Interfacing STD and GPIB busses

by

Anantha Padmanabha N. Srirama

A Thesis Submitted to the

Graduate Faculty in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Department: Electrical Engineering and Computer Engineering
Major: Computer Engineering

Iowa State University
Ames, Iowa

1987

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

Recent years have seen an explosion in the number of microcomputers. Today, microcomputers are being used in the laboratory environment to acquire and analyze experimental data. Hewlett-Packard (HP) foresaw this need to acquire experimental data by using computers in early 1960s.

Hewlett-Packard developed the HP-IB (GPIB) digital instrumentation bus to fulfill their vision of computerizing instrumentation. The HP-IB is optimized for controlling instruments. It is a flexible instrumentation bus with no restrictions on how the user can interconnect the devices. It can be used in situations where data transmission speeds are below 1 Mbytes/s. The Electrical Engineering and Computer Engineering department at Iowa State University uses many HP instruments in their laboratories, some of these instruments have the HP-IB interface.

It was desirable to build an HP-IB interface card for some of the computers to automate experimental data acquisition. The department also uses Z-80 microprocessor based STD bus computers in the laboratory for instruction. These computers were chosen to implement the HP-IB interface because of their availability and potential for use as teaching aid for future laboratory instruction.

The interface design uses the NEC μPD7210 as the talker, listener and controller integrated circuit (IC). The use of this IC resulted in a reduced component count for the interface card. The card uses only six ICs: a transceiver (transmitter/receiver) chip, a decoder chip, a hex inverter chip, one PD7210, and two HP-IB buffer ICs. A library of Z-80 subroutines was developed to effectively use the interface card.

LITERATURE REVIEW

Introduction to the GPIB Bus

Background

The General Purpose Interface Bus (GPIB) is an asynchronous digital instrumentation bus developed at Hewlett-Packard (HP).[1] At HP it is known by its original name, Hewlett Packard Interface Bus. The HP-IB evolved from their desire to computerize instrumentation during the '60s. It is a carefully designed and defined general purpose digital interface system, which helps the designer to integrate instruments and computers. The bus can be used in any system where:

1. Data exchanged among different interconnected instruments are digital.
2. The number of instruments does not exceed 15.
3. Data transmission speeds are below 1 Mbytes/Sec.
4. The total length of the bus is less than 20 m or 2 m/instrument, whichever is less (when the bus extension technique is not used).

Status today

HP submitted the standard, after extensive in-house testing, to the Institute of Electrical and Electronic Engineers (IEEE) for approval as a commercial standard. IEEE approved and published the document IEEE-488,

---

[1] Most of the material for this section was drawn from references 1-4 and 6-7.

in 1975, describing the standard in detail. Industry started referring to the bus by the number of the document and the name held. The standard is also known as the General Purpose Interface Bus (GPIB). The HP-IB bus standard has also been accepted by the American National Standards Institute (document MC1.1) and the International Electrotechnical Commission (document IEC 625-1). The IEC standard differs only in the mechanical portion of the interface, specifying a different connector.

The interface

The HP-IB is a byte serial and bit parallel bus. The bus contains 16 signal wires and 8 ground wires. The 16 signal lines can be divided into data group containing 8 wires and bus management lines constituting the other 8 lines. The bus does not put any restriction on the way devices are interconnected. Star and linear cabling patterns may be mixed. The data are exchanged asynchronously between the devices, using a three-wire handshake technique. The peak speed of 1Mbytes/s can be achieved only in systems with total length less than 15 m and at least one device/m. In a full size network with 20 m accumulated cable length, speeds of 250-500 Kbytes can be realized. Each instrument on the bus has a unique HP-IB address.

Functions provided by the interface

Every HP-IB instrument must be capable of performing one or more of the following interface functions:

1. Listener - An instrument or device capable of receiving data over the interface when addressed to listen. There can be a

maximum of 14 active listeners simultaneously on the interface. Examples of devices with listener function implemented are: printers, display devices, etc.

2. Talker - An instrument capable of sending data over the interface when addressed to talk. There can be only one active talker at any given instant of time. Examples of devices with talker function implemented are: tape readers, voltmeters, etc.
3. Controller - An instrument capable of specifying a talker and listeners for a data transfer (including itself). A system can have any number of controllers. There can only be one active controller at any given instant of time. A computer with an appropriate Input/Output (I/O) card is a good example of a device implementing the controller function.

A device can be designed with other HP-IB functions of the interface:

1. Source handshake - This function gives the source handshake capability to a device. This guarantees proper transfer of multiline commands (commands that use more than one line) from the controller to acceptor(s). It also guarantees proper transmission of data from talker to listener(s).
2. Acceptor handshake - This function gives the acceptor handshake capability to a device. This guarantees proper reception of the multiline commands sent over the interface by the controller. It also guarantees proper reception of data by the listener(s) connected to the bus.

3. Remote/Local - This function gives the capability to a device by which the device can operate in remote or local mode. Local corresponds to the front panel control of the device and remote corresponds the input of control information through the HP-IB interface. In the remote mode of operation, front panel controls of the device are disabled and the device is controlled by the commands received over the HP-IB. The device can be returned to local control by pushing the local switch in the front panel of the device or by the Go To Local (GTL) interface command. To avoid accidental return of the device to local mode of operation, the controller can disable the local switch by sending the Local LockOut (LLO) command, a multiline command, after placing the device in remote mode.
4. Service request - A device implementing this function has the capability to request service from the active controller for any of the faulty, device dependent, conditions.
5. Parallel poll - A device implementing this function can identify itself if it requires service when the bus controller is polling the devices on the bus, in response to a service request.
6. Device clear - A device implementing this function can reset itself to a predetermined device dependent state, when the system controller desires so. The predetermined state is device dependent and not a part of the interface. It is important to note that this command is different from the

interface clear command, a uniline command, which clears only the interface of the device.

7. Device trigger - A device implementing this function has the capability of starting a programmed task in synchronization with other devices on the bus. The Group Execute Trigger (GET) command, a multiline command, is sent by the active controller to initiate the task.

Control of the interface

All HP-IB bus management and data lines are asserted low (negative logic or low-true logic). Data transmitted over the bus is generally 7-bit ASCII characters. The eighth bit can be used as a parity bit if needed. The data transfer among devices is synchronized by the use of the following three bus management lines:

1. DAV/[2] - DAta Valid. This line is controlled by the source (active talker or the controller) to indicate to the listeners when the data is valid on the bus.
2. NRFD/ - Not Ready For Data. This line is controlled by the active listeners to indicate to the source when the device is ready to accept the next byte of data.
3. NDAC/ - Not Data ACcept. This line is controlled by the active listeners to indicate to the source that data sent over the bus has been accepted.

---

[2] A '/' after the name of a signal line indicates that the signal is asserted low.

Three wire handshake The above mentioned lines are used in the following way to transfer data from the talker to the listener(s).

1. Source places the data on the data lines of the bus and validates it by asserting the DAV/ line.
2. First acceptor receives the data and asserts the NRFD/ line to indicate that it is no longer ready for a new byte.
3. The NDAC/ line of the interface is de-asserted when the data have been received by all the active listeners, as indicated by the assertion of the NRFD/ line.
4. The source invalidates the data on the interface data lines by de-asserting the DAV/ line.
5. All acceptors assert their NDAC/ output to indicate to the source that they are ready for the next byte of data.
6. The sequence is repeated to start the transmission of a new byte over the interface.

figure 1 explains graphically the steps described above.

Control lines The reason for making all the bus management lines of the interface asserted low is to facilitate the use of wired-OR (logical-AND). The low-true logic is also less susceptible to noise in the asserted state. It uses less power during de-asserted or disconnected condition.

The 5 bus management lines are used for the orderly flow of information across the interface. These 5 lines are:

1. ATN/ - ATteNtion. When ATN/ is asserted, the bus is in command mode and all the devices are to treat the data over the bus as

