

**Tektronix®**

**GPIB  
PROGRAMMING  
GUIDE**

INSTRUCTION MANUAL

**Tektronix**<sup>®</sup>

COMMITTED TO EXCELLENCE

**PLEASE CHECK FOR CHANGE INFORMATION  
AT THE REAR OF THIS MANUAL.**

# **GPIB PROGRAMMING GUIDE**

INSTRUCTION MANUAL

Tektronix, Inc.  
P.O. Box 500  
Beaverton, Oregon 97077

Serial Number \_\_\_\_\_


First Printing AUG 1981  
Revised Printing OCT 1981

070-3985-00

The information presented in this manual is provided for instructional purposes only. Tektronix, Inc. does not warrant or represent in any way the accuracy or completeness of any program herein or its fitness for a user's particular purpose.

Copyright © 1981 Tektronix, Inc. All rights reserved. Contents of this publication may not be reproduced in any form without the written permission of Tektronix, Inc.

Products of Tektronix, Inc. and its subsidiaries are covered by U.S. and foreign patents and/or pending patents.

TEKTRONIX, TEK, SCOPE-MOBILE, and  are registered trademarks of Tektronix, Inc. TELEQUIPMENT is a registered trademark of Tektronix, U.K. Limited.

Printed in U.S.A. Specification and price change privileges are reserved.

PREFACE

Tektronix GPIB instruments are designed to go beyond GPIB compatibility to enhance the GPIB capability of instrument functions you choose for your system. To accomplish this, Tektronix instruments on the bus use common codes and formats. As a result, the software interface between controller, instruments, and you, the user, can be more efficient. System integration and application programming can be faster and easier.

This manual is intended to help you realize these benefits. The sections that follow contain instructions and program examples that are the software equivalent of the GPIB cable you use to make electrical connection between the controller and instruments in your system. With this manual you can make the software connection between the pieces of your system.

GPIB Interface Capabilities in Detail	1-1
Controller Function Descriptions	1-2
Signal Acquisition (SAC)	1-3
Service Request (SRQ)	1-4
Interface Clear (IPC)	1-5
Remote Enable (REN)	1-6
List or Memory (LOI)	1-7
High-Level Output—The PRINT Statement	1-8
Input	1-9
Explanation	1-10
Special Keys—The Default Response Codes	1-11
Handling Device-Dependent Messages with PRINT	1-12
High-Level Input—The INPUT Statement	1-13
Input	1-14
Handling Device-Dependent Data	1-15
Special INPUT Codes	1-16
Special Characters—Hexes	1-17
Mixed String and Numeric Variables	1-18
Alternate Delimiters	1-19

UNIT

Technical staff instruments are designed to be beyond GIB capabilities to enhance the GIB capabilities of instrument functions you choose for your system. To accomplish this, Technical Instruments on the GIB are common color and format. As a result, the volume interface between controller instruments, and you, the user, can be more efficient. System Instruments and application programming can be faster and easier.

This manual is intended to help you realize these benefits. It contains that information on instrument and system operation that the volume instrument should provide. This manual is intended to help you realize these benefits. It contains that information on instrument and system operation that the volume instrument should provide. This manual is intended to help you realize these benefits. It contains that information on instrument and system operation that the volume instrument should provide.

Technical staff instruments are designed to be beyond GIB capabilities to enhance the GIB capabilities of instrument functions you choose for your system. To accomplish this, Technical Instruments on the GIB are common color and format. As a result, the volume interface between controller instruments, and you, the user, can be more efficient. System Instruments and application programming can be faster and easier.

TABLE OF CONTENTS

<b>SECTION 1 - GETTING STARTED</b>	<b>1-1</b>
Hardware <u>and</u> Software Compatibility	1-1
Flexible Instrumentation Through Modular Design	1-2
Instruments in the TM 5000 Series	1-2
Device-Dependent Messages	1-3
Status Reporting	1-4
A Learn Mode Program	1-6
A Talk/Listen Routine	1-7
 <b>SECTION 2 - 4050 GCS CONTROLLERS</b>	 <b>2-1</b>
 4050-SERIES GPIB CAPABILITY	 2-3
GPIB Interface Compatibility In Detail	2-3
Controller Function Considerations	2-4
Signal Attention (ATN)	2-4
Service Request (SRQ)	2-4
Interface Clear (IFC)	2-5
Remote Enable (REN)	2-5
End or Identify (EOI)	2-5
 4050-SERIES GPIB INPUT/OUTPUT	 2-6
Addressing	2-6
 High-Level Output--The PRINT Statement	 2-8
Purpose	2-8
Explanation	2-8
Special Case--The Default Secondary Address	2-9
Sending Device Dependent Messages with PRINT	2-9
 High Level Input--The INPUT Statement	 2-11
Purpose	2-11
Receiving Device Dependent Data	2-12
Special INPUT Cases	2-14
Default Secondary Address	2-14
Mixed String and Numeric Variables	2-14
Alternate Delimiters	2-14

Low-Level Output--The WYBTE Statement	2-16
Purpose	2-16
Explanation	2-17
Sending Interface Messages	2-17
Universal Commands	2-17
Addressed Commands	2-17
Sending Binary Data to Specified Listeners	2-18
Low-Level Input--The RBYTE Statement	2-19
Purpose	2-19
Explanation	2-19
GPIB INTERRUPTS	2-21
Handling SRQ Interrupts	2-21
Using ERR?	2-23
Handling EOI Interrupts	2-25
4050 INTERRUPT-HANDLING STATEMENTS	2-27
The OFF Statement	2-27
Purpose	2-27
Explanation	2-27
The ON...THEN... Statement	2-28
Purpose	2-28
Explanation	2-28
The POLL Statement	2-29
Purpose	2-29
Explanation	2-29
Unspecified Devices on the Bus	2-31
Specified Devices Not on the Bus	2-31

The WAIT Statement	2-32
Purpose	2-32
Explanation	2-32
The WAIT Routine	2-33
Purpose	2-33
Explanation	2-33
 UTILITY ROUTINES	 2-34
Get A Status Byte with WBYTE and RBYTE	2-34
Convert a Status Byte to an 8-Bit Array	2-35
Send Identical Messages to Three GPIB Addresses Simultaneously	2-36
 4052/GPIB SEND AND RECEIVE	 2-37
 <b>SECTION 3 - DC 5010 AND DC 5009 UNIVERSAL COUNTER/TIMERS</b>	 <b>3-1</b>
DC 5009 and DC 5010 Functions	3-1
Frequency A	3-1
Period A	3-1
Width A	3-1
Time A to B	3-2
Rise/Fall A	3-2
Probe Compensation	3-2
Null	3-2
Totalize A	3-2
Events B During A	3-2
Ratio B/A	3-3
Measurement Control	3-3
Averaging	3-3
Auto-Trigger	3-3
Auto-Resolution	3-3
Input Signal Condition	3-4
 GPIB Operation	 3-4
IEEE 488 Bus Address	3-4
Power-Up	3-4



Making GPIB Counter Measurements	3-6
Setting Up the Measurement	3-6
Front-Panel Controls	3-6
Input Conditioning	3-7
Rise/Fall	3-7
Getting a Reading	3-8
Time Interval and Width Measurements	3-10
Triggering An Updated Reading	3-11
Amplitude Measurements	3-13
Application Programs	3-14
Signal Characterization	3-14
Measuring Phase	3-18
Error Decoding	3-19
Status Routine	3-21
<b>SECTION 4 - DM 5010 PROGRAMMABLE DIGITAL MULTIMETER</b>	<b>4-1</b>
DM 5010 Functions	4-1
DC Volts	4-1
AC Volts	4-1
Ohms	4-2
Diode Test	4-2
Current	4-2
Autorange	4-2
Front/Rear Input	4-2
Null	4-2
Calculations	4-3
Averaging	4-3
Scaling and Offset	4-3
dBm or dBr	4-3
Compare	4-3

GPIB Operation	4-3
IEEE 488 Bus Address	4-3
Power-Up	4-4
Selecting DM 5010 Functions	4-5
Default Range Settings	4-5
Initializing Settings	4-5
Calculations	4-6
Averaging	4-6
Scaling	4-6
dBm, dBr	4-7
Compare to Limits	4-7
Triggering and Acquiring Readings	4-8
Run Mode	4-8
Triggered Mode	4-8
Ac Settling Time	4-9
Application Examples	4-10
Data Logging	4-10
Power Supply Setting--Three DM 5010 Display Modes	4-12
A Resistor Sorting Program	4-14
Monitoring Line Voltage	4-15
In-Circuit Current Measurement	4-17
Averaging Using RDY? Or OPC	4-18
Find Max Value	4-20
P-P Reading of Square Wave	4-22
Low-Frequency Rms	4-23
Error Decoding	4-26
<b>SECTION 5 - FG 5010 PROGRAMMABLE FUNCTION GENERATOR</b>	<b>5-1</b>
FG 5010 Functions	5-1
Frequency	5-1
Amplitude	5-1
Offset	5-1

## GPIB PROGRAMMING GUIDE

Normal/Complement	5-2
Triggered, Gated, or Nburst	5-2
Phase	5-2
Symmetry	5-2
Output Hold	5-2
Phase Lock	5-2
Modulation	5-3
Swept Frequency	5-3
GPIB Operation	5-3
IEEE 488 Bus Address	5-3
Power-Up Self-Test	5-3
Programming FG 5010 Functions	5-5
Setting Commands and Queries	5-5
High-Level Settings Query	5-5
Low-Level Settings Query	5-6
Special Functions with Group Execute Trigger (GET)	5-6
Status Reporting	5-7
FG 5010 Applications Programs	5-7
FG 5010 ERR? and Status Byte Message Table	5-8
FG 5010 Software Frequency Sweep	5-11
FG 5010 VCF Mode Voltage Calculation	5-12
FG 5010 Pulse Generator Emulator	5-15
FG 5010 LEARN MODE -- Create Settings Different From INIT	5-20
Binary Settings Query	5-21
Low-Level Stored Settings Query	5-23
<b>SECTION 6 - PS 5010 PROGRAMMABLE POWER SUPPLY</b>	<b>6-1</b>
GPIB Operation	6-4
IEEE Bus Address	6-4
Power-Up Conditions	6-4
Programming PS 5010 Functions	6-6
Initialized Settings	6-6
Programming Voltage and Current Limits	6-6

Incrementing Voltages and Currents	6-7
Incrementing for Specified Random Values	6-7
Output Control	6-9
Front Panel Lockout	6-10
Regulation Status Reporting	6-11
Query All Supplies	6-11
Regulation Interrupt Reporting	6-11
Low-Level Setting	6-15
Application Programs Using the PS 5010	6-17
Series-Connected Supplies	6-17
Error Decoding	6-20
Zener-Diode Test	6-22
Graphic Subroutine	6-28
<b>SECTION 7 - SYSTEM APPLICATIONS</b>	<b>7-1</b>
Frequency Response Plot Program	7-1
FG 5010, DM 5010, and 4052	
Generating a Higher-Resolution DC Voltage	7-6
Automatic Checkout of Device Under Test	7-10
Full Error Message Decoding	7-18
<b>APPENDIX - INSTRUMENT COMMANDS</b>	<b>A-1</b>
INPUT/OUTPUT COMMANDS	A-15
STATUS COMMANDS	A-20
SYSTEM COMMANDS	A-23
STATUS AND ERROR REPORTING	A-25
TM 5000 IEEE 488 INTERFACE FUNCTIONS	A-30
TM 5000 RESPONSE TO IEEE 488 INTERFACE MESSAGES	A-31

1-1	Introduction
1-2	Objectives
1-3	Scope
1-4	Methodology
1-5	Organization
1-6	References
1-7	Appendix
1-8	Index
1-9	Summary
1-10	Conclusion
1-11	Recommendations
1-12	References
1-13	Appendix
1-14	Index
1-15	Summary
1-16	Conclusion
1-17	Recommendations
1-18	References
1-19	Appendix
1-20	Index
1-21	Summary
1-22	Conclusion
1-23	Recommendations
1-24	References
1-25	Appendix
1-26	Index
1-27	Summary
1-28	Conclusion
1-29	Recommendations
1-30	References
1-31	Appendix
1-32	Index
1-33	Summary
1-34	Conclusion
1-35	Recommendations
1-36	References
1-37	Appendix
1-38	Index
1-39	Summary
1-40	Conclusion
1-41	Recommendations
1-42	References
1-43	Appendix
1-44	Index
1-45	Summary
1-46	Conclusion
1-47	Recommendations
1-48	References
1-49	Appendix
1-50	Index
1-51	Summary
1-52	Conclusion
1-53	Recommendations
1-54	References
1-55	Appendix
1-56	Index
1-57	Summary
1-58	Conclusion
1-59	Recommendations
1-60	References
1-61	Appendix
1-62	Index
1-63	Summary
1-64	Conclusion
1-65	Recommendations
1-66	References
1-67	Appendix
1-68	Index
1-69	Summary
1-70	Conclusion
1-71	Recommendations
1-72	References
1-73	Appendix
1-74	Index
1-75	Summary
1-76	Conclusion
1-77	Recommendations
1-78	References
1-79	Appendix
1-80	Index
1-81	Summary
1-82	Conclusion
1-83	Recommendations
1-84	References
1-85	Appendix
1-86	Index
1-87	Summary
1-88	Conclusion
1-89	Recommendations
1-90	References
1-91	Appendix
1-92	Index
1-93	Summary
1-94	Conclusion
1-95	Recommendations
1-96	References
1-97	Appendix
1-98	Index
1-99	Summary
1-100	Conclusion
1-101	Recommendations
1-102	References
1-103	Appendix
1-104	Index
1-105	Summary
1-106	Conclusion
1-107	Recommendations
1-108	References
1-109	Appendix
1-110	Index
1-111	Summary
1-112	Conclusion
1-113	Recommendations
1-114	References
1-115	Appendix
1-116	Index
1-117	Summary
1-118	Conclusion
1-119	Recommendations
1-120	References
1-121	Appendix
1-122	Index
1-123	Summary
1-124	Conclusion
1-125	Recommendations
1-126	References
1-127	Appendix
1-128	Index
1-129	Summary
1-130	Conclusion
1-131	Recommendations
1-132	References
1-133	Appendix
1-134	Index
1-135	Summary
1-136	Conclusion
1-137	Recommendations
1-138	References
1-139	Appendix
1-140	Index
1-141	Summary
1-142	Conclusion
1-143	Recommendations
1-144	References
1-145	Appendix
1-146	Index
1-147	Summary
1-148	Conclusion
1-149	Recommendations
1-150	References
1-151	Appendix
1-152	Index
1-153	Summary
1-154	Conclusion
1-155	Recommendations
1-156	References
1-157	Appendix
1-158	Index
1-159	Summary
1-160	Conclusion
1-161	Recommendations
1-162	References
1-163	Appendix
1-164	Index
1-165	Summary
1-166	Conclusion
1-167	Recommendations
1-168	References
1-169	Appendix
1-170	Index
1-171	Summary
1-172	Conclusion
1-173	Recommendations
1-174	References
1-175	Appendix
1-176	Index
1-177	Summary
1-178	Conclusion
1-179	Recommendations
1-180	References
1-181	Appendix
1-182	Index
1-183	Summary
1-184	Conclusion
1-185	Recommendations
1-186	References
1-187	Appendix
1-188	Index
1-189	Summary
1-190	Conclusion
1-191	Recommendations
1-192	References
1-193	Appendix
1-194	Index
1-195	Summary
1-196	Conclusion
1-197	Recommendations
1-198	References
1-199	Appendix
1-200	Index

## SECTION 1

## GETTING STARTED

One of the benefits possible by choosing Tektronix instruments when you configure a GPIB system is reduced time from shipping cartons to an operating system. This section has that in mind by: 1) Introducing the concepts designed into Tektronix instruments for easier programming, and 2) Listing programs you can enter and run for fast results.

**Hardware and Software Compatibility**

To meet the objectives of both easier user programming and flexible system configuration, Tektronix has developed a standard for codes and formats. This standard puts GPIB instrument control into terms that are very close to self-explanatory while placing as few constraints as possible on how the designer chooses to implement instrument functions.

This standard was used to coordinate the design of the new TM 5000 Series of instruments and Tektronix' IEEE-488 line of digitizing oscilloscopes, spectrum analyzers, transient digitizers, and other products. This means you need to learn only one format, BASIC syntax, and a uniform set of codes. There is a common message structure for all the system's instruments. This contrasts with a typical systems programmer's problem of dealing with half a dozen different sets of format and syntax, or having to write half a dozen software drivers to translate them into a common format.

Status and error reporting is handled the same way by all instruments. This allows common support subroutines and fewer program modifications to follow system changes.

Program source code is high-level, and instrument commands are in "engineering English." (See the command list in the Appendix for examples.) Selecting commands is like selecting front panel controls: commands equal front panel labels, which equal the engineering terms commonly used for those functions. For example, a typical command string to the function generator might be "FUNCTION SQUARE;FREQUENCY 1500;OFFSET 2.2;AMPLITUDE 4.5;SYMMETRY 15." Not only are the commands simple to remember, but the program listings are readable and self-documenting. This is particularly important when you or someone else picks up the program listing later and wants to understand what the program does.

There is even a provision for "programming in reverse." The LEARN mode permits someone more used to instrument operation than programming to "write" a program by setting front panel controls for the desired conditions and then, with a single keystroke, converting the configuration into error-free program instructions.

### Flexible Instrumentation Through Modular Design

To meet the objective of flexible system configuration, the TM 5000 line provides compactness and flexibility built up of plug-in instrumentation modules with common dimensions and interface characteristics. The TM 5000 Series of fully programmable IEEE-488 instruments is based on the TM 500 concept and on plug-in concepts pioneered by Tektronix through several generations of oscilloscopes.

This integrated set of modular instruments has the convenience and appearance of a compact monolithic system, while retaining the configuration flexibility and expandability of separate instruments. TM 5000 instruments are not only modular, but compact and lightweight. These features allow small systems to be used in confined spaces and at remote locations. Programmable instrument systems are particularly appropriate for many such applications because their features permit lightly skilled operators to perform relatively complex maintenance and test procedures under software guidance and control.

### Instruments in the TM 5000 Series

The initial set of TM 5000 Series instrument modules includes: a 135-MHz universal counter/timer; a 350-MHz universal counter/timer; a "smart" digital multimeter; an extremely flexible 20-MHz function generator, and a triple power supply that furnishes the entire range of commonly used voltages. All are IEEE-488 compatible and all are microprocessor-based designs.

Every front-panel function--even features such as AC/DC coupling and trigger level--is programmable via the IEEE-488 bus. A major convenience common to all these Tektronix TM 5000 Series instruments is the ability to read each instrument's primary address in its front panel digital display by pushing an INST ID button on the front panel. This eliminates having to locate a DIP switch tucked away behind or inside an instrument, then decoding its binary code into its decimal equivalent.

All the instruments, when first powered up, go through an extensive self-test and diagnostic routine to check the health of ROM, RAM, I/O, and the functionality of other blocks. If no internal error is found, the instrument enters the local state with a standard set of default settings. In addition, SRQ on the IEEE-488 bus is asserted to let the controller know that the instrument is on-line.

If an internal error should be discovered, an error code is immediately displayed on the instrument front panel, which would lead, with the help of the instrument's instruction manual, to the location of the specific problem. The self-test routine of any instrument can later be triggered at any time from the controller by the simple command "TEST".

### Device-Dependent Messages

The Tektronix Codes and Formats standard defines a message to be a complete block of information. It begins when a device starts sending data and ends when EOI is sent or received concurrently with the last data byte. Because an instrument sending a message may be interrupted by the controller taking control, perhaps conducting a serial poll, when the instrument becomes a talker again, it should resume sending the message. Thus, a message is a complete block that begins when a device enters the talker active state following a reset or a previously sent EOI and ends with EOI.

There is a further refinement to the message convention. When a device is made a talker, it should always say something. If it has nothing to say, it sends a byte of all ones concurrent with EOI. This prevents tying up the GPIB while the controller waits for a device to talk that will never send a message. Measurement instruments, such as the DM 5010 Programmable Digital Multimeter, interpret the "talked with nothing to say" condition as a request for data and output a reading.

Device dependent messages are ASCII characters. The message syntax unit for a settings command is:

```
HEADER<SPACE>ARGUMENT;
```

Example: OUTPUT ON;

The header is a mnemonic for a function. The ";" delimits a message unit so more than one may be sent in a message (it is optional at the end



of a message). Each message unit denotes a front panel function or system function; any number of message units may be sent together. Tektronix instruments do not finish syntax checking until the last byte of the entire message is received. The settings are then executed only when no errors or settings conflicts are found. If an error is found, the instrument asserts the SRQ line and reports an appropriate status byte. The message is discarded if errors are found, even if an error occurs in only one message unit. **Under no circumstances should a device execute a message it does not understand.** Some non-Tektronix devices do not follow this convention--with disastrous results. (A particular power supply can be sent four letter O's instead of four zeros, a common human mistake, and this supply will go to its maximum voltage output instead of the intended zero volts.)

Problems can be created by instruments which execute each individual command as received. For example, a programmable high-voltage power supply that can be set as high as 1000 V is set to 10 V out with a current limit of 2A, then is sent the message "VOLTS 1000, CURR 10E-03", that is, 1000 volts output limited at 10 mA. If the supply goes to 1000 V output immediately upon receiving the first part of the message while the current limit is still 2 A, the value from the previous setting, the supply is likely to crowbar or damage the equipment connected to it. For proper operation, the programmer would have to change the current setting first, and then change the voltage in a separate message. The power supply should wait until the entire message is received before executing any command in the message, as do Tektronix instruments. Then, it should execute all commands in the message at once so they are order independent.

### Status Reporting

While the IEEE 488 standard defines a service request function for an instrument to send a status byte to the controller, the standard does not define the meaning of the bits except for bit 7. (Bit 7 means that a device is, or is not, requesting service.) Because there is a common need for instruments to report certain kinds of status or errors to the controller, Tektronix defines a status byte convention for this. One common need is for instruments to report if they are busy or ready (assigned to bit 5). Another common need is for instruments to report if they are encountering abnormal conditions (assigned to bit 6).

There are more complex conditions besides busy/ready or normal/abnormal. These are listed in the Appendix. While these status bytes are generally useful for most purposes, certain instruments may have conditions that are peculiar to them. Bit 8 is used to indicate that the status byte is not the common type but particular to an instrument.

Standard coding for the status byte enhances user programming. Because the instruments have common status byte coding, a common status byte handling routine is written for all instruments, not a separate one for each.

Beyond the status byte reported during a serial poll, more information is available to the programmer through the use of queries. Queries instruct Tektronix instruments to send back specified information. For example: "ERR?" sent to a Tektronix instrument prompts the reply "ERR <NUM>," when the instrument is talk addressed. In the case of "ERR?", the response defines the particular cause of an SRQ, whose general type is reported in the status byte.

Queries take the form of a Header followed by a question mark.

HEADER?

Some other queries and their uses are:

ID? makes an instrument identify itself by sending information as to instrument type, model number, firmware version, etc. This feature is useful for identifying a particular device for self-configuring systems and interrupt handling.

SET? requests an instrument to send the computer its present settings and other current state information. Sending this information back to the instrument at a later time returns the instrument to the state it was in when it received SET? This query makes it possible to develop a program using an instrument's front panel in a LEARN mode as discussed above.

Defining a standard way to obtain output from Tektronix instruments is more than a convenience. When all instruments use the same form to perform similar functions, you only have to learn one convention, not many, making your programs simpler and faster to write.

A Learn Mode Program

Here's a program that uses the "SET?" command to obtain up to five setups from a TM 5000 instrument. To try it out:

1. Enter the program on a 4050-Series Controller. **Change** the number "22" in line 140 if you intend to operate this program with an instrument whose GPIB address is other than 22. The factory-set addresses for TM 5000 instruments are: 16=DM 5010, 18=DC 5009, 20=DC 5010, 22=PS 5010, and 24=FG 5010.
2. Connect a **single** TM 5000 instrument to the controller with a GPIB cable (one mainframe with one plug-in).
3. Turn on instrument power.
4. Type RUN and press RETURN.

```

100 REM           [ High level learn mode program ]
110 INIT
120 ON SRQ THEN 700
130 DIM A$(300),B$(300),C$(300),D$(300),E$(300),G$(300),S$(300)
140 P=22
150 PRINT @P:"USER ON"
160 PAGE
170 PRINT
180 PRINT "This program allows you to store up to five front panel ";
190 PRINT "settings."
200 PRINT
210 PRINT "Press the INST ID button to store the front panel settings."
220 PRINT
230 REM           ***** Send settings to the controller *****
240 FOR I=1 TO 5
250 PRINT "          READY FOR FRONT PANEL SET-UP #";I
260 WAIT
270 IF S<>67 THEN 260
280 PRINT @P:"SET?"
290 INPUT @P:S$
300 GO TO I OF 310,330,350,370,390
310 A$=S$
320 GO TO 410
330 B$=S$
340 GO TO 410
350 C$=S$
360 GO TO 410
370 D$=S$
380 GO TO 410
390 E$=S$
400 GO TO 410
410 NEXT I

```

## GPIB PROGRAMMING GUIDE

```
420 REM ***** Send stored settings back to the instrument *****
430 PRINT
440 PRINT "Press INST ID button to send the stored settings back."
450 PRINT
460 FOR J=1 TO 5
470 WAIT
480 IF S<>67 THEN 470
490 PRINT "STORED SETTINGS #";J;":"
500 GO TO J OF 510,540,570,600,630
510 PRINT @P:A$
520 PRINT A$
530 GO TO 660
540 PRINT @P:B$
550 PRINT B$
560 GO TO 660
570 PRINT @P:C$
580 PRINT C$
590 GO TO 660
600 PRINT @P:D$
610 PRINT D$
620 GO TO 660
630 PRINT @P:E$
640 PRINT E$
650 GO TO 660
660 PRINT
670 NEXT J
680 PRINT @P:"USER OFF"
690 END
700 REM ***** SRQ Service routine *****
710 POLL D,S;P
720 RETURN
```

### A Talk/Listen Routine

This program allows you to converse with TM 5000 instruments, prompting you to enter the instrument address and commands. The program handles all other details to establish communication and print the results.

To use this program:

1. Enter the program, changing the address list in line 250 to include only those instruments in the system.
2. Connect the instruments to the GPIB.
3. Turn on power.
4. Type RUN and press RETURN.

GPIB PROGRAMMING GUIDE

```

100 INIT
110 ON SRQ THEN 250
120 DIM O$(200),P$(200)
130 PRINT "JITM 5000 TALKER/LISTENER ProgramG"
140 PRINT "JENTER ADDRESS (1-30) OF INSTRUMENT TO TALK TO: ";
150 INPUT O
160 PRINT "ENTER MESSAGE TO INSTRUMENT: ";
170 INPUT O$
180 PRINT @O:O$
190 IF POS(O$,"?",1) OR POS(O$,"SEND",1) THEN 210
200 GO TO 140
210 INPUT @O:P$
220 PRINT
230 PRINT "Response = ";P$
240 GO TO 140
250 POLL Q,R;16;18;20;22;24
260 PRINT "JIISTATUS BYTE = ";R
270 PRINT @O:"ERR?"
280 INPUT @O:P$
290 PRINT "JIIEERROR CODE = ";P$
300 RETURN

```

SECTION 2

4050 GCS CONTROLLERS

The 4050 GRAPHIC COMPUTING SYSTEMS (4051, 4052, 4054) are well known for their capabilities as desktop graphic computers. Not so well known are their capabilities as IEEE 488 (GPIB) instrument controllers.

The 4052 and 4054 integrate a high-speed bit slice microcomputer, a high-resolution graphics display, 300K bytes of magnetic tape mass storage, and a GPIB interface in a single compact unit.

The 4051 is a lower cost, lower performance GCS controller featuring the same language mass storage and GPIB interface capability as the 4052 and 4054.

The 4050 SERIES GRAPHIC COMPUTING SYSTEMS all provide powerful hardware, supported by an enhanced version of easy-to-learn 4050 BASIC. 4050 BASIC incorporates a flexible I/O structure that allows simple addressing of GPIB instruments and peripherals. It also includes extensions for signal processing, graphics, and GPIB control.

The intent of this section is to provide 4050 GCS programming information which is specific to instrument control. The major topics are:

- \* 4050 GCS controller IEEE 488 capabilities
- \* GPIB Input/Output
- \* Interrupt handling
- \* Interrupt handling statements
- \* Utility routines
- \* 4052/GPIB send and receive program

Some of the application examples use routines such as CALL "WAIT" built into the 4052 and 4054 but not the 4051. These examples must be modified to run on a 4051.

The 4050 GRAPHIC COMPUTING SYSTEMS will be referred to as 4050 SYSTEM CONTROLLERS.

Users should have a firm understanding of the 4050 series Input/Output operations before continuing.

## GPIB PROGRAMMING GUIDE

For additional detailed information on 4050 BASIC or the GPIB refer to the 4050 SERIES GRAPHIC SYSTEMS REFERENCE MANUAL and the IEEE 488-1978 Standard, respectively.

4050-SERIES GPIB CAPABILITY

GPIB Interface Compatibility In Detail

The 4050 SERIES CONTROLLERS fall into the following interface function subsets as defined in IEEE Standard 488-1978:

Section 2.3 SH (Source Handshake Function)

SH1 -- completely compatible to guarantee the proper transfer of multiline messages. This function utilizes the DAV, NRFD, and NDAC handshake lines to effect each message byte transfer.

Section 2.4 AH (Acceptor Handshake Function)

AH1 -- completely compatible to guarantee the proper reception of remote multiline messages. This function utilizes the DAV, NRFD, and NDAC handshake lines to effect each message byte transfer.

Section 2.5 T (Talker Function)

TE3 -- basic extended talker, however, the 4050 SERIES CONTROLLER addresses itself internally and not over the GPIB. (T3--to the outside world).

Section 2.6 L (Listener Function)

LE1 -- basic extended listener, however, the 4050 SERIES CONTROLLER addresses itself internally and not over the GPIB. (L1--to the outside world).

Section 2.7 SR (Service Request Function)

SRO -- no capability to issue SRQ.

Section 2.8 RL (Remote Local Function)

RLO -- no capability.

Section 2.9 PP (Parallel Poll Function)

PPO -- no capability.

Section 2.10 DC (Device Clear Function)

DCO -- no capability.



Section 2.11 DT (Device Trigger Function)

DT0 -- no capability.

Section 2.12 C (Controller Function)

- C1 -- System controller
- C2 -- Sends IFC and takes charge
- C3 -- Sends REN
- C4 -- Responds to SRQ
- C28 -- Sends Interface Messages

**Controller Function Considerations**

**Signal Attention (ATN)**

This signal line is activated by the controller when peripheral devices are being assigned as listeners and talkers. Only peripheral addresses and control messages can be transferred over the data bus when ATN is active low (asserted). After ATN goes high (unasserted), only those peripheral devices which are assigned as listeners and talkers can take part in the data transfer. The 4050 SERIES CONTROLLER assumes it is the only source of this signal.

**Service Request (SRQ)**

Any peripheral device on the GPIB can request the attention of the controller by setting SRQ active low. The controller can respond by setting ATN active low and executing a serial poll to see which device is requesting service. This response to the SRQ interrupt is enabled by an ON SRQ THEN statement which is executed in the BASIC program. The serial poll is taken by executing the POLL statement. After the device requesting service is found, BASIC program control can be transferred to a service routine for that device. When the service routine is finished executing, program control returns to the main program. The device requesting service unasserts the SRQ line when polled. At least half of the devices connected to the GPIB must be turned on to prevent a spurious SRQ signal.

**Interface Clear (IFC)**

The IFC signal line is activated by the controller when it wants to place all interface circuitry in a predetermined quiescent state. The 4050 SERIES CONTROLLER assumes that it is the only source of this signal. IFC is activated momentarily at power up and each time the INIT statement is executed.

**Remote Enable (REN)**

The REN signal line is activated whenever the system is operating under program control. REN enables devices on the GPIB to ignore their front panel controls and respond to control messages received over the GPIB.

**End or Identify (EOI)**

The EOI signal can be used by the talker to indicate the end of a data transfer sequence. The talker activates EOI as the last byte is transmitted. When the controller is listening, it assumes that a data byte received with EOI is the last byte in the transmission. When the controller is talking, it can activate EOI concurrent with the last byte.

## 4050-SERIES GPIB INPUT/OUTPUT

**Addressing**

The IEEE 488-1978 Standard defines the basic addressing scheme for GPIB instruments by specifying how a device's listen address (MLA), talk address (MTA), and the secondary address (MSA) are established based on the primary address. However, it leaves several options open to the instrument designer, so it is important to know the individual addressing requirements for the instruments in your system.

All GPIB instruments must be set to a primary address within the range of 1 to 30. The 4050 SERIES CONTROLLER reserves address zero for itself and cannot address a GPIB device at that address.

Some instruments use one or more secondary addresses for subfunctions or parts of the instrument. These secondary addresses are also set within the range of 1 to 30. Other addresses are reserved for devices within the 4050 SERIES CONTROLLER. All instruments on the bus must have different addresses to assure proper data transfer.

All GPIB devices and peripherals (internal and external) are addressed in the same way. For example, the PRINT statement can write data on the screen, on the internal magnetic tape, or send ASCII data to an external GPIB interface device. The difference is the address specified in the statement.

The primary and secondary addresses are converted to the appropriate "absolute" MTA, MLA, or MSA by virtue of a specific 4050 BASIC I/O statement.

These 4050 BASIC I/O statements consist of PRINT and INPUT for high level data transfer and WBYTE and RBYTE for direct data line control and interface control.

TABLE 2-1  
GENERAL PURPOSE INTERFACE BUS ADDRESSES

GPIB PRIMARY and SECONDARY ADDRESSES			
DEVICE #	LISTEN ADDRESS	TALK ADDRESS	SECONDARY ADDRESS
# 0	32	64	96
# 1	33	65	97
# 2	34	66	98
# 3	35	67	99
# 4	36	68	100
# 5	37	69	101
# 6	38	70	102
# 7	39	71	103
# 8	40	72	104
# 9	41	73	105
# 10	42	74	106
# 11	43	75	107
# 12	44	76	108
# 13	45	77	109
# 14	46	78	110
# 15	47	79	111
# 16	48	80	112
# 17	49	81	113
# 18	50	82	114
# 19	51	83	115
# 20	52	84	116
# 21	53	85	117
# 22	54	86	118
# 23	55	87	119
# 24	56	88	120
# 25	57	89	121
# 26	58	90	122
# 27	59	91	123
# 28	60	92	124
# 29	61	93	125
# 30	62	94	126
# 31	63	95	127

High-Level Output--The PRINT Statement

Purpose

The PRINT statement outputs data items to a specified GPIB device in ASCII code. If a device address is not specified, then the ASCII data is sent to the 4050 display by default. (This default address is 32,12.)

**Syntax Form:**

$$\begin{aligned}
 & [ \text{Line number} ] \text{ PRI } [ \text{I/O address} ] \left[ \text{USI } \left\{ \begin{array}{l} \text{string constant} \\ \text{string variable} \\ \text{line number} \end{array} \right\} ; \right] \\
 & \left[ \left\{ \begin{array}{l} \text{string constant} \\ \text{string variable} \\ \text{numeric expression} \end{array} \right\} \left[ \left\{ \begin{array}{l} \text{string constant} \\ \text{string variable} \\ \text{numeric expression} \end{array} \right\} \right] \right] \dots [ ; ]
 \end{aligned}$$

**Descriptive Form:**

$$\begin{aligned}
 & [ \text{Line number} ] \text{ PRINT } [ \text{I/O address} ] \left[ \text{USING } \left\{ \begin{array}{l} \text{format string} \\ \text{format string variable} \\ \text{IMAGE line number} \end{array} \right\} ; \right] \\
 & \left[ \text{item to be printed } \left[ \left\{ \begin{array}{l} ; \\ ; \end{array} \right\} \text{item to be printed} \right] \right] \dots [ ; ]
 \end{aligned}$$

Explanation

Like all 4050 BASIC I/O statements, the PRINT statement provides for an optional I/O address. This primary address selects a peripheral device to receive the ASCII data.

GPIB primary addresses (1-30) are automatically converted to Listen Addresses. The controller listen addresses the instrument, handshakes all ASCII data items listed in the PRINT statement, asserts EOI with the last byte, and then untalks and unlistens all devices on the bus.

### Special Case--The Default Secondary Address

If no secondary address is specified, secondary address 12 (MSA 108) is assumed and will be automatically sent after the primary listen address:

```
PRINT @24:"FREQ 10E3"
```

sends MLA 56 (32+24), MSA 108 (96+12) to listen address device #24 (secondary address 12 is automatically sent).

To suppress this default secondary address, specify 32 as the secondary address in the PRINT statement:

```
PRINT @24,32:"FREQ 10E3"
```

sends only listen address 56 for device #24; the secondary address is suppressed by using secondary address 32.

TM 5000 instruments (and most others without extended listener capabilities) will accept either of the above cases.

The 7912AD illustrates the use of secondary addressing capabilities.

```
PRINT @1,1:"V/D 5"
```

sends MLA 33, MSA 97 (96+1) for device #1 (7912AD), secondary address #1 (7912AD default MSA of 0+1=MSA for 7A16P Programmable Amplifier).

### Sending Device Dependent Messages with PRINT

The following examples illustrate how the PRINT statement is used to send ASCII coded device dependent messages.

1. To send a literal message to device #24, the FG 5010 Function Generator at its factory-set address:

```
PRINT @24:"FREQ 2.0E+7"
```

The message in quotes must contain a valid ASCII coded and formatted message for the device which has been addressed. See the Appendix for a list of TM 5000 instrument commands.

2. String variables can be predefined before being sent to a device:

```
A$="FREQ 2.0E+7"
PRINT @24:A$
```

3. Numeric variables can be defined and used with either literal strings (a) or string variables (b) to complete a message:

```
(a) N=10
    PRINT @24:"AMPL ";N
```

```
(b) A$="AMPL "
    PRINT @24:A$;N;
```

In (a), the semicolon suppresses extra spaces between data items.

In (b), the second semicolon suppresses an automatic carriage return after the last data item. A space ("AMPL ") must separate the data items. As an alternative, a comma may be used to guarantee a space between the command header ("AMPL") and the argument N. (Actually, the comma causes a number of spaces, but TM 5000 instruments disregard them.)

4. Multiple message units can be strung together into one complex message with the use of message unit delimiters defined by the device being addressed. Tektronix Codes and Formats specifies a semicolon.

```
PRINT @24:"FREQ 2.0E+7;AMPL 10"
```

In this case, both the frequency and amplitude on device #24 are executed with one ASCII string.

5. To combine literal string and variables in a complex message be aware of the differences between the use of 4050 BASIC delimiters and their functions and where device dependent message unit delimiters are used. Example 4 could also be sent in the following format:

```
F=2.0E+7
PRINT @24:"FREQ ";F;" ;AMPL 10"
```

suppress  
extra  
spaces

message unit delimiter  
for "FREQ 2.0E+7:"

or

```
F=2.0E+7
A$="AMPL "
N=10
PRINT @24:"FREQ ";F;" ";A$;N
```

suppress  
extra  
spaces

suppress  
extra  
spaces

message limit delimiter

### High Level Input--The INPUT Statement

#### Purpose

The INPUT statement is used to input ASCII data from a peripheral device. Data is assigned to the numeric or string variables in the order listed in the INPUT statement. The occurrence of a specific End-of-Record character (default CR) or End-of-File character is used to delimit the ASCII data as it is assigned to string variables. Numeric variables have values assigned to them which are legal numbers detected in the ASCII data being received (all other ASCII data is discarded).

**Syntax Form:**

$$[ \text{Line number} ] \text{ INP } [ \text{I/O address} ] \left\{ \begin{array}{l} \text{array variable} \\ \text{string variable} \\ \text{numeric variable} \end{array} \right\} \left[ , \left\{ \begin{array}{l} \text{array variable} \\ \text{string variable} \\ \text{numeric variable} \end{array} \right\} \right] \dots$$

**Descriptive Form:**

[ Line number ] INPUT [ I/O address ] target variables for incoming  
data items which are formatted in ASCII code



The primary and secondary addresses are converted to MTA and MSA automatically. ASCII data input terminates when one of several conditions occurs:

1. When EOI occurs and the last target variable listed in the INPUT statement has a value assigned to it. If the instrument is through sending data and asserts EOI before data has been assigned to the last target variable, the controller will leave the device talk addressed even though no additional data is forthcoming.

NOTE

There is no "time-out" for this condition and the program "hangs" on the INPUT statement.

2. When the Record Separator character is encountered before the last ASCII data item is sent.
3. When the End-of-File character is encountered in the ASCII data (default is binary equivalent of decimal 255).

In cases 2 and 3, the controller may untalk and unlisten the bus before the talker has asserted EOI. The talker may still have data to transmit the next time it is addressed to talk.

Receiving Device Dependent Data

Receiving device-dependent data can be illustrated by using the query response capability of TM 5000 Series instruments.

Queries are commands that request information such as settings. The INPUT statement supplies the talk address, reads the data, and stores it in a variable.

```
DIM S$(300)
PRINT @22:"SET?"
INPUT @22:S$
PRINT "SETTINGS=";S$
```

In this case, the INPUT statement stored ASCII data in the string variable S\$; S\$ was dimensioned larger than the expected response (the 4050 Series default string dimension is 72 characters).

The same format is used to receive discrete setting data.

```
PRINT @22:"VPOS?;IPOS?"
INPUT @22:V$
```

Here the response would be VPOS <NUM>;IPOS <NUM> which is less than 72 characters and need not be dimensioned.

If the number argument is all you wish to obtain from the response, specify a numeric variable:

```
PRINT @22:"VPOS?;IPOS?"
INPUT @22:V,I
PRINT "V= ";V,"I= ";I
```

The numeric variable V contains the programmed value of the positive voltage setting and I contains the programmed value of the positive current setting.

