P&T-488 INTERFACE INSTRUCTION MANUAL

Custom Software Package

PICKLES & TROUT
P. O. BOX 1206, GOLETA, CA 93116, (805) 685-4641

P&T-488 INTERFACE
INSTRUCTION MANUAL

Pickles & Trout
P.O. Box 1206
Goleta, CA 93116

WARRANTY

This Pickles & Trout product is warranted against defects in materials and workmanship for 90 days from the date of shipment. Pickles & Trout will, at its option, repair or replace products which prove to be defective within the warranty period provided they are returned to Pickles & Trout. Repairs necessitated by modification, alteration or misuse of this product are not covered by this warranty.

NO OTHER WARRANTIES ARE EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE. PICKLES & TROUT IS NOT LIABLE FOR CONSEQUENTIAL DAMAGES.

FOREWORD

This manual contains the information necessary to understand and use the P&T-488 interface as well as provide instruction in the basic concepts of the IEEE-488 bus.

Those who are already familiar with the IEEE-488 bus (also known as the HP-IB, GPIB and ASCII bus) and the concepts of a Talker, Listener and Controller may skip to the chapter "Installation of the P&T-488". It is recommended that those who are not acquainted with Talkers, Listeners and Controllers read the chapter "The IEEE-488 Bus" first.

The P&T-488 interface consists of two major components: the P&T-488 interface board and the P&T-488 custom system interface software package. The software package consists of a single program named PNT488. Also included is a program named 488TEST which performs a complete functional test of the P&T-488 interface board. Additional programs are provided as examples of how one can use the P&T-488 interface to communicate with 488 devices.

Table of Contents

***** Appendices *****

- CAST OF CHARACTERS -

The 488 bus is populated by three major types of devices. One is the **Controller**, which sends commands over the bus to other devices. Another is the **Talker**, which sends data over the bus to one or more devices of the third kind: the **Listeners.** The Listeners and Talker communicate with a handshake on each data transfer, and the communication proceeds at the maximum rate allowed by the Talker and the slowest Listener. This communication is completely asynchronous and may be interrupted at specific points in the handshake cycle without causing any loss of data.

It can be useful to liken the bus to a meeting which has a chairman (Controller), a recognized speaker (Talker) and an audience (Listeners). As is true of most meetings, some of the audience is paying no attention whatever to the proceedings (some of the devices on the bus may be Idle), while some of those that are listening want to interrupt the Talker. Sometimes a member of the audience is audacious enough to indicate that it should be the chairman. The 488 bus specification allows the Controller to designate another device as his successor.

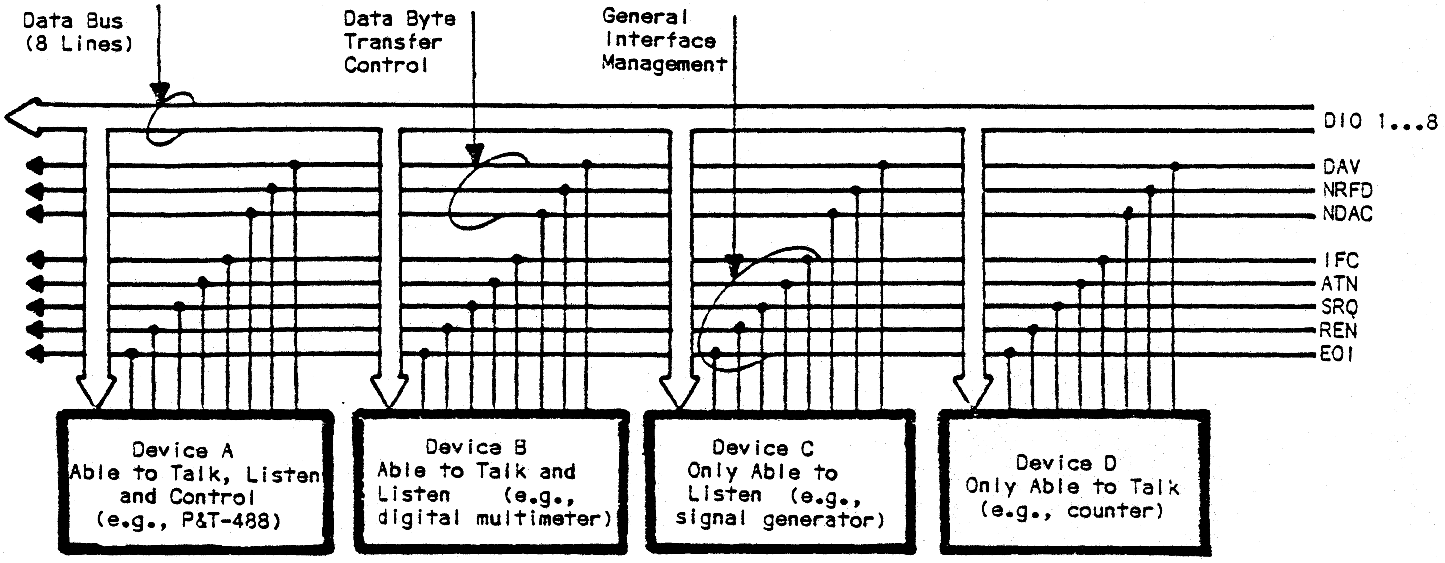
It is the **Controller's** responsibility to make sure that communication takes place in an orderly manner: it is he that says who can talk and who should listen at any given time. It is also the Controller that takes care of such matters as telling everyone to shut up (Universal Untalk **UNT** command), everyone to go back to their desks (Interface Clear **IFC**), or listen to someone trying to gain the floor (Service Request **SRQ**). Even though the Controller has (in theory) complete command over everyone else, problems can arise. One possible problem is that the Controller has made the unwise choice of telling more than one device that it can be a Talker, which results in sheer bedlam. Another way for the Controller to lose control of the situation is if a Talk Only (**ton**) device is placed on the bus. Some Talk Only devices are notoriously deaf and don't pay any attention to anybody, even the Controller!

A **Talker**, on the other hand, leads a simple life. It does not concern itself with disputes over who has the right to be heard, and when. It only puts data on the bus, waits until the slowest listener indicates it is ready for data, says the data is valid, waits until the slowest Listener says it has accepted the data, then says that it is removing the data and follows up on its threat. About the only thing that bothers a Talker is to find that no one is listening to him. Most get really upset and let the Controller know about this impolite state of affairs. Talkers that don't complain have a tendency to sit there with their mouths open, caught in mid-word. Either way, no communication is taking place and this is not considered a desirable state of affairs.

Listeners can be a little more complicated. They let the Talker know when they are ready for another word and when they have received it. Some also let the Controller know that they want some special attention. The Controller waits until the Talker can be interrupted so that no Listener is deprived of the latest bit of wisdom imparted by the Talker. Then the Controller tries to find out which device wants the attention. Two ways to do this are **Serial Poll**, in which each device is allowed to speak (one at a time) and **Parallel Poll**, which allows several devices to simultaneously inform the Controller of their need by a bit pattern each puts onto the eight data lines.

- HARDWARE OVERVIEW -

The 488 bus is made up of 16 signal lines: eight are used for data, three are needed for the interlocking handshake used to communicate the data, and the remaining five are used for bus management. Since there are eight data lines, a full eight bit byte can be communicated in each handshake cycle. This is what is meant by the phrase "**bit parallel - byte serial**" transmission. It is an alternative to the slower RS 232C standard, in which only one data line is used (and which is referred to as being a "**bit serial**" interface standard).



There are three basic concepts which are important to an understanding of how the hardware of the 488 bus works. The first is that only one of two voltages is allowed on each line, and the lower allowed voltage is ground. The second is that the 488 bus uses **negative true** logic, which means that the lower of the two voltage levels has the value TRUE, while the higher voltage has the value FALSE. The third is that the bus uses open-collector drivers. An open-collector driver can be thought of as a switch with one terminal connected to the line and the other to ground. When the driver is ON, it is as if the switch is closed, and so connects the line to ground. If the driver is OFF, it is as if the switch is open, so no connection is made between the line and ground. There is a resistor connecting the line to a voltage supply, so the voltage on the line rises to the higher of the two allowed levels if the line is not grounded. Since the 488 uses negative true logic, a line is given the value TRUE by turning the open-collector driver ON, or the value FALSE by turning the driver OFF. The phrases "**active true**" and "**passive false**" are used to describe this system; active true because the line must be actively connected to ground to impress a value of true on it, passive false because no action is needed (no connection is made) to make the value of the line false.

Each 488 device has one open-collector driver for each 488 line that it uses. More than one open-collector driver (that is, more than one 488 device) can be connected to each line. If all drivers are off the voltage on the line will be high, which means it has the value false. However, if one or more open-collector drivers are on, the line's voltage will be low, and it will have the value true. This is called a "**wire-or**" system because the logical value of the line is the logical OR of the logical values impressed on it by the several open-collector drivers connected to it. Thus each 488 device sends a true to the line by turning on its driver, or a false by turning the driver off. Note that if any device asserts a particular line true, that line will have the value true. However, if a device asserts a false (high) signal, it will be overridden by any device which asserts a true.

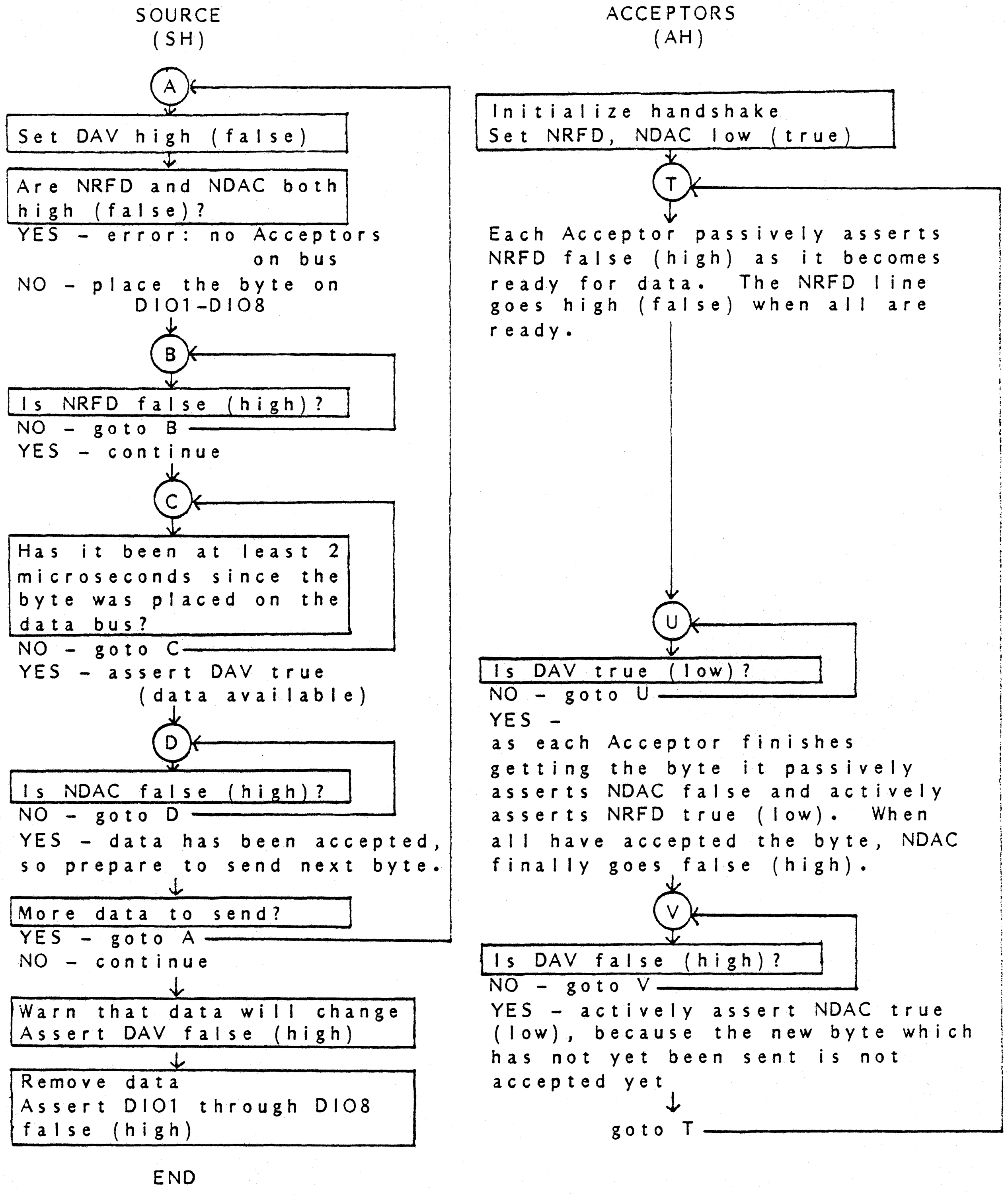
The eight data lines are named **DIO1** through **DIO8** (DIO stands for Data Input/Output). The least significant bit appears on DIO1, the most significant on DIO8. One point of possible confusion is that the data bits in an S-1ØØ system are numbered Ø through 7, while the 488 data lines are numbered 1 through 8. Another is that S-1ØØ systems assume positive true logic (high means TRUE, low means FALSE). Just remember that S-1ØØ data bit 7 appears on DIO8, etc. and a 488 byte is the one's complement of an S-1ØØ byte and everything should be all right.

The proper IEEE title for the three handshake lines is "**Data Byte Transfer Control**" lines. They are individually known as follows:
DAV (Data Valid) - when true the data on the eight data lines is valid.
NRFD (Not Ready For Data) - when true the 488 devices are not ready to accept data.
NDAC (Not Data Accepted) - when true the devices have not yet accepted the data.

The remaining five lines are known as the "**General Interface Management**" lines. They are as follows:
IFC (Interface Clear) - place all 488 devices in their default state.
ATN (Attention) - used to distinguish between a Controller and a Talker.
SRQ (Service Request) - indicates that a device needs attention.
REN (Remote Enable) - allows 488 devices to be programmed either by their local controls (front panel switches, etc.), or by information sent over the 488 bus.
EOI (End or Identify) - indicates the end of a string if ATN is false, otherwise it indicates a Parallel Poll is in progress.

- BYTE COMMUNICATION -

Byte communication requires that there be a device which is generating the byte to be communicated (the "**source**") and one or more devices which receive the byte (the "**acceptors**"). The Source and Acceptors communicate by use of an interlocking handshake using the three Data Byte Transfer Control lines (DAV, NRFD and NDAC). The byte itself is sent on the eight data lines (DIO1 through DIO8). The handshake is schematized in the following flow chart.



SOURCE
(SH)

A

Set DAV high (false)

Are NRFD and NDAC both high (false)?
YES - error: no Acceptors on bus
NO - place the byte on DIO1-DIO8

B

Is NRFD false (high)?
NO - goto B
YES - continue

C

Has it been at least 2 microseconds since the byte was placed on the data bus?
NO - goto C
YES - assert DAV true (data available)

D

Is NDAC false (high)?
NO - goto D
YES - data has been accepted, so prepare to send next byte.

More data to send?
YES - goto A
NO - continue

Warn that data will change
Assert DAV false (high)

Remove data
Assert DIO1 through DIO8 false (high)

END

ACCEPTORS
(AH)

Initialize handshake
Set NRFD, NDAC low (true)

T

Each Acceptor passively asserts NRFD false (high) as it becomes ready for data. The NRFD line goes high (false) when all are ready.

U

Is DAV true (low)?
NO - goto U
YES -
as each Acceptor finishes getting the byte it passively asserts NDAC false and actively asserts NRFD true (low). When all have accepted the byte, NDAC finally goes false (high).

V

Is DAV false (high)?
NO - goto V
YES - actively assert NDAC true (low), because the new byte which has not yet been sent is not accepted yet

goto T

- A More Detailed Look at the 488 Inhabitants -

A **TALKER** is a device which sends data over the 488 interface to other devices. There are two major types and various subtypes. One major type is the Talk Only (ton), which may be used in a 488 system which has no Controller. This device always talks, and so it must be the only device which can talk. The other major type must be told when to talk ("**addressed to talk**"). A Controller is needed because it is the only kind of 488 device that is allowed to address Talkers and Listeners. All Talkers use the Source Handshake (SH) function to send a message over the 488 bus.

A **LISTENER** is a device which receives data over the 488 interface. As with the Talker, there are two major types: Listen Only (lon) and addressed Listener. A Listen Only device always listens to the 488 bus, while an addressed Listener listens only when the Controller tells it to. The Listen Only device can operate in a 488 system which does not have a Controller since it does not need to be told what to do and when to do it. All Listeners use the Acceptor Handshake (AH) function to receive messages on the 488 bus.

A **CONTROLLER** is a device which issues commands on the 488 bus. These include commands which are used to reset all devices on the bus Interface Clear (IFC), indicate which device is to Talk (when the Controller relinquishes the bus) and which devices are to Listen (i.e. it sends the Talk and Listen addresses of those devices over the bus), perform a Poll of 488 devices (Serial Poll and Parallel Poll), and a myriad of other special functions. The commands fall into two general classifications: **Uniline** and **Multiline**. Each uniline command uses only one line out of the five General Interface Management lines. Examples of uniline messages are Remote Enable (REN), Interface Clear (IFC) and Parallel Poll. Multiline messages use the eight data (DIO1-DIO8) lines to issue the command. Examples of multiline messages include performing a Serial Poll and commanding 488 devices to Talk or Listen. Multiline messages are sent using the Source Handshake (SH) function, just like a Talker. The way that a device determines whether it is hearing a Talker or the Controller is that the ATN (Attention) line is true (low) when the Controller is issuing a message, but false (high) when a Talker is saying something. The Controller is the device which controls the ATN line. Whenever ATN is true, all addressed Talkers shut up so that the Controller can say its piece. However, some Talk Only devices don't, and so they garble commands issued by the Controller. Generally speaking, a Talk Only device should be used only in a 488 system which has no Controller. Whenever the Controller passively asserts ATN false (lets it go high), the Talker (if any) begins to send its message.

- MULTILINE COMMANDS -

Telling a 488 device to Listen is one example of a multiline command. The Controller places the Listen address of the selected device on the data lines (DIO1 through DIO8) and then performs the Source Handshake (SH) function. In other words, it "speaks" the address while ATN is true (low). Whenever the Controller is active (that is, whenever ATN is true), all devices on the 488 bus interpret whatever is said (via the data lines and the Source Handshake function) as a command rather than data. ALL devices hear what is said by the Controller. They ALL execute the Acceptor Handshake function, without regard to whether they are normally a Talker, Listener or whatever.

Another example of a multiline command is the **Serial Poll.** The order of events is that the Controller sends out the Serial Poll Enable (**SPE**) command, which tells each device that when it is addressed as a Talker that it is to say either **SBN** (Status Byte - service Not requested) or **SBA** (Status Byte - service request Acknowledged). Those are the only two messages that are allowed. Then the Controller addresses each device as a Talker in turn and Listens to the response of each. To conclude a Serial Poll, the Controller sends the Serial Poll Disable (**SPD**) command so that any device later addressed as a Talker can speak data (instead of SBN or SBA). Finally, the Controller performs whatever service is needed, which is device dependent.

- UNILINE COMMANDS -

An example of a uniline command is **Parallel Poll.** Parallel Poll is both simpler and more complicated than Serial Poll. It is simpler because only one command is given (Identify **IDY**: the logical AND of **ATN** and **EOI**) and all devices respond at once. It is possibly more complicated in that it may be more difficult to sort out which device wants service. Whenever a 488 device receives the IDY message, it immediately places its Parallel Poll Response byte on the eight data lines. For systems of eight devices or less, it is common for each device to be assigned a unique bit which it asserts true when it needs service. For example, one device would have a Parallel Poll response byte in which bit 1 is true if it needs service, otherwise bit 1 is false, and bits 2 through 8 are always false. Another device would use bit 2 to indicate its need for service and all other bits would always be false in its response byte. A third device would use bit 3. When a Parallel Poll is performed, the response sensed by the Controller will be the logical OR of all the Parallel Poll Response bytes (due to the fact that the 488 bus is a wire-or system). If the response has bits 1 and 3 true, and all other bits false, it means that the first and third devices need service, while the second does not.

If the 488 system uses more than eight devices, some alternate scheme must be used. One would be to have only eight devices respond to a Parallel Poll, and use Serial Poll on the remaining devices. Another scheme would be to have several devices share the same Parallel Poll Response byte. If the response to a Parallel Poll shows that at least one of the devices that shares a common response needs service, a Serial Poll can be used to find which ones they are.

- OVERVIEW -

The P&T-488 has four read/write registers which appear as four input/output (I/O) ports to the S-1ØØ host machine. The ports are addressed as four consecutive I/O ports with the first port address an integral multiple of 4 (Ø, 4, 8, ØC, ..., N*4, ..., FC). For ease of description these registers will be referred to as registers Ø through 3, even though what is called register Ø may be Port Ø, 4, 8, ..., N*4, ..., FC.

The addresses used by the P&T-488 are set by means of a DIP switch on the upper left corner of the interface board. All boards are set at the factory for I/O ports 7C through 7F Hex, and all software supplied by Pickles & Trout assumes these addresses. The address used by both the board and the software can be changed by the user. The addresses used by the software and the board must be the same. To change the addresses assumed by the software, refer to the instructions given with the program.

To change the addresses used by the board, first note that the labels "A7" through "A2" appear to the left of the switch. Switches A2 through A7 are set according to the following table:

Address (Hex)	A7	A6	A5	A4	A3	A2
ØØ-Ø3	ON	ON	ON	ON	ON	ON
Ø4-Ø7	ON	ON	ON	ON	ON	OFF
Ø8-ØB	ON	ON	ON	ON	OFF	ON
ØC-ØF	ON	ON	ON	ON	OFF	OFF
1Ø-13	ON	ON	ON	OFF	ON	ON
14-17	ON	ON	ON	OFF	ON	OFF
18-1B	ON	ON	ON	OFF	OFF	ON
1C-1F	ON	ON	ON	OFF	OFF	OFF
2Ø-23	ON	ON	OFF	ON	ON	ON
24-27	ON	ON	OFF	ON	ON	OFF
28-2B	ON	ON	OFF	ON	OFF	ON
2C-2F	ON	ON	OFF	ON	OFF	OFF
3Ø-33	ON	ON	OFF	OFF	ON	ON
34-37	ON	ON	OFF	OFF	ON	OFF
38-3B	ON	ON	OFF	OFF	OFF	ON
3C-3F	ON	ON	OFF	OFF	OFF	OFF
4Ø-43	ON	OFF	ON	ON	ON	ON
44-47	ON	OFF	ON	ON	ON	OFF
48-4B	ON	OFF	ON	ON	OFF	ON
4C-4F	ON	OFF	ON	ON	OFF	OFF
5Ø-53	ON	OFF	ON	OFF	ON	ON
54-57	ON	OFF	ON	OFF	ON	OFF
58-5B	ON	OFF	ON	OFF	OFF	ON
5C-5F	ON	OFF	ON	OFF	OFF	OFF
6Ø-63	ON	OFF	OFF	ON	ON	ON
64-67	ON	OFF	OFF	ON	ON	OFF
68-6B	ON	OFF	OFF	ON	OFF	ON

Address (Hex)	A7	A6	A5	A4	A3	A2
6C-6F	ON	OFF	OFF	ON	OFF	OFF
7Ø-73	ON	OFF	OFF	OFF	ON	ON
74-77	ON	OFF	OFF	OFF	ON	OFF
78-7B	ON	OFF	OFF	OFF	OFF	ON
7C-7F	ON	OFF	OFF	OFF	OFF	OFF
8Ø-83	OFF	ON	ON	ON	ON	ON
84-87	OFF	ON	ON	ON	ON	OFF
88-8B	OFF	ON	ON	ON	OFF	ON
8C-8F	OFF	ON	ON	ON	OFF	OFF
9Ø-93	OFF	ON	ON	OFF	ON	ON
94-97	OFF	ON	ON	OFF	ON	OFF
98-9B	OFF	ON	ON	OFF	OFF	ON
9C-9F	OFF	ON	ON	OFF	OFF	OFF
AØ-A3	OFF	ON	OFF	ON	ON	ON
A4-A7	OFF	ON	OFF	ON	ON	OFF
A8-AB	OFF	ON	OFF	ON	OFF	ON
AC-AF	OFF	ON	OFF	ON	OFF	OFF
BØ-B3	OFF	ON	OFF	OFF	ON	ON
B4-B7	OFF	ON	OFF	OFF	ON	OFF
B8-BB	OFF	ON	OFF	OFF	OFF	ON
BC-BF	OFF	ON	OFF	OFF	OFF	OFF
CØ-C3	OFF	OFF	ON	ON	ON	ON
C4-C7	OFF	OFF	ON	ON	ON	OFF
C8-CB	OFF	OFF	ON	ON	OFF	ON
CC-CF	OFF	OFF	ON	ON	OFF	OFF
DØ-D3	OFF	OFF	ON	OFF	ON	ON
D4-D7	OFF	OFF	ON	OFF	ON	OFF
D8-DB	OFF	OFF	ON	OFF	OFF	ON
DC-DF	OFF	OFF	ON	OFF	OFF	OFF
EØ-E3	OFF	OFF	OFF	ON	ON	ON
E4-E7	OFF	OFF	OFF	ON	ON	OFF
E8-EB	OFF	OFF	OFF	ON	OFF	ON
EC-EF	OFF	OFF	OFF	ON	OFF	OFF
FØ-F3	OFF	OFF	OFF	OFF	ON	ON
F4-F7	OFF	OFF	OFF	OFF	ON	OFF
F8-FB	OFF	OFF	OFF	OFF	OFF	ON
FC-FF	OFF	OFF	OFF	OFF	OFF	OFF

For example, to address the P&T-488 interface board to use I/O ports 7C through 7F Hex, A7 must be ON and A2 through A6 OFF.

The P&T-488 allows direct access to the 8 signal lines of the IEEE 488-1978 (hereafter called 488) data bus (Register 2) and the 8 lines of the 488 Data Byte Transfer Control Bus and General Interface Management Bus (Register 1). In addition, a register is provided to allow a software settable response to a Parallel Poll (Register 3). Finally, a register is provided which indicates transitions occurring on the various 488 Control Bus and Management Bus lines (Register Ø). Additional features of the P&T-488 include software disable of interrupts from the P&T-488 (without having to disable all interrupts of the S-1ØØ system) and immediate response of the interface to Attention (ATN), Interface Clear (IFC) and Parallel Poll without intervention of the S-1ØØ system's CPU.

The data transfer rate is highly dependent on the software, CPU and system memory of the S-1ØØ system, but with the supplied software, an 8Ø8Ø running at 2.Ø MHz and no memory wait states, the transfer rate is about 3 KBytes/sec. For applications requiring higher rates, the same S-1ØØ system can get data rates of over 9 KBytes/sec in the Talk Only mode.

REGISTER FUNCTIONS

No.	FUNCTION
Ø	Interrupt Status (read only)
Ø	Interrupt Reset (write only)
1	Command Line Register (read and write)
2	Data Line Register (read and write)
3	Parallel Poll Response (write only)

REGISTER BIT MAP

No.	I/O	D7	D6	D5	D4	D3	D2	D1	DØ
Ø	IN	DAV	NRFD	NDAC	XIFC	XATN	SRQ	REN	POC
		+-	+	+	-	+-	-	+	-
Ø	OUT	DAV	NRFD	NDAC	XIFC	XATN	SRQ	TALK/ LISTN	DI/ EI
1	I/O	DAV	NRFD	NDAC	IFC	ATN	SRQ	REN	EOI
2	I/O	DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1
3	OUT	DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1

NOTES:

+ means the bit goes low on a LOW to HIGH transition
- means the bit goes low on a HIGH to LOW transition

DI means 488 interface interrupts are disabled
EI means 488 interface interrupts are enabled

The 488 data lines are numbered from 1 to 8, while the data lines on the S-1ØØ system are numbered Ø to 7

X as in XATN, XIFC signifies that some device other than the P&T-488 has made the level on the line (ATN or IFC) active true (low).

- REGISTER 3 -

This register holds the Parallel Poll Response byte. Whatever has been output to Register 3 will appear on the 488 data lines in response to a Parallel Poll (ATN and EOI).

- REGISTER 2 -

This register is connected to the 488 data lines through bus transceivers. The state of the data lines can be sensed by reading Register 2, and the P&T-488 will assert on the data lines whatever was last written into Register 2. However, if either the XATN flag or XIFC flag in Register Ø is set, the output buffers to the 488 bus are disabled which precludes assertion of what was last written into Register 2. Remember that the 488 bus uses negative logic so that any bit that is low is asserted (or logically true). Also the 488 bus is a wire-or system, so if any piece of equipment is asserting a particular line true, that line will be a logical true. But if a device asserts a false (high) signal, it is overridden by any device that asserts a true. Hence the terminology of **active true** and **passive false.** Thus if the P&T-488 is being used as a Listener all bits of Register 2 should be written high (logic false) so that the data asserted by the Talker can be properly read.

- REGISTER 1 -

This register allows direct setting and sensing of the 488 Control and Management bus lines. If the XIFC flag is set in Register Ø, the interface will not assert any of the lines, regardless of what was last written into Register 1. Similarly, if XATN flag is set in Register Ø, the interface will not assert any line except Not Ready For Data (NRFD) and Service Request (SRQ). SRQ will be asserted active true (low) only if the SRQ bit (bit D2) of Register 1 was written low. NRFD will always be asserted active true (low). The reason that NRFD is asserted true is so that the System Controller will not send any commands until the S-1ØØ CPU is ready to accept them. Note that XATN has precedence over XIFC, so an externally applied IFC followed by an externally applied ATN will cause NRFD to be active true, SRQ to be true if the SRQ bit in Register 1 was written low, and all other 488 lines will be passive false.

- REGISTER Ø -

This is the Interrupt Status/Reset Register. Since the P&T-488 uses only one interrupt vector, one needs to be able to determine which condition caused the interrupt. Each bit of this register is associated with an interrupt-causing condition. By writing a low in the corresponding bits, one can individually reset the status bits associated with Data Valid (DAV), Not Ready For Data (NRFD), Not Data Accepted (NDAC), External Interface Clear (XIFC), External Attention (XATN) and Service Request (SRQ). If Bit 1 is set low status bit 7 will ignore any activity on the DAV line. This is useful when the interface is used as a Talker or Controller. If Bit 1 is set high, Bits 5 and 6 will ignore any activity on the NDAC and NRFD lines, which is useful when the interface is used as a Listener. If Bit Ø is set low, status Bits Ø (POC/RESET) and 1 (REN) will be cleared and the P&T-488 will be prevented from interrupting the S-1ØØ system (but the interrupt status bits will continue to respond to 488 Control and Management line activity). If Bit Ø is set high the interface can interrupt the S-1ØØ system.

If Bit 4 (IFC) of Register 1 is asserted there is no way of determining if an external Controller is also asserting IFC, so interrupt status bit 4 (XIFC) will ignore any activity due to an external Controller. A similar argument is true for ATN and XATN (Bit 3 of Registers 1 and Ø). This is not a problem because the IEEE standard allows only the System Controller to assert IFC, and only the Controller-in-Charge may assert ATN. The standard further specifies that there may be no more than one System Controller and no more than one Controller-in-Charge.

P&T-488 Functional Test

The program 488TST81 performs seven different kinds of tests on the P&T-488 interface board and its 488 cable. The first group of four are done with no 488 device or test plug connected to the P&T-488. The last three are made with the special test plug connected to the P&T-488.

The program starts by printing a message to the operator to disconnect all 488 devices from the P&T-488. The operator signifies this has been done by pressing any key on the keyboard. After a key has been pressed the program begins its tests.

NOTE: Any time a Control C is pressed, the program is aborted and control is returned to the monitor (operating system).

The first test checks the data register (Register 2) by outputting a byte to the 488 data lines then reading the data lines to see if their state corresponds to the byte output to them. Each of the 256 possible bytes is tried in turn. If any errors occur, a message "**DATA ERROR - bits in error are ...**" with the bit names is printed. If there are no errors, no message is printed.

In a similar manner, the second test checks the command line register (Register 1). If there are any errors, the message "**COMMAND LINE ERROR - bits in error are ...**" is printed. Again, if there is no error, no message is printed.

The third test checks the Parallel Poll Response register (Register 3) by first making ATN and EOI true. Thus anything output to the Parallel Poll Response Register should appear on the 488 data lines. If the Command Line test failed with bits Ø and/or 3 in error, the results of this third test are meaningless. As with the first two tests, each of the 256 possible byte values is tried and any errors are reported: this time the error message is "**PARALLEL POLL ERROR - bits in error are ...**".

The fourth test checks the Interrupt Service Register (Register Ø). If the second test failed, this one will probably fail also. Errors are reported with the message "**INTERRUPT SERVICE REGISTER ERROR - bits in error are ...**".

After these four tests have been made, (they take less than a tenth of a second), the operator is told to attach the special test plug and then press any key on the keyboard to continue the tests. The plug connects the eight data lines to the eight 488 command lines, so that the 488 cable can be tested for continuity, shorts or incorrect wiring. It also allows testing the response of the P&T-488 board to ATN and IFC asserted true by an external Controller.

The fifth test checks the 488 cable and reports any bits in error. If either the first (data line) or second (command line) tests failed, the results of this test will be meaningless. If the first four tests were passed without error, but this one shows errors, it means either the cable and/or test plug is open, shorted, miswired or improperly plugged. If all bits are in error, the 488 cable is either not connected to the P&T-488 interface board or the special test plug is not plugged into the cable.

The sixth test checks the response of the P&T-488 to an IFC (Interface Clear) presented by an external Controller. What is really done, of course, is to use the data port to assert a true on the IFC line through the special shorting plug, but the P&T-488 can't tell the difference between this and an external Controller making IFC true. The results are meaningful only if the first five tests passed with no errors.

The seventh test checks the response of the P&T-488 to an ATN (Attention) presented by an external Controller. The technique is the same as used in the sixth test. Again, the results are meaningful only if the first five tests were passed without any errors.

After the seventh test has been completed, the message NO ERRORS is printed if all tests were passed without error. Then the message "**P&T 488 functional test complete**" is printed and the program jumps back to the monitor.

WHAT TO DO IN CASE OF ERROR -

If any of the first four tests fail, check the following:

1. The P&T-488 interface board must be addressed to the same ports that the test routine tests. The base address (lowest address of the four) used by the P&T-488 must be in location 1Ø3 Hex for CP/M systems, 3ØØ3 Hex for North Star. The program is supplied with the base address set to 7C Hex.

2. All 488 devices must be disconnected from the P&T-488.

3. Make sure you are using the correct test routine. 488TST81 is to be used on **ONLY** Revision 81A boards (serial number 5ØØØ and up). 488TEST is to be used on **ONLY** boards with serial numbers under 5ØØØ.

If any of the first four tests fail, try disconnecting the 488 cable from the P&T-488 interface board. If they STILL fail, the P&T-488 is faulty and should be returned to Pickles & Trout for repair or replacement. Be sure to include a printout of the test results. If the first four tests are passed without error after the cable has been disconnected, the cable is defective (a short between lines or a short to ground).

If no error message is printed before the "**Attach test plug...**" message to the operator, the first four tests were passed without error. If the error message "EXTERNAL ATN ERROR - **bits in error are 2**" is displayed, it is likely that you are using the wrong test routine. 488TEST is to be used on only boards with serial numbers under 5ØØØ; 488TST81 is to be used only on boards with serial numbers over 4999. USE **THE CORRECT TEST.** If the error message "**EXTERNAL INTERFACE CLEAR ERROR - ...**" is printed with no error message preceding it, the P&T-488 is faulty. If the error message "**EXTERNAL ATN ERROR - ...**" is printed, and either there is no other error message or only the EXTERNAL INTERFACE CLEAR ERROR message, the P&T-488 is faulty and should be returned for repair or replacement.

RETURN POLICY -

The P&T-488 interface board, its 488 connecting cable and the special test plug are warranted to be free of defects in materials and workmanship for 90 days from the date of sale. If they should be found faulty within the warranty period, Pickles & Trout will (at its option) repair or replace them upon receipt of the defective pieces. Repairs necessitated by alteration, modification or misuse of these products are not covered by this warranty. Out of warranty interface boards which have not been modified or otherwise tampered with will be repaired or replaced for a flat fee. As of January, 1981, the fee is $45.00.

NOTICE - A handling fee of $45.00 will be charged for any board that is returned for repair because the wrong test routine was used. THIS INCLUDES BOARDS STILL IN WARRANTY.

When returning equipment to Pickles & Trout, be sure to include the following information:

1 NAME and ADDRESS of the owner.

2 NAME and PHONE NUMBER of the person who is using the P&T-488.

3 Description of the failure and how it was found. PRINTOUT OF THE TEST RESULTS IS REQUIRED.

4 Description of the S-100 machine and operating system. Include manufacturer and model name of the CPU board, system clock rate, and the name of the organization that authored the operating system, as well as any information on systemic modifications made to it.

For example: IMSAI 8080 with Ithaca Audio Z-80 CPU board with a system clock of 4 MHz, North Star single density 5.25" floppy disk drive and controller, Digital Research CP/M as modified by Lifeboat Associates for North Star disks.

5 If the equipment is still in warranty, enclose a copy of the bill of sale. Otherwise enclose a check for the repair and shipping and handling fees. The shipping and handling fee is $5.00 for addresses within the contiguous US, $7.50 for Alaska and Hawaii. There is no shipping fee for foreign addresses because the equipment will be returned freight collect.

The repairs/replacements will be made within five business days and the equipment returned postage paid to US addresses, freight collect to foreign addreses.

INSTALLATION of the P&T-488

The P&T-488 interface card uses four contiguous I/O ports, and is supplied configured to use ports 7C through 7F Hex. Be sure there is no port address conflict with other I/O boards in your S-100 system BEFORE installing the P&T-488. If it is necessary to change the I/O ports that the P&T-488 uses, refer to the chapter entitled "Hardware Description" for instructions.

When you are satisfied that there is no I/O port address conflict between the P&T-488 interface and other devices in your S-100 system, turn off the power to the S-100 system and wait at least twenty seconds (to allow sufficient time for the S-100 power supply to discharge) before installing the P&T-488 card. Attach the cable to the back panel of the S-100 system using the metric hardware supplied with the cable (this hardware mates with the standard lockscrews used on 488 cables supplied by Hewlett-Packard, Beldon and others), and plug the cable onto the top edge connector of the P&T-488 interface card. Note that the plug and edge connector are keyed.

If the I/O port addresses of the board have been changed from 7C through 7F Hex, it will be necessary to modify 488TEST and PNT488. The fourth byte in the program 488TEST is supposed to contain the lowest address of the four that is used by the P&T-488 interface card. If, for example, the card has been addressed to use ports 60 through 63 Hex, you should change the value in the location BASPRT (103 Hex of 488TEST) to 60 Hex.

The programs 488TEST and PNT488 should now be loaded so they can be modified (if necessary) and run. Programs supplied on cassette tape are recorded in Kansas City format and may be read by the BITWIGGLER™ (see Appendix C for the source listing) or any other cassette interface which understands the Kansas City format.

Next the P&T-488 should be tested for proper operation. Make any necessary modifications to 488TEST (see Appendix B for details) and then run the modified program. Refer to the chapter "Functional Test" for information about the meanings of the various messages.

After the test has been completed with no errors, you are ready to use the 488 interface. You will have to write a set of short routines to complete the integration of the P&T-488 with your particular system. The chapters "Custom Package Routines" and "User-Supplied Routines" define the purpose of each of the routines, and the chapters "488 Bus Monitor" and "Sample Program" each give examples of how the routines can be written and used.

PAGE	ROUTINE NAME	FUNCTION
CS-7	CNTRL	Performs the Controller function (sends commands)
CS-8	GIM	Allows direct control of the General Interface Management lines
CS-8	INIT	Clears the interface, leaves all lines passive false
CS-8	LISTN	Performs the Listen-Only function
CS-9	PPIDL	Puts the Parallel Poll function in the Idle state
CS-9	PPQRY	Performs a Parallel poll
CS-9	PISTT	Sets the "ist" (individual status) message true
CS-10	PISTF	Sets the "ist" message false
CS-11	SPIDL	Puts the Service Request function in the Idle state
CS-11	SPQRY	Serial Poll query routine (performs a Serial Poll)
CS-12	SPSRQ	Service Request routine
CS-12	STADR	Sets the talker, listener addresses
CS-13	STATE	Passes information on the state of the interface to the user
CS-15	TALK	Performs the Talk-Only function
CS-15	XCTRL	Respond to an External Controller

DESIGN PHILOSOPHY

This software package was written with several objectives in mind. The first is that the routines should relieve the user of as much of the burden of dealing with the 488 bus protocol as possible. In place of having to test and respond to the signals on the bus the user need only set up a buffer (when appropriate) for the commands or data to be sent or received and then call a routine. The second is that **ALL** commands actually appear on the bus: there is nothing more frustrating than trying to debug a system in which a "smart" controller sees that it is going to address itself as a Talker, and then does so without putting the Talk address on the bus. The third consideration is that the design be closely identifiable with the state-space representation of bus functions. The memory locations TSTAT, LSTAT, etc. hold the present state of the interface functions. The fourth consideration is that the code for the interface routines be "pure" so that it can be put into ROM (Read Only Memory).

The routines supplied with the P&T-488 interface board allow it to act as a Controller, Talker or Listener, and provide the additional ability of conveniently handling commonly encountered situations. These include requesting service, either by means of the SRQ (Service Request) function or the PP (Parallel Poll) function, ceasing to request service, performing a Parallel Poll, performing a Serial Poll and responding to an external Controller (i.e., a Controller that is not the P&T-488 itself).

The P&T-488 interface depends heavily on the support software in order to communicate on the 488 bus. For this reason it is necessary for the S-100 system to execute P&T-488 routines in order to perform 488 bus functions. This includes not only the "assertive" functions, such as Talk and Control, but also the "responsive" functions, which include responding to a Serial Poll, being addressed as a Talker or a Listener by the Controller, etc. The only 488 bus function which the P&T-488 interface board can complete without any software intervention is respond to a Parallel Poll.

Communication between the S-100 system and the P&T-488 takes place by means of jump tables, state tables and string buffers. The user accesses routines within the P&T-488 software package by means of a jump table that resides within it. The user supplies several routines which are used by the P&T-488: these routines are accessed by means of a jump table which the user also supplies. The jump table within the P&T-488 interface software package is near the beginning and starts at memory location ENTBL. The user is expected to use it and it only as the means of calling the various P&T-488 routines. The reason the jump table should be used instead of going directly to the P&T-488 routine is that later versions of the interface software may change the location of the routine, while the placement of the jump to that routine in the jump table WILL NOT change. Thus if the user uses only the jump table, he can use subsequent versions of the interface software without changing his software in any way.

P&T-488 Ver. 1.4 Jump Table
Organization

Routine	Entry Point
INIT	ENTBL
TALK	ENTBL+3
LISTN	ENTBL+6
STADR	ENTBL+9
CNTRL	ENTBL+12 (decimal)
GIM	ENTBL+15
STATE	ENTBL+18
XCTRL	ENTBL+21
SPQRY	ENTBL+24
SPSRQ	ENTBL+27
SPIDL	ENTBL+30
PPQRY	ENTBL+33
PISTT	ENTBL+36
PISTF	ENTBL+39
PPIDL	ENTBL+42

The P&T-488 interface software needs several user supplied routines in order to complete the integration into his system. It is expected that the user will provide a jump table which points to these routines. The details of the jump table and the operation of the routines appears in the section **User-Supplied Routines.**

Many of the P&T-488 interface routines cause the 488 interface functions to change state. The routine STATE allows the user to quickly determine the state of the more commonly desired interface functions. If the user needs additional detailed information about the states of the various interface functions he may look at the state table which is stored in memory starting at location TSTAT.

The P&T-488 routines which allow the S-100 system to be the 488 bus Controller or a Talker require strings which are stored in output string buffers. The user informs the P&T-488 routines of the location of the buffer by setting the register pair HL to the beginning address of the string and DE to the address of the end of the string before calling the P&T-488 routine. This technique allows the user flexibility in the definition of the strings and their length. For those strings which are needed on a recurring basis, the user may just point to that string rather than copying it into an intermediate buffer before calling the P&T-488 routine.

One other P&T-488 interface function may require a string buffer. That function is the 488 Listen routine. The conditions under which it needs a buffer are detailed in the description of the routine LISTN. If a buffer is needed, the location of that buffer is passed to the routine by the HL and DE register pairs, just as was done for the Talk and Control functions.

