GPIB-SBX

User Manual

January 1996 Edition

Part Number 320015-01

National Instruments Corporate Headquarters
6504 Bridge Point Parkway
Austin, TX 78730-5039
(512) 794-0100
Technical support fax: (800) 328-2203
(512) 794-5678

Branch Offices:
Australia 03 9 879 9422, Austria 0662 45 79 90 0, Belgium 02 757 00 20, Canada (Ontario) 519 622 9310,
Canada (Québec) 514 694 8521, Denmark 45 76 26 00, Finland 90 527 2321, France 1 48 14 24 24,
Germany 089 741 31 30, Hong Kong 2645 3186, Italy 02 48301892, Japan 03 5472 2970,
Korea 02 596 7456, Mexico 5 202 2544, Netherlands 03480 33466, Norway 32 84 84 00,
Singapore 2265886, Spain 91 640 0085, Sweden 08 730 49 70, Switzerland 056 20 51 51,
Taiwan 02 377 1200, U.K. 01635 523545

Limited Warranty

The GPIB-SBX is warranted against defects in materials and workmanship for a period of two years from the date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace equipment that proves to be defective during the warranty period. This warranty includes parts and labor.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. National Instruments will pay the shipping costs of returning to the owner parts which are covered by warranty.

National Instruments believes that the information in this manual is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, National Instruments reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult National Instruments if errors are suspected. In no event shall National Instruments be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, NATIONAL INSTRUMENTS MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF NATIONAL INSTRUMENTS SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. NATIONAL INSTRUMENTS WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF. This limitation of the liability of National Instruments will apply regardless of the form of action, whether in contract or tort, including negligence. Any action against National Instrument must be brought within one year after the cause of action accrues. National Instruments shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failures caused by owner's failure to follow the National Instruments installation, operation, or maintenance instructions; owner's modification of the product; owner's abuse, misuse, or negligent acts; and power failure or surges, fire, flood, accident, actions of third parties, or other events outside reasonable control.

Copyright

Trademarks

Warning Regarding Medical and Clinical Use of National Instruments Products

National Instruments products are not designed with components and testing intended to ensure a level of reliability suitable for use in treatment and diagnosis of humans. Applications of National Instruments products involving medical or clinical treatment can create a potential for accidental injury caused by product failure, or by errors on the part of the user or application designer. Any use or application of National Instruments products for or involving medical or clinical treatment must be performed by properly trained and qualified medical personnel, and all traditional medical safeguards, equipment, and procedures that are appropriate in the particular situation to prevent serious injury or death should always continue to be used when National Instruments products are being used. National Instruments products are NOT intended to be a substitute for any form of established process, procedure, or equipment used to monitor or safeguard human health and safety in medical or clinical treatment.

Preface

This manual contains information you need to install, configure, and use the GPIB-SBX interface board. Included in this manual are descriptions of the interface registers, programming considerations, diagnostic tests, and sample programs.

Organization of the *GPIB-SBX User Manual*

The *GPIB-SBX User Manual* is organized as follows:

- Chapter 1, *Introduction*, contains a brief description of the GPIB-SBX kit, a list of equipment supplied, and a list of optional equipment.
- Chapter 2, *General Description*, contains the physical and electrical specifications for the GPIB-SBX and describes the characteristics of key interface board components.
- Chapter 3, *Configuration and Installation*, describes the steps needed to configure and install the GPIB-SBX hardware.
- Chapter 4, *Register Descriptions*, contains detailed descriptions of the interface registers as well as summary tables for easy reference.
- Chapter 5, *Programming Considerations*, explains important considerations for persons programming the GPIB-SBX.
- Chapter 6, *Theory of Operation*, contains a detailed technical discussion of the major elements of the GPIB-SBX.
- Chapter 7, *Diagnostic and Troubleshooting Test Procedures*, contains test procedures for determining if the GPIB-SBX is installed properly and operating correctly.
- Appendix A, *Parts List and Schematic Diagram*, includes a parts list and detailed schematic diagram.
- Appendix B, *Sample Programs*, provides sample programs in 8080 and 8088 Assembly language code for implementing the most commonly used GPIB functions. Line-by-line comments provide an explanation of each function.
- Appendix C, *Multiline Interface Messages*, contains a listing of the multiline GPIB interface messages.
- Appendix D, *Mnemonics Key*, contains an alphabetical listing of all mnemonics used in this manual and indicates whether the mnemonic represents a bit, register, function, remote message, local message, or a state.
- The *Index* contains an alphabetical list of key terms and topics used in this manual, including the page where each one can be found.

Terminology

The term *set* is used throughout this manual to mean writing a value of 1 to a memory element. The term *clear* is used to mean writing a value of 0 to a memory element. The terms *set to zero*, *set false*, *preset*, and *reset* are avoided in preference to the use of *set* and *clear*. Bit signatures are written in upper case letters. An asterisk (*) after a bit signature indicates the signal is active low. An asterisk is equivalent to an overbar. For example, if bit ATN*=0, the GPIB ATN signal line is asserted. After the mnemonic of a register name is introduced and the name written out, the mnemonic is used thereafter.

Where it is necessary to specify a particular bit of a register the bit position appears as a decimal number in square brackets after the mnemonic (for example, ISR1[2] indicates the ERR bit of Interrupt Status Register 1).

Uppercase mnemonics are used for control, status, data registers, register contents, and interface functions, as well as GPIB remote messages, commands, and logic states as defined in the IEEE-488 standard.

The term *addressed* means the interface has been configured to perform a function from the GPIB side, while the term *programmed* means that it has been configured from the SBX bus side. This distinction is important to make because many functions, such as making the interface a Talker or Listener, can be activated from either side.

In logical expressions a minus sign (−) is used to indicate logical negation, an ampersand (&) represents AND, and a plus sign (+) represents OR. In some cases, a plus sign is used to indicate addition, such as I/O addresses for interface registers in Chapter 4.

All numbers are decimal unless specified otherwise. Register offsets are given in hexadecimal.

Abbreviations

The following abbreviations are used in the text of this manual.

≥	greater than or equal to
<	less than
A	ampere
hex	hexadecimal
in.	inch
kbytes/sec	1,000 bytes per second
K	1,024 bytes of memory
m	meter
µA	microampere
µsec	microsecond
mA	milliampere
MHz	megahertz
nsec	nanosecond
V	volt
VDC	volts direct current

Related Documents

The following documents contains information that you may find helpful as you read this manual:

- *IEEE Standard Digital Interface for Programmable Instrumentation*, ANSI/IEEE Standard 488.1-1987
- *µPD7210 GPIB-IFC User's Manual*, NEC Electronics USA, Inc., One Natick Executive Park, Natick, MA 01760
- *µPD7210 Intelligent GPIB Interface Controller*, Engineering Data Sheet, NEC Electronics USA, Inc., Microcomputer Division
- *How to Interface a Microcomputer System to a GPIB*, (& The NEC µPD7210 TLC), NEC Electronics USA, Inc.
- *Intel iSBX Bus Specification*, (manual order number 142686-002) Intel Corporation, 3065 Bowers Ave., Santa Clara, CA 95051
- *Designing iSBX Multimodule Boards*, Stephen Grubb, OEM Microcomputer Systems Applications, Intel Corporation

Customer Communication

We appreciate communicating with the people who use our products. To make it easy for you to communicate with us, we provide a *User Comment Form* for documentation comments. Please mail it to the address printed at the bottom of the form.

If you encounter any technical problems, you can use the following toll-free number between the hours of 8:00 a.m. and 5:30 p.m. (central time) to reach the National Instruments applications engineering department:

(512) 794-0100
(800) IEEE-488 (toll-free U.S. and Canada)

Contents

Figures

Tables

Chapter 1
Introduction

This chapter contains general information about National Instruments GPIB-SBX interface, pictured in Figure 1-1. This information includes a brief description of the GPIB-SBX kit, a list of equipment supplied, and a list of optional equipment.

GPIB-SBX Interface Kit Description

The GPIB-SBX circuit board interfaces the IEEE-488 General Purpose Interface Bus (GPIB) to the iSBX (or SBX) bus. The SBX bus, designed by Intel Corporation, provides I/O expansion capability to host circuit boards such as single-board computers via piggy-back boards called *multimodules*. The GPIB-SBX meets all requirements of the IEEE-488 standard and the Intel iSBX bus standard.

The GPIB-SBX provides a means to implement a test and measurement or communication system with standard interconnecting cables. The GPIB-SBX interface kit includes hardware and programming examples to implement the GPIB functions. Optional cables can be ordered for interconnection with other devices on the GPIB.

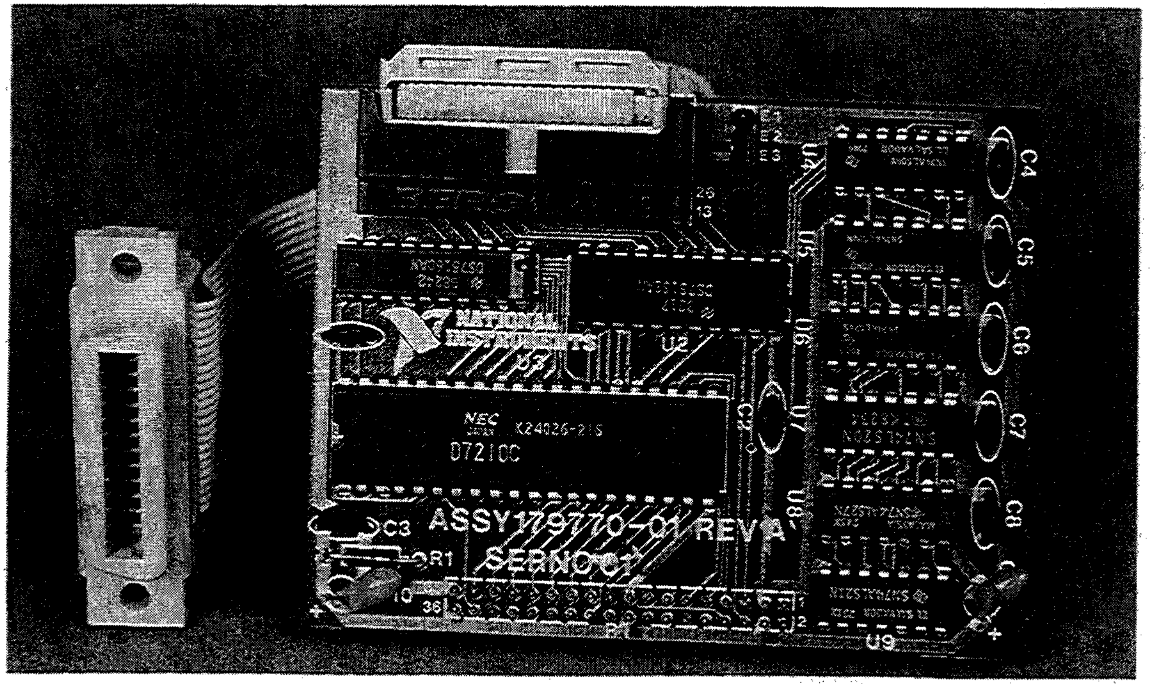


Figure 1-1. GPIB-SBX Interface Board

What Your Kit Should Contain

Your kit should contain the following components:

Component	Part Number
GPIB-SBX Interface Board	179770-01
20-in. Interface Cable (chassis mount)	179753-01
GPIB-SBX User Manual	320015-01

Make sure each of these items is in your kit. If any item is missing, contact National Instruments.

Optional Equipment

Equipment	Part Number
Interface Cable Assembly:	
2 m	179754-02
4 m	179754-04
Single-Shielded Cable:	
GPIB Type X1 Cable - 1 m	763001-01
GPIB Type X1 Cable - 2 m	763001-02
GPIB Type X1 Cable - 3 m	763001-03

Chapter 2
General Description

This chapter contains the physical and electrical specifications for the GPIB-SBX and describes the characteristics of key interface board components.

GPIB-SBX Specifications

The physical and electrical specifications are described in the following sections.

Physical Specifications

The GPIB-SBX measures 2.85 in. by 3.7 in. and is supplied with a 36-pin plug and spacing studs for mounting on the host board. A 26-pin male header connects to the GPIB adapter cable.

Electrical Specifications

Three major components of the GPIB-SBX interface connect directly to the SBX bus and the GPIB (via the cable supplied), as described in Table 2-1.

Table 2-1. Electrical Connections of the GPIB-SBX Interface

Component	Purpose	Connected Directly to
µPD7210	Talker/Listener Controller	SBX
75160A	Data Line Transceivers	GPIB
75162A	Bus Management and Handshake Line Transceivers	GPIB

The GPIB transceivers meet the requirements of the IEEE-488 standard. The GPIB-SBX requires a regulated +5 VDC power from the SBX bus. Current load is typically 0.5 A (1 A maximum).

All integrated circuits used on the GPIB-SBX meet the requirements of the SBX specification. Table 2-2 shows the loading on incoming SBX signal lines.

Table 2-2. Loading on Incoming SBX Signal Lines

Signal	**Loading**
MD7-MD0 MA2-MA0 MCS0 RESET IORD* IOWRT* MCLK	1 NMOS plus 1 or 2 ALSTTL Iih < 50 uA Iil < -0.2 mA
MDACK*	3 ASTTL plus 13K ohm pull-up

Table 2-3 shows the drive capability of output SBX signal lines.

Table 2-3. Drive Capability of Output SBX Signal Lines

Signal	**Loading**
MD7-MD0 MINTR0 MDRQT	2 mA @ 0.45 V, -0.4 mA @ 2.4 V
MWAIT	8 mA @ 0.35 V, -0.4 mA @ 3.4 V

Additional circuitry permits the GPIB-SBX to automatically become System Controller or give up control automatically when the GPIB-SBX receives the Take Control or Release Control auxiliary commands. No special software support or hardware setting are necessary to make the GPIB-SBX System Controller or not System Controller.

SBX Interface Bus Signals

The 36 SBX interface bus signals are grouped into six functional classes as follows:

- 9 Control Lines
- 5 Address and Chip Select Lines
- 8 to 16 Data Lines
- 2 Interrupt Lines
- 2 Option Lines
- 2 Power Lines

The following paragraphs describe the signals used by the GPIB-SBX. The GPIB-SBX uses 29 of the available 36 SBX bus signals. Table 2-4 lists all 36 signals and indicates which ones are used and which ones are not used.

Table 2-4. GPIB-SBX Connector Pin Assignments

Pin	Mnemonic	Description	Used/ Not Used
1	+12V	+12 Volts	Not Used
2	-12V	-12 Volts	Not Used
3	GND	Signal Ground	Used
4	+5V	+5 Volts	Used
5	MRESET	Reset	Used
6	MCLK	M Clock	Used
7	MA2	M Address 2	Used
8	MPST*	M Present	Used
9	MA1	M Address 1	Used
10		Reserved	Not Used
11	MA0	M Address 0	Used
12	MINTR1	M Interrupt 1	Grounded
13	IOWRT*	I/O Write Command	Used

(continues)

Table 2-4. GPIB-SBX Connector Pin Assignments (continued)

Pin	Mnemonic	Description	Used/ Not Used
14	MINTR0	M Interrupt 0	Used
15	IORD*	I/O Read Command	Used
16	MWAIT*	M Wait	Used
17	GND	Signal Ground	Used
18	+5V	+5 Volts	Used
19	MD7	M Data Bit 7	Used
20	MCS1*	M Chip Select 1	Not Used
21	MD6	M Data Bit 6	Used
22	MCS0*	M Chip Select 0	Used
23	MD5	M Data Bit 5	Used
24		Reserved	Not Used
25	MD4	M Data Bit 4	Used
26	TDMA	Terminate DMA	Grounded
27	MD3	M Data Bit 3	Used
28	OPT1	Option 1	Not Used
29	MD2	M Data Bit 2	Used
30	OPT0	Option 0	Not Used
31	MD1	M Data Bit 1	Used
32	MDACK*	M DMA Acknowledge	Used
33	MD0	M Data Bit 0	Used
34	MDRQT	M Request	Used
35	GND	Signal Ground	Used
36	+5V	+5 Volts	Used

Control Lines

The control lines of the SBX bus determine when events occur and are described in the following paragraphs.

Command Lines (IORD*, IOWRT*)

The command lines are active-low signals that determine the direction of data, that is, from the GPIB-SBX to the host board (read), or from the host board to the GPIB-SBX (write). When either of these is active, the address lines are valid.

DMA Lines (MDRQT, MDACK*, TDMA)

The DMA lines are used to coordinate DMA transfers using the DMA Controller function of the host board. MDRQT is an active-high output signal used by the GPIB-SBX to request a DMA cycle. MDACK* is an active-low signal from the host board to the GPIB-SBX acknowledging that the DMA cycle has been granted. TDMA is an active-high signal from the GPIB-SBX to the host board used to terminate a DMA cycle. TDMA is tied to ground so that it does not float to a true state.

Initialize Line (MRESET)

MRESET is an active-high signal used by the host board to place the GPIB-SBX in a known quiescent state.

Clock Line (MCLK)

MCLK is a 10-MHz timing signal provided by the host board to the GPIB-SBX.

System Control Lines (MWAIT*, MPST*)

The system control signal lines are output signals from the GPIB-SBX to synchronize the transfer of data across the SBX bus and indicate its presence on the SBX bus.

Address and Chip Select Lines (MA2-MA0, MCS1*-MCS0*)

The host board generates the address and chip select signals. The signals are generally derived from a subset of the address bus lines of the computer bus. The GPIB-SBX requires the three address lines and MCS0* for selecting the eight interface registers.

The address lines select one of the eight interface registers of the GPIB-SBX.

The GPIB-SBX decodes eight consecutive I/O mapped interface register addresses, as shown in Table 2-5.

Table 2-5. GPIB-SBX Internal Registers

Value of MA2-MA0	Read Registers	Write Registers
000	Data In (DIR)	Control/Data Out (CDOR)
001	Interrupt Status 1 (ISR1)	Interrupt Mask 1 (IMR1)
010	Interrupt Status 2 (ISR2)	Interrupt Mask 2 (IMR2)
011	Serial Poll Status (SPSR)	Serial Poll Mode (SPMR)
100	Address Status (ADSR)	Address Mode (ADMR)
101	Command Pass Through (CPTR)	Auxiliary Mode (AUXMR)
110	Address 0 (ADR0)	Address (ADR)
111	Address 1 (ADR1)	End of String (EOSR)

MCS0* is an active-low signal generated by the host board to enable communication with the GPIB-SBX. MCS1* is not used.

Data Lines (MD7-MD0)

The MD signal lines are active high and are used to transmit data in both directions across the SBX bus. MD7 is the most significant bit.

Interrupt Line (MINTR1-MINTR0)

MINTR0 is an active-high signal generated by the GPIB-SBX to send an interrupt to the host board. MINTR1 is tied to ground so that it does not float to a true state.

Power and Ground Lines (+5 V, +-12 V, GND)

The GPIB-SBX requires a +5 volt power source and ground. Twelve volt power is not used.

Data Transfer Modes and Performance

The GPIB-SBX is capable of operating in two modes of data transfer: Programmed I/O or Direct Memory Access (DMA). In Programmed I/O mode, data is written to the CDOR or read from the DIR with port output/input or memory write/read instructions, depending upon how the address space of the GPIB-SBX is mapped. Performance in this mode is highly dependent upon the structure of the driver software that manipulates the hardware. Using the sample code in Appendix B with a 14-MHz Z80 microprocessor, data transfer rates up to about 30 kbytes/sec can be achieved. If the driver is written in a high-level language and/or is embedded within an operating system, the transfer rate usually drops significantly due to code inefficiencies and system overhead. In DMA mode, data is transferred directly between memory and the CDOR/DIR, using the facilities of a DMA Controller that is external to the GPIB-SBX (that is, it is a function of the host board or the computer system). The data transfer in this mode can exceed 500 kbytes/sec provided that the Controller, memory, and external GPIB devices are capable of sustaining this rate.

Description of the General Purpose Interface Bus (GPIB)

The IEEE-488 GPIB provides a means for communications among a group of interconnected devices. Two types of messages are carried by the bus:

- Interface messages, which are used for bus management, and

- Device-dependent messages, which are communicated between the various devices via the interface bus, but are not used or processed by the bus.

The three types of devices that organize and manage the flow of information on the bus are a Listener, a Talker, and (optionally) a Controller. A Listener has the capability of being addressed by an interface message to receive device-dependent messages. A Talker has the capability of being addressed by an interface message to send device-dependent messages. A Controller can address devices to Listen or Talk. A Controller can also send interface messages to command other specific actions within interfaced devices. A single bus can have one or more Controllers. If more than one Controller is connected to the bus, one is designated as the System Controller and can temporarily pass control to any other Controller.

The GPIB-SBX is capable of being a Listener, a Talker, a Controller, and, in particular, a System Controller.

GPIB Operation

This section contains a simplified description of the operation of the GPIB. For more details, refer to *IEEE Standard Digital Interface for Programmable Instrumentation*.

Bus operation is divided into two logical functions: interface functions (for bus management) and device functions (for device control and communication). The interface functions, described in this manual, are basically used to establish an environment in which device functions may be performed. The device functions are unique to individual instruments and are beyond the intended scope of this manual.

The Source Handshake (SH) and Acceptor Handshake (AH) interface functions are used to transmit and receive bit-parallel, byte-serial messages (multiline messages) on the bus. These messages are interpreted either as commands to the interface or as data bytes for the device, depending upon the status of the bus signal Attention (ATN). ATN is asserted for interface commands.

The Talker (T) or Extended Talker (TE) and Listener (L) or Extended Listener (LE) interface functions are used to set up devices for data transfers. The transfers are made using the handshake functions. When an interface has been addressed as a Talker, the applicable device function is allowed to use the interface Source Handshake function to transmit data bytes. The device functions of any interfaces that were addressed as Listeners must use the Acceptor Handshake function to receive the data bytes. Only one device at a time can be the Talker, although multiple Listeners can exist.

The Controller (C) function is the originator of all interface messages. Only one interface at a time can be the Controller-In-Charge; protocol exists to pass the control capability among interfaces. When the Controller-In-Charge is asserting the ATN line, it is in its active state and is often referred to as the Active Controller. When not asserting ATN, it is in its standby state. Any Controller that is not the Controller-In-Charge remains in its idle state. At any time, there is at most one interface on the bus that has the capability to make itself the Controller-In-Charge. It is referred to as the System Controller.

The Controller is responsible for addressing and unaddressing Talkers and Listeners, as well as performing other bus management operations. These other operations include conducting a Parallel Poll using the uniline message Identify (IDY); setting device functions in remote or local mode using the uniline message Remote Enable (REN); and initializing the bus by asserting the uniline message Interface Clear (IFC). The latter two operations are actually capabilities only of the System Controller and are not transferable by the transfer of control protocol.

GPIB Signal Lines

A total of 24 signal lines are used to implement the bus. Of these lines, 16 are signal lines, one is ground, one is the cable shield, and six are twisted pair common for six of the signal lines. The 16 signal lines are used to carry all information, interface messages, and device-dependent messages among interconnected devices.

The 16 signal lines are organized into three sets, as follows:

- 8 data input/output lines
- 3 handshake lines
- 5 interface management signal lines

Negative logic with standard TTL levels is used on the GPIB. That is, a false (0) logic state corresponds to a TTL high level of 2.0 V or higher, and a true (1) logic state corresponds to a TTL low level of 0.8 V or lower.

The eight data lines, DI01-DI08, carry all data, including input, output, and device-dependent messages. In many instruments, data is based upon the seven bit ASCII code set.

The three handshake lines, NRFD, DAV, and NDAC, are used to effect the transfer of each byte of data on the DIO lines from an addressed Talker to all addressed Listeners. The three handshake lines provide a means to asynchronously transfer data between instruments.

The NRFD (Not Ready for Data) line is used to indicate the condition of readiness of devices to accept data. All instruments drive NRFD false when ATN becomes true. Only addressed Listeners drive NRFD false when ATN becomes false. The NRFD line is monitored by the Controller when ATN is true and by the addressed Talker when ATN is false. The NRFD line is false when all Listeners are ready for data and true when one or more Listeners are not ready for data.

The DAV (Data Valid) line is used to indicate the validity of data on the data lines. DAV is driven by the Controller when ATN is true and by the addressed Talker when ATN is false. The DAV line is monitored by all instruments if ATN is true and by addressed Listeners when ATN is false.

The NDAC (Not Data Accepted) line is used to indicate acceptance of data by addressed Listeners when ATN is false and acceptance of commands by all devices when ATN is true. Listeners indicate acceptance of data by setting NDAC false. When NDAC is true, one or more Listeners have not accepted the data.

The five bus management lines are ATN, IFC, REN, SRQ, and EOI. ATN and IFC are used by all instruments while the remaining three may or may not be used by a particular instrument.

All devices on the GPIB must monitor the ATN line. This line is set true by the Controller-In-Charge when it sends interface messages, such as device talk and listen addresses, secondary addresses, and polling configuration messages. When ATN is false, the active Talker can send device-dependent messages, such as data and programming information, to active Listeners.

The IFC (Interface Clear) line is used by the System Controller to place the bus in a known quiescent state. The IFC line can only be driven true by the System Controller and must be monitored by all other instruments. In order to clear a device, the IFC line must be set true for at least 100 µsec. IFC may be set true by the System Controller at any time.

The REN (Remote Enable) line is used to send the REN message. REN must be asserted as one of the conditions for operating an instrument in remote mode. The use of the remote function is optional. The REN line is driven true only by the System Controller and can be changed at any time. Instruments that use the REN line must monitor it at all times and return to local control whenever it becomes false.

The SRQ (Service Request) line is used by an instrument to asynchronously request service from the Controller-In-Charge.

The EOI (END Or Identify) line is used either to indicate the end of a data string or to conduct a Parallel Poll, depending upon the state of the ATN line. When ATN is false, the addressed Talker can send the END message to indicate the end of its data by setting EOI true at the same time it places the last data byte on the DIO lines. The Controller-In-Charge can send the IDY message to initiate a Parallel Poll of all instruments with Parallel Poll capability by setting ATN and EOI true simultaneously.

GPIB Physical Characteristics

Bus cables are used to interconnect instruments. There are three basic restrictions on cable length:

- The maximum length of any single span between two devices (loads) is 4 m.
- The maximum average length is 2 m per device.
- The maximum total length for all interconnected devices is 20 m.