An alternate method for storing numeric data is to dimension an array large enough to store each expected response and specify its position in the array.

```
DIM V(2,2)
PRINT @22:"VPOS?;IPOS?;VNEG?;INEG?"
INPUT @22:V(1,1),V(1,2),V(2,1),V(2,2)
PRINT V
```

The example shows the numeric value of the positive supply's programmed voltage to be in row 1, column 1, and the positive current supply's programmed value to be in row 1, column 2. The negative supply data is stored in row 2 of the array.

The INPUT statement's high level method of talk addressing is also used to receive measurement data:

```
PRINT @16: "SEND"
INPUT @16: M1
```

Device #16 represents the DM 5010 at its factory-set address. The "SEND" command tells the DM 5010 to send a measurement. The value of the measurement is stored in numeric variable M1.

## Special INPUT Cases

**Default Secondary Address.** If the INPUT statement does not include a secondary address, the 4050 controller will issue secondary address 13 by default:

```
INPUT @24:A$
```

sends MTA 88 (24+64) and MSA 109 (13+96) and assigns ASCII to A\$.

To suppress this, use secondary address 32 if no secondary addressing is desired:

```
INPUT @24,32:B
```

sends MTA 88 (24+64) and assigns first legal ASCII number to B.

**Mixed String and Numeric Variables.** The following case poses a potential problem:

```
INPUT @24:A$,B
```

sends MTA 88 (24+64) and assigns all ASCII data before the default-record-separator (CR) to A\$; the next ASCII number is assigned to B. The 4050 controller will hang on this statement if the talker finishes talking without sending a CR in the message. TM 5000 instruments do not send CR as part of their message structure; use Alternate Delimiters in this special case.

**Alternate Delimiters.** The INPUT statement using an @ sign checks for a CR character in the incoming ASCII data. This is the Record Separator character used to assign ASCII data to variables specified in the INPUT statement. If an End-of-File character (decimal 255) is detected, all input stops and the talker is untalked. Alternate delimiter characters can be used for the End-of-File and Record Separator characters. The alternate delimiters apply if the % sign is used in the INPUT statement syntax, rather than the @ sign.

Use a special PRINT statement to designate these characters prior to using the % sign.

```
PRINT @37,Ø: <NUM 1>, <NUM 2>, <NUM 3>
```

Record Separator, End-of-File, Delete Character

## GPIB PROGRAMMING GUIDE

Each number (range 0 to 255) is the decimal equivalent of an ASCII character. NUM 1 is the Record Separator character used to delimit ASCII data assigned to variables. It is discarded and not included in the string it delimits. NUM 2 is the End-of-File (EOF) character. When this EOF character is encountered, all ASCII input ceases and the GPIB device is untalked. This character is discarded and not included in the ASCII data. NUM 3 is a character to be deleted when it is encountered in the incoming ASCII data. If NUM 3 is greater than 127 then no characters are discarded other than the Record Separator and End-of-File characters.

The following example shows how to use an alternate delimiter to break up the "SET?" response and display each message unit on a line by itself.

```
1000 REM DIMENSION A$ FOR 42 CHARACTER MAX LENGTH
1010 DIM A$(42)
1020 REM SEND SET? TO FG5010 AT ADDRESS 24
1030 PRINT @24:"SET?"
1040 REM TALK ADDRESS FG5010 AND RECEIVE FRONT PANEL SETTINGS DATA
1045 REM INTO ONE STRING VARIABLE
1050 REM USE DEFAULT DELIMITERS; RECORD SEPARATOR="CR", EOF=255,
1055 REM DELETE NO CHARACTERS
1060 INPUT @24:A$
1070 PRINT A$
1080 REM SPECIFY ALTERNATE DELIMITERS REC.SEP=";", EOF="4",
1085 REM DELETE CHARACTER=SPACE
1090 PRINT @37,0:59,52,32
1090 REM SEND SET? AND RECEIVE REPLY IN 4 STRING VARIABLES USING
1100 REM ALTERNATE DELIMITERS
1110 PRINT @24:"SET?"
1120 INPUT %24:B$,C$,D$,E$
1130 PRINT B$,C$,D$,E$
```

The display printed by line 1070 would look like this:

```
FREQ 15.0E+6;AMPL 15.0;OFFS 3.4;SYM 50;
```

Note that A\$ is 42 characters long due to the DIM statement line 1010. The balance of the SET? query reply will be input but discarded by the controller.

The display printed by line 1130 would look like this:

```
FREQ15.0E+6
AMPL15.0
OFFS3.
```

followed by an undefined variable error message.

B\$, C\$, and D\$ are delimited by the ";" character and spaces are deleted. Using the ASCII character 4 for EOF caused the statement to terminate input with D\$ (dropping the "4" in "OFFS 3.4"). E\$ was never acquired and is therefore undefined.

TM 5000 instruments remain ready to transmit the rest of the reply to the SET? query sent by the controller in line 1110. Data transmission will resume with ";SYM 50;...".

### Low-Level Output--The WYBTE Statement

#### Purpose

The WYBTE (Write Byte) statement is used to send 8-bit binary data bytes to a device on the GPIB. This statement gives the user complete control over the 8-line data bus and control over the ATN (Attention) and EOI (End or Identify) signal lines.

#### Syntax Form:

```
[ Line number ] WBY [ @ numeric expression [ , numeric expression ] ... : ]
                    [ numeric expression ] [ , numeric expression ] ...
```

#### Descriptive Form:

```
[ Line number ] WBYTE [ @ absolute address [ , absolute address ] ... : ]
                    [ data bytes to be sent out over the General Purpose Interface Bus ]
```

**Explanation**

Data bytes must be decimal integers in the range of -255 to +255. Negative 0 is not allowed. Integers 0 through 255 represent the decimal equivalent of the 8-bit binary data byte. Negative values specify that the EOI line is to be asserted as the data byte is transferred. The (-) sign does not alter the 8-bit data byte and is not transmitted as data. Data bytes specified between the at sign (@) and the colon (:) are sent with the ATN line asserted. These data bytes are GPIB device addresses or interface messages. Data specified after the colon (:) is device dependent data and has meaning only to the device addressed to receive it.

**Sending Interface Messages**

**Universal Commands.** GPIB Universal Command Group (UCG) messages are easily sent with the WBYTE statement. UCG messages defined by IEEE 488 consist of Local Lockout (LLO=17), Device Clear (DCL=20), Parallel Poll Unconfigure (PPU=21), Serial Poll Enable (SPE=24), and Serial Poll Disable (SPD=25).

**Example:**

```
1200 REM SEND LOCAL LOCKOUT (17) INTERFACE MESSAGE
1210 WBYTE @17:
1220 REM SEND DEVICE CLEAR (20) INTERFACE MESSAGE
1230 WBYTE @20:
1240 REM SEND PARALLEL POLL UNCONFIGURE (21) INTERFACE MESSAGE
1250 WBYTE @21:
1260 REM SEND SERIAL POLL ENABLE (24) INTERFACE MESSAGE
1270 WBYTE @24:
1280 REM SEND SERIAL POLL DISABLE (25) INTERFACE MESSAGE
1290 WBYTE @25:
```

**Addressed Commands.** Addressed Commands are interface messages which are sent to devices which have been listen addressed. These include GO TO LOCAL (GTL=1), Selected Device Clear (SDC=4), Parallel Poll Configure (PPC=5), Group Execute Trigger (GET=8), and Take Control (TCT=9).

Example:

```

1300 REM SEND GO TO LOCAL (1) TO PRIMARY DEVICE #18 (MLA=50),
1305 REM THEN SEND UNLISTEN (63)
1310 WBYTE @50,1,63:
1320 REM SEND SELECTED DEVICE CLEAR (4) TO PRIMARY DEVICE #28
1325 REM (MLA=60), THEN SEND UNL (63)
1330 WBYTE @60,4,63:
    
```

### Sending Binary Data to Specified Listeners

Send binary data to devices with the WBYTE statement. Many listeners can be addressed to receive the same data.

Example: Send the device dependent message "ID?;ERR?" to devices at primary addresses 1, 15, 21 and 28.

```

1400 REM SEND LISTEN ADDRESSES WITH ATN ASSERTED
1405 REM SEND BINARY EQUIVALENT OF ASCII DATA "ID?;ERR?",
1406 REM ASSERT EOI LINE WITH LAST CHARACTER
1410 WBYTE @33,47,53,60:73,68,63,59,69,82,82,-63
1420 REM UNLISTEN THE LISTENERS
1430 WBYTE @63:
    
```

See also an expanded program for this purpose under Utility Routines later in this section. The longer program automatically codes the ASCII string for binary transmission.

Low-Level Input--The RBYTE Statement

**Purpose**

The RBYTE (Read Byte) statement receives one or more data bytes from a GPIB device and assigns each data byte to a numeric variable. The controller becomes a listener and handshakes the data byte from a previously specified talker.

**Syntax Form:**

[ Line number ] RBY numeric variable [ , numeric variable ] ...

**Descriptive Form:**

[ Line number ] RBYTE target variable for incoming data byte [ , target variable  
for incoming data byte ] ...

**Explanation**

Data bytes received with EOI asserted will have a negative sign. This negative sign specifies that the talker has sent the last byte of the message.

Example: Talk address device #17 and receive the data into four numeric variables.

```

1500 REM TALK ADDRESS DEVICE #17
1510 WBYTE @17+64:
1520 REM HANDSHAKE 4 BYTES, ASSIGN TO VARIABLES A1, A2, A3, A4
1530 RBYTE A1,A2,A3,A4
1540 REM UNTALK AND UNLISTEN THE BUS
1550 WBYTE @95,63:
1560 REM DISPLAY THE DATA
1570 PRINT A1,A2,A3,A4
    
```



A deadlock can occur if the talker finishes talking before all numeric variables have had values assigned to them. Program execution will hang up on the RBYTE statement while the listener (4050 Controller) waits to handshake a byte which is not forthcoming from the talker. This situation can be avoided if the device asserts EOI with the last byte as Tektronix instruments do. Alternatively, never specify more variables in the RBYTE statement than the talker is expected to send.

## GPIB INTERRUPTS

The 4050 SERIES CONTROLLERS respond to two types of interrupts associated with the GPIB. These are the EOI (End or Identify) interface line and the SRQ (Service Request) interface line on the GPIB. When any GPIB device asserts either of these two lines, an interrupt has occurred which is sensed by the controller. Response to these interrupts is enabled with the ON THEN statement. Response is disabled with the OFF statement or INIT statement.

The action taken when an interrupt condition occurs is specified in the ON...THEN... statement; one such statement must be executed for each interrupt condition. When an ON...THEN... statement is evaluated during program execution, no immediate action is taken; however the ON...THEN... statement arms the controller to respond to the specified interrupt condition. Program execution continues until the specified interrupt occurs; when it does, the controller finishes executing the current statement and then transfers program control to the ON...THEN... statement, which transfers program control to a subroutine.

When subroutine execution is terminated with a RETURN statement, program control is transferred back to the interruption point in the main program. Program execution continues with the statement which would have been executed next if the interrupt hadn't occurred.

## Handling SRQ Interrupts

SRQ interrupts to the 4050 controller must be handled by an ON SRQ THEN... statement in the current program. The subroutine specified by the ON SRQ THEN... statement usually contains a serial poll routine which services several GPIB addresses specified in a list. The serial poll may be implemented with a POLL statement or a low level WBYTE, RBYTE routine, depending on the special needs of the programmer. Once the device which has asserted the SRQ line has been polled, it unasserts the line and the service request interrupt is cleared. Failure to specify a service routine with the ON SRQ THEN... statement will cause 4050 program execution to abort when an SRQ occurs; an error message will be displayed on screen.

The following program establishes an interrupt handler which serial polls a list of bus addresses. TM 5000 instruments, which can assert an SRQ when their front panel INST ID buttons are pushed, are located at

addresses 16, 18, and 20. The WAIT statement is used to cause program execution to pause until an interrupt occurs. A message is printed when the interrupt handler is executed. Instrument power must be turned on prior to running the program.

## NOTE

Allow instruments on the bus to complete their self-test after power-up before running a 4050-Series program (one to five seconds). Otherwise, the controller may detect an illegal condition on the handshake lines while the instruments are performing a self-test, causing the program to be aborted.

```

1000 INIT
1100 ON SRQ THEN 100000
1200 PRINT @16:"USER ON"
1300 PRINT @18: "USER ON"
1400 PRINT @20: "USER ON"
1500 WAIT
1600 GO TO 1500
100000 POLL P,S;16;18;20
100100 GO TO P OF 101000,102000,103000
100200 RETURN
101000 PRINT "SRQ INTERRUPT DEVICE 16, STATUS BYTE ";S
101100 RETURN
102000 PRINT "SRQ INTERRUPT DEVICE 18, STATUS BYTE ";S
102100 RETURN
103000 PRINT "SRQ INTERRUPT DEVICE 20, STATUS BYTE ";S
103100 RETURN

```

## Remarks:

Lines 120 to 140 -- "USER ON" enables the INST ID button interrupt capability in the TM 5000 instruments. See the example above for sending binary data to specified listeners for another way to accomplish this.

Line 150 -- The 4050 waits for an SRQ interrupt. Any interrupt will cause program execution to branch to line 10000 and return to line 160 after a RETURN statement is executed in the subroutine.

Line 10000 -- A serial poll is performed on instruments at addresses 16, 18, and 20. Only one interrupt will be cleared by the POLL statement. This interrupt routine is non-interruptible if entered by means of the ON SRQ THEN 1000 statement at line 110.

### Using ERR?

More detailed status information is available from Tektronix instruments by using the "ERR?" query command. If this query is included in the SRQ routine after the POLL statement, it requests a code to further define the cause of the SRQ. Thus, the status byte identifies the class of event and the "ERR?" response identifies the particular event causing the SRQ.

Section 7 includes a routine to decode the number obtained by "ERR?" and print a message that explains the code. A version of this routine is derived for each instrument in the section for that instrument. Other examples of using "ERR?" occur throughout Sections 3-6.

A word of caution when using the ERR? query in the poll routine: It causes device dependent I/O which is asynchronous to other GPIB I/O in the program. This means that instrument output requested by the program may be lost if an interrupt occurs and an ERR? is received prior to output of the pending data from the instruments output buffer.

This loss occurs if the program sends a query or other output command to an instrument, is interrupted by an SRQ from that instrument, and sends "ERR?" as part of the routine that handles the SRQ. The loss arises not from some special property of "ERR?", but from sending it asynchronous to other I/O--any device-dependent message received by the instrument updates the message buffer, destroying any pending output.

Here's an example of the power-up SRQ causing requested settings data from the PS 5010 to be lost:

```

100 REM ***** ACQUIRE SETTINGS *****
110 INIT
120 ON SRQ THEN 1000
130 DIM R$(300)
140 D=22
150 PRINT @D:"SET?"
160 INPUT @D:R$

```

```
170 PRINT R$
180 END
1000 REM ***** SERIAL POLL ROUTINE *****
1010 POLL D1,S;D
1020 PRINT @D;"ERR?;ID?"
1030 INPUT @D:E$
1040 PRINT "DEVICE # ";D1;"      STB = ";S
1050 PRINT E$
1060 RETURN
```

This program prints only the error and status messages the first time it is run after instrument power-up. The reason is that the ON SRQ statement is not executed until after the PRINT statement is executed at line 150. By the time the program inputs data at line 160, it only receives a null string because program execution jumped to the poll routine, which requested output for "ERR?;ID?" at line 1020 and read it at line 1030.

After the power-up SRQ is serviced, the program produces the expected output.

Use care, also, with "ERR?" in an SRQ routine that responds to the operation complete interrupt (OPC ON). In fact, the example in Section 4 that uses OPC to signal that the DM 5010 data is available does not even incorporate "ERR?" to avoid such a situation. An alternative approach is to test for OPC and jump around the lines that send "ERR?" and input the response.

## Handling EOI Interrupts

EOI interrupt handling in 4050 BASIC is optional. Program execution is not aborted if a response to an EOI interrupt is not enabled with an ON EOI THEN... statement. EOI interrupt handling is useful for low level GPIB input/output and control using WBYTE and RBYTE. Since most bus instruments assert the EOI line with the last data byte sent, the EOI interrupt can serve to notify the controller when this has occurred.

Talkers and listeners other than the 4050 controller can be specified to take part in data transactions. The controller must specify who the talker(s) and listener(s) are and then wait until the talker is finished before resuming use of the bus.

The following program routine illustrates this procedure. The controller specifies a talker at primary address 20 and a listener at primary address 2 and then waits for the EOI. When the EOI interrupt occurs, the controller untalks and unlistens the bus and returns to the next statement after the WAIT statement.

```

1600 ON EOI THEN 1800
1610 WBYTE %63,95,2+32,20+64:
1620 WAIT
1630 OFF EOI
1640 END
.
.
.
1800 WBYTE @63,95:
1810 RETURN

```

## Remarks:

Line 1600 -- Enables the EOI interrupt handler at line 1800.

Line 1610 -- The 4050 controller sends UNTALK and UNLISTEN to clear the bus and then designates device 2 a listener and device 20 a talker. The percent sign (%) tells the 4052 to release control of the bus--until it senses EOI.

Line 1620 -- The 4050 controller waits for the EOI interrupt sent by the talker at the end of the data transaction.

# GPIB PROGRAMMING GUIDE

Line 1630 -- The EOI interrupt handler is disabled with the OFF statement.

Line 1800 -- The 4050 controller sends UNTALK and UNLISTEN to clear the bus of talkers and listeners before resuming program execution.

4050 INTERRUPT-HANDLING STATEMENTS

The OFF Statement

**Purpose**

The OFF statement disables the current program from responding to the specified interrupt. If no interrupt condition is specified as a parameter in the OFF statement, then the program response to all interrupt conditions is disabled.

**Syntax Form:**

[ Line number ] OFF [ { EOF ( numeric constant )  
EOI  
SIZE  
SRQ } ]

**Descriptive Form:**

[ Line number ] OFF [ interrupt condition ]

**Explanation**

If an OFF SRQ statement or INIT statement is executed, a fatal error will result when an SRQ interrupt occurs before another ON SRQ THEN... statement is re-executed. If an OFF EOI statement or INIT statement is executed, program control continues un-interrupted when an EOI interrupt occurs.



The ON...THEN... Statement

**Purpose**

The ON...THEN... statement transfers program control to the specified subroutine line number in response to the specified interrupt.

**Syntax Form:**

```

Line number  ON  { EOF ( numeric constant )
                  EOI
                  SIZE
                  SRQ }  THEN  line number
    
```

**Descriptive Form:**

```

Line number  ON  interrupt condition  THEN  line number
    
```

**Explanation**

Execution of the subroutine RETURN statement transfers program control back to the main program--to the next line following the line number being executed when the interrupt occurred. The ON...THEN... statement may be executed at any time before the interrupt condition occurs. The subroutine specified in the ON...THEN... statement is non-interruptible.

## The POLL Statement

## Purpose

The POLL statement executes a serial poll of all specified GPIB devices to determine which device is requesting service.

## Syntax Form:

```
[ Line number ] POL  numeric variable , numeric variable ; primary address [ , secondary
                        address ] [ ; primary address [ , secondary address ] ] ...
```

## Descriptive Form:

```
[ Line number ] POLL target variable for device identifier , target variable for return
                        status information ; address list
```

## Explanation

The POLL statement obtains status bytes from devices in its list of primary and secondary addresses. Each status byte is checked when it is received to determine if bit 7 is set (logic one). This indicates that the device which sent the status byte has asserted the service request line. The decimal value of this status byte is assigned to the second variable in the POLL statement. The device address position in the poll list is assigned to the first variable specified in the POLL statement. The device requesting service releases the SRQ line when it has been serial polled.

Example: Devices 18 and 9 have requested service. Device 18 reports status byte 0100 0001 (decimal 65).

```
1900 REM POLL ADDRESSES 21, 16, 18, AND 9
```

```
1910 POLL A,B;21;16;18;9
```

```
1920 PRINT A,B
```

```
1930 RETURN
```

In the example, A would be set to 3 and B to 65. Status bytes were obtained from devices 21 and 16 but were discarded because they had not asserted SRQ and bit 7 was not set. Device 9 was not polled because device 18 was polled first, reporting a status byte which indicated that it had asserted the SRQ line. In the event that device 9 had also asserted SRQ, it would not be polled until the POLL statement was executed again.

The POLL statement is typically used in a service routine specified by the ON SRQ THEN... statement. This service routine is non-interruptible and must be exited with a RETURN statement before the controller can respond to new or existing SRQ interrupts. If two or more devices have asserted the SRQ line, they must each be serviced by re-entering the service routine. This would happen each time program control was passed back to the main program.

Example: Devices 16 and 9 have asserted SRQ during execution of the main program.

```

100 INIT
110 ON SRQ THEN 2000
.
.
.
    [MAIN PROGRAM]
2000 REM SERVICE REQUEST ROUTINE
2010 POLL A,B;21;16;18;9
2020 GO TO A OF 2100,2200,2300,2400
2100 REM SERVICE ROUTINE FOR DEVICE 21
.
.
.
2190 RETURN
2200 REM SERVICE ROUTINE FOR DEVICE 16
.
.
.
2290 RETURN
2300 REM SERVICE ROUTINE FOR DEVICE 18
.
.
.
    
```

```

2390 RETURN
2400 REM SERVICE ROUTINE FOR DEVICE 9
.
.
.
2490 RETURN
    
```

Device 16 would be serviced first and main program execution would resume. Device 9 is still asserting the SRQ line.

A few main program lines will be executed before the controller responds to the SRQ asserted by device 9.

**Unspecified Devices on the Bus.** A value of zero (0) is assigned to both variables specified in the POLL statement if none of the devices listed have asserted the SRQ line. This may happen if a device is attached to the bus and its address is not included in the POLL statement. Since an interrupt from this device is never serviced, the device never relinquishes the SRQ line. The main program will be continually re-interrupted as the servicing routine polls all addresses in its list in the effort to clear the interrupt.

**Specified Devices Not on the Bus.** All devices specified in POLL list must be powered up and attached to the bus when the POLL statement is executed, otherwise program execution will pause while the POLL statement waits for a status byte from a non-existent talker. Program execution will not continue until the situation is corrected.

## The WAIT Statement

**Purpose**

The WAIT statement is a convenient means of halting program execution until an interrupt occurs.

**Syntax Form:**

[ Line number ] WAIT

**Descriptive Form:**

[ Line number ] WAIT

**Explanation**

An interrupt which occurs while the WAIT statement is executing will invoke the interrupt handler enabled by a previously executed ON...THEN... statement. A RETURN from that subroutine will cause program execution to continue on the next program line following the WAIT statement.

## The WAIT Routine

**Purpose**

The WAIT routine halts program execution for a specified number of seconds, or until an interrupt condition occurs, whichever comes first.

This routine is not implemented in the 4051.

**Syntax Form:**

```
[Line number] CALL { "WAIT"
                    { string variable } } [numeric variable]
```

**Descriptive Form:**

```
[Line number] CALL routine name [number of seconds]
```

**Explanation**

The WAIT routine produces a pause in program execution. If an interrupt occurs while a WAIT routine is being processed, the interrupt is handled immediately and no further waiting is done.

If the numeric expression is negative, a zero is assumed and the system does not pause.

If no pause interval is specified, the routine operates like the WAIT statement.

Use the WAIT routine to allow peripheral GPIB devices to settle or process information before proceeding with the 4050 program.

UTILITY ROUTINES

Get A Status Byte with WBYTE and RBYTE

This routine obtains a status byte from the instrument at primary address "A". No checking is done for the SRQ bits in the status byte.

```

4000 WBYTE @63,24,A+64:
4010 RBYTE S
4020 WBYTE @25,95:
4030 RETURN
    
```

Line 4000 -- WBYTE is used to send interface commands UNL (63), SPE (24), and MTA (primary address + 64).

Line 4010 -- RBYTE is used to handshake the status byte from the "talked" instrument. The received 8-bit byte is placed in variable S as a decimal number.

Line 4020 -- WBYTE is again used to send interface commands SPD (25) and UNT (95) in order to terminate the serial poll sequence.

Line 4030 -- RETURN from this subroutine.

Refer to Section 6.5.2 Serial Poll of the IEEE-488 standard for more detailed information on serial poll.

## Convert a Status Byte to an 8-Bit Array

This routine uses the status byte S obtained from a serial poll in decimal form and converts it into a digital word in an 8 bit numeric array. Each element in the array is the corresponding bit in the 8 bit digital number. This provides an easy means of testing the value of individual bits.

```

6000 REM CONVERT STATUS BYTE TO 8 BIT ARRAY
6010 DIM B(8)
6020 FOR D=8 TO 1 STEP -1
6030 B(D)=INT(S/2^(D-1))
6040 IF NOT(B(D)) THEN 6060
6050 S=S-2^(D-1)
6060 IF D<>4 THEN 6080
6070 S1=S
6080 PRINT B(D);
6090 NEXT D
6100 PRINT
6110 PRINT S1
6120 RETURN

```

Line 6010 -- Dimensions eight elements in numeric array B.

Lines 6020-6090 -- Defines each element of B and prints it left to right, most significant bit first. S1 is the decimal value of the low order 4 bits. Use this for device dependent or system status messages.

## Example:

```

S=165
RUN 6000

```

```

RESULT
10100101
5

```



Send Identical Messages to Three GPIB Addresses Simultaneously

This routine defines an ASCII string, converts it to binary equivalents and then transmits it to three listeners simultaneously; it uses the WBYTE statement for the low level I/O.

```

3300 A$="ID?;SET?"
3310 A=LEN(A$)
3320 DIM B(A)
3330 FOR I=1 TO A
3340 C$=SEG(A$,I,1)
3350 B(I)=ASC(C$)
3360 NEXT I
3370 B(I-1)=-B(I-1)
3380 WBYTE @63,52,54,58:B
3390 WBYTE @63:
3400 END
    
```

Lines 3310 to 3360 -- Convert A\$ to a numeric array B.

Line 3370 -- Negate the last element in the array so that the EOI line will be asserted when that last byte is sent.

Line 3380 -- Listen address devices 20, 22, and 26 and send array B.

Line 3390 -- Send UNL to clear the bus of listeners.

This program may be used as a subroutine. Delete B and define A\$ before calling the routine at line 3310. Also change 3400 to RETURN.

4052/GPIB SEND AND RECEIVE

This program uses user definable keys to control communications with three instruments on the bus. ASCII data (device dependent messages) can be sent to the instruments and received back using the standard 4052 I/O delimiters or special delimiters of your choosing. Certain interface control messages may also be sent.

This program provides an illustration of GPIB I/O and interrupt handling. It is written in subroutine modules called by the user definable keys. Different device addresses could be used by changing lines 150, 300, 960, 990, 1010, and 1030 to include the appropriate addresses. Two sample hard copies are shown in Fig. 2-1 and Fig. 2-2.

GPIB SEND/RECEIVE with STATUS BYTE REPORTING

TURN ON POWER TO GPIB DEVICES 18,20, and 24.

Press the 'RETURN' key to continue . . .

STATUS BYTE = 65    DEVICE 18

STATUS BYTE = 65    DEVICE 20

STATUS BYTE = 65    DEVICE 24

SELECT ADDRESS

VALID ADDRESSES = 18,20, or 24.    ENTER ONE ADDRESS : 20

Fig. 2-1. Beginning display of 4052/GPIB SEND and RECEIVE program.

## PRESS USER DEFINABLE KEYS

- #1 SEND ASCII DATA
- #2 RECEIVE ASCII DATA (with DEFAULT DELIMITERS)
- #3 SELECT ADDRESS
- #4 SELECT ALTERNATE DELIMITERS
- #5 SEND INTERFACE MESSAGES
- #7 RECEIVE ASCII DATA (with ALTERNATE DELIMITERS)

Fig. 2-2. Display during 4052/GPIB SEND and RECEIVE program.

```

1 REM ----- {4052/GPIB SEND & RECEIVE (Alt. Delimiters)} -----
2 GO TO 100
4 REM ----- SEND ROUTINE - USER KEY 1 -----
5 GOSUB 380
6 RETURN
8 REM ----- RECEIVE ROUTINE - USER KEY 2 -----
9 INPUT @A,32:R$
10 GOSUB 500
11 RETURN
12 REM ----- SELECT ADDRESS ROUTINE - USER KEY 3 -----
13 GOSUB 260
14 RETURN
16 REM ----- SELECT ALTERNATE DELIMITERS ROUTINE - USER KEY 4 -----
17 GOSUB 660
18 RETURN
20 REM ----- SEND INTERFACE MESSAGE - USER KEY 5 -----
21 GOSUB 780
22 RETURN
28 REM ----- RECEIVE with ALTERNATE DELIMITERS ROUTINE - USER KEY 7 ---
29 INPUT %A,32:R$
30 GOSUB 500
31 RETURN
100 INIT
110 ON SRQ THEN 930
120 DIM R$(10000)
130 PAGE
140 PRINT "IGPIB SEND/RECEIVE with STATUS BYTE REPORTING"
150 PRINT "JJGTURN ON POWER TO GPIB DEVICES 18,20, and 24."
160 PRINT "JPress the `RETURN' key to continue . . .";
170 INPUT S$
180 ON SRQ THEN 960

```

```

190 REM ----- GET AN ADDRESS with ADDRESS INPUT ROUTINE -----
200 GOSUB 280
210 SET KEY
220 REM ----- USER KEY-PRESS LOOP -----
230 GO TO 210
240 REM
250 REM
260 PRINT "LJ"
270 REM ----- SELECT ADDRESS ROUTINE -----
280 SET NOKEY
290 PRINT "JSELECT ADDRESS"
300 PRINT "JVALID ADDRESSES = 18,20, or 24. ";
310 PRINT "  ENTER ONE ADDRESS : ";
320 INPUT A
330 PAGE
340 GOSUB 550
350 RETURN
360 REM
370 REM ----- SEND ASCII DATA ROUTINE -----
380 PAGE
390 SET NOKEY
400 PRINT "JEnter ASCII Data to SEND to Device #";A
410 PRINT
420 INPUT S$
430 PRINT @A,32:S$
440 CALL "WAIT",0.5
450 GOSUB 550
460 RETURN
470 REM
480 REM ----- PRINT ASCII DATA RECEIVED ROUTINE -----
490 REM
500 PRINT "LRECEIVED ASCII Data from Device #";A;"JJG"
510 PRINT R$
520 GOSUB 550
530 RETURN
540 REM ----- PRINT USER KEY MENU ROUTINE -----
550 PRINT "JJJIPRESS USER DEFINABLE KEYS"
560 PRINT "JJ#1 SEND ASCII DATA"
570 PRINT "J#2 RECEIVE ASCII DATA (with DEFAULT DELIMITERS)"
580 PRINT "J#3 SELECT ADDRESS"
590 PRINT "J#4 SELECT ALTERNATE DELIMITERS"
600 PRINT "J#5 SEND INTERFACE MESSAGES"
610 PRINT "J#7 RECEIVE ASCII DATA (with ALTERNATE DELIMITERS)"
620 RETURN
630 REM

```

```

640 REM ----- SELECT ALTERNATE DELIMITERS  ROUTINE -----
650 REM
660 SET NOKEY
670 PRINT "LISELECT ALTERNATE DELIMITERS"
680 PRINT "JJENTER RECORD-SEPARATOR CHARACTER CODE (0-255) - ";
690 INPUT N1
700 PRINT "JENTER END-OF-FILE CHARACTER CODE (0-255) - ";
710 INPUT N2
720 PRINT "JENTER DELETE CHARACTER CODE (0-255) - ";
730 INPUT N3
740 REM ----- ASSIGN ALTERNATE DELIMITERS -----
750 PRINT @37,0:N1,N2,N3
760 GOSUB 550
770 RETURN
780 REM ----- SEND INTERFACE MESSAGES -----
790 SET NOKEY
800 PRINT "LISEND INTERFACE MESSAGES"
810 PRINT "JINTERFACE MESSAGES"
820 PRINT "J1 GO TO LOCAL"
830 PRINT "4 SELECTED DEVICE CLEAR"
840 PRINT "8 GROUP EXECUTE TRIGGER"
850 PRINT "17 LOCAL LOCKOUT"
860 PRINT "20 DEVICE CLEAR"
870 PRINT "JENTER MESSAGE CODE TO SEND : ";
880 INPUT N
890 WBYTE @63,A+32,N,63,95:
900 GOSUB 550
910 RETURN
920 REM ----- WAIT FOR INSTRUMENT POWER-ON DIAGNOSTICS TO COMPLETE ---
930 CALL "WAIT",5
940 REM
950 REM --- SRQ INTERRUPT SERVICING ROUTINE / DISPLAY STATUS BYTE ---
960 POLL L,B;18;20;24
970 GO TO L OF 990,1010,1030
980 RETURN
990 PRINT "JJISTATUS BYTE = ";B;" DEVICE 18"
1000 RETURN
1010 PRINT "JJISTATUS BYTE = ";B;" DEVICE 20"
1020 RETURN
1030 PRINT "JJISTATUS BYTE = ";B;" DEVICE 24"
1040 RETURN

```

## SECTION 3

## DC 5010 AND DC 5009 UNIVERSAL COUNTER/TIMERS

The DC 5009 and DC 5010 implement a microprocessor-based ratio architecture to make programmable universal counter/timer measurements. The reciprocal technique allows high resolution of low-frequency signals much faster than conventional counting techniques. Averaging in all modes and identical A and B channels enhance accuracy.

Programming ease is enhanced by a high-level command set that provides full control and reporting of front-panel functions, interrupts, and measurement acquisition and output.

## DC 5009 and DC 5010 Functions

The following universal counter/timer functions may be selected from the front panel or by commands over the GPIB:

**Frequency A**

Measures Channel A signal period, then calculates and displays its frequency.

Range = 36 microhertz to 350 megahertz for DC 5010,  
100 microhertz to 135 megahertz for DC 5009.

**Period A**

Measures and displays Channel A signal period.

Range = 3.125 nanoseconds to 7.6 hours for DC 5010,  
7.4 nanoseconds to 3.05 hours for DC 5009.

**Width A**

Measures pulse width of Channel A signal.

Easier accurate setup--one channel, one probe, one trigger level.

Polarity (+ or -) selected by SLOPE control.

Range = 4 nanoseconds to 7.6 hours for DC 5010,  
15 nanoseconds to 3.05 hours for DC 5009.

**Time A to B**

Measures time from first event on A to first succeeding event on B.  
 Range = 2 nanoseconds to 7.6 hours for DC 5010,  
 15 nanoseconds to 3.05 hours for DC 5009.

**Rise/Fall A**

DC 5010 automatically measures signal amplitude, calculates and sets trigger levels at 10 and 90 percent, and measures time between. DC 5009 can also make this measurement under program control as shown in an example later in this section.

**Probe Compensation**

Display guides user in compensating high-impedance probe.

**Null**

DC 5010 saves the current measurement and subtracts it from subsequent measurements.  
 In time interval mode, removes time differential mismatch between A and B and external probes or cabling.  
 May be implemented by program using DC 5009.

**Totalize A**

Channel A events are counted.  
 DC 5010 also counts A-B and A+B events.

**Events B During A**

Measures number of pulses on B while gated by signal on A as set by A trigger slope and level controls.  
 Range  $\leq$  350 megahertz (B) and  $\geq$  4 nanoseconds (A) for DC 5010,  
 $\leq$  125 megahertz (B) and  $\geq$  15 nanoseconds (A) for DC 5009.

**Ratio B/A**

Measures ratio of B events to A events over same time interval.  
 Range = DC to 350 megahertz both A and B for DC 5010,  
 DC to 135 megahertz (A) and DC to 125 megahertz (B) for DC 5009.

**Measurement Control**

START and STOP commands allow full control of measurements.  
 RESET command clears counter, then causes a new measurement;  
 used with STOP for single measurement.

**Averaging**

Up to 1.0E+9 for DC 5010 or 1.0E+8 for DC 5009.  
 Allows useable resolution to 1 picoseconds for DC 5010, 5  
 picoseconds for DC 5009.  
 Auto-averages for convenient 0.3 second measurement time.  
 Pseudo-random, phase-modulated time base eliminates clock-  
 synchronous errors in time interval, width, and rise/fall.

**Auto-Trigger**

Selects optimum trigger points.  
 Read out trigger levels over GPIB.  
 Program other trigger levels over GPIB.  
 Minimum and maximum peak voltage values are also available over  
 the bus using "MIN?" and "MAX?".

**Auto-Resolution**

No "false" digits--extraneous resolution from large number  
 of averages--reported.



## Input Signal Conditioning

Both channels fully programmable (coupling, attenuation, slope, etc.).

## GPIB Operation

### IEEE 488 Bus Address

Pressing the INST ID button will display the counter's primary address. A decimal following the address indicates that the message terminator is set to EOI/LF (the counter responds to either EOI or LF when listening and sends LF with EOI asserted when talking on the bus).

For convenience, the counters are shipped with the address switch set to primary address 18 for the DC 5009 and 20 for the DC 5010; the terminator to EOI only. These are the addresses used in example listings in this section. Secondary address capability is not used by the counters.

Switches for all of these settings are internal; refer to a qualified service person for a change. If the switches are changed, use an address between 1 and 30 and EOI only to operate with Tektronix controllers.

### Power-Up

At power-up, the counters perform a self-test of ROM and RAM and the count chains. If the test fails, the counter displays an error number. If the test succeeds, a counter goes to local control with power-up default settings, asserts SRQ, and prepares to report power-up status. Although the counters respond normally to GPIB messages whether or not the GPIB controller reads the power-up status, a 4050 BASIC program aborts if it does not include the ON SRQ THEN... statement and an SRQ handler (see Section 2 for more on this.)

TABLE 3-1  
POWER-UP DEFAULT SETTINGS

Front Panel (applies to DC 5010 only)

FREQ	Frequency mode
AVE	-1 (auto average)
AUTO	A

Channel A&B input signal conditioning:

COU	DC
ATT	1
SLO	POS
SOURCE	EXT

CHA	A (channel pointer)
TERM	OFF
PRE	OFF
FILTER	OFF
NULL	OFF

Interrupts

RQS	ON
OPC	OFF
OVER	OFF
USER	OFF

Device Trigger •

DT	OFF
----	-----

DC 5009 goes to settings of front-panel controls at power-on with channel pointer at A and prescaler off.

### Making GPIB Counter Measurements

Program examples throughout this section apply to both the DC 5009 and DC 5010 unless noted as exceptions. Factory-set addresses of 18 (DC 5009) and 20 (DC 5010) are used in the examples.

#### Setting Up the Measurement

Setting up counter measurements under remote control is much like setting up measurements under local control. With Tektronix GPIB counters, the DC 5009 and DC 5010, programming commands are the same as front panel labels in almost all cases. (The entire command set is included in the Appendix.)

**Front-Panel Controls.** Setting counter front panel controls is just a matter of inserting easy-to-recognize instrument commands in the PRINT statement of a Tektronix controller, all in a high-level format.

For example, commands that select the DC 5009 measurement function, automatic averaging, and signal conditioning can be combined as:

```
PRINT @18: "FREQ A;CHAN A;ATT 1;COU DC;SLO POS;AUTO;AVE-1"
```

With the DC 5010, this particular setup could be achieved by initializing the instrument to the power-up state.

```
PRINT @20: "INIT"
```

#### NOTE

"INIT" returns DC 5009 front-panel functions to the control settings rather than a predefined state. As an alternative, build a string variable of desired settings and send it instead of "INIT".

In either case, the DC 5009 and DC 5010 would light the GATE light and continue taking frequency readings as long as the input signal is within the sensitivity range of the counter. The GATE light blinks between readings. If the signal is not within range, the counter continues to display the last reading and leaves the GATE light on. The counter resets the display and begins a new measurement when any control is changed that would affect a valid reading. Examples are a change in function (frequency, period, etc.), trigger level, or signal conditioning.

**Input Conditioning.** When changing input conditioning (X1, X5, etc.), it is of course necessary to specify which of the two channels one wishes to change. Rather than require that the expression "CHA A" or "CHA B" precede each input conditioning command, The DC 5010 and DC 5009 instead use the "CHA A" etc. commands as a pointer. This pointer, once set, remains set and insures that each input command that follows will refer to that channel only. Since Channel A is used for all the single channel functions, both the DC 5009 and DC 5010 power up with the pointer set to CHA A. The "INIT" command also sets the pointer to the A channel.

**Rise/Fall.** In the rise/fall mode, the DC 5010 measures input signal peak (100%) and valley (0%) voltages, and uses that information to calculate and set the 10% and 90% trigger levels. For risetime, the instrument is configured to take a time A->B measurement, Channel A and B are set to + slope, Channel A level is set to the 10% point and Channel B is set to the 90% point.

Relays route the signal coming in the A BNC connector to both Channel A and Channel B through a power splitter. When rise/fall is first selected (and before the autotrigger gets the 0% and 100% points), all of the input settings (AC/DC, etc.) of Channel A are automatically duplicated down onto the B channel. From the front panel, rise time and fall time are selected by first setting the Channel A slope to + (rise) or - (fall) before pushing RISE/FALL; over the bus, use either the "RISE" or "FALL" command (slope selection is not required). Should there be any significant change in signal voltage levels, then request "RISE" or "FALL" again, so that the new 10% and 90% levels can be measured and calculated.

In the rise/fall mode, the DC 5010 routes the signal arriving at the Channel A BNC connector into both the Channel A and B input conditioning and amplifiers. This is done automatically with an internal 50-ohm power splitter. The power splitter maintains a uniform 50-ohm environment, but it also divides the voltage of the signal by a factor of two. Since the LEV, MIN and MAX numbers all refer to the signal actually entering the input conditioning area, the reported voltages will be smaller by a factor of two than what is actually delivered to the front panel BNC. This is of interest only when using these numbers as absolute voltages. It is not a consideration if, for example, one wishes to set 20% and 80% trip points for rise/fall instead of the instruments 10% and 90% points. In that case, merely read MAX and MIN, calculate the 20% and 80% levels and program them as though there were no factor of two difference, since you are dealing only with percentages and not absolute voltages.

To summarize, this factor of two decrease in reported voltage occurs only in RISE/FALL with 50-ohm termination, and then only affects the reporting of the absolute magnitude of the voltage.

When the counters report trigger level voltage, be aware of several other considerations.