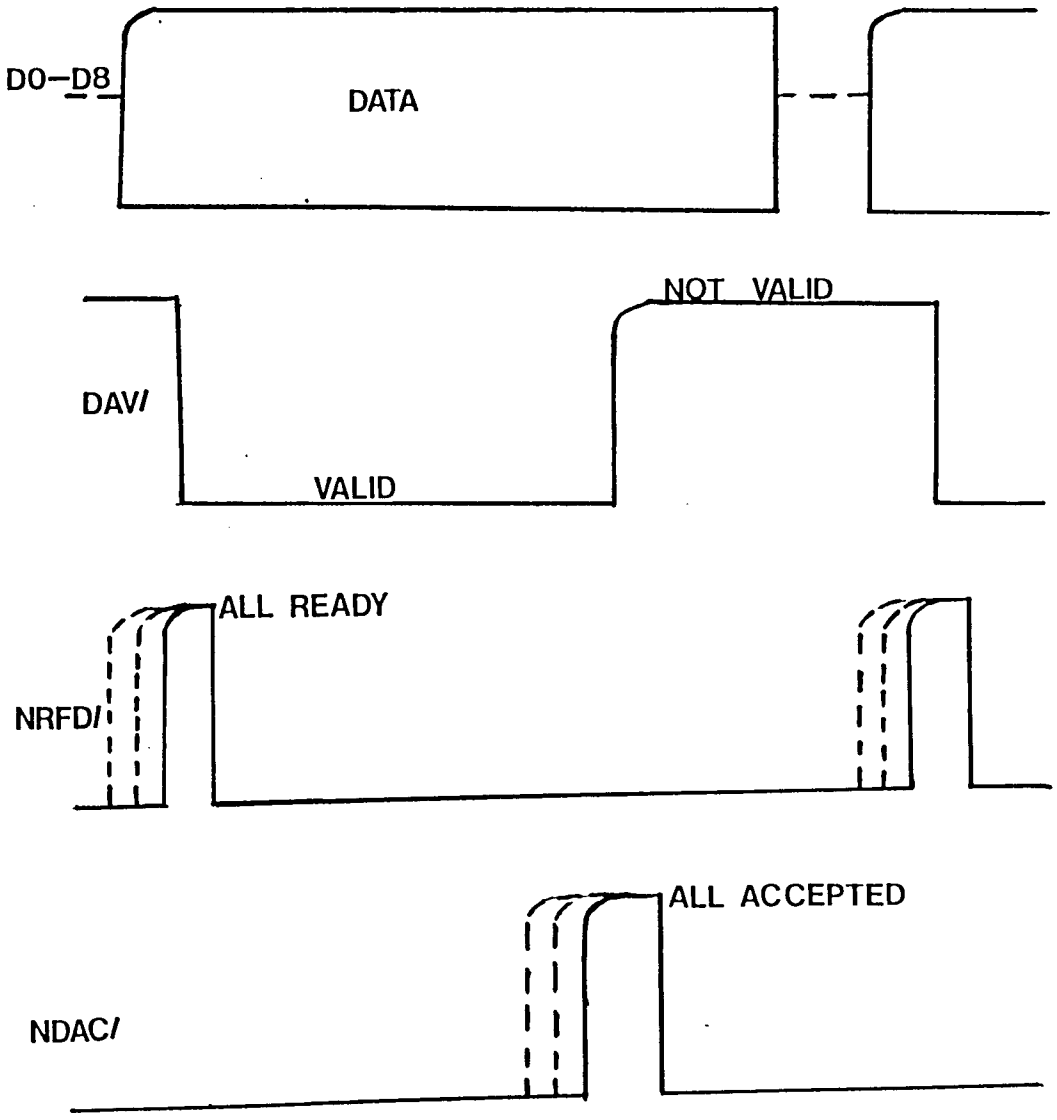

FIGURE 1. Three wire handshake

interface commands. When ATN/ is de-asserted, the bus is in data mode and device dependent data are transmitted over the bus. All devices are required to monitor this line at all times and must respond within 200 μs.

2. IFC/ - InterFace Clear. When asserted, IFC/ causes the devices on the bus to clear the bus interface of the device. All the talker and listeners are unaddressed and the bus is put in idle state with no activity. The interface functions of the device(s) are placed in idle state. Only the bus controller can assert the IFC/ line. All devices must monitor IFC/ at all times and respond to it within 100 μs.
3. SRQ/ - Service ReQuest. When asserted, SRQ/ indicates to the controller that a device connected to the bus requires service. SRQ/ line is generally used by a device to indicate to the controller of a predetermined, generally faulty, condition.
4. REN/ - Remote ENable. A device can be placed in remote mode of operation by sending the My Listen Address (MLA) command, a multiline command, after asserting this line. Thereafter, the device gets input from the HP-IB interface rather than the front panel of the device. It will remain in this mode as long as the REN/ is asserted and returns to local mode automatically on de-assertion of REN/ line by the controller. All devices capable of remote/local operation must monitor the REN/ line at all times. The devices should respond within 100 μs.
5. EOI/ - End Or Identify. When the bus is in data mode, EOI/ is

asserted by the talker to indicate the end of a data transfer (the line is asserted simultaneously with the transmission of the last byte of the sequence). In command mode the controller requests the devices on the bus to output their status bit during a parallel poll by asserting this line.

In the command mode, active controller can do any of the following:

1. Assign the talker and listener by sending appropriate talk and listen addresses respectively.
2. Send universal commands like IFC/, REN/, ATN/, etc.
3. Send addressed commands where only addressed devices respond to the interface commands.
4. Send secondary addresses if an extended talker or listener is connected to the bus.
5. It can send secondary commands like Parallel Poll Enable (PPE) or Parallel Poll Disable (PPD).

The talk and listen addresses of a HP-IB device are generally the same (there is no restriction by the interface specification itself) though they can be different. The talk and listen addresses are resettable by address switch, software, or jumpers. The controller can designate a device to be a listener on the bus by sending the MLA command, which is of the form X01L LLLL (X is a don't care). The five least significant bits (L LLLL) of the MLA command constitute the device address. The controller can designate a device to be a talker by sending the My Talk Address (MTA) command, which is of the form X10T TTTT (X is a don't care). The five least significant bits (T TTTT) of the MTA command

constitute the device address. The controller can request all talkers and listeners to stop talking or listening by sending the untalk (X101 1111) or (X011 1111) unlisten commands. Since the address 31 is used in the untalk and unlisten commands, address 31 (1 1111) is not a valid address for any device on the bus. A new talker can be assigned simply by sending a new talk address on the interface, this automatically untalks the previous talker on the bus. Special care is required with listeners, once addressed to listen a device will stay in that state until an unlisten command is sent over the bus by the active controller. An HP-IB device can have more than one talk or listen address.

Universal commands (multiline) There are five multiline universal commands. Since these commands are universal, devices must respond to these commands whether they are active or not. The five commands are:

1. DCL - Device CLear command (0001 0100). This command causes the devices on the bus to return to a device dependent, predetermined state. It should be noted that IFC/ when asserted clears only the interface of the device, the device stays in the state it was before assertion.
2. LLO - Local LockOut (0001 0001). This command disables the return-to-local switch on the front panel of the devices implementing this function when REN/ line is asserted. If the REN/ line is de-asserted in this state, the return-to-local switch is enabled automatically.
3. SPE - Serial Poll Enable (0001 1000). This command puts all responding talkers in the serial poll mode. When the device is

addressed to talk it will transmit its status byte to the controller.

4. SPD - Serial Poll Disable (0001 1001). This command terminates serial poll of the interface by the controller. The devices return to their normal talker state.
5. PPU - Parallel Poll Unconfigure (0001 0101). This command puts all responding devices capable of parallel poll in idle state.

Addressed commands (multiline) The following addressed commands are provided by the interface specification. Devices implementing any of these functions should be addressed to listen to respond to the commands. They are:

1. GET - Group Execute Trigger (0000 1000). This command causes addressed listener devices with device trigger capability to initiate a preprogrammed task. The GET command provides the means for taking different kinds measurements at the same time.
2. SDC - Selected Device Clear (0000 0100). The SDC command resets the currently addressed device to a predetermined device dependent state. This command has the same effect as the universal command DCL, but only addressed devices respond to the command.
3. GTL - Go To Local (0000 0001). The GTL command causes the currently addressed device to return to front panel control of the device. The device will return to Remote mode when it is addressed to listen again. This command has the same effect as de-asserting the REN/ line on the responding devices. Using

this command the user can selectively place devices in Remote/Local mode.

4. PPC - Parallel Poll Configure (0000 0101). The PPC command causes the addressed listener to enter the parallel poll mode. The controller sends the parallel poll enable following this command.
5. TCT - Take ConTrol (0000 1001). The TCT command is used only in a multicontroller system. The present controller-in-charge can send this command to another controller thereby passing the control of the bus to that controller.

Secondary commands (multiline) There are two secondary commands. A secondary command should be sent after sending a primary command (addressed multiline command). The secondary commands are:

1. PPE - Parallel Poll Enable (0110 XXXX). The parallel poll enable command configures the devices which have received the PPC command, primary addressed command, to respond to a parallel poll.
2. PPD - Parallel Poll Disable (0111 0000). The parallel poll disable command disables devices which have received he PPC command, primary addressed command, from responding to a parallel poll.

Example of operation

Let there be a system with three devices interconnected, one controller (CON) and two devices (A and B) with talk and listen capability. The following paragraph describes the bus transactions which

take place in order to initiate a preprogrammed task in both A and B.

The controller asserts the IFC/ line stopping all activity on bus and clearing the interfaces of A and B. Devices A and B are placed in remote mode of operation by asserting the REN/ line and subsequently sending the listen addresses of A and B. Controller sends the unlisten command to unlistening devices A and B. Devices A and B remain in remote mode of operation because REN/ is still asserted. The controller sends the command MLA with As address, placing A in listener addressed state. The controller designates itself as the the talker and de-asserts the ATN/ line. The device A goes to listener active state on de-assertion of ATN/. A series of bytes (control characters for A) are sent by the talker (CON) to the listener A with the aid of three-wire handshake. The talker asserts the EOI/ line terminating the data transmission. The controller (CON) takes control of the bus asserting the ATN/ line. The controller can send data to device B in the same way as it sent data to device A. The controller can send a GET command, thereby initiating the programmed tasks in devices A and B.

