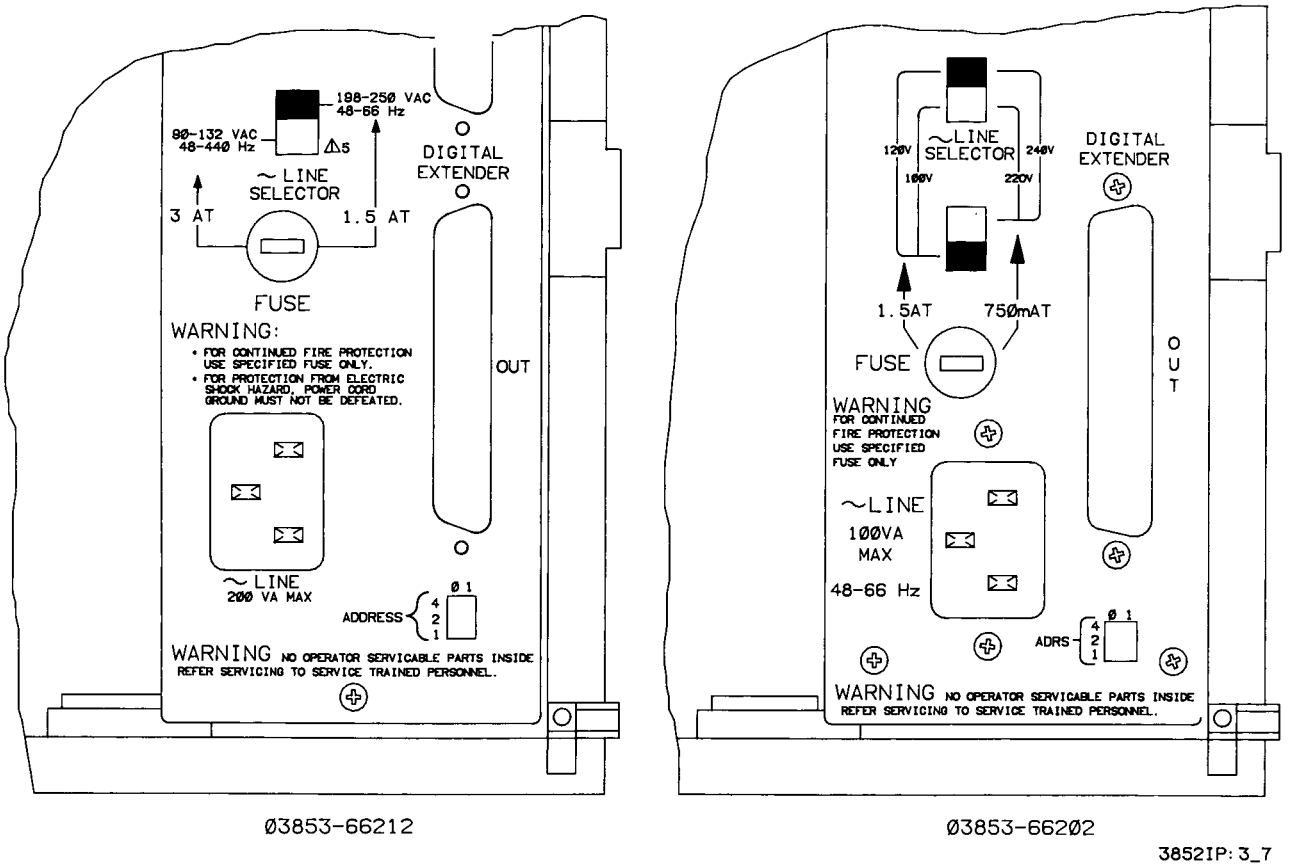


Figure 3-7. FUSE Holder and LINE SELECTOR Switch Locations

To insert the fuse, place one end of the fuse into the fuse cap. Insert into the fuse holder and press the cap down until it is even with the rim. Rotate the cap clockwise to lock the fuse in place.

To remove a fuse, use your finger or a small flatblade screwdriver and depress the fuse cap while rotating it counter clockwise. Once removed, the fuse is easily separated from the fuse cap. Replacement fuses and caps are available from Hewlett-Packard for the part numbers shown on the tag or in Figure 3-8.

www.valuetronics.com

POWER SUPPLY	LINE VOLTAGE	LINE FUSE	FUSE CAP
03852-66212	90 to 132 VAC (nominal) 198 to 250 VAC (nominal)	3.0 AT - HP P/N 2110-0003 1.5 AT - HP P/N 2110-0043	Gray, HP P/N 2110-0565
03852-66202	100 to 120 VAC (nominal) 220 to 240 VAC (nominal)	1.5 AT - HP P/N 2110-0304 750 mAT - HP P/N 2110-0346	Gray, HP P/N 2110-0565

Figure 3-8. Replacement Line Fuses and Fuse Cap

Setting the LINE SELECTOR Switch

The LINE SELECTOR switch must be set for the line voltage in your area. The switch is located above the FUSE holder on the rear panel of the extender's power module (Figure 3-7). To set the switch, use a small flatblade screwdriver and position the switch as shown in Figure 3-9 for the line voltage present.

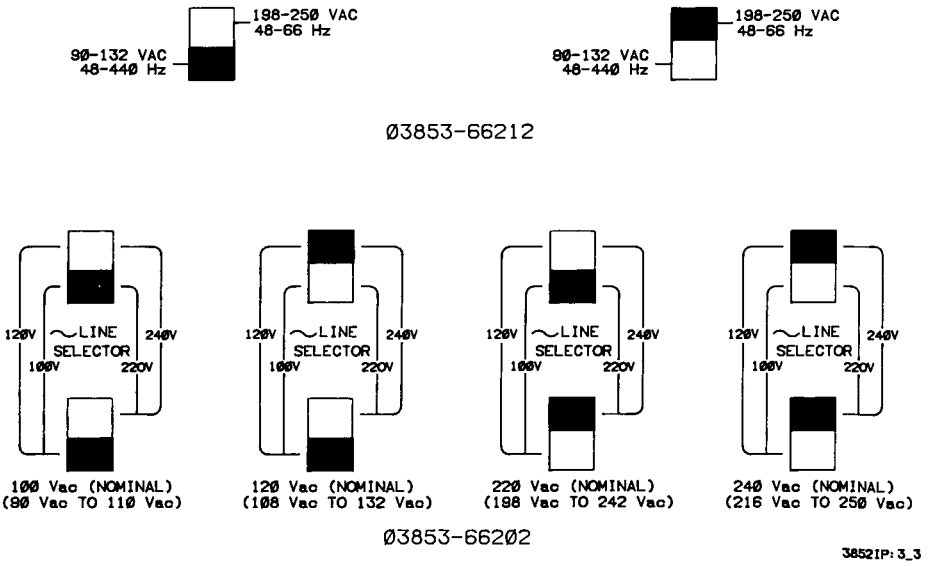


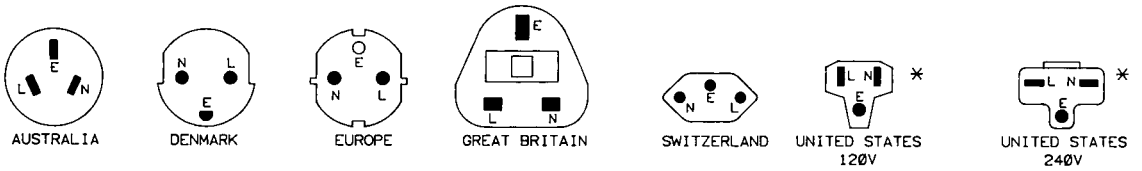
Figure 3-9. LINE SELECTOR Switch Positions

www.valuetronics.com

Power Cord and Grounding Requirements

Extenders are shipped with a three conductor AC power cord which, when plugged into an approved three-contact electrical outlet, grounds the instrument. The type of power cord shipped with the extender depends on the country of destination. Figure 3-10 shows the power cords that are available. The extender's power jack and the supplied power cable meet International Electrotechnical Commission (IEC) safety standards. If the appropriate power cord was not received, notify the Order Administration Department of your nearest HP Sales and Service Office.

POWER CORDS



Country	Part Number	Opt.	Voltage
Australia	8120-1369	901	250V 6A
Denmark	8120-2956	912	250V 6A
Europe	8120-1689	902	250V 6A
Great Britain	8120-1351	900	250V 6A
Switzerland	8120-2104	906	250V 6A
*United States	8120-1378	903	120V 10A
*United States	8120-0698	904	240V 10A

Power cords supplied by HP have polarities matched to the power input socket on the instrument:

- L = Line or Active Conductor (also called "live" or "hot").
- N = Neutral or Identified Conductor
- E = Earth or Safety Ground

NOTE: Plugs are viewed from connector end. Shape of molded plug may vary within country.

* CSA certification includes only these Power Plugs

Figure 3-10. Power Cords for the HP 3853A

WARNING

If a replacement power cord is needed, be sure to order a Hewlett-Packard power cord that is identical to the original. Otherwise, electrical shock or equipment damage may result.

---

www.valuetronics.com

Positioning the Extender

The extender can be used as either a bench or rack mounted instrument. This section covers placement of extenders in both environments.

Bench Operation

If used on the bench, choose a location that provides at least four inches (100 mm) of clearance at the back of the instrument and one inch (25 mm) of clearance at the front and on each side. The fan located behind the front panel grill draws air through the grill and directs it between the card cage and the bottom cover to cool the instrument. When adequate clearance is not provided, excessive temperatures can be generated inside the extender thus reducing equipment reliability. The four inches of clearance at the back of the instrument accomodate any cables or wires that are connected.

Rack Mounting

Options 908 and 909 (Rack Mounting Kits) enable extenders to be mounted in 30" or 56" high x 19" wide EIA racks. Installation instructions are included with the rack mount kit ordered. For further information on HP rack mounting kits, order the HP SYSTEM II Rack Mounting Kits & Accessories brochure (part number 5952-0095). This brochure is available from your nearest HP Sales and Service Office.

NOTE

To avoid interference effects between HP 3852A systems and their installation environment, close attention should be paid to electromagnetic compatibility. Wiring should follow good RF practice, such as ensuring that all interconnections are enclosed by shields connected or bypassed to local earth grounds via low impedance paths.

NOTE

If the extender is to be installed in an HP 44742A Industrial Cabinet, refer to the 44742-90000 manual for installation procedures.

---

www.valuetronics.com

Distance Between the Mainframe and Extenders

Data and Control information flow between mainframe and extenders via digital extender and analog extender cables. To insure the integrity of information across the digital extender cable, restrictions are placed on its length and thus, on the distance an extender can be separated from the mainframe. The total digital extender cable length between a mainframe and its extenders is 28 metres. Seven extenders for example, could be linked together by digital extender cables each a different length, provided the total cumulative length of the cables does not exceed 28 metres. This restriction applies whether the extenders are part of a bench or rack mounted system (see Figure 3-11).

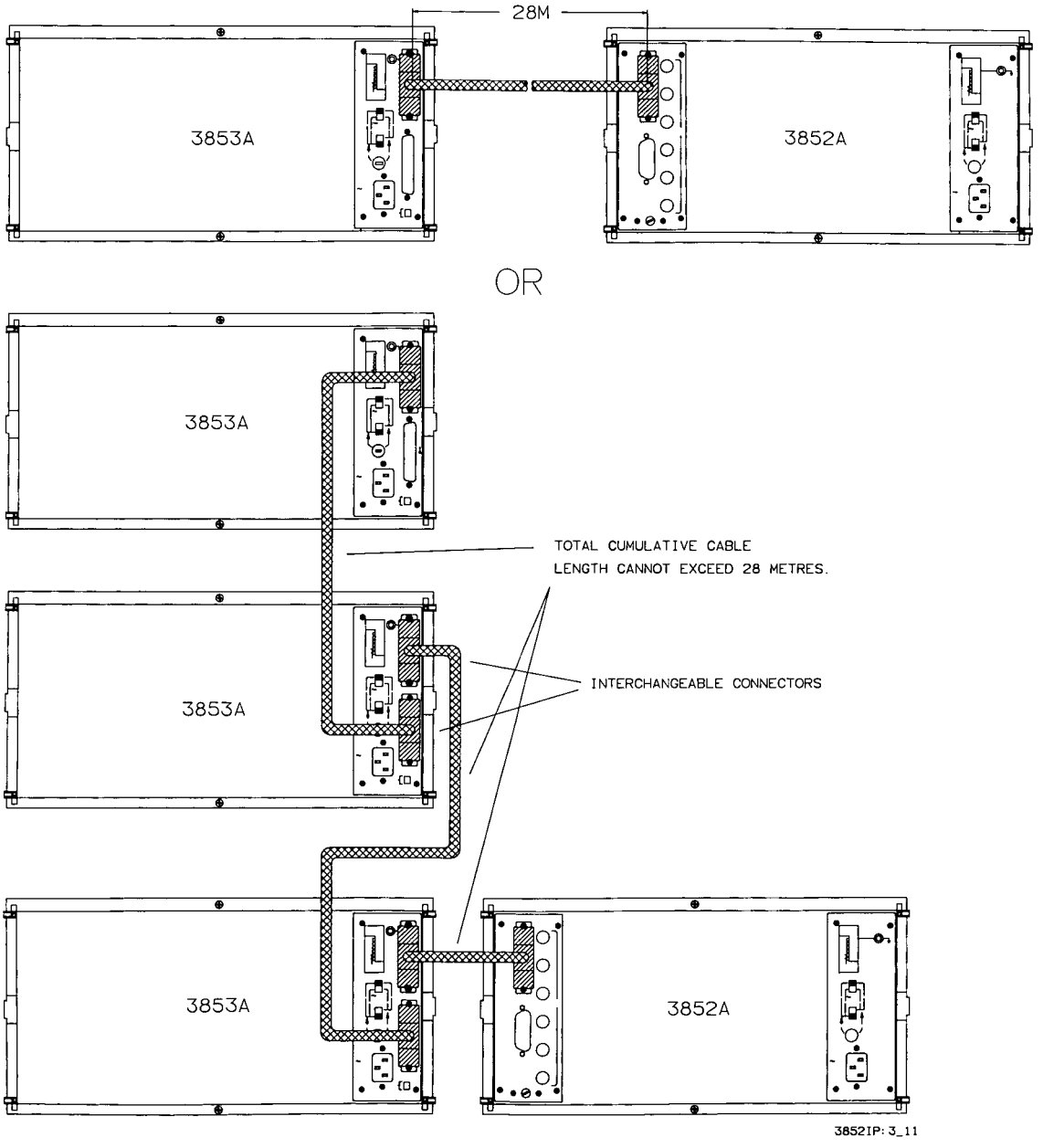


Figure 3-11. Cable Length Restrictions

www.valuetronics.com

The standard digital extender cable shipped with the extender is one metre. If your configuration requires a digital extender cable of a different length, contact your nearest HP Sales and Service Office regarding availability.

Two analog extender cables are shipped with the extender and each is one metre. Other lengths are available through Hewlett-Packard. Note that the total length of the analog extender cables should not exceed the total length of the digital extender cables.

Connecting the Extender to the Mainframe

This section explains how (and when) the digital and analog extender cables are connected to the mainframe.

The Digital Extender Cable

The extender's primary link to the mainframe is through the digital extender cable. The cable is a bi-directional interface that transfers data and control information between the mainframe and the extender(s)/accessories.

To connect the extender to the mainframe, first ensure that the mainframe and extender are turned off. Next, connect one end of the cable to the mainframe port labeled "Digital Extender Out". Connect the other end of the cable to the "IN" port on the extender. Tighten all four thumbscrews to secure the cable in place.

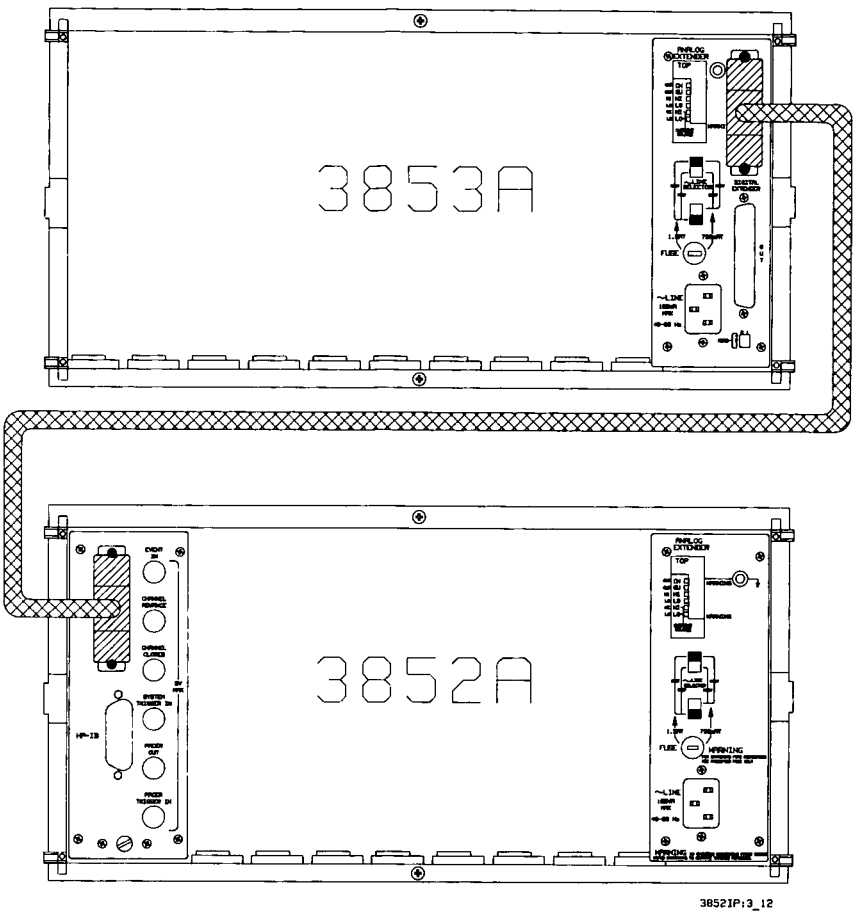
NOTE

The "IN" and "OUT" ports are interchangeable. Two ports are provided on each extender to link multiple extenders to the mainframe.

Connecting Multiple Extenders

Multiple extenders are added by connecting a digital extender cable from the unused port ("OUT") on the extender connected to the mainframe, to the "IN" port on the second extender. A third extender is added by connecting the cable between the "OUT" port on the second extender and the "IN" port on the third extender (Figure 3-13).

www.valuetronics.com



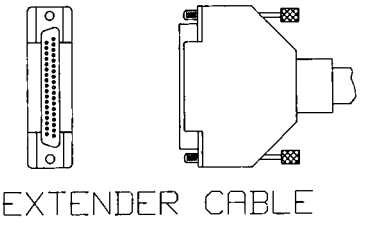


Figure 3-12. Connecting the Digital Extender Cable

www.valuetronics.com

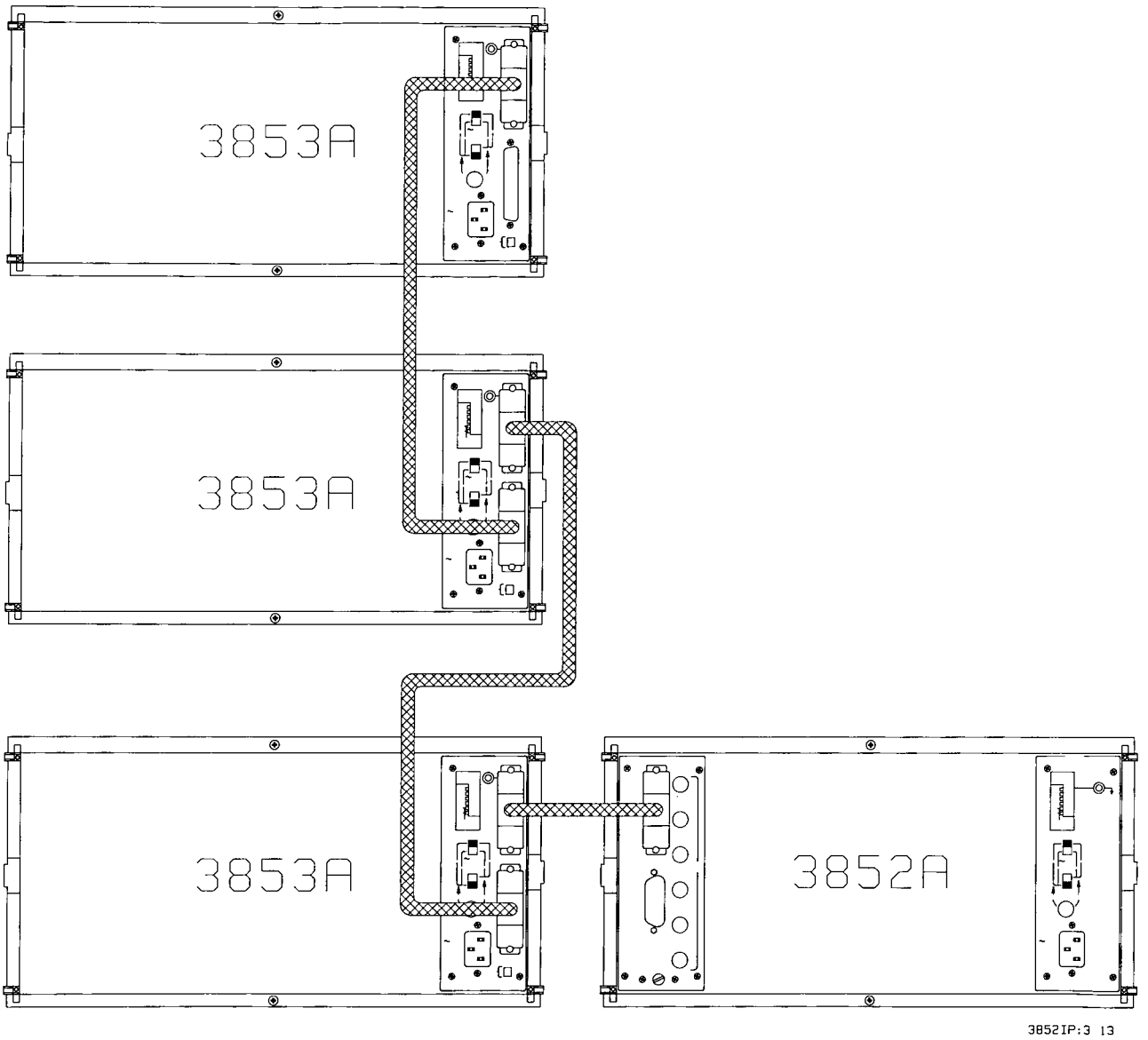


Figure 3-13 Connecting Multiple Extenders

www.valuetronics.com

When to Connect the Analog Extender Cable

The analog extender cable links the mainframe's analog bus to the extender's analog bus. This backplane bus routes analog input signals from the multiplexer to the voltmeter accessories (Figure 3-14).

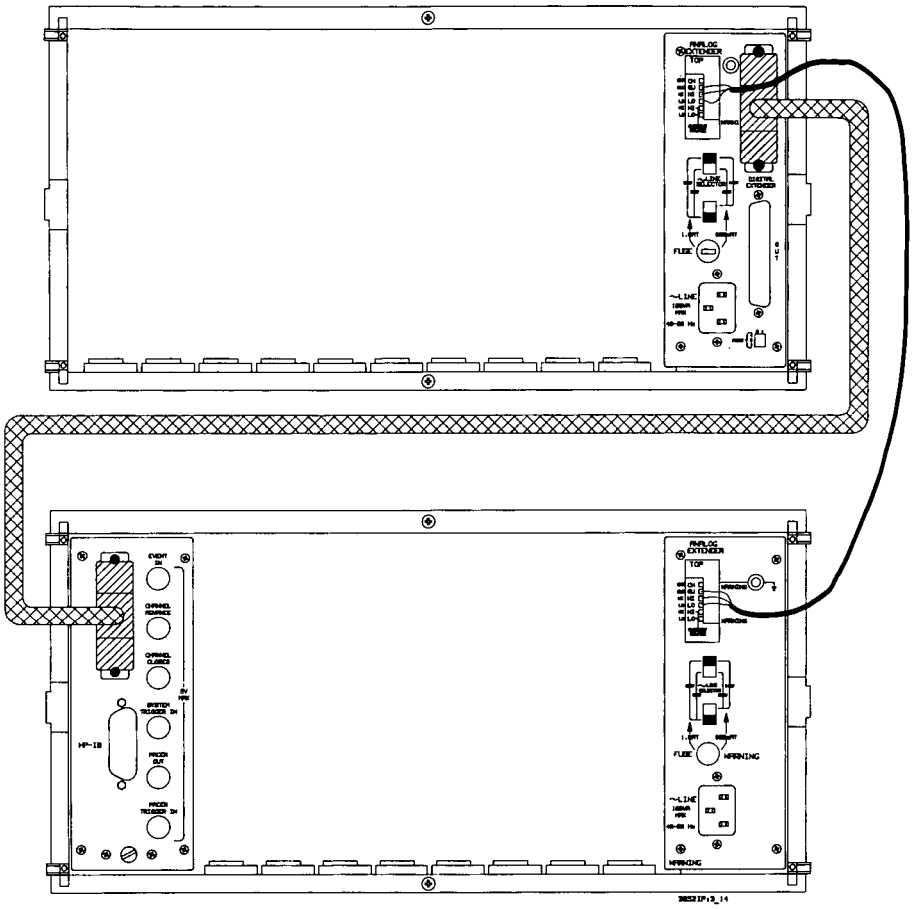


Figure 3-14. The Backplane Analog Bus

Unlike the digital extender cable, the analog extender cable(s) need only be connected between mainframe and extender or between extenders when measurements involve a voltmeter and multiplexer accessory that are not installed together in the same instrument (Figure 3-15).

www.valuetronics.com

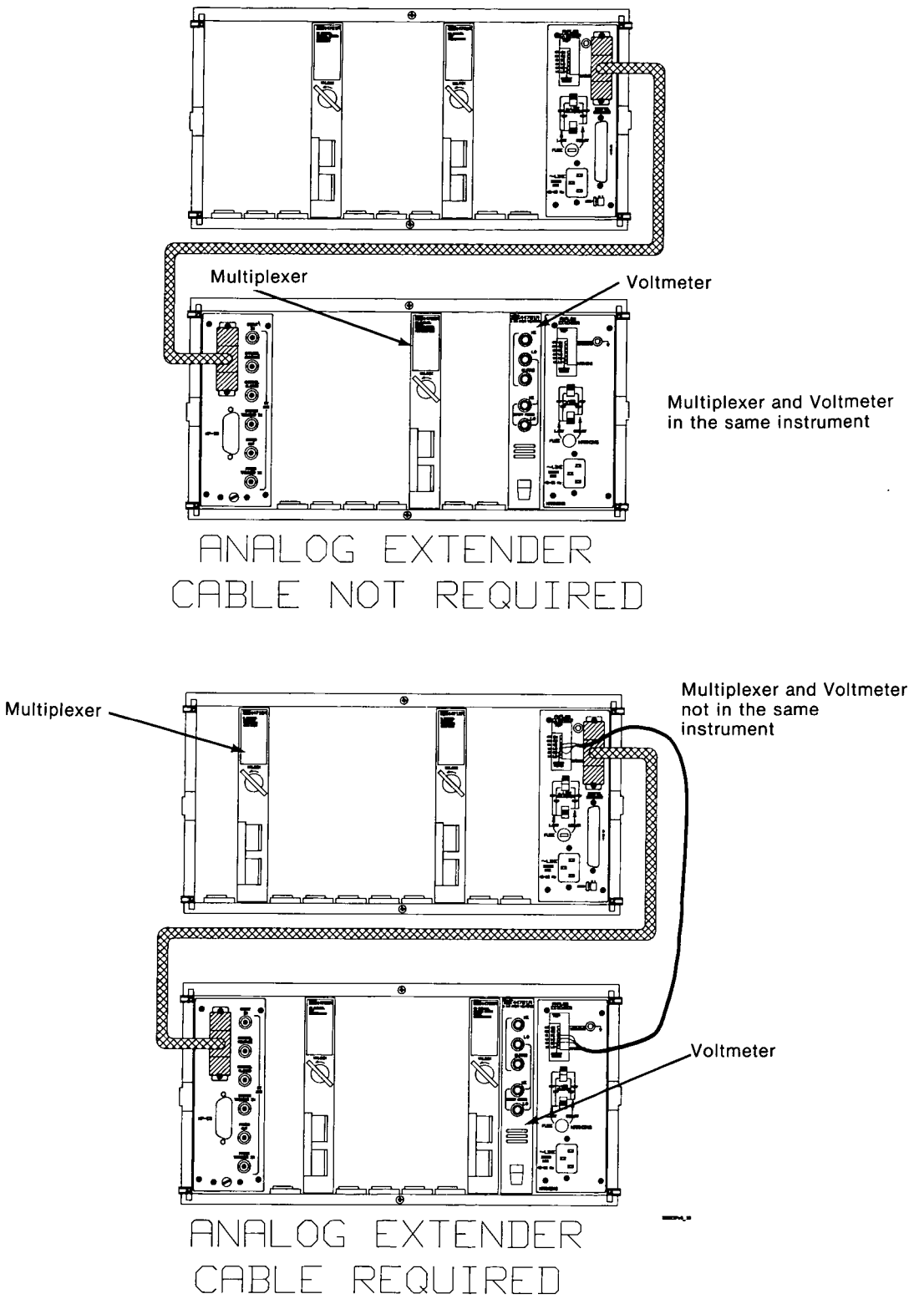


Figure 3-15. When to Connect the Analog Extender Cable

www.valuetronics.com

WARNING

The HP 3852A and certain plug-in accessories allow up to 350 V peak to be applied to the backplane analog bus. Voltages present on the bus are also present on the terminals and screws of the analog extender connector. These parts are recessed for operator safety and must be accessed by service-trained personnel only after all field wiring power has been disconnected.

NOTE

Consider installing those multiplexer and voltmeter accessories rated for higher voltages together in the mainframe or same extender. Install those multiplexer and voltmeter accessories rated for lower voltages in a second instrument. In this configuration, you would not have to connect the analog bus between the mainframe and extender when otherwise, you risk damage to all of your equipment in case of equipment failure or programming error.

To connect the analog extender cable(s):

1. Turn the mainframe and extender power off and disconnect all field wiring power to the analog bus or accessories.
2. Loosen the screws of the connector terminals where the cable wires are to be connected (Figure 3-16). Table 3-2 lists the terminal connections for common measurements.
3. Insert the cable wire into the terminal until only the insulation is visible (Figure 3-17). Tighten the screw to secure the wire in place.

www.valuetronics.com

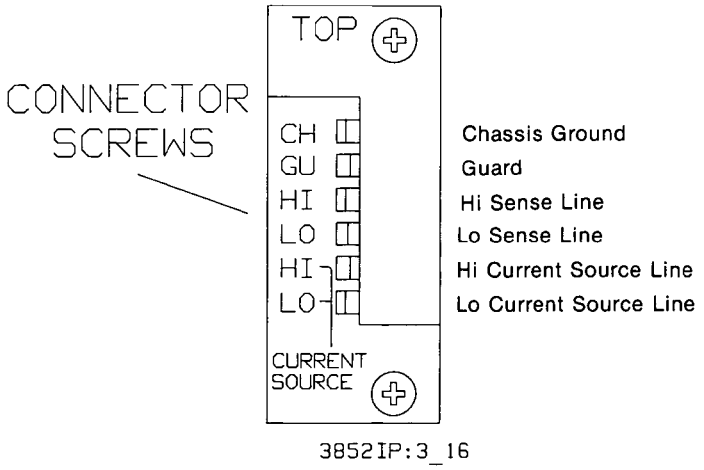


Figure 3-16. Analog Extender Connector Terminals

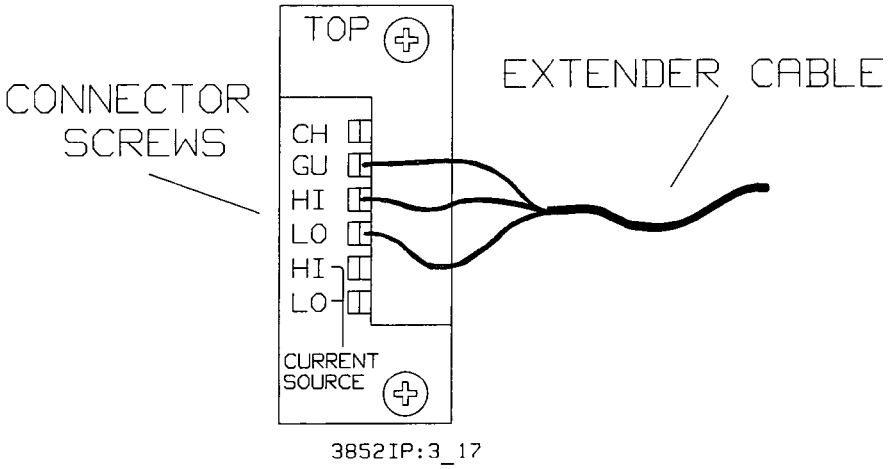


Figure 3-17. Connecting the Analog Extender Cable

WARNING

⚠ 1 *Due to the potential allowed on the backplane bus (up to 350 V peak), ensure that none of the analog extender wires becomes frayed or is exposed when you connect them.*

---

www.valuetronics.com

WARNING

When connecting the other end of the cable, ensure that the wires are inserted into the corresponding connector terminals.

Measurement Connections

The analog extender terminals of the mainframe and extenders to which the analog cable is connected depends on the measurement. Table 3-2 lists the terminal connections for measurements involving the voltmeter and multiplexer accessories. See the appropriate plug-in accessory manuals for wiring and programming information on the accessories and measurement involved.

Table 3-2. Measurement Connections

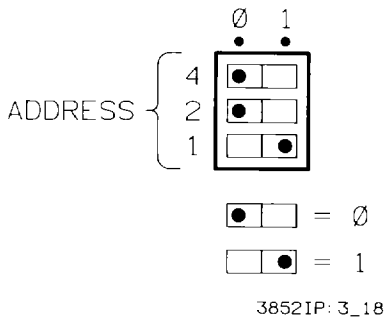
Measurement	Terminal Connections
ACV	HI, LO, GU
DCV	HI, LO, GU
OHMS: 2-wire (2 cables)	HI, LO, GU, HI LO - Current
OHMS: 4-wire (2 cables)	HI, LO, GU, HI LO - Current
TEMPERATURE -Thermocouple -Thermistor -RTD	HI, LO, GU, HI LO - Current
STRAIN	HI, LO, GU

Setting the Extender Number

Up to seven extenders can be used with a single mainframe. To distinguish between each extender and the mainframe during operation, the extender is assigned an extender number. Extender numbers are from one to seven (maximum seven extenders per mainframe), and an extender can be assigned any number regardless of how many extenders are part of the configuration. When multiple extenders are used, each extender number must be unique.

Extenders are shipped from the factory with an extender number of "1". The extender number is set by a column of switches located near the power socket on the back of the instrument. To change the extender number, use a small flatblade screwdriver or similar tool and set the switches as indicated in Figure 3-18 for the extender number desired.

www.valuetronics.com



Extender # (decimal)	SWITCH 4	SWITCH 2	SWITCH 1
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Figure 3-18. Setting the Extender Number

NOTE

An extender number of "0" (switches 4,2,1 in the 0 position) is the "extender number" of the mainframe. If an extender is set to 0, that extender becomes unuseable.

If two or more extenders have the same extender number, unpredictable and erroneous readings will result when the extender number is addressed.

Applying Power

Once the appropriate line fuse has been installed and the LINE SELECTOR switches have been set, the instruments can be turned on. Ensure that the extender cables are secure, then plug the power cord into the extender's power jack. Turn the extender(s) on by depressing the front panel LINE switch. When extenders are used, the extenders must be turned on in addition to the mainframe in order for the mainframe to complete its power-on sequence.

www.valuetronics.com

Turn the mainframe on by depressing its front panel LINE switch. The mainframe will then start its power-on sequence as described in Figure 3-5.

Power-on Failures

If the extender(s) and mainframe failed to turn on as indicated, verify the following:

1. Check that the front panel LINE switches are in the "1" position.
2. Check that the power cords are firmly into the instruments' power jack and electrical outlet.
3. Check that power is present at the outlet.
4. Check the extender and mainframe line fuses and replace if necessary. Refer to the **AC Line Power Requirements** section for procedures.

If the mainframe did not perform its power-on sequence but displayed: ERROR 38: CHECK POWER, an extender has not been turned on. Turning the extender on will then enable the mainframe to complete the power-on sequence. (If the extenders are turned on, ERROR 38 may indicate equipment failure. See the HP 3852A Assembly Level Service Manual.)

If the mainframe displayed an error code at power-on, refer to **Power-on and Self Test Error Codes** for references to further information.

NOTE

Turning off any extender while the mainframe is turned on will cause the mainframe to lock up, pause a few seconds, then display: ERROR 38: CHECK POWER. To restore control of the mainframe, the extender must be turned back on. This resets the mainframe to its power-on conditions thus erasing any pre-programmed states.

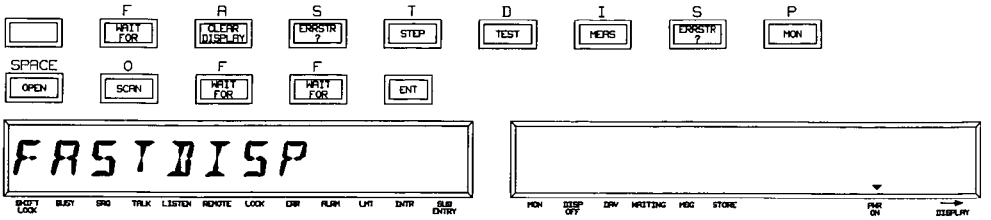
If an extender is connected to the mainframe while power is applied to both instruments, the extender will not be recognized by the mainframe unless the mainframe is reset or the power is cycled on the extender.

---

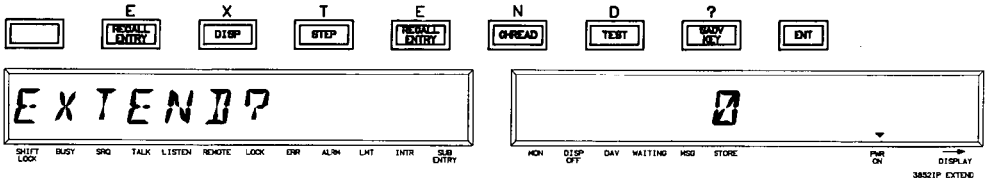
www.valuetronics.com

Verifying Extender Installation

The EXTEND? command can be used to verify that the digital extender cable is securely fastened to mainframe and extender(s) and that the extender number is set as intended. When executed, the EXTEND? command returns seven consecutive numbers. If no extenders are connected, seven 0's are returned. If an extender with an extender number of 5 is connected to the mainframe: 0,0,0,0,5,0,0 would be returned one number at a time. In order to view extender numbers returned, the fast display mode of the mainframe must be off before EXTEND? is executed. To turn the fast display mode off, press:



Execute the EXTEND? command by pressing:



It may be necessary to execute the command again in order to note the extender numbers displayed. Consecutive 0's will appear as though only one number is returned.

Sending the following program over the HP-IB will enable you to view the extender numbers on a controller CRT regardless of whether FASTDISP is on or off:

```
10 INTEGER Ext(0:6)
20 OUTPUT 709;"EXTEND?"
30 ENTER 709;Ext(*)
40 PRINT Ext(*)
50 END
```

Installing the Plug-in Accessories

Plug-in accessory installation includes the following:

- Separating the Modules
- Removing the Terminal Module Cover
- Installing the Accessories
- Accessory Power Consumption
- Verifying Accessory Installation
- Installation Hints
- Installing the Extended Memory Cards

www.valuetronics.com

NOTE

Accessory installation as covered in this manual refers to the physical placement of the accessories in the mainframe/extender. Accessory configuration and field wiring are covered in the individual accessory configuration and programming manuals.

Separating the Modules

Many of the HP 3852A plug-in accessories consist of a terminal module and a component module (Figure 3-19). When shipped, the terminal module is connected to the component module.

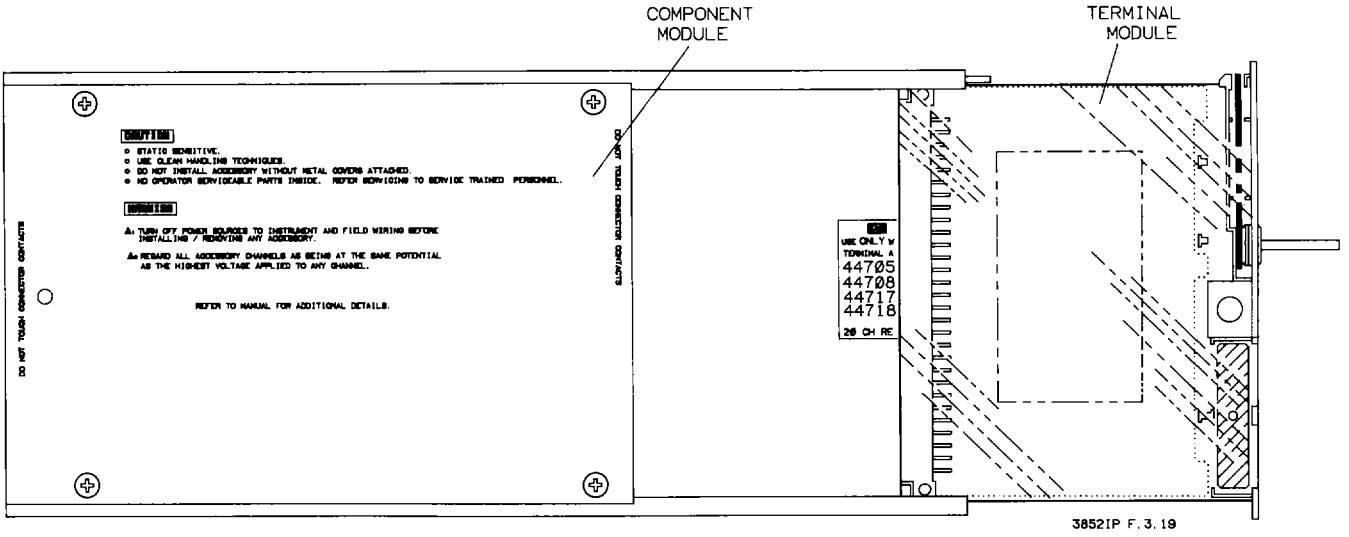


Figure 3-19. Terminal Module and Component Module

CAUTION

When handling the plug-in accessories, avoid touching the edge connector contacts. Touching the contacts can subject the components to static discharges that could damage them. Accessories can be held at any point provided you do not bump or touch the edge connectors.

www.valuetronics.com

To separate the modules, grasp the accessory by the component module. Ensure that the locking ring on the terminal module is in the "UNLOCK" position. While holding the component module firmly, separate the modules by pulling on the ring.

Removing the Terminal Module Cover

The terminal module cover can only be removed when the module is separated from the component module. To remove the cover, use a flatblade screwdriver and rotate the spring-loaded fastener counterclockwise one quarter turn, then raise the cover and lift it off (Figure 3-20).

To replace the module cover, insert the tabs on the cover into the module slots. Lower the cover, then firmly press down on the spring-loaded fastener and rotate it clockwise.

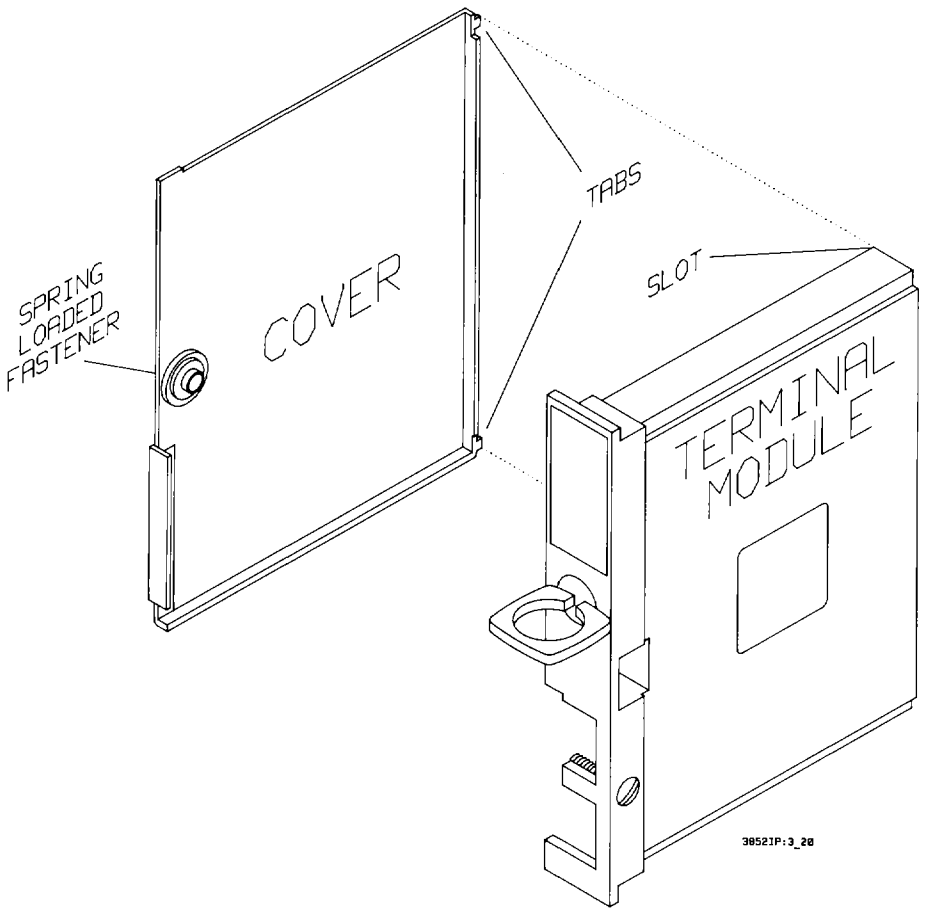
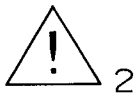


Figure 3-20. Removing the Terminal Module Cover

www.valuetronics.com

Installing the Accessories

The plug-in accessories can be installed in any of the eight slots (0-7) in the HP 3852A mainframe and in any of the 10 slots (0-9) in the HP 3853A extender. All accessories occupy one slot with the exception of the HP 44702A/B 13 bit voltmeter accessory which occupies two slots. The following information on accessory installation is divided into three parts: installing accessories that consist of a terminal module and a component module, installing single module accessories, and installing the HP 44702A/B voltmeter accessory with the HP 44711A, HP 44712A, and HP 44713A high-speed FET multiplexers.



WARNING

The mainframe and extender analog busses interconnect the multiplexer and voltmeter accessories to form one circuit. To protect against possible personal injury due to equipment failure or programming error, limitations are placed on the potentials that can appear on the bus. These limitations are listed below for the mainframe, extenders, and all plug-in accessories. For any given set of accessories installed in the mainframe or extender, the maximum potential on the analog bus should not exceed the LOWEST peak voltage limitation listed.

Instrument/Accessory	*Max. Peak Voltage Allowed on Analog Bus*
HP 3852A (Mainframe)	*350V*
HP 3853A (Extender)	*350V*
HP 44701A (Integrating Voltmeter)	*350V*
HP 44702A/B (High-Speed Voltmeter)	*42V*
HP 44705A/HP 44708A (Relay Multiplexers)	*170V*
HP 44705H/HP 44708H (Relay Multiplexers)	*350V*
HP 44706A (Relay Multiplexer)	*42V*
HP 44717A/HP 44718A (Strain Gage Relay Multiplexers)	*170V*
HP 44709A/10A/11A/12A/13A/19A/20A (FET Multiplexers)	*42V*
HP 44714A (Stepper Motor Controller/Pulse Output)	*350V*
HP 44715A (5-Channel Counter/Totalizer)	*350V*
HP 44721A/22A (Digital Inputs with Totalize and Interrupt)	*350V*
HP 44723A (High-Speed Digital Sense/Control)	*350V*
HP 44724A (16-Channel Digital Output)	*350V*
HP 44725A (General Purpose Switch)	*350V*
HP 44728A (Relay Actuator)	*350V*
HP 44729A (Power Controller)	*350V*
HP 44726A (2-Channel Arbitrary Waveform DAC)	*350V*
HP 44727A/B/C (Digital to Analog Converters)	*350V*
HP 44730A (Track/Hold with Signal Conditioning)	*42V*
HP 44732A/33A (Dynamic Strain Gage FET Multiplexers)	*42V*

www.valuetronics.com

1

WARNING

Accessory installation should be performed with the mainframe and extender turned off and all field wiring power disconnected.

NOTE

If accessories are installed with the power on, they are not recognized by the mainframe and, therefore, will not receive any commands intended for them. The mainframe must either be reset or have the power cycled before the accessories will be recognized.

NOTE

Consider installing those multiplexer and voltmeter accessories rated for higher voltages together in the mainframe or same extender. Install those multiplexer and voltmeter accessories rated for lower voltages in a second instrument. In this configuration, you would not have to connect the analog bus between the mainframe and extender when otherwise, you risk damage to all of your equipment in case of equipment failure or programming error.

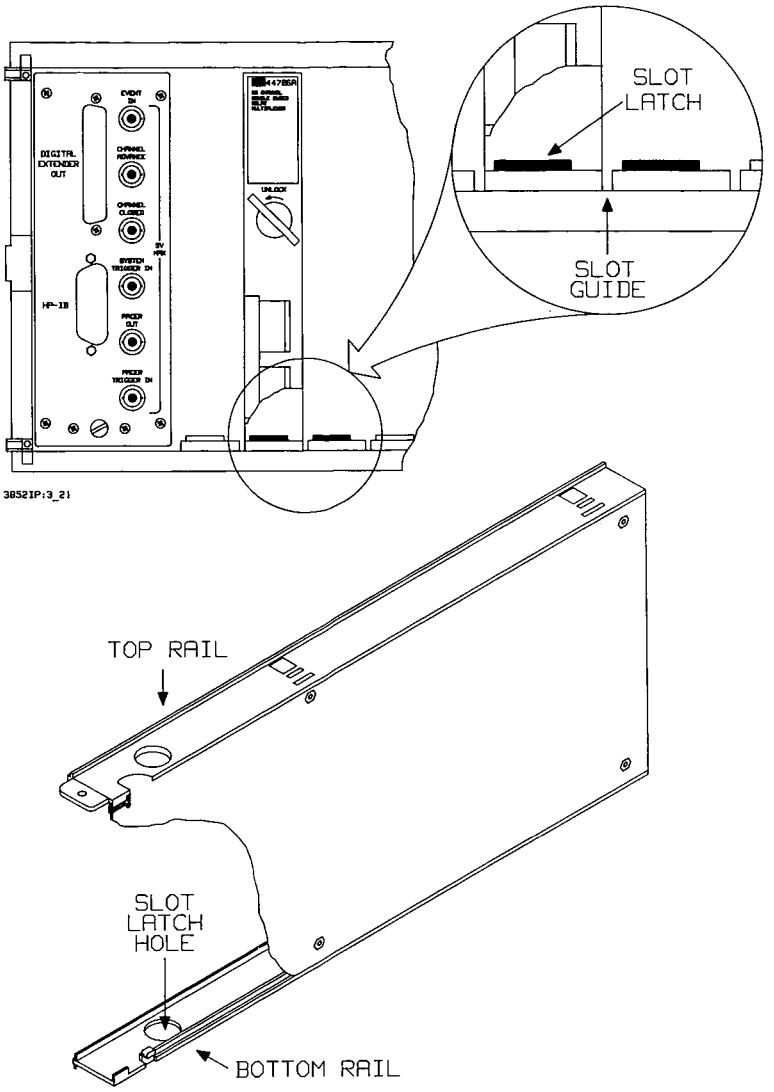
Terminal Module and Component Module Accessories

As noted, many of the plug-in accessories consist of a terminal module and a component module. Installation of these accessories is described in Figure 3-21.

Single Module Accessories

Installation of the single module accessories is described in Figure 3-22.

www.valuetronics.com



Installing an Accessory

1. Connect the terminal module to the component module and turn the locking ring to the right to lock the terminal module to the component module.

2. Line up the ridges on the top and bottom rails of the accessory with the slot guides to the left of the slot number on the mainframe. Make sure the terminal module cover is facing left.

3. Slide the accessory into the slot and press firmly on the terminal to lock the accessory in the slot. You'll hear a click when the accessory locks.

Removing an Accessory

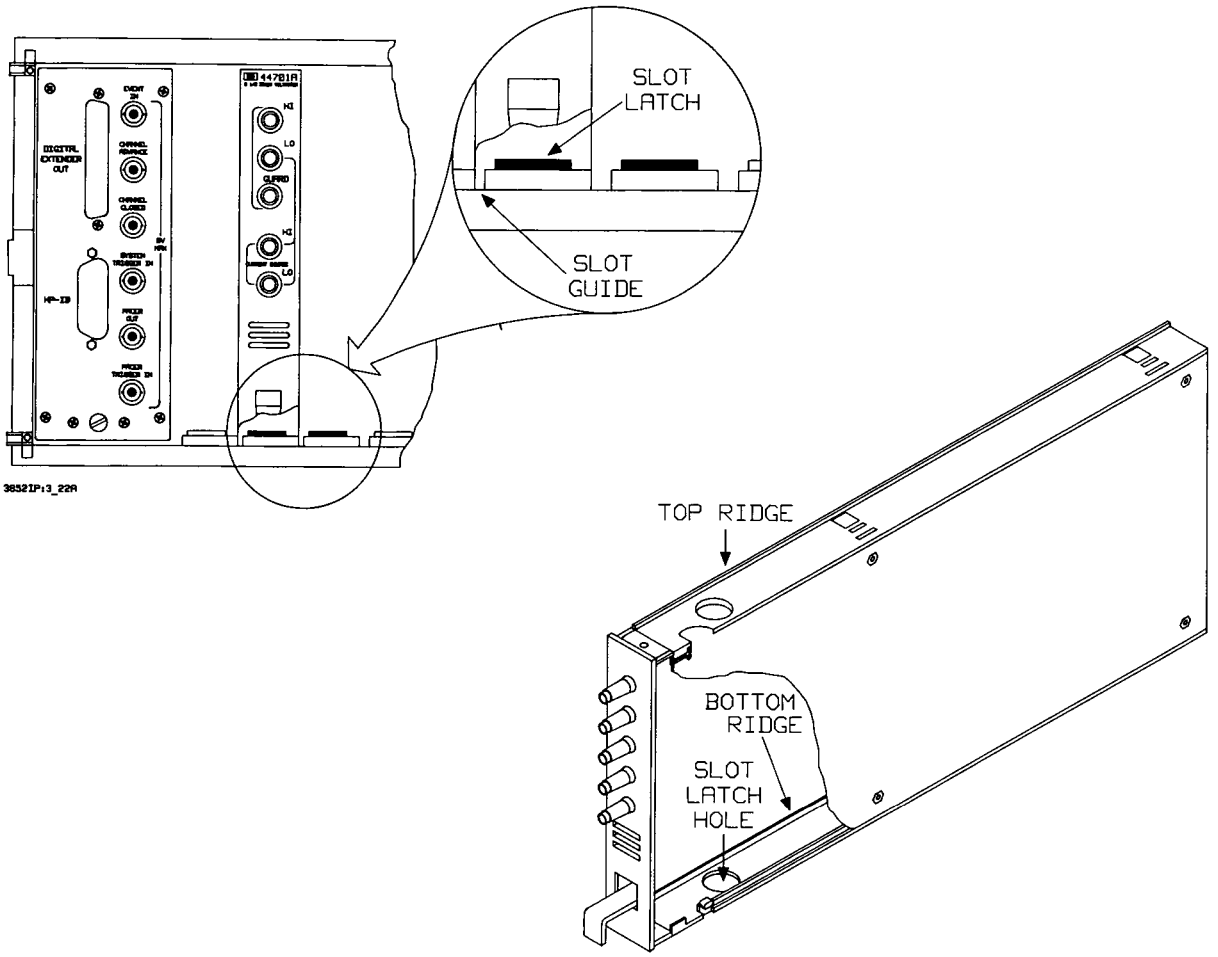
1. Turn the locking ring to the UNLOCK position and pull on the ring to remove the terminal module. The component module will remain in the slot.

2. To remove the component module, place your right forefinger in the hole in the top rail of the component module and your left forefinger on the slot latch in the bottom rail of the component module.

3. Press down on the slot latch while pulling firmly on the component module. BE CAREFUL—you can easily pinch your left forefinger between the slot latch and the bottom plastic rail of the component module.

Figure 3-21. Installing the Accessories

www.valuetronics.com



Installing the Accessory

1. Line up the ridges on the top and bottom rails of the accessory with the slot guides to the left of the slot number on the mainframe.

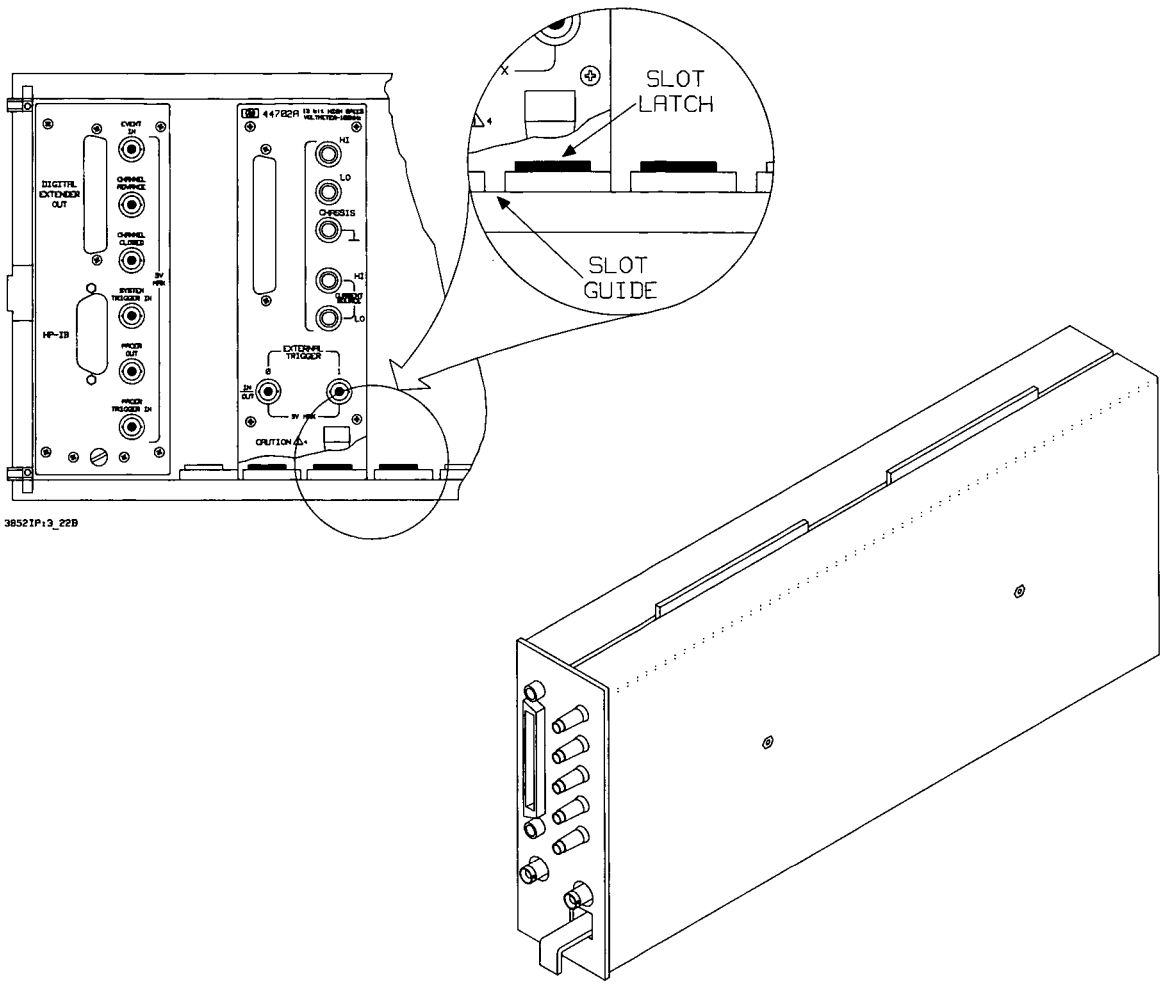
2. Slide the accessory into the slot and press firmly to lock the accessory in the slot. You'll hear a click when the accessory locks.

Removing the Accessory

To remove the accessory, lift the release lever FIRST. Then, firmly pull the accessory from the slot. If you try to pull it from the slot before you lift the release lever, you may jam the mechanism.

Figure 3-22. Installing the Single Module Accessories

www.valuetronics.com



Installing the HP 44702A/B Voltmeter

1. The HP 44702A/B voltmeter requires two slots. Because of a metal support in the mainframe, the voltmeter can't be installed in slots 3 and 4.

2. Line up the ridges on the top and bottom center of the accessory with the slot guide between the two slots the voltmeter is to occupy. For example, to install the voltmeter in slots 6 and 7, line up the ridges with the guide between slots 6 and 7.

3. Slide the voltmeter into the slots and press firmly to lock the voltmeter in the slot. You'll hear a click when the voltmeter locks.

Removing the HP 44702A/B Voltmeter

To remove the voltmeter, lift the release lever FIRST. Then, firmly pull the voltmeter from the slot. If you try to pull the voltmeter from the slot before you lift the release lever, you may jam the mechanism.

Figure 3-22. Installing the Single Module Accessories (Cont'd)

www.valuetronics.com

Installing the High-Speed FET Multiplexers

For many data acquisition applications the HP 44702A/B voltmeter accessory is used with a family of high-speed FET multiplexer accessories (HP 44711A/12A/13A). In these applications, the multiplexers and the voltmeter can be connected by a ribbon cable. The ribbon cable carries measurement and control data between the multiplexers and the voltmeter. This frees the backplane bus for other tasks and enables the high-speed FETs to be scanned at a faster rate since they are under voltmeter control.

The ribbon cable between the HP 44702A/B voltmeter and the high-speed multiplexer(s) is connected as shown in Figure 3-23.

Connecting an External Voltmeter

Multiplexer accessories that are installed in the mainframe or extender can be used with a voltmeter that is external to the instrument. The multiplexers are linked to the external voltmeter through the analog extender cable. The cable serves as an "extension" of the mainframe/extender backplane analog bus when connected between the analog extender port and the input terminals of the voltmeter. The scanning and measurement sequence between the mainframe and the voltmeter is controlled through trigger signals involving the "voltmeter complete" and "external trigger" ports of the voltmeter, and the CHANNEL ADVANCE and CHANNEL CLOSED ports of the mainframe.

Figure 3-24 shows you how the HP 3852A is connected to an external voltmeter to make voltage and resistance measurements. See the appropriate plug-in accessory manual for information on wiring and configuring the accessory terminal module for the measurement you plan to make.

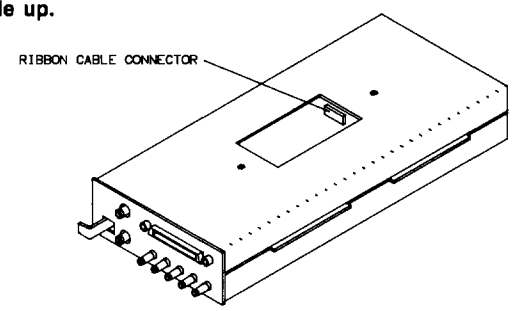
www.valuetronics.com

CAUTION

To prevent possible damage to the ribbon cable, check the ribbon cable connections before fully removing an HP 44711A, 44712A, or 44713A from its slot. Disconnect ribbon cable before removing the multiplexer.

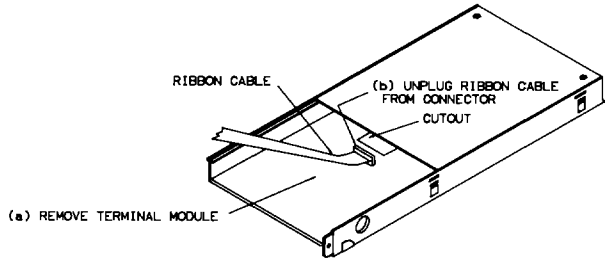
① **Position Voltmeter**

Place the HP 44702A/B on its side with ribbon cable connector side up.



② **Disconnect Ribbon Cable**

Remove HP 44711A, 44712A, or 44713A terminal module. Disconnect ribbon cable from connector and feed cable through cutout on multiplexer.



③ **Connect Cable to Voltmeter**

Attach ribbon cable from multiplexer to voltmeter ribbon cable connector. Replace multiplexer terminal module.

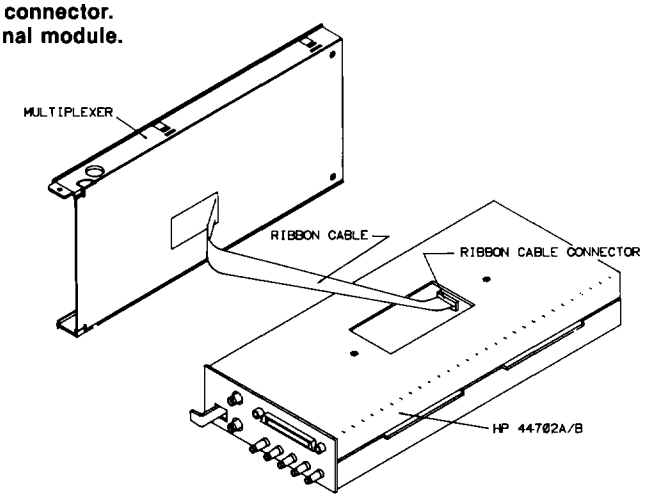
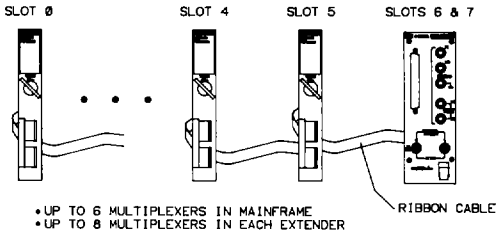


Figure 3-23. Installing the High-Speed FET Multiplexers

www.valuetronics.com

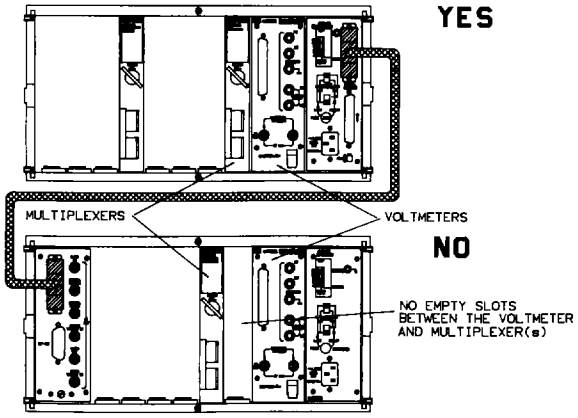
④ **Connect Multiplexers**

For each frame, join additional multiplexers by connecting the ribbon cable between multiplexers. For ribbon cable operation, an HP 44702A/B is required in the mainframe and in EACH extender containing multiplexers.



⑤ **Install Voltmeter(s)/Multiplexer(s)**

Install voltmeters and multiplexers in desired slots. The voltmeter cannot be installed in slots 3 and 4 of mainframe or in slots 4 and 5 of an extender. Recommend installing voltmeter in mainframe slots 6 and 7 and in extender slots 8 and 9.



Notes

1. If an HP 44711A, 447712A, or 44713A is used in other than ribbon cable mode, the ribbon cable must be connected to its connector on the component module.

Figure 3-23. Installing the High-Speed FET Multiplexers (Cont'd)

www.valuetronics.com

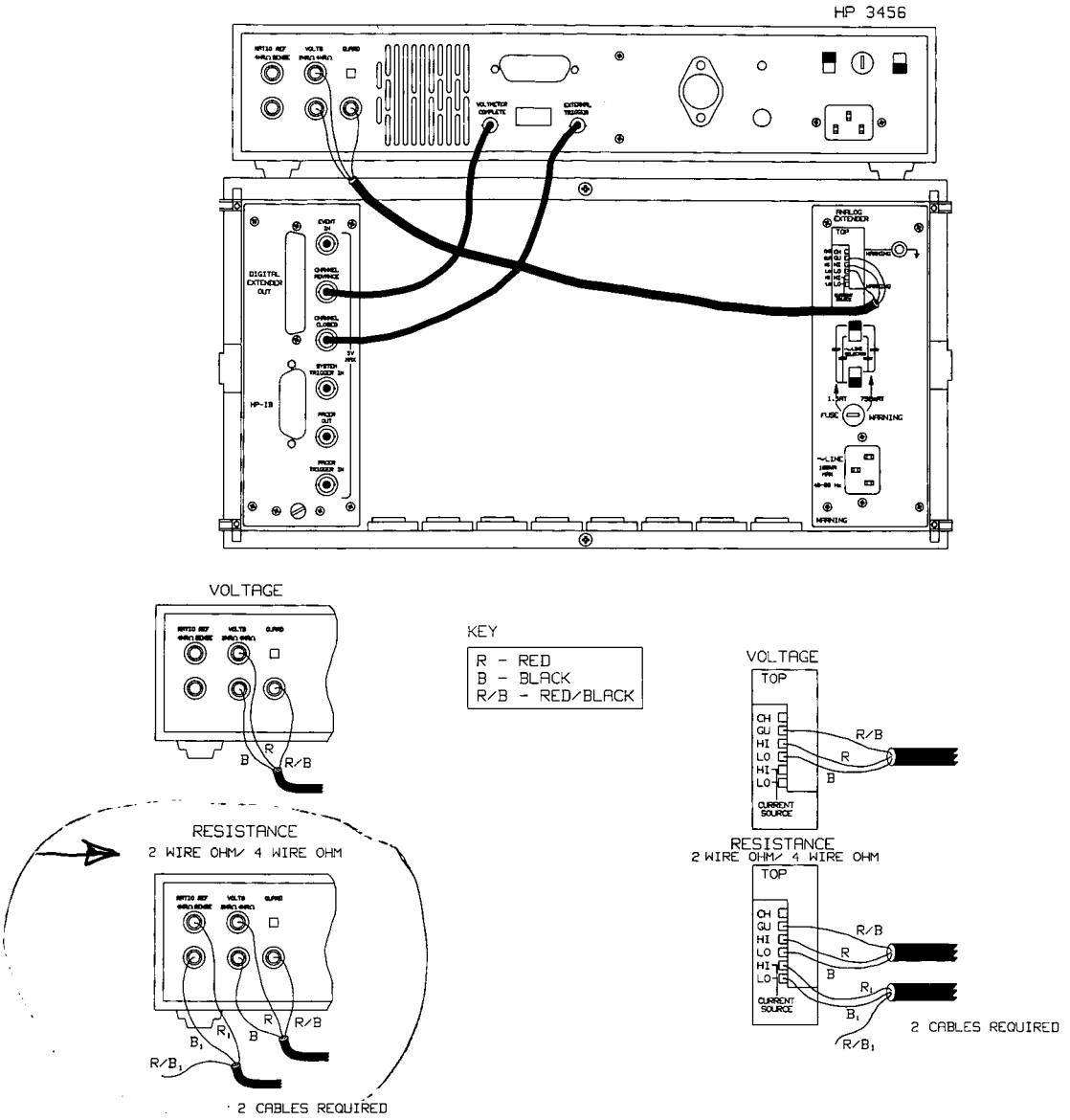
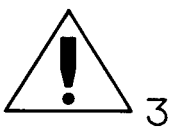


Figure 3-24. Connecting an External Voltmeter

Accessory Power Consumption



In addition to the number of slots in the mainframe and extender, the relative power consumption of the plug-in accessories also limits the number of accessories that can be installed in a single instrument. Relative power consumption is the amount of power used by an accessory "relative" to the power allocated for one slot. Table 3-3 lists the relative power consumption factors for each accessory based on the power supply shipped with the HP 3852A mainframe and HP 3853A extenders. The power supply shipped with the mainframe and extenders prior to December 1, 1987 was part number 03852-66202. When installing accessories in these instruments, the sum of the relative power factors shown cannot exceed eight in the mainframe, or 10 in the extender. For the new power supply (part number 03852-66212), the factors shown cannot exceed 10 in either the mainframe or extender.

www.valuetronics.com

Table 3-3. Accessory Power Consumption

Accessory	Relative Power Consumption
HP 44701A	1.2
HP 44702A/B	3.0
HP 44705A, HP 44705H, HP 44706A, HP 44708A, HP 44708H	0.2
HP 44709A, HP 44710A, HP 44711A, HP 44712A, HP 44713A	0.3
HP 44714A	1.0
HP 44715A	1.0
HP 44717A, HP 44718A	0.2
HP 44719A, HP 44720A	0.3
HP 44721A, HP 44722A	0.5
HP 44723A	0.7
HP 44724A	0.3
HP 44725A	1.0
HP 44726A	2.0
HP 44728A	0.5
HP 44729A	0.9
HP 44730A	3.0
HP 44732A, HP 44733A	3.0
HP 44788A	0.7

HP 3852A: total relative power consumption must be ≤8

HP 3853A: total relative power consumption must be ≤10

As an example, if two HP 44702A/B 13-bit High Speed Voltmeters and four HP 44711A High Speed FET Multiplexers were installed in the mainframe, the total relative power consumption would be 7.2.

Exceeding the total relative power consumption limits of the mainframe or extender can result in degraded and often unpredictable system performance.

Power Supply Part Number 03852-66202

Accessory	Relative Power Consumption
HP 44701A	1.2
HP 44702A/B	1.5
HP 44705A, HP 44705H, HP 44706A, HP 44708A, HP 44708H	0.1
HP 44709A, HP 44710A, HP 44711A, HP 44712A, HP 44713A	0.1
HP 44714A	0.3
HP 44715A	0.8
HP 44717A, HP 44718A	0.1
HP 44719A, HP 44720A	0.1
HP 44721A, HP 44722A	0.3
HP 44723A	0.7
HP 44724A	0.2
HP 44725A	1.0
HP 44726A	1.1
HP 44727A, HP 44727B, HP 44727C	1.4
HP 44728A	0.5
HP 44729A	0.9
HP 44730A	1.0
HP 44732A, HP 44733A	1.1
HP 44788A	0.1

HP 3852A: total relative power consumption must be ≤10

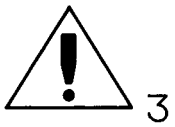
HP 3853A: total relative power consumption must be ≤10

As an example, if two HP 44701A voltmeters and three HP 44717A strain gage accessories were installed, the total relative power consumption would be 2.7.

Power Supply Part Number 03852-66212

www.valuetronics.com

HP 44727A/B/C Power Consumption



The relative power consumption factors for the HP 44727A/B/C digital-to-analog converter (DAC) accessories are based on the total output current of all four channels. Table 3-4 lists the relative power consumption factors for different output currents. Note that the factors and limitations shown apply to power supply part number 03852-66202.

Table 3-4. HP 44727A/B/C Relative Power Consumption

Total Output Current/DAC Accessory	Relative Power Consumption
0 to 20 mA	1.5
20 to 40 mA	1.6
40 to 60 mA	1.7
60 to 80 mA	1.8

HP 3852A: total relative power consumption must be ≤8

HP 3853A: total relative power consumption must be ≤10

Note - if a DAC channel or channels are configured for voltage, the user will need to divide the programmed amplitude of that channel by the load resistance to determine the output current. The sum of each channel's output current is the total current for that accessory.

Accessory Installation Hints

The following installation hints suggest methods for optimizing your system's performance.

Interrupt Priorities

If interrupts will be used with the accessories installed, consider installing the accessories in the lower numbered slots of the mainframe or extender. More importantly, if an accessory is to interrupt on a condition that occurs very infrequently or is to have the highest priority, the accessory should be installed in slot 0 of the mainframe. The reason for this is the polling routine used by the mainframe to locate the channel which interrupted. As the mainframe services the interrupt, it starts a polling routine beginning with the mainframe and continuing through each extender by increasing extender number until the frame the interrupt came from is located. Once the frame is known, the mainframe polls each slot in the frame until the interrupting slot is located. Once the slot is known, each channel is polled until the channel is located.

When multiple backplane interrupts have occurred, the polling sequence begins as described above with the lowest channel interrupt being located and handled first. Note, however, that after the interrupt has been handled, the polling routine restarts at the beginning rather than where it stopped when the first interrupting channel was located. It is possible then, that interrupts occuring on higher numbered channel would not be handled if interrupts on lower channels occur frequently enough, thus restarting the polling routine each time. See Chapter 8 - Using Interrupts for additional information on setting up and enabling backplane (accessory) interrupts.

www.valuetronics.com

Identifying Installed Accessories

Many of the plug-in accessories consist of a component module that is used with two or more different terminal modules. These accessories are identified by a label similar to that shown in Figure 3-25. The label is located on the component module and is visible when the terminal module is removed.

The label shown in the figure indicates that this particular component module is used with the HP 44705A, HP 44708A, HP 44717A, and HP 44718A accessories.

If accessories that use the same component module are installed in the mainframe or same extender, it is often difficult to recall the specific slot an accessory was installed in if the terminal modules are removed for configuration or wiring. To avoid this problem, write down the number of the slot in which the accessory is installed on the terminal module label. The label was designed for this and to allow room for any additional notes. Use a pencil or an ink pen as both can be erased should the slot number or application change.

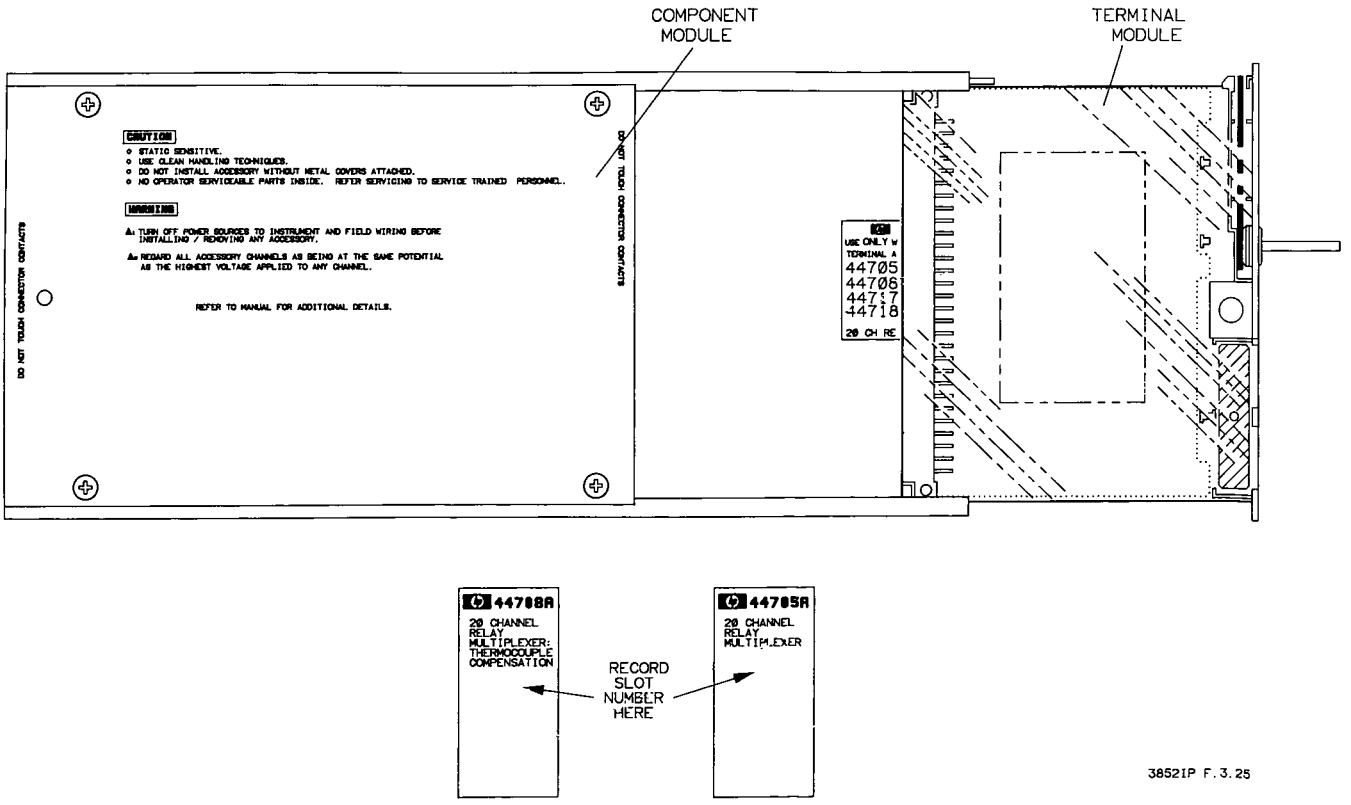


Figure 3-25. Identifying Accessories

www.valuetronics.com

HP 44727A/B/C Labels

The labels for the HP 44727A/B/C DAC accessories differ somewhat from those on the other accessories due to the fact that you can change the configuration of the DACs. For example, a DAC accessory shipped as an HP 44727A can be changed by the user via jumpers on the terminal module to either an HP 44727B, HP 44727C, or to none of the three (e.g. 3 channels voltage or current). Referring to Figure 3-26, if the accessory is shipped as an HP 44727A (channels 0-3 configured for voltage - V), the bar underneath the "A" will be darkened. If shipped as an HP 44727B (channels 0-3 configured for current - I) the bar under the "B" will be darkened. If the accessory is shipped as an HP 44727C (channels 0-1 voltage, channels 2-3 current), the bar under the "C" will be darkened.

Should you decide to change the configuration, mark the boxes on the label to help you identify the change. Use a pencil or an ink pen to mark the boxes as either can be erased.

NOTE

Earlier versions of the HP 44727A/B/C have different labels than those shown in Figure 3-26.

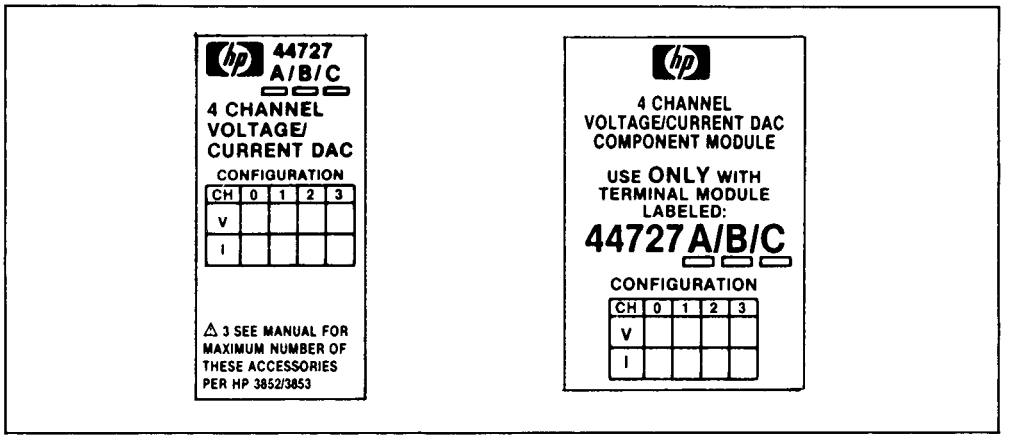
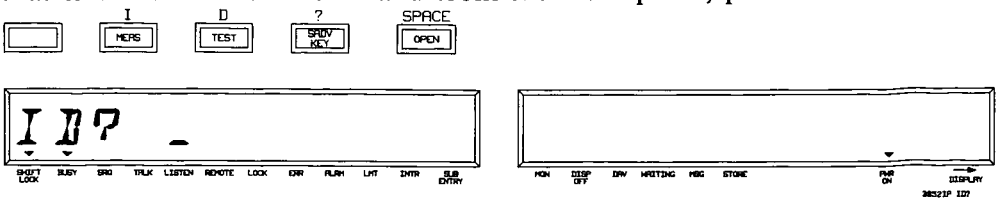


Figure 3-26. HP 44727A/B/C Identification

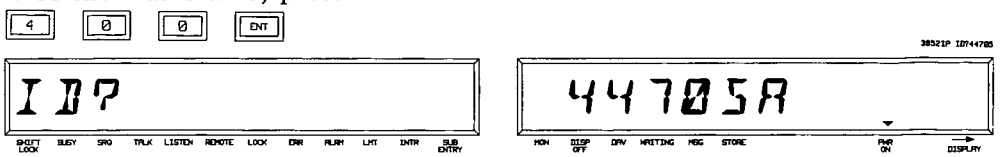
www.valuetronics.com

The ID? Command

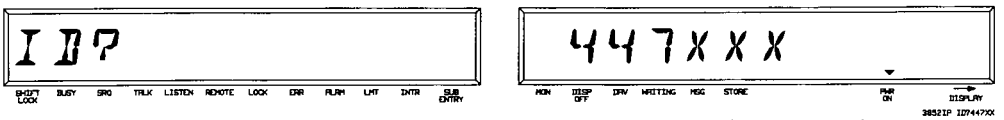
Another means of identifying accessories is through the ID? command. When a slot number is specified with this command, the mainframe responds by displaying the product number of the accessory installed in that slot. To enter the command from the front panel, press:



Assume that we want to know the identity of an accessory installed in slot 4 of the mainframe, press:



This example tells you that an HP 44705A 20-Channel Relay Multiplexer accessory is installed in the slot. Note that if the terminal module was removed from an accessory at power up or following a reset, the mainframe will return:



If no accessory is installed in the slot specified, 000000 is returned.

NOTE

The ID? command returns 44727X if an HP 44727A, HP 44727B, or HP 44727C DAC (digital-to-analog converter) accessory is installed in the slot specified in the command.

The following program can be sent from the controller to identify an installed accessory. (The program asks the identity of the accessory installed in slot 4 of the mainframe).

```
10 OUTPUT 709;"ID? 400"
20 ENTER 709;Identity$
30 PRINT Identity$
40 END
```

Table 3-5 represents the addressing conventions used to specify slots and channels within an HP 3852A system. Use the table when specifying addresses with the ID? command.

www.valuetronics.com

Table 3-5. HP 3852A Addressing Conventions

An addressing command contains a slot, channel, or a channel list as one of its parameters. Slots are specified as ES00, channels are specified as ESCC, and channel lists are specified as ESCC [-ESCC] [ESCC [-ESCC]...].

Definition:

E = Extender number	**S = Slot number in extender "E" where accessory is installed.**	**CC = Channel on accessory installed in slot "S".**
HP 3852A Mainframe: E = 0 HP 3853A Extender: E = 1-7* *Up to seven extenders per mainframe are allowed.	HP 3852A Mainframe: S = 0-7 HP 3853A Extender: S = 0-9* *Up to 78 slots per system (8 per mainframe, 10 per extender).	The maximum number of channels for any given slot is determined by the accessory in that slot.

When the parameter is a slot: (ES00)

Examples: (ES00)

0200	Specifies slot 2 in the HP 3852A mainframe. Leading zeros are optional.
A	Specifies the slot equivalent to the value of the variable A. If A = 100 for example, slot 1 in the mainframe is specified.
0	Specifies slot 0 in the HP 3852A mainframe.
1700	Specifies slot 7 in HP 3853A extender number 1.
OUTPUT 709;"ID? 3500"	Asks the identity of the accessory installed in slot 5 of HP 3853A extender number 3.

When the parameter is a channel: (ESCC)

Examples: (ESCC)

10	Specifies channel 10 of an accessory installed in slot 0 of the HP 3852A mainframe (leading zeros are optional).
1315	Specifies channel 15 of an accessory installed in slot 3 of HP 3852A extender number 1.
(SIN (1.57))	Specifies channel 1 on an accessory installed in slot 0 of the HP 3852A mainframe.
(SQR (25))	Specifies channel 5 of an accessory installed in slot 0 of the HP 3852A mainframe.
OUTPUT 709;"CLOSE 100"	Closes channel 0 of an accessory installed in slot 1 of the HP 3852A mainframe.

When the parameter is a channel list: (ESCC [-ESCC] [ESCC [-ESCC]...])

Examples: (ESCC [-ESCC] [ESCC [-ESCC]...])

ADRS	Specifies the list of channels represented in the array ADRS. If ADRS contains the numbers 200 and -202, the channel list is 200 through 202.
1110, 1201-1205	Specifies channel 10 in slot 1 of HP 3853A extender number 1 and channels 1 through 5 in slot 2 of extender number 1.
410-400	Specifies channels 10 through 0 in slot 4 of the HP 3852A mainframe.
0-7999	Specifies all channels in all slots of the HP 3852A mainframe and all HP 3853A extenders.
OUTPUT 709;"CLOSE? 0-1312"	Determines the channel state of channel 0 on an accessory installed in slot 0 of the HP 3852A mainframe, through channel 12 on an accessory installed in slot 3 in HP 3853A extender number 1. Note that in this example, all slots in the mainframe through slot 3 in the extender are checked for accessories with channel closures.

www.valuetronics.com

Installing the Extended Memory Boards

In the standard HP 3852A, readings and subroutines are stored in approximately 11000 bytes of mainframe memory. The memory can be expanded an additional 256k, 1M, 2M, or 4Mbytes with the various extended memory boards available. The extended memory boards are installed on the mainframe controller module and, therefore, do not require a separate mainframe slot.

