GPIB SYSTEM CONCEPTS

The first of two parts in this appendix describes the digital interface specified in IEEE Standard 488-1978, "Standard Digital Interface for Programmable Instrumentation." At Tektronix, the digital interface is commonly called the General Purpose Interface Bus (GPIB). This first part discusses signal-line definitions and other GPIB fundamentals.

WHAT IS THE GPIB?

The GPIB is a digital interface that allows efficient communication between the components of an instrumentation system.

The primary purpose of the GPIB is to connect self-contained instruments to other instruments or devices. This means that the GPIB is an interface system independent of device functions.

There are four elements of the GPIB: mechanical, electrical, functional, and operational.

Of these four, only the last is device-dependent. Operational elements state the way in which an instrument reacts to a signal on the bus. These reactions are device-dependent characteristics and state the way in which the instruments use the GPIB via application software.

Mechanical Elements. The standard defines the mechanical elements: cables and connectors. Standardizing the connectors and cables ensures that GPIB-compatible instruments can be physically linked together with complete pin compatibility.

The connector has 24 pins, with 16 assigned to specific signals and eight to shields and grounds. Instruments on the bus may be arranged in a linear or star configuration.

Electrical Elements. The voltage and current values required at the connector nodes for the GPIB are based on TTL technology (power source not to exceed +5.25V referenced to logic ground). The standard defines the logic levels as follows. Logical 1 is true state, low-voltage level ($\leq$+0.8V), signal line is **asserted.** Logical 0 is false state, high-

BACKGROUND

The General Purpose Interface Bus (GPIB) is a control bus that interfaces with a microcomputer (processor) and external peripheral devices.

Prior to the original development of the GPIB by Hewlett-Packard in 1975, the CAMAC (Computer Automated and Measurement Control) interface was developed by the nuclear industry. The IEEE promulgated this interface under several standards: Std 583 (Basic CAMAC), Std 595 (CAMAC), and Std 596 (Parallel CAMAC). CAMAC was comprised of a rigorously-specified mainframe (chassis) housing 25 plug-in modules. Outside of the nuclear industry and some electrical power companies, CAMAC was rarely used in industrial applications. The Parallel CAMAC transmitted data at a rate near 5 megabits/second over 86 lines. The major obstacles to its acceptance has been its expense and impracticality for microprocessor control.

The GPIB was developed as a control bus for instruments. This interface is oriented toward system configurations that use a variety of peripherals. Any type of 8-bit data is sent or received over the data bus in a byte-serial, bit-parallel fashion. Bus extenders can be used, at reduced data rates, to interface with common carriers. The GPIB is a very practical and inexpensive means to transmit or receive data from microcomputers over relatively short distances (up to 20 meters without bus extenders).

INTERFACE FUNCTION	SYMBOL
Source Handshake	SH
Acceptor Handshake	AH
Talker or Extended Talker	T or TE
Listener or Extended Listener	L or LE
Service Request	SR
Remote-Local	RL
Parallel Poll	PP
Device Clear	DC
Device Trigger	DT
Controller	C

Table 1. The ten major interface functions for the GPIB.

voltage level ($\geq$+2.0V), signal line is **not asserted.**

Messages can be sent over the GPIB as either active-true or passive-true signals. Passive-true signals occur at a high-voltage level and must be carried on a signal line using open-collector devices. Active-true signals occur at a low-voltage level.

Functional Elements. The functional elements of the GPIB cover three areas:

Ten interface functions that define the use of specific signal lines so that an instrument can receive, process, and send messages (the ten interface functions - with their allowable subsets - provide an instrumentation system with complete communications and control capabilities).

The **specific protocol** by which the interface functions send and receive their limited set of messages.

The **logical and timing relationships** between allowable states for the interface signal lines.

INTERFACE FUNCTIONS

Not every instrument on the bus has all ten functions (listed in table I), because only those functions important to a particular instrument's purpose need be implemented.

TYPICAL SYSTEM ON THE GPIB

Figure 1 illustrates an example of the GPIB and the nomenclature for the 16 active signal lines. Only four instruments are shown, but the GPIB can support up to 15 instruments connected directly to the bus. However, more than 15 devices can be interfaced to a single bus if they do not connect directly to the bus but are interfaced through a primary device. Such a scheme can be used for programmable plug-ins housed in a mainframe where the mainframe is addressed with a primary address code and the plug-ins are addressed with a secondary address code.

The instruments connected to a single bus cannot be separated by more than 20 meters (total cable length) and at least one more than half the number of instruments must be in the power-on state. To maintain the electrical characteristics of the bus, a device load must be connected for each two meters of cable length. Although instruments are usually spaced no more than two meters apart, they can be separated farther if the required number of device loads are lumped at any one point.

CONTROLLERS, TALKERS, AND LISTENERS

A **talker** is an instrument that can send data over the bus; a **listener** is an instrument that can accept data from the bus. No instrument can communicate until it is enabled to do so by the controller in charge of the bus.

A **controller** is an instrument that determines, by a software routine, which instrument will talk and which instruments will listen during any given time interval. The controller also has the ability to assign itself as a talker or listener whenever the program routine requires. In addition to designating the current talker and listeners for a particular communication sequence, the controller has the task of sending special codes and commands (called **interface messages**) to any or all of the instruments on the bus.