Single and Double Byte Addresses
And How the P&T-488 Uses Them

The IEEE-488 standard defines two general ways of addressing Talkers and Listeners. One way is by a single byte, and is called "single byte address" or "non-extended address". In terms of function mnemonics, the Talker function is known as the T Interface function, and the Listener as the L Interface function. The other method of addressing is known as "extended address" or "two byte address". The corresponding function mnemonics are TE and LE for Extended Talker and Extended Listener Interface functions, respectively. The P&T-488 and this software package are set up so that the P&T-488 may be addressed either way. If the Controller sends the primary Listen address of the P&T-488 and follows it with a secondary address, the secondary address is stored in the memory location LSTNS. If the primary address was not followed by a secondary address, a dummy secondary address of 7F Hex (which is an illegal secondary address) is stored in that location. The memory location TALKS is used in a similar manner to record the secondary address (or lack thereof) sent by the Controller after the P&T-488's primary Talk address. The user can make use of the optional secondary address for many different purposes. One example of a use of multiple secondary addresses is the following: Assume that the S-100 system is monitoring activity of the 488 bus and printing the results on its printer. Assume also that there are several different print formats possible and that the user wants the 488 Controller to be able to specify which format is to be used. One way of accomplishing this goal is to assign two different Listen addresses to the P&T-488: one for passing formatting information and the other for passing characters to be printed. The two addresses must have the same primary address and so differ only in the secondary address. Assume that the P&T-488 has been assigned the primary Listen address of ! (21 Hex), and the secondary address for formatting information is b (62 Hex), while that for data to be printed is a (61 Hex). Whenever the S-100 system calls the Listen function it first looks at the memory location LSTNS to see what the secondary Listen address is. If it finds the character b, it interprets the string that is heard as formatting information. If it finds the character a, it prints the string, for it is data. And if it finds any other character it means that neither of these functions has been called for.

This brings up a point that should be made about good practice concerning configuration of the IEEE-488 bus. It is generally a good rule to assign a given primary Listen or Talk address to only one 488 device. This way if an address gets garbled (the wrong secondary address sent with the proper primary address), it becomes obvious that there is an error.

Serial Poll and Service Request

Overview

The two functions Serial Poll and Service Request are closely intertwined. Basically, the Service Request function is used by a 488 device to tell the Controller that it needs some special attention. The Serial Poll function is used by the Controller to determine which one of the devices attached to the 488 bus is calling for help.

All 488 devices which have the Service Request function share the single 488 line known as SRQ. Any one which needs special attention asserts an active true (connects the line to ground). It can be seen that the SRQ line is false (high) only when all the devices do not need service. Since several devices share the one line, the Controller must find which device(s) need attention before it can service it (them). This is done by performing a Serial Poll, which consists of first informing all devices that a Serial Poll is going to begin (the Controller sends the Serial Poll Enable message), addressing each device as a Talker one at a time, and listening to its response. The response byte has a true (low) value on line DIO7 if that device is requesting service, and that device also asserts a passive false (high) on the SRQ line as it sends the response byte. If the device is not requesting service, line DIO7 is false (high).

When the Controller has finished the Serial Poll, it informs all devices that the function is finished by sending the SPD (Serial Poll Disable) message. This is done so that any device which is subsequently addressed as a Talker will speak normal data instead of the Serial Poll response byte.

Summary of Functions

IEEE-488 Functions Implemented

The IEEE-488 standard assigns mnemonics to the allowed subsets of each interface function, so a 488 device can be tersely but fully described by just a few words. The following table indicates what interface functions are implemented by the P&T-488 and Ver 1.4 software, and includes a brief description of the meaning of the mnemonics used.

AH1
Complete Acceptor Handshake capability

SH1
Complete Source Handshake capability

T5
The device can operate as a Basic Talker, respond to a Serial Poll, be placed into a Talk Only mode of operation, and will unaddress itself as a Talker if the Controller sends its Listen Address. This last operation means that the device will cease being an addressed Talker when the Controller commands it to be a Listener.

TE5
The device can operate as a Basic Extended Talker, respond to a Serial Poll, be placed into a Talk Only mode of operation, and will unaddress itself as a Talker if the Controller sends its Listen Address. This last operation means that the device will cease being an addressed Talker when the Controller commands it to be a Listener.

L3
The device can operate as a Basic Listener, can be placed into a Listen Only mode of operation, and will unaddress itself as a Listener if the Controller sends its Talk Address. This last operation means that the device will cease being an addressed Listener when the Controller commands it to be a Talker.

LE3
The device can operate as a Basic Extended Listener, can be placed into a Listen Only mode of operation, and will unaddress itself as a Listener if the Controller sends its Talk Address. This last operation means that the device will cease being an addressed Listener when the Controller commands it to be a Talker.

SR1
The device has complete Service Request capability.

RL0
The device has no Remote-Local function capability.

PP1
The device has complete Parallel Poll response capability. This means that the Parallel Poll function can be configured by the Controller (which in turn means that the Controller can assign a specific Parallel Poll response message to the device).

PP2
The device is not capable of being configured (assigned a Parallel Poll response message) by the Controller. The response is assigned by the local message lpe, which in this case is done by the S-100 system.

The user should note that the PP1 and PP2 functions are mutually exclusive. The P&T-488 and its associated software package have been constructed so that the user could pick whichever function is most suited to his needs. But for proper operation of the 488 bus, it is imperative that he use only one of the two functions in any particular bus configuration.

DC1
The device has complete Device Clear capability.

DT1
The device has complete Device Trigger capability.

C1
The device can operate as the System Controller.

C2
The device can send IFC and take charge of the 488 bus.

C3
The device can send the REN (Remote Enable) message.

C4
The device can respond to the SRQ (Service Request) message.

C25
The device can send IF messages (e.g., Listen and Talk addresses, etc.), can perform a Parallel Poll and can Take Control Synchronously. However, the device can not pass or receive control to or from another Controller.

The user should be aware of the fact that these are capabilities **offered** by the P&T-488 and that he does not have to use **all** of them. Indeed, some are mutually contradictory so he must not use both. The mutually exclusive capabilities offered are the T5/TE5 pair, the L3/LE3 pair and the PP1/PP2 pair. It is the user's obligation to pick at most only one function capability out of each of these pairs. It is allowable for the user to pick neither, but it is **not** allowable for the user to pick both.

CNTRL

Become the Controller

This routine is used to perform the various Controller functions, such as addressing Listeners, Talkers, sending Remote Enable, etc. It is important that this routine be called only when the user is sure that the DAV line is passive high, (i.e., take Control synchronously); otherwise there is the possibility of the current Talker being interrupted by the Controller while it is in the middle of transferring a byte of data. This could result in a spurious command being sent over the 488 bus and may destroy the data byte as well. In those cases where the P&T-488 is not participating in data transfer on the 488 bus but it is necessary for it to become the Controller from time to time, one can use the non-buffered Listener function provided by the routine LISTN to insure that the P&T-488 will take control synchronously. Note that the routines TALK and LISTN either return to the user's calling routine or call his routine BREAK at a point in the handshake cycle where a call to CNTRL will result in a synchronous assumption of the Controller function by the P&T-488.

The register pair HL must contain the address of the first character of the command string to be sent, DE contains the address of the last character of the string, and BC contains the address of the beginning of the user-supplied jump table. CNTRL calls the user routine BREAK after each character in the string has been sent (this allows the user to interrupt or defer further commands while other devices on the S-100 system are being serviced). If a Service Request (SRQ) is detected from some 488 device, a call is made to the user-supplied routine SVCRQ.

When CNTRL has finished sending the string of commands, it returns to the user's calling routine with the address of the last character sent in register pair HL, and the 488 lines ATN and DAV are left passive false. (Thus the P&T-488 has relinquished control of the bus.) If the P&T-488 has been selected as a Listener or is to perform Listen Handshake, the 488 line NRFD is left active true. This prevents the Talker from saying anything until the S-100 system has started execution of the routine LISTN. Finally, the Controller is left in STANDBY (CSBS in IEEE 488 notation). Thus the P&T-488 is assumed by the other programs to be the Controller-In-Charge until CSTAT (a memory location) is set to the Controller Idle State (CIDS) either directly by the user, or by the user executing the routine INIT.

GIM
General Interface Management

This routine allows the user to directly control several of the General Interface Management lines. A call to GIM is made with the appropriate bit pattern in the A register.

D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	IFC	X	SRQ	REN	EOI

If a bit is high (positive logic 1), the corresponding line is made active true. Those bits marked by an X are disregarded. For example, if it is desired to make EOI active true, and IFC, SRQ and REN passive false, one would call GIM with 01 Hex in the A register. (Because of the disregarded bits, the A register could contain 09 Hex, 21 Hex, etc. without changing the result.) GIM returns to the calling routine with all registers restored except the accumulator and flags.

INIT
Initialize Interface

A call to INIT clears the P&T-488 by setting all data and control lines passive false, sets the Parallel Poll Response to all lines passive false, and sets all functions (Talker, Controller, Listener, etc) to their idle states. If the B register is zero when INIT is called, an IFC (Interface Clear) pulse is also sent on the 488 bus to initialize all devices to a known state. Note that only the Controller is allowed to send the IFC message, so the user should set register B non-zero if the P&T-488 is not the Controller.

LISTN
Listen-Only

This routine performs the Listen function, which allows another device on the 488 bus to send information to the S-100 computer. The information can be in any byte-oriented form: it may be ASCII characters with or without parity, it may be BCD values, binary values, etc.

The accumulator (A register) determines which of four modes is selected: if Bit 0 of A is 0 no buffer is used and the user must get the byte of data by looking at the A register each time BREAK is called. If the Bit 0 is 1 when LISTN is called, the data is put into a buffer as well as appearing in the A register each time BREAK is called. Bit 1 of the A register determines whether the Listen function will terminate on a End Of String (EOS) byte. If Bit 1 is 1, then an EOS will cause LISTN to return to the calling program. The routine BREAK is called as each byte of data is received, which allows the user to interrupt or defer further 488 transactions while he performs some other operation, or allows him to check each byte for special information.

The register pair HL must contain the address of the beginning of the listen buffer, and DE contain the address of the end of the buffer. Note that HL and DE need to be defined only if a buffer is used. The register pair BC contains the address of the beginning of the user-supplied jump table.

A jump is made to the user-supplied routine BUFUL when the buffer is filled, so the user can then transfer or otherwise manipulate the data and clear the buffer.

When the buffer is emptied, a call to LISTN will continue the transfer of data. LISTN returns to the calling routine when it senses EOI (End Or Identify) true.

The SRQ (Service Request) line is tested before each byte is received, and if it is active true, the routine determines whether the P&T-488 is the Controller-In-Charge. If it is, then a call is made to the user-supplied routine SVCRQ. After the user has serviced the Service Request, he need only execute a RETurn to continue listening from where LISTN left off.

This routine implements the Listen Only (lon) function described in the IEEE-488 standard. Thus execution of this routine sets the Listen State byte to Listener Addressed. Execution of this routine also resets the Talk State byte to the Idle (TIDS) State.

If the user wishes instead to implement the Addressed Listen function described in the IEEE-488 standard (i.e., the transition from LIDS to LADS should occur only if the Controller has addressed the P&T-488 as a Listener), he should call the routine STATE and then call LISTN only if the Listen State byte shows that the P&T-488 is addressed to Listen.

The non-buffered Listen function can be used for those cases where the P&T-488 is not the Talker or Listener but is expected to assume Control from time to time. The technique is to use the LISTN routine but ignore the data. Each time BREAK is called is a time that the P&T-488 can assume Controller status without garbling a data byte. So each time BREAK is called the S-100 system determines whether it needs to become the 488 Controller: if so, it does so then, but if not it merely RETurns to the calling routine. Note that the routine BREAK is called AFTER each data byte has been communicated; this technique will lock up the S-100 system until the Talker says something. If it turns out that there is no Talker or the Talker never speaks, there is no way for the S-100 system to regain control.

PPIDL
Parallel Poll Idle

This routine puts the Parallel Poll response function in the Idle state. Thus, whenever the Controller performs a Parallel Poll, the P&T-488 will give a non-affirmative response, regardless of the state of the "ist" (individual status) message and the Sense bit of the most recent PPE (Parallel Poll Enable) message received by the P&T-488.

PPQRY
Parallel Poll

This routine causes the P&T-488 to conduct a Parallel Poll. The response to the Parallel Poll is returned in the accumulator and also in the memory location LBYTE. Note that the IEEE-488 standard specifies that only the Controller is allowed to conduct a Parallel Poll; it is up to the user to refrain from using this routine unless the P&T-488 is the 488 Controller.

PISTT
Parallel Poll - ist True

This routine sets the "ist" (individual status) message in the P&T-488 true. If the sense bit of the most recent PPE (Parallel Poll Enable) message received by the P&T-488 is the same as the value of the "ist" message, (in this case, true), the affirmative response byte is put into the Parallel Poll response register of the P&T-488. Otherwise, the non-affirmative response byte is put into the Parallel Poll response register. What all this means is that when the 488 Controller conducts a Parallel Poll, the P&T-488 will respond affirmatively if the sense bit of the PPE message was true, non-affirmatively if the sense bit of the PPE message was false. This routine also places the Parallel Poll function in the Standby (PPSS) state. Note that the Parallel Poll response will change if the routines PISTF or PPIDL are called or if the 488 Controller sends another PPE to the P&T-488.

PISTF
Parallel Poll - ist False

This routine is the same as PISTT **except** that it sets the "ist" message false. Thus if the sense bit of the most recent PPE message received by the P&T-488 is FALSE, the AFFIRMATIVE response is put into the Parallel Poll Response register. Otherwise the NON-AFFIRMATIVE response is put there. Note that this is just the opposite of what happens when the routine PISTT is called. Execution of this routine places the Parallel Poll function in the Standby (PPSS) state.

Additional Comments
Parallel Poll - How to use it

There are several ways in which the Parallel Poll response function may be programmed using the P&T-488 and this interface software package. One way is for the 488 Controller (which may or may not be the P&T-488 itself) to address the P&T-488 as a Listener, send the PPC (Parallel Poll Configure) message, then send the PPE (Parallel Poll Enable) message. This will put the Parallel Poll function of the P&T-488 into the Standby (PPSS) state and also define which one of the eight 488 data lines will be used by the P&T-488 when the Controller performs a Parallel Poll. Another method is to put the PPE byte into the memory location reserved for the Parallel Poll response byte. This can be done by defining a five byte string consisting of the P&T-488's Primary Talk address, Primary Listen address, Serial Poll Response byte, Parallel Poll response byte (the desired PPE message), and the EOS (End Of String) byte, then calling the routine STADR. This method defines the response byte, but the Parallel Poll response function of the P&T-488 still needs to be enabled (put into the Standby state). Do do this, a call can be made to the routine PISTT or PISTF. PISTT will make the "ist" message true, while PISTF will make it false. Since an affirmative Parallel Poll response is given only if the "ist" and sense bit of the PPE have the same logical value, one would call PISTT if he wanted the P&T-488 to respond affirmatively to a Parallel Poll and the PPE message was the character h, i, j, k, l, n or o.

By the use of the routines PISTT and PISTF one can readily cause the P&T-488 to give either a non-affirmative or an affirmative Parallel Poll response. One use of this ability would be to define an affirmative response as meaning that the S-100 system wants the Controller to perform some special function (which could be something as simple as to alert the operator that the printer is out of paper), and a non-affirmative response means that the Controller is to continue with normal operation. For the sake of a concrete example, assume that the P&T-488's Listen address is the character _!_ (21 Hex). Assume also that the Controller has sent the string ?!<PPC>h? where the characters <PPC> mean that the PPC message (05 Hex) was sent, **not** that the five characters <, P, P, C and > were sent. Thus the sense bit of the PPE is true, and the P&T-488 is assigned to use data line DIO1 for its Parallel Poll response. Now assume that the S-100 system is listening to transactions on the 488 bus (via the Listen function of the P&T-488) and printing each character on a printer as it is heard. Whenever the printer's status indicates that it is out of paper, the routine PISTT should be called, for it will set the "ist" message true and cause the P&T-488 to respond affirmatively to a Parallel Poll. When the printer has been serviced, the routine PISTF should be called so that the P&T-488's response to a Parallel Poll will be non-affirmative.

One thing that the user should be aware of is that **all** Listeners which are in the addressed state will be assigned the same Parallel Poll response byte when the Controller sends the string <PPC><PPE>. This can give rise to utter confusion when a Parallel Poll is actually executed, so it is wise to have the Controller explicitely unaddress all Listeners (with the Unlisten command, which is the character ?), address the Listener that is to have its Parallel Poll response byte configured, then send the PPC and PPE message, followed by **another** Unlisten.

The P&T-488 along with this software package implements the full Parallel Poll (PP1) function as defined by the IEEE-488 standard. As such, the function may be put back into its Idle state (PPIS) by the Controller addressing the P&T-488 as a Listener and sending the PPC character followed by the PPD character, or by the Controller sending the PPU (Parallel Poll Unconfigure) message, or by calling the routine PPIDL, which implements the "local poll not enabled" message defined in the standard.

SPIDL
Service Request Idle

This routine resets the Service Request function to the Idle state. As a consequence, it also insures that the P&T-488 is passively asserting SRQ false and that the Serial Poll response byte is non-affirmative. Thus execution of this routine is equivalent to the S-100 system making the local message **rsv** (request service) false. This routine is the complement of the routine SPSRQ, which makes the local message **rsv** true.

SPQRY
Serial Poll Query

This routine is called when the user wishes to determine (by means of Serial Poll) which device is requesting service. The Talker addresses in the buffer are sent out one by one and the response monitored to find which one is requesting service. The routine returns when the appropriate device is found or when the buffer with the Talker addresses is emptied.

The register pair HL must contain the address of the first byte of the Serial Poll Query buffer, DE must contain the address of the end of the buffer, and BC the first address of the user-supplied jump table. The Serial Poll Query buffer must contain a character string made up of the Talk or Talk Extended addresses (in any order) of the devices to be tested for Service Request.

This routine causes the Controller function of the P&T-488 to enter the Active state, issue a UNL (Unlisten) message so that devices that had been addressed to Listen will not hear the Serial Poll response bytes sent by each Talker, then issue a SPE (Serial Poll Enable) message, and then send each Talk address in turn. As a precaution against the possibility of a device not unaddressing itself as a Talker whenever another Talk address is sent over the 488 bus, each Talk address is preceeded by a UNT (Untalk) command. When a Talker responds affirmatively to the Poll or when there are no more Talker addresses left in the buffer, this routine issues a SPD (Serial Poll Disable) message and then returns to the calling program.

To allow for the possibility of addressing both normal (single address byte) and extended address (two address bytes) Talkers (otherwise known as T and TE Talkers), this routine sends the first address and then looks to see if a secondary address is to be sent also. If not, it listens for the Talker's response. If there is a secondary address to be sent, it sends it then listens to the Talker's response.

If a Talker responded affirmatively to the Serial Poll, the routine returns to the calling program with 00 Hex in the accumulator, the Serial Poll response byte in register B, and the register pair HL points to the buffer location that contains the Primary Address of that Talker. If no Talker responds affirmatively, the A register contains 40 Hex, register B contains the response of the last Talker, and HL points to the memory location holding the address of that last Talker.

Note that the IEEE-488 standard allows only the Controller to perform a Serial Poll. It is up to the user to insure that this routine is called by his programs only when the P&T-488 is the 488 Controller. Another point the user should be aware of is that this routine does **not** check for valid Talk addresses. It is the user's responsibility to put only valid Talk addresses in the buffer. Since the P&T-488 must wait for the addressed Talker to respond to the Serial Poll, if a non-existant Talk address is in the buffer, the P&T-488 will wait forever for the non-existant Talker to speak its Serial Poll response byte.

SPSRQ
Service Request

A call to this routine causes the P&T-488 to make the SRQ (Service Request) line active true and puts the Service Request function of the P&T-488 into the Service Request (SRQS) state. Thus execution of this routine is equivalent to the S-100 system making the local message rsv (request service) true. This routine then tests the Controller State of the interface. If it is Not Idle, a jump is made to the user-supplied routine SVCRQ. Otherwise the routine waits until the Talker address of the interface is sent out and responds properly to the Serial Poll performed by an external controller. After it has responded, the routine returns to the calling program. The register pair BC must contain the base address of the user-supplied jump table before this routine is called.

If the P&T-488 Controller state is Idle, the P&T-488 ignores all data communication on the 488 bus until it has been polled by the Controller. Thus if the P&T-488 had been a Listener, it will miss everything the Talker says between the time SPSRQ was called and a Serial Poll is conducted by the Controller.

STADR
Set Talker, Listener addresses

This routine copies the Talker and Listener addresses, Parallel Poll and Serial Poll Response bytes and the End Of String (EOS) byte from a table to the P&T-488 interface routines. The register pair HL must contain the address of the beginning of this table. Note that the Parallel Poll response byte is not copied into the interface Parallel Poll Response register. The Parallel Poll Response byte is interpreted in the same manner as the PPE/PPD (Parallel Poll Enable/Parallel Poll Disable) messages received from the Controller during a Parallel Poll Configure.

STATE

Show the state of the P&T-488

This routine passes abbreviated state information to the user in the A register and sets HL to the beginning of the State table. Thus the user can determine the states of the various interface functions if the abbreviated information returned in the A register is insufficient.

The states of various interface functions are mapped into the following bit positions of the A register:

.... ..00	Both Talk and Listen functions are idle	
.... ..01	TIDS-	(Not Talker Idle State)
.... ..10	LIDS-	(Not Listener Idle State)
.... .0..	PPIS	(Parallel Poll Idle State)
.... .1..	PPSS	(Parallel Poll Standby State)
...0 0...	LOCS	(Local State)
...0 1...	LWLS	(Local With Lockout State)
...1 0...	REMS	(Remote State)
...1 1...	RWLS	(Remote With Lockout State)
.0..	CIDS	(Controller Idle State)
.1..	CIDS-	(Not Controller Idle State)

Example: If the Controller State is Not Idle, the Remote-Local State is LOCAL, Parallel poll is Idle and Talker Not Idle, the A register would contain 41 Hex.

The state table itself is comprised of six bytes, each one of which is associated with one 488 interface function. The actual state of the function is represented by the bit pattern of its associated state byte. Some states have the same bit pattern and are distinguished only by what routine is being executed. For example, if you look at the encoding for the Talk states you will find that TADS, TACS and SPAS are all represented by the same bit pattern. However, the P&T-488 interface software can distinguish among them by the fact that if it is not running either the Talk routine or the Serial Poll response routine, the state is TADS. If it is running the Talk routine, the state is TACS, and if it is running the Serial Poll response routine, the state is SPAS. The user does not need to concern himself with which one of the three states the Talk function is in because he only needs to know whether the Talk function has been addressed by a Controller, and he will make the inquiry at a time when neither the Talk nor the Serial Poll response routines are being executed.

State Table

TSTAT Talk Interface Function State byte

....0	TIDS	Talk Idle State
....1	TADS	Talk Addressed State
....1	TACS	Talk Active State
....1	SPAS	Serial Poll Active State
.... .0..	SPIS	Serial Poll Idle State
.... .1..	SPMS	Serial Poll Mode State
.... 0...	TPIS	Talk Primary Idle State
.... 1...	TPAS	Talk Primary Addressed State

LSTAT Listen Interface Function State byte

....	...0	LIDS	Listen Idle State
....	...1	LADS	Listen Addressed State
....	...1	LACS	Listen Active State
....	.0..	LPIS	Listen Primary Idle State
....	.1..	LPAS	Listen Primary Addressed State
....	0...	....	non-buffered Listen function
....	1...	....	buffered Listen function
...0	....	....	do not return from Listen routine upon receipt of EOS message
...1	....	....	return from Listen routine upon receipt of EOS message

SSTAT Service Request Interface Function State byte

..00	....	NPRS	Negative Poll Response State
..01	....	SRQS	Service Request State
..10	....	APRS	Affirmative Poll Response State

RSTAT Remote-Local Interface Function State byte
(Note: This function is not implemented, but these definitions will be used when it is.)

...0	0...	LOCS	Local State
...0	1...	LWLS	Local With Lockout State
...1	0...	REMS	Remote State
...1	1...	RWLS	Remote With Lockout State

PSTAT Parallel Poll Interface Function State byte

....	...0	PPIS	Parallel Poll Idle State
....	...1	PPSS	Parallel Poll Standby State
....	...1	PPAS	Parallel Poll Active State
....	..0.	....	ist (individual status) message is false
....	..1.	....	ist message is true
....	.0..	PUCS	Parallel Poll Unaddressed to Configure
....	.1..	PACS	Parallel Poll Addressed to Configure

CSTAT Controller Interfacte Function State byte

....	0000	CIDS	Controller Idle State
....	0001	CADS	Controller Addressed State
....	0010	CTRS	Controller Transfer State (not yet implemented)
....	0011	CACS	Controller Active State
....	0011	CPWS	Controller Parallel Poll Wait State
....	0011	CPPS	Controller Parallel Poll State
....	0011	CAWS	Controller Active Wait State
....	0110	CSBS	Controller Standby State
....	1000	CSWS	Controller Synchronous Wait State
...0	....	CSNS	Controller Service Not Requested State
...1	....	CSRS	Controller Service Requested State
..0.	....	SNAS	System Control Not Active State
..1.	....	SACS	System Control Active State

TALK
Talk-Only

This routine allows the user to send data from the S-100 system to other devices on the 488 bus. The data may be in any byte oriented form: ASCII characters (with or without parity), BCD, binary, etc. The information is put into a buffer in memory before the routine is called.

The register pair HL must contain the address of the beginning of the buffer, DE must contain the address of the end, and BC the address of the beginning of the user-supplied jump table. If the accumulator (A register) contents are non-zero, the last byte in the buffer will be sent with EOI (End Or Identify) active true, otherwise the last byte will be sent with EOI passive false. All other bytes of the string are sent with EOI passive false.

A call is made to the user-supplied routine BREAK after each byte is sent, which allows the user to interrupt or defer further 488 bus transactions while he executes some other routine. To continue the Talk function, he need only execute a RETurn. All registers may be changed between the time BREAK was entered and the RETurn to the Talker routine was executed.

The SRQ (Service Request) line is checked after each byte is transmitted, and if it is active true, the routine determines whether the P&T-488 is the Controller-In-Charge. (Actually, CSTAT is tested to see if the Controller function is in the non-Idle state.) If it is the Controller-In-Charge, then a call is made to the user-supplied routine SVCRQ. After the user has serviced the Service Request, he need only execute a RETurn to continue talking from where the routine left off.

This routine implements the Talk Only (ton) function described in the IEEE-488 standard. Thus execution of this routine sets the Talk State byte to Talker Addressed. Execution of this routine also resets the Listen State byte to the Idle (LIDS) State.

If the user wishes instead to implement the Addressed Talker function described in the IEEE-488 standard, (i.e., the transition from TIDS to TADS should occur only if the Controller has addressed the P&T-488 as a Talker), he should call the routine STATE and then call TALK only if the Talk State byte shows that the P&T-488 is addressed to Talk.

XCTRL
Respond to External Controller

Each command presented by an external Controller (some device other than the P&T-488) is examined in turn and the states of the various interface functions are modified as necessary. A return is made to the calling program when the external Controller relinquishes the bus (asserts ATN passive false). An exception is made when the external Controller is conducting a Serial Poll: in this case the routine responds appropriately to the poll and returns to the calling program after the poll is concluded (a Serial Poll Disable command has been received followed by ATN going passive false).

This routine is to be called only upon ATN being made active true (low) by an external Controller. Load the register pair BC with the base address of the user-supplied jump table before calling XCTRL.

Since the states of the interface functions may have changed (due to commands from the external Controller), it may not be appropriate to return to the routine that was interrupted by the external Controller.

PAGE	ROUTINE NAME	FUNCTION
CS-17	BREAK	Allows S-100 operations during buffered 488 communication
CS-18	BUFUL	Fixup for Listen Buffer full
CS-18	DVCLR	Application dependent. A Device Clear (DCL) was detected
CS-18	IFCLR	Re-initialize due to 488 Interface Clear (IFC)
CS-18	NOLSN	No listeners on 488 bus - ERROR
CS-18	POC	Re-initialize due to S-100 Power-On Clear or Reset
CS-18	SVCRQ	The 488 Service Request line is active true Find the device and service it
CS-19	TRIGR	Start whatever function that was waiting for Group Execute Trigger (GET)
CS-19	XATN	Some other device made the 488 ATN line true

The P&T-488 interface software uses a jump table to access the user-supplied routines. It is the user's responsibility to provide the jump table, and it must have the form shown below. The user must set the register pair BC to the address of the first entry of the user jump table before calling routines supplied in the P&T-488 sofware package.

User-Supplied Jump Table
Organization

```
JMP     TRIGR
JMP     DVCLR
JMP     BUFUL
JMP     IFCLR
JMP     BREAK
JMP     NOLSN
JMP     SVCRQ
JMP     POC
JMP     XATN
```

BREAK

After each data byte or command is transferred on the 488 bus, a call is made to BREAK. The accumulator (A register) contains the byte last communicated, and the register pair HL points to the buffer location of the last byte sent or received. This routine allows the user to interrupt or defer until later any further 488 transactions, so that he may perform other operations. Examples include polling the keyboard for operator input, performing a background print routine, etc. It also gives the user the opportunity to regain control of the S-100 system short of pushing RESET or turning off the power.

The BREAK routine is also useful for those cases in which the Talker does not make EOI true on the last byte; since the routine LISTN does not return to the user's calling routine until it sees an EOI (or optionally an EOS), one can see there is a fundamental problem. However, since a call is made to BREAK after each byte, the user can test each byte and determine if it is the end of transmission.

The only register that needs to be preserved is the Stack Pointer (SP). Transactions on the 488 bus may be resumed by executing a RETurn.

BUFUL
Listen Buffer Full

A jump is made to this routine when the Listen buffer is filled. The user should empty or redefine the buffer, then continue Listening by reinitializing all registers (A, BC, DE and HL) and calling LISTN.

DVCLR
Detected a Device Clear

A jump is executed to this routine whenever the Controller sends a Device Clear command. The user should perform whatever function Device Clear means in his system. (The proper response is device dependent.)

IFCLR
Detected an Interface Clear

A jump is made to this routine whenever an external Controller sends an Interface Clear (IFC) command. The P&T-488 must be re-intialized (for example, use INIT followed by STADR).

NOLSN
Nobody's Listening

A jump is made to this routine whenever the P&T-488 was to have said something as a Talker but found that no one was Listening. This is an error condition: correct it, reinitialize the registers and then call TALK again. (The only time that Not Ready For Data (NRFD) and Not Data Accepted (NDAC) can both be false at the same time is if there are no Listeners. It is this condition that causes a jump to NOLSN.)

POC
S-100 Power-On Clear or Reset

A jump is made to this routine whenever the P&T-488 interface senses an S-100 Reset or Power-On Clear. It will have to be re-initialized (use INIT followed by STADR).

SVCRQ
488 Service Request

This routine is CALLed whenever the 488 Service Request (SRQ) line is true and the P&T-488 is the Controller-In-Charge. Find the device (by using SPQRY), service it, then execute a RETurn to resume 488 transactions. The only register that needs to be preserved is the Stack Pointer.

TRIGR
488 Group Execute Trigger

This routine is CALLed whenever the Group Execute Trigger (GET) command is received. Start whatever function was waiting for the trigger, then RETurn to resume 488 transactions. The only register that needs to be preserved is the Stack Pointer.

XATN
An External Controller wants Control

This routine is CALLed whenever some other device on the 488 bus has made ATN active true (low). Call STATE to get the present Talker, Listener, etc. state information. Save this information, put the base address of the user-supplied jump table in register pair BC and call XCTRL. Then call STATE again to find out if the external Controller has changed the states of the Talk, Listen, etc. functions. If not, just execute a RETurn to resume 488 transactions from where they were interrupted by the external Controller. If the states are changed, perform whatever function the external Controller has commanded.

488 Bus Monitor

Description

This program shows all data and all commands sent over the IEEE-488 bus. Common non-printing characters (space, horizontal tab, carriage return and line feed) are shown as a message enclosed in angle brackets. As an example, "<HT>" is printed on the console printer each time a horizontal tab is detected.

The program begins by placing dummy Listen and Talk addresses in the interface. The parity bit is set (logic 1), so there is no way that the 488 interface can be addressed as either a Listener or a Talker by the Controller. (The parity bit of each address sent by the Controller is set to zero before comparing it to the interface Listen and Talk addresses.)

After the addresses are set up, the interface is cleared by a call to the routine INIT. Note that the B register is non-zero because we do not want to send an IFC (interface clear) signal over the bus. Only the System Controller is allowed to send IFC, and we are not he.

Then the interface is set to the Controller Standby state (at statement label RST2) which causes the 488 routines to assume that we are the Controller-in-Charge. We are not, but this is done so that the Listen routine will branch to the user-supplied routine SREQ each time a Service Request (SRQ) is detected. Otherwise there is no easy way of making this program print a special message each time a Service Request is pending.

Finally the Stack Pointer is reset, register pair BC is set pointing to the jump table of user-supplied routines, and the Listen routine is called. No buffer is used and the End-of-String (EOS) byte is ignored. The Listen routine will return each time it receives an END byte (a data byte with the EOI line active true). A special message is printed on the system console to show that an END byte was received and then the program is restarted.

The user-supplied routine BRK is called each time a byte of data or command appears on the 488 bus. All printing characters are sent to the console printer as is and a RETurn is made to the calling routine. The non-printing characters space (20 Hex), Horizontal Tab (9) and Line Feed (0A Hex) are replaced with the messages <SPACE>, <HT> and <LF> respectively. The non-printing character Carriage Return (0D Hex) causes the message <CR> to be printed followed by a carriage return and a line feed.

The user-supplied routine XTN prints a message to show that a Controller is active (ATN active true) and then calls the routine XCTRL to listen to the commands sent by the Controller. Each byte sent by the Controller is placed in location LBYTE and a branch is made to the routine BRK.

Special Cases:

Each time the Controller becomes active (asserts ATN active true), a carriage return-line feed is sent to the console device, followed by the string "COMMAND:", followed by another carriage return-line feed pair. Similarly, each time the Controller becomes inactive (ATN is false), a carriage return, line feed, the string "DATA:", carriage return and a line feed is sent to the console. All characters printed after "COMMAND:" and before "DATA:" are sent by the Controller, and are instructions to the various 488 devices (for example, "?" means Unlisten, which means that no device should be a Listener when the Controller relinquishes the bus).

All characters which are printed after "DATA:" and before "COMMAND:" are data (otherwise known as device-dependent messages). They may be readings from a DVM which has been commanded to be a Talker, etc.

```
;       488 BUS MONITOR PROGRAM
;
;       All activity on the 488 bus is shown by messages printed
;       on the console printer.
;
;
        ORG     ØB7ØØH
;
CSTAT   EQU     8ØØAH   ;controller state byte
ENTBL   EQU     8Ø26H   ;memory addr of beginning of P&T-488
                        ;  jump table
;
INIT    EQU     ENTBL
LISTN   EQU     ENTBL+6
XCTRL   EQU     ENTBL+15H
STADR   EQU     ENTBL+51H
;
MNITR   EQU     ØØØØH   ;system monitor entry address
PRT     EQU     ØD1Ø6H  ;console print routine entry address
;
;       It is assumed that the routine PRT prints the character
;       held in the A register, then returns to the calling
;       routine.  All registers (except the flags) are assumed
;       to be unmodified by PRT.
;
;
START:  LXI     SP,STAK ;initialize the stack pointer
        LXI     H,DUMAD ;set up dummy listen, talk addresses
        CALL    STADR
RSTRT:  MVI     B,2     ;clear 488 interface, but do not send IFC
        CALL    INIT
RST2:   MVI     A,6     ;set CSTAT to standby (thus fooling the
                        ;other routines into jumping to SVCRQ
        STA     CSTAT   ;upon detection of a service request)
        LXI     SP,STAK ;initialize stack pointer
        LXI     B,JTBL  ;set up pointers
        MVI     A,Ø     ;non-buffered listener, ignore EOS byte
        CALL    LISTN
        LXI     B,ENDMS ;show that an <END> message has been
        CALL    MSG     ;  received
        JMP     RST2
;
;
;       USER-SUPPLIED JUMP TABLE
```

```
;
JTBL:   JMP     TRGR
        JMP     DVCL
        JMP     BFL
        JMP     ICLR
        JMP     BRK
        JMP     NLS
        JMP     SREQ
        JMP     POCRST
        JMP     XTN
;
TRGR:   LXI     B,TMS   ;print trigger message
        CALL    MSGCR
        RET
;
DVCL:   LXI     B,DVMS  ;print device clear message
        CALL    MSGCR
        RET
;
BFL:    LXI     B,BMS   ;we should never get this message
        CALL    MSGCR   ;but if we do, print it and go to monitor
        JMP     MNITR
;
ICLR:   LXI     B,IFMS  ;print interface clear message
        CALL    MSGCR
        JMP     RSTRT   ;restart (initialize 488 interface)
;
;       LOOK AT THE LAST COMMUNICATED CHARACTER
;
BRK:    CPI     ØDH     ;<CR>?
        JZ      CRMSG   ;..print <CR> message
        CPI     ØAH     ;<LF>?
        JZ      LFMSG   ;..print <LF> message
        CPI     9       ;<HORIZONTAL TAB>?
        JZ      HTMSG   ;..print <HT> message
        CPI     2ØH     ;<SPACE>?
        JZ      SPMSG   ;..print <SPACE> message
        CALL    PRT     ;print char
        RET
;
CRMSG:  PUSH    A       ;save character for later
        LXI     B,CRMS  ;print <CR> message
        CALL    MSG
        POP     A
        CALL    PRT     ;then do the carriage return
        MVI     A,ØAH   ;finish with a line feed
        CALL    PRT
        RET
;
LFMSG:  LXI     B,LFMS  ;print <LF> message
        CALL    MSG
        RET
;
HTMSG:  LXI     B,HTMS  ;print <HT> message
        CALL    MSG
        RET
;
SPMSG:  LXI     B,SPMS  ;print <SPACE> message
        CALL    MSG
        RET
;
NLS:    LXI     B,NLMS  ;we should never reach this point
        CALL    MSGCR   ;but if we do, print message and
                        ;   go to the monitor
        JMP     MNITR
;
SREQ:   LXI     B,SRQMS ;print service request message
        CALL    MSGCR
        RET             ;let the controller-in-charge take care
                        ;   of the service request
;
POCRST: LXI     B,POCMS ;print S-1ØØ reset message
        CALL    MSGCR
        JMP     RSTRT   ;re-initialize the 488 interface
;
XTN:    LXI     B,XTNMS ;print external ATN message
        CALL    MSGCR
        LXI     B,JTBL
        CALL    XCTRL   ;listen to the commands and update
                        ;   state of interface
        LXI     B,DATMS ;print data message
        CALL    MSGCR
        JMP     RST2    ;go back to listen-only function
;
MSG:    LDAX    B
        CALL    PRT     ;print message
        ANI     8ØH     ;see if parity set
        INX     B
        JZ      MSG     ;..no, so print some more
        RET
;
MSGCR:  CALL    MSG     ;print the message, terminate with CRLF
        MVI     A,ØDH   ;output a carriage return
        CALL    PRT
        MVI     A,ØAH   ;then a line feed
        CALL    PRT
        RET
;
;  DUMMY LISTEN, TALK ADDRESSES-
;       The parity bit is set, preventing the 488 interface
;       from ever recognizing a talk or listen address
;
DUMAD:  DB      ØAØH    ;dummy listen address
        DB      ØCØH    ;dummy talk address
        DB      ØFFH    ;parallel poll response byte (no response)
        DB      ØFFH    ;serial poll response byte (no response)
        DB      ØAH     ;EOS CHARACTER (IGNORED IN THIS PROGRAM)
TMS:    DB      'DEVICE TRIGGE', ØD2H
DVMS:   DB      'DEVICE CLEA', ØD2H
BMS:    DB      'LISTEN BUFFER FUL', ØCCH
```

```
IFMS:    DB      ØDH, ØAH,'INTERFACE CLEA', ØD2H
NLMS:    DB      'NO LISTENE', ØD2H
SRQMS:   DB      ØDH,ØAH,'SRQ ACTIVE TRU', ØC5H
POCMS:   DB      ØDH,ØAH,'POC/RESET TRU', ØC5H
XTNMS:   DB      ØDH,ØAH,'COMMAND',ØBAH
ENDMS:   DB      '<END',ØBEH
CRMS:    DB      '<CR',ØBEH
LFMS:    DB      '<LF',ØBEH
HTMS:    DB      '<HT',ØBEH
SPMS:    DB      '<SPACE',ØBEH
DATMS:   DB      ØDH,ØAH,'DATA',ØBAH
;
         DS      64D      ;stack area
STAK:
;
         END
```

488 Sample Program

Description

This program demonstrates how to set up the P&T-488 as a Controller to send out bus commands (in this case, the Talk and Listen addresses of two devices), then become a Listener. It also illustrates how to allow for an abort command (by the use of the routine BRK).

The program begins by setting up the Stack Pointer and then sets the Listen and Talk addresses of the P&T-488 interface. The 488 bus and interface are cleared by a call to the routine INIT, which is followed by a call to CNTRL, which sends out the contents of the buffer CMDSTR. These commands first tell all active Listeners to stop Listening, then all active Talkers to stop Talking. Talker 5 is then told it is the designated Talker, and Listener 3 (which in this case is the P&T-488) is told it is the sole Listener.

The state of the interface is checked by a call to the routine STATE after the commands are sent. If the Listen state is in the IDLE mode, a jump is made to the routine NTLSN, which prints an error message on the printer and then jumps to the system monitor. (Since the Listen address of the P&T-488 was sent as a command this particular branch should never be executed.)

As preparation for the use of the routine LISTN, the mode switch (A register) is set so that a buffer will be used and the EOS byte will not cause LISTN to RETurn to the calling program. Each time the LISTN routine returns (due to an END byte; i.e. a data byte sent with EOI active true) or the buffer fills (i.e., a branch is made to BFL), the contents of the buffer are printed, the buffer pointers are reset and the LISTN routine is called again.

The user-supplied routine BRK is used to allow the user to suspend 488 transactions and jump back to the system monitor by pressing Control C on the keyboard. It is assumed that the keyboard status is available at Port 0, bit 2 is zero when a key has been depressed, and the keycode is available at Port 1.

