

## Technical Notes: GPIB

The GPIB (general purpose interface bus) was specifically designed to connect computers, peripherals and laboratory instruments so that data and control information could pass between them. It is also known as IEEE-488 or HP-IB, and is electrically equivalent to IEC-625 bus. It is defined completely in the IEEE standard 488.1-1987 Standard Digital Interface for Programmable Instrumentation.

To use the GPIB you need a GPIB adaptor card in your computer and a GPIB lead. Fourteen devices can be connected to one GPIB and data can be transferred at up to 200000 bytes per second. However, devices shouldn't be more than a couple of metres from the computer.

A GPIB adaptor from Biodata comes with driver software and interfaces to popular programming languages. Biodata also supply ready-to-run applications for data acquisition over the GPIB, where no programming is required and no understanding of the GPIB is necessary.

Technically, the GPIB uses a 16 line parallel connection which has strictly defined mechanical and electrical properties. The 16 lines are divided into 8 data lines, 3 handshake lines to synchronise the transfer and 5 management lines to control use of the bus.

At any time there must be one device on the bus which is the controller. This device issues commands to other devices, and in our systems is always the computer. Other devices may be Talkers - putting data onto the bus, Listeners - reading from the bus, or inactive - neither talking nor listening. Only 1 device may talk at once, but more than 1 may listen to the Talker.

## Bus History

The IEEE-488 bus was developed to connect and control programmable instruments, and to provide a standard interface for communication between instruments from different sources. Hewlett-Packard originally developed the interfacing technique, and called it HP-IB. The interface quickly gained popularity in the computer industry. Because the interface was so versatile, the IEEE committee renamed it GPIB (General Purpose Interface Bus).

## IEEE-488 Overview

Almost any instrument can be used with the IEEE-488 specification, because it says nothing about the function of the instrument itself, or about the form of the instrument's data. Instead the specification defines a separate component, the interface, that can be added to the instrument. The signals passing into the interface from the IEEE-488 bus and from the instrument are defined in the standard. The instrument does not have complete control over the interface. Often the bus controller tells the interface what to do. The Active Controller performs the bus control functions for all the bus instruments.

## System Controller and Active Controller

At power-up time, the IEEE-488 interface that is programmed to be the System Controller becomes the Active Controller in charge. The System Controller has several unique capabilities including the ability to send Interface Clear (IFC) and Remote Enable (REN) commands. IFC clears all device interfaces and returns control to the System Controller. REN allows devices to respond to bus data once they are addressed to listen. The System Controller may optionally Pass Control to another controller, which then becomes Active Controller.

## Listeners, Talkers and Controllers

There are 3 types of devices that can be connected to the IEEE-488 bus (Listeners, Talkers, and Controllers). Some devices include more than one of these functions. The standard allows a maximum of 15 devices to be connected on the same bus. A minimum system consists of one Controller and one Talker or Listener device (i.e., an HP 700 with an IEEE-488 interface and a voltmeter).

It is possible to have several Controllers on the bus but only one may be active at any given time. The Active Controller may pass control to another controller which in turn can pass it back or on to another controller. A Listener is a device that can receive data from the bus when instructed by the controller and a Talker transmits data on to the bus when instructed. The Controller can set up a talker and a group of listeners so that it is possible to send data between groups of devices as well.

### Interface Signals

The IEEE-488 interface system consists of 16 signal lines and 8 ground lines. The 16 signal lines are divided into 3 groups (8 data lines, 3 handshake lines, and 5 interface management lines).

### Data Lines

The lines DIO1 through DIO8 are used to transfer addresses, control information and data. The formats for addresses and control bytes are defined by the IEEE 488 standard. Data formats are undefined and may be ASCII (with or without parity) or binary. DIO1 is the Least Significant Bit (note that this will correspond to bit 0 on most computers).

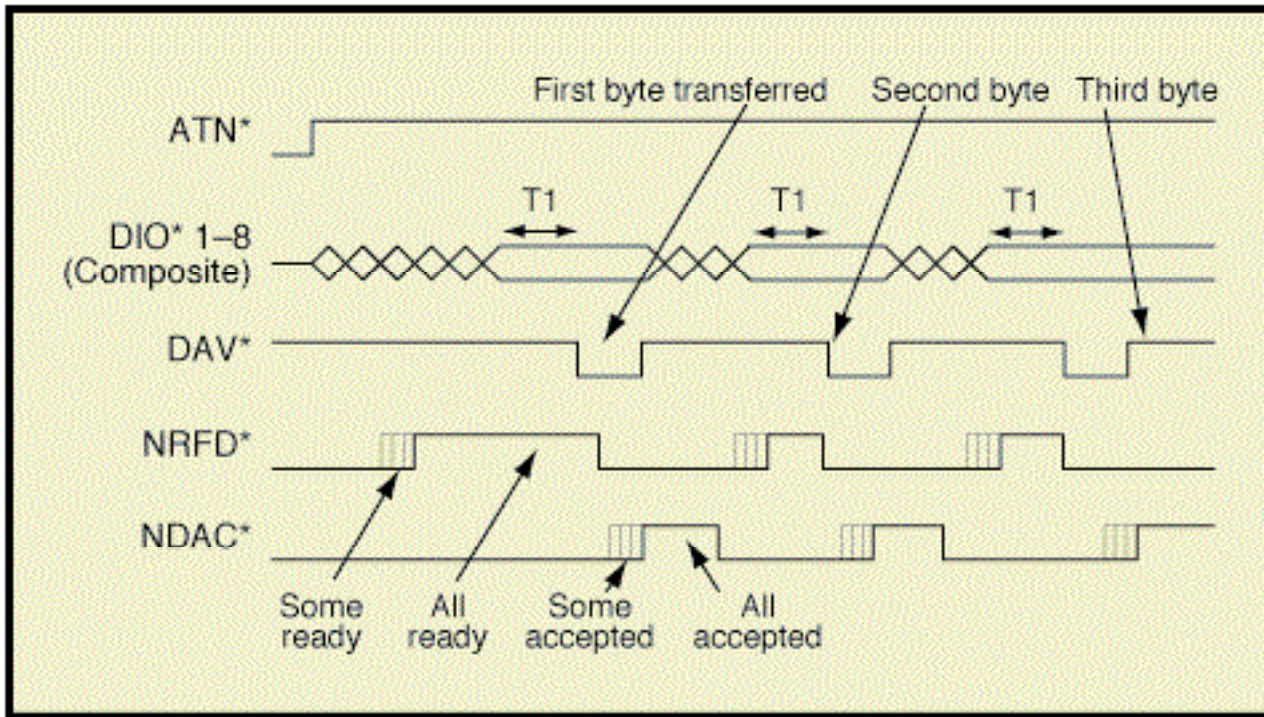


Fig. 2 IEEE 488.1 Handshake

#### Handshake Lines

The three handshake lines (NRFD, NDAC, DAV) control the transfer of message bytes among the devices and form the method for acknowledging the transfer of data. This handshaking process guarantees that the bytes on the data lines are sent and received without any transmission errors and is one of the unique features of the IEEE-488 bus.

