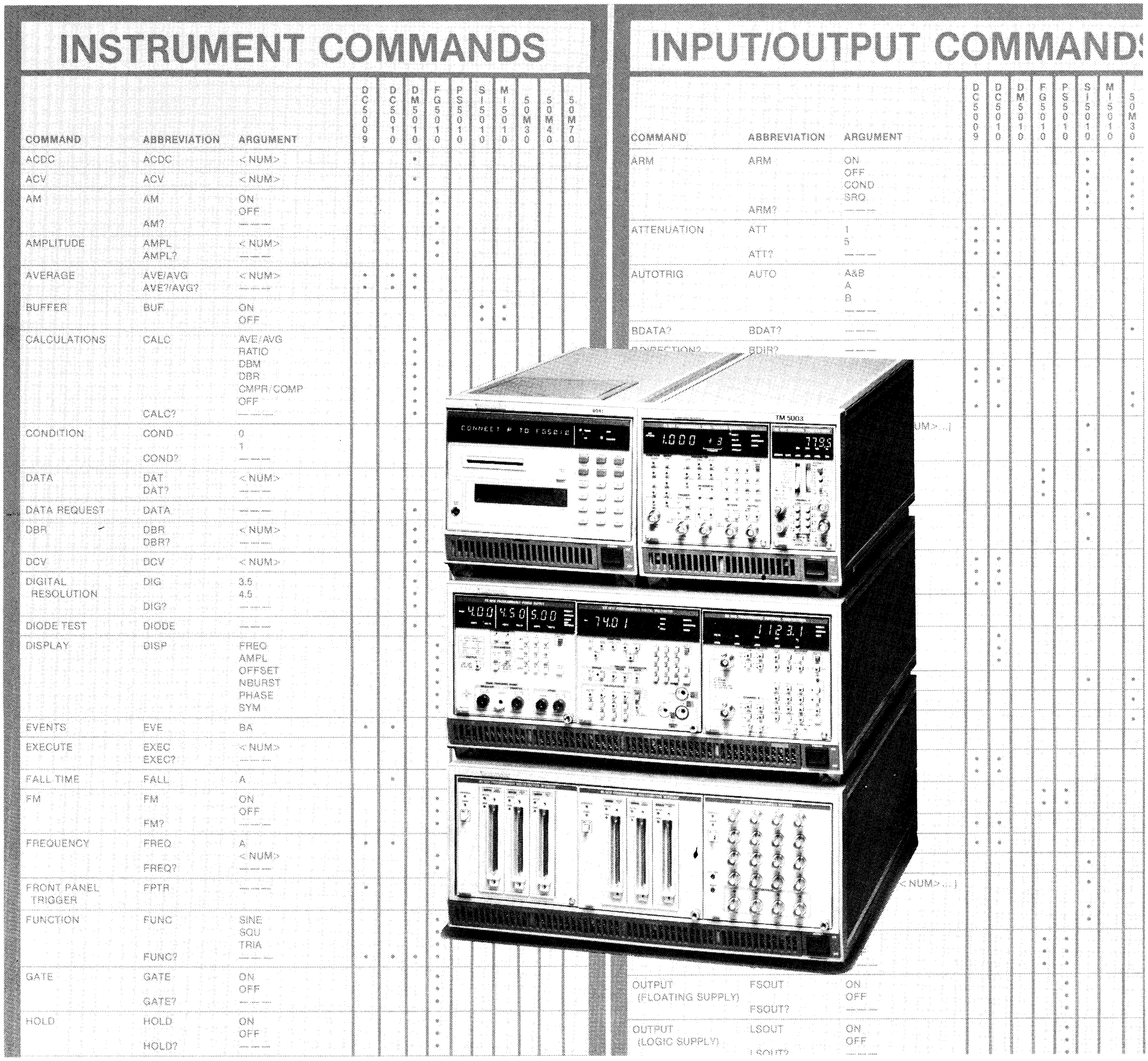
TEK APPLICATION INFORMATION

VOL. 6 NO. 2
FALL 1981

HANDSHAKE

NEWSLETTER OF SIGNAL PROCESSING AND INSTRUMENT CONTROL



Introducing TM 5000 Programmable Instruments—

a systematic approach to measurement systems

Tektronix®
COMMITTED TO EXCELLENCE

HANDSHAKE
NEWSLETTER OF SIGNAL PROCESSING AND INSTRUMENT CONTROL

Table of contents

Return the reply card in this issue for a free subscription to HANDSHAKE.

Managing Editor: A. Dale Aufrecht
Editing and Writing Staff:
Bob Ramirez
Mark Tilden
Graphics: Joann Crook
Production: Brenda Smethers

HANDSHAKE is published quarterly by the **HANDSHAKE** Group. Permission to reprint material in this publication may be obtained by writing to:

HANDSHAKE Editor
Group 157 (94-384)
Tektronix, Inc.
P.O. Box 500
Beaverton, Oregon 97077

Trimming the hidden cost of measurement automation

The goal of HANDSHAKE has always been to provide useful and reliable information on signal processing and instrument control. For the most part, the approach has been to discuss specific applications and application programs. It's been a look at how to do it.

Occasionally, however, it's necessary to step back and take a look at what to do it with. Measurement success is as much a matter of the right tools as the right technique. In this issue of HANDSHAKE, we take a look at some new tools—the new Tektronix TM 5000 line of programmable IEEE-488 compatible instruments and the new TEKTRONIX 4041 System Controller.

In considering any instrument, basic measurement capability will, of course, be the first criterion of selection: Can the instrument make the measurement? But, for programmable instrumentation, there is an equally important second question: Can you make the instrument make the measurement? Or, more specifically, what will your measurement program development time be?

That latter question—program development time—is an area that TM 5000 instruments have been designed to address. This issue of HANDSHAKE takes a close look at those features that reduce program development time and, hence, trim away the hidden cost of measurement automation. And, with this background, future issues will look at specific applications and application programs.

If you would like to receive these future issues and are not already on the HANDSHAKE mail list, be sure to fill out and mail in the reply card bound into the center of this issue. There is no charge for HANDSHAKE, and it is published on a quarterly basis.

TM 5000—an instrument concept for automation ease

TM 5000 is the nomenclature for a new, broad-based line of IEEE-488 compatible test and measurement instruments from Tektronix, Inc. It's a line of programmable instruments designed specifically to give you—

- capability in the R&D lab,
- flexibility on the designer's bench,
- standardization on the manufacturing floor,
- and programming ease for productivity in any atmosphere.

How is all of this put into a single instrument line?

Begin with modularity for configurability

The TM 5000 concept begins with a broad base of instrument types. Take the most commonly needed ones to start the base—a power supply, a digital multimeter, a universal counter/timer, and a function generator. Then add to the base some signal handling units—a high-frequency (to 350 MHz) software configurable switching matrix and a multifunction switching/control unit for interfacing to relay drivers, foot switches, steppers, or any other apparatus. Package everything in a standard-size module, a size that is a submultiple of the standard industrial instrument rack width. Then build power modules that two, three, or more of the instruments will plug into.

The result is the neat, compact instrument package shown in Fig. 1. It's portable and rugged for field service needs. It's a space saver on the designer's bench. It's easily rackmountable for instrument van, shipboard, or production floor use. And, because the instruments plug into the power module, you don't have to unstack or unrack to change instruments. Just plug in the instrument configuration you want.

You can even mix the long-established line of manually operated TM 500 plug-in instruments with your new TM 5000 programmables. And all TM 5000 instruments are UL listed.

Add IEEE-488 compatibility and programmability

Convenient, flexible packaging is just the beginning. The TM 5000 concept continues by making each instrument IEEE-488 compatible and fully programmable.