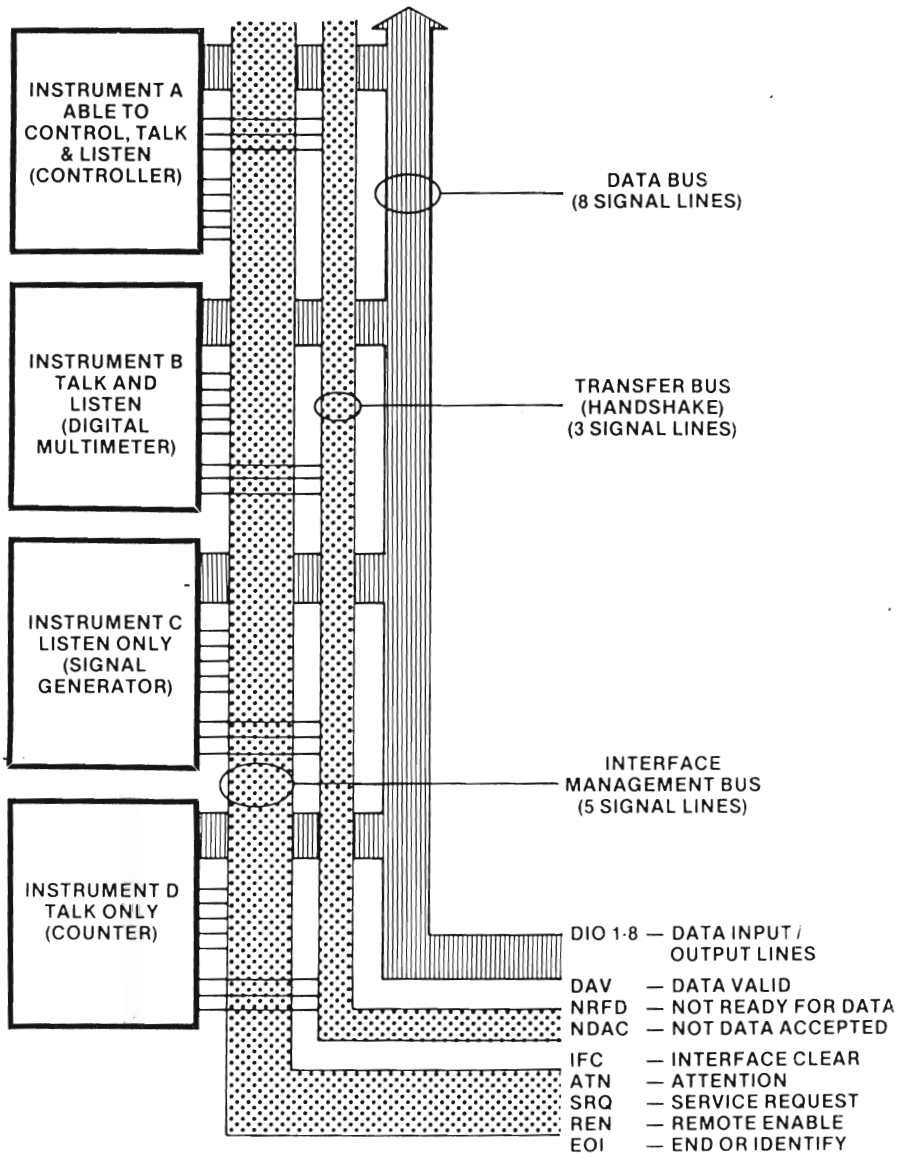

Figure 1. A typical system using the general purpose interface bus (GPIB).

ASCII & IEEE 488 (GPIB) CODE CHART

B7 B6 B5 / BITS / B4 B3 B2 B1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	CONTROL		NUMBERS SYMBOLS		UPPER CASE		LOWER CASE	
0 0 0 0	0 NUL 0 0	20 DLE 10 16	40 SP 20 32	60 0 30 48	100 @ 40 64	120 P 50 80	140 ` 60 96	160 p 70 112
0 0 0 1	1 GTL SOH 1 1	21 LLO DC1 11 17	41 ! 21 33	61 1 31 49	101 A 41 65	121 Q 51 81	141 a 61 97	161 q 71 113
0 0 1 0	2 STX 2 2	22 DC2 12 18	42 " 22 34	62 2 32 50	102 B 42 66	122 R 52 82	142 b 62 98	162 r 72 114
0 0 1 1	3 ETX 3 3	23 DC3 13 19	43 # 23 35	63 3 33 51	103 C 43 67	123 S 53 83	143 c 63 99	163 s 73 115
0 1 0 0	4 SDC EOT 4 4	24 DCL DC4 14 20	44 $ 24 36	64 4 34 52	104 D 44 68	124 T 54 84	144 d 64 100	164 t 74 116
0 1 0 1	5 PPC ENQ 5 5	25 PPU NAK 15 21	45 % 25 37	65 5 35 53	105 E 45 69	125 U 55 85	145 e 65 101	165 u 75 117
0 1 1 0	6 ACK 6 6	26 SYN 16 22	46 & 26 38	66 6 36 54	106 F 46 70	126 V 56 86	146 f 66 102	166 v 76 118
0 1 1 1	7 BEL 7 7	27 ETB 17 23	47 ' 27 39	67 7 37 55	107 G 47 71	127 W 57 87	147 g 67 103	167 w 77 119
1 0 0 0	10 GET BS 8 8	30 SPE CAN 18 24	50 (28 40	70 8 38 56	110 H 48 72	130 X 58 88	150 h 68 104	170 x 78 120
1 0 0 1	11 TCT HT 9 9	31 SPD EM 19 25	51) 29 41	71 9 39 57	111 I 49 73	131 Y 59 89	151 i 69 105	171 y 79 121
1 0 1 0	12 LF A 10	32 SUB 1A 26	52 * 2A 42	72 : 3A 58	112 J 4A 74	132 Z 5A 90	152 j 6A 106	172 z 7A 122
1 0 1 1	13 VT B 11	33 ESC 1B 27	53 + 2B 43	73 ; 3B 59	113 K 4B 75	133 [5B 91	153 k 6B 107	173 { 7B 123
1 1 0 0	14 FF C 12	34 FS 1C 28	54 , 2C 44	74 < 3C 60	114 L 4C 76	134 \ 5C 92	154 l 6C 108	174 ¦ 7C 124
1 1 0 1	15 CR D 13	35 GS 1D 29	55 − 2D 45	75 = 3D 61	115 M 4D 77	135] 5D 93	155 m 6D 109	175 } 7D 125
1 1 1 0	16 SO E 14	36 RS 1E 30	56 . 2E 46	76 > 3E 62	116 N 4E 78	136 ^ 5E 94	156 n 6E 110	176 ~ 7E 126
1 1 1 1	17 SI F 15	37 US 1F 31	57 / 2F 47	77 UNL ? 3F 63	117 O 4F 79	137 UNT _ 5F 95	157 o 6F 111	177 RUBOUT (DEL) 7F 127
	ADDRESSED COMMANDS	UNIVERSAL COMMANDS	LISTEN ADDRESSES		TALK ADDRESSES		SECONDARY ADDRESSES OR COMMANDS	

Interface messages are sent with ATN asserted.

