Digital488 and Digital488/32/OEM

IEEE 488 to Digital I/O Interface

IOtech, Inc.
25971 Cannon Road
Cleveland, OH 44146-1833
Phone: (440) 439-4091
Fax: (440) 439-4093
E-mail (Product Information): sales@iotech.com
E-mail (Technical Support): productsupport@iotech.com
Internet: www.iotech.com

Digital488 and Digital488/32/OEM

IEEE488 to Digital I/O Interface

p/n **110-0901** Rev **6.0**

916895

Printed in the United States of America.

Warranty Information

Your IOtech warranty is as stated on the *product warranty card*. You may contact IOtech by phone, fax machine, or e-mail in regard to warranty-related issues.
Phone: (440) 439-4091, fax: (440) 439-4093, e-mail: sales@iotech.com

Limitation of Liability

IOtech, Inc. cannot be held liable for any damages resulting from the use or misuse of this product.

Copyright, Trademark, and Licensing Notice

CE Notice

Many IOtech products carry the CE marker indicating they comply with the safety and emissions standards of the European Community. As applicable, we ship these products with a Declaration of Conformity stating which specifications and operating conditions apply.

Warnings, Cautions, Notes, and Tips

Refer all service to qualified personnel. This caution symbol warns of possible personal injury or equipment damage under noted conditions. Follow all safety standards of professional practice and the recommendations in this manual. Using this equipment in ways other than described in this manual can present serious safety hazards or cause equipment damage.

This warning symbol is used in this manual or on the equipment to warn of possible injury or death from electrical shock under noted conditions.

This ESD caution symbol urges proper handling of equipment or components sensitive to damage from electrostatic discharge. Proper handling guidelines include the use of grounded anti-static mats and wrist straps, ESD-protective bags and cartons, and related procedures.

This symbol indicates the message is important, but is not of a Warning or Caution category. These notes can be of great benefit to the user, and should be read.

In this manual, the book symbol always precedes the words "Reference Note." This type of note identifies the location of additional information that may prove helpful. References may be made to other chapters or other documentation.

Tips provide advice that may save time during a procedure, or help to clarify an issue. Tips may include additional reference.

Specifications and Calibration

Specifications are subject to change without notice. Significant changes will be addressed in an addendum or revision to the manual. As applicable, IOtech calibrates its hardware to published specifications. Periodic hardware calibration is not covered under the warranty and must be performed by qualified personnel as specified in this manual. Improper calibration procedures may void the warranty.

Quality Notice

IOtech has been an ISO 9001 registered firm since 1996. Prior to shipment, we thoroughly test our products and review our documentation to assure the highest quality in all aspects. In a spirit of continuous improvement, IOtech welcomes your suggestions.

Table of Contents

4 - IEEE 488 Primer

5 - Service Information

Appendix A Digital488 Command Summary

Appendix B IEEE Command and Address Messages

Appendix C Digital488/OEM Mechanical Dimensions

Introduction 1

General Description

Digital488

The **Digital488/32/OEM** is a board level interface with the same capabilities as the **Digital488**. All descriptions in this manual refer to both products unless otherwise stated. When the model number **Digital488** is used in this manual, **Digital488/32/OEM** is also implied.

The **Digital488** is a digital input and output interface to the IEEE 488 bus. Each unit has 40 TTL level digital I/O lines, which are divided, into 5 eight-bit ports. Each port is software programmable as input or output. The **Digital488** has several features, which give it versatile interface capability. A trigger output signal is asserted on the Group Execute Trigger (GET) command. Edge-triggered inputs can generate a Service Request on the bus. Six data formats are software programmable, including ASCII hexadecimal, ASCII character, ASCII binary, binary, high speed binary and ASCII decimal. There are also individual bit set and bit clear commands. Programmable terminators are provided to facilitate interfacing to various controllers.

A status mode enables the controller to interrogate the programmed status of the **Digital488** at any time. A self-test is initiated at power-on which checks for proper RAM and ROM operation.

When addressed to talk, the **Digital 488** will output data from all forty bits or a selected 8-bit port. When addressed to listen, the unit will input data and programming information from the controller and output the data to the appropriate I/O port.

Digital488/32/OEM

The **Digital488/32/OEM** is a 4 in. by 4 in., 32 I/O line interface board for transferring data between the IEEE 488 (GBIP, HP-IP) bus and devices equipped with up to 32-bit wide digital ports.

The **Digital488/32/OEM**’s 32 TTL-level digital I/O lines are programmable in 4-bit ports as either inputs or outputs. When addressed to talk, the **Digital488/32/OEM** will output data from all thirty-two bits or a selected 8 bit port. The board also offers six handshake lines for implementing clear, data strobe, external data ready, inhibit, trigger, and SRQ functions. Its firmware includes a complete command set for facilitating the implementation of all its functions. This command set is identical to that employed by the other board-level and external interfaces in IOtech’s industry-standard Digital488 family, facilitating quick prototyping and making the **Digital488/32/OEM** compatible with other **Digital488** family units.

Available Accessories

Additional accessories that can be ordered for the Digital488 include:

CA-7-1	1.5 foot IEEE 488 Cable
CA-7-2	6 foot IEEE 488 Cable
CA-7-3	6 foot shielded IEEE 488 Cable
CA-7-4	6 foot reverse entry IEEE 488 Cable
CA-8-50†	6 foot, 50-conductor ribbon cable with a card edge connector on one end, the other end un-terminated.
CA-46-40	6 foot digital I/O header connector to ribbon cable for the **Digital488/32/OEM**
CN-6-50†	50 Pin solder tab edge connector.
CN-20	Right Angle IEEE 488 adapter, male and female
CN-22	IEEE 488 Multi-tap bus strip, four female connectors in parallel
CN-23	IEEE 488 panel mount feed-through connector, male and female
Rack488-3†	5-1/4" by 19" rack mount for one **Digital488**
Rack488-4†	5-1/4" by 19" rack mount for two **Digital488**s
TR-2†	110 volt Wall mount power supply for the **Digital488**
TR-2E†	220 volt Wall mount power supply for the **Digital488**
TR-5	110 volt Wall mount power supply for the **Digital488OEM**
TR-5E	220 volt Wall mount power supply for the **Digital488OEM**

† For use with Digital488 Only

Specifications

Digital488 Specifications

Digital I/O

Terminal Installation Category:
Standard: Not Applicable. *CE:* Category 1.

Transistor-Transistor Logic (TTL) Levels:
Outputs will drive 2 TTL loads.

Connector:
One 50-pin card edge (mating connector supplied).

IEEE 488

CAUTION

The IEEE 488 terminal must only be used to control a non-isolated IEEE 488 system. The common mode voltage (cable shell to earth) must be zero.

Terminal Installation Category:
Standard: Not Applicable. *CE:* Category 1.

Implementation:
SH1, AH1, T6, TE0, L4, LE0, SR1, RL0, PP0, DC1, DT1, C0, E1.

Terminators:
Selectable CR, LF, LF-CR, and CR-LF with EOI.

Programmable:
IEEE Terminators, EOI, SRQ Mask, Port Data, Active Levels, Handshake Lines, Format and Configuration.

Connector:
Standard IEEE 488 connector with metric studs.

General

Configuration:
Five 8-bit ports, programmable as inputs or outputs. Also included are programmable handshake lines, data latching capability, Clear and Trigger outputs, and a Service Request (SRQ) input.

Terminal Installation Category:
Standard: Not Applicable. *CE:* Category 1 for all terminals.

Dimensions:
188 mm deep x 140 mm wide x 68 mm high (7.39" x 5.5" x 2.68").

Weight: 1.55 kg. (3.6 lbs).

Operating Environment:
Standard: Indoor, 0° to 50°C; 0 to 70% RH to 35°C. Linearly derate 3% RH/°C from 35 to 50°C.
CE: Indoor use at altitudes below 2000 meters, 0° to 40°C; 80% maximum RH up to 31°C decreasing linearly 4% RH/°C to 40°C.

Controls:
Power switch (external), and IEEE parameter switches (internal).

Indicators:
LED indicators for IEEE TALK, LISTEN, SRQ, ERROR, and POWER.

Power:
An external power supply is provided with the Digital488: Input is 105-125 VAC, or 210-250 VAC; 50/60 Hz, 10 VA maximum. The external power supply 9 VDC output is to be connected to the Digital488 power input marked: 10 VDC MAX @ 500 mA.

WARNING

Do not use this interface outdoors. The interface is intended for indoor use only. Outdoor conditions could result in equipment failure, bodily injury, or death.

CAUTION

Do not connect AC power line directly to the Digital488. Direct AC connection will damage equipment.

Digital488/32/OEM Specifications

WARNING
Do not use this interface outdoors. The interface is intended for indoor use only. Outdoor conditions could result in equipment failure, bodily injury, or death.

CAUTION
Never disassemble the interface case while it is connected to the AC power line. Internal voltage potentials exist which could cause bodily injury or death.

Digital I/O

Configuration:
Four 8-bit ports, programmable as inputs or outputs.

Transistor-Transistor Logic (TTL) Levels:
Outputs will drive 2 TTL loads.

Connector:
One 40 pin header, organized as two rows of 20 pins.

IEEE 488

Implementation:
SH1, AH1, T6, TE0, L4, LE0, SR1, RL0, PP0, DC1, DT1, C0, E1.

Terminators:
Selectable CR, LF, LF-CR, and CR-LF with EOI.

Programmable:
IEEE Terminators, EOI, SRQ Mask, Port Data, Active Levels, Handshake Lines, Format and Configuration.

Connector:
Standard IEEE 488 connector with metric studs.

General

Configuration:
Four 8-bit ports, programmable as inputs or outputs. Also included are programmable handshake lines, data latching capability, Clear and Trigger outputs, and a Service Request (SRQ) input.

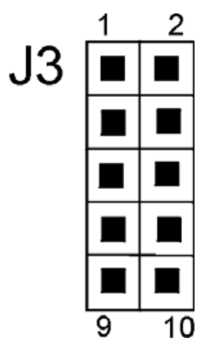
Dimensions:
101.6mm square x 16.51mm high (4” square x 0.65” high)

Weight: 0.13 kg. (0.29 lbs).

Operating Environment:
Standard: Indoor, 0° to 50°C; 0 to 70% RH to 35°C. Linearly derate 3% RH/°C from 35 to 50 °C.

Controls:
IEEE parameter switches.

Indicators:
On-board and 10 pin header for remote use. The external LEDs are connected to VCC through a resistor network. The pin-out table for the LED status header is located in Figure 1.1.



PIN #	LED CONNECTOR
1	Error (Cathode)
2	Error (Anode)
3	SRQ (Cathode)
4	SRQ (Anode)
5	Listen (Cathode)
6	Listen (Anode)
7	Talk (Cathode)
8	Talk (Anode)
9	Power (Cathode)
10	Power (Anode)

Figure 1.1: LED Indicators

Power:
User supplied +5 volts ±0.25% at 1 amp. Mating power connector with 8-inch leads provided.

WARNING

Do not use this interface outdoors. The interface is intended for indoor use only. Outdoor conditions could result in equipment failure, bodily injury, or death.

CAUTION

Never disassemble the interface case while it is connected to the AC power line. Internal voltage potentials exist which could cause bodily injury or death.

Abbreviations

The following IEEE 488 abbreviations are used throughout this manual.

addr n	IEEE bus address "n"
ATN	Attention line
CA	Controller Active
CR	Carriage Return
data	Data String
DCL	Device Clear
GET	Group Execute Trigger
GTL	Go To Local
LA	Listener Active
LAG	Listen Address Group
LF	Line Feed
LLO	Local Lock Out
MLA	My Listen Address
MTA	My Talk Address
PPC	Parallel Poll Configure
PPU	Parallel Poll Unconfigure
SC	System Controller
SDC	Selected Device Clear
SPD	Serial Poll Disable
SPE	Serial Poll Enable
SRQ	Service Request
TA	Talker Active
TAD	Talker Address
TCT	Take Control
term	Terminator
UNL	Unlisten
UNT	Untalk
*	Unasserted

Getting Started 2

Inspection

The unit was carefully inspected, both mechanically and electrically, prior to shipment.
When you receive the interface, carefully unpack all items from the shipping carton and check for any obvious signs of physical damage, which may have occurred during shipment. Report any such damage found to the shipping agent immediately. Remember to retain all shipping materials in the event that shipment back to the factory becomes necessary.

Every Digital488 is shipped with the following....

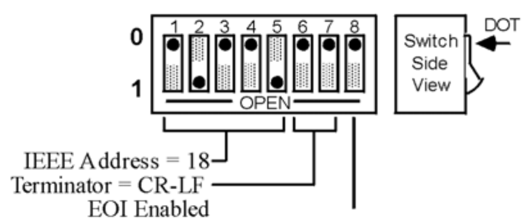
- **Digital488** IEEE Digital I/O Converter
- **CN-8-50** Digital I/O Port Mating Connector
- **Digital488** User's Manual
- **Power Supply** TR-2; 115V or
- TR-2E; 220V

Every Digital488/32/OEM is shipped with the following....

- **Digital488/32/OEM** IEEE Digital I/O Converter
- **Digital488** User's Manual
- **CA-106** 1 foot ribbon cable to IEEE488 connector
- **Macro488OEM-002 Power Plug Assembly**

Configuration

The **Digital488** has one internal 8 position switch which determines the unit's IEEE address and its default IEEE bus output terminator. The switch is only read when the unit is powered on, and should only be set prior to applying power. The following figure illustrates the factory default setting for **SW1.**



SW1 Factory Default Settings

To modify any of these defaults, follow this simple procedure. Disconnect the power supply from the AC line and from the interface. Disconnect any IEEE or digital I/O cables prior to disassembly.

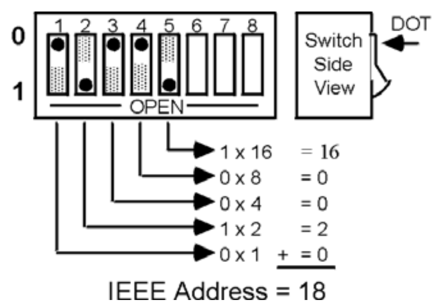
WARNING

Never open the Digital488 case while it is connected to the AC line. Failure to observe the warning may result in equipment failure, personal injury or death.

Remove the four screws located in each corner of the rear panel. Hold the case firmly and pull the rear panel outward, noting the slot location of the main circuit board. Modify those parameters, which are appropriate for your installation and reassemble the unit. Slide the main circuit board into the previously noted slot and finish reassembly by tightening the four screws into the rear panel.

IEEE 488 Address Selection

The IEEE 488 bus address is set by SW1-1 through SW1-5. The address can be set from 0 through 30 and is read only at power on. The address is selected by simple binary weighting with SW1-1 being the least significant bit and SW1-5 the most significant bit. The factory default is address 18.
If address 31 is selected, it defaults to address 30 because the IEEE 488 standard has reserved address 31.