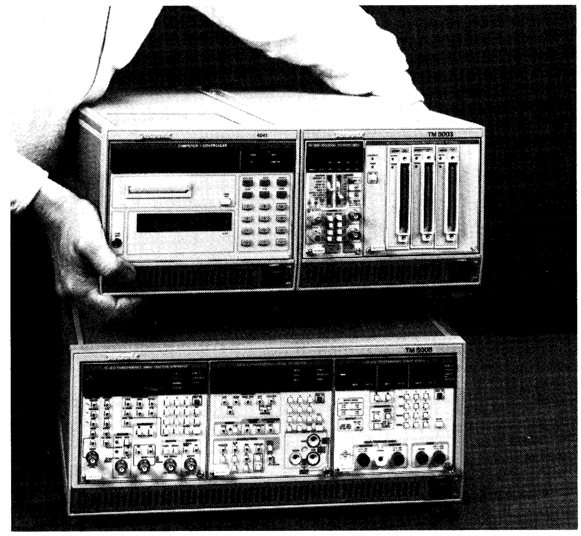


Fig. 1. *The TM 5000 concept makes instruments portable, stackable, or rackable.*

Each TM 5000 instrument contains an interface that conforms to IEEE Standard 488-1978. An IEEE-488 bus, more commonly referred to as the General Purpose Interface Bus or just GPIB, extends across the back plane of the TM 5000 power module and goes to a common GPIB (IEEE 488) connector. This back-plane bus saves instrument cabling time and confusion—just plug the TM 5000 instruments in and they are connected to the power module's single GPIB cable.

Although the power module provides a single GPIB cable for several instruments, bus addressing still remains simple. You don't have to address the power module, then secondary address plug-in instruments. Each TM 5000 plug-in instrument is still individually addressable through its own primary address.

Each TM 5000 instrument also has a readily accessible set of switches for setting its primary address (Fig. 2). These switches also allow selection of one of two message terminator options for compatibility with the greatest number of instrument controllers. And, for even more convenience, TM 5000 instruments with displays allow you to quickly check the address and terminator settings by pressing the ID button on the instrument's front panel.

TM 5000 programmable instruments are designed for easy use by people...there are no obscure or specialized symbols.

Even though they are usually connected to an instrument controller, TM 5000 instruments can still be operated manually. You can still manually select voltage levels, frequencies, measurement functions, etc. from the instrument front panel.

But you can also set up and operate all the instrument functions under program control through each instrument's GPIB interface. For example, you can set the FG 5010 function generator for a 2-volt peak-to-peak 3-KHz sinusoidal output by pressing the following front-panel button sequence: FREQ, 3, 0, 0, 0, ENTER, AMP, 2, ENTER. Or you can just send it the following message sequence over the bus: FREQ 3000;AMP 2. The internal microprocessor takes care of interpreting the messages and setting up the instrument.

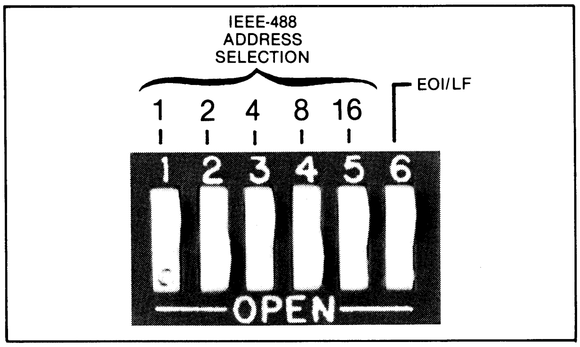


Fig. 2. *Switchable addressing and message termination is just one of many conveniences that enhance system configurability and compatibility.*

In fact, the internal microprocessor is quite tolerant when it comes to message interpretation. To program a frequency measurement, for example, you can send the TM 5000 counter/timer any of the following messages: "FREQUENCY A; FREQ A;FREQ;FRE;FR." Each of these message forms will be correctly interpreted by both the DC 5009 and DC 5010 counter/timer.

Each instrument's microprocessor takes care of a lot of other things for you too. Whenever an instrument is powered up, the microprocessor runs diagnostics to check general instrument operation. Whenever settings are entered, either manually or under program control, the microprocessor checks them to make sure they are valid combinations and in-range. If they aren't, an error code is generated for use over the bus. Plus, the microprocessor assesses and stores instrument status for a variety of operations and conditions. You can use this status information in your programs to monitor or change the direction of measurement sequences.

The internal microprocessor in each instrument also offers the opportunity for some additional measurement features. For example, the DC 5010 Programmable Universal Counter/Timer can make rise-time measurements as well as the standard counter/timer measurements. And, as another example, the DM 5010 Programmable Digital Multimeter can make several calculations, including decibel conversions, from measurements or entered constants.

Make it easy to program

Realizing that the key to productivity is still people, TM 5000 instruments are designed for easy use by people. The front-panel controls are laid out in logical groupings. Each control is labeled with obvious mnemonics describing its function. There are no obscure or specialized symbols.

The same approach is taken in the programming messages for each instrument—no obscure or specialized symbols. The messages are descriptive abbreviations of the front-panel labels and instrument functions (Fig. 3). For example, to set the DC 5010 to measure the frequency of the signal at Channel A, just push the FREQ A button or send it the message FREQ A over the GPIB. It's just that simple. The instrument commands are designed for the convenience of people, not microprocessors.

If you know how to manually operate a TM 5000 instrument, you already know the majority of control messages for programming it. And the rest of the messages are just as easy.

Take SET? for example. To ask any TM 5000 instrument what its current control settings are, just send it SET? over the bus. The instrument responds by assembling a message string describing the instrument's current setup, including several internal conditions. This settings message can be stored by program in a single string variable and used later to duplicate

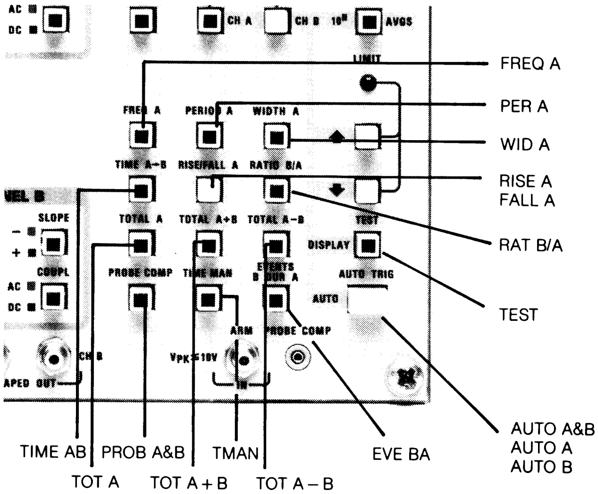


Fig. 3. *Section of DC 5010 Programmable Universal Counter/Timer front panel with associated control messages indicated—the direct relationship between instrument functions and instrument control messages makes TM 5000 programming natural and easy.*

the test setup under program control. Dozens of setups can be stored and executed as needed.

SET? can also be used to the same effect with other GPIB-compatible instruments from Tektronix. But don't confuse the SET? capability with the IEEE Standard 488 requirements. SET?, plus several other convenient instrument query capabilities, is specified by TEK Codes and Formats. TEK Codes and Formats is a Tektronix standard that takes up where the IEEE 488 standard leaves off. English-like commands, common message structures, common data formats—all of these and more are specified by TEK Codes and Formats.

The consistency engendered by TEK Codes and Formats ensures compatibility beyond the IEEE 488 standard. Not only are IEEE-488 compatible instruments from Tektronix easier to program individually, but they are easier to operate and program as a system.