The skeleton of the user-supplied routine SREQ is shown, in which a Serial Poll is made of 488 devices 1, 17, 7 and 3. The address of the first device to respond is placed in the A register but the rest of the routine is device dependent. For example, a printer may request service when it is out of paper, the ribbon jams, or some other error condition. A reasonable response to a paper out condition would be a message sent to the console (assuming it is not the printer needing service) informing the operator of the printer's problem.

```
;        488 SAMPLE PROGRAM
;
;        Assert control, send out the talk address of some
;        other device, the listen address of the P&T-488,
;        and then listen to the talker
;
;
         ORG      ØB7ØØH
;
CSTAT    EQU      8ØØAH    ;addr of controller state byte
ENTBL    EQU      8Ø26H    ;addr of beginning of P&T-488
                           ;  jump table
;
INIT     EQU      ENTBL
LISTN    EQU      ENTBL+6
CNTRL    EQU      ENTBL+ØCH
STATE    EQU      ENTBL+12H
XCTRL    EQU      ENTBL+15H
SPQRY    EQU      ENTBL+18H
STADR    EQU      ENTBL+51H
;
MNITR    EQU      ØØØØH    ;system monitor entry address
PRT      EQU      ØD1Ø6H   ;console print routine entry address
;
;        It is assumed that the routine PRT prints the character
;        held in the A register, then returns to the calling
;        routine.  All registers (except the flags) are assumed
;        to be unmodified by PRT.
;
;
START:   LXI      SP,STAK ;initialize the stack pointer
         LXI      H,ADRTBL ;set up P&T-488 listen, talk addresses
         CALL     STADR
         MVI      B,Ø      ;clear 488 interface and send IFC
         CALL     INIT
         LXI      H,CMDSTR ;load HL with beginning address
                           ;  of command string
         LXI      D,CMDEND ;load DE with end addr of command string
         LXI      B,JTBL   ;load BC with beginning addr of jump table
         CALL     CNTRL    ;send the commands
         CALL     STATE    ;find out what P&T-488 state is
         ANI      2        ;keep only listener bit
         JZ       NTLSN    ;..P&T-488 in listener idle mode
LSNLUP:  MVI      A,1      ;use buffer, ignore EOS character
         LXI      H,LSNTBL
         LXI      D,LSNEND ;addr of last byte of listen buffer
         LXI      B,JTBL   ;set up pointers
         CALL     LISTN
LSNPRT:  LXI      D,LSNTBL ;now print the contents of the
                           ;  listen buffer
         DCX      D
LSNPR1:  INX      D        ;point to next byte in buffer
         LDAX     D
         CALL     PRT
```

```
        MOV     A,E     ;have we done the last byte yet?
        CMP     L
        JNZ     LSNPR1
        MOV     A,D
        CMP     H
        JNZ     LSNPR1
        JMP     LSNLUP  ;printed the last byte, so start
                        ; listening again
;
;       USER-SUPPLIED JUMP TABLE
;
JTBL:   JMP     TRGR
        JMP     DVCL
        JMP     BFL
        JMP     ICLR
        JMP     BRK
        JMP     NLS
        JMP     SREQ
        JMP     POCRST
        JMP     XTN
;
;       The following routines should not be entered in
;       this program.  If they are, a message is printed
;       (to aid in debugging) and then a jump is made to
;       the system monitor.
;
TRGR:   LXI     B,TMS   ;print trigger message
        CALL    MSGCR
        JMP     MNITR
;
DVCL:   LXI     B,DVMS  ;print device clear message
        CALL    MSGCR
        JMP     MNITR
;
ICLR:   LXI     B,IFMS  ;print interface clear message
        CALL    MSGCR
        JMP     MNITR
;
NLS:    LXI     B,NLMS  ;print no listener message
        CALL    MSGCR
        JMP     MNITR
;
POCRST: LXI     B,POCMS ;print S-1ØØ reset/power-on clear
        CALL    MSGCR
        JMP     MNITR
;
XTN:    LXI     B,XTNMS ;print external controller message
        CALL    MSGCR
        JMP     MNITR
;
NTLSN:  LXI     B,NTLMS ;get P&T not listening message
        CALL    MSGCR   ;print it
        JMP     MNITR   ;then go to the system monitor
;
;*****************  END OF ABNORMAL BRANCHES  ***************
```

```
;
BFL:     JMP      LSNPRT  ;print the contents of the buffer
                          ;  then continue listening
;
BRK:     IN       Ø       ;get keyboard status
         ANI      2       ;look at ready bit
         RNZ              ;..no key has been depressed
         IN       1       ;get char from keyboard
         ANI      7FH     ;strip parity bit
         CPI      3       ;<Control C>?
         RNZ              ;..no, continue with 488 transactions
         JMP      MNITR   ;user pressed Control C.  ABORT!!!!!
;
SREQ:    LXI      H,SPSTR ;put beginning address of serial poll
                          ;  string in HL
         LXI      D,SPEND ;and end address in DE
         LXI      B,JTBL  ;user jump table address in BC
         CALL     SPQRY   ;find out which device wants service
         MOV      A,M     ;put device's address in A register
          *
          *               THE REST IS DEPENDENT ON THE DEVICE
          *
         RET
;
MSG:     LDAX     B
         CALL     PRT     ;print message
         ANI      8ØH     ;see if parity set
         INX      B
         JZ       MSG     ;..no, so print some more
         RET
;
MSGCR:   CALL     MSG     ;print the message, terminate with CRLF
         MVI      A,ØDH   ;output a carriage return
         CALL     PRT
         MVI      A,ØAH   ;then a line feed
         CALL     PRT
         RET
;
ADRTBL:  DB       '#'     ;listen address 3
         DB       'C'     ;talk address 3
         DB       ØFFH    ;parallel poll response byte (no response)
         DB       ØFFH    ;serial poll response byte (no response)
         DB       ØAH     ;EOS character (ignored in this program)
CMDSTR:  DB       '?'     ;universal unlisten
         DB       ' '     ;universal untalk
         DB       'Ē'     ;primary talk address 5
CMDEND:  DB       '#'     ;primary listen address 3 (P&T-488)
;
SPSTR:   DB       'A'     ;primary talk address 1
         DB       'Q'     ;primary talk address 17
         DB       'G'     ;primary talk address 7
SPEND:   DB       'C'     ;primary talk address 3
;
TMS:     DB       'DEVICE TRIGGE', ØD2H
DVMS:    DB       'DEVICE CLEA', ØD2H
```

```
BMS:     DB      'LISTEN BUFFER FUL', ØCCH
IFMS:    DB      ØDH, ØAH,'INTERFACE CLEA', ØD2H
NLMS:    DB      'NO LISTENE', ØD2H
POCMS:   DB      ØDH,ØAH,'POC/RESET TRU', ØC5H
XTNMS:   DB      ØDH,ØAH,'EXTERNAL CONTROLLE',ØD2H
NTLMS:   DB      ØDH,ØAH,'P&T NOT ADDRESSED AS A LISTENE',ØD2H
;
LSNTBL:  DS      255       ;listen buffer
LSNEND:  DS      1         ;last byte of listen buffer
;
         DS      64D       ;stack area
STAK:
;
         END
```

DINK

Description

The program DINK has been included for several reasons. The first is that it allows the user to easily exercise the functions provided by the P&T-488 Custom Software package and interface card. Another is that it allows the user to easily experiment with a 488 device so that he can thoroughly understand what messages it needs before he writes the assembly language code. Finally, by looking at how DINK is written, the user can see how the P&T-488 Custom Software package can be used. It should be noted, however, that not all functions provided by the P&T-488 software package are used in DINK. As an example, DINK uses only the non-buffered Listen function, so one cannot learn from DINK how to use the buffered Listen function.

The routine DINK was written so that it is fairly easy to see what is going on. As a consequence, the code is not optimal, either in execution speed or in the amount of memory that it requires. One could shorten it considerably, but at the expense of clarity.

In order to use DINK, the user must first add two routines: the first is for console input (which is called KBIN) and the other is console output (called PRT). The routine KBIN should get a character from the console keyboard and return with the character in the accumulator. No other register (except the flags) may be changed. The routine PRT should print the character held in the accumulator on the console output device, and then return. Again, no register (except the flags) may be altered. Examples of these routines are given at the end of the listing of DINK. The examples shown use the console input and output routines which are available in the CP/M operating system (CP/M is a product of Digital Research). The console output routine of CP/M needs the character in register C, and the CP/M input routine returns with the character in register A.

Once these two routines have been added to DINK, the user should modify (if necessary) the EQUate for ENTBL and the ORG and then assemble DINK. (The EQUate for ENTBL must be modified if the ORG of PNT488 has been changed.) PNT488 should also be assembled, then it and DINK should be loaded into memory. Finally the routine DINK should be executed from the location START.

Now that DINK is running, what does one do with it? The first thing is to respond to the message it sent out. Assuming that PRT was correctly written and DINK was properly assembled, loaded and run, the user should see the message

DINK 1-2-80
Enter P&T-488 Listen and Talk addresses, Parallel Poll response
Serial Poll status and the End-of-String (EOS) bytes

If this message did not appear on the console, the subroutine PRT should be carefully checked, and the steps of assembly, loading and executing DINK should be tried again.

Now that the message has appeared on the console answer it with the appropriate characters. The computer will store the characters in a line buffer but will not act upon them until the user indicates that he is finished with his response by pressing the <carriage return> key. The line input routine incorporates several editing functions. Individual characters may be "erased" from the line by pressing the <delete> (sometimes labelled RUBOUT) key. The computer will "forget" the preceding character and the console output device will print the character DELCHR in response. (This character can be changed by the user to whatever code is appropriate for his console. The usual characters are Ø8 Hex (backspace) or 7F Hex (delete).) Multiple characters may be erased by pressing the delete key once for each character to be erased. The whole line can be erased by typing a Control X (press and hold the CONTROL key, then press the X key, then release both keys). A # will be printed and the console will advance to the next line to show that the line is being restarted.

The line input routine has one more special function key: ESCAPE. The line input routine will not perform any special function associated with the first key depressed after the ESCAPE key. Instead, it will put the key code into the line buffer just as it does for any normal character. Thus the ESCAPE key allows any key code to be placed into the buffer, including the codes for <carriage return>, <ESCAPE>, <Control X> and <delete>. For instance, if one types ABC<Control X>EF<carriage return> the computer accepts this as the same as EF<carriage return> (remember that Control X> erases everything that was typed before it). However, if ABC<ESCAPE><Control X>EF<carriage return> were typed, the computer remembers this as the key codes ABC<Control X>EF because the ESCAPE caused the line input routine to place the key following the ESCAPE into the buffer instead of performing the special function.

Valid Listen addresses are any single character from <space> through >, inclusive. (See the table **Code Assignments for "Command Mode" of Operation** for further details.) Valid Talk addresses are any single character @ through ↑, inclusive. The Parallel Poll response byte should be one character selected from <accent grave> through o, inclusive. This byte is really the same as a Parallel Poll Enable byte sent by the Controller, in that the three least significant bits of the byte indicate which data (DIO) line will be used by the P&T-488 to respond to a Parallel Poll, and the fourth least significant bit is the Sense bit which selects an affirmative poll response if it has the same logical value as the **ist** (individual status) message. The Serial Poll status byte and the EOS byte may be set to any character. (Remember that <delete>, <Control X>, <ESCAPE> or <carriage return> must be preceded by <ESCAPE> to prevent the line input routine from deleting a character, deleting the line or terminating the collection of the string, respectively). These characters are used to set up the P&T-488's own Listen and Talk addresses as well as the bytes it will respond with when it responds to a Parallel or Serial Poll. The EOS byte may be used by the Listen function to detect the end of a string sent by the Talker. If it is desired to make the EOS character a carriage return, remember to press the <ESCAPE> key before the carriage return.

After the line has been entered DINK will print
Enter function code
on the console. The code is a single character, and the following table shows the codes and their corresponding functions.

Code	Function Performed
A	Get new Listen, Talk addresses, Poll response bytes
C	Become the 488 Controller
G	Use function GIM to control 488 General Interface Management lines manually
I	Initialize the P&T-488 and optionally send IFC true
L	Listen to the Talker and print what he says
M	Put the P&T-488 into the Parallel Poll Idle (PPIS) state
N	Make the local "ist" (individual status) message false
O	Make the local "ist" (individual status) message true
P	Do a Parallel Poll and print the response
Q	Do a Serial Poll and print the response

R Put the P&T-488 into the active Service Request (SRQS) state
S Print a summary of the state of the P&T-488 and of the
  488 Data and GIM lines (all numbers in Hex)
T Talk on the 488 bus
V Put the P&T-488 into the No Service Requested (NPRS) state

The following paragraphs are expansions of the descriptions of each of the functions.

Function A sets the Listen and Talk addresses, the Parallel and Serial Poll responses, and the End-of-String bytes, just as was done when DINK was first started. By use of this function one can change the addresses or the poll responses of the P&T-488. Note that the Parallel Poll response can also be changed by the Controller. It can send the Listen address of the P&T-488 followed by the PPC (Parallel Poll Configure) byte, then the PPE (Parallel Poll Enable) byte. The PPE sets the Parallel Poll response byte of the P&T-488.

Function C causes the P&T-488 to become the 488 Controller. Note that it asserts control immediately, so the user must take care that he is not interrupting the current Talker if it is desired to take control synchronously. Then the routine asks for a string. When the user has typed in the string and terminated it with a carriage return, the string is sent over the bus. Remember that the characters of the string have special meaning, as they are now commands. For example, the character _ (underscore) means UNTALK (all talkers are to revert to the Talker Idle (TIDS) state).

This function will lock up the S-100 system until all commands have been sent. If one of the devices on the 488 bus is performing the Acceptor Handshake but does not complete it, the S-100 system will remain locked up. If there are not any devices connected to the P&T-488, it will send the command string on the bus (even though no one is there to hear it) because another section of the P&T-488 is performing the Acceptor Handshake. It is doing this so that the state of the P&T-488 will be updated in response to what the Controller says.

Function G allows one to manually set or reset selected General Interface Management lines of the 488 bus. It is provided so that this software package is compatible with programs written for an older package (Version 1.3). In general, the user should be discouraged from using the function GIM because most of the functions can be better performed by calling other routines.

The lines that function G allows access to are IFC, SRQ, REN and EOI. IFC is better controlled from the routine INIT (function code I), SRQ from the routines SPIDL (function code V) and SPSRQ (function code R), and EOI from TALK (function code T) or PPQRY (function code P). The only line that is not accessible from a better routine is REN.

One can set/reset these lines by typing in an appropriate character followed by a carriage return. The character is placed in the A register and then the PNT488 routine GIM is called. By referring to the description of GIM, it is seen that the IFC, SRQ and EOI lines can be made false while REN can be made true by using any one of the characters <Control B>, ", B or b.

The user should be aware that the routines in PNT488 may not be aware of changes made to the lines by use of this function, and things can get quite confused. The ONLY reason that this function should even be considered is to gain access to the REN line. If it is used, the user should note the state of the other three lines and preserve their state while changing REN.

Function I causes all P&T-488 states to revert back to their Idle state. The user is asked whether an IFC (Interface Clear) is to be sent over the 488 bus also. If he answers with a Y an IFC will be sent; this will cause all the other 488 devices to revert back to their initialized states. If the answer is N then an IFC will not be sent and the 488 devices other than the P&T-488 will not be affected by this function. If any other character is typed as the first character of the string a message is printed on the console informing the user that only these two responses are allowed.

Function L sets up the P&T-488 as a non-buffered Listener. Each character heard by the P&T-488 is printed on the console as it is heard. Control (non-printing) characters are printed as two-character strings, the first being uparrow (↑) and the second being the character with 40 Hex added to it to make it printable. For example, a null will be printed as ↑@, a <Control X> (otherwise known as CAN or CANCEL) as ↑X, etc. The user is asked

Return upon receipt of EOS byte?

If the response is Y or y the function will terminate when a character matching the EOS byte is received. Upon termination of this function the user is asked to select the next function. The function will always terminate upon receipt of an END message (the EOI line made true by the Talker while speaking a byte). In this case the message **<END>** is printed on the console and the user is asked to select the next function.

Function M causes the routine PPIDL in PNT488 to be called, which in turn places the P&T-488 into the Parallel Poll Idle (PPIS) state. All that this means is that the P&T-488 will not participate in a Parallel Poll.

Function N causes the routine PISTF in PNT488 to be called. This routine sets the ist (individual status) message false and then puts the appropriate response byte in the Parallel Poll response register of the P&T-488. It then puts the P&T-488 into the Parallel Poll Standby (PPSS) state. See the description of PISTF for more details.

Function O causes the routine PISTT in PNT488 to be called. This routine sets the ist message true then puts the appropriate response byte in the Parallel Poll response register of the P&T-488. It then puts the P&T-488 into the Parallel Poll Standby (PPSS) state. See the description of PISTT for more details.

Function P causes the routine PPQRY in PNT488 to be called, which in turn executes a Parallel Poll. The response is then printed (in Hex) on the console.

Function Q sets up the P&T-488 to do a Serial Poll. The user is asked to enter a string, which should be the Talk addresses of the devices to be polled. Then the routine SPQRY in PNT488 is called, which actually performs the poll. SPQRY will return upon receipt of an affirmative response or after the string of talk addresses has been exhausted, whichever occurs first. The commands sent by the P&T-488 while it is conducting the Serial Poll are echoed on the console. The string will appear as

?↑X_...↑Y

where the character ? means UNListen, ↑X is the command SPE (Serial Poll Enable), _ is the command UNTalk, the ellipsis (...) represents the Talk addresses that are sent by the P&T-488, and the ↑Y is SPD (Serial Poll Disable). If an affirmative response has been detected, DINK will print the Talk address of the device that responded affirmatively as well as the response, and then ask for the next function code. If no device responded affirmatively, DINK will print

```
No affirmative response to Serial Poll
Try another Serial Poll (Y/N)?
```

and then wait for the user to respond. If a string beginning with N is entered, DINK will ask for the next function code. If a string beginning with Y is entered, DINK will ask for another string of Talk addresses to be polled. It is important that only Talk addresses of devices which are currently connected to the 488 bus and capable of responding to a Serial Poll be entered in the string. The reason is that the P&T will send out the address and then listen for the addressed Talker to speak its poll response. If there is no Talker, there will never be a response, and the whole system will wait forever for that response.

Function R causes the routine SPSRQ in PNT488 to be called, which in turn asserts a true on the SRQ line and places the P&T-488 in the Service Request (SRQS) state. If the P&T-488 is not the Controller, the S-100 system will wait for an external Controller (i.e., some device other than the P&T-488) to assert Control and perform a Serial Poll. When the poll is made, the P&T-488 will respond affirmatively and then go into the Affirmative Poll Response (APRS) state. Then the user will be asked to select the next function.

If, on the other hand, the P&T-488 is the Controller, it will assert Control, then ask the user to enter a string of the Talk addresses of the devices to be Serial Polled. After the string has been entered the P&T-488 will poll each of these devices and then return when it has found the one requesting service or has finished polling all devices. The commands sent by the P&T-488 while it is conducting the Serial Poll are echoed on the console. The string will appear as

```
?↑X_...↑Y
```

where the character ? means UNListen, ↑X is the command SPE (Serial Poll Enable), _ is the command UNTalk, the ellipsis (...) represents the Talk addresses that are sent by the P&T-488, and the ↑Y is SPD (Serial Poll Disable). If the user had included the P&T-488's own Talk address in the string and no other device in the string before it has responded affirmatively to the poll, the P&T-488 will respond affirmatively to the poll and go into the Affirmative Poll Response (APRS) state, then return.

As in the case of function Q, it is important that only the Talk addresses of devices actually connected to the bus and capable of responding to a Serial Poll be placed in the poll string; otherwise the S-100 system will wait forever for the response of a non-existent device.

Function S will display the state of the P&T-488, the secondary Talk and Listen addresses and the state of the 488 bus lines. All values displayed are in Hex, and the user should refer to the function STATE for a description of the meaning of the various states. The value shown on the line labelled "Abbreviated State of P&T-488" is the value that the routine STATE returned in the accumulator.

The secondary addresses shown for the Talk and Listen functions are 7F Hex if the respective function has been addressed by the 488 Controller without a secondary address (single byte addressing). Otherwise the secondary addresses shown are the characters sent by the Controller as the secondary address when the Controller last addressed the Talk and Listen functions.

The state of the eight data lines and eight command lines of the 488 bus is also displayed. The values given are in Hex, which really has no particular meaning for the eight command lines. However, the order (weighting) of the command lines is shown on the same line as a handy reminder. The weights of the command lines are shown in the following table.

Line	Weight	Line	Weight
DAV	80H	ATN	8
NRFD	40H	SRQ	4
NDAC	20H	REN	2
IFC	10H	EOI	1

Function T sets up the P&T-488 as a Talker. The user is asked whether the END message (EOI line true) is to be sent with the last character of the talk string. The only responses allowed are strings beginning with Y or N. The user is then asked for the string that the P&T-488 is to speak. Then the routine TALK of PNT488 is called and the P&T-488 speaks the string on the bus. If there are no Listeners the P&T-488 recognizes this as an error and prints a message on the console informing the user that there are no Listeners on the 488 bus. Otherwise the whole string is said and then the user is asked for the next function code.

Function V causes the routine SPIDL in PNT488 to be called, which in turn puts the P&T-488 into the No Service Requested (NPRS) state. This is equivalent to the S-100 making the local message **rsv** (request service) false. The P&T-488 is also set to assert a passive false on the SRQ line. Then the routine returns and the user is asked for the next function code.

Special Considerations

The P&T-488 is heavily dependent upon the support software (in this case, PNT488) in order to communicate on the 488 bus. The S-100 system must execute one of the interface subroutines if the P&T-488 is to perform nearly any 488 bus function. This includes not only the "assertive" functions, such as Talk and Control, but the "responsive" functions, such as responding to a Serial Poll, being addressed as a Talker or Listener by the Controller, etc. The only 488 function that the P&T-488 can perform without any software support is respond to a Parallel Poll.

This limitation can create problems unless the user is aware of it and allows for it in his configuration of the 488 bus and how he uses the P&T-488. For instance, assume that some device other than the P&T-488 is the bus Controller and that it will perform a Parallel Poll periodically. The P&T-488 will respond to the poll properly, but the interface will lock up the 488 handshake function until the S-100 system releases it. This happens because the poll was done by an external Controller, so XATN was made true while the poll was performed. The P&T-488 responds to XATN true by asserting NRFD active true and by asserting all other command lines and all data lines passive false. The P&T-488 remains in this state until the S-100 system resets the XIFC bit in the ISR register. Since NRFD is active true, no handshake can proceed. The reason the P&T-488 behaves in this fashion is that if the external Controller wanted to issue commands (instead of do a Parallel Poll), it is necessary to keep it from saying anything until the S-100 system is ready to respond. The S-100 system indicates its readiness by resetting the XIFC bit in the ISR register of the P&T-488.

Another consequence of the need of the P&T-488 for software support in order to perform 488 bus functions is that something may happen on the 488 bus and the S-100 system will not find out about it until one of the PNT488 subroutines is called. For example, some device may assert an active true on the SRQ line, indicating that it wants service. The S-100 system will find out about it if any one of the routines TALK, CNTRL or LISTN are executed, but not otherwise. The P&T-488 interface card can be set up to issue an interrupt to the S-100 system upon this and other conditions, but most customers have stated very explicitly that they **do not** want an interrupt driven system. Thus the P&T-488 has been strapped to defeat interrupts, and the routines in PNT488 poll the P&T-488 to find out if anything interesting is happening.

There are several things which can happen which are not a direct response to the function code the user selects. For instance, if the Listen function is selected and an External Controller asserts Control, DINK will print a message on the console informing the user of this fact and will then call the routine XCTRL in PNT488. This routine will get the commands from the External Controller and will update the states of the various interface functions of the P&T-488 as necessary. When the External Controller releases control of the bus, XCTRL will return to DINK, which in turn will ask for the next function code. At this point the user should select the SHOW function (code = S) to find out how the state of the P&T-488 has been changed by the External Controller.

Another response the user may get is that DINK informs him that either the S-100 POC (Power-On Clear) or the S-100 Reset line has been (or is) true. Either of these conditions has the effect of putting the P&T-488 interface into its idle mode, which means that it has released all 488 data and control lines. The user should perform the Initialize (code = I) function to reset the P&T-488 to a known state.

PROGRAM LISTING

```
                ;
Ø1ØØ                    ORG     1ØØH
                ;
ØØ7F =          DELCHR  EQU     7FH     ;CHARACTER TO BE ECHOED UPON RECEIPT
                                        ; OF A DELETE CODE (DELETE AND BACKSPACE
                                        ; ARE THE MOST COMMON CHOICES)
                ;
ØØ7D =          CMDPT   EQU     7DH     ;PORT ADDR OF 488 COMMAND LINES
ØØ7E =          DATPT   EQU     7EH     ;PORT ADDR OF 488 DATA LINES
                ;
ØØ8Ø =          BUFSIZ  EQU     128     ;NUMBER OF BYTES IN INPUT BUFFER               M8229
                ;
8Ø26 =          ENTBL   EQU     8Ø26H   ;ADDRESS OF FIRST ENTRY IN PNT488 JUMP TABLE
                ;
8Ø26 =          INIT    EQU     ENTBL
8Ø29 =          TALK    EQU     ENTBL+Ø3H
8Ø2C =          LISTN   EQU     ENTBL+Ø6H
8Ø2F =          STADR   EQU     ENTBL+Ø9H
8Ø32 =          CNTRL   EQU     ENTBL+ØCH
8Ø35 =          GIM     EQU     ENTBL+ØFH
8Ø38 =          STATE   EQU     ENTBL+12H
8Ø3B =          XCTRL   EQU     ENTBL+15H
8Ø3E =          SPQRY   EQU     ENTBL+18H
8Ø41 =          SPSRQ   EQU     ENTBL+1BH
8Ø44 =          SPREL   EQU     ENTBL+1EH
8Ø47 =          PPQRY   EQU     ENTBL+21H
8Ø4A =          PPREQ   EQU     ENTBL+24H
8Ø4D =          PPREL   EQU     ENTBL+27H
8Ø5Ø =          PPIDL   EQU     ENTBL+2AH
                ;
                ;       EQUATES FOR CP/M CBIOS ROUTINES
                ;
DAØ9 =          CONIN   EQU     ØDAØ9H  ;CONSOLE INPUT ROUTINE                         M8229
DAØC =          CONOUT  EQU     ØDAØCH  ;CONSOLE OUTPUT ROUTINE                        M8229
                ;
Ø1ØØ 31ECØ8     START:  LXI     SP,STAK ;INITIALIZE THE STACK POINTER
Ø1Ø3 Ø1D6Ø5             LXI     B,IDMS  ;PRINT ID MESSAGE                              M8229
Ø1Ø6 CD67Ø4             CALL    MSG     ;                                              M8229
Ø1Ø9 CD79Ø1             CALL    ADRSET  ;SET THE LISTEN, TALK ADDR, ETC                M8229
Ø1ØC Ø6Ø1               MVI     B,1     ;CLEAR 488 INTERFACE BUT DO NOT SEND IFC
Ø1ØE CD268Ø             CALL    INIT
                ;
                ;       GET FUNCTION TO BE PERFORMED
                ;
Ø111 31ECØ8     GETFN:  LXI     SP,STAK ;RE-INITIALIZE STACK POINTER (STACK WILL BE LEFT IN
                                        ;  DISARRAY IF 'ATN' IS MADE TRUE WHILE TALKING OR
                                        ;  LISTENING)                                  M1Ø2Ø
Ø114 97                 SUB     A       ;CLEAR ECHO FLAG SO THAT UNLESS THE FLAG
Ø115 3228Ø8             STA     ECHO    ;  IS SET LATER, EACH CHAR COMMUNICATED
                                        ;  ON THE 488 BUS IS NOT ECHOED TO THE
                                        ;  CONSOLE
Ø118 Ø1AFØ5             LXI     B,FCNMS ;SEND "FUNCTION?" MESSAGE
Ø11B CD67Ø4             CALL    MSG
Ø11E CDECØ3             CALL    FILBFR  ;GET OPERATOR'S RESPONSE
Ø121 CA11Ø1             JZ      GETFN   ;..NOTHING IN BUFFER                           M8229
Ø124 3A2AØ8             LDA     BUFBEG  ;LOOK AT FIRST CHARACTER
Ø127 3229Ø8             STA     FCN     ;SAVE IT FOR LATER
Ø12A FE41               CPI     'A'
Ø12C CA98Ø1             JZ      SETADR  ;..SET NEW P&T-488 ADDRESSES
Ø12F FE43               CPI     'C'
Ø131 CA9EØ1             JZ      CONTRL  ;..CONTROLLER
Ø134 FE47               CPI     'G'
Ø136 CAA7Ø1             JZ      GIMSET  ;..SET GIM LINES
Ø139 FE49               CPI     'I'
Ø13B CABØØ1             JZ      INITL   ;..INITIALIZE
Ø13E FE4C               CPI     'L'
Ø14Ø CAC6Ø1             JZ      LSN     ;..LISTEN
Ø143 FE4D               CPI     'M'
Ø145 CAF6Ø1             JZ      PIDL    ;..PUT PP IN IDLE STATE
Ø148 FE4E               CPI     'N'
Ø14A CA11Ø2             JZ      PNSET   ;..SET IST=Ø
Ø14D FE4F               CPI     'O'
```

```
Ø14F CAØBØ2            JZ      PSET     ;..SET IST=1
Ø152 FE5Ø              CPI     'P'
Ø154 CAFCØ1            JZ      PPOLL    ;..DO A PARALLEL POLL
Ø157 FE51              CPI     'Q'
Ø159 CAA8Ø3            JZ      QRY      ;..DO A SERIAL POLL QUERY
Ø15C FE52              CPI     'R'
Ø15E CA17Ø2            JZ      REQ      ;..DO A SERVICE REQUEST
Ø161 FE53              CPI     'S'
Ø163 CA2ØØ2            JZ      SHO      ;..SHO THE STATE OF THE P&T-488
Ø166 FE54              CPI     'T'
Ø168 CAA7Ø2            JZ      TALKR    ;..TALK
Ø16B FE56              CPI     'V'
Ø16D CAA1Ø2            JZ      SREL     ;..RELEASE SRQ LINE
Ø17Ø Ø1FBØ4            LXI     B,BADMS  ;PRINT "INVALID FCN" MESSAGE
Ø173 CD72Ø4            CALL    MSGCR
Ø176 C311Ø1            JMP     GETFN    ;GET FUNCTION AGAIN
                ;
Ø179 Ø182Ø4     ADRSET: LXI    B,ADRMS  ;SEND "GET ADDRESSES" MESSAGE                M8229
Ø17C CD67Ø4            CALL    MSG
Ø17F CDECØ3            CALL    FILBFR   ;GET RESPONSE AND PUT IN BUFFER
Ø182 78                MOV     A,B      ;MAKE SURE THAT THERE ARE AT LEAST 5         M8229
Ø183 FEØ5              CPI     5        ;  CHARACTERS IN THE RESPONSE                M8229
Ø185 F291Ø1            JP      SET1     ;..5 OR MORE CHARS                           M8229
Ø188 Ø1C3Ø5            LXI     B,FEWMS  ;PRINT TOO FEW CHARS IN BUFFER MESSAGE       M8229
Ø18B CD67Ø4            CALL    MSG      ;                                            M8229
Ø18E C379Ø1            JMP     ADRSET   ;AND GET THE INFO AGAIN                      M8229
                ;
Ø191 212AØ8     SET1:  LXI     H,BUFBEG ;SET UP P&T 488 LISTEN, TALK ADDRESSES       M8229
Ø194 CD2F8Ø            CALL    STADR    ;PERFORM THE FUNCTION
Ø197 C9                RET              ;                                            M8229
                ;
Ø198 CD79Ø1     SETADR: CALL   ADRSET   ;SET THE LISTEN, TALK ADDR, ETC              M8229
Ø19B C311Ø1            JMP     GETFN
                ;
Ø19E CDF6Ø2     CONTRL: CALL   GETSTR   ;FILL BUFFER AND SET POINTERS
Ø1A1 CD328Ø            CALL    CNTRL    ;PERFORM THE FUNCTION
Ø1A4 C311Ø1            JMP     GETFN    ;GET ANOTHER FUNCTION FROM OPERATOR
                ;
Ø1A7 CDDDØ2     GIMSET: CALL   GETCHR   ;GET THE CHARACTER
Ø1AA CD358Ø            CALL    GIM
Ø1AD C311Ø1            JMP     GETFN
                ;
Ø1BØ Ø1E8Ø5     INITL: LXI     B,IFCMS  ;ASK IF IFC TO BE SENT
Ø1B3 CD67Ø4            CALL    MSG
Ø1B6 CDC1Ø2            CALL    YESNO    ;GET RESPONSE (ZERO FLAG SET IF NO)
Ø1B9 Ø6Ø1              MVI     B,1      ;SET UP FOR NO IFC
Ø1BB CACØØ1            JZ      NOIFC    ;..NO, SO DO NOT SEND IFC
Ø1BE Ø6ØØ              MVI     B,Ø      ;..YES, SO SEND IFC
Ø1CØ CD268Ø     NOIFC: CALL    INIT
Ø1C3 C311Ø1            JMP     GETFN
                ;
Ø1C6 3EFF       LSN:   MVI     A,ØFFH   ;SET ECHO FLAG SO THAT EACH CHARACTER IS
Ø1C8 3228Ø8            STA     ECHO     ;  SHOWN ON THE CONSOLE
Ø1CB Ø18EØ5            LXI     B,EOSMS  ;PRINT "STOP ON EOS?"                        M8229
Ø1CE CD67Ø4            CALL    MSG      ;                                            M8229
Ø1D1 CDC1Ø2            CALL    YESNO    ;GET THE RESPONSE                            M8229
Ø1D4 C2E8Ø1            JNZ     LSN1     ;..STOP ON EOS BYTE                          M8229
Ø1D7 3EØØ              MVI     A,Ø      ;NON-BUFFERED LISTENER, IGNORE EOS BYTE
Ø1D9 Ø116Ø3            LXI     B,JTBL   ;BC POINT TO USER JUMP TABLE
Ø1DC CD2C8Ø            CALL    LISTN
Ø1DF Ø187Ø5            LXI     B,ENDMS  ;SHOW THAT AN END MESSAGE HAS BEEN RECEIVED
Ø1E2 CD72Ø4            CALL    MSGCR
Ø1E5 C311Ø1            JMP     GETFN
                ;
Ø1E8 3EØ2       LSN1:  MVI     A,2      ;NON-BUFFERED LISTENER, STOP ON EOS BYTE     M8229
Ø1EA Ø116Ø3            LXI     B,JTBL   ;POINT BC TO USER JUMP TABLE                 M8229
Ø1ED CD2C8Ø            CALL    LISTN    ;                                            M8229
Ø1FØ CD75Ø4            CALL    CRLF     ;                                            M8229
Ø1F3 C311Ø1            JMP     GETFN    ;                                            M8229
                ;
Ø1F6 CD5Ø8Ø     PIDL:  CALL    PPIDL    ;PUT PP IN IDLE STATE
Ø1F9 C311Ø1            JMP     GETFN
```

```
                  ;
Ø1FC CD478Ø       PPOLL:  CALL    PPQRY   ;PERFORM A PARALLEL POLL
Ø1FF CD46Ø4               CALL    HEXO    ;PRINT THE RESPONSE
Ø2Ø2 Ø175Ø6               LXI     B,PPMS
Ø2Ø5 CD72Ø4               CALL    MSGCR   ;AND ID
Ø2Ø8 C311Ø1               JMP     GETFN

                  ;
Ø2ØB CD4A8Ø       PSET:   CALL    PPREQ   ;SET "IST" TRUE AND UPDATE PARALLEL POLL
Ø2ØE C311Ø1               JMP     GETFN   ;  RESPONSE REGISTER
                  ;
Ø211 CD4D8Ø       PNSET:  CALL    PPREL   ;SET "IST" FALSE AND UPDATE PARALLEL POLL
Ø214 C311Ø1               JMP     GETFN   ;  RESPONSE REGISTER
                  ;
Ø217 Ø116Ø3       REQ:    LXI     B,JTBL  ;POINT TO USER JUMP TABLE
Ø21A CD418Ø               CALL    SPSRQ   ;PERFORM THE FUNCTION
Ø21D C311Ø1               JMP     GETFN
                  ;
Ø22Ø CD388Ø       SHO:    CALL    STATE   ;GET THE STATE OF THE P&T-488
Ø223 CD46Ø4               CALL    HEXO    ;PRINT VALUE IN REG A IN HEX
Ø226 Ø1D8Ø6               LXI     B,SØMSG ;PRINT "ABBR. STATE" MESSAGE
Ø229 CD72Ø4               CALL    MSGCR
Ø22C 7E                   MOV     A,M
Ø22D CD46Ø4               CALL    HEXO    ;PRINT HEX VALUE OF THE STATE BYTE
Ø23Ø Ø1F5Ø6               LXI     B,S1MSG
Ø233 CD72Ø4               CALL    MSGCR
Ø236 23                   INX     H       ;POINT TO THE NEXT STATE BYTE
Ø237 7E                   MOV     A,M
Ø238 CD46Ø4               CALL    HEXO    ;PRINT HEX VALUE OF THE STATE BYTE
Ø23B Ø1Ø5Ø7               LXI     B,S2MSG
Ø23E CD72Ø4               CALL    MSGCR
Ø241 23                   INX     H       ;POINT TO THE NEXT STATE BYTE
Ø242 7E                   MOV     A,M
Ø243 CD46Ø4               CALL    HEXO    ;PRINT HEX VALUE OF THE STATE BYTE
Ø246 Ø117Ø7               LXI     B,S3MSG
Ø249 CD72Ø4               CALL    MSGCR
Ø24C 23                   INX     H       ;POINT TO THE NEXT STATE BYTE
Ø24D 7E                   MOV     A,M
Ø24E CD46Ø4               CALL    HEXO    ;PRINT HEX VALUE OF THE STATE BYTE
Ø251 Ø132Ø7               LXI     B,S4MSG
Ø254 CD72Ø4               CALL    MSGCR
Ø257 23                   INX     H       ;POINT TO THE NEXT STATE BYTE
Ø258 7E                   MOV     A,M
Ø259 CD46Ø4               CALL    HEXO    ;PRINT HEX VALUE OF THE STATE BYTE
Ø25C Ø14AØ7               LXI     B,S5MSG
Ø25F CD72Ø4               CALL    MSGCR
Ø262 23                   INX     H       ;POINT TO THE NEXT STATE BYTE
Ø263 7E                   MOV     A,M
Ø264 CD46Ø4               CALL    HEXO    ;PRINT HEX VALUE OF THE STATE BYTE
Ø267 Ø163Ø7               LXI     B,S6MSG
Ø26A CD72Ø4               CALL    MSGCR
Ø26D 23                   INX     H       ;POINT TO LISTEN SECONDARY ADDRESS
Ø26E 7E                   MOV     A,M
Ø26F CD46Ø4               CALL    HEXO
Ø272 Ø1Ø9Ø6               LXI     B,LSMSG
Ø275 CD72Ø4               CALL    MSGCR
Ø278 23                   INX     H       ;POINT TO TALK SECONDARY ADDRESS
Ø279 7E                   MOV     A,M
Ø27A CD46Ø4               CALL    HEXO
Ø27D Ø1FCØ7               LXI     B,TSMSG
Ø28Ø CD72Ø4               CALL    MSGCR
Ø283 DB7D                 IN      CMDPT   ;SHOW WHAT'S ON THE 488 COMMAND LINES
Ø285 2F                   CMA
Ø286 CD46Ø4               CALL    HEXO
Ø289 Ø137Ø5               LXI     B,CLMS
Ø28C CD72Ø4               CALL    MSGCR
Ø28F DB7E                 IN      DATPT   ;AND THEN WHAT'S ON THE 488 DATA LINES
Ø291 2F                   CMA
Ø292 CD46Ø4               CALL    HEXO
Ø295 Ø16CØ5               LXI     B,DLMS
Ø298 CD72Ø4               CALL    MSGCR
Ø29B CD75Ø4               CALL    CRLF    ;PUT IN AN EXTRA CARRIAGE RETURN-LINE FEED
Ø29E C311Ø1               JMP     GETFN   ;GET ANOTHER FUNCTION
```

```
                    ;
Ø2A1 CD448Ø         SREL:   CALL    SPREL   ;RELEASE SRQ, PUT SR FCN IN NPRS
Ø2A4 C311Ø1                 JMP     GETFN   ;GET ANOTHER FUNCTION
                    ;
Ø2A7 Ø1ABØ7         TALKR:  LXI     B,TLKMS ;PRINT "SEND END WITH LAST CHAR"
Ø2AA CD67Ø4                 CALL    MSG
Ø2AD CDC1Ø2                 CALL    YESNO
Ø2BØ 3EØØ                   MVI     A,Ø     ;SET FLAG FOR NO END
Ø2B2 CAB6Ø2                 JZ      NOEOI
Ø2B5 3C                     INR     A       ;SET FLAG FOR END
Ø2B6 F5             NOEOI:  PUSH    PSW     ;SAVE END FLAG
Ø2B7 CDF6Ø2                 CALL    GETSTR  ;FILL BUFFER AND SET POINTERS
Ø2BA F1                     POP     PSW     ;GET END FLAG AGAIN
Ø2BB CD298Ø                 CALL    TALK    ;PERFORM THE FUNCTION
Ø2BE C311Ø1                 JMP     GETFN   ;GET ANOTHER FUNCTION FROM THE OPERATOR
                    ;
Ø2C1 CDECØ3         YESNO:  CALL    FILBFR
Ø2C4 CAD2Ø2                 JZ      YESN1   ;..BUFFER EMPTY - INVALID RESPONSE           M8229
Ø2C7 3A2AØ8                 LDA     BUFBEG  ;GET THE FIRST CHARACTER
Ø2CA FE59                   CPI     'Y'     ;IS IT YES?
Ø2CC CADBØ2                 JZ      COK     ;..CHARACTER OK
Ø2CF FE4E                   CPI     'N'     ;IS IT NO?
Ø2D1 C8                     RZ              ;CHARACTER OK
Ø2D2 Ø12DØ6         YESN1:  LXI     B,NOGUD ;INVALID RESPONSE                            M8229
Ø2D5 CD72Ø4                 CALL    MSGCR
Ø2D8 C3C1Ø2                 JMP     YESNO   ;TRY AGAIN
                    ;
Ø2DB B7             COK:    ORA     A       ;UNSET THE ZERO FLAG
Ø2DC C9                     RET
                    ;
Ø2DD Ø125Ø5         GETCHR: LXI     B,CHRMS ;PRINT CHARACTER PROMPT
Ø2EØ CD67Ø4                 CALL    MSG
Ø2E3 CDECØ3                 CALL    FILBFR  ;GET THE CHARACTER
Ø2E6 C2F2Ø2                 JNZ     GETCH1  ;..AT LEAST ONE CHARACTER IS IN THE BUFFER   M8229
Ø2E9 Ø1C3Ø5                 LXI     B,FEWMS ;POINT TO 'TOO FEW' MSG                      M8229
Ø2EC CD72Ø4                 CALL    MSGCR   ;  THEN PRINT IT                             M8229
Ø2EF C3DDØ2                 JMP     GETCHR  ;AND GET INFO FROM USER AGAIN                M8229
                    ;
Ø2F2 3A2AØ8         GETCH1: LDA     BUFBEG  ;PUT FIRST CHARACTER IN REG A                M8229
Ø2F5 C9                     RET
                    ;
Ø2F6 Ø19CØ7         GETSTR: LXI     B,STRMS ;PRINT STRING PROMPT
Ø2F9 CD72Ø4                 CALL    MSGCR
Ø2FC CDECØ3                 CALL    FILBFR  ;GET A CHAR STRING FROM THE OPERATOR
Ø2FF C2ØBØ3                 JNZ     GETS1   ;..AT LEAST ONE CHARACTER IS IN THE BUFFER   M8229
Ø3Ø2 Ø1C3Ø5                 LXI     B,FEWMS ;POINT TO 'TOO FEW' MSG                      M8229
Ø3Ø5 CD72Ø4                 CALL    MSGCR   ;  THEN PRINT IT                             M8229
Ø3Ø8 C3F6Ø2                 JMP     GETSTR  ;AND GET INFO FROM USER AGAIN                M8229
                    ;
Ø3ØB 2AAAØ8         GETS1:  LHLD    BUFPTR  ;PUT ADDR OF LAST VALID CHAR IN HL           M8229
Ø3ØE EB                     XCHG            ;PUT ADDR OF LAST VALID CHAR IN DE
Ø3ØF 212AØ8                 LXI     H,BUFBEG ;LOAD HL WITH ADDRESS OF FIRST CHAR
Ø312 Ø116Ø3                 LXI     B,JTBL  ;LOAD BC WITH BEGINNING ADDR OF JUMP TABLE
Ø315 C9                     RET
                    ;
                    ;
                    ;       USER-SUPPLIED JUMP TABLE
                    ;
Ø316 C331Ø3         JTBL:   JMP     TRGR
Ø319 C338Ø3                 JMP     DVCL
Ø31C C33FØ3                 JMP     BFL
Ø31F C348Ø3                 JMP     ICLR
Ø322 C351Ø3                 JMP     BRK
Ø325 C379Ø3                 JMP     NLS
Ø328 C3A2Ø3                 JMP     SREQ
Ø32B C382Ø3                 JMP     POCRST
Ø32E C38BØ3                 JMP     XTN
                    ;
Ø331 Ø1CFØ7         TRGR:   LXI     B,TMS   ;PRINT TRIGGER MESSAGE
Ø334 CD72Ø4                 CALL    MSGCR
Ø337 C9                     RET
                    ;
```

```
Ø338 Ø17BØ5     DVCL:   LXI     B,DVMS  ;PRINT DEVICE CLEAR MESSAGE
Ø33B CD72Ø4             CALL    MSGCR
Ø33E C9                 RET
                ;
Ø33F Ø113Ø5     BFL:    LXI     B,BMS   ;WE SHOULD NEVER REACH THIS POINT, BUT
Ø342 CD72Ø4             CALL    MSGCR   ;  IF WE DO, PRINT MESSAGE
Ø345 C311Ø1             JMP     GETFN
                ;
Ø348 Ø1F8Ø5     ICLR:   LXI     B,IFMS  ;PRINT INTERFACE CLEAR MESSAGE
Ø34B CD72Ø4             CALL    MSGCR
Ø34E C311Ø1             JMP     GETFN   ;ASK FOR NEW FUNCTION
                ;
Ø351 47         BRK:    MOV     B,A     ;SAVE LAST CHAR COMMUNICATED ON 488 BUS
Ø352 3A28Ø8             LDA     ECHO    ;LOOK AT THE ECHO FLAG
Ø355 B7                 ORA     A
Ø356 78                 MOV     A,B     ;GET THE LAST CHAR AGAIN
Ø357 C8                 RZ              ;..ECHO FCN NOT ENABLED, SO DON'T PRINT
                                        ;  THE CHARACTER
Ø358 FE2Ø               CPI     2ØH     ;CONTROL CHARACTER?
Ø35A D275Ø3             JNC     NOTCC   ;..NO
Ø35D FEØ9               CPI     Ø9H     ;TAB?
Ø35F CA75Ø3             JZ      NOTCC   ;..YES, SO PRINT AS IS
Ø362 FEØA               CPI     ØAH     ;LINE FEED?
Ø364 CA75Ø3             JZ      NOTCC
Ø367 FEØD               CPI     ØDH     ;CARRIAGE RETURN?
Ø369 CA75Ø3             JZ      NOTCC
Ø36C F64Ø               ORI     4ØH     ;MAKE THE CHAR INTO A PRINTING CHAR
Ø36E F5                 PUSH    PSW     ;SAVE THE CHARACTER
Ø36F 3E5E               MVI     A,'↑'   ;PRINT UPARROW TO FLAG IT AS A
Ø371 CDECØ8             CALL    PRT     ;  CONTROL CHARACTER
Ø374 F1                 POP     PSW
Ø375 CDECØ8     NOTCC:  CALL    PRT     ;PRINT THE CHARACTER
Ø378 C9                 RET
                ;
Ø379 Ø122Ø6     NLS:    LXI     B,NLMS  ;PRINT NO LISTENER MESSAGE
Ø37C CD72Ø4             CALL    MSGCR
Ø37F C311Ø1             JMP     GETFN   ;ASK FOR NEW FUNCTION
                ;
Ø382 Ø165Ø6     POCRST: LXI     B,POCMS ;PRINT S-1ØØ RESET/POWER-ON CLEAR
Ø385 CD72Ø4             CALL    MSGCR
Ø388 C311Ø1             JMP     GETFN   ;ASK FOR NEW FUNCTION
                ;
Ø38B Ø113Ø8     XTN:    LXI     B,XTNMS ;PRINT EXTERNAL CONTROLLER MESSAGE
Ø38E CD72Ø4             CALL    MSGCR
Ø391 3EFF               MVI     A,ØFFH  ;SET ECHO FLAG SO THAT THE CONTROLLER'S
Ø393 3228Ø8             STA     ECHO    ;  COMMANDS ARE SHOWN ON THE CONSOLE
Ø396 Ø116Ø3             LXI     B,JTBL  ;POINT TO USER JUMP TABLE
Ø399 CD3B8Ø             CALL    XCTRL   ;DO WHATEVER THE CONTROLLER SAYS
Ø39C CD75Ø4             CALL    CRLF
Ø39F C311Ø1             JMP     GETFN   ;ASK FOR NEW FUNCTION
                ;
Ø3A2 Ø179Ø7     SREQ:   LXI     B,SRQMS ;PRINT "DEVICE REQUESTING SERVICE" MSG
Ø3A5 CD72Ø4             CALL    MSGCR
Ø3A8 3EFF       QRY:    MVI     A,ØFFH  ;SET ECHO SO THAT THE SERIAL POLL IS
Ø3AA 3228Ø8             STA     ECHO    ;  SHOWN ON THE CONSOLE
Ø3AD CDF6Ø2             CALL    GETSTR  ;GET STRING OF 488 DEVICES TO BE POLLED

Ø3BØ CD3E8Ø             CALL    SPQRY   ;FIND OUT WHICH DEVICE WANTS SERVICE
Ø3B3 CD75Ø4             CALL    CRLF    ;TERMINATE THE ECHOED POLL WITH CRLF
Ø3B6 B7                 ORA     A       ;SEE IF ANY AFFIRMATIVE RESPONSE
Ø3B7 CACDØ3             JZ      AFIRM   ;..YES
Ø3BA Ø13EØ6             LXI     B,NORSP ;PRINT "NO RESPONDING DEVICE"
Ø3BD CD72Ø4             CALL    MSGCR
Ø3CØ Ø1DDØ7             LXI     B,TRYAGN        ;ASK IF WANT TO TRY AGAIN
Ø3C3 CD72Ø4             CALL    MSGCR
Ø3C6 CDC1Ø2             CALL    YESNO
Ø3C9 C8                 RZ
Ø3CA C3A2Ø3             JMP     SREQ    ;..YES, SO REDO SERIAL POLL
                ;
Ø3CD C5         AFIRM:  PUSH    B       ;SAVE RESPONSE BYTE
Ø3CE E5                 PUSH    H       ;SAVE ADDR OF RESPONDING DEVICES TALK ADDR
Ø3CF Ø1B3Ø6             LXI     B,RSPMS ;PRINT "REQUESTING DEVICE IS "
```

```
Ø3D2 CD67Ø4             CALL    MSG
Ø3D5 E1                 POP     H       ;GET ADDR OF TALK ADDR AGAIN
Ø3D6 7E                 MOV     A,M     ;PUT DEVICE'S ADDRESS IN A REGISTER
Ø3D7 CDECØ8             CALL    PRT     ;PRINT THE DEVICE'S TALK ADDR
Ø3DA CD75Ø4             CALL    CRLF    ;TERMINATE WITH A NEW LINE
Ø3DD Ø191Ø6             LXI     B,RSBMS ;PRINT RESPONSE BYTE MESSAGE
Ø3EØ CD67Ø4             CALL    MSG
Ø3E3 C1                 POP     B       ;PRINT VALUE OF RESPONSE BYTE IN HEX
Ø3E4 78                 MOV     A,B
Ø3E5 CD46Ø4             CALL    HEXO
Ø3E8 CD75Ø4             CALL    CRLF    ;FINISH WITH CRLF
Ø3EB C9                 RET
                ;
                FILBFR:
Ø3EC Ø6Ø1       FIL1:   MVI     B,1     ;RESET CHARACTER COUNT TO ZERO
Ø3EE 212AØ8             LXI     H,BUFBEG ;  AND POINTER TO BEGINNING OF BUFFER
Ø3F1 CDFBØ8     FIL2:   CALL    KBIN    ;GET A CHARACTER FROM THE KEYBOARD
Ø3F4 77                 MOV     M,A     ;PUT IT INTO THE BUFFER
Ø3F5 FEØD               CPI     ØDH     ;CARRIAGE RETURN?
Ø3F7 CA2DØ4             JZ      FILXIT  ;..YES, SO QUIT ALREADY
Ø3FA FE18               CPI     18H     ;CONTROL X (CANCEL)?
Ø3FC C2ØAØ4             JNZ     NOTX
Ø3FF 3E23               MVI     A,'#'   ;PRINT OCTOTHORPE AS CANCEL CHARACTER
Ø4Ø1 CDECØ8             CALL    PRT
Ø4Ø4 CD75Ø4             CALL    CRLF    ;DO A CARRIAGE RETURN AND LINE FEED
Ø4Ø7 C3ECØ3             JMP     FIL1    ;RESTART BUFFER FILL PROCESS
                ;
Ø4ØA FE7F       NOTX:   CPI     7FH     ;DELETE?
Ø4ØC C221Ø4             JNZ     NOTD
Ø4ØF 3E7F               MVI     A,DELCHR        ;ECHO THE DELETE CHARACTER
Ø411 CDECØ8             CALL    PRT
Ø414 2B                 DCX     H       ;DECREMENT BUFFER POINTER (TO DELETE CHAR)
Ø415 Ø5                 DCR     B       ;DECREMENT CHARACTER COUNT
Ø416 C2F1Ø3             JNZ     FIL2    ;GET NEXT CHAR
Ø419 3EØ7               MVI     A,7     ;DELETED MORE CHARS THAN IN BUFFER
                                        ;  SO RING BELL
Ø41B CDECØ8             CALL    PRT
Ø41E C3ECØ3             JMP     FIL1    ;RE-START BUFFER FILL ROUTINE
                ;
Ø421 FE1B       NOTD:   CPI     1BH     ;ESCAPE?
Ø423 C236Ø4             JNZ     NESC    ;..NO
Ø426 CDFBØ8             CALL    KBIN    ;GET ANOTHER CHARACTER AND PUT IT IN            M8229
Ø429 77                 MOV     M,A     ;  THE BUFFER IN PLACE OF THE ESCAPE            M8229
Ø42A C336Ø4             JMP     NESC    ;  WITHOUT REGARD TO WHAT THE CHAR IS           M8229
                ;
Ø42D 2B         FILXIT: DCX     H       ;POINT TO LAST VALID CHARACTER
Ø42E 22AAØ8             SHLD    BUFPTR  ;UPDATE BUFFER POINTER
Ø431 CD75Ø4             CALL    CRLF    ;OUTPUT A CARRIAGE RETURN AND LINE FEED         M8229
Ø434 Ø5                 DCR     B       ;SET ZERO FLAG IF BUFFER EMPTY                  M8229
Ø435 C9                 RET
                ;
Ø436 23         NESC:   INX     H       ;INCREMENT BUFFER POINTER
Ø437 4F                 MOV     C,A     ;SAVE CHARACTER                                 M8229
Ø438 Ø4                 INR     B       ;INCREMENT CHARACTER COUNT
Ø439 3E8Ø               MVI     A,BUFSIZ ;SEE IF BUFFER OVERFLOWED                      M8229
Ø43B B8                 CMP     B       ;                                               M8229
Ø43C 79                 MOV     A,C     ;GET THE CHARACTER AGAIN                        M8229
Ø43D CA2DØ4             JZ      FILXIT  ;..BUFFER FULL, SO RETURN TO CALLER
Ø44Ø CDECØ8             CALL    PRT     ;ECHO THE CHARACTER ON THE CONSOLE
Ø443 C3F1Ø3             JMP     FIL2    ;GET NEXT CHARACTER
                ;
Ø446 F5         HEXO:   PUSH    PSW     ;SAVE THE BYTE TO BE PRINTED IN HEX
Ø447 ØF                 RRC             ;GET HIGH NIBBLE INTO LOW NIBBLE
Ø448 ØF                 RRC
Ø449 ØF                 RRC
Ø44A ØF                 RRC
Ø44B CD58Ø4             CALL    HEXL    ;PRINT THE NIBBLE (NOW LOW NIBBLE)
Ø44E F1                 POP     PSW     ;GET THE BYTE AGAIN
Ø44F CD58Ø4             CALL    HEXL    ;PRINT THE LOW NIBBLE
Ø452 3E2Ø               MVI     A,' '   ;PRINT A SPACE
Ø454 CDECØ8             CALL    PRT
Ø457 C9                 RET
```

```
                ;
Ø458 E6ØF       HEXL:    ANI     ØFH       ;STRIP HIGH NIBBLE
Ø45A F63Ø                ORI     3ØH       ;CONVERT TO PRINTING CHARACTERS
Ø45C FE3A                CPI     ':'       ;SEE IF VALUE GREATER THAN 9
Ø45E DA63Ø4              JC      NUM       ;..NO
Ø461 C6Ø7                ADI     7         ;..YES, SO ADD OFFSET TO GET A-F
Ø463 CDECØ8     NUM:     CALL    PRT       ;PRINT THE CHARACTER
Ø466 C9                  RET
                ;
Ø467 ØA         MSG:     LDAX    B
Ø468 CDECØ8              CALL    PRT       ;PRINT MESSAGE
Ø46B E68Ø                ANI     8ØH       ;SEE IF PARITY SET
Ø46D Ø3                  INX     B
Ø46E CA67Ø4              JZ      MSG       ;..NO, SO PRINT SOME MORE
Ø471 C9                  RET
                ;
Ø472 CD67Ø4     MSGCR:   CALL    MSG       ;PRINT THE MESSAGE, TERMINATE WITH CRLF
Ø475 F5         CRLF:    PUSH    PSW       ;PRESERVE ALL REGISTERS
Ø476 3EØD                MVI     A,ØDH     ;OUTPUT A CARRIAGE RETURN
Ø478 CDECØ8              CALL    PRT
Ø47B 3EØA                MVI     A,ØAH     ;THEN A LINE FEED
Ø47D CDECØ8              CALL    PRT
Ø48Ø F1                  POP     PSW       ;RESTORE ALL REGISTERS
Ø481 C9                  RET
                ;
                ;
Ø482 ØDØA456E74ADRMS:    DB      ØDH, ØAH,'Enter P&T-488 Listen and Talk addresses,'
Ø4AC 2Ø5Ø617261          DB      ' Parallel Poll response',ØDH,ØAH
Ø4C5 5365726961          DB      'Serial Poll status and the End-of-String '
Ø4EE 28454F5329          DB      '(EOS) bytes.',ØAØH
Ø4FB ØDØA494E56BADMS:    DB      ØDH, ØAH, 'INVALID FUNCTION CODE',ØAØH
Ø513 4C49535445BMS:      DB      'LISTEN BUFFER FUL', ØCCH
Ø525 456E746572CHRMS:    DB      'Enter a character',ØAØH
Ø537 343838204ЗCLMS:     DB      '488 Control lines:DAV NRFD NDAC IFC  ATN SRQ REN EOI',ØAØH
Ø56C 343838Ø44DLMS:      DB      '488 Data lines',ØAØH
Ø57B 4445564943DVMS:     DB      'DEVICE CLEA', ØD2H
Ø587 ØDØA3C454EENDMS:    DB      ØDH, ØAH,'<END',ØBEH
Ø58E 5265747572EOSMS:    DB      'Return upon receipt of EOS byte?',ØAØH ;                  M8229
Ø5AF 456E746572FCNMS:    DB      'Enter function code',ØAØH
Ø5C3 546F6F2Ø66FEWMS:    DB      'Too few characters',ØAØH                ;                 M8229
Ø5D6 ØDØA44494EIDMS:     DB      ØDH, ØAH,'DINK   1-2-8Ø',ØDH,8AH          ;                 M1Ø2Ø
Ø5E8 53656E642ØIFCMS:    DB      'Send IFC (Y/N)?',ØAØH
Ø5F8 ØDØA494E54IFMS:     DB      ØDH, ØAH,'INTERFACE CLEA', ØD2H
Ø6Ø9 4C69737465LSMSG:    DB      'Listen Secondary Address',ØAØH
Ø622 4E4F2Ø4C49NLMS:     DB      'NO LISTENE', ØD2H
Ø62D ØDØA592Ø6FNOGUD:    DB      ØDH,ØAH,'Y or N ONLY!!!',ØAØH
Ø63E 4E6F2Ø6166NORSP:    DB      'No affirmative response to Serial Poll',ØAØH
Ø665 ØDØA5Ø4F43POCMS:    DB      ØDH,ØAH,'POC/RESET TRU', ØC5H
Ø675 5Ø6172616CPPMS:     DB      'Parallel Poll Response byte',ØAØH
Ø691 5468652Ø76RSBMS:    DB      'The value of the response byte is',ØAØH
Ø6B3 5468652Ø34RSPMS:    DB      'The 488 device requesting service is',ØAØH
Ø6D8 4162627265SØMSG:    DB      'Abbreviated State of P&T-488',ØAØH
Ø6F5 54616C6B2ØS1MSG:    DB      'Talk State byte',ØAØH
Ø7Ø5 4C69737465S2MSG:    DB      'Listen State byte',ØAØH
Ø717 5365727669S3MSG:    DB      'Service Request State byte',ØAØH
Ø732 52656D6F74S4MSG:    DB      'Remote-Local State byte',ØAØH
Ø74A 5Ø6172616CS5MSG:    DB      'Parallel Poll State byte',ØAØH
Ø763 436F6E7472S6MSG:    DB      'Controller State byte',ØAØH
Ø779 412Ø343838SRQMS:    DB      'A 488 device is requesting service',ØAØH
Ø79C 456E746572STRMS:    DB      'Enter a string',ØAØH
Ø7AB 53656E642ØTLKMS:    DB      'Send END with last character (Y/N)?', ØAØH
Ø7CF 4445564943TMS:      DB      'DEVICE TRIGGE', ØD2H
Ø7DD 5472792Ø61TRYAGN:   DB      'Try another Serial Poll (Y/N)?',ØAØH
Ø7FC 54616C6B2ØTSMSG:    DB      'Talk Secondary Address',ØAØH
Ø813 ØDØA455854XTNMS:    DB      ØDH,ØAH,'EXTERNAL CONTROLLE',ØD2H
                ;
Ø828 ØØ         ECHO:    DB      Ø         ;ECHO FLAG.  IF Ø DO NOT PRINT CHAR EACH
                                           ;  TIME BRK IS CALLED
Ø829 ØØ         FCN:     DB      Ø         ;AREA TO SAVE FUNCTION CODE
                ;
Ø82A            BUFBEG:  DS      BUFSIZ    ;STRING BUFFER                                M8229
Ø8AA 2AØ8       BUFPTR:  DW      BUFBEG    ;STRING BUFFER POINTER
```

```
                ;
Ø8AC                    DS      64D     ;STACK AREA
                STAK:
                ;
                ;       IT IS ASSUMED THAT THE ROUTINE PRT PRINTS THE CHARACTER
                ;       HELD IN THE A REGISTER, THEN RETURNS TO THE CALLING
                ;       ROUTINE.  ALL REGISTERS (EXCEPT THE FLAGS) ARE ASSUMED
                ;       TO BE UNMODIFIED BY PRT.
                ;
                ;       SIMILARY, IT IS ASSUMED THAT THE ROUTINE KBIN GETS A
                ;       CHARACTER FROM THE KEYBOARD AND RETURNS WITH IT IN
                ;       THE A REGISTER.  ALL OTHER REGISTERS ARE TO BE UNAFFECTED
                ;
                ;
                ;       AN EXAMPLE OF PRT WRITTEN TO USE CP/M'S
                ;       CONSOLE OUTPUT ROUTINE IN CBIOS
                ;
                ;  NOTE:  THE STARTING ADDRESSES OF CONOUT AND CONIN CAN BE
                ;  FOUND IN THE FOLLOWING MANNER:
                ;  1.  GET THE ADDRESS STORED IN THE WORD AT LOCATION ØØØ1
                ;       (LOW BYTE OF ADDR IN ØØØ1, HIGH BYTE IN ØØØ2)
                ;  2.  ADD 6 TO THAT ADDRESS.  THE RESULT IS THE ADDRESS OF
                ;       A JUMP TO THE ROUTINE CONIN.
                ;  3.  ADD 3 TO THE ADDRESS CALCULATED FOR CONIN.  THIS IS
                ;       THE ADDRESS OF A JUMP TO THE ROUTINE CONOUT.
                ;
Ø8EC E5         PRT:    PUSH    H
Ø8ED D5                 PUSH    D
Ø8EE C5                 PUSH    B
Ø8EF F5                 PUSH    PSW     ;SAVE ALL REGISTERS
Ø8FØ E67F               ANI     7FH     ;STRIP PARITY BIT                        M8229
Ø8F2 4F                 MOV     C,A     ;PUT CHAR INTO REG C AS NEEDED BY CBIOS
Ø8F3 CDØCDA             CALL    CONOUT  ;OUTPUT THE CHARACTER
Ø8F6 F1                 POP     PSW     ;RESTORE REGISTERS
Ø8F7 C1                 POP     B
Ø8F8 D1                 POP     D
Ø8F9 E1                 POP     H
Ø8FA C9                 RET
                ;
                ;       AN EXAMPLE OF KBIN WRITTEN TO USE CBIOS CONSOLE
                ;               INPUT ROUTINE
                ;
Ø8FB E5         KBIN:   PUSH    H       ;SAVE REGISTERS
Ø8FC D5                 PUSH    D
Ø8FD C5                 PUSH    B
Ø8FE CDØ9DA             CALL    CONIN   ;GET THE CHAR (CP/M RETURNS WITH CHAR
                                        ;  IN REG A)
Ø9Ø1 E67F               ANI     7FH     ;STRIP PARITY BIT                        M8229
Ø9Ø3 C1                 POP     B       ;RESTORE REGISTERS
Ø9Ø4 D1                 POP     D
Ø9Ø5 E1                 POP     H
Ø9Ø6 C9                 RET
                ;
Ø9Ø7                    END     1ØØH
```