With special high-performance cable, such as the HP 10833 series, the distance between devices is not critical, as long as the second and third restrictions are met. Instruments can be connected in a linear or star configuration. For applications requiring greater device separation, an extender such as National Instruments GPIB-100 can be used.

Two 24-pin piggy-back connectors, one male and one female, are used on either end of the interconnecting cables. An overall cable shield is used to reduce susceptibility to noise. This shield is normally grounded only at one instrument, usually the System Controller.

GPIB-SBX Functional Description

In the simplest terms, the GPIB-SBX can be characterized as a bus translator, converting messages and signals of the SBX bus into appropriate GPIB messages and signals. Expressed in GPIB terminology, the GPIB-SBX implements GPIB interface functions for communicating with other GPIB devices and device functions for communicating with the central processor and memory. From the point of view of the host board, the GPIB-SBX is an interface to the outside world.

Figures 2-1 and 2-2 show typical applications for the GPIB-SBX. In Figure 2-1, the GPIB-SBX is used to interface an assortment of test instruments to a Multibus computer system, which then functions as an intelligent System Controller. This is the traditional role of the GPIB.

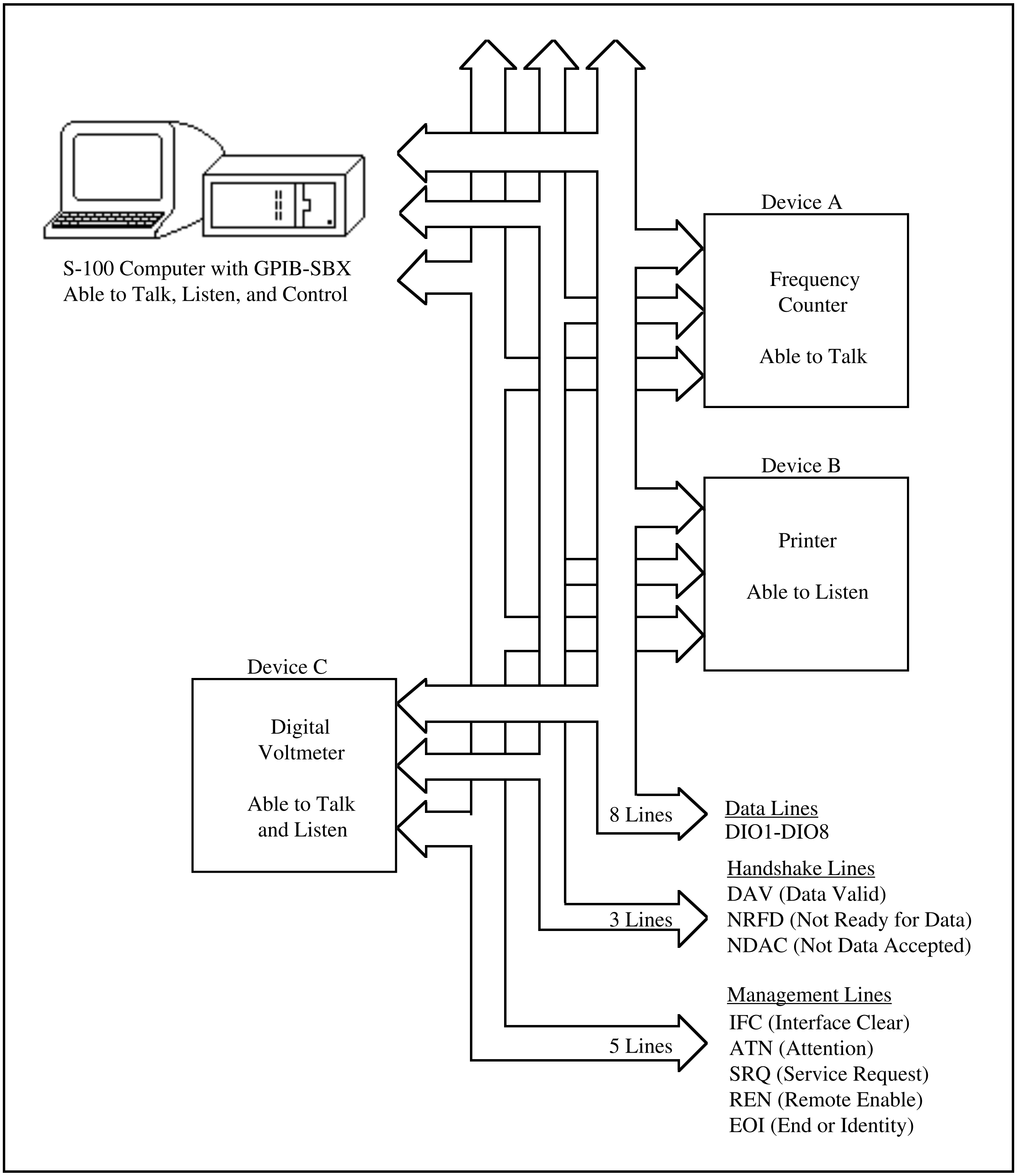


Figure 2-1. GPIB-SBX with a Multibus Computer

Figure 2-2 shows the GPIB-SBX used along with other National Instruments interface boards to connect a Multibus computer to other processors in order, for example, to transfer files electrically rather than manually (via a removable storage medium) or to perform other interprocessor communication functions.

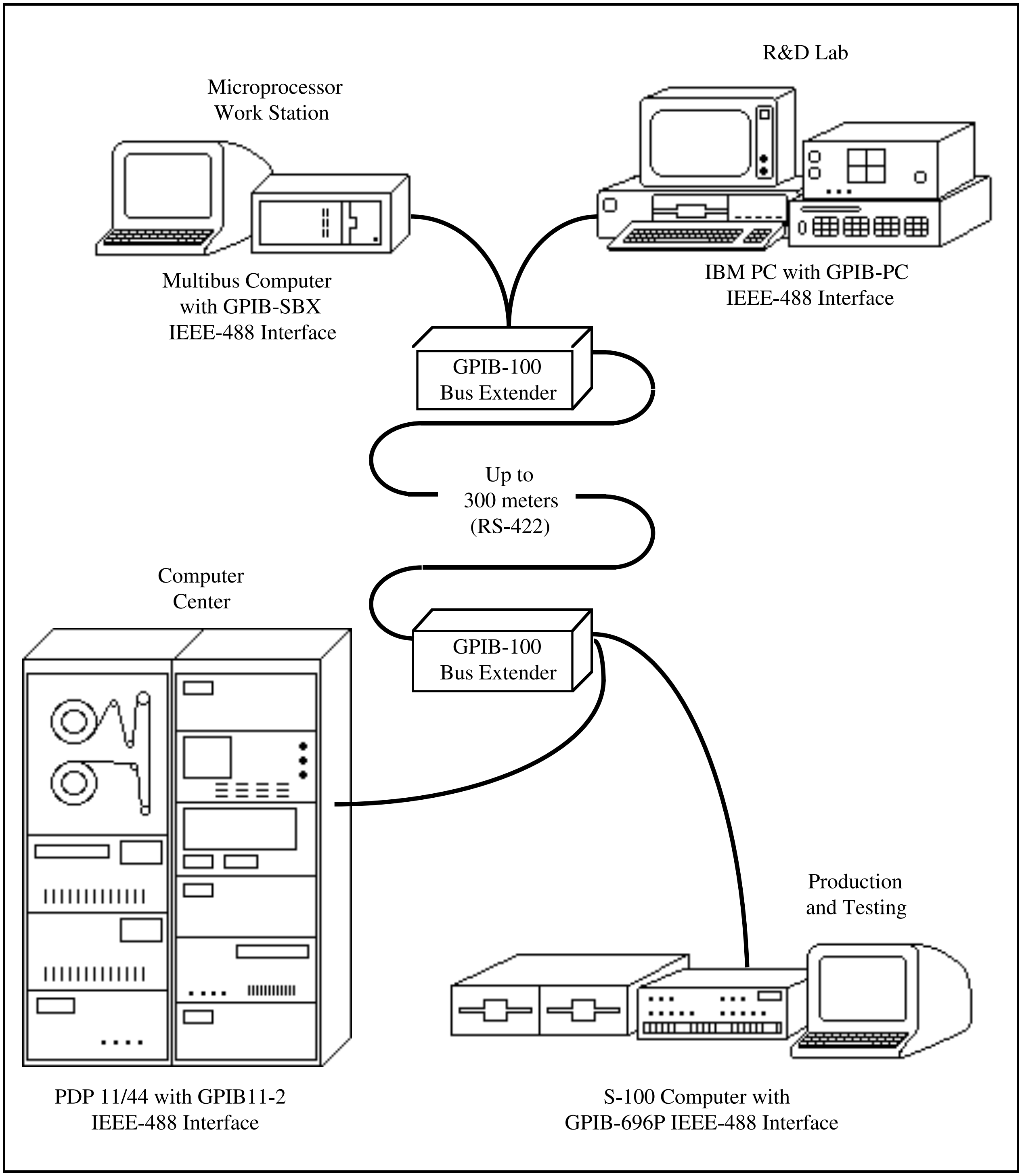

R&D Lab
Microprocessor Work Station
Multibus Computer with GPIB-SBX IEEE-488 Interface
IBM PC with GPIB-PC IEEE-488 Interface
GPIB-100 Bus Extender
Up to 300 meters (RS-422)
Computer Center
GPIB-100 Bus Extender
Production and Testing
PDP 11/44 with GPIB11-2 IEEE-488 Interface
S-100 Computer with GPIB-696P IEEE-488 Interface

Figure 2-2. GPIB-SBX in a Multiprocessor Application

Figure 2-3 is a block diagram of the GPIB-SBX-1.

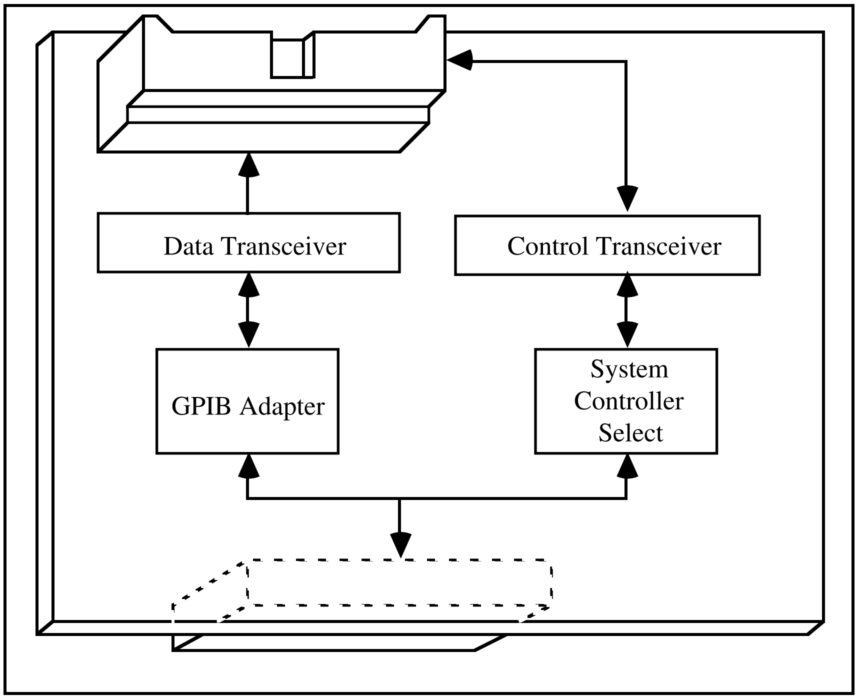


Figure 2-3. GPIB-SBX Block Diagram

The interface consists of these major sections that are discussed in greater detail in Chapters 4, 5, and 6:

- SBX Bus Interface
- Host Board Function - GPIB TLC
  - Data In Register (DIR)
  - Command/Data Out Register (CDOR)
  - Interrupt Status Register 1 (ISR1)
  - Interrupt Mask Register 1 (IMR1)
  - Interrupt Status Register 2 (ISR2)
  - Interrupt Mask Register 2 (IMR2)
  - Serial Poll Status (SPSR)
  - Serial Poll Mode (SPMR)
  - Address Status Register (ADSR)
  - Address Mode Register (ADMR)
  - Command Pass Through (CPTR)
  - Auxiliary Mode Register (AUXMR)
  - Address Register 0 (ADR0)
  - Address Register (ADR)
  - Address Register 1 (ADR1)
  - End of String Register (EOSR)

- GPIB Interface

The GPIB-SBX interface consists of the connector that mates the GPIB-SBX to the host circuit card. The host side of the interface provides all buffering of address, data, status, and control lines and required address decoding, and DMA controls. The program registers within the TLC control program accesses to configure, control, and monitor the GPIB interface functions. The program registers can be used by the interface to interrupt the controlling program to inform it of the occurrence of an anticipated event. Special transceivers interface the TLC to the GPIB itself.

Table 2-6 lists the capabilities of the GPIB-SBX in terms of the codes in Appendix C of the IEEE-488 standard.

Table 2-6. GPIB-SBX IEEE-488 Interface Capabilities

Capability Code	Description
SH1	Complete Source Handshake capability
AH1	Complete Acceptor Handshake capability DAC and RFD holdoff on certain events
T5	Complete Talker capability Basic Talker Serial Poll Talk Only mode Unaddressed on MLA Send END or EOS Dual primary addressing
TE5	Complete Extended Talker capability Basic Extended Talker Serial Poll Talk Only Mode Unaddressed on MSA*LPAS Send END or EOS Dual extended addressing with software assist
L3	Complete Listener capability Basic Listener Listen Only mode Unaddressed on MTA Detect END or EOS Dual primary addressing
LE3	Complete Extended Listener capability Basic Listener Listen Only mode Unaddressed on MSA*TPAS Detect END or EOS Dual extended addressing with software assist

(continues)

Table 2-6. GPIB-SBX IEEE-488 Interface Capabilities (continued)

Capability Code	Description
SR1	Complete Service Request capability
RL1	Complete Remote Local capability with software interpretation
PP1	Remote Parallel Poll configuration
PP2	Local Parallel Poll configuration with software assist
DC1	Complete Device Clear capability with software interpretation
DT1	Complete Device Trigger capability with software interpretation
C1-5	Complete Controller capability System Controller Send IFC and take charge Send REN Respond to SRQ Send interface messages Receive Control Pass Control Parallel Poll Take Control Synchronously or Asynchronously
E1/2	Three-state bus drivers with automatic switch to open collector drivers during Parallel Poll

The GPIB-SBX has complete Source and Acceptor Handshake capability.

- The GPIB-SBX can operate as a basic Talker or Extended Talker and can respond to a Serial Poll. It can be placed in a Talk Only mode, and is unaddressed to Talk when it receives its Listen address.

- The interface can operate as a basic Listener or Extended Listener. It can be placed in a Listen Only mode, and is unaddressed to Listen when it receives its Talk address.

The GPIB-SBX has full capabilities for requesting service from another Controller. The ability to place the GPIB-SBX in local mode is included but the interpretation of remote versus local mode is software-dependent. Full Parallel Poll capability is included in the interface, although local configuration requires software assistance. Device Clear and Trigger capability is included in the interface, but the interpretation is software-dependent. All Controller functions as specified by the IEEE-488 standard are included in the GPIB-SBX.

These include the capability to do the following:

- Be System Controller
- Initialize the interface
- Send Remote Enable
- Respond to Service Request
- Send multiline command messages
- Receive control
- Pass control
- Conduct a Parallel Poll
- Take control synchronously or asynchronously

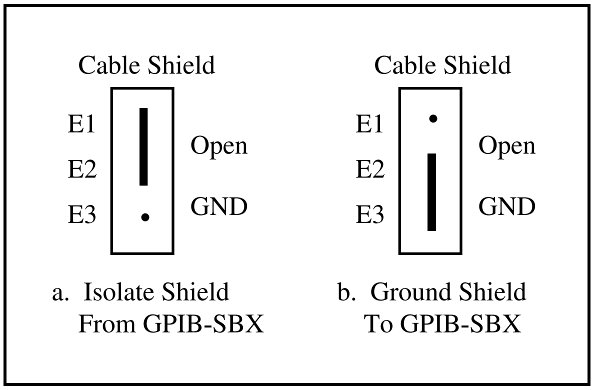
Chapter 3
Configuration and Installation

This chapter describes the configuration and installation of the GPIB-SBX.

Configuration

The GPIB cable shield is usually connected to ground at only one point on the IEEE-488 bus, and that point is usually at the System Controller. A jumper is provided on the GPIB-SBX for connection of the GPIB cable ground to the SBX bus digital logic ground.

Connect E2 to E3 to short the GPIB cable shield to the SBX bus digital logic ground. Connect E1 to E2 to leave the GPIB cable shield unconnected. Select one configuration depending upon whether or not the GPIB-SBX is the GPIB System Controller and whether or not the shield is grounded elsewhere. Figure 3-1 shows the two possible configurations.



a. Isolate Shield From GPIB-SBX

b. Ground Shield To GPIB-SBX

Figure 3-1. GPIB Cable Shield Grounding

Installation

The GPIB-SBX is installed on a host board by means of the SBX connector and a nylon spacer provided in the interface kit.

Warning: Never attempt to install the GPIB-SBX on a host board that has power applied to it. Serious damage to the GPIB-SBX, the host board and the computer can result. It is recommended that you remove the host board from the computer bus according to the computer manufacturer's instructions before attempting to install the GPIB-SBX.

Perform the following steps in the order indicated:

1. Carefully align the male connector of the GPIB-SBX with the female connector on the host board and press firmly into place.

2. Place the nylon threaded spacer between the component side of the host board and the circuit side of the GPIB-SBX directly in line with the mounting holes provided.

3. Insert screws from the component side of the GPIB-SBX into the spacer and from the circuit side of the host board into the spacer.

4. Plug the GPIB cable into the J1 header of the GPIB-SBX.

5. Insert the host board into the computer bus according to the manufacturer's instructions.

6. Attach the ribbon cable to the GPIB-SBX and mount the GPIB plug in the computer chassis.

Figure 3-2 illustrates the installation steps.

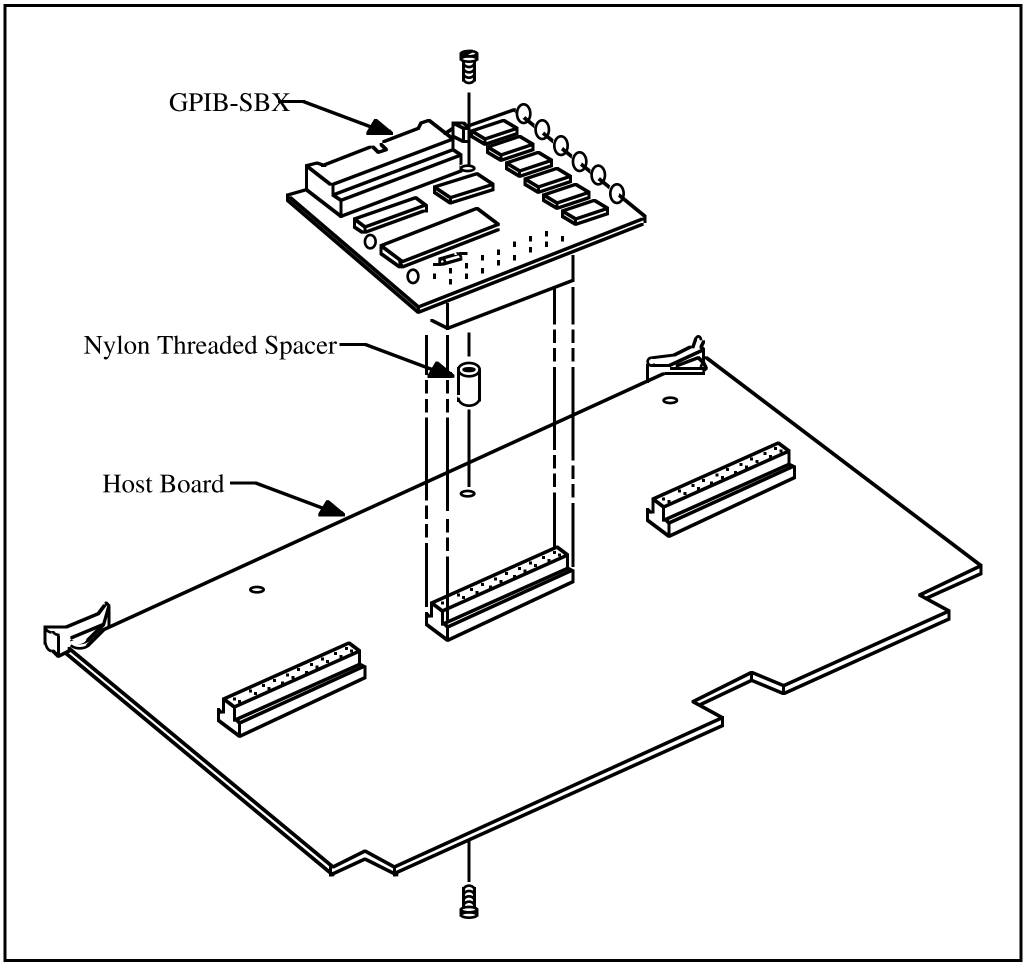


Figure 3-2. GPIB-SBX Installed on Host Board

Cabling

Two optional cables are available to connect the GPIB-SBX to other GPIB devices. A short 20-in. cable is used where the outboard GPIB-style connector is mounted to the back panel of the computer chassis, and other GPIB devices are connected using standard GPIB extension cables. A longer 2-m or 4-m GPIB cable is used where the outboard end of the cable is connected directly to a GPIB device. The cable is generally run through the back panel of the computer and strain relieved in some manner.

Both cables connect to the GPIB-SBX at the right-angle header labeled J1 at the top of the circuit card. Take care when installing the connector. Pin 26 of the cable connector and the associated pin of the J1 are keyed to prevent the connectors from being installed backwards.

Caution: National Instruments manufactures two styles of the longer cable, with the only difference between the two being the order of the row assignment at the connector. Use only the cable with RED shrink tubing at the point where the cable sheath is cut back.

Figure 3-3 shows the signals present on the GPIB cable connector. Refer to Figure 3-4 to ensure proper cable-to-interface board connection.

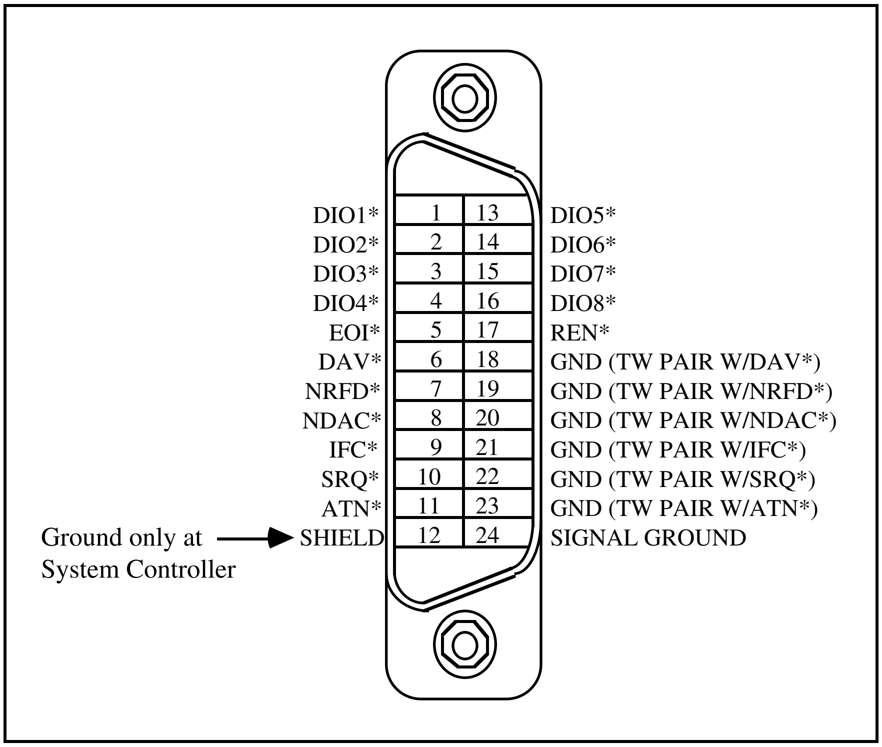

DIO1* 1 13 DIO5*
DIO2* 2 14 DIO6*
DIO3* 3 15 DIO7*
DIO4* 4 16 DIO8*
EOI* 5 17 REN*
DAV* 6 18 GND (TW PAIR W/DAV*)
NRFD* 7 19 GND (TW PAIR W/NRFD*)
NDAC* 8 20 GND (TW PAIR W/NDAC*)
IFC* 9 21 GND (TW PAIR W/IFC*)
SRQ* 10 22 GND (TW PAIR W/SRQ*)
ATN* 11 23 GND (TW PAIR W/ATN*)
SHIELD 12 24 SIGNAL GROUND
Ground only at System Controller

Figure 3-3. GPIB Cable Connector

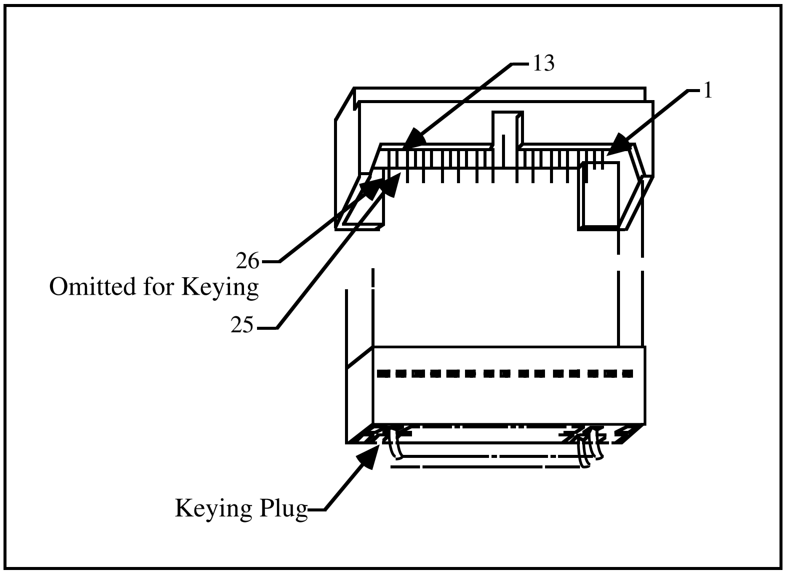


Figure 3-4. GPIB-SBX Cable-to-Interface Board Connection

Verification Testing

The GPIB-SBX is tested twice and receives a 96-hour burn-in before shipment. Nonetheless, a performance verification test should be run to ensure that the board has not been damaged during shipment and also to ensure that the board has been configured correctly. To do this requires an interactive control program or an equivalent mechanism, such as front panel control switches or front panel emulator, that can load and read memory and I/O addresses. (National Instruments highly recommends such a mechanism for developing and debugging application software that uses the GPIB-SBX.)