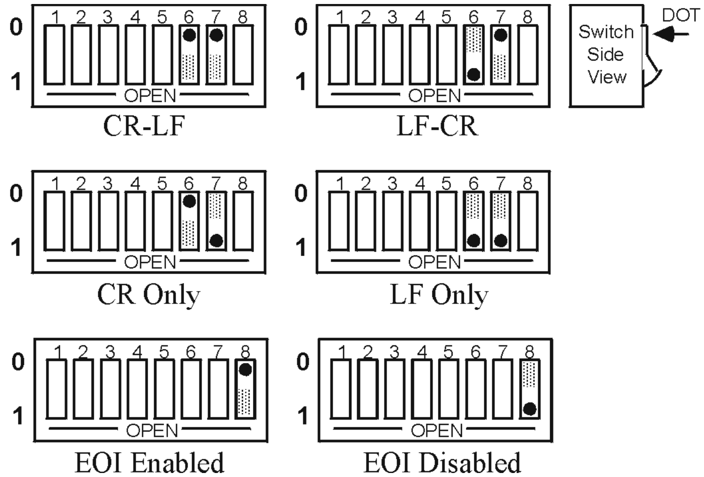


SW1 View for IEEE Bus Address Selection

IEEE 488 Bus Output Terminator Selection

The terminating characters sent on output by the **Digital488** are determined by **SW1-6** through **SW1-8.** The terminator switches are read only at power on, but can be changed by the controller through the **Terminator** command. If power is cycled after receipt of the **Terminator** command, then the unit will again default to the switch settings. The factory default settings are Carriage Return - Line Feed with EOI asserted.

The **Digital488** ignores all terminators <u>received</u> from the bus controller. Only the **Execute** command **(X)** is used to signal the **Digital488** that a command string has been completed.



SW1 View for Terminator Selection

Digital Input/Output Ports

The **Digital488** has 40 data lines, which can be programmed in groups of 8 as either input or output. At power on, all 40 bits are in the input mode. Each 8 bit group is one port, beginning with **Port 1** as the least significant 8 bits, and **Port 5** as the most significant 8 bits.

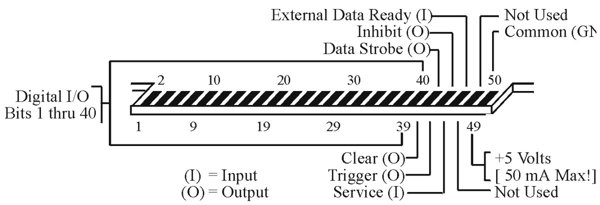
The **Digital488/32/OEM** has 32 data lines, which can be programmed in groups of 8 as either input or output. Each 8 bit group is one port, beginning with **Port 1** as the least significant 8 bits, and **Port 4** as the most significant 8 bits.

Logic Levels

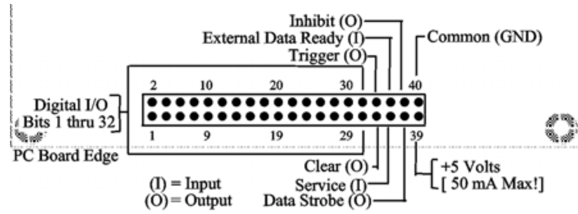
The data and handshake output lines will drive two TTL loads. In addition, ports 1 and 2 outputs are 5 Volt CMOS compatible. All input lines are less than 1.5 TTL loads. All inputs are protected against damage due to high static voltages. Normal precautions should be taken to limit the input voltages to -0.3 to +7.0 volts. All I/O lines are referenced to COMMON (Pin 50).

Digital I/O Port Pin Outs

The following diagram illustrates the digital I/O edge connector as view from the rear of the Digital488 and the top PC Board edge view of the Digital488/32/OEM.



Digital488 Rear Panel I/O Connector Pin Out



Digital488/32/OEM I/O Connector Pin Out

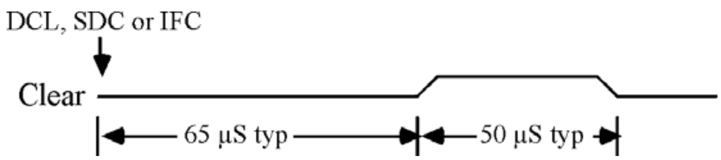
Pin Digital488	**Description**	**Pin Digital488/32/OEM**
1 thru 8 Least Significant Port	DATA PORT1 (Input or Output). Pin 1 is bit 1 (LSB), Pin 8 is bit 8 (MSB).	1 thru 8 Least Significant Port
9 thru 16	DATA PORT2 (Input or Output) Pin 9 is bit 1 (LSB), Pin 16 is bit 8 (MSB)	9 thru 16
17 thru 24	DATA PORT3 (Input or Output) Pin 17 is bit 1 (LSB), Pin 24 is bit 8 (MSB)	17 thru 24
25 thru 32	DATA PORT4 (Input or Output) Pin 25 is bit 1 (LSB), Pin 32 is bit 8 (MSB)	25 thru 32 Most Significant Port
33 thru 40 Most Significant Port	DATA PORT5 (Input or Output Pin 33 is bit 1 (LSB), Pin 40 is bit 8 (MSB)	N/A
41	CLEAR (Output)	33
42	DATA STROBE (Output)	37
43	TRIGGER (Output)	34
44	INHIBIT (Output)	38
45	SERVICE INPUT (Input).	35
46	EXTERNAL DATA READY [EDR] (Input)	36
47,48	Not used	N/A
49	+5 Volts (Do not exceed 50 mA load)	39
50	I/O COMMON (Gnd)	40

Control Lines

Six control lines enable handshaking of digital I/O data transfer to the **Digital488**. They are automatically activated with the corresponding I/O activity and can be independently activated with the **Handshake** (Hn) command. Note that the pin numbers for the **Digital488/32/OEM** are different from the pin numbers for the **Digital488**.

*Clear (Digital488: Pin 41
Digital488/32/OEM: Pin 33)*

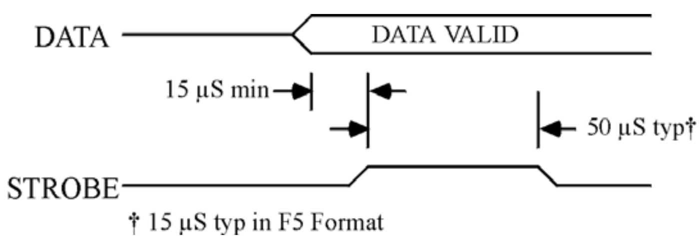
The **Clear** output is pulse for approximately 50 microseconds after a Device Clear (DCL), Selected Device Clear (SDC), or Interface Clear (IFC) command has been sent on the bus. The **Clear** line is normally active high. The **Invert** command (**I8**) will program it active low. The **Handshake** command **(H0)** can pulse the **Clear** line, independent of any I/O operations.



Timing Diagram for Clear Output

*Data Strobe (Digital488: Pin 42
Digital488/32/OEM: Pin 37)*

The **Data Strobe** output is pulse for approximately 50 microseconds after new data is output on the I/O port. The **Data Strobe** line is normally active high but may be programmed active low by the **Invert** command (**I4**). The **Handshake** command **(H1)** can pulse the **Data Strobe** line, independent of an I/O operation.



Timing Diagram for Strobe Output

*External Data Ready [EDR] (Digital488: Pin 46
Digital488/32/OEM: Pin 36)*

The **External Data Ready** [**EDR**] line is an edge sensitive input which is used to latch input data. It is used in conjunction with the **Data Ready** command (**R1**). The **EDR** signal must be at least 1 microsecond wide and must have a rise and fall time of less than one microsecond. The **EDR** line is normally rising-edge sensitive but can be programmed with the **Invert** command (**I32**) to be falling-edge sensitive.
Refer to the following diagram for timing relationships.

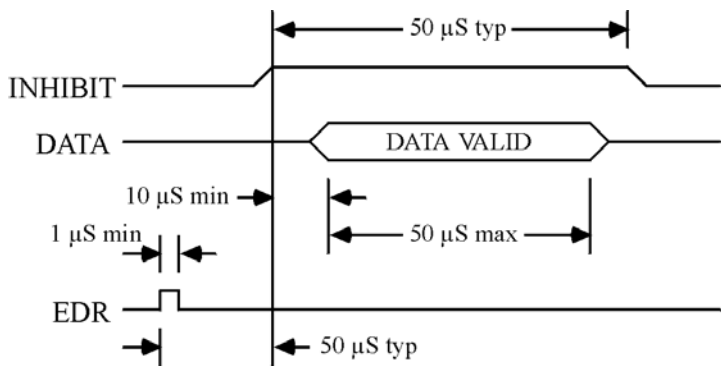
When using the **EDR** line with the **R1** command, data is not read when the **Digital488** is addressed to talk as with **R0**. The **Digital488** will only output data when the **EDR** line transitions.

EDR is not functional in the high-speed binary (**F5**) format.

*Inhibit (Digital488: Pin 44
Digital488/32/OEM: Pin38)*

The **Inhibit** output is asserted while data on the selected I/O port is being read into the I/O port buffer. This line is normally active high but may be programmed active low by the **Invert** command (**I1**).
The **Inhibit** line can be programmed independent of any I/O operations with the **Inhibit** command (**Qn).**
Refer to the following diagram for timing relationships.

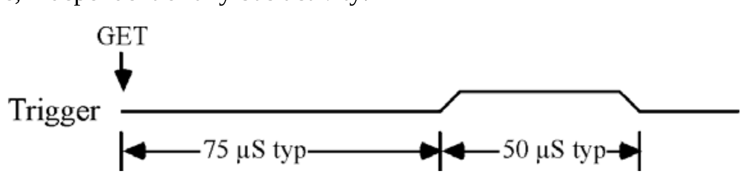
The **Inhibit** line is asserted once for each data read operation for all format [**Fn**] modes except high-speed binary [**F5**]. In this mode, it is asserted for the first data read after the **Digital488** is addressed to talk.
On the last data-byte transfer, the data is read again with **Inhibit** asserted in anticipation of another data transfer. If **Inhibit** is used to sequence external hardware, you should be aware that this line will pulse N+1 times; where N is the number of total (5 byte) data transfers.



Timing Diagram for EDR Input and Inhibit Output

*Trigger (Digital488: Pin 43
Digital488/32/OEM: Pin34)*

The **Trigger** output is pulse for approximately 50 microseconds after a **GET** (Group Execute Trigger) command is received from the bus controller. The trigger pulse is normally active high, but can be made active low with the **Invert** command (**I2**). The **Handshake** command (**H2**) can independently pulse the **Trigger** line, independent of any bus activity.



Timing Diagram for Trigger Output

Service (Digital488: Pin 45
Digital488/32/OEM: Pin35)

The **Service** input is an edge sensitive input capable of generating a bus **Service Request** (**SRQ**). It is enabled with the **SRQ** command (**M1**) and defaults to rising-edge sensitive. The **Invert** command (**I64**) can be used to program it to be falling-edge sensitive.

IEEE 488 Bus Implementation

The Digital488 implements many of the capabilities defined by the IEEE 488 1978 specification. These are discussed in the following sections. Those bus uniline and multiline commands that the Digital488 does not support or respond to include:

Remote Enable (REN)	Parallel Poll (PP)
Go to Local (GTL)	Parallel Poll Configure (PPC)
Local Lockout (LLO)	Parallel Poll Unconfigure (PPU)
Take Control (TCT)	Parallel Poll Disable (PPD)

My Talk Address (MTA)

When the **Digital488** is addressed to talk (**R0**) it asserts **Inhibit**, reads the data from all ports, un-asserts **Inhibit** and outputs the data to the bus in the format as defined by the **Fn**, **Pn** and **Gn** commands. The output bus terminators are appended to the output with the exception of the **F4** and **F5** formats. **F4** does not append terminators. The output format of **F5** will be described separately. After output in the **F0** through **F4** formats, the **Digital488** must be re-addressed to talk to perform subsequent reads.

In the **R1** mode, it will wait for the selected **EDR** transition before reading the data and formatting it for output. If the **EDR** line has transitioned prior to being addressed to talk, the data read at the time of **EDR** will be buffered for output when next addressed to talk. If **EDR** transitions again before the previous **EDR** buffered data has been output, the **Digital488** will generate an **EDR Overrun** error and ignore the **EDR** read request. After output in the **F0** through **F4** formats, the **Digital488** must be re-addressed to talk to perform subsequent buffered output of **EDR** captured data.

In either **Rn** mode, the **Digital488** can send requested status (**Un**) without affecting the data ports or **Inhibit**. After the requested status is output, the presently programmed **Rn** mode returns.

EDR cannot be used to capture data in the high-speed binary format (**F5**). When addressed to talk in this format it asserts **Inhibit**, reads the data from all ports, un-asserts **Inhibit** and outputs the binary data to the bus with EOI asserted on the fifth byte. When the last data byte is transferred, the data is read again in anticipation of another data transfer. If **Inhibit** is used to sequence external hardware, this line will pulse N+1 times; where N is the number of total (5 byte) data transfers. In this format, the **Digital488** does not have to be re-addressed to talk to read the ports multiple times.

With all **Fn** formats, using the **Digital488** the data is output in a PORT5, PORT4, PORT3, PORT2, PORT1 sequence. Using the **Digital488/32/OEM** the data is output in a PORT4, PORT3, PORT2, PORT1 sequence.

My Listen Address (MLA)

When the **Digital488** is addressed to listen in the **F0** through **F4** format, it accepts characters from the active talker and interprets these characters as commands and command parameters. These commands are explained in Chapter 3.

In the high-speed binary format (**F5**), the command interpreter is disabled. The **Digital488** treats all bytes received as data to be output to the Digital I/O ports. Each time it receives five bytes or detects EOI it pulses the **Data Strobe** for approximately 15 microseconds. Using the **Digital488** data is expected I n a PORT5, PORT4, PORT3, PORT2, PORT1 sequence. Using the **Digital488/32/OEM** data is expected in a PORT4, PORT3, PORT2, PORT1 sequence.

If only two bytes are received, with EOI asserted on the second byte, the **Digital488** will update PORT5 with the first byte received PORT4 with the second and pulse the Data Strobe.
If using **Digital488/32/OEM** will update PORT4 with the first byte received, and PORT3 with the second and pulse the Data Strobe. Since the interface treats all received characters as data, the Status (Un) command will not be recognized.

Device Clear (DCL and SDC)

In the **F0** thru **F4** formats, Device Clear resets the **Digital488** to power on defaults and pulses the **Clear** output line for approximately 50 microseconds.

In the high-speed binary format (**F5**), it enables the command interpreter and changes the format to **F0**. All other parameters remain unchanged. In addition, the **Clear** output line is not pulsed by DCL or SDC when the interface is in **F5**. This is the only programmable method to exit the **F5** format.

Group Execute Trigger (GET)

When the **Digital488** recognizes a GET, it pulses the Trigger output line for approximately 50 microseconds.

Interface Clear (IFC)

IFC places the **Digital488** in the Talker/Listener Idle State and pulses the Clear output line for approximately 50 microseconds.

Serial Poll Enable (SPE)

When Serial Poll Enabled, the **Digital488** sets itself to respond to a serial poll with its serial poll status byte if addressed to talk. When the serial poll byte is accepted by the controller, any pending SRQs are cleared. The **Digital488** will continue to try to output its serial poll response until it is serial poll disabled by the controller.