SYMBOL TABLE

```
Ø482 ADRMS    Ø179 ADRSET   Ø3CD AFIRM    Ø4FB BADMS    Ø33F BFL
Ø513 BMS      Ø351 BRK      Ø82A BUFBEG   Ø8AA BUFPTR   ØØ8Ø BUFSIZ
Ø525 CHRMS    Ø537 CLMS     ØØ7D CMDPT    8Ø32 CNTRL    Ø2DB COK
DAØ9 CONIN    DAØC CONOUT   Ø19E CONTRL   Ø475 CRLF     ØØ7E DATPT
ØØ7F DELCHR   Ø56C DLMS     Ø338 DVCL     Ø57B DVMS     Ø828 ECHO
Ø587 ENDMS    8Ø26 ENTBL    Ø58E EOSMS    Ø829 FCN      Ø5AF FCNMS
Ø5C3 FEWMS    Ø3EC FIL1     Ø3F1 FIL2     Ø3EC FILBFR   Ø42D FILXIT
Ø2F2 GETCH1   Ø2DD GETCHR   Ø111 GETFN    Ø3ØB GETS1    Ø2F6 GETSTR
8Ø35 GIM      Ø1A7 GIMSET   Ø458 HEXL     Ø446 HEXO     Ø348 ICLR
Ø5D6 IDMS     Ø5E8 IFCMS    Ø5F8 IFMS     8Ø26 INIT     Ø1BØ INITL
Ø316 JTBL     Ø8FB KBIN     8Ø2C LISTN    Ø6Ø9 LSMSG    Ø1C6 LSN
Ø1E8 LSN1     Ø467 MSG      Ø472 MSGCR    Ø436 NESC     Ø622 NLMS
Ø379 NLS      Ø2B6 NOEOI    Ø62D NOGUD    Ø1CØ NOIFC    Ø63E NORSP
Ø375 NOTCC    Ø421 NOTD     Ø4ØA NOTX     Ø463 NUM      Ø1F6 PIDL
Ø211 PNSET    Ø665 POCMS    Ø382 POCRST   8Ø5Ø PPIDL    Ø675 PPMS
Ø1FC PPOLL    8Ø47 PPQRY    8Ø4D PPREL    8Ø4A PPREQ    Ø8EC PRT
Ø2ØB PSET     Ø3A8 QRY      Ø217 REQ      Ø691 RSBMS    Ø6B3 RSPMS
Ø6D8 SØMSG    Ø6F5 S1MSG    Ø7Ø5 S2MSG    Ø717 S3MSG    Ø732 S4MSG
Ø74A S5MSG    Ø763 S6MSG    Ø191 SET1     Ø198 SETADR   Ø22Ø SHO
8Ø3E SPQRY    8Ø44 SPREL    8Ø41 SPSRQ    Ø2A1 SREL     Ø3A2 SREQ
Ø779 SRQMS    8Ø2F STADR    Ø8EC STAK     Ø1ØØ START    8Ø38 STATE
Ø79C STRMS    8Ø29 TALK     Ø2A7 TALKR    Ø7AB TLKMS    Ø7CF TMS
Ø331 TRGR     Ø7DD TRYAGN   Ø7FC TSMSG    8Ø3B XCTRL    Ø38B XTN
Ø813 XTNMS    Ø2D2 YESN1    Ø2C1 YESNO
```

UNOFFICIAL PHRASEBOOK
IEEE 488 to ENGLISH

IEEE used the following conventions when they assigned the names used in the standard:

> Lower Case names are associated with local messages (messages between a device and its interface; they MIGHT NOT appear on the 488 bus).
>
> Upper Case names are divided into three groups:
>
> > One or two letters name interface functions,
> >
> > Three letter mnemonics are remote messages (communications over the 488 bus from one interface to another) and
> >
> > Four letter names ending in "S" identify the state of an interface function.

The numbers following an entry are the pages of the IEEE Standard (Apr 4, 1975) which give further information.

ACDS ACcept Data State
21,22

ACG Addressed Command Group - multiline messages (ØØ-ØF Hex) which affect only addressed devices. The messages GTL (Go To Local), SDC (Selective Device Clear), PPC (Parallel Poll Configure) and GET (Group Execute Trigger) operate only on devices in the LADS (Listener Addressed) state. TCT (Take Control) operates on the device in the TADS (Talk Addressed) state.
48,77

ACRS ACceptor Ready State
21,22

Addressed Commands - Commands belonging to the Addressed Command Group (See ACG)
43

AH Acceptor Handshake - the device function which allows proper reception of data and commands appearing on the eight data lines of the 488 bus (i.e., multiline messages). The DAV (Data Available) line is sensed to determine when the multiline message is valid, and the AH function indicates its readiness for data by asserting a passive false on the NRFD (Not Ready For Data) line, and that it has received the message by asserting a passive false on the NDAC (Not Data Accepted) line. Note that it is illegal for the AH to assert both NDAC and NRFD passive false simultaneously.
20

Active False - an active false message asserted on the 488 bus is one in which it is guaranteed that a false value is received. It overrides a passive true. The standard is constructed so that it is not possible for an active true and an active false message to be asserted on the bus at the same time.
16

Active True - a message which when asserted on the 488 bus is guaranteed to be received as true. It overrides a passive false. The standard is constructed so that it is not possible for an active true and an active false message to be asserted on the bus at the same time.
16

AIDS ACceptor Idle State
20, 21

ANRS Acceptor Not Ready State
20,21

APRS Affirmative Poll Response State
32

ATN ATtentioN - a uniline remote message indicating that a Controller is sending commands (as contrasted to a Talker sending data) over the eight data (DIO) lines.
19,21,24,29,35,41,48,75-76

AWNS Acceptor Wait for New cycle State
21,22

C Controller interface function - the interface function which allows a device to send device addresses, universal commands and addressed commands over the 488 bus. It also allows the device to conduct a Parallel Poll to determine which device needs service.
41

CACS Controller ACtive State
41,42

CADS Controller ADdressed State
41,42

CAWS Controller Active Wait State
41,43

CIDS Controller IDle State
41

CPPS Controller Parallel Poll State
41,43

CPWS Controller Parallel poll Wait State
41,43

CSBS Controller StandBy State
41,43

CSNS Controller Service Not requested State
41,44

CSRS Controller Service Requested State
41,44

CSWS Controller Synchronous Wait State
41,43

CTRS Controller TRansfer State
41,44

DAB DAta Byte - a multiline sent by the Source Handshake (SH) over the eight data (DIO) lines
25,48,75-76

DAC Data ACcepted - the complement appears on the NDAC line. See AH, SH for further information.
19,22,48,75-76

Data Byte Transfer Control lines - the three lines (DAV, NRFD and NDAC) that are used by the Source and Acceptor functions to perform the handshake cycle.
12,18-22,67

DAV DAta Valid - a uniline message sent by the Source Handshake (SH) function over the DAV line. See SH.
48,75-76

DC Device Clear interface function - the interface function which allows a device to be cleared (initialized) either individually or as part of a group. The group may be either part or all of the addressed devices in one system.
37-38

DCAS Device Clear Active State
38

DCIS Device Clear Idle State
37,38

DCL Device CLear - a multiline message (14 Hex) sent by the Controller over the eight data lines indicating that all devices are to go into the Clear state. The details are device dependent, but usually the device is left in the same state as when its power is first turned on.
38,43,48,75-77

Dense Subset - A subset of the Primary Command Group, consisting of only the Listen Address Group (LAG) and Talk Address Group (TAG). ISO codes Space through Underline, inclusive. (Values 20 Hex through 5F Hex).
77

DIOn Data Input/Output line n (n goes from 1 through 8)
54

DT Device Trigger interface function - the interface function which allows a device to start its basic operation started either individually or as part of a group. This function may be used to start several devices simultaneously.
38-39

DTAS Device Trigger Active State
39

DTIS Device Trigger Idle State
39

END END - a uniline message sent by a Talker (EOI line active true) at the same time a data byte is sent on the data (DIO) lines. The message indicates that this is the last data byte to be sent. (See EOS for an alternate way of terminating a string sent by a Talker).
23,48,75-76

EOI End Or Identify - a uniline message which serves two purposes: if asserted true by a Talker it indicates that the last byte of a string is being sent. If asserted true by a Controller it initiates a Parallel Poll.

EOS End Of String - a multiline message sent by a Talker to indicate that the last byte of a string has been sent. Its value (ISO code) is determined by what the Listener(s) recognize.
48

General Interface Management lines - the five lines used to perform system operations, such as Parallel Poll, Interface Clear, etc. Several of the lines are also used in data transactions: an example is EOI, which may be used to signal the end of a multibyte transaction. The five lines are ATN, EOI, IFC, REN and SRQ.
12

GET Group Execute Trigger - a multiline message (Ø8 Hex) sent by the Controller indicating that all devices addressed as Listeners are to start performing their respective functions. This command is often used to start several pieces of equipment in synchronism.
39,43,48,75-77

GTL Go To Local - a multiline message (Ø1 Hex) sent by the Controller indicating that all devices addressed as Listeners are to go to the Local state: i.e., local controls on the front or back panel (instead of device dependent messages on the 488 bus) control device operation. (See Local Control)
33,43,48,75-77

gts go to standby - a local message sent by a device to its Controller interface function telling it that it is finished sending commands. The response is that the Controller function releases the bus so that other operations (e.g., a Talker sending data to Listeners) may proceed.
41,75

IDY IDentifY - a uniline message sent by the Controller during a Parallel Poll telling the other devices to assert their Parallel Poll responses on the data bus.
35,48,75-76

IFC InterFace Clear - a uniline message sent by the System Controller telling all other devices on the bus to go to the Idle state. This message is used to place all devices in a known state. It should be used sparingly because any bus transaction is terminated by this function.
24,29,41-42,48,75-76

ISO Code - a seven bit code equivalent to the American National Code for Information Interchange, ANSI X3.4-1968 (often called ASCII).
46,50,77

isr individual service request - a local message sent by a device to its Parallel Poll interface function. If the individual status (see "ist") message is equal to the S (Sense) bit received as part of the most recently received PPE (Parallel Poll Enable) command, the PPR (Parallel Poll Response) byte specified by the three bits P1-P3 of the most recent PPE command must be sent true upon receipt of an IDY (Identify) command from the Controller. Alternately, if subset PP2 (Parallel Poll function cannot be configured by the Controller) is used, local messages are substituted for S, P1-P3.
35-37,75

ist individual status - a local message used by the Parallel Poll function to determine the proper response to an IDY (Identify) command from the Controller. See "isr".
35-36

L Listen interface function - the function which allows a device to receive data from the 488 bus.
28

LACS Listener ACtive State
29-30

(LAD) the listen address of a specific device (received as MLA). See "MLA".
43

LADS Listener ADdressed State
28-29

LAG Listen Address Group - a subset of the ISO-7 codes, being characters SPACE through ? (2Ø Hex through 3F Hex).
48, 77

LE Listen Extended interface function - similar to the Listen function except that a Secondary Address must be used as well as the Primary Address used for the Listen function.
30

LIDS Listener IDle State
28-29

LLO Local LockOut - a multiline command (11 Hex) sent by the Controller which tells all devices with the RL (Remote Local) interface function to obey device dependent messages sent over the 488 bus instead of their local controls (e.g., front panel).
33,43,48,75-77

LOCS LOCal State
33

local control - the device is programmed by its controls instead of by the 488 interface. An example is a digital multimeter; the range, function, sample rate, etc. are set by front panel controls if it is under local control.
33

local message - a message sent between a device function and an interface function. It may cause a remote message to be sent from the interface function over the 488 bus.
15

lon listen only - a local message which causes the Listen function of the device to act as if it had been addressed by the Controller.
29,75

LPAS Listener Primary Addressed State
29,30

lpe local poll enable - a local message which causes the Parallel Poll function of the device to act as if it has received a PPE (Parallel Poll Enable) from the Controller. When lpe is false, the device is to act as if it has received a PPD (Parallel Poll Disable) while in the PACS (Parallel Poll Addressed to Configure state) or a PPU (Parallel Poll Unconfigure) command from the Controller.
35,75

LPIS Listener Primary Idle State
29-30

ltn listen - a local message which when true and the Controller is in the active state causes the L (Listen) or LE (Listen Extended) function to go from the Idle (LIDS) to the Addressed (LADS) state.
29,75

lun local unlisten - a local message which when true and the Controller is in the active state (CACS) causes the L (Listen) or LE (Listen Extended) function to go from the Addressed (LADS) to the Idle (LIDS) state.
29,75

LWLS Local With Lockout State
33-34

MLA My Listen Address - the address which the L (Listen) or LE (Listen Extended) function will respond to. Note that the standard does not allow a 488 bus system to have both an L and an LE interface function which respond to the same primary address. MLA must belong to the LAG (Listen Address Group).
48,75-76

MSA My Secondary Address - the secondary address which the TE (Talk Extended) or LE (Listen Extended) functions will respond to if they are in the Primary Addressed state (TPAS or LPAS, respectively). MSA must belong to the SCG (Secondary Command Group).
24,48,75-76

MTA My Talk Address - the primary address which the T (Talk) or TE (Talk Extended) function will respond to. Note that the standard does not allow a 488 bus system to have both a T and TE interface function simultaneaously with the same primary address. MTA must belong to the TAG (Talk Address Group).
24,29,48,75-76

multiline message - a message that is sent over two or more lines of the 488 bus. An example is Device Clear (DCL) (14 Hex sent out on the data (DIO1-DIO8) lines by the Controller).
45

nba new byte available - a local message sent by a device to its Source Handshake (SH) function to inform it that another byte is available for it to place on the bus data (DIO1-DIO8) lines.
19,75

NDAC Not Data ACcepted - one line of the 488 bus which carries the complement of the Data ACcepted (DAC) message. It is one of the three Data Byte Transfer Control lines. (See DAC).

NPRS Negative Poll Response State
32

NRFD Not Ready For Data - one line of the 488 bus. It carries the complement of the Ready For Data (RFD) message, and is one of the three Data Byte Transfer Control lines. (See RFD).

NUL null byte: all eight bits are false.
23,42,48

OSA Other Secondary Address - a secondary address which is not the same as the secondary address of the TE (Talk Extended) function while it is in the TPAS (Talk Primary Addressed state), or of the LE (Listen Extended) function while it is in the LPAS (Listen Primary Addressed state). OSA must belong to the SCG (Secondary Command Group).
48,75-76

OTA Other Talk Address - an address other than a device's own talk address. Some devices which are capable of talking unaddress themselves if they sense that the Controller is addressing another Talker. This feature can be convenient because an UNTalk (UNT) command is not needed. OTA must belong to the TAG (Talk Address Group).
24,48,75-76

PACS Parallel poll Addressed to Configure State
35-36

Passive False - a message which when asserted on the 488 bus is NOT guaranteed to be received as false. It is overridden by an active true message.
16

Passive True - a message which when asserted on the 488 bus is NOT guaranteed to be received as true. It is overridden by an active false message.
16

PCG Primary Command Group - a subset of the ISO-7 code. It consists of all characters NUL through UNDERLINE (ØØ Hex through 5F Hex). It includes all of the ACG (Addressed Command Group), UCG (Universal Command Group), LAG (Listen Address Group) and TAG (Talk Address Group).
35,49,75-77

pon power on - a local message sent by the device to its own interface to inform it that power has just been applied. The interface should reset all functions (e.g., Listen, AH, Talk, etc.) to their Idle states.
75

PP Parallel Poll interface function - the function which allows a device to respond to a Parallel Poll from the Controller.
35

PPAS Parallel Poll Active State
35-36

PPC Parallel Poll Configure - a multiline message (Ø5 Hex) sent by the Controller which causes the device presently addressed as a Listener (e.g., in the LADS state) to go into the PACS (Parallel Poll Addressed to Configure) state. While in the PACS, the PP (Parallel Poll) function is to obey the PPE (Parallel Poll Enable) and PPD (Parallel Poll Disable) messages sent by the Controller.
35,43,75-77

PPD Parallel Poll Disable - a multiline message (7Ø Hex) sent by the Controller which will place all devices in the PACS (Parallel Poll Addressed to Configure) state into the PPIS (Parallel Poll Idle) state.
35,43,49,75-76

PPE Parallel Poll Enable - a multiline message (6Ø-6F Hex) sent by the Controller which will change all devices in the PPIS (Parallel Poll Idle) state to the PPSS (Parallel Poll Standby) state. It also specifies the PPRn (Parallel Poll Response byte) to be used and the S (Sense) of the PPR. The form of the message is (from most significant bit to least)

X 1 1 Ø S P3 P2 P1

where X means don't care (may be either high or low), and the binary value formed by P3-P1 indicates which PPRn is to be used. Note that n of PPRn indicates which data line is to be made active true (i.e., DIO3 will be made active true when PPR3 is placed on the bus).
35,43,49,75-76

PPIS Parallel Poll Idle State
35-36

PPRn Parallel Poll Response n (See PPE)
35,49,75-76

PPSS Parallel Poll Standby State
35-36

PPU Parallel Poll Unconfigure - a multiline message (15 Hex) sent by the Controller which takes all devices in the PPSS (Parallel Poll Standby) state and puts them into the PPIS (Parallel Poll Idle) state.
35,43,49,75-77

PUCS Parallel poll Unaddressed to Configure State
35-36

rdy ready for next message - a local message sent by a device to its AH (Acceptor Handshake) interface function to indicate it is ready for another message byte from the 488 bus (i.e, another multiline remote message).
21,75

remote control - a device is programmed by its 488 interface instead of by local controls. An example is a DMM whose function, range selection, etc are selected by messages sent to it over the 488 bus. See local control for contrast.
33

REMS REMote State
33-34

REN Remote ENable - one of the five General Interface Management lines. Also, a uniline message sent by the Controller to put devices addressed as Listeners into the REMS (Remote) state. When the Controller makes the REN message false, all devices are to go to the LOCS (Local) state.
33,42,49,75-76

RFD Ready For Data - the complement appears on the NRFD line. This uniline message is used by the AH (Acceptor Handshake) function to indicate that it is ready to accept the next byte (multiline message). See AH for further information.
19,22,49,75-76

RL Remote Local interface function - if present it allows a device to be switched from local to remote control and vice versa.
33

rpp request parallel poll - a local message sent to the Controller interface function when the device wants a Parallel Poll performed.
41,75

RQS ReQuest Service - the byte sent by the current Talker in response to a Serial Poll. Data bit 7 (DIO7) is true.
23,49,75-76

rsc request system control - a local message sent to the Controller interface function by the device when it wants to go to the SACS (System Control Active) state.
41,75

rsv request service - a local message sent by a device to its Service Request interface function to cause it to go to the SRQS (Service Request) state. As a consequence, the uniline message SRQ is sent active true until either rsv is sent false, or the Controller performs a Serial Poll of this device.
32,75

rtl return to local - a local message sent by a device to its Remote/Local interface function. The LOCS (Local) state is entered if neither LLO (Local Lockout) nor ACDS (Accept Data State) are true.
33,75

RWLS Remote With Lockout State
33,34

SACS System Control Active State
41,44

(SAD) Secondary ADdress - the seconday address of a specific device, and is received as either My Seconday Address (MSA) or Other Secondary Address (OSA). Its value must lie in the range 6Ø-7E Hex. (See SCG).
43

(SBA) Status Byte, service request Acknowledged. A message sent over the 488 bus by the current Talker in response to a Serial Poll. This message indicates that this device was requesting service. Data bit 7 (DIO7) is true. (See RQS)
62

(SBN) Status Byte, service Not requested. Same as SBA but indicates that this device does not need service. Data bit 7 (DIO7) is false.
62

SCG Secondary Command Group. A subset of the ISO-7 code consisting of characters ACCENT GRAVE through TILDE (6Ø Hex through 7E Hex). Secondary Talk and Listen addresses must be selected from this group. (Note that DEL is not allowed as a secondary address).
49, 77

SDC Selected Device Clear - a multiline message (Ø4 Hex) sent by the Controller indicating that all devices addressed as Listeners are to go into the DCAS (Device Clear Active) state. The details are device dependent, but usually the device is left in the same state as when its power is first turned on.
38,43,49,75-77

SDYS Source DelaY State
18-19

Secondary Commands - the commands PPE, PPD and (SAD).
43

SGNS Source GeNerate State
18-19

SH Source Handshake interface function. The function used by a Talker or Controller to insure proper communication of multiline messages. The NRFD and NDAC lines are sensed to determine whether the AH (Acceptor Handshake) function of some device is active (if both NRFD and NDAC are false simultaneously, there is no AH function on the bus, which is an error). The multiline message is placed on the eight data lines (DIO1-DIO8) and a 2 microsecond timeout is started. When NRFD is sensed false and the timeout has been completed (to insure the data lines have settled) DAV is asserted true (to show that the data is available and settled). Upon sensing NDAC false the SH asserts DAV false (to indicate that the data may no longer be valid) then removes the data. The whole cycle is repeated for subsequent bytes of data. (See AH for the other half of the handshake cycle).
18

SIAS System control Interface clear Active State
41,44

sic send interface clear - a local message which causes the devices' Controller interface function to enter the SIAS (System Control Interface Clear Active) state if it is the System Controller (i.e., it is in the SACS (System Control Active) state). As a consequence, the IFC (Inteface Clear) signal is sent active true. (IFC is a uniline message sent on the IFC line).
41,75

SIDS Source IDle State
18-19

SIIS System control Interface clear Idle State
41,44

SINS System control Interface clear Not active State
41,44

SIWS Source Idle Wait State
19-20

SNAS System control Not Active State
41,44

SPAS Serial Poll Active State
24,26

SPD Serial Poll Disable - a multiline message (19 Hex) sent by the Controller. It informs all devices capable of being Talkers that they are to speak data when they are addressed to talk. (See SPE for contrast).
43,49,75-77

SPE Serial Poll Enable - a mulitline message (18 Hex) sent by the Controller. It informs all devices capable of being Talkers that they are to speak their Serial Poll Status Byte (instead of data) when they are addressed to talk. See SBA, SBN, STB for further information about the status byte.
43,49,75-77

SPIS Serial Poll Idle State
24,26

SPMS Serial Poll Mode State
24,26

SR Service Request interface function. This function allows a device to asynchronously request service from the Controller-In-Charge.
31

SRAS System control Remote enable Active State
41,45

sre send remote enable - a local message sent by a device to its Control interface function. It causes the function to enter the SRAS (System Control Remote Enable Active) state only if it was already in the SACS (System Control Active) state. The uniline message REN is sent active true as long as the Controller remains in the SRAS state.
41,75

SRIS System control Remote enable Idle State
41,44

SRNS System control Remote enable Not active State
41,45

SRQ Service ReQuest - a uniline message sent on the SRQ line by the SR (Service Request) interface function. It is the duty of the Controller to provide the service needed.
49,75-76

SRQS Service ReQuest State
32

STB STatus Byte. Data bits 1 through 6 and bit 8 (DIO1-DIO6, DIO8) sent in response to a Serial Poll. STB is combined with RQS to form the complete byte. (See SBA, SBN).
25,49,75-76

STRS Source TRansfer State
18-19

SWNS Source Wait for New cycle State
18-19

T Talk interface function. This function allows a device to send information to other devices on the 488 bus. Only one byte (selected from the Talker Address Group) need be sent to address the Talker.
23

TACS Talker ACtive State
24,26

(TAD) the Talk ADdress of a specific device. It is received as either My Talk Address (MTA) or Other Talk Address (OTA). It must be a member of the TAG (Talk Address Group).
43

TADS Talker ADdressed State
23-24

TAG Talker Address Group. A subset of the ISO-7 code consisting of all characters from @ through UNDERLINE (4Ø Hex through 5F Hex). The address of a Talker (or the primary address of an Extended Talker) must be selected from this group. Note that UNDERLINE cannot be used as an address, for it is reserved as the Universal Untalk command.
49, 77

tca take control asynchronously - a local message sent by a device to its Controller interface function. It causes the function to go from the CSBS (Controller Standby) state to the CSWS (Controller Synchronous Wait) state, where it waits for at least 500 nsec (to allow the other devices on the 488 bus to respond to the active true assertion of the uniline message ATN), then proceed to the CAWS (Controller Active Wait) state. ATN is active true in both CSWS and CAWS.
41,75

tcs take control synchronously - a local message sent by a device to its Controller interface function. It operates the same as tca EXCEPT that the function goes from CSBS to CSWS only when the AH (Acceptor Handshake) function is in the ANRS (Acceptor Not Ready) state. The effect is to insure that a message sent by a Talker is not garbled or misinterpreted as a message sent by the Controller; ATN will not become active true until the Source Handshake is complete (i.e., DAV is false, showing that the message is no longer valid).
21,41,75

TCT Take ConTrol - a multiline message (Ø9 Hex) sent by the Controller to inform the device currently addressed as a Talker that it is to become the Controller-in-Charge.
41,43,49,75-77

TE Talker Extended interface function. Similar to the Talker (T) function except that this one is addressed by two bytes. The first must be selected from the Talker Address Group (TAG) and the second from the Secondary Command Group (SCG).
23

TIDS Talker IDle State
23-24

ton talk only - a local message sent by a device to its Talk interface function. If IFC (Interface Clear) is false, the Talker function enters the TADS (Talker Addressed) state. Remember that only one Talker may be addressed at a time, so as long as ton is true no other device may have ton true or be addressed as a Talker by the Controller.
24,75

TPAS Talker Primary Addressed State
24,26

TPIS Talker Primary Idle State
24,26

UCG Universal Command Group - A subset of the ISO-7 code consisting of all characters from DLE through US (10 Hex through 1F Hex). These commands operate upon all devices which are capable of responding to a Controller; the devices are not individually addressed. For contrast see Addressed Command Group (ACG).
43,49,77

uniline message - a message that uses only one line of the 488 bus. An example is Service ReQuest (SRQ).

Universal Command Group - See UCG

UNL UNListen - a multiline message (3F Hex or the character "?") sent by the Controller which forces the Listen function of all devices into the LIDS (Listen Idle) state.
29,43,49,75-77

UNT UNTalk - a multiline message (5F Hex or the character "_") sent by the Controller which forces the Talk function of all devices into the TIDS (Talk Idle) state.
49,77

Program Notes

The following listing of the P&T-488 Functional Test program is a version written to run under CP/M (an operating system produced by Digital Research). Only these few things need to be changed for it to run with any specific system:

1 MONITR (a name) - should be SET to the entry point of the user's monitor

2 PRINT (a routine at 03CA) The Functional test program CALLs the subroutine PRINT with the character to be printed in register A. Register pair HL **must** be preserved. All other registers may be trashed.

