Unraveling the Mystery on the GPIB

IEEE 488 Bus — Going Your Way?

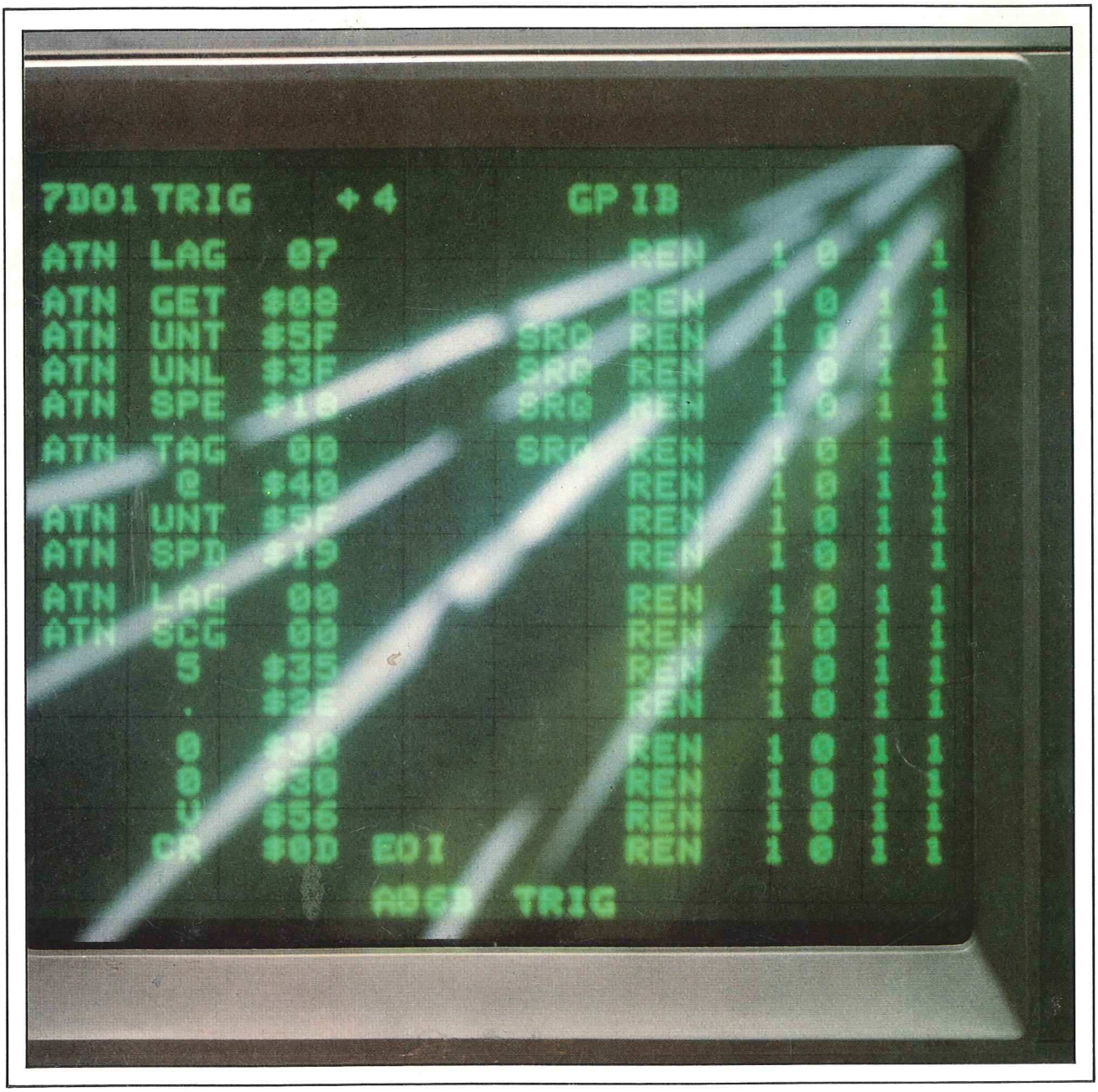
Pulse Generator or Function Generator? Making an Intelligent Choice

New Products

Volume 10
Number 2

1978

Tekscope



Tektronix®
COMMITTED TO EXCELLENCE

CONTENTS

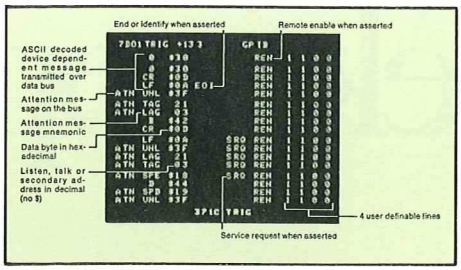
Tekscope

Customer information from
Tektronix, Inc.
Beaverton, Oregon 97077

Editor: Gordon Allison

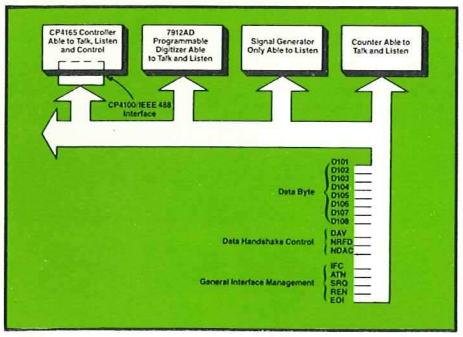
Unraveling the Mystery on the GPIB

The new DF2 Display Formatter, a companion plug-in for the 7D01 Logic Analyzer, lets you take a close-up view of action on the GPIB, displayed just the way it was programmed, in disassembled IEEE 488 mnemonic instructions.



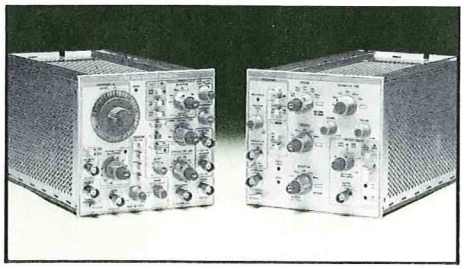
IEEE 488 Bus — Going Your Way?

Shedding a little more light on the IEEE Standard 488-1975, and discussing some pluses and minuses for the system designer.



Pulse Generator or Function Generator — Making an Intelligent Choice

Function generators with pulse capabilities suitable for logic and other applications are supplanting pulse generators normally purchased for these functions. A careful look at their respective capabilities will help you make the best choice for your needs.

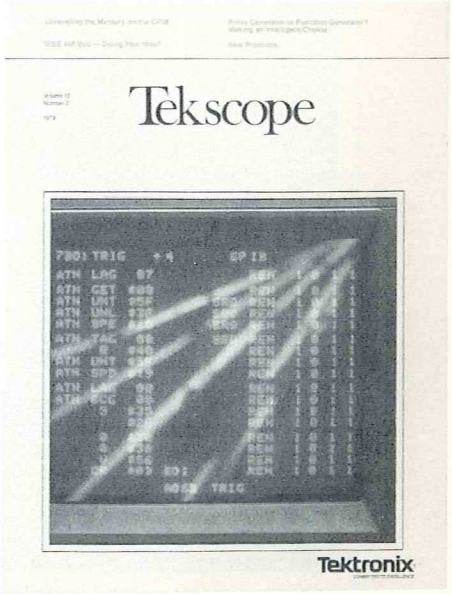


Tekscope is a bimonthly publication of Tektronix, Inc. In it you will find articles covering the entire scope of Tektronix' products. Technical articles discuss what's new in circuit and component design, measurement capability, and measurement technique. A new products section gives a brief description of products recently introduced and provides an opportunity to request further information.

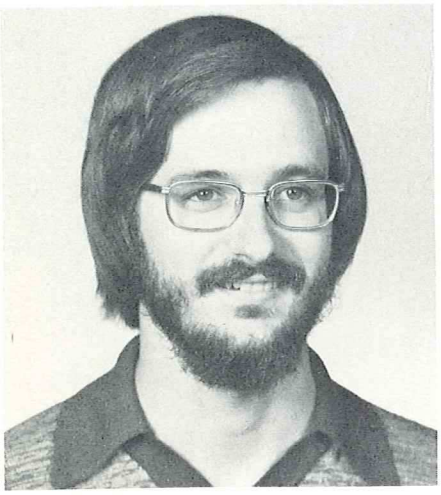
To better serve customers who maintain their TEKTRONIX instruments, the service information formerly appearing in Tekscope will be expanded and published in a publication dedicated to the service function.

Cover

The DF2 Display Formatter illuminates the activity taking place on the GPIB by disassembling the instructions and displaying them in familiar IEEE 488 message mnemonics.



Unraveling the Mystery on the GPIB



Bruce Ableidinger joined Tek 4 years ago upon completion of his BSEE at Washington State in '74. After six months in Component Evaluation he joined the TM 500 engineering group, working on digital products. His latest assignment has been Project Engineer on the DF2 Display Formatter for the Logic Development Products group.

When people try to communicate it helps if they all speak the same language. Even so, there are misunderstandings.

We now have a standard system for connecting instruments together so they can communicate. It's called the General Purpose Interface Bus (GPIB). The formal term is IEEE Standard 488-1975, "Digital Interface for Programmable Instrumentation." This standard specifies the mechanical, electrical, and functional elements of the digital interface system. One would suppose that with such a standard, a group of instruments could be connected together as a system and function perfectly. Not necessarily so, any more than people who speak the same language communicate perfectly.

When a group of instruments connected together fails to function properly, it is often difficult to determine the cause. Is it a defective instrument, faulty connection, timing problem, or programming error? How can one determine which?

