Designer's Guide To The GPIB/HPIL

by Paul Snigier

The Hewlett-Packard Interface Bus or HPIB was standardized in 1975 into the IEEE - 488 and became the general-purpose interface bus for test and measurement instruments that are used in automatic test equipment (ATE) and for programmable bench instruments and µC-to-peripheral data transfer devices. The bus standard has greatly simplified system design when assembling programmable test and measurement instruments from different manufacturers.

If you are interested in designing test instruments into a benchtop or field environment, consider the GPIB and HPIL to meet your design needs.

Prior to 1975, instruments were manufactured with dedicated interface structures that had unique signal, control and data lines. Data rates, logic levels, and codes also varied. These instruments were often interconnected with cables that had up to 100 wires.

In 1972, when the situation had grown intolerable, an IEEE-488 standards advisory committee met to explore a standard bus. The result, the IEEE-488 Standard - 1975 "Digital Interface for Programmable Instrumentation," was published in April of that year and upgraded slightly in 1978 to clarify textual ambiguities. Minor changes included a 0.4 to 0.5V change for drivers to handle certain new Schottky devices. Others included coded identification I/O markings on instruments to prevent obfuscation.

The HPIB was later adopted by companies other than HP as the General-Purpose Interface Bus (GPIB). The 488 bus standard permits up to 15 devices to operate on the bus, with a maximum 20m distance for interconnecting cables. The standard is divided into mechanical, electrical, functional and operational specifications. These define the type of cable and connectors to be used, the circuits and voltage levels into and between nodes and the interface signals.

Unlike the rigidly defined STD bus, covered in April's *Digital Design* the 488 bus specifications leave some interfaces optional or undefined. As a result, noncompatibility resulting from device-dependent program codes, output data formats and a data coding can lead to problems. If care is taken in reading the product literature the