The **NRFD** (Not Ready for Data) handshake line is asserted by a Listener to indicate it is **not yet ready for the next data** or control byte. Note that the Controller will not see NRFD released (i.e., ready for data) until all devices have released it.

The NDAC (Not Data Accepted) handshake line is asserted by a Listener to indicate it has **not yet accepted the data** or control byte on the data lines. Note that the Controller will not see NDAC released (i.e., data accepted) until all devices have released it.

The DAV (Data Valid) handshake line is asserted by the Talker to indicate that a **data or control byte has been placed on the data lines** and has had the minimum specified stabilizing time. The byte can now be safely accepted by the devices.

#### Handshaking

The handshaking process is outlined as follows. **When the Controller or a Talker wishes to transmit data on the bus, it sets the DAV line high (data not valid), and checks to see that the NRFD and NDAC lines are both low, and then it puts the data on the data lines.**

**When all the devices that can receive the data are ready, each releases its NRFD (not ready for data) line. When the last receiver releases NRFD, and it goes high, the Controller or Talker takes DAV low indicating that valid data is now on the bus.**

**In response each receiver takes NRFD low again to indicate it is busy and releases NDAC (not data accepted) when it has received the data. When the last receiver has accepted the data, NDAC will go high and the Controller or Talker can set DAV high again to transmit the next byte of data.**

Note that if after setting the DAV line high, the Controller or Talker senses that both NRFD and NDAC are high, an error will occur. Also if any device fails to perform its part of the handshake and releases either NDAC or NRFD, data cannot be transmitted over the bus. Eventually a timeout error will be generated.

The speed of the data transfer is controlled by the response of the slowest device on the bus, for this reason it is difficult to estimate data transfer rates on the IEEE-488 bus as they are always device dependent.

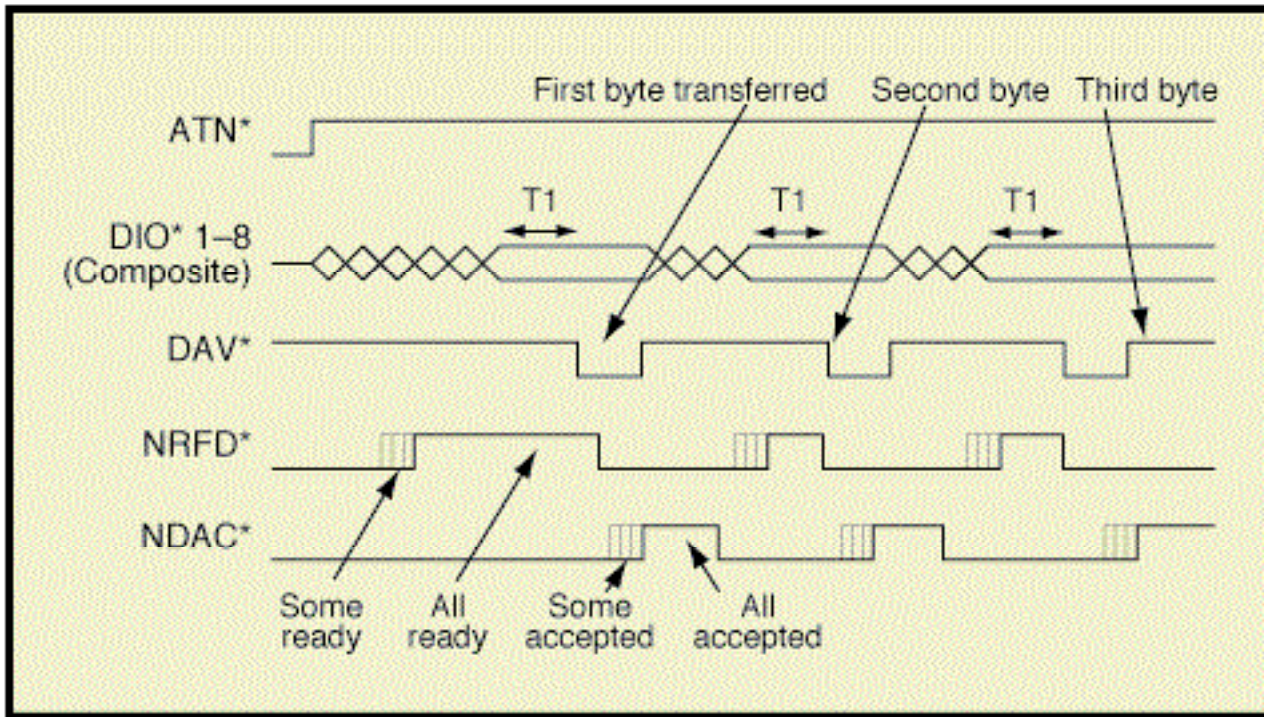


Fig. 2 IEEE 488.1 Handshake

#### Interface Management Lines

The five interface management lines (ATN, EOI, IFC, REN, SRQ) manage the flow of control and data bytes across the interface.

The **ATN (Attention)** signal is asserted by the Controller to indicate that it is placing an address or control byte on the data bus. ATN is released to allow the assigned Talker to place status or data on the data bus. The Controller regains control by reasserting ATN; this is normally done synchronously with the handshake to avoid confusion between control and data bytes.

The **EOI (End or Identify)** signal has two uses. A Talker may assert EOI simultaneously with the last byte of data to indicate

end-of-data. The Controller may assert EOI along with ATN to initiate a parallel poll. Although many devices do not use parallel poll, all devices should use EOI to end transfers (many currently available ones do not).

The IFC (Interface Clear) signal is asserted only by the System Controller in order to initialize all device interfaces to a known state. After releasing IFC, the System Controller is the Active Controller.

The REN (Remote Enable) signal is asserted only by the System Controller. Its assertion does not place devices into remote control mode; REN only enables a device to go into remote mode when addressed to listen. When in remote mode, a device should ignore its local front panel controls.

The SRQ (Service Request) line is like an interrupt: it may be asserted by any device to request the Controller to take some action. The Controller must determine which device is asserting SRQ by conducting a serial poll. The requesting device releases SRQ when it is polled.

#### Device Addresses

The IEEE-488 standard allows up to 15 devices to be interconnected on one bus. Each device is assigned a unique primary address, ranging from 0-30, by setting the address switches on the device. A secondary address may also be specified, ranging from 0-30. See the device documentation for more information on how to set the device primary and optional secondary address.

#### Physical Characteristics

You can link devices in either a linear, star or combination configuration using a shielded 24-conductor cable. The standard IEEE-488 cable has both a plug and receptacle connector on both ends. This connector is the Amphenol CHAMP or Cinch Series 57 MICRO RIBBON type. Special adapters and non-standard cables are available for special interconnect applications.

