the smart approach to instrumentation ™

IOtech, Inc.
25971 Cannon Road Cleveland, OH 44146
Phone: (440) 439-4091 Fax: (440) 439-4093
E-mail: sales@iotech.com Internet: http://www.iotech.com

Micro 488/EX User's Manual

Stand Alone Bus Controller

p/n Micro488/EX-901, Rev 2.1

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

FCC Statement

IOtech devices emit radio frequency energy in levels compliant with Federal Communications Commission rules (Part 15) for Class A devices. If necessary, refer to the FCC booklet *How To Identify and Resolve Radio-TV Interference Problems* (stock # 004-000-00345-4) which is available from the U.S. Government Printing Office, Washington, D.C. 20402.

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 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.

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 maintained ISO 9001 certification 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

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Introduction

1.1 Description

The Micro488/EX Stand Alone Bus Controller converts a host RS-232 or RS-422 computer into an IEEE 488 bus talker, listener and controller. The Micro488/EX provides full IEEE 488-1978 bus implementation including advance capabilities such as PASS CONTROL, RECEIVE CONTROL, PARALLEL POLL, SERIAL POLL and SECONDARY ADDRESSING. The device may be located several hundred feet from the host and may control as many as fourteen 488 bus instruments. In the non-controller mode the Micro488/EX converts the host into a bus peripheral for data processing and mass storage. The Micro488/EX interprets simple high level commands sent from the computer's serial port and performs the necessary, and usually complex, bus control and handshaking. The commands and protocol are similar to those used by the Hewlett Packard HP-85 computer.

A unique feature of the Micro488/EX is its macro capability. Macros are mini-program sequences that can be used to collect data for later recall by the serial host computer. Up to 100 different macros can be defined and maintained in non-volatile memory. The data collected by macros can be immediately sent to the serial host or placed into a non-volatile log buffer for later recall. The data collected can also be date and time stamped using the Micro488/EX's internal time-day-date clock. Macros can also be executed as a result of some external event, such as SRQs or Errors.

Additional features provide a transparent IEEE to serial converter and a serial to IEEE pass-thru controller.

As a serial to IEEE 488 converter, the Micro488/EX receives data from a serial host then automatically performs the bus sequences necessary to send this data to the IEEE 488 device. If desired, data can be requested from the IEEE 488 device and returned to the host.

As an IEEE 488 to serial converter, the Micro488/EX is a peripheral to an IEEE 488 controller. Data received from the controller is sent to the serial device and data received from the serial device is buffered for transmission to the IEEE 488 controller. The Micro488/EX can inform the host, by the serial poll status byte, that it has received data from the serial device.

1.2 Available Accessories

Additional accessories that can be ordered for the Micro488/EX 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-11	IBM PC/XT/PS2 to Micro488/EX RS-232 Cable
CA-12	Macintosh 512 to Micro488/EX RS-232 Cable
CA-22	Macintosh II/SE/Plus to Micro488/EX RS-232 Cable
CA-23	IBM AT to Micro488/EX RS-232 Cable
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
ABC488	IEEE 488 ABC switch
Rack488-3	5-1/4" by 19" rack mount for one Micro488/EX
Rack488-4	5-1/4" by 19" rack mount for two Micro488/EX's
127-0920	Additional instruction manual

CAUTION

Please read this manual carefully! If equipment is used in any manner not specified in this manual, the protection provided by the equipment may be impaired.

1.3 Specifications

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:

- C1, C2, C3, C4 and C28 controller subsets. SH1, AH1, T6, TE0, L4, LE0, SR1, RL0, PP0, DC1, DT1, E1

Terminators:

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

Connector:

- Standard IEEE 488 connector with metric studs.

Serial Interface

CAUTION

The RS-232/RS-422 terminal is only for connecting devices having signals at serial communication levels.

Terminal Installation Category:

- **Standard**: Not Applicable.
- **CE**: Category 1.

EIA RS-232C:

- AB, BA, BB, CA, CB.

EIA RS-422A:
- Balanced voltage on TxD and RxD.

Character Set:
- Asynchronous bit serial.

Output Voltage:
- ±5 volts minimum (RS-232C); 3.5 volts typical (RS-422A).

Input Voltage:
- ±3 volts minimum; ±15 volts maximum.

Baud Rate:
- Selectable 110,300,600,1200,1800,2400,3600,4800,7200,9600, and 19200.

Data Format:
- Selectable 7 or 8 data bits; 1 or 2 stop bits; odd, even, mark, space and no parity on transmit.

Duplex:
- Full with Echo/No Echo.

Serial Control:
- Selectable CTS/RTS or XON/XOFF.

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

Connector:
- 25-pin Sub-D male. RS-232C DCE configured.

Clock

Accuracy:
- 1 minute per month typical.

Battery Life:
- 10 years typical.

Information Provided:
- Provides hours, minutes, seconds, day, month, date and year.

General

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 use, 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 m, 0 to 40°C; 80% maximum RH up to 31°C decreasing linearly 4% RH/°C to 40°C.

Data Buffer:

- Approximately 29000 characters total, dynamically allocated and non-volatile. Data maintained for up to 10 years typical.

Controls:

- Power Switch (external), IEEE and Serial parameter switches (internal). Jumper selection of RS-232 or RS-422 operation (internal)

Indicators:

- LED indicators for TALK, LISTEN, SRQ, ERROR and POWER.

Power:

- An external power supply is provided with the Micro488/EX: Input is 105 to 125 VAC or 210 to 250 VAC; 50-60 Hz, 10 VA maximum. External power supply 9 VDC output is to be connected to the Micro488/EX power input marked 10 VDC MAX @ 600 mA.

CAUTION

Do not connect AC power line directly to the Micro488/EX. Otherwise, equipment may be damaged.

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!

1.4 Abbreviations

The following IEEE 488 abbreviations are used throughout this manual.

addr n	IEEE bus address "n"
ATN	Attention line
CA	Controller Active
CO	Controller
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
PE	Peripheral
PPC	Parallel Poll Configure
PPU	Parallel Poll Unconfigure
REN	Remote Enable
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

2.1 Inspection

The Micro488/EX 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. Immediately report any such damage found to the shipping agent. Remember to retain all shipping materials in the event that shipment back to the factory becomes necessary.

Every Micro488/EX is shipped with the following....

• Micro488/EX	IEEE 488 Bus Controller
• 127-0920	Instruction Manual
• Power Supply	TR-2; 115V or TR-2E; 220/230V

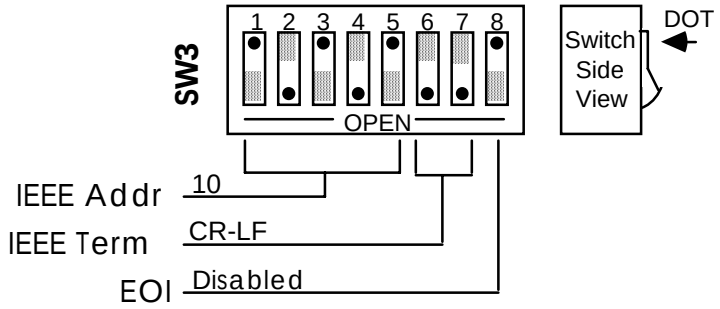
2.2 Configuration

Three DIP switches internal to the Micro488/EX set the configuration of the interface. NOTE: Selectable functions are read ONLY at power-on and should only be set prior to applying power to the interface. The following figures illustrate the factory default conditions which are:

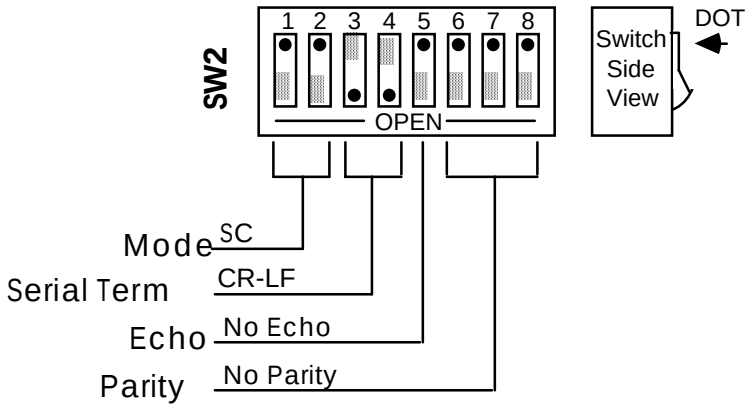
Serial Port:
9600 Baud
8 Data Bits
2 Stop Bits
No Parity
Serial Terminator = CR-LF
Echo Disabled
RTS/CTS Handshake

IEEE:
Mode = System Controller
Address = 10
Bus Terminator = CR-LF; EOI Disabled
Talk-back Enabled

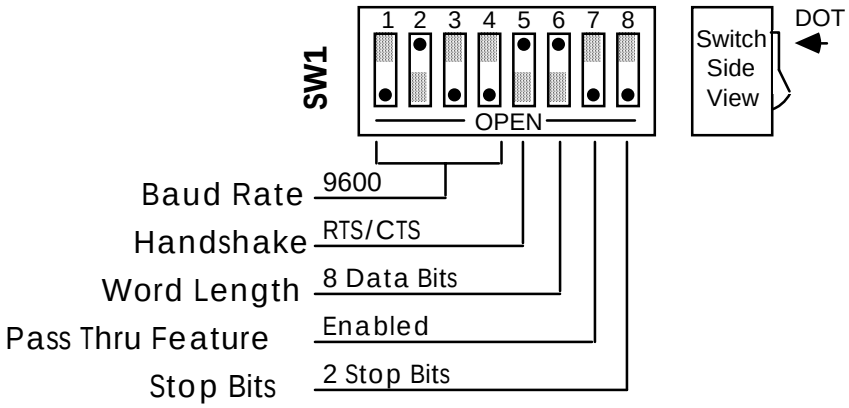
SW3 Factory Default Settings



SW2 Factory Default Settings



SW1 Factory Default Settings



Note that the Micro488/EX comes configured as an IEEE controller. In this mode the Micro488/EX is designed to allow a serial host computer to control up to 14 IEEE 488 devices. This mode of operation is described in detail, along with its command descriptions, in Sections 3, 4 and 5. Theses sections also cover the peripheral mode of operation.

The Micro488/EX can be configured to transparently communicate with a single IEEE peripheral, such as a plotter. This Controller Pass-Thru mode is described in detail in Section 6.

The Micro488/EX may also be configured as a transparent IEEE Pass-Thru Peripheral. As a Pass-Thru Peripheral, the Micro488/EX allows an IEEE controller to communicate with a serial device. The Peripheral Pass-Thru mode of operation is described in detail in Section 7.

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 serial cables prior to disassembly.

WARNING

Never open the Micro488/EX case while it is connected to the AC line. Failure to observe this 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.

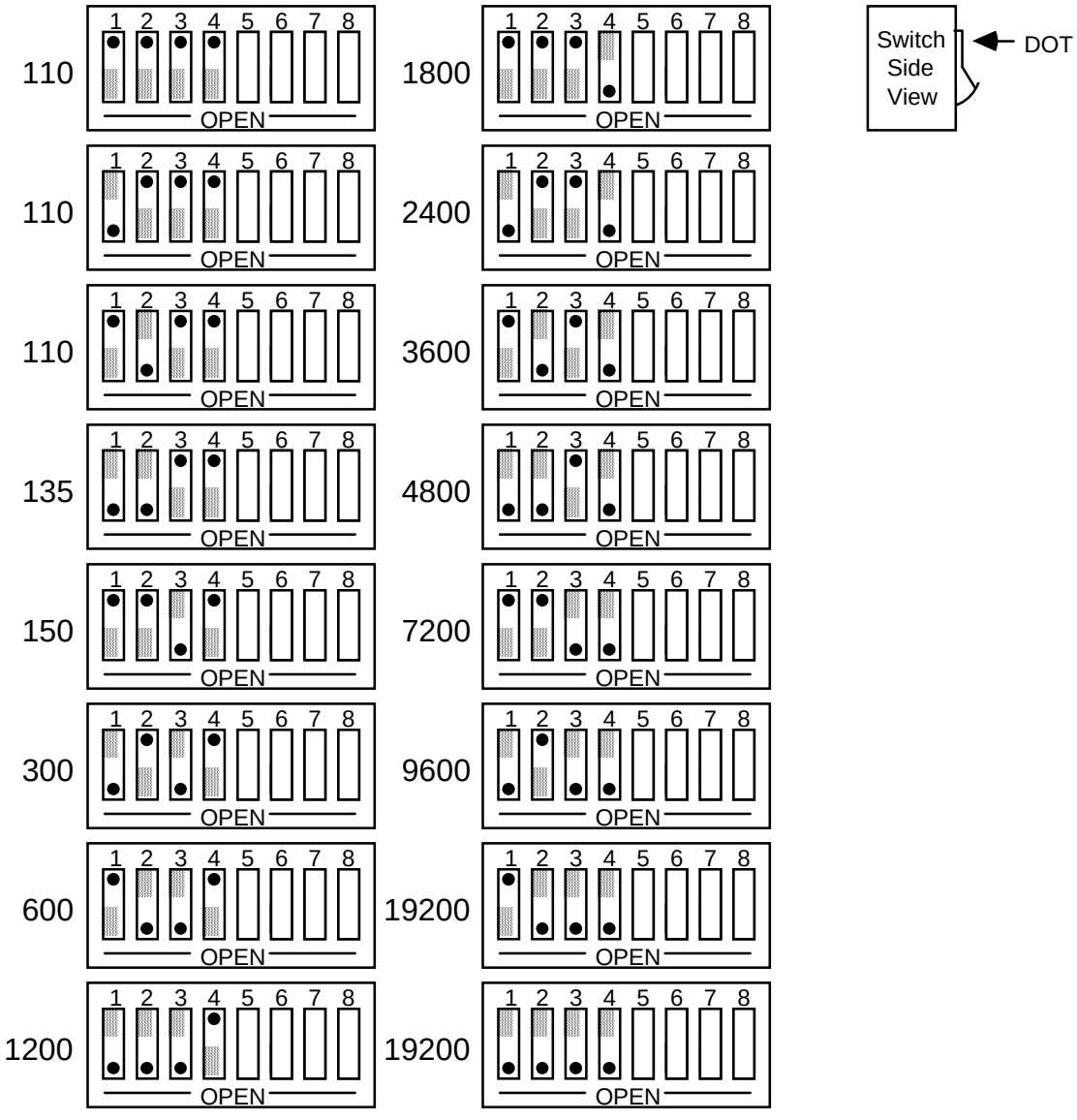
2.3 Serial Port Settings

The first parameters to configure are those that correspond to the RS-232/RS-422 port. These include baud rate, word length, number of stop bits, parity selection and type of serial handshake. Each of these are described in the following sections.

2.3.1 Serial Baud Rate Selection

Baud rate defines the number of serial bits per second transferred into and out of the serial interface. SW1-1 through SW1-4 determine the serial baud rate. The factory default baud rate is 9600 baud. Baud rates may be selected from 110 to 19200 baud. Refer to the following diagram for specific baud rates.

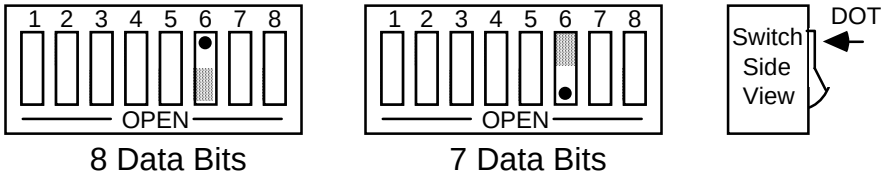
SW1 View for Serial Baud Rate Selection



2.3.2 Serial Word Length Selection - Data Bits

SW1-6 determines the number of data bits, often referred to as word length, for each serial character transmitted or received. The factory default is 8 data bits.

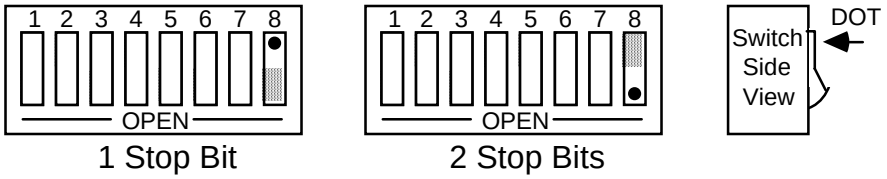
SW1 View of Serial Word Length (Data Bits) Selection



2.3.3 Serial Stop Bit Selection

Switch SW1-8 determines the number of stop bits contained in each serial character transmitted and received. The factory default is 2 stop bits.

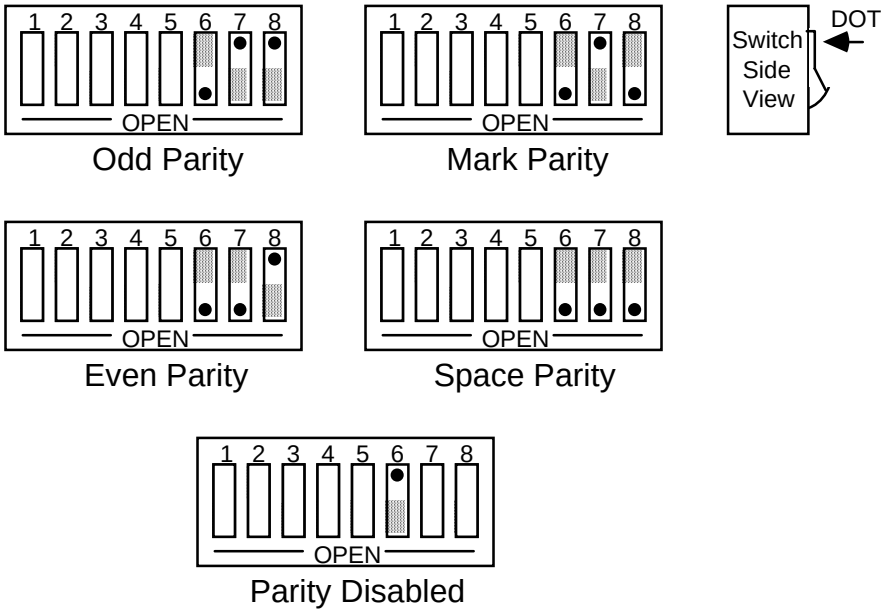
SW1 View for Serial Stop Bit Selection



2.3.4 Serial Parity Selection

Serial Parity is selected with S2-6 through S2-8. The Micro488/EX generates the selected parity during serial transmissions but it does not check parity on data that is received. The factory default is parity disabled.

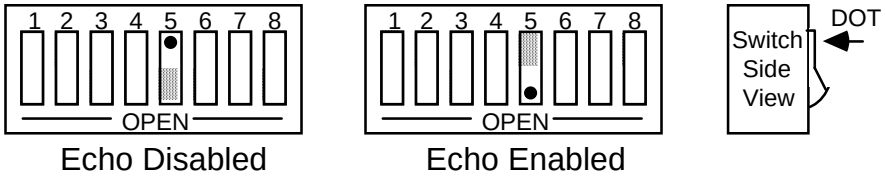
SW2 View for Serial Parity Selection



2.3.5 Serial Echo Selection

Serial data sent to the Micro488/EX will be echoed back to the serial host if SW2-5 is set to the open position. Factory default is Echo Disabled.

SW2 View for Echo Selection



2.3.6 Serial Handshake Selection

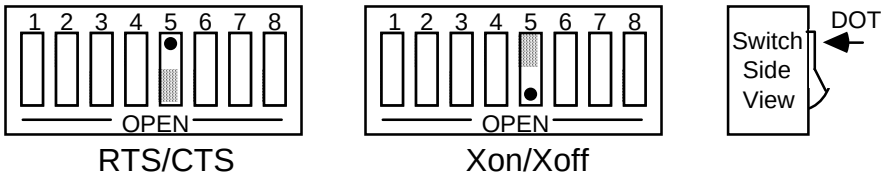
Switch SW1-5 is used to select between hardware [RTS/CTS] or software [XON/XOFF] serial handshake control.

With XON/XOFF, the Micro488/EX issues an XOFF character [ASCII value of &H13] when its buffer memory is near full. When issued, there are greater than 1000 character locations remaining to protect against buffer overrun. When it is able to accept more information it issues an XON character [ASCII value of &H11]. With this handshake, the Micro488/EX issues an XON character at reset or power-on. The Micro488/EX also accepts XON/XOFF on transmit from the serial host it is communicating with. RTS/CTS serial control becomes inactive when XON/XOFF is enabled. The RTS output is, however, set to an active high state. The CTS input is not used for this handshake and may be left floating (unconnected).

With RTS/CTS, the Micro488/EX un-asserts RTS (low) when its buffer memory is near full. When un-asserted, there are greater than 1000 character locations remaining to protect against buffer overrun. When it is able to accept more information it asserts (high) RTS. The Micro488/EX will not transmit data to the serial host if it detects the CTS input un-asserted (low) when configured for this hardware handshake.

The factory default serial control is hardware, RTS/CTS.

SW1 View for Serial Handshake Selection



2.4 Terminator Selection

In the Controller and Peripheral Modes, the Micro488/EX is not sensitive as to whether CR or LF is used as a serial input terminator to a command. In general, it requires only one of either to cause command execution. The IEEE input terminator is fixed to LF. The switches that allow terminator selection, shown in the following diagrams, set only the serial output and IEEE output terminators for these modes of operation.

In the transparent Pass-Thru modes, the Micro488/EX can be configured to provide RS-232 to IEEE 488 and IEEE 488 to RS-232 terminator substitution. This is useful when interfacing an RS-232 device which only issues carriage return [CR] as an output terminator to an IEEE controller which expects a carriage return followed by a line feed [CR-LF].

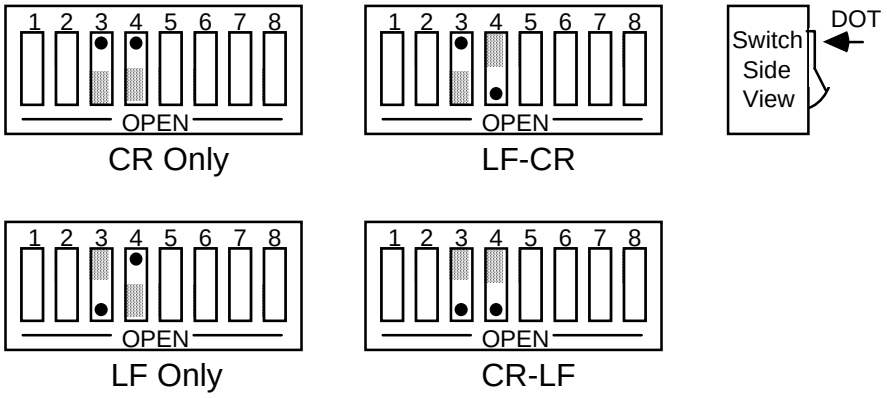
In the above case, the serial terminator should be selected for CR Only while the IEEE terminator is set to CR-LF. When a serial CR character is received, it is discarded and substituted with an IEEE CR-LF. In the IEEE to RS-232 direction, the IEEE CR is unconditionally discarded. Upon receipt of the IEEE LF, a serial CR is substituted.

The Micro488/EX can be made totally data transparent in the Pass-Thru modes by setting both the serial and IEEE terminators to be CR Only or LF Only.

2.4.1 Serial Terminator Selection

SW2-3 and SW2-4 select the serial terminators for the serial input (Pass-Thru Modes Only) and output. The factory default is CR-LF.

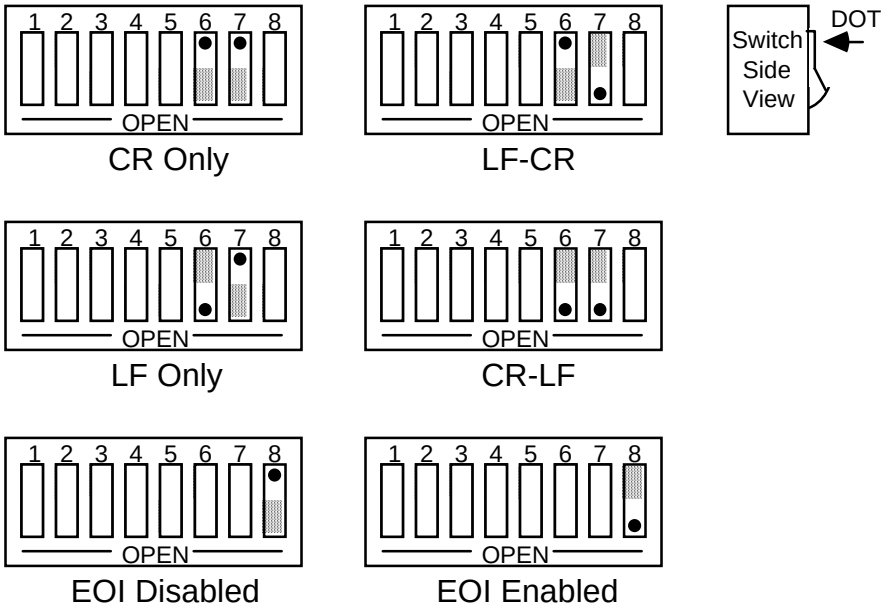
SW2 View for Serial Terminator Selection



2.4.2 IEEE Bus Terminator Selection

SW3-6 through SW3-8 set the IEEE bus terminators used for data sent or received (Pass-Thru modes only) by the Micro488/EX. EOI, a line used to signal the end of a multiple character bus transfer, may also be enabled. If enabled, EOI is asserted when the last selected bus terminator is sent. Factory default is CR-LF with EOI disabled.

SW3 View for IEEE Bus Terminator Selection



2.5 Mode Selection

SW2-1 and SW2-2 set the major operating mode of the Micro488/EX. There are four distinct modes of operation.

1. System Controller
2. Peripheral
3. Controller Pass-Thru
4. Peripheral Pass-Thru

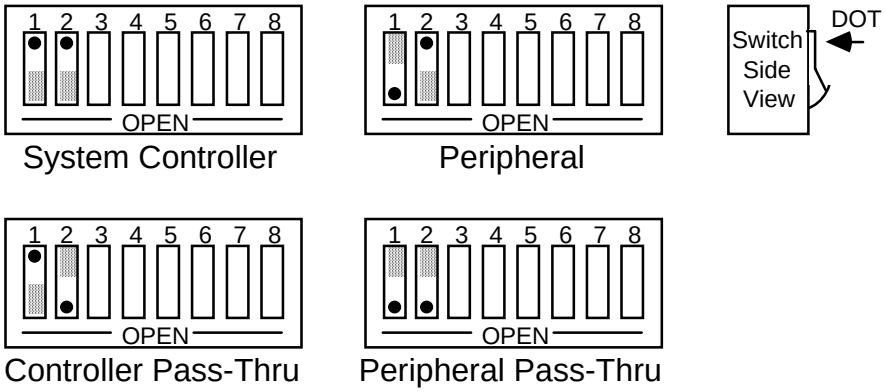
As a System Controller, the Micro488/EX accepts simple high-level ASCII commands from a serial host. It interprets these commands and performs the required bus action to bi-directionally communicate with up to 14 IEEE devices. As a Peripheral, the Micro488/EX becomes a bus device. It accepts simple high-level ASCII commands from a serial host and interprets these commands and status to communicate with another IEEE controller. Applications include computer controlled automatic test systems. These modes of operation are discussed in Sections 3, 4 and 5.

The IEEE Controller Pass-Thru (RS-232 to IEEE Converter) mode allows a serial host device to send data to a single IEEE bus peripheral. Applications include interfacing a listen-only or addressable IEEE printer/plotter to a serial printer port. Refer to Section 6 for more detailed information on the Controller Pass-Thru mode of operation.

The Peripheral Pass-Thru mode is used when interfacing a serial device to an IEEE controller. Data which is sent by the IEEE controller to the Micro488/EX is transmitted out its serial port. Data received from the serial device is buffered by the Micro488/EX until read by the IEEE controller. Refer to Section 7 for more detailed information on the Peripheral Pass-Thru mode of operation.

The factory default is the System Controller mode.

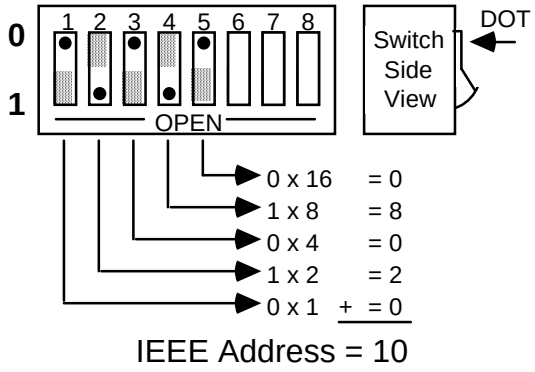
SW2 View for Mode Selection



2.6 IEEE Address Selection

SW3-1 through SW3-5 select the IEEE bus address of the Micro488/EX when in the System Controller, Peripheral and Peripheral Pass-Thru modes. These same switches are used in the Controller Pass-Thru mode to select the address of the device that will be controlled. [Refer to Section 6 for additional information]. The address is selected by simple binary weighting with SW3-1 being the least significant bit and SW3-5 the most significant. The factory default is address 10.

SW3 View for IEEE Address Selection



2.7 Feature Selections

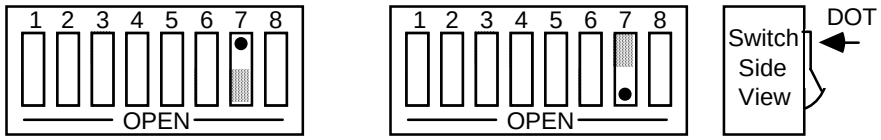
The functions of the remaining switches are dependent on the mode selected. A brief description of each of these features follows. You should refer to the listed sections for additional information.

2.7.1 Controller Pass-Thru Features

In the IEEE Controller (RS-232 to IEEE 488 Converter) mode, SW1-7 is used to determine whether the interface should, after sending the IEEE bus terminators, address the attached bus device to talk. The factory default is Talk-back On Terminator enabled.

Refer to Section 6 for complete details on these features.

SW1 View for Controller Talk-Back on Terminator Selection



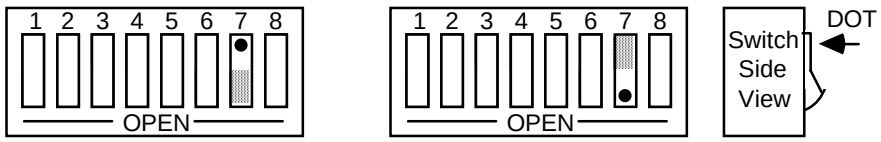
Talk Back on Terminator Disabled

Talk Back on Terminator Enabled

2.7.2 Peripheral Pass-Thru Features

In the Peripheral Pass-Thru (IEEE 488 to RS-232 converter) mode, SW1-7 enables the interface to assert the SRQ IEEE bus interface line to indicate that it has received the last switch selected serial terminator character from the serial device. Refer to Section 7 for more information.