Introduction to the STD Bus

The STD bus is a synchronous 8-bit microprocessor system bus (reference 5). The bus standardizes the electrical and mechanical characteristics of modular microprocessor card systems. The microprocessor card communicates with the memory and I/O cards by using the 56 signal lines provided by the bus. Theses signal lines are laid out as a bus on a printed circuit board with 56 pin sockets (slots) to accept

STD bus cards. The STD bus card has an edge connector at one end which mates with the socket in the printed circuit board. The microprocessor card occupies one of the slots in the bus. The memory and I/O cards can occupy any of the available slots in the bus. Mechanical stability is imparted to the bus by the use of a card cage which holds the cards in place. The bus is internal because it serves as a interconnection path among the microprocessor, memory, and I/O peripheral cards. The microprocessor communicates with the "outside world" through I/O cards. The I/O cards interface with different devices through suitable connectors. The STD bus signal lines can be grouped into the following four groups:

1. Data bus - A collection of 8 lines (D0-D7) capable of transmitting data between different STD bus cards. The STD bus cards employ bidirectional, three-state drivers to buffer the signal on to the bus. The data transfer along the data bus is synchronous with the system clock provided by the microprocessor. The bidirectional data bus uses asserted high logic.
2. Address bus - A collection of 16 lines (A0-A15) which are used to address memory or I/O ports. All the STD bus cards use unidirectional, three-state drivers to buffer the address lines. The address bus operates in synchronization with the system clock provided by the microprocessor. The unidirectional address bus uses asserted high logic. During a direct memory access (DMA), the microprocessor relinquishes its

control over the address lines so that the DMA controller can use them to address memory or I/O ports.

3. Control bus - A collection of 22 lines that are used in bus management. The priority chain in, priority chain out, I/O expansion, and memory expansion signals are asserted high. The rest of the control signals are asserted low. All control signals are unidirectional except IOEXP and MEMEX signals.
4. Miscellaneous - A collection of 10 lines providing the power and ground for the STD bus cards. There are two +5V and two -5V logic power supplies. There is one +12v and one -12V auxiliary power supply. There are two logic ground and two auxiliary ground lines.

The STD bus pinout is organized into four different groups:

- Dual power busses: Pins 1-6 and 53-56
- Data bus: Pins 7-14
- Address bus: Pins 15-30
- Control bus: Pins 31-52

STD bus design goal was to provide a generic system bus for 8-bit microprocessors. Since all processors are not similar in operation, the STD bus provides certain processor specific control lines which are employed by a particular processor. Therefore not all processors use all of the control bus signals. The REFRESH/, MCSYNC/, STATUS1/, and STATUS2/ are processor specific signals. The REFRESH/ signal line is used by the Z-80 processor to refresh dynamic memory, this line is not used by the 8085 processor. Due to this, cards designed to work with one processor

may not work with other processors. If a STD bus card can only operate with a Z-80 based STD bus computer, it is designated as "STD Z-80 Bus" card indicating its compatibility. If a card can work with all the STD bus processors, it is designated as "STD Bus" card.

The edge connector of the STD bus has 28 pins on each side. The 28 odd numbered pins are on the component side of the card and the 28 even numbered pins are on the circuit side of the card. A notch is provided between pins 25 and 27 (26 and 28 on the opposite side) of the card to prevent accidental upside-down insertion of the card into the card cage. The physical dimensions of the card are also specified, 165.1 mm X 114.3 mm X 1.58 mm. All STD bus cards must have buffers at their inputs for all signals except SYSRESET/ and CLOCK/. All ICs connected to the bus must be TTL compatible. The bus drivers must be three-state drivers.

The CPU card communicates with the peripheral or memory cards with the aid of A0-A15, D0-D17, RD/, WR/, IORQ/, MEMRQ/, IOEXP/, and MEMEX/ signals. The control bus signals are described below.

- RD/ - Read request. When RD/ is asserted, the STD bus cards are to recognize the transaction on the bus as a read request from the processor or DMA controller. This line is used during memory and I/O read operations. This signal is generally used to select the direction of the transceivers employed by the cards to buffer the data.
- WR/ - Write request. When WR/ is asserted, the STD bus card should recognize the transaction on the bus as a write request from the processor or DMA controller. This line is used during

memory and I/O write operations. This line may be used, instead of RD/ signal, to select the direction of transceivers employed by the cards to buffer the data.

- IORQ/ - I/O request. The processor or DMA controller asserts this line to perform a I/O read or write. Address bus operation is processor specific during an I/O operation. Some processors duplicate the address of the I/O port on both bytes of the address bus, whereas some processors might just use the lower order byte of the address bus.
- MEMRQ/ - Memory request. The processor or DMA controller asserts this line to perform a memory read or write. The 16 bit address is placed on the address bus prior to asserting this line.
- IOEXP - I/O expansion. This line is rarely used to expand the number of I/O ports available to the processor. It is not used in small systems.
- MEMEX - Memory expansion. This line is used to expand the memory space of a microprocessor by selecting between two banks of memory with same addresses. It is not used in small systems.

The STD bus cards employ the control signals explained above along with the address lines to decode addresses. If the card is addressed, it must place or latch the data on the D0-D7 in synchronization with the CLOCK/. If the card is slow it must generate WAITRQ/, requesting the processor to wait. The cards can interrupt the processor, if needed, by the use of INTRQ/ line on the STD bus.

PD 7210 - Talker, Listener, and Controller

The HP-IB interface card designed for the STD Z-80 bus uses a NEC μPD7210 (PD7210) talker, listener, and controller (TLC). The card is STD Z-80 compatible, meaning it can operate with any Z-80 based STD bus computer. The PD7210 can operate in the clock frequency range of 1-8 MHz. The Z-80 microprocessor in the system being used runs at a clock speed of 3.6864 MHz.

Description

The PD7210 was chosen because it implements all of the functional requirements specified for the Talker, Listener, and Controller (TLC) specified by the HP-IB standard. The TLC manages all the HP-IB functions and the required time delays specified in the standard. The microprocessor is relieved of low level HP-IB bus management chores like managing the assertion and de-assertion of control lines. The TLC has DMA capability. The TLC can present an interrupt to the processor for any of 13 selected conditions. All interrupt conditions are maskable. They are:

- CPT - Command Pass Through. An interrupt is presented if an undefined HP-IB command is received over the interface. An interrupt is presented if a secondary command is received just after the reception of an undefined primary command.
- APT - Address Pass Through. When a secondary address is received, in address mode 3, an interrupt is presented to the processor requesting validation of address. The processor is required to validate the address in command pass through

register.

- DET - Device Trigger. An interrupt is presented to the processor to indicate that the chip is in device trigger active state.
- END - An interrupt is presented to the processor to indicate that EOI line was asserted on the HP-IB. If the end of string register is enabled, an interrupt is presented when the data received over HP-IB matches with the contents of the end of string register.
- DEC - Device Clear. An interrupt is presented to the processor to indicate that the chip is in device clear active state.
- ERR - Error. When a data byte is sent over the HP-IB without a specified listener, an interrupt is presented to bring this condition to the notice of the processor. An interrupt is presented if an attempt is made to write to the byte out register of the chip before the completion of the transmission of the previous byte.
- DO - Data Out. An interrupt is presented to indicate that the chip is ready to transmit the next byte of data.
- DI - Data In. An interrupt is presented to indicate that a new byte has been received over the HP-IB.
- SRQI - Service Request. When the chip receives a service request from a device on the HP-IB, an interrupt is presented bringing this situation to the attention of the processor.
- LOKC - Lockout Change. When a change occurs in the value of the LOK bit, indicating that a change in the local lockout function

has occurred, an interrupt is presented to the processor to bring this condition to notice.

- REMC - Remote Change. An interrupt is presented to indicate that a change in the remote/local function of the chip has occurred.
- CO - Command Output. An interrupt is presented to the processor, to indicate the readiness of the chip to transmit a command over the HP-IB.

Registers of PD7210 The TLC has 16 user accessible registers of which 8 are read-only status registers. The remaining 8 are write-only command registers. Data written into the command registers configure the TLC for any desired state. The state of the TLC can be read from the status registers. figure 2 gives the read/write registers of PD7210. Registers of the PD7210 include:

- Byte Out (0W) - This command register holds the data byte that is to be transmitted over the HP-IB. The register holds the present byte until a new byte is written into it.
- Interrupt Mask 1 (1W) - This command register holds the interrupt mask for 8 of the 13 possible interrupt conditions. The bits in this register enables or disables the corresponding interrupt condition. When a bit is set, the corresponding interrupt condition is enabled.
- Interrupt Mask 2 (2W) - This command register holds interrupt mask for 5 of the 13 possible interrupt conditions. As in the case of interrupt mask 1 register, the bits of interrupt mask 2 enable or disable the corresponding interrupt conditions. Two of

DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0	0R
CPT	APT	DET	END	DEC	ERR	DO	DI	1R
INT	SRQI	LOK	REM	CO	LOKC	REMC	ADSC	2R
S8	PEND	S6	S5	S4	S3	S2	S1	3R
CIC	ATN	SPMS	LPAS	TPAS	LA	TA	MJMN	4R
CPT7	CPT6	CPT5	CPT4	CPT3	CPT2	CPT1	CPT0	5R
X	DT0	DL0	AD5-0	AD4-0	AD3-0	AD2-0	AD1-0	6R
EC1	DT1	DL1	AD5-1	AD4-1	AD3-1	AD2-1	AD1-1	7R

Read registers

BO7	BO6	BO5	BO4	BO3	BO2	BO1	BO0	0W
CPT	APT	DET	END	DEC	ERR	DO	DI	1W
0	SRQI	DMAO	DMAI	CO	LOKC	REMC	ADSC	2W
S8	rsv	S6	S5	S4	S3	S2	S1	3W
ton	lon	TRM1	TRM0	0	0	ADM1	ADM0	4W
CNT2	CNT1	CNT0	COM4	COM3	COM2	COM1	COM0	5W
ARS	DT	DL	AD5	AD4	AD3	AD2	AD1	6W
EC7	EC6	EC5	EC4	EC3	EC2	EC1	EC0	7W

Write registers

FIGURE 2. Read/Write Registers of PD7210 (Courtesy of NEC)

the remaining three bits in the register, which are non-interrupt mask bits, are used for DMA. The most significant bit is unused and is always 0.