The following installation procedure applies to all extended memory boards. Switch locations and settings for establishing the starting location (address) of the extended memory are different for each board thus all settings and locations are included.

CAUTION

The extended memory boards are sensitive to electrostatic discharge. Carefully follow the procedures and precautions outlined in the following steps. If possible, install the memory board at a static-controlled workstation that includes personnel grounding provisions to protect against static build-up.

---

1. If you have unpacked your memory board, keep it in the anti-static bag and set it aside temporarily.
2. Turn the HP 3852A off and remove the power cord. Remove the controller module. The controller module contains the BNC ports, HP-IB connector, and Digital Extender cable connector and is located on the left side of the instrument when viewed from the rear. Using a flatblade screwdriver, turn the screw in the bottom center of the module one-quarter turn to the left. Firmly pull the module out from the backplane connector.

NOTE

It may be necessary to remove the accessory installed in slot 0 in order to get a grip on the controller module.

---

3. Lay the module on its side with the sheet metal opening face up. Locate the two stand-offs and remove the screws (Figure 3-27).

www.valuetronics.com

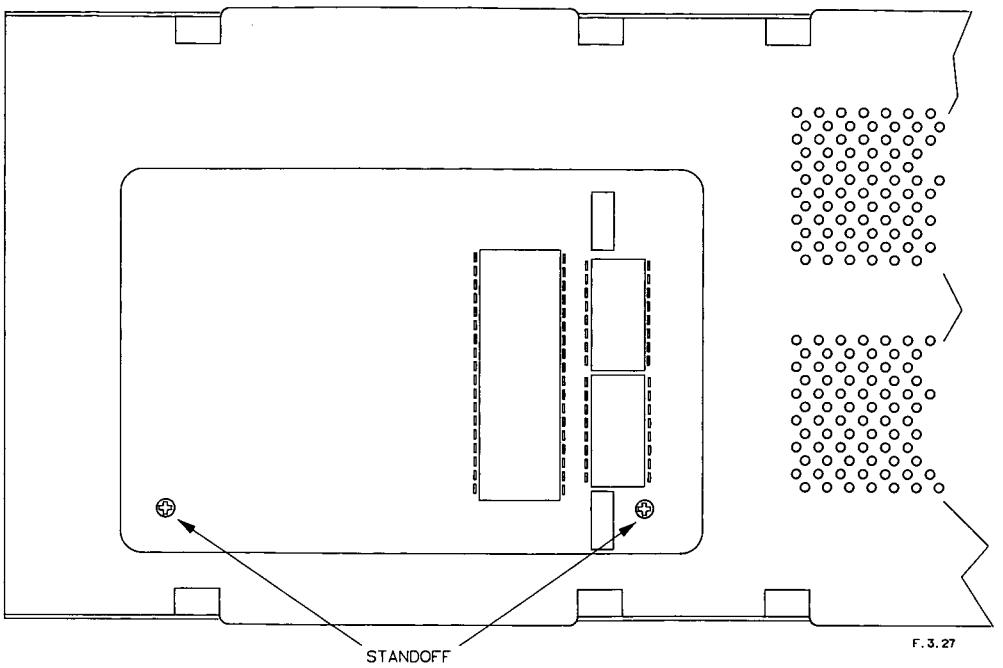
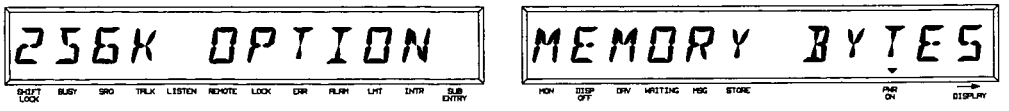


Figure 3-27. Controller Module "Stand-off" Location

4. Remove the extended memory board from the anti-static bag and set the board on top of the bag. Handle the board by the plastic extractors located on the corners. Avoid touching any of the metal circuitry.
5. Locate the DIP switch on your particular board (Figure 3-28). Set the address as shown in the figure.
6. Insert the board through the "window" in the sheet metal and align the edge pins on the board with the edge connector on the controller module. Lower the board into the module until it rests on the stand-offs, then firmly press the board into the edge connector.
7. Replace the stand-off screws to secure the board in place. Take your time and avoid dropping the screws on the printed circuit board.
8. Align the controller module in the appropriate tracks of the mainframe and re-install the module by firmly pressing it into the backplane connector. Rotate the locking screw one-quarter turn to the right to secure the module in place.
9. Connect the power cord to the mainframe then turn the instrument on and note the power-on sequence. If, for example, an HP 44703A extended memory board was installed and the address switches were properly set, the mainframe will display:



www.valuetronics.com

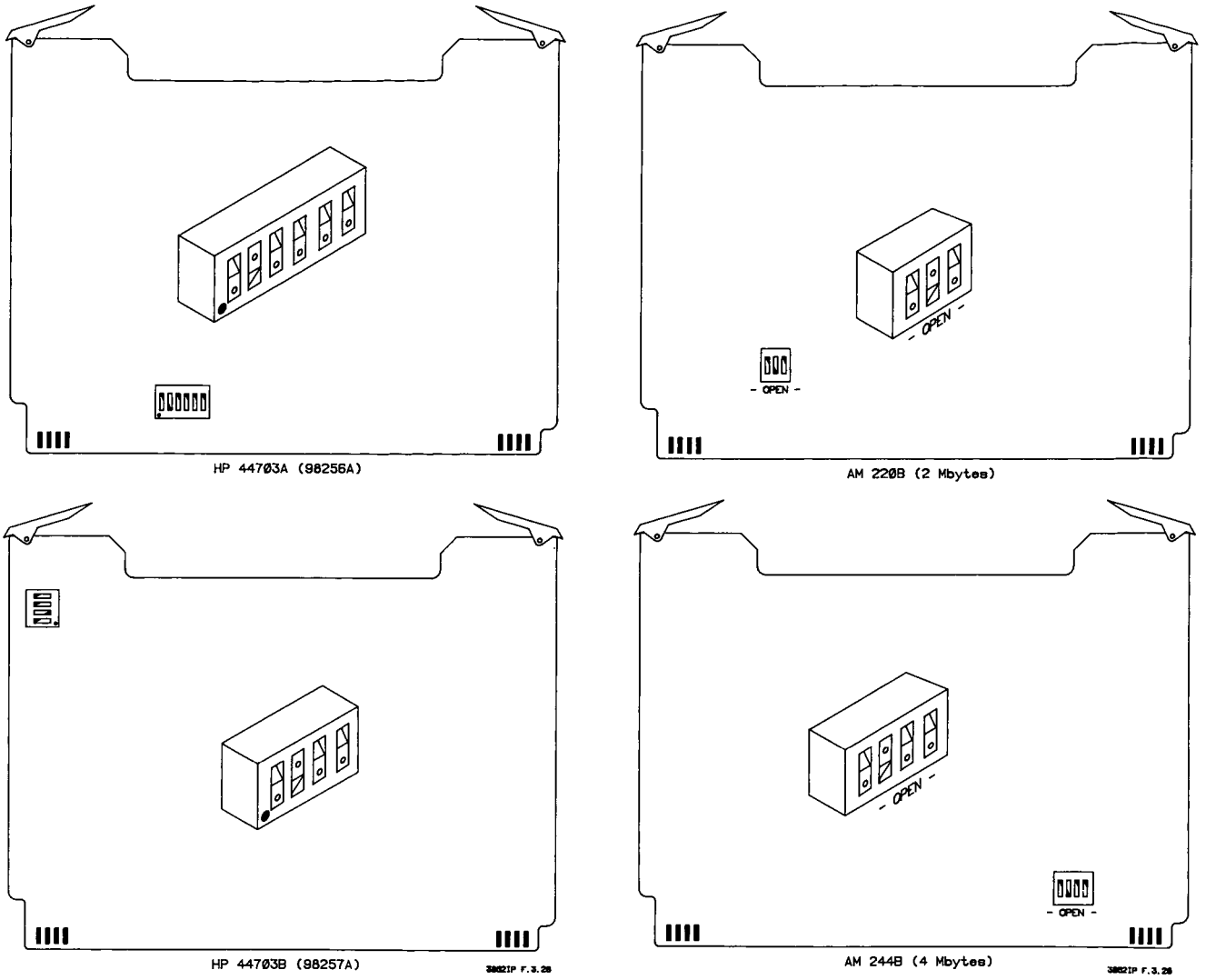
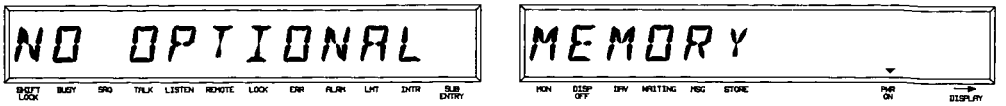


Figure 3-28. Memory Board Switch Locations and Address Settings

momentarily during its power-on sequence. If the address on the board is set incorrectly, the mainframe will complete its power-on sequence, however, a power-on test failure may be registered and the instrument will also display:



Connecting the HP-IB

Connecting the HP-IB includes:

- Introduction to the HP-IB
- Connecting the HP-IB Cable
- Setting the HP-IB Address

www.valuetronics.com

NOTE

The HP-IB information in this section covers installation of the interface and setting the HP-IB address only. HP 3852A HP-IB interface capabilities plus the operational and programming aspects of the interface are covered in Chapter 5 - "HP-IB Communication".

Introduction to the HP-IB

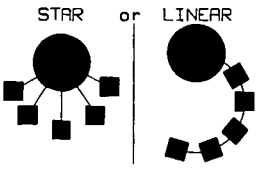
The Hewlett Packard Interface Bus (HP-IB) is the communication link between a controller and the HP 3852A mainframe. The HP-IB is Hewlett-Packard's implementation of IEEE Standard 488-1978, "Standard Digital Interface For Programmable Instrumentation". Since the HP-IB is a standard interface, the HP 3852A mainframe can be used with a wide variety of controllers and other instruments.

The HP-IB interface carries data plus control and programming information between the controller and the mainframe. The interface has 16 signal lines that connect the devices. Eight lines are used to pass data and information between the devices while the other eight manage the flow and sequence of the transfer (Figure 3-29).

Optimum performance of the HP-IB requires a properly installed interface. The following installation guidelines and restrictions apply to the HP 3852A as well as any device that is connected to the bus:

NUMBER OF INTERCONNECTED DEVICES - Up to 15 devices (including a controller) can be connected to a single interface.

INTERCONNECTION PATH - Devices can be connected to the interface in either a "star" or "linear" fashion. If connected in a star network, avoid stacking more than two cable connectors on any one HP-IB port. If the stack is too large, any pressure on the stack may damage the connectors and affect HP-IB operation.



www.valuetronics.com

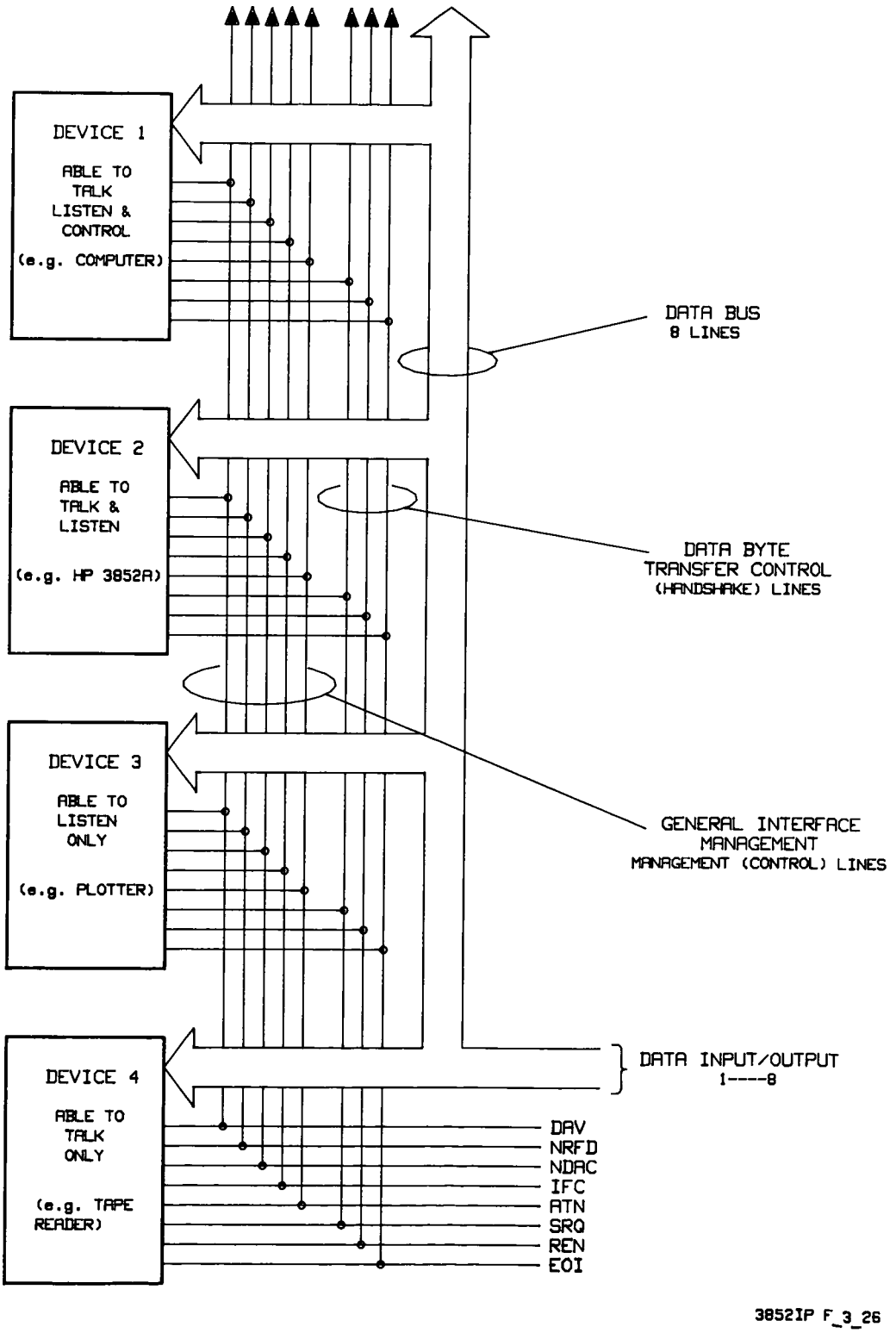


Figure 3-29. HP-IB Interface Structure

www.valuetronics.com

CABLE LENGTH - The cable length of the interface must be less than or equal to two metres times the number of devices that will be connected. The maximum cable length between any two devices is four metres. The total cumulative cable length for the system, however, cannot exceed 20 metres (65.6 ft.). For example, a system containing six devices can be connected together with cables that have a total length less than or equal to 12 metres (two metres/device times six devices). The individual lengths of cable can be distributed in any manner desired, provided there is no more than four metres of cable between devices and the cumulative length does not exceed 12 metres.

Connecting the HP-IB Cable

To connect the HP-IB cable between the HP 3852A and the controller:

1. Connect one end of the HP-IB cable to the HP-IB port on the mainframe. Tighten the connector to the port using the thumbscrews (Figure 3-30). The cable connector fits into the port only one way. It may be necessary to rotate the plug 180 degrees to make the connection.
2. Connect the other end of the HP-IB cable to the HP-IB port on the controller. Tighten the connector to the port using the thumbscrews (Figure 3-30).

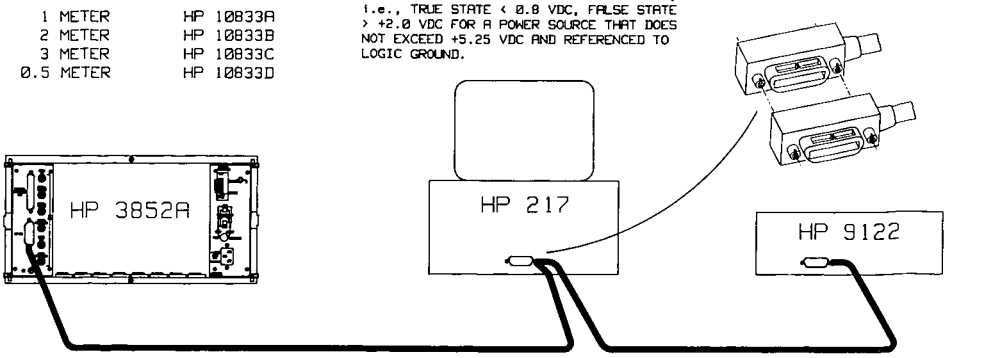
Table 3-6. HP-IB Addresses

Device	Talk	Listen
0	@	SP HP-IB Disc Drives
1	A	! HP-IB Printers
2	B	"
3	C	#
4	D	$
5	E	% HP-IB Plotters
6	F	&
7	G	'
8	H	(
9	I	) HP 3852A Factory Setting
10	J	*
11	K	+
12	L	,
13	M	-
14	N	.
15	O	/
16	P	0
17	Q	1
18	R	2
19	S	3
20	T	4
21	U	5 usually the controller
22	V	6
23	W	7
24	X	8
25	Y	9
26	Z	:
27	[	;
28	\	<
29	]	=
30		Λ
Listen Only		

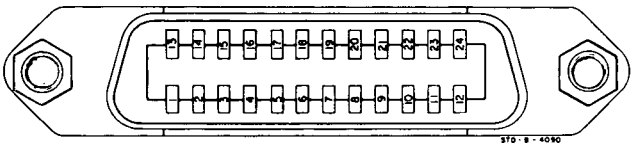
www.valuetronics.com

CABLE LENGTH	PART NUMBER
1 METER	HP 10833A
2 METER	HP 10833B
3 METER	HP 10833C
0.5 METER	HP 10833D

THE HP LOGIC LEVELS ARE TTL COMPATIBLE, i.e., TRUE STATE < 0.8 VDC, FALSE STATE > +2.0 VDC FOR A POWER SOURCE THAT DOES NOT EXCEED +5.25 VDC AND REFERENCED TO LOGIC GROUND.



3852IP F_3_27



PIN	LINE	
1	DI01	
2	DI02	
3	DI03	
4	DI04	
13	DI05	
14	DI06	
15	DI07	
16	DI08	
5	EOI	
17	REN	
6	DAV	
7	NRFD	
8	NDAC	
9	IFC	
10	SRQ	
11	ATN	
12	SHIELD–CHASSIS GROUND	
18	P/O TWISTED PAIR WITH PIN 6	THESE PINS ARE INTERNALLY GROUNDED
19	P/O TWISTED PAIR WITH PIN 7	
20	P/O TWISTED PAIR WITH PIN 8	
21	P/O TWISTED PAIR WITH PIN 9	
22	P/O TWISTED PAIR WITH PIN 10	
23	P/O TWISTED PAIR WITH PIN 11	
24	ISOLATED DIGITAL GROUND	

CAUTION

The 3852A contains metric threaded HP-IB cable mounting studs as opposed to English threads. Metric threaded HP 10631A, B, or C HP-IB cable lockscrews must be used to secure the cable to the instrument. Identification of the two types of mounting studs and lockscrews is made by their color. English threaded fasteners are colored silver and metric threaded fasteners are colored black. DO NOT mate silver and black fasteners to each other or the threads of either or both will be destroyed. Metric threaded HP-IB cable hardware illustrations and part numbers follow.

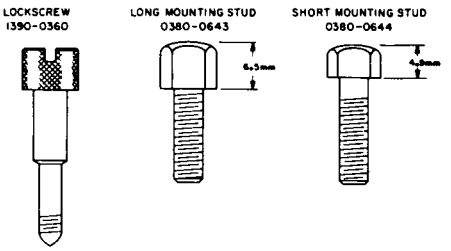


Figure 3-30. Connecting the HP-IB Cable

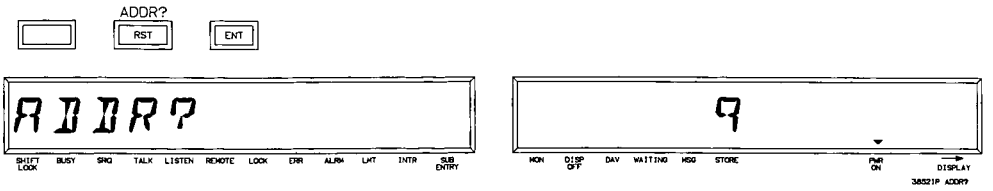
Setting the HP-IB Address

The HP 3852A has a factory set HP-IB address of "9". Normally, you need only view the address and record it for programming purposes. However, since each device on the HP-IB interface must have its own address, it may be necessary at some point in time to change the address.

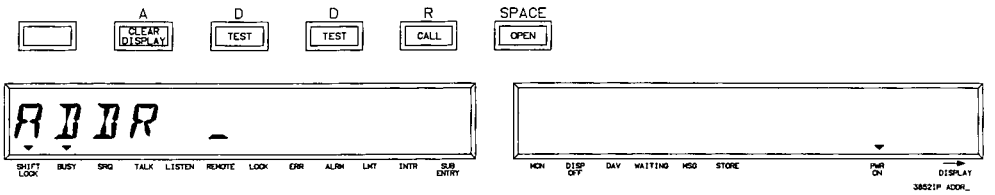
The mainframe address can be set from the front panel only. Addresses are any decimal number between 0 and 30 (see Table 3-6). You may want to note the factory set addresses of peripherals that sometimes are part of an HP 3852A system before changing the mainframe address.

www.valuetronics.com

The mainframe's HP-IB address is displayed momentarily as part of the power on sequence. The HP-IB address can also be viewed by entering the ADDR? command. To execute the command, press:



To set the HP-IB address, press:



The instrument is now waiting for its address. Enter the desired address using the numeric keys and execute the command by pressing:

The address is now stored in nonvolatile memory and is not lost when the instrument is cleared, reset, or if the power is cycled.

www.valuetronics.com

Contents

www.valuetronics.com

Chapter 4

Front Panel Operation

Introduction

The purpose of this chapter is to teach you how to operate the HP 3852A Data Acquisition/Control Unit from its front panel. The chapter is divided into four major sections: Introduction, Front Panel Familiarization, Entering Commands, and Recognizing Functions.

- **Introduction** reviews the contents of this chapter and calls attention to a series of mainframe installation and configuration checks that should be made before power is applied to the instrument. This section tells you how to turn on the instrument and describes its power-on sequence. Since keyboard exercises are included in each of the following sections, mistakes could occur when pressing the keys shown in the examples. This section shows you how to correct various "programming" mistakes and get the instrument into a known operating state.

- **Front Panel Familiarization** explains how the keyboard is arranged and how specific characters are accessed. This section also describes the instrument's display, what it monitors, and how it is controlled.

- **Entering Commands** covers the command format associated with the mainframe and its accessories. It then shows you how commands are entered from the front panel in that format.

- **Recognizing Functions** shows you how to recognize instrument (system) capabilities based on the way the front panel keys are grouped.

www.valuetronics.com

Before Applying Power

This chapter assumes that the HP 3852A is already installed. If the instrument is not currently installed and before you apply power:

1. Inspect the instrument for damages.
2. Ensure that the proper line fuse is installed.
3. Check that the LINE SELECTOR switches are set for the line voltage present.

Under no circumstances should you apply power to the instrument if damage is apparent. Chapter 3 contains detailed installation information on the HP 3852A. Refer to that chapter for information on installing the proper line fuse and setting the LINE SELECTOR switch.

CAUTION

Various commands used throughout this chapter to emphasize keystrokes require that a plug-in accessory be installed before the command will be accepted. As a precaution, a service-trained individual should disconnect all field wiring and power sources connected to the accessories or the mainframe/extender backplane bus. This will prevent damage to your equipment or application should you accidently press the wrong key or enter the wrong command when following the examples found throughout this chapter.

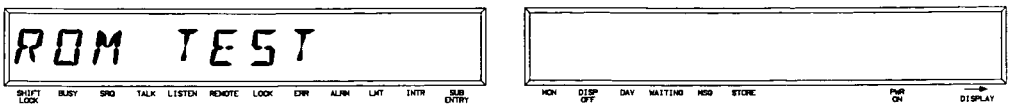
CAUTION

Connecting the mainframe/extender to the line voltage when the LINE SELECTOR switch is improperly set may destroy the fuse. Using the wrong fuse value and type for the line voltage present can result in damage to the circuitry inside the instruments.

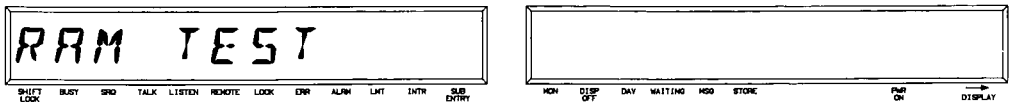
Applying Power

When applying power to your system, all extenders (if connected) must be turned on as well as the mainframe in order for the mainframe to complete its power-on sequence. Depress the LINE switch on the extender(s) then depress the line switch on the mainframe. As the mainframe is turned on, it "beeps" once and its cooling fan begins to run. Simultaneously, the power-on sequence of Figure 4-1 occurs. The mainframe will not accept inputs from its keyboard or over the HP-IB during the power-on sequence.

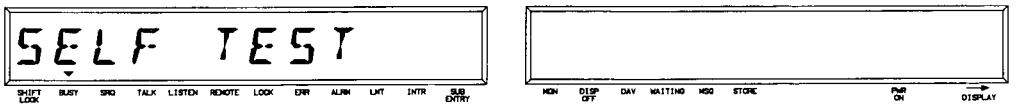
www.valuetronics.com



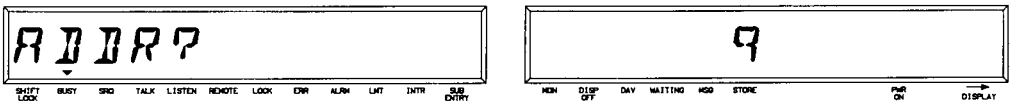
Tests the mainframe's read only memory



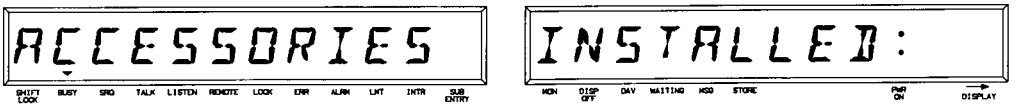
Tests the mainframe's read/write memory



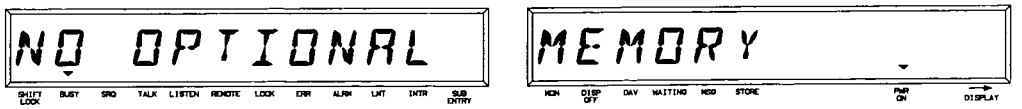
Tests the mainframe's controller module and front panel



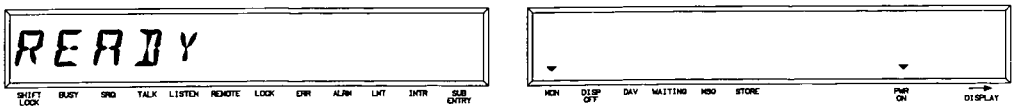
Displays the mainframe's HP-IB address. (The factory set address is 9.)



Indicates the number of accessories installed in the mainframe and extender (s)



Indicates whether or not an extended memory board is installed.



The mainframe completes its power-on sequence and is "READY" to accept commands.

Figure 4-1. HP 3852A Power-on Sequence

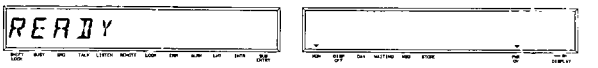
www.valuetronics.com

The Power-on State

Once the instrument successfully completes its power-on sequence and before any commands are entered, the mainframe is said to be in its power-on state. Table 4-1 lists the power-on state of the mainframe.

Table 4-1. HP 3852A Power-On State

Front Panel/HP-IB Modes

Display: 

Display Modes:	-	DISP ON, FASTDISP ON, MON ON
Keyboard Modes:	-	LOCK OFF, BEEP ON
HP-IB Modes:	-	BLOCKOUT OFF, FASTOUT OFF, SYSOUT OFF

Rear Panel BNC Ports

EVENT IN:	-	disabled (enabled by WAIT FOR event)
CHANNEL ADVANCE:	-	disabled (enabled by STRIG SADV or SADV CHADV)
CHANNEL CLOSED:	-	idle (outputs a negative-going TTL level pulse when a channel is closed)
SYSTEM TRIGGER IN:	-	disabled (enabled by TRG EXT)
PACER OUT:	-	idle (sends continuous pulse train, 500 ns negative going pulses occurring every 1 μs when pacer trigger is received)
PACER TRIGGER IN:	-	disabled (enabled by PTRIG EXT)

Internal

MEMORY:	-	all memory is cleared except HP-IB address, POWEROFF state, and the LCL bit in the Status Register
SYSTEM TRIGGER:	-	disabled (see TRG)
REAL TIME CLOCK: and CALENDAR	-	not affected (ALRM is disabled)
SERVICE REQUEST: MODE (RQS)	-	enabled
BUFFERS:	-	input buffer, output buffer, and error buffer are cleared
BUFFER MODES:	-	INBUF OFF, OUTBUF OFF
USE Channel:	-	the lowest slot number and accessory channel number in that slot for which the USE command is valid

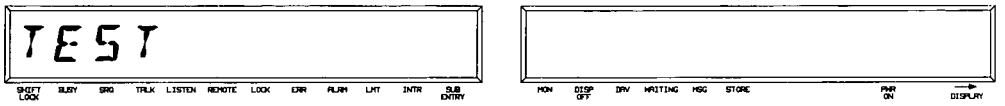
www.valuetronics.com

Self Test The self test that occurs at power-on provides a 90% confidence level that operation of the mainframe will be within its specifications if no error messages were displayed. Another self test of the mainframe can be performed by executing the TEST command. The self test activated by the TEST command provides a 60% confidence level that the mainframe is operating within specification. This self test is useful when the mainframe has been pre-programmed as none of the instrument's current operating states are affected. Recall that the power-on self test occurs with all mainframe operating parameters being set to their power-on conditions. To activate the self test, press:

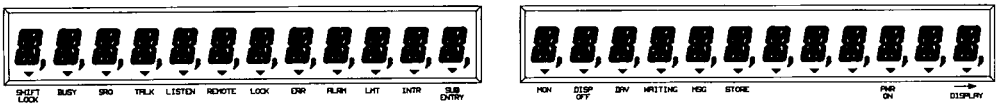
D

TEST ENT

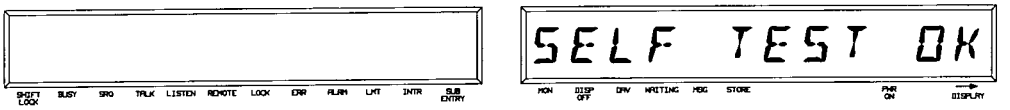
When executed, the HP 3852A shows the display sequence of Figure 4-2.



The TEST command is entered into the mainframe and the self test begins.



The display and "beeper" are tested as part of the self test.



Indicates that the self test passed.

Figure 4-2. HP 3852A Self Test Sequence

www.valuetronics.com

Power-on and Self Test Error Codes

If the HP 3852A displayed an error code during its power-on or self test sequence, refer to Chapter 3 for a description of the error code and references to further information.

Correcting Mistakes

As you learn to operate the instrument, examples and commands are bound to be typed in incorrectly. No problem. As cautioned before, as long as all field wiring is disconnected from your application, your equipment cannot be damaged should you accidently press the wrong key or enter the wrong command. The following remedies should get you past any problems you might encounter.

Problem - the wrong key was pressed, but not entered

Remedy - if you need to make a correction and the command has not been entered (ENT key pressed), press:

to erase one character at a time. Characters that are retyped will start at the position of the cursor. Another remedy is to press:

Pressing this key clears the entire display enabling you to start at the beginning. (Note - if the keyboard is in the shifted mode, the letter "A" will appear when the key is pressed.)

You can CLEAR or ReSeT the instrument, however, these commands also set many of the system's operating parameters to predefined power-on conditions which may not always be desirable. See Clearing and Resetting the Instrument in this section when deciding which function to perform.

Problem - pressed, then entered the wrong key or command

Remedy - if you entered the wrong key or command, the instrument may either accept it or display an error message. In either case, simply enter the command again. Pressing:

www.valuetronics.com

will restore the last command entered where it can then be modified starting with the BACK SPACE key. If an error message was displayed, the ERR annunciator in the display will remain on indicating that an error occurred and the error message is presently stored in a buffer. See "Error Messages" in this section for instructions on accessing and clearing this buffer.

Problem - front panel "locks up"

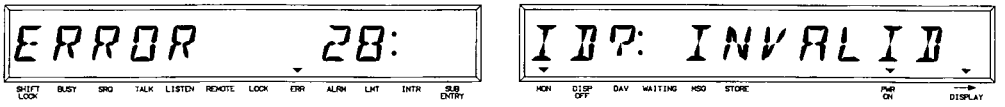
Remedy - if the front panel locks up it is probably due to the fact that the mainframe is still executing the previous command, however, clearing the instrument should help you regain control. If the mainframe is in the remote operating mode (receiving commands from a controller), you will only have "partial" control of the mainframe from the front panel. Pressing:

Z
LOCAL

will restore complete front panel control if the controller is no longer sending commands. Note that if the keyboard has been "locked out" by the HP 3852A LOCK command or by the HP-IB universal command LLO, the LOCAL key is disabled. Sending the appropriate commands to "unlock" the keyboard or cycling the power on the mainframe will restore front panel control.

Error Messages

When a front panel or HP-IB programming error occurs, an error message is displayed. An error message consists of an error code and its accompanying message.



CODE MESSAGE DW412

When displayed, the error code appears in the left display window while the message appears in the right display window. Pressing:

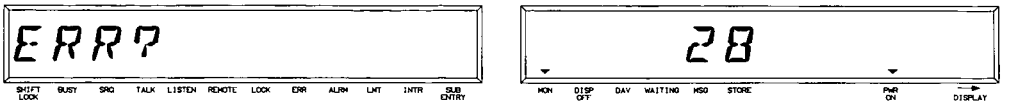
Q
DISPLAY

when the message first appears will scroll the display so that you can view the entire message. A list of all error messages associated with the HP 3852A and its plug-in accessories can be found in Appendix B of this manual.

www.valuetronics.com

The Error Buffer

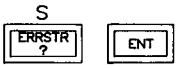
When the error message is displayed, it is simultaneously stored into an error buffer. The error buffer can store up to four error messages. If the buffer is full and errors occur, the messages are displayed but not stored. The ERR annunciator indicates when an error message is in the buffer. The annunciator will remain on until the message is retrieved from the buffer, even though the error message that appeared simultaneously in the display may be gone. Error messages in the error buffer have no effect on the operation of the instrument. Error messages are retrieved from the buffer using either the ERR? or ERRSTR? commands. The ERR? (or ERROR?) command reads the error register yet only ERR? and the error code are displayed.

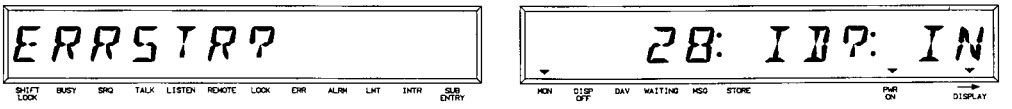


DW413

The ERR? command is executed by pressing:

Executing the ERRSTR? command reads the error buffer and displays both the error code and message. When executed, ERRSTR? will appear in the left display window and the error code and message will appear in the right display window. To retrieve an error message using this command, press:



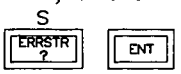


DW415

Pressing:

will scroll the right display so that you can view the entire message.

If the ERR annunciator remains on after ERR? or ERRSTR? has been sent, one or more error messages remain in the buffer. Pressing:



repeatedly or sending the ERR? command repeatedly (four times at most) will eventually clear the buffer, thus turning off the annunciator. Note that error messages are retrieved from the buffer in the order in which they occurred.

www.valuetronics.com

Clearing and Resetting the Instrument

During operation of the mainframe it may be necessary to place the instrument into a "known" operating state. This is done by clearing or resetting the instrument. While these procedures initialize the instrument, their overall effect on the system are somewhat different. To clear the instrument, press:

Clearing the instrument:

- Clears partially entered commands (front panel and HP-IB).
- Disables event interrupt recognition (DISABLE INTR SYS), limit interrupts (DISABLE LMT), and alarm interrupts (DISABLE ALRM). Backplane interrupts and channel logging remain enabled (if previously enabled).
- Sets RQS ON and RQS NONE (masks all bits in the HP 3852A Status Register).
- Clears the service request bit (bit 6) in the Status Register, the SRQ annunciator in the display, and the HP-IB SRQ line.
- Clears the HP-IB input and output buffers.
- If the display is enabled, READY is displayed on the front panel.
- Attempts to minimize other state changes (relay settings, output states, memory values, etc.).

With the RST command you can reset your entire system by resetting the mainframe, or you can reset a specific plug-in accessory. To reset the system, press:

Resetting the instrument:

- Returns the mainframe to its power-on state (Table 4-1).
- Returns all plug-in accessories (in mainframe and extender) to their power-on state.

www.valuetronics.com

The following are exceptions to a system reset.

- The HP-IB addressed state (LOCS, LWLS, REMS, RWLS) is not changed.
- The Status Register mask and mode are not changed.
- The self-test is not performed.
- The power-on "beep" does not occur.
- The power-on messages are not displayed.
- The LCL bit in the Status Register is not set.

Resetting an Accessory

When you specify a slot address with the RST command, you only reset the accessory in that slot, rather than the entire system.

NOTE

Before using the RST command from the front panel to reset a slot, recall that the command is executed the moment the RST key is pressed, thus resetting the entire system. Therefore, to specify a slot number, you must enter the command a character at a time using the letters printed in blue above the front panel keys.

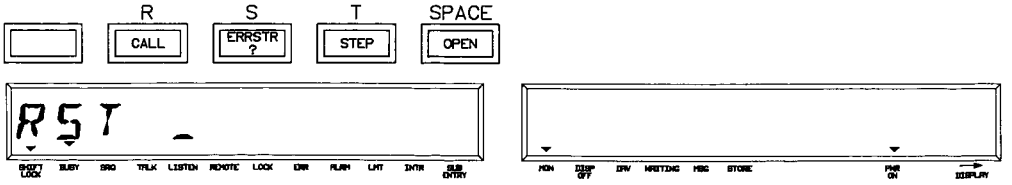
The format of the RST command when it is used to reset a slot is:

RST [*slot*]

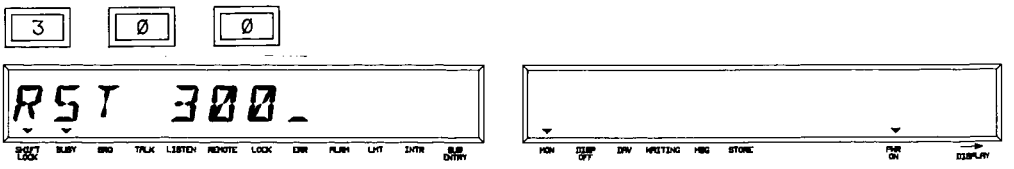
or

RESET [*slot*]

Therefore, to reset a slot press:



Here we have spelled out the command and added a space. The instrument is now waiting for you to enter the slot address of the accessory you want to reset. As an example, to reset the accessory installed in slot 3 of the mainframe, press:



www.valuetronics.com

Pressing:

executes the command.

If you want to reset the instrument while it is currently performing a measurement, you will need to clear the instrument first to regain front panel control and then press RST.

In addition to CLEAR and RST, LOCAL and SRQ are executed as the key is pressed. They are not entered (pressing ENT afterwards) as are the other commands. Be careful when executing the ADDR? (HP-IB address) command. Ensure that the keyboard is in the shift-lock mode (SHIFT LOCK annunciator will be on) before you press the RST key. Otherwise you may unintentionally reset the instrument.

The RST HARD Command

Another command used to initialize the HP 3852A and the plug-in accessories is the RST HARD command. The syntax of the command is shown below:

RST HARD

or

RESET HARD

Executing the command is equivalent to cycling power - the mainframe executes its power-on sequence (Figure 4-1) and then the mainframe and accessories are set to their power-on state.

Front Panel Familiarization

This section describes the HP 3852A keyboard and how specific characters on the front panel are accessed. This section also describes the display and how it is controlled.

The Keyboard

The HP 3852A keyboard is arranged into a command/character key group and a numeric key group (Figure 4-3). Each key in the command/character key group with the exception of the BLUE key is labeled with an HP 3852A or plug-in accessory command header (e.g. MEAS) or command parameter (e.g. DCV). In addition, the keys in this group serve as dual purpose keys as indicated by the characters printed in blue above the keys. The numeric key group consists of the numbers 0-9, a decimal point (.), a comma (,), an exponent key (E), a plus/minus (+/−) key, and a BACK SPACE key. The numeric key group also contains the ENT key which enters/executes commands originating from the front panel.

www.valuetronics.com

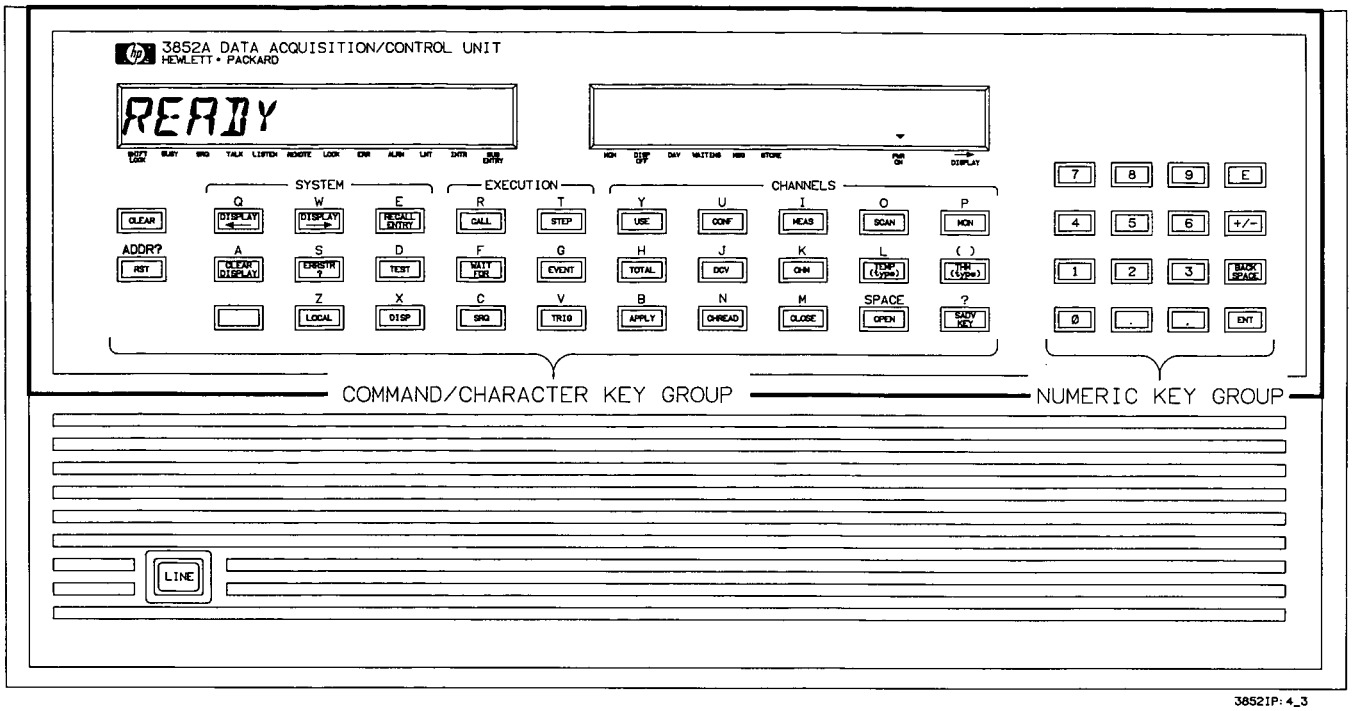


Figure 4-3. HP 3852A Front Panel

The "BLUE" Key

In Figure 4-3 or on the instrument itself, locate the BLUE key on the bottom left corner of the keyboard. When this key is pressed, the SHIFT LOCK annunciator in the display will be on and the keys in the command/character key group correspond to the letter, word, or character printed above them. When the keyboard is in this "shifted" mode, it can be visualized as shown in Figure 4-4. Note that when the BLUE key is pressed, the keyboard essentially becomes a typewriter. The keyboard will remain shifted until the character, letter, or word is entered by the ENT key or until the BLUE key is pressed again.

The key groups depicted in Figures 4-3 and 4-4 are interactive. This means that for various commands in the HP 3852A and plug-in accessories command set, certain parts of the command are entered from the unshifted keyboard while other parts are entered from the shifted key group.

www.valuetronics.com

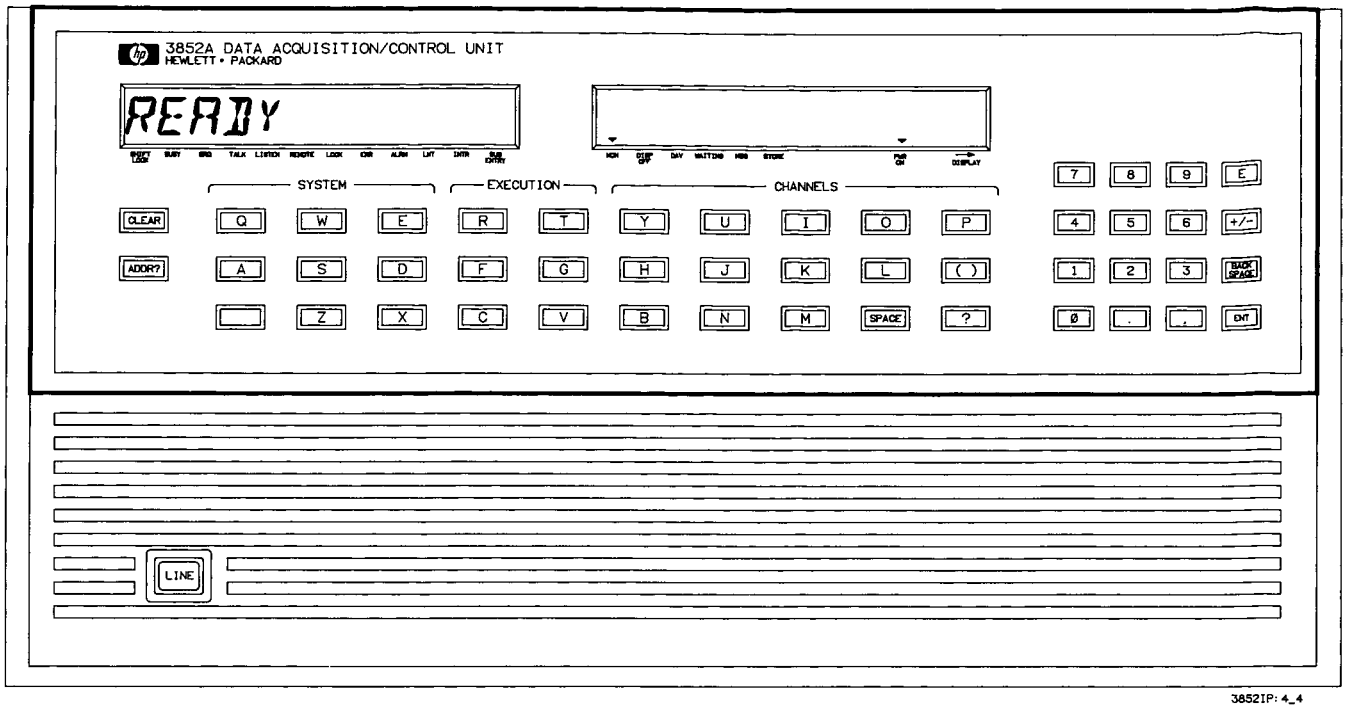
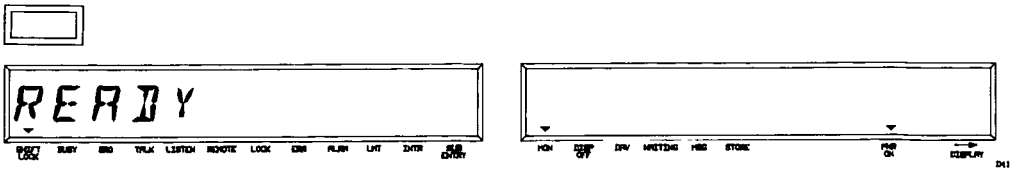


Figure 4-4. The Shifted Keyboard

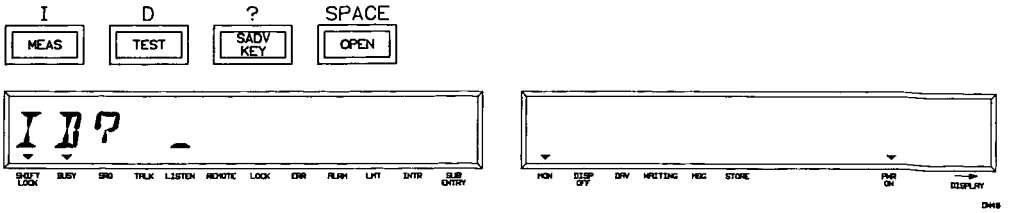
Using the BLUE Key

To familiarize yourself with how the BLUE key functions, let's enter the ID? command. This command is used to identify accessories that are installed in the slots of the mainframe or extenders. However, by not specifying a slot, the HP 3852A will simply identify itself.

Since no key in the command/character key group is labeled with the ID? command, you must literally spell it out. Begin by pressing:

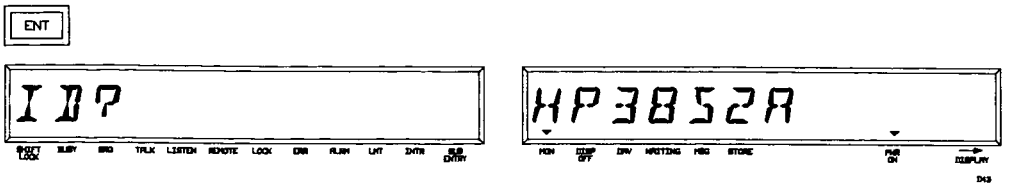


Notice that the SHIFT LOCK annunciator is on indicating that the shifted keyboard is in affect. Next, press:



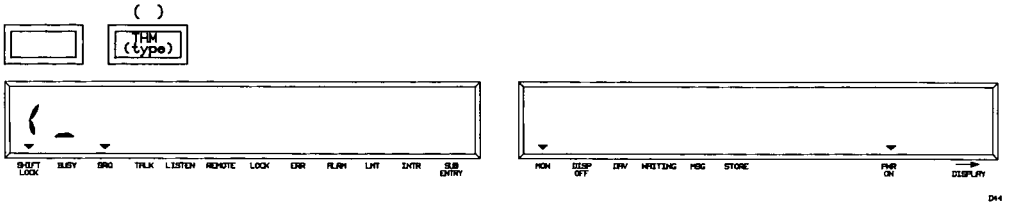
www.valuetronics.com

Here we have spelled out the command. The keyboard is still in the shifted mode and the BUSY annunciator is indicating that the instrument is in the process of receiving a command. The keyboard will remain in the shifted mode until the command is entered by pressing:

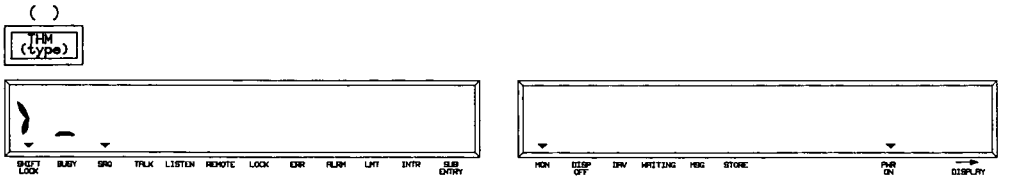


Special Characters: ( )

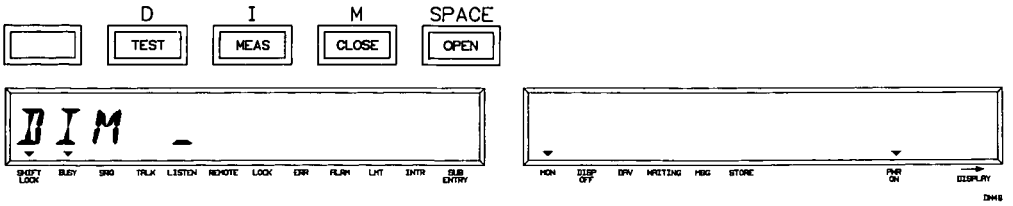
Located on the keyboard above the THM (type) key are a set of parentheses ( ). They are printed in blue indicating that they are accessed from the shifted keyboard. The parentheses are used to specify the size of arrays in the mainframe's internal memory that are set up by the DIM, REAL, INTEGER, and PACKED commands (e.g. REAL DAT(10)). To access the parentheses, press:



to get the left parenthesis, then press:

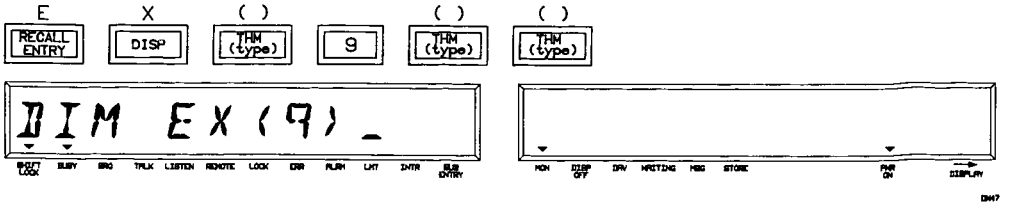


again to get the right parenthesis. Note that pressing THM (type) repeatedly with the keyboard in the shifted mode will alternately give you the left or right parenthesis. For an understanding of how the parentheses are used, let's declare an array in the mainframe's internal memory to store 10 readings in a Real number format. Clear the display then press:



www.valuetronics.com

Here we have spelled out the command DIM and added a space. Next we need to name the array and specify its size. Press:



Pressing:

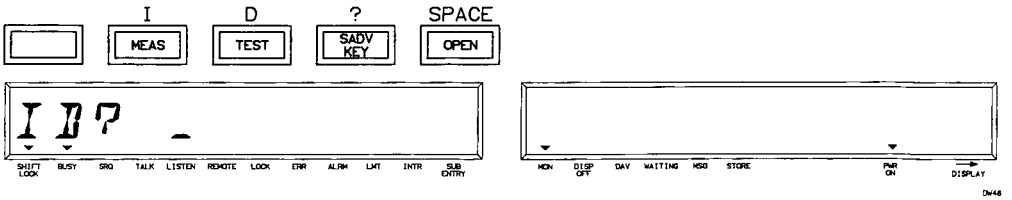
executes the command. (Arrays in the mainframe's internal memory are one-dimensional with a starting index of 0).

Special Characters: SPACE

The commands in the HP 3852A and plug-in accessory command set normally consist of a command header and one or more parameters. The header and the parameters are separated by either a space or a comma. The SPACE key on the shifted keyboard and the comma (,) in the numeric key group provide you both of these delimiters. For an example of using the SPACE key, consider again the ID? command and its format:

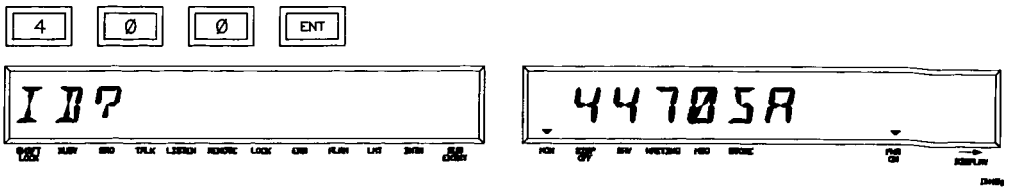
ID? [*slot*]

When a slot number is specified with this command, the product number of the accessory in that slot is returned. For this command, the slot number must be separated from ID? by either a space or comma. To execute the command, press:



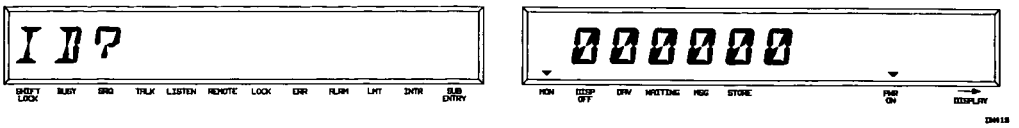
Here from the shifted keyboard we have spelled out I D ? and used the SPACE key to move the cursor one space to the right. The keyboard is still in the shifted mode and the BUSY annunciator is indicating that the instrument is in the process of receiving a command.

We can now specify a slot number to determine the accessory installed in that slot. Assume we want to know the identity of the accessory installed in slot 4 of the mainframe. Press:



www.valuetronics.com

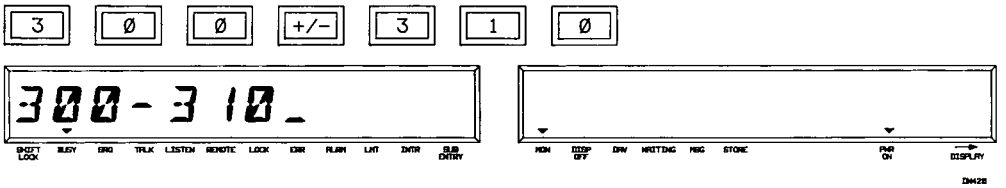
In this example, an HP 44705A 20-Channel Relay Multiplexer is installed in the slot. Had there been no accessories installed, the mainframe would have displayed:



Special Characters: +/–

Another character of special interest is the +/– key in the numeric key group. This key is used to differentiate between positive and negative numbers and is also used when specifying a channel list. Note that numbers entered from the keyboard (or a controller) are assumed to be positive unless prefixed by a minus (–) sign.

When entering a channel list, the minus (–) sign is used to indicate the range of channels. For example, if you wanted to measure channels 0 through 10 on an accessory installed in slot 3 of the mainframe, the channel list would be specified as:



If you keyed in this sequence, press CLEAR DISPLAY. (See Chapter 6 for information on slot and channel addressing.)

Special Characters: The E Key

In addition to specifying an exponent, repeatedly pressing the E key in the numeric key group accesses the following characters:

= * / < > " ; : @ _

These characters enable all HP 3852A commands to be entered from the front panel as well as over the HP-IB. When it becomes necessary to enter any of these characters back-to-back, enter the first character, space over one position, then enter the second character.

Assigning Softkey Definitions

With the HP 3852A you have the ability to assign softkey definitions to keys 0 through 9. Definitions are assigned and deleted using the EDIT KEY and SCRATCH KEY commands. Refer to the Command Reference Manual for additional information and examples on the use of the commands.

www.valuetronics.com

The Displays

The HP 3852A has a 24 character alphanumeric LCD display (colons, periods, and commas are not counted as characters unless used consecutively). The instrument's display buffer can hold up to 60 characters. For displays greater than 24 characters, the extra characters can be viewed using the left arrow or right arrow keys. Notice that there are two display windows. The left display shows commands as they are entered into the mainframe and as they are executed. The left display also shows channel numbers for associated readings during scanning. The right display shows the data returned as the commands are executed.

The Annunciators

Listed in a row underneath the display windows are 20 triangular annunciators. The annunciators indicate the mainframe's present operating condition. The following list describes each of these annunciators reading from left to right:

SHIFT LOCK - indicates that the BLUE shift key has been pressed and the "shifted" keyboard is engaged. This annunciator will remain on until the blue key is pressed again or until the command is entered.

BUSY - indicates that a command is being entered from the front panel or HP-IB, or that the mainframe is executing a command. The BUSY annunciator will also appear when the HP 3852A receives an interrupt signal from an accessory or when an alarm occurs. This annunciator remains on until the command is entered and has finished executing or until the interrupt (or alarm) has been serviced.

SRQ - (service request) indicates that the HP 3852A has requested service from the controller by having pressed the SRQ key on the front panel or having sent the SRQ command. This annunciator is turned off when the service request bit in the mainframe Status Register is cleared by the STB? command or by a serial poll (SPOLL) issued by the controller.

TALK - indicates the HP 3852A is currently addressed to talk (e.g. the controller executed ENTER 709;...).

LISTEN - indicates the HP 3852A is currently addressed to listen (e.g. the controller executed OUTPUT 709;"...).

REMOTE - indicates the HP 3852A is in an IEEE-488 remote state (i.e. programmed by a controller over the HP-IB). This annunciator will appear simultaneously with the LISTEN annunciator. The REMOTE annunciator will turn off when front panel (local) control is restored.

LOCK - indicates that front panel operation of the instrument is "locked out" as a result of the instrument having received the LOCK command.

www.valuetronics.com

ERR - indicates that an error occurred. This annunciator will appear at the same time the error message first appears in the display and remain on until the error message or messages are retrieved from the error buffer by the ERRSTR? or ERR? command.

ALRM - (alarm) indicates an alarm from the real-time clock occurred. This annunciator is turned off by executing the STA? command or by resetting the instrument.

LMT - (limit) indicates that an out-of-limit condition was reached. This annunciator is turned off by executing the STA? command or by resetting the instrument.

INTR - (interrupt) indicates that a backplane (accessory) interrupt occurred. This annunciator is turned off by executing the STA? command or by resetting the instrument.

SUB ENTRY - (subprogram entry) set when the mainframe receives the SUB command. The annunciator remains on while the subprogram is being entered and is turned off when the SUBEND command is executed. Note that it is easy to begin downloading a subroutine and forget to execute the SUBEND command. If commands you continue to enter seem to have no affect, check this annunciator to be sure they are not being entered as a subroutine.

MON - (monitor) occurs at power-on and when the instrument is reset. Indicates whether all, or selected mainframe operations will appear in the display. This annunciator is turned on and off by the MON command.

DISP OFF - (display off) indicates that the display is turned off. This annunciator is turned on by the DISP OFF command and will remain on until the display is turned back on or until the instrument is reset.

DAV - (data available) indicates data is present in the mainframe's HP-IB output buffer. The annunciator turns off when the data is retrieved from the buffer or by clearing or resetting the instrument.

WAITING - indicates that the HP 3852A is waiting for a predefined condition to occur as a result of the WAITFOR command, or it is waiting a predefined period of time as determined by the WAIT command. The annunciator turns off after the condition occurs or after the period elapses.

MSG - (message) indicates that the information in the display is a user-defined message as allowed by the DISP command. The annunciator turns off as soon as the results from another command are displayed.

www.valuetronics.com

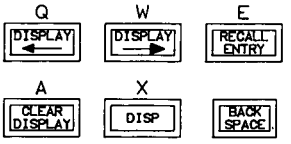
STORE - this annunciator comes on momentarily while data returned by a command is being stored into the mainframe's internal memory.

PWR ON - (power on) indicates the mainframe is turned on. Note that if the display has been turned off by the DISP command, this annunciator is also turned off.

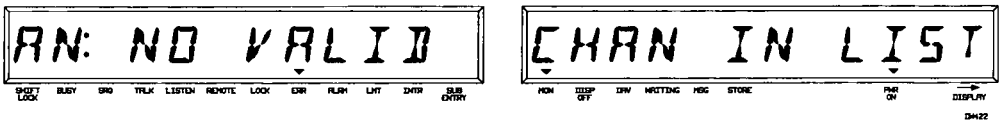
DISPLAY → - indicates there are more characters in the display than are visible through the display windows. Scrolling the display to the left will enable you to see this information. This annunciator turns off when there are no more characters to the right of the display window.

Controlling the Display

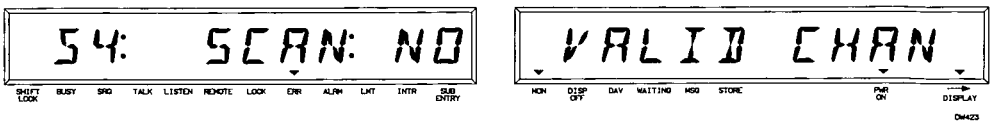
The mainframe display is controlled from the front panel by the following six keys:



Pressing this key scrolls the display in the direction of the arrow (left). During command entry this key will scroll the left display, and scroll both displays when an error message occupies both windows. This key will scroll the right display when the results of a command are shown or when an error message is retrieved from the error buffer.

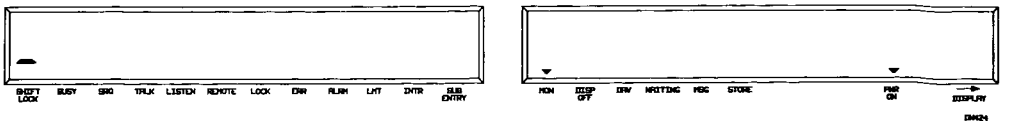


Pressing this key scrolls the display(s) in the direction of the arrow (right) under the same conditions as described above.



Pressing this key will restore the last command executed from the front panel. Use this key to save time when modifying a command.

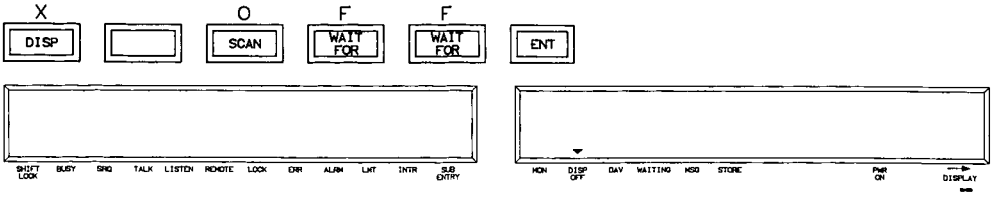
Pressing this key will erase the entire display. Use this key to erase a partially entered command and start again.



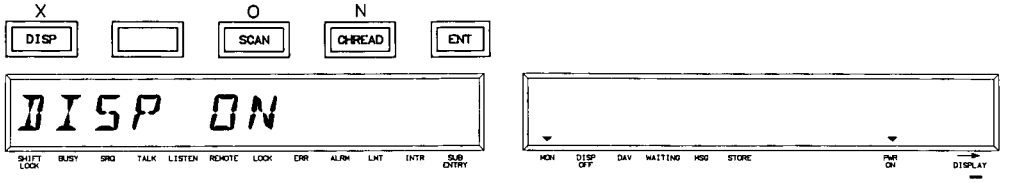
www.valuetronics.com

X
DISP

From the front panel, this key is a command used to turn the display off and on, or display numeric expressions depending on the parameter specified. To use this key (command) to turn the display off, press:



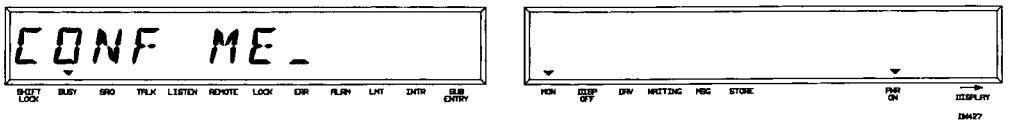
When the display is turned off, you will still see commands and characters as you enter them from the front panel, however, you will not see any data returned. If the display is off when commands are sent over the HP-IB, you will not see the command as it is entered nor will you see the data returned. To turn the display back on, press:



See Recognizing Functions later in this chapter for additional information on the DISP command.

BACK SPACE

Pressing this key will move the cursor in the display to the left, erasing each character it encounters. Commands or characters that are entered start at the position of the cursor. Use this key together with the RECALL ENTRY key to modify commands.



www.valuetronics.com

Entering Commands

This section describes the format in which commands are presented throughout the manuals and how the format is reflected in the individual command statements. The section then shows you how to read and translate the statement into keystrokes to enter the command.

Command Format and Statements

The commands which program the mainframe and the plug-in accessories generally consist of a command header and its associated parameters which have the format, or can be variations of the format:

HEADER *parameter* [*parameter*]

An actual command statement in this particular format appears as:

CONF *function* **[USE** *ch*]

which could then translate to:

CONF DCV, USE 100

In addition to identifying the headers and parameters of a command, the command statement indicates those parameters that are required, and those that are optional.

Headers

In the command shown above, CONF is the command header. Many of the keys on the mainframe front panel (including CONF) are labeled with a command header (e.g. MEAS, SCAN, APPLY). Note that some commands will consist only of the header. Within the command statement (**CONF** *function* **[USE** *ch*]) the header will always appear in capital letters.

Parameters

In the command shown above, *function* and **[USE** *ch*] are the parameters. Parameters can be functions, addresses, conditions (on/off), events, data format specifiers, and variable specifiers. The choices associated with each parameter depend on the particular command.

Parameters not enclosed in brackets are required parameters and must be specified each time the command is executed. Parameters enclosed in brackets [] are optional parameters. When specifying an optional parameter, the brackets are not included. Note that when an optional parameter is not specified, a default value is often assumed by the system. See Power-on and Default values in the section "How to Enter Commands" before specifying or omitting optional parameters.

The parameters in a command statement usually appear in lower case italics. Certain words that are always included with a particular parameter (e.g. **USE, INTO**) are shown in upper case bold.

www.valuetronics.com

Delimiters: Spaces, Commas, Semicolons

Delimiters are used within a command to enable the mainframe and a controller to recognize the header and parameters of a command, and distinguish one command from another. When entering commands into the mainframe, the header and parameters are separated by either a space, comma, or carriage return. The mainframe will accept commands with successive spaces, commas, carriage returns, or combinations nested within them.

The semicolon (;), line feed (LF), and EOI (on HP-IB) are end of command delimiters and are used to separate multiple commands that are together in one command string. The ENT key also terminates commands and is the only way to signal the end of a command from the front panel.

How to Enter Commands

The keystrokes involved in entering a command depend on the command and the parameters that are specified. Recall that the front panel keys are labeled with command headers and parameters, and when the blue key has been pressed, the keys correspond to those on a typewriter.

The flexibility of the front panel provides three methods in which the mainframe and plug-in accessory command set can be entered from the front panel. First, if the command header and parameter are labeled on keys, the keys can be pressed, and then the command entered accordingly. Second, if the header and parameter of a command are not labeled on keys, the keyboard is shifted and the command is entered a character at a time. The third method will involve a command where the header is labeled on a key but the parameter(s) will have to be entered a character at a time from the shifted keyboard. In the examples that follow, each method described above is illustrated. The examples begin with a command statement from which the header and parameters are identified, and the method for entering the command is determined.

CAUTION

In the examples that follow, a brief description of the command depicted in the statement is included. Note, however, the examples are intended only to emphasize keystrokes. They do not necessarily suggest or imply a "preferred" method of performing a function. The examples contain no accessory configuration or field wiring information. To prevent damage to your equipment, all accessory field wiring should be disconnected before attempting to enter any of the example key sequences found throughout this chapter.

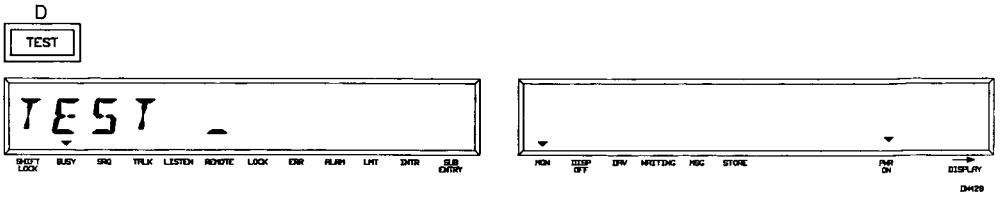
---

www.valuetronics.com

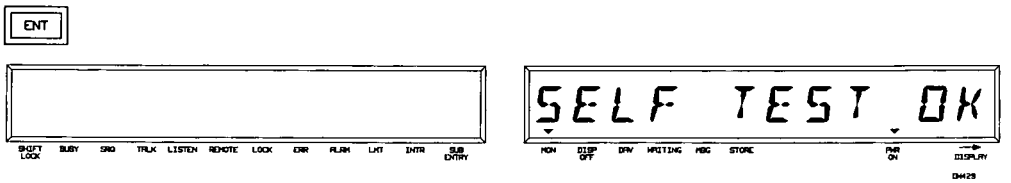
Example 1: Entering a command using labeled keys

TEST [*slot*]

The TEST command initiates a mainframe or accessory self test. From the command statement, we can recognize that TEST is the header and [*slot*] is an optional parameter. Since [*slot*] is an optional parameter, one does not have to be specified. The next step is to determine the keystrokes required to enter and execute the command. On the mainframe front panel is a key labeled "TEST". Therefore, to enter this command, press:



The command is executed by pressing:



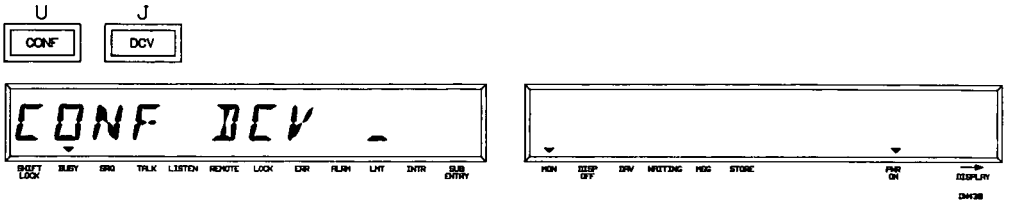
What happened?

- When TEST was pressed, the command was entered into the display.
- When ENT was pressed, the command was executed and the self test was performed.

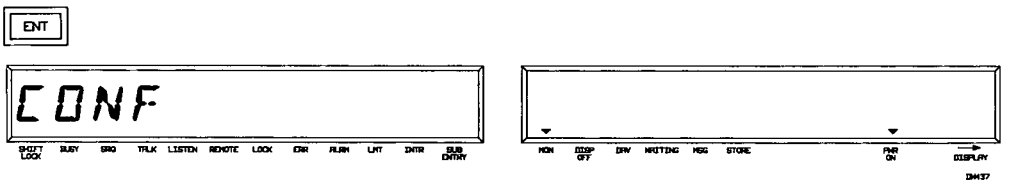
In order to execute this command, two keystrokes were required. For all commands entered from the front panel, the ENT key is used to complete the command entry sequence and execute the command. The only exceptions to this are the CLEAR, RST, SRQ, and LOCAL commands. These commands are entered and executed with a single keystroke.

www.valuetronics.com

The CONF command configures an accessory or accessory channel specified by the **[USE** *ch***]** parameter to measure the function specified by the *function* parameter. We can determine from the command statement that CONF is the command header, *function* is a required parameter, and **[USE** *ch***]** is an optional parameter. To determine the method for entering the command, first look at the front panel to see if the header or any other part of the command is labeled on a key. CONF is labeled on a key and for this example, we'll assume the function we want to measure is DCV, also labeled on a key. In this example, **[USE** *ch***]** will not be specified. In order for this command to be accepted by the mainframe, however, an HP 44701A or HP 44702A/B voltmeter accessory should be installed in slot 0 of the mainframe. Given this condition, the command is entered by pressing:



The command is executed by pressing:



What happened?

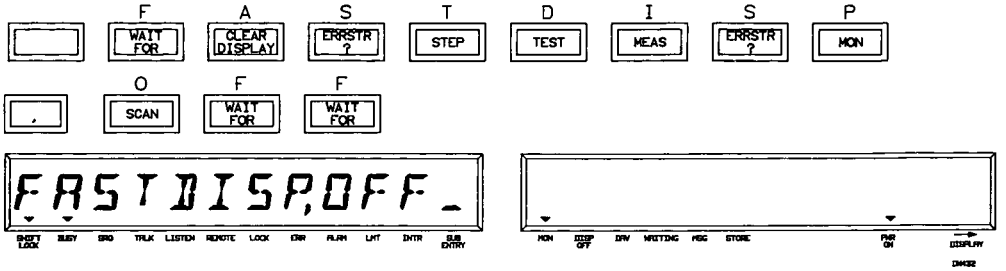
- The command was entered by pressing the keys labeled CONF and DCV. Note that when CONF was pressed, the mainframe recognized this as a command header and automatically included a space before DCV was pressed.
- When ENT was pressed, the command was executed.

www.valuetronics.com

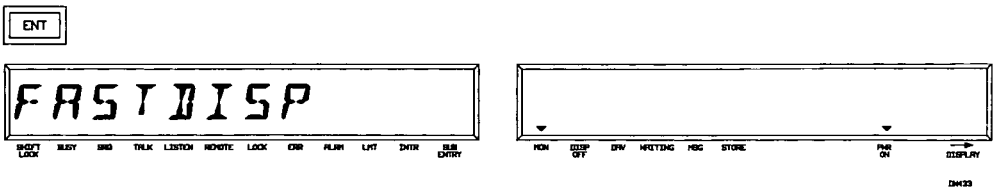
Example 2: Entering a command from the shifted keyboard

FASTDISP [*mode*]

FASTDISP OFF is used to "slow down" the display so that if a command returns several readings, each individual reading can be seen. From the command statement, FASTDISP is the header and [*mode*] is an optional parameter. Looking at the front panel, neither the header nor the parameter choices (ON/OFF) are labeled on a key. The method for entering this command is to press the blue key, then enter the command a character at a time. In this example, we want to turn FASTDISP OFF. Begin by pressing:



To execute the command, press:



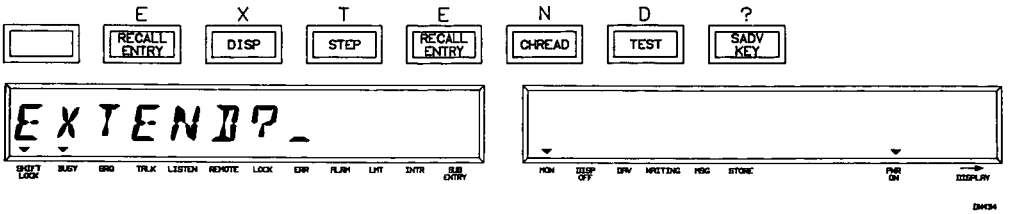
What happened?

- The shifted keyboard was engaged when the blue key was pressed and remained engaged while the command was typed in. A comma was used rather than a space for the delimiter although either could have been used.
- The command was executed when ENT was pressed.

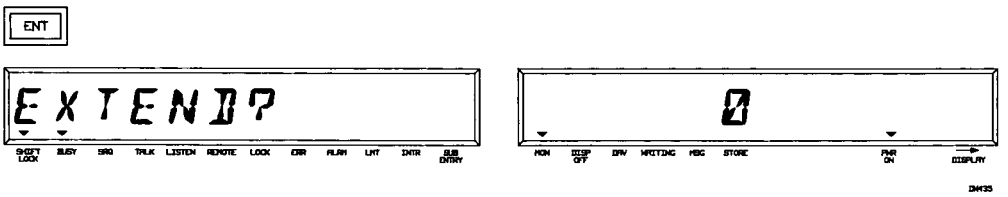
www.valuetronics.com

EXTEND? [INTO *name*] or [*fmt*]

The EXTEND? command asks the mainframe if any extenders are connected, and if so, what their extender number is. From the command statement, EXTEND? is the header and [**INTO** *name*] and [*fmt*] are optional parameters. If [**INTO** *name*] is specified, the data returned by EXTEND? is stored into the array name , previously declared in the mainframe's internal memory. If [*fmt*] is specified, the data is returned to the front panel and/or HP-IB output buffer in the format specified. This example takes advantage of the FASTDISP mode being off (previous command) as the data returned by this command will be seven consecutive numbers: 0's if no extenders are connected, or 0's and the numbers to which the extenders are set. Since EXTEND? is not labeled on a key, the keyboard must be shifted and the command must be entered a character at a time. In this example, neither optional parameter will be specified. To enter the command, press:



To execute the command, press:



What happened?

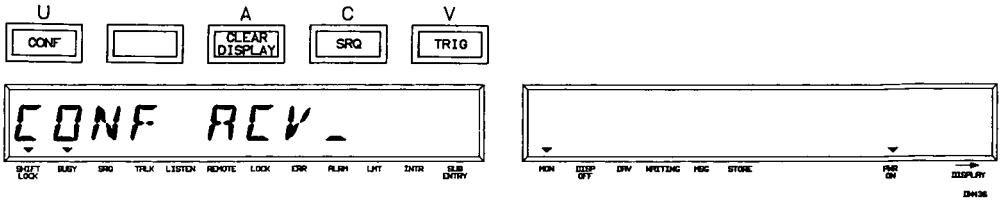
- The command was entered a character at a time.
- The command was executed when ENT was pressed and seven numbers were returned.

www.valuetronics.com

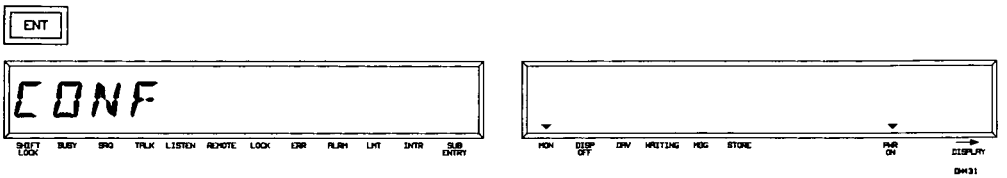
Example 3: Entering a command using labeled and shifted keys

CONF *function* **[USE** *ch***]**

In example 1, the CONF command was used to configure a voltmeter accessory to measure DCV. This example shows how to enter the CONF command when the function is ACV. Since ACV is not labeled on a key, the parameter must be entered using the shifted keyboard. When specifying an unlabeled parameter that is used with a labeled header, the command is entered by pressing:



The command is executed by pressing:

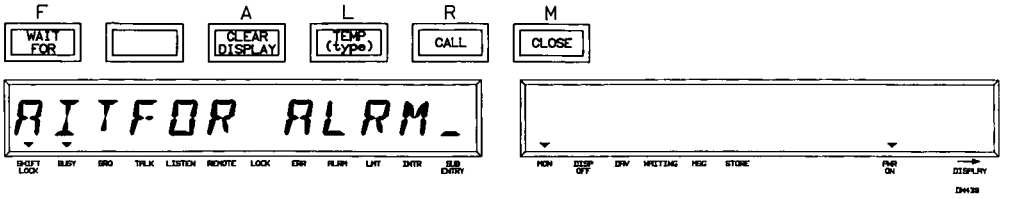


What happened?

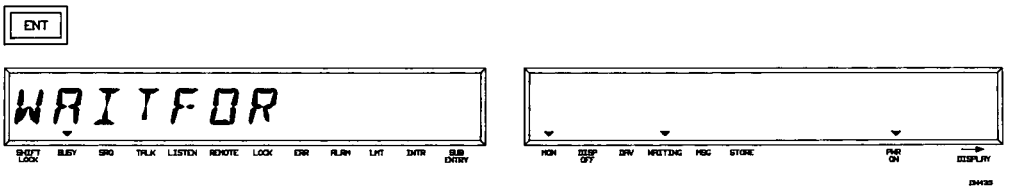
- The command header and space were entered when the key labeled CONF was pressed. The parameter was then entered a character at a time from the shifted keyboard.
- The command was executed when ENT was pressed.

www.valuetronics.com

The WAITFOR command specifies a condition that must occur before another command is executed or before program execution continues. From the command statement, you can determine that WAITFOR is the command header and *condition* is a required parameter. For this example, the *condition* will be an alarm (ALRM) that we'll assume was previously set. A quick glance at the keyboard determines the method in which the command will be entered. The header WAITFOR is labeled on a key, however, the condition must be entered a character at a time. To enter the command, press:



To execute the command, press:



What happened?

- The header and a space were entered into the display, then the parameter was entered a character at a time.
- The command was executed when ENT was pressed.

The busy annunciator indicates the mainframe is waiting for the alarm to occur. When the alarm does occur, command/program execution can continue. At this point, however, press CLEAR to restore normal operation.

www.valuetronics.com

Entering Numbers

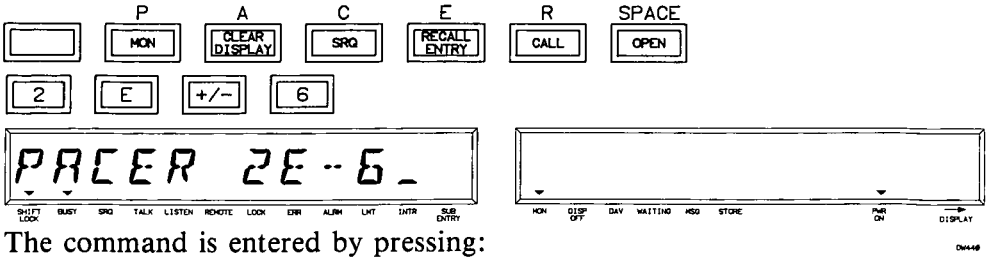
Many commands contain parameters that are numbers (addresses, readings per trigger, period, etc.). Numbers are entered from the front panel using the numeric keys, the exponent key (E), and the +/− key. A parameter requiring a number can be specified as a free field ASCII number, an array, array element, variable, or as a parenthesized numeric expression (similar to those allowed in BASIC) formed by combining numbers or arrays with math functions (+, −, *, /, ^ , PI, ABS, EXP, FRACT, INT, LGT, LOG, SGN, SQR), trigonometric operations (ATN, COS, SIN), or binary functions (BINAND, BINCMP, BINEOR, BINIOR, BIT, ROTATE, SHIFT). When specifying a number in a command, consider that for example:

15
015
00015
15.0
0.15E+2
0.0015E+4
(SQR (225))
(10 + 5)
A (if A has the value of 15)

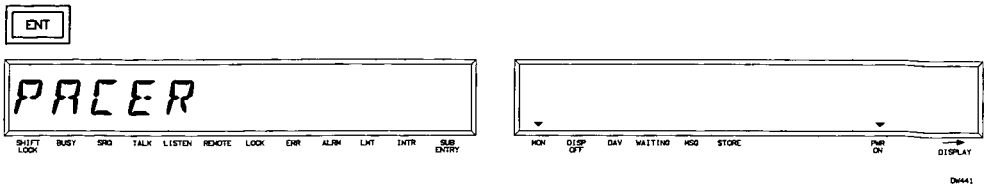
all represent 15.

The PACER command can be used to demonstrate how to enter a number and also demonstrate the use of the E and +/− keys. When triggered, the system pacer outputs from its rear panel BNC connector, a continuous pulse train of 500 ns wide negative-going pulses. The period at which these pulses occur can be set using the PACER command. For example, if we wanted the pulses to occur every 2 μs, we would enter the command and specify the period as follows:

PACER *period* [*count*]



The command is entered by pressing:



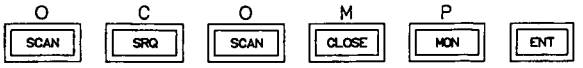
www.valuetronics.com

Power-on and Default Parameters

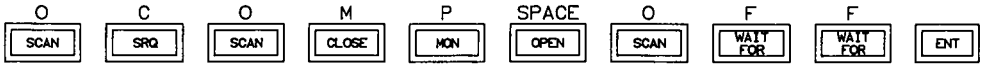
The majority of parameters associated with the command set have predefined power-on and default values. A power-on value is assigned at the time the instrument is turned on or when the instrument is reset (RST). A default value is assumed when an optional parameter is not specified directly. Power-on and default values are often not the same. To determine when a parameter's power-on value or default value will be used, consider the command:

OCOMP [*mode*] [**USE** *ch*]

In this command, *mode* is either OFF or ON. These states happen to correspond to the power-on and default states respectively. By entering the command as:



we do not specify the *mode* parameter directly, therefore, the parameter was defaulted and the "ON" state is assumed. If the command is entered as:



mode is specified directly, and in the case of OCOMP OFF, this is the power-on value.

Keep in mind that [**USE** *ch*] is also an optional parameter of the OCOMP command. However, if this parameter is not specified, the parameter defaults to the "USE channel" set up previously by the USE command. This may or may not have been the USE channel set at power-on.

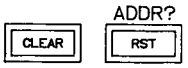
Recognizing Functions

The purpose of this section is to help you recognize HP 3852A and system capabilities based on front panel key groups and specific key labels. This section explains the "SYSTEM", "EXECUTION", and "CHANNELS" keys, and as functional key groupings are identified, individual keys are identified as a command header or command parameter. In certain instances, the same key will appear in different key groups. This is because many instrument functions are not performed by a single command, but are set up and executed by a series of commands. Note that the plug-in accessories most commonly programmed by the key groups are also identified.

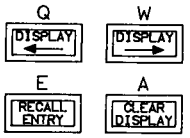
www.valuetronics.com

SYSTEM Keys

SYSTEM keys are essentially maintenance keys. They are used to initialize and test the mainframe and its systems, control the keyboard and display, and monitor operating and programming errors.



These keys initialize the mainframe and the system. Unlike the other commands, these commands (keys) are executed with a single keystroke. The CLEAR and ReSeT functions are described in a previous section of this chapter.



These keys control the display. They do not correspond to a command header or parameter. They maintain control over the display during front panel operation and DISPLAY ← and DISPLAY → also respond during remote (HP-IB) control. Display control is also covered in a previous section of this chapter.

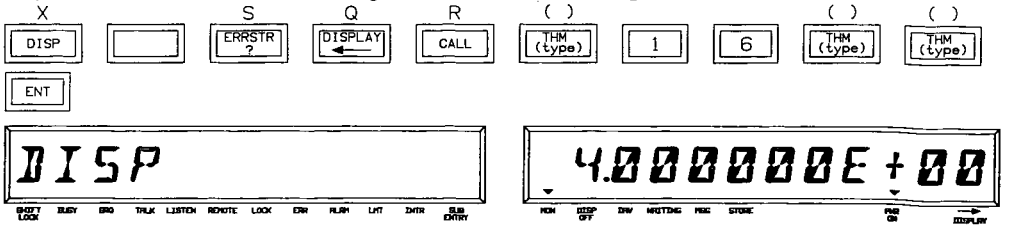
This key/command is used to retrieve error messages from the mainframe's error buffer. The error buffer and the ERRSTR? command are covered in a previous section of this chapter.