Provide a variety of controller capabilities

Since TM 5000 instruments are GPIB compatible, they can be cabled up to any GPIB instrument controller. Switches on each TM 5000 instrument let you set the instrument to the message terminator required by a particular controller. So GPIB compatibility is maintained, no matter what your controller choice might be.

However, the greatest degree of compatibility—and programming ease—is achieved when TM 5000 instruments are interfaced to Tektronix supplied controllers. That's because Tektronix controllers go beyond mere GPIB compatibility. They are optimized for instrument control, and particularly for instruments following TEK Codes and Formats. The result is a total systems approach to compatibility, capability, and ease-of-use.

Three Tektronix instrument controllers are currently available for use with your TM 5000 instruments. All use an extended BASIC language for easy, yet powerful, instrument control and measurement processing.

There's the new 4041 System Controller with a detachable keyboard and program-definable front-panel buttons (Fig. 4). It's packaged like TM 5000 instruments so that it fits easily on a designer's bench with its keyboard for program development. Or the keyboard can be detached and the 4041 rackmounted for push-button program selection on the production floor.

Then there are the 4050-Series Graphic Computing Systems. These offer a controller, keyboard, display screen, and magnetic tape storage, all integrated into a single desk-top or pedestal-mounted unit. Additional computing, analysis, and graphics capabilities are available through an extensive program library and numerous ROM packs, including signal processing ROM packs.

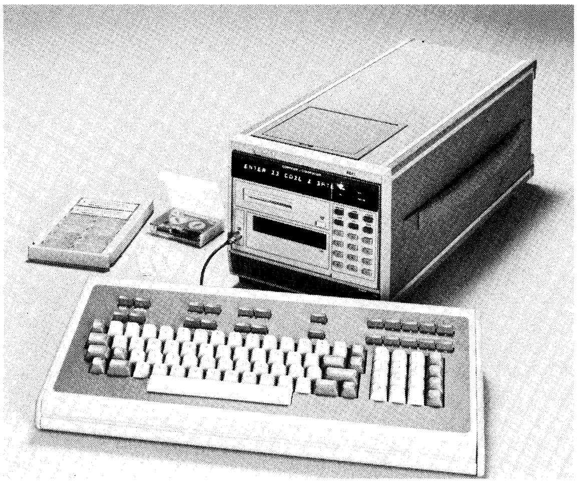


Fig. 4. *The new TEKTRONIX 4041 System Controller with its detachable program development keyboard.*

Or, you can choose TEK SPS BASIC software to base your system on. TEK SPS BASIC is a complete instrument control and signal processing software package designed to work with DEC* PDP-11 series minicomputers using Tektronix supplied IEEE-488 Interfaces.

**DEC and PDP are registered trademarks of the Digtal Equipment Corporation.*

For help in selecting the instrument controller and TM 5000 instrument configuration to meet your measurement needs, contact your local Tektronix Sales Engineer or Sales Representative. Or simply tell us, by checking the appropriate box on the HANDSHAKE reply card, that you want to see a Tektronix Sales Engineer. We'll pass the word on for you.

By Bob Ramirez,
HANDSHAKE Staff.

The New TEKTRONIX 4041 System Controller—

A small package with total flexibility

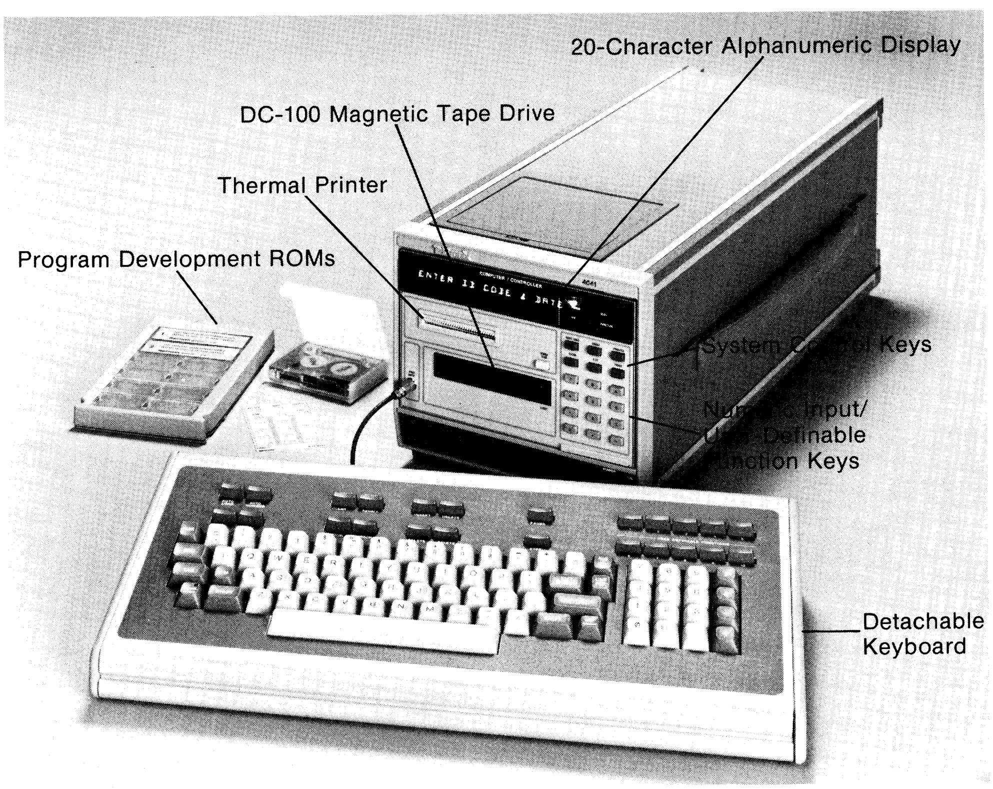


Fig. 1. *The 4041 System Controller and it's major operator features.*

The 4041 System Controller introduces a new concept in IEEE-488 compatible instrument controllers. It's a concept that lets you configure the controller to meet your needs exactly. No compromise!

Need a rugged rack-mounted instrument controller with push-button simplicity and tamper-proof programs for production floor use? You can set up the 4041 System Controller to be just exactly that. Or it can be a bench-top controller with an easy-to-use language that lets you work on engineering problems instead of programming technique. Or it can be a controller with the operating system capabilities required for sophisticated program development. Just choose the features you need and configure them.

Push-button simplicity

In its simplest form, the TEKTRONIX 4041 System Controller is an execute-only controller. Previously developed instrument control and measurement routines are loaded via the DC-100 magnetic tape cartridge drive (Fig. 1).

Program loading and simple control is provided through five front-panel buttons—AUTO LOAD, ABORT, CLEAR, PROCEED, and PAUSE. Any other operator action or input beyond the functions of those five buttons is controlled by the program in use. Application programs can be written to execute without any further operator action or input. Or application programs can be written to allow varying degrees of operator interaction.

For interaction, the program can convey short messages to the operator via a 20-character LED display. Or longer messages can be output, 20-character line at a time, to the thermal printer.

The messages can be anything, from instructions on what test points to probe, to a request for numeric input, or a menu of several application programs that can be selected via the user-definable keys.

The numeric input key pad consists of the digits 0 through 9, -, ., and EEX for entering exponents. How the key pad is used is determined by the application program. If the program is set up to ignore key-pad input, then pressing the numeric keys has no effect. However, if the program asks the operator for a numeric value, the keys become active for numeric input. As a third possibility, keys 0 through 9 can be programmed to allow operator selection of various instrument control and measurement routines. This latter mode is referred to as the user-defined mode.

