

The General-Purpose Interface Bus

Richard Gilbert
University of South Florida

The IEEE-488 interface standard, known as the General-Purpose Interface Bus, or GPIB, offers a uniform method of sending parallel code from one device to another. The standard has been revised twice, and communications based on it are now quite reliable. Most current problems with GPIB stem from inadequate documentation by individual manufacturers. This causes unnecessary implementation difficulties, especially when instruments from different sources are involved. This article addresses these problems by reviewing the IEEE-488 standard itself, discussing the methods of its implementation, and presenting specific applications to illustrate how the standard is applied to data-collection situations.

Review of the IEEE-488 standard

The fundamental requirements for a GPIB system are presented in Figure 1. The GPIB is here strictly defined as the wire connections between device 1 and device 2. (Although the term GPIB is often used to refer to a complete system, we limit its meaning to the cable that connects the devices in a GPIB system.) Figure 1 implies that an interface circuit is needed in each device attached to the GPIB; this interface is also defined by the IEEE-488 standard. Finally, the diagram suggests that a GPIB system requires a minimum of two devices, two interfaces, and one cable connection for each successful operation. In fact, there can be 15 devices on the GPIB. The maximum length of cable connecting a group of devices within a normal bus system is either two meters times the number of devices on the bus or 20 meters, whichever is less. Repeaters and optical bus extenders are available.

Figure 2 shows a simple GPIB system. This representation accents the fact that the usual minimal system requires a controller with its interface. The controller decides which device on the GPIB is to communicate with any other device(s).

The GPIB cable has IEEE-488-defined wire and connector requirements. The geometry of the connectors is such that it is best to purchase a GPIB cable when the GPIB instrument is obtained. There are 24 wires in the GPIB; 16 are used to transmit information and eight serve as ground wires. The wires are grouped as data lines, control lines, and management lines. The control lines are

- NRFD, not ready for data;
- DAV, data valid; and
- NDAC, not data accepted.

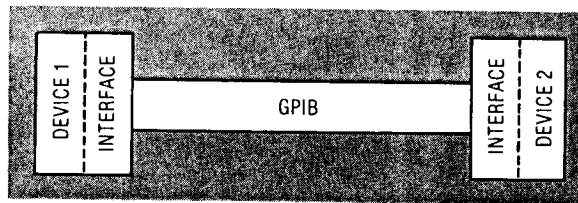


Figure 1. Minimum requirements for a GPIB system.

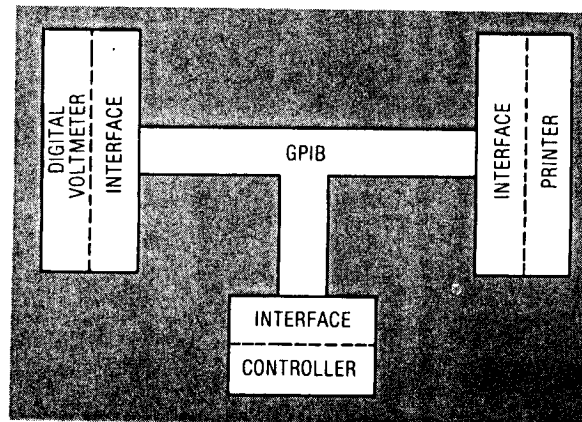


Figure 2. Usual minimal system configuration.

The management lines are

- ATN, attention;
- SRQ, service request;
- IFC, interface clear;
- EOI, end or identify; and
- REN, remote enable.

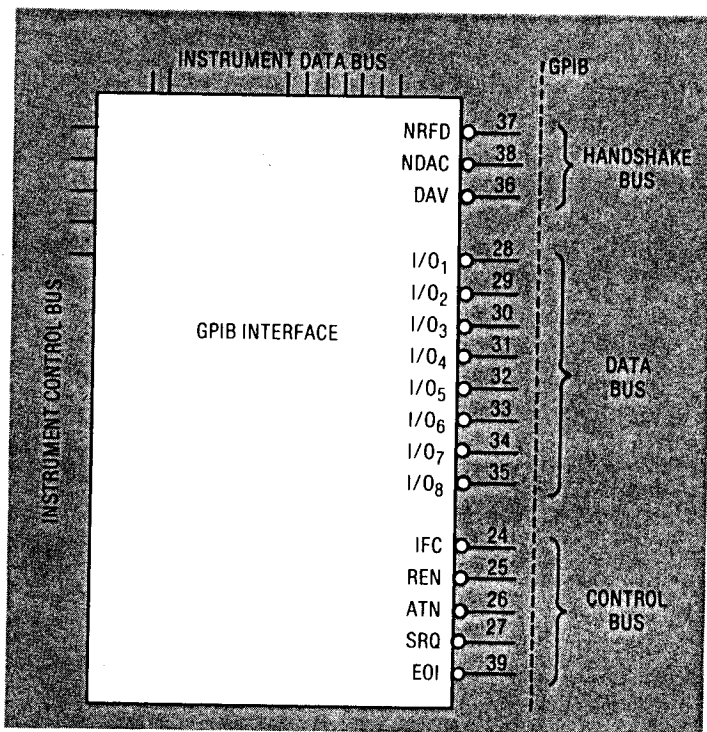


Figure 3. General view of GPIB interface.

Each of the 24 wires on the GPIB is connected via the standard connector to the GPIB interface on the IEEE-488-compatible device. The bus protocol is labeled *active low*, which indicates that the function defined by the wire in question is active when the wire is at zero volts. For example, if the GPIB interface is to recognize the *attention* function and then perform some predefined activity, the wire labeled ATN must be brought to zero volts.

Figure 3 presents a general view of an interface used in an IEEE-488 system. Several firms manufacture these interface chips. There are three groups of GPIB control connections:

(1) The *data bus* is the group of wires that provides the paths for the actual codes to be sent from one instrument interface to another.

(2) The *handshake bus* is used to establish communication across the interface before the actual code is transmitted on the data bus.

(3) The *control bus* contains the special-function wires that control and enhance the GPIB operations. Although there are five control functions in the control bus, a minimal system requires only ATN. If two controllers are to coexist on the bus, the IFC must be operational. The EOI and SRQ are control lines that allow the various instruments on the GPIB to communicate their status to the bus controller. The REN gives the controller a method of removing an instrument from local operator control.

Figure 4 accents the various duties of the GPIB interface, which has three general responsibilities. First, it prepares the digital code to be sent on the GPIB, an activity performed in the driver and receiver sections. Second, it encodes and decodes information that travels on the GPIB. Third, the interface performs the set of IEEE-488-defined functions.

The driver and receiver sections of the interface operate under IEEE-488 guidelines. If a GPIB system uses GPIB instruments, there will be no problems with the drivers and receivers. By contrast, if the user intends to communicate with equipment developed in-house, he must pay more attention to the bus line receiver and drivers.

The message-coding section of the GPIB interface is of interest for what it does, but not for how it does it. The coding section has the responsibility of providing the bus lines with a specific pattern of high and low voltage signals—logic zeros and ones—when a message character is to be sent over the GPIB.

Table 1 summarizes one character code used on the GPIB. This code, the American Standard Code for Information Interchange, or ASCII, contains control characters, letters of the alphabet, digits, and miscellaneous characters commonly found on typewriter keyboards. The table shows control characters in the first two columns and typewriter characters and digits in the next two columns. Capitals, small letters, and miscellaneous characters are in columns four, five, six, and seven. Although ASCII control characters are used in several communication schemes, they are not employed in a GPIB system.