Serial Poll Disable (SPD)

Disables the **Digital488** from responding to serial polls by the controller.

Unlisten (UNL)

UNL places the **Digital488** in the Listener Idle State.

Untalk (UNT)

UNT places the **Digital488** in the Talker Idle State.

Installation

To begin operating the **Digital488,** plug the external power supply into the rear jack on the interface.

CAUTION
Never install the power supply into the interface while it is connected to AC line power. Failure to observe this caution may result in damage to the Digital488.

WARNING
Do not use this interface outdoors. The interface is intended for indoor use only. Outdoor conditions could result in equipment failure, bodily injury, or death.

After installing the power supply connector into the interface, turn on the **Digital488** by depressing the rear panel power switch. All the front panel LEDs should light for approximately one second while the **Digital488** performs an internal ROM and RAM self check. At the end of this self-check, all indicators should turn off except **POWER**.

If you obtain the above response then your **Digital488** is alive and well. If all LEDs remain on, then a ROM error has occurred. If all LEDs continue to flash (except the power LED), then a RAM error has occurred. Try cycling the power to the **Digital488** to determine that the error is repeatable.

If the LEDs do not flash and the **POWER** indicator does not remain lit, there may not be any power supplied to the interface. In this event, check to make sure the AC power is supplied to the power supply, and that the supply is properly installed into the unit. If the problem is unresolved, refer to the **Service Information** section of this manual.

Notes

Command Descriptions 3

Control of the **Digital 488** is implemented with 17 bus commands, described here in detail. Examples are given for many of the commands using a Hewlett-Packard 85 computer in the immediate mode. It is implied that each command is terminated by the 'END LINE' key on the HP-85 in order to execute the command. The **Digital488** bus address should be set to 18 for all examples.

It is necessary that the **EXECUTE command (X)** follow all command strings sent to the **Digital488**. No commands are executed until an **X** is received by the **Digital488.**

Bit Set *An*

The **Bit Set** command programs a logic one output to a bit described by the argument 'n'. Setting a bit may represent either a +5 volt or 0 volt output, depending on whether an **Invert** command (**I16**) has been sent. If data is active high (default condition), then **Bit Set** outputs +5 volts. If multiple bits are to be set within the same command string, an **Execute** command (**X**) must be included after every **Bit Set** command.

The bit which is being set must have been configured as an output bit by the **Configure** command to be valid. The **Strobe** output line is not pulsed when the **Bit Set** command is sent.

An Bit n (1 thru 40) is set to logic one

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"C5X"`	configure all ports as output
`OUTPUT 718;"A22X"`	set bit 22 to a logic one
`OUTPUT 718;"A23XA24X"`	set bits 23 and 24 to a logic one

Bit Clear *Bn*

The **Bit Clear** command will clear to a logic zero an output bit described by the argument 'n'. Clearing a bit may represent either a 0 volt or +5 volt output, depending on whether an **Invert** command (**I16**) has been sent. If data is active high (default condition), then **Bit Clear** outputs 0 volts. When multiple **Bit Clear** commands are used in the same command string, an **Execute** command (**X**) must follow each command.

The bit that is being cleared must have been defined as an output by the **Configure** command in order to be valid. The **Strobe** output line is not pulsed when the **Bit Clear** command is sent.

Bn Bit n (1 thru 40) is cleared to a logic 0

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"C5X"`	configure all ports as output
`OUTPUT 718;"A7XA8XA9X"`	set bits 7, 8, and 9 to +5 volts
`OUTPUT 718;"B7X"`	clear bit 7 to zero volts
`OUTPUT 718;"B8XB9X"`	clear bits 8 and 9 to zero volts

Bus Input/Output — *Gn*

The **Bus Output** command determines whether input port data, output port data or both will be transmitted on the bus when the **Digital488** is addressed to talk. The amount of data sent is dependent on the **Pn** command.

The **G0** default mode causes all input and output port data to be sent to the controller when addressed to talk. The **G1** mode causes only data from the ports programmed as inputs to be returned when addressed to talk. The **G2** mode causes only data from ports programmed as outputs to be returned when addressed to talk.

If all ports are programmed as outputs with **G1** selected and the **Digital488** is addressed to talk, nothing will be transmitted and the bus will hang. The converse will also cause the bus to hang with all ports programmed as inputs and **G2** selected.

G0	<u>Input</u> and <u>output</u> port data is send on talk
G1	Only <u>input</u> port data is sent on talk
G2	Only <u>output</u> port data is sent on talk

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"P0C1X"`	port1 as output, ports 2-5 as input
`OUTPUT 718;"G1X"`	select only input ports
`ENTER 718; A$`	read data from the input ports
`DISP A$`	display shows FFFFFFFF (data is dependent on what is connected.
`OUTPUT 718;"G2X"`	select output ports
`ENTER 718; A$`	read data from the output ports
`DISP A$`	display shows 00 (outputs default to 0)

Configure — *Cn*

Ports 1 thru 5 are configured as inputs or outputs with the **Configure** command.
Each port is eight bits wide. At power-on, all ports are initialized as inputs. The **Configure** command is usually the first command to be sent after power on. All ports programmed as outputs will be set to a logic zero after receiving the **Configure** command. The actual output level is dependent on the **Invert** command (**I16**).

Cn Mode n (0 thru 5) defines which ports are input and output

Port	**5**	**4**	**3**	**2**	**1**
C0	in	in	in	in	in
C1	in	in	in	in	out
C2	in	in	in	out	out
C3	in	in	out	out	out
C4	in	out	out	out	out
C5	out	out	out	out	out

in = programmed as an input port
out = programmed as an output port

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"C1X"`	select port 1 as output, ports 2 thru 5 as inputs

Data — Dn....Z

The **Data** command outputs up to 40 bits of data to the output ports. The number of bits, which can be sent with the Data command, is limited by the number of bits programmed as outputs. For formats **F0** through **F3**, if the amount of data sent is less than the number of bits programmed as outputs, the least-significant bits will contain the data sent and the most-significant bits will be cleared to logic zero. If a single port is selected with the **Port** command, only eight bits may sent with the **Data** command. The **Data Strobe** output is pulse for approximately 50 microseconds after new data is output on the selected ports.

For formats **F0** through **F3**, data sent by the controller is contained within a prefix (**D**) and a suffix (**Z**). In format **F4**, the five bytes immediately following the prefix (**D**) is interpreted as data and the suffix (**Z**) is not used. For the high-speed binary **F5** format, all bytes received are treated as data and the prefix and suffix are not used. Refer to the **Fn** command for additional details.

Dn...Z **n...** represents the data to be outputted, terminated by **Z**.

In the F4 mode, the Z terminator is not allowed

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"C5P1X"`	all ports as output, select port 1
`OUTPUT 718;"D55ZX"`	send 55 to port 1
`ENTER 718; A$`	read data from port 1
`DISP A$`	display shows 55
`OUTPUT 718;"P0X"`	select all ports
`OUTPUT 718;"D1234567890ZX"`	send data to all 40 bits
`ENTER 718;A$`	read data from the **Digital488**
`DISP A$`	display shows 1234567890
`OUTPUT 718;"D123ZX"`	send 12 bits of data to the least significant bits
`ENTER 718; A$`	read data from the **Digital488**
`DISP A$`	display shows 0000000123
`OUTPUT 718;"P5D21ZX"`	set port 5 only
`OUTPUT 718;"P0X"`	select all ports
`ENTER 718; A$`	read data from the **Digital488**
`DISP A$`	display shows 2100000123

Data Ready *Rn*

The **Data Ready** command enables digital input data to be latched. When used in conjunction with the **Service Request** (**M2**) command, the **External Data Ready** line can both latch the input data and signal the controller that new data is available.

In the default mode, (**R0**) data is read when the **Digital488** is addressed to talk. In the **R1** mode, it will wait for the selected **External Data Ready** (**EDR**) transition before reading the data and formatting it for output. If the **Digital488** is addressed to talk before **EDR** is asserted, the bus will hang up until the **EDR** pulse occurs. Once **EDR** is asserted, the data will remain latched until the interface is addressed to talk and the data is read by the controller. If **EDR** transitions again before the previous **EDR** buffered data has been output, the **Digital488** will generate an **EDR Overrun** error and ignore the **EDR** read request.

After output in the **F0** through **F4** formats, the **Digital488** must be re-addressed to talk to perform subsequent buffered output of **EDR** captured data. **EDR** cannot be used to capture data in the **F5** high-speed binary format

The **EDR** signal must be at least 1 microsecond wide and should have a rise and fall time of less than 1.0 microsecond. The **EDR** line defaults to rising-edge sensitive but can be changed to falling-edge sensitive with the **Invert** command (**I32**).

R0	Data is not latched, and is read whenever the **Digital488** is addressed to talk
R1	Data is latched on an **EDR** transition

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"R1X"`	data is only read after a rising-edge signal is applied to the **EDR** line

End or Identify (EOI) *Kn*

The **EOI** line is one of five interface management lines on the IEEE 488 Bus. It is used by a talker to indicate the end of a multiple byte transfer sequence. At power-on, the setting of Switch **S1** determines the default **EOI** mode**.** The controller can change the **EOI** mode by programming the **Digital488** from the bus. In the **K0** mode, the **EOI** line is asserted by the **Digital488** on the last byte of every bus output string. In the **K1** mode the **EOI** function is disabled (except when using the binary modes [**F4** and **F5**])**.**

K0	**EOI** enabled, assert **EOI** on last byte transferred
K1	**EOI** disabled, do not assert **EOI** on last byte transferred

Example:

`OUTPUT 718;"K1X"`	disables EOI on last byte

Execute *X*

Commands sent to the **Digital488** will result in no action until the unit is instructed to execute these commands. This is done by sending an **X**, usually as the last character of a command string. Commands sent without an **X** are stored in the internal buffer until an **X** is received. Any number of **Execute** commands may be inserted into the same command string. Certain commands, such as **Bit Set** require an **X** after each command in a string if more than one of that command is within the same string.

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"F2"`	send "F2" to the **Digital488** command input buffer
`OUTPUT 718;"X"`	instruct the **Digital488** to execute its command input buffer
`OUTPUT 718;"A1XA2X"`	Two Bit Set (A) commands are within the same string, requiring an X after each command.

Format *Fn*

The **Format** command determines the method by which input and output data will be described. Six data formats are available.

F0	ASCII Hexadecimal (4 bits per character)
F1	ASCII Character (4 bits per character)
F2	ASCII Binary (1 bit per character)
F3	ASCII Decimal (8 bits per number)
F4	Binary (each byte represents 8 bits)
F5	High Speed Binary (each byte represents 8 bits)

F0 Format- ASCII Hexadecimal

In the default **F0** format, the data is described in ASCII hexadecimal, with each character having a value from 0 thru 9 or A thru F. Each ASCII character describes 4 bits of data.

F0 Character	Decimal Equiv	F0 Character	Decimal Equiv
0	0	8	8
1	1	9	9
2	2	A	10
3	3	B	11
4	4	C	12
5	5	D	13
6	6	E	14
7	7	F	15

Data received for output to the digital ports must be contained within a prefix (**D**) and a suffix (**Z**). If the amount of data sent is less than the number of bits programmed as outputs, the least-significant bits will contain the data sent and the most-significant bits will be cleared to logic zero. If the data sent is greater than the number of bits programmed for output or selected by the **Pn** command, the **Digital488** will generate a conflict error and ignore the entire command string. The **Data Strobe** output is pulse for approximately 50 microseconds after new data is output on the selected port(s).

When the **Digital488** is addressed to talk (**R0**) it asserts **Inhibit**, reads the data from <u>all</u> ports, unasserts **Inhibit** and outputs the number of characters determined by the **Gn** and **Pn** commands. Leading zeros are not suppressed and the bus terminators are appended to the output. After output the **Digital488** <u>must</u> be re-addressed to talk to perform subsequent reads. **EDR** (**R1**) may also be used to capture data in this format.

Example:

`DIM A$[50]`	dimension the length of A$
`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"C2G2X"`	configure ports 1 & 2 as output
`OUTPUT 718;"D4E6BZX"`	output hexadecimal `4E6B` to ports 1 & 2
`ENTER 718; A$`	read data from the **Digital488**
`DISP A$`	display shows `4E6B`

F1 Format - ASCII Character

In the **F1** format, the data is coded and transmitted in ASCII Characters with the four least significant bits of each ASCII character representing four bits of data.

F1 Character	Decimal Equiv	F1 Character	Decimal Equiv
0	0	8	8
1	1	9	9
2	2	:	10
3	3	;	11
4	4	<	12
5	5	=	13
6	6	>	14
7	7	?	15

Data received for output to the digital ports must be contained within a prefix (**D**) and a suffix (**Z**). If the amount of data sent is less than the number of bits programmed as outputs, the least-significant bits will contain the data sent and the most-significant bits will be cleared to logic zero. If the data sent is greater than the number of bits programmed for output or selected by the **Pn** command, the **Digital488** will generate a conflict error and ignore the entire command string.

The **Data Strobe** output is pulse for approximately 50 microseconds after new data is output on the selected port(s).

When the **Digital488** is addressed to talk (**R0**) it asserts **Inhibit**, reads the data from <u>all</u> ports, unasserts **Inhibit** and outputs the number of characters determined by the **Gn** and **Pn** commands. Leading zeros are not suppressed and the bus terminators are appended to the output. After output the **Digital488** <u>must</u> be re-addressed to talk to perform subsequent reads. **EDR** (**R1**) may also be used to capture data in this format.

Example:

`OUTPUT 718;"F1X"`	select ASCII Character format
`ENTER 718; A$`	read data from the **Digital488**
`DISP A$`	display shows `4>6;`
`OUTPUT 718;"D1??2ZX"`	send `1??2` to the **Digital488**
`ENTER 718; A$`	read data from the **Digital488**
`DISP A$`	display shows `1??2`

F2 Format - ASCII Binary

In the **F2** format, the each data bit is described with an ASCII 0 or 1. Each byte is formatted in two 4-bit multiples separated by semicolons.

F2 String	Decimal Equiv	F2 String	Decimal Equiv
0000;0000	0	0000;1001	9
0000;0001	1	0000;1010	10
0000;0010	2	0000;1011	11
0000;0011	3	0000;1100	12
0000;0100	4	0000;1101	13
0000;0101	5	0000;1110	14
0000;0110	6	0000;1111	15
0000;0111	7	1000;0001	129
0000;1000	8	1111;1111	255

Data received for output to the digital ports must be contained within a prefix (**D**) and a suffix (**Z**) and each 4-bit quantity must be separated by semicolons. Leading zeros are not required. If the amount of data sent is less than the number of bits programmed as outputs, the least-significant bits will contain the data sent and the most-significant bits will be cleared to logic zero. If the data sent is greater than the number of bits programmed for output or selected by the **Pn** command, the **Digital488** will generate a conflict error and ignore the entire command string. The **Data Strobe** output is pulse for approximately 50 microseconds after new data is output on the selected port(s).