It is important to note that the 4041 front-panel buttons deal with program operation only. They cannot be used to modify or generate program code. In a low-skilled operator environment, this keeps your application programs tamper-proof. Yet, program development or maintenance people can easily access the 4041 for program development, modification, or debugging simply by installing the program development ROMs and plugging in the optional program development keyboard.

Of course if you are going to use the 4041 as a bench-top engineering tool, you'll want to leave the program development ROMs and keyboard installed. Or, for engineering applications requiring extensive listings or graphics, an RS-232C interfaced graphics terminal can be used with the 4041.

If you need expanded instrument and peripheral capabilities, you can add a second set of GPIB and RS-232 ports.

Microprocessor power, plenty of memory

The 4041 control and processing functions are carried out by three microprocessors, with the central processing unit being the Motorola 68000. Although the 16-bit 68000 is more than adequate in most situations, some 32-bit features are occasionally needed, for example in memory addressing. So the 4041 is designed to provide those features, like 24-bit direct memory addressing so that you don't have to use confusing paging techniques.

Standard memory size for the 4041 is 32 kilobytes with about 25 kilobytes being user addressable. However, additional internal RAM memory is available in 32-kilobyte increments.

With all optional memory installed, the 4041 has 160 kilobytes of internal memory. That's plenty for most applications. Still, you may need more for very large data logging operations. So additional peripheral memory—tape drives, disks, etc.—can be interfaced to the 4041 via its IEEE-488 port.

Interfacing flexibility

The 4041 System Controller comes standard with one IEEE-488 interface port and one RS-232C interface port. The IEEE-488 (or GPIB) port allows you to interface and control up to 14 GPIB-compatible instruments with the 4041 (see Table 1 for implemented functions). The RS-232 port provides the additional capability of interfacing a terminal or any other RS-232 compatible peripheral to the 4041 System Controller. All standard RS-232 transmission rates, from 75 to 9600 baud, are supported and can be selected under program control.

Additional interfaces are also available as options.

One optional interface capability is the addition of a TTL, 8-bit parallel interface port. This allows interfacing and control of a variety of custom or nonstandard devices. It also allows for driving displays, parts handlers, etc.

Instrument and controller friendliness is perhaps the most visible aspect of TEK Codes and Formats.

Or, if you need expanded instrument and peripheral capabilities, you can add a second set of GPIB and RS-232 ports in lieu of the TTL interface. Having two GPIB ports available increases your instrument control options dramatically. With two ports, you can drive up to 28 instruments, 14 per port. Plus, you can increase throughput by segregating fast instruments to one GPIB port and slow instruments to the other.

Another advantage of the optional second GPIB port is that it adds DMA (direct memory access) capability for high-speed transfers such as needed in data logging. The supported DMA transfer rate exceeds 600 kilobytes/sec on input and 240 kilobytes/sec on output. This, along with the standard Interrupt Mode transfer rate (greater than 5 kilobytes/sec, input and output) and Fast Mode transfer rate (greater than 16.5 kilobytes/sec on input and 19.5 kilobytes/sec on output), gives you three transfer rates to operate with.

Instrument compatible, people compatible

There's no denying that IEEE Standard 488 is an important step in making automated test equipment an off-the-shelf possibility. But too often it is forgotten that the standard specifies only an interface, not a system. And, even as an

TABLE 1

IEEE 488-1978 Standard functions implemented in the 4041 GPIB interface

FUNCTION	SUBSET	CAPABILITY
Source Handshake	SH1	Complete
Acceptor Handshake	AH1	Complete
Basic Talker	T6	Basic talker, serial poll, unaddress if MLA (my listen address) is received
Basic Listener	L4	Basic listener, unaddress if MTA (my talk address) is received
Service Request	SR1	Complete
Remote-Local	RL1	Complete
Parallel Poll	PP1	Remote configuration
Device Clear	DC1	Complete
Device Trigger	DT1	Complete
Controller	C1	System controller
Controller	C2	Send IFC and take charge
Controller	C3	Send REN
Controller	C4	Respond to SRQ
Controller	C9	Send IF messages, receive control, pass control, parallel poll, take control synchronously

interface, there is plenty of leeway in how the standard can be implemented. The result is differences in interfaces. These differences are not so much a matter of compatibility as they are a matter of capability. How many of the interface subsets are implemented? How much control capability does the implementation provide?

GPIB compatibility with minimal capability can reduce interface design costs. But the danger of minimal capability is that it can create more systems designers than measurement systems!

At Tektronix, GPIB products are designed to be GPIB capable as well as compatible. This design philosophy is detailed in the TEK Codes and Formats Standard, a Tektronix specification that augments IEEE Standard 488. TEK Codes and Formats specifies the message structure and syntax to be used in GPIB communication between Tektronix instruments and instrument controllers. The intent is to provide the highest possible degree of consistency and programming ease while still being receptive to the communication methods used by other instruments on the market. The result is a line of GPIB products, including the 4041 System Controller, that is highly capable and very friendly, both with other GPIB instruments and with the people using them.

Instrument and controller friendliness is perhaps the most visible aspect of TEK Codes and Formats. Things are done in the most universally accepted and familiar manner.

For example, data from Tektronix GPIB-compatible instruments is primarily ASCII coded with binary data being sent as an option or in cases where it provides an obvious advantage. And Tektronix controllers are streamlined for ASCII-coded data transfers, although they are also capable of efficiently dealing with binary data should the need arise. Instrument control messages are English abbreviations of the instrument functions being controlled, so the message repertoire is easy to understand and learn. And Tektronix instrument controllers all use BASIC for their operating system. The result is instruments, controllers, and systems that understand English.

4040-Series BASIC, a language of convenience

TEKTRONIX 4040-Series BASIC is the operating system for the 4041 System Controller. It is based on the standard, widely known and widely used language of BASIC. What makes 4040 BASIC different is that it has been enhanced for easier instrument control and for broad appeal to a variety of programming needs and skills. The special features of 4040 BASIC are too numerous to mention here, but the following is a synopsis of some of the more popular and far-reaching aspects of this new instrument control package.

Eight character variable names. While most programming languages allow use of two character variable names, 4040 BASIC allows use of up to eight characters for variable naming. Thus, instead of being restricted to something like RT for storing rise-time data, you can spell it out—RISETIME.

For anyone who has tried to decipher the listing of a program written by someone else, or even their own programs several months after they've been written, the advantage of full variable naming is obvious. Instead of trying to remember the meaning of cryptic notations such as RT, FT, TP, BA, L1, L9, you can assign and list variable meanings directly—RISETIME, FALLTIME, TOP, BASE, LEVEL10, LEVEL90, etc.

Named program lines. 4040 BASIC is a statement-oriented language, with each statement preceded by a line number and program execution normally progressing in line-number order. The order of execution can, however, be changed by branching ahead or back with program control statements such as GOTO, GOSUB, IF...THEN...ELSE..., etc.

To help you keep track of branches, 4040 BASIC lets you label statements and branch to the labels instead of to the statement line number. As an example of the usefulness of this, consider a program that requires polar conversion of complex numbers at various stages of execution.

Usually, a single polar conversion routine will be written and placed at a high line number position, say 10000. Then, whenever a polar conversion is needed in the main program, it can be computed by branching to the polar conversion subroutine with a GOSUB 10000. That's standard BASIC. The trouble with it is, when you are building programs, debugging them, modifying them, or just trying to understand what's going on from a listing, you have to remember what each subroutine does or find the subroutine in the

Subprograms can be written to operate completely independent of the main program.

listing and hope for an explanatory remark. And there can be a lot of subroutines and subroutine calls in a large program.