This key/command initiates a system or accessory self test. The self test routine and sequence is described in Figure 4-2 of this chapter.

This key engages the shifted keyboard. When engaged, the keys on the front panel correspond to the letters or characters printed in blue above them.

This key/command is used to transfer control of the mainframe from a controller to the front panel and is executed with a single keystroke. This key remains functional during HP-IB operation unless the keyboard has been disabled by the LOCK command or by the HP-IB Local Lockout (LLO) command.

This key/command performs a variety of different functions depending on the parameter specified. This command is used to turn the display off and on, and evaluate numeric expressions. For example:



The DISP command is also used to display a user-defined message.

www.valuetronics.com

EXECUTION Keys

EXECUTION keys synchronize system operation through the triggering, interrupt, and execution functions they represent.

These keys/commands represent the mainframe's ability to store and execute HP 3852A subroutines. The CALL key/command calls and executes previously stored subroutines. The STEP key/command also calls a previously stored subroutine, however, the subroutine is executed one line at a time as you "step" through it. Chapter 9 shows you how to create, store, and call a subroutine.

The WAITFOR key/command is used to specify an event that must occur before the next command is executed or before program execution will continue. The WAITFOR command is often used within a subroutine.

The EVENT key/parameter is a parameter of the WAITFOR command. If EVENT is the specified parameter in the command, program execution will pause until the EVENT key is pressed, or until a pulse occurs on the rear panel EVENT IN BNC.

This key/command is executed by a single keystroke. If properly enabled, pressing this key will generate an interrupt which sends a service request message to the controller. Chapter 8 covers interrupts including the programmed service request (SRQ).

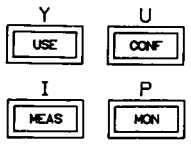
This key/command is used to set the trigger source and/or trigger various plug-in accessories. Refer to the appropriate plug-in accessory manual for detailed information on triggering a specific accessory.

CHANNELS Keys

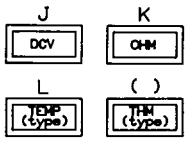
CHANNELS keys are keywords and parameters of commands that perform a function involving a specified accessory or accessory channel.

This key/command/parameter selects the accessory that will perform the specified function, or selects the channel on which the specified function will be performed. Depending on the command, USE is either a command header or a command parameter.

www.valuetronics.com



These keys/commands represent the methods in which analog measurements are made. The USE key specifies the accessory that is to perform the measurement, the CONF command initializes the accessory specified to HP-defined default conditions suitable for the measurement, and the MEAS command actually initiates the channel scan and performs the measurement. The MON key is used with the MEAS key to form the MONMEAS command which in turn takes continuous readings on the accessory channel specified. Additionally, the MON command can be used to select a single channel within a list of channels whose readings will be the only ones monitored.



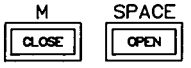
These keys/parameters are parameters of the CONF, MEAS, and MONMEAS commands listed above. You can append an "F" to the OHM parameter to specify a four-wire ohms measurement. The TEMP(type) and THM(type) parameters specify certain thermocouples and thermistors respectively (parentheses are not included).

These keys/commands are for use with the voltmeter accessories or with a voltmeter that is external to the mainframe or extender. The SCAN command sequences through a list of accessory channels enabling the voltmeter to take one or more readings on each channel. If configured, the SADV KEY manually sequences through the specified list of channels as the key is pressed.

This key/parameter is a parameter of the CONF and FUNC commands. This parameter sets the totalizing function of the HP 44715A.

This key/command is used with the digital-to-analog converter (DAC) accessories. When this command is entered, user specified voltages or currents are applied to the channels of the DAC.

This key/command is used to read the results of measurements, inputs, and counts occurring on channels specified by the USE command.



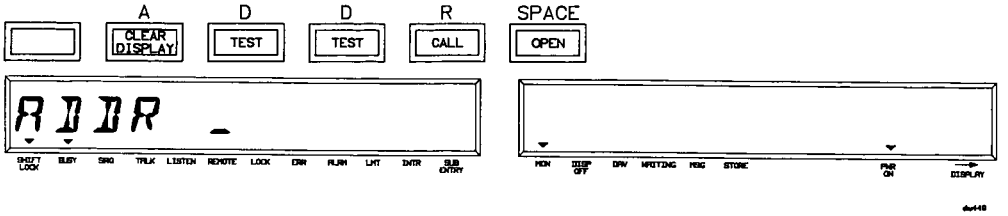
These keys/commands are used to open and close channels on the multiplexer and digital output accessories. A "closed" channel may be used to route input signals to a voltmeter or output a control signal to the user's application.

www.valuetronics.com

"Front Panel Only" Command

The only command in the command set that must be executed from the front panel is the ADDR command which is used to set the mainframe's HP-IB address. The HP-IB address is set to 9 at the factory, however, you can set the address to any number between 0 and 30.

To set the address, you will need to enter the command a character at a time from the shifted keyboard. To do this, press:



the mainframe is now waiting for its address. Enter the desired address using the numeric keys then execute the command by pressing:

The address is now stored in nonvolatile memory and is not lost when the instrument is cleared, reset, or when the power is cycled.

www.valuetronics.com

Contents

www.valuetronics.com

Chapter 5

HP-IB Communication

Introduction

The HP 3852A can be programmed from the front panel or from a controller over the Hewlett-Packard Interface Bus (HP-IB). The HP-IB is Hewlett-Packard's implementation of IEEE Standard 488-1978 and ANSI MC1.1. The HP-IB is the communication link between the HP 3852A and the controller. The interface carries data plus control and programming information between the HP 3852A, the controller, and any other devices connected to the bus.

This chapter explains how to send commands to the HP 3852A and receive data from the HP 3852A over the interface. To do this, the chapter has been divided into three sections: Sending Commands, Entering Data, and HP-IB Overview. An overview of each section is given below.

- **Sending Commands** reviews how to view and set the HP 3852A's HP-IB address and shows how commands are sent. This section also describes the HP 3852A's command input buffer and how it is used.

- **Entering Data** shows how data from the HP 3852A is entered into the controller. The section describes the HP 3852A's output buffer and how data in the buffer can be overwritten or appended to.

- **HP-IB Overview** describes the differences between remote and local operation of the mainframe and also describes the interface functions implemented into the design of the HP 3852A.

NOTE

This chapter shows how the HP 3852A responds to HP-IB and HP 3852A commands issued from an HP Series 200/300 controller. Program examples in this chapter and throughout the manual use syntax applicable to these controllers. Modify the syntax as required for your controller.

---

www.valuetronics.com

Sending Commands

This section covers the addressing requirements of the HP 3852A, the HP 3852A command format, how commands are sent to the HP 3852A, and command buffering.

HP-IB Address Requirements

In order to talk to the HP 3852A from a controller, you must know the HP 3852A's HP-IB address. The complete HP-IB address is a combination of the Interface Select Code (ISC) and Device Address (DA) in the form: (ISC x 100) + DA.

The ISC is determined by the address of the HP-IB interface card on your controller and is usually a number between 1 and 12. In most HP computers, the ISC for an HP-IB interface is factory-set to 7. However, you should refer to the operating manual of your computer to verify the setting.

The device address for the HP 3852A is factory set to 9. The address can be viewed and changed using the ADDR? and ADDR comands. These commands have the syntax shown below:

ADDR? [INTO *name*] or [*fmt*]

ADDR *number*

The simplest method of reading the HP-IB address is to press the blue (shift) key followed by the RST and ENT keys. When ADDR? is entered, the HP-IB address appears in the mainframe's right display window. (The INTO name and fmt parameters indicate that the address returned can be stored in mainframe memory or returned in a particular data format. See Chapter 6.)

ADDR which is used to change the HP-IB address can only be executed from the mainframe's front panel. Refer to "Connecting the HP-IB" in Chapter 3 for information on how the address is set and the range of addresses that can be used. That section also explains how the HP-IB cable is connected.

Throughout this chapter and the entire manual set, all programming examples use an ISC of 7 and a DA of 9. Thus, the HP-IB address used in all examples is (7 x 100) + 9 = 709.

HP 3852A Command Format

The commands which program the mainframe and the plug-in accessories generally consist of a command header and its related parameters which have the format, or can be variations of the format:

HEADER *parameter* [*parameter*]

www.valuetronics.com

An actual command in this particular format appears as:

CONF *function* **[USE** *ch***]**

which could then translate to:

CONF DCV USE 100

Command headers can be entered in upper or lower case. In command statements appearing in the manuals, command headers are shown in upper case bold. Parameters can also be entered in upper or lower case, however, in command statements, parameters appear in lower case italics. Certain words that are always included with a particular parameter (e.g. **USE**, **INTO**) are shown in upper case bold. Brackets [] around a parameter indicate the parameter is optional and, therefore, does not have to be specified in the command.

Sending the Command

Using an HP Series 200/300 controller, commands are sent to the HP 3852A using the controller's OUTPUT statement. The OUTPUT statement may specify a device selector (i.e. HP-IB address) or an I/O path name as shown in the following examples.

```
10    OUTPUT 709;"RST"
20    OUTPUT 709;"FASTDISP OFF"
30    OUTPUT 709;"IDN?"
.
.
.
```

```
10    ASSIGN @Hp_3852 TO 709
20    OUTPUT @Hp_3852;"RST"
30    OUTPUT @Hp_3852;"FASTDISP OFF"
40    OUTPUT @Hp_3852;"IDN?"
.
.
.
```

Throughout this chapter and the entire manual set, example programs, program segments, and program lines specify a device selector (709).

Delimiters

Delimiters are used within a command to enable the mainframe to recognize the header and parameters of a command and distinguish one command from another. When sending a command to the mainframe, the header and parameters must be separated by one or more spaces or commas (,).

www.valuetronics.com

Similar to the ENT key which signals the end of a command entered from the front panel, the HP 3852A accepts the line feed (LF), semicolon (;), and End Or Identify (EOI) signal as the end of a command sent from the controller. These delimiters also enable the mainframe to recognize different commands sent in the same command string. The following program lines illustrate the use of delimiters in programming the HP 3852A.

```
10    OUTPUT 709;"CONFMEAS DCV 400 USE 700"

10    OUTPUT 709;"CONFMEAS,DCV,400,USE,700"

10    OUTPUT 709;"CONFMEAS, DCV, 400, USE, 700"

10    OUTPUT 709;"RQS ON;RQS INTR;RQS?"
```

Mainframe/ Controller Synchronization

With the HP 3852A and a controller you often have parallel processing. This is when the HP 3852A and the controller are simultaneously executing commands. A particularly useful function of command delimiters is that they can help maintain synchronization between the mainframe and controller while both are executing commands. This ensures, for example, that command execution by either the mainframe or controller is held off until the current command finishes, or ensures that the data entered into the controller corresponds to the command that generated the data.

An example of how mainframe/controller synchronization is maintained using delimiters is shown in the following program segments.

```
10    OUTPUT 722;"TRIG HOLD"
20    OUTPUT 709;"INBUF OFF"
30    OUTPUT 709;"CLOSE 100"
40    OUTPUT 722;"TRIG SGL"
.
.
.
```

```
10    OUTPUT 722;"TRIG HOLD"
20    OUTPUT 709;"INBUF OFF"
30    OUTPUT 709;"CLOSE 100;"
40    OUTPUT 722;"TRIG SGL"
.
.
.
```

Both program segments show a measurement involving channel 0 in slot in slot 1 of the mainframe and an external voltmeter (HP-IB address 722). In the segment on the left, the end of command delimiter in each of the program lines is the carriage return/line feed (CR/LF) output by the controller (HP Series 200/300). Note that after the mainframe accepts and begins to execute line 30 (CLOSE), the controller continues through the program by sending a triggering command to the voltmeter (line 40). The voltmeter will then attempt a measurement. If the accessory channnel is not fully closed, an invalid measurement is made. This situation can be prevented by using a delimiter to hold off command execution. This is shown by the segment on the right. In line 30, the semicolon delimiter

www.valuetronics.com

following the channel address (100) is recognized as an end of command delimiter. When the semicolon is received, command execution begins. Note that the controller still issues CR/LF. The mainframe, however, will not accept CR/LF until the current command (CLOSE 100) finishes executing. By holding off CR/LF, the controller is prevented from continuing through the program and triggering the voltmeter before the channel is fully closed. When the mainframe accepts the CR/LF, it is interpreted as a "null" command with no affect on the mainframe and operation continues.

Another method of synchronizing command execution is to monitor the HP 3852A's ready bit (RDY) in the Status Register. The RDY bit is set each time the HP 3852A is not accepting or executing a command or subroutine. An example of this method is shown below.

```
10    OUTPUT 722;"TRIG HOLD"
20    OUTPUT 709;"INBUF OFF"
30    OUTPUT 709;"CLOSE 100"
40    WHILE NOT BIT (SPOLL (709),4)
50    WAIT .01
60    END WHILE
70    OUTPUT 722;"TRIG SGL"
.
.
.
```

Lines 40 through 60 are repeated by the controller until the ready condition is sensed. This in effect, holds off execution of line 70 until the command executed in line 30 has finished executing.

Using delimiters to ensure the proper data is returned is covered in the "Entering Data" section of this chapter.

Command Buffering

Commands sent to the HP 3852A over the HP-IB enter the input buffer. At power-on, the buffer stores only one command and delimiter (CR/LF, semicolon (;), EOI) at a time regardless of the number of commands sent. The additional commands are held on the bus until the current command finishes executing or has been stored if it is a subroutine entry.

With the HP 3852A's INBUF command, you can enable the input buffer so that multiple commands can be stored in the buffer, plus you can set the size of the buffer which varies the number of commands the buffer can hold. The syntax and parameters of the INBUF command are shown below:

INBUF [*mode*] or [*size*]

www.valuetronics.com

mode - enables and disables the input buffer. When *mode* is OFF (power-on), the buffer is disabled and only one command and delimiter enter the HP 3852A at a time. When *mode* is ON, multiple commands and delimiters are stored in the buffer. When the buffer fills, additional commands are held on the bus until room is available. The default setting for *mode* is ON.

size - sets the size of the input buffer. The minimum size that can be specified is 5 bytes and the maximum size depends on the amount of mainframe memory available. Once a size has been entered with INBUF, the mainframe must be reset in order to load the buffer size into the operating system. At this point, the new buffer size takes affect. The *size* parameter is only available with firmware revision 3.0 or greater.

The memory used to increase the size of the buffer is taken from the mainframe memory available to the user for subroutine and data storage. If the buffer size specified is extremely large (e.g. INBUF 4200000), ERROR 24: ARGUMENT OUT OF RANGE is reported on execution of the INBUF command. If the size specified is within the argument range but not enough memory is available, ERROR 1: OUT OF MEMORY - HP-IB BUFFERS/SYMTAB is reported during the next reset. When this occurs, the input buffer is set to 198 bytes, the output buffer is set to 1029 bytes, and the symbol table size (SYMSIZE command) is set to accommodate 150 entries.

Note that any time power is cycled, the size of the input buffer is set to 198 bytes.

HP 3852A/ Controller Deadlock

A deadlock between the HP 3852A and the controller is when the HP 3852A is holding off the controller pending command completion, however, controller action is required before the command can complete. A deadlock situation can occur with the INBUF mode OFF or ON when multiple commands or end of command delimiters are included in a single command line and usually when the data returned by a command fills the output buffer. The following programs illustrate the various situations in which a deadlock occurs, and in each case, how the deadlock can be avoided. For each program, the power-on setting for INBUF is assumed (INBUF OFF).

```
      DEADLOCK
10    DIM D(0:999)
20    OUTPUT 709;"REAL A(999)"
30    OUTPUT 709;"VREAD A"
40    OUTPUT 709;"IDN?"
50    ENTER 709;D(*)
60    PRINT D(*)
70    END
```

```
      REMEDY
10    DIM D(0:999),Iden$(0:3)[17]
20    OUTPUT 709;"REAL A(999)"
30    OUTPUT 709;"VREAD A"
40    ENTER 709;D(*)
50    OUTPUT 709;"IDN?"
60    ENTER 709;Iden$(*)
70    PRINT D(*)
80    PRINT USING "K,/";Iden$(*)
90    END
```


www.valuetronics.com

The program on the left deadlocks because as the VREAD command fills the output buffer, execution of the command is paused until data is somehow removed from the buffer. Since the HP 3852A will not accept another command until the VREAD command completes, the controller is held off and is not able to reach the ENTER statement (line 50). Therefore a deadlock occurs.

The "remedy" program on the right shows the method for preventing this type of deadlock; that is, to always enter the data requested by the particular HP 3852A command into the controller immediately after the command executes. Notice in lines 30/60 of the remedy program that the data returned by an HP 3852A command is entered immediately before the next data generating command (i.e. IDN?) is issued. This not only avoids a deadlock, but ensures that the data entered corresponds directly to the command which generated the data.

```
      DEADLOCK
10    DIM D(0:999)
20    OUTPUT 709;"REAL A(999)"
30    OUTPUT 709;"VREAD A;"
40    ENTER 709;D(*)
50    PRINT D(*)
60    END
```

```
      REMEDY
10    DIM D(0:999)
20    OUTPUT 709;"REAL A(999)"
30    OUTPUT 709;"VREAD A"
40    ENTER 709;D(*)
50    PRINT D(*)
60    END
```

Recall that the previous deadlock remedy was to enter the data into the controller as soon as it becomes available. At first glance, it appears that's what we are doing here. However, note in line 30 of the deadlock program that multiple end of command delimiters are sent. Since the semicolon is accepted as the delimiter, the CR/LF issued by the controller (also a delimiter) is held off until the VREAD command finishes executing. In the meantime, VREAD fills the output buffer. With CR/LF held off, the controller is held off and cannot execute the ENTER statement in order to remove the data from the output buffer which allows the VREAD command to complete. Thus a deadlock occurs.

The remedy for this deadlock situation is to avoid sending multiple end of command delimiters in a single OUTPUT statement if there is a chance the output buffer will be filled.

```
      DEADLOCK
10    OUTPUT 709;"USE 700"
20    OUTPUT 709;"CONF DCV"
30    OUTPUT 709;"TERM EXT"
40    OUTPUT 709;"TRIG SYS"
50    OUTPUT 709;"TRG GET"
60    OUTPUT 709;"CHREAD 700;"
```

```
      REMEDY
10    OUTPUT 709;"USE 700"
20    OUTPUT 709;"CONF DCV"
30    OUTPUT 709;"TERM EXT"
40    OUTPUT 709;"TRIG SYS"
50    OUTPUT 709;"TRG GET"
60    OUTPUT 709;"CHREAD 700"
```


www.valuetronics.com

```
70    TRIGGER 709
80    ENTER 709;D
90    PRINT D
100   END
```

```
70    TRIGGER 709
80    ENTER 709;D
90    PRINT D
100   END
```

Not all deadlocks are the result of data filling the mainframe's output buffer. The programs above set up a voltmeter accessory to make a DC voltage measurement. The voltmeter is to be triggered by the controller. The program on the left deadlocks because of the two end of command delimiters issued in line 60. The program executes as intended until line 60 is reached. Again, as the mainframe accepts the semicolon delimiter, the CR/LF is held off until the CHREAD command executes. However, the CHREAD command does not finish executing until the voltmeter is triggered and a reading is available. While CR/LF is held off, the controller is held off and the command which triggers the voltmeter (line 70) is never executed. This is a deadlock situation completely unrelated to the amount of data in the output buffer.

The remedy to the situation is to use care when sending multiple commands.

Using Input Buffering

Input command buffering may not be desirable for all applications. If used when there is no output data buffering (OUTBUF OFF), the data in the output buffer is overwritten as the buffered commands are executed (at an unknown rate). If command buffering is used when output data buffering is enabled (OUTBUF ON), a reset (RST) or output buffer clearing command (CLROUT) may be executed at the wrong time resulting in a complete loss of data. Generally, command buffering is most useful when the controller is required to do other tasks while the HP 3852A is executing a series of commands.

Entering Data

This section explains how data from the HP 3852A is entered into the controller. The section also covers output data buffering, data rates, and how to suppress the End Or Identify signal concurrent with the last byte of data returned by an individual command.

Entering Data into the Controller

Using an HP Series 200/300 controller, data is entered from the HP 3852A's output buffer into the controller using the controller's ENTER statement. The ENTER statement may specify a device selector (i.e. HP-IB address) or an I/O path name as shown in the following examples.

www.valuetronics.com

```
10  OUTPUT 709;"ID?"
20  ENTER 709;Identity$
30  PRINT Identity$
.
.
.
```

```
10  ASSIGN @Hp_3852 TO 709
20  OUTPUT @Hp_3852;"ID?"
30  ENTER @Hp_3852;Identity$
40  PRINT Identity$
.
.
.
```

Data can also be entered into the controller using the TRANSFER statement.

```
10  INTEGER Bdata(0:9) BUFFER
20  ASSIGN @Hp_3852 to 709
30  ASSIGN @Bdata TO BUFFER Bdata(*);FORMAT OFF
40  OUTPUT @Hp_3852;"CONFMEAS DCV 0-9 USE 100 PACK"
50  TRANSFER @Hp_3852 TO @Bdata;END,WAIT
.
.
.
```

Throughout this chapter and the entire manual set, example programs, program segments, and program lines entering data use the ENTER statement and a device selector (709).

Data Buffering

If the data generated by a command sent over the HP-IB is not stored in the HP 3852A's internal memory, the data is returned to the output buffer. The output buffer can be configured such that the data returned will overwrite any data currently in the buffer or be appended to the data which already exists. The size of the output buffer can also be changed to accommodate commands generating large amounts of data.

If the buffer fills while a command is executing, execution is paused until enough data is removed from the buffer to allow the command to continue. If multiple commands are being sent when the buffer fills, a deadlock condition may occur (see HP 3852A/Controller Deadlock).

The command which sets the buffering mode and the size of the output buffer is the OUTBUF command. The syntax of the command is given below:

OUTBUF [*mode*] or [*size*]

www.valuetronics.com

mode - causes data to overwrite, or be appended to the data currently in the buffer. When *mode* is ON, data sent to the buffer is appended to the data currently in the buffer. The default setting for *mode* is ON. When *mode* is OFF, data sent to the buffer overwrites data in the buffer from the previous command. At power-on, mode is OFF. Note that the data returned by a command has a four-byte overhead.

size - sets the size of the output buffer. The minimum size that can be specified is 5 bytes. The maximum size depends on the amount of mainframe memory available. Once a size has been entered with the OUTBUF command, the HP 3852A must be reset in order to load that value into the operating system. At that point, the new size takes affect. The *size* parameter is only available with firmware revision 3.0 or greater.

The memory used to increase the size of the buffer is taken from the mainframe memory available to the user for subroutine and data storage. If the buffer size specified is extremely large (e.g. OUTBUF 4200000), ERROR 24: ARGUMENT OUT OF RANGE is reported on execution of the OUTBUF command. If the size specified is within the argument range but not enough memory is available, ERROR 1: OUT OF MEMORY - HP-IB BUFFERS/SYMTAB is reported during the next reset. When this occurs, the output buffer is set to 1029 bytes, the input buffer is set to 198 bytes, and the symbol table size (SYMSIZE command) is set to accommodate 150 entries.

Note that any time power is cycled, the size of the output buffer is set to 1029 bytes.

Potential Data Errors

If the output buffering mode is off and the controller is sending multiple commands which return data, the actual data entered by the controller may be invalid or a controller error may occur. Refer to the following example.

```
10  OUTPUT 709;"IDN?"
20  OUTPUT 709;"ID?"
30  ENTER 709;Identity$
40  PRINT Identity$
50  END
```

In this program, the controller sends the HP 3852A two commands (lines 10 and 20) which generate data. The IDN? command returns the following:

```
HEWLETT PACKARD (company name)
3852A           (model number)
0               (serial number - unknown)
2.2             (firmware revision)
```

The ID? command returns:

```
HP3852A
```


www.valuetronics.com

Because a string variable is specified in line 30, the controller is attempting to enter the data returned by the ID? command. However, due to the speed at which the data is entered into the controller (line 30) relative to when the ID? command executes, the HEWLETT PACKARD portion of the IDN? command is returned rather than the data generated by ID?. Similarly, if the program is executed as:

```
10  DIM Identity$(0:3)[17]
20  OUTPUT 709;"ID?"
30  OUTPUT 709;"IDN?"
40  ENTER 709;Identity$(*)
50  PRINT USING "K,/";Identity$(*)
60  END
```

the controller will probably display an error message indicating the data returned was not appropriate for the array declared, since the data in the buffer at the time the ENTER statement occurs corresponds to ID? rather than IDN?.

Avoiding Data Errors

Three ways to prevent the data errors described above and ensure the data returned corresponds to the right command are given below. These methods are shown by modifying the two previous programs. Note that the first two methods can also prevent a deadlock condition should a command's data fill the output buffer (see HP 3852A/Controller Deadlock).

Enter the Data Immediately

```
10  DIM Ident$(0:3)[17]
20  OUTPUT 709;"IDN?"
30  ENTER 709;Ident$(*)
40  OUTPUT 709;"ID?"
50  ENTER 709;Identity$
60  PRINT USING "K,/";Ident$(*)
70  PRINT
80  PRINT Identity$
90  END
```

The method shown here is to enter the data into the controller as soon as it is generated. This clears the buffer before the next data-generating command (ID?) is sent.

www.valuetronics.com

Wait for Command Completion

```
10  DIM Identity$(0:3)[17]
20  OUTPUT 709;"ID?"
30  OUTPUT 709;"IDN?"
40  WHILE NOT BIT(SPOLL (709),4)
50  WAIT .01
60  END WHILE
70  ENTER 709;Identity$(*)
80  PRINT USING "K,/";Identity$(*)
90  END
```

Recall that the intent of this program was to enter the the data returned by the IDN? command. In this program, the controller waits until the HP 3852A has finished executing IDN? by monitoring the "ready" state within the mainframe (lines 40/60). Once the command finishes executing and a ready condition is sensed, the data is entered (line 70). This again ensures that the data entered from the buffer is the data returned by IDN?.

Use Multiple End Of Command Delimiters

```
10  OUTPUT 709;"IDN?"
20  OUTPUT 709;"ID?;"
30  ENTER 709;Identity$
40  PRINT Identity$
50  END
```

```
10  DIM Identity$(0:3)[17]
20  OUTPUT 709;"ID?"
30  OUTPUT 709;"IDN?;"
40  ENTER 709;Identity$(*)
50  PRINT USING "K,/";Identity$(*)
60  END
```

To ensure that the desired data is returned, a semicolon end of command delimiter is included following the command whose data is the intended data returned (lines 20 and 30 of the respective programs). Recall that when the HP 3852A receives the semicolon, command execution begins. However, the carriage return/line feed (CR/LF) also issued by the controller is held off until the command (ID? or IDN?) completes. Again, this prevents the controller from executing the ENTER statement (lines 30 and 40) until these commands complete and their data is in the buffer.

This method of avoiding data errors should only be used if you are certain the data returned by a command will not fill the output buffer. This will prevent a deadlock.

www.valuetronics.com

The CLROUT Command

In addition to clearing or resetting the system (CLR and RST commands), the CLROUT command also clears the HP 3852A's output buffer. This command has the syntax:

CLROUT

Use caution when sending the CLROUT command if input command buffering is used. Since buffered commands are executed at an unknown rate, the time at which the buffer is cleared is also unknown.

Entering Data with OUTBUF ON

With OUTBUF ON, the data returned by HP 3852A commands is appended to the data currently in the buffer. As the controller enters data from the buffer, the HP 3852A asserts the End Or Identify (EOI) signal concurrent with the last byte of data corresponding to each individual command. Although there may be data from several commands in the buffer, only one command's worth of data is entered into the controller per ENTER statement, due to EOI. Refer to the following example.

```
10   DIM Dcrdgs(0:9)
20   OUTPUT 709;"OUTBUF ON"
30   OUTPUT 709;"USE 700"
40   OUTPUT 709;"CONFMEAS DCV 600-604"
50   OUTPUT 709;"RANGE 10"
60   OUTPUT 709;"MEAS DCV 605-609"
70   ENTER 709;Dcrdgs(*)
80   END
```

This program enables output buffering then performs a series of DC voltage measurements. As the program executes, DC voltage is measured on channels 0 through 4 and stored in the output buffer. The range is then changed and the voltage is measured on channels 5-9. This data is appended to the data (channels 0-4) already in the buffer. The ENTER statement in line 70 attempts to enter both sets of data into controller array Dcrdgs. Note, however, that as the data is read, the mainframe asserts EOI when the last byte of data returned by CONFMEAS is read. This terminates the entry after only five readings. Since the controller is expecting 10 readings (DIM Dcrdgs (0:9)), the controller reports an error condition and none of the readings are entered.

To enter all of the data generated in the previous program, the program must be modified as shown below.

```
10   DIM Dcrdgs(0:4)
20   DIM Dcrdgs1(0:4)
30   OUTPUT 709;"OUTBUF ON"
40   OUTPUT 709;"USE 700"
50   OUTPUT 709;"CONFMEAS DCV 600-604"
60   OUTPUT 709;"RANGE 10"
70   OUTPUT 709;"MEAS DCV 605-609"
80   ENTER 709;Dcrdgs(*)          !Enters DCV on channels 0-4
90   ENTER 709;Dcrdgs1(*)         !Enters DCV on channels 5-9
100  END
```


www.valuetronics.com

The END Command

The EOI signal asserted by the HP 3852A when the last byte of a command's data is read is the reason multiple ENTER statements are required. For HP 3852A's with mainframe firmware revision 2.2 or greater, the EOI signal can be suppressed by the END command. By suppressing the EOI signal, the data from multiple commands can be entered into the controller with a single ENTER statement.

The END command has the syntax:

END *mode*

The *mode* parameters are OFF and ON. When the mode is OFF, the EOI signal is suppressed and the data from several commands can be entered into a single controller array with a single ENTER statement. This is shown in the following program.

```
10  DIM Dcrdgs(0:9)
20  OUTPUT 709;"OUTBUF ON"
30  OUTPUT 709;"END OFF"
40  OUTPUT 709;"USE 700"
50  OUTPUT 709;"CONFMEAS DCV 600-604"
60  OUTPUT 709;"RANGE 10"
70  OUTPUT 709;"MEAS DCV 605-609"
80  ENTER 709;Dcrdgs(*)
90  END
```

Recall that when this program was executed without the END command (line 30), the controller was not able to enter all of the readings because EOI was asserted following the data generated by CONFMEAS (line 50). However, since END suppresses EOI, the ENTER statement in line 80 enters both sets of readings.

At power-on or any time END ON is set, EOI is asserted with the last byte of data returned by each command.

NOTE

The firmware revision of your HP 3852A can be determined with the IDN? command.

---

www.valuetronics.com

Output Data Rates

For most HP-IB configurations, the maximum output data rate by the HP 3852A is 120 kbytes/sec. With the FASTOUT command and a specific HP-IB configuration, the output data rate can be increased to approximately 140 kbytes/sec. The syntax of the FASTOUT command is

FASTOUT [*mode*]

The *mode* parameters are OFF and ON. With the mode OFF, the data rate is 120 kbytes/sec. With the mode ON, the output data rate is 140 kbytes/sec for the following HP-IB bus conditions:

- Power to all devices on the bus
- < 50 pF capacitive loading at each device
- ≤ 15 metres of HP-IB cable
- At least one device load per metre of cable

HP-IB Overview

This section explains the differences and the transfer between remote and local control of the HP 3852A. This section also covers the HP-IB interface functions designed into the HP 3852A.

Remote/ Local Control

At power-on and during the time the HP 3852A is operated from the front panel, the mainframe is said to be under local (front panel) control. When you have local control of the mainframe, all HP 3852A commands can be entered from the front panel.

When the mainframe is operated from a controller via the HP-IB, the mainframe is under remote control. The transfer from local to remote control occurs when REN (remote enable) is asserted and an HP 3852A command is sent over the HP-IB. The HP 3852A also enters the remote state when the controller sends its REMOTE (or equivalent) statement together with the HP 3852A's HP-IB address (e.g. REMOTE 709). When the HP 3852A is in the remote state, only a subset of the HP 3852A's commands can be entered from the front panel. These commands are shown in Table 5-1.

www.valuetronics.com

Table 5-1. "Front Panel" Commands Allowed in Remote

ADDR?	EXTEND?	MONITOR	SUB	WHILE...END WHILE
ALRM	FASTDISP	NEXT	SUBEND	
BEEP	ID?	PAUSE	TIME	
CAT	IDN?	POWEROFF	TIMEDATE	
CLOSE?	IF...END IF	RQS?	USE?	
DISP	INDEX?	SIZE?	VREAD	
ERR?	INTR?	SRQ	WAIT	
ERRSTR?	LOCAL	STATE?	WAITFOR	

Restoring Local Control

Local control of the HP 3852A is restored with the LOCAL command. This command has the syntax:

LOCAL

LOCAL can be sent from the controller:

```
OUTPUT 709;"LOCAL"
```

or executed by pressing the LOCAL key on the front panel. The LOCAL key is an "immediate execute" key which means the command executes as the key is pressed.

There are two conditions under which the LOCAL command (or key) will not "directly" restore front panel control. The first condition is when the keyboard has been locked out by the HP 3852A's lock command. The second condition is when the HP 3852A is in the remote with lockout state (RWLS). Both conditions are discussed below.

The LOCK Command

The HP 3852A's LOCK command is used to disable the mainframe's front panel. This prevents anyone from restoring front panel control or in anyway using the front panel. The HP 3852A can still be programmed over the HP-IB. The LOCK command has the syntax:

LOCK [*mode*]

The *mode* parameters are OFF and ON. At power-on or anytime the mode is OFF, the user has complete control of the front panel if the mainframe is under local control. If the mainframe is under remote control when the LOCK mode is off, the user has access to the subset of commands listed in Table 5-1.

When the LOCK mode is ON, the keyboard is disabled entirely. The keyboard can only be re-enabled by cycling the power or by sending LOCK OFF over the HP-IB.

www.valuetronics.com

If the keyboard has been disabled (LOCK ON), local control of the mainframe can be restored two ways. The first method is to send the LOCAL command followed by the LOCK OFF command as shown below.

```
70  OUTPUT 709;"LOCAL"
80  OUTPUT 709;"LOCK OFF"
```

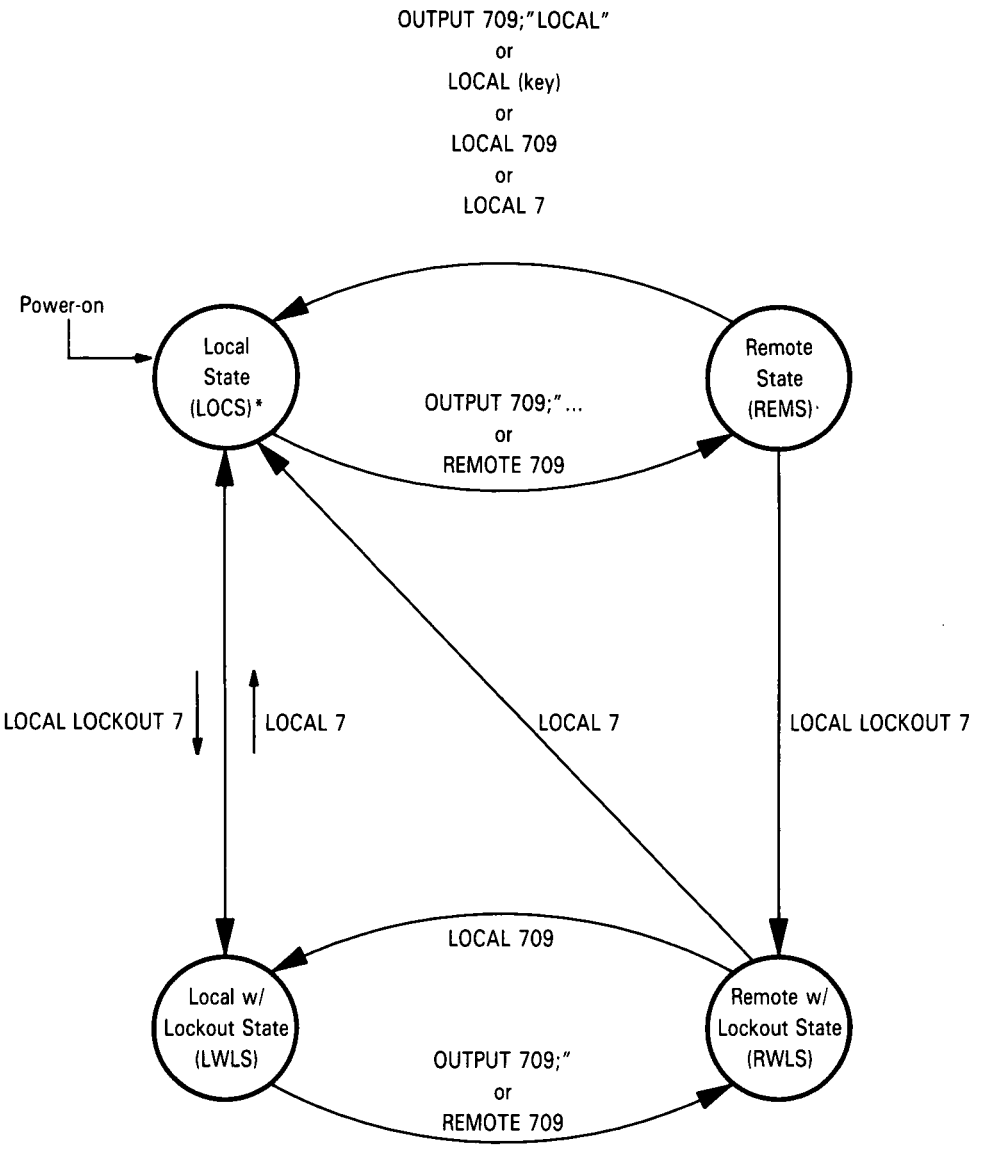
Note that sending the LOCAL command restores local control. However, if the LOCK command is not sent, the keyboard remains disabled and cannot be used.

The second method of restoring local control when the keyboard is locked is to send LOCK OFF over the HP-IB, then press the LOCAL key on the front panel.

HP 3852A Remote/Local States

The Remote/Local (RL) interface function designed into the HP 3852A enables the mainframe to be programmed over the HP-IB and from its front panel. Figure 5-1 identifies the various remote and local control states of the HP 3852A and the HP 3852A and HP Series 200/300 commands which cause the mainframe to enter those states.

www.valuetronics.com



*REMOTE 7 followed by an OUTPUT statement or REMOTE 709 is required to go to REMS if you entered LOCS from LWLS or RWLS

REMOTE 7 is required prior to LOCAL LOCKOUT 7 in order to return to LWLS if you entered LOCS from LWLS or RWLS.

Figure 5-1. HP 3852A REMOTE/LOCAL States

www.valuetronics.com

The Local State (LOCS)

When the HP 3852A is in the local state (LOCS), you have local control over the mainframe and access to all HP 3852A commands and functions allowed by the front panel keys. The HP 3852A powers-up in the local state.

The Remote State (REMS)

When the HP 3852A is in the remote state (REMS), the mainframe is under remote (HP-IB) control and only the subset of commands shown Table 5-1 can be entered from the front panel. The HP 3852A enters the remote state when it receives a command (with REN previously enabled) or the REMOTE statement from the controller as in the program lines:

```
10 OUTPUT 709;"ID?"
```

or

```
10 REMOTE 709
```

You can return to the local state (LOCS) from the remote state (REMS) using the HP 3852A's LOCAL key or command, or the controller's LOCAL 7 or LOCAL 709 statement.

The Local with Lockout State (LWLS)

From the local state (LOCS), you can enter the local with lockout state (LWLS) by sending LOCAL LOCKOUT 7 from the controller. The purpose of the lockout state is to prevent the HP 3852A's LOCAL command (or key) from restoring local control. Note, however, when LWLS is entered from LOCS, you still have local control of the mainframe until the mainframe is placed in remote (which becomes the remote with lockout state (RWLS)).

You can return to the local state (LOCS) from LWLS by executing the controller's LOCAL 7 statement. Then, to enter the remote state (REMS), you must execute the controller's REMOTE 7 statement followed by an HP 3852A command (i.e. OUTPUT 709;"...), or execute REMOTE 709. To go back to LWLS from LOCS, you must first execute REMOTE 7 followed by LOCAL LOCKOUT 7.

www.valuetronics.com

The Remote with Lockout State (RWLS)

In the remote with lockout state (RWLS), the HP 3852A's LOCAL command or key cannot be used to restore local control. The difference between RWLS and LWLS is that the HP 3852A is under remote control. As shown in Figure 5-1, RWLS can be entered from LOCS two ways. The first way is to enter the remote state (REMS). Once in remote, RWLS is reached by executing the controller's LOCAL LOCKOUT 7 statement. The second way to reach RWLS is to first enter LWLS, and then place the mainframe in remote by sending a command (i.e. OUTPUT 709;"...) or the controller's REMOTE 709 statement.

To enter LOCS from RWLS, execute LOCAL 7. To enter the remote state (REMS) from LOCS after returning from RWLS, either execute REMOTE 7 followed by a command (OUTPUT 709;"...), or execute REMOTE 709. To enter LWLS from LOCS, execute REMOTE 7 followed by LOCAL LOCKOUT 7.

NOTE

Detailed information on the Remote/Local HP-IB interface function can be found in the document describing IEEE Standard 488-1978: "IEEE Standard Digital Interface for Programmable Instrumentation."

Keyboard Capability

Table 5-2 summarizes the HP 3852A's keyboard capability for each remote/local state described previously, and with LOCK OFF and LOCK ON.

Table 5-2. HP 3852A Remote/Local Keyboard Capability

STATE	KEYBOARD CAPABILITY
Local, LOCK OFF, LOCS, LWLS	Full
Remote, LOCK OFF, REMS	Table 5-1
RWLS	Table 5-1 (except LOCAL key)
LOCK ON	None

www.valuetronics.com

HP 3852A Interface Functions

The HP 3852A's interface functions are HP-IB capabilities designed into the instrument. Table 5-3 identifies these capabilities by their mnemonics and provides a brief description of each.

NOTE

Detailed information on all HP-IB interface functions can be found in the document describing IEEE Standard 488-1978: "IEEE Standard Digital Interface for Programmable Instrumentation". Additional information is also available in Hewlett-Packard's "Tutorial Description of the Hewlett-Packard Interface Bus" (Part Number 5952-0156).

Table 5-3. HP 3852A Interface Functions

INTERFACE FUNCTION	MNEMONIC	DESCRIPTION
Source Handshake	SH1	This enables the HP 3852A to properly transfer multiline messages.
Acceptor Handshake	AH1	This enables the HP 3852A to guarantee proper reception of messages over the HP-IB.
Talker	T6	The HP 3852A is a basic talker which enables it to send data over the HP-IB. The HP 3852A can also respond to a serial poll.
Listener	L4	The HP 3852A is a basic listener which enables it to receive information over the HP-IB.
Service Request	SR1	This capability enables the HP 3852A to send a service request message to the controller.
Remote/Local	RL1	This capability enables the HP 3852A to be programmed over the HP-IB or from its front panel. (See HP 3852A Remote/Local States.)
Parallel Poll	PP1	This capability enables the HP 3852A to respond to a parallel poll. When a parallel poll is issued by the controller, the HP 3852A provides the state of its RDY (ready) Status Register bit.
Device Clear	DC1	With this capability, the HP 3852A is cleared by the Device Clear command issued from the controller.
Device Trigger	DT1	This capability enables the HP 3852A to be triggered over the HP-IB.

www.valuetronics.com

Contents

www.valuetronics.com

Chapter 6

Managing Data

Introduction

The purpose of this chapter is to describe how commands and measurement data are directed and stored within the HP 3852A. The information in this chapter is divided into five major sections: Addressing Conventions, Data Destinations and Formats, Storing Data in the Mainframe, Managing Packed Data, and Packed Data Transfer and Conversions. An overview of the content of each section is given below.

Addressing Conventions. This section describes the HP 3852A addressing scheme that allows you to direct commands to specific accessories or accessory channels. The section describes the mainframe slot and accessory channel numbering conventions and how they combine to form slot and channel addresses. This section also introduces the concept of the USE Channel and identifies the command parameters that when specified, require a mainframe/extender slot or accessory channel address.

Data Destinations and Formats. This section explains where measurement data is returned as a function of where the command(s) originated. This section also introduces the data formats associated with the HP 3852A and describes their representation, usage, and how a particular format is specified.

Storing and Reading Data. This section shows you how to store, manage, and retrieve data using the mainframe's internal memory. This includes information on declaring mainframe arrays and variables and on directing and retrieving data to and from memory. This section also covers how additional information is appended to an existing array and how arrays and variables are redeclared or deleted.

Managing Packed Data. This section shows you how to acquire and store packed data. Packed data is mainframe and accessory data that is sent to the mainframe's processor in an unconverted format rather than in a REAL or INTEGER (converted) format. Since packed data is unconverted, the data can be managed at faster rates.

www.valuetronics.com

Packed Data Transfer and Conversions. This section describes how packed data is retrieved from mainframe memory and output to a controller or transferred to another array. The section shows how packed data formats are converted to REAL and INTEGER formats. Programming examples summarize the packed data conversion routines.

Addressing Conventions

This section describes the addressing convention used with the HP 3852A and its plug-in accessories. Addressing enables you to direct commands to specific accessories or specify a particular channel or list of channels within a command.

ESCC

When an accessory or an accessory channel is specified within a command, the address (location) of the accessory and/or channel is actually given. An address identifies where the accessory is installed (mainframe, extender), the slot the accessory is installed in, and the channel that is selected. The information contained in the address is represented using the convention: ESCC, where E is the mainframe or extender number, S is a mainframe or extender slot number and CC is an accessory channel number.

Mainframe and Extender Numbering (E)

With the ESCC addressing convention, E will be a number from 0 to 7 depending on whether the accessory or channel specified is in the mainframe or in an extender. For any accessory installed in the mainframe, E is 0. Extenders used with the mainframe are assigned an extender number from 1 to 7 (Chapter 3). For an accessory installed in an extender, E is equal to the number set for that particular extender.

Mainframe and Extender Slot Numbering (S)

The S parameter within the ESCC convention represents the mainframe or extender slot number in which the accessory is installed. There are eight slots in the mainframe numbered 0-7. There are ten slots in the extender numbered 0-9.

Accessory Channel Numbering (CC)

The CC parameter within the convention represents an accessory channel. The range of channels depends on the individual accessory. Note that the first channel on all accessories is channel 0. Therefore, on a 20-channel accessory, the range of channels is 0-19. The HP 44701A and HP 44702A/B voltmeter accessories do not contain channels in the same sense as the multiplexer accessories for example. When the address of a voltmeter is specified, the voltmeter is assumed to be in "channel 0" of that particular slot. Thus CC is always 00 when a voltmeter address is specified.

www.valuetronics.com

In many commands a single channel or a list of channels is often specified. A channel list is specified using the ESCC convention in the form: ESCC [-ESCC] [ESCC [-ESCC]...]. What this form shows is that a channel list can be a single channel (ESCC), a list of channels (ESCC-ESCC), a combination of single channels and channel lists (ESCC, ESCC-ESCC), or a group of channel lists (ESCC-ESCC, ESCC-ESCC). A channel list can include channels in all slots of the mainframe or all channels between a mainframe and an extender(s). For example, 0000-7999 represents all possible channels within the HP 3852A mainframe and seven HP 3853A extenders. Note also that a channel list can be specified with channels in increasing or decreasing order (e.g. 0010-0000).

Specifying an Address

The ESCC convention which represents an address can be specified using INTEGER numbers (1, 2, 10, 500), REAL numbers (1.4, 20.67, 1E2), or can be the result of a numeric expression involving math functions (+, -, *, /, ^, PI, ABS, EXP, FRACT, INT, LGT, LOG, SQR), trigonometric operations (ATN, COS, SIN), binary functions (BINAND, BINCMP, BINEOR, BINIOR, BIT, ROTATE, SHIFT), or comparison operators (=, < >, <, >, ≤ , ≥). The following command is an example of an address specified as a math function:

```
10  OUTPUT 709;"CONFMEAS DCV,(SQR(25)),USE 700"
```

when this command executes, the math function is evaluated (square root of 25) and DC voltage is measured on channel 5 in mainframe slot 0.

An address can also be the value(s) stored in a mainframe array or variable. The following program is an example of a channel list specified within a mainframe array:

```
10  OUTPUT 709;"INTEGER ADRS(1)"
20  OUTPUT 709;"VWRITE ADRS 200,-202"
30  OUTPUT 709;"CONFMEAS DCV,ADRS,USE 700"
40  END
```

Line 10 in the program declares an Integer array large enough to store two numbers. Line 20 writes the numbers 200 and -202 to the array. When the measurement is performed in line 30, array ADRS is interpreted as channels 200 through 202 and DC voltage measurements are made on those channels.

Table 6-1 summarizes the preceding information on addressing and illustrates the different methods of specifying slots, channels, and channel lists within a command.

www.valuetronics.com

Table 6-1. HP 3852A Addressing Conventions

An addressing command contains a slot, channel, or a channel list as one of its parameters. Slots are specified as ES00, channels are specified as ESCC, and channel lists are specified as ESCC [-ESCC] [ESCC [-ESCC]...].

Definition:

E = Extender number	**S = Slot number in extender "E" where accessory is installed.**	**CC = Channel on accessory installed in slot "S".**
HP 3852A Mainframe: E = 0 HP 3853A Extender: E = 1-7* *Up to seven extenders per mainframe are allowed.	HP 3852A Mainframe: S = 0-7 HP 3853A Extender: S = 0-9* *Up to 78 slots per system (8 per mainframe, 10 per extender).	The maximum number of channels for any given slot is determined by the accessory in that slot.

When the parameter is a slot: (ES00)

Examples: (ES00)

0200	Specifies slot 2 in the HP 3852A mainframe. Leading zeros are optional.
A	Specifies the slot equivalent to the value of the variable A. If A = 100 for example, slot 1 in the mainframe is specified.
0	Specifies slot 0 in the HP 3852A mainframe.
1700	Specifies slot 7 in HP 3853A extender number 1.
OUTPUT 709;"ID? 3500"	Asks the identity of the accessory installed in slot 5 of HP 3853A extender number 3.

When the parameter is a channel: (ESCC)

Examples: (ESCC)

10	Specifies channel 10 of an accessory installed in slot 0 of the HP 3852A mainframe (leading zeros are optional).
1315	Specifies channel 15 of an accessory installed in slot 3 of HP 3852A extender number 1.
(SIN (1.57))	Specifies channel 1 on an accessory installed in slot 0 of the HP 3852A mainframe.
(SQR (25))	Specifies channel 5 of an accessory installed in slot 0 of the HP 3852A mainframe.
OUTPUT 709;"CLOSE 100"	Closes channel 0 of an accessory installed in slot 1 of the HP 3852A mainframe.

When the parameter is a channel list: (ESCC [-ESCC] [ESCC [-ESCC]...])

Examples: (ESCC [-ESCC] [ESCC [-ESCC]...])

ADRS	Specifies the list of channels represented in the array ADRS. If ADRS contains the numbers 200 and -202, the channel list is 200 through 202.
1110, 1201-1205	Specifies channel 10 in slot 1 of HP 3853A extender number 1 and channels 1 through 5 in slot 2 of extender number 1.
410-400	Specifies channels 10 through 0 in slot 4 of the HP 3852A mainframe.
0-7999	Specifies all channels in all slots of the HP 3852A mainframe and all HP 3853A extenders.
OUTPUT 709;"CLOSE? 0-1312"	Determines the channel state of channel 0 on an accessory installed in slot 0 of the HP 3852A mainframe, through channel 12 on an accessory installed in slot 3 in HP 3853A extender number 1. Note that in this example, all slots in the mainframe through slot 3 in the extender are checked for accessories with channel closures.

www.valuetronics.com

The USE Channel

The USE channel is an accessory or a specific accessory channel that is assigned to perform a specified function. The USE channel is assigned by the USE command (**USE** *ch*) or the USE parameter (e.g. **TRIG** [*source*] [**USE** *ch*]). The channel (*ch*) specified by the USE command or parameter is an accessory slot or channel address with the convention ESCC.

The USE Command

The USE command designates the accessory or accessory channel that will perform the functions indicated by those commands that contain [**USE** *ch*] as a command parameter. Specifying an accessory or channel with the USE command eliminates the need for specifying the [**USE** *ch*] parameter within a command as the channel specified by the USE command becomes the designated accessory or channel for those commands. For example, if an HP 44701A voltmeter is installed in slot 7 of the mainframe and you execute:

```
10 OUTPUT 709;"USE 700"
```

that voltmeter is used for all commands except for those where another accessory or channel is specified by the [**USE** *ch*] parameter.

Keep in mind that the USE channel specifies either an accessory slot in the case of the HP 44701A and HP 44702A/B voltmeters, or an accessory channel when, for example, the HP 44715A counter is used. Thus, the USE channel specified at a particular point in the program may not be suitable for all commands. A USE channel designated by the USE command is changed only by another USE command or when the instrument is reset or when the power is cycled.

The [USE ch] Parameter

Specifying the [**USE** *ch*] parameter (brackets not included) within a command enables you to designate a USE channel unique to that command. To understand what happens when [**USE** *ch*] is specified, consider the command:

CONFMEAS *function ch_list* [**NSCAN** *number*][**USE** *ch*][**INTO** *name*] or [*fmt*]

and the command sequence:

```
100 OUTPUT 709;"USE 700"
110 OUTPUT 709;"CONFMEAS DCV, 0-19, USE 200"
120 OUTPUT 709;"CONFMEAS DCV, 310-315"
```


www.valuetronics.com

Specifying [**USE** *ch*] in line 110 selects the voltmeter in mainframe slot 2 to be configured and perform the measurement for that command only. For the command in line 120, the voltmeter specified in line 100 is used. Notice that the brackets [ ] around **USE** *ch* indicate the parameter is optional. If [**USE** *ch*] had not been specified in line 110, the voltmeter specified in line 100 is used.

The Default USE Channel

The default USE channel is either the last slot or channel specified by the USE command or the power-on value if the USE command has not been executed previously.

The Power-on USE Channel

At power-on or following a reset, the selected USE channel is the lowest slot number and channel number of an accessory for which the USE channel command is valid. For example, if an HP 44721A is installed in slot 2 of the mainframe, the power-on USE channel would be 200 (mainframe slot 2, channel 0) since the USE command and parameter are valid for this accessory and provided no other accessory that responds to the USE command is installed in mainframe slots 0 or 1.

The USE? Command

The current USE channel as set by the USE command can be identified by sending the USE? command. The USE? command has the syntax:

USE? [INTO *name*] or [*fmt*]

Given this syntax, the USE channel returned by the USE? command can be stored into an array or variable in mainframe memory if **INTO** *name* is specified, or returned to the display and/or HP-IB output buffer if *fmt* is specified. If neither parameter is specified, the USE channel is returned to the display and/or output buffer in a default (IASC) format.

Data Destinations and Formats

This section covers the destination of data returned by commands and introduces you to the data formats in which data is managed within the HP 3852A.

Data Destinations

The destination of data returned by a command depends on where the command originated (front panel, HP-IB, subroutine) and on your response to the [**INTO** *name*] parameter within those commands. Data destinations include the mainframe's display, its HP-IB output buffer for entry into a controller, and the mainframe's internal memory. When the HP 44702A/B high-speed voltmeter is used, the voltmeter's internal memory or its GPIO interface are also available destinations.

As you read through the following information on data destinations, refer to Figure 6-1 for a visual summary of command origins, parameters, and prerequisites that determine the destination.

www.valuetronics.com

Commands Entered from Front Panel and HP-IB

When a command is entered from the HP 3852A front panel, the destination of the data is the mainframe's display or the mainframe's internal memory. Data is displayed only if the display has been enabled by the DISP (display) command and the data is in a mainframe display format (see Mainframe Display Formats).

When a command is entered from a controller over the HP-IB, the destination of the data is the mainframe's HP-IB output buffer and display, or mainframe memory. Note that data is displayed only if the data is in a mainframe display format, the display is enabled (DISP command), and the data is routed to the display with the MON (monitor) command.

Commands Executed within Subroutines

When a command that returns data is executed within an HP 3852A subroutine, the destination of the data depends on where the subroutine was called (CALL command) from. If the subroutine is called from the front panel, the destination is the mainframe's display or internal memory as described above. If the subroutine is called over the HP-IB, the destination is the HP-IB output buffer and display, or internal memory as also described above.

If a subroutine is called by the ON ALRM, ON LMT, or ON INTR command, the destination(s) depend on whether the command (ON ALRM, etc.,) was entered from the front panel or over the HP-IB.

The INTO name Parameter

Most of the HP 3852A commands that return data contain **INTO** *name* as a parameter. Specifying this parameter indicates an array, array element, or variable in mainframe memory is the destination for the data being returned. As an example of what happens when **INTO** *name* is specified, consider the commands:

REAL *name* [*(max_index)*][*name*[*(max_index)*]...]

CONFMEAS *function ch_list* [**NSCAN** *number*][**USE** *ch*][**INTO** *name*] or [*fmt*]

executed as:

```
40  OUTPUT 709;"REAL RGS(9)"
50  OUTPUT 709;"CONFMEAS DCV, 0-9, USE 200, INTO RGS"
```

Specifying **INTO** *name* directs the data into the previously defined array (RGS) in mainframe memory. Note that the data goes directly into memory and is not displayed or sent to the HP-IB output buffer. This occurs regardless of where the command originated. The brackets that enclose [**INTO** *name*] indicate that it is an optional parameter. If this parameter is not specified, the destination of the data then becomes a function of where the command originates.

www.valuetronics.com

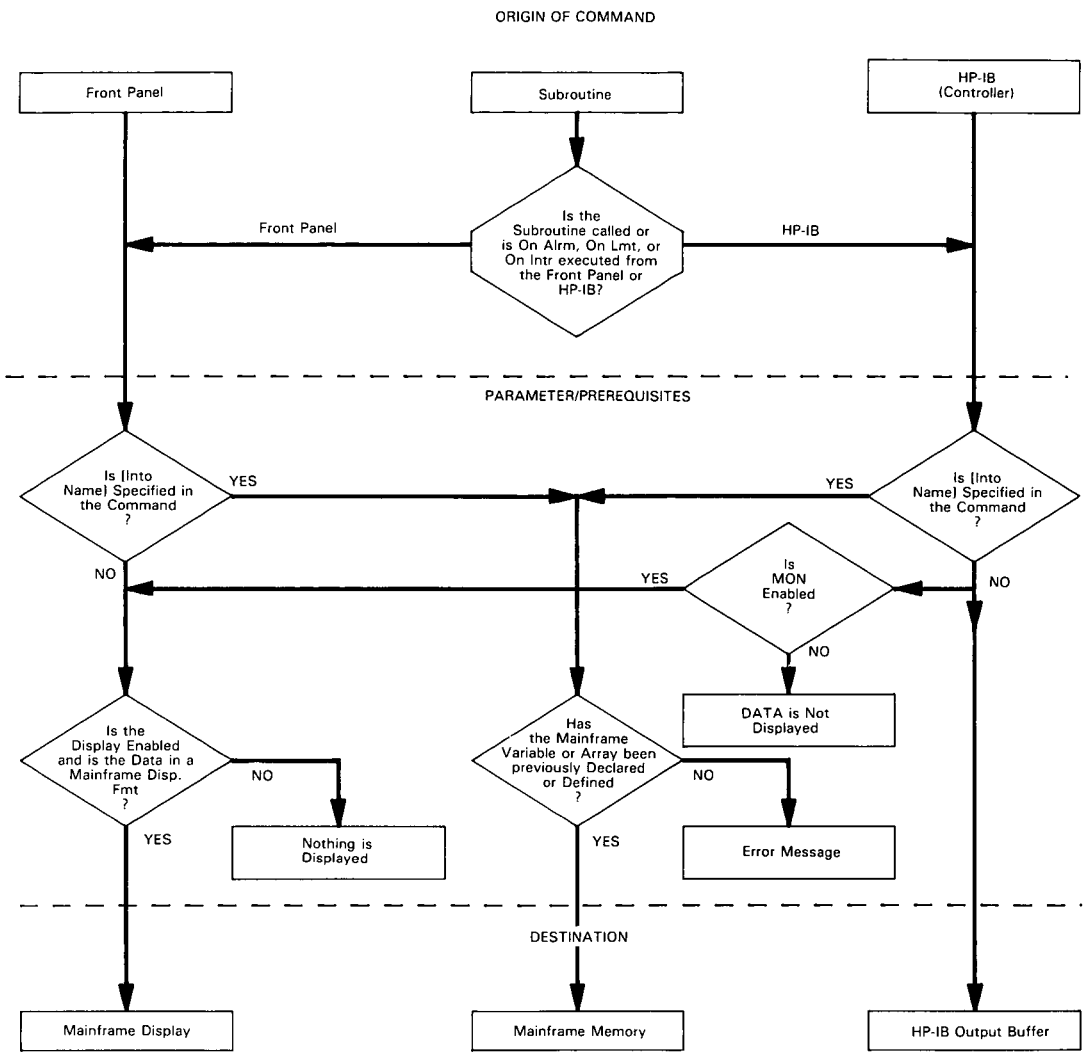


Figure 6-1. HP 3852A Data Destinations

www.valuetronics.com

Data Formats

The HP 3852A mainframe displays, stores, and outputs data to a controller in a variety of ASCII*, binary, and packed data formats. This part of the Data Destinations and Formats section shows you how to specify a particular format within a command and describes what the available formats are and when they can be used.

The fmt Parameter

Most of the commands that return data contain *fmt* along with **INTO** *name* as a command parameter. The *fmt* parameter is used to specify the format of the data returned by the command. As an example of what happens when *fmt* is specified, recall the command:

CONFMEAS *function ch_list* [**NSCAN** *number*][**USE** *ch*][**INTO** *name*] or [*fmt*]

executed as:

```
50 OUTPUT 709;"CONFMEAS DCV, 0-10, USE 200, IASC"
```

When *fmt* is specified, the data is returned in the IASC format to either the front panel display, the HP-IB output buffer, or both depending on where the command originated. Data is not returned to mainframe memory when *fmt* is specified as *fmt* and **INTO** *name* cannot be specified at the same time in the same command. If neither *fmt* nor **INTO** *name* is specified, data is returned in the command's default format to the front panel or HP-IB output buffer depending on where the command originated.

The Data Formats

When data is generated and returned by a command, the data is said to be in its packed (original) format. Packed formats vary with the data and the format is dependent on the command, the function specified, and the accessory returning the data.

The purpose of the *fmt* parameter described above is to enable the user to convert the data from its packed (original) format to a format that can be displayed by the mainframe or stored within the mainframe or a controller. The data formats that packed data can be converted to, and which are specified by the *fmt* parameter, are described in Table 6-2. Recall that when the *fmt* and **INTO** *name* parameters are not specified, the data is automatically converted to a default format (which will be one of the ASCII formats shown in the table). For example, the command:

CONFMEAS *function ch_list* [**NSCAN** *number*][**USE** *ch*][**INTO** *name*] or [*fmt*]

executed as:

```
100  OUTPUT 709;"CONFMEAS DCV, 0-10, USE 200"
```

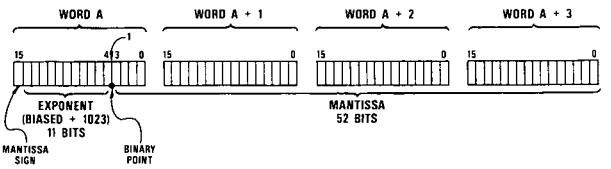
*American National Standard Code for Information Interchange.

www.valuetronics.com

the data is returned to the output buffer and display (if properly enabled) in the command's default format RASC.

See Managing Packed Data for descriptions of the various packed data formats.

Table 6-2. HP 3852A Data Formats

Format	Name	Representation
IASC	Short ASCII Integer	Bytes 0-5 | Bytes 6-7 Bytes 0-5: integer number, includes (–) sign and/or preceding spaces. Bytes 6-7: CR/LF Display Format: – 12345 Range: – 32768 to 32767
LASC	Long ASCII Integer	Bytes 0-10 | Bytes 11-12 Bytes 0-10: integer number, includes (–) sign and/or preceding spaces. Bytes 11-12: CR/LF Display Format: – 1234567890 Range: – 2147483648 to 2147483647
RASC	Real ASCII Number	Byte 0 | Bytes 1-8 | Byte 9 | Bytes 10-12 | Bytes 13-14 Byte 0: (–) sign or space Bytes 1-8: normalized 7-digit mantissa Byte 9: E Bytes 10-12: sign and 2-digit exponent Bytes 13-14: CR/LF Display Format: – 1.234567E + 12 Range: – 1.000000E + 38 to – 1.000000E – 37, 0, + 1.000000E – 37 to + 1.000000E + 38
DASC	Double Real ASCII Number	Byte 0 | Bytes 1-17 | Byte 18 | Bytes 19-22 | Bytes 23-24 Byte 0: (–) sign or space Bytes 1-17: normalized 16-digit mantissa Byte 18: E Bytes 19-22: sign and 3-digit exponent Bytes 23-24: CR/LF Display format: – 1.234567890123456E + 123 Range: – 1.797693134862315E + 308 to – 2.225073858507202E – 307, 0, + 2.225073858507202E – 307 to + 1.797693134862315E + 308
IN16	16-Bit Integer	16-bit 2's Complement Integer
RL64	64-Bit Real Number	
PACK	Packed Data	Accessory Dependent Format.

www.valuetronics.com

Mainframe Output Formats

Output data formats for the HP 3852A are the ASCII formats (IASC, LASC, RASC, DASC), the binary formats (IN16, RL64) and the packed formats described in Managing Packed Data. Output formats are the formats in which data can be returned to the HP-IB output buffer. The *fmt* parameter can be used to specify a particular output format.

Mainframe Display Formats

The HP 3852A displays data in the ASCII formats IASC, LASC, RASC, and DASC. Data can be displayed in an ASCII format other than a default format using the *fmt* parameter. ASCII data can also be sent to the output buffer. Due to the number of bytes used to represent readings in the ASCII formats, ASCII data transfers are much slower than binary and packed data transfers. Note that data cannot be stored in the mainframe in an ASCII format.

Each ASCII format is terminated with Carriage Return (CR) and Line Feed (LF) with the HP-IB EOI signal coincident with the LF character. If multiple readings are returned, the EOI line is coincident only with the last LF character even though CR LF occurs after each reading.

Mainframe Storage Formats

Data is stored in the HP 3852A in either a binary format (RL64, IN16) or in its packed (original) format. When data is stored in the mainframe, the format is specified by the INTEGER, DIM, REAL, or PACKED command rather than with the *fmt* parameter (see Storing and Reading Data). Data in a binary or packed format can either be stored in memory or sent to the output buffer. Binary or packed data formats cannot be displayed by the mainframe.

Each binary format and the packed formats are terminated with the HP-IB EOI signal coincident with the last byte transferred. If multiple readings are returned, only the last byte is coincident with the EOI signal line.

The END Command

Available in HP 3852As with mainframe firmware revision 2.2 or greater is the END command. The END command is used to suppress the EOI signal that is concurrent with the last byte of ASCII or binary data returned by an individual command. The END command has the syntax:

END *mode*

where END is the command header and *mode* either asserts or suppresses the End-or-Identify signal. *mode* = ON asserts EOI with the last byte of data. *mode* = OFF suppresses EOI. Note that with OUTBUF ON and EOI suppressed, data from several commands can be entered into the controller with a single ENTER statement.

See the Command Reference Manual for more information and examples of using the END command.

www.valuetronics.com

Default Formats

Associated with each command which returns data (e.g. CONFMEAS, CHREAD, VREAD) is a default data format. A default format is used when:

1. No other data format has been specified with the *fmt* parameter.
2. The data is not stored into another array or variable with the **INTO** *name* parameter.

The default format for most of the HP 3852A and plug in accessory commands will be one of the ASCII formats shown in Table 6-2. The Command Reference Manual and the Quick Reference Guide list the default format for each command.

Data Format Precision and Speed

For ASCII formats, precision is a function of the number of digits in the mantissa (display format). For binary formats, precision is a function of the number of bits in the mantissa which represent the number. Note that the REAL number formats are inherently higher precision than INTEGER formats as INTEGER formats are "rounded" and do not contain a fractional part.

Format "speed" is an indication of how quickly data in a particular format is managed throughout the mainframe. Specifically, format speed represents how fast data is converted from its packed (original) format to a default format or to a format specified by *fmt*, and also how fast the data is passed to the HP-IB output buffer. Data is generally managed faster when it is not displayed (DISP OFF) and not in an ASCII format. Data is managed at the fastest possible rates when the data remains in its packed format. Refer to "Maximizing System Throughput" in the "Packed Data Transfer and Conversions" section for information on optimizing system speed.

Data Headers

The SYSOUT and BLOCKOUT commands assign headers to data output by the HP 3852A. The headers identify the number of readings taken by each command, the data format (*fmt*) of the readings, the number of bytes/reading for the specified format, and the number of data bytes following the header.

The SYSOUT Header

The system output (SYSOUT) header is assigned by the SYSOUT command. The header prefaces data returned by the mainframe and output over the HP-IB.

The SYSOUT header consists of three numbers. The first number indicates the number of readings taken by the command that is returning the data. This number is in the IASC format and cannot be specified otherwise. The second number of the header indicates the format (*fmt*) the data is in. Table 6-3 lists the numbers used to represent these formats. The second number is returned in the LASC format and cannot be specified otherwise.

www.valuetronics.com

The third number of the header indicates the number of bytes/reading for data in the format specified (does not include the bytes for CR/LF (carriage return/line feed)). The number of bytes/reading for each format is found in Table 6-3. The third number of the header is returned in the IASC format and it also cannot be specified otherwise.

Table 6-3. SYSOUT Header Parameters

1st Parameter - Number Readings	2nd Parameter - Data Format [a]	3rd Parameter - Bytes/Reading
Number of readings taken by command	IN16 = ± 1 RL 64 = ± 2 PACK = ± 5 IASC = ± 6 LASC = ± 7 RASC = ± 8 string = ± 9 DASC = ± 11	IN16 = 2 RL64 = 8 PACK = [b] IASC = 6 LASC = 11 RASC = 13 string = [c] DASC = 22

[a] = Number returned is positive (+) for BLOCKOUT OFF, negative (-) for BLOCKOUT ON.

[b] = Bytes/Reading for the PACK format is accessory command and function dependent.

[c] = Number of characters in the string, including quotes (if any) but not including CR/LF (carriage return/line feed).

Using the SYSOUT Header

As an example of how to assign the header with the SYSOUT command and to illustrate how the header appears, look at the program and output that follow:

```
10  OUTPUT 709;"RST"
20  REAL RGS(0:6)
30  OUTPUT 709;"SYSOUT ON"
40  OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700, RASC"
50  ENTER 709;RGS(*)
60  PRINT RGS(*)
70  END

4       8       13      1.314127        5.02645         1.553484        1.545791
```

In the output, the first three numbers represent the SYSOUT header. Recall from the previous discussion that the first number in the header indicates the number of readings taken by the command. If you look at line 40 of the program you can see that the command scans and measures four channels (300-303). The second byte of the header indicates the format in which the data was returned. Referring to Table 6-3, "8" corresponds to the RASC format which is the format specified in line 40. The third byte of the header indicates the bytes/reading for the specified format (RASC). Referring to the output and Table 6-3, there are 13 bytes/reading when the data is returned in the format RASC.

A possible application of the SYSOUT command is illustrated in the following program. The program assigns the SYSOUT header, performs a series of DC voltage measurements, then it reads the SYSOUT header and declares an INTEGER, REAL, or string array in the HP Series 200 or Series 300 computer to store the data returned.

www.valuetronics.com

```
10   INTEGER A(1:3)
20   OUTPUT 709;"OUTBUF ON"
30   OUTPUT 709;"SYSOUT ON"
40   OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700"
50   GOSUB Sys
60   OUTPUT 709;"ERRSTR?"
70   GOSUB Sys
80   STOP
90   Sys:  !
100  ENTER 709;A(*)
110  PRINT "SYSOUT HEADER IS: ";A(*)
120  PRINT
130  ! THIS SERIES OF COMMANDS ASSIGN THE SYSOUT HEADER, PERFORM THE
140  ! DC VOLTAGE MEASUREMENTS, WAIT UNTIL THE MAINFRAME HAS FINISHED
150  ! EXECUTING, THEN READ AND PRINT THE SYSOUT HEADER.
160  !
170  SELECT A(2)
180   CASE 1 ! A(2) = 1 (DATA IS IN IN16 FORMAT)
190     ALLOCATE INTEGER In16(1:A(1))
200     ENTER 709 USING "#,W";In16(*)
210     PRINT "READINGS ARE: ";In16(*)
220     PRINT
230     DEALLOCATE In16(*)
240  ! THESE COMMANDS ALLOCATE AN INTEGER ARRAY WHEN THE FORMAT
250  ! IDENTIFIED BY THE SYSOUT HEADER IS IN16. THE DATA RETURNED IS
260  ! THEN ENTERED AND PRINTED.
270  !
280   CASE 2 ! A(2) = 2 (DATA IS IN RL64 FORMAT)
290     ALLOCATE REAL RL64(1:A(1))
300     ASSIGN @Hp_3852 TO 709; FORMAT OFF
310     ENTER @Hp_3852;RL64(*)
320     PRINT "READINGS ARE: ";RL64(*)
330     PRINT
340     DEALLOCATE RL64(*)
350     ASSIGN @Hp_3852 TO *
360  ! THESE COMMANDS ALLOCATE A REAL ARRAY WHEN THE FORMAT IDENTIFIED
370  ! BY THE SYSOUT HEADER IS RL64. THE DATA RETURNED IS THEN
380  ! ENTERED AND PRINTED.
390  !
400   CASE 6,7 ! A(2) = 6 or 7 (DATA IS IN IASC or LASC FORMAT)
410     ALLOCATE INTEGER Iasc(1:A(1))
420     ENTER 709;Iasc(*)
430     PRINT "READINGS ARE: ";Iasc(*)
440     PRINT
450     DEALLOCATE Iasc(*)
460  ! THESE COMMANDS ALLOCATE AN INTEGER ARRAY WHEN THE FORMAT
470  ! IDENTIFIED BY THE SYSOUT HEADER IS EITHER IASC OR LASC. THE
480  ! DATA RETURNED IS THEN ENTERED AND PRINTED.
490  !
500   CASE 8,11 !A(2) = 8 OR 11 (DATA IS IN RASC OR DASC FORMAT)
```


www.valuetronics.com

```
510     ALLOCATE REAL R_d_asc(1:A(1))
520     ENTER 709;R_d_asc(*)
530     PRINT "READINGS ARE: ";R_d_asc(*)
540     PRINT
550     DEALLOCATE R_d_asc(*)
560  ! THESE COMMANDS ALLOCATE A REAL ARRAY WHEN THE FORMAT IDENTIFIED
570  ! BY THE SYSOUT HEADER IS EITHER RASC OR DASC. THE DATA RETURNED
580  ! IS THEN ENTERED AND PRINTED.
590  !
600   CASE 9 ! A(2) = 9 (DATA IS A STRING)
610     ALLOCATE String$(1:A(1))[A(3)+2]
620     ENTER 709;String$(*)
630     PRINT "STRING IS: ";String$(*)
640     PRINT
650     DEALLOCATE String$(*)
660  ! THESE COMMANDS ALLOCATE A STRING VARIABLE WHEN THE SYSOUT
670  ! HEADER INDICATES STRING DATA IS BEING RETURNED. THE STRING
680  ! RETURNED IS THEN PRINTED.
690  !
700   CASE ELSE
710  END SELECT
720  RETURN
730  END
```

The output generated by this program for the commands executed (CONFMEAS, ERRSTR?) anmd for the format specified (RASC) is shown below:

```
SYSOUT HEADER IS:  4              8              13

READINGS ARE:  .1881445      5.02612      1.55197      1.484533

SYSOUT HEADER IS:  1              9              60

STRING IS:"    0: NO ERROR                                        "
```

The BLOCK-OUT Header

The programming practices and data formats associated with the HP 3852A comply with IEEE Standard 728-1982 "Recommended Practice for Code and Format Conventions." Associated with this standard is a block output data header. This header is assigned to binary data returned by the HP 3852A using the BLOCKOUT command.

The BLOCKOUT header is four bytes long. The first byte is the # sign, the second byte is the letter A, and the third and fourth bytes indicate the number of data bytes which follow the header. The bytes representing the CR/LF (carriage return/line feed) are not included in the count. The BLOCKOUT header precedes data output in a binary format (IN16, RL64, PACK) only. A new header precedes each command returning binary data. If a particular command returns more than 32760 bytes of data with the

www.valuetronics.com

BLOCKOUT header assigned, the data is divided into blocks of 32760 bytes, with a header preceding each block and a CR/LF terminating each block as shown below:

<header> <data (32760 bytes)> <CR/LF> <header> <data> <CR/LF> ...

If the BLOCKOUT header and SYSOUT header are both assigned, the SYSOUT header will always precede the BLOCKOUT header.

The following program illustrates how the BLOCKOUT header is assigned, how data with the header is read into a controller, and how the header appears. The program deals with the RL64 format only.

```
10  OUTPUT 709;"RST"
20  OUTPUT 709;"WAIT 0.1"
30  PRINT USING "@"
40  INTEGER Num_bytes,Num_readings
50  DIM Block_header$[2],Crlf$[2]
60  ASSIGN @Hp_3852 TO 709;FORMAT OFF
80  OUTPUT 709;"BLOCKOUT ON"
90  OUTPUT 709;"CONFMEAS DCV, 0-3, USE 700, RL64"
100  GOSUB Block_enter
110  OUTPUT 709;"CONFMEAS DCV 5-8, USE 700, RL64"
120  GOSUB Block_enter
130  STOP
140 Block_enter:  !
150  REPEAT
160   ENTER 709 USING "#,K";Block_header$
170   ENTER @Hp_3852;Num_bytes
180   Num_readings=Num_bytes/8
190   ALLOCATE REAL A(1:Num_readings)
200   PRINT "BLOCKOUT HEADER IS: ";Block_header$,Num_bytes
210   PRINT
220   ENTER @Hp_3852;A(*)
230   ENTER 709;Crlf$
240   PRINT "READINGS ARE: ";A(*)
250   PRINT
260   DEALLOCATE A(*)
270  UNTIL NOT BIT(SPOLL(709),0)
280  RETURN
290  END
```


www.valuetronics.com

The output generated by this program for the commands executed (CONFMEAS) and the format specified (RL64) is:

```
BLOCKOUT HEADER IS: #A        32

READINGS ARE: .951553         5.02627     1.549021     1.540837

BLOCKOUT HEADER IS: #A        32

READINGS ARE: 1.503689        1.474124    1.453725     1.426588
```

Storing and Reading Data

This section describes the prerequisites and procedures for storing data and reading data from the mainframe's internal memory. The section describes how arrays and variables are declared, how they are redeclared or deleted, how data is directed into them, and how data is retrieved from them.

Mainframe Memory Size

When a command generates data, the destination of the data returned can be an array or variable in the mainframe's internal memory. In addition to measurement data, user-defined variables and HP 3852A subroutines can also be stored internally.

There are approximately 11000 bytes of available memory in the standard HP 3852A. The memory can be expanded an additional 256k, 1M, 2M, or 4 MBytes with the various Extended Memory boards available. Extended Memory board installation and ordering information is given in Chapter 3.

Memory Size Vs Reading Length

When declaring arrays in the mainframe, memory size (256 kbytes, 1 Mbyte, etc.,) should not be interpreted as the maximum number of readings that can be stored. The number of bytes required to store a single reading in the mainframe depends on the storage type (REAL, INTEGER), or the accessory returning data if packed formats are used.

REAL Arrays

Data stored in a REAL array (RL64 format) requires eight bytes of mainframe memory to store each reading. There are also 72 bytes of overhead associated with each real array declared. The overhead contains information such as the array name, the type, and its maximum index. For example, a REAL array declared to store 10 readings uses 152 bytes of memory. 80 bytes are required for the readings (10 readings at 8 bytes/reading) plus the 72 bytes of overhead.

www.valuetronics.com

When declaring REAL arrays, note that the same amount of memory is required to store 1 reading in a single element array as is required to store up to three readings in a three element array. This is because the mainframe allocates a minimum of 28 bytes of storage space per array. Thus, to store 1 reading in a REAL array, 100 bytes of memory are used. 72 bytes are overhead and 28 bytes are reserved for the data, rather than eight bytes (1 reading at 8 bytes/reading) as might be expected.

INTEGER Arrays Data stored in an INTEGER array (IN16 format) requires two bytes of mainframe memory to store each reading. There are also 72 bytes of overhead associated with each INTEGER array declared containing name, type, and maximum index information. For example, an INTEGER array declared to store 20 readings uses 112 bytes of mainframe memory. 40 bytes are required for the readings (20 readings at 2 bytes/reading) plus the 72 bytes of overhead.

When declaring INTEGER arrays, note that the same amount of memory is required to store 1 reading in a single element array as is required to store up to 14 readings in a 14 element array. This is again due to the minimum of 28 bytes the mainframe allocates per array. Thus to store 1 reading in an INTEGER array, 100 bytes of memory are used. 72 bytes are overhead and 28 bytes are reserved for the data, rather than two bytes (1 reading at 2 bytes/reading) as might be expected.

PACKED Arrays Packed arrays also have a 72 byte overhead. However, the number of bytes per reading depend on the accessory, the command, and the function specified. See Table 6-6 for a listing of the bytes/reading for the packed formats.

Variables For each REAL and INTEGER variable declared, 44 bytes of memory are used. This includes the overhead and the reading storage space allotted.

Table 6-4 summarizes the bytes/reading for each storage type as well as the overhead associated with each mainframe array and variable declared.

Table 6-4. Data Type Bytes/Reading

Storage Type	Bytes/Reading	Overhead Per Array	Total Bytes Per Variable
REAL (RL64)	8	72	44
INTEGER (IN16)	2	72	44
PACKED	*	72	—

*Depends on accessory, command, and function

www.valuetronics.com

Declaring Arrays and Variables

Before data is stored in the mainframe, an array or variable in memory must be declared. The HP 3852A commands used to declare arrays and variables are DIM, REAL, INTEGER, and PACKED.

The DIM, REAL, INTEGER, and PACKED commands can be executed from the front panel, over the HP-IB, or within an HP 3852A subroutine. Arrays and variables within the mainframe are global. This means that regardless of where the array or variable was declared (front panel, HP-IB, subroutine), they can be accessed by commands returning data or by the user from any command origination location.

When declaring REAL and INTEGER arrays, the maximum index (i.e. size) specified is the maximum number of readings to be stored in the array, and not the number of bytes allocated to that array. When PACKED arrays are declared, however, the maximum index specified is the number of bytes of mainframe memory to be allocated to that array rather than the number of readings to be stored.

Only the DIM, REAL, and INTEGER commands are covered in this section. Refer to Managing Packed Data for information on declaring PACKED arrays.

NOTE

The portion of mainframe memory (including extended memory) in which arrays and variables are declared is volatile. This means that if the mainframe is turned off or accidently loses power, all arrays, variables, and subroutines are erased. Resetting the instrument with the RST key on the front panel or sending the RST, RST HARD, or SCRATCH commands also deletes all previously declared arrays, variables, and subroutines.

Array and Variable Names

Arrays and variables in mainframe memory are accessed by name. The name of the array or variable is a parameter specified within the DIM, REAL, INTEGER, and PACKED commands. The naming conventions used by the HP 3852A are outlined as follows:

1. Array and variable names cannot exceed eight characters in length.
2. Names can consist of letters, numbers, ?, or _.
3. Names must begin with a letter, ?, or _. Names cannot begin with a number.

www.valuetronics.com

4. The array or variable name cannot be an HP 3852A command header or command parameter (e.g. READ, RDGS, ON).

The following program lines are examples of valid array (or variable) names:

```
20 OUTPUT 709;"DIM RKS (4)"

20 OUTPUT 709;"REAL ?D? (9)"

20 OUTPUT 709;"INTEGER STOR_RDS (20)"
```

The DIM Command

The DIM (DIMension) command declares REAL arrays. When data is stored in an array declared by the DIM command, each reading is stored in the binary format RL64 (8 bytes/reading). The syntax of the DIM command is:

DIM *name (max_index)* [*name (max_index). . .*]

To understand how to use the DIM command, let's look at the headers and parameters of the syntax statement.

DIM - This is the command header which is entered in either upper or lower case.

name - This parameter specifies the name of the array. The name used must follow the conventions previously described.

(max_index) - This parameter specifies the maximum index of the array. The maximum index is the maximum number of readings you want to store in that particular array. Note that arrays declared by the DIM command have a starting index of 0. This means that to set up an array to store 10 readings, *(max_index)* would be 9, as the first reading is stored in element 0 of the array. Similarly, if you wanted to set up an array to store one reading, *(max_index)* would be 0. Note that when specifying the maximum index, the parentheses are included. When using the DIM command, you must always specify a maximum index.

[*name (max_index). . .*] - The brackets [ ] around these parameters indicate that more than one array can be declared with a single DIM command. The brackets are not included, however.

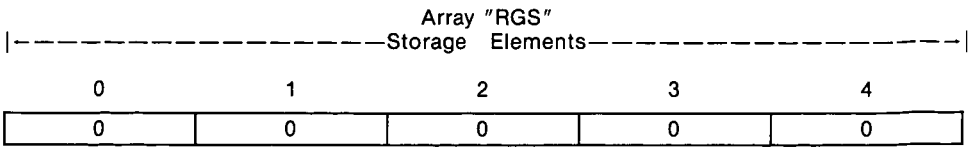
Declaring Arrays with the DIM Command

To illustrate the use of the DIM command, refer to the following program line:

```
10 OUTPUT 709;"DIM RGS (4)"
```


www.valuetronics.com

When this line is executed, an array (RGS) with a maximum index of four is declared in mainframe memory. The array can store up to five readings (starting index is 0) as indicated in Figure 6-2. Note that when an array is declared with the DIM command, each element contains the number 0 until a reading is stored in that element.




```
10 OUTPUT 709;"DIM RGS (4)"
```

Figure 6-2. Declaring an Array with the DIM Command

When data is stored in a mainframe array declared by the DIM command, the array must be declared before the data is returned as illustrated in the following program:

```
10 OUTPUT 709;"DIM RGS (4)"
20 OUTPUT 709;"CONFMEAS DCV, 300-304, USE 700, INTO RGS"
30 END
```

To declare more than one array, you can execute the DIM command as shown in the following example:

```
10 OUTPUT 709;"DIM RGS (4), DAT (9)"
```

The REAL Command

The REAL command declares REAL arrays and variables. When data is stored in an array or variable declared by the REAL command, each reading is stored in the binary format RL64 (8 bytes/reading). The syntax of the REAL command is:

REAL *name*[*(max_index)*][*name* [*(max_index)*]. . .]

To understand how to use the REAL command, we'll again look at the header and parameters of the command.

REAL - This is the command header which is entered in either upper or lower case.

name - This parameter specifies the name of the array or variable. The name used must follow the conventions previously described.

[*(max_index)*] - This parameter specifies the maximum index of the array. The maximum index is the maximum number of readings you want to store in that particular array. Note that arrays declared by the REAL command have a starting index of 0. This means that to set up an array to store a

www.valuetronics.com

maximum of 10 readings, *(max_index)* would be 9, as the first reading is stored in element 0 of the array. Similarly, if you want to set up an array to store one reading, *(max_index)* would be 0. When the maximum index of an array is specified, the parentheses are included. Name without a maximum index declares a REAL variable.

[*name* [*(max_index)*]. . .] - The brackets [ ] around these parameters indicate that more than one array or variable can be declared by a single REAL command. The brackets are not included, however.

Declaring REAL Arrays

To illustrate the use of the REAL command, refer to the following program line:

```
10  OUTPUT 709;"REAL DAT (9)"
```

When this line is executed, an array (DAT) with a maximum index of 9 is declared in mainframe memory. The array can store a maximum of 10 readings (starting index is 0) as indicated in Figure 6-3. When an array is declared with the REAL command, each element contains the number 0 until a reading is stored in that element.

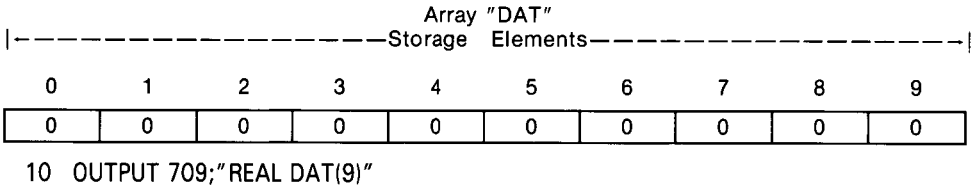


Figure 6-3. Declaring an Array with the REAL Command

When data is stored in a mainframe array declared by the REAL command, the array must be declared before the data is returned as illustrated in the following program:

```
10  OUTPUT 709;"REAL DAT (9)"
20  OUTPUT 709;"CONFMEAS DCV, 300-304, USE 700, INTO DAT"
30  END
```

More than one REAL array, variable, or combination of both can be declared with a single REAL command as previously indicated. For example, the program line:

```
10  OUTPUT 709;"REAL DATA (9), NUM (4)"
```

uses a single REAL command to declare the arrays DATA and NUM.

www.valuetronics.com

Declaring REAL Variables

The brackets around the first *(max_index)* parameter indicate the parameter is optional. When *(max_index)* is not specified, a REAL variable is declared. For example, if we execute the REAL command as:

```
10  OUTPUT 709;"REAL DAT"
```

we declare the variable DAT in mainframe memory. When a variable is declared with the REAL command, it has the value 0 until a different value is stored or assigned.