The IEEE-488 bus specifies a maximum total cable length of 20 meters

with no more than 20 devices connected to the bus and at least two-thirds of the devices powered on. A maximum separation of 4 meters between devices and an average separation of 2 meters over the full bus should be followed. Bus extenders and expanders are available to overcome these system limits.

The bus uses standard TTL level negative logic. When NRFD is true for example it is a TTL low level, and when NRFD is false, it is a TTL high level.

#### Summary

The IEEE-488.1 standard greatly simplified the interconnection of programmable instruments by clearly defining mechanical, hardware, and electrical protocol specifications. For the first time, instruments from different manufactures were connected by a standard cable. This standard does not address data formats, status reporting, message exchange protocol, common configuration commands, or device specific commands.

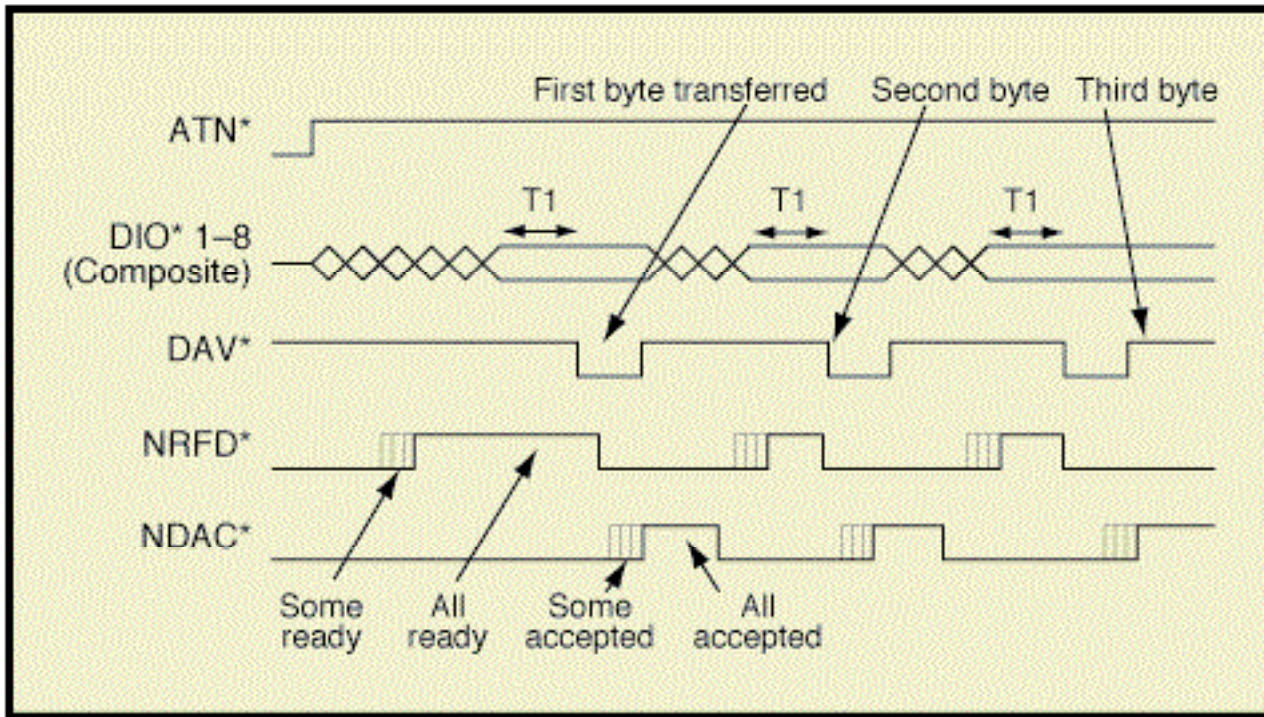


Fig. 2 IEEE 488.1 Handshake

The IEEE-488.2 standard enhances and strengthens the IEEE-488.1 standard by specifying data formats, status reporting, error handling, controller functionality, and common instruments commands. It focuses mainly on the software protocol issues and thus maintains compatibility with the hardware-oriented IEEE-488.1 standard. IEEE-488.2 systems tend to be more compatible and reliable.