KEY

octal 25 PPU — GPIB code
NAK — ASCII character
hex 15 21 — decimal

Figure 2. ASCII & IEEE 488 (GPIB) Code Chart.

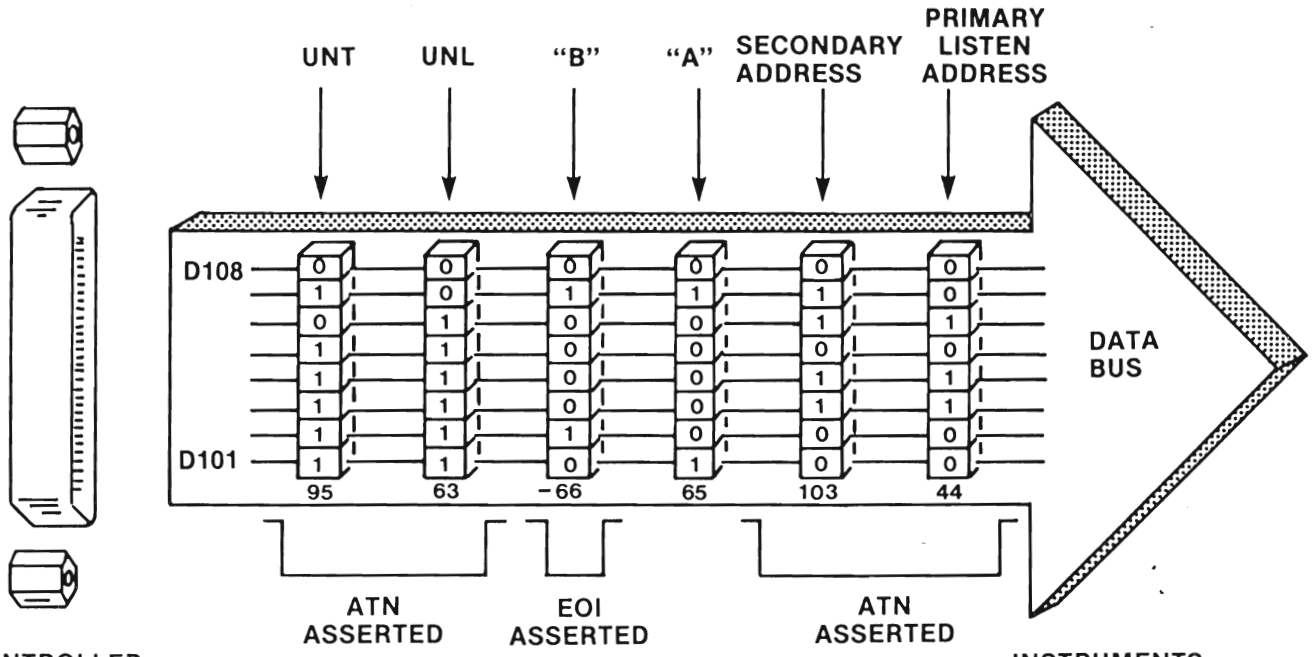

Figure 3. An example of data byte traffic on the GPIB.

INTERFACE MESSAGES

The IEEE standard specifies that the interface messages, as shown in figure 2, ASCII & IEEE 488 (GPIB) Code Chart, be used to address and control instruments interfaced to the GPIB. Interface messages are sent and received only when the controller asserts the ATN bus line. The user can correlate interface message coding to the ISO 7-bit code by relating data bus lines DI01 through DI07 to bits 1 through 7, respectively.

Interface messages include the primary talk and listen addresses for instruments on the bus, addressed commands (only instruments previously addressed to listen respond to these commands), universal commands (all instruments, whether they have been addressed or not respond to these), secondary addresses for devices interfaced through their primary instrument, and secondary commands. At present, the standard classifies only two interface messages as secondary commands, Parallel Poll Enable (PPE) and Parallel Poll Disable (PPD). (Parallel Poll Enable means that *after* the controller configures the system for a parallel poll (PPC command), all instruments respond at the same time with status information on receipt of PPE.)

DEVICE-DEPENDENT MESSAGES

The IEEE 488-1975 does not specify coding of device-dependent messages, messages that control the device's internal operating functions. After addressing (via interface messages) a talker and listener(s), the controller unasserts the ATN bus line. When ATN becomes false, any commonly-understood 8-bit binary code may be used to represent a device-dependent message.

The standard recommends that the alphanumeric codes associated with the numbers, symbols, and upper case characters (decimal 32 to decimal 94) in the ASCII Code Chart be used for device-dependent messages. One example of a device-dependent message is the ASCII character string

MODE V; 2.5MV; FREQ 1E3

which may tell an instrument to set its front-panel controls to the voltage mode, with 2.5 millivolt output at a frequency of 1000Hz.

When 8-bit binary codes other than the ISO 7-bit code are used for device-dependent messages, the most significant bit must be on data line DI08 (for bit 8).

To summarize the difference between interface and device-dependent messages, remember that any message sent or received when the ATN line is asserted (true) is an interface message. Any message (data bytes) sent or received when the ATN line is unasserted (false) is a device-dependent message.

GPIB SIGNAL LINE DEFINITIONS

Figure 1 shows the 16 signal lines of the GPIB functionally divided into three component busses: an eight-line data bus, a three-line transfer control (handshake) bus, and a five-line management bus.

The Data Bus. The data bus has eight bidirectional signal lines, DI01 through DI08. Information, in the form of data bytes, is transferred over this bus. A handshake sequence between an enabled talker and the enabled listeners transfers one data byte (eight bits) at a time. Data bytes in an interface or device-dependent message are sent and received in a byte-serial, bit-parallel fashion over the data bus.

Since the GPIB handshake sequence is an asynchronous operation, the data transfer rate is only as fast as the slowest instrument involved in a data byte transfer at any one time. A talker cannot place data bytes on the bus faster than any one listener can accept them.

Figure 3 illustrates the flow of data bytes when a typical controller sends ASCII data to an assigned listener on the bus. The first data byte, decimal 44, enables device 12 as a primary listener and the secondary address, decimal 108, enables a plug-in device as the final destination of the data to follow. The data is the two ASCII characters, A and B (decimal 65 and decimal 66).