The tests presented in Chapter 7 of this manual consist of a series of steps written in a pseudo (processor-independent) language and supplemental instructions. The steps generally involve writing programming data to specific GPIB-SBX device registers, followed by reading other GPIB-SBX registers to verify that the programming was correct. These tests, which are all performed before shipment, exercise virtually all of the major functions of the GPIB-SBX, including I/O communications and GPIB communications. All functions except GPIB communications can be performed stand-alone, (that is, without another GPIB device). To completely check the GPIB functions requires a bus tester or analyzer (such as National Instruments GPIB-400) that can monitor and control GPIB signals; emulate GPIB Talker, Listener, and Controller devices; and single-step through the Source and Acceptor Handshakes.

Chapter 4
Register Descriptions

Introduction

All software control of the GPIB-SBX is performed through a set of interface registers located within a block of eight consecutive I/O addresses. Some of the registers are read only and some are write only. Some registers are not storage registers at all, but buffers through which status signals can be read or through which control pulses can be sent.

Terminology

Refer to the Preface for a description of the terminology used in this chapter. Once a mnemonic has been defined, the mnemonic is used in the remainder of the chapter. Mnemonics are assigned to messages, states, registers and bits. Most mnemonics contain a clue to their meaning. Table 4-1 contains a list of clues for which to look. Refer also to Appendix D, which contains an alphabetical list of mnemonics.

Table 4-1. Clues to Understanding Mnemonics

Clue	**Mnemonic Probably Stands For:**
Ends in IE	Interrupt enable bit
Ends in EN	Enable bit
4 letters, ends in S	Interface function as defined in the IEEE-488 standard
Ends in R, R0, R1, R2	GPIB program register
3 letters, uppercase	Remote GPIB message
3 letters, lowercase	Local GPIB message

Interface Registers

The interface registers are contained in the NEC µPD7210 Talker/Listener/Controller (TLC) integrated circuit. Each of the eight interface registers is addressed using the three register select lines (MA2-MA0), the I/O Read or Write command (IORD, IOWRT*), and the SBX device select command.

Figures 4-1 and 4-2 show the register and bit mnemonics of each GPIB-SBX register, its I/O port offset, and its read/write accessibility (the value of MA2-MA0). Figure 4-1 shows the regular GPIB-SBX interface registers. Figure 4-2 shows the hidden registers and illustrates the method of writing to those registers via the Auxiliary Mode Register (explained in the *Hidden Registers* section later in this chapter). A detailed function description of all eight interface registers is provided in the paragraphs following the figures.

Legend

(Contents of Read Register)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

(Contents of Write Register)

	Address Offset (hex)									**Read/ Write**
DIR	+0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0	R
CDOR		CDO7	CDO6	CDO5	CDO4	CDO3	CDO2	CDO1	CDO0	W
ISR1	+1	CPT	APT	DET	END RX	DEC	ERR	DO	DI	R
IMR1		CPT IE	APT IE	DET IE	END IE	DEC IE	ERR IE	DO IE	DI IE	W
ISR2	+2	INT	SRQI	LOK	REM	CO	LOKC	REMC	ADSC	R
IMR2		0	SRQI IE	DMAO	DMAI	CO IE	LOKC IE	REMC IE	ADSC IE	W
SPSR	+3	S8	PEND	S6	S5	S4	S3	S2	S1	R
SPMR		S8	rsv	S6	S5	S4	S3	S2	S1	W
ADSR	+4	CIC	ATN*	SPMS	LPAS	TPAS	LA	TA	MJMN	R
ADMR		ton	lon	TRM1	TRM0	0	0	ADM1	ADM0	W
CPTR	+5	CPT7	CPT6	CPT5	CPT4	CPT3	CPT2	CPT1	CPT0	R
AUXMR		CNT2	CNT1	CNT0	COM4	COM3	COM2	COM1	COM0	W
ADR0	+6	X	DT0	DL0	AD5-0	AD4-0	AD3-0	AD2-0	AD1-0	R
ADR		ARS	DT	DL	AD5	AD4	AD3	AD2	AD1	W
ADR1	+7	EOI	DT1	DL1	AD5-1	AD4-1	AD3-1	AD2-1	AD1-1	R
EOSR		EOS7	EOS6	EOS5	EOS4	EOS3	EOS2	EOS1	EOS0	W

Note: X indicates a don't care bit.

Figure 4-1. GPIB-SBX Interface Registers

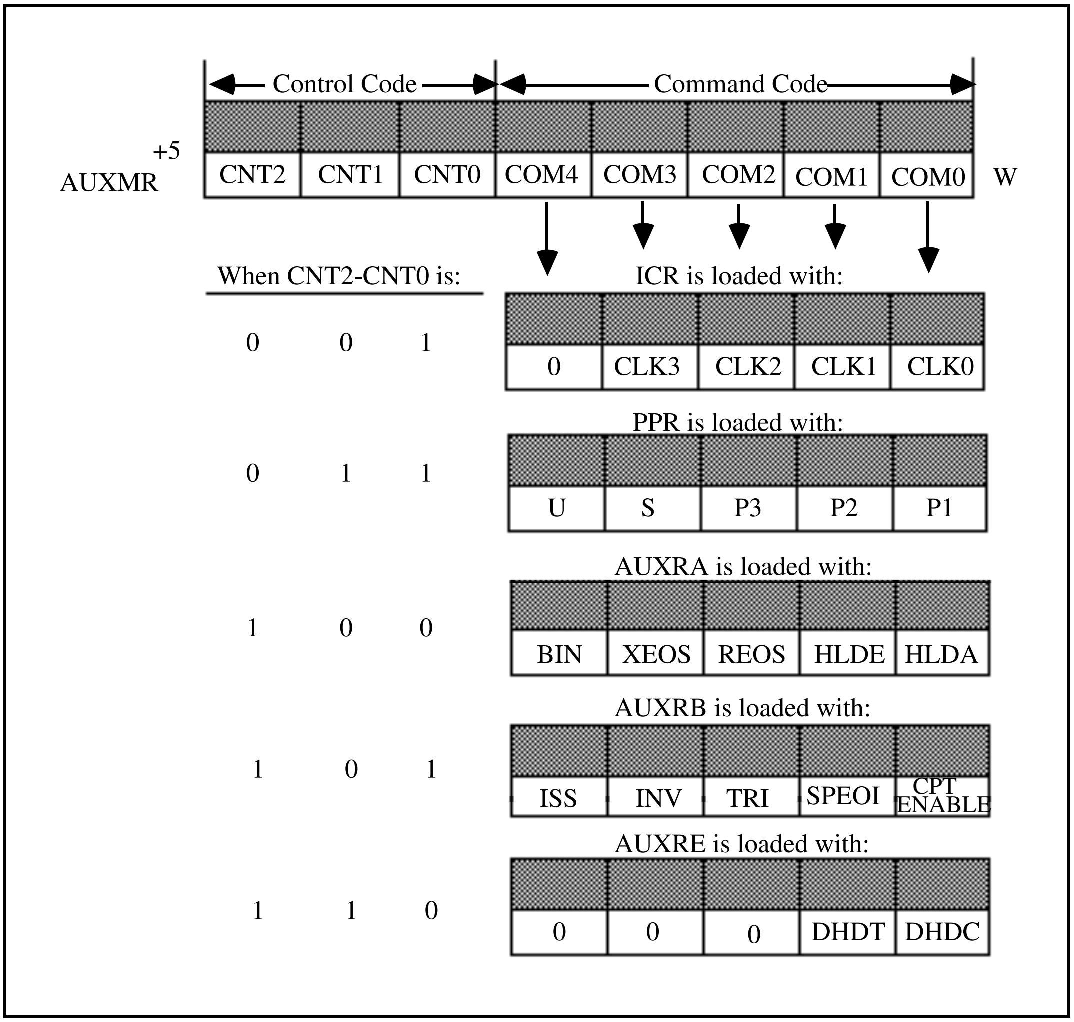


Figure 4-2. Writing to the Hidden Registers

Data In Register (DIR)

I/O Port Offset: + 0

Attributes: Read Only

7	6	5	4	3	2	1	0	R
DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0	

The Data In Register (DIR) is used to move data from the GPIB to the SBX bus when the interface is a Listener. Incoming information is separately latched by this register and is not destroyed by a write to the Command/Data Out Register (CDOR). The GPIB Ready For Data (RFD) message is held false until the byte is removed from the DIR by an I/O read from the SBX bus. The Acceptor Handshake (AH) completes automatically after the byte has been read. In RFD Holdoff mode (refer to the description of Auxiliary Register A in this chapter), the GPIB Handshake is not finished until the Finish Handshake (FH) auxiliary command is issued telling the TLC to release the Holdoff. By using the RFD Holdoff mode, the same byte may be read several times or a GPIB Talker may be held off until the program is ready to proceed.

DI0 is the least significant bit of the data byte and corresponds to GPIB DI01. DI7 is the most significant bit of the data byte and corresponds to GPIB DI08.

Bit	**Mnemonic**	**Description**
7-0r	DI[7-0]	Data In Bits 7 through 0

Command/Data Out Register (CDOR)

I/O Port Offset: + 0

Attributes: Write Only

7	6	5	4	3	2	1	0
CDO7	CDO6	CDO5	CDO4	CDO3	CDO2	CDO1	CDO0

W

The Command/Data Out Register (CDOR) is used to move data from the SBX bus to the GPIB when the interface is the GPIB Talker or the Active Controller. Outgoing data is separately latched by this register and is not destroyed by a read from the DIR. When a byte is written to the CDOR, the TLC GPIB Source Handshake (SH) function is initiated and the byte is transferred to the GPIB.

Bit	**Mnemonic**	**Description**
7-0w	CDO[7-0]	Command/Data Out Bits 7 through 0

Interrupt Status Register 1 (ISR1)

I/O Port Offset: + 1

Attributes: Read Only,
Bits are cleared when read

Interrupt Mask Register 1 (IMR1)

I/O Port Offset: + 1

Attributes: Write Only

7	6	5	4	3	2	1	0	
CPT	APT	DET	END RX	DEC	ERR	DO	DI	R
CPT IE	APT IE	DET IE	END IE	DEC IE	ERR IE	DO IE	DI IE	W

The Interrupt Status Register 1 (ISR1) is composed of eight Interrupt Status bits. The Interrupt Mask Register 1 (IMR1) is composed of eight Interrupt Enable bits that directly correspond to the Interrupt Status bits in ISR1. As a result, ISR1 and IMR1 service eight possible interrupt conditions, where each condition has an Interrupt Status bit and an Interrupt Enable bit associated with it. If the Interrupt Enable bit is true when the corresponding status condition or event occurs, an interrupt request is generated. Bits in ISR1 are set and cleared by the TLC regardless of the status of the Interrupt Enable bits in IMR1. If an interrupt condition occurs at the same time ISR1 is being read, the TLC holds off setting the corresponding Status bit until the read has finished.

Bit	Mnemonic	Description
7r	CPT	Command Pass Through Bit
7w	CPT IE	Command Pass Through Interrupt Enable Bit

CPT is set on:

> [UCG + ACG & (TADS + LADS)] & undefined & ACDS & -(CPT ENABLE) + UDPCF & SCG & ACDS & CPT ENABLE

CPT is cleared by:

> pon + (read ISR1)

Notes

UCG:	GPIB Universal Command Group message
ACG:	GPIB Addressed Command Group message
TADS:	GPIB Talker Addressed State
LADS:	GPIB Listener Addressed State

Bit **Mnemonic** **Description**

defined:	GPIB command automatically recognized and executed by TLC
undefined:	GPIB command not automatically recognized and executed by TLC
ACDS:	GPIB Accept Data State
CPT ENABLE:	AUXRB[0]w
UDPCF:	Undefined Primary Command Function
SCG:	GPIB Secondary Command Group message
pon:	Power On Reset
TAG:	GPIB Talk Address Group message
LAG:	GPIB Listen Address Group message
read ISR1:	Bit is cleared immediately after it is read

UDPCF is set on:

> [UCG + ACG & (TADS + LADS)] & undefined & ACDS & CPT ENABLE

UDPCF is cleared on:

> [(UCG + ACG) & defined + TAG + LAG] & ACDS + CPT ENABLE* + pon

The CPT bit flags the occurrence of a GPIB command not recognized by the TLC, and all following GPIB secondary commands when the Command Pass Through feature is enabled by the CPT ENABLE bit, AUXRB[0]w. Any GPIB command message not decoded by the TLC is treated as an undefined command (for example, the GO TO Local command, GTL); however, any addressed command is automatically ignored when the TLC is not addressed.

Undefined commands are read using the Command Pass Through Register (CPTR). The TLC holds off the GPIB Acceptor Handshake in the Accept Data State (ACDS) until the Valid auxiliary command function code, 0F hex, is written to the AUXMR. If the CPT feature is not enabled, undefined commands are simply ignored.

6r	APT	Address Pass Through
6w	APT IE	Address Pass Through Interrupt Enable

APT is set by:

> ADM1 & ADM0 & (TPAS + LPAS) & SCG & ACDS

APT is cleared by:

> pon + (read ISR1)

Bit **Mnemonic** **Description**

Notes

ADM1:	Address Mode Register Bit 1, ADMR[1]w
ADM0:	Address Mode Register Bit 0, ADMR[0]w
TPAS:	GPIB Talker Primary Addressed State
LPAS:	GPIB Listener Primary Addressed State
SCG:	GPIB Secondary Command Group
ACDS:	GPIB Accept Data State
pon:	Power On Reset
Read ISR1:	Bit is cleared immediately after it is read

The APT bit indicates that a secondary GPIB address has been received and is available in the CPTR for inspection.

Note: The application program must check this bit when using TLC address mode 3.

When APT is set, the Data Accepted (DAC) message is held and the GPIB Handshake stops until either the Valid or Non-Valid auxiliary command is issued. The secondary address can be read from the CPTR.

5r	DET	Device Execute Trigger Bit
5w	DET IE	Device Execute Trigger Interrupt Enable Bit

DET is set by:

DTAS

DET is cleared by:

pon + (read ISR1)

Notes

DTAS:	GPIB Device Trigger Active State
pon:	Power On Reset
read ISR1:	Bit is cleared immediately after it is read

The DET bit indicates that the GPIB Device Execute Trigger (DET) command has been received while the TLC was a GPIB Listener (the TLC has been in DTAS).

4r	END RX	End Received Bit
4w	END IE	End Received Interrupt Enable Bit

END RX is set by:

LACS & (EOI + EOS & REOS) & ACDS

Bit **Mnemonic** **Description**

END RX is cleared by:

pon + (read ISR1)

Notes

LACS:	GPIB Listener Active State
EOI:	GPIB End Or Identify Signal
EOS:	GPIB END Of String message
ACDS:	GPIB Accept Data State
pon:	Power On Reset
read ISR1:	Bit is cleared immediately after it is read

The END RX bit is set when the TLC is a Listener and the GPIB uniline message, END, is received with a data byte from the GPIB Talker, or the data byte in the DIR matches the contents of the End Of String Register (EOSR).

Bit	Mnemonic	Description
3r	DEC	Device Clear Bit
3w	DEC IE	Device Clear Interrupt Enable Bit

DEC is set by:

DCAS

DEC is cleared by:

pon + (read ISR1)

Notes

DCAS:	GPIB Device Clear Active State
pon:	Power On Reset
read ISR1:	Bit is cleared immediately after it is read

The DEC bit indicates that the GPIB Device Clear (DCL) command has been received or that the GPIB Selected Device Clear (SDC) command has been received while the TLC was a GPIB Listener (the TLC is in DCAS).

Bit	Mnemonic	Description
2r	ERR	Error Bit
2w	ERR IE	Error Interrupt Enable Bit

ERR is set by:

TACS & SDYS & DAC & RFD + SIDS & (write CDOR) + (SDYS to SIDS)

ERR is cleared by:

pon + (read ISR1)

Bit	**Mnemonic**	**Description**

Notes

TACS:	GPIB Talker Active State
SDYS:	GPIB Source Delay State
DAC:	GPIB Data Accepted message
RFD:	GPIB Ready For Data message
SIDS:	GPIB Source Idle State
write CDOR:	Bit is set immediately after writing to the Command/Data Out Register
SDYS to SIDS:	Transition from GPIB Source Delay State to Source Idle State
pon:	Power On Reset
read ISR1;	Bit is cleared immediately after it is read

The ERR bit indicates that the contents of the CDOR have been lost. ERR is set when data is sent over the GPIB without a specified Listener or when a byte is written to the CDOR during SIDS or during the SDYS to SIDS transition.

1r	DO	Data Out Bit
1w	DO IE	Data Out Interrupt Enable Bit

DO is set as:

(TACS & SGNS) becomes true

DO is cleared by:

(read ISR1) + TACS* + SGNS*

Notes

TACS:	GPIB Talker Active State
SGNS:	GPIB Source Generate State
read ISR1:	Bit is cleared immediately after it is read

The DO bit indicates that the TLC is ready to accept another data byte from the SBX bus for transmission onto the GPIB when the TLC is the GPIB Talker. The DO bit is cleared when a byte is written to the CDOR and also when the TLC ceases to be the Active Talker.

Bit	Mnemonic	Description
0r	DI	Data In Bit
0w	DI IE	Data In Interrupt Enable Bit Bit

DI is set by:

LACS & ACDS & (Continuous mode)

DI is cleared by:

pon + (read ISR1) + (Finish Handshake) & (Holdoff mode)
+ (read DIR)

Notes

LACS:	GPIB Listener Active State
ACDS:	GPIB Accept Data State
Continuous mode:	Listen in continuous mode auxiliary command in effect
pon:	Power On Reset
read ISR1:	Bit is cleared immediately after it is read
Finish Handshake:	Finish Handshake auxiliary command issued
Holdoff mode:	RFD Holdoff state
read DIR:	Read Data In Register

The DI bit indicates that the TLC, as a GPIB Listener, has accepted a data byte from the GPIB Talker.

Interrupt Status Register 2 (ISR2)

I/O Port Offset: + 2

Attributes: Read Only,
Bits are cleared when read

Interrupt Mask Register 2 (IMR2)

I/O Port Offset: + 2

Attributes: Write Only

7	6	5	4	3	2	1	0	
INT	SRQI	LOK	REM	CO	LOKC	REMC	ADSC	R
0	SRQI IE	DMAO	DMAI	CO IE	LOKC IE	REMC IE	ADSC IE	W

The Interrupt Status Register 2 (ISR2) consists of six Interrupt Status bits and two TLC Internal Status bits. The Interrupt Mask Register 2 (IMR2) consists of five Interrupt Enable bits and two TLC Internal Control bits. If the Interrupt Enable bit is true when the corresponding status condition or event occurs, an interrupt request is generated. Bits in ISR2 are set and cleared regardless of the status of the bits in IMR2. If a condition occurs that requires the TLC to set or clear a bit or bits in ISR2 at the same time ISR2 is being read, the TLC holds off setting or clearing the bit or bits until the read is finished.

Bit **Mnemonic** **Description**

7r INT Interrupt Bit

This bit is the logical OR of all the Enabled Interrupt Status bits in both ISR1 and ISR2, each one ANDed with its Interrupt Enable bit. There is no corresponding Mask bit for INT. If INT=1, the INT output pin of the TLC, signal GPIB IR*, is asserted.

Note: Program the INT output pin of the TLC to be active low; see description of AUXRB.

INT is set by:

(CPT & CPT IE) + (APT & APT IE) + (DET & DET IE) + ERR & ERR IE) + (END RX & END IE) + (DEC & DEC IE) + (DO & DO IE) + (DI & DI IE) + (SRQI & SRQI IE) + (REMC & REMC IE) + (CO & CO IE) + (LOKC & LOKC IE) + (ADSC & ADSC IE)

7w 0 Unused Bit

Write zero to this bit.

Bit **Mnemonic** **Description**

6r SRQI Service Request Input Bit
6w SRQI IE Service Request Input Interrupt Enable Bit

SRQI is set when:

(CIC & SRQ & (RQS & DAV)*) becomes true or
(CIC & SRQ & RQS & DAV) becomes true

Note: The second equation shown above for setting SRQI applies exclusively to situations in which two or more devices are issuing the SRQ message.

SRQI is cleared by:

pon + (read ISR2)

Notes

CIC:	GPIB Controller-In-Charge
SRQ:	GPIB Service Request message
RQS:	GPIB Request Service message
DAV:	GPIB Data Valid message
pon:	Power On Reset
read ISR2:	Bit is cleared immediately after it is read

The SRQI bit indicates that a GPIB Service Request (SRQ) message has been received while the TLC function is active (CIC=1).

5r LOK Lockout Bit

LOK is used, along with the REM bit, to indicate the status of the TLC GPIB Remote/Local (RL) function. If set, the LOK bit indicates that the TLC is in Local With Lockout State (LWLS) or Remote With Lockout State (RWLS). LOK is a Non-Interrupt bit.

5w DMAO DMA Out Enable Bit

4r REM Remote Bit

This bit is true when the TLC GPIB RL function is in one of two states: Remote State (REMS) or Remote With Lockout State (RWLS). The TLC RL function transfers to one of these states when the System Controller has asserted the Remote Enable line (REN), and the CIC addresses the TLC as a Listener.

4w DMAI DMA Input Enable Bit

Bit	**Mnemonic**	**Description**
3r	CO	Command Out Bit
3w	CO IE	Command Out Interrupt Enable Bit

CO is set when:

(CACS & SGNS) becomes true

CO is cleared by:

(read ISR2) + CACS* + SGNS*

Notes

CACS:	GPIB Controller Active State
SGNS:	GPIB Source Generate State
read ISR2:	Bit is cleared immediately after it is read

CO = 1 indicates that the CDOR is empty and that another command can be written to it for transmission over the GPIB without overwriting a previous command.

2r	LOKC	Lockout Change Bit
2w	LOKC IE	Lockout Change Interrupt Enable Bit

LOKC is set by:

any change in LOK

LOKC is cleared by:

pon + (read ISR2)

Notes

LOK:	ISR2[5]r
pon:	Power On Reset
read ISR2:	Bit is cleared immediately after it is read

LOKC is set when there is a change in the LOK bit, ISR2[5]r, (REMS + RWLS).

1r	REMC	Remote Change Bit
1w	REMC IE	Remote Change Interrupt Enable Bit

REMC is set by:

any change in REM

REMC is cleared by:

pon + (read ISR2)

Bit **Mnemonic** **Description**

Notes

REM:	ISR2[4]r
pon:	Power On Reset
read ISR2:	Bit is cleared immediately after it is read

REMC is set when there is a change in the REM bit, ISR2[4]r, (REMS + RELS).

Bit	Mnemonic	Description
0r	ADSC	Addressed Status Change Bit
0w	ADSC IE	Addressed Status Change Interrupt Enable Bit

ADSC is set by:

> [(any change in TA) + (any change in LA) + (any change in CIC) + (any change in MJMN)] & (lon + ton)*

ADSC is cleared by:

> pon + (read ISR2)

Notes

TA:	Talker Active bit, ADSR[1]r
LA:	Listener Active bit, ADSR[2]r
CIC:	Controller-In-Charge bit, ADSR[7]r
MJMN:	Major/Minor bit, ADSR[0]r
lon:	Listen Only bit, ADMR[6]w
ton:	Talk Only bit, ADMR[7]w
pon:	Power On Reset
read ISR2:	Bit is cleared immediately after it is read
ADSR:	Address Status Register
ADMR:	Address Mode Register

ADSC is set when there is a change in one of the four bits TA, LA, CIC, MJMN of the Address Status Register (ADSR).

Serial Poll Status Register (SPSR)

I/O Port Offset: + 3

Attributes: Read Only

Serial Poll Mode Register (SPMR)

I/O Port Offset: + 3

Attributes: Write Only

7	6	5	4	3	2	1	0	R
S8	PEND	S6	S5	S4	S3	S2	S1	
S8	rsv	S6	S5	S4	S3	S2	S1	
								W

Bit	Mnemonic	Description
7r 7w,	S8	Serial Poll Status Bit 8
5-0r, 5-0w	S[6-1]	Serial Poll Status Bits 6 through 1 Cleared by Power On Reset (pon), or by issuing the Chip Reset auxiliary command. These bits are used for sending device- or system-dependent status information over the GPIB when the TLC is serially polled. When the TLC is addressed as the GPIB Talker and receives the GPIB multiline Serial Poll Enable (SPE) command message, it transmits a byte of status information, SPMR[7- 0], to the Controller-In-Charge after the Controller Goes to Standby and becomes an Active Listener.
6r	PEND	Pending Bit PEND is set when rsv=1 and cleared when (Negative Poll Response State [NPRS] & Request Service [rsv]) = 1. Reading the PEND status bit can confirm that a request was accepted and that the Status Byte (STB) was transmitted (PEND=0).
6w	rsv	Request Service Bit The rsv bit is used for generating the GPIB local rsv message. When rsv is set and the GPIB Active Controller is not serially polling the TLC, the TLC enters the Service Request State (SRQS) and asserts the GPIB SRQ signal. When the Active Controller reads the STB during the poll, the TLC clears rsv at the Affirmative Poll Response State (APRS). The rsv bit is also cleared by pon, and by issuing the Chip Reset auxiliary command.

Address Status Register (ADSR)

I/O Port Offset: + 4

Attributes: Read Only

7	6	5	4	3	2	1	0 R
CIC	ATN*	SPMS	LPAS	TPAS	LA	TA	MJMN

The Address Status Register (ADSR) contains information that can be used to monitor the TLC GPIB address status.