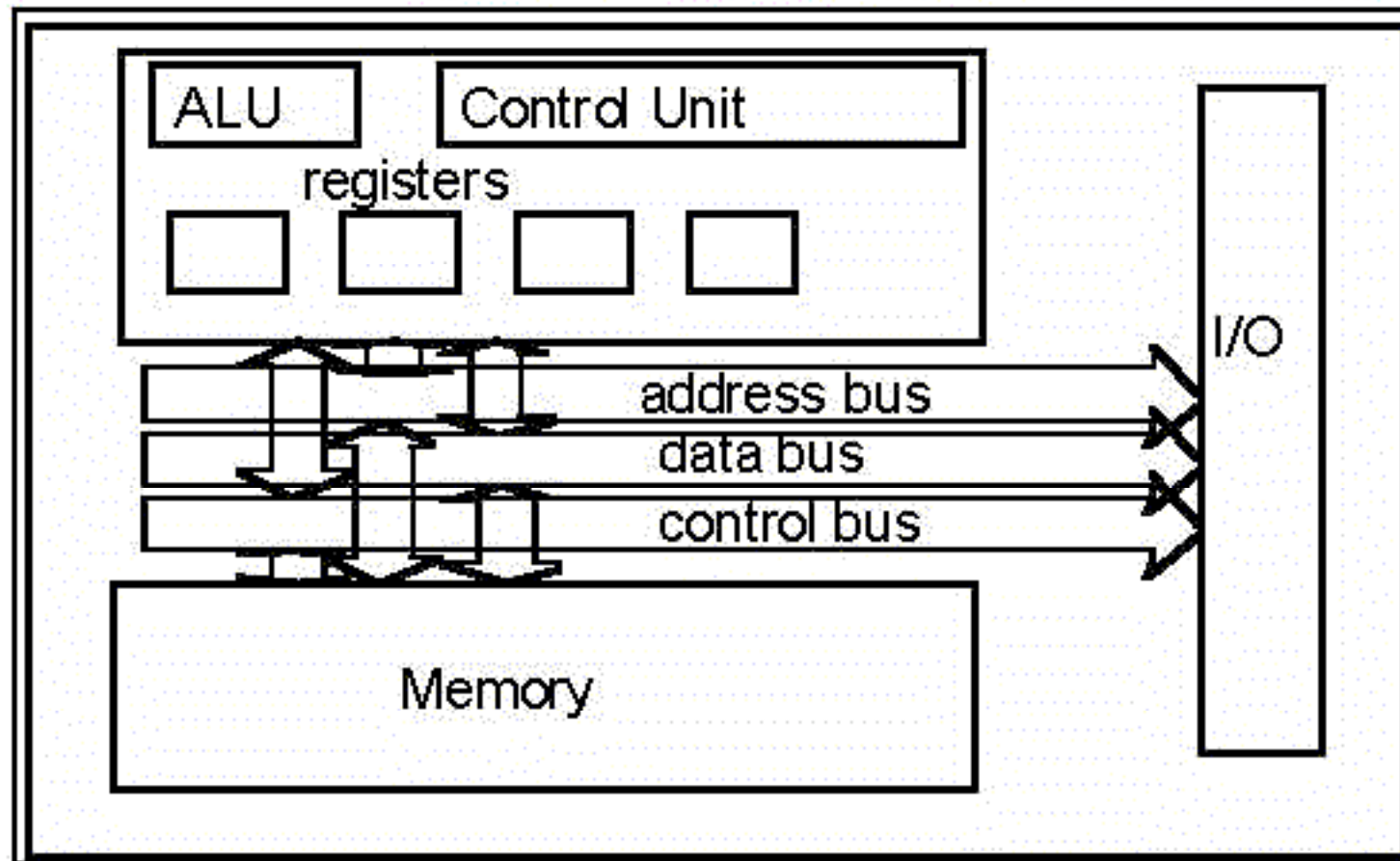
IEEE 488

Standard digital interface for programmable instrumentation  
HP interface

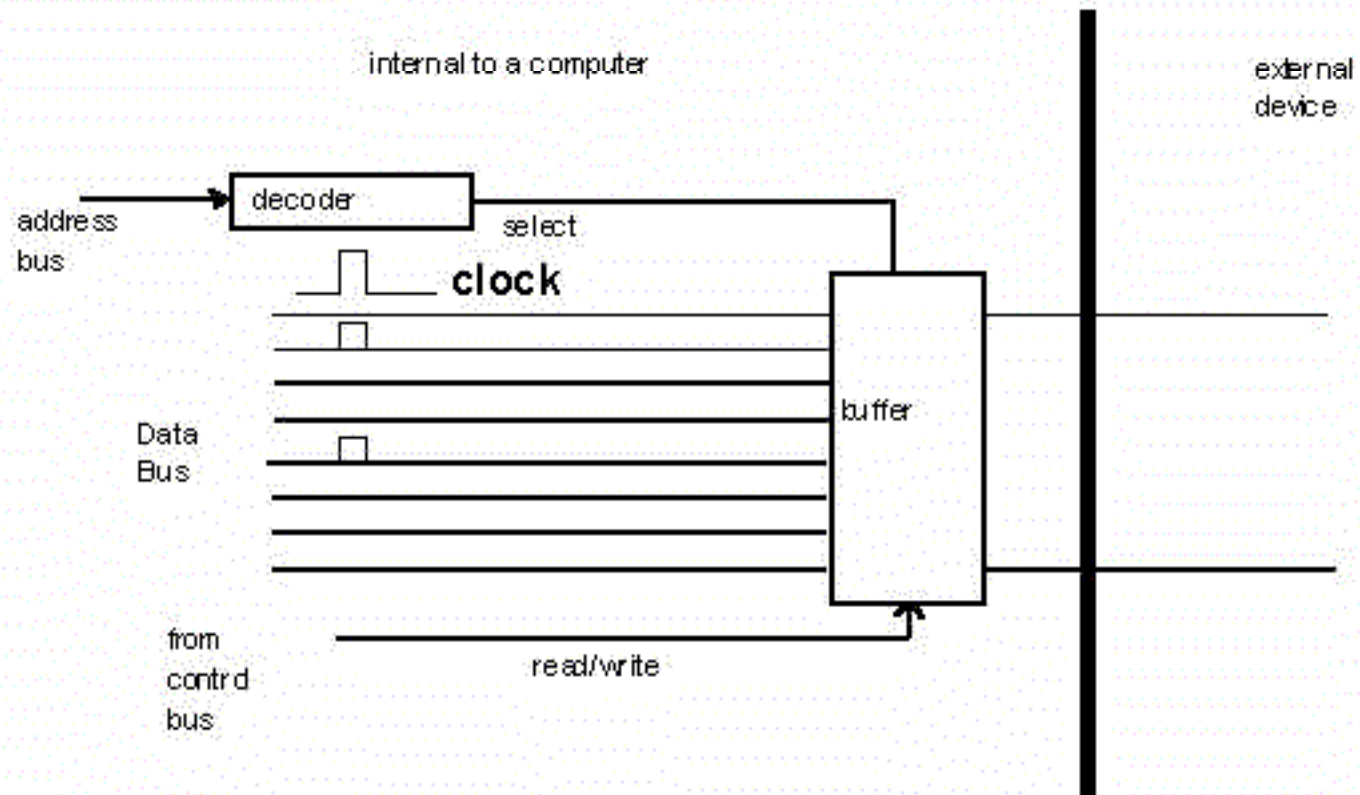
GPIB (General Purpose Interface Bus)