1. The DC 5009 and DC 5010 both correct for their attenuating settings. For example, if the user program is in X1 (no attenuation) and sets a trigger level of 1.3 volts and then at some later time selects X5 attenuation, the instrument, if queried by "LEV?" will respond that the effective trigger point is now at 6.5 volts (1.3 X 5). This is true over the bus and from the front panel except in the one situation where the LEV and the ATTEN setting are made in the same message. In that case, the final trigger level at the input connector will be the voltage specified by the LEV command independent of the ATTEN setting.
2. One feature that these counters do **not** have is a "probe identify" ring such as on some Tektronix oscilloscopes. Thus the counter doesn't sense when an X5 or X10 probe is attached to the front of the counter. This would only affect absolute voltages read out with the LEV?, MIN? and MAX? commands.
3. In the one case of rise/fall at 1 megohm, the power splitter does not divide the signal in half, but it does change the input impedance from 1 megohm, approximately 22 picofarads, to about 500 kilohms and 50 picofarads. (This leaves the probe compensation, which depends on R\*C, essentially unchanged.) With rise/fall at 1 megohm with attenuating probes, however, the situation is less obvious. For example, an X10 probe has a 9 megohm impedance in the probe, which with the 1 megohm input impedance, leads to a 10 to 1 division ratio as marked on the probe. When this X10 probe instead works into the 500 kilohm input impedance of the DC 5010 in rise/fall mode, the division ratio is  $(9 \text{ megohm} + 500 \text{ kilohm}) / 500 \text{ kilohm}$  or 19X. Similarly, an X5 probe will effectively have an X9 attenuation factor.

### Getting a Reading

The 4050 INPUT statement obtains a reading from the counter output buffer if there is one available. If there is no reading available, the counter returns a byte with all bits set to one (decimal 255), asserting EOI concurrently. If input as a numeric variable, as

```
T=0
INPUT @18: T
```

the variable is not updated (T would still equal zero). If input as a string:

```
INPUT @18: T$
```

the string is set to a NULL (empty) string.

Use the SEND command to guarantee that the counter gets and reports a reading, no matter how long the 4050 has to wait on INPUT:

```
PRINT @18: "SEND"
INPUT @18: R
```

Some measurements take significant time to complete, such as the period of a low repetition-rate signal or the average of a large number of readings. The program can handle other tasks in the meantime, relying on the operation complete SRQ to signal that the counter measurement is available:

```
10 REMARK OPC INTERRUPT-DRIVEN MEASUREMENT
100 ON SRQ THEN 190
110 B=0
120 PRINT @18:"PER;OPC ON"
130 WAIT
140 IF B=0 THEN 130
150 PRINT @18:"OPC OFF"
160 PRINT "PERIOD = ";B
170 END
180 REMARK POLL AND GET READING ON OPC
190 POLL D,S;18
200 IF S=66 OR S=82 THEN 230
210 PRINT "COUNTER REPORTS STATUS ";S
220 GO TO 230
230 INPUT @18:B
240 RETURN
```

Replace the WAIT with your program.

A technique incorporating the "RDY?" command may be used instead of the interrupt-driven input above:

```

10 REMARK GET MEASUREMENT ON "RDY 1"
100 ON SRQ THEN 190
110 PRINT @20:"INIT;PERIOD"
120 R=0
130 PRINT @20:"RDY?"
140 INPUT @20:R
150 IF NOT(R) THEN 130
160 INPUT @20:B
170 PRINT "PERIOD = ";B
180 END
190 POLL D,S;20
200 PRINT "COUNTER REPORTS STATUS ";S
210 RETURN

```

This technique can be adapted to allow the controller to perform tasks uninterrupted by an SRQ, "polling" the counter using "RDY?" to determine if a measurement is completed.

#### Time Interval and Width Measurements

Programmable trigger level controls and high resolution allow the counters to resolve narrow time intervals and pulse widths.

In this example, the DC 5010 returns the time relationship between two TTL signals. (An example later in this section extends this capability to make phase measurements.)

```

10 REMARK TIM EXAMPLE
100 ON SRQ THEN 380
300 PRINT @20:"CHA A;SLO POS"
310 PRINT @20:"ATT 1;COU DC;LEV .275"
320 PRINT @20:"CHA B;SLO POS"
330 PRINT @20:"ATT 1;COU DC;LEV .275"
340 PRINT @20:"AVE 1;TIME;SEND"
350 INPUT @20:T
360 PRINT "TIME A TO B = ";T
370 END
380 POLL D,S;20
390 PRINT "STATUS = ";S
400 RETURN

```

Lines 300 to 310 -- Set up Channel A. The 0.275 volt trigger level assumes X5 probes for trigger level at the probe tip of 1.375 volts (TTL threshold).

Lines 320 to 330 -- Duplicate setup on Channel B.

Lines 340 to 360 -- Select time interval function, request a reading, then input and print result.

If the time interval was a single pulse, the slope could be reversed in line 320 ("SLO NEG" instead of "SLO POS"). An alternative is to select the pulse width function:

```

10 REMARK PULSE WIDTH MEASUREMENT
300 PRINT @20:"INIT;WIDTH;LEV .275"
340 PRINT @20:"SEND"
350 INPUT @20:T
360 PRINT "PULSE WIDTH = ";T
370 END
    
```

Again, as with Time A to B, choose the polarity of the pulse to be measured with "SLO POS" or "SLO NEG." Only one LEV command is required to set up the measurement.

### Triggering An Updated Reading

The DC 5010 and DC 5009 may be set for continuous updating and reporting of new measurements. However, you may wish to control the counters directly through your program to get and output an updated reading. This can be done in several ways.

An updated value can be found by:

1. Any change in a function that would affect the validity of a measurement causes the counter to discard and not report a previous reading. This applies to a change in function, or signal conditioning.
2. The "INIT" command also causes an old reading to be discarded and initiates a new one.



3. The command sequence "STOP;RESET" sets the counter to stopped mode, resets the count chains, and initiates a new reading.

4. The command "STOP" followed by the GET interface message is equivalent to case 3.

Here is an example of a single time A to B measurement.

```
10 REMARK GET NEW MEASUREMENT
300 PRINT @18:"AVE 1;TIME"
310 PRINT @18:"STOP;RESET;SEND"
320 INPUT @18:R
330 PRINT "TIME INTERVAL = ";R
340 END
```

Line 300 -- Request one time interval reading (no averaging).

Line 310 -- Initiate new reading; ask for output.

Lines 320 to 330 -- Get and print reading.

The next example uses GET (Group Execute Trigger) interface message.

```
1 REMARK TRIGGER WITH <GET>
2 GO TO 310
4 GOSUB 340
5 RETURN
300 REMARK SET UP MEASUREMENT
310 PRINT @18:"TIME;STOP;AVE 1;DT TRIG"
320 END
330 REMARK TAKE READING
340 WBYTE @50,8,63:
350 PRINT @18:"SEND"
360 INPUT @18:R
370 PRINT "TIME INTERVAL = ";R
380 RETURN
```

Lines 2 to 5 -- Call measurement subroutine when UDK #1 is pressed.

Lines 310 to 320 -- Set up single updated measurement and end (wait for UDK).

Lines 340 to 370 -- Trigger measurement, then get reading and print it. In line 340, 50=MLA (18+32), 8=GET, and 63=unlisten. The SEND command in line 350 is recommended to handle a longer duration pulse. Without "SEND", the counter may time out and report <NULL>.

If you adapt this technique, use the command sequence in line 310, changing TIME to another function and number of averages as desired. It may be necessary to allow some time for the instrument to respond to DT TRIG before sending the first GET; this program does because it requires operator action to send GET.

### Amplitude Measurements

Either counter performs as a peak-reading voltmeter when executing the AUTO trigger command. A simple example that uses this function is:

```

10 REM CHECK PULSE AMPLITUDE
200 PRINT @20:"CHA A;FREQ;ATT 5;AUTO;MAX?;MIN?"
210 INPUT @20:M1,M2
220 IF M1>3 AND M2<0.7 THEN 240
230 GO TO 200
240 PRINT "PULSE AMPLITUDE PASSES: LO < .7, HI > 3.0"
250 END

```

The test in line 220 could be expanded to check that both the high and low levels are within some bounds (not just greater than 3.0 and less than 0.7).

Because the probe compensation function also detects peak voltages, it can be used in place of AUTO. This restricts the low-frequency cut-off (works down to 100 rather than 20 hertz), but works faster (less than 0.5 second rather than 2.0 seconds, worst case). Just change line 200 to:

```
200 PRINT @20: "CHA A;ATT 5;PROBE;FREQ;MAX?;MIN?"
```

For the DC 5009 only, add line 195:

```
195 PRINT @18: "CHA A;LEV 0"
```

### Application Programs

Some program examples are shown here that use the DC 5009 and DC 5010. Address 18 indicates the program was developed using the DC 5009 and address 20 indicates the DC 5010 was used (these are the factory-set addresses). Generally, the programs may be used with either counter by changing the address in the program or the address of the counter. There are some exceptions, however, and these are noted.

### Signal Characterization

This program illustrates the use of the DC 5009 in the measurement of such signal characteristics as maximum and minimum peak values, frequency, duty cycle, rise time, and slew rate. The program could, with small changes, use the DC 5010 rather than the DC 5009.

These following formulas are used in the program:

$$\text{Duty cycle} = \text{width/period}$$

$$\text{Slew rate} = (90\% \text{ voltage level} - 10\% \text{ voltage level})/\text{rise time}$$

Rise time is measured by connecting the signal of interest to both the Channel A input and the Channel B input using either a T connector or a power splitter. A splitter is preferred if the signal source must be terminated at 50 ohms. An auto trigger is then performed on both channels. A maximum and minimum query is performed on each channel and then the 10% level is calculated using the Channel A minimum and maximum values. The 90% level is calculated by using the Channel B minimum and maximum values. Channel A trigger level is then programmed to the 10% point and Channel B to the 90% point. Rise time is then obtained by measuring time A to B.

**SIGNAL CHARACTERIZATION**

```

P-P AMPLITUDE      = 5.575 VOLTS
ATTEN = 1
COUPLING = DC
MIN PEAK = -2.95
MAX PEAK = 2.625

FREQUENCY          = 10221.615 KHz
DUTY CYCLE         = 48.5%
RISETIME           = 0.02397 uS
SLEW RATE          = 186.065915 V/us
    
```

**PRESS COUNTER INST ID TO TAKE A MEASUREMENT  
AND CHARACTERIZE THE SIGNAL.**

Fig. 3-1. Example 4052 display of signal characterization program.

```

100 INIT
110 ON SRQ THEN 740
120 PAGE
130 PRINT "      JJJJJ   PRESS COUNTER INST ID TO TAKE A MEASUREMENT"
140 PRINT "      AND CHARACTERIZE THE SIGNAL."
150 R1=0
160 C$="DC;"
170 PRINT @18:"INIT;USER ON;CHA A;ATT 5;CHA B;ATT 5"
180 WAIT
190 IF E<>403 THEN 180
200 PAGE
210 REM ----- FREQUENCY MEASUREMENT -----
220 PRINT @18:"CHA A;COU ";C$;"CHA B;COU ";C$;"AUTO;FREQ A;SEND"
230 INPUT @18:F
240 REM ----- PULSE WIDTH/DUTY CYCLE MEASUREMENT -----
250 PRINT @18:"WID;SEND;PER;SEND"
260 INPUT @18:W,P
270 REM ----- RISE/FALL - SLEW RATE MEASUREMENT -----
280 PRINT @18:"MIN?;MAX?;CHA B;MIN?;MAX?"
290 INPUT @18:A0,A9,B0,B9
300 IF A0>-3.175 AND A9<3.2 THEN 480
310 IF A0=-3.175*5 OR A9=3.2*5 THEN 330
320 GO TO 510
330 IF C$="AC;" THEN 360
340 C$="AC;"
350 GO TO 220
360 PAGE
370 PRINT "J"
380 FOR I=1 TO 5
390 PRINT "      GGGG V E R V O L T A G E K"
400 NEXT I
    
```

```

410 PRINT "JJJTHE P-P VALUE OF THE SIGNAL AND/OR ITS DC OFFSET ";
420 PRINT "EQUALS OR EXCEEDS THE LIMITS OF THE COUNTER`S ";
430 PRINT "MEASUREMENT CAPABILITY."
440 PRINT "J      * The p-p signal must fall within ";-3.175*5;" volts"
450 PRINT "      and ";3.2*5;" volts to guarantee p-p amplitude and"
460 PRINT "      risetime measurements."
470 GO TO 510
480 PRINT @18:"ATT 1;CHA A;ATT 1"
490 PRINT @18:"AUTO;MIN?;MAX?;CHA B;MIN?;MAX?"
500 INPUT @18:A0,A9,B0,B9
510 REM ----- CALCULATE 10% POINT AT CHANNEL A -----
520 PRINT @18:"ATT?"
530 INPUT @18:A5
540 A1=A0+(A9-A0)*0.1
550 REM ----- CALUCULATE 90% POINT AT CHANNEL B -----
560 B8=B9-(B9-B0)*0.1
570 PRINT @18:"CHA A;LEV ";A1;" ;CHA B;LEV ";B8
580 PRINT @18:"TIME;SEND"
590 INPUT @18:R
600 REM ----- CALCULATE CHA A 90% POINT FOR SLEW RATE -----
610 A8=A9-(A9-A0)*0.1
620 REM ----- PRINT OUT RESULTS OF SIGNAL MEASUREMENTS -----
630 PRINT "JJJI SIGNAL CHARACTERIZATION"
640 PRINT " IJJP-P AMPLITUDEI= ";A9-A0;" VOLTS"
650 PRINT " I ATTEN = ";A5
660 C$=REP(" ",3,1)
670 PRINT " I COUPLING = ";C$
680 PRINT " I MIN PEAK = ";A0;" I MAX PEAK = ";A9
690 PRINT " IJFREQUENCYI= ";F/1000;" KHz"
700 PRINT " IDUTY CYCLEI= ";INT(W/P*1000)/10;"%"
710 PRINT " IRISETIMEI= ";R*1000000;" uS"
720 PRINT " ISLEW RATEI= ";INT((A8-A1)/R)/1000000;" V/us"
730 GO TO 130
740 REM ----- SRQ ROUTINE -----
750 POLL X,Y;18
760 PRINT @18:"ERR?"
770 INPUT @18:E
780 IF E=403 THEN 800
790 PRINT "JERROR CODE: ";E;" HAS BEEN REPORTED"
800 RETURN

```

Line 100 -- Returns system environmental parameters to a known state.

Line 110 -- Informs controller of SRQ handler.

Lines 220 to 230 -- Frequency measurement routine.

Lines 250 to 260 -- Pulse width and period measurement routine.

Lines 280 to 290 -- Obtain the maximum and minimum values of the signal.

Line 300 -- Checks to see if the input signal is within the X1 attenuation limits. If it is, attenuation is changed to X1; if not, the attenuation remains at X5.

Line 310 -- Limits on the maximum and minimum of the input signal are determined by the limits of the trigger level circuit. These limits are  $\text{min} = -3.175$  and  $\text{max} = 3.2$  volts in the X1 attenuation mode and five times each respective limit in the X5 attenuation mode. This line determines whether either of the X5 limits have been exceeded and, if they have, then coupling is changed from DC to AC. Consider that AC coupling can make the rise/fall measurement duty-cycle dependent.

Line 330 -- If the coupling is already AC, then a warning is displayed on the screen informing the operator of possible errors in the measurements.

Lines 520 to 540 -- Calculate the 10% point and set Channel A level to it.

Lines 560 to 590 -- Calculate the 90% point, set Channel B level to it, and calculate rise time by measuring time A to B.

Line 750 to 800 -- SRQ handler.

This program includes a number of pieces that may be used separately. For example, lines 280 to 300 and 480 to 500 are a good approach to performing auto attenuation. The algorithm is basically this:

1. Start out in the X5 attenuation mode.
2. Check to see whether the limits in the X1 mode have been exceeded. These limits are  $-3.175$  to  $+3.2$  volts for the DC 5009 and  $-2.044$  to  $+2.048$  for the DC 5010. These values are multiplied by 5 to obtain the X5 limits.

3. If these limits have not been exceeded, then change the attenuation to X1.

For the auto coupling routine, the algorithm is basically this:

1. Start out in the DC coupled mode.
2. Check to see whether either limit has been exceeded.
3. If either of them have then change coupling to AC.

Note that whenever the attenuation or the coupling is changed, previous values for maximum and minimum are no longer valid, so a maximum and minimum query must again be executed, preceded by an auto trigger.

To change this program for the DC 5010:

1. Change the address from 18 to 20 (the DC 5010 factory-set address) or have the DC 5010 address changed to 18.
2. Change the trigger level limits in line 300 to 310 and 440 to 450 to -2.044 and +2.048.
3. Replace lines 520 through 610 with:

```

510 REMARK GET RISE/TIME AND 10%, 90% LEVELS
520 PRINT @20: "RISE;SEND;CHA A;LEV?;CHA B;LEV?"
530 INPUT @20: R,A1,A8
    
```

To use this program as modified for the DC 5010, connect the signal only to Channel A, not A and B, as the DC 5010 has an interval power splitter.

### Measuring Phase

A phase measurement is easily accomplished using the Time A to B function. Here's an example that would return the phase angle in degrees of a signal connected to B with respect to a signal connected to A.

```

10 REMARK MEASURE PHASE
100 PRINT @20:"INIT;PER;SEND"
110 INPUT @20:P
120 PRINT @20:"TIME;SEND"
130 INPUT @20:T
140 PRINT "DEGREES PHASE = ";INT(360*T/P)
150 END

```

To add this measurement to the characterization program above, convert the result to phase of A with respect to a reference signal connected to B by subtracting the result, INT(360\*T/P) from 360.

### Error Decoding

This listing includes an expanded SRQ subroutine that decodes the ERR? response and prints the instrument ID, the error number, and the corresponding error message. It is based on the full error-decoding routine in Section 7 that handles all TM 5000 instruments.

```

100 REM *****
110 REM *** SUBROUTINE TO PRINT DC 5009/DC5010 ERR? MESSAGES ***
120 REM *****
130 INIT
140 ON SRQ THEN 3000
150 PAGE
160 PRINT "TURN POWER ON"
170 B$=""
180 WAIT
190 ON SRQ THEN 3070
200 REM .
210 REM . ***** YOUR PROGRAM GOES HERE *****
220 REM .
230 REM .
3000 REM ***** DELAY FOR PON AND POLL CONFIGURATION *****
3010 CALL "WAIT",5
3020 DIM D(2)
3030 REM ***** DC 5009 @ ADDRESS 18 / DC 5010 @ ADDRESS 20 *****
3040 DATA 18,20
3050 RESTORE 3040
3060 READ D
3070 REM ***** POLL ROUTINE *****
3080 POLL X,Y;D(1);D(2)
3090 PRINT @D(X):"ERR?;ID?"
3100 INPUT @D(X):E,E$
3110 E$=SEG(E$,9,6)
3120 GOSUB 4000
3130 RETURN

```



```

4000 REM      ***** CODING FOR REPORTING ERROR? INFORMATION *****
4010 IF LEN(B$) THEN 4700
4020 DELETE B$
4030 DIM B$(2500)
4040 B$="      0      No errors or events to report"
4050 B$=B$&" 101      Command Header Error"
4060 B$=B$&" 102      Header Delimiter Error"
4070 B$=B$&" 103      Command Argument Error"
4080 B$=B$&" 104      Argument Delimiter Error"
4090 B$=B$&" 105      Non-numeric Argument (numeric expected)"
4100 B$=B$&" 106      Missing Argument"
4110 B$=B$&" 107      Invalid Message Unit Delimiter"
4120 B$=B$&" 201      Command Not Executable in Local"
4130 B$=B$&" 202      Settings lost due to rtl"
4140 B$=B$&" 203      I/O Buffers full, Output dumped"
4150 B$=B$&" 205      Argument Out of Range"
4160 B$=B$&" 206      Group Execute Trigger ignored"
4170 B$=B$&" 301      Interrupt Fault"
4180 B$=B$&" 302      System Error"
4190 B$=B$&" 313      Serial I/O Fault"
4200 B$=B$&" 320      Fault at U1221A or Input Amplifier (DC 5009)"
4210 B$=B$&"          Fault at U1000A or Input "
4220 B$=B$&"Amplifier (DC 5010)"
4230 B$=B$&" 321      Fault at U1211A (DC 5009)          "
4240 B$=B$&"          Fault at U1011A (DC 5010)"
4250 B$=B$&" 322      Fault at U1201A (DC 5009)          "
4260 B$=B$&"          Fault at U1810A (DC 5010)"
4270 B$=B$&" 323      Fault at U1113A (DC 5009)          "
4280 B$=B$&"          Fault at U1801A (DC 5010)"
4290 B$=B$&" 324      Fault at U1112A (DC 5009)          "
4300 B$=B$&"          Fault at U1120A (DC 5010)"
4310 B$=B$&" 325      Fault at U1111A"
4320 B$=B$&" 326      Fault at U1332"
4330 B$=B$&" 329      ""A"" chain failed to reset to zero"
4340 B$=B$&" 330      Fault at U1221B or Input Amplifier (DC 5009)"
4350 B$=B$&"          Fault at U1011C or Input "
4360 B$=B$&"Amplifier (DC 5010)"
4370 B$=B$&" 331      Fault at U1211B (DC 5009)          "
4380 B$=B$&"          Fault at U1011B (DC 5010)"
4390 B$=B$&" 332      Fault at U1201B (DC 5009)          "
4400 B$=B$&"          Fault at U1810B (DC 5010)"
4410 B$=B$&" 333      Fault at U1113B (DC 5009)          "
4420 B$=B$&"          Fault at U1801B (DC 5010)"
4430 B$=B$&" 334      Fault at U1112B (DC 5009)          "
4440 B$=B$&"          Fault at U1120B (DC 5010)"
4450 B$=B$&" 335      Fault at U1111B"
4460 B$=B$&" 336      Fault at U1012B"
4470 B$=B$&" 339      ""B"" chain failed to reset to zero"
4480 B$=B$&" 340      System RAM Error (U1332) (DC 5009)          "
4490 B$=B$&"          System RAM Error (U1410) (DC 5010)"
4500 B$=B$&" 341      System RAM Error (U1610)"
4510 B$=B$&" 342      System RAM Error (U1311)"
4520 B$=B$&" 343      System RAM Error (U1210)"

```

```

4530 B$=B$&" 361 1000 ROM Placement Error"
4540 B$=B$&" 374 E000 ROM Placement Error"
4550 B$=B$&" 375 F000 ROM Placement Error"
4560 B$=B$&" 380 0800 ROM Placement Error"
4570 B$=B$&" 381 1000 ROM Checksum Error"
4580 B$=B$&" 394 E000 ROM Checksum Error"
4590 B$=B$&" 395 F000 ROM Checksum Error"
4600 B$=B$&" 401 Power On"
4610 B$=B$&" 402 Operation Complete"
4620 B$=B$&" 403 User Request"
4630 B$=B$&" 602 Channel A Protect"
4640 B$=B$&" 603 Channel B Protect"
4650 B$=B$&" 604 No Prescaler"
4660 B$=B$&" 711 Channel A Overflow"
4670 B$=B$&" 712 Channel B Overflow "
4680 REM ***** SORTING THRU B$ *****
4690 DIM A$(200)
4700 A$=STR(E)
4710 A$=A$&" "
4720 E1=POS(B$,A$,1)
4730 E2=POS(B$," ",E1)
4740 A$=SEG(B$,E1,E2-E1+1)
4750 REM ***** REPORTING INSTRUMENT AND CODE *****
4760 PRINT "J ";E$;" --- ";A$
4770 RETURN

```

### Status Routine

This listing uses the low-level serial poll routine from Section 2 to get a status byte whether or not SRQ is asserted (operates with RQS OFF). The routine decodes the status byte and prints a label for the class of status reported.

```

1 REM {DC 5010 STB PARSER with ON SRQ}
100 INIT
110 ON SRQ THEN 50000
120 DIM B1(10,2)
130 DATA 97,113,98,114,99,115,65,81,66,82,67,83
140 DATA 102,118,193,209,194,210,0,16
150 RESTORE 130
160 READ B1
200 REM append here
50000 REM get stb
50010 WBYTE @63,24,84:
50020 RBYTE S
50030 WBYTE @25,95:
50040 FOR I=1 TO 10
50050 IF S=B1(I,1) OR S=B1(I,2) THEN 50090
50060 NEXT I
50070 PRINT "_STB = ";S;" Status Message Not FoundGG"
50080 RETURN

```

```
50090 GO TO I OF 50120,50140,50160,50180,50200,50220
50100 GO TO I-8 OF 50240,50260,50280,50300
50110 RETURN
50120 PRINT "_COMMAND ERROR"
50130 RETURN
50140 PRINT "_EXECUTION ERROR"
50150 RETURN
50160 PRINT "_INTERNAL ERROR"
50170 RETURN
50180 PRINT "_POWER UP"
50190 RETURN
50200 PRINT "_OPERATION COMPLETE"
50210 RETURN
50220 PRINT "_USER REQUEST FOR SERVICE"
50230 RETURN
50240 PRINT "_DEVICE WARNING"
50250 RETURN
50260 PRINT "_CHANNEL A OVERFLOW"
50270 RETURN
50280 PRINT "_CHANNEL B OVERFLOW"
50290 RETURN
50300 PRINT "_NO ERROR OR EVENT TO REPORT"
50310 RETURN
```

## SECTION 4

## DM 5010 PROGRAMMABLE DIGITAL MULTIMETER

The DM 5010 offers the GPIB programmer full control of multimeter measurements. Front-panel functions may be selected over the GPIB. Readings may be triggered and output over the GPIB (or triggered by an external signal). Readings may also be acquired "on-the-fly" (as soon as available in a free-run mode). Either a normal (4 1/2 digit) or fast (3 1/2 digit) conversion rate may be selected.

A number of calculation functions can massage readings to supply answers that are more useful than the raw data that was acquired.

Because DM 5010 messages are coded in high-level mnemonics programs are easier to write and are self-documenting.

A microprocessor provides the intelligence to handle measurements and math functions, as well as front-panel controls and indicators and high-level messages over the GPIB.

## DM 5010 Functions

The DM 5010 measures electrical parameters with the following functions:

## DC Volts

-1000 to +1000 volts.

Maximum resolution = 10 microvolts on the 200 millivolt range.

Basic accuracy = 0.015% +1 count.

Input Z = 1000 megohms to 20 volts and 10 megohms above 20 volts.

## AC Volts

0 to 700 volts.

True-rms responding, either AC only or AC+DC.

Maximum resolution = 10 microvolts on 200 millivolt range.

10 Hz to 100 kHz with crest factor of 4.

**Ohms**

0 to 20 megohms.

Maximum resolution = 10 milliohms on the 200 ohm range.

Low applied voltage does not turn on silicon p-n junctions.

**Diode Test**

Reads diode forward voltage drop at about 1 milliamp.

Reads open circuit for reverse direction.

**Current**

Current is calculated from voltage drop across external, user-supplied, resistor.

**Autorange**

Microprocessor selects range--override either from GPIB or front panel.

**Front/Rear Input**

Select either front or rear input from GPIB or front panel.

**Null**

Microprocessor nulling of lead resistance in ohms or set up a single-probe quasi-differential measurement with any function.

## Calculations

### Averaging

Average up to 19999 readings--programmable SRQ can alert controller when a measurement is ready.

### Scaling and Offset

(X-B)/A function enables a variety of measurements such as current (B=0 and A=R), temperature sensors, and percent deviation.

### dBm or dBr

Read dBm directly or enter a reference and read dBr.

### Compare

Enter high and low limits and read HI/PASS/LO--programmable SRQ can alert controller to activate alarm or log data on out-of-limits reading.

For the full set of specifications, see the instrument instruction manual.

## GPIB Operation

### IEEE 488 Bus Address

The DM 5010 is supplied with a primary address of 16. To observe the current address, press the INST ID button. A decimal point following the number indicates that LF or EOI has been selected as the device-dependent message terminator. The factory setting is EOI only, indicated by the absence of a decimal point. A negative sign is displayed if the instrument has been set to talk-only mode.

Switches for all of these settings are internal; refer to a qualified service person for a change. If the switches are changed, use an address between 1 and 30, EOI only, and talk-only off to operate with TEKTRONIX controllers.

### Power-Up

At power-up, the DM 5010 performs a self-test of ROM and RAM. If the test fails, the DM 5010 displays an error number. If the test succeeds, the DM 5010 goes to the local control state with power-up default settings, asserts SRQ, and prepares to report power-up status. Although the DM 5010 responds normally to GPIB messages whether or not the GPIB controller reads the power-up status, a 4050 BASIC program aborts if it does not include the ON SRQ THEN... statement and an SRQ handler (see Section 2 for more on this.)

TABLE 4-1  
POWER-UP DEFAULT SETTINGS

#### Front Panel

DCV	-1.E+3
NULL	0.0
LFR	OFF
MODE	RUN
DIGIT	4.5
AVE	2
RATIO	1.0, 0.0
DBR	1.0
LIMITS	0.0, 0.0
CALC	OFF

#### Interrupts

RQS	ON
MONITOR	OFF
OPC	OFF
OVER	OFF
USER	OFF

#### Device Trigger

DT	OFF
----	-----

### Selecting DM 5010 Functions

Simple commands select DM 5010 functions--often the same mnemonics used as labels on the front-panel buttons. Insert these commands in controller GPIB ASCII output statements to program the DM 5010 front panel. For example:

```
PRINT @16:"DCV"
```

instructs the DM 5010 to measure volts. This command also selects autoranging because a number argument (for the range) is omitted.

String DM 5010 commands together in a single message, even mix in query commands:

```
PRINT @16:"ACV -20;LFR?;MODE RUN"
INPUT @16:F$
```

sets the DM 5010 to measure AC volts starting in the 20 volt range, but autoranging (because of the negative argument). The query "LFR?" asks whether the low-frequency response function is on or off. Finally, "MODE RUN" selects continuous (internally triggered) readings. The INPUT statement stores the DM 5010 response to "LFR?" in string variable F\$.

**Default Range Settings.** You need not enter the exact value of the range desired. Suppose you want a range large enough to accommodate a variable not known to you when you are writing a program. Use the variable as the argument for the range and the DM 5010 will round the variable up to select a large enough range. For example:

```
PRINT @16:"OHMS ";R
```

sets the DM 5010 to the 20 kilohm range if the value of R is 7 kilohms, the 200 kilohm range if R is 50 kilohms, etc. This trick is used in the resistor sort program later in this section.

**Initializing Settings.** It is not necessary to program all DM 5010 functions to establish a known instrument state. Use the INIT command to restore the power-up state; if the desired instrument state differs, just add the necessary commands. For example, to set the instrument to measure the true RMS value of both AC and DC components of a signal with all other instrument parameters set to power-up default values:

```
PRINT @16:"INIT;ACDC"
```



## Calculations

The DM 5010 microprocessor can operate on readings to provide measurement answers directly, freeing the system controller from many operations. For example, the DM 5010 can null lead resistance or voltage offsets, average up to 19999 readings and report the result, scale and offset the reading, convert the reading to dBm or dB referred to a programmable reference, and determine whether a reading is within limits. These math functions may be cascaded in this order: averaging, scaling, dBm or dBr conversion, and compare to limits. (The null function, if selected, is performed before any of these four.) For instance, null, scaling, and limits functions are used in the power supply set program later in this section.

**Averaging.** Averaging improves the signal-to-noise ratio of a DC measurement under certain conditions. This is particularly apparent if the noise is asymmetrical and low-frequency. The common-mode rejection and normal-mode (noise on a single lead) rejection characteristics of the DM 5010 may overcome noise on a signal in many instances. Also, symmetrical noise at higher frequencies, such as white noise, is averaged by the integrating-type DM 5010 converter. To reduce noise by microprocessor averaging, use:

```
PRINT @16:"AVE ";N;" ;CALC AVE"
```

where N has been set to a positive number. Note that semicolons surround the variable N--this is part of the syntax of the 4050 BASIC statement. The semicolon in ";CALC AVE" is part of the DM 5010 message syntax and is required to separate the AVE command that selects the number of averages and the CALC command that selects the averaging function.

Averaging requires some time to complete--N/3 seconds for 4 1/2 digit readings, but the DM 5010 holds off any requested output until the averaging calculation is complete, as discussed under Triggering and Acquiring Readings below. Several techniques shown later in this section as application examples allow your program to execute other tasks in the meantime.

**Scaling.** The  $(X-B)/A$  function offsets the measurement by B (equivalent to NULL B) and scales the result by 1/A. This may be used to make accurate in-circuit current measurements. For example:

```
PRINT @16:"DCV;CALC RATIO;RATIO ";R,Ø
```

sets the DM 5010 to display current by dividing the voltage across R by the value previously assigned to R. The in-circuit current measurement program later in this section shows a way to do this.

In the above statement, the ratio arguments are sent outside the quote marks using the 4050 BASIC PRINT syntax as shown. This allows easy incorporation of variables in DM 5010 commands. Note also that the RATIO constants may be sent either before or after (as is the case above) the CALCulate RATIO command as long as both the constants and calculate command are part of the same message. Another use, % deviation, is shown as part of the power supply set program later in this section.

**dBm, dBr.** The dBm function converts readings to dB referenced to 1 mW in 600 ohms. The dBr function converts readings to dB compared to your reference:

```
PRINT @16;"CALC DBR;DBR ";R
```

where R is a variable set by the user. This is illustrated in the frequency response program as a system application, Section 7.

**Compare to Limits.** This function compares the measurement to programmable limits. Selecting the compare function causes the DM 5010 to display HI, LO, or PASS and output a "1.", "2.", or "3.", in response to the SEND command. Here's an example that sets the DM 5010 to compare an ohms reading to limits of 0.3 and 1. A reading below 0.3 causes the 4052 to beep faster; a reading between 0.3 and 1 causes the 4052 to beep slower (pause between beeps); and a reading above 1 causes the 4052 to skip the line that makes a beep.

```
10 REMARK COMPARE TO LIMITS
90 ON SRQ THEN 180
100 PRINT @16:"INIT;OHMS;LIMIT 0.3,1;CALC COMP;SEND"
110 INPUT @16:R1
120 REMARK      ***** BRANCH ON R1 *****
130 GO TO R1 OF 160,150,110
140 GO TO 110
150 CALL "WAIT",R1
160 PRINT "G";
170 GO TO 110
180 POLL A,B;16
190 PRINT "STATUS ";B
200 RETURN
```

As the program is running, try pressing FAST CONVERSION RATE when the DM 5010 is reading LO. The 4052 will beep at a different rate because part of the time between beeps occurs at line 110 while the 4052 waits for the DM 5010 to supply a reading.

Several programs shown later in this section illustrate other uses of the compare mode.

### Triggering and Acquiring Readings

**Run Mode.** Internally triggered (MODE RUN), the DM 5010 supplies readings as quickly as available--three readings/second in DIGIT 4.5 and more than 25 readings/second in DIGIT 3.5 (both assuming voltage measurements with LFR not selected). All that is required to get a reading is the 4050-series INPUT statement. An example that acquires readings as fast as available is shown in the low-frequency waveform rms program later in this section.

**Triggered Mode.** A reading may be triggered over the GPIB (MODE TRIG) by the GET interface message (DT TRIG), by SEND, or by the INPUT statement alone. Setting the DM 5010 to MODE TRIG clears any old reading. Each trigger causes the DM 5010 to take as many readings as required to output a valid reading even if autoranging or averaging is performed.

An example using triggered mode is:

```

10 REMARK TRIGGERED MODE EXAMPLE
100 DIM R(100)
110 PRINT @16:"MODE TRIG"
120 FOR I=1 TO 100
130 PRINT @16:"SEND"
140 INPUT @16:R(I)
150 REM
160 REMARK      ***** SET UP PROGRAMMABLE SYSTEM *****
170 REMARK      ***** HERE FOR NEXT MEASUREMENT *****
180 REM
190 NEXT I

```

Line 130 is not required unless an averaging operation is triggered that would take longer than five seconds. If the DM 5010 is responding to the INPUT statement alone, it times out after five seconds and outputs a byte with all bits set to one.

The DATA command may be used to request the last reading taken by the DM 5010 without triggering a new reading. The main purpose of DATA is to recover the reading that triggered an out-of-limits SRQ in compare mode:

for the MON ON condition, the DM 5010 saves an out-of-limits reading until read with the DATA command. See the line voltage and max value programs later in this section for applications of the DATA command.

**Ac Settling Time.** A filter in the true-rms converter causes a worst-case step response of 1.2 seconds to rated accuracy for ac measurements. Use CALL "WAIT" 4050-series routine to force as much delay as proves necessary (up to 1.2 seconds) for reliable data. If the amplitude of the ac signal does not change as a step function, little (or no) delay may be necessary.

A range change also causes the rms converter to see a step change. In this case also, delay may be necessary before acquiring an ac measurement.

If autoranging, the DM 5010 automatically takes another measurement after a range change. This is an adequate safeguard for dc or ohms measurements, so you can rely on the first dc or ohms reading reported after a range change. For ac measurements, however, the rms converter filter may still be changing after a step change in input signal amplitude. For this reason, it is up to the program to accomplish the required delay.

An overrange interrupt is provided: "OVER ON" causes the DM 5010 to assert SRQ and report overrange status. The program can branch to a CALL "WAIT" routine on this interrupt. This is illustrated in the frequency response application in Section 7. The rms converter response to a downrange condition is quicker, so a program delay may not be necessary.

### Application Examples

The following measurement routines illustrate ways to use DM 5010 functions with a GPIB controller. These examples are intended to suggest solutions for programmable measurement tasks. You may benefit by adapting and incorporating these examples in your own programs or by applying the techniques that appear in the examples.

Most are complete programs that incorporate an SRQ handler. All use primary address 16 to communicate with the DM 5010.

#### Data Logging

This program logs DM 5010 readings to tape. The DM 5010 must be in MODE RUN--the program does not trigger readings. The WAIT routine spaces out data samples. This program also illustrates how to read the data file from tape. After doing so, the 4052 proceeds to compute and print the mean.

This program does not change DM 5010 functions--select DCV or ACV, DIGIT 3.5 or 4.5, or calculations, etc. before running.

```

10 REMARK DM5010 TO TAPE DATA LOGGING
20 ON SRQ THEN 360
100 ON EOF (0) THEN 320
110 REMARK ***** FIND TAPE FILE *****
120 FIND 3
130 REMARK ***** LOG READINGS *****
140 F1=0
150 FOR I=1 TO 100
160 PRINT @16:"SEND"
170 INPUT @16:R
180 WRITE @33:R
190 IF F1 THEN 220
200 CALL "WAIT",0.01
210 NEXT I
220 I=I-1
230 PRINT I;" DATA LOGGED"
240 REMARK ***** READ FILE AND FIND MEAN *****
250 DELETE S
260 DIM S(I)
270 FIND 3
280 READ @33:S
290 PRINT "FILE READ; MEAN = ";SUM(S)/I
300 END
310 REMARK ***** EOF SUBROUTINE *****
320 F1=1
330 PRINT "FILE FULL;";
340 RETURN
350 REMARK ***** POLL SUBROUTINE *****
360 POLL A,B;16
370 PRINT "#";A;" REPORTS ";B
380 RETURN

```

Line 120 -- Change 3 to the number of any desired tape file that is marked large enough (2000 is enough for 100 data).

Line 140 -- Clears F1, a flag used by the EOF subroutine.

Lines 150 to 210 -- Read and store 100 readings unless the EOF subroutine sets F1. Vary line 150 for the desired number of data values. Vary line 200 for the desired delay (0.01 second is negligible to allow loop to complete quickly).

Line 220 -- Corrects I, the data counter. I=101 if FOR/NEXT loop completes 100 readings. If the loop does not complete, this line keeps last data value, which may be invalid, from counting.

Line 230 -- Prints message to operator.

Lines 250 to 290 -- Finds data (again change file number as desired), reads data, and computes mean making use of SUM function.

Lines 320 to 340 -- Handles the end-of-file condition (tape file marked too small). This subroutine only sets F1; it is up to the logging routine to test the flag after each tape write.

Lines 360 to 380 -- Handles unexpected SRQ events.

### Power Supply Setting--Three DM 5010 Display Modes

Setting a voltage, such as a power supply output, is a common DMM use.

In addition to the normal DMM display of volts, three other DM 5010 display modes are available for this purpose. The following program asks the operator to select the mode, and the program then sets up the display. This program also illustrates how to incorporate the user request (INST ID button) SRQ.

```

10 REMARK DM5010 POWER SUPPLY SET PROGRAM
20 ON SRQ THEN 410
100 REMARK          ***** INIT DM5010 *****
110 PRINT @16:"INIT"
120 REMARK          ***** INPUT VOLTS, DM5010 DISPLAY MODE *****
130 PRINT "ENTER IDEAL VOLTAGE AND PERCENT TOLERANCE"
140 INPUT V,P
150 PRINT "ENTER DM5010 DISPLAY MODE:I1=DIFFERENCE"
160 PRINT "I2=% DEVIATION"
170 PRINT "I3=PASS/FAIL"
180 INPUT M
190 PRINT "***** PRESS DM 5010 INST ID BUTTON WHEN DONE"
200 REMARK          ***** SET UP NULL *****
210 PRINT @16:"NULL ";V
220 IF M=1 THEN 290
230 REMARK          ***** SET UP % DEVIATION IF REQUESTED *****
240 PRINT @16:"RATIO ";V*0.01,0;"CALC RATIO"
250 IF M=2 THEN 290
260 REMARK          ***** SET UP PASS/FAIL IF REQUESTED *****
270 PRINT @16:"LIMITS ";P,-P;"CALC RATIO,CMPR"
280 REMARK          ***** LOOP TILL USER INTERRUPTS *****
290 E=0
300 PRINT @16:"USER ON"
310 IF E<>403 THEN 310
320 PRINT @16:"USER OFF"
330 REMARK          ***** CHECK FOR CORRECT V *****
340 PRINT @16:"CALC OFF"
350 INPUT @16:V1
360 IF ABS(V1/V*100)>P THEN 380
370 END
380 PRINT "GGGGGREPEAT ADJUSTMENT FOR CORRECT SETTING"
390 GO TO 220
400 REMARK          ***** POLL ROUTINE *****
410 POLL A,B;16
420 PRINT @16:"ERR?"
430 INPUT @16:E
440 IF B<=67 THEN 460
450 PRINT "STATUS ";B;" ERROR ";E
460 RETURN

```



Lines 130 to 190 -- Prints instructions and inputs operator choices.