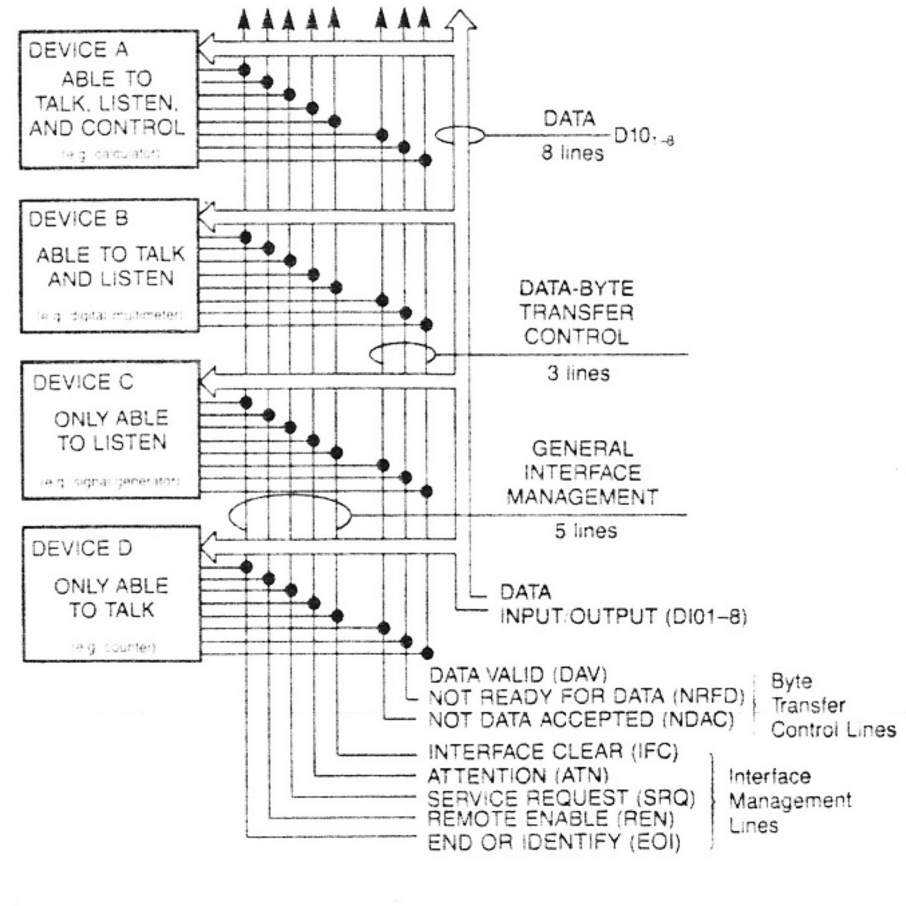


Figure 1: The byte-serial, bit-parallel, negative-logic scheme of TTL-level signals transfer data and commands between instruments or components on 16 lines: (1) data on the DI01–8 lines, (2) handshaking commands on the three data-byte-transfer-control lines, and (3) service requests on the five interface-management lines.

interface problems can be minimized. The standard avoids rigid specification, thus leaving the door open for designers to try new approaches.

After wading through the volumnous 488 document, we must agree with critics who accuse it of being tedious to read. In all features, we do feel that the state diagram notation is more precise and lucid than timing diagrams and does create less ambiguities, compared to other interface standard documentation. Let's now turn to the 488 bus standards in greater detail.

Point-to-point transfer of data over 16 signal lines on the 488 bus is accomplished via devices that are either "talkers" (transmitters), "listeners" (receivers), controllers, or a combination. A talker sends data over the bus, listeners only accept data from the bus, and a controller supervises the others by addressing devices and letting talkers talk. At any time, only one controller or talker can be active. All devices have separate addresses, so each can be uniquely addressed by the controller. Although a controller may address many listeners, data is generally transferred between a speaker and one listener. A printer could be a listener, a waveform generator could be a talker and a digital cassette drive could be both.

All three device types terminate signal lines with 6.2kΩ pulldown and 3kΩ pullup resistors, thereby providing a uniform bus impedance and good noise immunity. Drivers sink 48mA; receivers use Schmitt-type inputs.

The 488 bus byte-serial, bit-parallel, negative-logic scheme has three categories of TTL-level signals. These are eight bidirectional data lines (DIO_{1-8}), three data-byte-transfer-control lines and five general-interfaces-management lines. The eight bidirectional data lines can carry either one byte of either address, data or command.

The three byte-transfer-control lines are "data valid" (DAV), "not ready for data" (NRFD) and "not data accept" (NDAC). When a talker device forces the "data valid" (DAV) line low the device signals to all the listeners that information on the bus is valid and all listeners can read it. A listener that is ready to accept an information byte releases the wired-ored "not ready for data" (NRFD) line, which then floats HIGH. After a listener receives and accepts data bus information, it alerts all other devices on the bus by releasing the wired-ored "not data accept" (NDAC) line. After all listeners release it, the NDAC line goes HIGH and the talker removes its message from the data line. In this way, data transfers proceed along at the pace of the slowest listener, and data is not lost.

The five interface-management lines include: ATN, IFC, SRQ, REN and EOI. The "attention" (ATN) line is pulled LOW by the controller [illegible] that the da[illegible] address [illegible] clear" (IFC [illegible] the contr[illegible] tion. The [illegible] reset line [illegible] tions 100 ms [illegible] If a device [illegible] the "servic[illegible] ORed line [illegible] remotely [illegible] panel con[illegible] able" (REN [illegible] vices go [illegible] the cont[illegible] the listen[illegible] message-[illegible] data I/O [illegible] pulled LOW [illegible] talker [illegible] be pulled [illegible]