1 mbps

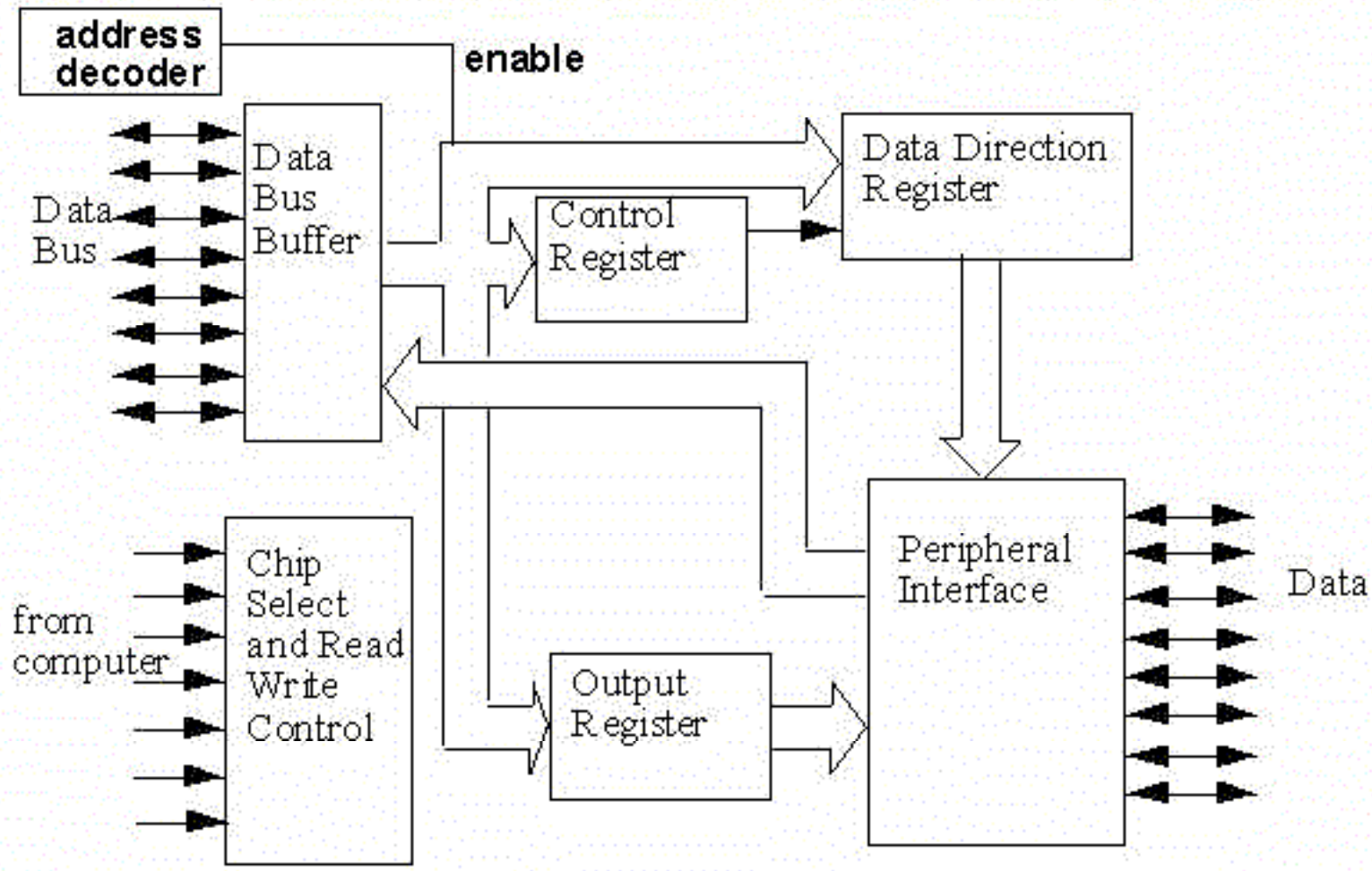
# A COMPUTER



# AN I/O BUFFER



# PARALLEL INTERFACE ADAPTER



# PC Parallel Port

Pin	Description	I/O
1	Strobe -	Out
2	+Data Bit 0	Out
3	+Data Bit 1	Out
4	+Data Bit 2	Out
5	+Data Bit 3	Out
6	+Data Bit 4	Out
7	+Data Bit 5	Out
8	+Data Bit 6	Out
9	+Data Bit 7	Out
10	-Acknowledge	Out
11	+Busy	In
12	+Paper End	In
13	+Select	In
14	-Auto Feed	In
15	-Error	Out
16	-Initialize Printer	In
17	-Select Input	Out
18	-Data Bit 0 returnn (GND)	In
19	-Data Bit 1 returnn (GND)	In
20	-Data Bit 2 returnn (GND)	In
21	-Data Bit 3 returnn (GND)	In
22	-Data Bit 4 returnn (GND)	In
23	-Data Bit 5 returnn (GND)	In
24	-Data Bit 6 returnn (GND)	In
25	-Data Bit 7 returnn (GND)	In

# IEEE 488

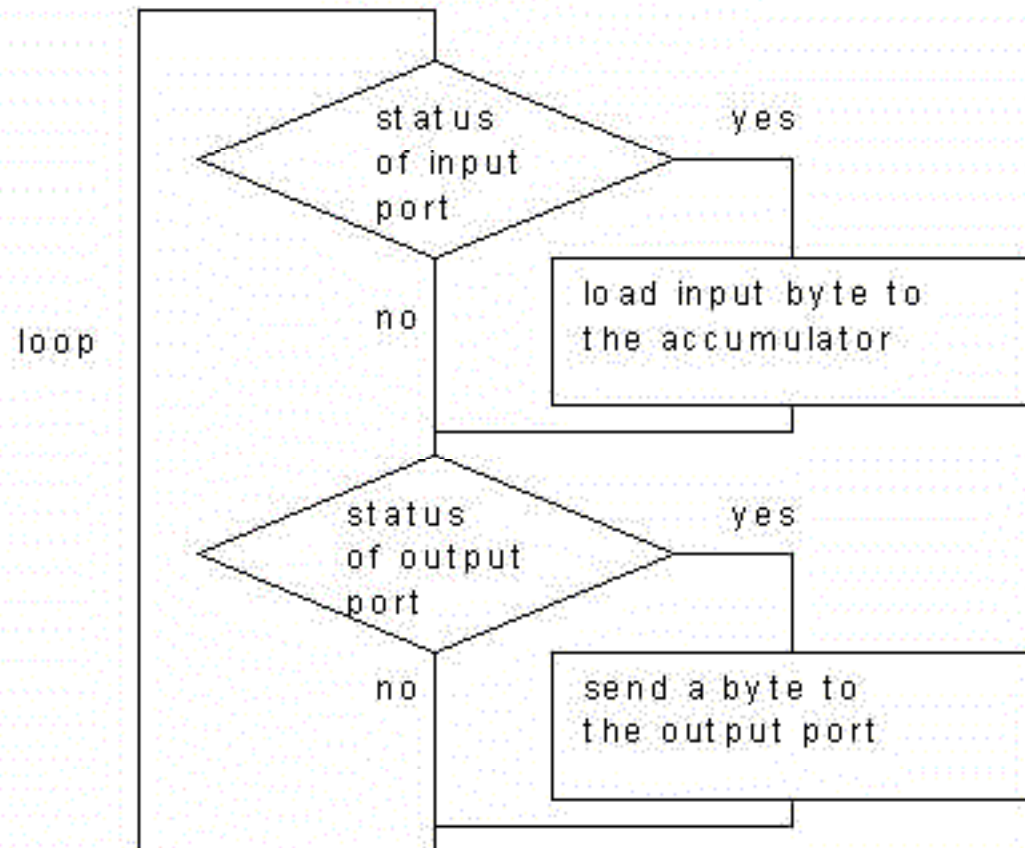
Standard digital interface for programmable instrumentation

HP interface

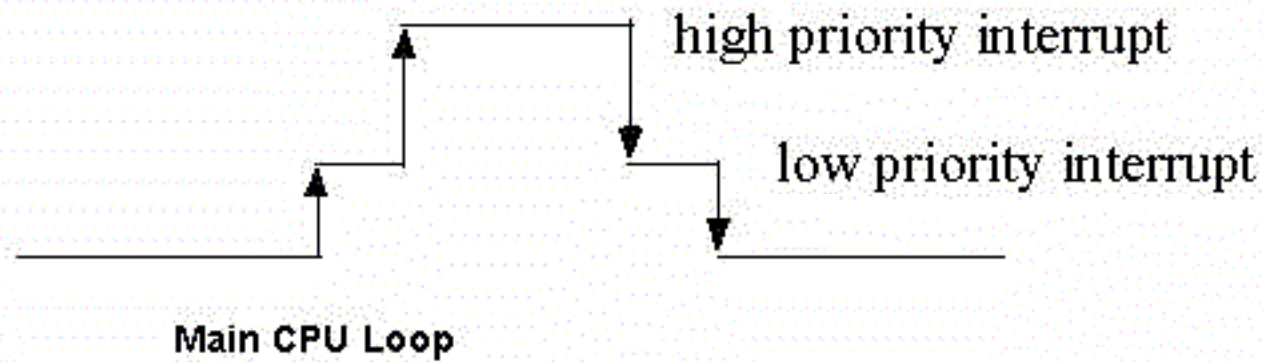
GPIB (General Purpose Interface Bus)