One of the most useful instruments in working with digital systems is the logic analyzer. Early analyzers usually presented digital information in one of two formats — logic timing or logic state. The application dictated the type selected. Circuit designers preferred timing diagrams, while software designers chose state table displays. A third mode, mapping, appeared later on some logic state analyzers, providing the system designer a convenient means of monitoring program flow.

With the introduction of the Tektronix 7D01 Logic Analyzer and DF1 Display Formatter[1], the user was given a choice of all three display modes in a single instrument.

Now, the increasing demand for programmable instrumentation using the GPIB has created a need to view data in still another format — IEEE 488 mnemonics. These mnemonics represent, or are the equivalent of, particular logic patterns occurring on the 8 data lines and 4 status lines of the GPIB, and are defined in the standard itself. The new DF2 Display Formatter with its companion 7D01 Logic Analyzer, converts the binary values appearing on the data and status lines into their equivalent mnemonics and displays the results on a crt.

How it works

Front panel pushbuttons on the DF2 give the user a choice of seven display modes: timing, map, binary, octal, hexadecimal, GPIB, and ASCII. In the GPIB and ASCII display modes, the user is given the further choice of displaying data in binary, octal, or hexadecimal.

In the GPIB mode, data is acquired from the GPIB by the 7D01

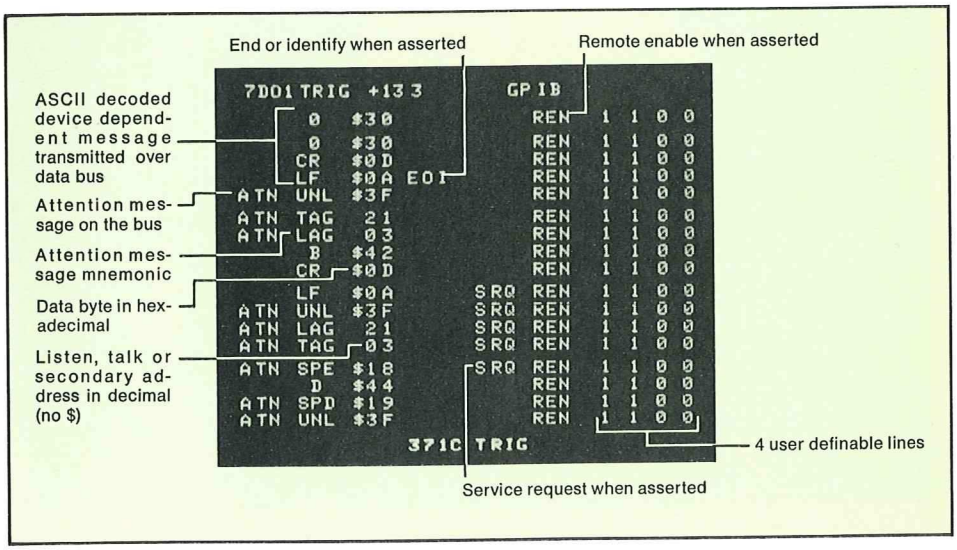


Fig. 1. GPIB display of activity on the data bus, transfer bus (handshake lines), and management bus (control lines). Disassembled instructions are displayed on the crt in IEEE 488 message mnemonics familiar to the GPIB user.

Logic Analyzer through an adapter that provides a convenient transition from the GPIB connector to the two P6451 Data Acquisition Probes used with the 7D01.
Note that specific assignments are given each probe input with the exception of channels 0-3 and the probe qualifier input.

Channels 0-3 and the qualifier input are user definable. They can be used to view signals from the user's circuitry or applied to the handshake lines available on the GPIB Adapter.

Data acquired by the 7D01 is transferred to the DF2 memory and processed for display under microprocessor control. A program ROM in the DF2 provides permanent storage for the microprocessor instructions.

Figure 1 is a GPIB display. Note that it is so labeled at the top of the right hand column. The GPIB transactions are displayed in hexadecimal, as indicated by a dollar sign ($) preceding the alphanumeric hex value. You will note a couple of exceptions — the listen, talk, and secondary address groups are displayed in decimal, and the four usable-definable lines are presented in binary. The number of clock cycles occurring between the trigger and the first line of the display (at top of screen) is also displayed in decimal, while the trigger word (at bottom of screen) is displayed in the previously selected format — in this instance, hexadecimal. This intermingling of formats on a single display lets you view each piece of data in its most useful form.

Seventeen GPIB events, plus the cursor position and trigger word, can be displayed on-screen at one time. As the data is "scrolled" manually using the 7D01 cursor control, all 254 events acquired by the 7D01 can be viewed. Each of the 17 lines (bus transactions) contains data information from the interface bus, displayed in IEEE 488 mnemonic message format.

Versatile triggering and storage

One of the keys to unraveling the mystery on the GPIB is the ability to capture the action occurring on the bus at any selected point in time, and examine events leading up to and following that action. The 7D01/DF2 provides that ability.

The user has a choice of synchronous or asynchronous triggering and synchronous or asynchronous data storage. This makes possible four combinations of triggering and storage to create a very versatile system. In synchronous operation, triggering and storage are related to an external clock, while asynchronous operation utilizes a very accurate internal clock. Let's consider where each of these triggering/storing modes might be used in a GPIB interface environment.

Synchronously triggering on an input value set by the word recognizer switches, and synchronously storing data into the logic analyzer clocked by the negative edge of data valid (DAV) is the most common mode for looking at data and control transactions across the GPIB. For example, one could look at device-dependent messages being transmitted to the controller from one particular device, by setting the word recognizer to trigger on the talk address of that device.

Triggering asynchronously on the input word equal to the word recog-

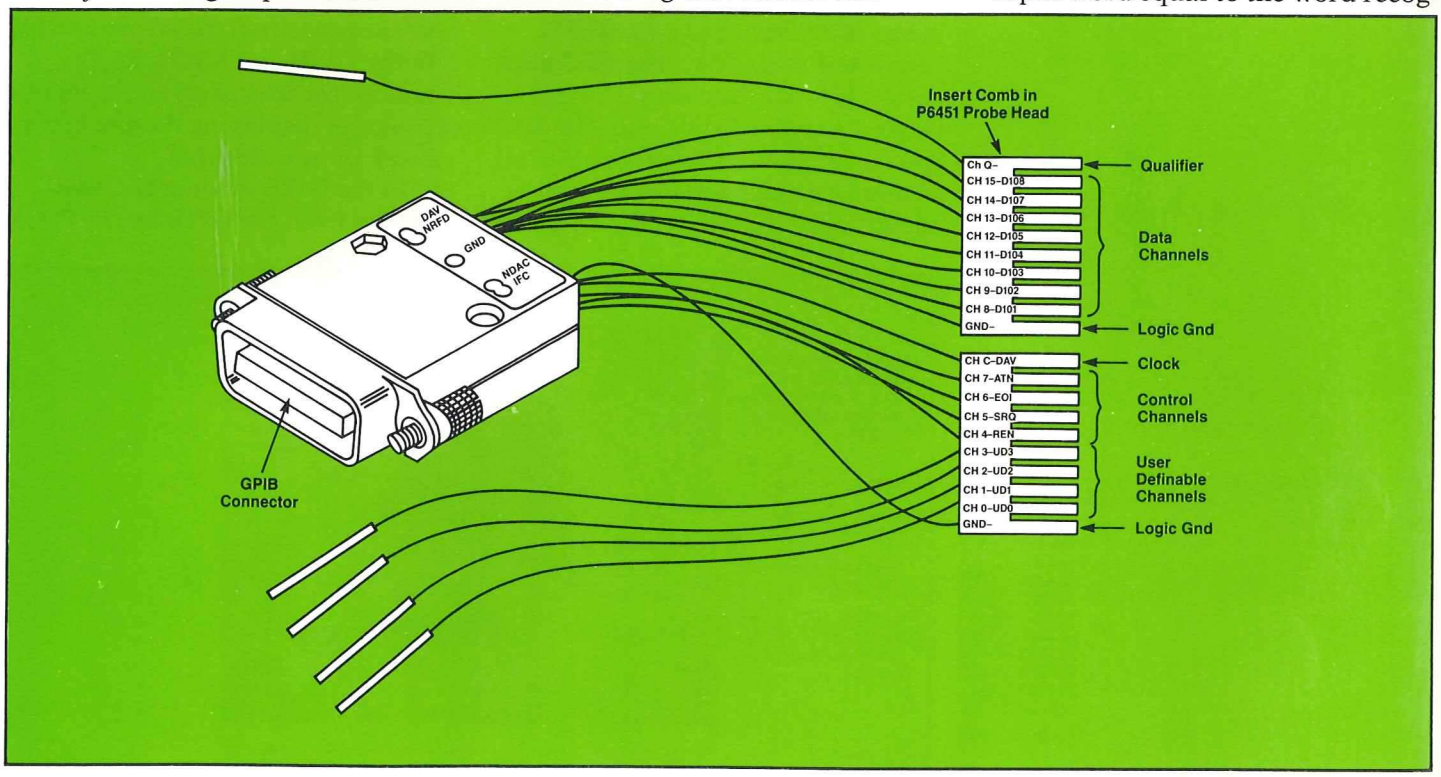


Fig. 2 The GPIB Probe Adapter provides a convenient means of interfacing the GPIB connector to the P6451 Probes used with the 7D01 Logic Analyzer. Note that four of the signal inputs and an external qualifier input are user-definable.