When the **Digital488** is addressed to talk (**R0**) it asserts **Inhibit**, reads the data from all ports, unasserts **Inhibit** and outputs the number of characters determined by the **Gn** and **Pn** commands. Leading zeros are not suppressed and the bus terminators are appended to the output. After output the **Digital488** must be re-addressed to talk to perform subsequent reads. **EDR** (**R1**) may also be used to capture data in this format.

Example:

`OUTPUT 718;"F2X"`	select ASCII/binary mode
`ENTER 718;A$`	read data from the **Digital488**
`DISP A$`	display shows `0001;1111;1111;0001`
`OUTPUT 718;"D1111;0;1010;0101ZX"`	
`ENTER 718; A$`	read data from the **Digital488**
`DISP A$`	display shows `1111;0000;1010;0101`

F3 Format - ASCII Decimal

In the F3 format, the data is described in decimal 8 bit multiples and transmitted in ASCII. Each decimal number (0 to 255) to be output must be separated by semicolons.

F3 Number	Decimal Equiv	F3 Number	Decimal Equiv
000	0	008	8
001	1	009	9
002	2	010	10
003	3	020	20
004	4	100	100
005	5	200	200
006	6	210	210
007	7	255	255

Data received for output to the digital ports must be contained within a prefix (**D**) and a suffix (**Z**). If the amount of data sent is less than the number of bits programmed as outputs, the least-significant bits will contain the data sent and the most-significant bits will be cleared to logic zero. If the data sent is greater than the number of bits programmed for output or selected by the **Pn** command, the **Digital488** will generate a conflict error and ignore the entire command string. The **Data Strobe** output is pulse for approximately 50 microseconds after new data is output on the selected port(s).

When the **Digital488** is addressed to talk (**R0**) it asserts **Inhibit**, reads the data from all ports, unasserts **Inhibit** and outputs the number of characters determined by the **Gn** and **Pn** commands. Leading zeros are not suppressed and the bus terminators are appended to the output. After output, the **Digital488** must be re-addressed to talk to perform subsequent reads. **EDR** (**R1**) may also be used to capture data in this format.

Example:

`OUTPUT 718;"F3X"`	select decimal mode
`ENTER 718; A$`	read data from the **Digital488**
`DISP A$`	display shows `240;165`
`OUTPUT 718;D100;200ZX`	output `100 & 200` to the **Digital488**
`ENTER 718; A$`	read data from the **Digital488**
`DISP A$`	display shows `100;200`

F4 Format - Binary

In the F4 binary format, no error checking is performed and caution must be exercised when using this mode to avoid locking the IEEE bus.

When addressed to listen, the **Digital488** expects the "D" prefix followed by five bytes of data beginning with PORT5 without the "Z" suffix. If any digital I/O port is configured as input, the data to that input port will be ignored.

When the **Digital488** is addressed to talk (R0) it asserts Inhibit, reads the data from all ports, unasserts Inhibit and outputs 5 bytes beginning with PORT5 with EOI asserted on the last bye. Bus terminators, with the exception of EOI, are not appended to the output. After output, the **Digital488** must be re-addressed to talk to perform subsequent reads. EDR (R1) may also be used to capture data in this format.

F5 Format - High Speed Binary

In the **F5** high-speed binary format, the command interpreter is disabled. When addressed to listen, the **Digital488** treats all bytes received as data to be output to the Digital I/O ports. Each time it receives five bytes **or** detects EOI asserted, it pulses the **Data Strobe** for approximately 15 microseconds. Data is expected in a PORT5, PORT4, PORT3, PORT2, PORT1 sequence. If only two bytes are received, with EOI asserted on the second byte, the **Digital488** will update PORT5 with the first byte received PORT4 with the second and pulse the **Data Strobe**. Since the interface treats all received characters as data, the **Un** command will not be recognized.

To place the **Digital488** in the **F5** format, the 3-character string "**F5X**" should be the last command sent to the interface **without** terminators. Any characters appended to this command, such as carriage return or line feed, will be considered data and the output ports will reflect those character values.

When addressed to talk in this format, it asserts **Inhibit**, reads the data from all ports, unasserts **Inhibit** and outputs the binary data to the bus with EOI asserted on the fifth byte. When the last data byte is transferred, the data is read again in anticipation of another data transfer. If **Inhibit** is used to sequence external hardware, this line will pulse N+1 times; where N is the number of total (5 byte) data transfers. In this format, the **Digital488** does **not** have to be re-addressed to talk to read the ports multiple times. **EDR** cannot be used to capture data in the **F5** high-speed binary format.

The only programmable method to exit the **F5** high-speed binary format is device clear (**DCL**) or Selected Device Clear (**SDC**). When received, it enables the command interpreter and changes the format to **F0**. All other parameters remain unchanged. In addition, the **Clear** output line is not pulsed by DCL or SDC when the interface is in **F5**.

Handshake *Hn*

The **Handshake** control command enables software control of the handshake lines, independent of any other I/O operations. When the **Digital488** receives a **Hn** command, the respective handshake line is pulsed for approximately 50 microseconds. It returns to its steady-state condition after pulsing.
The **Invert** command may be used to change the active state of any of the handshake lines.

H0	The **Clear** line is pulsed
H1	The **Strobe** line is pulsed
H2	The **Trigger** line is pulsed

Example:

`OUTPUT 718;"H1X"`	the Strobe line is pulsed

Inhibit *Qn*

The **Inhibit** control command allows software control of the **Inhibit** line, independent of any other I/O activities. The 'set' and 'clear' levels of the **Inhibit** line are determined by the **Invert** command.

Q0	Clear the **Inhibit** line (return to unasserted state)
Q1	Set the **Inhibit** line (place in the asserted state)

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"Q1X"`	set the Inhibit line

Invert *In*

The **Invert** command is used to change the polarity of the handshake and data lines. At power up all handshake and control lines are active high (logic one = + 5 volts). The **Invert** command can selectively change the polarity of each of the handshake lines, and of the data lines. If multiple **Invert** commands are contained within the same string, then an **Execute** command (**X**) should be included between each **Invert** command. An alternative is to add the values of each **Invert** command desired, and send one command with the sum of the desired commands. The **Invert** commands are ORed together as received. To delete any one command, it is necessary to program the default mode **I0**, then reprogram the desired commands.

I0	All control lines are active high, all data lines are high true
I1	**Inhibit** output is active low
I2	**Trigger** output is active low
I4	**Data Strobe** output is active low
I8	**Clear** output is active low
I16	**Data** is low true
I32	**EDR** input is falling-edge sensitive
I64	**Service** input is falling-edge sensitive

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"I32XI64X"`	select EDR and Service input as falling-edge sensitive

Note:

`OUTPUT 718;"I96X"`	performs the same function as above

Port *Pn*

The **Port** command determines which port is selected for data input/output. In the default mode (**P0**), all ports are selected. The **P1** thru **P5** commands select a specific eight-bit port.

It is recommend that the **Bus Output** command be used with the **PO** mode to determine which ports will be output when the **Digital488** is addressed to talk. Data in modes **P1** through **P5** will be input or output in-groups of eight bits.

P0	All five ports are selected
P1	Port 1 is selected
P2	Port 2 is selected
P3	Port 3 is selected
P4	Port 4 is selected
P5	Port 5 is selected

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"P4X"`	select port 4

Service Request Mask (SRQ) *Mn*

The **Service Request (SRQ)** mode is used by the **Digital488** to alert the controller to one of several conditions described below. Multiple **SRQ** conditions can be enabled simultaneously by issuing them separately or by combining them in one command. If multiple **SRQ** commands are contained within the same command string, each **SRQ** command should be followed by an **Execute** command **(X).**
The programmed **SRQ** modes will remain enabled until the **M0** command is sent, or the controller sends a Device Clear (DCL), Selected Device Clear (SDC), or Interface Clear (IFC) command.

M0	SRQ is disabled
M1	SRQ on **Service** input transition
M2	SRQ on **EDR** input transition
M4	SRQ on bus error
M8	SRQ on Self-Test error
M16	SRQ on Ready

MO default mode disables the SRQ function, preventing the **Digital488** from generating a Service Request.

M1 will generate a Service Request when the Service Input line makes a transition. Refer to the Invert command (I64) description for programming the polarity of the Service input line.

M2 will generate a Service Request when the EDR input makes a transition. Refer to the Invert command (I32) description for programming the polarity of the EDR input line.

M4 will generate a Service Request when a bus error occurs. The most common bus error is sending an invalid command to the **Digital488**. For example, attempting to select an 'F6' format when no 'F6' format exists will generate a Service Request when the M4 mode is selected.

M8 will generate a **Service Request** when the **Digital488** self-test fails. Refer to the **Test** command **(T0)** description for details on self-tests.

M16 will generate a **Service Request** when the **Digital488** has completed the execution of a set of commands from the bus controller. This is used by the controller to assure the completion of a set of commands before sending a subsequent set of commands.

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"M4X"`	select SRQ on Bus error
`OUTPUT 718;"F7X"`	send an invalid bus command.
	Note: ERROR and SRQ LEDs should illuminate
`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"M1XM4X"`	select SRQ on Bus error and SRQ on Service input.
`OUTPUT 718;"M5X"`	This has the same effect as the command above where `M1X` plus `M4X` equals `M5X`.

Serial Poll Status Byte

The **Serial Poll Output** byte is sent upon receiving the serial poll command from the controller. Refer to the **SRQ** description for details on how the **Serial Poll** byte is affected. Below is a description of the significance of each bit in the **Serial Poll** byte.

Bit Location	Significance	(SRQ Bit Value if set to logic 1)
DIO1(LSB)	1	Service Input transition
DIO2	2	EDR input transition
DIO3	4	Bus error
DIO4	8	Test error
DIO5	16	Ready for more commands
DIO6	32	not assigned, always 0
DIO7	64	Service Request bit
DIO8 (MSB)	128	not assigned, always 0

Serial Poll Bit Description

DIO1 When enabled by the **M1** command, **DIO1** is set by a transition on the Service Input line (active transition state determined by the **Invert** command **(I64)**). **DIO1** is cleared after the controller serial polls the **Digital488.**

DIO2 When enabled by the **M2** command, DIO2 is set on an **EDR** transition (active transition state determined by the **Invert** command **(I32)**). **DIO2** is cleared after the controller serial polls the **Digital488.**

DIO3 **DIO3** is set when an invalid command is sent to the **Digital488.** The **M4** command will enable a **Service Request** to occur then an invalid command is received. The bit is cleared after the controller sends a **Status** command **(U0X)** and reads the status string from the **Digital488.**

DIO4 The status of **DIO4** is determined after the **Test** command **(T0X)** is sent to the **Digital488.**   If the self-test passes, the **DIO4** bit will remain a zero. If the self-test fails, **DIO4** will be set to a logic 1.

The **M8** command will cause a **Service Request** to be generated in addition to **DIO4** being set if the self-test fails. The **DIO4** bit is cleared after the controller sends a **Status** command **(U0X)** and reads the status string from the **Digital488.**

DIO5	The **DIO5** bit is set after an entire command string has been received and processed by the **Digital488.** The bit is clear while the **Digital488** is processing commands that have been received from the controller. When used with the **M16** command, a **Service Request** will also be generated when the **DIO5** bit is set. An **Execute** command **(X)** must be received before the **DIO5** bit can be cleared.
DIO6	**DIO6** is not used, and is always a logic zero.
DIO7	When the **Digital488** generates a **Service Request,** the **DIO7** will be set to a logic one. This is used by the controller to determine that the **Service Request** was generated by the **Digital488.**
DIO8	**DIO8** is not used, and is always a logic zero.

Example:

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"M4X"`	select SRQ on Bus error
`OUTPUT 718;"F7X"`	send an invalid bus command.
	ERROR and SRQ LEDs should illuminate
`SPOLL (718)`	display should be `84` (64+16+4)

Sixty-four denotes the **Digital488** was the source of the SRQ. Sixteen denotes the **Digital488** is READY for more commands. Four denotes a Bus error.

When serial polled, the SRQ LED will turn off.

Status — *Un*

The **Status** command (**U0**) will cause the **Digital488** to send the status message when next addressed to talk. The status of the **Digital488** may be read at any time without interfering with normal operation. Any error conditions are cleared after the status string is read by the controller. The **Status** command (**Un**) also enables the controller to read any single bit from the I/O ports **(U1** through **U40).**

U0	Send the **Digital488** status when next addressed to talk
Un	Send the status of bit n (1 thru 40) when next addressed to talk

The format of the status byte returned by the **Digital488** after receiving a **U0** command is as follows:

```
*.*C#E#F#G#I###K#M###P#R#Y#
```

where each `#` equals the number corresponding to that command. The leading information `*.*` is the revision level of the **Digital488** firmware.

Example:

`DIM A$[50]`	dimension A$
`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"U0X"`	send `U0` to the **Digital488**
`ENTER 718; A$`	read the status byte
`DISP A$`	display = `1.0C0E0F0G0I000K0M000P0R0Y0`

The status returned after receiving a **U1** through **U40** is an ASCII character '1' or '0', depending on the level of the line, and the state of the **Invert** command (**I16).**

`CLEAR 718`	reset the **Digital488**
`OUTPUT 718;"U22X"`	request the status of bit 22
`ENTER 718;A$`	read the status bit
`DISP A$`	display shows a `0` (dependent on the signal applied to the input)

Below is a summary of the **Status** (**U0**) information.

C#	**Configuration**
C0	All ports are inputs
C1	Port 1 is an output, ports 2 thru 5 are inputs
C2	Ports 1 and 2 are outputs, ports 3 thru 5 are inputs
C3	Ports 1 thru 3 are outputs, ports 4 and 5 are inputs
C4	Ports 1 thru 4 are outputs, port 5 is an input
C5	All ports are outputs

E#	**Error Message**
0	No error
1	Unrecognized command (ex. W3)
2	Illegal command option (ex. F8)
3	Conflict (attempt to output data to an input port)
4	ROM error
5	RAM error

F#	**Data Format**
F0	Hexadecimal
F1	ASCII
F2	Binary
F3	Decimal
F4	High Speed Binary

I###	**Invert Control Lines**
I0	All control and data lines are active high
I1	**Inhibit** output is active low
I2	**Trigger** output is active low
I4	**Data Strobe** output is active low
I8	**Clear** output is active low
I16	**Data** is active low
I32	**EDR** input is falling edge sensitive
I64	**Service** input is falling edge sensitive

Note: the status indication reflects the sum of all received **Invert** commands.

<u>K#</u>	**<u>End Or Identify</u>**
K0	EOI enabled
K1	EOI disabled

<u>M##</u>	**<u>Service Request</u>**
M0	SRQ is disabled
M1	SRQ on **Service** input transition
M2	SRQ on **EDR** input transition
M4	SRQ on Bus error
M8	SRQ on Test error
M16	SRQ on Ready

Note: the status indication reflects the sum of all received **Service Request** commands.

