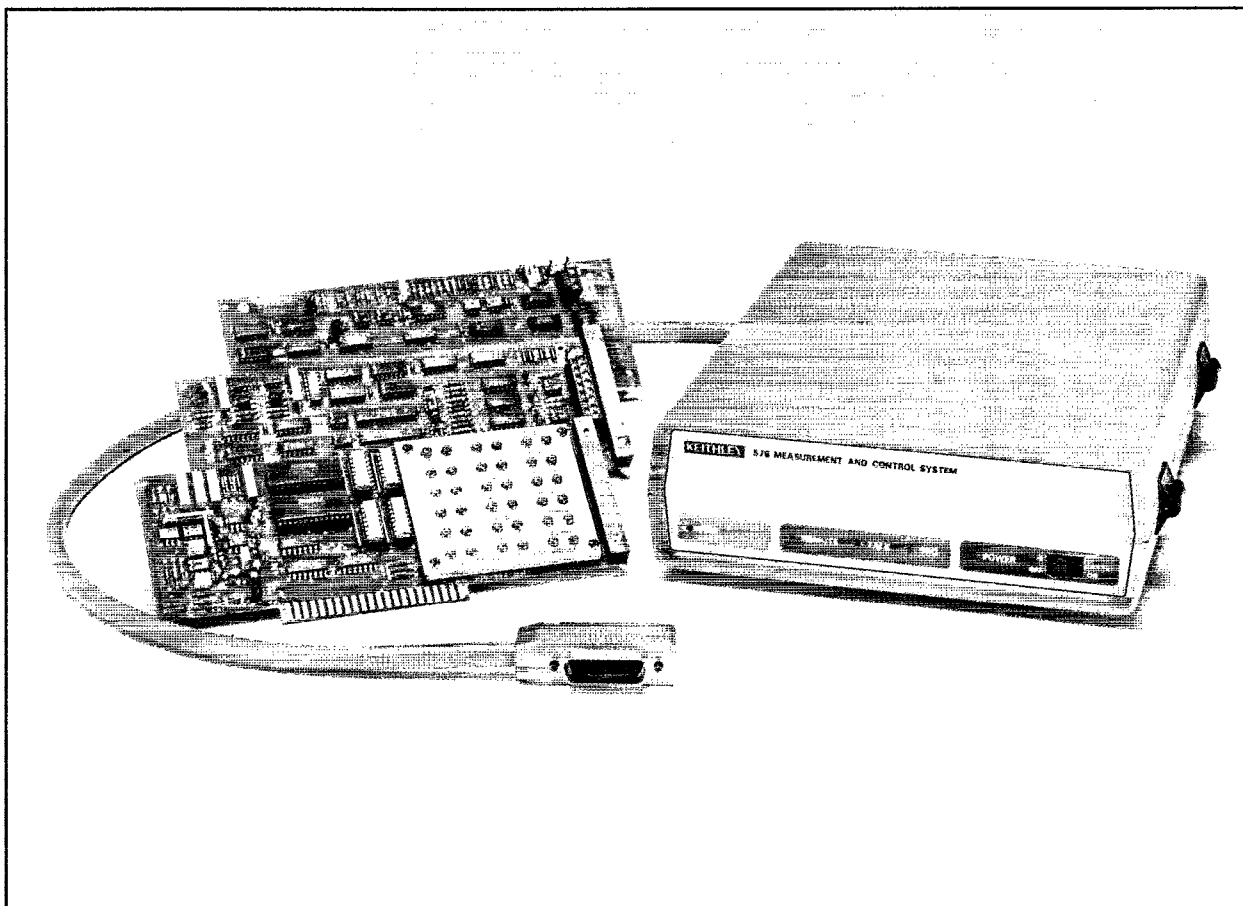


KEITHLEY DATA ACQUISITION

Model 576 High Speed Data Logging System

For GPIB Controllers, or Computers with GPIB Control Capability



Publication Date: October 1991
Document Number: 576-901-01 Rev. C

Third Edition (October 1991)

All rights reserved. No part of this manual may be reproduced in any form or by any electronic means, including information storage and retrieval systems, without permission in writing from Keithley. Changes are made periodically to the information contained herein. These changes will be incorporated in the new edition of this publication.

Keithley warrants this product to be free from defects in material and workmanship for a period of 1 year from date of shipment.

Keithley warrants the following items for 90 days from the date of shipment: probes, cables, rechargeable batteries, diskettes, and documentation.

During the warranty period, we will at our option, either repair or replace any product that proves to be defective.

To exercise this warranty, write or call your local Keithley representative, or contact Keithley in Taunton, MA. You will be given prompt assistance and return instructions. Send the product, transportation prepaid, to the indicated service facility. Repairs will be made and the product returned, transportation prepaid. Repaired or replaced products are warranted for the balance of the original warranty period, or at least 90 days.

DISCLAIMER OF WARRANTIES AND LIABILITY

The information contained in this manual is believed to be accurate and reliable. However, Keithley assumes no responsibility for any errors, omissions, or inaccuracies whatsoever. Without limiting the foregoing, KEITHLEY DISCLAIMS ANY AND ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING THE WARRANTY OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE INFORMATION CONTAINED IN THIS MANUAL AND THE SOFTWARE DESCRIBED HEREIN. The entire risk as to the quality and performance of such information and software is upon the buyer or user. Keithley shall not be liable for any damages, including special or consequential damages, arising out of the use of such information or software even if Keithley has been advised in advanced of the possibility of such damages. The use of the information contained in the manual and software described herein is subject to Keithley, standard license agreement, which must be executed by the buyer or user before the use of such information or software.

NOTICE

Keithley reserves the right to make improvements in the product described in this manual at any time and without notice.

Copyright © 1990, Keithley Instruments, Inc.
Data Acquisition Division
440 Myles Standish Blvd.
Taunton, MA 02780
1-508-880-3000

All Keithley product names are trademarks or registered trademarks of Keithley Instruments, Inc. Other brand and product names are trademarks or registered trademarks of their representative holders.

WARNING

This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

All Keithley product names are trademarks or registered trademarks of Keithley Instruments, Inc.

Other brand and product names are trademarks or registered trademarks of their respective holders.

Model 576 Errata

This addendum contains new information covering the Keithley Model 576. Please add the attached new pages to your 576 manual, and read the following carefully.

AMM Module Installation for Model 576 Combo Systems

The Model 576 is available in "combo" packages with the AMM module and MEM option (if ordered) installed at the factory, and any applicable ZCAL or SYST :CAL calibration steps already performed.

If your Model 576 came with the AMM module factory-installed, you may ignore the following sections of the manual during system set-up:

- a. Portions of QuickStart Steps 2 and 3 which discuss installing the AMM module and CA-85 cable. These components should already be in place. You will, however, need to install any option module you have ordered.
- b. The portion of the QuickStart Section which discusses "Applying the AMM Calibration Constant...." The 576 has been calibrated at the factory.
- c. Any information in the manual or Appendix section pertaining to ZCAL.

If you have any questions or difficulties concerning unpacking or setting up the Model 576 Combo package, contact the product support department at Keithley Instruments, Inc., Data Acquisition Division at (508)880-3000.

Using RESET ALL with Fast Computers and ROM-Based GPIB Adapters

With some types of GPIB adapter cards and computers, and beginning with firmware revision E02, you must include a time delay of 2 seconds immediately after a RESET ALL command or the Model 576 may issue an error message on the next 576 command.

Specifically, GPIB interface cards having their operating software stored in ROM may encounter a timing problem when RESET ALL is issued. Affected cards include some CEC and National products. This situation has not been observed with GPIB boards using driver software loaded from diskette (e.g. IOtech Personal 488). It also does not apply to the RESET MEM or RESET OUT commands.

The need for the delay results from RESET ALL performing several additional tasks in revision E02. Normally, a program should check the status of bit 7 in the 576 SPOLL byte after any 576 command, before the next 576 command is issued. The 576 sets this bit while RESET ALL is busy. However, a fast computer with ROM-based GPIB interface may be fast enough to read the SPOLL byte and return a 0 ("not busy" status) before the 576 can actually set the bit. This results in an "Invalid command..." error as the system attempts to execute the 576 command following RESET ALL.

To eliminate the possibility of problems, follow any RESET ALL command with a 2 second delay. In BASICA or QuickBASIC, this is done as follows:

```
t!=TIMER: WHILE TIMER-t!<2: WEND
```

The delay time is not critical as long as it is sufficient. If, following RESET ALL, the computer executes other non-GPIB activities requiring 2 seconds or more, the delay may not be necessary. However, using the delay will preclude any problems, regardless of any computer or interface type ultimately used with the program.

New Command — ZCAL

A new command "ZCAL" enables a user to determine the zener reference cal constant for an AMM1A or AMM2 module plugged into the Model 576. ZCAL is available in all versions of the Model 576 firmware. Detailed instructions for the use of ZCAL are attached to this addendum.

576 Power Transformers

The power transformers available for the Model 576 now include units for European, Japanese, and U.S. domestic power mains:

Keithley P.N.	Input	Output	Application
PS-29	125VAC, 50/60Hz	12VAC @ 3A	U.S.
PS-31	220VAC, 50Hz	12VAC @ 4.2A	Europe
PS-33	100VAC, 50Hz	12VAC @ 3.3A	Japan

CHANNEL AMPLITUDE, DUTY, and FREQUENCY Commands

The CHAN :AMPL, :DUTY, and :FREQ commands are grouped together at the end of the CHAN commands. They pertain to the WAV1 module, but are alphabetically out of order with the rest of the CHAN commands.

Calibrating the $\pm 10V$ Reference — Section 8

The "Calibrating the $\pm 10V$ Reference" section discusses a " $\pm 10V$ Reference". The voltage references actually consist of a +10V reference on the AMM1A or AMM2 module, and a mirror circuit on the 576 motherboard which generates the -10V reference. This point is unclear in the manual. The procedure is correct, however.

Appendix C — Changes to Accommodate ROM-Based GPIB Interfaces

All programs in Appendix C have been modified to provide a 2 second delay after RESET ALL. This topic is described above. Depending on the revision level of the 576 Utility disk, the programs in the QCKSTART directory of the 576 Example/Utility diskette may not have been modified. Check them.

Quick Start Guide for the Keithley Model 576

SECTION 1 — Introduction

SECTION 2 — Terms, Conventions, Safety Considerations

Terms, Conventions, Safety Considerations	2-1
A Few Precautions	2-2
Designing Safe Control Set-ups	2-2
In Case of Problems or Malfunctions	2-3
Using this Manual	2-4

SECTION 3 — Setup

Unpacking Your System	3-1
Setting Up	3-2
Completing Setup	3-5
In Case of Problems or Malfunctions	3-5

SECTION 4 — Hardware Configuration

Self-ID Feature	4-1
Changing Default Configuration Parameters	4-2
Hardware Switch Configuration	4-2
Setting Up Measurement and Control	4-2
Signal Connections	4-2
“Channel” and “Slot”	4-3
Analog Input — Slot 1	4-7
Trigger — Slot 2	4-7
Setting Jumpers for Trigger Modes	4-10
Option Slot — Slot 3	4-10
Analog Output — Slot 4	4-10
Digital I/O — Slot 5	4-11
Power Control — Slot 5	4-11
External Input — Slot 6	4-12
Advanced Topics	4-12

SECTION 5 — 576 Commands

Introduction and Overview	5-1
Configuration	5-1
Data and Memory Management	5-2
Triggering and Program Control	5-2
Timestamping	5-3
Subroutines	5-3
Engineering Units Conversion and Data Formats	5-3
Using this section of the manual	5-3
Some important notes on 576 commands	5-3

Terms and Conventions used in this Section	5-4
576 PROGRAMMING COMMAND SET	5-4
BUFF CLEAR	5-6
BUFF DIM	5-7
BUFF INDEX	5-9
BUFF MOVE	5-10
BUFF READ	5-11
BUFF STAT	5-13
BUFF WRITE	5-14
CALL	5-15
CHAN	5-16
CHAN :FILTER	5-17
CHAN :GAIN	5-18
CHAN :MODE	5-20
CHAN :OFFSET	5-22
CHAN :RANGE	5-23
CHAN :FUNC	5-24
CHAN :AMPL	5-25
CHAN :DUTY	5-26
CHAN :FREQ	5-27
DEBUG	5-28
DO LOOP	5-29
HALT	5-30
IF ELSE ENDIF	5-31
IREAD	5-34
IWRITE	5-35
ONINT INTOFF	5-36
PEEK	5-38
POKE	5-39
READ	5-40
READ	5-42
RESET	5-43
SUBR RETSUB ENDSUB	5-44
SYST	5-45
SYST :AMM	5-46
SYST :BUF	5-47
SYST :CAL	5-48
SYST :CLOCK	5-50
SYST :EOI	5-51
SYST :ERR	5-52
SYST :FORM	5-53
SYST :IDN	5-54
SYST :MEM	5-55
SYST :PEEK	5-56
SYST :POKE	5-57
SYST :SAVE	5-58
SYST :SLOT	5-59
SYST :SRQ	5-60

SYST :STAMP	5-61
SYST :TERM	5-62
SYST :TRIG	5-63
SYST :UNIT	5-64
TEST	5-65
TRIG — HARDWARE TRIGGER	5-66
WAIT	5-68
WAVE	5-69
WHILE WEND	5-70
WRITE	5-72
X	5-73
ZCAL	5-74
Internal Data Buffer Description	5-77
Data Transfer Modes	5-78
Device Status (SPOLL) Byte	5-81
Serial Poll Bit Test	5-81

SECTION 6 — Applications for the Keithley Model 576

Introduction	6-1
Battery Life Testing	6-1
Lab Acquisition with the Model 576 and ASYSTANT GPIB	6-4
Multi-channel Voltage Measurements with Hardware Triggering	6-8
Process Monitoring and Control: Triggering and Time Stamping	6-9
Portable and Remote Acquisition	6-12
Testing 8-bit DACs with the Model 576	6-15
Low Current Measurements Using the Model 576 with a GPIB Current Amplifier	6-17
Data Logging with the Apple Macintosh	6-18
High Speed Data Transfers	6-21

SECTION 7 — 576 REFERENCE

INTRODUCTION	7-1
THEORY OF OPERATION	7-1
Overall Functional Description	7-1
Theory of Operation	7-5
System Control Circuitry	7-5
Power Supply	7-5
MODEL 576 MOTHER BOARD	7-6
Commands	7-10
Mother Board Functions	7-10
Analog Input	7-11
Gain	7-12
Programmable Filter	7-12
Analog-to-Digital Conversion	7-12
Analog Input Command Locations	7-12
Analog Trigger	7-15

Trigger Command Locations	7-16
Option Slot	7-18
Installation	7-18
Option Slot Command Locations	7-18
Analog Output	7-19
Output Limitations	7-19
Automatic Register Sequencing	7-20
Analog Output Command Locations	7-20
Digital Input and Output	7-23
Digital Input Terminals	7-23
Digital I/O Command Locations	7-24
Relay Control	7-26
Relay Board	7-26
Relay Control Command Locations	7-26
External Input	7-28
External Analog Input Command Locations	7-28
Command Locations in Numeric Order	7-30
CMD1A SELECT A/D CHANNEL	7-30
CMD1B SELECT SLOT	7-30
CMD2A ANALOG TRIGGER AND IRQ CONFIGURATION	7-30
CMD2B ANALOG TRIGGER INPUT CONFIGURATION	7-31
CMD3A OPTION SLOT COMMAND A	7-31
CMD3B OPTION SLOT COMMAND B	7-31
CMD4A D/A CONTROL FOR ANALOG OUTPUT	7-31
CMD4B D/A DATA FOR ANALOG OUTPUT	7-31
CMD5A DIGITAL I/O PORT SELECTION AND CONFIGURATION	7-32
CMD5B DIGITAL I/O DATA	7-32
CMD2C ANALOG TRIGGER VOLTAGE (0-255) COUNTS	7-32
CMD3C UNSPECIFIED OPTION SLOT COMMAND	7-33
CMD1C GLOBAL GAIN	7-33
CMD1D A/D START/STATUS	7-33
SLOT 1 HARDWARE IDENTIFICATION	7-33
STROBE GLOBAL ANALOG OUTPUT UPDATE	7-33
GLOBAL GLOBAL COMMAND 1	7-33
31 RESET	7-33
Self-ID Circuitry	7-33
Considerations for Older, Non-Self-ID Modules	7-34
576-Series Measurement and Control System Specifications	7-34

SECTION 8 — Calibration, Maintenance, and Troubleshooting

Introduction	8-1
CALIBRATION INFORMATION	8-1
When Should you Calibrate	8-1
Environmental Conditions	8-2
Recommended Calibration Equipment	8-2
Calibrating the AMM1A or AMM2	8-2

Calibrating the +5 Volt Supply	8-2
Calibrating the ± 10 Volt Reference	8-2
TRUBLESHOOTING INFORMATION	8-3
Isolating the Problem	8-3
System Checks	8-3
Power Supply Checks	8-4
Mother Board Checks	8-4
Signal Checks	8-4
Analog Input Modules	8-6
Analog Output	8-6
Digital Modules	8-6
SPECIAL HANDLING OF STATIC SENSITIVE DEVICES	8-6
REPLACEABLE PARTS	8-7

SECTION 9 — OPTION MODULES AND INTERFACES

Introduction	9-1
Keithley Module Library	9-1

APPENDICES

APPENDIX A Device Clear and Interface Clear	A-1
APPENDIX B Error Conditions and Messages	B-1
APPENDIX C QuickStart Program Listings	C-1
APPENDIX D IEEE-488 Bus Overview	D-1
APPENDIX E Calibration of the Model 576 Using the SYST:CAL and ZCAL Commands ..	E-1

Quick Start Guide for the Keithley Model 576

The Keithley Model 576 includes full documentation for unpacking, installing, and operating the hardware. For best results, you should read this information prior to working with your computer and the Model 576. If you are in a hurry to set up the Model 576, you may use the following "Quick Start" Guide and the supplied example programs to set up and check out the system. However, you should still perform a complete installation according to the 576 manual after the Quick Start.

These instructions assume that you are using an IOtech, CEC, or National GPIB interface in conjunction with an IBM-compatible PC or PS/2 computer and Interpreter (Advanced or GW) BASIC. If you are using another controller/language combination, you may use the supplied example programs as guides to writing your own programs.

These programs simultaneously use the source and measurement capabilities of the Model 576 to generate and measure signals. You will need eight 4-inch lengths and one 18-inch length of 18-24 ga. hook-up wire stripped 3/16" on each end to make connections

Step 1 — Make sure your GPIB interface is working

If you are using a personal computer with GPIB interface, install the interface and confirm that it is working properly. Refer to the computer and interface manufacturer's documentation for details.

Step 2 — Unpack the Keithley hardware

Unpack the Model 576, power supply, and AMM1A or AMM2 module. The AMM module includes a small metal right-angle bracket which secures the rear corner of the module to the rear panel of the Model 576. Do not install the bracket at this time. Put it in a safe place; you will need it later. If you have a module for the 576 option slot, also set it aside for now.

Step 3 — Configure the Model 576 and AMM module.

Refer to the attached diagram. Position the jumper J3 on the AMM module over pins 1 and 2. Position the jumper W201 on the Model 576 mother board over pins 1 and 2. Install the trigger cable CA-85-1 from J7 on the AMM module to J201 on the 576 mother board. Install the AMM module in the lower expansion slot in the Model 576. This is a temporary installation — the metal right-angle bracket will have to be installed later.

Step 4 — Check the Model 576 address switches

The address switch is located on the rear of the Model 576 motherboard. The factory default address is 3 (switches 1 and 2 ON, and switches 3, 4, and 5 OFF). You may use a different address, but all example programs for the 576 use address 3.

Step 5 — Connect the computer and Model 576

Attach a GPIB cable with metric threads to the rear panel of the Model 576. Connect the other end of the cable to the

GPIB interface in the PC. Do not connect any other GPIB equipment to the GPIB bus at this time.

Step 6 — Connect the power supply to the Model 576

Turn the Model 576 power switch OFF. Verify that you have the proper power supply for your local mains. Plug the power supply into an AC mains outlet. Plug the power connector at the end of the transformer cable into the power input jack in the 576 rear panel.

Step 7 — Turn on the equipment

Turn on the computer and the Model 576. The computer should boot up as usual. The Model 576 should come on with only the power light illuminated. The "RUN" light will flash three times during self-check. If the SRQ lamp flashes continuously or remains on, turn off the equipment and recheck your work. Refer to the startup section of the 576 manual. If you cannot locate the problem, contact Keithley technical support.

Step 8 — Load your system GPIB support software

Load the driver for your GPIB interface. This may require running a driver program from disk, or the driver may reside in ROM-based firmware which loads automatically when you power up the controller. See documentation for your GPIB interface and driver software for details.

Step 9 — Run GWBASIC

The supplied example programs are designed to run under Microsoft GW-BASIC or IBM BASICA, version 3.0 or later.

Step 10 — Load and run the example programs

Connect the Model 576 analog and digital I/O channels as shown in the Figure. Insert the Model 576 Quick Start diskette into the disk drive on your computer. The supplied programs will perform the following tasks:

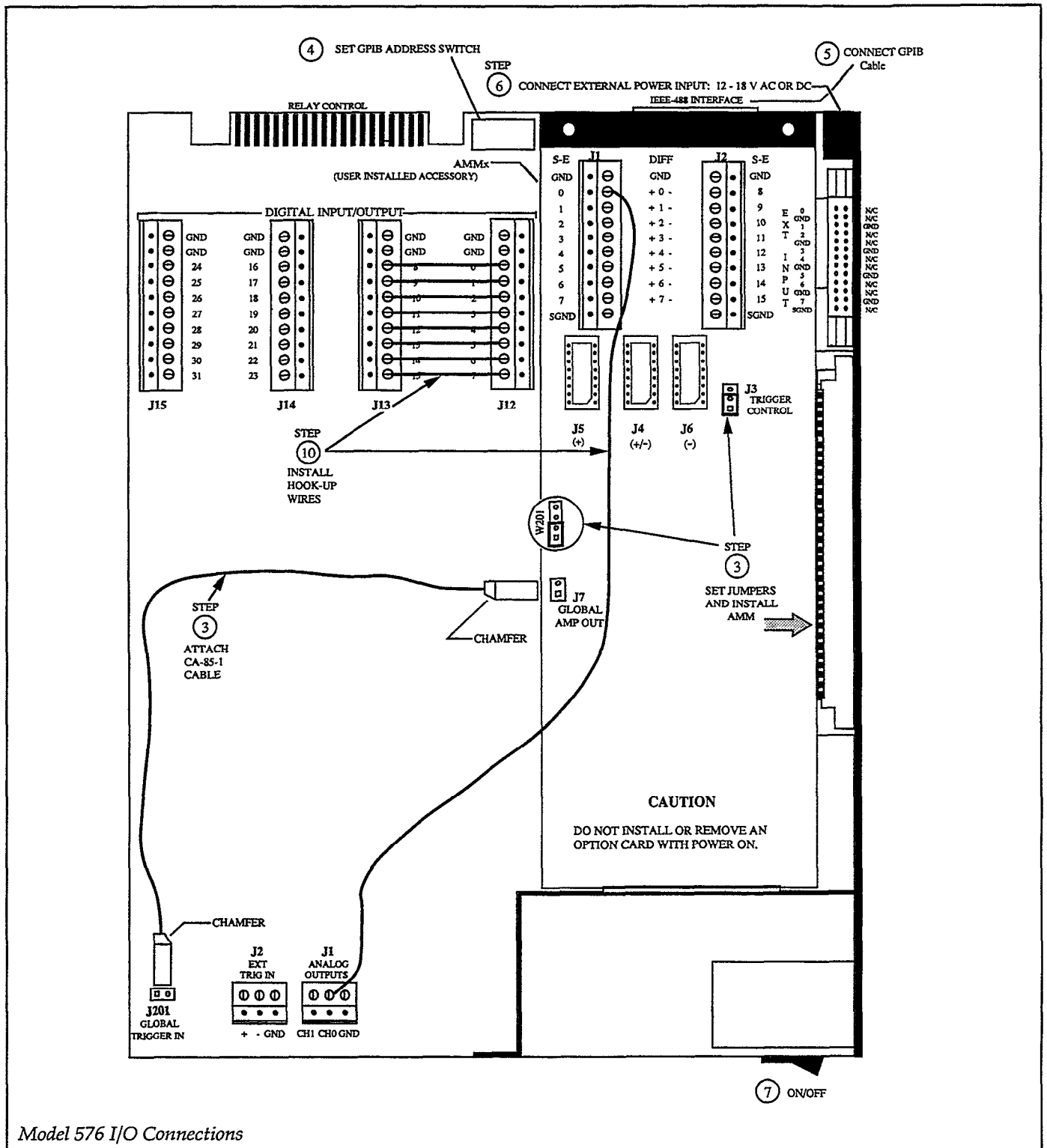
ANALOG.xxx — Analog input/output program. Output portion outputs a voltage from the 576 analog output channel. Input portion uses 576 analog input channel to collect data from output channel. Data is time stamped and retrieved into the computer.

DIGITAL.xxx — Digital input/output program. Digital port B is programmed for output; port A for input. Output port steps through 0-255 while input port displays the byte value read at the port A.

In each case, "xxx" will be "IOT" for an IOtech interface, "CEC" for a CEC GPIB interface, "NAT" for a National GPIB interface, "MBC" for a Metrabyte GPIB interface, or "SER" for the 500-SERIAL adaptor. See the appendix section for program listings.

Step 11 — Reinstall the Model 576 using the manual

The manual provides all the details for installation and operation of the Model 576.



Applying the AMM Calibration Constant to the Keithley Model 576

AMM1A and AMM2 modules shipped for use with the Model 576 bear a calibration sticker carrying a 5-digit number. This number is the calibration constant for use with the 576 "SYST :CAL" command. Do not confuse the calibration sticker with the 6-digit serial number, which is printed on a metal foil label.

When properly entered, the calibration constant configures and enables the automatic gain and offset correction feature of the AMM1A and AMM2 global amplifiers. This enables the AMM module to achieve the "corrected" specifications for analog measurements as published in the Model 576 manual. If the calibration factor is not entered, the "uncorrected" accuracy specifications will apply.

An important benefit of the Calibration factor concerns measurements made with an analog input module in the option slot. With the calibration factor applied, the accuracy of such measurements will be the accuracy of the module itself. It will not be necessary to factor in the specifications of the AMM module.

If your AMM module does not have a calibration sticker attached, contact Keithley DAC Technical Support for further instructions.

Entering the AMM Calibration Constant

The Model 576 can retain calibration data in its configuration memory. Thus, it is necessary to enter the factor only when:

1. you first set up the system,
2. you physically recalibrate your AMM module and determine a new calibration factor, or
3. you install a new AMM module having a different calibration constant.

The RESET command also clears any calibration constant previously stored in the 576.

Use the following procedure to enter the calibration constant and calibrate the 576:

1. Open the Model 576 and set switch number 8 on the address/SAVE switch to ON.
2. Send the following commands to the Model 576 ("xxxxx" is the AMM calibration constant):

```
SYST :SAVE CNFG;  
SYST :CAL xxxxx;  
SYST :CAL;
```

3. Confirm that the calibration factor has successfully been stored. First, turn off the power to the Model 576, and then turn the unit back on. Issue the following command to the Model 576:

```
SYST :SAVE ?;
```

The Model 576 should respond:

```
SAVE CNFG
```

4. Confirm that the calibration was successful by issuing the following command to the Model 576:

```
SYST :CAL ?;
```

The Model 576 should respond:

```
AMM CAL xxxxx <corr1>,<corr2>,...<corr10>
```

See documentation for the SYST :CAL command in the 576 manual for details. If you do not obtain the expected messages, review your work.

Note: The Model 576 will retain the calibration data when switched off *only* if switch 8 of the Address/SAVE Model switch is ON and the 576 has been issued a "SYST :SAVE CNFG;" command.

Model 576 Notes

Input Resistor Pin-Out Error

The input resistor sockets J4, J5, and J6 shown on the overlay inside beta-version and early production Model 576 units incorrectly indicate the position of pin 1. The correct position of pin 1 is shown in the diagrams in the 576 manual and AMM module manuals.

Use of the CAL Command

The Model 576 firmware includes a "SYST :CAL" command which enables the self-calibration feature of the AMM1A and AMM2 global amplifiers. This command is intended to compensate for the cumulative effects of time and temperature drift. The SYST :CAL command will not enhance the performance of these modules beyond specifications. The effect of SYST :CAL may not be dramatic or even noticeable if the AMM module is already operating within or close to specifications.

Additional QuickStart Programs

The 576 Example Disk also includes ANALOG.SER and DIGITAL.SER QuickStart programs for use with the Keithley 500-SERIAL adapter. These programs are not mentioned in the QuickStart section, but they are described on the disk's README.1ST file.

Use of "Immediate" vs "Program" Commands

The choice of when to use IMMEDIATE versus PROGRAM mode commands may not seem obvious to first-

time users of the Model 576, or even to seasoned users of traditional GPIB instruments.

IMMEDIATE commands cause the Model 576 to behave most like traditional GPIB instruments, and provide the easiest means of using the system. Immediate commands execute at once when they are sent to the Model 576. An immediate data read (IREAD) results in the data being instantly placed in the Model 576 output queue where it can be read via the GPIB bus. Similarly, an immediate data write (IWRITE) outputs the data directly to the selected channel. In each case, there is no buffering or use of the 576's internal data memory. System configuration (SYST) commands are also immediate in that the configuration instructions are written directly to the system memory.

PROGRAM commands facilitate more intelligent operation for the 576, including subroutines, conditional triggering, automatic program restart, and other features beyond the capabilities of most GPIB instruments. PROGRAM commands are stored in the 576's program memory, and do not take effect until an Execute (X) instruction is encountered in the program. PROGRAM mode input and output commands work in conjunction with the 576's data buffers, and the firmware also includes commands which dimension, read, and write data to the buffers.

Refer to the Example Disk programs and the Applications section of the Model 576 manual for examples IMMEDIATE and PROGRAM modes.

Battery Replacement

The Model 576 contains a battery which is used to backup data and programs stored in system RAM. Under normal conditions, the battery should last for a minimum of five years with the standard 128k memory, or 2-1/2 years with the 512k memory option.

This battery is an internal part of the socket holding the system data/program memory chip U304, and is not user-serviceable. The socket must be replaced when the bat-

tery is exhausted. This condition will become evident when the Model 576 no longer retains the correct time when the system is switched off.

The socket is Keithley part number SO-129, or Dallas Semiconductor part DS-1216D.

See topics in the Reference and Maintenance sections concerning maintenance and servicing of the battery, data memory chip, and socket.

SECTION 1

Introduction

Welcome to the Model 576 GPIB High Speed Data Logging System.

The 576 is a general-purpose data acquisition and control device – an interface between a GPIB controller and countless research, industrial, and educational applications. With the 576, you can measure voltages, currents, and many other types of signals, as well as implement intelligent process control systems.

The Model 576 is a complete system. It integrates hardware and documentation to make measurement and control technology easy to use. The Model 576 combines the most often used analog and digital I/O functions into a single package.

The Model 576 has two expansion slots, one of which is normally occupied by a master analog input module. This leaves one slot for other modules you might want to add. If your application does not require analog input, you may use both slots for expansion.

The Model 576 interfaces readily to many types of controllers, and is equally easy to operate. It includes ROM-based firmware and an easy-to-use programming language. The programming command set can be used with many language environments such as BASIC, C, and Pascal. In many cases, the controller will be a computer which is equipped with an IEEE-488 (GPIB) interface

card. This includes a wide variety of configurations such as:

- IBM PC, XT, AT, and PS/2 Personal Computers equipped with an IOtech, National, CEC, Metrabyte, or other IBM-compatible GPIB interface.
- Apple Macintosh and Macintosh SE computers equipped with a suitable external GPIB interface (e.g. IOtech MacSCSI488, Mac488B).
- Apple Macintosh II computers equipped with a suitable Nubus® GPIB interface (e.g. IOtech MacII488).
- Most other computers which have an RS-232 or RS-422 compatible serial port. The Keithley 500-Serial adapter will convert such ports to GPIB input/output.

(If your computer is not listed, contact the Keithley Data Acquisition and Control Applications Support Group. Many possible methods exist for connecting the Model 576 to various computers.)

Features of the Model 576

- Microprocessor-controlled for stand-alone operation.
- 128K of mother board data RAM memory. Factory option to 512K.
- Battery-backed mother board memory for program and data retention.
- Real-time clock allows time stamping and time referenced triggering.
- Shielded all-metal case.
- One option slot for adding a signal conditioning module.

- Accepts 12-bit 62.5kHz AMM1A or 16-bit 50kHz AMM2 Master Analog Measurement Module for analog input. Add up to 32 more analog input channels via option slot.
- Trigger circuitry for synchronizing data acquisition to external analog or digital events. Provides oscilloscope-like trigger modes.
- Dual high-speed 13-bit analog output channels. Add up to five more analog output channels via option slot.
- Thirty-two digital I/O channels organized as four 8-bit ports. Programmable in groups of 8 channels for input or output. Add up to 32 more digital I/O channels via option slot.
- Sixteen power control channels use digital I/O ports 2 and 3. Programmable for sensing or switching of AC or DC loads.
- Convenient mass-termination connector for direct connection of up to 8 Analog Devices 3B signal conditioning modules or other single-ended analog inputs.
- Operates from transformer or automotive cable. Requires 12-18V AC or DC @ 40VA max.

SECTION 2

Terms, Conventions, Safety Considerations


This manual is a comprehensive introduction to the Model 576 scientific workstations, a set of integrated, computer-controlled measurement and control systems.

Terms and Conventions used in the Manual

The 576 is compatible with many types of computers using a GPIB interface which may be installed in or attached to the computer. In this manual, the controlling computer will be referenced generically as "controller", "computer", or "PC" unless the information refers to a specific type or model. In that case, the type or model will be called out.

A wide range of computers and host languages can be used to control the 576. It is thus impractical to anticipate every environment under which the 576 will be used. Therefore, it is important that you refer to your selected software, computer, and GPIB interface documentation as part of installation and operation.

The following safety terms are used in this manual or found on the instrument.

The symbol  indicates that the user should refer to the operating instructions in this manual for further details.

The **WARNING** heading used in this manual explains dangers that could result in personal injury or death. Always read the associated information very carefully before performing the indicated procedure.

The **CAUTION** heading used in this manual explains hazards that could damage the instrument. Such damage may invalidate the warranty.

The **NOTE** heading is used in this manual to identify information which may be useful or helpful.

Following these warnings will reduce the chances of personal injury or system damage.

A Few Precautions

This manual will instruct you to open the case of the Model 576 during installation and when making connections to the system.

WARNING

User supplied lethal voltage may be present on connections and option card. To avoid the possibility of electric shock, disable external power sources before making any adjustments or connections to this product. The Model 576 is not for use in electrically sensitive areas, nor for connection to humans.

CAUTION

Do not exceed the input ratings of the Model 576. Apply no more than $\pm 15\text{VDC}$ to a non-isolated analog input of the Model 576.

CAUTION

Do not apply more than +5.5 VDC or negative voltages to any digital input of the Model 576.

CAUTION

Do not install or remove a module from the Model 576 option slot while power to the Model 576 is on.

Even though it is convenient when setting up an experiment or process to leave the top open on the 576, the system should not normally be operated in this way. Dirt and small objects can easily become lodged in the 576 circuitry. This can cause unreliable performance or damage the system

CAUTION

To minimize electromagnetic interference, operate the system only with the cover closed.

A second precaution concerns connection of signals with voltages that exceed the specified ranges. Information on the voltage ranges accepted by each module can be found in the Specifications section of each module's documentation. Damage to the system caused by connecting voltages which exceed these ranges is not covered by the

product warranty. If necessary, replacement parts are available from Keithley Data Acquisition and Control. (See Servicing Information sections for parts information.)

CAUTION

"Non-isolated" as used in this manual means that internal circuit common is at power ground potential, and that external circuit common must also be at power line ground potential.

Designing Safe Control Set-Ups

Keithley Instruments manufactures its data acquisition and control products to the highest technical and safety standards. However, you, the user of this equipment, have ultimate control over how the equipment is used in the field. Always use techniques and procedures which result in safe operation of external equipment and processes.

Before beginning power up, make sure the computer, 576 and all external equipment are off.

Turn on the 576 and controller.

After the 576 and controller have stabilized, turn on the external equipment or process.

For power-down, turn off or otherwise prepare the external equipment before turning off the GPIB controller or 576.

Please consider the following points about the 576 and your particular applications when you design control configurations:

Where loss of power, interruption of the control program, or failure of any equipment can lead to unsafe conditions, do not leave equipment unattended.

Before you energize any external equipment or processes, make sure the controller and Model 576 have been switched on and have stabilized.

Revision A, B, and C of the DOM1 and PCM1 modules, which you may use in the 576 option slot, do not contain a power-on reset (the feature was added with Revision D). Output lines on pre-Rev D. modules may power up in a random state under some circumstances. Do not use pre-Rev D. versions of these modules where random output at power up is undesirable. Older modules can be factory-modified to include the power-on reset. Contact Keithley Data Acquisition and Control for more information.

The DIO1 module does not contain a power-on reset. Use the DIO1 only to sense TTL-level digital signals, or for digital applications where random output at power-up is of no consequence. The newer DIO1A does include a power-on reset.

The AOM1, AOM2, AOM3, and AOM4 analog output modules do not contain a power-on reset. Therefore, these modules may power up with random output. The AOM5 does contain power up circuitry and powers up to 0V output.

The AIM7 module does not incorporate an "open thermocouple sense". Make sure thermocouples used with the AIM7 are in good condition and operating properly.

In some cases, external equipment or processes must be returned to a particular state before control can be interrupted or power can safely be removed. Take all necessary precautions and make all necessary preparations before you interrupt a control program or switch off external equipment or processes.

CAUTION

When some personal computers are switched off or lose power, they must remain off for a short period to permit the computer supply to "bleed down" and reset. This period may be up to 30 seconds or more. If power is restored before the specified reset period has elapsed, the computer will not resume operation. Under these conditions, a 576, which contains its own power supply, will power on when power is restored, regardless of computer. Do not leave equipment unattended where a power failure may result in undesirable operation of equipment.

ALWAYS observe the following safety rules:

If you need to open the controller:

Disconnect the IEEE cable from the 576.

Turn off the controller and disconnect the power cord.

If you need to open the 576:

Turn off or disconnect any equipment connected to the inputs of the 576.

Set the 576 ON/OFF switch to OFF before opening the 576 case.

NOTE

It is not necessary to turn off the controller when you open only the 576.

WARNING

To avoid electric shock, use only a proper GPIB cable to connect the 576 to the GPIB interface. If the controller is AC-powered, note that it must be plugged into a properly grounded 3-wire outlet.

WARNING

Attach one end of the supplied safety ground wire between the ground binding post on the rear of the 576 and a safety earth ground.

In Case of Problems or Malfunctions

This manual will provide you with all the information you need to set up your system quickly and correctly. If you do have problems, however, call the Keithley Data Acquisition and Control Product Support Department at 216-248-0400.

If a 576 mainframe or module in the system is determined to be defective you will receive a return authorization (R.A.) number. Pack the module carefully in the original antistatic bag and mark the R.A. number on the outside of the box. If at any time the entire unit must be returned for servicing be sure to pack it in the original shipping materials.

Using This Manual

This manual is not a replacement for documentation which came with your controller or software. Consult the appropriate controller and software documentation for any and all information relating to the controller itself or the use of various operating systems and programming languages.

With that exception, every effort has been made to explain all important theoretical ideas, unfamiliar terms, and useful formulas. If a topic is important but lengthy you may be referred to another source for additional information.

Setup

The Setup section contains general instructions for unpacking the system, positioning jumpers and switches, and installing system cabling. These procedures generally need to be performed only once.

Hardware Configuration

The Hardware Configuration section describes how to set up the input and output functions of the Model 576. These topics include programming the various operating parameters, as well as pin-out information for each I/O function.

576 Programming

The Programming section describes how to use the Model 576 Command set which is stored in firmware within the system.

576 Applications

This section describes several typical applications and the corresponding example programs for the Model 576.

Reference

The reference section discusses the internal operation of the Model 576, as well as the programmable register associated with each of the Model 576's I/O functions.

Calibration, Maintenance, and Troubleshooting

This section describes how to isolate problems to the subsystem level, and how to calibrate a properly operating Model 576. A list of spare parts is included.

Option Modules and Interfaces

The Modules and Interfaces sections provides information on compatible option modules, their power requirements, and ID numbers. You may insert other manuals, such as those from your option modules and GPIB interface, in this space.

Appendix Section

The appendix section includes additional information about the Model 576 and GPIB bus which you may find helpful.

SECTION 3

Setup

Unpacking Your System

The Model 576, documentation, and accessories are shipped in one or more cartons. Locate all shipping cartons, and check that you have the following items.

In addition to the Model 576 manual which you are reading, you should have the following:

Model 576 High Speed Data Logger

AMM1A or AMM2 Analog Master Measurement Module with mounting hardware and trigger cable assembly. (Model 576-1 and 576-2 only).

AC Line adapter

Ground wire assembly

Any options you may have purchased

Take the 576 unit out of its packing box. Be sure to save the box and packing in case you need to return the instrument to Keithley for service.

Open the 576 by unfastening the latch on the left side of the case and swinging the lid to the right.

At the bottom of the case you will see the mother board with screw terminals for making connections to the system. The expansion slots are mounted on a PC board which is on the right side of and perpendicular to the mother board.

The 576 front panel includes power ON/OFF switch and five indicator lamps. The ON/OFF switch controls power to the 576's internal circuits. The lamps include a POWER indicator, a REMOTE lamp, an SRQ lamp, a TALK lamp, and a RUN lamp, which indicates that the 576's data acquisition circuitry is being accessed.

Mounted at the back of the mother board, on the lower rear panel, is the 24-pin connector which mates with the cable to the computer and GPIB Interface.

The rear panel also contains an external power input, power control relay rack edge connector, and a ground lug.

You will find that any module(s) you ordered are not installed in the 576 expansion slots. A detailed description of module installation is found later in this manual.

Before going any further, check the packing slip and make sure you have received the correct modules.

When you have checked the Model 576 and the modules, please fill out your purchaser registration card and mail it to:

Product Support Department
Keithley Instruments, Inc.
Data Acquisition and Control Division
28775 Aurora Road
Cleveland, Ohio 44139

Setting Up

Use of the Model 576 requires a suitable controller. This may be a dedicated GPIB controller, or a computer with the proper monitor and GPIB interface.

If you have just purchased the controller hardware, take the time to review all documentation before continuing.

In general, the 576 can be used under the same environmental conditions as the controller. It should be kept at normal room temperature, out of strong sunlight. If exposed to extreme heat or cold, the system should be allowed to return to room temperature before it is turned on. Otherwise, performance may be outside the specified tolerances, or the system may be damaged.

At this point, you should set up your GPIB controller, or install a GPIB interface in your computer. Please consult the appropriate documentation as necessary. In particular, carry out any instructions concerning the setting of switches, jumpers, or other user-configured components on this equipment. If the controller, computer, or interface includes drivers, diagnostics, or other support software, please install and run the software to be certain everything is operating correctly before you connect the 576.

Next, check the address of the 576 and reset it if necessary. The 576 address switch sits inside the case, near the IEEE-488 connector. The 576 permits the setting of a primary address only; a secondary address is not implemented. The DIP switch positions 1-5 set the address of the 576 between 0 and 31 (refer to Figure 3-1).

For a desired address, simply turn on the switches whose bit values add up to the address. The factory default address for the 576 is 3, which requires that switches 1 and 2 be set to the "ON" position. Address 13 would require

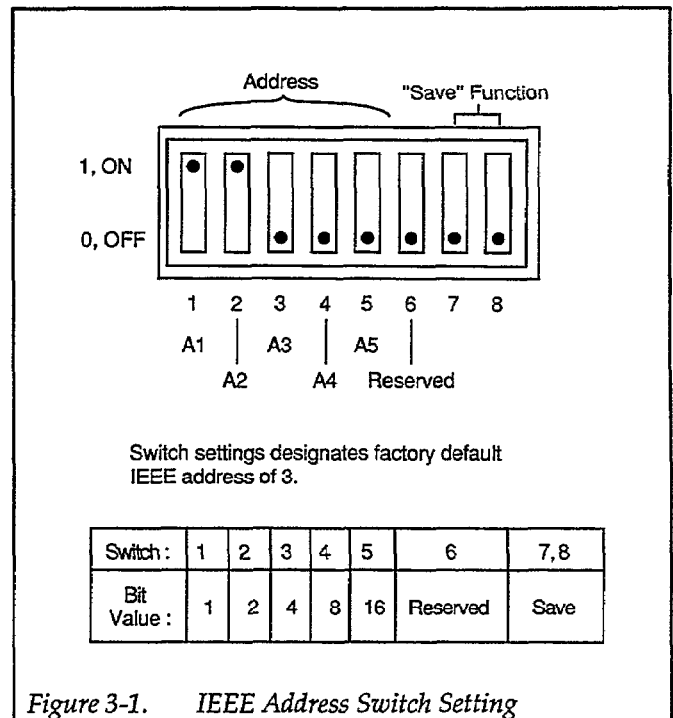


Figure 3-1. IEEE Address Switch Setting

that switches 1, 3, and 4 be turned on. Address 0 results when all switches are off, and 31 results when all switches are on. Make sure that the address you have set does not conflict with any other instruments or equipment you plan to connect to the GPIB bus.

Make the connection between the GPIB interface and 576 by attaching the cable to the GPIB connector at the rear of the 576 mother board. See Figure 3-2.

Note that the Model 576 is designed to accept an IEEE-488 connector having metric (black) attachment screws. Do not attempt to use a connector with silver screws or you will damage the threads.

Line up the connector on the cable with the connector on the rear panel of the instrument.

Tighten the screws securely, but do not overtighten them.

Make sure the other end of the cable is properly connected to the controller. Some controllers have an IEEE-488 type connector, while others do not. Consult the instruction manual for your controller for the proper connecting method. If you have followed these instruc-

tions you have now completed the connection between the 576 and the computer.

If your system is a 576-1 or 576-2, it includes an AMM module. You must install this module in the 576 before installing any option modules. The AMM modules are physically identical and are installed in the same way.

Make sure the 576 is turned off. Unlatch the fastener on the side of the 576 cabinet and open the 576.

Prepare the AMM module for installation. Confirm that trigger select jumper J3 on the AMM module is over pins 1 and 2 (Figure 3-3). Confirm that trigger select jumper W201 on the 576 mother board is over pins 1 and 2.

If you wish to connect your signals to the AMM module at this time, consult Section 4 of this manual covering the AMM connectors. Remove the cable clamp along the back edge of the AMM module. Attach your input signals to the quick-disconnect blocks on the AMM module and replace the connectors. Replace the cable clamp and attach the mounting screw nearest the card edge connector on the AMM module.

A small right-angle bracket and screw are included to fasten the rear corner of the AMM module to the rear panel of the 576. Install this bracket to the other cable clamp mounting hole on the front edge of the AMM module with the remaining cable clamp mounting screw (see Figure 3-4).

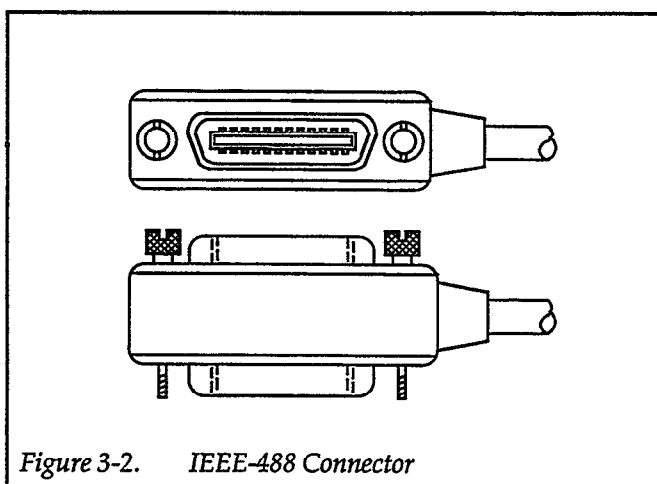


Figure 3-2. IEEE-488 Connector

Install the module in the 576 now. Hold the module with the component side facing upward. Insert the rear edge

of the module into the lower module guide located on the power supply shield, and slide the module into the lower option slot connector. Fasten the support tab to the rear panel of the 576 with the supplied screw.

Attach the trigger cable from the GLOBAL OUTPUT J7 at the top edge of the AMM module to the trigger GLOBAL INPUT terminal J201 on the 576 mother board using the supplied cable. Pay close attention to the orientation of the beveled corner on the connector at each end of the cable. (see Figure 3-3).

NOTE

If you reverse the connections of the trigger transmission cable at J7 or J201, you may introduce noise into all analog input measurements, or invert the polarity of the trigger signal.

To install an optional module, see the manual for your option module for any required configuration of gains, ranges, switches, etc. Set up the module and make connections according to your application. Hold the module with the component side facing upward. Insert the rear edge of the module into the upper module guide located on the power supply shield, and slide the module into the upper option slot connector. Attach any necessary mounting screws or strain relief to complete the hardware installation.

WARNING

Attach one end of the supplied safety ground wire between the ground binding post on the rear of the 576 and a safety earth ground.

Connect an external power source capable of supplying 12-18V AC or DC at 40 VA to the 576.

This completes installation of the Model 576 hardware.

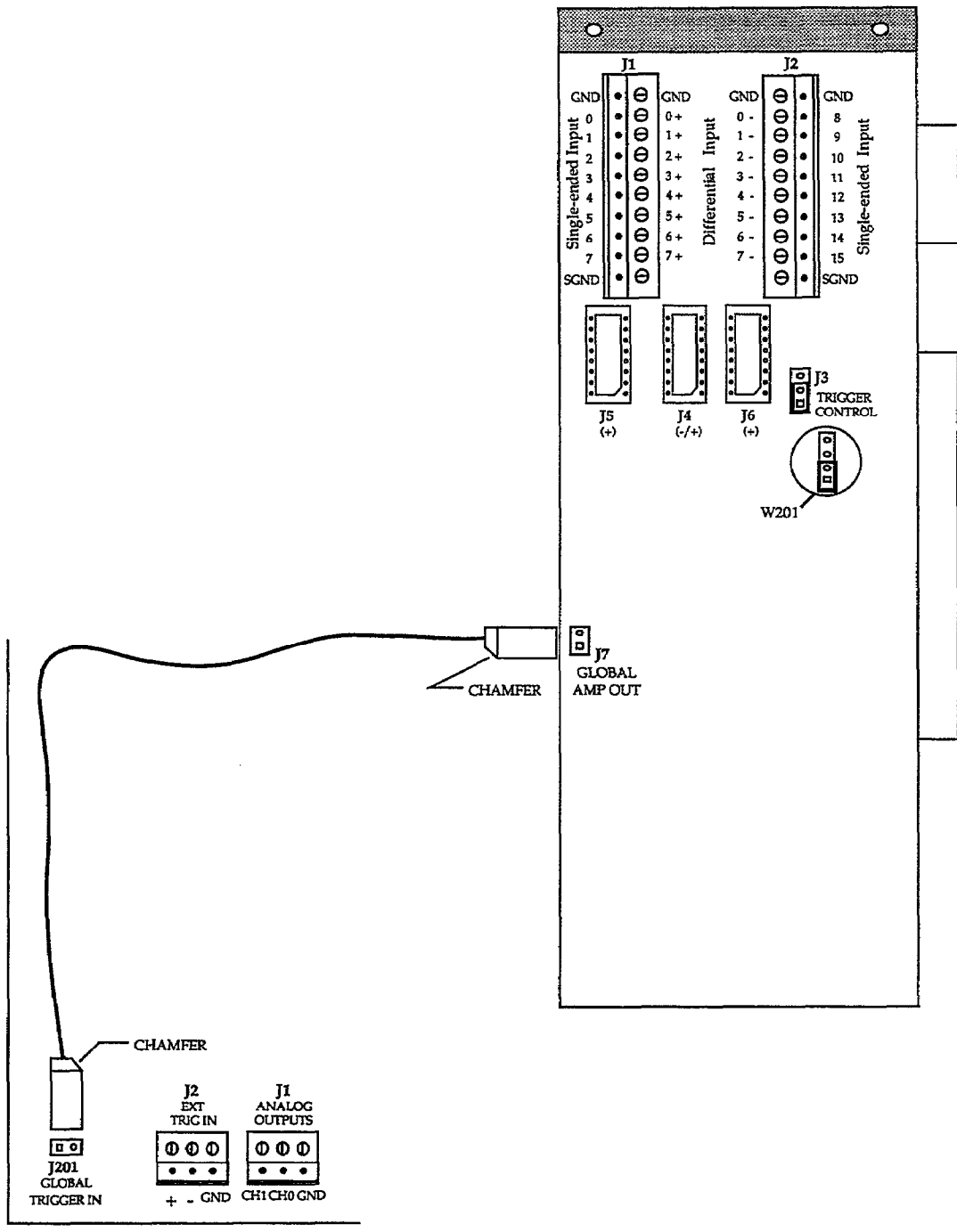


Figure 3-3. AMM Module and Analog Trigger Cable

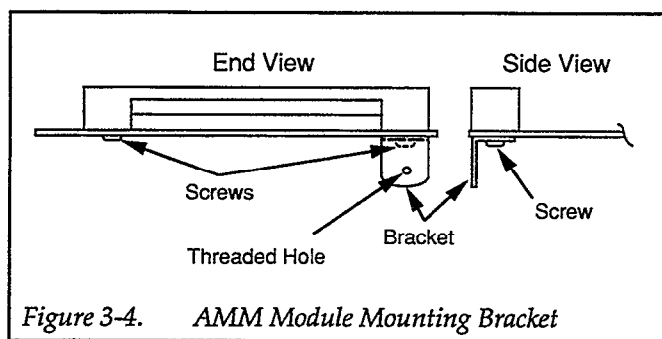


Figure 3-4. AMM Module Mounting Bracket

Completing Setup

After the system has been assembled, make sure any covers have been reinstalled on the controller and 576.

Once connected and plugged in, the controller and 576 can be turned on and off in any order. The controller can, of course, be operated normally with the interface installed, or with the interfacing cable disconnected.

When you turn on the 576, pay attention to the front-panel "SRQ" lamp. The 576 performs automatic power-on diagnostics, and will signal important error conditions via the SRQ lamp. If the SRQ lamp flashes continuously at 2 flashes/second, a problem exists with the on-

board firmware ROM. The SRQ lamp flashing at a rate of 20 flashes/second (or appearing to be on permanently) indicates a problem with on-board RAM. In either case, contact the Keithley Data Acquisition and Control Applications Support Group for further instructions.

WARNING

To avoid electric shock, use only a proper GPIB cable to connect the 576 to the GPIB interface. If the controller is AC-powered, note that it must be plugged into a properly grounded 3-wire outlet.

In Case of Problems or Malfunctions

If you have problems with setting up the 576 call the Keithley Data Acquisition and Control Product Support Department at 216-248-0400.

If a 576 mainframe or module in the system is determined to be defective you will receive a return authorization (R.A.) number. Pack the module carefully in the original antistatic bag and mark the R.A. number on the outside of the box. If at any time the entire unit must be returned for servicing be sure to pack it in the original shipping materials.

SECTION 4

Hardware Configuration

The 576 accepts signal conditioning and measurement modules from the Keithley signal conditioning module library. These are the same modules used in other Keithley 500-series data acquisition systems.

Self-ID Feature

All current production versions of Keithley modules include “self-ID” capability. Self-ID enables the 576 firmware to identify an option module’s type when the system is powered up provided an AMM module is installed in slot 1.

It is conceivable that some users may have older Keithley modules and wish to use these modules in the 576. These modules may or may not include self-ID, depending on when they were manufactured. The self-ID component is a precision resistor which is connected across pins 4 and 41 of the module’s card edge connector. If there is some question about a module’s self-ID capability, check the module or its documentation.

Most modules without self-ID are also compatible with the 576; however, a few older modules are specifically not compatible. These exceptions are as follows:

The AIM1, ADM1, ADM2 are obsolete, and are not compatible with the 576.

The AMM1 (pre-“A” revision) is not compatible with the 576. However, its replacement, the AMM1A, is fully compatible.

The 500GPIB is not compatible with the 576.

When a 576 system is turned on, its firmware will identify any modules that support the self-ID feature provided an AMM1A or AMM2 module is mounted in slot 1. If a module is an older unit which does not include the self-ID capability, the 576 will simply not detect the module. No error message will be issued. If no AMM module is installed, the 576 must be configured using the program language.

If a compatible module does not have self-ID capability, it still can be used in the 576 provided it is identified manually from within the programming environment. The SYST :SLOT command assigns a module to a specified slot. For example, an older AIM7 module lacking self-ID could be assigned to slot 3 with the command:

```
SYST :SLOT 3, AIM7;      ' Assign an AIM7 to slot 3
```

If you attempt to assign a module to a slot for which the firmware has already identified a module, an error will be issued.

Changing Default Configuration Parameters

When you manually assign a module to a slot, certain default information will be assumed for the module. This information includes items such as gain switch settings and single-ended or differential input configuration, and will vary according to the module type. You are free to reprogram these defaults to any legal values for a given module. You may also check the defaults for any module by checking the parameters in question with the CHAN command.

Hardware Switch Configuration

Some modules, notably the DIO1 and PIM2, have performance features which are set by physical switches on the module. There is no direct method for the firmware to detect the positions of these switches, so the information must be passed to the system within the test program at run time. The "CHAN" command enables you to enter this additional information about a module which is not covered by other commands. Review the "CHAN" command in the programming section of this manual.

Setting Up Measurement and Control

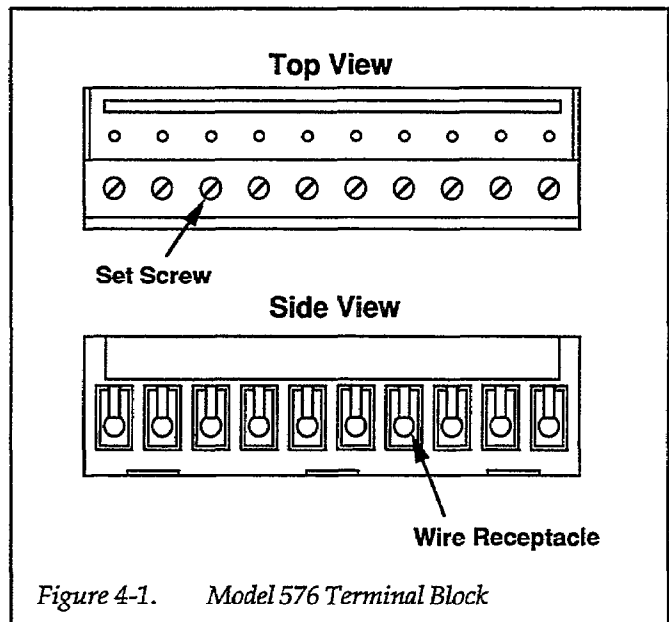
The Model 576 permits you to conveniently make measurements and generate control signals for a wide variety of conditions. This section of the 576 Manual presents information you will need for setting up the various acquisition and control functions of the Model 576.

Signal Connections

The Model 576 uses quick-disconnect terminal blocks for convenient connection and disconnection of signal leads. Where an AMM module is plugged into slot 1, a ribbon cable mass termination can also be used for analog input connections. This hardware is available with newer AMM1A and AMM2 modules.

Open your Model 576 and you will note a series of brown screw-terminal blocks mounted on the rear of the 576 mother board. These terminals are for digital I/O. You will also note a pair of smaller brown terminal blocks located at the front of the motherboard near the front panel. These terminals are for analog output and trigger input. Finally, the AMM1A or AMM2 module, as well as most other modules you may mount in the remaining option

slot, will have these brown terminal blocks. See Figure 4-1.



A quick-disconnect terminal block can be removed from the motherboard by pulling it straight off the motherboard with a firm, even pressure. Do not pry the terminals with a screwdriver or sharp object, or you may damage the motherboard.

CAUTION

Some older Keithley signal conditioning modules do not have removable terminal blocks. These blocks are blue, rather than brown. You may find these types of terminals on modules which you are installing in the 576 option slot(s). Do not attempt to remove the blue terminal blocks or you will damage the board.

To make connections to a quick-disconnect terminal block, first strip 3/16 of insulation from the end of the wire which you want to attach. Loosen the desired terminal screw on the block and slide the bare end of the wire into the hole beneath the metal tab visible in the hole. Tighten the screw to compress the tab against the wire.

After you have attached all the desired signal wires to a terminal block, replace the terminal block on the motherboard or module from which it was removed.

There are also four holes each in the 576 front and rear panels that will accommodate standard BNC connectors. To install a BNC connector in the front panel, use a pointed knife to cut the panel overlay (the rear panel holes are open). Solder leads to the BNC connector and mount the connector in the open hole. Connect the leads to the desired terminal block according to the steps listed above.

“Channel” and “Slot”

This manual uses the terms “slot” and “channel”. Channel refers to an independent path over which signals travel between the Model 576 and the outside world. Individually-numbered screw terminals on each Model 576 terminal block provide connection to its various input and output channels. One to three screws make up a channel connection, depending on the type of measurement or control signal.

The meaning of “slot” is not as obvious as “channel”. The Model 576’s companion product, the 500-series, will accept up to 10 optional I/O modules which plug into

physical slots on the Model 500 mother board. Thus, one describes modules as being mounted in certain “slots”.

The Model 576 has two physical slots in which modules may be mounted. One slot is normally occupied by an AMM1A or AMM2 module, leaving the other slot for an optional signal conditioning module. However, the term “slot” is still associated with other I/O functions on the Model 576 motherboard. The firmware views the Model 576 as a series of slots into which certain modules have been mounted. Slots 2, 4, 5, and 6 are “virtual” slots which consist of circuitry built into the 576 motherboard. Table 4-1 shows these I/O functions and their corresponding slot numbers.

The following information describes connections and jumpering for the various functions of the 576. You should also consult the manual for the AMM1A or AMM2 if one is mounted in slot 1. Similarly, consult the documentation for any optional module you have mounted in slot 3. Throughout the following discussion, refer to Figure 4-2, Figure 4-3, and Figure 4-4 which are component and terminal I/O drawings of the Model 576 motherboard and sideboard.

Table 4-1. Model 576 Functions and Slot Assignments

Slot	Type	Function
1	Physical slot	Used to mount an AMM1A or AMM2 (described hereout as “AMM”) analog input module, or an option module where analog I/O is not required.
2	Virtual slot	Trigger circuitry performs all functions of the TRG1 module.
3	Physical slot	Used to mount optional module.
4	Virtual slot	Analog output circuitry performs functions of an AOM5/2 analog output module.
5	Virtual slot	Digital I/O circuitry performs all functions of DIO1 TTL digital I/O module.
6	Virtual slot*	Eight single-ended analog inputs, with connector compatible with 3B signal conditioning subsystem.

*NOTE: These analog inputs are shown to exist in slot 6 according to the 576 internal configuration. This is for convenience only. The external function actually uses the AMM global analog inputs 3-10 which feed the A/D converter of the AMM module in slot 1. If an analog input module is used in slot 3, only 7 external inputs will be available.

SECTION 4
Hardware Configuration

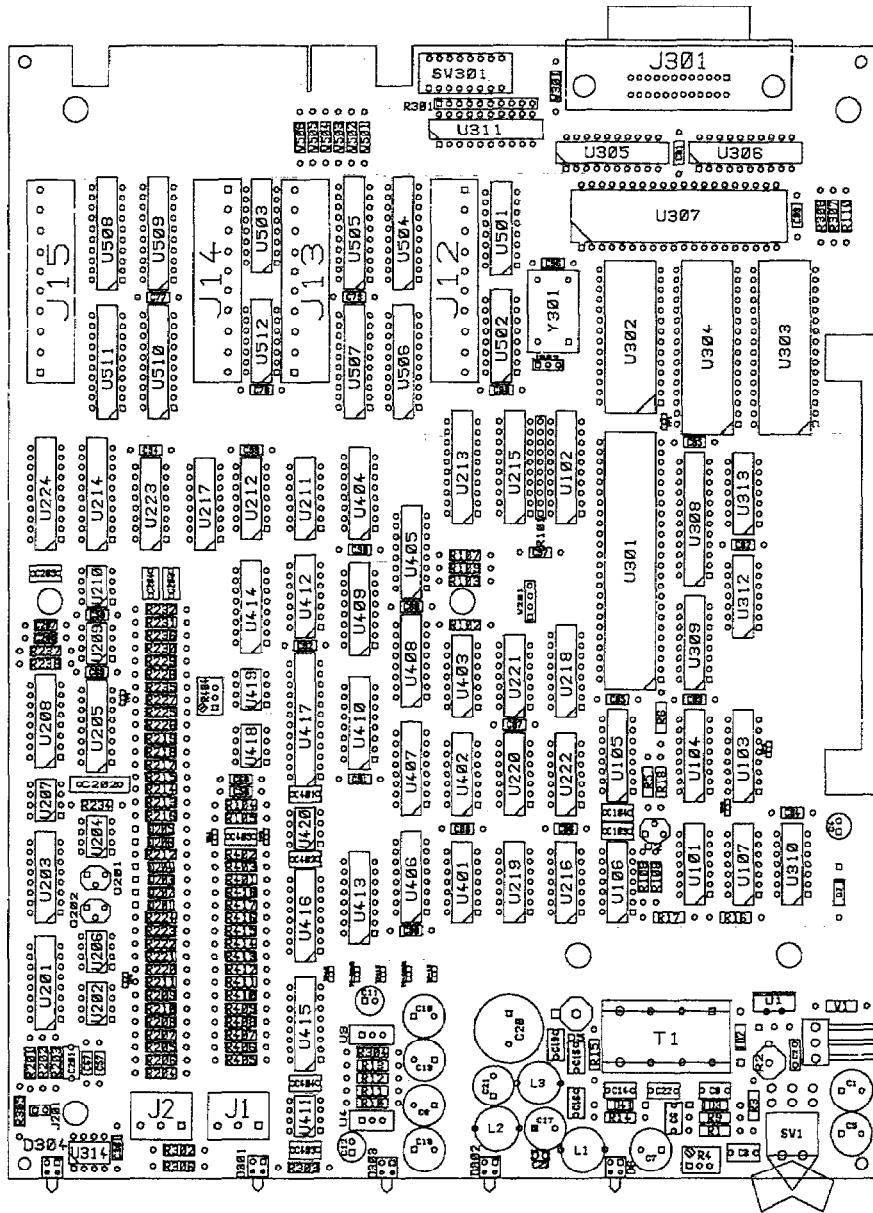


Figure 4-2. Model 576 Mother Board

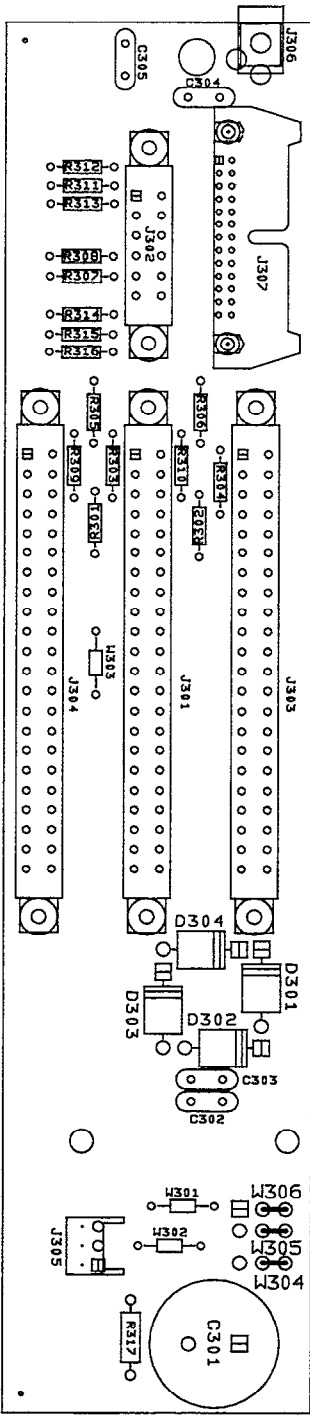


Figure 4-3. Model 576 Side Board

SECTION 4
Hardware Configuration

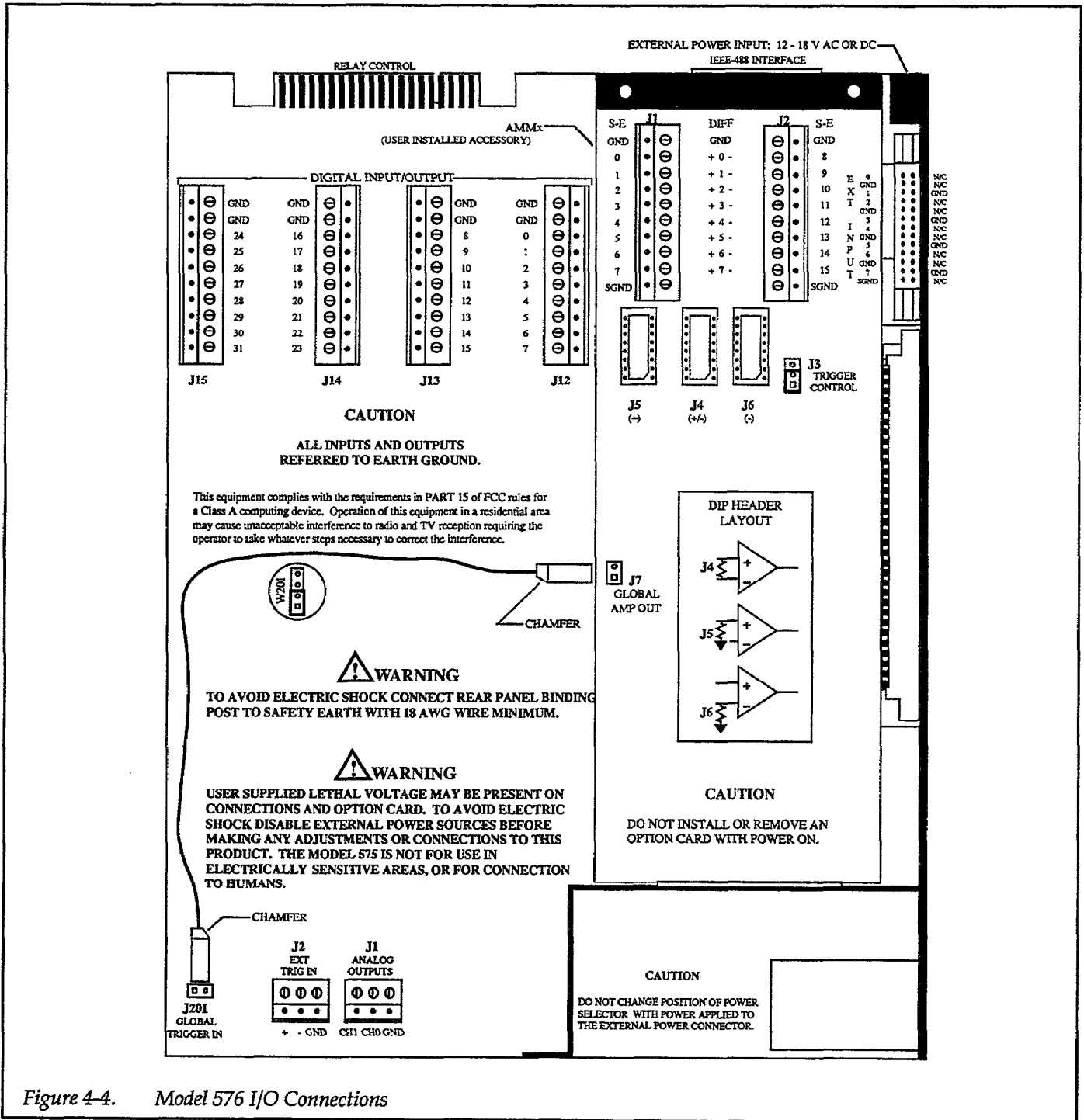


Figure 4-4. Model 576 I/O Connections

Analog Input – Slot 1

The lower option slot is normally occupied by AMM1A in the Model 576-1, or AMM2 module in the Model 576-2. The AMM1A offers 62.5kHz/12-bit A/D resolution, while the AMM2 offers a 50kHz/16-bit A/D. Both AMM modules are physically identical. If your system is a Model 576-1 or 576-2, it includes an AMM module and corresponding manual. Consult the AMM manual for more information on the AMM1A or AMM2 module.

The AMM modules contain quick-disconnect terminals for connection of up to 8 differential channels or 16 single-ended channels. If the AMM module is used in single-ended mode, connect the input signal to the desired signal terminal screw (0-15). Connect signal low to an AMMSGND terminal. If the AMM module is used in differential mode, connect the signal high input to the desired signal terminal screw (0+ to 7+). Connect the signal low to the corresponding low input terminal (0- to 7-). Attach the cable ground to the AMM GND terminal screw. See Figure 4-4.

NOTE

For best analog input performance, be sure to use the proper ground screw on the AMM module for single-ended or differential measurements. Single-ended measurements use the SGND screw for signal low. Differential measurements use the GND screw for the cable shield connection. Do not use the SGND terminal as part of differential measurements.

NOTE

You may have to detach the AMM mounting bracket from the rear panel of the 576 to make or change analog connections. Be sure to reassemble the bracket when you are done.

NOTE

The AMM module input mode is controlled through software. Be sure to indicate whether single-ended or differential input is desired. The default operating modes for the AMM1A and AMM2 are 16 single-ended inputs and $\pm 10V$ A/D converter range.

The AMM modules are fully software programmable; there are no hardware switches. Programmable features

include local amplifier gains of X1 and X10, global amplifier gains of x1, x2, x5, and x10, programmable filter (2kHz or 100kHz), and 0-10V unipolar or $\pm 10V$ bipolar A/D ranges. The AMM modules also contain DIP sockets for current shunt and pull-down resistors which are sometimes useful in making analog measurements. The A/D converter on the AMM modules also performs A/D conversion for any other analog input module which may be plugged into the system. The AMM modules can also perform an A/D gain and offset calibration under software control.

If analog input is not required, slot 1 may be used for another module.

NOTE

The precision reference on the AMM module is used by the 576 analog output circuitry. If an AMM module is not mounted in the 576, the analog output circuitry will not achieve its rated accuracy.

Trigger – Slot 2

The trigger function is built into the 576 motherboard and provides one trigger input channel. The 576 firmware does not directly support a TRG1 in option slot 3. PEEK/POKE may be used.

The trigger function is a true hardware trigger by which data acquisition and control can be synchronized to external digital or analog signals. The trigger input is located on terminal block J2 at the forward left corner of the 576 mother board. Jumper W201, which also controls trigger operation, is located on the 576 motherboard.

The trigger function only operates in the AMM module's high-speed "auto-acquire" mode which is driven by the crystal oscillator on the AMM module.

The 576 motherboard also includes a global input to the trigger circuitry, which is accessed via the "GLOBAL IN" connector J201. J201 is located in the forward left corner of the 576 motherboard. This connector permits a signal input to the AMM module to also be used as the trigger input without requiring that the signal be connected to J2.

NOTE

The GLOBAL OUT terminal on the AMM1A and AMM2 modules is located near the top edge of the module. If you want to use the global input feature of the trigger circuitry, you must connect the GLOBAL OUT terminal pins of the AMM module to the GLOBAL IN terminal pins of the 576. A cable for this purpose is provided. Refer to Figure 4-4.

NOTE

Make sure the trigger cable is installed correctly or you may introduce noise into the measurements. The trigger cable extends from the GLOBAL OUTPUT J7 at the top edge of the AMM module to the trigger GLOBAL INPUT terminal J201 on the 576 motherboard. Pay close attention to the orientation of the beveled corner on the connector at each end of the cable. The bevel must be over pin 1 on J7 and over pin 2 on J201.

Typical trigger modes include the following (see Figure 4-5):

1. Trigger at a user-defined threshold on the falling edge of the trigger signal. Stop when the prescribed number of points have been acquired.
2. Trigger at a user-defined threshold on the rising edge of the trigger signal, and acquire data only when the signal is above the threshold. Stop when the prescribed number of points have been acquired.
3. Trigger at a user-defined threshold on the falling edge of the trigger signal, and acquire data only while the signal is below the threshold level.
4. Trigger at a user-defined threshold on the rising edge of the trigger signal and take only one reading. (Normally used to trigger a reading of another channel.)
5. Trigger at a user-defined threshold on the falling edge of the trigger signal and acquire one reading. Repeat each time the trigger condition is met until the prescribed number of points have been acquired.