Lines 210 to 220 -- Subtracts ideal voltage from current reading; skips to difference display; if this display was requested, the program skips other set-ups.

Lines 240 to 250 -- Calculates % deviation ( $A=0.01*V$  and  $B=0$ ) and skips pass/fail display if it was not requested.  $B=0$  because the reading was already nulled.

Line 270 -- Sets up limits and compare mode for HI/LO/PASS display. Note in line 270 (and 240) that multiple parameters for the RATIO and LIMITS commands may be included in the PRINT statement as variables outside the quote marks.

Lines 300 to 320 -- Enables, checks for, and then disables the user interrupt, relying on the error code E obtained by the POLL routine.

Lines 340 to 370 -- Checks for correct absolute voltage and ends program.

Lines 410 to 460 -- A poll routine that incorporates an error query for more detailed information than the instrument reports in the status byte. The status and error codes are printed unless the status byte corresponds to the user interrupt.

#### A Resistor Sorting Program

This program sets up the desired limits for good readings and aids the operator in sorting resistors. The program could be adapted to activate a parts handler by branching on the value of M.

```

10 REMARK RESISTOR SORT ROUTINE
90 ON SRQ THEN 270
100 PRINT "ENTER RESISTOR VALUE: ";
110 INPUT R
120 PRINT "ENTER % TOLERANCE: ";
130 INPUT T
140 REMARK          ***** SET UP DM5010 *****
150 PRINT @16:"INIT;OHMS ";R*(1+0.01*T)
160 PRINT @16:"RATIO ";0.01*R;R;";LIM ";T;-T;";CALC RATIO,CMPR"
170 REMARK          ***** WAIT FOR TWO GOOD READINGS *****
180 FOR I=1 TO 2
190 INPUT @16:M
200 IF M=2 THEN 220
210 I=0
220 NEXT I
230 REMARK          ***** RING BELL FOR PASS *****
240 PRINT "G";
250 GO TO 180
260 REMARK          ***** SRQ HANDLER (N.A.) *****
270 POLL A,B;16
280 PRINT @16:"ERR?"
290 INPUT @16:E$
300 PRINT "I$STATUS ";B,E$
310 RETURN

```

Lines 100 to 160 -- Sets up DM 5010 compare mode using values input by the operator.

Lines 180 to 250 -- Fail-safe check--pass only if two consecutive readings are good. If so, signal the operator.

Lines 270 to 310 -- Handles unexpected SRQs.

### Monitoring Line Voltage

This program sets up the DM 5010 for unattended data logging to tape the time and data for out-of-limits line voltage conditions. This saves storage by logging only the desired readings. It can be adapted to monitor many other conditions by changing the DM 5010 function and limits in line 150. You may stop the program with the BREAK before the tape file is filled. After running, follow the instructions in line 20 to reset the 4050 page-full mode.

This program uses the File Manager ROM Pack to obtain the time; it could use the Real Time Clock ROM Pack instead by modifying line 260.

```

10 REMARK LINE VOLTAGE MONITOR PROGRAM
20 REMARK RESET PAGE FULL PARAMETER AFTER END (PRI@32,26:0)
100 PRINT @32,26:2
110 FIND 3
120 ON EOF (0) THEN 360
130 ON SRQ THEN 180
140 REMARK          ***** SET UP DM5010 AS MONITOR *****
150 PRINT @16:"INIT;ACV 200;LIMITS 105,125;MONITOR ON"
160 WAIT
170 GO TO 160
180 REMARK          ***** POLL FOR STAU5 BYTE AND ERROR CODE *****
190 POLL X,Y;16
200 PRINT @16:"ERR?"
210 INPUT @16:E
220 IF E=701 OR E=703 THEN 250
230 PRINT "STATUS ";Y;" ERROR ";E;" REPORTED"
240 GO TO 350
250 REMARK          ***** LOG OUT-OF-LIMITS DATA *****
260 CALL "TIME",A$
270 PRINT @16:"DATA"
280 INPUT @16:D$
290 PRINT @16:"MON OFF"
300 A$=D$&A$
310 PRINT A$
320 WRITE @33:A$
330 CALL "WAIT",10
340 PRINT @16:"MON ON"
350 RETURN
360 END
1000 REMARK READ AND PRINT DATA FROM TAPE
1010 PRINT @32,26:0
1020 FIND 3
1030 FOR I=1 TO 1000
1040 READ @33:A$
1050 PRINT A$
1060 NEXT I

```

Line 100 -- Keep page-full condition from holding up program execution.

Lines 110 to 130 -- Set up tape and program; change 110 to find any marked file, as desired.

Lines 150 to 170 -- Set up DM 5010 and wait for interrupt.

Lines 190 to 240 -- Poll and query error; check error for out-of-limits error (701=low, 703=high). Branch to log out-of-limits reading or print other (unexpected) status and error.

Lines 260 to 350 -- Get the time and out-of-limits reading (DATA), concatenate, and write on tape. Shut off further immediate out-of-limits readings, wait 10 seconds, then resume limits monitoring and return.

The file manager time must be initialized previously with

```
CALL "SETTIME", "dd-mmm-yy hh:mm:ss"
```

To operate with the Real Time Clock ROM Pack, change line 260 to:

```
260 CALL "RDTIME",A$
```

and initialize the time with:

```
CALL "SETTIME", "dd-mmm-yy hh:mm:ss"
```

Lines 1010 to 1060 -- You may survey data on tape by entering "RUN 1000" after the previous lines are executed. If you changed line 110, change line 1020 to match.

#### In-Circuit Current Measurement

This program prompts the operator to measure a resistor, enters the value, and uses it to calculate current based on the voltage measured. The program turns off power to the circuit while the resistor is being measured.

```

10 REMARK IN-CIRCUIT CURRENT MEASUREMENT
90 ON SRQ THEN 1000
100 PRINT @22:"OUT OFF"
110 PRINT @16:"INIT;OHMS"
120 PRINT "1) SHORT DMM LEADS TO SAME SIDE OF RESISTOR ";
130 PRINT "AND PRESS ""NULL"""
140 PRINT "2) MOVE ONE LEAD TO OTHER SIDE OF RESISTOR ";
145 PRINT "AND PRESS ""RETURN"""
150 INPUT A$
160 INPUT @16:R1
170 PRINT @16:"DCV;CALC RATIO;RATIO ";R1,0
180 PRINT @22:"OUT ON"
190 END
1000 POLL A,B;16;22
1010 PRINT "STATUS ";B;" FROM ADDRESS ";16+6*(A-1)
1020 RETURN

```

Lines 100 and 180 -- control power supplied to circuit by a PS 5010.

Lines 110 to 150 -- Set up DM 5010 and instruct operator; wait on INPUT until operator performs the instructions.

Lines 160 to 170 -- INPUT the resistor value and begin displaying current by measuring the voltage and scaling the reading using R1.

#### Averaging Using RDY? Or OPC

Averaging takes awhile to complete as noted earlier in this section. Here's how to use RDY? or the OPC interrupt to check for completion.

First, the RDY? query. This program uses GET to trigger a new measurement only after the old one is read.

```

10 REMARK AVERAGE WITH RDY? QUERY
90 ON SRQ THEN 240
100 REMARK      ***** SET UP AVERAGED MEASUREMENT *****
110 PRINT @16:"DT TRIG;MODE TRIG;AVE 30;CALC AVE"
120 REMARK      ***** SEND GET INTERFACE MESSAGE *****
130 CALL "WAIT",0.2
140 WBYTE @32+16,8,63:
150 REMARK      ***** PROGRAM GOES HERE; TEST RDY? FOR DONE *****
160 PRINT @16:"RDY?"
170 INPUT @16:D1
180 IF D1 THEN 200
190 GO TO 160
200 INPUT @16:R1
210 PRINT "AVERAGE READING = ";R1
220 GO TO 140
230 REMARK      ***** SRQ HANDLER *****
240 POLL S,B;16
250 PRINT @16:"ERR?"
260 INPUT @16:E$
270 PRINT E$;" STATUS ";B
280 RETURN

```

Lines 110 to 140 -- Set up triggered, averaged measurement; wait for execution of command string; then send GET.

Lines 150 to 190 -- Your program can do other tasks, returning to check for measurement ready as is illustrated here.

Lines 200 to 220 -- Get reading and go for another one.

Lines 240 to 280 -- Handle any unexpected SRQs.

Second, the OPC interrupt. This program also operates in triggered mode, but uses SEND to trigger a measurement.

```

10 REMARK AVERAGE WITH OPC INTERRUPT HANDLER
90 ON SRQ THEN 190
100 REMARK      ***** SET UP MEASUREMENT AND DONE FLAG *****
110 PRINT @16:"MODE TRIG;AVE 30;CALC AVE;OPC ON"
120 PRINT @16:"SEND"
130 F1=1
140 REMARK      ***** PROGRAM GOES HERE;TEST FLAG FOR DONE *****
150 IF F1 THEN 150
160 PRINT "AVERAGE READING = ";R1
170 GO TO 120
180 REMARK      ***** SRQ HANDLER *****
190 POLL S,B;16
200 IF B=66 OR B=82 THEN 230
210 PRINT B;" STATUS REPORTED"
220 RETURN
230 INPUT @16:R1
240 F1=0
250 RETURN

```

Line 110 -- Set up averaged, triggered modes.

Line 120 -- Trigger a measurement.

Lines 130 to 170 -- Set flag and test, looping until reading is acquired, then print it. Repeat loop indefinitely.

Line 190 to 250 -- Poll on interrupt, printing unexpected status (line 210) or inputting reading (line 230) if status indicated operation complete.

### Find Max Value

The SRQ subroutine is expanded to update a variable that holds the biggest DM 5010 reading (greater than zero). The DM 5010 performs the task of testing each reading against the current champion, which the controller has loaded in as one of the DM 5010 limits.

```

10 REMARK SRQ SUBROUTINE FINDS MAX VALUE
20 ON SRQ THEN 220
30 WBYTE @48,20,63:
40 PRINT @16:"INIT"
50 PRINT "PRESS <RETURN> WHEN READY....."
60 INPUT A$
100 REMARK          ***** SET UP DM5010 *****
110 PRINT @16:"FUNC?"
120 INPUT @16:S$
130 S=VAL(S$)
140 S=ABS(S)
150 S$=SEG(S$,1,3)
160 PRINT @16:"MODE RUN;LIMITS -99,0;CALC CMPR;";S$,S
170 PRINT @16:"MON ON;OVER ON;RQS ON"
180 REMARK          ***** WAIT LOOP HOLDS PLACE FOR YOUR PROGRAM *****
190 WAIT
200 GO TO 190
210 REMARK          ***** SRQ ROUTINE UPDATES RANGE OR LIMITS *****
220 POLL A,B;16
230 PRINT @16:"ERR?"
240 INPUT @16:E
250 IF E<>601 THEN 300
260 M1=S
270 IF S=1000 THEN 350
280 S=10*S
290 PRINT @16:S$,S
300 IF E<>703 THEN 350
310 PRINT @16:"DATA"
320 INPUT @16:M1
330 PRINT @16:"LIMIT -99,";M1
340 PRINT "NEW MAX =";M1
350 RETURN

```

Lines 20 to 60 -- Restore power-up values and start when operator is ready. Add to line 40 if you want other than power-up settings.

Lines 110 to 170 -- Take instrument out of autorange, but leave in current range; guarantee free-run mode, set up limits and compare mode; and turn on interrupts.

Lines 190 to 200 -- Ad lib your own program here.



Lines 220 to 250 -- Get status and error codes.

Lines 250 to 290 -- For overrange condition, remember top of current range S as max value M1. If not already at highest range, move up one range (10\*S).

Lines 300 to 340 -- For beyond-upper-limit condition, get out-of-limits reading (DATA) as new M1; put it into DM 5010 limits parameter; and print new max value.

### P-P Reading of Square Wave

This technique acquires square wave positive and negative peak values up to 20 hertz if you have selected the fast readings mode and set the DCV range big enough for the signal.

```

10 REMARK SQUARE WAVE PEAK-PEAK READING
20 ON SRQ THEN 260
100 REMARK          ***** GET READINGS *****
110 DELETE R
120 DIM R(100)
130 FOR I=1 TO 100
140 INPUT @16:R(I)
150 NEXT I
160 REMARK          ***** FIND MAX, MIN *****
170 M1=R(1)
180 M2=R(1)
190 FOR I=2 TO 100
200 M1=M1 MAX R(I)
210 M2=M2 MIN R(I)
220 NEXT I
230 PRINT "MAX = ";M1,"MIN = ";M2
240 END
250 REMARK          ***** POLL ROUTINE *****
260 POLL A,B;16
270 PRINT "STATUS ";B;" REPORTED"
280 RETURN
    
```

Lines 100 to 150 -- Fill a 100-point array.

Lines 170 to 230 -- Sort through the array, point-by-point, for biggest and smallest readings; print them.

## Low-Frequency Rms

DM 5010 true-rms ac readings yield rated accuracy down to 10 hertz and are useful below that. At very low frequencies, however, readings begin to follow the input as though it were a varying dc signal. Here's a program that returns rms readings for signals down to about 0.4 hertz by digitizing a waveform, finding a full cycle, and calculating its rms value. The input waveform must be continuously varying--the find-cycle routine fails on square waves.

```

10 REMARK LOW FREQUENCY WAVEFORM RMS
20 ON SRQ THEN 700
100 REMARK      ***** GET DATA *****
110 PRINT @16:"FUNC?"
120 INPUT @16:S
130 S=ABS(S)
140 PRINT @16:"DIGIT 3.5;DCV ";S
150 DELETE B5
160 DIM B5(90)
170 FOR I=1 TO 90
180 INPUT @16:B5(I)
190 NEXT I
200 REMARK      ***** SCALE GRAPH *****
210 PAGE
220 VIEWPORT 20,120,10,90
230 WINDOW 0,90,-S,+S
240 AXIS 14.3,S/4,1,0
250 PRINT @32,21:0,51
260 PRINT " V/D="
270 PRINT S/4
280 PRINT @32,21:60,5
290 PRINT "T/D=0.5"
300 REMARK      ***** GRAPH DATA *****
310 MOVE 1,B5(1)
320 FOR I=2 TO 90
330 DRAW I,B5(I)
340 NEXT I
350 REMARK      ***** FIND FULL CYCLE *****
360 F1=0
370 IF B5(2)>B5(1) THEN 390
380 F1=1
390 FOR I=2 TO 90
400 IF B5(I)>B5(1) THEN 430
410 NEXT I
420 GO TO 660
430 FOR J=I+1 TO 90
440 IF B5(J)<B5(I) THEN 470
450 NEXT J

```

```

460 GO TO 660
470 T1=J
480 IF F1 THEN 540
490 FOR K=J+1 TO 90
500 IF B5(K)>B5(J) THEN 530
510 NEXT K
520 GO TO 660
530 T1=K
540 REMARK          ***** RMS CALCULATION *****
550 B5=B5^2
560 R=B5(1)/2
570 FOR I=1 TO T1-1
580 R=R+B5(I)
590 NEXT I
600 R=R+B5(T1)/2
610 R=R/T1
620 R=SQR(R)
630 PRINT USING ""^""3L50T""RMS = ""3D.3D""V""":R
640 GO TO 680
650 REMARK BRANCH HERE IF FULL CYCLE SEARCH FAILS
660 PRINT
670 PRINT " ** ROUTINE DID NOT FIND A FULL CYCLE FOR RMS CALCULATION **"
680 END
690 REMARK          ***** POLL ROUTINE *****
700 POLL A,B;16
710 PRINT "STATUS ";B;" REPORTED"
720 RETURN

```

Lines 110 to 140 -- Set up fast readings mode and defeat autorange (assumes present range is large enough for waveform).

Lines 150 to 190 -- Get 90 readings; this limits lowest frequency at which a full cycle is acquired to about 0.4 hertz and is a compromise to limit wait for acquiring higher frequencies.

Lines 220 to 290 -- Size graph and scale the plot of the 90 readings, then label axes.

Lines 310 to 340 -- Move to first point and draw lines to rest of the points.

Lines 360 to 390 -- Clear F1 (initial slope = positive); set F1 if initial slope is negative.

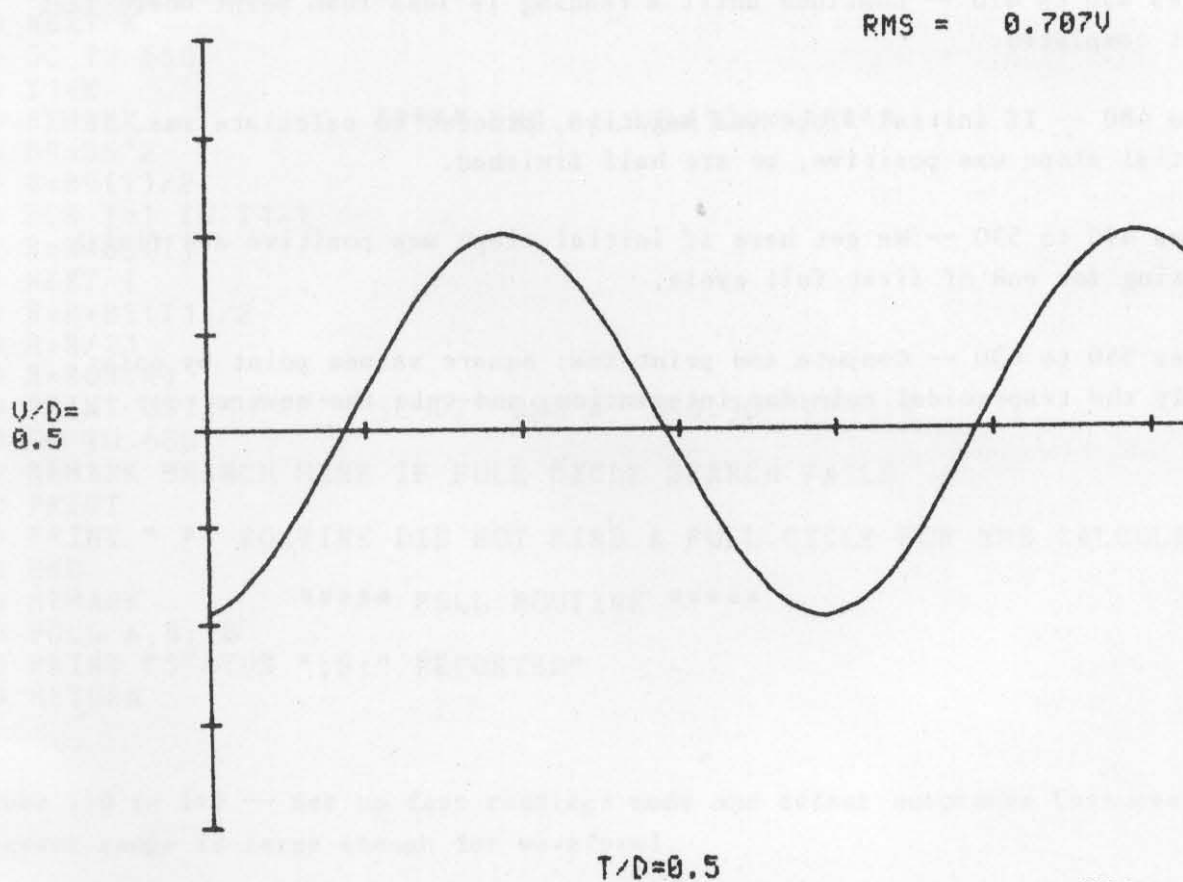
Lines 390 to 420 -- Find next reading that is greater than the initial reading. If initial slope was negative, we are half finished. If initial slope was positive, we are just beginning. If this test exhausts all readings, fail at 660.

Lines 430 to 470 -- Continue until a reading is less than point where last test completed.

Line 480 -- If initial slope was negative, proceed to calculate rms. If initial slope was positive, we are half finished.

Lines 490 to 530 -- We get here if initial slope was positive and finish testing for end of first full cycle.

Lines 550 to 630 -- Compute and print rms: square values point by point, apply the trapezoidal rule for integration, and take the square root.



3985-01

Fig. 4-1. Sample output of rms program.

### Error Decoding

This is an expanded SRQ subroutine that decodes the ERR? response to print an error message. It is based on the full error-decoding routine in Section 7 that handles not only the DM 5010, but other TM 5000 instruments as well.

```

100 REM *****
110 REM *** SUBROUTINE TO PRINT ERROR MESSAGE IN RESPONSE TO ERR? ***
120 REM *****
130 INIT
140 ON SRQ THEN 3070
150 REM
160 REM ***** YOUR PROGRAM GOES HERE *****
170 REM
3060 REM ***** POLL ROUTINE *****
3070 POLL X,Y;16
3080 PRINT @16:"ERR?;ID?"
3090 INPUT @16:E,E$
3100 E$=SEG(E$,9,6)
3110 GOSUB 4000
3120 RETURN
4000 REM ***** CODING FOR REPORTING ERROR? INFORMATION *****
4010 IF LEN(B$) THEN 5700
4020 DELETE B$
4030 DIM B$(1500)
4040 B$=" 101 Command Header Error"
4050 B$=B$&" 102 Header Delimiter Error"
4060 B$=B$&" 103 Command Argument Error"
4070 B$=B$&" 104 Argument Delimiter Error"
4080 B$=B$&" 105 Non-numeric Argument (numeric expected)"
4090 B$=B$&" 106 Missing Argument"
4100 B$=B$&" 107 Invalid Message Unit Delimiter"
4130 B$=B$&" 201 Command Not Executable in Local"
4140 B$=B$&" 202 Settings lost due to rtl"
4150 B$=B$&" 203 I/O Buffers full, Output dumped"
4160 B$=B$&" 204 Settings Conflicts"
4170 B$=B$&" 205 Argument Out of Range"
4180 B$=B$&" 206 Group Execute Trigger ignored"
4200 B$=B$&" 231 Not in Calibrate Mode"
4210 B$=B$&" 232 Beyond Calibration capability"
4300 B$=B$&" 301 Interrupt Fault"
4310 B$=B$&" 302 System Error"
4320 B$=B$&" 303 Math Pack Error"
4330 B$=B$&" 311 Timeout (measurement not completed)"
4610 B$=B$&" 340 System RAM Error"
4620 B$=B$&" 341 System RAM Error (low nibble)"
4720 B$=B$&" 351 Calibration RAM Checksum Error"
4850 B$=B$&" 372 C000 ROM Placement Error"
4860 B$=B$&" 373 D000 ROM Placement Error"
4870 B$=B$&" 374 E000 ROM Placement Error"
5010 B$=B$&" 392 C000 ROM Checksum Error"
5020 B$=B$&" 393 D000 ROM Checksum Error"
5030 B$=B$&" 394 E000 ROM Checksum Error"
5040 B$=B$&" 395 F000 ROM Checksum Error"
5050 B$=B$&" 401 Power On"
5060 B$=B$&" 402 Operation Complete"
5070 B$=B$&" 403 User Request"
5080 B$=B$&" 521 Display During Signature Analysis"
5090 B$=B$&" 601 Overage"
5150 B$=B$&" 701 Below Limits"
5160 B$=B$&" 703 Above Limits "

```

```
5690 REM          ***** SORTING THRU B$ *****
5700 A$=STR(E)
5710 A$=A$&" "
5720 E1=POS(B$,A$,1)
5730 E2=POS(B$," ",E1)
5740 A$=SEG(B$,E1,E2-E1+1)
5750 REM          ***** REPORTING INSTRUMENT AND CODE *****
5760 PRINT "J ";E$;" --- ";A$
5770 RETURN
```

Your program should initialize B\$ (B\$="") before calling this routine the first (and only the first) time. See the longer version of this program in Section 7 for an example of how this is done.

## SECTION 5

## FG 5010 PROGRAMMABLE FUNCTION GENERATOR

The FG 5010 is a versatile signal source with programmable frequency, amplitude, offset voltage, and waveshape. A variety of operating modes are available in addition to the continuous (free-running) mode. When stimulated by external signals applied to the TRIG/GATE input, the AM input, or the FM/VCF input, the FG 5010 can provide almost any combination of parameters in an output waveform.

## FG 5010 Functions

The FG 5010 provides sine, square/pulse, and triangle/ramp waveforms, which can be programmed for:

**Frequency**

0.002 to 20 MHz.

Microprocessor monitors frequency using counter techniques for 0.1% accuracy in free-run mode.

**Amplitude**

20 mV to 20 V p-p open circuit.

10 mV to 10 V p-p into 50 ohms.

4-digit resolution.

**Offset**

0 to +7.5 V open circuit.

0 to +3.75 V into 50 ohms.

Amplitude plus offset  $\leq$  15 V (open circuit), 7.5 V (50 ohms).

Resolution = 20 mV (open circuit), 10 mV (50 ohms).



**Normal/Complement**

Output can be inverted around the programmed offset.

**Triggered, Gated, or Nburst (1 to 9,999 cycles)**

Responds to external signal,  
GET interface message, or  
Manual trigger.

**Phase**

+90 degrees from control signal.  
One degree steps.  
Sets baseline in triggered, gated, or burst mode.

**Symmetry**

10% to 90% to create pulses and ramps.  
One percent steps.

**Output Hold**

0.002 to 200 Hz.  
Can hold instantaneous level.

**Phase Lock**

20 Hz to 20 MHz range.  
Automatically scans and locks to control signal.  
Programmable SRQ for phase-lock condition.

**Modulation**

Uses external modulation signal.  
 AM or FM.  
 DC to 100 kHz.

**Swept Frequency**

Uses external sweep signal.  
 0 to 10 V produces 1:1000 frequency change.

**GPIB Operation**

**IEEE 488 Bus Address**

Pressing the INST ID button causes the FG 5010 to display its GPIB primary address and message terminator. A decimal point indicates the message terminator is set for LF or EOI; the absence of a decimal point indicates it is set for EOI only.

The FG 5010 is set at the factory for address 24 and EOI only; these settings may be changed by changing the switches on the back of the plug-in.

Select any address from 1 to 30 and EOI only to operate with 4050-Series controllers. The FG 5010 does not use and ignores secondary addresses.

**Power-Up Self-Test**

The FG 5010 performs an internal diagnostic self-test after the instrument has powered up or received the TEST command. In the case of power-up, the front panel controls are always set to fixed defaults and the SRQ line is asserted to interrupt the controller; the SRQ line is not released until the controller serial polls the FG 5010 for the power-up status byte. Diagnostic errors found during power-up cause the FG 5010 to continue to assert the SRQ line until an additional status byte is reported to indicate the internal error.

FG 5010 settings may be restored to their power-up state by the INIT command.

TABLE 5-1  
POWER-UP DEFAULT SETTINGS

Front Panel

FREQ	1.0E+3
AMPL	500.0E-3
OFFS	0.0
SYMM	50
PHASE	0
NBUR	10
FUNC	SINE
MODE	CONT
SLOPE	POS
OUT	OFF
COMP	OFF
AM	OFF
FM	OFF

Interrupts

VCF	OFF
HOLD	OFF
GATE	OFF
PLI	OFF
USER	OFF
RQS	ON

Device Trigger

DT	OFF
----	-----

## Programming FG 5010 Functions

### Setting Commands and Queries

One or more setting commands and queries may be sent in a message to the FG 5010 in 4050-Series PRINT statements:

```
PRINT @24:"FUNC SQUARE"
PRINT @24:"FUNC SQUARE;FREQ 1.1E+6;AMPL?"
```

The FG 5010 accepts variations that make the programming task less rigorous. For instance, "FUNC SQUARE" could be sent as "FUNCTION SQUARE", "FUNC SQU", or simply "SQUARE".

In the example above, "AMPL?" readies the FG 5010 to return: "AMPL <NUMBER>", which may be read by the 4050-Series statements:

```
INPUT @24:A$
INPUT @24:A1
```

as the string A\$ or as just a number in variable A1. Most setting commands may be turned into queries by adding a question mark to the command mnemonic as in the case of "AMPL?". Executing the 4050-Series INPUT statement without first sending a query causes the FG 5010 to output a single byte with all bits set to one (decimal 255).

All FG 5010 commands are included in the Appendix. Refer to the power-up settings above for a list of programmable settings. And for a more complete discussion of FG 5010 programming formats and syntax, refer to the instrument instruction manual.

### High-Level Settings Query

FG 5010 settings may be queried as a group with "SET?". The response is a sequence of message units that defines all the front panel settings and system settings. Use this capability to learn all front panel settings and store them in computer memory as string variables for use at a later time. For more information on how to use "SET?" with 4050-Series BASIC, see Section 2.

The SET query response contains valid instrument commands and will be executed if sent back to the instrument.

### Low-Level Settings Query

The FG 5010 has the added capability of sending its current programmed status back in the form of a binary block message. To make use of this message, you must send it back to the instrument in the form in which it was received; no alteration is permitted. This low-level message will be sent by the FG 5010 only when it is prompted with the "LLSET?" query or the "SEND <n>" command (n for one of ten stored settings). The syntax of the binary block is of the general form:

```
HEADER <SPACE> % <BINARY BLOCK> ;
```

The bytes sent in the binary block are not ASCII and must be stored as numeric variables in the controller's memory. Refer to the low-level settings application programs later in this section for more details.

### Special Functions with Group Execute Trigger (GET)

The response of the FG 5010 to receipt of the Group Execute Trigger interface control message is determined by the DT command. Four possible actions can be programmed: no action (DT OFF), update to new settings (DT SET), execute one output event of TRIG or NBURST (DT TRIG), or toggle the gate on or off in GATE mode (DT GATE).

The appropriate response to GET must be programmed before the GET message is received. Use GET when the FG 5010 must execute operations in software synchronization with other instruments in the system. Such a situation might occur when a single output cycle is desired from the function generator at the same moment a DC 5009 counter or DM 5010 multimeter is to begin its measurement interval.

To set up the FG 5010 for DT TRIG and trigger a single pulse out with the GET message:

```
1000 PRINT @24:"INIT;OUT ON;MODE TRIG;DT TRIG"
1010 WBYTE @56,24,63:
```

Line 1000 sets up the FG 5010 at address 24 with default settings, output on, triggered mode, and trigger on receipt of GET.

Line 1010 sends the GET command.

**Status Reporting**

Some status bytes reported by the FG 5010 have special meaning in relation to FG 5010 functions, as shown in Table 5-2. The full set of status and error codes returned by the FG 5010 and other TM 5000 instruments is listed in the Appendix.

**TABLE 5-2  
SPECIAL FG 5010 STATUS BYTES**

CONDITION	NOT BUSY	BUSY
Generator went out of phase lock	202	218
Generator went into phase lock	206	222
Trigger input low	129	145
Trigger input toggling	130	146
Trigger input high	131	147
Phase lock mode, out of lock, trigger low	137	153
Phase lock mode, out of lock, trigger toggling	138	154
Phase lock mode, out of lock, trigger high	139	155
Phase lock mode, in lock, trigger toggling	142	158
Not in phase lock, trigger status not available mode	128	144

**FG 5010 Applications Programs**

The programs in this section feature special FG 5010 capabilities. The routines are written in 4052 basic as stand-alone programs or as subroutines which operate on predefined variables. The FG 5010 is assumed to be the only device on the bus and set to address 24.

## FG 5010 ERR? and Status Byte Message Table

This program provides the framework for the rest of the routines. Full decoding of status bytes and error messages is accomplished in the SRQ interrupt handler. Append other programs to line 1000.

Variables: B\$ = Error code messages  
 B1 = Status byte code table  
 A\$ = Error code printed message  
 E1, E2, E = Error code reporting variables  
 P,S = Serial Poll variables  
 B2 = Status byte code table read flag  
 I = Status byte table decode FOR NEXT loop

```

1 REM {FG 5010 ERR? and Status Byte MESSAGE TABLE}
100 INIT
110 B$=""
120 B2=0
130 ON SRQ THEN 50000
140 PAGE
1000 REM  append programs here for FG5010 at address 24.
49980 END
49990 REM ----- SRQ Interrupt Handler -----
50000 POLL P,S;24
50010 REM  parse the status byte message
50020 GOSUB 50090
50030 PRINT @24:"ERR?"
50040 INPUT @24:E
50050 REM  parse the ERR? message
50060 GOSUB 50580
50070 RETURN
50080 REM ----  ERR? and Status Byte message table -----
50090 IF B2 THEN 50160
50100 DIM B1(16,2)
50110 DATA 97,113,98,114,99,115,65,81,67,82,0,16,202,218,206,222,129,145
50120 DATA 130,146,131,147,137,153,138,154,139,155,142,158,128,144
50130 RESTORE 50110
50140 READ B1
50150 B2=1
50160 FOR I=1 TO 16
50170 IF S=B1(I,1) OR S=B1(I,2) THEN 50210
50180 NEXT I
50190 PRINT "_STB = ";S;"  Status Message Not FoundGG"
50200 RETURN
50210 PRINT " GG* INTERRUPT * ";
50220 GO TO I OF 50250,50270,50290,50310,50330,50350,50370,50390
50230 GO TO I-8 OF 50410,50430,50450,50470,50490,50510,50530,50550
50240 RETURN

```

```

50250 PRINT " COMMAND ERROR";
50260 RETURN
50270 PRINT " EXECUTION ERROR";
50280 RETURN
50290 PRINT " INTERNAL ERROR";
50300 RETURN
50310 PRINT " POWER UP";
50320 RETURN
50330 PRINT " USER REQUEST FOR SERVICE";
50340 RETURN
50350 PRINT " NOTHING TO REPORT";
50360 RETURN
50370 PRINT " PHASE LOCK MODE - GENERATOR OUT OF LOCK";
50380 RETURN
50390 PRINT " PHASE LOCK MODE - GENERATOR IN LOCK";
50400 RETURN
50410 PRINT " TRIGGER INPUT LOW";
50420 RETURN
50430 PRINT " TRIGGER INPUT TOGGLING";
50440 RETURN
50450 PRINT " TRIGGER INPUT HIGH";
50460 RETURN
50470 PRINT " PHASE LOCK MODE - OUT OF LOCK - TRIGGER LOW";
50480 RETURN
50490 PRINT " PHASE LOCK MODE - OUT OF LOCK - TRIGGER TOGGLING";
50500 RETURN
50510 PRINT " PHASE LOCK MODE - OUT OF LOCK - TRIGGER HIGH";
50520 RETURN
50530 PRINT " PHASE LOCK MODE - IN LOCK - TRIGGER TOGGLING";
50540 RETURN
50550 PRINT " NOT IN PHASE LOCK MODE - TRIGGER STATUS NOT AVAILABLE";
50560 RETURN
50570 REM ***** CODING FOR REPORTING ERROR? INFORMATION *****
50580 IF LEN(B$) THEN 51210
50590 DELETE B$
50600 DIM B$(2400)
50610 B$=" 101 Invalid Command Header"
50620 B$=B$&" 102 Header Delimiter Error"
50630 B$=B$&" 103 Command Argument Error"
50640 B$=B$&" 106 Missing Argument"
50650 B$=B$&" 107 Invalid Message Unit Delimiter"
50660 B$=B$&" 108 Binary Block Checksum Error"
50670 B$=B$&" 109 Binary Block Byte-count Error"
50680 B$=B$&" 201 Command Not Executable in Local state"
50690 B$=B$&" 202 Settings lost due to rtl"
50700 B$=B$&" 203 I/O Buffers full (output dumped to avoid deadlock)"
50710 B$=B$&" 205 Argument Out of Range"
50720 B$=B$&" 206 Group Execute Trigger ignored"
50730 B$=B$&" 251 Symmetry-Frequency Conflict"
50740 B$=B$&" 252 Amplitude-Offset Conflict"
50750 B$=B$&" 254 Hold Mode-Phase Lock Conflict"
50760 B$=B$&" 255 Hold Mode-Frequency Conflict"
50770 B$=B$&" 256 Phase Lock-FM Conflict"
50780 B$=B$&" 257 Phase Lock-VCF Conflict"
50790 B$=B$&" 258 Gate (ON/OFF)-Mode Conflict"

```



```

50800 B$=B$&" 301 Interrupt Fault"
50810 B$=B$&" 302 System Error"
50820 B$=B$&" 311 Period measurement failed to complete"
50830 B$=B$&" 312 Period measurement overflow occurred"
50840 B$=B$&" 313 Shift register (in VIA) failed to function"
50850 B$=B$&" 314 Mag-latch relay strobe interrupt failed to occur"
50860 B$=B$&" 315 Phase lock status over and under range"
50870 B$=B$&" 316 Automatic frequency correction range exceeded"
50880 B$=B$&" 320 Via fault on CPU board"
50890 B$=B$&" 321 Trig/Gate control error on CPU board"
50900 B$=B$&" 322 4 MHz Reference Frequency Clock or Counter Fault"
50910 B$=B$&" 323 Frequency Control Logic Fault on Loop 2 board"
50920 B$=B$&" 324 Loop cycle counter fault on Loop 2 board"
50930 B$=B$&" 325 Frequency prescaler fault on Loop 2 board"
50940 B$=B$&" 326 Low Frequency Prescaler Fault on Loop 2 board"
50950 B$=B$&" 327 No Signal Detected from Loop 1 board"
50960 B$=B$&" 328 Inadequate Frequency Range - 2 KHz range"
50970 B$=B$&" 329 Inadequate Frequency Range - 20 KHz range"
50980 B$=B$&" 330 Inadequate Frequency Range - 200 KHz range"
50990 B$=B$&" 331 Inadequate Frequency Range - 2 MHz range"
51000 B$=B$&" 332 Inadequate Frequency Range - 20 MHz range"
51010 B$=B$&" 333 Burst Counter Fault"
51020 B$=B$&" 334 Offset Generator Fault"
51030 B$=B$&" 335 Amplitude DAC Error"
51040 B$=B$&" 336 Amplitude Attenuator Error"
51050 B$=B$&" 337 Waveform Shaping Error"
51060 B$=B$&" 338 Normal/Complement Error"
51070 B$=B$&" 339 Low Frequency Generator DAC Error"
51080 B$=B$&" 340 Faulty RAM found (U1400)"
51090 B$=B$&" 341 Faulty RAM found (U1500)"
51100 B$=B$&" 350 Faulty RAM found in uP chip"
51110 B$=B$&" 370 ROM Placement Error at address A000"
51120 B$=B$&" 372 ROM Placement Error at address C000"
51130 B$=B$&" 374 ROM Placement Error at address E000"
51140 B$=B$&" 390 ROM Checksum Error at address A000"
51150 B$=B$&" 392 ROM Checksum Error at address C000"
51160 B$=B$&" 394 ROM Checksum Error at address E000"
51170 B$=B$&" 401 Power On"
51180 B$=B$&" 403 User Request (via INST ID button)"
51190 B$=B$&" 731 Generator went out of phase lock"
51200 B$=B$&" 732 Generator went into phase lock_"
51210 A$=STR(E)
51220 A$=A$&" "
51230 E1=POS(B$,A$,1)
51240 E2=POS(B$," ",E1)
51250 A$=SEG(B$,E1,E2-E1+1)
51260 A$=" CODE"&A$
51270 PRINT A$
51280 RETURN

```

Any of the other programs listed below can be appended into this program at line 1000. The result is a complete program with interrupt handling for the FG 5010. All programs assume the FG 5010 is set to primary address 24 as shipped from the factory.

### FG 5010 Software Frequency Sweep

This program sweeps the FG 5010 frequency in decades between a predetermined START and STOP frequency (150 Hz and 1.2 MHz in this example). Two FOR/NEXT loops are nested in lines 1090 to 1190 to control the FG 5010 frequency.

Variables: F1 = START frequency (50 Hz)  
 F2 = STOP frequency (1.2 MHz)  
 D = Number of frequency increments per decade  
 W = Seconds of wait interval at each frequency increment  
 F8 = START frequency decade (1=10 to 100, 2=100 to 1000, etc.)  
 F9 = Stop frequency decade  
 D0 = Frequency increment step size  
 D1 = Last frequency step in each decade

```

1000 REMARK S/W FREQUENCY SWEEP
1010 F1=50
1020 F2=1200000
1030 D=5
1040 W=0.2
1050 F8=1
1060 F9=6
1070 D0=9/D
1080 D1=10-D0
1090 PRINT @24:"FREQ ";F1
1100 FOR E=F8 TO F9
1110 FOR G=1 TO D1 STEP D0
1120 F=G*10^E
1130 IF F<F1 THEN 1170
1140 IF F>F2 THEN 1190
1150 CALL "WAIT",W
1160 PRINT @24:"FREQ ";F
1170 NEXT G
1180 NEXT E
1190 PRINT @24:"FREQ ";F2
1200 END

```

Lines 1000 to 1050 -- Set variables. Make changes here for desired start, stop, increment, etc.

Lines 1060 to 1070 -- Compute steps within decades. The "9" follows from the concept of a decade: 10 to 90, 100 to 900, etc.

Line 1080 -- A numeric argument is sent with a literal string instrument command for the start frequency.

Line 1090 -- Increment through the decades.

Line 1100 to 1160 -- Increment through the decade. Skip frequency steps below the start frequency (line 1120). Jump out of the loop above the stop frequency (line 1130). Wait as desired at line 1140.

Line 1180 -- Send the stop frequency.

#### FG 5010 VCF Mode Voltage Calculation

The FG 5010 Voltage Controlled Frequency mode provides a means of sweeping the frequency via an external voltage input. The FG 5010 frequency control circuit utilizes ranges for frequency setting purposes. The top of the frequency ranges are: 20 MHz, 2 MHz, 200 kHz, 20 kHz, 2 kHz, 200 Hz, 20 Hz, 2 Hz, .2 Hz, and .02 Hz. VCF mode prevents further frequency range switching and allows selection of  $f=0$  Hz. Therefore, an appropriate range must be selected before entering the VCF mode.