- Serial Poll Mode (3W) - This command register holds the status byte to be transmitted when the PD7210 is in serial poll active state. When the controller-in-charge polls the PD7210, the contents of this register are transmitted over the data bus of HP-IB.
- Address Mode (4W) - The address mode of the PD7210 is selected by writing into this command register. The functions of the pins T/R2 and T/R3 are selected by writing to this command register.
- Auxiliary Mode (5W) - By writing to this command register the processor can accomplish a variety of functions. When the CNT2-CNT0 (refer to Figure 2) bits are 000, the processor can use the 5 least significant bits to send auxiliary commands to the chip. When the CNT2-CNT0 bits are 001, the 5 least significant bits are used to set the reference clock frequency of the chip. When the CNT2-CNT0 bits are 011, the 5 least significant bits are written into the parallel poll register (transparent otherwise). When the CNT2-CNT0 bits are 100, the 5 least significant bits are written into the auxiliary register A (transparent otherwise). The bits of auxiliary register A determine the handshake mode of the chip. The bits also enable or disable the end of string register. When the CNT2-CNT0 bits are 101, the 5 least significant bits are written into the auxiliary register B

(transparent otherwise). The bits of auxiliary register B determine the active level of the INT pin of chip along with some other functions. When the CNT2-CNT0 bits are 110, the 5 least significant bits are written into the auxiliary register E (transparent otherwise). The bits of auxiliary register E enable or disable the holdoff of the DAC/ line during the device clear active state and device trigger active state. The user should refer to manufacturers data sheet for further details on these register.

- Address 0/1 (6W) - The PD7210 address on the HP-IB is set by writing into this command register. When the ARS (refer to Figure 2) bit is 1, the 5 least significant bits are written into address register 1 (read-only status register). When the ARS bit is 0, the 5 least significant bits are written into address register 0 (read-only status register). The DT (refer to Figure 2) bit determines if the corresponding address in the register is detected as a talk address or not. The DL (refer to Figure 2) bit determines if the corresponding address in the register is detected as the listen address or not.
- End Of String (7W) - The user can use this register to detect the end of a transmission when the PD7210 is in listener active state. When the byte received over the data lines of the interface match with the contents of this register, the END bit in the interrupt status 1 is set.
- Data In (0R) - The data byte received over the HP-IB in listener

active state is stored in this status register. The contents of this register are unaltered until the reception of a new byte.

- Interrupt Status 1 (1R) - This status register contains the interrupt status bits for 8 of the 13 different interrupt conditions. When a bit is set, and the corresponding interrupt is not masked then an interrupt is presented to the processor by asserting the INT pin of the PD7210. The contents of this status register are cleared by reading it. If an interrupt condition occurs during the read, the request is held until after the read is completed. It is then placed in the register.
- Interrupt Status 2 (2R) - This status register contains 5 of the 13 interrupt status bits. The operation of this register is similar to the operation interrupt status 1 register.
- Serial Poll Status (3R) - The status byte the PD7210 is going to transmit when it is polled can be read from this status register.
- Address Status (4R) - The bits in this status register indicate if the PD7210 is in the active state of listener, or talker, or controller. The user can find out if the PD7210 is in controller standby state by checking the ATN bit of this register. It also indicates whether a major or minor address has been detected.
- Command Pass Through (5R) - This status register holds the HP-IB command sent by a controller if it is undefined, and sets the CPT bit in the interrupt status 1 to indicate the error condition. The processor can read the undefined command from this status register. In address mode 3, this status register holds the

secondary address received over the bus. The processor should read the address and validate it.

- Address 0 (6R) - The user can read the address of the PD7210 on the HP-IB by reading this status register.
- Address 1 (7R) - The user can read the address of the PD7210 on the HP-IB by reading this status register.

HP-IB Buffers

Output drivers of the SN75160 can be three-state or open-collector driver (wired-OR), which is selected by the voltage of the pin PE. When voltage of pin PE is high, three-state operation is chosen while open-collector driver option is chosen when the voltage of pin PE is low. When the voltage of pin TE is high the drivers are in a high impedance state irrespective of the voltage of PE. The direction of signal flow in the SN75161 is controlled by the voltage on pin DC. The TE pin voltage controls if the outputs are in an active low-impedance state or in-active high-impedance state.

THE STD Z-80 INTERFACE CARD

STD Bus Interface of the card

The interface card occupies the address 90H-9FH in the I/O map of the Z-80. This area of the map was chosen because it is not used by the processor for any of its peripheral devices. Address decoding is done with a SN74LS138 (138 in Figure 3) decoder chip. figure 3 gives the schematic diagram of the interface card. The decoder uses address lines A4-A7 along with the IORQ/ line to decode the I/O address. The IORQ/ line is used to assure that memory addresses with lower order byte of 90H-9FH is not decoded. When any I/O address between 90H and 9FH (1001 XXXX) is placed on the lower order byte of address lines of the STD bus, the Y1 output of the decoder is asserted. The Y1 output of the decoder drives the CS/ pin of PD7210 (7210 in Figure 3) and G/ pin of SN74LS245 (245 in Figure 3) enabling both of them. The RD/ signal of the control bus determines the direction of transfer of data through the SN74LS245. The RD/ and WR/ signals of the control bus drive the RD/ and WR/ pins of the PD7210, which has already been enabled by the assertion of CS/ (Y1 of SN74LS245). Registers of the PD7210 are selected by the RS0-RS2 pins, which are driven by the address lines A0-A2. The address line A3 is a don't care during read or write operation. The PD7210 read registers are addressed from 90H-97H. Write registers of PD7210 are addressed from 98H-9FH.

The PD7210 is initially reset by the SYSRESET/ line of the control bus at power on. Since RESET of PD7210 is asserted high, SYSRESET/ is

passed through an inverter before it is connected to the PD7210. The CLOCK/ line of the control bus drives the CLOCK pin of the PD7210. The interrupt and DMA capability of the PD7210 is not utilized. The DMAACK/ pin of the PD7210 is wired to +5V since DMA capability is not used.

HP-IB Interface of the card

The input/output pins of PD7210 which interact with the HP-IB are buffered before they are connected to the connector. Data lines are buffered by the SN75160 (160 in Figure 3), while control lin SN75161 (161 in Figure 3). These two ICs provide complete buffering for the HP-IB and meet all the specifications for an interface bus driver. The Talk Enable (TE/) pins of the buffers are driven by the Transmit/Receive 1 (T/R1) of the PD7210. When TE/ is de-asserted the buffers are disabled, isolating the interface card from the HP-IB. The SN75160 has both open-collector and three-state bus drivers at the outputs. Process Enable (PE) pin of the buffer driven by the T/R3 control output of PD7210. The direction of different control lines of the HP-IB depends on the mode of the bus. When controller is active ATN/, DAV/, REN/, IFC/, and EOI/ pins are outputs. The SRQ/, NDAC/, and NRFD/ pins are inputs when the controller is active. The directions are of these lines are exactly the opposite during the controller in-active state. To meet these requirements T/R2 pin of the PD7210 drives the Direction Control (DC) pin of the SN75161, controlling the directions of the control lines.