Bit **Mnemonic** **Description**

7r CIC Controller-In-Charge Bit

CIC = -(CIDS + CADS)

CIC indicates that the TLC GPIB Controller function is in an active or standby state, with ATN* on or off, respectively. The Controller function is in an idle state, with ATN* off if CIC=0.

6r ATN* Attention* Bit

ATN* is a Status bit that indicates the current level of the GPIB ATN* signal. If ATN* = 0, the GPIB ATN* signal is asserted.

5r SPMS Serial Poll Mode State Bit

If SPMS=1, the TLC GPIB Talker (T) or Talker Extended (TE) function is enabled to participate in a Serial Poll. SPMS is set when the TLC has been addressed as a GPIB Talker and the GPIB Active Controller has issued the GPIB Serial Poll Enable (SPE) command message. SPMS is cleared when the GPIB SPD (Serial Poll Disable) command is received (by pon or by issuing the Chip Reset auxiliary command).

4r LPAS Listener Primary Addressed State Bit

The LPAS bit is used when the TLC is configured for extended GPIB addressing and, when set, indicates that the TLC has received its primary listen address. In mode 3 addressing (see *Address Mode Register* in this chapter), LPAS=1 indicates that the secondary address being received on the next GPIB command may represent the TLC extended (secondary) GPIB listen address. LPAS is cleared by pon, or by issuing the Chip Reset auxiliary command.

Bit	**Mnemonic**	**Description**
3r	TPAS	Talker Primary Addressed State Bit TPAS is used when the TLC is configured for extended GPIB addressing, and, when set, indicates that the TLC has received its primary GPIB talk address. In mode 3 addressing, TPAS=1 indicates that the secondary address being received as the next GPIB command message can represent the TLC extended (secondary) GPIB talk address.
2r	LA	Listener Active Bit LA is set when the TLC has been addressed or programmed as a GPIB Listener; that is, the TLC is in the Listener Active State (LACS) or the Listener Addressed State (LADS). The TLC can be addressed to listen either by sending its own listen or extended listen address while it is CIC, or by receiving its listen address from another CIC. It can also be programmed to listen using the Listen Only (lon) bit in the Address Mode Register (ADMR). If the TLC is addressed to listen, it is automatically unaddressed to talk. LA is cleared by pon, or by issuing the Chip Reset auxiliary command.
1r	TA	Talker Active Bit TA is set when the TLC has been addressed or programmed as the GPIB Talker; that is, the TLC is in the Talker Active State (TACS), the Talker Addressed State (TADS), or the Serial Poll Active State (SPAS). The TLC can be addressed to talk either by sending its own talk or extended talk address while it is CIC or by receiving its talk address from another CIC. It can also be programmed to talk using the Talk Only (ton) bit in the ADMR. If the TLC is addressed to talk, it is automatically unaddressed to listen. TA is cleared by pon, or by issuing the Chip Reset auxiliary command.
0r	MJMN	Major-Minor Bit The MJMN bit is used to determine whether the information in the other ADSR bits applies to the TLC major or minor Talker and Listener functions. MJMN is set to one when the TLC GPIB minor talk address or minor listen address is received. MJMN is cleared on receipt of the TLC major talk or major listen address. Notice that only one Talker and Listener can be active at any one time; thus, the MJMN bit indicates which, if either, of the TLC Talker and Listener functions is addressed or active. MJMN is always zero unless a dual primary addressing mode (mode 1 or mode 3) is enabled (see *Address Mode Register* in this chapter). The MJMN bit is cleared by pon or by issuing the Chip Reset auxiliary command.

Address Mode Register (ADMR)

I/O Port Offset: + 4

Attributes: Write Only

7	6	5	4	3	2	1	0
ton	lon	TRM1	TRM0	0	0	ADM1	ADM0

W

Bit **Mnemonic** **Description**

7w ton Talk Only Bit

By setting ton programs, the TLC becomes a GPIB Talker. If ton is set, the lon, ADM1, and ADM0 bits should be cleared. This method should be used in place of the addressing method when the TLC will be only a Talker.

Note: Clearing ton does not by itself take the TLC out of GPIB Talker Active State (TACS). It is also necessary to execute the Chip Reset, Immediate Execute pon auxiliary command, or receive another Talk Address when either ADM1 or ADM0 is set.

6w lon Listen Only Bit

By setting lon programs, the TLC becomes a GPIB Listener. If lon is set, ton, ADM1, and ADM0 should be cleared. This method should be used in place of the addressing method when the TLC will be only a Listener.

Note: Clearing lon does not, by itself, take the TLC out of GPIB Listener Active State (LACS). It is also necessary to execute the Chip Reset or Immediate Execute pon auxiliary command.

5-4w TRM[1-0] Transmit/Receive Mode Bit

TRM1 and TRM0 control the function of the TLC T/R2 and T/R3 output pins in the following manner:

TRM1	TRM0	T/R2	T/R3
0	0	EOI OE	TRIG
0	1	CIC	TRIG
1	0	CIC	EOI OE
1	1	CIC	PE

Bit **Mnemonic** **Description**

Key

EOI OE	=	GPIB EOI signal output enable
CIC	=	Controller-In-Charge
TRIG	=	Trigger
PE	=	Pull-up Enable

For proper operation, set both TRM1 and TRM0 (which selects both T/R2 = CIC and T/R3 = PE).

3-2w 0 Unused Bits

Write zeros to these bits.

1-0w ADM[1-0] Address Mode Bits 1 through 0

These bits specify the addressing mode that is in effect, that is, the manner in which the information in ADR0 and ADR1 is interpreted (see *Address Register 0* and *Address Register 1* later in this chapter). If both bits are zero, then the TLC does not respond to GPIB address commands; instead, the ton and lon bits are used to program the Talker and Listener functions, respectively. The ton and lon bits must be cleared if mode 1, 2, or 3 addressing is selected, and the ADM[1-0] bits must be cleared if either of the bits ton or lon are set.

Mode	ADM1	ADM0	Title
0	0	0	ton/lon
1	0	1	Normal dual addressing
2	1	0	Extended single addressing
3	1	1	Extended dual addressing

In mode 1, ADR0 and ADR1 contain the major and minor addresses, respectively, for dual primary GPIB address applications; that is, the TLC responds to two GPIB addresses, a major one and a minor one. The MJMN bit in the ADSR indicates which address was received. In applications where the TLC needs to respond to only one address, the major Talker and Listener function is used and the minor Talker and Listener function should be disabled by setting the Disable Talker (DT) and Disable Listener (DL) bits in the Address Register (ADR).

In mode 2, the TLC recognizes two sequential GPIB address bytes, a primary followed by a secondary. Both GPIB address bytes must be received in order to enable the TLC to talk or listen. In this manner, mode 2 addressing implements the Extended Talker and Extended Listener functions as defined in IEEE-488, without requiring computer program intervention. In mode 2, ADR0 and ADR1 contain the TLC primary and secondary GPIB addresses, respectively.

Bit	Mnemonic	Description
		In mode 3, the TLC handles addressing just as it does in mode 1, except that each major or minor GPIB primary address must be followed by a secondary address. All secondary GPIB addresses must be verified by computer program when mode 3 is used. When the TLC is in Talker Primary Addressed State (TPAS) or Listener Primary Addressed State (LPAS) and a secondary address byte is on the GPIB DIO lines, the APT bit of ISR2 is set and the secondary GPIB address may be inspected in the CPTR. The TLC Acceptor Handshake is held up in the Accept Data State (ACDS) until the Valid or Non-Valid auxiliary command is written to the AUXMR, signaling a valid or invalid secondary address, respectively, to the TLC.

Command Pass Through Register (CPTR)

I/O Port Offset: + 5

Attributes: Read Only

7	6	5	4	3	2	1	0	R
CPT7	CPT6	CPT5	CPT4	CPT3	CPT2	CPT1	CPT0	

Bit	Mnemonic	Description
7-0r	CPT[7-0]	Command Pass Through Bits 7 through 0 These bits are used to transfer undefined multiline GPIB command messages from the GPIB DIO lines to the computer. When the CPT feature is enabled (CPT ENABLE=1, AUXRB[0]w), any GPIB Primary Command Group (PCG) message not decoded by the TLC is treated as an undefined command. All GPIB Secondary Command Group (SCG) messages following an undefined GPIB PCG message are also treated as undefined. In such a case, when an undefined GPIB message is encountered, it is held in the CPTR and the TLC Acceptor Handshake function is held off (in ACDS) until the Valid auxiliary command is written to the AUXMR. The CPTR is also used to inspect secondary addresses when mode 3 addressing is used. The TLC Acceptor Handshake function is held off (in ACDS) until the Valid or Non-Valid auxiliary command is written to the AUXMR. The CPTR is read during a TLC-initiated Parallel Poll operation to fetch the Parallel Poll response. The PPR message is latched into the CPTR when CPPS is set, and held valid until CIDS is set, or until a command byte is sent over the GPIB.

Auxiliary Mode Register (AUXMR)

I/O Port Offset: + 5

Attributes: Write Only,
Permits Access to Hidden Registers

7	6	5	4	3	2	1	0
CNT2	CNT1	CNT0	COM4	COM3	COM2	COM1	COM0
							W

The AUXMR is used to issue auxiliary commands. It is also used to program the five hidden registers:

- Internal Counter Register (ICR)
- Parallel Poll Register (PPR)
- Auxiliary Register A (AUXRA)
- Auxiliary Register B (AUXRB)
- Auxiliary Register E (AUXRE)

Table 4-2 shows the control and command codes implemented.

Bit **Mnemonic** **Description**

7-5w CNT[2-0] Control Code Bits 2 through 0

These bits specify the control code, that is, the manner in which the information in bits COM[4-0] is to be used. If CNT[2-0] are all zero, the special command selected by COM[4-0] is executed. Otherwise, the hidden register selected by CNT[2-0] is loaded with the data from COM[4-0].

4-0w COM[4-0] Command Code Bits 4 through 0

These bits specify the command code of the special function if the control code is 000. Table 4-2 is a summary of the implemented special functions. Table 4-3 explains the details of each special function. If the control code is not 000, these bits are written to one of the hidden registers (indicated by the control code in CNT[2-0]).

Table 4-2. Auxiliary Command Summary

Function Code* **(COM4-COM0)** **4 3 2 1 0**	**Hex** **Code****	**Auxiliary Command**
0 0 0 0 0	00	Immediate Execute pon
0 0 0 1 0	02	Chip Reset
0 0 0 1 1	03	Finish Handshake
0 0 1 0 0	04	Trigger
0 0 1 0 1	05	Return to Local
0 0 1 1 0	06	Send EOI
0 0 1 1 1	07	Non-Valid Secondary Command or Address
0 1 1 1 1	0F	Valid Secondary Command or Address
0 0 0 0 1	01	Clear Parallel Poll Flag
0 1 0 0 1	09	Set Parallel Poll Flag
1 0 0 0 1	11	Take Control Asynchronously (Pulsed)
1 0 0 1 0	12	Take Control Synchronously
1 1 0 1 0	1A	Take Control Synchronously on End
1 0 0 0 0	10	Go To Standby
1 0 0 1 1	13	Listen
1 1 0 1 1	1B	Listen in Continuous Mode
1 1 1 0 0	1C	Local Unlisten
1 1 1 0 1	1D	Execute Parallel Poll
1 1 1 1 0	1E	Set IFC
1 0 1 1 0	16	Clear IFC
1 1 1 1 1	1F	Set REN
1 0 1 1 1	17	Clear REN
1 0 1 0 0	14	Disable System Control

* CNT[2-0] set to 000 binary
** Represents all eight bits of the Auxiliary Mode Register

Table 4-3 shows the functions that are executed when the AUXMR Control Code (CNT[2-0]) is loaded with 000 (binary) and the Command Code (COM[4-0]) is loaded.

Table 4-3. Auxiliary Commands: Detail Description

<table>
<tr><th>Command Code
(COM4-COM0)
4 3 2 1 0</th><th>Description</th></tr>
<tr><td>0 0 0 0 0</td><td>Immediate Execute pon
This command generates a local pon message that places the following GPIB interface functions into these idle states:
<table>
<tr><td>AIDS</td><td>Acceptor Idle State</td></tr>
<tr><td>CIDS</td><td>Controller Idle State</td></tr>
<tr><td>LIDS</td><td>Listener Idle State</td></tr>
<tr><td>LOCS</td><td>Local State</td></tr>
<tr><td>LPIS</td><td>Listener Primary Idle State</td></tr>
<tr><td>NPRS</td><td>Negative Poll Response State</td></tr>
<tr><td>PPIS</td><td>Parallel Poll Idle State</td></tr>
<tr><td>PUCS</td><td>Parallel Poll to Unaddressed to Configure State</td></tr>
<tr><td>SIDS</td><td>Source Idle State</td></tr>
<tr><td>SIIS</td><td>System Control Interface Clear Idle State</td></tr>
<tr><td>SPIS</td><td>Serial Poll Idle State</td></tr>
<tr><td>SRIS</td><td>System Control Remote Enable Idle State</td></tr>
<tr><td>TIDS</td><td>Talker Idle State</td></tr>
<tr><td>TPIS</td><td>Talker Primary Idle State</td></tr>
</table>
If the command is sent while a pon message is already active (by either an external reset pulse or the Chip Reset auxiliary command) the local pon message becomes false.</td></tr>
<tr><td>0 0 0 1 0</td><td>Chip Reset
The Chip Reset command resets the TLC in the same way as an external reset pulse. The System Controller bit is also cleared. The TLC is reset to the following conditions:
• The local pon message is set and the interface functions are placed in their idle states.
• All bits of the SPMR are cleared.
• The EOI bit is cleared.
• All bits of the AUXRA, AUXRB, and AUXRE are cleared.
• The Parallel Poll Flag and RSC local message are cleared.
• The contents of the ICR are set to 1000 (binary).
• The TRM0 bit and the TRM1 bit are cleared.
Once a Chip Reset command is executed, an Immediate Execute pon command must be executed before the TLC can be programmed or addressed.</td></tr>
</table>

(continues)

Table 4-3. Auxiliary Commands: Detail Description (continued)

Command Code (COM4-COM0) **4 3 2 1 0**	**Description**
0 0 0 1 1	Finish Handshake (FH) The Finish Handshake command finishes a GPIB Handshake that was stopped because of a Holdoff on RFD or DAC.
0 0 1 0 0	Trigger **Note:** Trigger is not implemented on the GPIB-SBX. The Trigger command generates a high pulse on the TRIG pin (T/R3 pin when TRM1=0) of the TLC. The Trigger command performs the same function as if the DET (Device Trigger) bit (ISR1[5]r) were set. (The DET bit is not set by issuing the Trigger command.)
0 0 1 0 1 0 1 1 0 1	Return to Local (rtl) Return to Local (rtl) The two Return to Local commands implement the rtl message as defined by IEEE-488. When COM3 is zero, the message is generated in the form of a pulse. When COM3 is one, the rtl command is set.
0 0 1 1 0	Send EOI (SEOI) The Send EOI command causes the GPIB End Or Identify (EOI) line to go true with the next byte transmitted. The EOI line is then cleared upon completion of the Handshake for that byte. The TLC recognizes the Send EOI command only if TA=1 (that is, the TLC is addressed as the GPIB Talker).
0 0 1 1 1	Non-Valid Secondary Command or Address The Non-Valid command releases the GPIB DAC message held off by the Address Pass Through (APT). The TLC is permitted to operate as if an Other Secondary Address (OSA) message has been received.
0 1 1 1 1	Valid Secondary Command or Address The Valid command releases the GPIB DAC message held off by APT and allows the TLC to function as if a My Secondary Address (MSA) message had been received. The DAC message is released at the time of Command Pass Through (CPT). DAC is also released if DCAS or DTAS is in Holdoff state.

(continues)

Table 4-3. Auxiliary Commands: Detail Description (continued)

Command Code (COM4-COM0) 4 3 2 1 0	**Description**
0 0 0 0 1 0 1 0 0 1	Clear Parallel Poll Flag Set Parallel Poll Flag These commands set the Parallel Poll Flag to the value of COM3. The value of the Parallel Poll Flag is used as the local message ist when the ISS bit of Auxiliary Register B is zero. The value of SRQS is used as the ist when ISS=1.
1 0 0 0 0 TLC	Go To Standby The Go To Standby command sets the local message gts if the is in Controller Active State (CACS) or when it enters CACS. When the TLC leaves CACS, gts is cleared.
1 0 0 0 1	Take Control Asynchronously The Take Control Asynchronously command pulses the local message tca.
1 0 0 1 0	Take Control Synchronously The Take Control Synchronously command sets the local message tcs. The local message tcs is effective only when the TLC is in Controller Standby State (CSBS) or Controller Synchronous Wait State (CSWS). The local message tcs is cleared when the TLC enters Controller Active State (CACS).
1 1 0 1 0	Take Control Synchronously on END The Take Control Synchronously on END command sets the local message tcs when the data block transfer End message (END bit equal to one) is generated at CSBS. The tcs message is cleared when the TLC enters CACS.
1 0 0 1 1	Listen The listen command generates the local message ltn in the form of a pulse. It should be issued when the controller is in CACS.

(continues)

Table 4-3. Auxiliary Commands: Detail Description (continued)

Command Code (COM4-COM0) 4 3 2 1 0	Description
1 1 0 1 1	Listen in Continuous Mode The Listen in Continuous Mode command generates the local message ltn in the form of a pulse and places the TLC in continuous mode. In continuous mode, the local message rdy is issued when the Acceptor Not Ready State (ANRS) is initiated unless data block transfer end is detected (END bit is one). When END is detected, the TLC is placed in the RFD Holdoff state, preventing generation of the rdy message. In continuous mode, the DI bit is not set when a data byte is received. The continuous mode caused by the Listen in Continuous Mode command is released when the Listen auxiliary command is issued or the TLC enters the Listener Idle State (LIDS).
1 1 1 0 0	Local Unlisten The Local Unlisten command generates the local message lun in the form of a pulse. It should be issued when the controller is in CACS.
1 1 1 0 1	Execute Parallel Poll The Execute Parallel Poll command sets the local message Request Parallel Poll (rpp). The rpp message is cleared when the TLC enters either Controller Parallel Poll State (CPPS) or Controller Idle State (CIDS). The transition of the TLC Controller interface function is not guaranteed if the local messages rpp and Go To Standby (gts) are issued simultaneously when the TLC is in Controller Active State (CACS) and Source Transfer State (STRS) or Source Delay State (SDYS).
1 1 1 1 0 1 0 1 1 0	Set IFC Clear IFC These commands generate the local message Request System Control (rsc) and set Interface Clear (IFC) to the value of COM3. In order to meet IEEE-488 requirements, you must not issue the Clear IFC command until IFC has been held true for at least 100 µsec.

(continues)

Table 4-3. Auxiliary Commands: Detail Description (continued)

Command Code **(COM4-COM0)** **4 3 2 1 0**	**Description**
1 1 1 1 1 1 0 1 1 1	Set REN Clear REN These commands generate the local message rsc and set REN to the value in COM3. In order to meet IEEE-488 requirements, you must not issue the Set REN command until REN has been held false for at least 100 µsec.
1 0 1 0 0	Disable System Control The Disable System Control command clears the local message rsc.

Hidden Registers

The hidden registers are loaded through the Auxiliary Mode Register (AUXMR). AUXMR[7-5] is loaded with the hidden register number, and AUXMR[4-0] is loaded with the data to be transferred to the hidden register. The hidden registers cannot be read, and in some cases the contents are setable only; that is, they can be cleared or reset to initialized conditions only by issuing the Chip Reset auxiliary command, or by a pon. Figure 4-2 shows the five hidden registers and illustrates how they are loaded with data from the AUXMR.

Internal Counter Register (ICR)

I/O Port Offset: + 5

AUXMR Control Code: 001 (Binary, Bits 7 - 5)

Attributes: Write Only,
Accessed through AUXMR

4	3	2	1	0
0	CLK3	CLK2	CLK1	CLK0

W

Bit **Mnemonic** **Description**

4w 0 Unused Bit

Write zero to this bit.

3-0w CLK[3-0] Clock Bits 3 through 0

The contents of the ICR are used to divide internal counters that generate TLC state change delay times that are used by the IEEE-488 specification. The most familiar of these delay times, T1, is the minimum delay between placing the data or command bytes on the GPIB DIO lines and asserting DAV. These delay times vary depending upon the type of transfer in progress and the value of the AUXRB bit TRI.

For proper operation in the GPIB-SBX, ICR should be set to five because the TLC is clocked at 5 MHz.

Parallel Poll Register (PPR)

I/O Port Offset: + 5

AUXMR Control Code: 011 (Binary, Bits 7 - 5)

Attributes: Write Only,
Internal to TLC

4	3	2	1	0
U	S	P3	P2	P1
				W

Writing to the Parallel Poll Register (PPR) is done via the AUXMR. Writing the binary value 011 into the Control Code (CNT[2-0]) and writing a bit pattern into the command code portion (COM[4-0]) of the AUXMR causes the command code to be written to the PPR. When COM[4-0] is written to the PPR, the bits are named as shown in the PPR. This 5-bit command code determines the manner in which the TLC responds to a Parallel Poll.

When using the remote Parallel Poll Configure (IEEE-488 capability code PP1), do not write to the PPR. The TLC implements remote configuration fully and automatically without software assistance. The hardware recognizes, interprets, and responds to Parallel Poll Configure (PPC), Parallel Poll Enable (PPE), Parallel Poll Disable (PPD), and Identify (IDY) messages. The user need only set or clear the individual status (ist) message (using Set/Clear Parallel Poll Flag auxiliary commands) according to pre-established system protocol convention.

When using the local PPC (capability code PP2), a valid PPE or PPD message should be written to the PPR in advance of the poll.

Bit **Mnemonic** **Description**

4w U Unconfigure Bit

The U bit determines whether or not the TLC participates in a Parallel Poll. If U=0, the TLC participates in Parallel Polls and responds in the manner defined by PPR[3] through PPR[0] and by ist. If U=1, the TLC does not participate in a Parallel Poll.

The U bit is equivalent to the local message lpe* (Local Poll Enable, active low). When U=0, S and P[3-1] mean the same as the bit of the same name in the PPE message, and the I/O write operation (to the PPR) is the same as the receipt of the PPE message from the GPIB Controller. When U=1, S and P[3-1] do not carry any meaning, but they should be cleared.

Bit **Mnemonic** **Description**

3w S Status Bit Polarity (Sense) Bit

The S bit is used to indicate the polarity (or sense) of the TLC local ist message. If S=1, the status is *in phase*, meaning that if, during a Parallel Poll response, S=ist=1, and U=0, the TLC responds to the Parallel Poll by driving one of the eight GPIB DIO lines low, thus asserting it to a logic one. If S=1 and ist=0, the TLC does not drive the DIO line.

If S=0, the status is *in reverse phase*, meaning that if, during a Parallel Poll, ist=0, and U=0, the TLC responds to the Parallel Poll by driving one of the eight GPIB DIO lines low. If S=0 and ist=1, the TLC does not drive the DIO line.

Refer to the *Auxiliary Register B* and *Clear Parallel Poll Flags/Set Parallel Poll Flags* sections in this chapter for more information.

2-0w P[3-1] Parallel Poll Response Bits 3 through 1

PPR bits 3 through 1, designated P[3-1], contain an encoded version of the Parallel Poll response. P[3-1] indicate which of the eight DIO lines is asserted during a Parallel Poll (equal to N-1). The GPIB-SBX normally drives the GPIB DIO lines using three-state drivers. During Parallel Poll responses, however, the drivers automatically convert to Open Collector mode, as required by IEEE-488. For example, if P[3-1]=010 (binary), GPIB DIO line DIO3* is driven low (asserted) if the GPIB-SBX is Parallel Polled (and S=ist).

Table 4-4 contains some examples of configuring the Parallel Poll Register.

Table 4-4. Examples for Configuring the PPR

Written to the AUXMR								
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**Result**
0	1	1	1	0	0	0	0	Unconfigures PPR
0	1	1	0	0	0	0	0	0 0 0 0 0 is written to the PPR. The GPIB-SBX participates in a Parallel Poll, asserting the DIO1 line if ist=0. Otherwise, the GPIB-SBX does not participate.
0	1	1	0	1	0	0	1	0 1 0 0 1 is written to the PPR. The GPIB-SBX participates in a Parallel Poll, asserting the DIO2 line if ist=1. Otherwise, the GPIB-SBX does not participate.

Auxiliary Register A (AUXRA)

I/O Port Offset: + 5

AUXMR Control Code: 100 (Binary, Bits 7 - 5)

Attributes: Write Only,
Internal to TLC

4	3	2	1	0
BIN	XEOS	REOS	HLDE	HLDA

W

Writing to Auxiliary Register A (AUXRA) is done via the AUXMR. Writing the binary value 100 into the Control Code (CNT[2-0]) and a bit pattern into the Command Code (COM[4-0]) portion of the AUXMR causes the Command Code to be written to Auxiliary Register A. When the data is written to AUXRA, the bits are denoted by the mnemonics shown in the figure above. This 5-bit code controls the data transfer messages Holdoff and EOS/END.

Bit **Mnemonic** **Description**

4w BIN Binary Bit

The BIN bit selects the length of the EOS message. Setting BIN causes the End Of String Register (EOSR) to be treated as a full 8-bit byte. When BIN=0, the EOSR is treated as a 7-bit register (for ASCII characters) and only a 7-bit comparison is done with the data on the GPIB.

3w XEOS Transmit END with EOS Bit

The XEOS bit permits or prohibits automatic transmission of the GPIB END message at the same time as the EOS message when the TLC is in Talker Active State (TACS). If XEOS is set and the byte in the CDOR matches the contents of the EOSR, the EOI line is sent true along with the data.

2w REOS END on EOS Received Bit

The REOS bit permits or prohibits setting the END bit (ISR1[4]r) at reception of the EOS message when the TLC is in Listener Active State (LACS). If REOS is set and the byte in the DIR matches the byte in the EOSR, the END RX bit (ISR1[4]r) is set.

Bit	**Mnemonic**	**Description**
1-0w	HLDE HLDA	Holdoff on End Bit Holdoff on All Bit

HLDE and HLDA together determine the GPIB data receiving mode. The four possible modes are as follows:

HLDE	HLDA	Data Receiving Mode
0	0	Normal Handshake
0	1	RFD Holdoff on All Data
1	0	RFD Holdoff on END
1	1	Continuous

In Normal Handshake mode, the local message rdy is generated when data is received from the GPIB. When the received data is read from the DIR, rdy is generated in Acceptor Not Ready State (ANRS), the RFD message is transmitted, and the GPIB Handshake continues.