<u>P#</u>	**<u>Selected Port</u>**
P0	All ports selected
P1	Port 1 selected
P2	Port 2 selected
P3	Port 3 selected
P4	Port 4 selected
P5	Port 5 selected

<u>R#</u>	**<u>Data Ready</u>**
R0	Data is not latched, but is read when Digital 488 is addressed to talk
R1	Data is latched on **EDR** transition

<u>T#</u>	**<u>Test LED</u>**
T0	Perform RAM and ROM test

<u>Y#</u>	**<u>Terminator</u>**
Y0	CR LF
Y1	LF CR
Y2	CR only
Y3	LF only

Terminator — *Yn*

The IEEE 488 bus terminator defaults at power-on to the settings on Switch S1. It also may be programmed for any combination of Carriage Return (CR) and Line Feed (LF). The Y0 mode is the most commonly accepted terminator, CR-LF. Y1 reverses the sequence to send LF-CR. Y2 sends CR only and Y3 sends LF only.

Y0	CR LF
Y1	LF CR
Y2	CR only
Y3	LF only

Example:

```
CLEAR 718
OUTPUT 718;"Y3X"
```
select line feed terminator

The **Test** command is used to verify hardware and LED operation.

T0 Perform RAM and ROM test

The **T0** command will cause the **Digital488** to initiate a ROM/RAM test. If the test is successful, all LEDs will flash for one-half second. If a test fails, the **Error** LED will remain illuminated. Use the **Status** command to determine the cause of the self-test error.

Example:

`CLEAR 718`	reset the Digital488
`OUTPUT 718;"T0X"`	send self test command

Notes

History

The **IEEE 488** bus is an instrumentation communication bus adopted by the Institute of Electrical and Electronic Engineers in 1975 and revised in 1978. The **Digital488** conforms to this most recent revision designated **IEEE 488-1978**.

Prior to the adoption of this standard, most instrumentation manufacturers offered their own versions of computer interfaces. This placed the burden of system hardware design on the end user. If his application required the products of several different manufacturers, then he might need to design several different hardware and software interfaces. The popularity of the **IEEE 488** interface (sometimes called the **G**eneral **P**urpose **I**nterface **B**us or **GPIB**) is due to the total specification of the electrical and mechanical interface as well as the data transfer and control protocols. The use of the **IEEE 488** standard has moved the responsibility of the user from design of the interface to design of the high level software that is specific to the measurement application.

General Structure

The main purpose of the **GPIB** is to transfer information between two or more devices. A device can be either an instrument or a computer. Before any information transfer can take place, it is first necessary to specify which will do the talking (send data) and which devices will be allowed to listen (receive data). The decision of who will talk and who will listen usually falls on the **System Controller** which is, at power on, the **Active Controller**.

The **System Controller** is similar to a committee chairman. On a well-run committee, only one person may speak at a time and the chairman is responsible for recognizing members and allowing them to have their say. On the bus, the device which is recognized to speak is the **Active Talker**. There can only be one Talker at a time if the information transferred is to be clearly understood by all. The act of "giving the floor" to that device is called **Addressing to Talk**. If the committee chairman can not attend the meeting, or if other matters require his attention, he can appoint an acting chairman to take control of the roceedings. For the **GPIB**, this device becomes the **Active Controller**.

At a committee meeting, everyone present usually listens. This is not the case with the **GPIB**. The **Active Controller** selects which devices will listen and commands all other devices to ignore what is being transmitted. A device is instructed to listen by being **Addressed to Listen**. This device is then referred to as an **Active Listener**. Devices which are to ignore the data message are instructed to **Unlisten**.

The reason some devices are instructed to **Unlisten** is quite simple. Suppose a college instructor is presenting the day's lesson. Each student is told to raise their hand if the instructor has exceeded their ability to keep up while taking notes. If a hand is raised, the instructor stops his discussion to allow the slower students the time to catch up. In this way, the instructor is certain that each and every student receives all the information he is trying to present. Since there are many students in the classroom, this exchange of information can be very slow. In fact, the rate of information transfer is no faster than the rate at which the slowest note-taker can keep up. The instructor, though, may have a message for one particular student. The instructor tells the rest of the class to ignore this message (**Unlisten**) and tells it to that one student at a rate which he can understand. This information transfer can then happen much quicker, because it need not wait for the slowest student.

The **GPIB** transfers information in a similar way. This method of data transfer is called **handshaking**. More on this later.

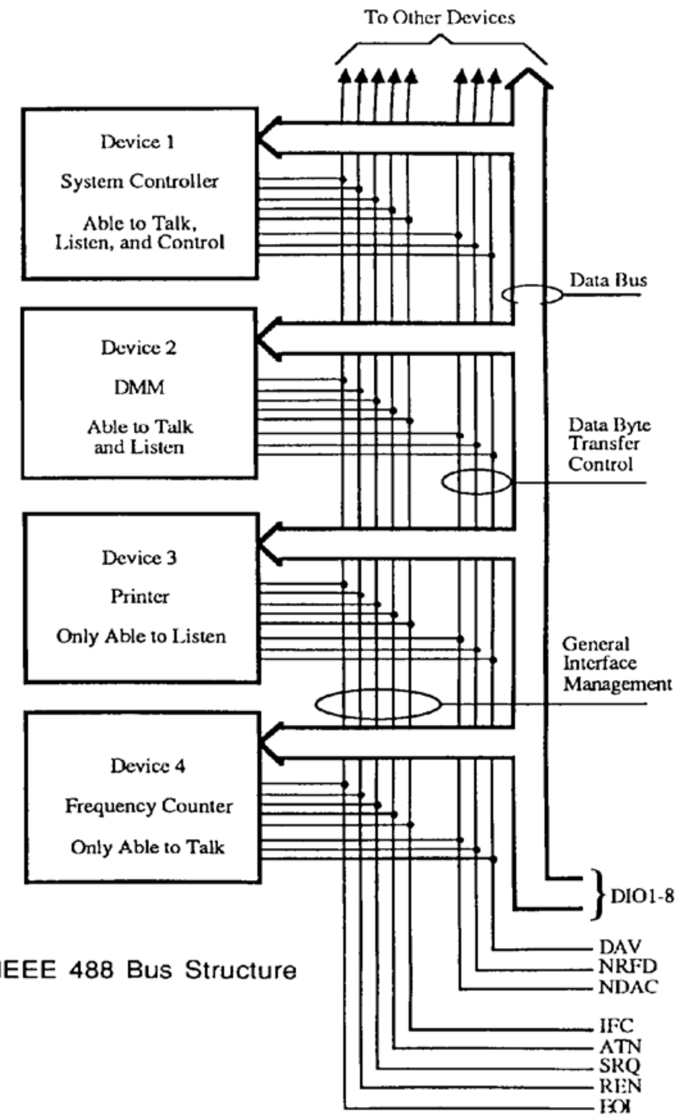
For data transfer on the **IEEE 488**, the **Active Controller** must…

a) **Unlisten** all devices to protect against eavesdroppers.

b) Designate who will **talk** by **addressing** a device to **talk**.

c) Designate all the devices that are to **listen** by **addressing** those devices to **listen**.

d) Indicate to all devices that the data transfer can take place.



IEEE 488 Bus Structure

Send It To My Address

In the previous discussion, the terms **Addressed to Talk** and **Addressed to Listen** were used. These terms require some clarification.

The **IEEE 488** standard permits up to 15 devices to be configured within one system. Each of these devices must have a unique address to avoid confusion. In a similar fashion, every building in town has a unique address to prevent one home from receiving another home's mail. Exactly how each device's address is set is specific to the product's manufacturer. Some are set by DIP switches in hardware, others by software. Consult the manufacturer's instructions to determine how to set the address.

Addresses are sent with **universal** (**multiline**) commands from the **Active Controller**. These commands include **My Listen Address** (MLA), **My Talk Address** (MTA), **Talk Address Group** (TAG), and **Listen Address Group** (LAG).

Bus Management Lines

Five hardware lines on the **GPIB** are used for bus management. Signals on these lines are often referred to as **uniline** (single line) commands. The signals are active low, i.e. a low voltage represents a logic "1" (asserted), and a high voltage represents a logic "0" (unasserted).

Attention (ATN)

ATN is one of the most important lines for bus management. If Attention is asserted, then the information contained on the data lines is to be interpreted as a multiline command. If it is not, then that information is to be interpreted as data for the **Active Listener**s. The **Active Controller** is the only bus device that has control of this line.

Interface Clear (IFC)

The **IFC** line is used only by the **System Controller**. It is used to place all bus devices in a known state. Although device configurations vary, the **IFC** command usually places the devices in the Talk and Listen Idle states (neither **Active Talker** nor **Active Listener**).

Remote Enable (REN)

When the **System Controller** sends the **REN** command, bus devices will respond to remote operation. Generally, the **REN** command should be issued before any bus programming is attempted.
Only the **System Controller** has control of the **Remote Enable** line.

End or Identify (EOI)

The **EOI** line is used to signal the last byte of a multibyte data transfer. The device that is sending the data asserts **EOI** during the transfer of the last data byte. The **EOI** signal is not always necessary as the end of the data may be indicated by some special character such as carriage return.

The **Active Controller** also uses **EOI** to perform a **Parallel Poll** by simultaneously asserting **EOI** and **ATN**.

Service Request (SRQ)

When a device desires the immediate attention of the **Active Controller,** it asserts **SRQ**. It is then the Controller's responsibility to determine which device requested service. This is accomplished with a **Serial Poll** or a **Parallel Poll**.

Handshake Lines

The **GPIB** uses three handshake-lines in an "I'm ready - Here's the data - I've got it" sequence. This handshake protocol assures reliable data transfer, at the rate determined by the slowest Listener. One line is controlled by the Talker, while the other two are shared by all Active Listeners. The handshake lines, like the other **IEEE 488** lines, are active low.

Data Valid (DAV)

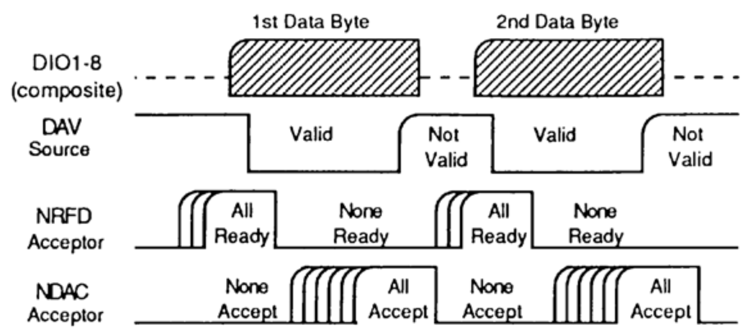
The **DAV** line is controlled by the **Talker**. The **Talker** verifies that **NDAC** is asserted (active low) which indicates that all Listeners have accepted the previous data byte transferred. The **Talker** then outputs data on the bus and waits until **NRFD** is unasserted (high) which indicates that all Addressed Listeners are ready to accept the information. When **NRFD** and **NDAC** are in the proper state, the **Talker** asserts **DAV** (active low) to indicate that the data on the bus is valid.

Not Ready for Data (NRFD)

This line is used by the **Listeners** to inform the **Talker** when they are ready to accept new data. The **Talker** must wait for each **Listener** to unassert the line (high) which they will do at their own rate when they are ready for more data. This assures that all devices that are to accept the information are ready to receive it.

Not Data Accepted (NDAC)

The **NDAC** line is also controlled by the **Listeners**. This line indicates to the **Talker** that each device addressed to listen has accepted the information. Each device releases **NDAC** (high) at its own rate, but the **NDAC** will not go high until the slowest Listener has accepted the data byte.



IEEE Bus Handshaking

Data Lines

The **GPIB** provides eight data lines for a bit parallel/byte serial data-transfer. These eight data lines use the convention of **DIO1** through **DIO8** instead of the binary designation of **D0** to **D7**. The data lines are bi-directional and are active low.

Multiline Commands

Multiline (bus) commands are sent by the Active Controller over the data bus with ATN asserted. These commands include addressing commands for talk, listen, Untalk and Unlisten.

Go To Local (GTL)

This command allows the selected devices to be manually controlled. ($01)

Listen Address Group (LAG)

There are 31 (0 to 30) listen addresses associated with this group. The 3 most significant bits of the data bus are set to 001 while the 5 least significant bits are the address of the device being told to listen.

Unlisten (UNL)

This command tells all bus devices to Unlisten. The same as Unaddressed to Listen. ($3F)

Talk Address Group (TAG)

There are 31 (0 to 30) talk addresses associated with this group. The 3 most significant bits of the data bus are set to 010 while the 5 least significant bits are the address of the device being told to talk.

Untalk (UNT)

This command tells bus devices to Untalk. The same as Unaddressed to Talk. ($5F)

Local Lockout (LLO)

Issuing the **LLO** command prevents manual control of the instrument's functions. ($11)

Device Clear (DCL)

This command causes all bus devices to be initialized to a pre-defined or power up state. ($14)

Selected Device Clear (SDC)

This causes a single device to be initialized to a pre-defined or power up state. ($04)

Serial Poll Disable (SPD)

The **SPD** command disables all devices from sending their Serial Poll status byte. ($19)

Serial Poll Enable (SPE)

A device which is Addressed to Talk will output its Serial Poll status byte after **SPE** is sent and **ATN** is unasserted. ($18)

Group Execute Trigger (GET)

This command usually signals a group of devices to begin executing a triggered action. This allows actions of different devices to begin simultaneously. ($08)

Take Control (TCT)

This command passes bus control responsibilities from the current **Controller** to another device, which has the ability to control. ($09)

Secondary Command Group (SCG)

These are any one of the 32 possible commands (0 to 31) in this group. They must immediately follow a talk or listen address. ($60 to $7F)

Parallel Poll Configure (PPC)

This configures devices capable of performing a **Parallel Poll** as to which data bit they are to assert in response to a **Parallel Poll**. ($05)

Parallel Poll Unconfigure (PPU)

This disables all devices from responding to a **Parallel Poll**. ($15)

More On Service Requests

Most of the commands covered, both uniline and multiline, are the responsibility of the Active Controller to send and the bus devices to recognize. Most of these happen routinely by the interface and are totally transparent to the system programmer. Other commands are used directly by the user to provide optimum system control. Of the uniline commands, SRQ is very important to the test system and the software designer has easy access to this line by most devices. Service Request is the method by which a bus device can signal to the Controller that an event has occurred. It is similar to an interrupt in a microprocessor-based system.

Most intelligent bus peripherals have the ability to assert SRQ. A DMM might assert it when its measurement is complete, if its input is overloaded or for any of an assortment of reasons. A power supply might SRQ if its output has current limited. This is a powerful bus feature that removes the burden from the System Controller to periodically inquire, "Are you done yet?" Instead, the Controller says, "Do what I told you to do and let me know when you're done" or "Tell me when something is wrong."

Since SRQ is a single line command, there is no way for the Controller to determine which device requested the service without additional information. This information is provided by the multiline commands for Serial Poll and Parallel Poll.

Serial Poll

Suppose the **Controller** receives a service request. For this example, let's assume there are several devices that could assert **SRQ**. The **Controller** issues an **SPE** (Serial Poll enable) command to each device sequentially. If any device responds with DIO7 asserted it indicates to the **Controller** that it was the device that asserted **SRQ**. Often times the other bits will indicate why the device wanted service.
This **Serial Polling** sequence, and any resulting action, is under control of the software designer.