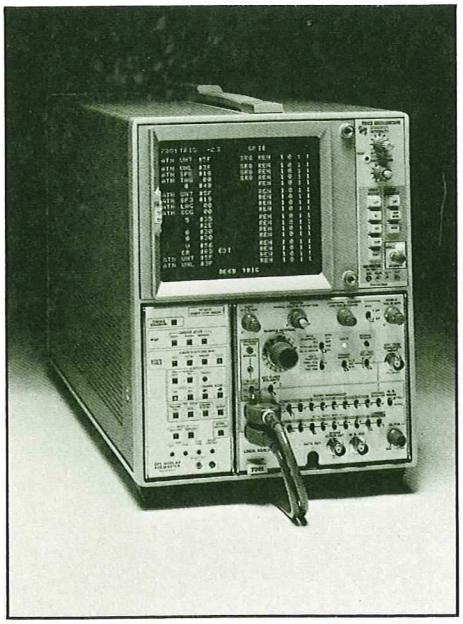


Fig. 3. The 7D01 Logic Analyzer with the DF2 Display Formatter converts any 7000-Series Mainframe to a Logic Analyzer with seven ways to look at logic.

nizer switch settings, and storing GPIB transactions synchronously clocked by DAV, is useful when the user wants to trigger on a single combinatorial event in the circuit and look at the GPIB transactions that led up to that event. The user-definable lines can be connected to the circuit, and the 7D01 can be triggered asynchronously relative to the data being stored. A typical example is triggering on the interface clear line (IFC) which occurs asynchronously to the handshake, and looking at the GPIB transactions before and after the IFC occurrence to discover why it was asserted and the effects it had on the system operation.

Triggering synchronously on a GPIB transaction and storing data asynchronously is generally used when timing information before, during, and/or after a GPIB transaction is desired. The most common example is studying the handshake cycle around a particular attention (ATN) message for correctness and speed. The user can also look at the effect a particular ATN message has on the circuit handling the interface.

Triggering the logic anaylzer on an asynchronous event and recording the timing information of the input lines is the normal operating mode of a timing logic analyzer. It is useful for studying the GPIB handshake and speed performance of the system. Typical examples would be examining the asynchronous parallel poll as the controller asserts ATN and End or Identify (EOI) to command those devices, with the capability, to put their status on the data lines; and looking at Service Request (SRQ), IFC, and Remote Enable (REN) for their time of assertion and relationship to other GPIB lines.

Design applications

The design, debug, and evaluation stages of a product containing a GPIB are all ideal application areas for the 7D01/DF2. Whether the design is accomplished using hardware techniques or with a microprocessor/peripheral implementation, the DF2 can save time in finding errors, validating the design, and measuring the performance of the interface.

Probably the most common design for the GPIB involves a subset of the total interface capability. The beauty of the IEEE 488 bus is that only a portion of its full capability need be implemented, depending on what the designer wants the interface to do. This allows the use of the GPIB while minimizing costs and design time.

For example, a designer may want to interface an existing line printer to the GPIB. This task involves implementing the Basic Listener function (and optionally Listen Only). The Basic Listener interface for the printer requires the receipt of the My Listen Address (MLA) attention command, before it prints device-dependent messages (those messages sent with attention false). Listen Only forces the printer to always listen and hence prints all device-dependent messages across the bus. These types of designs are best implemented in TTL hardware because of their simplicity.

After the design is finished and built, the DF2 can quickly check that the handshake cycle functions properly when attention is true and when it is false; that the interface handles My Listen Address correctly (if the Listen Only Mode switch is not set); and that the device-dependent messages are being printed correctly and no characters are lost or translated.

Microprocessor-based interface design

When designing a microprocessor-based GPIB interface, the DF2 can aid in the program development, debug, and performance measurement phase. A microprocessor-based GPIB interface presents more difficult design and evaluation problems than does a hardware-designed interface. Usually a more sophisticated portion of the total GPIB capability is to be implemented. For example, functions such as listening, talking, serial poll, remote-local, and even parallel poll may be included.

The IEEE 488 document needs to be understood very well in order to know all of the interdependencies and interactions of the state diagrams. To glean from the IEEE 488 standard the methods of implementing this sophisticated set of functions is far from a trivial task. Also, the firmware must implement these functions with the least amount of interaction. It is easy for subtle errors to crop up in such a complex system.

A microprocessor is inherently a sequential device which means it can handle only one task at a time; tasks such as reading from a port, analyzing the data, making a decision, and outputting an "answer" to another port. A strict hardware design can operate in a parallel mode, executing more than one function at a time.

While the IEEE 488 state diagrams show various events or messages occurring simultaneously when moving from one state to the next, the microprocessor must handle these messages sequentially. The designer must learn the legal order of occurrences and the maximum

amount of delay permitted between them.

Finally, implementing the remote-local function and/or the interface clear function requires using interrupts. This adds complexity to the entire code because the interrupts occur asynchronously. The software designer must write programs with the knowledge that the microprocessor can be interrupted any time the interrupts are not masked. The designer must also consider how fast the interrupt has to be handled. The interrupts cannot be masked for too long because either Remote Enable (REN) going false or Interface Clear (IFC) going true must be recognized and handled within the GPIB specification of 100 microseconds.

The DF2 is useful in monitoring all of these design and debug tasks. For example, it can be used to measure the time the microprocessor takes to handle an interrupt, look for and illuminate implementation errors, and measure the speed performance of the bus to determine whether or not the design goals were met.

Controller functions analyzed

One of the functions of the GPIB standard is that of controller. Most controllers on the market today are calculator-based, or are an extension of a minicomputer. These devices control the bus through execution of program steps previously loaded into memory.

With another device on the bus, or perhaps a talker-listener simulator built up as a tool to handshake with the controller full speed, the controller can be checked out for correct "macro" instruction implementation. One program step in the controller memory can be translated into many GPIB attention commands to perform a specific function. For example, the TEKTRONIX 4051 GPIB controller has a poll statement in BASIC which handles service requests. The controller must Serial Poll Enable all devices on the bus (SPE is a universal attention command), then talk address each device sequentially to find out if that device was asserting Service Request (SRQ). The DF2 quickly and easily shows controller management of the SRQ function. It displays the sequence of attention messages that set up the serial poll, the status byte put out by the device that was talk addressed, and the sequence of attention messages to handle further device polling and serial poll disabling.

Other controller measurements the DF2 can perform include monitoring the asynchronous parallel poll for correct implementation and speed; verifying that one controller passes control to another correctly; and measuring the handshake speed of the controller for both attention messages and device-dependent messages into and out of the controller.

Summary

The adoption of IEEE Standard 488-1975 is a big step toward the goal of effective integration of a group of instruments and peripherals to form an automated instrumentation system.

The number of companies manufacturing GPIB-compatible instruments is growing rapidly, and it is becoming easier for the end user to find the particular devices needed for his or her application. If a system is being put together, it is very likely that more than one company's products will be represented in that system. Integrating these multiple-vendor purchases can be very difficult and frustrating because of the non-standard data transmission formats used and because of the less-than-adequate documentation supplied by some manufacturers. The 7D01 Logic Analyzer and DF2 Display Formatter can transform this normally time consuming and frustrating chore into a very short system-integration task.

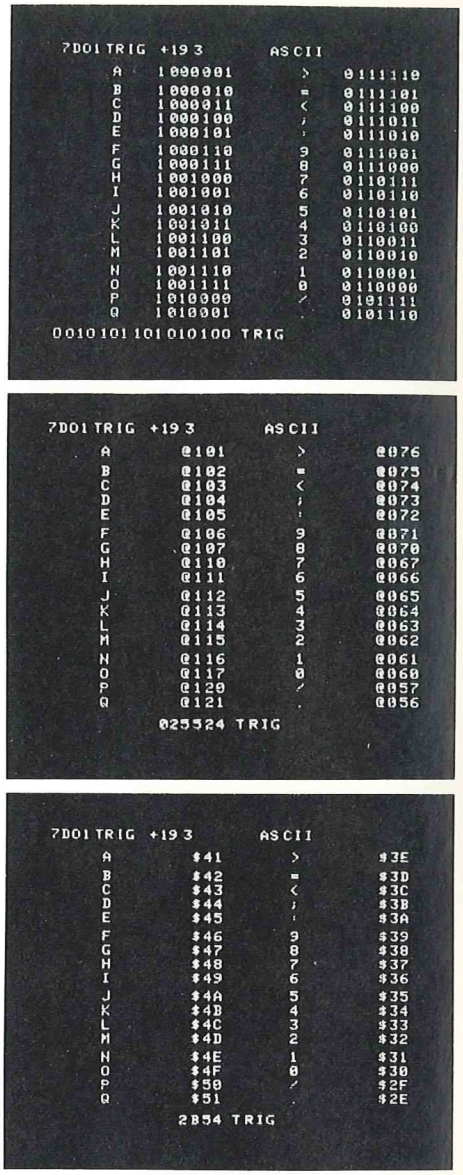
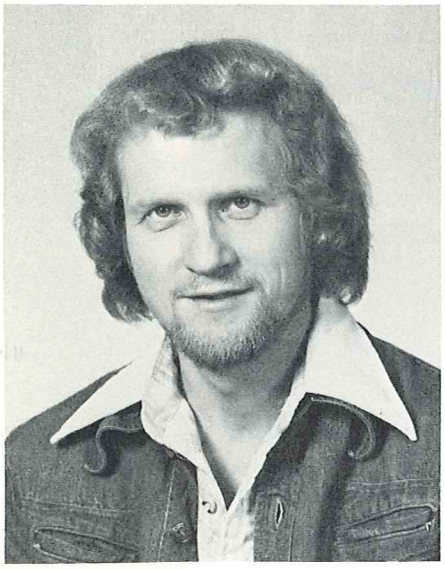


Fig. 4. In the ASCII mode, data may be displayed in binary (top), octal (center), or hexadecimal (bottom).

