Operator's Manual
SD385 NOMAD Portable Signal Analyzer
Appendix - Part One

Legacy Manual

COGNITIVE VISION, INC.
7220 Trade Street, Suite 101
San Diego, CA 92121-2325 USA

analyzers@cognitivevision.com
www.cognitivevision.com

Tel: 1.858.578.3778 / Fax: 1.858.578.2778
In USA: 1.800.VIB.TEST (842.8378)

LEGACY MANUAL POLICY

Cognitive Vision Legacy manuals are those product manuals and documentation that accompanied earlier products and product lines which have since been discontinued ("Legacy Products"). Over the past thirty years, these include products that were sold by Spectral Dynamics, Scientific Atlanta and Smiths Industries.

Cognitive Vision, Inc. provides downloadable copies of these manuals strictly as a courtesy to its customers who continue to use Legacy Products. IMPORTANT: Please read the following Disclaimer carefully. Any use of this manual indicates your express agreement with this policy.

If you have any questions regarding this policy, or for additional information regarding the serviceability of any Legacy Product(s), please call our service department.

DISCLAIMER

IN DOWNLOADING THIS MANUAL, THE USER UNDERSTANDS AND EXPRESSLY AGREES THAT COGNITIVE VISION MAKES NO WARRANTIES WHATSOEVER, EITHER EXPRESS OR IMPLIED, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN USING THIS MANUAL, THE USER ACKNOWLEDGES THAT ALL PREVIOUS PRODUCT WARRANTIES ISSUED BY SPECTRAL DYNAMICS, SCIENTIFIC ATLANTA AND SMITHS INDUSTRIES FOR LEGACY PRODUCTS HAVE SINCE EXPIRED.

IN PROVIDING THIS MANUAL, COGNITIVE VISION ASSUMES NO LIABILITY OR RESPONSIBILITY WHATSOEVER TO THE USER OF THIS MANUAL, THE USER'S AGENTS AND/OR CUSTOMERS, OR ANY OTHER PARTY, FOR ANY CLAIMED INACCURACY IN THIS MANUAL, OR FOR DAMAGE CAUSED OR ALLEGED TO BE CAUSED DIRECTLY OR INDIRECTLY BY ANY USE OF THIS MANUAL, REGARDLESS OF WHETHER COGNITIVE VISION WAS INFORMED ABOUT THE POSSIBIILITY OF SUCH DAMAGES, OR FOR ANY CLAIM MADE AGAINST THE USER'S ORIGINAL PRODUCT WARRANTY.

FURTHER, COGNITIVE VISION SHALL NOT BE RESPONSIBLE FOR ANY INTERRUPTION OF SERVICE, LOSS OF BUSINESS, ANTICIPATORY PROFITS, CONSEQUENTIAL DAMAGES, OR INDIRECT OR SPECIAL DAMAGES ARISING UNDER ANY CIRCUMSTANCES, OR FROM ANY CAUSE OF ACTION WHATSOEVER INCLUDING CONTRACT, WARRANTY, STRICT LIABILITY OR NEGLIGENCE.

NOTWITHSTANDING THE ABOVE, IN NO EVENT SHALL COGNITIVE VISION'S LIABILITY TO THE USER EXCEED AN AMOUNT EQUAL TO THE REPLACEMENT COST OF THIS MANUAL.

COGNITIVE VISION, INC.
7220 Trade Street, Suite 101
San Diego, CA 92121-2325 USA

analyzers@cognitivevision.com
www.cognitivevision.com

Telephone: 1.858.578.3778 / Fax: 1.858.578.2778
IN USA: 1.800.VIB.TEST (842.8378)

APPENDIX A

SD385 IEEE-488(78) (GPIB) APPENDIX.

This appendix documents the functions, performance, and use of the SD385 GPIB interface.

A-1 DESCRIPTION OF THE FEATURES

This appendix will describe the SD385 IEEE-488 feature. "Formal" function definitions (per the standard) and understandable explanations will be used. An overview description of the interface and market-place implementations is included. The appendix concludes with a more detailed description of the SD385 implementation of the IEEE-488 interface.

A-1.1 SD385 IEEE-488 FUNCTIONS IMPLEMENTED

Description of the Features

The SD385 IEEE-488 interface (or GPIB) is a standard feature included on all SD385s. The performance of the GPIB is not affected by the installation of options.

This implementation of the IEEE-488 is a standard talker-listener with Service Request and Serial Poll capability, and the ability to be configured in a "talk-only" mode (primarily for digital plotting without the expense of a controller).

This feature gives the user with an IEEE capable computer the ability to remotely control the instrument and read analysis results from it for processing. Users with IEEE interface HP-GL plotters can produce quality hard copy using the IEEE and the SD385's built-in plotter capability.

The inclusion of the Service Request (SRQ) capability allows the instrument to request attention from the computer upon events the computer identifies as being important. Serial Poll capability allows a computer to identify the requesting device and provides some information about the cause of the SRQ.