With 4040 BASIC you can make things a lot easier to follow. Just label the first statement of the subroutine with a word or word combination that explains the operation. For example, the subroutine can look like this:

```
10000 POLARCON:MGNITUDE=SQR(A*A+B*B)
10010 PHASE=ATAN(B/A)
10020 RETURN
```

Then you can call the subroutine simply by using GOSUB POLARCON wherever needed in the main program.

This makes for easier programming and program modification. You don't have to remember beginning line numbers for routines. Just their labels. And, if you have to change the line numbers of the subroutine, you don't have to worry about all the GOSUBs calling it since the branch is to a label instead of a line number.

Any statement in 4040 BASIC can be labeled and branched to by label with any of the standard BASIC branching statements. Of course, if you prefer, you can ignore labeling and branch to line numbers instead. Whatever is convenient for you is convenient for 4040 BASIC.

User-defined functions. The functions standard to most BASICs—SIN, COS, SQR, etc.—are also provided in 4040 BASIC. However, there are occasions when additional special-purpose functions are needed.

With 4040 BASIC, you can write routines for special-purpose functions, name them, and use them in the same manner as the standard functions. As an example, let's say you frequently need to compute average power from measurements taken on reactive networks. So, for your purposes, it would be nice to have a PAVE (average power) function. To implement such a function in 4040 BASIC, simply write a routine for computing average power and then precede the routine with a function declaration. This declaration includes the word FUNCTION, gives the name of the function (PAVE for example), and specifies the input variables. Once you've written the function and declared it, you can compute average power by invoking PAVE the same way you would any other function. The program pattern is indicated in Fig. 2.

FORTRAN-like subprograms. When it comes to implementing large programs, subprograms can save a lot of time and expense. This is because subprograms can be written to operate completely independent of the main program, with the only exception being the passing of input and output parameters between the main program and the subprogram. This means that you can initially design a large program as a set of functional blocks with specified inputs and outputs. Then the actual programming task can be attacked block-by-block in more manageable chunks or done in parallel by several programmers, each working independently to satisfy the input-output requirements of their assigned subprograms.

Local variables can be declared. Most forms of BASIC use global variables only. That is, the value assigned to a variable can be changed at any time the variable is operated on by any portion of the program. Usually, this isn't a problem until programs get very large or several people are involved in writing different portions of the program. Then, you must guard against using the same variable name for different purposes.

```
.
.
200 AVEPOWER=PAVE(VOLTS,VPHASE,CURRENT,IPHASE)
.
.
main program continues
.
20000 FUNCTION PAVE(V,VP,C,IP)
.
.
compute root-mean-square, relative phase, etc.
.
20200 PAVE=VR*IR*COS(PD)
20210 RETURN
20220 END
```

Fig. 2. *With 4040 BASIC, special functions can be written and then invoked just like standard BASIC functions. In this example, PAVE is invoked at 200 and passes voltage, current, and phase information to FUNCTION PAVE at 20000 which computes average power and passes the value back to AVEPOWER.*

With 4040 BASIC, user-defined functions and subprograms allow modular construction and

```
10 REM **** THRESHOLD PLOTTING ****
20 REM **** FOR TEK SPS BASIC ****
30 INITG
40 PRINT "INPUT NAME OF DATA FILE: "\INPUT N$
50 G$=N$&".DAT"
60 PRINT "INPUT DATE:"\INPUT D$
70 PAGE
80 WAVEFORM WA IS A(511),SA,HA$,VA$
90 CLOSE #2
100 OPEN #2 AS DX1:G$ FOR READ
110 PRINT "DO YOU WISH TO FIND THE MAX AND MIN DATA VALUES (Y OR N)"
120 INPUT H$
130 IF H$="N" THEN GOTO 150
140 GOSUB 450
150 PAGE
160 RESET #2
170 PRINT "INPUT THRESHOLD LEVEL (% MAX SIG HEIGHT)"
180 INPUT TH
190 REM - SET VERTICAL STEP SIZE (IN MM)
200 V=2
210 REM - SET UP SIZE OF DISPLAY (NOTE: MUST BE SQUARE)
220 VIEWPORT 100,900,100,750
230 WINDOW 0,100,0,80
240 SETGR VIEW,WIND,NOPLOT,TICS 10,8,5,5,GRAT 2,2,2,2
250 PAGE
260 GRAPH A
270 WINDOW 0,511,0,80
280 SMOVE 700,750\PRINT "THRESHOLD =";TH;"% MAX SIG"
290 SMOVE 100,30\PRINT "THRESHOLD DISPLAY FOR DATA FILE: ";G$
300 SMOVE 800,30\PRINT "[";D$;"]"
310 SMOVE 30,450\PRINT "(MM)"
320 SMOVE 940,80\PRINT "(MM)"
330 Y=0
340 FOR II=1 TO K
350 READ #2,WA
360 FOR X=0 TO 511
370 IF A(X)<TH*DH/100 THEN GOTO 400
380 MOVE X,Y
390 DRAW X,Y
400 NEXT X
410 Y=V*II
420 NEXT II
430 SMOVE 0,0
440 END
450 REM - SUBROUTINE TO FIND DATA MAX AND MIN VALUES
460 PRINT "DATA FILE: ";G$;" NOW BEING SEARCHED FOR MAX AND MIN VALUES"
470 DH=-1000000
480 DL=1000000
490 K=0
500 READ #2,WA
510 EOF #2 GOTO 560
520 IF DH<MAX(A) THEN DH=MAX(A)
530 IF DL>MIN(A) THEN DL=MIN(A)
540 K=K+1
550 GOTO 500
560 RETURN
```

Fig. 2. *TEK SPS BASIC program to produce threshold level plot.*

the values calculated the first time through the program to save time.

Line 170 requests the desired threshold level. This value is used to determine whether a data point is displayed (above the threshold) or not (below the threshold). Lines 190-200 set up the vertical step size and lines 210-230 set up the viewport and the window. Lines 240-320 draw the axes.

The original data file is read again at line 350 and lines 360-440 interpret the data levels against the threshold and display the results.

By P.J. Highmore and B.S. Gray
United Kingdom Atomic Energy Authority
Risley Nuclear Power Development Laboratories
K11/RD3
Risley, Warrington WA3 6AT
England

This application is condensed from an application report entitled **Graphical Representations of Ultrasonic Data Using a Digital Oscilloscope**, *ND-R-411(R). Copies of the report can be obtained by writing Dr. Highmore at the above address.*

GPIB Programming Guide available

This programming guide discusses the concepts of controlling instruments over the GPIB and provides a variety of specific examples using the 4052 Controller and the new TM 5000 line of programmable instruments. The **GPIB Programming Guide** can be ordered by part number 070-3985-00 from any Tektronix Field Office or Sales Representative. The programs listed in the **GPIB Programming Guide** are also available on the **TEKniques Program Library** tape, Vol. 5 No. 4. This 4052-compatible magnetic tape can be ordered by part number 062-5981-01.

HANDSHAKE
Group 157 (94-384)
Tektronix, Inc.
P.O. Box 500
Beaverton, Oregon 97077

BULK RATE
U.S. POSTAGE
PAID
Tektronix, Inc.

Tektronix
COMMITTED TO EXCELLENCE

33A-4886

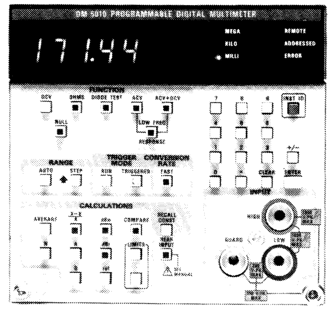
TEKTRONIX TM 5000 PROGRAMMABLE INSTRUMENT FEATURES

Test and measurement systems have never been this easy...to configure...to program.