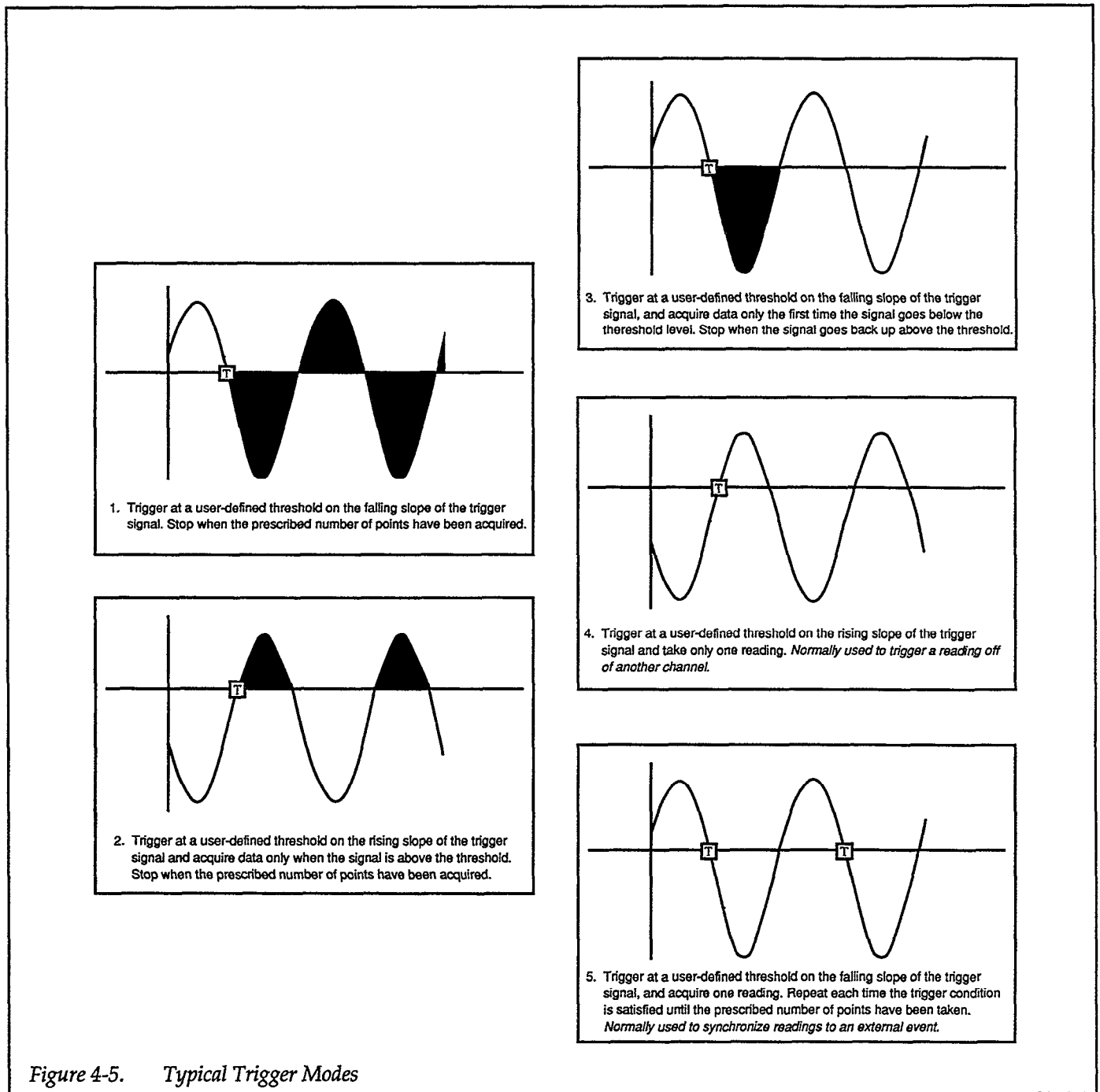


Figure 4-5. Typical Trigger Modes

Setting Jumpers for Trigger Modes

Jumpers on the 576 motherboard and the AMM module select and configure the trigger function.

Single Trigger Input — to trigger an AMM1A or AMM2 analog input module off a single trigger signal fed to the 576, set jumpers and connections as follows:

576 trigger jumpers — Locate W201. Jumper pin 1 to pin 2. Connect the trigger signal to J2.

AMM module — place jumper J3 over pins 1 and 2. Connect the signal of interest to one of the AMM input channels.

NOTE

To gain access to the jumpers on the AMM module and 576 motherboard, you must remove the AMM module from slot 1. If you have a module in slot 3, you must also remove it to gain access to the AMM module and 576 motherboard.

Option Slot — Slot 3

The Model 576 has two option slots. Slot 1 is normally used for an AMM1A or AMM2 analog measurement module. The second option slot can be used to operate a module to provide a capability which is not built into the Model 576. Examples are modules for isolated input, current input or output, thermocouples, strain gauges, frequencies, and LVDTs.

Consult the Keithley catalog for more information on specific available module functions and capabilities. If you are in doubt as to the compatibility of a certain module with the Model 576, contact Keithley Data Acquisition and Control product support in the U.S. at (216)248-0400.

CAUTION

Before you install or remove a module from the Model 576 option slot, turn off the 576 front panel power switch or you will damage the module.

To install an optional module, see the manual for your option module for any required configuration of gains, ranges, switches, etc. Set up the module and make con-

nections according to your application. Hold the module with the component side facing upward. Insert the rear edge of the module into the upper module guide located on the power supply shield, and slide the module into the upper option slot connector. Attach any necessary brackets, mounting screws, or strain relief to complete the hardware installation. If you are using both slots for option modules, install the slot 1 (lower) option module first. The steps are the same as outlined for slot 3.

Analog Output — Slot 4

The analog output function is built into the 576 motherboard, and consists of two high-speed 13-bit output channels which behave as a 2 channel AOM5 module. Terminal screws are located on J1 at the forward left corner of the 576 mother board. The 576 analog output circuitry has a 5 μ S settling time, and can theoretically achieve speeds upwards of 200kHz. However, the speed of the 576 microprocessor limits the analog output speed.

There are restrictions as to the output capabilities of each channel. Generally, there is an upper limit on the amount of capacitance and a lower limit to the resistance that can be connected across the output. To avoid possible oscillation, output capacitance must be less than 200pF.

If it is necessary to drive a capacitive load larger than 200pF, a 100 Ω or larger resistor must be placed in series with the output. This will slow down the settling time somewhat, depending on the value of the capacitive load. If an analog output channel must drive a load with both low resistance and high capacitance, the output must be buffered by an external voltage amplifier.

Similar restrictions apply to the output current, which is determined largely by the resistive component of the load connected across the output. If the resistance is too small, accuracy will suffer. To maintain rated accuracy, the load resistance should be no smaller than 2k Ω with a maximum output of ± 10 V. Maximum current output is 5mA.

The analog output circuitry also offers an "auto-sequence" mode which can be implemented through PEEK and POKE commands. This feature makes it possible to write optimized high-speed analog output routines. It is described later in this manual.

NOTE

The 576 analog output function uses 10.00V precision reference on the AMM module. If an

AMM1A or AMM2 module is not mounted in slot 1, the 576's analog output feature cannot achieve the rated accuracy.

Digital I/O — Slot 5

The digital input and output functions are built into the 576 motherboard. Digital I/O consists of 32 non-isolated, TTL-compatible channels which are configured in groups of eight channels, called ports, for input or output (see Figure 4-4).

TTL standards define an input "0" or "low" as being less than 0.8V, and an input "1" or "high" as being greater than 2.0V. A typical output high level is 3.75-4V.

The ports are configured through software control. When the Model 576 is initially switched on, all digital ports on the 576 motherboard initialize in a high-impedance, tri-state condition. Four terminal blocks (J12 – J15) are located at the rear of the 576 motherboard. Each terminal block provide connections to a port plus two ground screws for ports 0, 1, 2, and 3, respectively. The ports correspond to the connectors as shown in Table 4-2.

Table 4-2. Digital Channels and Ports

Connector	Port Designation
J12	0
J13	1
J14	2
J15	3

CAUTION

Do not short a digital output terminal to ground or you may damage the digital circuitry of the Model 576.

WARNING

The digital ports 2 and 3 are used for power control as well. Do not leave a PCM3 relay board connected to the 576 motherboard unless you are doing power control.

Power Control — Slot 5

Ports 2 and 3 of the digital I/O circuitry can also be used for power control. These ports are internally connected to the card-edge J16 located at the rear of the 576 motherboard (see Figure 4-4). Thus, they are shared by the power control connector and 2/3 port digital terminals. When the ports are used for power control, they should not be used for digital I/O, and vice versa.

Power control requires the optional PCM3 relay card and ribbon cable assembly, plus whatever relays are needed for the application. The power control function includes switching of external loads as well as sensing when current is flowing through an external circuit. The ports 2 and 3 can both be configured for sensing, control, or one port can be dedicated to each function. You must select relays according to the voltage range, function, and AC or DC voltage. See Table 4-3 for available types.

Table 4-3. Power Control Relays for the PCM3

Keithley Part No.	Description
500-OAC1	120V AC Control Relay
500-OAC2	240V AC Control Relay
500-IAC1	120V AC Sensing Relay
500-IAC2	240V AC Sensing Relay
500-ODC1	60V AC Control Relay
500-ODC2	200V DC Control Relay
500-IDC1	32V DC Sensing Relay

WARNING

The digital ports 2 and 3 are used for digital I/O as well. Do not leave a PCM3 relay board connected to the 576 motherboard unless you are doing power control.

When programming the power control (output) function, note that the logic is "low-true". The output of the 576 channel corresponding to a selected relay must be set to logic "0", or "low" in order to turn on the relay. A channel will be read as "low" when a sensing relay detects current flow to an external load.

External Input — Slot 6

The external input function offers eight single-ended analog input channels for direct connection to an optional Analog Devices 3B signal conditioning rack, or any single-ended input. Connection is made through mass-termination connector J307 which is located near the lower option slot on the 576 sideboard assembly. The connector is a 26-pin ribbon-cable type. You will normally make connection to this termination using a cable you receive with the Analog Devices 3B subsystem. The pinout data for this connector is shown on Figure 4-4.

Input signals from J307 are routed directly to the global multiplexer of the AMM module in slot 1 along analog pathways 3-10 as shown in Table 4-4. This frees all the input channels on AMM module for other signals. As such, the external input is really an extension of slot 1, not slot 6. Slot 6 is used in the 576's configuration for user convenience only.

Table 4-4. External Input Channels — Slot 6

External Input Channel	AMM Global Multiplexer Input
0	10
1	9
2	8
3	7
4	6
5	5
6	4
7	3

The external signals are selected by the AMM global multiplexer and sent to the AMM A/D converter for digitizing. Programmable global gains of x1, x2, x5, and x10 can be applied to external analog input signals. Local gain is fixed at x1.

NOTE

If an analog input module is used in slot 3 of the 576, the external channel 7 will not be available for use.

NOTE

The EXTERNAL function requires an AMM module in Slot 1. The input signal may be analog or digital.

Advanced Topics

The following information covers topics which will enable you to optimize the performance of your Model 576.

Gain

Most analog input modules compatible with the 576 offer some type of gain which can be applied to the input signal. The gain will be set according to software parameters, or with older modules, according to hardware switches on the module.

The value of applying gain conditioning is that it can increase a relatively low input voltage before the voltage is digitized by the A/D converter. This provides a greater number of A/D counts which improves the effective resolution of the reading.

For instance, with a 16-bit A/D converter input range of $\pm 10V$, an input of 0V produces 32768 counts from the A/D converter, while 0.1V produces 33095 counts. With a gain of X100 applied, the 0.1V signal will be amplified to 10V at the input of the A/D converter. This corresponds to 65535 counts from the A/D converter. Thus, the A/D converter now breaks a difference of 0.1V into 32767 steps, not 327. This substantially improves the resolution of the measurement.

Two types of gain are available. First, the AMM1A and AMM2 modules and several other analog input modules include a front-end amplifier called the "local gain amplifier" or "instrument amplifier". This stage applies gain to the signals input directly to the screw terminals of that module. For the AMM1A and AMM2 modules, local gain is selected through software control, and consists of x1 and x10 gain ranges. Other modules may have available gains of x1, x10, x100, or x1000, set through software control or hardware switches depending on the module. Some "high-level input" modules, such as the AIM2, have a gain of only x1.

The second type of gain available in the Model 576 is applied by an amplifier stage immediately before the AMM A/D converter. The amplifier stage is called a “global gain amplifier” because its gain can be applied “globally” to any analog input signal as it is routed to the system A/D. This circuitry and the A/D converter are both located on the analog master measurement module. In the case of the Model 576, this is an AMM1A or AMM2 module. The global gain amplifier normally has ranges of $\times 1$, $\times 2$, $\times 5$, and $\times 10$, which are selected on a channel-by-channel basis through software.

Local gain and global gain can both be applied to a signal simultaneously, and the overall gain is multiplicative. Therefore, gains as high as $\times 10,000$ are available (AIM8 module local gain at $\times 1000$, followed by A/D global gain of $\times 10$).

NOTE

To optimize low-level measurements, use shielded cable and other low-noise measurement techniques, such as differential input. As much as possible, avoid noisy environments. These are good practices for any measurement set-up, but especially important when measuring low-level signals.

A/D Converter Range

A second general method of optimizing analog input signals is to change the A/D converter input range. The AMM1A and AMM2 modules used in the Model 576 provide two A/D ranges: $\pm 10V$ and 0 to $+10V$. This range is selected through software.

Changing the A/D converter range allows you to more precisely match the converter to the voltage produced by the instrument amplifier. You can make measurements with different A/D ranges, global amplifier gains, and local amplifier gains in your test programs. This matches the system to a wide range of voltage inputs.

Single-Ended vs Differential Input

Analog inputs applied to the Model 576 will fall into two general categories: single-ended inputs and differential inputs. These input schemes can usually be identified by the numbers of wires which must be connected for one signal.

Single-ended inputs normally have two wires, or a wire and shield. For two wires, one wire is generally considered the signal lead, while the other is the ground lead. For a shielded cable, the center wire is usually signal, while the shield is ground.

WARNING

To prevent a shock hazard, the ground lead of equipment powered from 115V AC must be connected to ground potential. The voltage difference between the equipment ground and the Model 576 ground should never exceed a few millivolts.

CAUTION

Maximum input voltage for Model 576 inputs is $\pm 15V$. Note that if any input exceeds $\pm 10V$, all inputs will be inoperative.

Differential signals may have two or three wires. For two-wire circuits, one wire is signal positive and the other is signal negative. Unlike single-ended configurations, this negative lead is not necessarily at ground potential. Three-wire differential inputs normally consist of a positive lead, a negative lead, and a ground shield.

Differential outputs are commonly found on laboratory instrumentation and other precision instrumentation. The use of two leads plus ground offers maximum immunity from common mode noise and ground loops. The shield should be connected to the GROUND terminal screw on the analog input module.

Whether you use a differential or single-ended input depends on several factors. You can measure many types of differential signals in single-ended mode. Three-wire differential signals must be measured in differential mode. Low-level signals should be measured in differential mode to minimize noise. Differential input should also be used where maximum precision is required. Single-ended mode affords the greatest number of channels, and can be used where you need the maximum number of inputs for measuring higher-level voltages.

Measuring Currents

The Model 576 cannot measure current directly, but it can measure the voltage drop produced across a resistor when current flows through the resistor. The AMM1A and AMM2 modules have three 16-pin DIP sockets (J4, J5,

and J6), each of which accepts a "header" plug which can hold up to eight analog input resistors. Each resistor must be connected from a pin on one side of the header to the corresponding pin on the other side of the header. See the AMM1A or AMM2 manual for details.

For differential current measurements, the header must be plugged into J4 on the AMM module. This places the resistor across the (+) and (-) inputs of the differential channel.

For single-ended current measurements, the header must be plugged into J5 on the AMM module. This places the resistor across the (+) and ground inputs of the single-ended channel.

Choose a value for the resistor which produces a voltage drop of one to a few volts. Calculate the resistor value you need with Ohm's law $E = I \times R$,

$$E \text{ (desired voltage drop)} = I \text{ (in amperes)} \times R \text{ (in ohms)}$$

Generally, you may choose any resistor value which gives a suitable voltage drop. To avoid heating which may affect measurement accuracy, make sure the maximum power dissipated in the resistor is well within the power rating of the resistor. Calculate this dissipation with another Ohm's law equation,

$$P = I^2 \times R$$

Measuring Floating Sources

You may need to install analog input resistors when you measure a floating signal in differential mode. A floating signal typically has a "+" terminal and a "-" terminal neither of which is at ground potential. The signal has no low-impedance return path to ground. Under these circumstances, the input of the AMM module or other analog input module may exhibit a capacitive effect and build a charge from the input signal. Over several minutes, the common mode voltage may rise until the analog input no longer functions properly. This causes no damage to the module, but does make it impossible to measure signals.

The solution is to connect a resistor of 10K ohms or less from the negative analog input terminal to ground for

each signal input. These resistors can be soldered to headers plugged into input resistor sockets on the module. The header should be plugged into J6 on the AMM module.

Input Filtering

When noise is a problem, filtration may be applied to analog input signals. The AMM1A and AMM2 modules in the Model 576 both include a programmable filter between the global gain amplifier and A/D converter stages. This filter has cut-off frequencies of 100kHz and 2kHz.

Where further filtering is desired, a single-pole input filter may be placed on the incoming signal line for any analog input (see Figure 4-6). The relative RC values will depend on a variety of factors, including the frequency of the noise, the required attenuation, and the necessary response time. The RC values can be computed from the formula:

$$f(3dB) = \frac{1}{2\pi RC}$$

Where f is in Hz, C is farads, and R is ohms. The resulting system response time within .01% is then equal to $9.2 \times RC$.

As an example, assume that 10 counts of 60Hz noise is present in the signal. To reduce the noise to one count, an attenuator factor of 10(20dB) will be necessary at 60Hz. A single-pole filter will roll off at a rate of 20dB per decade. Thus, a 3dB point of 6Hz would be chosen to attenuate to 60Hz noise by 20dB. Rearranging the above equation to solve for R we have:

$$R = \frac{1}{2\pi \times C \times f(3db)}$$

Picking a nominal value of $0.5\mu F$ for C , the necessary resistance is:

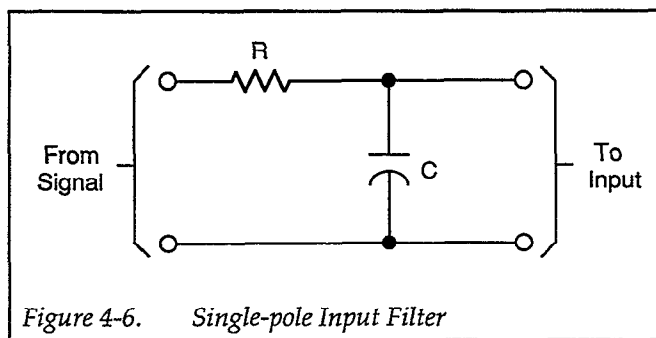
$$R = \frac{1}{2\pi \times (0.5 \times 10^{-6}) \times 6}$$

$$R = 53,000 \text{ ohms}$$

The resulting response time (T_r) is:

$$\begin{aligned} T_r &= 9.2 \times RC \\ T_r &= 9.2 \times 53000 \times 0.5 \times 10^{-6} \\ T_r &= 0.24\text{sec} \end{aligned}$$

Note that there are a number of RC values that can be used in a given application. To minimize the effects of the series resistance, however, it is recommended that the value of R be kept as low as possible.



Analog Output Range

The analog output range is set through software control.

You may select from several ranges for the Model 576 analog output function. The default D/A range is $\pm 10V$. Available ranges of analog output are $\pm 10V$, $\pm 5V$, $\pm 2V$, and $\pm 1V$.

The analog output converter circuitry data format is 12 bits plus a sign bit. For unipolar ranges, the D/A output will be divided over 4096 steps. For bipolar operation, the sign bit causes the polarity of the converter to be positive or negative. The result is that the full A/D range of -10 to $+10V$ is effectively divided into 8192 steps. By selecting a narrower output range, the output resolution can be improved further at the expense of the available voltage span. Programming $-0V$ or $+0V$ results in the same output. Note that if you are writing raw D/A counts to the analog output channels, the output value will be written as two 8-bit words. The sign bit is the most significant bit. Format:

S XXX HHHH LLLLLLLL

where

- S = Sign
- H = High-order bits
- L = Low-order bits
- X = Don't care.

SECTION 5

576 Commands

Introduction and Overview

The 576 is programmed over the GPIB bus, which is the standard implementation of IEEE standard 488-1978. The GPIB interface is the communication link between the 576(s), other GPIB devices, and controller. The interface and bus supply all the necessary programming, data and control information to operate the 576. The 576 programming language includes commands for data and memory management, creating subroutines, rate control, triggering, timestamping, system configuration, and engineering units conversion.

The 576 fully supports all the series 500 I/O modules except the AIM1, ADM1, ADM2, AMM1, MEM1, PCM2, STEP1/STEP2, TRG1 and GPIB modules. A revised version of the AMM1, the AMM1A, is fully compatible with the 576. The MEM1, STEP1, STEP2, TRG1, PROTO, and WAV1 modules may be used in the 576 if they are accessed through the PEEK and POKE commands.

Configuration

The SYST command is used to configure or interrogate the system functions of the 576. The command contains a function parameter which allows the user the following capabilities:

1. select the filter of the system A/D (:AMM),
2. get memory and buffer information (:MEM?, :BUF?),
3. set default engineering units conversions and data format (:UNIT, :FORMAT),
4. set up GPIB parameters (:EOI, :SRQ, :TERM),
5. set and read the real time clock (:CLOCK),
6. return system error message (:ERR?),
7. set system power up/down options (:SAVE),
8. configure the slots with modules (:SLOT),
9. set up timestamping (:STAMP),
10. and return the system ID (:IDN?).
11. Apply calibration factor to AMM module (:CAL)
12. Read and write data directly to hardware (PEEK and POKE)

All the system commands are executed immediately upon receipt and those ending with "?" will immediately buffer the requested data for a reading by the controller.

The CHAN command is used to configure or interrogate the input / output functions of the 576. The CHAN command also contains a function parameter which allows the configuration of:

1. the filter of the specified channel or range of channels. (:FILTER),
2. the gain applied of the specified channel or range of channels. (:GAIN),
3. the offset enable or disable (:OFFSET),
4. the mode of the specified channel or range of channels. (:MODE), and
5. the range of the specified channel or range of channels. (:RANGE).

The use of these functions is dependent on the module being configured. Consult your hardware manual for the detail description of the hardware. These commands are also immediately executed upon receipt and those ending with "?" will immediately buffer the requested data for a read by the controller.

The 576 supports calibration of the AMM modules through use of the SYST:CAL command. This calibration is a gain and offset correction for the global input amplifier. To allow the system to perform correction it is required that the calibration constant supplied with the AMM module be sent to the 576. This constant is stored in battery backed up RAM and need only be downloaded to the 576 the first time the system is used. If calibration is disabled and later re-enabled, if the AMM module is recalibrated, or if the battery support of the RAM fails. Any time after the constant is sent to the 576 the command "SYST:CAL;" , without the constant, can be issued to generate all the calibration coefficients used by the 576. This command should be issued only after the 576 has been powered up long enough to meet the warm-up time specification.

Data and Memory Management

The 576 allows up to twenty buffers, named B0 through B19, to be dynamically allocated from the 100,000 bytes of system data memory. The system data memory size can be increased to over 480,000 bytes through the use of the memory expansion option.

Most data in the 576 is managed through buffered operations using the BUFF commands. The BUFF READ command is used to convert and format data in the specified buffer to the specified engineering units and data format. The data is then ready for a read by the controller. The BUFF WRITE command is used to send data, in the specified engineering units, from the controller to the specified buffer. The READ command collects data from an input device and stores it to a specified buffer. The WRITE command sends data in a specified buffer to an output device.

All data stored in 576 buffers is stored as raw binary data. Data being sent to the 576 from a controller is converted as it is sent, based on specified engineering units conversion and system data format, to raw binary and then stored in the buffer. Data stored in the 576 buffers is converted from raw binary to the specified engineering units and format as it is sent to the controller.

Data buffers are created in the 576 using the BUFF DIM (dimension buffer) command. Buffers are of a specified type (BYTE, WORD, LONG, TC) and are dimensioned as the number of channels in a scan by the number of scans. The TC specifier handles the special case of reading thermocouples and automatically allocates an additional channel (WORD) per scan to store the cold junction reference. The dimension statement also enables timestamping through use of the STAMP specifier. When this feature is specified, six additional bytes are automatically allocated per scan or per buffer (see SYST:STAMP).

Data in a buffer can be copied from one buffer to another to change the data format and/or to apply a scale and offset to the data using the BUFF MOVE command. For example, this command allows buffered A/D data collected in a 12 bit format to be scaled, offset, and then stored in another buffer in a 16 bit format for output to a 16 bit D/A.

Triggering and Program Control

The 576 supports the use of conditional triggers, similar to BASIC'S IF...ELSE...ENDIF constructs. These permit a sequence of user-specified commands to be executed based on the trigger condition being true or false. The trigger conditions compare an input to a user specified value or to two user values in the case of windowed triggering. Triggering can be done on any 576 input data, the DATE and / or TIME, or on the status of any buffer (full or not full).

The 576 also provides an analog trigger, using the TRIG command, for time critical triggering. The TRIG command is used in conjunction with the READ command in the QUICK mode.

Looping constructs are also supported, permitting the user to implement a standard DO...LOOP and WHILE...WEND loop construct similar to that used in BASIC. The WHILE triggering can be done on any 576 input data (also allows windowed trigger), the DATE and / or TIME, or on buffer status (full or not full).

Delays can be set in a 576 program using the WAIT command. This command can be used to halt a program an absolute amount of time, from 1 millisecond to 65 minutes, or until a group execute trigger (GET) is received. There is also a HALT command which can stop the program execution on an SRQ condition or when the command is encountered.

The 576 can be programmed when to begin execution of the stored program. Use the SYST :TRIG function to select the start option. The options include: start on receipt of X command, start on receipt of group execute trigger (GET), or start at a specified TIME and/or DATE.

Timestamping

Several modes for timestamping of data are provided with the 576. The SYST :STAMP function is used to configure the timestamp mode. Timestamping can be configured to stamp a buffer one time (ONCE) or can stamp the buffer at the start of each scan of data (SCAN). The type of timestamp can be configured to contain the time only (TIME) or may contain both time and date (CLOCK). The resolution of the timestamp is 10 milliseconds. When timestamping is enabled and hardware triggering (TRIG) is being used the 576 applies a special timestamp mode with a resolution of 1 microsecond.

Timestamping of a buffer is enabled through the dimension buffer (BUFF DIM) command by selecting the optional timestamp specifier (STAMP). The 576 will automatically allocate the appropriate amount of buffer space based on the timestamp mode and type set in the SYST :STAMP command.

Subroutines

A subroutine in the 576 is a series of commands beginning with a SUBR "name" command and ending with an ENDSUB command. The SUBR "name" command assigns a name to the subroutine which can later be called with the CALL "name" command, or to execute the subroutine when an interrupt occurs, it can be called with the ONINT "name" command at the interrupt rate specified in the command. The RETSUB command can be used at any time in a subroutine to return to the place in the program from which the subroutine was called.

When using the ONINT command only the last specified ONINT subroutine name will be active. Ample time must be allotted in the interrupt rate specified to allow complete execution of the subroutine or an interrupt overrun error will occur.

Up to ten user defined subroutines can be loaded in the system at one time. Any 576 command can be used within a subroutine.

Using subroutines and the ONINT command allows the capability to chain subroutines. This is done by issuing the ONINT command from within a subroutine to execute a different subroutine at the next interrupt. The interrupt rate must be set high enough to allow time to complete execution of the longest subroutine. With this feature the user can create a time - sliced multiprogram system in the 576. The RETSUB command used in a subroutine with interrupts active allows the creation of a limited time sliced multitasking system. The command INTTOFF is used to turn off interrupts.

Engineering Units Conversion and Data Formats

Engineering units conversions include volts, milliamps, hertz, and degrees C or F for thermocouple types J, K, S, T, E, B, and R, and RTDs. The default conversion is set using the SYST :UNIT function. The system default can be overridden using the optional "unit" parameter in various commands.

Data formats supported in the 576 are MOTOROLA binary, INTEL binary, ASCII with a prefix, and ASCII without a prefix. The default data format is set using the SYST :FORMAT function.

Using this section of the manual

This section of the 576 manual contains an alphabetic listing of all the commands used in controlling the 576. Each command includes a description of the command, the command syntax or format, programming notes, and programming examples.

The programming examples are shown for selected commands. These programming examples list only the 576 commands required to do a given task; they do not include any other syntax which may be specific to any controller programming language. Actual test programs can be written by combining the supplied 576 commands and appropriate commands from the user's chosen programming language environment. Comments may follow some commands, and are preceded by an apostrophe ('). If entered into a program, these comments must adhere to the specific language's conventions for identifying comments or remarks.

Some important notes on 576 commands

1. **ALL 576 COMMANDS** must be terminated with a semi-colon (;).
2. Commands can be delimited with a space, a comma, or a tab.
3. Only the first four characters of any command or function are required by the 576. Those commands having less than four characters require the full name. Any characters entered after the first four characters are ignored. Using the full command names are recommended for good program documentation since all commands are very english-like.
4. The parameters in 576 commands are position dependent. All parameters, including optional parameters must be entered in the order documented in the manual or an error will be issued.
5. The 576 is case insensitive. Upper and lower case are accepted.

Terms and Conventions used in this Section

1. All 576 commands shown in this section will show the command name followed by a space and will separate the command parameters with commas.
2. Some commands contain optional parameters. Optional parameters are enclosed in square brackets ("["]") and are position dependent.
3. Some commands allow the use of command extenders. The command extender character is the colon (":") and is used to string functions in a single command together without reissuing the command each time.
4. Some commands require the selection of one of several options. The use of braces ("{ }") denotes that one of the items in the braces must be selected. A vertical bar ("|") is used to indicate an OR condition

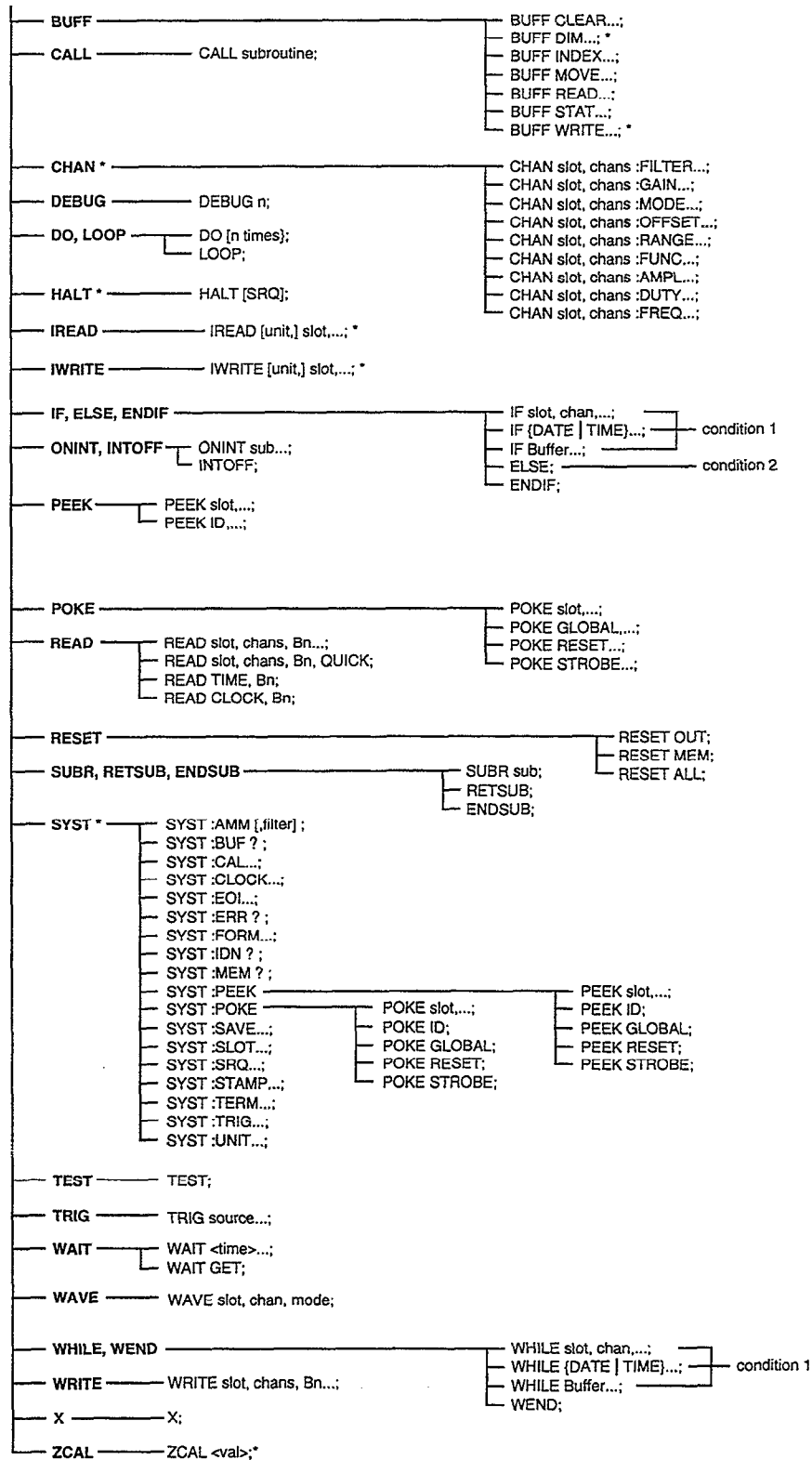
where the choice of one or more of the several options listed must be made.

5. "<>" denotes user supplied values or data.
6. All information shown in italics are controller specific commands.
7. Where a range of channels is specified, there can be no spaces included in the channel range parameter, i.e. "2-7" is correct; "2 - 7" is not.

576 PROGRAMMING COMMAND SET

*BUFF — Buffer Clear, Dim, Index, Move, Read, Stat, Write
CALL — Call Subroutine
**CHAN — Channel Configuration
DEBUG — Debug Tag
DO, LOOP — Loop Control
HALT — Halt
**IREAD — Immediate READ
**IWRITE — Immediate WRITE
IF, ELSE, ENDIF — Conditional Execution
ONINT, INTOFF — Setup Background Subroutine
PEEK — Peek at address
POKE — Poke to address
READ — Read Input Device
RESET — Reset System
SUBR, RETSUB, ENDSUB — Create Subroutine
**SYST — System Configuration
TEST — Self Test
TRIG — Hardware Trigger
WAIT — Time or GET delay
WAVE — Control WAV module output.
WHILE, WEND — While Condition Control
WRITE — Write Output Device
X — Execute
**ZCAL — Zener reference calibration

* BUFF DIM and BUFF WRITE are immediate commands
** = Immediate commands



* = Immediate Commands

BUFF CLEAR

Clear buffer contents

Purpose The BUFF CLEAR command clears the contents of the specified buffer.

Format **BUFF CLEAR Bn;**

Parameters Bn -- any valid buffer number

**Programming
Notes**

1. A "BUFF READ Bn" command (no [chan] option) automatically reads back the entire contents of the specified buffer and then clears the contents of a buffer.
2. A "BUFF READ [chan] Bn" command reads the specified channel's data from a buffer but does not clear the contents of the buffer. Therefore, it is necessary to use BUFF CLEAR after a "BUFF READ [chan] Bn" command before any new data can be put into the buffer.

BUFF DIM

Dimension buffer

Purpose To dimension and allocate a data buffer for input or output operations. There are up to twenty buffers that may be allocated from system memory, named B0 through B19.

Format **BUFF DIM [type,]Bn, #chan, #scan [,STAMP];**

Parameters

type	- optional array type specifier, default is WORD (16 bits). BYTE = 8 bits, WORD = 16 bits (default), LONG = 32 bits, TC = temperature array. See programming notes 1 and 2 below.
Bn	- number of the buffer to be allocated: B0-B19.
#chan	- number of channels per scan, up to 32 maximum.
#scan	- number of scans per channel to allocate for the buffer.
STAMP	- optional flag to enable timestamping on specified buffer.

Programming Notes

1. BUFF DIM is an immediate command.
2. The buffer dimension 'type' needed to store a reading varies with the module and type of reading:

Digital reading (port)	BYTE
Analog reading	WORD (non thermocouple)
Thermocouple reading	TC allocates 1 WORD per reading +1 WORD per scan for the cold reference junction. Example: Ch0 data value 1, Ch1 data value 1, reference 1, Ch0 data value 2, Ch1 data value 2, reference 2, . . . Ch0 data value 5, Ch1 data value 5, reference 5
PIM1, PIM2 (16-bit count)	WORD
PIM2 (32-bit count)	LONG
PCM1, DIO1A	BYTE
TIME	LONG (for READ TIME only)
CLOCK	LONG (for READ CLOCK only, must specify 2 in '#chan')
3. When dimensioning an array for temperature measurements using thermocouples the array must be of type 'TC'. This will allocate an additional WORD per scan to store the cold junction reference data which is read on each scan. This is only valid on modules with a cold junction reference (AIM5, AIM7).
4. If TIME is read with the READ command, the buffer must be dimensioned for type 'LONG' and one channel. If CLOCK is used, the type is also 'LONG' but '#chans' must be 2, one long for the date and another for the time. See READ TIME command for more information on reading real time clock information to buffers.
5. See the BUFF READ or BUFF WRITE command for more information on using buffers.
6. See SYST :BUF?; command for information on checking buffer status.
7. See SYST :MEM?; command for information on finding blocks of free memory.

8. To conserve system memory space specify the optional array type specifier when dimensioning arrays for digital I/O or PIM I/O.
9. Timestamping requires the 576 to allocate additional space per buffer. 6 bytes per buffer or scan (depending on the mode and type) is allocated automatically by the 576 if the optional STAMP flag is specified. See SYST :STAMP for more information on timestamp configuration.

Programming Examples

Example 1. Temperature Array

```

BUFF DIM TC,B0,1,10;      'allocate a TC array for 1 channel of temperature 10 scans.
                          'Note: an extra WORD (2 bytes) per scan will be allocated in
                          ' B0 for the cold junction reference.

SYST :SLOT 3, AIM7;      'AIM7 in slot 3
READ 3,5,B0,FILL;      'read AIM7 in slot 3, channel 5 and cold junction reference to
X;                       ' buffer 0, fill buffer.

```

Example 2. Timestamped Array

```

SYST :STAMP SCAN,CLOCK;  'time stamp time and date on every scan
BUFF DIM B2,2,100,STAMP; 'allocate word array for 2 channels of 100 scans. Note: 6 ex
                          'tra bytes allocated per scan for timestamp.

DO 100;
  IF 1,2,GT,3,DCV;      'if voltage on slot 1 channel 2 greater than 3 volts
    READ 1,0-1,B2;      'read slot 1 channels 0-1 with a time/date stamp
  ENDIF;
LOOP;
X;

```

BUFF INDEX

Report last point accessed in buffer

Purpose The BUFF INDEX command returns the number of scans of data that have been written to the specified buffer. The information is returned as an unsigned long integer.

Format BUFF INDEX Bn;

Parameters Bn – valid 576 buffer number, B0-B19

Programming Notes

1. The buffer specified in BUFF INDEX must exist or an error will be generated.

BUFF MOVE

Translate/move buffer contents

Purpose The BUFF MOVE command copies data from one buffer to another translating it to the format required by the output device which is identified by the module in the specified slot. This command also allows a scale and / or offset ($y=mx+b$) to be applied during the conversion. The scale value is a floating point multiplier and the offset value is in units of volts.

Format **BUFF MOVE slot,chans,Bnew, Bold,[,<scale>,<offset>];**

Parameters

slot	- slot number of the module which is the destination for the translated data.
chans	- channel or range of channels that specify data conversion.
Bnew	- number of the buffer to store the translated data: B0-B19.
Bold	- number of the buffer to translate: B0-B19.
scale	- (optional) parameter which will scale the buffer data as it is being converted from one slot format to another.
offset	- (optional) parameter which will offset the buffer data as it is being converted from one slot format to another.

Programming Notes

1. Buff Move permits scaling and offset of data for use by a subsequent READ or WRITE command.
2. 'Bold' and 'Bnew' must be dimensioned identically.
3. Scale and offset are any valid single precision floating point number.
4. If values for scale or offset are out of valid range for the input/output device the value will be clipped at the maximum or minimum value for the device. If values are scaled down, rounding errors may result in loss of resolution of original signal.
5. The offset value is in units of volts only.
6. If the data in 'Bold' is to be read to the controller, it should be read before the BMOVE command is issued. BUFF MOVE resets the buffer pointers and disables subsequent BUFF READ's of the data. BUFF MOVE may be issued more than one time on Bold.
7. BUFF MOVE will not accept buffers dimensioned with "STAMP".

Programming Examples

SYST :SLOT 3,AOM4;	'Slot 3 has an AOM4
BUFF DIM B0, 2, 100;	
BUFF DIM B1, 2, 100;	
BUFF DIM B2, 2, 100;	
READ 1,0-1,B0,FILL;	'read 2 channels, 100 pts each into buffer 0
BUFF MOVE 4,0-1,B2,B0;	'translate B0 data to 12 bits + sign for AOM5
BUFF MOVE 3,1-2,B1,B0,2,.1;	'translate B0 data to 12 bit for AOM4 with a gain of 2 and an ' offset of +.1V ($y = 2x + .1$)
WRITE 3,1-2,B1,1;	'write 1 cycle of B1 to AOM4 chans 1-2
WRITE 4,0-1,B2,1;	'write 1 cycle of B2 to AOM5 chans 0-1
X;	'execute program

BUFF READ

Read data from a buffer

Purpose The BUFF READ command reads a specified channel's data from a specified data buffer or channel into the controller's memory. The data is read from the buffer, converted to specified engineering units and data format, and buffered for reading by the host through the GPIB port. If a channel is not specified, the entire buffer is returned.

Format **BUFF READ [unit,][chan,]Bn;**

Parameters

unit – (optional) EU specifier, overrides system default if specified.
RAW = raw A/D counts no EU conversion
DCV = volts

TCn [C|F] = TC type n: J, K, S, T, E, B, or R. Reading in C or F.
RTDn [C|F] = RTD type n: 85 or 92. Reading in C or F.
HZ = Hertz
TIME = data to be read is time information
CLOCK = data to be read is date/time information

chan – optional channel number. If the chan option is used, only the specified channel's data is returned to the controller. If the chan option is not used, the entire buffer will be returned. The CHAN option is not valid with TIME or CLOCK units.

Bn – number of the buffer to use: B0–B19

Programming Notes

1. Data is read from the buffer according to the units specified in the command. The system defaults are used if the units option is not specified. See SYST:UNIT and SYST:FORMAT for information on setting up default system engineering units and data transfer formats.
2. If the specified buffer or channel does not exist, an error will be generated.
3. Each BUFF READ of a buffer returns the contents of the entire buffer. If the [chan] option is used, data will be returned only for the specified channel. After a BUFF READ without [chan] option, the buffer may be reused immediately. If a BUFF READ is performed with [chan] option, the buffer may not be reused until either a BUFF READ of the entire buffer is performed, a BUFF CLEAR is executed.
4. When reading a buffer that was configured for time stamping, the data is returned in the following format:

Mode,Type	Output Format
ONCE, TIME	TIME,<scan1 data><scan2 data>...
ONCE, CLOCK	DATE,TIME,<scan1 data><scan2 data>...
SCAN, TIME	TIME,<scan1 data>,TIME,<scan2 data>...
SCAN, CLOCK	DATE,TIME,<scan1 data>,DATE,TIME,<scan2 data>...

5. Timestamp information is returned in ASCII format except when engineering units of "RAW" is specified. If the engineering units flag is "RAW", data is returned in packed BCD format. See the appendix section for information on buffer format.
6. The TIME and CLOCK unit options must be specified when the buffer data is collected using READ TIME or READ CLOCK commands. TIME and CLOCK engineering units cannot be used if the "[chan]" option is used.
7. A BUFF CLEAR command must be issued to clear a buffer's contents before a READ command can place new data in the buffer.
8. Time/Date information is returned in ASCII format except when engineering units of "RAW" is specified. Engineering units of "RAW" returns data in packed BCD format.

BUFF STAT

Calculate buffer statistics

Purpose The BUFF STAT command provides Minimum, Maximum, and Mean values for the specified buffer channel. This function generates statistical information based on the currently selected Engineering Units.

Format **BUFF STAT [unit,][chan,]Bn;**

Parameters

unit	– Optional Engineering Unit conversion specifier. Use to override the system default settings. TIME and CLOCK are invalid Units for this function.
chan	– Optional Channel specifier. If the optional channel is specified, then the statistics returned are for the given channel only.
Bn	– Valid 576 buffer number, B0-B19.

Programming Notes

1. The format of the data returned from the BUFF STAT command is:
<Min Value>,<Max Value>,<Mean Value>
2. If the optional channel number is not specified, then the BUFF STAT command returns a minimum, maximum, and mean value for each channel in the buffer.
3. This function does not perform a buffer clear operation upon completion.

BUFF WRITE

Write/convert data to buffer

Purpose Writes data from controller to selected data buffer. Everything sent to the 576 after the BUFF WRITE command will be considered data until the specified number of samples for each channel have been placed in the data array.

Format **BUFF WRITE [unit,]slot,chans,Bn,<data>,<data>,<data>,...<data>;**

Parameters

unit	- (optional) engineering units specifier, default is system default. RAW = Raw A/D counts, no conversion DCV = Volts MA = Milliamperes
slot	- slot number of the module the data is for.
chans	- channel or range of channels to be used. A range of channels is specified as "start chan-stop chan" (eg. "2-5").
Bn	- number of the buffer to be written: B0-B19.
<data>	- stream of data sent to the specified buffer. The number of bytes, words, or longs placed in the buffer is specified by the BUFF DIM command.

Programming Notes

1. BUFF WRITE is an immediate command.
2. As data is written to the 576 it is converted from the specified data format and engineering units to raw binary based on the configuration of the specified channels. The system default unit is used if the option is not specified. See SYST :UNIT and SYST :FORMAT for information on setting up default engineering units conversions and data transfer formats.
3. If the specified buffer does not exist an error will be generated.
4. The number of samples that must be written to the buffer are specified in the BUFF DIM command for the buffer. The system will generate an error if a line terminator is received before all data is transmitted.

CALL

Call subroutine

Purpose The CALL command allows the inline execution of a subroutine created using the SUBR, ENDSUB commands.

Format CALL subname;

Parameters subname – name of a user defined subroutine.

- Programming Notes**
1. The subroutine must exist at run time or an error will occur during program execution.
 2. See the SUBR, RETSUB, ENDSUB commands for information on creating subroutines.
 3. See the ONINT, INTOFF command for information about background subroutine execution.

Programming Examples

```
SUBR TstB0Ful;           'Create subroutine TstB0Ful
  IF B0, FU;             'Test if buffer 0 is FULL
    BUFF READ DCV, B0;   'If B0 is full, send data to host
  ENDIF;
ENDSUB;                  'End subroutine
```

```
Main Program
BUFF DIM BYTE, B0, 3, 100; 'Create buffer 0 for 3 channels, 100 bytes each
DO;                          'Loop forever
  READ 5, 0-2, B0;          'Take 1 reading, slot 5 DIO1A , ports 0-2 to B0
  CALL TstB0Ful;           ' Call subroutine to test for buffer full
LOOP;
X;
```

Purpose The CHAN command is used to configure and interrogate channel functions. These functions are listed below..

General CHAN slot,chans :function[,options];

Format CHAN slot,chans :function ?;

Parameters

slot	– slot number of the module to be configured.
chans	– channel or range of channels to be configured. A range of channels is specified as 'start chan-stop chan' (eg. '2-4'). Legal values depend on the module type.
:function	– module function to be configured or interrogated. The CHAN command must always be followed by a function. If the function is followed by a "?", the setup or value of the specified function will be returned. If the function is followed by parameters, the function will be configured by the specified data.
options	– function parameters, required and/or optional, for the specified function. See specific function for details.

Functions

:FILTER	sets the filter for the specified channel
:GAIN	sets the total channel gain to the specified value
:MODE	set up operating mode of module (single-ended/differential, gated/normal, read/read&reset, input/output)
:OFFSET	enable or disable offset (AIM8 and WAV1.)
:RANGE	sets the range of an A/D, D/A, PIM1 frequency, or WAV1 module.
:FUNC	sets wave shape. Used for WAV1 module.
:AMPL	sets wave amplitude. Used for WAV1 module.
:DUTY	sets wave duty cycle. Used for WAV1 module.
:FREQ	sets wave frequency. Used for WAV1 module.

Programming Notes

1. All CHAN commands are immediate.
2. The use of the colon (":") in front of a function is required. It is used as a command extender to allow multiple functions to be declared in a single command. For example to configure a channel on an AMM module:

```
CHAN 1, 3 :GAIN 20 :MODE DF :RANGE 10B;
```

sets channel 3 of the AMM in slot 1 to a gain of 20, in differential input mode, on a $\pm 10V$ A/D range.
3. The AMM filter can be programmed on a channel-by-channel basis with the SYSTEM:AMM command.
4. See the RESET command for information on resetting module functions to their default states.

CHAN :FILTER

Set local filter

Purpose The FILTER function allows the user to select which local filter will be used during the acquisition of data. The command must precede the acquisition or a default value will be used.

Format CHAN slot,chans :FILTER val;
CHAN slot,chans :FILTER ?;

Parameters

val – filter value, depending on the module in the slot.
? – returns the current filter setting for the slot/chan in the form: "FILTER [chans] val[, val]...".

Programming Notes

1. All CHAN commands are immediate.
2. Once a filter value has been assigned to a channel, it remains in use until another filter command is issued for that channel, or the 576 is reset to the power-up state.
3. AMM modules do not have local filters. See the SYST :AMM command for information on configuring the system filter on the AMM1A and AMM2 modules.
4. If wideband filter operation is desired on the AIM8 module, specify "NONE" as the val parameter.

Programming Examples

```
SYST :SLOT 3, AIM9;           ' AIM9 in option slot 3
SYST :AMM 100K;              ' Set AMM system filter to 100KHz
CHAN 3,0 :FILTER 2;          ' AIM9 filter = 2Hz
READ 3,0,B1;                 ' Read slot 3, channel 0 into buffer 1
BUFF READ B1;                ' Read buffer 1
X;
```

Module	Loc. Filter	Val
AIM8	10Hz	10
	1kHz	1K
	WIDEBAND ¹	"NONE" ²
AIM9	2Hz	2
	20Hz	20
	200Hz ¹	200
TRG1	300Hz	300
	1kHz	1K
	3kHz	3K
	10kHz	10K
	30kHz	30K
	100kHz	100K
	300kHz	300K
1MHz ¹	1M	
Note 3	n Hz	n
	n kHz	nK
	n MHz	nM

1. Default
2. Use "NONE" (without quotes) for wideband filter
3. General format, where filter information for module specifies nHertz, nkHz, or nMHz. See module for available filters.

CHAN :GAIN

Set channel gain

Purpose The GAIN function allows the user to set the total gain desired on the specified channel(s). The function will set the programmable global gain amplifier of the master analog module in slot 1, and the local programmable gain amplifier of analog input modules with software programmable gain amplifiers, or the gain switch settings to the 576 so that engineering units conversions can compensate for the gain applied.

Format CHAN slot,chans :GAIN total gain;
CHAN slot,chans :GAIN global, local;
CHAN slot,chans :GAIN ?;

Parameters

- total gain – the value of the total gain to be applied to the specified chans.
- global – optional gain specifier for the global gain setting applied to the analog input channels (must be used with local gain).
- local – optional gain specifier used to notify the 576 of the local gain setting of the analog input chans (must be used with global gain).
- ? – returns the current gain setting for the slot/chan in the form: “GAIN [chans] val, local [val, local]...”.

Default Gain switch defaults to X1 on all slots where applicable. Gain local defaults to the lowest value applicable for each slot.

- Programming Notes**
1. All CHAN commands are immediate.
 2. This command will set the total channel gain. It will always set the highest gain possible nearest to the input signal and the rest of the gain at the AMM programmable gain amplifier. It is possible to alter this by specifying the ‘local gain’ that you desire closest to the input signal.
 3. For modules with switch selectable gain, the “local” gain parameter must be issued if the switch is set to a gain other than the default gain.
 4. The ‘local gain’ parameter can also be used to override the gain settings set by the system as described in program note 1.

Programming Examples

BUFF DIM B0, 4, 100;	‘Dimension B0 for 4 channels, 100 scans
CHAN 1, 0-2 :GAIN 100;	‘Set total gain on channels 0, 1, 2 to 100 (x10 local, x10 global)
CHAN 1, 3 :GAIN 10, 1;	‘Set total gain on channel 3 to 10 (override local to x1, global to x10)
READ 1, 0-3, B0, FILL;	‘Read channels 0-3 to B0 and fill buffer
BUFF READ B0;	‘Read buffer B0
X;	‘Execute

Module Gains and Input Ranges				
Module	Global	Local	Default	Voltage Range ¹
AMM1A AMM2	1, 2, 5, 10	1, 10	1	10 / 1 / 0.1
AIM2	AMM global gain may be applied to any analog input channel	1	1	10 / 1
AIM3A		1, 10, 100	1	10 / 1 / 0.1 / 0.01
AIM4 ²		1, 100	1	5 / 0.5 / 0.05 / 0.005
AIM5 ²		100	100	0.05 / 0.005 / TC
AIM6		50	50	1000 Ω RTD Module
AIM7		100	100	1.0 / 0.01 / TC
AIM8		1, 10, 100, 1000	1	10 / 1 / 0.1 / 0.01 / 0.001
AIM9 ³		1	1	LVDT / RVDT Module
TRG1 ⁴		1, 10	1	10 / 1 / 0.1
All analog input modules	When specifying TOTAL GAIN: ⁵ Total = Local × Global		Same as above	Same as above

1. Voltage shown at gains of x1, x10, ... x10,000 as is available with a given module. Other intermediate ranges are available using global gains of x2 or x5.
2. Isolated input. Maximum input = 5V for AIM4, 0.05V for AIM5.
3. Module gain continuously adjustable from x1 to x20. Firmware always treats local gain as x1.
4. When TRG1 is used as a single analog input channel.
5. Where several combinations equal the same total, highest appropriate local gain will be used, with any remainder applied as global.

CHAN :MODE

Set module operating mode

Purpose The MODE function sets the mode of operation of the module in the specified slot.

Format CHAN slot,chans :MODE mode;
CHAN slot,chans :MODE ?;

Parameters

- mode – operating mode for the module in the indicated slot. See table below.
- ? – returns the current mode setting for the slot/chan in the form:
"MODE [chans] mode [,option] mode[, option]..."

Module	modes	Explanation
AMM1A AMM2 AIM3A	SE or DF	Single-ended mode Differential mode
DIO1 DIO1A	IN or OUT	Set port for Input Set port for Output
PIM1	NORM, {CONT RESET} GATE, {CONT RESET} FREQ	Normal event counting mode, continuous count or read/reset Gated event counting mode, continuous count or read/reset Frequency mode
PIM2	CONT, {16 32} RESET, {16 32}	Continuous counting – read PIM2 and allow counting to continue, use specified chan in 16 or 32 bit mode Read and Reset – read PIM2 value and immediately restart count at 0, use specified chan in 16 or 32 bit mode

Default

On power up, modes are set as follows:
AMM and AIM3A – Single ended (SE). Other defaults ($\pm 10V$ A/D (10B) and gain = x1) are set by other commands.
DIO1/DIOA – Ports 0 and 1 are input, ports 2 and 3 are output
PIM1 – Frequency mode and 62.5k range
PIM2 – 16-bit reading with reset
AOM5 – $\pm 10V$ (10B)

Programming Notes

1. All CHAN commands are immediate.
2. The PIM2 32-bit mode uses channels 0-1 as one channel pair and 2-3 as another and the mode assigned is used by the pair. In 16 bit mode each channel may have individual mode settings.
3. When an input mode is set for analog input, it applies to all channels on the associated module. Single-ended and differential can not be mixed on one module.

Programming Examples

```
CHAN 3,0 :RANGE 62 :MODE FREQ; ' Set PIM1 in slot 3, chan 0 to frequency , range  
                                ' to 62.5kHz  
CHAN 3,0 :MODE RESET,32;      ' Set PIM2 in slot 3, chan 0 (and 1) to Read and  
                                ' Reset, 32 bits
```

CHAN :OFFSET

Set offset

Purpose This function is used to set the offset on a module. At present only the AIM8 and WAV modules implement this feature.

Format **CHAN slot, chans :OFFSET offs; (WAV1)**
CHAN slot,chans :OFFSET option; (AIM8)
CHAN slot,chans :OFFSET ?;

Parameters

- offs – the desired DC offset value, in volts, for a WAV1 module output waveform.
- option – is the condition of the offset
ENABLE = enable offset
DISABLE= disable offset
- ? – returns the current offset setting for the slot/chan in the form: "OFFSET [chans] option option...".

Default On power up the offset condition is DISABLE for an AIM8 and 0 volts for a WAV1.

Programming Notes

1. All CHAN commands are immediate.
2. When used with a WAV1 module, CHAN :OFFSET sets the DC bias of an output waveform. The offset cannot exceed the amplitude range set for the module (e.g. $\pm 1V$ or $\pm 10V$ for a WAV1).

Programming Examples

CHAN 3,0 :OFFSET ENABLE;	'Enable offset on AIM8 in slot 3, chan 0
CHAN 3,1 :OFFSET DISABLE;	'Disable offset on AIM8 in slot 3, chan 1
BUFF DIM B0, 2, 1;	'Dimension buffer
READ 3,0-1,B0;	'Read slot 3 channels 0-1 into buffer 0
BUFF READ B0;	'Read buffer 0
X;	'Execute

CHAN :RANGE

Set A/D, D/A, or frequency range

Purpose The RANGE function is used to inform the 576 of A/D, D/A, frequency, or voltage ranges on a slot and channel basis.

Format CHAN slot,chans :RANGE range;
CHAN slot,chans :RANGE freq; (PIM1)
CHAN slot,chans :RANGE ?;

Parameters

range – the operating range for the indicated module:

10B	= ±10V	(default) (for Analog Input)
10U	= 0-10V	
10B	= ±10V	(default) (for Analog Output)
10U	= 0-10V	
5B	= ±5V	
5U	= 0-5V	
2B	= ±2.5V or ±2V	
1B	= ±1V	

freq – Frequency range for counter module.

62K	= 62 kHz	(default) (for PIM1)
125K	= 125 kHz	
250K	= 250 kHz	
1M	= 1 MHz	
2M	= 2 MHz	
4M	= 4 MHz	
8M	= 8 MHz	

? – returns the range setting in the form: "RANGE [chans] range/freq range/freq ..."

Programming Notes

1. All CHAN commands are immediate.
2. On power up all ranges are set according to defaults.
3. WAV1 output range is set by a switch on the module. Range programmed with CHAN :RANGE must match this setting (i.e. 1B or 10B).

Programming Examples

CHAN 3,0-2 :RANGE 10B; 'Set the output range of channels 0 to 2 of the AOM1/5
in slot 3 to ±10V

CHAN :FUNC

Set waveform shape

Purpose Sets the output waveform shape for a WAVI module.

Format CHAN slot, chan :FUNC function

Parameters

slot	- slot in which module is mounted (3 = 576 option slot).
chan	- the channel being addressed.
function	- the desired output waveform shape:
	SINE = sine wave
	TRIA = triangle wave
	SQUA = square wave
	NONE = DC output, no waveform.

**Programming
Notes**

1. All CHAN commands are immediate.
2. The CHAN :FUNC command is used in conjunction with the WAVE command. Any errors in this command will not be reported until the program containing the WAVE command is executed.

CHAN :AMPL

Set output amplitude

Purpose Sets the peak-to-peak output amplitude of a WAV1 module.

Format CHAN slot, chan :AMPL amplitude

Parameters

- slot – slot in which module is mounted (3 = 576 option slot).
- chan – the channel being addressed.
- amplitude – the desired output amplitude, specified in volts peak-to-peak. The specified amplitude must be less than the maximum.

Programming Notes

1. All CHAN commands are immediate.
2. The CHAN :AMPL command is used in conjunction with the WAVE command. Any errors in this command will not be reported until the program containing the WAVE command is executed.
3. The specified amplitude must be less than the maximum by at least one bit's value. For example, the WAV1's nominal output range is 20V p-p. The maximum that can be programmed is 19.9951V for the 20V p-p range, or 2V p-p max value is 1.9951.
4. The WAV1 module has two output ranges ($\pm 1V$ or $\pm 10V$) which are selected via a switch on the module. The programmed amplitude must conform to the maximum permitted by the switch position.

CHAN :DUTY

Set output duty cycle

Purpose Sets the duty cycle for the waveform output by a WAV module.

Format CHAN slot, chan :DUTY duty_cycle

Parameters

- slot – slot in which module is mounted (3 = 576 option slot).
- chan – the channel being addressed.
- duty_cycle – the desired duty cycle, expressed as whole number percent, e.g. specify 50 for 50%.

Programming Notes

1. All CHAN commands are immediate.
2. The CHAN :DUTY command is used in conjunction with the WAVE command. Any errors in this command will not be reported until the program containing the WAVE command is executed.
3. The specified duty cycle must be within the permissible range for the module, e.g. 5–95(%) for the WAV1 module.

CHAN :FREQ

Set waveform frequency

Purpose Sets the output frequency for a WAV module.

Format CHAN slot, chan :FREQ frequency [,freq_range]

Parameters

- slot – slot in which module is mounted (3 = 576 option slot).
- chan – the channel being addressed.
- frequency – the desired frequency, expressed in Hz.
- freq_range – optional parameter which selects a specific frequency range:
 - AUTO =autorange to the optimum range.
 - 2 =2Hz range
 - 20 =20Hz range
 - 200 =200Hz range
 - 2K =2kHz range
 - 20K =20kHz range
 - 200K =200kHz range

Programming Notes

1. All CHAN commands are immediate.
2. The CHAN :FREQ command is used in conjunction with the WAVE command. Any errors in this command will not be reported until the program containing the WAVE command is executed.
3. The specified frequency must be within the permissible range of the module. If the freq_range option is omitted, or if AUTO is specified as the freq_range parameter, the firmware will select the best frequency range for the desired frequency and duty cycle.

Purpose

The DEBUG command allows the user to tag commands sent to the 576 for debugging purposes. Each time the DEBUG command is encountered, the value of its associated debug number will be stored in the serial poll byte, bits 0 through 2. If an error is encountered in a program, the value stored in the serial poll byte can be used to indicate the location in the program where an error occurred. By monitoring the serial poll byte, the DEBUG command can also be used to indicate the status of program execution.

Format

DEBUG number;

Parameters

number – is a number from 0 to 7.

Programming Notes

1. The numbers assigned to the debug command do not have to be in sequential order or start with the number 0.
2. See SYST :ERR? command for information on getting error information to the controller. A listing of error strings and an explanation of the error is included in the Appendix section of this manual.
3. See SYST :SRQ ERR; command for information on generating an SRQ for an error condition.

Programming Examples

1. Monitoring program status
DO;
 DEBUG 1; ' Set serial poll byte bits 0, 1, 2 to 1
 READ 1,0,B0,FILL; ' read CHANNEL 0
 DEBUG 2; ' set serial poll byte bits 0, 1, 2 to 2
 READ 1,1,B1,FILL; ' read channel 1
 DEBUG 3; ' set serial poll byte bits 0, 1, 2 to 3
 BUFF READ B0; ' read buffers B0 and B1
 BUFF READ B1;
LOOP;
X;

DO
SPOLL 03 ' Serial poll the 576
INPUT POLLBYTE ' Read the serial poll byte
IF BIT 0, 1, 2 = 1
 PROGRAM STATUS = READING CHAN 0
IF BIT 0, 1, 2 = 2
 PROGRAM STATUS = READING CHAN 1
IF BIT 0, 1, 2 = 3
 PROGRAM STATUS = READ BUFFERS
LOOP

DO LOOP

Loop control

Purpose The DO and LOOP commands are used together to define a program loop to be executed in the 576. The loop construct allows an operation similar to the BASIC FOR..NEXT loop.

Format DO [n];
LOOP;

Parameters n – loop count, 1 – 4,294,967,295 ($2^{32}-1$), or LOOP FOREVER if n is not specified.

- Programming Notes**
1. Every DO command must have an associated LOOP or an error will be issued.
 2. Loops can be nested to eight levels.

Programming Examples

Example 1. Execute program in loop indefinitely

```
DO;                                ' loop forever
<program>                          ' execute program
LOOP; X;                            ' loop end
```

Example 2. Execute program in loop 1000 times

```
DO 1000;                            ' Loop 1000 times
  READ 1,0,B0;                       ' Read slot 1 channel 0 into buffer 0
LOOP;                                ' End loop
X;
```

HALT

Halt program

Purpose The HALT command causes the 576 to stop executing the program. This command may be used with a conditional trigger command to stop the 576 from overwriting data collected prior to a specific condition becoming true.

Format **HALT [SRQ];**

Parameters SRQ – will stop program execution when an SRQ is issued. If SRQ is not included, program stops when HALT is issued.

Programming Notes

1. Halt on SRQ is an immediate command.
2. The HALT command, like a DEVICE CLEAR, stops program execution.
3. See the SYST :SRQ function for information on generating service requests.

Programming Examples

Example 1. Halt program execution on SRQ (buffer full)

```
SYST :SRQ BUFF;           ' Issue SRQ on buffer full
BUFF DIM B0,5,10;
HALT SRQ;                 ' Halt program on SRQ
DO;                       ' Loop forever
    READ 1,0-4,B0;        ' Read slot 1, channel 0-4, into buffer 0, 1 scan
    WAIT 1,MIN;          ' Sample about every minute.
LOOP; X;
```

WAIT ON SRQ
DEVICE CLEAR ' Reinitialize communication with 576
READ DATA

Example 2. Stop program execution on condition

```
DO;                       ' Loop forever
    IF 1,0, GT, 2.5, DCV; ' If slot 1, channel 0 >2.5V then
        HALT;            ' Halt
    ENDIF;               ' End if
    READ 1,0,B0;        ' Read slot 1, channel 0 into buffer 0
LOOP;                     ' End loop
X;                       ' Execute
```

IF ELSE ENDIF

Conditional execution

Purpose

The IF command allows the definition of a code segment that only gets executed if the conditional portion of the IF construct evaluates TRUE. IF, ELSE, and ENDIF can be nested to eight levels.

The ELSE command allows the definition of a segment of code that gets executed only if the condition expression portion of the IF command evaluates FALSE.

The ENDIF command terminates the IF command.

Format

```
IF slot,chan,[NOT,]cond,<val1> [[,<val2>] [,unit]];  
IF {DATE|TIME}, [NOT,]cond, <val1>;  
IF Bn, [NOT,]FU;  
    (action 1)  
ELSE;  
    (action 2 )  
ENDIF;
```

Parameters

slot	– location of the module being tested.
chan	– number of the channel being tested.
NOT	– optional specifier to test for condition being NOT TRUE.
cond	– expression which must evaluate TRUE. (See Valid Expression Table for all available condition operators)
<val1>	– value typically used in an expression when a single limit or low threshold value is required. If DATE or TIME is used val1 is the date, in the last format specified or the time.
<val 2>	– optional value typically used in an expression as an upper limit or high threshold value.
unit	– optional engineering units conversion specifier. The lower and upper threshold values, val1 and val2, will be used in the 'units' specified otherwise the default units will be used. See SYST: UNIT for setting default engineering units.
Bn	– number of the buffer to be tested: B0–B19. Test is for buffer full (FU) or not full (NOT,FU).

Programming Notes

1. If specifying date/time stamps you must specify the date in the same mode that was used to set the real time clock. Also note that you can not specify both the time and date in one IF command. To do conditional triggering on both date and time use two nested IF statements. See SYST :CLOCK for more information.

2. Every IF must have a matching ENDIF or a run time error will be issued upon receiving X (execute).
3. IF's can be nested 8 levels deep.
4. With Bit Test Operators the val1 parameter specifies the bits to be tested.
5. Nested IF's must be used to simulate a between (BT) or NOT between condition for time and date.
6. If the specified operator is BT, the lower and upper limit are excluded from the conditional test, i.e. VAL must be less than the high limit and greater than the low limit.
7. If the specified operator is NOT BT, then the low and high limits are included in the conditional test, i.e. the VAL must be greater than or equal to the high limit, or less than or equal to the lower limit.

Expression Table

Operator	Meaning	Val1 Required	Val2 Required
LT	Less Than	Yes	No
GT	Greater Than	Yes	No
EQ	Equal To	Yes	No
LE	Less Than or Equal To	Yes	No
GE	Greater Than or Equal To	Yes	No
FU	Buffer Full	No	No
BT	Between	Yes	Yes
OR	Bit Test – Any bits set in val	Yes	No
AND	Bit Test – All bits set in val	Yes	No
NOT OR	Bit test – all bits not set	Yes	No
NOT AND	Bit test – Any bit not set	Yes	No

Programming Examples

Example 1. Generic example to show nesting

```

IF ();          'level 1
IF ();          'level 2
IF ();          'level 3
ELSE;          'else level 3
  IF();         'level 4
  ELSE;         'else level 4
  ENDIF;       'end level 4
ENDIF;        'end level 3
ENDIF;        'end level 2
ELSE;         'else level 1
ENDIF        'end level 1

```

Example 2. Trigger Buffer Full Example

```

BUFF DIM B0, 1, 25;
DO;
  READ 1,0,B0;          'read slot 1, chan 0, to buffer 0
  IF B0,FU;             'when buffer becomes full
    BUFF READ B0;      'read buffer 0
  ENDIF;
LOOP;

```

Example 3. Trigger Conditional example taken 1 step further

```
SYST :UNIT TCJ,C;
DO;                                     'loop forever
  IF 1,0,GT,4000;                       'if slot 1,channel 0 is > than 4000 counts
    READ 1,0,B0;                         'read slot 1, channel 0
    BUFF READ B0;                         'read buffer
  ELSE;
    IF 3,0,LT,23;                         'if temp is less than 23 deg C
      READ 3,0,B1;                         'read temperature
      BUFF READ B1;                       'send buffer 1 up bus
    ENDIF;                                'end if (level 2)
  ENDIF;                                  'end if (level 1)
LOOP;                                    'end loop
```

Example 4. Trigger on DATE / TIME

```
IF DATE, GT, 4-4-1990;                  ' if the date is greater than April 4, 1990
  IF TIME, GT, 12:00;                   'and the time is greater than 12:00
    READ 1,0,B0;                         'read slot 1, channel 0
    BUFF READ B0;                         'send data up the bus
  ENDIF;
ENDIF;
```

IREAD

Immediate data read

Purpose The IREAD command is used to read data immediately from an input channel to the controller. The command reads data from the start channel to the stop channel of the specified slot, applies the specified engineering unit conversion, and immediately puts the data in the 576 output queue for reading by the GPIB controller.

Format IREAD [unit,] slot, chans [,<Avg>];

Parameters

unit – (optional) EU specifier, overrides system default if specified.

 RAW = raw D/A counts no EU conversion
 DCV = volts
 TCn [C|F] = TC type n: J, K, S, T, E, B, or R. Reading in C or F.
 RTDn [C|F] = RTD type n: 85 or 92. Reading in C or F.
 Hz = Hertz

slot – slot number of the module being read.

chans – channel or range of channels to be read (specify range as
 “start chan-stop chan”, e.g. “1-5”).

<Avg> – the number of points to be averaged (1-65535).

Programming Notes

1. IREAD is an immediate command, and not intended for program mode.
2. If the channel list is invalid for the module, an error will be generated.
3. If the specified engineering unit is illegal for the module, an error will be generated. (MA is illegal for IREAD).
4. Typical command for averaging: IREAD 1, 0, 100; will take 100 readings from channel 0, sum the readings, divide the sum by 100, and return the result to the 576 output queue.
5. See discussion of immediate vs program mode at the end of the command section.

IWRITE

Immediate data write

Purpose The IWRITE command is used to write data immediately from the controller to an output channel. The command writes data from the start channel to the stop channel of the specified slot.

Format IWRITE [unit,] slot, chans, <data>...;

Parameters

unit – (optional) EU specifier. System default is used if unit is not specified:

- RAW = A/D counts, no conversion
- DCV = volts
- MA = milliamperes

slot – slot number of the module the data is for.

chans – channel or range of channels (specify range as “start chan-stop chan”, e.g. “1-5”).

<data> – stream of data to the specified buffer.

Programming Notes

1. IWRITE is an immediate command, and not intended for program mode.
2. If the channel list is invalid for the module, an error will be generated.
3. If the specified engineering unit is illegal for the module, an error will be generated.
4. A data value must be specified for each channel in the list or an error will be generated.
5. When writing a negative voltage as a raw value to an AOM5 or slot 4, calculate the raw value as the required data bits plus 32768.
6. See discussion of immediate vs program mode at the end of the command section.

ONINT INTOFF

Setup background subroutine

Purpose The ONINT command allows the user to have a subroutine execute on an interrupt at the rate specified. This command gives the system a 'background' processing capability. This command also allows the chaining of subroutines on subsequent interrupts. The INTOFF command is used to turn interrupts off.

Format ONINT sub[,<rate>,<unit>];
INTOFF;

Parameters

sub	– the name of one of ten user definable subroutines. The subroutine name specified will be executed at the specified rate.
rate	– the magnitude of the interrupt period.
unit	– the time frame of the specified rate. The range of rate for each unit is shown below.
	USEC = microseconds, 125 – 65535
	MSEC = milliseconds, 1 – 65535
	SEC = seconds, 1 – 3267
	MIN = minutes, 1 – 65
	MLHZ = millihertz, 1 – 65535
	HZ = hertz, 1 – 8000

Programming Notes

1. Only the last issued ONINT command subroutine will be executed at the specified rate.rate. Every time ONINT is issued, 'sub' becomes the active background subroutine.
2. To suspend background activity, the command 'INTOFF;' will turn off interrupts. A subsequent ONINT command will resume execution of the specified routine.
3. The interrupt rate specified must allow adequate time to execute the subroutine specified in the background. If not enough time is specified to complete the background execution before the next interrupt, an interrupt overrun error will occur.

To determine the amount of time required to complete execution of subroutine send a program to the system which begins by reading the real time clock, then executes the subroutine in question using a CALL command, and finally reads the real time clock again. The execution time of the subroutine can be calculated by subtracting the first real time clock reading from the last.

```
READ TIME,B2;           ' Read real time clock to buffer 2
CALL GETDATA;          ' Execute subroutine GETDATA using a CALL
READ TIME,B3;          ' Read real time clock to buffer 3
BUFF READ TIME,B3;     ' Read real time clock values in buffers
BUFF READ TIME,B2;
TIME DIFFERENCE B2 to B3 ' Calculate elapsed time to execute subroutine
```

If the time difference is 0, the timed routine executes in less than 10 milliseconds.

4. When changing interrupt rates, the new rate will take effect upon completion of the execution of the present interrupt cycle.

5. If a READ ..,QUICK is encountered while interrupts are active, interrupts will be suspended until the READ is complete. See READ for more information.
6. If no subroutines have been defined, the error "Only 10 subroutines can be defined" will be issued.

Programming Examples

```

Example 1. Simple subroutine in background
BUFF DIM TC, B0, 3, 100;      ' Dimension arrays
BUFF DIM B1, 5, 100;
ONINT GETDATA,1, SEC;       ' Execute subroutine GETDATA in background at a rate
                             ' of every 1 second
DO;                           ' loop forever
  IF B0, FU;                  ' if buffer 0 full
    BUFF READ TCJ,F,B0;      ' read B0, J type thermocouple deg F
  ENDIF;
  IF B1, FU;                  ' if buffer 1 full
    BUFF READ DCV,B1;       ' read B1, DCV
  ENDIF;
LOOP;
SUBR GETDATA;                ' Create subroutine GETDATA
  READ 3, 1-3, B0;          ' Read AIM7 slot 3, channels 1 to 3 to buffer 0
  READ 1, 0-4, B1;         ' Read AMM slot 1, channels 0 to 4 to buffer 1
ENDSUB;
X;

```

```

Example 2. Example of multi-tasking subroutines
BUFF DIM TC, B0, 3, 100;      ' Dimension arrays
BUFF DIM B1, 5, 100;
ONINT GETTEMP,1, SEC;       ' Execute subroutine GETTEMP in background at a rate
                             ' of every 1 second
DO;                           ' Loop forever
  IF B0, FU;                  ' If buffer 0 full
    BUFF READ TCJ,F,B0;      ' read buffer 0, J type thermocouple degrees F
  ENDIF;
  IF B1, FU;                  ' if buffer 1 full
    BUFF READ DCV, B1;       ' read buffer 1, DCV
  ENDIF;
LOOP;
SUBR GETTEMP;                ' create subroutine GETTEMP
  READ 3, 1-3, B0;          ' read AIM7 slot 3, channels 1 to 3 to buffer 0
  ONINT GETAMM;             ' change to subroutine GETAMM next interrupt
ENDSUB;
SUBR GETAMM;                 ' create subroutine GETAMM
  READ 1, 0-4, B1;         ' read AMM slot 1, channels 0 to 4 to buffer 1
  ONINT GETTEMP;           ' change to subroutine GETTEMP next interrupt
ENDSUB;
X;

```

PEEK

Read data at address

Purpose The PEEK command allows the user to examine the contents of the 576 hardware, and places the data in the specified buffer.

Format PEEK slot, cmd, Bn [,FILL];
PEEK ID, Bn [,FILL];

Parameters

- slot – the slot number of the module to be read. See table below.
- cmd – the command to be used for the read. Valid commands are A, B, C, or D. See table below.
- Bn – the number of the buffer to use: B0-B19.
- FILL – (optional) flag used to specify peek until buffer full.
- ID – reads the contents at the self ID address location

Slot	Function	Command
1	option slot 1	A B C D
2	TRG1	A B C
3	option slot 2	A B C
4	AOM5	A B
5	DIO1A	A B

Programming Notes

1. PEEKing at invalid locations will generate an error.
2. Buffers should be dimensioned using "BYTE" type specifier.
3. See 576 reference section for details on CMDA, CMDB, CMDC, and CMDD.

Programming Examples

```
BUFF DIM BYTE,B0,1,10;           ' dimension B0 for 10 byte inputs
PEEK 3,A,B0,FILL; X;             ' Store 10 samples of PEEK on Command A of module
                                  ' in slot 3 to B0

BUFF DIM BYTE,B1,1,10;           ' dimension B1 for 10 byte inputs
PEEK 3,B,B1; X;                  ' Store a sample of PEEK on Command B of module in
                                  ' slot 3 to B1
```

POKE

Write data to address

Purpose The POKE command allows the user to write data (0-255) to any accessible 576 hardware location from the specified buffer.