SW1 View for Peripheral SRQ on Last Serial Terminator



SRQ on Last Terminator Disabled

SRQ on Last Terminator Enabled

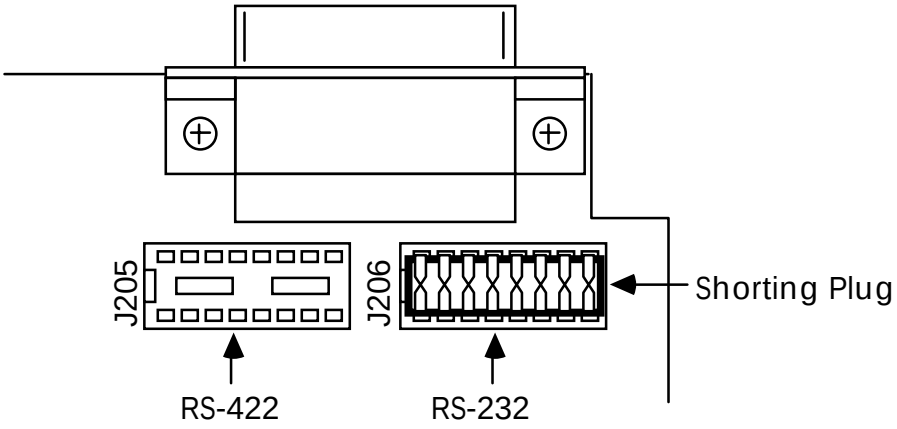
2.8 Serial Interface

The Micro488/EX has the ability to output signal levels that are compatible with either RS-232 or RS-422. An internal DIP shorting plug determines which electrical specification is chosen. If the interface is to be connected to an IBM PC/XT/AT/PS2 or compatible, the RS-232 level should be selected. If it will be connected to a Macintosh 512K/Plus/SE/II, the RS-422 level should be used. For connection to other computers, refer to the manufacturer's manual to determine which levels are supported.

2.8.1 RS-232/RS-422 Signal Level Selection

The Micro488/EX's factory default signal levels are compatible with RS-232. To select RS-422 levels, carefully remove the 8 position shorting plug with a small flat blade screwdriver from J106. Install the DIP jumper into J205 making certain that all of the pins on the shorting plug are inserted correctly.

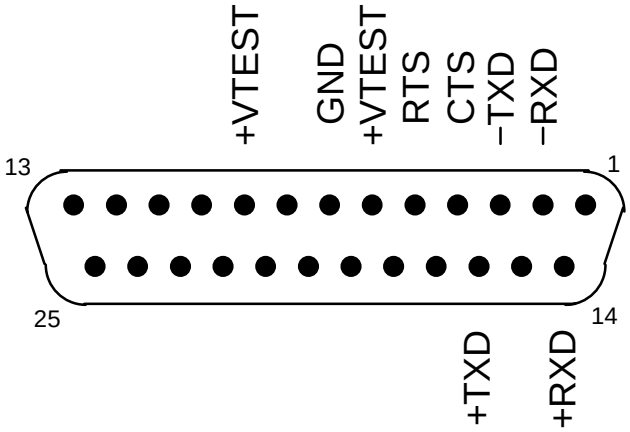
Selecting RS-232 or RS-422 Signal Levels



2.8.2 Serial Signal Descriptions

The Micro488/EX is equipped with a standard DB-25S connector on its rear panel and requires a standard DB-25P mating connector. The Micro488/EX's connector is configured as DCE type equipment for RS-232 communications, which means the Micro488/EX always transmits data on Pin 3 and receives data on Pin 2. The following lists and describes the RS-232 and RS-422 signals provided on the Micro488/EX.

Rear View of the Micro488/EX's Serial Connector



-RxD Receive Data - Input - Pin 2
This pin accepts serial data sent by the RS-232 or RS-422 host. The serial data is expected with the word length, baud rate, stop bits and parity selected by the internal switches. The signal level is low true.

-TxD Transmit Data - Output - Pin 3
This pin transmits serial data to the RS-232 or RS-422 host. The serial data is sent with the word length, baud rate, stop bits and parity selected by the internal switches. The signal level is low true.

CTS Clear To Send - Input - Pin 4
The CTS input is used as a hardware handshake line to prevent the Micro488/EX from transmitting serial data when the RS-232 host is not ready to accept it. When RTS/CTS handshake is selected on the internal switches, the Micro488/EX will not transmit data out -TxD while this line is un-asserted (low). If the RS-232 host is not capable of driving this line it can be connected to the Vtest output (Pin 6) of the Micro488/EX. If XON/XOFF handshake is selected, the CTS line is not tested to determine if it can transmit data.

RTS Request To Send - Output - Pin 5
The RTS output is used as a hardware handshake line to prevent the RS-232/RS-422 host from transmitting serial data if the Micro488/EX is not ready to accept it. When RTS/CTS handshake is selected on the internal switches, the Micro488/EX will drive the RTS output high when there are greater than 1000 character locations available in its internal buffer. If the number of available locations drops to less than 1000, the Micro488/EX will un-assert (low) this output. If XON/XOFF handshake is selected, the RTS line will be permanently driven active high.

Vtest Test Voltage - Output - Pin 6
This pin is connected to +5 volts through a 1KΩ resistor. It is also common to Vtest on pin 9.

Gnd Ground - Pin 7
This pin sets the ground reference point for the other RS-232 inputs and outputs.

`Vtest` Test Voltage - Output - Pin 9
This pin is connected to 5 volts through a 1KΩ resistor. It is also common to `Vtest` on pin 6.

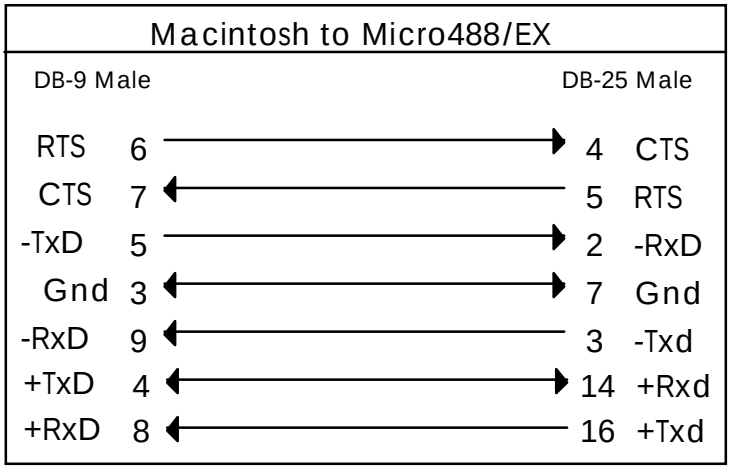
`+RxD` Receive Data Plus - Input - Pin 14
This pin accepts serial data sent by the RS-422 host. The serial data is expected with the word length, baud rate, stop bits and parity selected by the internal switches. The signal level is high true and only connected to this pin when RS-422 operation is selected. It is 180° out of phase with `-RxD`.

`+TxD` Transmit Data Plus - Output - Pin 16
This pin transmits serial data to the RS-422 host. The serial data is sent with the word length, baud rate, stop bits and parity selected by the internal switches. The signal level is high true and only connected to this pin when RS-422 operation is selected. It is 180° out of phase with `-TxD`.

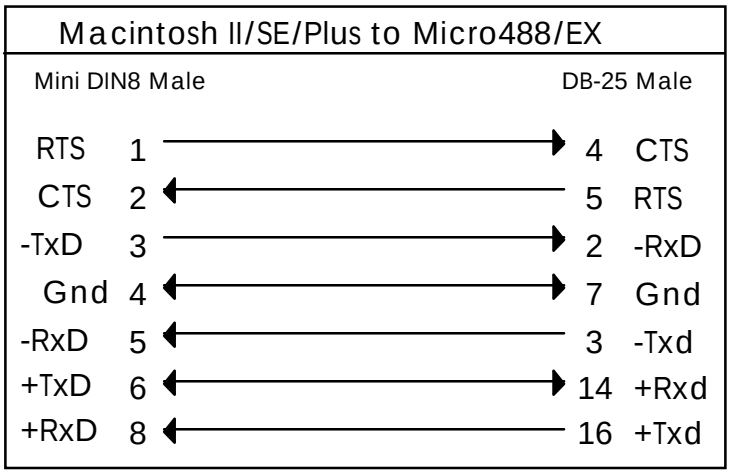
2.8.3 Serial Cable Wiring Diagrams

If a cable was not purchased with the interface, the following diagrams will be helpful in making your own cable. Simple soldering skills and an attention to detail will ensure successful construction.

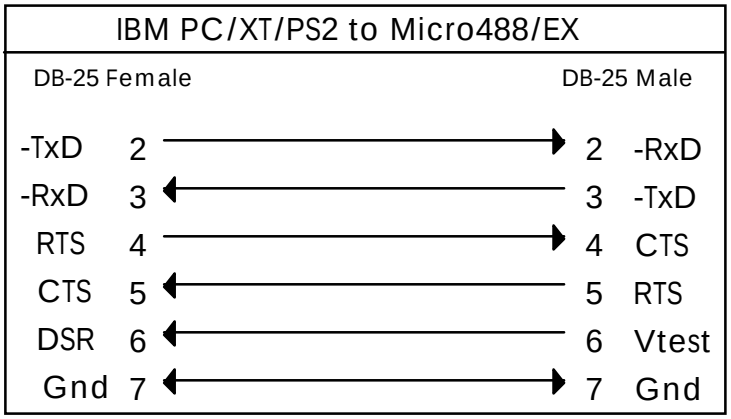
Macintosh to Micro488/EX Wiring Diagram (RS-422)



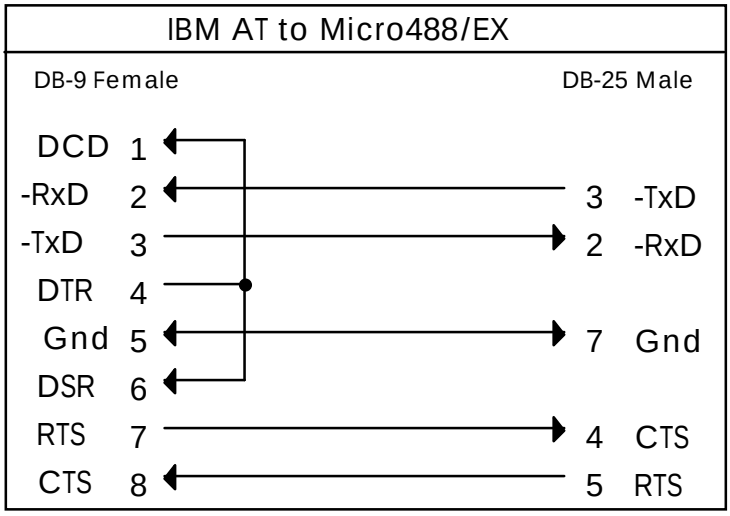
Macintosh II/SE/Plus to Micro488/EX Wiring Diagram (RS-422)



IBM PC/XT/PS2 to Micro488/EX Wiring Diagram (RS-232)



IBM AT to Micro488/EX Wiring Diagram (RS-232)



Note: Standard AT 9 Pin to 25 Pin adapter cables are not wired as shown above and will not work with the Micro488/EX. Order IOtech PN CA-23.

2.9 General Operation

Refer to the following sections for specific operational modes. This sub-section gives a general test of functionality. After setting the power on defaults and reassembling the Micro488/EX, plug the power supply connector 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 Micro488/EX.

WARNING

The power supply provided with the interface is intended for INDOOR USE ONLY. Failure to observe this warning could result in equipment failure, personal injury or death.

After installing the power supply connector into the interface, plug the power supply into AC line power. Place the rear panel power switch in the ON [1] position. All the front panel indicators should light momentarily while the Micro488/EX performs an internal ROM and RAM self check. At the end of this self check all indicators except POWER should turn off.

If there is an error in the ROM checksum, all of the LEDs will remain on. Flashing LEDs indicates a RAM failure. Should such an error occur, turn the rear panel switch to the OFF [0] position and retry the above procedure.

If the front panel indicators 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 the AC line and the rear panel connection of the power supply for proper installation. If the problem is unresolved, refer to the Service Information section of this manual.

If proper operation is obtained, connect an interface cable to the rear of the Micro488/EX [25-Pin Sub-D]. Connect the other end to the host's serial port. Except for connecting IEEE bus instruments, the Micro488/EX is installed and ready to use.

WARNING

The Micro488/EX makes its earth ground connection through the serial interface cable. It should only be connected to IEEE bus devices after being first connected to the host. Failure to do so may allow the Micro488/EX to float to a bus device test voltage. This could result in damage to the interface, personal injury or death.

2.10 Is Anyone Out There?

Before connecting any IEEE bus devices to the Micro488/EX, try this simple operational check. The Micro488/EX must be configured for either System Controller of Peripheral mode operation. This test will not work in either of the Pass-Thru modes.

Running BASIC on the host, or any programming language which supports the serial ports, type the following (or its equivalent).

```
OPEN "COM1:9600,N,8,2,cd,ds" AS 1   [Return]
PRINT #1,"HELLO"                    [Return]
LINE INPUT #1,A$                    [Return]
PRINT A$                            [Return]
```

The Micro488/EX will respond with (and the host will display):

```
Micro488/EX Revision N.N Copyright (C) 1988 IOtech Inc.
```

where `N.N` is the release and revision number of the firmware.

If you obtain the above response then your Micro488/EX is alive and well and ready to connect your host to the powerful IEEE-488 General Purpose Interface Bus. If you did not receive the above message, check for proper connection and fit of the interface cable. If, after reviewing the interface for proper installation, you do not obtain the desired results then refer to the Service Information section of this manual.

IEEE Operating Modes

3.1 Introduction

There are four types of IEEE bus devices: Active Controllers, Peripherals, Talk-only devices, and Listen-always devices. Talk-only and Listen-always devices are usually used together, in simple systems, such as a Talk-only digitizer sending results to a Listen-always plotter. In these simple systems no controller is needed because the talker assumes that it is the only talker on the bus, and the listener(s) assume that they are all supposed to receive all the data send over the bus. This is a simple and effective method of transferring data from one device and another, but is not adequate for more complex systems where, for example, one computer is controlling many different bus devices.

In more complex systems, the Active Controller sends commands to the various bus Peripherals telling them what to do. Commands such as Unlisten, Listen Address Group, Untalk, and Talk Address Group are sent by the controller to specify which device is to send data, and which are to receive it. For more details about the IEEE bus protocols see the section 'IEEE 488 Primer'.

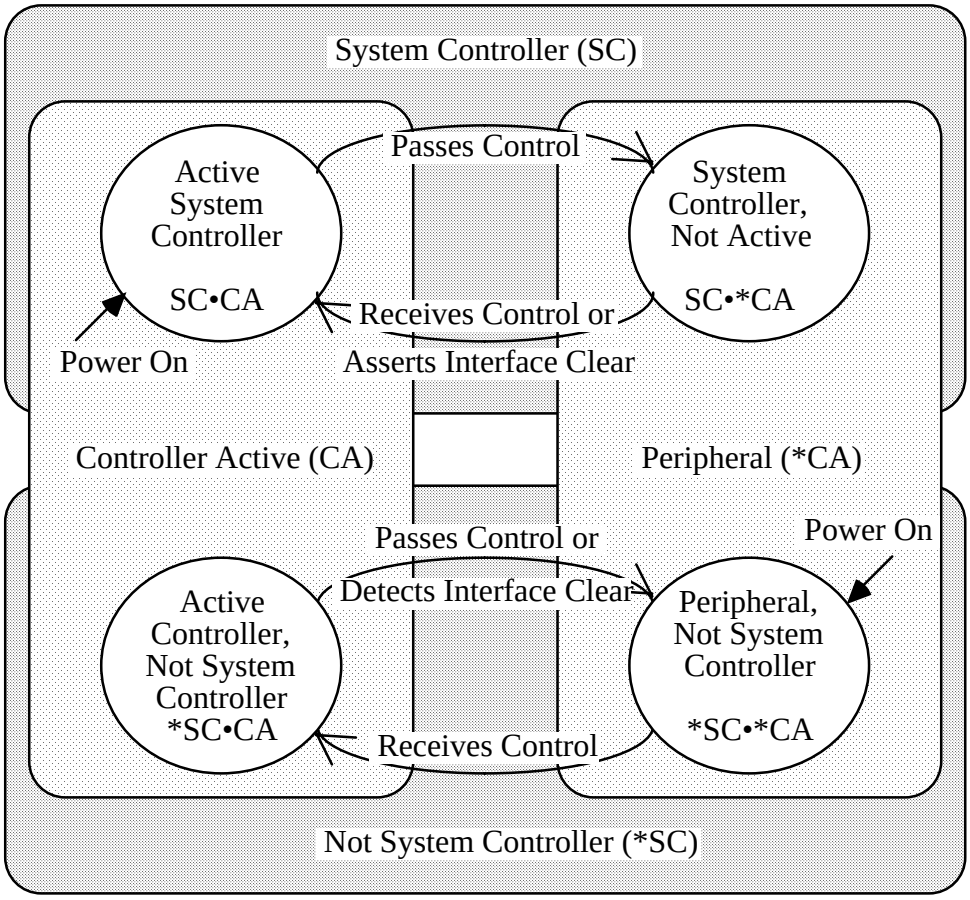
When an IEEE bus system is first turned on, some device must be the Active Controller. This device is the System Controller and always keeps some control of the bus. In particular, the System Controller controls the Interface Clear (IFC) and Remote Enable (REN) bus management lines. By asserting Interface Clear, the System Controller forces all the other bus devices to stop their bus operations, and regains control as the Active Controller.

3.2 Operating Mode Transitions

The System Controller is initially the Active Controller. It can, if desired, Pass Control to another device and thereby make that device the Active Controller. Note that the System Controller remains the System Controller, even when it is not the Active Controller. Of course, the device to which control is passed must be capable of taking on the role of Active Controller. It would make no sense to try to pass control to a printer. Control should only be passed to other computers that are capable, and ready, to become the Active Controller. Further, note that there must be exactly one System Controller on the IEEE bus. All other potential controllers must be configured as Peripherals when they power up.

The state diagram below shows the relationships between the various operating modes. The top half of the state diagram shows the two operating states of a System Controller. At power on, it is the active controller. It directs the bus transfers by sending the bus commands mentioned previously. It also has control of the Interface Clear and Remote Enable bus lines. The System Controller can pulse Interface Clear to reset all of the other bus devices.

As shown in the diagram, the System Controller can pass control to some other bus device and thereby become a Peripheral to the new Active Controller. If the System Controller receives control from the new Active Controller, then it will once again become the Active Controller. The System Controller can also force the Active Controller to relinquish control by asserting the Interface Clear signal.



IEEE Bus Operating Modes State Diagram

The bottom half of the state diagram shows the two operating states of a Not System Controller device. At power on, it is a Peripheral to the System Controller which is the Active Controller. If it receives control from the Active Controller, it becomes the new Active Controller. Even though it is the Active Controller, it is still not the System Controller. The System Controller can force the Active Controller to give up control by asserting Interface Clear. The Active Controller can also give up control by Passing Control to another device, which may or may not be the System Controller.

In summary, a bus device is set in hardware as either the sole System Controller in the system, or as a non-System Controller. At power on, the System Controller is the Active Controller, and the other devices are Peripherals. The System Controller can give up control by Passing Control, and can regain control by asserting Interface Clear, or by receiving control. A Peripheral can become the Active Controller by receiving control, and can give up control by Passing Control, or upon detecting Interface Clear.

3.3 System Controller Mode

The most common the Micro488/EX configuration is as the System Controller, controlling several IEEE bus instruments. In this mode, the Micro488/EX can perform all of the various IEEE bus protocols necessary control and communicate with any IEEE 488 bus devices. As the System Controller in the Active Controller mode, the Micro488/EX can use all of the commands available for the Active Controller state, plus control the Interface Clear and Remote Enable lines. The allowed bus commands and their actions are as follows:

`ABORT`	Pulse Interface Clear.
`LOCAL`	Unassert Remote Enable, or send Go To Local to selected devices.
`REMOTE`	Assert Remote Enable, optionally setting devices to Remote.
`LOCAL LOCKOUT`	Prevent local (front-panel) control of bus devices.
`CLEAR`	Clear all or selected devices.
`TRIGGER`	Trigger selected devices.

ENTER	Receive data from a bus device.
OUTPUT	Send data to bus devices.
PASS CONTROL	Give up control to another device which becomes the Active Controller.
SPOLL	Serial Poll a bus device, or check the Service Request state.
PPOLL	Parallel Poll the bus.
PPOLL CONFIG	Configure Parallel Poll responses.
PPOLL DISABLE	Disable the Parallel Poll response of selected bus devices.
PPOLL UNCONFIG	Disable the Parallel Poll response of all bus devices.
SEND	Send low-level bus sequences.
RESUME	Unassert Attention. Used to allow Peripheral-to- Peripheral transfers.

3.4 System Controller, Not Active Controller Mode

After Passing Control to another device, the System Controller is no longer the Active Controller. It acts as a Peripheral to the new Active Controller, and the allowed bus commands and their actions are modified accordingly. However, it still maintains control of the Interface Clear and Remote Enable lines. The available bus commands and their actions are:

ABORT	Pulse Interface Clear.
LOCAL	Unassert Remote Enable.
REMOTE	Assert Remote Enable.
ENTER	Receive data from a bus device as directed by the Active Controller.
OUTPUT	Send data to bus devices as directed by the Active Controller.
REQUEST	Set own Serial Poll request (including Service Request) status.
SPOLL	Get own Serial Poll request status.

As a bus Peripheral, the Micro488/EX must respond to the commands issued by the Active Controller. The controller can, for example, address the Micro488/EX to listen in preparation for sending data. There are two ways of detecting our being addressed to listen: through the STATUS command, or by detecting an event with the ARM or ON <event> DOMACRO commands.

The STATUS 1 command can be used to watch for commands from the Active Controller. The Operating Mode, which is a "P" while the Micro488/EX is a Peripheral, will change to a "C" if the Active Controller Passes Control to the Micro488/EX. The Addressed State will go from Idle ("I") to Listener ("L") or Talker ("T") if the Micro488/EX is addressed to listen or to talk, and will go back to Idle ("I") when the Active Controller issues Unlisten (UNL), Untalk (UNT), or specifies another talker (TAG). The Triggered ("T1") and Cleared ("C1") indicators will be set when the Micro488/EX is triggered or cleared, and reset when STATUS 1 is read. The Address Change indicator will be set ("G1") when the address state changes. These indicators allow the program to sense the commands issued to the Micro488/EX by the Active Controller. The following BASIC program fragment illustrates the use of the Address Change and Addressed State indicators to communicate with the Active Controller:

First we check STATUS until it indicates that there has been an address change:

```
200 PRINT#1,"STATUS1"
210 INPUT#2 ST$
220 'Has there been no Address Change?
230 IF MID$(ST$,7,1)="0" THEN 200
240 'Are will still in the idle state?
250 STATE$=MID$(ST$,9,1)
260 IF STATE$="I" THEN 200
270 'Are we addressed to listen?
280 IF STATE$="L" THEN 400
290 'Are we addressed to talk?
300 IF STATE$="T" THEN 500
310 PRINT "BAD ADDRESSED STATE VALUE: ";ST$: STOP
```

If we are addressed to listen then we ENTER a line from the controller and print it out.

```
400 'Listen state
410 PRINT#1,"ENTER"
420 LINE INPUT#1,A$
430 PRINT A$
440 GOTO 200
```

If we are addressed to talk then we INPUT a line from the keyboard and OUTPUT it to the controller.

```
500 'Talk state
510 LINE INPUT A$
520 PRINT#1,"OUTPUT;";A$
530 GOTO 200
```

It is also possible to detect these conditions with the ARM or ON <event> DOMACRO commands and handle them in an exception as described in Section 5. The various arm conditions and their meanings are as follows:

SRQ	The internal Service Request state is set. See the SPOLL command in Section 5.
PERIPHERAL	the Micro488/EX is in the Peripheral (*CA) operating mode.
CONTROLLER	the Micro488/EX is the Active Controller (CA).
TRIGGER	the Micro488/EX, as a Peripheral, has received a Trigger bus command.
CLEAR	the Micro488/EX, as a Peripheral, has received a Clear bus command.
TALK	the Micro488/EX is in the Talk state and can OUTPUT to the bus.
LISTEN	the Micro488/EX is in the Listen state and can ENTER from the bus.
IDLE	the Micro488/EX is in neither the Talk nor Listen state.

CHANGE	An Address Change has occurred, i.e. a change between Peripheral and Controller, or among Talk, Listen, and Idle has occurred.
ERROR	An error, either command or bus, has been detected by the Micro488/EX.
STARTUP	The non-volatile Macro capability provides the stand-alone controller feature by allowing the Micro488/EX to execute a Macro at STARTUP.

3.5 Not System Controller Mode

If the Micro488/EX is configured as not the System Controller then, at power on, it will be a bus Peripheral. It might use a program like the one described previously to communicate with the Active Controller. The bus commands available to the Micro488/EX when it is not the System Controller and not the Active Controller (*SC•*CA) are:

ENTER	Receive data from a bus device as directed by the Active Controller.
OUTPUT	Send data to bus devices as directed by the Active Controller.
REQUEST	Set own Serial Poll request (including Service Request) status.
SPOLL	Get own Serial Poll request status.

3.6 Active Controller, Not System Controller Mode

If the Active Controller Passes Control to the Micro488/EX then it will become the new Active Controller. This can be detected by the STATUS command or as an ARMed event. As an Active Controller, but not the System Controller, the following bus commands are available:

*ABORT	Assert Attention and send My Talk Address to stop any bus transfers.

LOCAL	Send Go To Local to selected devices.
LOCAL LOCKOUT	Prevent local (front-panel) control of bus devices.
CLEAR	Clear all or selected devices.
TRIGGER	Trigger selected devices.
ENTER	Receive data from a bus device.
OUTPUT	Send data to bus devices.
PASS CONTROL	Give up control to another device which becomes the Active Controller.
SPOLL	Serial Poll a bus device, or check the Service Request state.
PPOLL	Parallel Poll the bus.
PPOLL CONFIG	Configure Parallel Poll responses.
PPOLL DISABLE	Disable the Parallel Poll response of selected bus devices.
PPOLL UNCONFIG	Disable the Parallel Poll response of all bus devices.
SEND	Send low-level bus sequences.
RESUME	Unassert Attention. Used to allow Peripheral-to- Peripheral transfers.

3.7 Controller Pass-Thru Mode

This mode is intended to provide bi-directional data transparent conversion between an RS-232/RS-422 host computer and an IEEE 488 peripheral, such as a printer or a HPIB™ plotter. The operation of this mode is covered in Section 6 of this manual.

3.8 Peripheral Pass-Thru Mode

This mode is intended to provide bi-directional data transparent conversion between an IEEE 488 controller and a serial device. This Peripheral Pass-Thru mode does not require the serial device to control data to it. The is no command line parser and, therefore, requires no serial commands. This mode of operation is described in Section 7 of this manual.

General Programming

4.1 Introduction

This section provides example information for programming the Micro488/EX in the Controller and Peripheral modes using the MACRO and clock features. The concentration is placed on Micro488/EX commands and not specifically on the programming language. All the examples shown use one of the TERMINAL programs listed in Appendix D. Other languages may be used as long as they provide similar functionality. Using this type of terminal program is recommended to become familiar with the Micro488/EX and any new bus device as it allows direct human-device interaction.

4.2 Memory Usage

To better understand MACRO programming on the Micro488/EX, a description of how the system allocates memory for different functions will allow the user to tailor programming for the most effective use.

Memory in the Micro488/EX is dynamically allocated for the serial input, serial output, MACRO and LOG buffers. This allows for the most efficient partitioning of memory for any given application. This memory is kept in the USER 'heap' (a vernacular for 'heap of memory') until required by the system. A MEMORY command has been included in the Micro488/EX to report the available memory in the USER heap.

4.2.1 Serial I/O Buffers

At power on, each serial buffer is allocated an empty 127 byte mini-buffer or queue. When the serial input [or output] requires more buffer space, additional queues are allocated. When a queue is empty, it is released from the buffer so that it may be re-allocated when, and where, required.

There are approximately 240 available queues for a total of 29,000 bytes of buffer (character) space. Queues are continually allocated and released as required. Of the 240 available queues, 230 are assigned to the

USER heap and issued without regard to controlling the receipt and buffering of additional data.

When the serial input buffer requests one of the last 10 queues (1270 character locations left), it signals the serial host that it should stop sending data. This is accomplished by either un-asserting the serial hardware handshake RTS line or issuing an XOFF character (&h13), depending on which serial handshake control has been switch selected. When more than 10 queues become available, it asserts RTS or issues an XON character (&h11).

4.2.2 Log Buffer

The LOG buffer is similar to the serial output buffer. It can, under program control, accept the data which would normally go to the serial output buffer. It differs from the serial output buffer in that the data in the LOG buffer is maintained when the power is off. The LOG buffer is initially allocated a 127 byte queue and, if required, acquires additional memory from the USER heap. It retains these queues until ERASEd by the user under program control.

4.2.3 Macro Buffers

MACRO queues are not allocated until a macro is defined. If a MACRO is ERASEd, all the memory allocated to it is returned to the USER heap. The MACRO text is stored into this queue exactly the way it was received from the serial input. Syntax error checking is not performed when the MACRO is created. If the length of the MACRO is longer than the 127 bytes allocated, additional memory is allocated as required in 127 byte lengths. A single MACRO could be defined to consume the entire available USER heap.

When a MACRO is executed, a copy of the MACRO buffer is made and placed at the start of the serial input buffer. If there is no available memory in the USER heap, an OUT OF MEMORY error is generated which will cause the front panel error LED to flash.

4.2.4 Changing Operational Modes

If memory is allocated from the USER heap for LOG and MACRO buffers in the Controller or Peripheral modes, and the operational mode is changed to one of the Pass-Thru modes, this memory remains allocated and is not available to the Pass-Thru mode. Pass-Thru modes will function as described but will not have the full memory available to them. Upon returning to the Controller or Peripheral mode, the LOG and MACRO buffers will be un-modified.

4.3 Clock and Timer Functions

The Micro488/EX has a built in time of day, day of week and date clock. Most timer features, with the exception of TIME OUT and DELAY, are directly a function of this hardware. TIME OUT and DELAY are functions of the Micro488/EX's firmware operating system. As a result, some time skew may be observed between these commands with respect to the clock.

Both 12 and 24 hour time output formats are available from the clock. The relationship between the 12 and 24 hour operations are…

	24 Hour	12 Hour
Midnight	00:00:00	12:00:00 AM
	01:00:00	01:00:00 AM
	…	…
	11:59:59	11:59:59 AM
Noon (Midday)	12:00:00	12:00:00 PM
	13:00:00	01:00:00 PM
	…	…
	23:59:59	11:59:59 PM

4.3.1 Time

While running one of the TERMINAL programs in Appendix D, type the following on the PC's keyboard…

HELLO <return>

The Micro488/EX will respond with…

```
Micro488/EX Revision 1.0 Copyright (C) 1988 IOtech Inc.
```

Now type…

```
SET TIME 1:00 PM                <return>
TIME                            <return>
```

The Micro488/EX will respond with…

```
01:00:02 PM
```

The actual time returned will depend, of course, on how long an interval passes between typing the `SET TIME` and `TIME` commands.