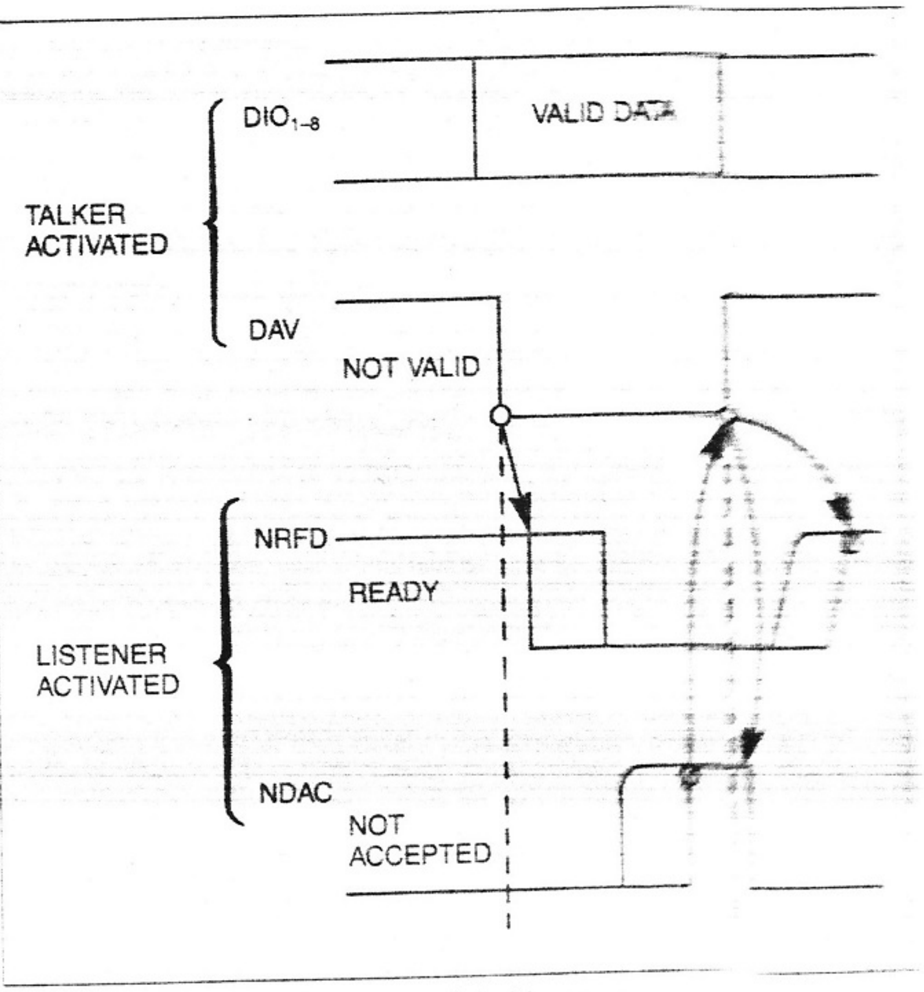


Figure 2: Every byte transfer is accomplished by an a[illegible] shake on the byte-transfer-control lines. Only when [illegible] talker will the listeners read the data on DIO1–8. N[illegible] ready-for-data line LOW. The not-data-accepted line [illegible] cepted, after which the talker removes the data from DIO[illegible] is the slowest one). Once the listener releases NRFD, [illegible] listener is ready for the next data byte.

GPIB Characteristics At A Glance.

1. Interconnected devices: permits a maximum of 15 devices on any contiguous bus.
2. Interconnection path: Linear or star network with a maximum 20m length.
3. Signal lines: 16 active, total; 8 data lines; and 8 for critical control/status messages.
4. Message transfer: scheme is byte-serial, bit-parallel, asynchronous data transfer via interlocked three-wire handshaking.
5. Maximum data rate: 1 Mbyte/s over limited distances with perfect design; otherwise, 250 to 500 kbytes/s over full path.
6. Addressing: primary addressing permits a maximum of one talker and up to 14 listeners (at a time). Primary addresses—31 talk and 31 listen; secondary (two-byte) addresses—961 listen.
7. Control transfer (shift): if there is more than one controller, only one is active (at a time). The presently-active one passes control to one of the others, and only that controller designated as system controller can assume control.
8. Interface circuits: both driver and receiver circuits are TTL-compatible, with the higher power dissipation and higher speeds of this logic family.