1. A Display Formatter — the Indispensable Tool for the Data Domain — Tekscope Volume 8 Number 4 1976.

The IEEE 488 Bus — Going Your Way?



Jim Kimball has worked at Tek for four years as an electronics technician and engineering writer on digital instruments and systems. His interests in electronics include high-speed analog-to-digital conversion and data interfacing. He received an A.S. degree from Lane Community College and is presently working toward a BSEE degree at the University of Portland.

Many designers and electronic instrument manufacturers are getting on the IEEE 488 bus and many more are about to board. But as with any bus system, there are a few unhappy riders.

IEEE 488 critics are asking questions such as, "Sure, I won't get smoke if the instruments I connect are IEEE 488-compatible, but will they work?" And, "Even if I carefully select instruments for the capabilities I need, will they speak the same language?" And, "How can I be sure the instruments I choose for my system all implement the optional features in the standard that I need?"

A lot of confusion comes from too little understanding of the standard. So let's find out what the standard is and what it is not.

A closer look

The bus concept specified in IEEE Standard 488-1975* is partial to small automatic test systems. The IEEE 488 bus provides more flexiblity for such systems than other interface standards such as CAMAC (IEEE Standard 583-1975).

This flexibility is no accident. A primary IEEE 488 objective is to specify only those interface functions necessary for clear and orderly communication. Device-dependent functions — what the instruments do and how they do it — are specifically left up to the instrument designer.

The standard defines an interface system: first, the bus that connects the instruments, including the connectors, the signal lines, and the voltage levels on the lines; second, the functions used by the instrument interfaces to exchange data; and third, control messages and protocols.

Devices on the bus perform one of three roles: talker, listener, or controller. As in any good repertory theater company, each device may take more than one role as the occasion demands. As in a stage production, the parts are assigned by the director, called the controller-in-charge by IEEE 488. This role can be passed around, just as players in a repertory company may direct one production, but act in another. At any time, however, there can be only one director. Just as actors do not (or should not) step on each other's lines, only one device can talk at a time, though more than one can listen. The producer, called the system controller, retains ultimate authority and can double as director — that is, be both system controller and controller-in-charge.

To stage an IEEE 488 production, we must select the skills needed by the actors. These are the interface functions defined by the standard;
Source Handshake
Acceptor Handshake
Talker
Listener
Service Request
Remote Local
Parallel Poll
Device Clear
Device Trigger
Controller

The designer can choose from a list of subsets of these functions to put together the most cost-effective combination. A collection of subsets sounds strange until you understand that it's just a shorthand way of noting the device's interface functions. C0, for instance, states that a device has zero capability as a controller. DT1 means a device can be triggered to perform a designer-chosen function when it receives the device trigger interface message.

To see how these interface functions are used on the bus, let's take a look at the diagram of the bus signal lines shown in Figure 1. The 16 lines comprise three groups: data, handshake, and interface management.

The eight data lines are used together to transfer eight-bit bytes, hence the term "bit-parallel, byte-serial interface."

The handshake lines control an asynchronous, three wire handshake. DAV (Data Valid) is asserted by the transmitting device, and NRFD (Not Ready For Data) and NDAC (Not Data Accepted) are asserted by the receiving device to pace the dialogue on the bus.

The interface management lines have several uses. ATN (Attention) is like the director's megaphone: when the controller-in-charge asserts ATN, everyone pays attention. Two lines are reserved for the system controller: REN (Remote Enable) is asserted for remote control of devices on the bus, and IFC (Interface Clear) tells all devices to reset their interface functions. SRQ (Service Request) can be asserted by any device to interrupt the controller. EOI (End Or Identify) indicates the end of a data transfer, but can also be used with ATN in a special polling mode.

A byte at a time

How are these lines used by the interface functions? Let's take the source and acceptor handshakes first. Actually, they are two parts of

the same handshake. Figure 2 shows the states of these lines as they are set by a talker using the source handshake and a listener using the acceptor handshake. Note that the timing diagram relates the electrical signals on the bus to the states of the source and acceptor handshakes. By looking at both, it may be easier to grasp the sequence of the interlocked handshakes than it is to absorb the infamous state diagrams in the standard.

1) To begin, the source (talker) goes to the Source Generate State (SGNS). In this state, the source is not asserting a data byte on the data lines or DAV. When no bus driver is asserting a line, it rises to the high level set by the bus terminating network. The acceptors (listeners) are in the Acceptor Not Ready State (ANRS), asserting both NRFD and NDAC. In this condition, NRFD and NDAC are low.

2) The source sets the data byte on the data lines and enters the Source Delay State (SDYS). If this is the last byte in the message, the source can assert EOI, as well. The source waits for the data to settle on the lines and for all acceptors to reach the Acceptor Ready State (ACRS).

3) Each acceptor says, "I'm ready" by releasing NRFD to move to ACRS. This is one of the points in the handshake designed to accommodate slow listeners. The NRFD line can be thought of as a wired-OR input to the source handshake logic. Any acceptor can delay the source handshake by asserting this line.

4) When the source sees NRFD high, it enters the Source Transfer State (STRS) by validating the data with DAV. The source then waits for the data to be accepted.

5) When the receiving devices see DAV asserted, they go to the Accept Data State (ACDS). Each device asserts NRFD because it is busy with the current data byte and is not ready for another.

6) As each device accepts the data, it releases NDAC to move from the ACDS to the Acceptor Wait for New cycle State (AWNS). Again, all receiving devices must release the NDAC line for the source to see a high level. When the source sees NDAC high (all have accepted the data), it enters the Source Wait for New cycle State (SWNS).

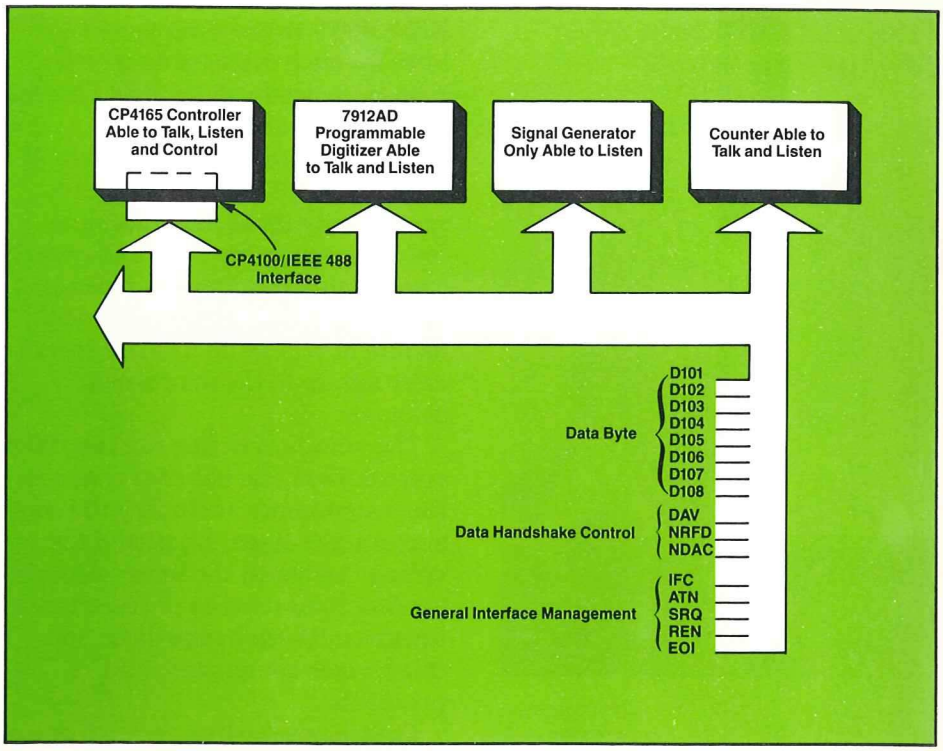


Fig. 1. IEEE 488 bus organization showing the bus signal lines and some typical bus instruments.

7) In the SWNS, the source can release DAV. This causes the acceptors to proceed to the ANRS, their intial state in the handshake. In ANRS, they assert NDAC.

8) The source continues to the SGNS, its intitial state in the handshake. In this state, it can change the data lines to prepare a new byte for transmission.

This is a typical sequence. The exact sequence is defined by the state diagrams in the standard. The only requirement is that if what happens on the signal lines differs from the above sequence, it must still conform to the state diagrams.

Who's in charge here?

We've just staged a dialogue between a talker and listener(s) using the source and acceptor handshakes, but they're not the only ones allowed to use the handshakes. The source handshake is also used by the controller-in-charge to send system control messages; these are called interface messages to distinguish them from device-dependent messages sent from talkers to listeners (see Fig. 3). Remember the director's megaphone — ATN? The controller asserts ATN to get the attention of all devices on the bus and then uses the source handshake to send interface messages on the data lines.

The interface messages that constitute the controller's vocabulary are defined by the standard. They can be thought of as ASCII codes given a new meaning when sent by the controller with ATN asserted.

Three groups of interface messages are reserved for the listen, talk, and secondary addresses. For instance, when a device sees its talk address (called My Talk Address) and ATN simultaneously, it must become a talker. When the controller removes ATN, the device begins the source handshake to transmit its data. Similarly, My Listen Address and ATN tell a device to listen to the data sent by the talker. Secondary addresses provide unique addresses for devices or functions that share a

single listen or talk address. In the TEKTRONIX 7912AD Programmable Digitizer, for example, secondary addresses select among the mainframe and plug-ins, all of which share the same IEEE 488 bus interface.