In RFD Holdoff on All Data (HLDA) mode, RFD is not sent true after data is received until the Finish Handshake auxiliary command is issued. Unlike normal Handshake mode, the RFD HLDA mode does not generate the rdy message even if the received data is read through the DIR; that is, the GPIB RFD message is not generated.

In RFD Holdoff on End mode, operation is the same as the RFD HLDA mode, but only when the end of the data block (EOS or END message) is detected; that is, the END message is received or, if REOS is set, the EOS character is received. Handshake Holdoff is released by the Finish Handshake auxiliary command.

In continuous mode, the rdy message is generated when in ANRS until the end of the data block is detected. A Holdoff is generated at the end of a data block. The Finish Handshake auxiliary command must be issued to release the Holdoff. The continuous mode is useful for monitoring the data block transfer without actually participating in the transfer (no data reception). In continuous mode, the DI bit (ISR1[0]r) is not set by the reception of a data byte.

Auxiliary Register B (AUXRB)

I/O Port Offset: + 5

AUXMR Control Code: 101 (Binary, Bits 7 - 5)

Attributes: Write Only,
Internal to TLC

4	3	2	1	0
ISS	INV	TRI	SPEOI	CPT ENABLE
				W

Writing to Auxiliary Register B (AUXRB) is done via the AUXMR. Writing the value 101 into the Control Code (CNT[2-0]) and a bit pattern into the Command Code portion (COM[4-0]) of the AUXMR causes the Command Code to be written to AUXRB. When the data is written to AUXRB, the bits are denoted as shown in the figure above. This 5-bit code affects several interface functions, as described in the following paragraphs.

Bit **Mnemonic** **Description**

4w ISS Individual Status Select Bit

The ISS bit determines the value of the TLC ist. When ISS=1, ist becomes the same value as the TLC Service Request State (SRQS). (The TLC is asserting the GPIB SRQ message when it is in SRQS.) When ISS=0, ist takes on the value of the TLC Parallel Poll Flag. The Parallel Poll Flag is set and cleared using the Set Parallel Poll Flag and Clear Parallel Poll Flag auxiliary commands.

3w INV Invert Bit

The INV bit affects the polarity of the TLC INT pin. Setting INV causes the polarity of the Interrupt (INT) pin on the TLC to be active low. As implemented on the GPIB-SBX, configuring the INT pin to active low results in interrupt request errors. Consequently, INV should always be cleared (0) and should never be set except for diagnostic purposes.

INV = 0 : INT pin is active high

INV = 1 : INT pin is active low

Bit	**Mnemonic**	**Description**
2w	TRI	Three-State Timing Bit The TRI bit determines the TLC GPIB Source Handshake Timing, T1. TRI can be set to enable high-speed data transfers (T1 ≥ 500 nsec) when tri-state GPIB drivers are used. (The GPIB-SBX uses tri-state GPIB drivers except during Parallel Poll responses, in which case the GPIB drivers automatically switch to Open Collector.) Setting TRI enables high-speed timing as T1 of the GPIB Source Handshake after transmission of the first byte. Clearing TRI sets the low-speed timing (T1 ≥ 2 µsec).
1w	SPEOI	Send Serial Poll EOI Bit The SPEOI bit permits or prohibits the transmission of the END message in Serial Poll Active State (SPAS). If SPEOI is set, EOI is sent true when the TLC is in SPAS; otherwise, EOI is sent false in SPAS.
0w	CPT ENABLE	Command Pass Through Enable Bit The CPT ENABLE bit permits or prohibits the detection of undefined GPIB commands and permits or prohibits the setting of the CPT bit (ISR1[7]r) on receipt of an undefined command. When CPT ENABLE is set and an undefined command has been received, the DAC message is held and the Handshake stops until the Valid auxiliary command is issued. The undefined command can be read from the CPTR and processed by the software.

Auxiliary Register E (AUXRE)

I/O Port Offset: + 5

AUXMR Control Code: 110 (Binary, Bits 7 - 5)

Attributes: Write Only,
Internal to TLC

4	3	2	1	0
0	0	0	DHDT	DHDC

W

Writing to Auxiliary Register E (AUXRE) is done via the AUXMR. Writing the binary value 110 into the Control Code (CNT[2-0]) and a bit pattern into the the lower five bits of the AUXMR (COM[4-0]) causes the two lowest order bits to be written to AUXRE. The 2-bit code, DHDC and DHDT, determines how the TLC uses DAC Holdoff.

Bit	**Mnemonic**	**Description**
4-2w	0	Unused Bits Write zeros to these bits.
1w	DHDT	DAC Holdoff on DTAS Bit Setting DHDT enables DAC Holdoff when the TLC enters Device Trigger Active State (DTAS). Clearing DHDT disables DAC Holdoff on DTAS. Issuing the Finish Handshake auxiliary command releases the Holdoff.
0w	DHDC	DAC Holdoff on DCAS Bit Setting DHDC enables DAC Holdoff when the TLC enters Device Clear Active State (DCAS). Clearing DHDC disables DAC Holdoff on DCAS. Issuing the Finish Handshake auxiliary command releases the Holdoff.

Address Register 0 (ADR0)

I/O Port Offset: + 6

Attributes: Read Only
Internal to TLC

7	6	5	4	3	2	1	0 R
X	DT0	DL0	AD5-0	AD4-0	AD3-0	AD2-0	AD1-0

Address Register 0 (ADR0) reflects the internal GPIB address status of the TLC as configured using the ADMR. In addressing mode 2, ADR0 indicates the address and enable bits for the primary GPIB address of the TLC. In dual primary addressing (modes 1 and 3) ADR0 indicates the TLC major primary GPIB address. Refer to the *Address Mode Register* description in this chapter for information on addressing modes.

Bit **Mnemonic** **Description**

7r X Don't Care Bit

Read as zero or one.

6r DT0 Disable Talker 0 Bit

If DT0 is set, it indicates that the mode 2 primary (or mode 1 and 3 major) Talker is not enabled; that is, the TLC does not respond to a GPIB talk address matching AD[5-0—1-0]. If DT0=0, the TLC responds to a GPIB talk address matching bits AD[5-0—1-0].

5r DL0 Disable Listener 0 Bit

If DL0 is set, it indicates that the mode 2 primary (or mode 1 and 3 major) Listener is not enabled; that is, the TLC does not respond to a GPIB listen address matching bits AD[5-0—1-0]. If DL0=0, the TLC responds to a GPIB listen address matching bits AD[5-0—1-0].

4-0r AD[5-0—1-0] TLC GPIB Address Bits 5-0 through 1-0

These are the lower 5 bits of the TLC GPIB primary (or major) address. The primary talk address is formed by adding hex 40 to AD[5-0—1-0], while the listen address is formed by adding hex 20.

Address Register (ADR)

I/O Port Offset: + 6

Attributes: Write Only,
Internal to TLC

7	6	5	4	3	2	1	0
ARS	DT	DL	AD5	AD4	AD3	AD2	AD1
							W

The Address Register (ADR) is used to load the internal registers ADR0 and ADR1. Both ADR0 and ADR1 must be loaded for all addressing modes.

Bit **Mnemonic** **Description**

7w ARS Address Register Select Bit

ARS is zero or one to select whether the seven low-order bits of ADR are to be loaded into internal registers ADR0 or ADR1, respectively.

6w DT Disable Talker Bit

DT should be set if recognition of the GPIB talk address formed from AD5 through AD1 (ADR[4-0]w) is not to enable.

5w DL Disable Listener Bit

DL should be set if recognition of the GPIB listen address formed from AD5 through AD1 (ADR[4-0]w) is not to enable.

4-0w AD[5-1] TLC GPIB Address Bits 5 through 1

These bits specify the five low-order bits of the GPIB address that is to be recognized by the TLC. The corresponding GPIB talk address is formed by adding hex 40 to AD[5-1], while the corresponding GPIB listen address is formed by adding hex 20. The value written to AD[5-1] should not be all ones, since the corresponding talk and listen addresses would conflict with the GPIB Untalk (UNT) and GPIB Unlisten (UNL) commands.

Address Register 1 (ADR1)

I/O Port Offset: + 7

Attributes: Read Only,
Internal to TLC

7	6	5	4	3	2	1	0 R
EOI	DT1	DL1	AD5-1	AD4-1	AD3-1	AD2-1	AD1-1

Address Register 1 (ADR1) indicates the status of the GPIB address and enable bits for the secondary address of the TLC if mode 2 addressing is used, or the minor primary address of the TLC if dual primary addressing is used (modes 1 and 3). If mode 1 addressing is used and only a single primary address is needed, both the talk and listen addresses must disable in this register. If mode 2 addressing is used, the talk and listen disable bits in this register must match those in ADR0.

Bit **Mnemonic** **Description**

7r EOI End or Identify Bit

EOI indicates the value of the GPIB EOI line latched when a data byte is received by the TLC GPIB Acceptor Handshake (AH) function. If EOI=1, the EOI line was asserted with the received byte. EOI is cleared by pon, or by using the Chip Reset auxiliary command. EOI is updated after each byte is received.

6r DT1 Disable Talker 1 Bit

If DT1 is set, the mode 2 secondary (or mode 1 and 3 minor) Talker function is not enabled; that is, the TLC does not respond to a secondary address (or minor primary talk address) formed from bits AD[5-1—1-1]. If DT1 is cleared, the secondary address is checked only if the TLC received its primary talk address (that is, is in TPAS).

5r DL1 Disable Listener 1 Bit

If DL1=1, the mode 2 secondary (or mode 1 and 3 minor) listen function is not enabled; that is, the TLC does not respond to a secondary address (or minor primary listen address) formed from bits AD[5-1—1-1]. If DL1 is cleared and the TLC received its primary listen address (that is, is in LPAS), the secondary address is checked.

Bit	Mnemonic	Description
4-0r	AD[5-1—1-1]	TLC GPIB Address Bits 5-1 through 1-1 These are the lower five bits of the TLC secondary or minor address. The secondary address is formed by adding hex 60 to bits AD[5-1—1-1]. The minor talk address is formed by adding hex 40 to AD[5-1—1-1], while the listen address is formed by adding a hex 20.

End Of String Register (EOSR)

I/O Port Offset: + 7

Attributes: Write Only,
Internal to TLC

7	6	5	4	3	2	1	0
EOS7	EOS6	EOS5	EOS4	EOS3	EOS2	EOS1	EOS0

W

The End Of String Register (EOSR) holds the byte used by the TLC to detect the end of a GPIB data block transfer. A 7- or 8-bit byte (ASCII or binary) can be placed in the EOSR to be used in detecting the end of a block of data. The length of the EOS byte to be used in the comparison is selected by the BIN bit in Auxiliary Register A, AUXRA[4]w.

If the TLC is a Listener and bit REOS of AUXRA is set, the END bit is set in ISR1 whenever the byte in the DIR matches the EOSR. If the TLC is a Talker and the data is being transmitted, and bit XEOS of AUXRA is set, the END message (GPIB EOI* line asserted low) is sent along with the data byte whenever the contents of the CDOR matches the EOSR.

Bit	**Mnemonic**	**Description**
7-0w	EOS[7-0]	End of String Bits 7 through 0

Chapter 5
Programming Considerations

This chapter explains important considerations for programming the GPIB-SBX.

Initialization

On power up (pon), the CPU issues a bus reset by driving the reset drive, which generates the RESET pulse on the GPIB-SBX.

The assertion of the RESET line resets the NEC µPD7210 TLC integrated circuit as follows:

- The local message pon is set and the interface functions are placed in their idle states (SIDS, AIDS, TIDS, SPIS, TPIS, LIDS, LPIS, NPRS, LOCS, PPIS, PUCS, CIDS, SRIS, SIIS).
- All bits of the Serial Poll Mode Register (SPMR) are cleared.
- The End or Identify (EOI) bit is cleared.
- All bits of the Auxiliary Register A (AUXRA), Auxiliary Register B (AUXRB), and Auxiliary Register E (AUXRE) are cleared.
- The Parallel Poll Flag and Request System Control (rsc) local message are cleared.
- The Internal Count Register (ICR) is set to a count of eight.
- The Transmit/Receive Mode 0 (TRM0) and Transmit/Receive Mode 1 (TRM1) bits in the Address Mode Register (ADMR) are cleared.

All other register contents should be considered as undefined while the RESET line is asserted and after the RESET line has been cleared. All Auxiliary Mode Register (AUXMR) commands are cleared and cannot be executed. All other GPIB-SBX registers can be programmed while the GPIB-SBX internal signal pon is set. When pon is released or cleared (by issuing an immediate execute pon auxiliary command to the GPIB-SBX), the interface functions are released from the pon state and the auxiliary commands can be executed.

A typical programming initialization sequence for the GPIB-SBX might include the following steps:

1. Write the Chip Reset command to the AUXMR to place the GPIB-SBX in a known quiescent state.
2. Set or clear the desired Interrupt Enable bits in the Interrupt Mask Register 1 (IMR1) and the Interrupt Mask Register 2 (IMR2).
3. Load the GPIB-SBX primary GPIB address in the Address Register 0 (ADR0) and the Address Register 1 (ADR1).

4. Enable or disable the GPIB Talker and Listener functions and addressing mode using the ADMR.

5. Load the Serial Poll Response in the SPMR.

6. Load the Parallel Poll response in the Parallel Poll Register (PPR) if local configuration is used. If using remote configuration, clear the PPR.

7. Clear Power On (pon) by issuing the Immediate Execute pon auxiliary command.

8. Execute the desired auxiliary commands.

The GPIB-SBX as GPIB Controller

The GPIB-SBX Controller function is generally in one of two modes: idle or in charge. When in charge, the Controller function is either active (asserting Attention (ATN)) or standby (not asserting ATN). The following paragraphs discuss the various transitions between these two modes.

Becoming Controller-In-Charge (CIC) and Active Controller

The GPIB-SBX can become CIC either by being the System Controller and taking control (by issuing the Set IFC auxiliary command) or by being passed control of the GPIB from the current Active Controller.

To take control, if the GPIB-SBX is the System Controller, issue the Set IFC auxiliary command, wait for a minimum of 100 µsec, and then issue the Clear IFC auxiliary command. The ensuing GPIB IFC message initializes the GPIB interface functions of all devices on the bus. As soon as any existing CIC goes to idle (unasserting ATN if it was active) the GPIB-SBX becomes CIC and Active Controller and asserts the GPIB ATN line.

In addition to asserting IFC, the Set IFC auxiliary command also causes the GPIB transceivers for IFC* and REN* to be configured as GPIB line drivers, thus allowing the IFC and REN lines from the GPIB-SBX to be driven onto the GPIB. The transceivers remain configured as drivers until a bus reset occurs or the Disable System Control auxiliary command is issued, which causes the transceivers to be reconfigured to be receivers. If the GPIB-SBX is not the System Controller, the initialization sequence should include issuing the Disable System Control auxiliary command to ensure that the transceivers are configured as receivers.

Another Active Controller can pass control to the GPIB-SBX by sending the GPIB-SBX GPIB talk address (My Talk Address (MTA)) followed by the GPIB Take Control (TCT) message. The GPIB-SBX, upon receiving these two messages (MTA and TCT), automatically becomes CIC when ATN is unasserted. The exact sequence of events is as follows:

1. MTA is received by the GPIB-SBX, causing a transition to Talker Addressed State (TADS); this operation can be transparent to a program. The Talker Active (TA) bit in the Address Status Register (ADSR) is set when the GPIB-SBX receives its GPIB talk address.

2. The GPIB TCT message is received by the GPIB-SBX.

3. The current Active Controller sees the completed Handshake, goes to idle and unasserts ATN.

4. As soon as the ATN line on the GPIB is unasserted, the GPIB-SBX automatically becomes CIC and asserts ATN.

As soon as the GPIB-SBX becomes CIC, the CIC bit in the ADSR and the Command Output (CO) bit in Interrupt Status Register 2 (ISR2) are set. Using these two bits, the program can unambiguously determine that the GPIB-SBX is the GPIB Active Controller and can send remote messages.

Sending Remote Multiline Messages (Commands)

When the GPIB-SBX is Active Controller, commands (interface messages) can be sent by writing to the Control/Data Out Register (CDOR) in response to the CO status bit in ISR2.

The GPIB-SBX can address itself to be both Talker and Listener in address mode 1 or 2; that is, the TLC recognizes its address when it sends or receives it.

Going from Active to Standby Controller

If the GPIB-SBX is GPIB Active Controller, the Controller Standby State (CSBS) is entered upon reception of the Go To Standby auxiliary command. The ATN line is unasserted as soon as the GPIB-SBX enters CSBS. Even though the GPIB-SBX GPIB Controller state machine (functional module within the TLC) is in standby, the CIC bit in the ADSR is still set. Do not issue the Go To Standby auxiliary command unless the CO bit in ISR2 is set.

There are three cases to consider when going to standby:

Case 1. The GPIB-SBX will become the GPIB Talker when ATN is unasserted. To do this, wait for CO to be set, send the MTA, wait for CO to be set again, and then issue the Go To Standby (gts) auxiliary command.

Case 2. The GPIB-SBX will become a GPIB Listener when ATN is unasserted. To do this, wait for CO to be set, issue the My Listen Address (MLA), wait for CO to be set again, and then issue gts.

Case 3. The GPIB-SBX will neither be GPIB Talker nor Listener. In this case, issue the Listen auxiliary command and set the Holdoff on End (HLDE) and Holdoff on All (HLDA) bits in AUXRA before going to standby. Once this mode is enabled, the GPIB-SBX participates in the GPIB handshake without setting the Data In (DI) bit. When Holdoff occurs, the GPIB-SBX can take control synchronously. This means that the Talker must finish its transmission with the END or EOS message. It can then take control synchronously when necessary.

Going from Standby to Active Controller

The manner in which the GPIB-SBX resumes GPIB Active Control depends upon how it went to standby. Consider the three cases:

Case 1. The GPIB-SBX as a Talker takes control upon receipt of the Take Control Asynchronously (tca) auxiliary command. Do not issue tca until there are no more bytes to send and the DO bit is set.

Case 2. The GPIB-SBX as a Listener takes control upon receipt of the Take Control Synchronously (tcs) auxiliary command. If programmed I/O is used, tcs should be issued between seeing a DI status bit and reading the last byte from the Data In Register (DIR).

Case 3. The GPIB-SBX as neither Talker nor Listener takes control synchronously with the tcs auxiliary command after detecting the End Received (END RX) bit set in Interrupt Status Register 1 (ISR1). This indicates that a Holdoff is in progress.

When the tcs auxiliary command is used, the GPIB-SBX takes control of the GPIB only at the end of a data transfer. This implies that one transfer must follow or be in progress when the tcs auxiliary command is issued. If this is not the case, the tca auxiliary command must be used. Of course, the tca auxiliary command may be used in place of the tcs auxiliary command when the possibility of disrupting an in-progress GPIB Handshake (before all GPIB Listeners have accepted the data byte) is acceptable.

In Cases 2 and 3 above, the Enable Interrupt on End Received (END IE) bit in IMR1 can also be set to indicate to the program that the GPIB-SBX (functioning as a GPIB Listener) has received its last byte.

In all cases, a CO status indicates that the GPIB-SBX is now Active Controller.

Going from Active to Idle Controller

Going from Active to Idle GPIB Controller, also known as passing control, requires that the GPIB-SBX be the Active Controller initially (in order to send the necessary GPIB command messages). After the GPIB-SBX has become the GPIB Active Controller, the sequence of events required to pass control are as follows:

1. Write the GPIB talk address of the device being passed control to the CDOR.

2. In response to the next CO status, write the TCT message to the CDOR.

3. As soon as the TCT command message is accepted by all devices on the GPIB, the GPIB-SBX automatically unasserts ATN and the new Controller asserts ATN.

The GPIB-SBX as GPIB Talker and Listener

The GPIB-SBX may be either GPIB Talker or Listener, but not both simultaneously. Either function is deactivated automatically if the other is activated. The Talker Active (TA), Listener Active (LA), and ATN* bits in the ADSR together indicate the specific state of the GPIB-SBX:

ATN*	TA	LA	
0	1	0	Addressed Talker – cannot send data
1	1	0	Active Talker – can send data
0	0	1	Addressed Listener – cannot receive data
1	0	1	Active Listener – can receive data

The status bits Addressed Status Change (ADSC), Command Output (CO), Address Pass Through (APT), Data Out (DO), and Data In (DI) are used to prompt the program (with an interrupt request if enabled) when a change of state occurs.

The following paragraphs discuss several aspects of data transfers.

Programmed Implementation of Talker and Listener

When there is no Controller in the GPIB system, the ton and lon address modes are used to activate the GPIB-SBX GPIB Talker and Listener functions (refer to *Address Mode Register* in Chapter 4). If used, Talker Only (ton) or Listener Only (lon) should be set during GPIB-SBX initialization.

Addressed Implementation of Talker and Listener

When the GPIB-SBX is the GPIB Active Controller, it can address itself to talk or listen by sending its own GPIB talk or listen address using the CO bit and the CDOR. When there is another device on the GPIB acting as Controller, the GPIB-SBX is addressed with GPIB command messages to become a Talker or Listener.

Address Mode 1

If the GPIB-SBX ADMR has been configured for address mode 1, the GPIB-SBX responds to the reception of two primary GPIB addresses: major and minor. Upon receipt of its major or minor MTA or its major or minor MLA from the GPIB Active Controller, the GPIB-SBX is addressed as Talker or Listener. If the GPIB-SBX has received its GPIB Talk address, the TA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DO bit in ISR1 is set. If the GPIB-SBX has received its GPIB listen address, the LA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DI bit in ISR1 is set when the first GPIB data byte is received.

Address Mode 2

Address mode 2 is used when Talker Extended (TE) or Listener Extended (LE) functions are to be used. TE and LE functions require receipt of two addresses (primary and secondary) before setting the TA or the LA bits. The GPIB-SBX GPIB primary address is specified by the byte written to ADR0. The secondary address is specified by the byte written to ADR1. Upon receipt of both the primary and secondary GPIB addresses, the GPIB-SBX becomes an addressed Talker or Listener. If the GPIB-SBX has received its primary GPIB talk address, the Talker Primary Addressed State (TPAS) bit in the ADSR is set. If the GPIB-SBX receives its secondary GPIB talk address before receiving another GPIB Primary Command Group (PCG) message that is not its MTA, the TA bit in the ADSR, the ADSC bit in the ISR2, and the DO bit in the ISR1 are set. If the GPIB-SBX has received its primary GPIB listen address, the Listener Primary Addressed State (LPAS) bit in the ADSR is set. If the GPIB-SBX receives its secondary GPIB listen address before receiving another GPIB PCG message that is not its MLA, the LA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DI bit in ISR1 is set when the first GPIB data byte is received. The Major-Minor (MJMN) bit in the ADSR indicates whether the address status refers to the major or minor address.

Address Mode 3

Address mode 3, like address mode 2, is used to implement extended GPIB talk and listen address recognition. However, unlike address mode 2, address mode 3 provides for both major and minor primary addresses, and the user's program must identify the secondary address by reading the Command Pass Through Register (CPTR). The sequence of events for proper operation using address mode 3 is as follows:

1. During initialization of the GPIB-SBX, enable address mode 3 (and optionally set the Enable Interrupt on Address Pass Through (APT IE) bit in IMR1 to enable an interrupt request on receipt of a secondary GPIB address). Write the GPIB-SBX major GPIB primary address to Address Register 0 (ADR0) and the GPIB-SBX minor GPIB primary address to Address Register 1 (ADR1).

2. Receipt of the GPIB-SBX major or minor primary MTA, or major or minor primary MLA, sets TPAS or LPAS, indicating that the primary address has been received.

3. If the next GPIB command following the primary address is a secondary address, the APT bit is set and a Data Accepted (DAC) Handshake Holdoff is activated (the GPIB DAC message is held false).

4. In response to APT, the program must:

   - Determine whether the command just received is a listen, talk, major, or minor address by reading the LPAS, TPAS, and MJMN bits of the ADSR.

   - Read the secondary address in the CPTR and determine whether or not it is the address of the GPIB-SBX.

5. If it *is not* the GPIB-SBX address, issue the Non-Valid auxiliary command. If it *is* the GPIB-SBX address, issue the Valid auxiliary command.

6. If the Valid auxiliary command is issued, the GPIB-SBX assumes that the My Secondary Address (MSA) message has been received, which causes:

   - Either

     - The LA bit to be set and the TA bit to be cleared:

       Listener Addressed State (LADS) = Talker Idle State (TIDS) = 1

       if LPAS was set, or
     - The TA bit to be set and the LA bit to be cleared:

       Talker Addressed State (TADS) = Listener Idle State (LIDS) = 1

       if TPAS was set.

   - The GPIB DAC message to be sent true and the GPIB Handshake to finish.

7. If the Non-Valid auxiliary command is issued, the GPIB-SBX assumes that an Other Secondary Address (OSA) message has been received, which causes:

   - The GPIB-SBX Talker or Listener function to go to its idle state (TIDS=1 or LIDS=1) if either the TPAS or LPAS bit was set; and

   - The GPIB DAC message to be sent true and the GPIB Handshake to finish.

Until a PCG message is received (that is, as long as the subsequent messages are secondary addresses), the APT bit is set and a DAC Holdoff is in effect each time a GPIB secondary address is received. In this way, the GPIB CIC can address several devices having the same primary address each time. If a PCG message is received before a secondary address is received, the TPAS and LPAS bits are cleared.

Sending/Receiving Messages

When the GPIB-SBX is a GPIB Talker or Listener, data (device-dependent messages) can be sent or received.

Sending and Receiving Data (Programmed I/O Mode)

To send data, wait until the GPIB-SBX has been programmed or addressed to talk and the CDOR is empty. When this occurs, the DO bit in the ISR1 is set, indicating that it is safe to write a byte to the CDOR. The DO bit is set again once the byte has been received by all GPIB Listeners.

To receive data, wait until the GPIB-SBX has been programmed or addressed to listen and the CDOR is empty. When this occurs, the DI bit in the ISR1 is set, indicating that the GPIB Talker has written a byte to the DIR. Once that byte has been read, the DI bit is set again when a new byte is received from the GPIB Talker.