Table 2 is an enlarged section of the ASCII code table, with two additional pieces of information. The first is the seven-bit pattern for each code element and the second is

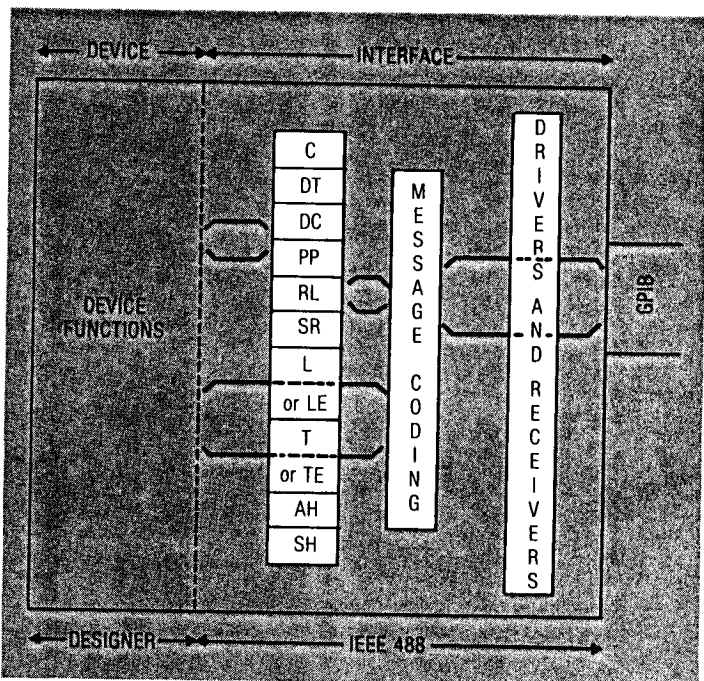


Figure 4. Outline of GPIB interface functions.

the decimal equivalent for each code element. A brief review of Table 2 indicates its function. For example, the percent character is located in column two, row five of both Tables 1 and 2 (note that there is a column zero and a row zero). This character can be represented as 37₁₀, as 25 in the hexadecimal code, or as the seven-bit pattern 0100101. This seven-bit pattern has a decimal value of 37. The bit patterns and decimal values for the other ASCII characters are column-oriented. Thus, the question-mark character has a 0111111 bit pattern and a decimal value of 63. The space bar, —, has a decimal value of 95 and a 10111111 bit pattern. The last ASCII character, DEL, has a 11111111 pattern and a decimal value of 127.

Interface functions. Figure 4 also summarizes the 10 interface functions supported by the IEEE-488 parallel communications standard. These include

- SH, the source handshake;
- AH, the acceptor handshake;
- T, the talk, and TE, the extended talk;
- L, the listen, and LE, the extended listen;
- SR, the service request;
- RL, the remote/local;
- PP, the parallel poll;
- DC, the device clear;
- DT, the device trigger; and
- C, the control functions.

The first four functions—SH, AH, T, and L—are the minimum requirements for successful GPIB system operation. The SH and AH functions are operated automatically by the interface.

Figure 5 presents the general task for the GPIB interface. It shows an eight-bit byte of code that is to be transferred via the GPIB from the device assigned as the talker to the device assigned as the listener. The bit pattern represents the ASCII code for the question-mark character. The eighth, and most significant, bit is assigned a default value when the seven-bit ASCII code is employed. The actual transfer of the question mark from the talker to the talker interface is talker-dependent and not defined by the IEEE-488 standard. However, the character transfer procedure from the talker interface to the listener interface via the GPIB data lines is defined by the GPIB standard. As the figure implies, the NRFD, NDAC, and DAV lines are needed to produce a successful character transfer down the data lines. The NRFD and NDAC lines are controlled by the listener; the DAV line is manipulated by the talker. The actual procedure for this code transfer is known as a GPIB handshake.

GPIB handshake. The GPIB handshake involves the systematic cycling of the voltage states on the NRFD and NDAC lines with those on the DAV line. The NRFD and NDAC lines are controlled by the listener's interface; the DAV line is the responsibility of the talker's interface. Row (1) of Figure 6 presents the conditions of these lines before the three-wire GPIB handshake begins. When the handshake is complete, the question mark will have been successfully sent from the talker to the listener.

Figure 6 shows that when the listener holds NRFD and NDAC at zero volts, it is not ready for any code transfer

from the talker and it is not processing any previous code character. Thus, the listener is not ready to participate in a handshake cycle.

The DAV line is controlled by the talker. In Figure 6, the line is initially at five volts and the talker has not placed valid data on the GPIB data lines.

Table 1.
American Standard Code for Information Interchange.

	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	\$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	␣	7	G	W	g	w
8	BS	CAN	(8	H	X	h	x
9	HT	EM)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	:	K	[k	{
C	FF	FS	.	<	L	\	l	
D	CR	GS	-	=	M]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

Table 2.
Enlarged section of ASCII code.

	B ₇ B ₆ B ₅				MOST SIGNIFICANT DIGITS				
	B ₄	B ₃	B ₂	B ₁	0	1	2	3	4
0	0	0	0	0	NUL ₀	DLE ₁₆	SP ₃₂	0 ₄₈	@ ₆₄
1	0	0	0	1	SOH ₁	DC1 ₁₇	! ₃₃	1 ₄₉	A ₆₅
2	0	0	1	0	STX ₂	DC2 ₁₈	" ₃₄	2 ₅₀	B ₆₆
3	0	0	1	1	ETX ₃	DC3 ₁₉	# ₃₅	3 ₅₁	C ₆₇
4	0	1	0	0	EOT ₄	DC4 ₂₀	\$ ₃₆	4 ₅₂	D ₆₈
5	0	1	0	1	ENQ ₅	NAK ₂₁	% ₃₇	5 ₅₃	E ₆₉

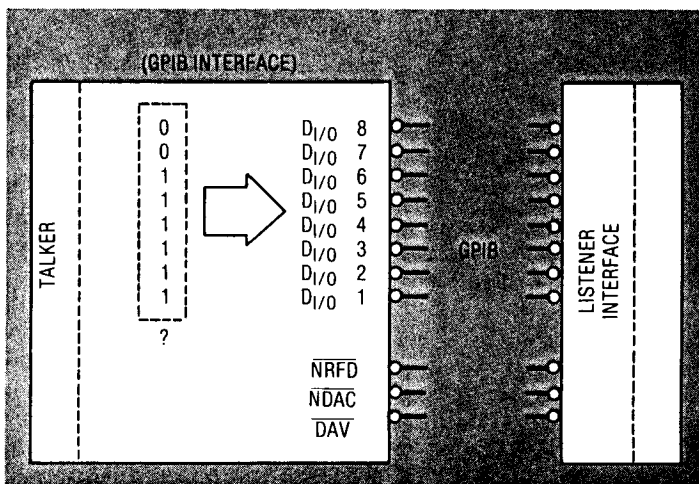


Figure 5. General task of GPIB interface.

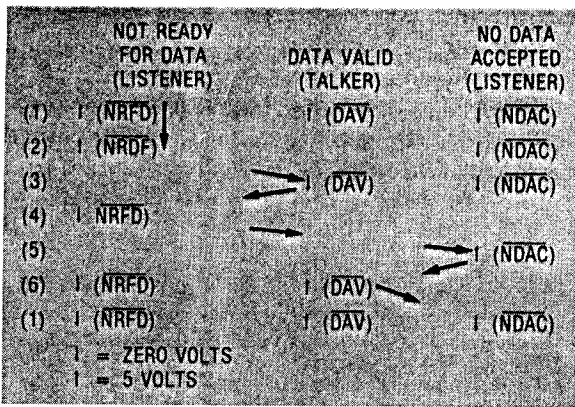


Figure 6. Summary of handshake cycles. Each line cycles states once per handshake.