This program prompts the user for a desired STOP and START frequency and then calculates the VCF sensitivity and voltage which must be applied. The FG 5010 is programmed for the STOP frequency in order to establish the range. Then the VCF mode is turned ON and the START frequency is programmed.

FG5010 VCF MODE Voltage Calculation

---

ENTER STOP FREQUENCY 45000

ENTER START FREQUENCY 1000

START FREQ. =1000            STOP FREQ. =45000

Freq Range = 200000 Hz      VCF Sensitivity = 20000 Hz/V

Apply 2.199 volts.

---

ENTER STOP FREQUENCY 19E+6

ENTER START FREQUENCY 100E+3

START FREQ. =100000        STOP FREQ. =1.9E+7

Freq Range = 2.0E+7 Hz      VCF Sensitivity = 2000000 Hz/V

Apply 9.449 volts.

---

ENTER STOP FREQUENCY

Fig. 5-1. Display for several runs of program that does VCF mode voltage calculation.

- Variables:
- F2 = STOP frequency
  - F1 = START frequency
  - G1 = Maximum frequency of range
  - Z = VCF sensitivity for that range
  - U = Frequency difference between F2 and F1
  - V = Volts to apply to VCF input.

```

1000 PAGE
1010 PRINT "IFG5010 VCF MODE Voltage Calculation"
1020 PRI "J" -----"
1030 PRINT " ENTER STOP FREQUENCY ";
1040 INPUT F2
1050 IF F2<=2.0E+7 AND F2=>0.002 THEN 1080
1060 PRINT " STOP FREQUENCY Out of Range (.002 to 20 MHz)GGGG"
1070 GO TO 1020
1080 PRINT " ENTER START FREQUENCY ";
1090 INPUT F1
1100 REM FIND RANGE
1110 G1=2*10^INT(LGT(F2+1.0E-6))
1120 REM TEST IF STOP FREQ < RANGE
1130 IF F2<G1 THEN 1150
1140 G1=G1*10
1150 Z=G1/10
1160 REM Z=v/Hz (range G1)...U=FREQ. DIFFERENTIAL...V=APPLIED VOLTS
1170 U=F2-F1
1180 V=U/Z
1190 PRINT " START FREQ. =";F1;" STOP FREQ. =";F2
1200 PRINT " Freq Range = ";G1;" Hz VCF Sensitivity = ";Z;" Hz/V"
1210 PRINT " Apply ";INT(V*1000)/1000;" volts."
1220 PRINT @24:"VCF OFF;FREQ ";F2
1230 PRINT @24:"VCF ON;FREQ ";F1
1240 CALL "wait",0.5
1250 GO TO 1020

```

Lines 1030 to 1090 -- Input and check start and stop frequencies.

Lines 1110 to 1140 -- Compute top of range that is required.

Lines 1150 to 1210 -- Compute VCF signal required and inform the operator.

Line 1220 -- Turn off VCF mode and program the STOP frequency (automatic range selection).

Line 1230 -- Turn on VCF mode and program START frequency (actual frequency with zero volts applied).

Line 1240 -- Wait for an interrupt; this could occur if START frequency is greater than STOP frequency. (CALL "WAIT" is not available on the 4051.)

## FG 5010 Pulse Generator Emulator

The FG 5010 square wave functions can be used to fill a variety of pulse generator applications. Period, duration, amplitude, offset, and complement are among the parameters of interest in specifying pulse parameters. FG 5010 instrument commands "FREQ", "SYM", "AMPL", "OFFS", and "COMP" specify pulse parameters in function generator terms. "FREQUENCY" and "SYMMETRY" are alternate ways to specify period and duration. "AMPLITUDE" and "OFFSET" specify peak-to-peak amplitude and DC offset. Frequently however, pulse parameters specify p-p amplitude and negative peak voltage, or positive peak and negative peak voltage.

The FG 5010 Pulse Generator Emulator program converts the user's choice of terminology to instrument commands the FG 5010 can understand. A menu of parameter choices is provided for the user to select from. Figure 5-2 is a hard copy of the 4052 display. The prompts indicate allowable ranges.

## FG5010 PULSE GENERATOR EMULATOR

**PARAMETERS:**

- 1) PERIOD & DURATION
- 2) REPETITION & DUTY CYCLE
- 3) AMPLITUDE & OFFSET
- 4) NEG PEAK & POS PEAK
- 5) COMPLEMENT (SYMMETRY)

## ENTER PARAMETER SELECTION -

```

#1 Enter PERIOD in seconds (200 sec. to 50 ns) : 0.2
   DURATION Range = 0.02 to 0.18
   Enter DURATION: .05

#2 Enter REPETITION RATE in Herz (.002 to 20 MHz): 15E+6
   DUTY CYCLE Range = 38 to 61
   Enter DUTY CYCLE : 40

#3 Enter P-P AMPLITUDE in volts (20v max open ckt): 18
   Negative Peak Baseline Offset range = -15 volt to -3 volts.
   Enter OFFSET: -10

#4 Enter NEGATIVE PEAK in volts (-15 to +7.5): -10
   POSITIVE PEAK range = -10 to 10 volts.
   Enter POSITIVE PEAK in volts: +8

#5 COMPLEMENT SYMMETRY
  
```

Fig. 5-2. Display provided by pulse generator emulator program.

GPIB PROGRAMMING GUIDE

Symmetry range is 10% to 90% up to 4 MHz frequency. The range reduces to a fixed 50% at 20 MHz. This reduction in symmetry range is factored into the program for items 1 and 2 on the menu. Minimum pulse duration is limited to the greater of 25.5 ns or 10% (conversely 90%) symmetry. Maximum pulse duration is limited to period--25.5 ns or 90% symmetry, whichever is least.

Each item in the menu is a separate subroutine which stands alone.

```

1000 REM  FG5010  PULSE GENERATOR EMULATOR
1010 PAGE
1020 PRINT @24:"INIT;FUNC SQUARE;OUT ON;"
1030 PRINT " IFG5010 PULSE GENERATOR EMULATOR"
1040 PRINT "JJJPARAMETERS: K"
1050 IMAGE 15T,30A
1060 PRINT USING 1050:"1) PERIOD & DURATION"
1070 PRINT USING 1050:"2) REPETITION & DUTY CYCLE"
1080 PRINT USING 1050:"3) AMPLITUDE & OFFSET"
1090 PRINT USING 1050:"4) NEG PEAK & POS PEAK"
1100 PRINT USING 1050:"5) COMPLEMENT (SYMMETRY)"
1110 PRINT "JJJENTER PARAMETER SELECTION - ";
1120 INPUT N
1130 PRINT "J"
1140 IF ABS(N)<1 OR ABS(N)>5 THEN 1040
1150 GOSUB N OF 1180,1360,1530,1750,1960
1160 PRINT "JJ"
1170 GO TO 1040
1180 REM ----- 1) PERIOD -----
1190 PRINT "J      #1 Enter PERIOD in seconds (200 sec. to 50 ns) : ";
1200 INPUT P
1210 IF 1/P<0.002 OR 1/P>2.0E+7 THEN 1340
1220 D1=P*0.9
1230 IF P-D1>2.55E-8 THEN 1250
1240 D1=P-2.55E-8
1250 PRINT "J          DURATION Range = ";P*0.1 MAX 2.55E-8;" to ";D1
1260 PRINT "J          Enter DURATION: ";
1270 INPUT D
1280 IF D=>2.55E-8 AND D<=D1 THEN 1310
1290 PRINT "J          DURATION out of range.GGG"
1300 RETURN
1310 PRINT @24:"FREQ ";1/P;" ;SYM ";100*(D/P)
1320 CALL "WAIT",0.5
1330 RETURN
1340 PRINT "JPERIOD OUT OF RANGE.GGG"
1350 RETURN

```

GPIB PROGRAMMING GUIDE

```

1360 REM ----- 2) REPETITION RATE -----
1370 PRINT "J      #2  Enter REPETITION RATE in Herz (.002 to 20 MHz): ";
1380 INPUT F
1390 P1=1/F
1400 D1=2.55E-8
1410 D2=P1-2.55E-8
1420 E1=INT(100*(D1/P1)) MAX 10
1430 E2=INT(100*(D2/P1)) MIN 90
1440 PRINT "J      DUTY CYCLE Range = ";E1;" to ";E2
1450 PRINT "J      Enter DUTY CYCLE : ";
1460 INPUT C
1470 IF NOT(C=>E1 AND C<=E2) THEN 1510
1480 PRINT @24:"FREQ ";F;" ;SYM ";C
1490 CALL "WAIT",0.5
1500 RETURN
1510 PRINT "J      DUTY CYCLE out of range.GG"
1520 RETURN
1530 REM ----- 3) AMPLITUDE -----
1540 PRI "J      #3  Enter P-P AMPLITUDE in volts (20v max open ckt): ";
1550 INPUT V
1560 V=ABS(V)
1570 IF V<=20 THEN 1600
1580 PRINT "J      AMPLITUDE out of range (0 to 20 v P-P)GG"
1590 RETURN
1600 PRINT "J      Negative Peak Baseline Offset range = ";
1610 IF 7.5+V/2<15 THEN 1640
1620 V2=15-V
1630 GO TO 1650
1640 V2=7.5-V/2
1650 PRINT " -15 volt to ";V2;" volts."
1660 PRINT "J      Enter OFFSET: ";
1670 INPUT O
1680 IF O>=-15 AND O<=V2 THEN 1710
1690 PRINT "J      OFFSET out of range.GG"
1700 RETURN
1710 O1=V/2+O
1720 PRINT @24:"AMPL ";V;" ;OFFS ";O1
1730 CALL "WAIT",0.5
1740 RETURN
1750 REM ----- 4) NEG PEAK & POS PEAK -----
1760 PRINT "J      #4  Enter NEGATIVE PEAK in volts (-15 to +7.5): ";
1770 INPUT V1
1780 IF V1=>-15 AND V1<=7.5 THEN 1810
1790 PRINT "J      NEGATIVE PEAK out of range (-15 to +7.5)GG"
1800 RETURN
1810 IF V1<0 THEN 1840
1820 V2=15-V1
1830 GO TO 1850
1840 V2=V1+20
1850 PRINT "J      POSITIVE PEAK range = ";V1;" to ";V2;" volts."
1860 PRINT "J      Enter POSITIVE PEAK in volts: ";
1870 INPUT V3
1880 IF V3<=V2 AND V3>V1 THEN 1910
1890 PRINT "J      POSITIVE PEAK out of range.GGG"
1900 GO TO 1850

```



```

1910 V=V3-V1
1920 O1=V1+V/2
1930 PRINT @24:"AMPL ";V;" ;OFFS ";O1
1940 CALL "WAIT",0.5
1950 RETURN
1960 REM ----- 5) COMPLEMENT SYMMETRY -----
1970 PRINT "J      #5 COMPLEMENT SYMMETRY"
1980 PRINT @24:"COMP?"
1990 INPUT @24:S$
2000 IF POS(S$,"ON",1) THEN 2030
2010 PRINT @24:"COMP ON"
2020 RETURN
2030 PRINT @24:"COMP OFF"
2040 RETURN

```

Subroutine #1: Convert period and duration to frequency and symmetry.

Variables: P = Period  
D = Duration  
D1 = maximum positive pulse width.

Line 1310 -- Note that the semicolon message unit delimiter is required after the FREQ argument and before the SYM header. The semicolon is inside the quotes and is sent as an ASCII character. The other semicolons in the print statement are 4052 syntax elements.

Subroutine #2: Convert repetition rate and duty cycle to frequency and symmetry.

Variables: F = Repetition rate  
E1 = Minimum duty cycle  
E2 = Maximum duty cycle  
C = Duty cycle  
D1 = Minimum duration  
D2 = Maximum duration

**Subroutine #3:** Convert p-p amplitude and negative pulse peak to amplitude and dc offset.

Variables: V = p-p amplitude  
 V2 = Upper end of negative pulse peak voltage range  
 0 = Negative pulse peak voltage  
 01 = dc offset.

**Subroutine #4:** Convert negative pulse peak and positive pulse peak voltages to p-p amplitude and dc offset.

Variables: V1 = Negative pulse peak voltage  
 V2 = Maximum positive pulse peak voltage  
 V3 = Positive pulse peak voltage  
 V = p-p pulse amplitude  
 01 = dc offset voltage.

**Subroutine #5:** Complement symmetry (toggle current COMP setting).

Variables: \$\$ = Current COMPLEMENT setting.

## FG 5010 LEARN MODE -- Create Settings Different From INIT

This program uses the FG 5010 SET? to learn the current front panel setup. The ASCII data is read into a string variable and consists of about 182 characters. Frequently however, only a few parameters may be different from the default front panel settings of the "INIT" command. In this case, it is convenient to send "INIT" followed by the settings which differ. The result is a front panel setting message of only 20 or 30 characters rather than the full 182 or more.

This program displays the INIT settings and the current front panel settings and then creates a string consisting of "INIT;...difference settings" which is much shorter. The "difference" settings can then be sent back to reprogram the function generator. Use this technique to reduce the number of program lines necessary to define a full front panel setup. This also conserves computer memory by compressing the instrument front panel setup data to a minimum.

Variables: I\$ = INIT settings  
 N = Position in I\$ of ";" characters  
 R\$ = Response from FG 5010 to SET? command  
 S\$ = "INIT;" + message units in R\$ which differs from I\$  
 T\$ = Message unit in R\$  
 U\$ = Message unit in I\$, same header as T\$ but  
 default argument.

```

1000 REM FG 5010 LEARN MODE--CREATE SETTINGS DIFFERENT FROM INIT
1010 PAGE
1020 PRI "FG5010 LEARN SETTINGS - Create Settings Different from INIT"
1030 PRINT "JJ INIT"," NEWJ"
1040 DIM I$(200),R$(200),S$(200),N(20)
1050 S$=""
1060 I$="FREQ 1.0E+3;AMPL 500.0E-3;OFFS 0.0;SYM 50;PHASE 0;"
1070 I$=I$&"NBUR 10;FUNC SINE;MODE CONT;SLOPE POS;OUT OFF;COMP OFF;"
1080 I$=I$&"AM OFF;FM OFF;VCF OFF;HOLD OFF;GATE OFF;PLI OFF;DT OFF;"
1090 I$=I$&"USER OFF;RQS ON;"
1100 DATA 0,13,28,38,46,55,64,74,84,94,102,111,118,125,133,142,151,159
1110 DATA 166,175,182
1120 RESTORE 1100
1130 READ N
1140 PRINT @24:"SET?"
1150 INPUT @24:R$
1160 N1=0
1170 N0=1
1180 FOR I=1 TO 20

```

```

1190 N1=POS(R$,";",NO+1)
1200 T$=SEG(R$,NO,N1-NO+1)
1210 NO=N1+1
1220 IF I<>20 THEN 1250
1230 U$=SEG(I$,N(I)+1,LEN(I$)-N(I))
1240 GO TO 1260
1250 U$=SEG(I$,N(I)+1,N(I+1)-N(I))
1260 PRINT U$,T$
1270 IF U$=T$ THEN 1290
1280 S$=S$&T$
1290 NEXT I
1300 S$="INIT;"&S$
1310 PRINT " DIFFERENCE SETTINGS ARE :J"
1315 PRINT S$
1320 PRINT "JPress RETURN to send only these Difference Settings back."
1330 INPUT A$
1340 PRINT @24:S$
1350 END

```

The FOR-NEXT loop in lines 1180 to 1290 sequentially compares message units in I\$ with the same message unit in R\$. The headers should be the same but arguments may be different.

### Binary Settings Query

The FG 5010 binary settings data consists of only 50 bytes of data to describe the full front panel setup. Fewer bytes means less time to handshake data between the controller and the FG 5010. Since binary data is assumed to have been originally created by the FG 5010, minimal time is spent by the FG 5010 processor in checking out the settings data before updating hardware. This means higher throughput rates when sending binary settings.

This program acquires the current front panel binary block setup in a numeric array R. This data is position dependent and should not be altered in any way before being sent back to the FG 5010.

```
1000 REM    binary settings query
1010 PAGE
1020 PRINT "IFG5010 LOW LEVEL SETTINGS QUERY_"
1030 DIM R(50)
1040 PRINT @24:"LLSET?"
1050 WBYTE @63,95,88:
1060 RBYTE R
1070 WBYTE @62,95:
1080 PRINT "FG5010 LOW LEVEL SET DATA FOLLOWS:"
1090 PRINT R;
1100 PRI "JJJPress RETURN to send Low Level Setting back to the FG5010."
1110 INPUT A$
1120 WBYTE @63,95,56:R
1130 WBYTE @63,95:
1140 END
```

Line 1040 -- Send "LLSET?" query.

Line 1050 -- Send UNTALK, UNLISTEN, TALK address.

Line 1060 -- Handshake binary settings into array R.

Line 1070 -- Send UNTALK, UNLISTEN.

Line 1120 -- Send UNTALK, UNLISTEN, listen address, array R to the FG 5010.

Line 1130 -- Send UNLISTEN, UNTALK.

## Low-Level Stored Settings Query

This program allows the user to learn the settings data in any of ten registers in the FG 5010. The "SEND n" command instructs the FG 5010 which register to send back in binary block format. The binary data consists of 52 bytes per register. These are stored in array R. Program execution is the same as #6, only the query is different. The data is position dependent and should not be altered in any way before being sent back to the FG 5010.

```

1000 REM    LOW LEVEL  STORED SETTINGS QUERY
1010 DIM R(52)
1020 PAGE
1030 PRINT "IFG5010 LOW LEVEL STORED SETTINGS QUERY"
1040 PRINT "____ENTER Register # for Stored Setting Query - ";
1050 INPUT N
1060 PRINT @24:"SEND ";N
1070 CALL "WAIT",0.5
1080 WBYTE @63,95,88:
1090 RBYTE R
1100 WBYTE @63,95:
1110 PRINT "____Low Level Stored Settings  Register ";N;" Data:"
1120 PRINT R;
1130 PRI "____Press RETURN to re-store the Low Level Data in Register ";N
1140 INPUT A$
1150 WBYTE @63,95,56:R
1160 WBYTE @63,95:
1170 END

```

Line 1060 -- Send the "SEND n" command.

Line 1080 to 1100 -- Acquire binary bytes in array R.

Lines 1150 to 1160 -- Send array R binary block data back to the FG 5010.

The first part of the report is a general overview of the project. It describes the objectives, the scope of the work, and the organization of the report. The second part is a detailed description of the methodology used in the study. This includes a discussion of the data collection methods, the statistical analysis techniques, and the software used for data processing. The third part presents the results of the study, which are discussed in the context of the research objectives. The final part of the report is a conclusion and a list of references.

The methodology section is divided into two main parts: data collection and data analysis. In the data collection section, we describe the sampling method used to select the participants for the study. We also discuss the instruments used to collect data, including questionnaires and interviews. In the data analysis section, we describe the statistical methods used to analyze the data, including descriptive statistics, inferential statistics, and regression analysis. We also discuss the software used for data processing, including SPSS and Excel.

The results section is divided into two main parts: descriptive statistics and inferential statistics. In the descriptive statistics section, we present the mean, standard deviation, and range of the variables measured in the study. We also present the frequency distribution of the variables. In the inferential statistics section, we present the results of the statistical tests used to compare the groups in the study. We also discuss the implications of the results for the research objectives.

The conclusion section summarizes the main findings of the study and discusses their implications for practice and research. We also provide a list of references for the sources used in the study. The references are listed in alphabetical order and include books, journal articles, and web pages. The list of references is as follows:

- 1. Smith, J. (2005). The impact of stress on performance. *Journal of Applied Psychology, 90*(1), 1-10.
- 2. Johnson, M. (2007). The effects of sleep deprivation on cognitive function. *Journal of Experimental Psychology: Applied, 13*(1), 1-10.
- 3. Brown, K. (2008). The relationship between stress and health. *Journal of Health Psychology, 27*(1), 1-10.
- 4. Davis, L. (2009). The impact of stress on the immune system. *Journal of Behavioral Medicine, 32*(1), 1-10.
- 5. White, R. (2010). The effects of stress on the cardiovascular system. *Journal of Hypertension, 28*(1), 1-10.

## SECTION 6

## PS 5010 PROGRAMMABLE POWER SUPPLY

The PS 5010 is a microprocessor based, triple power supply designed to operate in a TM 5000 power module. It provides a complete and rapid, high performance solution for many GPIB system power supply applications, three independently programmable supplies with a wide range of voltage and current capabilities, simulated DMM displays coupled with automatic crossover to provide visual feedback of the known output and regulation status.

In addition to the visual regulation feedback, the PS 5010 can report its regulation status by sending SRQs and status bytes over the bus. This is important for program branching and decision making.

All front panel functions are programmable over the GPIB and their status can be sent to the controller in learn mode command-argument format.

Additional remotely programmable features are:

- \* Independent output control for the DUAL FLOATING and LOGIC supplies
- \* Programmable user generated interrupts
- \* Responds to variety of interface control messages
- \* Error/status reporting
- \* Low level (binary) setting capability
- \* SRQ disabling



PS 5010 FUNCTIONS  
 POSITIVE and NEGATIVE FLOATING SUPPLIES

Constant voltage mode:		
Range		
Positive supply	0 to +32.0 V	
Negative supply	0 to -32.0 V	
Step size (resolution)	10 mV <u>+10</u> mV to 10.0 V 100 mV <u>+40</u> mV above 10 V	
Overall accuracy (total effect)	<u>+(0.5% + 20 mV)</u>	
Voltage change response time:	<b>No load</b>	<b>Max load</b>
Up	1 ms	1 ms
Down	20 ms	1 ms
Constant current mode:		
Range		
High power compartment	50 mA to 0.750 A (1.60 A at 15 V and below)	
Standard compartment	50 mA to 400 mA (0.750 A at 15 V and below)	
Step size (resolution)	50 mA <u>+15</u> mA	
Overall accuracy	<u>+(5% + 20 mA)</u>	
Current change response time:		
Up	20 mS	
Down	20 mS	

Programming time:  Group Execute Trigger (GET)  Without output on/off change With output on/off change	10 mS typical  30 mS typical
---	------------------------------------

LOGIC SUPPLY

Constant voltage mode:  Voltage range Voltage step size Overall accuracy	4.50 to 5.50 V 10 mV <u>+10</u> mV <u>+50</u> mV
Current limit:  Range Step size Accuracy	100 mA to 3.0 A 100 mA <u>+30</u> mA <u>+(5% + 20</u> mA)
Programming time:  Without on/off change With on/off change	3 mS typical 35 mS typical

## GPIB Operation

## IEEE Bus Address

Pressing the INST ID button will display the PS 5010's primary address. A decimal following the address indicates that the message terminator is set to EOI/LF.

The PS 5010 address switch, which is located on the CPU board, is factory set to primary address 22 and terminator to EOI only. Secondary address capability is not used by the PS 5010.

The PS 5010 GPIB address and message terminator can be reset by a qualified service person. Binary addresses between 0 and 31 are valid primary addresses, however, setting the primary address to 31 causes the PS 5010 to ignore GPIB commands (31+32=63=UNLISTEN) and many controllers, including the 4050 Series Controllers, reserve zero for themselves.

## Power-Up Conditions

When powered up, the PS 5010 performs a diagnostic, self-test to check the ROM and RAM functionality. When an error is found during this procedure an SRQ will occur and the error code will be displayed in the LOGIC display (see Appendix for error codes). If there are no internal errors, the instrument enters the LOCAL state (LOCS) with the default settings and the SRQ line is asserted.

TABLE 6-1

POWER-UP DEFAULT SETTINGS

Current

INEG	0.4
IPOS	0.4
ILOG	1.0

Voltage

VNEG	0.0
VPOS	0.0
VLOG	5.0

Output

LSOUT	OFF
FSOUT	OFF

Interrupt Conditions

RQS	ON
USER	OFF
NRI	OFF
PRI	OFF
LRI	OFF

Device Trigger Condition

DT	OFF
----	-----

## Programming PS 5010 Functions

## Initialized Settings

The PS 5010 can be initialized to its power-on state via the INIT command.

EXAMPLE: PRINT @22:"INIT"

"INIT" sent to an instrument as is a device dependent command only initializes the instrument that has been addressed.

## Programming Voltage and Current Limits

Voltage and current limits are programmed using the commands ITRA, VTRA, INEG, VNEG, IPOS, VPOS, ILOG, and VLOG (I=current; V=voltage) followed by a numeric argument.

To program the PS 5010 negative and positive supplies for voltage limits of 15 volts and current limits of 400 mA:

EXAMPLE (1): PRINT @22:"VPOS 15;VNEG 15;IPOS .4;INEG .4"

EXAMPLE (2): PRINT @22:"VTRA 15;ITR .4"

EXAMPLE (3): PRINT @22:"INIT;VTRA 15"

Example (1) illustrates a literal interpretation of the problem.

Example (2) accomplishes the same result using the Dual Floating supplies tracking capability.

Example (3) initializes the instrument to a known state which includes currents set to 400 mA and then sets the voltage using the tracking capability.

### Incrementing Voltages and Currents

To increment voltage and current limits over the GPIB, any of several BASIC incrementing methods may be used. For instance, FOR/NEXT looping can increment a ramp.

To increment the POSITIVE voltage from 5 to 12 volts in 100 mV steps:

```
300 FOR V1=5 TO 12 STEP .1
310 PRINT @22:"VPOS ";V1
320 CALL "WAIT",0.01
330 NEXT V1
```

Line 300 -- Controls the number of times the loop is executed (5 to 12 volts in 100 mV steps).

Line 310 -- Addresses the PS 5010 to set its POSITIVE output voltage to the value in V1.

Note the syntax: the statement VPOS is followed by a space inside the quotes and the argument (V1) is outside the quotes. The semi-colon suppresses the carriage return, which syntactically positions the argument (V1) after the space.

Line 320 -- Approximates the delay for output settling time. Line 320 can be adjusted or removed according to the settling time required for the application.

Line 330 -- Returns program to line 300 until loop execution is complete.

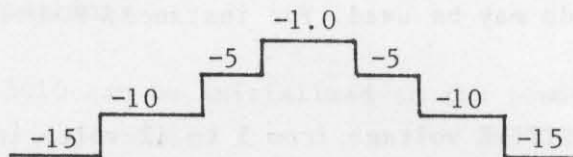
Note: to reverse the incrementing process (12 to 5 volts), change line 300 to:

```
300 FOR V1=12 TO 5 STEP -.1
```

### Incrementing for Specified Random Values ("a Staircase")

The GO TO statement may be used in place of the FOR/NEXT loop as shown in the next example.

Here's an example that steps the NEGATIVE supply for the following waveform using programmed time delays.



```

400 REM *****
410 REM ***** STEPPED WAVEFORM *****
420 REM *****
430 INIT
440 ON SRQ THEN 1000
450 P=22
460 C=1
470 DIM V2(7,2)
480 DATA 15,0.01,10,0.05,5,0.1,1,0.1,5,0.5,10,0.1,15,0
490 RESTORE 480
500 READ V2
510 PRINT @P:"FSOUT ON;INEG .30"
520 PRINT @P:"VNEG ";V2(C,1)
530 CALL "WAIT",V2(C,2)
540 C=C+1
550 IF C=8 THEN 570
560 GO TO 520
570 GO TO 460
1000 POLL D,S;P
1010 RETURN
    
```

Lines 430 to 460 -- Initialize, set up SRQ and define variables.

Line 470 -- Dimensions the array V2 for seven voltage values (1-7,1) and seven time values (1-7,2).

Line 480 -- Specifies the voltage and time values in pairs in the order which they are to be executed.

Line 490 -- Restores the invisible data pointer to the first element in the data statement.

Line 500 -- Reads all DATA elements into array V2.

Line 510 -- Addresses the PS 5010 (P) to turn the floating supply output on and set the NEGATIVE current limit.

Line 520 -- Addresses the PS 5010 (P) to change its negative supply voltage to the "C" element in column "1" of array V2.

Line 530 -- Delays the loop time by the "C" element in column 2 of array V2.

Line 540 -- Increments C to the next pair.

Line 550 -- Checks C for end of loop execution at which time the program branches to Line 570.

Line 560 -- Repeats loop from line 520 until C=8.

Line 570 -- Causes the program to loop back to begin the cycle again.

#### Output Control

The dual floating supplies and the logic supply outputs can be switched on and off independently over the GPIB with the statements FSOUT <ON>or<OFF> for the dual floating supplies and LSOUT <ON>or<OFF> for the logic supply or simultaneously with the statement OUT <ON>or<OFF>.

For example, to set the logic supply to 5.3 volts, 1 amp, floating supplies output off, and logic supply output on:

```
PRINT @22:"INIT;VLOG 5.3;LSOUT ON"
```

INIT causes the instrument to return to a known state which includes turning the outputs off and setting the LOGIC supply current to 1 amp.

To set the POSITIVE supply to 15 volts, .5 amps, and the LOGIC supply to 5 volts, 2 amps, and turn the outputs on:

```
PRINT @22:"VPOS 15;IPOS .5;ILOG 2;OUT ON"
```



## Front Panel Lockout

Front-panel controls are locked out by sending the low level Local Lockout (LLO--17) command with the WBYTE statement. Since LLO is a universal GPIB command, all devices which are already in the remote state (REMS) will be affected. This command can only be sent under program control (with a line number) since the instruments automatically return to the LOCS when the REN line becomes unasserted.

EXAMPLE (1): <LINE NUMBER>WBYTE @17:

EXAMPLE (2): <LINE NUMBER>WBYTE @54,17,63:

Example (1) locks out all instruments on the bus which are in the remote state.

Example (2) accomplishes the same thing, however, it sends MLA 54 (22+32) to the PS 5010 prior to executing the LLO (17) message and then returns it to an UNLISTEN (63) state. This insures that the specified device (PS 5010) is in the remote state (REMS). The LLO message still offsets all devices on the bus which are in the remote state.

In remote with lockout state (RWLS), all front panel buttons which can cause a change in instrument settings are locked out. The PS 5010 locks out all front panel except the VOLTAGE and CURRENT parameter buttons and the INST ID button.

The PS 5010 can be returned to the local with lockout state (LWLS) by sending the low level addressed command GTL(1), however, it will return to RWLS (LLO) the next time it is addressed.

EXAMPLE: <LINE NUMBER>WBYTE @22+32,1:

It will only return to the local state (LOCS) when the REN line becomes unasserted by terminating program control (BREAK, END, STOP).

In the remote state (REMS) pressing any front panel button will return the PS 5010 to LOCS.

## Regulation Status Reporting

Regulation status reporting is available both at the front panel and over the GPIB.

From the front panel the displays will always indicate the actual known output parameter and value because of a combination of microprocessor feedback, simulated DMM displays, and automatic crossover features. This means, whenever a load change causes a supply to change from constant voltage to constant current, or vice versa, the display will also change to indicate the actual known output value and regulation mode of the supply. If the load change causes the supply to unregulate, the display will blank indicating that actual output of the supply is unknown.

The same changes visible at the front panel can be detected over the GPIB either by enabling the PS 5010 regulation interrupt capabilities to report the changes when they occur or by querying the supplies at specific points in a program.

**Query All Supplies.** The regulation query (REG?) is used to query the PS 5010 for the regulation status of all of its supplies. The response to this command is REG<num>,<num>,<num>. The <num> will be a 1, 2, or 3 depending on the regulation status of each supply.

- 1 = VOLTAGE REGULATION MODE
- 2 = CURRENT REGULATION MODE
- 3 = UNREGULATED MODE

The order of the response follows the display (NEGATIVE, POSITIVE, LOGIC).

The response to REG? where the NEGATIVE supply is in current regulation, the POSITIVE supply is in voltage regulation, and the LOGIC supply is unregulated, would be:

REG 2,1,3

**Regulation Interrupt Reporting.** The regulation reporting capability is independently enabled for each supply via the commands NRI, PRI, and LRI, followed by an ON or OFF argument.

When the regulation interrupt is enabled, an SRQ will be reported if a load change causes a supply to change from CV to CC or visa versa, or when it goes into an unregulated state.

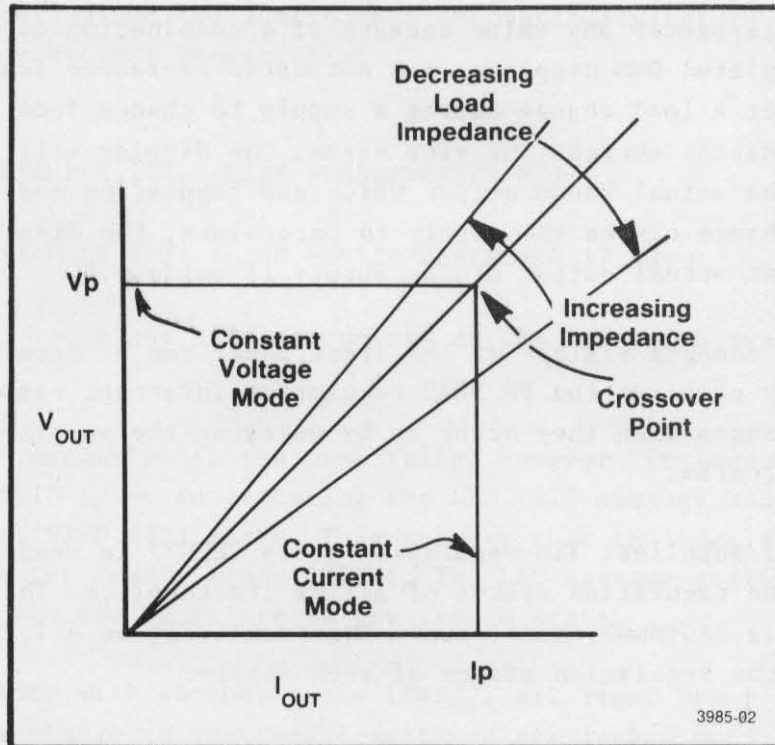


Fig. 6-1. Load lines for individual load impedances.

Since there are three possible regulation conditions for each supply (voltage regulated, current regulated, and unregulated) the SRQ will report a different status byte and/or error code (see below) for each mode change. RQS must be ON for all SRQ reporting capabilities.

STATUS BYTE	ERROR CODE	
197	721	Neg. change to CV
198	722	Neg. change to CC
199	723	Neg. change to Unreg.
201	724	Pos. change to CV
202	725	Pos. change to CC
203	726	Pos. change to Unreg.
205	727	Log. change to CV
206	728	Log. change to CC
207	729	Log. change to Unreg.

Here's an example that tests an overvoltage protection circuit to current limit @ 50 mA when the voltage exceeds 18 volts. It checks the circuit to 18 volts +10% (16.2-19.8) and indicates whether the circuit "passed" or "failed" at +5% of 18 volts.

```

600 REM
610 REM           {PS 5010 REGULATION AND USER}
620 REM           {INTERRUPT REPORTING}
630 REM
640 INIT
650 ON SRQ THEN 1000
660 P=22
670 PRINT @P:"INIT;IPOS .05;USER ON"
680 PRINT "JJPRESS INST ID BUTTON TO BEGIN TEST"
690 WAIT
700 IF S<>67 THEN 690
710 PRINT @P:"VPOS 0;PRI ON;FSOUT ON"
720 FOR V3=16.2 TO 19.8 STEP 0.1
730 PRINT @P:"VPOS ";V3
740 CALL "WAIT",0.11
750 IF S=202 THEN 770
760 NEXT V3
770 PRINT @P:"VPOS?"
780 INPUT @22:V4
790 IF V4>17.1 AND V4<18.9 THEN 830
800 PRINT "J          OVERVOLTAGE TEST FAILED @";V4;" VOLTS"
810 PRINT @P:"PRI OFF;FSOUT OFF"
820 GO TO 680
830 PRINT "J          OVERVOLTAGE TEST PASSED @";V4;" VOLTS"
840 PRINT @P:"PRI OFF;FSOUT OFF"
850 GO TO 680
1000 REM ***** POLL ROUTINE *****
1010 POLL D,S;P
1020 RETURN
    
```

Lines 640 to 660 -- Initialize, set up on SRQ and defines variables.

Line 670 -- Addresses the PS 5010 to initialize its settings, set the positive supply current and turn the user generated interrupt on.

Line 690 -- Causes the program to wait for an interrupt to occur.

Line 700 -- Checks the status byte(s) for 67 which signifies that the INST ID button has been pressed. If S is not equal (<>) to the user requested interrupt then go back to line 690.

Line 710 -- Sets the positive voltage, turns on the regulation reporting and turns on the floating supply outputs.

Line 720 -- Establishes the ramping limits of T and 10% of 18 volts.

Line 730 -- Programs the PS 5010 to the value in V3 which is incremented in 100 mV steps until the overvoltage protection circuit turns on or V3=19.9 (one step beyond the end of the loop).

Line 740 -- Establishes approximately a 110 mS delay for a regulation interrupt to be detected.

Line 750 -- Checks the variable S to determine whether the interrupt (202) that occurs when the positive supply changes to a current regulated status. If S=202 the program jumps out of the loop; otherwise it goes to the NEXT V3.

Lines 770 to 790 -- Queries the PS 5010 for the present programmed value of the positive supply and puts the response in V4. The program then checks V4 for greater than the lower 5% limit or less than the upper 5% limit and if it's within the set limits branches to line 830 and reports that the test "passed." If V4 is not within the limits the program falls through and reports that the test failed.

Lines 810 to 840 -- Handle turning the positive regulation interrupt and floating supply outputs off before looping back to line 680 to begin another test.

Low-Level Setting

Low-level setting capability allows rapid PS 5010 set-ups under program control.

Sending the low level settings command, "LLSET <binary block>", causes the PS 5010 to change all of its settings to the states specified in the binary block argument.

To acquire the binary block argument, send the ASCII command LLSET? and store the response in an array variable dimensioned for exactly 26 elements using the low levels 4050 BASIC I/O statements WBYTE and RBYTE:

```

100 REM *****
110 REM *  LOW LEVEL SETTINGS  *
120 REM *****
130 INIT
140 ON SRQ THEN 1010
150 DIM L1(26)
160 PRINT @22:"LLSET?"
170 WBYTE @86:
180 RBYTE L1
190 WBYTE @63,95:
200 PRINT L1
210 END
500 END
1000 REM ***** POLL ROUTINE *****
1010 POLL D,S;22
1020 RETURN
    
```

Line 150 -- Dimensions array variable L1 for 26 elements. Twenty-six must be used to acquire a complete PS 5010 LLSET? response.

Line 160 -- Sends the command LLSET? to the PS 5010 (primary address P). This tells the PS 5010 to be prepared to send its settings in binary block format when it receives MTA.

Line 170 -- Talk addresses (MTA=primary address +64) the power supply, device P.

Line 180 -- Assigns incoming data elements to array variable L1.

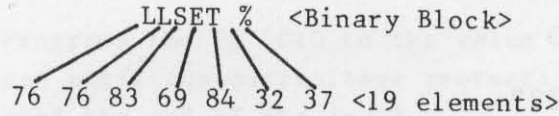
Line 190 -- Unlisten (63), untalk (95) everyone on the bus.

Line 200 -- Prints X (see below).

An example of the 26 elements in array X is:

76	76	83	69
84	32	37	0
16	10	35	122
7	122	50	10
0	255	255	0
0	0	255	255
144	-59		

The first seven elements represent the decimal equivalent of



the next 19 elements include the two byte count, data bytes, and the checksum byte. The last element -59 is the semicolon message unit delimiter (minus indicates EOI).

To execute the settings stored in L1, send MLA using the BASIC I/O statement WBYTE.

EXAMPLE: WBYTE 54:L1

The data in L1 is complete (including the LLSET command) for execution of LLSET. Do not alter this data because the microprocessor expects binary block format that was returned in response to the LLSET? and will not check for errors. Errors could be damaging to the instrument.

Address 54 represents primary plus MTA (22+32=54).

## Application Programs Using the PS 5010

## Series-Connected Supplies

The outputs of two or more PS 5010s can be connected in series as shown in Fig. 6-2 to obtain an output voltage equal to the sum of the output voltages from each supply. Each supply must be programmed individually to obtain the desired output voltage.

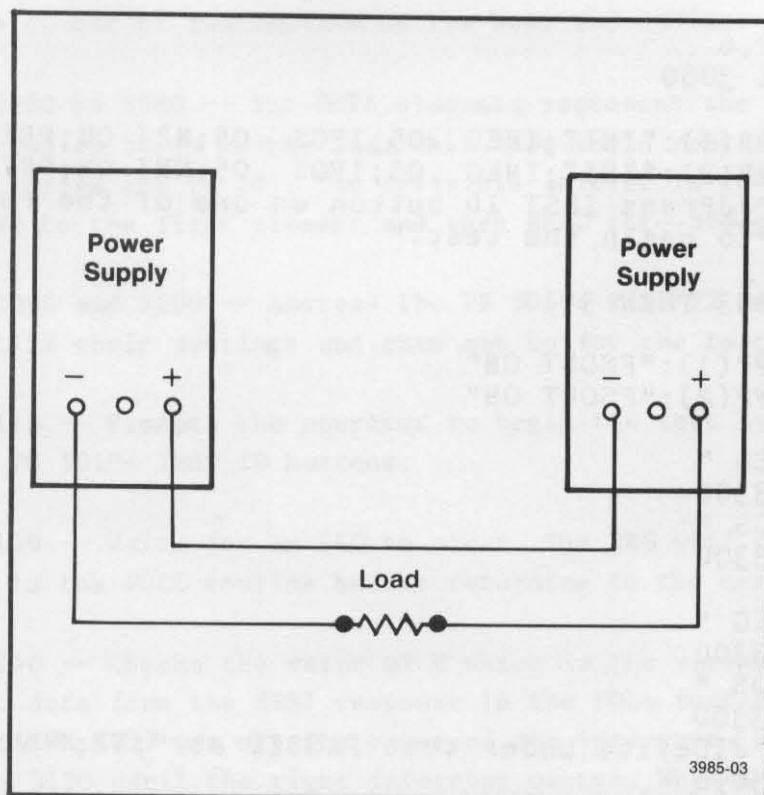


Fig. 6-2. Supplies series connected.

## NOTE

The PS 5010 has internal diodes connected across the output to protect the series-connected supplies against reverse polarity if the load is shorted, or one of the supplies is not on.