The decimal value for B is specified as negative to activate the EOI line and signify the end of the device-dependent message. The controller activates the ATN line again and sends the universal unlisten (UNL) and untalk (UNT) commands to clear the bus. Six handshake cycles on the Transfer Bus are required to send the six data bytes.

The Transfer Bus (Handshake). Each time a data byte is sent over the data bus, an enabled talker and all enabled listeners execute a handshake sequence via the **transfer bus**. The transfer-bus signal lines are defined below. Figure 4 illustrates the basic timing relationship between the three signals. The ATN line is shown to illustrate the controller's role in the process. A flowchart for the handshake sequence is shown in figure 5.

Not Ready For Data (NRFD). An asserted NRFD signal line indicates one or more assigned listeners are not ready to receive the next data byte from the talker. When all of the assigned listeners for a particular data byte transfer have released NRFD, the NRFD line becomes unasserted (high). The RFD message (Ready For Data) tells the talker it may place the next data byte on the data bus.

Data Valid (DAV). The DAV signal line is asserted (low) by the talker after the talker places a data byte on the data bus. When asserted, DAV tells each assigned listener that a new data byte is on the data bus. The talker is inhibited from asserting DAV as long as any listener holds the NRFD signal line asserted.

Not Data Accepted (NDAC). Each assigned listener holds the NDAC signal line low-true (asserted) until the listener accepts the data byte currently on the data bus. When all assigned listeners accept the current data byte, the NDAC line becomes unasserted, telling the talker to remove the data byte from the bus. The DAC message (Data Accepted) tells the talker that all assigned listeners accepted the current data byte.

When one handshake cycle transfers one data byte, the listeners reset the NRFD line high and the NDAC line low before the talker asserts DAV for the next data byte transfer. NDAC and NRFD both high at the same time is an invalid state on the bus.

The Management Bus. The management bus is a group of five signal lines which are used to control the operation of the GPIB: IFC, ATN, SRQ, REN, and EOI.

Interface Clear (IFC). The system controller asserts the IFC signal line to place all interface circuitry in a predetermined quiescent state which may or may not be the power-on state.

Only the system controller can generate this signal. IEEE 488-1975 specifies that only three interface messages (universal commands) be recognized while IFC is asserted: Device Clear (DCL), Local Lockout (LLO), and Parallel Poll Unconfigure (PPU).

Attention (ATN). A controller asserts the ATN signal line when instruments connected to the bus are being enabled as talkers or listeners and for other interface control traffic. As long as the ATN signal line

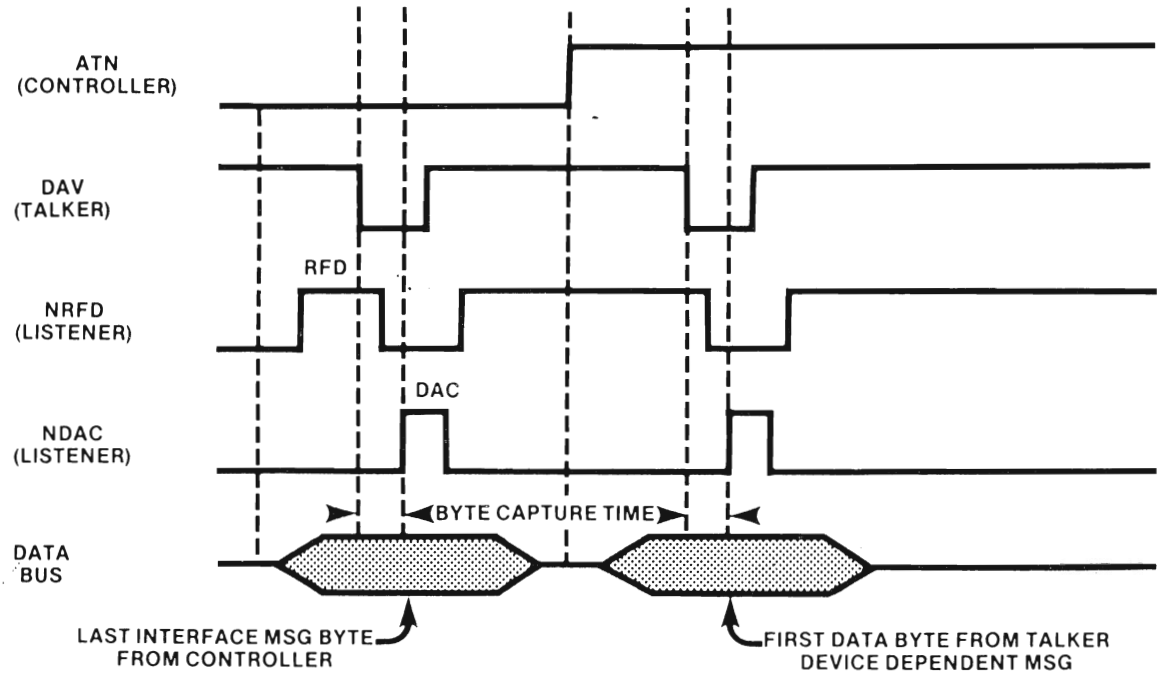

Figure 4. A typical handshake timing sequence (idealized). Byte capture time is dependent on the slowest instrument involved in the handshake.