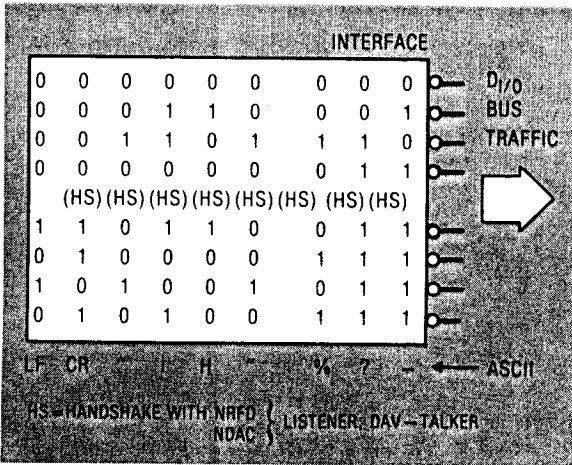


Figure 7. Procedure for code transfer down the bus.

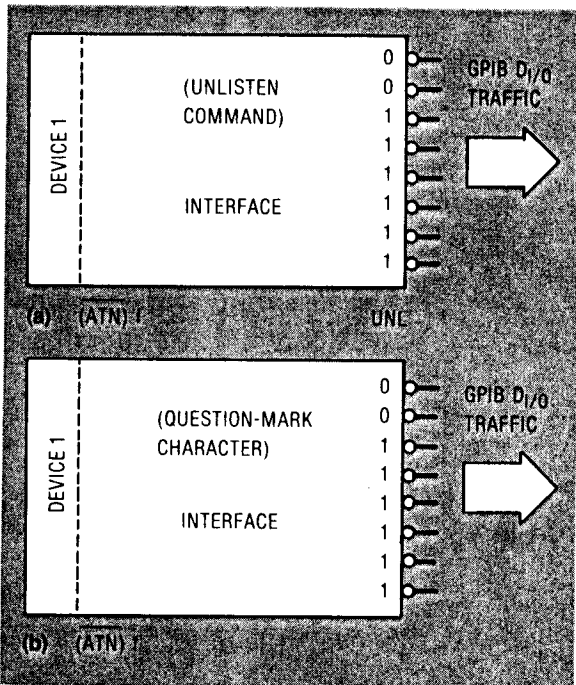


Figure 8. Code types for GPIB transmission; (a) commands and addresses; (b) messages and data.

Row (1) shows the initial state for the listener. Rows (2) through (6) illustrate one complete three-wire GPIB handshake cycle; row (2) indicates a change in the NRFD voltage from zero to five volts. This is interpreted by the talker as a signal to place the code character onto the GPIB data bus. Once this is accomplished, the talker brings DAV to zero volts, an operation shown in row (3). After DAV goes low, the listener interface proceeds to accept the code character that the talker interface has placed on the bus. Simultaneously, the listener returns NRFD to low to prevent the acceptance of an additional character, and begins to process the current code character, in this case the question mark. When the listener interface has successfully passed the received code character to its device, it brings NDAC to five volts, as shown in row (5). Once NDAC is sensed high by the talker interface, the talker DAV line is returned to five volts [row (6)]. This is the last step in the handshake cycle, which is then repeated for the next character.

The three-wire handshake only appears to be complex—it is, in fact, a straightforward procedure. Each wire goes through one voltage cycle for every handshake cycle. The NRFD line starts at zero volts and goes high when the listener desires a code character. It returns low after the listener interface collects the code from the bus. The talker's DAV line starts high and drops to zero volts when the talker interface has placed valid code on the bus. The line returns high when the code is invalid and a new code character is to be placed on the bus. Finally, the listener NDAC line starts low and goes high when the listener's interface has processed the transmitted code character and passed it to the listener. The NDAC line then returns low to begin the entire process again.

Figure 7 presents the general procedure for sending a series of ASCII characters onto the GPIB. The figure shows the bit patterns for a string of characters (—, ?, %, “, H, i, ”, CR, and LF) that are to be sent down the bus one at a time, with a handshake on each. The CR and LF are the ASCII carriage return and line feed control characters, respectively. They can be sent as code characters on the GPIB, but exercise no GPIB control activity.

As mentioned earlier, the handshake procedure, although required on each code transfer, is transparent to the user. It is, however, an asynchronous process that can cause problems for the user. Consider again steps (5) and (6) in Figure 6. The time involved in the transfer of code from the GPIB through the listener interface to the listener is listener-dependent. If the listener takes an inordinately long time to process the code character, the bus handshake process halts and waits for the NDAC to indicate that the code character has been accepted by the listener. A user could interpret such a delay as a fault in the GPIB system.

GPIB code. The discussion so far has centered on the transmission of ASCII characters from a talker to a listener via the GPIB. In practice, the situation is more complicated because some characters sent down the bus are not ASCII code.

Figure 8 shows the different types of eight-bit characters that can be sent on the GPIB. The type of character sent depends on the voltage condition on the ATN line. A

Table 3.
 GPIB command and address code.

		ATN							
		MOST SIGNIFICANT DIGIT							
		ADDRESS COMMAND GROUP	UNIVERSAL COMMAND GROUP		LISTEN ADDRESS GROUP	TALK ADDRESS GROUP		SECOND COMMAND GROUP	
HEX		0	1	2	3	4	5	6	7
0	—	—	—	32	48	64	80	96	112
1	GTL	—	LLO	33	49	65	81	97	113
2	—	—	—	34	50	66	82	98	114
3	—	—	—	35	51	67	83	99	115
4	SDC	—	DCL	36	52	68	84	100	116
5	PPC	—	PPU	37	53	69	85	101	117
6	—	—	—	38	54	70	86	102	118
7	—	—	—	39	55	71	87	103	119
8	GET	—	SPE	40	56	72	88	104	120
9	TCT	—	SPD	41	57	73	89	105	121
A	—	—	—	42	58	74	90	106	122
B	—	—	—	43	59	75	91	107	123
C	—	—	—	44	60	76	92	108	124
D	—	—	—	45	61	77	93	109	125
E	—	—	—	46	62	78	94	110	126
F	—	—	—	47	UNL	79	UNT	111	127

low voltage on this line indicates that command or address code characters are being sent down the bus. Thus, the two cases in Figure 8 show the same bit pattern, 00111111, to be sent down the bus. In the first case, it is the GPIB unlisten command, or UNL; in the second, it is the ASCII question-mark character.

Table 3 summarizes all GPIB command and address code characters. Its arrangement parallels that of Table 1. Thus, the character located in row one, column two of Table 3 represents GPIB listen address 33, with a hexadecimal code value of 21. The actual bit pattern—00100001—that represents listen address 33 also represents the exclamation point in ASCII code. The GPIB listener interface distinguishes listen address 33 from the ASCII exclamation point by the voltage state of the ATN line.

Figure 9 returns to the example of a series of eight-bit code bytes to be sent down the bus. Unlike Figure 7, it emphasizes the condition of the ATN line during the code transfer. The same sequence of bytes is to be sent down the bus, but the first three bytes—01011111, 00111111, and 00100101—have different meanings. In Figure 9, ATN is low while the first three bytes are sent. As a result, the three-bit patterns are not interpreted by the listener as the ASCII space, question mark, and percent, respectively. Instead, they are accepted by the listener interface as two GPIB commands, untalk and unlisten, and a single GPIB address, listen address 37.