3 INSTAT (a routine at 03B5) The Functional Test routine CALLs the subroutine INSTAT. If no key has been pressed on the keyboard, INSTAT is to RETurn with the zero flag **set.** If a key has been pressed, INSTAT should check to see if it is a Control C. If it is a Control C, INSTAT should jump to the user's monitor, otherwise it is to RETurn to the calling program with the zero flag **cleared.**

4 BASPRT (a byte at 0103) The third byte of the Functional Test **must** contain the lowest I/O port address used by the P&T-488. If the address switch on the P&T-488 interface board has been changed from 7C Hex, the value contained in this location must also be changed.

```
                  ;
                  ;        P&T 488 TEST ROUTINES
                  ;
                  ;        RUNS UNDER CP/M
                  ;
                  ;
Ø1ØØ                       ORG     Ø1ØØH

ØØØØ #          MONITR    SET     Ø         ;CPM RE-ENTRY POINT
ØØØ5 #          CPMIO     SET     5         ;CPM I/O ROUTINE ENTRY POINT

Ø1ØØ C3C5Ø2     ENTRY:    JMP     SELFCN    ;GO TO SELECT FUNCTION ROUTINE
                ;
Ø1Ø3 7C         BASPRT:   DB      7CH       ;BASE ADDR OF P&T 488 INTERFACE
Ø1Ø4 ØØ         ERBYT:    DB      Ø         ;ANY BIT SET TO 1 IS IN ERROR
Ø1Ø5 ØØ         ERFLG:    DB      Ø         ;PRINT 'NO ERRORS' IF ZERO
Ø1Ø6 ØDØA       STRTMS:   DB      ØDH,ØAH
Ø1Ø8 5Ø26542Ø34           DB      'P&T 488 Functional Test         12-2Ø-78'
Ø13Ø ØDØA                 DB      ØDH,ØAH
Ø132 ØDØA                 DB      ØDH,ØAH
Ø134 446973636F           DB      'Disconnect all 488 devices from P&T 488 then',ØDH,ØAH
Ø162 2Ø7Ø726573           DB      ' press any key to begin test',ØDH,ØAH
Ø18Ø 285468652Ø           DB      '(The power does not have to be turned off before',ØDH,ØAH
Ø1B2 646973636F           DB      'disconnecting 488 devices)',ØDH,ØAH,ØDH,8AH
Ø1DØ 444154C1   DATMS:    DB      'DAT','A'+8ØH
Ø1D4 434F4D4D41CMDMS:     DB      'COMMAND LIN','E'+8ØH
Ø1EØ 5Ø4152414CPOLMS:     DB      'PARALLEL POL','L'+8ØH
Ø1ED 494E544552ISRMS:     DB      'INTERRUPT SERVICE REGISTE','R'+8ØH
Ø2Ø7 ØDØA417474PLUGMS:    DB      ØDH,ØAH,'Attach test plug then press any key',ØDH,8AH
Ø22E 3438382Ø43CBLMS:     DB      '488 CABL','E'+8ØH
Ø237 4558544552XIFMS:     DB      'EXTERNAL INTERFACE CLEA','R'+8ØH
Ø24F 4558544552XATMS:     DB      'EXTERNAL AT','N'+8ØH
Ø25B 4E4F2Ø4552NOERR:     DB      'NO ERRORS',ØDH,8AH
Ø266 5Ø26542Ø34TSTDUN:    DB      'P&T 488 functional test complete',ØDH,8AH
Ø288 2Ø4552524FBITER:     DB      ' ERROR - bits in error are',' '+8ØH
Ø2A3 ØD8A       CRLF:     DB      ØDH,8AH
                ;
Ø2A5                      DS      2ØH       ;STACK AREA
                STAK:
                  ;
                  ;***************************************************************
                  ;
                  ;       TEST EACH FUNCTION IN TURN
                  ;
                  ;***************************************************************
                  ;
Ø2C5 31C5Ø2       SELFCN: LXI     SP,STAK ;SET STACK POINTER
Ø2C8 97                   SUB     A
Ø2C9 32Ø5Ø1               STA     ERFLG   ;RESET ERROR FLAG
Ø2CC CD39Ø3               CALL    SETUP   ;SET UP 488 PORT ROUTINES
Ø2CF 21Ø6Ø1               LXI     H,STRTMS  ;PRINT STARTUP MESSAGE
Ø2D2 CD7EØ3               CALL    PRNT8
Ø2D5 CDB5Ø3       STRTW8: CALL    INSTAT  ;SEE IF A KEY HAS BEEN PUSHED
Ø2D8 CAD5Ø2               JZ      STRTW8  ;..NO, SO WAIT UNTIL ONE IS
Ø2DB CDE3Ø3               CALL    DATA    ;..CHECK DATA PORT OPERATION
Ø2DE 21DØØ1               LXI     H,DATMS
Ø2E1 CD72Ø3               CALL    ERTEST  ;..PRINT ANY NEEDED ERROR MESSAGE
Ø2E4 CDØEØ4               CALL    CMND    ;..CHECK COMMAND PORT OPERATION
Ø2E7 21D4Ø1               LXI     H,CMDMS
Ø2EA CD72Ø3               CALL    ERTEST
Ø2ED CD1DØ4               CALL    PPR     ;..CHECK PARALLEL POLL RESPONSE
Ø2FØ 21EØØ1               LXI     H,POLMS
Ø2F3 CD72Ø3               CALL    ERTEST
Ø2F6 CD37Ø4               CALL    ISRV    ;..CHECK INTERRUPT SERVICE REGISTER
Ø2F9 21EDØ1               LXI     H,ISRMS
Ø2FC CD72Ø3               CALL    ERTEST
                  ;
Ø2FF 21Ø7Ø2               LXI     H,PLUGMS  ;TELL OPERATOR TO ATTACH PLUG
Ø3Ø2 CD7EØ3               CALL    PRNT8
Ø3Ø5 CDB5Ø3       PLUGW8: CALL    INSTAT  ;SEE IF A KEY HAS BEEN PRESSED
Ø3Ø8 CAØ5Ø3               JZ      PLUGW8  ;..NO, SO WAIT UNTIL ONE HAS BEEN
Ø3ØB CDA2Ø4               CALL    CBLTST  ;CHECK CONTINUITY OF 488 CABLE
Ø3ØE 212EØ2               LXI     H,CBLMS
Ø311 CD72Ø3               CALL    ERTEST
Ø314 CDC9Ø4               CALL    XIFC    ;CHECK RESPONSE TO EXTERNAL IFC
Ø317 2137Ø2               LXI     H,XIFMS
Ø31A CD72Ø3               CALL    ERTEST
Ø31D CDF7Ø4               CALL    XATN    ;CHECK RESPONSE TO EXTERNAL ATN
Ø32Ø 214FØ2               LXI     H,XATMS
Ø323 CD72Ø3               CALL    ERTEST
Ø326 3AØ5Ø1               LDA     ERFLG   ;HAVE ANY ERRORS OCCURRED?
Ø329 215BØ2               LXI     H,NOERR
Ø32C B7                   ORA     A
Ø32D CC7EØ3               CZ      PRNT8   ;..NO, SO PRINT 'NO ERRORS'
Ø33Ø 2166Ø2               LXI     H,TSTDUN
Ø333 CD7EØ3               CALL    PRNT8   ;PRINT 'TEST COMPLETE'
Ø336 C3ØØØØ               JMP     MONITR
                  ;
Ø339 3AØ3Ø1       SETUP:  LDA     BASPRT  ;GET PORT ADDRESS
Ø33C E6FC                 ANI     ØFCH    ;MAKE SURE IT IS A VALID ISR PORT ADDR
Ø33E 325BØ3               STA     ISRI1
Ø341 325EØ3               STA     ISRO1
Ø344 3C                   INR     A       ;CALCULATE COMMAND LINE PORT ADDR
Ø345 3261Ø3               STA     CMDI1
Ø348 3264Ø3               STA     CMDO1
Ø34B 3C                   INR     A       ;CALCULATE DATA LINE PORT ADDR
Ø34C 3267Ø3               STA     DATI1
Ø34F 326AØ3               STA     DATO1
Ø352 3C                   INR     A       ;CALCULATE PARALLEL POLL RESPONSE ADDR
Ø353 326DØ3               STA     PPI1
Ø356 327ØØ3               STA     PPO1
Ø359 C9                   RET
                  ;
Ø35A DB           ISRI:   DB      ØDBH    ;IN  ISR
Ø35B ØØ           ISRI1:  DB      Ø
Ø35C C9                   RET
                  ;
Ø35D D3           ISRO:   DB      ØD3H    ;OUT ISR
Ø35E ØØ           ISRO1:  DB      Ø
Ø35F C9                   RET
                  ;
```

```
Ø36Ø DB          CMDI:    DB      ØDBH     ;IN  CMDPORT
Ø361 ØØ          CMDI1:   DB      Ø
Ø362 C9                   RET
                 ;
Ø363 D3          CMDO:    DB      ØD3H     ;OUT CMDPORT
Ø364 ØØ          CMDO1:   DB      Ø
Ø365 C9                   RET
                 ;
Ø366 DB          DATI:    DB      ØDBH     ;IN  DATPORT
Ø367 ØØ          DATI1:   DB      Ø
Ø368 C9                   RET
                 ;
Ø369 D3          DATO:    DB      ØD3H     ;OUT DATPORT
Ø36A ØØ          DATO1:   DB      Ø
Ø36B C9                   RET
                 ;
Ø36C DB          PPI:     DB      ØDBH     ;IN  PARPOLL
Ø36D ØØ          PPI1:    DB      Ø
Ø36E C9                   RET
                 ;
Ø36F D3          PPO:     DB      ØD3H     ;OUT PARPOLL
Ø37Ø ØØ          PPO1:    DB      Ø
Ø371 C9                   RET
                 ;
Ø372 3AØ4Ø1      ERTEST:  LDA     ERBYT    ;GET CUMULATIVE ERRORS FOR THIS TEST
Ø375 B7                   ORA     A
Ø376 C8                   RZ               ;..NO ERRORS
Ø377 32Ø5Ø1               STA     ERFLG    ;SET ERROR FLAG SO 'NO ERRORS' MESSAGE
                                           ;  WILL NOT BE PRINTED AT END OF TEST
Ø37A CD8AØ3               CALL    ERPRNT   ;PRINT ERROR MESSAGE
Ø37D C9                   RET
                 ;
Ø37E 7E          PRNT8:   MOV     A,M      ;GET THE CHAR TO BE PRINTED
Ø37F CDCAØ3               CALL    PRINT    ;PRINT IT ON CONSOLE DEVICE
Ø382 7E                   MOV     A,M      ;GET THE CHAR AGAIN
Ø383 23                   INX     H        ;POINT TO NEXT CHAR
Ø384 E68Ø                 ANI     8ØH      ;SEE IF CARRY SET
Ø386 CA7EØ3               JZ      PRNT8    ;..NO, SO PRINT NEXT CHARACTER
Ø389 C9                   RET
                 ;
Ø38A CD7EØ3      ERPRNT:  CALL    PRNT8    ;PRINT MESSAGE POINTED TO BY HL
Ø38D 2188Ø2               LXI     H,BITER  ;PRINT 'BITS IN ERROR' MESSAGE
Ø39Ø CD7EØ3               CALL    PRNT8
Ø393 2E3Ø                 MVI     L,'Ø'    ;PUT ASCII Ø IN L
Ø395 3AØ4Ø1               LDA     ERBYT
Ø398 ØF          BITLP:   RRC              ;PUT BIT IN CARRY
Ø399 67                   MOV     H,A      ;SAVE ROTATED VALUE IN H
Ø39A D2A6Ø3               JNC     NOBIT
Ø39D 7D                   MOV     A,L      ;PRINT ASCII CHAR IN L
Ø39E CDCAØ3               CALL    PRINT
Ø3A1 3E2Ø                 MVI     A,' '    ;FOLLOW WITH A SPACE
Ø3A3 CDCAØ3               CALL    PRINT
Ø3A6 2C          NOBIT:   INR     L        ;ADVANCE BIT NUMBER
Ø3A7 3E38                 MVI     A,'8'
Ø3A9 BD                   CMP     L        ;HAVE WE FINISHED?
Ø3AA 7C                   MOV     A,H      ;GET BITS AGAIN
Ø3AB C298Ø3               JNZ     BITLP    ;..NO, MORE BITS TO TEST
Ø3AE 21A3Ø2               LXI     H,CRLF   ;FINISH WITH <CR><LF>
Ø3B1 CD7EØ3               CALL    PRNT8
Ø3B4 C9                   RET
                 ;
Ø3B5 ØEØB        INSTAT:  MVI     C,11D    ;DO CPM CONSOLE READY FUNCTION
Ø3B7 CDØ5ØØ               CALL    CPMIO
Ø3BA E6Ø1                 ANI     1        ;LOOK AT ONLY LSB
Ø3BC C8                   RZ               ;NO CHARACTER READY
Ø3BD ØEØ1                 MVI     C,1      ;GET THE CHARACTER
Ø3BF CDØ5ØØ               CALL    CPMIO
Ø3C2 E67F                 ANI     7FH
Ø3C4 FEØ3                 CPI     3        ;CONTROL C?
Ø3C6 CAØØØØ               JZ      MONITR   ;..YES, SO ABORT
Ø3C9 C9                   RET
                 ;
```

```
Ø3CA E5         PRINT:  PUSH    H       ;PRESERVE HL (ONLY REGISTERS THAT
                                        ;NEED TO BE PRESERVED)
Ø3CB 5F                 MOV     E,A     ;PUT CHAR IN E, AS NEEDED BY CPM
Ø3CC ØEØ2               MVI     C,2     ;WRITE TO CONSOLE DEVICE
Ø3CE CDØ5ØØ             CALL    CPMIO   ;CPM I/O ENTRY POINT
Ø3D1 E1                 POP     H
Ø3D2 C9                 RET
                ;
                ;**************************************************************
                ;
                ;       CLEAR THE INTERRUPT SERVICE REGISTER AND RELEASE
                ;       ALL COMMAND LINES AT THE PORTS CORRESPONDING TO
                ;       THE FIRST ENTRY IN BASPRT
                ;
                ;**************************************************************
                ;
Ø3D3 3EFF       RELCLR: MVI     A,-1
Ø3D5 CD63Ø3             CALL    CMDO    ;RELEASE ALL COMMAND LINES
Ø3D8 CD69Ø3             CALL    DATO    ;RELEASE ALL DATA LINES
Ø3DB CD6FØ3             CALL    PPO     ;RELEASE ALL PARALLEL POLL LINES
Ø3DE 97                 SUB     A       ;ZERO A REGISTER
Ø3DF CD5DØ3             CALL    ISRO    ;CLEAR ISR
Ø3E2 C9                 RET
                ;
                ;**************************************************************
                ;
                ;       CHECK DATA REGISTER FOR PROPER OPERATION
                ;
                ;       THE DATA REGISTER CORRESPONDING TO THE FIRST ADDRESS
                ;       IN BASPRT IS WRITTEN TO AND READ FROM.  ALL BITS WHICH
                ;       ARE IN ERROR SHOW UP AS 1'S IN ERBYT (THE ERRORS ARE
                ;       CUMULATIVE) AND ARE ALSO SHOWN AS LIT BITS ON THE
                ;       PROGRAMMED OUTPUT DISPLAY
                ;
                ;**************************************************************
                ;
Ø3E3 CDD3Ø3     DATA:   CALL    RELCLR  ;CLEAR COMMAND, ISR
                                        ;  (TO INSURE THAT WE ARE NOT LOCKED
                                        ;  OUT OR SEEING PARALLEL POLL REGISTER)
Ø3E6 3AØ3Ø1             LDA     BASPRT  ;GET PORT BASE ADDRESS
Ø3E9 E6FC               ANI     ØFCH    ;MAKE SURE THE ADDRESS IS A VALID
Ø3EB F6Ø2               ORI     2       ;  DATA PORT ADDRESS
Ø3ED 5F                 MOV     E,A     ;SET UP OUTPUT PORT
Ø3EE 57                 MOV     D,A     ;  AND INPUT PORT
Ø3EF C3F2Ø3             JMP     PORTST  ;GO TO COMMON PORT TEST ROUTINE




                ;
                ;**************************************************************
                ;
                ;       PORTST          PORT TEST ROUTINE
                ;
                ;       OUTPUTS Ø,1,2,...254,255 TO PORT WHOSE ADDRESS IS IN
                ;       THE E REGISTER, AND READS PORT WHOSE ADDRESS IS IN THE
                ;       D REGISTER.  ANY BITS WHICH DO NOT MATCH ARE ACCUMULATED
                ;       AS CORRESPONDING 1'S IN THE C REGISTER AND IN MEMORY
                ;       LOCATION ERBYT.
                ;
                ;**************************************************************
                ;
Ø3F2 7B         PORTST: MOV     A,E     ;SET UP OUTPUT PORT NUMBER
Ø3F3 32ØØØ4             STA     OUTDR
Ø3F6 7A                 MOV     A,D     ;AND INPUT PORT
Ø3F7 32Ø2Ø4             STA     INDR
Ø3FA ØEØØ               MVI     C,Ø     ;INITIALIZE BIT ERROR REGISTER
Ø3FC Ø6ØØ               MVI     B,Ø     ;INITIALIZE BYTE TEST REGISTER
```

```
Ø3FE 78          DATLUP: MOV     A,B      ;OUTPUT TEST BYTE
Ø3FF D3                  DB      ØD3H
Ø4ØØ ØØ          OUTDR:  DB      Ø
Ø4Ø1 DB                  DB      ØDBH     ;READ PORT
Ø4Ø2 ØØ          INDR:   DB      Ø
Ø4Ø3 A8                  XRA     B        ;DETERMINE WHICH BITS ARE IN ERROR
Ø4Ø4 B1                  ORA     C        ;ADD IN PREVIOUS ERRORS
Ø4Ø5 4F                  MOV     C,A      ;AND SAVE UPDATED ERRORS
Ø4Ø6 32Ø4Ø1              STA     ERBYT
Ø4Ø9 Ø4                  INR     B        ;INCREMENT TEST BYTE
Ø4ØA C8                  RZ               ;..IF HAVE DONE ALL 256 POSSIBLE TESTS
Ø4ØB C3FEØ3              JMP     DATLUP
                 ;
                 ;**************************************************************
                 ;
                 ;       CHECK COMMAND REGISTER
                 ;
                 ;       THE COMMAND REGISTER CORRESPONDING TO THE FIRST ADDRESS
                 ;       IN BASPRT IS WRITTEN TO AND READ FROM.  ALL BITS WHICH
                 ;       ARE IN ERROR SHOW UP AS 1'S IN MEMORY LOCATION ERBYT
                 ;       AND ARE ALSO SHOWN ON THE PROGRAMMED OUTPUT.
                 ;
                 ;**************************************************************
                 ;
Ø4ØE CDD3Ø3      CMND:   CALL    RELCLR   ;CLEAR COMMAND, ISR
Ø411 3AØ3Ø1              LDA     BASPRT   ;GET PORT NUMBER
Ø414 E6FC                ANI     ØFCH     ;MAKE IT INTO A VALID COMMAND PORT
Ø416 F6Ø1                ORI     1
Ø418 5F                  MOV     E,A      ;SET UP OUTPUT PORT
Ø419 57                  MOV     D,A      ;  AND INPUT PORT
Ø41A C3F2Ø3              JMP     PORTST   ;GO TO COMMON PORT TEST ROUTINE
                 ;
                 ;**************************************************************
                 ;
                 ;       CHECK PARALLEL POLL RESPONSE
                 ;
                 ;       THE EOI AND ATN LINES OF THE COMMAND PORT ARE PULLED
                 ;       LOW (ASSERTED TRUE) TO GET THE PARALLEL POLL RESPONSE
                 ;       ONTO THE DATA BUS.  THEN TEST BYTES ARE WRITTEN INTO
                 ;       THE PARALLEL POLL RESPONSE REGISTER AND THE DATA
                 ;       REGISTER IS READ.  ANY BITS WHICH ARE IN ERROR ARE
                 ;       SAVED IN MEMORY LOCATION ERBYT AND SHOWN ON THE
                 ;       PROGRAMMED OUTPUT.
                 ;
                 ;**************************************************************
                 ;
Ø41D CDD3Ø3      PPR:    CALL    RELCLR   ;CLEAR COMMAND, ISR
Ø42Ø 3AØ3Ø1              LDA     BASPRT   ;GET PORT NUMBER
Ø423 E6FC                ANI     ØFCH     ;CHANGE IT INTO A COMMAND PORT
Ø425 F6Ø1                ORI     1
Ø427 32330Ø4             STA     PPOUT    ;STORE IT AS OPERAND OF OUTPUT INSTRUCTION
Ø42A F6Ø2                ORI     2        ;CHANGE IT INTO A PARALLEL POLL PORT
Ø42C 5F                  MOV     E,A      ;SET UP AS OUTPUT PORT
Ø42D E6FE                ANI     ØFEH     ;CHANGE IT INTO A DATA PORT
Ø42F 57                  MOV     D,A      ;SET UP AS AN INPUT PORT
Ø43Ø 3EF6                MVI     A,ØF6H   ;ASSERT ATN, EOI TRUE
Ø432 D3                  DB      ØD3H     ;BY OUTPUTTING TO COMMAND PORT
Ø433 ØØ          PPOUT:  DB      Ø
Ø434 C3F2Ø3              JMP     PORTST   ;THEN JUMP TO COMMON PORT TEST ROUTINE
                 ;
                 ;**************************************************************
                 ;
                 ;       CHECK INTERRUPT SERVICE REGISTER
                 ;
                 ;               WIGGLE EACH COMMAND LINE IN TURN AND CHECK
                 ;               FOR PROPER ISR RESPONSE
                 ;
                 ;**************************************************************
                 ;
Ø437 CDD3Ø3      ISRV:   CALL    RELCLR   ;CLEAR COMMAND, ISR
Ø43A ØEØØ                MVI     C,Ø      ;INITIALIZE ERROR REGISTER
Ø43C 1EØØ                MVI     E,Ø      ;  AND ISR RESET BYTE
```

```
Ø43E 217ØØ4              LXI     H,TSTBL1 ;POINT TO TALK TEST TABLE
Ø441 46         ISTST:   MOV     B,M      ;GET NUMBER OF TESTS TO BE PERFORMED
Ø442 7B                  MOV     A,E      ;RESET ISR
Ø443 CD5DØ3              CALL    ISRO
Ø446 23         ISPAS:   INX     H        ;POINT TO COMMAND TO BE SENT
Ø447 16Ø2                MVI     D,2      ;INITIALIZE ASSERT COMMAND FLAG
                                          ; (D=1 TO RELEASE COMMAND)
Ø449 7E                  MOV     A,M      ;GET COMMAND
Ø44A CD63Ø3     RELES:   CALL    CMDO     ;AND OUTPUT IT
Ø44D CD5AØ3              CALL    ISRI     ;READ ISR
Ø45Ø 23                  INX     H        ;POINT TO EXPECTED RESPONSE
Ø451 AE                  XRA     M        ;SET BITS IN ERROR
Ø452 B1                  ORA     C        ;UPDATE ERROR BYTE
Ø453 4F                  MOV     C,A
Ø454 32Ø4Ø1              STA     ERBYT    ;AND ERROR MEMORY LOCATION
Ø457 7B                  MOV     A,E
Ø458 CD5DØ3              CALL    ISRO     ;RESET ALL ISR LATCHES
Ø45B 3EFF                MVI     A,-1     ;SET UP TO RELEASE ALL COMMAND LINES
Ø45D 15                  DCR     D        ;CHECK RELEASE/DONE FLAG
Ø45E C24AØ4              JNZ     RELES    ;..PERFORM RELEASE FUNCTION
Ø461 Ø5                  DCR     B        ;DECREMENT COUNT OF TESTS TO BE PEFORMED
Ø462 C246Ø4              JNZ     ISPAS    ;..IF MORE TESTS ARE TO BE DONE
Ø465 1D                  DCR     E
Ø466 1C                  INR     E        ;DID WE JUST DO TALK OR LISTEN?
Ø467 CØ                  RNZ              ;..IF LISTEN (SECOND SET OF TESTS)
Ø468 1EØ2                MVI     E,2      ;FROM NOW ON PUT ISR IN LISTEN MODE
Ø46A 2189Ø4              LXI     H,TSTBL2 ;POINT TO LISTEN TEST TABLE
Ø46D C341Ø4              JMP     ISTST    ;AND PERFORM ITS TESTS
                ;
                ;************************************************************
                ;
                ;        INTERRUPT SERVICE REGISTER TEST TABLE
                ;
                ;        TABLE OF COMMANDS AND CORRESPONDING ISR CONTENTS FOR
                ;        THE ASSERTION AND THEN THE RELEASE OF THE COMMANDS.
                ;
                ;        THE FIRST BYTE IS THE NUMBER OF TESTS TO BE PERFORMED
                ;
                ;************************************************************
                ;
                ;        TESTS FOR TALK MODE
                ;
Ø47Ø Ø8         TSTBL1:  DB      8D       ;8 TESTS ARE TO BE PERFORMED
Ø471 7F                  DB      Ø7FH     ;ASSERT DAV
Ø472 FFFF                DW      -1       ;LOW BYTE=RESPONSE OF ISR TO ASSERTION
                                          ;  OF DAV, HIGH BYTE=RESPONSE TO RELEASE
                                          ;  OF DAV
Ø474 BF                  DB      ØBFH     ;ASSERT NRFD
Ø475 FFBF                DW      ØBFFFH
Ø477 DF                  DB      ØDFH     ;NDAC
Ø478 FFDF                DW      ØDFFFH
Ø47A EF                  DB      ØEFH     ;IFC
Ø47B FFFF                DW      -1
Ø47D F7                  DB      ØF7H     ;ATN
Ø47E FFFF                DW      -1
Ø48Ø FB                  DB      ØFBH     ;SRQ
Ø481 FBFF                DW      ØFFFBH
Ø483 FD                  DB      ØFDH     ;REN
Ø484 FFFD                DW      ØFDFFH
Ø486 FE                  DB      ØFEH     ;POC/RESET
Ø487 FFFF                DW      -1
                ;
                ;        LISTEN MODE
                ;
Ø489 Ø8         TSTBL2:  DB      8D       ;8 TESTS
Ø48A 7F                  DB      7FH      ;DAV
Ø48B 7F7F                DW      7F7FH
Ø48D BF                  DB      ØBFH     ;NRFD
Ø48E FFFF                DW      -1
Ø49Ø DF                  DB      ØDFH     ;NDAC
Ø491 FFFF                DW      -1
Ø493 EF                  DB      ØEFH     ;IFC
```

```
Ø494 FFFF               DW      -1
Ø496 F7                 DB      ØF7H    ;ATN
Ø497 FFFF               DW      -1
Ø499 FB                 DB      ØFBH    ;SRQ
Ø49A FBFF               DW      ØFFFBH
Ø49C FD                 DB      ØFDH    ;REN
Ø49D FFFD               DW      ØFDFFH
Ø49F FE                 DB      ØFEH    ;POC/RESET
Ø4AØ FFFF               DW      -1
                ;
                ;*************************************************************
                ;
                ;       CHECK CABLE BY BRINGING EACH COMMAND LINE LOW
                ;       ONE AT A TIME AND OBSERVING WHETHER THE CORRESPONDING
                ;       DATA LINE IS ALSO BROUGHT LOW
                ;
                ;*************************************************************
                ;
Ø4A2 CDD3Ø3     CBLTST: CALL    RELCLR  ;RELEASE ALL DATA, COMMAND LINES
Ø4A5 ØEØØ               MVI     C,Ø     ;CLEAR CUMULATIVE ERROR REGISTER
Ø4A7 1EFE               MVI     E,ØFEH  ;MAKE ONLY ONE BIT TRUE IN TEST BYTE
Ø4A9 21C1Ø4             LXI     H,CBLTBL ;POINT TO EXPECTED RESPONSES
Ø4AC 7B         CBLUP:  MOV     A,E     ;PUT TEST BYTE IN ACCUMULATOR
Ø4AD CD63Ø3             CALL    CMDO    ;  AND THEN ON 488 COMMAND LINES
Ø4BØ CD66Ø3             CALL    DATI    ;GET BYTE FROM 488 DATA LINE PORT
Ø4B3 AE                 XRA     M       ;SET ANY BITS WHICH DISAGREE WITH
                                        ;EXPECTED RESPONSE
Ø4B4 B1                 ORA     C       ;ADD TO CUMULATIVE ERRORS
Ø4B5 4F                 MOV     C,A
Ø4B6 32Ø4Ø1             STA     ERBYT
Ø4B9 23                 INX     H       ;POINT TO NEXT EXPECTED RESPONSE
Ø4BA 7B                 MOV     A,E     ;GET TEST BYTE AGAIN
Ø4BB Ø7                 RLC             ;PREPARE TO CHECK NEXT LINE OF CABLE
Ø4BC 5F                 MOV     E,A     ;SAVE TEST BYTE
Ø4BD DAACØ4             JC      CBLUP   ;..CARRY SET IF THERE ARE MORE LINES TO TEST
Ø4CØ C9                 RET
                ;
Ø4C1 DF         CBLTBL: DB      ØDFH    ;DIO6 CORRESPONDS TO EOI
Ø4C2 EF                 DB      ØEFH    ;DIO5 ..  REN
Ø4C3 FB                 DB      ØFBH    ;DIO3 ..  SRQ
Ø4C4 F7                 DB      ØF7H    ;DIO4 ..  ATN
Ø4C5 FD                 DB      ØFDH    ;DIO2 ..  IFC
Ø4C6 FE                 DB      ØFEH    ;DIO1 ..  NDAC
Ø4C7 7F                 DB      Ø7FH    ;DIO8 ..  NRFD
Ø4C8 BF                 DB      ØBFH    ;DIO7 ..  DAV




                ;
                ;*************************************************************
                ;
                ;       CHECK RESPONSE TO XIFC
                ;
                ;       (DIO2 IS CONNECTED TO XIFC BY SHORTING PLUG)
                ;
                ;*************************************************************
                ;
Ø4C9 3EØA       XIFC:   MVI     A,ØAH   ;MAKE ALL DATA LINES (EXCEPT DIO2,4) TRUE
Ø4CB CD69Ø3             CALL    DATO
Ø4CE 3E18               MVI     A,18H   ;MAKE ALL COMMAND LINES (EXCEPT IFC AND
                                        ;ATN) TRUE
Ø4DØ CD63Ø3             CALL    CMDO
Ø4D3 97                 SUB     A
Ø4D4 CD5DØ3             CALL    ISRO    ;CLEAR ISR
Ø4D7 ØEØØ               MVI     C,Ø     ;CLEAR CUMULATIVE ERROR REGISTER
Ø4D9 3EØ8               MVI     A,8     ;NOW PULL DOWN DIO2 AS WELL
                                        ;  (THIS APPLIES XIFC)
```

```
Ø4DB CD69Ø3                 CALL    DATO
Ø4DE CD5AØ3                 CALL    ISRI      ;LOOK AT ISR
Ø4E1 EE8D                   XRI     8DH       ;COMPARE TO EXPECTED VALUE
Ø4E3 B1                     ORA     C         ;UPDATE CUMULATIVE ERROR REGISTER
Ø4E4 4F                     MOV     C,A
Ø4E5 CD6ØØ3                 CALL    CMDI      ;LOOK AT COMMAND LINES
Ø4E8 EEFF                   XRI     ØFFH      ;COMPARE TO EXPECTED VALUE
Ø4EA B1                     ORA     C         ;UPDATE CUMULATIVE ERROR REGISTER
Ø4EB 4F                     MOV     C,A
Ø4EC CD66Ø3                 CALL    DATI
Ø4EF EEFF                   XRI     ØFFH
Ø4F1 B1                     ORA     C
Ø4F2 4F                     MOV     C,A
Ø4F3 32Ø4Ø1                 STA     ERBYT
Ø4F6 C9                     RET
                  ;
                  ;************************************************************
                  ;
                  ;         CHECK RESPONSE TO XATN
                  ;
                  ;         (DIO4 IS CONNECTED TO XATN BY THE SHORTING PLUG)
                  ;
                  ;************************************************************
                  ;
Ø4F7 3EØA         XATN:     MVI     A,ØAH     ;MAKE ALL DATA LINES (EXCEPT DIO2,4) TRUE
Ø4F9 CD69Ø3                 CALL    DATO
Ø4FC 3E58                   MVI     A,58H     ;MAKE ALL COMMAND LINES (EXCEPT NRFD,
                                              ;ATN AND IFC) TRUE
Ø4FE CD63Ø3                 CALL    CMDO
Ø5Ø1 97                     SUB     A
Ø5Ø2 CD5DØ3                 CALL    ISRO      ;CLEAR ISR
Ø5Ø5 ØEØØ                   MVI     C,Ø       ;CLEAR CUMULATIVE ERROR REGISTER
Ø5Ø7 3EØ2                   MVI     A,2       ;NOW PULL DOWN DIO4 AS WELL
                                              ;  (THIS APPLIES XATN)
Ø5Ø9 CD69Ø3                 CALL    DATO
Ø5ØC CD5AØ3                 CALL    ISRI      ;LOOK AT ISR
Ø5ØF EED5                   XRI     ØD5H      ;COMPARE TO EXPECTED VALUE
Ø511 B1                     ORA     C         ;UPDATE CUMULATIVE ERROR REGISTER
Ø512 4F                     MOV     C,A
Ø513 CD6ØØ3                 CALL    CMDI      ;LOOK AT COMMAND LINES
Ø516 EEBF                   XRI     ØBFH      ;COMPARE TO EXPECTED VALUE
Ø518 B1                     ORA     C         ;UPDATE CUMULATIVE ERROR REGISTER
Ø519 4F                     MOV     C,A
Ø51A CD66Ø3                 CALL    DATI
Ø51D EE7F                   XRI     7FH       ;DATA LINES ARE FF, BUT NRFD IS CONNECTED
                                              ;  TO DIO8 BY THE SHORTING PLUG
Ø51F B1                     ORA     C
Ø52Ø 4F                     MOV     C,A
Ø521 32Ø4Ø1                 STA     ERBYT
Ø524 C9                     RET

Ø525                        END


Ø1Ø3 BASPRT     Ø288 BITER      Ø398 BITLP      Ø22E CBLMS      Ø4C1 CBLTBL
Ø4A2 CBLTST     Ø4AC CBLUP      Ø361 CMDI1      Ø36Ø CMDI       Ø1D4 CMDMS
Ø364 CMDO1      Ø363 CMDO       Ø4ØE CMND       Ø2A3 CRLF       Ø3E3 DATA
Ø367 DATI1      Ø366 DATI       Ø3FE DATLUP     Ø1DØ DATMS      Ø36A DATO1
Ø369 DATO       Ø1ØØ ENTRY      Ø1Ø4 ERBYT      Ø1Ø5 ERFLG      Ø38A ERPRNT
Ø372 ERTEST     Ø4Ø2 INDR       Ø3B5 INSTAT     Ø446 ISPAS      Ø35B ISRI1
Ø35A ISRI       Ø1ED ISRMS      Ø35E ISRO1      Ø35D ISRO       Ø437 ISRV
Ø441 ISTST      Ø3A6 NOBIT      Ø25B NOERR      Ø4ØØ OUTDR      Ø2Ø7 PLUGMS
Ø3Ø5 PLUGW8     Ø1EØ POLMS      Ø3F2 PORTST     Ø36D PPI1       Ø36C PPI
Ø37Ø PPO1       Ø36F PPO        Ø433 PPOUT      Ø41D PPR        Ø3CA PRINT
Ø37E PRNT8      Ø3D3 RELCLR     Ø44A RELES      Ø2C5 SELFCN     Ø339 SETUP
Ø2C5 STAK       Ø1Ø6 STRTMS     Ø2D5 STRTW8     Ø47Ø TSTBL1     Ø489 TSTBL2
Ø266 TSTDUN     Ø24F XATMS      Ø4F7 XATN       Ø4C9 XIFC       Ø237 XIFMS
```

```
;*************************************************************
;
;        P&T-488 CUSTOM SOFTWARE PACKAGE          VERSION 1.4
;
;        COPYRIGHT 1979  PICKLES & TROUT
;
;
;
;  CNTRL
;        THIS ROUTINE IS USED TO SEND COMMANDS TO THE 488 BUS
;        BY MEANS OF THE P&T 488.  TO USE, POINT HL TO THE FIRST
;        BYTE OF THE STRING TO BE SENT, DE TO THE LAST BYTE, AND
;        BC TO THE USER-SUPPLIED JUMP TABLE.  THEN CALL CNTRL;
;        THE ROUTINE RETURNS WITH HL POINTING TO THE LAST BYTE
;        SENT, ATN AND DAV FALSE.  IF THE P&T 488 HAS BEEN
;        EITHER SELECTED AS A LISTENER OR IS TO PERFORM LISTENER
;        HANDSHAKE, THE ROUTINE RETURNS WITH NRFD TRUE, OTHERWISE
;        IT RETURNS WITH NRFD FALSE.
;  NOTE: THIS ROUTINE CAUSES THE P&T 488 TO EXERCISE CONTROL
;        IMMEDIATELY.  IT IS UP TO THE USER TO INSURE THAT
;        CONTROL IS ASSUMED SYNCHRONOUSLY (THE INITIALIZE,
;        TALK AND LISTEN ROUTINES ALL RETURN OR BREAK AT
;        POINTS WHICH WILL GUARANTEE SYNCHRONIZATION).  ALSO,
;        THE CONTROLLER STATE REGISTER IS LEFT IN THE STANDBY
;        STATE - THUS THE P&T 488 IS ASSUMED TO BE THE CONTROLLER
;        IN CHARGE UNTIL EITHER INIT IS CALLED, OR CSTAT IS
;        CLEARED BY THE USER.
;
;  GIM  (GENERAL INTERFACE MANAGEMENT)
;        A ROUTINE WHICH ALLOWS THE USER TO CONTROL THE STATE OF
;        THE IFC, SRQ, REN AND EOI LINES.  CALL GIM WITH THE
;        APPROPRIATE BIT PATTERN IN THE A REGISTER.
;
;        D7   D6   D5   D4   D3   D2   D1   DØ
;        X    X    X    IFC  X    SRQ  REN  EOI
;
;        X MEANS DON'T CARE.
;
;        IF THE BIT CORRESPONDING TO A PARTICULAR LINE IS HIGH (1)
;        THAT LINE IS ASSERTED TRUE ON THE 488 BUS.  EG, TO SEND
;        OUT REN AND EOI, THE A REGISTER WOULD CONTAIN Ø3H (OR ØE3H,
;        Ø63H, ETC).  IT IS UP TO THE USER TO RELEASE THE LINES.
;        FOR INSTANCE, TO CLEAR (INITIALIZE) THE 488 BUS, ONE
;        WOULD CALL GIM WITH A=1ØH (ASSERT IFC TRUE) AND THEN
;        CALL GIM WITH A=ØØH.  NOTE THAT THE 488 STANDARD REQUIRES
;        THAT IFC MUST BE ACTIVE NO LESS THAN 1ØØ MICROSECONDS,
;        SO BE SURE TO WAIT BETWEEN THE TWO CALL GIM INSTRUCTIONS.
;
;  INIT
;        CALL INIT TO CLEAR THE P&T 488 INTERFACE.  ALL 488
;        DATA LINES AND CONTROL LINES ARE LEFT IN THE PASSIVE
;        FALSE STATE, AND THE PARALLEL POLL RESPONSE IS SET TO
;        ALL LINES PASSIVE FALSE.
;           TO CLEAR THE 488 BUS, SET REGISTER B TO ØØ BEFORE
;        CALLING INIT.  AN INTERFACE CLEAR (IFC) WILL BE SENT
;        OUT ON THE 488 BUS AND PUT ALL DEVICES IN A KNOWN STATE.
;
;  LISTN
;        TO RECEIVE DEVICE-DEPENDANT MESSAGES FROM THE 488 BUS,
;        POINT HL TO THE BEGINNING OF A MEMORY BUFFER, DE TO ITS
;        END, AND BC TO THE USER-SUPPLIED JUMP TABLE, AND THEN
;        CALL LISTN.  TWO BITS OF THE A REGISTER ARE USED AS FLAGS
;        FOR SPECIAL OPTIONS.  IF BIT Ø (THE LEAST SIGNIFICANT
;        BIT) IS ZERO, THE LISTEN HANDSHAKE FUNCTION IS PERFORMED:
```

```
;        NO BUFFER IS USED AND THE P&T 488 PARTICIPATES IN THE
;        HANDSHAKE PROCESS.  THIS FUNCTION IS PRIMARILY TO ALLOW
;        THE P&T TO ASSUME CONTROL SYNCHRONOUSLY, BUT CAN ALSO
;        BE USED AS A BYTE-AT-A-TIME LISTENER, WITHOUT REQUIRING
;        A BUFFER.  IF BIT Ø OF THE A REGISTER IS NON-ZERO, THE
;        NORMAL BUFFERED LISTENER FUNCTION IS PERFORMED.  IF
;        BIT 1 OF THE A REGISTER IS 1 THEN THE ROUTINE RETURNS
;        WHEN EITHER END (EOI AND DAV TRUE, ATN FALSE) OR THE
;        EOS BYTE IS SENSED.  IF BIT 1 IS Ø THEN A RETURN IS
;        MADE ONLY UPON DETECTION OF END.  THE ROUTINE RETURNS
;        WITH HL POINTING TO THE LAST BYTE RECEIVED AND NRFD IS
;        LEFT ASSERTED TRUE.
;   NOTE: THIS ROUTINE PERFORMS THE LISTEN-ONLY FUNCTION.  IT
;        SETS THE LISTENER STATE TO ACTIVE WHEN CALLED, AND
;        WHEN IT RETURNS IT LEAVES THE LISTENER STATE REGISTER
;        IN THE LISTENER ADDRESSED STATE.  USE THE ROUTINE
;        "STATE" FIRST IF YOU WANT TO EXECUTE THE P&T 488
;        LISTEN FUNCTION ONLY IF THE CONTROLLER HAS ADDRESSED
;        IT AS A LISTENER.
;
;   PISTF        SET "IST" FALSE
;        A ROUTINE WHICH SETS THE "IST" (INDIVIDUAL STATUS) MESSAGE
;        FALSE.  IF THE SENSE BIT OF THE MOST RECENT PARALLEL POLL
;        ENABLE COMMAND WAS TRUE, THE PARALLEL POLL RESPONSE BYTE IS
;        SET TO A NON-AFFIRMATIVE RESPONSE.  THE PPR (PARALLEL POLL
;        RESPONSE) MESSAGE IS DETERMINED BY THE LOW ORDER FOUR
;        BITS OF THE BYTE STORED AT PPRSP.  THE PARALLEL POLL
;        FUNCTION IS PUT INTO THE PPSS (STANDBY) STATE.
;
;   PISTT        SET "IST" TRUE
;        A ROUTINE WHICH SETS THE "IST" (INDIVIDUAL STATUS) MESSAGE
;        TRUE.  IF THE SENSE BIT OF THE MOST RECENT PARALLEL POLL
;        ENABLE COMMAND WAS TRUE, THE PARALLEL POLL RESPONSE BYTE
;        IS SET TO AN AFFIRMATIVE RESPONSE.  THE PPR (PARALLEL POLL
;        RESPONSE) MESSAGE IS DETERMINED BY THE LOW ORDER FOUR
;        BITS OF THE BYTE STORED AT PPRSP.  THE PARALLEL POLL
;        FUNCTION IS PUT INTO THE PPSS (STANDBY) STATE.
;
;   PPIDL        CLEAR PARALLEL POLL RESPONSE BYTE
;        A ROUTINE WHICH CLEARS THE PARALLEL POLL RESPONSE BYTE
;        TO ALL ZEROS, THUS PREVENTING THE P&T-488 FROM RESPONDING
;        TO A PARALLEL POLL.  THE PARALLEL POLL FUNCTION IS PUT
;        INTO THE PPIS (IDLE) STATE.  EXECUTION OF THIS ROUTINE
;        IS EQUIVALENT TO THE SENDING THE LOCAL MESSAGE "LPE"
;        FALSE.
;
;   PPQRY        PARALLEL POLL
;        A ROUTINE WHICH PERFORMS A PARALLEL POLL AND RETURNS
;        THE RESPONSE IN THE ACCUMULATOR.  NOTE THAT NO CHECK
;        IS MADE TO SEE IF THE P&T-488 IS THE CONTROLLER-IN-CHARGE,
;        SO IT IS UP TO THE USER TO USE THIS ROUTINE ONLY WHEN
;        THE P&T-488 (AND NOT SOME OTHER CONTROLLER) IS IN CHARGE.
;        THE CONTROLLER IS LEFT IN THE STANDBY (CSBS) STATE.
;
;   SPIDL        RESET SERVICE REQUEST (SR) FCN TO IDLE
;        A ROUTINE WHICH PLACES A PASSIVE FALSE ON THE SERVICE
;        REQUEST (SRQ) LINE AND PLACES THE SERVICE REQUEST FCN
;        IN THE IDLE (NPRS) STATE.
;
;   SPQRY
;        SERIAL POLL QUERY: THIS ROUTINE SENDS OUT THE COMMANDS
;        UNL (UNIVERSAL UNLISTEN) AND SPE (SERIAL POLL ENABLE).
;        IT THEN CALLS CNTRL WITH THE TALK ADDRESSES IN ITS BUFFER
;        CNTRL RETURNS EITHER WHEN A DEVICE RESPONDS THAT IT IS
;       THE ONE DESIRING SERVICE, OR WHEN THE END OF BUFFER
;       IS REACHED.  THEN SPQRY SENDS THE COMMAND SPD (SERIAL
;       POLL DISABLE), POINTS HL TO THE PRIMARY (AS CONTRASTED
;       TO SECONDARY) TALK ADDRESS OF THE LAST DEVICE POLLED AND SETS
;       UP THE ACCUMULATOR WITH 4Ø HEX IF NO DEVICE RESPONDED
;       TO THE POLL, OR ØØ HEX IF A DEVICE DID RESPOND.  THE
;       SERIAL POLL RESPONSE BYTE SENT BY THE RESPONDING DEVICE
;       IS RETURNED IN REGISTER B.  (NOTE THAT THE CONTENTS OF
;       REGISTER B ARE MEANINGLESS IF NO DEVICE RESPONDED
;       AFFIRMATIVELY TO THE SERIAL POLL.)
;
;  SPSRQ    SERIAL POLL SERVICE REQUEST
;       A ROUTINE WHICH SETS THE SERVICE REQUEST (SRQ) LINE
;       TRUE THEN DETERMINES WHETHER THE P&T 488 IS THE CONTROLLER-
;       IN CHARGE.  IF SO, IT JUMPS TO THE USER ROUTINE SVCRQ.
;       OTHERWISE IT WAITS FOR AN EXTERNAL CONTROLLER TO DO
;       A SERIAL POLL, TO WHICH IT RESPONDS THEN RETURNS TO THE
;       CALLING PROGRAM.
;
;  STADR
;       CALL STADR TO SET TALKER, LISTENER ADDRESSES, SERIAL
;       POLL STATUS AND END-OF-STRING (EOS) BYTES.  HL MUST
;       POINT TO ADDRESS OF FIRST OF FIVE BYTES.
;
;    EXAMPLE:
;       ADDRS:  DB      '%'     ;PRIMARY LISTENER ADDRESS = %
;               DB      'B'     ;PRIMARY TALKER ADDRESS = B
;               DB      7FH     ;PARALLEL POLL RESPONSE BYTE
;               DB      ØFFH    ;SERIAL POLL STATUS BYTE
;               DB      ØAH     ;END OF STRING BYTE = LINE FEED
;               .
;               .
;               .
;               LXI     H,ADDRS ;POINT HL TO BEGINNING OF ADDRESSES
;               CALL    STADR   ;TRANSFER THEM TO 488 HANDLERS
;               .
;               .
;               .
;
;  TALK
;       TO SEND DEVICE-DEPENDANT MESSAGES ON THE 488 BUS, POINT
;       HL TO THE BEGINNING OF THE STRING OF BYTES TO BE SENT,
;       POINT DE TO THE LAST BYTE OF THE STRING, AND POINT BC
;       TO THE BEGINNING OF THE USER-SUPPLIED JUMP TABLE.
;       CALL TALK; THE ROUTINE RETURNS WITH DAV FALSE.  IF THERE
;       IS NO INTERRUPTION, HL WILL POINT TO LAST BYTE OF STRING,
;       BUT IF AN INTERRUPTION OCCURRED (SUCH AS SOME DEVICE
;       REQUESTING SERVICE AND THE P&T 488 IS CONFIGURED AS THE
;       SYSTEM CONTROLLER OR NO LISTENERS ON THE BUS), HL POINTS
;       TO THE LAST BYTE SENT.  IF THE A REGISTER IS NON-ZERO
;       WHEN THE ROUTINE IS CALLED, THE LAST BYTE IN THE BUFFER
;       WILL BE SENT WITH EOI ACTIVE TRUE.
;  NOTE: THIS ROUTINE PERFORMS THE TALK-ONLY FUNCTION (IT DOES
;       NOT CHECK TO SEE WHETHER THE P&T 488 HAS BEEN ADDREESSED
;       AS A TALKER BY THE CONTROLLER).  EXECUTION OF THIS ROUTINE
;       AUTOMATICALLY SETS THE TALK STATUS REGISTER TO ADDRESSED,
;       AND WHEN THE BUFFER IS EMPTIED THE TALK STATUS REGISTER
;       IS LEFT SET TO TALKER ADDRESSED.  IF YOU WANT TO GO TO THE
;       TALK MODE ONLY IF THE CONTROLLER HAS ADDRESSED THE
;       P&T 488 AS A TALKER, USE THE ROUTINE "STATE" TO DETERMINE
;       WHETHER THE TALK FUNCTION HAS BEEN ADDRESSED.
;
```

```
;  XCTRL        EXTERNAL CONTROLLER RESPONSE ROUTINE
;       THIS ROUTINE ACCEPTS THE COMMANDS PRESENTED ON THE 488
;       BUS BY AN EXTERNAL CONTROLLER (THAT IS, SOME DEVICE
;       OTHER THAN THE P&T 488 IS THE CONTROLLER) AND UPDATES
;       THE VARIOUS STATE REGISTERS AS NECESSARY.  IT RETURNS TO
;       THE CALLING PROGRAM WHEN THE EXTERNAL CONTROLLER CEASES
;       SENDING COMMANDS (WHEN ATN BECOMES FALSE).  BOTH NRFD
;       AND NDAC ARE LEFT TRUE (LOW) TO PREVENT THE TALKER
;       FROM SAYING ANYTHING UNTIL THE S-1ØØ SYSTEM IS READY
;       TO LISTEN.
;
;
;  USER SUPPLIED JUMP TABLE
;       THIS TABLE PROVIDES THE ENTRY POINTS TO SPECIAL ROUTINES
;       REQUIRED BY THE P&T 488 INTERFACE.  IT IS THE USER'S
;       RESPONSIBILITY TO PURGE THE STACK IF HE DOES NOT TERMINATE
;       ANY OF THESE ROUTINES WITH A RETURN.  THE TABLE MUST
;       BE ORGANIZED IN THE ORDER SHOWN.  THE USER NEED NOT RESTORE
;       ANY OF THE REGISTERS BEFORE RETURNING.
;
;    EXAMPLE:
;       JMTBL:  JMP     TRIGR   ;DETECTED DEVICE TRIGGER
;               JMP     DVCLR   ;DETECTED DEVICE CLEAR
;               JMP     BUFUL   ;LISTEN BUFFER IS FULL
;               JMP     IFCLR   ;DETECTED INTERFACE CLEAR
;               JMP     BREAK   ;AFTER EACH BYTE TRANSFER ON THE
;                               ;  488 BUS, A CALL IS MADE TO BREAK.
;                               ;  THIS ALLOWS THE USER TO REGAIN
;                               ;  CONTROL OF THE S-1ØØ SYSTEM BEFORE
;                               ;  A COMPLETE BUFFERFUL OF BYTES
;                               ;  HAS BEEN SENT OVER THE 488 BUS.
;                               ;  IF THE USER DOES NOT WANT TO
;                               ;  INTERRUPT 488 OPERATION, HE MERELY
;                               ;  EXECUTES A RETURN.  THE A REGISTER
;                               ;  CONTAINS THE LAST BYTE COMMUNICATED
;                               ;  OVER THE 488 BUS, AND HL POINT
;                               ;  TO THE BUFFER ADDRESS CONTAINING
;                               ;  THAT BYTE.  THUS THE USER CAN
;                               ;  TERMINATE LISTENING ON A PARTICULAR
;                               ;  ASCII CODE OR NUMBER OF CHARACTERS
;                               ;  COMMUNICATED.
;               JMP     NOLSN   ;NOBODY'S LISTENING!
;               JMP     SVCRQ   ;DETECTED SERVICE REQUEST AND P&T 488
;                               ;  IS THE CONTROLLER.  HL POINTS
;                               ;  TO THE LAST BYTE IN THE BUFFER THAT
;                               ;  HAS BEEN INPUT/OUTPUT.
;               JMP     POC     ;DETECTED S-1ØØ RESET/POWER-ON-CLEAR
;               JMP     XATN    ;SOMEBODY ELSE ASSERTED ATN TRUE!
;
;  STATE
;       THIS ROUTINE PASSES INFORMATION TO THE USER ABOUT THE