Parallel Poll

The **Parallel Poll** is another way the **Controller** can determine which device requested service.
It provides the who but not necessarily the why. When bus devices are configured for Parallel Poll, they are assigned one bit on the data bus for their response. By using the Status bit, the logic level of the response can be programmed to allow logical OR/AND conditions on one data line by more than one device.
When **SRQ** is asserted, the **Controller** (under user's software) conducts a **Parallel Poll**. The **Controller** must then analyze the eight bits of data received to determine the source of the request. Once the source is determined, a **Serial Poll** might be used to determine the why.

Of the two polling types, the **Serial Poll** is the most popular due to its ability to determine the who and why. In addition, most devices support **Serial Poll** only.

Service Information 5

Factory Service

IOtech maintains a factory service center in Cleveland, Ohio. If problems are encountered in using the Digital488 you should first telephone the factory. Many problems can be resolved by discussing the problems with our applications department. If the problem cannot be solved by this method, you will be instructed as to the proper return procedure.

Theory of Operation

The Heart of the **Digital488** is a 6809 microprocessor [U101] supported by 8K bytes of firmware EPROM [U102 (2764)] and 8K bytes of static RAM [U103 (6264)]. A Versatile Interface Adapter [U104 (65B22)] is used to generate real-time interrupts for the firmware operating system. The front panel annunciators are also driven by U104 through an inverter [U113 (74LS04)].

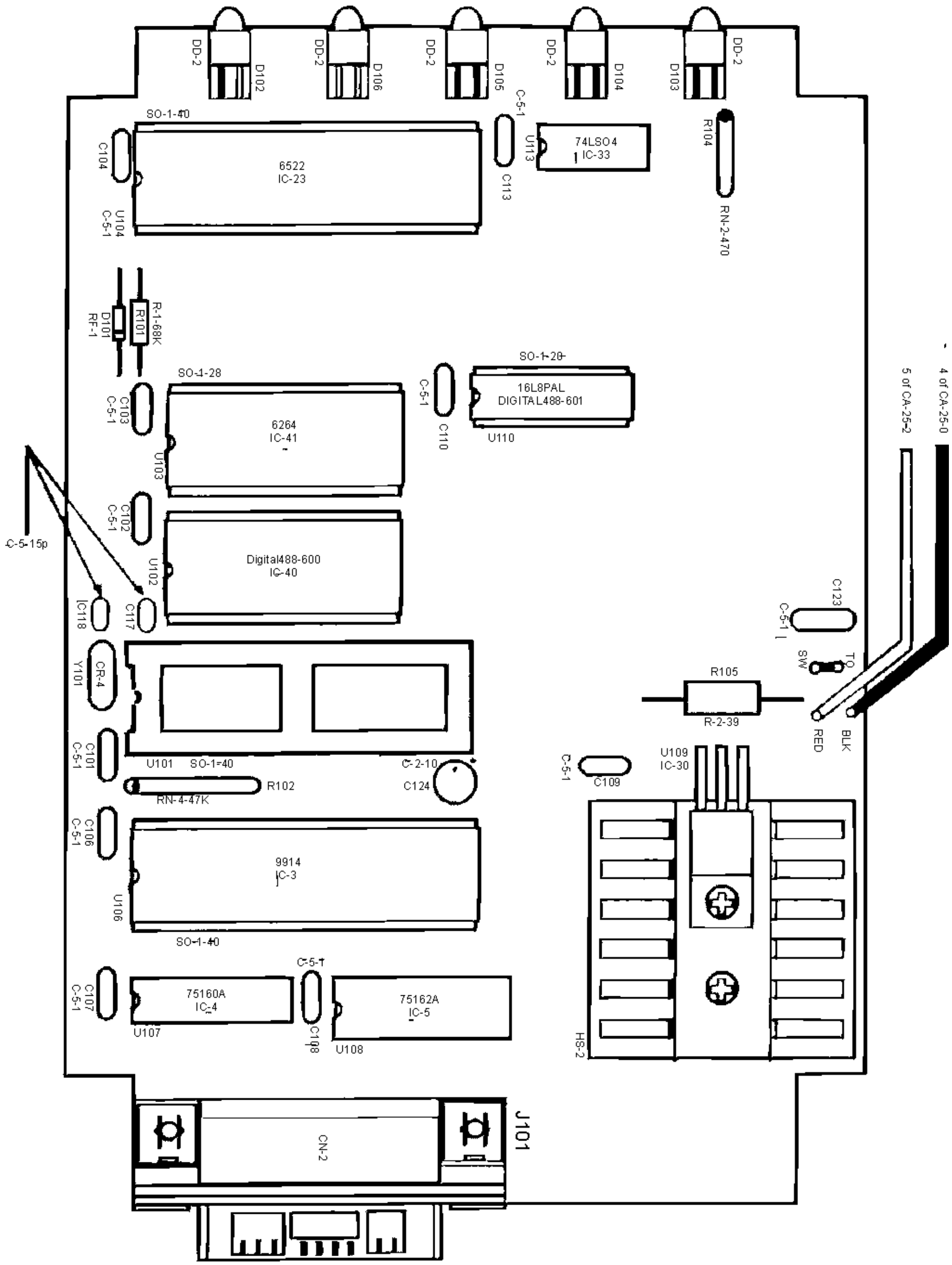
The IEEE 488 bus interface is accomplished by a TMS9914A [U106] controller with drivers U107 and U108. The digital I/O ports are controlled by 'PIA's [U202-U204 (68B21)]. SW1 is read through one port of U204.

Power is supplied by an external unregulated 9 volt wall mount supply. Regulation to the required +5 volts is provided by U109 [7805]. Decoding of the microprocessor address space is accomplished with a Programmable Logic Array [U110 (16L8)]. The Memory space allocation is...

Address	Device	Part Number	Function
$6000-$7FFF	U103	6264	Static RAM
$9200-$9204	U202	6821	Digital I/O
$9400-$9404	U203	6821	Digital I/O
$9800-$9804	U204	6821	Digital I/O
$A000-$A007	U106	TMS9914A	IEEE Controller
$B000-$B00F	U104	R65C22	VIA
$E000-$FFFF	U102	2764	Programmed EPROM

Digital488 Mother Board

Component Layout



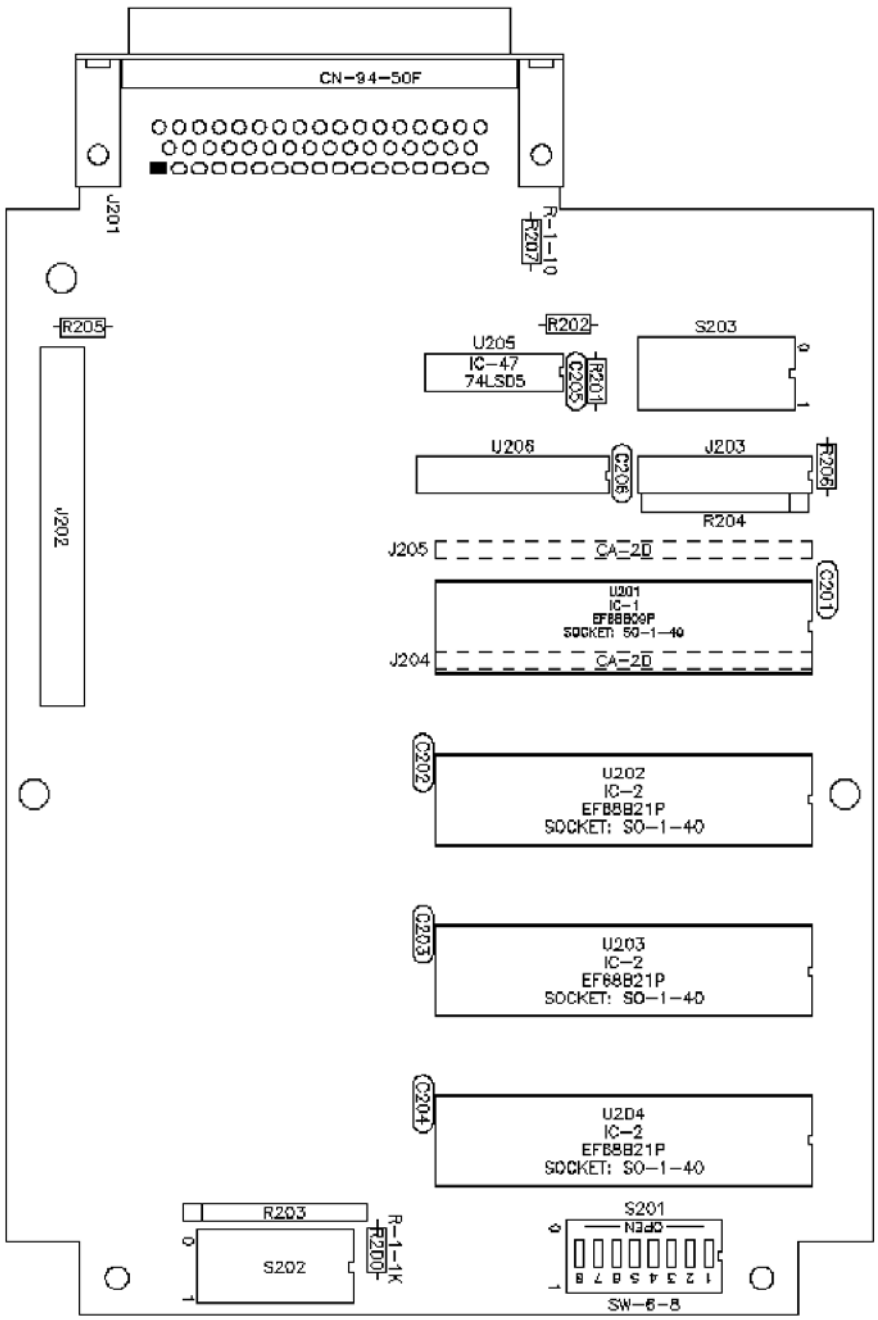
Digital488 Motherboard, Replaceable Parts

Schematic	Part Number	Description
C101-C108	C-5-.1	Ceramic, 25v
C110,C113	C-5-.1	Ceramic, 25v
C117,C118	C-5-15p	Ceramic, 25v
C124	C-2-10	Electrolytic, 25v
C123	C-5-1	Ceramic, 25v
D101	RF-1	Small Signal Diode
D102-D106	DD-2	Red PC Mount
J101	CN-2	IEEE 488 Connector
R101	R-1-68K	68K½, 1/4w carbon
R102	RN-4-4.7K	4.7K½ x 7 SIP
R104	RN-2-470	470½ x 5 SIP
R105	R-2-39	39½, 1w carbon
U102	Digital488-600	Programmed EPROM
U103	IC-41	6264-15 8K x 8 CMOS SRAM
U104	IC-23	65B22 Versatile Interface Adapter
U106	IC-3	TMS9914ANL IEEE Controller
U107	IC-4	SN75160BN IEEE Driver
U108	IC-5	SN75162BN IEEE Driver
U110	Digital488-601	Programming Equation - 16L8 PAL
U113	IC-33	74LS04 Hex Inverter
U109	IC-30	LM7805CT Regulator - +5v
U201	IC-1	MC68B09P Microprocessor
Y101	CR-5	8.0000 MHz Crystal