The following program illustrates using two PS 5010s in series by effectively ramping the voltage from 0 to 128 volts (the PS 5010 can be floated to 150 V with respect to ground) and enabling the regulation interrupt reporting capability to report any regulation changes.

```

2990 REM *****
3000 REM * TWO PS 5010 IN SERIES USING REGULATION INTERRUPTS *
3010 REM *****
3020 INIT
3030 ON SRQ THEN 3460
3040 DIM P(2)
3050 E=0
3060 DATA 22,6
3070 RESTORE 3060
3080 READ P
3090 PRINT @P(1):"INIT;INEG .05;IPOS .05;NRI ON;PRI ON;USER ON"
3100 PRINT @P(2):"INIT;INEG .05;IPOS .05;NRI ON;PRI ON;USER ON"
3110 PRINT "JJPress INST ID button on one of the PS 5010`s"
3120 PRINT "to begin the test."
3130 WAIT
3140 IF E<>403 THEN 3130
3150 V2=0
3160 PRINT @P(1):"FSOUT ON"
3170 PRINT @P(2):"FSOUT ON"
3180 A=22
3190 A$="VNEG "
3200 GOSUB 3300
3210 A$="VPOS "
3220 GOSUB 3300
3230 A=6
3240 A$="VNEG "
3250 GOSUB 3300
3260 A$="VPOS "
3270 GOSUB 3300
3280 PRINT "JIDevice under test PASSED at ";V2;" voltsGGG"
3290 GO TO 3090
3300 FOR V1=0 TO 32 STEP 0.5
3310 PRINT @A:A$;V1
3320 CALL "WAIT",0.11
3330 IF E>721 AND E<726 THEN 3370
3340 NEXT V1
3350 V2=V1+V2-0.5
3360 RETURN
3370 PRINT @P(1):"OUT OFF"
3380 PRINT @P(2):"OUT OFF"
3390 A$=REP("?",5,0)
3400 PRINT @A:A$
3410 INPUT @A:V3
3420 V4=V2+V3
3430 PRINT "JIDevice under test FAILED: "
3440 PRINT "I * supply current limited at ";V4;" voltsGGG"
3450 GO TO 3090

```

```
3460 REM ***** POLL ROUTINE *****
3470 POLL D,S;P(1);P(2)
3480 IF D=0 THEN 3510
3490 PRINT @P(D):"ERR?"
3500 INPUT @P(D):E
3510 RETURN
```

Lines 3020 to 3050 -- Initializes, establishes ON SRQ, dimensions array P for the number of instruments on the bus, and defines variables.

Lines 3060 to 3080 -- The DATA elements represent the primary addresses of the two PS 5010s on the bus. This list would contain all GPIB instruments in the system (up to 14). The invisible pointer in the DATA statement is RESTORED to the first element and then READ into array variable P.

Lines 3090 and 3100 -- Address the PS 5010s P(1)=22 and P(2)=6 to first INITIALIZE their settings and then set up for the test.

Line 3110 -- Prompts the operator to begin the test by pressing either one of the PS 5010s INST ID buttons.

Line 3130 -- Waits for an SRQ to occur. The SRQ will cause the program to branch to the POLL routine before returning to the next line.

Line 3140 -- Checks the value of E which is the variable containing the numeric data from the ERR? response in the POLL routine. If the INST ID button (ERR 403) was not the cause of the interrupt, control is sent back to line 3130 until the right interrupt occurs. When E=403 program flow continues.

Lines 3160 and 3170 -- Turn the outputs on.

Lines 3180 to 3270 -- Establish the primary address and the supply (neg or pos) which is to have its voltage ramped before branching to subroutine 3300. These three steps are repeated until either the desired 128 volts are reached or an interrupt occurs.

Line 3300 -- Sets up the incrementing limits of 0 to 32 in 500 mV steps.

Line 3310 -- Programs the PS 5010 at address A, and the supply defined by B\$ for the voltage set by V1.

Line 3320 -- Delays for regulation interrupt SRQ.

Line 3330 -- Checks E (ERR? response) for a regulation interrupt value. If E is within the limits, the program jumps to line 3370, otherwise it goes to the next V1.

Line 3350 -- V2 sums the last value of V1 before returning to the main program if a regulation interrupt has not occurred.

Lines 3370 and 3380 -- Turn both P(1) and P(2) outputs off.

Line 3390 -- Converts A\$ to a query command by replacing the space after the command with a "?".

Line 3400 -- Queries the last supply addressed (A) for its voltage value.

Line 3410 -- Inputs the response to voltage query to variable V3.

Line 3420 -- V4 equals the accumulated value in V2 plus V3 to indicate the voltage at which the output current limited.

Line 3430 -- Reports the test failed and V4 equals voltage at which the supply current limited.

Line 3450 -- Loops back to repeat the test.

Lines 3460 to 3510 -- Provide the POLL routine.

### Error Decoding

This is an expanded SRQ subroutine that decodes the ERR? response to print an error or status message. It is based on the full error-decoding routine in Section 7 that handles not only the DM 5010 but other TM 5000 instruments as well.

GPIB PROGRAMMING GUIDE

```

100 REM *****
110 REM * PRINTING PS 5010 ERROR MESSAGES IN RESPONSE TO ERR? *
120 REM *****
130 INIT
140 ON SRQ THEN 3000
150 B$=""
160 ON SRQ THEN 3030
170 REM .
180 REM .
190 REM .
3000 REM ***** DELAY ROUTINE *****
3010 CALL "WAIT",1
3020 RETURN
3030 REM ***** POLL ROUTINE ****
3040 POLL D,S;22
3050 PRINT @22:"ERR?;ID?"
3060 INPUT @22:E,E$
3070 E$=SEG(E$,9,6)
3080 GOSUB 4000
3090 RETURN
4000 REM ***** CODING FOR PS 5010 REPORTING ERROR? *****
4010 IF LEN(B$) THEN 4420
4020 DELETE B$
4030 DIM B$(1300)
4040 B$=" 101 Command Header Error"
4050 B$=B$&" 102 Header Delimiter Error"
4060 B$=B$&" 103 Command Argument Error"
4070 B$=B$&" 104 Argument Delimiter Error"
4080 B$=B$&" 106 Missing Argument"
4090 B$=B$&" 107 Invalid Message Unit Delimiter"
4100 B$=B$&" 108 Checksum Error"
4110 B$=B$&" 109 Bytecount Error"
4120 B$=B$&" 201 Command Not Executable in Local"
4130 B$=B$&" 202 Settings lost due to rtl"
4140 B$=B$&" 203 I/O Buffers full, Output dumped"
4150 B$=B$&" 204 Settings Conflicts"
4160 B$=B$&" 205 Argument Out of Range"
4170 B$=B$&" 206 Group Execute Trigger ignored"
4180 B$=B$&" 302 System Error"
4190 B$=B$&" 303 Math Pack Error"
4200 B$=B$&" 340 System RAM Error"
4210 B$=B$&" 341 System RAM Error (low nibble)"
4220 B$=B$&" 372 C000 ROM Placement Error"
4230 B$=B$&" 373 D000 ROM Placement Error"
4240 B$=B$&" 374 E000 ROM Placement Error"
4250 B$=B$&" 375 F000 ROM Placement Error"
4260 B$=B$&" 392 C000 ROM Checksum Error"
4270 B$=B$&" 393 D000 ROM Checksum Error"
4280 B$=B$&" 394 E000 ROM Checksum Error"
4290 B$=B$&" 395 F000 ROM Checksum Error"
4300 B$=B$&" 401 Power On"
4310 B$=B$&" 403 User Request"
4320 B$=B$&" 521 Display During Signature Analysis"

```

```

4330 B$=B$&" 721 Neg. Supply Change to Voltage Regulation"
4340 B$=B$&" 722 Neg. Supply Change to Current Regulation"
4350 B$=B$&" 723 Neg. Supply Change to Unregulated"
4360 B$=B$&" 724 Pos. Supply Change to Voltage Regulation"
4370 B$=B$&" 725 Pos. Supply Change to Current Regulation"
4380 B$=B$&" 726 Pos. Supply Change to Unregulated"
4390 B$=B$&" 727 Log. Supply Change to Voltage Regulation"
4400 B$=B$&" 728 Log. Supply Change to Current Regulation"
4410 B$=B$&" 729 Log. Supply Change to Unregulated"
4420 A$=STR(E)
4430 A$=A$&" "
4440 E1=POS(B$,A$,1)
4450 E2=POS(B$," ",E1)
4460 A$=SEG(B$,E1,E2-E1+1)
4470 PRINT "J ";E$;" ERR";A$
4480 RETURN

```

#### Zener-Diode Test

The Zener-Diode program combines the settability, accuracy and interrupt reporting features of the PS 5010 with the graphic capabilities of the 4050 Series Controllers to provide a typical PS 5010 application for incoming inspection or component evaluation.

In concept the program is an expanded version of the Regulation and User Interrupt program illustrated earlier in this section.

The use of predefined variables rather than specific values throughout the programs provides the basis for countless applications. Change the value of Z in line 370 to the components nominal value and establish test and select limits in lines 380 through 410 according to your application and run the program.

Additional flexibility could be added by prompting the operator for the spec and limit values.

```

1 REM [ ZENER - DIODE TEST ]
2 GO TO 100
40 REM ***** UDK #10 TO GRAPH DATA *****
41 GOSUB 1140
42 RETURN
100 REM
110 REM ***** BEGIN TEST *****
120 REM
130 INIT
140 ON SRQ THEN 950
150 SET NOKEY

```

```

160 REM          ***** VARIABLES *****
170 REM
180 REM          D Primary address for the PS 5010
190 REM          D1,S Device number in POLL list and status byte
200 REM             respectively
210 REM          E Error/Status code response to ERR?
220 REM          F,P Counter for failures (F) and passes (P)
230 REM          I Counter for several FOR/NEXT loops
240 REM          L,L1 Lower test (10%) and acceptance (5%) limits
250 REM             respectively
260 REM          N Number of components to be tested - operator input
270 REM          U,U1 Upper test (10%) and acceptance (5%) limits
280 REM             respectively
290 REM          V FOR/NEXT incrementor for setting ramp voltage
300 REM          V1 Array of test results
310 REM          V3 Running sum of "passed" component values
320 REM          X X-axis tic interval: 1 if N<50 or 10 if N>50
330 REM          Z Nominal specification value of component under test
340 REM          C$ Null string for INPUT of carriage return
350 REM
360 D=22
370 Z=5.1
380 L=Z-Z*0.1
390 L1=Z-Z*0.05
400 U=Z+Z*0.1
410 U1=Z+Z*0.05
420 V3=0
430 P=0
440 F=0
450 C=0
460 PAGE
470 REM ***** INITIALIZE BASIC SETTINGS AND DIMENSION PARAMETERS *****
480 REM
490 PRINT @D:"INIT;IPOS .05;USER ON"
500 WBYTE @17:
510 PRINT "This test will check zener diodes to within +H 10% of their"
520 PRINT "nominal value and determine whether they passed or failed"
530 PRINT "within +H 5%."
540 PRINT "JJHow many components are to be tested? ";
550 INPUT N
560 PAGE
570 PRINT "JCONNECT ZENER AND PRESS INST ID TO BEGIN EACH TEST:_"
580 DIM V1(N)
590 FOR I=1 TO N
600 PRINT USING ""Ready to test component #""2DS":I
610 REM          ***** WAIT FOR USER REQUEST SRQ *****
620 WAIT

```

```

630 IF E<>403 THEN 620
640 REM          ***** LOOP TO RAMP VOLTAGE *****
650 FOR V=L TO U STEP 0.01
660 PRINT @D:"FSOUT ON;PRI ON"
670 IF E=725 THEN 770
680 PRINT @D:"VPOS ";V
690 REM          ***** DELAY TO ALLOW SRQ *****
700 CALL "WAIT",0.11
710 NEXT V
720 REM          ***** REPORT PROGRAMMED VOLTAGE IF NO SRQ OCCURED *****
730 GOSUB 1010
740 REM
750 REM          ***** TURN OUTPUT OFF WHEN SRQ OR LIMITS ARE MET ****
760 REM
770 PRINT @D:"OUT OFF"
780 NEXT I
790 SET KEY
800 PRINT "JJJThe test on ";N;" zeners is complete."
810 PRINT "JJJJJPress <RETURN> to continue"
820 INPUT C$
830 PAGE
840 REM          ***** REPORT SUMMARIZED DATA *****
850 REM
860 PRINT "JJJThe following zener voltages were reported:"
870 PRINT V1
880 PRINT "J      ";P;" components passed (";INT(P/N*1000)*0.1;"%)"
890 PRINT "J      ";F;" components failed (";INT(F/N*1000)*0.1;"%)"
900 IF P=0 THEN 930
910 PRINT "JThe average voltage for the components which passed was ";
920 PRINT INT(V3/P*1000)/1000
930 PRINT "JJJJJJJ          Press UDK #10 to graph the data."
940 END
950 REM          ***** SERIAL POLL ROUTINE *****
960 POLL D1,S;D
970 PRINT @D:"ERR?"
980 INPUT @D:E
990 IF E=725 THEN 1010
1000 RETURN
1010 REM          ***** QUERY FOR KNOWN PROGRAMMED VOLTAGE VALUE *****
1020 REM
1030 PRINT @D:"VPOS?"
1040 INPUT @D:V1(I)
1050 REM          ***** TEST FOR PASS OR FAIL *****
1060 IF V1(I)>L1 AND V1(I)<U1 THEN 1100
1070 PRINT "      Component FAILED at ";V1(I);" voltsGGG"
1080 F=F+1
1090 RETURN
1100 PRINT "      Component PASSED at ";V1(I);" voltsGGG"
1110 V3=V3+V1(I)
1120 P=P+1
1130 RETURN

```

```

1140 REM                ***** SUBROUTINE TO GRAPH DATA *****
1150 IF N>50 THEN 1180
1160 X=1
1170 GO TO 1190
1180 X=10
1190 PAGE
1200 PRINT @32,21:30,95
1210 PRINT "          ZENER VOLTAGES TESTED"
1220 REM
1230 REM ***** ESTABLISH VIEWPORT AND WINDOW PARAMETERS FOR LEGEND *****
1240 REM
1250 VIEWPORT 0,130,0,100
1260 WINDOW 0,130,0,100
1270 MOVE 75,10
1280 FOR I=1 TO 5
1290 RDRAW 2,0
1300 RMOVE 2,0
1310 NEXT I
1320 PRINT " Acceptable limits"
1330 MOVE 75,5
1340 FOR I=1 TO 10
1350 RDRAW 1,0
1360 RMOVE 1,0
1370 NEXT I
1380 PRINT " Average passes"
1390 MOVE 15,10
1400 PRINT "X interval = ";X;" component(s)"
1410 MOVE 15,5
1420 PRINT "Y interval = .1 volts"
1430 REM
1440 REM ***** ESTABLISH VIEWPORT AND WINDOW PARAMETERS FOR GRAPH *****
1450 REM
1460 VIEWPORT 10,120,15,100
1470 IF N=1 THEN 1500
1480 WINDOW 0,N-1,INT(L),INT(U+1)
1490 GO TO 1520
1500 WINDOW 0,N,INT(L),INT(U+1)
1510 REM                ***** ESTABLISH AXIS PARAMETERS
1520 AXIS X,0.1,0,Z
1530 REM                ***** DRAW DASHED LINE FOR LOWER LIMIT (L1) *****
1540 MOVE 0,L1
1550 FOR I=1 TO 30
1560 RDRAW N/60,0
1570 RMOVE N/60,0
1580 NEXT I
1590 REM                ***** DRAW DASHED LINE FOR UPPER LIMIT (U1) *****
1600 MOVE 0,U1
1610 FOR I=1 TO 30
1620 RDRAW N/60,0
1630 RMOVE N/60,0
1640 NEXT I

```



```

1650 REM          ***** DRAW DASHED LINE FOR AVERAGE VALUE
1660 IF P=0 THEN 1730
1670 MOVE 0,V3/P
1680 FOR I=0 TO 60
1690 RMOVE N/120,0
1700 RDRAW N/120,0
1710 NEXT I
1720 REM          ***** GRAPH DATA FROM TESTED COMPONENTS (V1)
1730 MOVE 0,V1(1)
1740 FOR I=1 TO N
1750 DRAW I-1,V1(I)
1760 NEXT I
1770 VIEWPORT 0,120,15,100
1780 REM
1790 REM          ***** PRINT VALUE OF Y-AXIS TIC INTERVALS
1800 REM
1810 FOR I=INT(L) TO INT(U+1) STEP 0.5
1820 MOVE 0,I
1830 PRINT I
1840 NEXT I
1850 MOVE 0,Z
1860 PRINT Z
1870 RETURN

```

Lines 40 to 42 -- Provide the necessary steps for exercising UDK#10.

Line 140 -- Identifies the POLL routine.

Line 150 -- Disables the user definable keys.

Line 490 -- Initializes, sets the positive current, and turns on the user request interrupt capability.

Line 550 -- Defines "N" which establishes the value for looping parameters, statistical computations and graph limits based on the number of components tested.

Line 580 -- Dimensions array V1 for N values.

Line 590 -- Begins loop for testing components.

Line 630 -- WAITs for an interrupt, line 640 checks for the specific interrupt.

Lines 650 to 720 -- Ramp to the voltage value in "V" until the loop is interrupted because the PS 5010 positive supply changed from CV to CC. Line 670 turns the floating supply and positive regulation interrupt on.

Line 740 -- Forces a query of the supply voltage if an interrupt does not occur. This insures proper accounting of all components tested.

Line 780 -- Turns the PS 5010 output off after each component has been tested.

Line 790 -- Returns program to next increment of I to test another component.

Line 800 -- Turns the user request interrupt off after all components are tested. This prevents accidental user interrupts.

Line 810 -- Enables the user definable keys.

Lines 820 and 830 -- Signal the operator that the test is complete.

Lines 850 to 950 -- Report summarized information about the test.

Lines 900 and 910 -- Computes the percentage of passes and failures by taking the integer value of the count divided by number of components for the average, then multiplying by 1000 and multiplying result by .1 to get a three-digit percentage reading.

Line 920 -- Bypasses the average passes computation if P=0 to eliminate a 4050 SIZE ERROR.

Line 940 -- Calculates the average passes.

Lines 970 to 1020 -- Provide the POLL routine unless E=725 at which time the routine branches to line 1030 to test and report pass/fail information before returning to the main program.

Lines 1100 and 1140 -- Count passes and failures.

Line 1130 -- Keeps a running sum of passed values.

Graphic Subroutine

(See 4050 Series Graphic System Reference Manual for more information.)

Line 1160 -- Begins the graphing subroutine.

Line 1170 -- Checks the value of N to scale the X tic interval (x).

Line 1220 -- Moves the cursor to 30,95 to print line 1230.

Lines 1270 and 1280 -- Set VIEWPORT and WINDOW to their default condition.

Lines 1290 to 1450 -- Print the legend.

Lines 1460 to 1520 -- Establish VIEWPORT and WINDOW parameters for the graph. Variables are used so the parameters will adjust according to number of components tested and the established limits.

Line 1540 -- Establishes X and Y tic intervals and intercept points.

Lines 1550 to 1600 -- Draws a dashed line at the lower acceptable limit.

Lines 1610 to 1660 -- Draws a dashed line at the upper acceptable limit.

Lines 1670 to 1730 -- Draws a dashed line at the average value of "passed" components, if P=0 then the routine is by-passed.

Lines 1740 to 1780 -- Graph the values in V1.

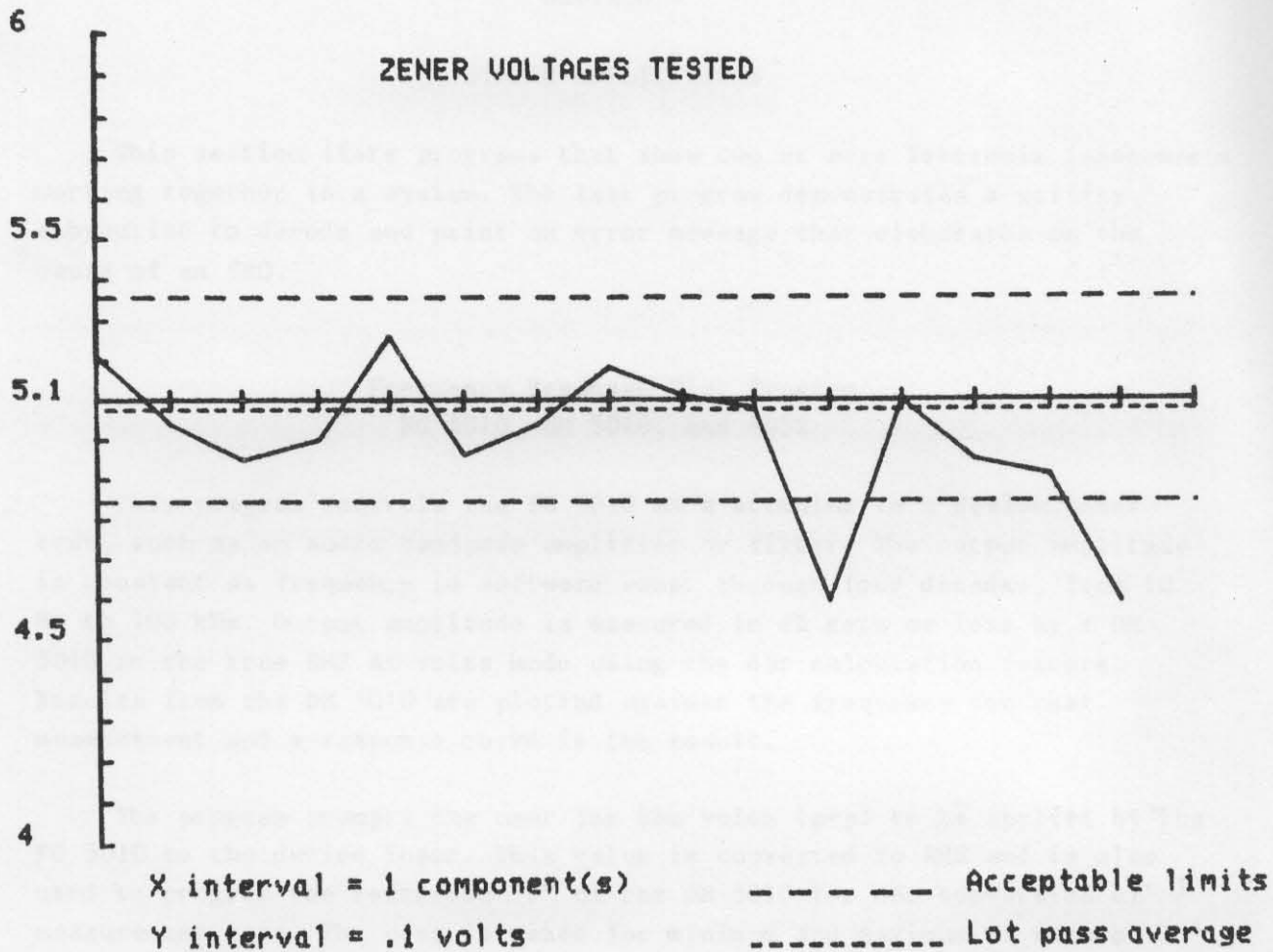
Line 1790 -- Re-establishes VIEWPORT large enough to print to the left of the axis.

Lines 1810 to 1860 -- Print the adjusted (line 1830) Y internal values and prints them at 500 mV increments (STEP 0.5).

Lines 1870 and 1880 -- Insure that the X,Y intercept value (Z=nominal spec) is printed even if it isn't at the 500 mV interval.

Line 1890 -- Returns to line 960 which terminates the program.

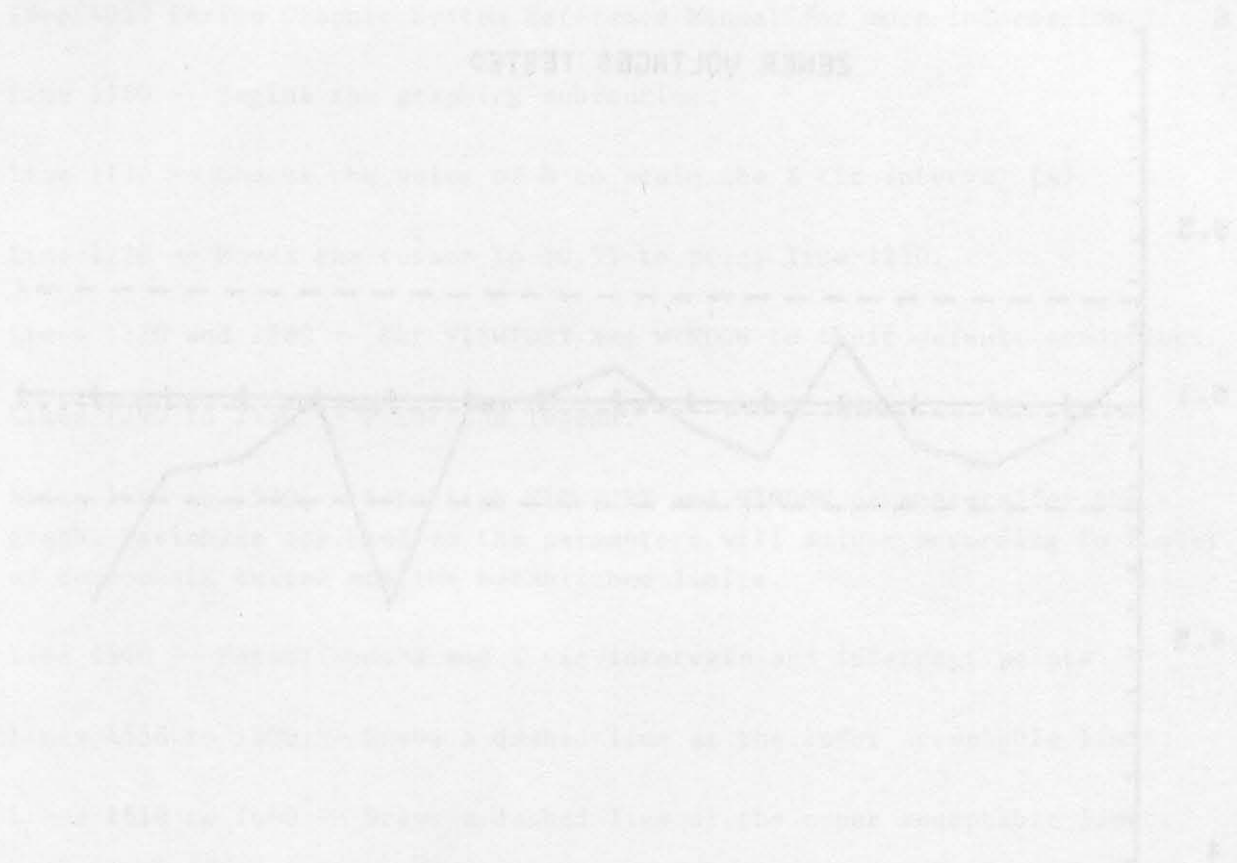
NOTE: The CALL "WAIT" statement in line 710 would have to be changed to a FOR/NEXT loop equal to 110 mS to run on a 4051.



3985-04

Fig. 6-3. Graphic output from Zener-Diode test.

SEWER VOLTAGES TESTED



SEWER VOLTAGES TESTED

1970-1971

SEWER VOLTAGES TESTED

1970-1971

SEWER VOLTAGES TESTED

1970-1971

## SECTION 7

## SYSTEM APPLICATIONS

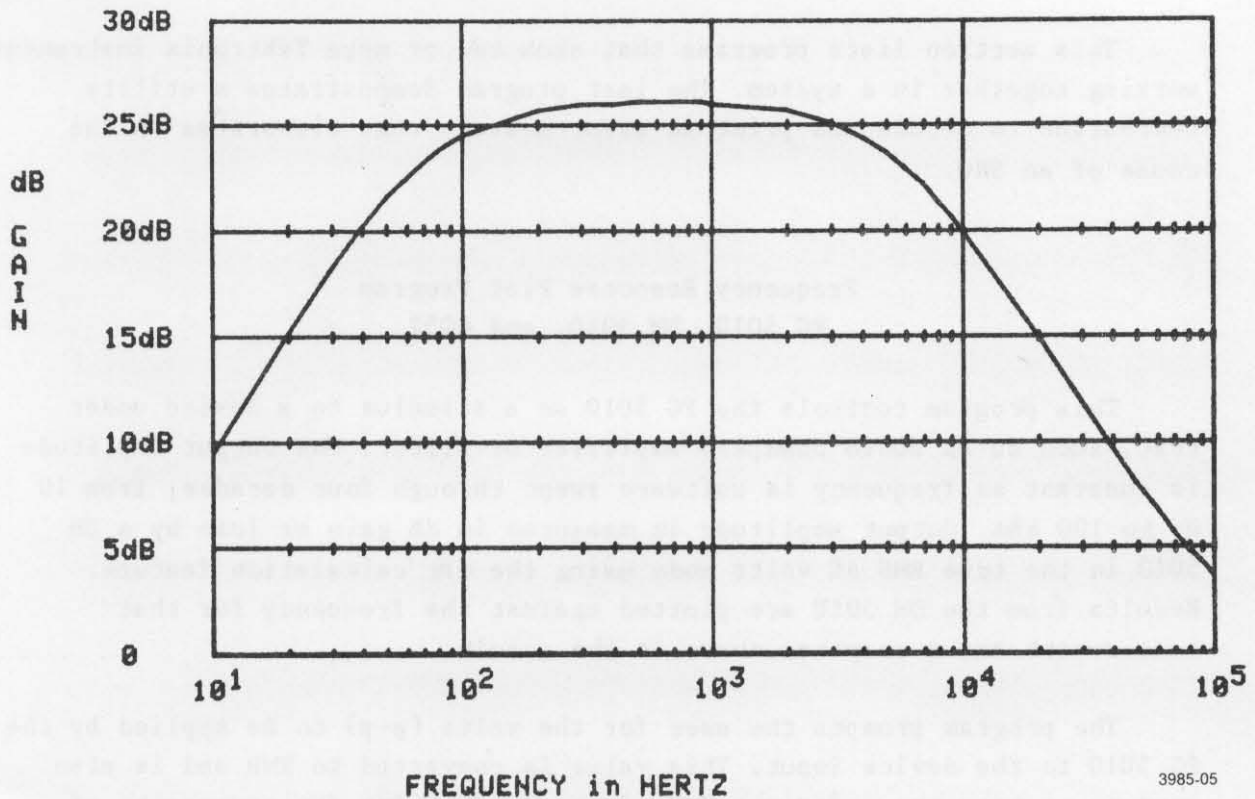
This section lists programs that show two or more Tektronix instruments working together in a system. The last program demonstrates a utility subroutine to decode and print an error message that elaborates on the cause of an SRQ.

**Frequency Response Plot Program**  
FG 5010, DM 5010, and 4052

This program controls the FG 5010 as a stimulus to a device under test, such as an audio bandpass amplifier or filter. The output amplitude is constant as frequency is software swept through four decades, from 10 Hz to 100 kHz. Output amplitude is measured in dB gain or loss by a DM 5010 in the true RMS AC volts mode using the dBr calculation feature. Results from the DM 5010 are plotted against the frequency for that measurement and a response curve is the result.

The program prompts the user for the volts (p-p) to be applied by the FG 5010 to the device input. This value is converted to RMS and is also used to program the reference "r" of the DM 5010 for dBr conversion of measurement data. The user is asked for minimum and maximum dB values before the graph is drawn and measurements begin. This permits "windowing-in" on a specific region of gain or loss over the range of frequencies. Figure 7-1 is a response plot of an audio amplifier with AC input coupling. A smaller range of frequencies within 10 Hz to 100 kHz could be specified by changing only a few lines of the program. Variables P and I $\emptyset$  control the decades of frequency to be plotted. Variables Q and I1 control frequency increments per decade. This program plots nine frequencies per decade. Typical time to plot is about 40 seconds for 38 data points.

FREQUENCY RESPONSE PLOT 0.5 Volts P-P Input



3985-05

Two instruments are supported in the program; the FG 5010 at address 24, and the DM 5010 at address 16. Instrument power must be applied before the program is run.

- Variables:
- A: POLL address pointer
  - D1: Bottom vertical scale dB limit
  - D2: Top vertical scale dB limit
  - D3: Vertical scale span (top limit - bottom limit)
  - F: Frequency used to program the FG 5010
  - G: dB Gain measurement from DM 5010
  - I0: FOR/NEXT loop counter - increment decades of frequency
  - I1: FOR/NEXT loop counter - increment per decade of frequency
  - I3: FOR/NEXT loop counter - draw freq. scale tic marks
  - P: FOR/NEXT loop counter - increment decades of frequency

GPIB PROGRAMMING GUIDE

- Q: FOR/NEXT loop counter - increment per decade of frequency
- R: Input voltage to test ckt. - reference for dBr mode
- S: Poll status byte from GPIB
- S1: Overrange flag set by DM 5010 overrange interrupt
- T:  $10 * \text{LOG}$  of frequency, used to plot test results

```

1 REM {FREQ. RESPONSE PLOT PROGRAM}
2 REM ---
100 INIT
110 ON SRQ THEN 1300
120 PAGE
130 S1=0
140 REM
150 PRINT "JJJIFREQUENCY RESPONSE PLOTJJJ"
160 PRINT "ENTER INPUT VOLTS (P-P) for device testing ";
170 INPUT R
180 R=ABS(R)
190 PRINT @16:"INIT;ACV;AVE 1;CALC DBR;DIGIT 3.5;OVER ON"
200 REM ----- SET DM REFERENCE TO RMS VALUE OF `R'
210 PRINT @16:"DBR ";R/(2*SQR(2))
220 REM ----- LOCKOUT ALL INSTRUMENT FRONT PANELS -----
230 WBYTE @48,56,17:
240 WBYTE @63,95:
250 PRINT "JENTER Vertical Scale Limits"
260 PRINT "  BOTTOM dB Limit: ";
270 INPUT D1
280 REM ----- TEST IF D1 IS EVENLY DIVISIBLE BY 5 -----
290 IF INT(D1/5)-D1/5=0 THEN 320
300 REM ----- SET D1 TO NEXT LOWER WHOLE MULTIPLE OF 5 -----
310 D1=INT(D1/5)*5
320 PRINT "  TOP dB Limit: ";
330 INPUT D2
340 REM ----- TEST IF D2 IS EVENLY DIVISIBLE BY 5 -----
350 IF INT(D2/5)-D2/5=0 THEN 380
360 REM ----- SET D2 TO NEXT WHOLE MULTIPLE OF 5 -----
370 D2=(INT(D2/5)+1)*5
380 REM ----- TEST FOR D1 > D2 -----
390 IF D2>D1 THEN 420
400 PRINT "JLIMITS INVERTEDGGG?"
410 GO TO 250
420 D3=D2-D1
430 REM ----- TITLE GRAPH AND LABEL AXES -----
440 PAGE
450 PRINT "IFREQUENCY RESPONSE PLOT ";R;" Volts P-P Input"
460 IMAGE 10L2A//A/A/A/A
470 PRINT USING 460:"dB","G","A","I","N"
480 MOVE 0,5
490 PRINT "I    FREQUENCY in HERTZ"
500 VIEWPORT 20,120,20,85

```



GPIB PROGRAMMING GUIDE

```

510 REM  -- WINDOW FOR 10 units/decade (1 to 5) and D1 dB to D2 dB  ---
520 WINDOW 10,50,D1,D2
530 MOVE 10,D2
540 DRAW 10,D1
550 DRAW 50,D1
560 REM  ----- INCREMENT THRU 2 TO 10 IN DECADES 1 TO 4  -----
570 FOR P=1 TO 4
580 REM - alter next line to specify different freq. intervals/decade
590 FOR Q=2 TO 10
600   T=10*LGT(Q*10^P)
610   MOVE T,D1
620   REM ----- DRAW SCALE LINES BY DECADES -----
630   FOR I3=D1 TO D2 STEP 5
640     REM ----- DRAW HORIZONTAL TIC MARKS -----
650     RMOVE 0,-0.2
660     RDRAW 0,0.4
670     MOVE T,I3
680   NEXT I3
690 NEXT Q
700 REM ----- MOVE ACCORDING TO SCALE AND LABEL THE HORIZ AXIS  --
710 MOVE P*10-0.025*D3,D1-0.075*D3
720 PRINT "10"
730 REM ----- PRINT THE EXPONENT OF 10 AS A SUPERSCRIPT -----
740 RMOVE 0.7,0.0175*D3
750 PRINT P
760 REM ----- DRAW VERTICAL SOLID LINES -----
770 MOVE T,D1
780 DRAW T,D2
790 NEXT P
800 REM ----- LABEL LAST DECADE -----
810 MOVE P*10-0.025*D3,D1-0.075*D3
820 PRINT "10"
830 RMOVE 0.7,0.0175*D3
840 PRINT P
850 REM ----- LABEL VERTICAL AXIS -----
860 FOR P=D1 TO D2 STEP 5
870   MOVE 5,P-0.5
880   IF P=0 THEN 910
890   PRINT P;"dB"
900   GO TO 920
910   PRINT " ";P
920   MOVE 10,P
930   DRAW 50,P
940 NEXT P
950 REM ----- Now Setup FG5010 -----
960 PRINT @24:"INIT;FREQ 10;AMPL ";R;"";OUT ON"
970 REM ----- LET THE DM5010 SETTLE BEFORE CONTINUING -----
980 CALL "WAIT",1
990 PRINT "KGGGGGGG"
1000 REM -----

```

GPIB PROGRAMMING GUIDE

```

1010 REM ----- SWEEP THE FG & ACQUIRE DM READINGS -----
1020 FOR IO=1 TO 4
1030 REM - alter next line to specify different freq. intervals/decade
1040 FOR I1=1 TO 9 STEP 1
1050 F=I1*10^IO
1060 PRINT @24:"FREQ ";F
1070 CALL "WAIT",0.7
1080 REM - IF OVERANGE INTERRUPT THEN WAIT 1.5 SEC. FOR ADDL. SETTling
1090 IF NOT(S1) THEN 1120
1100 CALL "WAIT",1.5
1110 S1=0
1120 PRINT @16:"SEND"
1130 INPUT @16:G
1140 REM ----- G IS THE CURRENT GAIN MEASUREMENT -----
1150 REM ----- MOVE TO FIRST READING, DON'T DRAW TO IT. -----
1160 REM
1170 IF NOT(IO=1 AND I1=1) THEN 1200
1180 REM ----- PLOT THE GAIN (G) VERSUS FREQUENCY (F) -----
1190 MOVE 10*LGT(F),G
1200 DRAW 10*LGT(F),G
1210 REM -- TEST LAST FREQ. IN LAST DECADE eg. 10*10^4 -----
1220 IF NOT(IO=4 AND I1=9) THEN 1250
1230 I1=10
1240 GO TO 1050
1250 NEXT I1
1260 NEXT IO
1270 END
1280 REM
1290 REM
1300 POLL A,S;16;24
1310 REM ---- POLL ROUTINE FOR DM5010 AT PRIMARY ADDRESS 16 ----
1320 REM ----- FG5010 AT PRIMARY ADDRESS 24 -----
1330 GO TO A OF 1350,1450
1340 RETURN
1350 IF S=102 OR S=118 THEN 1430
1360 REM ----- S1 IS OVERANGE FLAG -----
1370 REM ----- S1=0 IF NOT OVERANGE -----
1380 REM ----- S1=1 IF OVERANGE -----
1390 S1=0
1400 PRINT @32,21:0,95
1410 PRINT "DMM INTERRUPT STATUS BYTE ";S
1420 RETURN
1430 S1=1
1440 RETURN
1450 PRINT @32,21:0,90
1460 PRINT "FG INTERRUPT STATUS BYTE ";S
1470 RETURN

```

### Generating a Higher-Resolution DC Voltage

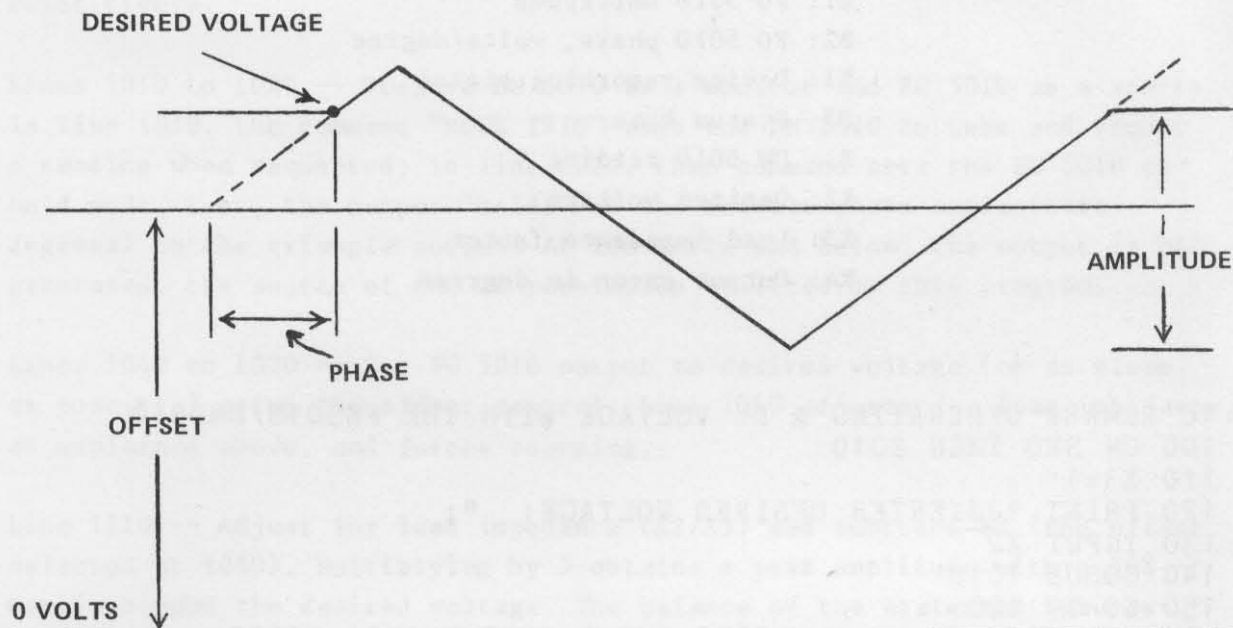
The FG 5010 Function Generator might not be your first choice for a DC voltage source. However, if your system includes an FG 5010 and a DM 5010 Digital Multimeter, you can obtain a high-resolution DC output using a software feedback loop. Just connect the FG 5010 output to the DM 5010 input as well as the stimulus node and incorporate the following routine in your program. In this version, the routine is nested in a program that prompts the operator to enter the desired voltage and also handles SRQs, such as those that result if you ask for an output level outside the range of the FG 5010 (+15 volts).