The controller uses other kinds of interface messages for other tasks. One is the Serial Poll Enable command (SPE) used by the service request interface function. Suppose an instrument is designed to assert SRQ when it has acquired some data. The controller must poll the devices to find the interrupting device since any one (or more than one) can assert SRQ. To conduct the poll, the controller sends SPE, a universal command, and then addresses each device in turn and reads a status byte from each. If the device asserted SRQ, it can code the status byte to tell the controller why.

Parallel Poll Configure (PPC) is an example of an addressed command. It prepares addressed devices to indicate who is requesting service. When ready, the devices respond together; so a parallel poll is quicker than a serial poll, though more complicated.

Device trigger is another function that uses an addressed command: Group Execute Trigger (GET). It's up to the instrument designer to decide whether to use this function and for what purpose. This function is provided in the 7912AD to synchronize, when desired, a digitize operation with the GET command.

The controller issues the Device Clear message (DCL) to initialize internal functions of devices on the bus. A universal command, DCL, applies to all devices. Its effect on each instrument, however, is decided by the designer who can choose to initialize any device function to any state that suits the purpose of the instrument.

So far, in addition to reviewing some of the interface messages, we've used all the control lines except two, REN and IFC. These lines are reserved for the system controller. Just as the producer holds the ultimate authority in the theater, the system controller holds the ultimate authority on the bus with these two lines. Setting REN raises the curtain; that is, it enables remote operation of the devices on the bus. Releasing REN brings down the curtain; when REN goes false, all devices must return to local mode. IFC is used by the system controller to call a halt to action on the bus; talkers and listeners are reset to receive commands from the controller-in-charge, a role that automatically reverts to the system controller following an interface clear.

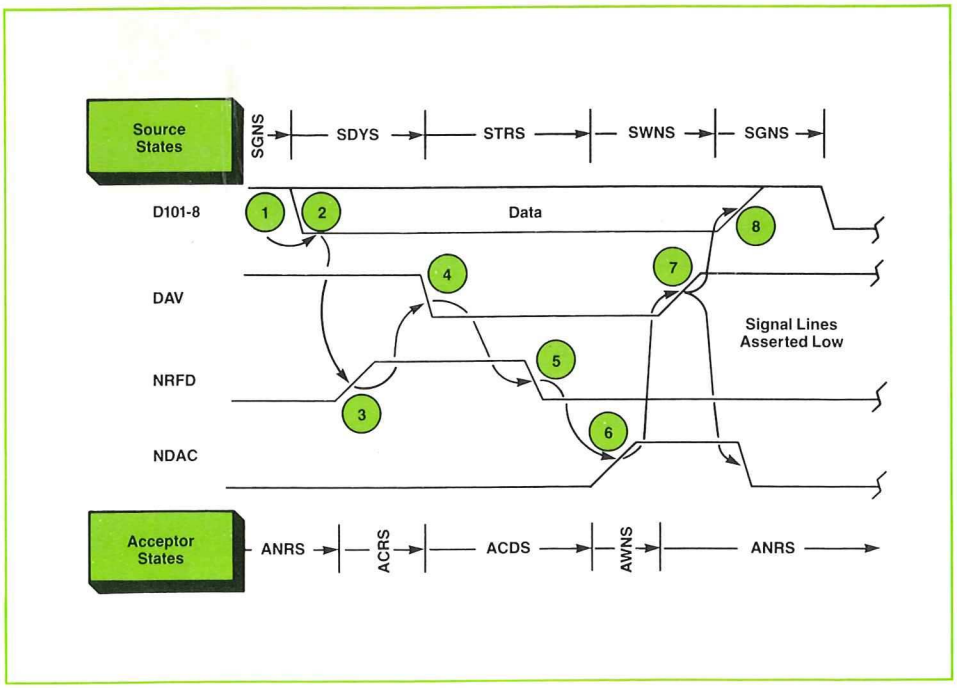


Fig. 2. Handshake sequence to move one byte from a talker to a listener. The numbers refer to steps described in text.

Adding it up

With this introduction to the IEEE 488 concept, let's add up the pluses and minuses for the system designer.

IEEE 488 is a parallel system. PLUS: Flexible configuration — it works, either in a star or linear configuration (Fig. 4). MINUS: Distance is limited in either configuration to 20 meters (about 65 feet). To maintain the bus electrical characteristics, a device load must be connected for each two meters of cable (no more than 15 devices connected). There's a fine point of etiquette here. Although you might expect devices to be spaced no more than two meters apart, they can be separated more if the required number of device loads are lumped at any point.

A variety of instruments can work together. PLUS: Both simple and complex instruments can be connected; they need only conform to the parts of the IEEE 488 standard they implement. You pay only for what you need. MINUS: Not all instruments include all functions. For example, not all instruments have the parallel poll function. The device trigger is another function that may be missing. At the end of a message, TEK instruments assert EOI when they finish talking; some others do not. They may send a special character or just quit talking, leaving it up to the controller to decide how long to wait before jumping in to take control.

Data freedom (or anarchy). PLUS: Designers are free to choose appropriate device-dependent messages. Even if instruments speak a different language, the standard allows them to be connected. MINUS: At least one of the instruments in a Talker/Listener pair must be smart enough to translate if the instru-

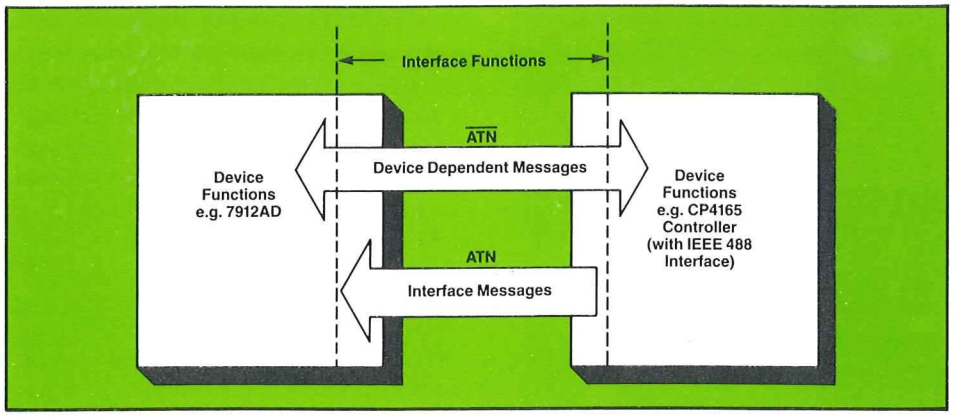


Fig. 3. The ATN line is used to distinguish between interface (system control) and device-dependent messages.

ments use different commands or formats. One voltmeter may switch to the 100 mV range when it receives a character string "VOL 100 E-3"; another may require, instead, a single ASCII character. The first instrument may send data in floating point decimal; the other in binary.

Asynchronous data transfer. PLUS: The handshake allows both slow and fast talkers to play together on the bus. MINUS: Everyone goes at the pace of the slowest device. As long as the race belongs not to the swift, but to the steady, that's all right. But that may mean a data rate of only 20 kilobytes per second, it may mean 10 kilobytes per second, or even less. Attention to design is required to get the device interfaces to go faster; it can be done, but it only takes one slowpoke to hold all devices back.

Common data byte size. PLUS: Eight bits is a useful size for data. It fits data processing that operates on words that are eight bits in length (or multiples of eight bits), as is often the case. MINUS: Eight-bit bytes are unhandy if your data happens to be nine bits wide, or your controller is a 12-bit machine.

Standard implementation. PLUS: The IEEE 488 electrical characteristics rely on the most common logic family — TTL. MINUS: The system is limited to medium speeds; ECL could go faster but doesn't fit the spec.

Making it play

There isn't room here to fit all the praise and blame heaped on the IEEE 488 standard to date. If you're trying to put together a system, you want to know if you can make it play. How can you select the right instruments and get them to speak to each other?

In many cases, the best answer is to let someone else do it for you. Buy the system, or at least the main components, from a single manufacturer who has already done the homework. For instance Tektronix offers the CP4165 Controller with an IEEE 488 interface and TEK SPS BASIC software to run an IEEE 488 system. Another all-in-one controller with an IEEE 488 port is the TEKTRONIX 4051 Graphic Computing System. Either way, you're off to a good start because the software that remains to be written amounts to filling in the blanks with commands and parameters specific to the instruments in the system. TEK SPS BASIC commands for TEKTRONIX acquisition instruments, such as the 7912AD, make it even easier with high-level software that needs no fill-ins.

Another advantage of buying from one manufacturer is that the system includes pieces that were meant to play together. Some of the minuses mentioned above are removed and some of the pluses enhanced.

But perhaps you want to put the system together yourself for any number of reasons. If doing it yourself includes designing an interface or writing extensive software, there's no way around reading the standard. Just as it's expected that a federal judge understands the Constitution, it's expected that an IEEE 488 designer understands the standard. It's dry reading, but it is thorough and complete. Be assured you'll eventually find the fine points. For instance, one more than half the number of instruments connected to the bus must be powered-up to maintain the bus termination. Some users found this out on their own when they discovered intermittents on the SRQ line, but they could have read it first in the standard.

The standard is thorough, but as designers continue to work with it, more questions are going to be raised. New design tools such as the 7D01 Logic Analyzer and DF2 Display Formatter, discussed elsewhere in this issue, will help answer some of these. Others may suggest topics for future articles on the standard.

*ANSI Standard *MC 1.1-1975* and proposed as an IEC standard. The interface system is often called the General Purpose Interface Bus (GPIB).