Format POKE slot, cmd, {Bn [,CYCLES] | <val>};
POKE GLOBAL, {Bn [,CYCLES] | <val>};
POKE RESET, {Bn [,CYCLES] | <val>};
POKE STROBE, {Bn [,CYCLES] | <val>};

Parameters

- slot – the slot number of the module to be written. See table below.
- cmd – the command to be used for the write. Valid commands are A, B, C, or D. See table below.
- Bn – the number of the buffer to use: B0-B19.
- CYCLES – the number of cycles through the buffer to perform, if not specified one data value is written. Maximum number of CYCLES is 65535. See Note 4.
- <val> – a byte value from 0-255. Data is evaluated MODULO 256
- GLOBAL – write data to the GLOBAL 1 location.
- RESET – write data to the RESET location. (aka GLOBAL 2).
- STROBE – write data to the GLOBAL STROBE location.

Slot	Function	Command
1	option slot 1	A B C D
2	TRG1	A B C
3	option slot 2	A B C
4	AOM5	A B
5	DIO1A	A B

Programming Notes

1. Poking at invalid locations will generate an error.
2. CAUTION: Poking at STROBE, GLOBAL, and RESET will cause a strobe on the selected lines to all modules.
3. Buffers should be dimensioned using 'BYTE' type specifier.
4. The maximum number for CYCLES may be less than 65535. The maximum number of cycles is $2^{32} / \text{\#scans}$. If the specified CYCLES number is too large the 576 will generate an error.
5. See 576 reference section for details on CMDA, CMDB, CMDC, and CMDD.

Programming Examples

```
Poke an array or 100 points to the PROTO module in slot 3
BUFF DIM BYTE,B0,1,100;           'setup buffer for 100 points
BUFF WRITE RAW,3,0,B0, <100 data bytes>; 'fill buffer with byte data
POKE 3,A,B0,2;                    'write 200 bytes to PROTO card slot 3,
X;                                 'command A . (2 cycles of 100 points)
```

READ

Read data from channel

Purpose Read Input(s) — Sets up the 576 to read a specified number of samples per channel from the starting channel to the stop channel of the specified slot and place the data into the specified buffer.

Format **READ slot, chans, Bn;**
READ slot, chans, Bn ,FILL;
READ slot, chans, Bn, QUICK;
READ slot, chans, Bn ,<Avg>;

Parameters

slot	– the slot number of the module to be read
chans	– the channel or range of channels to be read. A range of channels is specified as “start chan-stop chan” (eg. “2-5”).
Bn	– the number of the buffer to use: B0-B19.
FILL	– (optional) flag used to specify READ until buffer full
QUICK	– flag for high speed READ. (AMM1A - 31.25KHz, AMM2 - 25KHz)
<Avg>	– the number of points to be averaged (1-65535)

Programming Notes

1. A single READ command cannot scan across more than 1 card. Two read commands cannot access one buffer.
2. If the specified buffer does not exist an error will be generated.
3. When collecting data to a buffer and more samples are collected than memory allocated, the data collected beyond the buffer size is lost. The data will not be stored anywhere. To avoid this problem, don't collect data beyond the amount of memory allocated, or use the IF...ENDIF conditional commands and trigger on a buffer full (FU) condition and then read the data.
4. When type “TC” is specified in the dimension command for buffer, (for AIM5 or AIM7,) it forces the 576 to automatically read the cold reference junction on the module once per scan. See the BUFF DIM command for information on temperature reading.
5. The QUICK option can not be used with the AIM7 or AIM5 if “TC” is specified.
6. The QUICK option is only valid with selected analog input modules (AMM1A, AMM2, AIM2, and AIM3A) if multiple channels are specified. Errors will be generated for other modules. Gain or range cannot be changed in the middle of a scan.
7. The QUICK option will force the READ to run to completion before resuming program execution. This includes suspension of the ONINT command.
8. The QUICK option can be queued using the hardware trigger command (TRIG). See TRIG command for information on hardware triggering.
9. For single-channel acquisition, the QUICK option will allow acquisition rates of 50KHz on the AMM2 and 62.5KHz on the AMM1A. If timestamping is enabled, hardware triggering is used, or both, the rate will drop to 25KHz and 31.25KHz respectively.
10. See the BUFF DIM command for information on buffer setup and allocation. See SYST :STAMP and BUFF DIM for information on using the time stamping option with the READ command.

11. A typical average operation is as follows: for a command READ 1, 0, B0, 100; -- 100 readings are taken and summed. The sum is divided by the number of points (100) and stored as a single reading in buffer 0.

Programming Examples

Example 1. Read inputs and read buffer

```
BUFF DIM B0, 3, 10;           'setup buffer B0 for 3 channels, 10 scans
READ 1, 0-2, B0, FILL;       'read 10 scans of slot 1, channels 0-2 into B0
BUFF READ DCV, B0;          'format B0 data in DC volts for a read by the controller
X;
```

Example 2. Read Quick (576 with AMM2 in slot 1)

```
BUFF DIM B0, 8, 100;        'read 100 samples from slot 1 channels 0 to 7
READ 1, 0-7, B0, QUICK;     'into buffer 0 at 25KHz
BUFF READ B0;               'format B0 for a read in default units and format
X;
```

READ

Read real time clock

Purpose Read Real Time Clock – Reads the time and date information from the real time clock and stores the data to the specified buffer. The SYST :CLOCK command allows the real time clock to be read directly, (without specifying a buffer).

Format **READ TIME, Bn;** **Read time only**
 READ CLOCK, Bn; **Read date and time**

Parameters Bn – the number of the buffer to use: B0-B19. See the programming note for information on using the BUFF DIM command for these buffers.

- Programming Notes**
1. See the BUFF DIM command for information on allocating buffers for BUFFER READ operations of the real time clock.
 2. See SYST :CLOCK for more information on configuring the real time clock and doing immediate reads of the clock.

Programming Examples

BUFF DIM LONG, B2,1,1;	' setup buffer B2 for 1 long, 1 reading
BUFF DIM LONG, B3,1,1;	' setup buffer B3 for 1 long, 1 reading
READ TIME, B2;	' read time to B2
CALL ANYSUB;	' call a subroutine that will be timed
READ TIME, B3;	' read time to B3
BUFFER READ TIME, B2;	' format read of B2 and B3 to controller's memory
BUFF READ TIME, B3;	' as soon as 576 becomes talker.
X;	

RESET

Reset system

Purpose Reset specified sections of 576 hardware to power up state.

Format **RESET mode;**

Parameters

mode	– specifies what is to be reset
OUT	= clear outputs (i.e. set all analog and digital outputs to zero).
MEM	= clear memory (i.e. return all data memory to free space list).
ALL	= simulated cold boot (power up default state).

Programming Notes

1. When using "ALL", no other commands should be on the command line or awaiting execution. The ALL mode resets the 576 and clears the input buffer and program memory. Any commands on a line with ALL or pending execution will be lost. For example; the command string "RESET ALL;X;SYST:BUF?;" would cause the SYST:BUF?; command to be lost, resulting in a timeout error.
2. The RESET OUT; command sets:
 - DOM1 outputs to logic 1: (DOM1 uses inverting logic).
 - DIO1A port A and B outputs to logic 0: (if configured as outputs).
 - DIO1A port C and D outputs to logic 1: (if configured as outputs).
 - AOM outputs to 0V: (based on current range setting).
3. If the system is not programmed to reflect the actual switch settings of all AOM modules when RESET OUT is issued, other output voltages will result. See SYST:RANGE function for information on setting analog output ranges.
4. RESET ALL; contains execution of MEM and OUT and resets all CHAN and SYST commands to their default states. See appendix for module default conditions.
5. If a DEVICE CLEAR is sent immediately after a RESET ALL, the RESET ALL will not have sufficient time to initialize the 576. The RESET ALL command will have to be re-issued to fully reset the system. To assure that RESET ALL is complete, test the Serial Poll "IDLE" Bit after RESET ALL, and wait until the system is idle before sending additional commands.

Special Note

Beginning with firmware revision E02, and when you use a GPIB interface card having its operating software stored in ROM, you must wait 2 seconds immediately after a RESET ALL command, before issuing another command to the 576. Otherwise, an error may result on the next 576 command.

To eliminate the possibility of problems, follow any RESET ALL command with a 2 second delay. In BASICA or QuickBASIC, this is done as follows:

```
t! = TIMER: WHILE TIMER-t! <2: WEND
```

This does not apply to the RESET MEM or RESET OUT commands.

SUBR RETSUB ENDSUB

Create subroutine

Purpose

The SUBR command is used to create a program that can be called from the main program. These programs are similar to subroutines in other languages with the exception that these can not return any data or status. The ENDSUB command is the subroutine end flag and is used to mark the end of the program to be used as a subroutine. Subroutines can be executed using the CALL command or can be executed on an interrupt at a specified rate using the ONINT command. The RETSUB command allows the return from the subroutine to the calling program.

There can be up to ten user defined subroutines at one time in the system. Subroutine names may be up to eight characters in length and may use the characters 'A-Z', '0-9', '_', and '\$'.

Format

```
SUBR sub;  
RETSUB;  
ENDSUB;
```

Parameters

sub – the name of the subroutine to be created. The subroutine name may be up to eight characters in length and all characters are converted to upper case. Legal characters are 'A-Z', '0-9', '_', and '\$'.

Programming Notes

1. Subroutines can call other subroutines but they can not be nested (a subroutine cannot be defined within a subroutine).
2. See CALL command for more information on calling subroutines.
3. See ONINT command for more information on executing subroutines in the background.
4. Subroutine names longer than eight characters will be truncated to eight.
5. If RETSUB is issued in the active ONINT subroutine, execution will return to the calling routine until the next interrupt occurs.
6. There is a limit of 10 subroutines in the system at one time.

Programming Examples

```
SUBR TstB0Ful;           'Create subroutine TstB0Ful  
  IF B0, FU;            'Test if buffer 0 is FULL  
    BUFF READ DCV, B0;  'If B0 is full, send data to host  
  ENDIF;  
ENDSUB;                 ' End subroutine
```

```
Main Program  
BUFF DIM BYTE, B0, 3, 100; ' Create buffer 0 for 300 bytes  
DO;                        ' Loop forever  
  READ 5, 0-2, B0;        ' Read slot 5 DIO1A , ports 0-2 to B0  
  CALL TstB0Ful;         ' Test for buffer full  
LOOP;
```

Purpose The SYST command is used to configure and interrogate system features and functions. These system functions include the real time clock, the slot/module configuration, the IEEE data transfer parameters (EOI, terminator), the default data transfer format and engineering unit conversion, the system calibration, system peek and poke, the SRQ mask, the startup options, the time stamping setup, the system identification string, available memory, and the error string.

General Format SYST :function[,options];
SYST :function ?;

Parameters

:function – the system function to be configured or interrogated. The SYST command must always be followed by a function. If the function is followed by a “?”, the setup or value of the specified function will be returned. If the function is followed by data, the function will be configured with the specified data.

options – the function parameters, required and/or optional, for the specified function.

Functions

:AMM	Set up AMM system parameters
:BUF ?	Return buffer status information
:CAL	Setup system calibration
:CLOCK	Setup or Read the Real Time Clock
:EOI	Enable/Disable EOI flag
:ERR ?	Returns system error to controller
:FORM	Set system default data transfer format
:IDN ?	Return system identification string
:MEM ?	Return size of largest block of free RAM
:PEEK	Immediate read of 576 address location
:POKE	Immediate write to 576 address location
:SAVE	Setup system power up options
:SLOT	Configure slots with modules
:SRQ	Setup SRQ mask
:STAMP	Set up time stamping mode
:TERM	Set data transfer terminator sequence
:TRIG	Select start program execution trigger
:UNIT	Set up system default engineering units conversion

Programming Notes

1. All SYST commands are immediate.
2. The “:” used in front of the functions above is used as a command extender. A command extender allows the use of multiple functions declared in a single command. For example to configure a system:

```
SYST :AMM 100K :SLOT 3,AIM7 :UNIT TCJ,F;
```

This command will set up the system AMM filter for 100kHz, configure an AIM7 in slot 3, and set the default engineering units conversion for J type thermocouple and degrees F.

SYST :AMM

Set AMM global filter

Purpose The AMM function allows the user to set up the system A/D module filter.

Format **SYST :AMM filter;**
SYST :AMM ?;

Parameters

filter	- filter value to be programmed for AMM global filter.
100K	= 100 kHz filter
2K	= 2 kHz filter
?	- return system A/D module filter setting in the form: "AMM filter"

Default The default filter setting of the AMM1A and AMM2 is 100kHz (100K).

Programming Notes

1. All SYST commands are immediate.

SYST :BUF

Check buffer status

Purpose	The BUF? function allows the user to get the status, full or not full, of the buffers in the system.
Format	SYST :BUF ?;
Programming Notes	<p>This function returns the status of the system buffers in a 32 bit word. Each bit number corresponds to a buffer number. Bit 0 is buffer B0, bit 1 is buffer B1 and so on. Bits 20 through 31 are not used and will be set to 0.</p> <p>The status information of each bit is: 1 = BUFFER FULL 0 = BUFFER NOT FULL</p> <p>Bit 0 is always the rightmost bit (Least Significant Bit).</p>
Programming Notes	<ol style="list-style-type: none">1. All SYST commands are immediate.

Purpose AMM1A and AMM2 modules bear a calibration sticker carrying a 5-digit number. The SYST :CAL command provides for entry of this calibration constant to the Model 576. The calibration constant configures and enables the automatic gain and offset correction feature of the AMM1A and AMM2 global amplifier. This enables the AMM module to achieve the “corrected” specifications for analog measurements as published in the Model 576 manual. If the calibration factor is not entered, the “uncorrected” accuracy specifications will apply. The calibration factor also simplifies error budgets for measurements made with any analog input module in the option slot. With the calibration factor applied, the accuracy of these measurements will be the accuracy of the module itself. It will not be necessary to factor in the specifications of the AMM module.

Format SYST :CAL [<const>;
SYST :CAL;
SYST :CAL ?;

Parameters const – is the factory supplied calibration constant found on the AMM module. If not specified the 576 will generate the calibration coefficients for the AMM module.
? – will return the calibration constants in the form shown below

Programming Notes

1. All SYST commands are immediate.
2. The SYST :CAL command will not enhance the performance of AMM modules beyond specifications. The effect of SYST :CAL may not be dramatic or even noticeable if the AMM module is already operating within or close to specifications.
3. The gain and offset error correction will only be performed if the SYST :CAL function is issued with the calibration constant.
4. The command “SYST :CAL;” must be issued , without the calibration constant, to generate the correction coefficients used in the 576. This command should be issued only after the 576 has met the specified warm-up period.
5. To clear the gain and offset tables, issue the command RESET ALL;
6. If the Model 576 hasn’t been calibrated, only the “AMM CAL” portion of the string will be returned.
7. Data is returned in the format:
“AMM CAL, gain x1 UP, x1BP, x2 UP, x2BP, x5 UP, x5BP, x10 UP, x10BP,
Where
User the user-supplies a cal factor.
GAIN is the gain correction factor.
x1 UP is the x1 gain unipolar offset correction factor.
x1BP is the x1 gain bipolar offset correction factor.
x2 UP is the x2 gain unipolar offset correction factor.
x2BP is the x2 gain bipolar offset correction factor.
x5 UP is the x5 gain unipolar offset correction factor.
x5BP is the x5 gain bipolar offset correction factor.
x10 UP is the x10 gain unipolar offset correction factor.
x10BP is the x10 gain bipolar offset correction factor.

Programming Examples

```
SYST :CAL XXXX;           'Enter factory-supplied calibration constant
SYST :CAL;                'Issue calibrate command
SYST :CAL ?;              'Read cal. constant from 576
READ DATA
PRINT DATA
AMM CAL XXXXX 0.994322 -13.107200 6.553600
0.000000 0.000000 0.000000 0.019200
0.009000
```

SYST :CLOCK

Set, read, or disable clock

Purpose The CLOCK function allows the user to set and read the Real Time Clock. The Real Time Clock should be set before using any command which makes use of the clock.

Format **SYST :CLOCK [format,] [[date,][time]];**
SYST :CLOCK ?;
SYST :CLOCK DISABLE;

Parameters

- format
 - specifies the format of the date string.
 - STD = MM/DD/YYYY or MM-DD-YYYY (default)
 - EURO = DD.MM.YYYY
 - MIL = DD-*MMM*-YYYY
- date
 - values for date information
 - MM = month value between 1 to 12.
 - DD = day value between 1 and 31.
 - YYYY = year between 1990 and 2089
 - MMM = first characters of the month. (e.g. JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC)
- time
 - time is always in 24 hours format – HH:MM:SS
 - HH = hours 0 to 23
 - MM = minutes 0 to 59
 - SS = seconds 0 to 59
- ?
 - read the real time clock and return the data in the format: "date time", using the last set format or the default format.
- DISABLE
 - prolongs the life of the battery by disabling the oscillator for the real time clock. Clock is enabled by issuing command with time and/or date.

Default The default date format is STD, (MM/DD/YYYY)

Programming Notes

1. All SYST commands are immediate.
2. The real time clock will not be set unless the SYST :CLOCK command is issued.
3. See the READ command for more information on reading time and date to buffers.
4. See the SYST :STAMP and BUFF DIM commands for information on using the real time clock for timestamping.
5. See the IF, WHILE, and SYST :TRIG commands for more information about triggering control off of the real time clock.

Programming Examples

```
SYST :CLOCK, STD, 7/4/1990,0:0:0; 'set clock to midnight, July 4,1990
SYST :CLOCK ?; ' read date, time
READ DATA ' read 576 data
PRINT DATA ' display it to console
'07/04/1990,00:00:10' ' 576 returned string
```

SYST :EOI

Enable/disable EOI

Purpose The EOI (END or IDENTIFY) line on the GPIB bus is usually set low by a device during the last byte of its data transfer sequence. In this way the last byte is properly identified, allowing variable length data words to be transmitted. The 576 normally sends EOI during the last byte of its data string or status.

Format **SYST :EOI option;**
SYST :EOI ?;

Parameters

option	– specifies whether EOI is enabled or disabled
ENABLE	= enable the transmission of EOI
DISABLE	= disable the transmission of EOI
?	– returns the status of EOI usage in the form: “ENABLE” or “DISABLE”

Default EOI enabled at power-up

Programming Notes

1. All SYST commands are immediate.

SYST :ERR

Check error status

Purpose Returns system error in the form: "error string"

Format **SYST :ERR ?;**

SYST :ERR ? returns an ASCII string which gives a short explanation of the error.

Programming Notes

1. All SYST commands are immediate.
2. All system error messages are listed with explanation in the Appendix section.
3. See DEBUG command for information on using debugging tags.
4. If no errors are active, the string "No System Error" is returned.

SYST :FORM

Set data transmission format

Purpose The FORM function allows the user to specify the format of the data being transmitted across the bus. Data can be transmitted in ASCII or binary. If it is transmitted in ASCII, there are two sections that may be put together to make up each data value. First there is a prefix which specifies whether the data is normal or an overflow condition and what engineering units are represented by the data, the second is the data value.

Format **SYST :FORM format;**
SYST :FORM ?;

Parameters format – data transfer format specifier, default is ASCN.

The following table summarizes the data formats:

MOTO = Motorola binary format

INTL = Intel binary format

ASCI = ASCII with prefix

ASCN = ASCII without prefix

? – returns the current default system data transfer format in the form:
"FORMAT format"

Programming

1. All SYST commands are immediate.
2. Detailed information on data formats is included at the end of the command section (5).

SYST :IDN

Get system revision and ID

Purpose The IDN? function returns the system identification string in the form;
"Keithley Instruments Model 576, Rev x.xx".
where x.xx is the firmware revision level.

Format **SYST :IDN ?;**

Programming Notes

1. All SYST commands are immediate.

SYST :MEM

Report available memory

Purpose

The SYST :MEM? command returns the size of the largest free block of data memory and the available program memory. The memory size is returned in bytes, in the form <largest available block of data memory>,<available program memory>.

Format

SYST :MEM ?;

Programming Notes

1. All SYST commands are immediate.

Programming Examples

```
SYST :MEM ?;           'Ask for available memory
READ DATA
"MEMORY 102400,10240"
```

SYST :PEEK

Read system hardware data

Purpose The PEEK function allows the user to examine the contents of the 576 hardware, and places the data on the bus to be read immediately.

Format **SYST :PEEK slot, cmd;**
SYST :PEEK ID;
SYST :PEEK GLOBAL;
SYST :PEEK RESET;
SYST :PEEK STROBE;

Parameters

- slot – the slot number of the module to be read. See table below.
- cmd – the command to be used for the read. Valid commands are A, B, C, or D. See table below.
- ID – reads the contents at the self ID address location
- GLOBAL – reads the contents at the GLOBAL 1 location.
- RESET – reads the contents at the RESET location. (aka GLOBAL 2).
- STROBE – reads the contents at the GLOBAL STROBE location.

Slot	Function	Command
1	option slot 1	A B C D
2	TRG1	A B C
3	option slot 2	A B C
4	AOM5	A B
5	DIO1A	A B

Programming Notes

1. All SYST commands are immediate.
2. Peeking at invalid locations will generate an error.
3. CAUTION: Peeking at STROBE, GLOBAL, ID, and RESET will cause a strobe on the selected lines to all modules.
4. The SYST :PEEK function eliminates the need to setup and use buffered operations.

Programming Examples

```
SYST :PEEK 3,A;      ' PEEK on Command A of module in slot 3
...                  ' Wait for data ready, read data

SYST :PEEK 3,B;      ' PEEK on Command B of module in slot 3
...                  ' Wait for data ready, read data
```

SYST :POKE

Write data to system hardware

Purpose The POKE command allows the user to write a byte of data (0-255) to any accessible 576 hardware location.

Format **SYST :POKE slot, cmd, <val>;**
SYST :POKE ID, <val>;
SYST :POKE GLOBAL, <val>;
SYST :POKE RESET, <val>;
SYST :POKE STROBE, <val>;

Parameters

- slot – is the slot number of the module to be written. See table below.
- cmd – is the command to be used for the write. Valid commands are A, B, C, or D. See table below.
- ID – write data to the self ID address location.
- GLOBAL – write data to the GLOBAL 1 location.
- RESET – write data to the RESET location. (aka GLOBAL 2).
- STROBE – write data to the GLOBAL STROBE location.
- val – data to POKE, All data is evaluated MODULO 256.

Slot	Function	Command
1	option slot 1	A B C D
2	TRG1	A B C
3	option slot 2	A B C
4	AOM5	A B
5	DIO1A	A B

Programming Notes

1. All SYST commands are immediate.
2. Poking at invalid locations will generate an error.
3. CAUTION: Poking at STROBE, GLOBAL, ID, and RESET will cause a strobe on the selected line to all modules causing the desired action on all hardware which implements any features with the selected line.
4. The SYST :POKE function can be used in place of the WRITE command when there is limited amounts of data that need to be written. This function eliminates the need for setup and use of buffered operations.

SYST :SAVE

Set operating mode at power-up

Purpose The 576 includes a battery back-up feature for the program and system data memory. The battery back-up feature enables the 576 to retain data and/or program information when power is removed from the system. The SAVE function sets the system SAVE option.

Format **SYST :SAVE option;**
SYST :SAVE ?;

Parameters

- option – is the level of system configuration desired at power-up.
 - OFF = on power-up, the 576 executes self-ID for new hardware configuration, clears data buffers, and resets all outputs to 0. This is the default option.
 - CNFG = on power-up, the 576 uses previous hardware configuration, but clears data buffers and resets outputs to 0.
 - DATA = on power-up, the 576 retains previous hardware configuration and contents of data buffers, and resets all outputs to 0.
 - PROG = on power-up, the 576 retains previous hardware configuration, program, and contents of data buffers. Also resets data pointers to start of data buffers, resets outputs to 0, and restarts program (auto-restart mode).
- ? – Returns the current system save option in the form: “SAVE option”.

Switch 8 enables execution of a stored program when the 576 is powered up. Switch 7 disables re-execution of a 576 program after the program runs to completion.

Typical operation:

- a. Download program to 576.
- b. Set SYST :SAVE option to “PROG”.
- c. Turn off 576 and disconnect from bus.
- d. Transport 576 to test site.
- e. Apply power to 576. Program executes to completion.
- f. Turn off 576 and set switch 7 ON.
- g. Reconnect 576 to GPIB, turn on system, and retrieve data.

Programming Notes

1. All SYST commands are immediate.
2. Switch 8, on the motherboard of the 576, selects whether the SAVE function will be recognized. The default position for this switch (and the SAVE function) is OFF, or DISABLED. If a program issues the SAVE command and the feature is disabled, it will be ignored by the 576.
3. Switch 7, on the motherboard of the 576, disables program execution for SYST :SAVE PROG. (PROG option reverts to DATA option). The default position for switch 7 is OFF.
4. A Device Clear resets SYST :SAVE PROG to SYST :SAVE DATA.

SYST :SLOT

Assign module to slot 1 or 3

Purpose The SLOT function is used for module configuration of option slots 1 and 3 of the 576. It associates a module with a specified slot.

Format SYST :SLOT slot,module;
SYST :SLOT slot ?

Parameters slot – the slot location of the module being accessed.
module – one of the following: (default is EMPTY)

ANALOG INPUT	ANALOG OUTPUT	DIGITAL I/O	OTHER
AMM1A	AOM1/2	DIM1	PROTO*
AMM2	AOM1/5	DOM1	EMPTY
AIM2	AOM2/1	DIO1	
AIM3A	AOM2/2	DIO1A	
AIM4	AOM3	PCM1	
AIM5	AOM4	PIM1	
AIM6	AOM5	PIM2	
AIM7	WAV1*		
AIM8			
AIM9			
TRG1*			

*Identified, but not supported.

? Returns the module in the specified slot in the form: "SLOT slot, module"

Programming Notes

1. All SYST commands are immediate.
2. For modules which include the SELF-ID feature, the slot/module configuration is not necessary. The system will issue an error if the user tries to place a module in a slot which the self-id circuitry believes contains a valid module of a different type.
3. The module name 'EMPTY' is used to clear a slot entry.
4. Module IDs are determined upon power-up, or after a RESET ALL command.

Programming Examples

```
SYST :SLOT 1,AMM2;  
SYST :SLOT 3,AIM7;  
SYST :SLOT 3,DIO1A;  
SYST :SLOT 3,EMPTY;  
SYST :SLOT 3,PIM2;
```


SYST :SRQ

Set SRQ condition

Purpose The SRQ function allows the user to specify the condition(s) that will cause the 576 to generate a Service Request. The user selects one (or a combination delimited by commas) of the conditions shown below.

Format **SYST :SRQ cond;**
SYST :SRQ ?;

Parameters

cond	- an OR'ing of the conditions which will cause an SRQ: BUFF = Buffer Full DATA = Data Ready to be read from controller ERR = Error IDLE = 576 Not Busy, program complete NONE = No SRQ
?	- returns the current SRQ mask in the form: "SRQ cond [,cond...]"

Default NONE (do not use SRQ)

Programming Notes

1. All SYST commands are immediate.
2. When SYST :SRQ NONE; is issued, the 576 cannot issue a Service Request. NONE overrides all other mask conditions and if issued with other conditions an error will be issued. Data bits in the status poll byte are always set indicating the Service Request conditions.
3. Each time the SYST :SRQ command is written, the previous SRQ settings are overwritten.
4. If a Device Clear is issued to the 576, the SRQ mask is set to NONE.

Programming Examples

Example 1. SRQ on buffer full and data ready

SYST :SRQ BUFF,DATA;	' issue SRQ when any buffer is full or data ready
BUFF DIM B0,1,100;	'setup B0 for 1 channel of 100 points
READ 1,0,B0,FILL;	'read slot 1, channel 0 into buffer 0, fill buffer
X;	
WAIT ON SRQ	' wait for SRQ on buffer full then send buffer read
BUFF READ B0;	' setup for read of buffer 0 to controller
WAIT ON SRQ	' wait for SRQ on data ready to be read
READ BUFFER	' read data to controller

Example 2. SRQ on HALT

SYST :SRQ IDLE,BUFF;	'issue SRQ on halt or buffer full
BUFF DIM B0,1,100;	
DO;	
IF 1,2, GT, 3.5,DCV;	'if slot 1, channel 2 greater than 3.5 V
HALT;	'halt and issue SRQ
ELSE;	
READ 1,3,B0;	'else read slot 1, channel 3
ENDIF;	
LOOP; X;	
WAIT ON SRQ	
READ SERIAL POLL BYTE	
TAKE APPROPRIATE ACTION	

SYST :STAMP

Set time/date stamp mode

Purpose The TIME stamp function provides set up for the buffer time stamping facilities of the 576. With this command the 576 will add date and/or time stamping information to a buffer based on the mode and type settings.

Format SYST :STAMP mode, type;
SYST :STAMP ?;

Parameters

mode	- specifies how a buffer is to be stamped: ONCE = timestamp buffer on initial write only. SCAN = timestamp buffer on each scan.
type	- TIME = timestamp buffer with time only CLOCK = timestamp buffer with time and date
?	- return current time stamp settings in the form: "STAMP mode, type"

Default If SYST :STAMP is not issued, the default conditions are:
mode is set to ONCE per buffer, and **type** is set to TIME

Programming Notes

1. All SYST commands are immediate.
2. Acquisition rates above 100Hz will result in multiple scans of data with the same time stamp data.
3. Time stamping with hardware triggering active is a special case and the timestamp resolution is increased to 1 microsecond. Hardware triggering is used in conjunction with the READ.. QUICK command and will suspend any active rate specified in the ONINT command until it has run to completion. See the TRIG and READ...QUICK commands for more information on timestamping with hardware triggering.
4. See the BUFF DIM command for information on enabling the timestamp, and timestamping buffer requirements.

Programming Examples

```
SYST :CLOCK STD,2/1/1990,14:00:00; 'set clock 1/2/90, 2:00 PM
SYST :STAMP ONCE,TIME; 'stamp once/scan, time only
BUFF DIM B0,2,10, STAMP; 'enable timestamp
READ 1,1-2,B0,FILL; 'read 10 pts slot 0 chans 1 to 2
BUFF READ B0;
X;
```

```
WAIT FOR DATA READY
READ DATA
```

```
(sample output)
14:00:02.00000 'time stamp of initial reading
2.50 'channel 1's data (scan 1)
5.25 'channel 2's data
...(etc)...
```

SYST :TERM

Set terminator sequence

Purpose The TERM function allows the user to specify what terminator sequence is to be used.

Format SYST :TERM option;
SYST :TERM ?;

Parameter option – the type of terminator sequence to be used.
CRLF = carriage return, line feed (default)
LFCR = line feed, carriage return
CR = carriage return only
LF = line feed only
NONE = no terminator
? – returns the system terminator option in the form: "TERM option".

Default CRLF (Carriage return line feed).

Programming Notes 1. All SYST commands are immediate.

SYST :TRIG

Set start of program execution

Purpose	The TRIG function allows the user to choose how program execution begins.
Format	SYST :TRIG mode; SYST :TRIG ?;
Parameters	mode – is the condition that will start a program. EXEC = Begin program execution on X (Default). GET = Begin program execution on GET (Group Execute Trigger) [[date],[time]] = Begin program execution on specified date/time string. ? – returns current system trigger mode
Default	Begin program execution on receipt of “X” command.
Programming	<ol style="list-style-type: none">1. All SYST commands are immediate.2. This function determines when the next program received will begin execution. For execution. For example, to begin program execution on GET, the command “SYST :TRIG GET” must be executed before your program is executed. Then send down your application, execute it (another X), and issue the GET command from the host computer.3. The date must be specified in the same format as specified in the last issued SYST :CLOCK command or in the default format if SYST :CLOCK hasn’t been issued. For date and time formats refer to SYST :CLOCK function.4. It is not necessary to issue both the date and time but if both are issued the date must precede the time.5. The SYST :TRIG command is reset to EXEC upon receiving the command: “RESET ALL;”

SYST :UNIT

Set default EUF

Purpose Sets the system default engineering unit conversion flag for data being sent to or from the controller.

Format **SYST :UNIT unit;**
SYST :UNIT ?;

Parameters

unit	- sets the system default engineering unit conversion:
RAW	= Raw A/D counts, no conversion
DCV	= Volts
MA	= Milliamperes
TCn [C F]	= Thermocouple type, degrees C or degrees F, where n = J, K, S, T, E, B, R. Default degrees C
RTDn [C F]	= RTD type, degrees C or degrees F, where n = 85 or 92. Default is degrees C.
HZ	= Hertz
?	- returns the system default engineering unit flag in the form: "UNIT unit [vals]"

Default RAW (Raw data, no conversion)

Programming Notes

1. All SYST commands are immediate.
2. Only RAW, DCV, and MA are valid engineering units for input data conversions (i.e. BUFF WRITE). See BUFF WRITE command for more information.

Programming Examples

```
BUFF DIM TC,B0,4,1;
READ 3,0-3,B0;      'read thermocouple (AIM7) in slot 3, channel 0-3 into buffer 0.
SYST: UNIT,TCJ;    'eng. units conv. - deg C for J type TC
BUFF READ B0;      'read buffer 0 in Type J, degree C units
X;
WAIT FOR DATA READY
READ DATA
```

TEST

Self test

Purpose The TEST command causes the 576 to run through its internal diagnostic checks of RAM and ROM. If an error occurs in ROM or system RAM, the SRQ LED on the front panel will blink. A flash rate of two times per second implies a ROM error; 20 times per second implies a RAM error.

Format TEST;

Programming Notes A version of the self test is executed by the 576 on power up. If TEST fails and the SRQ LED does not blink, the 576 will not be able to respond correctly to any commands.

The 576 will be capable of operation only if the system RAM and ROM tests pass. If the tests fail, the SRQ LED will flash at the above specified rates. When the system RAM and ROM tests pass, the string: "System ROM and RAM – ok" will be returned.

CAUTION: ALL CONTENTS OF DATA AND PROGRAM MEMORY ARE DESTROYED WHEN TEST IS EXECUTED.

Programming Examples

```
TEST;  
X;           ' Execute self-test
```

Purpose The TRIG command is used to configure and enable hardware triggering in the 576 via the TRG1 in virtual slot 2. This command is using in conjunction with the analog input READ...QUICK command. NOTE: This command can only be used if there is an AMM1A or AMM2 in slot 1.

Format TRIG source,<threshold>,mode,coupling,level;

Parameters

source	- the source of the trigger signal. TRG1 = the local TRG1 input. AMM = the output of the global AMM amplifier. The channel is specified in the READ.. QUICK command.
<threshold>	- data value to set trigger level. Between +/-10V .
mode	- specifies trigger card operation: LATCH = Trigger in latched mode EVENT = Trigger in event mode

Of the above modes Latched can be used with multiple channel scans. Event mode is used with single channel scans only.

coupling	specifies signal coupling. DC = DC coupling AC = AC coupling
level	Specifies the trigger region. ABOVE = Trigger above the threshold BELOW = Trigger below the threshold

Ex. TRIG AMM,9.5,LATCH,AC,ABOVE;

Trigger on the specified AMM channel, using the latched mode with AC coupling and trigger when the signal is above the threshold value of 9.5 volts.

Programming Notes

1. If TRIG is executed while the trigger input signal is within the trigger region, nothing will happen. The trigger signal must enter the trigger region **after** the command is executed for triggering to occur.
2. If specifying multiple channels with the READ.. QUICK command the only valid trigger modes are LATCH. If multiple channels are to be scanned off of a hardware trigger it is recommended that the trigger signal be connected to the local input on the TRG1.
3. The 576 allows timestamping in conjunction with the hardware triggering. To setup timestamping on a hardware trigger, the SYST :STAMP command must also be configured and is dependant on the TRIG mode as follows:

TRIG mode	STAMP mode
LATCH, EVENT	ONCE SCAN

The buffer that will be used in the READ...QUICK command must then be dimensioned with the BUFF DIM command using the STAMP option to enable the desired timestamp. See the BUFF DIM command and the SYST :STAMP for more information on timestamping.

4. See the CHAN :FILTER function for information on setting the TRG1 filter. If no filter is specified, the TRG1 default is setup for a 1MHz filter.

Programming Examples

Example 1. Read until buffer full when voltage gets above .65V

```

BUFF DIM B0,5,10;
CHAN 2,0 :FILTER 100K ;           'Set TRG1 card to 100kHz filter
TRIG TRG1,.65,LATCH,AC,ABOVE;    'Setup TRG1 to use its local input, slot 2 channel 0.
                                  'Set mode to Latch, AC coupling, and Trigger when
                                  'signal is above .65 V
READ 1,1-6,B0,QUICK;            'Read quick slot 1, chans 1-6, 10 samples
X;                               'Execute program.

```

The above program will take 60 readings and place them into buffer 0 after the trigger condition is met.

Example 2. Hardware trigger with timestamping (to 1 microsecond resolution)

```

SYST :STAMP, SCAN, TIME;        'Set timestamp for once per trigger, time only
BUFF DIM B0,1,100,STAMP;       'Setup B0 for timestamping
CHAN 2,0 :FILTER 100K ;        'Set TRG1 card to 100kHz filter
TRIG TRG1,.65,LATCH,DC,ABOVE;  'Setup TRG1 to use its local input and trigger READ
                                  'to acquire data after the trigger signal is above
                                  '0.65 V
READ 3,0,B0,QUICK;            ' Read quick slot 3, channel 0, 100 data points
X;                               'Execute

```


WAIT

Time or 'GET' delay

Purpose The WAIT command is used to delay program execution the specified amount of time or until Group Execute Trigger (GET) is issued. The GET command may be used anywhere in a program to synchronize events to other devices on the IEEE bus.

Format WAIT <time>,unit;
WAIT GET;

Parameters

time	- is the amount of delay before program execution resumes. The range of time is: 1 - 65535, MSEC millisecond range, 1 - 3267, SEC seconds range, 1 - 65, MIN minutes range.
unit	- the time units associated with <time>: MSEC - milliseconds SEC - seconds MIN - minutes
GET	- the program will wait until a Group Execute Trigger (GET) is received.

Programming Examples

Example 1. Example of Timed WAIT

```
BUFF DIM B0, 1, 10;  
DO 10;                                'Loop 10 times  
    READ 1,0,B0,FILL;                 'Read 10 samples from channel 0, slot 1 to buffer B0  
    BUFF READ B0;                     'Read buffer B0  
    WAIT 2,SEC;                        'Wait 2 seconds  
LOOP;                                  'Loop end  
X;
```

Example 2. Example of Group Execute Trigger WAIT

```
BUFF DIM B0,1,10;  
DO;                                    ' Loop forever  
    WAIT GET;                          ' Wait for group execute trigger  
    READ 1,0,B0, FILL;                 ' Read slot 1 channel 0 into buffer 0  
    BUFF READ B0;                      ' Read buffer 0  
LOOP; ' End loop  
X;
```

The DO,LOOP in this example program will execute each time a Group Execute Trigger is issued.

WAVE

Controls WAVx output

Purpose Controls output of WAV module, e.g. turns WAV1 waveform ON or OFF, or produces a HAVER (pulse) waveform.

Format **WAVE slot, chan, mode;**

Parameters

slot	- slot in which module is mounted (3 = 576 option slot).
chan	- the channel being addressed.
mode	- module output: ON = turns on continuous waveform OFF = turns off waveform HAVER = outputs a haver waveform

Programming Notes

1. The WAVE command should be issued only after the WAV module has been configured with the CHAN :AMPL, :RANGE, :OFFS, :FUNC, :DUTY, and :FREQ commands have been issued. The CHAN commands may be issued in any order.
2. Any errors in the associated CHAN commands will not be reported until the program containing the WAVE command is executed.
3. A haver pulse is one complete cycle of the specified waveform, starting at minimum amplitude and returning to minimum.

WHILE WEND

WHILE Condition control

Purpose The WHILE command is used to create a section of a program that is executed only while the specified condition is met. The WEND command terminates the current level WHILE command, signaling the end of the statement to be executed while the while condition is true and where to begin processing when the condition goes false.

Format **WHILE slot,chan,[NOT,]cond,<val1> [[,<val2>], unit];**
WHILE {DATE|TIME},[NOT,] cond, <val1>;
WHILE Bn,[NOT,]FU;
WEND;

Parameters

slot	– the slot number of the module being tested.
chan	– number of the channel being tested.
NOT	– optional specifier to test for condition being NOT TRUE.
cond	– expression which must evaluate TRUE. (See Expression Table for all available condition operators)
val1	– value typically used in an expression when a single limit or low threshold value is required. The evaluation of val1's format is specified by units. If units is not specified the current SYST :UNIT default setting is used.
val 2	– optional value typically used in an expression as an upper limit or high threshold value. The evaluation of val2's format is specified by units. If units is not specified the current SYST :UNIT setting is used.
unit	– Optional engineering unit conversion specifier. The lower and upper threshold values, val1 and val2, will be used in the 'unit' specified. If unit is not specified, the default unit set in SYST :UNIT will be used. See SYST: UNIT for setting default engineering units.
Bn	– is the number of the buffer to be tested: B0-B19. Test is for buffer full (FU) or not full (NOT FU).

Programming Notes

1. Every WHILE must have a matching WEND or a run time error will be issued upon receiving X (execute).
2. You can not specify both the time and date in one WHILE command. To do conditional triggering on both date and time use two nested WHILE statements.
3. Date and time must be specified in the same mode that was used to set the real time clock. See SYST :CLOCK for more information.
4. WHILE's can be nested 8 levels deep.
5. With Bit Test Operators the val1 parameter specifies the bits to be tested.
6. Nested WHILEs must be used to simulate BETWEEN and NOT BETWEEN conditions for TIME and DATE.

Expression Table

Operator	Meaning	Val1 Required	Val2 Required
LT	Less Than	Yes	No
GT	Greater Than	Yes	No
EQ	Equal To	Yes	No
LE	Less Than or Equal To	Yes	No
GE	Greater Than or Equal To	Yes	No
FU	Buffer Full	No	No
BT	Between	Yes	Yes
OR	Bit Test – Any bits set in val	Yes	No
AND	Bit Test – All bits set in val	Yes	No

Programming Examples

Example 1. Generic example to show leveling

```

WHILE ...;           'While level 1
  WHILE...;         'While level 2
    WHILE...;       'While level 3
      WHILE...;     'While level 4
        WEND;        'End level 4
      WEND;          'End level 3
    WEND;            'End level 2
  WEND;              'End level 1
  
```

Example 2. While condition on buffer not full

```

WHILE B0,NOT, FU;   'while buffer B0 is Not Full
  READ1,0,B0;       'read another point
WEND ;
  
```

Example 3. While condition on input

```

SYST :UNIT RAW;     'set raw mode
BUFF DIM B0,1,1000;
DO;                 'loop forever
  WHILE 1,0,GT,32767; 'wait for slot 1 channel 0 > 32767
  WEND;              'do nothing until cond met
  READ 1,0,B0,FILL; 'read 1000 points
  BUFF READ B0;
LOOP;
X;
  
```

WRITE

Write data

Purpose The WRITE command is used for writing data from any buffer to an output signal conditioning module. The command will write data from the specified data buffer to the start channel through the stop channel of the specified slot. There is also an optional cycle parameter which can specify the number of complete cycles through the entire specified buffer.

Format WRITE slot,chans, Bn[,CYCLES];

Parameters

slot	- the slot number of the module to be written
chans	- the channel or range of channels to be written. A range of channels is specified as 'start chan-stop chan' (eg. '2-4').
Bn	- the number of the buffer to be accessed: B0-B19.
CYCLES	- the number of cycles through the specified buffer to perform, if not specified one scan is written. Maximum cycles is 65535. See Note 4.

Programming Notes

1. See the BUFF DIM and BUFF WRITE commands for information on buffer setup and allocation.
2. If the specified buffer does not exist an error will be generated.
3. When using the WRITE command to output data from a buffer and more samples are specified than the amount of memory allocated in the buffer, the buffer address will recycle through the buffer from the first point. This allows continuous waveforms to be output with a single WRITE command.
4. The maximum number for CYCLES may be less than 65535. The maximum number of cycles is $2^{32} / \text{\#scans}$. If the specified CYCLES number is too large the 576 will generate an error.
5. When writing a negative voltage as a raw value to an AOM5 or slot 4, calculate the raw value as the required data bits + 32768.

Programming Examples

```
Generate sine wave buffer of 100 points - Sine% (100)
Generate square wave buffer of 100 points - Squar% (100)
BUFF DIM B0,2,100;           ' Set up buffer 0 , 2 channels , 100 data samples
BUFF WRITE DCV,2, 0-1,B0    ' Write data to buffer 0. Data format = ASCII, no prefix.
                              ' Note : 200 data points are now expected by the 576

FOR I =0 to 98
  CMD$ = STR$(SINE%(i)) + "," + STR$(SQUARE%(i) + ","
:
  OUTPUT CMD$                ' Code to send CMD$ to 576
NEXT I
  CMD$ = STR$(SINE%(99)) + "," + STR$(SQUARE%(99))+ ","
:
  OUTPUT CMD$                ' Code to send CMD$ to 576
DO 10000;                    ' Loop 10000 times,100 cycles of 100 point sine
WRITE 2,0-1,B0;              ' Output to slot 2, channels 0 and 1
LOOP; X;                      ' End loop
```

Note: The above DO, LOOP could have been written using the CYCLE option:
WRITE 2,0-1,B0,100; '100 cycles of 100 point sinewave
X;

X

Execute

Purpose The X command causes the 576 to execute all commands that are in program memory.

Format X;

- Programming**
1. If the program is set to execute on a Group Execute Trigger (GET) or on the time and/or date the execute command (X) must still be issued as normal and will only arm the program to begin execution at the appropriate event.

See SYST :TRIG for more information about triggering the start of program execution from Group Execute Trigger (GET) or from the time and/or date.
 2. To stop the execution of any program, use 'HALT', 'HALT SRQ', or issue 'DEVICE CLEAR '.

Purpose

The purpose of ZCAL is to determine a calibration constant relating to the characteristics of the AMM zener voltage reference. This constant is similar to the one affixed to the AMM module, and is used with the SYST :CAL command to create gain and offset correction factors for the module. (also see SYST :CAL).

ZCAL may be used as part of a typical standard calibration cycle for the AMM module, or any time a repair to the AMM module is performed (especially if the zener diode voltage reference has been replaced). The long-term stability of the zener reference is quite high, so frequent use of ZCAL is generally unnecessary.

Format

ZCAL <val>;

Parameters

val - the calibrator output voltage that is fed into the AMM module.

Programming

1. <val> must be in the range of 6.0 to 8.0 volts. All other values are illegal except for 0 volts which is used to clear the zener calibration constant.
2. ZCAL is an immediate mode command.
3. As part of ZCAL, the 576 automatically executes the SYST :CAL command using the calibration constant generated by ZCAL. It will be unnecessary to run SYST :CAL during this procedure.
4. ZCAL data is returned in the format:

```
"AMM CAL const,GAIN,x1UP,x1BP,x2UP,x2BP,x5UP,x5BP,x10UP,x10BP"
```

where:

const - calibration constant determined by ZCAL
GAIN - gain correction factor
x1UPP - x1 gain unipolar offset correction factor
x1BP - x1 gain bipolar offset correction factor
x2UP - x2 gain unipolar offset correction factor
x2BP - x2 gain bipolar offset correction factor
x5UP - x5 gain unipolar offset correction factor
x5BP - x5 gain bipolar offset correction factor
x10UP - x10 gain unipolar offset correction factor
x10BP - x10 gain bipolar offset correction factor

5. The 576 maintains a reserved area of RAM for the storage of the zener calibration constant. The 576 first validates the value stored in this location. If the stored value is legal, the 576 uses it as the system calibration constant.

It is unnecessary to issue a SYST :CAL <val>; or SYST :CAL; command, or otherwise manually transfer the calibration constant to SYST :CAL when using ZCAL. These func-

tions are handled automatically by ZCAL. However, ZCAL will only save the new zener calibration constant if switch 6 of the 576 address switch is set ON. Otherwise, the effects of ZCAL are temporary (i.e. they will be in effect until a power down or RESET ALL). You should record the calibration constant for future use with the SYST :CAL command.

NOTE: The zener calibration constant is different than the system calibration constant. The system calibration constant is temporary. The zener calibration constant is stored in battery backed up RAM and will not be altered by power down or RESET ALL;

Use of “Immediate” vs “Program” Commands

The choice of when to use Immediate versus Program mode commands may not seem obvious to first-time users of the Model 576, or even to seasoned GPIB users. The following information will help you understand and make proper use of the command types.

An important difference in immediate and program commands concerns how these commands affect the various types of memory. The 576 contains three types of memory:

1. System memory, where system configuration and data memory allocation information are stored,
2. Code generator memory, where programs are stored, less those system- or memory-related commands which have been stored in system memory, and
3. Data memory, where data from BUFF READ, BUFF WRITE, READ, or WRITE are stored.

Immediate Commands

Immediate commands cause the Model 576 to behave more like a traditional GPIB instrument, and provide the easiest means of using the system. Immediate commands execute at once when they are received by the Model 576, and can be used in an interactive fashion to configure the system, and to input or output data one value at a time.

The following commands are immediate:

BUFF DIM and BUFF WRITE
IREAD and IWRITE
All SYST commands
All CHAN commands

Immediate commands do not make use of any 576 code generator memory. Configuration changes brought about through BUFF, SYST and CHAN commands are stored in the appropriate memory area. Specifically, the BUFF DIM, BUFF WRITE, system (SYST) and channel (CHAN) configuration commands cause the designated memory or hardware changes to be made as soon as the commands are received by the 576.

As an example, an immediate data read (IREAD) is processed as soon as it is received by the 576, and instantly

places data in the system output queue where it can be read back by the computer. Similarly, an immediate data write (IWRITE) outputs data directly to the selected channel without use of code generator or data memory areas. Neither command requires an “X” instruction to execute. Some aspects of the data, such as data format or engineering units conversion, are determined by information stored in the system memory. This may be default information, or new information written via the SYST and CHAN commands.

Program Commands

Program mode commands facilitate more intelligent operation for the 576, including subroutines, conditional triggering, looping, automatic program restart, and other features beyond the capabilities of most GPIB instruments or immediate mode commands. Program mode makes it possible to store and execute programs totally within the 576, effectively turning the 576 into a stand-alone data logger. All 576 commands not listed above as immediate mode commands are program mode commands.

Typically, program mode commands are used where a program must perform any of the following:

1. Run in a stand-alone fashion, without having the computer control the operation of the desired test,
2. Acquire data and store it in 576 data memory,
3. Output simple or complex analog or digital waveforms in real time,
4. Make use of special firmware features such as data averaging, timed acquisition, time-stamping, looping, conditional testing, subroutines, triggering, etc.
5. Acquire data at the fastest possible speeds. Many of the capabilities of program mode commands can be simulated using immediate commands, but the overall throughput of the system will be lessened due to speed limitations of the computer, program operations, and bus transfers.

These functions will all operate totally *within* the Model 576, freeing the computer once the program has been sent to the 576 and executed. The program itself will include appropriate program mode commands, plus selected immediate mode commands (BUFF DIM, BUFF WRITE, SYST, and CHAN), and may ultimately use any of the 576 memory areas.

Also note that a program may use different channel setups before various READS and WRITES throughout a

program. For example, slot 1 channel 0 (an AMM module) may be programmed for a desired gain using the CHAN:GAIN command, and a READ command used to input data. Later in the program, the same channel may be reprogrammed with a different gain, and another READ command issued. This feature permits different channel configurations to be used at different points in a program for maximum versatility.

When a program is loaded into the 576 from the computer, it is parsed by the 576. Any immediate commands will take affect as soon as they are received by the 576. These changes are parsed and written to the system memory area to provide the desired memory, system, and channel configurations. Any data uploaded to a 576 buffer (for output through analog or digital output channels) is written to the data memory area.

The program-mode commands are likewise parsed, but stored in the code generator memory. These commands will execute only after an Execute (X) instruction is encountered within the program or explicitly sent to the 576 at a later time.

NOTE

IREAD and IWRITE cannot be embedded in constructs based on program-mode commands, and should generally not be intermixed with program-mode commands. An attempt to use IREAD or IWRITE within a SUBR (subroutine), IF..ELSE..ENDIF, DO..LOOP, or WHILE..WEND loop will result in errors.

Refer to the Example Disk programs and the Applications section of the Model 576 manual for examples of Immediate and Program modes.

Internal Data Buffer Description

The following information details the internal layout of the 576 data buffers, i.e. how a BUFF DIM command is interpreted by the system. For this discussion, SCANS represents the number of samples allocated per channel. The general form of a 576 buffer is as follows:

```
SCAN #1 [chan 1][chan 2]...[chan n]
SCAN #2 [chan 1][chan 2]...[chan n]
:
:
```

```
:
:
SCAN #N [chan 1][chan 2]...[chan n]
```

In the above layout, "N" represents the number of samples or scans, and "n" is the number of channels that were used in a BUFF DIM command. The bracketed information, [chan ...], represents data values returned from the indicated channel. Each value will be either 1, 2, or 4 bytes of data depending upon the data type (BYTE, WORD, LONG, TC), and is always stored in the buffer in the format of the I/O device being read or written. The 576 will make one adjustment to the above format if the buffer has been dimensioned for temperature conversions using the TC data type. In this case, the 576 will add a channel for reading the cold junction (CJ) reference point to every scan:

```
SCAN #1 [chan 1][chan 2]...[chan n][CJ]
SCAN #2 [chan 1][chan 2]...[chan n][CJ]
:
:
SCAN #N [chan 1][chan 2]...[chan n][CJ]
```

Note, however, that the Cold Junction data is never returned when applying temperature conversions to the buffer data, and can only be read from the buffer if RAW or DCV conversion is specified.

If time stamping has been enabled for the buffer, then the internal format for the buffer will change depending upon the time stamping mode that has been selected with the SYST :STAMP command. Time stamping requires that an additional 6 bytes per stamp be allocated in the buffer. The following layout shows the buffer format when the buffer has been dimensioned with Time Stamping enabled and the time stamping mode set to ONCE.

```
[ TS information - 6 bytes ]
SCAN #1 [chan 1][chan 2]...[chan n]
SCAN #2 [chan 1][chan 2]...[chan n]
:
:
SCAN #N [chan 1][chan 2]...[chan n]
```

If time stamping has been enabled, and the Mode is set to SCAN:

```
SCAN #1 [ TS information - 6 bytes ][chan 1]
[chan 2]...[chan n]
SCAN #2 [ TS information - 6 bytes ][chan 1]
[chan 2]...[chan n]
:
:
```

```

:
:
SCAN #N [ TS information – 6 bytes ][chan 1]
      [chan 2]...[chan n]

```

Note that in SCAN mode, the 576 will allocate 6 additional bytes for each SCAN of data.

Time Stamp information is stored internally in 576 buffers in one of two formats, depending on whether the Real Time Clock (RTC) was read, or the Time Stamp was acquired as part of a Hardware Trigger (HT) operation (see TRIG command). If the time stamp information was obtained from the RTC, then the 6 bytes of data represent the time and date read from the DS-1216D SmartWatch, and are stored in the format received from the device:

[Byte 0]	represents Hours 0-23
[Byte 1]	represents Minutes 0-59
[Byte 2]	represents Seconds 0-59
[Byte 3]	represents Hundreds of Seconds [0-99]
[Byte 4]	represents the Month 1-12
[Byte 5]	represents the Date 1 -31

If the buffer data is returned in BINARY, the RTC time stamp information is returned in packed BCD format in the sequence of bytes listed above. The 576 will return time stamp information in ASCII when the ASCII transmission formats are used.

If the time stamp was the result of a Hardware Trigger (HT) operation, then the six bytes of data represent the number of microseconds that elapsed from the time when trigger module in SLOT 2 was enabled to the time when the trigger condition was met. In this mode, the 576 utilizes the 6 bytes reserved for time stamping as a 48 bit integer. This allows a maximum elapsed trigger time of 2^{48} microseconds or roughly 8.9 years. Hardware triggered time stamping is achieved by cascading all three timers found on the 6840 PTM configured in a count down mode. The results stored in the buffer must be subtracted from 2^{48} to obtain the actual results. If the buffer data is returned in BINARY mode, this correction will not be performed by the 576. In the ASCII transmission modes the 576 returns the adjusted HT time stamp data. Note that ONCE is the only valid time stamp mode when using time stamping with a hardware trigger.

Data Transfer Modes

The following information describes the data streams returned by the 576 for all of the data transfer modes available with the SYST :FORM command. The 576 can be configured to return the contents of its internal data buffers in two ASCII or two Binary modes. The returned data is always converted to the selected engineering units (configured with the SYST :UNITS command). Note that the transfer mode selected by the SYST :FORM command only effects the transfer of buffered or immediate data. Configuration, status, or error messages are always returned as ASCII text.

ASCII Transmission Modes: ASCII with Prefix and ASCII No Prefix

In these modes, all data returned by the 576 is converted to records of fixed length fields and ASCII text before being transmitted. In ASCII transfers, all fields within a scan are separated by commas, and all scans of data are separated by the user-specified terminator sequence (see SYST :TERM command). If the SYST :TERM is set to NONE, then it is left to the user to parse the scans of data based on rules for fixed field parsing. If the IEEE-488 End or Identify (EOI) bit has been enabled, then the EOI will be signaled to the controller on the LAST byte of data sent from the 576.

The general format for ASCII transfers is:

```

[System Time][TERMS]
Scan #1 [Header]<Buffer Data>[TERMS]
Scan #2 [Header]<Buffer Data>[TERMS]
:
:
:
Scan #n [Header]<Buffer Data>[TERMS]

```

This transfer data is made up of an optional System Time, followed by an optional header, followed by the buffer data. The System Time is the starting time and date of program execution and is only returned if the buffer data being transmitted contains time stamping information. The System Time is transmitted only once on a per buffer basis, and the format of the returned data follows what was set with the SYST :TIME command. When transmitted, this field will be 20 characters in Military format, or 19 characters for Standard or European formats. The header contains information pertinent to the data which follows (explain in further detail below) and is only transmitted if time stamping or ASCII with Prefix format has been specified. Finally, the buffer data is returned and converted to the engineering units specified via the SYST :UNIT command.

The optional header record can be further broken down into the following fields:

[Date,][Time,][Prefix,]

The optional Date field is returned if time stamp type specifier was set to CLOCK. This option returns the Date and Month that the current data was acquired. This field is 6 bytes in length for Military format and 5 for Standard or European. This field is not returned with Hardware Trigger (HT) time stamping.

The optional time field is returned regardless of the time stamping mode set and is either the time of day (24 hour format) to the hundredths of seconds, or the elapsed number of microseconds for hardware triggered (HT) time stamps. This field is 13 characters if HT time stamping information is being returned, or 11 characters if Real-Time Clock (RTC) time stamping was enabled.

Note that time stamp information is returned based on the time stamping mode that was set when the buffer was dimensioned. If the mode was set to ONCE then time stamp information is only sent along with the very first scan of data. If the mode was set to SCAN, then each scan of data sent will have the appropriate time stamp information sent as part of the header.

The Prefix field is returned only with ASCII prefix data transfers, and describes the type of engineering units conversions applied to the buffer data. The prefix field is always 4 characters in length. Valid prefix strings are:

NRAW	– raw data (i.e. A/D or D/A counts)
NDCV	– DC volts conversions
NDCA	– DC amperes
DEGC	– degrees Celsius
DEGF	– degrees Fahrenheit
FREQ	– frequency (in Hertz)

Buffer Data follows the optional header information, and is returned according to the engineering units conversion specified with the SYST :UNIT command. When engineering units conversion is applied to buffer data, the data is returned as a floating-point value:

SD.DDDDDDE DDD (14 characters)

where S is the sign bit for negative values (–), or a space character for positive values. When the engineering unit is RAW the format of the data returned will either be 3, 5, or 10 bytes depending on the data format of the device that was read or written. RAW data is always returned as an unsigned value in the ranges of:

0–255	– byte data (3 bytes – digital data)
0–65535	– word data (5 bytes – analog in/out data)
0–4292967295	– long data (10 bytes – PIM2 data)

Binary Transfer Modes: Motorola and Intel Data Formats

In Motorola and Intel transfer modes, the data is returned to the controller in binary form. The 576 will still apply all engineering units conversions (specified via the SYST :UNIT command) to buffered data prior to transmitting it to the controller. All data returned as a result of an engineering units conversion (except RAW mode) will be returned as a single-precision IEEE floating-point number (4 bytes in length). Raw data will be returned as a 1, 2, or 4 byte value. There are no delimiters sent to separate channels or scans for binary transfers. Other than byte data, data which is returned to the host in Motorola format is always returned as HIGH BYTE, LOW BYTE format. If the mode is Intel then data is returned to the controller as LOW BYTE, HIGH BYTE. Byte data is returned to the controller the same for both modes, as a sequence of single bytes. For example, the digits 1234 would be transmitted as 1234 in Motorola format and 4321 in Intel format. The correct binary transfer mode depends on the format used by the host CPU. If the IEEE-488 End or Identify (EOI) bit has been enabled, then the EOI will be signaled to the controller on the LAST byte of data sent from the 576.

All buffer data transmitted to the host is returned with a 6 byte header. This header contains information about the number of channels, type of data in the buffer, if time stamping was enabled, and the amount of data (in bytes) that is to be transmitted from the buffer. The format of the binary header is shown below:

[Data Configuration byte 1] [Data Configuration byte 2] [size (4 bytes)]

[Data Configuration Byte 1]:

TTCCCC

where:

TT = size of and type of data being transmitted.

- 00 BYTE data (1 byte per channel)
- 01 WORD data (2 bytes per channel)
- 10 LONG data (4 bytes per channel)
- 11 FLOAT data (4 bytes per channel)

CCCCC = channels. These bits represent either the channel number or the number of channels contained in the buffer depending upon whether a buffer or a channel read was performed on the buffer (See bit 6 of the configuration byte 2).

[Data Configuration Byte 2]:

XCSFMMTT

where:

- X = not used; always set to zero.
- C = channel/buffer read bit.

- 1 - channel read operation performed. Channel number is represented by the bits in configuration byte 1
- 0 - buffer read operation performed. The number of channels in the buffer are represented by the channel bits in configuration byte 1.

S = buffer statistics/buffer data.

- 1 - buffer statistics being returned (See BUFF STAT command)
- 0 - normal buffer data returned. (See BUFF READ command)

F = format bit

- 1 - Intel transfer mode selected
- 0 - Motorola transfer mode selected

MM = time stamp mode

- 00 - unused
- 01 - ONCE
- 10 - SCAN
- 11 - unused

TT = time stamp type

- 00 - NONE (disabled)
- 01 - Real Time Clock (RTC) stamp data.
- 10 - Hardware Triggered (HT) time stamp data.
- 11 - unused

[Size]:

Total number of bytes to follow. Note that this field is transmitted to the host according to the transfer mode set.

After the header has been transmitted, the remaining buffer data is transferred to the host in a format similar to the ASCN transfer mode. The general form is:

Binary header
[System Time]
[Time Stamp]<Buffer Data>

As in the ASCII transfer modes, the system time is transmitted only if time stamping information is stored in the buffer. In binary transfer modes, however, this data is 6 bytes of packed BCD data transmitted as:

[HRS][MINS][SECS][YY][MM][DD]

BCD data is always transmitted as byte data. The optional time stamp information is returned as packed BCD data if the Real Time Clock (RTC) was read, or as a 48-bit integer value if a hardware-triggered (HT) time stamp was done. In a case where the Real Time Clock (RTC) was read, the data is returned as:

[HRS][MINS][SECS][HSECS][MM][DD]

RTC data is returned as byte data and is therefore unaffected by the transfer format.

If the time stamp was the result of an HT operation, then the six bytes of data represent the number of microseconds that elapsed from when the trigger module in SLOT 2 was enabled to the time when the trigger condition was met. In this mode, the 576 utilizes the 6 bytes reserved for time stamping as a 48-bit integer. This allows a maximum elapsed trigger time of 2^{48} microseconds or roughly 8.9 years. HT time stamping is achieved by cascading all three timers found on the 6840 PTM configured in a countdown mode. The results stored in the buffer must be subtracted from 2^{48} to obtain the actual results. When the buffer data is returned in binary mode, this correction is not performed by the 576. The HT time stamp information is returned in the format specified by SYST :FORM.

Following the header information, buffer data is returned. Buffer data is always sent based on the transfer mode in effect. This is true for all data except byte data

which is transmitted as a series of single bytes. FLOAT data is returned in IEEE single-precision floating-point format (4 bytes), LONG data is returned as a 4 byte integer, WORD data is a 2 byte integer and byte data is returned a single byte stream. FLOAT, LONG, and WORD data are transmitted from the 576 in the transfer format set by SYST :FORM.

Device Status (SPOLL) Byte

The 576 Device Status Byte ("DSB", or "SPOLL" Byte) is an indicator of the current 576 status. Each bit within the DSB has special meaning. The status of a bit can change dynamically, or it can be latched via the SYST :SRQ command. The following describes how each bit can be set and also what conditions are required to clear it.

Bits within the 576 Device Status Byte are defined as follows:

BIT	FUNCTION	VALUE
Bit 0 (LSB)	Debug number, bit 0	1
Bit 1	Debug number, bit 1	2
Bit 2	Debug number, bit 2	4
Bit 3	BUFFER FULL bit	8
Bit 4	DATA READY bit	16
Bit 5	ERROR bit	32
Bit 6	RQS bit	64
Bit 7 (MSB)	IDLE bit	128

Logic is high-true, i.e. a condition exists if the corresponding bit equals "1".

The description of each bit follows:

IDLE — indicates when the 576 is executing a program. It will be cleared when a program is executing and set otherwise. When this bit goes from 0 to 1, it indicates that the program has completed.

RQS (ReQest for Service) — indicates that the 576 has generated a service request. The 576 can only generate service requests if the SYST :SRQ command has been issued. By default, the 576 will never request service. This bit is cleared via Serial Poll.

ERROR — indicates that some kind of error occurred in the 576. It could be a parse-time or a run-time error. This bit is cleared only when the SYST :ERR ? command is exe-

cuted. Refer to the Appendix section for more information on errors.

DATA RDY — indicates that data is available in the 576 output queue. It is set whenever any data is available to be read and is cleared when the 576 output queue goes empty. This is a generic bit that merely indicates the presence of data.

BUFF FULL — indicates that at least one of the 20 internal data buffers has been filled. It is cleared when all 20 buffers are empty. To determine which specific buffer is full, issue the SYST :BUF ? command.

DEBUG — three DEBUG bits reflect the current setting of the debug code. The debug code is set using the DEBUG command. These bits are always set to 0 by a Device Clear.

The command SYST :SRQ can be used to force the 576 to request service when a specific condition arises. The IDLE, ERROR, DATA RDY, and BUFF FULL conditions can all generate a service request. Any condition that causes a service request is automatically latched. The condition becomes unlatched when a Serial Poll is performed.

Serial Poll Bit Test

The value of any individual bit can be derived by ANDing the value of the Serial Poll Byte (sp%) with the value of the desired bit. If the result is 0, then the value of the tested bit is also 0. If the bit test results in a number other than 0, the value of the tested bit is 1.

Debug Number

Bits 0, 1, and 2 together represent the binary value of last debug number that was encountered in the program. Up to eight debug numbers (0-7) can be set in a 576 program using the DEBUG command. When any given DEBUG command is executed, the associated debug number will be assigned to bits 0-3 of the Serial Poll Byte. Typically, the debug number is used for checking the last code which was executed in a program.

The following shows how the bits may be tested using BASIC or QuickBASIC. The process would be similar in other programming languages.

Bit Test Operation for sp%	Bit Function
b0 = sp% AND 1	If b0 <> 0 then b0 = 1 (Debug number - bit 0)
b1 = sp% AND 2	If b1 <> 0 then b1 = 1 (Debug number - bit 1)
b2 = sp% AND 4	If b2 <> 0 then b2 = 1 (Debug number - bit 2)
b3 = sp% AND 8	If b3 <> 0 then b3 = 1 (BUFFER FULL bit)
b4 = sp% AND 16	If b4 <> 0 then b4 = 1 (DATA READY bit)
b5 = sp% AND 32	If b5 <> 0 then b5 = 1 (ERROR bit)
b6 = sp% AND 64	If b6 <> 0 then b6 = 1 (SRQ bit)
b7 = sp% AND 128	If b7 <> 0 then b7 = 1 (IDLE bit)

Last encountered DEBUG No. = $b0 \times 1 + b1 \times 2 + b2 \times 4$

Example:

A read of the Serial Poll Byte returns 37. Bits b0, b2 and b5 test TRUE ($1 + 4 + 32 = 37$), indicating that:

- a) an error has occurred, and
- b) the last DEBUG number was 5.

SECTION 6

Applications

INTRODUCTION

The Model 576 Applications section shows how to implement several typical test, measurement, and control applications using the Keithley Model 576 and selected personal computers. These applications include diagrams and example listings of the commands necessary to implement the programs. The commands can be used with any programming language and GPIB driver software.

The Model 576 Example Disk includes more complete versions of most of these applications. These are written for IBMPC/XT/AT and compatibles using QuickBASIC, Driver488 software, and the Keithley "UTIL" library (DEMO576.SAV is an Asystant GPIB program). Applications cover the following topics:

1. Battery Life Testing (BATTLIFE.BAS)
2. Lab Acquisition with the Model 576 and Asystant GPIB (DEMO576.SAV)
3. Multi-Channel Voltage Measurements with Hardware Triggering (HWTRIGGER.BAS)
4. Process Monitoring: Triggering and Time Stamping (PROCMON.BAS)
5. Portable and Remote Acquisition (REMOTEDN.BAS and REMOTEUP.BAS)
6. Testing 8-Bit DACs with the Model 576 (DAC-TEST.BAS)
7. Radiometry Using the Model 576 with a Model 428 Current Amplifier (GPIBTEST.BAS)
8. Data Logging with the Apple Macintosh (example in text)

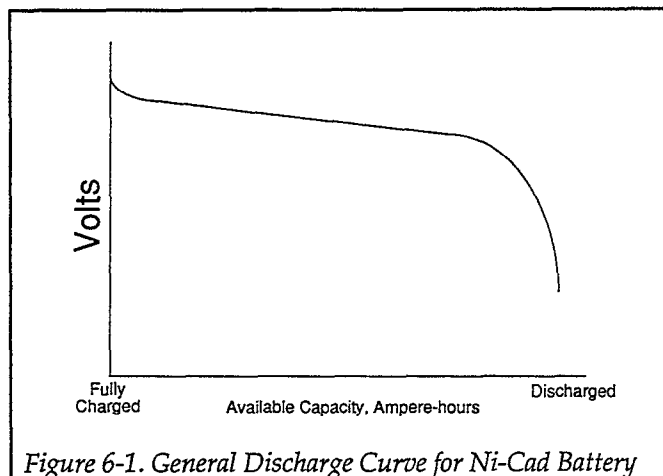
Before you attempt to run any of these applications programs, you should become familiar with the Model 576 hardware and command set. Also refer to the Model 576 "Quick Start" instructions and example programs which will assist you in setting up and running the Model 576. Refer to the Model 576 manual for detailed set-up and operating instructions.

Note: In the following program examples, italics indicate operations which must be written in the programming language actually being used. Some constructs, such as FOR-NEXT loops, are BASIC syntax and must be converted to the language in use. 576 commands are capitalized. Comments are preceded by an apostrophe (').

BATTERY LIFE TESTING

The basic function of a battery is to store energy so that it may be used to perform work conveniently, as need arises. Output voltage and current capacity are the main criteria which determine the suitability of a cell to a given task. However, cell voltage vs current capacity, shelf-life, and ability to deal with varying loads are also important. The relationship of voltage, current, and time is referred to as the "discharge curve" for the cell.

A typical discharge curve for a nickel-cadmium rechargeable cell is shown in Figure 6-1. The curve shows that the discharge rate is relatively constant until the cell is nearly depleted, at which time the voltage drops off sharply.



Factors Affecting Battery Discharge

A number of variables affect cell discharge characteristics. These include:

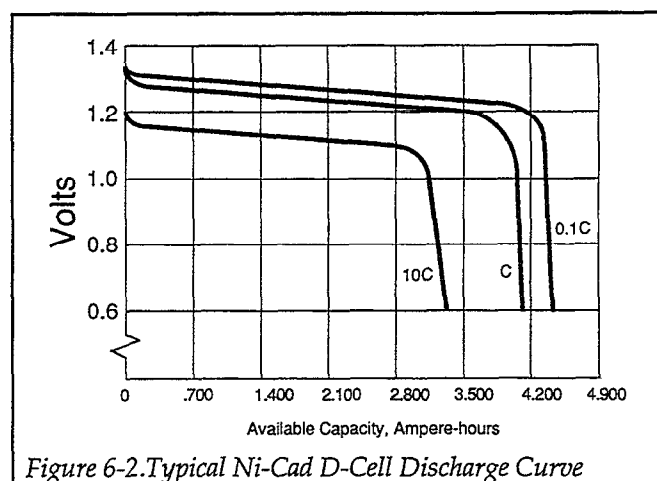
1. Load
2. Discharge/recovery times (duty cycle)
3. Ambient temperature
4. Charge rate
5. Charge time
6. Rest time after charge
7. Previous cycling history

Many manufacturers of nickel-cadmium cells rate the capacity of their cells in ampere-hours, based on a final voltage of 1.0V at its one-hour discharge rate. The term "C", which is found on many Ni-Cad discharge curves, indicates a discharge rate which is numerically equal to the rated ampere-hour capacity of the cell. Put simply, if the cell is rated at 4.5 ampere-hours capacity, then "C" represents a 4.5 ampere drain for a duration of one hour.

If a current is discharged from a cell at a rate other than "C", the discharge rate is expressed as a percent of the C rate. For example, a discharge rate of one-tenth the C rate is expressed as 0.1C. Likewise, 2C represents a discharge rate of 2 times the C rate. The cell will perform for a corre-

spondingly greater or shorter period, and there may be minor differences in the total energy delivered by the cell.

A typical discharge profile for a D-cell Ni-Cad battery is shown in Figure 6-2. The graph shows three different discharge curves of volts versus available capacity in ampere-hours. In the following example, we will measure voltage across a constant resistive load over time, from which we can plot a similar curve.



Test Configuration

Data to construct a discharge curve can be gathered with the Model 576 using time stamping, conditional triggering on events, and acquisition subroutines which run at different rates. These features enable the 576 to monitor the slow and fast discharge phases of the cell.

From the discharge curve in Figure 6-2, we see that initially, the voltage remains relatively constant. This portion of the curve can be monitored at a relatively slow rate. However, when the cell voltage approaches the knee of the curve, at about 1.2V, the voltage drops more quickly. At this point, the 576 should be shifted to a faster sampling rate. The actual curve is determined by completely discharging the cell through a constant resistance load. A typical test setup is shown in Figure 6-3.

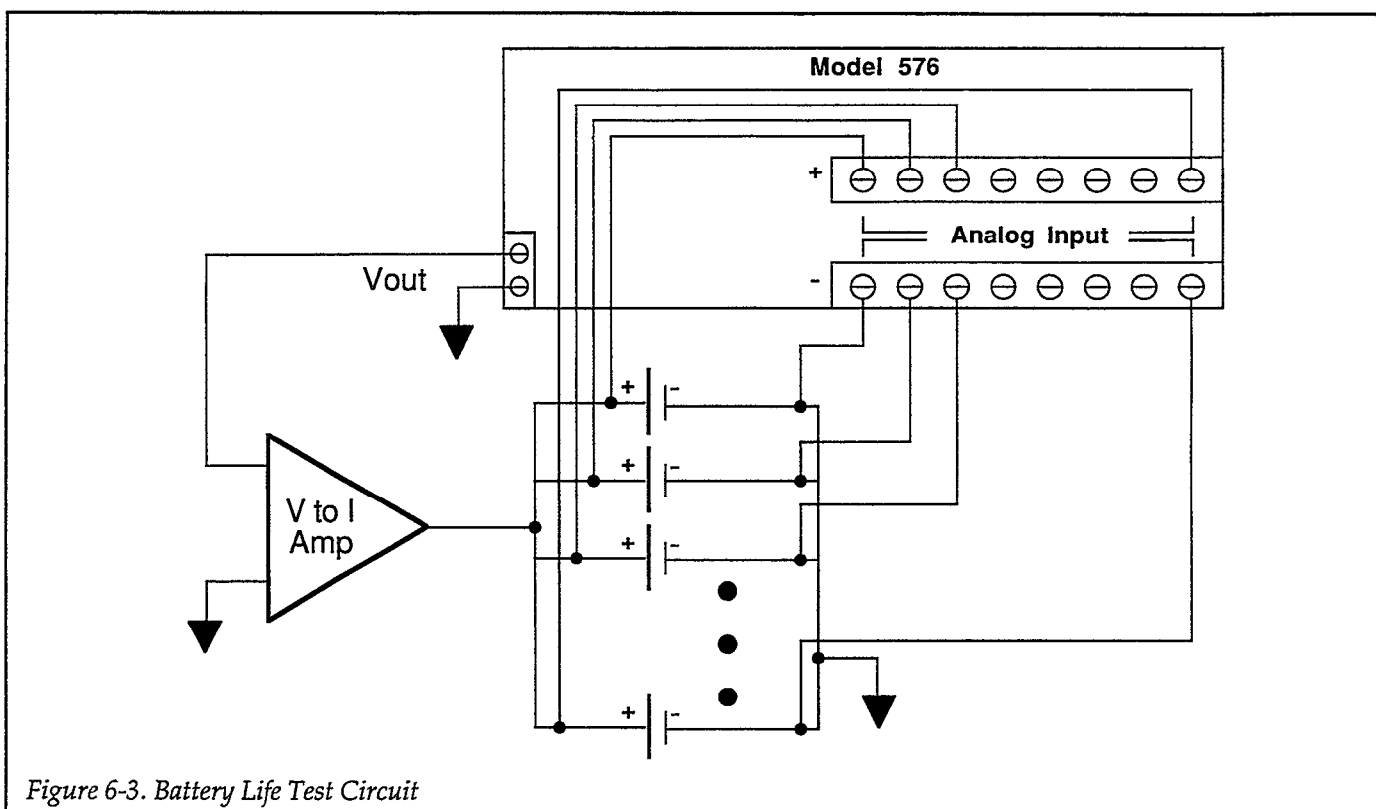


Figure 6-3. Battery Life Test Circuit

Program Description

Because a large number of points will be collected, the memory expansion option has been installed in the 576. The program can also be used without the memory option by decreasing the number points or reducing the sampling rates.