Now type the following to change the `TIME` output to a 24 hour format.

```
TIME FORMAT HH:MM:SS            <return>
TIME                            <return>
```

The Micro488/EX will respond with…

```
13:00:15
```

To test the functionality of the WAIT command, type…

```
SET TIME 1:00 PM                <return>
WAIT TIME 1:01 PM               <return>
TIME                            <return>
```

There will be a one minute delay at which time the Micro488/EX will respond with…

```
13:01:00
```

For more information, refer to the following TIME related commands.

```
SET TIME
TIME
TIME FORMAT
WAIT
```

4.3.2 Date

While running one of the TERMINAL programs in Appendix D, type the following on the PC's keyboard…

```
SET DATE JULY 4, 1988          <return>
DATE                           <return>
```

The Micro488/EX will respond with…

```
July 4, 1988
```

To test the functionality of the CASE command, type…

```
CASE UPPER                     <return>
DATE                           <return>
```

The Micro488/EX will respond with…

```
JULY 4, 1988
```

Now type the following to change the `DATE` output to a numeric format.

```
DATE FORMAT MM-DD-YY           <return>
DATE                           <return>
```

The Micro488/EX will respond with…

```
07-04-88
```

For more information, refer to the following DATE related commands.

```
SET DATE
DATE
DATE FORMAT
WAIT
```

4.3.3 Day of Week

While running one of the TERMINAL programs in Appendix D, type the following on the PC's keyboard…

```
SET DAY MONDAY                    <return>
DAY                               <return>
```

The Micro488/EX will respond with…

```
MONDAY
```

Now type the following to change the `DAY` output to a numeric format.

```
DAY FORMAT D                      <return>
DAY                               <return>
```

The Micro488/EX will respond with…

```
2
```

For more information, refer to the following DAY related commands.

```
SET DAY
DAY
DAY FORMAT
WAIT
```

4.3.4 Combinations

The DATE, DAY and TIME commands may be sent within a single command line in any order, as long as each command appears only once. Type the following on the PC's keyboard…

```
DAY FORMAT DAY                    <return>
DATE FORMAT MONTH DD, YYYY        <return>
TIME FORMAT HH:MM:SS  AM          <return>

DAY DATE TIME                     <return>
```

The Micro488/EX will respond with…

```
MONDAY JULY 4, 1988 1:05:00 PM
```

4.4 MACRO Programming

The MACRO command allows the user to build a file of sequential Micro488/EX commands and execute them with a single DOMACRO command. The Micro488/EX is capable of storing up to 100 different MACROs.

Defining a MACRO is initiated by issuing the MACRO command. Each subsequent character following this command, including terminators and intervening spaces, is saved in a buffer up to, and including, the ENDM. After the ENDM, the Micro488/EX appends the MACRO number to the MACRO buffer as a two digit decimal number. The macro can then be executed by issuing a DOMACRO command. Any syntax errors that are included within the MACRO are not checked until the MACRO is executed.

If a MACRO has not been defined, it does not consume any memory from the USER heap. If a MACRO has been defined, any MACRO buffer previously allocated is returned to the USER heap prior to requesting memory to re-define the MACRO. The initial size of an allocated MACRO buffer is 127 character locations (bytes). If more than 127 bytes are required to store the MACRO, additional memory is allocated in 127 byte increments. If there is no available memory in the USER heap, an 'OUT OF MEMORY' error occurs and any memory allocated to that MACRO is returned to the heap.

4.4.1 Creating a MACRO

The user defines a macro by the following command sequence:

```
MACRO 1
<list of valid Micro488/EX commands>
ENDM
```

Type the following on the PC's keyboard to create a simple macro…

MACRO 1	<return>
DAY	<return>
DATE	<return>
ENDM	<return>

Any of the MACRO command buffers can be reported, or sent, back to the host with the READ command. This command will output the data in the respective buffer but will not delete the information contained in the buffer. To have the Micro488/EX report the contents of this macro buffer type…

```
READ 1                                  <return>
```

The Micro488/EX will respond with…

```
DAY
DATE
ENDM01
```

4.4.2 Executing a MACRO

Macros are executed by the DOMACRO command. This command specifies the macro number to execute. Type the following…

```
DOMACRO 1                               <return>
```

The Micro488/EX will respond with…

```
MONDAY
JULY 4, 1988
```

Optionally, the DOMACRO command can specify the number of times to execute the MACRO (loop). The number of times a single MACRO can execute is 255. Type the following…

```
DOMACRO 1,3                             <return>
```

The Micro488/EX will respond with…

```
MONDAY
JULY 4, 1988
MONDAY
JULY 4, 1988
MONDAY
JULY 4, 1988
```

If it is desired to execute a MACRO more than the allowed 255 times, another MACRO can be created to invoke the first. For example…

```
MACRO 2                          <return>
DOMACRO 1,200                    <return>
DOMACRO 1,200                    <return>
DOMACRO 1,200                    <return>
ENDM                             <return>

DOMACRO 2                        <return>
```

will execute MACRO 1 a total of 600 (200+200+200) times while…

```
DOMACRO 2,100                    <return>
```

will execute MACRO 1 a total of 60000 (100*[200+200+200]) times.

An additional time interval specifier, in seconds, can be included with the number of times specifier, to set a precise delay from the start of one execution to the start of the next. Only one interval timer is provided. Having two MACROs trying to use it at the same time will cause a TIMER IN USE error. Type the following…

```
MACRO 1                          <return>
TIME                             <return>
ENDM                             <return>

MACRO 2                          <return>
SET TIME 12:00 PM                <return>
DOMACRO 1,3,5                    <return>
TIME                             <return>
ENDM                             <return>

DOMACRO 2                        <return>
```

The Micro488/EX will respond with…

```
12:00:00
                        < 5 Second Delay>
12:00:05
                        < 5 Second Delay>
12:00:10
12:00:10
```

MACROs can execute other MACROs but can not execute already executing MACRO. If a MACRO tries to invoke itself or an already executing MACRO, a 'MACRO RECURSION' error will be generated.

A method has been provide to determine the loop number of the MACRO being executed. Type…

```
MACRO 1                         <return>
COUNT                           <return>
TIME                            <return>
ENDM                            <return>

DOMACRO 2                       <return>
```

The Micro488/EX will respond with…

```
3
12:00:00
                        < 5 Second Delay>
2
12:00:05
                        < 5 Second Delay>
1
12:00:10
12:00:10
```

4.4.3 Debugging a MACRO

The TRACE ON command allows the embedded macro commands within the macro buffer to be echoed out the serial port to the host computer as the Macro is executed. This allows trace debugging during Macro execution. This feature is disabled with the TRACE OFF command.

Type…

```
TRACE ON                                <return>
DOMACRO 2                               <return>
```

The Micro488/EX will respond with…

```
SET TIME 12:00 PM
DOMACRO 1,3,5
COUNT
3
TIME
12:00:00
ENDM01
                        < 5 Second Delay>
COUNT
2
TIME
12:00:05
ENDM01
                        < 5 Second Delay>
COUNT
1
TIME
12:00:10
ENDM01
TIME
12:00:10
ENDM02
```

Type the following to disable the TRACE feature…

```
TRACE OFF
```

4.4.4 Logging MACRO Data

The Micro488/EX includes a non-volatile LOG buffer. Data generated during MACRO execution can be forced to this buffer rather than the serial output buffer. Type the following…

```
LOG ON                                  <return>
DOMACRO 2                               <return>
HELLO                                   <return>
```

After a period of approximately 10 seconds the Micro488/EX will respond with…

```
Micro488/EX Revision 1.0 Copyright (C) 1988 IOtech Inc.
```

Notice that the output from MACRO 2 which was previously displayed did not appear. This data went into the LOG buffer. Type the following…

```
LOG MEMORY                              <return>
```

The Micro488/EX will respond with the number of bytes in the LOG buffer of…

```
54
```

```
Now type…
```

```
READ LOG                                <return>
```

The Micro488/EX will respond data in the LOG buffer of…

```
3
12:00:00 PM
2
12:00:05 PM
1
12:00:10 PM
12:00:10 PM
```

Even though the data in the LOG buffer has been read, it has not been deleted. If logging continues, the new data will be appended to the old. You can read this data as many times as you want.

To delete the data in the LOG buffer type the following…

```
ERASE LOG                    <return>
LOG MEMORY                   <return>
```

The Micro488/EX will respond with …

```
0
```

4.4.5 Event Driven MACRO Execution

The ON <event> DOMACRO command allows the Micro488/EX to automatically execute a MACRO when one or more of the specified events occur. The events are polled between commands and when one of the events is detected as true, its assigned MACRO is executed. Once executed, the event is disabled from further execution and must be re-enabled with another ON <event> DOMACRO command.

There are two types of events, level sensitive and edge sensitive. Level sensitive events, such as SRQ, will cause MACRO execution every time they are enabled while the event condition persists. Usually, some action must be taken (eg SPOLL) to clear the condition prior to re-issuing the ON <event> DOMACRO command. Edge sensitive events, such as TRIGGER, are cleared when the MACRO executes.

Regardless of the event sensitivity, the ON <event> DOMACRO command must be re-sent after the MACRO executes to re-activate the event condition. The optional events include…

SRQ — This event is level sensitive. If the condition exists at the time the ON SRQ DOMACRO command is issued, the Micro488/EX will execute the assigned MACRO immediately.

`PERIPHERAL` This event occurs when the Micro488/EX is force from the Controller Active State (*SC•CA) to the Peripheral State (*SC•*CA) by receipt of IFC from the System Controller. This can be useful in detecting receipt of IFC when in the *SC•CA state. This event is edge sensitive.

`CONTROLLER` This event occurs when the Micro488/EX receives control of the bus and transitions from the Peripheral State (*SC•*CA) to the Controller Active State (*SC•CA). This occurs when the Take Control interface message is received by the Micro488/EX. This event is edge sensitive.

`TRIGGER` This event occurs when the Micro488/EX, as a Peripheral (*CA), receives a Group Execute Trigger (GET) command from the Active Controller. This event is edge sensitive. When the `MACRO` is executed, the internal status, as read by the `STATUS 1` command, is cleared.

`CLEAR` This event occurs when the Micro488/EX, as a Peripheral (*CA), receives a Device Clear (DCL) or a Selected Device Clear (SDC) command from the Active Controller. This event is edge sensitive. When the `MACRO` is executed, the internal status, as read by the `STATUS 1` command, is cleared.

`TALK` This event occurs when the Micro488/EX, as a Peripheral (*CA), detects its My Talk Address (MTA) command from the Active Controller. It indicates that the controller has requested information from the Micro488/EX. This event is edge sensitive.

`LISTEN` This event occurs when the Micro488/EX, as a Peripheral (*CA), detects its My Listen Address (MLA) command from the Active Controller. It indicates that the controller has information it wants to send to the Micro488/EX. This event is edge sensitive.

IDLE — This event occurs when the Micro488/EX, as a Peripheral (*CA), transitions from a Talker or Listener state to an idle state (neither talker or listener). It indicates that the controller has unaddressed the Micro488/EX with either an UNT or UNL command. This event is edge sensitive. When the MACRO is executed, the internal status, as read by the STATUS 1 command, is cleared. This event is tested prior to the CHANGE event and will clear the internal CHANGE status only when the addressed to un-addressed transition occurs.

CHANGE — This event occurs when the Micro488/EX, as a Peripheral (*CA), detects an addressed state change. This occurs on transitions from a Talker or Listener state to an idle state (neither talker or listener), or from an idle state to a talker or listener state. This event is edge sensitive. When the MACRO is executed, the internal status, as read by the STATUS 1 command, is cleared. This event is tested after the IDLE event. If IDLE is enabled, the CHANGE event's Macro will only execute when an un-addressed to an addressed transition occurs.

ERROR — This event occurs when the Micro488/EX detects an error condition. The error condition may be an un-recognized command from the serial host, an invalid parameter or a bus error. Refer to Appendix B for a listing of the error conditions which can be detected by the Micro488/EX. This event is level sensitive. When the MACRO executes, the error status, as read by any of the STATUS commands, must be read prior to enabling this event again.

STARTUP — This edge sensitive event occurs when the Micro488/EX is powered on, upon the receipt of an ID character, double ID characters or RESET command. If a MACRO has been assigned as a STARTUP macro, it sets LOG ON and executes. This is the only event which cannot be re-enabled within a MACRO. Although STARTUP is a valid condition within the ARM command, it does not ARM.

Once a condition is enabled it remains enabled until it is DISARMed, the event specified has occurred or until the Micro488/EX is reset. The ON <event> DOMACRO and ARM commands are mutually exclusive. The last command issued takes precedence.

The program which we have been using for the preceding examples, refer to Appendix D, is a good example of event driven MACRO execution. It uses the ERROR event to execute MACRO 4. It also uses the COMMENT command to print a BELL character (&H07) to the screen.

4.4.6 Defining a STARTUP MACRO

One of the major features of the Micro488/EX is the ability to define a MACRO which will execute at STARTUP (power-on). This allows the Micro488/EX to be locally programmed and moved to a remote location for stand-alone data collection from IEEE bus instruments.

A simple example of this STARTUP feature would be a test system used to monitor the power that a device consumes after Sunday at 12:30 AM over-night. Two instruments are used, one to measure the voltage and the other to measure the current.

The desired output format is one in which the data could be placed into a spreadsheet for plotting or analysis. As such, the serial terminator is disabled and an embedded TAB character is placed between data fields using COMMENT commands. This will cause the spreadsheet to place the data in sequential columns. The serial output terminator is set to CR to force the next execution of MACRO 1 to place data on a new row.

The defined MACROs would appear as…

```
MACRO 1
STERM NONE          Serial Output Terminator set to none
TIME                Time Stamp
COMMENT ''          Embedded TAB character to next column
ENTER 16            Get voltage reading
COMMENT ''          Embedded TAB character to next column
STERM CR            Serial Output Terminator set to CR to next row
ENTER 12            Get current reading
ENDM
```

```
MACRO 2
REMOTE 12/16                    Visual indication that all is working
OUTPUT 12;F2R5X                 Device Dependent Command for Device 12
OUTPUT 12;F0R4X                 Device Dependent Command for Device 16
COMMENT 'Power Test'            Title
DATE FORMAT MONTH DD,YYYY
TIME FORMAT HH:MM
DATE                            Date Stamp
COMMENT ' '                     Blank Line
WAIT DAY SUNDAY                 Wait until Sunday to begin test
WAIT TIME 12:30 AM              Wait until 12:30 AM
DOMACRO 1,100,600               Do #1, 100 times at 10 minute intervals
ENDM
```

Now, define the STARTUP MACRO as 2…

```
ON STARTUP DOMACRO 2
```

After the MACROs have been defined, power is removed from the Micro488/EX and it is located at the site of the test equipment. Apply power to the instruments first. After connecting the IEEE cables, apply power to the Micro488/EX. It is always a good practice to provide some visual indication that the STARTUP MACRO has executed. This is the reason for the REMOTE command in MACRO 2.

It was not necessary to issue the LOG ON command within the STARTUP MACRO. STARTUP executes LOG ON prior to invoking the STARTUP MACRO.

After the data has been collected, the power is removed from the Micro488/EX and returned to the local site. The data can now be READ from the LOG buffer.

4.4.7 Deleting a MACRO

The ERASE command is included to delete a MACRO. It can simply be re-defined as a new MACRO with null commands. This method, however, will still require the 127 byte memory allocation. A better way is to ERASE it from memory. This un-allocates memory assigned to the MACRO. An ERASE command without parameters will clear all MACRO buffers. Type the following…

```
READ 1      <return>
```

The Micro488/EX will respond with…

```
TIME
ENDM01
```

Type…

```
ERASE 1     <return>
READ 1      <return>
```

The Micro488/EX will respond with, as a result of the programs ON ERROR DOMACRO command, the following error status…

```
C 10 G0 I S0 E06 T0 C0 NO MACRO
```

4.4.8 Saving the LOG Buffer to Disk

The following program example reads the Micro488/EX's log buffer and sends the information it contains to a disk file. The example is written in PC basic and utilizes software, Xon/Xoff, serial handshaking.

Read Log Buffer Program

```
100 '
110 ' Program to read the Micro488/EX's log buffer
120 '      and save it to a disk file.
130 '    Serial control is set to Xon/Xoff
140 '             IOtech, Inc.
150 '
160 ' Open the serial port for communications
170 OPEN "com1:9600,n,8,2" AS 1
180 PRINT#1,"@@";  ' reset the Micro488/EX
```

```
190 GOSUB 570 ' get the output file name
200 PRINT:PRINT "Enter File Comment => ";: LINE INPUT C$
210 ' loop until serial input buffer is empty
220 WHILE NOT EOF(1) : A$ = INPUT$(1,1) : WEND
230 '
240 ' set up file header
250 '
260 PRINT: PRINT : PRINT "Reading Log Buffer.";
270 PRINT#2,"*"
280 PRINT#1,"Hello"
290 LINE INPUT #1,H$
300 PRINT#2,"*  ";H$ ' send the Macro488's Hello Message to
the file
310 PRINT#1,"Day Date Time"
320 LINE INPUT#1, D$ ' send Day, Date and Time to the file
330 PRINT#2,"*  ";D$
340 PRINT#2,"*  ";C$ ' send the User's comment to the file
350 PRINT#2,"*" : PRINT#2,""
360 '
370 ' get the amount of data in the log buffer
380 '
390 PRINT#1,"Log Memory"
400 INPUT #1,COUNT
410 IF COUNT = 0 THEN PRINT :PRINT:PRINT "Log Buffer Is
Empty" : END
420 PRINT#1,"Read Log"
430 IF LOC(1) > 50 THEN GOSUB 480 ' get data
440 IF LOC(1) <> COUNT THEN GOTO 430
450 IF COUNT > 0 THEN GOSUB 480
460 PRINT:PRINT : PRINT "Operation Complete"
470 END
480 '
490 ' read data from Micro488/EX to disk file
500 '
510 PRINT "."; ' for visual indication
520 PRINT#1,CHR$(19); ' Send Xoff to avoid PC buffer overrun
530 A = LOC(1) : COUNT = COUNT - A
540 PRINT#2,INPUT$(A,1);
550 PRINT#1,CHR$(17); ' Send Xon
560 RETURN
570 '
580 ' get output file name
590 '
600 CLS
610 PRINT : PRINT"Send Log data to what file? => ";
620 LINE INPUT F$
630 IF F$ = "" THEN PRINT CHR$(7) : GOTO 600 ' try again
640 ' add extension if not included
650 IF INSTR(F$,".") THEN GOTO 660 ELSE F$ = F$ + ".log"
660 ' check if file already exists
670 ON ERROR GOTO 790
```

```
680 OPEN F$ FOR INPUT AS #2
690 CLOSE #2
700  PRINT:  PRINT  CHR$(7);"Replace  existing  ";F$;"  File?
(Y/N) ";
710 K$ = INKEY$ : PRINT K$;
720 IF K$ = "y" OR K$ = "Y" THEN GOTO 750
730 IF K$ = "n" OR K$ = "N" THEN GOTO 610
740 GOTO 710 ' try agin for valid key press
750 ' open the disk file
760 ON ERROR GOTO 0 ' turn error off
770 OPEN F$ FOR OUTPUT AS #2
780 RETURN
790 ' file does not exist error
800 RESUME 750
```

The program can be modified to utilize CTS/RTS hardware serial handshaking by including and substituting the follow program lines. Hardware handshake is implemented by writing directly to the COM port hardware. The hardware address shown below is for the Modem Control Register of the 8250 UART in COM1. For COM2, refer to your hardware manual.

Add the following program lines…

```
175 MCR = &H03FE ' Modem control register of 8250 UART
176 ON = INP(MCR) OR  &H02
177 OFF = IMP(MCR) AND &HFD
```

and substitute the following program lines…

```
520 OUT MCR,OFF ' Un-Assert RTS
550 OUT MCR,ON ' Assert RTS
```

The following describes the functionality of the Read Log program on a line by line basis.

Line 170	Opens the COM1 port on the PC for serial communications with the Micro488/EX.
Line 180	Sends the ID command to the Micro488/EX to reset it to known power-on conditions. It utilizes the time delay provided by line 190; get the output file name.
Line 190	GOSUBs to a subroutine to get the output file name and open it.
Line 200	Provides a method to include a user comment in the log

file.

Lines 230-350 Sets up the file header with comments. These comments include the HELLO response, the DAY TIME DATE and the user's comment line from Line 200.

Lines 390-400 Determines the log buffer size or character COUNT.

Lines 430-470 The main program loop. Tests to determine if the number of serial characters received is >50 or equal to COUNT. If either is true, the subroutine at Line 480 is called.

Lines 480-560 Transfers the characters from the PC's serial input to the disk output file. Before it transfers the characters, it issues an Xoff character to the Micro488/EX to prevent the PC's serial buffer from being overrun. It adjusts COUNT by the number of characters transferred and issues the Xon character before returning to the main loop.

Lines 570-800 Gets the output file name from the user. If a file extension is not specified, the extension 'LOG' is used. After the user enters the file name, it tests to determine if the file already exists by opening the file for input. If the file does not exist, an error is generated and the program vectors to Line 790. Line 800 causes the program to resume at the Line 750 which opens the new output file.

4.4.9 Saving the MACRO Buffers to Disk

The following program example reads and records all of the Micro488/EX's macro buffers and sends the information they contain to an ASCII text disk file for later recall or edit. The example is written in PC basic. Serial handshaking has not been implemented in this program. Basic maintains a 255 character buffer and most macros will contain less than 127 characters. Since the macro is completely transfered to disk prior to requesting a READ of another macro, serial input buffer overrun on the PC is unlikely. Only defined macros contained in the Micro488/EX are sent to the disk file.

Read Macro Buffers Program

```
100 '
110 ' Program to read and record all of the Micro488/EX's
120 '   macro buffers and save them to a disk file.
130 '       Serial control is set to Xon/Xoff
140 '                   IOtech, Inc.
150 '
160 ' Open the serial port for communications
170 OPEN "com1:9600,n,8,2" AS 1
180 PRINT#1,"@@";  ' reset the Micro488/EX
190 ' get file name
200 GOSUB 580 ' get the output file name
210 PRINT:PRINT "Enter File Comment => ";: LINE INPUT C$
220 ' loop until serial input buffer is empty
230 WHILE NOT EOF(1) : A$ = INPUT$(1,1) : WEND
240 '
250 ' set up file header
260 '
270 WHILE NOT EOF(1) : A$ = INPUT$(1,1) : WEND ' empty
serial input
280 PRINT#2,"*"
290 PRINT#2,"*  Macro Backup "
300 PRINT#1,"Hello"
310 LINE INPUT #1,H$
320 PRINT#2,"*  ";H$
330 PRINT#1,"Day Date Time"
340 LINE INPUT#1, D$
350 PRINT#2,"*  ";D$
360 PRINT#2,"*  ";C$
370 PRINT#2,"*": PRINT#2,""
380 '
390 ' get macro programs
400 '
410 PRINT:PRINT"Reading Macros"
420 PRINT#1,"Error Number"
430 INPUT #1,A$ ' get initial error number
440 FOR MACRO = 0 TO 99
450 PRINT "."; ' for visual indication
460 PRINT#1,"read ";MACRO
470 LINE INPUT #1,A$
480 IF A$ = "6" THEN GOTO 550 ' No Macro error number
490 PRINT#2," ": PRINT#2,"* Macro";MACRO ' comment line
495 PRINT#2,"Macro";MACRO : PRINT#2,A$
500 WHILE NOT EOF(1)
510 LINE INPUT #1,A$
520 IF A$ = "0" THEN GOTO 550 ' got the completed macro
530 PRINT#2,A$
540 WEND
550 NEXT MACRO
```

```
560 PRINT:PRINT:PRINT"Macro Backup Completed"
570 END
580 '
590 ' get output file name
600 '
610 CLS
620 PRINT : PRINT"Send Macro Buffer data to what file? => ";
630 LINE INPUT F$
640 IF F$ = "" THEN PRINT CHR$(7) : GOTO 610 ' try again
650 ' add extension if not included
660 IF INSTR(F$,".") THEN GOTO 670 ELSE F$ = F$ + ".mcr"
670 ' check if file already exists
680 ON ERROR GOTO 800
690 OPEN F$ FOR INPUT AS #2
700 CLOSE #2
710  PRINT:  PRINT  CHR$(7);"Replace  existing  ";F$;"  File?
(Y/N) ";
720 K$ = INKEY$ : PRINT K$;
730 IF K$ = "y" OR K$ = "Y" THEN GOTO 760
740 IF K$ = "n" OR K$ = "N" THEN GOTO 620
750 GOTO 720 ' try agin for valid key press
760 ' open the disk file
770 ON ERROR GOTO 0 ' turn error off
780 OPEN F$ FOR OUTPUT AS #2
790 RETURN
800 ' file does not exist error
810 RESUME 760
```

The following describes the functionality of the Read Macro program on a line by line basis.

Line `170`	Opens the `COM1` port on the PC for serial communications with the Micro488/EX.
Line `180`	Sends the `ID` command to the Micro488/EX to reset it to known power-on conditions. It utilizes the time delay provided by line `190`; get the output file name.
Line `200`	`GOSUB`s to Line `580` to get the output file name and open it.
Line `210`	Provides a method to include a user comment in the file.
Line `230`	Removes the power-on Xon character from the PC's serial input.
Lines `240-370`	Sets up the file header with comments. These comments include the `HELLO` response, the `DAY TIME DATE` and the user's comment line from Line `210`.
Lines `420-430`	Sets the `ERROR` reporting feature of the Micro488/EX to

NUMBER. This causes the interface to respond at the conculsion of each command with an error number. Line 430 accepts the error number reported as a result of the command in Line 420.

Lines 440-570 The main program loop. Requests a READ of all the available macro buffers in the Micro488/EX. If a macro does not exist, an error of 6, NO MACRO, is reported and the program continues to the next macro buffer number. Otherwise, a comment line is sent to the disk file along with each command line of the macro. This continues until an error code of 0, NO ERROR, is detected.

Lines 580-810 Gets the output file name from the user. If a file extension is not specified, the extension 'MCR' is used. After the user enters the file name, it tests to determine if the file already exists by opening the file for input. If the file does not exist, an error is generated and the program vectors to Line 800. Line 810 causes the program to resume at the Line 760 which opens the new output file.

The following shows a sample of a file that this program would generate. Maintaining macros in this format is a convenient method of editing and programming macros. Using this method, any text editor could be used to program macros.

Macro File Example

```
*
*  Macro Backup
*  Micro488/EX Revision 1.0 Copyright 1988 (C) IOtech Inc.
*  Friday July 29, 1988 12:35:04 PM
*  This is a comment line
*

* Macro 4
Macro 4
Sterm None
Date
Comment ','
Time
Comment ','
Sterm CR
Enter 16
Endm04
```

```
* Macro 10
Macro 10
Date Format MM/DD/YY
Time Format HH:MM:SS
Wait 2:00 PM
Domacro 4,10,5
Endm10
```

4.4.10 Restoring the MACRO Buffers From Disk

Once a read macro file has been created, it can be used to program the macros in the Micro488/EX. The next program example, RESTORE.BAS, resets the interface to FACTORY conditions and programs the macros from a file specified by the user. All lines of text included in the file which do not begin with a "*" comment marker are sent to the Micro488/EX. Comment Lines are printed to the PC's screen.

By re-setting the Micro488/EX to FACTORY, all data in the log buffer and any previously defined macros within the Micro488/EX are destroyed.

Restore Macro Program

```
100 '
110 ' Program to restore the Micro488/EX's macro buffers
120 ' to what was saved with the READMACRO.BAS program.
130 '       Serial control is set to Xon/Xoff
140 '                  IOtech, Inc.
150 '
160 ' Open the serial port for communications
170 OPEN "com1:9600,n,8,2" AS 1
180 PRINT#1,"@@";  ' reset the Micro488/EX
190 ' get restoration file name
200 GOSUB 480
210  PRINT:PRINT:PRINT  TAB(10)  CHR$(7);"CAUTION:  If  you
continue with this"
220 PRINT TAB(10) "program the Macro488's log buffer and any
previous macros"
230 PRINT TAB(10) "                  WILL BE LOST!!!!"
240  PRINT:PRINT:PRINT chr$(7)  ;  "Do  you  want  to  continue?
(Y/N) ";
250 K$ = INKEY$ : PRINT K$;
260 IF K$ = "y" OR K$ = "Y" THEN GOTO 330
270 IF K$ = "n" OR K$ = "N" THEN GOTO 290
280 GOTO 250 ' try again for valid key press
290 ' abort the program
300 CLOSE
```

```
310 PRINT:PRINT TAB(10) "Program Aborted"
320 END
330 PRINT:PRINT:PRINT"Resetting Micro488/EX Memory"
340 PRINT#1,"Factory"
350 FOR N = 1 TO 10000 : NEXT N
360 PRINT#1,"Save"
370 '
380 ' restore macros
390 '
400 PRINT: PRINT:PRINT"Restoring Macros"
410 WHILE NOT EOF(2)
420   LINE INPUT #2,A$
430    IF LEFT$(A$,1) = "*" THEN PRINT A$ : GOTO 450 ' skip
comments
440   PRINT#1,A$ : PRINT".";
450 WEND
460 PRINT:PRINT:PRINT"Macro Restoration Completed"
470 END
480 '
490 ' get file name
500 '
510 CLS
520 PRINT:PRINT"Restore Macros from what file? => ";
530 LINE INPUT F$
540 IF F$ = "" THEN PRINT : PRINT CHR$(7) : FILES : GOTO 520
550 IF INSTR(F$,".") THEN GOTO 560 ELSE F$ = F$ + ".mcr"
560 ' check if file already exists
570 ON ERROR GOTO 610
580 OPEN F$ FOR INPUT AS #2
590 ON ERROR GOTO 0 ' turn error off
600 RETURN
610 PRINT:PRINT CHR$(7);F$;" File Not Found " : PRINT
620 FILES
630 RESUME 520
```

The following describes the functionality of the Restore Macro program on a line by line basis.

Line `170`	Opens the `COM1` port on the PC for serial communications with the Micro488/EX.
Line `180`	Sends the `ID` command to the Micro488/EX to reset it to known power-on conditions. It utilizes the time delay provided by line `200`; get the output file name.
Line `200`	`GOSUB`s to Line `480` to get the output file name and open it.

Line 210	Cautions that all previous macros and log buffer data will be destroyed.
Lines 290-320	Executes if the user selects 'NO' to the caution.
Line 340	Issues the FACTORY command to reset all buffers.
Line 350	Delays to allow the Micro488/EX to become ready for more commands.
Line 360	Issues the SAVE command to clear the NVRAM ERROR condition caused by the FACTORY command.
Lines 370-470	The main program loop. Outputs to the Micro488/EX every text line in the input file except those lines which begin with a comment character (*). Comments are printed to the PC's screen.
Lines 480-630	Gets the input file name from the user. If a file extension is not specified, the extension 'MCR' is used. If the file specified does not exist, an error is generated and the program vectors to Line 610. Line 630 causes the program to resume at the Line 520 which request a new input file.

4.5 Restoring Lost Memory

If, during program development, you notice that the USER heap is shrinking and it is not due to new MACRO definitions or LOGed data, the FACTORY command can be used to restore all memory.

In the unlikely event that the FACTORY command does not work, remove the top cover following the directions and cautions in Section 2. With a small flat blade screw driver, un-seat the RAM IC (U103) from the battery socket (U100). Carefully, re-install the RAM IC.