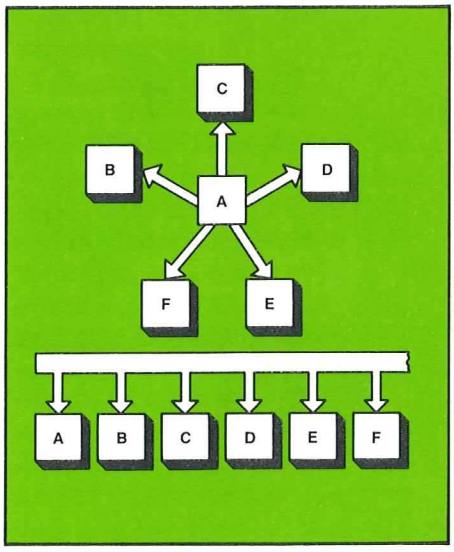
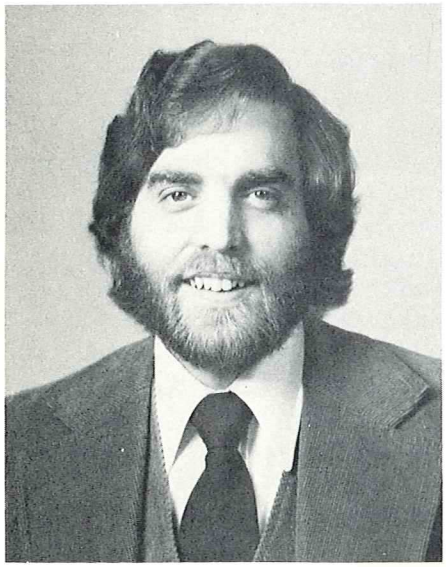


Fig. 4. An IEEE 488 system can be configured in either a star or a linear manner.

Pulse Generator or Function Generator? Making an Intelligent Choice



Bill Rasnake is well acquainted with customer needs, having served 4 years as a Tek field engineer. He continues to have frequent customer contact in his role in TM 500 Product Marketing. Bill has an Associate of Science degree from Victor Valley College in California, and served as an electronics instructor in the United States Air Force.

Versatility — the ability to do many things well — is basically a human characteristic. However, we often ascribe this virtue to instrumentation which can perform several functions or be used for many different jobs. When we consider buying a piece of test equipment and two instruments seem equally capable of doing our job, we usually choose the more versatile.

The 40-MHz TEKTRONIX FG 504 Function Generator and the 50-MHz PG 508 Pulse Generator often pose a dilemma for the potential pulse generator buyer. The comparison chart on page 14 gives a brief summary of the pulse generation characteristics of these two instruments. They are very similar in many respects and the difference in cost is small. The better choice might be obvious, but before deciding, let's consider some other factors:

What are the precise requirements for a pulse generator? Will these requirements change? Do we have other needs which the function generator would fulfill? What is the application for the instrument — production or engineering? Who will be using it — skilled or unskilled personnel?

The first question, of course, is the most important. While the pulse generation capabilities of the two instruments are similar, there are some important differences. We should examine these carefully to determine if both instruments can do our job, and what the tradeoffs may be.

Some important differences

Let's consider, first, how the two instruments generate pulses. The function generator produces a square wave which can be formed into a pulse by adjusting a front-panel symmetry control. The result is an output signal which maintains a relatively constant duty cycle with changing frequency. This is a very useful signal for checking the response of power amplifiers, pulse transformers, and other power sensitive devices without exceeding their dissipation limits.

The SYMMETRY control in the FG 504 varies the duty cycle of the square wave over a range of 7% to 93% for frequencies up to 4 MHz. Since the pulse width is a function of frequency, this imposes some operating constraints. For example, the narrower output pulses can be generated at an internally controlled repetition rate only at the higher frequencies. However, by triggering the FG 504 externally, pulse widths as narrow as 15 nanoseconds can be generated at repetition rates determined by the external trigger.

The pulse generator uses a different scheme for generating pulses. Separate circuits are used to determine period, or frequency, and pulse duration. This gives us the capability of generating constant width pulses independent of frequency — an important feature in working with logic and control circuits, and one usually preferred when working strictly with pulses.

The ability to complement the PG 508 output pulse makes it possible to achieve output duty cycles of 0%-100%, within 10 nanoseconds, the minimum pulse duration of the PG 508.

Figures 2 and 3 illustrate the change in pulse width and duty cycle with a change in frequency for both the function generator and pulse generator.

Delayed pulses

Another factor to be considered in choosing a pulse generator is pulse delay capability. Both the PG 508 and FG 504 have pulse delay capability but in quite a different sense. In the PG 508 you can select delays (65 ns to greater than 100 ms from an external signal) with a front-panel control, and operate the delay in any of three modes: pulse output delayed with respect to an external trigger input, pulse output delayed with respect to an internal trigger output, and a pulse output delayed with respect to an intitial pulse output. The latter is often referred to as double or paired pulse generation.

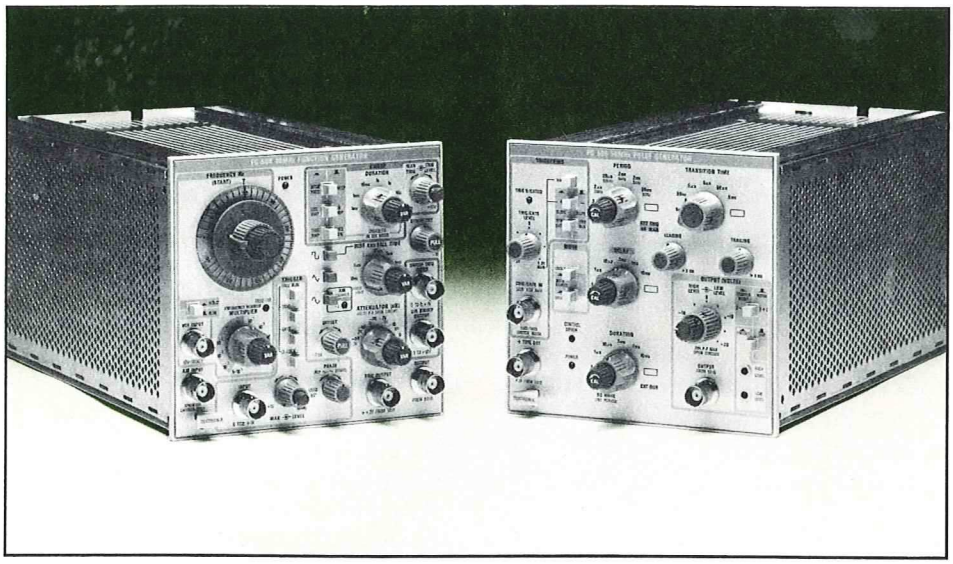


Fig. 1. Both the FG 504 Function Generator (left) and PG 508 Pulse Generator (right) offer a wide range of pulse capabilities. A thorough analysis of these capabilities may be necessary to detemine which instrument is best suited to your needs.

Paired pulses can be generated at selected repetition rates with the delay control determining the time between the two pulses.

Paired pulse operation is very convenient to use in evaluating a circuit's ability to differentiate between two closely spaced pulses.

In the FG 504, delay from an external signal can be achieved in several different ways. While not as convenient for pulse work as the delay capability in the PG 508, it has advantages for some applications. When the FG 504 main generator is operated in the triggered mode, the PHASE control can serve as a delay control covering a range equal to the pulse width being generated. The internal sweep generator also can serve as a source of delay. The sweep can be triggered externally and the sweep output connected via an external cable to the main generator trigger input. The LEVEL control then serves as the delay control. Delays from less than 10μs to greater than 100 s can be obtained at repetition rates to $\approx$ 5 kHz.

For many logic applications, the phase lock feature of the FG 504 also serves as a delay function. For example, the FG 504 can be triggered from, and locked to, the clock signal from a digital circuit. The PHASE control can then be used to control the start of the pulse output over a range of $\pm 90°$ with respect to the clock signal. This is a convenient means of generating a biphase clock.

Both the PG 508, with its delay capability, and the FG 504, with phase lock, are often used to generate muliphase clocks. The latter offers an advantage in operating convenience when used to generate the second, third, or fourth phase. The master clock frequency can be varied over a wide range and the phase-locked FG 504 will follow without major readjustment other than occasional range switching.

Pulse shaping capabilities

In many applications, the ability to adjust pulse rise and fall times is very important. For example, typical rise and fall times vary widely between logic families. If we are to accurately simulate logic circuit transactions, variable rise and fall times are a must.

Faster-than-specified drive signals can cause problems through capacitive coupling to adjacent circuitry. Conversely, slower-than-specified drive signals to a CMOS gate will cause its dissipation to increase greatly during each transition. Adjustable rise and fall times let us avoid these undesirable conditions.

Controllable rise and fall times are also useful in determining the characteristics of edge-triggered devices.

The fastest rise and fall times in the PG 508 and FG 504 are nearly the same — 5 ns and 6 ns, respectively. Both have adjustable rise and fall times, but with an important difference. In the PG 508, rise and fall time can be adjusted independently. In the FG 504, one control adjusts both at the same time. Each has advantages depending on the application. A single control is more convenient when simulating limited bandwidth. On the other hand, independent controls permit simulation of conditions where rise and fall times are unequal, such as in circuits having unequal slew rates or unequal

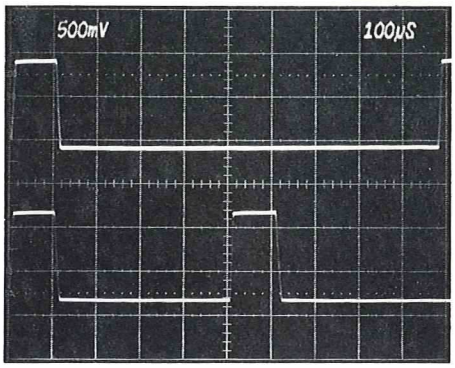


Fig. 2. In the PG 508 Pulse Generator, pulse width is independent of frequency. This gives the user flexibility and operating ease in setting up and making changes in either pulse width or repetition rate independently.