Programmable TM 5000 features

TM 5000 programmable instruments are designed to be easy to program, easy to configure, and to be compatible with each other and with over 40 TM 500 manual instruments. All TM 5000 programmable instruments include these major features:

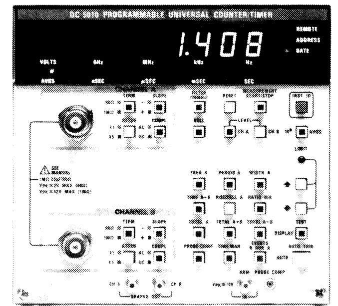
- IEEE 488-1978 Compatible Interfacing
- Tektronix Standard Codes and Formats
- UL Listed
- Exceeds MIL-T-28800B, Class 5, Requirements for Operating Environment
- Program Configurable to Measurement Needs
- English-Like Mnemonic Commands
- Commands Match Front-Panel Nomenclature
- Learn Mode for Easier Instrument Programming
- **INST ID** Button Programmable for Push-Button SRQ Capability
- Internal Diagnostics for Power-Up Self-Test
- Can be used with Manual TM 500 Plug-Ins



DM 5010 Programmable Digital Multimeter

The DM 5010 is a 4 1/2 digit multimeter which measures DC voltage, resistance, true RMS AC voltage, and true RMS AC + DC voltage. Current and dB can be calculated from these readings. Major features include:

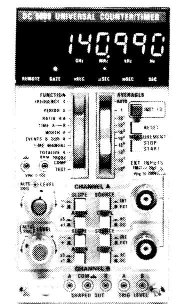
- DC Volts, 0.015% +1 Count
- Ohms, 0.015% +2 Counts
- True RMS (AC + DC)
- Fast-Slow Mode
- Microprocessor Self Nulling
- Averaging
- Offset and Scaling
- dB Conversion
- Hi-Low-Pass Mode
- External Guard
- 10^9-Ohms Input Impedance
- Diode Test
- In-Circuit Ohms Measurement
- Autoranging
- 4 1/2 Digit Display
- **INST ID** Button Displays GPIB Address/ Asserts SRQ
- Self-Contained Keyboard for Data Entry



DC 5010 Programmable Universal Counter/Timer

The DC 5010 is the programmable version of the DC 510 counter/timer. Major features include:

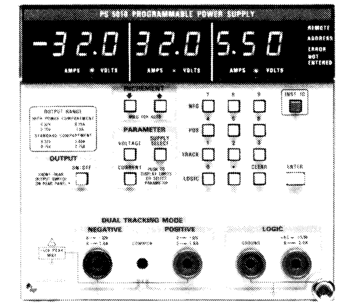
- DC to 350 Megahertz
- Two Identical Input Channels
- Manual or Program Selection of 20 MHz Filter
- 3.125-Nanosecond Clock
- 50 Ohm/1 Megohm Input
- Auto-Trigger
- Ratio Architecture
- Rise-Time/Fall-Time Measurement on Channel A
- Pulse Width Measurements on Channel A
- Null Function
- Probe Compensation
- Hysteresis Compensation
- Auto-Averaging
- Shaped Outputs
- Arming Input
- 9-Digit Display
- **INST ID** Button Displays GPIB Address/ Asserts SRQ



DC 5009 Programmable Universal Counter/Timer

The DC 5009 is the programmable version of the DC 509 and the lowest cost fully programmable universal counter/timer on the market today. Its major features include:

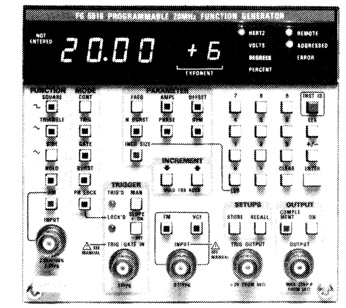
- DC to 135 Megahertz
- Two Input Channels
- 10-Nanosecond Clock
- Auto-Trigger
- Reciprocal Frequency
- Probe Compensation
- Auto-Averaging
- Shaped Outputs
- Arming Input
- Pulse Width Measurements on Channel A
- 8-Digit Display
- **INST ID** Button Displays GPIB Address/ Asserts SRQ



PS 5010 Programmable Power Supply

The PS 5010 is a triple power supply. Major features include:

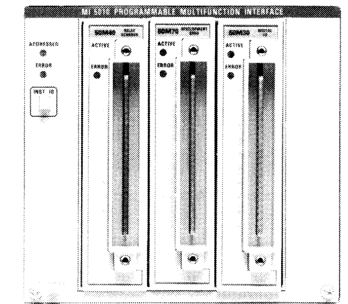
- Triple Output: 0 to -32, 0 to +32, and 4.5 to 5.5 Volts
- Programmable Voltage and Current
- Auto-Crossover with IEEE-488 Bus Interrupt on Constant Voltage or Current Mode Change
- Triple Three-Digit Displays: Volts or Current, and Constant Voltage, Constant Current Indication
- Dual Trackable Floating Supply
  - 0 to 32 Volts, 0.75 Amps Maximum
  - 0 to 15 Volts, 1.6 Amps Maximum
  - 50-Milliamp Current Steps
  - 10-Millivolt Steps, 0 to 10 Volts
  - 100-Millivolt Steps, 10 to 32 Volts
- Logic Supply
  - 4.5 to 5.5 Volts in 10 Millivolt Steps
  - 100-Milliamp Current Steps to 3 Amps
- Programmable Voltage and Current Limits
- Remote Voltage Sensing (Rear Interface Only)
- **INST ID** Button Displays GPIB Address/ Asserts SRQ
- Self-Contained Keyboard for Data Entry



FG 5010 Programmable 20 MHz Function Generator

The FG 5010 is a 20-Megahertz function generator. Major features include:

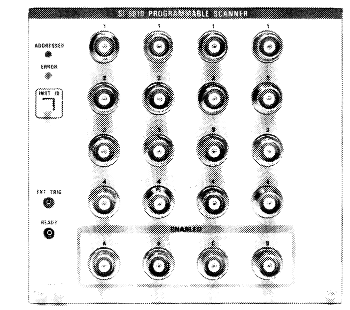
- 0.002 Hertz to 20 Megahertz
- Sine, Square, Triangle Functions
- 20 Millivolts to 20 Volts Peak-to-Peak Open-Circuit Amplitude
- Burst Mode (1-9999 Cycles)
- Programmable Symmetry
- Programmable Phase
- Auto Scan Phase Lock
- Amplitude Modulation, Frequency Modulation, Voltage-Controlled Frequency Capabilities
- Temporary Storage of up to 10 Front-Panel Setups
- Output Complement
- Waveform Hold (200 Hertz or Less)
- Haversine Function in Triggered Mode
- **INST ID** Button Displays GPIB Address/ Asserts SRQ
- Self-Contained Keyboard for Data Entry



MI 5010 Programmable Multifunction Interface

The MI 5010 is a general interface to provide input to or output from a TM 5000 system. Major features include:

- Signal Routing
- Control Functions
- Device Interfacing
- Buffered Mode for Controller-Unattended Operation
- Real-Time Clock
- Triggered Events
- Up to Six Function Cards (with MX 5010 Multifunction Interface Extender)
- 16-Bit Digital I/O (with 50M30 Programmable Digital Input/Output Card)
- 16-Contact Relay Scanner (with 50M40 Programmable Relay Scanner Card)
- Development Card (with 50M70 Programmable Development Card)
- **INST ID** Button Programmable for Asserting SRQ