Command Descriptions

5.1 Introduction

This section contains a detailed description of each of the high-level commands available for the Micro488/EX**.** There are two types of commands: bus commands and system commands. Bus commands communicate with the IEEE 488 bus. System commands configure or request information from the Micro488/EX.

Bus Commands:

ABORT	PASS CONTROL	REQUEST
CLEAR	PPOLL	RESUME
ENTER	PPOLL CONFIG	SEND
LOCAL	PPOLL DISABLE	SPOLL
LOCAL LOCKOUT	PPOLL UNCONFIG	TRIGGER
OUTPUT	REMOTE	

System Commands:

ARM	ERROR	SAVE
CASE	FACTORY	SET DATE
COMMENT	HELLO	SET DAY
COUNT	ID	SET TIME
DATE	LOG	STATUS
DATE FORMAT	LOG MEMORY	STERM
DAY	MACRO…ENDM	TERM
DAY FORMAT	MASK	TIME
DELAY	MEMORY	TIME FORMAT
DISARM	ON <event> DOMACRO	TIME OUT
DOMACRO	READ	TRACE
ERASE	READ LOG	WAIT
ERASE LOG	RESET	

5.2 Command Description Format

Each command description is divided into several areas:

5.2.1 Syntax

The syntax section of the command description describes the proper command syntax which must be sent to the Micro488/EX using the IBM BASIC `PRINT#` command, or its equivalent in other languages, to the COM port. The following conventions are used in the syntax descriptions:

No command, along with its options, may be more than 127 characters long. The `data` part of the `OUTPUT` command is not constrained by this length. It is, however, limited to the available USER MEMORY. The `OUTPUT #count;data` may contain be as long as necessary. Refer to the `OUTPUT` command for more information on this.

Items in capital letters, such as `ENTER` or `OUTPUT` must be used exactly as stated. Abbreviations may be used with some commands to reduce serial transmission traffic.

Items in lower case, such as `addr` or `count` represent parameters which must be substituted with an appropriate value.

Blank spaces in commands are generally ignored. Thus, `LOCAL LOCK OUT` is the same as `LOCALLOCKOUT`. Spaces are not ignored in four places: the `data` part of an `OUTPUT` command, within quoted strings in a `SEND` command, after an apostrophe (`'`) in a terminator specification (`term`), and after the semi-colon following the `ID` command.

The number sign character (`#`) and the semi-colon (`;`) must be present exactly as shown. A comma (`,`) represents an address separator. The oblique or slash character (`/`) or period (.) may be used in its place as the address separator.

Optional semicolons (`[;]`) may be used, if desired, for consistency with other IOtech products.

Items enclosed in square brackets (**[**`item`**]**) are optional. Multiple items enclosed in square brackets separated by vertical lines (**[**`item1`**|**`item2`**|**`item3`**]**) are optional, any one or none may be chosen. No more than one item may be selected.

Ellipses (**...**) within square brackets mean that the items in the brackets may be repeated as many times as desired. For example **[**`,addr`**...]** means that any number, to a maximum of 15, of address separator-address combinations may be used.

Braces, or curly brackets, (**{**`item1`**|**`item2`**}**) mean that exactly one of the enclosed items is required.

Combinations of brackets are possible. For example, **{**`term`**[**`term`**][**`EOI`**]|**`EOI`**}** allows the choice of `term`, `term EOI`, `term term`, `term term EOI`, or just `EOI`, but does not allow the choice of "nothing."

Numeric parameters (those that are given as numbers) are decimal unless preceded by `&H` in which case they considered to be hexadecimal. Thus `100` is decimal `100`, `&H64` is hexadecimal `64` which equals decimal `100`, `&HFF` is decimal `255`, and `0FF` is invalid because `F` is not a valid decimal digit. The only exception to this rule is that bus addresses, both primary and secondary, must be specified as two-digit decimal numbers. Hexadecimal bus addresses are not allowed.

Several of the commands require additional or optional parameters. These are further described with each command, but discussion of the more common ones follow.

5.2.1.1 Bus Addressing

`pri-addr` A two-digit primary device address in the range of `00` to `30`.

`sec-addr` An optional two-digit secondary device address in the range of `00` to `31`.

`addr` An IEEE bus address. A numeric primary address optionally followed by a secondary address. Thus addr is of the form…

`{pri-addr[sec-addr]}`

where `pri-addr` is a two-digit primary address in the range from `00` through `30` and `sec-addr` is a two-digit secondary address from `00` through `31`. Numeric addresses must be given as two-digit numbers, e.g. `05` for address 5, and `1601` for primary address 16, secondary address 1

`[,addr…]` An optional list of bus addresses, each one preceded by an address separator; either a comma (`,`) , a slash (`/`) or a period (.).

No more than 15 bus addresses are allowed in any single command.

5.2.1.2 Character Count

`#count` The number of characters to be transferred. A pound sign (#) followed by an integer in the range of 1 to 65535 (2^{16}-1). May be specified in hexadecimal by preceding it with `&H`. The hexadecimal range is &H1 to &HFFFF. A character count of zero is invalid.

5.2.1.3 ASCII Characters

`$char` A single character whose ASCII value is the number `char`, a decimal number in the range of 0 to 255 or a hexadecimal number in the range of &H0 to &HFF. For example, $65 is the letter "A", as is &H41.

`CR` The carriage return character ($13, &H0D).

`LF` The line feed character ($10, &H0A).

`'X` Any printable character. The apostrophe is immediately followed, without any intervening spaces, by a single character which is taken to be the character specified.

5.2.1.4 ASCII Character Strings

`data` An arbitrary string of characters. None of the special forms given above (`$char`, `CR`, `LF`, or `'X`) are used. For example, CRLF as `data` is taken as the letters, "C", "R", "L", and "F", not as carriage return line feed.

`'data'` An arbitrary string of characters enclosed in apostrophes (') or quotes(").

5.2.1.5 Terminators

`term` Any single character, specified as `CR`, `LF`, `'X`, or `$char` as described previously. Part of terminator sequence used to mark the end of lines of data and commands.

`[term]` An optional term character. `term[term]` means that one or two terminators may be specified.

`EOI` The IEEE bus End-Or-Identify signal. When asserted during the transfer of a character, EOI signals that that character is the last in the transfer. On input, EOI, if specified, causes the input to stop. On output, EOI causes the bus EOI signal to be asserted

during transmission of the last character transferred.

NONE — The no end-of-line character indicator. When STERM NONE is specified, Micro488/EX does not append any serial output terminator(s) to serially transmitted data.

5.2.2 Response

The response section of the command description describes the response that the user's program should read from the serial host's COM port after sending the command. If a response is provided, it must be read to maintain proper program sequence.

The ERROR command may be used to provide responses for the commands listed as NONE. Refer to the ERROR command description for details.

5.2.3 Mode

This section of the command description specifies the operating modes in which the command is valid. The Micro488/EX may be configured as the System Controller, in which case it will initially be the Active Controller, or as a Not System Controller, in which case it will initially be in the Peripheral state. The Micro488/EX configuration as System Controller or Not System Controller is fixed by a hardware switch setting and cannot be changed by software, but the Micro488/EX can change between Active Controller and Peripheral as required (see Section 3).

The modes are referred to by their names and states, as given in the table below:

Description	State
System Controller	SC
Not System Controller	*SC

Active Controller	CA
Peripheral (Not Active Controller)	*CA
Active System Controller	SC•CA
System Controller, Not Active Controller	SC•*CA
Not System Controller, Not Active Controller	*SC•*CA
Not System Controller, Active Controller	*SC•CA

5.2.4 Bus States

This section describes the bus command and data transfers using IEEE bus mnemonics abbreviated as follows:

		DIO lines							
		8	7	6	5	4	3	2	1
ATN	Attention								
data	Data String								
DCL	Device Clear	x	0	0	1	0	1	0	0
GET	Group Execute Trigger	x	0	0	0	1	0	0	0
GTL	Go To Local	x	0	0	0	0	0	0	1
IFC	Interface Clear								
LAG	Listen Address Group	x	0	1	a	d	d	r	n
LLO	Local Lock Out	x	0	0	1	0	0	0	1
MLA	My Listen Address	x	0	1	a	d	d	r	n
MTA	My Talk Address	x	1	0	a	d	d	r	n
PPC	Parallel Poll Configure	x	0	0	0	0	1	0	1
PPD	Parallel Poll Disable	x	1	1	1	0	0	0	0
PPE	Parallel Poll Enable	x	1	1	0	S	P3	P2	P1
PPU	Parallel Poll Unconfigure	x	0	0	1	0	1	0	1
REN	Remote Enable								
SDC	Selected Device Clear	x	0	0	0	0	1	0	0
SPD	Serial Poll Disable	x	0	0	1	1	0	0	1
SPE	Serial Poll Enable	x	0	0	1	1	0	0	0
SRQ	Service Request								
TAG	Talker Address Group	x	1	0	a	d	d	r	n
TCT	Take Control	x	0	0	0	1	0	0	1
UNL	Unlisten	x	0	1	1	1	1	1	1
UNT	Untalk	x	1	0	1	1	1	1	1

(x = "don't care")

If a command is preceded by an asterisk then that command is unasserted. For example, *REN states that the remote enable line is unasserted. Conversely, REN without the asterisk states that the line becomes asserted.

5.2.5 Examples

This section gives programming examples written in the BASIC language.

5.3 The Commands

The commands provided in the Micro488/EX, in alphabetical order, are described on the following pages.

@

The system command @, followed by a CR and/or LF, is used to unlock the Micro488/EX from an inappropriate command. An example of such a command would be requesting data from a nonexistent device with time outs disabled. It is not recommended for use but is provide in the Micro488/EX for upward compatability with other IOtech serial to IEEE controllers. Timeouts are a better method of signaling bus faults.

When the @ command is received, the serial handshake line (RTS) is un-asserted. It is asserted when the Micro488/EX is capable of buffering commands. If XON/XOFF handshake is selected, the software handshake state is not modified.

Issuing the @ command installs the power-on formats for DATE, DAY, TIME and CASE and software programmable terminators are returned to the power-on conditions. Macro and log buffer data is not effected. If a macro has been defined as the STARTUP, it will execute. It also is equivalent to issuing the following commands…

```
DISARM      MASK OFF
ERROR OFF   REQUEST 0 (with *SRQ)
ID;@        TIME OUT 0
LOG OFF     TRACE OFF
```

The @ character, referred to as the 'ID' character, can be changed or disabled by using the ID command. If it is anticipated that the ID character may be part of the data within an OUTPUT or SEND command, it should be disabled. It is also recommended that the ID character not be involked within a macro.

SYNTAX @

RESPONSE None

MODE Any

BUS STATES None

EXAMPLE PRINT #1,"@"

@@

Sending the system command @@ causes the Micro488/EX to return to power-on conditions. Serial input (pending commands) and serial output (pending data) are cleared, and any software programmable terminators are returned to the power-on conditions. If a macro has been assigned as the STARTUP, it will execute immediately upon command completion. Macro and log buffers are not effected by this command.

This is the only command which does not require a serial terminator to execute. Reset is executed upon receipt of the second @.

When the @@ command is received, the serial handshake line (RTS) is un-asserted. It is asserted when the Micro488/EX is capable of buffering commands. If XON/XOFF handshake is selected, the software handshake state is reset and an XON character is transmitted when the Micro488/EX is capable of buffering commands.

The @ character, referred to as the 'ID' character, can be changed or disabled by using the ID command. If it is anticipated that the ID character may be part of the data within an OUTPUT or SEND command, it should be disabled.

SYNTAX @@

RESPONSE None

MODE Any

BUS STATES: IFC,*IFC (SC)

EXAMPLE: PRINT #1,"@@"

ABORT

As the System Controller (SC), whether the Micro488/EX is the Active Controller or not, the ABORT command causes the Interface Clear (IFC) bus management line to be asserted for at least 500 microseconds. By asserting IFC, the Micro488/EX regains control of the bus even if one of the devices has locked it up during a data transfer. Asserting IFC also makes the Micro488/EX the Active Controller. If a Non-System Controller was the Active Controller, it will be forced to relinquish control to the Micro488/EX. ABORT forces all IEEE bus device interfaces into a quiescent idle state.

If the Micro488/EX is a Non System Controller in the Active Controller state (*SC•CA), it asserts attention (ATN), which halts any bus transactions, and sends its talk address to "untalk" any other talkers on the bus. It does not (and cannot) assert IFC if in the *SC state.

SYNTAX	ABORT or AB	
RESPONSE	None	
MODE	SC or *SC•CA	
BUS STATES	IFC, *IFC ATN•MTA	(SC) (*SC•CA)
EXAMPLES	PRINT#1,"ABORT"	
	PRINT#1,"AB"	Using abbreviated form

ARM

The ARM command allows the Micro488/EX to automatically send event messages to the serial host when one or more of the specified events occur. The event messages that are returned are the same non-abbreviated strings as those used to program the events.

There are two types of events, level sensitive and edge sensitive. Level sensitive events, such as SRQ, will be reported every time they are ARMed while the event condition persists. Usually, some action must be taken (eg SPOLL) to clear the condition prior to re-issuing the ARM. Edge sensitive events, such as TRIGGER, are cleared when reported.

Regardless of the event sensitivity, the ARM command must be re-sent after the event message is reported to re-activate the ARMed condition. The optional events include…

SRQ — The event message 'SRQ' is returned to the serial host when the state of the Service Request Bus Line is detected in the asserted state. This event is level sensitive. If the condition exists at the time the ARM SRQ command is issued, the Micro488/EX will return the event message immediately. If the ARM command is issued without any specified events, the SRQ event is assumed. This provides upward compatibility with the Micro488/EX in previous Micro488 systems.

PERIPHERAL — The event message 'PERIPHERAL' is returned to the serial host when the Micro488/EX is forced from the Controller Active State (*SC•CA) to the Peripheral State (*SC•*CA) by receipt of IFC from the System Controller. This can be useful in detecting receipt of IFC when in the *SC•CA state. This event is edge sensitive.

CONTROLLER — The event message 'CONTROLLER' is returned to the serial host when the Micro488/EX receives control of the bus and transitions from the Peripheral State (*SC•*CA) to the Controller Active State (*SC•CA). This occurs when the Take Control interface message is received by the Micro488/EX. This event is edge sensitive.

`TRIGGER` The event message '`TRIGGER`' is returned to the serial host when the Micro488/EX, as a Peripheral (*CA), receives a Group Execute Trigger (GET) command from the Active Controller. This event is edge sensitive. When the event message is sent, the internal status, as read by the `STATUS 1` command, is cleared.

`CLEAR` The event message '`CLEAR`' is returned to the serial host when the Micro488/EX, as a Peripheral (*CA), receives a Device Clear (DCL) or a Selected Device Clear (SDC) command from the Active Controller. This event is edge sensitive. When the event message is sent, the internal status, as read by the `STATUS 1` command, is cleared.

`TALK` The event message '`TALK`' is returned to the serial host when the Micro488/EX, as a Peripheral (*CA), detects its My Talk Address (MTA) command from the Active Controller. It indicates that the controller has requested information from the Micro488/EX. This event is edge sensitive.

`LISTEN` The event message '`LISTEN`' is returned to the serial host when the Micro488/EX, as a Peripheral (*CA), detects its My Listen Address (MLA) command from the Active Controller. It indicates that the controller has information it wants to send to the Micro488/EX. This event is edge sensitive.

`IDLE` The event message '`IDLE`' is returned to the serial host when the Micro488/EX, as a Peripheral (*CA), transitions from a Talker or Listener state to an idle state (neither talker or listener). It indicates that the controller has unaddressed the Micro488/EX with either an UNT or UNL command. This event is edge sensitive. When the event message is sent, the internal address change status, as read by the `STATUS 1` command, is cleared. This event is tested prior to the `CHANGE` event and will clear the internal `CHANGE` status only when the addressed to un-addressed transition occurs.

`CHANGE` The event message '`CHANGE`' is returned to the serial host when the Micro488/EX, as a Peripheral (*CA), detects an addressed state change. This occurs on transitions from a Talker or Listener state to an idle state (neither talker or listener), or from an idle state to a talker or listener state. This event is edge sensitive. When the event

message is sent, the internal status, as read by the STATUS 1 command, is cleared. This event is tested after the IDLE event. If IDLE is ARMed, the CHANGE event will only be reported when an un-addressed to an addressed transition occurs.

ERROR — The event message 'ERROR' is returned to the serial host when the Micro488/EX detects an error condition. The error condition may be an un-recognized command from the serial host, an invalid parameter or a bus error. Refer to Appendix B for a listing of the error conditions which can be detected and reported by the Micro488/EX. This event is level sensitive. When the event message is sent, the error status, as read by any of the STATUS commands, must be read prior to ARMing this event again.

Once a condition is ARMed it remains ARMed until it is DISARMed, the event specified has occurred or until the Micro488/EX is reset.

The ARM and ON <event> DOMACRO commands are mutually exclusive. The last command issued takes precedence.

SYNTAX

ARM [;] [event[event…]]
or
AR [;] [event[event…]]

where event… may include…

Event	Abbr Form
SRQ	SRQ
PERIPHERAL	PE
CONTROLLER	CO
TRIGGER	TR
CLEAR	CL
TALK	T
LISTEN	L
IDLE	I
CHANGE	CH
ERROR	ER

If no event is specified, ARM SRQ is assumed for upward compatibility with other IOtech products.

RESPONSE Event string sent when event occurs

MODE any

BUS STATES None

EXAMPLE

```
10 PRINT#1,"ARM TALK"          Enable Talk Condition
20 INPUT#1,A$                  Input 'TALK' Status Message
30 PRINT#1,"OUTPUT;This is a test"
40 GOTO 10                     Output  data and try again
```

CASE

The CASE command sets whether the DAY or DATE text is output in UPPER case or capitalized LOWER case. The power-on factory default is LOWER case. This power-on default can be modified with the SAVE command. Refer to the SAVE command for details.

SYNTAX CASE **[;]{**UPPER | LOWER**}**

UPPER specifies all DAY and DATE text to be output in upper case.
LOWER specifies all DAY and DATE text to be output in capitalized lower case.

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"DAY"
INPUT #1,D$      Read the day string
PRINT D$         Printed to the screen
Monday

PRINT#1,"CASE UPPER"
PRINT#1,"DAY"
INPUT #1,D$      Read the day string
PRINT D$         Printed to the screen
MONDAY
```

CLEAR

The CLEAR command causes the Device Clear (DCL) bus command to be issued by the Micro488/EX. If the optional addresses are included, the Selected Device Clear (SDC) command is issued to all specified devices. IEEE 488 bus devices which receive a Device Clear or Selected Device Clear command normally reset to their power-on state.

SYNTAX
```
CLEAR [addr[,addr…]]
```
or
```
CL [addr[,addr…]]
```

addr is a device address (primary with optional secondary).
, is the address separator, either a comma, a slash [/] or a period [.].

RESPONSE None

MODE CA

BUS STATES		
	ATN•DCL	(all devices)
	ATN•UNL,MTA,LAG,SDC	(selected devices)

EXAMPLES		
	`PRINT #1,"CLEAR"`	Issue a Device Clear to all devices.
	`PRINT #1,"CL 12,18"`	Issue a Selected Device Clear to devices 12 and 18.

COMMENT

The COMMENT command is provided to allow the user to place comment lines in a Macro buffer. The COMMENT string is enclosed in either apostrophes (') or quotation marks ("). When the Macro is executed, the COMMENT string is sent to the serial host, or if LOG ON is in effect it is sent to the LOG Buffer, appended with the serial output terminators. The serial output terminators may be suppressed by including a back-slash [\] as the last character of the COMMENT string.

SYNTAX

```
COMMENT [;] 'data'
```
or
```
COM [;] 'data'
```

'data' is an ASCII string delimited by apostrophes or quotation marks.

RESPONSE data is returned to the serial host

MODE Any

BUS STATES None

EXAMPLES

```
PRINT #1,"COMMENT 'This is a test comment'"
INPUT #1, C$     Read the comment string
PRINT C$         Print it to the screen
This is a test comment

PRINT #1,"COMMENT 'Available Memory = \'"
PRINT #1,"MEMORY"
INPUT #1, C$     Read the comment string with suppressed
                 serial output terminators and appended
                 memory value
PRINT C$         Print it to the screen
Available Memory = 29782
```

COUNT

The COUNT command returns the loop count, appended with the serial output terminator(s), of the last invoked Macro buffer. The number returned is the remaining number of loops left to execute. It, therefore, is a decrementing count. If Macro 1 requests a COUNT then calls Macro 0, any subsequent COUNT requests made by Macro 1 will return the Macro 0 loop COUNT. The following contents of Macro 1's buffer should illustrate this…

```
MACRO 1          Creates Macro Buffer #1
COUNT            This will return Macro #1's loop count
DOMACRO 0        This command executes Macro #0
COUNT            This will return Macro #0's loop count since it
                 was the last Macro to be invoked
ENDM01
```

If Macro #1, in the previous example, was invoked multiple times, each time the loop counts will be reported as described. This is due to the fact that looping is a re-invocation of the looped Macro. This command is only valid when contained within a Macro. Execution outside of a Macro will generate an 'INVALID COMMAND' error.

SYNTAX COUNT

RESPONSE numeric loop count (1 to 255) of last invoked Macro buffer.

MODE Any

BUS STATES None

EXAMPLE

```
10 PRINT #1,"MACRO"          Build Macro #0
20 PRINT #1,"COMMENT 'Loop Number =\'"
30 PRINT #1,"COUNT"
40 PRINT #1,"ENDM"
50 PRINT #1,"DOMACRO0,5"     Execute the Macro five times
60 FOR N = 1 TO 5
70 INPUT #1,L$:PRINT L$      Read Comment & Count
80 NEXT N
```

DATE

The DATE command outputs the internal clock date in the format determined by the DATE FORMAT command, followed by the programmed serial output terminators. Refer to the DATE FORMAT command for a list of the available format options. This date string can also include optional DAY and TIME information. If DAY or TIME information is requested, they are appended to the DATE information in the same order, delimited by spaces.

SYNTAX DATE **[** DAY | TIME | DAY TIME | TIME DAY **]**

DATE returns the date in the format determined by the DATE FORMAT command.
The optional DAY returns the day of the week in the format determined by the DAY FORMAT command.
The optional TIME returns the time in the format determined by the TIME FORMAT command.

RESPONSE Returns date and optionally day of week and time information

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"DATE"
INPUT #1,D$      Read the date string
PRINT D$         Printed to the screen
11-04-52         Output is format dependent

PRINT#1,"DATE DAY TIME"
INPUT #1,D$      Read the date, day and time string
PRINT D$         Printed to the screen
11-04-52 Monday 12:31 PM
                 Output is format dependent
```

DATE FORMAT

The DATE FORMAT command allows the user to select the format of the DATE command response. Several different formats are provide, including both American and European. DATE FORMAT is also used to determine how the date information will be interpreted when the date is set with the SET DATE command. Refer to the SET DATE command definition for more details. The month text can be sent as upper or capitalized lower case, as determined by the CASE command.

The power-on factory default is MONTH DD, YYYY. This default can be modified with the SAVE command. Refer to the SAVE command for details.

SYNTAX DATE FORMAT **[;] {**option**}**

option includes at one of the following format specifiers…

option	Example Output	Length†	Style
MM/DD/YY	12/04/76	8 bytes fixed	American
MM-DD-YY	12-04-76	8 bytes fixed	American
MM/DD/YYYY	12/04/1976	10 bytes fixed	American
MM-DD-YYYY	12-04-1976	10 bytes fixed	American
MON DD YYYY	DEC 04 1976	11 bytes fixed	American
MON DD, YYYY	DEC 04, 1976	12 bytes fixed	American
MONTH DD YYYY	DECEMBER 4 1976	variable	American
MONTH DD, YYYY	DECEMBER 4, 1976	variable	American
DD/MM/YY	04/12/76	8 bytes fixed	European
DD-MM-YY	04-12-76	8 bytes fixed	European
DD/MM/YYYY	04/12/1976	10 bytes fixed	European
DD-MM-YYYY	04-12-1976	10 bytes fixed	European
DD MON YYYY	04 DEC 1976	11 bytes fixed	European
DD MONTH YYYY	4 DECEMBER 1976	variable	European

† Length does not include programmable serial output terminators.

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"DATE FORMAT MONTH DD, YYYY"
PRINT#1,"DATE"
INPUT #1,D$          Read the date string
PRINT D$             Printed to the screen
November 4, 1952     Date output is date dependent
```

```
PRINT#1,"DATE FORMAT MON DD, YYYY"
PRINT#1,"DATE"
INPUT #1,D$          Read the date string
PRINT D$             Printed to the screen
Nov 04, 1952         Date output is date dependent
```

```
PRINT#1,"DATE FORMAT MM/DD/YY"
PRINT#1,"DATE"
INPUT #1,D$          Read the date string
PRINT D$             Printed to the screen
11/04/52             Date output is date dependent
```

DAY

The DAY command outputs the internal clock day of week in the format determined by the DAY FORMAT command, followed by the programmed serial output terminators. Refer to the DAY FORMAT command for a list of the available format options. This day of week string can also include optional DATE and TIME information. If DATE or TIME information is requested, they are appended to the DAY information in the same order, delimited by spaces.

SYNTAX `DAY [DATE | TIME | DATE TIME | TIME DATE]`

DAY returns the day of the week in the format determined by the DAY FORMAT command.
The optional DATE returns the date in the format determined by the DATE FORMAT command.
The optional TIME returns the time in the format determined by the TIME FORMAT command.

RESPONSE Returns day of week and optionally date and time information

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"DAY"
INPUT #1, D$      Read the date string
PRINT D$          Printed to the screen
Monday            Output is format dependent

PRINT#1,"DAY TIME DATE"
INPUT #1, D$      Read the date, day and time string
PRINT D$          Printed to the screen
Monday 12:31 PM 11-04-52
                  Output is format dependent
```

DAY FORMAT

The DAY FORMAT command allows the user to select the format of the DAY command response. Several different formats are provide including numeric, abbreviated text and un-abbreviated text. The power-on factory default is un-abbreviated text. This default can be modified with the SAVE command. Refer to the SAVE command for details. The day of week text can be sent as upper or capitalized lower case, as determined by the CASE command.

SYNTAX DAY FORMAT **[**;**]** **{**D | DA | DAY**}**

The option specifies one of the following formats…

options=	D	DA	DAY
	1	SUN	SUNDAY
	2	MON	MONDAY
	3	TUE	TUESDAY
	4	WED	WEDNESDAY
	5	THU	THURSDAY
	6	FRI	FRIDAY
	7	SAT	SATURDAY

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"DAY FORMAT D"
PRINT#1,"DAY"
INPUT #1,D$          Read the day string
PRINT D$             Printed to the screen
3                    Output is day dependent
```

```
PRINT#1,"DAY FORMAT DA"
PRINT#1,"DAY"
INPUT #1,D$          Read the day string
PRINT D$             Printed to the screen
TUE                  Output is day dependent

PRINT#1,"DAY FORMAT DAY"
PRINT#1,"DAY"
INPUT #1,D$          Read the day string
PRINT D$             Printed to the screen
TUESDAY              Output is day dependent
```

DELAY

The DELAY command is provided to allow the user to place time delays in the execution of a Macro. The amount of time delayed is specified in seconds in the range of 0 to 65535 (2^{16}) seconds. Although the DELAY interval is accurate to 0.01% ± 5 milliseconds, it is not a function of the internal clock on the Micro488/EX. Therefore, for delays of 65,000 seconds a shift of up to 6.5 seconds may be noticed relative to the internal clock.

SYNTAX `DELAY [;] time`

`time` is specified in seconds,0 to 65535 (2^{16})

RESPONSE None

MODE Any

BUS STATES None

EXAMPLE
```
PRINT#1,"DELAY 20"
PRINT #1,"COMMENT 'I am back'"
INPUT #1, C$      Read the comment string
PRINT C$          Printed to the screen 20 seconds later
```

DISARM

The DISARM command prevents the Micro488/EX from sending the event's status message to the serial host, even when the specified conditions occur. It is also used to disable the ON <event> DOMACRO response. The user's program can still check for the events by using the STATUS 1 command.

If the DISARM command is invoked without specifying any events, then all events will be disabled.

The ARM or ON <event> DOMACRO commands may be used to re-enable the event responses.

SYNTAX DISARM **[;] [**event**[**event**…]]**
or
DI **[;] [**event**[**event**…]]**

event is one of SRQ, PERIPHERAL, CONTROLLER, TRIGGER, CLEAR, TALK, LISTEN, IDLE, CHANGE, ERROR or STARTUP.

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

PRINT#1,"DISARM"	Disable all conditions
PRINT#1,"DISARM SRQ"	Do not respond to SRQ

DOMACRO

This command is used to execute the contents of a Macro buffer. A 'NO MACRO' error will occur if the specified Macro buffer is empty. If the optional macro buffer number is omitted, Macro 0 is assumed.

By specifying an optional count, the same Macro will execute count number of times, up to 255. If the optional interval is specified, the macro will execute count number of times with interval seconds (up to 65535 seconds) between the start of each invocation. If macro execution is longer than the interval time, the macro will execute immediately. Only one interval timer is available. If it is already committed to one macro and another DOMACRO command is issued with an interval specified, a 'TIMER IN USE' error will occur. The interval time is determined by the internal clock.

Macros can execute other Macros but can not execute already executing Macros. If a Macro tries to invoke itself or an already executing Macro, a 'MACRO RECURSION' error will be generated.

Refer to the Macro command for a description of the Macro features.

SYNTAX DOMACRO **[**;**] [**number**[**,count **[**,interval **]]]**
or
DO **[**;**] [**number**[**,count **[**,interval **]]]**

number is a Macro buffer number, from 0 to 99. If omitted, buffer #0 is assumed.
count is the number of times to execute the Macro from 1 to 255.
interval is the time interval, 1 to 65535, in seconds between the start of successive count macro invocations.

RESPONSE Dependent on the contents of the Macro buffer.

MODE any

BUS STATES Defined within the specified Macro buffer commands

EXAMPLES

`PRINT #1;"DOMACRO"`	Execute the commands in Macro0 Buffer.
`PRINT #1;"DO20"`	Execute the commands in Macro20 Buffer.
`PRINT #1;"DO6,30"`	Execute the commands in Macro6 Buffer 30 times.
`PRINT #1;"DO6,30,3600"`	Execute the commands in Macro6 Buffer 30 times at 1 hour intervals

ENTER (Controller mode)

The ENTER command reads data from the IEEE bus. If a device address (with optional secondary address) is specified, that device will be addressed to talk. If no address is specified, the Micro488/EX must already be configured to receive data, either as a result of an immediately preceding ENTER command, or as a result of a SEND sub-command. A time-out error will occur (if enabled) if the Micro488/EX does not receive a data byte within the time out period after issuing the ENTER command.

If the character count, count, is specified, then exactly that number of characters will be read from the device with the serial output terminators appended. Otherwise, ENTER terminates reception on detection of the line feed (LF) character, which may be overridden by specifying the terminator in the ENTER command.

If a terminator, term, option is specified, all CR and LF characters in the input data are unconditionally discarded. When the specified terminator is detected, it is discarded and replaced with the serial terminator(s) before being returned to the serial host. The optional terminator applies ONLY to the ENTER command it is sent with. The terminator returns to a Line Feed on subsequent ENTER commands.

If the EOI option is specified, all characters are returned to the host until the EOI line is detected. The character sent with EOI asserted is also returned followed by the serial output terminator(s).

SYNTAX `ENTER[addr][#count|term|EOI|;count|;term|;EOI]`
or
`EN[addr][#count|term|EOI|;count|;term|;EOI]`

`addr` is the IEEE bus device address.
`count` is the number of characters to ENTER.
`term` and EOI override the normal IEEE bus input LF terminator.

RESPONSE Device-dependent data. If `count` is specified, then `count` characters will be returned followed by the serial output terminators. Otherwise the response ends when the IEEE bus input terminator is detected and the serial output terminators are appended to the returned data.

MODE CA

BUS STATES ATN•UNL,MLA,TAG,*ATN,data…,ATN (With `addr`)
*ATN, data…, ATN (Without `addr`)

EXAMPLES

```
PRINT#1,"ENTER16"
INPUT#1,A$
```
Read data from device 16.