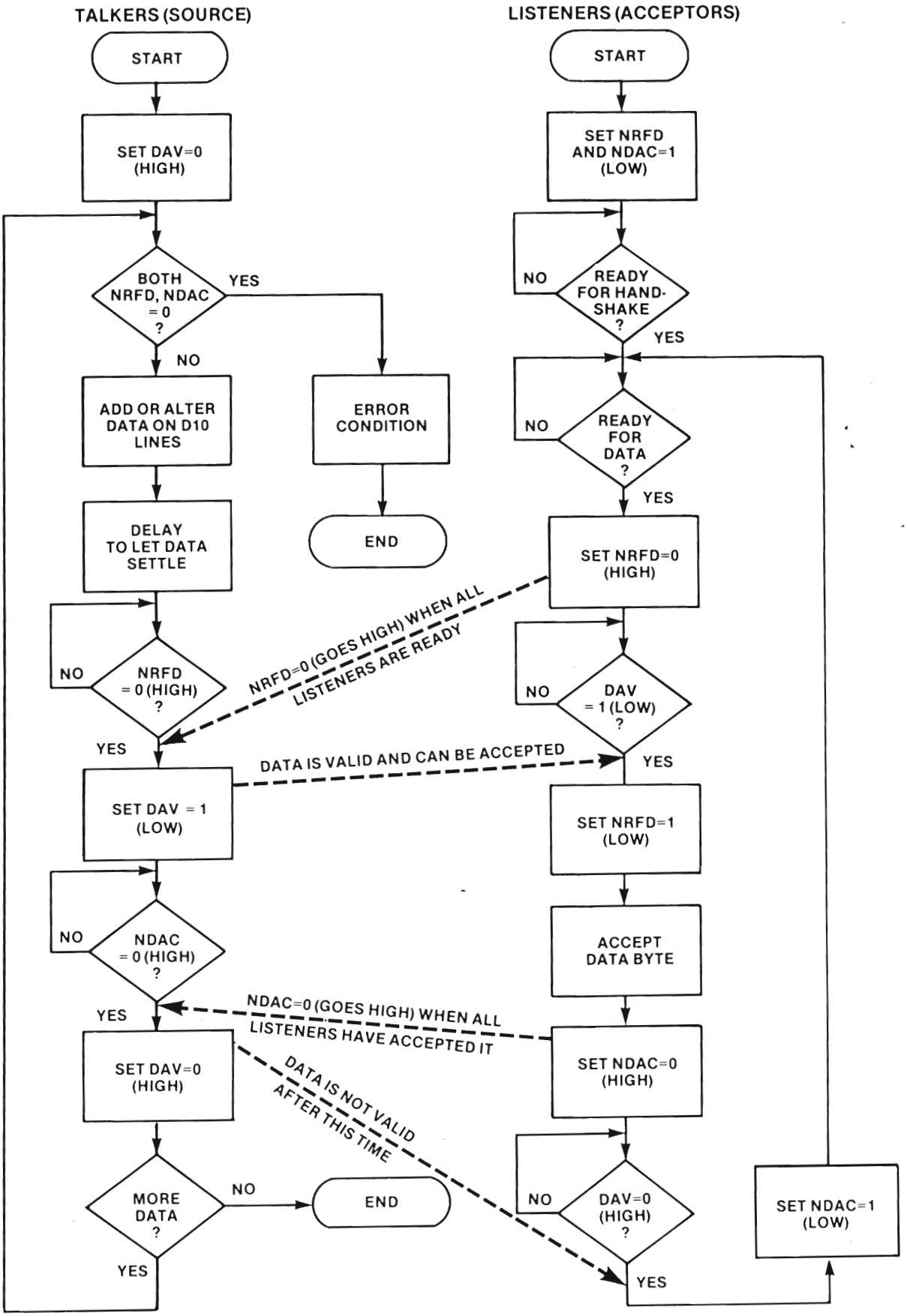

Figure 5. The handshake flow chart.

is asserted (ATN = 1), only instrument address codes and control messages are transferred over the data bus. With the ATN signal line unasserted, only those instruments enabled as a talker and listener(s) can transfer data. Only the controller can generate the ATN signal.

Service Request (SRQ). Any instrument connected to the bus can request the controller's attention by asserting the SRQ line. The controller responds by asserting ATN and executing a serial poll to determine which instrument is requesting service. (An instrument requesting service identifies itself by asserting its DI07 line after being addressed.) After the instrument requesting service is found, program control is transferred to a service routine for that instrument. When the service routine is completed, program control returns to the main program. When polled, the instrument requesting service unasserts the SRQ line.

Remote Enable (REN). The system controller asserts the REN signal line whenever the interface system operates under remote program control. Used with other control messages, the REN signal causes an instrument on the bus to select between two alternate sources of programming data. A remote-local interface function indicates to an instrument that the instrument will use either information input from the front-panel controls (Local) or corresponding information input from the interface (Remote).

End or Identify (EOI). A talker can use the EOI to indicate the end of a data-transfer sequence. The talker asserts the EOI signal line as the last byte of data is transmitted. In this case, EOI is essentially a ninth data line and must observe the same setup times as the DI0 lines. When the controller is listening, it assumes that a data byte received is the last byte in the transmission (if the EOI signal line has been asserted). When the controller is talking, it may assert the EOI signal line as the last byte is transferred. The EOI signal is also asserted with the ATN signal if the controller conducts a parallel polling sequence. EOI is not used during serial polling.

FOR MORE INFORMATION

For detailed information on GPIB specifications, refer to IEEE 488-1975 (Revised 1978), published by the Institute of Electrical and Electronics Engineers, 245 East 47th Street, New York, New York 11117.

GPIB INTERFACE FUNCTIONS AND MESSAGES

This second of two parts in this appendix discusses interface functions and the protocol for transferring data between instruments on the GPIB. This is intended to help you grasp the basics of GPIB operation as an aid in applying IEEE Standard 488-1978.

NTRODUCTION

The ten interface functions of the GPIB (listed in table 1) provide a variety of capabilities and options for an instrumentation system. These functions may be implemented in, or for, any particular instrument with nstrument hardware or with a programming routine (software).

INTERFACE FUNCTION	SYMBOL
Source Handshake	SH
Acceptor Handshake	AH
Talker or Extended Talker	T or TE
Listener or Extended Listener	L or LE
Service Request	SR
Remote-Local	RL
Parallel Poll	PP
Device Clear	DC
Device Trigger	DT
Controller	C

Table 1. The ten major interface functions for the GPIB.

Figure 1 illustrates the basic linkage between a GPIB controller and the interface functions implemented in another instrument on the bus. For a particular measurement or stimulus device, any or all of nine possible interface functions (SH through PP) may be selected to be linked to the tenth function (controller, C). Only those functions necessary for an instrument's purpose need be implemented by the instrument's designers; it is not likely that one instrument has all ten interface functions. For example, an instrument generally doesn't need to implement the Parallel Poll (PP) function if the instrument can respond to a serial polling sequence from the controller-in-charge (there may be more than one controller in a system).

The following is a discussion of the interface functions and their relationship to interface messages and commands.

The interface messages discussed in this article are shown in the ASCII and IEEE (GPIB) Code Chart in Table 2. All interface messages and commands, except IFC, discussed in this article are sent and received over the GPIB with the ATN line asserted low true.