If you adapt the program, you may want to include only lines 1000 through 1280 as a subroutine. If so, set the value of X2 (the desired voltage) and X3 (the load impedance factor) before calling the routine. If the load impedance is not high (e.g., 1 megohm), adjust the value of X3 in your program using the voltage divider rule. Just set X3 equal to the load impedance and then apply the rule, taking into account the FG 5010's 50-ohm source impedance. For the case of a 50-ohm load, the program statements would be:

```
X3=50
X3=X3/(50+X3)
```

In this case, X3=0.5. In the example below, X3=1, which assumes the FG 5010 is connected to a high impedance.

The routine obtains the desired voltage as shown in Fig. 7-2.



3985-06

Fig. 7-2. Reaching a DC level using the FG 5010 triangle output. The solid line indicates the expected waveshape if the FG 5010 received a trigger.

To reach the desired voltage, the routine performs this sequence:

1. Set up the FG 5010 for a triangle output in hold mode (waiting for a trigger at center of triangle).
2. Set FG 5010 output as close as possible to the desired voltage using the offset control (up to +7.5 volts).
3. Compute the triangle amplitude needed to reach the desired voltage (with some margin) if beyond the offset range.
4. Compute the point on the triangle that would match the desired voltage and set the phase control.
5. If the phase control resolution allows, change the phase angle to get even closer to the desired voltage as monitored by the DM 5010.

GPIB PROGRAMMING GUIDE

Variables: B0: FG 5010 offset  
 B1: FG 5010 amplitude  
 B2: FG 5010 phase, volts/degree  
 S1: Device reporting status  
 S2: Status byte  
 X1: DM 5010 reading  
 X2: Desired voltage  
 X3: Load impedance factor  
 X4: Output error in degrees

```

10 REMARK GENERATING A DC VOLTAGE WITH THE FG5010/DM5010
100 ON SRQ THEN 2010
110 X3=1
120 PRINT "JJIENTER DESIRED VOLTAGE: ";
130 INPUT X2
140 GOSUB 1010
150 GO TO 120
1000 REM          ***** SETUP INSTRUMENTS *****
1010 PRINT @16:"INIT;MODE TRIG"
1020 PRINT @24:"INIT;TRIA;MODE TRIG;FRE 100;OUT ON"
1030 REM          ***** SET OFFSET AS CLOSE AS POSSIBLE *****
1040 B0=INT(X2/X3*100+0.5)/100
1050 IF ABS(B0)<=7.5 THEN 1070
1060 B0=SGN(B0)*7.5
1070 PRINT @24:"OFFS ";B0
1080 REM          ***** SET AMPLITUDE TO GET CLOSER *****
1090 REM          THE MIN AMPL IS 0.18V AND MAX IS 15V OPEN CIRCUIT
1100 REM          (1 MEGOHM) BUT ONLY 0.18V TO 7.5V INTO 50 OHMS
1110 B1=ABS(X2/X3-B0)*3 MAX 0.18/X3 MIN 15
1120 PRINT @24:"AMP ";INT(B1/0.18+0.5)*0.18
1130 REM          ***** NOW USE PHASE FOR FINE RESOLUTION *****
1140 PRINT @24:"AMPL?"
1150 INPUT @24:B1
1160 REM          ***** SCALE PHASE RANGE IN VOLTS PER DEGREE *****
1170 B1=B1/180
1180 B2=0
1190 PRINT @24:"PHA ";B2
1200 INPUT @16:X1
1210 REM          ***** JUMP OUT OF LOOP IF WITHIN 0.6 DEGREE *****
1220 X4=(X2-X1)/X3
1230 X4=X4/B1
1240 IF ABS(X4)<=0.6 THEN 1280
1250 REM          ***** SOFTWARE CORRECT FG5010 OUTPUT *****
1260 B2=INT(B2+X4+0.5)
1270 GO TO 1190
1280 RETURN
2000 REM          ***** HANDLE SRQS *****
2010 POLL S1,S2;16;24
2020 PRINT S2;" STATUS REPORTED AT ";16+8*(S1-1)
2030 RETURN
    
```

Lines 110 to 150 -- Input the desired voltage and call the subroutine repetitively.

Lines 1010 to 1020 -- Prepare DM 5010 as a monitor and FG 5010 as a source. In line 1010, the command "MODE TRIG" arms the DM 5010 to take and report a reading when requested; in line 1020, that command sets the FG 5010 to hold mode, i.e., the output "holds" at the default phase angle (zero degrees) on the triangle output. At 200 Hertz and below, the output is DAC generated, the source of the DC resolution utilized by this program.

Lines 1040 to 1070 -- Set FG 5010 output to desired voltage (or as close as possible) using the offset control. Line 1040 adjusts for load impedance, as explained above, and forces rounding.

Line 1110 -- Adjust for load impedance (X2/X3) and subtract B0 (the offset selected at 1060). Multiplying by 3 obtains a peak amplitude with a 50% margin beyond the desired voltage. The balance of the statement assures that the requested amplitude is within the amplitude range of the output.

Line 1120 -- Set the FG 5010 amplitude to the calculated value after rounding.

Lines 1140 to 1170 -- Calculate volts/degree of phase control by dividing p-p amplitude (B1) by 180 (+90%).

Lines 1180 to 1200 -- Initialize phase control and get a DM 5010 reading.

Lines 1220 to 1240 -- Is the amplitude error X4 less than 0.6 degrees of phase control (X4 is scaled in degrees by line 1230)? Again, X3 corrects for interaction of source impedance and load impedance. If the error can be corrected, proceed.

Lines 1260 to 1270 -- If the error is not too small, update phase and try again. If the error is within 0.6 degree limit, go back and input a new X2.

Lines 2010 to 2030 -- Poll on SRQ; print status and device address (for #1 on list, S1=1 and address=16; for #2, S1=2 and address=24). Remember, you must change your serial poll routine to include all instruments in the system if you adapt this routine to another program.

### Automatic Checkout of Device Under Test

Here's a program that performs a full checkout on a device under test using a TM 5000 system.

The device under test (DUT) is a single-board demo aid that is configured using six switches (Fig. 7-3) and provides these functions:

1. Non-inverting AC or DC coupled variable gain amplifier.
2. Free-running triangle generator at 200 kHz, 330 Hz, or 1 Hz with auxiliary square wave output.
3. Zero-crossing generator.
4. Four-bit binary counter with preset inputs and LED display.
5. Crowbar over-voltage protection at +18 volts and reverse voltage protection.

The DUT plugs directly into the PS 5010 front-panel jacks for power. Inputs and outputs are connected automatically to the TM 5000 instruments by the SI 5010 Programmable Scanner.

The program includes the prompts needed by the operator to connect and configure the device. Although you may not have access to the device and run this program, you may use the listing as an example of how to conduct various programmable tests. Remarks are provided throughout the program for this purpose.

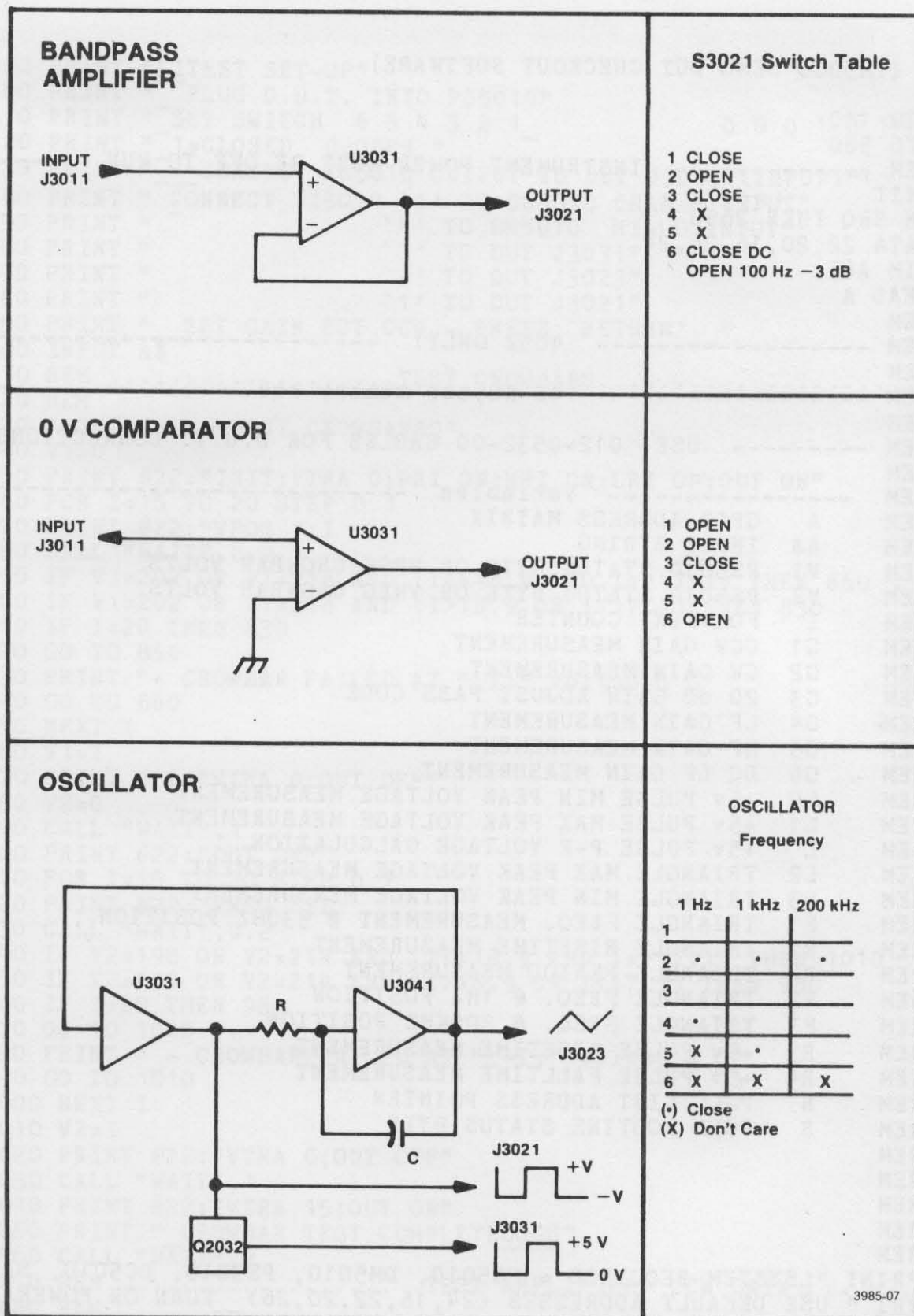


Fig. 7-3. DUT functions.



GPIB PROGRAMMING GUIDE

```

1 REM {TM5000 DEMO DUT CHECKOUT SOFTWARE}
2 REM
3 GO TO 110
4 GO TO 580
100 REM ----- INSTRUMENT POWER MUST BE OFF TO RUN -----
110 INIT
120 ON SRQ THEN 2650
130 DATA 22,20,16,26,24
140 DIM A(5)
150 READ A
160 REM
170 REM ----- 4052 ONLY! -----
180 REM
190 REM ----- 10 Kbytes memory reqd. -----
200 REM
210 REM ----- USE 012-0532-00 CABLES FOR D.U.T. CONNECTIONS -----
220 REM
230 REM ----- Variables -----
240 REM A GPIB ADDRESS MATRIX
250 REM A$ INPUT STRING
260 REM V1 PS5010 STATUS BYTE OR VPOS CROWBAR VOLTS
270 REM V2 PS5010 STATUS BYTE OR VNEG CROWBAR VOLTS
280 REM I FOR-NEXT COUNTER
290 REM G1 CCW GAIN MEASUREMENT
300 REM G2 CW GAIN MEASUREMENT
310 REM G3 20 dB GAIN ADJUST PASS CODE
320 REM G4 LF GAIN MEASUREMENT
330 REM G5 HF GAIN MEASUREMENT
340 REM G6 DC LF GAIN MEASUREMENT
350 REM L0 +5v PULSE MIN PEAK VOLTAGE MEASUREMENT
360 REM L1 +5v PULSE MAX PEAK VOLTAGE MEASUREMENT
370 REM L +5v PULSE P-P VOLTAGE CALCULATION
380 REM L2 TRIANGLE MAX PEAK VOLTAGE MEASUREMENT
390 REM L3 TRIANGLE MIN PEAK VOLTAGE MEASUREMENT
400 REM F1 TRIANGLE FREQ. MEASUREMENT @ 330HZ POSITION
410 REM R1 TRIANGLE RISETIME MEASUREMENT
420 REM R2 TRIANGLE PERIOD MEASUREMENT
430 REM F2 TRIANGLE FREQ. @ 1HZ POSITION
440 REM F3 TRIANGLE FREQ. @ 200KHZ POSITION
450 REM R3 +5v PULSE RISETIME MEASUREMENT
460 REM R4 +5v PULSE FALLTIME MEASUREMENT
470 REM N POLL LIST ADDRESS POINTER
480 REM S POLL ROUTINE STATUS BYTE
490 REM
500 REM
510 REM
520 REM
530 REM
540 PRINT "LSYSTEM REQUIRED = FG5010, DM5010, PS5010, DC5010, SI5010"
550 PRI " _USE DEFAULT ADDRESSES (24,16,22,20,26) _TURN ON POWER NOWGGG"
560 WAIT
570 SET KEY
580 ON SRQ THEN 2750

```

```

590 PRINT "LITEST SET-UP"
600 PRINT "___PLUG D.U.T. INTO PS5010"
610 PRINT "___SET SWITCH 6 5 4 3 2 1_           0 0 0 1 0 1"
620 PRINT "___1=CLOSED 0=OPEN ___"
630 PRINT "___CONNECT FG5010 OUTPUT TO DUT J3011 (INPUT)"
640 PRINT "___CONNECT SI5010 `A' TO DC5010 CHAN A INPUT"
650 PRINT "           `4' TO DM5010 HI-LO INPUT"
660 PRINT "           `3' TO DUT J3031"
670 PRINT "           `2' TO DUT J3023"
680 PRINT "           `1' TO DUT J3021"
690 PRINT "___SET GAIN POT CCW - PRESS `RETURN' ___"
700 INPUT A$
710 REM .....TEST CROWBARS.....
720 REM
730 PRINT "           TEST CROWBARSG"
740 V1=0
750 PRINT @22:"INIT;VTRA 0;PRI ON;NRI ON;LRI ON;OUT ON"
760 FOR I=15 TO 20 STEP 0.1
770 PRINT @22:"VPOS ";I
780 CALL "WAIT",0.2
790 IF V1=202 OR V1=218 AND (I<=18.9 AND I=>17.1) THEN 860
800 IF V1=202 OR V1=218 AND (I>18.9 OR I<17.1) THEN 830
810 IF I=20 THEN 830
820 GO TO 850
830 PRINT "+ CROWBAR FAILED AT ";I;" voltsG G G"
840 GO TO 860
850 NEXT I
860 V1=I
870 PRINT @22:"VTRA 0;OUT OFF"
880 V2=0
890 CALL "WAIT",1
900 PRINT @22:"OUT ON"
910 FOR I=15 TO 20 STEP 0.1
920 PRINT @22:"VNEG ";I
930 CALL "WAIT",0.2
940 IF V2=198 OR V2=214 AND (I<=18.9 AND I=>17.1) THEN 1010
950 IF V2=198 OR V2=214 AND (I>18.9 OR I<17.1) THEN 980
960 IF I=20 THEN 980
970 GO TO 1000
980 PRINT " - CROWBAR FAILED AT ";-I;" voltsG G G"
990 GO TO 1010
1000 NEXT I
1010 V2=I
1020 PRINT @22:"VTRA 0;OUT OFF"
1030 CALL "WAIT",1
1040 PRINT @22:"VTRA 15;OUT ON"
1050 PRINT " CROWBAR TEST COMPLETEGGGG"
1060 CALL "WAIT",2
1070 REM ..... TEST AMPLIFIER CCW GAIN .....
1080 REM
1090 PAGE

```

```

1100 PRINT "AMPLIFIER TEST  CCW GAIN  "
1110 PRINT @16:"INIT;ACV;DIG 3.5;CALC DBR;DBR .3535"
1120 PRINT @24:"INIT;SINE;AMPL 1;OUT ON"
1130 PRINT @26:"INIT;CONF 0,8,4,4;CLO 1,4;"
1140 CALL "WAIT",1
1150 INPUT @16:G1
1160 IF G1=>24 AND G1<=27 THEN 1200
1170 PRINT "CCW GAIN ERROR  - ";G1;" dBG G G"
1180 REM ..... TEST AMPLIFIER  CW GAIN .....
1190 REM
1200 PRINT " CW GAIN  SET GAIN POT TO CW END"
1210 PRINT "PRESS `RETURN' TO CONTINUE . . ."
1220 INPUT A$
1230 CALL "WAIT",1
1240 INPUT @16:G2
1250 IF G2=>-1 AND G2<=1 THEN 1290
1260 PRINT "CW GAIN ERROR  - ";G2;" dBG G G"
1270 REM ..... ADJUST 20 dB GAIN .....
1280 REM
1290 PRI " SET GAIN FOR 20.0 dB  +H 0.2 dB  - ADJUST GAIN FOR `PASS'"
1300 PRINT @16:"LIM 19.8,20.2;CALC DBR,CMPR;"
1310 REM GET 20 CONSEQUETIVE `PASSES' (a `2') in a row
1320 I=0
1330 INPUT @16:G3
1340 IF G3<>2 THEN 1320
1350 I=I+1
1360 IF I=20 THEN 1380
1370 GO TO 1330
1380 PRINT @16:"CALC DBR"
1390 INPUT @16:G3
1400 PRINT "GAIN ADJUSTEDGGG"
1410 REM ..... TEST LOW FREQ. BANDPASS .....
1420 REM
1430 PRINT " LF BANDPASS TEST"
1440 PRINT @24:"FREQ 50"
1450 PRINT @16:"CALC DBR;"
1460 CALL "WAIT",1
1470 INPUT @16:G4
1480 IF G4=>13 AND G4<=17 THEN 1520
1490 PRINT "LF BANDPASS ERROR  - ";G4;" dBG G G"
1500 REM ..... TEST HI FREQ. BANDPASS .....
1510 REM
1520 PRINT " HF BANDPASS TEST"
1530 PRINT @24:"FREQ 40E3"
1540 CALL "WAIT",1
1550 INPUT @16:G5
1560 IF G5=>9 AND G5<=11 THEN 1600
1570 PRINT "HF BANDPASS ERROR  - ";G5;" dBG G G"

```

```

1580 REM ..... TEST DC COUPLED RESPONSE .....
1590 REM
1600 PRINT " TEST DC COUPLED LF RESPONSE"
1610 PRI " SET DUT SWITCH #6 CLOSED - PRESS `RETURN' TO CONTINUE . . ."
1620 INPUT A$
1630 PRINT @24:"FREQ 50"
1640 CALL "WAIT",2
1650 INPUT @16:G6
1660 IF G6=>19.8 AND G6<=20.2 THEN 1710
1670 PRINT " DC COUPLED ERROR - ";G6;" dBG G G"
1680 CALL "WAIT",5
1690 REM ..... TEST SQUARE & TRIANGLE OSCILLATOR .....
1700 REM
1710 PRINT "LTEST OSCILLATOR OUTPUT AND FREQ."
1720 PRINT " SET SWITCH 6 5 4 3 2 1 0 1 0 0 1 0"
1730 PRINT " 1=CLOSED 0=OPEN PRESS `RETURN' TO CONTINUE . . .";
1740 INPUT A$
1750 PRINT @24:"INIT"
1760 PRINT @26:"INIT;CLO 3"
1770 CALL "WAIT",0.2
1780 PRINT @20:"FREQ;CHA A;COU DC;ATT 5;AUTO"
1790 CALL "WAIT",1
1800 REM ..... CHECK J3031 OUTPUT +5v PULSE .....
1810 REM
1820 PRINT " CHECK +5v AMPLITUDE_"
1830 PRINT @20:"CHA A;MIN?;MAX?"
1840 INPUT @20:L0,L1
1850 L=L1-L0
1860 IF L=>4.7 AND L<=5 THEN 1880
1870 PRINT " J3031 5v PULSE AMPLITUDE ERROR - ";L;" voltsG G G"
1880 REM ..... TEST J3023 AMPL AND FREQ AT 330 HZ .....
1890 REM
1900 PRINT " CHECK TRIANGLE Amplitude & 330Hz Frequency_"
1910 PRINT @26:"OPEN 3;CLO 2"
1920 CALL "WAIT",0.1
1930 PRINT @20:"AUTO"
1940 PRINT @20:"FREQ;CHA A;MAX?;MIN?;RESET;SEND"
1950 INPUT @20:L2,L3,F1
1960 IF L2>5.4 AND L2<6.6 THEN 1980
1970 PRINT "J3021 +PEAK AMPLITUDE ERROR - ";L2;" voltsG G G"
1980 IF L3>-6.6 AND L3<-5.4 THEN 2000
1990 PRINT "J3021 -PEAK AMPLITUDE ERROR - ";L3;" voltsG G G"
2000 IF F1>300 AND F1<345 THEN 2050
2010 PRINT "330 Hz FREQUENCY ERROR - ";F1;" HzG G G"
2020 REM ..... CHECK J3023 TRIANGLE SYMMETRY .....
2030 REM
2040 PRINT " CHECK TRIANGLE Symmetry (+Slope)_"
2050 PRINT @20:"RISE;SEND"
2060 INPUT @20:R1
2070 PRINT @20:"PER;SEND"
2080 INPUT @20:R2
2090 REM CHECK SYMMETRY
2100 IF R1>0.36*R2 AND R1<0.44*R2 THEN 2140
2110 PRINT " J3021 SYMMETRY/SLEW ERROR > 10%G G G"

```

GPIB PROGRAMMING GUIDE

```

2120 REM ..... CHECK 1Hz FREQUENCY .....
2130 REM
2140 PRINT " CHECK 1Hz SET SWITCH 6 5 4 3 2 1 0 0 1 0 1 0"
2150 PRINT "1=CLOSED 0=OPEN__PRESS `RETURN' TO CONTINUE . . ."
2160 INPUT A$
2170 REM TEST 1 Hz Freq.
2180 PRINT @20:"FREQ;AUTO;SEND"
2190 INPUT @20:F2
2200 IF F2>0.8 AND F2<1.1 THEN 2240
2210 PRINT " 1Hz FREQUENCY ERROR - ";F2;" HzG G G"
2220 REM ..... CHECK 200KHz FREQUENCY .....
2230 REM
2240 PRINT " L CHECK 200KHz Frequency"
2250 PRINT " SET SWITCH 6 5 4 3 2 1 0 0 0 0 1 0"
2260 PRINT "1=CLOSED 0=OPEN__PRESS `RETURN' TO CONTINUE . . ."
2270 INPUT A$
2280 PRINT @20:"AUTO;SEND"
2290 INPUT @20:F3
2300 IF F3<220000 AND F3>170000 THEN 2330
2310 PRINT " 200 KHz FREQUENCY ERROR ";F3;" HzG G G"
2320 REM
2330 REM ..... CHECK J3031 PULSE Tr and Tf .....
2340 REM
2350 PRINT " CHECK J3031 RISE & FALL TIME"
2360 PRINT @26:"OPEN 2;CLO 3"
2370 CALL "WAIT",0.1
2380 PRINT @20:"FREQ;CHA A;COU DC;ATT 5;RISE"
2390 PRI @20:"CHA A;LEV ";LO+0.2*L;";CHA B;LEV ";L1-0.2*L;";RESET;SEND"
2400 INPUT @20:R3
2410 IF R3<4.5E-7 AND R3>3.5E-7 THEN 2430
2420 PRINT " J3031 Tr ERROR - ";R3/1.0E-9;" nsG G G"
2430 PRINT @20:"FALL;"
2440 PRI @20:"CHA A;LEV ";L1-0.2*L;";CHA B;LEV ";LO+0.2*L;";RESET;SEND"
2450 INPUT @20:R4
2460 IF R4<2.1E-7 AND R4>1.1E-7 THEN 2500
2470 PRINT " J3031 Tf ERROR - ";R4/1.0E-9;" nsG G G"
2480 REM ..... CHECK COUNTER & DUT LED's .....
2490 REM
2500 PRINT " L.E.D. CHECKOUT"
2510 PRINT " GGVisually Check DUT LED's for EQUAL BRIGHTNESS"
2520 PRINT " Then SET SWITCH 4 CLOSED and observe proper BINARY count"
2530 PRINT " sequence on DUT LED's."
2540 PRINT " Press `RETURN' to continue . . . ";
2550 INPUT A$
2560 PRINT " ITEST CONCLUDED G G G G G"
2570 PRINT " IPress USER KEY ↑ tō RESTART"
2580 PRINT @22:"INIT"
2590 END
2600 REM
2610 REM
2620 REM
2630 REM
2640 REM

```

GPIB PROGRAMMING GUIDE

```

2650 REM ----- POWER ON INTERRUPT HANDLER -----
2660 REM ----- POWER ON SELF TEST PAUSE -----
2670 CALL "WAIT",5
2680 POLL N,S;22
2690 POLL N,S;20
2700 POLL N,S;16
2710 POLL N,S;26
2720 POLL N,S;24
2730 RETURN
2740 REM ----- INTERRUPT HANDLER -----
2750 POLL N,S;22;20;16;26;24
2760 IF N=1 THEN 2810
2770 PRINT @A(N):"ERR?;ID?"
2780 INPUT @A(N):A$
2790 PRINT "_SRQ INTERRUPT : ";A$
2800 RETURN
2810 REM CHECK STATUS BYTE FOR PS5010
2820 IF S=202 OR S=218 THEN 2860
2830 IF S=198 OR S=214 THEN 2880
2840 IF S=206 OR S=222 THEN 2900
2850 RETURN
2860 V1=S
2870 RETURN
2880 V2=S
2890 RETURN
2900 PRINT "_+5v CURRENT LIMIT"
2910 RETURN

```

## Full Error Message Decoding

The "ERR?" query command expands Tektronix instrument status reporting. The error code provides further information on the previous status byte reported by an instrument. The sequence is:

1. An instrument asserts SRQ to report power on, a command error, operation complete, or whatever. Whether the instrument asserts SRQ depends on GPIB status and error reporting commands listed in the Appendix.
2. The controller conducts a serial poll, obtaining a status byte.
3. The controller sends "ERR?" and inputs the response.

The following program demonstrates a subroutine that decodes the "ERR?" responses and prints the name of the reporting instrument, the error number, and a corresponding message. The point of this example is made by the subroutine; the talk/listen portion is just there as a vehicle to exercise the subroutine.

The serial poll and error decoding subroutines begin at line 360. The error list is inclusive of all TM 5000 instruments. The POLL statement assumes six instruments are in the system as configured earlier, so should be modified to fit your system. If you adapt the subroutines for another program, include also line 170. It should be executed once before the subroutine is called to initialize B\$. It should not be executed after that so that the subroutine reads the entire error list once and not again after that.

Variables: A\$: Error code and message decoded from E and B\$.  
 B\$: The full error TM 5000 error list.  
 D: Array of instrument addresses.  
 D1: Instrument address for talk/listen routine.  
 E: Error code from instrument received in reponse to "ERR?".  
 E\$: Balance of message requested by "ERR?;ID?" (contains only "ID?" response).  
 E1,E2: String handling variables.  
 M\$,R\$: Message to/from instrument in talk/listen routine.

```

100 REM *****
110 REM *** SUBROUTINE TO PRINT ERROR MESSAGE IN RESPONSE TO ERR? ***
120 REM *****
130 INIT
140 ON SRQ THEN 3000
150 PAGE
160 PRINT "TURN POWER ON"
170 B$=""
180 WAIT
190 ON SRQ THEN 3060
200 DIM R$(300)
210 PRINT "JENTER ADDRESS OF INSTRUMENT TO BE TALKED TO: ";
220 INPUT D1
230 PRINT "JENTER MESSAGE: ";
240 INPUT M$
250 PRINT @D1:M$
260 INPUT @D1:R$
270 PRINT "J";R$
280 GO TO 210
3000 REM ***** DELAY FOR PON AND POLL CONFIGURATION *****
3010 CALL "WAIT",5
3020 DIM D(6)
3030 DATA 16,18,20,22,23,24,26
3040 RESTORE 3030
3050 READ D
3060 REM ***** POLL ROUTINE *****
3070 POLL X,Y;D(1);D(2);D(3);D(4);D(5);D(6)
3080 PRINT @D(X):"ERR?;ID?"
3090 INPUT @D(X):E,E$
3100 E$=SEG(E$,9,6)
3110 GOSUB 4000
3120 RETURN
4000 REM ***** CODING FOR REPORTING ERROR? INFORMATION *****
4010 IF LEN(B$) THEN 6050
4020 DELETE B$
4030 DIM B$(9000)
4040 B$=" 0 No Errors or Events to Report"
4050 B$=B$&" 101 Command Header Error"
4060 B$=B$&" 102 Header Delimiter Error"
4070 B$=B$&" 103 Command Argument Error"
4080 B$=B$&" 104 Argument Delimiter Error"
4090 B$=B$&" 105 Non-numeric Argument (numeric expected)"
4100 B$=B$&" 106 Missing Argument"
4110 B$=B$&" 107 Invalid Message Unit Delimiter"
4120 B$=B$&" 108 Checksum Error"
4130 B$=B$&" 109 Bytecount Error"
4140 B$=B$&" 201 Command Not Executable in Local"
4150 B$=B$&" 202 Settings lost due to rtl"
4160 B$=B$&" 203 I/O Buffers full, Output dumped"
4170 B$=B$&" 204 Settings Conflicts"
4180 B$=B$&" 205 Argument Out of Range"
4190 B$=B$&" 206 Group Execute Trigger ignored"

```



GPIB PROGRAMMING GUIDE

4200	B\$=B\$&"	220	Select Error. No card in that slot"	
4210	B\$=B\$&"	231	Not in Calibrate Mode"	
4220	B\$=B\$&"	232	Beyond Calibration capability"	
4230	B\$=B\$&"	251	Symmetry/Frequency Conflict"	
4240	B\$=B\$&"	252	Amplitude/Offset Conflict"	
4250	B\$=B\$&"	253	Amplitude/AM Conflict"	
4260	B\$=B\$&"	254	Hold/Phase Lock Conflict"	
4270	B\$=B\$&"	255	Hold/Frequency Conflict"	
4280	B\$=B\$&"	256	Phase Lock/FM Conflict"	
4290	B\$=B\$&"	257	Phase Lock/VCF Conflict"	
4300	B\$=B\$&"	258	Gate/Mode Conflict"	
4310	B\$=B\$&"	301	Interrupt Fault"	
4320	B\$=B\$&"	302	System Error"	
4330	B\$=B\$&"	303	Math Pack Error"	
4340	B\$=B\$&"	311	Timeout (measurement not completed)"	
4350	B\$=B\$&"	312	Measurement Overflow"	
4360	B\$=B\$&"	313	Serial I/O Fault"	
4370	B\$=B\$&"	314	Mag-latch Relay Strobe to long"	
4380	B\$=B\$&"	315	Phase Lock Range Error"	
4390	B\$=B\$&"	316	Frequency correction range correction exceeded"	
4400	B\$=B\$&"	317	Front Panel Time Out"	
4410	B\$=B\$&"	318	Bad Calibration Constant"	
4420	B\$=B\$&"	320	Via fault on CPU board (FG 5010)	"
4430	B\$=B\$&"		Fault at U1221A or Input "	"
4440	B\$=B\$&"		Amplifier (DC 5009)	"
4450	B\$=B\$&"		Fault at U1000A or Input "	"
4460	B\$=B\$&"		Amplifier (DC 5010)"	"
4470	B\$=B\$&"	321	Trig/Gate control error on CPU board (FG 5010)	"
4480	B\$=B\$&"		Fault at U1211A (DC 5009)	"
4490	B\$=B\$&"		Fault at U1011A "	"
4500	B\$=B\$&"		(DC 5010)"	"
4510	B\$=B\$&"	322	4 MHz Reference Frequency Clock or Counter Fault-FG"	"
4520	B\$=B\$&"		Fault at U1201A (DC 5009)	"
4530	B\$=B\$&"		Fault at U1810A (DC 5010)"	"
4540	B\$=B\$&"	323	Frequency Control Logic Fault on Loop 2 board (FG)	"
4550	B\$=B\$&"		Fault at U1113A (DC 5009)	"
4560	B\$=B\$&"		Fault at U1801A (DC 5010)"	"
4570	B\$=B\$&"	324	Loop cycle counter fault on Loop 2 board (FG 5010)	"
4580	B\$=B\$&"		Fault at U1112A (DC 5009)	"
4590	B\$=B\$&"		Fault at U1120A (DC 5010)"	"
4600	B\$=B\$&"	325	Frequency prescaler fault on Loop 2 board (FG 5010)"	"
4610	B\$=B\$&"		Fault at U1111A (DC 5009)"	"
4620	B\$=B\$&"	326	Low Frequency Prescaler Fault on Loop 2 board (FG)	"
4630	B\$=B\$&"		Fault at U1332 (DC 5009)"	"
4640	B\$=B\$&"	327	No Signal Detected from Loop 1 board"	"
4650	B\$=B\$&"	328	Inadequate Frequency Range - 2 KHz range"	"
4660	B\$=B\$&"	329	Inadequate Frequency Range - 20 KHz range (FG 5010)"	"
4670	B\$=B\$&"		"A" chain failed to reset at zero"	"
4680	B\$=B\$&"		(DC 5010)"	"
4690	B\$=B\$&"	330	Inadequate Frequency Range - 200 KHz range(FG 5010)"	"
4700	B\$=B\$&"		Fault at U1221A or Input Amplifier"	"
4710	B\$=B\$&"		(DC 5009)	"
4720	B\$=B\$&"		Fault at U1011C or Input Amplifier (DC 5010)"	"

GPIB PROGRAMMING GUIDE

4730	B\$=B\$&"	331	Inadequate Frequency Range - 2 MHz range (FG 5010)	"
4740	B\$=B\$&"		Fault at U1211B (DC 5009)	"
4750	B\$=B\$&"		Fault at U1011B (DC 5010)	"
4760	B\$=B\$&"	332	Inadequate Frequency Range - 20 MHz range (FG 5010)	"
4770	B\$=B\$&"		Fault at U1201B (DC 5009)	"
4780	B\$=B\$&"		Fault at U1810B (DC 5010)	"
4790	B\$=B\$&"	333	Burst Counter Fault (FG 5010)	"
4800	B\$=B\$&"		Fault at U1113B (DC 5009)	"
4810	B\$=B\$&"		Fault at U1801B (DC 5010)	"
4820	B\$=B\$&"	334	Offset Generator Fault (FG 5010)	"
4830	B\$=B\$&"		Fault at U1112B (DC 5009)	"
4840	B\$=B\$&"		Fault at U1120B (DC 5010)	"
4850	B\$=B\$&"	335	Amplitude DAC Error (FG 5010)	"
4860	B\$=B\$&"		Fault at U1111B (DC 5009)"	"
4870	B\$=B\$&"	336	Amplitude Attenuator Error (FG 5010)	"
4880	B\$=B\$&"		Fault at U1012 (DC 5009)"	"
4890	B\$=B\$&"	337	Waveform Shaping Error"	"
4900	B\$=B\$&"	338	Normal/Complement Error"	"
4910	B\$=B\$&"	339	Low Frequency Generator DAC Error (FG 5010)	"
4920	B\$=B\$&"		"B" chain failed to reset to "	"
4930	B\$=B\$&"		zero (DC 5010)"	"
4940	B\$=B\$&"	340	System RAM Error"	"
4950	B\$=B\$&"	341	System RAM Error (low nibble)"	"
4960	B\$=B\$&"	342	RAM Error"	"
4970	B\$=B\$&"	343	RAM Error"	"
4980	B\$=B\$&"	344	RAM Error"	"
4990	B\$=B\$&"	345	RAM Error"	"
5000	B\$=B\$&"	346	RAM Error"	"
5010	B\$=B\$&"	347	RAM Error"	"
5020	B\$=B\$&"	348	RAM Error"	"
5030	B\$=B\$&"	349	RAM Error"	"
5040	B\$=B\$&"	350	CPU RAM Error"	"
5050	B\$=B\$&"	351	Calibration RAM Checksum Error"	"
5060	B\$=B\$&"	360	XXXX ROM Placement Error"	"
5070	B\$=B\$&"	361	XXXX ROM Placement Error"	"
5080	B\$=B\$&"	362	XXXX ROM Placement Error"	"
5090	B\$=B\$&"	363	XXXX ROM Placement Error"	"
5100	B\$=B\$&"	364	XXXX ROM Placement Error"	"
5110	B\$=B\$&"	365	XXXX ROM Placement Error"	"
5120	B\$=B\$&"	366	XXXX ROM Placement Error"	"
5130	B\$=B\$&"	367	XXXX ROM Placement Error"	"
5140	B\$=B\$&"	368	XXXX ROM Placement Error"	"
5150	B\$=B\$&"	369	XXXX ROM Placement Error"	"
5160	B\$=B\$&"	370	XXXX ROM Placement Error"	"
5170	B\$=B\$&"	371	XXXX ROM Placement Error"	"
5180	B\$=B\$&"	372	XXXX ROM Placement Error"	"
5190	B\$=B\$&"	373	XXXX ROM Placement Error"	"
5200	B\$=B\$&"	374	XXXX ROM Placement Error"	"
5210	B\$=B\$&"	375	XXXX ROM Placement Error"	"
5220	B\$=B\$&"	380	XXXX ROM Checksum Error"	"
5230	B\$=B\$&"	381	XXXX ROM Checksum Error"	"
5240	B\$=B\$&"	382	XXXX ROM Checksum Error"	"

5250	B\$=B\$&"	383	XXXX ROM Checksum Error"
5260	B\$=B\$&"	384	XXXX ROM Checksum Error"
5270	B\$=B\$&"	385	XXXX ROM Checksum Error"
5280	B\$=B\$&"	386	XXXX ROM Checksum Error"
5290	B\$=B\$&"	387	XXXX ROM Checksum Error"
5300	B\$=B\$&"	388	XXXX ROM Checksum Error"
5310	B\$=B\$&"	389	XXXX ROM Checksum Error"
5320	B\$=B\$&"	390	XXXX ROM Checksum Error"
5330	B\$=B\$&"	391	XXXX ROM Checksum Error"
5340	B\$=B\$&"	392	XXXX ROM Checksum Error"
5350	B\$=B\$&"	393	XXXX ROM Checksum Error"
5360	B\$=B\$&"	394	XXXX ROM Checksum Error"
5370	B\$=B\$&"	395	XXXX ROM Checksum Error"
5380	B\$=B\$&"	401	Power On"
5390	B\$=B\$&"	402	Operation Complete"
5400	B\$=B\$&"	403	User Request"
5410	B\$=B\$&"	521	Display During Signature Analysis"
5420	B\$=B\$&"	601	Overrange"
5430	B\$=B\$&"	602	Channel A Protect"
5440	B\$=B\$&"	603	Channel B Protect"
5450	B\$=B\$&"	604	No Prescaler"
5460	B\$=B\$&"	605	Time of Day Clock not initialized and WAIT UNTIL " "
5470	B\$=B\$&"		command was to be executed."
5480	B\$=B\$&"	701	Below Limits"
5490	B\$=B\$&"	703	Above Limits"
5500	B\$=B\$&"	711	Channel A Overflow"
5510	B\$=B\$&"	712	Channel B Overflow"
5520	B\$=B\$&"	721	Neg. Supply Change to Voltage Regulation"
5530	B\$=B\$&"	722	Neg. Supply Change to Current Regulation"
5540	B\$=B\$&"	723	Neg. Supply Change to Unregulated"
5550	B\$=B\$&"	724	Pos. Supply Change to Voltage Regulation"
5560	B\$=B\$&"	725	Pos. Supply Change to Current Regulation"
5570	B\$=B\$&"	726	Pos. Supply Change to Unregulated"
5580	B\$=B\$&"	727	Log. Supply Change to Voltage Regulation"
5590	B\$=B\$&"	728	Log. Supply Change to Current Regulation"
5600	B\$=B\$&"	729	Log. Supply Change to Unregulated"
5610	B\$=B\$&"	731	In Lock"
5620	B\$=B\$&"	732	Not Locked"
5630	B\$=B\$&"	741	Cannot Access PIA with all "'0`s'" (SI 5010)"
5640	B\$=B\$&"		741 Power on Errors on Card in "
5650	B\$=B\$&"	Slot 1	(MI 5010)"
5660	B\$=B\$&"	742	Power on Errors on Card in Slot 2"
5670	B\$=B\$&"	743	Power on Errors on Card in Slot 3"
5680	B\$=B\$&"	744	Power on Errors on Card in Slot 4"
5690	B\$=B\$&"	745	Power on Errors on Card in Slot 5"
5700	B\$=B\$&"	751	Cannot access PIA with all "'1`s'" (SI 5010)"
5710	B\$=B\$&"		751 Power on Errors on Card in "
5720	B\$=B\$&"	Slot 1	(MI 5010)"
5730	B\$=B\$&"	752	Power on Errors on Card in Slot 2"
5740	B\$=B\$&"	753	Power on Errors on Card in Slot 3"
5750	B\$=B\$&"	754	Power on Errors on Card in Slot 4"
5760	B\$=B\$&"	755	Power on Errors on Card in Slot 5"

```

5770 B$=B$&" 756 Power on Errors on Card in Slot 6"
5780 B$=B$&" 761 Power on Errors on Card in Slot 1"
5790 B$=B$&" 762 Power on Errors on Card in Slot 2"
5800 B$=B$&" 763 Power on Errors on Card in Slot 3"
5810 B$=B$&" 764 Power on Errors on Card in Slot 4"
5820 B$=B$&" 765 Power on Errors on Card in Slot 5"
5830 B$=B$&" 766 Power on Errors on Card in Slot 6"
5840 B$=B$&" 771 Hardware Errors on Card on Slot 1"
5850 B$=B$&" 772 Hardware Errors on Card on Slot 2"
5860 B$=B$&" 773 Hardware Errors on Card on Slot 3"
5870 B$=B$&" 774 Hardware Errors on Card on Slot 4"
5880 B$=B$&" 775 Hardware Errors on Card on Slot 5"
5890 B$=B$&" 776 Hardware Errors on Card on Slot 6"
5900 B$=B$&" 781 Hardware Errors on Card on Slot 1"
5910 B$=B$&" 782 Hardware Errors on Card on Slot 2"
5920 B$=B$&" 783 Hardware Errors on Card on Slot 3"
5930 B$=B$&" 784 Hardware Errors on Card on Slot 4"
5940 B$=B$&" 785 Hardware Errors on Card on Slot 5"
5950 B$=B$&" 786 Hardware Errors on Card on Slot 6"
5960 B$=B$&" 791 Armed SRQ (EXIT TRIG occurred) (SI 5010)"
5970 B$=B$&" 791 Armed Event Warning on Card"
5980 B$=B$&" in Slot 1 (MI 5010)"
5990 B$=B$&" 792 Armed Event Warning on Card in Slot 2"
6000 B$=B$&" 793 Armed Event Warning on Card in Slot 3"
6010 B$=B$&" 794 Armed Event Warning on Card in Slot 4"
6020 B$=B$&" 795 Armed Event Warning on Card in Slot 5"
6030 B$=B$&" 796 Armed Event Warning on Card in Slot 6"
6040 REM ***** SORTING THRU B$ *****
6050 DIM A$(300)
6060 A$=STR(E)
6070 A$=A$&" "
6080 E1=POS(B$,A$,1)
6090 E2=POS(B$," ",E1)
6100 A$=SEG(B$,E1,E2-E1+1)
6110 REM ***** REPORTING INSTRUMENT AND CODE *****
6120 PRINT "J ";E$;" -- ";A$
6130 RETURN

```

Lines 100 to 160 -- Initialize and get ready for power on.