Determining when the CDOR is empty or the DIR is full can be done by polling the ISR1 until the DO or DI status first appears or by allowing a program interrupt to occur on the respective event. Remember, however, that the status bits and interrupt signals are cleared when the ISR1 is read, so the absence of a true DO or DI status does not indicate that the CDOR is still full or that the DIR is still empty.

Sending and Receiving Data (DMA Mode)

To send data, configure the external DMA controller with the starting memory address of the buffer containing the data, the number of bytes to transfer, and the direction of the transfer. Arm the controller and set the DMAO bit in IMR2 to enable the TLC to request a DMA transfer when the CDOR is empty and the TLC is a Talker. The DMA controller will indicate in some manner, such as a terminal count status or interrupt, that all bytes have been transferred.

To receive data, configure and arm the DMA controller in the same manner but for the opposite direction. Set the DMAI bit in IMR2 to enable the TLC to request a DMA transfer when the external GPIB Talker device has written a byte to the TLC's DIR and the TLC is a Listener.

Sending END or EOS

The GPIB END message is sent by issuing the Send EOI auxiliary command just before writing the last data byte to the CDOR. The GPIB EOS message is sent simply by making the last byte the EOS code.

Terminating on END or EOS

The END status bit or interrupt is used to inform the program of the occurrence of an END message or an EOS message.

Serial Polls

Serial Polls allow the CIC to obtain detailed status information on each device that has been configured for responding.

Conducting Serial Polls

The GPIB-SBX, as CIC, serially polls other devices as described in the IEEE Standard 488-1987, *IEEE Standard Digital Interface for Programmable Instrumentation*. From the programming point of view, the GPIB-SBX must first become Active Controller to send the addressing and enabling commands to the device being polled, make itself a GPIB Listener by either issuing the Listen auxiliary command or sending its listen address, and go to standby with the Go To Standby auxiliary command in order to read the status byte.

Responding to a Serial Poll

The CIC can conduct Serial Polls to determine which device is asserting the GPIB SRQ signal to request service.

Before requesting service, you must check the Pending (PEND) bit of the SPSR to ensure that the GPIB-SBX is not presently in the middle of a Serial Poll (SPAS=0). If PEND=0, write the desired Status Byte (STB) to the SPMR with the rsv bit set. At that time, PEND sets and remains set until the Serial Poll (SP) completes.

Once Request Service (rsv) is set, the GPIB-SBX waits until any current SP is complete and asserts the GPIB SRQ signal. In response to that signal, the CIC starts the SP by addressing the GPIB-SBX to talk and sends the Serial Poll Enable (SPE) command. When the CIC unasserts ATN, the GPIB-SBX unasserts Service Request (SRQ) and transfers the STB message on to the GPIB data bus with DIO7 (the RSQ signal) asserted.

While the SP is in progress (SPAS=1), the CIC normally reads the STB only once but may read it any number of times provided that it asserts ATN between each one byte read. However, RSQ is set only during the first read; after the first read, rsv is cleared also. PEND is cleared when the CIC asserts ATN to terminate the SP.

The GPIB EOI line is asserted along with the status byte (that is, the END message is sent) during the SP if bit Send Serial Poll End Or Identify (SPEOI) of AUXRB is set.

To complete the SP, the CIC sends the Serial Poll Disable (SPD) command.

Parallel Polls

Parallel Polls are used by the GPIB Active Controller to check the status of several devices simultaneously. The meaning of the status returned by the devices being polled is device-dependent, but there are two general ways in which a Parallel Poll (PP) is useful.

- When the GPIB Controller recognizes SRQ asserted in a system with eight or less devices, it can determine quickly, usually using only one PP, which one needs to be serially polled.
- In systems in which the Controller response time requirement to service a device is low and the number of devices is small, PPs can replace SPs entirely, provided that the Controller polls frequently.

Although the Controller can obtain a Parallel Poll Response (PPR) quickly and at any time, there can be considerable front-end overhead during initialization to configure the devices to respond appropriately. This is contrasted with SPs, where the overhead, in the form of addressing and enabling command messages, occurs with each PP.

Conducting a Parallel Poll

The GPIB-SBX as Active Controller has the capability to conduct a PP. When the Execute Parallel Poll auxiliary command is issued and the GPIB-SBX internal local message rpp is set, the PP is executed (that is, the GPIB message IDY is sent true) as soon as the GPIB-SBX Controller interface function is placed in the proper state (CAWS or CACS). The PPR is automatically read from the GPIB DIO line into the CPTR and the rpp local message is cleared. A program can determine that the PP operation is complete based upon the condition of CO (CO=1 when the poll is complete). The response can be obtained by reading the contents of the CPTR. The response is held in the CPTR until a GPIB command is transmitted or the GPIB-SBX Controller function becomes inactive.

In response to IDY, each device participating in the PP drives only one GPIB DIO line (its PPR) active true or passive false, while it drives the other lines passive false.

Since there are eight data lines, and for each line there can be two responses (true or false), there are 16 possible responses. The line a device uses and how that device drives the line depends upon how it is configured and whether its local individual status message (ist) is one or zero. Thus, each device on the GPIB can be configured to drive its assigned DIO line true if ist=1 and to drive the DIO line false if ist=0; or it can be configured to do exactly the opposite (that is, to drive the DIO line true if ist=0 and false if ist=1). (The meaning of the value of ist, whether one or zero, is system- or device-dependent.)

Because the data lines are driven with the Open Collector driver during Parallel Polls, more than one device can respond on each line. The device or devices asserting the line true overrides any device asserting the line false. The Controller must know in advance whether a true response means the local ist message of the device is one or zero. To do this, the device must be configured to respond in one of the following ways:

- **Local configuration** (Parallel Poll capability code PP2) involves assigning a response line and a sense from the device side, in a similar manner to assigning the device GPIB address. Thus, one device might be assigned to respond with remote message PPR1 (driving DIO1), while a second device is assigned to respond with remote message PPR3 (driving DIO3), both positive (that is, a true response if ist=1). Local configuration is static in the sense that it does not change after the system is integrated (system installed and configured).

- **Remote configuration** (Parallel Poll capability code PP1) involves the dynamic assigning of response line and sense to devices on the GPIB. This is accomplished using Parallel Poll Enable (PPE) and Parallel Poll Disable (PPD) commands, which are issued by the Active Controller. The following procedure is recommended for remotely configuring devices:

    1. Place the GPIB-SBX in the Active Controller state.

    2. Send the GPIB UNL (Unlisten) message to unaddress all GPIB Listeners.

    3. Send the listen address of the first device to be configured.

4. Send the GPIB PPC message followed by the PPE message for that device.

5. Repeat from the second step (UNL) for each additional device.

The same procedure should be followed to disable polling with PPD (for example, when changing responses during reconfiguration).

Responding to a Parallel Poll

Before the GPIB-SBX can be polled by the CIC, the GPIB-SBX must be configured either locally by the user program at initialization time, or remotely by the CIC. Configuration involves the following:

- Enabling the GPIB-SBX to participate in polls
- Selecting the sense or polarity of the response
- Selecting the GPIB data line on which the response is asserted when the CIC issues the Identify (IDY) message

With remote configuration (PP1), the GPIB-SBX interprets the configuration commands received from the CIC, without any software assistance or interpretation from the user program. With local configuration (PP2), the three actions listed above must be explicitly handled in the software by writing the appropriate values to the U, S, and P3 to P1 bits of the PPR. Refer to the *Parallel Poll Register* description in Chapter 4 for more information.

Once the PPR is configured, all that remains for the user program is to determine the source and value of the local individual status (ist) message. If the Individual Status Select (ISS) bit in the AUXRB is zero, ist is set and cleared via the Set and Clear Parallel Poll auxiliary commands. If ISS is one, ist is set if the Service Request function of the GPIB-SBX is in the Service Request State (SRQS) and the GPIB-SBX is asserting the GPIB SRQ signal line and cleared otherwise. Consequently, setting ISS ties the Parallel Poll function to the Service Request function and also to the Serial Poll process.

The particular response sent by the GPIB-SBX during a Parallel Poll is determined by the value of ist and the configuration of the GPIB-SBX. The value of ist and the actual configuration must be decided by the GPIB system integrator. The response can be changed dynamically during program execution by changing the value of ist and, when remote configuration is used, by reconfiguration.

Interrupts

GPIB interrupt requests are enabled using the IE bits in IMR1 and IMR2. Once asserted, the interrupt request line remains asserted until the corresponding status register is read (ISR1 or ISR2). A hardware jumper may reside on the host board to additionally enable or disable the request from the GPIB-SBX from causing a system interrupt.

To emphasize once more, the status bits in ISR1 or ISR2 are all automatically cleared when the register is read, even if the conditions are still true. If two conditions are true at the same time (that is, more than one bit in ISR1 or ISR2 is set), a software copy of the register should be maintained if the program is going to analyze the conditions one at a time.

Chapter 6
Theory of Operation

This chapter explains the operation of each functional unit making up the GPIB-SBX, with references to signals and circuits shown in the schematic diagram in Appendix A. A brief description of the GPIB-SBX interface board along with a functional block diagram can be found in Chapter 2.

Terminology used throughout this chapter is explained in the Preface.

SBX Bus Interface

The host board to which the GPIB-SBX is attached provides all signal buffering, chip selection, clock source, and optionally, DMA control. Connected to the SBX bus are the following four circuits:

- µPD7210 GPIB Talker/Listener/Controller (TLC) circuit
- System Controller selection circuit
- 10-MHz clock divider
- DMA acknowledge extender

The TLC, which is the heart of the GPIB-SBX, is discussed in greater detail in the *GPIB Interface* section later in this chapter.

The System Controller (SC) selection circuit decodes auxiliary commands written to the TLC and sets or clears the SC flip-flop when specified commands are detected. The flip-flop is cleared when the following commands are detected:

- Set IFC
- Clear IFC
- Set REN
- Clear REN

When cleared, the Q* output of the flip-flop drives the SC input of the 75l6Z GPIB Control Transceiver high, which causes the IFC and REN lines to be driven by the TLC. The flip-flop is cleared when a bus reset (RESET) is asserted or when the Disable System Control command is detected.

Notice that the flip-flop is not cleared by the Immediate Execute pon or Chip Reset commands.

Another flip-flop divides the 10-MHz MCLK to 5 MHz for use by the TLC to generate proper GPIB timing.

Two flip-flops are used to extend the duration of MDACK* by up to two clock cycles to prevent a glitch on the MDRQT line.

GPIB Talker/Listener/Controller (TLC)

The TLC is a large scale integrated circuit (NEC µPD7210) that contains most of the logic circuitry needed to program, control, and monitor the GPIB interface functions that are implemented by the GPIB-SBX. Access to these functions is through eight read-only registers and 13 write-only registers, of which five are indirectly addressed. These registers occupy a block of eight I/O addresses.

The TLC is enabled during the MCS0* pulse and the SBX address signals MA2-MA0 are decoded internally to access the appropriate register. Data on the MD bus is strobed into write-only registers at the trailing edge of IOWRT*. Data in the read-only registers is placed on the MD bus a minimum access time after MCS0* and IORD* are both true.

Most of the interface functions can be implemented or activated from either side of the TLC; that is, the interface can be *programmed* to do these functions from the host board or it can be *addressed* to do them from the GPIB. In terms of the IEEE-488 standard, the distinction between the two is generally that between local and remote interface messages, respectively.

Command and data to the GPIB and data from the GPIB are pipelined through the Command/Data Out Register (CDOR) and the Data In Register (DIR), respectively.

Interrupt Mask Registers 1 and 2 (IMR1 and IMR2) are used to enable or disable the generation of the TLC INT signal on the occurrence of 13 key GPIB conditions or events. IMR2 also has two bits to enable DMA transfers to/from the TLC using the DMA controller facility of the host board. Interrupt Status Registers 1 and 2 (ISR1/2) record the occurrence of the 13 conditions or events.

Note: ISR1 and ISR2 are not true status registers in that the bits are cleared whenever they are read. Appropriate programming steps must be taken to derive status information from the information provided by these registers.

The status bits function independently of the corresponding mask bits.

The Address Status Register (ADRS) reveals the current status of the Controller, Talker, and Listener functions. The Address Mode Register (ADMR) is used to program or enable the desired addressing mode as well as determine the use of the TLC's three T/R output signals.

Interface messages (commands) that are not automatically deciphered and implemented by the TLC can be read by the driver program through the Command Pass Through Register (CPTR). The program can then decide what action to take, usually by writing commands to the Auxiliary Mode Register (AUXMR).

The AUXMR is used both to issue special commands to the TLC and to write to the five hidden registers. The Internal Clock Register (ICR) is used to set critical timing parameters based on the frequency of the TLC input clock, which on the GPIB-SBX is 5 MHz. The Parallel Poll Register (PPR) is used to locally configure the TLC for polling. Auxiliary Registers A, B, and E

(AUXRA, AUXRB, and AUXRE) provide a means to control a variety of diverse functions, such as enabling handshake holdoffs, transmitting END when the EOS byte is sent, setting the END RX bit when EOS is received, enabling high-speed transfers, and others.

The Address Register (ADR) is used to program two address registers, ADR0 and ADR1, which contain the base GPIB addresses recognized by the TLC as well as Talker and Listener disabling bits. The manner in which the TLC uses these registers depends upon the address mode established in the ADMR. A bit in ADR1 indicates if END was set on the last byte received.

The TLC can automatically determine when an end of string character (EOS) is received by comparing each byte against the character stored in the End of String Register (EOSR).

GPIB Interface

The TLC is interfaced to the GPIB through two special purpose transceivers: a 75160 for the data signals and a 75162 for the handshake and interface management signals. Signal direction routing through these transceivers is controlled by three signals from the TLC (T/R1-T/R3) and the SC signal from the System Controller Select circuit. T/R1 is high when the TLC is a Talker or Active Controller, and low when it is a Listener; it controls the direction of the data, handshake and EOI signals. T/R2 is high when the TLC is Controller-In-Charge and low otherwise; it controls the direction of the ATN and SRQ signals. T/R3 is high when the three-state driver mode is active and low when the open collector mode is active; when the GPIB-SBX is parallel polled, the transceiver switches to open collector mode during the poll. SC is set as explained in the *SBX Bus Interface* section earlier in this chapter. SC controls the direction of the IFC and REN signals, driving the GPIB when SC is high and receiving from the GPIB when it is low.

Chapter 7
Diagnostic and Troubleshooting Test Procedures

This chapter contains test procedures for determining if the GPIB-SBX is installed and operating correctly. The tests are similar to those used by National Instruments to verify correct hardware functioning. This method programs specific internal functions by writing to one or more registers, then reading other registers to confirm that the functions were implemented. You must have available an appropriate mechanism for writing to and reading from memory locations. A program such as an interactive control program, console emulator, monitor, or program debugger is ideal for this purpose.

Interpreting Test Procedures

The following test procedures are written in the form of simple equations. The left side of the equation contains the hexadecimal address offset from the GPIB-SBX base address and mnemonic for the register. The right side of the equation contains a hex value. Converting the hex value to binary results in a representation of the bit pattern in the register. For example, a hex value of FF corresponds to a bit pattern of 11111111, 40 (hex) corresponds to a bit pattern of 01000000.

Equations not followed by a question mark are instructions to the user to load the value shown into the designated register. Equations followed by a question mark are instructions to the user to read the register and verify that the value in the register is the one indicated.

The column to the left of each test step contains the relative register address. Comments written to the right of each test step briefly describe the action taken, and sometimes suggest the purpose.

The test procedures are designed to check the most elemental levels of functioning first, and then progress to tests of higher complexity. For this reason, users are advised to perform the tests in the order given.

If the GPIB-SBX does not perform as described in the test procedures, carefully perform the following steps.

1. Verify that the test instructions have been followed correctly.
2. Examine any read and write routines being used in connection with the checkout procedure for errors.
3. Recheck the jumper settings described in Chapter 3.

After these items have been carefully checked, if the interface is still not functioning properly, gather together the information concerning what the GPIB-SBX is and is not doing with regard to the expected results and contact National Instruments.

GPIB-SBX Hardware Installation Tests

1. Initialize TLC.

5	AUXMR = 2	Chip Reset
5	AUXMR = 0	Immediate execute pon

2. Send Chip Reset, then read registers and compare to reset values.

5	AUXMR = 2	Chip Reset
1	ISR1 = 0?	
2	ISR2 = 0?	
3	SPSR = 0?	
4	ADSR = 40?	
5	CPTR = 0?	

3. Test ton, DO, ERR, CPTR, TA.

5	AUXMR = 2	Chip Reset
4	ADMR = 80	ton
5	AUXMR = 0	Immediate execute pon
4	ADSR = 42?	TA
1	ISR1 = 2?	DO
0	CDOR = 51	write data byte
5	CPTR = 51?	verify
1	ISR1 = 6?	DO + ERR
1	ISR1 = 0?	bits cleared when read
5	AUXMR = 2	Chip Reset
4	ADMR = 0	disable ton
5	AUXMR = 0	Immediate execute pon
4	ADSR = 40?	not TA

4. Check lon, LA.

5	AUXMR = 2	Chip Reset
1	IMR1 = 0	no interrupts
2	IMR2 = 0	no interrupts
4	ADMR = 40	lon
5	AUXMR = 0	Immediate execute pon
4	ADSR = 44?	LA
5	AUXMR = 2	Chip Reset
4	ADSR = 40?	not LA

5. Test ATN, CIC, CO.

5	AUXMR = 2	Chip Reset
4	ADMR = 31	Address Mode 1
5	AUXMR = 0	Immediate execute pon
5	AUXMR = 1E	set IFC
5	AUXMR = 16	clear IFC
4	ADSR = 80?	CIC
2	ISR2 = 9?	CO + ADSC
5	AUXMR = 10	go to standby
4	ADSR = CO?	CIC + ATN*

Appendix A
Parts List and Schematic Diagram

This appendix contains the parts list and schematic diagram for the GPIB-SBX.

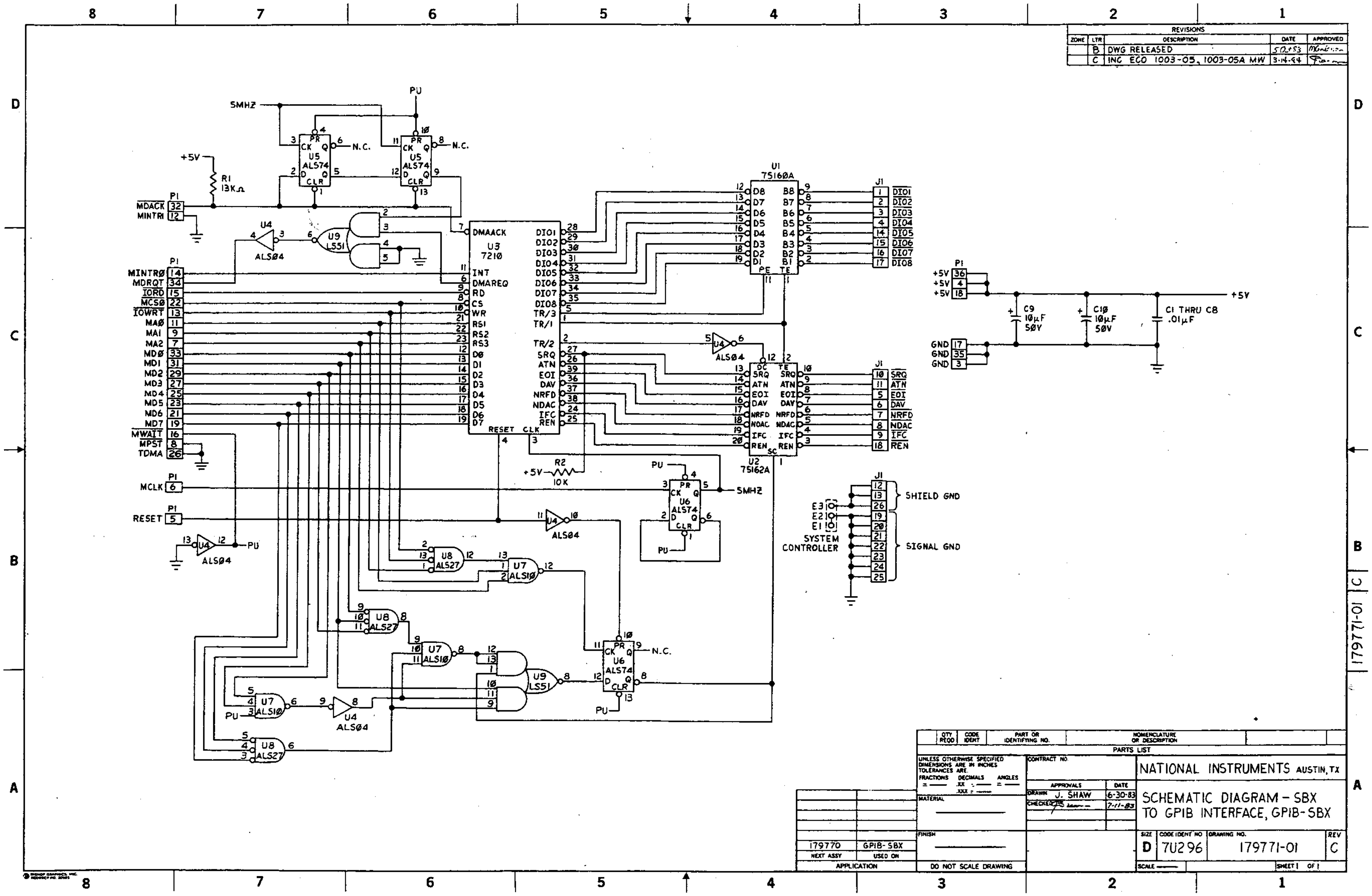
ITEM	QTY	MFR	PART/DWG	DESCRIPTION
NO	REQD	FSCM NO	STOCK NO	REFERENCE DESIGNATION
1	1	NI	179772-01	PWB, GPIB TO SBX INTFC-GPIB-SBX
		7U296	179772-01	
2	REF	NI	179771-01	SCHEM DIAG, GPIB TO SBX INTFC-GPIB-SBX
		7U296	179771-01	
3	8	CLB	UK50-103	CAPACITOR, RDL LEAD, 50 V, 20%, .01 UF
		71950	715003-01	C1-8
4	2	SPG	199D106X0016CE2	CAPACITOR, RDL LEAD, 16 V, 20%, 10 UF
			715062-01	C9, 10
5	3	BERG	65500-101	HEADER, SGL ROW, STR, .1 CTR, 1 POSN
		22526	760013-01	E1-3
6	1	BERG	65268-009	HEADER, SHLDED, .1 CTR, 26-POSITION
		22526	760010-01	J1
7	1	VIKING	000292-0001	PLUG, STACK CON SERIES
			760146-01	P1
8	1	-	RC07GF133J	RESISTOR, 1/4W, 5%, 13K OHM
			711118-01	R1
9	1	NS	DS75160AN	IC, OCTAL IEEE-488 DATA BUS XCVR
			700106-01	U1
10	1	NS	DS75162AN	IC, OCTAL IEEE-488 CONTROL BUS XCVR
			700107-01	U2
11	1	NEC	uPD7210	IC, GPIB INTERFACE CTRLR
			700200-01	U3
12	1	TI	SN74ALS04N	IC, 14-PIN DIP, PLASTIC, HEX INVERTER
		06668	700069-01	U4
13	2	TI	SN74ALS74N	IC, DUAL D-TYPE POS-EDGE- TRIGGERED FF
		06668	700072-01	U5, 6
14	1	TI	SN74ALS10N	IC, TPL 3-INPUT POS NAND GATE
		06668	700306-01	U7
15	1	TI	SN74ALS27N	IC, 14-PIN DIP, TRIPLE 3-INPUT NOR GATE
		06668	700262-01	U8
16	1	TI	SN74LS51N	IC, 2-WIDE 2-INPUT &-OR-INVERT GATE
		06668	700012-01	U9
17	1	AMP	531220-3	CONNECTOR, 2 POSI, .1 CTR, MINI-JUMP
		00779	760014-02	
18	2	HHS	2503	SCREW, NYLON, 6-32 X 1/4 IN., BIND HEAD
		83330	741216-01	

TITLE	FSCM NO	DWG NO	REV
PARTS LIST- CCA, GPIB TO SBX INTFC-GPIB-SBX		PL 179770-01	C

Thu Mar 29 19:31:08 1984

ITEM NO	QTY REQD	MFR FSCM NO	PART/DWG STOCK NO	DESCRIPTION REFERENCE DESIGNATION
19	1	HHS	4057	SPACER, NYLON, 6-32 THD, .5 IN. LENGTH
		83330	745092-01	
20	1	-	RC07GF103J	RESISTOR, 1/4 W, 5%, 10K OHM
		71785	711011-01	R2

TITLE	FSCM NO	DWG NO	REV
PARTS LIST- CCA, GPIB TO SBX INTFC-GPIB-SBX		PL 179770-01 Thu Mar 29 19:31:08 1984	C



REVISIONS				
ZONE	LTR	DESCRIPTION	DATE	APPROVED
	B	DWG RELEASED	5-2-83	
	C	INC ECO 1003-05, 1003-05A MW	3-14-84	

QTY REQD	CODE IDENT	PART OR IDENTIFYING NO.	NOMENCLATURE OR DESCRIPTION

PARTS LIST

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE:
FRACTIONS ± DECIMALS .XX ± .XXX ± ANGLES ±

MATERIAL

FINISH

DO NOT SCALE DRAWING

179770	GPIB-SBX
NEXT ASSY	USED ON
APPLICATION	

CONTRACT NO.

APPROVALS		DATE
DRAWN	J. SHAW	6-30-83
CHECKED		7-11-83

NATIONAL INSTRUMENTS AUSTIN, TX

SCHEMATIC DIAGRAM - SBX TO GPIB INTERFACE, GPIB-SBX

SIZE	CODE IDENT NO	DRAWING NO.	REV
D	7U296	179771-01	C

SCALE ——— SHEET 1 OF 1

179771-01 C

Appendix B
Sample Programs

This appendix contains listings of routines in 8080 and 8088 assembly language code that implement the essential elements of these major utility functions:

- Initialize the GPIB-SBX interface (INIT).
- Initialize the interface functions of the GPIB devices (IFC).
- Set or clear the GPIB REN line (REN).
- Accept data bytes from a Talker (RCV).
- Address Talker and read device-dependent messages (READ).
- Send data bytes to Listeners (DSEND).
- Address Listener and write device-dependent messages (WRITE).
- Send command bytes to Listeners (CSEND).
- Write interface messages (CMD).
- Pass GPIB control to another device (PASSC).