```
PRINT#1,"ENTER16"
LINE INPUT#1,A$
```
Read an entire line of data from device 16 even if it contains commas or other punctuation.

```
PRINT#1,"ENTER16;CR"
INPUT#1,A$
```
Read data from device 16 until CR is encountered.

```
PRINT#1,"ENTER16 EOI"
INPUT#1,A$
```
Read data until EOI is detected.

```
PRINT#1,"ENTER 0702"
INPUT#1,A$
```
Read data from device 7, secondary address 2.

```
PRINT#1,"EN 12 #5"
A$=INPUT$(5,#1)
```
Read 5 bytes from device 12. `INPUT$` gets 5 bytes from file #1

```
PRINT#1,"ENTER #20"
A$=INPUT$(20,#1)
```
Read 20 more bytes.

```
PRINT#1,"ENTER ;20"
A$=INPUT$(20,#1)
```
Read 20 more bytes.

ENTER (Peripheral mode)

In Peripheral mode, the ENTER command receives data from the bus under control of the Active Controller. The Active Controller must put the Micro488/EX into the Listen state and configure some bus device to provide the Micro488/EX with data. The Listen state can be checked with the STATUS 1 command, can cause a reported event message with the ARM command, or can force a Macro execution with the ON <event> DOMACRO command. A time-out error will occur (if enabled) if the Micro488/EX does not receive a data byte within the time out period after issuing the ENTER command.

If the character count, count, is specified, then exactly that number of characters will be read from the device with the serial output terminators appended. Otherwise, ENTER terminates reception on detection of the line feed (LF) character, which may be overridden by specifying the terminator in the ENTER command.

If a terminator, term, option is specified, all CR and LF characters in the input data are unconditionally discarded. When the specified terminator is detected, it is discarded and replaced with the serial terminator(s) before being returned to the serial host. The optional terminator applies ONLY to the ENTER command it is sent with. The terminator returns to a Line Feed on subsequent ENTER commands.

If the EOI option is specified, all characters are returned to the host until the EOI line is detected. The character sent with EOI asserted is also returned followed by the serial output terminator(s).

SYNTAX ENTER[#count|term|EOI|;count|;term|;EOI]
or
EN [#count|term|EOI|;count|;term|;EOI]

count is the number of characters to ENTER.
term and EOI override the normal IEEE bus input LF terminator.

RESPONSE Device-dependent data. If count is specified, then count characters will be returned followed by the serial output terminators. Otherwise the response ends when the IEEE bus input terminator is detected and the serial output terminators are appended to the returned data.

MODE *CA

BUS STATES Determined by the Active Controller

EXAMPLES

```
PRINT#1,"ENTER"
INPUT#1,A$
```
Read data into A$ until the default bus input terminator is detected.

```
PRINT#1,"ENTER CR"
INPUT#1,A$
```
Read data until CR is encountered.

```
PRINT#1,"EN $000"
INPUT#1,A$
```
Read data until a NULL is encountered.

```
PRINT#1,"ENTER EOI"
INPUT#1,A$
```
Read data until EOI is detected.

```
PRINT#1,"ENTER #5"
A$=INPUT$(5,#1)
```
Read 5 bytes.
INPUT$ gets 5 bytes from file #1

ERASE

The ERASE command is used to delete previously defined Macro Buffers and return the memory they occupy back to the USER heap. Macro Buffers can be individually specified by their number, or all Macro Buffers may be erased an returned by a single ERASE command without a buffer number specifier.

SYNTAX `ERASE[;][number]`

`number` is a Macro Buffer number from `0` to `99`. If not specified, all Macro Buffers are erased.

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

`PRINT#1,"ERASE"`	Erase all Macro buffers
`PRINT#1,"ERASE 20"`	Erase Macro buffer #20

ERASE LOG

The ERASE LOG command empties the non-volatile LOG buffer of any data that it contained. The memory occupied by the LOG buffer is returned to the USER heap.

SYNTAX ERASE LOG

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"MACRO 1"          Define a macro
PRINT#1,"HELLO"
PRINT#1,"ENDM"
PRINT#1,"LOG ON"              Enable logging
PRINT#1,"DOMACRO 1"        Execute the macro
                           The response is logged
PRINT#1,"LOG MEMORY"
INPUT #1,L                 Read the memory used
PRINT L                    Printed to the screen
54                         Number of bytes in buffer
PRINT#1,"ERASE LOG"        Erase the log buffer
PRINT#1,"LOG MEMORY"
INPUT #1,L                 Read the memory used
PRINT L                    Printed to the screen
0                          0 bytes in buffer
```

ERROR

The ERROR command enables or disables automatic reporting of the Micro488/EX error messages on command completion. ERROR MESSAGE enables error message string reporting, ERROR NUMBER enables error message number reporting and ERROR OFF disables it. ERROR OFF is the default condition.

While Macros are executing, error reporting is suspended until Macro completion.

SYNTAX ERROR[;]{MESSAGE|NUMBER|OFF}

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"ERROR OFF"          Disable error message reporting.
PRINT#1,"ERROR MESSAGE"      Enable error message reporting.
PRINT#1,"ERROR NUMBER"       Enable error number reporting.
```

HELLO

The HELLO command is used to verify communication with the Micro488/EX, and to read the software revision number. When the command is sent, the Micro488/EX returns a string similar to the following:

```
Micro488/EX Revision N.N Copyright (C) 1988 IOtech Inc.
```

where N.N is the revision and release number of the firmware.

SYNTAX	HELLO or HE	
RESPONSE	Micro488/EX Revision N.N Copyright (C) 1988 IOtech Inc. N.N is the revision and release number of the firmware.	
MODE	Any	
BUS STATES	None	

EXAMPLE	`PRINT#1,"HELLO"`
	`INPUT#1,A$`
	`PRINT A$`
		Get the HELLO response
		
		and display it.

ID

The ID command allows the user to change the @ or the @@ command character to any printable ASCII character. If the double ID character command is issued, the ID character will default back to '@'.

The desired character must immediately follow the semi-colon without intervening spaces. The @ and @@ command can be disabled by not including, eg ending the command with either a CR or LF character, the character following the required semicolon. It can be re-enabled again by issuing the ID command with a valid character.

If the ID character and TIME OUTs are disabled, and an invalid bus address is specified within a command, the only way to recover control is by re-powering the interface. If you anticipate that the data part of an OUTPUT or SEND command may contain the presently programmed ID character, it should be disabled.

SYNTAX `ID;[ASCII]`

`ASCII` is any printable ASCII character immediately following the semi-colon (`;`)

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES `PRINT #1,"ID;#"` change the ID character to #

`PRINT #1,"ID;"` disable the ID commands

`PRINT #1,"ID;@"` re-enable the ID character to @

LOCAL

In the System Controller mode, the LOCAL command without optional addresses causes the Micro488/EX to un-assert the Remote Enable line. This causes devices on the bus to return to manual operation. As the Active Controller, with bus addresses specified, bus devices are placed in the local mode by the Go To Local (GTL) bus command. If addresses are specified, the state of the Remote Enable line is not affected.

SYNTAX	LOCAL or LO	
RESPONSE	None	
MODE	SC	
BUS STATES	*REM	
EXAMPLE	PRINT#1,"LOCAL"	Un-assert the REN Line

SYNTAX	LOCAL addr[,addr…] or LO addr[,addr…] addr is a bus device address.	
RESPONSE	None	
MODE	CA	
BUS STATES	ATN•UNL, MTA, LAG,GTL	
EXAMPLE	PRINT#1,"LOCAL 12,16"	Send Go To Local to devices 12 and 16

LOCAL LOCKOUT or LOL

The `LOCAL LOCKOUT` command causes the Micro488/EX to issue a Local Lockout IEEE bus command. Bus devices that support this command are thereby inhibited from being controlled manually from their front panels.

SYNTAX	`LOCAL LOCKOUT`
	or
	`LOL`	

RESPONSE	None	
MODE	CA	
BUS STATES	ATN•LLO	
EXAMPLES	`PRINT#1,"LOCAL LOCKOUT"`	Send Local Lockout command.
	PRINT#1,"LOL"	Same as above.

LOG

The LOG command enables macro command responses to be placed into the LOG Buffer rather than being sent to the serial output buffer. Only those responses of commands issued from within macros are placed into the LOG buffer. Commands issued to the serial input are not LOGed.

The data in the log buffer is maintained in non-volatile storage until it is deleted with either of the ERASE LOG or FACTORY commands. The amount of characters in the log buffer can be requested with the LOG MEMORY command and the data can be requested with a READ LOG command.

When a STARTUP Macro is executed, LOG is automatically set to ON.

SYNTAX LOG **{** OFF **|** ON **}**

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"MACRO 1"          Define a macro
PRINT#1,"STATUS"
PRINT#1,"ENDM"
PRINT#1,"DOMACRO 1"        Execute the macro
CONTROLLER 10              Macro response
PRINT#1,"LOG ON"               Enable logging
PRINT#1,"DOMACRO 1"        Execute the macro
                           This time, the response is
logged
PRINT#1,"LOG OFF"          Disable logging
PRINT#1,"READ LOG"         Request Logged data
CONTROLLER 10              Logged Macro response
PRINT#1,"ERASE LOG"        Delete Logged data
```

LOG MEMORY

The LOG MEMORY command is used to inquire the number of characters contained within the LOG Buffer. After an ERASE LOG or FACTORY command, the LOG Buffer will contain zero [0] characters.

SYNTAX LOG MEMORY

RESPONSE Number of characters contained within the Log Buffer.

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"MACRO 1"          Define a macro
PRINT#1,"HELLO"
PRINT#1,"ENDM"
PRINT#1,"LOG ON"              Enable logging
PRINT#1,"DOMACRO 1"        Execute the macro
                           The response is logged
PRINT#1,"LOG MEMORY"
INPUT #1,L                 Read the memory used
PRINT L                    Printed to the screen
54                         Number of bytes in buffer
PRINT#1,"ERASE LOG"        Erase the log buffer
PRINT#1,"LOG MEMORY"
INPUT #1,L                 Read the memory used
PRINT L                    Printed to the screen
0                          0 bytes in buffer
```

MACRO…ENDM

The MACRO command allows the user to build a file of sequential commands and execute them with a single DOMACRO command. The Micro488/EX is capable of storing up to 100 different Macros which are maintained even when the power to the interface is off.

Defining a Macro is initiated by issuing the MACRO command. Each subsequent character following this command, including terminators and intervening spaces, is saved in a buffer up to, and including, the ENDM. After the ENDM, the Micro488/EX appends the macro number to the Macro buffer as a two digit decimal number. The macro can then be executed by issuing a DOMACRO command. Any syntax errors that are included within the Macro are not checked until the Macro is executed.

If a Macro has not been defined, it does not consume any memory from the USER heap. If a Macro has been defined, any Macro buffer previously allocated is returned to the USER heap prior to requesting memory to re-define the Macro. The initial size of an allocated Macro buffer is 127 character locations (bytes). If more than 127 bytes are required to store the Macro, additional memory is allocated in 127 byte increments. If there is no available memory in the USER heap, an 'OUT OF MEMORY' error occurs and any memory allocated to that Macro is returned to the heap.

Other useful commands with Macros follow. You should refer to the command description for complete information.

COMMENT	The COMMENT command allows the user to send a pre-defined ASCII string to the serial host when the Macro executes.
COUNT	The COUNT command returns the decrementing Macro loop count of the last executed macro. This command is only valid during macro execution.
DOMACRO	The DOMACRO command executes a Macro. An optionally specified loop count and interval can be included with this command to execute the Macro multiple times at regular intervals. This loop count is the value returned with the COUNT command.

ERASE — The ERASE command is used to delete a Macro and return the occupied memory to the USER heap. A specific Macro can be deleted or all can be ERASEd.

LOG — Data which is generated by a macro can be diverted from the serial output into a LOG buffer with the LOG ON command. This LOG buffer is also non-volatile and can be read by the serial host with the READ LOG command. This feature is disabled by the LOG OFF command.

ON...DOMACRO — The ON <event> DOMACRO command enables macro execution upon detection of an event, such as SRQ being asserted by a bus device. This feature is mutually exclusive with the ARM command.

READ — The READ command allows the Macro buffer to be transmitted back to the serial host. It is sent exactly as it was built with the exception of the appended Macro number to the ENDM portion. This command requires making a copy of the specified Macro buffer.

TRACE — The TRACE ON command allows the embedded macro commands within the macro buffer to be echoed out the serial port to the host computer as the Macro is executed. This allows trace debugging during Macro execution. This feature is disabled with the TRACE OFF command.

SYNTAX

```
MACRO[;][number]
[command list…]
ENDM
```

or

```
MA[;][number]
[command list…]
ENDM
```

number is a Macro Buffer number from 0 to 99. If no number is specified, Macro 0 is assumed.

RESPONSE Dependent on the included command list

MODE Any

BUS STATES Defined by the included command list

EXAMPLE

```
10 PRINT #1,"MACRO 10"          Build Macro #10
20 PRINT #1,"OUTPUT16;Q00B1X"
30 PRINT #1,"COMMENT 'Memory Left =\"
40 PRINT #1,"MEMORY"
50 PRINT #1,"TRIGGER16/10"
60 PRINT #1,"ENTER16"
70 PRINT #1,"ENTER10"
80 PRINT #1,"ENDM"              End Macro Mode

90 PRINT #1,"DOMACRO 10"        Execute the Macro
100 INPUT #1,M$:PRINT M$        Read Comment &
                                Memory
110 INPUT #1,R$:PRINT R$        Read data from
                                'ENTER16'
120 INPUT #1,R$:PRINT R$        Read data from
                                'ENTER10'

130 PRINT #1,"DOMACRO 10;8      Execute it 8 more times
140 FOR N = 1 TO 8              Read data 8 times
150 INPUT #1,M$:PRINT M$        Read Comment &
                                Memory
160 INPUT #1,R$:PRINT R$        Read data from
                                'ENTER16'
170 INPUT #1,R$:PRINT R$        Read data from
                                'ENTER10'
180 NEXT N

190 PRINT #1,"READ 10"          Request a copy of Macro
200 WHILE NOT EOF(1)            Loop until entire Macro
210  MA$=MA$+INPUT$(1,#1)       Buffer has been received
220 WEND
230 PRINT MA$
```

MASK

The MASK command is used to mask the high bit (MSB) of serial input data. Some serial host computers set the most significant bit when using eight bit serial data lengths. When the MASK ON command is issued, each serial character received is logically ANDed with &H7F (127 decimal).

MASK OFF is the power-on default. In default operation, all serial input data is automatically masked with &H7F. The exception to this is any data which follows a semi-colon (;), an apostrophe (') or a quotation mark ("). After a MASK ON command, all characters are masked.

SYNTAX MASK **{**ON | OFF**}**

RESPONSE None

MODE Any

BUS STATES None

```
EXAMPLE     PRINT#1,"MASK ON"
```

MEMORY

The MEMORY command returns the amount of memory which, at the time the command is executed, is available in the USER heap.

SYNTAX MEMORY
or
ME

RESPONSE numeric value of the remaining memory in the USER heap

MODE Any

BUS STATES None

EXAMPLE
```
PRINT#1,"MEMORY"        Request amount of available memory

INPUT#1,M : PRINT M
```

ON <event> DOMACRO

The ON <event> DOMACRO command allows the Micro488/EX to automatically execute a Macro when one or more of the specified events occur. The events are polled between commands and when one of the events is detected as true, its assigned Macro is executed. Once executed, the event is disabled from further execution and must be re-enabled with another ON <event> DOMACRO command.

There are two types of events, level sensitive and edge sensitive. Level sensitive events, such as SRQ, will cause Macro execution if they are enabled while the event condition persists. Usually, some action must be taken (eg SPOLL) to clear the condition prior to re-issuing the ON <event> DOMACRO command. Edge sensitive events, such as TRIGGER, are cleared when the Macro executes.

Regardless of the event sensitivity, the ON <event> DOMACRO command must be re-sent after the Macro executes to re-activate the event condition. The optional events include…

SRQ — This event is level sensitive. If the condition exists at the time the ON SRQ DOMACRO command is issued, the Micro488/EX will execute the assigned Macro immediately.

PERIPHERAL — This event occurs when the Micro488/EX is force from the Controller Active State (*SC•CA) to the Peripheral State (*SC•*CA) by receipt of IFC from the System Controller. This can be useful in detecting receipt of IFC when in the *SC•CA state. This event is edge sensitive.

CONTROLLER — This event occurs when the Micro488/EX receives control of the bus and transitions from the Peripheral State (*SC•*CA) to the Controller Active State (*SC•CA). This occurs when the Take Control interface message is received by the Micro488/EX. This event is edge sensitive.

TRIGGER — This event occurs when the Micro488/EX, as a Peripheral (*CA), receives a Group Execute Trigger (GET) command from the Active Controller. This event is edge sensitive. When the Macro is executed, the internal status, as read by the STATUS 1 command, is cleared.

CLEAR — This event occurs when the Micro488/EX, as a Peripheral (*CA), receives a Device Clear (DCL) or a Selected Device Clear (SDC) command from the Active Controller. This event is edge sensitive. When the Macro is executed, the internal status, as read by the STATUS 1 command, is cleared.

TALK — This event occurs when the Micro488/EX, as a Peripheral (*CA), detects its My Talk Address (MTA) command from the Active Controller. It indicates that the controller has requested information from the Micro488/EX. This event is edge sensitive.

LISTEN — This event occurs when the Micro488/EX, as a Peripheral (*CA), detects its My Listen Address (MLA) command from the Active Controller. It indicates that the controller has information it wants to send to the Micro488/EX. This event is edge sensitive.

IDLE — This event occurs when the Micro488/EX, as a Peripheral (*CA), transitions from a Talker or Listener state to an idle state (neither talker or listener). It indicates that the controller has unaddressed the Micro488/EX with either an UNT or UNL command. This event is edge sensitive. When the Macro is executed, the internal status, as read by the STATUS 1 command, is cleared. This event is tested prior to the CHANGE event and will clear the internal CHANGE status only when the addressed to un-addressed transition occurs.

CHANGE — This event occurs when the Micro488/EX, as a Peripheral (*CA), detects an addressed state change. This occurs on transitions from a Talker or Listener state to an idle state (neither talker or listener), or from an idle state to a talker or listener state. This event is edge sensitive. When the Macro is executed, the internal status, as read by the STATUS 1 command, is cleared. This event is tested after the IDLE event. If IDLE is enabled, the CHANGE event's Macro will only execute when an un-addressed to an addressed transition occurs.

ERROR — This event occurs when the Micro488/EX detects an error condition. The error condition may be an un-recognized command from the serial host, an invalid parameter or a bus error. Refer to Appendix B for a listing of the error conditions which can be detected by the Micro488/EX. This event is level sensitive. When the Macro executes, the error status, as read by any of the STATUS commands, must be read prior to enabling this event again.

STARTUP — This edge sensitive event occurs when the Micro488/EX is powered on, upon the receipt of an ID character, double ID characters or RESET command. If a macro has been asssigned as a STARTUP macro, it sets LOG ON and executes. This is the only event which cannot be re-enabled within a Macro. Although STARTUP is a valid condition within the ARM command, it does not ARM.

Once a condition is enabled it remains enabled until it is DISARMed, the event specified has occurred or until the Micro488/EX is reset.

The ON <event> DOMACRO and ARM commands are mutually exclusive. The last command issued takes precedence.

SYNTAX

ON **{**<event>**}** DOMACRO **[**number**]**
or
ON **{**<event>**}** DO **[**number**]**

where <event> must include one of the following options…

Event	Abbr Form
SRQ	SRQ
PERIPHERAL	PE
CONTROLLER	CO
TRIGGER	TR
CLEAR	CL
TALK	T
LISTEN	L
IDLE	I
CHANGE	CH
ERROR	ER
STARTUP	ST

RESPONSE None

MODE any

BUS STATES None

EXAMPLES

```
10 PRINT#1,"MACRO 0"     Create Macro #0
20 PRINT#1,"COMMENT 'I have received an SRQ'"
30 PRINT#1,"ENDM"
40 PRINT#1,"ON SRQ DOMACRO 0"
                         Enable Macro 0 for On SRQ
```

As a peripheral, the `ON LISTEN DOMACRO` can be used to input data from the active controller.

```
10 PRINT#1,"MACRO 20"    Create Macro # 20
20 PRINT#1,"ENTER"
30 PRINT#1,"ON LISTEN DOMACRO 20"
                         Re-enable for the next MLA
40 PRINT#1,"ENDM"

50 PRINT#1,"ON LISTEN DOMACRO 20"
                         Enable it for Listen
```

The data will be transmitted out to the serial host without having to re-issue an `ENTER` command from the host each time.

The next example is an abbreviated form of the previous. Although not as clear, it would require less USER heap memory if there were more being done in the Macro.

```
10 PRINT#1,"MA20"        Create Macro # 20
20 PRINT#1,"EN"
30 PRINT#1,"ON L DO20"   Re-enable for the next MLA
40 PRINT#1,"ENDM"
50 PRINT#1,"ON L DO20"   Enable it for Listen
```

The following example program defines two macros. The STARTUP event is used to invoke Macro 80. After running this program, the power is removed from the Micro488/EX. When power is re-applied, Macro 80 executes. It waits until 1:00 PM then executes Macro 70, 10 times and 20 second intervals. The STARTUP event sets LOG ON. The data that is collected by Macro 70 is then placed in the LOG buffer. The data separator between the TIME, DATE and Device 16 data was set to a horizontal TAB character by the COMMENT command.

After execution, the data can be read from the LOG buffer with the READ LOG command.

```
10  PRINT #1,"MACRO80"                  Define Macro 80
20  PRINT #1,"REMOTE16"                 An external indication that the
                                        Macro is executing
30  PRINT #1,"WAIT 1:00PM"              Wait Until 1:00 PM
40  PRINT #1,"DOMACRO 70,10,20"         Execute Macro 70, 10 times at
                                        20 second intervals
50  PRINT #1,"ENDM"

60  PRINT #1,"MACRO 70"                 Define Macro 70
70  PRINT #1,"STERM NONE"               Disable Serial terminators
80  PRINT #1,"TIME"                     Request the Time
90  PRINT #1,"COMMENT'";CHR$(9);"'" A Horizontal TAB comment
100 PRINT #1,"DATE"                     Request the Date
110 PRINT #1,"COMMENT'";CHR$(9);"'" A Horizontal TAB comment
120 PRINT #1,"STERM CR"                 Serial Terminator to CR
130 PRINT #1,"ENTER 16"                 A reading from device 16
140 PRINT #1,"ENDM"

150 PRINT #1,"ON STARTUP DO 80"         Set Macro 80 as the startup
```

OUTPUT (Controller mode)

The OUTPUT command sends data to the IEEE bus. The Remote Enable line is first asserted if the Micro488/EX is the System Controller. If device addresses are specified, those devices will then be addressed to listen. If addresses are not specified, the Micro488/EX must already be configured to send data, either as a result of an immediately preceding OUTPUT command or as the result of a SEND command.

If the character count, count, is specified then exactly that number of characters will be sent to the bus devices. Otherwise, OUTPUT terminates data transfer upon detection of a serial CR of LF terminator from the serial input. The serial input terminator(s) are replaced with the bus output terminator(s) before being sent to the bus devices.

The number of characters that can be set to a bus device is limited by the available USER heap. The exception to this is OUTPUT #count, in which the number of bytes is limited to 65,535.

The IEEE bus output terminators can be modified with the TERM command. Refer to this command description for complete information.

SYNTAX `OUTPUT[addr[,addr…]][#count];data`
or
`OU[addr[,addr…]][#count];data`

`addr` is a bus device address. Up to 15 addresses may be specified.
`count` is the number of characters to OUTPUT.
`data` is a string of characters to OUTPUT terminated by the serial terminator(s). (unless `count` is specified in which case no terminator is needed).

RESPONSE None

MODE CA

BUS STATES REN (if SC), *ATN, data (without `addr`)
REN (if SC), ATN•MTA, UNL, LAG, *ATN, data (with `addr`)

EXAMPLES

```
PRINT#1,"OUTPUT 22;R0C0T1X"
```
Send "R0C0T1X" to device 22.

```
PRINT#1,"OUTPUT 06,12;ABC"
```
Send "ABC" to devices 6 and 12.

```
PRINT#1,"OUTPUT;XYZ"
```
And send them "XYZ".

```
PRINT#1,"OUTPUT 0602;DEF"
```
Send "DEF" to device 6, sec addr 2.

```
PRINT#1,"OUTPUT06#26;abcdefghijklmnopqrstuvwxyz"
```
Send the 26 letters of the alphabet without terminators to device 6.

OUTPUT (Peripheral mode)

In Peripheral mode the OUTPUT command sends data to the IEEE bus under control of the Active Controller. The Active Controller must put the Micro488/EX into the Talk state and configure some bus device to accept the transferred data. The Talk state can be checked with the STATUS 1 command, can cause a reported condition via the ARM command or force execution of a Macro with the ON TALK DOMACRO command. A time-out error will occur, if enabled, if no bus device accepts the data within the time out period after issuing the OUTPUT command.

If the character count, count, is specified then exactly that number of characters will be sent to the bus devices. Otherwise, OUTPUT terminates data transfer upon detection of the serial CR of LF terminator(s) from the serial input. The serial terminator(s) are replaced with the bus output terminator(s) before being sent to the bus devices.

The number of characters that can be set to a bus device is limited by the available USER heap. The exception to this is OUTPUT #count, in which the number of bytes is limited to 65,535.

The IEEE bus output terminators can be modified with the TERM command. Refer to this command description for complete information.

Even as a Peripheral, the Micro488/EX might be the System Controller. If it is, then it will assert Remote Enable before sending any data.

SYNTAX `OUTPUT [#count];data`
or
`OU [#count];data`

`count` is the number of characters to OUTPUT.
`data` is a string of characters to OUTPUT terminated by the serial output terminator(s) unless `count` is specified.

RESPONSE None

MODE *CA

BUS STATES Determined by the Controller, REN asserted if SC

EXAMPLES
```
PRINT#1,"OUTPUT;DC VOLTS"
```
Send "DC VOLTS".

```
PRINT#1,"OUTPUT#5;ABCDE"
```
Send "ABCDE" without any bus terminators.

PASS CONTROL

The PASS CONTROL command allows the Micro488/EX to give control to another controller on the bus. After passing control, the Micro488/EX enters the Peripheral mode (*CA). If the Micro488/EX was the System Controller, then it remains the System Controller but it is no longer the Active Controller. The Controller now has command of the bus until it passes control to another device or back to the Micro488/EX. The System Controller can regain control of the bus at any time by issuing an ABORT command.

SYNTAX
```
PASS CONTROL addr
```
or
```
PA addr
```

addr is the bus address of the device to which control is passed.

RESPONSE None

MODE CA

BUS STATES ATN•UNL, MLA, TAG, UNL,TCT, *ATN

EXAMPLES
```
100 PRINT#1,"PASS CONTROL 22"
                              Control is passed to device 22.
110 PRINT#1,"STATUS 1"        Wait until we are controller
                              again.
120 INPUT#1,A$                Use STATUS 1 to check
130 IF LEFT$(A$,1)<>"C" THEN 110
140 <rest of program>
```

The next example uses the ARM command to determine when control has been given back to the Micro488/EX.

```
100 PRINT#1,"PA 22"           Control is passed to device 22.
110 PRINT#1,"ARM CO"
120 INPUT#1,A$                Wait until we are controller
130 <rest of program>
```

PPOLL

The Parallel Poll command, PPOLL, is used to request status information from many bus devices simultaneously. If a device requires service then it will respond to a Parallel Poll by asserting one of the eight IEEE bus data lines (DIO1 through DIO8, with DIO1 being the least significant). In this manner, up to eight devices may simultaneously be polled by the controller. More than one device can share any particular DIO line. In this case it is necessary to perform further Serial Polling to determine which device actually requires service.

Parallel polling is often used upon detection of a Service Request (SRQ), though it may also be performed periodically by the controller. In either case, PPOLL will respond with a number from 0 to 255 corresponding to the eight binary DIO lines.

Not every device supports parallel polling. Refer to the manufacturer's documentation for each bus device to determine if Parallel Poll capabilities are supported.

SYNTAX PPOLL

RESPONSE Number in the range of 0 to 255

MODE CA

BUS STATES ATN•EOI,<parallel poll response>, *EOI

EXAMPLE

```
PRINT#1"PPOLL"          Conduct a Parallel Poll
INPUT#1,PPSTAT          Receive the PPOLL status
PRINT PPSTAT
```

PPOLL CONFIG or PPC

`PPOLL CONFIG` (Parallel Poll Configure) configures the Parallel Poll response of a specified bus device. Not all devices support Parallel Polling and, among those that do, not all support software control of their Parallel Poll response. Some devices are configured by internal switches.

The Parallel Poll response is set by a four-bit binary number (S P2 P1 P0), `response`. The most significant bit of `response` is the Sense (S) bit. The Sense bit is used to determine when the device will assert its Parallel Poll response. Each bus device has an internal individual status (*ist*). The Parallel Poll response will be asserted when this *ist* equals the Sense bit value. *ist* is normally a logic "1" when the device requires attention, so the S bit should normally also be a logic "1". If the S bit is "0" then the device will assert its Parallel Poll response when its *ist* is a logic "0", i.e. it does not require attention. However, the meaning of *ist* can vary between devices, so refer to your IEEE bus device documentation.

The remaining 3 least significant bits of `response`, P2, P1, and P0, specify which DIO bus data line will be asserted by the device in response to a Parallel Poll. These bits form a binary number with a value from 0 through 7, specifying data lines DIO1 through DIO8, respectively.

SYNTAX

```
PPOLL CONFIG addr;response
or
PPOLL C addr;response
or
PPC addr;response
```

`addr` is a bus address.
`response` is the decimal equivalent of the four binary bits S, P2, P1, and P0.

RESPONSE None

MODE CA

BUS STATES ATN•UNL, MTA, LAG, PPC, PPE

EXAMPLES
```
PRINT #1,"PPC23;&H0D"
```
Configure device 23 to assert DIO6 when it desires service and it is Parallel Polled (&H0D = 1101 binary; S = 1, P2P1P0 = 101 = 5 decimal = DIO6).

PPOLL DISABLE or PPD

PPOLL DISABLE disables the Parallel Poll response of selected bus devices.

SYNTAX
```
PPOLL DISABLE addr[,addr…]
or
PPOLL D addr[,addr…]
or
PPD addr[,addr…]
```

addr is a bus device address

RESPONSE None

MODE CA

BUS STATES ATN•UNL, MTA, LAG, PPC, PPD

EXAMPLE
```
PRINT#1,"PPOLL DISABLE18,06,13"
```
Disable Parallel Poll response of devices 18, 6, and 13.

PPOLL UNCONFIG or PPU

PPOLL UNCONFIG (Parallel Poll Unconfigure) disables the Parallel Poll response of all bus devices.

SYNTAX PPOLL UNCONFIG
or
PPOLL U
or
PPU

RESPONSE None

MODE CA

BUS STATES ATN•PPU

EXAMPLE
```
PRINT #1,"PPOLL UNCONFIG
```
Disable the Parallel Poll response of all bus devices.

READ

The READ command is used to inspect the contents of a defined Macro Buffer. Macro Buffers can be individually specified by their number. When a READ command is received, a copy of the Macro buffer requested is sent to the serial output.

SYNTAX `READ[;][number]`

`number` is a Macro Buffer number from `0` to `99`. If not specified, Macro 0 is assumed.

RESPONSE The contents of the Macro buffer are returned to the serial output.

MODE Any

BUS STATES None

EXAMPLES

`PRINT#1,"READ"`	Read the contents of Macro 0
`PRINT#1,"READ 20"`	Read the contents of Macro 20

READ LOG

The READ LOG command is used to request a copy of the data contained within the LOG Buffer to be sent out the serial port. The logged data is not deleted when read. To delete the LOG Buffer, the ERASE LOG or FACTORY commands are used.

The number of characters contained within the LOG Buffer can be determined with the LOG MEMORY command. READing the LOG buffer during macro execution will generate a 'LOGGING ERROR'

SYNTAX READ LOG

RESPONSE Data contained within the LOG Buffer.

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"MACRO 1"          Define a macro
PRINT#1,"HELLO"
PRINT#1,"ENDM"
PRINT#1,"LOG ON"           Enable logging
PRINT#1,"DOMACRO 1"        Execute the macro
                           The response is logged
PRINT#1,"LOG MEMORY"
INPUT #1,L                 Read the memory used
PRINT L                    Printed to the screen
54                         Number of bytes in buffer
PRINT#1,"READ LOG"         Read the contents
INPUT #1,L$                Read the data
PRINT L$                   Printed to the screen
Micro488/EX Revision N.N Copyright (C) 1988 IOtech
PRINT#1,"ERASE LOG"        Erase the log buffer
PRINT#1,"LOG MEMORY"
INPUT #1,L                 Read the memory used
PRINT L                    Printed to the screen
0                          0 bytes in buffer
```

REMOTE

The REMOTE command asserts the Remote Enable (REN) bus management line. If the optional bus addresses are specified, then REMOTE also addresses those devices to listen, placing them in the Remote addressed state.