asking a device to identify itself after it sent a "service request" (SRQ) to the controller. To prevent the controller from servicing every device that sends a LOW SRQ signal the controller first asks for device identification. When the device receives this EOI (Identify) command, it places its unique address on the bus during the polling sequence. The controller initiates this parallel poll sequence by the EOI signal with the "attention" (ATN) command, which it activates by sending it LOW. ATN can be asserted only by the controller and only during the addressing or command sequence. The controller can poll up to eight devices.

Data Transfer

Data input/output lines (DOI 1–8) resemble µP I/O ports, and carry 8-bit parallel data to and from the CPU. When PRINT, INPUT, GET and similar operations are used the CPU controls the IOD lines which handle transfer control handshake signals and interface management line signals.

The control line logic levels are negative-true: if the level is under 0.8V, the signal is logic ONE; if it is from 2.0V to 5.0V the signal is a logic ZERO. The 0.8V to 2.0V level is undefined. This is the opposite of many µP systems. More about this later in the section on the CPU electrical characteristics.

Handshaking between talkers, listeners and controllers is functionally accomplished in the following manner. Every byte transfer occurs with an accompanying asynchronous three-wire handshake on the byte-transfer-control lines. The talker pulls DAV LOW (once it knows all listeners are listening), so that all devices know that the DOI 1–8 lines carry valid data. Then the fastest talker pulls NRFD low, which lets everyone know that the talker is busy with that byte. Later, the slowest listener lets go of NDAC, which tells the talker everyone got the data. Then this listener removes its DOI 1–8 message and pulls DAV HIGH, which tells the listeners to pull NDAC LOW. The talker and controller then are ready to be "spoon-fed" more data. The cycle then repeats: the talker asserts DAV as soon as it senses NDAC went LOW and NRFD, HIGH. There, in a nutshell, are the simple 488 bus basics.

Functional Partitioning and Messaging

Devices are divided or partitioned into four functional component areas. GPIB bidirectionally interfaces first through drivers and receivers that exchange data through the message coding section, which in turn, transfers its signals to the device through a collection of interface functions: SH, AH, T/TE, L/LE, SR, RL, PP, DC, DT and C. These "interface functions" regulate the bus control circuitry states. Since these interface functions are mutually independent, only one state is active at a given instant. State diagrams specify this set of functions. Obviously the device functions are not covered by the 488 standard. Device functions include modes of operation, capabilities, types of measurement and the like. The 488 drafters wisely left the device functions unspecified because these functions are unique to each instrument.

Of the ten interface functions, any given device probably will be capable of using only a few, although it could handle all of them. Each function about to be described uses one or more bus lines we previously discussed to perform the operation. The "source handshake" (SH) lets the talker talk. All the handshake interdevice transfer originates with SH. The listener returns the "acceptor handshake" (AH), with the slowest listener setting the return rate. The T/TE is used to send device-dependent data. The talker is active only if addressed by one byte while the extended talker requires a two-byte address. The corresponding "listener or extended listener" (L/LE) interface function receives device-dependent data with either a one-byte (listener) or two-byte (extended listener) address. The service request signal is sent to the controller as a single-bit reply in the status byte during a serial poll.

The remaining five are controller-related functions. "Remote local" (RL) enables or disables the manual front panel controls. "Parallel Polling" (PP) permits eight devices to send a status bit over the DIO lines. The "device clear" (DC) sequence clears and initializes devices, so that remote restarting of device operation can be done with the "device trigger" (DT). Finally, the "controller" (C) function can do several things, such as initiate device addresses and transmit universal and addressed commands. More than one controller can be on the bus, but when one device is transmitting commands or addressing devices, the others must be idle.