TALKER AND LISTENER FUNCTIONS (T/TE AND L/LE)

Although discussed under one heading, the T/TE and L/LE functions are independent of each other.

The T and TE functions provide an instrument and its secondary devices, if any, with the capability of sending device-dependent data (or, in the case of a controller, the capability to send interface messages or device-dependent program data) over the GPIB. The T (Talker) function is a normal function for a talker and uses only a one-byte address code called **MTA** (My Talk Address); the TE (Talker Extended) function uses a two-byte address code: an **MTA** code followed by an **MSA** code (My Secondary Address).

Only one instrument in the system can be in the talker active state at any given time. A non-controller commences talking when **ATN** is released and continues its talker status until an Interface Clear (**IFC**) message occurs, an Untalk (**UNT**) command is received from the controller-in-charge, the instrument is addressed as a listener (receives My Listen Address, **MLA**), or another instrument is addressed as a talker when a data byte called **OTA**, Other Talk Address, appears on the bus.

One or more instruments on the bus (up to a maximum of 14) can be programmed for the L (Listener) function by use of their specific primary listen address (**MLA**). All or none of these instruments may be programmed for the LE (Listener Extended) function (if implemented). The LE function requires a secondary listen address (**MSA**).

An instrument on the bus may be a talker only, or a listener only, or have both functions (T/TE and L/LE). In any case, its address code has the form X10TTTTT for a talker and X01LLLLL for a listener. For instruments with both functions, the T-bit binary values are equal to the binary value of the L bits. The system operator sets these five bits by

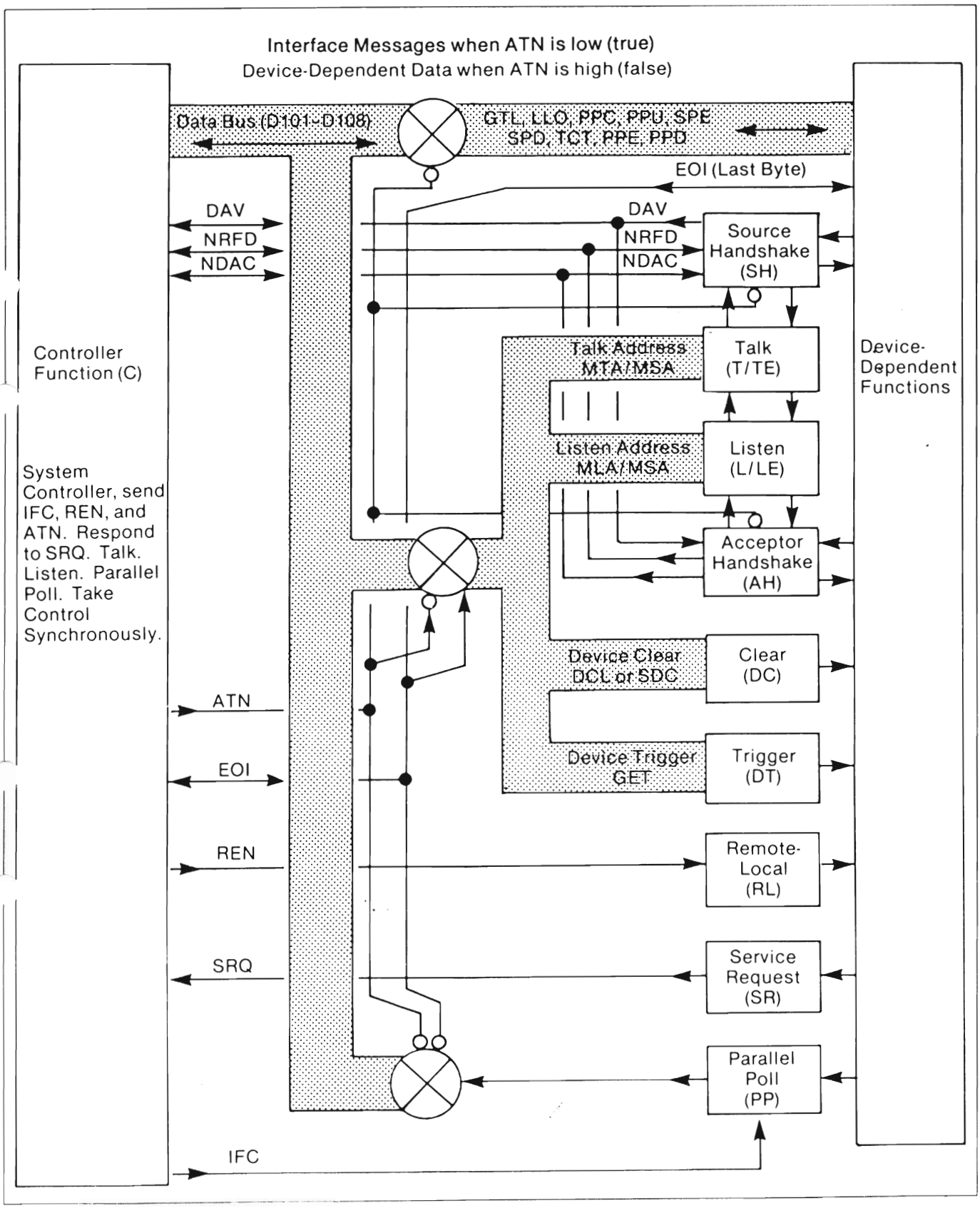

Figure 1. A maximum of ten interface functions can be interlinked between instruments on the GPIB.

means of address switches on each instrument before applying power to the system. The controller's address code may be implemented in software.

If an instrument has TE or LE functions, the five secondary address bits must not be set to the same value as the primary address bits. The system program, from the controller, designates the primary talker and primary listener status of the desired instruments by coding bits 6 and 7 (10 for talker and 01 for listener). Secondary listen addresses (or commands) are represented by the controller sending both bits (6 and 7) as a 1.

A talker may assert the **EOI** line with the last data byte or send a special "end of message terminator" code so that the assigned listeners will know that the talker has no more data transfer to send.

SOURCE AND ACCEPTOR HANDSHAKE FUNCTIONS (SH AND AH)

Like the T/TE and L/LE functions, SH and AH are totally independent of each other. The SH function guarantees proper transmission of data, while the AH function guarantees proper reception of data. The interlocked handshake sequence between these functions guarantees asynchronous transfer of each data byte.