SYNTAX REMOTE
or
REM

RESPONSE None

MODE SC

BUS STATES REN

EXAMPLES
```
PRINT #1,"REMOTE"
```
Assert Remote Enable

SYNTAX REMOTE addr[,addr…]
or
REM addr[,addr…]

addr is a bus device address

RESPONSE None

MODE SC•CA

BUS STATES REN, ATN•UNL, MTA, LAG

```
PRINT #1,"REMOTE16,28"
```
Assert Remote Enable and address devices 16 and 28 to listen.

REQUEST

In Peripheral mode, the Micro488/EX is able to request service from the Active Controller by asserting the Service Request bus signal. The REQUEST command sets the Serial Poll status (including Service Request) of the Micro488/EX. REQUEST takes a numeric argument in the range of 0 to 255 (&H0 to &HFF) that is used to set the Serial Poll status. When the Micro488/EX is Serial Polled by the Controller, it returns this byte on the DIO data lines.

The data lines are numbered DIO8 through DIO1. DIO8 is the most significant line and corresponds to a value of 128 (&H80). DIO7 is the next most significant line and corresponds to a value of 64 (&H40). DIO7 has a special meaning: It is the Request For Service (*rsv*) bit. REQUEST always forces this bit to a '1' which generates a service request (SRQ) to the controller.

When the Micro488/EX is Serial Polled, all eight bits of the Serial Poll status are returned to the Controller. The *rsv* bit is cleared when the Micro488/EX is Serial Polled by the Controller. This causes the Micro488/EX to stop asserting SRQ. The Micro488/EX will also un-assert *rsv* when it receives a Device Clear (DCL) or Selected Device Clear (SDC) from the active controller.

SYNTAX `REQUEST[;][status]`
or
`REQ[;][status]`

`status` is the service request status in the range of 0 to 255. If `status` is not specified, only *rsv* (DIO7) is asserted.

RESPONSE None

MODE *CA

BUS STATES SRQ

EXAMPLES
```
PRINT#1,"REQUEST";2+4
```
Generate an SRQ (64) with DIO2 (2) and DIO3 (4) set in the Serial Poll Response.
```
PRINT#1,"REQUEST"
```
Generate an SRQ (64)

RESET

The system command RESET provides a warm start of the interface. Issuing the RESET re-initializes the internal IEEE controller hardware, installs the power-on terminators and the default DAY, DATE, TIME and CASE formats.

If a STARTUP macro has been defined, it will execute. It is also equivalent to issuing the following commands…

ABORT (If System Controller)
DISARM
ERROR OFF
LOCAL (If System Controller)
REQUEST 0 (With *rsv* cleared if peripheral)
TIME OUT 0
TRACE OFF

The RESET command provides a warm start of the interface as well as clearing all error conditions. Upon detection of the RESET command, the Micro488/EX un-asserts its serial output handshake line (RTS). It re-asserts it when it is capable of accepting serial input data. If XON/XOFF handshake is selected, the handshake state is not effected by the RESET command.

SYNTAX	RESET or RESE
RESPONSE	None
MODE	Any
BUS STATES	IFC,*IFC,*REN (if SC)
EXAMPLE	PRINT#1,"RESET"

RESUME

The RESUME command un-asserts the Attention (ATN) bus signal. As the Active Controller, Attention is normally kept asserted by the Micro488/EX but it must be un-asserted to allow transfers to take place between two Peripheral devices. In this case, the Micro488/EX SENDs the appropriate talk and listen addresses, and the must un-assert Attention with the RESUME command.

SYNTAX	RESUME or RESU	
RESPONSE	None	
MODE	CA	
BUS STATES	*ATN	
EXAMPLE	PRINT#1,"RESUME"	Un-assert ATTENTION line.

SAVE

The SAVE command is used to clear NVRAM ERROR conditions after a FACTORY command and to SAVE user specified DATE FORMAT, DAY FORMAT, TIME FORMAT and CASE defaults to NVRAM.

SYNTAX SAVE

RESPONSE None

MODE Any

BUS STATES None

EXAMPLE

```
PRINT#1,"FACTORY"
FOR N = 1 to 10000 : NEXT N
PRINT#1,"DATE FORMAT MM-DD-YY"
PRINT#1,"TIME FORMAT HH:MM:SS"
PRINT#1,"DAY FORMAT DAY"
PRINT#1,"CASE LOWER"
PRINT#1,"SAVE"
```

SEND

The SEND command provides byte-by-byte control of data and control transfers on the bus and gives greater flexibility than the other commands. The command can specify exactly which operations will be executed by the Micro488/EX.

The following sub-commands are available within the SEND command:

UNT Send the multiline Untalk command. ATN is asserted.

UNL Send the multiline Unlisten command. ATN is asserted.

MTA Send My (the Macro488's) Talk Address. ATN is asserted.

MLA Send My (the Macro488's) Listen Address. ATN is asserted.

TALK addr Send Talk Address addr device (TAG). ATN is asserted.

LISTEN addr[,addr…]
Send Listen Addresses (LAG). ATN is asserted.

DATA {'data'|char[,char…]}
Send character strings data or characters with numeric ASCII values char with ATN unasserted.

EOI {'data'|char[,char…]}
Send character strings data or characters with numeric ASCII values char with ATN unasserted. EOI is asserted on the last character.

CMD {'data'|char[,char…]}
Send character strings data or characters with numeric ASCII values char with ATN asserted.

ENTER Request data from a device terminating on LF. ATN is unasserted.

The DATA, EOI and, CMD sub-commands send data bytes or characters over the bus. The characters to be sent are specified either as a quoted string ('data') or as individual ASCII values (char[,char…]). For example, DATA'R0X' sends the characters R, 0, and X to the active listeners, and DATA 13,&H0A sends carriage-return and line-feed. Multiple ASCII char bytes may be specified by separating them with commas.

The EOI sub-command is identical to the DATA sub-command except that the End Or Identify (EOI) signal is asserted on the transfer of the last character.

The CMD sub-command sends the data bytes with Attention (ATN) is asserted. This tells the bus devices that the characters are to be interpreted as IEEE bus commands, rather than as data. EOI is not asserted during CMD transfers. For example CMD &H3F is the same as Unlisten (UNL). Note that it is not possible to assert EOI during the transfer of a command byte because EOI and ATN together specify parallel poll.

The ENTER sub-command inputs data from the active talker, setup from either a TALK addr sub-command or a previous ENTER command. Addresses are not allowed to be specified as options to this sub-command. The ENTER sub-command will terminate upon detection of a Line Feed (LF) character.

Note that the maximum length of the SEND command, including any sub-commands, is 127 characters. If large amounts of data must be transferred using the SEND command, then multiple SEND commands must be used so that they are each less than 127 characters long. For example…

```
PRINT#1,"SEND UNT UNL MTA LISTEN 16 DATA 1,2,3,4,5,6"
```

is equivalent to…

```
PRINT#1,"SEND UNT UNL MTA LISTEN 16"
PRINT#1,"SEND DATA 1,2,3"
PRINT#1,"SEND DATA 4,5,6"
```

In this way, a long SEND command can be broken up into shorter commands.

SYNTAX `SEND [;] sub-command[sub-command...]`
or
`SE [;] sub-command[sub-command...]`

RESPONSE None or device data

MODE CA (any sub-commands)
Any (`DATA`, `EOI` and `ENTER` sub-commands only)

BUS STATES user defined

EXAMPLES
```
PRINT#1,"SEND MTA UNL LISTEN 16 DATA 'T1S0R2X'"
```
is the same as
```
PRINT#1,"OUTPUT16;T1S0R2X"
```

```
PRINT#1,"SEND CMD128,0,10 DATA156,35 EOI'ABC'"
```

sends the following byte sequence:

Data	ATN	EOI
10000000	ATN	*EOI
00000000	ATN	*EOI
00001010	ATN	*EOI
10011100	*ATN	*EOI
00100011	*ATN	*EOI
01000001	*ATN	*EOI
01000010	*ATN	*EOI
01000011	*ATN	EOI

SET DATE

The SET DATE command is used to set the internal date. Any of the output formats may be used to set the date. However, the Micro488/EX is sensitive to which American or European format are selected. If a European format has been selected, the SET DATE command requires the date setting information to be in the same order.

SYNTAX SET DATE **[**;**]** **{**date**}**

American date	Example
MM/DD/YY	12/04/76
MM-DD-YY	12-04-76
MM/DD/YYYY	12/04/1976
MM-DD-YYYY	12-04-1976
MON DD YYYY	DEC 04 1976
MON DD, YYYY	DEC 04, 1976
MONTH DD YYYY	DECEMBER 4 1976
MONTH DD, YYYY	DECEMBER 4, 1976

European date	Example
DD/MM/YY	04/12/76
DD-MM-YY	04-12-76
DD/MM/YYYY	04/12/1976
DD-MM-YYYY	04-12-1976
DD MON YYYY	04 DEC 1976
DD MONTH YYYY	4 DECEMBER 1976

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"DATE FORMAT MM-DD-YY"
PRINT#1,"SET DATE 11-04-52"    Set to American format
PRINT#1,"DATE"
INPUT #1,D$                    Read the date string
PRINT D$                       Printed to the screen
11-04-52                       Output is format dependent

PRINT#1,"SET DATE JANUARY 1, 1988"
PRINT#1,"DATE"
INPUT #1,D$                    Read the date string
PRINT D$                       Printed to the screen
01-01-88                       Output is format dependent
```

SET DAY

The SET DAY command is used to set the internal day of week. Any of the DAY command output formats may be used, either numeric, abbreviated or unabbreviated.

SYNTAX SET DAY **[**;**]** **{**d | da | day**}**

d is a numeric representation from 1 (sunday) to 7 (saturday).
da is an abbreviated representation (SUN, MON, TUE, WED, THU, FRI, SAT)
day is an un-abbreviated representation (SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY)

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"SET DAY 3"
PRINT#1,"DAY"
INPUT #1,D$      Read the day string
PRINT D$         Printed to the screen
TUESDAY          Output is format dependent

PRINT#1,"SET DAY WED"
PRINT#1,"DAY"
INPUT #1,D$      Read the day string
PRINT D$         Printed to the screen
WEDNESDAY        Output is format dependent

PRINT#1,"SET DAY FRIDAY"
PRINT#1,"DAY"
INPUT #1,D$      Read the day string
PRINT D$         Printed to the screen
FRIDAY           Output is format dependent
```

SET TIME

The `SET TIME` command is used to set the internal clock. 24 hour format is implied if the AM or PM specifier is not included. The time specified must include hours (`hh`) and minutes (`mm`). Seconds (`ss`), if omitted, are set to `00`.

Both 12 and 24 hour formats may be used to set the time. If AM or PM is not specified, 24 hour format is assumed. The relationship between the 12 and 24 hour operations are…

	24 Hour	12 Hour
Midnight	00:00:00	12:00:00 AM
	01:00:00	01:00:00 AM
	…	…
	11:59:59	11:59:59 AM
Noon (Midday)	12:00:00	12:00:00 PM
	13:00:00	01:00:00 PM
	…	…
	23:59:59	11:59:59 PM

SYNTAX `SET TIME [;] hh:mm [:ss ] [ AM | PM ]`

`hh` specifies hours in 24 hour format unless `AM` or `PM` is included
`mm` specifies minutes
`ss` specifies seconds, set to `00` if omitted

RESPONSE None

MODE Any

BUS STATES None

EXAMPLE
```
PRINT#1,"SET TIME 2:22 AM"
PRINT#1,"TIME"
INPUT #1,T$       Read the time string
PRINT T$          Printed to the screen
02:22:00 AM       Output is time and format dependent
```

SPOLL

In Active Controller mode the SPOLL command performs a Serial Poll of the bus device specified and responds with number from 0 to 255 representing the decimal equivalent of the eight-bit device response. If *rsv* (DIO7, decimal value 64) is set, then that device is signaling that it requires service. The meanings of the other bits are device-specific. Serial Polls are normally performed in response to assertion of the Service Request (SRQ) bus signal by some bus device. If multiple addresses are specified, the Micro488/EX will serial poll each device in sequence and output each device's response to the serial port with the serial output terminator(s) appended.

In Active Controller mode, with no bus address specified, the SPOLL command returns the external SRQ status. If the SRQ line is asserted, the Micro488/EX will return a "64". If it is not asserted, the Micro488/EX will return a "0".

In Peripheral mode the SPOLL command is issued without an address and returns Macro488's own serial poll status. If *rsv* (DIO7, decimal value 64) is set, then the Micro488/EX has not been Serial Polled since the issuing last REQUEST command. *rsv* is reset whenever the Micro488/EX is Serial Polled by the Controller.

SYNTAX `SPOLL [addr[,addr…]]`
or
`SP [addr[,addr…]]`

addr is the bus device(s) to be Serial Polled

RESPONSE 0 or 64 (without addr)
0 to 255 (with addr)

MODE CA

BUS STATES ATN•UNL, MLA, TAG, SPE, *ATN, data, ATN•SPD, UNT

EXAMPLES

```
PRINT#1,"SPOLL 16"        Serial Poll  device 16
INPUT#1,SPSTAT            Receive the Spoll status
IF SPSTAT AND 64 THEN…    Test rsv…
```

```
PRINT#1,"SPOLL"          Check the SRQ status
INPUT#1,SRQ
IF SRQ<>0 THEN…          If SRQ is asserted then …

PRINT#1,"SPOLL 10,12,16"
INPUT#1,SP10,SP12,SP16   Get SPOLL response from devices
                         10, 12 and 16.
```

SYNTAX `SPOLL`
or
`SP`

RESPONSE 0 to 255

MODE *CA

BUS STATES None

EXAMPLES
```
PRINT#1,"SPOLL" Get own Spoll Status
INPUT#1,SPSTAT
IF (SPSTAT AND 64) = 0 THEN…
```
rsv will be reset if we have been Spolled.

STATUS

STATUS is a system command which is useful for determining which mode the Micro488/EX is in, its current address or the type of ERROR that has occurred. At power-up, issuing the STATUS command will cause the Micro488/EX to return the string...

```
CONTROLLER 10
```

After issuing a PASS CONTROL or PERIPHERAL command the Micro488/EX will respond with...

```
PERIPHERAL 10
```

If PASS CONTROL was issued and the Micro488/EX then receives control again it will respond with....

```
CONTROLLER 10
```

This is useful for checking when control is returned after a PASS CONTROL has been issued.

If an ERROR has occurred, as indicated on the front panel of the Micro488/EX, issuing the STATUS command will cause an error message to be returned to the host. Once the error message is sent by the Micro488/EX the error condition is cleared. Refer to Appendix B for error message explanations.

The Micro488/EX also includes and extended status command, STATUS 1. The STATUS 1 command returns various items detailing the current state of the Micro488/EX. They are returned as one long character string as follows:

Item	Starting Col.	# Cols.	Values
Operating mode	1	1	C: Controller P: Peripheral
Bus address	3	2	Two-digit decimal number 00 to 30

Address change	6	2	G0: Address status change has not occurred. G1: Address status change has occurred.
Addressed state	9	1	T: Talker L: Listener I: Idle
Service Request	11	2	S0: SRQ is not asserted. S1: SRQ is asserted.
Error code	14	3	Enn: Letter 'E' followed by two-digit error code. Refer to Appendix B for Error explanations.
Triggered	18	2	T0: No IEEE Trigger command received. T1: Received the IEEE Trigger command.
Cleared	21	2	C0: No IEEE Clear command received. C1: Received the IEEE Clear command.
Error description	24	17	Text of error message

The Operating Mode (C or P) indicates whether or not the Micro488/EX is the Active Controller. If the Micro488/EX Passes Control to another device, then the Operating Mode indicator will change from "C" to "P". When the Micro488/EX regains control, then the indicator will again be "C". If the Micro488/EX is not the System Controller, then it will initially be a Peripheral and thus the indicator will be "P". It will, of course, become "C" when the Micro488/EX receives control from the Active Controller.

The Bus Address is the IEEE bus device address assigned to the Micro488/EX by the internal hardware switch.

The Address Change (G0, G1) indicator is set whenever the Micro488/EX transitions from the idle state to a Talker or Listener, or from a Talker or Listener state to an idle state. It will not indicate when a change is made from a listener to a talker or a talker to a listener. The address change is reset when `STATUS 1` is read.

The Addressed State is the current talker/listener state of the Micro488/EX. As a Peripheral, the Micro488/EX can check this status to see if it has been Addressed to Talk or Addressed to Listen by the Active Controller. In this way the desired direction of data transfer can be determined.

The Service Request indicator reflects the external SRQ status. If the SRQ line is asserted, S1 will be reported. If it is un-asserted, S0 will be reported.

The Error Code is 00 when no error has occurred. If it is non-zero, then the appropriate error message is appended to the `STATUS 1` response. For more details about the individual errors, refer to Appendix B. The Error Code is reset to 00 when `STATUS` is read.

The Triggered (T0, T1) and Cleared (C0, C1) indicators are set when, as a Peripheral, the Micro488/EX has received a GET (Group Execute Trigger) or SDC/DCL (Selected Device Clear/Device Clear) bus command. These two indicators are cleared when `STATUS1` is read.

By issuing the `STATUS 2` command, only the numeric error value is returned to the serial output port. If no error has occurred, the value sent is `0`. The error condition is clear when reported.

SYNTAX `STATUS [;][number]`
or
`ST [;][number]`

`number` is `0` to `2`. If not specified, `0` is assumed.

RESPONSE Character string as described previously

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"STATUS"          Read the Micro488/EX status
INPUT#1,A$
PRINT A$                  and display it.
CONTROLLER 10             Example of displayed STATUS 0

PRINT#1,"STATUS1"         Read the Micro488/EX extended
```

```
INPUT#1,A$                  status
PRINT A$                    and display it.
C 10 G0 I S0 E00 T0 C0 OK   Example of STATUS 1

PRINT#1,"STATUS2"           Read the Micro488/EX error
INPUT#1,A                   status
PRINT A                     and display it.
0                           Example of displayed STATUS 2
```

STERM

The STERM command sets the end-of-line terminators for output to the serial host. All output to the serial port is terminated by the serial output terminator(s). All input from the serial host must be terminated by either a Line Feed (LF) or Carriage Return (CR) except OUTPUT #count.

During INPUT, the Micro488/EX takes the data it receives from the bus device until it detects the LF of other optionally specified input terminating condition. It strips all CR and LF from the input data and appends the serial output terminator(s) before sending it to the serial host. The default serial terminators for output are set by internal DIP switches and are factory set for CR LF.

SYNTAX `STERM[;]{term[term]|[NONE]}`
or
`STE[;]{term[term]|[NONE]}`

term is one of CR, LF, $char, or 'X, specifying a terminator character.

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES
```
PRINT#1,"STERM CR"
```
Set the serial output terminator to carriage-return.

```
PRINT#1,"STERM NONE"
```
Disable sending any serial output terminators.

```
PRINT#1,"STERM $&H0D"
```
Set the serial output terminator to carriage-return.

```
PRINT#1,"STERM $0"
```
Set the serial output terminator to NULL.

TERM

The TERM command sets the end-of-line terminators for output to IEEE bus devices. All output to IEEE bus devices, except `OUTPUT #count`, is terminated by the IEEE bus output terminator. All `ENTER` input from IEEE bus devices is terminated on a Line Feed (LF) or optionally specified with the `ENTER` command.

During `OUTPUT`, the Micro488/EX takes the data it receives from the user's program, strips all CR and LF characters from it (except `OUTPUT #count`) and appends the IEEE bus output terminator before sending it to the IEEE bus device. The default terminators for output are set by internal DIP switches and are factory set to `CR LF`, which is appropriate for most bus devices.

`EOI` has a different meaning when specified for input than when it is specified for output. During input, EOI specifies that input will be terminated upon detection of the EOI bus signal, regardless of which characters have been received. During output, `EOI` specifies that the EOI bus signal is to be asserted during the last byte transferred.

SYNTAX `TERM[;]{term[term][EOI]|[EOI]|[NONE]}`
or
`TE[;]{term[term][EOI]|[EOI]|[NONE]}`

term is one of `CR`, `LF`, `$char`, or `'X`, specifying a terminator character.

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES
```
PRINT#1,"TERM CR LF EOI"
```
Set output bus terminators to carriage-return line-feed, with EOI asserted on line-feed.

```
PRINT#1,"TERM LF EOI"
```
Set output term to LF with EOI.

```
PRINT#1,"TERM 'Z"
```
Set bus term to the letter "Z".

```
PRINT#1,"TERM $0 EOI"
```
Set output term to NULL with EOI.

TIME

The TIME command outputs the internal clock time in the format determined by the TIME FORMAT command, followed by the programmed serial output terminators. Refer to the TIME FORMAT command for a list of the available format options. This time string can also include optional DATE and DAY information. If DATE or DAY information is requested, they are appended to the TIME information in the same order, delimited by spaces.

SYNTAX TIME [DATE | DAY | DATE DAY | DAY DATE]

TIME returns the time in the format determined by the TIME FORMAT command.
The optional DATE returns the date in the format determined by the DATE FORMAT command.
The optional DAY returns the day of the week in the format determined by the DAY FORMAT command.

RESPONSE Returns time and optionally date and day information

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"TIME"
INPUT #1,T$      Read the time string
PRINT T$         Printed to the screen
11:30:20 AM      Output is format dependent

PRINT#1,"TIME DAY DATE"
INPUT #1,T$      Read the date, day and time string
PRINT T$         Printed to the screen
11:30:20 AM Monday 11-04-52
                 Output is format dependent
```

TIME FORMAT

The TIME FORMAT command allows the user to select the format of the TIME command response. Several different formats are provide including 12 and 24 hour representations. The power-on factory default is HH:MM:SS AM. This default can be modified with the SAVE command. Refer to the SAVE command for details.

The relationship between the 12 and 24 hour operations are…

	24 Hour	12 Hour
Midnight	00:00:00	12:00:00 AM
	01:00:00	01:00:00 AM
	…	…
	11:59:59	11:59:59 AM
Noon (Midday)	12:00:00	12:00:00 PM
	13:00:00	01:00:00 PM
	…	…
	23:59:59	11:59:59 PM

SYNTAX TIME FORMAT **[;]** **{**option**}**

The option specifies one of the following formats…

option	Example Output	Length	12/24 Hr
HH:MM	12:21	5 Bytes Fixed	24 Hr
HH:MM:SS	12:21:32	8 Bytes Fixed	24 Hr
HH:MM /AM	12:21	5 Bytes Fixed	12 Hr
HH:MM:SS /AM	12:21:32	8 Bytes Fixed	12 Hr
HH:MM AM	12:21 AM	8 Bytes Fixed	12 Hr
HH:MM:SS AM	12:21:32 AM	11 Bytes Fixed	12 Hr

The AM or /AM is used to set the 12 hour output format. It may be interchaged with PM or /PM which has the same effect.

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"TIME FORMAT HH:MM:SS AM"
PRINT#1,"TIME"
INPUT #1,T$          Read the time string
PRINT T$             Printed to the screen
01:12:22 PM          Output is time dependent

PRINT#1,"TIME FORMAT HH:MM:SS /AM"
PRINT#1,"TIME"
INPUT #1,T$          Read the time string
PRINT T$             Printed to the screen
01:12:23             Output is time dependent

PRINT#1,"TIME FORMAT HH:MM:SS"
PRINT#1,"TIME"
INPUT #1,T$          Read the time string
PRINT T$             Printed to the screen
13:12:24             Output is time dependent
```

TIME OUT

The TIME OUT command sets the number of seconds that the Micro488/EX will wait for an IEEE bus transfer before declaring a time out error. The Micro488/EX checks for time out errors on every byte (including command bytes as a controller) it transfers.

Time out checking may be suppressed by specifying time out after zero seconds. The default time out is 0 seconds, or time out disabled.

SYNTAX `TIME OUT[;][n]`
or
`TI[;][n]`

`n` is the number of seconds to allow in the range of 0 to 65535. If `n` is zero or unspecified, ignore time outs.

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES

```
PRINT#1,"TIME OUT 10"      Wait 10 sec before time out
PRINT#1,"TIME OUT 3600"    Wait an hour before time out
PRINT#1,"TIME OUT 0"       Ignore time outs.
```

TRACE

The TRACE ON command allows the embedded macro commands within the macro buffer to be echoed out the serial port to the host computer as the Macro is executed. This allows trace debugging during Macro execution. This feature is disabled with the TRACE OFF command.

If tracing is enabled during LOG ON, the trace output will also go to the LOG buffer.

SYNTAX	TRACE{ON |OFF}	
RESPONSE	Echoes macro commands during macro execution	
MODE	Any	
BUS STATES	None	
EXAMPLES	PRINT#1,"TRACE ON"	Enable Tracing
	PRINT#1,"DOMACRO 0"	Execute and trace macro #0

TRIGGER

The TRIGGER command issues a Group Execute Trigger (GET) bus command to the specified devices. If no addresses are specified, then the GET will only affect those devices that are already in the listen state as a result of a previous OUTPUT or SEND command.

SYNTAX TRIGGER[addr[,addr…]]
or
TR[addr[,addr…]]

addr is a bus device address to be triggered.

RESPONSE None

MODE CA

BUS STATES ATN•GET (without addr)
ATN•UNL, MTA, LAG, GET (with addr)

EXAMPLES
```
PRINT#1,"TRIGGER02,04,16"
```
Issue Group Execute Trigger to devices 2, 4, and 16.