Variables, like arrays, must be declared before the data is returned or a value assigned as indicated in the following programs:

```
10  OUTPUT 709;"REAL DAT"
20  OUTPUT 709;"CONFMEAS DCV, 300, USE 700, INTO DAT"
30  END
```

```
10  OUTPUT 709;"REAL A"
20  OUTPUT 709;"LET A = 10"
30  END
```

The following program shows how a single REAL command can be used to declare more than one variable and a combination of arrays and variables:

```
10  OUTPUT 709;"REAL DAT, RGS"
20  OUTPUT 709;"REAL NUM (4), DAT1"
30  END
```

Declaring a variable is similar to declaring an array with a maximum index of 0. However, before declaring a variable rather than an array, note that an array can be redeclared (maximum index changed) if necessary, whereas a variable cannot (see Redeclaring and Deleting Arrays and Variables).

The INTEGER Command

The INTEGER command declares INTEGER arrays and variables. When data is stored in an array or variable declared by the INTEGER command, each reading is stored in the binary format IN16 (2 bytes/reading). The syntax of the INTEGER command is:

INTEGER *name* [*(max_index)*][*name* [*(max_index)*]. . .]

We can understand how to use the INTEGER command by looking at the command header and parameters.

INTEGER - This is the command header which is entered in either upper or lower case.

name - This parameter specifies the name of the INTEGER array or variable. The name used must follow the conventions previously described.

www.valuetronics.com

[(max_index)] - This parameter specifies the maximum index of the array. The maximum index is the maximum number of readings you want to store in that particular array. Note that arrays declared by the INTEGER command have a starting index of 0. This means that to set up an array to store a maximum of 10 readings, *(max_index)* would be 9, as the first reading is stored in element 0 of the array. Similarly, if you want to set up an array to store one reading, *(max_index)* would be 0. When the maximum index of an array is specified, the parentheses are included. Name without a maximum index declares an INTEGER variable.

[name [(max_index)]. . .] - The brackets [ ] around these parameters indicate that more than one array or variable can be declared by a single INTEGER command. The brackets are not included.

Declaring INTEGER Arrays

To illustrate the use of the INTEGER command, refer to the following program line:

```
10  OUTPUT 709;"INTEGER RDINT (19)"
```

When this line is executed, an array (RDINT) with a maximum index of 19 is declared in mainframe memory. The array can store a maximum of 20 readings (starting index is 0) as indicated in Figure 6-4. When an array is declared with the INTEGER command, each element contains the number 0 until a reading is stored in that element.

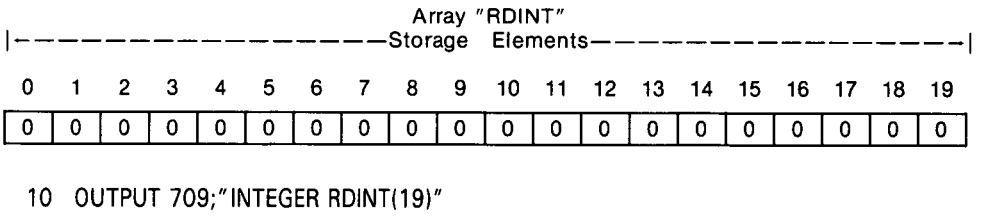


Figure 6-4. Declaring an Array with the INTEGER Command

When data is stored in a mainframe array declared by the INTEGER command, the array must be declared before the data is returned as illustrated in the following program:

```
10  OUTPUT 709;"INTEGER RDINT (19)"
20  OUTPUT 709;"CONFMEAS DCV, 300-304, USE 700, INTO RDINT"
30  END
```

More than one array, variable, or combination of both can be declared with a single INTEGER command as previously indicated. For example, the program line:

```
10  OUTPUT 709;"INTEGER RDINT (19), INT (10)"
```

uses a single INTEGER command to declare the arrays RDINT and INT.

www.valuetronics.com

Declaring INTEGER Variables

The brackets around the first *(max_index)* parameter indicate the parameter is optional. When *(max_index)* is not specified, an INTEGER variable is declared. For example, if we execute the INTEGER command as:

```
10 OUTPUT 709;"INTEGER RDINT"
```

we declare the variable RDINT in mainframe memory. When a variable is declared by the INTEGER command, it has the value 0 until a different value is assigned.

Variables, like arrays, must be previously declared before data is returned or a value assigned as illustrated in the following examples:

```
10 OUTPUT 709;"INTEGER RDINT"
20 OUTPUT 709;"CONFMEAS DCV, 300, USE 700, INTO RDINT"
30 END
```

```
10 OUTPUT 709;"INTEGER B"
20 OUTPUT 709;"LET B = 4"
30 END
```

The following program shows how a single INTEGER command can be used to declare more than one variable and a combination of arrays and variables:

```
10 OUTPUT 709;"INTEGER RDINT, DAINT"
20 OUTPUT 709;"INTEGER DAT (4), RGS"
30 END
```

Declaring a variable is similar to declaring an array with a maximum index of 0. However, before declaring a variable rather than an array, note that an array can be redeclared (maximum index changed) if necessary, whereas a variable cannot (see Redeclaring and Deleting Arrays and Variables).

Entering Data into Memory

After an array or variable has been declared by the DIM, REAL, or INTEGER command, data can then be stored in the mainframe. The most often used method for entering data is with the **INTO** *name* parameter associated with commands which return data. Other methods for entering data involve the LET and VWRITE commands which enable you to write data to an array or variable directly. Each method is discussed in the following paragraphs. Again, only REAL and INTEGER arrays are used.

See Managing Packed Data for information on directing data into packed arrays.

www.valuetronics.com

The INTO name Parameter

When measurement data is returned by a command, the data is directed into an array, array element, or variable in mainframe memory by specifying the command's **INTO** *name* parameter. In all commands, the **INTO** *name* parameter appears as shown in the following examples:

CONFMEAS *function ch_list* [**NSCAN** *number*][**USE** *ch*][**INTO** *name*] or [*fmt*]

CHREAD *ch* [**INTO** *name*] or [*fmt*]

TIME [**INTO** *name*] or [*fmt*]

To illustrate how the **INTO** *name* parameter is used, consider the following program:

```
10 OUTPUT 709;"REAL RGS (3)"
20 OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700, INTO RGS"
30 END
```

When we run this program, line 10 declares a REAL array (RGS) in memory to store four readings. When line 20 is executed, the voltmeter in mainframe slot 7 is configured, then it measures the DC voltage on channels 0-3 of the accessory installed in mainframe slot 3. When the readings are returned, they are stored in the mainframe array RGS as represented in Figure 6-5.

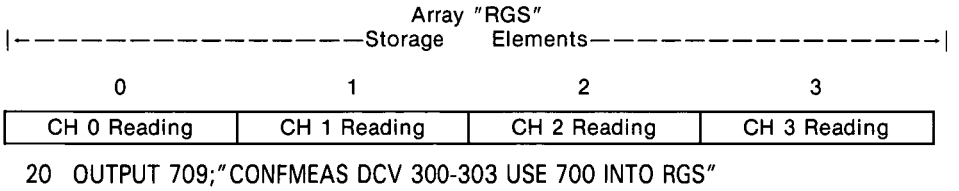


Figure 6-5. Storing Data with the INTO Parameter

The next program shows how readings can be returned to specific elements within an array.

```
10 OUTPUT 709;"REAL RGS (3)"
20 OUTPUT 709;"CONFMEAS DCV, 300-301, USE 700, INTO RGS(1)"
30 END
```

When this program runs, the reading taken on channel 0 is stored in element 1 of array RGS, and the reading on channel 1 is stored in element 2.

www.valuetronics.com

When specifying the **INTO** *name* parameter, a mainframe array or variable must be declared before the data is returned. If, in the previous programs the commands in lines 10 and 20 were reversed, the mainframe would display "ERROR 71: UNDEFINED WORD" since the array (and element) specified to store the data does not exist at that point.

The LET Command

The LET command enables the user to assign a value to a previously defined variable or array element. The value written to the variable or array element can be a number assigned directly, the result of a numeric expression, or a value copied from one variable or array element to another.

The LET command can only be used with arrays and variables declared by the DIM, REAL, and INTEGER commands. LET cannot assign values to PACKED arrays.

To understand how to use the LET command, let's look at the command syntax:

[LET] *variable* or *array(index)* = *number*

[LET] - This is the command header which is entered in either upper or lower case. The brackets [ ] indicate that the header is optional. This means that a variable or array element can be assigned a value by executing the command as:

variable or *array(index)* = *number*

variable - The variable is a Real or Integer variable previously declared by the REAL or INTEGER command.

array(index) - A specific element of a Real or Integer array previously declared by the DIM, REAL, or INTEGER command.

number - A number or numeric expression such as a math function (+, -, *, /, ABS, EXP, FRACT, INT, LGT, LOG, SGN, SQR), a trigonometric operation (ATN, COS, SIN), or a binary function (BINAND, BINCMP, BINEOR, BINIOR, BIT, ROTATE, SHIFT). *number* can also be a variable or array element from which the data is copied. Firmware revisions 3.5 and greater allow logical expressions (e.g. B=C, B<C, B>C, B≤C, B≥C) to be specified for the number parameter. For example A = (B>C) will set A = 1 if B>C, or to 0 if B≤C.

The following programs show how a variable and an array element are assigned values using the LET command:

www.valuetronics.com

1.

```
10  OUTPUT 709;"REAL A"
20  OUTPUT 709;"LET A=4"
30  END
```

2.

```
10  OUTPUT 709;"DIM A(4),B(9)"
20  OUTPUT 709;"VWRITE A 1,2,3,4,5"
30  OUTPUT 709;"LET B(4)=A(2)"
40  END
```

3.

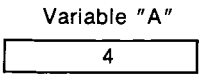
```
10  OUTPUT 709;"INTEGER A(9)"
20  OUTPUT 709;"A(3)=4"
30  END
```

4.

```
10  OUTPUT 709;"INTEGER A"
20  OUTPUT 709;"A=BINCMP(12)"
30  END
```

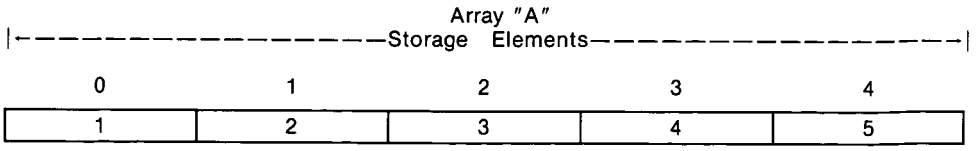
In program 1, a REAL variable (A) is declared in line 10. In line 20, the value of 4 is assigned to A. In program 2, the DIM command declares REAL arrays A and B with maximum indexes of 4 and 9 respectively. In line 20, the VWRITE command writes data to array A. Line 30 copies the value (3) from element 2 in array A to element 4 in array B. In program 3, an INTEGER array (A) with a maximum index of 9 is declared. In line 20, the LET command assigns the number 4 to the fourth element (starting index is 0) of array A. Recall that the LET header is optional, and, therefore, does not have to be entered. In program 4, an INTEGER variable is declared in line 10. In line 20, the binary compliment of 12 is evaluated and the result is assigned to variable A. Figure 6-6 represents the data stored in the variables and arrays specified in programs 1 through 4.

www.valuetronics.com

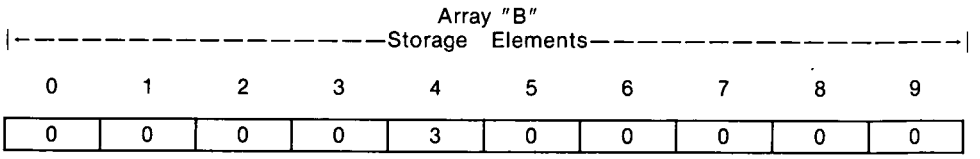



```
10 OUTPUT 709;"REAL A"
20 OUTPUT 709;"LET A=4"
```

a. Data entered by the LET command in Program 1

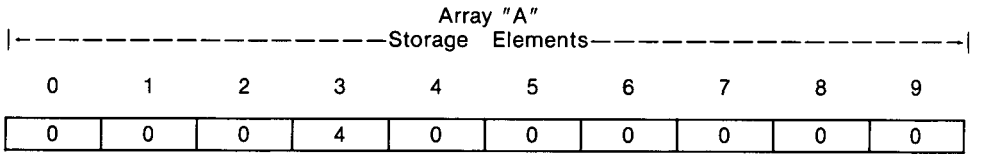



```
20 OUTPUT 709;"VWRITE A 1,2,3,4,5"
```



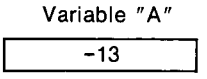

```
30 OUTPUT 709;LET B(4)=A(2)"
```

b. Data entered by the LET command in Program 2




```
20 OUTPUT 709;"A(3)=4"
```

c. Data entered by the LET command in Program 3




```
20 OUTPUT 709;"A=BINCMP(12)"
```

d. Data entered by the LET command in Program 4

Figure 6-6. Entering Data into Memory with the LET Command

www.valuetronics.com

The VWRITE Command

The VWRITE command enables you to write data to a previously defined array, array element, or variable, or write data from one array to another. The VWRITE command can only be used with arrays and variables declared by the DIM, REAL, or INTEGER commands. VWRITE cannot write to PACKED arrays.

To understand how to use the command, let's look at the command syntax:

VWRITE *array item_list* or *array (index) number* or *array(d)[(index)] array(s)[(index)]* or *variable number* or *variable(d) variable(s)*

VWRITE - This is the command header which is entered in upper or lower case.

array - This is the name of the array declared by the DIM, REAL, or INTEGER command that data is written to. After VWRITE is executed, the index pointer is set 1 + the last element written to.

item_list - This is the data written to the array specified by *array*. The data in the item list must be separated by a space or a comma and the list can contain no more than 10 items.

array (index) - This is the name of the array and the specific array element data is written to. (Parentheses must be included.) After the VWRITE command is executed, the index pointer is set to 1 + specified array element (*index* + 1).

number - This is the data that is written to the array element specified by *array (index)*. Can also be a parenthesized numeric expression.

array(d) [*(index)*] - Destination array written to by the source array (*array(s)*). The destination array must be the same size or larger than the source array. Specifying *index* copies data to a specific element in the destination array from a specific element in the source array. After VWRITE is executed, the index pointer is set to 1 + the last element written to, or 1 + the specified element (*index* + 1).

array(s) [*(index)*] - Source array whose contents are written to the destination array (*array(d)*). Specifying *index* copies data from a specific element in the source array to a specific element in the destination array.

variable - This is the name of the variable declared by the REAL or INTEGER command that data is written to.

number - This is the data written to the variable.

variable(d) - Destination variable written to from the source variable (*variable(s)*).

variable(s) - Source variable from which data is copied to the destination variable (*variable(d)*).

Writing to an Array

To illustrate the use of the VWRITE command, refer to the following example program:

www.valuetronics.com

```
10 OUTPUT 709;"DIM DATPT (4)"
20 OUTPUT 709;"VWRITE DATPT 1,2,3,4,5"
30 END
```

In this program, a REAL array (DATPT) with a maximum index of four is declared in line 10. In line 20, the VWRITE command writes the data 1, 2, 3, 4, 5 to the array DATPT. In this array, 1 is assigned to element 0 of the array, 2 to element 1, 3 to element 2, and so on. An example of how the data appears in the array is given in Figure 6-7.

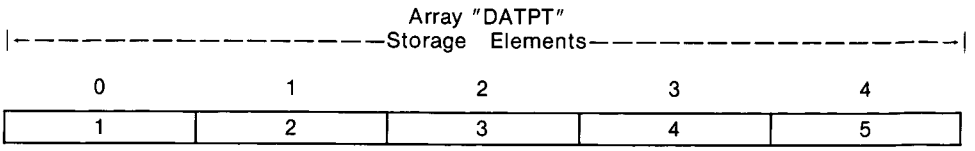


Figure 6-7. Writing Data to an Array with the VWRITE Command

Writing to an Array Element

With the VWRITE command we can write to a single element within an array. For example, if we had executed line 20 in the previous program as:

```
20 OUTPUT 709;"VWRITE DATPT (3),6"
```

the number 6 would have been written to array element 3 as illustrated in Figure 6-8.

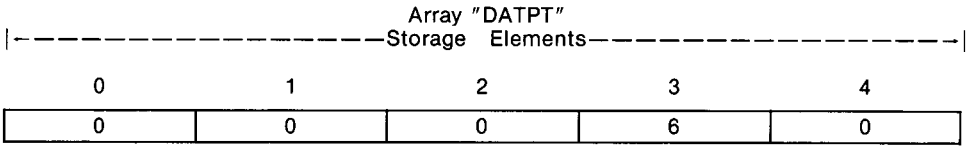


Figure 6-8. Writing Data to an Array Element

The following program shows how the VWRITE command is used to assign a value to a variable:

```
10 OUTPUT 709;"INTEGER A"
20 OUTPUT 709;"VWRITE A,4"
30 END
```

Copying Data to another Array

The following program shows how the VWRITE command can be used to copy data from one array to another.

```
10 OUTPUT 709;"REAL BDATA(4)"
20 OUTPUT 709;"INTEGER ADATA(4)"
30 OUTPUT 709;"VWRITE BDATA 1.1,2.2,3.3,4.4,5.5"
40 OUTPUT 709;"VWRITE ADATA, BDATA"
50 END
```

Notice in this program that the source and destination arrays do not have to be the same storage type; however, the destination array must be the same size or larger than the source array. If you were to read the contents

www.valuetronics.com

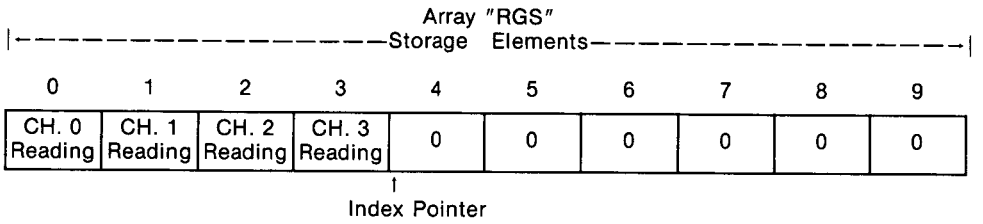
same size or larger than the source array. If you were to read the contents of the destination (INTEGER) array, the array would contain Integer numbers 1,2,3,4,5.

Adding Data to an Array

When an array is declared in the mainframe, the maximum index of the array can be greater than the amount of data to be stored. Refer to the following example:

```
10 OUTPUT 709;"DIM RGS (9)"
20 OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700, INTO RGS"
30 END
```

In line 10 an array (RGS) is declared with a maximum index of 9. This means that a maximum of 10 readings can be stored in that array. When line 20 executes, four measurements are taken (channels 0-3) and stored in the array. Figure 6-9 shows how the measurements are represented within the array.




```
10 OUTPUT 709;"DIM RGS(9)"
20 OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700, INTO RGS"
30 END
```

Figure 6-9. Array with Readings Less Than its Maximum Index

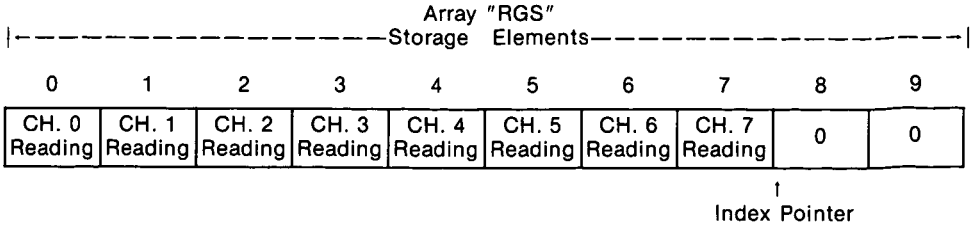
The Index Pointer

Notice the index pointer in Figure 6-9. The index pointer is used by the mainframe's processor to indicate the starting location (element or byte) in the array where the next reading will be stored. When an array is declared by the DIM, REAL, INTEGER, or PACKED command, the index pointer points to array element 0 (or byte 0 for packed readings) as the starting storage location. When data is entered or written into the array, the index pointer shifts from the 0 element to the array element (byte) that will be the starting location for the next group of data to be stored. For example, if we were to add line 25 to the preceding program:

```
25 OUTPUT 709;"CONFMEAS DCV, 304-307, USE 700, INTO RGS"
```

the data would be stored starting with array element 4 as shown in Figure 6-10, with the index pointer indicating the next available storage location.

www.valuetronics.com

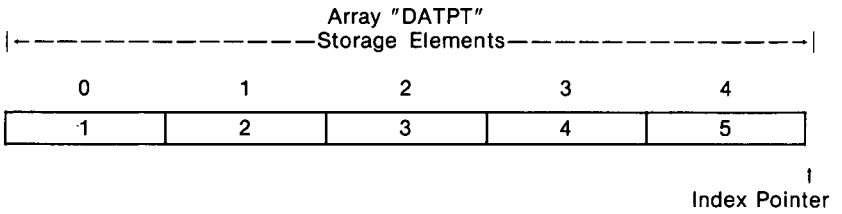



```
10 OUTPUT 709;"DIM RGS(9)"
20 OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700, INTO RGS"
25 OUTPUT 709;"CONFMEAS DCV, 304-307, USE 700, INTO RGS"
30 END
```

Figure 6-10. Storing Additional Data Within an Array

Adding Data with the VWRITE Command

The index pointer is shifted when data is entered into an array with the **INTO** *name* parameter or when data is written to an array with the VWRITE command. The index pointer is not shifted when data is written to an array element with the LET command. Figure 6-11 shows the position of the index pointer within an array following execution of the VWRITE command.




```
10 OUTPUT 709;"DIM DATPT(4)"
20 OUTPUT 709;"VWRITE DATPT 1,2,3,4,5"
30 END
```

Figure 6-11. Shifting the Index Pointer with the VWRITE Command

The INDEX? Command

When data from several commands are entered into the same array, the position of the index pointer becomes important. For example, in Figure 6-10, the index pointer indicates that the starting element in the array RGS is element 8. If we were to add another program line identical to lines 20 and 25 and then run the program in the figure as:

```
10 OUTPUT 709;"DIM RGS (9)"
20 OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700, INTO RGS"
25 OUTPUT 709;"CONFMEAS DCV, 304-307, USE 700, INTO RGS"
26 OUTPUT 709;"CONFMEAS DCV, 308-311, USE 700, INTO RGS"
30 END
```

the following error code and message will appear:

```
44: CONFMEAS: NOT ENOUGH VARIABLE SPACE
```

When this happens, the program will have stopped executing at line 26 and the index pointer will be set to element 0. Note, however, that the data

www.valuetronics.com

returned in lines 20 and 25 is stored in the array. Since the program detected there was not enough memory space in the array, it stopped executing at line 26, therefore, no data was lost. To avoid similar situations, you can read the position of the index pointer with the INDEX? command to determine the available memory space within an array. The syntax of the INDEX? command is:

INDEX? *name* [**INTO** *name*] or [*fmt*]

To understand how to use the command we'll describe its syntax.

INDEX? - This is the command header and can be entered in either upper or lower case.

name - This is the name of the array whose index setting will be returned.

INTO *name* - This parameter enables you to store the index setting returned in an array, array element, or variable.

fmt - This parameter enables you to return the index setting to the front panel or HP-IB output buffer in the format specified.

Determining Index Pointer Location

The following programs illustrate the use of the INDEX? command. Each program has been used previously, however, the INDEX? command is added to determine the position of the index pointer.

```
10  OUTPUT 709;"REAL RGS (9)"
20  OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700, INTO RGS"
30  OUTPUT 709;"INDEX? RGS"
40  ENTER 709;A
50  PRINT"INDEX POINTER IS AT ARRAY ELEMENT: ";A
60  END

INDEX POINTER IS AT ARRAY ELEMENT: 4
```

This program generates the data and positions the index pointer as indicated in Figure 6-9. This corresponds to the position returned by the INDEX? command in line 30, indicating array element 4 is the location for the next reading stored in RGS.

```
10  OUTPUT 709;"DIM DATPT (4)"
20  OUTPUT 709;"VWRITE DATPT 1,2,3,4,5"
30  OUTPUT 709;"INDEX? DATPT"
40  ENTER 709;A
50  PRINT"INDEX POINTER IS AT ARRAY ELEMENT: ";A
60  END

INDEX POINTER IS AT ARRAY ELEMENT: 5
```


www.valuetronics.com

This program writes the data 1,2,3,4,5 to array DATPT and positions the index pointer as shown in Figure 6-11. This corresponds to the position returned by the INDEX? command in line 30. Note that the command indicates storage element 5 is the location for the next reading to be stored in the array. However, since only five numbers were written to the array, the mainframe did not detect an error condition (error 44) and the pointer was moved accordingly. If six numbers were written to the array, error message 44 would be displayed and the data would not be written to the array.

The INDEX Command

When an array is declared by the DIM, REAL, INTEGER, or PACKED command, the index pointer is set to element (byte) 0 in the array. Data that is entered or written to the array then starts at the 0 element. The INDEX command enables you to enter or write data beginning with any storage element (byte) by shifting the index pointer. The syntax of the INDEX command is:

INDEX *name number*

The syntax and use of the command is described as follows:

INDEX - This is the command header, entered in upper or lower case.

name - This is the name of the array in which the index pointer will be set.

number - This is the storage element to which the index pointer is set. Data is then stored starting at this element.

Setting the Index Pointer

The following program shows how the index pointer is set within an array:

```
10  OUTPUT 709;"INTEGER SET_INDX (4)"
20  OUTPUT 709;"INDEX SET_INDX 2"
30  OUTPUT 709;"INDEX? SET_INDX"
40  ENTER 709;A
50  PRINT A
60  END

 2
```

In line 10 the array SET_INDX is declared and the index pointer is at the position shown in Figure 6-12a. When line 20 executes, the index pointer is set to the position shown in Figure 6-12b. Line 30 verifies the setting. Note that the index pointer can be set to any position greater than or less than its current position. Figure 6-12c shows the INDEX command used to set the index pointer from storage element 2 to element 1.

www.valuetronics.com

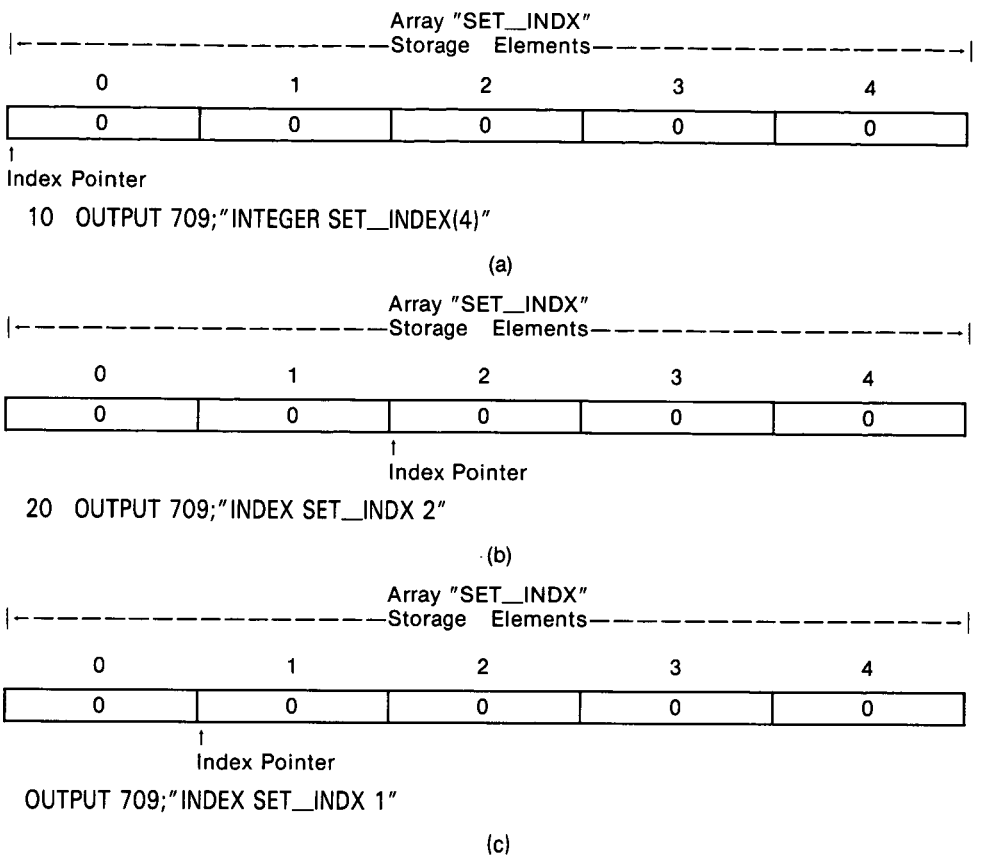


Figure 6-12. Setting the Index Pointer with the INDEX Command

www.valuetronics.com

The following program shows the result of moving the index pointer within an array then writing data to that array. The representation of the data within the array is shown in Figure 6-13.

```
10 OUTPUT 709;"DIM WRITEDAT (4)"
20 OUTPUT 709;"INDEX WRITEDAT 2"
30 OUTPUT 709;"VWRITE WRITEDAT 1,2,3"
40 END
```

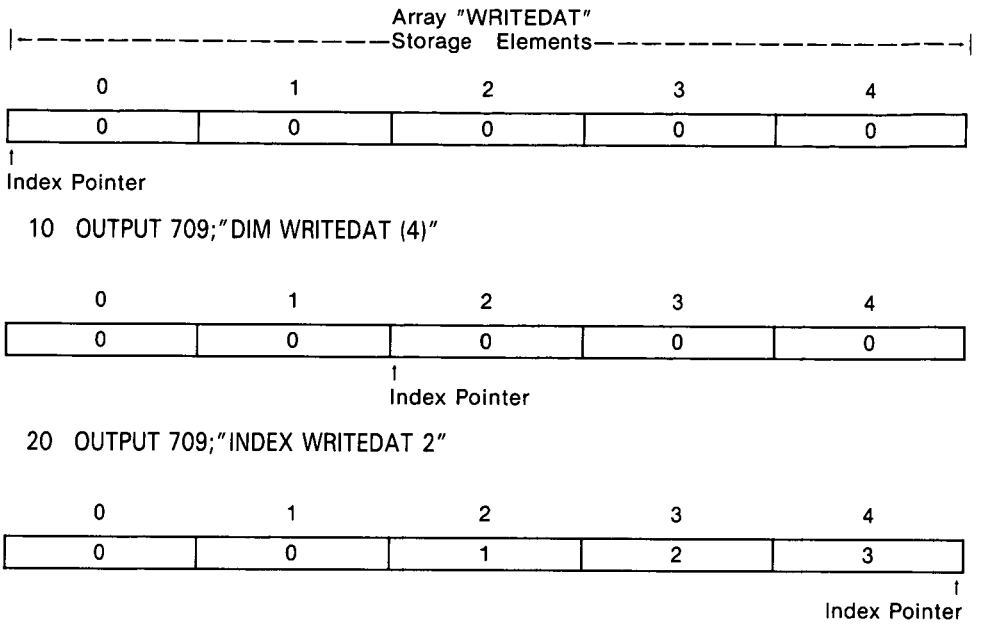


Figure 6-13. Moving the Starting Index

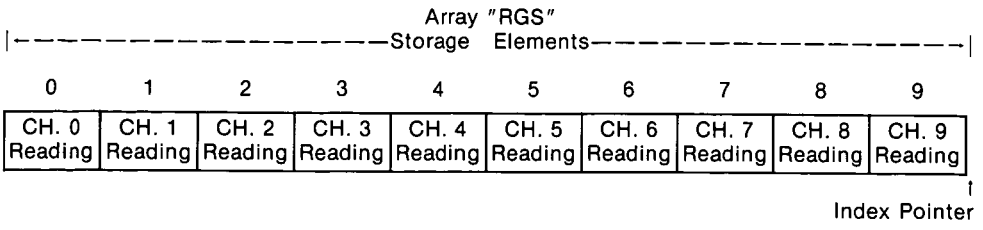
Re-Using Array Space

In certain instances it may be desireable to re-use storage space within an array. One method of recovery without deleting, then redeclaring the array is through the use of the INDEX command. Notice the array RGS in Figure 6-14a. Attempting to store data in this array would result in the error message: 44:NOT ENOUGH VARIABLE SPACE due to the position of the index pointer. However, executing the following command:

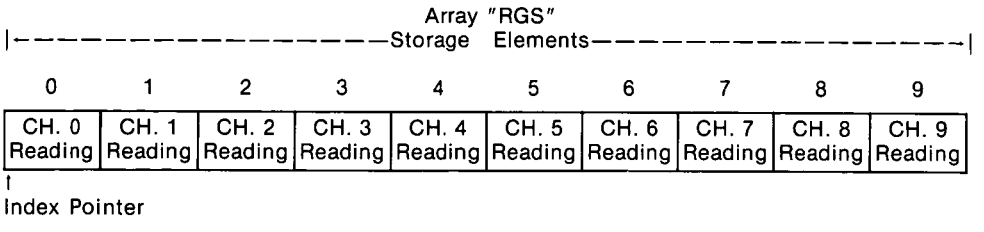
```
20 OUTPUT 709;"INDEX RGS 0"
```

moves the index pointer to the position shown in Figure 6-14b. Thus, when new data is stored in the array, it will overwrite the storage locations as necessary beginning with storage location 0. Note that the index pointer can be "set back" to any location in the array in order to accomodate the readings returned (e.g. INDEX RGS 4; INDEX RGS 7). See Redeclaring and Deleting Arrays and Variables for more information on re-using memory.

www.valuetronics.com



(a)




```
20  OUTPUT 709;"INDEX RGS 0"
```

(b)

Figure 6-14. Re-Using Array Space

Retrieving Data from Memory

The information in this section explains how data is retrieved from a REAL or INTEGER array or variable. Once retrieved, the data can be viewed on the front panel, sent to the HP-IB output buffer, or the data can be stored in another array or variable.

See Packed Data Transfers and Conversions for information on retrieving packed data.

The VREAD Command

The command used to retrieve the data from an array or variable is the VREAD command. The syntax of the VREAD command is:

VREAD *array*[*(index)*] or *variable* or *number* [**INTO** *name*] or [*fmt*]

To understand how to use the command, we'll first trace its syntax.

VREAD - This is the command header entered in upper or lower case.

array [*(index)*] - This is the name of the array whose content will be read. If the optional *(index)* parameter is specified, only the contents of that array element will be returned. Reading an entire array sets the index pointer to zero. Reading an array element does not affect the index pointer.

variable - This is the name of the variable whose value will be read.

number - Number or numeric expression that is evaluated and read into an array, array element, or variable. The result can also be returned to the display and/or HP-IB output buffer.

www.valuetronics.com

INTO *name* - This parameter enables you to store the contents read from an array, array element, or variable into another array, array element, or variable. When the *number* parameter is used, the number (or expression) is evaluated and assigned to the array, array element, or variable specified.

fmt - This parameter returns the data read from memory or the evaluated expression to the front panel or HP-IB output buffer in the format specified (Table 6-2).

The VREAD Default Format

The default format (*fmt*) for the VREAD command is RASC. This means anytime data is read with the VREAD command, the data is converted from its RL64, IN16, or PACKED storage format to its default format when no other format has been specified with the *fmt* parameter, or when the data is not stored in another array.

Reading Data from an Array

This program shows how data is read from an array then displayed on the controller (and on the mainframe front panel).

```
10  REAL Readings(0:4)
20  OUTPUT 709;"DIM RGS (4)"
30  OUTPUT 709;"CONFMEAS DCV, 300-304, USE 700, INTO RGS"
40  OUTPUT 709;"VREAD RGS"
50  ENTER 709;Readings(*)
60  PRINT Readings(*)
70  END

1.316414        5.04837        1.365335        1.319243        1.279616
```

In line 20, an array (RGS) is dimensioned to store five readings. In line 30, five measurements are taken and stored into RGS. In line 40, the VREAD command retrieves the readings from the array and places them in the HP-IB output buffer. Lines 50 and 60 enter the readings from the buffer into the controller where they are displayed. Note that if the mainframe display is enabled (MON) and the FASTDISPlay mode is OFF, each reading can be viewed on the display as it is read.

Note that the data is in the format RASC. Since no format was specified nor was the data stored in another array (line 40), the data was output in the VREAD default format.

When data is retrieved with the VREAD command, the data is not erased within the array. The data remains in the array until it is overwritten or until the array is deleted. Note that when an entire array is read, the index pointer in the array is set to the 0 storage element position. (This is similar to setting the index pointer to 0 with the INDEX command as illustrated in Figure 6-14).

www.valuetronics.com

Reading Data from an Array Element

This program shows how data is retrieved from a single array element.

```
10  INTEGER A
20  OUTPUT 709;"INTEGER DATPT (4)"
30  OUTPUT 709;"VWRITE DATPT 1,2,3,4,5"
40  OUTPUT 709;"VREAD DATPT (2)"
50  ENTER 709;A
60  PRINT A
70  END

3
```

In line 10 a variable is declared in the controller. In lines 20 and 30 an INTEGER array (DATPT) is declared in the mainframe and the numbers 1,2,3,4,5 are written to the array. Line 40 reads the number in storage element 2 and returns the value in the default format RASC. Lines 50 and 60 enter and display the contents of storage element 2 on the controller. This value can also be seen on the display.

When a single array element is read, it is not erased within the array. Note that when the VREAD command retrieves the data from a single array element, the index pointer is not moved from its previous position.

Reading Data from a Variable

The following program shows how the VREAD command is used to retrieve the contents from a variable.

```
10  INTEGER A
20  OUTPUT 709;"INTEGER A"
30  OUTPUT 709;"LET A = 4"
40  OUTPUT 709;"VREAD A"
50  ENTER 709;A
60  PRINT A
70  END

4
```

In line 10 an INTEGER variable is declared in the controller. In lines 20 and 30 a variable (A) is declared in the mainframe and is assigned the number 4. Line 40 reads the mainframe variable and returns the value to the HP-IB output buffer. Lines 50 and 60 enter and display the value of the variable. The value can also be seen on the mainframe display if MON and DISP are enabled. Reading a variable with the VREAD command does not change its value.

Transferring Data Between Arrays

The **INTO** *name* parameter in the VREAD command enables you to specify another array, array element, or variable as the destination for the data read by the command. Within the mainframe, data in a REAL array (declared by the DIM or REAL command) can be transferred to another REAL array or to an INTEGER array (declared by the INTEGER command). Similarly, data in an INTEGER array can be transferred to another INTEGER array or to a REAL array.

www.valuetronics.com

Refer to Packed Data Transfers and Conversions for information on transferring data between PACKED and REAL/INTEGER arrays.

Transfers Between REAL and INTEGER Arrays

Data stored in a REAL mainframe array is stored in the RL64 format shown in Table 6-2. When data is transferred from a REAL array to an INTEGER array it is converted to the IN16 format, then stored. The following program shows how data is transferred from a REAL array to an INTEGER array and how data appears when retrieved from the respective arrays.

```
10   REAL Real_fmt(0:3)
20   INTEGER Integer_fmt(0:3)
30   OUTPUT 709;"REAL RGS (3)"
40   OUTPUT 709;"INTEGER DATPT (3)"
50   OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700, INTO RGS"
60   OUTPUT 709;"VREAD RGS INTO DATPT"
70   OUTPUT 709;"VREAD RGS"
80   ENTER 709; Real_fmt (*)
90   OUTPUT 709:"VREAD DATPT"
100  ENTER 709;Integer_fmt(*)
110  PRINT Real_fmt(*)
120  PRINT
130  PRINT Integer_fmt(*)
140  END
```

```
1.384948          5.04972          1.600995          1.339591

1                 5                2                 1
```

In lines 10 through 40, arrays are declared in the controller and in the mainframe to store four readings. In line 50, the readings are taken and stored into the REAL array RGS. Line 60 reads the data from array RGS, converts it, then stores it into INTEGER array DATPT. Lines 70 through 130 read the data from the arrays then enter and print the readings on the controller.

This program transferred all of the readings from array RGS into array DATPT. If line 60 were executed as:

```
60   OUTPUT 709;"VREAD RGS (2) INTO DATPT (2)"
```

a single reading would be transferred from storage element 2 in RGS to storage element 2 in DATPT.

When data is transferred between REAL and INTEGER arrays, the destination array must have a maximum index the same size or larger than the source array, plus, there must be sufficient space available in the destination array (i.e. SIZE?(dest) - INDEX?(dest) > SIZE?(source)). Again, when data is read and transferred to another array, data still remains in the source array.

www.valuetronics.com

Assigning a Value to a Variable

The following program shows how the VREAD command is used to evaluate an expression and assign the result to a variable. The program is shown using HP Series 200/300 programming syntax; however, keep in mind that each command can also be entered from the front panel.

```
10  OUTPUT 709;"INTEGER A"
20  OUTPUT 709;"VREAD BINIOR(9,12) INTO A"
30  OUTPUT 709;"VREAD A"
40  END
```

Line 10 declares Integer variable A. Line 20 uses the VREAD command to evaluate the inclusive-or of the numbers specified and store the result in variable A. Line 30 reads the variable to show that the result has been stored.

Redeclaring and Deleting Arrays and Variables

In order to recover mainframe memory, arrays and variables can be deleted or redeclared. When redeclaring an array, only the maximum index of the array is changed. The type of the array (e.g. REAL, INTEGER, PACKED) remains the same and cannot be changed unless the array is deleted. Once an array is declared it cannot be changed to a variable by redeclaring the array and omitting its maximum index. Variables that have been declared can only be deleted. A variable cannot be changed to an array by specifying a maximum index.

The following information shows you how to redeclare and delete arrays and variables. First, however, this section shows you how to determine the size of an array then how to obtain a listing of all arrays and variables currently declared.

The SIZE? Command

For REAL and INTEGER arrays, the SIZE? command returns the number of readings that can be stored in the specified array. It should be emphasized that the size of an array is not the same as its maximum index. The SIZE? of an array is equivalent to the total number of storage elements, or readings, that can be stored. The maximum index of an array also determines how many readings can be stored, however; since all mainframe arrays have a starting index of 0, the maximum index of an array is one less than the "size" of the array. The SIZE? command has the following syntax:

SIZE? *name* [**INTO** *name*] or [*fmt*]

SIZE? - This is the command header and is entered in upper or lower case.

name - This is the name of the array whose size is returned.

INTO *name* - This parameter enables you to store the size returned in an array, array element, or variable.

fmt - This parameter enables you to return the size to the front panel or HP-IB output buffer in the format specified.

www.valuetronics.com

The relationship of the maximum index of an array to its size is illustrated in the following example:

```
10  OUTPUT 709;"REAL RGS(4)"
20  OUTPUT 709;"SIZE? RGS"
30  ENTER 709;A
40  PRINT A
50  END

5
```

In line 10 a REAL array is declared with a maximum index of 4. This means that up to five readings can be stored in the array since the starting index is 0. The size of the array is identified in line 20. Note that the SIZE? command always returns the total number of readings than can be stored in the array regardless of the number of readings currently in the array.

See Managing Packed Data for information on the SIZE? of PACKED arrays.

The CAT Command

The name, type, size, and total number of all arrays and variables declared in mainframe memory can be determined with the HP 3852A CAT command. In addition to identifying arrays and variables, the CAT command also lists all HP 3852A subroutines stored in memory.

The syntax of the CAT command consists only of the command header CAT. The following program demonstrates the use of the CAT command. The program declares several arrays and variables then uses the CAT command to "catalog" them. The program also illustrates how the data can be entered and displayed on an HP Series 200 or 300 computer.

```
10   INTEGER N
20   DIM Name$[20],Kind$[20]
30   OUTPUT 709;"REAL RGS(9),DAT1"
40   OUTPUT 709;"INTEGER DATREAD(9),A"
50   OUTPUT 709;"CAT"
60   ENTER 709;N
70   PRINT "TOTAL NUMBER OF ARRAYS AND VARIABLES IN MAINFRAME = ";N
80   PRINT
90   PRINT "      NAME      TYPE      SIZE"
100  PRINT
110  FOR I = 1 TO N
120  ENTER 709;Name$,Kind$
130  PRINT Name$,Kind$
140  NEXT I
150  END
```

In lines 10 and 20, variables are declared in order to store the data returned by the CAT command. Lines 30 and 40 declare REAL and INTEGER arrays and variables in mainframe memory. The CAT command

www.valuetronics.com

is executed in line 50 and lines 60 through 140 enter and display all mainframe arrays and variables on the controller CRT. The output based on this program is shown below.

```
TOTAL NUMBER OF ARRAYS AND VARIABLES IN MAINFRAME = 4
        NAME          TYPE          SIZE
"A              "   "INT            0"
"DATREAD        "   "IARR          10"
"DAT1           "   "REAL           0"
"RGS            "   "RARR          10"
```

Notice in the output the CAT command indicates that a total of four arrays and variables are currently declared in the mainframe. The command output gives the name of each array and variable, the type of each array and variable, and its size. Note that the size (number of storage elements) is returned rather than the maximum index. The last array or variable (or subroutine) declared is also the first one cataloged. Table 6-5 defines name, type, and size information associated with each method of mainframe data storage.

Table 6-5. Data Storage Information Returned by the CAT Command

CATEGORY	NAME	TYPE	SIZE
INTEGER array	array name	IARR	number of array elements (readings)
REAL array	array name	RARR	number of array elements (readings)
PACKED array	array name	PARR	number of bytes (no data stored)
			number of readings in particular format (data stored)
INTEGER variable	variable name	INT	0 (one storage element (reading))
REAL variable	variable name	REAL	0 (one storage element (reading))
Subroutine	sub name	SUB	length of subroutine in bytes

Redeclaring Arrays

When you redeclare an array, you change the maximum index of the array. The array is redeclared by executing the DIM, REAL, or INTEGER command while specifying the same array name and setting the maximum index as desired. The following examples show how arrays are redeclared using the DIM, REAL, and INTEGER commands. See Managing Packed Data for information on redeclaring Packed arrays.

```
10  OUTPUT 709;"DIM RGS(9)"
20  OUTPUT 709;"DIM RGS(4)"
```

In line 10 we initially declare the array RGS to store 10 readings. The array is redeclared in line 20 to store five readings.

```
30  OUTPUT 709;"REAL DATRGS(4)"
40  OUTPUT 709;"REAL DATRGS(9)"
```

In line 30 the REAL command is used to declare an array to store five readings. The array is redeclared in line 40 to store 10 readings.

www.valuetronics.com

```
50  OUTPUT 709;"INTEGER MESTOR(4)"
60  OUTPUT 709;"INTEGER MESTOR(0)"
```

In line 50 an INTEGER array is declared to store five readings. In line 60 the array is redeclared to store one reading. Note that MESTOR is now a single element array and not a variable.

Based on the examples presented above, the following restrictions apply when redeclaring arrays:

1. A maximum index must be specified each time an array is redeclared. An array cannot be changed to a variable by omitting the maximum index. Conversely, a variable cannot be changed to an array by specifying a maximum index.
2. When an array is redeclared, it can become larger or smaller than it presently is.
3. When redeclaring an array it must remain the same type. If a REAL array is declared, it must remain a REAL array if redeclared. Note that an array declared with the REAL command can be redeclared with the DIM command and vice versa.
4. An array can be redeclared an any point after it is first declared.
5. Any data in an array is erased when the array is redeclared.

Recovering Memory

Within the HP 3852A, all arrays, variables, and subroutines are deleted simultaneously when the appropriate command is executed. Although a single array or variable cannot be deleted, memory can still be recovered using the DELVAR command. The command has the following syntax:

DELVAR *name*

where *name* is the name of the array. The DELVAR command releases all memory allocated to data storage, plus 16 bytes of overhead for the specified array. The name and storage format of the array remain defined, however; there is no associated memory available. Since the name and format remain, the name can't be used again with a different storage format. However, it can be used when redeclaring the array.

The use of the DELVAR command is illustrated as follows.

```
10  OUTPUT 709;"INTEGER RGS(9)"
.
.
.
70  OUTPUT 709;"DELVAR RGS"
```

In line 10 an INTEGER array is declared to store 10 readings. In line 70, the memory associated with data storage (28 bytes - minimum) and 16 bytes of overhead for the array RGS is released and becomes part of the available mainframe memory. The array can be redeclared provided it remains an INTEGER array.

www.valuetronics.com

Deleting Arrays and Variables

All arrays and variables (and subroutines) within the mainframe are deleted simultaneously when the RST (reset), RST HARD, or the SCRATCH command is executed. The syntax of these commands is:

RST [*slot*]
RST HARD
SCRATCH

If a mainframe or extender slot is not specified in the RST command, the mainframe and all accessories are reset and mainframe memory is cleared. Note that the RST key on the mainframe front panel is an immediate-execute key. This means the mainframe resets immediately after the key is pressed.

Array Operations

The HP 3852A's MAT command enables you to initialize arrays to specific values and perform arithmetic operations between arrays or between arrays and numeric expressions. The syntax of the MAT command is given below:

MAT *array3* = *(expression)*
MAT *array3* = *array1 operator array2*
MAT *array3* = *(expression) operator array2*
MAT *array3* = *array2 operator (expression)*

array1, array2, array3 - Real or Integer arrays on which the operation is performed. All arrays must be the same size.

(expression) - a number or a numeric expression.

operator - math operation performed on the arrays. Operators include +, -, *, /. The operation is performed on every array element, and the results are placed in the corresponding elements of the result array. For example, **MAT** *array3* = *array1* * *array2* multiplies element i of *array1* by element i of *array2* and places the product in element i of *array3*.

Refer to the Command Reference Manual for additional information and an example on the use of the MAT command.

Managing Packed Data

This section of the chapter describes how data is managed throughout the mainframe in packed formats. A packed format is a format in which the accessory presents its data to the mainframe processor. Packed formats are accessory and command/function dependent, and packed readings can be from 2 bytes to 8 bytes in length. When readings are stored in a REAL or INTEGER array or output in an ASCII or binary format (Table 6-2), the readings are converted by the mainframe processor from their packed (original) format to the format specified. When data remains in its packed format, it is stored and output at faster rates since no conversion occurs.

Specifying a Packed Format

To return data from the accessory directly to the HP-IB output buffer in its packed format, you need to specify PACK for the fmt parameter in the command returning the data. Refer to the following command:

www.valuetronics.com

```
10 OUTPUT 709;"CONFMEAS DCV, 200-202, USE 700, PACK"
```

When this command executes, each DC voltage measurement on channels 0 through 2 will be returned to the HP-IB output buffer in the packed voltmeter format for the command and parameter specified. Tables 6-6 and 6-7 identify the reading lengths, bit patterns, and unpacking routines for the packed data formats of the various HP 3852A accessories.

Note that data in a packed format can either be stored in the mainframe or sent to the HP-IB output buffer. A packed reading cannot be displayed on the mainframe.

Declaring Packed Arrays

To store packed data in the mainframe you must first declare a packed array. Packed arrays are declared with the PACKED command. The PACKED command can be executed from the front panel, over the HP-IB, or within an HP 3852A subroutine. Packed arrays are global which means regardless of where the array was declared (e.g. front panel), the array can be accessed by any command returning data. The syntax of the PACKED command is:

PACKED *name (max_index)* [*name (max_index)...*]

To understand more about the command, we'll trace its syntax.

PACKED - This is the command header which is entered in either upper or lower case.

name - This parameter specifies the name of the array. The name used must follow the conventions described under Array and Variable Names.

(max_index) - This parameter specifies the maximum index of the array. The maximum index of a packed array is the maximum number of bytes of mainframe memory required to store data in the desired packed format. Before you specify a maximum index, it is important that you know the number of bytes per reading of the packed data you plan to store, and the number of readings you plan to take. For example, to store 100 DC voltage readings from the HP 44702A/B, the maximum index would have to be at least 199. This is because each packed reading is two bytes long (Table 6-6). Also, the first byte in a packed array is byte 0, therefore, 0 through 199 are 200 bytes. When the maximum index is specified, the parentheses must be included. Note that you can only declare packed arrays. You cannot declare packed variables.

[*name (max_index)*] - The brackets [ ] around these parameters indicate that one or more arrays can be declared with a single PACKED command. The brackets are not included, however.

Using the PACKED Command

The following paragraphs explain the use of the PACKED command. Recall that the maximum index of a packed array must specify a number of bytes equal to, or greater than, the number of bytes per reading times the total number of readings. Therefore, before declaring the array:

www.valuetronics.com

1. Determine the number of bytes/reading for the packed format you will be storing (Table 6-6).

2. Determine the number of readings you want to store then multiply that number by the bytes/reading determined in step 1. The result is the minimum number of bytes you must allocate to that array to store all of the readings.

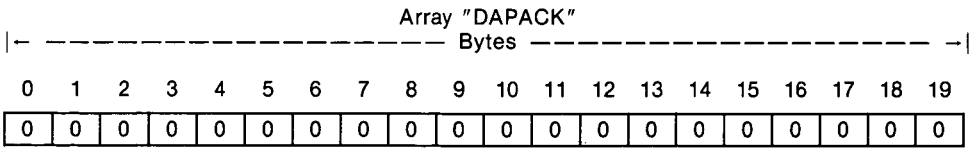
Note that there are 72 bytes of "overhead" associated with each packed array declared. The overhead contains information such as the array name, type, its maximum index, etc,. For example, a packed array declared to store 100 packed DC voltage readings from the HP 44702A/B would use 272 bytes of memory. 200 bytes are specified by the *max_index* parameter (100 readings at 2 bytes/reading) plus the 72 bytes of overhead.

Keep in mind that the minimum amount of memory allocated by the mainframe for data storage is 28 bytes. Thus, for some packed formats, it will require as much memory to store one reading in a single element array as it would several readings in a multi-element array (see Memory Size Vs Reading Length).

The use of the PACKED command is illustrated in the following program line:

```
20 OUTPUT 709;"PACKED DAPACK (19)"
```

When this line is executed, a packed array (DAPACK) with a maximum index of 19 is declared in mainframe memory (Figure 6-15). Recall that the first byte is byte 0, therefore, a maximum index of 19 represents 20 bytes of memory. When a packed array is declared, each byte has a value of 0 until a reading is stored in that byte.




```
10 OUTPUT 709;"PACKED DAPACK (19)"
```

Note:
A maximum of 10 2-byte readings can be stored or a maximum of 5 4-byte readings can be stored.

Figure 6-15. Declaring an Array with the PACKED Command

When packed data is stored in an array declared by the PACKED command, the array must be declared before the data is returned as illustrated in the following program:

www.valuetronics.com

```
10 OUTPUT 709;"PACKED DAT (39)"
20 OUTPUT 709;"CONFMEAS DCV, 200-209, USE 500, INTO DAT"
30 END
```

(Note - based on the command and function specified in the above program, a maximum of 20 packed readings can be stored if the voltmeter in slot 5 is an HP 44702A/B, or a maximum of 10 packed readings can be stored if the voltmeter is an HP 44701A.)

More than one packed array can be declared with a single PACKED command as illustrated in the following example:

```
10 OUTPUT 709;"PACKED DAPACK(19), DAT (39)"
```

Entering Packed Data into Memory

Packed data is entered into the array or into specific bytes of the array with the **INTO** *name* parameter. This parameter appears in those commands which perform measurements and transfer data such as the CONFMEAS and VREAD commands shown below:

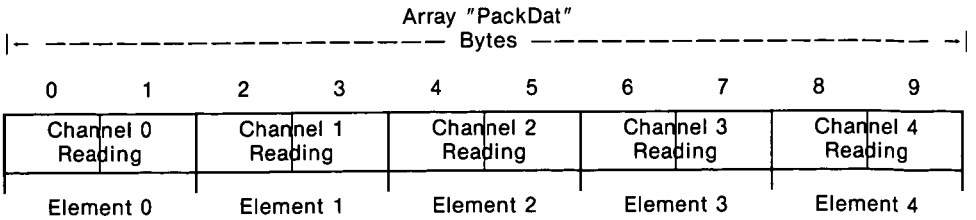
CONFMEAS *function ch_list* [**NSCAN** *number*][**USE** *ch*][**INTO** *name*] or [*fmt*]

VREAD *array* [*(index)*] or *variable* or *number* [**INTO** *name*] or [*fmt*]

The use of the **INTO** *name* parameter is demonstrated in the following program:

```
10 OUTPUT 709;"PACKED PACKDAT (9)"
20 OUTPUT 709;"CONFMEAS DCV, 200-204, USE 500, INTO PACKDAT"
30 END
```

When this program executes, line 10 declares a packed array (PACKDAT) in mainframe memory that is 10 bytes long (first byte is byte 0). The array can store up to 5 packed readings two bytes in length. When line 20 executes, the voltmeter in slot 5 (HP 44702A/B) is configured then it measures the DC voltage on channels 0 through 4. As the measurements are taken they are stored in the mainframe array PACKDAT as represented in Figure 6-16 (readings are 2 bytes in length).




```
10 OUTPUT 709;"PACKED PACKDAT(9)"
20 OUTPUT 709;"CONFMEAS DCV, 200-204, USE 500, INTO PACKDAT"
30 END
```

Figure 6-16. Storing Packed Data with the INTO name Parameter

www.valuetronics.com

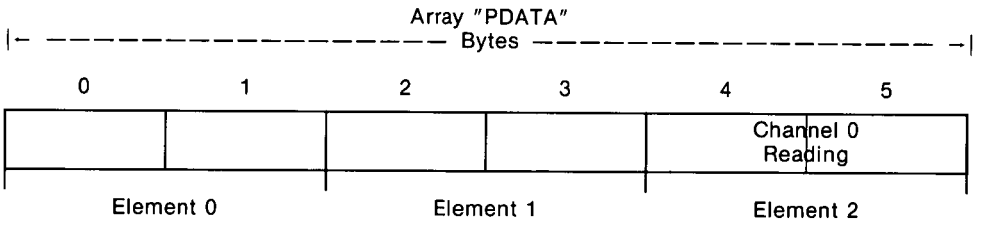
The next program shows how a single packed reading from the HP 44702A/B can be returned to a specific location within an array.

```
10 OUTPUT 709;"PACKED PDATA (5)"
20 OUTPUT 709;"CONFMEAS DCV, 200, USE 500, INTO PDATA (2)"
30 END
```

When this program runs, the packed reading on channel 0 is stored in element 2 (bytes 4 and 5) of array PDATA (Figure 6-17).

Packed Array Elements

Notice the location specified (PDATA (2)) in line 20 of the preceeding program. When packed data is stored into an array, the mainframe processor groups the number of array bytes equivalent to the number of bytes/reading of the packed format into array elements. As the previous program executes, the mainframe determines from the voltmeter in the slot specified and the command/function specified that the length of the packed readings will be two bytes each. Given the number of bytes (maximum index) of the packed array and the byte length of each reading, the mainframe separates the array bytes into elements. By specifying "element" 2 in line 20, the reading is stored in bytes 4 and 5 which comprise the third element of the array.




```
10 OUTPUT 709;"PACKED PDATA(5)"
20 OUTPUT 709;"CONFMEAS DCV, 200, USE 500, INTO PDATA(2)"
30 END
```

Figure 6-17. Storing Packed Data in Specific Bytes

The only method of entering data into packed arrays is through the **INTO** *name* parameter. You cannot write data to a packed array with the VWRITE or LET commands.

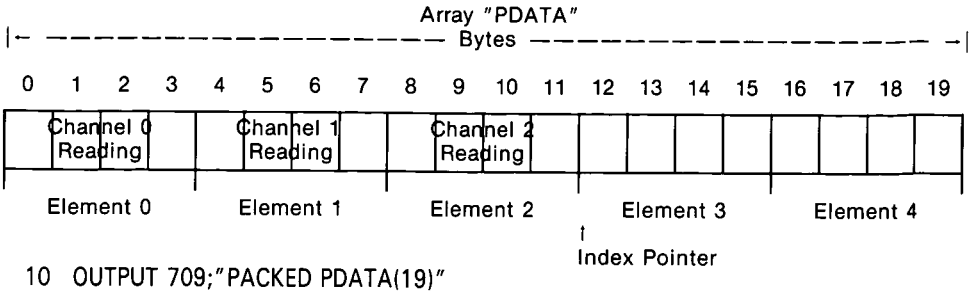
Adding Data to a Packed Array

When a packed array is declared in the mainframe, the maximum index of the array can be greater than the number of bytes of data returned by a given command. For example, refer to the following program:

```
10 OUTPUT 709;"PACKED PDATA(19)"
20 OUTPUT 709;"CONFMEAS DCV, 200-202, USE 700, INTO PDATA"
30 END
```


www.valuetronics.com

In line 10 a packed array is declared with a maximum index of 19. If the voltmeter in slot 7 is an HP 44701A, five packed readings each four bytes in length can be stored (Figure 6-18). When line 20 is executed, DC voltage measurements are taken on channels 0-2 and stored into the array.




```
10 OUTPUT 709;"PACKED PDATA(19)"
20 OUTPUT 709;"CONFMEAS DCV, 200-202, USE 500, INTO PDATA"
30 END
```

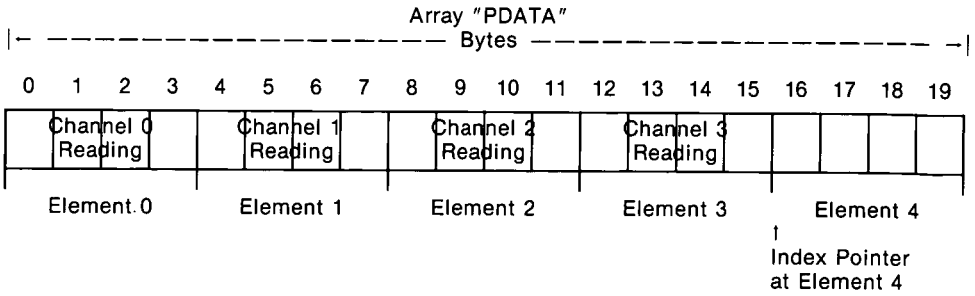
Figure 6-18. Packed Array with Readings Less Than its Maximum Index

The Index Pointer - Packed Arrays

Notice the index pointer in Figure 6-18. The index pointer is used by the mainframe's processor to indicate the starting location (element) in the array where the next reading will be stored. When an array is declared by the PACKED command, the index pointer points to byte 0 as the starting location. As data is stored in the array, elements are defined and the index pointer shifts from byte 0 to the element that will be the starting location for the next reading to be stored. For example, if you were to add line 24 to the preceeding program:

```
24 OUTPUT 709;"CONFMEAS DCV, 203, USE 700, INTO PDATA"
```

the reading would be stored in array element 3 as shown in Figure 6-19, with the index pointer indicating the next available storage element.




```
24 OUTPUT 709;"CONFMEAS DCV, 203, USE 700, INTO PDATA"
```

Figure 6-19. Adding Packed Data to an Array

www.valuetronics.com

The INDEX? and INDEX Commands

When packed readings from several commands are entered into the same array, the position of the index pointer becomes important. For instance, in Figure 6-19 the index pointer indicates the starting element for the next reading returned is element 4. If we wanted to take measurements on two additional channels as with the following program line:

```
27  OUTPUT 709;"CONFMEAS DCV, 204-205, USE 700, INTO PDATA"
```

the measurements on channels 4 and 5 would not be performed since the mainframe would have determined there was not enough array space. To avoid this, you can read the position of the index pointer with the INDEX? command before attempting to store data. The INDEX? command has the syntax:

INDEX? *name* [**INTO** *name*] or [*fmt*]

If we executed the INDEX? command after line 20 as follows, the command will return the position of the index pointer that is consistant with the position shown in Figure 6-18.

```
21  OUTPUT 709;"INDEX? PDATA"
3
```

Notice that once data is stored in a packed array, the index pointer indicates the next array **element** rather than the bytes in which the next reading wll be stored. Executing the INDEX? command after line 24 would show how the index pointer moved again.

If, based on the position of the index pointer, it is determined there is not enough room in the array to store the next group of readings, the INDEX command can be used to shift the index pointer to an element in the array to accomodate the readings. The INDEX command has the syntax:

INDEX *name number*

where *name* is the name of the packed array and *number* is the element within the array the index pointer is set to. Referring to Figure 6-19, the index pointer is at element 4 of the array. In order to store two additional readings, the index pointer will have to be set back at least to element 3 of the array. Executing the INDEX command:

```
33  OUTPUT 709;"INDEX PDATA 3"
```

moves the index pointer to element 3 and the array can now accomodate two new readings. Note that following the repositioning of the index pointer, any data currently in the elements to the "right" of the index pointer is overwritten as new data is stored.

www.valuetronics.com

Mixing Packed Readings

When a packed array is initially declared, a packed reading of any length (e.g. 2, 4, 6 bytes) can be stored in the array. Since the mainframe defines array elements based on the number of bytes per reading, each reading in the array must be the same length. For example, if a packed array is declared and packed data two bytes/reading is stored, the mainframe defines elements each two bytes in length. This means, if packed DC voltage readings from an HP 44702A/B are currently in an array (2 bytes/reading), DC voltage readings from an HP 44701A voltmeter (4 bytes/reading) cannot be stored in the same array.

Another factor that must be considered before storing packed readings is the bit pattern of the data returned. Note that you can not store HP 44702A/B resistance measurements in the same PACKED array as voltage measurements even though both are two bytes per reading. As data is initially stored in a PACKED array, the address of a mainframe conversion routine is stored as part of the array overhead. The conversion routine is then recalled when packed data is read from the array (VREAD command). Only one conversion routine is used per array. Thus, when you attempt to store additional data into the array, the mainframe first determines if the conversion routine applies. If it does, the data is stored. If not, an error message is returned.

Before attempting to store packed readings, refer to Table 6-7. Table 6-7 contains conversion routines that can be entered into an HP Series 200 or 300 controller to unpack the data external to the mainframe. If the conversion routine is the same for the command and function specified, different kinds of data can be stored in the same array. If not, separate PACKED arrays must be used.

Attempting to store packed data of different lengths or with different bit patterns will result in the error message: ERROR 35: DIFFERENT PACKED TYPES.

Redeclaring and Deleting Packed Arrays

In addition to recovering array (mainframe) memory by moving the index pointer, packed arrays can be redeclared or deleted. When redeclaring a packed array, only its maximum index is changed. An array declared as a packed array must remain a packed array. It cannot be changed to a REAL or INTEGER array.

The following paragraphs show you how packed arrays are redeclared and deleted.

Packed Array Size

The "size" of a packed array depends on whether or not data is stored in the array. The command which returns the size of packed arrays is the

www.valuetronics.com

SIZE? command. As described previously, the syntax of the SIZE? command is:

SIZE? *name* **[INTO** *name*] or [*fmt*]

When a packed array is declared but does not contain data, the size of the array is the number of bytes allocated to that array (i.e. maximum index). Note, however, that since the first byte of a packed array is byte 0, the maximum index of an array will always be one less than its size. The following program compares the size of a packed array to its maximum index when no data is stored.

```
10  OUTPUT 709;"PACKDATA (9)"
20  OUTPUT 709;"SIZE? PACKDATA"
30  ENTER 709;A
40  PRINT A
50  END

    10
```

In line 10 a packed array is declared with a maximum index of 9. This means 10 bytes of memory are allocated to the array (first byte is byte 0). The size of the array is determined in line 20 and printed below the program.

When data is stored in a packed array, the size of the array is equal to the number of readings that can be stored in the array in that packed format. For example, modifying the program above we get:

```
10  OUTPUT 709;"PACKDATA (9)"
20  OUTPUT 709;"CONFMEAS DCV, 200, USE 500, INTO PACKDATA"
30  OUTPUT 709;"SIZE? PACKDATA"
40  ENTER 709;A
50  PRINT A
60  END

    5
```

Again in line 10 a packed array is declared with a maximum index of 9. In line 20, a DC voltage is stored into the array. The program assumes an HP 44702A or HP 44702B is installed in slot 5. Since the packed reading length for the HP 44702A/B with the command CONFMEAS and function DCV is 2 bytes/reading, the SIZE? command indicates that a maximum of five readings can be stored in the array. Note that when data is stored in a packed array, the SIZE? command indicates the maximum number of readings that can be stored in the array, regardless of the number of readings currently in the array.

www.valuetronics.com

The HP 3852A CAT command catalogs all mainframe arrays, variables, and subroutines currently declared (see The CAT Command). Keep in mind that the size reported by the CAT command for a packed array also depends on whether or not data is in the array as described above.

Redeclaring PACKED Arrays

When you redeclare a packed array, you change the maximum index of the array. A packed array is redeclared by executing the PACKED command and setting the desired maximum index as indicated in the following example:

```
10  OUTPUT 709;"PACKED PACKDAT (9)"
20  OUTPUT 709;"PACKED PACKED (39)"
```

In line 10 a packed array 10 bytes long is declared. In line 20 the array is redeclared and given a maximum index of 40 bytes.

Restrictions for redeclaring PACKED arrays are as follows:

1. A maximum index must be specified each time a packed array is redeclared.
2. When a packed array is redeclared, it can become larger or smaller than it presently is. Also, any packed data format can be stored.
3. When a packed array is redeclared, it must remain a packed array.
4. A packed array can be redeclared at any point after it is first declared.
5. Any data in the packed array is erased when the array is redeclared. Following redeclaration, the SIZE? and CAT commands return bytes.

Deleting Packed Arrays

The procedure for recovering array space and deleting packed arrays is the same as for REAL and INTEGER arrays. Refer to Recovering Memory and Deleting Arrays and Variables for information.

NOTE

The portion of mainframe memory (including extended memory) in which arrays and variables are declared is volatile. This means that if the mainframe is turned off or accidently loses power, all arrays, variables, and subroutines are erased. Resetting the instrument with the RST key on the front panel or sending the RST, RST HARD, or SCRATCH commands also deletes all previously declared arrays, variables, and subroutines.

---

www.valuetronics.com

Packed Data Transfer and Conversions

This section describes how packed data is retrieved from an array and transferred to the HP-IB output buffer in either its packed format or in a converted format. It also describes how data is transferred from a packed array to an INTEGER or REAL array and vice versa. System throughput (speed) as a function of the various data formats is also covered in this section as are conversion routines that enable a Hewlett-Packard Series 200/300 computer to unpack HP 3852A data.

Retrieving Packed Data from Memory

The command used to retrieve packed data from an array and transfer it to the output buffer or to another array is the VREAD command. When reading a packed array, the syntax of the VREAD command is:

VREAD *array* [*(index)*][**INTO** *name*] or [*fmt*]

When packed data is retrieved with the VREAD command, the data is not erased within the array. The data remains until it is overwritten or until the array is deleted.

Transferring Packed Data to the Output Buffer

When reading the contents of a PACKED array with the VREAD command, specifying PACK for the format *fmt* will pass the data to the HP-IB output buffer in the same (packed) format in which it was stored. For example, refer to the following program segment:

```
10  OUTPUT 709;"PACKED PDATA (9)"
20  OUTPUT 709;"CONFMEAS DCV, 200-204, USE 500, INTO PDATA"
30  OUTPUT 709;"VREAD PDATA, PACK"
```

In line 10 a packed array is declared to store 5 packed readings from an HP 44702A/B voltmeter. In line 20 the measurements are taken and stored into the array PDATA. Line 30 reads the data from the array and places it into the HP-IB output buffer. At this point the controller would retrieve the data from the buffer and "unpack" the data in order to store or display the measurements. (See Packed Data Conversions for HP Series 200 and Series 300 unpacking routines.)

By specifying a format in Table 6-2 other than PACK, the mainframe will convert the data to that format then place it into the output buffer. For example, if in line 30 of the previous program segment the data is read from the buffer and the format specified is RL64:

```
30  OUTPUT 709;"VREAD PDATA, RL64"
```

The mainframe itself unpacks the data as it converts it to RL64 format. The readings could then be stored directly in an HP Series 200 or Series 300 computer.

www.valuetronics.com

If the format specified is an HP 3852A display format and DISP and MON are enabled, the readings will also appear on the mainframe front panel.

Note that specifying the PACK format when reading a REAL or INTEGER array is equivalent to specifying RL64 or IN16 as the output format. In other words, data in a REAL (RL64) array or INTEGER (IN16) array remains in that format when PACK is specified since the mainframe cannot "repack" the data as it has no way of tracking where it originated.

Transferring Packed Data Between Arrays

With the VREAD command, data within a packed array can be transferred to a REAL or INTEGER array, or to another PACKED array. Transfers of this type represent another method of using the mainframe to unpack the data before it is eventually transferred to the controller. The following paragraphs describe the procedures and prerequisites for these types of data transfers.

Transfers: PACKED to REAL/INTEGER

The following program shows how data is tranferred from a packed array to a REAL or INTEGER array. This type of transfer unpacks the data and stores it in either a REAL (RL64) or INTEGER (IN16) format. In the program, the destination array is a REAL array; however, the transfer procedure would be the same had an INTEGER array been declared.

```
10  REAL Pack_fmt (0:4)
20  OUTPUT 709;"PACKED PACKDAT (9)"
30  OUTPUT 709;"DIM PACTRANS (4)"
40  OUTPUT 709;"CONFMEAS DCV, 300-304, USE 700, INTO PACKDAT"
50  OUTPUT 709;"VREAD PACKDAT INTO PACTRANS"
60  OUTPUT 709;"VREAD PACKTRANS"
70  ENTER 709; Pack_fmt (*)
80  PRINT Pack_fmt (*)
90  END

 1.15365          5.05972          1.376156          1.360389
```

Line 10 declares an array in the controller. Lines 20 and 30 declare a PACKED and a REAL array in the mainframe. In this example, five DC voltage measurements from the HP 44702A/B are to be transferred from the PACKED to the REAL array. Therefore, the maximum index of the PACKED array is 9 (5 readings at 2 bytes/reading, first byte is byte 0) and the maximum index of the REAL array is 4 (five readings, first reading is reading 0). Line 40 performs the measurements then stores them into the PACKED array. Line 50 converts the readings from their packed format to RL64 format as it transfers them to the REAL array. Recall that even though the data is transferred from the PACKED array to the REAL array, data still remains in the PACKED array.

www.valuetronics.com

Note that an entire packed array or a single packed reading can be transferred to a REAL or INTEGER array.

Transfers: REAL/INTEGER to PACKED

The following program shows how data is transferred from a REAL/INTEGER array to a packed array. In the program, the transfer is between a REAL array and a packed array. The transfer procedure would be the same had an INTEGER array been declared.

Transferring data from a REAL or INTEGER array to a packed array does not "repack" the data. In other words, when data in a REAL array (RL64 format) is transferred to a PACKED array, the RL64 format becomes the "packed" format since the REAL array is the source of data for the PACKED array. Similarly, when data in an INTEGER array is transferred to a packed array, the IN16 format becomes the "packed" format.

If data from a REAL or INTEGER array is transferred to a PACKED array, the packed array must be empty or contain data that is in an RL64 or IN16 format.

```
10  OUTPUT 709;"REAL RGS (3)"
20  OUTPUT 709;"PACKED PACKRGS (31)"
30  OUTPUT 709;"CONFMEAS DCV, 300-303, USE 700, INTO RGS"
40  OUTPUT 709;"VREAD RGS INTO PACKRGS"
50  END
```

In line 10, an array (RGS) is declared to store four readings. Line 20 declares a packed array (PACKRGS) to which the readings will be transferred. Notice the maximum index of the PACKRGS array. Since the source of the readings for the PACKED array is the REAL array rather than a voltmeter accessory, the maximum index must be specified based on the number of readings to be transferred times the number of bytes/reading for storage format RL64. Therefore to store four readings at eight bytes/reading (RL64 format), 32 bytes (0-31) must be allocated to the packed array. Line 30 makes the voltage measurements and stores them into the REAL array. Line 40 transfers the readings to the packed array.

An entire array or a single reading within a REAL or INTEGER array can be transferred to a packed array.

Transfers: PACKED to PACKED

The following program shows how data can be transferred between packed arrays. Note that in the program, only one reading is transferred.

```
10  OUTPUT 709;"PACKED RGS(3)"
20  OUTPUT 709;"PACKED RGS1(7)"
30  OUTPUT 709;"CONFMEAS DCV, 100-101, USE 200, INTO RGS"
40  OUTPUT 709;"VREAD RGS(0) INTO RGS1"
50  END
```


www.valuetronics.com

In line 10, a packed array is declared to store two readings (two bytes/reading). In line 20, a packed array is declared into which a single reading will be transferred. Note the maximum index of this array (RGS1). When a single packed reading is transferred to another packed array, the VREAD command converts the reading to RL64 format. Thus the reading becomes eight bytes in length rather than two. Continuing through the program, line 30 takes the measurements and stores the readings into packed array RGS. Line 40 transfers one of the readings. Note that if all readings had been transferred rather than just one (e.g. line 40 executed as):

```
40 OUTPUT 709;"VREAD RGS INTO RGS1"
```

the destination array (RGS1) can be the same size as the source array (RGS(3)) because when **all** readings are transferred, the readings remain in their original packed format (e.g. 2 bytes/reading).

Maximizing System Throughput

System throughput is the transfer of data from an accessory directly to the controller, or to the controller via mainframe memory. The speed at which data is transferred throughout the system depends on the formats used. Generally, data is managed at the fastest rates in packed formats, with the display off, and when the number of format conversions are reduced. Figure 6-20 represents various paths through the mainframe when data is transferred from the digital input and voltmeter accessories. The figure shows generalized transfer rates to mainframe memory and to the controller for the formats available. (The "System Speed Characteristics" section of the HP 3852S Data Acquisition and Control System Data Book contains specific information on reading transfer rates.)

www.valuetronics.com

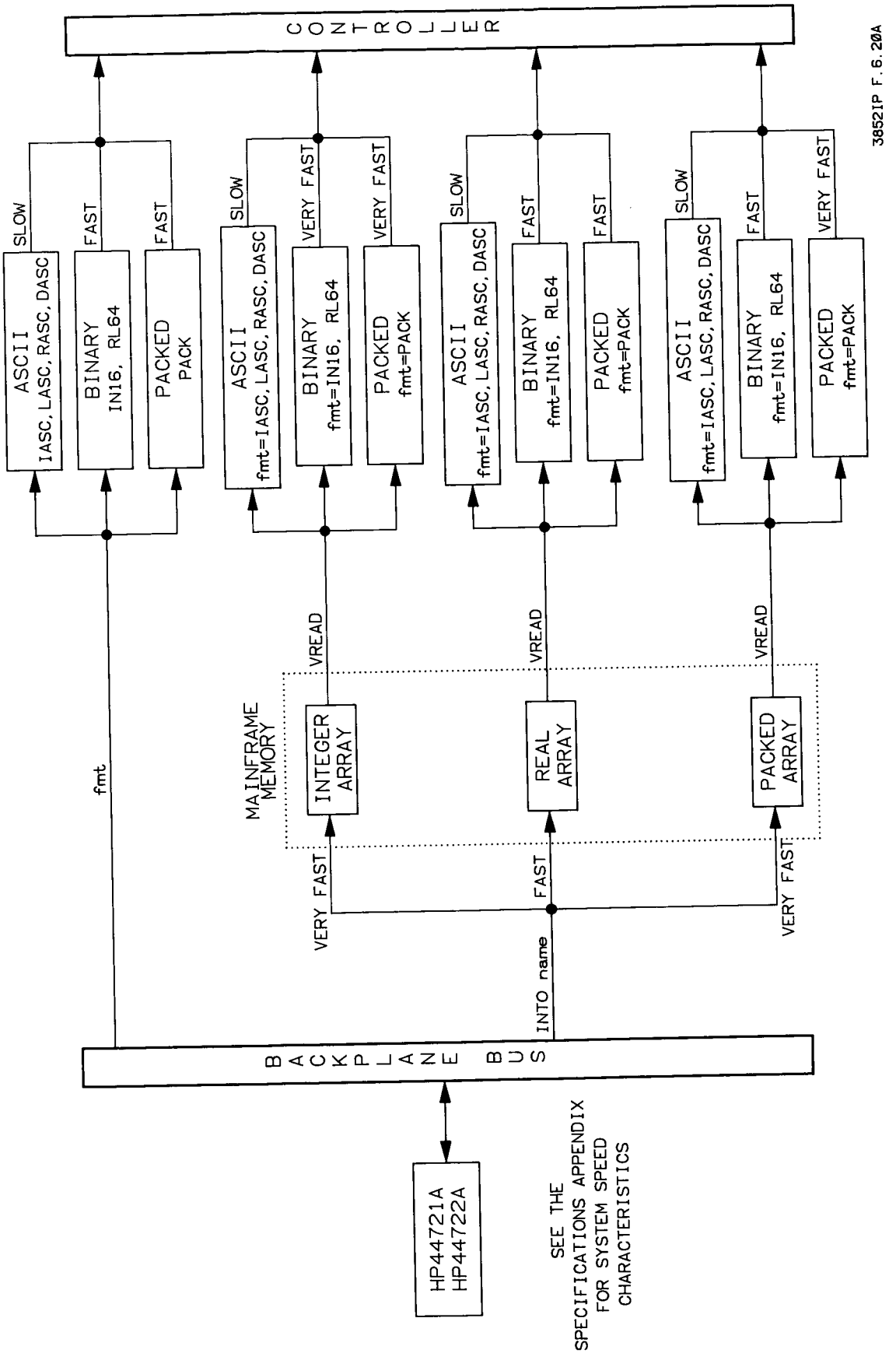


Figure 6-20. HP 3852A System Throughput

www.valuetronics.com

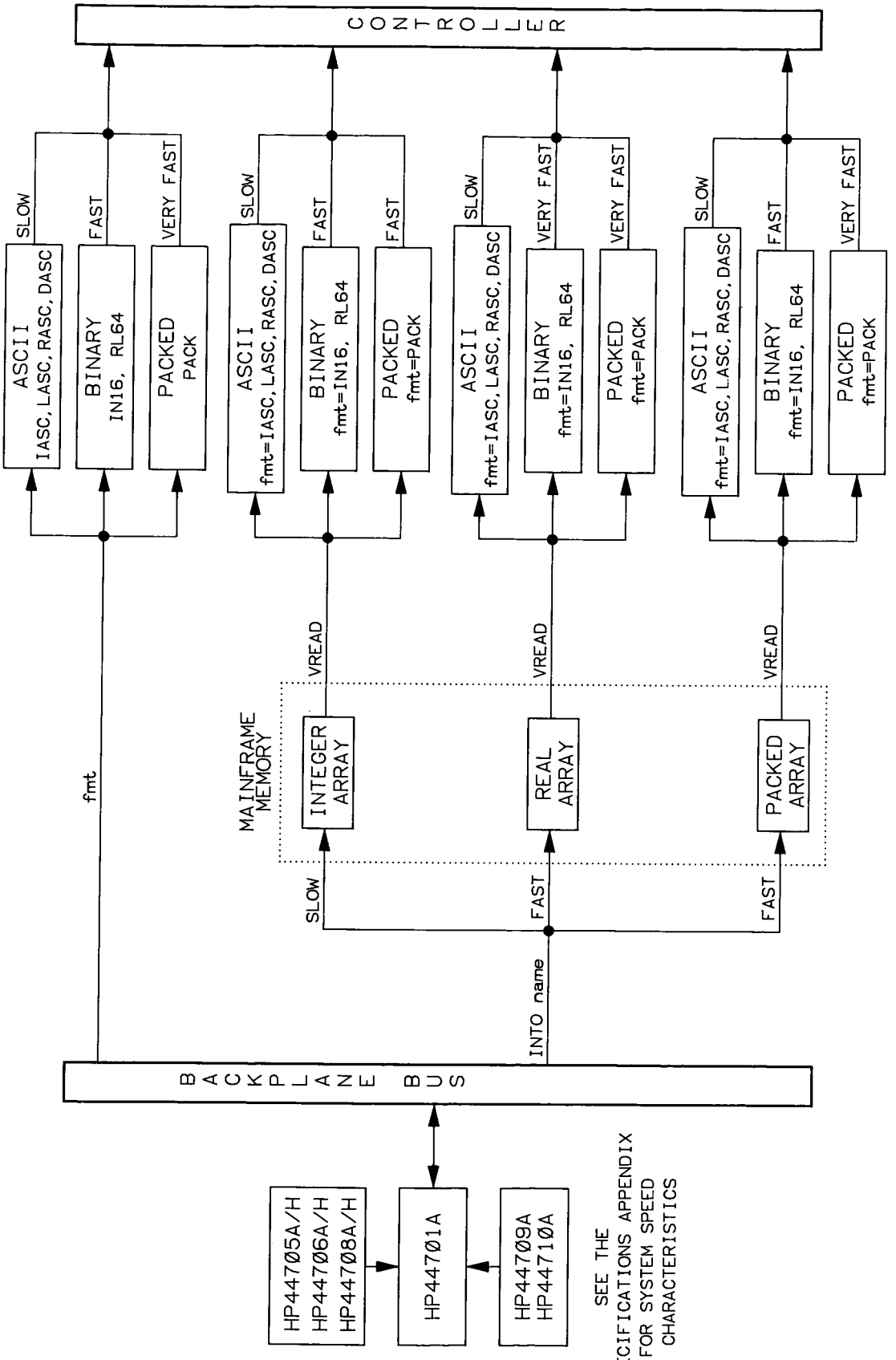


Figure 6-20. HP 3852A System Throughput (cont'd)

www.valuetronics.com

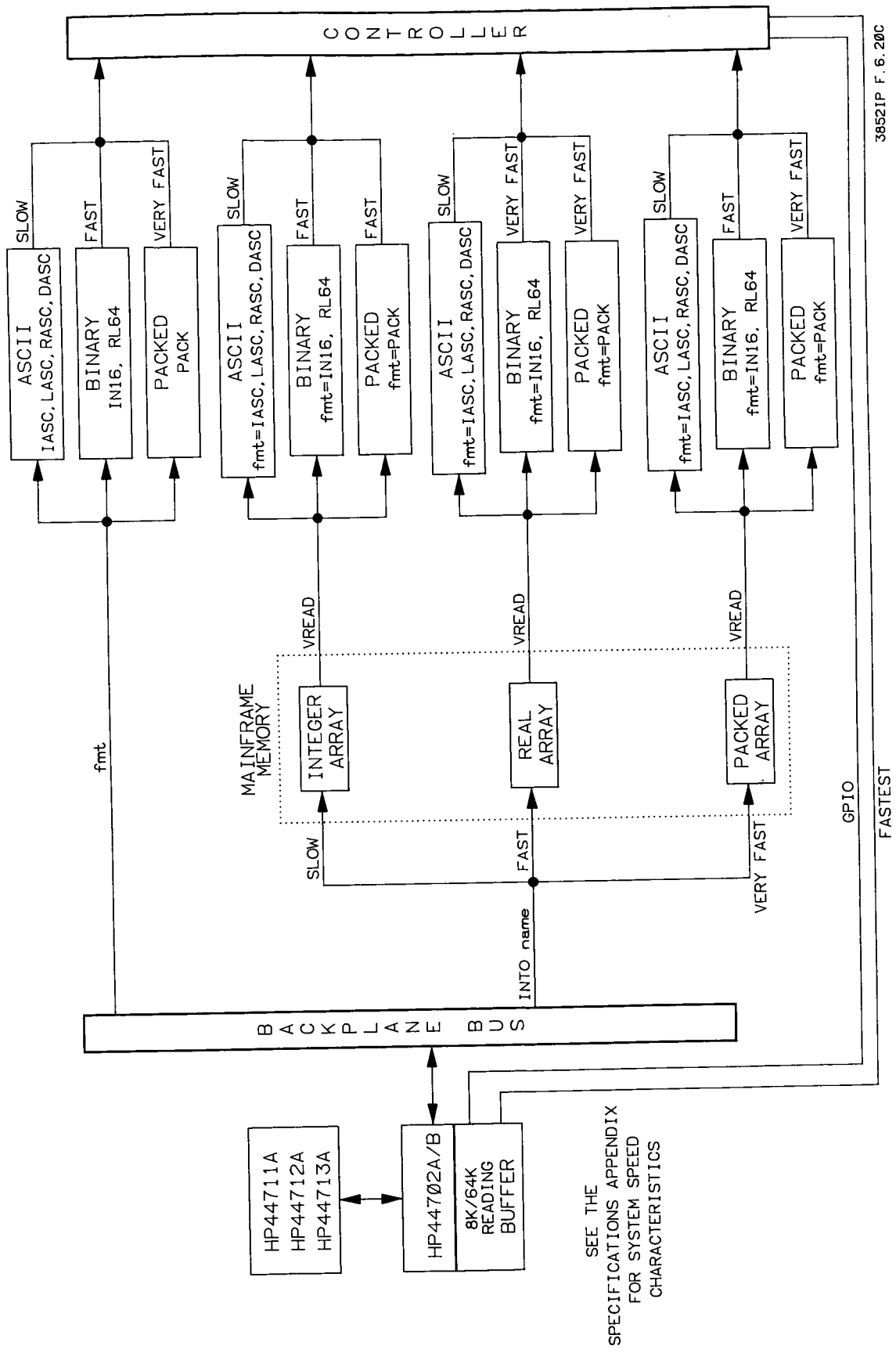


Figure 6-20. HP 3852A System Throughput (cont'd)

www.valuetronics.com

Format Conversions

When data is generated and returned by a command, the data is in its packed (original) format when it enters the mainframe's processor (Figure 6-20). Once the data is in the processor, it is converted to the format specified by *fmt* then passed to the output buffer, or it is converted to the storage type (e.g. REAL, INTEGER) and stored in the array specified by **INTO** *name*. The time in which this process occurs is a function of the packed reading length (number of bytes), the format/storage type specified, and whether or not the data is displayed.