;       STATE OF THE 488 INTERFACE.  AFTER A 'CALL STATE' THE
;       BIT PATTERN IN THE A REGISTER HAS THE FOLLOWING MEANING:
;
;       .... ..ØØ       BOTH TALK AND LISTEN ARE IDLE
;       .... ..Ø1       TIDS- (NOT TALKER IDLE STATE)
;       .... ..1Ø       LIDS- (NOT LISTENER IDLE STATE)
;       .... .Ø..       PPIS  (PARALLEL POLL IDLE STATE)
;       .... .1..       PPSS  (PARALLEL POLL STANDBY STATE)
;       ...Ø Ø...       LOCS  (LOCAL STATE)
;       ...Ø 1...       LWLS  (LOCAL WITH LOCKOUT)
;       ...1 Ø...       REMS  (REMOTE STATE)
;       ...1 1...       RWLS  (REMOTE WITH LOCKOUT)
;       .Ø.. ....       CIDS  (CONTROLLER IDLE STATE)
;       .1.. ....       CIDS- (CONTROLLER NOT IDLE STATE)
```

```
                ;
                ;       THE HL REGISTER PAIR IS LEFT POINTING TO THE FIRST
                ;       ENTRY OF THE STATE TABLE, THUS THE USER MAY GET MORE
                ;       DETAILED STATE INFORMATION BY ACCESSING THE TABLE
                ;       HIMSELF.
                ;
                ;
8ØØØ                    ORG     8ØØØH
                ;
ØØ7C =          ISRPT   EQU     7CH         ;ADDR OF 488 INTERRUPT STATUS PORT
ØØ7D =          CMDPT   EQU     7DH         ; ... OF COMMAND PORT
ØØ7E =          DATPT   EQU     7EH         ; ... OF DATA PORT
ØØ7F =          PPORT   EQU     7FH         ; ... OF PARALLEL POLL RESPONSE PORT
                ;
ØØØ1 =          GTL     EQU     1           ;ISO-7 BIT CODE FOR "GO TO LOCAL" COMMAND
ØØØ4 =          SDC     EQU     4           ;...SELECTIVE DEVICE CLEAR
ØØØ5 =          PPC     EQU     5           ;...PARALLEL POLL CONFIGURE
ØØØ8 =          GET     EQU     8           ;...GROUP EXECUTE TRIGGER
ØØØ9 =          TCT     EQU     9           ;...TAKE CONTROL
ØØ11 =          LLO     EQU     11H         ;...LOCAL LOCKOUT
ØØ14 =          DCL     EQU     14H         ;...DEVICE CLEAR
ØØ15 =          PPU     EQU     15H         ;...PARALLEL POLL UNCONFIGURE
ØØ18 =          SPE     EQU     18H         ;...SERIAL POLL ENABLE
ØØ19 =          SPD     EQU     19H         ;...SERIAL POLL DISABLE
ØØ3F =          UNL     EQU     3FH         ;...UNIVERSAL UNLISTEN
ØØ5F =          UNT     EQU     5FH         ;...UNIVERSAL UNTALK
                ;
                ;       VARIABLE AREA
                ;
8ØØØ 21         LSTNP:  DB      '!'         ;PRIMARY LISTEN ADDRESS
8ØØ1 41         TALKP:  DB      'A'         ;PRIMARY TALK ADDRESS
8ØØ2 FF         PPRSP:  DB      ØFFH        ;PARALLEL POLL RESPONSE
8ØØ3 FF         SPSTS:  DB      ØFFH        ;SERIAL POLL STATUS BYTE
8ØØ4 ØA         EOSB:   DB      ØAH         ;END OF STRING CHARACTER
                ;
8ØØ5 ØØ         TSTAT:  DB      Ø           ;TALK STATE (INITIALIZE TO TIDS)
                ;
                ;       .... ...Ø   TIDS    TALK IDLE STATE
                ;       .... ...1   TADS    TALKER ADDRESSED STATE
                ;       .... ...1   TACS    TALKER ACTIVE STATE*
                ;       .... ...1   SPAS    SERIAL POLL ACTIVE STATE*
                ;       .... .Ø..   SPIS    SERIAL POLL IDLE STATE
                ;       .... .1..   SPMS    SERIAL POLL MODE STATE
                ;       .... Ø...   TPIS    TALKER PRIMARY IDLE STATE
                ;       .... 1...   TPAS    TALKER PRIMARY ADDRESSED STATE
                ;
8ØØ6 ØØ         LSTAT:  DB      Ø           ;LISTEN STATE
                ;
                ;       .... ...Ø   LIDS    LISTENER IDLE STATE
                ;       .... ...1   LADS    LISTENER ADDRESSED STATE
                ;       .... ...1   LACS    LISTENER ACTIVE STATE*
                ;       .... .Ø..   LPIS    LISTENER PRIMARY IDLE STATE
                ;       .... .1..   LPAS    LISTENER PRIMARY ADDRESSED STATE
                ;       .... Ø...   ....    LISTEN HANDSHAKE - PARTICIPATES
                ;                           IN 488 COMMUNICATIONS BUT DOES
                ;                           NOT PLACE BYTE INTO BUFFER.
                ;                           MAINLY USED TO ALLOW SYNCHRONIZATION
                ;                           OF ASSUMPTION OF CONTROL BY THE
                ;                           P&T 488.  MAY ALSO BE USED TO
                ;                           READ A BYTE AT A TIME (A CALL
                ;                           TO THE USER SUPPLIED ROUTINE
                ;                           "BREAK" IS EXECUTED AFTER EACH
                ;                           BYTE IS HEARD).
```

```
                  ;         .... 1...       ....     BUFFER ORIENTED LISTENER
                  ;         ...Ø ....       ....     IGNORE EOS
                  ;         ...1 ....       ....     RETURN UPON RECEIPT OF EOS
                  ;
8ØØ7 ØØ           SSTAT:  DB        Ø       ;SERVICE REQUEST STATE
                  ;
                  ;         ..ØØ ....       NPRS     NEGATIVE POLL RESPONSE STATE
                  ;         ..Ø1 ....       SRQS     SERVICE REQUEST STATE
                  ;         ..1Ø ....       APRS     AFFIRMATIVE POLL RESPONSE STATE
                  ;
8ØØ8 ØØ           RSTAT:  DB        Ø       ;REMOTE-LOCAL STATE
                  ;
                  ;         ...Ø Ø...       LOCS     LOCAL STATE
                  ;         ...Ø 1...       LWLS     LOCAL WITH LOCKOUT STATE
                  ;         ...1 Ø...       REMS     REMOTE STATE
                  ;         ...1 1...       RWLS     REMOTE WITH LOCKOUT STATE
                  ;
8ØØ9 ØØ           PSTAT:  DB        Ø       ;PARALLEL POLL STATE
                  ;
                  ;         .... ...Ø       PPIS     PARALLEL POLL IDLE STATE
                  ;         .... ...1       PPSS     PARALLEL POLL STANDBY STATE
                  ;         .... ...1       PPAS     PARALLEL POLL ACTIVE STATE*
                  ;         .... ..Ø.       ....     IST=Ø
                  ;         .... ..1.       ....     IST=1
                  ;         .... .Ø..       PUCS     PARALLEL POLL UNADDRESSED TO CONFIGURE
                  ;         .... .1..       PACS     PARALLEL POLL ADDRESSED TO CONFIGURE
                  ;
8ØØA ØØ           CSTAT:  DB        Ø       ;CONTROLLER STATE
                  ;
                  ;         .... ØØØØ       CIDS     CONTROLLER IDLE STATE
                  ;         .... ØØØ1       CADS     .. ADDRESSED STATE
                  ;         .... ØØ1Ø       CTRS     .. TRANSFER STATE
                  ;         .... ØØ11       CACS     .. ACTIVE STATE
                  ;         .... ØØ11       CPWS     .. PARALLEL POLL WAIT STATE*
                  ;         .... ØØ11       CPPS     .. PARALLEL POLL STATE*
                  ;         .... Ø11Ø       CSBS     .. STANDBY STATE
                  ;         .... ØØ11       CAWS     .. ACTIVE WAIT STATE*
                  ;         .... 1ØØØ       CSWS     .. SYNCHRONOUS WAIT STATE
                  ;         ...Ø ....       CSNS     .. SERVICE NOT REQUESTED STATE
                  ;         ...1 ....       CSRS     .. SERVICE REQUESTED STATE
                  ;         ..Ø. ....       SNAS     SYSTEM CONTROL NOT ACTIVE STATE
                  ;         ..1. ....       SACS     SYSTEM CONTROL ACTIVE STATE
                  ;
                  ;  DEVICE CLEAR
                  ;       NO STATES PRESERVED IN MEMORY: CALLS DVCLR (A
                  ;       USER SUPPLIED ROUTINE, WHICH IS TO END WITH A RET)
                  ;
                  ;  DEVICE TRIGGER
                  ;       NO STATES PRESERVED IN MEMORY: CALLS DTRGR (A
                  ;       USER SUPPLIED ROUTINE, WHICH MUST END WITH A RET)
                  ;
                  ;  >> NOTES <<
                  ;       * - THE STATE IS NOT PRESERVED IN MEMORY.  THE STATE IS
                  ;           KNOWN IMPLICITELY BY THE FACT THAT A PARTICULAR
                  ;           ROUTINE IS BEING EXECUTED.
                  ;
8ØØB 7F           LSTNS:  DB        7FH     ;STORAGE AREA FOR SECONDARY LISTEN ADDRESS
8ØØC 7F           TALKS:  DB        7FH     ;... FOR SECONDARY TALK ADDRESS
8ØØD 1F           GIMTC:  DB        1FH     ;MOST RECENT OUTPUT TO COMMAND LINES
8ØØE ØØØØ         JMPAD:  DW        Ø       ;BEGINNING ADDRESS OF USER JUMP TABLE
8Ø1Ø ØØØØ         BPTR:   DW        Ø       ;POINTER OF BUFFER PRESENTLY IN USE
8Ø12 ØØØØ         TBPTR:  DW        Ø       ;TALK BUFFER POINTER
8Ø14 ØØØØ         TBEND:  DW        Ø       ;ADDRESS OF END OF TALK BUFFER
```

```
8Ø16 ØØØØ      LBPTR:  DW      Ø       ;LISTEN BUFFER POINTER
8Ø18 ØØØØ      LBEND:  DW      Ø       ;ADDRESS OF LISTEN BUFFER END
8Ø1A ØØØØ      CBPTR:  DW      Ø       ;CONTROLLER BUFFER POINTER
8Ø1C ØØØØ      CBEND:  DW      Ø       ;ADDRESS OF CONTROLLER BUFFER END
8Ø1E ØØØØ      SBPTR:  DW      Ø       ;SERIAL POLL BUFFER POINTER
8Ø2Ø ØØØØ      SBEND:  DW      Ø       ;ADDRESS OF SERIAL POLL BUFFER END
8Ø22 ØØ        SPRSP:  DB      Ø       ;SERIAL POLL RESPONSE BYTE
8Ø23 ØØ        LBYTE:  DB      Ø       ;CONTAINS BYTE MOST RECENTLY COMMUNICATED
8Ø24 ØØ        TEOI:   DB      Ø       ;MAKE EOI TRUE ON LAST TALKER BYTE IF <>Ø
8Ø25 ØØ        XSPRS:  DB      Ø       ;BUFFER FOR SERIAL POLL RESPONSE TO
                                       ;  AN EXTERNAL CONTROLLER
               ;
               ;       FIXED AREA - PROMMABLE
               ;
               ;*************************************************************
               ;
               ;       JUMP TABLE OF ENTRY POINTS
               ;
               ;*************************************************************
               ;
8Ø26 C3578Ø    ENTBL:  JMP     INIT    ;CLEAR P&T 488 (SEND IFC IF B=Ø)
8Ø29 C3988Ø            JMP     TALK    ;TALK ONLY ROUTINE
8Ø2C C35381            JMP     LISTN   ;LISTEN ONLY ROUTINE
8Ø2F C3878Ø            JMP     STADR   ;COPY LISTEN, TALK ADDRESSES
8Ø32 C37C82            JMP     CNTRL   ;CONTROLLER FUNCTION
8Ø35 C38F84            JMP     GIM     ;SET IFC, SRQ, REN, EOI
8Ø38 C3F684            JMP     STATE   ;DETERMINE THE STATE OF THE INTERFACE
8Ø3B C32985            JMP     XCTRL   ;EXTERNAL CONTROLLER SERVICE ROUTINE
8Ø3E C3Ø886            JMP     SPQRY   ;SERIAL POLL QUERY ROUTINE
8Ø41 C33187            JMP     SPSRQ   ;SERIAL POLL REQUEST ROUTINE
8Ø44 C36F87            JMP     SPIDL   ;PUT SERVICE REQUEST FCN IN IDLE STATE
8Ø47 C37C87            JMP     PPQRY   ;PARALLEL POLL ROUTINE
8Ø4A C3AD87            JMP     PISTT   ;SET THE "IST" MESSAGE TRUE
8Ø4D C3B987            JMP     PISTF   ;SET THE "IST" MESSAGE FALSE
8Ø5Ø C37484            JMP     PPIDL   ;DISABLE PARALLEL POLL RESPONSE
               ;
               ;*************************************************************
               ;
               ;       CONSTANTS
               ;
               ;*************************************************************
               ;
8Ø53 3F        BSPE:   DB      UNL     ;COMMANDS UNLISTEN, SERIAL POLL ENABLE
8Ø54 18                DB      SPE
8Ø55 19        BSPD:   DB      SPD     ;COMMAND SERIAL POLL DISABLE
8Ø56 5F        BUNT:   DB      UNT     ;COMMAND ANY TALKER TO UNADDRESS ITSELF
               ;
               ;*************************************************************
               ;
               ;       INIT - INITIALIZE P&T 488 AND 488 BUS
               ;
               ;*************************************************************
               ;
8Ø57 3EFF      INIT:   MVI     A,ØFFH  ;CLEAR ALL DATA, CONTROL LINES
8Ø59 D37E              OUT     DATPT
8Ø5B CD3B82            CALL    COMND
8Ø5E D37F              OUT     PPORT   ;CLEAR PARALLEL POLL RESPONSE PORT
8Ø6Ø 97                SUB     A       ;ZERO A REGISTER
8Ø61 D37C              OUT     ISRPT   ;CLEAR ALL INTERRUPT LATCHES, SET
                                       ;  P&T 488 TO NON-INTERRUPT MODE
8Ø63 32Ø68Ø            STA     LSTAT   ;UNADDRESS LISTEN FUNCTION
8Ø66 32Ø58Ø            STA     TSTAT   ;UNADDRESS TALK FUNCTION
8Ø69 32Ø78Ø            STA     SSTAT   ;NEGATIVE POLL RESPONSE (SERVICE REQUEST)
```

```
8Ø6C 32Ø88Ø             STA     RSTAT   ;LOCAL STATE (REMOTE-LOCAL)
8Ø6F 32Ø98Ø             STA     PSTAT   ;PARALLEL POLL IDLE STATE
8Ø72 32ØA8Ø             STA     CSTAT   ;CONTROLLER IDLE STATE
8Ø75 B8                 CMP     B       ;B=Ø?  (IF SO, DO INTERFACE CLEAR)
8Ø76 CØ                 RNZ
8Ø77 3EEF               MVI     A,ØEFH  ;NOW DO A INTERFACE CLEAR
8Ø79 D37D               OUT     CMDPT
8Ø7B E3         TWIDL:  XTHL            ;TWIDDLE THUMBS FOR AWHILE
8Ø7C E3                 XTHL            ;TO ALLOW OTHER DEVICES TO RESPOND
8Ø7D 3D                 DCR     A
8Ø7E C27B8Ø             JNZ     TWIDL
8Ø81 3EFF               MVI     A,ØFFH  ;REMOVE IFC
8Ø83 CD3B82             CALL    COMND   ;OUTPUT NEW COMMAND
8Ø86 C9                 RET
                ;
                ;************************************************************
                ;
                ;       STORE ADDRESSES - SETS TALKER, LISTENER ADDRESSES,
                ;                         PARALLEL POLL AND SERIAL POLL RESPONSE
                ;                         BYTES AND THE END-OF-STRING BYTE TO
                ;                         USER DEFINED VALUES
                ;
                ;************************************************************
                ;
8Ø87 1EØ5       STADR:  MVI     E,5     ;SET BYTE COUNTER TO 5
8Ø89 Ø1ØØ8Ø             LXI     B,LSTNP ;POINT TO CONTROLLER ADDRESS TABLE
8Ø8C 7E         NXTAD:  MOV     A,M     ;GET USER-SUPPLIED ADDRESS
8Ø8D Ø2                 STAX    B       ;SAVE IT IN CONTROLLER ADDRESS TABLE
8Ø8E 23                 INX     H       ;POINT TO NEXT USER-SUPPLIED ADDR LOCATION
8Ø8F Ø3                 INX     B       ;  AND TO NEXT CONTROLLER ADDR LOCATION
8Ø9Ø 1D                 DCR     E       ;DECREMENT BYTE COUNT
8Ø91 C28C8Ø             JNZ     NXTAD   ;..THERE'S MORE TO TRANSFER
8Ø94 CDC587             CALL    PPNBL   ;UPDATE PARALLEL POLL RESPONSE
8Ø97 C9                 RET
                ;
                ;************************************************************
                ;
                ;       TALK-ONLY FUNCTION
                ;
                ;************************************************************
                ;
8Ø98 32248Ø     TALK:   STA     TEOI    ;SAVE EOI FLAG
8Ø9B 3AØ58Ø             LDA     TSTAT   ;GET TALK STATUS
8Ø9E E6Ø4               ANI     4       ;KEEP ONLY SERIAL POLL MODE STATE
8ØAØ 3C                 INR     A       ;SHOW TALKER IS ADDRESSED
8ØA1 32Ø58Ø             STA     TSTAT
8ØA4 97                 SUB     A       ;CLEAR A REGISTER
8ØA5 32Ø68Ø             STA     LSTAT   ;UNADDRESS LISTENER
8ØA8 221Ø8Ø             SHLD    BPTR    ;INITIALIZE BUFFER POINTER
8ØAB 22128Ø             SHLD    TBPTR   ;  AS WELL AS TALK BUFFER POINTER
8ØAE EB                 XCHG
8ØAF 22148Ø             SHLD    TBEND   ;STORE END ADDRESS OF TALK BUFFER
8ØB2 6Ø                 MOV     H,B
8ØB3 69                 MOV     L,C
8ØB4 22ØE8Ø             SHLD    JMPAD   ;STORE USER JUMP TABLE ADDRESS
8ØB7 3AØD8Ø             LDA     GIMTC   ;GET IFC, ATN, SRQ, REN AND EOI STATE
8ØBA F6EØ               ORI     ØEØH    ;MAKE DAV, NRFD, NDAC PASSIVE FALSE
8ØBC D37D               OUT     CMDPT   ;OUTPUT COMMAND
8ØBE 32ØD8Ø             STA     GIMTC   ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
8ØC1 DB7C       TALK1:  IN      ISRPT   ;CHECK FOR POC, ATN AND IFC
8ØC3 2F                 CMA
8ØC4 E619               ANI     19H
8ØC6 C44182             CNZ     PAI
8ØC9 DB7D               IN      CMDPT   ;FIRST SEE IF THERE ARE ANY LISTENERS
```

```
8ØCB 2F                 CMA              ;488 USES NEGATIVE LOGIC
8ØCC E66Ø               ANI      6ØH     ;KEEP ONLY RFD, DAC
8ØCE CCB884             CZ       UNLSN   ;..NO LISTENERS.  I REFUSE TO TALK TO MYSELF.
8ØD1 E64Ø               ANI      4ØH     ;WAIT UNTIL READY FOR DATA IS TRUE
8ØD3 C2C18Ø             JNZ      TALK1
8ØD6 2A128Ø             LHLD     TBPTR   ;GET THE DATA BYTE
8ØD9 7E                 MOV      A,M
8ØDA 32238Ø             STA      LBYTE   ;UPDATE MOST RECENT BYTE REGISTER
8ØDD 2F                 CMA              ;488 HAS NEGATIVE TRUE LOGIC
8ØDE D37E               OUT      DATPT
8ØEØ 2A148Ø             LHLD     TBEND   ;IS THIS THE LAST BYTE IN THE TALK BUFFER?
8ØE3 EB                 XCHG
8ØE4 2A128Ø             LHLD     TBPTR
8ØE7 7C                 MOV      A,H
8ØE8 BA                 CMP      D
8ØE9 C2Ø281             JNZ      NTLST   ;..NO
8ØEC 7D                 MOV      A,L
3ØED BB                 CMP      E
8ØEE C2Ø281             JNZ      NTLST   ;..NO
8ØF1 3A248Ø             LDA      TEOI    ;IS EOI SUPPOSED TO BE TRUE?
8ØF4 B7                 ORA      A
8ØF5 CAØ281             JZ       NTLST   ;..NO
8ØF8 3AØD8Ø             LDA      GIMTC
3ØFB E6FE               ANI      ØFEH    ;FORCE EOI ACTIVE TRUE
8ØFD D37D               OUT      CMDPT   ;OUTPUT COMMAND
8ØFF 32ØD8Ø             STA      GIMTC   ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
81Ø2 3AØD8Ø    NTLST:   LDA      GIMTC   ;NOW SET DAV ACTIVE TRUE
81Ø5 E67F               ANI      7FH
81Ø7 F66Ø               ORI      6ØH     ;BUT SET NRFD, NDAC PASSIVE FALSE
81Ø9 D37D               OUT      CMDPT   ;OUTPUT COMMAND
81ØB 32ØD8Ø             STA      GIMTC   ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
81ØE DB7C      TALK2:   IN       ISRPT   ;CHECK FOR POC, ATN, IFC
811Ø 2F                 CMA
8111 E619               ANI      19H
8113 C44182             CNZ      PAI
8116 DB7D               IN       CMDPT   ;WAIT FOR DATA ACCEPTED
8118 E62Ø               ANI      2ØH     ;LOOK AT DAC BIT
811A CAØE81             JZ       TALK2   ;..DATA NOT ACCEPTED YET
811D 3AØD8Ø             LDA      GIMTC   ;GET STATE OF IFC, ATN, SRQ, REN, EOI
812Ø F6E1               ORI      ØE1H    ;MAKE DAV, NRFD, NDAC, EOI PASSIVE FALSE
8122 D37D               OUT      CMDPT   ;OUTPUT COMMAND
8124 32ØD8Ø             STA      GIMTC   ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
8127 3EFF               MVI      A,ØFFH  ;REMOVE DATA FROM LINES
8129 D37E               OUT      DATPT
812B 212B81    TCNTU:   LXI      H,TCNTU ;GET TALK CONTINUATION ENTRY ADDRESS
812E CDD184             CALL     SRVIS
8131 CDC184             CALL     UBRAK   ;SEE IF THE USER WANTS CONTROL OF S-1ØØ
8134 2A148Ø             LHLD     TBEND   ;SEE IF LAST BYTE WAS SENT
8137 EB                 XCHG
8138 2A128Ø             LHLD     TBPTR
813B 7C                 MOV      A,H
813C BA                 CMP      D
813D C24681             JNZ      NTEND   ;..NOT TALK BUFFER END
814Ø 7D                 MOV      A,L
8141 BB                 CMP      E
8142 C24681             JNZ      NTEND   ;..HAVE NOT FINISHED TALK BUFFER
8145 C9                 RET
8146 2A128Ø    NTEND:   LHLD     TBPTR   ;GET TALK BUFFER POINTER
8149 23                 INX      H       ;POINT TO NEXT BYTE
814A 22128Ø             SHLD     TBPTR
814D 22108Ø             SHLD     BPTR    ;UPDATE TALK BUFFER AND COMMON BUFFER POINTER
815Ø C3C18Ø             JMP      TALK1   ;KEEP TALKING UNTIL INTERRUPTED OR FINISHED

               ;
```

```
                    ;
                    ;*************************************************************
                    ;
                    ;       LISTEN-ONLY FUNCTION
                    ;
                    ;*************************************************************
                    ;
8153 C5             LISTN:  PUSH    B           ;SAVE BC FOR LATER
8154 1F                     RAR                 ;SEE IF BIT Ø OF A REG IS Ø
8155 D25D81                 JNC     BYTL        ;..YES, SO SET UP BYTE LISTENER
8158 Ø6ØA                   MVI     B,1ØD       ;SET LSTAT TO ACTIVE, BUFFERED
815A C36281                 JMP     EOST
815D 21188Ø         BYTL:   LXI     H,LBEND ;USE THIS LOCATION AS THE "BUFFER"
816Ø Ø6Ø1                   MVI     B,1         ;SET LSTAT TO ADDRESSED/ACTIVE, NON-BUFFERED
8162 1F             EOST:   RAR                 ;TEST FOR EOS OPTION
8163 78                     MOV     A,B
8164 D26981                 JNC     LSET        ;..LEAVE OPTION FLAG CLEARED
8167 F61Ø                   ORI     1ØH         ;SET OPTION FLAG IN LSTAT
8169 32Ø68Ø         LSET:   STA     LSTAT       ;AND STORE IN LISTENER STATE BYTE
816C C1                     POP     B           ;RESTORE BC REGISTERS
816D 3EFF                   MVI     A,ØFFH  ;ASSERT DATA LINES PASSIVE FALSE
816F D37E                   OUT     DATPT
8171 3AØD8Ø                 LDA     GIMTC       ;GET STATE OF IFC, ATN, SRQ, REN AND EOI
8174 E69F                   ANI     9FH         ;MAKE NRFD ACTIVE TRUE
8176 F6AØ                   ORI     ØAØH        ;MAKE DAV, NDAC PASSIVE FALSE
8178 D37D                   OUT     CMDPT       ;OUTPUT COMMAND
817A 32ØD8Ø                 STA     GIMTC       ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
817D 3AØ58Ø                 LDA     TSTAT       ;UNADDRESS TALKER, BUT LEAVE SERIAL POLL
818Ø E6Ø4                   ANI     4           ;  MODE STATE ALONE
8182 32Ø58Ø                 STA     TSTAT
8185 221Ø8Ø                 SHLD    BPTR        ;INITIALIZE BUFFER POINTER TO BEGINNING
                                                ;  OF BUFFER
8188 22168Ø                 SHLD    LBPTR       ;DO THE SAME FOR THE LISTEN BUFFER
818B EB                     XCHG
818C 22188Ø                 SHLD    LBEND       ;STORE ADDRESS OF LISTEN BUFFER END
818F 6Ø                     MOV     H,B
819Ø 69                     MOV     L,C
8191 22ØE8Ø                 SHLD    JMPAD       ;STORE USER JUMP TABLE ADDRESS
8194 DB7C           DAVH:   IN      ISRPT       ;CHECK FOR POC, ATN AND IFC
8196 2F                     CMA
8197 E619                   ANI     19H
8199 C45882                 CNZ     LPAI
819C DB7D           LSN1:   IN      CMDPT       ;WAIT UNTIL DAV IS HIGH (PASSIVE FALSE)
819E E68Ø                   ANI     8ØH
81AØ CA9481                 JZ      DAVH
81A3 3AØD8Ø                 LDA     GIMTC       ;SET NDAC, NRFD LOW
81A6 E69F                   ANI     9FH
81A8 F68Ø                   ORI     8ØH         ;SET DAV PASSIVE FALSE (HIGH)
81AA D37D                   OUT     CMDPT       ;OUTPUT COMMAND
81AC 32ØD8Ø                 STA     GIMTC       ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
81AF 21AF81         LCNTU:  LXI     H,LCNTU ;GET LISTEN CONTINUATION ADDRESS
81B2 CDD184                 CALL    SRVIS
81B5 3AØD8Ø                 LDA     GIMTC       ;GET LOW BITS OF CONTROL WORD
81B8 F6CØ                   ORI     ØCØH        ;SET "NDAC" LOW, "NRFD" HIGH
81BA E6DF                   ANI     ØDFH
81BC D37D                   OUT     CMDPT       ;OUTPUT COMMAND
81BE 32ØD8Ø                 STA     GIMTC       ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
81C1 DB7C           DAVL:   IN      ISRPT       ;CHECK FOR ATN, POC OR IFC
81C3 2F                     CMA
81C4 E619                   ANI     19H
81C6 C45882                 CNZ     LPAI
81C9 DB7D                   IN      CMDPT       ;NOW WAIT FOR "DAV" LOW (ASSERTED TRUE)
81CB E68Ø                   ANI     8ØH
81CD C2C181                 JNZ     DAVL
81DØ 3AØD8Ø                 LDA     GIMTC       ;SET ONLY NDAC, NFRD LOW
```

```
81D3 E69F               ANI     9FH
81D5 D37D               OUT     CMDPT     ;OUTPUT COMMAND
81D7 32ØD8Ø             STA     GIMTC     ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
81DA DB7E               IN      DATPT     ;GET THE DATA
81DC 2F                 CMA               ;488 USES ACTIVE LOW LOGIC
81DD 2A168Ø             LHLD    LBPTR     ;STORE BYTE IN BUFFER
81EØ 77                 MOV     M,A
81E1 32238Ø             STA     LBYTE     ;AND IN FIXED MEMORY LOCATION (FOR
                                          ;  BYTE ORIENTED LISTENER)
81E4 DB7D               IN      CMDPT
81E6 F5                 PUSH    PSW       ;KEEP IMAGE OF 488 CMD LINES SO CAN CHECK
                                          ;  FOR END
81E7 3AØD8Ø             LDA     GIMTC
81EA F6AØ               ORI     ØAØH      ;ASSERT ONLY "NRFD" (NDAC SET HIGH)
81EC D37D               OUT     CMDPT     ;OUTPUT COMMAND
81EE 32ØD8Ø             STA     GIMTC     ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
81F1 CDC184             CALL    UBRAK     ;SEE IF USER WANTS CONTROL OF S-1ØØ
81F4 F1                 POP     PSW       ;CHECK FOR EOI ACTIVE TRUE
81F5 E6Ø1               ANI     1
81F7 CA1282             JZ      LDUN      ;..LAST BYTE HAS EOI TRUE.
81FA 3AØ68Ø             LDA     LSTAT     ;TERMINATE ON EOS?
81FD E61Ø               ANI     1ØH
81FF CAØC82             JZ      NEOS      ;..NO
82Ø2 3AØ48Ø             LDA     EOSB      ;GET END-OF-STRING BYTE
82Ø5 2A168Ø             LHLD    LBPTR
82Ø8 BE                 CMP     M         ;COMPARE TO BYTE JUST RECEIVED-ARE THEY
                                          ;  THE SAME?
82Ø9 CA1282             JZ      LDUN      ;..YES
82ØC CD1C82     NEOS:   CALL    BFCHK     ;CHECK FOR FULL BUFFER
82ØF C39481             JMP     DAVH      ;REPEAT LOOP FOREVER
                ;
8212 3AØ68Ø     LDUN:   LDA     LSTAT
8215 E618               ANI     18H       ;KEEP HANDSHAKE AND EOS FLAGS
8217 3C                 INR     A         ;SHOW LISTEN STATE IS ADDRESSED (NOT ACTIVE)
8218 32Ø68Ø             STA     LSTAT
821B C9                 RET
                ;
821C 2A188Ø     BFCHK:  LHLD    LBEND     ;PUT BUFFER END ADDRESS IN DE
821F EB                 XCHG              ;  POINTER IN HL
822Ø 2A168Ø             LHLD    LBPTR
8223 3AØ68Ø             LDA     LSTAT     ;DETERMINE IF BYTE OR BUFFER ORIENTED LISTENER
8226 E6Ø8               ANI     8
8228 C8                 RZ                ;..BYTE ORIENTED
                ;
8229 7D                 MOV     A,L       ;CHECK FOR END OF BUFFER
822A BB                 CMP     E
822B C23382             JNZ     NOFLO     ;..MORE BUFFER AVAILABLE
822E 7C                 MOV     A,H
822F BA                 CMP     D
823Ø CAB284             JZ      UBFUL     ;..BUFFER FULL, GO TO USER FOR INSTRUCTIONS
8233 23         NOFLO:  INX     H         ;POINT TO NEXT BUFFER LOCATION
8234 221Ø8Ø             SHLD    BPTR      ;UPDATE BUFFER POINTER
8237 22168Ø             SHLD    LBPTR
823A C9                 RET
                ;
                ;*************************************************************
                ;
                ;       COMND               ROUTINE TO OUTPUT BYTE IN A REGISTER
                ;                           TO GENERAL INTERFACE MANAGEMENT AND
                ;                           DATA TRANSFER CONTROL PORT.  IT ALSO
                ;                           UPDATES GIMTC, A MEMORY IMAGE OF THE
                ;                           MOST RECENT COMMAND PLACED ON THE
                ;                           GIM & TC LINES.
                ;
                ;*************************************************************
```

```
               ;
823B D37D      COMND:  OUT     CMDPT    ;OUTPUT COMMAND
823D 32ØD8Ø            STA     GIMTC    ;UPDATE MEMORY IMAGE OF MOST RECENT COMMAND
824Ø C9                RET
               ;
               ;**********************************************************
               ;
               ;       PAI      CHECK FOR P&T 488 LOCKOUT DUE TO S-1ØØ RESET
               ;                OR 488 EXTERNAL CONTROLLER ASSERTING ATN
               ;                OR IFC TRUE.  EXTERNAL ATTENTION IS TO BE SERVICED
               ;        ONLY IF THE INTERFACE IS NOT LOCKED OUT DUE TO AN S-1ØØ POC
               ;        OR 488 IFC (INTERFACE CLEAR).
               ;
               ;**********************************************************
               ;
8241 214182    PAI:    LXI     H,PAI    ;RE-ENTER THIS ROUTINE UNTIL EACH OF
8244 E5        PAI1:   PUSH    H        ;  POC, ATN AND IFC HAVE BEEN CLEARED
8245 DB7C              IN      ISRPT
8247 1F                RAR              ;PUT POC BIT IN CARRY
8248 D2BE84            JNC     UPOC     ;..IF POC ACTIVE TRUE
824B 1F                RAR              ;REN > CARRY
824C 1F                RAR              ;SRQ > CARRY
824D 1F                RAR              ;XATN > CARRY
824E 1F                RAR              ;XIFC > CARRY
824F D29F84            JNC     UIFC     ;..XIFC IS ACTIVE TRUE
8252 17                RAL              ;XATN > CARRY
8253 D26F82            JNC     PUATN    ;..XATN HAS CHANGED STATES
8256 E1                POP     H        ;CLEAR RE-ENTRY ADDRESS ON STACK
8257 C9                RET
               ;
8258 219C81    LPAI:   LXI     H,LSN1   ;PUT COMMON LISTEN RETURN ADDRESS IN HL
                                        ;  IN CASE ATN IS ACTIVE TRUE
825B E3                XTHL
825C DB7C              IN      ISRPT
825E 1F                RAR
825F D2BE84            JNC     UPOC
8262 1F                RAR
8263 1F                RAR
8264 1F                RAR
8265 1F                RAR
8266 D29F84            JNC     UIFC
8269 17                RAL
826A D26F82            JNC     PUATN
826D E3                XTHL             ;RETURN TO CALLING PROGRAM
826E C9                RET
               ;
826F DB7D      PUATN:  IN      CMDPT    ;SEE IF ATN HAS BEEN ASSERTED OR RELEASED
8271 E6Ø8              ANI     8
8273 CAB584            JZ      UATN     ;..ASSERTED, KEEP RETURN ADDRESS
8276 3EF7              MVI     A,ØF7H   ;..RELEASED, SO RESET XATN BIT IN ISR
8278 D37C              OUT     ISRPT
827A E3                XTHL             ;PUT NORMAL RETURN ADDR BACK ON STACK
827B C9                RET
               ;
```

```
                  ;***************************************************************
                  ;
                  ;       CNTRL           TAKE CONTROL OF THE 488 BUS
                  ;
                  ;       OUTLINE OF OPERATION:
                  ;
                  ;       SET NRFD, NDAC LOW (TRUE)       SET UP AH (ACCEPTOR HANDSHAKE)
                  ;       SET DAV HIGH, ATN LOW           TAKE CONTROL OF THE BUS
                  ; CLUP: SET NRFD HIGH                   AH READY
                  ;       WAIT UNTIL NRFD HIGH            WAIT FOR OTHER DEVICES
                  ;       PLACE BYTE ON DATA LINES        CONTROLLER TELLS IT LIKE IT IS
                  ;       SET DAV LOW                     AND CLAIMS THE DATA LINES ARE VALID
                  ;       SET NRFD LOW                    AH PREPARES TO GET BYTE
                  ;       READ THE DATA LINES
                  ;       SET NDAC HIGH                   AH GOT THE BYTE AND CHEWS IT
                  ;       SET UP APPROPRIATE STATES
                  ;       WAIT FOR NDAC HIGH              CONTROLLER WAITS FOR OTHER DEVICES
                  ;       SET DAV HIGH                    CONTROLLER PLANS TO CHANGE DATA LINES
                  ;       END OF CONTROLLER BUFFER?
                  ;  YES: SET ALL DATA LINES HIGH         CLEAR DATA LINES
                  ;       SET NRFD LOW IF P&T IS A        LOCK UP UNTIL LISTEN FUNCTION READY
                  ;           LISTENER
                  ;       SET ATN HIGH                    RELINQUISH CONTROL
                  ;       RETURN TO CALLER
                  ;  NO:  CALL BREAK                      SEE IF USER WANTS SOMETHING
                  ;       ADVANCE BUFFER POINTER
                  ;       JMP CLUP                        SEND NEXT BYTE
                  ;
                  ;***************************************************************
                  ;
827C CD9282       CNTRL:  CALL    CTRL     ;DO THE CONTROLLER THING
827F 3AØA8Ø               LDA     CSTAT    ;PUT CONTROLLER INTO STANDBY (CSBS)
8282 E6FØ                 ANI     ØFØH     ;BUT KEEP OTHER STATE INFO
8284 F6Ø6                 ORI     6
8286 32ØA8Ø               STA     CSTAT
8289 3AØD8Ø               LDA     GIMTC    ;RELEASE ATN LINE
828C F6Ø8                 ORI     8
828E CD3B82               CALL    COMND
8291 C9                   RET
                  ;
8292 221Ø8Ø       CTRL:   SHLD    BPTR     ;INITIALIZE COMMON BUFFER POINTER
8295 221A8Ø               SHLD    CBPTR    ;  AS WELL AS CONTROLLER BUFFER POINTER
8298 EB                   XCHG
8299 221C8Ø               SHLD    CBEND    ;STORE END ADDRESS OF CONTROLLER BUFFER
829C 6Ø                   MOV     H,B
829D 69                   MOV     L,C
829E 22ØE8Ø               SHLD    JMPAD    ;STORE USER JUMP TABLE BASE ADDRESS
82A1 3AØA8Ø               LDA     CSTAT    ;TAKE CONTROLLER OUT OF IDLE STATE
82A4 E6FØ                 ANI     ØFØH
82A6 F6Ø3                 ORI     3        ;AND MAKE IT ACTIVE
82A8 32ØA8Ø               STA     CSTAT
82AB 3AØD8Ø               LDA     GIMTC    ;GET IFC, ATN, SRQ, REN, EOI STATE
82AE E69F                 ANI     9FH      ;PULL NRFD, NDAC LOW (ACTIVE TRUE)
82BØ F68Ø                 ORI     8ØH      ;MAKE DAV HIGH (PASSIVE FALSE)
82B2 CD3B82               CALL    COMND
82B5 E6F7                 ANI     ØF7H     ;ASSERT ATN TRUE (LOW)
82B7 CD3B82               CALL    COMND
82BA 3AØD8Ø       CLUP:   LDA     GIMTC
82BD F64Ø                 ORI     4ØH      ;SHOW ACCEPTOR HANDSHAKE READY
82BF CD3B82               CALL    COMND
```

```
82C2 CD4182    CTRL1:  CALL    PAI     ;CHECK FOR P&T 488 LOCKOUT DUE TO
                                       ;  EXTERNAL IFC OR S-1ØØ POC
                                       ;>> NOTE << THE ATN WE ARE ASSERTING MASKS
                                       ;ANY EXTERNAL APPLICATION OF ATN, SO WE
                                       ;NEED NOT WORRY ABOUT SOME OTHER CONTROLLER
                                       ;SENDING ATN ACTIVE TRUE.
82C5 DB7D              IN      CMDPT   ;SEE IF ALL DEVICES READY FOR BYTE
82C7 E64Ø              ANI     4ØH
82C9 CAC282            JZ      CTRL1   ;..NOT READY YET
82CC 2A1A8Ø            LHLD    CBPTR   ;GET THE BYTE
82CF 7E                MOV     A,M
82DØ 2F                CMA             ;488 HAS NEGATIVE LOGIC
82D1 D37E              OUT     DATPT
82D3 3AØD8Ø            LDA     GIMTC
82D6 E67F              ANI     7FH     ;MAKE DAV ACTIVE TRUE (LOW)
82D8 CD3B82            CALL    COMND
82DB E6BF              ANI     ØBFH    ;MAKE NRFD TRUE (LOW)
82DD CD3B82            CALL    COMND
82EØ DB7E              IN      DATPT   ;READ THE BYTE
82E2 2F                CMA             ;488 USES NEGATIVE LOGIC
82E3 47                MOV     B,A     ;SAVE IT FOR NOW
82E4 32238Ø            STA     LBYTE   ;SAVE IT IN THE LAST BYTE REGISTER
82E7 3AØD8Ø            LDA     GIMTC
82EA F62Ø              ORI     2ØH     ;SHOW CONTROLLER WE GOT IT
                                       ;  (MAKE NDAC PASSIVE FALSE)
82EC CD3B82            CALL    COMND
82EF CD7383            CALL    UPD8    ;LOOK THIS COMMAND OVER AND SEE IF ANY
                                       ;  OF THE INTERFACE FUNCTIONS ARE
                                       ;  AFFECTED.  UPDATE THE FUNCTION STATES
                                       ;  AS NECESSARY.  THE COMMAND IS IN REG B.
82F2 CD4182    CTRL2:  CALL    PAI     ;CHECK FOR LOCKOUT DUE TO POC, XATN OR XIFC
82F5 DB7D              IN      CMDPT   ;WAIT FOR NDAC HIGH (FALSE)
82F7 E62Ø              ANI     2ØH
82F9 CAF282            JZ      CTRL2   ;..NDAC LOW (TRUE)
82FC 3AØD8Ø            LDA     GIMTC   ;SET DAV HIGH (FALSE)
82FF F68Ø              ORI     8ØH
83Ø1 CD3B82            CALL    COMND
83Ø4 3EFF              MVI     A,ØFFH  ;RELEASE THE 488 DATA LINES
83Ø6 D37E              OUT     DATPT
83Ø8 21Ø883    CCNTU:  LXI     H,CCNTU ;SET UP SRQ RE-ENTRY ADDRESS
83ØB CDD184            CALL    SRVIS   ;CHECK FOR SRQ (SERVICE REQUEST)
83ØE CDC184    CTRL6:  CALL    UBRAK   ;SEE IF USER WANTS CONTROL OF S-1ØØ
8311 2A1C8Ø            LHLD    CBEND   ;GET CONTROLLER BUFFER END ADDRESS
8314 EB                XCHG
8315 2A1A8Ø            LHLD    CBPTR   ;AND POINTER ADDRESS
8318 7C                MOV     A,H
8319 BA                CMP     D
831A C22683            JNZ     NCEND   ;..NOT AT END OF CONTROLLER BUFFER
831D 7D                MOV     A,L
831E BB                CMP     E
831F C22683            JNZ     NCEND   ;..NOT AT END OF CONTROLLER BUFFER
8322 CD3383            CALL    ADDRES  ;FINISH ADDRESSING OF TALK, LISTEN
                                       ;  OF P&T-488
8325 C9                RET
               ;
8326 2A1A8Ø    NCEND:  LHLD    CBPTR   ;GET CONTROLLER BUFFER POINTER
8329 23                INX     H       ;POINT TO NEXT ENTRY IN BUFFER
832A 221A8Ø            SHLD    CBPTR
832D 221Ø8Ø            SHLD    BPTR    ;UPDATE COMMON BUFFER POINTER
833Ø C3BA82            JMP     CLUP    ;AND SEND NEXT BYTE
               ;
```

```
                ;*************************************************************
                ;
                ;  THIS ROUTINE CHECKS TO SEE IF THE TALK OR LISTEN FUNCTION
                ;  IS IN THE PRIMARY ADDRESSED STATE.  IF IT IS, THIS ROUTINE
                ;  CHANGES THE STATE TO ADDRESSED AND PUTS A DUMMY SECONDARY
                ;  ADDRESS IN THE SECONDARY ADDRESS STORAGE LOCATION.
                ;
                ;*************************************************************
                ;
8333 3AØ58Ø     ADDRES: LDA     TSTAT   ;SEE IF TALKER IN PRIMARY ADDRESSED STATE
8336 E6Ø8               ANI     8
8338 CA5383             JZ      NTPRI   ;..NO, SO LEAVE IT ALONE
833B 3AØ58Ø             LDA     TSTAT   ;GET TALKER STATE AGAIN
833E F6Ø1               ORI     1       ;SHOW IT AS ADDRESSED
834Ø E6F7               ANI     ØF7H    ;AND NO LONGER PRIMARY ADDRESSED
8342 32Ø58Ø             STA     TSTAT
8345 3E7F               MVI     A,7FH   ;PUT IN DUMMY SECONDARY ADDRESS
8347 32ØC8Ø             STA     TALKS   ; TO SHOW NON-EXTENDED TALKER
834A 3AØ68Ø     LNADR:  LDA     LSTAT   ;KEEP THE HANDSHAKE AND EOS FLAGS,
834D E618               ANI     18H     ; BUT UNADDRESS THE LISTEN FUNCTION
834F 32Ø68Ø             STA     LSTAT   ; SINCE THE TALK FUNCTION IS ADDRESSED
8352 C9                 RET
                ;
8353 3AØ68Ø     NTPRI:  LDA     LSTAT   ;NOW CHECK LISTENER STATE
8356 E6Ø4               ANI     4       ; TO SEE IF IN PRIMARY ADDRESSED STATE
8358 CA7283             JZ      NLPRI   ;..NO, GO ON TO NEXT FUNCTION
835B 3AØ68Ø             LDA     LSTAT   ;GET LISTEN STATE AGAIN
835E F6Ø1               ORI     1       ;SHOW IT AS ADDRESSED
836Ø E6FB               ANI     ØFBH    ;BUT NOT PRIMARY ADDRESSED
8362 32Ø68Ø             STA     LSTAT
8365 3E7F               MVI     A,7FH   ;AND PUT DUMMY SECONDARY ADDRESS TO
8367 32ØB8Ø             STA     LSTNS   ; SHOW NON-EXTENDED LISTENER
836A 3AØ58Ø     TNADR:  LDA     TSTAT   ;KEEP THE SERIAL POLL STATE BUT
836D E6Ø4               ANI     4       ; UNADDRESS THE TALK FUNCTION SINCE THE
836F 32Ø58Ø             STA     TSTAT   ; LISTEN FUNCTION IS ADDRESSED
8372 C9         NLPRI:  RET
                ;
                ;*************************************************************
                ;
                ;       UPD8    THE COMMAND FROM THE CONTROLLER-IN-CHARGE
                ;               IS IN THE B REGISTER.  LOOK AT THE COMMAND
                ;               AND UPDATE THE FUNCTIONAL STATE OF THE
                ;               INTERFACE AS IS NECESSARY.
                ;
                ;*************************************************************
                ;
8373 78         UPD8:   MOV     A,B     ;PUT THE COMMAND IN REGISTER A
8374 E67F               ANI     7FH     ;STRIP THE PARITY BIT
8376 47                 MOV     B,A     ;SAVE IT IN B FOR LATER USE
8377 FE6Ø               CPI     6ØH
8379 F2C683             JP      RSCG    ;..BELONGS TO SECONDARY COMMAND GROUP
                                        ; (SECONDARY ADDRESS, ETC)
837C CD3383             CALL    ADDRES  ;IF TALK OR LISTEN IS PRIMARY ADDRESSED
                                        ; CHANGE IT TO ADDRESSED AND PUT IN
                                        ; DUMMY SECONDARY ADDRESS
                ;
                ;       >>>     PRIMARY COMMAND GROUP   <<<<<
                ;
837F 78                 MOV     A,B     ;GET COMMAND AGAIN
833Ø FEØ5               CPI     PPC     ;IS IT PARALLEL POLL CONFIGURE?
8382 CA5784             JZ      RPPC    ;..YES, SO UPDATE THE PP STATE
8385 3AØ98Ø             LDA     PSTAT   ;..NO, SO PUT PP STATE INTO PUCS
8388 E6FB               ANI     ØFBH
```

```
838A 32Ø98Ø                 STA     PSTAT
838D 78                     MOV     A,B       ;GET THE COMMAND AGAIN
838E FE4Ø                   CPI     4ØH