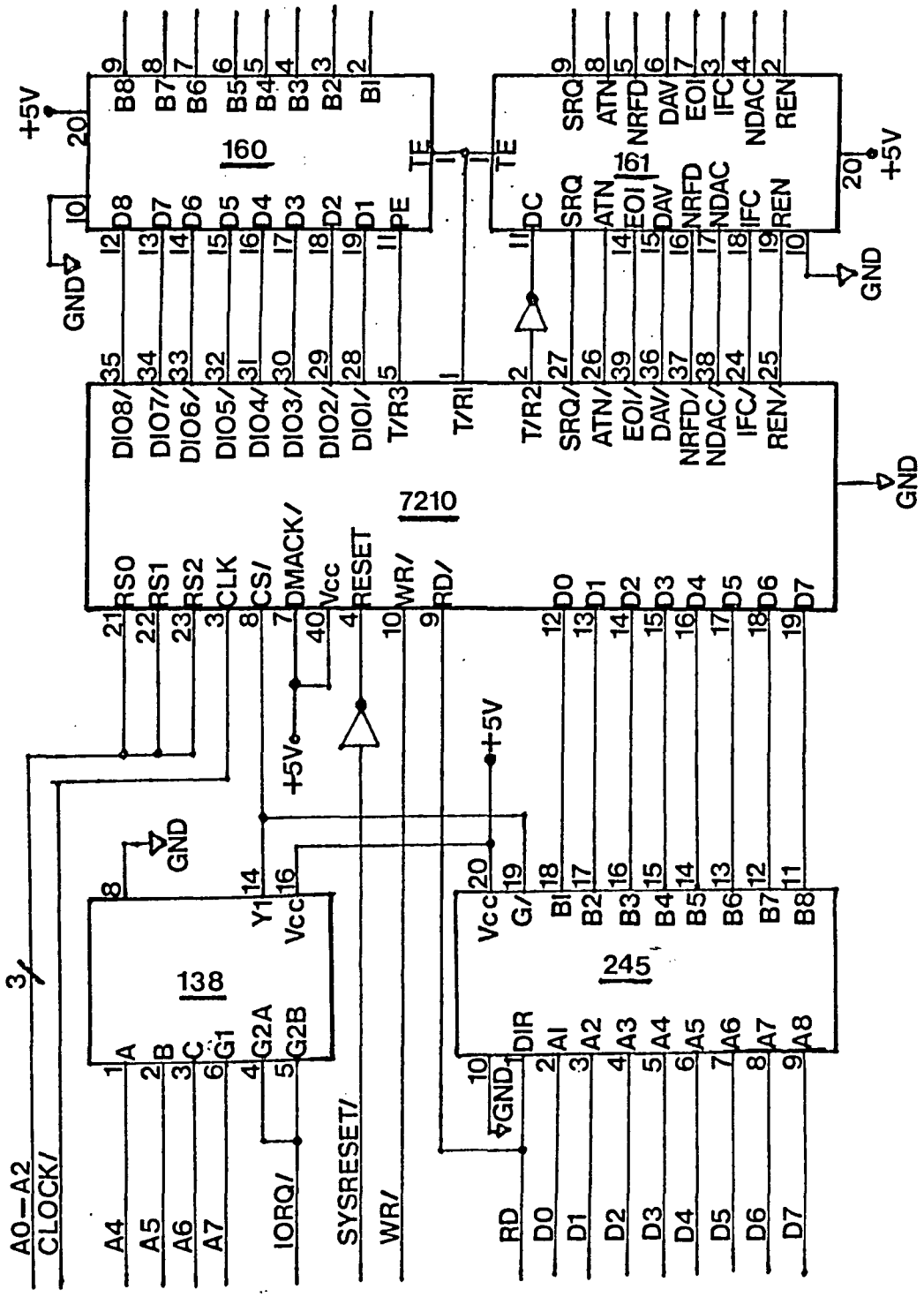

FIGURE 3. Schematic diagram of the card

OPERATION OF THE CARD

Library Subroutines

A library of Z-80 assembly language subroutines is provided for using the card effectively. These subroutines are on the accompanying floppy disk in the file PACKAGE.ASM. A copy of the source code for the subroutines are attached in the Appendix for reference. The application program can invoke these subroutines to perform the desired function.

Description

The subroutines are described in detail in the following discussion. Each subroutine is explained in terms of what it does when called to perform the task.

DO_TEST This subroutine should be called before writing a data byte into the byte out command register of PD7210. The subroutine checks to make sure if PD7210 is ready to transmit the next byte of data over HP-IB. The PD7210 should be in the talker active state when this subroutine is invoked. When the subroutine terminates, the user program can write a data byte into the byte out command register of the PD7210 for transmitting onto the HP-IB. The PD7210 will be in the talker active state on termination of this routine. The subroutine destroys the contents in the A and C registers of the Z-80.

DI_TEST This subroutine should be called before reading data in the status register of the PD7210. The subroutine checks to make sure that the data reception is complete before the data in the status register can be read. The PD7210 should be in listener active state before

invoking this routine. When the subroutine terminates, the user program can read a data byte from the data in register of the PD7210, which was received over the HP-IB. The PD7210 will be in listener active state on termination of this routine. The subroutine destroys the contents in A and C registers of the Z-80.

CO_TEST This subroutine should be called before writing a HP-IB command byte into the byte out command register of the PD7210. The subroutine makes sure that the transmission of the previous command is complete. The PD7210 should be in controller active state before invoking this routine. When the subroutine terminates the PD7210 is ready to transmit a command over the HP-IB. The user can write the required command into the byte out register of PD7210. On termination of the routine, PD7210 will be in the controller active state. This subroutine destroys the contents in A and C register of Z-80.

INITL This subroutine initializes the PD7210 after power-on. It sets the reference clock frequency, determining the time delays required by the interface standard, of the PD7210 by writing into the auxiliary mode command register. The interface card is assigned decimal address 30 on the bus by writing into the address 0/1 command register. Address mode 0 is chosen for the operation of the PD7210 by writing into address mode command register. In address mode 0 of operation of the PD7210 has the same talk and listen address. Normal handshake is chosen by writing into the auxiliary mode command register. The end of string command register is disabled prohibiting the detection of end of string. The auxiliary command, immediate execute power-on, is sent to the PD7210 by writing into

the auxiliary mode command register. This command places all the interface functions of the PD7210 in an idle state. This subroutine should be a part of a startup sequence. It destroys the contents of registers A and B of Z-80.

IFC This subroutine asserts the IFC/ line of the interface for at least 100 μs. The controller should be in the idle state before invoking this routine. On termination of this routine, the controller function of the PD7210 is in an active state. Contents of registers A and C of the Z-80 are destroyed by this routine.

REMOTE This subroutine when invoked asserts the REN/ line of the HP-IB. The PD7210 must be in the controller active state before calling this routine. When the subroutine terminates, the user can send the my listen address HP-IB command over the bus placing addressed devices in the remote mode of operation. The subroutine destroys the A register contents of the Z-80.

D_CLEAR This subroutine when invoked sends the HP-IB multiline command DCL, clearing all devices implementing this function. Before calling this routine, the PD7210 must be in the controller active test. The application subroutine must check to see if the PD7210 is ready for transmitting a command, by invoking the CO_TEST routine, before calling this routine. The contents of A register of the Z-80 are destroyed.

SD_CLEAR This subroutine sends the SDC command over the bus. The PD7210 must be in the controller active state before calling the routine. Before invoking this routine, the PD7210 must be ready to output a command onto the HP-IB. When invoked, the subroutine destroys the contents of A

register of Z-80.

D_TRIG This subroutine triggers the active listeners on the HP-IB by sending the GET multiline command over the bus. The PD7210 must be in the controller active state before invoking this routine. The CO_TEST subroutine must be invoked before calling this subroutine to assure that the PD7210 is ready for transmitting a command. The subroutine destroys the contents of the A register of the Z-80.

D_LOC This subroutine returns all the active listeners in remote mode to local mode of operation. This is accomplished by sending the GTL multiline command over the bus. The PD7210 must be in the controller active state before calling this routine. The CO_TEST subroutine must be called before invoking this subroutine to make sure that PD7210 is ready for transmitting a command. The subroutine destroys the contents of the A register of Z-80.

SERIAL This subroutine prepares the PD7210 for conducting a serial poll and is automatically invoked by the SPEX routine. This subroutine should not be directly used. When this subroutine terminates the SPEX subroutine can perform the serial poll. The contents of the A and C registers of Z-80 are destroyed.

UNSERIAL This subroutine returns the PD7210 to the normal controller active state after conducting a serial poll and is automatically invoked by the SPEX routine. This subroutine should not be directly used. The contents of the A and C registers of Z-80 are destroyed.

SPEX This subroutine can be invoked by the user program to conduct a serial poll of a device whose address is stored in the memory location "POLL". The PD7210 must be in controller active state before invoking this routine. The user must store the talk address of the device whose status byte is required in the memory location 'POLL'. The subroutine returns the status byte of the device in the memory location 'STATUS'. All the devices must be polled to find the interrupting device. On termination, the PD7210 is returned to the controller active state. The subroutine destroys the contents of A and C registers of Z-80.

TALK This subroutine configures the PD7210 in the talker active state. The PD7210 must be in controller active state before invoking this routine. The CO_TEST subroutine must be called before invoking TALK to make sure that the PD7210 is ready to transmit commands. The subroutine places the controller function in idle state. Device dependent data can be transmitted over the HP-IB after termination of this routine. The contents of A and C register of Z-80 are destroyed in this routine.

UNTALK This subroutine configures the PD7210 to controller active state. The PD7210 must be in the talker active state before invoking this routine. The DO_TEST subroutine must be called to assure the completion of the transfer of the last data byte sent over the HP-IB before calling this routine. The contents of A and C registers are destroyed in Z-80.

LISTEN This subroutine configures the PD7210 to listener active state. The PD7210 must be in controller active state before invoking this routine. The CO_TEST must be called to assure the PD7210 is ready for transmitting a command before calling this routine. The contents of A and

C registers of Z-80 are destroyed on termination of this routine.

UNLISTEN This subroutine configures the PD7210 to controller active state. The PD7210 must be in listener active state before invoking this routine. The DI_TEST must be called to assure the reception of the last data byte is complete before calling this routine. The contents of the A and C register of Z-80 are destroyed by this routine.

These subroutines meet the requirements of most users. More subroutines can be added in future to fine tune the card for a specific application.

An example program

An example program is listed in the Appendix to illustrate the usage of the library routines. This program controls the HP-6034A programmable system power supply and HP-3455A digital voltmeter (DVM). The output of the power supply is fed to the DVM. The DVM's address on the HP-IB is 24, and the power supply's address is 25. The power supply presents an interrupt to the active controller as a part of its power-on sequence.

The program initializes the interface card by first calling the INITL routine. IFC subroutine is then invoked to gain control of the bus. When the program returns from IFC, the PD7210 is in controller active state. Serial poll of the bus is conducted by calling the SPEX routine. Both devices are placed in remote mode of operation. Local lockout is sent over the bus disabling the front panel return-to-local switch of both devices. The user is then prompted to enter the control characters for the DVM. After the DVM is programmed, the user is prompted to enter the control string for the power supply. A dummy delay loop is inserted in

the program to provide time for the power supply output to settle. The GET multiline command is then sent forcing the DVM to take a measurement. Next the DVM is assigned as the talker and PD7210 as the listener to receive the measurement from the DVM. The user can visually verify that the front panel display of power supply matches with the output from the DVM printed on the terminal. After this, the user is prompted to exit the program or to remain in the program.