Packed Reading Lengths

If the packed reading is greater than two bytes or the packed reading is a Real number (i.e. HP 44702A/B DCV and OHMS functions), the mainframe's processor will convert the data to the RL64 format before converting it to the format (*fmt*) or storage type specified. Thus, if the specified format or storage type for packed data greater than two bytes is RL64 (REAL), the data is only converted once before it is output or stored. If, for example, the IN16 (INTEGER) format/storage type is specified, the processor first converts the data to a RL64 (REAL) format, then it rounds the data and converts it a second time to the IN16 format. Therefore, if we sent the command:

```
40 OUTPUT 709;"CONFMEAS DCV, 0-5, USE 200, RL64"
```

data would be returned to the output buffer (or stored) at a faster rate than if we sent the command as:

```
40 OUTPUT 709;"CONFMEAS DCV, 0-5, USE 200, IN16"
```

due to the rounding and second conversion to the IN16 format. Again, for packed data greater than two bytes/reading, the reading is first converted to RL64, then to the format/storage type specified.

When the length of the packed data returned is two bytes as it would be for the digital input and digital output command:

```
30 OUTPUT 709;"READ 402"
```

specifying an IN16 (INTEGER) format:

```
30 OUTPUT 709;"READ 402 IN16"
```

passes the data to the output buffer at a faster rate than if the RL64 (REAL) format were specified. This is because the mainframe will convert packed data two bytes in length (except Real numbers) to an IN16 (INTEGER) format before converting it to the format or storage type specified. Therefore, specifying IN16 as the format results in one conversion only. All other formats require two conversions.

www.valuetronics.com

Temperature and Strain Measurements, and Data Processing

Except for readings that will be converted by the COMPEN data processing command, the packed format of temperature and strain measurements made by the HP 3852A is RL64. This is because the mainframe must convert the reading to a Real number in order to express the measurement as a temperature or measure of strain. When the CONV, SCALE, and STAT data processing functions are used, the computations involved require the data to be in a Real number format, thus any "packed" results are expressed in RL64 format.

When declaring a PACKED array for temperature or strain measurements, note that each reading is 8 bytes long due to the conversion to RL64. Also because of the conversion, strain measurements and temperature measurements using thermistors, thermocouples, and RTDs can be stored in the same PACKED array.

Table 6-6 lists the bytes/reading for packed data returned to the mainframe. The table also identifies the format (*fmt*) and storage type (REAL, INTEGER) that when specified, results in only one data conversion, enabling data to be passed to the HP-IB output buffer or stored at a much faster rate.

Data Format Considerations

The speed at which data is removed from the buffer and stored in the controller depends mainly on the number of data bytes involved. Recall during the discussion of data conversions that it is faster for the mainframe processor to convert packed readings greater than two bytes to a RL64 (REAL) format. However, a reading in this format is eight bytes long. As a result, it takes longer for a controller to retrieve data from the buffer in a REAL format than it would to retrieve data in the IN16 (INTEGER) format (two bytes).

The format (IN16, RL64) which maximizes system (HP 3852A/controller) throughput depends on how fast the HP 3852A passes data to its output buffer, versus how fast the controller can retrieve the data. If the controller is able to retrieve data at a rate such that it has to wait for the mainframe to pass the data to the buffer, you should specify RL64 as the output format. The reason is the time required for the controller to retrieve the number of bytes involved enables the mainframe to keep pace with the controller. If, on the other hand, the mainframe is passing data to the buffer much faster than it can be retrieved by the controller, the output format specified should be IN16. Since only two bytes are involved, the controller may then be able to keep pace with the mainframe.

Packed Data Conversions

When packed data is returned to the HP-IB output buffer from a packed array or directly from a plug-in accessory, the controller must unpack the data in order to accurately store and display the readings. Table 6-7 lists conversion formulas and unpacking routines for each plug-in accessory that can be implemented by an HP Series 200 or Series 300 computer. The table also shows the bit pattern of the various packed formats. Following the table are examples showing how several of the routines can be implemented.

www.valuetronics.com

Table 6-6. Packed Data Bytes/Reading

Accessory	Command	Function	Bytes/ Reading	Recommended fmt/Storage Type
HP 44701A	CHREAD, CONFMEAS, MEAS, XRDGS	ACV, DCV, OHM, OHMF	4	RL64 REAL
		TEMP, REFT, THM, THMF, RTD, RTDF, STRVEX, STRUN, STRQ, STRHB, STRHP, STRFB, STRFBP, STRFP, STRQTEN, STRQCOMP	8	RL64 REAL
HP 44702A/B	CHREAD, CONFMEAS, MEAS, XRDGS	DCV, OHM, OHM10K, OHM100K, OHM1M, OHMF, OHMF10K, OHMF100K, OHMF1M	2*	RL64 REAL
		TEMP, REFT, THM, THMF, RTD, RTDF, STRVEX, STRUN, STRQ, STRHB, STRHP, STRFB, STRFBP, STRFP, STRQTEN, STRQCOMP	8	RL64 REAL
HP 44715A	CHREAD, CHREADZ, XRDGS	TOTALM, UDCM, CDM	2	IN16 INTEGER
		TOTAL, UDC, CM	4	RL64 REAL
	CHREAD, XRDGS	FREQ	4	RL64 REAL
	CHREAD, CHREADZ, XRDGS	PERD	4	RL64 REAL
		RAT	4	RL64 REAL
		PER	6	RL64 REAL
HP 44705A,05H, HP 44706A, HP 44708A,08H, HP 44709A, HP 44710A, HP 44711A, HP 44712A, HP 44713A	CLOSE?	—	2	IN16 INTEGER
HP 44721A HP 44722A	READ	—	2	IN16 INTEGER
	CHREAD, CHREADZ XRDGS	CHANNELS 0-15 or 0-7 specified	4	RL64 REAL
		CHANNELS 16-31 or 8-15 specified	2	IN16 INTEGER
HP 44724A, HP 44725A, HP 44728A, HP 44729A	CLOSE?	—	2	IN16 INTEGER
	READ	—	2	IN16 INTEGER
HP 3852A Mainframe	ADDR?, ERR?, EXTEND?, INTR?, RQS?, STA?, STATE?, STB?, USE	—	2	IN16 INTEGER
	ALRM, CONV, POWEROFF, SCALE, STAT, TIME, TIMEDATE	—	8	RL64 REAL
	SIZE?, INDEX?	—	4	RL64 REAL

*Reading is a Real number with a fractional part.

www.valuetronics.com

www.valuetronics.com

Table 6-7. Packed Data Conversion Routines

HP 44701A

Command: CHREAD, CONFMEAS, MEAS, XRDGS

Function: ACV, DCV, OHM, OHMF

Bit Pattern

Bytes 0-2	Byte 3

Bytes 0-2: Mantissa (2's complement integer)

Byte 3: Exponent, Base 10. If 80_H (128_{10}) overload, disregard Mantissa.

Conversion Formula and Routine

Formula:

reading = Mantissa x $10^{(exponent\ -6)}$

Routine:

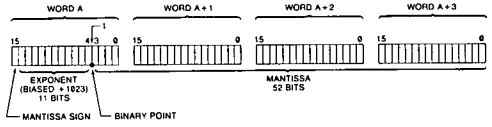
```
200  INTEGER Pack(0:1)
  • VOLTMETER COMMANDS WHICH SET UP
  • AND EXECUTE MEASUREMENT
  •
270  ENTER 709 USING "#,W";Pack(*)
280  PRINT FNHp44701(Pack(0),Pack(1))
290  END
300  DEF FNHp44701(INTEGER Pack0,Pack1)
310     INTEGER Expon
320     Expon=BINAND(Pack1,255)
330     IF Expon=128 THEN
340       RETURN 1.E+38
350     ELSE
360       RETURN(Pack0*256.+SHIFT(Pack1,8))*10^(Expon-256*
          (Expon>127)-6)
370     END IF
380  FNEND
```

HP44701A

Command: CHREAD, CONFMEAS, MEAS, XRDGS

Function: TEMP, REFT, THM, THMF, RTD, RTDF, STRQ, STRHB, STRFB, STRHP, STRFBP, STRFP

Bit Pattern



Conversion Formula and Routine

For Series 200/300, can be read directly into a Real Number

HP 44702A/B

Command: CHREAD, CONFMEAS, MEAS, XRDGS

Function: DCV, OHM, OHM10K, OHM100K, OHM1M, OHMF, OHMF10K, OHMF100K,OHMF1M

Bit Pattern

15	14-13	12	11-0

Bit 15: If 0, overload. Disregard Mantissa. If 1 and Mantissa = $OFFF_H$ (4095_{10}), overrange.

Bits 14-13: Range (see conversion formula)

Bit 12: Sign Bit. 0 = +, 1 = −

Bits 11-0: Mantissa—Integer number of counts.

Conversion Formula and Routine

Formula:

reading = Mantissa•(2.5mV/f(range))•1/I

Function	I	Bits 14/13	f(range)
DCV	1	00	256
OHM10K, OHMF10K	.001	01	32
OHM, OHMF, OHM100K, OHMF100K	.0001	10	4
OHM1M, OHMF1M	.00001	11	1

Routine (DCV):

```
200  INTEGER Pack
  • VOLTMETER COMMANDS WHICH SET
  • UP AND EXECUTE MEASUREMENT
  •
270  ENTER 709 USING "#,W";Pack
280  PRINT FNHp44702(Pack)
290  END
300  DEF FNHp44702(INTEGER Pack)
310     REAL R(0:3)
```

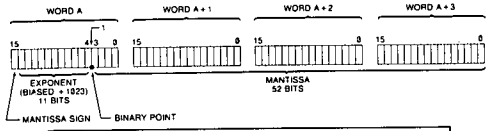
www.valuetronics.com

Table 6-7. Packed Data Conversion Routines (cont)

Accessory	Command	Function	Bit Pattern	Conversion Formula and Routine
				320 DATA 256.,32.,4.,1. 330 READ R(*) 340 M=BINAND(Pack,4095) 350 IF Pack>0 OR M=4095 THEN 360 RETURN 1.E+38 370 ELSE 380 V=M*.0025/R(BINAND(SHIFT(Pack,13),3)) 390 IF BIT(Pack,12) THEN V=-V 400 RETURN V 410 END IF 420 FNEND Routine (OHM10K,OHMF10K): Change Line 380 in DCV Routine to: 380 V=M*.0025/R(BINAND(SHIFT(Pack,13),3))/.001 Routine (OHM,OHMF,OHM100K,OHMF100K): Change Line 380 in DCV Routine to: 380 V=M*.0025/R(BINAND(SHIFT(Pack,13),3))/.0001 Routine (OHM1M,OHMF1M): Change Line 380 in DCV Routine to: 380 V=M*.0025/R(BINAND(SHIFT(Pack,13),3))/.00001

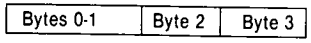
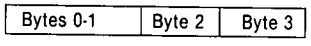
www.valuetronics.com

Table 6-7. Packed Data Conversion Routines (cont)

Accessory	Command	Function	Bit Pattern	Conversion Formula and Routine
HP 44702A/B	CHREAD, CONFMEAS MEAS, XRDGS	TEMP, REFT, THM, THMF, RTD, RTDF, STRQ, STRHB, STRHP, STRFBP, STRFP		For Series 200/300, can be read directly into a Real number.
HP 44715A	CHREAD, CHREADZ, XRDGS	TOTALM, UDCM, CDM	16-bit 2's Complement Integer	Routine (unsigned): 200 INTEGER Pack • • COUNTER COMMANDS WHICH SET UP AND • EXECUTE FUNCTION 270 ENTER 709 USING "#,W";Pack 280 PRINT FNIn16_unsigned(Pack) 290 END 300 DEF FNIn16_unsigned(INTEGER Pack) 310 RETURN Pack+65536.*(Pack<0) 320 FNEND NO CONVERSION IF SIGNED
HP 44715A	CHREAD, CHREADZ, XRDGS	TOTAL, UDC, CD	32-bit 2's Complement Integer	Routine (unsigned): 200 INTEGER Pack(0:1) • • COUNTER COMMANDS WHICH SET UP AND • EXECUTE FUNCTION 270 ENTER 709 USING "#,W";Pack(*) 280 PRINT FNIn32_unsigned(Pack(0),Pack(1)) 290 END 300 DEF FNIn32_unsigned(INTEGER Pack0,Pack1) 310 RETURN Pack0*65536.+4294967296. *(Pack0<0)+Pack1+65536.*(Pack1<0) 320 FNEND Signed: 200 INTEGER Pack(0:1) • • COUNTER COMMANDS WHICH SET UP AND • EXECUTE FUNCTION 270 ENTER 709 USING "#,W";Pack(*) 280 PRINT FNIn32_signed(Pack(0),Pack(1)) 290 END 300 DEF FNIn32_signed(INTEGER Pack0,Pack1) 310 RETURN Pack0*65536.+Pack1+65536.*(Pack1<0) 320 FNEND

www.valuetronics.com

Table 6-7. Packed Data Conversion Routines (cont)

Accessory	Command	Function	Bit Pattern	Conversion Formula and Routine
HP 44715A	CHREAD, XRDGS	FREQ	 Bytes 0-1: # of Periods counted Byte 2: Disregard Byte 3: Bits 7-4, Time Base. See Conversion Routine. Bits 3-0, Disregard.	Formula: reading = # of periods counted/Time Base Bits 7-4 / Time Base: 1101 — 10 msec 1110 — 100 msec 1111 — 1 sec Routine: 200 INTEGER Pack(0:1) • • COUNTER COMMANDS WHICH SET UP • AND EXECUTE FUNCTION 270 ENTER 709 USING "#,W";Pack(*) 280 PRINT FnFreq(Pack(0),Pack(1)) 290 END 300 DEF FnFreq(INTEGER Pack0,Pack1) 310 RANGE = BINAND(SHIFT(Pack1, – 4),15) 320 RETURN(Pack0 + 65536.*(Pack0<0))*10^(15 – RANGE) 330 FNEND
HP 44715A	CHREAD, CHREADZ, XRDGS	PERD	 Bytes 0-1: # of clocks counted Byte 2: Disregard Byte 3: Bits 7-4, Time Base Bits 3-0, Disregard.	Formula: reading = # of clocks counted*Time Base Bits 7-4 / Time Base: 1011 — 1 μsec 1100 — 10 μsec 1101 — 100 μsec 1110 — 1 msec 1111 — 10 msec Routine: 200 INTEGER Pack (0:1) • COUNTER COMMANDS WHICH SET UP • AND EXECUTE FUNCTION • 270 ENTER 709 USING "#,W";Pack(*) 280 PRINT FNPerd(Pack(0),Pack(1)) 290 END 300 DEF FNPerd(INTEGER Pack0,Pack1) 310 Range = BINAND(SHIFT(Pack1, – 4),15) 320 RETURN(Pack0 + 65536.*(Pack0<0))*10^(RANGE – 17) 330 FNEND

www.valuetronics.com

Table 6-7. Packed Data Conversion Routines (cont)

Accessory	Command	Function	Bit Pattern	Conversion Formula and Routine
HP 44715A	CHREAD, CHREADZ, XRDGS	RAT	Bytes 0-1 | Bytes 2-3 Bytes 0-1: B input counts Bytes 2-3: A input counts	Formula: reading = A input counts/B input counts Routine: 200 INTEGER Pack(0:1) • COUNTER COMMANDS WHICH SET UP • AND EXECUTE FUNCTION • 270 ENTER 709 USING "#,W";Pack(*) 280 PRINT FnRatio(Pack(0),Pack(1)) 290 END 300 DEF FnRatio(INTEGER Pack0,Pack1) 310 RETURN(Pack1 + 65536. * (Pack1 < 0))/(Pack0 + 65536. * (Pack0. < 0)) 320 FNEND
HP 44715A	CHREAD, CHREADZ, XRDGS	PER	Bytes 0-1 | Bytes 2-3 | Byte 4 | Byte 5 Bytes 0-1: # of periods averaged Bytes 2-3: # of clocks counted Byte 4: Disregard Byte 5: Bits 7-4, Time Base. Bits 3-0, Disregard	Formula: reading = (# of clocks counted / # of periods averaged) * Time Base Bits 7-4 — Time Base: 1011 — 1 μsec 1100 — 10 μsec 1101 — 100 μsec 1110 — 1 msec 1111 — 10 msec Routine: 200 INTEGER Pack(0:2) • • COUNTER COMMANDS WHICH SET UP • AND EXECUTE FUNCTION 270 ENTER 709 USING "#,W";Pack(*) 280 PRINT FNPer(Pack(0),Pack(1),Pack(2)) 290 END 300 DEF FNPer(INTEGER Pack0,Pack1,Pack2) 310 PERIOD = (Pack0 + 65536. * (Pack0 < 0))/ (Pack1 + 65536. * (Pack1 < 0)) 320 Range = BINAND(SHIFT(Pack2, − 4),15) 330 RETURN Period * 10 (RANGE − 17) 340 FNEND

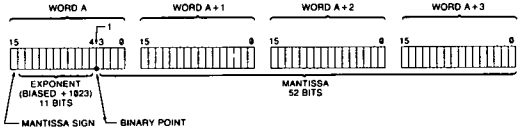
www.valuetronics.com

Table 6-7. Packed Data Conversion Routines (cont)

Accessory	Command	Function	Bit Pattern	Conversion Formula and Routine
HP 44705A/H, HP 44708A/H, HP 44709A, HP 44710A, HP 44711A, HP 44712A, HP 44713A	CLOSE?	—	16-bit 2's Complement Integer If output equals decimal: 0-channel open 1-channel closed 2-channel closed connected to sense bus 3-channel closed connected to source bus 4-channel closed connected to both busses	—
HP 44706A	CLOSE?	—	If output equals decimal: 0-channel open 1-channel closed - sense bus tree switch (channel 91 only) 2-closed - connected to sense bus 4-closed - connected to both busses	
HP 44721A, HP 44722A	READ	—	16-bit 2's Complement Integer Byte 0 is always 0_H for the HP 44722A	Routine (unsigned): 200 INTEGER Pack • • DIGITAL INPUT COMMANDS WHICH SET • UP AND EXECUTE FUNCTION 270 ENTER 709 USING "#,W";Pack 280 PRINT FNIn16_unsigned(Pack) 290 END 300 DEF FNIn16_unsigned(INTEGER Pack) 310 RETURN Pack+65536.*(Pack<0) 320 FNEND
HP 44721A, HP 44722A	CHREAD, CHREADZ, XRDGS	Channel 0-15 (0-7) specified	32-bit 2's Complement Integer	Routine: 200 INTEGER Pack(0:1) • • DIGITAL INPUT COMMANDS WHICH SET • UP AND EXECUTE FUNCTION 270 ENTER 709 USING "#,W";Pack(*) 280 PRINT FNIn32_unsigned(Pack(0),Pack(1)) 290 END 300 DEF FNIn32_unsigned(INTEGER Pack0,Pack1) 310 RETURN Pack0*65536.+4294967296.*(Pack0<0)+Pack1 +65536.*(Pack1<0) 320 FNEND
HP 44721A, HP 44722A	CHREAD, CHREADZ, XRDGS	Channel 16-31 (8-15) specified	Byte 0 | Byte 1 Byte 0: 0_H Byte 1: 1 or 0 on Bit 0 indicates input level on the specified channel	—

www.valuetronics.com

Table 6-7. Packed Data Conversion Routines (cont)

Accessory	Command	Function	Bit Pattern	Conversion Formula and Routine
HP 44724A, HP 44725A, HP 44728A, HP 44729A	CLOSE?	—	Byte 0 | Byte 1 Byte 0: 0_H Byte 1: 1 or 0 on Bit 0 indicates input level on the specified channel	—
HP 44724A, HP 44725A, HP 44728A, HP 44729A	READ	—	16-bit 2's Complement Integer Byte 0 is always 0_H for the HP 44728A and HP 44729A	
HP 3852A Mainframe	ADDR?, ERR?, EXTEND?, INTR?, RQS?, STA?, STATE?, STB?, USE?	—	16-bit 2's Complement Integer	—
HP 3852A Mainframe	EXTEND?	—	Returns seven readings in the above format	
HP 3852A Mainframe	STATE?	—	Returns two readings in the above format	
HP 3852A Mainframe	ALRM, CONV, POWEROFF, SCALE, STAT, TIME, TIMEDATE	—		—
HP 3852A Mainframe	SIZE?, INDEX?	—	32-Bit 2's Complement Integer	200 INTEGER Pack(0:1) • • MAINFRAME SIZE? OR INDEX? COMMAND • 270 ENTER 709 USING "#,W";Pack(*) 280 PRINT FNIn32_unsigned(Pack(0),Pack(1)) 290 END 300 DEF FNIn32_unsigned(INTEGER Pack0,Pack1) 310 RETURN Pack0*65536.+4294967296.* (Pack0<0)+Pack1+65536.*(Pack1<0) 320 FNEND
HP 3852A Mainframe	VREAD	—	Determined by the Data Read	—

Example 1. Converting HP 44701A DC Voltage Measurements

This example shows how the HP 44701A conversion routine is used to unpack three DC voltage measurements. The readings are taken using the CONFMEAS command and output directly to the HP-IB buffer by specifying PACK for the *fmt* parameter.

```
10   INTEGER Pack(0:5)
20   OUTPUT 709;"CONFMEAS DCV, 0-2, USE 700, PACK"
30   ENTER 709 USING "#,W";Pack(*)
40   FOR I=0 TO 4 STEP 2
50     PRINT FNHp44701(Pack(I),Pack(I+1))
60   NEXT I
70   END
80   DEF FNHp44701(INTEGER Pack0,Pack1)
90     INTEGER Expon
100    Expon=BINAND(Pack1,255)
110    IF Expon=128 THEN
120      RETURN 1.E+38
130    ELSE
140      RETURN (Pack0*256.+SHIFT(Pack1,8))*10^(Expon-256*(Expon>127)-6)
150    END IF
160  FNEND
```

Example 2. Converting HP 44702A/B DC Voltage Measurements

This example shows how the HP 44702A/B conversion routine is used to unpack three DC voltage measurements. In this example, the readings are taken using the CONFMEAS command and output directly to the HP-IB buffer by specifying PACK for the *fmt* parameter.

```
10   INTEGER Pack(0:2)
20   OUTPUT 709;"CONFMEAS DCV, 400-402, USE 500, PACK"
30   ENTER 709 USING "#,W";Pack(*)
40   FOR I=0 TO 2
50     PRINT FNHp44702(Pack(I),1)
60   NEXT I
70   END
80   DEF FNHp44702(INTEGER Pack,REAL I)
90     REAL R(0:3)
100    DATA 256.,32.,4.,1.
110    READ R(*)
120    M=BINAND(Pack,4095)
130    IF Pack>0 OR M=4095 THEN
140      RETURN 1.E+38
150    ELSE
160      V=M*.0025/R(BINAND(SHIFT(Pack,13),3))/I
170      IF BIT(Pack,12) THEN V=-V
180      RETURN V
190    END IF
200  FNEND
```


www.valuetronics.com

Example 3. Converting HP 44715A TOTALM Counts

This example illustrates the conversion routine for the HP 44715A command CHREAD and the function TOTALM. The routine is necessary to express packed counts greater than 32767 as a positive number. Note that this routine can also be used for the IN16 format.

```
10   INTEGER Counts
20   OUTPUT 709;"USE 300"
30   OUTPUT 709;"CONF TOTALM"
40   OUTPUT 709;"TERM NON"
50   OUTPUT 709;"NPER 40000"
60   OUTPUT 709;"TRIG AUTO"
70   PAUSE
80   OUTPUT 709;"CHREAD 300, PACK"
90   ENTER 709 USING "#,W";Counts
100  PRINT FNIn16_unsigned(Counts)
110  END
120  DEF FNIn16_unsigned(INTEGER Counts)
130      RETURN Counts+65536.*(Counts<0)
140  FNEND
```

Example 4. Converting HP 44715A TOTAL Counts (Unsigned)

This example illustrates the conversion routine for the HP 44715A command CHREAD and the function TOTAL. The routine is necessary to express packed counts greater than 214748348 as a positive number.

```
10   INTEGER Cntotal(0:1)
20   OUTPUT 709;"USE 300"
30   OUTPUT 709;"CONF TOTAL"
40   OUTPUT 709;"TERM NON"
50   OUTPUT 709;"TRIG AUTO"
60   PAUSE
70   OUTPUT 709;"CHREAD 300, PACK"
80   ENTER 709 USING "#,W";Cntotal(*)
90   PRINT FNIn32_unsigned(Cntotal(0),Cntotal(1))
100  END
110  DEF FNIn32_unsigned(INTEGER Cntotal0,Cntotal1)
120      RETURN Cntotal0*65536.+4294967296.*(Cntotal0<0)+Cntotal1+65536.*
(Cntotal1<0)
130  FNEND
```


www.valuetronics.com

Example 5. Converting HP 44715A Up/Down Counts (Signed)

This example illustrates the conversion routine for the HP 44715A command CHREAD and function UDC. The routine is used to express up and down counts greater than 65535 as positive and negative numbers.

```
10   INTEGER Up_dncnts(0:1)
20   OUTPUT 709;"USE 300"
30   OUTPUT 709;"TRIG HOLD"
40   OUTPUT 709;"TERM NON,NON"
50   OUTPUT 709;"EDGE LH,LH"
60   OUTPUT 709;"FUNC UDC"
70   OUTPUT 709;"TRIG SGL"
80   PAUSE
90   OUTPUT 709;"CHREAD 300, PACK"
100  ENTER 709 USING "#,W";Up_dncnts(*)
110  PRINT FNIn32_signed(Up_dncnts(0),Up_dncnts(1))
120  END
130  DEF FNIn32_signed(INTEGER Up_dncnts0,Up_dncnts1)
140      RETURN Up_dncnts0*65536.+Up_dncnts1+65536.*(Up_dncnts1<0)
150  FNEND
```

Example 6. Converting HP 44721A Edge Transitions

This example illustrates the conversion routine used to unpack and express a number of edge transitions greater than 214748348 as a positive number.

```
10   INTEGER Edge_trans(0:1)
20   OUTPUT 709;"USE 400"
30   OUTPUT 709;"EDGE LH"
40   OUTPUT 709;"CHREAD 300, PACK"
50   ENTER 709 USING "#,W";Edge_trans(*)
60   PRINT FNIn32_unsigned(Edge_trans(0),Edge_trans(1))
70  END
80  DEF FNIn32_unsigned(INTEGER Edge_trans0,Edge_trans1)
90      RETURN Edge_trans0*65536.+4294967296.*(Edge_trans0<0)+Edge_trans1
+65536.*(Edge_trans1<0)
100  FNEND
```


www.valuetronics.com

Contents

www.valuetronics.com

Chapter 7

Triggering and Pacing

Introduction

Many of the measurement and control functions performed by the HP 3852A are initiated and regulated by triggering, scanning, and pacing signals. This chapter contains information on the mainframe's system (backplane) trigger, its backplane scanning function, its internal pacer, and the functions of the system clock. This information has been grouped into three sections: System Triggering and Scanning, Pacing, and System Timing. A general overview of each section is given below.

- **System Triggering and Scanning** explains how to use the mainframe's system trigger which is applied simultaneously to all mainframe and extender slots. This section also describes backplane scanning as it applies to measurements using a voltmeter accessory and an external voltmeter.

- **Pacing** describes how to configure and use the system pacer to regulate (or initiate) measurement and control functions within the HP 3852A.

- **System Timing** explains how to set and read the system clock and calendar. Also included is information on establishing benchmarks.

System Triggering and Scanning

This section explains how to generate the HP 3852A's system trigger signal and how to set up the accessory or accessory channel to receive the signal. This section also covers the backplane scanning sequence and how it can be modified for use with a voltmeter accessory or an external voltmeter.

The TRG Command

The system (backplane) trigger enables you to trigger multiple accessory channels simultaneously. This trigger signal appears at all mainframe slots. If the digital extender cabled is connected, the trigger is also applied to each HP 3853A extender slot.

www.valuetronics.com

The source of the system trigger is set with the HP 3852A's TRG command. This command has the syntax:

TRG [*source*]

The source parameters are EXT, SGL, GET, and HOLD.

```
OUTPUT 709;"TRG EXT"
```

When TRG EXT is set, a system trigger occurs when a negative-going TTL pulse is applied to the mainframe's SYSTEM TRIGGER IN BNC. Note that the signal applied to this BNC is the actual trigger signal that appears on the backplane. If an accessory does not trigger when the signal is applied, it may be necessary to increase the pulse width.

```
OUTPUT 709;"TRG SGL"
```

When TRG SGL is set, a system trigger is generated when the command executes. After the trigger is issued, the mainframe sets TRG HOLD. Note that SGL is the default source for the TRG command. This means if TRG is executed without specifying a source, a single trigger (SGL) is issued.

```
OUTPUT 709;"TRG GET"
```

With TRG GET, a system trigger is generated when the mainframe executes its GET (group execute trigger) command or when the controller executes an HP-IB group execute trigger (e.g. TRIGGER 709).

NOTE

The HP 3852A's GET command is not executed until all previously entered commands have executed. The HP-IB group execute trigger causes an immediate system trigger.

```
OUTPUT 709;"TRG HOLD"
```

TRG HOLD disables the system trigger. At power-on, TRG HOLD is set.

Configuring the Accessory

Unless an accessory channel is specifically programmed to receive a system trigger, the trigger is ignored. To set the system trigger as the channel's trigger source, the *source* parameter SYS is specified in the appropriate accessory triggering command as shown in the examples below.

```
30  OUTPUT 709;"TRIG SYS"
30  OUTPUT 709;"SCTRIG SYS"
30  OUTPUT 709;"STTRIG SYS"
```


www.valuetronics.com

Triggering Examples

The following example programs show how voltmeter accessories are configured to respond to the system trigger. Keep in mind that TRG sets the source of the system trigger and TRIG, SCTRIG, STTRIG, etc., sets the source of the accessory trigger.

```
10   OUTPUT 709;"USE 700"
20   OUTPUT 709;"CONF DCV"
30   OUTPUT 709;"TERM EXT"
40   OUTPUT 709;"TRIG SYS"
50   OUTPUT 709;"TRG SGL"
60   OUTPUT 709;"CHREAD 700"
70   ENTER 709;A
80   PRINT A
90   END
```

In this program, a voltmeter accessory is configured to make a DC voltage measurement on its rear panel terminals. Line 40 sets the system trigger as the voltmeter's trigger source. When line 50 executes, a system trigger occurs which triggers the voltmeter. Once the measurement is taken, it is entered into the controller and displayed.

```
10   OUTPUT 709;"USE 700"
20   OUTPUT 709;"CONF DCV"
30   OUTPUT 709;"TERM EXT"
40   OUTPUT 709;"TRIG SYS"
50   OUTPUT 709;"USE 200"
60   OUTPUT 709;"CONF DCV"
70   OUTPUT 709;"TERM EXT"
80   OUTPUT 709;"TRIG SYS"
90   OUTPUT 709;"TRG GET"
100  TRIGGER 709
110  OUTPUT 709;"CHREAD 700"
120  ENTER 709;A
130  OUTPUT 709;"CHREAD 200"
140  ENTER 709;B
150  PRINT A,B
160  END
```

In this program, the system trigger is used to trigger two voltmeter accessories simultaneously. Each voltmeter is separately configured to make a DC voltage measurement on its rear panel terminals when the system trigger is received. Note in this example that the system trigger source is an HP-IB group execute trigger. When the trigger is issued (line 100), the readings are taken.

Figure 7-1 summarizes the relationship of the system trigger sources to the accessory trigger source SYS.

www.valuetronics.com

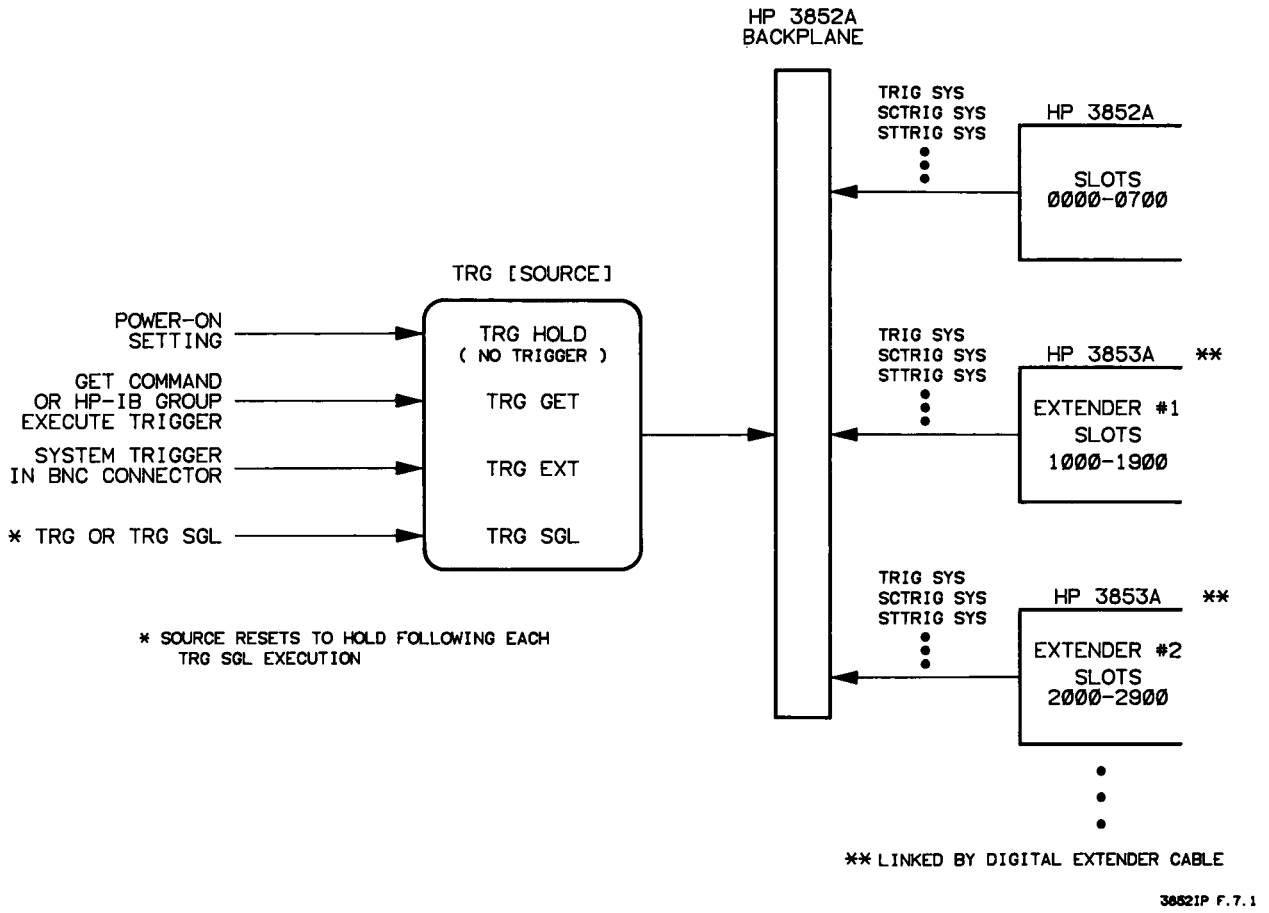


Figure 7-1. System and Accessory Trigger Source Relationship

www.valuetronics.com

Backplane Scanning

Backplane scanning is the process of closing a series of multiplexer accessory channels and using the mainframe's backplane analog bus to route the input signals to a voltmeter accessory or external device. The process of closing and opening each channel while advancing (scanning) through the channel list is the function of the mainframe's STRIG and SADV commands.

The STRIG Command

The STRIG command specifies the trigger source which directs the mainframe to close the first channel in the channel list to be connected to the analog bus. This trigger actually starts the scan through the channels but does not continue the operation. The advance through the channels is controlled by the SADV command. Note that a scan trigger is required for each pass through the channel list. The STRIG command has the syntax:

STRIG *source*

The *source* parameters are SCAN, CHADV, KEY, and PACER. Regardless of the trigger source and when the trigger occurs, the first channel is closed only after verification of the voltmeter parameters set for the function specified, and/or after verification of the channel list.

```
OUTPUT 709;"STRIG SCAN"
```

When SCAN is specified, the first channel is closed upon execution of the command. At power-on, STRIG SCAN is set.

```
OUTPUT 709;"STRIG CHADV"
```

When CHADV is specified, the first channel is closed when a negative-going TTL pulse is applied to the mainframe's CHANNEL ADVANCE BNC. Note that the pulse width of the signal must be greater than 0.5 μs.

```
OUTPUT 709;"STRIG KEY"
```

When KEY is specified, the first channel is closed when SADV KEY is pressed on the front panel.

```
OUTPUT 709;"STRIG PACER"
```

When PACER is specified, the channel is closed when a pacer pulse is received. Note that the pulse is sensed internally - no connection to the PACER OUT BNC is necessary.

www.valuetronics.com

The SADV Command

The SADV command specifies the trigger source which controls subsequent channel openings and closures thus advancing the scan through the channel list. The SADV command has the syntax:

SADV *source*

The *source* parameters are SCAN, CHADV, KEY, and PACER.

```
OUTPUT 709;"SADV SCAN"
```

When SCAN is specified, the scan is advanced (current channel opened, next channel closed) as soon as NRDGS (# of readings/channel) are available. If a measurement is not being taken (SCAN command), the scan is advanced as soon as the current channel is closed.

```
OUTPUT 709;"SADV CHADV"
```

When CHADV is specified, the scan is advanced when a negative-going TTL pulse is applied to the CHANNEL ADVANCE BNC.

```
OUTPUT 709;"SADV KEY"
```

When KEY is specified, the scan is advanced when SADV KEY is pressed on the front panel.

```
OUTPUT 709;"SADV PACER"
```

When PACER is specified, the scan is advanced on the pacer pulse as soon as NRDGS are available. If a measurement is not being taken, the scan is advanced by a pacer pulse as soon as the current channel is closed. Again, the pacer pulse is sensed internally and is not taken from the PACER OUT BNC.

Using STRIG and SADV

The following examples show how STRIG and SADV are used to control a backplane scan. In the first two examples, channels are scanned as measurements are made using the HP 3852A's voltmeter accessories. In the third example, STRIG and SADV are used with the SCAN command to make measurements with an external voltmeter.

Measurements Using CONFMEAS

The CONFMEAS command is a scanning command used by the voltmeter to make measurements via a backplane scan. The command CONFigures the voltmeter for the function specified then performs the MEASurement.

The CONFMEAS command has the syntax:

CONFMEAS *function ch_list* [**NSCAN** *number*] [**USE** *ch*] [**INTO** *name*] or [*fmt*]

www.valuetronics.com

In addition to setting voltmeter parameters, the CONF portion of the command also sets STRIG SCAN and SADV SCAN. The interaction between STRIG, SADV, and CONFMEAS can be described in the first example.

```
10    REAL Dcrdgs(0:4)
20    OUTPUT 709;"CONFMEAS DCV, 0-4, USE 700"
30    ENTER 709;Dcrdgs(*)
40    PRINT USING "K,/";Dcrdgs(*)
50    END
```

In this program, DC voltage measurements are taken on five multiplexer channels and then entered into the controller and displayed. When line 20 executes, the first channel (channel 0) is closed after the MEAS portion of the command verifies the parameters set by CONF are valid for the function specified, and valid channels are in the channel list (STRIG SCAN). After the measurement is taken (NRDGS = 1), the scan is advanced to channel 1 (SADV SCAN). The scan is advanced until the end of the channel list is reached.

Note that the NSCAN parameter has a default setting of 1. Had a number other than 1 been specified, the scan through the channel list would have been repeated that number of times. NSCAN is only available with mainframe firmware revision 2.2 or greater.

When CONFMEAS is executed, the STRIG and SADV settings cannot be changed since the MEAS portion of the command immediately follows the CONF portion. The following program shows how CONF and MEAS can be executed as separate commands in order to change the STRIG and SADV parameters set by CONF.

```
10    REAL Dcrdgs(0:14)
20    OUTPUT 709;"CONF DCV USE 700"
30    OUTPUT 709;"SADV KEY"
40    OUTPUT 709;"MEAS DCV 0-4 NSCAN 3 USE 700"
50    ENTER 709;Dcrdgs(*)
60    PRINT USING "K,/";Dcrdgs(*)
70    END
```

When line 20 executes, the voltmeter accessory in slot 7 is configured to measure DC voltage. Recall that CONF also sets STRIG SCAN and SADV SCAN. Line 30 changes SADV SCAN to SADV KEY. This means that the scan through the channel list is advanced each time SADV KEY is pressed. STRIG SCAN remains set, therefore, the first channel is closed after the MEAS command in line 40 verifies the voltmeter settings and the channel list. As line 40 executes, channel 0 is closed and a measurement is taken.

www.valuetronics.com

To advance to the next channel, SADV KEY must be pressed. Since NSCAN 3 is specified, three scans will be made through the channel list. Thus, SADV KEY must be pressed 15 times in order to complete the command.

External Voltmeter Measurements

A feature of the HP 3852A is its ability to scan a list of multiplexer channels and use an external voltmeter to perform the measurements. The HP 3852A command dedicated to this feature is the SCAN command.

The SCAN Command

The SCAN command directs the backplane scanning operation when an external voltmeter is used. SCAN specifies the backplane bus connections made by the multiplexer channels, the channel list to be scanned, and the number of times the list is scanned. The SCAN command has the syntax:

SCAN [*backplane_bus*] *ch_list* [**NSCAN** *number*]

The *backplane_bus* parameters are SENSE, COM, and SEP.

When SENSE is specified, the multiplexer channel connects to the sensing lines on the backplane bus. SENSE is used for voltage measurements only.

When COM is specified, the SENSE lines and SOURCE lines of the backplane bus are connected at the multiplexer channel. COM is used when performing 2-wire ohms measurements.

When SEP is specified, the SENSE lines are connected to the Bank A multiplexer channels and the SOURCE lines are connected to the Bank B multiplexer channels. When SEP is used, only Bank A channels are specified. SEP is used for 4-wire ohms measurements and is valid for the HP 44705A, HP 44709A, and HP 44711A multiplexer accessories only.

The *ch_list* parameter represents the channels to be scanned. The NSCAN *number* parameter specifies the number of scans to be made through the channel list. Note that NSCAN is only available with mainframe firmware revisions 2.2 and greater.

Example - 2-Wire Ohms Measurement

The following example uses the HP 3852A's STRIG, SADV, and SCAN commands to scan a list of multiplexer accessory channels and uses an HP 3456A Digital Voltmeter to make the resistance measurements. The HP 3456A is connected to the HP 3852A as shown in Figure 7-2. Note in the figure that connecting the cables from the analog extender port on the HP 3852A to the input terminals of the voltmeter actually extends the mainframe's backplane bus to the voltmeter. By connecting the voltmeters's VOLTMETER COMPLETE BNC to the HP 3852A's CHANNEL ADVANCE BNC, and the HP 3852A's CHANNEL CLOSED BNC to the voltmeter's EXTERNAL TRIGGER BNC, synchronization between the measurements and the backplane scan is maintained.

www.valuetronics.com

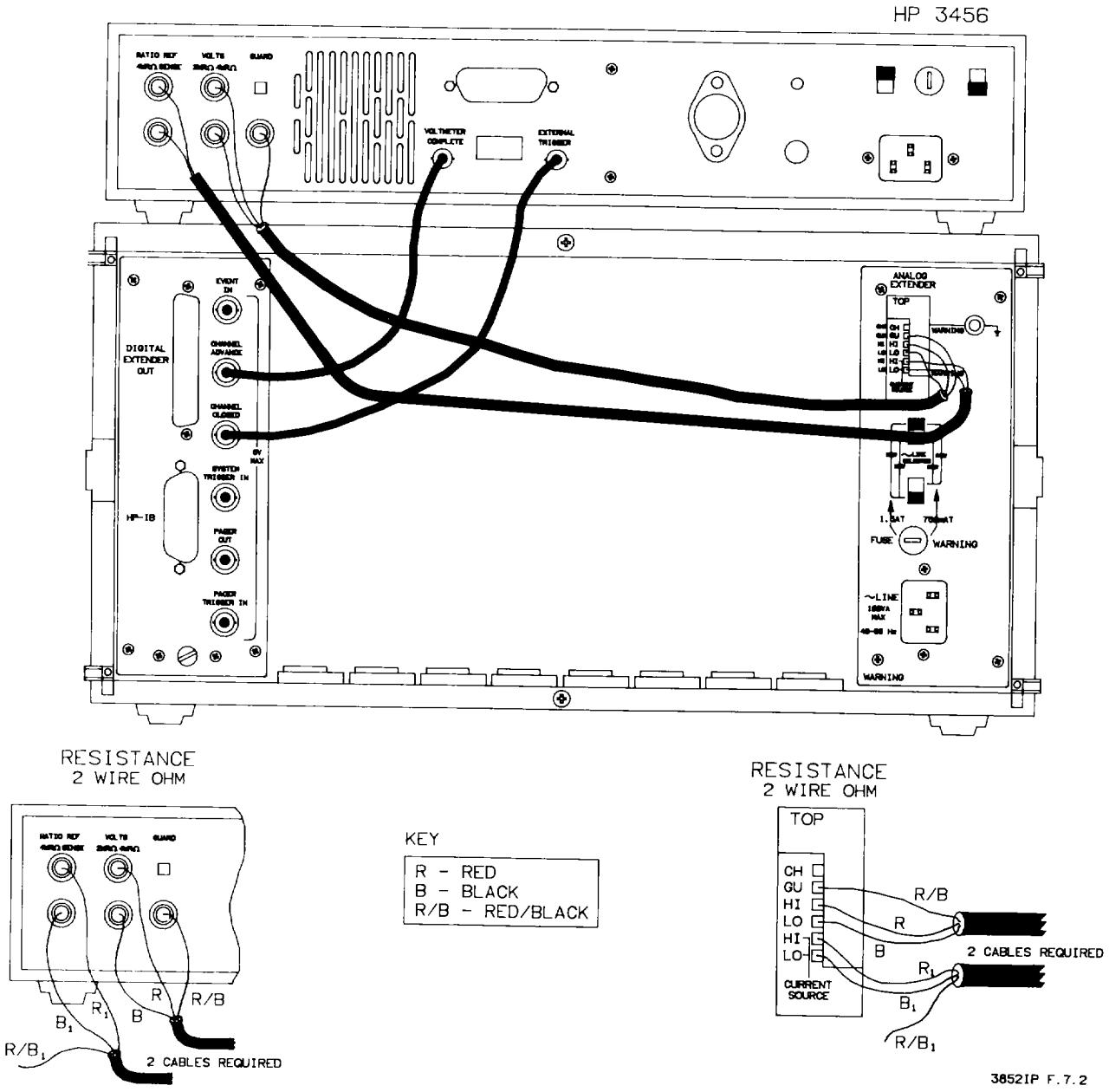


Figure 7-2. Measurements Using an External Voltmeter (2-Wire Ohms).

www.valuetronics.com

```
10   DIM V(0:11)
20   OUTPUT 709;"RST"
30   OUTPUT 709;"STRIG SCAN"
40   OUTPUT 709;"SADV CHADV"
50   OUTPUT 722;"HT4F4R4"
60   OUTPUT 722;"T2"
70   OUTPUT 709;"SCAN COM, 0-3, NSCAN 3"
80   FOR I=0 TO 11
90   ENTER 722;V(I)
100  PRINT V(I)
110  NEXT I
120  END
```

In line 30, STRIG SCAN is set so that channel 0 will be closed after the SCAN command (line 70) verifies the channel list. SADV CHADV (line 40) is set so that the pulse from the HP 3456A voltmeter when the measurement is complete will advance the scan. Lines 50 and 60 configure the voltmeter for 2-wire ohms measurements. The HP 3852A's SCAN command in line 70 specifies the backplane bus connections to be made, the channel list to be scanned, and the number of scans to be made through the list. Lines 80 through 110 enter and display the results on the controller.

Pacing

Contained in the HP 3852A mainframe is a system pacer. The pacer can be used to regulate the speed of a backplane scan, or it may be used to trigger functions within the HP 3852A or in external devices.

This section identifies the commands used to set up and activate the pacer and gives examples showing how the pacer is used.

Pacer Commands

When activated, the HP 3852A's pacer can output 1 to 65535, or a continuous stream of negative-going pulses. The pulse width can be set to either 0.5 μs or 5.0 μs. The pulses are capable of driving CMOS/HCMOS, TTL, and LSTTL loads.

The pulses are output from the PACER OUT BNC on the mainframe rear panel. For several mainframe functions which use the pacer to regulate or initiate an activity, the pacer pulse is sensed internally rather than taken from the BNC.

The commands which set up and activate the pacer are PACER, PDELAY, PTRIG, ENABLE PWIDE, and DISABLE PWIDE. The syntax and a description of each command is given below.

www.valuetronics.com

PACER *period* [*count*]

The PACER command sets the interval at which pacer pulses occur and the number of pulses generated once the pacer is triggered. The *period* parameter sets the interval. Pacer pulses can occur 1 μs apart or up to 4.19 seconds apart. Resolution is 250 ns. At power-on, the period is 1 μs. The *count* parameter specifies the number of pulses that are to occur when the pacer is triggered. As previously mentioned, the pacer can be programmed to output from 1 to 65535 pulses. If *count* is not specified, a continuous stream of pulses occur. If a count of 0 is specified, the programmed number or the continuous stream of pulses is halted.

PDELAY *trigger_delay*

The PDELAY command sets the time from when the pacer trigger is received to when the first pulse occurs. This time is set by the *trigger_delay* parameter. Pacer delays can range from 500 ns to 4.19 seconds. Resolution is 250 ns. At power-on, the pacer delay is 500 ns.

PTRIG [*source*]

The PTRIG command triggers (activates) the pacer. At power-on, the pacer is idle. A trigger is required each time to start the output of pulses. The *source* parameters are EXT, SGL, and HOLD.

```
OUTPUT 709;"PTRIG EXT"
```

When PTRIG EXT is set, the pacer is triggered by a negative-going pulse applied to the PACER TRIGGER IN BNC. The pulse width must be greater than 0.5 μs.

```
OUTPUT 709;"PTRIG SGL"
```

When PTRIG SGL is set, the pacer is triggered on execution of the command. SGL is the default source for the PTRIG command. If PTRIG is executed without specifying a source, a single trigger (SGL) is issued.

```
OUTPUT 709;"PTRIG HOLD"
```

PTRIG HOLD prevents the pacer from being triggered (power-on setting). Once the programmed number of pulses occur (PACER command), PTRIG HOLD is set. HOLD also disables the PACER TRIGGER IN BNC and stops any on-going pulse output.

www.valuetronics.com

ENABLE PWIDE

DISABLE PWIDE

ENABLE PWIDE changes the width of the pacer pulse from 0.5 μs to 5.0 μs. The pulse width can be changed during a fixed number or continuous stream of pulses. This command requires firmware revision 3.5 or greater and the 03852-66523 controller module.

DISABLE PWIDE changes the width of the pulse from 5.0 μs to 0.5 μs. At power-on or following a reset DISABLE PWIDE is set.

In summary, the most common sequence for setting up and activating the pacer is:

PACER

PDELAY (if a delay is desired)

PTRIG

Using the Pacer

The following programs show how the system pacer is used to regulate a backplane scan. In the first program, DC voltage measurements are made using the CONFMEAS command and an HP 44701A voltmeter accessory. Recall that the CONF portion of the command sets STRIG SCAN and SADV SCAN. This means that the scan is advanced from channel to channel at a rate approximately equal to the time it takes the voltmeter to make the measurement. Thus, given the voltmeter parameters set by the CONF portion of the command, a measurement is taken about every 20 ms (Figure 7-3).

```
10   DIM Dcrdgs (0:2)
20   OUTPUT 709;"CONFMEAS DCV 600-602 USE 700"
30   ENTER 709;Dcrdgs(*)
40   PRINT USING "K,/";Dcrdgs(*)
50   END
```

In the second program, DC voltage measurements are again made using an HP 44701A voltmeter. However, the CONF and MEAS commands are executed separately in order to configure and use the pacer to advance the scan. Here the DC voltage measurements occur approximately 1 second apart (Figure 7-3).

```
10   DIM Dcrdgs(0:2)
20   OUTPUT 709;"USE 700"
30   OUTPUT 709;"PACER 1"
40   OUTPUT 709;"CONF DCV"
50   OUTPUT 709;"STRIG PACER"
60   OUTPUT 709;"SADV PACER"
70   OUTPUT 709;"PTRIG SGL"
80   OUTPUT 709;"MEAS DCV 600-602"
90   ENTER 709;Dcrdgs(*)
100  PRINT USING "K,/";Dcrdgs(*)
110  END
```


www.valuetronics.com

Line 30 sets the pacer pulses to occur one second apart. Note that since a count was not specified, a continuous train of pulses will occur once the pacer is triggered. Since the CONF command (line 40) sets STRIG SCAN and SADV SCAN, lines 50 and 60 are required in order for the scan to be triggered and advanced by pacer pulses.

Although the readings taken in both programs are entered into the controller and displayed, the difference in scan rates between the two programs can be seen in the HP 3852A's display.

NOTE

Refer to the HP 3852A Command Reference Manual or the configuration and programming manual for your particular voltmeter accessory for information on all of the parameters set by the CONF command.

---

www.valuetronics.com

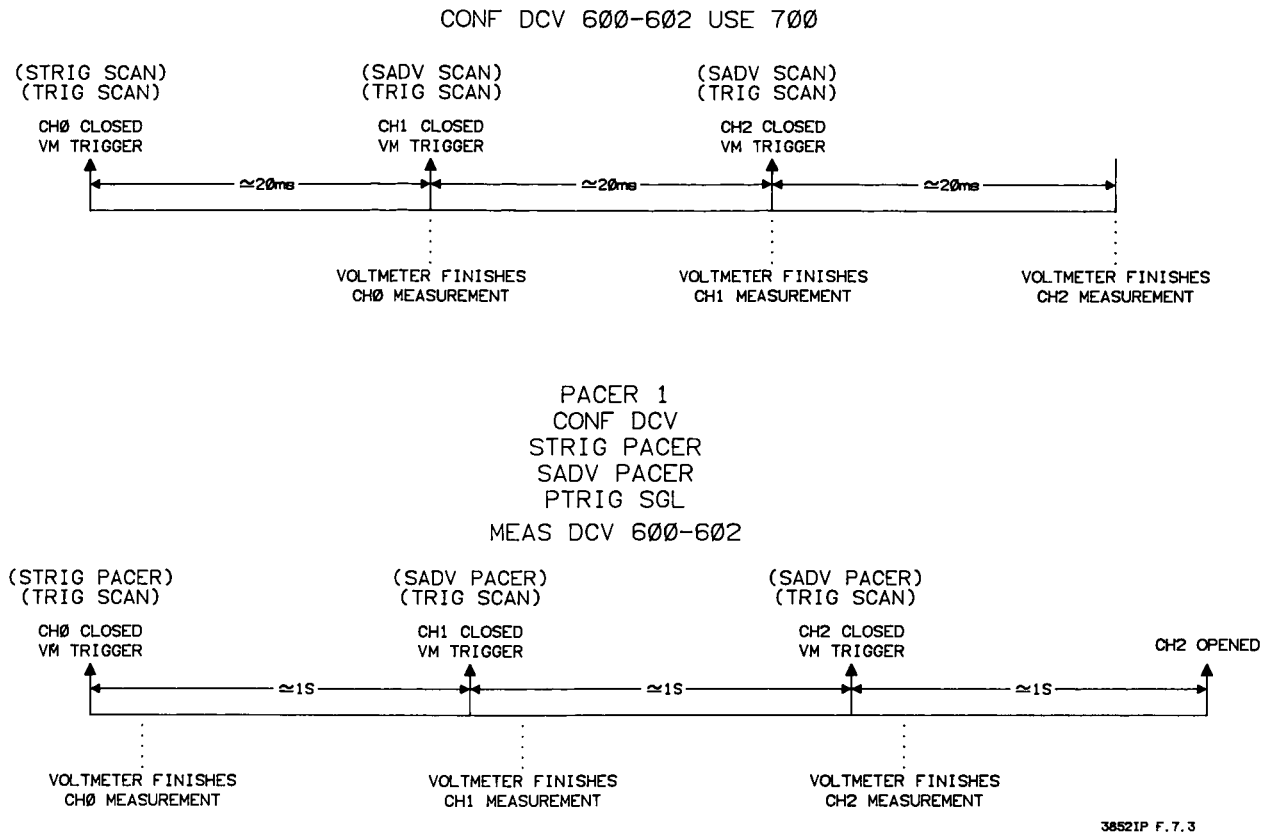


Figure 7-3. Regulating Backplane Scans with the System Pacer

www.valuetronics.com

Fixed Number of Pacer Pulses

In the PACER command of the previous program (line 30), the *count* parameter was not specified, therefore, a continuous train of pulses occurred once the pacer was triggered. Note that a continuous train of pulses is the easiest way to assure that there are enough pulses to complete the desired task. However, we could have used a fixed number of pulses in the previous program to accomplish the same results.

Regulating a Scan with a Fixed Number of Pulses

When using a fixed number of pulses to regulate a scan (SADV PACER) the count specified by the PACER command must be at least one greater than the number of channels in the channels list. Refer to the following program.

```
10   DIM Dcrdgs(0:8)
20   OUTPUT 709;"USE 700"
30   OUTPUT 709;"PACER 1, 10"
40   OUTPUT 709;"CONF DCV"
50   OUTPUT 709;"SADV PACER"
60   OUTPUT 709;"PTRIG SGL"
70   OUTPUT 709;"MEAS DCV 600-602 NSCAN 3"
80   ENTER 709;Dcrdgs(*)
90   PRINT USING "K,/";Dcrdgs(*)
100  END
```

This program scans the channel list three times (line 70) using pacer pulses to advance the scan (line 50). Three pulses are required for each pass. Since STRIG SCAN set by CONF closes channel 0, the first pulse is used to advance the scan from channel 0 to channel 1. The second pulse advances the scan from channel 1 to channel 2. The third pulse opens channel 2 thus completing the first scan. A total of nine pulses are required to make the three passes. The reason 10 pulses are specified, however, is due to the method in which the pacer is triggered. When PTRIG SGL is executed, the first of the 10 pulses occurs immediately and has no effect on system operation. It is the remaining nine pulses that are used.

Regulating a Scan with a Fixed Number of Pulses and a Pacer Delay

By using the PDELAY (pacer delay) command, you can specify a number of pacer pulses that is exactly equal to the number of channels in the channel list. The delay which occurs between the time the pacer is triggered and the first pulse, enables all the pulses to be used. By adding the PDELAY command to the previous program (see below), nine pulses can be specified by the PACER command.

www.valuetronics.com

```
10   DIM Dcrdgs(0:8)
20   OUTPUT 709;"USE 700"
30   OUTPUT 709;"PACER 1, 9"
40   OUTPUT 709;"PDELAY .05"
50   OUTPUT 709;"CONF DCV"
60   OUTPUT 709;"SADV PACER"
70   OUTPUT 709;"PTRIG SGL"
80   OUTPUT 709;"MEAS DCV 600-602 NSCAN 3"
90   ENTER 709;Dcrdgs(*)
100  PRINT USING "K,/";Dcrdgs(*)
110  END
```

There is no direct method for determining the amount of delay to specify. An exact delay is dependent on many factors including the voltmeter used and the length of the channel list. When the number of pulses specified is equal to the number of channels to be scanned, the delay must enable all pulses to be used so that the scan sequence completes (mainframe not busy). Generally, trial and error works best.

Initiating and Advancing the Scan with a Fixed Number of Pulses

The following program uses pacer pulses to initiate and advance the scan.

```
10   DIM Dcrdgs(0:8)
20   OUTPUT 709;"USE 700"
30   OUTPUT 709;"PACER 1, 13"
40   OUTPUT 709;"CONF DCV"
50   OUTPUT 709;"STRIG PACER"
60   OUTPUT 709;"SADV PACER"
70   OUTPUT 709;"PTRIG SGL"
80   OUTPUT 709;"MEAS DCV 600-602 NSCAN 3"
90   ENTER 709;Dcrdgs(*)
100  PRINT USING "K,/";Dcrdgs(*)
110  END
```

This program also scans the channel list three times (line 80). Since a delay is not used, 13 pulses (line 30) are specified. Recall that since PTRIG SGL is used (line 70), the first pulse occurs immediately and is not used. Thus, four pulses are required for each pass. The first pulse is required for the scan trigger (line 50) which also closes channel 0. The second and third pulses advance the scan from channel 0 to channel 1, and from channel 1 to channel 2. The fourth pulse opens channel 2 thus completing the first scan. The remaining eight pulses initate and advance the second and third scans by the same method.

www.valuetronics.com

Other Applications

Aside from its use with backplane scans, the system pacer can also be used to trigger or suspend system activities. The program that follows is a modified version of a previous program where an HP-IB group execute trigger (TRIGGER 709) is used to simultaneously trigger two voltmeters. By connecting the PACER OUT BNC to the SYSTEM TRIGGER IN BNC and modifying the program as shown below, the pacer can be used as a system trigger.

```
10   OUTPUT 709;"USE 700"
20   OUTPUT 709;"CONF DCV"
30   OUTPUT 709;"TERM EXT"
40   OUTPUT 709;"TRIG SYS"
50   OUTPUT 709;"USE 200"
60   OUTPUT 709;"CONF DCV"
70   OUTPUT 709;"TERM EXT"
80   OUTPUT 709;"TRIG SYS"
90   OUTPUT 709;"TRG EXT"
100  OUTPUT 709;"PACER 1"
110  OUTPUT 709;"PTRIG SGL"
120  OUTPUT 709;"CHREAD 700"
130  ENTER 709;A
140  OUTPUT 709;"CHREAD 200"
150  ENTER 709;B
160  PRINT A,B
170  END
```

The program shown below uses the HP 3852A's WAITFOR command to hold off the measurement of 10 channels until a pacer pulse occurs. In the program, WAITFOR PACER (line 80) is specified which means the pacer pulse is sensed internally. If WAITFOR EVENT were specified and the PACER OUT BNC were connected to the EVENT IN BNC, the pacer pulse would be sensed externally.

```
10   OUTPUT 709;"USE 700"
20   OUTPUT 709;"PACER 1"
30   OUTPUT 709;"PDELAY 4"
40   OUTPUT 709;"PTRIG"
50   OUTPUT 709;"CONF DCV"
60   OUTPUT 709;"STRIG PACER"
70   OUTPUT 709;"SADV PACER"
80   OUTPUT 709;"WAITFOR PACER"
90   OUTPUT 709;"MEAS DCV 600-609"
100  END
```


www.valuetronics.com

When the program runs, the measurements are held off for approximately four seconds (lines 30,40,80) following configuration of the voltmeter. Line 20 sets the pacer so that a pulse occurs every second (continuous stream). Following the delay, the 10 channels are measured one second apart (lines 70 and 90).

System Timing

This section explains how to set and read the mainframe's real-time clock and Julian date calendar. Also included is information on how the clock can be used to establish programming benchmarks.

Setting the Clock and Calendar

The clock and calendar are set, and their value returned, in the format of "seconds-since-midnight", with a maximum of 86400 seconds in a 24-hour period. The calendar portion of the real-time clock returns the Julian date, also expressed in seconds. Julian date and time are based on the standard Julian calendar. The internal clock has a resolution of one millisecond, and an accuracy of 0.004% of elapsed time since its last setting. The clock is battery-backed up and will run for up to four years without line power.

Setting the Clock

The real time clock is set using the HP 3852A's SET TIME command. This command has the syntax:

SET TIME *seconds*

The *seconds* parameter is the number of seconds since midnight. The range for *seconds* is 0 to 86399.999.

The real-time clock can be set from the front panel or from the controller. The following examples show how the clock is set using the front panel, an HP Series 200/300 controller, and a controller other than a Series 200/300.

NOTE

Examples which involve the Series 200/300 TIME and TIMEDATE commands assume the user has previously set the current date/time on the Series 200/300.

Note that in each example, the time is set to 4:30:46 PM, using the 24-hour international time base.

www.valuetronics.com

Setting the Time from the Front Panel

Before setting the time from the HP 3852A front panel, first convert the time to seconds since midnight using the formula:

(hours x 3600) + (minutes x 60) + seconds

4:30:46 = (16 x 3600) + (30 x 60) + 46 = 59446 seconds

When calculating the seconds, include an additional minute or so such that you have time to enter the command. For example, assuming the time is currently 4:27, you would use the equation to calculate the seconds representing 4:30:46 (59446).

Then:

1. From the shifted keyboard, spell out the command SET TIME (be sure to include a space between SET and TIME).

2. Add a space following the word TIME then enter the seconds calculated previously. Do not press the ENT key yet!

3. While monitoring your watch, a clock, or other source, press the ENT key at the precise time the source reaches the time for which the seconds were calculated. This sets the real time clock to the same time as the source (e.g. local time).

Setting the Time Using an HP Series 200/300 Controller

Using an HP Series 200/300 controller, the time can be set using any of the following methods.

1. OUTPUT 709;"SET TIME 59446"

Using this method, the desired setting is converted to seconds using the formula given above. The seconds are then specified directly in the command.

2. OUTPUT 709;"SET TIME ";TIME("16:30:46")

With this method, the controller's TIME command is used to make the seconds since midnight conversion.

3. OUTPUT 709;"SET TIME ";TIMEDATE MOD 86400.

This method sets the HP 3852A's real-time clock to the **controller's real-time clock setting** as reported by the controller's TIMEDATE command.

www.valuetronics.com

NOTE

Using this method to set the HP 3852A's real-time clock is useful for synchronizing mainframe time with controller time.

Setting the Time Using Other Controllers

For controllers which do not have functions similar to TIME and TIMEDATE, the HP 3852A's real-time clock can be set as shown below.

1. OUTPUT 709;"SET TIME 59446"

Here the time is converted to seconds using the formula then substituted directly into the command.

2. OUTPUT 709;"SET TIME ";16*3600.+30*60+46

In this program line, the numeric expression (formula) is evaluated by the controller (time is converted to seconds) then issued as the *seconds* parameter.

NOTE

The HP 3852A's real-time clock setting does not change when the instrument is cleared, reset, or if power is cycled. The setting is changed only by subsequent SET TIME commands.

Setting the Calendar

In order to set the HP 3852A's Julian calendar date, you must also set the time. The command used to set these parameters is the SET TIMEDATE command. This command has the syntax:

SET TIMEDATE *seconds*

The *seconds* parameter is the Julian date and time expressed in seconds. The range for *seconds* is 2.08662912E+11 (midnight March 1, 1900) minimum setting, through 4.768629999E+11 (11:59:59.999 February 29, 10,400) maximum setting.

If the HP 3852A's memory is lost, the clock and calendar are set to 2.08662912E+11 which represents midnight March 1, 1900. Time accumulates from that setting until changed by SET TIME or SET TIMEDATE.

www.valuetronics.com

The HP 3852A's real-time clock and calendar can be set from the front panel or from a controller. The following examples show how the time and date are set using an HP Series 200/300 controller and a controller other than a Series 200/300.

Setting the Time and Date Using an HP Series 200/300 Controller

1. OUTPUT 709;"SET TIMEDATE ";TIMEDATE

Using this method, the Series 200/300 controller's TIMEDATE command is used to make the seconds conversion. Note that this sets the HP 3852A's Julian date and time to the **controller's date and time setting** as reported by the TIMEDATE command.

2. OUTPUT 709;"SET TIMEDATE ";DATE("26 SEP 1987")+TIME("16:30:46")

This example uses the controller's DATE and TIME commands to make the seconds conversion. This method also sets the HP 3852A's time and date to the same settings as the controller.

Setting the Time and Date With Other Controllers

For controllers that do not have functions similar to TIMEDATE, DATE, and TIME, the following routine can be used to set the HP 3852A's time and date to any period within the allowable range. This routine can also be used by Series 200/300 controllers when you want to set a time and date that is different from the controller settings.

```
10   !Declare date and time variables. Assign date and time to be
20   !converted.
30   !
40   REAL Day,Month,Year,Hours,Minutes,Seconds,Td
50   Day=26
60   Month=9
70   Year=1987
80   Hours=16
90   Minutes=30
100  Seconds=46
110  !
120  !Call subroutine to make date and time seconds conversion.
130  !Execute SET TIMEDATE command once the conversion is made.
140  !
150  CALL Convrt(Day,Month,Year,Hours,Minutes,Seconds,Td)
```


www.valuetronics.com

```
160  OUTPUT 709 USING "K,D.14DE";"SET TIMEDATE ";Td
170  END
180  !
190  !Beginning of seconds conversion subroutine.
200  !
210  SUB Convrt(Day,Month,Year,Hours,Minutes,Seconds,Td)
220  DIM Days(1:12)
230  !
240  !Last Julian day of previous month with Julian year referenced to
250  !March 1.
260  !
270  !     J  F  M  A  M  J  J  A  S  O  N  D
280  !
290  DATA 306,337,0,31,61,92,122,153,184,214,245,275
300  READ Days(*)
310  !
320  !Determine Julian day of current month and day
330  !
340  Td=Days(Month)+Day-1
350  !
360  !Determine number of years since March 1, 1900
370  !
380  Jul_year=Year-1900
390  IF Month<3 THEN Jul_year=Jul_year-1
400  !
410  !Calculate number of days since March 1, 1900
420  !
430  Td=Td+Jul_year*365.+Jul_year DIV 4
440  !
450  !Compensate for leap years
460  !
470  Td=Td+Jul_year DIV 100+(Jul_year+300.) DIV 400.
480  !
490  !Seconds conversion
500  !
510  Td=Td*86400.+2.08662912E+11+Hours*3600.+Minutes*60+Seconds
520  SUBEND
```

Reading the Real-Time Clock

The command used to read the HP 3852A's real-time clock is the TIME command. This command has the syntax:

TIME [INTO *name*] or [*fmt*]

The time returned by the TIME command is the number of seconds since midnight. The **INTO** *name* and *fmt* parameters indicate this value can be stored in mainframe memory or entered into the controller/displayed in a particular data format. The TIME command can be entered from the front panel or from a controller.

www.valuetronics.com

Converting the Seconds Returned by the TIME Command

In order to determine the actual time based on the number of seconds returned, the seconds must be converted to an hours:minutes:seconds format. Two conversion routines follow. The first routine is suited to HP Series 200/300 controllers and the second routine applies to any controller.

Converting Seconds to Hours:Minutes:Seconds (Series 200/300 Controller)

```
10   OUTPUT 709;"TIME"
20   ENTER 709;T
30   PRINT TIME$(T)
40   END
```

This program uses the controller's TIME$ command to convert the number of seconds returned by the HP 3852A's TIME command to an HH:MM:SS format. Assuming a time of 4:30:46 PM, the time returned when this program executes might be:

16:30:49 (assuming a 3 second time lapse)

Converting Seconds to Hours:Minutes:Seconds (Any Controller)

```
10   OUTPUT 709;"TIME"
20   ENTER 709;T
30   Hour=T DIV 3600
40   Minute=(T MOD 3600) DIV 60
50   Second=T MOD 60
60   PRINT USING "2D,"":"",2Z,"":"",2Z";Hour,Minute,Second
70   END
```

This program also converts the number of seconds returned by the HP 3852A's TIME command (line 10) to a HH:MM:SS format.

Reading the Calendar

The Julian date and time are read with the HP 3852A's TIMEDATE command. This command has the syntax:

TIMEDATE [INTO *name*] or [*fmt*]

The data returned by the TIMEDATE command is the current value of the real-time clock. The **INTO** *name* and *fmt* parameters indicate that this value can be stored in mainframe memory or entered into the controller/displayed in a particular data format. The TIMEDATE command can be entered from the front panel or from a controller.

www.valuetronics.com

Converting the Seconds Returned by the TIMEDATE Command

Similar to the TIME command, the seconds returned by the TIMEDATE command must be converted to a day/month/year, hours:minutes:seconds format to determine the actual date and time. Two conversion routines follow. The first routine is suited to Series 200/30 controllers and the second routine applies to any controller.

Converting TIMEDATE Seconds to Day, Month, Year - Hours:Minutes:Seconds (Series 200/300 Controller)

```
10    OUTPUT 709;"TIMEDATE"
20    ENTER 709;A
30    PRINT DATE$(A),TIME$(A)
40    END
```

This program uses the controller's DATE$ and TIME$ commands to convert the seconds returned by the HP 3852A's TIMEDATE command to a day/month/year and HH:MM:SS format. Assuming a date and time of 26 September 1987 - 4:30:46 PM when the program executes, the following might be displayed:

```
26 Sep 1987 16:30:47
```

Converting TIMEDATE Seconds to Day, Month, Year - Hours:Minutes:Seconds (Any Controller)

For controller's that do not have functions similar to DATE$ and TIME$, the following routine can be used to convert the seconds returned by the HP 3852A's TIMEDATE command.

```
10    !Execute HP 3852A TIMEDATE command and enter seconds returned
20    !into controller.
30    !
40    OUTPUT 709;"TIMEDATE"
50    ENTER 709;Td
60    !
70    !Call subroutine to convert the seconds returned into day,
80    !month, year - HH:MM:SS format.
90    !
100   CALL Cnvrt(Date$,Time$,Td)
110   !
120   !Print the results of the conversion
130   !
140   PRINT Date$,Time$
150   END
160   !
170   !Beginning of seconds conversion subroutine.
```


www.valuetronics.com

```
180  !
190  SUB Cnvrt(Date$,Time$,Td)
200  DIM Month$(0:11)[3],Days(0:12)
210  !
220  !Last Julian day of month with Julian year referenced to March 1.
230  !
240  DATA MAR,APR,MAY,JUN,JUL,AUG,SEP,OCT,NOV,DEC,JAN,FEB
250  DATA 0,31,61,92,122,153,184,214,245,275,306,337,366
260  READ Month$(*),Days(*)
270  !
280  !Perform year, day, and month conversions.
290  !
300  Day=(Td-2.08662912E+11) DIV 86400.+109572.
310  Year=1600.+(Day DIV 146097.)*400.
320  Day=Day MOD 146097.
330  Day=Day+Day DIV 36524.-(Day=146096.)
340  Year=Year+(Day DIV 35625.)*100.
350  Day=Day MOD 36525.
360  Year=Year+(Day DIV 1461.)*4
370  Day=Day MOD 1461.
380  Day=Day+Day DIV 365.-(Day=1460.)
390  Year=Year+Day DIV 366.
400  Day=Day MOD 366.
410  Month=0
420  WHILE Day>=Days(Month+1)
430  Month=Month+1
440  END WHILE
450  Day=Day-Days(Month)+1
460  IF Month>9 THEN Year=Year+1
470  !
480  !Read day, month, and year conversion into variable Date$.
490  !
500  OUTPUT Date$ USING "#,2D,X,K,X,5D";Day,Month$(Month),Year
510  !
520  !Perform hours, minutes, and seconds conversion.
530  !
540  Time=Td MOD 86400
550  Hour=Time DIV 3600
560  Minute=(Time MOD 3600) DIV 60
570  Second=Time MOD 60
580  !
590  !Read hours, minutes, and seconds conversion into variable TIME$.
600  !
610  OUTPUT Time$ USING "#,2D,"":"",2Z,"":"",2Z";Hour,Minute,Second
620  SUBEND
```


www.valuetronics.com

The HP 3852A Alarm

Associated with the HP 3852A's real-time clock is an alarm function. The alarm is useful for scheduling measurements and other system functions. The alarm is set with the HP 3852A's SET ALRM command. This command has the syntax:

SET ALRM *seconds*

The *seconds* parameter is the number of seconds since midnight. The range for seconds is 0 to 86399.999. The time to which the alarm is set is stored in battery backed-up memory. If memory is lost, the alarm time is set to 0 (midnight). SET ALRM can be executed from the front panel or over the HP-IB.

When setting the alarm, note that it is referenced to the real-time clock. For example, if the local time is 2:00 PM, but the real-time clock reads 4:00 PM, and you set the alarm to occur at 3:00 PM, the alarm will not occur until 3:00 PM the following day (1:00 PM local time). To ensure that the alarm is set as intended, the real-time clock should be set to your local time.

Similar to setting the time with the SET TIME command, the time at which the alarm is to occur must be converted to seconds since midnight. Recall that this can be accomplished using the formula:

(hours x 3600) + (minutes x 60) + seconds

Also, any of the methods described under "Setting the Clock" can be used to make this conversion.

In the following example, the clock is set to 9:00 AM (current local time) and the alarm is set to 9:00:10 AM. For both settings, the formula listed above is used to make the seconds conversion.

Clock:

9:00 AM = (9 x 3600) + (0 x 60) + 0 = 32400 seconds since midnight

Alarm:

9:00:10 AM = (9 x 3600) + (0 x 60) + 10 = 32410 seconds since midnight

```
60    OUTPUT 709;"SET TIME 32400"
70    OUTPUT 709;"SET ALRM 32410"
```


www.valuetronics.com

For the alarm to be recognized by the mainframe when it occurs, the alarm must be enabled. The following program shows how the alarm is enabled and uses the alarm to call a subroutine.

NOTE

Refer to Chapter 8 - Using Interrupts for detailed information on using the HP 3852A's alarm to initiate (schedule) system functions.

```
10    OUTPUT 709;"RST"
20    OUTPUT 709;"SUB ALRMCHK"
30    OUTPUT 709;"DISP'ALARM HAS OCCURRED"'
40    OUTPUT 709;"SUBEND"
50    OUTPUT 709;"ON ALRM CALL ALARMCHK"
60    OUTPUT 709;"SET TIME 32400"
70    OUTPUT 709;"SET ALRM 32410"
80    OUTPUT 709;"ENABLE ALRM"
90    END
```

The reset command in line 10 enables this program to be executed repeatedly without an error due to downloading a subroutine which already exists. Again, line 60 sets the clock and line 70 sets the alarm to occur 10 seconds after the clock setting. When the alarm occurs, the subroutine is called and "ALARM HAS OCCURRED is displayed on the mainframe front panel.

Reading the Alarm Setting

The command used to read the alarm setting is the ALRM command. This command has the syntax:

ALRM [INTO *name*] or [*fmt*]

The setting returned by the alarm command represents the number of seconds since midnight. The **INTO** *name* and *fmt* parameters indicate this value can be stored in mainframe memory or entered into the controller/displayed in a particular format.

To determine the actual alarm setting, the number of seconds returned must be converted to an hours:minutes:seconds format. Any of the methods used for converting the seconds returned by the TIME command can also be used for the ALRM command. Simply replace TIME with ALRM in the various routines (see - "Converting the Seconds Returned by the TIME Command").

www.valuetronics.com

To verify the alarm setting of the previous program, we'll modify the program and use the Series 200/300 controller's TIME$ command to make the conversion. Again, note that the alarm setting is referenced to the HP 3852A's real-time clock.

```
10   OUTPUT 709;"RST"
20   OUTPUT 709;"SUB ALRMCHK"
30   OUTPUT 709;"DISP'ALARM HAS OCCURRED'"
40   OUTPUT 709;"SUBEND"
50   OUTPUT 709;"ON ALRM CALL ALARMCHK"
60   OUTPUT 709;"SET TIME 32400"
70   OUTPUT 709;"SET ALRM 32410"
80   OUTPUT 709;"ENABLE ALRM"
90   OUTPUT 709;"ALRM"
100  ENTER 709;A
110  PRINT TIME$(A)
120  END
```

When this program executes, the alarm setting will be displayed on the controller as shown below. The setting (32410) is temporarily displayed on the mainframe and is then replaced by the message, "ALARM HAS OCCURRED".

```
09:00:10
```

The POWEROFF Command

The HP 3852A's POWEROFF command is used to determine the date and time the HP 3852A was last turned off or when power was otherwise removed. This command has the syntax:

POWEROFF [INTO *name*] or [*fmt*]

The date and time of the last power down is expressed in seconds. The **INTO** *name* and *fmt* parameters indicate that this value can be stored in mainframe memory or entered into the controller/displayed in a particular data format.

To determine the actual date and time of the last powerdown, the seconds returned must be converted to a day/month/year - hours:minutes:seconds format. Any of the methods used for converting the seconds returned by the HP 3852A's TIMEDATE command can also be used for the POWEROFF command. It is only necessary to replace TIMEDATE with POWEROFF in the various routines (see - "Converting the Seconds Returned by the TIMEDATE Command").

www.valuetronics.com

The date and time returned by the POWEROFF command is referenced to the setting of the HP 3852A's real-time clock. For meaningful readings, the real-time clock and Julian date calendar should be set to the current date and local time before any possible power losses may occur. Otherwise, for example, a power loss that you knew occurred on January 17, 1987 at 9:30 AM, may be reported by POWEROFF as a different day and time if the clock and calendar are not accurately set.

The following programs demonstrate the use of the POWEROFF command. The first program sets the HP 3852A's real-time clock and Julian calendar to September 26, 1987; 4:30:46 PM. Prior to the second program, the HP 3852A is turned off and then turned back on. When the second program runs, the POWEROFF command is executed and the date and time of the last power loss is reported. Note that the program uses the Series 200/300 controller's DATE$ and TIME$ commands to convert the seconds returned to a day/month/year - hours:minutes:seconds format.

```
10   REAL Day,Month,Year,Hours,Minutes,Seconds,Td
20   Day=26
30   Month=9
40   Year=1987
50   Hours=16
60   Minutes=30
70   Seconds=46
80   CALL Convrt(Day,Month,Year,Hours,Minutes,Seconds,Td)
90   OUTPUT 709 USING "K,D.14DE";"SET TIMEDATE ";Td
100  END
110  SUB Convrt(Day,Month,Year,Hours,Minutes,Seconds,Td)
120  DIM Days(1:12)
130  DATA 306,337,0,31,61,92,122,153,184,214,245,275
140  READ Days(*)
150  Td=Days(Month)+Day-1
160  Jul_year=Year-1900
170  IF Month<3 THEN Jul_year=Jul_year-1
180  Td=Td+Jul_year*365.+Jul_year DIV 4
190  Td=Td+Jul_year DIV 100+(Jul_year+300.) DIV 400.
200  Td=Td*86400.+2.08662912E+11+Hours*3600.+Minutes*60+Seconds
210  SUBEND
```

(HP 3852A power is cycled)

```
10   OUTPUT 709;"POWEROFF"
20   ENTER 709;P
30   PRINT DATE$(P),TIME$(P)
40   END
```


www.valuetronics.com

When the second program executes, a date and time similar to the following should be displayed:

```
26 Sep 1987        16:30:48
```

The Wait Command

The HP 3852A's WAIT command enables you to pause program execution. The WAIT command has the syntax:

WAIT [*number*]

The *number* parameter is the number of seconds (rounded to the nearest thousandth of a second) the program is paused. The range for *number* is 0 to 86400 seconds. The default setting is 0.

Although the WAIT command references the real-time clock, it does not require any particular clock setting. The following example shows how the WAIT command can be used.

```
10    OUTPUT 709;"USE 700"
20    OUTPUT 709;"CONF DCV"
30    OUTPUT 709;"WAIT 5"
40    OUTPUT 709;"MEAS DCV 600"
50    END
```

When this program executes, the voltmeter accessory in slot 7 is configured to measure DC voltage. After a five second delay (line 30), the measurement is taken.

Establishing Benchmarks

In a data acquisition system, it is often helpful to know the rate at which data is acquired. The speed of measurements, conversions, etc., can be determined using the HP 3852A's real-time clock. This process involves taking a reading of the clock prior to the event to be timed, and a reading immediately after the event has occurred. The difference in readings is the time required for the measurement, conversion, etc.

The following example establishes as a benchmark, the time required to measure strain on three channels and store the results in mainframe memory (line 220). In the example, a ¼ bridge configuration is used. The clock readings are taken using the HP 3852A's TIME command.

www.valuetronics.com

```
10   !Declare controller array for strain readings. Declare HP 3852A
20   !timing variables, a mainframe array for strain measurements, and a
30   !mainframe array for the unstrained reference readings.
40   !
50   REAL Strain(0:2)
60   OUTPUT 709;"REAL T1,T2,T3"
70   OUTPUT 709;"REAL STRRDGS (2)"
80   OUTPUT 709;"REAL STR_REF(2)"
90   !
100  !Specify voltmeter to be used and perform unstrained reference
110  !measurements.
120  !
130  OUTPUT 709;"USE 700"
140  OUTPUT 709;"CONFMEAS STRUN 600-602 INTO STR_REF"
150  !
160  !Begin benchmark subroutine. Take clock reading just prior to the
170  !measurement to be timed. Perform measurement. Take clock reading
180  !immediately after measurement.
190  !
200  OUTPUT 709;"SUB STRTIME"
210  OUTPUT 709;"TIME INTO T1"
220  OUTPUT 709;"CONFMEAS STRQ 600-602 REF STR_REF GF 2E-6 INTO STRRDGS"
230  OUTPUT 709;"TIME INTO T2"
240  OUTPUT 709;"SUBEND"
250  !
260  !Apply stress to beam before subroutine is called.
270  !
280  OUTPUT 709;"WAIT 5"
290  OUTPUT 709;"CALL STRTIME"
300  !
310  !Calculate time of CONFMEAS command. Transfer benchmark to output
320  !buffer. Enter benchmark into controller.
330  !
340  OUTPUT 709;"T3=T2-T1"
350  OUTPUT 709;"VREAD T3"
360  ENTER 709;A
370  !
380  !Transfer strain measurements to output buffer. Enter measurements
390  !into controller.
400  !
410  OUTPUT 709;"VREAD STRRDGS"
420  ENTER 709;Strain(*)
430  !
440  !Print benchmark and measurements
450  !
460  PRINT A
470  PRINT
480  PRINT USING "K,/";Strain(*)
490  END
```


www.valuetronics.com

In this program we were interested in the time required to make the strain measurements and store the results (line 220). By reading the time just prior to the measurement (line 210) and again immediately after (line 230), that period was determined. Executing the CONFMEAS command within a subroutine represents the fastest possible method of executing the command. This is because the command is compiled when the subroutine is downloaded.

Note that in this example, we are not concerned with the real-time clock setting; only the difference in times. For events that may be measured over a period of days in which the time is read with the TIMEDATE command, the clock setting should be checked.

This program also contains commands for reading (converting) the time and reading the strain measurements. When this program runs, the controller should display something similar to the following:

```
.293

299.8768
711.5053
1119.439
```

From this listing, the time required to measure the strain and store the results was 293 milliseconds. The actual measurements are also listed.

Again, when establishing benchmarks, the event which is timed is the command or program segment that is enclosed by HP 3852A TIME or TIMEDATE commands.

www.valuetronics.com

Contents

www.valuetronics.com

Chapter 8

Using Interrupts

Introduction

This chapter explains how to use interrupts from the HP 3852A plug-in accessories and alarm or limit exceptions from the HP 3852A mainframe to alter the ongoing operation of the mainframe or the controller. The process of enabling interrupts, responding to interrupts, and monitoring the status of interrupts is covered in the three main sections of this chapter: Introduction, Handling Interrupts with the Mainframe, and Handling Interrupts with the Controller. An overview of each section is given below.

- **Introduction.** This section which is the present section, identifies the sources from which an interrupt signal can originate. The section explains how the mainframe "services" an interrupt and contains information on interrupt priorities based on the source, and on how the mainframe responds when multiple interrupts signals are received. The section concludes by identifying the plug-in accessories with interrupt capability.

- **Handling Interrupts with the Mainframe.** Once an interrupt occurs, either the mainframe or the controller can handle the interrupt. This section explains how the mainframe is enabled to sense and respond to an interrupt independent of the controller. The section also includes information on how mainframe interrupt sensing and handling capabilities are disabled and how interrupts are cleared. The section concludes with several detailed examples.

- **Handling Interrupts with the Controller.** This section explains how the mainframe is enabled to sense an interrupt and send a service request (SRQ) message to the controller so that it may respond. Included is information on unmasking bits in the mainframe's Status Register and on clearing the bits after an interrupt occurs. Several examples are included to demonstrate interrupts handled by the controller.

www.valuetronics.com

Interrupt Sources

An interrupt is used to suspend the current, ongoing operation of the HP 3852A mainframe or controller. Once operation is suspended, a higher priority program or routine within the mainframe or controller is called in response to the condition which caused the interrupt. After the appropriate action is taken, normal operation of the HP 3852A (usually) continues.

In an HP 3852A based system, there are nine sources from which an interrupt can originate. Note that for a particular source to cause an interrupt that will suspend or halt operation, that source must be properly set up and enabled which is the subject of this chapter. The nine sources are represented by bits in the HP 3852A Status Register and several sources are also indicated by annunciators in the display. The source of an interrupt is recorded by the Status Register (and annunciator) when the corresponding bit is set from a binary "0" to a binary "1".