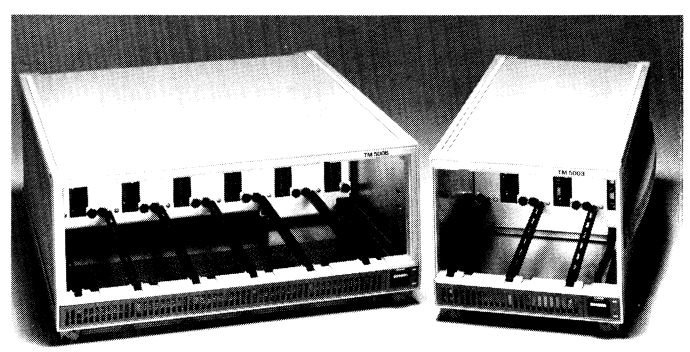
SI 5010 Programmable Scanner Interface

The SI 5010 is a 350-Megahertz, 16-channel switcher/scanner. Major features include:

- Coaxial Switching
- 350-Megahertz Bandwidth (1-Nanosecond Rise Time) in 4-1 Configuration
- Software Reconfigurable as Quad 4 Channel to 1 Channel, Dual 8 Channel to 1 Channel, 16 Channel to 1 Channel
- Buffered Mode for Controller-Unattended Operation
- Real-Time Clock
- Triggered Events
- **INST ID** Button Asserts SRQ

TM 5003 and TM 5006 Mainframe

The TM 5003 Mainframe accepts three single-width plug-ins or a combination of one single-width and one dual-width plug-in. The TM 5006 accepts up to three dual-width plug-ins or six single-width plug-ins (or a combination of these). Either manual TM 500 or programmable TM 5000 plug-ins may be used.



For more information on the Tektronix TM 5000 Series Programmable Instruments contact your local Tektronix Sales Engineer or Sales Representative, or use the reply card bound into this issue of HANDSHAKE.

TM 5000 Programming Reference
Instrument Control and I/O

INSTRUMENT COMMANDS

COMMAND	ABBREVIATION	ARGUMENT	DC5009	DC5010	DM5010	FG5010	PS5010	SI5010	MI5010	50M30	50M40	50M70
ACDC	ACDC	<NUM>			•							
ACV	ACV	<NUM>			•							
AM	AM	ON				•						
		OFF				•						
	AM?	---				•						
AMPLITUDE	AMPL	<NUM>				•						
	AMPL?	---				•						
AVERAGE	AVE/AVG	<NUM>	•	•	•							
	AVE?/AVG?	---	•	•	•							
BUFFER	BUF	ON						•	•			
		OFF						•	•			
CALCULATIONS	CALC	AVE/AVG			•							
		RATIO			•							
		DBM			•							
		DBR			•							
		CMPR/COMP			•							
		OFF			•							
	CALC?	---			•							
CONDITION	COND	0						•	•			
		1						•	•			
	COND?	---						•	•			
DATA	DAT	<NUM>								•		•
	DAT?	---								•		•
DATA REQUEST	DATA	---			•							
DBR	DBR	<NUM>			•							
	DBR?	---			•							
DCV	DCV	<NUM>			•							
DIGITAL RESOLUTION	DIG	3.5			•							
		4.5			•							
	DIG?	---			•							
DIODE TEST	DIODE	---			•							
DISPLAY	DISP	FREQ				•						
		AMPL				•						
		OFFSET				•						
		NBURST				•						
		PHASE				•						
		SYM				•						
EVENTS	EVE	BA	•	•								
EXECUTE	EXEC	<NUM>						•	•			
	EXEC?	---						•	•			
FALL TIME	FALL	A		•								
FM	FM	ON				•						
		OFF				•						
	FM?	---				•						
FREQUENCY	FREQ	A	•	•								
		<NUM>				•						
	FREQ?	---				•						
FRONT PANEL TRIGGER	FPTR	---	•									
FUNCTION	FUNC	SINE				•						
		SQU				•						
		TRIA				•						
	FUNC?	---	•	•	•	•						
GATE	GATE	ON				•						
		OFF				•						
	GATE?	---				•						
HOLD	HOLD	ON				•						
		OFF				•						
	HOLD?	---				•						
ILOGIC	ILOG	<NUM>					•					
	ILOG?	---					•					
INEGATIVE	INEG	<NUM>					•					
	INEG?	---					•					
IPOSITIVE	IPOS	<NUM>					•					
	IPOS?	---					•					
ITRACK	ITRA	<NUM>					•					
LIMITS	LIM	<NUM 1>,<NUM 2>			•							
	LIM?	---			•							
LOW FREQ RESPONSE	LFR	ON			•							
		OFF			•							
	LFR?	---			•							
MANUAL TRIGGER	MTRIG	---				•						
MODE	MODE	CONT				•						
		TRIG				•						
		GATE				•						
		BURST				•						
		LOCK				•						
		PHLOCK				•						
		RUN			•							
		TRIG			•							
	MODE?	---			•	•						
MONITOR	MON	ON			•							
		OFF			•							
	MON?	---			•							
NAME?	NAME?	---								•	•	•
NBURST	NBUR	<NUM>				•						
	NBUR?	---				•						
NULL	NULL	ON		•								
		OFF		•								
		<NUM>			•							
	NULL?	---		•	•							
OFFSET	OFFS	<NUM>				•						
	OFFS?	---				•						
OHMS	OHMS	<NUM>			•							
PERIOD	PER	A	•	•								
PHASE	PHAS	<NUM>				•						
	PHAS?	---				•						
PROBE COMPENSATION	PROB	A&B	•	•								
RATIO	RAT	B/A	•	•								
		<NUM 1>, <NUM 2>			•							
	RAT?	---			•							
READY ?	RDY?	---	•	•	•							
RECALL	REC	<NUM>				•						
RESET	RES	---	•	•								
RISETIME	RISE	A		•								
SELECT	SEL	<SLOT NUM>[, <CARD NAME>]							•			
	SEL?	---							•			
START	START	---	•	•								
STOP	STOP	---	•	•				•	•			
STORE	STOR	<NUM>				•						
		<NUM>,<BIN BLK>				•						
SYMMETRY	SYM	<NUM>				•						
	SYM?	---				•						
TIME	TIME	AB	•	•								
		<HH>:<MM>:<SS>						•	•			
	TIME?	---						•	•			
TMANUAL	TMAN	---	•	•								
TOTALIZE	TOT	A	•	•								
		A+B		•								
		A−B		•								
TRIGGER	TRIG	---						•	•			
	TRIG?	---				•		•	•			
UNTIL	UNT	<HH>:<MM>:<SS>				•		•	•			
	UNT?	---						•	•			
VCF	VCF	ON										
		OFF				•						
	VCF?	---				•						
VLOGIC	VLOG	<NUM>					•					
	VLOG?	---					•					
VNEGATIVE	VNEG	<NUM>					•					
	VNEG?	---					•					
VPOSITIVE	VPOS	<NUM>					•					
	VPOS?	---					•					
VTRACK	VTRA	<NUM>					•					
WAIT	WAIT	TRIG						•	•			
		COND						•	•			
		<NUM>						•	•			
		UNTIL						•	•			
		OFF						•	•			
	WAIT	---						•	•			
WIDTH	WID	A	•	•								