The program begins by initializing the GPIB interface bus and setting up the 576 system defaults. This example uses an engineering unit conversion of DC volts (DCV), and a data format of ASCII without a leading prefix (ASCN). Two data buffers are also dimensioned. One buffer stores the data collected at the slow rate, while the second buffer stores data collected at the fast rate.

The program then downloads the main body of commands to the 576. The downloaded program consists of two subroutines. One subroutine, RDSLOW, is called from the main body of the program. The second, RDFAST, is called within the RDSLOW subroutine when the cell voltage falls below 1.2 volts. When the RDFAST subroutine is executed, the interrupt rate changes from 10 Hz to 100 Hz in order to capture more points in the

steeper discharge curve. This illustrates the 576's ability to conditional trigger based on voltage.

As the 576 collects the data, the host computer monitors for a full buffer B1 by checking the status of the BUFFER FULL bit in the 576 Serial Poll Byte. When the host determines that the buffer is full, it sends a Device Clear to the 576 to re-establish communications with the 576, and begins the process of retrieving the data. Since the system switches buffers as it collects data, it is possible that the first buffer will not be totally filled. The BUFF INDEX command is used to return the number of scans read into the first buffer so that only valid data may be identified.

Once all the data has been collected in the computer, the data can be analyzed to determine the discharge characteristic of the battery under test.

The program was written for a single battery but could easily have been modified for multiple battery testing. The program can be expanded to charge the battery, or to do battery life expectancy testing in a charge/rest/discharge cycle. Such a test would determine the number of charge/discharge cycles required to reduce the available capacity to 50 percent of the cell's rate value.

Example Code:

```

SYST :UNIT DCV;           ' DC Volts as Engineering
                          ' Units
SYST :FORM ASCN;         ' ASCII Data Format
BUFF DIM B0, 1, 100000;  ' Dimension buffer 0
BUFF DIM B1, 1, 100000;  ' Dimension buffer 1

ONINT RDSLOW, 10, HZ;    ' Execute the RDSLOW
                          ' subroutine at 10 Hz rate

SUBR RDSLOW;             ' Define RDSLOW sub-
                          ' routine
  IF 1, 0, GT, 1.2;      ' If reading > 1.2V then read
                          ' battery slowly and
                          ' store in B0, else
    READ 1, 0, B0;
  ELSE;
    ONINT RDFAST, 100, HZ; ' execute the RDFAST sub-
                          ' routine at a 100Hz rate.
  ENDIF;                 ' End IF condition
ENDSUB;                  ' End of subroutine

SUBR RDFAST;             ' Define RDFAST sub-
                          ' routine
  READ 1, 0, B1;         ' Read battery volts, store
                          ' in buffer B1.
  IF B1, FU;             ' If buffer 1 is full,
    HALT;                ' Halt execution, issue SRQ.
  ENDIF;                 ' End IF condition
ENDSUB;                  ' End of subroutine
X;                       ' Execute instructions

WAIT ON BUFFER FULL DEVICE CLEAR

BUFF INDEX B0; X;        ' Get valid points in B0.
BUFF READ B0; X;        ' Read data in buffer 0 and
                          ' return in ASCII volts. Dis-
                          ' card data beyond point in-
                          ' dicated by BUFF INDEX.
BUFF READ B1; X;        ' Read data in buffer 1, re-
                          ' turn in ASCII volts.

```

ANALYZE DATA

LAB ACQUISITION WITH THE MODEL 576 AND ASYSTANT GPIB

Personal computers are often used in university research and industrial R&D labs to control or monitor experiments or processes, especially where there is a need to constantly analyze input data or adjust output parameters. The computer program may regulate temperatures or pressures, control valves, monitor for out-of-range conditions, count events, or perform countless other

tasks. The computer display may simultaneously graph voltages, currents, distance, temperatures, pressures, etc.

A Complete Hardware and Software System

A complete laboratory data acquisition hardware and software system can be assembled by combining a Model 576 with the Asystant GPIB Scientific Number Cruncher. The 576 offers specialized signal conditioning, easy signal connections, engineering unit conversion, and analog and digital input and output channels. The Asystant GPIB software package features menu-driven operation, sophisticated graphing, and GPIB control capabilities.

The 576 provides a complete set of engineering unit conversions which directly convert stored data to the most common electrical and physical units. These include:

RAW	= Raw A/D counts, no conversion
DCV	= DC Volts
MA	= Milliampere
TCn [C F]	= Thermocouple type, degrees C or F, where n = J, K, S, T, E, B, or R
RTDn [C F]	= RTD type, degrees C or F, where n is 85 or 92
HZ	= Hertz

The data can assume a variety of formats including ASCII with a leading prefix, ASCII without a leading prefix, Motorola binary, or Intel binary formats.

The Model 576 includes 16 single-ended or 8 differential analog inputs, dual analog output channels, and 32 TTL-compatible digital channels. In many cases, transducers may be used which require specialized signal conditioning. For this purpose, the Model 576 includes an expansion slot which accepts any of several optional signal conditioning modules. Commonly-used modules include the AIM7 for thermocouple inputs, the AIM6 for RTD inputs, the AIM8 for strain gages and load cells, and the AIM9 for LVDTs (Linear Variable Differential Transformers). The AIM4, AIM5, DIM, or DOM1 modules may also be used where isolated analog or digital I/O is required. These modules provide up to 500 volts peak channel-to-channel and channel-to-ground isolation.

Asystant GPIB Environment

Asystant GPIB provides an easy-to-use, menu-driven approach to GPIB communication, analysis, and graphics. Figure 6-4 shows that the GPIB programming screen is made up of four windows. The "GPIB Devices" window lists all the GPIB instruments which can communicate with the software. A device can be given a descriptive title to simplify programming and addressing. In this case, the name "K576" was declared as the device's primary address. The TIMEOUT (in seconds) is also declared.

The GPIB Main Menu contains choices for "Device Configuration", "Interactive Mode", and "Program Mode" (Figure 6-4). The Device Configuration sets up communication parameters for the GPIB device(s) being addressed. Interactive Mode enables the user to interactively communicate with GPIB instruments, while Pro-

gram Mode permits writing routines which automate the steps carried out in the Interactive Mode (Figure 6-4).

Measuring Temperature

A typical lab experiment might be to monitor the temperature of a chemical process such as titration or distillation. The operator can quickly write a routine in Asystant GPIB, such as the one shown in Figure 6-4, to acquire data from a thermocouple over a set period and then graph the data.

The routine first initializes the GPIB bus. The GPIB device (in this case "K576") is selected and cleared. Next, the Talk and Listen (T&L) index is reset so that the data array pointer points back to the first index of the array. This insures that any possible leftover data will be cleared from the array.

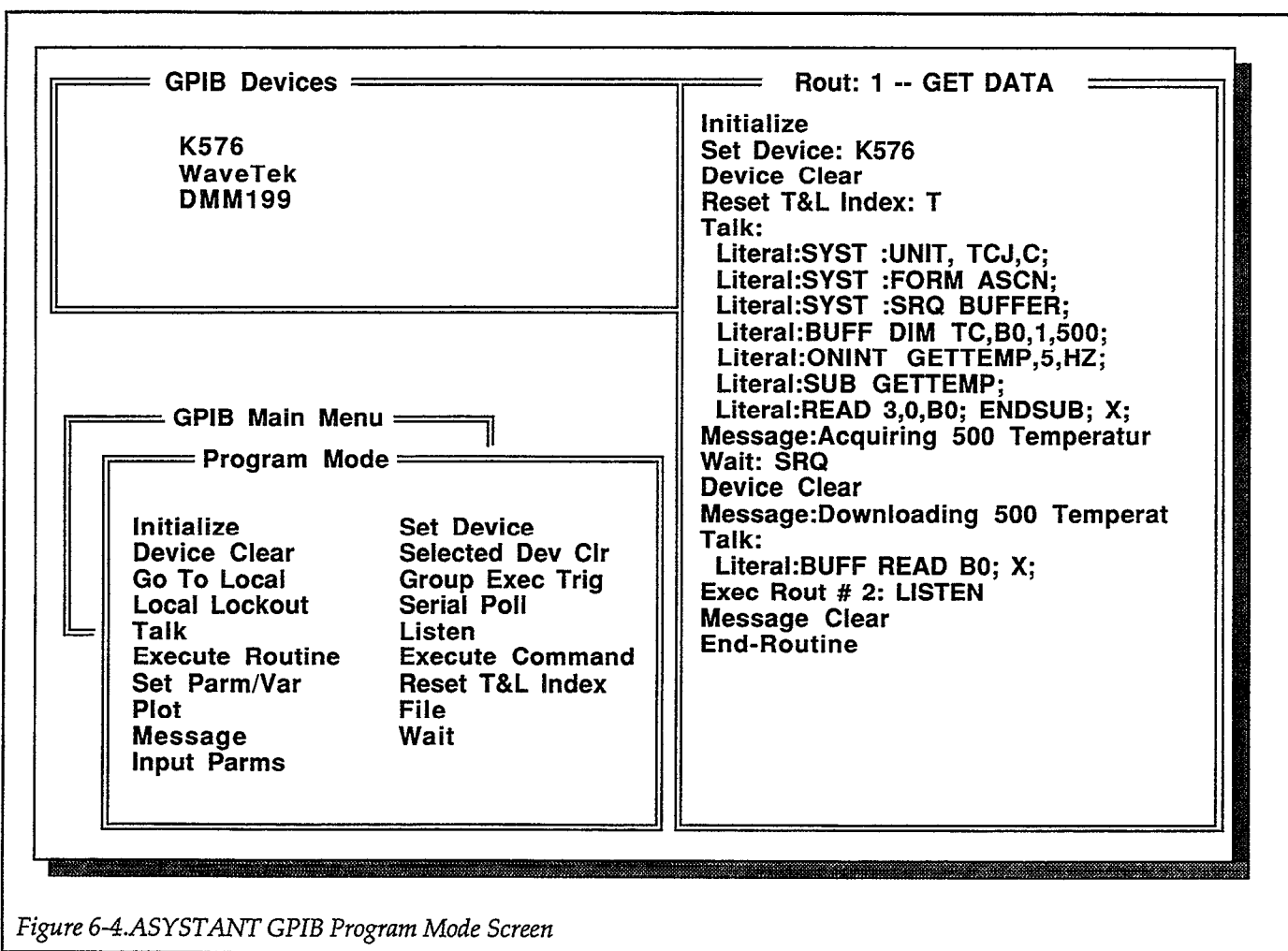


Figure 6-4. ASYSTANT GPIB Program Mode Screen

The main body of the program is sent out over the GPIB bus to the Model 576 using TALK literal strings. The literal strings tell the 576 to:

1. Allocate buffer space "B0" for 500 scans of one channel,
2. Set the engineering units conversion default to thermocouple units for a J-type TC, with degrees C as the temperature scale, and finally,
3. Issue a request for service (SRQ) when the data buffer has been filled.

The next set of literal talk strings tells the 576 to execute the subroutine GETTEMP at an interrupt rate of 5 Hertz. (Interrupt rate is set with the ONINT command.) The READ command in the subroutine GETTEMP instructs the 576 to read from channel 0 of slot 3, which contains an AIM7 TC card, and to store the readings in buffer B0.

A message is printed to the screen telling the operator that the 576 is in the process of acquiring 500 temperature readings. While the 576 collects data, Asystant GPIB waits for an SRQ. When the SRQ is received from the 576, the routine issues a DEVICE CLEAR command to re-establish communications with the 576. A message is

then displayed indicating that the PC is uploading the readings from the 576.

The final operation is to send another talk literal to the 576 to retrieve the data. The literal string "BUFF READ B0; X;" forces the 576 to read buffer 0, convert the data to the specified data format and engineering units. Asystant GPIB then executes routine #2 which performs a listen from the GPIB bus and places the data in variable T. When all of the data have been stored in variable T, Asystant GPIB's graphics and analysis functions can be used to plot or reduce the data.

Graphic Results

Graphics may be an essential part of an experiment because they immediately portray trends, data vs time relationships, or other information which may not be obvious from raw data. A typical plot of temperature is shown in Figure 6-5.

The entire graphing routine can be automated to repeat as many times as the user desires, thereby permitting constant data collection and plotting. This enables the operator to perform other tasks while the software and hardware collects data.

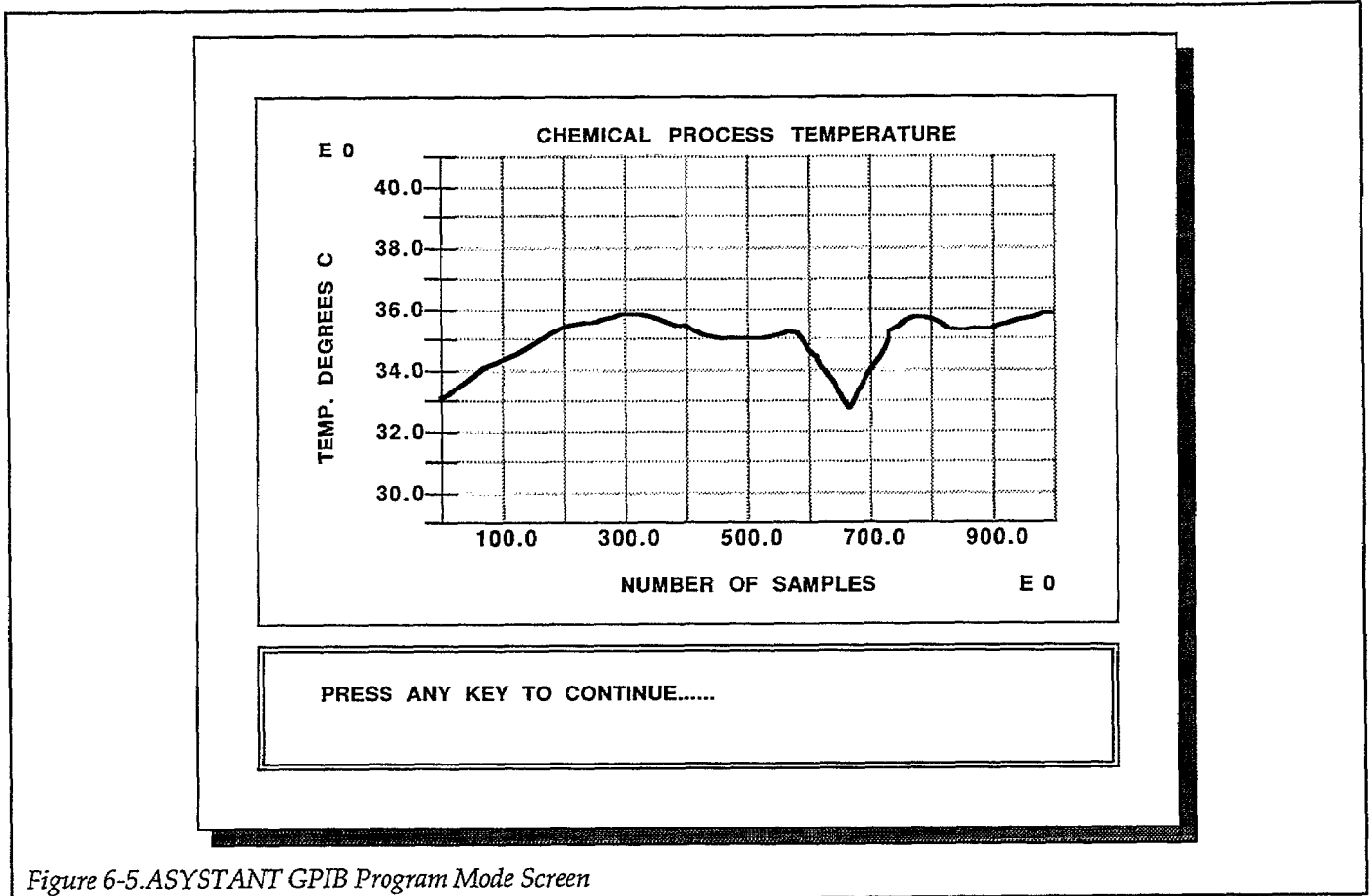


Figure 6-5. ASYSTANT GPIB Program Mode Screen

MULTI-CHANNEL VOLTAGE MEASUREMENTS WITH HARDWARE TRIGGERING

Physical and electrical testing often involves the evaluation of response, elasticity, recovery time, or other parameters after the application of a transient stimulus. Examples include monitoring pipes or rubber hoses after a momentary increase in pressure, or the testing of amplifier slew rate after a large change in input signal. Response may resemble Figure 6-6.

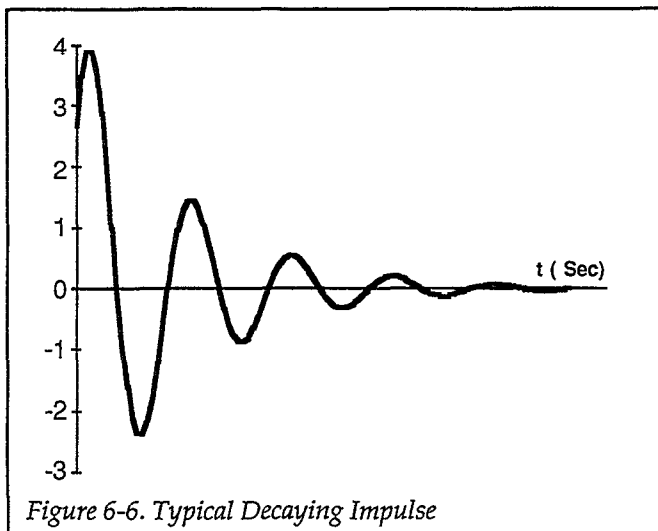


Figure 6-6. Typical Decaying Impulse

Multi-Channel, High-speed, Triggered Acquisition

Capturing high-speed, transient events, which may occur over a matter of milliseconds or microseconds, requires a fast sampling rate and a hardware-based trigger. It is also advantageous to trigger the start of acquisition so that system memory is not wasted on useless readings. If several phenomena are to be measured, multiple inputs will be needed to capture all the data. A conceptual block diagram of a high-speed, triggered, multi-channel acquisition system is shown in Figure 6-7.

Test Configuration

The Model 576 can acquire triggered data at a rate of 31,250 samples/second over multiple channels, and also initiate acquisition within 200 nanoseconds of the trigger signal (see Figure 6-7). The 576 is programmed to monitor a trigger channel, and acquires data only when the signal on the trigger channel has exceeded 3.0 volts. At that time, the Model 576 takes high-speed readings on 5 analog input channels until the 576 data buffer is full.

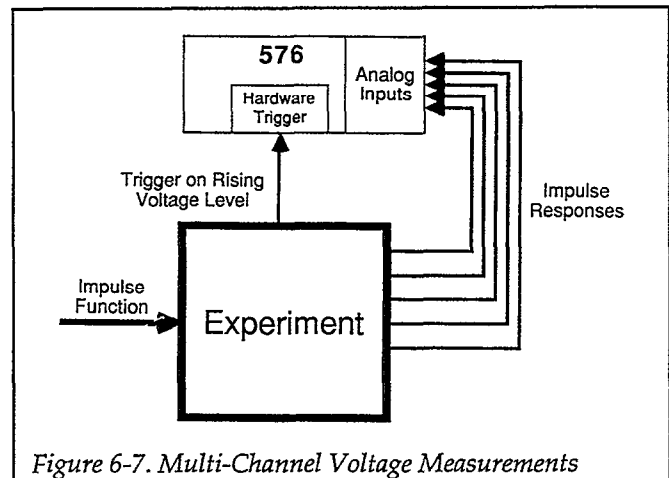


Figure 6-7. Multi-Channel Voltage Measurements

Program Description

The first section of the program initializes the GPIB bus, pre-programs the Model 576 for a default engineering units conversion of DC volts (DCV), and allocates buffer B0 for 10,000 scans of five channels.

The 576 trigger can be configured for level, pass band filter, coupling, slope, and related parameters. This application sets up the trigger filter for 100kHz. The filter prevents the hardware from being falsely triggered by noise.

The trigger level is set for 3.0 volts, and the trigger mode is set for "LATCH". In LATCH mode, acquisition continues after the trigger condition is first satisfied. Coupling is set for AC. The slope parameter is set to "above".

Once the initial test conditions have been set up, the last instruction commands the 576 to begin taking data at high speed when the trigger event tests true. This is accomplished by using the READ command with the "QUICK" option. The READ...QUICK command acquires all five channels at an aggregate throughput rate of 31.25 kHz.

When the buffer has been filled, the data can be retrieved from the 576 for analysis. This is accomplished by executing the 576 BUFFER READ command as shown in the program listing. When the data is in the host computer, graphing or other analysis of the data can be performed.

This program highlights the unique hardware triggering feature of the 576. The Model 576 hardware trigger includes FOLLOW and LATCH modes, AC and DC cou-

pling, filtering from 1kHz to 1MHz, and a trigger voltage range of -10 to +10 volts.

Example Code:

INITIALIZE SYSTEM

```

SYST :FORMAT ASCN;      ' Set default data format to
                        ' ASCII, no prefix.

SYST :UNIT, DCV;        ' Set Eng. Units to DCV

BUFF DIM B0, 5, 10000;  ' Allocate buffer B0 for 10k
                        ' readings from 5 channels.

CHAN 2, 0 :FILTER 100K; ' Set TRG1 to 100kHz filter.
TRIG TRG1, 3.0, LATCH, DC, ' Setup trigger to use TRG1
  ABOVE;                ' local input. Set LATCH
                        ' mode, DC coupling, and
                        ' trigger when signal is
                        ' above 3.0 V.

READ 1, 0-4, B0, QUICK; ' Read quick at 34,500kHz
                        ' rate from slot 1, channels
                        ' 0-4, 10000 samples/
                        ' channel.

BUFF READ B0;           ' Read points from 576 into
                        ' controller's memory as
                        ' soon as 576 becomes a
                        ' talker.

X;                       ' Execute program.

```

INPUT DATA FROM 576

PROCESS MONITORING AND CONTROL: TRIGGERING AND TIME STAMPING

Monitoring and control applications typically require a decision-making mechanism to effect a change when a particular condition exists. Time information may also be used in the decision-making process, i.e. a certain change can be initiated at a certain time, or locked out until a desired time. These control functions are normally implemented as closed-loop systems using conditional triggering.

Process Monitoring

Closed-loop process monitoring and control requires at least one input and one output. The input may be a generic voltage, or it may come from a specialized transducer such as a thermocouple, strain gage, or pres-

sure transducer. The Model 576 can accept most transducers directly, or with the aid of an option module.

Control output can be either digital or analog in nature. The Model 576 can directly sense and drive TTL-level signals, or even higher voltage and current loads with the optional PCM3 external relay board or DIM1 and DOM1 isolated digital modules. Analog output is available from on-board D/A channels, or from various optional voltage and 4-20mA current loop output modules.

Effective closed loop control requires a wide range of logical operators that can be used to compare inputs against pre-determined set-point values. The 576 firmware includes many conditional tests to help evaluate such conditions. These tests are used within IF... THEN...ELSE constructs, and include the following:

1. Less Than
2. Greater Than
3. Equal To
4. Not Equal To
5. Less Than or Equal To
6. Greater Than or Equal To
7. Buffer Full
8. Buffer Not Full
9. Between
10. Not Between

Conditional Triggering can also be based on a particular time or date.

Test Configuration

Figure 6-8 shows an example of a process monitoring and control application using the 576 with an AIM7 thermocouple input module. A J-type thermocouple is the input sensor. The 576 monitors the input from the thermocouple and makes decisions based on the time and temperature. These decisions may include turning the heating system on and off, shutting down the heating system or activating an alarm if the temperature is too high.

For this example, the vat temperature must be maintained at 70 degrees C, which is the control set-point. After a certain time, the 576 performs a time trigger to activate a data logging sequence in which data will be collected and time stamped. Lastly, if the process exceeds an upper temperature limit, or if the elapsed test time exceeds 30 minutes, the system performs a shutdown.

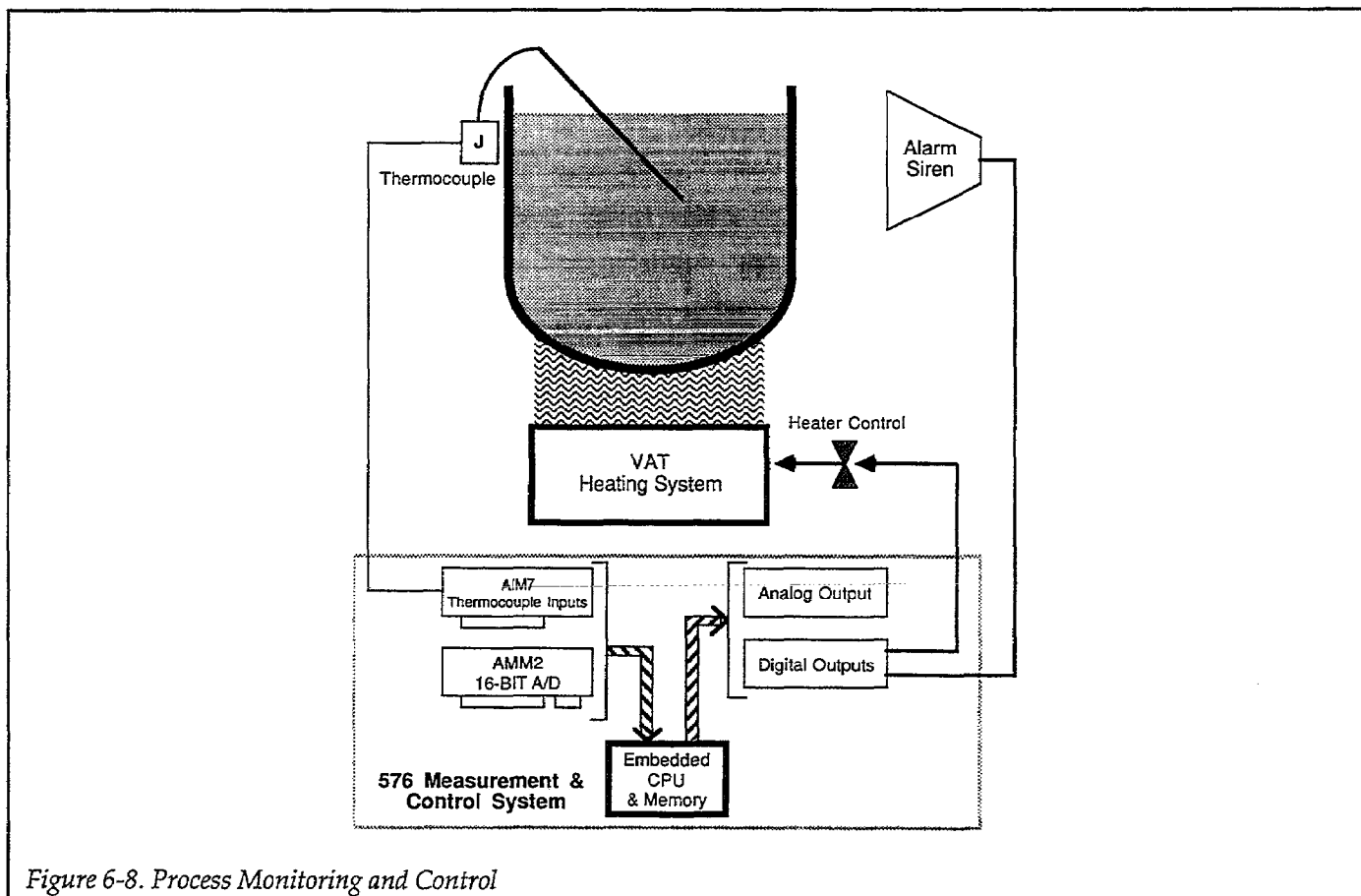


Figure 6-8. Process Monitoring and Control

Program Description

The programming of this application uses nested IF..ELSE..ENDIF commands in a subroutine to log and time stamp data. The initial segment of the program initializes the GPIB bus. The program prompts the user for the high and low set-points. For this example, the high set-point is 72 and the low set-point is 68 degrees C. The 576 is then initialized with various system set-up parameters including the data format (ASCII), and an engineering unit conversion of degrees C for Type-J thermocouple.

To time stamp the data, the 576 requires additional system configuration instructions. First, the 576 must be told to time stamp on each scan of the READ command. The system clock must be set for standard format, thus allowing the date and time to be initialized to 07/20/1990 and 00:00:00, respectively. This enables the system to store elapsed time based on a start time of 00:00:00.

Following the SYST commands, the BUFF DIM command allocates three data buffers. Buffers B0 and B1 will store one sample each, while B1 stores 1200 samples. Because the 576 will acquire temperature measurements, the TC specifier is included so that the cold junction reference temperature will be stored with other data. Thus, the total memory requirement per scan is 10 bytes: four bytes for the temperature measurement, and six bytes for time stamp data.

The main section of the program uses a single ONINT command telling the 576 to execute the subroutine "MONITOR" every 500 milliseconds. MONITOR contains the conditional triggering statements, and uses DEBUG numbers as markers to determine which part of the subroutine is executing at any given time.

The first IF statement checks to see if the temperature of the process is greater than 100 degrees. If the statement is true, the 576 turns on a siren, turns off the heater, and halts the 576 program. The POKE commands act directly

on the digital ports where the siren and heater control are connected.

If the initial condition tests false, the next IF statement checks whether the vat temperature is greater than the desired high set-point of 72 degrees C. If the condition is true, another POKE command turns off the heater. If this condition is false, the third IF statement tests whether the vat temperature is less than 68 degrees. If the condition is true, then the heater is turned on. If the condition is false, the DEBUG command sets binary bits 0-2 to "010" (2, decimal) in the serial poll byte.

The three DEBUG commands in the first nested IF..ELSE..ENDIF commands provide a means of tracking the portion of the program currently executing. If a given condition tests true, the corresponding code executes and the associated DEBUG value is assigned to bits 0-2 in the serial poll byte. These bits may be checked periodically in the program, and appropriate actions or graphics generated in response.

The next nested IF checks whether the time is greater than 00:15:00 minutes and less than 00:26:00. If the condition is true, the 576 logs and time stamps the temperature readings.

The next IF checks to see if the elapsed time is greater than or equal to 00:30:00 minutes. If the condition is false, then the current temperature and time are read and returned to the computer immediately. If the statement is true, then the Model 576 shuts down the control process and turns off interrupts. The program then reads the data buffer and halts.

Time Stamped Results

After a process time of 30 minutes, the 576 returns the time stamped data to the host computer over the GPIB

bus where it can be analyzed or graphed. A typical print-out of the acquired time stamped data might resemble the following:

```
0:16:00, 7.210450E001
0:16:01, 7.231431E001
0:16:02, 7.301015E001
```

Time stamping, which is a unique feature of the 576, has various modes for adding time and date to data scans. The above display mode is "SCAN, TIME", which stamps data with the time at the start of every scan. Other modes and their respective output formats are shown below:

SCAN CLOCK — The time and date are stamped at the beginning of each scan.

```
04/11, 00:15:00, .210450E0010
4/11, 00:15:01, .231431E0010
4/11, 00:15:02, .301015E001
```

ONCE, TIME — The time only is stamped once at the beginning of the acquisition.

```
00:15:00,
7.210450E001
7.231431E001
7.301015E001
```

ONCE, CLOCK — The time and date are stamped once at the beginning of the acquisition.

```
04/11/1990,00:15:00,
7.210450E001
7.231431E001
7.301015E001
```

Example Code:

INITIALIZE SYSTEM

```

SYST :EOI Disable;           ' Disable EOI

PROMPT FOR TEMPERATURE LIMITS

SYST :UNIT TCJ :FORM ASCN;  ' Set engineering unit con-
                             ' version for thermocouple
                             ' units type-J and ASCII
                             ' data format.

SYST :STAMP SCAN, TIME;     ' Set system to time stamp
                             ' on each scan.

SYST :SRQ IDLE;            ' Issue SRQ when not busy

SYST :CLOCK, STD,          ' Set Time and Date. Time
07/20/1990, 0:00:00;       ' is set for 0:00:00 for pur-
                             ' poses of timing.

SYST :POKE 5, A, 248;      ' Set port for output

BUFF DIM TC, B0, 1, 1;     ' Allocate data buffers.
BUFF DIM LONG, B1, 1, 1;
BUFF DIM TC, B2, 1, 1200, STAMP;

ONINT MONITOR, 500, MSEC;  ' Set up backgnd function
                             ' to run every 500ms

SUBR MONITOR;              ' Monitor subroutine
IF 3,0, GT, 100;           ' If temperature >100C,
POKE 5,A,1;                ' access digital port and
POKE 5,B,2;                ' Turn on alarm.
HALT;                      ' Halt 576 execution

ELSE;
V$ = "IF 3,0 GT " + HIGH$  ' Set up string
+ " TCJ,C; "

OUTPUT V$                  ' If temperature > high
                             ' set-pt,

DEBUG 0;                   ' Set 576 debug number 0,
POKE 5,A,1;                ' Access digital port and
POKE 5,B,0;                ' Turn heater OFF.

ELSE;
V$ = "IF 3,0 LT " + LOW$   ' Set up string
+ " TCJ,C; "

OUTPUT V$                  ' If temperature < low
                             ' set-pt,

DEBUG 1;                   ' Set 576 debug number 1,
POKE 5,A,1;                ' Access digital port and
POKE 5,B,1;                ' Turn on heater.

ELSE;
DEBUG 2;                   ' Set 576 debug number 2.
ENDIF;                     ' End nested IF
ENDIF;                     ' End nested IF

```

```

ENDIF;                      ' End IF

IF TIME, GE, 00:15:00;     ' If time >= 15 min.
IF TIME, LT, 00:26:00;    ' and < 26 min. elapsed
READ 3,0,B2;              ' Read time stamped temp.
ENDIF;                    ' END nested IF
ENDIF;                    ' END IF

IF TIME, GE, 00:30:00;    ' If time >= 30 min.,
POKE 5,A,1;               ' access digital port and
POKE 5,B,0;               ' turn off heater,
DEBUG 3;                   ' set 576 debug number 3,
INTOFF;                   ' Turn interrupts off,
BUFF READ B2;              ' Perform a buffer read,
HALT;                      ' HALT 576 execution
ENDIF;                    ' END IF

READ 3,0,B0;               ' Read temperature
BUFF READ TCJ,C B0;       ' Read buffer
READ TIME, B1;             ' Read real time clock
BUFF READ TIME, B1;       ' Read buffer
ENDSUB;                    ' END of subroutine

SET UP GRAPHICS SCREEN TO SHOW
TEMPERATURE STATES

X;                          ' Execute 576 program

ALLOW 1-SECOND DELAY FOR PROCESS TO COMMENCE
BEFORE ACTIVATING GRAPHICS.

UPDATE SCREEN BASED ON DEBUG NUMBER - CHECK
SPOLL BYTE FOR VALUE OF DEBUG BITS 0, 1, AND 2, AND
BRANCH TO APPROPRIATE CODE BASED ON VALUE.

END

```

PORTABLE AND REMOTE ACQUISITION

Checking flow rates and/or pressures are common tests in industries such as natural gas and petroleum. Monitoring these conditions often requires an engineer to go to a remote sight and carry out tests to verify that equipment is operating normally. Ideally, the associated test system should be light-weight, portable, and operate from a battery or some other portable power source.

Portability

A lap-top computer is a logical choice as a controller for a portable data acquisition system. Some lap-tops can operate from internal batteries or from an external DC source such as an automotive electrical system. However, most lap-top computers lack an IBM-compatible expansion slot, making them incompatible with virtually all data acquisition boards and standard communica-

tions interface cards. Most of these systems do have a serial (RS-232) port, however, so this limitation can be easily overcome using Keithley's 500-Serial RS-232 to GPIB adapter.

The 500-Serial adapter converts the GPIB connector and communications standards to RS-232 serial protocol. The unit comes with a 25-pin serial cable and 9-to-25 pin adapter, and will thus mate GPIB instruments to most lap-top, desk-top, and mini computer serial ports.

As a side benefit, the 500-Serial permits greater operating distances (up to 100 feet) than standard GPIB specifications, and uses flexible, relatively inexpensive serial cable. Figure 6-9 gives a simplified view of this application.

For operation at distances greater than 100 feet, a device called "Micro488A" enables programming and control of GPIB instruments at distances up to 400 feet from the computer. Like the 500-Serial, the Micro488A communicates GPIB commands and data over standard RS-232 serial ports and cabling.

If an application involves collecting a large amount of data, the Model 576 can be outfitted with a memory expansion option. This option extends the data capacity of the 576 to approximately 240,000 analog or 480,000 digital readings.

With appropriate programming techniques, the Model 576 can operate independently of the computer, or even be completely disconnected from the computer. The "SAVE" command options make it possible for the 576 to retain a program while the power is off, automatically execute the program when power is restored, and save the data. Thus, the computer need not be left at the test site. More than one Model 576 can be used at a remote site by simply downloading a program to each 576.

In a typical auto-startup application, the program would be downloaded into the Model 576. The 576 SAVE switch must then be turned ON, and the SAVE mode programmed for automatic program save and restart ("SYST:SAVE PROG"). The unit would then be switched off and transported to the test site. When power is restored, the 576 executes the stored program and logs data. After the data have been collected, the technician returns to the site, sets switch 7 ON to disable further program execu-

tion, disconnects the 576, and takes the unit back to the lab. When the system is powered up again, the data can be retrieved.

Test configuration

Figure 6-9 shows a configuration that may be used at a remote pumping station. The transducers may be installed permanently at the pumping station, permitting quick connection to the 576. The 500-Serial attaches directly to the 576, and an RS-232 cable connects the 500-Serial to the computer. The lap-top and the 576 are both powered from a 12VDC source. The 500-Serial receives power through the serial port.

Program Description

The following two programs illustrate how the 576 can be used for remote, unattended acquisition. Both rely on the 500-Serial adapter for communication between a lap-top computer and the 576. It is also assumed that the memory expansion option is installed in the 576. In this application, the memory option provides eight hours' run time at an approximate sampling rate of 1.4 Hz (ONINT rate = 720 milliseconds). This rate will collect a total of 200,000 analog readings.

The "downloader" program, REMOTEDN, begins by initializing the serial port and 576. It opens COM port 1 and sets the baud rate for 19,200 baud with no parity, 8 data bits, and 2 stop bits. Next, the program prompts for current time, date, and format to be used by the 576.

The next section of the program sets up system parameters. These include commands to SRQ when the data buffer is full, data format of ASCII without a leading prefix (ASCN), and DC volts (DCV) as the default engineering unit conversion. This section also instructs the 576 to commence execution when the time is 8:00 AM. Lastly, the 576 is told to save the contents of the buffers and program when the instrument is powered down.

The next section of the program allocates buffer space. Two buffers are used to store the data. Three channels of flow data will be stored in buffer B0 and two pressure signals will be stored in buffer B1. The first three channels require a gain of X10 and the remaining two channels require a gain of X2.

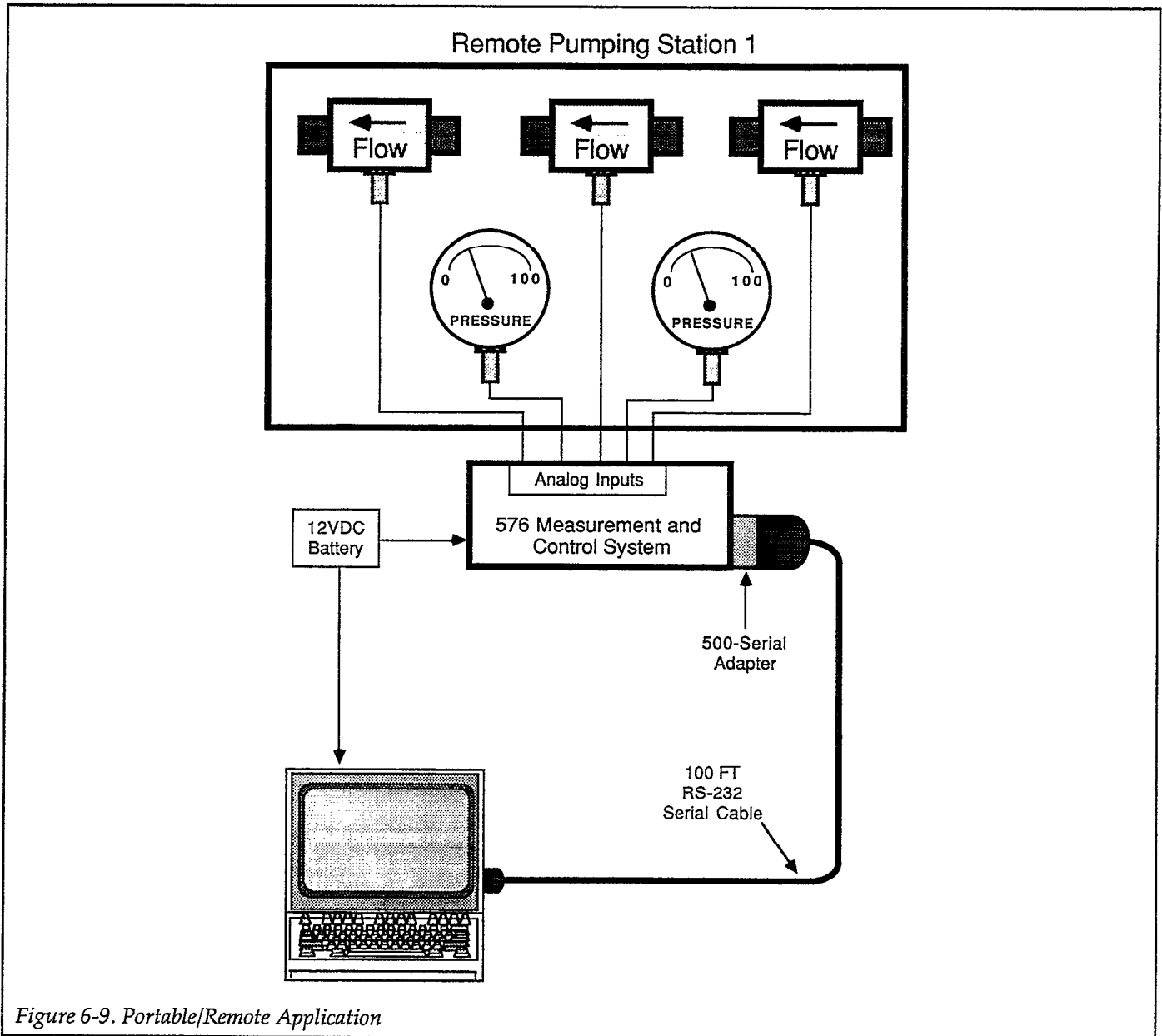


Figure 6-9. Portable/Remote Application

Next, the main body of the test is sent to the 576 instructing the 576 to execute the subroutine LOGDATA at an interrupt rate of 720 milliseconds. While the subroutine LOGDATA is running in the background, a foreground routine checks when buffer B1 has been filled. When buffer B1 is full, the program HALTs execution.

The subroutine LOGDATA consists of two READ commands which read the 5 channels and store the data in the designated buffers. When the buffers have been filled, the 576 issues an SRQ. The SRQ LED on the 576 front panel also illuminates, indicating that the test is completed.

The second program, "REMOTEUP", can now be loaded into the computer to recover the data from the 576. REMOTEUP sends the 576 a Device Clear to re-establish communications. Several SYST commands tell the 576 to use the ASCII/no prefix data format and DC Volts engineering unit conversion. Finally, a BUFFER READ is executed to retrieve the data from buffer B0. The program receives the data from the 576, and can store it in memory, write it to disk for future use, graph, print the results to the screen, etc. The data from buffer B1 is retrieved by the same method.

Example Code:

```
' REMOTEDN – Program Downloader
INITIALIZE THE SERIAL PORT
INITIALIZE THE MODEL 576
SYSTEM :IDN ?;          ' Request System ID
PROMPT FOR DESIRED DATE/TIME FORMAT ("FMT$"),
AND CURRENT TIME ("TIM$") AND DATE ("DT$")
ENTER SYSTEM INFO AND DESIRED START TIME:
SYST :SRQ BUFF;        ' SRQ on buffer full
SYST :UNIT DCV;        ' DCV for Eng Units
SYST :FORM ASCN;      ' ASCII format
SYST :TRIG 08:00:00;   ' Execute at 8:00AM
SYST :SAVE PROG;      ' Save Program and Data
CMD$= "SYST :CLOCK "+ FMT$+ " "+ DT$+ " "+ TIM$+ ";";
OUTPUT CMD$            ' Set Real Time Clock
SYST :CLOCK ?;        ' Return time for check
' PRINT DATE AND TIME
BUFF DIM B0,3,40000;   ' Allocate data space,
BUFF DIM B1,2,40000;   ' Buffer B0 and B1
CHAN 1,0-2 :GAIN 10;   ' Set channel gains
CHAN 1,3-4 :GAIN 2;
' SEND MAIN PROGRAM AND SUBROUTINE TO 576
ONINT LOGDATA, 720, MSEC; ' Set subroutine rate
DO;                    ' Check full buffer B1
  IF B1 FULL;          ' If true, then
    HALT;              ' halt execution.
  ENDIF;
LOOP;
SUBR LOGDATA;         ' Subroutine LogData
  READ 1,0-2,B0;      ' Read into buffer 0
  READ 1,3-4,B1;     ' Read into buffer 1
ENDSUB;              ' End of Subroutine
X;                    ' Execute Program
```

Example Code:

```
' REMOTEUP – Data Uploader
INITIALIZE GP488B IEEE CONTROLLER AND 576
SYSTEM :IDN ?;          ' Request System ID
```

```
' Send SYST commands and return current time from 576
SYST :SAVE OFF;        ' Disable Save option
SYST :UNIT DCV;        ' DC volts for E-Units
SYST :FORM ASCN;      ' ASCII format
SYST :CLOCK ?;        ' Return time for check
' READ AND PRINT DATA AND TIME
' Send BUFFER READ to read channels 0, 1, and 2 of B0
BUFF READ B0; X;      ' Read buffer B0
FOR COUNT = 1 TO 40000 ' Loop and print data
  INPUT DATA POINT    ' until no more data
  FROM 576 PRINT DATA
  TO SCREEN
NEXT COUNT
' Send BUFFER READ to read channels 0, 1, and 2 of B1
BUFF READ B1; X;      ' Read buffer B1
FOR COUNT = 1 TO 40000 ' Loop and print data
  INPUT DATA POINT    ' until no more data
  FROM 576 PRINT DATA
  TO SCREEN
NEXT COUNT
ANALYZE OR GRAPH DATA
```

TESTING 8-BIT DACS WITH THE MODEL 576

Testing Digital to Analog converters (DACs) might be a normal part of quality screening of incoming components to identify parts not meeting specifications.

Source/Delay/Measure Testing of 8-Bit DACs

Testing DACs requires an instrument that can perform a Source/Delay/Measure test. A simple and inexpensive method is available using the Model 576's digital output and analog input capabilities. The 576 can provide all the requirements in hardware to test monotonicity and relative accuracy of DACs.

The concept of a Source/Delay/Measure test is relatively simple. The 576 provides a digital output to drive the DAC, waits for the DAC output to settle, makes an analog measurement of the output voltage, and stores it in a data buffer.

Test Configuration

The entire test of an 8-bit DAC is controlled by the 576. A 576 buffer contains values 0-255 which supply all possible digital input values to the DAC. Each value is individually sent through one of the digital input/output ports on the 576 to the DAC. After a short delay, the output voltage of the DAC is read by the Analog Master Measurement Module in the 576. The measured voltages are stored in the 576 for later retrieval and analysis by the host computer. A simple functional diagram is shown in Figure 6-10.

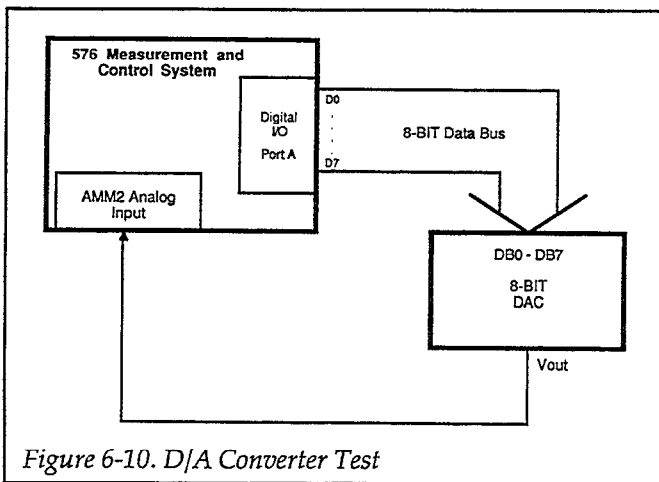


Figure 6-10. D/A Converter Test

Program Description

The program first initializes the GPIB bus and generates an array of digital points that will be applied to the DAC inputs. This data is stored in an array called DIGOUT%. Since the DAC is an 8-bit version, the digital values range from 0 to 255.

The next block of code initializes system information for the 576 including ASCII/no prefix data format (ASCN) and an engineering unit conversion of DCV. Two buffers of 256 points each are allocated in memory to hold the digital source values and the analog input measurements from the DACs.

Once the 576 has been initialized, the digital data are transferred from the host to the 576 with a BUFF WRITE command and a FOR-NEXT loop. Each string sent by the loop terminates in a comma (,) indicating that more information is to come. After the loop completes, a final string is sent terminated by a semi-colon (;) to indicate the end of the data. The BUFF WRITE command uses an engi-

neering unit of RAW to override the system default of DCV. Thus, the digital values are treated as raw integers, not voltages.

At this point, the program uses a DO...LOOP to drive the DAC through all available output levels. The 576 digital port outputs each of the 256 digital values in buffer B0, which the DAC converts to voltages. A delay of 10 milliseconds provides settling time for the DAC. Finally, the 576 measures the DAC analog output voltage and stores it in the data buffer B1.

When the test is completed, the measured points are transferred from the 576 to the host for analysis.

Modifying the Program

Instructions can be added to the program to increase the number of DACs tested. The 576 has four 8-bit digital ports which can be configured for output, thus controlling several 8-bit DACs. With some creativity, the experiment can be extended to a more complex applications, such as testing of 12-bit and 16-bit DACs.

Example Code:

```
FOR POINT = 0 TO 255          ' Create raw data points
  DIGOUT%(POINT) = POINT
NEXT POINT

SYST :FORM ASCN;             ' Data format is ASCII
SYST :UNIT DCV;              ' DC Volts engineering unit
SYST :SRQ DATA;            ' Issue SRQ when data is
                             ' ready
CHAN 5,0-3 :MODE OUT;       ' Set digital ports for output

BUFF DIM BYTE B0, 1, 256;    ' Allocate output buffer
BUFF DIM B1, 1, 256;        ' Allocate input buffer

BUFF WRITE RAW, 5, 1, B0     ' Write data to 576 data
                             ' buffer 0, override engi-
                             ' neering unit conversion
                             ' for RAW

FOR X=0 TO 254
  CMD$=STR$(DIGOUT%(X)) + ","
  OUTPUT CMD$                ' Code to send 255 CMD$ to
                             ' 576
NEXT X
CMD$=STR$(DIGOUT%(255)) + ";"

OUTPUT CMD$                  ' Code to send last CMD$
                             ' to 576
DO 256;                      ' Loop 256 times —
  WRITE 5, 1, B0;            ' Output single digital
                             ' point,
```

```

WAIT 10, MSEC;           ' delay 10 ms for settling,
READ 1, 0, B1;          ' read output of DAC.
LOOP;X;                 ' End loop and execute.

BUFF READ B1; X;        ' Read points from 576 into
                        ' controller's memory as
                        ' soon as 576 becomes a
                        ' talker.

SERIAL POLL             ' Is data ready to be read?

INPUT DATA FROM 576

PERFORM ACCURACY TEST AND MONOTONICITY TEST

```

LOW-CURRENT MEASUREMENTS USING THE MODEL 576 WITH A GPIB CURRENT AMPLIFIER

The Model 576, like most data acquisition systems, can measure analog signals ranging from a few millivolts up to 10 volts full-scale. For signals outside this range, it is usually necessary to provide signal conditioning in front of the data acquisition system's analog inputs. A specific example is using a high-gain amplifier as a "front end" to boost signals before they are applied to the Model 576. Many instruments have a high-level analog output which tracks the input signal. This output can be fed to an analog input of the Model 576.

Radiometric Measurements

This following application illustrates how the 576 can be used with the Keithley 428 Current Amplifier to form a powerful GPIB-based test and measurement system. The application concerns radiometry, which is the measurement of energy transfer through radiation. Specifically, the application tests the beam intensity of light emitting diodes (LEDs), a procedure which might be used to select or match the LEDs for luminance.

Beam intensity is defined as the radiant power per unit area normal to the direction of propagation, and is measured in watts/meter². A specialized photo detector, calibrated in amperes/watt, can be used for absolute measurements of intensity. In this example, relative brightness is the chief concern, so a simple LED similar to the test units is used as the sensor.

Test Setup

A block diagram of the test setup is shown in Figure 6-11. This application uses an analog output channel from the Model 576 to power the LED under test. A resistor limits current through the device. The light produced by the LED is directed to another LED which is connected to the input of the Keithley Model 428 Current Amplifier. The output of the detector is an extremely low current which requires using the 10⁹ V/A range on the Model 428. The output of the Model 428 is a high-level voltage that can be read easily by the Model 576.

Once the data is collected and returned to the host computer, the voltage can be scaled and converted back to current. This current is generally in the nanoamp or picoamp range.

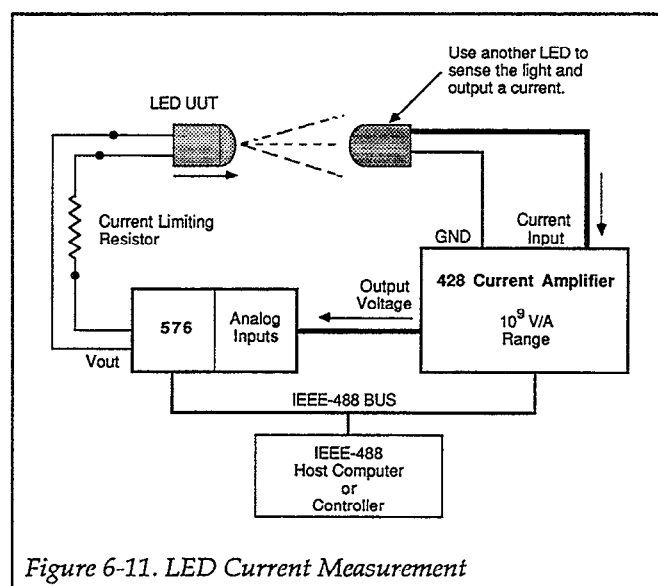


Figure 6-11. LED Current Measurement

Program Description

A simple program can control both the 428 and the 576. The program starts by initializing the GPIB bus, 576, and 428. The command "R9" sets for the 428 input range for 10⁹ V/A gain. Zero check is also disabled.

The next section of the program creates an array of 101 output values, in 0.1 volt increments. The points are generated using a simple FOR...NEXT loop, and stored in an array called "Vout". The 576 analog output section will generate a voltage ramp in 0.1V steps which drives the LED to increasing intensities.

Next, the 576 is configured for the test with a default engineering unit of DC volts (DCV) and a data transfer format of ASCII without leading prefix (ASCN). Two 101-point data buffers are allocated in the 576 for input and output data. A gain of x5 is programmed for analog input to increase the resolution of voltage measurements.

After initialization, the program writes the first 100 data points from "Vout" to B0, which has been associated with analog outputs lot 4, channel 0. Each point is delimited by a comma (,) indicating that the string is not complete. The 101st point terminates with a semi-colon, identifying it as the last data point.

The program next downloads a series of commands which instruct the 576 to output each data point held in buffer B0, wait 3 seconds, and read an analog voltage measurement into buffer B1. The "DO...LOOP" performs 101 passes.

Upon completion of the test, the program retrieves the analog input data in buffer B1 using a BUFF READ command. The data can then be graphed, analyzed, converted, or otherwise processed to determine LED performance and/or suitability for a given application.

Results

This test is useful for evaluating LEDs or other components where performance must be measured over a broad range of stimulus values. A more accurate means of measuring light energy is with a photodetector calibrated in amperes/watt. This type of detector makes it easy to calculate the light output of the source LED as a simple function of the measured current.

Example Code:

```
INITIALIZE GPIB BUS, 576, AND 428.

SET 428'S RANGE TO 109 V/A

DIMENSION INTEGER ARRAY "Vout%" OF 101 ELEMENTS

FOR POINT = 0 TO 100      ' Create data points for
  Vout%(POINT) = POINT/10 ' D/A
NEXT POINT

SYST :SRQ DATA;        ' SRQ when data is ready

SYST :UNIT, DCV, :FORMAT, ' Set up Engineering Units
  ASCN;                 ' to DCV and data format to
```

```
' ASCII without leading
' prefix.

BUFF DIM B0, 1, 101;    ' Allocate output buffer

BUFF DIM B1, 1, 101;    ' Allocate input buffer

CHAN 1, 0 :GAIN 5;      ' Set AMM gain to x5

BUFF WRITE 4, 0, B0     ' Write to output buffer

FOR pts% = 0 TO 99
  CMD$=STR$(Vout%(pts%))+", "
  OUTPUT CMD$           ' Code to send data
NEXT pts%               ' array to 576

CMD$=STR$(Vout%(100)) + ";"

OUTPUT CMD$            ' Send last data point

DO 101;                ' Loop 101 times
  WRITE 4, 0, B0;      ' Output 1 analog value,
  WAIT 3, SEC;         ' delay 3 seconds,
  READ 1, 0, B1;      ' read output of D/A,
LOOP;                  ' End loop

BUFF READ B1;          ' Read data from 576 as
                        ' soon as 576 becomes a
                        ' talker.
X;                     ' Execute

INPUT DATA FROM 576

PERFORM ANALYSIS
```

DATA LOGGING WITH THE APPLE MACINTOSH

A number of graphics, publishing, and related software packages are available for the Apple Macintosh. Many of these packages use pull-down menus and graphical interfaces, eliminating much of the labor of writing programs. Such features have also helped make the Macintosh a desirable controller for data acquisition.

Pull-Down Menu Approach

An example of acquisition software using pull-down menus is Labtech Notebook. The Macintosh version of Notebook is similar to the PC version, and supports three different GPIB interfaces (selection depends primarily on port/slot availability). These include the MacSCSI488 interface for a SCSI ports, the MacII488 interface for the Mac II NuBus, and the Mac488B interface for serial ports. Each interface converts input and output from the associated bus to GPIB-compatible signals (Figure 6-12). "MacDriver488" software, which is included with each interface, passes GPIB commands and data between the

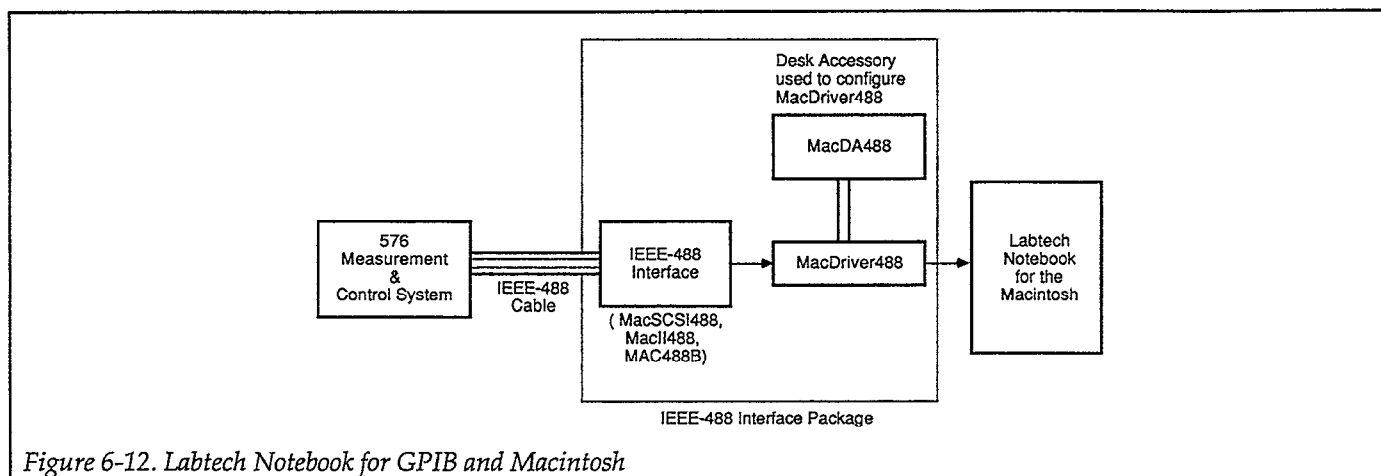


Figure 6-12. Labtech Notebook for GPIB and Macintosh

576 and Notebook. Once the data is in Notebook, it can be displayed and stored to disk in real time.

Collecting data from multiple channels on the 576 requires setting up different prompt command files for each channel. To illustrate this, suppose we wish to acquire data from channels 0 and 1 of an AMM2 card plugged into the 576. The following commands would be sent to read channel 0 on the AMM2, assuming the 576 has an GPIB address of 03:

```
OUTPUT 03; READ 1, 0, B0;
OUTPUT 03; BUFF READ DCV, B0; X;
ENTER 03
```

These commands could be stored in a file called "K576 CH0". A similar set of commands, shown below, reads channel 1 from the AMM2:

```
OUTPUT 03; READ 1, 1, B0;
OUTPUT 03; BUFF READ DCV, B0; X;
ENTER 03
```

These commands might be stored in a file "K576 CH1".

The first GPIB input channel in Notebook would then use "K576 CH0" as the prompt file, while the second GPIB input channel uses "K576 CH1". The 576 executes the prompt files when Notebook instructs the system to be-

gin the process. The resulting data can then be graphed in real time or written to disk for later analysis.

The 576 can also be used to input data directly into Macintosh applications, such as Excel or Microsoft Works, without the aid of dedicated acquisition software. This process requires using the MacDA488 Desk Accessory software in addition to MacDriver488. Both are bundled with the various interfaces. This system can write data directly to an Excel spreadsheet, for example, where it is immediately available for analysis.

Icon-Driven Approach

Icon-based software for the Macintosh has been very popular because it enables running applications graphically, without programming. "WorkBench", from Strawberry Tree, is an example of icon-based data acquisition software. Workbench also uses the MacDriver488 software for communications with the 576.

Workbench uses an object-oriented interface which requires minimal programming knowledge. The icons provide basic functions normally needed in a programming environment. The icons can be used over and over, and can be interconnected in any way desired. The worksheet can be changed at any time, even while a process is running, and the results can be seen instantly. A simple diagram of a GPIB worksheet is shown in Figure 6-13.

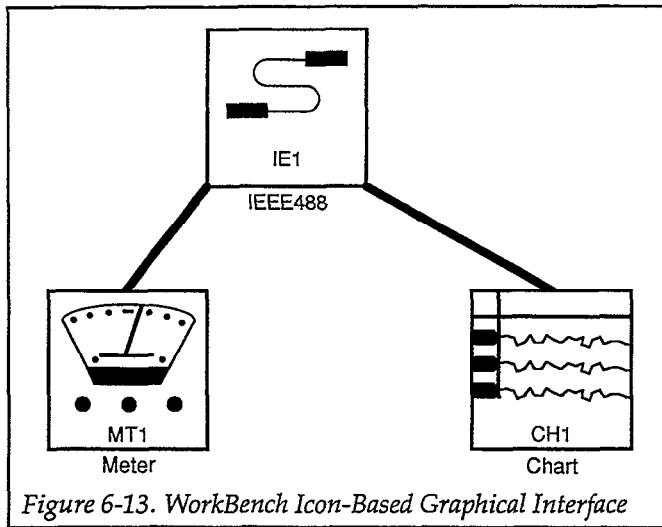


Figure 6-13. WorkBench Icon-Based Graphical Interface

The IE1 ICON contains built-in facilities for GPIB programming. The software does not require the MacDA488

desk accessory software to operate the 576 (see Figure 6-14).

As an example, the following simple example logs a thermocouple using the AIM7 card:

```
OUTPUT 03; READ 3, 0, B0;
OUTPUT 03; BUFF READ TCJ, C, B0; X;
ENTER 03
```

Very little code is required in the normal output mode for the icon. There is an initialization mode which is executed only once. This mode contains necessary initialization programming such as system information and buffer memory allocation.

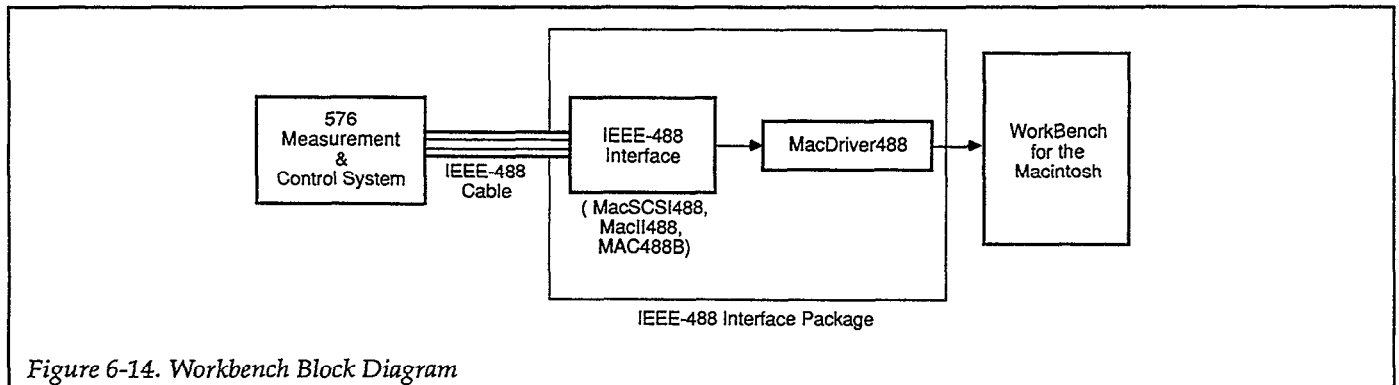


Figure 6-14. Workbench Block Diagram

HIGH SPEED DATA TRANSFERS

The Model 576 can store thousands of readings in its data memory, which is backed up by a battery. When the internal data buffers are full, the data must be transferred to the computer so that the memory can be used to store new measurements. The 576 can transfer data using a variety of data formats and techniques, the most common data format being ASCII. ASCII format also lends itself readily to data manipulation using the tools available in most software packages or programming languages. However, each character of each reading transferred in ASCII requires one byte. Because of this, ASCII files are larger than their binary equivalent, and also take longer to transfer.

Binary data files are more compact than ASCII, and thus require less time to transfer. However, the mechanism of binary transfers is more complex than for ASCII. For example, for a 16-bit analog reading (typically requiring 2 bytes), the binary bytes must be reconstructed in the proper order to produce the corresponding number of A/D counts, which must then be scaled and/or offset to produce meaningful readings. Although the 576 can convert readings to volts, degrees C, or other engineering units "in the box" as they are transmitted, the transfer will be most rapid in unconverted, raw binary format.

Binary Data Formats

In the Model 576, binary data transfers can be performed in either MOTOROLA or INTEL formats. Motorola format sends the most significant byte first, followed by the less significant byte(s). Intel format is the opposite: "low" byte first.

Depending on the type of measurement, a single data point can be returned as a single byte integer, as a WORD integer (2 bytes), or as a LONG integer (4 bytes). Analog data without engineering units applied returns as two bytes. Data subjected to engineering units conversion returns in IEEE single-precision floating point format (4 bytes).

Program Description

The following program illustrates high speed data transfer from the 576 using a binary data format and raw data values.

The program begins by initializing the GPIB bus, and then instructs the 576 to perform a simulated cold boot reset (RESET:ALL). After the 576 is reset, the system identification is requested (SYST:IDN) and printed to the screen. The program then prompts the user for the desired number of points.

Next, the program configures the 576 for the following system conditions: EOI (End or Identify) enabled, data format set to Intel (INTL), and system engineering units set to RAW (no engineering units conversion). The 576 is then commanded to issue an SRQ (Service ReQuest) when the data buffer is full.

Next, a data buffer is dimensioned in the 576 with BUFF DIM to hold the desired number of data points. BUFF DIM uses the default array type specifier "WORD", which is 16 bits. Thus, each reading will be two bytes long.

The ONINT LOGDATA command programs the 576 to execute the subroutine LOGDATA at a 1 millisecond rate. In the subroutine, the READ command is used to read channel 0 on the AMM module in slot 1. While the acquisition runs, a conditional loop executes to check when the data buffer is full.

When the buffer is full, the 576 performs a binary data transfer to an array dimensioned in the host computer's memory. The computer must then process the bytes representing one result to produce a voltage or other type of reading. Thermocouple readings may be returned in binary format, either with or without conversion to degrees applied. In the latter case, each reading must be converted to voltage, linearized, and then converted to temperature.

The next section of the program returns and prints the Binary Header from the 576 data file. All binary data transmitted to the host includes a 6-byte header. This header describes the number of channels, type of data in the buffer, if time stamping was enabled, and the amount of data (in bytes) transmitted from the 576 buffer. Refer to the command section of this manual for details on the format of data transmission.

The last block of the program prints the number of bytes to be returned, followed by the results in binary. The subroutine "PRINTEX" prints the binary numbers from the host computer's data array. The PRINTEX subrou-

SECTION 6
Applications

tine also breaks the binary number down and converts it to hexadecimal values, which are then converted to the corresponding raw count value and printed. Last, the raw count value is converted to volts through the following equation:

$$V = (20 \times \frac{\text{counts}}{65536}) - 10$$

The same equation is used for both the AMM2 and AMM1A. The AMM2 uses a 16 bit A/D converter, while the AMM1A uses a 16-bit A/D in which the four LSB's are hard wired to 0, effectively yielding 12-bit results.

At this point, the voltage reading might be scaled, offset, linearized, or processed in some other way to convert to desired engineering units.

See the 576 Example/Utility Disk for a detailed example of this program.

Example Code:

```
DIM buffer(10000)
' INITIALIZE SYSTEM
RESET ALL; X;
"SYST :IDN?;"          ' Get 576 Identification
RETURN SYSTEM ID      ' Print ID
PROMPT USER FOR NO. OF POINTS TO ACQUIRE
```

```
SYST :EOI ENABLE;      ' Enable EOI
SYST :FORMAT INTL :   ' Set data format
    UNIT RAW;
SYST :SRQ BUFF;       ' Set SRQ on buffer full

BUFF DIM B0, 1, #data points; ' Dimension buffer

ONINT LOGDATA, 1, MSEC; ' Set acquisition rate

SUBR LOGDATA;         ' Define subroutine
    READ 1,0,B0;      ' Read slot 1, channel 0
ENDSUB

DO;
    IF B0 FULL;       ' If buffer B0 is full,
        BUFF READ B0; ' get data.
    ENDIF;

LOOP;                 ' End of loop
X;                   ' Execute program

CHECK SPOLL BYTE "DATA READY BIT TO SEE IF
DATA IS READY TO READ

GET FLOATING POINT BINARY DATA

PRINT BINARY HEADER
PRINT NUMBER OF BYTES RETURNED
PRINT DATA USING PRINTHEX SUBROUTINE

PROMPT USER FOR ANOTHER RUN

END

SUB PRINTHEX
    PRINT BINARY DATA
    COMPUTE HEX, RAW, & VOLTS
    PRINT HEX, RAW, & VOLTS VALUES
END SUB
```

SECTION 7

Reference

INTRODUCTION

In this section, you will find discussions of the 576 mother board. Each module that you purchase with your system comes with a documentation package describing its features, capabilities, and use. This package is three-hole punched, and should be inserted into the 576 Manual in the section reserved for module manuals. Familiarity with this information will allow you to realize the full potential of the 500-series module library. Information of a more technical nature (schematics, calibration information, procedures, etc.) is given at the end of the manual which accompanies each module package.

Prefacing each module reference section is a general introduction to the module, outlining the major features of the hardware. Next, a summary of user-configured components is given, followed by more detailed descriptions of how to make signal connections and how to install and use optional hardware.

Topics pertaining to the use of the module are appended where appropriate. These topics include an introduction to the measurement of thermocouples (in the section covering the AIM7) and a discussion of methods for reducing signal noise (AOM2).

Finally, all the command locations used with the module are listed and then described with appropriate tables that summarize the protocol for using them.

THEORY OF OPERATION

This section contains a description of the operating theory for the various components in the 576. An overall functional description is presented, including a discussion of analog and digital signal paths.

Overall Functional Description

An overall block diagram of the 576 is shown in Figure 7-1. A controller supervises system operation through a GPIB interface.

576 Commands are sent from the controller to the 576 through an IEEE-488 cable. The system control circuitry processes the commands and acts on them. When the 576 performs analog or digital I/O, it decodes the control signals into the various command signals that control operation. The actions of the various commands will depend on the particular module to which they are sent. With the DIM1 and DOM1 modules, for example, commands are used to read or write data bits out of or into the channels on the board. With a PCM module, these commands control latching of data to turn the PCM outputs on or off. With the AOM modules, various commands latch data into DACs (Digital-to-Analog Converters).

Digital commands also control the analog input modules and the analog-to-digital conversion process that transforms analog signals into digital information that can be used by the computer. With the AMM1A or AMM2, for

example, these commands control which slot and channel are selected, as well as the gain applied by the programmable gain amplifier (PGA). These commands are also used to trigger A/D conversion and read the converted data.

Analog input signal processing centers around the master analog input module, as shown in Figure 7-2. In any system with analog input, a master analog measurement module must be present in slot 1.