Both functions utilize the **DAV**, **RFD**, and **DAC** messages to transfer each byte. (For details, see figures 2 and 5 and the discussion under "The Transfer Bus (Handshake)" in GPIB Systems Concepts, the first part of this appendix.) The SH and AH functions in some instruments, but especially in controllers, allow the source (talker) to listen to itself, with or without **ATN** asserted. Both functions must respond to the **ATN** message within 200 nanoseconds.

DEVICE CLEAR FUNCTION (DC)

The Device Clear function allows a controller-in-charge to **clear** (initialize) an instrument, either individually or as part of a group of instruments. The group can be either a part or all of the addressed instruments in one system.

The controller (under program direction) asserts **ATN** and sends either the universal Device Clear command (**DCL**) or the Selected Device Clear command (**SDC**). When the **DCL** message is received, all instruments on the bus must clear or initialize their internal device functions. When the controller sends the **SDC** command, only those instruments which have previously been addressed to listen must clear or initialize their internal device functions. The IEEE 488 standard does not specify the state an instrument goes to as a result of the **DCL** or **SDC** command; it may be, but does not have to be, the power-up default setting.

DEVICE TRIGGER FUNCTION (DT)

The Device Trigger function allows the controller-in-charge to start the basic operation of an instrument, either by itself or as part of a group of instruments. The group may be either a part or all of the addressed instruments in one system. The IEEE 488 standard does not specify an instrument's basic operation when it receives the **GET** (Group Executive Trigger) command. To issue this command, the controller asserts **ATN**, sends the listen addresses of the instruments which are to respond to the trigger, and then sends the **GET** message.

Once an instrument starts its basic operation, the instrument must not respond to subsequent trigger-state transitions until the current operation is complete. Only after completing the operation can the instrument start the same operation in response to the next **GET** message; thus the basic operating time is the major factor that determines how fast the instrument(s) can be repeatedly "triggered."

REMOTE-LOCAL FUNCTION (RL)

The Remote-Local (RL) function provides an instrument with the capability to select between two sources of information input. The function indicates to the instrument that its internal device-dependent functions respond to information input from the front panel (Local) or corresponding information input from the GPIB (Remote). Only the system controller is permitted to assert the **REN** line, whether or not it is the controller-in-charge at the time.

When the system controller asserts the **REN** line, an instrument on the GPIB goes to the remote mode when it is addressed as a listener with its primary address, not before. In this case only, the primary listen address is sufficient to cause an instrument to go to the remote mode. For example, if several instruments have a different secondary, but a common primary address, they will all go to the remote mode when the primary address is received.

An instrument remains in the remote mode until the **REN** line is released, a front-panel switch on the instrument is activated to request the local mode, or a Go to Local (**GTL**) command is received while the instrument is enabled as a listener. However, the controller can disable the local mode function of an instrument by sending a Local Lockout (**LLO**) command, which applies to all instruments on the bus, addressed or not. The **UNL** (Unlisten) command does not return an instrument to a local mode.

All instruments must recognize when the **REN** line goes false (a high voltage level) and go to the local mode within 100 microseconds. If data bytes are still being placed on the data bus when **REN** goes false, the system program should assure that the data bytes are sent and received with the knowledge that the system is in a local mode, not remote.

CONTROLLER FUNCTION (C)

The Controller function provides the capability to send primary talk and listen addresses, secondary addresses, and universal commands to all instruments on the bus, and secondary commands to previously-addressed instruments. The controller function also provides the capability of responding to a service request (**SRQ**) message or conducting a parallel poll routine to determine the status of any or all instruments.

If an instrumentation system has more than one controller, only the system controller is allowed to assert the **IFC** and **REN** lines at any time during the system operation, whether or not it is the controller-in-charge at the time.

ASCII & IEEE 488 (GPIB) CODE CHART

B7 B6 B5 / BITS / B4 B3 B2 B1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
	CONTROL		NUMBERS SYMBOLS		UPPER CASE		LOWER CASE	
0 0 0 0	0 NUL 0 0	20 DLE 10 16	40 SP 20 32	60 0 30 48	100 @ 40 64	120 P 50 80	140 ` 60 96	160 p 70 112
0 0 0 1	1 GTL SOH 1 1	21 LLO DC1 11 17	41 ! 21 33	61 1 31 49	101 A 41 65	121 Q 51 81	141 a 61 97	161 q 71 113
0 0 1 0	2 STX 2 2	22 DC2 12 18	42 " 22 34	62 2 32 50	102 B 42 66	122 R 52 82	142 b 62 98	162 r 72 114
0 0 1 1	3 ETX 3 3	23 DC3 13 19	43 # 23 35	63 3 33 51	103 C 43 67	123 S 53 83	143 c 63 99	163 s 73 115
0 1 0 0	4 SDC EOT 4 4	24 DCL DC4 14 20	44 $ 24 36	64 4 34 52	104 D 44 68	124 T 54 84	144 d 64 100	164 t 74 116
0 1 0 1	5 PPC ENQ 5 5	25 PPU NAK 15 21	45 % 25 37	65 5 35 53	105 E 45 69	125 U 55 85	145 e 65 101	165 u 75 117
0 1 1 0	6 ACK 6 6	26 SYN 16 22	46 & 26 38	66 6 36 54	106 F 46 70	126 V 56 86	146 f 66 102	166 v 76 118
0 1 1 1	7 BEL 7 7	27 ETB 17 23	47 ' 27 39	67 7 37 55	107 G 47 71	127 W 57 87	147 g 67 103	167 w 77 119
1 0 0 0	10 GET BS 8 8	30 SPE CAN 18 24	50 (28 40	70 8 38 56	110 H 48 72	130 X 58 88	150 h 68 104	170 x 78 120
1 0 0 1	11 TCT HT 9 9	31 SPD EM 19 25	51) 29 41	71 9 39 57	111 I 49 73	131 Y 59 89	151 i 69 105	171 y 79 121
1 0 1 0	12 LF A 10	32 SUB 1A 26	52 * 2A 42	72 : 3A 58	112 J 4A 74	132 Z 5A 90	152 j 6A 106	172 z 7A 122
1 0 1 1	13 VT B 11	33 ESC 1B 27	53 + 2B 43	73 ; 3B 59	113 K 4B 75	133 [5B 91	153 k 6B 107	173 { 7B 123
1 1 0 0	14 FF C 12	34 FS 1C 28	54 , 2C 44	74 < 3C 60	114 L 4C 76	134 \ 5C 92	154 l 6C 108	174 ¦ 7C 124
1 1 0 1	15 CR D 13	35 GS 1D 29	55 − 2D 45	75 = 3D 61	115 M 4D 77	135] 5D 93	155 m 6D 109	175 } 7D 125
1 1 1 0	16 SO E 14	36 RS 1E 30	56 . 2E 46	76 > 3E 62	116 N 4E 78	136 ^ 5E 94	156 n 6E 110	176 ~ 7E 126
1 1 1 1	17 SI F 15	37 US 1F 31	57 / 2F 47	77 UNL ? 3F 63	117 O 4F 79	137 UNT _ 5F 95	157 o 6F 111	177 RUBOUT (DEL) 7F 127
	ADDRESSED COMMANDS	UNIVERSAL COMMANDS	LISTEN ADDRESSES		TALK ADDRESSES		SECONDARY ADDRESSES OR COMMANDS	