```
PRINT#1,"TRIGGER"
```
Trigger all current listeners.

WAIT

The WAIT command informs the Micro488/EX to WAIT until a specified day, date or time before continuing additional command executions. If DATE, DAY or TIME is not specified with the WAIT command, TIME is assumed.

SYNTAX `WAIT [TIME] [;] hh:mm [:ss ] [ AM | PM ]`

hh specifies hours in 24 hour format unless AM or PM is included
mm specifies minutes
ss specifies seconds, set to 00 if omitted

or

`WAIT DATE [;] {date}`

If a European format has been selected, the WAIT DATE command requires the date specifier to be in the same order.

American date	Example
MM/DD/YY	12/04/76
MM-DD-YY	12-04-76
MM/DD/YYYY	12/04/1976
MM-DD-YYYY	12-04-1976
MON DD YYYY	DEC 04 1976
MON DD, YYYY	DEC 04, 1976
MONTH DD YYYY	DECEMBER 4 1976
MONTH DD, YYYY	DECEMBER 4, 1976

European date	Example
DD/MM/YY	04/12/76
DD-MM-YY	04-12-76
DD/MM/YYYY	04/12/1976
DD-MM-YYYY	04-12-1976
DD MON YYYY	04 DEC 1976
DD MONTH YYYY	4 DECEMBER 1976

or

`WAIT DAY [;] {d | da | day}`

`d` is a numeric representation from `1` (sunday) to `7` (saturday).
`da` is an abbreviated representation (`SUN`, `MON`, `TUE`, `WED`, `THU`, `FRI`, `SAT`)
`day` is an un-abbreviated representation (`SUNDAY`, `MONDAY`, `TUESDAY`, `WEDNESDAY`, `THURSDAY`, `FRIDAY`, `SATURDAY`)

RESPONSE None

MODE Any

BUS STATES None

EXAMPLES
```
PRINT#1,"WAIT DATE JAN 1, 1988"
PRINT#1,"WAIT TIME 1:00 AM"
PRINT#1,"LOG ON"
PRINT#1,"DOMACRO 1,100,60"
```

This example causes the Micro488/EX to wait until January 1st, 1988.
Then Wait until 1:00 AM, begin logging Macro #1, 100 times
with an interval of 1 minute between executions.

Controller Pass-Thru Operation

6.1 Introduction

The Controller Pass-Thru mode allows a serial RS-232 or RS-422 host device to transparently send data to a single IEEE bus peripheral or to multiple peripherals if they occupy the same bus address. Applications include interfacing a listen-only or addressable IEEE printer/plotter to a serial printer port.

Once the Micro488/EX has initialized itself after power-on, it waits for serial input data. When received, it addresses the selected IEEE device to listen with the following bus sequence:

ATN•UNL,MTA,LAG,*ATN

The data received from the serial host is placed into a circular serial input buffer. Simultaneously, characters are removed from that buffer and sent to the IEEE bus device. The serial terminator(s), if present, are not sent. Instead, the IEEE terminators are substituted and sent in their place.

So long as the serial input buffer is not empty, the Micro488/EX will continue to send data from it to the IEEE bus device. If the serial input buffer becomes emptied, the Micro488/EX will command the IEEE bus device to talk if the talk back feature is enabled. This allows the Micro488/EX to be used as a controller with devices, such as plotters or instruments, that return status and other information to the host computer.

When the Micro488/EX addresses the IEEE bus device to talk it uses the following bus sequence:

ATN•UNL,MLA,TAG,*ATN

The Micro488/EX then accepts data from the IEEE device and returns it to the host until the last selected IEEE terminator is detected. The IEEE bus terminators are replaced by the serial terminators and these are then sent to the serial host.

If the IEEE device has been addressed to talk but does not respond or finish transmission by the time additional characters are received into the circular serial input buffer, the talk sequence will be aborted to allow additional serial information to be sent to the IEEE device.

6.2 Serial and IEEE Terminator Substitution

The Micro488/EX can be configured to provide serial to IEEE 488 and IEEE 488 to serial terminator substitution. This is useful when interfacing a serial host which only issues carriage return [CR] as an output terminator to an IEEE peripheral which expects a carriage return followed by a line feed [CR-LF].

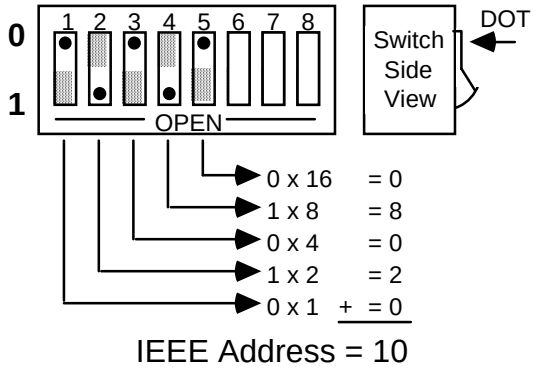
In this previous example, the serial terminator should be selected for CR Only while the IEEE terminator is set for CR-LF. When a serial CR character is received it is discarded and substituted with an IEEE CR followed by an IEEE LF. In the IEEE to serial direction, the IEEE CR is unconditionally discarded. Upon receipt of the IEEE LF a serial CR is substituted.

The Micro488/EX can be made totally data transparent by setting both the serial and IEEE terminators to be CR Only or LF Only. Refer to Section 2 for the proper switch settings for both the IEEE and serial terminators.

6.3 IEEE Address Selection

SW3-1 through SW3-5 select the IEEE bus address of the IEEE peripheral the Micro488/EX will be communicating with. In this mode, these switches set the address of the IEEE device that will be controlled, not the address of the Micro488/EX. The address of the Micro488/EX is automatically adjusted so that address conflicts will not occur. The address is selected by simple binary weighting with SW3-1 being the least significant bit and SW3-5 the most significant. If address 31 (reserved on the IEEE bus) is selected in the controller mode, address 30 is assigned as the device it will be communicating with. The following figure shows the IEEE address selection of 10.

SW3 View for IEEE Address Selection

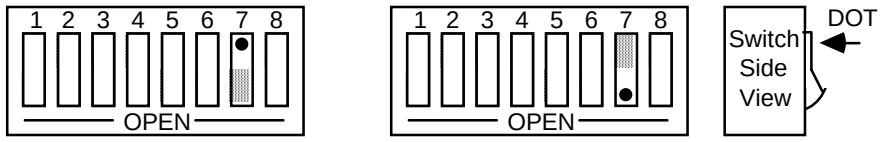


6.4 Talk Back On Terminator

A switch selectable talk back feature is included to provide bi-directional communication with the IEEE device. Whether the talk back feature should be enabled is dependent on the application.

SW1-7 is used to determine whether the interface should address the attached bus device to talk after sending the selected IEEE bus terminator(s). This feature is commonly used to provide bi-directional communication with a single IEEE instrument. Talk back will only occur if there is no serial data to output to the IEEE device.

SW1 View for Talk-Back Selection



When the serial input buffer becomes empty, the Micro488/EX checks the last characters sent to the IEEE bus device. If these were the IEEE bus terminators and Talk-Back is enabled, the IEEE bus device is addressed to talk. Any data received by the Micro488/EX from the bus device is sent to the serial host.

When the last IEEE bus terminator is detected from the IEEE device, the Micro488/EX disables the device from sending additional information by asserting Attention (ATN) on the bus.

If the IEEE device does not responded or finish transmission by the time additional characters are received into the serial input buffer, the talk sequence will be aborted to allow additional serial information to be sent to the IEEE device.

The following is an example of how this feature can be used to communicate with a single IEEE instrument. The program example is written in Basic on an IBM PC or compatible and communicates with a Keithley Model 196 DMM.

```
10   '
20   '      Example Program using Micro488/EX with
25   '  the Talk Back on Terminator Feature Enabled to
30   '   Communicate with a Keithley Model 196 DMM
40   '
50   ' Open Basic's serial communications port
60   OPEN "COM1: 9600,N,8,2" AS 1
70   ' Set the Model 196 DMM to the 30VDC range
80   PRINT #1,"F0R3X"; ' The ; suppresses terminators
90   ' Request 10 Readings from 196"
100  FOR N = 1 to 10
110     PRINT #1,"" ' Output terminator
120     LINE INPUT #1, A$ ' Get Reading from 196
130     PRINT A$ ' print it on the screen
140  NEXT N
150  END
```

6.5 Plotter Applications

To use the Micro488/EX to interface an HP-IB plotter to a serial computer port, you will need the following information about your system.

1. The serial data format that the application (plotting or graphics) program expects the plotter to communicate with. These parameters include baud rate, word length, stop bits, parity and serial control.

Some programs allow these parameters to be selected by the user. Other graphics programs depend on the RS-232 version of the plotter defaults. Usually, Hewlett Packard plotters use 9600 baud, 7 data bits, 1 stop bit, even parity and XON/XOFF serial control. Since these plotters are available with serial interfaces, the operator's manual of your IEEE plotter should contain this information.

2. The IEEE bus address of your plotter. This address is usually set by a DIP switch located on the rear of the plotter. The first five switches set the address which, for Hewlett Packard plotters, is usually address 5. Refer to the plotter's operator's manual for exact information.

Set the Micro488/EX's internal DIP switches to match the parameters determined above. Other parameters which should be selected include…

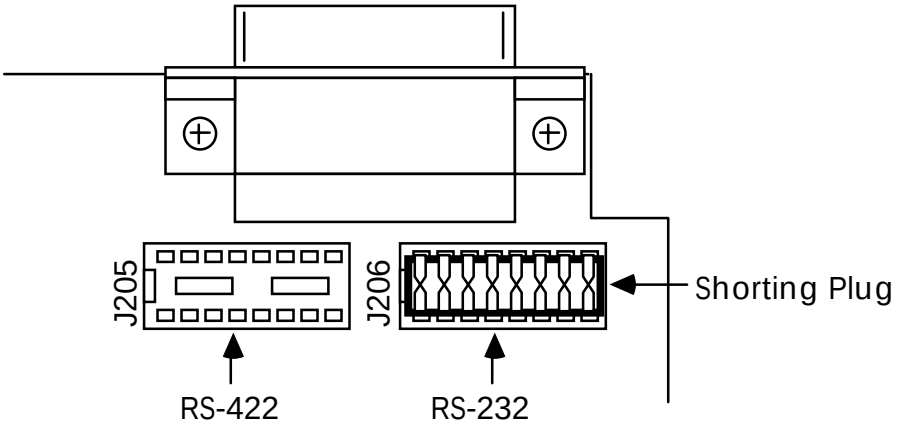
1. Talk Back Enabled.

2. Serial Terminators set to CR Only.

3. IEEE Terminators set to CR Only with EOI enabled.

The following shows the Micro488/EX's internal switch settings required to use a Hewlett Packard 7580A plotter on an IBM PC or compatible. Because PC and compatibles output RS-232 levels, the shorting DIP jumper should be set to the RS-232 position (J206).

Selecting RS-232 Signal Levels



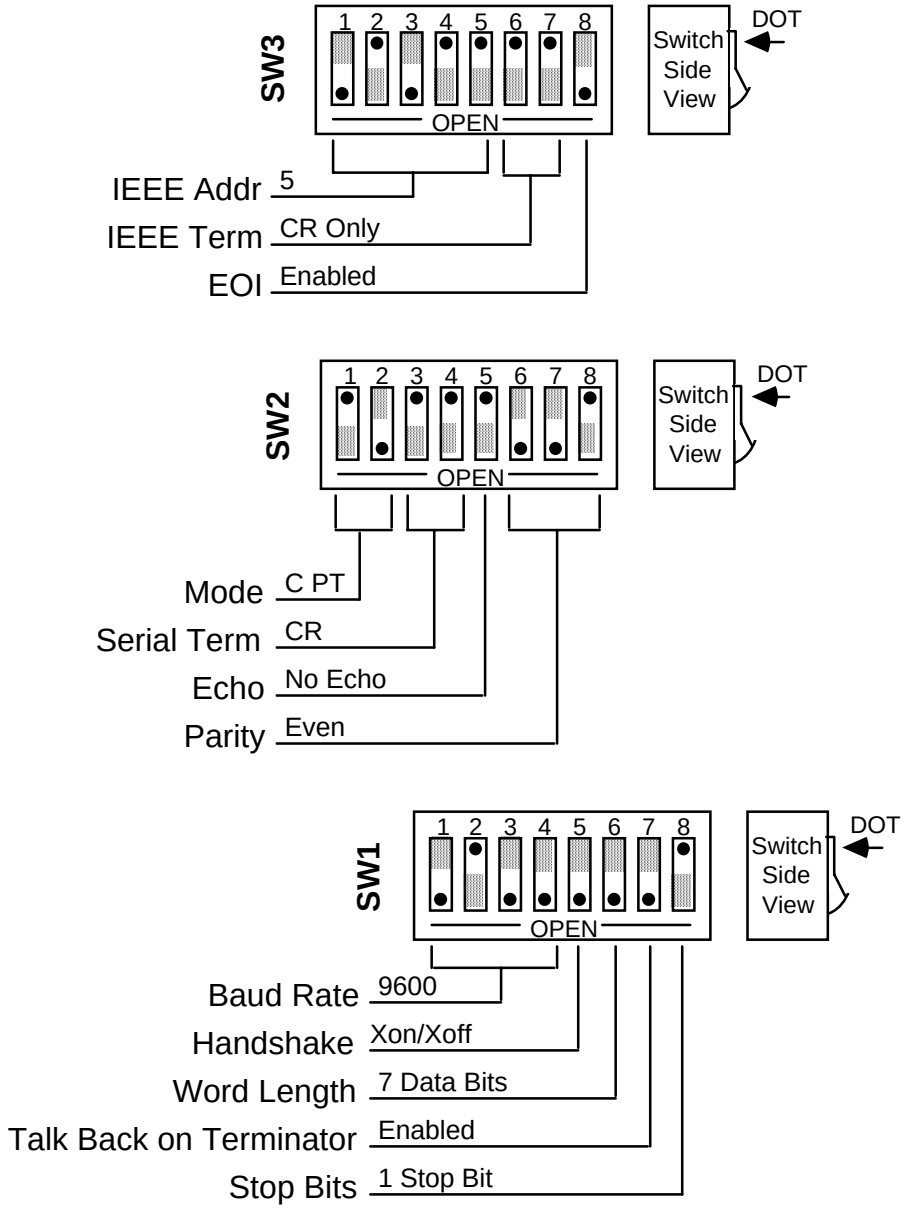
After configuration, turn on the plotter and the Micro488/EX. The Micro488/EX's front panel LEDs should all light momentarily while it performs an internal ROM and RAM test. All LEDs should go out except for the Power and Talk LED. The Talk LED indicates that the Micro488/EX has detected the plotter on the IEEE bus and has addressed it to listen.

When the serial host begins to send the Micro488/EX data, the Receive LED will flash. If it does not, this indicates that the interface is not receiving data from the serial host. Verify the cables are connected properly and the serial cable wiring. Verify the serial data format, word length, stop bits and parity.

6.6 Printer Applications

Most of the information given for plotter applications applies to applications for interfacing IEEE 488 printers to a serial host. Some high end printers have a secondary command setting which must be disabled for the Micro488/EX to control them. The Micro488/EX does not use secondary commands to control IEEE peripherals, such as printers or plotters. Refer to the printer's instruction manual if the is a question as to whether the printer requires secondary commands.

Micro488/EX Settings For Use With HP 7580A Plotter on an IBM PC



Peripheral Pass-Thru Operation

7.1 Introduction

The Peripheral Pass-Thru mode of operation is useful in interfacing a serial device, such as a serial printer, plotter or instrument, to an IEEE controller. Data which is sent by the IEEE controller to the Micro488/EX is buffered and transmitted out its serial port. Data received from the serial device is buffered by the Micro488/EX until read by the IEEE controller. The Micro488/EX can buffer approximately 29,000 bytes of data from both the IEEE input and the serial input. The actual amount of memory available for buffering is dependent on the memory which has been allocated for Macro and Log buffer storage in the Controller or Periperal Modes. Refer to Sections 4 and 5 for more information.

The Micro488/EX will refuse to accept more data from the IEEE controller when its buffer memory is full. It does this by preventing completion of the bus handshaking sequences. It will also request that additional serial data not be sent by negating its Request To Send (`RTS`) output or by transmitting the `XOFF` ASCII character. The serial handshake used is dependent on the handshake selection (Refer to Section 2).

7.2 Serial and IEEE Input Buffers

Memory in the Micro488/EX is dynamically allocated for the serial input and IEEE input buffers. This allows for the most efficient partitioning of memory for any given application.

At power on, or device clear, each buffer is allocated a 127 byte mini-buffer or queue. When the serial input [or IEEE input] requires more buffer space, additional queues are allocated. When a queue is empty, it is released from the input buffers so that it may be re-allocated when, and where, required.

There are approximately 240 available queues for a total of 29,000 bytes of buffer (character) space. Queues are continually allocated and released as required by the serial and IEEE input. Of the 240 available queues, 230 are issued without regard to controlling the receipt of additional serial or IEEE input data.

When the serial input buffer requests one of the last 10 queues (1270 character locations left), it signals the serial host that it should stop sending data. This is accomplished by either un-asserting RTS or issuing XOFF, depending on which serial handshake control has been switch selected. When more than 10 queues become available, it asserts RTS or issues XON.

The IEEE bus input signals that the IEEE input (or serial output) buffer is full when the number of queues available drops below 10 (1280 character locations left). When the number of available queues drops to 4 or less (512 character locations left), the IEEE interface of the Micro488/EX stops accepting data from the bus. This bus hold-off will only occur until additional queues (greater than 4) become available. At that time it will resume accepting bus data.

7.3 IEEE Data Transfers

The following methods may be used by the IEEE controller when sending data to the Micro488/EX:

7.3.1 Blind Bus Data Transfers

If the IEEE controller does not mind waiting an indefinite time for data space in the buffer to become available, the data can simply be sent to the Micro488/EX. This is referred to as blind data transfers because the IEEE controller is blind as to whether or not the Micro488/EX is capable of accepting data. In this case, the bus controller's output data transfer will be held off by the Micro488/EX if it is unable to buffer the data. It will resume accepting IEEE input data when memory becomes available. This type of control might be appropriate in a single user environment.

To illustrate how this would appear, let's assume the Micro488/EX is connected to a serial device which will accept data at 1200 baud or 110 bytes per second. The IEEE bus controller is capable of sending data to the Micro488/EX at a rate of 5000 bytes per second. The data would be transferred on the bus at 5000 characters per second for slightly over six seconds, filling over 31,000 locations. At that time, the IEEE input would hold off additional data transfers until 128 characters are sent out the serial port at rate of 110 characters per second. This 110 cps would then become the average bus data acceptance rate of the Micro488/EX.

If the controller is set to detect a data time-out error, then it will do so if the Micro488/EX holds off IEEE input data transfers for too long. The error can be used to alert the operator to the problem, such as a printer out of paper, so that it can be corrected. If the controller then restarts transmission exactly where it left off, no data will be lost.

If data is requested by the controller and no serial input data is available in the Micro488/EX, the bus will hang until serial data is received. If no serial data is received it will hang forever or until the controller times out.

7.3.2 Controlled Bus Data Transfers

If the controller must avoid waiting for the serial device, it can 'serial poll' the Micro488/EX. Serial poll is a method by which the controller can inquire the internal status of the interface without disturbing any data being transferred, slowing data transfers or locking up the bus. You should refer to the programming manual of your controller to determine the method of performing serial polls.

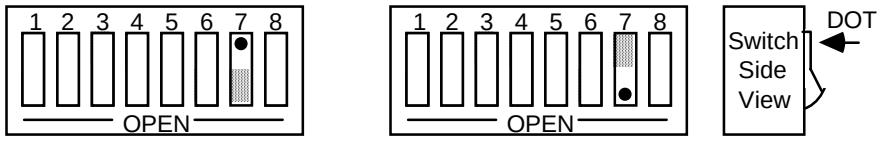
When serial polled, the Micro488/EX provides eight bits of status information to the controller. The most significant bit [DIO8] of the Macro488's serial poll byte is set to a logic "1" when the IEEE input buffer is NOT EMPTY. The term NOT EMPTY is used to signify that not all of the previous data sent to the interface has been transmitted to the serial device. If it is NOT EMPTY, the controller may avoid sending any more data to the Micro488/EX. If this bit is a logic "0", then the serial device has accepted all previous data and the IEEE controller may send more.

Another bit [DIO4] of the Serial Poll byte is used to indicate additional information concerning the IEEE input buffer. This bit is set to a logic "1" when there is 1280 or less locations in the buffer for data. It is cleared, set to a logic "0", when there is greater than 1280 locations available. This bit is referred to as the IEEE input buffer FULL bit.

When serial data is received, DIO5 of the Serial Poll byte is set, '1', to indicate to the IEEE controller that the serial input buffer is NOT EMPTY. If set, it indicates that at least one character is available in the serial input buffer to be read by the IEEE controller. Once all of the serial input data is read by the IEEE controller this bit is reset.

The Micro488/EX can generate a request for service on the bus when it receives the last serial terminator. To enable this feature, the Pass-Thru Feature switch, located on the internal switch bank of SW1, must be set to open. When enabled, the Micro488/EX will assert the IEEE bus SRQ line and set serial poll status bits DIO7 and DIO3 when the last serial terminator is detected. The IEEE controller must perform a serial poll on the interface to clear the SRQ. If the Pass-Thru Feature switch is in the closed position, there will not be any indication in the serial poll status byte that a serial terminator has been received.

SW1 View For Selecting SRQ on Last Terminator



SRQ on Last Terminator Disabled

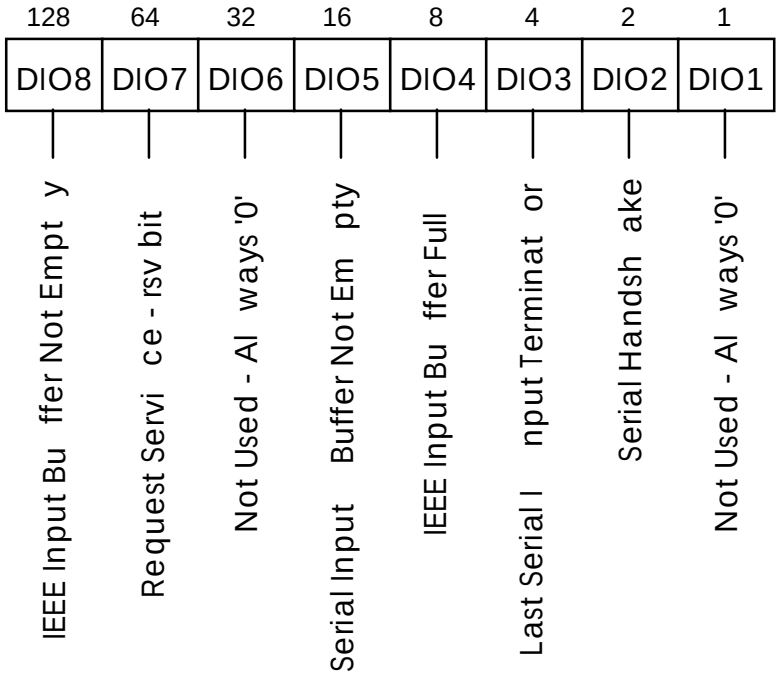
SRQ on Last Terminator Enabled

7.4 Serial Poll Status Byte Register

The following shows and describes the serial poll status information provided by the Micro488/EX.

DIO8 IEEE Input Buffer NOT EMPTY
This bit is set when the IEEE input buffer contains one or more data bytes which have not been sent out the serial port. It is cleared, set to "0", when the buffer is empty.

Serial Poll Status Byte



DIO7 *rsv*
This bit is defined by the IEEE 488 Specification and is used to indicate to the bus controller that the Micro488/EX is the bus device that requested service. It is cleared when the interface is serial polled by the controller.

DIO6 Not Defined - Always "0"

DIO5 Serial Input Buffer NOT EMPTY
This bit is set when the serial input buffer contains one or more data bytes which have not been sent out the IEEE bus. It is cleared, set to "0", when the buffer is empty.

DIO4 IEEE Input Buffer FULL
When this bit is set, it indicates that the Micro488/EX may hold off the controller on subsequent data transfers. The interface may continue to accept an additional 512 characters but this is dependent on the serial input buffer size.

DIO3 Received Last Serial Terminator
If the Peripheral SRQ feature is enabled, the Micro488/EX will issue a request for service by asserting the SRQ line and setting this bit along with the rsv bit [DIO7]. It is cleared, along with rsv, when serial polled by the controller. If this feature is not enabled, this bit is always "0".

DIO2 Serial Handshake
This bit indicates the present state of the serial handshake. If it is set to "1", the serial device connected to the Micro488/EX is capable of accepting serial data. If "0", the RTS line is unasserted, if configured for hardware handshake, or the XOFF character has been received, if configured for XON/XOFF software handshake.

DIO1 Not Used - Always "0"

7.5 Use of Serial and Bus Terminators

The Micro488/EX can be configured to provide serial to IEEE 488 and IEEE 488 to serial terminator substitution. This is useful when interfacing a serial device, which only issues carriage return [CR] as an output terminator, to an IEEE controller, which expects a carriage return followed by a line feed [CR-LF].

In this previous case, the serial terminator should be selected for CR Only while the IEEE terminator is set to CR-LF. When a serial CR character is received it is discarded and substituted with an IEEE CR followed by an IEEE LF. In the IEEE to serial direction, the IEEE CR is unconditionally discarded. Upon receipt of the IEEE LF a serial CR is substituted.

The Micro488/EX can be made totally data transparent by setting both the serial and IEEE terminators to be `CR Only` or `LF Only`. The choice of appropriate terminators may be determined by inspection of the serial device and IEEE controller's instruction manuals. For selection of the Micro488/EX's serial and bus terminators you should refer to Section 2 of this manual.

7.6 IEEE 488 Bus Implementation

As a pass-thru bus peripheral, the Micro488/EX implements many of the capabilities defined by the IEEE 488 1978 specification. These are discussed in the following sub-sections. The bus uniline and multiline commands that the Micro488/EX does not support or respond to include:

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

7.6.1 **My Talk Address** (MTA)

When the Micro488/EX is addressed to talk, it retrieves data from the serial input buffer and outputs it to the IEEE 488 bus. It substitutes the selected IEEE bus terminators for the received serial terminators. The Micro488/EX will continue to output serial input buffer data as long as the IEEE controller allows.

7.6.2 **My Listen Address** (MLA)

When the Micro488/EX is addressed to listen, it accepts data from the active talker and outputs this data through the serial interface. It substitutes the selected serial terminators for the received IEEE bus terminators.

7.6.3 **Device Clear** (DCL and SDC)

Device Clear resets the Micro488/EX's IEEE input and serial input buffers. Any pending data and Service Requests (SRQ), including the information they convey, are lost. In addition, the `XON` serial control character is transmitted if `XON/XOFF` is enabled.

7.6.4 **Interface Clear** (IFC)

IFC places the Micro488/EX in the Talker/Listener Idle State. It clears any pending requests for service (SRQ). The condition which caused the SRQ remains unmodified.

7.6.5 **Serial Poll Enable** (SPE)

When Serial Poll Enabled, the Micro488/EX 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 Micro488/EX will continue to try to output its serial poll response until it is 'Serial Poll Disabled' by the controller.

7.6.6 **Serial Poll Disable** (SPD)

Disables the Micro488/EX from responding to serial polls by the controller.

7.6.7 **Unlisten** (UNL)

UNL places the Micro488/EX in the Listener Idle State.

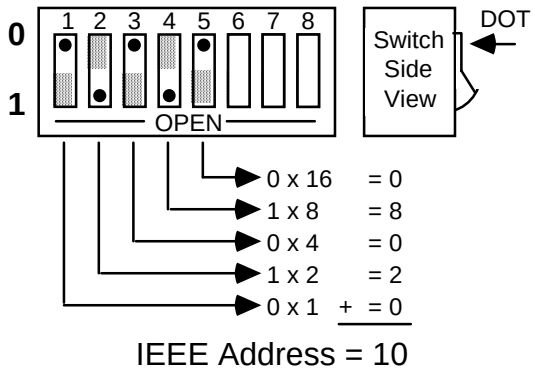
7.6.8 **Untalk** (UNT)

UNT places the Micro488/EX in the Talker Idle State.

7.7 IEEE Address Selection

SW3-1 through SW3-5 select the IEEE bus address of the Micro488/EX when in the Peripheral Pass-Thru mode. The address is selected by simple binary weighting with SW3-1 being the least significant bit and SW3-5 the most significant. The following figure shows the IEEE address of the Micro488/EX set to 10.

SW3 View for IEEE Address Selection



7.7.1 Listen Only Mode

Listen Only is a special type of Peripheral operation. In the Listen Only mode the Micro488/EX accepts all data transmitted on the bus and transfers it out its serial port. The Micro488/EX is set to Listen Only mode by setting its address to 31 (switches SW3-1 through SW3-5 all open).

7.8 IEEE to Serial Applications

The following program uses a Micro488/EX as an interface to a serial instrument or host computer. The IEEE controller is an IBM PC running GWBasic with the IOtech Personal488™ controller package. Communications are provided under direct interaction from the keyboard.

In this program example, key presses are detected and sent via the IEEE bus to the Micro488/EX. The character is then sent to the serial device. Any incomming serial characters are buffered by the Micro488/EX. The Micro488/EX is polled by the controller for any data in the serial input buffer. When data is detected, it is read by the controller one character at a time and printed on the PC's screen. The IEEE address of the Micro488/EX is 10.