1 mbps

# POLLING



# INTERRUPT

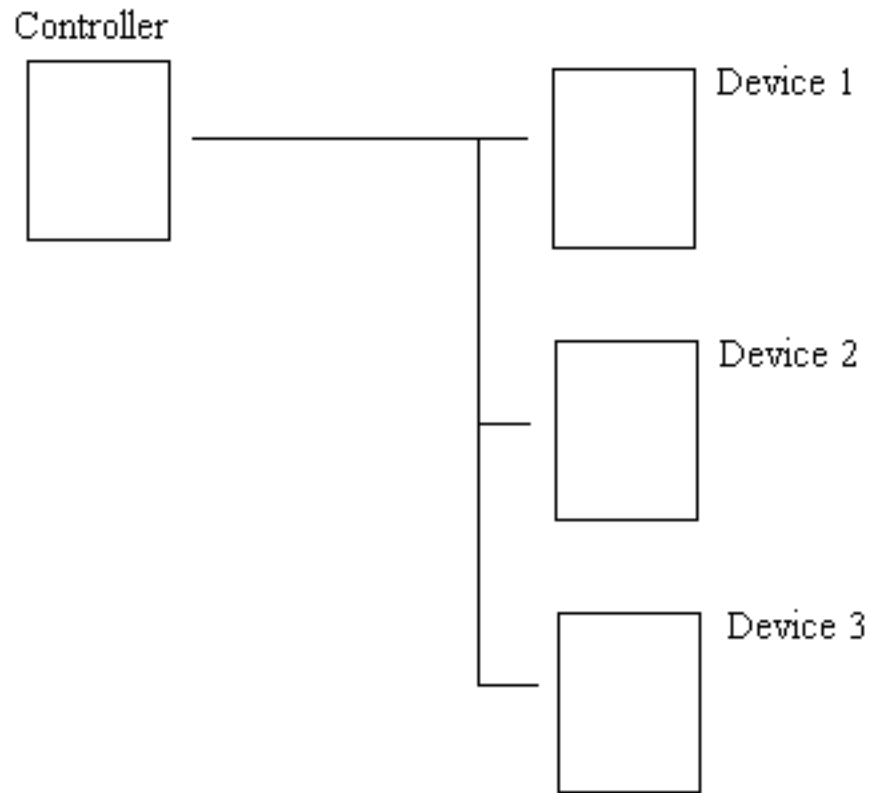


## General

An HP bidirectional parallel printer interface capable of handling 15 devices @ 500 kbps.

The maximum distance = number of devices \* 2 meters with an absolute maximum of 20 meter.

The bus consists of one controller and 1 to 15 devices in a daisy chain fashion shown in the picture below:



An IEEE 488 device has per device 4 different addresses for Command, Listen, Talk and Secondary. Let's say we have a plotter on address 0 then his addresses would be:

- 00 = Command Address
- 20 = LISTEN Address
- 40 = TALK Address
- 60 = Secondary Address



Male            Female

Pin	Signal	Abbr.	Source
1	Data Bit 1	DIO1	Talker
2	Data Bit 2	DIO2	Talker
3	Data Bit 3	DIO3	Talker
4	Data Bit 4	DIO4	Talker
5	End Or Indentity	EOI	Talker/Controller
6	Data Valid	DAV	Controller
7	Not Ready4 Data	NRFD	Listener
8	NoData Accepted	NDAC	Listener
9	Interface Clear	IFC	Controller
10	Service Request	SRQ	Talker
11	Attention	ATN	Controller
12	Shield		-
13	Data Bit 5	DIO5	Talker
14	Data Bit 6	DIO6	Talker
15	Data Bit 7	DIO7	Talker
16	Data Bit 8	DIO8	Talker
17	Remote Enabled	REN	Controller
18	Ground DAV		-
19	Ground NRFD		-
20	Ground NDAC		-
21	Ground IFC		-
22	Ground SRQ		-
23	Ground ATN		-
24	Logical Ground		-

**Technical Description**

The bus is TTL active low (open collector). Every device which asserts a signal low will override the high from another device.

Micro488/p™

Miniature Serial/IEEE 488 Controller

#### Features

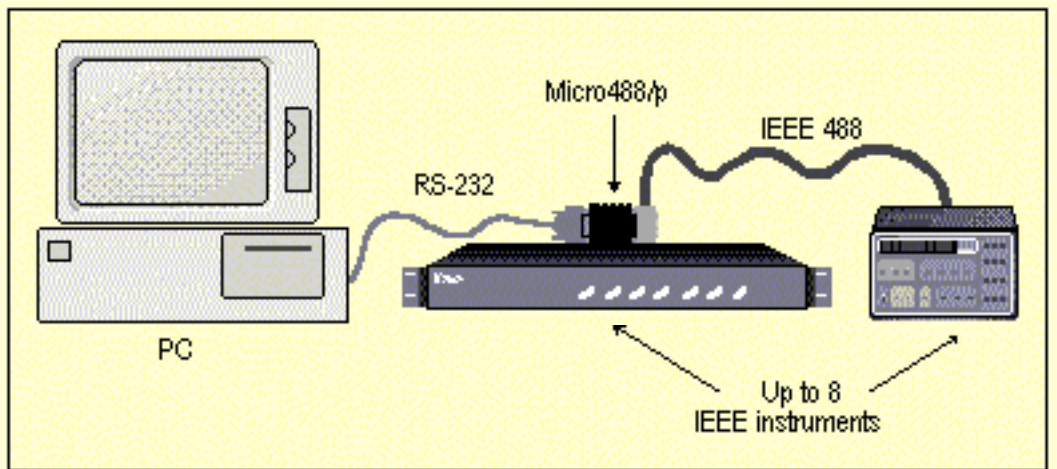


- Smallest serial to IEEE 488 controller available
- Controls up to eight IEEE 488 instruments
- Programmed with concise ASCII commands
- Requires no external power supply—powered from the computer's serial port
- Features standard DB25 serial & IEEE 488 connectors
- Enables remote control of IEEE 488 instruments
- Easily connects to laptop & notebook PCs

Not much larger than a standard IEEE 488 connector, the Micro488/p is the smallest serial to IEEE 488 controller on the market. Nonetheless, it features an on-board microcontroller, permitting it to interpret commands received on its serial port and to control as many as eight IEEE 488 instruments. Moreover, because the Micro488/p derives its power from the host computer's serial port, it requires no external power connection.

Because the Micro488/p's firmware converts commands received on its serial port into IEEE 488 protocol, it does not require that the host computer run special software. The host computer must simply have the ability to send and receive ASCII data via its serial port.

The Micro488/p attaches directly to the IEEE connector on the back of an instrument, eliminating the need for an additional IEEE cable. To control multiple instruments, you need only connect the instruments with standard IEEE cables and attach the Micro488/p to an open connector on any of the cables.