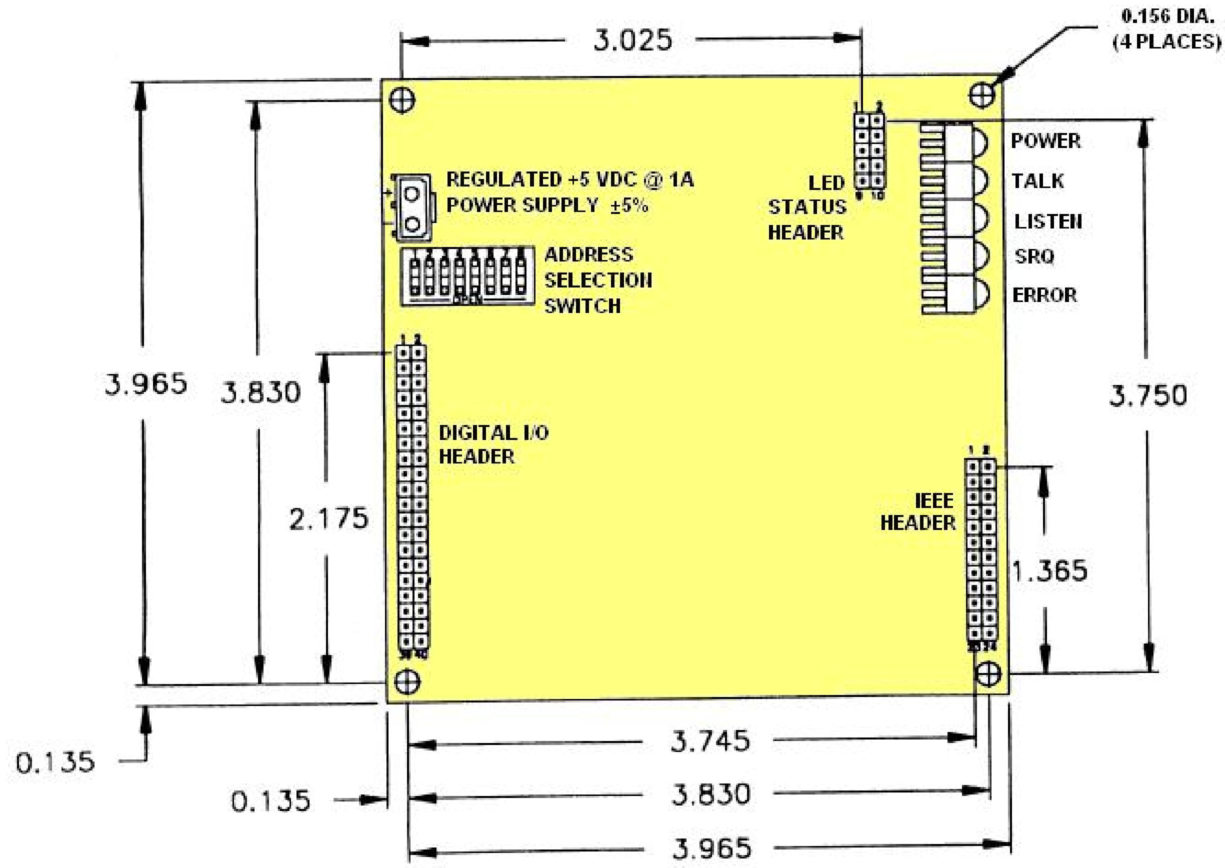
Digital488 I/O Board

Component Layout and Replaceable Parts

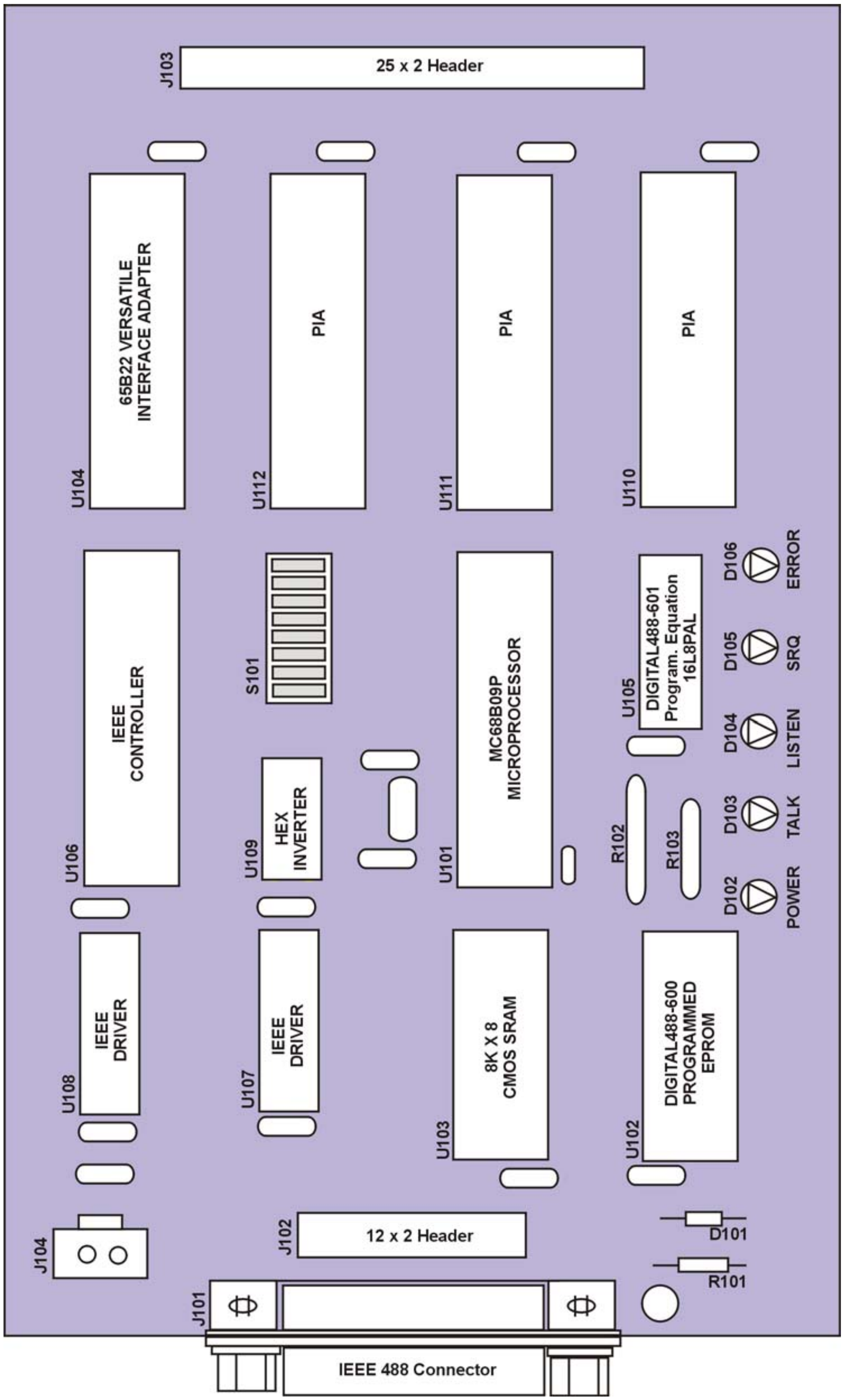
Schematic	Part Number	Description
C201-C205	C-5-.1	Ceramic, 25v
C123	C-5-1	Ceramic, 25v
R201-R202	R-1-1K	1K½, 1/4w carbon
R206	R-1-1K	1K½, 1/4w carbon
S201	SW-6-8	8 Pole DIP
U201	IC-1	MC68B09P Microprocessor
U202-U204	IC-2	68B21 PIA
U205	IC-47	74LS05
U206	IC-32	74LS375



Component Layout



Component Layout



Digital488/OEM, Replaceable Parts

Schematic	Part Number	Description
C101-C112	C-5-.1	Ceramic, 25v
C113	C-5-1	Ceramic, 25v
C114,C115	C-5-15p	Ceramic, 25v
C116	C-2-10	Electrolytic, 25v
D101	RF-1	Small Signal Diode
D102-D106	DD-2	Red PC Mount
J101	CN-2	IEEE 488 Connector
J102	CN-5-12	12 x 2 0.1" Header
J103	CN-5-25	25 x 2 0.1" Header
J104	CN-32-2	2 position Mate-N-Loc Socket
R101	R-1-68K	68K½, 1/4w carbon
R102	RN-4-4.7K	4.7K½ x 7 SIP
R103	RN-2-470	470½ x 5 SIP
S101	SW-6-8	8 Pole DIP
U101	IC-1	MC68B09P Microprocessor
U102	Digital488-600	Programmed EPROM
U103	IC-41	6264-15 8K x 8 CMOS SRAM
U104	IC-23	65B22 Versatile Interface Adapter
U105	Digital488-601	Programming Equation - 16L8 PAL
U106	IC-3	TMS9914ANL IEEE Controller
U107	IC-4	SN75160BN IEEE Driver
U108	IC-5	SN75162BN IEEE Driver
U109	IC-33	74LS04 Hex Inverter
U110-U112	IC-2	68B21 PIA
Y101	CR-5	8.0000 MHz Crystal

Notes

Appendix A

Command Summary

Command	Code	Description
Bit Set	An	Set bit n (1 thru 40)
Bit Clear	Bn	Clear bit n (1 thru 40)
Bus Output	G0	Input and Output port data sent on talk
	G1	Only Input port data sent on talk
	G2	Only Output port data sent on talk
Configure	C0	All ports are inputs
	C1	Port 1 is an output, ports 2 thru 5 are inputs
	C2	Ports 1 and 2 are outputs, ports 3 thru 5 are inputs
	C3	Ports 1 thru 3 are outputs, ports 4 and 5 are inputs
	C4	Ports 1 thru 4 are outputs, port 5 is an input
	C5	All ports are outputs
Data	Dn..Z	Data to be outputted is entered after "D" and terminated by "Z"
Data Ready	R0	Data is read when system is addressed to talk
	R1	Data is latched on EDR transition
EOI	K0	EOI enabled
	K1	EOI disabled
Execute	X	Execute preceding command string
Format	F0	ASCII Hexadecimal
	F1	ASCII Character
	F2	ASCII Binary
	F3	ASCII Decimal
	F4	Binary
	F5	High Speed Binary
Handshake	H0	Pulse the Clear line
	H1	Pulse the Strobe line
	H2	Pulse the Trigger line
Inhibit	Q0	Clear Inhibit line
	Q1	Set Inhibit line

Command	Code	Description
Invert	I0	All control line outputs are active high
	I1	Inhibit output is active low
	I2	Trigger output is active low
	I4	Data Strobe output is active low
	I8	Clear output is active low
	I16	Data is low true
	I32	EDR input is falling-edge sensitive
	I64	Service input is falling-edge sensitive
Port	P0	All ports selected
	P1	Port 1 selected
	P2	Port 2 selected
	P3	Port 3 selected
	P4	Port 4 selected
	P5	Port 5 selected
SRQ Mask	M0	SRQ is disabled
	M1	SRQ on Service Input transition
	M2	SRQ on EDR input transition
	M4	SRQ on Bus error
	M8	SRQ on Self-test error
	M16	SRQ on Ready
Status	U0	Send Status information when next addressed to talk (*.*C#E#F#G#I###K#M###P#R#Y#)
	Un	Read state of bit n (1 thru 40)
Terminator	Y0	CR LF
	Y1	LF CR
	Y2	CR only
	Y3	LF only
Test	T0	Perform RAM and ROM test

Appendix B

ASCII Code Summary

Decimal Values 00 to 63 – ACG, UCG & LAG

Box Items

Hexadecimal Value	$41 65	Decimal Value
	A	
Bus Message	01	(in center) ASCII Character

Addressed Command Group (ACG)

$00 00 **NUL**	$01 01 **SOH** GTL	$02 02 **STX**	$03 03 **ETX**	$04 04 **EOT** SDC	$05 05 **ENQ**	$06 06 **ACK**	$07 07 **BEL** PPD
$08 08 **BS** GET	$09 09 **HT** TCT	$0A 10 **LF**	$0B 11 **VT**	$0C 12 **FF**	$0D 13 **CR**	$0E 14 **SO**	$0F 15 **SI**

Universal Command Group (UCG)

$10 16 **DLE**	$11 17 **DC1** LLO	$12 18 **DC2**	$13 19 **DC3**	$14 20 **DC4** DCL	$15 21 **NAK** PPU	$16 22 **SYN**	$17 23 **ETB**
$18 24 **CAN** SPE	$19 25 **EM** SPD	$1A 26 **SUB**	$1B 27 **ESC**	$1C 28 **FS**	$1D 29 **GS**	$1E 30 **RS**	$1F 31 **US**

Listen Address Group (LAG)

$20 32 **SP** 00	$21 33 **!** 01	$22 34 **”** 02	$23 35 **#** 03	$24 36 **$** 04	$25 37 **%** 05	$26 38 **&** 06	$27 39 **’** 07
$28 40 **(** 08	$29 41 **)** 09	$2A 42 ***** 10	$2B 43 **+** 11	$2C 44 **,** 12	$2D 45 **-** 13	$2E 46 **.** 14	$2F 47 **/** 15
$30 48 **0** 16	$31 49 **1** 17	$32 50 **2** 18	$33 51 **3** 19	$34 52 **4** 20	$35 53 **5** 21	$36 54 **6** 22	$37 55 **7** 23
$38 56 **8** 24	$39 57 **9** 25	$3A 58 **:** 26	$3B 59 **;** 27	$3C 60 **<** 28	$3D 61 **=** 29	$3E 62 **>** 30	$3F 63 **?** UNL

Decimal Values 64 to 127 – TAG & SCG

Box Items

Hexadecimal Value	$41 65 A 01	Decimal Value
Bus Message		(in center) ASCII Character

<table>
<tr><th colspan="8">Talk Address Group (TAG)</th></tr>
<tr><td>$40 64
@
00</td><td>$41 65
A
01</td><td>$42 66
B
02</td><td>$43 67
C
03</td><td>$44 68
D
04</td><td>$45 69
E
05</td><td>$46 70
F
06</td><td>$47 71
G
07</td></tr>
<tr><td>$48 72
H
08</td><td>$49 73
I
09</td><td>$4A 74
J
10</td><td>$4B 75
K
11</td><td>$4C 76
L
12</td><td>$4D 77
M
13</td><td>$4E 78
N
14</td><td>$4F 79
O
15</td></tr>
<tr><td>$50 80
P
16</td><td>$51 81
Q
17</td><td>$52 82
R
18</td><td>$53 83
S
19</td><td>$54 84
T
20</td><td>$55 85
U
21</td><td>$56 86
V
22</td><td>$57 87
W
23</td></tr>
<tr><td>$58 88
X
24</td><td>$59 89
Y
25</td><td>$5A 90
Z
26</td><td>$5B 91
[
27</td><td>$5C 92
\
28</td><td>$5D 93
]
29</td><td>$5E 94
^
30</td><td>$5F 95
_
UNT</td></tr>
<tr><th colspan="8">Secondary Command Group (SCG)</th></tr>
<tr><td>$60 96
`
00</td><td>$61 97
a
01</td><td>$62 98
b
02</td><td>$63 99
c
03</td><td>$64 100
d
04</td><td>$65 101
e
05</td><td>$66 102
f
06</td><td>$67 103
g
07</td></tr>
<tr><td>$68 104
h
08</td><td>$69 105
i
09</td><td>$6A 106
j
10</td><td>$6B 107
k
11</td><td>$6C 108
l
12</td><td>$6D 109
m
13</td><td>$6E 110
n
14</td><td>$6F 111
o
15</td></tr>
<tr><td>$70 112
p
16</td><td>$71 113
q
17</td><td>$72 114
r
18</td><td>$73 115
s
19</td><td>$74 116
t
20</td><td>$75 117
u
21</td><td>$76 118
v
22</td><td>$77 119
w
23</td></tr>
<tr><td>$78 120
x
24</td><td>$79 121
y
25</td><td>$7A 122
z
26</td><td>$7B 123
{
27</td><td>$7C 124
|
28</td><td>$7D 125
}
29</td><td>$7E 126
~
30</td><td>$7F 127
DEL
31</td></tr>
</table>

ASCII Code Details

Decimal Values 00 to 31 – ACG & UCG Characteristics

ASCII Control Codes (Decimal 00 to 31)				
Dec Value	**Hex Value ($)**	**Character & Abbreviation**	**Name**	**Bus Message**
Addressed Command Group (ACG)				
00	**$00**	**None / NUL**	Null	None
01	**$01**	**^A / SOH**	Start of Header	Go To Local (**GTL**)
02	**$02**	**^B / STX**	Start of Text	None
03	**$03**	**^C / ETX**	End of Text	None
04	**$04**	**^D / EOT**	End of Transmission	Selected Device Clear (**SDC**)
05	**$05**	**^E / ENQ**	Inquiry	None
06	**$06**	**^F / ACK**	Acknowledgement	None
07	**$07**	**^G / BEL**	Bell	Parallel Poll Disable (**PPD**)
08	**$08**	**^H / BS**	Backspace	Group Execute Trigger (**GET**)
09	**$09**	**^I / HT**	Horizontal Tab	Take Control (**TCT**)
10	**$0A**	**^J / LF**	Line Feed	None
11	**$0B**	**^K / VT**	Vertical Tab	None
12	**$0C**	**^L / FF**	Form Feed	None
13	**$0D**	**^M / CR**	Carriage Return	None
14	**$0E**	**^N / SO**	Shift Out	None
15	**$0F**	**^O / SI**	Shift In	None
Universal Command Group (UCG)				
16	**$10**	**^P / DLE**	Data Link Escape	None
17	**$11**	**^Q / DC1**	Device Control 1	Local Lockout (**LLO**)
18	**$12**	**^R / DC2**	Device Control 2	None
19	**$13**	**^S / DC3**	Device Control 3	None
20	**$14**	**^T / DC4**	Device Control 4	Device Clear (**DCL**)
21	**$15**	**^U / NAK**	Negative Acknowledgement	Parallel Poll Unconfig (**PPU**)
22	**$16**	**^V / SYN**	Synchronous Idle	None
23	**$17**	**^W / ETB**	End of Transmission Block	None
24	**$18**	**^X / CAN**	Cancel	Serial Poll Enable (**SPE**)
25	**$19**	**^Y / EM**	End of Medium	Serial Poll Disable (**SPD**)
26	**$1A**	**^Z / SUB**	Substitute	None
27	**$1B**	**^[/ ESC**	Escape	None
28	**$1C**	**^\ / FS**	File Separator	None
29	**$1D**	**^] / GS**	Group Separator	None
30	**$1E**	**^^ / RS**	Record Separator	None
31	**$1F**	**^_ / US**	Unit Separator	None

Note: (1) ASCII control codes are sometimes used to “formalize” a communications session between communication devices. (2) **DC1, DC2, DC3, DC4, FS, GS, RS,** and **US** all have user-defined meanings, and may vary in use between sessions or devices. (3) **DC4** is often used as a general “stop transmission character.” (4) Codes used to control cursor position may be used to control print devices, and move the print head accordingly. However, not all devices support the full set of positioning codes.