Extended talk/listen function. The necessity for a GPIB talker and a GPIB listener is apparent; the IEEE-488 standard extends the talk/listen concept still further. At the discretion of the manufacturer, GPIB-compatible devices can have a primary and secondary address structure. The primary addresses are byte patterns assigned to the instrument and indicate whether the instrument is to talk or listen. For example, if an instrument is assigned

GPIB listen address 37, the device becomes a GPIB listener when its data bus receives 00100101 while ATN is low. Similarly, the same instrument becomes the GPIB talker if 01000101, or talk address 69, is received while ATN is low. The secondary address provides a second byte of address code, to be sent after the primary address has been transmitted over the bus. This secondary address provides the device with additional information it might need to perform its function. If the instrument is multi-functional, its operational mode might be defined by the secondary address.

The series of bytes shown in Figure 10 includes a secondary address character. This example suggests that the message portion of the code to be sent to a GPIB instrument must be preceded by a series of GPIB command and

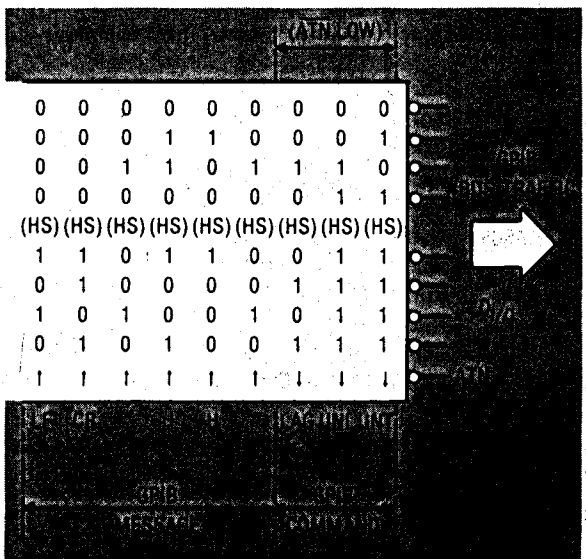


Figure 9. Example of GPIB command and address code.

address code characters. The first command characters to be sent are the universal UNT and UNL commands. These two commands, as well as those located in column one of Table 3, are recognized by all GPIB-compatible instruments when ATN is low. The UNT and UNL commands are sent down the bus to assure that all bus instruments stop their bus traffic so that the controller can reconfigure the bus. Once the previous talker and all of its listeners have stopped their activity, the new listen address and the new secondary address are placed on the bus. For the specific example shown in Figure 10, the listen address bit pattern is 00100101 and the secondary address pattern is 01100001. These two addresses can be expressed as decimal values (37₁₀ and 97₁₀), as hexadecimal code (25_{hex} and 61_{hex}), or as their corresponding listen address group and secondary command group values (LAG 5 and SCG 1). In any case, the actual bit patterns placed on the bus are 00100101 and 01100001, respectively. Once this is accomplished, listen address 37 and secondary address character 97 are recognized by the assigned GPIB instrument, which is placed in its proper operational mode.

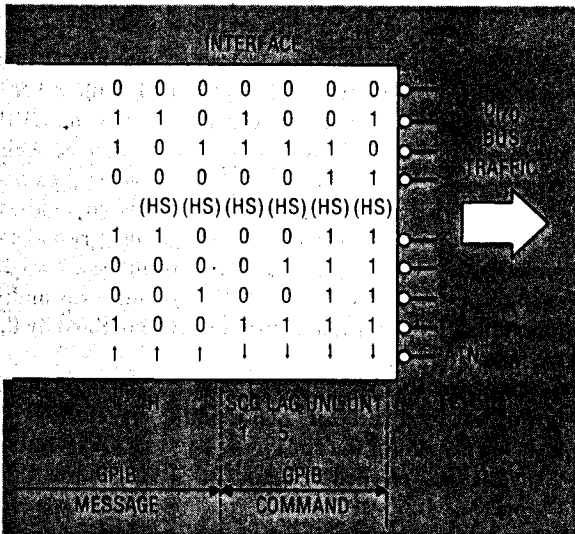


Figure 10. Example of code transfer with a secondary address.

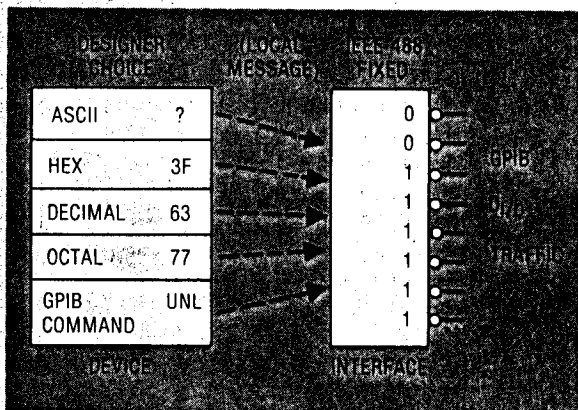


Figure 11. Methods for communicating with the GPIB interface.

Control function. The GPIB provides a method for two or more instruments to listen to another instrument as it sends code down the bus. The usual GPIB system has an instrument that not only acts as a talker or a listener, but can also control bus activities. Such control activities are defined by the controller function in the controller interface to the GPIB.

IEEE-488 implementation

The first decision in instrumentation implementation concerns the acquisition of equipment. The GPIB user can choose a single system, a mixed system, or a prototype system.

The single system is from one vendor—all equipment, including talk and listen devices and cables—comes from the same manufacturer. The single systems save time and intellectual effort, but this speed and ease have a price. The restriction to one vendor limits the user to instruments provided by that vendor, and the user must be content with the characteristics of the available instruments.

The mixed system, with instruments from a variety of manufacturers, is more common. Its advantages are equipment cost and choice of instruments. The user has the freedom to shop for the best instruments at a cost his laboratory can afford, but this freedom has its price. Although the manufacturer of GPIB instruments must build them to be compatible with IEEE-488; there are many ways to meet that standard. All GPIB functions supported by an instrument are not necessarily recognized by another instrument. In fact, similar instruments from various manufacturers are not likely to support all of the same GPIB functions. An instrument is considered IEEE-488-compatible if it can be addressed and can perform the talk/listen handshake operations. Other GPIB functions can be absent or partially supported, according to the guidelines in the IEEE standard. Ultimately, a user dealing with a mixed system must expend more time and effort in establishing initial communication among the instruments.

Another factor in choosing a mixed or single system is the available documentation and manufacturer's support. Although most equipment is reliable and has a satisfactory operational warranty, the single-system vendor generally provides more help in bringing the system on line. In contrast, a GPIB vendor is not likely to offer much help in getting his meter to communicate with another manufacturer's printer. As a result, a physically easy instrument interface might be intellectually difficult. The user should be prepared to spend the time and energy to sort it out.

The prototype system is one in which at least one instrument on the bus is not commercially available. The user might have a commercial controller but be developing his own instrument, which must ultimately be GPIB-compatible. The successful operation of a prototype system requires a great deal of user time, but little initial investment in GPIB parts.

The bit-pattern representation problem. Figure 11, which shows an IEEE-488 interface receiving a byte of

code, illustrates a major GPIB implementation problem: the GPIB standard does not control the way in which the manufacturer communicates with the interface. Thus, the instrument designer knows that the 00111111 bit pattern has a specific IEEE-488 meaning under defined circumstances, but he has several options as to how his instrument receives this byte in the first place. It is clear from the illustration that the ASCII question mark, the hexadecimal code value 3F, the decimal number 63, the octal code value 77, and the GPIB unlisten command can all represent the desired bit pattern. Any of these representations can be typed into the talker to send 00111111 to the IEEE-488 interface. The manufacturer's literature might not make clear exactly which is to be used.