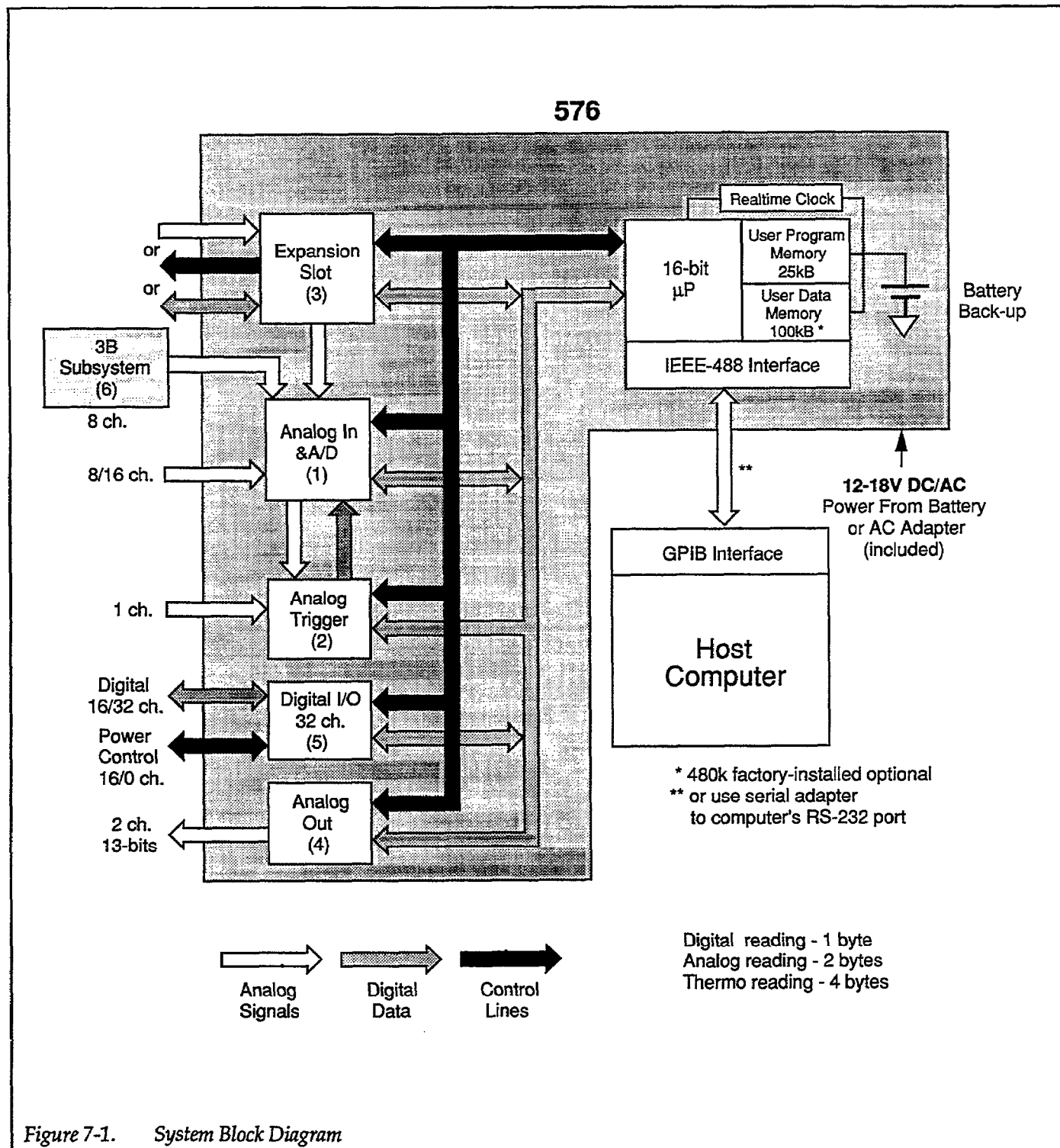


Figure 7-1. System Block Diagram

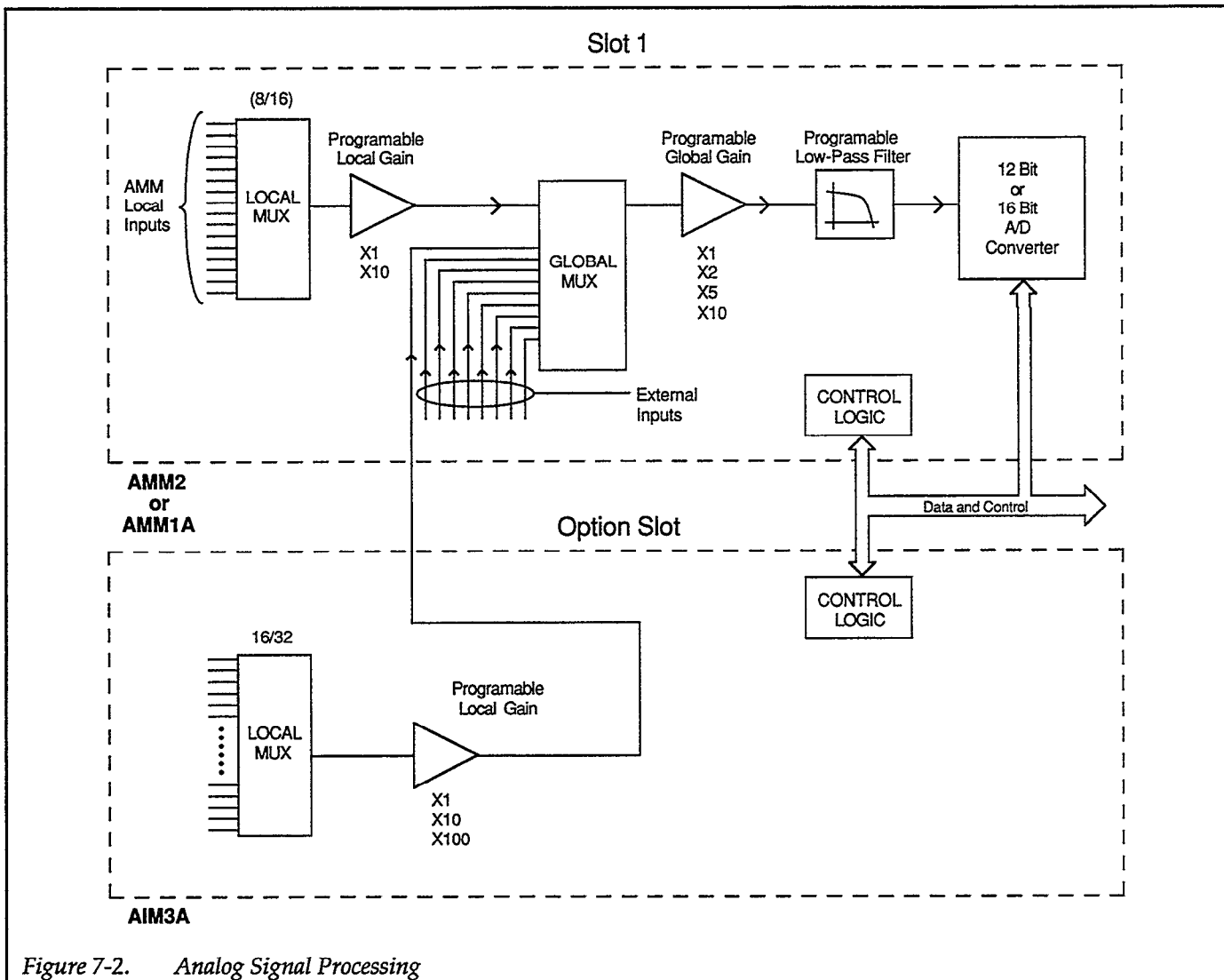


Figure 7-2. Analog Signal Processing

SPECIFICATIONS (1 year, 18°-28 °C):

Analog Input - Volts

Full-scale Ranges: ±100mV, ±200mV, ±500mV, ±1V, ±2V, ±5V, ±10V
 Channels: 8 differential, 16 single-ended plus 8 additional single-ended
 Maximum Input: ±30V (powered), ±15V (unpowered).
 Input Resistance: >100MΩ.
 Input Bias Current: <1nA.
 Input Noise: 576-1: <1 count; 576-2: greater of 6 counts or 50μV p-p.
 Filter: Single-pole low-pass, 100kHz or 2kHz.
 Settling Time : 576-1: 12μs @ 100kHz, 0.6ms @ 2kHz (to 0.05% of final reading).
 576-2: 16μs @ 100kHz, 0.8ms @ 2kHz (to 0.003% of final reading).

Model 576-1 (12-bits, 3-1/2 digit)

Range	Resolution	Accuracy (% of reading + mV)	
		Corrected	Uncorrected
±10V	4.9 mV	±0.04% + 4.9	±0.06% + 4.9
±1V	0.48mV	±0.09% + 0.98	±0.10% + 0.98
±100mV	49 μV	±0.09% + 0.67	±0.16% + 0.67

Model 576-2 (16-bits, 4-1/2 digit)

Range	Resolution	Accuracy (% of reading + mV)	
		Corrected*	Uncorrected*
±10V	305μV	±0.035% + 0.49	±0.065% + 0.66
±1V	31μV	±0.040% + 0.19	±0.065% + 0.19
±100mV	3μV	±0.044% + 0.18	±0.065% + 0.18

*With Reading Average Function Enabled

Analog Input - Thermocouples

(Model 576-2 with AIM7 Thermocouple Input Module, excluding user thermocouple errors):

Channels: 16 differential
 Maximum Input: ±30V (powered), ±15V (unpowered).

Type J	Accuracy (1 yr) 18° - 28° C	Type K	Accuracy (1 yr) 18° - 28° C
-200° to -100°	±1.5°C	-200° to 400°	±1.5°C
-100° to 100°	±0.9°C	400° to 750°	±2.0°C
100° to 400°	±1.5°C	750° to 1200°	±3.0°C
400° to 760°	±2.0°C	1200° to 1370°	±4.0°C

Type T	Accuracy (1 yr) 18° - 28° C	Type E	Accuracy (1 yr) 18° - 28° C
-170° to 0°	±1.5°C	-200° to 0°	±1.4°C
0° to 220°	±1.1°C	0° to 400°	±1.2°C
220° to 400°	±1.4°C	400° to 750°	±2.0°C
		750° to 1000°	±2.0°C

Also supports Type B, R, S transducers. Refer to the Manual for Specifications and extended temperature ranges for all thermocouple types.

Analog Output

Channel Capacity: 2, single-ended, referenced to chassis.
 Output Ranges: ±10V, ±5V, ±2V, ±1V.
 Resolution: 13-bits (12 data plus 1 polarity bit).
 Accuracy: ±10V Range, ±0.15%±5mV; Other Ranges, ±0.2%±4mV
 Output Load: 2kΩ (min), 100pF (max).
 Settling Time: 5μs (max) to 0.01% ± 1 LSB for any step size.

Digital Input/Output & Power Control

Channels: 32 non-isolated, programmable for input or output in groups of 8 channels. 16 channels can drive PCM3 power control rack.
 Input: TTL-level, high-true, 20μA source, 0.4mA sink.
 Output Drive Capability: 10 TTL loads, 24mA sink at 0.5V.

Trigger Function

Channels: 1, differential input.
 Trigger Source: Ext. input, any input channel, or software strobe.
 Ranges: 0 to +1V, 0 to -1V, 0 to +10V or 0 to -10V, 8-bit resolution.
 Level Accuracy: ±2% + 1LSB.
 Input Impedance: 10 MΩ (External Input).
 Input Protection: ±30V max (powered), ±15V max (unpowered).
 Input Coupling: AC or DC
 Trigger Slope: Rising or falling slope.
 Trigger Input Filter: Software selectable single-pole low-pass filter with cut-off frequencies of 1MHz, 300kHz, 100kHz, 30kHz, 10kHz, 3kHz, 1kHz, 300Hz

General

Architecture: Plug-in A/D plus one expansion slot. Accepts one module for additional I/O channels or signal conditioning.
 Program Storage: Up to 10,000 bytes plus 18,000 bytes for system parameters.
 Data Storage: Up to 100,000 bytes available for use in up to twenty (20) user sized buffers. Expandable to 480,000 bytes with 576-MEM factory installed option.
 Battery Life: 128k CMOS RAM: 10 yrs; 512k CMOS RAM: 2 yrs. @ 25°C.
 Clock Accuracy: ±1 minute/month @ 25°C.
 Initialization: Digital Outputs power-up in tri-state mode. Analog Output at 0V.
 Front Panel: REMOTE, SRQ, TALK, RUN, POWER indicators. Power switch.
 Rear Panel: Power input, IEEE-488 connector, and grounding posts.
 Case: Rugged aluminium.
 Warm-up: 15 minutes to rated accuracy.
 Environment: Operating: 0° to 50°C, 80% R.H. non-condensing at up to 35°C.
 Storage: -20°C to +65°C.
 Power: 12-18V 40VA (max) AC or DC. External power module supplied for 105-125VAC input. Order /E option for 205-230VAC mains operation.
 Dimensions: 85 cm (H) x 270 cm (W) x 302 cm (D) [3.5 in. x 10.5 in. x 12 in.].
 Weight: Net: 3.0kg (6.5 lbs). Shipping: 15kg (33 lbs).
 Certification: Meets FCC PART 15J, Class A.

IEEE-488 Bus Implementation

Interface: IEEE-488-1978 standard.
 Multiline Commands: DCL, SDC, GET, GTL, UNT, UNL, SPE, SPD.
 Uniline Commands: IFC, REN, EOI, SRQ, ATN.
 Interface Functions: SH1, AH1, T6, TE0, L4, LE0, SR1, RL0, PP0, DC1, DT1, E1
 Progr. Parameters: Read, Write, Buffer size, Filter, Gain, Offset, Range, Units, Loop, If...Else, While, Time, Time Stamp, Clock, Subroutine, On Interrupt, Trigger, Wait, Test, Peek/Poke, Reset, Halt, SRQ, ID, Terminator, Save, EOI, Cal

Typical Rates

Input/Output (channels/second)	Immediate Mode*		Buffered Mode†	
	1 channel	16 channels	1 channel	16 channels
DC Volt Input [576-1]	26	40	62,500	34,500
[576-2]	26	40	50,000	34,500
Thermocouple Input (all Types)	15	25	215	400
Analog Output (2 channels only)	55	85	40,500	54,000
Digital Input (Port Read)	55	790	90,100	125,000
Digital Output (Port Write)	49	860	72,750	108,000

* Command Sent from computer. Input data converted to ASCII and read for transfer to computer.
 † For 1,000 scans executed from the Model 576 program memory. Input data saved in internal memory.

Source-Delay-Measure Cycle Time Time to output buffer point, Delay=0, Input Data and increment output to next value. 250 μs

Data Transfer Rate (to ideal listener from Model 576 memory):

(in readings/second)	ASCII (no prefix)	IBM Binary	HP Binary
Raw A/D Counts	250	34,000	47,000
DC Volts	36	260	265
Thermocouple Input (all Types)	25	52	53

Maximum GPIB Transfer Rate: 112,500 bytes/second

Theory of Operation

In the sections that follow, you will find a discussion of the Model 576 system control circuitry, power supply, mother board, including theories of operation and command locations (addresses). Command locations are generally of interest only to users who want to access the 576 on a low level, such as when performing diagnostics or writing custom software.

System mother board information includes an explanation on how the Model 576 system controller circuitry operates.

Hardware information pertaining to the operating functions of the Model 576 is broken down into the following sections (see Figure 7-1):

1. Analog Input and A/D Conversion
2. Trigger Control
3. Optional Modules
4. Analog Output
5. Digital Input and Output and Relay Control
6. External Input Function

System Controller Circuitry

Refer to schematic diagram 576-106 page 7 for the following discussion.

The 576 is internally controlled by U301, an MC68008 microprocessor.

IEEE-488 communications are performed by U307, a 9914A GPIB adapter IC. U307 is connected to J301, a 24 pin IEEE connector through U305 and U306, two buffers.

IEEE-488 address setting is done on SW301. U311, a 74LS240 tri-state buffer allows the microprocessor to read the address switches.

Sample timing is a function of U302, a HD63B40 programmable timer. The time interval is programmed into U302, at which point it is able to signal the microprocessor of the passing of the interval by generating interrupts.

The firmware which operates U301 is located in EPROM U303.

The U301's scratch memory, along with 100K of user buffer space, and 5K of user program space is located in U304, a 128K × 8 CMOS static RAM. RAM can be expanded to 512K × 8.

Battery power to the mother board RAM along with a real time clock is provided by the socket which U304 is plugged into.

Setting the 576 in IEEE-488 REMOTE mode causes indicator D301 to light. Likewise, TALK mode causes D303 to light, and a 576 Service ReQuest (SRQ) causes D302 to light.

As the 576 accesses the data acquisition I/O circuitry, RUN indicator D304 will light.

Power Supply

Components on the 576 mother board are powered by +5 volt and ±15 volt power supplies. Power is supplied by a wall mount transformer or automotive power adapter (12-18VDC, 40 VA max). This versatility results from the use of a switching power supply located on the mother board, and a full wave bridge rectifier located on the side board.

The power consumed by the power supply remains nearly constant across the specified supply range for a particular 576 configuration. This means that less supply current is drawn at higher supply voltages. The Model 576 requires 10-40 VA.

Refer to schematic drawing number 576-106, page 2, for the following discussion:

Power Pack/Automotive Adapter Operation

The power pack or automotive adapter plugs in to J101 on the rear of the 576 cabinet. The bridge rectifier formed by D301 through D304 provides rectification of the AC input from the power pack, or passes the DC from the automotive adapter or DC source through one half of the bridge.

The unregulated voltage is then applied to the input of the switching regulator U1 (LT1070CT). U1 operates as a pulse-width modulated flyback switching regulator, and outputs a square wave with varying duty cycle to the T1 primary. Diodes D3 and D5 rectify the output of T1. After filtering by a pi-type LC filter, linear post-regulators U3

and U4 provide a +15 and -15 volt supply to all of the analog circuitry in the system 576.

CAUTION

The outer connector of J101 is connected directly to the 576 chassis. Use caution when placing the 576 on or near an electrically conductive surface.

Power Indicator and Reset Circuit

Components U2, U101D, U101E, U106C and associated components function as a combined power-up reset circuit and power supply test circuit. U2 is a 2.5V reference. The switching supply +5V output is reduced by voltage divider R5/R6 and compared to U2's output by U106C. When the power supply output exceeds 4.4V, the output of U106C goes to a high impedance state, allowing capacitor C4 charges through R17. When the voltage on C4 reaches about 2V, Schmitt trigger gate U101E switches low, turning on the power indicator D6 and releasing the RESET line. If the power supply cannot start up or is overloaded, the power supply +5 level will be below 4.4V, keeping D6 off and asserting RESET low.

MODEL 576 MOTHER BOARD

The mother board, shown in Figure 7-3, contains circuitry for trigger operation, digital input, digital output, and relay control. The mother board also includes the system control circuitry which is responsible for the generation of commands.

The required control and data processing signals are fed through the interface cable to the mother board. The system control circuitry further decodes the control signals into the various command signals that control operation. The purpose of the various commands will depend on the particular operating function. With the digital input, for example, commands are used to read data bits out of the channel. Similarly, these commands control latching of data to turn the PCM outputs on or off. With the analog output section, data is latched into DACs (Digital to Analog Converters). For the analog triggering circuitry, these commands control latching of data into trigger level and sense circuitry and the selection of various triggering modes.

Digital commands also control the analog input and the analog-to-digital conversion process that transforms analog signals into digital information that can be used by the computer. Analog to digital conversion is controlled by the particular Analog Master Measurement module (AMM) that is installed in slot 1 of the side board. See the AMM manual for specific information on the AMM module.

Two option slots are located on the side board assembly which is mounted vertically to the right edge of the mother board. Slot 1 is used for the AMM1A or AMM2 module. The other, slot 3, is used for optional modules. If analog input is not required, both slots can be used for motion control, analog output, or other digital applications. See Table 7-1 for slot functions and types.

The side board, shown in Figure 7-4, contains a full-wave bridge rectifier for the AC transformer (which will pass

Table 7-1. Model 576 Slot Assignments

Slot	Type	Function
1	Physical slot	AMM1A or AMM2 analog input module, or other module where analog I/O is not required.
2	Virtual slot	Trigger circuitry
3	Physical slot	Optional signal conditioning module.
4	Virtual slot	Analog output circuitry.
5	Virtual slot	Digital I/O circuitry and relay control.
6	Virtual slot	External input function for Analog Devices 3B or 100% equivalent signal conditioning subsystems. Termination on 576 sideboard for ribbon cable connector.

NOTE: These analog inputs are shown to exist in slot 6 according to the 576's internal configuration. This is for convenience only. The external function actually uses the AMM global analog inputs 3-10 which feed the A/D converter of the AMM module in slot 1. If an analog input module is used in slot 3, only 7 external inputs will be available.

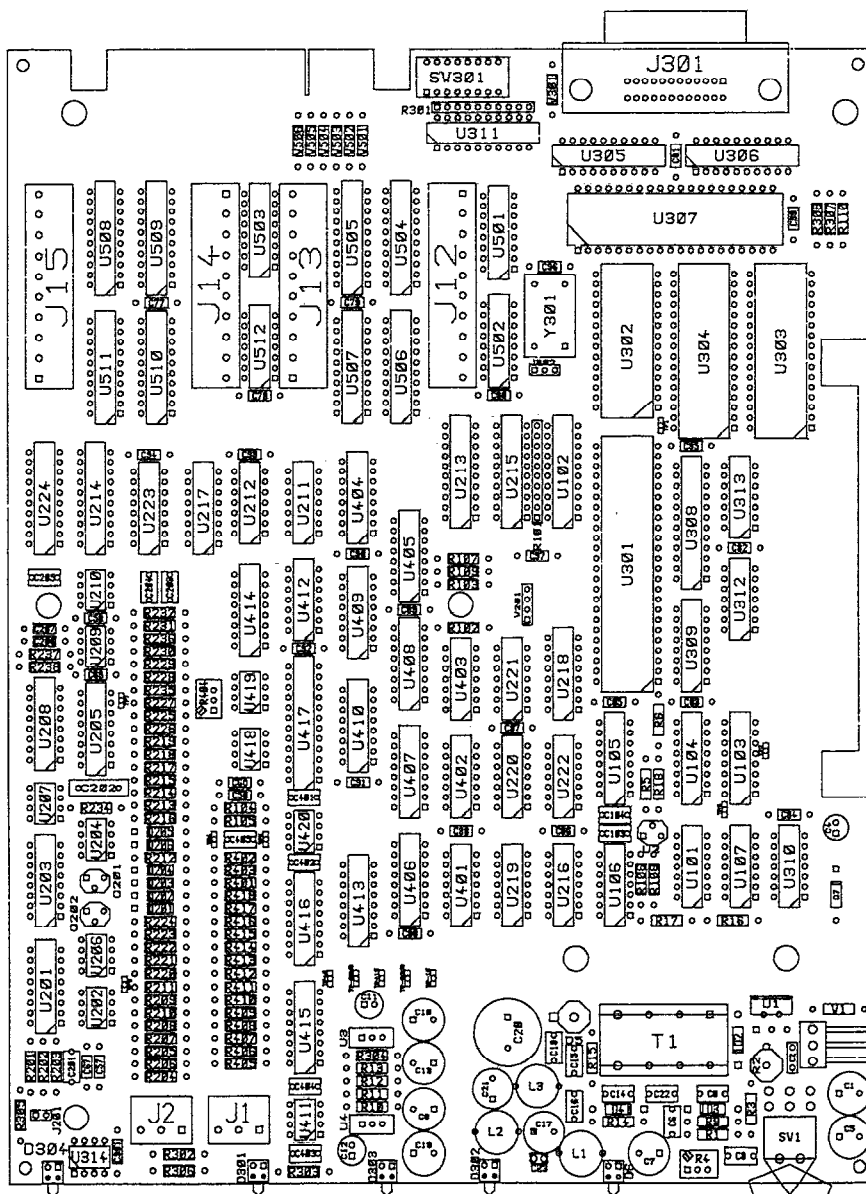


Figure 7-3. System Mother Board

DC from the automotive adapter), a connector to interface an Analog Devices Series 3B modular rack to the 576, and three card edge receptacles (the connector for the mother board, the connector for slot 1 and an AMM module, and the third connector for an optional module (this can be any single module from the Series 500 module catalog, except the STEP modules).

Figure 7-5 illustrates the pinout diagram of the option slot connector. Power for the mother board (+5V, +15V and -15V supplies) comes from the power supply in the 576 which, in turn, comes from the external voltage source.

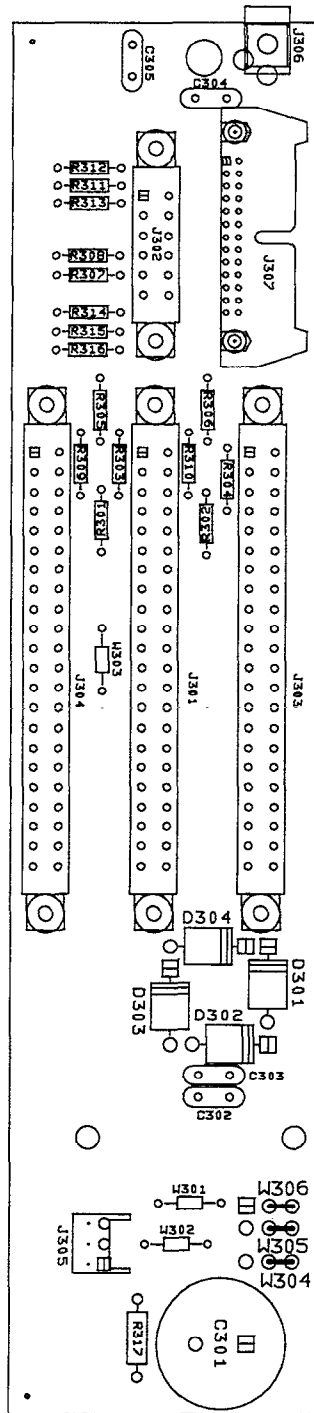
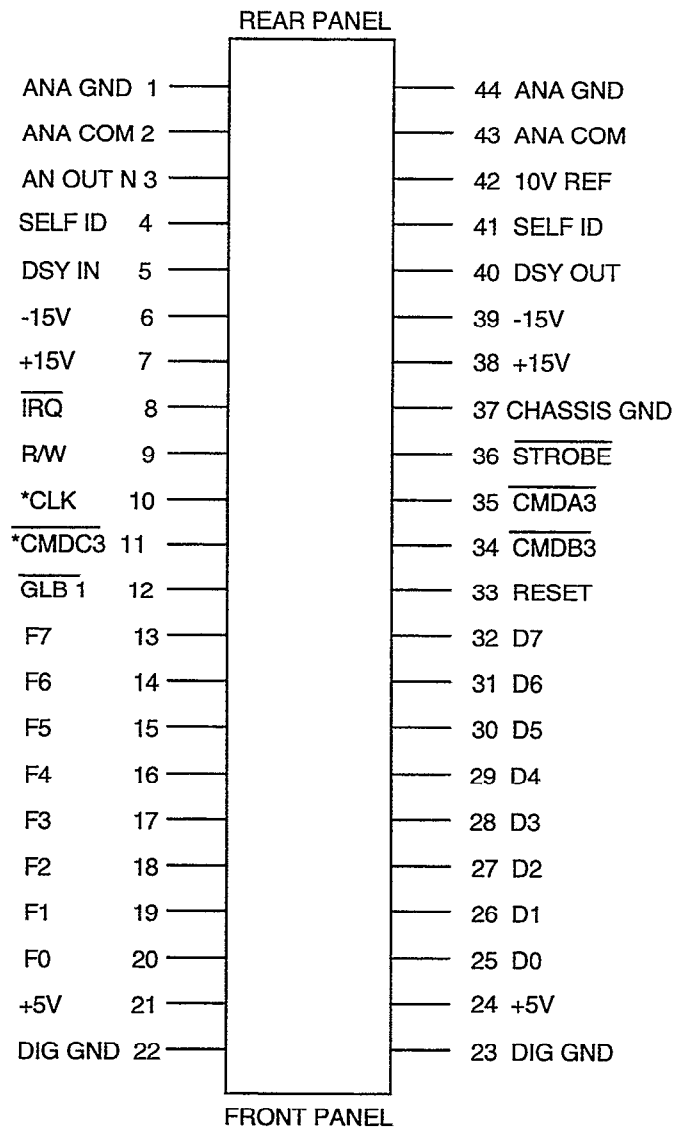


Figure 7-4. System Side Board



* In slot 1 pin 10 carries $\overline{\text{CMDC}}$ and pin 11 carries $\overline{\text{CMDD}}$.
In slot 3, pin 11 carries $\overline{\text{CMDC}}$.

Figure 7-5. Option Slot Pinout

Commands

Commands used by the 500-series modules are keyed to the 576 mother board slots. Each slot of the mother board, whether an actual slot or a virtual slot, is assigned two command locations, CMDA and CMDB, which vary in function depending on which module makes use of them. In addition to CMDA and CMDB, slot 1 has been assigned CMDC and CMDD for use when the AMM1A or AMM2 modules are in the system. Slots 2 and 3 have also been assigned command CMDC.

Besides the command locations associated with the 576 mother board and installed modules, there are a number of locations assigned to Self-ID and other support functions.

Table 7-2 lists the locations of slot dependent commands (CMDA–CMDD where CMD1A indicates command A for slot one, CMD2A indicates command A for slot two, etc.). Detailed descriptions of the commands as they apply to each module are included in the reference sections for the modules.

Mother Board Functions

The mother board-based features and functions of the Model 576 include the hardware trigger, option slot, analog output, digital input and output, power control, and the “external” input function. The following information describes the functions and command locations for each slot in the Model 576. This includes the physical slots 1 and 3, and the virtual slots 2, 4, 5, and 6.

Table 7-2. Mother Board Signal Lines

Signal Name	Description
ANALOG OUTPUT (AN OUT)	A private analog path to the primary multiplexer of the AMM1A or AMM2 module (when either module is installed), used to direct input signal to the measurement modules, and to monitor internal A/D conversion.
COMMAND (CMDA-CMDD)	Locations within the 576 mainframe through which commands from the logic are relayed to the functional modules. CMDA and CMDB apply to all slots. Slot 1 has two additional command locations, CMDC and CMDD, and slots 2 and 3 have a single additional location, CMDC.
DATA (D0-D7) FUNCTION (F0-F7)	Data lines used to transmit or receive 8 bits of data to or from the host computer.
DAISY (DSY IN - DSY OUT)	A special purpose electrical path that transmits information to adjacent slots in a daisy chain format.
GLOBAL (GLB1)	Command location global to the entire system, that changes function according to which module makes use of it.
READ/WRITE (R/W)	A TTL level signal which indicates the direction for data flow in the system. When high, this indicates that a Read cycle is in progress and data is moving from the 576 mother board to the controller.
10V REFERENCE (10V REF)	A reference voltage shared by the conversion components as the primary measurement standard.
STROBE (STROBE)	A third global command used by the analog output module to implement simultaneous updating of several analog output channels.
RESET	A TTL level signal which returns the internal trigger, analog output, and digital I/O lines to power up conditions.

Analog Input

All analog input functions are served by an Analog Master Measurement module AMM installed in slot 1 of the side board. The default analog input range accepts signals up to $\pm 10V$ full scale with $300\mu V$ resolution-per-bit. The AMM1A is actually a 16-bit module in which the four lowest-order bits are permanently set to 0, thus giving 12 bit performance and resolving $4.88mV$ for $\pm 10V$ full-scale. Both modules offer 16 channels of single-ended input or 8 channels of differential input, and a 50kHz AMM2, 62.5kHz AMM1A.

The AMM modules also provides high-speed multiplexing, and local gain amplification of $\times 1$ or $\times 10$. Global gain signal conditioning is provided by a high-speed programmable gain amplifier. The programmable gain ("global") amplifier offers software-controlled gain steps of $\times 1$, $\times 2$, $\times 5$, and $\times 10$.

CAUTION

Always turn off system power before making any connections or adjustments to the Model 576. To minimize the possibility of EMI radiation, never operate the system with the top cover open.

Jumpers, optional resistors, and screw terminals are user-configured components on the AMM modules. DIP headers are provided for the installation of optional resistors between the positive and negative input terminals. With these resistors in place, the analog input can be modified to allow for current to voltage conversion. DIP headers also permit the installation of a resistor from input low or high to ground for each channel. With the appropriate resistors, the analog input can be modified to accept current inputs in the single-ended mode, or provide a return path for bias currents from "floating source" signals in the differential mode.

Two quick-disconnect terminal blocks provide signal connection for all analog inputs. Each terminal block provides eight terminals for signal inputs. Screw terminals accept 16-24 gauge wire stripped 3/16 of an inch. The terminal blocks lift off the mother board to simplify the connection process.

The AMM modules have provisions for a maximum of 16 single-ended input channels or 8 differential input channels. For many applications, differential measurements floated from ground are required; these measurements must be made using the differential mode.

NOTE

When the differential mode is used, noise common to both input lines is reduced due to increased Common-Mode Rejection.

CAUTION

To minimize the possibility of EMI radiation, use shielded cable for input signals. Connect the shield to module ground, but do not connect the opposite end of the shield to anything. Maximum input voltage is $\pm 15V$. If any input exceeds $\pm 10V$, all inputs will be inoperative.

Table 7-3. 576 User Accessible Memory Map I/O Address Space

ID	Reads the contents at the self ID address location	
GLOBAL	Reads the contents at the GLOBAL 1 location.	
RESET	Reads the contents at the RESET location, (aka GLOBAL 2).	
STROBE	Reads the contents at the GLOBAL STROBE location.	
SLOT	FUNCTION	COMMAND
1	option slot 1	A B C D
2	TRG1	A B C
3	option slot 2	A B C
4	AOM5	A B
5	DIO1A	A B

Gain

The AMM module contains circuitry which allows you to apply a software-programmed global gain. This gain will affect any signal which is connected to the analog input of the Model 576 (including signals connected to an optional AIM module). This is because incoming signals are routed through the programmable gain amplifier before they are sent to the A/D converter. Since this is a programmable amplifier, different gain can be programmed for each measurement channel.

Programmable global gain is available in four steps: x1, x2, x5, and x10. To apply a particular gain, you must write the appropriate value to the GLOBAL GAIN command location. See the discussion of the GLOBAL GAIN command for more information.

Programmable Filter

The AMM modules include a programmable filter immediately before the A/D sample-and-hold input. The filter is a single-pole type with cutoff frequencies of 2kHz and 100kHz. The 100kHz filter requires a 16 μ S settling time, and prevents high-frequency noise from affecting the A/D conversion. The 2kHz filter requires a 1 millisecond settling time, and reduces noise that may have been picked up with the signal. It is especially useful with higher gains, where it reduces thermal noise by a factor of 5.

Analog-to-Digital Conversion

Analog-to-Digital Conversion is the final element of the multiplexed analog input subsystem. The analog-to-digital converter receives conditioned signals from all analog input channels via the global circuitry.

The A/D converter offers fast, accurate measurement and digitization. A conversion time of 16 μ S for the AMM2, and a sample and hold acquisition time of only 4 μ S allows sampling speeds as high as 50 kHz. The

AMM1A requires 12 μ s + 4 μ s, for a total of 16 μ s and a corresponding speed of 62.5kHz.

To take full advantage of the converter's resolution, ranges of ± 10 V and 0-10V can be selected through software.

When programming high-speed sampling sequences, certain timing constraints of the A/D conversion cycle should be kept in mind. When the A/D START COMMAND is issued, the converter begins immediately to assess the value of the signal, a process which takes 12 μ s or 16 μ s to complete. During this time, the sample and hold remains in the hold mode, freezing the signal for the duration of the conversion. When the conversion is complete, new data is made available for reading, and the sample and hold returns automatically to the sample mode and begins to track the signal again.

If the signal has changed significantly since the beginning of the last conversion, the sample and hold requires some time to adjust to the new voltage. This period is called the "acquisition time" of the sample and hold, and is 4 μ S. Thus, a new conversion cannot be started for at least 4 μ S following the completion of the last conversion.

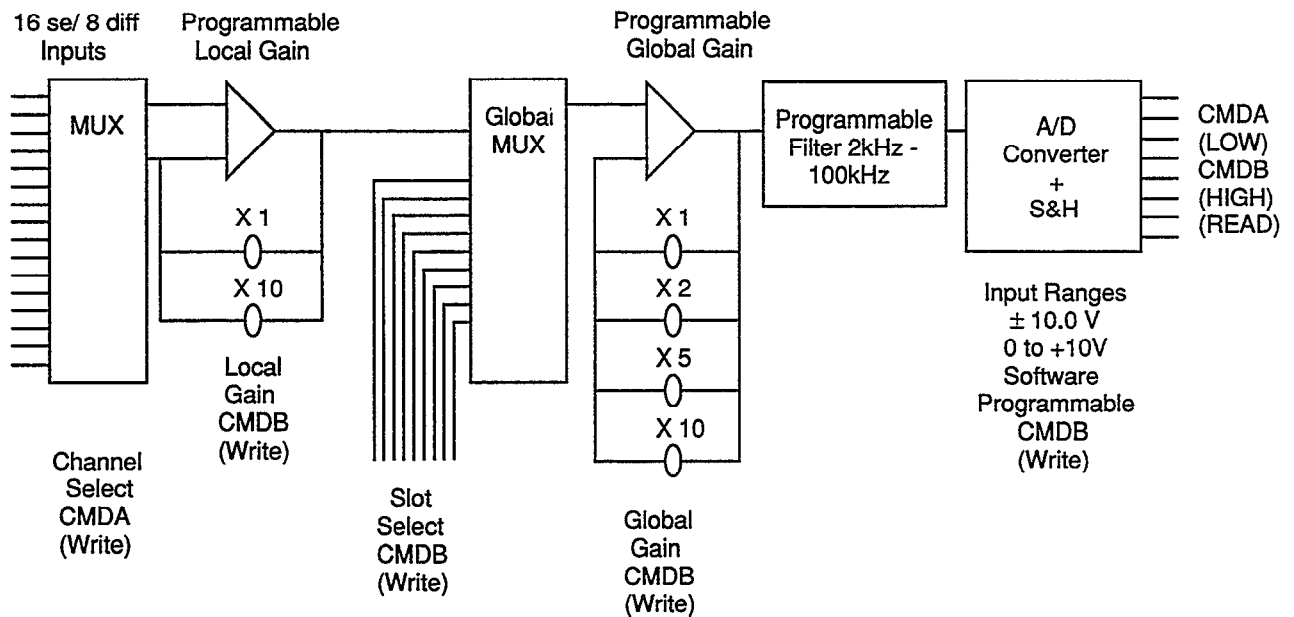
To increase system throughput, a data latch has been provided which makes data from the last conversion available while the converter is busy with another conversion. The data is refreshed (updated) as soon as the converter has finished its current conversion cycle.

Analog Input Command Locations

The commands associated with Slot 1 are Commands A, B, C, and D (CMDA1, CMDB1, CMDC1, and CMDD1). Since slot 1 is normally associated with the AMM1A and AMM2 modules, the following information concerns those modules. Note, however, that if slot 1 is used for any other module, the applicable command functions are determined by that module. See the following chart and pages for detailed information. You may also consult the AMM1A or AMM2 module manuals for details.

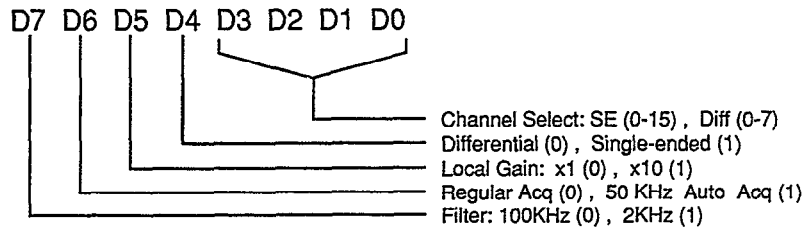
Table 7-4. Slot 1 (AMM) Command Locations and Functions

	COMMAND	FUNCTION
Read Functions:	CMDA1	(D4=1) Read low data byte. (D4=0) Read status.
	CMDB1	Read high data byte.
	CMDC1	Not used.
	CMDD1	Read Status.
Write Functions:	CMDA1	Select channel, gain, modes, filters.
	CMDB1	Select slot, range, global gain, read mode.
	CMDC1	Reset and recalibrate A/D gain and offset.
	CMDD1	Start conversion

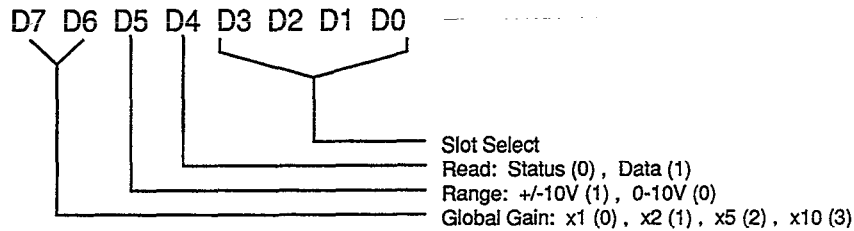


Slot 1 (AMM) Command Locations and Functions Continued

CMDA1 (write) - Select: channel, local gain, filter, plus misc



CMDB1 (write) - Select: slot, range, global, gain, plus misc

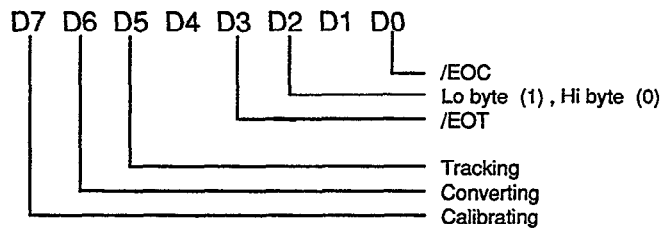


CMDC1 (write) - Reset and recal the A/D (360 mSec).

CMDD1 (write) - Start Conversion

CMDA1 (read) - (D4 = 1) Read Low Data Byte

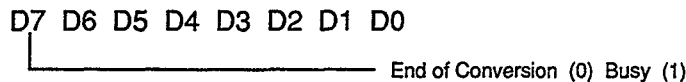
CMDA1 (read) - (D4 = 0) Read Status



CMDB1 (read) - Read High Data Byte

CMDC1 (read) Not Used

CMDD1 (read) - Read status



Analog Trigger

The system 576 contains circuitry for real-time triggering with a wide variety of input and output options. Input ranges are 0 to +1V, 0 to -1V, 0 to +10V, and 0 to -10V with a resolution of 8 bits.

Refer to schematic drawing 576-106, page 4 for the following discussion:

The triggering circuit can be divided into three sections: the command decoding circuitry, the analog input and comparator circuitry, and the trigger/level selection circuitry.

In the command circuitry, input data from the PC bus is buffered by octal latch/buffers U213 and U214 (74LS273). Output data to the bus is buffered out by the octal transparent latch U215 (74LS373). Command and read/write information is decoded by U211, U212, and U216. Input selection is accomplished by a 4-to-2 demultiplexer U223 (74LS153) and quad analog SPST switches U201 and U203 (DG211). The CMDA2 write cycle controls trigger input and configures the trigger and IRQ outputs. The command B2 write cycle controls the selection of filter, range, AC or DC coupling, and triggering edge. The command C2 write cycle latches the trigger level data (in counts) into the D/A converter U208 (AD7523JN) by the octal latch/buffer U224 (74LS273). The command A2 read cycle latches the trigger status information to the PC data bus from U215. The command B2 read cycle also retrieves status information. Additionally, it performs a manual reset of the trigger and IRQ latching circuits. Command C2 read is not implemented.

There are four possibilities for input selection: external analog input (EXT TRIG IN) at J2, global amplifier input (GLOBAL IN) at J201, update from the global strobe, and no trigger input. Input selection is accomplished by electrically switching the input with the quad SPST analog switches U201 and U203; by disabling both inputs and strobing the addressing of the demultiplexer at U223; or deselecting all inputs. Gain selection is performed by switching precision resistors in and out of the feedback circuitry of the dual JFET op amp U202 (LF412CN). Filtering of the input signal is accomplished by a series of precision resistors (R213 through R219), an 8-channel analog demultiplexer U205 (IH6108) and a capacitor (C202). The filter cap is fed by switching in one of the seven (or no) resistors to form a single-pole low pass filter with cut-off

frequencies of 300Hz, 1kHz, 3kHz, 10kHz, 30kHz, 100kHz, 300kHz, and 1 MHz.

Polarity selection is performed by switching the reference voltage on the D/A converter U208 between +10 and -10 volts. Edge selection is performed by demultiplexing the inputs from the dual one shot U217 (74LS221) in the 4-to-2 demultiplexer U218 (74LS153) to separate IRQ and trigger pulses dependent upon the trigger region status. The POS PULSE output from U217A is enabled whenever the input is above the trigger voltage. This 500nsec pulse is switched to the 2Y output of U218 when triggering on the rising edge is selected. Similarly, the NEG PULSE output from U217B is switched to the 2Y output when the input is below the trigger voltage and triggering on the falling edge is selected. The IRQ is handled similarly but is dependent upon the selection of IRQ on trigger start or trigger finished.

For input to the trigger circuit from the AMM global amplifier, a low-loss transmission line is included with the System 576. Since the global output amplifier on the AMM card is capable of amplifying at relatively high frequency, a low capacitance connection is required to minimize distortion of the input to the trigger circuit. For low frequency applications, a regular wired connection would be sufficient. In any case, a wired path must be provided to the global amp input of the trigger circuit (if used) as none is provided on the mother board.

In applications requiring the use of two trigger circuits (the circuit on the mother board and the modularized version of the Series 500 TRG1 which is completely compatible), two jumpers are located on the board in a single 4 pin header identified as W201. The second trigger circuit must be accessed with PEEK and POKE commands.

The trigger circuit, through the use of a power up reset circuit, allows power up in a known condition. Power up conditions are as follows:

No trigger input selected

IRQ and A/D trigger disabled

The trigger is set up for continuous mode on all events of the trigger condition

The 1MHz filter is selected

Trigger is set for falling edge and DC coupling.

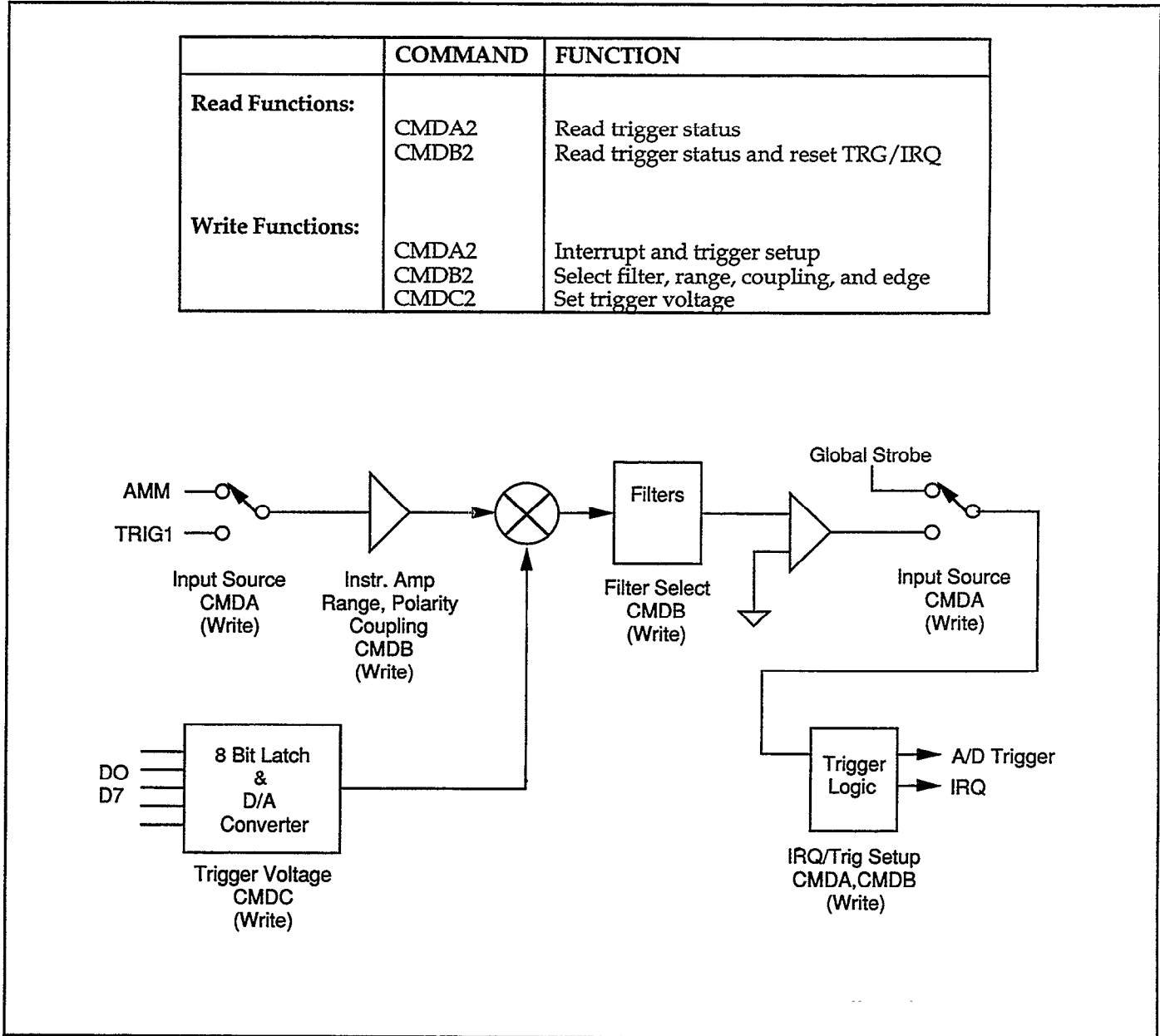
Trigger Command Locations

The commands associated with the slot 2 Trigger function are Commands A, B, and C (CMDA2, CMDB2 and

CMDC2). See the following chart and pages for detailed information.

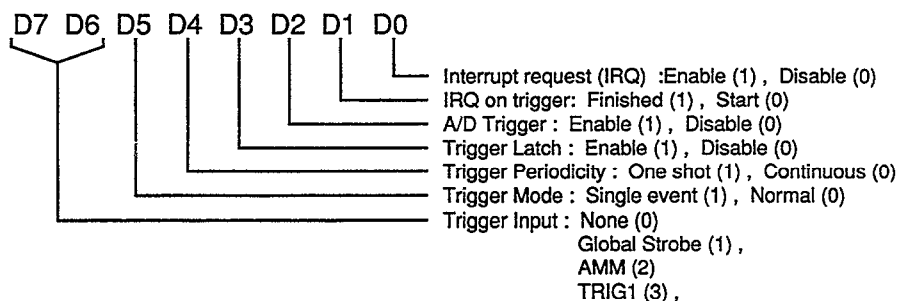
Table 7-5. Slot 2 (TRIGGER) Command Locations and Functions

	COMMAND	FUNCTION
Read Functions:	CMDA2 CMDB2	Read trigger status Read trigger status and reset TRG/IRQ
Write Functions:	CMDA2 CMDB2 CMDC2	Interrupt and trigger setup Select filter, range, coupling, and edge Set trigger voltage

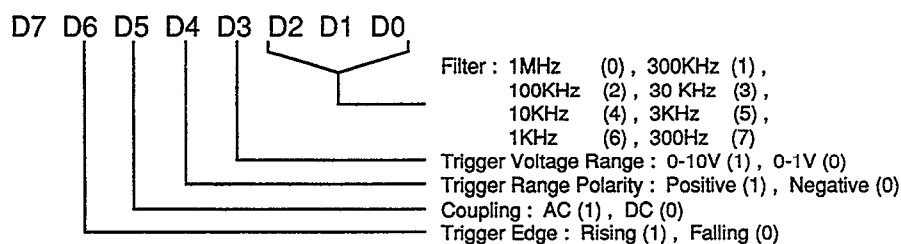


Slot 2 (TRIGGER) Command Locations and Functions (Cont.)

CMDA2 (write) Address 2 - IRQ and Trigger setup

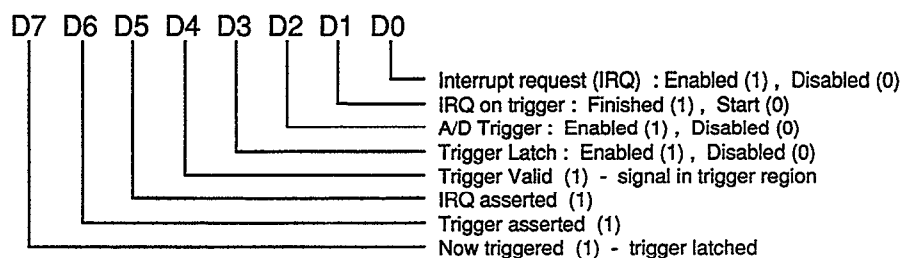


CMDB2 (write) Address 3 - Select Filter, Range, Coupling, Edge



CMDC2 (write) Address 24 - Trigger Voltage (0 - 255)

CMDA2 (read) Address 2 - Read Status



**CMDB2 (read) Address 3 - Reset Trigger and One Event Latch
Data same as CMDA Read Above**

Option Slot

All Keithley modules compatible with the Model 576 are listed in the MODULE section of this manual. Always consider the power consumption of any module you plug into the option slot, especially those which supply excitation to several channels (e.g. AOM4 and AIM8). Some modules require more operating current than others.

Installation

Perform the following procedure to install a compatible Keithley module in the Model 576:

1. Turn off the power to the Model 576.
2. Hold the module with its component side facing upward and its cable clamp toward the rear of the

Model 576. Make sure the rear edge of the module is positioned in the card guide located on the 576 power supply shield. Slide the module into the option slot until it is firmly and evenly seated in the option slot connector. Mount the module to the 576 rear panel using the L-bracket provided with the 576.

3. Close the top cover and turn on the 576 front panel power switch.
4. When programming the Model 576 keep in mind that the module is located in the option slot which is "Slot 3".

Option Slot Command Locations

The functions of the option slot command locations depend on the module installed in the slot. Command lines A, B, and C are available. See the following table and pages for specifics.

Table 7-6. Slot 3 OPTION Command Locations and Functions

	COMMAND	FUNCTION
Read and Write:	CMDA3	See module manual for installed module.
	CMDB3	See module manual for installed module.
	CMDC3	See module manual for installed module.

The option slot (slot 3, directly above the AMM(x) module included with the Model 575 unit) utilizes the following command locations :

CMDA3 (READ OR WRITE) Address 4

CMDB3 (READ OR WRITE) Address 5

CMDC3 (READ OR WRITE) Address 25

Usage of these command locations depends upon the module that is installed in the option slot. Consult the appropriate manual for the command information necessary to the proper operation of optional modules.

Analog Output

The 576 provides two channels of high-speed analog output. Each channel has an independent D/A converter. A system strobe feature, supported by two levels of data latching in the D/A converter, allows both D/A channels to be updated simultaneously.

The D/A converters offer true 13-bit resolution with a maximum nonlinearity of $\pm 0.024\%$. Four output ranges are available for each converter: $\pm 10V$, $\pm 5V$, $\pm 2V$, and $\pm 1V$. These ranges are selected through software. The D/A converters offer true 13-bit resolution plus a sign bit. A sign bit, the MSB of the high-order byte, switches the output of the 12-bit converter either positive or negative. Thus, the effective resolution for a bipolar range is 8192 steps from the negative output limit to the positive output limit. Programming $+0V$ or $\pm 0V$ results in the same output.

Refer to schematic drawing 576-106, page 5, for the following discussion:

The analog output circuitry can be divided into three groups: multiplying D/A conversion circuitry for each channel, command decoding circuitry for each channel, and data buffering circuitry.

The primary components of the D/A conversion circuitry are a single 12-bit dual channel D/A converter (AD7537), reference buffer amplifiers (LF353N), and dual DPDT analog switches (DG423DJ). The dual channel converter contains high speed analog switches, two levels of data latching, and a precision resistor ladder network.

The D/A converter is designated U417 and serves both output channels through the use of output amplifiers U420 and U421 (LF412CN). Output range selection is accomplished by switching precision resistors into the feedback networks of the output amps with quad SPST analog switches U415 and U416 (DG211DJ). Switches are selected by the dual 2 to 4 decoder U413 (74LS139).

The command decoding circuitry is composed of U401 thru U409 and U412. A quad transparent latch U404 (74LS75) stores the 4-bit command select data. A 4 bit presetable counter U405 (74LS163) provides auto sequencing of the write commands for register selection.

Write commands are decoded by U401, U402, U403, and U412.

Selection of polarity is accomplished by switching a $+10$ volt or -10 volt reference to the reference inputs of the D/A converter U417 by the analog switch U414. The D/A output amplifiers U420A and U420B invert the output signal. A negative reference is used to develop a positive output, and vice-versa. A negative polarity is selected when the D7 bit is set in the MSB of the 2 byte D/A data word.

Calibration of the analog output circuit is not necessary other than the calibration of the ± 10 volt reference voltage (described in the Model 576 calibration procedure).

NOTE

The 576 analog output circuitry uses the 10V precision reference of the AMM card, and will not achieve rated accuracy without an AMM module mounted in slot 1.

NOTE

For analog output connections, use shielded cable to minimize the possibility of EMI radiation. Connect the shield to ground. Leave the other end of the shield disconnected.

Output Limitations

There are restrictions as to the output capabilities of each channel. Generally, there is an upper limit on the amount of capacitance and a lower limit to the resistance that can be connected across the output. To avoid possible oscillation, output capacitance must be less than 200pF.

If it is necessary to drive a capacitive load larger than 200pF, a 100 Ω or larger resistor must be placed in series with the output. This will slow down the settling time somewhat, depending on the value of the capacitive load. If an analog output channel must drive a load with both low resistance and high capacitance, the output must be buffered by an external voltage amplifier.

Similar restrictions apply to the output current, which is determined largely by the resistive component of the load connected across the output. If the resistance is too small, accuracy will suffer. To maintain rated accuracy, the load resistance should be no smaller than 2k Ω with a

maximum output of $\pm 10V$. Maximum current output is 5mA or less.

Automatic Register Sequencing

The 576 analog output circuitry includes an automatic incrementing circuit for the analog output range and data registers. The incrementing circuitry aids in high-speed output programming. The following information will be useful for generating analog output by directly accessing the CMDA4 and CMDB4 registers. These operations are normally handled by the 576 firmware.

Generally, standard (non-auto sequenced) analog output is generated by first writing register select information to CMDA4, followed by writing the corresponding data to CMDB4. These steps are repeated until all the necessary range and output data have been written for a desired channel. For channel 0, a typical sequence might be as follows:

1. Write "15" to CMDA4 to select the channel 0 range register.
2. Write the desired range to CMDB4.
3. Write "0" to CMDA4 to indicate that the following data will be analog output low-order byte for channel 0.
4. Write the channel 0 low-order data byte to CMDB4.
5. Write "1" to CMDA4 to indicate that the following data will be analog output high-order byte for channel 0.
6. Write the channel 0 high-order data byte to CMDB4. (Note that bit D7 governs the polarity of the output.)
7. Write to the GLOBAL STROBE location to update the channel 0 output.

Automatic register sequencing automates several of the write operations listed above. It first requires that a control byte be written to CMDA4 (see Table 7-7). This byte must include the register selection and last channel desired for auto sequencing. The most significant bit (MSB) of the byte must be 0 to disable the global strobe function.

Next, data must be written to CMDB4. This data may be range data or the output low-order or high-order data byte, according to the information written to CMDA4. The information written to CMDA4 also sets the "entry point" in the autosequencing loop, thus establishing the expected order of subsequent bytes written to CMDB4.

The auto sequence logic assumes that the next bytes will conform to the following sequence:

Register No.	Description	Sequence
0	Channel0 LSB data	
1	Channel 0 MSB data	
2	Channel 1 LSB data	
3	Channel 1 MSB data	
14	Channel 1 range	
15	Channel 0 range	

Note that entry points in the loop may be range information or output data. As an example, if the initial write to CMDA4 is "14", the analog output circuitry would assume that the next byte is the channel 1 range, followed by the channel 0 range, the channel 0 least significant data byte, the channel 0 most significant data byte, and so on.

Once the sequence moves out of the range registers, it will cycle continuously through the channel registers without returning to registers 14 and 15.

If the first control byte written to CMDA4 is 0, 1, 2, or 3, the auto sequence logic will expect that the next bytes written to CMDB4 will be data. The loop will not enter the range selection registers at all.

If the first control byte written to CMDA4 specifies that channel 0 is the last channel for auto sequencing, then the loop will run only through registers 0 and 1 (channel 0 LSB and MSB data) and not include registers 2 and 3. This path will confine output to channel 0 and permit the maximum output speed from channel 0.

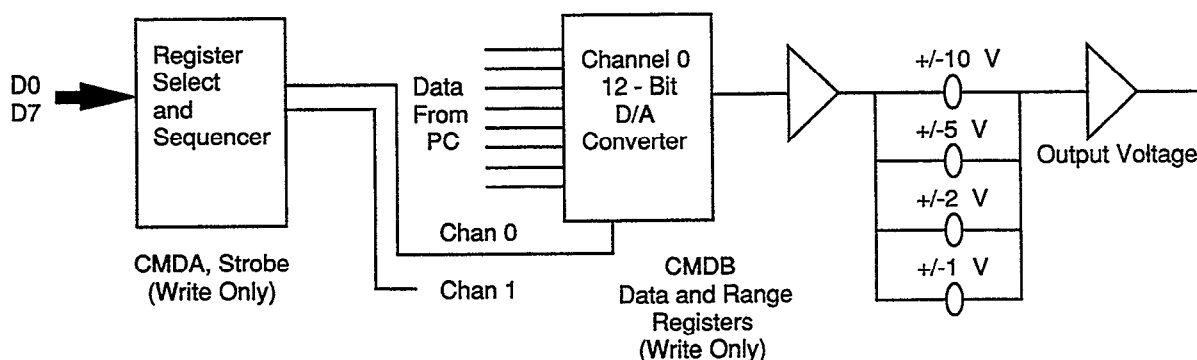
The GLOBAL STROBE must also be disabled for auto sequencing. This enables the output of a channel will be updated as soon as the MSB data for the channel is written to the channel MSB register.

Analog Output Command Locations

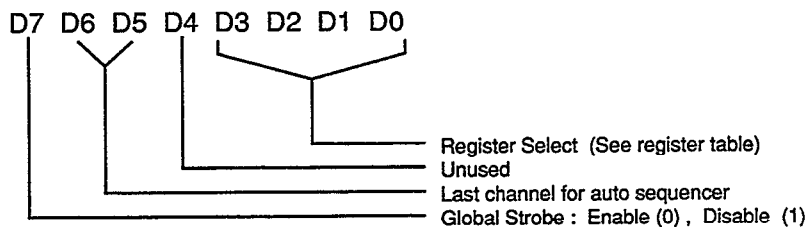
The commands associated with the slot 4 Analog Output function are Commands A and B (CMDA4 and CMDB4), plus the analog output GLOBAL STROBE. There are no read functions for analog output. See the following chart and pages for detailed information.

Table 7-7. Slot 4 (ANALOG OUT) Command Locations and Functions

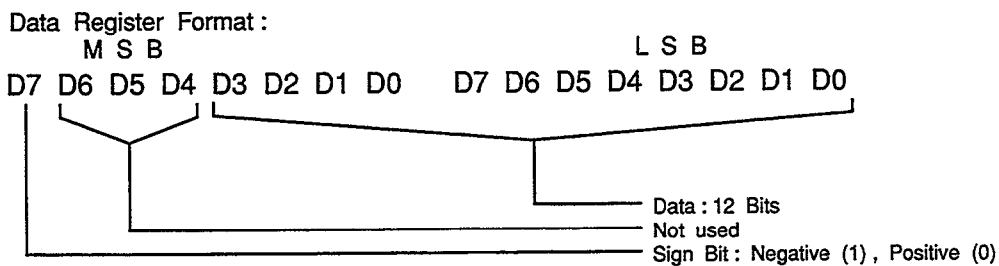
	COMMAND	FUNCTION
Write Function:	CMDA4	D/A control and data register select
	CMDB4	Range and output data STROBE
	CMDC4	Channel update



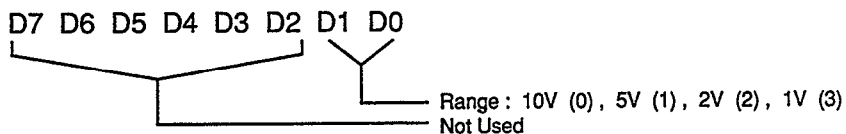
CMDA4 (write) D/A Control.



CMDB4 (write) D/A Data.



Range Register Format :



Slot 4 (ANALOG OUT) Command Locations and Functions (Cont.)

CMDB Write Register Table :

Register Number	Description	Count Sequence
0	Channel 0 LSB	
1	Channel 0 MSB	
2	Channel 1 LSB	
3	Channel 1 MSB	
4	Not Used	
5	Not Used	
6	Not Used	
7	Not Used	
8	Not Used	
9	Not Used	
10	Not Used	
11	Not Used	
12	Not Used	
13	Not Used	
14	Channel 1 Range	
15	Channel 0 Range	

last channel = 0
last channel = 1

STROBE (WRITE) See description.

Global Strobe can be used to simultaneously update all D/A outputs in system. Must be enabled using CMDA write bit D7.

Use of a 12-bit D/A Converter implies that there exist 4096 steps to full scale output. To determine the binary data value corresponding to your desired output voltage, use the following formula: $DATA\ VALUE = ABS(VOUT)/RANGE * 4096$

Note: If desired VOUT is negative, the sign bit in CMDB write for the appropriate channel must be set.

TIMING SUMMARY

the slew rate of the D/A Converter is 5 microseconds (to .01% of full-scale transition).

Digital Input and Output

The Model 576 provides up to 32 single-ended, non-isolated digital I/O channels. The digital channels are software-configured in groups of 8 channels for input or output. Thus, 8, 16, 24, or all 32 channels may be used for input or output.

Digital input channels are grouped into four ports of eight channels. Each port is treated as a single byte from firmware. This grouping allows simultaneous reading or writing of eight channels as a byte.

The Model 576 is designed to read and write TTL-compatible levels. With TTL logic, any input signal less than 0.8V is read as "off" or "low", and any input signal greater than 2.0V is read as "on", or "high". A typical logic "high" output is 3.5V or more.

CAUTION

Digital outputs should only be connected to other TTL-compatible equipment. Shorting a digital output to ground or excessive current draw may damage the Model 576.

Digital Input Terminals

There are four banks of quick-disconnect blocks. Each block has screw terminals for eight digital signals and two grounds. These terminals accept 16-24 gauge wire stripped 3/16 of an inch.

NOTE

To minimize the possibility of EMI radiation, use shielded cable for connections. One end of the shield should be connected to ground, and the other end of the shield should be left disconnected. In this configuration the shield cannot be used as one of the signal-carrying wires.

Refer to schematic diagram 576-106, page 6 for the following discussion:

Digital input and output is possible through four 8-bit software configurable ports. Each port can be configured for input or output individually. The inputs and outputs are for TTL compatible signals only (each input represents a single TTL load, and each output can drive 20 standard TTL loads) and are not optically isolated. All ports power up in a high impedance state configured as inputs through the use of a power on reset circuit.

Command information is decoded and latched by quad latches U502 and U503 (74LS75 and 74LS175, respectively), U412D (74LS02), U512A, U512B (74LS32), and a 3 to 8 line decoder U501 (74LS138). U512A and U512B prevent U503 from changing the port input/output configuration on writes to CMDA5 unless data bit 7 is a "1" at the time of the CMDA write cycle. Writing of data out on the ports is accomplished by latching data from the bus into the tri state octal latches U504, U506, U508, or U510 (74LS244). Reading of data from the ports is accomplished by enabling data into the bus from the octal buffers U505, U507, U509, or U511 (74LS374).

Digital I/O Command Locations

The commands associated with the slot 5 digital I/O function are Commands A and B (CMDA5 and CMDB5).

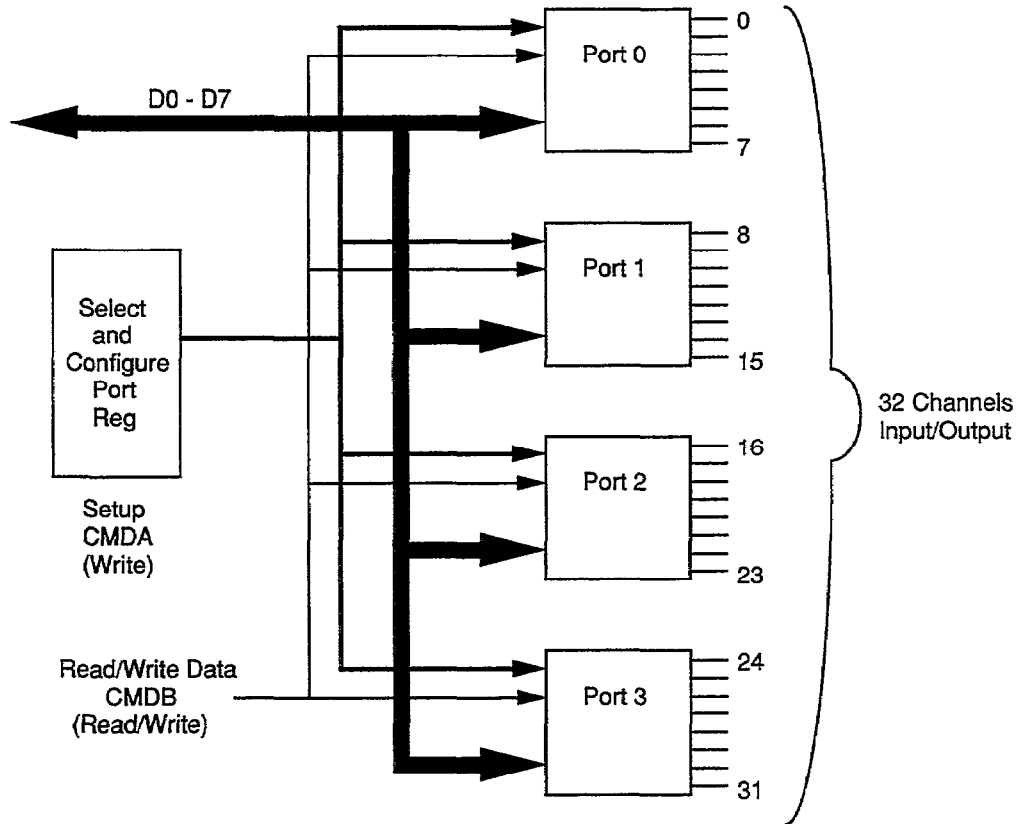
Note that a read or write require that you first configure the circuitry by writing to CMDA5.

Table 7-8. Slot 5 (DIGITAL I/O) Command Locations and Functions

	COMMAND	FUNCTION
Read Function:	CMDA5 CMDB5	N/A Read input data. Before a read, port select and configuration data must be written to CMDA5.
Write Function:	CMDA5 CMDB5	Select and configure port* Digital output data

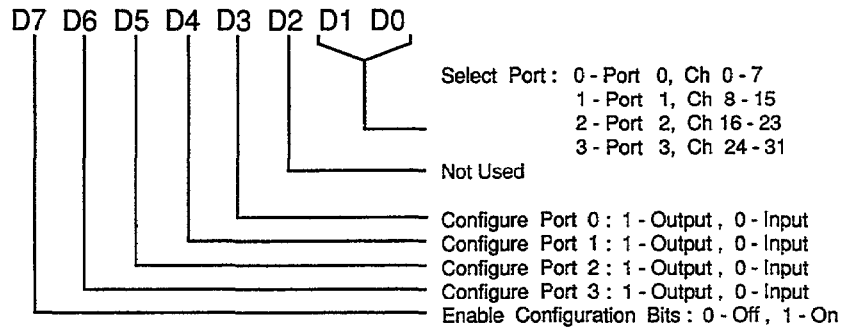
* It is not necessary to send port configuration information to CMDA5 every time a port is selected. If D0=7, then D6, D5, D4, and D3 will be ignored by the hardware, and the last port I/O configuration will remain in effect.

Slot 5 (DIGITAL I/O) Command Locations and Functions (Cont.)



CMDA5 (write) - Select port, configure port

Must be issued in order to read or write data to a specified port.



CMDB5 (read) Digital Input

Read data from specified port provided port is configured as an input port.

CMDB5 (write) Digital Output

--Write data to specified port provided port is configured as an output port.

Relay Control

The Model 576 has a built in power control interface for the purpose of controlling and sensing AC and DC solid-state relays. The power control lines are ports 2 and 3 of the digital I/O. Both ports can be used for power control or power sensing, or one port can be used for each function. Connections to these ports are available at the Relay Control card edge connector located at the rear of the Model 576. The power control channels follow positive logic. Writing a logic 1 to a power control channel causes the corresponding pin on the card edge connector to go to logic 1. Turning on a power control relay requires that its control line be taken to logic 0. Therefore, a logic 0 must be written to a power control channel on the 576 to turn on its respective relay.

To prevent overheating and subsequent damage, all power relay control circuits must be located outside of the Model 576 case. Figure 7-6 identifies the pins of the relay control connector.

Data from the bus is buffered and latched by U509 and U511 (ports 2 and 3 in the digital I/O circuitry) to provide

for relay control. The TTL compatible outputs from U509 and U511 are routed to the relay control card edge connector J16. These outputs have sufficient output current to drive solid state relay output modules, such as the PCM3 Power Control Board/Cable.

Relay Board

The optional Keithley Model PCM3 is a general purpose, 16-channel control subsystem for the control and sensing of AC and DC loads. The PCM3 utilizes solid state switching with plug-in relays for DC and AC output. Each channel is fuse protected. See the PCM3 manual for specifications and installation of the ribbon cable and relays.

Relay Control Command Locations

The commands associated with the slot 5 Digital I/O function are Commands A and B (CMDA5 and CMDB5). See the following chart and pages for detailed information.

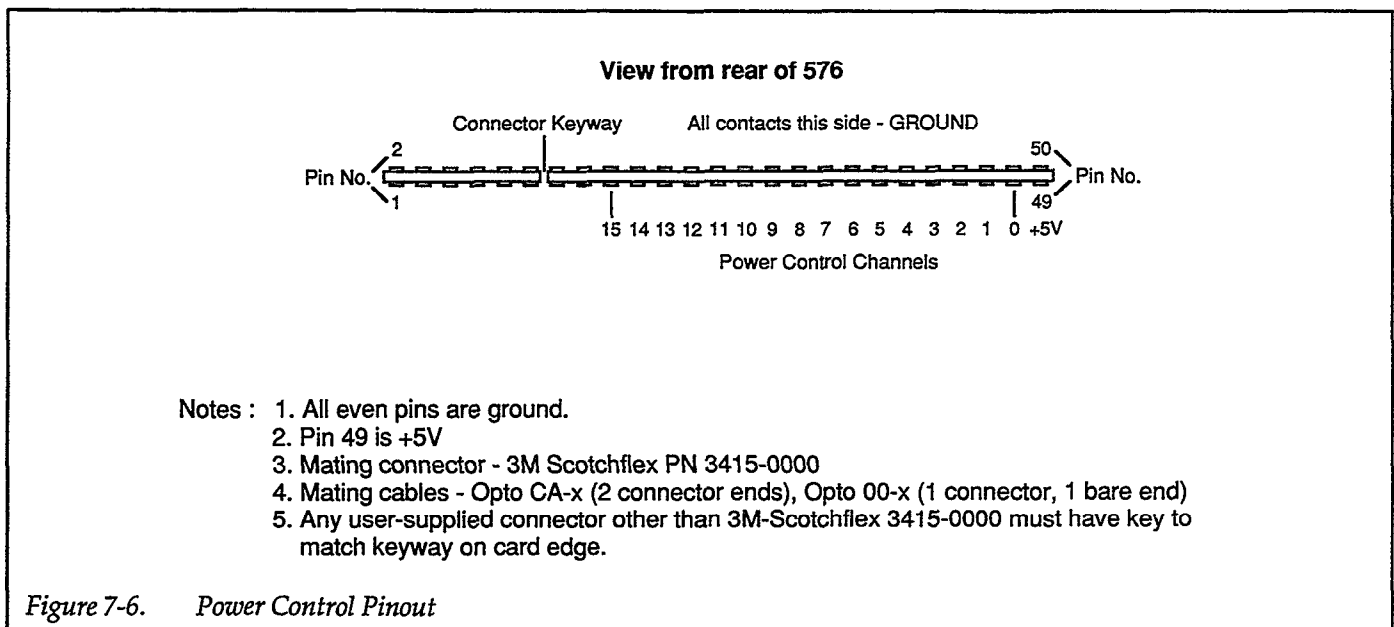


Table 7-9. Slot 5 (RELAY) Command Locations and Functions

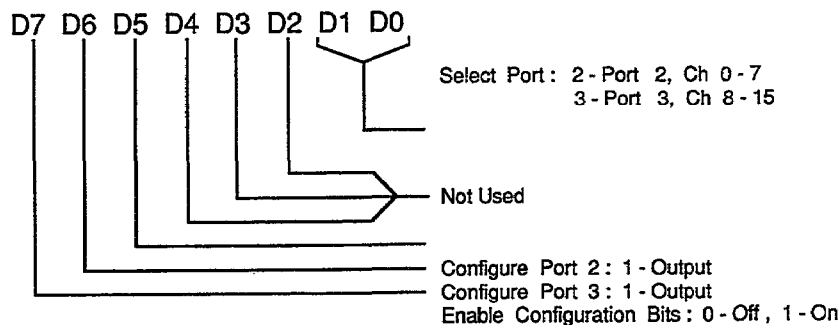
	COMMAND	FUNCTION
Read Function:	CMDA5 CMDB5	Not used Read input data. Before a read, port select and configuration data must be written to CMDA5.
Write Function:	CMDA5 CMDB5	Select and configure port* Digital output data

* It is not necessary to output port configuration information to CMDA5 every time a port is selected. If D0=7, then D6, D5, D4, and D3 will be ignored by the hardware, and the last port I/O configuration will remain in effect.

Relay control is provided through ports 2 and 3 of the Digital I/O circuitry. The commands used to access relay control are the same as those used for the control of the Digital I/O:

CMDA5 (write) - Select and configure Relay Control Port

Must be issued in order to write control words to the relay control ports



CMDB5 (read) - Read status of selected Relay Control Port

Reads status of the 8 control bits currently latched onto the port.

CMDB5 (write) - Write Relay Control Word to selected Port

Latches relay control data onto the port.

External Input

As an extension of its "slot" architecture, the Model 576 will accept input from 8 additional single-ended inputs. The input connector is compatible to an Analog Devices 3B rack. You will note that the 576's internal configuration shows 8 conventional single-ended inputs existing in slot 6. These 8 channels are tied directly to the global multiplexer of the AMM module. Therefore, selection and use of the external function is actually accomplished by reads and writes to the command locations (CMDA, B, C, and D) for slot 1.

There are no commands sent to the Analog Devices 3B rack. Signal flow is limited to those coming to the 576 from the various signal conditioning modules on the 3B rack.

Access to the external channels is via the 26-pin mass termination connector J307 on the side board. Each external input channel is mapped into one of the analog input pathways 3-10 that can be selected by the AMM global multiplexer, and can be accessed by the SELECT SLOT command. Global gain, which is supplied after the multiplexer stage, is available for further conditioning of the external inputs.

The mapping of external channels to analog pathways is as follows:

External Input Channel	AMM Global Multiplexer Input
0	10
1	9
2	8
3	7
4	6
5	5
6	4
7	3

Note that the CH7 input from the 3B rack maps into the Slot 3 input on the AMM module. Therefore, no analog input module can be used in the option slot 3 if the same channel is used for 3B operation. However, this design has the advantage of freeing up all 16 of the inputs channels on the AMM module for other uses. This gives the system the capability of a total of 55 analog inputs — channels 0-6 from the 3B rack, 16 channels on the AMM module, and 32 channels from a Keithley Series 500-AIM3 module installed in the option slot.

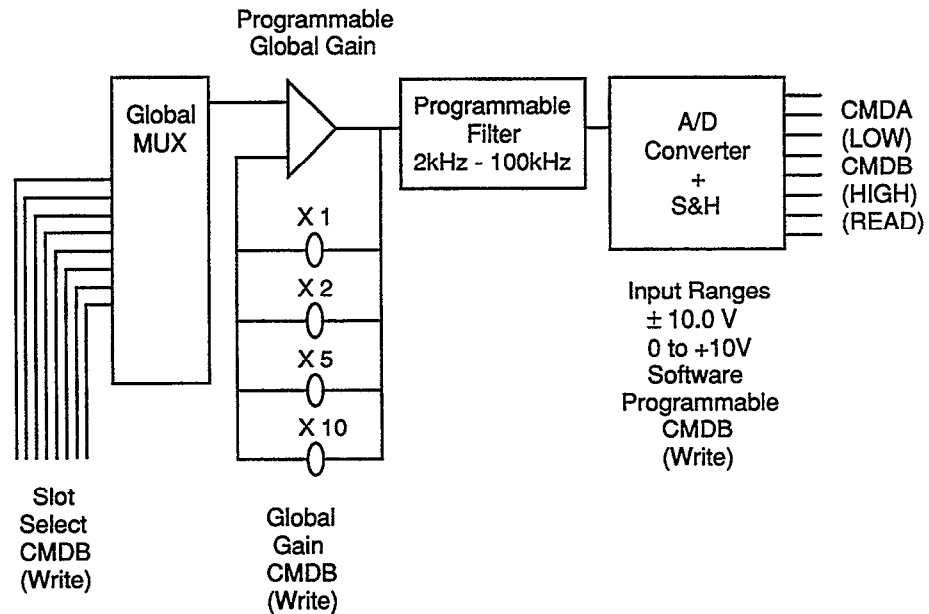
External Analog Input Command Locations

The commands associated with the slot 6 external input function are Commands A and B (CMDA1 and CMDB1). See the following chart and pages for detailed information.