Interface messages are sent with ATN asserted.

KEY

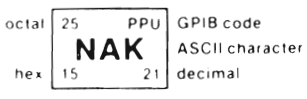

Table 2. ASCII and IEEE 488 (GPIB) Code Chart.

The controller function has specified time intervals for certain operations. For example, the execution time for parallel polling instruments on the bus cannot be less than 2 microseconds. The **ATN** message must have a controller delay of at least 500 nanoseconds to allow a current talker to see the **ATN** line asserted before placing a new data byte on the bus. The **IFC** message must be asserted for at least 100 microseconds.

If a controller requests system control from another controller and receives an internal message to send the remote enable message (**REN**), the controller must verify that the **REN** line remains unasserted (false) for at least 100 microseconds before asserting **REN.** The time interval that **REN** is asserted depends on the remote programming sequence and will vary with the program.

If a controller is in the controller active wait state and does not receive an internal message to conduct a parallel poll, it must wait at least 1.5 microseconds before going to the controller active state to give the NRFD, NDAC, and EOI lines sufficient time to assume their valid states.

Taking Control (Asynchronous or Synchronous). All data bytes transmitted over the GPIB with **ATN** asserted are interpreted as system control information. Asserting **ATN** directly at any moment is an asynchronous operation with respect to the bus and may cause loss of data if a handshake cycle is in progress. To prevent loss of data, a controller can take control synchronously with the handshake cycle (if it is in progress) by first asserting the NRFD line to stop the next handshake cycle, and then automatically asserting **ATN** when the current talker releases DAV.

Taking control synchronously presents problems; the **ATN** line may not become asserted automatically if the data transfer has stopped for some reason. For example, (1) the talker may have finished talking (sent the last data byte), or (2) the talker may be very slow to send the next data byte, or (3) an instrument on the bus may not be functioning properly. Programmers can solve the first problem if they know that all talkers on the bus assert **EOI** with the last data byte. The second and third problems may require asserting **IFC** to clear the bus and then asserting **ATN** asynchronously.

Performing a Serial Poll. The controller may conduct a serial poll at any time, whether or not an instrument has asserted the **SRQ** line. Most, but not all, instruments have the Service Request (**SRQ**) function.

To perform a serial poll, the controller first asserts **ATN** and issues the Untalk (**UNT**) and Unlisten (**UNL**) commands. The controller then sends Serial Poll Enable (**SPE**) command, followed by the talk address of the first instrument to be polled. The controller then releases **ATN**, and the addressed talker responds by sending its status byte over the data bus. If the addressed talker has requested service, it must assert bit seven of the status byte and encode the remaining seven bits of the status byte to indicate the reason for asserting **SRQ**. Status bytes are device-dependent and are not specified in the IEEE 488 standard.

An addressed instrument will release its **SRQ** line when serial polled, but other instruments may still be holding it asserted. When the controller has read the status byte of an addressed instrument, it should send the **UNT** and Serial Poll Disable (**SPD**) commands before repeating the procedure to poll the remaining instruments. The routine should continue until the controller no longer detects **SRQ** asserted.

Performing a Parallel Poll. The Parallel Poll (PP) function provides an instrument with the capability to present one, and only one, bit of status information to the controller without being previously addressed to talk. The parallel polling capability requires a commitment by the system program to periodically conduct a parallel poll sequence.

When an instrument responds to a parallel poll, the single data bit presented to the controller may or may not indicate a need for service. If the data bit is used as a service request function, the controller should perform a serial poll in order to obtain a complete status byte.

Before an instrument can respond to a parallel poll, the GPIB system must first be configured. The controller asserts **ATN,** sends the **UNT** command followed by the listen addresses of all instruments to be included in the parallel poll, and then sends the Parallel Poll Configure (**PPC**) command. The **PPC** command is followed immediately with the Parallel Poll Enable (**PPE**) command to cause all listeners to go to the parallel poll standby state.

If the **EOI** line has been asserted along with **ATN,** all selected instruments have up to 200 nanoseconds to go to the parallel poll active state. **EOI** may be asserted along with the **PPE** command, or at any time after the **PPE** command. An instrument does not present its one bit of status information to the controller until it sees both **ATN** and **EOI** asserted.

The **PPE** message sent by the controller has the form X11OSPPP. Bit 4 (S) is called the sense bit, and PPP is an octal number (000=0 through 111=7) designating a specific data line (DI01 - DI08) that an instrument must assert if its internal status message has the same value as the sense bit (S may equal 1 or 0). If so designed, the controller can read the data lines while **ATN** is asserted to interpret the status of the instruments.

To conclude the parallel poll, the controller releases **EOI** and then sends the Parallel Poll Disable (**PPD**) command. If the system needs to be reconfigured, the Parallel Poll Unconfigure (**PPU**) command is sent, followed by the Unlisten (**UNL**) command.

Passing Control. As a controller-in-charge, the system controller (program) may relinquish control to any other instrument in the system capable of acting as a controller. The controller-in-charge first addresses the other controller as a talker, and then sends the Take Control (**TCT**) command and other desired control messages. The other controller becomes controller-in-charge when **ATN** is released. □