Example implementations. Consider the implementation (with the Tektronix 4051 controller) of the following GPIB code structure:

WBYTE @ 95,63,37,97:34,72,105, - 34.

The WBYTE command effects direct access to the GPIB interface from the keyboard of the 4051. The command includes the decimal equivalent of the GPIB code character to be sent down the bus. The ATN line on the GPIB is brought to zero volts by the @ symbol and returned to five volts by the colon.

Figure 12 illustrates the code structure presented above. The drawing presents the bit patterns for the UNT, UNL, LAG 5, and SCG 1 GPIB command and address characters. The "Hi" message bit pattern is also shown. Note the changes in voltage on the ATN line. The last character, the final quote, is sent down the bus when the EOI line is brought low.

Still another communication complication is the GPIB address group concept. Table 4 presents the four address groups used to classify the various GPIB addressing situations. The four are:

- DAG, the device address group;
- LAG, the listen address group;
- TAG, the talk address group; and
- SCG, the secondary command group.

The DAG, TAG, and LAG assignments are interlocked, while use of the SCG value is an option of the designer. For example, if a GPIB instrument is assigned the 10th device address, that instrument also is given LAG 10 and TAG 10. No SCG need be assigned to the instrument unless the designer desires the extra byte of addressing capability.

Some confusion centers on the decision as to when to employ the assigned DAG, LAG, or TAG. The device address is not used when direct communication via the GPIB interface is desired. In such cases, if the instrument is to function as a listener, the listen address must be used. Likewise, if the instrument is to be configured as the talker, the specific talk address must be used. When direct code access to the bus is not necessary, the controller provides software that can be used to configure the bus. These manufacturer-defined commands use the DAG and, when necessary, the SCG to communicate with the instruments in the system. Figure 13 offers a clarifying ex-

ample for three different controllers. In all three cases, the ubiquitous but innocuous "Hi" message is sent to GPIB device number five. Employment of the device address considerably reduces the user's GPIB communication responsibilities. Comparison of the 4051 example in Figure 13a with the WBYTE statement for the same message in Figure 12 confirms this observation. In the situation of Figure 13, the user need not be concerned with the

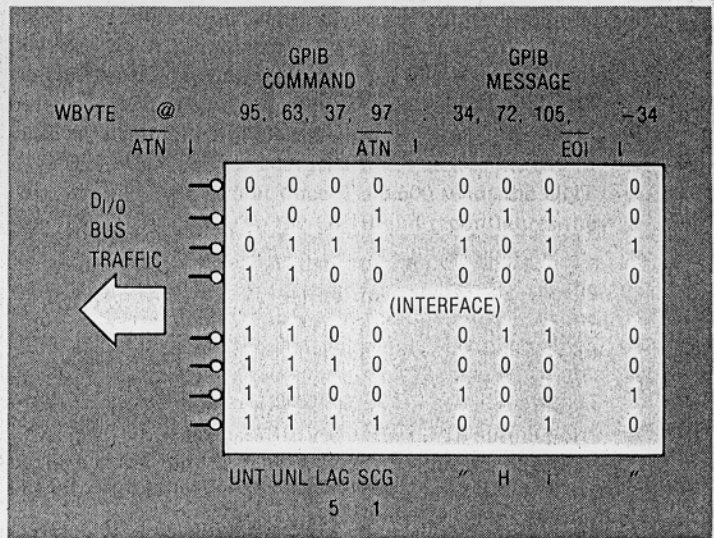


Figure 12. Expanded illustration of WBYTE command.

Table 4. Summary of GPIB address groups.

DAG		LAG		TAG		SCG	
0	1	2	3	4	5	6	7
0	16	0	16	0	16	0	16
1	17	1	17	1	17	1	17
2	18	2	18	2	18	2	18
3	19	3	19	3	19	3	19
4	20	4	20	4	20	4	20
5	21	5	21	5	21	5	21
6	22	6	22	6	22	6	22
7	23	7	23	7	23	7	23
8	24	8	24	8	24	8	24
9	25	9	25	9	25	9	25
10	26	10	26	10	26	10	26
11	27	11	27	11	27	11	27
12	28	12	28	12	28	12	28
13	29	13	29	13	29	13	29
14	30	14	30	14	30	14	30
15	31	15	31	15	31	15	31

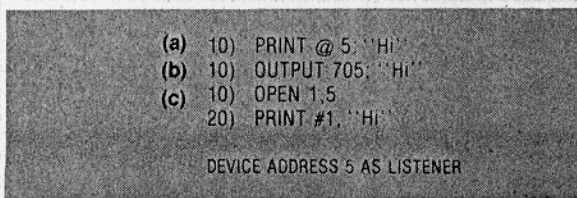


Figure 13. Example of DAG used to configure a bus listener: (a) Tektronix 4051; (b) HP-85; (c) PET.

code, illustrates a major GPIB implementation problem: the GPIB standard does not control the way in which the manufacturer communicates with the interface. Thus, the instrument designer knows that the 00111111 bit pattern has a specific IEEE-488 meaning under defined circumstances, but he has several options as to how his instrument receives this byte in the first place. It is clear from the illustration that the ASCII question mark, the hexadecimal code value 3F, the decimal number 63, the octal code value 77, and the GPIB unlisten command can all represent the desired bit pattern. Any of these representations can be typed into the talker to send 00111111 to the IEEE-488 interface. The manufacturer's literature might not make clear exactly which is to be used.

Example implementations. Consider the implementation (with the Tektronix 4051 controller) of the following GPIB code structure:

WBYTE @95,63,37,97:34,72,105,-34.

The WBYTE command effects direct access to the GPIB interface from the keyboard of the 4051. The command includes the decimal equivalent of the GPIB code character to be sent down the bus. The ATN line on the GPIB is brought to zero volts by the @ symbol and returned to five volts by the colon.

Figure 12 illustrates the code structure presented above. The drawing presents the bit patterns for the UNT, UNL, LAG 5, and SCG 1 GPIB command and address characters. The "Hi" message bit pattern is also shown. Note the changes in voltage on the ATN line. The last character, the final quote, is sent down the bus when the EOI line is brought low.

Still another communication complication is the GPIB address group concept. Table 4 presents the four address groups used to classify the various GPIB addressing situations. The four are:

- DAG, the device address group;
- LAG, the listen address group;
- TAG, the talk address group; and
- SCG, the secondary command group.

The DAG, TAG, and LAG assignments are interlocked, while use of the SCG value is an option of the designer. For example, if a GPIB instrument is assigned the 10th device address, that instrument also is given LAG 10 and TAG 10. No SCG need be assigned to the instrument unless the designer desires the extra byte of addressing capability.