INPUT/OUTPUT COMMANDS

COMMAND	ABBREVIATION	ARGUMENT	DC5009	DC5010	DM5010	FG5010	PS5010	SI5010	MI5010	50M30	50M40	50M70
ARM	ARM	ON						•		•	•	•
		OFF						•		•	•	•
		COND						•		•	•	•
		SRQ						•		•	•	•
	ARM?	---						•		•	•	•
ATTENUATION	ATT	1	•	•								
		5	•	•								
	ATT?	---	•	•								
AUTOTRIG	AUTO	A&B		•								
		A		•								
		B		•								
		---	•	•								
BDATA?	BDAT?	---								•		•
BDIRECTION?	BDIR?	---										•
CHANNEL	CHA	A	•	•								
		B	•	•								
		<NUM>								•		•
	CHA?	---	•	•						•		•
CLOSE	CLO	<NUM> [,<NUM>...]						•			•	
	CLO?	---						•			•	
COMPLEMENT	COMP	ON				•						
		OFF				•						
	COMP?	---				•						
CONFIGURE	CONF	<W>,<X>, <Y>,<Z>						•				
	CONF?	---						•			•	
COUPLING	COU	AC	•	•								
		DC	•	•								
	COU?	---	•	•								
DIRECTION	DIR	<NUM>										•
	DIR?	---										•
FILTER	FIL	ON		•								
		OFF		•								
	FIL?	---		•								
FLAG?	FLAG?	---						•		•	•	•
FSETTINGS?	FSET?	---								•	•	•
HDATA?	HDAT?	---								•		•
HDIRECTION?	HDIR?	---										•
LEVEL	LEV	<NUM>	•	•								
	LEV?	---	•	•								
LOW LEVEL SETTINGS	LLSET	<BIN BLK>				•	•					
	LLSET?	---				•	•					
MAXIMUM?	MAX?	---	•	•								
MINIMUM?	MIN?	---	•	•								
NEXT	NEXT	---						•			•	
OPEN	OPEN	<NUM>[, <NUM>...]						•			•	
		ALL						•			•	
	OPEN?	---						•			•	
OUTPUT	OUT	ON				•	•					
		OFF				•	•					
	OUT?	---				•	•					
OUTPUT (FLOATING SUPPLY)	FSOUT	ON					•					
		OFF					•					
	FSOUT?	---					•					
OUTPUT (LOGIC SUPPLY)	LSOUT	ON					•					
		OFF					•					
	LSOUT?	---					•					
PRESCALE	PRE	ON	•	•								
		OFF	•	•								
	PRE?	---	•	•								
SCAN	SCAN	<NUM>[,<NUM>...]						•			•	
	SCAN?	---						•			•	
SEND	SEND	---	•	•	•							
		<NUM>				•						
SLOPE	SLO	POS	•	•		•						•
		NEG	•	•		•						•
	SLO?	---	•	•		•						•
SOURCE	SOU	REAR	•		•							
		FRONT	•		•							
		INT	•									
		EXT	•									
	SOU ?	---	•		•							
TERMINATION	TERM	LO		•								
		HI		•								
	TERM?	---		•								

STATUS COMMANDS

COMMAND	ABBREVIATION	ARGUMENT	DC5009	DC5010	DM5010	FG5010	PS5010	SI5010	MI5010	50M30	50M40	50M70
LOCK?	LOCK?	---				•						
LRI	LRI	ON					•					
		OFF					•					
	LRI?	---					•					
NRI	NRI	ON					•					
		OFF					•					
	NRI?	---					•					
OPC	OPC	ON	•	•	•			•	•			
		OFF	•	•	•			•	•			
	OPC?	---	•	•	•			•	•			
OVERFLOW OR OVERRANGE	OVER	ON	•	•	•							
		OFF	•	•	•							
	OVER?	---	•	•	•							
PLI	PLI	ON				•						
		OFF				•						
	PLI?	---				•						
PRI	PRI	ON					•					
		OFF					•					
	PRI?	---					•					
REGULATION	REG?	---					•					
RQS	RQS	ON	•	•	•	•	•	•	•			
		OFF	•	•	•	•	•	•	•			
	RQS?	---	•	•	•	•	•	•	•			
USEREQUEST	USER	ON	•	•	•	•	•	•	•			
		OFF	•	•	•	•	•	•	•			
	USER?	---	•	•	•	•	•	•	•			

SYSTEM COMMANDS

COMMAND	ABBREVIATION	ARGUMENT	DC5009	DC5010	DM5010	FG5010	PS5010	SI5010	MI5010	50M30	50M40	50M70
DEVICE TRIGGER	DT	TRIG	•	•	•	•	•	•	•			
		OFF	•	•	•	•	•	•	•			
		GATE	•	•	•	•	•	•	•			
		SET	•	•	•	•	•	•	•			
	DT?	---	•	•	•	•	•	•	•			
ERROR?	ERR?	---	•	•	•	•	•	•	•			
IDENTIFY?	ID?	---	•	•	•	•	•	•	•			
INITIALIZE	INIT	---	•	•	•	•	•	•	•			
SETTINGS QUERY	SET?	---	•	•	•	•	•	•	•			
TEST	TEST	---	•	•	•	•	•	•	•			

GPIB STATUS AND ERROR REPORTING

STATUS BYTE (DECIMAL)	ERROR CODE (ERR?)	DESCRIPTION
Command Errors		
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
Execution Errors		
98	201	Command not executable in local
98	202	Settings lost due to rel
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
Internal Errors		
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 Error
*	340	System RAM error
*	341	System RAM error (low nibble)
*	342-349	Reserved for additional RAM errors
*	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
Internal Warning		
102	601	Overrange
102	602	Channel A 50 Ω Protect
102	603	Channel B 50 Ω Protect
102	604	No prescaler
102	605	Time of day clock not initialized and WAIT UNTIL command was to be executed
Device Status		
193	701	Below limits
195	703	Above limits
193	711	Channel A overflow
194	712	Channel B overflow
197	721	Neg. supply change to voltage reg.
198	722	Neg. supply change to current reg.
199	723	Neg. supply change to unregulated
201	724	Pos. supply change to voltage reg.
202	725	Pos. supply change to current reg.
203	726	Pos. supply change to unregulated
205	727	Log. supply change to voltage reg.
206	728	Log. supply change to current reg.
207	729	Log. supply change to unregulated
204	731	In lock
200	732	Not locked
225	74X**	Power on errors on card in slot X —Cannot clear PIA Data Direction Registers {(50M70), (50M30)} —Cannot write "1's" to PIA (50M40)
225	741	Cannot access PIA with all "0's" (S15010)
225	75X**	Power on errors on card in slot X —Cannot properly operate PIA Control Register {(50M70), (50M30)} —Cannot write "0's" to PIA (50M40)
225	751	Cannot access PIA with all "1's" (S15010)
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) (S15010)
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 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 (128)	7 (64)	6 (32)	5 (16)	4 (8)	3 (4)	2 (2)	1 (1) LSB	Condition
0	1	0	X	0	0	0	1	Power-on (65-81)
0	X	0	X	X	X	X	X	System event (64-79, 80-95)
0	0	0	X	0	0	0	0	Ordinary operation (0)
0	X	1	X	0	0	0	1	Command error (97, 113)
0	X	1	X	0	0	1	0	Execution error (98, 114)
0	X	1	X	0	0	1	1	Internal error (99, 115)
0	X	1	X	0	1	1	0	Internal error warning (102, 118)
1	X	1	X	X	X	X	X	Device status (192-207, 208-223)

- Bits 4–1: Four-bit status code
- Bit 5: Microprocessor busy condition ("1" if busy)
- Bit 6: Abnormal/normal condition ("1" if abnormal)
- Bit 7: SRQ ("1" if requesting service)
- Bit 8: If 0, STB indicates event class; If 1, STB indicates device class