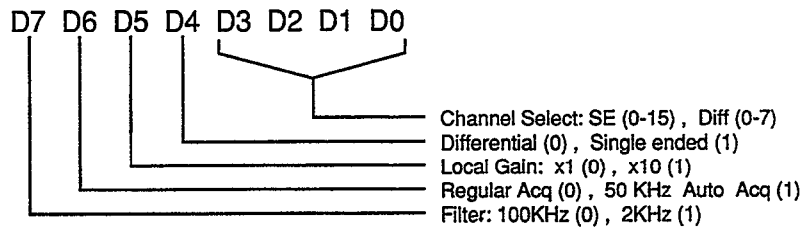
Table 7-10. Slot 6 (EXT) Command Locations and Functions

	COMMAND	FUNCTION
Read Function:	CMDA1	(D4=1) Read low data byte. (D4=0) Read status.
	CMDB1	Read high data byte
	CMDC1	Not used.
	CMDD1	Read status.
Write Function:	CMDA1	Select channel, gain, modes, filters.
	CMDB1	Select slot, range, global gain, read mode.
	CMDC1	Reset and recalibrate A/D gain and offset.
	CMDD1	Start conversion

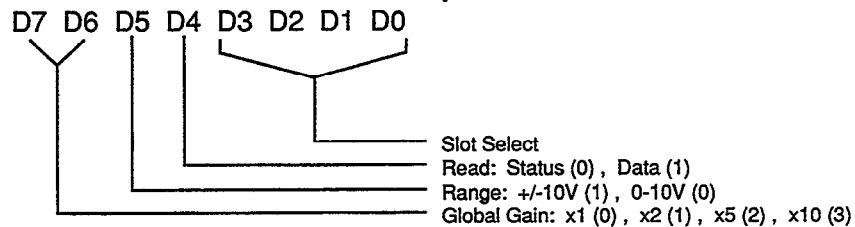
Slot 6 (EXT) Command Locations and Functions (Cont.)



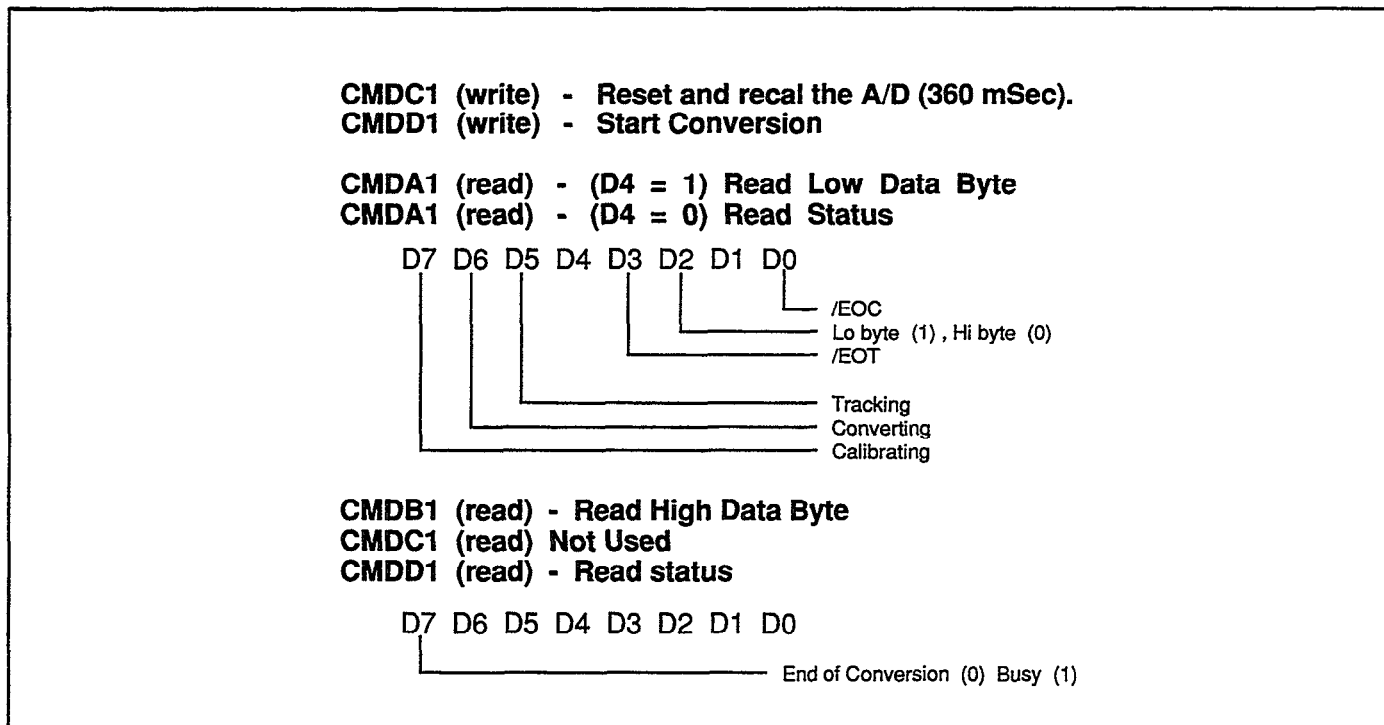
CMDA1 (write) - Select: channel, local gain, filter, plus misc



CMDB1 (write) - Select: slot, range, global, gain, plus misc



Slot 6 (EXT) Command Locations and Functions (Cont.)



Command Locations in Numeric Order

The following information provides more details on the use of command locations in the Model 576.

For a further discussion of analog input commands, refer to the appropriate AMM manual for the module installed in the System 576.

CMD1A SELECT A/D CHANNEL

The SELECT A/D CHANNEL command is used to control the local signal multiplexer on the AMM module installed in slot 1. Refer to the appropriate AMM1A or AMM2 manual for a discussion of this command.

CMD1B SELECT SLOT

The SELECT SLOT command controls the global multiplexer on the AMM module installed in slot 1. Refer to the appropriate AMM manual for a discussion of this command. The select slot is also important in selecting the EXT function for use of Analog Devices 3B subsystems. Specifically, the mass-termination connector on the side board can be cabled to an Analog Devices 3B module

rack, and the 3B rack outputs can be sent directly to the AMM module slot inputs for global multiplexing.

CMD2A ANALOG TRIGGER AND IRQ CONFIGURATION

Writing to this command location configures the interrupt request, trigger mode, and selects the trigger input for the analog trigger circuit. Reading this command location returns the contents of the trigger status register.

Writing to this command location allows the configuration of the input to the analog trigger circuit; selection of the input filter, trigger voltage range, trigger voltage polarity, input coupling, and triggering edge are selected by the appropriate binary data word. Reading this command location also returns the contents of the status register, and additionally performs a reset of the trigger and one event latch if the trigger is configured for one event or single mode triggering.

The triggering and IRQ output can operate in several modes. IRQ can be enabled or disabled, and asserted on either the falling or rising edge of the trigger signal. IRQ can operate even if the trigger is disabled. However, the triggering parameters must still be configured as if triggering were going to be used. Triggering can be enabled

or disabled. The triggering can be latched or automatically reset. Additionally, the triggering can be used in one shot mode, where the trigger pulses briefly when a triggering condition is satisfied; or in a continuous mode where the trigger is latched into an asserted state until the triggering condition is no longer satisfied. Finally, the trigger can be set to trigger as a single event (where a reset must be performed before a trigger can be asserted again) or in a normal mode where the trigger asserts with each entry into the trigger region.

Reading from this location returns a byte that can be interpreted as the TRIGGER STATUS word. This should be used if status information is desired, but no reset of the trigger is to be asserted.

CMD2B ANALOG TRIGGER INPUT CONFIGURATION

The analog trigger circuit produces a trigger signal that can start A/D conversion on the AMM module installed in slot 1. Trigger input can come from any of 3 sources: from asserting a GLOBAL STROBE, from the external analog input (at J2), or from the output of the global amplifier on the AMM module in slot 1. The four possible selections for input include those mentioned above in addition to a no input setting, used at power up of the circuit.

The main feature of the analog triggering circuit is its ability to provide a triggering signal with control similar to that of an oscilloscope. Therefore, the object input signal must be analyzed to determine if it meets a triggered condition. These conditions include input magnitude, input polarity, and whether the signal is on a rising edge or falling edge. These conditions, as well as filtering (8 ranges, from 300 Hz to 1 MHz) and input coupling, are selectable by writing to location 4.

Performing a read of address 4 will return the same TRIGGER STATUS word as reading address 3. However, reading address 4 will perform a reset of the trigger latch when the trigger latch reset bit is set to manual or the trigger mode bit is set to single event in the CONFIGURE IRQ AND TRIGGER command above.

CMD3A OPTION SLOT COMMAND A

This command location is the CMDA location of the option slot. The command specified at this location depends on the optional module that is installed. See the appropri-

ate manual for the module installed in the option slot for the usage of this command.

CMD3B OPTION SLOT COMMAND B

This location is the CMDB location of the option slot. The command specified at this location depends on the optional module that is installed. See the appropriate manual for the module installed in the option slot for the usage of this command.

CMD4A D/A CONTROL FOR ANALOG OUTPUT

Writing to this command location controls the register selection, auto sequencing, and global strobe updating of the D/A converter in the analog output circuitry.

DAC control must precede any change in range register data to reset the register auto-sequencing circuit to the proper register. The lower four bits represent the register to be written to first. Bits D5 and D6 represent the last channel for auto sequencing of the data written to the output data registers (registers 0 through 3). Setting bit D7 enables global strobe (see below) to update analog outputs simultaneously.

CMD4B D/A DATA FOR ANALOG OUTPUT

Through the use of register auto sequencing, the various D/A control registers are filled by writing repeatedly to this location. Range registers are filled first, in descending order from 1 to 0. After filling the range registers, the DAC data bytes are written for each channel, LSB first. The 12-bit DAC requires 2 write operations to supply the 13 bits necessary for data and polarity information. The range registers are only set once, until a write to CMD4A points to the range registers again, and the data registers are continuously updated to allow variable output. When the global strobe update feature is not enabled, the output channel is automatically updated upon receipt of the second byte of data. When the global strobe update feature is enabled, data is not latched into the conversion register of the D/A converter until receipt of the global strobe signal. Only six of the available 16 registers are implemented in this circuitry.

Initially, a D/A control is issued which must select either register 14 (channel 1 range) or register 15 (channel 0 range). Additionally, the D/A control must select the last

channel for auto sequencing, and either enable or disable the global strobe update feature.

After the D/A control is issued, the D/A data is loaded. The command circuitry selects the appropriate range register, and register control is relinquished to the auto-sequencer. The range registers are filled with the proper range data. The auto sequencer drops to the output data registers. D/A output data is written, and the sequencer automatically "points to" the next register to be written to. The data is written LSB first, then MSB, going from channel 0 to channel 1. If the global strobe update feature is disabled (in the D/A control word) the output of the D/A converter is updated immediately upon receipt of the MSB of data (including the polarity bit). If the strobe input is enabled, the data is not latched into the output registers of the D/A converter until receipt of the active low strobe input.

To determine the digital value to input for a given voltage, it is necessary to know the output range setting of the DAC. With the 12-bit DAC, there are 4096 possible voltage levels, specified with digital values of 0-4095. Therefore, the actual full-scale value is the nominal full-scale value minus 1 LSB. This corresponds to a resolution of 1 part in 4096, or about 2.44mV on the 10 volt range. The DAC counts for a particular output are given by:

$$\text{COUNTS} = \text{ABS}[(\text{VOLTS} / \text{RANGE}) \times 4096]$$

where counts = DAC data, volts = desired voltage output, and range = the output range setting for the particular channel. It should be noted that the digital data must be adjusted to include the sign bit (the D7 bit in the MSB of the data). This may be accomplished adding 128 to the MSB if negative voltage output is desired.

CMD5A DIGITAL I/O PORT SELECTION AND CONFIGURATION

Writing to this command location selects and configures the digital input/output ports. Note that ports 2 and 3 are also used for relay control via the PCM connector on the back of the 576 mother board.

The digital input and output ports on the System 576 mother board are fully software programmable for directionality in sets of 8 channels. Port 0 corresponds to channels 0-7; Port 1, channels 8-15; Port 2, channels 16-23; and Port 3, channels 24-31. Each port is completely independ-

ent of the others for these configuration purposes. Generally, more than one write to this location can be used; one to configure all the ports (where the lower 3 bits are insignificant), and the rest to select ports for input or output (in which the upper 6 bits are insignificant). Once the ports are configured, they will remain that way until another write to this command enables the configuration bits and provides new configuration data.

Ports 2 and 3 provide for power control and sensing exterior to the System 576 unit. When power control is desired (using the optional Keithley 500-PCM3 relay rack), configure Ports 2 and 3 as output. For power sensing applications, use the PCM3 with input modules and configure the ports as input ports.

CMD5B DIGITAL I/O DATA

Writing to this command location sends digital data to the port selected by the CMD5A write above. For power control, Writing to ports 2 and 3 sends data to the PCM connector. Reading this location retrieves the digital data from the specified port or reads the current state of the power sensing modules.

For any port configured as an output port and selected by performing the above command, writing to address 9 will latch the data onto the port (any value between 0 and 255). Digital output follows positive logic. When ports 2 and 3 are being used for power control application, the one's complement of the logical output must be written to the port. The PCM3 rack is wired such that a logical 0 must be written to the input to turn on the relay...negative logic.

For any port configured as an input port by the above command, reading address 9 will latch the data at the port onto the data bus (again, a value between 0 and 255). Digital input also follows positive logic; when ports 2 and 3 are being used for power sensing applications, the one's complement of the logical input will be read into the port (the input modules for use as power sensing modules will provide a logic 0 when the module is sensing the rated voltage or current...again, negative logic).

CMD2C ANALOG TRIGGER VOLTAGE (0-255 COUNTS)

Writing to this command location sets the output of the D/A converter in the analog triggering circuit to a volt-

age between 0 and 10 volts, with a resolution of approximately 47mV (1 part in 256). To determine the counts necessary, use the following formula:

$$\text{COUNTS} = \text{ABS}[(\text{VOLTS} / \text{RANGE}) \times 256]$$

where volts = the desired trigger voltage; range = the setting of the range bit (10V or 1V); and counts = number of DAC counts necessary for the desired output (this value is what will be written to the address 24).

Writing to this location sets the absolute magnitude at which triggering will occur in DAC counts. A conversion, based upon selected range and level, will give the proper number of counts (see the command description given above for the formula).

CMD3C UNSPECIFIED OPTION SLOT COMMAND

This command location is the CMD3C location of the option slot. The command specified at this location depends on the optional module that is installed. See the appropriate manual for the installed module for the usage of this command.

CMD1C GLOBAL GAIN

The GLOBAL GAIN command controls the PGA (Programmable Gain Amplifier) located on the AMM module installed in slot 1. For a discussion of this command, refer

CMD1D A/D START/STATUS

Writing to this command location starts A/D conversion on the AMM module installed in slot 1. Any value can be written to trigger conversion; however, a value of 255 should be written to minimize noise. Reading this location returns the status byte of the A/D conversion (busy or ready). A value of 255 (FF in hex) indicates that the conversion process is under way. A value of 127 (7F hex) indicates that the conversion is complete.

SLOT 1 HARDWARE IDENTIFICATION

The Model 576 provides for self identification of the hardware installed in slot 1 to the PC host. Reading this

command location should return a value of 252 (FC hex) if no module is installed, a value of 253 (FD hex) if an AMM1A is installed, or a value of 254 (FE hex) if an AMM2 is installed. Any other value returned from this location indicates that there is an unidentified module installed.

STROBE GLOBAL ANALOG OUTPUT UPDATE

The strobe command is used to synchronously update two or more analog output channels. The strobe feature is global, affecting the D/A channels in the Model 576, as well as the D/A channels in any analog output module installed in the option slot.

Writing to the global strobe command location causes the STROBE line to go active low, and allows global update of all DAC outputs if the analog output circuit is so configured. Global strobe affects any analog output, whether the analog output circuit is on the mother board or an analog module in the option slots.

GLOBAL GLOBAL COMMAND 1

This command location is not currently implemented in the virtual slots of the System 576. However, to allow for future upgrades to modules, the physical slots have this signal available. This command signal is simultaneously active (low at pin 12) at both physical slots when a write or read to the GLOBAL 1 command location is performed.

31 RESET

This command location causes the mother board trigger circuit, mother board analog outputs, and mother board digital I/O to go to its power-up reset condition.

Self-ID Circuitry

An automatic module identification ("Self-ID") feature has been implemented on the latest versions of all current 500-series modules. This feature enables the 576 firmware to automatically determine the presence and slot location of these modules when they are plugged into a 576 mainframe.

Note the following points of compatibility:

1. The AIM1, ADM1, and ADM2 are not compatible with the 576 and do not offer self-ID operation.
2. Self-ID requires Self-ID versions of modules.
3. Some older modules lacking the Self-ID feature require minor modifications to operate in the 576. These modifications can be performed by Keithley Instruments. Contact Keithley Data Acquisition and Control Customer Support for more information. See the section "CONSIDERATIONS FOR OLDER, NON-SELF-ID MODULES" below.

The identifying element on Self-ID modules is a precision resistor connected between card-edge positions 4 and 41 on each module. Circuitry on the 576 mother board reads the voltage drop across these connectors, and deduces the module type. The following values are used for Self-ID:

Considerations for Older, Non-Self-ID Modules

Most older 500-Series modules can be operated in the 576 mainframe without modification. However, minor modifications must be made to some modules to ensure compatibility. In other cases, you must follow special procedures to use older modules in the 576. The following modules are affected:

AIM4, AIM5, AIM6 — Older AIM4, AIM5, and AIM6 modules lacking Self-ID must be reworked for proper "master/slave" operation.

AIM9 — Older AIM9 modules lacking Self-ID must be reworked so that master/slave oscillator functions are compatible.

Please contact Keithley Data Acquisition and Control for more information.

Prolonging the Life of the Clock/Calendar/Memory Backup Battery

The Model 576 contains a battery which is used to power the system clock/calendar, and to back-up data and programs stored in RAM. This battery and clock/calendar

are an internal part of the socket holding the system data/program memory chip U304.

Under normal conditions, the battery should last for a minimum of five years with the standard 128k memory, or 2¹/₂ years with the 512k memory option. Battery life can be extended by using either of the following procedures:

1. Use the SYST :CLOCK DISABLE command to disable the clock. This will reduce but not completely eliminate battery drain, and may be done when the Model 576 is unused for an extended period, or where time/date stamping is not required.
2. Keep the Model 576 powered up at all times. This will eliminate drain on the battery altogether.

Refer to the Calibration, Maintenance, and Troubleshooting section of this manual for instructions on renewing the battery.

576-Series Measurement and Control System Specifications

ACCESSORIES AVAILABLE:

500-AMM1A — AMM1 16-Channel (8 Differential) Master Measurement Module w/12-Bit 62.5kHz A/D

500-AMM2 — AMM2 16-Channel (8 Differential) Master Measurement Module w/16-Bit 50kHz A/D

500-AIM2 — AIM2 32-Channel High Level Analog Input Module

500-AIM3A — AIM3A 32-Channel (16 Differential) High Sensitivity Analog Input Module

500-AIM4 — AIM4 4-Channel Isolated Analog Input Module

500-AIM5 — AIM5 4-Channel Isolated Low Level Analog Input Module

500-AIM6 — AIM6 4-Channel RTD Analog Input Module

500-AIM7 — AIM7 16-Channel Thermocouple Input Module

500-AIM8 — AIM8 4-Channel Strain Gage Analog Input Module

500-AIM9 — AIM9 2-Channel LVDT/RVDT Analog Input Module	500-DOM1 — DOM1 16-Channel Isolated Output Module
500-AOM1/2 — AOM1/2 12-Bit D/A, 2-Channel Analog Output Module	500-EXTND — Module Extender Board
500-AOM1/5 — AOM1/5 12-Bit D/A, 5-Channel Analog Output Module	500-PCM1 — PCM1 4-Channel AC Power Control Module (includes 4 ea. 500-OAC1)
500-AOM2/1 — AOM2/1 16-Bit D/A, 1-Channel High Resolution Analog Output Module	500-PIM1 — PIM1 8-Channel Frequency Measurement Module
500-AOM2/2 — AOM2/2 16-Bit D/A, 2-Channel High Resolution Analog Output Module	500-PIM2 — PIM2 4-Channel Event Counting Module
500-AOM3 — AOM3 4-Channel Current Loop Output Module	500-PROTO — Prototyping Module*
500-AOM4 — AOM4 4-Channel Programmable Excitation Output Module	500-STEP1 — STEP1 μ P Stepper Motor Controller Module*
500-AOM5 — AOM5 4-Channel Analog Output Module	500-STEP2 — STEP2 1-Channel Stepper Motor Indexer Module*
500-DIM1 — DIM1 16-Channel Isolated Digital Input Module	500-TRG1 Hardware Trigger Card*
500-DIO1A — DIO1A 32-Channel TTL Digital Input Output Module	*Not supported by 576 commands, directly, may be accessed through use of PEEK and POKE commands.
	500-DIO1 is also supported in 576.

SECTION 8

Calibration, Maintenance, and Troubleshooting

INTRODUCTION

This section contains information necessary to service your Model 576. The following information is included:

Calibration — instructions for calibrating the Model 576.

Troubleshooting — guidelines for troubleshooting the mainframe.

Replaceable Parts — describes parts which you may need to replace or add to the Model 576.

Some of the information presented is intended for skilled technical personnel who are familiar with sophisticated equipment and the necessary servicing procedures. Do not attempt certain procedures unless you are qualified.

WARNING

Some of the procedures described in this section may expose you to potentially lethal voltages. Use standard safety practices when such dangerous voltages are encountered.

CAUTION

Always follow the indicated procedure exactly as written. Failure to do so may damage equipment, possibly voiding the warranty.

CALIBRATION INFORMATION

This section contains general field calibration information for the Model 576. The procedures given are not necessarily as accurate as factory calibration. Also, the procedures given assume a certain amount of expertise on the part of the user. If you are not familiar with calibrating equipment, do not attempt calibration. The procedures in this section assume that you are familiar with general module operation. Refer to the appropriate manual for details on calibrating each module.

When You Should Calibrate

The Model 576 is calibrated at the factory, and should require no further calibration before use. Calibration is necessary only under the following conditions:

1. If you are performing periodic calibration as part of an established maintenance procedure.
2. If you suspect the Model 576 of faulty or inaccurate operation.

NOTE

If an input or output function which had been working correctly suddenly becomes inaccur-

rate by more than a few percent, the problem is more likely a malfunction and not a calibration problem. If you cannot calibrate the Model 576 after two attempts, you should return it to Keithley for repair or calibration at the factory.

Environmental Conditions

Calibration should be performed at an ambient temperature of $23^{\circ}\text{C} \pm 5$ degrees. Turn on the system power and allow at least 10 minutes of warm up before beginning calibration procedures. Consult the manual for the calibration equipment listed below for similar required warm up periods.

Recommended Calibration Equipment

The following equipment is recommended for Model 576 calibration. Other equipment may be substituted as long as accuracy specifications are at least as good as those given below:

1. Keithley Model 175 DMM ($\pm 0.1\%$ basic DC accuracy). 4-1/2 digit resolution is required for calibrating the 576. Tighter specifications may be required for calibrating an AMM2 module. Refer to the AMM1A or AMM2 manual for AMM calibration instructions.
2. Electro Development Corporation (EDC) Model E100C DC Millivolt Reference Source, or other similar equipment with a basic accuracy of $\pm 0.05\%$.

Complete calibration of the 576 requires calibration of the AMM module, calibration of the analog output $\pm 10\text{V}$ reference, and calibration of the +5V power supply. These steps may be done in any order.

Calibrating the AMM1A or AMM2

The 10V reference on the AMM module is used as a reference for the 576 on-board analog output circuitry. Therefore, you must have a properly functioning and calibrated AMM module in place before you can use the analog output channels.

Calibrating the +5 Volt Supply

1. Ensure power is off on the front panel of the 576.

2. Connect a suitable power source to the external power input. If possible, use the power source which will normally be used to power the 576.
3. Connect the negative lead of the DMM to TP-GND test point and the positive lead of the DMM to the TP+5.
4. Verify that the power supply is fully loaded as in normal operation. The AMM module should be in slot 1, and the option module (if used) should be in slot 3. Any external sensors or loads normally used with the option module should be connected.
5. Turn on the 576 power switch. Verify that the power light comes on. If the power light does not come on, check for the following conditions:
 - A. Check that the power supply is not over loaded. If the power supply cannot handle the initial surge requirements of the 576, or if the external power supply sags to less 12V, the 576 power supply may fail to start up.

If the power light comes on, go to the next step.

6. Adjust R4 to obtain a reading of $5.00\text{V} \pm 0.05\text{V}$.
7. This completes calibration of the +5V power supply.

Calibrating the ± 10 Volt Reference

Normally, calibration of the $\pm 10\text{V}$ reference is necessary only in the case of repairs to the 576. Calibration of the +10V reference on the AMM module will normally correct errors in the analog outputs. See the AMM manual for a calibration of the +10V reference.

NOTE

A properly functioning AMM1A or AMM2 module must be installed in slot 1

1. Confirm that power is turned off at front panel of the 576.
2. Connect the negative (black) lead of DMM to pin 1 of J1 (remove the terminal block to facilitate this connection).
3. Connect the positive (red) lead of DMM to TP4, the +10V reference test point.
4. Apply power. Allow several minutes of warm up time for +10 volt reference on the AMM module to stabilize.
5. Note the DMM reading to the nearest millivolt, for example 10.013V.
6. Connect the positive (red) lead of DMM to pin 1 of J1.
7. Connect the negative (black) lead of DMM to TP5, the $\pm 10\text{V}$ reference test point.

8. Adjust R404 to obtain the same reading that was noted in step 5, within 0.001V.
9. Calibration of the ± 10 volt reference is now complete.

TROUBLESHOOTING INFORMATION

This section contains information necessary to troubleshoot the Model 576. Information is presented on two levels; a procedure designed to aid the typical user in isolating faults to a specific region or board, and more detailed information intended for the skilled technician who has access to electronic test equipment.

If a defective component is found, replacement parts may be obtained from Keithley. If factory service is desired, the Model 576 may be returned for repair. For information on replacement parts or factory service, see the Parts List section of this manual.

Isolating the Problem

The following four symptoms suggest specific system problems:

NOTE

The following checks assume that the PC is functioning properly.

576 power indicator not on.

The power indicator on the front panel of the Model 576 is controlled by circuitry which checks several conditions within the 576. If the power-on indicator does not light, check the following:

1. The 576 front panel power switch is on.
2. The external power supply has sufficient capacity.
3. A power fault, such as a momentary overload, has caused the 576 power supply to shut down. This condition can be reset by switching the 576 off and back on.

As long as the front panel power indicator is off, the 576 internal reset circuitry will hold the microprocessor and all internal registers in the reset state and the 576 will not respond to commands.

System inoperative (POWER indicator turns on, but unit is not able to communicate over the IEEE-488 bus):

1. Check that the IEEE-488 interface address switch is set correctly.
2. Check for a shorted module by removing it from the option slot.
3. Other possible causes include defective interfacing cable, defective ± 15 V supplies, defective controller, or defective logic circuits.

No module works in the option slot:

1. Inadequate power source (see the previous discussions)
2. Check for a bad option slot connector.
3. Check for bad mother board logic circuit.

A particular module is inoperative in the option slot:

1. Inadequate power source (see the previous discussions)
2. The module is defective and should be repaired.

NOTE

For defective modules, refer to the module instruction manual for troubleshooting information.

System Checks

The following troubleshooting information is intended for skilled electronics servicing personnel who are familiar with electronic test equipment. Information is provided to enable troubleshooting to the circuit level. Troubleshooting to the component level is left up to the technician. Use the component layouts and schematic diagrams located at the end of this manual as an aid in troubleshooting. In some cases, information contained in the Theory of Operation section may also be helpful.

The success of any troubleshooting procedure depends on the use of accurate, reliable test equipment. The following equipment is recommended as an aid in troubleshooting the Model 576: 4-1/2 digit DMM with 0.03% basic DC accuracy (Keithley Model 175 or equivalent). A dual-trace triggered-sweep oscilloscope with 25MHz bandwidth will also be necessary in monitoring digital waveforms. In addition, a logic probe may be useful in tracing digital signals.

Power Supply Checks

A good technique is to always check the power supply voltages first, since improper levels could cause partial or complete system failure. Check power supply voltages at the mother board first. The allowable voltage ranges of the supplies are listed in Table 8-1.

Table 8-1. Power Supply Voltage Tolerances

Supply	Acceptable Range
+5V	4.75 to 5.25V
+15V	14.5V to 15.5V
-15V	-14.5V to -15.5V

Mother Board Checks

Power is delivered to the mother board via the wall transformer or automotive adapter. Perform the following procedure to check the power supply voltages at the mother board:

1. Remove the option card if one is installed.
2. Connect the DMM low to chassis ground.
3. Turn on the Model 576.
4. Check the voltage levels at the appropriate test points. The allowable voltage readings are listed in Table 8-1.

Signal Checks

Operation of the Model 576 is supervised by the command signal lines. These logic signals are generated by the MC68008 microprocessor. Integrity of the various command lines can be verified by programming the 576 to enter a loop in which the various command line addresses are poked and then testing the points listed in Table 8-2. The following program demonstrates how to do this with an IBM PC or compatible running Interpretive BASIC and using an IOTech IEEE-488 interface card. The program could be modified to run in other environments.

```

10  *
20  * Open communications with IOTech IEEE-488 in-
    interface card
30  *
40  OPEN "\dev\ieeeeout" FOR OUTPUT AS #1

```

```

50  IOCTL #1, "BREAK"
60  OPEN ""\dev\ieeeein" FOR INPUT AS #2
70  '
80  ' *
90  ' * Read back and display string
100 * from IEEE-488 device driver
110 *
120 PRINT #1, "hello"
130 INPUT #2, AN$
140 PRINT AN$
150 '
160 *
170 * Initialize 576, read back model
180 * ID string, and display string
190 *
200 PRINT #1, "clear"
210 PRINT #1, "output 03; SYS :IDN ?;"
220 PRINT #1, "enter 03"
230 LINE INPUT #2, AN$
240 PRINT AN$
250 '
260 *
270 * Poke command line addresses and loop
280 *
290 GOSUB 600: PRINT #1, "output 03; DO;"
300 GOSUB 600: PRINT #1, "output 03; POKE 1,A,0;"
    POKE 1,A,255;"
310 GOSUB 600: PRINT #1, "output 03; POKE 1,B,0;"
    POKE 1,B,255;"
320 GOSUB 600: PRINT #1, "output 03; POKE 2,A,0;"
    POKE 2,A,255;"
330 GOSUB 600: PRINT #1, "output 03; POKE 2,B,0;"
    POKE 2,B,255;"
340 GOSUB 600: PRINT #1, "output 03; POKE 3,A,0;"
    POKE 3,A,255;"
350 GOSUB 600: PRINT #1, "output 03; POKE 3,B,0;"
    POKE 3,B,255;"
360 GOSUB 600: PRINT #1, "output 03; POKE 4,A,0;"
    POKE 4,A,255;"
370 GOSUB 600: PRINT #1, "output 03; POKE 4,B,0;"
    POKE 4,B,255;"
380 GOSUB 600: PRINT #1, "output 03; POKE 5,A,0;"
    POKE 5,A,255;"
390 GOSUB 600: PRINT #1, "output 03; POKE 5,B,0;"
    POKE 5,B,255;"
400 GOSUB 600: PRINT #1, "output 03; POKE 2,C,0;"
    POKE 2,C,255;"
410 GOSUB 600: PRINT #1, "output 03; POKE 3,C,0;"
    POKE 3,C,255;"
420 GOSUB 600: PRINT #1, "output 03; POKE 1,C,0;"
    POKE 1,C,255;"
430 GOSUB 600: PRINT #1, "output 03; POKE 1,D,0;"
    POKE 1,D,255;"
440 GOSUB 600: PRINT #1, "output 03; POKE GLOBAL,0;"
    POKE GLOBAL 255;"
450 GOSUB 600: PRINT #1, "output 03; POKE RESET,0;"
    POKE RESET,255;"
460 GOSUB 600: PRINT #1, "output 03; POKE STROBE,0;"
    POKE STROBE,255;"
470 GOSUB 600: PRINT #1, "output 03; LOOP; X;"
480 PRINT " — Press any key to terminate test —"
490 '
500 *

```

```

510 ** Clear the 576 and terminate
520 **
530 WHILE INKEY$="": WEND
540 PRINT #1, "clear 03"
550 END
560 '
570 **
580 ** Subroutine to ensure that 576 is NOT BUSY
590 **
600 SPBYTE% = 0: NOT.BUSY% = &H80
610 WHILE (SPBYTE% AND NOT.BUSY%) = 0
620     PRINT #1, "spoll 03"
630     INPUT #2, SPBYTE%
640 WEND
650 RETURN
    
```

For all the signal checks, connect the oscilloscope low signal lead to chassis ground, and connect the high signal lead to the indicated terminal of the designated component. Adjust the time base, triggering and input attenuator controls as required. Timing of the waveforms is not critical since failure is most likely to be seen as a complete absence of signal. A storage scope may be necessary to catch short pulses. Alternatively, a logic probe may be used to verify the presence of signals.

All command lines accessed by the program above are low true logic. Therefore, a low-going pulse will be seen on the appropriate terminal.

Table 8-2. Signal Checks

Function	Command	Component*	Terminal(s)
All	D0	J303	20, 25
All	D1	J303	19, 26
All	D2	J303	18, 27
All	D3	J303	17, 28
All	D4	J303	16, 29
All	D5	J303	15, 30
All	D6	J303	14, 31
All	D7	J303	13, 32
All	STROBE	J303	36
All	R/W	J303	9
ALL	GLOBAL1	J303	12
ALL	RESET	J303	33
Analog In	CMDA1	J301	35
	CMDB1	J301	34
	CMDC1	J301	10
	CMDD1	J301	9
Trigger	CMDA2	U211	5
	CMDB2	U211	3
	CMDC2	U211	1
Option Slot	CMDA3	J303	35
	CMDB3	J303	34
	CMDC3	J303	11
Analog Out	CMDA4	U401	2
	CMDB4	U401	12
Digital I/O Relay Control	CMDA5	U501	1, 4, 5
	CMDB5	U501	4

Analog Input

The analog input circuitry is all contained on the AMM1A or AMM2 Analog Master Measurement module (AMM) that was included with the system unit. A separate troubleshooting and diagnostic procedure may be found in the appropriate AMM manual.

Analog Output

Analog output converts 12-bit digital values into analog voltages. Use the procedure listed below to troubleshoot analog output.

1. Connect the DMM to the output terminals of the channel to be tested.
2. Verify the presence of the +10V reference from the AMM module.
3. Using either the WRite command or the POke command, program the Model 576 analog output circuitry for minimum output (for example: $\pm 10V$ on the $\pm 10V$ range (see the Reference section of this manual for PEEK/POKE command address). Check the DMM for an accurate reading.
4. Program the analog output for a mid-range output (e.g. 0V on the $\pm 10V$ range) and check the voltage with the DMM.
5. Program for the analog output for a maximum range value and read the voltage on the DMM. The voltage should be 9.997V on the $\pm 10V$ range.

Digital Input/Output

Trouble within digital input/output and power control can be traced simply by storing all logic "low" and then all logic "high" to all the various channel bit positions. A logic probe or DMM can be used to trace through the circuitry for each channel to determine the location of the fault. This can be done either through the Write or the POke commands.

SPECIAL HANDLING OF STATIC SENSITIVE DEVICES

CMOS devices are designed to operate at very high impedance levels for low power consumption. As a result, any normal static charge that builds up on your person or clothing may be sufficient to destroy these devices if they are not handled properly. When handling these devices, use the following precautions to avoid damaging them.

1. The devices should be transported and handled only in containers specially designed to prevent static build-up. Typically, these parts will be received in static-protected containers of plastic or foam. Keep these devices in their original containers until ready for installation.
2. Remove the devices from their protective containers only at a properly grounded work station. Also ground yourself with a suitable wrist strap.
3. Handle the devices only by the body; do not touch the pins.
4. Any printed circuit board into which the device is to be inserted must also be grounded to the bench or table.
5. Use only anti-static type soldering irons.
6. Use only grounded soldering irons.
7. Once the device is installed on the PC board, the device is normally adequately protected and normal handling may resume.

CHANGING THE DATA MEMORY CHIP OR CLOCK/BATTERY SOCKET

The following instructions cover changing the memory chip and clock/memory back-up battery for Revision E and later 576 mother boards. Revision E can be identified by the board label, located near the power supply and sideboard, and by the presence of jumper J302 near the clock oscillator Y301.

Beta 576 units and a few early production units lack jumper J302, and must be reworked slightly. If your system is an earlier revision, contact Keithley DAC for further instructions.

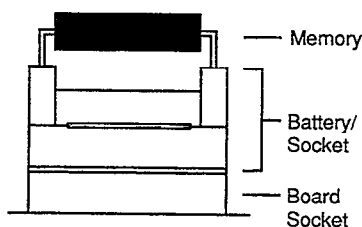
The Model 576 is available with a 128k static RAM memory chip as standard equipment, or with a 512k RAM chip as an option. This is a factory-installed option, but may be upgraded in the field by a qualified technician.

The clock/calendar/data memory back-up battery is an internal part of the socket holding the system memory chip and clock (U304). The battery is not user-serviceable; the socket must be replaced when the battery is exhausted. This condition will become evident when the Model 576 no longer retains the correct time when the system is switched off. The clock/battery socket may also be renewed in the field by a qualified technician.

Replacing the Battery and Memory

The standard 128k RAM is Keithley part number LSI-104 or Sony part number CXK581000-15L. The optional 512k RAM is Keithley part number LSI-111 or Hitachi part number HM66205-10. Other equivalent memory chips may not seat properly in the memory/clock socket.

The battery/clock/calendar socket is Keithley part number SO-129, Dallas Semiconductor part number DS-1216D, or Radio Shack part number 25-1033. Note that SO-129 looks like two sockets. These are permanently joined, and reside in a low-profile socket which is soldered to the 576 mother board.



CAUTION

The RAM chip is susceptible to static damage. Work only on an approved anti-static mat, or take other appropriate anti-static measures to assure you do not destroy the memory chip. This damage is not covered under warranty.

1. Turn off the Model 576 and unplug the power cable.
2. Open the Model 576 and remove any modules in the option slots.
3. Refer to the mother board component layout to locate the static RAM chip U304 and jumper J302. U304 is mounted in the SO-129 socket, and located approximately 3 in. (7.5 cm) from the rear panel and 1.5 in. (3.8 cm) from the side board assembly. Jumper J302 is near the clock oscillator Y301.
4. Remove the memory chip. Use a small screw driver to alternately lift each end of the memory chip a little at a time until it can be removed from the socket. Transfer the chip to piece of anti-static foam or an anti-static tube.

CAUTION

It is extremely important that you alternately lift each end of the chip a little at a time until it is free. The 512k RAM chip is manufactured on a thin substrate which may crack if the chip is stressed excessively during removal. This damage is not covered under warranty.

5. If you are renewing the battery socket, remove the existing SO-129 using a small screw driver. Alternately lift each end of the socket a little at a time until it can be removed from the underlying mother board socket. Keithley recommends that you change the socket if you are upgrading to the 512k memory chip.

Orient and align the new SO-129 socket correctly over the mother board socket. Gently press the new SO-129 socket into the mother board socket until it is fully seated.

6. Reinstall the memory chip (or upgrade). Orient chip U304 correctly in socket SO-129, and line up all pins with their respective socket holes. Gently seat the chip in the socket, using your thumbs and finger tips to distribute pressure uniformly over the side edges of the chip. If you are changing chip types, confirm that jumper J302 is correctly positioned for the type of chip installed (over pins 1 and 2 for the 128k RAM, or pins 2 and 3 for the 512k RAM).
7. Reinstall any module(s) removed in step 1.
8. Restore power to the system. The 576 is now ready to use.

REPLACEABLE PARTS

This contains replacement parts information for the 576 system.

Part numbers for individual components are listed on the component layout for the individual boards. Do not confuse the component designation with the part number. A typical part number could be R-76-10k, while the component designation might be R105. Additional available parts are listed in Table 8-4.

Keithley Data Acquisition and Control maintains an inventory of all normal replacement parts. To place an order, or to obtain information concerning replacement parts, first contact the Keithley Data Acquisition and Control Customer Service Department. When ordering parts, include the following information:

Model number
 Serial number
 Module type and part number (as marked on the board)
 Part Description

Circuit designation, including schematic and component layout designations (if applicable)
 Part number

Table 8-3. Model 576 Mother Board, Parts List

Item	Quantity	Reference	Part Number
1	32	C57-C59, C67-C69, C77-C99, C206, C207, C301	C-365-.1
2	4	J12-J15	CS-521-2
3	4	U504, U506, U508, U510	IC-230
4	4	U505, U507, U509, U511	IC-242
5	2	U502, U404	IC-366
6	3	U412, U212, U401	IC-179
7	1	U512	IC-257
8	1	U503	IC-157
9	5	U501, U103, U104, U105, U408	IC-182
10	8	W1, W301, W501-W506	J-3
11	1	U101	IC-137
12	2	C4, C2	C-237-1
13	2	R10, R13	R-88-121
14	1	R2	R-3-470
15	1	R3	R-76-2.7K
16	3	L1-L3	CH-20-2
17	3	D2-D4	RF-70
18	2	C11-C12	C-314-22
19	1	J101	CS-699-3
20	1	U1	IC-676
21	4	R9, R14, R237, R238	R-76-100
22	1	R15	R-76-15
23	2	R11, R12	R-88-1.3K
24	4	C9, C14, C103, C104	C-64-1000P
25	6	C3, C6, C15, C16, C19, C22	C-22-.01
26	1	C20	C-351-2200
27	4	D6, D301-D303	PL-86-2
28	7	R16, R227, R228, R236, R302-R304	R-76-390
29	1	R17	R-76-15K
30	1	U3	IC-308
31	1	U4	IC-309
32	1	R6	R-88-249K
33	1	R18	R-76-2.2K
34	1	R5	R-8-187K
35	1	SW1	SW-479
36	1	U106	IC-219
37	9	C1, C5, C7, C8, C10, C13, C17, C18, C21	C-314-220
38	1	R4	RP-104-2K
39	10	TP-DGND, TP1-TP5, TP+5, TP+15, TP-15, TP-AGND	CS-553

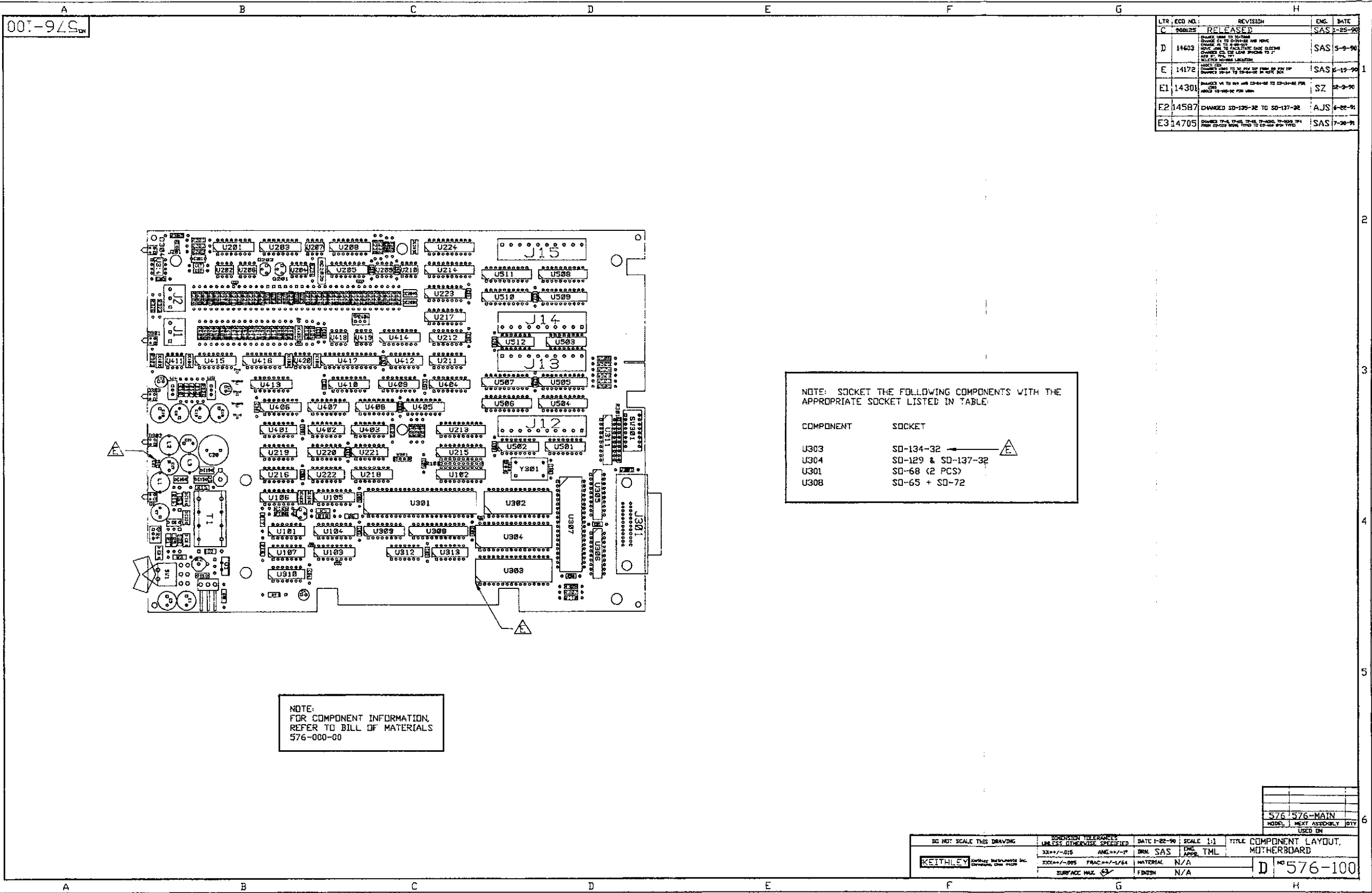
Item	Quantity	Reference	Part Number
40	1	U2	IC-677
41	1	D5	RF-53
42	1	R1	R-88-1K
43		T1	TR-268A
44	2	U402, U216	IC-163
45	2	U403, U220	IC-233
46	1	U406	IC-389
47	1	U407	IC-269
48	5	U409-U411, U221, U313	IC-144
49	1	U413	IC-190
50	1	U405	IC-272
51	2	J1, J2	CS-521-1
52	4	U415, U416, U201, U203	IC-320
53	1	U417	IC-678
54	1	U418	IC-203
55	3	U206, U207, U419	IC-246
56	4	U202, U204, U420, U421	IC-504
57	4	R401, R402, R406, R413	-10K
58	1	R403	R-88-22.1
59	3	R215, R405, R412	R-88-4.99K
60	2	R407, R414	R-263-6K
61	2	R408, R415	R-88-1.82K
62	4	R409, R411, R416, R418	R-263-2K
63	2	R410, R417	R-88-3.16K
64	1	R404	RP-104-200
65	4	C401-C404	C-64-22P
66	2	C405, C205	C-64-100P
67	1	U414	IC-686
68	2	U218, U223	IC-386
69	2	U107, U211	IC-186
70	3	U22, U213, U214	IC-263
71	1	U215	IC-357
72	1	U217	IC-558
73	1	U219	IC-215
74	1	U222	IC-231
75	1	U205	IC-267
76	3	R212, R229, R230	R-76-100K
77	5	R225, R226, R231, R232, R235	R-76-4.7K
78	8	R208-R211, R220-R223	R-263-20K
79	1	R224	R-76-470K
80	2	R202, R201	R-76-1K
81	1	R203	R-76-10M
82	2	R207, R206	R-88-5.49K
83	3	R204, R205, R217	R-88-49.9K
84	1	R213	R-88-301
85	1	R214	R-88-1.33K
86	1	R216	R-88-15K
87	1	R218	R-88-150K
88	1	R219	R-88-499K
89	3	R102, R103, R234	R-76-10K
90	2	U209, U210	IC-173
91	1	U208	IC-321
92	1	Q202	TG-84
93	1	Q201	TG-47
94	1	W201	CS-339-4
95	2	C203, C204	C-64-150P

SECTION 8
Calibration, Maintenance, and Troubleshooting

Item	Quantity	Reference	Part Number
96	1	J201	CS-339-2
97		D201-D206	RF-28
98	1	C201	C-305-.047
99	1	C202	C-306-.001
100	1	U102	IC-307
101	1	R108	R-76-5.6M
102	2	R107, R106	R-176-100K
103	1	R105	R-176-4.75K
104	1	R104	R-176-4.99K
105	1	R109	R-88-422K
106	1	R101	TF-114-1
107	1	R110	R-88-1.62K
108	1	U310	IC-384
109	1	U311	IC-259
110	1	U312	IC-270
111	1	U309	IC-237
112	1	U302	LSI-90
113	1	U307	LSI-49
114	1	J301	CS-501
115	1	SW301	SW-389
116	1	R301	TF-99
117	1	U301	LSI-69
118	1	U303	LSI-109
119	1	U304	LSI-104
120	1	U308	IC-728A
121	1	R306	R-76-330
122	1	D304	PL-86-1
123	1	R305	R-76-270K
124	1	U314	IC-71
125	1	Y301	CR-28-1
126	1	U306	IC-726
127	1	U305	IC-725

Table 8-4. Model 576 Side Board, Parts List

Item	Quantity	Reference	Part Number
1	2	R301, R302	R-176-10K
2	2	R303, R304	R-88-21
3	2	R305, R306	R-176-1.09K
4	4	D301-D304	RF-34
5	10	R307-R316	R-76-1K
6	3	W301-W303	J-3
7	3	J301, J303, J304	CS-481-6
8	1	J302	CS-481-1
9	1	R317	R-1-1K
10	1	J307	CS-295-1
11	1	J305	CS-618-3
12	1	J306	CS-711
13	1	C301	C-342-10000
14	3	W304-W306	22-AWG-BARE-BUS



LTR. ECD NO.	REVISION	ENG.	DATE
C 146025	RELEASED	SAS	1-25-90
D 14603	CHANGE U303 TO SD-134 CHANGE U304 TO SD-129 & SD-137 CHANGE U301 TO SD-68 (2 PCS) CHANGE U308 TO SD-65 + SD-72	SAS	5-9-90
E 14172	CHANGE U303 TO SD-134 CHANGE U304 TO SD-129 & SD-137 CHANGE U301 TO SD-68 (2 PCS) CHANGE U308 TO SD-65 + SD-72	SAS	5-19-90
E1 14301	CHANGE U303 TO SD-134 CHANGE U304 TO SD-129 & SD-137 CHANGE U301 TO SD-68 (2 PCS) CHANGE U308 TO SD-65 + SD-72	SZ	5-23-90
E2 14587	CHANGE SD-129-32 TO SD-137-32	AJS	6-22-91
E3 14705	CHANGE U303 TO SD-134 CHANGE U304 TO SD-129 & SD-137 CHANGE U301 TO SD-68 (2 PCS) CHANGE U308 TO SD-65 + SD-72	SAS	7-26-91

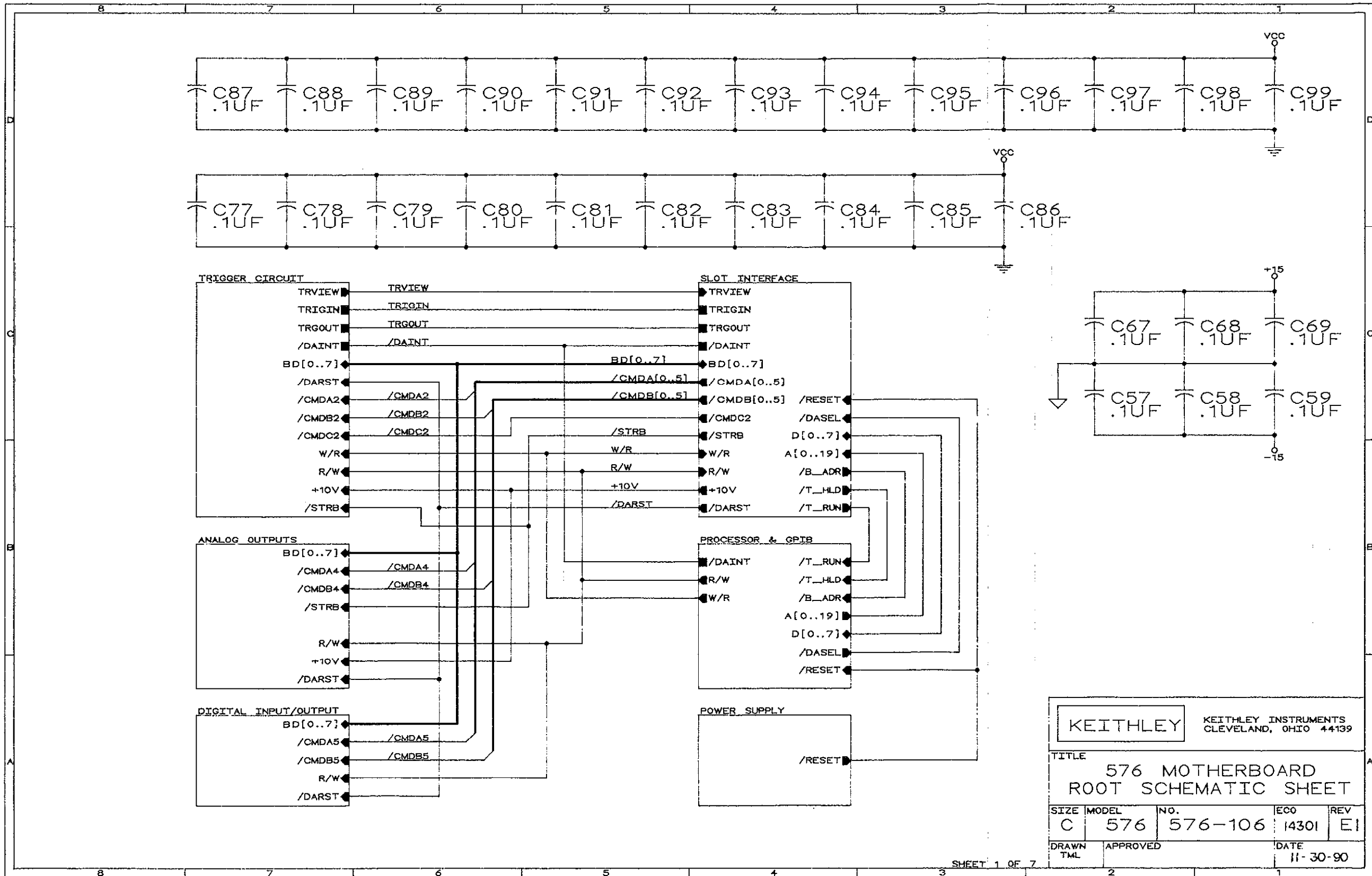
NOTE: SOCKET THE FOLLOWING COMPONENTS WITH THE APPROPRIATE SOCKET LISTED IN TABLE:

COMPONENT	SOCKET
U303	SD-134-32
U304	SD-129 & SD-137-32
U301	SD-68 (2 PCS)
U308	SD-65 + SD-72

NOTE:
FOR COMPONENT INFORMATION,
REFER TO BILL OF MATERIALS
576-000-00

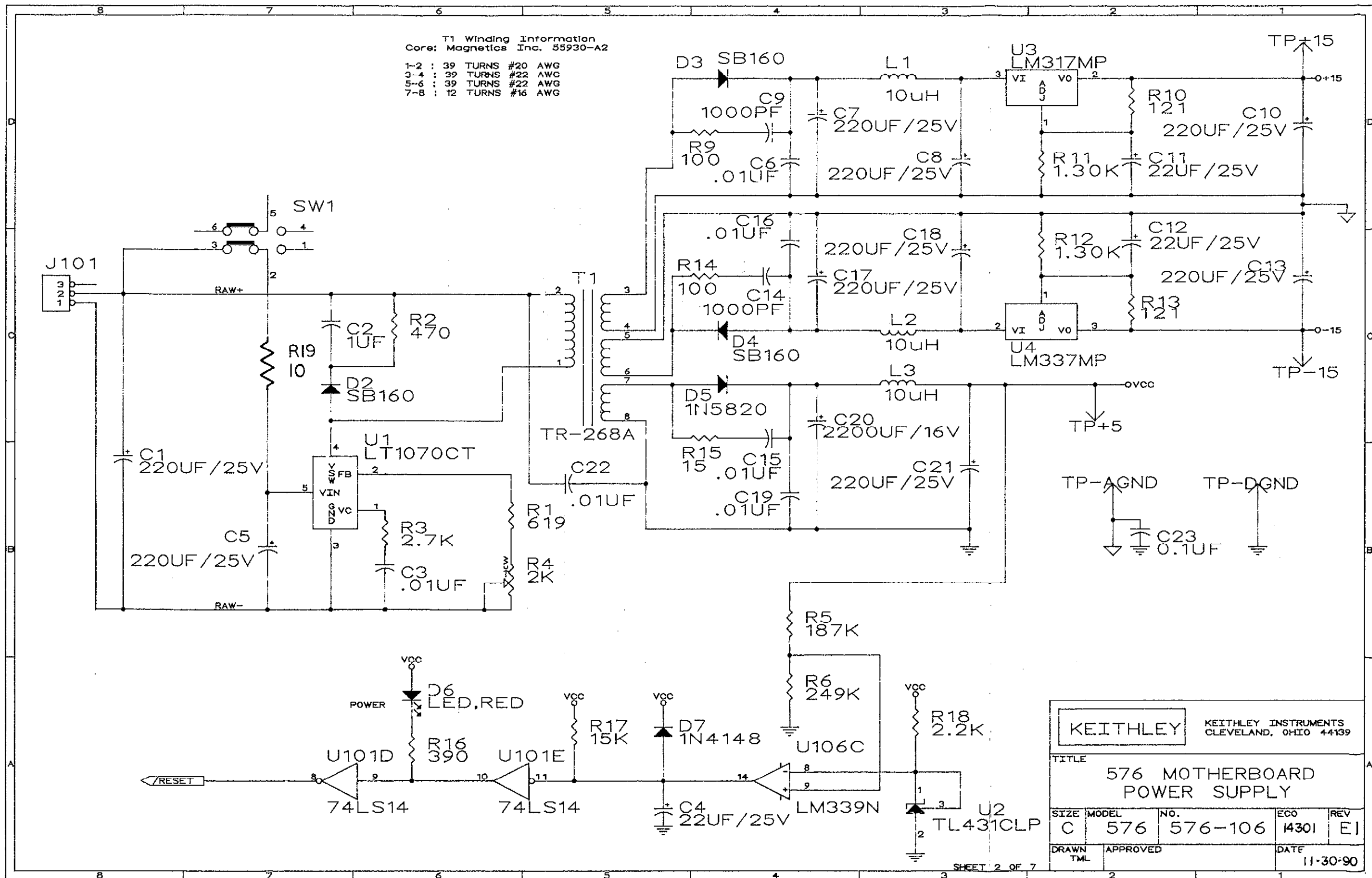
576-576-MAIN	
MODEL	NEXT ASSEMBLY QTY
USED ON	

DO NOT SCALE THIS DRAWING	DIMENSION TOLERANCES UNLESS OTHERWISE SPECIFIED	DATE 1-22-90	SCALE 1:1	TITLE COMPONENT LAYOUT, MOTHERBOARD
KEITHLEY Keithley Instruments Inc. 10000 University Ave. 19709	32±.015 ANG±.1°	DRN SAS	ENG TML	
	300±.005 FRAC±.0164	MATERIAL N/A		
	SURFACE MAX 63	FINISH N/A		
				D 576-100

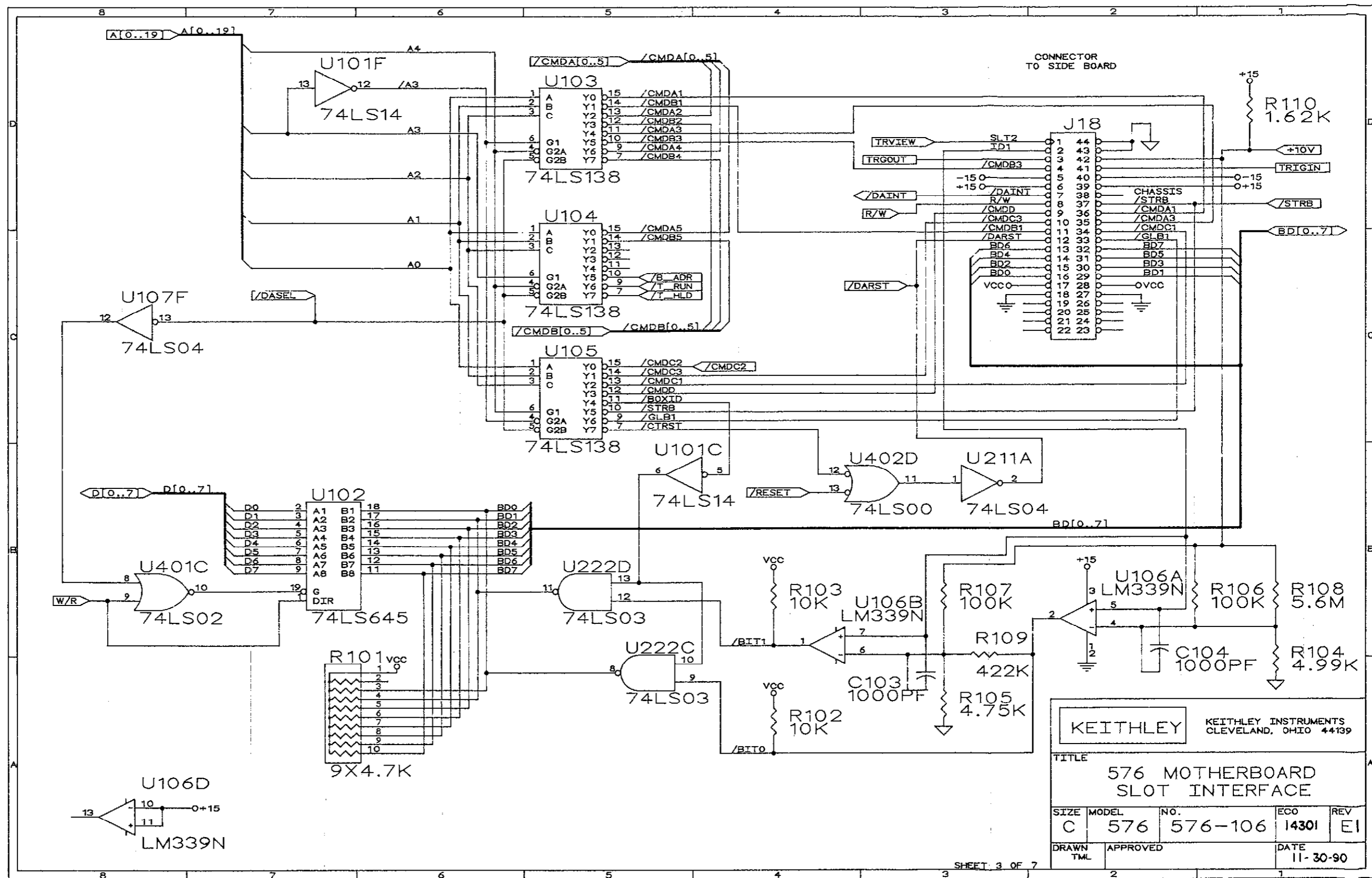


KEITHLEY		KEITHLEY INSTRUMENTS CLEVELAND, OHIO 44139			
		TITLE 576 MOTHERBOARD ROOT SCHEMATIC SHEET			
SIZE C	MODEL 576	NO. 576-106	ECO 14301	REV E1	
DRAWN TML	APPROVED	DATE 11-30-90			

T1 Winding Information
 Core: Magnetics Inc. 55930-A2
 1-2 : 39 TURNS #20 AWG
 3-4 : 39 TURNS #22 AWG
 5-6 : 39 TURNS #22 AWG
 7-8 : 12 TURNS #16 AWG



KEITHLEY		KEITHLEY INSTRUMENTS CLEVELAND, OHIO 44139			
		TITLE 576 MOTHERBOARD POWER SUPPLY			
SIZE	MODEL	NO.	ECO	REV	
C	576	576-106	14301	E1	
DRAWN	APPROVED		DATE		
TML			11-30-90		

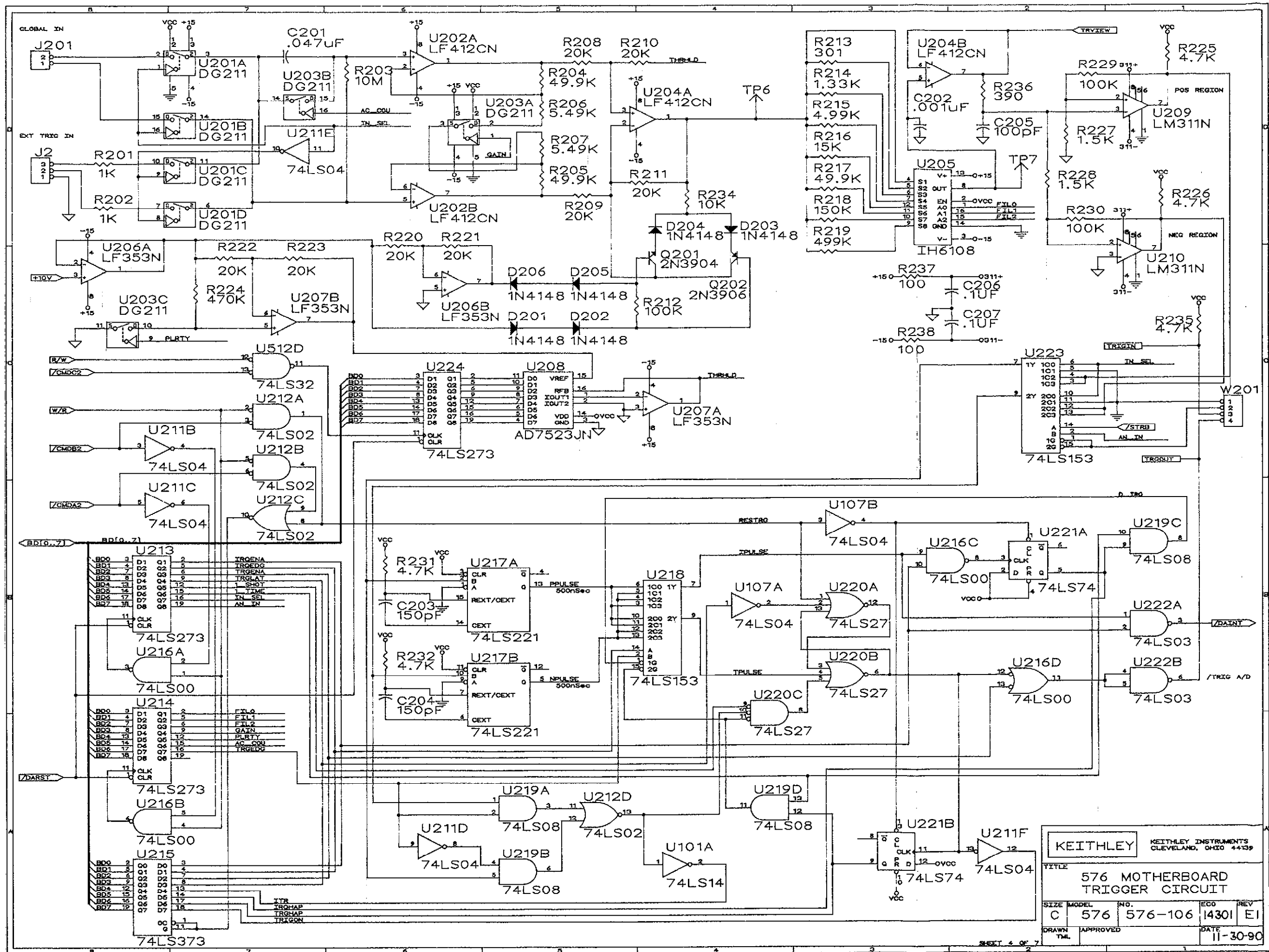


KEITHLEY KEITHLEY INSTRUMENTS
CLEVELAND, OHIO 44139

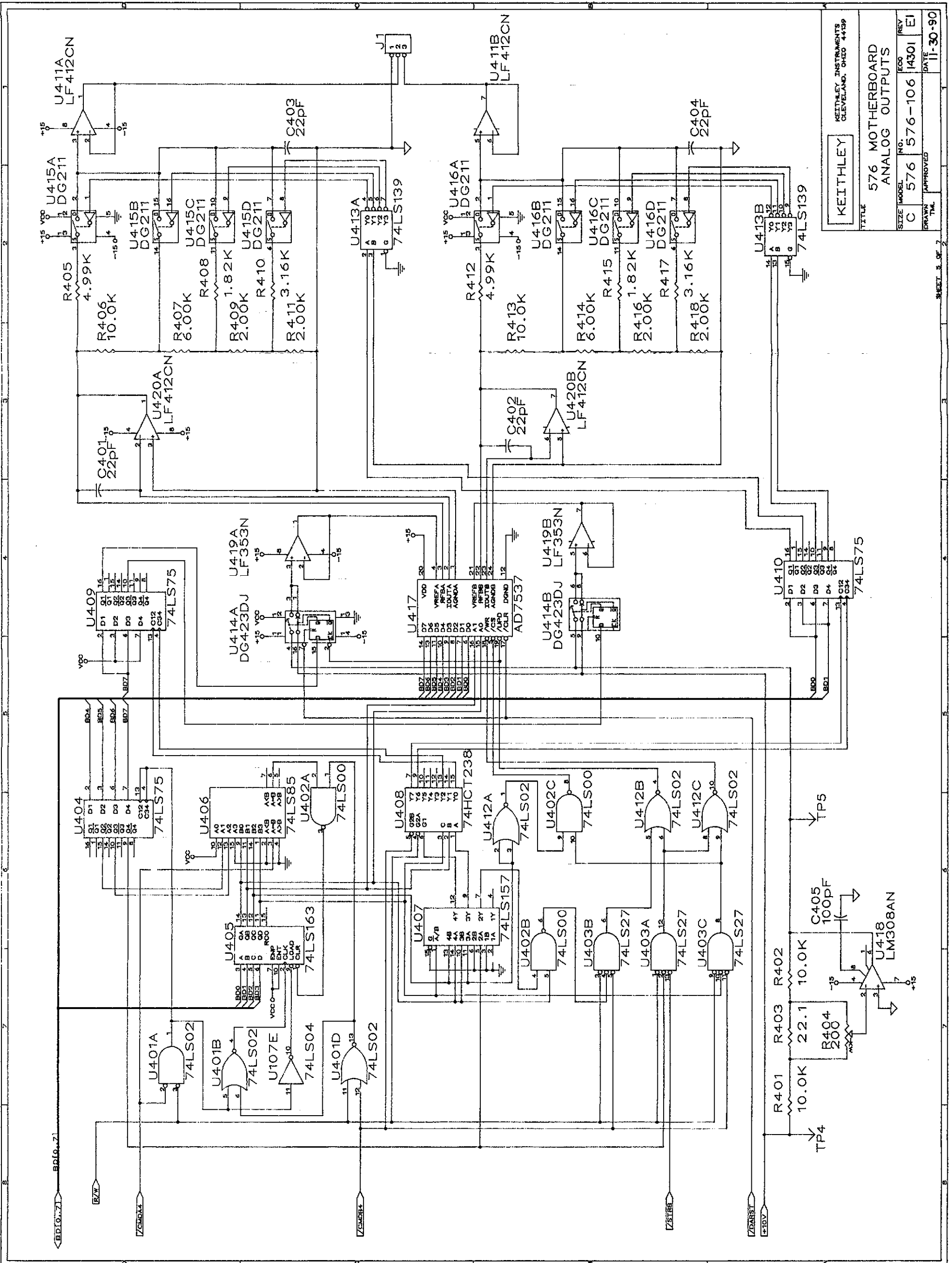
TITLE
576 MOTHERBOARD
SLOT INTERFACE

SIZE	MODEL	NO.	ECO	REV
C	576	576-106	14301	E1

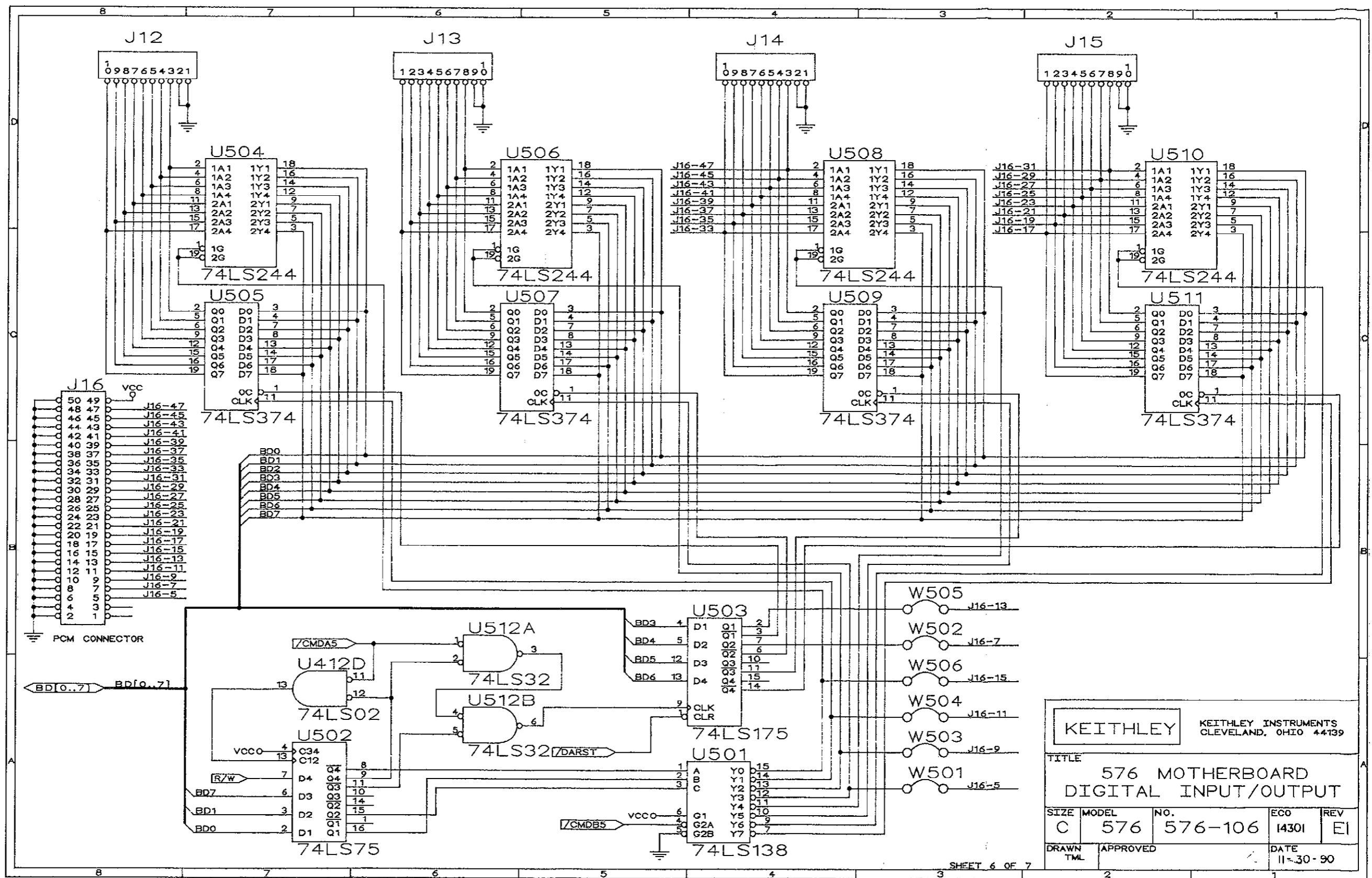
DRAWN	APPROVED	DATE
TML		11-30-90



KEITHLEY		KEITHLEY INSTRUMENTS CLEVELAND, OHIO 44139	
TITLE 576 MOTHERBOARD TRIGGER CIRCUIT			
SIZE	MODEL	NO.	ECO
C	576	576-106	14301
REV	REV		REV
E1	E1		E1
DATE	DATE		DATE
11-30-90	11-30-90		11-30-90



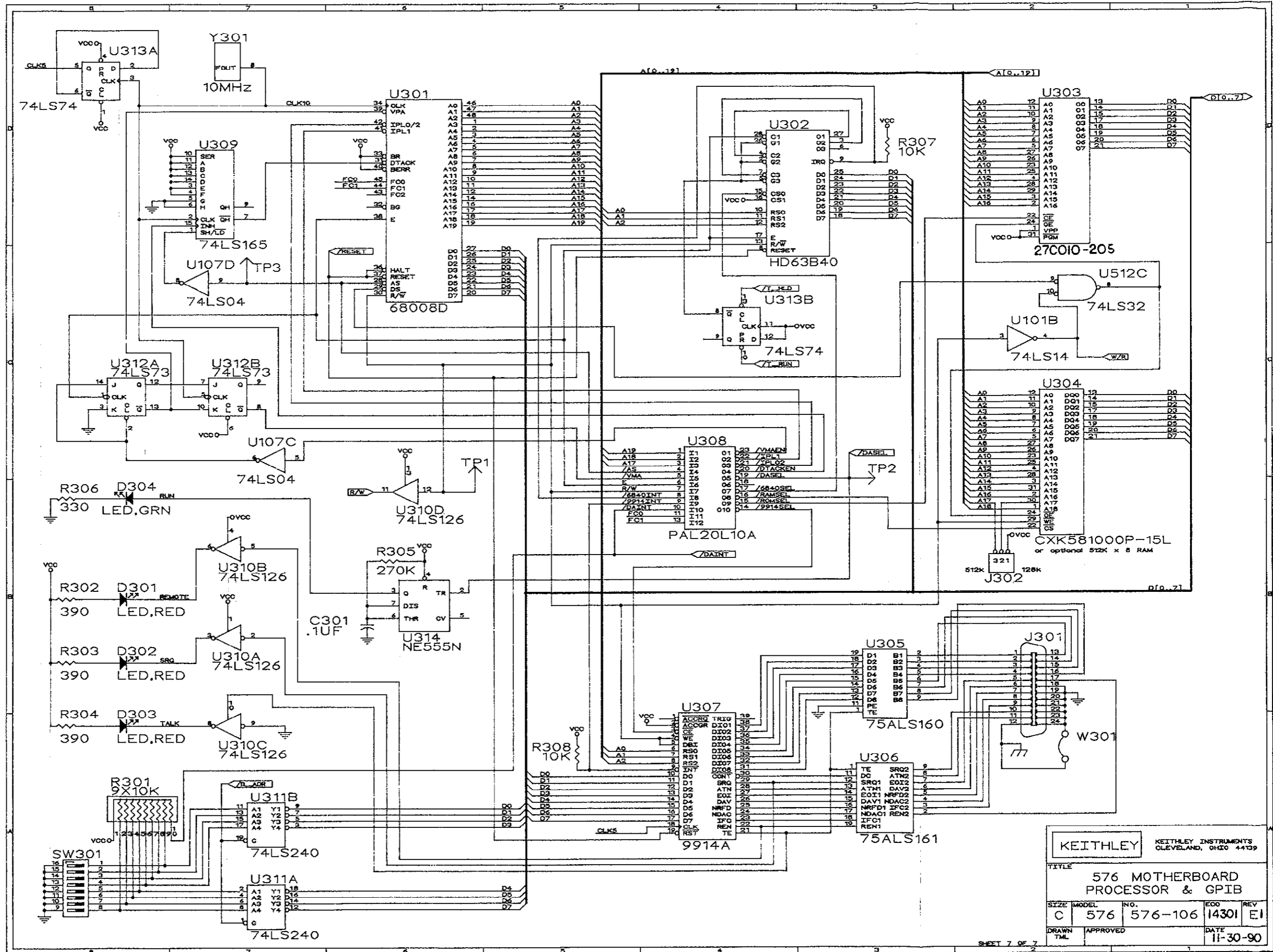
KEITHLEY		KEITHLEY INSTRUMENTS CLEVELAND, OHIO 44134	
TITLE 576 MOTHERBOARD ANALOG OUTPUTS			
SIZE	MODEL	NO.	REV
C	576	576-106	14301 EI
DRAWN	T.M.	APPROVED	DATE
			11-30-90



KEITHLEY KEITHLEY INSTRUMENTS
CLEVELAND, OHIO 44139

TITLE
**576 MOTHERBOARD
DIGITAL INPUT/OUTPUT**

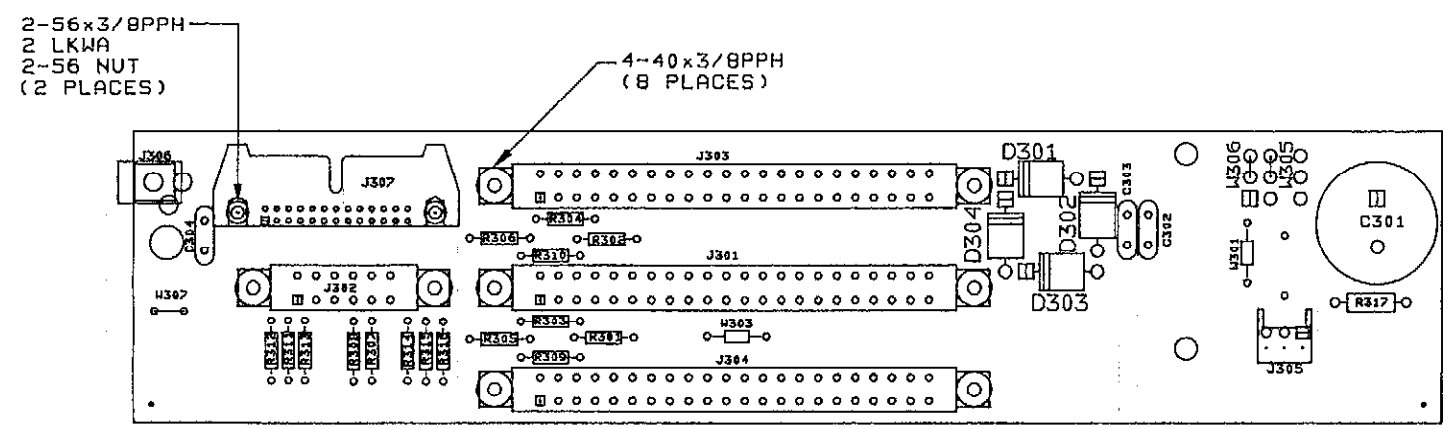
SIZE C	MODEL 576	NO. 576-106	ECO 14301	REV E1
DRAWN TML	APPROVED	DATE 11-30-90		



KEITHLEY KEITHLEY INSTRUMENTS CLEVELAND, OHIO 44139				
TITLE 576 MOTHERBOARD PROCESSOR & GPIB				
SIZE	MODEL	NO.	ECO	REV
C	576	576-106	14301	E1
DRAWN	APPROVED	DATE		
TML		11-30-90		

021-925 .DN

LTR.	ECO NO.	REVISION	ENG.	DATE
D	900215	RELEASED	SZ	2-12-90
D1	14172	DELETED W302 & W304. W307 HAS C305.	AS	9-17-90
E	14424	CHG'D ARTWORK REV FROM D TO E.		7-30-91

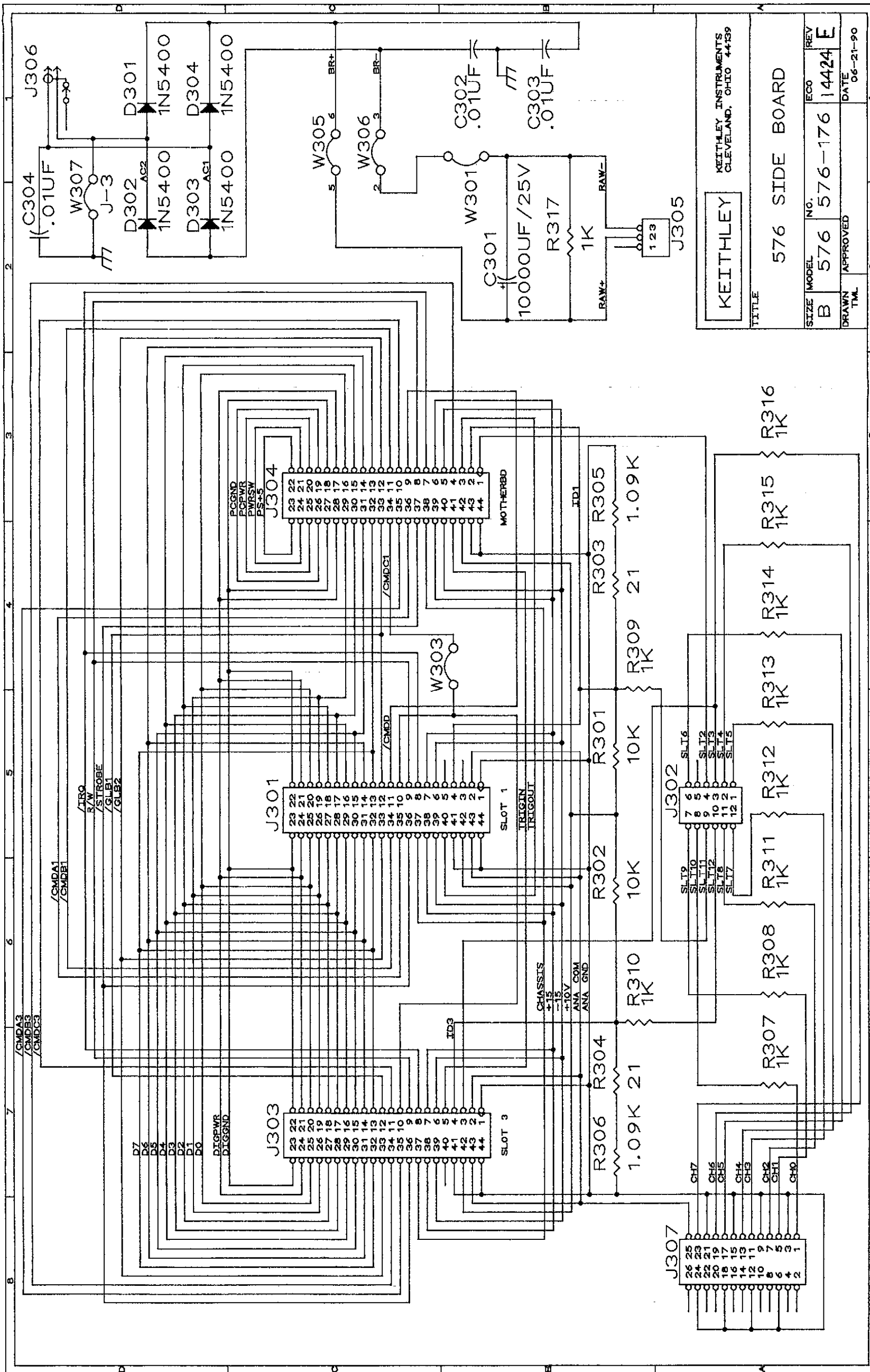


*BOARD TO BE LEADSAWED TO .050".