Decimal Values 00 to 31 – ACG & UCG Descriptions

ASCII Control Codes (00 to 31)		
Dec	**Name**	**Description**
Addressed Command Group (ACG)		
00	Null (**NUL**)	Space filler character. Used in output timing for some device drivers.
01	Start of Header (**SOH**)	Marks beginning of message header.
02	Start of Text (**STX**)	Marks beginning of data block (text).
03	End of Text (**ETX**)	Marks end of data block (text).
04	End of Transmission (**EOT**)	Marks end of transmission session.
05	Inquiry (**ENQ**)	Request for identification or information.
06	Acknowledgement (**ACK**)	"Yes" answer to questions or "ready for next transmission." Used in asynchronous protocols for timing.
07	Bell (**BEL**)	Rings bell or audible alarm on terminal.
08	Backspace (**BS**)	Moves cursor position back one character.
09	Horizontal Tab (**HT**)	Moves cursor position to next tab stop on line.
10	Line Feed (**LF**)	Moves cursor position down one line.
11	Vertical Tab (**VT**)	Moves cursor position down to next "tab line."
12	Form Feed (**FF**)	Moves cursor position to top of next page.
13	Carriage Return (**CR**)	Moves cursor to left margin.
14	Shift Out (**SO**)	Next characters do not follow ASCII definitions.
15	Shift In (**SI**)	Next characters revert to ASCII meaning.
Universal Command Group (UCG)		
16	Data Link Escape (**DLE**)	Used to control transmissions using "escape sequences."
17	Device Control 1 (**DC1**)	Not defined. Normally used for ON controls.
18	Device Control 2 (**DC2**)	Usually user-defined.
19	Device Control 3 (**DC3**)	Not defined. Normally used for OFF controls.
20	Device Control 4 (**DC4**)	Usually user-defined.
21	Negative Acknowledgement (**NAK**)	"No" answer to questions or "errors found, re-transmit." Used in asynchronous protocols for timing.
22	Synchronous Idle (**SYN**)	Sent by asynchronous devices when idle to insure sync.
23	End of Transmission Block (**ETB**)	Marks block boundaries in transmission.
24	Cancel (**CAN**)	Indicates previous transmission should be disregarded.
25	End of Medium (**EM**)	Marks end of physical media, as in paper tape.
26	Substitute (**SUB**)	Used to replace a character known to be wrong.
27	Escape (**ESC**)	Marks beginning of an Escape control sequence.
28	File Separator (**FS**)	Marker for major portion of transmission.
29	Group Separator (**GS**)	Marker for submajor portion of transmission.
30	Record Separator (**RS**)	Marker for minor portion of transmission.
31	Unit Separator (**US**)	Marker for most minor portion of transmission.

Note: (1) ASCII control codes are sometimes used to "formalize" a communications session between communication devices. (2) **DC1, DC2, DC3, DC4, FS, GS, RS,** and **US** all have user-defined meanings, and may vary in use between sessions or devices. (3) **DC4** is often used as a general "stop transmission character." (4) Codes used to control cursor position may be used to control print devices, and move the print head accordingly. However, not all devices support the full set of positioning codes.

Decimal Values 32 to 63 – LAG

ASCII Character Set (Decimal 32 to 63)				
Dec	**Hex**	**Character**	**Name**	**Bus Message**
Listen Address Group (LAG)				
32	**$20**	**<space>**	Space	Bus address 00
33	**$21**	**!**	Exclamation Point	Bus address 01
34	**$22**	**“**	Quotation Mark	Bus address 02
35	**$23**	**#**	Number Sign	Bus address 03
36	**$24**	**$**	Dollar Sign	Bus address 04
37	**$25**	**%**	Percent Sign	Bus address 05
38	**$26**	**&**	Ampersand	Bus address 06
39	**$27**	**‘**	Apostrophe	Bus address 07
40	**$28**	**(**	Opening Parenthesis	Bus address 08
41	**$29**	**)**	Closing Parenthesis	Bus address 09
42	**$2A**	*****	Asterisk	Bus address 10
43	**$2B**	**+**	Plus Sign	Bus address 11
44	**$2C**	**,**	Comma	Bus address 12
45	**$2D**	**-**	Hyphen or Minus Sign	Bus address 13
46	**$2E**	**.**	Period	Bus address 14
47	**$2F**	**/**	Slash	Bus address 15
Listen Address Group (LAG)				
48	**$30**	**0**	Zero	Bus address 16
49	**$31**	**1**	One	Bus address 17
50	**$32**	**2**	Two	Bus address 18
51	**$33**	**3**	Three	Bus address 19
52	**$34**	**4**	Four	Bus address 20
53	**$35**	**5**	Five	Bus address 21
54	**$36**	**6**	Six	Bus address 22
55	**$37**	**7**	Seven	Bus address 23
56	**$38**	**8**	Eight	Bus address 24
57	**$39**	**9**	Nine	Bus address 25
58	**$3A**	**:**	Colon	Bus address 26
59	**$3B**	**;**	Semicolon	Bus address 27
60	**$3C**	**<**	Less Than Sign	Bus address 28
61	**$3D**	**=**	Equal Sign	Bus address 29
62	**$3E**	**>**	Greater Than Sign	Bus address 30
63	**$3F**	**?**	Question Mark	Unlisten (**UNL**)

Decimal Values 64 to 95 – TAG

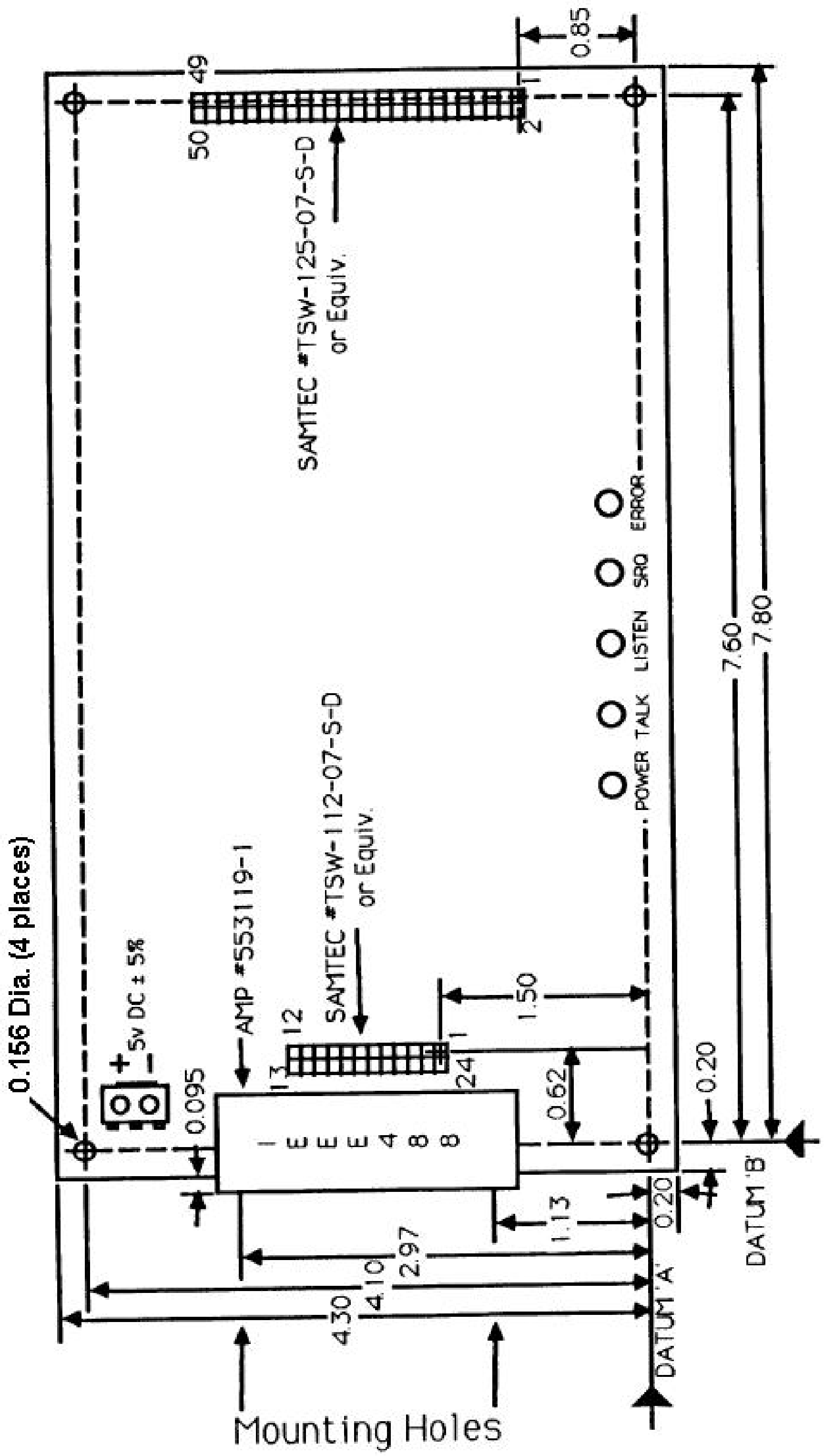
ASCII Character Set (Decimal 64 to 95)				
Dec	**Hex**	**Character**	**Name**	**Bus Message**
Talk Address Group (TAG)				
64	**$40**	**@**	At Sign	Bus address 00
65	**$41**	**A**	Capital A	Bus address 01
66	**$42**	**B**	Capital B	Bus address 02
67	**$43**	**C**	Capital C	Bus address 03
68	**$44**	**D**	Capital D	Bus address 04
69	**$45**	**E**	Capital E	Bus address 05
70	**$46**	**F**	Capital F	Bus address 06
71	**$47**	**G**	Capital G	Bus address 07
72	**$48**	**H**	Capital H	Bus address 08
73	**$49**	**I**	Capital I	Bus address 09
74	**$4A**	**J**	Capital J	Bus address 10
75	**$4B**	**K**	Capital K	Bus address 11
76	**$4C**	**L**	Capital L	Bus address 12
77	**$4D**	**M**	Capital M	Bus address 13
78	**$4E**	**N**	Capital N	Bus address 14
79	**$4F**	**O**	Capital O	Bus address 15
Talk Address Group (TAG)				
80	**$50**	**P**	Capital P	Bus address 16
81	**$51**	**Q**	Capital Q	Bus address 17
82	**$52**	**R**	Capital R	Bus address 18
83	**$53**	**S**	Capital S	Bus address 19
84	**$54**	**T**	Capital T	Bus address 20
85	**$55**	**U**	Capital U	Bus address 21
86	**$56**	**V**	Capital V	Bus address 22
87	**$57**	**W**	Capital W	Bus address 23
88	**$58**	**X**	Capital X	Bus address 24
89	**$59**	**Y**	Capital Y	Bus address 25
90	**$5A**	**Z**	Capital Z	Bus address 26
91	**$5B**	**[**	Opening Bracket	Bus address 27
92	**$5C**	****	Backward Slash	Bus address 28
93	**$5D**	**]**	Closing Bracket	Bus address 29
94	**$5E**	**^**	Caret	Bus address 30
95	**$5F**	**_**	Underscore	Untalk (**UNT**)

Decimal Values 96 to 127 – SCG

ASCII Character Set (96 to 127)				
Dec	**Hex**	**Character**	**Name**	**Bus Message**
Secondary Command Group (SCG)				
96	**$60**	**'**	Grave	Command 00
97	**$61**	**a**	Lowercase A	Command 01
98	**$62**	**b**	Lowercase B	Command 02
99	**$63**	**c**	Lowercase C	Command 03
100	**$64**	**d**	Lowercase D	Command 04
101	**$65**	**e**	Lowercase E	Command 05
102	**$66**	**f**	Lowercase F	Command 06
103	**$67**	**g**	Lowercase G	Command 07
104	**$68**	**h**	Lowercase H	Command 08
105	**$69**	**i**	Lowercase I	Command 09
106	**$6A**	**j**	Lowercase J	Command 10
107	**$6B**	**k**	Lowercase K	Command 11
108	**$6C**	**l**	Lowercase L	Command 12
109	**$6D**	**m**	Lowercase M	Command 13
110	**$6E**	**n**	Lowercase N	Command 14
111	**$6F**	**o**	Lowercase O	Command 15
Secondary Command Group (SCG)				
112	**$70**	**p**	Lowercase P	Command 16
113	**$71**	**q**	Lowercase Q	Command 17
114	**$72**	**r**	Lowercase R	Command 18
115	**$73**	**s**	Lowercase S	Command 19
116	**$74**	**t**	Lowercase T	Command 20
117	**$75**	**u**	Lowercase U	Command 21
118	**$76**	**v**	Lowercase V	Command 22
119	**$77**	**w**	Lowercase W	Command 23
120	**$78**	**x**	Lowercase X	Command 24
121	**$79**	**y**	Lowercase Y	Command 25
122	**$7A**	**z**	Lowercase Z	Command 26
123	**$7B**	**{**	Opening Brace	Command 27
124	**$7C**	**|**	Vertical Line	Command 28
125	**$7D**	**}**	Closing Brace	Command 29
126	**$7E**	**~**	Tilde	Command 30
127	**$7F**	**DEL**	Delete	Command 31

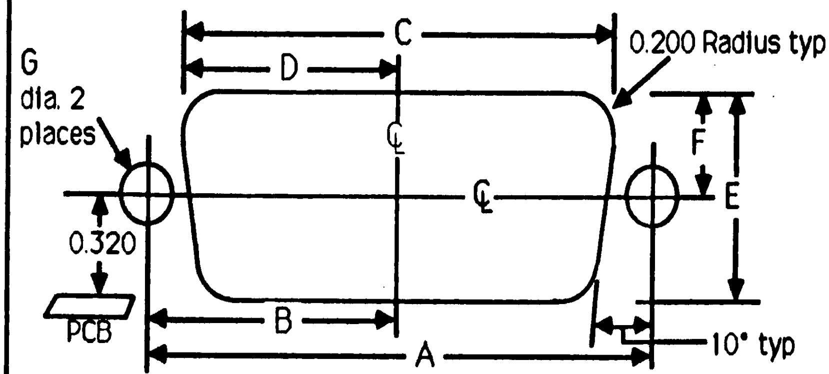
Notes

Digital488/OEM Mechanical Dimensions



Board Dimensions

IEEE Connector Mounting & Hole Punch Out



A	B	C	D	E	F	G
1.842	0.921	1.578	0.789	0.640	0.320	0.194

Note: Does not imply orientation

IEEE Connector Mounting and Hole Punch Out