Some confusion centers on the decision as to when to employ the assigned DAG, LAG, or TAG. The device address is not used when direct communication via the GPIB interface is desired. In such cases, if the instrument is to function as a listener, the listen address must be used. Likewise, if the instrument is to be configured as the talker, the specific talk address must be used. When direct code access to the bus is not necessary, the controller provides software that can be used to configure the bus. These manufacturer-defined commands use the DAG and, when necessary, the SCG to communicate with the instruments in the system. Figure 13 offers a clarifying ex-

ample for three different controllers. In all three cases, the ubiquitous but innocuous "Hi" message is sent to GPIB device number five. Employment of the device address considerably reduces the user's GPIB communication responsibilities. Comparison of the 4051 example in Figure 13a with the WBYTE statement for the same message in Figure 12 confirms this observation. In the situation of Figure 13, the user need not be concerned with the

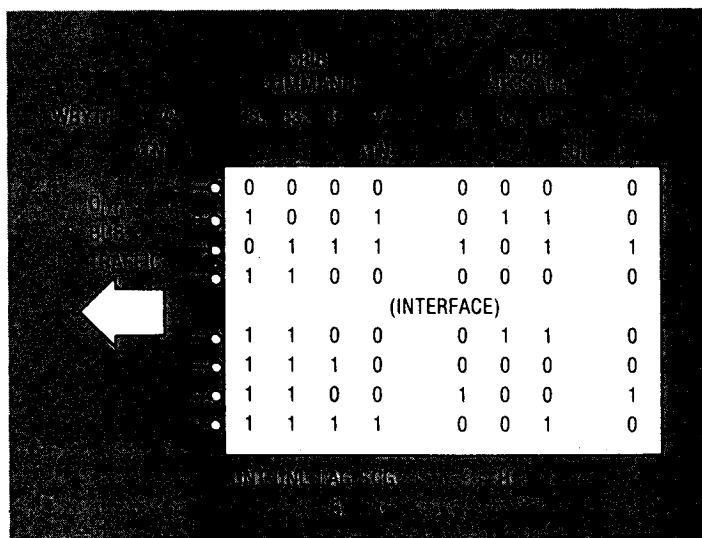


Figure 12. Expanded illustration of WBYTE command.

Table 4.
Summary of GPIB address groups.

DAG		LAG		TAG		SCG	
0	1	2	3	4	5	6	7
0	16	0	16	0	16	0	16
1	17	1	17	1	17	1	17
2	18	2	18	2	18	2	18
3	19	3	19	3	19	3	19
4	20	4	20	4	20	4	20
5	21	5	21	5	21	5	21
6	22	6	22	6	22	6	22
7	23	7	23	7	23	7	23
8	24	8	24	8	24	8	24
9	25	9	25	9	25	9	25
10	26	10	26	10	26	10	26
11	27	11	27	11	27	11	27
12	28	12	28	12	28	12	28
13	29	13	29	13	29	13	29
14	30	14	30	14	30	14	30
15	31	15	31	15	31	15	31

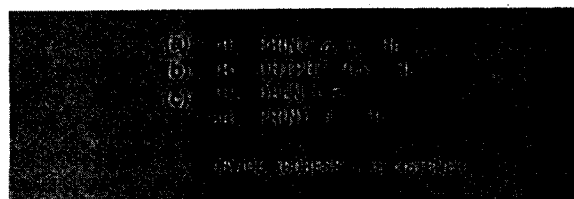


Figure 13. Example of DAG used to configure a bus listener: (a) Tektronix 4051; (b) HP-85; (c) PET.

GPIB commands or specific address structure. The controller still performs these required transactions and directs them to DAG 5 in operations that are transparent to the user. The actual ASCII message to be sent is simplified in Figure 13.

Figure 13b is an example of the use of the HP-85 controller. In this instance, the format is similar to that employed by the 4051. Instead of PRINT, the OUTPUT command is employed; the 7 is added to inform the HP-85 to send the ASCII message to output port number 7. (Port 7 has been designated by HP as their IEEE port. Other communication standards are supported at other HP-85 output ports.) The HP-85 returns ATN to five volts in response to a semicolon; the 4051 uses a colon for this purpose. The HP-85 does not designate a unique character to lower ATN to zero, while the 4051 requires the @ charac-

ter for that task. The HP-85 and 4051 use the same format to send the ASCII message.

Figure 13c presents a PET example. The use of the DAG is a bit more involved, but still uncomplicated. Line 10 of this example is used to establish a dummy file for the device 5 instrument. Once this is accomplished, the print statement in line 20 is used to access file 1 and send the "Hi" message to device 5. The # and common characters control the status of the ATN line.

In each of the examples in Figure 13, the chosen command selects instrument 5 as a GPIB listener. This is done with no further assistance from the user.

The assignment of a GPIB talker by means of the DAG is equally straightforward. Figure 14 represents the analogous examples for this case. In cases (a), (b), and (c), the incoming bus code is to be placed in the A\$ string variable. The 4051 example differs from its counterpart in Figure 13a only in the replacement of the PRINT command with the INPUT command. This change reflects the different direction in which the code is traveling. In Figure 13a, the message leaves the 4051 and arrives at device 5; thus, device 5 was assigned the role of GPIB listener. In Figure 14a, the 4051 is assigning device 5 as the talker and itself as the listener. Once device 5 has been assigned as the GPIB talker, the message code from GPIB instrument 5 is collected by the 4051 and stored in the A\$ variable. ATN is still controlled by the @ and colon characters. Figures 14b and 14c follow a similar line of reasoning. The ENTER command for the HP-85 program and the INPUT statement in the PET program are used to collect the input code string into A\$.

Figure 15 illustrates the implementation of the secondary address in conjunction with the device address. In each example, DAG 5 and SCG 1 are used. A quick glance at Figure 15a shows that for the 4051's controller, the implementation of the secondary address merely requires placing the SCG address of interest behind the DAG value. The comma is a delimiter.

The HP-85 utilization of SCG values is more involved. The SEND 7 portion of line 10 in Figure 15b is necessary for the HP-85 to address its GPIB output port. The characters after the semicolon send GPIB commands to the interfaces of interest. The UNT and UNL commands are sent first. My talk address, or MTA, is sent next. The HP-85 recognizes this command as a signal to put its own interface talk address on the bus. Because the HP-85 controller can also function as a system controller, it requires the MTA. Unlike the 4051, the HP-85 need not always be the active controller. As a consequence of this flexibility, the HP-85's GPIB interface must be addressed each time the HP-85 is to communicate with bus instruments. The MTA is sent automatically when the HP-85 OUTPUT command is used. The last two commands, LISTEN 5 and SCG 1, configure instrument 5 as a listener to perform the function defined by SCG 1. The result of all this command code traffic is a GPIB system configured with the HP-85 interface as the talker and the instrument at listen address 37 as the listener. In addition, device 37 has been instructed to perform its SCG 1 function. This particular function is defined by the manufacturer of the addressed listener device.

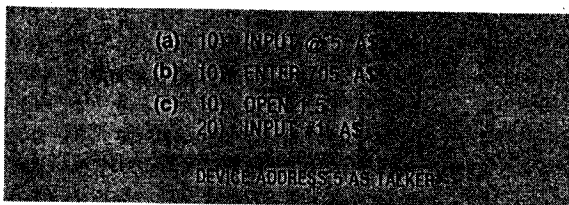


Figure 14. Example of DAG used to configure a bus talker: (a) Tektronix 4051; (b) HP-85; (c) PET. In each case, the tag is entered by the interface.

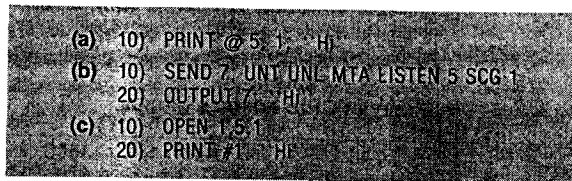


Figure 15. Example of DAG and SCG with (a) Tektronix 4051, (b) HP-85, and (c) PET controllers.