NOTE:
FOR COMPONENT INFORMATION,
REFER TO 576 PRODUCT STRUCTURE.

576	1
MODEL	NEXT ASSEMBLY QTY.
USED ON	

DO NOT SCALE THIS DRAWING	DIMENSIONAL TOLERANCES UNLESS OTHERWISE SPECIFIED	DATE 2-12-90	SCALE 1:1	TITLE COMPONENT LAYOUT, SIDE BOARD
KEITHLEY KEITHLEY INSTRUMENTS INC. CLEVELAND, OHIO 44139	XX=±.015 ANG.=±°	DRN. SZ	ENG. APPR. ESB	NO. 576-170
	XXX=±.005 FRAC.=±1/64	MATERIAL		
	SURFACE MAX. 63	FINISH		



KEITHLEY		KEITHLEY INSTRUMENTS CLEVELAND, OHIO 44139	
TITLE 576 SIDE BOARD			
SIZE	MODEL	NO.	REV
B	576	576-176	14424 E
DRAWN	APPROVED	DATE	
TML		06-21-90	

SECTION 9

Option Modules and Interfaces

Introduction

This section of the Model 576 Manual includes generic information on Keithley signal conditioning modules. It also provides a localized place within the manual where you may insert documentation for your modules.

Keithley Module Library

The following modules are compatible with the Model 576. New modules are added to the library from time to time. The specified ranges include the cumulative effects of global and local gains. If you do not see what you need, contact Keithley DAC Technical Support.

Table 9-1. Signal Conditioning Modules

Module	Description
AMM1A	16 single-ended or 8 differential channel Analog Master Measurement module. 62.5kHz, 12-bit A/D converter. Programmable filter. 100mV/1V/10V input.
AMM2	16 single-ended or 8 differential channel Analog Master Measurement module. 50kHz, 16-bit A/D converter. Programmable filter. 100mV/1V/10V input.
AIM2	32 single-ended analog input channels. 1V/10V ranges.
AIM3A	32 single-ended or 16 differential analog input channels. 10mV/100mV/1V/10V ranges.
AIM4	4 isolated analog input channels. 5mV/50mV/500mV/5V ranges.
AIM5	4 isolated analog low-level input channels. 5mV/50mV ranges.
AIM6	4 channel analog input module for 100Ω resistive temperature devices (RTDs).
AIM7	16 channel thermocouple and analog input module for types J, K, S, T, B, E, and R. Reference junction. 10mV/100mV ranges.
AIM8	4 channel strain gage and low-level analog input module. Excitation and bridge completion circuitry. Programmable filters. 1mV/10mV/100mV/1V/10V ranges.
AIM9	2 channel LVDT/RVDT/ carrier amplifier module with 1/2/5/10/20kHz AC excitation.
AOM1	2 or 5 channel analog output module. 12-bit ±2.5V/±5V/±10V/0-5V/0-10V ranges.
AOM2	1 or 2 channel analog output module. 16-bit ±10V/0-10V ranges.
AOM3	4 channel 0-20mA current loop analog output module.
AOM4	4 channel 0-10V excitation analog output module.
AOM5	4 channel 13-bit analog output module. ±1/±2/±5/+10V ranges
DIM1	16 channel isolated digital input module.
DIO1A	32 channel TTL-level digital input/output module.
DOM1	16 channel isolated digital output module.
PCM1	4 channel isolated AC power control module, 12-140VAC.
PCM3	16-channel external relay mounting board for use with Model 576, 575 and 570. Includes cable. Does not require an expansion slot.
PIM1	7 channel isolated high-level input plus 1 channel direct low-level input pulse/frequency counting module.
PIM2	4 channel TTL-level event counter. Settable as four 16-bit or two 32-bit counters.
PROTO*	Prototyping module with one bidirectional data port.
STEP1*	Stepper Motor Controller Module (1 per up to 4 STEP2 modules)
STEP2*	Stepper Motor Indexer Module (1 per motor).
TRG1*	Programmable Hardware Trigger Module.
WAV1*	Programmable waveform generator module.

* Supported by PEEK and POKE commands in the Model 576. STEP set uses both available slots in the Model 576; precludes use of analog input. Analog output may be used, but accuracy will be less than rated accuracy (analog output requires the precision reference on the AMM module.)

Table 9-2. Module Power Consumption

Module	±15V	+5V	Remarks
AMM1A	65	125	
AMM2	65	125	
AIM2	2	20	
AIM3	18	30	
AIM4	30	30	
AIM5	30	30	
AIM6	30	30	
AIM7	15	15	
AIM8	75	75	Plus excitation current
AIM9	125	60	
AOM1	90	60	
AOM2	30	180	
AOM3	150	65	
AOM4	200	75	
AOM5			
DIM1	0	277	max1
DIO1A	0	270	Internal power, all outputs on
DOM1	0	650	max2
PCM1	0	90	max3
PCM3	0	0	Power supplied by Model 576
PIM1	0	380	
PIM2	0	475	
TRG1	22	135	
WAV1	85	65	

Notes: 1 – 14mA for each input on. 2 – 15mA for each output on (EXT), or 35mA (INT) 3,4 – 12mA for each output on

Table 9-3. Module Self-ID Resistor Values

Module	Resistor Value Ω	ID Number
AMM1A	845	29
AMM2	976	31
AIM2	105	32
AIM3A	1090	33
AIM4	1150	34
AIM5	1260	35
AIM6	1320	36
AIM7	1400	37
AIM8	1520	38
AIM9	1600	39
PIM1	1740	40
PIM2	1870	41
STEP1	1960	42
STEP2	2100	43
AOM1/2	2260	44
AOM1/5	2430	45
AOM2/1	2640	46
AOM2/2	2910	47
AOM3	3160	48
AOM4	3440	49
DIM1	3790	50
DOM1	4070	51
DIO1A	4530	52
PCM1	5050	53
PROTO	7590	56
TRG1	8760	57
AOM5	10700	58
WAV1	13300	59

Module Manuals

(Use this space to insert the dividers and manuals for your modules)

APPENDIX A

Device Clear and Interface Clear

Device Clear

Whenever the 576 receives a Device Clear (DCL) or a Selective Device Clear (SDC), the 576 is restored to a known state. The following conditions occur:

1. RESET OUT is executed to clear all 576 outputs.
2. All 576 buffers can be re-dimensioned. However any data that was acquired can still be manipulated (i.e. BUFF READ, BUFF STAT, etc.).
3. Any 576 program, or portion of a program that was sent to the 576, is erased. This means that DCL can be used to HALT a 576 program or clear a "wrong" program statement. Program statements are those that must be eXecuted. Immediate mode commands such as SYST and CHAN are not affected. The IDLE bit in the 576 device status byte will be set.
4. Any data in the 576 input queue is cleared. This means any unprocessed commands that were sent to the 576 prior to the DCL will be lost. This includes data sent to the 576 (i.e. BUFF WRITe).
5. Any data in the 576 output queue is erased. This means that if a DCL is issued before data from the 576 is read, the data will no longer be available. The DATA RDY bit in the 576 device status byte (serial poll) will also be cleared.
6. The hardware trigger in the 576 is reset. The TRIGger command must be re-issued in order to use the hardware trigger.
7. SYST :SRQ NONE; is set so that service requests will not be generated by the 576. The SYST :SRQ command must be re-issued to enable service requests.
8. DEBUG 0 is executed to reset the debug code to 0.
9. All the bits in the 576 device status byte are set to reflect the current system status. Any bits that were latched due to SYST :SRQ are unlatched. Refer to the 576 Manual command section for more information on the 576 device status byte.
10. The SYST :SAVE PROG command, if set, reverts to a SYST :SAVE DATA command. The following conditions are NOT affected by Device Clear:
11. BUFFer dimensions and BUFFer data are not modified.
12. Any configuration set by the SYST or CHAN commands with the exception of SYST :SRQ, is not affected (i.e. gains, ranges, etc., are not reset).

Interface Clear

Whenever an Interface Clear (IFC) is received by the 576 the 576 releases the IEEE-488 bus. Any data in the 576 input or output queues is not affected.

APPENDIX B

Error Conditions and Messages

The 576 generates two types of errors: Parse-Time errors and Run-Time Errors. Parse-time errors are generated when the 576 receives a command containing an illegal parameter, too many parameters, or when the specified operation does not match the hardware for which it is requested. Run-time errors are generated during the execution of a 576 program. Examples of run-time errors are 1) a subroutine was called but never defined, 2) a buffer read operation was attempted prior to information being placed into the buffer, etc.

When a parse-time error occurs, the command that was in error and any further text in the input buffer will be lost until the 576 receives a semi-colon character. At this time, the 576 will attempt to continue parsing the input stream.

If an error occurs at run time, the system will log the error but continue to operate. This is the default operation of the 576, but can be overridden by using the "SYST :SRQ on ERROR" command and issuing a HALT on SRQ within a program. In this case the 576, will stop executing the program as soon as an error occurs.

To retrieve the error status of the 576, a Serial Poll command must be issued (see the 576 Manual command section for further information on obtaining Device Status). To retrieve a textual error message from the 576, issue a SYST : ERR ? command. The 576 will return the current error message from the system. If the SYST :ERR ? com-

mand is issued and the 576 has not detected an error condition, the string "NO ERRORS" will be returned. A complete list of the 576 errors strings follow. If an error occurs in a program and the error does not match one of these, it is likely that the error was generated by the GPIB interface, programming language, or controller. If so, consult documentation for other components of the total system to determine the cause of the error.

576 ERROR STRINGS (RETURNED WITH SYST :ERR ? COMMAND)

AMM NOT VALID TRIGGER CHANNEL FOR MULTIPLE CHANNELS
ATTEMPTED TO WRITE TO AN INPUT BUFFER
AVERAGING ALLOWED WITH ANALOG INPUT MODULES ONLY
BAD CHANNEL NUMBER SPECIFIED
BAD CHARACTER IN SUBROUTINE NAME
BAD IMMEDIATE INTEGER DATA RECEIVED
BAD NUMERIC VALUE RECEIVED
BAD SLOT NUMBER SPECIFIED
BUFFER ALREADY EXISTS
BUFFER EMPTY
BUFFER IS THE WRONG TYPE
BUFFER MUST BE DIMENSIONED BEFORE USE
CAN'T CALIBRATE A/D MODULE
CAN'T HAVE MULTIPLE HARDWARE TRIGGERS DEFINED

CAN'T HAVE NESTED ELSE'S
CAN'T READ FROM CHANNEL GATE
CHANNEL CONFIGURED FOR INPUT
CHANNEL CONFIGURED FOR OUTPUT
CHANNELS DON'T MATCH BUFFER DIMENSION
ENABLE OR DISABLE EXPECTED
ENDING CHANNEL MUST BE LARGER THAN
START CHANNEL
ENDSUB ENCOUNTERED WITHOUT SUBR STATE-
MENT
EXPECTED NUMERIC INPUT NOT FOUND
INTERNAL ERROR, SYSTEM RESTARTED
INTERRUPT RATE IS TOO FAST
INVALID BUFFER COMMAND OPTION
INVALID BUFFER SPECIFIED
INVALID CALIBRATION CONSTANT SPECIFIED
INVALID CHANNEL - DOESN'T EXIST IN BUFFER
INVALID CHANNEL GAIN SPECIFIED
INVALID CHANNEL MODE OPTION
INVALID CHANNEL RANGE SPECIFIED
INVALID CHANNEL SUBOPTION
INVALID CLOCK OPTION SPECIFIED FOR COM-
MAND
INVALID COMMAND LOCATION SPECIFIED
INVALID COMMAND SPECIFIED
INVALID CONDITIONAL SPECIFIED
INVALID CONVERSION FOR BUFFER WRITE COM-
MAND
INVALID CONVERSION SPECIFIED
INVALID COUPLING SPECIFIED
INVALID DATE SPECIFIED
INVALID DEBUG NUMBER SPECIFIED
INVALID END OF LINE TERMINATOR SPECIFIED
INVALID ENGINEERING UNITS FOR CHANNEL
READ
INVALID ENGINEERING UNITS SPECIFIED
INVALID FILTER VALUE SPECIFIED
INVALID HALT MODE SPECIFIED
INVALID INTERRUPT RATE SPECIFIED
INVALID LIMIT VALUE SPECIFIED
INVALID MODE OPTION SPECIFIED
INVALID MODULE NAME SPECIFIED
INVALID MODULE OPTION SPECIFIED
INVALID NUMBER OF CYCLES SPECIFIED
INVALID OR INCOMPLETE PIM1 MODE SPECIFIED
INVALID OR INCOMPLETE PIM2 MODE SPECIFIED
INVALID OR MISSING HIGH LIMIT VALUE
INVALID PIM OPERATING MODE
INVALID POKE COMMAND
INVALID READ OPTION SPECIFIED
INVALID RESET MODE SPECIFIED
INVALID REUSE OF BUFFER
INVALID SAVE OPTION SPECIFIED
INVALID SRQ MASK OPTION SPECIFIED
INVALID SYSTEM AMM OPTION SPECIFIED
INVALID SYSTEM CLOCK COMMAND
INVALID SYSTEM COMMAND OPTION
INVALID SYSTEM SUBOPTION SPECIFIED
INVALID SYSTEM TRIGGER MODE SPECIFIED
INVALID TIME PERIOD SPECIFIED
INVALID TIME SPECIFIED
INVALID TIME UNITS SPECIFIED
INVALID TIMESTAMP MODE SPECIFIED
INVALID TIMESTAMP RANGE SPECIFIED
INVALID TRANSFER FORMAT SPECIFIED
INVALID TRIGGER LEVEL SPECIFIED
INVALID TRIGGER MODE FOR MULTIPLE CHAN-
NELS
INVALID TRIGGER MODE SPECIFIED
INVALID TRIGGER THRESHOLD VALUE
INVALID TRIGGER VOLTAGE SPECIFIED
INVALID WAIT OPTION SPECIFIED
MIXED 16 AND 32 BIT EVENT COUNTING NOT AL-
LOWED
MIXED EVENT AND FREQUENCY COUNTING NOT
ALLOWED
MODULE CAN'T BE PLACED INTO SPECIFIED SLOT
MODULE DOES NOT SUPPORT QUICK READ
MODULE DOES NOT SUPPORT READ OPERATION
MODULE DOES NOT SUPPORT WRITE OPERATION
MULTIPLE CHANNELS NOT ALLOWED FOR PIM1
EVENT
NESTING LEVEL EXCEEDED
NO CALIBRATION CONSTANT SPECIFIED
NO ERRORS
NO IF COMMAND GIVEN
NO READ OR WRITE COMMAND GIVEN FOR
BUFFER

NO WHILE STATEMENT FOUND
NOT ENOUGH MEMORY AVAILABLE FOR OPERA-
TION
NUMBER OF AVERAGES MUST BE GREATER THAN 0
NUMBER OF SCANS EXPECTED
OFFSET REQUIRED IF SCALE IS SPECIFIED
OLD & NEW BUFFERS DIMENSIONED DIFFERENTLY
ONLY 10 SUBROUTINES CAN BE DEFINED
OPEN IF/WHILE/DO COMMAND EXISTS
OUT OF CGM MEMORY
OUT OF INTERNAL SYSTEM MEMORY

READ OPTION FLAG
SLOT OCCUPIED BY ANOTHER MODULE
START OF LOOP NOT SPECIFIED
SUBROUTINE ALREADY DEFINED
SUBROUTINE NOT DEFINED
SYST :STAMP [MODE] SHOULD BE ONCE
SYST :STAMP [MODE] SHOULD BE SCAN
SYSTEM CALIBRATION HAS NOT BEEN PER-
FORMED
TIME STAMPING NOT ALLOWED
UNEXPECTED DATA RECEIVED AT END OF COM-
MAND

APPENDIX C

QuickStart Program Listings

QuickStart Programs for Interpreter BASIC, IOtech Driver488 Software Driver, and Compatible GPIB Interface

```
10 | *****
20 |
30 | ANALOG.IOT - 576 QUICKSTART Analog I/O Program in Interpreter BASICA
40 | DATE: 7-1-90
50 |
60 | This program is designed to run with the Driver488 GPIB driver software
70 | and a compatible GPIB interface card on an IBM PC/XT/AT or compatible.
80 | It assumes an address of "03" for the 576.
90 |
100 | (c) Copyright 1990 - Keithley Data Acquisition and Control
110 |
120 | *****
130 |
140 | CLS:CLOSE:KEY OFF
150 |
160 | Declare variables
170 |
180 | SPOLLBIT% = 0           ' Used as input to SPOLL check subroutine
190 | BITSTAT% = 0           ' Used as output from SPOLL check subroutine
200 | ADDR$ = "03"           ' Address of 576 as set on rear panel
210 | O$ = "output 03; "     ' Initial output string for Driver488
220 |
230 | Initialize Driver488 and interface for output.
240 |
250 | OPEN "\dev\ieeeeout" FOR OUTPUT AS #1
260 | IOCTL #1, "BREAK"
270 | PRINT #1, "RESET"
280 |
290 | Initialize Driver488 and interface for input
300 |
310 | OPEN "\dev\ieeeein" FOR INPUT AS #2
320 |
330 | Do some general GPIB housekeeping
340 |
350 | PRINT #1, "FILL ERROR"
360 | PRINT #1, "TIMEOUT 5"
370 | PRINT #1, "CLEAR"
380 |
390 | System should respond with driver revision and date, indicating
400 | successful communication with driver and GPIB interface
410 |
420 | PRINT #1, "HELLO"
430 | LINE INPUT #2, A$
440 | PRINT A$
```

```

450 '
460 LOCATE 3,1: PRINT"*** 576 QuickStart Analog I/O Test ***"
470 '
480 ' Reset 576
490 '
500 OUTPUT$ = "reset all;x;"
510 PRINT #1, OS + OUTPUT$
515 T!=TIMER: WHILE TIMER-T!<2: WEND ' Delay for RESET ALL. See
516 ' RESET command for details.
520 SPOLLBIT% = 7: GOSUB 2000
530 '
540 ' Set data transmission format
550 '
560 OUTPUT$="syst :format ASCII;"
570 PRINT #1, OS + OUTPUT$
580 SPOLLBIT% = 7: GOSUB 2000
590 '
600 ' Set up analog input (A/D) range and mode
610 '
620 OUTPUT$="chan 1, 0, :mode se;"
630 PRINT #1, OS + OUTPUT$
640 SPOLLBIT% = 7: GOSUB 2000
650 '
660 OUTPUT$="chan 1, 0, :gain 1;"
670 PRINT #1, OS + OUTPUT$
680 SPOLLBIT% = 7: GOSUB 2000
690 '
700 OUTPUT$="chan 1, 0, :range 10b;"
710 PRINT #1, OS + OUTPUT$
720 SPOLLBIT% = 7: GOSUB 2000
730 '
740 ' Set up analog output (D/A) channel 0
750 '
760 OUTPUT$="chan 4, 0, :range 10b;"
770 PRINT #1, OS + OUTPUT$
780 SPOLLBIT% = 7: GOSUB 2000
790 '
800 ' *****
810 '
820 ' Prompt for desired output voltage and write value to slot 4, channel 0
830 '
840 LOCATE 6,1:PRINT SPACE$(78)
850 LOCATE 6,1:INPUT "Output voltage, (-10 to +10, or 'Q' to quit) ";OV$
860 IF LEFT$(OV$,1) = "Q" OR LEFT$(OV$,1) = "q" THEN END
870 '
880 OUTPUT$ = "iwrite DCV 4, 0, " + OV$ + "; "
890 PRINT #1, OS + OUTPUT$
900 SPOLLBIT% = 7: GOSUB 2000
910 '
920 ' Do readings of AMM channel 0 and return as volts
930 '
940 OUTPUT$ = "syst :clock ?;iread dcv, 1, 0;"
950 PRINT #1, OS + OUTPUT$
960 SPOLLBIT% = 4: GOSUB 2000
970 '
980 ' Input time value from GPIB
990 PRINT #1, "ENTER O3"
1000 LINE INPUT #2, ATIM$
1010 '
1020 ' Input voltage value from GPIB
1030 PRINT #1, "ENTER O3"
1040 LINE INPUT #2, IV$
1050 '
1060 ' Do readings of AMM channel 0 and return as A/D counts
1070 '
1080 OUTPUT$ = "syst :clock ?; iread RAW, 1, 0;"
1090 PRINT #1, OS + OUTPUT$
1100 SPOLLBIT% = 4: GOSUB 2000
1110 '
1120 ' Input time/date value from GPIB

```

```

1130         PRINT #1, "ENTER 03"
1140         LINE INPUT #2, CTIM$
1150 '
1160 ' Input data value from GPIB
1170         PRINT #1, "ENTER 03"
1180         LINE INPUT #2, CTSS$
1190 '
1200 ' Write information to the screen
1210 '
1220 LOCATE 10,1: PRINT "Voltage written to analog output = ";OV$;" "
1230 LOCATE 12,1: PRINT "Input Voltage and time = ";ATIM$,IV$;" "
1240 LOCATE 13,1: PRINT "Raw counts and time = ";CTIM$,CTSS$;" "
1250 '
1260 GOTO 820
1270 '
1280 END
1290 '
2000 ' *****
2010 '
2020 ' Define a subroutine for checking the serial poll byte.
2030 ' The bit to be tested will be passed in as "spollbit%".
2040 ' The subroutine will return a value to the variable "bitstat%".
2050 ' The error bit will be tested automatically each time the subroutine
2060 ' is entered. If the error bit is set, program stops and prints message.
2070 '
2080 ' SPOLL BYTE / ERROR BIT TEST SUBROUTINE
2090 '
2100 SB% = 2 ^ SPOLLBIT%
2110 SP% = 0: BITSTAT% = 0
2120 LOCATE 25,1: PRINT "Testing SPOLL Bit";SPOLLBIT%;" Status = ";
2130 PRINT #1, "spoll 03"
2140 INPUT #2, SP%
2150 BITSTAT% = SP% AND SB%
2160 IF BITSTAT% = 0 THEN LOCATE 25,31: PRINT BITSTAT%;: GOTO 2130
2170 IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25,31: PRINT BITSTAT%;
2180 '
2190 ES% = SP% AND 32
2200 IF ES% = 0 THEN RETURN
2210 BEEP:PRINT #1, "Clear"
2220 PRINT #1, O$+"syst :err ?;"
2230 PRINT #1, "enter 03"
2240 INPUT #2, ERIN$
2250 LOCATE 25,1:PRINT "ERROR! - ";ERIN$;" - ";OUTP$;:LOCATE 20,1:END
2260 RETURN
2270 '
2280 ' *****

```

```

10 : *****
20 :
30 : DIGITAL.IOT - 576 QUICKSTART Digital I/O Program in Interpreter BASICA
40 : DATE: 7-1-90
50 :
60 : This program is dedsigned to run with the Driver488 GPIB driver software
70 : and a compatible GPIB interface card on an IBM PC/XT/AT or compatible.
80 : It assumes an address of "03" for the 576.
90 :
100 : (c) Copyright 1990 - Keithley Data Acquisition and Control
110 :
120 : *****
130 :
140 : CLS:CLOSE:KEY OFF
150 :
160 : Declare variables
170 :
180 : SPOLLBIT% = 0           ' Used as input to SPOLL check subroutine
190 : BITSTAT% = 0          ' Used as output from SPOLL check subroutine
200 : ADDR$ = "03"          ' Address of 576 as set on rear panel
210 : O$ = "output 03; "    ' Initial output string for Driver488
220 :
230 : Initialize Driver488 and interface for output.
240 :
250 :     OPEN "\dev\ieeeeout" FOR OUTPUT AS #1
260 :     IOCTL #1, "BREAK"
270 :     PRINT #1, "RESET"
280 :
290 : Initialize Driver488 and interface for input
300 :
310 :     OPEN "\dev\ieeeein" FOR INPUT AS #2
320 :
330 : Do some general GPIB housekeeping
340 :
350 :     PRINT #1, "FILL ERROR"
360 :     PRINT #1, "TIMEOUT 5"
370 :     PRINT #1, "CLEAR"
380 :
390 : System should respond with driver revision and date, indicating
400 : successful communication with driver and GPIB interface
410 :
420 :     PRINT #1, "HELLO"
430 :     LINE INPUT #2, A$
440 :     PRINT A$
450 :
460 :     LOCATE 3,1: PRINT"*** 576 QuickStart Digital I/O Test ***"
470 :
480 : Reset 576
490 :
500 :     OUTP$ = "reset all;x;"
510 :     PRINT #1, O$ + OUTP$
515 :     T!=TIMER: WHILE TIMER-T!<2: WEND      ' Delay for RESET ALL. See
516 :                                           ' RESET command for details.
520 :     SPOLLBIT% = 7: GOSUB 2000
530 :
540 : Set up digital port A for input
550 :
560 :     OUTP$="chan 5, 0, :mode in;"
570 :     PRINT #1, O$ + OUTP$
580 :     SPOLLBIT% = 7: GOSUB 2000
590 :
600 : Set up digital port B for output
610 :
620 :     OUTP$="chan 5, 1, :mode out;"
630 :     PRINT #1, O$ + OUTP$
640 :     SPOLLBIT% = 7: GOSUB 2000
650 :
660 : *****
670 :
680 : Loop to output data and make readings

```



```

690 '
700 LOCATE 5,1:PRINT "Press any key to stop...";
710 '
720 FOR PASS%=0 TO 255
730 '   write outval% to digital port B
740     OUTP$ = "iwrite raw 5, 1, " + STR$(PASS%) + "; "
750     PRINT #1, 0$ + OUTP$
760     SPOLLBIT% = 7: GOSUB 2000
770 '
780 '   read value at digital port A
790     OUTP$ = "iread raw, 5, 0;"
800     PRINT #1, 0$ + OUTP$
810     SPOLLBIT% = 4: GOSUB 2000
820 '
830 '   input data value from GPIB
840     PRINT #1, "ENTER 03"
850     LINE INPUT #2, A$
860 '
870 ' Print data to screen
880 '
890 LOCATE 10,1
900 PRINT "Written to 576 Digital Port B: ";PASS%;" "
910 PRINT "Read from 576 Digital Port A: ";VAL(A$);" "
920 '
930 I$ = INKEY$: IF I$ = "" THEN NEXT PASS%
940 '
950 END
960 '
2000 ' *****
2010 '
2020 ' Define a subroutine for checking the serial poll byte.
2030 ' The bit to be tested will be passed in as "spollbit%".
2040 ' The subroutine will return a value to the variable "bitstat%".
2050 ' The error bit will be tested automatically each time the subroutine
2060 ' is entered. If the error bit is set, program stops and prints message.
2070 '
2080 ' SPOLL BYTE / ERROR BIT TEST SUBROUTINE
2090 '
2100 SB% = 2 ^ SPOLLBIT%
2110 SP% = 0: BITSTAT% = 0
2120 LOCATE 25,1: PRINT "Testing SPOLL Bit";SPOLLBIT%;": Status = ";
2130 PRINT #1, "spoll-03"
2140 INPUT #2, SP%
2150 BITSTAT% = SP% AND SB%
2160 IF BITSTAT% = 0 THEN LOCATE 25,31: PRINT BITSTAT%;: GOTO 2130
2170 IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25,31: PRINT BITSTAT%;
2180 '
2190 ER% = SP% AND 32
2200 IF ER% = 0 THEN RETURN
2210 BEEP:PRINT #1, "Clear"
2220 PRINT #1, 0$+"sys:err ?;"
2230 PRINT #1, "enter 03"
2240 INPUT #2, ERIN$
2250 LOCATE 25,1:PRINT "ERROR! - ";ERIN$;" - ";OUTP$;:LOCATE 20,1:END
2260 RETURN
2270 '
2280 ' *****

```

QuickStart Programs for Interpreter BASIC, CEC Software Driver, and Compatible GPIB Interface

```

10 : *****
20 :
30 : ANALOG.CEC - 576 QUICKSTART Analog I/O Program in Interpreter BASICA
40 : DATE: 7-20-90
50 :
60 : This program is designed to run with the Capital Equipment Corporation
70 : driver software and a CEC GPIB interface card on an IBM PC/XT/AT or
80 : a compatible.
90 :
100 : It assumes an address of "03" for the 576.
110 :
120 : (c) Copyright 1990 - Keithley Data Acquisition and Control
130 :
140 : *****
150 :
160 : CLS : CLOSE : KEY OFF
170 :
180 : Declare variables
190 :
200 : SPOLLBIT% = 0           ' Used as input to SPOLL check subroutine
210 : BITSTAT% = 0           ' Used as output from SPOLL check subroutine
220 : K576% = 3              ' Address of 576 as set on rear panel
230 :
240 : Initialize CEC Driver and interface for output.
250 :
260 : DEF SEG = &HC400        ' Memory Address
270 : INITIALIZE = 0         ' INITIALIZE Subroutine Offset
280 : TRANSMIT = 3           ' TRANSMIT Subroutine Offset
290 : SEND = 9               ' SEND Subroutine Offset
300 : ENTER = 21            ' ENTER Subroutine Offset
310 : SPOLL = 12             ' SPOLL Subroutine Offset
320 : RECEIVE = 6           ' RECEIVE Subroutine Offset
330 :
340 : MY.ADDRESS% = 21       ' CEC-488 GPIB Address
350 : CONTROLLER% = 0       ' System control
360 :
370 : CALL INITIALIZE(MY.ADDRESS%, CONTROLLER%)
380 :
390 : Do some general GPIB housekeeping
400 :
410 : CMD$ = "IFC DCL REN MTA LISTEN 3"
420 : CALL TRANSMIT(CMD$, STATUS%)
430 : IF STATUS% <> 0 THEN PRINT "ERROR IN TRANSMISSION. PROGRAM HALTED.":END
440 :
450 : System should respond with 576 revision, indicating successful
460 : communication with driver, GPIB interface, and 576.
470 :
480 : OUTP$ = "Syst :IDN?;"
490 : GOSUB 3000              'Send output string
500 : SPOLLBIT%=4:GOSUB 2000
510 :
520 : GOSUB 4000
530 : LOCATE 1,1:PRINT RECV$
540 :
550 :
560 : LOCATE 3, 1: PRINT "**** 576 QuickStart Analog I/O Test ****"
570 :
580 : Reset 576
590 :
600 : OUTP$ = "reset all;x;"
610 : GOSUB 3000
615 : T!=TIMER: WHILE TIMER-T!<2: WEND      ' Delay for RESET ALL. See
616 :                                     ' RESET command for details.
620 : SPOLLBIT% = 7: GOSUB 2000
630 :
640 : Set data transmission format

```

```

650 '
660   OUTP$ = "syst :format ASCII;"
670   GOSUB 3000
680   SPOLLBIT% = 7: GOSUB 2000
690 '
700 ' Set up analog input (A/D) range and mode
710 '
720   OUTP$ = "chan 1, 0, :mode se;"
730   GOSUB 3000
740   SPOLLBIT% = 7: GOSUB 2000
750 '
760   OUTP$ = "chan 1, 0, :gain 1;"
770   GOSUB 3000
780   SPOLLBIT% = 7: GOSUB 2000
790 '
800   OUTP$ = "chan 1, 0, :range 10b;"
810   GOSUB 3000
820   SPOLLBIT% = 7: GOSUB 2000
830 '
840 ' Set up analog output (D/A) channel 0
850 '
860   OUTP$ = "chan 4, 0, :range 10b;"
870   GOSUB 3000
880   SPOLLBIT% = 7: GOSUB 2000
890 '
900 ' *****
910 '
920 ' Prompt for desired output voltage and write value to slot 4, channel 0
930 '
940   LOCATE 6, 1: PRINT SPACE$(78)
950   LOCATE 6, 1: INPUT "Output voltage, (-10 to +10, or 'Q' to quit) "; OV$
960   IF LEFT$(OV$,1) = "Q" OR LEFT$(OV$,1) = "q" THEN END
970 '
980   OUTP$ = "iwrite DCV 4, 0, " + OV$ + "; "
990   GOSUB 3000
1000  SPOLLBIT% = 7: GOSUB 2000
1010 '
1020 ' Do readings of AMM channel 0 and return as volts
1030 '
1040  OUTP$ = "syst :clock ?;iread dcv, 1, 0;"
1050  GOSUB 3000
1060  SPOLLBIT% = 4: GOSUB 2000
1070 '
1080 '
1090 ' Input time value from GPIB
1100   GOSUB 4000
1110   ATIM$=RECV$
1120 '
1130 ' Input voltage value from GPIB
1140   GOSUB 4000
1150   IV$=RECV$
1160 '
1170 ' Do readings of AMM channel 0 and return as A/D counts
1180 '
1190  OUTP$ = "syst :clock ?; iread RAW, 1, 0;"
1200  GOSUB 3000
1210  SPOLLBIT% = 4: GOSUB 2000
1220 '
1230 ' Input time/date value from GPIB
1240   GOSUB 4000
1250   CTIM$=RECV$
1260 '
1270 ' Input data value from GPIB
1280   GOSUB 4000
1290   CTSS$=RECV$
1300 '
1310 ' Write information to the screen
1320 '
1330  LOCATE 10, 1: PRINT "Voltage written to analog output = "; OV$; " "
1340  LOCATE 12, 1: PRINT "Input Voltage and time = "; ATIM$, IV$; " "

```

```

1350 LOCATE 13, 1: PRINT "Raw counts and time = "; CTIM$, CTSS; " "
1360 '
1370 GOTO 920
1380 '
1390 END
1400 '
2000 ' *****
2020 '
2040 ' Define a subroutine for checking the serial poll byte.
2060 ' The bit to be tested will be passed in as "spollbit%".
2080 ' The subroutine will return a value to the variable "bitstat%".
2100 ' The error bit will be tested automatically each time the subroutine
2120 ' is entered. If the error bit is set, program stops and prints message.
2140 '
2160 ' SPOLL BYTE / ERROR BIT TEST SUBROUTINE
2180 '
2200 SB% = 2 ^ SPOLLBIT%
2220 SP% = 0: BITSTAT% = 0
2240 LOCATE 25, 1: PRINT "Testing SPOLL Bit"; SPOLLBIT%; ": Status = ";
2260 CALL SPOLL(K576,SP%,STATUS%)
2280 '
2300 IF STATUS%=8 THEN PRINT "Timeout. Program Halted.....":END
2320 '
2340 BITSTAT% = SP% AND SB%
2360 IF BITSTAT% = 0 THEN LOCATE 25, 31: PRINT BITSTAT%; : GOTO 2260
2380 IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25, 31: PRINT BITSTAT%;
2400 '
2420 ER% = SP% AND 32
2440 IF ER% = 0 THEN RETURN
2460 CMD$ = "DCL MTA LISTEN 3"
2480 CALL TRANSMIT(CMD$, STATUS%)
2500 IF STATUS% <> 0 THEN PRINT "ERROR IN TRANSMISSION. PROGRAM HALTED.":END
2520 '
2540 CMD$ = "Syst :ERR?;"
2560 CALL SEND(K576, CMD$, STATUS%)
2580 IF STATUS% = 8 THEN PRINT "Timeout. Program Halted.....":END
2600 '
2620 ERIN$ = SPACES(255)
2640 CALL ENTER(ERIN$, LENGTH%, K576,STATUS%)
2660 IF STATUS% = 8 THEN PRINT "Timeout. Program Halted.....":END
2680 ERIN$ = LEFT$(ERIN$,LENGTH%)
2700 LOCATE 23,1: PRINT "ERROR! - "; ERIN$; " - "; OUTP$; : LOCATE 20, 1: END
2720 RETURN
2740 '
2760 '
3000 ' *****
3010 ' Define a subroutine for sending command strings to the 576. If the
3020 ' transmission times out, the program will halt, otherwise the program
3030 ' continues.
3040 ' *****
3050 '
3060 CALL SEND(K576, OUTP$, STATUS%)
3070 IF STATUS% = 8 THEN PRINT "Timeout. Program Halted.....":END
3080 RETURN
3090 '
3100 '
4000 ' *****
4010 ' Define a subroutine for receiving data from the 576. If the
4020 ' transmission times out, the program will halt, otherwise the program
4030 ' continues.
4040 ' *****
4050 '
4060 RECV$ = SPACES(255)
4070 CALL ENTER(RECV$, LENGTH%, K576,STATUS%)
4080 IF STATUS% = 8 THEN PRINT "Timeout. Program Halted.....":END
4090 RECV$ = LEFT$(RECV$, LENGTH%)
4100 RETURN
4110 '

```

```

10 | *****
20 |
30 | DIGITAL.CEC - 576 QUICKSTART Digital I/O Program in Interpreter BASICA
40 | DATE: 7-1-90
50 |
60 | This program is dedsigned to run with the Capital Equipment Corporation
70 | driver software and a CEC interface card on an IBM PC/XT/AT or
80 | a compatible.
90 |
100 | It assumes an address of "03" for the 576.
110 |
120 | (c) Copyright 1990 - Keithley Data Acquisition and Control
130 |
140 | *****
150 |
160 | CLS:CLOSE:KEY OFF
170 |
180 | Declare variables
190 |
200 | SPOLLBIT% = 0          ' Used as input to SPOLL check subroutine
210 | BITSTAT% = 0          ' Used as output from SPOLL check subroutine
220 | K576% = 3             ' Address of 576 as set on rear panel
230 |
240 | Initialize CEC Driver and interface
250 |
260 | DEF SEG = &HC400      'Memory Address
270 | INITIALIZE = 0        'INITIALIZE Subroutine Offset
280 | TRANSMIT = 3         'TRANSMIT Subroutine Offset
290 | SEND = 9             'SEND Subroutine Offset
300 | ENTER = 21           'ENTER Subroutine Offset
310 | SPOLL = 12           'SPOLL Subroutine Offset
320 | RECEIVE = 6          'RECEIVE Subroutine Offset
330 |
340 | MY.ADDRESS% = 21     'CEC-488 GPIB Address
350 | CONTROLLER% = 0     'System Controller
360 |
370 | CALL INITIALIZE(MY.ADDRESS%, CONTROLLER%)
380 |
390 | Do some general GPIB housekeeping
400 |
410 | CMD$ = "IFC DCL REN MTA LISTEN 3"
420 | CALL TRANSMIT(CMD$, STATUS%)
430 | IF STATUS% <> 0 THEN PRINT "ERROR IN TRANSMISSION. PROGRAM HALTED.":END
440 |
450 | System should respond with 576 revision, indicating successful
460 | communication with driver, GPIB interface, and 576.
470 |
480 | OUTP$ = "Syst :IDN?;"
490 | GOSUB 3000
500 | SPOLLBIT%=4:GOSUB 2000
510 |
520 | GOSUB 4000
530 | LOCATE 1,1:PRINT RECV$
540 |
550 |
560 | LOCATE 3,1: PRINT"*** 576 QuickStart Digital I/O Test ***"
570 |
580 | Reset 576
590 |
600 | OUTP$ = "reset all;x;"
610 | GOSUB 3000
615 | T!=TIMER: WHILE TIMER-T!<2: WEND      ' Delay for RESET ALL. See
616 |                                     ' RESET command for details.
620 | SPOLLBIT% = 7: GOSUB 2000
630 |
640 | Set up digital port A for input
650 |
660 | OUTP$="chan 5, 0, :mode in;"

```

```

670   GOSUB 3000
680   SPOLLBIT% = 7: GOSUB 2000
690 '
700 ' Set up digital port B for output
710 '
720   OUTP$="chan 5, 1, :mode out;"
730   GOSUB 3000
740   SPOLLBIT% = 7: GOSUB 2000
750 '
760 ' *****
770 '
780 ' Loop to output data and make readings
790 '
800   LOCATE 5,1:PRINT "Press any key to stop...";
810 '
820   FOR PASS%=0 TO 255
830     write outval% to digital port B
840       OUTP$ = "iwrite raw 5, 1, " + STR$(PASS%) + "; "
850       GOSUB 3000
860       SPOLLBIT% = 7: GOSUB 2000
870 '
880     read value at digital port A
890       OUTP$ = "iread raw, 5, 0;"
900       GOSUB 3000
910       SPOLLBIT% = 4: GOSUB 2000
920 '
930     input data value from GPIB
940       GOSUB 4000
950       A$=RECV$
960 '
970 ' Print data to screen
980 '
990   LOCATE 10,1
1000  PRINT "Written to 576 Digital Port B: ";PASS%;"  "
1010  PRINT "Read from 576 Digital Port A: ";VAL(A$);"  "
1020 '
1030  I$ = INKEY$: IF I$ = "" THEN NEXT PASS%
1040 '
1050 END
1060 '
2000 ' *****
2010 '
2020 ' Define a subroutine for checking the serial poll byte.
2030 ' The bit to be tested will be passed in as "spollbit%".
2040 ' The subroutine will return a value to the variable "bitstat%".
2050 ' The error bit will be tested automatically each time the subroutine
2060 ' is entered. If the error bit is set, program stops and prints message.
2070 '
2080 ' SPOLL BYTE / ERROR BIT TEST SUBROUTINE
2090 '
2100  SB% = 2 ^ SPOLLBIT%
2110  SP% = 0: BITSTAT% = 0
2120  LOCATE 25,1: PRINT "Testing SPOLL Bit";SPOLLBIT%;": Status = ";
2130  CALL SPOLL(K576%,SP%,STATUS%)
2140  '
2150  IF STATUS%=8 THEN PRINT "Timeout. Program Halted....":END
2160  '
2170  BITSTAT% = SP% AND SB%
2180  IF BITSTAT% = 0 THEN LOCATE 25,31: PRINT BITSTAT%;: GOTO 2130
2190  IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25,31: PRINT BITSTAT%;
2200 '
2210  ER% = SP% AND 32
2220  IF ER% = 0 THEN RETURN
2230  CMD$ = "DCL MTA LISTEN 3"
2240  CALL TRANSMIT(CMD$, STATUS%)
2250  IF STATUS% <> 0 THEN PRINT "ERROR IN TRANSMISSION. PROGRAM HALTED.":END
2260 '
2270  CMD$ = "SYST :ERR?:"
2280  CALL ENTER(ERIN$, LENGTH%, K576%, STATUS%)
2290  IF STATUS%=8 THEN PRINT "Timeout. Program Halted....":END

```

```

2300 ERIN$ = LEFT$(ERIN$,LENGTH%)
2310 LOCATE 23,1:PRINT "ERROR! - ";ERIN$;" - ";OUTP$;:LOCATE 20,1:END
2320 RETURN
2330 '
2340 '
3000 ' *****
3010 ' Define a subroutine for sending command strings to the 576. If the
3020 ' transmission times out, the program will halt, otherwise the program
3030 ' continues.
3040 ' *****
3050 '
3060 CALL SEND(K576%, OUTP$, STATUS%)
3070 IF STATUS%=8 THEN PRINT "Timeout. Program Halted....":END
3080 RETURN
3090 '
3100 '
4000 ' *****
4010 ' Define a subroutine for receiving data from the 576. If the
4020 ' transmission times out, the program will halt, otherwise the program
4030 ' continues.
4040 ' *****
4050 '
4060 RECV$ = SPACE$(255)
4070 CALL ENTER(RECV$, LENGTH%, K576%, STATUS%)
4080 IF STATUS%=8 THEN PRINT "Timeout. Program Halted....":END
4090 RECV$ = LEFT$(RECV$, LENGTH%)
4100 RETURN
4110 '
4120 ' *****

```

QuickStart Programs for Interpreter BASIC, National Software Driver, and Compatible GPIB Interface

```

10 ' *****
20 '
30 ' ANALOG.NAT - 576 QUICKSTART Analog I/O Program in Interpreter BASIC
40 ' DATE: 7-25-90
50 '
60 ' This program is designed to run with the National Instruments driver
70 ' software and a National GPIB interface card on an IBM PC/XT/AT or
80 ' a compatible.
90 '
100 ' It assumes an address of "03" for the 576.
110 '
120 ' (c) Copyright 1990 - Keithley Data Acquisition and Control
130 '
140 ' *****
150 '
160 ' CLS : CLOSE : KEY OFF
170 '
180 ' National Instruments Header
190 '
200 ' CLEAR , 16000! ' BASIC Declarations
210 ' IBINIT1 = 16000!
220 ' IBINIT2 = IBINIT1 + 3
230 ' BLOAD "c:\bib.m", IBINIT1

240 CALL IBINIT1(IBFIND, IBTRG, IBCLR, IBPCT, IBSIC, IBLOC, IBPPC, IBBNA, IBONL, IBRSC, IBSRE, IBRSV, IBPAD,
IBSAD, IBIST, IBDMA, IBEOS, IBTMO, IBEOT, IBRDF, IBWRTF)

250 CALL IBINIT2(IBGTS, IBCAC, IBWAIT, IBPOKE, IBWRT, IBWRTA, IBCMD, IBCMDA, IBRD, IBRDA, IBSTOP, IBRPP,
IBRSP, IBDIAG, IBXTRC, IBRDI, IBWRTI, IBRDIA, IBWRTIA, IBSTA%, IBERR%, IBCNT%)

260 '
270 ' Declare variables
280 '
290 ' SPOLLBIT% = 0 ' Used as input to SPOLL check subroutine
300 ' BITSTAT% = 0 ' Used as output from SPOLL check subroutine
310 ' K576% = 3 ' Address of 576 as set on rear panel
320 '
330 ' Initialize National driver and interface for output.
340 '
350 ' NA$ = "GPIB0": CALL IBFIND(NA$, BRD0%) 'find board descriptor
360 ' NA$ = "DEV1": CALL IBFIND(NA$, K576%) 'find instrument descr.
370 ' V% = 12: CALL IBTMO(K576%, V%) 'set 3 sec timeout
380 ' V% = 3: CALL IBPAD(K576%, V%) 'set primary addr to 3
390 ' V% = 1: CALL IBSRE(BRD0%, V%) 'set REN true
400 '
410 ' Do some general GPIB housekeeping
420 '
430 ' CALL IBSIC(K576%) 'send IFC
440 ' CALL IBCLR(K576%) 'send SDC
450 '
460 ' System should respond with 576 revision, indicating successful
470 ' communication with driver, GPIB interface, and 576.
480 '
490 ' OUTP$ = "Syst :IDN?;"
500 ' GOSUB 3000 ' sends commands to 576
510 ' SPOLLBIT% = 4: GOSUB 2000 ' tests serial poll byte
520 '
530 ' GOSUB 4000 ' reads data from 576
540 ' LOCATE 1, 1: PRINT RECV$
550 '
560 '
570 ' LOCATE 3, 1: PRINT "*** 576 QuickStart Analog I/O Test ***"
580 '
590 ' Reset 576
600 '
610 ' OUTP$ = "reset all;x;"

```



```

620     GOSUB 3000
625     T!=TIMER: WHILE TIMER-T!<2: WEND           ' Delay for RESET ALL. See
626                                           ' RESET command for details.
630     SPOLLBIT% = 7: GOSUB 2000
640 '
650 ' Set data transmission format
660 '
670     OUTP$ = "syst :format ASCII;"
680     GOSUB 3000
690     SPOLLBIT% = 7: GOSUB 2000
700 '
710     OUTP$ = "syst :term NONE;"
720     GOSUB 3000
730     SPOLLBIT% = 7: GOSUB 2000
740 '
750 ' Set up analog input (A/D) range and mode
760 '
770     OUTP$ = "chan 1, 0, :mode se;"
780     GOSUB 3000
790     SPOLLBIT% = 7: GOSUB 2000
800 '
810     OUTP$ = "chan 1, 0, :gain 1;"
820     GOSUB 3000
830     SPOLLBIT% = 7: GOSUB 2000
840 '
850     OUTP$ = "chan 1, 0, :range 10b;"
860     GOSUB 3000
870     SPOLLBIT% = 7: GOSUB 2000
880 '
890 ' Set up analog output (D/A) channel 0
900 '
910     OUTP$ = "chan 4, 0, :range 10b;"
920     GOSUB 3000
930     SPOLLBIT% = 7: GOSUB 2000
940 '
950 ' *****
960 '
970 ' Prompt for desired output voltage and write value to slot 4, channel 0
980 '
990     LOCATE 6, 1: PRINT SPACES(78)
1000    LOCATE 6, 1: INPUT "Output voltage, (-10 to +10, or 'Q' to quit) "; OV$
1010    IF LEFT$(OV$,1) = "Q" OR LEFT$(OV$,1) = "q" THEN GOTO 1470
1020 '
1030    OUTP$ = "iwrite DCV 4, 0, " + OV$ + "; "
1040    GOSUB 3000
1050    SPOLLBIT% = 7: GOSUB 2000
1060 '
1070 ' Do readings of AMM channel 0 and return as volts
1080 '
1090    OUTP$ = "syst :clock ?;iread dcv, 1, 0;"
1100    GOSUB 3000
1110    SPOLLBIT% = 4: GOSUB 2000
1120 '
1130 ' Input time value from GPIB
1140 '
1150         GOSUB 4000
1160         ATIM$ = RECV$
1170 '
1180 ' Input voltage value from GPIB
1190 '
1200         GOSUB 4000
1210         IV$ = RECV$
1220 '
1230 ' Do readings of AMM channel 0 and return as A/D counts
1240 '
1250    OUTP$ = "syst :clock ?; iread RAW, 1, 0;"
1260    GOSUB 3000
1270    SPOLLBIT% = 4: GOSUB 2000
1280 '
1290 ' Input time/date value from GPIB

```

```

1300 '
1310         GOSUB 4000
1320         CTIM$ = RECV$
1330 '
1340 ' Input data value from GPIB
1350 '
1360         GOSUB 4000
1370         CTS$ = RECV$
1380 '
1390 ' Write information to the screen
1400 '
1410 LOCATE 10, 1: PRINT "Voltage written to analog output = "; OV$; "      "
1420 LOCATE 12, 1: PRINT "Date, time and voltage = "; ATIM$, IV$; "      "
1430 LOCATE 13, 1: PRINT "Date, time and raw counts = "; CTIM$, CTS$; "      "
1440 '
1450 GOTO 970
1460 '
1470 V% = 0: CALL IBONL(K576%, V%)           'close instrument file
1480 CALL IBONL(BRD0%, V%)                 'close board file
1490 LOCATE 20,1
1500 END
1510 '
2000 ' *****
2010 '
2020 ' Define a subroutine for checking the serial poll byte.
2030 ' The bit to be tested will be passed in as "spollbit%".
2040 ' The subroutine will return a value to the variable "bitstat%".
2050 ' The error bit will be tested automatically each time the subroutine
2060 ' is entered. If the error bit is set, program stops and prints message.
2070 '
2080 ' SPOLL BYTE / ERROR BIT TEST SUBROUTINE
2090 '
2100 SB% = 2 ^ SPOLLBIT%
2110 SP% = 0: BITSTAT% = 0
2120 LOCATE 25, 1: PRINT "Testing SPOLL Bit"; SPOLLBIT%; ": Status = ";
2130 CALL IBRSP(K576%, SP%)
2140 IF IBSTA% < 0 THEN PRINT "Error reading serial poll byte.": STOP
2150 '
2160 BITSTAT% = SP% AND SB%
2170 IF BITSTAT% = 0 THEN LOCATE 25, 31: PRINT BITSTAT%; ": GOTO 2130
2180 IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25, 31: PRINT BITSTAT%;
2190 '
2200 ER% = SP% AND 32           'check for 576 error
2210 IF ER% = 0 THEN RETURN
2220 '
2230 CALL IBCLR(K576%)         'send SDC
2240 '
2250 CMD$ = "Syst :ERR?;"
2260 CALL IBWRT(K576%, CMD$)   'send command string
2270 IF IBSTA% < 0 THEN PRINT "Error sending " + CMD$: STOP
2280 '
2290 ERIN$ = SPACE$(255)
2300 CALL IBRD(K576%, ERIN$)
2310 IF IBSTA% < 0 THEN PRINT "Error reading err$." : STOP
2320 ERIN$ = LEFT$(ERIN$, IBCNT%)
2330 LOCATE 23, 1: PRINT "ERROR! - "; ERIN$; " - "; OUTP$;
2340 RETURN
2350 '
2360 '
3000 ' *****
3010 ' Define a subroutine for sending command strings to the 576. If the
3020 ' transmission times out, the program will halt, otherwise the program
3030 ' continues.
3040 ' *****
3050 '
3060 CALL IBWRT(K576%, OUTP$)   'send command string
3070 IF IBSTA% < 0 THEN PRINT "Error sending " + OUTP$: STOP
3080 RETURN
3090 '
3100 '

```

```

4000 ' *****
4010 ' Define a subroutine for receiving data from the 576. If the
4020 ' transmission times out, the program will halt, otherwise the program
4030 ' continues.
4040 ' *****
4050 '
4060   RECV$ = SPACE$(255)
4070   CALL IBRD(K576%, RECV$)           'read from instrument
4080   IF IBSTA% < 0 THEN PRINT "Error reading from device.": STOP
4090   RECV$ = LEFT$(RECV$, IBCNT%)
4100   RETURN
4110 '
4120 ' *****

```

```

10 ' *****
20 '
30 ' DIGITAL.NAT - 576 QUICKSTART Analog I/O Program in Interpreter BASIC
40 ' DATE: 7-25-90
50 '
60 ' This program is designed to run with the National Instruments driver
70 ' software and a National GPIB interface card on an IBM PC/XT/AT or
80 ' a compatible.
90 '

```

```

100 ' It assumes an address of "03" for the 576.

```

```

110 '
120 ' (c) Copyright 1990 - Keithley Data Acquisition and Control

```

```

130 '
140 ' *****

```

```

150 '
160   CLS : CLOSE : KEY OFF

```

```

170 '
180 ' National Instruments Header
190 '

```

```

200   CLEAR , 16000!
210   IBINIT1 = 16000!
220   IBINIT2 = IBINIT1 + 3
230   BLOAD "c:\bib.m", IBINIT1

```

```

240 CALL IBINIT1(IBFIND, IBTRG, IBCLR, IBPCT, IBSIC, IBLOC, IBPPC, IBBNA, IBONL, IBRSC, IBSRE, IBRSV, IBPAD,
IBSAD, IBIST, IBDMA, IBEOS, IBTMO, IBEOT, IBRDF, IBWRTF)

```

```

250 CALL IBINIT2(IBGTS, IBCAC, IBWAIT, IBPOKE, IBWRT, IBWRTA, IBCMD, IBCMDA, IBRD, IBRDA, IBSTOP, IBRPP,
IBRSP, IBDIAG, IBXTRC, IBRDI, IBWRTI, IBRDIA, IBWRTIA, IBSTA%, IBERR%, IBCNT%)

```

```

260 '
270 ' Declare variables

```

```

280 '
290   SPOLLBIT% = 0           ' Used as input to SPOLL check subroutine
300   BITSTAT% = 0          ' Used as output from SPOLL check subroutine
310   K576% = 3              ' Address of 576 as set on rear panel

```

```

320 '
330 ' Initialize National driver and interface for output.

```

```

340 '
350   NAS = "GPIB0": CALL IBFIND(NAS, BRD0%)   'find board descriptor
360   NAS = "DEV1": CALL IBFIND(NAS, K576%)   'find instrument descr.
370   V% = 12: CALL IBTMO(K576%, V%)         'set 3 sec timeout
380   V% = 3: CALL IBPAD(K576%, V%)         'set primary addr to 3
390   V% = 1: CALL IBSRE(BRD0%, V%)         'set REN true

```

```

400 '
410 ' Do some general GPIB housekeeping

```

```

420 '
430   CALL IBSIC(K576%)           'send IFC
440   CALL IBCLR(K576%)         'send SDC

```

```

450 '
460 ' System should respond with 576 revision, indicating successful
470 ' communication with driver, GPIB interface, and 576.

```

```

480 '
490   OUTP$ = "Syst :IDN?;"

```

```

500 GOSUB 3000 ' sends commands to 576
510 SPOLLBIT% = 4: GOSUB 2000 ' tests serial poll byte
520 '
530 GOSUB 4000 ' reads data from 576
540 LOCATE 1, 1: PRINT RECV$
550 '
560 '
570 LOCATE 3, 1: PRINT "*** 576 QuickStart Analog I/O Test ***"
580 '
590 ' Reset 576
600 '
610 OUTP$ = "reset all;x;"
620 GOSUB 3000
625 T!=TIMER: WHILE TIMER-T!<2: WEND ' Delay for RESET ALL. See
626 ' RESET command for details.
630 SPOLLBIT% = 7: GOSUB 2000
635 '
640 ' Set up digital port A for input
650 '
660 OUTP$ = "chan 5, 0, :mode in;"
670 GOSUB 3000
680 SPOLLBIT% = 7: GOSUB 2000
690 '
700 ' Set up digital port B for output
710 '
720 OUTP$ = "chan 5, 1, :mode out;"
730 GOSUB 3000
740 SPOLLBIT% = 7: GOSUB 2000
750 '
760 ' *****
770 '
780 ' Loop to output data and make readings
790 '
800 LOCATE 5, 1: PRINT "Press any key to stop...";
810 '
820 FOR PASS% = 0 TO 255
830 ' write outval% to digital port B
840 OUTP$ = "iwrite raw 5, 1, " + STR$(PASS%) + "; "
850 GOSUB 3000
860 SPOLLBIT% = 7: GOSUB 2000
870 '
880 ' read value at digital port A
890 OUTP$ = "iread raw, 5, 0;"
900 GOSUB 3000
910 SPOLLBIT% = 4: GOSUB 2000
920 '
930 ' input data value from GPIB
940 GOSUB 4000
950 A$ = RECV$
960 '
970 ' Print data to screen
980 '
990 LOCATE 10, 1
1000 PRINT "Written to 576 Digital Port B: "; PASS%; " "
1010 PRINT "Read from 576 Digital Port A: "; VAL(A$); " "
1020 '
1030 I$ = INKEY$: IF I$ = "" THEN NEXT PASS%
1040 '
1470 V% = 0: CALL IBONL(K576%, V%) 'close instrument file
1480 CALL IBONL(BRD0%, V%) 'close board file
1490 LOCATE 20, 1
1500 END
1510 '
2000 ' *****
2010 '
2020 ' Define a subroutine for checking the serial poll byte.
2030 ' The bit to be tested will be passed in as "spollbit%".
2040 ' The subroutine will return a value to the variable "bitstat%".
2050 ' The error bit will be tested automatically each time the subroutine
2060 ' is entered. If the error bit is set, program stops and prints message.