A brief description of each interrupt source, its representative bit in the Status Register, and its annunciator are presented in Table 8-1. The Status Register is covered in greater detail in a later section of this chapter.

Handling Interrupts

Interrupt handling is a user-defined response by the mainframe or controller when notified of an interrupt. Prior to receiving an interrupt, the user must have determined whether the mainframe or controller will respond. Handling an interrupt with the mainframe involves the use of the commands:

ON *event* **CALL** *name*

or

WAITFOR *event*

The ON command calls an HP 3852A subroutine in response to the interrupt and the WAITFOR command suspends mainframe operation until an accessory interrupt occurs. Note that the HP 3852A can only handle interrupts from the following sources:

- **Alarm**
- **Limit**
- **Backplane**

A controller, on the other hand, can handle interrupts from any of the sources identified in Table 8-1.

www.valuetronics.com

Table 8-1. HP 3852A Interrupt Sources

Source	Bit	Annunciator	Description
Alarm	ALRM	ALRM	This exception (interrupt) is an alarm from the mainframe's internal clock. The alarm occurs when the clock equals the time previously set by the SET ALRM command.
Limit	LMT	LMT	This exception (interrupt) is due to a measurement outside predefined limits established during real-time limit testing.
Backplane	INTR	INTR	The source of this interrupt is an accessory channel. Presently, HP 3852A accessories with interrupt capability include the HP 44701A, HP 44702A/B, HP 44714A, HP 44715A, HP 44721A/22A, HP 44723A, and the HP 44726A.
Error	ERR	ERR	An interrupt from this source is a result of a configuration or programming error. The error and its error message are stored in the mainframe's error buffer.
Mainframe Ready	RDY	—	An interrupt from this source occurs when the mainframe's input buffers (HP-IB, front panel) are empty and no command or subroutine is executing.
Return to Local	LCL	—	An interrupt from this source occurs when front panel control of the mainframe is restored by the LOCAL key.
Front Panel Service Request	FPS	SRQ	When enabled, an interrupt from this source occurs when the SRQ command is sent or when the SRQ key is pressed.
Mainframe Power Loss	PWR	—	An interrupt from this source occurs when the mainframe's power supply falls below +18V or when an HP 3853A extender connected to the mainframe is turned off.
Data Available	DAV	DAV	A data available interrupt occurs when data from a command is in the mainframe's HP-IB output buffer.

www.valuetronics.com

Servicing Interrupts

Interrupt servicing is the detection of an interrupt and signaling to the user that one occurred. In an HP 3852A system, every interrupt that occurs is serviced by the HP 3852A regardless of whether it is also handled by the HP 3852A or handled by the controller.

Phase 1

Interrupt servicing is a two phase process. During phase 1, the HP 3852A's processor senses the interrupt and its source and sets the appropriate bit in the mainframe Status Register. If the mainframe's service request (SRQ) capability is enabled, an SRQ message is immediately sent to the controller.

When the interrupt source is an alarm (ALRM), the mainframe also disables the alarm which prevents it from causing another interrupt. This is equivalent to the user issuing the DISABLE ALRM command. (Note - limit interrupts are not disabled.)

The first backplane interrupt that occurs will set the INTR Status Register bit indicating an accessory channel interrupt occurred. When this happens, the mainframe disables its backplane (i.e. accessory) interrupt sensing capability. This is equivalent to the user issuing the DISABLE INTR SYS command. Note that while this sensing capability is disabled, other accessory interrupts can occur which were not responsible for setting the bit in the Status Register. These interrupts will be detected during the polling routine of the accessories and accessory channels that occurs during phase 2 of the servicing routine.

Phase 2

Phase 2 begins when the mainframe first enters an idle state. An idle state is when the mainframe completes the command it was executing when the interrupt occurred or when it finishes the subroutine that was executing when the interrupt occurred. For interrupts where the source is an alarm or a limit that was exceeded, phase 2 calls an HP 3852A subroutine (if) previously defined to respond to the interrupt.

For backplane (accessory) interrupts, phase 2 begins with a polling routine initiated by the mainframe to determine the specific accessory channel(s) on which there is an interrupt. The routine (Figure 8-1) begins by polling the mainframe and each extender (if any) by increasing extender number to determine the frame the interrupt came from. Once the frame is known, each slot in the frame is polled beginning with slot 0 and continuing in increasing order. Once the slot has been determined, each channel in the slot is polled beginning with channel 0. Once the interrupting channel is found, the mainframe disables its interrupt capability. This is equivalent to the user issuing DISABLE INTR USE ch, where ch is the channel that interrupted. At this point, the mainframe's backplane interrupt sensing capability temporarily disabled in phase 1 is re-enabled (equivalent to sending ENABLE INTR SYS). If the interrupt is handled by the

www.valuetronics.com

mainframe, a subroutine is then called. Note that if other interrupts occurred prior to when ENABLE INTR SYS was re-enabled, the polling routine will begin again after the handling of the current interrupt is complete.

For alarm and limit interrupts handled by the controller and for error, mainframe ready, return to local, front panel service request, mainframe power loss, and data available interrupts, there is no phase 2 servicing on the part of the mainframe.

NOTE

Because of the polling sequence used, the first interrupt serviced may not have been the interrupt responsible for setting the INTR bit in the Status Register (see Backplane Interrupt Priority).

If DISABLE INTR SYS was executed between phases 1 and 2, ENABLE INTR SYS will not be re-enabled during phase 2.

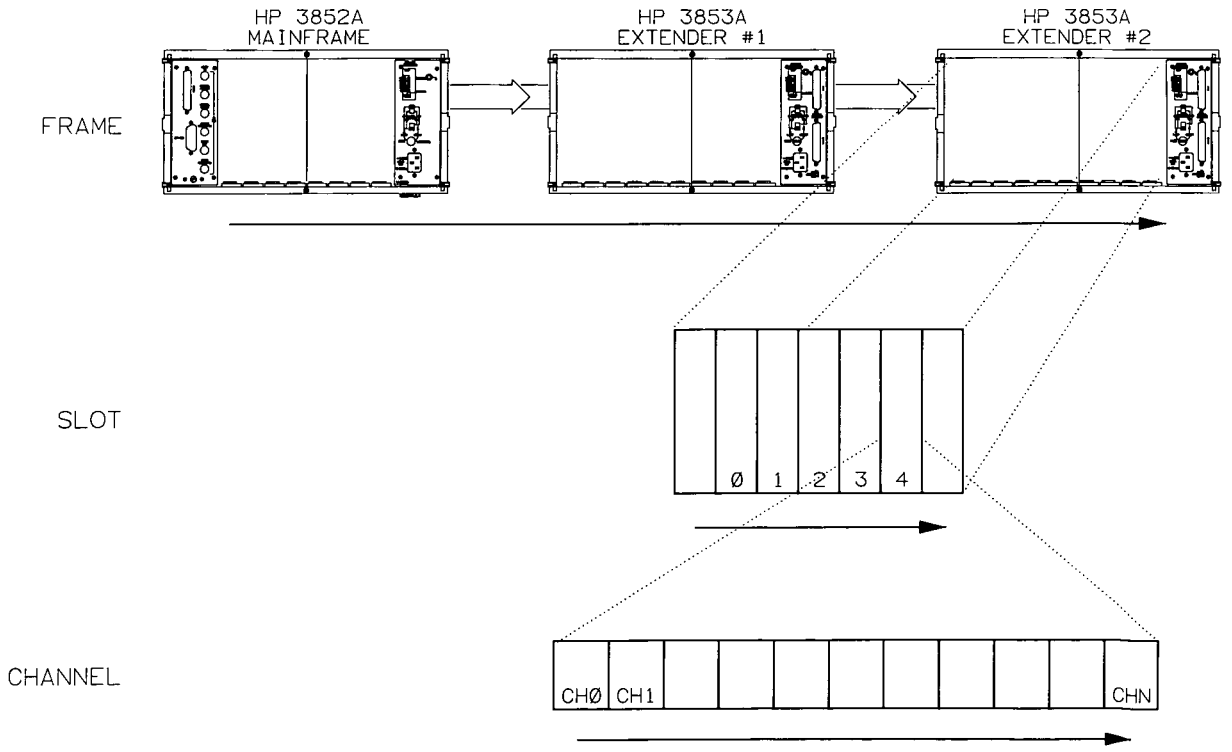


Figure 8-1. HP 3852A Polling Routine

www.valuetronics.com

The INTR? Command

The last backplane interrupt serviced by the mainframe can be determined with the INTR? command. This command has the syntax:

INTR? [**INTO** *name*] or [*fmt*]

INTR? returns the address of the channel interrupt serviced last. If no channel was serviced, -1 is returned. The [**INTO** *name*] and [*fmt*] parameters indicate that the address returned can either be stored in mainframe memory or entered/displayed in a particular format.

Interrupt Priorities

The interrupt with the highest priority is the interrupt which is serviced first. In an HP 3852A based system where the mainframe will also handle the interrupt, interrupt priority is as follows:

1. Alarm interrupts.
2. Backplane (accessory) interrupts.
3. Limit interrupts.

When interrupts are handled by the controller, interrupt priority is established by the controller.

Backplane Interrupt Priority

Recall that the phase 2 polling routine (Figure 8-1) locates the interrupting channel by polling the mainframe, any extenders, each slot beginning with slot 0, then each channel beginning with channel 0. Thus the highest priority channel for an accessory interrupt is channel 0 in slot 0 of the mainframe.

Multiple Accessory Interrupts

When an interrupt occurs on an accessory channel and sets the bit in the Status Register, the mainframe disables its backplane interrupt sensing capability. While this capability is disabled, however, additional accessory interrupts may occur. The first channel interrupt encountered during the polling routine is serviced then handled by the mainframe or the controller. Once the interrupt has been handled (i.e. the mainframe returns to an idle state), the polling routine begins again to locate another interrupt. Note that the routine starts at the beginning of the sequence and not where it stopped after the first interrupting channel was located. The polling routine is repeated until all accessory interrupts are located and serviced.

Interrupt Frequency

Since the polling routine restarts at the beginning once an interrupting channel is serviced and handled, it is possible that interrupts occurring on higher numbered channels would not get serviced if interrupts on lower numbered channels occur frequently enough, thus resetting the polling sequence. Therefore, interrupts with a high priority or those which occur very infrequently should be enabled on the lower channels of an accessory with a low slot number.

www.valuetronics.com

Accessory Interrupt Capability

Presently, the HP 3852A plug-in accessories with interrupt capability are the HP 44701A and HP 44702A/B voltmeter accessories, the HP 44714A Stepper Motor Controller/Pulse Output, the HP 44715A 5-Channel Counter, the HP 44721A/22A 16/8-Channel Digital Inputs, the HP 44723A 16-Channel High-Speed Digital Sense/Control, and the HP 44726A 2-Channel Arbitrary Waveform DAC. These accessories can interrupt from any slot in the mainframe or extender.

Refer to the accessory manual for information on setting up a specific interrupt condition.

Using an Interrupt as a Trigger Source

Backplane (accessory) interrupts can also function as a trigger source for other accessories. The command:

ENABLE INTR BNC

connects the backplane interrupt line to the mainframe's CHANNEL CLOSED BNC. This enables an interrupt to be serviced by the mainframe, and initiate events independent of the mainframe. Refer to the Command Reference Manual for additional information and an example on the use of the command.

www.valuetronics.com

Handling Interrupts with the Mainframe

This section explains how to implement the use of interrupts in an HP 3852A based system. The section covers how the mainframe is configured to sense and handle interrupts independent of the controller. This section also explains why it is necessary to disable and clear interrupts and the differences between the two.

Enabling the Interrupt

As stated in the previous section, the mainframe services all interrupts but can only handle those interrupts whose source is an alarm or limit exception or a backplane (accessory) interrupt. The procedure for handling interrupts with the mainframe is essentially the same regardless of the source.

The steps involved with handling interrupts with the mainframe are given here. The commands associated with each step correspond to the sequence they should occur in your program.

Step 1 (Mainframe)

The most common method of handling interrupts with the mainframe is to call a subroutine in response to the condition which caused the interrupt. The first step is to setup a subroutine to be called when the exception or interrupt occurs. The commands used to do this are:

SUB *name*

.
.
.

SUBEND

ON *event* **CALL** *name*

SUB, SUBEND, and the commands between them form the subroutine. The use of HP 3852A subroutines is covered in Chapter 9. ON event CALL name calls the subroutine when the exception or interrupt occurs. The event parameters of the ON command are:

ALRM
LMT
INTR [USE *ch***]**

When ALRM is specified, the subroutine is called when an alarm occurs. When LMT is specified, the subroutine is called when a limit exception occurs. When INTR [USE ch] is specified, the subroutine is called when an interrupt on the channel specified by USE ch occurs.

www.valuetronics.com

The *name* parameter is the subroutine called.

The following program lines illustrate the use of these commands. This program enables and handles an interrupt from an HP 44721A digital input accessory.

```
10  OUTPUT 709;"RST"
20  OUTPUT 709;"SUB EXAMPLE"
30  OUTPUT 709;"DISP 'INTERRUPT HAS OCCURRED'"
40  OUTPUT 709;"SUBEND"
50  OUTPUT 709;"ON INTR USE 316 CALL EXAMPLE"
```

The OFF Command

In certain situations it may be desirable to prevent an alarm, limit, or channel interrupt from calling a subroutine without deleting the subroutine or changing the configuration which sets up the interrupt. One way to do this is with the OFF command. The OFF command has the syntax:

OFF *event*

The *event* parameters of the OFF command are:

ALRM
LMT
INTR [**USE** *ch*]

When ALRM is specified, the subroutine is not called when an alarm occurs. When LMT is specified, no subroutine is called when a limit occurs. When INTR [USE ch] is specified, no subroutine is called when an interrupt occurs on the USE ch specified.

Step 2 (Mainframe)

Depending on the source of the interrupt, the next step is to set the alarm, set up limit testing, or configure the accessory channel to interrupt on a desired condition. The commands used to set the alarm and set up limit testing are:

SET ALRM *time*

LMT *min max index_store*

To configure an accessory channel for a specific interrupt condition, refer to that accessory's configuration and programming manual. To continue the program, however, channel 0 of the digital input installed in mainframe slot 3 is configured to interrupt when a low to high signal transition occurs on the channel. Thus:

```
60  OUTPUT 709;"EDGE LH USE 316"
```


www.valuetronics.com

Step 3 (Mainframe)

Again, based on the source of the interrupt, the next step is to either enable the alarm to interrupt, initiate real-time limit testing, or enable the specific accessory channel to interrupt. The commands used to do this are:

ENABLE ALRM

ENABLE LMT

ENABLE INTR [USE *ch*]

Continuing the program, step 3 is to enable the specific accessory channel to interrupt. Thus,

```
70 OUTPUT 709;"ENABLE INTR USE 316"
```

Note that you can only enable one channel per ENABLE INTR command.

Step 4 (Mainframe)

Step 4 is to enable the mainframe to sense an interrupt from an accessory channel. The command used to do this is:

ENABLE INTR SYS

Note that Step 4 is required for backplane (accessory) interrupts only. Since the program developed in this procedure is for a backplane interrupt, Step 4 is executed as:

```
80 OUTPUT 709;"ENABLE INTR SYS"
```

The complete program listing is:

```
10 OUTPUT 709;"RST"
20 OUTPUT 709;"SUB EXAMPLE"
30 OUTPUT 709;"DISP 'INTERRUPT HAS OCCURRED'"
40 OUTPUT 709;"SUBEND"
50 OUTPUT 709;"ON INTR USE 316 CALL EXAMPLE"
60 OUTPUT 709;"EDGE LH USE 316"
70 OUTPUT 709;"ENABLE INTR USE 316"
80 OUTPUT 709;"ENABLE INTR SYS"
90 END
```

When this program is executed and a low to high signal transition occurs on channel 0 of the digital input, the following message is displayed on the mainframe:

```
INTERRUPT HAS OCCURRED
```


www.valuetronics.com

The reset (RST) command in line 10 resets the mainframe each time the program runs. If the mainframe were not reset, an error (59: SUB ALREADY EXISTS) would occur each time (except the first time) since the program tries to download a subroutine which already exists (EXAMPLE). Note that the program does run to completion when the interrupt occurs; however, the error is reported.

The program above can be simplified somewhat by letting the USE command specify the channel for which interrupt capability is enabled. This eliminates the need to specify the USE ch parameter as shown below:

```
10  OUTPUT 709;"RST"
20  OUTPUT 709;"USE 316"
30  OUTPUT 709;"SUB EXAMPLE"
40  OUTPUT 709;"DISP 'INTERRUPT HAS OCCURRED'"
50  OUTPUT 709;"SUBEND"
60  OUTPUT 709;"ON INTR CALL EXAMPLE"
70  OUTPUT 709;"EDGE LH"
80  OUTPUT 709;"ENABLE INTR"
90  OUTPUT 709;"ENABLE INTR SYS"
100 END
```

The steps for handling alarm and limit exceptions and backplane interrupts with the mainframe are summarized in Figure 8-2.

Clearing and Disabling Interrupts - Mainframe

When handling multiple interrupts from the same source, it is necessary to clear the interrupt each time it is serviced and handled so that the mainframe can detect and respond to the next interrupting event which occurs.

When an interrupt occurs, a signal line from the source to the mainframe microprocessor is pulled. Clearing the interrupt releases the signal line. Depending on the source, an interrupt may be cleared "internally" or may require a response from the user.

Disabling an interrupt prevents an interrupt from occurring without changing the configuration which set up the interrupting condition. The commands used to disable interrupts from the sources handled by the mainframe are:

DISABLE ALRM

DISABLE LMT

DISABLE INTR [USE *ch*]

Note that for certain sources and interrupt conditions, disabling an interrupt also clears the interrupt.

www.valuetronics.com

Clearing Alarm Interrupts

When the interrupt source is an alarm, the interrupt is cleared when alarm interrupt capability is disabled during the mainframe's servicing routine. Again, this is equivalent to the mainframe internally issuing the DISABLE ALRM command. Alarm interrupt capability is re-enabled with the ENABLE ALRM command; and, since the previous interrupt was cleared, the mainframe will sense the next alarm interrupt that occurs.

Limit Interrupts

Limit interrupts are not clearable interrupts nor is the interrupt capability disabled when a limit interrupt is serviced. Note that if during a measurement sequence (e.g. CONFMEAS THM10K 100-103) readings on more than one channel exceed a limit, it is recognized as a single interrupt.

Clearing Backplane Interrupts

The manner in which backplane interrupts are cleared depends on the accessory and the condition which caused the interrupt. Refer to the individual plug-in accessory manuals for information on the commands/ actions required to clear the interrupt.

DISABLE INTR SYS vs. DISABLE INTR

Backplane interrupts can be disabled at essentially two levels. DISABLE INTR SYS prevents the mainframe from sensing a backplane interrupt but does not disable channel interrupt capability. Recall that during the servicing routine for a backplane interrupt, interrupt sensing on the backplane is temporarily disabled. If other channels interrupt while the capability is disabled, when the capability is re-enabled, a backplane interrupt is sensed and the servicing routine begins again.

www.valuetronics.com

DISABLE INTR [USE ch] disables an accessory channel from interrupting but does not disable backplane interrupt sensing capability. Note that only one channel is disabled per DISABLE INTR [USE ch] command.

The advantage of the DISABLE INTR SYS command is that a backplane interrupt can still occur yet servicing of the interrupt can be held off until the mainframe is ready. For example, assume an accessory channel has been configured and enabled to interrupt and the HP 3852A is executing the following commands sent over the HP-IB:

```
100  OUTPUT 709;"REAL OHMRDGS (3)"
110  OUTPUT 709;"CONF OHMF"
120  OUTPUT 709;"RANGE 1K"
130  OUTPUT 709;"MEAS OHMF 100-103 USE 700 INTO OHMRDGS"
```

If an interrupt occurs while the mainframe is executing line 110, the interrupt will be handled once the command (CONF) completes. Assume the subroutine handling the interrupt is:

```
40  OUTPUT 709;"SUB SVCINTR"
50  OUTPUT 709;"REAL DCVRDGS (9)"
60  OUTPUT 709;"CONFMEAS DCV 400-409 USE 700 INTO DCVRDGS"
70  OUTPUT 709;"SUBEND"
```

When the subroutine completes and execution of commands sent over the HP-IB continues (line 120), an error would occur at line 120 since a 1K range is not allowed for the DCV function for which the voltmeter was configured during the subroutine. The MEAS command (line 130) changes the function back to OHMF; however, autorange rather than the 1K range is set. By disabling the mainframe's interrupt sensing capability then enabling it at a later point as in the sequence:

```
90   OUTPUT 709;"DISABLE INTR SYS"
100  OUTPUT 709;"REAL OHMRDGS (3)"
110  OUTPUT 709;"CONF OHMF"
120  OUTPUT 709;"RANGE 1K"
130  OUTPUT 709;"MEAS OHMF 100-103 USE 700 INTO OHMRDGS"
140  OUTPUT 709;"ENABLE INTR SYS"
```

servicing the interrupt that occurred during line 110 is held off until after line 140 executes. This enables the MEAS command to make the resistance measurements as configured.

www.valuetronics.com

The RST Command

Resetting (RST) an accessory clears and disables interrupts. The RST command has the syntax:

RST [*slot*]

If a slot is specified, only the accessory in that slot is reset. If a slot is not specified, the mainframe and all accessories are reset.

Using the RST command to clear and disable interrupts is not recommended since a reset of the mainframe erases all stored subroutines and resetting an accessory sets the accessory to its power-on state. The RST command in the example programs is intended to set the mainframe to a known operating state at the start of program execution, rather than the method of clearing and disabling the interrupt.

Mainframe Examples

The following examples show how the mainframe is configured to sense and handle interrupts from an alarm, limit exception, and from each plug-in accessory with interrupt capability.

Handling Alarm Interrupts

This program shows how the mainframe's alarm function is configured and enabled to interrupt. In the program, the alarm schedules 10 temperature measurements. The alarm is set to interrupt every five seconds. Each time an interrupt occurs, a subroutine is called and a temperature measurement is taken. After all 10 measurements are made, the readings are displayed.

```
10   REAL Temprdgs(0:9)                    !Declare controller array
20   OUTPUT 709;"RST"                      !Reset the HP 3852A
30   OUTPUT 709;"REAL TRDGS (9),T,P"       !Declare mainframe array and vars.
40   OUTPUT 709;"SUB TMEAS"                !Begin subroutine to handle intr.
50   OUTPUT 709;"CONFMEAS THM10K 0 USE 700 INTO TRDGS"
60                                         !Measure and store temp readings
70   OUTPUT 709;"INDEX? TRDGS INTO P"      !Read position of index pointer
80   OUTPUT 709;"IF P > 9 THEN"            !If index pointer indicates array
90   OUTPUT 709;"VREAD TRDGS"              !is full, read array and display
100  OUTPUT 709;"ELSE"                     !temps. Else, enable and set
110  OUTPUT 709;"T= T + 5"                 !alarm to interrupt in 5 seconds
120                                        !which calls the subroutine again
130  OUTPUT 709;"SET ALRM T"               !Set/enable alarm intr capability
140  OUTPUT 709;"ENABLE ALRM"              !disabled when intr was serviced
150  OUTPUT 709;"END IF"
160  OUTPUT 709;"SUBEND"                   !End of subroutine
170  OUTPUT 709;"ON ALRM CALL TMEAS"       !Call subroutine on alarm intr
180  OUTPUT 709;"SET TIME 0"               !Set mainframe real time clock
190  OUTPUT 709;"T=5"
200  OUTPUT 709;"SET ALRM T"               !Set alarm w/reference to clock
```


www.valuetronics.com

```
210   OUTPUT 709;"ENABLE ALRM"        !Enable alarm interrupt capability
220   ENTER 709;Temprdgs(*)           !Enter readings into controller
230   FOR I=0 TO 9                    !when mainframe array is full
240   PRINT Temprdgs(I)               !Display readings
250   NEXT I
260   END
```

A typical output based on this program is:

```
27.41113
28.99121
29.83203
27.91895
27.50977
27.32324
27.17285
27.04102
26.92969
26.81738
```

Notice in the program that the subroutine is downloaded first but is not executed until it is called when an interrupt occurs. Note also that the alarm interrupt capability is re-enabled again inside the subroutine. This is because alarm interrupts are disabled during servicing.

Handling Limit Interrupts

This program shows one example of how interrupts involving real-time limit testing are set up and enabled. The program uses the mainframe's alarm function (similar to the previous example) to schedule thermistor measurements every two seconds. The thermistors are measured continuously until a temperature limit is exceeded. Once a limit is exceeded, an interrupt occurs and the temperature on both channels is displayed.

www.valuetronics.com

```
10   REAL Chs(0:1);Temp(0:1)                    !Declare controller arrays
20   OUTPUT 709;"RST"                           !Reset the HP 3852A
30   OUTPUT 709;"OUTBUF ON"                     !Append data in the output buf
40   OUTPUT 709;"REAL MINTEMP(0),MAXTEMP(0),EXTEMP(1),TEMPRDGS(1),CH(1)"
50                                              !Declare mainframe arrays
60   OUTPUT 709;"INTEGER T"                     !Declare mf time variable
70   OUTPUT 709;"VWRITE MINTEMP 20"             !Assign minimum limit
80   OUTPUT 709;"VWRITE MAXTEMP 30"             !Assign maximum limit
90   OUTPUT 709;"SUB SCHEDMES"                  !Begin scheduling subroutine
100  OUTPUT 709;"INDEX CH 0"                    !Set ch logging array index to 0
110  OUTPUT 709;"CONFMEAS THM10K 0-1 USE 700 INTO TEMPRDGS(0)"
120                                             !Temp meas performed every 2s
130  OUTPUT 709;"T=T+2"                         !Increment time var by 2s
140  OUTPUT 709;"SET ALRM T"                    !Set alarm to occur in 2s
150  OUTPUT 709;"ENABLE ALRM"                   !Enable interrupt on alarm
160  OUTPUT 709;"SUBEND"                        !End of scheduling subroutine
170  OUTPUT 709;"SUB TEMPLIM"                   !Begin limit exception sub
180  OUTPUT 709;"DISABLE ALRM"                  !Halt meas after limit detected
190  OUTPUT 709;"VREAD CH"                      !Read array containing ch #'s
200  OUTPUT 709;"VREAD TEMPRDGS"                !Read temps from all channels
210  OUTPUT 709;"SUBEND"                        !End of limit exception sub
220  OUTPUT 709;"ON LMT CALL TEMPLIM"           !Call subroutine on limit intr
230  OUTPUT 709;"LMT MINTEMP,MAXTEMP, EXTEMP"
240                                             !Specify limit testing arrays
250  OUTPUT 709;"ENABLE LMT"                    !Enable limit testing
260  OUTPUT 709;"LOGCHAN CH"                    !Specify channel logging array
270  OUTPUT 709;"ENABLE LOGCHAN"                !Enable channel logging
280  OUTPUT 709;"SET TIME 0"                    !Set real time clock to 0s
290  OUTPUT 709;"T=2"                           !Increment time var by 2s
300  OUTPUT 709;"SET ALRM T"                    !Set alarm to occur in 2s
310  OUTPUT 709;"ENABLE ALRM"                   !Enable interrupt on alarm
320  OUTPUT 709;"ON ALRM CALL SCHEDMES"         !Call subroutine on alarm intr
330  ENTER 709;Chs(*)                           !Enter and display readings
340  ENTER 709;Temp(*)                          !on controller
350  PRINT "CH     TEMP"
360  PRINT
370  FOR I=0 TO 1
380    PRINT Chs(I),Temp(I)
390  NEXT I
400  END
```


www.valuetronics.com

An output based on this program is shown below. The temperature on channel 0 is the temperature which exceeded the limit.

```
CH    TEMP

0   30.27832
1   21.68164
```

Handling Backplane Interrupts

The following programs are examples of how backplane interrupts may be enabled and handled by the mainframe.

HP 44721A 16-Channel Digital Input

This program enables channel 3 of an HP 44721A 16-Channel Digital Input in mainframe slot 2 to interrupt when a low to high edge transition occurs. Refer to the HP 44721A manual for detailed information on configuring digital input interrupts.

```
10    OUTPUT 709;"RST"                    !Reset the HP 3852A and HP 44721A
20    OUTPUT 709;"USE 219"                !Use ch 3 in slot 2 (event intr)
30    OUTPUT 709;"SUB DATA"               !Define subroutine DATA
40    OUTPUT 709;"DISP 'INTR-CH 203'"
50                                        !Display msg when interrupt occurs
60    OUTPUT 709;"SUBEND"                 !End subroutine
70    OUTPUT 709;"ON INTR CALL DATA"
80                                        !Call subroutine on edge transition
90    OUTPUT 709;"EDGE LH"                !Detect a low to high transition
100   OUTPUT 709;"ENABLE INTR"            !Enable channel to interrupt
110   OUTPUT 709;"ENABLE INTR SYS"        !Sense the backplane interrupt
120   OUTPUT 709;"END"
```

Note that the sequence of commands above follow the procedure described previously. The subroutine to handle the interrupt is downloaded into memory and called when the interrupt occurs. The digital input is then configured to interrupt on the desired condition. Next, channel interrupt capability is enabled and the mainframe is enabled to sense the interrupt.

Recall that when a backplane interrupt is serviced, the channel's interrupt capability is disabled. For the digital input accessories, servicing also clears the interrupt.

www.valuetronics.com

HP 44715A 5-Channel Counter/Totalizer

This program enables channel 5 of the HP 44715A counter to interrupt after 10 counts. Refer to the HP 44715A manual for additional information on setting HP 44715A interrupt conditions.

```
10   OUTPUT 709;"RST"                    !Reset HP 3852A and HP 44715A
20   OUTPUT 709;"USE 305"                !Use channel 5 in slot 3
30   OUTPUT 709;"SUB DATA"               !Define subroutine DATA
40   OUTPUT 709;"DISP 'CTR OVERFLOW'"    !Display msg when intr occurs
50   OUTPUT 709;"SUBEND"                 !End subroutine
60   OUTPUT 709;"ON INTR CALL DATA"      !Call subroutine on ctr overflow
70   OUTPUT 709;"EDGE LH"                !Count positive-going edges
80   OUTPUT 709;"FUNC TOTAL"             !Set ch for TOTAL function
90   OUTPUT 709;"CNTSET -10"             !Interrupt after 10 edges
100  OUTPUT 709;"ENABLE INTR"            !Enable channel to interrupt
110  OUTPUT 709;"ENABLE INTR SYS"        !Sense the backplane interrupt
120  OUTPUT 709;"TRIG SGL"               !Trigger once
130  END
```

This program also follows the steps outlined for enabling an interrupt. The channel is disabled and the interrupt is cleared when serviced.

HP 44701A Integrating Voltmeter

This program enables the HP 44701A voltmeter to interrupt after a measurement is taken on its rear panel terminals.

```
10   OUTPUT 709;"RST"                    !Reset the HP 3852A and HP 44701A
20   OUTPUT 709;"USE 100"                !Use voltmeter in slot 1
30   OUTPUT 709;"SUB DATA"               !Define subroutine DATA
40   OUTPUT 709;"DISP 'DATA READY'"      !Message displayed when intr occurs
50   OUTPUT 709;"WAIT 3"                 !Display message for 3 seconds
60   OUTPUT 709;"CHREAD 100"             !Display measurement on mainframe
70   OUTPUT 709;"SUBEND"                 !End subroutine
80   OUTPUT 709;"ON INTR CALL DATA"      !Call sub when interrupt occurs
90   OUTPUT 709;"CONF DCV"               !Configure for DC volts
100  OUTPUT 709;"TERM EXT"               !Set ext. terminals as the vm input
110  OUTPUT 709;"TRG EXT"                !System trigger source is SYSTEM
120                                      !TRIGGER IN BNC
130  OUTPUT 709;"ENABLE INTR"            !Enable voltmeter to interrupt
140  OUTPUT 709;"ENABLE INTR SYS"        !Sense the backplane interrupt
150  OUTPUT 709;"TRIG SYS"               !Vm trigger source is system trig
160  END
```

Again, this program follows the steps outlined previously. The voltmeter's interrupt capability is disabled when the interrupt is serviced; however, the interrupt is not cleared until the data is read from the voltmeter (CHREAD).

www.valuetronics.com

The WAITFOR Command

The WAITFOR command can be used to suspend the operation of the HP 3852A while awaiting an interrupt. The WAITFOR command has the syntax:

WAITFOR *condition*

where the conditions are a signal on the EVENT IN BNC or pressing the EVENT key on the front panel (EVENT), an alarm from the mainframe clock (ALRM), an output pulse from the PACER OUT BNC (PACER), or an interrupt from an accessory channel (INTR). The following example shows how the WAITFOR command suspends voltmeter measurements pending an interrupt from a digital input accessory.

```
10   REAL RGS (0:4)                          !Declare controller array
20   OUTPUT 709;"RST"                        !Reset the HP 3852A and all access.
30   OUTPUT 709;"USE 116"                    !Use ch 0 of digital input in slot 1
40   OUTPUT 709;"EDGE LH"                    !Detect low to high transition
50   OUTPUT 709;"ENABLE INTR"                !Enable channel to interrupt
60   OUTPUT 709;"ENABLE INTR SYS"            !Sense the backplane interrupt
70   OUTPUT 709;"WAITFOR INTR"               !Suspend operation until interrupt
80   OUTPUT 709;"CONFMEAS DCV 0-4 USE 700"
90                                           !Measure DCV when interrupt occurs
100  ENTER 709;RGS(*)                        !Enter and display readings
110  PRINT USING "K,/";RGS(*)
120  END
```

A typical output based on this program is shown below:

```
.966852
5.05801
1.55567
1.579121
1.618728
```


www.valuetronics.com

Handling Interrupts with the Controller

This section explains how the mainframe is enabled to sense an interrupt and send a service request (SRQ) message to the controller so that it can respond. The section explains how the mainframe's Status Register is masked so that only selected interrupt conditions will generate an SRQ and how bits in the register are cleared so that the controller can respond to multiple interrupts.

The Status Register

The purpose of the HP 3852A Status Register is to allow interrupts from selected sources to generate a service request (SRQ) message which is sent to the controller so that the controller can handle the interrupt. The interrupt source(s) is selected by placing a "mask" over the Status Register bits that you do not want to generate a service request. When the service request mode is enabled and an interrupt occurs and sets a bit which is unmasked, an SRQ message is generated on the bit transition.

As stated at the beginning of this chapter, there are nine sources from which an interrupt can originate which are represented by bits in the Status Register. The source of the interrupt is recorded by the register when the corresponding bit changes from a binary 0 (clear) to a binary 1 (set).

The Status Register is only used when interrupts and exceptions are handled by the controller. When handled by the mainframe, the interrupt or exception is sensed prior to the time either sets a bit in the register.

Table 8-2 identifies the bits in the Status Register and the interrupt sources (and conditions) they represent. The table also lists the mnemonics and decimal equivalents used to unmask the bits.

www.valuetronics.com

Table 8-2. The HP 3852A Status Register

Bit	Mnemonic	Decimal	Interrupt Conditions
15-12	-	-	Not used.
11	ALRM	2048	Set TRUE (1) when alarm occurs. Execute STA? to clear bit.
10	LMT	1024	Set TRUE (1) when limit reached. Execute STA? to clear bit.
9	INTR	512	Set TRUE (1) when interrupt occurs on accessory channel. Bit not set unless ENABLE INTR SYS and ENABLE INTR [USE ch] are set. Execute STA? to clear bit.
8	-	-	Not used.
7	-	-	Not used.
6	-	64	Service request bit. Set TRUE (1) when any other unmasked bits in Status Register go from FALSE (0) to TRUE (1) and when RQS ON is set. Execute a Serial Poll (SPOLL) or STB? to clear the bit.
5	ERR	32	Set TRUE (1) when error (programming, configuration, etc.) occurs. Execute ERR? or ERRSTR? (max of four) to empty the error buffer and clear the bit.
4	RDY	16	Set TRUE (1) when mainframe input buffers (HP-IB and front panel) are empty and no command or subroutine is executing or being accepted (i.e., a partial command leaves RDY clear even though the buffer is empty). Bit is cleared when byte received or when another interrupt occurs.
3	LCL	8	Set TRUE (1) when the HP 3852A turned on or when the LOCAL key causes the mainframe to enter the local operating state. Execute STA? to clear bit.
2	FPS	4	Set TRUE (1) when SRQ command is executed. Execute STA? to clear bit.
1	PWR	2	Set TRUE (1) when mainframe loses power. Bit is set when mainframe power supply goes below +18V or when an HP 3853A which is connected to the HP 3852A is turned off. Bit cleared when power restored to mainframe (or extender).
0	DAV	1	Set TRUE (1) when data returned from a command is available in HP-IB output buffer. Bit is cleared by CLROUT command or when data is otherwise removed from the output buffer.

Unmasking the Status Register

At power-on or when the mainframe is cleared (CLR command), all bits in the Status Register are masked and the service request mode is enabled. Should an interrupt be enabled and occur while a bit is masked, the bit is set; however, no SRQ message is generated. For an interrupt to generate an SRQ message, the service request mode must be enabled and the bit which is set must be unmasked. The command used to enable the service request mode and unmask Status Register bits is the RQS command. The RQS command has the syntax:

RQS *mode* or *unmask*

www.valuetronics.com

The *mode* parameters are OFF and ON. When *mode* is ON, the service request mode is enabled. When enabled, bit 6 in the register (Table 8-2) is set in addition to the SRQ message that is generated when an unmasked bit is set. (Note that bit 6 is not a maskable bit.) If RQS OFF is set and an interrupt occurs which sets an unmasked bit, no SRQ message is generated nor is bit 6 set.

The *unmask* parameters are listed under the "Mnemonic" and "Decimal" columns in Table 8-2. To unmask a Status Register bit, either specify the mnemonic or decimal equivalent of the bit. For example, to unmask bit 9 which is set by a backplane (accessory) interrupt, the RQS command would be executed as:

```
OUTPUT 709;"RQS INTR"
```

or

```
OUTPUT 709;"RQS 512"
```

Several bits can be unmasked with a single RQS command by listing each mnemonic or entering the sum of the decimal equivalents as shown in the following program statements:

```
OUTPUT 709;"RQS INTR,ALRM"
```

or

```
OUTPUT 709;"RQS 2560"
```

(2560 = 2048 (ALRM) + 512 (INTR))

Reading the Status Register Mask

The bits in the Status Register that have been unmasked by the RQS command can be determined by the RQS? command. The RQS? command has the syntax:

RQS? [INTO *name*] or [*fmt*]

RQS? returns the sum of each unmasked bit's decimal equivalent. Also, if RQS ON is issued, 64 is included in the sum. In the following example, the RQS? command is executed after the bits unmasked previously:

```
10  OUTPUT 709;"RQS ON"
20  OUTPUT 709;"RQS INTR,ALRM"
30  OUTPUT 709;"RQS?"
40  ENTER 709;A
50  PRINT A
60  END
```


www.valuetronics.com

As this program executes, 2624 is returned (2048 + 512 + 64). 2048 is the decimal equivalent of the ALRM bit that was unmasked (line 20), 512 is the decimal equivalent of the INTR bit also unmasked, and 64 is included because RQS ON was issued (line 10). The **INTO** *name* and *fmt* parameters indicate the sum returned by RQS? can be stored in the mainframe or entered/displayed in a particular format.

Determining the Bits Set in the Register

The STA? command is used to determine the Status Register bits, either **masked** or **unmasked**, which are set. The syntax of the STA? command is:

STA? [INTO *name*] or [*fmt*]

STA? returns the decimal equivalent sum of each bit set. If STA? is executed while an SRQ message is being asserted, 64 is included in the sum. The **INTO** *name* and *fmt* parameters indicate that the sum returned can be stored in mainframe memory or entered/displayed in a particular data format.

Another command often used to determine the bits (masked or unmasked) set in the Status Register is the STB? command. The STB? command has the syntax:

STB? [INTO *name*] or [*fmt*]

The STB? command reads the mainframe's status byte and returns the decimal equivalent sum of each bit set. If STB? is executed while an SRQ message is being asserted, 64 is included in the sum. In the status byte, bit 7 represents the logical OR of the ALRM, LMT, and INTR interrupt sources. If an interrupt from any of these three sources occurs, bit 7 is set. Note that it is the status byte that is sent to the controller in response to a serial poll (SPOLL). Status byte bit definitions are shown in Table 8-3.

www.valuetronics.com

Table 8-3. Status Byte Bit Definitions

Bit/Mnemonic		Decimal	Description	Annunciator
0	DAV	1	Data Available	DAV
1	PWR	2	System Power Below Acceptable Level	—
2	FPS	4	Programmed Service Request (SRQ command)	SRQ
3	LCL	8	Power-on or LOCAL Key when in Remote	—
4	RDY	16	Ready to Execute Command	BUSY
5	ERR	32	Error	ERR
6	—	64	Service Request Bit	—
	INTR or		Accessory Channel Interrupting or	INTR
7	LMT or	128	Limit Condition Reached or	LMT
	ALRM		Alarm Condition Exists	ALRM

The **INTO** *name* and *fmt* parameters indicate the sum returned by STB? can also be stored or entered/displayed in a particular format.

Clearing the Status Register

When multiple interrupts from the same source are handled by the controller, the corresponding bit and the service request bit (bit 6) in the Status Register must be cleared each time the interrupt is serviced and handled. This enables the mainframe to detect, and the controller to handle, each "new" interrupt. Note that when backplane (accessory) interrupts are used, clearing the Status Register bit (INTR) and the service request bit is required in addition to clearing the interrupt at the accessory level.

The ALRM, LMT, INTR, LCL, and FPS bits in the Status Register (Table 8-2) are cleared by the STA? command which also returns the decimal equivalent sum of all set bits.

The service request bit (bit 6) is cleared by the STB? command or by a serial poll (SPOLL) issued by the controller.

The ERR bit set when an error message(s) is in the error buffer, is cleared when the ERR? or ERRSTR? command is executed until the buffer is empty.

The RDY bit which is set when the mainframe is not accepting or executing a command, is cleared when a command is received or is executing.

The PWR bit set when the mainframe or an extender loses power, is cleared when power is restored.

The DAV bit set when data is in the HP-IB output buffer, is cleared when the data is removed from the buffer.

www.valuetronics.com

Enabling the Interrupt

The controller can handle interrupts from any of the sources represented by the bits in the Status Register. The procedure for setting up the mainframe so that it sends a service request (SRQ) message to the controller when an interrupt occurs is basically the same regardless of the source.

The steps involved with setting up and enabling the mainframe are given here. The commands associated with each step correspond to the sequence they should occur in the program.

Step 1 (Controller)

The first step when handling an interrupt with the controller is to set up and enable interrupt-initiated branches within the controller. The examples at the end of this section show one method used by Hewlett-Packard Series 200 and Series 300 computers. If necessary, refer to the operating manual of your particular controller for additional information.

The following statements are the beginning of a program developed throughout this procedure which sets up a digital input accessory interrupt.

```
10  ON INTR 7 GOTO RESULTS
20  ENABLE INTR 7;2                !enables SRQ interrupt
```

Step 2 (Controller)

The next step is to ensure that the Status Register bit corresponding to the source of the interrupt is cleared. This prevents an SRQ message from being generated prematurely if the bit is already set when the interrupt is enabled. Refer to "Clearing the Status Register" for information on how each bit is cleared.

Since a backplane interrupt is set up in this procedure, the INTR bit must be cleared. INTR is cleared by the STA? command. Since it is not necessary to store or display the data returned by STA?, neither **INTO** *name* or *fmt* are specified. In certain situations, you may want to store the data returned by STA? in order to keep the mainframe's output buffer clear if measurement data is returned as part of the interrupt handling routine.

```
30  OUTPUT 709;"STA?"
```

Step 3 (Controller)

The next step is to enable the mainframe's service request capability and unmask the bit(s) in the Status Register that will cause a service request when set. The command used to do this is the RQS command. Recall that the command has the syntax:

RQS *mode* or *unmask*

where *mode* = ON enables service request capability (i.e. enables the service request bit (bit 6) to be set), and *unmask* is the mnemonic(s) or decimal equivalent of the bit(s) to be unmasked. Since the digital input

www.valuetronics.com

interrupt is a backplane interrupt, the INTR bit is unmasked in the program as follows:

```
40 OUTPUT 709;"RQS ON"
50 OUTPUT 709;"RQS INTR"
```

Step 4 (Controller)

Depending on the source of the interrupt, the next step is to set the alarm, set up limit testing, or configure the accessory channel to interrupt on a desired condition. When setting up interrupts from other sources, continue with Step 7. The commands used to set the alarm and set up limit testing are:

SET ALRM *time*

LMT *min max index_store*

To configure an accessory channel for a specific interrupt condition, refer to that accessory's configuration and programming manual. To continue the program, however, channel 3 of the digital input installed in mainframe slot 2 is configured to interrupt when the counter overflows after 10 counts. Thus:

```
60 OUTPUT 709;"USE 203"
70 OUTPUT 709;"EDGE LH"
80 OUTPUT 709;"CNTSET -10"
```

Step 5 (Controller)

Again, based on the source of the interrupt, the next step is to either enable the alarm to interrupt, initiate real-time limit testing, or enable the specific accessory channel to interrupt. The commands used to do this are:

ENABLE ALRM

ENABLE LMT

ENABLE INTR [USE *ch*]

Continuing the program, step 5 is to enable the specific accessory channel to interrupt. Thus,

```
90 OUTPUT 709;"ENABLE INTR"
```

Note that you can only enable one channel per ENABLE INTR command. Also, since the USE command was executed in line 60, the **[USE** *ch*] parameter does not have to be specified.

www.valuetronics.com

Step 6 (Controller)

Step 6 is to enable the mainframe to sense an interrupt from an accessory channel. The command used to do this is:

ENABLE INTR SYS

Note that Step 6 is required for backplane (accessory) interrupts only. Since the program developed throughout this procedure is for a backplane interrupt, Step 6 is executed as:

```
100  OUTPUT 709;"ENABLE INTR SYS"
```

Step 7 (Controller)

The final step in setting up an interrupt handled by the controller is to define the function the controller will perform when the interrupt is received and then clear the mainframe's service request bit (bit 6) set when the interrupt occurred. The service request bit is cleared by a serial poll (SPOLL) issued by the controller or by the STB? command. Recall that the STB? command has the syntax:

STB? [INTO *name*] or [*fmt*]

Note also that STB? returns the decimal equivalent sum of all bits set in the register. Therefore if STB? is used, you may want to store the sum returned in a mainframe variable if data is to be returned when the interrupt is handled. Since in this program only a message is displayed by the controller, neither **[INTO** *name*] or [*fmt*] is specified.

```
110  GOTO 110
120  Results: !
130  PRINT "Overflow on ch 203"
140  OUTPUT 709;"STB?"
150  END
```

A complete program listing is shown below:

```
10   ON INTR 7 GOTO Results
20   ENABLE INTR 7;2
30   OUTPUT 709;"STA?"
40   OUTPUT 709;"RQS"
50   OUTPUT 709;"RQS INTR"
60   OUTPUT 709;"USE 203"
70   OUTPUT 709;"EDGE LH"
80   OUTPUT 709;"CNTSET-10"
90   OUTPUT 709;"ENABLE INTR"
100  OUTPUT 709;"ENABLE INTR SYS"
110  GOTO 110
120  Results: !
130  PRINT "Overflow on ch 203"
140  OUTPUT 709;"STB?"
150  END
```

The steps for handling interrupts with the controller are summarized in Figure 8-2. The command flow for interrupts handled by the mainframe and by the controller is contrasted in Figure 8-3.

www.valuetronics.com

Controller Examples

The following examples show how the mainframe is enabled to sense an interrupt and send a service request message to the controller so that it can respond. The examples include an interrupt from the SRQ key on the mainframe front panel and from selected plug in accessories with interrupt capability.

Front Panel SRQ Interrupt

This program shows one method of using the front panel SRQ key to generate an SRQ message which is sent to the controller. The program loops continuously until the SRQ key is pressed. When pressed, an interrupt occurs which is serviced by the mainframe and then handled by the controller.

```
10   OUTPUT 709;"RST"                      !Reset the HP 3852A
20   ON INTR 7 GOTO 70                     !Go to 70 when SRQ key pressed
30   ENABLE INTR 7;2                       !Enable controller intr path
40   OUTPUT 709;"RQS ON"                   !Enable service request mode
50   OUTPUT 709;"RQS FPS"                  !Unmask bit 2
60   GOTO 60                               !Loop until SRQ key pressed
70   PRINT "Front panel SRQ interrupt"     !Display interrupt message
80   OUTPUT 709;"STA?"                     !Read Status Reg bits set
90                                         !(also clears bit 2)
100  ENTER 709;A                           !Enter sum of values of bits set
110  PRINT "Status Register bits set = ";A !Display sum of values
120  B=SPOLL(709)                          !Clear bit 6 in Status Reg
130  END
```

When the interrupt occurs, the controller handles the interrupt by displaying " Front panel SRQ interrupt " and then sending the STA? command. A sum of 68 is returned indicating that bit 2 (FPS) [weight = 4] and bit 6 (service request - set when bit 2 was set) [weight = 64] are set.

www.valuetronics.com

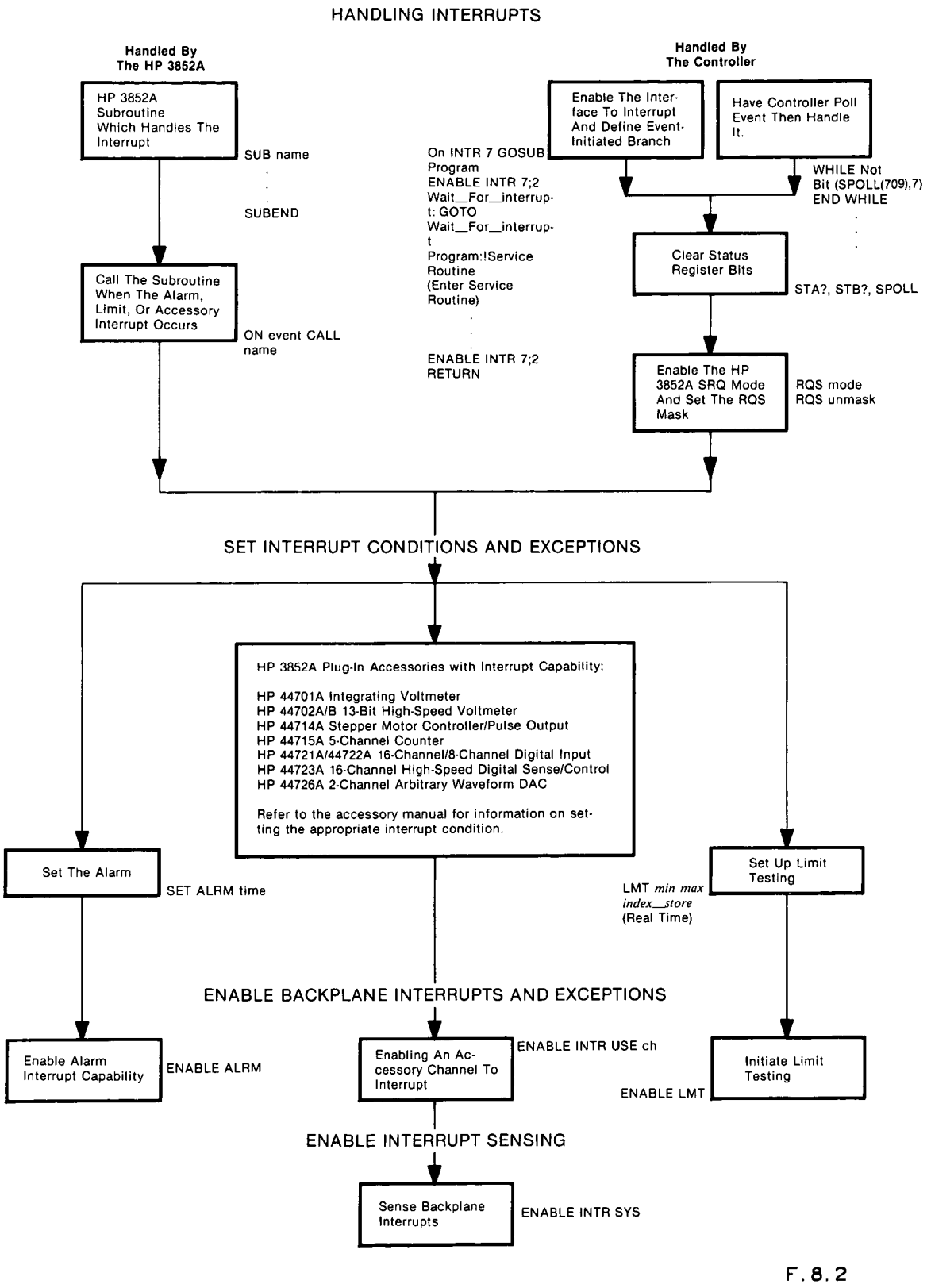


Figure 8-2. Handling Interrupts

www.valuetronics.com

Backplane Interrupts (Controller)

The following programs are examples of how backplane interrupts are handled by the controller.

HP 44721A 16-Channel Digital Input

This program enables channel 3 of an HP 44721A 16-channel digital input in slot 2 of the mainframe to interrupt when the channel counter overflows (from -1 to 0). The digital input channel is preset to overflow after 10 counts. (Refer to the HP 44721A manual for more information on configuring digital input interrupts.)

```
10   ON INTR 7 GOTO Results          !Define event-initiated branch
20   ENABLE INTR 7;2                 !Enable interrupt capability
30   OUTPUT 709;"RST"                !Reset the HP 3852A and HP 44721A
40   OUTPUT 709;"STA?"               !Clear INTR bit in Status Register
50   OUTPUT 709;"RQS ON"             !Set RQS mode ON
60   OUTPUT 709;"RQS INTR"           !Unmask INTR bit in Status Register
70   OUTPUT 709;"USE 203"            !Use channel 3 in slot 2
80   OUTPUT 709;"EDGE LH"            !Count positive edges
90   OUTPUT 709;"CNTSET -10"         !Overflow after 10 pos edges
100  OUTPUT 709;"ENABLE INTR"        !Enable channel to interrupt
110  OUTPUT 709;"ENABLE INTR SYS"    !Sense the backplane interrupt
120  GOTO 120                        !Loop until intr occurs
130  Results: !                      !Line label for branch
140  PRINT "Overflow on ch 203"      !Display message
150  A=SPOLL(709)                    !Clear HP 3852 service request bit
160  END
```

The sequence of commands in this program follow the procedure described previously. The interrupt-initiated branch is set up and enabled. The STA? command is executed to ensure that the INTR bit is cleared. Next, the RQS mode is enabled and the INTR bit is umasked. The digital input is then configured to interrupt on the desired condition and channel interrupt capability is enabled. The mainframe is then enabled to sense the interrupt and the handling routine within the controller is defined. Note that the service request bit (bit 6) is cleared by a serial poll (SPOLL) rather than with the STB? command.

Note that digital input interrupts are cleared when serviced. Since STA? and SPOLL are included in the program, the interrupt is also cleared at the Status Register level.

www.valuetronics.com

HP 44715A 5-Channel Counter/Totalizer

This program enables channel 3 of an HP 44715A counter in slot 2 of the mainframe to interrupt the controller when the counter channel overflows (from -1 to 0). The counter is preset to overflow after 10 counts. Refer to the HP 44715A manual for additional information on setting HP 44715A interrupt conditions.

```
10   ON INTR 7 GOTO Results                 !Define event initiated branch
20   ENABLE INTR 7;2                        !Enable controller intr capability
30   OUTPUT 709;"RST"                       !Reset HP 3852A and HP 44715A
40   OUTPUT 709;"STA?"                      !Clear INTR bit in Status Register
50   OUTPUT 709;"RQS ON"                    !Set RQS mode ON
60   OUTPUT 709;"RQS INTR"                  !Unmask INTR bit in Status Register
70   OUTPUT 709;"USE 203"                   !Use channel 3 in slot 2
80   OUTPUT 709;"EDGE LH"                   !Count positive edges
90   OUTPUT 709;"FUNC TOTAL"                !Set channel for TOTAL function
100  OUTPUT 709;"CNTSET -10"                !Overflow after 10 pos edges
110  OUTPUT 709;"ENABLE INTR"               !Enable channel to interrupt
120  OUTPUT 709;"ENABLE INTR SYS"           !Sense the backplane interrupt
130  OUTPUT 709;"TRIG SGL"                  !Trigger once
140  GOTO 140                               !Loop until interrupt occurs
150  Results: !                             !Line label for branch
160  PRINT "Overflow on ch 203"             !Display message
170  A=SPOLL(709)                           !Clear HP 3852 service request bit
180  END
```

This program also follows the steps covered for setting up an interrupt and handling the interrupt with the controller. Because of the counter function specified, the interrupt is cleared at the accessory level when serviced. Thus the interrupt is cleared at both levels as required.

HP 44702A/B High-Speed Voltmeter

This program enables an HP 44702A voltmeter in slot 5 of the mainframe to interrupt when a scan of the specified channels completes. The controller handles the interrupt by displaying "SCAN IS COMPLETE" and then storing the readings.

```
10   ON INTR 7 GOSUB Results                !Call sub on interrupt
20   ENABLE INTR 7;2                        !Enable controller intr capability
30   DIM Rdgs(0:9)                          !Dimension controller array
40   OUTPUT 709;"RST"                       !Reset the HP 3852A and HP 44702A/B
50   OUTPUT 709;"RQS ON"                    !Set RQS mode on
60   OUTPUT 709;"RQS INTR"                  !Unmask the INTR bit (bit 9)
70   OUTPUT 709;"STA?"                      !Ensure Status Register bit 9 is cleared
80   OUTPUT 709;"CLROUT"                    !Clear the output buffer
90   OUTPUT 709;"USE 500"                   !Use the voltmeter in slot 5
100  OUTPUT 709;"SCANMODE ON"               !Turn scanner mode on
110  OUTPUT 709;"CONF DCV"                  !Configure for DC voltage measurements
120  OUTPUT 709;"CLWRITE SENSE 400-409"
```


www.valuetronics.com

```
130                                     !Specify ch list and ribbon cable conn.
140   OUTPUT 709;"RDGSMODE END"         !Interrupt when scan completes
150   OUTPUT 709;"ENABLE INTR"          !Enable voltmeter to interrupt
160   OUTPUT 709;"ENABLE INTR SYS"      !Enable mainframe to sense interrupt
170   OUTPUT 709;"SCTRIG EXT0"          !Externally trigger voltmeter
180   GOTO 180                          !Loop until interrupt occurs
190   Results: !                        !Start controller subroutine
200   PRINT "SCAN IS COMPLETE"          !Display message
210   A=SPOLL(709)                      !Clear HP 3852 service request bit
220   OUTPUT 709;"XRDGS 500"            !Transfer readings to output buffer
230   ENTER 709;Rdgs(*)                 !Enter readings into controller
240   END
```

Again, this program follows the steps outlined previously for setting up an interrupt that is handled by the controller. Because of the parameter specified for the RDGSMODE command (END), the interrupt is cleared at the voltmeter level when serviced. By including STA? and issuing a serial poll (SPOLL), the interrupt is also cleared at the Status Register level.

www.valuetronics.com

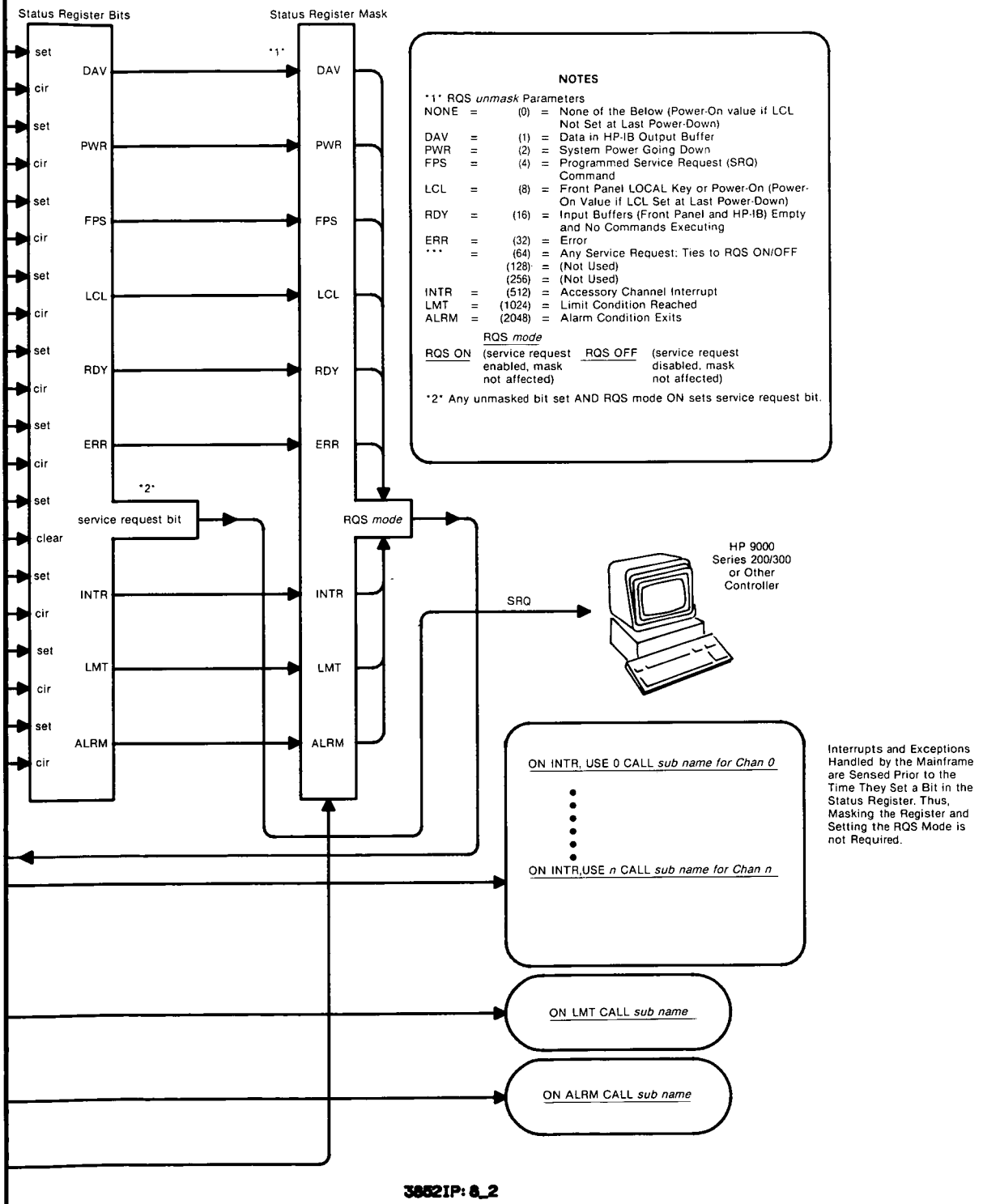


Figure 8-3. Interrupt Command Flow

www.valuetronics.com

Contents

www.valuetronics.com

Chapter 9

Using Subroutines

Introduction

The HP 3852A has the capability to store and execute (on command) BASIC language subroutines. These subroutines can either be downloaded into the HP 3852A mainframe memory from a remote controller (such as one of the HP Series 200/300 computers), or in many cases they may be entered from the front panel keyboard.

The purpose of this chapter is to acquaint you with the various commands that are specifically intended to be used with subroutines. Other commands that may be used either in or out of subroutines are also discussed. The chapter is divided into four major sections which are identified below:

- **Introduction** provides an overview of subroutines, what they are, and how they may be used in the HP 3852A.

- **Subroutine Commands** lists the various kinds of commands that can and cannot be used inside subroutines.

- **Guidelines For Using Subroutines** discusses rules and hints for using HP 3852A subroutines that should be followed to avoid errors and delays in programming.

- **Subroutine Program Examples** provides several examples of BASIC language programs that down-load subroutines into an HP 3852A mainframe, and then execute those subroutines upon later command.

Subroutine Definition

What is a Subroutine?

An HP 3852A subroutine is a series of commands beginning with a **SUB** *name* command and ending with a **SUBEND** command. The SUB name command assigns a name to the subroutine which can be called later for immediate execution by the **CALL** *name* [*number*] command, or when an interrupt occurs, it can be called with the **ON** *event* **CALL** *name* command.

www.valuetronics.com

Subroutines, downloaded into the HP 3852A prior to making measurements and then called later with a single command from the controller, allow the controller to perform other tasks while the HP 3852A is busy taking the desired measurements on its own. Also, commands in a subroutine are executed faster by the HP 3852A than those same commands received over the HP-IB because of the way the subroutine commands are stored internally.

What commands are allowed inside a Subroutine?

Almost any HP 3852A command may be stored and executed inside a subroutine. Exceptions include certain commands that scratch memory, and subroutine execution commands, all of which are listed under the heading "Subroutine Commands (Excluded Commands)."

How Long Can a Subroutine Be?

When subroutines are downloaded into mainframe memory, the mainframe reserves 8400 bytes of memory for storing each subroutine. If a subroutine that is downloaded does not use the full 8400 bytes, the remaining memory is released and may be used as part of the 8400 byte limit for the next subroutine, or it can be used when declaring arrays and variables. If a subroutine exceeds the 8400 byte limit while it is downloading, ERROR 9: SUB CODE TOO LONG is generated and that portion of the subroutine previously stored is erased. Note that each subroutine is allocated 8400 bytes regardless of the amount of mainframe memory available (including 256k, 1M, 2M, or 4Mbytes of extended memory).

If 8400 bytes of memory are not available when the subroutine is downloaded, the mainframe will allocate as much memory as possible down to a minimum of 800 bytes. If 800 contiguous bytes are not available, the mainframe will display ERROR 1: OUT OF MEMORY when you first attempt to download the subroutine. If the subroutine is less than 8400 bytes and the mainframe does have more than 800 bytes available, the subroutine will begin to download and then display ERROR 9 if the length of the subroutine exceeds the memory available.

There is no simple method for determining the number of bytes in a subroutine prior to, or while it is downloading. However, the CAT command discussed later in the chapter returns the number of bytes in each subroutine after it is downloaded. The DELSUB command also discussed later, erases the commands in the subroutine but not the subroutine name. Thus, downloading a portion of the subroutine, reading the length with the CAT command, using DELSUB to delete the commands, then downloading

www.valuetronics.com

a larger portion of the subroutine (i.e. moving the SUBEND statement) and repeating the CAT/DELSUB sequence is one method of determining the length of a subroutine.

Writing and Loading Subroutines

The subroutine program examples in this chapter are samples only, but they do actually work. They illustrate simple operations of the HP 3852A that may be copied and used in more complex mainline programs of your own design. BASIC language was chosen for the controller program examples because it is the most commonly used computer language for data acquisition applications and because it is easily converted to most other computer languages. Since there are many variations of BASIC, refer to the BASIC language user's manual that applies to your computer for the proper command syntax to use in your programs.

Subroutines, variables, and arrays are erased from memory when power is removed and when certain commands are executed. Therefore, we recommend that you write all of your non-trivial subroutines inside of your controller, rather than entering them manually from the front panel, and then store them on disc or tape for later use and for easy revision.

There are three conditions which erase all existing subroutines, variables, and arrays from mainframe memory:

- Removing power from the HP 3852A mainframe.
- Executing a system reset command (**RST** or **RST HARD**).
- Executing a **SCRATCH** memory command.

Accidently or deliberately causing one of these conditions to occur would necessitate reloading your subroutines into the HP 3852A. If they were originally entered from the front panel and not backed up on disc or tape, you would have to reload them manually, one command at a time.

Also, down-loading the subroutines from your remote controller means they are available for easy editing inside the controller. They can then be reloaded quickly into the HP 3852A mainframe memory, something that is not possible from the front panel. Since there is no edit capability in the HP 3852A, to edit a subroutine from the front panel, you would have to reload the entire subroutine manually, with the desired corrections made beforehand.

www.valuetronics.com

Subroutine Commands

In this section, all of the HP 3852A commands are listed or discussed, either individually or as groups. Special attention is paid to the way each command or group of commands relates to subroutines.

The actual commands associated with subroutines are shown in Table 9-1. These commands fall into three general categories:

- Internal commands ...(Commands allowed only *inside* subroutines.)
- Optional commands ...(Commands allowed inside *or* outside subroutines.)
- Excluded commands ...(Commands allowed only *outside* subroutines.)

All of the HP 3852A commands (except ADDR) can be executed over the HP-IB. Most of them (including ADDR) can also be executed from the front panel. The exceptions include all commands that use the =, *, /, <, >, or quotes ('') symbols.

Table 9-1. Subroutine Related Commands

Command Type	Internal	Optional	Excluded
Definition/Deletion Commands *			
SUB	X	—	—
SUBEND	X	—	—
DELSUB	—	—	X
SCRATCH	—	—	X
CAT	—	X	—
Internal Only Commands **			
FOR...NEXT	X	—	—
IF...END IF	X	—	—
WHILE...END WHILE	X	—	—
PAUSE	X	—	—
Execution Commands ***			
CALL	—	X	—
STEP	—	—	X
CONT	—	—	X
ON...CALL	—	X	—
OFF	—	X	—
All other Commands ****			
	—	X	—

* Definition/deletion commands identify the beginning and end of subroutines, store and delete subroutines from memory, and list the subroutines currently in memory. Note that only one SUB and one SUBEND command are allowed per subroutine.

** Internal commands must be executed from **inside** a subroutine. Optional commands may either be executed from inside a subroutine, or they may be executed directly, outside of HP 3852A subroutines. Excluded commands are only allowed **outside** of subroutines.

*** Execution commands control the execution of subroutines.

**** All HP 3852A commands not specifically excluded are allowed in subroutines.

www.valuetronics.com

Command Types

Subroutine related commands are *only* used with subroutines. There are three types of subroutine commands: definition/deletion, internal, and execution. Definition/deletion commands deal with the storage, viewing, and deletion of subroutines from mainframe memory. Execution commands control execution of subroutines from inside or outside a subroutine, and internal commands must be executed from inside a subroutine.

Definition/ Deletion Commands

Subroutine definition/deletion commands identify the beginning and end of subroutines, store and delete subroutines from memory, and list the subroutines presently stored in mainframe memory.

The syntax formats for the subroutine definition/deletion commands are as follows:

SUB *name*

.
.
.

SUBEND

DELSUB *name*

SCRATCH

CAT

Note that the only parameter required in these commands is *name*, which is the name of the subroutine. The name must be a string less than or equal to eight characters in length. It cannot begin with a digit and must contain only letters, digits, "_", and "?". Also, the name cannot be the same as any keyword for an HP 3852A command, or any variable or array name.

Every HP 3852A subroutine must contain the two commands: SUB name and SUBEND. The SUB name command must be the first line of all HP 3852A subroutines. It identifies where the subroutine begins and assigns the control name to the subroutine. It also begins the storing of the subroutine into mainframe memory.

The SUBEND command must be the last line of all subroutines. It identifies where the subroutine ends and also terminates the entry of the subroutine. All commands listed between the SUB name and the SUBEND command are executed, in order, every time the subroutine is called for execution.

Only one SUB name and one SUBEND command are allowed in any one subroutine. Additional SUB name or SUBEND commands generate errors.

www.valuetronics.com

The DELSUB name command deletes *only* the commands for the named subroutine from mainframe memory. It does *not* delete the subroutine name itself from the catalog of subroutine, variable, and array names.

The SCRATCH command deletes (scratches) *all* subroutines, variables, and arrays from mainframe memory. It also deletes all name definitions from the catalog.

The CAT command (catalog) is used to view a list of all the names of the subroutines, variables, and arrays that are presently stored or declared in mainframe memory. CAT also displays the number of bytes required to store each subroutine, and the number of elements dimensioned for each array. If a subroutine has been defined, and then deleted using DELSUB, it will be listed with a size of zero.

Refer to the heading "Subroutine Program Examples" for a sample program using the CAT command. Program Example #5 reads and displays a catalog list of subroutines, variables, and arrays stored in mainframe memory.

Internal Commands

Subroutine internal commands must be executed from inside a subroutine. If the HP 3852A receives one of these commands for execution and it was not preceded by a SUB name command, an error is generated.

The following commands may only be executed from inside a subroutine:

FOR...NEXT

IF...END IF

WHILE...END WHILE

PAUSE [*target*]

The first three of these commands are standard BASIC constructs. For details on their usage, refer to the discussions for each in the HP 3852A Command Reference Manual. Program examples for each of these are shown later in this chapter.

The PAUSE command is used to pause the subroutine.
The subroutine can then be continued using the STEP or CONT commands. PAUSE may not reside inside a "nested" subroutine nor inside a subroutine that is called where *number* in the CALL command is more than one. When *target* is specified, the run task subroutine is paused. The *target* parameter is used when the HP 3852A is in the multitasking mode. Multitasking is available with firmware revision 3.0 or greater.

www.valuetronics.com

Execution Commands

Subroutine execution commands control the execution of a subroutine. There are five execution commands:

CALL *name* [*number*]

STEP [*name*]

CONT [*target*]

ON *event* **CALL** *name*

OFF *event*

The CALL name [number] command activates the named subroutine. When the HP 3852A receives the CALL name [number] command, it searches the mainframe memory for the named subroutine (*name*). That entire subroutine is then executed one time (default value) unless the optional parameter *number* is specified, then it is executed *number* times. The CALL name [number] command may also be used inside a subroutine to call another subroutine. This provides the expanded capability of "nested" subroutines.

The STEP name command also activates the named subroutine. When the optional parameter *name* is used with STEP, the named subroutine is set up for execution and then paused automatically. Execution begins with the first command after the SUB name command.

Once a subroutine has been halted with a PAUSE command or a STEP name command, it can be continued with either STEP or CONT. STEP causes only the next command in the subroutine to be executed and then the subroutine pauses again. This is repeated as often as STEP is executed until the SUBEND command is reached.

CONT lets the subroutine continue running to completion, starting from the next command after the PAUSE command. The STEP and CONT commands may not reside inside subroutines.

If a subroutine called from the HP-IB is paused, then CONTinued from the front panel, the destination of any data returned is the display, if the data is not being stored internally. This means that data previously returned to the output buffer will now be returned to the display.

If no subroutine is currently paused when STEP or CONT are executed, an error is generated. If *target* is specified, the run task subroutine paused by the PAUSE *target* command is continued. The *target* parameter is used in the multitasking mode which requires firmware revision 3.0 or greater.

In the ON event CALL name command, *name* is the name of a subroutine that is called when a specific *event* occurs. For both the ON and the OFF conditions, *event* can be an interrupt (INTR) from a backplane (accessory) channel, a limit condition that has been exceeded (LMT), or an alarm function (ALRM) which occurred. The OFF event command cancels the ON event CALL name command.

www.valuetronics.com

For more information on calling a subroutine when an interrupt or exeception occurs, refer to Chapter 8 - Using Interrupts.

Optional Commands

Subroutine optional commands may be executed from inside a subroutine or may be executed from a controller program that contains no subroutines at all.

All HP 3852A commands except those specifically excluded under the heading "Excluded", may be located inside of a subroutine.

Three subroutine execution commands which may optionally be executed from inside a subroutine are:

CALL *name* [*number*]

ON *event* **CALL** *name*

OFF *event*

A "nested subroutine" capability may be obtained by inserting a CALL name [number] command inside a subroutine. In this way one subroutine may call a second (nested) subroutine for execution before the first subroutine is finished executing. When the second subroutine has finished executing, the first subroutine continues with the next command following the imbedded CALL name [number] command.

There are three requirements for using nested subroutines. First, the subroutine called from within another subroutine must be stored in mainframe memory before the subroutine doing the calling gets stored. This is because the HP 3852A checks the syntax of each command in a subroutine as it is being stored. When it encounters an imbedded CALL name [number] command, it checks to see if a subroutine by that name exists in memory. If not, it generates an error.

Second, a PAUSE command may not reside inside a nested subroutine since the STEP and CONT commands can not tell which subroutine to resume executing.

Third, subroutines may not be nested more than 10 levels deep.

It is also permissible to use the ON event CALL name command inside a subroutine as well as outside a subroutine. However, regardless of whether it is used internally or externally, if the specified *event* occurs while a subroutine is executing, the subroutine named by this command is not called until the first subroutine is completely through executing.

This is not the same as the CALL name [number] command which, if encountered in a subroutine, immediately calls the named subroutine. In this respect the subroutine named in the ON event CALL name command is not a true imbedded (or nested) subroutine.

www.valuetronics.com

Excluded Commands

Subroutine excluded commands are never allowed inside subroutines. The following commands are excluded commands:

CONT

STEP

DELSUB *name*

SCRATCH

Guidelines For Using Subroutines

Using subroutines in the HP 3852A is really very easy. Just keep in mind the following information, obey the indicated rules, and you should have no problems.

Storage and Display

1. All non-trivial subroutines should be stored in the HP 3852A mainframe memory via HP-IB, with a copy of the subroutine being maintained in your controller mass storage. That way, the subroutine can be reloaded quickly if the HP 3852A is powered down or is reset. Either of these two actions erases all subroutines and variables from mainframe memory.

This also provides a rapid means of editing the subroutine, using the system controller, and then re-storing the modified subroutine back into mainframe memory. You must use the DELSUB name command to delete the existing subroutine commands from memory before you re-store the modified version.

2. Subroutines entered from the front panel keyboard are treated the same as subroutines entered over the HP-IB.

3. All subroutines, variables, and arrays are erased from mainframe memory whenever a power failure occurs, or whenever a SCRATCH, RST, or RST HARD command is executed.

4. To view the execution of a subroutine on the front panel display, it is necessary to slow down the commands so they can be seen one at a time. If the **FASTDISP OFF** command is sent before the CALL name [number] command, the subroutine commands are executed slow enough to be seen. You can also use the STEP command to control execution speed.

www.valuetronics.com

The **FASTDISP OFF** command may also be used when you execute the CAT command from the front panel and want to view the subroutines, variables, and arrays on the front panel display.

5. To send message strings to the HP 3852A front panel display, use the following display statement construction:

```
70  OUTPUT 709;"DISP 'I=10'"
```

Data Management

6. All variables and arrays must be declared first, before they can be used either inside or outside of a subroutine.

7. All variables and arrays are global. If you declare/change a variable or an array inside a subroutine, it is also declared/changed outside the subroutine (and vice-versa).

8. It is permissible to redeclare an existing array, but you cannot change the storage type of an array or variable. For example, you can first execute **REAL K(4)**, and then later execute **REAL K(9)**, but you cannot later execute **INTEGER K(4)**.

Also, you cannot delete an array or variable from the internal catalog once they are declared, unless you execute a **SCRATCH, RST,** or **RST HARD** command to delete *all* variables, arrays, and subroutines. You can, however, use the **DELVAR** *name* command to dimension an array to zero.

9. The following mathematical expressions are allowed everywhere a number is used, including within subroutine commands:

+, -, *, /, ^, PI	Add, Subtract, Multiply, Divide, Exponentiation
( )	Parentheses
SIN, COS, ATN	Sine, Cosine, Arctangent ...in Radians
ABS, EXP, FRACT	Absolute value, Exponentiation, Fractional Part
INT, LGT, LOG	Largest integer, Log Base 10, Natural Log
SGN, SQR	Sign of Argument, Square Root
BINAND, BINCMP	Binary And, Binary Complement
BINEOR, BINIOR	Binary Exclusive Or, Binary Inclusive Or
BIT	Value, 0 or 1, of the specified bit
ROTATE, SHIFT	Value obtained by rotating the 16-bit binary expression the number of bit positions specified (wraparound). Value obtained by shifting the 16-bit binary expression the number of bit positions specified (no wraparound).
AND, OR	Logical And, Logical Inclusive Or

The following relational operators may be used in the IF and WHILE constructs:

```
=, ≤, ≥, <>, <, >
```


www.valuetronics.com

Execution and Exiting

10. The following BASIC language constructs *must* be located inside subroutines:

IF...END IF

FOR...NEXT

WHILE...END WHILE

The maximum number of **FOR...NEXT** loops that may be nested is 10.

11. The following commands may *not* be used inside a subroutine: DELSUB name, SCRATCH, STEP [name], CONT, a second SUB name, and a second SUBEND.

12. A subroutine must be paused or in the STEP mode to use CONT or STEP.

13. A PAUSE command cannot be located inside a nested subroutine or inside a subroutine that is called to execute more than one time using the CALL name [number] command.

14. In the power-on mode, a subroutine, once called, runs to completion before the HP 3852A executes another command outside of the subroutine. The subroutine identified by the CALL name [number] command is considered by the HP 3852A mainframe controller to be a single command that must be executed completely before another command can be brought in for execution. Therefore, the only ways to exit a subroutine early are (1) use the **CLR** or system **RST** commands inside the subroutine, (2) press the CLEAR button on the front panel, or (3) send the HP-IB Device Clear command over the HP-IB.

15. If a sytem RST command is executed from inside a subroutine, the rest of the subroutine never gets executed because RST erases the subroutine from memory. RST is a drastic way to exit a subroutine, but there are some real applications. For example, if you had a subroutine that was monitoring any circuit or digital input for safety reasons and a dangerous situation occurred, a system reset command could be used to immediately open all channels on all accessory cards. If only a few channels needed to be opened, the RST *slot* command could be used.

16. If a CLR command is executed from inside a subroutine, the subroutine stops executing but it is not erased from memory. CLR provides a way to exit a subroutine early or conditionally. Refer to the CLR command in the Command Reference Manual for other things that are affected when the CLR command is used.

17. To use the ON event CALL name command with a backplane (accessory) interrupt (INTR) as the event which calls the subroutine, the interrupt must first be set up and enabled. Refer to Chapter 8 - Using Interrupts for information on using interrupts to call subroutines.

www.valuetronics.com

18. If an error occurs anywhere inside of a subroutine, the subroutine is aborted. If a nested subroutine is aborted, the calling subroutine continues.

Subroutine Program Examples

This section contains eight program examples that illustrate various ways to use subroutine commands. The examples start simply and build in complexity. Each example is explained and annotated fully to assist you in understanding how subroutine commands may be used in the HP 3852A.

NOTE

These program examples are provided to illustrate the proper way to program subroutine commands. They are not intended as application examples.

Program Example #1

This example shows the format for a simple subroutine. When the program runs, the HP 3852A beeps once, waits 2 seconds, and beeps again.

```
10   OUTPUT 709; "SUB BEEPER"        ! Name of the subroutine is BEEPER.
20   OUTPUT 709; "   BEEP"           ! Beep once.
30   OUTPUT 709; "   WAIT 2"         ! Wait 2 seconds
40   OUTPUT 709; "   BEEP"           ! Beep once more.
50   OUTPUT 709; "SUBEND"            ! End of the BEEPER subroutine.
       .
       .
       .
140  OUTPUT 709; "CALL BEEPER"       ! Call BEEPER for immediate execution.
```


www.valuetronics.com

Program Example #2

This example uses the front panel display area to list the powers of two that are less than 1000. It also shows the use of the WHILE...END WHILE commands. When the program runs, the HP 3852A front panel display counts upward from 2 to 512 in powers of 2 (2,4,8,16,32,64,128, 256,512). It will then display "DONE".

```
10   OUTPUT 709; "INTEGER POWER"            ! Declare the variable POWER.
20   OUTPUT 709; "SUB DSP_PWRS"             ! Name of the subroutine is DSP_PWRS.
30   OUTPUT 709; "  POWER = 1"              ! Initialize POWER to 1.
40   OUTPUT 709; "  WHILE POWER < 1000"     ! While condition is true, double POWER.
50   OUTPUT 709; "  POWER = 2 * POWER"      ! Double POWER.
60   OUTPUT 709; "  WAIT 1"                 ! Wait 1 second.
70   OUTPUT 709; "  DISP POWER"             ! Display POWER on the front panel display.
80   OUTPUT 709; "  END WHILE"              ! End the power update.
90   OUTPUT 709; "  IF POWER > 1000 THEN"   ! If condition is true, display message.
100  OUTPUT 709; "  DISP 'DONE'"            ! Display the message.
110  OUTPUT 709; "  END IF"                 ! End display condition.
120  OUTPUT 709; "SUBEND"                   ! End of the subroutine DSP_PWRS.
       .
       .
       .
210  OUTPUT 709; "CALL DSP_PWRS"            ! Execute DSP_PWRS.
```

Program Example #3

This example illustrates a subroutine FOR/NEXT loop. When the program runs, the HP 3852A displays decreasing numbers from 10 to 1, followed at two second intervals by the display of the square root for each number.

```
10   OUTPUT 709; "INTEGER NUM"              ! Declare the Integer variable NUM.
20   OUTPUT 709; "SUB SQROOT"               ! Name of the subroutine is SQROOT.
30   OUTPUT 709; "  FOR NUM=10 TO 1 STEP    ! Start the display loop.
     -1"
40   OUTPUT 709; "  DISP NUM"               ! Display the number NUM.
50   OUTPUT 709; "  WAIT 2"                 ! Wait 2 seconds.
60   OUTPUT 709; "  DISP SQR(NUM)"          ! Display the square root of NUM.
70   OUTPUT 709; "  WAIT 2"                 ! Wait 2 seconds.
80   OUTPUT 709; "  NEXT NUM"               ! Continue the loop.
90   OUTPUT 709; "SUBEND"                   ! End of the subroutine SQROOT.
       .
       .
       .
200  OUTPUT 709; "CALL SQROOT"              ! Call the subroutine SQROOT for
                                              immediate execution.
```


www.valuetronics.com

Program Example #4

This example shows how to first pause and then to continue a subroutine. When executed, the controller prints the numbers 1 and 2. The fast display mode is turned off so that you can see the subroutine execute each command on the front panel display.

```
10   OUTPUT 709; "INTEGER FLAG, ONE, TWO"  ! Declare the variables FLAG, ONE, TWO.
20   OUTPUT 709; "ONE=1;TWO=2"             ! Initialize variables ONE and TWO.
30   OUTPUT 709; "SUB TOGGLE"              ! Name of the subroutine is TOGGLE.
40   OUTPUT 709; "  VWRITE FLAG,ONE"       ! Write 1 into FLAG.
50   OUTPUT 709; "  PAUSE"                 ! Pause the subroutine.
60   OUTPUT 709; "  VWRITE FLAG,TWO"       ! Write 2 into FLAG.
70   OUTPUT 709; "SUBEND"                  ! End of the subroutine TOGGLE.
       .
       .
       .
130  OUTPUT 709; "FASTDISP OFF"            ! Turn the fast display mode OFF.
140  OUTPUT 709; "CALL TOGGLE"             ! Call the subroutine TOGGLE for immediate
                                             execution. TOGGLE loads FLAG with 1
                                             and then pauses.
150  OUTPUT 709; "VREAD FLAG,IASC"         ! Read FLAG into the HP-IB output buffer.
160  ENTER 709; Flag_val                   ! Enter the flag value.
170  PRINT Flag_val                        ! Print the flag value.
180  OUTPUT 709; "CONT"                    ! Continue the subroutine. FLAG is loaded
                                             with 2 and the subroutine completes.
190  OUTPUT 709; "VREAD FLAG,IASC"         ! Read FLAG into the HP-IB output buffer.
200  ENTER 709; Flag_val                   ! Enter the flag value.
210  PRINT Flag_val                        ! Print the flag value.
```

Instead of PAUSE/CONTinue, we could STEP/CONTinue this program by deleting line 50 and changing line 140 to read:

```
140  OUTPUT 709; "STEP TOGGLE;STEP"
```

Then, when the program runs, line 140 steps to and executes the beginning command in TOGGLE (VWRITE FLAG,ONE), and then returns to line 150, continuing the program from that point the same as before.

www.valuetronics.com

Program Example #5

This example stores a subroutine, declares a variable and an array, and then uses the CAT (catalog) command to view all of the current variables, arrays, and subroutines in mainframe memory.

```
10   OUTPUT 709; "RESET"                    ! Delete all subs, variables, and arrays.
20   INTEGER I, N
30   DIM Name$[19],Kind$[19]
40   OUTPUT 709; "SUB BEEPER"               ! Start of subroutine BEEPER.
50   OUTPUT 709; "   BEEP"
60   OUTPUT 709; "SUBEND"                   ! End of subroutine BEEPER.
70   OUTPUT 709; "REAL R"                   ! Declare R as a Real number variable.
80   OUTPUT 709; "INTEGER A(19)"            ! Delare A as an Integer array with 20
                                              elements.
90   OUTPUT 709; "CAT"                      ! List the catalog entries over HP-IB.
100  ENTER 709;N                            ! Enter the total number of subs, arrays,
                                              and variables in memory into N.
110  PRINT "Subs, arrays, and variables in  ! Print the statement.
     HP 3852A memory = ";N
120  PRINT
130  PRINT "Name   "," Type   Size"         ! Print the column headings.
140  PRINT
150  FOR I = 1 TO N                         ! Start the catalog item entry loop.
160  ENTER 709;Name$,Kind$                  ! Enter and print the name, type, and size
170  PRINT Name$,Kind$                        for each catalog entry.
180  NEXT I                                 ! Finish the loop.
190  END                                    ! Terminate.
```

A typical printout for this program follows:

```
Subs, arrays, and variables in HP 3852A memory = 3

        Name          Type          Size

 "A             "   "IARR         20"
 "R             "   "REAL          0"
 "BEEPER        "   "SUB         110"
```


www.valuetronics.com

Program Example #6

This example illustrates a subroutine IF/THEN construct. When the program runs, the HP 3852A tests three times for I = 0, displaying the actual value of I each time. If I = 0, I is changed to 1. If I <> 0, then a message is displayed.

```
10   OUTPUT 709; "INTEGER I"                ! Declare the Integer variable I.
20   OUTPUT 709; "SUB IFTEST"               ! Name of the subroutine is IFTEST.
30   OUTPUT 709; "   IF I=0 THEN"           ! If condition is true (If I=0).
40   OUTPUT 709; "   I = I+1"               ! Increment I.
50   OUTPUT 709; "   ELSE"                  ! If condition is not true (I< >0).
60   OUTPUT 709; "   DISP 'NO CHANGE'"      ! Display the message.
70   OUTPUT 709; "   WAIT 2"                ! Wait 2 seconds.
80   OUTPUT 709; "   END IF"                ! End the condition check.
90   OUTPUT 709; "SUBEND"                   ! End of the subroutine IFTEST.
          .
          .
          .
200  OUTPUT 709; "DISP I"                   ! Display I (has initial value of 0)
210  OUTPUT 709; "WAIT 2"                   ! Wait 2 seconds.
220  OUTPUT 709; "CALL IFTEST"              ! Call IFTEST for immediate execution.
230  OUTPUT 709; "DISP I"                   ! Display I (I is now 1).
240  OUTPUT 709; "WAIT 2"                   ! Wait 2 seconds.
250  OUTPUT 709; "CALL IFTEST"              ! Call IFTEST for immediate execution.
260  OUTPUT 709; "DISP I"                   ! Display I (no change. . .I is still 1).
```

When this program runs, the HP 3852A should display the following (in sequence) at 2-second intervals:

```
0
1
No Change
1
```


www.valuetronics.com

Program Example #7

This example sets the HP 3852A alarm timer to real time plus 10 seconds. When the alarm occurs, the HP 3852A calls a subroutine to save the alarm time, beep and display a message. Next, the program inputs, converts, and displays the actual alarm time.

```
10   OUTPUT 709; "RST"                  ! Reset the HP 3852A.
20   REAL T,H,M,S                       ! Declare program variables T,H,M,S.
30   INTEGER I                          ! Declare program variable I.
40   OUTPUT 709; "REAL TM"              ! Declare the HP 3852A real variable TM.
50   OUTPUT 709; "SUB TIMER"            ! Name of the subroutine is TIMER.
60   OUTPUT 709; "   TIME INTO TM"      ! Place real time into TM (alarm time).
70   OUTPUT 709; "   DISP 'ALARM'"      ! Display the message.
80   OUTPUT 709; "   BEEP"              ! Beep.
90   OUTPUT 709; "   WAIT 1"            ! Wait one second.
100  OUTPUT 709; "   BEEP"              ! Beep.
110  OUTPUT 709; "SUBEND"               ! End of the subroutine TIMER.
120  OUTPUT 709; "ON ALRM CALL TIMER"   ! At the alarm, call TIMER.
130  OUTPUT 709; "TIME INTO TM"         ! Place real time into TM (prepare alarm).
140  OUTPUT 709; "SET ALRM,(TM+10)"     ! Set the alarm for real time + 10 seconds.
150  OUTPUT 709; "ENABLE ALRM"          ! Enable the alarm.
160  FOR I=1 TO 10                      ! Start a count loop to wait for the alarm.
170  DISP I                             ! Display the counter (I).
180  WAIT 1                             ! Wait 1 second.
190  NEXT I                             ! Continue the count.
200  PAUSE                              ! Pause the program. After the HP 3852A
                                          beeps twice, continue the program.
210  OUTPUT 709; "VREAD TM"             ! Tell HP 3852A to output the alarm time.
220  ENTER 709;T                        ! Enter the alarm time (in seconds).
230  DISP "Alarm at ";TIME$(T)          ! Display the alarm time.
240  OUTPUT 709; "OFF ALRM"             ! Cancel the ON...CALL command
                                          (line 120).
250  END                                ! Terminate.
```

If the alarm occurs at (for example) 4:32:27 pm, the program displays:

```
Alarm at 16:32:27
```

Since the HP 3852A timer resets to zero at midnight (86400 seconds), be sure to add a "midnight check" if your program is to be used near midnight. For instance, in program example #7 we could add the following lines to see if present time is within ten seconds of midnight. If no, the program continues at line 140 as before. If yes, the program sets the alarm to 10 seconds after midnight and branches around line 140.

```
131  OUTPUT 709; "VREAD TM"
132  ENTER 709; T
133  IF T+10 < 86400 THEN 140
134  OUTPUT 709; "SET ALRM, 10"
135  GOTO 150
```


www.valuetronics.com

Program Example #8

Suppose you want to monitor a process over a 24 hour period, beginning at midnight. You want to sample 10 channels of analog DCV information, one set of 10 channel readings every hour.