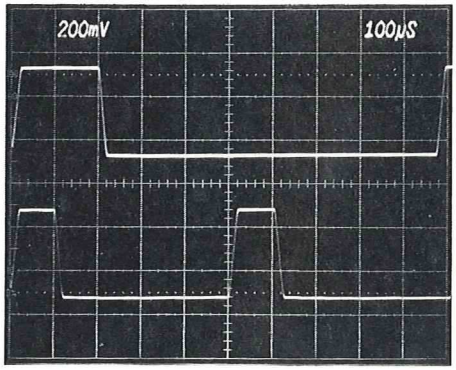


Fig. 3. In the FG 504 Function Generator, pulse width is a function of frequency as shown in this photo. This type of signal is ideal for applications requiring a constant duty cycle.

output impedances to rising and falling signals.

Pulse output voltage capability

One of the most important factors to be considered when purchasing a pulse generator is its output capability. It would appear from the comparison chart on page 14 that the FG 504 has a higher amplitude output capability than the PG 508. Let's take a closer look. Note the seeming contradiction between the peak-to-peak signal output capabilities and the unipolar output capabilities.

A unipolar pulse is one which does not cross ground. Both the PG 508 and FG 504 can generate unipolar pulses. But here again there are important differences in how they are achieved.

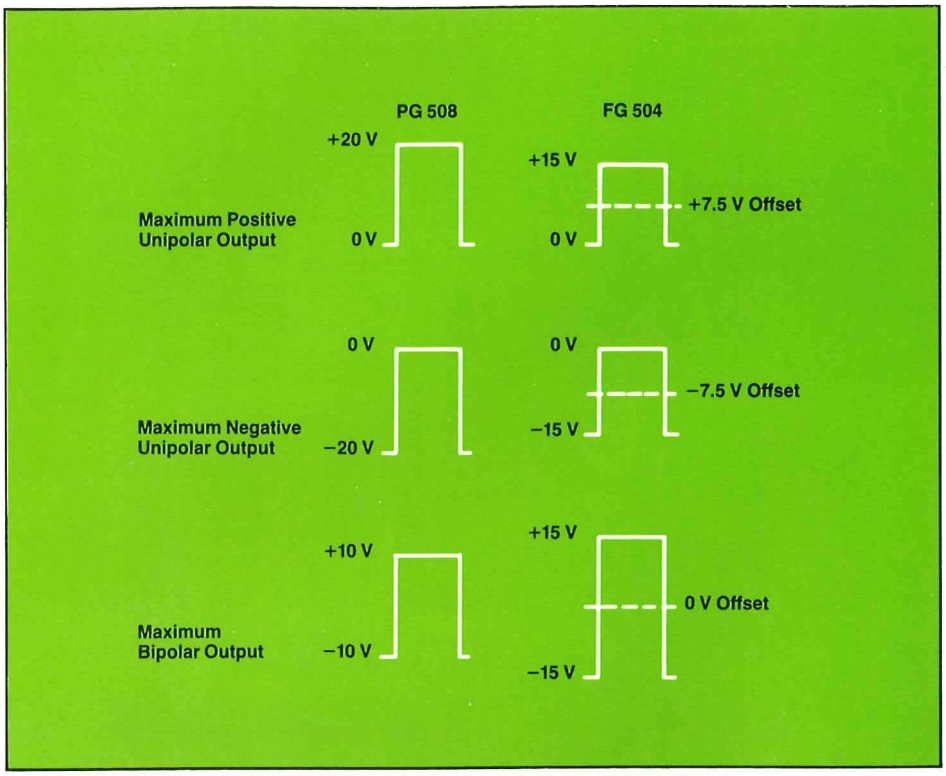


Fig. 4. Maximum unipolar and bipolar pulses obtainable with the PG 508 Pulse Generator and FG 504 Function Generator.

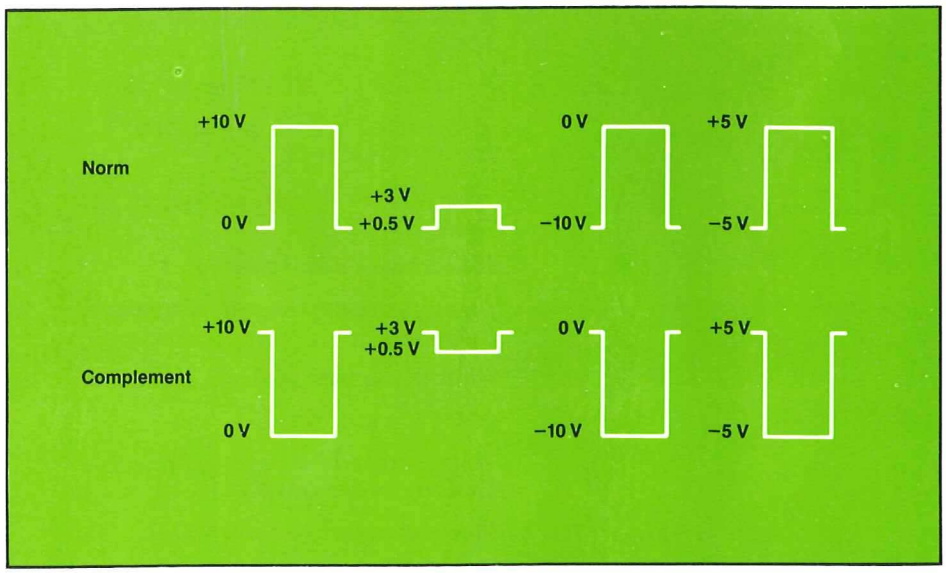


Fig. 5. Typical output pulse waveforms of the PG 508 Pulse Generator illustrating versatile output level control. Front-panel pushbutton complements the output pulse.

On the PG 508, independent HIGH LEVEL and LOW LEVEL controls allow you to precisely set the output pulse over a range of ±20 volts. Thus, it is a simple matter to set up either positive or negative unipolar pulses up to 20 volts in amplitude (see Figure 4). The ability to set the output levels precisely is a great convenience for many logic applications such as measuring noise immunity or rapidly setting the output to actual logic levels.

The output of the FG 504, on the other hand, normally swings ±15 volts around ground, and attenuator and offset controls are used to achieve the desired pulse amplitude and level. The OFFSET control range is ±7.5 volts which allows us to achieve a unipolar pulse 15 volts in amplitude.

The constant amplitude output with post-attenuator offset provides an ideal signal for determining comparator or logic thresholds, and also for determining optimum amplifier bias voltages.

The PG 508 has provision for presetting the output voltages, a time-saving convenience in many instances. For example, if you are working regularly with a particular logic family, the preset controls can be set for that logic family and be available simply by pressing a front-panel pushbutton. The output voltages can also be made to track an external voltage — a safety feature for some CMOS applications where the device can be damaged if the driving signal exceeds supply voltage.

Another convenience feature on the PG 508 is the ability to obtain the complement of the output pulse simply by pressing a front-panel pushbutton (see Figure 5). This is particularly useful in logic testing where a pulse going in the opposite direction is often needed.

Clean 50 Ω source.

An important specification to check when evaluating pulse output

A Comparison of the PG 508 Pulse Generator and the FG 504 Function Generator.

Similar Features	PG 508	FG 504
Output Frequency	50 MHz	40 MHz
Output Amplitude	1 V-20 V p-p from 50 Ω ±20 V Window	0.1 V-30 V p-p from 50 Ω ±20 V Window
Unipolar Output	20 V Max	15 V Max
Rise and Fall Time	5 ns	6 ns
Minimum Pulse Width	10 ns	15 ns
External Trigger/Gate	50 MHz	20 MHz
Output Source Impedance	Low Reactance 50 Ω	Low Reactance 50 Ω
Custom Timing Positions	Yes	Yes
Counted Burst Output (with DD 501 Digital Delay)	Yes	Yes

Unique Features

PG 508	FG 504
Constant pulse width with changes in frequency	Constant duty cycle with changes in frequency
Delay and double pulse	Phase lock to an external signal with phase control adjustment
High/Low level output voltage controls	Attenuator and offset controls
0% to 100% duty cycle (within 10 ns)	7% to 93% duty cycle (to 4 MHz)
Independent rise/fall time control with 100:1 range	Simultaneous rise/fall time control
External control of output voltage	Attenuator to 100 mV amplitude
Selectable 1 MΩ/50 Ω trigger/gate input impedance	10 kΩ trigger/gate/phaselock input impedance
±3 V, ± slope, trigger controls	−1 V to +10 V, + slope trigger control.
Complement pulse output	Haversine or $\sin^2$ pulses
3-state trigger light	Triangle/ramp output, Sinewave output
Control error light	Repetition rate to 0.001 Hz
Preset High/Low level output controls	AM/FM modulation
	Triggerable Lin/Log sweep with Linear Sweep output.
	Waveform hold below 400 Hz

amplitude is the load into which the output is specified. Both the PG 508 and FG 504 are designed to provide a clean 50 Ω source. Thus, it is not necessary to terminate the output cable into 50 ohms for many applications. This makes the full output voltage available to drive MOS, CMOS, and other high impedance devices that require larger amplitude signals.

Burst capability

Both the PG 508 and FG 504 can provide gated pulse bursts. This is a very useful feature for applications such as loading shift registers and counters, or simulating serial data.