```

```

2070 '
2080 ' SPOLL BYTE / ERROR BIT TEST SUBROUTINE
2090 '
2100   SB% = 2 ^ SPOLLBIT%
2110   SP% = 0; BITSTAT% = 0
2120   LOCATE 25, 1: PRINT "Testing SPOLL Bit"; SPOLLBIT%; ": Status = ";
2130   CALL IBRSP(K576%, SP%)
2140   IF IBSTA% < 0 THEN PRINT "Error reading serial poll byte.": STOP
2150 '
2160   BITSTAT% = SP% AND SB%
2170   IF BITSTAT% = 0 THEN LOCATE 25, 31: PRINT BITSTAT%; : GOTO 2130
2180   IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25, 31: PRINT BITSTAT%;
2190 '
2200   ER% = SP% AND 32                                'check for 576 error
2210   IF ER% = 0 THEN RETURN
2220 '
2230   CALL IBCLR(K576%)                                'send SDC
2240 '
2250   CMD$ = "Syst :ERR?;"
2260   CALL IBWRT(K576%, CMD$)                          'send command string
2270   IF IBSTA% < 0 THEN PRINT "Error sending " + CMD$: STOP
2280   '
2290   ERIN$ = SPACE$(255)
2300   CALL IBRD(K576%, ERIN$)
2310   IF IBSTA% < 0 THEN PRINT "Error reading err$." : STOP
2320   ERIN$ = LEFT$(ERIN$, IBCNT%)
2330   LOCATE 23, 1: PRINT "ERROR! - "; ERIN$; " - "; OUTP$;
2340 RETURN
2350 '
2360 '
3000 ' *****
3010 ' Define a subroutine for sending command strings to the 576. If the
3020 ' transmission times out, the program will halt, otherwise the program
3030 ' continues.
3040 ' *****
3050 '
3060   CALL IBWRT(K576%, OUTP$)                          'send command string
3070   IF IBSTA% < 0 THEN PRINT "Error sending " + OUTP$: STOP
3080   RETURN
3090 '
3100 '
4000 ' *****
4010 ' Define a subroutine for receiving data from the 576. If the
4020 ' transmission times out, the program will halt, otherwise the program
4030 ' continues.
4040 ' *****
4050 '
4060   RECV$ = SPACE$(255)
4070   CALL IBRD(K576%, RECV$)                          'read from instrument
4080   IF IBSTA% < 0 THEN PRINT "Error reading from device.": STOP
4090   RECV$ = LEFT$(RECV$, IBCNT%)
4100   RETURN
4110 '
4120 ' *****

```

QuickStart Programs for Interpreter BASIC, Metrabyte Software Driver, and Compatible GPIB Interface

```
10 ' *****
20 '
30 ' ANALOG.MBC - 576 QUICKSTART Analog I/O Program in Interpreter BASIC
40 ' DATE: 7-27-90
50 '
60 ' This program is designed to run with the Metrabyte DOS resident driver
70 ' and a Metrabyte MBC-488 GPIB interface card on an IBM PC/XT/AT or
80 ' a compatible.
90 '
100 ' It assumes an address of "3" for the 576.
110 '
120 ' (c) Copyright 1990 - Keithley Data Acquisition and Control
130 '
140 ' *****
150 '
160 CLS : CLOSE : KEY OFF
170 ON ERROR GOTO 5000
180 '
190 ' Declare variables
200 '
210 SPOLLBIT% = 0 ' Used as input to SPOLL check subroutine
220 BITSTAT% = 0 ' Used as output from SPOLL check subroutine
230 '
240 ' Establish communication with device driver.
250 '
260 OPEN "$DV488" FOR OUTPUT AS #1
270 PRINT #1, "BUFFERCLEAR"
280 OPEN "$DV488" FOR INPUT AS #2
290 '
300 ' Initialize MBC-488 board using "SYSCON" command.
310 '
320 PRINT #1, "SYSCON MAD1=0 CIC1=1 BA1=&H300" ' board addr at 300 Hex
330 '
340 ' Do some general GPIB housekeeping
350 '
360 A% = 100
370 PRINT #1, "TIMEOUT", A% ' set timeout A% * 56ms
380 PRINT #1, "CLEAR 3" ' clear device
390 PRINT #1, "REMOTE 3" ' 576 in remote
400 '
410 ' System should respond with 576 revision, indicating successful
420 ' communication with driver, GPIB interface, and 576.
430 '
440 OUTP$ = "Syst :IDN?;"
450 GOSUB 3000 ' sends commands to 576
460 SPOLLBIT% = 4: GOSUB 2000 ' tests serial poll byte
470 '
480 GOSUB 4000 ' reads data from 576
490 LOCATE 1, 1: PRINT RECV$
500 '
510 '
520 LOCATE 3, 1: PRINT "**** 576 QuickStart Analog I/O Test ****"
530 '
540 ' Reset 576
550 '
560 OUTP$ = "reset all;x;"
570 GOSUB 3000
575 T!=TIMER: WHILE TIMER-T!<2: WEND ' Delay for RESET ALL. See
576 ' RESET command for details.
580 SPOLLBIT% = 7: GOSUB 2000
590 '
600 ' Set data transmission format
610 '
620 OUTP$ = "syst :format ASCII;"
630 GOSUB 3000
640 SPOLLBIT% = 7: GOSUB 2000
```

```

650 '
660 ' Set up analog input (A/D) range and mode
670 '
680   OUTP$ = "chan 1, 0, :mode se;"
690   GOSUB 3000
700   SPOLLBIT% = 7: GOSUB 2000
710 '
720   OUTP$ = "chan 1, 0, :gain 1;"
730   GOSUB 3000
740   SPOLLBIT% = 7: GOSUB 2000
750 '
760   OUTP$ = "chan 1, 0, :range 10b;"
770   GOSUB 3000
780   SPOLLBIT% = 7: GOSUB 2000
790 '
800 ' Set up analog output (D/A) channel 0
810 '
820   OUTP$ = "chan 4, 0, :range 10b;"
830   GOSUB 3000
840   SPOLLBIT% = 7: GOSUB 2000
850 '
860 ' *****
870 '
880 ' Prompt for desired output voltage and write value to slot 4, channel 0
890 '
900   LOCATE 6, 1: PRINT SPACE$(78)
910   LOCATE 6, 1: INPUT "Output voltage, (-10 to +10, or 'Q' to quit) "; OV$
920   IF OV$ = "Q" OR OV$ = "q" THEN GOTO 1380
930 '
940   OUTP$ = "iwrite DCV 4, 0, " + OV$ + "; "
950   GOSUB 3000
960   SPOLLBIT% = 7: GOSUB 2000
970 '
980 ' Do readings of AMM channel 0 and return as volts
990 '
1000  OUTP$ = "syst :clock ?; iread dcv, 1, 0;"
1010  GOSUB 3000
1020  SPOLLBIT% = 4: GOSUB 2000
1030 '
1040 ' Input time value from GPIB
1050 '
1060  GOSUB 4000
1070  ATIM$ = RECV$
1080 '
1090 ' Input voltage value from GPIB
1100 '
1110  GOSUB 4000
1120  IV$ = RECV$
1130 '
1140 ' Do readings of AMM channel 0 and return as A/D counts
1150 '
1160  OUTP$ = "syst :clock ?; iread RAW, 1, 0;"
1170  GOSUB 3000
1180  SPOLLBIT% = 4: GOSUB 2000
1190 '
1200 ' Input time/date value from GPIB
1210 '
1220  GOSUB 4000
1230  CTIM$ = RECV$
1240 '
1250 ' Input data value from GPIB
1260 '
1270  GOSUB 4000
1280  CTSS$ = RECV$
1290 '
1300 ' Write information to the screen
1310 '
1320  LOCATE 10, 1: PRINT "Voltage written to analog output = "; OV$; "   "
1330  LOCATE 12, 1: PRINT "Date, time and voltage = "; ATIM$, IV$; "   "
1340  LOCATE 13, 1: PRINT "Date, time and raw counts = "; CTIM$, CTSS$; "   "

```

```

1350 '
1360 GOTO 880
1370 '
1380 CLOSE
1390 LOCATE 20, 1
1400 END
1410 '
2000 ' *****
2010 '
2020 ' Define a subroutine for checking the serial poll byte.
2030 ' The bit to be tested will be passed in as "spollbit%".
2040 ' The subroutine will return a value to the variable "bitstat%".
2050 ' The error bit will be tested automatically each time the subroutine
2060 ' is entered. If the error bit is set, program stops and prints message.
2070 '
2080 ' SPOLL BYTE / ERROR BIT TEST SUBROUTINE
2090 '
2100 SB% = 2 ^ SPOLLBIT%
2110 SP% = 0: BITSTAT% = 0
2120 LOCATE 25, 1: PRINT "Testing SPOLL Bit"; SPOLLBIT%; ": Status = ";
2130 PRINT #1, "STATUS 3"
2140 INPUT #2, SP%
2150 '
2160 BITSTAT% = SP% AND SB%
2170 IF BITSTAT% = 0 THEN LOCATE 25, 31: PRINT BITSTAT%; : GOTO 2130
2180 IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25, 31: PRINT BITSTAT%;
2190 '
2200 ER% = SP% AND 32 'check for 576 error
2210 IF ER% = 0 THEN RETURN
2220 '
2230 PRINT #1, "CLEAR 3" 'send SDC
2240 '
2250 CMD$ = "Syst :ERR?;"
2260 PRINT #1, "OUTPUT 3 $ +", CMD$
2270 '
2280 ERIN$ = SPACE$(255)
2290 PRINT #1, "ENTER 3 $"
2300 INPUT #2, ERIN$
2310 LOCATE 23, 1: PRINT "ERROR! - "; ERIN$; " - "; OUTP$;
2320 RETURN
2330 '
2340 '
3000 ' *****
3010 ' Define a subroutine for sending command strings to the 576. If the
3020 ' transmission times out, the program will halt, otherwise the program
3030 ' continues.
3040 ' *****
3050 '
3060 PRINT #1, "OUTPUT 3 $ +", OUTP$
3070 RETURN
3080 '
4000 ' *****
4010 ' Define a subroutine for receiving data from the 576. If the
4020 ' transmission times out, the program will halt, otherwise the program
4030 ' continues.
4040 ' *****
4050 '
4060 RECV$ = SPACE$(255)
4070 PRINT #1, "ENTER 3 $"
4080 LINE INPUT #2, RECV$
4090 RETURN
4100 '
5000 ' *****
5010 ' Define a subroutine for handling BASIC or interface errors. (All
5020 ' MBC-488 errors are handled in DOS by the resident driver.)
5030 ' *****
5040 '
5050 IF (ERR <> 68) AND (ERR <> 57) THEN LOCATE 15,1: PRINT "BASIC ERROR # "; ERR; " IN LINE "; ERL
5060 INPUT #2, E$
5070 PRINT "$DV488 driver returned error number -", E$

```



```

5080 INPUT #2, E$
5090 PRINT E$
5100 INPUT #2, E$
5110 PRINT E$
5120 STOP
5130 : *****

```

```

10 : *****
20 :
30 : DIGITAL.MBC - 576 QUICKSTART Analog I/O Program in Interpreter BASIC
40 : DATE: 7-27-90
50 :
60 : This program is designed to run with the Metrabyte DOS resident driver
70 : and a Metrabyte MBC-488 GPIB interface card on an IBM PC/XT/AT or
80 : a compatible.
90 :
100 : It assumes an address of "3" for the 576.
110 :
120 : (c) Copyright 1990 - Keithley Data Acquisition and Control
130 :
140 : *****
150 :
160 CLS : CLOSE : KEY OFF
170 ON ERROR GOTO 5000
180 :
190 : Declare variables
200 :
210 SPOLLBIT% = 0 ' Used as input to SPOLL check subroutine
220 BITSTAT% = 0 ' Used as output from SPOLL check subroutine
230 :
240 : Establish communication with device driver.
250 :
260 OPEN "$DV488" FOR OUTPUT AS #1
270 PRINT #1, "BUFFERCLEAR"
280 OPEN "$DV488" FOR INPUT AS #2
290 :
300 : Initialize MBC-488 board using "SYSICON" command.
310 :
320 PRINT #1, "SYSICON MAD1=0 CIC1=1 BA1=&H300" ' board addr at 300 Hex
330 :
340 : Do some general GPIB housekeeping
350 :
360 A% = 100
370 PRINT #1, "TIMEOUT", A% ' set timeout A% * 56ms
380 PRINT #1, "CLEAR 3" ' clear device
390 PRINT #1, "REMOTE 3" ' 576 in remote
400 :
410 : System should respond with 576 revision, indicating successful
420 : communication with driver, GPIB interface, and 576.
430 :
440 OUTP$ = "Syst :IDN?;"
450 GOSUB 3000 ' sends commands to 576
460 SPOLLBIT% = 4: GOSUB 2000 ' tests serial poll byte
470 :
480 GOSUB 4000 ' reads data from 576
490 LOCATE 1, 1: PRINT RECV$
500 :
510 :
520 LOCATE 3, 1: PRINT "**** 576 QuickStart Analog I/O Test ****"
530 :
540 : Reset 576
550 :
560 OUTP$ = "reset all;x;"
570 GOSUB 3000
575 T!=TIMER: WHILE TIMER-T!<2: WEND ' Delay for RESET ALL. See
576 ' RESET command for details.
580 SPOLLBIT% = 7: GOSUB 2000
590 :

```

```

635 '
640 ' Set up digital port A for input
650 '
660 '   OUTP$ = "chan 5, 0, :mode in;"
670 '   GOSUB 3000
680 '   SPOLLBIT% = 7: GOSUB 2000
690 '
700 ' Set up digital port B for output
710 '
720 '   OUTP$ = "chan 5, 1, :mode out;"
730 '   GOSUB 3000
740 '   SPOLLBIT% = 7: GOSUB 2000
750 '
760 ' *****
770 '
780 ' Loop to output data and make readings
790 '
800 '   LOCATE 5, 1: PRINT "Press any key to stop...";
810 '
820 '   FOR PASS% = 0 TO 255
830 '       write outval% to digital port B
840 '           OUTP$ = "iwrite raw 5, 1, " + STR$(PASS%) + "; "
850 '           GOSUB 3000
860 '           SPOLLBIT% = 7: GOSUB 2000
870 '
880 '       read value at digital port A
890 '           OUTP$ = "iread raw, 5, 0;"
900 '           GOSUB 3000
910 '           SPOLLBIT% = 4: GOSUB 2000
920 '
930 '       input data value from GPIB
940 '           GOSUB 4000
950 '           A$ = RECV$
960 '
970 ' Print data to screen
980 '
990 '   LOCATE 10, 1
1000 '   PRINT "Written to 576 Digital Port B: "; PASS%; " "
1010 '   PRINT "Read from 576 Digital Port A: "; VAL(A$); " "
1020 '
1030 '   I$ = INKEY$: IF I$ = "" THEN NEXT PASS%
1040 '
1480 '   CLOSE
1490 '   LOCATE 20, 1
1500 ' END
1510 '
2000 ' *****
2010 '
2020 ' Define a subroutine for checking the serial poll byte.
2030 ' The bit to be tested will be passed in as "spollbit%".
2040 ' The subroutine will return a value to the variable "bitstat%".
2050 ' The error bit will be tested automatically each time the subroutine
2060 ' is entered. If the error bit is set, program stops and prints message.
2070 '
2080 ' SPOLL BYTE / ERROR BIT TEST SUBROUTINE
2090 '
2100 '   SB% = 2 ^ SPOLLBIT%
2110 '   SP% = 0: BITSTAT% = 0
2120 '   LOCATE 25, 1: PRINT "Testing SPOLL Bit"; SPOLLBIT%; ": Status = ";
2130 '   PRINT #1, "STATUS 3"
2140 '   INPUT #2, SP%
2150 '
2160 '   BITSTAT% = SP% AND SB%
2170 '   IF BITSTAT% = 0 THEN LOCATE 25, 31: PRINT BITSTAT%; : GOTO 2130
2180 '   IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25, 31: PRINT BITSTAT%;
2190 '
2200 '   ER% = SP% AND 32           'check for 576 error
2210 '   IF ER% = 0 THEN RETURN
2220 '
2230 '   PRINT #1, "CLEAR 3"       'send SDC

```

```

2240 '
2250 CMD$ = "Syst :ERR?;"
2260 PRINT #1, "OUTPUT 3 $ +", CMD$
2270 '
2280 ERIN$ = SPACE$(255)
2290 PRINT #1, "ENTER 3 $"
2300 INPUT #2, ERIN$
2310 LOCATE 23, 1: PRINT "ERROR! - "; ERIN$; " - "; OUTP$;
2320 RETURN
2330 '
2340 '
3000 ' *****
3010 ' Define a subroutine for sending command strings to the 576. If the
3020 ' transmission times out, the program will halt, otherwise the program
3030 ' continues.
3040 ' *****
3050 '
3060 PRINT #1, "OUTPUT 3 $ +", OUTP$
3070 RETURN
3080 '
4000 ' *****
4010 ' Define a subroutine for receiving data from the 576. If the
4020 ' transmission times out, the program will halt, otherwise the program
4030 ' continues.
4040 ' *****
4050 '
4060 RECV$ = SPACE$(255)
4070 PRINT #1, "ENTER 3 $"
4080 LINE INPUT #2, RECV$
4090 RETURN
4100 '
5000 ' *****
5010 ' Define a subroutine for handling BASIC or interface errors. (All
5020 ' MBC-488 errors are handled in DOS by the resident driver.)
5030 ' *****
5040 '
5050 IF (ERR <> 68) AND (ERR <> 57) THEN PRINT: PRINT "BASIC ERROR # "; ERR; " IN LINE "; ERL
5060 INPUT #2, E$
5070 PRINT "$DV488 driver returned error number -", E$
5080 INPUT #2, E$
5090 PRINT E$
5100 INPUT #2, E$
5110 PRINT E$
5120 STOP
5130 ' *****

```

QuickStart Programs for Interpreter BASIC, and 500-Serial GPIB to RS-232 Converter

```

10 : *****
20 :
30 : ANALOG.SER - 576 QUICKSTART Analog I/O Program in Interpreter BASICA
40 : DATE: 8-16-90
50 :
60 : This program is designed to run with the 500-Serial RS232 to IEEE488
70 : converter box.
80 :
90 : The Auto Baud on the 500-Serial is set for 9600 baud.
100 :
110 : (c) Copyright 1990 - Keithley Data Acquisition and Control
120 :
130 : *****
140 :
150 : CLS : CLOSE : KEY OFF
160 :     ON ERROR GOTO 1880
170 :
180 : Declare variables
190 :
200 :     SPOLLBIT% = 0           ' Used as input to SPOLL check subroutine
210 :     BITSTAT% = 0          ' Used as output from SPOLL check subroutine
220 :
230 : Initialize 500-Serial interface.
240 :
250 :     OPEN "COM1:9600,N,8,2,CS,DS,CD" AS #1
260 :     T!=TIMER: WHILE TIMER-T!<.1: WEND
270 :
280 :     FOR I=1 TO 5
290 :         PRINT #1,CHR$(13);: T!=TIMER: WHILE TIMER-T!<.1: WEND
300 :     NEXT I
310 :
320 :     PRINT #1, "EC;0"       ' Turn the ECHO OFF
330 :     T!=TIMER: WHILE TIMER-T!<.1: WEND
340 :     A$=INPUT$(LOC(1),#1):A$=INPUT$(LOC(1),#1)
350 :
351 : Set 500-Serial bus terminator
352 :
353 :     PRINT #1,"TB;4"       ' Set CRLF
354 :
355 : System should respond with 576 revision, indicating successful
356 : communication with driver, GPIB interface, and 576.
357 :
358 :
359 :     OUTP$ = "Syst :IDN?;"
360 :     GOSUB 1660             'Send output string
361 :     SPOLLBIT%=4:GOSUB 1320
362 :
363 :     GOSUB 1770
364 :     LOCATE 1,1:PRINT RECV$
365 :
366 :     LOCATE 3, 1: PRINT "*** 500-Serial & 576 QuickStart Analog I/O Test ***"
367 :
368 : Reset 576
369 :
370 :     OUTP$ = "reset all;x;"
371 :     GOSUB 1660
372 :     T!=TIMER: WHILE TIMER-T!<2: WEND   ' Delay for RESET ALL. See
373 :                                         ' RESET command for details.
374 :     SPOLLBIT% = 7: GOSUB 1320
375 :
376 : Set data transmission format
377 :
378 :     OUTP$ = "syst :format ASCI;"
379 :     GOSUB 1660
380 :     SPOLLBIT% = 7: GOSUB 1320
381 :
382 :

```

```

610 ' Set up analog input (A/D) range and mode
620 '
630   OUTP$ = "chan 1, 0, :mode se;"
640   GOSUB 1660
650   SPOLLBIT% = 7: GOSUB 1320
660 '
670   OUTP$ = "chan 1, 0, :gain 1;"
680   GOSUB 1660
690   SPOLLBIT% = 7: GOSUB 1320
700 '
710   OUTP$ = "chan 1, 0, :range 10b;"
720   GOSUB 1660
730   SPOLLBIT% = 7: GOSUB 1320
740 '
750 ' Set up analog output (D/A) channel 0
760 '
770   OUTP$ = "chan 4, 0, :range 10b;"
780   GOSUB 1660
790   SPOLLBIT% = 7: GOSUB 1320
800 '
810 ' *****
820 '
830 ' Prompt for desired output voltage and write value to slot 4, channel 0
840 '
850   LOCATE 6, 1: PRINT SPACES$(78)
860   LOCATE 6, 1: INPUT "Output voltage, (-10 to +10, or 'Q' to quit) "; OV$
870   IF LEFT$(OV$,1) = "Q" OR LEFT$(OV$,1) = "q" THEN END
880 '
890   OUTP$ = "iwrite DCV 4, 0, " + OV$ + "; "
900   GOSUB 1660
910   SPOLLBIT% = 7: GOSUB 1320
920 '
930 ' Do readings of AMM channel 0 and return as volts
940 '
950   OUTP$ = "syst :clock ?;iread dcv, 1, 0;"
960   GOSUB 1660
970   SPOLLBIT% = 4: GOSUB 1320
980 '
990 '
1000 '   Input time value from GPIB
1010       GOSUB 1770
1020       ATIM$=RECV$
1030 '
1040 '   Input voltage value from GPIB
1050       GOSUB 1770
1060       IV$=RECV$
1070 '
1080 ' Do readings of AMM channel 0 and return as A/D counts
1090 '
1100   OUTP$ = "syst :clock ?; iread RAW, 1, 0;"
1110   GOSUB 1660
1120   SPOLLBIT% = 4: GOSUB 1320
1130 '
1140 '   input time/date value from GPIB
1150       GOSUB 1770
1160       CTIM$=RECV$
1170 '
1180 '   input data value from GPIB
1190       GOSUB 1770
1200       CTSS$=RECV$
1210 '
1220 ' Write information to the screen
1230 '
1240   LOCATE 10, 1: PRINT "Voltage written to analog output = "; OV$; " "
1250   LOCATE 12, 1: PRINT "Input Voltage and time = "; ATIM$, IV$; " "
1260   LOCATE 13, 1: PRINT "Raw counts and time = "; CTIM$, CTSS$; " "
1270 '
1280   GOTO 830
1290 '
1300 END

```

```

1310 '
1320 ' *****
1330 '
1340 ' Define a subroutine for checking the serial poll byte.
1350 ' The bit to be tested will be passed in as "spollbit%".
1360 ' The subroutine will return a value to the variable "bitstat%".
1370 ' The error bit will be tested automatically each time the subroutine
1380 ' is entered. If the error bit is set, program stops and prints message.
1390 '
1400 '
1410 '
1420 SB% = 2 ^ SPOLLBIT%
1430 SP% = 0: BITSTAT% = 0
1440 LOCATE 25, 1: PRINT "Testing SPOLL Bit"; SPOLLBIT%; ": Status = ";
1450 PRINT#1, "SP;03"
1460 INPUT#1, SP$
1470 SP%=VAL("&H"+SP$)
1480 '
1490 BITSTAT% = SP% AND SB%
1500 IF BITSTAT% = 0 THEN LOCATE 25, 31: PRINT BITSTAT%; : GOTO 1450
1510 IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25, 31: PRINT BITSTAT%;
1520 '
1530 ER% = SP% AND 32
1540 IF ER% = 0 THEN RETURN
1550 '
1560 '
1570 CMD$ = "Syst :ERR?;"
1580 PRINT #1, "OA;03;" + CMD$
1590 '
1600 GOSUB 1770
1610 '
1620 LOCATE 23,1: PRINT "ERROR! - "; RECV$; " - "; OUTP$; : LOCATE 20, 1: END
1630 RETURN
1640 '
1650 '
1660 ' *****
1670 ' Define a subroutine for sending command strings to the 576. If the
1680 ' transmission times out, the program will halt, otherwise the program
1690 ' continues.
1700 ' *****
1710 '
1720 OUTPUT$="OA;03;" + OUTP$
1730 PRINT #1, OUTPUT$
1740 RETURN
1750 '
1760 '
1770 ' *****
1780 ' Define a subroutine for receiving data from the 576. If the
1790 ' transmission times out, the program will halt, otherwise the program
1800 ' continues.
1810 ' *****
1820 '
1830 RECV$ = SPACES(255)
1840 PRINT#1, "EN;03"
1850 LINE INPUT#1, RECV$
1860 RETURN
1870 '
1880 RESUME NEXT

10 ' *****
20 '
30 ' DIGITAL.SER - 576 QUICKSTART Digital I/O Program in Interpreter BASICA
40 ' DATE: 8-16-90
50 '
60 ' This program is designed to run with the 500-Serial RS232 to IEEE488
70 ' converter box.
80 '
90 ' The Auto Baud on the 500-Serial is set for 9600 baud.
100 '

```

```

110 ' (c) Copyright 1990 - Keithley Data Acquisition and Control
120 '
130 ' *****
140 '
150 CLS : CLOSE : KEY OFF
160 ON ERROR GOTO 1640
170 '
180 ' Declare variables
190 '
200 SPOLLBIT% = 0 ' Used as input to SPOLL check subroutine
210 BITSTAT% = 0 ' Used as output from SPOLL check subroutine
220 '
230 ' Initialize 500-Serial interface.
240 '
250 OPEN "COM1:9600,N,8,2,CS,DS,CD" AS #1
260 T!=TIMER: WHILE TIMER-T!<.1: WEND
270 '
280 FOR I=1 TO 5
290 PRINT #1,CHR$(13);: T!=TIMER: WHILE TIMER-T!<.1: WEND
300 NEXT I
310 '
320 PRINT #1, "EC;0" ' Turn the ECHO OFF
330 T!=TIMER: WHILE TIMER-T!<.1: WEND
340 A$=INPUT$(LOC(1),#1):A$=INPUT$(LOC(1),#1)
350 '
360 ' Set proper 500-SERIAL BUS terminators
370 '
380 PRINT #1, "TB;4"
390 '
400 ' System should respond with 576 revision, indicating successful
410 ' communication with driver, GPIB interface, and 576.
420 '
430 OUTP$ = "Syst :IDN?;"
440 GOSUB 1420 'Send output string
450 SPOLLBIT%=4:GOSUB 1080
460 '
470 GOSUB 1530
480 LOCATE 1,1:PRINT RECV$
490 '
500 '
510 LOCATE 3, 1: PRINT "*** 500-Serial & 576 QuickStart Digital I/O Test ***"
520 '
530 ' Reset 576
540 '
550 OUTP$ = "reset all;x;"
560 GOSUB 1420
570 T!=TIMER:WHILE TIMER-T!<2:WEND 'Provide adequate delay for RESET
580 SPOLLBIT% = 7: GOSUB 1080
590 '
600 ' Set up digital Port A of input
610 '
620 OUTP$ = "chan 5, 0, :mode in;"
630 GOSUB 1420
640 SPOLLBIT% = 7: GOSUB 1080
650 '
660 ' Set up digital Port B for output
670 '
680 OUTP$ = "chan 5, 1, :mode out;"
690 GOSUB 1420
700 SPOLLBIT% = 7: GOSUB 1080
710 '
720 OUTP$ = "chan 1, 0, :gain 1;"
730 GOSUB 1420
740 SPOLLBIT% = 7: GOSUB 1080
750 '
760 ' *****
770 '
780 ' Loop to output data and make readings
790 '
800 LOCATE 5, 1: PRINT "Press any key to stop...";

```

```

810 /
820   FOR PASS%=0 TO 255
830 /     Write outval% to digital port B
840       OUTP$ = "iwrite raw 5, 1, " + STR$(PASS%) + "; "
850       GOSUB 1420
860       SPOLLBIT% = 7: GOSUB 1080
870 /
880 /     Read value at digital port A
890 /
900       OUTP$ = "iread raw, 5, 0;"
910       GOSUB 1420
920       SPOLLBIT% = 4: GOSUB 1080
930 /
940 /     Input voltage value from GPIB
950       GOSUB 1530
960       A$=RECV$
970 /
980 /     Print data to screen
990 /
1000    LOCATE 10, 1
1010    PRINT "Written to 576 Digital Port B: ";PASS%;"   "
1020    PRINT " Read from 576 Digital Port A: ";VAL(A$);"   "
1030 /
1040    I$=INKEY$:IF I$="" THEN NEXT PASS%
1050 /
1060 END
1070 /
1080 / *****
1090 /
1100 / Define a subroutine for checking the serial poll byte.
1110 / The bit to be tested will be passed in as "spollbit%".
1120 / The subroutine will return a value to the variable "bitstat%".
1130 / The error bit will be tested automatically each time the subroutine
1140 / is entered. If the error bit is set, program stops and prints message.
1150 /
1160 /
1170 /
1180    SB% = 2 ^ SPOLLBIT%
1190    SP% = 0: BITSTAT% = 0
1200    LOCATE 25, 1: PRINT "Testing SPOLL Bit"; SPOLLBIT%; ": Status = ";
1210    PRINT#1, "SP;03"
1220    INPUT#1, SP$
1230    SP%=VAL("&H"+SP$)
1240 /
1250    BITSTAT% = SP% AND SB%
1260    IF BITSTAT% = 0 THEN LOCATE 25, 31: PRINT BITSTAT%; : GOTO 1210
1270    IF BITSTAT% <> 0 THEN BITSTAT% = 1: LOCATE 25, 31: PRINT BITSTAT%;
1280 /
1290    ER% = SP% AND 32
1300    IF ER% = 0 THEN RETURN
1310 /
1320 /
1330    CMD$ = "Syst :ERR?;"
1340    PRINT #1, "OA;03;" + CMD$
1350 /
1360    GOSUB 1530
1370 /
1380    LOCATE 23,1: PRINT "ERROR! - "; RECV$; " - "; OUTP$; : LOCATE 20, 1: END
1390 RETURN
1400 /
1410 /
1420 / *****
1430 / Define a subroutine for sending command strings to the 576. If the
1440 / transmission times out, the program will halt, otherwise the program
1450 / continues.
1460 / *****
1470 /
1480    OUTPUT$="OA;03;" + OUTP$
1490    PRINT #1, OUTPUT$
1500    RETURN

```



```
1510 '  
1520 '  
1530 ' *****  
1540 ' Define a subroutine for receiving data from the 576. If the  
1550 ' transmission times out, the program will halt, otherwise the program  
1560 ' continues.  
1570 ' *****  
1580 '  
1590     RECV$ = SPACE$(255)  
1600     PRINT#1, "EN;03"  
1610     LINE INPUT#1, RECV$  
1620     RETURN  
1630 '  
1640 RESUME NEXT
```

QuickStart Programs for Microsoft C or Quick C, IOtech Driver488 Software Driver, and Compatible GPIB Interface

```

/*****
** ANALOG.C - 576 QUICKSTART Analog I/O Program For Microsoft C or **
** Microsoft QuickC. **
** DATE: 5-20-91 **
** **
** This program is designed to run with the IOtech Driver488 GPIB **
** driver software and a compatible GPIB interface card on an **
** IBM PC/XT/AT or compatible. It assumes an address of "03" for the **
** 576. **
** **
** (c) Copyright 1991 - Keithley Data Acquisition and Control **
** **
*****/

#define BUFFER_FULL 3.0 /* Define values for serial poll bits */
#define DATA_READY 4.0
#define ERROR 5.0
#define SRQ 6.0
#define IDLE 7.0

#include <ieeeeios.h> /* Renamed for Microsoft C */
#include <stdio.h>
#include <graph.h>
#include <conio.h>
#include <time.h>
#include <math.h>

time_t newtime,oldtime;

main()
{
    char response[256], rtc1[22], rtc2[22], output_volts[10],
        input_volts[21], raw_counts[12];

    /* Establish communications with Personal488 */
    if ( ieeeiinit()==-1 )
    {
        printf("Cannot initialize IEEE system.\n");
        exit(1);
    }

    _clearscreen(_GCLEARSCREEN);

    ieeewt("TIMEOUT 2\n");
    ieeewt("CLEAR\n");

    /* Read the Driver488 revision number */
    ieeewt("HELLO\n");
    ieeerd(response);
    printf("%s\n",response);

    /* Do read of Keithley ID */
    ieeewt("Output 03; Syst :IDN?;\n");
    spoll(DATA_READY);
    ieeewt("Enter 03\n");
    ieeerd(response);
    _settextposition(2,1);
    printf("Keithley ID String: %s.\n",response);

    /* Reset 576 */
    ieeewt("Output 03; Reset All; X;\n");

    /* Two second delay */
    time(&oldtime);
    while ( (time(&newtime) - oldtime) < 2);
    printf("Reset Complete...\n");
}

```

```

/* Send various system and channel commands */
ieewt("Output 03; Syst :Format ASCII;\n");
spoll(IDLE);

ieewt("Output 03; Chan 1,0 :Mode SE;\n");
spoll(IDLE);

ieewt("Output 03; Chan 1,0 :Gain 1;\n");
spoll(IDLE);

ieewt("Output 03; Chan 1,0 :Range 10B;\n");
spoll(IDLE);

ieewt("Output 03; Chan 4,0 :Range 10B;\n");
spoll(IDLE);

while (output_volts[0] != 'q' || output_volts[0] != 'Q')
{
    _settextposition(6,1);
    printf("Output voltage (-10 and +10, or 'Q' to quit)           ");

    _settextposition(6,46);
    gets(output_volts);
    if ( output_volts[0] == 'q' || output_volts[0] == 'Q' )
    {
        _settextposition(20,1);
        exit(1);
    }

    ieeprtf("Output 03; IWrite DCV,4,0, %s;\n",output_volts);
    ieewt("Output 03; Syst :Clock ?; IRead DCV,1,0;\n");
    ieewt("Output 03; Syst :Clock ?; IRead RAW,1,0;\n");

    spoll(DATA_READY);
    ieewt("Enter 03\n");
    ieeerd(&rtc1);

    spoll(DATA_READY);
    ieewt("Enter 03\n");
    ieeerd(&input_volts);

    spoll(DATA_READY);
    ieewt("Enter 03\n");
    ieeerd(&rtc2);

    spoll(DATA_READY);
    ieewt("Enter 03\n");
    ieeerd(&raw_counts);

    _settextposition(10,1);
    printf("Voltage written to analog output = %3.7f\n",
atof(output_volts));
    printf("Time and Input voltage = %s , %s\n", rtc1, input_volts);
    printf("Time and Raw Counts   = %s , %s\n", rtc2, raw_counts);
}
}

```

```

/*****

```

Define a subroutine for checking the serial poll byte. The bit to be tested will be passed in as a pre-defined value. The subroutine will print the result of the serial poll. The error bit will be tested automatically each time the subroutine is entered. If the error bit is set, the program stops and prints the error message.

SPOLL BYE / ERROR BIT TEST SUBROUTINE

*****/

```

spoll(testbit)
double testbit;

{
    int sb;
    int bitstat = 0;
    char response[256];
    unsigned int sp, es;
    sb = pow(2.0,testbit);

    _settextposition(24,1);
    printf("Testing SPOLL bit %.0f : Status = ",testbit);
    while ( bitstat == 0 )
    {
        ieeevt("Spoll 03\n");
        ieeeconf("%d",&sp);
        bitstat = sp & sb;
        if ( bitstat == 0 ) {
            _settextposition(24,32);
            printf("%d", bitstat);

            /* Error handling routine */
            es = sp & 32;
            if ( es != 0 ) {
                ieeevt("CLEAR\n");
                ieeevt("Output 03; Syst :ERR ?;\n");
                ieeevt("Enter 03\n");
                ieeeord(response);
                _settextposition(25,1);
                printf("ERROR! - %s - \n",response);
                _settextposition(20,1);
                exit(1); }
            }

        if ( bitstat != 0 ) {
            bitstat = 1;
            _settextposition(24,32);
            printf("%d",bitstat);
        }
    }
}

```

```

/*****
** DIGITAL.C - 576 QUICKSTART Digital I/O Program For Microsoft C or **
** Microsoft QuickC. **
** DATE: 5-20-91 **
** This program is designed to run with the IOtech Driver488 GPIB **
** driver software and a compatible GPIB interface card on an **
** IBM PC/XT/AT or compatible. It assumes an address of "03" for the **
** 576. **
** (c) Copyright 1991 - Keithley Data Acquisition and Control **
** **
*****/

```

```

#define BUFFER_FULL 3.0 /* Define values for serial poll bits */
#define DATA_READY 4.0
#define ERROR 5.0
#define SRQ 6.0
#define IDLE 7.0

```

```

#include <ieeeios.h>          /* Renamed for Microsoft C */
#include <stdio.h>
#include <graph.h>
#include <conio.h>
#include <time.h>
#include <math.h>

time_t newtime,oldtime;

main()
{
    char          response[256], digin[5];
    unsigned int  pass;

    /* Establish communications with Personal488 */
    if ( ieeeiinit()==-1 )
        {
            printf("Cannot initialize IEEE system.\n");
            exit(1);
        }

    _clearscreen(_GCLEARSCREEN);

    ieeewt("TIMEOUT 2\n");
    ieeewt("CLEAR\n");

    /* Read the Driver488 revision number */
    ieeewt("HELLO\n");
    ieeerd(response);
    printf("%s\n",response);

    /* Do read of Keithley ID */
    ieeewt("Output 03; Syst :IDN?;\n");
    spoll(DATA_READY);
    ieeewt("Enter 03\n");
    ieeerd(response);
    _settextposition(2,1);
    printf("Keithley ID String: %s.\n",response);

    /* Reset 576 */
    ieeewt("Output 03; Reset All; X;\n");

    /* Two second delay */
    time(&oldtime);
    while ( (time(&newtime) - oldtime) < 2);
    printf("Reset Complete...\n");

    /* Send various system and channel commands */
    ieeewt("Output 03; Chan 5,0 :Mode In;\n");
    spoll(IDLE);

    ieeewt("Output 03; Chan 5,1 :Mode Out;\n");
    spoll(IDLE);

    /* Loop to output data and make readings */
    _settextposition(6,1);

    for ( pass=0; pass<256; pass++ )
        {
            ieeeptrf("Output 03; IWrite RAW,5,1,%u;\n",pass);
            spoll(IDLE);
            ieeewt("Output 03; IRead RAW,5,0;\n");
            spoll(DATA_READY);
            ieeewt("Enter 03\n");
            ieeerd(&digin);

            _settextposition(10,1);
            printf("Written to 576 Digital Port B: %u\n", pass);
            printf("Read from 576 Digital Port A: %s\n", digin);
        }
}

```

}

/******

Define a subroutine for checking the serial poll byte. The bit to be tested will be passed in as a pre-defined value. The subroutine will print the result of the serial poll. The error bit will be tested automatically each time the subroutine is entered. If the error bit is set, the program stops and prints the error message.

SPOLL BYE / ERROR BIT TEST SUBROUTINE

*****/

```
spoll(testbit)
double    testbit;

{
    int      sb;
    int      bitstat = 0;
    char     response[256];
    unsigned int sp, es;
    sb = pow(2.0,testbit);

    _settextposition(24,1);
    printf("Testing SPOLL bit %.0f : Status = ",testbit);
    while ( bitstat == 0 )
    {
        ieeevt("Spoll 03\n");
        ieeeescnf("%d",&sp);
        bitstat = sp & sb;
        if ( bitstat == 0 ) {
            _settextposition(24,32);
            printf("%d", bitstat);

            /* Error handling routine */
            es = sp & 32;
            if ( es != 0 ) {
                ieeevt("CLEAR\n");
                ieeevt("Output 03; Syst :ERR ?;\n");
                ieeevt("Enter 03\n");
                ieeeerd(response);
                _settextposition(25,1);
                printf("ERROR! - %s - \n",response);
                _settextposition(20,1);
                exit(1); }
            }

        if ( bitstat != 0 ) {
            bitstat = 1;
            _settextposition(24,32);
            printf("%d",bitstat);
        }
    }
}
```

APPENDIX D

IEEE-488 Bus Overview

INTRODUCTION

Basically, the IEEE-488 bus is simply a communication system between two or more electronic devices. A device can be either an instrument or a computer. When a computer is used on the bus, it serves to supervise the communication exchange between all the devices and is known as the controller. Supervision by the controller consists of determining which device will talk and which device will listen. As a talker, a device will output information and as a listener, a device will receive information. To simplify the task of keeping track of the devices, a unique address number is assigned to each one.

On the bus, only one device can talk at a time and is addressed to talk by the controller. The device that is talking is known as the active talker. The devices that need to listen to the talker are addressed to listen by the controller. Each listener is then referred to as an active listener. Devices that do not need to listen are instructed to unlisten. The reason for the unlisten instruction is to optimize the speed of bus information transfer since the task of listening takes up bus time.

Through the use of control lines, a handshake sequence takes place in the transfer process of information from a talker to a listener. This handshake sequence helps ensure the credibility of the information transfer. The basic handshake sequence between an active controller (talker) and a listener is as follows:

1. The listener indicates that it is ready to listen.
2. The talker places the byte of data on the bus and indicates that the data is available to the listener.
3. The listener, aware that the data is available, accepts the data and then indicates that the data has been accepted.
4. The talker, aware that the data has been accepted, stops sending data and indicates that data is not being sent.
5. The listener, aware that there is no data on the bus, indicates that it is ready for the next byte of data.

BUS DESCRIPTION

The IEEE-488 bus, which is also frequently referred to as the GPIB (General Purpose Interface Bus), was designed as a parallel transfer medium to optimize data transfer without using an excessive number of bus lines. In keeping with this goal, the bus has only eight data lines that are used for both data and with most commands. Five bus management lines and three handshake lines round out the complement of bus signal lines.

A typical set up for controlled operation is shown in Figure B-1. Generally, a system will contain one controller and a number of other instruments to which the commands are given. Device operation is categorized into three operators: controller, talker and listener. The controller does what its name implies; it controls the instruments on the bus. The talker sends data while a listener receives data. Depending on the type of instrument, any particular device can be a talker only, a listener only or both a talker and listener.

There are two categories of controllers: system controller, and basic controller. Both are able to control other instruments, but only the system controller has the absolute authority in the system. In a system with more than one controller, only one controller may be active at any given time. Certain protocol is used to pass control from one controller to another.

The IEEE-488 bus is limited to 15 devices, including the controller. Thus, any number of talkers and listeners up to that limit may be present on the bus at one time. Although several devices may be commanded to listen simultaneously, the bus can have only one active talker, or communications would be scrambled.

A device is placed in the talk or listen state by sending an appropriate talk or listen command. These talk and listen commands are derived from an instrument's primary address. The primary address may have any value between 0 and 31, and is generally set by rear panel DIP switches or programmed in from the front panel of the instrument. The actual listen address value sent out over the bus is ob-

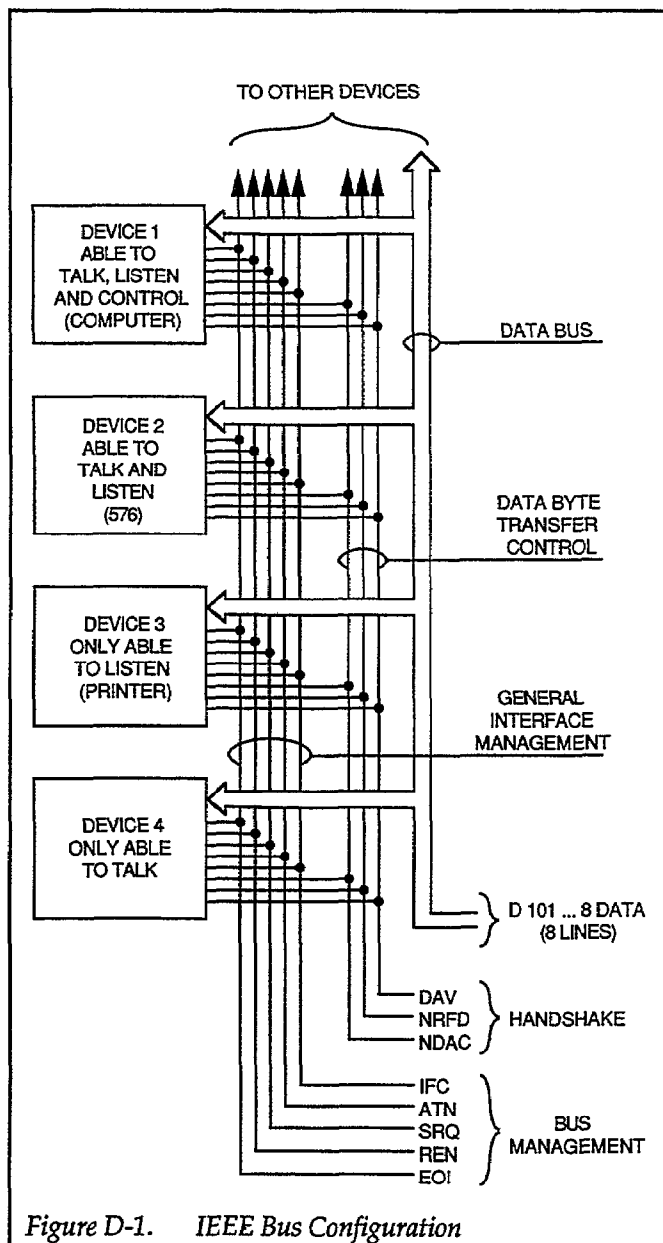


Figure D-1. IEEE Bus Configuration

tained by ORing the primary address with \$20. For example, if the primary address is 8 (\$08), the actual listen address is \$28 (\$28 = \$08 + \$20). In a similar manner, the talk address is obtained by ORing the primary address with \$40. With the present example, the talk address derived from a primary address of 8 decimal would be \$48 (\$48 = \$08 + \$40).

The IEEE-488 standards also include another addressing mode called secondary addressing. Secondary addresses lie in the range of \$60-\$7F. Note, however, that many devices, including the 576, do not use secondary addressing.

Once a device is addressed to talk or listen, the appropriate bus transactions take place. For example: if the instrument is addressed to talk, it places its data string on the bus one byte at a time. The controller reads the information and the appropriate software can be used to direct the information to the desired location.

BUS LINES

The signal lines on the IEEE-488 bus are grouped into three different categories: data lines, management lines and handshake lines. The data lines handle bus data and commands, while the management and handshake lines ensure that proper data transfer and operation takes place. Each bus line is active low, with approximately zero volts representing a logic 1 (true). The following paragraphs describe the operation of these lines.

Data Lines

The IEEE-488 bus uses eight data lines that transfer data one byte at a time. DIO1 (Data Input/Output) through DIO8 (Data Input/Output) are the eight data lines used to transmit both data and multiline commands and are bidirectional. The data lines operate with low true logic.

Bus Management Lines

The five bus management lines help to ensure proper interface control and management. These lines are used to send the uniline commands.

ATN (Attention) — The ATN line is one of the more important management lines in that the state of this line determines how information on the data bus is to be interpreted.

IFC (Interface Clear) — As the name implies, the IFC line controls clearing of instruments from the bus.

REN (Remote Enable) — The REN line is used to place the instrument on the bus in the remote mode.

EOI (End or Identify) — The EOI is usually used to mark the end of a multi-byte data transfer sequence.

SRQ (Service Request) — This line is used by device when they require service from the controller.

Handshake Lines

The bus handshake lines that operate in an interlocked sequence. This method ensures reliable data transmission regardless of the transfer rate. Generally, data transfer will occur at a rate determined by the slowest active device on the bus.

One of the three handshake lines is controlled by the source (the talker sending information), while the remaining two lines are controlled by accepting devices (the listener or listeners receiving the information). The three handshake lines are:

DAV (DATA VALID) — The source controls the state of the DAV line to indicate to any listening devices whether or not data bus information is valid.

NRFD (Not Ready For Data) — The acceptor controls the state of NRFD. It is used to signal to the transmitting device to hold off the byte transfer sequence until the accepting device is ready.

NDCA (Not Data Accepted) — NDCA is also controlled by the accepting device. The state of NDCA tells the source whether or not the device has accepted the data byte.

The complete handshake sequence for one data byte is shown in Figure D-2. Once data is placed on the data lines, the source checks to see that NRFD is high, indicating that all active devices are ready. At the same time, NDAC should be low from the previous byte transfer. If these conditions are not met, the source must wait until NDAC and NRFD have the correct status. If the source is a controller, NRFD and NDAC must be stable for at least 100nsec after ATN is set true. Because of the possibility of a bus hang up, many controllers have time-out routines that display messages in case the transfer sequence stops for any reason.

Once all NDAC and NRFD are properly set, the source sets DAV low, indicating to accepting devices that the byte on the data lines is now valid. NRFD will then go low, and NDAC will go high once all devices have accepted the data. Each device will release NDAC at its own rate, but NDAC will not be released to go high until all devices have accepted the data byte.

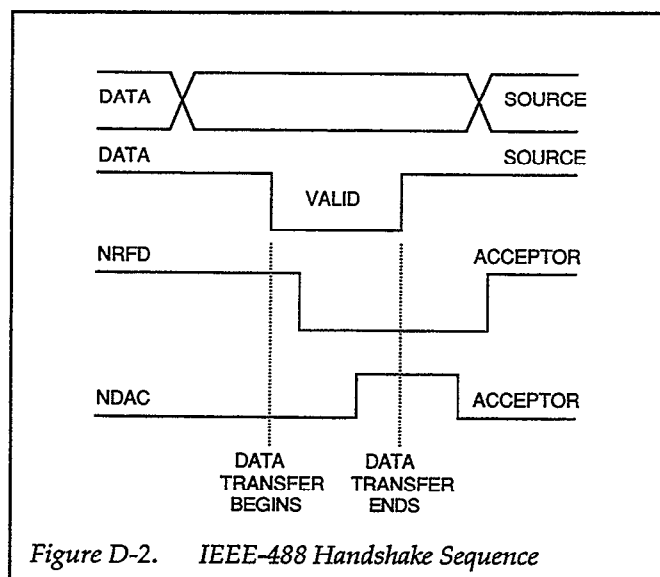


Figure D-2. IEEE-488 Handshake Sequence

The sequence just described is used to transfer both data, talk and listen addresses, as well as multiline commands. The state of the ATN line determines whether the data bus contains data, addresses or commands as described in the following paragraph.

BUS COMMANDS

The instrument may be given a number of special bus commands through the IEEE-488 interface. This section briefly describes the purpose of the bus commands which are grouped into the following three categories.

6. Uniline Commands - Sent by setting the associated bus lines true. For example, to assert REN (Remote Enable), the REN line would be set low (true).
7. Multiline Commands - General bus commands which are sent over the data lines with the ATN line true (low).
8. Device-dependent Commands - Special commands whose meanings depend on device configurations; sent with ATN high (false).

These bus commands and their general purpose are summarized in Table D-1.

Uniline Commands

ATN, IFC and REN are asserted only by the controller. SRQ is asserted by an external device. EOI may be asserted either by the controller or other devices depending

on the direction of data transfer. The following is a description of each command. Each command is sent by setting the corresponding bus line true.

REN (Remote Enable) — REN is sent to set up instruments on the bus for remote operation. When REN is true, devices will be removed from the local mode. Depending on device configuration, all front panel controls except the LOCAL button (if the device is so equipped) may be locked out when REN is true. Generally, REN should be sent before attempting to program instruments over the bus.

EOI (End or Identify) — EOI is used to positively identify the last byte in a multi-byte transfer sequence, thus allowing data words of various lengths to be transmitted easily.

IFC (Interface Clear) — IFC is used to clear the interface and return all devices to the talker and listener idle states.

ATN (Attention) — The controller sends ATN while transmitting addresses or multiline commands.

SRQ (Service Request) — SRQ is asserted by a device when it requires service from a controller.

Universal Multiline Commands

Universal commands are those multiline commands that require no addressing. All devices equipped to implement such commands will do so simultaneously when the commands is transmitted. As with all multiline commands, these commands are transmitted with ATN true.

LLO (Local Lockout) — LLO is sent to the instrument to lock out their front panel controls.

DCL (Device Clear) — DCL is used to return instruments to some default state. Usually, instruments return to their power-up conditions.

SPE (Serial Poll Enable) — SPE is the first step in the serial polling sequence which is used to determine which device has requested service.

SPD (Serial Poll Disable) — SPD is used by the controller to remove all devices on the bus from the serial poll mode and is generally the last command in the serial polling sequence.

Table D-1. IEEE-488 Bus Command Summary

Command Type	Commands	ATN Line	State Of Comments
Uniline	REN (Remote Enable)	X	Sets up devices for remote operation.
	EOI	X	Marks end of transmission.
	IFC (Interface Clear)	X	Clears interface.
	ATN (Attention)	Low	Defines data bus contents.
	SRQ	X	Controlled by external Device.
Universal	Multiline		
	LLO (Local Lockout)	Low	Locks out local operation.
	DCL (Device Clear)	Low	Returns device to default conditions.
Addressed	SPE (Serial Enable)	Low	Enables serial polling.
	SPD (Serial Poll Disable)	Low	Disables serial polling.
	SDC (Selective Device Clear)	Low	Returns unit to default conditions.
	GTL (Go To Local) Uniline	Low	Returns device to local.
Unaddressed	UNL (Unlisten)	Low	Removes all listeners from bus.
	UNT (Untalk)	Low	Removes any talkers from bus.
Device-dependent		High	Programs Model 576 for various modes.

Addressed Multiline Commands

Addressed commands are multiline commands that must be preceded by the device listen address before that instrument will respond to the command in question. Note that only the addressed device will respond to these commands. Both the commands and the address preceding it are sent with ATN true.

SDC (Selective Device Clear) — The SDC command performs essentially the same function as the DCL command except that only the addressed device responds. Generally, instruments return to their power-up default conditions when responding to the SDC command.

GTL (Go To Local) — The GTL command is used to remove instruments from the remote mode. With some instruments, GTL also unlocks front panel controls if they were previously locked out with the LLO commands.

GET (Group Execute Trigger) — The GET command is used to trigger devices to perform a specific action that depends on device configuration (for example, take a reading). Although GET is an addressed command, many devices respond to GET without addressing.

Address Commands

Addressed commands include two primary command groups and a secondary address group. ATN is true when these commands are asserted. The commands include:

LAG (Listen Address Group) — These listen commands are derived from an instrument's primary address and are used to address devices to listen. The actual command byte is obtained by ORing the primary address with \$20.

TAG (Talk Address Group) — The talk commands are derived from the primary address by ORing the address with \$40. Talk commands are used to address devices to talk.

SCG (Secondary Command Group) — Commands in this group provide additional addressing capabilities. Many devices (including the 576) do not use these commands.

Unaddress Commands

The two unaddress commands are used by the controller to remove any talkers or listeners from the bus. ATN is true when these commands are asserted.

UNL (Unlisten) — Listeners are placed in the listener idle state by the UNL command.

UNT (Untalk) — Any previously commanded talkers will be placed in the talker idle state by the UNT command.

Device-dependent Commands

The purpose of the device-dependent commands will depend on the configuration of the instrument. Generally, these commands are sent as one or more ASCII characters that tell the device to perform a specific function.

The IEEE-488 bus actually treats these commands as data in that ATN is false when the commands are transmitted.

Command Codes

Command codes for the various commands that use the data lines are summarized in Figure D-3. Hexadecimal and the decimal values for the various commands are listed in Table D-2.

Table D-2. Hexadecimal and Decimal Command Codes

Command	Hex Value	Decimal Value
GTL	01	1
SDC	04	4
LLO	11	17
DCL	14	20
SPE	18	24
SPD	19	25
LAG	20-3F	32-63
TAG	40-5F	64-95
SGG	60-7F	96-127
UNL	3F	63
UNT	5F	95

Figure D-3. Command Codes

D ₇ D ₆ D ₅ D ₄						X 0 0 0	COMMAND	X 0 0 1	COMMAND	X 0 1 0	PRIMARY ADDRESS	X 0 1 1	PRIMARY ADDRESS	X 1 0 0	PRIMARY ADDRESS	X 1 0 1	PRIMARY ADDRESS	X 1 1 0		X 1 1 1	
	Bits	D ₃ ↓	D ₂ ↓	D ₁ ↓	D ₀ ↓	COLUMN ROW	0(A)	0(B)	1(A)	1(B)	2(A)	2(B)	3(A)	3(B)	4(A)	4(B)	5(A)	5(B)	6(A)	6(B)	7(A)
0	0	0	0	0	0	NUL		DLE		SP	0	0	16	@	0	P	16				p
0	0	0	0	1	1	SOH	GTL	DC1	LLO	!	1	1	17	A	1	Q	17	a			q
0	0	0	1	0	2	STX		DC2		"	2	2	18	B	2	R	18	b			r
0	0	0	1	1	3	ETX		DC3		#	3	3	19	C	3	S	19	c			s
0	1	0	0	0	4	EOT	SDC	DC4	DCL	\$	4	4	20	D	4	T	20	d			t
0	1	0	0	1	5	ENQ	PPC*	NAK	PPU*	%	5	5	21	E	5	U	21	e			u
0	1	1	0	0	6	ACK		SYN		&	6	6	22	F	6	V	22	f			v
0	1	1	1	1	7	BEL		ETB		'	7	7	23	G	7	W	23	g			w
1	0	0	0	0	8	BS	GET	CAN	SPE	(8	8	24	H	8	X	24	h			x
1	0	0	0	1	9	HT	TCT*	EM	SPD)	9	9	25	I	9	Y	25	i			y
1	0	1	0	0	10	LF		SUB		•	10	:	26	J	10	Z	26	j			z
1	0	1	1	1	11	VT		ESC		+	11	;	27	K	11	[27	k			{
1	1	0	0	1	12	FF		FS		,	12	<	28	L	12	\	28	l			:
1	1	0	1	0	13	CR		GS		-	13	=	29	M	13]	29	m			}
1	1	1	1	0	14	SO		RS		.	14	>	30	N	14	^	30	n			≡
1	1	1	1	1	15	SI		US		/	15	?	UNL	O	15	_	UNT	o			DEL

ADDRESSSED COMMAND GROUP (ACG)					UNIVERSAL COMMAND GROUP (UCG)					LISTEN ADDRESS GROUP (LAG)					TALK ADDRESS GROUP (TAG)				
										PRIMARY COMMAND GROUP (PCG)					SECONDARY COMMAND GROUP (SDC)				

*PPC (PARALLEL POLL CONFIGURE), PPU (PARALLEL POLL UNCONFIGURE), AND TCT (TAKE CONTROL) NOT IMPLEMENTED BY MODEL 576.

NOTE: D₀ = DIO1 . . . D₇ = DIO8
X = DON'T CARE

Typical Command Sequences

For the various multiline commands, a specific bus sequence must take place to properly send the command. In particular, the correct listen address must be sent to the instrument before it will respond to addressed commands. Table D-3 lists a typical bus sequence for sending the addressed multiline commands. In this instance, the SDC command is being sent to the instrument. UNL is generally sent as part of the sequence to ensure that no other active listeners are present. Note that ATN is true for both the listen command and the SDC command byte itself.

Table D-4 gives a typical device-dependent command sequence. In this instance, ATN is true while the instrument is being addressed, but it is set high while sending the device-dependent command string.

IEEE Command Groups

Command groups supported by the 576 are listed in Table D-5. Device-dependent commands are not included in this list.

Table D-3. Typical Addressed Command Sequence

Step	Command	ATN State	Data Bus		
			ASCII	Hex	Decimal
1	UNL	Set low	?	3F	63
2	LAG*	Stays low	(28	40
3	SDC	Stays low	EOT	04	4
4		Returns high			

*Assumes primary address = 8.

Table D-4. Typical Device-dependent Command Sequence

Step	Command	ATN State	Data Bus		
			ASCII	Hex	Decimal
1	UNL	Set low	?	3F	63
2	LAG*	Stays low	(28	40
3	Data	Set high	F	52	82
4	Data	Stays high	0	30	48
5	Data	Stays high	X	58	88

*Assumes primary address = 8.

Table D-5. IEEE Command Group

<p>HANDSHAKE COMMAND GROUP NDAC = NOT DATA ACCEPTED NRFD = NOT READY FOR DATA DAV = DATA VALID</p> <p>UNIVERSAL COMMAND GROUP ATN = ATTENTION DCL = DEVICE CLEAR IFC = INTERFACE CLEAR REN = REMOTEENABLE SPD = SERIAL POLL DISABLE SPE = SERIAL POLL ENABLE</p> <p>ADDRESS COMMAND GROUP LISTEN: LAG = LISTEN ADDRESS GROUP MLA = MY LISTEN ADDRESS UNL = UNLISTEN TALK: TAG = TALK ADDRESS GROUP MTA = MY TALK ADDRESS UNT = UNTALK OTA = OTHER TALK ADDRESS</p> <p>ADDRESSED COMMAND GROUP ACG = ADDRESSED COMMAND GROUP GTL = GO TO LOCAL SDC = SELECTIVE CLEAR</p> <p>STATUS COMMAND GROUP RQS = REQUEST SERVICE SRQ = SERIAL POLL REQUEST STB = STATUS BYTE EOI = END</p>
--

APPENDIX E

Calibration of the Model 576 Using the SYST:CAL and ZCAL Commands

When used in the Model 576, calibration facilities for the AMM1A and AMM2 modules include hardware calibration potentiometers, the ZCAL command, and the SYST:CAL command. The potentiometers are used in conjunction with the hardware calibration instructions contained in the AMM manuals. The purpose of the commands is as follows:

ZCAL — ZCAL determines a calibration constant reflecting the characteristics of the zener reference on the AMM1A or AMM2. This constant is then used by the SYST:CAL command to create gain and offset correction factors for the AMM module's global amplifier and A/D section. ZCAL generally need not be performed unless:

- a. the existing calibration is in question,
- b. the AMM module has received extensive repair, or
- c. the zener reference has been disturbed or replaced. Using ZCAL requires additional calibration instrumentation.

SYST:CAL — The SYST:CAL command is to store the calibration constant produced by ZCAL in the 576's battery-backed memory from which correction factors are generated and applied analog measurements. The correction factors affect the AMM global amp and A/D converter of the AMM1A or AMM2, and are applied automatically by the 576 firmware.

A calibration constant for each AMM1A or AMM2 module is printed on a sticker and attached to the module at the factory. It is a simple matter to issue this constant to the Model 576 using the procedure described in the SYST:CAL command documentation. ZCAL can be used to generate a new constant.

By using ZCAL and SYST:CAL, it is possible to achieve the published "improved" accuracy specifications for the AMM1A and AMM2. When an analog input module is used in the option slot of the 576, the overall accuracy of

the combination will essentially be the accuracy of the option module.

When and How Should You Calibrate?

The AMM module will generally be calibrated for three reasons:

1. The module has been repaired.
2. The module is suspected of having drifted out of calibration.
3. The module is on a periodic maintenance/calibration schedule.

In each case, perform a hardware calibration, followed by ZCAL (ZCAL automatically performs a SYST:CAL). If a suitable precision voltage calibrator is not available, and if the zener reference on the AMM module has not been disturbed, you can omit the ZCAL and use SYST:CAL, assuming that the current calibration constant is still valid. The long-term stability of the zener reference circuitry is quite high, so the ZCAL constant should not change appreciably under normal circumstances.

A module which returns readings which are incorrect by more than a few percent may be defective, rather than simply out of calibration. Calibration is not a substitute for repairing a faulty module.

How to Perform a Complete AMM Field Calibration

A complete hardware calibration of the AMM1A or AMM2 module consists of adjusting the potentiometers for the local amplifier, global amplifier, and A/D converter sections, followed by ZCAL to trim the calibration of the global amplifier and A/D converter sections. ZCAL will run SYST:CAL. The AMM module manual contains a complete listing of steps and equipment needed for these procedures.

APPENDIX E

Calibration of the Model 576 Using the SYST:CAL and ZCAL Commands

If you plan to adjust the calibration pots on the AMM module, you should do so before you use ZCAL. This will provide optimum adjustment of the AMM.

After you run ZCAL, do not readjust the AMM module calibration pots. The calibration factor produced by ZCAL applies to the module in the state of calibration prevailing when ZCAL was executed. If you change any adjustments on the board after ZCAL is executed, you will need to repeat ZCAL.

ZCAL does not affect the AMM local amplifier.

The following equipment is required to determine the calibration constant using ZCAL:

1. Voltage calibrator with an accuracy of 0.001% or better.
2. A DIP clip for a 28-pin 0.6 inch wide IC.
3. A Model 576 with host computer.
4. AMM1A or AMM2 module, revision C or later.
5. Component layout and schematic for the AMM module.

Procedure:

1. Install the AMM1A or AMM2 in slot 1 of the 576.
2. Remove any module from slot 3 to gain access to U8 on the AMM1A or AMM2.
3. Install the DIP clip on U8 of the AMM module.
4. Turn off output of the voltage calibrator.

5. Connect the voltage calibrator “-” output to U8 pin 19 via the DIP clip. Also connect the voltage calibrator “+” output to U8 pin 22 using the DIP clip.
6. Set switch 6 on the 576 address switch ON to allow the 576 to save the ZCAL information.
7. Turn on the power to the 576.
8. Set the voltage calibrator to 7.00000* volts, then turn it on.
9. Allow 15 minutes for the 576 and calibrator to warm up.
10. Issue the command ZCAL 7.0*; to the 576. The 576 will respond with the calibration information.

It is unnecessary to issue a SYST :CAL <val>; or SYST :CAL; command, or otherwise manually transfer ZCAL's calibration constant to SYST :CAL. These steps are handled automatically by ZCAL. However, ZCAL will only save the new zener calibration constant if switch 6 of the 576 address switch is set ON. Otherwise, the effects of ZCAL are temporary (i.e. they will be in effect until a power down or RESET ALL). You should record the calibration constant for future use with the SYST :CAL command.

Switch 6 should be turned OFF after the calibration is completed. This will protect against inadvertent changes to the calibration.

*If you use a voltage other than 7.000V in step 8, that voltage must be specified in step 10. The legal range is 6.00V to 8.00V.

Index

Symbols

:AMM, 5-45
:BUF?, 5-45
:CAL, 5-45
:CLOCK, 5-45
:EOI, 5-45, 5-51
:ERR?, 5-45
:FILTER, 5-16, 5-17
:FORM, 5-45
:function, 5-16
:GAIN, 5-16
:IDN?, 5-45
:MEM?, 5-45
:MODE, 5-16, 5-20
:OFFSET, 5-16, 5-22
:PEEK, 5-45
:POKE, 5-45
:RANGE, 5-16, 5-23
:SAVE, 5-45, 5-58
:SLOT, 5-45, 5-59
:SRQ, 5-45
:STAMP, 5-45, 5-61
:TERM, 5-45
:TRIG, 5-45, 5-63
:UNIT, 5-45
"?", 5-2
"acquisition time", 7-12
"CHAN", 4-2
"Channel", 4-3
"corrected", 5-48
"header" plug, 4-14
"LATCH", 6-8
"NO ERRORS", B-1
"No System Error", 5-52
"Slot", 4-3
"SRQ", 3-5
"SYST :CAL", 5
"TC", 5-40

"uncorrected", 5-48
"EMPTY", 5-59
(GET), 5-2, 5-73
(X), 5-73
+10 Volt Reference, 8-2
+5 Volt Supply, 8-2

Numbers

10.00V precision reference, 4-10,
7-10, 7-19
100kHz filter, 7-12
16 single-ended channels, 4-7
2kHz filter, 7-12
3B, 4-12, 7-28
3dB point, 4-14
500-Serial adapter, 6-13
500-series, 4-3
500GPIB, 4-1
576 Commands, 5-1
576 commands, 5-4
576 Programming Command Set, 5-4
576-1, 3-3
576-2, 3-3, 4-7
8 differential channels, 4-7

A

A/D Converter, 4-13, 7-12
A/D resolution, 4-7
A/D START, 7-12
AC Line adapter, 3-1
Accessories, 7-34
Acquisition time, 7-12
ADM1, 5-1
ADM2, 5-1

Address, 3-2
Address Commands, D-5
Advanced Topics, 4-12
AIM1, ADM1, ADM2, 4-1, 5-1
AIM2, 5-19
AIM3, 5-20
AIM3A, 5-19
AIM4, 5-19
AIM5, 5-19
AIM6, 5-19
AIM7, 5-19
AIM8, 5-17, 5-19
AIM9, 5-17, 5-19
AMM, 3-3, 4-1, 4-10, 4-12, 4-14, 5-66,
7-19
AMM1, 4-1, 5-1
AMM1A, 4-2, 4-7, 4-13, 5-19, 5-20,
5-48, 7-11, 8-2
AMM2, 4-2, 4-13, 5-19, 5-20, 8-2
AMM2 global amplifier, 5-48
AN OUT, 7-10
ANALOG.xxx, 2
Analog Input, 5-23, 7-11, 8-6
Analog Input – Slot 1, 4-7
Analog Output, 5-23, 7-19, 8-6
Analog Output – Slot 4, 4-10
Analog Output Commands, 7-20
Analog Output Range, 4-15
Analog output speed, 4-10
Analog pathways, 7-28
Analog Trigger, 7-15
AOM5, 4-10, 5-35
Appendix, 2-4
Applications, 2-4, 6-1
ASCII, 5-53
ASCII, 5-3, 5-53
ASCII no Prefix, 5-76
ASCII Transmission, 5-76
ASCII with Prefix, 5-76

ASCN, 5-53
Asystant GPIB, 6-1, 6-5
ATN, D-2, D-4
Attenuator, 4-14
Auto-acquire, 4-7
Auto-sequence, 4-10
Automatic Register Sequencing, 7-20
Automotive power adapter, 7-5

B

BASIC, 1
Battery, 8-6, 8-7
Battery Life Testing, 6-1
BINARY, 5-76
Binary Transfer, 5-77
Block diagram, 7-1
BNC, 4-3
BUFF, 5-2
BUFFER FULL, 5-79
Buffers, 5-2
Bus Commands, D-3
Bus Lines, D-2
BYTE, 5-7, 5-75
bracket, 3-3

C

CAUTION, 2-1
Calibration, 5, 4-7, 8-1
Calibration Constant, 5, 5-48
Calibration Equipment, 8-2
Calibration Maintenance and Troubleshooting, 2-4
Calibration sticker, 5-48
Capacitive effect, 4-14
Capacitive load, 7-19
CEC, 1, 1-1, C-1
CHAN, 4-2, 5-1
Channel 7, 4-12
CLOCK, 5-61
Clock Battery, 8-6, 8-7
CMD1A Select A/D Channel, 7-30
CMD1B Select Slot, 7-30
CMD1C Global Gain, 7-33
CMD1D A/D Start/Status, 7-33
CMD2A Trigger, 7-30
CMD2B Trigger, 7-31
CMD2C Trigger, 7-32
CMD3A Option Slot, 7-31
CMD3B Option Slot, 7-31
CMD3C Option Slot, 7-33
CMD4A Analog Output, 7-31
CMD4B Analog Output, 7-31
CMD5A Digital I/O, 7-32
CMD5B Digital I/O, 7-32
CMDA, 7-10
CMDA1, 7-12
CMDA2, 7-16
CMDA4, 7-20

CMDA5, 7-26
CMDB, 7-10
CMDB1, 7-12
CMDB2, 7-16
CMDB4, 7-20
CMDB5, 7-26
CMDC, 7-10
CMDC1, 7-12
CMDC2, 7-16
CMDD, 7-10
CMDD1, 7-12
CNFG, 5-58
CONFIGURATION, 5-1
Cold junction reference, 5-75
Command Codes, D-5
Command Locations, 7-30
Command Sequences, D-7
Common mode, 4-14
Completing Setup, 3-5
Condition expression, 5-31
Connections, 4-2
Controller, 1-1, 3-2
Conversion time, 7-12
Coupling, 5-66
CR, 5-62
CRLF, 5-62
Current Amplifier, 6-17
Cutoff frequencies, 7-12
CYCLES, 5-72

D

D/A, 4-15
D/A converters, 7-19
DACs, 6-1
DAISY, 7-10
DATA, 5-58
Data Backup Battery, 8-6, 8-7
DATA RDY, 5-79
DATA READY, 5-79
DAV, D-3
Data and Memory Management, 5-2
Data Configuration, 5-78
Data Formats, 5-3
Data Logging, 6-1
Data memory, 5-2, 5-55
Data Transfer Modes, 5-76
Data transfer formats, 5-11
Date, 5-50
Date field, 5-77
Date string, 5-50
Date/time stamps, 5-31
Day, 5-50
DCL, D-4
DCV, 5-64, 6-4
DEBUG, 5-79
DEGC, 5-77
DEGF, 5-77
Debug, 5-79
Default, 5-20
Default address, 3-2

Default Configuration, 4-2
Default data format, 5-3
Device Clear, A-1
Device Configuration, 6-5
Device Status (SPOLL), 5-79
Device-dependent Commands, D-5
DIGITAL, 5-59
DIGITAL.xxx, 2
DIO1, 5-20
DIO1A, 5-20
Diagnostic checks, 5-65
Diagnostics, 3-5
Differential current, 4-14
Differential input configuration, 4-2
Digital I/O, 7-23
Digital I/O – Slot 5, 4-11
Digital I/O Command, 7-24
Digital Input and Output, 7-23
Digital Input Terminals, 7-23
Digital Input/Output, 8-6
DO and LOOP, 5-29
DO...LOOP, 5-2
Driver488, C-1
Drivers, 3-2
DS-1216D Smart Watch, 5-76

E

END or IDENTIFY, 5-51, D-2, D-4
ENDSUB, 5-3
Engineering Units, 5-2, 5-3, 5-11, 5-64
Environmental, 3-2, 8-2
EOI, D-2, D-4
ERROR, 5-79
ERROR STRINGS, B-1
Error Conditions, B-1
Error messages, 5-52, B-1
EU, 5-11
EVENT, 5-66
EXEC, 5-63
Example Code, 6-4, 6-9, 6-12, 6-15, 6-16, 6-18
Expression Table, 5-32, 5-71
External Analog Input Command, 7-28
External channels, 7-28
External Input, 7-28
External Input – Slot 6, 4-12

F

Features, 1-1
FILTER, 5-46, 7-12, 7-15
Firmware revision, 5-54
Floating Sources, 4-14, 7-11
FOR...NEXT, 5-29
FREQ, 5-77
Frequencies, 4-10
Functional Description, 7-1
front panel, 3-1

G

Gain, 4-2, 4-12, 5-18, 7-12
Gain amplifier, 7-11
GET, 5-63
GLOBAL, 7-10, 7-11
GLOBAL GAIN, 4-12, 4-13, 7-12
GLOBAL IN, 4-7, 4-8
GLOBAL OUT, 4-8
GLOBAL STROBE, 7-20
Global, 5-18, 7-33
Global amplifier, 4-7
Global multiplexer, 4-12
GND, 4-7
GPIB, 5-1
GROUND, 4-13
Ground screw, 4-7
Ground wire assembly, 3-1

H

HALT, 5-2
Handshake, D-3
Hardware Configuration, 2-4, 4-1
Hardware Switch Configuration, 4-2
Hardware Trigger (HT), 5-76
Hardware Triggering, 6-8
Header, 5-76
Hours, 5-50
HZ, 5-36, 5-64, 6-4

I

Icon, 6-19
ID, 7-11
IDLE, 5-79
IEEE Command Groups, D-7
IEEE-488 Bus, D-1
IF...ELSE...ENDIF, 5-2
IF, ELSE, and ENDIF, 5-31
IFC, D-2, D-4
INPUT, 5-59
INTEL, 5-3, 5-53, 5-77
INTL, 5-53
Input Command Locations, 7-12
Input Filtering, 4-14
Input Ranges, 5-19
Input resistor, 4-14
Instrument amplifier, 4-12
Interactive Mode, 6-5
Interface Clear, A-1
Interfaces, 9-1
Internal Data Buffer Description, 5-75
Introduction, 1-1
IOtech, 1, 1-1
Isolated input, 4-10

J

J12-J15, 4-11
J2, 4-7
J201, 4-7
J203, 8-7
J307, 7-28
J4, J5 and J6, 4-13
J5, 4-14

L

LATCH, 5-66
Lap-tops, 6-12
Legal characters, 5-44
LF, 5-62
LFCR, 5-62
LLO, D-4
LONG, 5-7, 5-75
Load, 7-19
Local, 5-18
Local amplifier, 4-7
Local gain, 4-12
Looping, 5-2
Low-Current Measurements, 6-17
Low-noise measurement, 4-13
LVDTs, 4-10

M

MA, 5-34, 5-64, 6-4
Macintosh, 6-1, 6-18
Maintenance, 8-1
Maximum current output, 4-10
MEM1, 5-1
Measuring Currents, 4-13
Metrabyte, 1-1, C-1
MIN, 5-36, 5-68
Minutes, 5-50
MLHZ, 5-36
MOTO, 5-53
MOTOROLA, 5-3
Module Gains, 5-19
Module Library, 9-1
Module Manuals, 9-4
Month, 5-50
Mother Board, 7-6
Mother Board Checks, 8-4
Mother Board Signal Lines, 7-10
Motorola, 5-53, 5-77
MSEC, 5-36, 5-68
Multi-Channel, 6-1, 6-8
Multiline, D-4, D-5

N

National, 1, 1-1, C-1
NDCA, 5-77, D-3
NDCV, 5-77
Negative voltage, 5-35

NOTE, 2-1
Non-Self-ID Modules, 7-34
NRAW, 5-77
NRFD, D-3

O

Offset, 5-10
Ohm's law, 4-14
ON/OFF, 3-1
ONCE, 5-61, 5-67
ONCE, CLOCK, 6-11
ONCE, TIME, 6-11
ONINT, 5-3
Open thermocouple sense, 2-3
Option Modules, 2-4, 9-1
Option Slot, 7-18
Option Slot - Slot 3, 4-10
Option Slot Commands, 7-18
Option Slot Pinout, 7-9
Optional module, 3-3
OUTPUT, 5-59
Output capacitance, 4-10
Output current, 4-10, 7-19
Output Limitations, 7-19
Output ranges, 7-19

P

PC, XT, AT, PS/2 Personal Computers, 1-1
PCM1, DIO1A, 5-7
PCM2, 5-1
PCM3, 4-11, 7-26
Physical slot, 4-3
PIM1, 5-7, 5-20, 5-23
PIM2, 5-7, 5-20
POWER, 3-1
Portable, 6-1
Portable Acquisition, 6-12
Ports, 4-11, 7-23
Power, 7-5
Power Consumption, 9-3
Power Control - Slot 5, 4-11
Power control, 4-11, 7-26
Power Indicator, 7-6
Power Supply, 7-5
Power Supply Checks, 8-4
Power Supply Voltage, 8-4
Power source, 3-3
Power-on reset, 2-3
PROG, 5-58
PROTO, 5-1, 5-59
Precautions, 2-2
Prefix Field, 5-77
Primary address, 3-2
Process Monitoring, 6-1, 6-9
Product Support, 2-3, 3-5
Program Control, 5-2
Program Mode, 6-5
Program memory, 5-55, 5-73
Programmable gain amplifier (PGA), 7-2

Programmable filter, 4-14
Programming, 2-4
Pull-Down Menu, 6-18

Q

QUICK option, 5-40
Quick Start, 1
Quick-disconnect terminal, 4-2
QuickStart, C-1

R

RAM, 3-5, 5-65
RAM error, 5-65
RAW, 5-64, 6-4
Radiometry, 6-1, 6-17
Raw binary, 5-2
Raw D/A counts, 4-15
RC, 4-14, 4-15
READ...QUICK, 6-8
READ/WRITE, 7-10
REMOTE, 3-1
REMOTEDN, 6-13
REMOTEU, 6-14
REN, D-2, D-4
RESET, 5-43, 7-10, 7-11
RETSUB, 5-3
Real Time Clock, 5-76
Reference, 2-4, 7-1
Registration, 3-2
Relay Board, 7-26
Relay Control, 7-26
Relay Control Command, 7-26
Relays, 4-11
Remote, 6-1
Replaceable Parts, 8-7
Reset, 7-33
Reset Circuit, 7-6
Return authorization, 2-3
ROM, 3-5, 5-65
ROM error, 5-65
RQS, 5-79
RS-232, 6-12
RTC, 5-76
RTDn [C|F], 6-4
RTDn[C|F], 5-64
RUN, 3-1

S

SAVE option, 5-58
Safe Control Set-Ups, 2-2
Safety ground wire, 3-3
Save, 3-2
SCAN, 5-61, 5-67
SCAN CLOCK, 6-11
SCANS, 5-75
Scale, 5-10

SEC, 5-36, 5-68
SELECT SLOT, 7-28
SELF-ID, 5-59
Secondary address, 3-2
Seconds, 5-50
Self identification, 7-33
Self-ID, 4-1, 7-33
Self-ID component, 4-1
Self-ID Resistor, 9-3
Self-test, 5-65
Serial Poll, 5-79
Setting Up, 3-2, 4-2
Setup, 2-4
SGND, 4-7
Sideboard, 4-12, 7-28
Signal Checks, 8-4, 8-5
Signal Conditioning, 9-2
Single-Ended vs Differential Input, 4-13
Single-ended input configuration, 4-2
Single-pole input filter, 4-14
Slot, 5-57
Slot 1, 7-13, 7-33
Slot Assignments, 7-6
Slot connector, 7-7
Solid-state relays, 7-26
Source/Delay/Measure, 6-15
SPD, D-4
SPE, D-4
Specifications, 7-4, 7-34
SRQ, 3-1, 5-2, D-2
STAMP, 5-7
STEP1/STEP2, 5-1
STROBE, 7-10, 7-11
Static Sensitive Devices, 8-6
Strain, 4-10
Strobe Analog Output, 7-33
SUBR, 5-3
Subroutines, 5-3
Supply current, 7-5
Switch 7, 5-58
Switch 8, 5-58
SYST, 5-1
SYST :ERR ?, B-1
SYST :FORM, 5-76
SYST :SLOT, 4-1
SYST :STAMP, 5-3, 5-75
System Checks, 8-3
System Controller, 7-5
System problems, 8-3

T

TALK, 3-1
TC, 5-7, 5-75
TCn[C|F], 5-64, 6-4
Terms and Conventions, 2-1
“Computer”, 2-1
“Controller”, 2-1
“PC”, 2-1

Braces “{ }”, 5-4
Colon “:”, 5-4
Range of channels, 5-4
Square brackets “[]”, 5-4
Vertical bar “|”, 5-4
Testing 8-Bit DACs, 6-15
Theory of Operation, 7-1, 7-5
Thermocouples, 4-10
Threads, 3-2
Threshold, 5-66
TIME, 5-61
Time, 5-50
Time field, 5-77
Time Stamping, 5-3, 5-75, 6-9
TRG1, 4-7, 5-1, 5-2, 5-17, 5-19, 5-66, 7-15
TRG1 input, 5-66
Trigger – Slot 2, 4-7
Trigger Command, 7-16
Trigger cable, 3-3, 4-8
Trigger circuit, 7-15
Trigger input, 4-7
Trigger jumpers, 4-10
Triggered Acquisition, 6-8
Triggering, 5-2, 6-9
Troubleshooting, 8-1, 8-3
TTL, 4-11, 7-23

U

Unaddress Commands, D-5
Uniline, D-3
Unpacking, 3-1
USEC, 5-36
User-configured components, 7-11

V

Virtual slot, 4-3
Voltage drop, 4-14

W

W201, 4-7
WAIT, 5-2
WARNING, 2-1
WAV1, 5-1
Wall mount transformer, 7-5
WEND, 5-70
WHILE, 5-70
WHILE...WEND, 5-2
WORD, 5-7, 5-75
WorkBench, 6-19

Y

Year, 5-50

Service Form

Model No. _____ Serial No. _____ Date _____

Name and Telephone No. _____

Company _____

List all control settings, describe problem and check boxes that apply to problem. _____

- | | | |
|--|--|--|
| <input type="checkbox"/> Intermittent | <input type="checkbox"/> Analog output follows display | <input type="checkbox"/> Particular range or function bad; specify |
| <input type="checkbox"/> IEEE failure | <input type="checkbox"/> Obvious problem on power-up | <input type="checkbox"/> Batteries and fuses are OK |
| <input type="checkbox"/> Front panel operational | <input type="checkbox"/> All ranges or functions are bad | <input type="checkbox"/> Checked all cables |

Display or output (check one)

- | | |
|-----------------------------------|--|
| <input type="checkbox"/> Drifts | <input type="checkbox"/> Unable to zero |
| <input type="checkbox"/> Unstable | <input type="checkbox"/> Will not read applied input |
| <input type="checkbox"/> Overload | |

- | | |
|---|--|
| <input type="checkbox"/> Calibration only | <input type="checkbox"/> Certificate of calibration required |
| <input type="checkbox"/> Data required | |

(attach any additional sheets as necessary)

Show a block diagram of your measurement system including all instruments connected (whether power is turned on or not). Also, describe signal source.

Where is the measurement being performed? (factory, controlled laboratory, out-of-doors, etc.)

What power line voltage is used? _____ Ambient temperature? _____ °F

Relative humidity? _____ Other? _____

Any additional information. (If special modifications have been made by the user, please describe.)

Be sure to include your name and phone number on this service form.

KEITHLEY DATA ACQUISITION

Data Acquisition Division

Keithley Instruments, Inc. • 440 Myles Standish Blvd. • Taunton, MA 02780 • (508) 880-3000 • Fax: (508) 880-0179

AUSTRIA: Keithley Instruments GesmbH • Rosenhügelstrasse 12 • A-1120 Wien • 0222-804-6548 • Fax: 0222-804-3597
FRANCE: Keithley Instruments SARL • 3 Allée des Garays • B.P. 60 • 91121 Palaiseau Cédex • 01-60-11-51-55 • Fax: 01-60-11-77-26
GERMANY: Keithley Instruments GmbH • Landsberger Str. 65 • D-8034 Germering • 089-849307-0 • Fax: 089-84930759
GREAT BRITAIN: Keithley Instruments, Ltd. • The Minster • 58 Portman Road • Reading, Berkshire RG3 1EA • 0734-575666 • Fax: 0734-596469
ITALY: Keithley Instruments SRL • Viale S. Gimignano 38 • 20146 Milano • 02-48303008 • Fax: 02-48302274
JAPAN: Keithley Instruments Far East KK • Sumiyoshi 24 Bldg., Room 201 • 2-24-2 Sumiyoshi-cho • Naka-ku, Yokohama 231 • 81-45-201-2246 • Fax: 81-45-201-2247
NETHERLANDS: Keithley Instruments BV • Avelingen West 49 • 4202 MS Gorinchem • Postbus 559 • 4200 AN Gorinchem • 01830-35333 • Fax: 01830-30821
SWITZERLAND: Keithley Instruments SA • Kriesbachstrasse 4 • 8600 Dübendorf • 01-821-9444 • Fax: 01-820-3081
TAIWAN: Keithley Instruments Taiwan • 3rd Floor, Spring Plaza 6 • Section 3, Min Chuan East Road • Taipei, R.O.C. • 886-2-501-7065 • Fax: 886-2-501-5329