Line 170 -- Define B\$ as a null string to set up test at line 4010.

Line 180 -- Wait for instruments to complete self-tests and assert SRQ to report power-up.

Line 190 -- Arm post power-up SRQ response.

Line 200 -- Make R\$ bigger than any expected instrument response. "SET?" returns longest string, about 150 to 200 characters, depending on instrument and what functions are selected.

Line 210 to 280 -- Operator prompts and message handling for talk/listen routine. Line 260 returns a null string from Tektronix instrument if M\$ was not an output command except for TM 5000 measurement instruments (counters and multimeters). In the latter case, they return a measurement.

Lines 3000 to 3050 -- Wait to be sure all instruments complete power-up tests and then configure array D with the factory-set addresses for one each of all TM 5000 instruments. Modify lines 3020 and 3030 to fit your system.

Lines 3060 to 3120 -- Poll for the status byte and query for the error code and ID. E contains the error number and E\$ contains the balance of the instrument response--the "ID?" response.

Lines 4000 to 4030 -- Check that B\$ is empty. If so, delete it, dimension it, and fill it with the full TM 5000 error list. If B\$ is already filled, skip to the error printing statements.

Lines 4040-6030 -- Build B\$ using control "\_" (RUBOUT) as a delimiter, the full error message list. The last error code segment must have a Control "\_" at the end to complete the segmenting process.

Lines 6040 to 6100 -- Build A\$ from the decimal string equivalent of the error code E, space " ", and the portion of B\$ containing the error message.

Lines 6110 to 6120 -- Print the instrument ID and error message.

Line 6130 -- Returns to main program.

APPENDIX

INSTRUMENT COMMANDS

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
ACDC	ACDC	<NUM>	Selects the ACV+DCV function. Argument selects range. Argument $\leq \emptyset$ or omitted selects auto-range.	DM 5010
.....				
ACV	ACV	<NUM>	Selects the ACV function. Argument selects range. Argument $\leq \emptyset$ or omitted selects auto-range.	DM 5010
.....				
AM	AM	ON or OFF	Enables or disables amplitude modulation. AM OFF is the power-up setting.	FG 5010
	AM?		Returns "AM ON;" or "AM OFF;"	FG 5010
.....				
AMPLITUDE	AMPL	<NUM>	Sets p-p open-circuit output voltage to value stated by argument. Power-up setting is 0.5 volt (500.0E-3). The range is 0.0 to 20.0.	FG 5010
	AMPL?		Returns "AMPL <NUM>:"	FG 5010
.....				

GPIB PROGRAMMING GUIDE

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
AVERAGE	AVE/AVG	<NUM>	Sets number of measurements averaged (in decades only).	DC 5009, DC 5010
		<NUM>	Sets the number of conversions used in CALC AVE program. Argument is truncated to integers.	DM 5010
	AVE?/AVG?		Query returns AVE (NUM); (-1 for AUTO AVERAGES).	DC 5009, DC 5010
	AVE?/AVG?		Returns "AVE <number>;".	DM 5010
.....				
CALCULATIONS	CALC	AVE or AVG	Calculates the average of the next "N" readings.	DM 5010
		RATIO	Subtracts offset and divides by scale factor set by RATIO command.	DM 5010
		DBM	Calculates power ratio, referenced to 1 mV dissipated in 600 ohms. Disables CALC DBR.	DM 5010
		DBR	Calculates logarithmic ratio of measurement to value of DBR command. Disables CALC DBM.	DM 5010
		CMPR or COMP	Compares input to limits set by LIMITS command.	DM 5010

NOTE: See DATA and SEND under Input/Output Commands and MONITOR under Instrument Commands.

DATA returns reading (saves out-of-limits measurement if MON ON).

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
	CALC	OFF	Disables all calculations	DM 5010
	CALC?		Returns "CALC OFF;" or list of enabled calculations.	DM 5010
.....				
DBR	DBR	<NUM>	Numeric sets value of reference used in CALC DBR command.	DM 5010
	DBR?		Returns "DBR <number>;".	DM 5010
.....				
DCV	DCV	<NUM>	Selects the DCV function. Argument selects range. Argument $\leq 0$ or omitted selects auto-range.	DM 5010
.....				
DIGITAL RESOLUTION	DIG	3.5	Selects FAST CONVERSION RATE (3.5 digit resolution).	DM 5010
		4.5	Selects normal CONVERSION RATE (4.5 digit resolution).	DM 5010
	DIG?		Returns "DIGIT 3.5;" or "DIGIT 4.5;".	DM 5010
.....				
DIODE TEST	DIODE		Selects DIODE TEST function. No argument.	DM 5010
.....				
DISPLAY	DISP	FREQ	Displays programmed frequency. This is the power-up setting.	FG 5010
		AMPL	Displays programmed p-p amplitude.	FG 5010
		OFFSET	Displays programmed offset voltage.	FG 5010
		NBURST	Displays programmed nburst value.	FG 5010



GPIB PROGRAMMING GUIDE

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
	DISP	PHASE	Displays programmed phase.	FG 5010
		SYM	Displays programmed symmetry.	FG 5010
.....				
EVENTS	EVE	BA	Counts Channel B during Channel A pulse width.	DC 5009, DC 5010
.....				
FALL TIME	FALL	A	Measures the fall time of the signal on Channel A.	DC 5010
.....				
FM	FM	ON	Enables frequency modulation. Disables VCF function. Center frequency is defined by the frequency setting.	FG 5010
		OFF	Frequency modulation is disabled; this is the power-up condition.	FG 5010
	FM?		Interrogates FM function. Response is "FM OFF;" or "FM ON;".	FG 5010
.....				
FREQUENCY	FREQ	A	Measures frequency of input signal on Channel A.	DC 5009, DC 5010
		<NUM>	Sets output frequency to argument value. Power-up setting is 1 kHz. Range is 2.0E-3 to 20.0E+6 (0.002 Hz to 20 MHz).	FG 5010
	FREQ?		Interrogates output frequency. Response is "FREQ <num>;".	FG 5010
.....				
FRONT PANEL TRIGGER	FPTR		Enables control of trigger levels by front-panel knobs.	DC 5009
.....				

GPIB PROGRAMMING GUIDE

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
FUNCTION	FUNC	SINE	Selects sinewave output. This is the power-up setting.	FG 5010
		SQU	Selects square-wave output.	FG 5010
		TRI	Selects triangle-wave output.	FG 5010
.....				
	FUNC?		Query returns current instrument function. Negative argument indicates auto-range.	DC 5009, DC 5010, DM 5010
	FUNC?		Returns "FUNC SINE;" , "FUNC SQUARE;" , or "FUNC TRIANGLE;" .	FG 5010
.....				
GATE	GATE	ON or OFF	Turns the gate on or off; FG5010 output occurs during gate time. Off is the power-up setting.	FG 5010
		GATE?	Returns "GATE OFF;" or "GATE ON;" .	FG 5010
.....				
HOLD	HOLD	ON	Stops the output signal at the voltage level that occurs when HOLD is executed. The hold function is inoperative if the output frequency exceeds 200 Hz and in Phase Lock Mode.	FG 5010
		OFF	Permits the FG 5010 to generate the output signal. This is the power-up setting.	FG 5010
		HOLD?	Returns "HOLD OFF;" or "HOLD ON;" .	FG 5010
.....				
ILOGIC	ILOG	<NUM>	Sets the logic supply current limit.	PS5010
		ILOG?	Returns ILOG <NUM> .	PS5010
.....				

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
INEGATIVE	INEG	<NUM>	Sets the negative floating supply current limit.	PS5010
	INEG?		Returns INEG <NUM>.	PS5010
.....				
IPOSITIVE	IPOS	<NUM>	Sets the positive floating supply current limit.	PS5010
	IPOS?		Returns IPOS <NUM>.	PS5010
.....				
ITRACK	ITRA	<NUM>	Sets both the positive and negative floating supplies current limits.	PS5010
.....				
LIMITS	LIM	<NUM1>, <NUM2>	Sets limits used in CALC CMPR program.	DM 5010
	LIM?		Returns "LIMITS <NUM>,<NUM>".	DM 5010
.....				
LOW FREQ RESPONSE	LFR	ON	Enables or disables the LOW FREQ RESPONSE function (instrument computes the average of four ACV or ACV+DCV measurements).	DM 5010
		or OFF		
	LFR?		Returns "LFR OFF;" or "LFR ON;".	DM 5010
.....				
MANUAL TRIGGER	MTRIG		Causes the FG 5010 to produce one cycle in triggered mode or N cycles in burst mode.	FG 5010
.....				
MODE	MODE	CONT	Sets the FG 5010 to generate a continuous output signal. Trigger events are ignored. "CONT" is the power-up setting.	FG 5010

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
	MODE	TRIG	Sets the FG 5010 to triggered mode. One cycle of the output signal occurs for each trigger event.	FG 5010
		GATE	Sets the FG 5010 to the gated mode. The output occurs only during one of the following: <ul style="list-style-type: none"> <li>a. Manual Trigger button pressed.</li> <li>b. "GATE ON" remote command is executed.</li> <li>c. Trigger/gate input signal is true.</li> </ul>	FG 5010
		BURST	Sets the FG 5010 to the burst mode. When a trigger occurs, the FG 5010 produces a burst of the programmed output signal; the number of cycles is determined by the "NBURST" setting.	FG 5010
		LOCK	Sets the FG 5010 to the external phase lock mode, which locks the phase and frequency of the output signal to that of the trigger/gate input signal.	FG 5010
		PHLOCK	Alternate abbreviation for "MODE LOCK".	FG 5010
		RUN	Selects the free-run trigger mode.	DM 5010

GPIB PROGRAMMING GUIDE

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
	MODE	TRIG	<p>Selects triggered mode: triggers a conversion only upon receipt of one of the following:</p> <ul style="list-style-type: none"> <li>- SEND command</li> <li>- Group Execute Trigger &lt;GET&gt; (only if device trigger (DT) is enabled).</li> <li>- My Talk Address (MTA) with the output unspecified.</li> <li>- Rear Interface Trigger (EXTRIG). Requires internal jumper installation. To cause a single trigger, this line must remain low between 0.5 and 10 microseconds. Holding this line low longer causes continuous readings.</li> </ul>	DM 5010
	MODE?		Returns "MODE RUN;" or "MODE TRIG;".	DM 5010
	MODE?		Returns "MODE CONT;", "MODE TRIG;", "MODE GATE;", "MODE BURST;", or "MODE LOCK;".	FG 5010
.....				
MONITOR	MON	ON	<p>Enables monitor SRQ. Saves a measurement outside the limits set by the LIMITS command (if CALC CMPR selected) and generates an SRQ. Returns this measurement in response to DATA. Subsequent out-of-limits measurements are not reported until the SRQ is serviced and measurement is read via DATA command.</p>	DM 5010

GPIB PROGRAMMING GUIDE

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
	MON	OFF	Disables the monitor SRQ.	DM 5010
	MON?		Returns "MONITOR ON;" or MONITOR OFF;".	DM 5010
.....				
NBURST	NBUR	<NUM>	Sets the number of cycles in a burst to that specified by the argument. Power-up setting is 10. Range is 1 to 9999.	FG 5010
	NBUR?		Returns "NBUR <NUM>;".	FG 5010
.....				
NULL	NULL	ON	Subtracts present reading from succeeding readings.	DC 5010
		OFF	Resets null value to 0.	DC 5010
	NULL	<NUM>	Subtracts argument from all readings ( $\emptyset$ disables).	DM 5010
	NULL?		Query returns NULL ON or NULL OFF.	DC 5010
			Returns "NULL <NUM>;".	DM 5010
.....				
OFFSET	OFFS	<NUM>	Sets the output open circuit offset voltage to the stated argument value in volts.	FG 5010
		OFFS?	Returns "OFFS <NUM>;".	FG 5010
.....				
OHMS	OHMS	<NUM>	Selects OHMS function. Argu- ment selects range. Argument $\leq \emptyset$ or omitted selects auto-range.	DM 5010
.....				
PERIOD	PER	A	Measures period of Channel A signal.	DC 5009, DC 5010
.....				

GPIB PROGRAMMING GUIDE

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
PHASE	PHAS	<NUM>	Sets the FG 5010 output signal phase to the argument in degrees (-90° to +90°), in relation to the trigger/gate input signal. Power-up setting is 0°.	FG 5010
	PHAS?		Returns "PHASE <NUM>;".	FG 5010
.....				
PROBE COMPENSATION	PROBE	A&B	Enables probe compensation.	DC 5009, DC 5010
.....				
RATIO	RAT	B/A	Measures ratio of B events to A events.	DC 5009, DC 5010
		<NUM 1>, <NUM 2>	Sets values for offset and scale used in CALC RATIO command. First argument is for scale, second for offset.	DM 5010
	RAT?		Returns "RATIO <NUM>,<NUM>;".	DM 5010
.....				
READY?	RDY?		Query returns RDY 1 for new data ready or RDY 0 for new data not ready.	DC 5009, DC 5010, DM 5010
.....				
RECALL	REC	<NUM>	Sets the FG 5010 to the settings stored in the location specified by the argument, except that DT, PLI, RQS, and USEREQ settings remain unchanged. Argument options are 0-9. If a location is recalled that has no settings stored, the instrument reverts to power-up settings.	FG 5010
.....				
RESET	RES		Resets counters, restarts current measurement.	DC 5009, DC 5010
.....				

GPIB PROGRAMMING GUIDE

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
RISETIME	RISE	A	Measures the rise-time of the signal on Channel A.	DC 5010
.....				
SINE FUNCTION	SINE		Same result as "FUNC SINE".	FG 5010
.....				
SQUARE FUNCTION	SQUARE		Same result as "FUNC SQUARE".	FG 5010
.....				
START	START		Starts TMANual, STOPped, or TOTALize measurement.	DC 5009, DC 5010
.....				
STOP	STOP		Stops any measurement except TEST or PROBECOMP.	DC 5009, DC 5010
.....				
STORE	STOR	<NUM>	The current settings of the FG 5010 (DT, PLI, RQS, and USEREQ) are saved in the location stated by the argument. Multiple arguments are allowed if connected by comma or space. Argument options are 0-9.  See the FG 5010 response to SEND to obtain another form of this command.	FG 5010
.....				
SYMMETRY	SYM	<NUM>	Sets output signal symmetry to stated argument value. Range is 10-90%. Power-up setting is 50%.	FG 5010
	SYM?		Returns "SYM <NUM>;".	FG 5010
.....				
TIME	TIME	AB	Measures time from A event to B event.	DC 5009, DC 5010
.....				



(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
TMANUAL	TMAN		Manual timing function (stop watch).	DC 5009, DC 5010
.....				
TOTALIZE	TOT	A	Totalizes Channel A events.	DC 5009, DC 5010
		A+B	Totalizes Channel A events. Totalizes Channel B events.	DC 5010
		A-B	Totalizes Channel A events. Totalizes Channel B events.	DC 5010
.....				
TRIANGLE FUNCTION	TRI		Same result as "FUNC TRI".	FG 5010
.....				
TRIGGER	TRIG?		Interrogates trigger/gate input status. Response is one of the following:  "TRIG 0;" -- trigger status unavailable.  "TRIG 1;" -- input potential is below level setting with + SLOPE, or above level set- ting with - SLOPE. This cor- responds to the "TRIG'D" LED off.  "TRIG 2;" -- input triggered at 3 Hz or faster. This cor- responds to the "TRIG'D" LED flashing.	FG 5010

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
	TRIG?		"TRIG 3;" -- input potential is above level setting with + SLOPE, or below level setting with - SLOPE. In GATE mode, the output is continuous; in TRIG and BURST modes, the last transition of the trigger caused a cycle or burst (depending on mode), and the TRIG'D lamp is lighted.	
.....				
VCF	VCF	ON	Enables the voltage-controlled frequency function and disables FM function. VCF stop frequency is the top of the range selected when the function is enabled. Range switching is not permitted when this function is selected. Frequency may be changed when in VCF, as long as the top of the range is not exceeded. In VCF only, the FREQUENCY parameters may be programmed to zero.	FG 5010
		OFF	Disables voltage-controlled frequency function. This is the power-up setting.	FG 5010
	VCF?		Returns "VCF OFF;" or "VCF ON;".	FG 5010
.....				
VLOGIC	VLOG	<NUM>	Sets logic supply voltage limit.	PS5010
	VLOG?		Returns VLOG <NUM>;.	PS5010
.....				

GPIB PROGRAMMING GUIDE

(INSTRUMENT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
VNEGATIVE	VNEG	<NUM>	Sets negative floating supply voltage limit.	PS5010
	VNEG?		Returns VNEG <NUM>;.	PS5010
.....				
VPOSITIVE	VPOS	<NUM>	Sets positive floating supply voltage limit.	PS5010
	VPOS?		Returns VPOS <NUM>;.	PS5010
.....				
VTRACK	VTRA	<NUM>	Sets positive and negative floating supplies for the same voltage limit.	PS5010
.....				
WIDTH	WID	A	Measures pulse width of Channel A signal.	DC 5009, DC 5010
.....				

GPIB PROGRAMMING GUIDE

INPUT/OUTPUT COMMANDS

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
ATTENUATION	ATT	1 or 5	Selects 1X or 5X attenuation.	DC 5009, DC 5010
	ATT?		Query returns ATT <NUM>; for designated channel.	DC 5009, DC 5010
.....				
AUTOTRIG	AUTO	A&B	Sets trigger level to signal midpoint (both channels).	DC 5009, DC 5010
		A	Sets Channel A trigger level to signal midpoint.	DC 5009, DC 5010
		B	Sets Channel B trigger level to signal midpoint.	DC 5010
.....				
CHANNEL	CHA	A or B	Selects channel for succeeding input settings.	DC 5009, DC 5010
	CHA?		Query returns CHA A or CHA B.	DC 5009, DC 5010
.....				
COMPLEMENT	COMP	ON or OFF	Enables or disables the complement function (output signal is inverted, but offset voltage remains the same).	FG 5010
	COMP?		Returns "COMP OFF;" or "COMP ON;".	FG 5010
.....				
COUPLING	COU	AC or DC	Sets input coupling mode.	DC 5009, DC 5010
	COU?		Query returns COU AC or COU DC.	DC 5009, DC 5010
.....				
DATA REQUEST	DATA		Returns reading (see also MON ON).	
.....				

GPIB PROGRAMMING GUIDE

(INPUT/OUTPUT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
FILTER	FIL	ON	Limits the Channel A and Channel B bandwidth to approximately 20 MHz.	DC 5010
		OFF	Turns off filter.	DC 5010
	FIL?		Query returns FIL ON or FIL OFF.	DC 5010
.....				
FLOATING SUPPLY OUTPUT	FSOUT	ON	Connects or disconnects floating supplies to	PS5010
		OFF	output terminals.	
	FSOUT?		Returns FSOUT ON; or FSOUT OFF;.	PS5010
.....				
LEVEL	LEV	<NUM>	Sets selected channel trigger level. Num range = +3.200 to -3.175 (x1) or +16.00 to -15.876 (x5).	DC 5009,
		<NUM>	Sets selected channel trigger level. Num range = +2 to -2 (x1) or +10 to -10 (x5).	DC 5010
	LEV?		Query returns trigger level setting of selected channel. LEV 9999 indicates front-panel control for DC 5009.	DC 5009, DC 5010
.....				
LOW LEVEL SETTINGS	LLSET	<BIN BLK>	Compact binary version of full instrument set-up; send <b>only</b> the exact response received for "LLSET?" or "SEND<n>".	FG 5010, PS5010
		LLSET?	Returns binary version of current instrument set-up.	FG 5010, PS5010
.....				

(INPUT/OUTPUT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
LOGIC SUPPLY OUTPUT	LSOUT	ON or OFF	Connects or disconnects the logic supply to the output terminals.	PS5010
	LSOUT?		Returns LSOUT ON; or LSOUT OFF;.	PS5010
MAXIMUM	MAX?		Query returns last AUTOtrig maximum peak voltage.	DC 5009, DC 5010
MINIMUM?	MIN?		Query returns last AUTOtrig minimum peak voltage.	DC 5009, DC 5010
OUTPUT	OUT	ON	Connects the FG 5010 output signal to the front-panel connector. "OUT OFF" is the power-up setting.	FG 5010,
		ON	Connects all supplies to their output terminals.	PS5010
		OFF	Disconnects the output from the front-panel connector.	FG 5010, PS5010
	OUT?		Returns "OUT ON;" or "OUT OFF:".	FG 5010,
			Returns "FSOUT ON;" or "FSOUT OFF;" and "LSOUT ON;" or "LSOUT OFF;".	PS5010
PRESCALE	PRE	ON	Enables prescaler and internal scaling.	DC 5009, DC 5010
		OFF	Disables prescaler and internal scaling.	DC 5009, DC 5010
	PRE?		Query returns PRE ON or PRE OFF.	DC 5009, DC 5010

GPIB PROGRAMMING GUIDE

(INPUT/OUTPUT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
SEND	SEND		Obtains and formats new measurement results.	DC 5009, DC 5010
	SEND		Moves latest measurement to the output buffer. If no measurement is available, the instrument triggers a conversion and puts measurement in the output buffer. If CALC CMPR is enabled, the SEND command returns:  "3.;" if input is above both limits.  "2.;" if input is between or equal to both limits.  "1.;" if input is below both limits.	DM 5010
	SEND	<NUM>	FG 5010 transmits the contents of the stored settings location stated by the argument. Multiple arguments are permitted. Argument options are 0-9. Response is "STOR <NUM>:<BINARY BLOCK>;".	FG 5010
.....				
SLOPE	SLO	POS	Triggers on positive slope.	DC 5009, DC 5010
		POS	Sets the trigger/gate input to respond to a positive transition or state. Gate is "ON" when the input is at the high state. "SLO POS" is the power-up setting.	FG 5010

GPIB PROGRAMMING GUIDE

(INPUT/OUTPUT COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
	SLO	NEG	Triggers on negative slope.	DC 5009, DC 5010
		NEG	Sets the trigger/gate input to respond to a negative transition or state. Gate is "ON" when the input is at the low state.	FG 5010
	SLO?		Query returns SLO NEG or SLO POS.	DC 5009, DC 5010
			Interrogates trigger slope status. Response is "SLOPE NEG;" or "SLOPE POS;".	FG 5010
.....				
SOURCE	SOU	REAR or INT	Selects rear interface as signal source.	DC 5009
		REAR	Measures rear interface connector inputs.	DM 5010
		FRONT or EXT	Selects front-panel connector as signal source.	DC 5009
			Measures front panel connector inputs.	DM 5010
	SOU?		Query returns SOUR REAR or SOUR FRONT.	DC 5009, DM 5010
.....				
TERMINATION	TERM	LO	Sets channel input to 50 ohm termination.	DC 5010
		HI	Sets channel input to 1 meg-ohm, 23 picofarad.	DC 5010
	TERM?		Query returns TER HI or TER LO.	DC 5010
.....				



STATUS COMMANDS

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
LOCK?	LOCK?		Returns external phase lock mode: 1, 0, or -1.	FG 5010
			These codes mean as follows:	
			1: FG 5010 locked to trigger/gate input signal.	
			0: FG 5010 unlocked.	
			-1: PHASE LOCK not selected.	
.....				
LRI	LRI	ON or OFF	Enables or disables the logic supply regulation interrupt.	PS 5010
	LRI?		Returns LRI ON; or LRI OFF;.	PS 5010
.....				
NRI	NRI	ON or OFF	Enables or disables the negative floating supply regulation interrupt.	PS5010
	NRI?		Returns NRI ON; or NRI OFF;	PS5010
.....				
OPC	OPC	ON	Enables assertion of SRQ on operation complete.	DC 5009, DC 5010
			Enables operation complete SRQ. DM 5010 asserts SRQ whenever a new measurement is available.	DM 5010
		OFF	Disables SRQ on operation complete.	DC 5009, DC 5010, DM 5010

GPIB PROGRAMMING GUIDE

(STATUS COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
	OPC?		Query returns "OPC ON;" or "OPC OFF;".	DC 5009, DC 5010, DM 5010
.....				
OVERFLOW or OVERRANGE	OVER	ON	Enables asserting of SRQ on counter overflow.	DC 5009, DC 5010
		ON	Enables overrange SRQ. If overranged when talked, returns "1.E+99;".	DM 5010
		OFF	Disables SRQ on counter overflow.	DC 5009, DC 5010
		OFF	Disables overrange SRQ. If overranged when talked, returns "1.E+99;".	DM 5010
	OVER?		Query returns OVER ON or OVER OFF.	DC 5009, DC 5010, DM 5010
.....				
PLI	PLI	ON or OFF	Enables or disables the phase-lock interrupt.	FG 5010
	PLI?		Returns "PLI OFF;" or "PLI ON;".	FG 5010
.....				
PRI	PRI	ON or OFF	Enables or disables the positive floating supply regulation interrupt.	PS5010
	PRI?		Returns "PRI ON;" or "PRI OFF;".	PS5010
.....				

(STATUS COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
REGULATION	REG?		Returns REG <NUM>, <NUM>, <NUM>. Sequence is neg, pos, logic. Numbers are 1=voltage, 2=current, 3=unregulated.	PS5010
.....				
RQS	RQS	ON	Enables instrument to generate service requests. This is the power-up state.	DC 5009, DC 5010, DM 5010, FG 5010, PS5010
		OFF	Disables all service requests except for power-on.	DC 5009, DC 5010, DM 5010, FG 5010, PS5010
	RQS?		Query returns "RQS ON;" or "RQS OFF;".	DC 5009, DC 5010, DM 5010, FG 5010, PS 5010
.....				
USEREQ	USER	ON or OFF	Enables or disables asserting of SRQ when INST ID button is pressed.	DC 5009, DC 5010, DM 5010,
			Power-up setting is "USER OFF".	FG 5010, PS 5010
	USER?		Query returns USER ON or USER OFF.	DC 5009, DC 5010, DM 5010, FG 5010, PS 5010
.....				

SYSTEM COMMANDS

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
DEVICE TRIGGER	DT	GATE	The GET interface message toggles the gate setting: on-to-off or off-to-on. Can be overridden by external signal.	FG 5010
		GATE	GET controls start and stop.	DC 5009, DC 5010
		SET	Instrument waits for GET to update settings.	FG 5010, PS 5010
		TRIG	GET triggers reading MODE TRIG.	DM 5010
		TRIG	GET performs RESET.	DC 5009, DC 5010
		TRIG	GET triggers a cycle (TRIG mode) or burst (NBURST mode).	FG 5009
		OFF	Disables response to GET (power-on condition).	DC 5009, DC 5010, DM 5010, FG 5010, PS 5010
	DT?		Returns current setting of the DT function.	DC 5009, DC 5010, DM 5010, FG 5010, PS 5010

.....

GPIB PROGRAMMING GUIDE

(SYSTEM COMMANDS cont.)

COMMAND	HEADER	ARGUMENT	DESCRIPTION	INSTRUMENT
ERROR?	ERR?	RQS ON:	Returns error code for most recent status reported, or 0 if no SRQ.	DC 5009, DC 5010, DM 5010, FG 5010, PS 5010
			RQS OFF: Returns error code for event waiting to be reported.	DC 5009, DC 5010, DM 5010, FG 5010, PS 5010
.....				
IDENTIFY	ID?		Returns instrument model number, Codes and Formats version number, and firmware version number.	DC 5009, DC 5010, DM 5010, FG 5010, PS 5010
.....				
INITIALIZE	INIT		Initializes all programmable settings to power-on state (and front-panel) settings in case of DC 5009).	DC 5009, DC 5010, DM 5010, FG 5010, PS 5010
.....				
SETTINGS QUERY	SET?		Returns all instrument settings that may be queried.	DC 5009, DC 5010, DM 5010, FG 5010, PS 5010
.....				
SELF-TEST	TEST		Performs self-test and returns 0 if OK or code if error detected.	DC 5009, DC 5010, FG 5010, PS 5010
			Returns "ERR 0;" if calibration checksum OK; "ERR 351;" if not.	DM 5010
.....				

## STATUS AND ERROR REPORTING

## STATUS TABLE

STATUS BYTE (DECIMAL)	ERROR CODE (ERR?)	DESCRIPTION
<b>Command Errors</b>		
97	101	Command header error
97	102	Header delimiter error
97	103	Command argument error
97	104	Argument delimiter error
97	105	Non-numeric argument (numeric expected)
97	106	Missing argument
97	107	Invalid message unit delimiter
97	108	Checksum error
97	109	Bytecount error
<b>Execution Errors</b>		
98	201	Command not executable in local
98	202	Settings lost due to rtl
98	203	I/O buffers full, output dumped
98	204	Settings conflicts
98	205	Argument out of range
98	206	Group execute trigger ignored
98	220	Select error. No card in that slot.
98	231	Not in calibrate mode
98	232	Beyond calibration capability
98	251	Symmetry/frequency conflict
98	252	Amplitude/offset conflict
98	253	Amplitude/AM conflict
98	254	Hold/phase lock conflict
98	255	Hold/frequency conflict
98	256	Phase lock/FM conflict
98	257	Phase lock/VCF conflict
98	258	Gate/mode conflict

STATUS BYTE (DECIMAL)	ERROR CODE (ERR?)	DESCRIPTION
<b>Internal Errors (displayed in front panel)</b>		
99	301	Interrupt fault
99	302	System error
99	303	Math pack error
99	311	Timeout (measurement not completed)
99	312	Measurement overflow
99	313	Serial I/O fault
99	314	Mag-latch relay strobe too long
99	315	Phase lock range error
99	316	Frequency correction range exceeded
99	317	Front panel time out
99	318	Bad calibration constant
99	320-339	Device-dependent (see instrument manual)
*	340	System RAM error
*	341	System RAM error (low nibble)
*	342	Reserved for additional RAM errors
	349	
*	350	CPU RAM error
*	351	Calibration RAM checksum error
*	360-375	XXXX ROM placement error
*	380-395	XXXX ROM checksum error

**System Events**

65	401	Power on
66	402	Operation complete
67	403	User request

**Execution Warning**

*	521	Displayed during signature analysis
---	-----	-------------------------------------

GPIB PROGRAMMING GUIDE

STATUS BYTE (DECIMAL)	ERROR CODE (ERR?)	DESCRIPTION
<b>Internal Warning</b>		
102	601	Overrange
102	602	Channel A protect
102	603	Channel B protect
102	604	No prescaler
102	605	Time of day clock not initialized and WAIT UNTIL command was to be executed
<b>Device Status</b>		
193	701	Below limits
195	703	Above limits
193	711	Channel A overflow
194	712	Channel B overflow
197	721	Negative supply change to voltage regulated
198	722	Negative supply change to current regulated
199	723	Negative supply change to unregulated
201	724	Positive supply change to voltage regulated
202	725	Positive supply change to current regulated
203	726	Positive supply change to unregulated
205	727	Logic supply change to voltage regulated
206	728	Logic supply change to unregulated
207	729	Logic supply change to unregulated
204	731	In lock
200	732	Not locked



GPIB PROGRAMMING GUIDE

STATUS BYTE (DECIMAL)	ERROR CODE (ERR?)	DESCRIPTION
225	74x **	Power on errors on card in slot x (50M70) -- Cannot clear PIA data direction registers (50M30) -- Cannot write "1's" to PIA (50M40)
225	741	Cannot access PIA with all "0's" (SI5010)
225	75x **	Power on errors on card in slot x (50M70) -- Cannot properly operate PIA control register (50M30) -- Cannot write "0's" to PIA (50M40)
225	751	Cannot access PIA with all "1's" (SI5010)
225	76x **	Power on errors on card in slot x
226	77x **	Hardware errors on card in slot x
226	78x **	Hardware errors on card in slot x
192+x	79x **	Armed event warning on card in slot x -- Armed SRQ
193	791	Armed SRQ (EXIT TRIG occurred) (SI5010)
128	0	Not in phase lock trigger status, note available mode
128	0	Active, no errors to report
129	0	Trigger input low
130	0	Trigger input toggling
131	0	Trigger input high
132	0	Reading available
136	0	Waiting for trigger
137	0	Phase lock mode, out of lock, trigger low
138	0	Phase lock mode, out of lock, trigger toggling
139	0	Phase lock mode, out of lock, trigger high
142	0	Phase lock mode, in lock, trigger toggling
202	0	Generator went out of phase lock
206	0	Generator went into phase lock

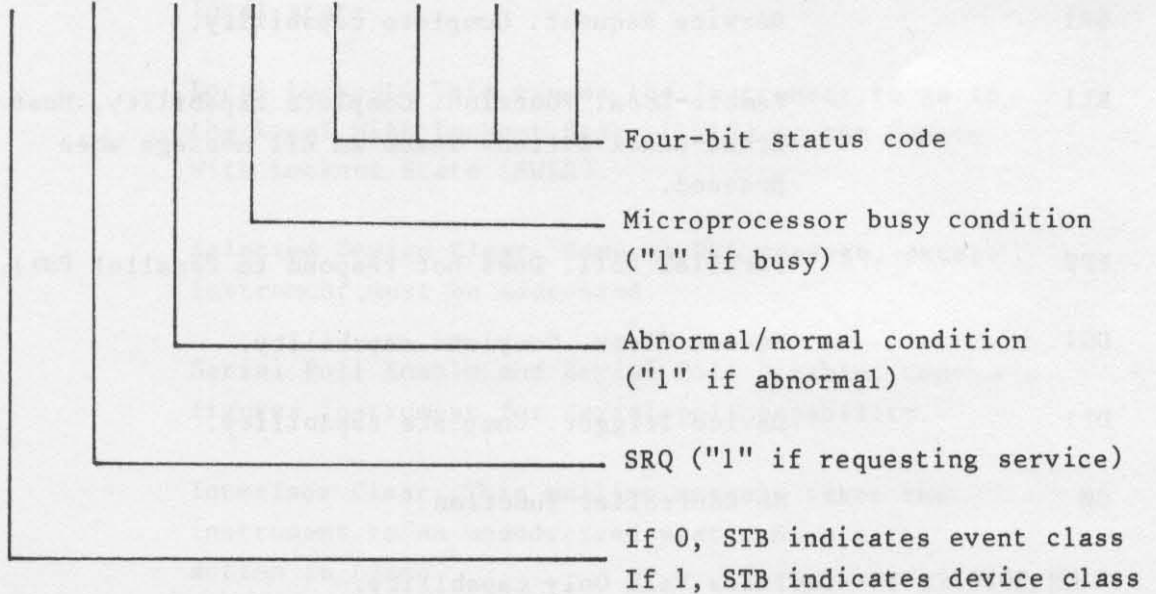
\* Not reported over GPIB so no entry in Status Table Required.  
(Error Code displayed on front panel)

\*\*The value of "x" in the ERROR QUERY response column depends on which slot the MI5010 cards are located. The range of "x" is 1 through 6.

NOTE: The status byte numbers assume SRQ has been asserted and the BUSY bit is not set; add 16 for busy status.

STATUS BYTE (response to serial poll)

MSB	8	7	6	5	4	3	2	1	Condition
(128)	(64)	(32)	(16)	(8)	(4)	(2)	(1)	LSB	
0	1	0	X	0	0	0	0	1	Power-on (65-81)
0	X	0	X	X	X	X	X	X	System event (64-79,80-95)
0	0	0	X	0	0	0	0	0	Ordinary operation (0)
0	X	1	X	0	0	0	0	1	Command error (97,113)
0	X	1	X	0	0	1	0	0	Execution error (98,114)
0	X	1	X	0	0	1	1	1	Internal error (99,115)
0	X	1	X	0	1	1	0	0	Internal error warning (102,118)
1	X	1	X	X	X	X	X	X	Device status (192-207, 208-223)



TM 5000 IEEE 488 INTERFACE FUNCTIONS

FUNCTION SUBSET	DESCRIPTION
SH1	Source Handshake. Complete capability.
AH1	Acceptor Handshake. Complete capability.
T6	Basic Talker. Responds to Serial Poll, Untalk if My Listen Address (MLA) is received.*
L4	Basic Listener. Unlisten if My Talk Address (MTA) is received.
SR1	Service Request. Complete capability.
RL1	Remote-Local Function. Complete capability. Most front-panel buttons issue an RT1 message when pressed.
PPØ	Parallel Poll. Does not respond to Parallel Poll.
DC1	Device Clear. Complete capability.
DT1	Device Trigger. Complete capability.
CØ	No Controller function.

\* DM 5010 is T6--includes Talk Only capability.

TM 5000 RESPONSE TO IEEE 488 INTERFACE MESSAGES

INTERFACE MESSAGE	INSTRUMENT RESPONSE
DCL	Device Clear. This message terminates device dependent message processing, resets the input and output buffers, clears any buffered settings and clears the instrument status, except for the power on status.
GET	Group Execute Trigger. The instrument responds as defined by the DT command.
GTL	Go To Local. This causes the instrument to go to a local state.
LLO	Local Lockout. This causes the instrument to go to the Local With Lockout State (LWLS) or the Remote With Lockout State (RWLS).
SDC	Selected Device Clear. Same as DCL message, except instrument must be addressed.
SPE, SPD	Serial Poll Enable and Serial Poll Disable. Configures instrument for serial poll capability.
IFC	Interface Clear. This uniline message takes the instrument to an unaddressed state. No other action is taken.

INSTRUMENT MESSAGE

INSTRUMENT MESSAGE

INSTRUMENT MESSAGE

INSTRUMENT MESSAGE

Device Class: This message contains device class information and device class name. The device class name is the name of the device class that the instrument belongs to. The device class name is used to identify the instrument in the system.

100

Device Class: This message contains device class information and device class name. The device class name is the name of the device class that the instrument belongs to. The device class name is used to identify the instrument in the system.

101

Device Class: This message contains device class information and device class name. The device class name is the name of the device class that the instrument belongs to. The device class name is used to identify the instrument in the system.

102

Device Class: This message contains device class information and device class name. The device class name is the name of the device class that the instrument belongs to. The device class name is used to identify the instrument in the system.

103

Device Class: This message contains device class information and device class name. The device class name is the name of the device class that the instrument belongs to. The device class name is used to identify the instrument in the system.

104

Device Class: This message contains device class information and device class name. The device class name is the name of the device class that the instrument belongs to. The device class name is used to identify the instrument in the system.

105

106

## MANUAL CHANGE INFORMATION

At Tektronix, we continually strive to keep up with latest electronic developments by adding circuit and component improvements to our instruments as soon as they are developed and tested.

Sometimes, due to printing and shipping requirements, we can't get these changes immediately into printed manuals. Hence, your manual may contain new change information on following pages.

A single change may affect several sections. Since the change information sheets are carried in the manual until all changes are permanently entered, some duplication may occur. If no such change pages appear following this page, your manual is correct as printed.

## MANUAL CHANGE INFORMATION

At the time we originally filed our 1987 Schedule C, we reported that we had received a letter from the IRS dated 1/15/88, advising us that we had been selected for an audit. We have since received a letter from the IRS dated 1/15/88, advising us that we have been selected for an audit. We have since received a letter from the IRS dated 1/15/88, advising us that we have been selected for an audit.

As a result of the audit, we have received a letter from the IRS dated 1/15/88, advising us that we have been selected for an audit. We have since received a letter from the IRS dated 1/15/88, advising us that we have been selected for an audit. We have since received a letter from the IRS dated 1/15/88, advising us that we have been selected for an audit.

A single change to our 1987 Schedule C was made as a result of the audit. The change was made to the amount of the deduction for the cost of goods sold. The amount of the deduction was increased from \$10,000 to \$12,000. The amount of the deduction was increased from \$10,000 to \$12,000. The amount of the deduction was increased from \$10,000 to \$12,000.