```
10 ' Open IOtech Driver488 Files and initialize
20 OPEN "\DEV\IEEEOUT" FOR OUTPUT AS 1
30 IOCTRL #1,"BREAK"
40 PRINT #1,"RESET"
50 OPEN "\DEV\IEEEIN" FOR INPUT AS 2
60 ' Look for PC Key Press
70 K$ = INKEY$
80 IF K$="" THEN GOTO 110
90 ' Output Key Press to Micro488/EX
100 PRINT #1,"OUTPUT 10;";K$;
110 ' Test for Serial data
120 PRINT #1,"SPOLL 10" : INPUT #2,SPOLL
130 IF NOT (SPOLL AND 16) THEN GOTO 70
140 ' Enter One Byte From Micro488/EX and print it
150 PRINT #1,"ENTER 10 #1" : S$ = INPUT$(1,1) : PRINT
S$;
160 GOTO 120 ' Try for more
```

IEEE 488 Primer

8.1 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 Macro488 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.

8.2 General Structure

The main purpose of the GPIB is to transfer information between two or more devices. A device can either be 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 proceedings. For the GPIB, this device becomes the Active Controller.

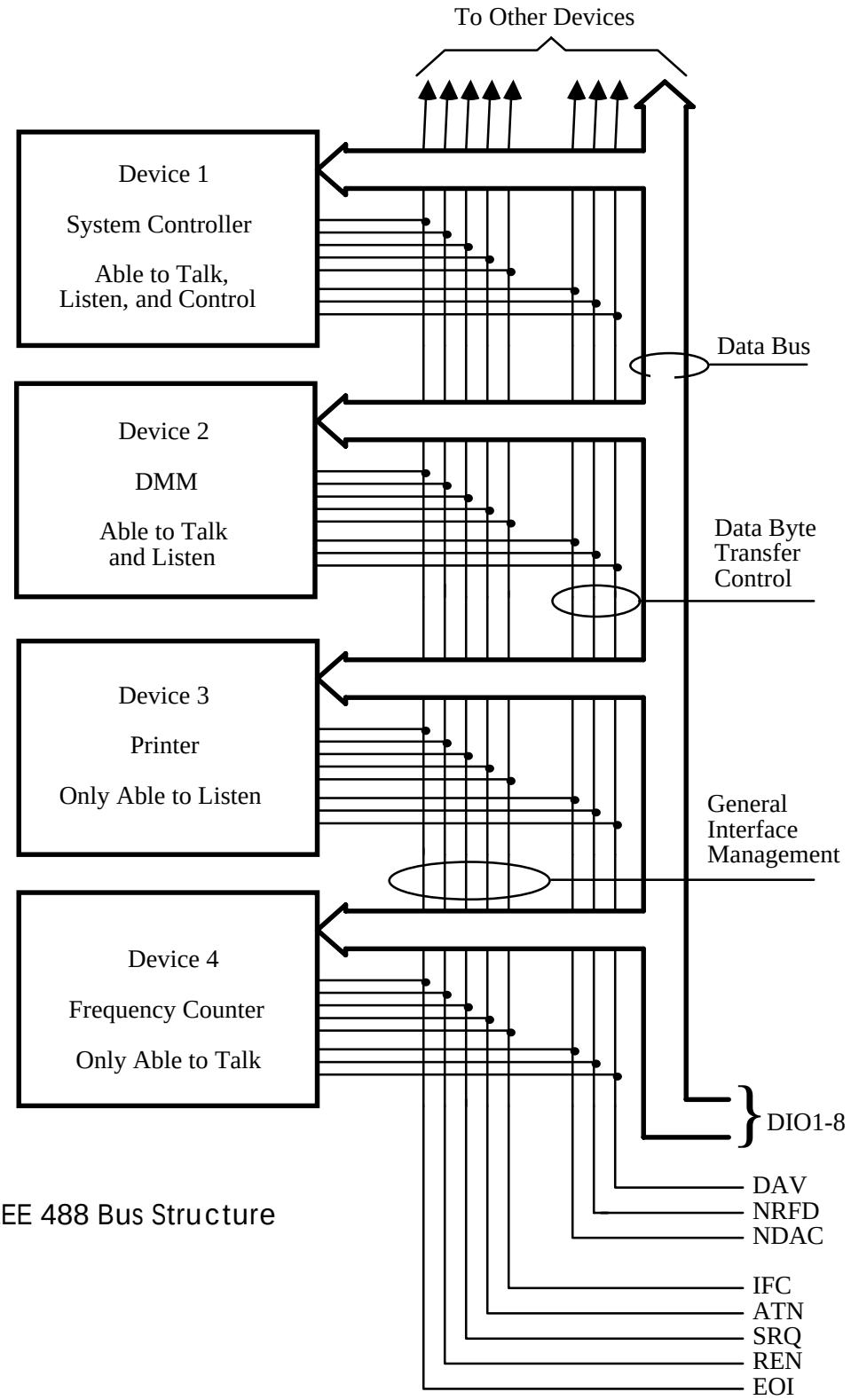
At a committee meeting, everyone present usually listens. This is not the case with the GPIB. The Active Controller selects which devices will listen and commands all other devices to ignore what is being transmitted. A device is instructed to listen by being Addressed to Listen. This device is then 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 a lot of 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 who 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

8.3 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).

8.4 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).

8.4.1 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 Listeners. The Active Controller is the only bus device that has control of this line.

8.4.2 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).

8.4.3 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.

8.4.4 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.

8.4.5 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.

8.5 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.

8.5.1 Data Valid (DAV)

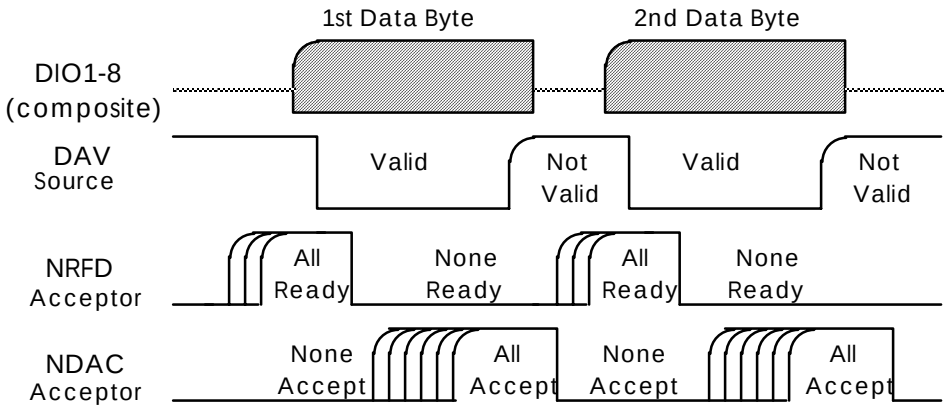
The DAV line is controlled by the Talker. The Talker verifies that NDAC is asserted (active low) which indicates that all Listeners have accepted the previous data byte transferred. The Talker then outputs data on the bus and waits until NRFD is unasserted (high) which indicates that all Addressed Listeners 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.

8.5.2 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 this 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.

8.5.3 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

8.6 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 bidirectional and are active low.

8.7 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.

8.7.1 Go To Local (GTL)

This command allows the selected devices to be manually controlled. (`&H01`)

8.7.2 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.

8.7.3 Unlisten (UNL)

This command tells all bus devices to Unlisten. The same as Unaddressed to Listen. (`&H3F`)

8.7.4 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.

8.7.5 Untalk (UNT)

This command tells bus devices to Untalk. The same as Unaddressed to Talk. (`&H5F`)

8.7.6 Local Lockout (LLO)

Issuing the LLO command prevents manual control of the instrument's functions. (`&H11`)

8.7.7 Device Clear (DCL)

This command causes all bus devices to be initialized to a pre-defined or power up state. (`&H14`)

8.7.8 Selected Device Clear (SDC)

This causes a single device to be initialized to a pre-defined or power up state. (`&H04`)

8.7.9 Serial Poll Disable (SPD)

The SPD command disables all devices from sending their Serial Poll status byte. (`&H19`)

8.7.10 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. (`&H18`)

8.7.11 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. (`&H08`)

8.7.12 Take Control (TCT)

This command passes bus control responsibilities from the current Controller to another device which has the ability to control. (`&H09`)

8.7.13 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. (`&H60` to `&H7F`)

8.7.14 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. (`&H05`)

8.7.15 Parallel Poll Unconfigure (PPU)

This disables all devices from responding to a Parallel Poll. (`&H15`)

8.8 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.

8.8.1 Serial Poll

Suppose the Controller receives a service request. For this example, let's assume there are several devices which 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.

8.8.2 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

9.1 Factory Service

IOtech maintains a factory service center in Cleveland, Ohio. If problems are encountered in using the Micro488/EX 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.

9.2 Theory Of Operation

The Heart of the Micro488/EX is a 6809 microprocessor [U101] supported by 16K bytes of firmware EPROM [U102 (27128)] and 32K bytes of static RAM [U103 (58256)]. A Versatile Interface Adapter [U104 (65B22)] is used to generate real-time interrupts for the firmware operating system. Non-volatile memory and clock data is provided by a DS1216 smart socket.

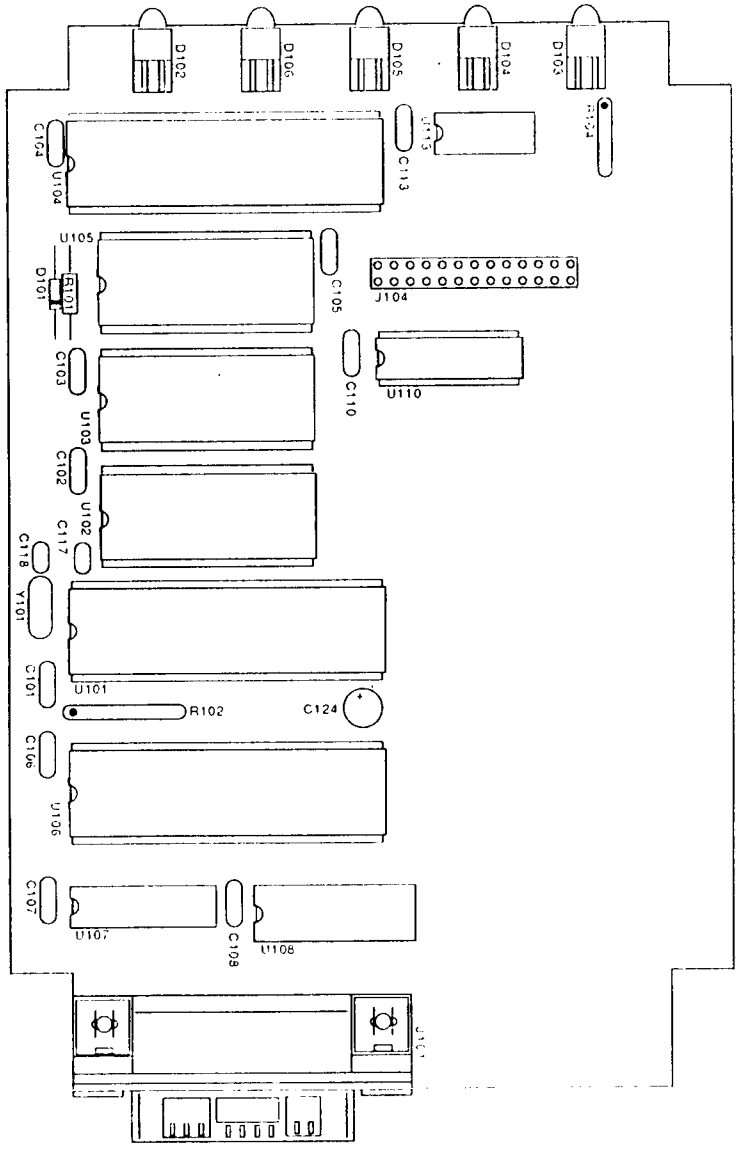
The front panel annunciators are also driven by U104 through an inverter [U113 (74LS04)]. The IEEE 488 bus interface is accomplished by a TMS9914A [U106] controller with drivers U107 and U108. The serial interface is provide by the UART [6551 (U105)]. If RS-232 levels are chosen, they are provided by the RS-232 transceiver (U209). If RS-422 levels are selected, the differential driver [26LS30 (U207)] and receiver [26LS33 (U208)] are used.

The internal DIP switches [SW1, SW2 and SW3] are read via 74HCT244 tri-state buffers [U201, U202 and U203]. Power is supplied by an external unregulated 9 volt wall mount supply. Regulation to the required +5 volts is provided by U206 [7805].

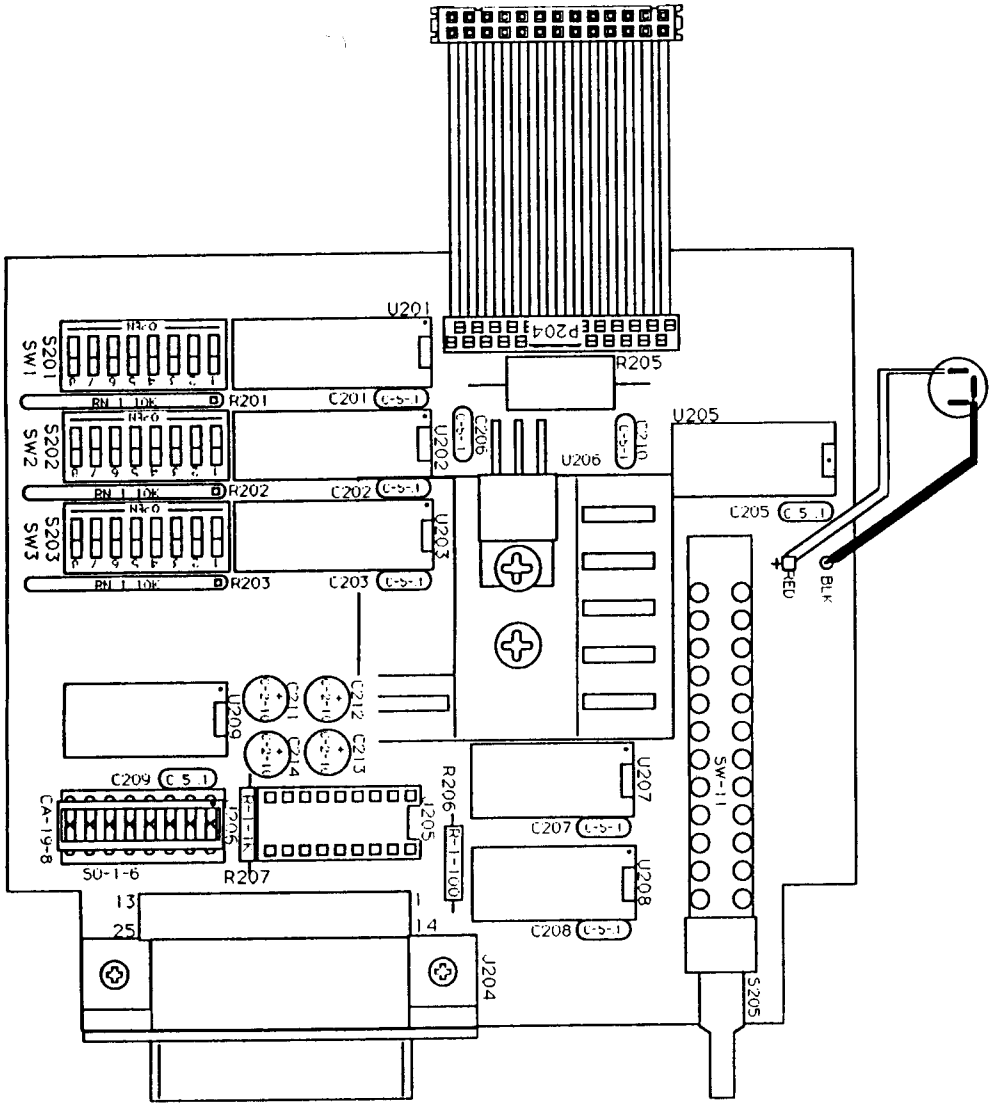
Decoding of the microprocessor address space is accomplished with a Programmable Logic Array [U110 (16L8)]. The Memory space allocation is

Address	Device	Part Number	Function
$0000-$7FFF	U103	58258	Static RAM/Clock
$A000-$A007	U106	9914A	IEEE Controller
$A800-$A807	U105	6551	UART
$B000-$B00F	U104	65B22	VIA
$B800	U201	74HCT244	SW1 (S201)
$B801	U202	74HCT244	SW2 (S202)
$B802	U203	74HCT244	SW3 (S203)
$C000-$FFFF	U102	27128	Programmed EPROM

9.3 Micro488/EX Mother Board Component Layout



9.4 Micro488/EX Serial I/O Board Component Layout



9.5 Replaceable Parts List

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
C211-C214	C-2-10	Electrolytic - 25v
C201-C203	C-5-.1	Ceramic, 25v
C205-C209	C-5-.1	Ceramic, 25v
C210	C-5-1	Ceramic, 25v
D101	RF-1	Small Signal Diode
D102-D106	DD-2	Red PC Mount
J101	CN-2	IEEE 488 Connector
J104	CN-5-13	13 x 2 0.1" Header
J201	CN-11	9 Volt Power Jack
J204	CN-8-25	PC Mount Female DB-25
J205	CA-19-8	8 Pos. DIP Jumper
P204	CA-6	26 Conductor Ribbon Assembly
R101	R-1-68K	68K•, 1/4w carbon
R102	RN-4-4.7K	4.7K• x 7 SIP
R104	RN-2-470	470• x 5 SIP
R201-R203	RN-1-10K	10K• x 9 SIP
R205	R-2-39	39•, 1w carbon
R206	R-1-100	100•, 1/4w carbon
R207	R-1-1K	1K•, 1/4w carbon
S201-S203	SW-6-8	8 Pole DIP
S205	SW-11	8 Pole DT Push Push
U100	IC-	DS1216 Smart Socket
U101	IC-1	MC68B09P Microprocessor
U102	Macro488-600	Programmed EPROM
U103	IC-78	58256-15 32K x 8 CMOS SRAM
U104	IC-23	65B22 Versitile Interface Adapter
U105	IC-16	R6551AP UART
U106	IC-3	TMS9914ANL IEEE Controller
U107	IC-4	SN75160BN IEEE Driver
U108	IC-5	SN75162BN IEEE Driver
U110	Macro488-602	Programming Equation - 16L8 PAL
U113	IC-33	74LS04 Hex Inverter
U201-U203	IC-39	74HCT244 Octal Buffer
U205	IC-21	SN74LS139 Dual Decoder
U206	IC-30	LM7805CT Regulator - +5v
U207	IC-38	26LS30 RS-423 Driver
U208	IC-36	26LS33 RS-422 Receiver
U209	IC-82	LT1081 RS-232 Transceiver
Y101	CR-4	7.3728 MHz Crystal

Micro488/EX Command Summary

@	@
	If followed immediately by a CR of LF, unlocks the Micro488/EX from an inappropriate command such as an ENTER from a non-existent bus device.
@@	@@
	Return Micro488/EX to power-on conditions.
ABORT	AB[ORT]
	Send IFC (SC) or MTA (*SC•CA). Stops bus activity.
ARM	AR[M] [;] [event[event…]]
	Enables transmission of event serial messages.
CASE	CASE[;]{UPPER|LOWER}
	Specifies whether the DAY and DATE text will be sent in upper case or in capitalized lower case.
CLEAR	CL[EAR] [addr[,addr…]]
	Clear all or selected devices.
COMMENT	COM[MENT] [;] 'data'
	Sends data to the serial host when executed.
COUNT	COUNT
	Returns the loop count of the last invoked Macro buffer.

DATE	DATE [DAY | TIME | DAY TIME | TIME DAY]
	Returns the date in the format determined by the DATE FORMAT command. Optionally returns DAY and TIME information.
DATE FORMAT	DATE FORMAT [;] {option}
	Sets the format of the DATE output. Refer to the DATE FORMAT command description for option details.
DAY	DAY [DATE | TIME | DATE TIME | TIME DATE]
	Returns the day of the week in the format determined by the DAY FORMAT command. Optionally returns DATE and TIME information.
DAY FORMAT	DAY FORMAT [;] {D | DA | DAY}
	Sets the format of the DAY output.
DELAY	DELAY [;] time
	Waits until time (in seconds) has elapsed before continuing.
DISARM	DI [SARM] [;] [event [event…]]
	Disables transmission of event messages and ON <event> DOMACRO actions.
DOMACRO	DO [MACRO] [;] [number [,loop [,interval]]]
	Execute commands in the number MACRO buffer loop number of times at the specified interval (seconds).

ENTER	EN[TER][addr][#count|term|EOI] EN[TER][addr][;count|;term|;EOI]
	Direct I/O. Read from bus device.
ERASE	ERASE[;][number]
	Erase selected or all MACRO buffers.
ERASE LOG	ERASE LOG
	Erase the data in the LOG buffer and returns the memory it consumed to the USER heap .
ERROR	ERROR[;]{MESSAGE|NUMBER|OFF}
	Select automatic error reporting.
FACTORY	FACTORY
	Resets the Micro488/EX to factory defaults. All Log Buffer data and Macros are lost. Causes an NVRAM Error which is cleared by issuing the SAVE command.
HELLO	HE[LLO]
	Read the Micro488/EX revision identification.
ID	ID;[ASCII]
	Change the ID character to ASCII.

LOCAL	LO[CAL][addr[,addr…]]
	Un-assert REN line (SC) or send Go To Local commands (CA).
LOCAL LOCKOUT	LOCAL L[OCKOUT] or LOL
	Prevent bus devices from acting upon manual (front panel) controls.
LOG	LOG {ON|OFF}
	Redirects command output that normally goes to the serial output buffer to the LOG buffer.
LOG MEMORY	LOG MEMORY
	Report the amount of memory in the LOG buffer.
MACRO…ENDM	MA[CRO][command list…]ENDM
	Build a MACRO with all the commands between MACRO and ENDM.
MASK	MASK{ON|OFF}
	Enable or disable serial input mask [&H7F] on OUTPUT and delimited sting data.
MEMORY	ME[MORY]
	Report the amount of available memory in the USER heap.
OUTPUT	OU[TPUT][addr[,addr…]][#count];data
	Direct I/O. Send data to bus device(s).

ON…DOMACRO `ON <event> DO[MACRO] [number]`

Enable Macro buffer `number` execution on `<event>`.

PASS CONTROL `PA[SS CONTROL] addr`

Make another bus device the Active Controller. Enter the Peripheral (*CA) state.

PPOLL `PPOLL`

Read the Parallel Poll response from all bus devices.

PPOLL CONFIG `PPOLL C[ONFIG] addr,response`
or `PPC addr,response`

Set the Parallel Poll response of a bus device. `response` is the decimal equivalent of four bits: `S P2 P1 P0`.

PPOLL DISABLE `PPOLL D[ISABLE] addr[,addr…]`
`or PPD addr[,addr…]`

Prevent bus device(s) from responding to a Parallel Poll.

PPOLL UNCONFIG `PPOLL U[NCONFIG]` or `PPU`

Prevent all devices from responding to a Parallel Poll.

READ `READ[;] [number]`

Send a copy of the specified `number` Macro buffer out to the serial host.

READ LOG `READ LOG`

Send a copy of the `LOG` buffer out to the serial host.

REMOTE REM[OTE] [addr[,addr…]]

Assert the Remote Enable line. Optionally address the specified devices to listen, placing them in the Remote State.

REQUEST REQ[UEST] [;] [status]

Generate a service request (SRQ) with the status value in the serial poll status register.

RESET RESE[T]

Warm start the Micro488/EX to default parameters. Also ABORT if SC.

RESUME RESU[ME]

Un-assert Attention. Used to allow Peripheral to Peripheral data transfers.

SAVE SAVE

Saves the user DATE, DAY, TIME and CASE formats into NVRAM. Clears NVRAM Errors.

SEND SE[ND] [;] [sub-command[sub-command…]

Send low level bus sequences. Sub-commands include: UNT, UNL, MTA, MLA, LISTEN addr[,addr…], TALK addr, ENTER, DATA'data',char, EOI'data',char and CMD'data',char.

SET DATE SET DATE[;]{option}

Used to set the date on the Micro488/EX's internal clock. Refer to the SET DATE command description for option details.

SET DAY	SET DAY[;]{option} Used to set the day of the week on the Micro488/EX's internal clock. Refer to the SET DAY command description for option details.
SET TIME	SET TIME[;]hh:mm[:ss][AM|PM] Used to set the time on the Micro488/EX's internal clock.
SPOLL	SP[OLL][addr[,addr…] Read device(s') Serial Poll response. Get internal SRQ state (CA), or Serial Poll status (*CA).
STATUS	ST[ATUS][;][number] Read the status of Micro488/EX. Used to reset error conditions.
STERM	STE[RM][;]{term[term]|NONE} Set the serial output terminator.
TERM	TE[RM][;]{term[term][EOI]|EOI|NONE} Set the serial output terminator.
TIME	TIME [DATE|DAY|DATE DAY|DAY DATE] Returns the time in the format determined by the TIME FORMAT command. Optionally returns DATE and DAY information.
TIME FORMAT	TIME FORMAT[;]{option}

Sets the format of the TIME output. Refer to the TIME FORMAT command description for option details.

TIME OUT `TI[ME OUT][;][time]`

Set the number of time seconds to wait for a bus transfer before a time out error will be declared. Checking for Time Out is suppressed by specifying 0 seconds.

TRACE `TRACE{ON|OFF}`

Enable or disable tracing during Macro buffer execution.

TRIGGER `TR[IGGER][addr[,addr…]]`

Trigger bus devices by optionally listen addressed followed by Group Execute Trigger.

WAIT `WAIT[TIME][;]hh:mm[:ss][AM|PM]`

WAIT until the time specified before continuing.

WAIT DATE `WAIT DATE [;]{option}`

WAIT until the DATE specified before continuing. Refer to the WAIT command description for an option list.

WAIT DAY `WAIT DAY [;]{option}`

WAIT until the day of the week specified before continuing. Refer to the WAIT command description for an option list.

Micro488/EX Error Messages

The following error messages are returned if an error condition exists and the `STATUS 1` or `STATUS 2` command is executed. The error condition is reset after the message is sent. Only the most recent error is maintained.

Error No.	Error Text and Description
00	OK
01	INVALID ADDRESS Caused by an invalid address outside the allowable IEEE 488 bus range of 00 to 30 for primary addresses and 00 to 31 for secondary addresses.
02	INVALID COMMAND Caused by an unrecognized command or invalid parameter.
03	WRONG MODE Caused by trying to execute a command not allowed within the present state of the interface. [eg REMOTE16 as a peripheral]
04	Unassigned - Reserved
05	Unassigned - Reserved
06	NO MACRO A `DOMACRO` or `READ` command was received but the MACRO buffer specified is empty.
07	MACRO OVERFLOW No memory is available to allocate as a MACRO buffer.
08	COMMAND OVERFLOW More than 127 characters were received and interpreted as a command.
09	ADDRESS OVERFLOW More than 15 primary address/secondary address pairs were received.

10 MESSAGE OVERFLOW
No memory is available to buffer the received data of the OUTPUT command.

11 NOT A TALKER
As the Active Controller, an un-addressed OUTPUT, a SEND DATA or a SEND CMD was received and the Micro488/EX was not in the Talk Addressed State.

12 NOT A LISTENER
As the Active Controller, an un-addressed ENTER or a SEND ENTER was received and the Micro488/EX was not in the Listen Addressed State.

13 BUS ERROR
The Micro488/EX tried to output data to the bus but there was no active listener to accept it.

14 TIMEOUT - WRITE
The specified TIME OUT time has elapsed before the last command or data byte sent by the Micro488/EX was accepted by an external device.

15 TIMEOUT - READ
The specified TIME OUT time has elapsed while the Micro488/EX was waiting for a data byte from an external device.

16 OUT OF MEMORY
The Micro488/EX was unable to perform the last command requested due to the lack of sufficient memory in the USER heap. It aslo indicates that there is no more memory available for the log buffer This error will cause the ERROR LED to flash.

17 MACRO RECURSION
A DOMACRO command specified a Macro buffer that was already executing.

18 NVRAM FAILURE
An error was detected in the NVRAM which resulted in the necessity to reinitialize memory. Any data in the LOG and MACRO buffers is lost.

19 LOGGING ERROR
A READ LOG command was encountered within a macro.

20 TIMER IN USE
The macro incremental timer was in use when another macro with a specified interval was involked.

$00 0 NUL	$10 16 DLE	$20 32 SP 00	$30 48 0 16	$40 64 @ 00	$50 80 P 16	$60 96 ` SCG	$70 112 p SCG
$01 1 SOH GTL	$11 17 DC1 LLO	$21 33 ! 01	$31 49 1 17	$41 65 A 01	$51 81 Q 17	$61 97 a SCG	$71 113 q SCG
$02 2 STX	$12 18 DC2	$22 34 " 02	$32 50 2 18	$42 66 B 02	$52 82 R 18	$62 98 b SCG	$72 114 r SCG
$03 3 ETX	$13 19 DC3	$23 35 # 03	$33 51 3 19	$43 67 C 03	$53 83 S 19	$63 99 c SCG	$73 115 s SCG
$04 4 EOT SDC	$14 20 DC4 DCL	$24 36 $ 04	$34 52 4 20	$44 68 D 04	$54 84 T 20	$64 100 d SCG	$74 116 t SCG
$05 5 ENQ PPC	$15 21 NAK PPU	$25 37 % 05	$35 53 5 21	$45 69 E 05	$55 85 U 21	$65 101 e SCG	$75 117 u SCG
$06 6 ACK	$16 22 SYN	$26 38 & 06	$36 54 6 22	$46 70 F 06	$56 86 V 22	$66 102 f SCG	$76 118 v SCG
$07 7 BEL	$17 23 ETB	$27 39 ' 07	$37 55 7 23	$47 71 G 07	$57 87 W 23	$67 103 g SCG	$77 119 w SCG
$08 8 BS GET	$18 24 CAN SPE	$28 40 (08	$38 56 8 24	$48 72 H 08	$58 88 X 24	$68 104 h SCG	$78 120 x SCG
$09 9 HT TCT	$19 25 EM SPD	$29 41) 09	$39 57 9 25	$49 73 I 09	$59 89 Y 25	$69 105 i SCG	$79 121 y SCG
$0A 10 LF	$1A 26 SUB	$2A 42 * 10	$3A 58 : 26	$4A 74 J 10	$5A 90 Z 26	$6A 106 j SCG	$7A 122 z SCG
$0B 11 VT	$1B 27 ESC	$2B 43 + 11	$3B 59 ; 27	$4B 75 K 11	$5B 91 [27	$6B 107 k SCG	$7B 123 { SCG
$0C 12 FF	$1C 28 FS	$2C 44 , 12	$3C 60 < 28	$4C 76 L 12	$5C 92 \ 28	$6C 108 l SCG	$7C 124 | SCG
$0D 13 CR	$1D 29 GS	$2D 45 - 13	$3D 61 = 29	$4D 77 M 13	$5D 93] 29	$6D 109 m SCG	$7D 125 } SCG
$0E 14 SO	$1E 30 RS	$2E 46 . 14	$3E 62 > 30	$4E 78 N 14	$5E 94 ^ 30	$6E 110 n SCG	$7E 126 ~ SCG
$0F 15 SI	$1F 31 US	$2F 47 / 15	$3F 63 ? UNL	$4F 79 O 15	$5F 95 _ UNT	$6F 111 o SCG	$7F 127 DEL SCG
ACG	UCG	LAG		TAG		SCG	

ACG = Addressed Command Group
UCG = Universal Command Group
LAG = Listen Address Group
TAG = Talk Address Group
SCG = Secondary Command Group

Sample Terminal Programs

The following program can be used to interact directly from a PC's keyboard with the Micro488/EX. Hardware handshaking is accomplished directly to the 8250 UART in the PC as Basic does not support the handshaking process. The program initalizes the Micro488/EX to…

Use only CR for the serial output terminator.
Set a time out value of 10 seconds
Disable the ID character
Disable Error Reporting
Define a special error reporting macro which re-installs itself at execution
Installs the Error Reporting macro to execute on an error.

From then on, any keys which are typed at the keyboard are printed to the PC's screen and set out the serial port to the Micro488/EX. Any data returned from the Micro488/EX is printed to the PC's screen. Errors are automatically reported and cleared.

```
100 '
110 '     Hardware Handshake Terminal
120 '
130 CLS
140 OPEN "com1:9600,n,8,2,ds,cd" AS 1
150 ' MCR = Modem Control Register
160 MCR = &H3FC : RTSON = INP(MCR) : RTSOFF = RTSON + &H2
170 'set the serial output terminator to CR only
180 PRINT#1,"sterm cr"
190 ' enable a timeout of 10 seconds
200 PRINT#1,"time out 10"
210 ' disable the ID character
220 PRINT#1,"ID;"
230 ' turn error reporting off
240 PRINT#1,"error off"
250 ' define a macro to execute on error detection
260 PRINT#1,"macro 4"
270 PRINT#1,"com '";CHR$(7);"\'"
280 PRINT#1,"status1"
290 PRINT#1,"on error domacro 4"
300 PRINT#1,"endm"
310 ' enable Macro #4 to execute on error detection
320 PRINT#1,"on error domacro 4"
330 '
```

```
340 ' Main Loop
350 '
360 WHILE NOT EOF(1)
370 K$ = INKEY$ : PRINT#1,K$; : PRINT K$;
380 PRINT INPUT$(1,1);
390 ' if greater than 100 characters in buffer then /RTS
400 IF LOC(1) > 100 THEN OUT MCR,RTSOFF
410 WEND
420 ' buffer is empty - RTS
430 OUT MCR,RTSON
440 K$ = INKEY$ : PRINT#1,K$; : PRINT K$;
450 GOTO 360
```

The following program can be used to interact directly from a PC's keyboard with the Micro488/EX. Software Xon/Xoff handshaking is accomplished directly by the program as Basic does not support the software handshaking. The program initalizes the Micro488/EX to…

Use only CR for the serial output terminator.
Set a time out value of 10 seconds
Disable the ID character
Disable Error Reporting
Define a special error reporting macro which re-installs itself at execution
Installs the Error Reporting macro to execute on an error.

From then on, any keys which are typed at the keyboard are printed to the PC's screen and set out the serial port to the Micro488/EX. Any data returned from the Micro488/EX is printed to the PC's screen. Errors are automatically reported and cleared.

```
100 '
110 '     Xon/Xoff Terminal
120 '
130 CLS
140 OPEN "com1:9600,n,8,2,ds,cd" AS 1
150 XON$ = CHR$(&H11) : XOFF$ = CHR$(&H13)
160 'set the serial output terminator to CR only
170 PRINT#1,"sterm cr"
180 ' enable a timeout of 10 seconds
190 PRINT#1,"time out 10"
200 ' disable the ID character
210 PRINT#1,"ID;"
220 ' turn error reporting off
230 PRINT#1,"error off"
240 ' define a macro to execute on error detection
250 PRINT#1,"macro 4"
260 PRINT#1,"com '";CHR$(7);"\'"
270 PRINT#1,"status1"
280 PRINT#1,"on error domacro 4"
290 PRINT#1,"endm"
```

```
300 ' enable Macro #4 to execute on error detection
310 PRINT#1,"on error domacro 4"
320 '
330 ' Xon Main Loop
340 '
350 PRINT#1,XON$;
360 WHILE NOT EOF(1)
370 K$ = INKEY$ : PRINT#1,K$; : PRINT K$;
380 PRINT INPUT$(1,1);
390 ' if greater than 100 characters in buffer then Xoff
400 IF LOC(1) > 100 THEN GOTO 440
410 WEND
420 K$ = INKEY$ : PRINT#1,K$; : PRINT K$;
430 GOTO 360
440 '
450 ' Xoff Main Loop
460 '
470 PRINT#1,XOFF$;
480 WHILE NOT EOF(1)
490 K$ = INKEY$ : PRINT#1,K$; : PRINT K$;
500 PRINT INPUT$(1,1);
510 WEND
520 GOTO 350
```