839Ø F2ØF84                 JP      RTAG      ;..TALK ADDRESS GROUP
8393 FE2Ø                   CPI     2ØH
8395 F22884                 JP      RLAG      ;..LISTEN ADDRESS GROUP
8398 FEØ1                   CPI     GTL
839A CA4D84                 JZ      RGTL      ;..GO TO LOCAL
839D FEØ4                   CPI     SDC
839F CA4E84                 JZ      RSDC      ;..SELECTIVE DEVICE CLEAR
83A2 FEØ8                   CPI     GET
83A4 CA6684                 JZ      RGET      ;..GROUP EXECUTE TRIGGER
83A7 FEØ9                   CPI     TCT
83A9 CA6F84                 JZ      RTCT      ;..TAKE CONTROL
83AC FE11                   CPI     LLO
83AE CA7Ø84                 JZ      RLLO      ;..LOCAL LOCKOUT
83B1 FE14                   CPI     DCL
83B3 CA7184                 JZ      RDCL      ;..UNIVERSAL DEVICE CLEAR
83B6 FE15                   CPI     PPU
83B8 CA7484                 JZ      RPPU      ;..PARALLEL POLL UNCONFIGURE
83BB FE18                   CPI     SPE
83BD CA7D84                 JZ      RSPE      ;..SERIAL POLL ENABLE
83CØ FE19                   CPI     SPD
83C2 CA8684                 JZ      RSPD      ;..SERIAL POLL DISABLE
83C5 C9                     RET               ;DON'T RECOGNIZE THE COMMAND
                    ;
83C6 3AØ58Ø         RSCG:   LDA     TSTAT     ;SEE IF IN TALKER PRIMARY ADDRESS STATE
83C9 E6Ø8                   ANI     8
83CB CADF83                 JZ      RSCG1     ;..NO
83CE 78                     MOV     A,B       ;GET SECONDARY ADDRESS AGAIN
83CF 32ØC8Ø                 STA     TALKS     ;AND SHOW IT AS TALK SECONDARY ADDRESS
83D2 3AØ58Ø                 LDA     TSTAT     ;GET TALKER STATE AGAIN
83D5 E6Ø4                   ANI     4         ;KEEP ONLY SERIAL POLL MODE STATE
83D7 3C                     INR     A         ;SHOW TALKER IS ADDRESSED
83D8 32Ø58Ø                 STA     TSTAT
83DB CD4A83                 CALL    LNADR     ;UNADDRESS DUE TO MY TALK ADDRESS
83DE C9                     RET               ;DONE INTERPRETING THE COMMAND
                    ;
83DF 3AØ68Ø         RSCG1:  LDA     LSTAT     ;SEE IF IN LISTENER PRIMARY ADDRESSED STATE
83E2 E6Ø4                   ANI     4
83E4 CAF883                 JZ      RSCG2     ;..NO
83E7 78                     MOV     A,B       ;SAVE LISTENER SECONDARY ADDRESS
83E8 32ØB8Ø                 STA     LSTNS
83EB 3AØ68Ø                 LDA     LSTAT     ;GET LISTENER STATE AGAIN
83EE E618                   ANI     18H       ;KEEP LISTEN HANDSHAKE AND EOS FLAGS
83FØ 3C                     INR     A         ;SHOW STATE AS ADDRESSED LISTENER
83F1 32Ø68Ø                 STA     LSTAT
83F4 CD6A83                 CALL    TNADR     ;UNADDRESSED DUE TO MY LISTEN ADDRESS
83F7 C9                     RET               ;DONE INTERPRETING COMMAND
83F8 3AØ98Ø         RSCG2:  LDA     PSTAT     ;SEE IF PARALLEL POLL IS TO BE CONFIGURED
83FB E6Ø4                   ANI     4         ;PARALLEL POLL IN PACS?
83FD CAØE84                 JZ      RSCG3     ;..NO
84ØØ 78                     MOV     A,B       ;GET THE COMMAND AGAIN
84Ø1 E61Ø                   ANI     1ØH       ;IS IT PPD (PARALLEL POLL DISABLE)?
84Ø3 C27484                 JNZ     PPIDL     ;..YES, SO PUT PP INTO PPIS
84Ø6 78                     MOV     A,B       ;..NO, SO SAVE PPE MESSAGE
84Ø7 32Ø28Ø                 STA     PPRSP
84ØA CDC587                 CALL    PPNBL     ;PUT THE APPROPRIATE PPR MESSAGE IN
                                              ;  THE PARALLEL POLL RESPONSE REGISTER
84ØD C9                     RET
                    ;
84ØE C9             RSCG3:  RET               ;NO OTHER FUNCTIONS DECODED YET++++
                    ;
```

```
84ØF 3AØ58Ø    RTAG:   LDA    TSTAT    ;GET TALK STATUS
8412 E6Ø4              ANI    4        ;KEEP SERIAL POLL MODE STATE
8414 4F                MOV    C,A      ;SAVE IT IN REGISTER C
8415 3AØ18Ø            LDA    TALKP    ;GET PRIMARY TALK ADDRESS
8418 B8                CMP    B
8419 C22384            JNZ    NTLK     ;..COMMAND DOES NOT MATCH PRIMARY TALK
                                       ;  ADDRESS
841C 79                MOV    A,C      ;GET TALK STATE AGAIN
841D F6Ø8              ORI    8        ;SHOW PRIMARY ADDRESSED STATE
841F 32Ø58Ø            STA    TSTAT
8422 C9                RET             ;DONE INTERPRETING THE COMMAND
               ;
8423 79        NTLK:   MOV    A,C      ;GET TALK STATE AGAIN
8424 32Ø58Ø            STA    TSTAT    ;SHOW IT AS UNADDRESSED (BECAUSE THIS
                                       ;  COMMAND WAS EITHER UNIVERSAL UNTALK
                                       ;  OR OTHER TALK ADDRESS)
8427 C9                RET             ;DONE INTERPRETING THE COMMAND
               ;
8428 3AØ68Ø    RLAG:   LDA    LSTAT    ;GET LISTEN STATE
842B E618              ANI    18H      ;KEEP ONLY HANDSHAKE AND EOS FLAGS
842D 4F                MOV    C,A      ;SAVE IT IN REGISTER C FOR LATER
842E 3AØØ8Ø            LDA    LSTNP    ;GET PRIMARY LISTEN ADDRESS
8431 B8                CMP    B
8432 C23C84            JNZ    NLSN     ;..DOES NOT MATCH COMMAND
8435 79                MOV    A,C      ;GET UNADDRESSED LISTEN AGAIN
8436 F6Ø4              ORI    4        ;SHOW IT AS PRIMARY ADDRESS STATE
8438 32Ø68Ø            STA    LSTAT
843B C9                RET
               ;
843C 3AØ68Ø    NLSN:   LDA    LSTAT    ;THIS IS NOT MY LISTEN ADDRESS, SO
843F E6FB              ANI    ØFBH     ;  INSURE THAT P&T-488 IS IN LPIS STATE
8441 32Ø68Ø            STA    LSTAT
8444 78                MOV    A,B      ;GET COMMAND
8445 FE3F              CPI    3FH      ;UNIVERSAL UNLISTEN?
8447 CØ                RNZ             ;OTHER LISTEN ADDRESS, SO LEAVE LSTAT ALONE
8448 79                MOV    A,C      ;GET UNADDRESSED LISTEN STATE
8449 32Ø68Ø            STA    LSTAT    ;DUE TO UNIVERSAL UNLISTEN COMMAND
844C C9                RET
               ;
844D C9        RGTL:   RET             ;GO TO LOCAL FUNCTION NOT IMPLEMENTED
               ;
844E 3AØ68Ø    RSDC:   LDA    LSTAT    ;SELECTIVE DEVICE CLEAR
8451 E6Ø3              ANI    3        ;IS THE LISTEN MODE ADDRESSED?
8453 C8                RZ              ;..NO
8454 C3AF84            JMP    UDVCL    ;YES, SO CLEAR THE DEVICE
               ;
8457 3AØ68Ø    RPPC:   LDA    LSTAT    ;PARALLEL POLL CONFIGURE
845A E6Ø1              ANI    1        ;SEE IF LISTEN FCN IN LADS
845C C8                RZ              ;..NO, SO IGNORE PPC COMMAND
845D 3AØ98Ø            LDA    PSTAT    ;..YES, SO PUT PP INTO PACS
846Ø F6Ø4              ORI    4
8462 32Ø98Ø            STA    PSTAT
8465 C9                RET
               ;
8466 3AØ68Ø    RGET:   LDA    LSTAT    ;GROUP EXECUTE TRIGGER
8469 E6Ø3              ANI    3        ;SEE IF LISTEN FUNCTION ADDRESSED
846B C8                RZ              ;..NO
846C C3AC84            JMP    UTRGR    ;YES, SO PERFORM DEVICE TRIGGER
               ;
846F C9        RTCT:   RET             ;TAKE CONTROL - NOT IMPLEMENTED
               ;
847Ø C9        RLLO:   RET             ;LOCAL LOCKOUT - NOT IMPLEMENTED
               ;
8471 C3AF84    RDCL:   JMP    UDVCL    ;UNIVERSAL DEVICE CLEAR
```

```
                ;
                RPPU:
8474 97         PPIDL:  SUB     A       ;PARALLEL POLL UNCONFIGURE
8475 32Ø98Ø             STA     PSTAT   ;PUT PP INTO PPIS/PUCS
8478 3EFF               MVI     A,ØFFH  ;CLEAR RESPONSE BYTE REGISTER
847A D37F               OUT     PPORT
847C C9                 RET
                ;
847D 3AØ58Ø     RSPE:   LDA     TSTAT   ;SET SERIAL POLL MODE BIT IN TALKER
848Ø F6Ø4               ORI     4       ;  STATE REGISTER
8482 32Ø58Ø             STA     TSTAT
8485 C9                 RET
                ;
8486 3AØ58Ø     RSPD:   LDA     TSTAT   ;CLEAR SERIAL POLL MODE BIT IN
8489 E6FB               ANI     ØFBH    ;  TALKER STATE REGISTER
848B 32Ø58Ø             STA     TSTAT
848E C9                 RET
                ;
                ;
                ;************************************************************
                ;
                ;       GIM - GENERAL INTERFACE MANAGEMENT
                ;
                ;               A ROUTINE WHICH ALLOWS THE USER TO SET THE
                ;               STATE OF THE IFC, SRQ, REN AND EOI LINES
                ;
                ;************************************************************
                ;
848F C5         GIM:    PUSH    B       ;IMMEDIATELY SET IFC, SRQ, REN AND EOI LINES
849Ø E617               ANI     17H     ;STRIP OUT DON'T CARES
8492 2F                 CMA             ;488 USES NEGATIVE LOGIC
8493 47                 MOV     B,A
8494 3AØD8Ø             LDA     GIMTC   ;GET STATE OF LOCAL ATN, ETC
8497 F617               ORI     17H     ;STRIP OUT IFC, SRQ, ETC
8499 AØ                 ANA     B       ;COMBINE INTO NEW COMMAND
849A CD3B82             CALL    COMND   ;OUTPUT NEW COMMAND
849D C1                 POP     B       ;RESTORE BC
849E C9                 RET
                ;
                ;************************************************************
                ;
                ;       CALCULATE AND JUMP TO APPROPRIATE ENTRY IN
                ;       USER-SUPPLIED JUMP TABLE
                ;
                ;************************************************************
                ;
849F 97         UIFC:   SUB     A       ;ZERO REG A
84AØ 32Ø68Ø             STA     LSTAT   ;PUT LISTEN FCN IN IDLE
84A3 32Ø58Ø             STA     TSTAT   ;PUT TALK FCN IN IDLE
84A6 32ØA8Ø             STA     CSTAT   ;PUT CONTROLLER FCN IN IDLE
84A9 1EØ9               MVI     E,9
84AB Ø1                 DB      1
84AC 1EØØ       UTRGR:  MVI     E,Ø     ;E=DIFFERENCE BETWEEN USER JUMP TABLE
84AE Ø1                 DB      1       ;  BASE ADDRESS AND DESIRED ENTRY POINT
84AF 1EØ3       UDVCL:  MVI     E,3     ;  BC=GARBAGE
84B1 Ø1                 DB      1
84B2 1EØ6       UBFUL:  MVI     E,6
84B4 Ø1                 DB      1
84B5 1E18       UATN:   MVI     E,24D
84B7 Ø1                 DB      1
84B8 1EØF       UNLSN:  MVI     E,15D
84BA Ø1                 DB      1
84BB 1E12       USRQ:   MVI     E,18D
```

```
84BD Ø1                 DB      1
84BE 1E15       UPOC:   MVI     E,21D
84CØ Ø1                 DB      1
84C1 1EØC       UBRAK:  MVI     E,12D
84C3 16ØØ               MVI     D,Ø
84C5 2AØE8Ø             LHLD    JMPAD   ;GET BASE ADDRESS OF USER JUMP TABLE
84C8 19                 DAD     D       ;CALCULATE ACTUAL ADDRESS
84C9 E5                 PUSH    H       ;AND PUT IT ON THE STACK
84CA 3A238Ø             LDA     LBYTE   ;PUT LAST BYTE HEARD IN A REG
84CD 2A1Ø8Ø             LHLD    BPTR    ;AND POINTER OF CURRENT BUFFER IN HL
84DØ C9                 RET             ;THEN "RET" TO USER JUMP TABLE
                ;
                ;*************************************************************
                ;
                ;       SRVIS - CHECK FOR SERVICE REQUEST.  IF SRQ IS TRUE,
                ;               THE STATE TABLE IS CHECKED TO DETERMINE IF
                ;               THE P&T 488 IS THE CONTROLLER-IN-CHARGE.  IF
                ;               IT IS, THE ADDRESS IN REGISTERS HL IS SUBSTITUTED
                ;               FOR THE RETURN ADDRESS AND A BRANCH IS MADE TO
                ;               THE USER-SUPPLIED ROUTINE SVCRQ.  IF THE CONDITIONS
                ;               ARE NOT MET, A RETURN IS MADE TO THE CALLING ROUTINE.
                ;
                ;*************************************************************
                ;
84D1 DB7D       SRVIS:  IN      CMDPT   ;CHECK FOR SRQ TRUE (LOW)
84D3 E6Ø4               ANI     4
84D5 CAE184             JZ      SRV1    ;..SRQ TRUE: SHOULD WE IGNORE IT?
84D8 3AØA8Ø             LDA     CSTAT
84DB E6EF               ANI     ØEFH    ;PUT CONTROLLER INTO CSNS STATE
84DD 32ØA8Ø             STA     CSTAT
84EØ C9                 RET
                ;
84E1 3AØA8Ø     SRV1:   LDA     CSTAT   ;IGNORE SRQ LINE IF IT HAS ALREADY BEEN
84E4 E61Ø               ANI     1ØH     ;   DETECTED
84E6 CØ                 RNZ
                ;
84E7 3AØA8Ø             LDA     CSTAT   ;SET CSRS STATE IN CONTROLLER STATE
84EA F61Ø               ORI     1ØH
84EC 32ØA8Ø             STA     CSTAT
84EF E6ØF               ANI     ØFH     ;SEE IF P&T 488 CONTROLLER FUNCTION IS IDLE
84F1 C8                 RZ              ;CONTROLLER FUNCTION IN IDLE STATE -
                                        ;  SOMEBODY ELSE IS TO TAKE CARE OF THE SRQ
84F2 E3                 XTHL            ;SUBSTITUE CONTENTS OF HL FOR RETURN ADDRESS
84F3 C3BB84             JMP     USRQ    ;AND GO TO USER-SUPPLIED SRQ ROUTINE
                ;
                ;*************************************************************
                ;
                ;       STATE - RETURNS WITH ABBREVIATED STATE INFORMATION
                ;               IN THE A REGISTER, AND HL POINTING TO THE
                ;               FIRST ENTRY OF THE STATE TABLE.  THUS IF
                ;               THE USER REQUIRES DETAILED STATE INFORMATION,
                ;               HE CAN LOOK INTO THE STATE TABLE.
                ;
                ;*************************************************************
                ;
84F6 C5         STATE:  PUSH    B       ;PRESERVE BC REGISTERS
84F7 3AØ58Ø             LDA     TSTAT   ;GET TALKER STATE
84FA E6Ø1               ANI     1       ;SEE IF ADDRESSED
84FC 47         TIDL:   MOV     B,A
84FD 3AØ68Ø             LDA     LSTAT   ;GET LISTENER STATE
85ØØ E6Ø3               ANI     3       ;SEE IF ADDRESSED OR ACTIVE
85Ø2 CAØ785             JZ      LIDL    ;..LISTENER IDLE
85Ø5 3EØ2               MVI     A,2     ;PUT LISTENER NOT-IDLE STATE IN BIT 1
```

```
8507 BØ         LIDL:   ORA     B         ;OR IN TALKER STATE AT BIT Ø
8508 47                 MOV     B,A       ;SAVE IT IN B.
8509 3AØ98Ø             LDA     PSTAT     ;GET PARALLEL POLL STATE
85ØC E6Ø1               ANI     1
85ØE Ø7                 RLC
85ØF Ø7                 RLC
851Ø BØ                 ORA     B
8511 47                 MOV     B,A
8512 3AØ88Ø             LDA     RSTAT     ;GET REMOTE-LOCAL STATE
8515 E618               ANI     18H
8517 BØ                 ORA     B
8518 47                 MOV     B,A
8519 3AØA8Ø             LDA     CSTAT     ;GET CONTROLLER STATE
851C E6ØF               ANI     ØFH
851E CA2385             JZ      CIDL      ;CONTROLLER IS IN IDLE STATE
8521 3E4Ø               MVI     A,4ØH     ;SHOW CONTROLLER NOT IDLE
8523 BØ         CIDL:   ORA     B         ;GET THE REST OF THE STATE INFORMATION
8524 C1                 POP     B         ;RESTORE BC
8525 21Ø58Ø             LXI     H,TSTAT   ;POINT HL TO FIRST STATE TABLE ENTRY
8528 C9                 RET
                ;
                ;***********************************************************
                ;
                ;       XCTRL   EXTERNAL CONTROLLER RESPONSE ROUTINE
                ;
                ;               THIS ROUTINE LOOKS AT THE COMMANDS PRESENTED
                ;               BY AN EXTERNAL CONTROLLER AND UPDATES THE
                ;               STATE OF THE INTERFACE AS NECESSARY.
                ;
                ;***********************************************************
                ;
8529 E5         XCTRL:  PUSH    H         ;SAVE USER JUMP TABLE ADDRESS
852A 6Ø                 MOV     H,B
852B 69                 MOV     L,C
852C 22ØE8Ø             SHLD    JMPAD
852F E1                 POP     H
853Ø 3AØD8Ø     XCTLØ:  LDA     GIMTC     ;SET UP ACCEPTOR HANDSHAKE
8533 E69F               ANI     9FH       ;BY SETTING NRFD, NDAC LOW (TRUE)
8535 F68Ø               ORI     8ØH       ;AND DAV HIGH (PASSIVE FALSE)
8537 CD3B82             CALL    COMND
853A 3EFF               MVI     A,ØFFH    ;CLEAR DATA LINES
853C D37E               OUT     DATPT
853E 3EF6               MVI     A,ØF6H    ;CLEAR XATN BIT IN ISR, LEAVE INTERRUPTS
854Ø D37C               OUT     ISRPT     ;  DISABLED
8542 CDF985     XCTL1:  CALL    PI        ;CHECK FOR LOCKUP DUE TO POC OR IFC
8545 DB7D               IN      CMDPT     ;NOW CHECK FOR ATN
8547 E6Ø8               ANI     8
8549 C2AØ85             JNZ     XCDUN
854C DB7D               IN      CMDPT     ;WAIT UNTIL DAV IS HIGH (PASSIVE FALSE)
854E E68Ø               ANI     8ØH
855Ø CA4285             JZ      XCTL1
8553 3AØD8Ø             LDA     GIMTC     ;NOW SET NRFD HIGH (WE'RE READY)
8556 F64Ø               ORI     4ØH
8558 CD3B82             CALL    COMND
855B CDF985     DAVT:   CALL    PI        ;WAIT FOR DAV LOW (TRUE)
855E DB7D               IN      CMDPT
856Ø E6Ø8               ANI     8         ;XATN TRUE?
8562 C2AØ85             JNZ     XCDUN     ;..NO, SO QUIT THIS ROUTINE
8565 DB7D               IN      CMDPT
8567 E68Ø               ANI     8ØH
8569 C25B85             JNZ     DAVT
856C 3AØD8Ø             LDA     GIMTC     ;SET NRFD, NDAC LOW (TRUE)
856F E69F               ANI     9FH
```

```
8571 CD3B82             CALL    COMND
8574 DB7E               IN      DATPT   ;GET THE COMMAND FROM THE EXTERNAL CONTROLLER
8576 2F                 CMA             ;THE 488 BUS USES NEGATIVE LOGIC
8577 E67F               ANI     7FH     ;STRIP PARITY BIT
8579 47                 MOV     B,A     ;SAVE THE COMMAND IN REGISTER B
857A 32238Ø             STA     LBYTE   ;AND IN LAST BYTE REGISTER
857D 3AØD8Ø             LDA     GIMTC   ;TELL THE CONTROLLER WE GOT IT
858Ø F62Ø               ORI     2ØH     ;BY SETTING NDAC HIGH (FALSE)
8582 CD3B82             CALL    COMND
8585 CDF985     DAVF:   CALL    PI      ;NOW WAIT FOR DAV FALSE (HANDSHAKE COMPLETE)
8588 DB7D               IN      CMDPT
858A E68Ø               ANI     8ØH
858C CA8585             JZ      DAVF
858F 3AØD8Ø             LDA     GIMTC   ;COMPLETE HANDSHAKE BY SETTING NDAC LOW
8592 E69F               ANI     9FH
8594 CD3B82             CALL    COMND
8597 CD7383             CALL    UPD8    ;FIGURE OUT WHAT THE COMMAND MEANS
859A CDC184             CALL    UBRAK   ;SEE IF USER WANTS CONTROL OF S-1ØØ
859D C34285             JMP     XCTL1   ;GET THE NEXT COMMAND
                ;
85AØ CD3383     XCDUN:  CALL    ADDRES  ;FINISH ADDRESSING TALK/LISTEN FCNS
                                        ;  OF P&T-488
85A3 3EF6               MVI     A,ØF6H  ;CLEAR THE XATN BIT IN THE ISR
85A5 D37C               OUT     ISRPT
85A7 3AØ58Ø             LDA     TSTAT   ;CHECK TO SEE IF IN SERIAL POLL MODE
85AA E6Ø4               ANI     4
85AC C8                 RZ              ;..NO, GO BACK TO CALLING ROUTINE
85AD 3AØ58Ø             LDA     TSTAT   ;ARE WE ADDRESSED AS THE TALKER?
85BØ E6Ø1               ANI     1
85B2 CAE485             JZ      NTLKR   ;..NO, WAIT FOR NEXT COMMAND
85B5 2AØE8Ø             LHLD    JMPAD   ;PUT USER JUMP TABLE ADDRESS IN BC
85B8 44                 MOV     B,H
85B9 4D                 MOV     C,L
85BA 21258Ø             LXI     H,XSPRS ;POINT TO EXTERNAL CONTROLLER SERIAL
                                        ;  POLL RESPONSE BYTE BUFFER
85BD 3AØ38Ø             LDA     SPSTS   ;GET SERIAL POLL STATUS BYTE
85CØ E6BF               ANI     ØBFH    ;MAKE IT SERVICE NOT REQUESTED
85C2 77                 MOV     M,A     ;AND PUT INTO BUFFER
85C3 3AØ78Ø             LDA     SSTAT   ;ARE WE REQUESTING SERVICE?
85C6 E63Ø               ANI     3ØH
85C8 CADF85             JZ      SRSP    ;..NO
85CB 3AØD8Ø             LDA     GIMTC   ;CLEAR SRQ LINE
85CE F6Ø4               ORI     4
85DØ CD3B82             CALL    COMND
85D3 3E2Ø               MVI     A,2ØH   ;AND PUT INTO THE AFFIRMATIVE POLL
                                        ;  RESPONSE (APRS) STATE
85D5 32Ø78Ø             STA     SSTAT
85D8 3E4Ø               MVI     A,4ØH   ;SET SERIAL RESPONSE TO SERVICE REQUEST
                                        ;  ACKNOWLEDGED
85DA B6                 ORA     M
85DB 77                 MOV     M,A
85DC C3DF85             JMP     SRSP
                ;
85DF 54         SRSP:   MOV     D,H     ;MESSAGE IS ONLY THE ONE BYTE
85EØ 5D                 MOV     E,L
85E1 CD988Ø             CALL    TALK    ;SAY THE RESPONSE MESSAGE
85E4 3AØD8Ø     NTLKR:  LDA     GIMTC   ;RELEASE NRFD, NDAC SO THE THE ADDRESSED
85E7 F66Ø               ORI     6ØH     ;  TALKER CAN RESPOND WITH ITS SERIAL POLL
85E9 CD3B82             CALL    COMND   ;  RESPONSE BYTE
85EC CDF985     NTLK1:  CALL    PI      ;CHECK FOR IFC OR POC
85EF DB7D               IN      CMDPT   ;WAIT FOR RE-APPLICATION OF EXTERNAL ATN
85F1 E6Ø8               ANI     8       ;LOOK AT ONLY XATN
85F3 CA3Ø85             JZ      XCTLØ   ;..XATN TRUE, SO GO TO EXTERNAL
                                        ;  CONTROLLER ROUTINE
85F6 C3EC85             JMP     NTLK1   ;REPEAT LOOP UNTIL NEXT COMMAND COMES
```

```
                ;
85F9 DB7C       PI:     IN      ISRPT   ;CHECK FOR POC OR IFC
85FB E6Ø1               ANI     1       ;LOOK AT ONLY POC
85FD CABE84             JZ      UPOC
86ØØ DB7C               IN      ISRPT
86Ø2 E61Ø               ANI     1ØH     ;LOOK AT ONLY IFC
86Ø4 CA9F84             JZ      UIFC
86Ø7 C9                 RET
                ;
                ;*************************************************************
                ;
                ;       SPQRY   SERIAL POLL QUERY
                ;
                ;       SENDS OUT THE COMMANDS UNL (UNIVERSAL UNLISTEN),
                ;       SPE (SERIAL POLL ENABLE), THEN THE TALK ADDRESSES
                ;       THAT ARE IN ITS BUFFER (BY CALLING CNTRL).  UPON
                ;       RETURN TO SPQRY, THE COMMAND SPD (SERIAL POLL
                ;       DISABLE) IS SENT, THEN RETURNS TO THE CALLING PROGRAM
                ;
                ;*************************************************************
                ;
                ;
86Ø8 221E8Ø     SPQRY:  SHLD    SBPTR   ;INITIALIZE SERIAL POLL BUFFER POINTER
86ØB EB                 XCHG
86ØC 222Ø8Ø             SHLD    SBEND   ;STORE END ADDRESS OF SERIAL POLL BUFFER
86ØF 6Ø                 MOV     H,B
861Ø 69                 MOV     L,C
8611 22ØE8Ø             SHLD    JMPAD   ;STORE USER JUMP TABLE ADDRESS
8614 21538Ø             LXI     H,BSPE  ;POINT TO UNL, SPE MESSAGE
8617 54                 MOV     D,H
8618 5D                 MOV     E,L
8619 13                 INX     D       ;MESSAGE IS ONLY THE TWO BYTES
861A CD9282             CALL    CTRL    ;SEND THE TWO COMMANDS BUT DO NOT
                                        ;  RELEASE THE ATN LINE
861D 2AØE8Ø     SPQ1:   LHLD    JMPAD   ;GET ADDRESS OF USER'S JUMP TABLE
862Ø 44                 MOV     B,H
8621 4D                 MOV     C,L     ;AND PUT INTO BC
8622 21568Ø             LXI     H,BUNT  ;POINT TO "UNT" MESSAGE
8625 54                 MOV     D,H
8626 5D                 MOV     E,L
8627 CD9282             CALL    CTRL    ;AND SEND IT BEFORE THE TALK ADDR
862A 2AØE8Ø             LHLD    JMPAD   ;GET THE ADDR OF THE USER'S JUMP TABLE
862D 44                 MOV     B,H
862E 4D                 MOV     C,L
862F 2A2Ø8Ø             LHLD    SBEND
8632 EB                 XCHG            ;POINT TO SERIAL POLL BUFFER
8633 2A1E8Ø             LHLD    SBPTR   ;  (TALK ADDRESSES) AND SEND THEM
                                        ;  ONE BY ONE
                ;
                ;  SEE IF THERE IS ANOTHER ADDRESS IN THE SERIAL POLL BUFFER
                ;
8636 7C                 MOV     A,H
8637 BA                 CMP     D
8638 C24Ø86             JNZ     NSPEND  ;..NOT END OF BUFFER
863B 7D                 MOV     A,L
863C BB                 CMP     E
863D CA4B86             JZ      NSPSEC  ;..END OF BUFFER, THUS THERE IS NO
                                        ;  SECONDARY ADDRESS
                ;
                ;  THERE IS ANOTHER ADDRESS: NOW SEE IF IT IS A SECONDARY ADDR
                ;
864Ø 23         NSPEND: INX     H
8641 7E                 MOV     A,M
```

```
8642 FE6Ø              CPI     6ØH
8644 DA4B86            JC      NSPSEC  ;..NOT A SECONDARY ADDRESS
8647 EB                XCHG            ;..IT IS A SECONDARY ADDR, SO SEND IT ALSO
8648 C34F86            JMP     SENDSP
                ;
864B 2A1E8Ø     NSPSEC: LHLD   SBPTR   ;SEND ONLY THE ONE BYTE
864E EB                XCHG
864F D5         SENDSP: PUSH   D       ;SAVE ADDR OF LAST BYTE SENT
865Ø 2A1E8Ø            LHLD    SBPTR   ;POINT TO FIRST BYTE TO BE SENT
8653 CD7C82            CALL    CNTRL   ;ACTUALLY SEND THE ADDRESS(ES)
                ;
                ;  NOW LISTEN TO THE RESPONSE SENT BY THE ADDRESSED TALKER
                ;
8656 3AØ58Ø            LDA     TSTAT   ;ARE WE ADDRESSED TO TALK?
8659 E6Ø1              ANI     1
865B C29586            JNZ     WETLK   ;..YES
865E 3AØD8Ø            LDA     GIMTC   ;..NO, SO BECOME A LISTENER
8661 F6D8              ORI     ØD8H    ;SET DAV, NRFD, IFC, ATN FALSE
8663 E6DF              ANI     ØDFH    ;AND NDAC TRUE (LOW)
8665 CD3B82            CALL    COMND
8668 CD4182     SPQ2:  CALL    PAI     ;WAIT UNTIL DAV IS TRUE (LOW)
866B DB7D              IN      CMDPT
866D E68Ø              ANI     8ØH
866F C26886            JNZ     SPQ2
8672 3AØD8Ø            LDA     GIMTC   ;RESPOND TO TALKER WITH NRFD
8675 E69F              ANI     9FH
8677 CD3B82            CALL    COMND
867A DB7E              IN      DATPT   ;GET THE SERIAL POLL STATUS BYTE
867C 2F                CMA             ;488 USES NEGATIVE LOGIC
867D 32228Ø            STA     SPRSP   ;AND SAVE IT FOR LATER USE
868Ø 3AØD8Ø            LDA     GIMTC
8683 F62Ø              ORI     2ØH     ;NOW MAKE NDAC FALSE (HIGH)
8685 CD3B82            CALL    COMND
8688 CD4182     SPQ3:  CALL    PAI     ;WAIT FOR DAV FALSE (HIGH)
868B DB7D              IN      CMDPT
868D E68Ø              ANI     8ØH
868F CA8886            JZ      SPQ3
8692 C3FB86            JMP     SPQ7
                ;
8695 3AØ78Ø     WETLK: LDA     SSTAT   ;ARE WE THE ONE REQUESTING SERVICE?
8698 E63Ø              ANI     3ØH
869A CAB386            JZ      NSPRQ   ;..NO, WE'RE NOT IT
869D 3AØD8Ø            LDA     GIMTC   ;CLEAR THE SRQ BIT
86AØ F6Ø4              ORI     4
86A2 CD3B82            CALL    COMND
86A5 3E2Ø              MVI     A,2ØH   ;AND PUT INTO THE AFFIRMATIVE POLL
                                       ;  RESPONSE (APRS) STATE
86A7 32Ø78Ø            STA     SSTAT
86AA 3AØ38Ø            LDA     SPSTS   ;GET THE SERIAL POLL STATUS BYTE
86AD 2F                CMA             ;488 USES NEGATIVE LOGIC
86AE E6BF              ANI     ØBFH    ;ZERO BIT 6 (DIO7 ON 488 BUS)
86BØ C3B986            JMP     WTLK1
                ;
86B3 3AØ38Ø     NSPRQ: LDA     SPSTS   ;GET SERIAL POLL STATUS BYTE
86B6 2F                CMA             ;488 USES NEGATIVE LOGIC
86B7 F64Ø              ORI     4ØH     ;MAKE BIT 6 NON-ZERO (WE DON'T NEED SERVICE)
86B9 D37E       WTLK1: OUT     DATPT   ;PUT MESSAGE ON DATA LINES
86BB DB7E              IN      DATPT   ;GET SERIAL POLL RESPONSE FROM 488 BUS
86BD 2F                CMA             ;488 USES NEGATIVE LOGIC
86BE 32228Ø            STA     SPRSP   ;AND SAVE SERIAL POLL RESPONSE
86C1 3AØD8Ø            LDA     GIMTC
86C4 F6D8              ORI     ØD8H    ;SHOW LISTENER READY, TALKER NOT
86C6 CD3B82            CALL    COMND
```

```
86C9 CD4182    WTLK2:  CALL   PAI      ;CHECK FOR POC/IFC
86CC DB7D              IN     CMDPT    ;WAIT UNTIL NRFD FALSE
86CE E64Ø              ANI    4ØH
86DØ CAC986            JZ     WTLK2    ;SOMEBODY ELSE IS SLOWING US DOWN
86D3 3AØD8Ø            LDA    GIMTC
86D6 E67F              ANI    7FH      ;MAKE DAV TRUE (TALKER SAYING IT)
86D8 CD3B82            CALL   COMND
86DB E6BF              ANI    ØBFH     ;THEN NRFD TRUE (LISTENER GETTING IT)
86DD CD3B82            CALL   COMND
86EØ F62Ø              ORI    2ØH      ;NDAC FALSE (LISTENER GOT IT)
86E2 CD3B82            CALL   COMND
86E5 CD4182    WTLK3:  CALL   PAI      ;CHECK FOR POC/IFC
86E8 DB7D              IN     CMDPT    ;WAIT UNTIL NDAC FALSE
86EA E62Ø              ANI    2ØH
86EC CAE586            JZ     WTLK3
86EF 3AØD8Ø            LDA    GIMTC
86F2 F68Ø              ORI    8ØH      ;DAV FALSE (TALKER REMOVING DATA)
86F4 CD3B82            CALL   COMND
86F7 3EFF              MVI    A,ØFFH   ;RELEASE THE 488 DATA LINES
86F9 D37E              OUT    DATPT
86FB D1        SPQ7:   POP    D        ;GET BFR ADDR OF LAST BYTE SENT
86FC 3A228Ø            LDA    SPRSP    ;HAVE WE FOUND THE NEEDY DEVICE YET?
86FF E64Ø              ANI    4ØH
87Ø1 C21987            JNZ    SPQ9     ;..YES, SO TERMINATE POLL
87Ø4 2A2Ø8Ø            LHLD   SBEND    ;ANYTHING LEFT IN THE BUFFER?
87Ø7 7A                MOV    A,D
87Ø8 BC                CMP    H
87Ø9 C21187            JNZ    SPQ8     ;..YES, SO CONTINUE THE POLL
87ØC 7B                MOV    A,E
87ØD BD                CMP    L
87ØE CA1987            JZ     SPQ9     ;..NO, SO TERMINATE POLL
8711 13        SPQ8:   INX    D        ;POINT TO NEXT ENTRY IN BUFFER
8712 EB                XCHG
8713 221E8Ø            SHLD   SBPTR    ;AND UPDATE THE BUFFER POINTER
8716 C31D86            JMP    SPQ1     ;THEN POLL NEXT DEVICE
               ;
8719 2AØE8Ø    SPQ9:   LHLD   JMPAD    ;GET ADDRESS OF USER'S JUMP TABLE
871C 44                MOV    B,H
871D 4D                MOV    C,L      ;AND PUT INTO BC
871E 21558Ø            LXI    H,BSPD   ;POINT TO SPD MESSAGE
8721 54                MOV    D,H
8722 5D                MOV    E,L      ;COMMAND IS ONLY ONE BYTE
8723 CD7C82            CALL   CNTRL
8726 2A1E8Ø            LHLD   SBPTR    ;POINT TO WINNING ENTRY IN SERIAL POLL BFR
8729 3A228Ø            LDA    SPRSP    ;GET RESPONSE TO SERIAL POLL
872C 47                MOV    B,A      ;AND SAVE IN REG B FOR THE USER
872D 2F                CMA
872E E64Ø              ANI    4ØH      ;SET A REGISTER
873Ø C9                RET             ;GO BACK TO CALLING PROGRAM
               ;
               ;************************************************************
               ;
               ;       SPSRQ   SERIAL POLL REQUEST
               ;
               ;               SET THE SRQ (SERVICE REQUEST) LINE TRUE (LOW)
               ;       THEN DETERMINE IF P&T 488 IS THE CONTROLLER-IN-CHARGE.
               ;       IF SO, JMP TO SVCRQ.
               ;       IF NOT, WAIT UNTIL EXTERNAL CONTROLLER RESPONDS WITH
               ;       A SERIAL POLL.  ANSWER THE POLL, THEN RETURN TO THE
               ;       CALLING PROGRAM.
               ;
               ;************************************************************
               ;
```

```
8731 6Ø        SPSRQ:  MOV    H,B
8732 69                MOV    L,C       ;SAVE USER JUMP TABLE ADDRESS
8733 22ØE8Ø            SHLD   JMPAD
8736 3E1Ø              MVI    A,1ØH     ;UPDATE SERVICE REQUEST STATE BYTE TO
8738 32Ø78Ø            STA    SSTAT     ;  THE SERVICE REQUEST (SRQS) STATE
873B 3AØA8Ø            LDA    CSTAT     ;PUT THE CONTROLLER IN THE CSRS STATE
873E E62F              ANI    2FH       ;  IF THE P&T-488 IS THE CONTROLLER
874Ø CA4887            JZ     NCTRL
8743 F61Ø              ORI    1ØH
8745 32ØA8Ø            STA    CSTAT
8748 DB7C      NCTRL:  IN     ISRPT     ;SEE IF LOCKED UP DUE TO CHANGE IN XATN
874A E6Ø8              ANI    8
874C C25C87            JNZ    NLOK      ;..NOT LOCKED
874F 3EFF              MVI    A,ØFFH    ;PRESERVE HANDSHAKE LOCK, BUT RELEASE
8751 CD3B82            CALL   COMND     ;  XATN BIT IN ISR
8754 3EBF              MVI    A,ØBFH    ;MAKE ONLY NRFD TRUE
8756 D37D              OUT    CMDPT
8758 3EF7              MVI    A,ØF7H    ;RELEASE XATN BIT IN ISR
875A D37C              OUT    ISRPT
875C 3AØD8Ø    NLOK:   LDA    GIMTC
875F E6FB              ANI    ØFBH      ;MAKE SRQ TRUE (LOW)
8761 CD3B82            CALL   COMND
8764 3AØA8Ø            LDA    CSTAT     ;CONTROLLER IN IDLE STATE AND NOT SYSTEM
                                        ;  SYSTEM CONTROL ACTIVE?
8767 E62F              ANI    2FH
8769 C2BB84            JNZ    USRQ      ;..NO, LET THE USER SERVICE THIS
876C C3E485            JMP    NTLKR     ;..YES, SO WAIT FOR CONTROLLER TO DO
                                        ;  A SERIAL POLL
               ;
               ;************************************************************
               ;
               ;       SPIDL           PUT SRQ FCN IN IDLE STATE
               ;
               ;************************************************************
876F 3AØD8Ø    SPIDL:  LDA    GIMTC     ;RELEASE SRQ LINE
8772 F6Ø4              ORI    4
8774 CD3B82            CALL   COMND
8777 97                SUB    A         ;PUT SRQ FCN IN IDLE MODE
8778 32Ø78Ø            STA    SSTAT
877B C9                RET
               ;
               ;************************************************************
               ;
               ;       PPQRY           PERFORM A PARALLEL POLL
               ;
               ;************************************************************
877C CD4182    PPQRY:  CALL   PAI       ;CHECK FOR POC, XIFC, XATN
                                        ;  RESET XATN IF ATN NO LONGER TRUE
877F 3AØD8Ø            LDA    GIMTC     ;GET IMAGE OF WHAT'S ON COMMAND LINES
8782 E6F7              ANI    ØF7H      ;MAKE ATN TRUE
8784 CD3B82            CALL   COMND     ;DO IT
8787 E6F6              ANI    ØF6H      ;NOW MAKE EOI TRUE ALSO
8789 CD3B82            CALL   COMND
878C DB7E              IN     DATPT     ;GET THE RESPONSE TO THE PARALLEL POLL
878E 2F                CMA              ;488 USES NEGATIVE LOGIC
878F 32238Ø            STA    LBYTE     ;SAVE THE RESPONSE
8792 3AØD8Ø            LDA    GIMTC
8795 F6Ø1              ORI    1         ;MAKE EOI FALSE
8797 CD3B82            CALL   COMND
879A F6Ø8              ORI    8         ;MAKE ATN FALSE
879C CD3B82            CALL   COMND
879F 3AØA8Ø            LDA    CSTAT     ;PUT CONTROLLER IN STANDBY
87A2 E6FØ              ANI    ØFØH      ;KEEP SYSTEM CONTROL/SRQ STATES
```

```
87A4 F6Ø6                ORI     6
87A6 32ØA8Ø              STA     CSTAT
87A9 3A238Ø              LDA     LBYTE    ;GET THE RESPONSE AGAIN
87AC C9                  RET
                ;
                ;***********************************************************
                ;
                ;       PISTT               SET "IST" MESSAGE TRUE AND PUT THE
                ;                           PROPER PARALLEL POLL RESPONSE MESSAGE
                ;                           IN THE PARALLEL POLL RESPONSE REGISTER
                ;
                ;***********************************************************
                ;
87AD 3AØ98Ø     PISTT:   LDA     PSTAT    ;SET IST BIT TO 1 (TRUE)
87BØ F6Ø2                ORI     2
87B2 32Ø98Ø              STA     PSTAT
87B5 CDC587              CALL    PPNBL    ;CALCULATE THE RESPONSE BYTE (PPR)
                                          ;  AND PUT FCN INTO STANDBY (PPSS)
87B8 C9                  RET
                ;
                ;***********************************************************
                ;
                ;       PISTF               SET "IST" MESSAGE FALSE AND PUT THE
                ;                           PROPER PARALLEL POLL RESPONSE MESSAGE
                ;                           IN THE PARALLEL POLL RESPONSE REGISTER
                ;
                ;***********************************************************
                ;
87B9 3AØ98Ø     PISTF:   LDA     PSTAT    ;SET IST BIT TO Ø (FALSE)
87BC E6FD                ANI     ØFDH
87BE 32Ø98Ø              STA     PSTAT
87C1 CDC587              CALL    PPNBL    ;CALCULATE THE RESPONSE BYTE (PPR)
                                          ;  AND PUT FCN INTO STANDBY (PPSS)
87C4 C9                  RET
                ;
                ;***********************************************************
                ;
                ;       PPNBL
                ;  THIS ROUTINE CALCULATES THE PARALLEL POLL RESPONSE WHICH
                ;  CORRESPONDS TO THE FOUR LOW-ORDER BITS OF PPRSP.  IT THEN
                ;  DETERMINES WHETHER THE PARALLEL POLL RESPONSE IS TO BE TRUE
                ;  OR FALSE ON THE BASIS OF THE IST BIT OF PSTAT.  THE PROPER
                ;  RESPONSE BYTE IS PLACED IN THE PARALLEL POLL RESPONSE REGISTER
                ;  OF THE P&T-488, AND THE PP FUNCTION IS PUT INTO THE STANDBY
                ;  (PPSS) STATE.
                ;
                ;***********************************************************
                ;
87C5 3AØ98Ø     PPNBL:   LDA     PSTAT    ;PUT PP INTO PARALLEL POLL STANDBY (PPSS)
87C8 F6Ø1                ORI     1
87CA 32Ø98Ø              STA     PSTAT
87CD E6Ø2                ANI     2        ;LOOK AT THE IST BIT
87CF 17                  RAL              ;AND PUT RESULT IN FOURTH BIT POSITION
87DØ 17                  RAL
87D1 4F                  MOV     C,A      ;AND SAVE RESULT IN REG C
87D2 3AØ28Ø              LDA     PPRSP    ;GET THE PPE BYTE
87D5 E6Ø8                ANI     8        ;KEEP ONLY THE SENSE BIT
87D7 A9                  XRA     C        ;COMPARE SENSE BIT AND IST BIT
87D8 C2EB87              JNZ     PPCLR    ;..IST<>S, THUS PPR MESSAGE IS FALSE
87DB 3AØ28Ø              LDA     PPRSP    ;GET THE PPE BYTE
87DE E6Ø7                ANI     7        ;KEEP THE LOW THREE BITS
87EØ 4F                  MOV     C,A
87E1 3EØ1                MVI     A,1      ;CALCULATE THE PPR MESSAGE
```

```
87E3 ØD         PPRCAL: DCR     C       ;DECREMENT SHIFT COUNT
87E4 FAEC87             JM      PPRDUN  ;DONE SHIFTING
87E7 87                 ADD     A       ;LEFT SHIFT ONCE MORE
87E8 C3E387             JMP     PPRCAL  ;DO UNTIL DONE
                ;
87EB 97         PPCLR:  SUB     A       ;ZERO A REGISTER
87EC 2F         PPRDUN: CMA             ;PUT INTO PPR PORT (REMEMBER 488 USES
                                        ;NEGATIVE LOGIC, SO THE VALUE WE PUT
                                        ;IN THE PPR PORT IS THE COMPLEMENT OF
                                        ;WHAT THE CONTROLLER WILL SEE WHEN IT
                                        ;DOES A PARALLEL POLL.)
87ED D37F               OUT     PPORT   ;PUT BYTE IN PARALLEL RESPONSE PORT
87EF C9                 RET
                ;
87FØ                    END


                >>>>>           SYMBOL  TABLE           <<<<<

8333 ADDRES     821C BFCHK      8Ø1Ø BPTR       8Ø55 BSPD       8Ø53 BSPE
8Ø56 BUNT       815D BYTL       8Ø1C CBEND      8Ø1A CBPTR      83Ø8 CCNTU
8523 CIDL       82BA CLUP       ØØ7D CMDPT      827C CNTRL      823B COMND
8ØØA CSTAT      8292 CTRL       82C2 CTRL1      82F2 CTRL2      83ØE CTRL6
ØØ7E DATPT      8585 DAVF       8194 DAVH       81C1 DAVL       855B DAVT
ØØ14 DCL        8Ø26 ENTBL      8ØØ4 EOSB       8162 EOST       ØØØ8 GET
848F GIM        8ØØD GIMTC      ØØØ1 GTL        8Ø57 INIT       ØØ7C ISRPT
8ØØE JMPAD      8Ø18 LBEND      8Ø16 LBPTR      8Ø23 LBYTE      81AF LCNTU
8212 LDUN       85Ø7 LIDL       8153 LISTN      ØØ11 LLO        834A LNADR
8258 LPAI       8169 LSET       819C LSN1       8ØØ6 LSTAT      8ØØØ LSTNP
8ØØB LSTNS      8326 NCEND      8748 NCTRL      82ØC NEOS       875C NLOK
8372 NLPRI      843C NLSN       8233 NOFLO      864Ø NSPEND     86B3 NSPRQ
864B NSPSEC     8146 NTEND      8423 NTLK       85EC NTLK1      85E4 NTLKR
81Ø2 NTLST      8353 NTPRI      8Ø8C NXTAD      8241 PAI        8244 PAI1
87B9 PISTF      87AD PISTT      85F9 PI         ØØØ5 PPC        87EB PPCLR
8474 PPIDL      87C5 PPNBL      ØØ7F PPORT      877C PPQRY      87E3 PPRCAL
87EC PPRDUN     8ØØ2 PPRSP      ØØ15 PPU        8ØØ9 PSTAT      826F PUATN
8471 RDCL       8466 RGET       844D RGTL       8428 RLAG       847Ø RLLO
8457 RPPC       8474 RPPU       83C6 RSCG       83DF RSCG1      83F8 RSCG2
84ØE RSCG3      844E RSDC       8486 RSPD       847D RSPE       8ØØ8 RSTAT
84ØF RTAG       846F RTCT       8Ø2Ø SBEND      8Ø1E SBPTR      ØØØ4 SDC
864F SENDSP     ØØ19 SPD        ØØ18 SPE        876F SPIDL      861D SPQ1
8668 SPQ2       8688 SPQ3       86FB SPQ7       8711 SPQ8       8719 SPQ9
86Ø8 SPQRY      8Ø22 SPRSP      8731 SPSRQ      8ØØ3 SPSTS      85DF SRSP
84E1 SRV1       84D1 SRVIS      8ØØ7 SSTAT      8Ø87 STADR      84F6 STATE
8Ø98 TALK       8ØC1 TALK1      81ØE TALK2      8ØØ1 TALKP      8ØØC TALKS
8Ø14 TBEND      8Ø12 TBPTR      812B TCNTU      ØØØ9 TCT        8Ø24 TEOI
84FC TIDL       836A TNADR      8ØØ5 TSTAT      8Ø7B TWIDL      84B5 UATN
84B2 UBFUL      84C1 UBRAK      84AF UDVCL      849F UIFC       ØØ3F UNL
84B8 UNLSN      ØØ5F UNT        8373 UPD8       84BE UPOC       84BB USRQ
84AC UTRGR      8695 WETLK      86B9 WTLK1      86C9 WTLK2      86E5 WTLK3
85AØ XCDUN      853Ø XCTLØ      8542 XCTL1      8529 XCTRL      8Ø25 XSPRS
```