This example downloads a measurement subroutine into the HP 3852A, and then calls the subroutine for execution once every hour. The process begins at midnight and continues for 24 hours. After each set of measurements is taken, the program retrieves the measurement data from the mainframe memory and stores the data in computer memory. After 24 hours, the program prints each set of measurement data together with the real time when the measurements were taken.

In this example, the program displays real time on the controller screen while it is waiting for the next set of measurements. However, since the timer operates as a program interrupt, the controller could be programmed to perform other tasks while waiting.

```
10   PRINTER IS 701                          ! Identify the printer address.
20   OUTPUT 709; "RST"                       ! Reset the HP 3852A.
30   REAL Readings(23,9),Realtime(23)        ! Declare two program arrays.
40   INTEGER Hour,Channel                    ! Declare two program variables.
50   Hour = 0                                ! Initialize Hour.
60   OUTPUT 709; "REAL R(9)"                 ! Declare array R in the HP 3852A.
70   OUTPUT 709; "SUB MEASURE"               ! Name of the subroutine is MEASURE.
80   OUTPUT 709; "   USE 600"                ! Use the voltmeter in mainframe slot 6.
90   OUTPUT 709; "   CONFMEAS DCV 0-9        ! Take DCV measurements from channels
     INTO R"                                   0-9 (slot 0) and store them in array R.
100  OUTPUT 709; "SUBEND"                    ! End of subroutine.
110  ON TIME TIME("00:00:00") GOTO 130       ! Set timer for midnite.
120  GOTO 220                                ! Wait for midnite. (Note: Program may be
                                               used for other tasks while waiting.)
130  Realtime(Hour) = TIMEDATE               ! Store real time for this set of readings
                                               to start the measurement routine.
140  ON TIME ((Realtime(Hour) MOD 86400)     ! Set timer to reading time plus one hour for
     +3600) GOTO 130                           next set of readings.
150  OUTPUT 709; "CALL MEASURE"              ! Call MEASURE. Take one set of readings.
160  OUTPUT 709; "VREAD R"                   ! Transfer readings from array R to HP-IB
                                               output buffer.
170  FOR Channel = 0 to 9                    ! Set up data transfer loop.
180  ENTER 709; Readings(Hour,Channel)       ! Transfer channel data into Readings( , ).
190  NEXT Channel                            ! Repeat until finished.
200  Hour = Hour+1                           ! Update Hour for the next set of readings.
210  IF Hour=24 THEN 250                     ! If done, start data printout.
220  DISP "The time is ";TIME$(TIMEDATE)     ! Display the present time.
```


www.valuetronics.com

```
230  WAIT 1                                  ! Wait one second.
240  GOTO 220                                ! Repeat time display until next set of
                                               readings is due. (Program may perform
                                               other tasks while waiting.)
250  Hour = 0                                ! Begin printout of data. Initialize Hour.
260  OFF TIME                                ! Turn off the timer.
270  PRINT "Readings taken at ";             ! Print the actual time the next set of
     TIME$(Realtime(Hour))                     printed readings was taken.
280  PRINT                                   ! Space one line.
290  FOR Channel = 0 TO 9                    ! Begin loop to print this reading set.
300  PRINT "Ch. ";Channel;" = ";             ! Print one reading.
     Readings(Hour,Channel)
310  NEXT Channel                            ! Continue the loop.
320  PRINT                                   ! Space one line.
330  Hour = Hour+1                           ! Update Hour for next reading set.
340  IF Hour=24 THEN 360                     ! If finished, then exit.
350  GOTO 270                                ! Print the next set of readings.
360  END                                     ! Terminate.
```


www.valuetronics.com

Contents

www.valuetronics.com

Chapter 10

Data Processing

Introduction

The HP 3852A's data processing functions allow you to keep a log of measurement channels, perform math operations, keep track of measurement statistics, determine when readings exceed high or low limits, create a user defined lookup table, and convert resistance and voltage measurements to temperatures and strain. This chapter shows you how to use each of these data processing functions and consists of the following sections:

- **Introduction** contains an overview of the chapter.
- **Command Summary** summarizes the data processing commands.
- **Logging Channels** describes how to use the ENABLE LOGCHAN, DISABLE LOGCHAN, and LOGCHAN commands to keep a list of the measurement channels used.
- **Math Operations** describes how to use the SCALE command to perform mathematical modifications on data.
- **Statistics** describes how to use the STAT command to keep track of the high value, low value, mean, and standard deviation of a set of readings.
- **Limit Checking** describes how to use the *post processing* and *real time* LMT command to determine when readings are beyond specified limits.
- **User Defined Lookup Table** describes how to use the CONV command to create a lookup table which modifies readings according to your specifications.
- **Post Process Temperature and Strain Conversion** describes how to use the COMPEN command and to convert resistance and voltage measurements to corresponding temperatures and strain.

Command Summary

The data processing commands are COMPEN, CONV, DISABLE LOGCHAN, ENABLE LOGCHAN, LMT (post process), LMT (real time), LOGCHAN, SCALE, and STAT.

COMPEN Enables you to measure the electrical parameters (i.e. resistance, voltage) of a thermistor, RTD, thermocouple, or strain gage, and later convert those parameters to temperatures or strain.

www.valuetronics.com

CONV Provides a linear interpolation of measurement data using a user defined lookup table.

DISABLE LOGCHAN Disables measurement channel logging previously enabled with the ENABLE LOGCHAN command. (In the power-on state, logging is disabled).

ENABLE LOGCHAN Causes the channel number for each reading taken to be stored in a mainframe array.

LMT (post-processing) Compares readings stored in an array to specified minimum and maximum limits. The indices of any readings out of limits are stored in a separate array.

LMT (real-time) Compares each reading taken to specified minimum and maximum limits. The indices of any readings out of limits are stored in a separate array.

LOGCHAN Designates the array in which the channel numbers are stored when channel logging is enabled (see ENABLE LOGCHAN command).

SCALE Subtracts the specified offset value from the stored reading and divides the result by the specified scale value.

STAT Performs statistical analysis (high reading, low reading, mean, standard deviation) on stored readings.

Indexing

The data processing commands rely on mainframe arrays to store user specified data, measured readings, and results of computations or record keeping. The data processing commands affect the indexing of these arrays as follows:

- Any array that is *read from* during the execution of a data processing command has its index preset to 0 before being read and is incremented following each read.

- Any array that is *written to* during execution of a data processing command is not preset to 0 but is incremented after each element is stored. This allows you to append data from successive command executions to the same array without overwriting the previous data. If you want to overwrite previous data, execute the INDEX command specifying the array and the index number of the first element you wish to overwrite (usually element 0).

www.valuetronics.com

Channel Logging

When enabled, the channel logging function stores the channel number of each reading taken into a mainframe array. You enable channel logging using the ENABLE LOGCHAN command. You designate the array in which to store the channel numbers using the LOGCHAN command.

NOTE

When using the MEAS or CONFMEAS command to make measurements, the corresponding multiplexer channel numbers will be logged. When using the CHREAD or XRDGS command, only the channel number of the measurement device (e.g., a voltmeter) is logged.

Before you can identify the array in which to store channel numbers (LOGCHAN command) you must declare the array. Since channel numbers are always Integers, it is best to declare the array using the INTEGER command. For example, the following program segment declares an array to hold 7 channel numbers (CHS) and an array to hold 7 readings (RGS).

```
10  OUTPUT 709;"INTEGER CHS(6)"
20  OUTPUT 709;"REAL RGS(6)"
```

You can now identify the channel number array and enable channel logging as follows:

```
30  OUTPUT 709;"LOGCHAN CHS"
40  OUTPUT 709;"ENABLE LOGCHAN"
```

NOTE

You must establish the channel number array (LOGCHAN command) before enabling channel logging (ENABLE LOGCHAN command).

Now, as a method to generate seven measurements (assuming a voltmeter in mainframe slot 1 and a multiplexer in slot 2), send the following:

```
50  OUTPUT 709;"CONFMEAS DCV,202-205,207-209,USE 100,INTO RGS"
```

The channel numbers and the readings have been stored in their respective arrays. The following program combines the previous program segments with additional HP 3852A and controller commands to display the channels and readings on the controller.

www.valuetronics.com

```
5     DIM Chan(0:6),Reads(0:6)
10    OUTPUT 709;"INTEGER CHS(6)"
20    OUTPUT 709;"REAL RGS(6)"
30    OUTPUT 709;"LOGCHAN CHS"
40    OUTPUT 709;"ENABLE LOGCHAN"
50    OUTPUT 709;"CONFMEAS DCV,202-205,207-209,USE 100,INTO RGS"
60    OUTPUT 709;"VREAD CHS"
70    ENTER 709; Chan(*)
80    OUTPUT 709;"VREAD RGS"
90    ENTER 709;Reads(*)
100   FOR I = 0 TO 6
110   PRINT Chan(I);" ";Reads(I)
120   NEXT I
130   END
```

Typically, the controller might display:

```
202   1.270625
203   1.196875
204   1.12065
205   1.043125
207   .988125
208   .939375
209   .905
```

NOTE

You do not have to store the readings in order to log the channel numbers. Readings can be directly transferred over HP-IB and the channel numbers logged.

You can determine the total number of channels logged using the INDEX? command. Refer to INDEX? in the command reference for more information.

After logging channel numbers, you can disable the channel logging function by executing:

```
51  OUTPUT 709;"DISABLE LOGCHAN"
```


www.valuetronics.com

Math Operations

This section describes how the SCALE command is used to modify measurement data previously stored in the mainframe.

SCALE

The *scale* operation modifies stored readings by subtracting an offset and dividing by a scale factor. The equation is:

Result = (*reading* - *offset*)/ *scale*

The syntax for the SCALE command is:

SCALE *offset scale readings* [**INTO** *name*] or [*fmt*]

Where:

offset = the name of an INTEGER or REAL variable or array containing offset(s) to be subtracted from the reading(s).

scale = the name of an INTEGER or REAL variable or array containing scale factor(s) to be divided into the result(s) of (*reading* - *offset*).

readings = the name of an INTEGER or REAL variable or array containing readings to be modified.

INTO *name* = the name of a variable or an array in which to store results of the scale operation.

fmt = the output buffer or display format specified.

Specifying Values

If a variable containing a single offset or scale factor is used, the single value is used for each reading. If arrays containing multiple offsets and scale factors are specified, each reading is modified by its correspondingly numbered offset and scale factor. That is, the first reading has the first offset subtracted from it and is divided by the first scale factor. The second reading uses the second offset, second scale factor, and so on. If there are more readings than specified offsets/scale factor values, a wraparound occurs and the values are re-used. This allows you to specify a limited number of offsets/scale factors when repeatedly modifying readings from a fixed number of channels.

Using the SCALE Command

Before you can use the SCALE command, you must declare the variables or arrays that will hold the offset(s), the scale factor(s), the readings, and the results (if desired). The following program line declares these arrays (each array is given 10 elements):

```
10  OUTPUT 709;"REAL OFST(9),SCL(9),RGS(9),RES(9)"
```


www.valuetronics.com

You can now store the offset and scale factor values into the OFST and SCL arrays. For example, the following program segment stores 10 offset values and 10 scale values into the appropriate arrays:

```
20 OUTPUT 709;"VWRITE OFST,1,2,3,4,5,6,7,8,9,10"
30 OUTPUT 709;"VWRITE SCL,10,9,8,7,6,5,4,3,2,1"
```

NOTE

The HP 3852A allows only ten values per list. If you need to write more than ten values to an array you must use a series of VWRITE commands. For example, if you are writing 15 values to the OFST array, it could be done as follows:

```
20 OUTPUT 709;"VWRITE OFST,1,2,3,4,5,6,7,8,9,10"
21 OUTPUT 709;"VWRITE OFST,11,12,13,14,15"
```

Line 21 appends offsets to the array starting at array index 10 (eleventh element). Note that the array (OFST) when declared, must be large enough for the additional offsets.

Now, as a method to generate ten readings and store them in the RGS array (assuming a voltmeter in mainframe slot 1 and a multiplexer in slot 2), send the following:

```
40 OUTPUT 709;"CONFMEAS DCV,200-209,USE 100,INTO RGS"
```

You can now execute the SCALE command. The following program line performs the offset and scale operations on each stored reading. For the first reading, an offset of 1 and scale of 10 is used. For the second reading, offset = 2, scale = 9, and so on. The results are stored in the RES array.

```
50 OUTPUT 709;"SCALE OFST,SCL,RGS,INTO RES"
```


www.valuetronics.com

The following program combines the previous program segments with additional HP 3852A and controller commands to display the scaled results on the controller.

```
5    DIM Rslt(0:9)
10   OUTPUT 709;"REAL OFST(9),SCL(9),RGS(9),RES(9)"
20   OUTPUT 709;"VWRITE OFST,1,2,3,4,5,6,7,8,9,10"
30   OUTPUT 709;"VWRITE SCL,10,9,8,7,6,5,4,3,2,1"
40   OUTPUT 709;"CONFMEAS DCV,200-209,USE 100,INTO RGS"
50   OUTPUT 709;"SCALE OFST,SCL,RGS,INTO RES"
60   OUTPUT 709;"VREAD RES"
70   ENTER 709;Rslt(*)
80   PRINT USING "K,/";Rslt(*)
90   END
```

Typically, the controller might display:

```
-.1
-.222202
-.375
-.5714282
-.8333333
-1.2
-1.75
-2.666667
-4.5
-10
```

Statistics

You can tabulate statistics (high reading, low reading, mean, and standard deviation) on a group of stored readings using the STAT command.

The syntax for the STAT command is:

STAT *min max mean std var*

Where:

min = Name of REAL or INTEGER variable or array to receive the lowest value.

max = Name of REAL or INTEGER variable or array to receive the highest value.

mean = Name of REAL or INTEGER variable or array to receive the mean of all values.

www.valuetronics.com

std = Name of REAL or INTEGER variable or array to receive the standard deviation of all values.

var = Name of variable or array containing values from which the statistics are generated.

NOTE

You do not necessarily have to set up a separate variable or array for min, max, mean, and std. You can specify the same array for each and the statistics will be stored in the array in the order listed in the STAT command (first value = min, second = max, third = mean, fourth = std. deviation).

The STAT command sets the index pointer of the var array to 0 before and after command execution.

Using the STAT Command

Before you can use the STAT command, you must declare the variables or arrays that will hold the low reading, the high reading, the mean, and the standard deviation. The following program segment declares variables L = low, H = high, M = mean, S = standard deviation, and an array to hold measurements (DATA):

```
10  OUTPUT 709;"REAL L,H,M,S"
20  OUTPUT 709;"REAL DATA(19)"
```

Now, as a method to generate 20 readings and store them in the DATA array (assuming a voltmeter in mainframe slot 1 and a multiplexer in slot 2), send the following:

```
30  OUTPUT 709;"CONFMEAS DCV,200-219,USE 100,INTO DATA"
```

You can now execute the STAT command as follows:

```
40  OUTPUT 709;"STAT L,H,M,S,DATA"
```


www.valuetronics.com

The STAT command tabulates the statistics and stores them in the appropriate variables. The following program combines the previous program segments with additional HP 3852A and controller commands to display the statistics on the controller.

```
10   OUTPUT 709;"REAL L,H,M,S"
20   OUTPUT 709;"REAL DATA(19)"
30   OUTPUT 709;"CONFMEAS DCV,200-219,USE 100,INTO DATA"
40   OUTPUT 709;"STAT L,H,M,S,DATA"
50   OUTPUT 709;"VREAD L"
60   ENTER 709;A
70   PRINT "Lowest value = ";A
80   OUTPUT 709;"VREAD H"
90   ENTER 709;B
100  PRINT "Highest value =";B
110  OUTPUT 709;"VREAD M"
120  ENTER 709;C
130  PRINT "Mean = ";C
140  OUTPUT 709;"VREAD S"
150  ENTER 709;D
160  PRINT "Std. Deviation = ";D
170  END
```

Typically, the controller might display:

```
Lowest Value = .525096
Highest Value = 4.77375
Mean = .8900179
Std. Deviation = .917129
```

Note that in the previous program, an array is declared to store 20 readings (DATA(19)) and 20 measurements are taken (CONFMEAS DCV 200-219). The STAT command does a statistical analysis based on the data in all elements of the array. If only 15 readings had been taken, the 0's in the five remaining elements of the array would be interpreted and reported as the lowest value measured. This, in turn, affects the accuracy of the data returned by the STAT command. Thus, to ensure accurate data when you are using the STAT command, the array (*var*) should contain a valid reading in each element.

www.valuetronics.com

The previous program can be simplified somewhat by setting up a single array to store the results of the STAT command. Recall that the data returned will appear in the array in the order listed by the STAT command (i.e. first value = min, second = max, third = mean, fourth = standard deviation). An example is shown below.

```
10   OUTPUT 709;"REAL RES(3)"
20   OUTPUT 709;"REAL DATA(19)"
30   OUTPUT 709;"CONFMEAS DCV, 200-219, USE 100, INTO DATA"
40   OUTPUT 709;"STAT RES, RES, RES, RES, DATA"
50   OUTPUT 709;"VREAD RES"
60   ENTER 709;A,B,C,D
70   PRINT "Lowest value = ";A
80   PRINT "Highest value = ";B
90   PRINT "Mean = ";C
100  PRINT "Standard Deviation = ";D
110  END
```

Limit Checking

The HP 3852A can perform limit checking by comparing each reading against user specified upper and lower limits. The indices of any readings out of limits are then stored in a separate array. Limit checking is set up using the LMT command. Two limit checking modes are available: post processing, and real time. Post processing performs limit checking on a group of stored readings. Real time performs limit checking on readings as soon as they become available.

NOTE

The HP 3852A determines whether the LMT command is post processing or real time by the presence or absence of the var parameter. If the var parameter is included in the command statement, the command is post processing. If not, the command is real time.

---

www.valuetronics.com

Setting Limits

Minimum and maximum limits may be a single value stored in a variable or multiple values stored in arrays. If you specify a single value for the upper limit and a single value for the lower limit, every reading is compared against these limits. If you specify arrays for upper and lower limits, each reading is compared to its correspondingly numbered limits. That is, the first reading is compared to the first limits, the second reading to the second limits, and so on. The limit values are numbered in the order specified; first entered = first compared. The *min* and *max* arrays will wraparound; that is, if there is a greater number of readings than specified limits, the first limits are re-used. This allows you to have a small number of limits being repetitively compared to a larger number of readings.

Post Processing Limit Checking

Post processing limit checking compares stored readings against upper and lower limits and stores the indices of out of limit readings in a separate array. Unlike real time limit checking, post processing limit checking does not set the LMT bit in the Status Register and does not turn on the LMT annunciator when a limit is exceeded. Thus, post processing limit checking is not an interrupt source and, therefore, is unaffected by the ENABLE LMT and DISABLE LMT commands.

The syntax for the post processing LMT command is:

LMT *min max index_store var*

Where:

min = Name of a REAL or INTEGER variable or array containing the lower limit(s).

max = Name of a REAL or INTEGER variable or array containing the upper limit(s).

index_store = Name of a REAL or INTEGER array that will store index numbers for readings out of specified limits. The index numbers correspond to the index numbers of the stored readings.

var = Name of an array containing the stored readings to be compared against the limits.

Before you can perform post processing limit checking, you must declare the variables or arrays that will hold the upper and lower limits, an array for the index numbers, and an array for the readings. The following program segment declares variables for the upper and lower limit and 20 element arrays for the readings to be compared and for the index numbers to be stored.

www.valuetronics.com

```
10    OUTPUT 709;"REAL UPPER,LOWER"
20    OUTPUT 709;"INTEGER INDX(19)"
30    OUTPUT 709;"REAL RGS(19)"
```

You can now store the upper and lower limits into the UPPER and LOWER arrays:

```
40    OUTPUT 709;"VWRITE UPPER,10.25"
50    OUTPUT 709;"VWRITE LOWER,5.12"
```

Now, as a method to generate 20 readings (assuming a voltmeter in mainframe slot 1 and a multiplexer in slot 2), send the following:

```
60    OUTPUT 709;"CONFMEAS DCV,200-219,USE 100,INTO RGS"
```

The HP 3852A will now store the readings into the RGS array.

Now, to enable post processing limit checking and designate the appropriate arrays, send:

```
70    OUTPUT 709;"LMT LOWER,UPPER,INDX,RGS"
```

Each reading stored in the RGS array is compared against the upper and lower limit. The indices of any readings out of limits are stored in the INDX array.

The following program combines the previous program segments with additional HP 3852A and controller commands to determine if any readings are out of limit and if so, what their indices are.

```
5     DIM Ind(0:19)
10    OUTPUT 709;"REAL UPPER,LOWER"
20    OUTPUT 709;"INTEGER INDX(19)"
30    OUTPUT 709;"REAL RGS(19)"
40    OUTPUT 709;"VWRITE UPPER,10.25"
50    OUTPUT 709;"VWRITE LOWER,5.12"
60    OUTPUT 709;"CONFMEAS DCV,200-219,USE 100,INTO RGS"
70    OUTPUT 709;"LMT LOWER,UPPER,INDX,RGS"
80    OUTPUT 709;"INDEX? INDX"
90    ENTER 709;A
100   PRINT "NUMBER OF READINGS OUT OF LIMITS = ";A
110   IF A<1 THEN 180
120   OUTPUT 709;"VREAD INDX"
130   ENTER 709;Ind(*)
140   PRINT "INDICES ARE:"
150   FOR I=0 TO A-1
160   PRINT Ind(I)
170   NEXT I
180   END
```


www.valuetronics.com

For example, if the first, fourth, and eighth readings are out of limits, the controller displays:

```
NUMBER OF READINGS OUT OF LIMITS = 3
INDICES ARE:
0
3
7
```

If none of the readings are out of limits, the controller displays:

```
NUMBER OF READINGS OUT OF LIMITS = 0
```

Real Time Limit Checking

Real time limit checking compares each reading to lower and upper limits as the reading is taken. Each reading that is out of limits is indicated by an array index number which corresponds to the order in which the reading was taken.

A reading out of limits is also the source of an interrupt that can be handled by the mainframe or by the controller. A reading out of limits during real time testing sets the LMT bit in the mainframe Status Register and turns on the LMT annunciator. If the LMT bit in the register is unmasked and the service request mode is enabled, a service request message is sent to the controller so the controller can respond. If the limit exception interrupt is to be handled by the mainframe, the interrupt is sensed by the mainframe prior to the time the bit is set in the Status Register. See Chapter 8 for more information on setting up and responding to limit exception interrupts.

The syntax for the real time LMT command is:

LMT *min max index_store*

Where:

min = Name of the REAL or INTEGER variable or array holding the lower limit(s).

max = Name of the REAL or INTEGER variable or array holding the upper limit(s).

index_store = Name of REAL or INTEGER array that will store index numbers for readings out of the specified limits. The index numbers correspond to the order of the readings taken in each command returning data.

www.valuetronics.com

Before you can use the LMT command, you must declare the variables or arrays that will hold the upper and lower limits, and an array to store the index numbers of the readings out of limits. The following program segment declares upper and lower limit arrays which will contain three sets of limits. The *index_store* array is also declared to store the index numbers of all the readings taken which could be out of limit.

```
10 OUTPUT 709;"REAL UPPER(2),LOWER(2)"
20 OUTPUT 709;"INTEGER INDX(2)"
```

You can now store the upper and lower limits into the UPPER and LOWER arrays. The following program segment stores three upper limits and three lower limits into the appropriate arrays:

```
30 OUTPUT 709;"VWRITE UPPER,5.25,4.5,7.83"
40 OUTPUT 709;"VWRITE LOWER,3.75,1.414,.707"
```

NOTE

The HP 3852A allows only ten values per list. If you need to write more than ten values to an array you must use a series of VWRITE commands. For example, if you are writing 15 values to the UPPER array, it could be done as follows:

```
30 OUTPUT 709;"VWRITE UPPER,1,2,3,4,5,6,7,8,9,10"
31 OUTPUT 709;"VWRITE UPPER,11,12,13,14,15"
```

Line 31 appends limits to the array starting at array index 10 (eleventh element). Note that the array (UPPER) when declared, must be large enough for the additional limits.

You can now designate the arrays for real time limit checking by sending:

```
50 OUTPUT 709;"LMT LOWER,UPPER,INDX"
```

To enable real time limit checking, send:

```
60 OUTPUT 709;"ENABLE LMT"
```

NOTE

You must execute the LMT command to designate the arrays before executing the ENABLE LMT command.

---

www.valuetronics.com

Now, as a method to generate three readings (assuming a voltmeter in mainframe slot 1 and a multiplexer in slot 2), send the following:

```
70  OUTPUT 709;"CONFMEAS DCV,200-202,USE 100"
```

The HP 3852A will now perform limit checking on each reading as it is taken. For the first reading, an upper limit of 5.25 and a lower limit of 3.75 is used. For the second reading, upper = 4.5, lower = 1.414. For the third reading, upper = 7.83, lower = .707.

The following program modifies some of the previous program segments and combines them with additional HP 3852A and controller commands to determine if any readings are out of limits. If one or more readings are out of limit, the indices and the readings themselves are displayed.

```
5     DIM Ind(0:2),Limrgs(0:2)
10    OUTPUT 709;"REAL UPPER(2),LOWER(2)"
20    OUTPUT 709;"INTEGER INDX(2)"
21    OUTPUT 709;"REAL RGS(2)"
30    OUTPUT 709;"VWRITE UPPER,5.25,4.5,7.83"
40    OUTPUT 709;"VWRITE LOWER,3.75,1.414,.707"
50    OUTPUT 709;"LMT LOWER,UPPER,INDX"
60    OUTPUT 709;"ENABLE LMT"
70    OUTPUT 709;"CONFMEAS DCV 200-202,USE 100,INTO RGS"
80    OUTPUT 709;"INDEX? INDX"
90    ENTER 709;A
100   PRINT"NUMBER OF READINGS OUT OF LIMITS = ";A
110   IF A<1 THEN 230
120   OUTPUT 709;"VREAD INDX"
130   ENTER 709;Ind(*)
140   PRINT"INDICES ARE:"
150   FOR I=0 TO A-1
160   PRINT Ind(I)
170   NEXT I
180   PRINT
190   OUTPUT 709;"VREAD RGS"
200   ENTER 709;Limrgs(*)
210   PRINT "READINGS WERE:"
220   PRINT USING "K,/";Limrgs(*)
230   END
```


www.valuetronics.com

For example, if all readings are out of limits, the controller displays:

```
NUMBER OF READINGS OUT OF LIMITS = 3
INDICES ARE:
0
1
2

READINGS WERE:
5.89299
.2599538
.2524343
```

If none of the readings are out of limits, the controller displays:

```
NUMBER OF READINGS OUT OF LIMITS = 0
```

NOTE

If none of the readings are out of limits, the INDX array is filled with zeros. In the case of the first reading (position 0), you can't determine if it was in or out of limits since an out of limits reading for position 0 returns 0 as the index number. For this reason, you must check the index pointer position. If the pointer is at position 1, the first reading was out of limits. If the pointer is at position 0, the first reading was within the limits. The above program shows a good method for checking the position of the index pointer (lines 80 through 100).

To disable real time limit checking, send:

```
71  OUTPUT 709;"DISABLE LMT"
```

NOTE

Pressing the front panel CLEAR key, executing the CLR command, or executing the HP-IB CLEAR command also disables real time limit checking.

---

www.valuetronics.com

User Defined Lookup Table

You can use a user defined lookup table using the CONV command. You create the lookup table by entering two sets of data as input/output pairs. The HP 3852A then evaluates each stored reading by finding the nearest input values on either side of each reading and then linearly interpolating the result between the corresponding output values (those with the same index).

The CONV command is particularly useful when using sensors such as strain gages; pressure transducers; and thermistors or thermocouples not supported by the HP 3852A's conversion routines.

The syntax for the CONV command is:

CONV *domain range var* [**INTO** *name*] or [*fmt*]

Where:

domain = REAL array of x-coordinates (input values) on the graph of the measurements to be interpolated. This array must have the same maximum index as the range array and the x values must be in order of increasing magnitude. That is, X(0) < X(1) < X(2), and so on.

range = REAL array of y-coordinates (output values) on the graph of the measurements to be interpolated. These values correspond to the x values having the same index. This array must have the same maximum index as the domain array. For best accuracy, the y values, not necessarily the x values, should be spaced equal distances from each other.

var = array containing readings which will be interpolated according to the x to y mapping defined by the domain and range arrays.

INTO *name* = stores the converted data in mainframe memory.

fmt = output buffer/display format specifier. When used, the reading is returned to the output buffer and/or display in the format specified.

NOTE

The domain, range, var, and destination (if used) arrays must be separate arrays.

---

www.valuetronics.com

Entering Values

You enter values into the domain and range arrays (using the VWRITE command) as input-output pairs matched by index. For example, in Table 10-1, an input value of 2 in element 0 of the domain array corresponds to an output value of 4 in element 0 of the range array, 3 in element 1 corresponds to 9, and so on.

Table 10-1. Input/Output Pair Example

array element (index):	0	1	2	3	4	5	...
domain (x) value:	2	3	4	5	6	7	...
range (y) value:	4	9	16	25	36	49	...

The measurements in the *var* array are linearly interpolated by solving for y in the equation:

$$\frac{(y - y1)}{(y2 - y1)} = \frac{(x - x1)}{(x2 - x1)}$$

where x is a measurement in the *var* array falling on, or within, a set of matched pairs, x1 is the x value of the lower matched pair, y1 is the y value of the lower matched pair, x2 is the x value of the upper matched pair, and y2 is the y value of the upper matched pair.

As an example, in the previous table, the set of ordered pairs (x,y) is (2,4), (3,9), (4,16)... If a measurement in the *var* array is 3.5, it will be interpolated to 12.5 based on the equation which is solved for y, given the domain and range values specified.

$$y = y1 + \frac{(x - x1)}{(x2 - x1)}(y2 - y1) \rightarrow y = 9 + \frac{(3.5 - 3)}{(4 - 3)}(16 - 9) = 12.5$$

The more array elements specified, the more accurate the conversion. The domain and range arrays must contain the same number of elements. After all the readings are converted, the index pointers of the var, domain, and range arrays are reset to zero (0).

NOTE

If a value in the var array does not fall within the bounds of the data in the domain array, an error will be generated (ERROR 49: CONV: DATA OUT OF BOUNDS) and the resulting value will be 1.0 E+38 (overload). Following the error, the interpolation will continue until all readings are converted.

---

www.valuetronics.com

Using CONV

As an example of using the CONV command, we'll enter the conversion values for a thermistor having a resistance of 3 kohms at 25°C. This particular thermistor is not supported by the HP 3852A's temperature conversion routines. Figure 10-1 is a graph showing the temperature vs. resistance of a 3 kohm thermistor between the temperatures of 10°C and 35°C.

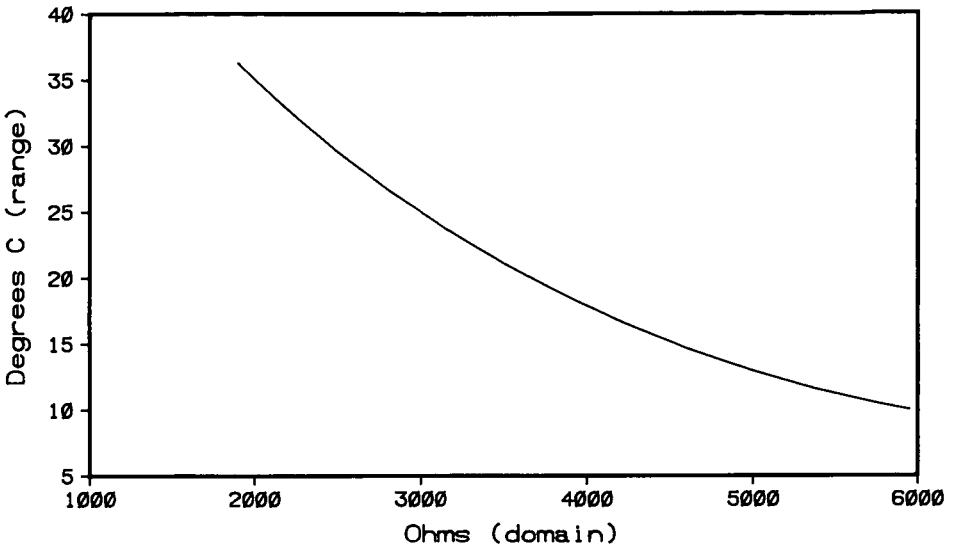


Figure 10-1. Temperature vs. Resistance, 3k Thermistor

Before you can use the CONV command, you must declare the arrays that will hold the x-coordinates, the y-coordinates, the readings, and, if desired, the converted readings. The following program line declares these arrays:

```
10  OUTPUT 709;"REAL DOM(13),RNGE(13),INPT(9),OTPT(9)"
```

You can now store the input/output pairs into the domain and range arrays. For the 3 kohm thermistor, 1880 Ω corresponds to 36° C, 2042 Ω = 34° C and so on. The following program segment writes the resistances and their corresponding temperatures to the domain and range arrays.

```
20  OUTPUT 709;"VWRITE DOM,1880,2042,2221,2417,2633,2872,3135,
    3426,3748,4105"
30  OUTPUT 709;"VWRITE DOM,4500,4939,5427,5971"
40  OUTPUT 709;"VWRITE RNGE,36,34,32,30,28,26,24,22,20,18"
50  OUTPUT 709;"VWRITE RNGE,16,14,12,10"
```


www.valuetronics.com

NOTE

Notice in the above segment that two VWRITE commands are used to write the information to each array. This is because the HP 3852A allows only ten elements in a list. Since we are writing 14 elements to each array, it must be done using two lists per array.

Now, as a method to generate ten 4-wire ohms readings and store them in the INPT array (assuming a voltmeter in mainframe slot 1 and a multiplexer in slot 2 measuring 10 3k thermistors), send the following:

```
60  OUTPUT 709;"CONFMEAS OHMF,200-209,USE 100,INTO INPT"
```

You can now execute the CONV command as follows:

```
70  OUTPUT 709;"CONV DOM,RNGE,INPT,INTO OTPT"
```

The CONV command now converts each stored reading into a temperature based on the values assigned to the domain and range arrays. The following program combines the previous program segments with additional HP 3852A and controller commands to display the temperature readings on the controller.

```
5     DIM Conv(0:9)
10    OUTPUT 709;"REAL DOM(13),RNGE(13),INPT(9),OTPT(9)"
20    OUTPUT 709;"VWRITE DOM,1880,2042,2221,2417,2633,2872,3135,
      3426,3748,4105"
30    OUTPUT 709;"VWRITE DOM,4500,4939,5427,5971"
40    OUTPUT 709;"VWRITE RNGE,36,34,32,30,28,26,24,22,20,18"
50    OUTPUT 709;"VWRITE RNGE,16,14,12,10"
60    OUTPUT 709;"CONFMEAS OHMF,200-209,USE 100,INTO INPT"
70    OUTPUT 709;"CONV DOM,RNGE,INPT,INTO OTPT"
80    OUTPUT 709;"VREAD OTPT"
90    ENTER 709;Conv(*)
100   PRINT USING"K,/";Conv(*)
110   END
```

Typically, the computer might display the following (notice the last reading is out of bounds):

```
23.29553
28.21875
19.07843
23.811
32.19099
26.70711
31.70408
18.48319
31.12213
1.E+38
```


www.valuetronics.com

Post Process Temperature and Strain Conversion

The COMPEN command enables you to measure the electrical parameters (i.e. resistance, voltage) of a thermistor, RTD, thermocouple, or strain gage, and later convert those parameters to temperatures or strain.

When applications involve a thermistor or an RTD, the readings converted by the COMPEN command are resistance (OHM, OHMF) measurements. When applications involve a thermocouple or strain gage, the readings converted are voltage (DCV) measurements. By specifying a resistance or voltage measurement with the appropriate transducer, the measurements occur much faster since the conversion to temperature or strain does not occur at the time the readings were taken.

Command Syntax

The syntax of the COMPEN command and a description of its parameters are given below.

COMPEN *thermistor* or *RTD ohms_array* [**GAIN** *corr*][**INTO** *name*] or [*fmt*]

thermistor or RTD - thermistor or RTD whose resistance is converted to temperature. Thermistors and RTDs supported by this command include:

THM2252
THM5K
THM10K
RTD85
RTD92

ohms_array - Real, Integer, or Packed array containing the resistance measurements of the thermistor or RTD used.

GAIN *corr* - Real or Integer array or a number containing values by which the readings in *ohms_array* are divided. If a number is specified, the value is divided into each of the readings in *ohms_array*. If an array is specified, there is a 1-for-1 correspondence between the correction array and *ohms_array*. If necessary, the correction array will wraparound and the correction factors will be used again.

INTO *name* - Array in which the converted readings are stored.

fmt - Data format in which the converted readings are returned. The default format is RASC.

COMPEN *thermocouple volts_array* **REF** *reft_array* [**GAIN** *corr*][**INTO** *name*] or [*fmt*]

www.valuetronics.com

thermocouple - thermocouple whose voltage is converted to temperature. Thermocouples supported by this command include:

TEMPB	(B type thermocouple)
TEMPE	(E type thermocouple)
TEMPJ	(J type thermocouple)
TEMPK	(K type thermocouple)
TEMPN14	(N14 type thermocouple)
TEMPN28	(N28 type thermocouple)
TEMPR	(R type thermocouple)
TEMPS	(S type thermocouple)
TEMPT	(T type thermocouple)

volts_array - Real, Integer, or Packed array containing the voltage measurements of the thermocouple used.

REF *reft_array* - Real or Integer array or a number containing the measurement(s) of the isothermal block (REFT). If a number is specified, the single reference is used for all voltage measurements in the volts array. If an array is specified, there is a 1-for-1 correspondence between the reference array and the volts array. If necessary, the reference array will wraparound and the references used again.

GAIN *corr* - Real or Integer array or a number containing values by which the readings in *volts_array* are divided. If a number is specified, the value is divided into each of the readings in *volts_array*. If an array is specified, there is a 1-for-1 correspondence between the correction array and *volts_array*. If necessary, the correction array will wraparound and the correction factors will be used again.

INTO *name* - Array in which the converted readings are stored.

fmt - Data format in which the converted readings are returned. The default format is RASC.

COMPEN *strain_function bridge_volts* **STRVEX** *ex_array* **REF** *ref_buf* [**GF** *factor*][**NU** *ratio*][**GAIN** *corr*][**INTO** *name*] or [*fmt*]

strain_function - bridge configuration whose output voltage is converted to strain. The strain functions which represent these configurations and which are supported by this command are:

STRQ	(¼ bridge strain)
STRHB	(bending ½ bridge strain)
STRFB	(bending full bridge strain)
STRHP	(½ bridge Poisson strain)
STRFBP	(bending full bridge Poisson strain)
STRFP	(full bridge Poisson strain)

www.valuetronics.com

bridge_volts - Real, Integer, or Packed array containing the bridge output voltage measurements.

STRVEX *ex_array* - Real or Integer array or a number containing the measurement(s) of the bridge excitation voltage. If a number is specified, the single excitation voltage is used for all measurements in the bridge volts array. If an array is specified, there is a 1-for-1 correspondence between the excitation voltages and the bridge volts array. If necessary, the excitation voltage array will wraparound and the voltages will be used again.

REF *ref_buf* - Real or Integer array or a number containing the unstrained reference measurement(s). If a number is specified, the single reference is used for all measurements in the bridge volts array. If an array is specified, there is a 1-for-1 correspondence between the references and the bridge volts array. The reference array will also wraparound if necessary.

GF *factor* - Real or Integer array or a number containing a gage factor. A single gage factor is used for all readings in the bridge volts array, or there is a 1-for-1 correspondence depending on whether a number or an array is specified. The gage factor array will wraparound if necessary. If no gage factor is specified, a default gage factor of 2.0 is used. Specifying a gage factor with an exponent of -6 will return the converted readings in microstrain.

NU *ratio* - Real or Integer array or a number containing a Poisson ratio. A single Poisson ratio is used for all readings in the bridge volts array, or there is a 1-for-1 correspondence depending on whether a number or an array is specified. The Poisson ratio array will wraparound if necessary. Note that when STRHP, STRFBP, or STRFP is specified, **NU** *ratio* is a required parameter of the COMPEN command.

GAIN *corr* - Real or Integer array or a number containing values by which the readings in *bridge_volts* are divided. If a number is specified, the value is divided into each of the readings in *bridge_volts*. If an array is specified, there is a 1-for-1 correspondence between the correction array and *bridge_volts*. If necessary, the correction array will wraparound and the correction factors will be used again.

INTO *name* - Array in which the converted readings are stored.

fmt - Data format in which the converted readings are returned. The default format is RASC.

www.valuetronics.com

Using the COMPEN Command

In the following program, 30 voltage measurements are made using a T type thermocouple. The COMPEN command is then used to convert the voltages to corresponding temperatures.

```
10   !Reset the HP 3852A and turn off the display to increase
20   !command execution speed.
30   !
40   OUTPUT 709;"RST"
50   OUTPUT 709;"DISP OFF"
60   !
70   !Download the measurement channel list into a mainframe array.
80   !Ten measurements are to be taken on each channel (200, 201,
90   !202) in the list.
100  !
110  OUTPUT 709;"SUB LOADLIST"
120  OUTPUT 709;"   INTEGER C,CH_LIST(19)"
130  OUTPUT 709;"   FOR C=0 TO 9"
140  OUTPUT 709;"      VWRITE CH_LIST 200,-202"
150  OUTPUT 709;"   NEXT C"
160  OUTPUT 709;"SUBEND"
170  OUTPUT 709;"CALL LOADLIST"
180  !
190  !Perform the measurements using the T type thermocouple. Measure
200  !the isothermal block for each channel. Specify a DC voltage
210  !measurement and store the readings in a packed array for increased
220  !measurement speed.
230  !
240  OUTPUT 709;"PACKED VOLT_RGS(119)"
250  OUTPUT 709;"REAL T(2)"
260  OUTPUT 709;"CONFMEAS REFT 200-202 USE 700 INTO T(0)"
270  OUTPUT 709;"CONFMEAS DCV CH_LIST USE 700 INTO VOLT_RGS(0)"
280  !
290  !Convert the voltage measurements to temperatures once all
300  !measurements have been made.
310  !
320  OUTPUT 709;"REAL COMPRDGS(29)"
330  OUTPUT 709;"COMPEN TEMPT VOLT_RGS REF T INTO COMPRDGS(0)"
340  !
350  !Read the converted readings into the output buffer. Enter and
360  !display those readings on the controller.
370  !
380  OUTPUT 709;"VREAD COMPRDGS"
390  REAL Rdgs(0:9,0:2)
400  ENTER 709;Rdgs(*)
410  PRINT USING "10(3(MDD.5D,3X),/)";Rdgs(*)
420  END
```


www.valuetronics.com

A typical output based on the program is shown below:

29.37598	31.16699	30.10547
29.33789	31.19238	30.13965
29.34082	31.23633	30.16211
29.35547	31.25293	30.16895
29.37793	31.26563	30.20313
29.35352	31.30859	30.22559
29.39160	31.30371	30.24219
29.42871	31.31055	30.24414
29.43750	31.33008	30.28809
29.44336	31.34570	30.29004

The Command Reference Manual contains additional examples on the use of the COMPEN command.

www.valuetronics.com

Contents

www.valuetronics.com

Chapter 11

HP 3852A Multitasking

Introduction

In a typical data acquisition system, data may be acquired or monitored from several sources and many devices may be under HP 3852A control. A single program to accomplish multiple data acquisition and control tasks would be long and complex, and not easily modified. With the multitasking and real-time interrupt capability of the HP 3852A, you can avoid a single, complex program by developing subroutines to handle each task and then running those subroutines concurrently.

This chapter explains how multitasking is achieved with the HP 3852A. The chapter describes the operation of a multitasking system and shows you how to integrate subroutines (or any set of commands) into a multitasking system using the multitasking commands available. Several examples are included at the end of this chapter to demonstrate the power and flexibility of this capability.

Note that multitasking is only available with HP 3852As with firmware revision 3.0 or greater. The IDN? command can be used to determine the firmware revision of your particular instrument.

www.valuetronics.com

Description of Operation

Operation of the HP 3852A's multitasking feature can easily be described by drawing an analogy between the multitasking process and the procedures at the checkout stand of a grocery store.

A Grocery Store

Figure 11-1 shows the checkout stands used to make the analogy. Shown in the figure are the components found at the checkout stands and beneath them, the multitasking terms they most closely associate with. For this particular stand, the following conditions exist:

- There are four checkout aisles.
- One checkout clerk (Y) will manage all four registers.
- Customers (X) are distributed among each of the four aisles.
- The clerk will operate each register for a period of three minutes. When the period expires, the clerk will move to the next register. However, if the clerk is checking out an item when the time expires, he must finish with that item. Also, the clerk may or may not have checked out all of the customer's items in the three minute period.
- Price checks may be necessary at various points in time.
- Customers may request that the clerk checkout a particular group of items before moving, even though the three minute period has expired.

www.valuetronics.com

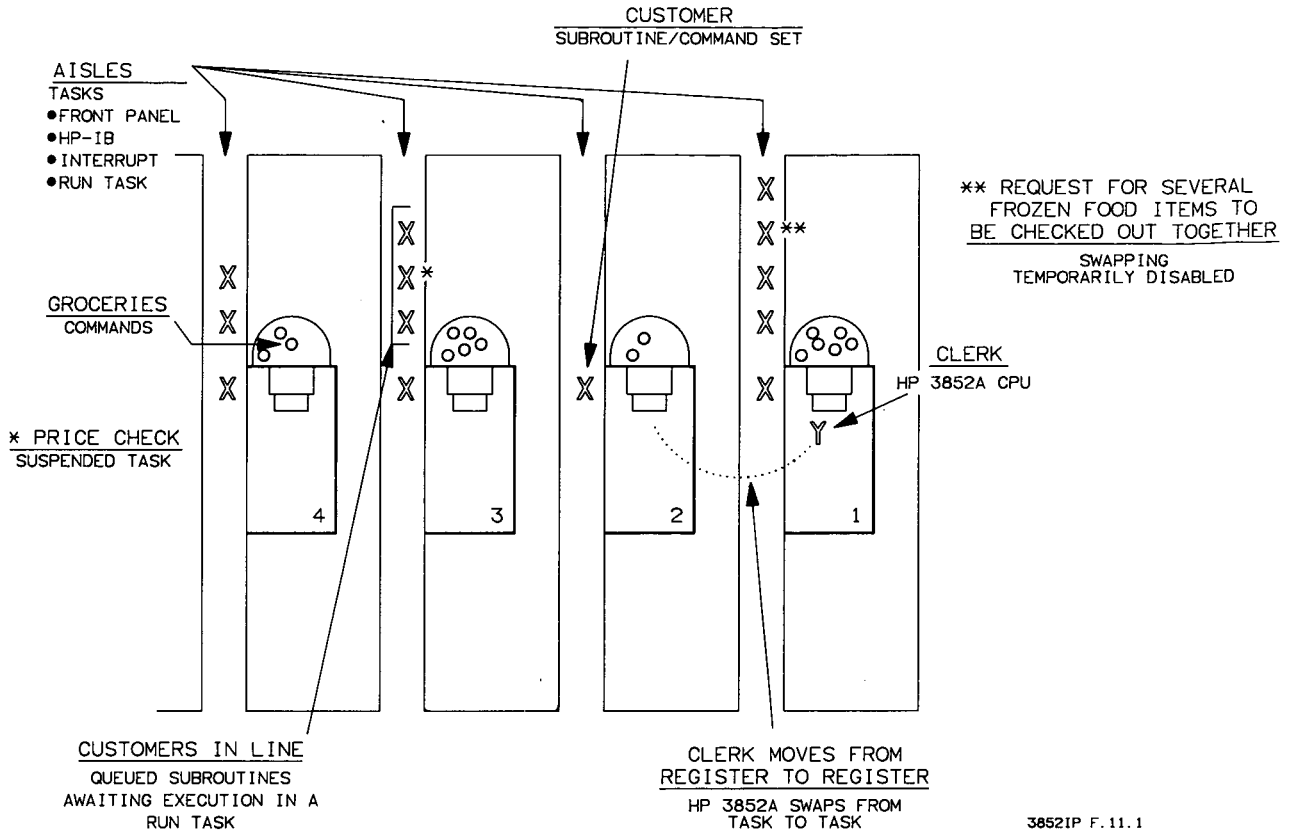


Figure 11-1. Grocery Store Analogy

We can associate HP 3852A multitasking with the checkout stand by describing the operation of the checkout stand as follows.

Checking Out The clerk begins checking out the food items for the customer first in line in aisle 1. When the three minute period expires, the clerk moves to aisle 2 and begins checking out items in a similar manner. When the period expires, the clerk moves to aisle 3 and the process is repeated as the clerk moves to aisle 4, and then back to aisle 1.

www.valuetronics.com

After cycling through the aisles a number of times, say for example, the customer at the front of aisle 1 and the only customer in aisle 2 finish. At this point, the customer that was second in line in aisle 1 moves up and since aisle 2 is now empty, the clerk moves between aisles 1, 3, and 4.

Price Check

Assume that while the clerk is at aisle 3 a price check is required. At this point, the clerk stops checking items at the register. Rather than waiting for the price to be called up from the back of the store, the clerk moves on to aisle 4, then aisle 1, and so on. If the clerk happens to be at aisle 4 when the price of the item is received, the clerk will move to aisle 1 before resuming with aisle 3.

Frozen Food

In another instance, a customer in aisle 1 has several frozen food items he wants checked out and placed in bags before they melt. However, given the number of items, the three minute period would expire before all of the desired items were checked. By notifying the clerk, the customer can prevent the clerk from moving until these items are checked. When the clerk is again allowed to move, the clerk moves to the next aisle for the three minute period and operation continues as before.

Definition of Terms

Figure 11-1 uses components of checkout stands to represent the terms/functions associated with a multitasking system. This section describes those terms in detail and introduces other functional aspects of HP 3852A multitasking.

Tasks

In a grocery store, customers go to aisles to checkout or to wait to checkout. In the HP 3852A multitasking system, commands and subroutines are directed to, and execute within environments called tasks. Specifically, there are four tasks within the HP 3852A's multitasking system:

Front Panel
HP-IB
Interrupt
Run Tasks

Front Panel Task

Commands entered from the mainframe's keyboard are said to execute within the front panel task. If the CALL command used to execute a subroutine is entered from the front panel, the subroutine executes within the front panel task.

HP-IB Task

Commands sent over the HP-IB execute within the HP-IB task. If the CALL command is sent over the HP-IB, the subroutine executes within the HP-IB task.

www.valuetronics.com

Interrupt Task A subroutine executed by the ON event CALL name command is called in response to an interrupt that has occurred. This subroutine is said to execute in the Interrupt task. Interrupts within the HP 3852A which can call subroutines are:

Alarms
Backplane (accessory) Interrupts
Limit Exceptions

Unlike subroutines executed by the CALL command, subroutines called in response to an interrupt execute within the interrupt task regardless of the task that executed the ON event CALL name command.

Run Tasks Run tasks execute subroutines that have been directed to particular run task numbers. In a multitasking system, you can create up to eight run tasks and have a subroutine execute within each run task. In addition, you can have up to 20 subroutine names in the program queue to execute in a single run task or which are distributed among the eight run tasks (Figure 11-2). The run tasks and the queue are analogous to the aisles at the checkout stand, with customers checking out or waiting to checkout (Figure 11-2).

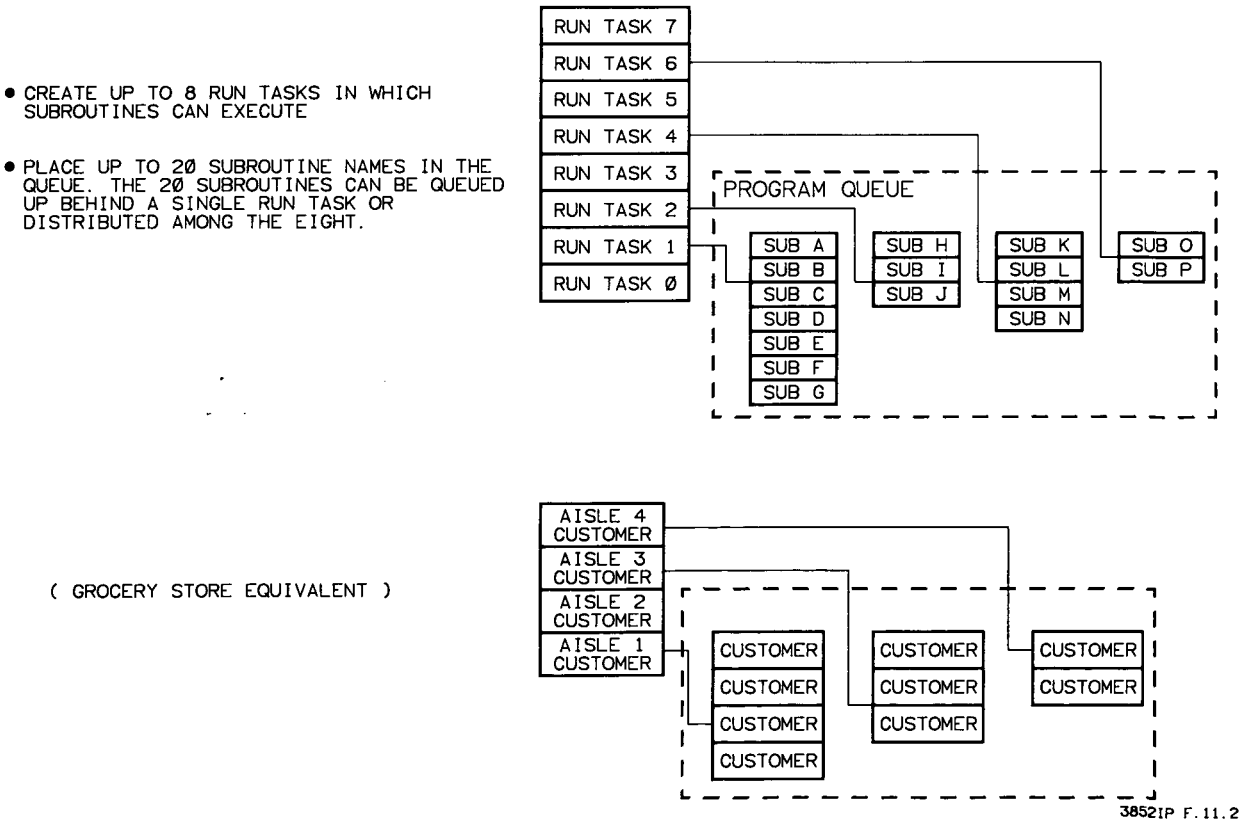


Figure 11-2. The Program Queue

www.valuetronics.com

Active, Scheduled, and Queued Tasks

In a multitasking system, a subroutine directed to a run task is either active, scheduled, or queued. For example, in Figure 11-3, subroutines A, B, C, and D have been directed to run tasks. While subroutine A is executing, it is termed "active", while subroutines C and D are "scheduled". When the mainframe swaps to run task 1, subroutine C becomes active while A and D are scheduled, and so on. Subroutine B is a "queued" subroutine behind subroutine A. Subroutine B will remain in the queue until A completes or is aborted.

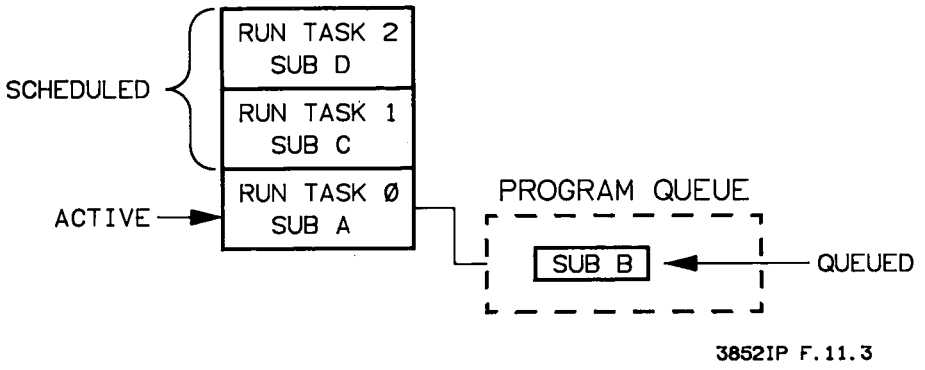


Figure 11-3. Active, Scheduled, and Queued Tasks

It should be noted that with HP 3852A multitasking, only one task is ever active (executing) at a time. This also includes the front panel, HP-IB, and Interrupt tasks. The HP 3852A swaps from task to task, but commands within multiple tasks never execute simultaneously.

Suspended Tasks

At one point in the store, the clerk stopped checking out food items pending a price check. However, rather than remaining idle, the clerk moved on to the next aisle to continue checking out items. With HP 3852A multitasking, command execution within tasks can be suspended. When a task is suspended, the mainframe continues command execution within other tasks. When the task is no longer suspended, command execution resumes.

Task Priorities

When the HP 3852A enters the multitasking mode, the tasks (environments) are assigned priorities which subsequently can be changed by the user. The priorities set are:

Front Panel	25	(highest)
HP-IB	45	
Interrupt	65	
Run Tasks	85	(lowest)

www.valuetronics.com

A command or subroutine from a high priority task will preempt a subroutine (or set of commands) within a lower priority task and execute to completion before allowing execution of the lower priority subroutine to resume. However, commands within high priority tasks only preempt after completion of the currently executing command.

Although upon entering the multitasking mode the priority of the run tasks are set the same (85), the priority of each run task can be set individually.

Time-Slicing

All tasks (Front Panel, HP-IB, Interrupt, and Run Tasks) which have equal priority (at any priority level) will time-slice (Figure 11-4). In the example of the grocery store, each aisle can be considered as having the same priority. Thus, the clerk spent the same amount of time at each aisle before moving to the next aisle. In a multitasking system, the user specifies the period of time the mainframe will execute commands within a task before swapping to the next task.

Most often, the tasks you want to time-slice are the run tasks. By setting several run tasks to the same priority (a priority below that of other tasks), the run tasks can, in effect, run in the background. This enables commands from higher priority tasks to execute on demand. Also, the time-slicing between these equal priority tasks helps ensure that all run task subroutines execute.

www.valuetronics.com

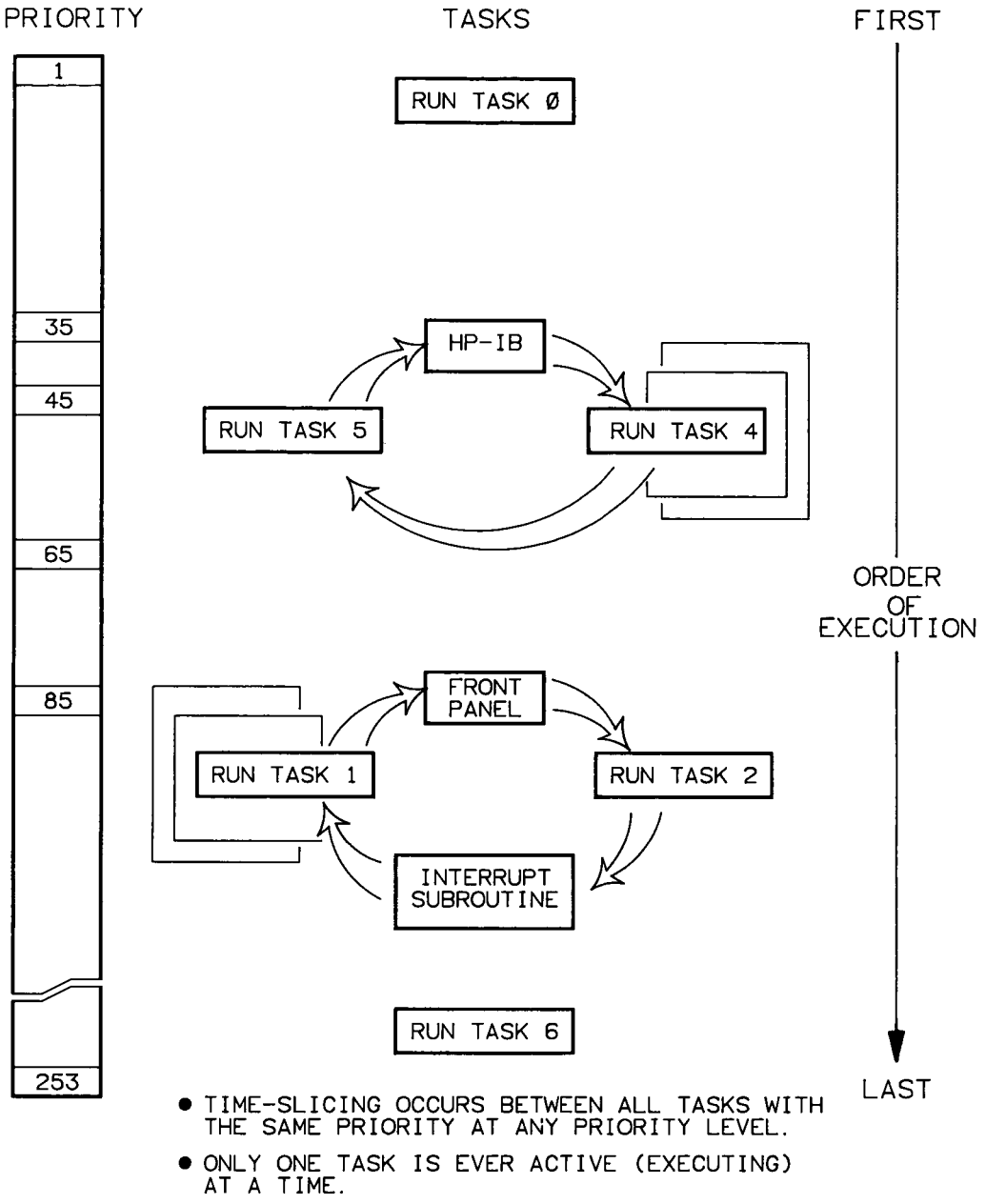


Figure 11-4. Time-Slicing Among Tasks

Swapping

Swapping is when the mainframe switches from executing commands within one task to executing commands within another task. As the swap is made, the mainframe stores the status (i.e. program counter location) of the task it swaps from. Thus, when the mainframe swaps back to the task, execution resumes where it left off.

www.valuetronics.com

In the HP 3852A's multitasking system, the mainframe swaps to another task when the time-slice period expires or when a command within a higher priority task is received. However, the swapping is end of command swapping which means that if the mainframe is executing a command when the time-slice period expires or when the command is received, the swap does not occur until command completion. Should a subroutine finish before the time-slice period expires, the mainframe swaps at that time rather than waiting for the period to elapse.

Enabling/Disabling Swapping

During operation of the grocery store, a customer requested that the clerk check out several frozen food items before swapping to the next register, even though the three minute period expired. Likewise, with HP 3852A multitasking, swapping can be disabled to ensure that specific commands execute together while a task is currently active. This feature is useful when, for example, one accessory is used in multiple applications (e.g. a voltmeter). By disabling swapping temporarily, specific configurations can be programmed and remain set throughout the measurements. If swapping were not disabled, a swap could occur and change the configuration thus invalidating the measurements.

Subroutines

Analogous to food items and the customer, commands which are executed as the mainframe swaps from task to task most often reside in subroutines. Subroutines enable you to divide your data acquisition and control tasks into separate modules (Figure 11-5). Since each module is handled by a subroutine, the subroutine is generally much smaller than one mainline program making debugging and modification easier. Also, once downloaded into mainframe memory, subroutines can be repeatedly called upon to execute within any task environment.

Real-Time Interrupts

When the HP 3852A is not in the multitasking mode, a subroutine must execute to completion before an interrupt will be serviced or before commands from any other origin can execute. However, in the multitasking mode, a currently executing subroutine can be interrupted or preempted following command execution. This is referred to as a real-time interrupt.

www.valuetronics.com

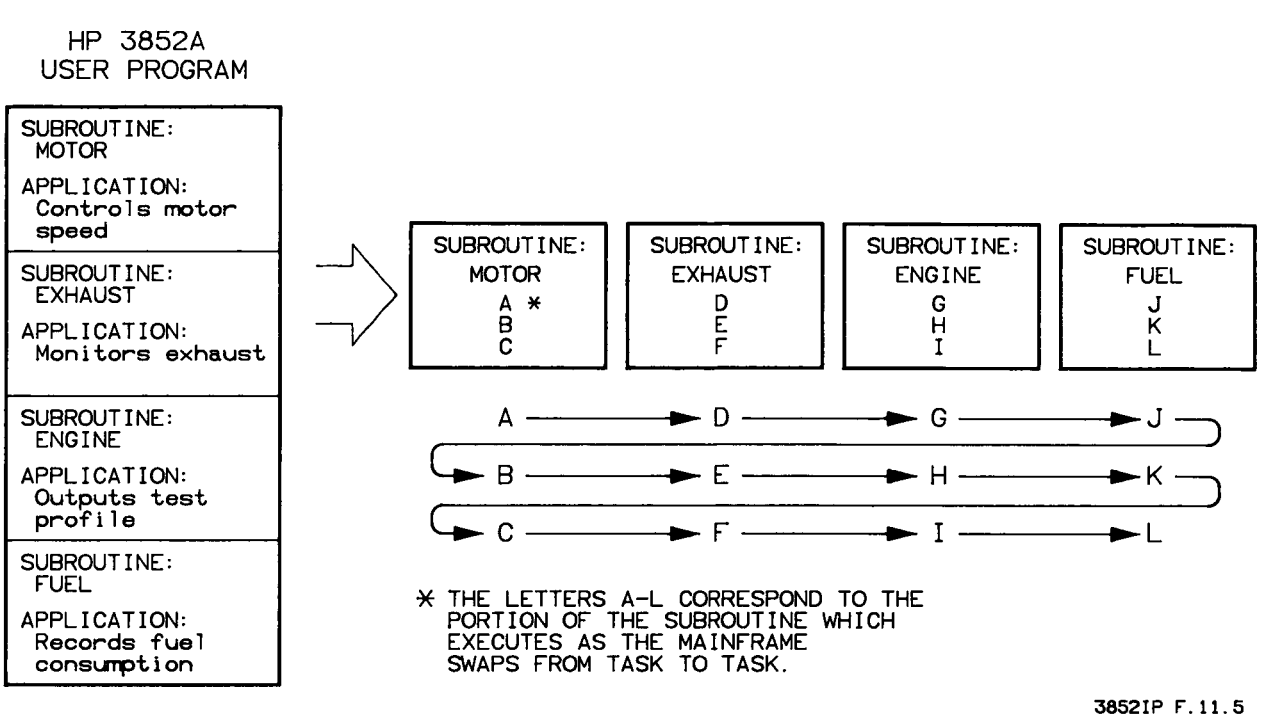


Figure 11-5. Development and Execution of Application Subroutines

System Characteristics

System Characteristics are typical, non-warranted performance parameters of HP 3852A multitasking systems. Their intent is to provide information that is helpful when the multitasking feature is used. A list of these characteristics is found in Table 11-1. Note that parameters, limitations, ranges, etc., of the multitasking commands are given in the next section where these commands are described.

Table 11-1. Multitasking System Characteristics

Characteristic	Period
Minimum Time-Slice	1.024 msec
Resolution of Time-Slice	1.024 msec
Time to swap from one task to another	475 usec
Time to direct a subroutine to a run task and start execution of the subroutine	1 msec
Time from backplane interrupt to execution of subroutine (no other tasks active)	1.2 msec
Time from alarm or limit interrupt to execution of subroutine (no other tasks active)	< 1 msec

www.valuetronics.com

Modes of Operation

Before discussing the HP 3852A's multitasking commands, it is important to note that there are two modes of operation associated with the HP 3852A. These modes can be described as the power-on mode and the multitasking mode. In the power-on mode, the HP 3852A operates the same as previous versions of the instrument. In the multitasking mode, many rules of operation change, plus most of the commands covered in the following paragraphs are allowed only in this mode. Table 11-2 lists the key differences between the modes of operation.

Table 11-2. Power-On Vs. Multitasking Modes of Operation

Power On Mode	Multitasking Mode
Interrupts are handled only at the end of the currently executing subroutine.	Interrupts can be handled at the end of the currently executing command
Subroutines execute to completion before another subroutine starts.	Subroutines execute until the time-slice period is up or until a higher priority task interrupts.
No tasks.	Up to eight run tasks can be designated.
No program queue.	Up to 20 subroutine names can be placed in the queue.
No time-slicing.	All tasks of equal priority will time-slice.
System defined priorities.	User-defined priorities from 1 to 253.
USE channel is global.	The USE channel set by the USE command is local to the task from which the USE command was executed.

Using the Multitasking Commands

This section identifies the HP 3852A's multitasking commands and shows you how to use these commands to integrate subroutines and commands from the various tasks into multitasking systems.

The commands and procedures described are in the sequence that is recommended for setting up your multitasking system. However, as you become more familiar with the multitasking capability, it becomes obvious that the power and flexibility of this feature provides many valid solutions.

Note that summary information on each multitasking command can be found in the Command Reference Manual. An example program emphasing the command is part of each command reference entry.

Command Summary

The HP 3852A's multitasking commands can be divided into five functional groups: system configuration, prioritizing, task synchronization, activation, and system status. Table 11-3 provides a brief description of the commands within each group.

www.valuetronics.com

Table 11-3. HP 3852A Multitasking Command Summary

System Configuration	
Command	**Description**
TSLICE	Sets the time-slice period.
NTASKS	Specifies the number of run tasks and sets the size of the program queue.
NLOCKS	Specifies the number of locks that can be requested within a multitasking system.
ENABLE MULTI	Places the HP 3852A in the multitasking mode.
DISABLE MULTI	Removes the HP 3852A from the multitasking mode.
Prioritizing	
Command	**Description**
URGENCY	Sets task priorities within the HP 3852A multitasking system.
CREATE RUN	Sets the priority of a run task environment before a subroutine is directed to that task.
Task Synchronization	
Command	**Description**
ENABLE EOL SWAP	Enables end of command swapping.
DISABLE EOL SWAP	Disables end of command swapping.
REQUEST/RELEASE	Dedicates resources (e.g. voltmeters, arrays) to the task that requests a lock.
SUSPEND	Suspends or defers the execution of commands/subroutines within a task for a specified number of seconds.
SUSPEND UNTIL	Suspends or defers the execution of commands/subroutines within a task until a specified Julian date and time.
WAITFOR SIGNAL	Suspends execution of commands/subroutines within a task until a signal (SIGNAL command) is received.
SIGNAL	Signals the task suspended by WAITFOR SIGNAL to resume command/subroutine execution.
PAUSE	Pauses HP 3852A subroutine execution.
CONT	Continues a paused (PAUSE command) HP 3852A subroutine.
Activation	
Command	**Description**
RUN	Directs a subroutine to a run task and specifies the number of times and how often the subroutine is to execute.
ON ... RUN	Directs a subroutine to a run task when a backplane, alarm, or limit interrupt occurs.
System Status	
Command	**Description**
RUN?	Returns the operating status of run task subroutines in a multitasking system.
ABORT	Aborts the currently executing subroutine called/activated by the specified task.
DISABLE PROBE	Disables the operating system probe.
ENABLE PROBE	Activates the probe.
PROBE	Designates the arrays which will store the data returned by the probe.

www.valuetronics.com

System Configuration

The first step in developing a multitasking program is to configure the system. This involves setting a time-slice period, setting the number of run task environments and the size of the program queue, specifying the number of locks which can be requested, and placing the HP 3852A in the multitasking mode. This process is achieved with the TSLICE, NTASKS, NLOCKS, and ENABLE MULTI commands.

The system configuration parameters are stored in the mainframe's non-volatile memory. Thus, once the desired configuration is set, it does not have to be re-loaded each time you execute or modify your multitasking program.

Setting the Time-Slice Period

The time-slice period is the amount of time commands will execute within a task before the system swaps to another task of equal priority. Should the time-slice period expire while a command is executing, the command is allowed to finish since swapping occurs only on command completion. If the commands within a task complete before the time-slice period expires, a swap is made on completion of the last command. The system does not wait for the period to elapse. If commands within a higher priority task are entered in the middle of the time-slice period, the high priority commands execute and then command execution within the lower priority task resumes for the remainder of the period. Figure 11-6 shows the conditions under which the time-slice period is exited.

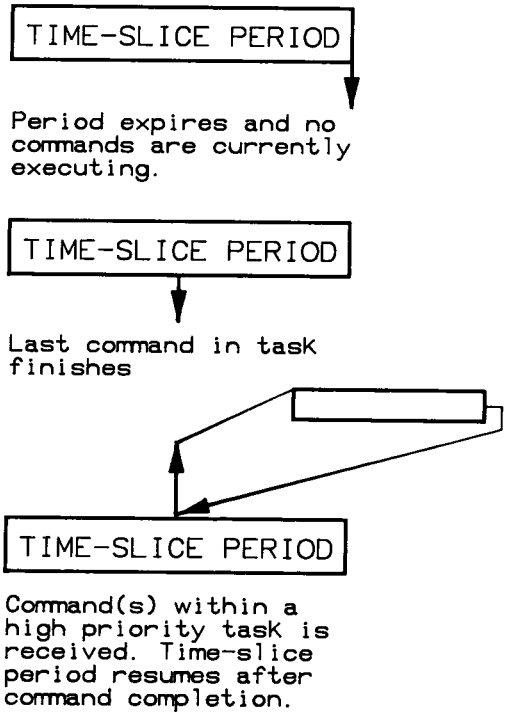


Figure 11-6. Exiting the Time-Slice Period

www.valuetronics.com

The command used to set the time-slice period is the TSLICE command. The syntax of the command and a description of its parameters are given below:

TSLICE *seconds* [*tic_interval*[*INT/EXT*]]

seconds - time-slice period. The range for *seconds* is 0 to 16.71 seconds. To obtain time-slice periods less than 65.536 ms, the tic interval parameter must be set equal to, or faster than time-slice period.

tic_interval - interval at which the HP 3852A operating system checks the state of the multitasking system. At each interval, the operating system determines whether a task is to be swapped, a suspended task is to be activated, a task is to begin, or whether operation is to continue in the current state.

The brackets around *tic_interval* indicate the parameter is optional. If a tic interval is not specified, a default interval of 65.536 ms is used. As a result, the parameter is specified only when time-slice periods less than 65.536 ms are required. Table 11-4 lists the tic_intervals available (from the internal clock) and the corresponding minimum and maximum time-slice periods.

Table 11-4. Tic_Intervals and Time-Slice Periods (Internal Clock)

TIC_INTERVAL	MINIMUM TIME-SLICE	MAXIMUM TIME-SLICE
1.024 ms	1.024 ms	0.261 s
2.048 ms	2.048 ms	0.522 s
4.096 ms	4.096 ms	1.044 s
8.192 ms	8.192 ms	2.088 s
16.384 ms	16.384 ms	4.177 s
32.768 ms	32.768 ms	8.355 s
65.536 ms	65.536 ms	16.71 s

INT/EXT - source of the clock from which the tic interval is derived. *INT* selects the mainframe's internal clock. *EXT* selects an external clock applied through the CHANNEL ADVANCE BNC on the HP 3852A's rear panel. The default source is *INT*. If *EXT* is selected, recall that an external clock determines the interval at which the operating system checks the multitasking system. Thus, the time-slice period should be set accordingly (i.e the same as the tic interval or a multiple of the tic interval).

Loading the Time-Slice Period

The time-slice period set by TSLICE is not actually used until it is loaded into the mainframe's operating system. This is done by either resetting the mainframe or cycling power after you execute the TSLICE command. Two examples of loading the time-slice period are given below:

```
10 OUTPUT 709;"TSLICE .065"
20 OUTPUT 709;"RST"
```

or

```
10 OUTPUT 709;"TSLICE .065"
20 OUTPUT 709;"ENABLE MULTI"
```


www.valuetronics.com

In the first segment, the reset which loads the time-slice period is caused by the RST command. In the second segment, the reset caused by the execution of ENABLE MULTI loads the period. (Note in both segments only the "seconds" parameter is specified, thus the default *tic_interval* is used.)

Once in the operating system, the time-slice period is stored in non-volatile memory. The period need only be reloaded when it is changed.

Resetting the instrument and cycling power erases all variables, arrays, and downloaded subroutines in mainframe memory. Thus, the desired time-slice period should be specified and loaded before memory is used for your multitasking applications.

Determining a Time-Slice Period

When setting a time-slice period, it is important to select a period small enough to allow time-slicing to occur. For example, if the time-slice period is relatively large compared to the execution times of commands in your tasks, the commands may finish before the time-slice period expires. As a result, the tasks execute in sequence and the benefits of multitasking are not achieved.

In applications where various time-slice periods are desired, it is often helpful to know the execution times of the commands or subroutines in your tasks. Knowing these times can help you estimate the portion of a task that will execute during the time-slice period(s) you select.

The execution time of a command or subroutine can be determined by reading the mainframe's real-time clock immediately before and immediately after command (or subroutine) execution. The difference in these times is the execution time.

Timing a Command

```
10   OUTPUT 709; "REAL T0,T1,T2,DCRDGS (9)"
20   OUTPUT 709; "SUB TMEAS"
30   OUTPUT 709; "  DISP OFF"
40   OUTPUT 709; "  TIME INTO T0"
50   OUTPUT 709; "  CONFMEAS DCV 600-609 USE 700 INTO DCRDGS"
60   OUTPUT 709; "  TIME INTO T1"
70   OUTPUT 709; "  DISP ON"
80   OUTPUT 709; "SUBEND"
90   OUTPUT 709; "CALL TMEAS"
100  OUTPUT 709; "T2=T1-T0"
110  OUTPUT 709; "VREAD T2"
120  ENTER 709;A
130  PRINT A
140  END
```


www.valuetronics.com

In this program we are timing the execution of the CONFMEAS command in line 50. Therefore, readings of the real-time clock are taken in lines 40 and 60. The difference calculated in line 100 is the time required for the command to execute. Note that the command timed is within a subroutine. Commands executed within a subroutine and with the display off (line 30) execute at faster rates. This is due to the fact that the commands are already compiled when the subroutine is downloaded and with the display off, neither commands or data are displayed. The rate at which data is sent to the output buffer is also increased when the display is off.

Timing a Subroutine

```
10   OUTPUT 709;"REAL T0,T1,T2"
20   OUTPUT 709;"DISP OFF"
30   OUTPUT 709;"SUB A"
40   OUTPUT 709;" INTEGER I"
50   OUTPUT 709;" FOR I=0 to 9"
60   OUTPUT 709;" DISP I"
70   OUTPUT 709;" NEXT I"
80   OUTPUT 709;"SUBEND"
90   OUTPUT 709;"SUB B"
100  OUTPUT 709;" TIME INTO T0"
110  OUTPUT 709;" CALL A"
120  OUTPUT 709;" TIME INTO T1"
130  OUTPUT 709;"SUBEND"
140  OUTPUT 709;"CALL B"
150  OUTPUT 709;"T2=T1-T0"
160  OUTPUT 709;"VREAD T2"
170  OUTPUT 709;"DISP ON"
180  ENTER 709;A
190  PRINT A
200  END
```

A subroutine is timed by reading the mainframe's clock (line 100) immediately before execution of the subroutine and again immediately after (line 120). The difference calculated in line 150 is the time required for the subroutine to run to completion.

Related Errors

The only error which may occur upon execution of the TSLICE command is Error 24: Argument Out Of Range, which is caused by specifying a time-slice period or *tic_interval* outside of the allowable range.

Setting the Number of Run Tasks and the Queue Size

A run task is an environment to which a subroutine is directed and executed. Since only one subroutine at a time can execute in a specific run task, the names of multiple subroutines directed to the same run task number are held in a program queue.

www.valuetronics.com

The command used to set the number of run task environments and the size of the queue is the NTASKS command. The syntax of the command and a description of the parameters are given below:

NTASKS *number* [*size*]

number - the the number of run task environments the system is to allow. For an HP 3852A multitasking system, the maximum number of run task environments allowed is eight and the range for *number* is 0 to 8.

size - size of the program queue. The maximum queue size is 20 and the range for *size* is 0 to 20. The brackets around *size* indicate that the parameter is optional. If size is not specified, the size of the queue is set equal to the number of run tasks.

Note that a queue size is specified even though no subroutine names may be held. The reason for this is that although a subroutine may be sent "directly" to a run task, the name of the subroutine passes through the queue first. A standard rule of thumb when selecting a queue size is one queue space per run task environment. If memory allows, a queue size of 20 should be specified in order to accommodate any future expansion of your system.

Loading the NTASKS Parameters

Along with the time-slice period, the number of run tasks and the size of the queue do not become effective until these parameters are loaded into the mainframe's operating system. This is done by either resetting the instrument or cycling power after you execute the NTASKS command. Two examples of loading the NTASKS parameters are given below:

```
10  OUTPUT 709;"NTASKS 2,2"
20  OUTPUT 709;"RST"
```

or

```
10  OUTPUT 709;"TSLICE .065"
20  OUTPUT 709;"NTASKS 2,2"
30  OUTPUT 709;"ENABLE MULTI"
```

In the first segment, the reset which loads the NTASKS parameters is caused by the RST command. In the second segment, the reset caused by the execution of ENABLE MULTI loads the parameters. Note that by executing TSLICE and NTASKS prior to ENABLE MULTI, a single reset can be used to load the parameters of both commands.

The number of run tasks and the size of the queue are also stored in non-volatile memory. Thus, these parameters need only be reloaded when they are changed.

Again, resetting the instrument and cycling power erases all variables, arrays, and downloaded subroutines in mainframe memory. Therefore, the desired number of run tasks and queue size should be set and loaded before memory is used for your multitasking applications.

www.valuetronics.com

System Overhead

By "forcing" you to specify the number of run task environments you plan to use and the queue size you need, you minimize the amount of overhead required to manage your multitasking system.

Each run task specified by the NTASKS command uses 160 bytes of overhead. When the run task is activated by the RUN command, another 1216 bytes of overhead are required. The amount of memory allocated for the queue is the queue size specified times 24 bytes.

Since memory used for overhead is no longer available to the user, it is often helpful to determine the total amount of memory that will be allocated before the system is downloaded. This can be accomplished with the formula:

```
N1*160 + N2*1216 + N3*24
```

where:

```
N1 = number of run tasks specified by NTASKS
N2 = number of run tasks activated by the RUN command
N3 = size of the queue specified by NTASKS
```

As an example, assume the NTASKS command is executed as follows and that two RUN commands are also executed:

```
 10   OUTPUT 709;"TSLICE .065"
 20   OUTPUT 709;"NTASKS 2,2"
 30   OUTPUT 709;"ENABLE MULTI"
  .
  .
  .
250   OUTPUT 709;"RUN 1 A"
260   OUTPUT 709;"RUN 2 B"
```

As shown in line 20, the number of run tasks in the system is 2 and the size of the queue is also 2. These numbers correspond to N1 and N3 in the formula. Since two RUN commands are used, N2 is also 2. Thus, the formula with values specified for N1, N2, and N3 appears as shown below:

```
2*160 + 2*1216 + 2*24 = 2800 bytes of overhead
```

Therefore, 2800 bytes of memory are used by the multitasking system in addition to the memory used by subroutines and by any variables or arrays declared in the subroutines.

www.valuetronics.com

Related Errors

The errors associated with the NTASKS command are:

ERROR 23: SETTINGS CONFLICT - one or more run tasks is specified together with a queue size of 0.

ERROR 24: ARGUMENT OUT OF RANGE - the number of run tasks or the queue size specified is outside of the allowable range.

Setting the Number of Locks

A third (and optional) step of the system configuration process is setting up the number of locks in the system using the NLOCKS command. A lock is a method of dedicating resources such as a voltmeter or an array to a specific task.

Since locks are a tool for the advanced user and are optional, the commands associated with them (NLOCKS, REQUEST/RELEASE) are covered only in the Command Reference Manual.

Entering the Multitasking Mode

The HP 3852A is placed in the multitasking mode upon execution of the ENABLE MULTI command. The syntax of the command is:

ENABLE MULTI

When ENABLE MULTI is executed, a system reset occurs. The reset returns each plug-in accessory to its power-on state and erases all declared variables, arrays, and downloaded subroutines in mainframe memory. The reset also clears the mainframe's input and output buffers. Since a reset is required to load the time-slice period, the number of run tasks and queue size, (and the number of locks) into the operating system, TSLICE and NTASKS (and NLOCKS) are often executed prior to ENABLE MULTI.

Mainframe Operation

Certain operating parameters of the mainframe function differently in the multitasking mode as opposed to the power-on mode. Specifically, these parameters are the execution of commands within task environments, the USE channel, and the handling of interrupts.