8088 code appears on the left, 8080 code appears in the middle, and comments applying to both sets of code are on the right. Assumptions regarding the state of the GPIB-SBX appear at the beginning of each routine and must be adhered to for proper, error-free operation.

The following characteristics of the code must be considered:

- The GPIB-SBX base address is 50 hex.
- Normal (non-extended) GPIB addressing is used.
- Time-out on subroutine calls is not implemented.
- Register values are not saved on subroutine calls.
- Program interrupt is not used; status checking is by register polling.
- Constants and variables listed in the User-Specified Constants section of the listings must be initialized to correct values.

- In operands containing expressions, + is used in place of logical OR for convenience. An arithmetic addition yields the same result for the instances here.

- The 8088 code assumes throughout that segment registers have been appropriately initialized.

- Also in the 8088 code, where the OFFSET operator is used, the user may also need to use the GROUP name as an override operator. The user is advised to refer to the 8088 manual for specific needs.

- In comments where a reference is made to a register, the 8088 register name appears first and is not enclosed in parentheses while the 8080 register name appears immediately after the 8088 name and is enclosed in parentheses. For example, a comment written as

  ```
  ;
  ;  Status on exit:
  ;   -The bx(bc) register pair contains . . .
  ;
  ```

  indicates that the 8088 code uses the bx register pair, while the 8080 code uses the bc register pair.

```
                          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                          ;                                        ;
                          ;  GPIB-SBX Sample Functions for Driver:    ;
                          ;                                        ;
                          ;       INIT  (Initialize the GPIB-SBX)   ;
                          ;       IFC   (Send Interface Clear)      ;
                          ;       REN   (Set/Clear Remote Enable)   ;
                          ;       RCV   (Receive)                   ;
                          ;       READ  (Read Data)                 ;
                          ;       DSEND (Data Send)                 ;
                          ;       WRITE (Write Data)                ;
                          ;       CSEND (Command Send)              ;
                          ;       CMD   (Write Commands)            ;
                          ;       PASSC (Pass Control)              ;
                          ;                                        ;
                          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                     ;
  ;       8088 Code                8080 Code       ;            Comments
  ; -------------------------  ---------------------------- ; ------------------------------------------
                                     ;
     (same as 8080 code)      BASE    equ     50h          ; Base address of GPIB-SBX interface
registers
                          DIR     equ     BASE + 0      ; Data In Register (read)
                          CDOR    equ     BASE + 0      ; Control/Data Out Register (write)
                          ISR1    equ     BASE + 1      ; Interrupt Status Register 1 (read)
                          IMR1    equ     BASE + 1      ; Interrupt Mask Register 1 (write)
                          ISR2    equ     BASE + 2      ; Interrupt Status Register 2 (read)
                          IMR2    equ     BASE + 2      ; Interrupt Mask Register 2 (write)
                          SPSR    equ     BASE + 3      ; Serial Poll Status Register (read)
                          SPMR    equ     BASE + 3      ; Serial Poll Mask Register (write)
                          ADSR    equ     BASE + 4      ; Address Status Register (read)
                          ADMR    equ     BASE + 4      ; Address Mode Register (write)
                          CPTR    equ     BASE + 5      ; Command Pass Thru Register (read)
                          AUXMR   equ     BASE + 5      ; Auxiliary Mode Register (write)
                          ADR0    equ     BASE + 6      ; Address Register 0 (read)
                          ADR     equ     BASE + 6      ; Address Register (write)
                          ADR1    equ     BASE + 7      ; Address Register 1 (read)
                          EOSR    equ     BASE + 7      ; End Of String Register (write)

                                    ; ISR1 Bits
                          DI      equ     001q        ; Data in
                          DO      equ     002q        ; Data out
                          ERR     equ     004q        ; Error
                          ENDRX   equ     020q        ; END received
```

```
;        8088 Code                   8080 Code        ;               Comments
; ------------------------  ---------------------------- ; ----------------------------------------
                                              ;

                                          ; ISR2 Bits
  (same as 8080 code)       CO      equu    010q          ; Command out

                                          ; ADSR Bits
                            NATN    equ     100q          ; Not ATN

                                          ; ADMR Bits
                            MODE1   equ     001q          ; Address Mode 1
                            TRM     equ     060q          ; GPIB-SBX functions for T/R2 and T/R3

                                          ; AUXMR Hidden Registers
                            ICR     equ     040q          ; Internal Counter Register
                            PPR     equ     140q          ; Parallel Poll Register
                            AUXRA   equ     200q          ; Auxiliary Register A
                            AUXRB   equ     240q          ; Auxiliary Register B
                            AUXRE   equ     300q          ; Auxiliary Register E

                                          ; AUXMR Commands
                            IEPON   equ     000q          ; Immediate execute power on
                            FH      equ     003q          ; Finish (release) handshake
                            SEOI    equ     006q          ; Send END
                            GTS     equ     020q          ; Go to standby
                            TCA     equ     021q          ; Take control asynchronously
                            TCS     equ     022q          ; Take control synchronously
                            TCSE    equ     032q          ; Take control synchronously on END
                            LTN     equ     023q          ; Listen
                            LTNC    equ     033q          ; Listen continuously
                            LUN     equ     034q          ; Unlisten
                            SIFC    equ     036q          ; Set IFC
                            CIFC    equ     026q          ; Clear IFC
                            SREN    equ     037q          ; Set REN
                            CREN    equ     027q          ; Clear REN

                                          ; GPIB Commands
                            TCT     equ     011q          ; Take control
                            UNL     equ     077q          ; Universal unlisten
                            UNT     equ     137q          ; Universal untalk


;        8088 Code                   8080 Code        ;               Comments
; ------------------------  ---------------------------- ; ----------------------------------------
                                              ;

                                          ; User Specified Constants
  (same as 8080 code)       SEL0    equ     0             ; Select ADR0
                            SEL1    equ     200q          ; Select ADR1
                            MA      equ     0             ; GPIB address of GPIB-SBX
                            SC      equ     020q          ; System Controller (set to 0 if not System
Controller)
```

```
                                    ; Program Variables/Buffers
    cic     db      0       ;    cic     db      0          ; Controller-In-Charge flag (non-zero if CIC)
    cmdbuf  db      100 dup(?);  cmdbuf  ds      100        ; Command buffer for interface messages
    cmdct   db      0       ;    cmdct   db      0          ; Number of commands to be sent
    count   db      0       ;    count   db      0          ; Current number of commands transfered
    datbuf  ds      100 dup(?);  datbuf  db      100        ; Data buffer for device dependent messages
    datct   db      0       ;    datct   db      0          ; Number of data bytes to be sent
    ola     db      0       ;    ola     db      0          ; Listen address passed to WRITE
    sre     db      0       ;    sre     db      0          ; REN flag (zero to not set REN, non-zero to set
REN)
    tctadr  db      0       ;    tctadr  db      0          ; TCT address of new Active Controller
    vseoi   db      0       ;    vseoi   db      0          ; SEOI flag (zero to not send, non-zero to send
                                                            ; END message with last DSEND byte)


                                    ;    * * * * * * * * * * * *
                                    ;    *  INITIALIZE - INIT  *
                                    ;    * * * * * * * * * * * *
                                    ;
                                    ; Summary:
                                    ;       - Initialize the GPIB-SBX hardware
                                    ;
                                    ; Assumptions on entry:
                                    ;       - User specified constants MA and SC
                                    ;         have been initialized
                                    ;       - Mode 1 primary addressing is used
                                    ;       - Low speed timing is used
                                    ;       - Interrupts are not used
                                    ;       - Status byte will be set elsewhere
                                    ;       - Remote Parallel Poll configuration
                                    ;         will be used
                                    ;
                                    ; Actions:
                                    ;       - Pulse IEPON to put hardware in known
                                    ;         reset state
                                    ;       - Disable interrupts and clear status registers
                                    ;       - Set hardware registers to desired
                                    ;         values
                                    ;
                                    ; Status on return:
                                    ;       - The following registers are cleared:
                                    ;         ISR1/2, IMR1/2, SPMR, SPSR,
                                    ;         PPR, AUXRA, AUXRB, AUXRE
                                    ;       - Other registers are configured as
                                    ;         described
                                    ;       - The GPIB-SBX interface functions are reset
                                    ;         to their idle states and are enabled
                                    ;
                                    ;
                                    ;
    ;     8088 Code                  8080 Code          ;                  Comments
    ;----------------------  ---------------------------- ; ---------------------------------------------
                                                   ;
    INIT:   mov     al,IEPON;       mvi     a,IEPON         ; Initialize and enable TLC functions
            out     AUXMR,al;       out     AUXMR           ;
            out     IMR1,al ;       out     IMR1            ; Disable TLC interrupts
            out     IMR2,al ;       out     IMR2            ;
            in      al,ISR1 ;       in      ISR1            ; Clear status bits by reading registers
            in      al,ISR2 ;       in      ISR2            ;
            mov     al,MODE1+TRM;   mvi     a,MODE1+TRM     ; Set address mode, Talker/Listener
inactive,
            out     ADMR,al ;       out     ADMR            ; and proper T/R signal mode
            mov     al,MA+SEL0;     mvi     a,MA+SEL0       ; Set GPIB address (mode 1 primary
only), with
            out     ADR,al  ;       out     ADR             ; Talker/Listener enabled
            mov     al,DT1+DL1+SEL1;mvi     a,DT1+DL1+SEL1  ; Disable secondary address
recognition
            out     ADR,al  ;       out     ADR             ;

    ;     8088 Code                  8080 Code          ;                  Comments
    ;-----------------------  -------------------------- ; -----------------------------------------------
                                                   ;
            mov     al,ICR+5;       mvi     a,ICR+5         ; Set clock divider for 5MHz, low speed
            out     AUXMR,al;       out     AUXMR           ;
            ret             ;       ret                     ;


                                                   ; * * * * * * * * * * * * * * *
                                                   ; *   INTERFACE CLEAR - IFC   *
                                                   ; * * * * * * * * * * * * * * *
                                                   ;
                                                   ;
                                                   ; Summary:
                                                   ;       - Initialize the interface function of
                                                   ;         other GPIB devices
                                                   ;
                                                   ; Assumptions on entry:
                                                   ;       - GPIB-SBX has been initialized
                                                   ;
                                                   ; Actions:
                                                   ;       - Assert GPIB IFC (the interface
                                                   ;         automatically becomes System Controller
                                                   ;         and Active Controller)
                                                   ;       - Wait at least 100 microseconds
                                                   ;       - Unassert IFC
                                                   ;
                                                   ; Status on return:
                                                   ;       - GPIB-SBX is Active Controller
                                                   ;       - Interface functions of other GPIB
                                                   ;         devices are reset to their idle states
                                                   ;
```

```
;     8088 Code                 8080 Code          ;                  Comments
;----------------------  --------------------------   ; ---------------------------------------------
                                                   ;
IFC:    mov     al,SIFC ;FC:    mvi     a,SIFC          ; Set the IFC signal
        out     AUXMR,al;       out     AUXMR           ;
        mov     cx,100  ;       mvi     a,100           ; Wait at least 100 microseconds
IFC1:   loop    IFC1    ;IFC1:  dcr     a               ;
                        ;       jp      IFC1            ;
        mov     al,CIFC ;       mvi     a,CIFC          ; Clear IFC
        out     AUXMR,al;       out     AUXMR           ;
        ret             ;       ret                     ;

                                                   ;    * * * * * * * * * * * *
                                                   ;    * REMOTE ENABLE - REN *
                                                   ;    * * * * * * * * * * * *
                                                   ;
                                                   ; Summary:
                                                   ;      - Set or clear GPIB Remote Enable signal
                                                   ;
                                                   ; Assumptions on entry:
                                                   ;      - User specified sre is non-zero if REN
                                                   ;        is to be asserted and is zero if REN
                                                   ;        is to be unasserted
                                                   ;      - GPIB-SBX is System Controller and
                                                   ;        Active Controller
                                                   ;
                                                   ; Actions:
                                                   ;      - Check sre flag.
                                                   ;           if non-zero (true) send REN
                                                   ;           else send clear REN
                                                   ;
                                                   ; Status on return:
                                                   ;      - REN is asserted or unasserted
                                                   ;
;     8088 Code                 8080 Code          ;                  Comments
;----------------------  --------------------------   ; ---------------------------------------------
                                                   ;
                                                   ;
REN:    mov     al,sre  ;REN:   lda     sre             ; Turn on the REN signal if sre is non-zero
        cmp     al,0    ;       cpi     0               ;
        jz      REN1    ;       jz      REN1            ;
        mov     al,SREN ;       mvi     a,SREN          ;
        out     AUXMR,al;       out     AUXMR           ;
        jmp     REN2    ;       jmp     REN2            ;
REN1:   mov     al,CREN ;REN1:  mvi     a,CREN          ; Else, turn off REN if sre is zero
        out     AUXMR,al;       out     AUXMR           ;
REN2:   ret             ;REN2:  ret                     ;
```

```
                                        ;     * * * * * * * * *
                                        ;     * RECEIVE - RCV *
                                        ;     * * * * * * * * *
                                        ;
                                        ; Summary:
                                        ;       - Called by READ to receive data if
                                        ;         GPIB-SBX is Controller-In-Charge
                                        ;       - Called directly from main program to receive
                                        ;         data if GPIB-SBX is Idle Controller
                                        ;
                                        ; Assumptions on entry:
                                        ;       - GPIB-SBX is Standby or Idle Controller
                                        ;       - GPIB-SBX is or will be addressed to listen
                                        ;       - The GPIB Talker has been or will be addressed
                                        ;       - The Talker will send END with last byte
                                        ;         if the number of bytes sent is less than the byte
                                        ;         count
                                        ;       - The bx(bc) register pair contains the byte count
                                        ;       - The di(hl) register pair contains the address
                                        ;         of the data buffer
                                        ;       - The user-specified variable cic is set
                                        ;         properly
                                        ;
                                        ; Actions:
                                        ;       - Release any holdoff in progress
                                        ;       - Set up handshake holdoffs as required
                                        ;         by Controller status (cic)
                                        ;       - Wait for GPIB END message or byte count
                                        ;       - If END set bx(bc) register pair to number of
                                        ;         bytes received
                                        ;       - Holdoff handshake
                                        ;
                                        ; Status on return:
                                        ;       - If GPIB-SBX is Standby Controller, or
                                        ;         if the transfer terminated on receiving END
                                        ;         a NRFD handshake holdoff is in effect;
                                        ;         otherwise, no holdoffs are active
                                        ;       - The number of bytes transfered is in bx(bc)
                                        ;
;     8088 Code              8080 Code        ;              Comments
;---------------------- ---------------------------- ; ----------------------------------------------
                                        ;
RCV:    mov   al,FH   ;RCV:  mvi    a,FH        ; Release any handshake holdoff in progress
        out   AUXMR,al;      out    AUXMR
        mov   al,cic  ;      lda    cic         ; Is GPIB-SBX Controller-In-Charge?
        cmp   al,0    ;      cpi    0           ;
        jz    RCV1    ;      jz     RCV1        ;
        mov   al,AUXRA+202q;
                      ;      mvi    a,AUXRA+22q ; Yes--Set HLDE and BIN in AUXRA
        out   AUXMR,al;      out    AUXMR       ;
        jmp   RCV2    ;      jmp    RCV2
```

```
RCV1:   mov     al,AUXRA;
                   ;RCV1:  mvi     a,AUXRA        ; No--Clear any HLDE or HLDA in effect
        out     AUXMR,al;       out     AUXMR


;     8088 Code                 8080 Code         ;                  Comments
;----------------------  ---------------------------- ; ------------------------------------------------
RCV2:   mov     cx,0    ;RCV2:  mvi     d,0            ; Clear byte counter
                   ;       mvi     e,0
                                    ;
RCV3:   in      al,ISR1 ;RCV3:  in      ISR1           ; Read status
        and     al,END+DI;      ani     END+DI         ; Wait for GPIB END or DI
        jz      RCV3    ;       jz      RCV3           ;
        ani     al,END  ;       ani     END            ; Look for END
        jnz     RCV5    ;       jnz     RCV5           ;
        in      al,DIR  ;       in      DIR            ; Read byte
        mov     byte ptr [di],al ;
                   ;       mov     m,a
        inc     di      ;       inx     h              ; Position next byte
        inc     cx      ;       inx     d              ; Inc byte counter
        cmp     bx,cx   ;       mov     a,d            ; More bytes to read ?
                   ;       cmp     b
                   ;       jnz     RCV3
                   ;       mov     a,e
                   ;       cmp     c
        jnz     RCV3    ;       jnz     RCV3           ; Yes--continue
        jmp     RCV6    ;       jmp     RCV6           ; No--exit
RCV5:                   ;RCV5:
        in      al,DIR  ;       in      DIR            ; END--read last byte
        mov     byte ptr [di],al
                   ;       mov     m,a            ;
        mov     bx,cx   ;       mov     b,d            ; Record bytes read
                   ;       mov     c,e            ;
RCV6:                   ;RCV6:                         ;
        mov     al,AUXRA+1;     lda     AUXRA+1        ; Send HLDA
        out     AUXMR,al;       out     AUXMR
        ret             ;       ret
```

```
                                    ; * * * * * *
                                    ; *  READ   *
                                    ; * * * * * *
                                    ;
                                    ; Summary:
                                    ;       - Called to read device dependent (data)
                                    ;         messages when the GPIB-SBX is Controller-
                                    ;         In-Charge (RCV is called when the
                                    ;         GPIB-SBX is Idle Controller)
                                    ;
                                    ; Assumptions on entry:
                                    ;       - GPIB-SBX is Controller-In-Charge
                                    ;       - The Talker address is placed in first
                                    ;         location of cmdbuf
                                    ;       - The variable cmdct is set to 1
                                    ;       - The buffer datbuf is free to place
                                    ;         incoming data
                                    ;       - The number of bytes to read is placed
                                    ;         in datct
                                    ; Actions:
                                    ;       - Set up cmdbuf and cmdct and
                                    ;         call CMD to address the Talker and
                                    ;         unaddress all other devices
                                    ;       - Program the GPIB-SBX to listen
                                    ;       - Go to standby and unassert ATN
                                    ;       - Transfer the contents of datct to the
                                    ;         bx(bc) register pair
                                    ;       - Load the di(hl) register pair with the address
                                    ;         of datbuf
                                    ;       - Call RCV to receive the data
                                    ;       - Call CMD to unaddress all devices
                                    ;       - Program the GPIB-SBX to unlisten
                                    ;
                                    ; Status on return:
                                    ;       - GPIB-SBX is Active Controller
                                    ;       - Acceptor handshake is held off at NRFD
                                    ;       - All GPIB devices are unaddressed
                                    ;
  ;     8088 Code              8080 Code         ;                 Comments
  ;----------------------  ---------------------------- ; ---------------------------------------------
                                    ;
  READ:   mov     al,cmdbuf;READ: lda     cmdbuf         ; Put Untalk and Unlisten commands
before
          mov     cmdbuf+2,al;    sta     cmdbuf + 2     ; Talker address in the buffer
          mov     al,UNT  ;       mvi     a,UNT
          mov     cmdbuf,al;      sta     cmdbuf
          mov     al,UNL  ;       mvi     a,UNL
          mov     cmdbuf+1,al;    sta     cmdbuf + 1
          mov     al,cmdct;       lda     cmdct
          add     al,2    ;       adi     2
```

```
        mov    cmdct,al;     sta    cmdct
        call   CMD    ;      call   CMD          ; Command routine will address the Talker
        mov    al,LTN  ;     mvi    a,LTN        ; Program GPIB-SBX to be a Listener
        out    AUXMR,al;     out    AUXMR        ; so it can take control synchronously
        mov    al,GTS  ;     mvi    a,GTS        ; later; then go to standby and drop ATN
        out    AUXMR,al;     out    AUXMR
        mov    bx,datct;     lhld   datct        ; Preset bx(bc) register pair with byte count


;     8088 Code                 8080 Code         ;                 Comments
;---------------------- ----------------------------- ; ---------------------------------------------
                                             ;
                       ;     mov    b,h          ;
                       ;     mov    c,l          ;
        mov    di,OFFSET datbuf;                 ;
                       ;     lxi    h,datbuf     ; Preset di(hl) register pair with the address
                                             ; of the data buffer
        call   RCV    ;      call   RCV          ; Receive routine will read data
        mov    al,TCS  ;     mvi    a,TCS        ; Take control
        out    AUXMR,al;     out    AUXMR        ;
READ1:  in     al,ADSR ;READ1: in   ADSR         ; Wait for ATN, indefinitely
        and    al,NATN ;     ani    a,NATN       ;
        jnz    READ1   ;     jnz    READ1        ;
                       ;     lda    cmdct        ; Prepare to unaddress all Talkers
        dec    cmdct   ;     dcr    a            ; and Listeners
                       ;     sta    cmdct
        call   CMD    ;      call   CMD
        mov    al,LUN  ;     mvi    a,LUN        ; Send Local Unlisten command
        out    AUXMR,al;     out    AUXMR
        ret            ;     ret
```

```
                             ; * * * * * * * * * * *
                             ; * DATA SEND - DSEND *
                             ; * * * * * * * * * * *
                             ;
                             ;
                             ;
                             ; Summary:
                             ;      - Called by WRITE to transmit data messages
                             ;        if the GPIB-SBX is Controller-In-Charge
                             ;      - Called directly from the main program
                             ;        if the GPIB-SBX is not CIC
                             ;
                             ; Assumptions on entry:
                             ;      - The GPIB-SBX is Standby or Idle Controller
                             ;      - GPIB-SBX is or will be addressed to talk
                             ;      - If the GPIB-SBX is Idle Controller, the current CIC
                             ;        will go to standby
                             ;      - The bx(bc) register pair contains the byte count
                             ;      - The si(hl) pair contains the address
                             ;        of the data buffer
                             ;      - The user specified variable veoi has been
                             ;        set properly
                             ;
                             ; Actions:
                             ;      - Copy byte count to cx(de)
                             ;      - Wait until the CDOR is empty
                             ;      - Decrement cx(de)
                             ;      - If last byte, assert EOI if in use
                             ;      - Write a byte
                             ;      - Check for a GPIB error
                             ;      - Loop until all bytes are transfered
                             ;      - On an error, set bx(bc) to -1
                             ;
                             ; Status on return:
                             ;      - The bx(bc) register pair contains the
                             ;        byte count or a -1 to indicate an error
                             ;
;     8088 Code              8080 Code       ;              Comments
;----------------------  ---------------------------- ; -----------------------------------------------
                             ;
DSEND:  mov     cx,bx   ;DSEND: mov     d,b         ; Copy byte count
                ;       mov     e,c         ;
DSEND1: in      al,ISR1 ;DSEND1:in      ISR1        ;
      and     al,DO+ERR;      ani     DO+ERR        ; Wait for CDOR or ERR
      jz      DSEND1  ;       jz      DSEND1        ;
      and     al,ERR  ;       ani     ERR           ; Look for error
      jnz     DSEND3  ;       jnz     DSEND3        ;
      dec     cx      ;       dcx     d             ; dec byte counter
                ;       mov     a,d
                ;       ora     d
```

```
        jl      DSEND4  ;       jm      DSEND4          ; Have all bytes been sent ?
                        ;       ora     e
        jnz     DSEND2  ;       jnz     DSEND2          ; No--Is this last byte ?
                        ;       lda     vseoi           ;


;       8088 Code                8080 Code          ;                 Comments
;----------------------  ---------------------------- ; ----------------------------------------------
        cmp     vseoi,0 ;       cpi     0               ; Yes--EOI in use ?
        jz      DSEND2  ;       jz      DSEND2          ; No
        mov     al,SEOI ;       mvi     a,SEOI          ; Send EOI with last byte
        out     AUXMR,al;       out     AUXMR           ;
                                                        ;
DSEND2: mov     al,byte ptr [si] ;
                        ;DSEND2:mov     a,m             ; Next byte
        out     CDOR,al ;       out     CDOR            ;
        inc     si      ;       inx     h               ;
        jmp     DSEND1  ;       jmp     DSEND1          ;
DSEND3: mov     bx,177777q ;DSEND3:mvi  b,377q          ; Return (-1) indicating error
                                mvi     c,377q
DSEND4: ret             ;DSEND4:ret                     ;
```

```
                                        ; * * * * * * *
                                        ; *   WRITE   *
                                        ; * * * * * * *
                                        ;
                                        ;
                                        ; Summary:
                                        ;       - Called to send device dependent (data) messages
                                        ;         when the GPIB-SBX is Controller-In-Charge
                                        ;         (DSEND is called when the interface is
                                        ;         Idle Controller)
                                        ;
                                        ; Assumptions on entry:
                                        ;       - GPIB-SBX is CIC
                                        ;       - One Listener is addressed and its address
                                        ;         is placed in the variable ola
                                        ;       - The data to be sent is placed in datbuf
                                        ;       - The variable datct contains the number of
                                        ;         bytes to send
                                        ;
                                        ; Actions:
                                        ;       - Set up cmdbuf and cmdct and call CMD
                                        ;         to address the GPIB-SBX as Talker, to address
                                        ;         the Listener, and to unaddress all other devices
                                        ;       - Go to standby and unassert ATN
                                        ;       - Transfer the contents of datct to the
                                        ;         bx(bc) register pair
                                        ;       - Load si(hl) register pair with the address of datbuf
                                        ;       - Call DSEND to write the data
                                        ;       - When the last byte has been sent, take
                                        ;         control
                                        ;       - Call CMD to unaddress all devices
                                        ;
                                        ; Status on return:
                                        ;       - The GPIB-SBX is Active Controller
                                        ;       - All GPIB devices are unaddressed
                                        ;
;     8088 Code                  8080 Code          ;                   Comments
;----------------------  ---------------------------- ; ---------------------------------------------
                                        ;
WRITE:  mov    cmdct,4 ;  WRITE:  mvi     a,4         ; Put Untalk, Unlisten, MTA, and OLA
                  ;       sta     cmdct          ; commands in the buffer
        mov    cmdbuf,UNT;    mvi     a,UNT          ;
                  ;       sta     cmdbuf         ;
                  ;       mvi     a,UNL          ;
        mov    cmdbuf+1,UNL;  sta     cmdbuf+1       ;
                  ;       mvi     a,MA+100q      ;
        mov    cmdbuf+2,MA+100q;                     ;
                  ;       sta     cmdbuf+2       ;
        mov    al, ola ;      lda     ola            ;
        mov    cmdbuf+3,al;   sta     cmdbuf+3       ;
        call    CMD     ;       call    CMD             ;
        mov     al,GTS  ;       mvi     a,GTS           ; Go to standby and drop ATN
        out     AUXMR,al;       out     AUXMR           ;


;       8088 Code               8080 Code          ;               Comments
;---------------------- ---------------------------- ; ------------------------------------------------
                                                   ;
        mov     bx,datct;       lhld    datct      ; Preset bx(bc) register pair with byte count
                        ;       mov     b,h        ;
                        ;       mov     c,l        ;
        mov     si,OFFSET datbuf;
                        ;       lxi     h,datbuf   ; Preset si(hl) register pair with address of data buffer
        call    DSEND   ;       call    DSEND      ; Source Handshake-Data will write data
WRITE1: in      al,ISR1 ;WRITE1:in      ISR1       ; Wait until last byte has been sent
        and     al,DO   ;       ani     DO         ;
        jz      WRITE1  ;       jz      WRITE1     ;
        mov     al,TCA  ;       mvi     a,TCA      ; Then take control
        out     AUXMR,al;       out     AUXMR      ;
                        ;       lda     cmdct      ; Prepare to unaddress all Talkers
        dec     cmdct   ;       dcr     a          ; and Listeners
                        ;       sta     cmdct      ;
        call    CMD     ;       call    CMD        ;
        ret             ;       ret                ;
```

```
                                        ;    * * * * * * * * * * * * *
                                        ;    *  COMMAND SEND - CSEND *
                                        ;    * * * * * * * * * * * * *
                                        ;
                                        ;
                                        ; Summary:
                                        ;      - Called by CMD to send interface (command) messages
                                        ;
                                        ; Assumptions on entry:
                                        ;      - The GPIB-SBX is Active Controller
                                        ;      - The bl(b) register contains the number of bytes
                                        ;        to send
                                        ;      - The si(hl) register pair contains the address
                                        ;        of cmdbuf
                                        ;
                                        ; Actions:
                                        ;      - Initialize a count variable
                                        ;      - Wait until the CDOR is empty
                                        ;      - Write a byte and increment the counter
                                        ;      - Check for a GPIB error
                                        ;      - Loop until all bytes are transfered
                                        ;      - On an error, set bl(b) to -1
                                        ;
                                        ; Status on return:
                                        ;      - bl(b) register contains number of bytes
                                        ;        sent or -1 if an error occurred
                                        ;
;     8088 Code                  8080 Code          ;                  Comments
;----------------------  ---------------------------- ; ----------------------------------------------
                                        ;
CSEND:  mov     count,0 ;CSEND: mvi     a,0            ; Initialize count variable
                        sta     count           ;
CSEND1: in      al,ISR2 ;CSEND1:in      ISR2           ; Wait till CDOR is empty
        and     al,CO   ;       ani     CO             ;
        jz      CSEND1  ;       jz      CSEND1         ;
        mov     al,count;       lda     count          ; Have all commands been sent?
        cmp     al,bl   ;       cmp     b              ;
        jz      CSEND3  ;       jz      CSEND3         ; Yes
                        ;       inr     a              ; No--Increment counter and write
        inc     count   ;       sta     count          ; the next command
        mov     al,byte ptr [si];mov    a,m            ;
        out     CDOR,al ;       out     CDOR           ;
        in      al,ISR1 ;       in      ISR1           ; If there are no Listeners, return -1
        and     ERR     ;       ani     ERR            ; in bl(b) register
        jnz     CSEND2  ;       jnz     CSEND2         ;
        inc     dx      ;       inx     h              ;
        jmp     CSEND1  ;       jmp     CSEND1         ;
CSEND2: mov     bl,377q ;CSEND2:mvi     b,377q         ;
CSEND3: ret             ;CSEND3:ret                    ;
```

```
                                        ;    * * * * * * * * *
                                        ;    * COMMAND - CMD *
                                        ;    * * * * * * * * *
                                        ;
                                        ; Summary:
                                        ;      - Send GPIB interface or command messages
                                        ;
                                        ; Assumptions on entry:
                                        ;      - The GPIB-SBX is Controller-In-Charge
                                        ;      - The commands to be sent are in cmdbuf
                                        ;      - The variable cmdct contains the number
                                        ;        of commands to be sent, which must be
                                        ;        less than 256
                                        ;      - Interruption of any data transfer in
                                        ;        progress is acceptable
                                        ;
                                        ; Actions:
                                        ;      - Issue TCA command to assert ATN in case the
                                        ;        GPIB-SBX is at standby
                                        ;      - Load the si(hl) register pair with the address of cmdbuf
                                        ;      - Load bl(b) with the number of commands
                                        ;      - Call CSEND to transmit the bytes
                                        ;
                                        ; Status on return:
                                        ;      -GPIB-SBX is Active Controller
                                        ;      -GPIB devices are programmed as implied by
                                        ;       command bytes
                                        ;
;    8088 Code                8080 Code       ;                Comments
;---------------------  ----------------------------- ; ----------------------------------------------
                                        ;
CMD:   mov    al,TCA  ;CMD:  mvi    a,TCA        ; Take control in case at standby
       out    AUXMR,al;      out    AUXMR        ;
       mov    si,OFFSET cmdbuf;                  ;
                      lxi    h,cmdbuf      ; Set up registers for CSEND call
       mov    bl,cmdct;      lda    cmdct
                  ;      mov    b,a
       call   CSEND  ;      call   CSEND
       ret           ;      ret
```

```
                                       ;  * * * * * * * * * * * * * *
                                       ;  *  PASS CONTROL - PASSC  *
                                       ;  * * * * * * * * * * * * * *
                                       ;
                                       ; Summary:
                                       ;      - Passes GPIB Controller-In-Charge status
                                       ;        to another device
                                       ;
                                       ; Assumptions on entry:
                                       ;      - The GPIB-SBX is Controller-In-Charge
                                       ;      - The primary GPIB address of the new controller
                                       ;        is placed in tctadr
                                       ;
                                       ; Actions:
                                       ;      - Send TCA command to take control in case the
                                       ;        GPIB-SBX is at standby
                                       ;      - Set up the command buffer and command count
                                       ;      - Call CMD to send the command bytes
                                       ;
                                       ; Status on return:
                                       ;      - The GPIB-SBX is Idle Controller
                                       ;
                                       ;
;    8088 Code                 8080 Code         ;                  Comments
;----------------------  ----------------------------  ; ----------------------------------------------
                                       ;
PASSC:  mov    al,TCA  ;PASSC: mvi     a,TCA           ; Take control in case at standby
        out    AUXMR,al;       out     AUXMR           ;
        mov    al,UNT  ;       mvi     a,UNT           ; Set up the command buffer
        mov    cmdbuf,al;      sta     cmdbuf          ;
        mov    al,UNL  ;       mvi     a,UNL           ;
        mov    cmdbuf+1,al;    sta     cmdbuf+1        ;
        mov    al,tctadr;      mvi     a,tctadr        ;
        mov    cmdbuf+2,al;    sta     cmdbuf+2        ;
        mov    al,TCT  ;       mvi     a,TCT           ; The GPIB-SBX automatically releases
        mov    cmdbuf+3,al;    sta     cmdbuf+3        ; control when TCT is sent
                               mvi     a,4             ;
        mov    cmdct,4;        sta     cmdct           ;
        call   CMD     ;       call    CMD             ;
        ret            ;       ret                     ;
```