Code Assignments for "Command Mode" of Operation.

(SENT AND RECEIVED WITH ATN TRUE)

② Bits b7 / b6 / b5					0 0 0	MSG ①	0 0 1	MSG	0 1 0	MSG	0 1 1	MSG	1 0 0	MSG	1 0 1	MSG	1 1 0	MSG	1 1 1	MSG
b4 ↓	b3 ↓	b2 ↓	b1 ↓	COLUMN → ROW ↓	0		1		2		3		4		5		6		7	
0	0	0	0	0	NUL		DLE		SP	MLA ASSIGNED TO DEVICE	0	MLA ASSIGNED TO DEVICE	@	MTA ASSIGNED TO DEVICE	P	MTA ASSIGNED TO DEVICE		MEANING DEFINED BY PCG CODE	p	MEANING DEFINED BY PCG CODE
0	0	0	1	1	SOH	GTL	DC1	LLO	!		1		A		Q		a		q	
0	0	1	0	2	STX		DC2		"		2		B		R		b		r	
0	0	1	1	3	ETX		DC3		#		3		C		S		c		s	
0	1	0	0	4	EOT	SDC	DC4	DCL	$		4		D		T		d		t	
0	1	0	1	5	ENQ	PPC ③	NAK	PPU	%		5		E		U		e		u	
0	1	1	0	6	ACK		SYN		&		6		F		V		f		v	
0	1	1	1	7	BEL		ETB		'		7		G		W		g		w	
1	0	0	0	8	BS	GET	CAN	SPE	(		8		H		X		h		x	
1	0	0	1	9	HT	TCT	EM	SPD	)		9		I		Y		i		y	
1	0	1	0	10	LF		SUB		*		:		J		Z		j		z	
1	0	1	1	11	VT		ESC		+		;		K		[		k		{	
1	1	0	0	12	FF		FS		,		<		L		\		l		¦	
1	1	0	1	13	CR		GS		–		=		M		]		m		}	
1	1	1	0	14	SO		RS		.		>		N		^		n		~	
1	1	1	1	15	SI		US		/		?	UNL	O		___	UNT	oo		DEL	

④

ADDRESSED COMMAND GROUP (ACG)

UNIVERSAL COMMAND GROUP (UCG)

LISTEN ADDRESS GROUP (LAG)

TALK ADDRESS GROUP (TAG)

PRIMARY COMMAND GROUP (PCG)

SECONDARY COMMAND GROUP (SCG)

NOTES:
① MSG = INTERFACE MESSAGE
② b_1 = DIO1 . . . b_7 = DIO7
③ REQUIRES SECONDARY COMMAND
④ DENSE SUBSET (COLUMN 2 THROUGH 5). ALL CHARACTERS USED IN BOTH COMMAND & DATA MODES.

Courtesy of Hewlett-Packard Co.

P&T-488 Auxilliary Programs for CP/M †

The program BUSMON monitors and reports all transactions on the IEEE-488 bus. 488TODSK records data sent over the 488 bus into a disk file. DSKTO488 sends the contents of a disk file over the bus as data. HANDSHAK.ASM contains the source code for routines which perform the Source and Acceptor Handshake functions. An example of how to use HANDSHAK.ASM is given in the program SAMPLHS.ASM.

BUSMON

The program BUSMON monitors and reports all transactions which occur on the IEEE-488 bus. The operator can choose two different forms for the report. The **normal** form displays the transactions without any special handling. The other form is **expanded**, which means that non-printing characters are replaced with strings of printable characters. This form is especially useful for those cases where one is trying to distinguish between tabs and spaces, or determine whether line feed precedes carriage return, etc. The form of the report can be selected by typing a character on the console keyboard while the program is running. Once the form has been selected, its action may be repeated by typing any key on the keyboard.

The operator can set BUSMON to stop on one of three different conditions: on each carriage return, line feed, or each character. The condition is selected by using one of the four **stop code** keys. The stop code can be changed at any time by typing the appropriate stop code key. The stop code keys and the corresponding stop conditions are shown in the following table. Note that typing a stop code key will NOT cause a repeat of the previous stop condition, but will invoke a new stop condition. The program starts in the Carriage Return mode.

Expand/Normal Option

N or n Show characters normally

X or x Expand the non-printing characters. Space (20 Hex), Horizontal Tab (9) and Line Feed (0A Hex) are replaced by the strings <SPACE>, <HT> and <LF> respectively. The non-printing character Carriage Return (0D Hex) causes the message <CR> to be printed followed by a carriage return and a line feed. All other non-printing characters are replaced with the two character string of an up arrow followed by a capital letter. Thus the non-printing character 01 Hex is replaced by the string ↑A, while the character 1A Hex is printed as ↑Z.

Stop Codes

Carriage Return Display all transactions up to and including the next carriage return.

Line Feed Display all transactions up to and including the next line feed.

Space Display the next transaction (allows stepping one byte at a time).

G or g Go. Display all transactions continuously without stopping on Line Feed, Carriage Return or next byte.

† CP/M is a trademark of Digital Research

Abort

Control C Abort. Go back to the CP/M command mode.

Console/Printer Switch

Ø Direct all output to the console.
1-9 Direct all output to the system printer.

NOTE: to direct output to both the console and printer, select the console and then press Control P.

IEEE-488 Functions

I or i Assert IFC (perform an Interface Clear).

R or r Make REN true (assert Remote Enable).
L or l Make REN false (all instruments will go to Local mode).

Q or q Make SRQ true (request service).
W or w Make SRQ false (cease requesting service).

P or p Perform a Parallel Poll and report the results.

S or s Show the state of the IEEE-488 lines.

T or t Talk - collect a string of characters from the operator then send it over the bus as a Talker.

C or c Control - collect a string and send it over the bus as a Controller.

NOTE: While collecting a string for Talk or Control the following keys have special meaning:

Control X Delete the string and restart collection. This allows errors to be corrected.

RETURN Terminate the collection of the string. The carriage return is <u>not</u> included in the string.

ESCAPE Put the next character into the string. This allows ESCAPE, RETURN and Control X to be put into the string. For instance, to get the string ?A<ESCAPE>12<RETURN><LINE FEED>, you would type ?A<ESCAPE><ESCAPE>12<ESCAPE><RETURN><LINE FEED><RETURN>. In this example, the string <ESCAPE> means that the ESCAPE key is pressed, not that the 8 keys <, E, S, C, A, P, E and > are pressed. Similarly, <RETURN> and <LINE FEED> mean that the RETURN and LINE FEED keys are used.

Each time the Controller becomes active (asserts ATN active true), a carriage return-line feed is sent to the console, followed by the string **COMMAND:**, followed by another carriage return-line feed pair. Similarly, each time the Controller becomes inactive (ATN is false), a carriage return, line feed, the string **DATA:**, carriage return and a line feed is sent to the console. Thus all characters printed after COMMAND: and before **DATA:** are instructions sent by the Controller, (for example, "?" means

UNLISTEN). All characters printed after DATA: and before COMMAND: are data (otherwise known as device-dependant. messages). Examples are readings from a DVM which has been commanded to be a Talker, etc.

Messages are also printed on the console to indicate occurances of IFC (Interface Clear), indicate a change of the state of the REN (Remote Enable) line, and of the SRQ (Service Request) line. The message **>>> S-1ØØ POC/RESET TRUE <<<** is printed whenever the Power On Clear or the RESET line of the S-1ØØ system becomes true.

Whenever the Controller is active, a descriptive string is substituted for special non-printing messages. For example, **>> GO TO LOCAL <<** is printed when Ø1 Hex is received and ATN is true. The list of messages and the corresponding non-printing characters is as follows:

Character Hex	Message
Ø1	>> GO TO LOCAL <<
Ø4	>> SELECTIVE DEVICE CLEAR <<
Ø5	>> PARALLEL POLL CONFIGURE <<
Ø8	>> GROUP EXECUTE TRIGGER <<
Ø9	>> TAKE CONTROL <<
11	>> LOCAL LOCKOUT <<
14	>> UNIVERSAL DEVICE CLEAR <<
15	>> PARALLEL POLL UNCONFIGURE <<
18	>> SERIAL POLL ENABLE <<
19	>> SERIAL POLL DISABLE <<

The results of this program can be misleading for the following reasons:

1. This program functions as a Listener on the 488 bus. If there were no Listeners on the bus before this routine was run, any Talker would have been unable to say a thing. However, when this routine is run, the Talker has someone to talk to. Thus the operation of the 488 system may be changed by the fact that the Bus Monitor routine is run.

2. This routine is slow compared to the speed that communication on the 488 bus is capable of attaining. Thus 488 throughput may be drastically slowed by using the bus monitor.

3. This routine is incapable of sensing a Parallel Poll issued by another controller, or the response to that Parallel Poll. If it happens that this routine tests the EOI line at the time of a Parallel Poll, it will show the message <END>, even though **ATN** is true.

488TODSK

The program **488TODSK** is used to record all data transactions directly into a CP/M disk file. To use the program type

488TODSK filename.ext x<CR>

where **filename.ext** is the file name and extension of the file into which the data is to be recorded, and x is the option code. Note that there must be one and only one space between 488TODSK and the file name, and also one and only one space between the file name and the option code. The characters <CR> mean that the Carriage Return key is pressed, **not** that the four keys <, C, R and > are pressed.

Three different options are available: none, Z and E. The option E means that the file will be closed and control passed back to the console upon receipt of the 488 END message. The option Z means that the file will be closed and control passed back to the console upon receipt of a Control Z in the data stream (the Control Z is also placed in the file). This option can be useful because CP/M text files are terminated by a Control Z. If no option is selected (that is, a Carriage Return follows the file name), the file can be closed only by pressing Control C on the console. Note that Control C can be used at any time to abort the program: all data received up to the time the Control C was pressed is saved in the file. Some garbage will also appear at the end of the file because the whole buffer is saved in the disk file, and the buffer probably was not filled at the time Control C is pressed.

Error messages are printed on the console if the disk directory is full, the data area is full, or any other disk write error occurs. In each case the function is aborted. If the name of the file is the same as one which is already on the disk, the operator is asked if it is OK to replace the old file. If the operator responds by typing any character other than "Y" or "y", the function is aborted and the old file is left untouched. If the operator responds with either "Y" or "y", the old file is erased and the new one takes its place.

DSKTO488

The program DSKTO488 sends the contents of a CP/M disk file over the 488 bus. The program is called by the string

DSKTO488 filename.ext x

where **filename.ext** is the name of the file that is to be sent and **x** is the option code. Only two options are available: none and Z. The Z option causes the Control Z to be sent with the 488 END message when a Control Z is found in the file, then the program returns control to the console. This can be useful for text files that are terminated by a Control Z. If no option code is selected, the entire file is sent followed by a null with the 488 END message, then control is returned to the console. The program may be aborted at any time by typing Control C on the console.

Error messages are printed on the console if there is no Listener on the bus, if the file is not on the disk, or if an invalid option code is selected. In each case the program is aborted and control is returned to the console.

If you have two systems and want to send a file from one to the other via the 488 bus, you would type

488TODSK filename.ext E<CR>

on the system which is to receive the file, and

DSKTO488 filename.ext<CR>

on the one which is sending the file. (It is not necessary to use the same file name or extension.) Note that the system receiving the file must be started first, otherwise the first byte of the file will be lost or the sending system will complain that there are no listeners.

HANDSHAK

The source file **HANDSHAK.ASM** is actually two subroutines: a routine for Source handshake and a routine for Acceptor handshake. These routines can be useful in special applications where it is desired to use the S-1ØØ system as a Talk Only or Listen Only device, or where increased data rate on the 488 bus is needed. These routines are capable of running much faster than the larger Custom System, CPM488 or 488BAS routines because the larger routines check for the existance of another Controller on the bus, check for excessive time in the handshake cycle, and many other things.

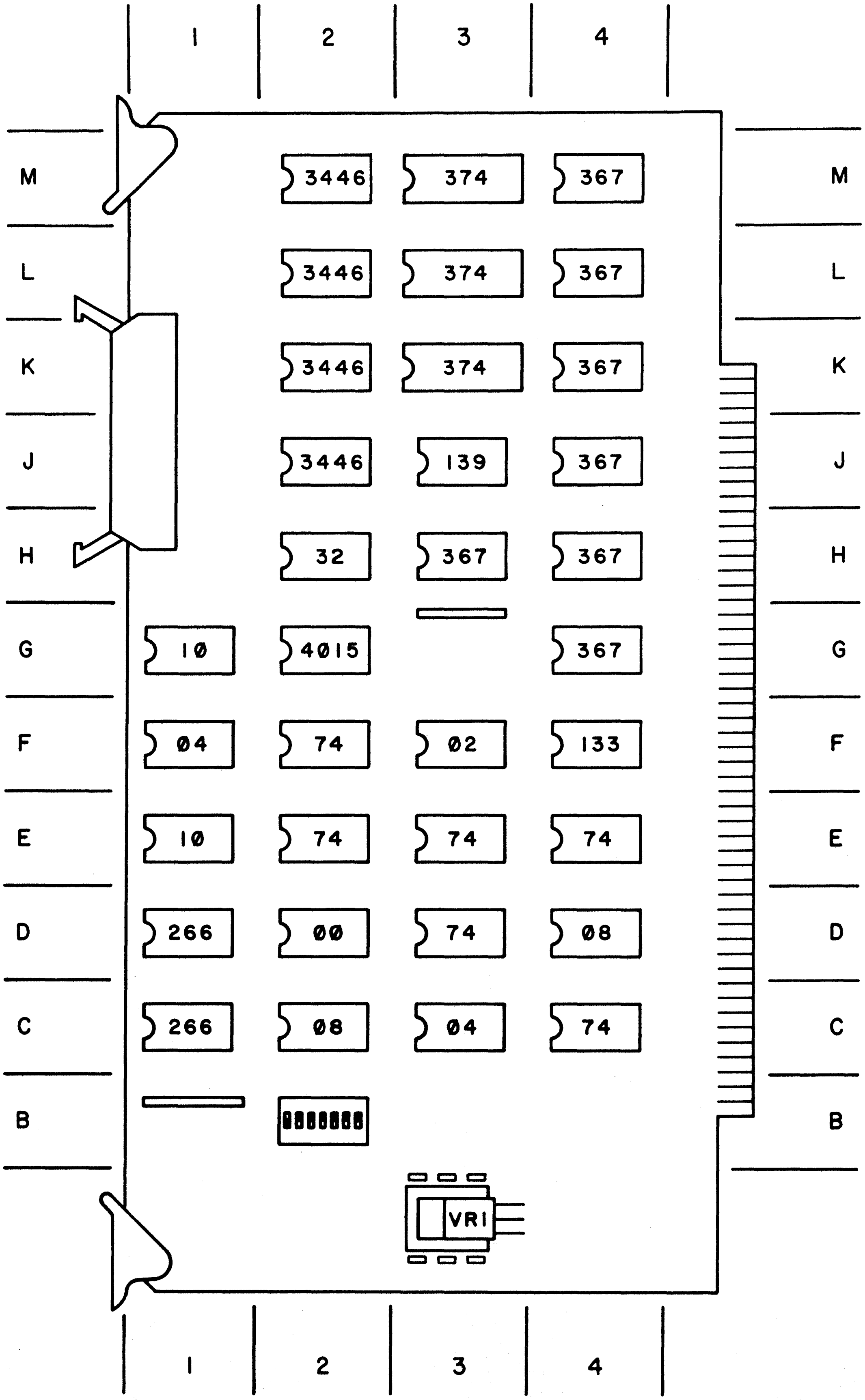
Refer to the chapter titled **Hardware Description** in the P&T-488 manual for information about the bit mapping of the ports and the 488 bus lines.

SAMPLHS

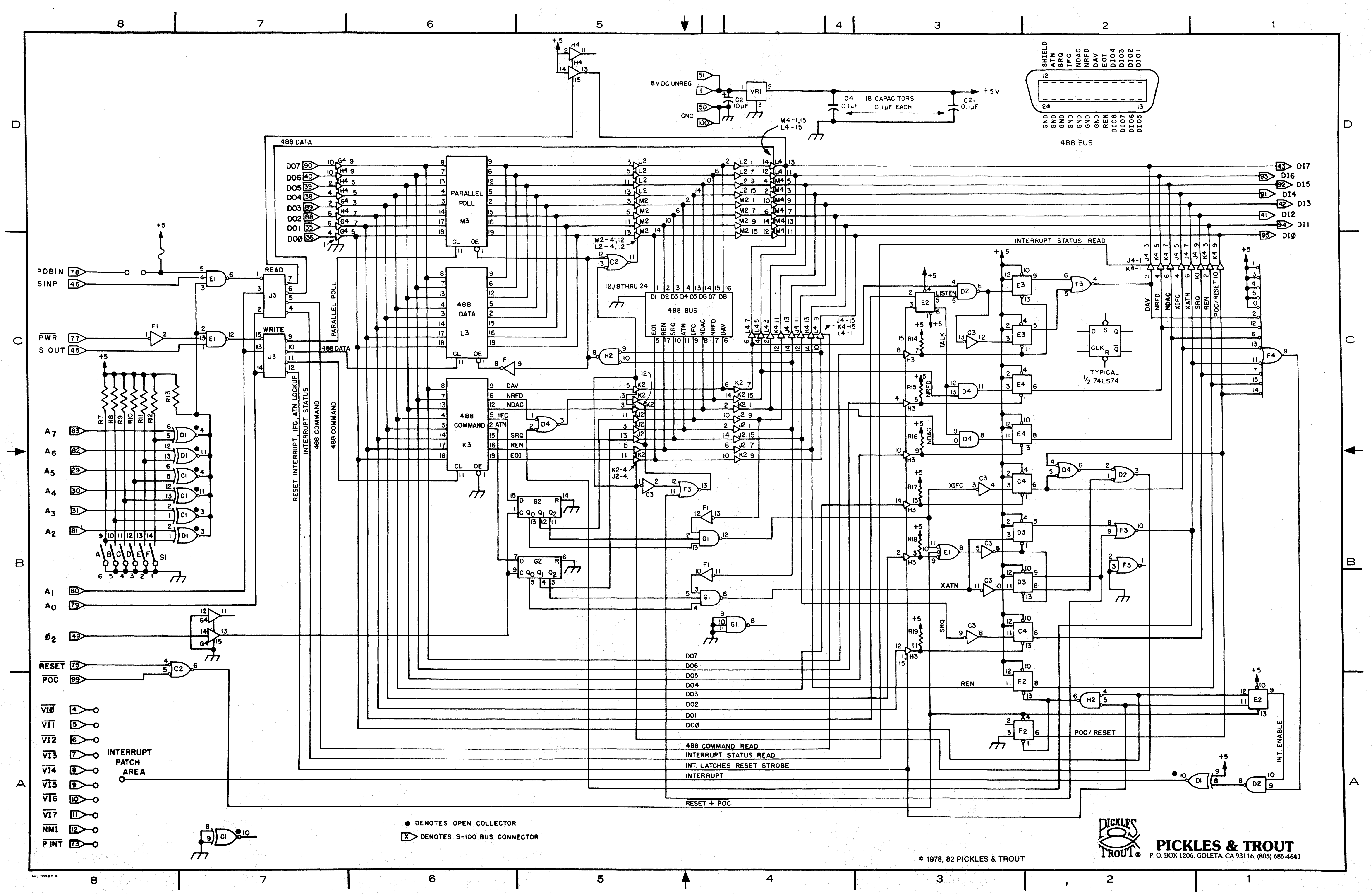
This file contains the source code for a routine which uses the Source, Acceptor and Initialization subroutines in HANDSHAK to take data from the IEEE-488 bus and display it on the console.

```
;*********************************************************
;
;       Source and Acceptor Handshake listings
;
;*********************************************************

ISRPT   EQU     7CH
CMDPT   EQU     ISRPT+1
DATPT   EQU     ISRPT+2
PPORT   EQU     ISRPT+3

MONITR  SET     Ø           ;CP/M warmstart entry
CPMIO   SET     5           ;CP/M I/O entry point

CR      SET     ØDH         ;ASCII  carriage return
LF      SET     ØAH         ;ASCII  line feed
ES      SET     '$'         ;CP/M buffered print string terminator

BUFPRN  SET     9           ;CP/M fcn. number for buffered print
;
;       TALK
;
TLKT:   LDA     GIMTC       ;get the image of the byte last sent
                            ;  to the command line port
        ORI     8           ;make sure that ATN is false (high)
        STA     GIMTC       ;  when do source handshake
;
;*********************************************************
;
;       SOURCE HANDSHAKE
;
;  This routine takes the byte in memory location CHAR and says
;  it on the 488 bus as a Talker.  If either the S-1ØØ RESET
;  or Power On Clear line is or has been true, or if the
;  488 ATN or IFC lines are or have been true, then an error
;  message is printed and the routine jumps to the system
;  monitor.
;
;*********************************************************
;
SRCHS:  LDA     GIMTC       ;get 488 command line image
        ORI     6ØH         ;set NRFD, NDAC high (false)
        CALL    COMND
SRC1:   CALL    INTRPT      ;check for POC, ATN or IFC
        JNZ     BYE         ;..abort if POC, ATN or IFC true
        IN      CMDPT       ;see if there are any listeners
        CMA
        ANI     6ØH         ;check only NRFD, NDAC
        JZ      NOLSN       ;..no listeners error
        ANI     4ØH         ;wait until NRFD is high (false)
        JNZ     SRC1
        LDA     CHAR        ;get the data byte
        CMA                 ;488 uses negative logic
        OUT     DATPT
        LDA     GIMTC
        ANI     7FH     ;make DAV true (low)
        CALL    COMND
SRC2:   CALL    INTRPT  ;check for POC, ATN or IFC
        JNZ     BYE     ;..abort if POC, ATN or IFC true
        IN      CMDPT
        ANI     2ØH     ;look at NDAC line
        JZ      SRC2    ;...data not accepted yet
        LDA     GIMTC
        ORI     81H     ;make DAV & EOI false (high)
        CALL    COMND
        MVI     A,ØFFH
        OUT     DATPT   ;make all data lines passive false
        RET
;
;**************************************************************
;
;       ACCEPTOR HANDSHAKE
;
;   This routine gets one byte from the 488 bus and returns with
;   it in register A.  If either the S-1ØØ RESET or Power On
;   Clear line is or has been true, or if the 488 ATN or IFC
;   lines are or have been true, then an error message is printed
;   and the routine jumps to the system monitor.
;
;**************************************************************
;
ACEPTR: LDA     GIMTC
        ORI     8       ;make ATN false
        ANI     9FH     ;  and NRFD true, NDAC true
        CALL    COMND
        LDA     GIMTC
        ORI     4ØH     ;now make NRFD false
        CALL    COMND
ACEPT1: CALL    INTRPT  ;see if received POC, ATN or IFC
        JNZ     BYE     ;..abort
        IN      CMDPT   ;look at DAV
        ANI     8ØH
        JNZ     ACEPT1  ;..DAV still false
        IN      DATPT   ;get the data
        CMA             ;488 uses negative logic
        MOV     D,A     ;keep the data in register D
        LDA     GIMTC
        ORI     2ØH     ;NDAC false
        ANI     ØBFH    ;NRFD true
        CALL    COMND
ACEPT2: CALL    INTRPT
        JNZ     BYE     ;..abort
        IN      CMDPT   ;wait for DAV false
        ANI     8ØH
        JZ      ACEPT2  ;...DAV still true
        LDA     GIMTC
        ANI     9FH     ;NRFD true, NDAC true
        CALL    COMND
        MOV     A,D     ;put the data back in register A
        RET
;
;       Initialize 488 board
;
;  This routine should be called after every S-1ØØ RESET or
;  Power On Clear
;
INIT:   MVI     A,ØFFH
        OUT     PPORT   ;clear parallel poll response port
        OUT     DATPT   ;  and 488 data port
        CALL    COMND   ;  and 488 control lines and image byte
        SUB     A
        OUT     ISRPT   ;clear Interrupt Service Register
        STA     RETCOD  ;  clear return code
        STA     CHAR    ;  and CHAR
        RET
;
;  COMND keeps track of the last byte that was output to the
;  command port.  It is necessary to keep track of what the
;  P&T-488 interface board is asserting on the bus because
;  the 488 bus is an open-collector wire-or system, so it is
;  not possible to determine what the P&T-488 is asserting
;  on the 488 bus by merely sensing the 488 lines.
;
COMND:  STA     GIMTC   ;update the 488 command line image
        OUT     CMDPT   ;put it on the command lines
        RET
;
;       Check for interrupt due to ATN, IFC or POC
;
;  NOTE:  This function does not reset the interrupts in the
;         Interrupt Service Register (ISR)
;
INTRPT: IN      ISRPT   ;look at the interrupt service register
        RAR             ;put POC bit in carry
        CNC     IPOC    ;..set POC bit in return code byte if
                        ;  no carry
        RAR             ;REN  > CARRY
        RAR             ;SRQ  > CARRY
        RAR             ;ATN  > CARRY
        CNC     IATN    ;..set the XATN bit
        RAR             ;IFC  > CARRY
        CNC     IIFC    ;..set the XIFC bit
        LDA     RETCOD
        ANI     ØFØH    ;look at only POC, IFC and ATN
        RET
;
IPOC:   PUSH    A
        LDA     RETCOD
        ORI     8ØH
ICOM:   STA     RETCOD
        POP     A       ;restore reg A and carry
        RET
;
IATN:   PUSH    A
        LDA     RETCOD
        ORI     20H
        JMP     ICOM
;
IIFC:   PUSH    A
        LDA     RETCOD
        ORI     40H
        JMP     ICOM
;
;       Print the reason for aborting then jump to the monitor
;
BYE:    PUSH    PSW     ;save the error code
        LXI     D,MS2   ;power on clear
        ANI     80H
        CNZ     PRINT
        POP     PSW     ;get the error code again
        PUSH    PSW
        LXI     D,MS3   ;XIFC
        ANI     40H
        CNZ     PRINT
        POP     PSW
        LXI     D,MS4   ;XATN
        ANI     20H
        CNZ     PRINT
        JMP     MONITR
;
;       No listeners present - print error message then
;       jump to the monitor
;
NOLSN:  LXI     D,MS1   ;print no listener msg
;
;       Print error message and return to monitor
;
ERROR:  CALL    PRINT
        JMP     MONITR
;
;       print the line pointed to by DE
;
PRINT:  MVI     C,BUFPRN
        CALL    CPMIO
        RET
;
;
GIMTC:  DB      0       ;image of last byte sent to CMDPT
CHAR:   DB      0
RETCOD: DB      0       ;a byte containing the error code

MS1:    DB      'No listeners on the bus',CR,LF,ES
MS2:    DB      'S-100 POWER ON CLEAR or RESET',CR,LF,ES
MS3:    DB      'Another 488 Controller is asserting IFC true',CR,LF,ES
MS4:    DB      'Another 488 Controller is asserting ATN true',CR,LF,ES
```

```
;**************************************************************
;
;       SAMPLHS.ASM
;
;  This program uses the Acceptor handsahke routine to get a
;  data byte from the IEEE-488 bus and display it on the
;  system console.
;
;**************************************************************
;
        ORG     100H

MONITR  SET     0       ;CP/M warmstart entry point
CPMIO   SET     5       ;CP/M I/O routine entry point

GETCHR  SET     1       ;CP/M function code for console input
PUTCHR  SET     2       ;CP/M function code for console output
CONSTAT SET     11      ;CP/M function code for console status

        LXI     SP,2000H        ;initialize stack pointer
        CALL    INIT            ;initialize the P&T-488 card
LOOP:   CALL    ACEPTR          ;get a byte from the 488 bus
        MOV     E,A             ;put it in register E for CP/M
        MVI     C,PUTCHR        ;function to print on console
        CALL    CPMIO           ;CP/M I/O routine entry point
        MVI     C,CONSTAT       ;look to see if a key is pressed
        CALL    CPMIO
        ANI     1
        JZ      LOOP            ;..no key pressed
        MVI     C,GETCHR        ;get the key
        CALL    CPMIO
        CPI     3               ;CONTROL C?
        JNZ     LOOP            ;..no, so continue getting data
                                ;  from the bus
        JMP     MONITR          ;..yes, so do a warmstart

;**************************************************************
;
;       Insert the Handshake routines here
;
;**************************************************************

        END
```



P & T 488 REV. 81A



● DENOTES OPEN COLLECTOR

X DENOTES S-100 BUS CONNECTOR

PICKLES & TROUT
P. O. BOX 1206, GOLETA, CA 93116, (805) 685-4641

MIL 10950 K