#### Micro488/p Example Program

The following BASIC program excerpt for use with a PC controls a digital multimeter (address 16) and collects and displays a reading.

```
OPEN "COM1:9600,N,8,1"   For random as #1
PRINT#1, "I"             Initialize Micro488/p
PRINT#1, "C;16"         Clear meter
PRINT#1, "0A;16;RANGE 4" Select range 4
PRINT#1, "EN;16"       Request reading
INPUT#1, A$             Put reading into A$
PRINT A$
```

## Command Summary

The host computer sends the commands described below to the Micro488/p via RS-232. The Micro488/p employs RS-232 to send all the responses it receives from the IEEE port to the host computer.

Command	Description
A	(ABORT) regain control of IEEE 488 bus
C	Send DEVICE CLEAR to all devices
C;09	Send SELECTED DEVICE CLEAR to device 09
EC;1	Enable echo mode on serial port
EO;1	Terminate IEEE reads upon receipt of an EOI, and send EOI with the last byte of an IEEE write
EN	Accept data from a device previously addressed to TALK
EN;24	Address device 24 to TALK, and accept data
H;1	Enable RTS and CTS hardware handshake on serial port
I	Initialize Micro488/p to power on conditions
L	Unassert the REMOTE ENABLE bus line
L;14	Send GO TO LOCAL command to device 14
LL	Send LOCAL LOCKOUT
O;data	Send "data" to devices addressed to listen
OA;22;data	Send "data" to device 22
RE	Assert the REMOTE ENABLE bus line
RE;22	Assert the REMOTE ENABLE bus line and address device 22 to listen
RS	Unassert the ATTENTION bus line
SP;30	SERIAL POLL device 30 and send the response to the serial port
SQ	Query the Micro488/p as to the state of the SRQ line; A "Y" is returned on the serial port if asserted, an "N" is returned if the port is not asserted
TB;n	Selects any combination of Carriage Return and Line Feed for bus terminator
TC;n	Selects any combination of Carriage Return and Line Feed for serial terminator
TR	Send GROUP EXECUTE TRIGGER to all devices in the LISTEN mode
TR;17	Send GROUP EXECUTE TRIGGER to device 17
X;1	Enable Xon/Xoff protocol
/A	Assert ATTENTION
/L;04	Address device 04 to listen

/ML Place Micro488/p in LISTEN mode upon receipt of RS command  
/MT Place Micro488/p in TALK mode upon receipt of RS command  
/T;10 Address device 10 to TALK  
/UL Send UNLISTEN  
/UT Send UNTALK  
<Ctrl> Q XON character, restarts serial transmission  
<Ctrl> S XOFF character, halts serial transmission  
<Ctrl> A

Unlock Micro488/p from inappropriate command, and reset input buffer



*The Micro488/p attaches directly to an instrument's IEEE 488 connector*

## Specifications

### Serial Interface

Electrical Characteristics: RS-232

Duplex: Full with echo/no-echo

Data Bits: 8-bit ASCII

Stop Bits: 1 or 2

Parity: none

Baud Rates: 300, 1200, 2400, 4800, 9600 and 19200, set automatically upon the receipt of a carriage return from the host computer

Terminator: On Transmit, software configurable as LF, CR, LF-CR, or CR-LF; On Receive, CR

Handshaking: Supports Clear To Send (CTS), Request To Send (RTS), and XON/XOFF, software configurable

Serial I/O Buffers: 120 character input buffer

Connector: Accepts 25-Pin Sub-D male (DB25P)

### IEEE 488 Interface



*The low-cost 2" x 2.3" Micro488/p enables IEEE 488 control from any computer equipped with a serial port*

Controller Subsets: C1, C2, C3, C4, and C28

Terminator: Programmable for LF, CR, LF-CR, CR-LF, and EOI  
Connector: Plugs directly into a standard IEEE 488 female receptacle,  
either directly on an instrument or on a cable

#### General

Power: Draws less than 5 mA from the DTR or DSR serial lines on the host computer  
Environment: 0° to 50°C; 0 to 90% RH non-condensing  
Dimensions: 50 mm deep x 60 mm wide x 25 mm high (2" x 2.3" x 0.9")  
Weight: 50.2g (1.8 oz)

#### Ordering Information

Description	Part No.
Miniature RS-232 to IEEE controller	Micro488/p
RS-232 cable, including adapter for 25- or 9-pin PC COM port, 6 ft CA-35-6	
Shielded IEEE 488 cable, 6 ft	CA-7-3

Serial488/4™Micro488A™ & Micro488/EX™  
® Copyright 1998, IOtech Inc.

Category: IEEE 488 Interfaces & Support : Support :



## Micro488/p

Not much larger than a standard IEEE 488 connector, the Micro488/p is the smallest serial to IEEE 488 controller on the market. Nonetheless, it features an on-board microcontroller, permitting it to interpret commands received on its serial port and to control as many as eight IEEE 488 instruments. Moreover, because the Micro488/p derives its power from the host computer's serial port, it requires no external power connection. See this product in our online Catalog!

Price: \$ 395.00 / each  
Product Code: Micro488/p

## Understanding The IEEE 488 Controversy

What are the pros and cons of the new HS 488 proposal?

By Dr. Philip J. Fleming, PhD

The recent public and volatile debate over a proposed change to the well used and understood IEEE-488.1 Standard has, unfortunately, provided little in the way of high quality technical information for the engineering audience that actually uses this specification. Contrary to a number of comments seen in recent publications, the roots of this debate go back more than 6 years to at least 1992 and have obviously not been resolved.

### Overview

Good engineering practice will always require a rigorous questioning process at the beginning of application development to determine the benefits and shortcomings involved with any new technology or methodology. In particular, one should ask:

- 1.What are the known features or attributes of this methodology that are important to my application?
- 2.What is different from that which my engineering resources are familiar with?
- 3.What are the risks and/or adverse consequences of implementing my application?
- 4.What benefits will be derived for my products, my customers, my company, and my career from this decision?

Applying these questions to the 488 issue in brief, some of the key features of the existing IEEE-488.1 and of the proposed "HS-488" can be summarized by looking at Table 1:

<u>Feature</u>	<u>Standard IEEE-488.1</u>	<u>HS-488 as Proposed</u>
Bus Structure	16 line, bit parallel, byte serial	Same
Handshake Type	Interlocked	Non-Interlocked in HS Mode
Handshake Lines	3	11,2
Data Lines	8	8
Interface Management Lines	5	5
Interface Logic Complexity	11 State Machines	28 State Machines
New Bus Commands	N/A	CFE (Configuration Enable)
Compatibility with Existing Applications	100% for Applications adhering to IEEE-488.1 / IEC-625-1	Some Yes Some Unknown?
Data Transfer Rate	1 MByte/s max. over full range of cable configurations to 15 meters	8 Mbytes/s max. (Variable with cable length)
Bus Overhead Management Time	Depends on data capture or sampling increments	Same
New Design Constraints	N/A	Critical Cable-length determinations - error source?
Cost of Circuitry (IC or Standard Cell)	<\$10 ea. For stand-alone controller circuits - much less for integrated solutions	Pricing Unavailable Doubling of circuit complexity is a serious cost issue.
Sources of Supply	>5 Known, but shrinking interest	1 (no more on horizon)

- 1. NDAC (Not Data Accepted) line can be used by an acceptor that is unable to buffer more data.**
- 2. NRFD(Not Ready For Data) line , when activated by any acceptor, signals HS controller to switch to standard 488 mode.**

#### Analysis

Having created a list of "features" that might impact our choice of methods for data manipulation on the 488(GPIB) bus, the items with common or similar answers can be set aside while we look at the other

elements.