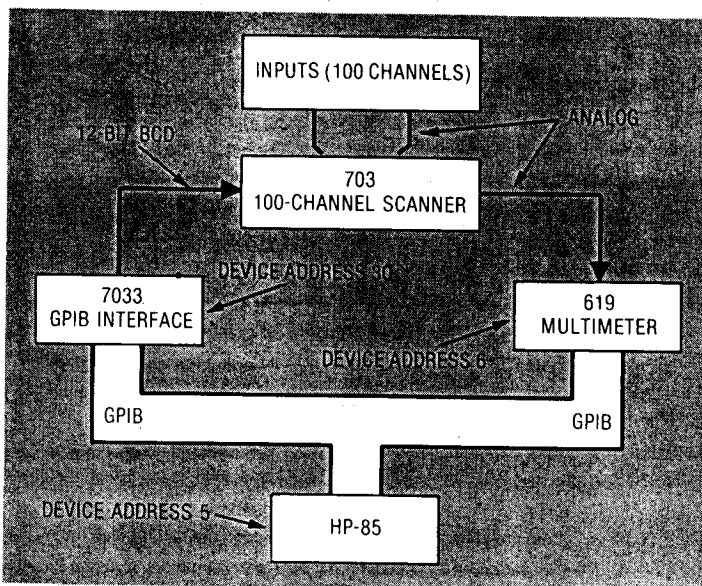


Figure 16. A Keithley data system.

The PET example (Figure 15c) is a logical extension of that in Figure 13c. The introduction of the secondary address is accomplished in line 10 of the PET program in Figure 19c. This extra address code is added to the dummy file structure after the DAG value. Finally, a careful review of Figure 12 will reveal the use of the 4051 WBYTE command structure in the issuance of an SCG value. In Figure 12, SCG 1 is sent. This task is accomplished by placing its decimal value (97) into the command string after the decimal value (37) of the listen address.

IEEE-488 applications

GPIB applications are restricted by instrument-supported data rates, by the number of instruments on the bus, by cable length limitations, and by the imagination of the user. Three examples of general-purpose application are presented below, along with appropriate software. The program sections supplied, however, require user embellishment.

The Keithley data system. The first example, the Keithley data system, is presented in Figure 16; the associated software is in Figure 17. The system uses an HP-85 controller, a Keithley 614 multimeter, a Keithley 703, a 100-channel scanner, and a Keithley 7033 GPIB interface. Figure 16 shows a schematic arrangement of the instruments. The 703 scanner provides a 100-channel analog input port to the data system. The channel is selected by a 12-bit BCD code, and the selected channel response is directed to the 619 multimeter. The multimeter functions and the 7033 interface are controlled by the HP-85. In this example, any of the three GPIB devices—the 7033, the 619, or the HP-85—can perform talk or listen operations. The usual function of the 7033 interface is to accept GPIB instructions from the HP-85 and translate these commands to the 703-channel scanner into BCD code. The 619 meter must accept GPIB commands from the HP-85, comply with these commands by performing the selected operations, accept the analog signal from the data port, and send the appropriate digital reading back to the HP-85.

Figure 17 shows the program components needed to accomplish these tasks. Line 100 provides the formatting information used on incoming code from the 619 multimeter. Lines 220 and 230 provide a way of supplying the desired meter conditions to the HP-85 interface. The actual characters put into the C\$ variable depend on the instrument manufacturer; this information must be obtained from the appropriate instrument manual. This task can be formidable, since many GPIB instrument manufacturers do not clearly specify which characters to use for which instrument functions. Line 210 defines two string variables that provide the program with the X character (E\$) and the quotation mark (D\$). These two ASCII characters are required by both the 7033 and 619 as part of their command structures. For example, if the two-character instruction F1 is to be sent to the 619 meter, then the 619 requires that the X character precede F1 and that all three characters be enclosed in quotation marks. Line 240 is used to assemble the command character string, C\$, in-

cluding the required X character and surrounding quotation marks. The 703 scanner has similar command format requirements. Line 330 accepts the command characters as defined by the 703 instruction manual, and line 340 arranges them with the X character and quotes into the V\$ character string.

The remaining program lines in Figure 17 present the requirements for actual bus communications. Line 500 activates output port 7 on the HP-85 and sequentially sends the UNT, UNL, MTA, LISTEN 6, and SCG 1 GPIB command and address codes to the bus. When line 500 is successfully executed, the GPIB is configured with device 6, the 619 multimeter, as the listener and the HP-85 as the talker. After execution of line 500, the 619 multimeter waits for code to be placed on the GPIB. After line 550, the contents of C\$ have been accepted by the 619. The inclusion of the X character in C\$ causes the 619 to execute the command at once. Line 600 sends the UNT and UNL commands to the GPIB and reconfigures the system with the MTA and listen 30 address commands. This time a new listener, the 7033 interface, is on the GPIB, but the HP-85 remains the bus talker. As with line 550, line 650 sends the correctly structured commands to a GPIB device. The output address in line 650 has a different form from that of line 550. Line 650 uses address 730; line 550 merely uses 7 to access the GPIB output port. (Note the use of a secondary address command in line 500.) The need for SCG 1 to be sent to the 619 meter forced line 550 to be directed to the GPIB output instead of DAG 6 because—unlike the 4051—the HP-85 recognizes only the SEND command as the method to transmit SCG bytes down the bus. Line 600 does not have an SCG because device 30 does not require one. The final program elements, lines 800 and 850 in Figure 17, configure the bus so that the HP-85 is the listener and the multimeter is the talker. Line 850 allows the data code to be brought from device 6 into the HP-85 and stored in the two variables Z\$ and Z in a format defined in line 100.

Figure 18 summarizes the program elements needed if the 4051 is the controller. Line 150 is used to change the significance of certain ASCII characters recognized by the 4051. (Note that 37 is the listen address for the 4051 microprocessor and that SCG 0 is the secondary address

```

100 IMAGE YA, MD, SDE
210 ES = 'X'; DS = CHR $(34)
220 PRINT "INPUT METER CONDITIONS"
230 INPUT DS
240 C$ = DS & 'E' & DS & ES & DS
320 PRINT "INPUT SCANNER CONDITIONS"
330 INPUT VS
340 V$ = VS & 'E' & VS & ES & DS
500 SEND 7, UNT, UNL, MTA, LISTEN 6, SCG 1
550 OUTPUT 7, C$
600 SEND 7, UNT, UNL, MTA, LISTEN 30
650 OUTPUT 730, V$
800 SEND 7, UNT, MTA, TALK 6, SCG 1
850 ENTER 706 USING 100, Z$, Z

```

Figure 17. Key HP-85 commands for Keithley system.

```

150 PRINT @ 32.0 10.255 13
500 PRINT @ 6.0 1.85
600 PRINT @ 30.32 VS
800 INPUT % 6.1 B5
850 PRINT @ 32 USING 100 B5

```

Figure 18. Tektronix 4051 program parts for Keithley system.

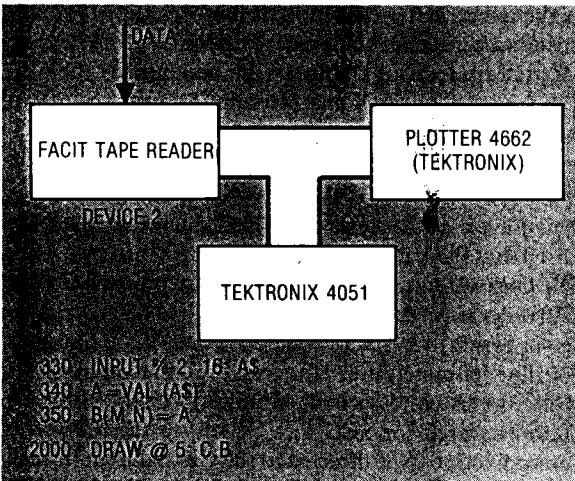


Figure 19. 4051 tape-reader-to-plotter program elements.