Messages are local (between device and interface function logic) or remote (between device and bus). If remote, they can cause a state transition in some interface functions or depend upon device(s) for internal control, with remote messages determined by the designer. When two devices try to transmit simultaneously, the active transfer takes priority over the passive one.

With this background behind us, we can proceed to look at a recent extension to the HPIB or GPIB: the HP Interface Loop (HPIL). Although only introduced a few months ago, HPIL already promises to become a de facto standard. Aside from a few differences, it is identical to the GPIB, and the material we have already covered also applies.

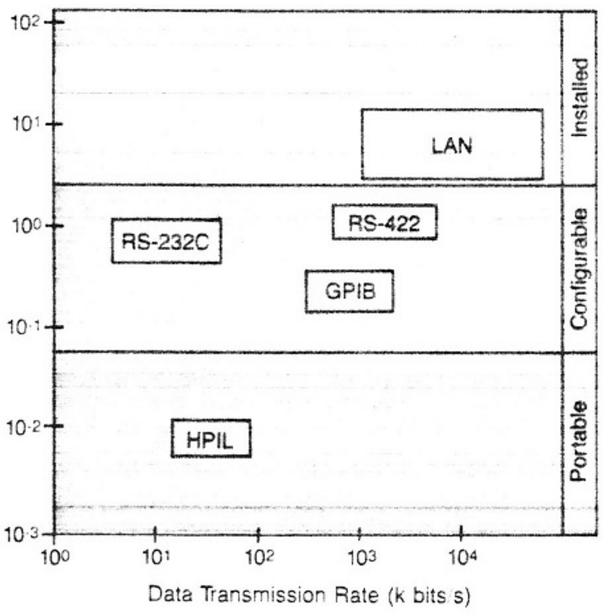


Figure 3: When portability, low cost and battery operation is paramount, but the transmission rate is not, consider the HPIL loop system. GPIB, being byte-wide and TTL compatible, is faster but consumes more power. Though not as easy to assemble as HPIL, GPIB can be assembled to a desired configuration, but with a bit more work. In comparison, local area networks (LAN) are installed, less flexible, consume more power, but are fastest.

Portable Instrumentation

Like the GPIB, the Hewlett-Packard Interface Loop (HPIL) interfaces instruments, controllers, peripherals and the like. Unlike its big brother, however, HPIL is intended for low-power, portable instrumentation.

HPIL is suited to CMOS, battery-powered instruments. HPIL devices transmit data serially in a loop at up to 20 kBytes/sec. HPIL interfaces HP-41, 83 and 85 hand calculators to small cassette drives, microprinter/plotters, small DMM LVDTs and the like built by Hewlett-Packard. HPIL could be hooked up aboard an attack class sub, for example, with an HP-41 running a tape cassette and microprinter/plotter, and be quickly disconnected after the tests. On the other hand, a sophisticated avionics test system will mandate GPIB, as speeds are higher.

HPIL, unlike GPIB, has a remote, power-down control, so the controller can put devices in a standby-state of low power consumption. Unlike GPIB's byte-wide data bus (IOD 1–8), the serial data format helps cut power use. Only one device at a time uses power to transmit to the next device in line. To cut costs, two-wire, differential - drive links cancel out noise and raise SNRs over distances of 100m rather than GPIB's 20m. An added advantage, automatic addressing on HPIL makes programming easier, an area that has caused designers to grumble about the GPIB. Compared to GPIB's linking 15 devices maximum the HPIL can link up to 31 devices, and with extended addressing can address 961 devices, on a low-cost, two-wire cable.

The 82166A convertor provides a general-purpose I/O-to-device interface for interfacing to a noncompatible device. With this converter the engineer can connect several peripherals from different vendors. Under a typical arrangement, an HP-41C could be pre-programmed to take DMM and counter readings, with its commands retransmitted by each device and only acted upon by the addressed device, which also re-transmits the message. In this application the data is transmitted at a slow 5 kbytes/sec through the closed loop and is rechecked for errors by the HP-41C. Typically, the HP-41C could command a thermal printer/plotter to print out data.