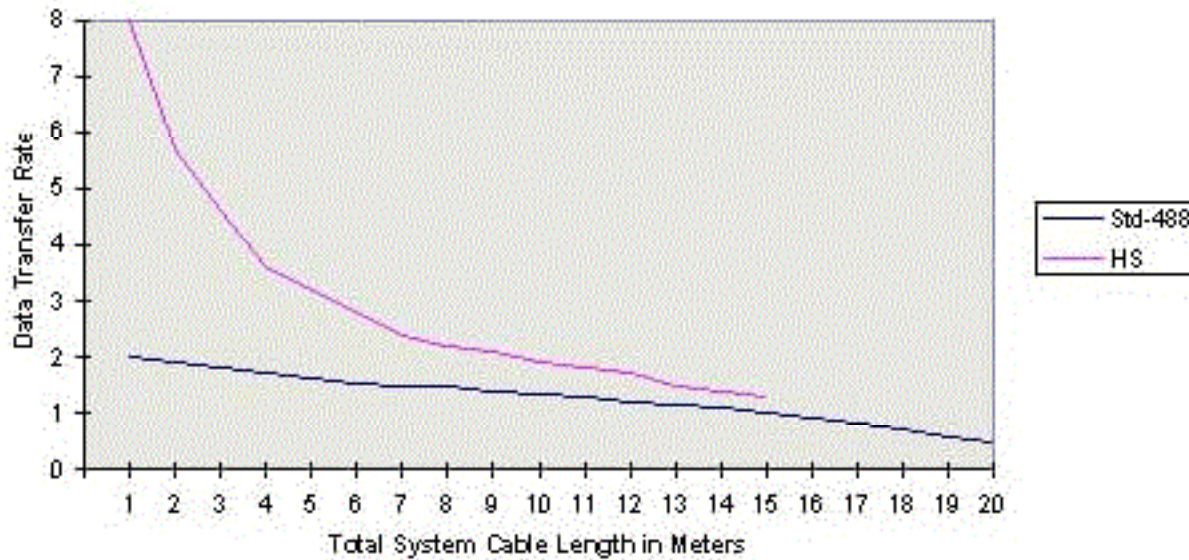
On the positive side of the scale for choosing HS-488 we see that it uses the same GPIB cables and connectors currently in use. The data-streaming methodology is well suited to the transfer of very large amounts of data. The maximum data transfer rate appears to be nearly 8 times faster than existing 488.1 solutions, but we need to examine this further. There is a method for handling non-HS-488 compliant devices connected to the bus, although non-HS compliant devices receive no benefit in performance from an HS controller.

On the negative side, however, three elements from our list show up very prominently as potentially serious problems:

1. The first problem results from "CFE". What is CFE? CFE is a new command (Configuration Enable) introduced to the GPIB bus by an "HS" controller. This command is "unaddressed" which means that it is sent to all devices on the GPIB bus at the same time to inform these devices of the cabling configuration that has been programmed. Reviewing the IEEE-488.1 specification, we find that, for system compatibility reasons, it clearly disallows introduction of a command such as this. Further, it implies that the 488 designer may treat such signals in almost any manner, e.g. error messages to shut down the system. This makes it impossible to guarantee a transparent interface with other 488.1 devices that may be active on the GPIB bus.

2. Cable length is directly related to data transfer speed. From the IEEE-488.1 specification again, we can see that the Settling Time T1 and the Data Valid Time T2 have been set for "worst-case" cable length of a full 15 meters and with 15 active devices. Thus, the 1 MByte/s data transfer rate is "guaranteed" over the full range of cable configurations. The "HS" solution, however, is not so simple to understand. In fact, there are 6 ranges of cable length that are specified and need careful attention. If we extract these two sets of data and plot the results, the following graph gives us some better insight:

**Maximum Data Transfer Rate  
(Mbytes / second)**



**( Note good agreement with results of Pieper, et al. Ref.  
[Http://ourworld.compuserve.com/acea/hs\\_gpib.htm](http://ourworld.compuserve.com/acea/hs_gpib.htm) )**

[BACK TO ARTICLE](#)

It now becomes clear that the eight Mbytes/s advertised for HS-488 becomes something considerably less if you wish to use "HS" with an application that is more complex in terms of cabling runs. If you have only a single node application and less than two meters of cable length, the HS methodology does indeed seem to give better performance. Beyond two meters, there is little performance improvement over the existing 488 solutions.

3. The third element of concern is COST. Since the decoding logic has more than doubled for the HS

solution, it doesn't require a great deal of intellectual effort to realize that the cost of the HS controller is going to be more expensive than existing 488 controllers. Simple rule-of-thumb analysis tells us that a 2X increase in chip area equates to a 4X higher probability of defects. This equates directly to higher cost per working circuit, particularly where functions such as 488 are integrated into much larger and more complex circuits that would find such an increase in silicon real estate (and subsequent cost penalties) totally unacceptable and disastrous to their product developments.

#### Conclusions

The analysis described here is simple and quite fundamental towards understanding the nuances of implementing virtually any technology. Use of such a technique allows engineers and technology managers to see quickly beyond the gossamer strands woven by the marketers and ascertain the true value or risk involved with new technology.

In this case, the technique shows us that the "HS-488" methodology may provide some service to a portion of the 488 domain of applications. However, "HS" value is clearly limited to data-streaming applications with large quantities of data traversing short lengths (less than two meters) of cable. For the vast collection of other applications, which use the 488 bus but do not require data streaming, there is no benefit to be gained from the "HS" methodology at all. In fact, from the adverse consequences noted above, we see that there is very high risk and disadvantage to the overwhelmingly larger segment of 488 applications that require transmission of smaller packets of data, more operating nodes, and longer cable runs. Based on this analysis, we would not recommend adoption of HS-488 as a new IEEE standard. Rather, it should be used for what it does best as a self-sustaining product.

(What's your spin on the IEEE-488.1/HS-488 controversy? Exchange ideas with a host of industry experts by entering the Test and Measurement Online Discussion Forum.)

Philip J. Fleming & Associates, Inc., 4665 Newstead Place, Colorado Springs, CO 80906-4867.  
Phone: 719- 576-5980; Fax: 719-540-2849; e-mail: phil-pjfa@worldnet.att.net.

Edited by Bruce A. Bennett