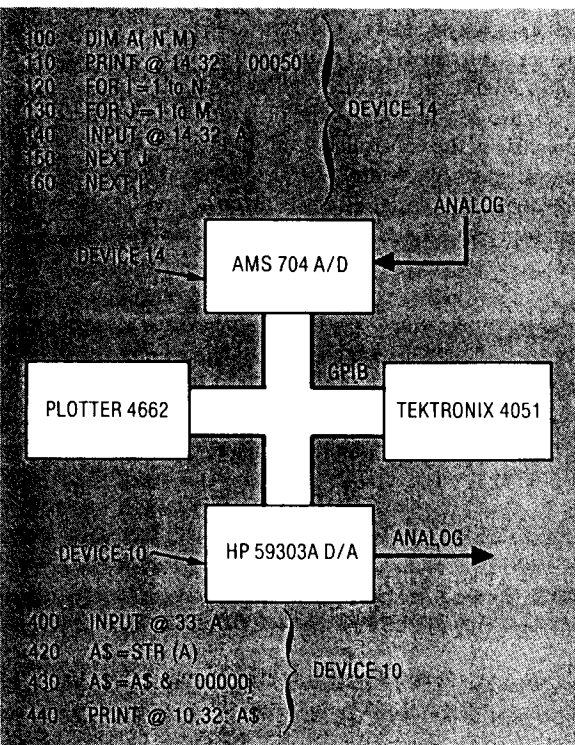


Figure 20. 4051 program elements for an A/D-D/A system.

for a status byte register in that microprocessor.) Lines 500 and 600 are used to send information to devices 6 and 30, respectively. The SCG 32 in line 600 is used to prevent the 4051 from sending a default SCG with the PRINT statement. Lines 800 and 850 address the multimeter and direct the 4051 to accept its data. The percent character in line 800 is used by the 4051 to recognize that some characters—those defined by line 150 in the coming data train—have an alternate meaning.

Transferring code from paper tape to printer. The second GPIB example, an uncluttered GPIB application, is presented in Figure 19, which also lists the 4051 program elements. The task is to transfer code from the paper tape to a printer. In this case, the characters are assigned to the A\$ string variable. Then, the numerical value for each string is assigned to the A variable. The values for each string are finally assigned to a matrix for further manipulation or display. In this example, the tape is only involved in the transmission of ASCII codes that represent digits.

An A/D and D/A system. Figure 20 shows the elements of an A/D and D/A system. Lines 100 through 160 in the 4051 program provide a method of acquiring the data from the AMS 704 and placing it in the variable A. The character string in line 110 is used to select a desired AMS channel. The AMS 704 in this example has been assigned DAG 14, but requires no SCG. A problem might develop, since the 4051 automatically sends SCG 12 with every PRINT command. To assure that no SCG is sent, SCG 32 is inserted to the immediate right of DAG 14 in the PRINT statement in line 110.

The D/A operation is presented in lines 400 to 440. In this case, data is taken from the 4051's internal magnetic tape file, device 33 on the GPIB, and sent to the HP 59303A. Lines 400 and 420 provide a way to take the data from the tape and convert it to an ASCII character string. The data string is added to an HP 59303A command string in line 430. Line 440 is used to send this concatenated string to device 10, the HP 59303A. It also provides the method of sending this data and command string to the outside world as an analog signal. Again, note that SCG 32 has been added to the PRINT statement in line 440. This was done to prevent the 4051 from sending any SCG to the HP 59303A. The result of line 440's execution is a steady DC output voltage from the HP 59303A. This voltage represents the digital data stored in the variable A.

User concerns

The addition of GPIB equipment to the experimenter's arsenal provides an enormous expansion of his ability to inexpensively collect and process data. Automated evaluation of development products, computer control of complex experiments, and development of routine quality control schemes are easily accomplished. However, there are subtle problems with the GPIB system. Before obtaining GPIB equipment, the user must consider the instrument's

- compatibility,
- command of the bus,

- secondary address requirements,
- implementation of the other GPIB functions,
- support literature, and
- data collection rate.

These considerations are briefly discussed below for a simple talk/listen GPIB system.

Function compatibility in the GPIB interface is a primary concern. The instruments to be placed on the GPIB may or may not require a secondary address; this decision is made by the instrument manufacturer. If the instrument of choice has a secondary address, the user must learn what it is and how to put that code into the bus. He must also confirm that his controller can send secondary addresses and must understand exactly how that is accomplished. The user should investigate the extended talk and extended listen capabilities of any GPIB instrument or controller he intends to procure. He should also compile a list of the desired GPIB functions and compare it with the list of functions provided by each instrument under consideration. No manufacturer is required to supply all of the possible GPIB-supported functions, so the user must know which are actually supported by each instrument.

A final compatibility consideration is the possibility of several controllers existing on the same bus. A system controller can transfer bus control to other controllers and then reestablish itself as the bus controller at a later time. The minimal GPIB controller function does not require one controller to allow another to function on the bus. As a result, some controllers (the HP-85, for example) can be configured as system controllers, while others (such as the Tektronix 4051) cannot.

Most manufacturers provide detailed manuals on the operation and maintenance of their equipment, but this practice does not always include the GPIB instructions for the GPIB instruments. The user should examine the GPIB information provided with an instrument *before* the instrument is purchased. If this material is unclear or unavailable, an alternate vendor should be considered. It is very difficult to implement a GPIB instrument if the manufacturer does not supply strong support documents.

The data collection rates accepted by the IEEE-488 standard are so wide that considerable attention must be paid to particular instrument values. The user must be familiar with each instrument's data rate in order to understand the code transfer characteristics of his GPIB system. ■

Bibliography

AMS 704 Manual, AMS, Lake Elmo, Minn.

HP-85 I/O Programming Guide, Hewlett-Packard, Corvallis, Ore., 1981.

IEEE Standard Digital Interface for Programmable Instruments, IEEE, New York, 1978.



Keithley 619 Multimeter Manual, Keithley Instruments, Cleveland, Ohio, 1980.

Tektronix 4051 Reference Manual, Tektronix, Inc., Beaverton, Ore., 1975.



Richard Gilbert is an assistant professor in the chemical/mechanical engineering department of the College of Engineering at the University of South Florida, Tampa, Florida. His research areas include sensor development, computer interface technology, and instrumentation for process control and environmental monitoring. He is also actively interested in application developments for data analysis techniques such as factor and Fourier analysis.

Gilbert's address is Chemical/Mechanical Engineering Dept., University of South Florida, Tampa, FL 33620.

ELECTRICAL ENGINEERS

A POWERFUL SOFTWARE PACKAGE FOR YOU

SHORT CIRCUIT ANALYSIS-E3MB
 FUSE-CIRCUIT BREAKER-WIRE COORDINATION-E4M
 LIGHTING DESIGN-E5M

fully prompted, table printout, cbst analysis options,
 design capacity for high rise and industrial projects

send for our new catalog

Name _____
 Company _____
 Address _____ City _____
 State _____ Zip _____

McClintock Corp.
 P.O. Box 430980, Miami, FL 33143
 (305) 666-1300 Telex: 441582