Tasks

In the multitasking mode, commands and subroutines execute within environments called tasks. The tasks established upon execution of ENABLE MULTI are:

Front Panel
HP-IB
Interrupt
Run Tasks

Descriptions of these tasks can be found under "Definition of Terms" which is in a previous section of this chapter.

The USE Channel

In the multitasking mode, the USE channel specified by the USE command is local to the task. This means a USE channel specified from a particular task applies only to the commands in that task. Commands within other tasks which do not specify a USE channel when required will use the default USE channel set at power-on or following a reset.

www.valuetronics.com

Real-Time Interrupts

If a subroutine is executing in the power-on mode and an interrupt occurs, the interrupt is "recorded", however, it is not serviced until the subroutine finishes executing. If a subroutine is executing in the multitasking mode and an interrupt occurs, execution of the subroutine is suspended while the interrupt is serviced and handled (Chapter 8). Once the handling routine completes, execution of the suspended subroutine resumes. This type of interrupt is a real-time interrupt since only completion of the currently executing command, not completion of the subroutine, is required before the interrupt is handled.

Exiting the Multitasking Mode

Once ENABLE MULTI is executed, the HP 3852A remains in the multitasking mode until power is cycled, RST HARD is executed, or until DISABLE MULTI is executed. The syntax of the command is:

DISABLE MULTI

When executed, DISABLE MULTI also causes a system reset. As a result, any arrays, variables, and subroutines in mainframe memory are erased as the mainframe enters the power-on mode.

Task Priorities

In the multitasking mode, the task environments are assigned priorities. A task's priority determines when its commands will execute relative to commands in other tasks, and whether or not the task will time-slice with other tasks. This section explains the priority scale of the multitasking mode and how priorities are set.

The Priority Scale

Upon entering the multitasking mode (ENABLE MULTI), the task priorities are set as follows:

Front Panel	(25)
HP-IB	(45)
Interrupt	(65)
Run Tasks	(85)

The scale with these particular priority settings is shown in Figure 11-7. This figure shows that the lower the priority number, the higher the task's priority. Thus, if commands are entered from the front panel at the same time commands are received over the HP-IB, the front panel commands execute first because of the higher priority task. Similarly, if commands are entered from the front panel while a run task subroutine is executing, the commands preempt the subroutine and execute, then allow the subroutine to continue. Any tasks which have the same priority will time-slice. In

www.valuetronics.com

Figure 11-7, each of the run tasks are set to the same priority and therefore, time-slice. Note that if the priority of the front panel or HP-IB tasks, for example, was set the same as the run tasks, those tasks would time-slice also.

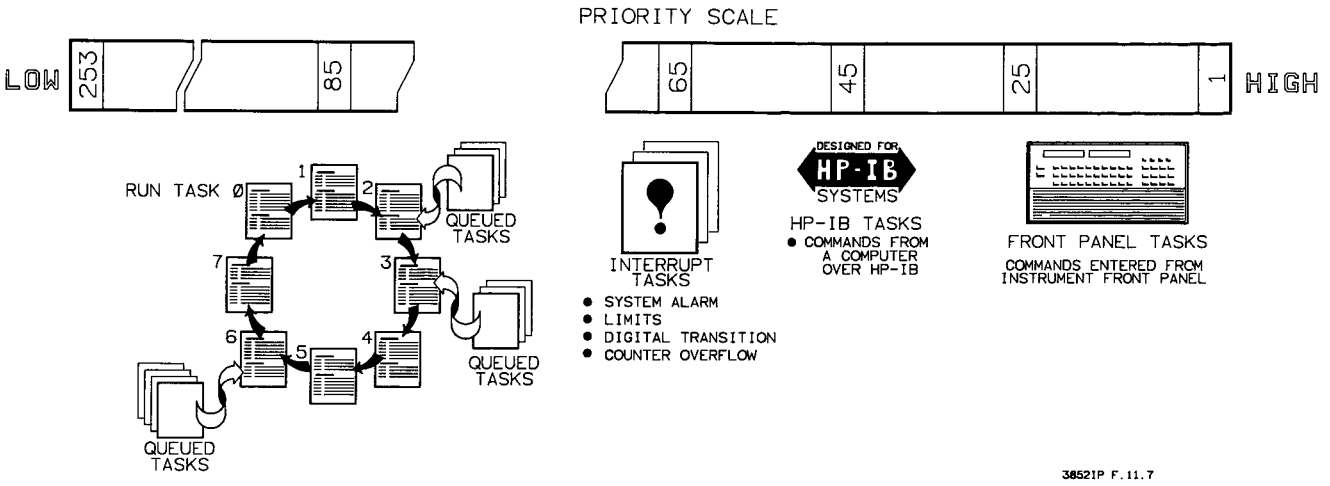


Figure 11-7. Multitasking Priority Scale

Setting Task Priorities

The task priorities set when the HP 3852A enters the multitasking mode are changed with the URGENCY command. The syntax of the command and a description of its parameters are given below:

URGENCY [*task*] *number*

task - environment whose priority is raised or lowered. A task can set the priority of any other task although if the *task* parameter is not specified, the priority set pertains to the environment in which the URGENCY command is executed. The environment choices for the *task* parameter are:

HPIB	Sets the priority for all commands and subroutines executed within the HP-IB task.
KYBD	Sets the priority for all commands and subroutines executed within the front panel task.
INTR	Sets the priority for all interrupt-called subroutines.
run task number	Sets the priority of the run task specified.

www.valuetronics.com

number - sets the priority of the task specified. The range for number is 1 to 253. In this range, 1 is the highest priority a task could have and 253 is the lowest.

Executing the URGENCY Command

Once in the multitasking mode, the URGENCY command can be executed within any task at any point in the program. When setting the priority of a run task, URGENCY can only be executed after the run task has been created by the RUN command.

URGENCY can be used at the beginning of a task to establish specific priorities which remain set throughout the program, or URGENCY may reside in the middle of a task to raise or lower the priority of the tasks in response to conditions that arise.

In the following program segment, the URGENCY command is executed within a run task to lower the task's priority raised previously by the URGENCY command in line 550 (HP-IB task).

```
350  OUTPUT 709;"SUB DCMEAS"
360  OUTPUT 709;" INTEGER K"
370  OUTPUT 709;" REAL DCRDGS (49)"
380  OUTPUT 709;" USE 700"
390  OUTPUT 709;" CONF DCV"
400  OUTPUT 709;" FOR K=0 TO 4"
410  OUTPUT 709;" MEAS DCV 600-609 INTO DCRDGS"
420  OUTPUT 709;" NEXT K"
430  OUTPUT 709;" URGENCY 85"
440  OUTPUT 709;"SUBEND"
  .
  .
  .
540  OUTPUT 709;"RUN 3 DCMEAS 2"
550  OUTPUT 709;"URGENCY 3 40"
```

Notice in line 430 of this segment that a task is not specified. Thus, the priority changed is that of run task 3 (subroutine DCMEAS).

Remote/Local States and Front Panel Priority

Regardless of the priority of commands in the front panel (KYBD) task, the HP 3852A must be in the LOCAL state before a command will be accepted. The HP 3852A enters the REMOTE state anytime a command is sent over the HP-IB (i.e. OUTPUT 709;"...). Refer to Chapter 5 for more information on the REMOTE/LOCAL states.

www.valuetronics.com

Run Task Priorities

Upon entering the multitasking mode, the priority of all run tasks is set the same (85). However, with the URGENCY command, only the priority of the run task specified is changed. The priority of the other run tasks remains as set by ENABLE MULTI or as set by subsequent URGENCY commands. Note that a subroutine in the queue will execute at the same priority as the run task subroutine it replaces.

Using CREATE RUN

In addition to URGENCY, the CREATE RUN command is also used to set the priority of a run task environment. CREATE RUN enables subroutines directed to run tasks to start execution faster since the urgency of the task can be set prior to the RUN command. The syntax and parameters of CREATE RUN are given below:

CREATE RUN *task_number* [*urgency*]

task_number - run task whose urgency is set. The range for *task_number* is 0 to 7.

urgency - urgency assigned to the specified run task. The range for *urgency* is 1 to 253. The default *urgency* is 85.

Notice CREATE RUN does not direct subroutines to run tasks nor does it initiate subroutine execution. CREATE RUN loads the specified (or default) urgency for the specified run task into the operating system so that it does not have to be done by the subsequent RUN command. This, as mentioned before, enables subroutine execution to start faster.

Additional examples on the use of the URGENCY command can be found in the "Multitasking Examples" section of this chapter. The Command Reference Manual also contains detailed examples on the use of URGENCY and CREATE RUN.

Interrupt Priorities

When the priority of interrupt-called subroutines is set, the priority in which interrupts are handled by the mainframe remains the same:

ALARM
BACKPLANE
LIMIT

This means that should these interrupts occur simultaneously, ON ALRM CALL *name* would occur first and run to completion followed by ON INTR CALL *name* then ON LMT CALL *name*. Note, however, that given the priorities set when the HP 3852A enters the multitasking mode, a subroutine called by any of these interrupt conditions would execute to completion before allowing the system to resume executing a subroutine in a lower priority run task.

www.valuetronics.com

Related Errors

Errors associated with the URGENCY and CREATE RUN commands are given below:

ERROR 24: ARGUMENT OUT OF RANGE - a parameter specified in either URGENCY or CREATE RUN is outside of the allowable range.

ERROR 84: RUN TASK DOES NOT EXIST - the run task number specified in the URGENCY command has not been created by the RUN command.

ERROR 86: MULTITASKING NOT ENABLED - URGENCY and CREATE RUN can only execute in the multitasking mode.

Task Synchronization

Task synchronization commands provide you with several methods for controlling the execution of commands within tasks. These commands can execute within any task, however; they are generally used in the subroutines which handle your applications. The commands covered in this section are ENABLE EOL SWAP, DISABLE EOL SWAP, SUSPEND, SUSPEND UNTIL, WAITFOR SIGNAL, SIGNAL, PAUSE, and CONT. REQUEST and RELEASE which are used with locks are also introduced. However, since they are intended for the advanced user, the discussion on the use of the commands is limited to the Command Reference Manual.

Swapping Between Tasks

When the time-slice period expires or a command within a higher priority task is received, the mainframe "swaps" to the next scheduled task or to the high priority task upon completion of the currently executing command. Swapping between tasks under these conditions is allowed with the ENABLE EOL SWAP command. The time required for the swap to occur is approximately 475 us. ENABLE EOL SWAP is set on execution of the ENABLE MULTI command.

End of command swapping is disabled with the DISABLE EOL SWAP command. Once swapping is disabled, the mainframe executes commands in the current task until the task finishes or until swapping is re-enabled.

Using the ENABLE/DISABLE EOL SWAP Commands

As one method for controlling the execution of commands within tasks, ENABLE EOL SWAP and DISABLE EOL SWAP are used in applications where commands need to execute together without being disrupted by a swap to an equal, or higher priority task. ENABLE and DISABLE EOL SWAP which enclose the commands to execute, have the syntax shown below:

ENABLE EOL SWAP

DISABLE EOL SWAP

www.valuetronics.com

An example of how these commands are used is given in the following program segment:

```
100  OUTPUT 709;"DISABLE EOL SWAP"
110  OUTPUT 709;"REAL DCRDGS(99)"
120  OUTPUT 709;"USE 700"
130  OUTPUT 709;"CONF DCV"
140  OUTPUT 709;" RANGE 10"
150  OUTPUT 709;" TERM EXT"
160  OUTPUT 709;" NRDGS 100"
170  OUTPUT 709;"TRIG SGL"
180  OUTPUT 709;"XRDGS 700,100 INTO DCRDGS"
190  OUTPUT 709;"ENABLE EOL SWAP"
```

In this segment, the voltmeter in slot 7 (line 120) is configured for a set of measurements on its rear panel terminals. By disabling swapping (line 100), the mainframe will execute lines 110/180 without swapping to another task of equal or higher priority. This ensures that the voltmeter's configuration remains set until the measurements are complete.

Nested Subroutines

If swapping has been disabled and the subroutine which is executing calls another subroutine, the mainframe will execute the called subroutine. Once that subroutine completes, the mainframe returns to the calling subroutine which then executes to completion or until swapping is re-enabled.

Suspended Tasks

If command execution within a task is suspended by the WAITFOR SIGNAL, PAUSE, SUSPEND, or SUSPEND UNTIL command after swapping has been disabled, the next scheduled task will become active and begin to execute. If during a subsequent task SIGNAL occurs, the program is continued, or the suspended period expires, the suspended task will resume; however, it may or may not immeditately follow completion of the active task. It is recommended that swapping be enabled before program execution is suspended.

Examples on the use of ENABLE EOL SWAP and DISABLE EOL SWAP can be found in the "Multitasking Examples" section of this chapter and in the Command Reference Manual.

Related Errors

The only error associated with the execution of ENABLE EOL SWAP and DISABLE EOL SWAP is ERROR 86: MULTITASKING NOT ENABLED, which indicates that both commands can execute only in the multitasking mode.

Suspending a Task

In an HP 3852A multitasking system, you can suspend or defer the execution of commands or subroutines within a task for a specified period of time. This is often referred to as "putting a task to sleep." When a task is suspended (asleep), time-slicing continues among all other tasks of equal priority and higher priority commands are allowed to interrupt or preempt a task on command completion. When the period a task is suspended

www.valuetronics.com

expires, if it is a high priority task, command or subroutine execution may resume immediately following the currently executing command. If the suspended task has a priority equal to other tasks, the suspended task will be rescheduled in the time-slice sequence and resume execution.

Using the SUSPEND and SUSPEND UNTIL Commands

Applications for suspending a task are those instances where the period of time to suspend command execution is known, and it is acceptable for the system to continue with command execution in other tasks. Also, a suspended task only requires that the period of time expires before command execution resumes.

The commands used to suspend a task are the SUSPEND and SUSPEND UNTIL commands. The syntax of these commands and a description of their parameters are given below.

SUSPEND *seconds*

seconds - number of seconds the task is suspended. The range for *seconds* depends on the tic interval set by the TSLICE command.

SUSPEND UNTIL *seconds*

seconds - the Julian date and time (expressed in seconds) at which the task is no longer suspended. The setting of the mainframe clock is subtracted from the *seconds* specified, and the difference is the length of time the task is suspended. If the number of *seconds* specified corresponds to a time previous to the current clock setting, the task is not suspended.

The range for *seconds* also depends on the tic interval set by the TSLICE command. For the tic intervals available, the following ranges apply to both SUSPEND and SUSPEND UNTIL:

TIC_INTERVAL	MINIMUM PERIOD SUSPENDED	MAXIMUM PERIOD SUSPENDED
1.024 ms	1.024 ms	4398046.51 s (50.9 days)
2.048 ms	2.048 ms	8796093.02 s (101.8 days)
4.096 ms	4.096 ms	17592186.04 s (203.6 days)
8.192 ms	8.192 ms	35184372.08 s (407.2 days)
16.384 ms	16.384 ms	70368744.161 s (814.4 days)
32.768 ms	32.768 ms	140737488.322 s (1628.9 days)
65.536 ms	65.536 ms	281474976.645 s (3257.8 days)

The default tic interval is 65.536 ms. Thus, the range most commonly used will be 65.536 ms to 281,474,976.645 s.

When choosing between SUSPEND and SUSPEND UNTIL it is important to note that with SUSPEND, the number of seconds specified is the amount of time which elapses before the task resumes. With SUSPEND UNTIL, the setting of the mainframe clock is subtracted from the Julian date and time specified. The difference is the length of time suspended.

www.valuetronics.com

The following program segments demonstrate the use of SUSPEND UNTIL and SUSPEND, and show the differences between the two. In the first segment, SUSPEND UNTIL suspends a task until the day following the setting of the mainframe clock. The segment assumes the HP 3852A is already in the multitasking mode.

```
100  !Set the mainframe clock to the current date and time
110  !
120  OUTPUT 709;"SET TIMEDATE ";DATE("3 NOV 1987")+TIME("9:45:00")
130  !
140  !Download a run task subroutine. Suspend the task until the
150  !following day.
160  !
170  OUTPUT 709;"SUB A"
180  OUTPUT 709;" SUSPEND UNTIL ";DATE("4 NOV 1987")+TIME("7:30:00")
.
.
.
```

In this segment, the mainframe clock is set to the current date and time. The HP Series 200/300 controller's DATE and TIME functions are used to convert the date and time to Julian time (seconds). Next, the run task subroutine is suspended until the following day. The DATE and TIME functions are again used to make the conversion to seconds. When SUSPEND UNTIL executes, the setting of the clock (line 120) is subtracted from the date and time specified (line 180) and the period the task is suspended is determined.

The next program segment shows what is required in order for the SUSPEND command to suspend a task for the same period of time.

```
100  !Set the mainframe clock to the current date and time
110  !
120  OUTPUT 709;"SET TIMEDATE ";DATE("3 NOV 1987")+TIME("9:45:00")
130  !
140  !Download the run task subroutine. Suspend the subroutine until
150  !the following day.
160  !
170  OUTPUT 709;"SUB A"
180  OUTPUT 709;" REAL T"
190  OUTPUT 709;" TIMEDATE INTO T"
200  OUTPUT 709;" SUSPEND (";DATE("4 NOV 1987")+TIME("7:30:00");"-T)"
.
.
.
```


www.valuetronics.com

Notice in this segment that the setting of the clock (T) had to be subtracted from the date and time the task was to no longer be suspended. If this were not done, SUSPEND would attempt to suspend the task for the total number of seconds converted to by DATE and TIME. This would result in an error since the number of seconds would be outside the allowable range.

Examples illustrating the SUSPEND command can be found in the "Multitasking Examples" section of this chapter. The Command Reference Manual also contains detailed examples on the use of SUSPEND and SUSPEND UNTIL.

Related Errors

The errors associated with the execution of SUSPEND and SUSPEND UNTIL are:

ERROR 24: ARGUMENT OUT OF RANGE - the number of seconds specified is outside of the allowable range.

ERROR 86: MULTITASKING NOT ENABLED - SUSPEND and SUSPEND UNTIL can only execute in the multitasking mode.

Signaling Command Execution

In certain applications it may be desirable to suspend command execution within a task until "signaled" that a specific condition exists. The HP 3852A's WAITFOR SIGNAL and SIGNAL commands provide this capability.

Using WAITFOR SIGNAL and SIGNAL

Command execution within a task is suspended upon execution of the WAITFOR SIGNAL command. This command has the syntax shown below:

WAITFOR SIGNAL

Once the task has been suspended, the mainframe swaps to the next task and time-slicing/command execution continues with the other tasks. The suspended task remains suspended until it is signaled by the execution of the SIGNAL command. This command has the syntax shown below:

SIGNAL *task*

task - task to which the SIGNAL command is directed and which has been suspended by WAITFOR SIGNAL.

HPIB	Re-enables command/subroutine execution within the HP-IB task.
KYBD	Re-enables command/subroutine execution within the front panel task.
INTR	Resumes the execution of an interrupt-called subroutine.
run task number	Resumes execution of the subroutine in the specified run task. Only one run task number can be specified per SIGNAL command.

Should SIGNAL be directed to a task that is not yet suspended, the SIGNAL for that task is stored. When WAITFOR SIGNAL occurs, the

www.valuetronics.com

stored SIGNAL is detected immediately and the task is not suspended.

The following program segments show the interaction between WAITFOR SIGNAL and SIGNAL. In the run task subroutine STATS, WAITFOR SIGNAL (line 350) suspends execution until signaled that the data required for the STAT command (line 360) is available. Once signaled and when the STAT command completes, the run task signals the suspended HP-IB task (line 510) to enter the results of the STAT command.

```
330  OUTPUT 709;"SUB STATS"
340  OUTPUT 709;" REAL RSLTS(3)"
350  OUTPUT 709;" WAITFOR SIGNAL"
360  OUTPUT 709;" STAT RSLTS,RSLTS,RSLTS,RSLTS,STATRGS"
370  OUTPUT 709;" SIGNAL HPIB"
380  OUTPUT 709;"SUBEND"
  .
  .
  .
430  OUTPUT 709;"RUN 1 DCMEAS"
440  OUTPUT 709;"RUN 2 STATS"
  .
  .
  .
510  OUTPUT 709;"WAITFOR SIGNAL"
520  OUTPUT 709;"VREAD RSLTS"
530  ENTER 709;A,B,C,D
```

Signaling the Task

The point at which command execution resumes following execution of the SIGNAL command depends on the priority (urgency) of the task. If the suspended task has a higher priority, command execution may resume immediately after the SIGNAL command. If the suspended task has a priority equal to other tasks, the task will be rescheduled following completion of the SIGNAL command and continue time-slicing with the other tasks.

Related Errors

The only error associated with this method of suspending command execution is ERROR 84: RUN TASK DOES NOT EXIST. This occurs when the SIGNAL command is directed to a run task that has not been created by the RUN or CREATE RUN command.

Pausing/ Continuing a Run Task

Another method for controlling command execution is through pausing/continuing run task subroutines. When a run task subroutine is paused, all other run tasks continue to time-slice, and higher priority commands are allowed to interrupt or preempt a task on command completion.

Pausing a run task is useful in applications where conditions may arise that require the attention of the user. Once the subroutine is paused, any necessary modifications can be made and the subroutine can then be continued on command.

www.valuetronics.com

Using the PAUSE and CONT Commands

The PAUSE command pauses the run task subroutines. This command has the syntax given below:

PAUSE [*target*]

target - number of the run task containing the subroutine to be paused. The range for *target* is 0 to 7. If *target* is not specified, the run task subroutine in which PAUSE was executed is paused.

The PAUSE command is used to pause subroutines in both the power-on and multitasking modes. The *target* parameter is only used in the multitasking mode which requires mainframe firmware revision 3.0 or greater.

Executing the PAUSE Command

When the *target* parameter is specified, the PAUSE command can be executed from the front panel or over the HP-IB as well as within a subroutine.

Note that in the power-on mode or in the front panel, HP-IB, and interrupt tasks of the multitasking mode, PAUSE without *target* cannot be located inside a nested subroutine or in a subroutine called more than once. PAUSE without *target* can, however, be located in a nested run task subroutine. PAUSE *target* can be in a nested subroutine in the front panel, HP-IB, and interrupt environments as long as the subroutine does not target itself. Run task subroutines can target themselves.

Continuing a Paused Subroutine

A paused run task subroutine is continued with the CONT command. This command has the syntax shown below:

CONT [*target*]

target - number of the run task containing the subroutine to be continued. The range for *target* is 0 to 7. If *target* is not specified, the subroutine that was paused without the *target* parameter is continued.

The CONT command is used to continue subroutines in both the power-on and multitasking modes. The *target* parameter is only used in the multitasking mode which requires mainframe firmware revision 3.0 or greater.

Executing the CONT Command

When the *target* parameter is specified, CONT can be executed within a subroutine as well as from the front panel and HP-IB. When *target* is not specified, CONT cannot be executed within a subroutine.

www.valuetronics.com

In the following segment, the PAUSE command executed within run task 2 (subroutine B) is directed to run task 1 (subroutine A). While run task 1 is paused, run task 2 executes using successive time-slice periods. As the subroutine completes, the CONT command is directed to run task 1 enabling execution of that subroutine to resume.

```
260  OUTPUT 709;"SUB B"
270  OUTPUT 709;" INTEGER J"
280  OUTPUT 709;" PAUSE 1"
290  OUTPUT 709;" FOR J=0 TO 299"
300  OUTPUT 709;" DISP 'A'"
310  OUTPUT 709;" DISP 'B'"
320  OUTPUT 709;" DISP 'C'"
330  OUTPUT 709;" NEXT J"
340  OUTPUT 709;" CONT 1"
350  OUTPUT 709;"SUBEND"
  .
  .
  .
400  OUTPUT 709;"RUN 1 A"
410  OUTPUT 709;"RUN 2 B"
```

Subroutines Outside Run Tasks

PAUSE and CONT should only be used with subroutines which execute in a run task environment. If a subroutine in the front panel, HPIB, or interrupt task is paused, then continued from the front panel, the subroutine resumes execution in the front panel task, with all data returned to the display if it is not being stored internally. Also, the USE channel for that subroutine is the USE channel local to the front panel task.

When subroutines executing in run tasks are paused, and then continued from the front panel, the destination of the data returned and the USE channel assigned remain unchanged.

Related Errors

The programming errors associated with the PAUSE and CONT commands are given below:

ERROR 11: NO ACTIVE SUB - TASK READY - the run task specified in the PAUSE command is not executing a subroutine.

ERROR 24: ARGUMENT OUT OF RANGE - the run task number specified in either the PAUSE or CONT command is outside of the allowable range.

ERROR 76: INSIDE NESTED SUB - the PAUSE command without the target parameter or the PAUSE command in a subroutine targeting itself cannot be located in a nested subroutine executing in a task other than a run task.

ERROR 84: RUN TASK DOES NOT EXIST - the run task number specified in either the PAUSE or CONT command has not been activated by the RUN command.

www.valuetronics.com

ERROR 87: TASK NOT PAUSED - the run task targeted by the CONT command is not paused.

Requesting a Lock

The last method of controlling command execution is through the use of a lock. When a task requests a lock, all resources (i.e. voltmeter, array) accessed by the task are used only by that task. Other tasks which request the same lock must wait until the lock is released before they gain access to the resources.

Locks are the most complex method of controlling command execution and are intended for the advanced user. Thus, the commands associated with them (NLOCKS,REQUEST,RELEASE) and examples of how they are used are contained only in the Command Reference Manual.

Activating Run Tasks

As stated at the beginning of this chapter, the tasks you most often want to time-slice are the run tasks. By directing your application subroutines to these environments, portions of each subroutine execute as the system swaps from task to task. The time-slicing continues until the subroutines finish. The commands which direct subroutines to run tasks and which initiate subroutine execution are the RUN and ON INTR...RUN commands.

Using the RUN Command

The RUN command can execute within any task environment, however, it most often is executed in the HP-IB environment once all application subroutines have been downloaded into memory. The syntax and parameters of the RUN command are given below:

RUN *task_number name* [*number*][**EVERY** *seconds*]

task_number - run task to which a subroutine is directed. The range for *task_number* is 0 to 7. Any run task number can be specified regardless of the number of subroutines to be directed by subsequent RUN commands.

name - name of the subroutine directed to the run task.

number - number of times the subroutine is to execute. If *number* = 0, the subroutine executes continuously. If *number* is > 0, the subroutine executes the number of times specified. If *number* is not specified and the **EVERY** parameter is not specified, the default *number* is 1. If *number* is not specified and the **EVERY** parameter is specified, the subroutine executes continuously.

EVERY *seconds* - interval at which subsequent subroutine executions begin following the RUN command. For example, if `RUN 1 B 3 EVERY 5` is executed, subroutine B is directed to run task 1 and begins to execute. If the subroutine takes three seconds to execute, the subroutine will begin execution the second time in two seconds. Following the next three seconds of execution time, the operating system will wait two seconds then start execution the third (final) time.

www.valuetronics.com

The range for *seconds* is 0 to 4,294,967,296/tic interval. See the TSLICE command under "System Configuration" for tic interval settings. If **EVERY** *seconds* is not specified, there is no waiting between subsequent executions.

When Run Tasks become Activated

The subroutine directed to a run task which begins execution first is the subroutine specified by the first RUN command in the program. Provided additional run tasks have the same priority, the system swaps to the subroutine specified by the next RUN command encountered. For example, if three run tasks have the same priority and RUN commands are executed as:

```
OUTPUT 709;"RUN 4 C; RUN 2 B; RUN 7 T"
```

the run task executing subroutine C will become active first and then swap to the run task executing subroutine B. The run task executing B then swaps to the run task executing subroutine T which swaps back to C. The run tasks time-slice in this sequence until they finish.

Directing Subroutines to the Same Run Task

If more than one subroutine is directed to the same run task, the subroutine in the first RUN command will time-slice to completion or until it is aborted before the next subroutine directed to the run task will begin. For example, in the following program segment:

```
100  OUTPUT 709;"RUN 4 C"
110  OUTPUT 709;"RUN 4 K"
120  OUTPUT 709;"RUN 2 B"
130  OUTPUT 709;"RUN 7 T"
```

subroutines C, B, and T will time-slice until subroutine C completes, at which time subroutine K will time-slice with B and T. As subroutines C, B, and T are pulled from the queue, the run task subroutine that is executing is termed "active" while the others are "scheduled". Subroutine K is a "queued" subroutine until it is pulled from the queue when subroutine C completes.

Examples of how the RUN command is used can be found in each of the programs in the "Multitasking Examples" section of this chapter. Additional examples, including the use of the **EVERY** *seconds* parameter, can be found in the Command Reference Manual.

www.valuetronics.com

Using the ON ... RUN Command

A second method of directing subroutines is with the ON ... RUN command. This command directs the subroutine to the run task when a backplane interrupt, alarm interrupt, or limit interrupt occurs. The syntax of the command and a description of its parameters are given below.

ON *event* **RUN** *task_number name*

event - interrupt (backplane, alarm, limit) which directs the specified subroutine to the specified run task. The *event* parameters are:

Parameter	Description
INTR [USE ch]	Directs a subroutine to a run task when a backplane interrupt occurs on the USE channel specified.
ALRM	Directs a subroutine to a run task when an alarm occurs.
LMT	Directs a subroutine to a run task when a limit is exceeded during real-time limit testing.

task_number - run task to which the subroutine is directed when the interrupt occurs. The range for *task_number* is 0 to 7. Only one run task number can be specified per command.

name - name of the subroutine directed to the run task.

ON ... RUN Priorities

When an interrupt occurs and directs a subroutine to a run task, the interrupt is handled at the interrupt task priority (default = 65). However, the subroutine executes at the priority of the run task (default = 85). When used with CREATE RUN, ON ... RUN can direct a subroutine to a run task at any priority level.

An example on the use of the ON ... RUN command can be found in the Command Reference Manual.

Related Errors

Programming errors related to the execution of the RUN and ON ... RUN commands are:

ERROR 10: SUB DELETED - the subroutine name specified in the RUN or ON ... RUN command was deleted prior to the execution of the command.

ERROR 24: ARGUMENT OUT OF RANGE - the task number specified was outside of the allowable range.

ERROR 83: PROGRAM QUEUE FULL - the program queue cannot hold another subroutine name. Occurs on execution of the RUN or ON ... RUN command when the queue set by NTASKS is full.

ERROR 86: MULTITASKING NOT ENABLED - RUN and ON ... RUN can only execute in the multitasking mode.

www.valuetronics.com

System Status

An HP 3852A multitasking system allows you to monitor the status of run task subroutines and trace the swapping of tasks that occurs during program execution.

The run task information includes the number of run task environments available, the size of the program queue, the number of subroutine names in the queue, and whether a run task subroutine is executing, suspended, or has generated an error.

Tracing the execution of your multitasking program allows you to record the sequence in which tasks were swapped to. Also reported is the status of the task from which the swap occurred.

The commands which provide the system status information described above are the RUN? and PROBE/ENABLE PROBE commands. Also covered in this section is the ABORT command which allows you to stop subroutine execution in any task environment.

Using the RUN? Command

The operating status of the run task subroutines described previously is returned by the RUN? command. RUN?, which can execute in any task environment, has the following syntax:

RUN? [**INTO** *name*] or [*fmt*]

INTO *name* - when specified, the data returned is stored in a mainframe array (*name*). Note that the array must contain at least 11 elements.

fmt - the data returned can be sent to the output buffer in any of the formats allowed by the *fmt* parameter (Chapter 6).

www.valuetronics.com

Data Returned The status of the run task system is represented by 11 integers. The integers which are returned in the sequence shown, represent the following:

Integer	Description
1st	Number of run tasks available to the user as specified by the NTASKS command.
2nd	Size of the queue as set by the NTASKS command.
3rd	Number of subroutines in the queue awaiting execution in a run task.
4th	Status of the subroutine directed to run task 0.
5th	Status of the subroutine directed to run task 1.
6th	Status of the subroutine directed to run task 2.
7th	Status of the subroutine directed to run task 3.
8th	Status of the subroutine directed to run task 4.
9th	Status of the subroutine directed to run task 5.
10th	Status of the subroutine directed to run task 6.
11th	Status of the subroutine directed to run task 7.

The "status" that is returned for each of the subroutines (the 4th through 11th integers) is the sum of the following seven bit code:

Bit	Value	Status
0	1	The run task has been created.
1	2	The run task subroutine is presently executing (time-slicing) with other run tasks.
2	4	Execution of the run task subroutine is suspended by the SUSPEND command.
3	8	The subroutine generated an error.
4	16	The subroutine is currently paused or has been requested to pause by the PAUSE command.
5	32	Execution of the run task subroutine is suspended by the WAITFOR SIGNAL command.
6	64	Execution of the run task subroutine is suspended pending the release of a lock

Note that if a subroutine finishes and is replaced by a subroutine from the queue, the status reported is that of the subroutine pulled from the queue.

www.valuetronics.com

Using the System PROBE

The swapping of tasks throughout a multitasking program can be traced with the system probe. The commands associated with the probe are PROBE and ENABLE PROBE. PROBE designates the arrays which will store the data returned by the probe. ENABLE PROBE activates the probe. The syntax and a description of the commands are given below:

PROBE *prev_stat new_task swap_time*

ENABLE PROBE

prev_stat - Array in which the status of the task from which the swap occurred is stored. The status is represented by the following seven bit code:

Bit	Value	Status
0	1	The task has been created.
1	2	The task has been activated and is running.
2	4	The task has been suspended by the SUSPEND or SUSPEND UNTIL command.
3	8	An error has occurred within the task.
4	16	The task has been paused by the PAUSE command.
5	32	The task has been suspended by the WAITFOR SIGNAL command.
6	64	The task has been suspended pending the release of a lock

The *prev_stat* array must be an Integer array and should be the same size as the *new_task* and *swap_time* arrays.

new_task -array which stores the number of the task as it is swapped to. For the probe, the tasks are numbered as follows:

Number	Task
0 through 7	Run tasks 0 through 7.
-2	Keyboard task.
-3	HPIB task.
-4	Interrupt task.
-5	Instrument ready.
-1	Device clear.

The *new_task* array must also be an Integer array and should be the same size as the *prev_stat* and *swap_time* arrays.

swap_time - array which stores the reading of the mainframe's clock at the time a swap occurs. The *swap_time* array must be a Real array and should be the same size as the *prev_stat* and *new_task* arrays.

Disabling the Probe

The system probe is disabled when the instrument is reset, when the power is cycled, or upon execution of the DISABLE PROBE command.

Detailed examples on the use of the RUN? and PROBE/ENABLE PROBE commands can be found in the Command Reference Manual.

www.valuetronics.com

Using the ABORT Command

The ABORT command aborts the active or scheduled subroutine executing in the specified task environment. Queued subroutines, however, are not aborted. The syntax and parameters of the ABORT command are given below:

ABORT [*task*]

task - task from which the subroutine to be aborted was called. If a task is not specified, the subroutine containing the ABORT command is aborted.

Task	Description
HPIB	Currently executing subroutine called from the HP-IB task is aborted.
KYBD	Currently executing subroutine called from the front panel task is aborted.
INTR	Currently executing subroutine called in response to an interrupt is aborted.
run task number	Number of the active/scheduled run task subroutine that is aborted.

Suspended Subroutines

If a subroutine has been suspended by the SUSPEND, SUSPEND UNTIL, WAITFOR SIGNAL, or PAUSE command, the ABORT command if directed to that subroutine will abort the subroutine.

When a subroutine is aborted, execution stops, however, the subroutine is still located in mainframe memory. Thus, if necessary, the subroutine can be directed to a run task or called again.

A detailed example on the use of the ABORT command can be found in the Command Reference Manual.

Related Errors

The errors associated with the ABORT command are:

ERROR 84: RUN TASK DOES NOT EXIST - the run task number specified by the ABORT command has not been activated by the RUN command.

ERROR 86: MULTITASKING NOT ENABLED - the ABORT command can only execute in the multitasking mode.

www.valuetronics.com

Multitasking Examples

This section contains six examples which demonstrate HP 3852A multitasking. In the examples, the subroutines developed for each application either contain, or are integrated into a multitasking system using the multitasking commands covered previously.

The multitasking examples contained in this section are:

1. Time-Slicing
2. Queued tasks
3. Priorities
4. Interactive programming
5. Data logging
6. I/O buffers

A description of the program and the HP 3852A's configuration is included with each program listing.

Example 1: Time-Slicing

This program contains three subroutines (GRAD, PRES, and FLOW) which are directed to run tasks. Each of these run task subroutines contain infinite loops such that they will run continuously. Since the run tasks all have the same priority (the priority set upon entering the multitasking mode = 85), the tasks will time-slice indefinitely unless an HP-IB or front panel clear is performed.

HP 3852A Configuration

The slots and channel ranges specified in the following program correspond to the following configuration:

Slot 0 HP 44708A 20-Channel Relay Multiplexer (T/C)

Slot 1 HP 44701A Integrating Voltmeter

Slot 2 HP 44715A 5-Channel Counter

www.valuetronics.com

```
10    ! Sample program 1
20    ! TIME-SLICING
30    !
40    !   3 separate measurements
50    !
60    ! 1) Measure 2 temperatures test for gradient
70    ! 2) Measure 3 pressures, record if changed
80    ! 3) Measure 4 flows, test for blockage in flows
90    !
100   CLEAR 7
110   OUTPUT 709;"TSLICE .065;NTASKS 8,20;ENABLE MULTI"  ! Sets up multitasking
120   !
130   OUTPUT 709;"SUB GRD"                                !Sub informs user
140   OUTPUT 709;"DISP ""GRADIENT"""""
150   OUTPUT 709;"SUBEND"
160   !
170   OUTPUT 709;"SUB GRAD"
180   OUTPUT 709;"USE 100"                                !Voltmeter in slot 1
190   OUTPUT 709;"WHILE 1"                                !Infinite loop
200   OUTPUT 709;"REAL G(2)"                              !Array G is temperature
210   OUTPUT 709;"CONF MEAS TEMPT 5,6 INTO G"             !Measure temperature
220   OUTPUT 709;"IF ABS(G(0)-G(1))>.5 THEN"              !Check gradient
230   OUTPUT 709;"CALL GRD"                               !If yes then call sub to
240   OUTPUT 709;"ELSE"                                   ! Warn operator
250   OUTPUT 709;"DISP "" """""                           !Otherwise clear display
260   OUTPUT 709;"END IF"
270   OUTPUT 709;"END WHILE"                              !Measure temperatures again
280   OUTPUT 709;"SUBEND"
290   !
300   OUTPUT 709;"SUB PRES"
310   OUTPUT 709;"USE 100"                                !Voltmeter is in slot 100
320   OUTPUT 709;"REAL P_L(3)"                            !P_L last pressure recorder
330   OUTPUT 709;"REAL P_T(100)"                          !P_T record of times
340   OUTPUT 709;"REAL P_C(300),I,J"                      !P_C record of pressures
350   OUTPUT 709;"WHILE 1"                                !Loop forever
360   OUTPUT 709;"REAL P(3);P(3)=0"                       !P is current pressure
370   OUTPUT 709;"CONF MEAS DCV 8-10 INTO P"              !Measure pressures
380   OUTPUT 709;"FOR I=0 TO 2"                           !See if there is change
390   OUTPUT 709;"P(3)=ABS(P_L(I)-P(I))+P(3)"
400   OUTPUT 709;"NEXT I"
410   OUTPUT 709;"IF P(3)>.1 THEN"                        !If change then
420   OUTPUT 709;"DISABLE EOL SWAP"                       !Disable swapping
430   OUTPUT 709;"INDEX P_L 0"                            !Point to start of array
440   OUTPUT 709;"TIME INTO P_T"                          !Record time
450   OUTPUT 709;"CONF MEAS DCV 8-10 INTO P_C"            !Record pressures
```


www.valuetronics.com

```
460   OUTPUT 709;"CONF MEAS DCV 8-10 INTO P_L"      !Update last pressures
470   OUTPUT 709;"DISP ""PRESSURE """""             !Warn operator
480   OUTPUT 709;"ENABLE EOL SWAP"                  !Turn swapping on
490   OUTPUT 709;"END IF"
500   OUTPUT 709;"END WHILE"                        !Measures again
510   OUTPUT 709;"SUBEND"
520   !
530   OUTPUT 709;"SUB FLOW"
540   OUTPUT 709;"REAL F,K"
550   OUTPUT 709;"FOR K=200 TO 203"                 !For loop to set up the
560   OUTPUT 709;"USE K"                            ! USE channel
570   OUTPUT 709;"CONF FREQ"                        !Set up counter
580   OUTPUT 709;"TERM NON"
590   OUTPUT 709;"TRIG AUTO"
600   OUTPUT 709;"NEXT K"
610   OUTPUT 709;"WHILE 1"                          !
620   OUTPUT 709;"FOR K=200 TO 203"                 !Use a loop to read
630   OUTPUT 709;"CHREAD K INTO F"                  ! the counter channels
640   OUTPUT 709;"IF F< 90 THEN"                    !Check to see if flow is
650   OUTPUT 709;"DISP ""FLOW BLOCKED"""""          ! blocked; if so, warn
660   OUTPUT 709;"BEEP"                             ! the operator
670   OUTPUT 709;"END IF"
680   OUTPUT 709;"NEXT K"
690   OUTPUT 709;"END WHILE"                        !Measure flows again
700   OUTPUT 709;"SUBEND"
710   !
720   OUTPUT 709;"RUN 0 GRAD"                       !Start up the tasks
730   OUTPUT 709;"RUN 1 PRES"
740   OUTPUT 709;"RUN 2 FLOW"
750   END
```

Example 2: Queued Tasks

When several subroutines are directed to the same run task, the first subroutine directed to the task executes while the others remain in the program queue. In the following program, subroutine "BOILER" is directed to run task 1. The subroutines "TEMP", "PRES", and "FLOW" are directed to run task 0. Since only one subroutine at a time can execute in a specific run task environment, subroutine TEMP executes first while subroutines PRES and FLOW remain in the queue. On program execution, run task 0 (TEMP) and run task 1 (BOILER) time-slice until TEMP completes. At that point, PRES time-slices with BOILER. When PRES completes, FLOW time-slices with BOILER. When FLOW completes, BOILER executes by itself since it contains an infinite loop.

LIMIT is a subroutine that is called when an out-of-limit condition exists in the subroutine BOILER.

www.valuetronics.com

The slots and channel ranges specified in the following program correspond to the following configuration:

Slot 0 HP 44708A 20-Channel Relay Multiplexer (T/C)

Slot 1 HP 44701A Integrating Voltmeter

Slot 2 HP 44715A 5-Channel Counter

```
10   ! Sample program 2
20   ! QUEUED TASKS
30   !
40   !   3 Programs to be done in sequence, 1 continuous
50   !
60   ! 1) Measure 5 temperatures
70   ! 2) Measure 3 pressures
80   ! 3) Measure 4 flows
90   ! 4) Monitor boiler temperatures
100  !
110  ! Recover data in sequence
120  !
130  CLEAR 7
140  DIM T(4),P(2),F(3)
150  OUTPUT 709;"TSLICE .065;NTASKS 8,20;ENABLE MULTI"
160  OUTPUT 709;"OUTBUF ON"
170  !
180  OUTPUT 709;"SUB LIMIT"                          !Sub to be called when
190  !      Code to correct limit exceeded           ! the boiler limit
200  OUTPUT 709;"SUBEND"                             ! is exceeded
210  !
220  OUTPUT 709;"SUB BOILER"                         !Monitor boiler
230  OUTPUT 709;"WHILE 1"                            !Loop forever
240  OUTPUT 709;"REAL B_T(3)"                        !Set up variable
250  OUTPUT 709;"USE 100"                            !Voltmeter in slot 1
260  OUTPUT 709;"CONF MEAS TEMPT 5-8 INTO B_T"       !Measure boiler temp
270  OUTPUT 709;"IF (B_T(1)-B_T(2))> 100 THEN"       !If room temp minus boiler
280  OUTPUT 709;"CALL LIMIT;END IF"                  !Exceeds the limits then
290  OUTPUT 709;"END WHILE"                          !Call a subroutine
300  OUTPUT 709;"SUBEND"
310  !
320  OUTPUT 709;"RUN 1 BOILER"                       !Start boiler monitor
330  !
340  !      Some time later .....
350  !
```


www.valuetronics.com

```
360  OUTPUT 709;"SUB TEMP;USE 100"                    !Voltmeter is in slot 1
370  OUTPUT 709;"REAL T(4)"                           !Array T is temperature
380  OUTPUT 709;"CONF MEAS TEMPT 0-4 INTO T"          !Measure temperature
390  OUTPUT 709;"VREAD T"                             !Move data to HPIB buffer
400  OUTPUT 709;"SUBEND"
410  !
420  OUTPUT 709;"SUB PRES"
430  OUTPUT 709;"USE 100"                             !Voltmeter is in slot 100
440  OUTPUT 709;"REAL P(2)"                           !Array P is pressure
450  OUTPUT 709;"CONF MEAS DCV 8-10 INTO P"           !Measure pressures
460  OUTPUT 709;"VREAD P"                             !Move data to HPIB buffer
470  OUTPUT 709;"SUBEND"
480  !
490  OUTPUT 709;"SUB FLOW"
500  OUTPUT 709;"REAL F(3),K"
510  OUTPUT 709;"FOR K=200 TO 203"                    !For loop to increment
520  OUTPUT 709;"USE K"                               ! Use channels
530  OUTPUT 709;"CONF FREQ"                           !Set up counter for freq
540  OUTPUT 709;"TERM NON"                            !TTL input is used
550  OUTPUT 709;"TRIG AUTO"                           !
560  OUTPUT 709;"CHREAD K INTO F"                     ! Counter channels
570  OUTPUT 709;"NEXT K"
580  OUTPUT 709;"VREAD F"                             !Move data to HPIB buffer
590  OUTPUT 709;"SUBEND"
600  !
610  OUTPUT 709;"RUN 0 TEMP;RUN 0 PRES;RUN 0 FLOW"    !Put subs into the queue
620  !
630  ENTER 709;T(*)                                   !Recover data
640  PRINT T(*)
650  ENTER 709;P(*)
660  PRINT P(*)
670  ENTER 709;F(*)
680  PRINT F(*)
690  END
```

Example 3: Priorities

In this example, the task priorities are set as shown:

Priority	Task	Commands/Subroutines
5	Front Panel	User-defined commands
10	Run task 0	T_OVEN subroutine
15	HP-IB	User-defined commands
20	Interrupt subroutine	FLOW subroutine
25	Run task 1	PRES subroutine

www.valuetronics.com

Commands executed from the front panel of the instrument or subroutines that are called from the front panel have a priority of 5. If no front panel commands are entered, the HP 3852A will execute run task 0 (T_OVEN). T_OVEN uses the DISABLE EOL SWAP command to prevent a swap from occurring in the middle of a measurement sequence. It also uses the SUSPEND command to put the subroutine "to sleep" for one second. During this time, other tasks of any priority can execute. Due to its relatively high priority, when T_OVEN "wakes up", it will preempt the lower priority tasks.

The HP 3852A system alarm calls the subroutine FLOW every three seconds. Although scheduled every three seconds, due to the task's low priority the subroutine must wait until all higher priority tasks have finished or have been suspended before starting execution.

Commands entered within the HP-IB task are executed at some point in time after run task 0 has started. To ensure that run task 0 does not preempt the HP-IB task during the transfer of data, DISABLE EOL SWAP is used again.

HP 3852A Configuration

The slot and channel ranges specified in the following program correspond to the following HP 3852A configuration:

Slot 0 HP 44708A 20-Channel Relay Multiplexer (T/C)

Slot 1 HP 44701A Integrating Voltmeter

Slot 2 HP 44715A 5-Channel Counter

```
10   ! Sample program 3
20   ! PRIORITIES
30   !
40   !Priority                                   Task
50   !  5          Keyboard (lets user abort run tasks)
60   !  10         Run 0 (temperature monitor sub "T_OVEN")
70   !  15         HPIB (recover data)
80   !  20         Interrupts (flow measurements sub "FLOW")
90   !  25         Run 1 (pressure monitor sub "PRES")
100  !
110  DIM Temp(20),Time(20),Pres(2),Flow(2)
120  CLEAR 7
```


www.valuetronics.com

```
130  OUTPUT 709;"TSLICE .065;NTASKS 8,20;ENABLE MULTI"  !Set up multitasking
140  OUTPUT 709;"INBUF ON;OUTBUF ON"                    !Turn on HP-IB buffers
150  !
160  OUTPUT 709;"SUB T_OVEN"
170  OUTPUT 709;"USE 100"                               !Voltmeter is in slot 1
180  OUTPUT 709;"REAL T_TEMP(20),T_TIME(20),I"          !Arrays for history
190  OUTPUT 709;"WHILE 1"                               !Loop forever
200  OUTPUT 709;"DISABLE EOL SWAP"                      !Disallow swapping for
210  OUTPUT 709;"TIME INTO T_TIME"                      !    time & temperature
220  OUTPUT 709;"CONF MEAS TEMPT 23 INTO T_TEMP"        !    measurement correlation
230  OUTPUT 709;"INDEX? T_TEMP INTO I"                  !Check for wrap around
240  OUTPUT 709;"IF I=20 THEN;INDEX T_TIME 0"           !    in arrays. If so then
250  OUTPUT 709;"INDEX T_TEMP 0;END IF"                 !    set index in array to 0
260  OUTPUT 709;"ENABLE EOL SWAP"                       !Enable swapping
270  OUTPUT 709;"SUSPEND 1"                             !Sleep for 1 second
280  OUTPUT 709;"END WHILE"
290  OUTPUT 709;"SUBEND"
300  !
310  OUTPUT 709;"SUB PRES"                              !
320  OUTPUT 709;"USE 100"                               !Voltmeter is in slot 1
330  OUTPUT 709;"REAL P(2)"                             !P is pressure
340  OUTPUT 709;"CONF MEAS DCV 8-10 INTO P"             !Measure current pressures
350  OUTPUT 709;"IF P(0)<P(1) THEN"                     !Check for pressure change
360  OUTPUT 709;"DISP ""WARNING P"";END IF"             !Warn operator if changed
370  OUTPUT 709;"SUBEND"
380  !
390  OUTPUT 709;"SUB FLOW"
400  OUTPUT 709;"REAL F(2),K"                           !
410  OUTPUT 709;"FOR K=200 to 202"                      !Increment through channels
420  OUTPUT 709;"USE K"                                 !Use channel
430  OUTPUT 709;"CONF FREQ"                             !Set up counter for freq
440  OUTPUT 709;"TERM NON"                              !Terminals are TTL
450  OUTPUT 709;"TRIG AUTO"                             !
460  OUTPUT 709;"CHREAD K INTO F"                       !Record frequency
470  OUTPUT 709;"NEXT K"
480  OUTPUT 709;"SUBEND"
490  !
```


www.valuetronics.com

```
500  OUTPUT 709;"ON ALRM CALL FLOW"                !Measure flow on alarm
510  OUTPUT 709;"ENABLE INTR SYS;ENABLE ALRM "     !Enable the interrupts
520  OUTPUT 709;"SET TIME 0;SET ALRM 3"            !Set the alarm for 3 seconds
530  !
540  OUTPUT 709;"RUN 0 T_OVEN;RUN 1 PRES"          !Start monitoring
550  OUTPUT 709;"URGENCY KEYBD 5;URGENCY HPIB 15"  !Set priorities
560  OUTPUT 709;"URGENCY 1 25;URGENCY INTR 20"     !
570  OUTPUT 709;"URGENCY 0 10"
580  !
590  WAIT 8     !SOME TIME LATER
600  !
610  OUTPUT 709;"DISABLE EOL SWAP"                 !Prevent race with "T_OVEN"
620  OUTPUT 709;"VREAD T_TIME;VREAD T_TEMP;VREAD I" !Move data into HP-IB buffers
630  OUTPUT 709;"ENABLE EOL SWAP"                  !Enable swapping
640  ENTER 709;Time(*)                             !Recover time data from
650  PRINT Time(*)                                 ! HP-IB buffer
660  ENTER 709;Temp(*)                             !Recover temperature data
670  PRINT Temp(*)                                 ! From HP-IB buffer
680  ENTER 709;I                                   !Recover index into arrays
690  PRINT I
700  OUTPUT 709;"VREAD P"                          !Recover pressure
710  ENTER 709;Pres(*)
720  PRINT Pres(*)
730  OUTPUT 709;"VREAD F"                          !Recover flow
740  ENTER 709;Flow(*)
750  PRINT Flow(*)
760  END
```


www.valuetronics.com

Example 4: Interactive Programming

When the HP 3852A enters the multitasking mode, the HP-IB task has a higher priority than the run tasks. This allows the user to preempt run tasks with commands sent over the HP-IB. To ensure that these commands do not disturb critical measurements, DISABLE EOL SWAP is used in selected run task subroutines which follow.

In the following program, subroutine BOILER (run task 1) time-slices with HISTORY (run task 2). The system alarm calls the subroutine OVEN every second to make measurements and perform PID control. Since OVEN executes within a higher priority task, it will run to completion before the time-slicing tasks resume. At anytime (except during critical measurement sequences protected by DISABLE EOL SWAP), the user can preempt any of these tasks with commands over HP-IB.

The subroutine STOP is CALLed by BOILER if the boiler temperature is out of limits.

Type	Priority	Subroutine	Purpose
Run task 1	85	BOILER	Continuously monitors boiler temperature.
Run task 2	85	HISTORY	Keeps a history of temperature readings. Uses the SUSPEND command to put the subroutine to sleep for three seconds.
Subroutine	85	STOP	Shuts down the process. Called from run task 1.
System alarm	65	OVEN	Performs PID control on an oven every second.
HP-IB	45	Commands and data	Sends new PID constants to the instrument and retrieves data from HP 3852A memory.

HP 3852A Configuration

The slot numbers and channel ranges specified throughout the program correspond to the following HP 3852A configuration:

Slot 0 HP 44708A 20-Channel Relay Multiplexer (T/C)

Slot 1 HP 44701A Integrating Voltmeter

Slot 4 HP 44724A 16-Channel Digital Output

Slot 7 HP 44727A Digital-to-Analog Converter

```
10    ! Sample Program 4
20    ! INTERACTIVE PROGRAMMING
30    !
40    ! Alarm interrupt is used to control an oven
```


www.valuetronics.com

```
50   ! Run task 1 monitors a boiler continuously
60   ! Run task 2 takes historical data once every 3 seconds
70   !
80   DIM H_t(100)
90   OUTPUT 709;"TSLICE .065;NTASKS 8,20;ENABLE MULTI"   !Sets up multitasking
100  OUTPUT 709;"INBUF ON;OUTBUF ON"                     !Turn on HP-IB input buffer
110  !
120  OUTPUT 709;"REAL O_T,O_S,O_E,O_EO,O_I,V"  !Variables for PID
130  OUTPUT 709;"REAL K1,K2,K3"                !Variable for constants
140  OUTPUT 709;"O_S=55"                       !Initial set point for oven
150  !
160  OUTPUT 709;"SUB OVEN"                     !Measures a channel every second
170  OUTPUT 709;"USE 100"                      !Voltmeter is in slot 1
180  OUTPUT 709;"DISABLE EOL SWAP"             !Disable swapping
190  OUTPUT 709;"SET TIME 0;ENABLE ALRM"       !Set up alarm clock
200  OUTPUT 709;"CONF MEAS TEMPT 23 INTO O_T"  !Measure temperature
210  OUTPUT 709;"O_E=O_T-O_S"                  !Compute error
220  OUTPUT 709;"O_I=O_I+O_E"                  !Compute I term
230  OUTPUT 709;"V=K1*O_E+K2*(O_E-O_EO)+K3*(O_I)+V"
240  OUTPUT 709;"O_EO=O_E"                     !Store error term
250  OUTPUT 709;"IF V>5 THEN;V=5;END IF"       !Clip output to 5 volts
260  OUTPUT 709;"IF V<-5 THEN;V=-5;END IF"     !Clip output to -5 volts
270  OUTPUT 709;"APPLY DCV 700 V"              !Output control voltage
280  OUTPUT 709;"ENABLE EOL SWAP"              !Enable swapping
290  OUTPUT 709;"SUBEND"
300  !
310  OUTPUT 709;"SUB STOP"                     !
320  OUTPUT 709;"OPEN 400,403;CLOSE 402,404"   !Shut down experiment
330  OUTPUT 709;"DISP ""WARNING 1"""""         !Warn operator
340  OUTPUT 709;"SUBEND"
350  !
360  OUTPUT 709;"SUB BOILER"                   !Monitor boiler
370  OUTPUT 709;"USE 100"                      !Voltmeter is in slot 1
380  OUTPUT 709;"REAL B_T"
390  OUTPUT 709;"WHILE 1"                      !Loop forever
400  OUTPUT 709;"CONF MEAS TEMPT 2 INTO B_T"   !Check temperature
410  OUTPUT 709;"IF B_T>130 THEN"              !If out of range then
420  OUTPUT 709;"URGENCY 1"                    !Set to highest priority
430  OUTPUT 709;"CALL STOP"                    ! and stop
440  OUTPUT 709;"END IF"
450  OUTPUT 709;"END WHILE"
460  OUTPUT 709;"SUBEND"
470  !
480  OUTPUT 709;"SUB HISTORY"
490  OUTPUT 709;"USE 100"                      !Voltmeter is in slot 1
500  OUTPUT 709;"REAL H_T(100),H_I"            !Arrays for history
```


www.valuetronics.com

```
510  OUTPUT 709;"WHILE 1"                                  !Loop forever
520  OUTPUT 709;"CONF MEAS TEMPT 5 INTO H_T"               !Collect data
530  OUTPUT 709;"INDEX? H_T INTO H_I"                      !Check for wraparound
540  OUTPUT 709;"IF H_I=100 THEN;INDEX H_T 0"              !
550  OUTPUT 709;"END IF"                                   !
560  OUTPUT 709;"SUSPEND 3"                                !Sleep 3 for seconds
570  OUTPUT 709;"END WHILE"
580  OUTPUT 709;"SUBEND"
590  !
600  OUTPUT 709;"RUN 1 BOILER;RUN 2 HISTORY"               !Start up tasks
610  OUTPUT 709;"ENABLE INTR SYS;ENABLE ALRM"              !Set alarm
620  OUTPUT 709;"ON ALRM CALL OVEN"
630  OUTPUT 709;"SET TIME 0;SET ALRM 1"
640  !
650  Constants !
660  INPUT "CHANGE CONTROL LOOP OR RECOVER HISTORY (C/H)",X$
670  IF X$="C"THEN
680    INPUT "CONTROL CONSTANTS",K1,K2,K3                  !Type in new constants
690    OUTPUT 709;"DISABLE EOL SWAP"                       !Disable swapping
700    OUTPUT 709;"K1=";K1;";K2=";K2;";K2=";K3             !Send new constants
710    OUTPUT 709;"ENABLE EOL SWAP"                        !Enable swapping
720  END IF
730  IF X$="H"THEN
740    OUTPUT 709;"VREAD H_T;VREAD H_I"                    !Read variables in the
750    ENTER 709;H_t(*)                                    ! instrument
760    ENTER 709;H_i
770    PRINT H_t(*),H_i
780  END IF
790  GOTO Constants
800  END
```


www.valuetronics.com

Example 5: Data Logger

The multitasking operating system and the SUSPEND command simplify data logging programming. The program of Example 5 contains three run task subroutines, each of which is suspended for a different period of time, thus executing at different rates. If more than one task is not suspended, the tasks time-slice. DISABLE EOL SWAP is used to ensure that critical portions of the run task subroutines are not disrupted by time-slicing or higher priority tasks.

Using variables, the suspend time for each task can be specified. Data acquired by the three tasks is placed into a circular buffer (array) and the time the data is taken is placed into another buffer. Using variables, the size of this buffer can be specified. At anytime, the controller can read the entire buffer and determine (using an index variable) which reading has been in the buffer the longest. Reading the time buffer allows the user to correlate the time of the measurement with the measurement itself.

HP 3852A Configuration

The slot numbers and channel ranges specified throughout the program correspond to the following HP 3852A configuration:

Slot 0 HP 44708A 20-Channel Relay Multiplexer (T/C)

Slot 1 HP 44701A Integrating Voltmeter

Slot 2 HP 44715A 5-Channel Counter

```
10    ! Sample Program 5
20    ! DATA LOGGER
30    !
40    ! Sub A records 5 temperatures every 5 seconds (A_rate=seconds of period)
50    ! Sub B records 2 flows every second (B_rate=seconds of period)
60    ! Sub C records 4 voltages every 2 seconds (C_rate=seconds of period)
70    ! Data buffer size is 1 minute (min_dat=minutes of data in buffer)
80    !
90    DIM A_time(1000),A_temp(5000)          !Set up large arrays for
100   DIM B_time(1000),B_flow(2000)          ! data.
110   DIM C_time(100),C_volt(400)
120   CLEAR 709
```


www.valuetronics.com

```
130  OUTPUT 709;"TSLICE .065;NTASKS 8;ENABLE MULTI"  !Set up multitasking
140  OUTPUT 709;"REAL A_RATE,B_RATE,C_RATE,MIN_DAT"  !Declare constants
150  !
160  OUTPUT 709;"SUB A"
170  OUTPUT 709;"INTEGER A_SIZE"                     !A_size is the size of
180  OUTPUT 709;"A_SIZE=MIN_DAT*60/A_RATE"  ! data arrays
190  OUTPUT 709;"REAL I,A_TIME(A_SIZE),A_TEMP(5*A_SIZE)"
200  OUTPUT 709;"WHILE 1"                            !Loop forever
210  OUTPUT 709;"DISABLE EOL SWAP"                   !Disable swapping
220  OUTPUT 709;"TIME INTO A_TIME "                  !Timestamp
230  OUTPUT 709;"CONF MEAS TEMPT 0-4 INTO A_TEMP"  !Measure temperature
240  OUTPUT 709;"INDEX? A_TIME INTO I"               !Check index pointer
250  OUTPUT 709;"IF I=A_SIZE THEN;INDEX A_TIME 0"
260  OUTPUT 709;"INDEX A_TEMP 0;END IF"
270  OUTPUT 709;"ENABLE EOL SWAP"                    !Enable swapping
280  OUTPUT 709;"SUSPEND A_RATE"                     !Subroutine sleeps for
290  OUTPUT 709;"END WHILE"                          ! A_RATE seconds
300  OUTPUT 709;"SUBEND"
310  !
320  OUTPUT 709;"SUB B"
330  OUTPUT 709;"INTEGER B_SIZE;B_SIZE=MIN_DAT*60/B_RATE"
340  OUTPUT 709;"REAL I,J,B_TIME(B_SIZE),B_FLOW(2*B_SIZE)"
350  OUTPUT 709;"FOR J=201 TO 202"
360  OUTPUT 709;"CONF FREQ USE J"                    !Set up counters
370  OUTPUT 709;"TRIG SYS USE J"                     ! to measure frequency
380  OUTPUT 709;"TBASE .1 USE J"
390  OUTPUT 709;"TERM NON USE J"
400  OUTPUT 709;"NEXT J"
410  OUTPUT 709;"WHILE 1"                            !Loop forever
420  OUTPUT 709;"DISABLE EOL SWAP"                   !Disable swapping
430  OUTPUT 709;"TRG"
440  OUTPUT 709;"TIME INTO B_TIME "                  !Timestamp
450  OUTPUT 709;"CHREAD 201 INTO B_FLOW"             !Read the counters
460  OUTPUT 709;"CHREAD 202 INTO B_FLOW"
470  OUTPUT 709;"INDEX? B_TIME INTO I"               !Check array index
480  OUTPUT 709;"IF I=B_SIZE THEN;INDEX B_TIME 0"
490  OUTPUT 709;"INDEX B_FLOW 0;END IF"
500  OUTPUT 709;"ENABLE EOL SWAP"                    !Enable swapping
510  OUTPUT 709;"SUSPEND B_RATE"                     !Subroutine sleeps for
520  OUTPUT 709;"END WHILE"                          ! B_RATE seconds
530  OUTPUT 709;"SUBEND"
540  !
550  OUTPUT 709;"SUB C"
560  OUTPUT 709;"INTEGER C_SIZE;C_SIZE=MIN_DAT*60/C_RATE"
570  OUTPUT 709;"REAL I,C_TIME(C_SIZE),C_VOLT(4*C_SIZE)"
580  OUTPUT 709;"WHILE 1"                            !Loop forever
```


www.valuetronics.com

```
590  OUTPUT 709;"DISABLE EOL SWAP"                    !Disable swapping
600  OUTPUT 709;"TIME INTO C_TIME "                   !Timestamp
610  OUTPUT 709;"CONF MEAS DCV 6,8,9,11 USE 100 INTO C_VOLT"  ! Voltages
620  OUTPUT 709;"INDEX? C_TIME INTO I"                !Check array index
630  OUTPUT 709;"IF I=C_SIZE THEN;INDEX C_TIME 0"
640  OUTPUT 709;"INDEX C_VOLT 0;END IF"
650  OUTPUT 709;"ENABLE EOL SWAP"                     !Enable swapping
660  OUTPUT 709;"SUSPEND C_RATE"                      !Subroutine sleeps for
670  OUTPUT 709;"END WHILE"                           ! C_RATE seconds
680  OUTPUT 709;"SUBEND"
690  !
700  OUTPUT 709;"A_RATE=5;B_RATE=1;C_RATE=2;MIN_DAT=.25"  !Load set
     points
710  !
720  OUTPUT 709;"RUN 0 A;RUN 1 B;RUN 2 C;DISP OFF"  !Run subroutines
730  !
740  OUTPUT 709;"INBUF ON;OUTBUF ON"                  !Turn I/O buffers ON
750  OUTPUT 709;"DISP OFF"                            !Turn off the display
760  PAUSE                                            !Let HP 3852A accumulate
770                                                   ! data
780  ! Press continue when ready
790  !
800  OUTPUT 709;"VREAD A_SIZE;VREAD B_SIZE;VREAD C_SIZE"  !Read variables
810  ENTER 709;A_size
820  ENTER 709;B_size
830  ENTER 709;C_size
840  DISP A_size,B_size,C_size
850  OUTPUT 709;"INDEX? A_TIME;INDEX? A_TEMP INTO I;VREAD A_TEMP;INDEX
     A_TEMP I"
860  OUTPUT 709;"INDEX? A_TIME INTO I;VREAD A_TIME;INDEX A_TIME I"
870  OUTPUT 709;"INDEX? B_TIME;INDEX? B_FLOW INTO I;VREAD B_FLOW;INDEX
     B_FLOW I"
880  OUTPUT 709;"INDEX? B_TIME INTO I;VREAD B_TIME;INDEX B_TIME I"
890  OUTPUT 709;"INDEX? C_TIME;INDEX? C_VOLT INTO I;VREAD C_VOLT;INDEX
     C_VOLT I"
900  OUTPUT 709;"INDEX? C_TIME INTO I;VREAD C_TIME;INDEX C_TIME I"
910  ENTER 709;Index_a
920  FOR I=0 TO A_size*5
930    ENTER 709;A_temp(I)
940  NEXT I
950  FOR I=0 TO A_size
960    ENTER 709;A_time(I)
970  NEXT I
980  ENTER 709;Index_b
990  FOR I=0 TO B_size*2
1000   ENTER 709;B_flow(I)
```


www.valuetronics.com

```
1010  NEXT I
1020  FOR I=0 TO B_size
1030    ENTER 709;B_time(I)
1040  NEXT I
1050  ENTER 709;Index_c
1060  FOR I=0 TO C_size*4
1070    ENTER 709;C_volt(I)
1080  NEXT I
1090  FOR I=0 TO C_size
1100    ENTER 709;C_time(I)
1110  NEXT I
1120  !
1130  PRINT "                                                    TEMPERATURE DATA"
1140  FOR I=Index_a TO A_size-1
1150    PRINT USING "6D.3D,5(8D.3D)";A_time(I),A_temp(I*5),A_temp(I*5+1),A_temp
      (I*5+2),A_temp(I*5+3),A_temp(I*5+4)
1160  NEXT I
1170  FOR I=0 TO Index_a-1
1180    PRINT USING "6D.3D,5(8D.3D)";A_time(I),A_temp(I*5),A_temp(I*5+1),A_temp
      (I*5+2),A_temp(I*5+3),A_temp(I*5+4)
1190  NEXT I
1200  PRINT "                                                    FLOW DATA"
1210  FOR I=Index_b TO B_size-1
1220    PRINT USING "6D.3D,2(8D.3D)";B_time(I),B_flow(I*2),B_flow(I*2+1)
1230  NEXT I
1240  FOR I=0 TO Index_b-1
1250    PRINT USING "6D.3D,2(8D.3D)";B_time(I),B_flow(I*2),B_flow(I*2+1)
1260  NEXT I
1270  PRINT "                                                    VOLTAGE DATA"
1280  FOR I=Index_c TO C_size-1
1290    PRINT USING "6D.3D,3X,4(5D.6D)";C_time(I),C_volt(I*4),C_volt(I*4+1),C_volt
      (I*4+2),C_volt(I*4+3)
1300  NEXT I
1310  FOR I=0 TO Index_c-1
1320    PRINT USING "6D.3D,3X,4(5D.6D)";C_time(I),C_volt(I*4),C_volt(I*4+1),C_volt
      (I*4+2),C_volt(I*4+3)
1330  NEXT I
1340  GOTO 760
1350  END
```

Example 6: I/O Buffers

The HP 3852A input and output buffers allow the computer (over HP-IB) to periodically gather data from the instrument and download new set points without disturbing an on-going process in the instrument. In the following program, data generated by three run task subroutines is sent to the output buffer along with a header. This allows the computer to pull the data out of the buffer at anytime, using the header to distinguish which run task subroutine generated each data point.

www.valuetronics.com

HP 3852A Configuration

The slot numbers and channel ranges specified throughout the program correspond to the following HP 3852A configuration:

Slot 0 HP 44708A 20-Channel Relay Multiplexer (T/C)

Slot 1 HP 44701A Integrating Voltmeter

```
10   ! Sample Program 6
20   ! I/O BUFFERS
30   !
40   !Readings are taken by several tasks
50   ! Each task tags the data with a header number
60   !
70   DIM T1(5),T2(3)
80   CLEAR 7
90   WAIT 1
100  OUTPUT 709;"TSLICE .065;NTASKS 8,20;ENABLE MULTI"  !Set up multitasking
110  OUTPUT 709;"INBUF ON;OUTBUF ON"                    !Turn on HP-IB buffers
120  !
130  OUTPUT 709;"SUB T1"
140  OUTPUT 709;"USE 100"                               !Voltmeter is in slot 1
150  OUTPUT 709;"REAL T11(5)"                           !Temperature variable
160  OUTPUT 709;"WHILE 1"                               !Loop forever
170  OUTPUT 709;"SUSPEND 1"                             !Sleep for 1 second
180  OUTPUT 709;"CONF MEAS TEMPT 0-5 INTO T11"          !Measure temperatures
190  OUTPUT 709;"DISABLE EOL SWAP"                      !Disable swapping
200  OUTPUT 709;"VREAD 1;VREAD T11"                     !Read variables
210  OUTPUT 709;"ENABLE EOL SWAP"                       !Enable swapping
220  OUTPUT 709;"END WHILE"
230  OUTPUT 709;"SUBEND"
240  !
250  OUTPUT 709;"SUB T2"
260  OUTPUT 709;"USE 100"                               !Voltmeter is in slot 1
270  OUTPUT 709;"REAL T21(3)"                           !Temperature variable
280  OUTPUT 709;"WHILE 1"                               !Loop forever
290  OUTPUT 709;"SUSPEND 2"                             !Sleep for 2 seconds
300  OUTPUT 709;"CONF MEAS TEMPT 6-9 INTO T21"          !Measure temperatures
310  OUTPUT 709;"DISABLE EOL SWAP"                      !Disable swapping
320  OUTPUT 709;"VREAD 2;VREAD T21"                     !Read variables
330  OUTPUT 709;"ENABLE EOL SWAP"                       !Enable swapping
```


www.valuetronics.com

```
340   OUTPUT 709;"END WHILE"
350   OUTPUT 709;"SUBEND"
360   !
370   OUTPUT 709;"SUB T3"
380   OUTPUT 709;"URGENCY 2 25"                    !Set priority
390   OUTPUT 709;"USE 100"
400   OUTPUT 709;"REAL T31"                        !Temperature variable
410   OUTPUT 709;"WHILE 1"                         !Loop forever
420   OUTPUT 709;"SUSPEND .1"                      !Sleep for 1/10 second
430   OUTPUT 709;"CONF MEAS TEMPT 23 INTO T31"
440   OUTPUT 709;"DISABLE EOL SWAP"
450   OUTPUT 709;"VREAD 3;VREAD T31"
460   OUTPUT 709;"ENABLE EOL SWAP"
470   OUTPUT 709;"END WHILE"
480   OUTPUT 709;"SUBEND"
490   !
500   OUTPUT 709;"DISP OFF;RUN 0 T1;RUN 1
      T2;RUN 2 T3"                                 !Start subroutines
510   WHILE 1                                      !Loop forever
520     ENTER 709;H                                !Get data from instrument's
530   !                                              I/O buffers
540     IF H=1 THEN
550       ENTER 709;T1(*)
560       CONTROL 1;1,2
570       PRINT T1(*)
580     END IF
590     !
600     IF H=2 THEN
610       ENTER 709;T2(*)
620       CONTROL 1;1,6
630       PRINT T2(*)
640     END IF
650     !
660     IF H=3 THEN
670       ENTER 709;T3
680       CONTROL 1;1,10
690       PRINT T3
700     END IF
710     !
720   END WHILE
730   END
```


www.valuetronics.com

Appendix A
Specifications

The performance specifications for the HP 3852A and each plug-in accessory are found in the HP 3852S Data Acquisition and Control System Data Book and related supplements. The data book and supplements are available from your nearest Hewlett-Packard Sales and Support Office.

www.valuetronics.com

Appendix B
Error Messages

www.valuetronics.com

Appendix B

Error Messages

Introduction

Appendix B lists the error messages returned by the HP 3852A. An error message is displayed when a command specifies a parameter, defines a condition, or sets an operating state that is not allowed or not recognized by the HP 3852A. Error messages consist of an error code and a message. Messages have the format **ERROR dd: command: message**, where:

dd = a two-digit error code.

command = command which caused error to occur. (Command appears only if it is syntactically correct.)

message = message for the corresponding error code. The string of characters which caused the error is appended to the message. Message also includes additional information describing the error.

Error Messages

Error Code	Message	Description
0	NO ERROR	No error messages in the buffer when the error buffer is read.
1	OUT OF MEMORY	Not enough memory to do command listed in the error message.
2	SYMBOL TOO LONG	Array, variable, subroutine name, or displayed message is too long. Number specified is too long.
3	BAD NUMBER FORMAT	Number incorrectly specified (e.g. 1+.).
4	SYNTAX	Parameter specified is not a valid word, number, or character for that particular command.
5	SUBEND WITHOUT SUB	SUBEND command was encountered before the SUB command.
6	MISSING FOR	NEXT statement was enountered before the FOR statement.
7	NOT ALLOWED IN SUB	A command not allowed within a subroutine was encountered when the subroutine is being loaded.
8	ALLOWED ONLY IN SUB	Command allowed only within a subroutine was entered outside a subroutine.

www.valuetronics.com

Error Code	Message	Description
9	SUB CODE TOO LONG	Not enough available memory for the subroutine currently being entered.
10	SUB WAS DELETED	Subroutine that was called was previously deleted.
11	NO ACTIVE SUB	A subroutine was stepped or continued before it was paused or set up to be stepped.
12	CANNOT RETYPE A VARIABLE	Variable or array was assigned a format different from its original format.
13	MISSING IF	END IF or ELSE statement was encountered before the IF statement.
14	MISSING WHILE	END WHILE statement is encountered before the WHILE statement.
15	IMPROPER FOR/NEXT MATCHING	Loop counter variable names not the same (e.g. FOR I...NEXT J).
16	SUBSCRIPT OUT OF BOUNDS	Reading or writing to an array element greater than its maximum index.
17	END OF COMMAND INSIDE STRING	Message associated with the DISP command does not have ending quotes.
18	SYSTEM ERROR	Internal processor is in an illegal state.
19	INVALID CHAR RECEIVED	Programming character not recognized by the HP 3852A.
20	COMMAND BUFFER OVERFLOW	Too many parameters are specified or the parameter is specified by a complex numeric expression.
21	TOO MANY ARGS	Command used with multiple accessories where too many parameters are specified for that particular accessory.
22	CANNOT EXECUTE	Command cannot be executed as the HP 3852A is in local lockout.
23	SETTINGS CONFLICT	Command specifies a condition that is incompatible with the previously programmed accessory state.
24	ARGUMENT OUT OF RANGE	Parameter value specified is out of the valid range.
25	DEVICE FAILURE	Hardware failure.
26	POWER ON TEST FAILURE	Mainframe or an accessory failed the power on self test.
27	SELF TEST FAILED	Mainframe or an accessory failed the self test initiated by the TEST command. This message may occur if the self test is performed on an accessory that was installed with the power on. The HP 44701A may fail if the voltmeter is busy when the TEST command is issued.
28	INVALID SLOT	Slot address is incorrectly specified.
29	SPURIOUS FAST SCAN INTERRUPT	Can occur when the HP 44702A/B is installed with HP 3852A or HP 3853A power on, or may indicate a possible hardware failure.
30	SPURIOUS NORMAL SCAN INTERRUPT	Can occur when the HP 44701A is installed with HP 3852A or HP 3853A power on, or may indicate a possible hardware failure.

www.valuetronics.com

Error Code	Message	Description
31	INVALID COMMAND FOR ACCESSORY	Command is not used by the accessory whose slot or channel address was specified. Can also occur if a high-speed multiplexer is used for a backplane measurement while the ribbon cable is connected.
32	NO ACCESSORY PRESENT	Syntactically correct command is sent to an empty slot.
33	INVALID CHANNEL	Channel address is incorrectly specified.
34	INVALID REGISTER	Register address is incorrectly specified.
35	DIFFERENT PACKED TYPES	Data cannot be stored into a PACKED array containing readings with a different bit pattern or whose bytes per reading are not the same.
36	DATA LOST DUE TO FORMAT	Magnitude of the data returned cannot be represented in the format specified.
37	TRIGGER TOO FAST	Not reported.
38	CHECK POWER	An HP 3853A Extender is powered down or there are fluctuations in the line power.
39	MEMORY LOST	Occurs at power-on. Battery backed up memory lost power while the instrument was off.
40	CANNOT EXECUTE IN REMOTE	Command entered from the front panel cannot be executed while the HP 3852A is in remote.
41	CAN EXECUTE FROM FP ONLY	Command sent over the HP-IB can only be entered and executed from the front panel.
42	MATH ERROR	Indicates one of the following conditions: Real overflow, Real underflow, divide by zero, Integer overflow, SIN or COS argument, logarithm argument, square root argument, invalid Real number, BCD conversion, TYPE conversion.
43	END OF ARRAY REACHED	Amount of data written to the array is greater than the size of the array; however, the condition could not have been detected previously by the mainframe (e.g. error code 44). Error message is usually associated with real-time and post processing limit testing.
44	NOT ENOUGH VARIABLE SPACE	Array is not large enough or starting index is at a position where there isn't enough room left to store the data.
45	ARRAY NOT REAL	The domain and range arrays associated with the CONV command must be REAL arrays.
46	VARIABLE NOT DEFINED	Channel logging or real-time limit testing was enabled (ENABLE LOGCHAN, ENABLE LMT) before the necessary arrays were defined (LOGCHAN, LMT).
47	PACKED NOT ALLOWED	A packed array cannot be written into via the command.
48	ARRAY SIZES DIFFER	When performing post processing data conversions (CONV), the domain and range arrays must be the same size.
49	DATA OUT OF BOUNDS	Data associated with post processing data conversions (CONV) is outside the domain array.

www.valuetronics.com

Error Code	Message	Description
50	EMPTY ARRAY	Referencing or reading from a deleted array. Also occurs when you read a packed array that has not been written to.
51	SYMBOL TABLE FULL	Too many variables, arrays, and subroutines are presently defined.
52	SCAN IN PROGRESS	Voltmeter configuration cannot be changed while it's scanning a list of channels.
53	NO SCAN LIST	The list of channels to be scanned were not specified in previous commands.
54	NO VALID CHANNEL IN LIST	Channel or channel list did not include any channel for which this command is valid.
55	STRUCTURED COMMANDS NESTED TOO DEEP	The maximum number of nested BASIC constructs (FOR...NEXT, IF...END IF, WHILE...END WHILE) is 10.
56	SUBEND IN STRUCTURED COMMAND	The SUBEND command cannot reside within a FOR...NEXT, IF...END IF, or WHILE...END WHILE BASIC construct.
57	LIST TOO LONG	Too many items of a parameter were specified (e.g. more than 10 channels were listed individually - 1,2,3,...).
58	SUBS NESTED TOO DEEP	The maximum number of nested subroutines is 10.
59	SUB ALREADY EXISTS	The subroutine name specified already exists.
60	ACCESSORY INTERFACE ERROR	Mainframe/accessory interface or configuration error. Indicates a potential hardware failure.
61	CALIBRATION RAM ERROR	Bad CAL RAM or voltmeter out of calibration.
62	CALIBRATION FAILURE	Accessory unable to be calibrated.
63	SCAN LIST TOO BIG	Follows the CLWRITE or MEAS command with TERM = RIBBON for the HP 44702A/B where NRDGS x (# of channels - 1) is > 4095.
64	MUST USE DIFFERENT VARIABLES	When using the data processing commands CONV, LMT, and SCALE, the respective variables or arrays must have unique names.
65	NO RESPONSE - ACCESSORY REMOVED?	Accessory did not respond to command. Can be caused by removing an accessory with the power on.
66	INVALID CHANNEL FOR COMMAND	Channel address specified was not capable of executing the command.
67	OVERVOLTAGE ON BACKPLANE	Indicates a voltage on the backplane approximately equal to ±25V was sensed by the HP 44702A/B when TERM was set to INT. Its inputs are then disconnected.
68	SUB NAME NOT EXPECTED	A subroutine name appeared in a command where a subroutine name is not allowed.
69	SCALAR NAME NOT EXPECTED	A variable name appeared in a command where a variable is not allowed.
70	ARRAY NAME NOT EXPECTED	An array name appeared in a command where an array is not allowed.

www.valuetronics.com

Error Code	Message	Description
71	UNDEFINED WORD	A word that is not a variable, array, or subroutine name, or a command header appeared in the command.
72	THIS KEYWORD NOT EXPECTED	Command header appeared in a command where another command header is not allowed.
73	NO READINGS TO TRANSFER	No readings were available when the XRDGS command was executed.
74	COMMAND END NOT EXPECTED	Incomplete command sent (too many or too few parameters).
75	INSIDE SUB CALLED MORE THAN ONCE	Cannot have a PAUSE statement that is inside a subroutine which is called more than once.
76	INSIDE NESTED SUB	A PAUSE statement cannot be used inside a nested subroutine.
77	NOT ALLOWED WHILE STORING SUB	SCRATCH command cannot be executed over the HP-IB while a subroutine is being stored from the front panel.
78	NOT ALLOWED DURING HP-IB COMMAND	SCRATCH command cannot be executed from the front panel while a command is partially entered over the HP-IB.
79	STANDARD DEVIATION NOT DEFINED	STAT command is executed and the var array has a maximum index of 1.
80	—	Power on test failed; instrument locks up. See the HP 3852A Assembly Level Service Manual.
81	TOO MANY READINGS REQUESTED	Command requested > 134217727 readings.
82	SYMBOL ALREADY EXISTS	Variable, array, or subroutine being declared from the front panel and over the HP-IB at the same time will only be accepted from one source. This message is returned to the other.
83	PROGRAM QUEUE FULL	The program queue cannot hold another subroutine name. Occurs on execution of the RUN command when the queue set by NTASKS is full.
84	RUN TASK DOES NOT EXIST	The run task number specified has not been created by the RUN command.
85	SUB ACTIVE	A subroutine executing in a task environment cannot be deleted (DELSUB command).
86	MULTITASKING NOT ENABLED	The command entered can only execute in the multitasking mode.
87	TASK NOT PAUSED	The run task targeted by the CONT command is not paused.
88	TOO MANY RUN TASKS	The number of run tasks created by the RUN command exceeds the number of run task environments specified by NTASKS.
89	ACCESSORY BUSY	A move is in progress.
90	NO ACTION DEFINED	TRIG SGL encountered without a corresponding MOVE or SUSTAIN command preceeding it.
91	MUST STOP TO CHANGE DIRECTION	A running SUSTAIN command must be stopped with HALT or SUSTAIN 0 before a command to reverse direction can be executed.
92	NOT VALID IN WIDTH MODE	A MOVE command cannot be executed when the PROFILE command is in the width mode.
93	TERMINAL CARD TEST JUMPER SET	Will not execute a MOVE or SUSTAIN command when test jumper is in the TEST position.

www.valuetronics.com

Error Code	Message	Description
94	CANNOT MOVE WHILE STANDBY ON	Standby is powering down the motor
95	POWER OUTPUT IS CURRENT LIMITING	QPWR output is current limiting.
96	REQUIRED PARAMETER MISSING	Required parameter was not specified.
97	INSUFFICIENT ACCESSORY MEMORY	There is not enough unused memory on the accessory channel to perform the requested task.
98	WAVEFORM ALREADY EXISTS	A waveform cannot be overwritten. The existing waveform must first be deleted.
99	WAVEFORM NOT DEFINED	The waveform specified cannot be applied or used as a source of information since it has not been defined and stored.
100	INVALID ELEMENT SUBRANGE	The subrange specified or implied exceeds the number of successive waveform points available. Or, the LAST point specified in the command preceeds the FIRST point specified.
101	WAVEFORM IN USE	The waveform cannot be deleted because it is currently the active waveform. Setting TARM OFF or selecting a different waveform will enable the waveform to be deleted.
102	NO WAVEFORM SELECTED	The waveform specified in the command has not been selected (APPLY WFV), or was previously deleted.
103	WRONG ARRAY TYPE	The type (Real, Integer, Packed) of array specified cannot be used by the command.
104	WRONG PACKED TYPE	The data in the Packed array specified is in a format that cannot be used by the accessory.
105	SINGLE ELEMENT WAVEFORM NOT ALLOWED	Waveforms must contain at least two amplitude points.
106	DATA ALTERED - WAS OUT OF RANGE	One or more amplitudes or number of time base intervals received was outside of the allowable range. An acceptable value is used in place of each such value and the waveform is modified accordingly.
107	HARDWARE DOES NOT SUPPORT COMMAND	The command requires the 03852-66523 controller module.
108	ARRAY NOT INTEGER	An array required by the command must be an Integer array.
109	PATH NAME NOT EXPECTED	Attempting to use a path name where not allowed.
110	IMPROPER FILE NAME	File names are limited to 10 characters. Foreign characters are allowed, but punctuation is not.
111	IMPROPER DEVICE TYPE	The msus has the correct general form, but the characters used for a device are not recognized.
112	IMPROPER MSUS	The characters used for a msus do not form a valid specifier.
113	UNSUPPORTED DRIVE TYPE	Drive does not use the CS/80 or SS/80 command set.
114	UNSUPPORTED SECTOR SIZE	Sector size too large. Must be 256 bytes/record. Sectors larger are not supported.
115	DRIVE NOT FOUND OR BAD ADDRESS	The msus contains an improper device selector, or no external disc is connected.

www.valuetronics.com

Error Code	Message	Description
116	INVALID UNIT NUMBER	The msus contains a unit number that does not exist on the specified device.
117	INVALID MASS STORAGE PARAMETER	A mass storage statement contains a parameter that is out of range, such as a negative record number or an out of range number of records.
118	MEDIA CHANGED OR NOT IN DRIVE	Either there is no disc in the drive or the drive door was opened while a file was assigned.
119	MEDIA IS WRITE PROTECTED	Attempting to write to a write-protected disc.
120	DIRECTORY FULL	Although there may be room on the media for the file, there is no room in the directory for another file name.
121	NO ROOM ON DISK	There is not enough contiguous free space for the specified file size. The disc is full.
122	FILE NOT FOUND	The specified file name does not exist in the directory. Check the contents of the disc with the CAT " " command.
123	DUPLICATE FILE NAME	The specified file name already exists in directory. It is illegal to have two files with the same name on one volume.
124	IMPROPER FILE TYPE	Many mass storage operations are limited to certain file types.
125	PATH NAME NOT ASSIGNED	Must assign path name before its use.
126	FILE OPEN	The specified file is assigned an I/O path name which has not been closed.
127	END OF FILE FOUND	No data left when reading a file, or no space left when writing a file.
128	INITIALIZATION FAILED	Too many bad tracks found. The disc is defective, damaged, or dirty.
129	MASS STORAGE SYSTEM ERROR	Usually a problem with the hardware or the media.
130	BAD SELECT CODE	No HP-IB card in slot, slot is bad, or select code is >30.
131	I/O OPERATION NOT ALLOWED	Attempting to use a CLEAR, TRIGGER, or SPOLL command with a path assigned to a file.

www.valuetronics.com

Index

www.valuetronics.com

D

www.valuetronics.com

www.valuetronics.com

www.valuetronics.com

www.valuetronics.com

www.valuetronics.com

www.valuetronics.com

www.valuetronics.com

HEWLETT PACKARD

9282-1078
Binder Only Part No

Made in U S A