With a special interface card (costing above the $125 base price for the HPIL module and calibre), the HP-41 will interface to the HP-83 and HP-85 and 87 desktop computers. Thus the HP-41 can be programmed to collect data in the field or bench and later transfer this data into the HP-85 for analysis and storage. This new data can then be downloaded later into the HP-41 for later use. A more complicated hookup would allow several controllers, with the first taking control upon powerup and others taking over according to the application-specific protocols. Protocols permit a higher-priority-but-idle controller to interrupt another active one.

Code and Interfacing

Since HPIL uses two-wire lines and is asynchronous, a self-clocking three-level code is used. HPIL uses four control bits: 1,0, 1S and 0S— the last two used for reference synchronization at the start of each frame.

The three levels are ±1.5V and 0V. Logic ONE is a positive and negative pulses; a ZERO, just the reverse (negative and zero pulses). Pulse width is about 1μs, with each bit sequence followed by a delay (0V) of 2μs or more. In this way, a spike or transient coming in is overlooked. The sync bit 1S or ONE-S is just a ONE repeated twice with no inter-sequence delay, that is the trailing edge of the first ONE, which is rising to 0V simply becomes the positive-going leading edge of the next ONE. The 0S or ZERO-S, likewise, is two zeros without any inter-sequence gap. This is a reliable coding scheme, although lower in bit density than two-level codes (such as NRZI, Delta Distance, F2F and the like).

Transmission time is as follows: a sync bit takes 6μs; a frame bit, 4μs. A complete frame of 11 bits, one data byte, takes 46μs (10 times 4 plus 6). Ideally, a 21.7 kbytes/sec loop data-transmission rate could be approached; realistically, existing devices are between 3.0 and 5.0 kbytes/sec.

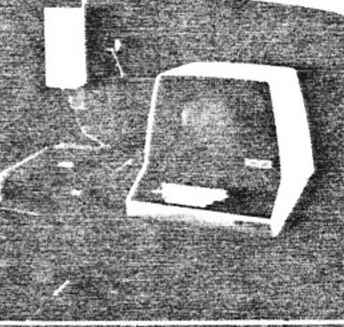
The talker will not send a frame until it gets back and checks the previous one. To avoid transmission delays, each device copies the command frame and passes the frame on, thus permitting parallel processing.

Handshaking and Messages

The talker transmits its message one 11-bit frame at a time. The talker refuses to send the next frame until it gets back the previous one and does an error-correction check on it.

To avoid this unacceptable slowness, each device copies the command frame and passes the frame on, thus permitting parallel processing. After the controller gets back this command frame, it then will not transmit the next command frame until it transmits a "ready for command" (RFC) message and receives it back. The RFC signal is passed along from device to device in the loop—but only after each device has handled the previous command frame.

Updated information is available on the 488 Bus Standard from IEEE at a cost of $10 ($9 for IEEE members), plus $2 for shipping. Write to: IEEE Service Center, 445 Hoes Ln., Piscataway, NJ 08854. □



LOW COST IMAGE PROCESSING

(EyeCom II) Measure variations in image brightness, separate image features, enhance edges and contrast, measure areas of interest. System compatible with existing PDP-II or LSI-II computers.

REAL TIME IMAGE PROCESSING

(EyeCom III) Real time arithmetic processing, spatial filtering, histograms. B/W, pseudo or true color display. Operations 30 times/second. Complete flexibility under software control.

TOTAL IMAGE PROCESSING

Spatial Data Systems, the pioneer in image processing systems and software, has the answers you need for any type of video processing. Whether you're in basic research or real time image processing—if you're thinking about low cost, full capability image processing—think Spatial Data.

FROM **SD Spatial Data**

LogE/Spatial Data Systems, Inc.

508 So. Fairview Ave., P.O. Box 978, Goleta, CA 93116 · TEL: (805) 967-2383 TWX 910-334-3462

Write 28 on Reader Inquiry Card