For some operations, simply gating the generator on with an external pulse is adequate. For those applications where the number of bits (pulses) must be accurately controlled, the DD 501 Digital Delay unit may be used with the PG 508 or FG 504 to provide such a capability up to 100,000 pulses.

The number of pulses desired in the burst is dialed up on the DD 501, and the delayed trigger output from the DD 501 is used to gate the PG 508 or FG 504. A jumper inside the DD 501 converts the DELAYED TRIGGER OUT to a delay interval gate for this application.

Some unique capabilities

We have seen that both the PG 508 and FG 504 can perform similar pulse functions. There are, however, operating differences and functional differences that may make one or the other more suitable for your needs. Both have additional capabilities that you should consider before making a final decision.

The PG 508 triggering section has some features particularly well suited to logic applications. A 3-state trigger light tells you whether the input trigger signal is above or below the TRIGGER/GATE LEVEL control setting, and flashes when the PG 508 is being triggered. It, thus, can serve as a logic probe as well as being an operating convenience. The PG 508 can be triggered on either the plus or minus slope of the trigger signal. Trigger input impedances of 50 Ω and 1 MΩ are selectable by an internal switch.

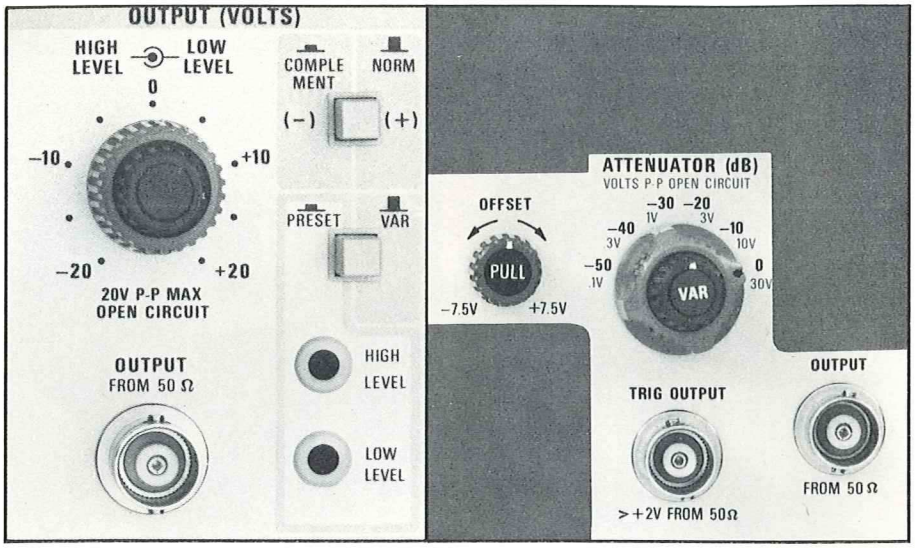


Fig. 6. A comparison of the output sections of the PG 508 Pulse Generator (left) and the FG 504 Function Generator (right) reveals some important differences in the output capabilities of the two instruments.

The FG 504 can only be triggered from a positive going signal, but the LEVEL control has an extended range of −1 V to +10 V, ideal for triggering from ramps. The VCF input on the FG 504 can be used to externally frequency modulate the generator providing pulse width modulation or jitter simulation capability.

There are two other features on the PG 508 which we should discuss. A CONTROL ERROR light indicates when the setting of any one of nine controls conflicts with the setting of another. If there is a timing conflict, for example, DURATION exceeds PERIOD, the light will flash. If there is a mode conflict, for example, SQ WAVE DURATION and DELAY pulse mode selected simultaneously, the light will come on steady.

The DURATION control on the PG 508 has an EXT DUR position. In this mode, the PG 508 serves as a pulse amplifier. The output signal will be of the same duration as the trigger input signal, with the output amplitude and levels determined by the output controls. This is a convenient mode to use in converting from one logic family to another. The rise and fall time controls remain operable in this mode allowing you to simulate precisely, or change, the rise and fall times of the incoming signal.

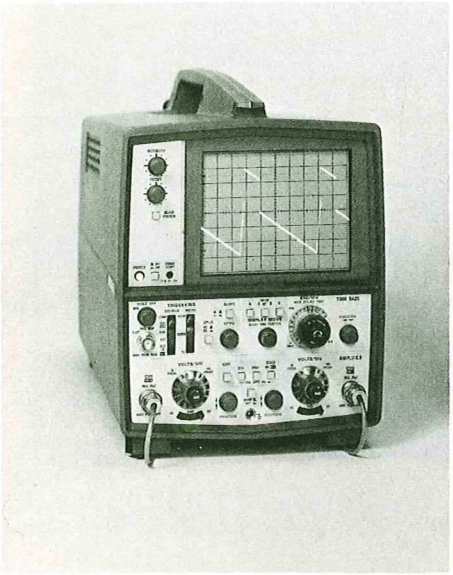
The FG 504, in addition to its pulse generation capabilities, offers all the versatility of a high performance function generator: a wide range sweep generator that provides logarithmic and linear sweeps; sine, haversine, $\sin^2$, square, and triangle waves; variable symmetry and rise time controls; amplitude and frequency modulation; phase lock operation; independent selection of start and stop frequencies for swept applications; and very low output frequencies to 0.001 Hz.

Conclusion

For the most part, pulse generators are the easiest to use when pulses are your only requirement. On the other hand, high quality function generators which include pulse capabilities suitable for digital logic and other applications offer greater versatility. One should carefully consider these and other differences before deciding which instrument is best suited to his or her needs.

New Products

The T935A Portable Oscilloscope



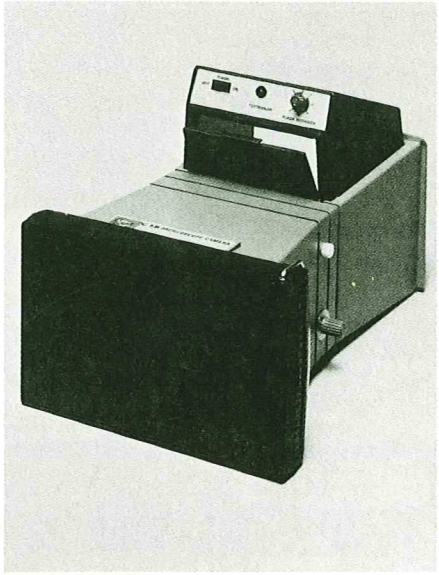
Two New T900-Series Oscilloscopes

The new T932A and T935A combine features previously not found together in low-cost instruments — including dual trace, differential display, versatile triggering capabilities, full sensitivity x-y and variable trigger holdoff.

The T932A boasts a 35-MHz bandwidth at 2 mV/div and 10 ns/div sweep rate with X10 magnification control. The T935A, identical in all these respects to the T932A also has delayed sweep capabilities.

Like the other T900 family members (with bandwidths starting at 10-MHz), the T932A and T935A are reliable, low-cost instruments well suited for education, consumer electronics repair, and other general service applications. Features like color-coded front panels, a beam finder, built-in delay lines, and a large 8 x 10 cm crt are designed to make accurate scope measurements easier than ever.

The New C-5B Oscilloscope Camera



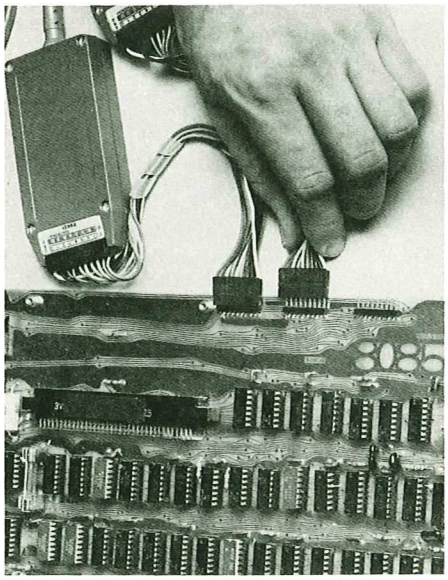
Low-cost Crt Camera Features Electric Shutter and Sharp Three-Element Lens.

The new C-5B Oscilloscope Camera is an improved version of our popular model C-5A. The camera now features a reliable electric shutter with speeds from 0.1 to 5 seconds and an improved fixed-focus, three-element, f16 lens.

The C-5B, with its interchangeable adapter hoods, mounts easily to the crt bezel of most TEKTRONIX oscilloscopes and small monitors. All adapter hoods have an opening in the top for a lift-up viewing door or a Xenon flash unit (to illuminate the crt on instruments without built-in graticule illumination). The flash unit has a flipdown viewing door.

The C-5B is the perfect camera for those requiring waveform photography on a limited budget and who don't need the more-sophisticated features of other TEKTRONIX cameras, such as single-sweep recording capability. It is especially recommended for use with the low-cost TEKTRONIX T900-Series Oscilloscopes.

Harmonica Connector to Logic Probe Adapter



New Logic Probe Adapter Speeds Multi-Pin Connection

Tektronix' new 10-Pin Harmonica Connector provides efficient, time saving plug-in capabilities for the P6451 Probe, used with the 7D01 and LA501 Logic Analyzers. The Harmonica Connector's circuitry terminal eliminates the time consuming mechanics of attaching connectors individually. This feature gives convenient access to data during development, manufacturing, and service of circuitry designed to accommodate the .025-inch square pins of the Harmonica Connector's plug. Also, the probe head terminal attaches quickly and easily to the P6451 Probe. The patented 10-pin Harmonica Connector contains one ground, one clock or qualifier, and eight data lines.

Prices of these products are shown on the inquiry card in the centerfold. Use the card to request further information or a demonstration.

4/78
A-3921

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