SUMMARY

The HP-IB is an asynchronous digital instrumentation bus. A maximum of 15 instruments can be connected to the bus. The system can have more than one controller, but only one can be active at any time. The bus is either in data mode or in command mode at any time. In data mode, device dependent data is transmitted on the bus by the active Talker. In command mode, interface commands are sent over the bus by the active controller. There can only be one active talker at any time and a maximum of 14 active listeners on the bus at any time. Every device on the bus has a unique address. The talk and listen addresses for an instrument are generally the same. Address 31 (1 1111) is not a valid address for any device, because it is used by the standard to untalk and unlisten talkers and listeners respectively.

STD bus is a synchronous 8-bit microprocessor bus. It can support a variety of 8-bit microprocessors. All the lines in the STD bus are defined unlike other standards where some of the lines are defined. The STD bus provides certain processor specific control lines, thereby leading to processor specific cards. All STD bus cards must have TTL compatible ICs. The bus can be controlled by a DMA card if needed.

The HP-IB interface card described in the thesis uses PD7210 as the talker, listener, and controller. The PD7210 can operate in the clock frequency range of 1-8 MHz. The STD bus card is Z-80 processor specific. The Z-80 used in the system runs at a clock frequency of 3.6864 MHz. The interface card occupies the 90H - 9FH portion of the I/O map of Z-80. The I/O ports 90H-97H are read-only, whereas the ports 98H-9FH are write-only.

The SN75160 and SN75161 buffers are used between the PD7210 and the HP-IB. The card can be programmed by writing to 8 write-only registers. The state of the interface card can be read by reading from the 8 read-only registers. A small set of library subroutines are provided to use the card effectively. More subroutines can be added as needed. An example program is provided to illustrate the use of these library routines.

BIBLIOGRAPHY

1. Hewlett-Packard. "Tutorial description of the Hewlett-Packard Interface Bus". Hewlett-Packard Company, Palo Alto, California, 1980.

2. IEEE Standard 488-1978. "Digital interface for programmable instrumentation". The IEEE Inc., 345 East 47th St., New York, Nov. 1978.

3. D. E. Knoblock, D. C. Loughry and Vissers C. A. "Insight into interfacing". IEEE Spectrum, 12, No. 5 (May 1975): 50-57.

4. J. Summers. "IEC Interface Bus (IEEE-488) Simplified". Electronics Engineering, 38, No. 12 (Dec 1979):45-55.

5. C. A. Titus, J. A. Titus and D. G. Larsen. "STD Bus Interfacing". Blacksburg Continuing Education Series, Blacksburg, Virginia, 1982.

6. William Twaddell. "Low-cost, high capability LSI chips ease interface bus implementation". EDN, 24, No. 23 (Nov 5, 1979):36-40.

7. Richard Young. "Implementing an IEEE-488 Bus Controller with Microprocessor Software". IEEE Transactions on Industrial Electronics and Control Instrumentation, IECI-27, No. 1 (Feb 1980):10-15.