Appendix C
Multiline Interface Messages

This appendix contains an interface message reference list, which describes the mnemonics and messages that correspond to the interface functions. These multiline interface messages are sent and received with ATN TRUE.

For more information on these messages, refer to the ANSI/IEEE Std 488.1-1987, *IEEE Standard Digital Interface for Programmable Instrumentation*.

Multiline Interface Messages

Hex	Oct	Dec	ASCII	Msg	Hex	Oct	Dec	ASCII	Msg
00	000	0	NUL		20	040	32	SP	MLA0
01	001	1	SOH	GTL	21	041	33	!	MLA1
02	002	2	STX		22	042	34	"	MLA2
03	003	3	ETX		23	043	35	#	MLA3
04	004	4	EOT	SDC	24	044	36	$	MLA4
05	005	5	ENQ	PPC	25	045	37	%	MLA5
06	006	6	ACK		26	046	38	&	MLA6
07	007	7	BEL		27	047	39	'	MLA7
08	010	8	BS	GET	28	050	40	(	MLA8
09	011	9	HT	TCT	29	051	41	)	MLA9
0A	012	10	LF		2A	052	42	*	MLA10
0B	013	11	VT		2B	053	43	+	MLA11
0C	014	12	FF		2C	054	44	,	MLA12
0D	015	13	CR		2D	055	45	-	MLA13
0E	016	14	SO		2E	056	46	.	MLA14
0F	017	15	SI		2F	057	47	/	MLA15
10	020	16	DLE		30	060	48	0	MLA16
11	021	17	DC1	LLO	31	061	49	1	MLA17
12	022	18	DC2		32	062	50	2	MLA18
13	023	19	DC3		33	063	51	3	MLA19
14	024	20	DC4	DCL	34	064	52	4	MLA20
15	025	21	NAK	PPU	35	065	53	5	MLA21
16	026	22	SYN		36	066	54	6	MLA22
17	027	23	ETB		37	067	55	7	MLA23
18	030	24	CAN	SPE	38	070	56	8	MLA24
19	031	25	EM	SPD	39	071	57	9	MLA25
1A	032	26	SUB		3A	072	58	:	MLA26
1B	033	27	ESC		3B	073	59	;	MLA27
1C	034	28	FS		3C	074	60	<	MLA28
1D	035	29	GS		3D	075	61	=	MLA29
1E	036	30	RS		3E	076	62	>	MLA30
1F	037	31	US		3F	077	63	?	UNL

Message Definitions

DCL	Device Clear
GET	Group Execute Trigger
GTL	Go To Local
LLO	Local Lockout
MLA	My Listen Address
MSA	My Secondary Address
MTA	My Talk Address
PPC	Parallel Poll Configure
PPD	Parallel Poll Disable

Multiline Interface Messages

Hex	Oct	Dec	ASCII	Msg	Hex	Oct	Dec	ASCII	Msg
40	100	64	@	MTA0	60	140	96	`	MSA0,PPE
41	101	65	A	MTA1	61	141	97	a	MSA1,PPE
42	102	66	B	MTA2	62	142	98	b	MSA2,PPE
43	103	67	C	MTA3	63	143	99	c	MSA3,PPE
44	104	68	D	MTA4	64	144	100	d	MSA4,PPE
45	105	69	E	MTA5	65	145	101	e	MSA5,PPE
46	106	70	F	MTA6	66	146	102	f	MSA6,PPE
47	107	71	G	MTA7	67	147	103	g	MSA7,PPE
48	110	72	H	MTA8	68	150	104	h	MSA8,PPE
49	111	73	I	MTA9	69	151	105	i	MSA9,PPE
4A	112	74	J	MTA10	6A	152	106	j	MSA10,PPE
4B	113	75	K	MTA11	6B	153	107	k	MSA11,PPE
4C	114	76	L	MTA12	6C	154	108	l	MSA12,PPE
4D	115	77	M	MTA13	6D	155	109	m	MSA13,PPE
4E	116	78	N	MTA14	6E156	110	n	MSA14,PPE	
4F	117	79	O	MTA15	6F	157	111	o	MSA15,PPE
50	120	80	P	MTA16	70	160	112	p	MSA16,PPD
51	121	81	Q	MTA17	71	161	113	q	MSA17,PPD
52	122	82	R	MTA18	72	162	114	r	MSA18,PPD
53	123	83	S	MTA19	73	163	115	s	MSA19,PPD
54	124	84	T	MTA20	74	164	116	t	MSA20,PPD
55	125	85	U	MTA21	75	165	117	u	MSA21,PPD
56	126	86	V	MTA22	76	166	118	v	MSA22,PPD
57	127	87	W	MTA23	77	167	119	w	MSA23,PPD
58	130	88	X	MTA24	78	170	120	x	MSA24,PPD
59	131	89	Y	MTA25	79	171	121	y	MSA25,PPD
5A	132	90	Z	MTA26	7A	172	122	z	MSA26,PPD
5B	133	91	[	MTA27	7B	173	123	{	MSA27,PPD
5C	134	92	\	MTA28	7C	174	124	|	MSA28,PPD
5D	135	93	]	MTA29	7D	175	125	}	MSA29,PPD
5E	136	94	^	MTA30	7E	176	126	~	MSA30,PPD
5F	137	95	_	UNT	7F	177	127	DEL	

PPE	Parallel Poll Enable	SPE	Serial Poll Enable
PPU	Parallel Poll Unconfigure	TCT	Take Control
SDC	Selected Device Clear	UNL	Unlisten
SPD	Serial Poll Disable	UNT	Untalk

Appendix D Mnemonics Key

This appendix contains a mnemonics key that defines the mnemonics (abbreviations) used throughout this manual for functions, remote messages, local messages, states, bits, registers, integrated circuits, and SBX bus signals.

The mnemonic types in the key that follows are abbreviated to mean the following:

B	Bit
F	Function
IC	Integrated Circuit
LM	Local Message
R	Register
RM	Remote Message
ST	State
SX	SBX Bus Signal

Mnemonic	Type	Definition
A		
ACDS	ST	Acceptor Data State (AH function)
ACG	RM	Addressed Command Group
ACRS	ST	Acceptor Ready State
AD[5-0—1-0]	B	Mode 2 Primary TLC GPIB Address Bits 5 through 1
AD[5-1—1-1]	B	Mode 2 Secondary TLC GPIB Address Bits 5 through 1
ADM[1-0]	B	Address Mode Bits 1 through 0
ADMR	R	Address Mode Register
ADR	R	Address Register
ADR0	R	Address Register 0
ADR1	R	Address Register 1
ADSC	B	Addressed Status Change Bit
ADSC IE	B	Enable Interrupt on Addressed Status Change Bit
ADSR	R	Address Status Register
AH	ST	Acceptor Handshake
AIDS	ST	Acceptor Idle State
ANRS	ST	Acceptor Not Ready State
APRS	ST	Affirmative Poll Response State
APT	B	Address Pass Through Bit
APT IE	B	Enable Interrupt on Address Pass Through Bit
ARS	B	Address Register Select Bit
ATN	ST	Attention
ATN*	B	Attention Bit
AUXMR	R	Auxiliary Mode Register
AUXRA	R	Auxiliary Register A
AUXRB	R	Auxiliary Register B
AUXRE	R	Auxiliary Register E
AWNS	ST	Acceptor Wait for New Cycle State
B		
BIN	B	Binary Bit
C		
C	F	Controller
CACS	ST	Controller Active State (C function)
CADS	ST	Controller Addressed State
CAWS	ST	Controller Active Wait State
CDOR	R	Control/Data Out Register
CDO[7-0]	B	Control/Data Out Bits 7 through 0
CIC	B	Controller-In-Charge Bit
CIDS	ST	Controller Idle State
CLK[3-0]	B	Clock Bits 3 through 0
CNT[2-0]	B	Control Code Bits 2 through 0
CO	B	Command Output Bit
CO IE	B	Enable Interrupt on Command Output Bit

Mnemonic	Type	Definition
COM[4-0]	B	Command Code Bits 4 through 0
CPPS	ST	Controller Parallel Poll State
CPT	B	Command Pass Through Bit
CPT ENABLE	B	Command Pass Through Enable Bit
CPT IE	B	Enable Interrupt on Command Pass Through Bit
CPTR	R	Command Pass Through Register
CPT[7-0]	B	Command Pass Through Bits 7 through 0
CPWS	ST	Controller Parallel Poll Wait State
CSBS	ST	Controller Standby State
CSHS	ST	Controller Standby Hold State
CSNS	ST	Controller Service Not Requested State
CSRS	ST	Controller Service Requested State
CSWS	ST	Controller Synchronous Wait State
CTRS	ST	Controller Transfer State (C function)

D

Mnemonic	Type	Definition
DAB	RM	Data Byte
DAC	RM	Data Accepted
dacr		DAC holdoff release
DAG	RM	Device Address Group
DAV	B	GPIB Data Valid Signal Bit
DAV	RM	Data Valid
DC	F	Device Clear
DCAS	ST	Device Clear Active State
DCIS	ST	Device Clear Idle State
DCL	RM	Device Clear
DEC	B	Device Clear Bit
DEC IE	B	Enable Interrupt on Device Clear Bit
DET	B	Device Execute Trigger Bit
DET IE	B	Enable Interrupt on Device Execute Trigger Bit
DHDC	B	Data Accepted Holdoff on Device Clear Active State Bit
DHDT	B	Data Accepted Holdoff on Device Trigger Active State Bit
DI	B	Data In Bit
DI[7-0]	B	Data In Bits 7 through 0
DI IE	B	Enable Interrupt on Data In Bit
DIO[8-1]	B	GPIB Data Drive and Read Register Bits 8 through 1
DIR	R	Data In Register
DL	B	Disable Listener Bit
DL0	B	Disable Listener 0 Bit
DL1	B	Disable Listener 1 Bit
DMAI	B	Direct Memory Access Input Enable Bit
DMAO	B	Direct Memory Access Output Enable Bit
DO	B	Data Out Bit
DO IE	B	Enable Interrupt on Data Out Bit
DT	F	Device Trigger
DT	B	Disable Talker Bit
DT0	B	Disable Talker 0 Bit

Mnemonic	Type	Definition
DT1	B	Disable Talker 1 Bit
DTAS	ST	Device Trigger Active State
DTIS	ST	Device Trigger Idle State

E

Mnemonic	Type	Definition
END	RM	End
END IE	B	Enable Interrupt on End Received Bit
END RX	B	End Received Bit
EOI	B	End or Identify Bit
EOI	RM	End or Identify
EOS	RM	End of String
EOS[7-0]	B	End of String Bits 7 through 0
EOSR	R	End of String Register
ERR	B	Error Bit
ERR	RM	Error
ERR IE	B	Enable Interrupt on Error Bit

F

Mnemonic	Type	Definition
FH	RM	Finish Handshake

G

Mnemonic	Type	Definition
GET	RM	Group Execute Trigger
GND	SX	Ground
GTL	RM	Go To Local
gts	LM	Go to Standby

H

Mnemonic	Type	Definition
HLDA	B	Holdoff on All Bit
HLDE	B	Holdoff on End Bit

I

Mnemonic	Type	Definition
ICR	R	Internal Count Register
IDY	RM	Identify
IFC	RM	Interface Clear
IMR1	R	Interrupt Mask Register 1
IMR2	R	Interrupt Mask Register 2
INT	B	Interrupt Bit
INV	B	Invert Bit
IORD*	SX	I/O Read

Mnemonic	Type	Definition
IOWRT*	SX	I/O Write
ISR1	R	Interrupt Status Register 1
ISR2	R	Interrupt Status Register 2
ISS	B	Individual Status Select Bit
ist	LM	Individual Status

L

L	F	Listen
LA	B	Listener Active Bit
LACS	ST	Listener Active State (L function)
LADS	ST	Listener Addressed State (L function)
LAG	RM	Listener Address Group
LE	F	Listener Extended
LIDS	ST	Listener Idle State
LLO	RM	Local Lockout
LOCS	ST	Local State
LOK	B	Lockout Bit
LOKC	B	Lockout Change Bit
LOKC IE	B	Enable Interrupt on Lockout Change Bit
lon	B	Listen Only Bit
lon	LM	Listen Only
LPAS	B	Listener Primary Addressed State Bit
LPAS	ST	Listener Primary Addressed State
lpe	LM	Local Poll Enabled
LPIS	ST	Listener Primary Idle State
ltn	LM	Listen
lun	LM	Local Unlisten
LWLS	ST	Local With Lockout State

M

MA[2-0]	SX	Address Lines
MCLK*	SX	SBX Clock
MCSOA*	SX	Chip Select 0, Connector A
MCSOB*	SX	Chip Select 0, Connector B
MCSOC*	SX	Chip Select 0, Connector C
MCS1A*	SX	Chip Select 1, Connector A
MCS1B*	SX	Chip Select 1, Connector B
MCS1C*	SX	Chip Select 1, Connector C
MD[7-1]	SX	Data Lines
MDACK*	SX	DMA Acknowledge
MDRQT	SX	DMA Request
MINTRA0	SX	Interrupt 0, Connector A
MINTRA1	SX	Interrupt 1, Connector A
MINTRB0	SX	Interrupt 0, Connector B
MINTRB1	SX	Interrupt 1, Connector B

Mnemonic	Type	Definition
MINTRC0	SX	Interrupt 0, Connector C
MINTRC1	SX	Interrupt 1, Connector C
MJMN	B	Major-Minor Bit
MLA	RM	My Listen Address
MPST*	SX	Present
MRESET	SX	Reset
MSA	RM	My Secondary Address
MTA	RM	My Talk Address
MWAIT	SX	Wait

N

nba	LM	New Byte Available
NDAC	RM	Not Data Accepted
NPRS	ST	Negative Poll Response State
NRFD	RM	Not Ready For Data
NUL	RM	Null byte

O

OPT[1-0]	SX	Option 1, 0
OSA	RM	Other Secondary Address
OTA	RM	Other Talk Address

P

P[3-1]	B	Parallel Poll Response Bits 3 through 1
PACS	ST	Parallel Poll Addressed to Configure State
PCG	RM	Primary Command Group
PEND	B	Pending Bit
pof	LM	Power Off
pon	LM	Power On
PP	F	Parallel Poll (scan all status flags)
PPAS	ST	Parallel Poll Active State
PPC	RM	Parallel Poll Configure
PPD	RM	Parallel Poll Disable
PPE	RM	Parallel Poll Enable
PPIS	ST	Parallel Poll Idle State
PPR	RM	Parallel Poll Response
PPR	R	Parallel Poll Register
PPSS	ST	Parallel Poll Standby Active
PPU	RM	Parallel Poll Unconfigure
PUCS	ST	Parallel Poll Unaddressed to Configure State

Mnemonic	Type	Definition
R		
rdy	LM	Ready for next message
REM	B	Remote Bit
REMC	B	Remote Change Bit
REMC IE	B	Enable Interrupt on Remote Change Bit
REMS	ST	Remote State
REN	RM	Remote Enable
REOS	B	End on End Of String Received Bit
RFD	RM	Ready For Data
RL	F	Remote/Local
rpp	LM	Request Parallel Poll
RQS	RM	Request Service
rsc	LM	Request System Control
rsv	B	Request Service Bit
rsv	LM	Request Service
rtl	LM	Return To Local
RWLS	ST	Remote With Lockout State
S		
S	B	Status Bit Polarity (Sense) Bit
SACS	ST	System Control Active State
SCG	RM	Secondary Command Group
SDC	RM	Selected Device Clear
SDYS	ST	Source Delay State
SEOI	RM	Send EOI
SGNS	ST	Source Generate State
SH	F	Source Handshake
SIAS	ST	System Control Interface Clear Active State
sic	LM	Send Interface Clear
SIDS	ST	Source Idle State
SIIS	ST	System Control Interface Clear Idle State
SINS	ST	System Control Interface Clear Not Active State
SIWS	ST	Source Idle Wait State
SNAS	ST	System Control Not Active State
SP	F	Serial Poll (scanning flags)
SPAS	ST	Serial Poll Active State (T function)
SPD	RM	Serial Poll Disable
SPE	RM	Serial Poll Enable
SPEOI	B	Send Serial Poll End Or Identify Bit
SPIS	ST	Serial Poll Idle State
SPMR	R	Serial Poll Mode Register
SPMS	B	Serial Poll Mode State Bit
SPMS	ST	Serial Poll Mode State
SPSR	R	Serial Poll Status Register
SR	F	Service Request
SRAS	ST	System Control Remote Enable Active State

Mnemonic	Type	Definition
sre	LM	Send Remote Enable
SRIS	ST	System Control Remote Enable Idle State
SRNS	ST	System Control Remote Enable Not Active State
SRQ	RM	Service Request
SRQI	B	Service Request Input Bit
SRQI IE	B	Enable Interrupt on Service Request Input Bit
SRQS	ST	Service Request State
STB	RM	Status Byte
STRS	ST	Source Transfer State
SWNS	ST	Source Wait for New Cycle State

T

T	F	Talker
TA	B	Talker Active Bit
TACS	ST	Talker Active State (T function)
TADS	ST	Talker Addressed State
TAG	RM	Talk Address Group
tca	LM	Talk Control Asynchronously
tcs	LM	Take Control Synchronously
TCT	RM	Take Control
TDMA	SX	Terminate DMA
TE	F	Extended Talk
TIDS	ST	Talker Idle State
TLC	IC	Talker/Listener/Controller (GPIB Adapter)
ton	B	Talker Only Bit
ton	LM	Talker Only
TPAS	B	Talker Primary Addressed State Bit
TPAS	ST	Talker Primary Addressed State
TPIS	ST	Talker Primary Idle State
TRI	B	Three-State Timing Bit
TRM[1-0]	B	Transmit/Receive Mode Bits 1 through 0

U

U	B	Unconfigure Bit
UCG	RM	Universal Command Group
UDPCF	F	Undefined Primary Command
UNL	RM	Unlisten command
UNT	RM	Untalk command

X

X	B	Don't Care Bit
XEOS	B	Transmit End with End Of String Bit

User Comment Form

National Instruments encourages you to give us your comments on the documentation supplied with its products. This information helps us provide quality products to meet your needs.

Title: **GPIB-SBX User Manual**

Edition Date: **January 1996**

Part Number: **320015-01**

Please comment on the completeness, clarity, and organization of the manual.

____

____

____

____

____

____

If you find errors in the manual, please record the page numbers and describe the errors.

____

____

____

____

____

____

Thank you for your help.

Name ____

Title ____

Company ____

Address ____

____

Phone (_______)_______________________________

Mail to: Technical Publications
National Instruments Corporation
6504 Bridge Point Parkway
Austin, TX 78730-5039

Index