APPENDIX - LIBRARY SUBROUTINES AND AN EXAMPLE PROGRAM

```
        ;PACKAGE OF ROUTINES TO CONTROL AND USE THE 7210
        ;
        ;
        .ORG    1000H
        ;
        ;
        ;7210 REGISTERS
        ;
        ;READ-ONLY STATUS REGISTERS
        ;
DIN:    .EQU    90H     ;DATA COMING IN FROM GPIB
INTS1:  .EQU    91H     ;INTERRUPT STATUS 1
INTS2:  .EQU    92H     ;INTERRUPT STATUS 2
SPS:    .EQU    93H     ;SERIAL POLL STATUS
ADRS:   .EQU    94H     ;ADDRESS STATUS
CPT:    .EQU    95H     ;COMMAND PASS THROUGH
ADR0:   .EQU    96H     ;ADDRESS 0
ADR1:   .EQU    97H     ;ADDRESS 1
        ;
        ;
        ;WRITE-ONLY COMMAND REGISTERS
        ;
DOUT:   .EQU    98H     ;DATA GOING OUT OF INTERFACE TO GPIB
INTM1:  .EQU    99H     ;INTERRUPT MASK 1
INTM2:  .EQU    9AH     ;INTERRUPT MASK 2
SPM:    .EQU    9BH     ;SERIAL POLL MODE
ADRM:   .EQU    9CH     ;ADDRESS MODE
AUXM:   .EQU    9DH     ;AUXILIARY MODE
ADR01:  .EQU    9EH     ;ADDRESS 0/1
EOS:    .EQU    9FH     ;END OF STRING
        ;
        ;
        ;7210 AUXILIARY COMMANDS
        ;
XIEP:   .EQU    00H     ;IMMEDIATE EXECUTE PON
XCRS:   .EQU    02H     ;CHIP RESET
XFHS:   .EQU    03H     ;FINISH HANDSHAKE IN HOLDOFF MODE
XTRIG:  .EQU    04H     ;TRIGGER
XSRTL:  .EQU    0DH     ;SET RETURN TO LOCAL
XRRTL:  .EQU    05H     ;RESET RETURN TO LOCAL
XSEOI:  .EQU    06H     ;SEND EOI WITH NEXT BYTE
XVLD:   .EQU    0FH     ;VALIDATE THE UNDEFINED COMMAND
XNVLD:  .EQU    07H     ;INVALIDATE THE UNDEFINED COMMAND
XSPPF:  .EQU    09H     ;SET PARALLEL POLL FLAG
XRPPF:  .EQU    01H     ;RESET PARALLEL POLL FLAG
XGTS:   .EQU    10H     ;GO TO STANDBY
XTCA:   .EQU    11H     ;TAKE CONTROL ASYNCHRONOUSLY
XTCS:   .EQU    12H     ;TAKE CONTROL SYNCHRONOUSLY
XTCSE:  .EQU    1AH     ;TAKE CONTROL SYNCHRONOUSLY AT END
XLTN:   .EQU    13H     ;LISTEN
XLTNC:  .EQU    1BH     ;LISTEN IN CONTINUOUS MODE
XLUN:   .EQU    1CH     ;LOCAL UNLISTEN
```

```
XEPP:    .EQU    1DH      ;EXECUTE PARALLEL POLL
XSIFC:   .EQU    1EH      ;SET INTERFACE CLEAR
XRIFC:   .EQU    16H      ;RESET INTERFACE CLEAR
XSREN:   .EQU    1FH      ;SET REMOTE ENABLE
XRREN:   .EQU    17H      ;RESET REMOTE ENABLE
XDSC:    .EQU    14H      ;DISABLE SYSTEM CONTROL
         ;
         ;
         ;GPIB REMOTE MESSAGES
         ;
DCL:     .EQU    14H      ;DEVICE CLEAR
LLO:     .EQU    11H      ;LOCAL LOCKOUT
PPU:     .EQU    15H      ;PARALLEL POLL UNCONFIGURE
SPD:     .EQU    19H      ;SERIAL POLL DISABLE
SPE:     .EQU    18H      ;SERIAL POLL ENABLE
GET:     .EQU    08H      ;GROUPT EXECUTE TRIGGER
GTL:     .EQU    01H      ;GO TO LOCAL
PPC:     .EQU    05H      ;PARALLEL POLL CONFIGURE
SDC:     .EQU    04H      ;SELECTED DEVICE CLEAR
TCT:     .EQU    09H      ;TAKE CONTROL
UNL:     .EQU    3FH      ;UNLISTEN
UNT:     .EQU    5FH      ;UNTALK
PPD:     .EQU    70H      ;PARALLEL POLL DISABLE
         ;
         ;
         ;THESE ARE THE MONITOR ROUTINES USED TO INPUT
         ;AND OUTPUT CHARACTERS FROM THE TERMINAL
         ;
GETCHAR: .EQU    0088H
STRING:  .EQU    009CH
PUTCHAR: .EQU    008CH
MINE:    .EQU    0084H
         ;
         ;
         JP      MAIN
         ;
         ;
         ;BUFFERS
         ;
DVMCOM:  .DS     20
PSCOM:   .DS     20
STATUS:  .DS     1
POLL:    .DS     1
         ;
         ;
         ;PROMPTS
         ;
PRO1:    .DB     'ENTER COMMAND FOR DVM PROGRAMMING (COMMAND IS ',00H
PRO2:    .DB     'TERMINATED BY &)', 0DH, 0AH, 00H
PRO3:    .DB     'COMMAND : ', 00H
PRO4:    .DB     'ENTER COMMAND FOR POWER SUPPLY SETTING (COMMAND ',00H
PRO5:    .DB     'IS TERMINATED BY &)', 0DH, 0AH, 00H
```

```
PRO6:   .DB     'COMMAND : ',00H
PRO7:   .DB     'ONE MORE SETTING? (Y/N) : ',00H
PRO8:   .DB     'OUTPUT VOLTAGE IS : ', 00H
        ;
        ;
        ;THIS ROUTINE PRINTS THE ADDRESS STATUS REGISTER
        ;
PADRS:
        IN      A,(ADRS)
        CALL    0098H
        RET
        ;
        ;
        ;THIS ROUTINE PRINTS OUT THE CONTENTS OF DIN, INTS1,
        ;INTS2, SPM, AND ADRS REGISTERS OF THE 7210
        ;
PRINT:
        IN      A,(DIN)         ;GET DIN REGISTER
        CALL    0098H           ;PRINT
        IN      A,(INTS1)       ;GET INTS1 REGISTER
        CALL    0098H           ;PRINT
        IN      A,(INTS2)       ;GET INTS2 REGISTER
        CALL    0098H           ;PRINT
        IN      A,(ADRS)        ;GET ADRS REGISTER
        CALL    0098H           ;PRINT
        RET
        ;
        ;
        ;DELAY INTRODUCES A DELAY BETWEEN COMMANDS
        ;SIDE EFFECTS           : CONTENTS OF REGISTER A DESTROYED
        ;                       : CONTENTS OF REGISTER C DESTROYED
        ;
DELAY:
        LD      A,03H
        LD      C,A
LOOP0:  CALL    LONGER
        DEC     C
        JP      NZ,LOOP0
        RET
        ;
        ;
        ;INTRODUCES A LONGER DELAY
        ;SIDE EFFECTS           : CONTENTS OF A DESTROYED
        ;
LONGER:
        LD      A,FFH
LOP0:   DEC     A
        JP      NZ,LOP0
        RET
        ;
        ;
        ;TESTS IF THE NEXT BYTE CAN BE SENT OVER GPIB
```

```
        ;TLC STATE BEFORE CALLING       : TALKER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : TALKER ACTIVE STATE
        ;SIDE EFFECTS                   : INTS1 REG OF 7210 CLEARED
        ;                               : REGISTER A CONTENTS DESTROYED
DO_TEST:
        IN      A,(INTS1)       ;GET INTERRUPT STATUS 1
        BIT     1,A             ;CHECK DO BIT
        JP      NZ,DO_TEST      ;WAIT
        CALL    DELAY           ;WAIT
        RET
        ;
        ;
        ;CHECKS IF A NEW BYTE IS RECEIVED OVER THE GPIB
        ;TLC STATE BEFORE CALLING       : TALKER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : TALKER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;                               : CONTENTS OF C DESTROYED
        ;                               : INTS1 REG CLEARED
        ;
DI_TEST:
        IN      A,(INTS1)       ;GET INTERRUPT STATUS 1
        BIT     0,A             ;TEST DI BIT
        JP      NZ,DI_TEST      ;WAIT
        CALL    DELAY           ;WAIT
        RET
        ;
        ;
        ;CHECKS IF THE PREVIOUS COMMAND TRANSMISSION IS
        ;COMPLETE SO THAT THE NEXT COMMAND CAN BE SENT
        ;TLC STATE BEFORE CALLING       : CONTROLLER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : CONTROLLER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF INTS1 CLEARED
        ;                               : CONTENTS OF A DESTROYED
        ;                               ; CONTENTS OF C DESTROYED
        ;
CO_TEST:
        IN      A,(INTS2)       ;GET INTERRUPT STATUS 2
        BIT     3,A             ;CHECK THE CO BIT
        JP      NZ,CO_TEST      ;WAIT
        CALL    DELAY
        RET
        ;
        ;
        ;INITIALIZATION ROUTINE
        ;TLC STATE BEFORE CALLING       : POWER ON STATE
        ;TLC STATE AFTER CALLING        : ALL TLC FUNCTIONS IN IDLE
        ;                                 STATE, NORMAL HANDSHAKE MODE,
        ;                                 MODE 1 OF TALKER AND LISTENER,
        ;                                 ALL INTERRUPTS MASKED
        ;
INITL:
        LD      A,XCRS          ;CHIP RESET
```

```
        OUT     (AUXM),A
        LD      A,00H           ;DISABLE ALL INTERRUPTS
        OUT     (INTM1),A
        OUT     (INTM2),A
        LD      A,31H           ;ADDRESS MODE 1
        OUT     (ADRM),A
        LD      A,98H           ;NORMAL HANDSHAKE MODE AND REG AUXA
        OUT     (AUXM),A
        LD      A,A8H           ;AUXILIARY REGISTER B
        OUT     (AUXM),A
        LD      A,C0H           ;AUXILIARY REGISTER E
        OUT     (AUXM),A
        LD      A,24H           ;DELAY BETWEEN STATE TRANSITIONS
        OUT     (AUXM),A
        LD      A,1EH           ;MAJOR ADDRESS IS 30
        OUT     (ADR01),A
        LD      A,E0H           ;NO MINOR ADDRESS
        OUT     (ADR01),A
        LD      A,XIEP          ;ALL TLC FUNCTIONS IN IDLE STATE
        OUT     (AUXM),A
        RET
        ;
        ;
        ;THIS ROUTINE ASSERTS THE IFC LINE OF THE BUS FOR
        ;ATLEAST 100 MICROSECONDS AND PLACES THE TLC IN
        ;CONTROLLER IN CHARGE STATE
        ;TLC STATE BEFORE CALLING       : CONTROLLER IDLE STATE
        ;TLC STATE AFTER CALLING        : CONTROLLER ACTIVE STATE
        ;SIDE EFFECTS                   : REGISTER A CONTENTS DESTROYED
        ;                               : REIGSTER C CONTENTS DESTROYED
        ;
IFC:
        LD      A,XSIFC         ;SEND IFC
        OUT     (AUXM),A
        CALL    DELAY           ;WAIT FOR ATLEAST 100 MICRO SECONDS
        LD      A,XRIFC         ;REMOVE IFC
        OUT     (AUXM),A
        RET
        ;
        ;
        ;THIS ROUTINE ASSERTS THE REN LINE, SO THAT DEVICES
        ;CAN BE PLACED IN REMOTE MODE WHEN ADDRESSED TO LISTEN
        ;THE TLC IS IN COMMAND MODE
        ;TLC STATE BEFORE CALLING       : CONTROLLER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : CONTROLLER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;
REMOTE:
        LD      A,XSREN         ;ASSERT REMOTE
        OUT     (AUXM),A
        RET
        ;
```

```
        ;
        ;THIS ROUTINE SENDS THE REMOTE MESSAGE DCL, THUS
        ;CLEARING ALL DEVICES TO DEVICE DEPENDENT IDLE STATE
        ;THIS ROUTINE CLEARS ALL DEVICES.
        ;TLC STATE BEFORE CALLING       : CONTROLLER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : CONTROLLER ACITVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;
D_CLEAR:
        LD      A,DCL           ;SEND DEVICE CLEAR
        OUT     (DOUT),A
        RET
        ;
        ;
        ;THIS ROUTINE CLEARS DEVICES WHICH ARE CURRENTLY
        ;LISTENERS ON THE BUS.  THE TLC SHOULD BE IN
        ;COMMAND MODE
        ;TLC STATE BEFORE CALLING       : CONTROLLER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : CONTROLLER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;
SD_CLEAR:
        LD      A,SDC           ;SEND SELECTED DEVICE CLEAR
        OUT     (DOUT),A
        RET
        ;
        ;
        ;THIS ROUTINE TRIGGERS DEVICES WHICH ARE ADDRESSED
        ;TO LISTEN.  TLC SHOULD BE IN COMMAND MODE
        ;TLC STATE BEFORE CALLING       : CONTROLLER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : CONTROLLER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;
D_TRIG:
        LD      A,GET           ;SEND GROUP EXECUTE TRIGGER
        OUT     (DOUT),A
        RET
        ;
        ;
        ;THIS ROUTINE RETURNS CURRENTLY LISTENING DEVICES
        ;TO LOCAL CONTROL.  TLC SHOULD BE COMMAND MODE
        ;TLC STATE BEFORE CALLING       : CONTROLLER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : CONTROLLER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;
D_LOC:
        LD      A,GTL           ;SEND GO TO LOCAL
        OUT     (DOUT),A
        RET
        ;
        ;
        ;THIS ROUTINE CONFIGURES THE TLC FOR SERIAL POLL
```

```
        ;
SERIAL:
        CALL    CO_TEST         ;WAIT
        LD      A,UNL           ;UNLISTEN ALL LISTENERS
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        LD      A,SPE           ;SEND SERIAL POLL ENABLE
        OUT     (DOUT),A
        RET
        ;
        ;
        ;THIS ROUTINE RECONFIGURES THE TLC FOR CONTROLLER MODE
        ;WITH NORMAL HANDSHAKE AFTER A SERIAL POLL
        ;
UNSERIAL:
        LD      A,XTCS          ;TAKE CONTROL SYNCHRONOUSLY
        OUT     (AUXM),A
        CALL    DELAY           ;WAIT
        LD      A,XLUN          ;SEND LOCAL UNLISTEN
        OUT     (AUXM),A
        CALL    DELAY           ;WAIT
        CALL    CO_TEST         ;WAIT
        LD      A,SPD           ;SEND SERIAL POLL DISABLE
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        LD      A,UNT           ;UNTALK
        OUT     (DOUT),A
        RET
        ;
        ;
        ;THIS ROUTINE EXECUTES SERIAL POLL
        ;
SPEX:
        CALL    SERIAL          ;CONFIGURE TLC FOR SERIAL POLL
        CALL    CO_TEST         ;WAIT
        LD      A,(POLL)        ;ADDRESS OF POWER SUPPLY
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        LD      A,XLTN          ;PUT TLC IN LISTENER MODE
        OUT     (AUXM),A
        CALL    DELAY
        LD      A,XGTS          ;PUT CONTROLLER IN STANDBY
        OUT     (AUXM),A
        CALL    DELAY
        CALL    DI_TEST         ;WAIT FOR STATUS BYTE
        CALL    UNSERIAL        ;CONFIGURE FOR CONTROLLER IN CHARGE
        IN      A,(DIN)         ;GET THE STATUS BYTE
        LD      (STATUS),A      ;SAVE IT
        RET
        ;
        ;
        ;THIS ROUTINE CONFIGURES THE TLC IN TALK MODE
```

```
        ;TLC STATE BEFORE CALLING       : CONTROLLER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : CONTROLLER STANDBY STATE
        ;                               : TALKER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;                               : CONTENTS OF C DESTROYED
        ;
TALK:
        LD      A,5EH           ;TALK ADDRESS IS 30
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        LD      A,XGTS          ;PUT CONTROLLER IN STANDBY
        OUT     (AUXM),A
        CALL    DELAY           ;WAIT
        RET
        ;
        ;
        ;THIS ROUTINE CONFIGURES THE TLC IN CONTROLLER IN CHARGE
        ;MODE WHEN IT IS IN TALKER MODE
        ;TLC STATE BEFORE CALLING       : CONTROLLER STANDBY STATE
        ;                               : TALKER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : TALKER IDLE STATE
        ;                               : CONTROLLER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;                               : CONTENTS OF C DESTROYED
        ;
UNTALK:
        LD      A,31H           ;ADDRESS MODE 1
        OUT     (ADRM),A
        CALL    DELAY           ;WAIT
        LD      A,XTCA          ;TAKE CONTROL ASYNCHRONOUSLY
        OUT     (AUXM),A
        CALL    DELAY           ;WAIT
        CALL    CO_TEST         ;WAIT
        LD      A,UNT           ;UNTALK ALL TALKERS
        OUT     (DOUT),A
        RET
        ;
        ;
        ;THIS ROUTINE CONFIGURES THE TLC IN LISTENER MODE WHEN
        ;IT IS IN CONTROLLER IN CHARGE MOD
        ;TLC STATE BEFORE CALLING       : CONTROLLER ACTIVE STATE
        ;TLC STATE AFTER CALLING        : CONTROLLER STANDBY STATE
        ;                               : LISTENER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;                               : CONTENTS OF C DESTROYED
        ;
LISTEN:
        LD      A,XLTN          ;PUT TLC IN LISTENER MODE
        OUT     (AUXM),A
        CALL    DELAY           ;WAIT
        LD      A,XGTS          ;GO TO STANDBY
        OUT     (AUXM),A
```

```
        CALL    DELAY           ;WAIT
        RET
        ;
        ;
        ;THIS ROUTINE RETURNS THE TLC TO CONTROLLER IN CHARGE
        ;STATE WHILE IT WAS LISTENING
        ;TLC STATE BEFORE CALLING       : LISTENER ACTIVE STATE
        ;                               : CONTROLLER STANDBY STATE
        ;TLC STATE AFTER CALLING        : LISTENER IDLE STATE
        ;                               : CONTROLLER ACTIVE STATE
        ;SIDE EFFECTS                   : CONTENTS OF A DESTROYED
        ;                               : CONTENTS OF C DESTROYED
        ;
UNLISTEN:
        LD      A,31H           ;ADDRESS MODE 1
        OUT     (ADRM),A
        CALL    DELAY           ;WAIT
        LD      A,XTCA          ;TAKE CONTROL ASYNCHRONOUSLY
        OUT     (AUXM),A
        CALL    DELAY           ;WAIT
        CALL    CO_TEST         ;WAIT
        LD      A,UNL           ;UNLISTEN ALL LISTENERS
        OUT     (DOUT),A
        RET
        ;
        ;
        ;THIS ROUTINE READS THE COMMANDS FROM TERMINAL
        ;
READ:
        CALL    GETCHAR         ;GET NEXT INPUT
        LD      (IX+0),A        ;STORE THE CHARACTER
        INC     IX
        CP      '&'             ;END OF COMMAND?
        JP      NZ,READ
        RET
        ;
        ;
        ;THIS ROUTINE WRITES THE COMMANDS OUT TO GPIB
        ;
WRITE:
        CALL    DO_TEST         ;WAIT
        LD      A,(IX+1)        ;GET NEXT COMMAND FROM BUFFER
        CP      '&'             ;IS IT END OF COMMAND
        JP      Z,DONE
        LD      A,(IX+0)
        OUT     (DOUT),A
        LD      C,A
        INC     IX
        JP      WRITE
DONE:   LD      A,XSEOI         ;SEND EOI WITH NEXT BYTE
        OUT     (AUXM),A
        CALL    DELAY           ;WAIT
```

```
        LD      A,(IX+0)
        OUT     (DOUT),A
        RET
        ;
        ;
        ;MAIN BODY OF THE PROGRAM
        ;THIS IS AN EXAMPLE PROGRAM SHOWING HOW TO USE THE SUBROUTINES
        ;THIS PROGRAM PROGRAMS THE DVM TO TAKE MEASUREMENT FROM THE
        ;POWER SUPPLY AND RETURN THE MEASUREMENT OVER THE INTERFACE
MAIN:
        CALL    INITL           ;INITIALIZE 7210
        CALL    IFC             ;PUT CONTROLLER IN CHARGE
        CALL    DELAY           ;WAIT
        LD      A,45H           ;ADDRESS OF POWER SUPPLY
        LD      (POLL),A
        CALL    SPEX            ;DO SERIAL POLL
        CALL    DELAY           ;DELAY
        CALL    REMOTE          ;PLACE ALL DEVICES IN REMOTE
        CALL    CO_TEST         ;WAIT
        LD      A,25H           ;POWER SUPPLY IS LISTENER
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        LD      A,24H           ;DVM LISTENER
        OUT     (DOUT),A
        CALL    CO_TEST
        LD      A,LLO           ;LOCAL LOCKOUT
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        LD      A,UNL           ;UNLISTEN ALL LISTENERS
        OUT     (DOUT),A
AGAIN:  CALL    CO_TEST         ;WAIT
        CALL    PUTCHAR         ;SEND CARRIAGE RETURN
        LD      A,24H           ;DVM IS LISTENER
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        CALL    TALK            ;CONFIGURE TLC FOR TALKING
        LD      HL,PRO1         ;PROMPT USER
        CALL    STRING
        LD      HL,PRO2
        CALL    STRING
        LD      HL,PRO3
        CALL    STRING
        LD      IX,DVMCOM       ;LOAD POINTER
        CALL    READ            ;READ INPUT FROM TERMINAL
        LD      IX,DVMCOM       ;LOAD POINTER
        CALL    WRITE           ;WRITE THE COMMANDS ON GPIB
        CALL    DO_TEST         ;WAIT
        CALL    UNTALK          ;RECONFIGURE TLC FOR CONTROLLER
        CALL    CO_TEST         ;WAIT
        LD      A,UNL           ;UNLISTEN ALL DEVICES
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
```

```
        LD      A,25H           ;POWER SUPPLY IS LISTENER
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        CALL    PUTCHAR         ;SEND CARRIAGE RETURN
        CALL    TALK            ;CONFIGURE TLC FOR TALK
        LD      HL,PRO4         ;PROMPT USER
        CALL    STRING
        LD      HL,PRO5
        CALL    STRING
        LD      HL,PRO6
        CALL    STRING
        LD      IX,PSCOM        ;LOAD POINTERS
        CALL    READ            ;GET NEXT INPUT
        LD      IX,PSCOM        ;LOAD POINTERS
        CALL    WRITE           ;WRITE COMMANDS TO GPIB
        CALL    DO_TEST         ;WAIT
        CALL    UNTALK          ;RECONFIGURE FOR CONTROLLER IN CHARGE
        CALL    CO_TEST         ;WAIT
        LD      A,UNL           ;UNLISTEN ALL LISTENERS
        OUT     (DOUT),A
        CALL    PUTCHAR         ;SEND CARRIAGE RETURN
        CALL    CO_TEST         ;WAIT
        LD      A,24H           ;DVM IS LISTENER
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        LD      A,FFH           ;GIVE TIME FOR OUTPUT OF POWER SUPPLY
        LD      C,A             ;TO SETTLE DOWN BEFORE TAKING MEASUREMENT
DUMM:   CALL    LONGER
        DEC     C
        JP      NZ,DUMM
        CALL    D_TRIG          ;TAKE MEASUREMENT
        CALL    CO_TEST         ;WAIT
        LD      A,UNL           ;UNLISTEN DVM
        OUT     (DOUT),A
        CALL    CO_TEST         ;WAIT
        LD      A,44H           ;DVM IS TALKER
        OUT     (DOUT),A        ;
        CALL    CO_TEST         ;WAIT
        LD      HL,PRO8         ;PRINT OUTPUT PROMPT
        CALL    STRING
        CALL    LISTEN          ;CONFIGURE TLC IN LISTENER MODE
LOOP1:  IN      A,(INTS1)       ;CHECK FOR LAST BYTE RECEIVED
        CP      11H             ;IF EQUAL LAST BYTE RECEIVED
        JP      Z,LAST
        CP      01H             ;IS A NEW BYTE RECEIVED?
        JP      NZ,LOOP1        ;NO, SO GO BACK AND WAIT FOR BYTE
        IN      A,(DIN)         ;READ BYTE
        LD      C,A
        CALL    MINE            ;PRINT IT OUT
        JP      LOOP1           ;GO BACK FOR NEXT BYTE
LAST:   IN      A,(DIN)         ;GET BYTE
        LD      C,A
```

```
CALL    MINE            ;PRINT IT OUT
CALL    DI_TEST
CALL    UNLISTEN        ;PUT TLC BACK IN CONTROLLER MODE
CALL    CO_TEST         ;WAIT
LD      A,UNT           ;UNTALK SUPPLY
OUT     (DOUT),A
CALL    CO_TEST         ;WAIT
LD      HL,PRO7         ;PROMPT USER FOR NEXT SETTING
CALL    STRING          ;PROMPT USER FOR NEXT SETTING
CALL    GETCHAR
CP      'Y'             ;DONE?
JP      Z,AGAIN
RST     8
.END
```