Exact Functions Defined per Standard

The SD385 implements the following functions per Standard IEEE-488-1978: "IEEE STANDARD DIGITAL INTERFACE FOR PROGRAMMABLE INSTRUMENTATION"

- SH1	Complete source handshake
- AH1	Complete acceptor handshake
- T5	Basic talker with Serial Poll, talk only capability, unaddress if MLA
- L4	Basic listener, no listen only, unaddress if MTA
- SR1	Complete Service Request capability
- RL0	No remote/local capability
- PP0	No parallel poll capability
- DC1	Complete Device Clear capability
- DT0	No device trigger capability
- C0	No controller capability

Messages Sent (per standard).
The SD385 sends the following uni-line messages:

DAC - data accepted (when listening or ATN is true)

RFD - ready for data (when listening or ATN is true)

DAV - Data Valid (when talking)

END - end of data (when talking and output terminator setting is EOI)

SRQ - Service Request

The SD385 sends the following multi-line messages:

DAB - data byte

STB - Serial Poll status byte

RQS - request service (bit 6 of STB)

SD385 messages received (per standard)

The SD385 receives (and acknowledges) the following uni-line messages:

DAC - data accepted (when talking)

RFD - ready for data (when talking)

DAV - Data Valid (when listening or ATN is true)

END - end of data (when listening and input terminator setting is EOI)

ATN - attention (asserted by controller-in-charge)

IFC - interface clear

The SD385 receives (and acknowledges) the following multi-line messages:

DCL - universal device clear

SPE - Serial Poll enable

SPD - Serial Poll disable

OTA - other talk address

UNL - unlisten

MLA - my listen address (x01ppppp; ppppp = SD385 address)

MTA - my talk address (x10ppppp)

SDC - selective device clear (clears only listening devices)

A-1.2 Overview of the IEEE-488 Interface.

Description

The IEEE-488 interface, formally defined in 1975, is a bit-parallel, byte serial interface. The definition allows asynchronous data exchange, multiple listeners, and several special protocols. More than one device can be connected to the "bus". The current "controller-in-charge" directs all activities on the bus.

Electrically, each device on the bus will implement bi-directional drivers for the 8 data lines, 3 byte-handshake lines and some of the remaining (bus "management") lines. All of the devices can "float" the data/handshake lines when they are not participating in bus transactions. Because of this, the 488 interface is described as a "party-line bus system". This makes it possible to connect several devices to the bus without non-participants in an activity interfering. The advantage to this is obvious: a controller-computer need be provided with only one hardware option (an IEEE interface) to control and use data from several devices.

Users may be aware of 1975, 1978, and 1980 dates used on IEEE-488 interface products in the market-place. The original specification, published in 1975, was revised in 1978. These revisions were not major and should not normally affect the compatibility of two 488 devices. The only anomoly noted to date is that some '78 implementations may fail to properly receive the "END " message from some '75 implementations. The wide usage of LSI and SSI components has eliminated most of the compatibility problems encountered in the past. These were due not to any '75/'78 incompatibilities, but rather to differences in interpretation by individual designers in their efforts to design 488 interfaces with discrete components.

In 1980 the IEEE standards committee issued a supplement to the 1978 standard whose only affect was to clarify some points on the controller function. There is no 1980 IEEE standard for end users to be concerned about.

Besides the pin-out and electrical and mechanical connector specifications, the 488 standard specifies all command messages to be sent by the controller and proper responses on the part of the controlled talkers, listeners, and talker/listeners. Most controllers on the marketplace today implement language extentions and/or accessory software packages which relieve the programmer from a need to know the exact format of these defined command messages. For the convenience of the designer whose controller is not so equipped, IEEE definition of messages and bus sequences are listed in the Summary tables located at the end of this appendix.

Bus Lines and Their Usage

Figure A-1 is a descriptive block diagram of the IEEE-488 interface bus.

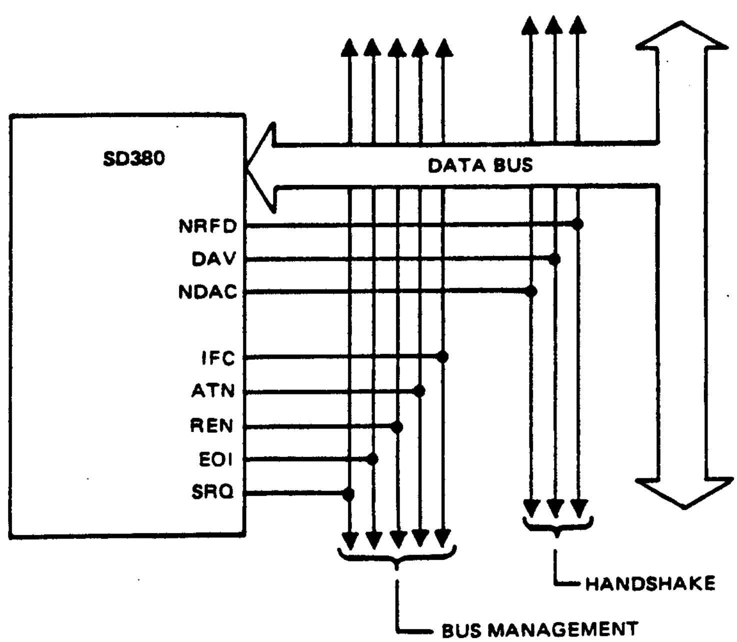


Figure A-1. IEEE-488 Bus Interface

The 16 active bus lines of the interface are divided into three groups:

8-bit DATA BUS. Used to send commands and data from outputting devices to inputting devices. Controller asserts bus-management line ATN to send commands out on this bus.

3-wire HANDSHAKE lines. These lines are used for the "3-wire handshake". The talker asserts Data Valid (DAV) after observing that all listeners have allowed NRFD to float HIGH (e.g all are RFD). The talker then places data on the bus and asserts DAV. Data and DAV remain until all listeners have allowed NDAC to float HIGH (e.g. all have accepted data) then the cycle starts again. In the manner, the bus handles both asynchronous and multiple listeners.

NOTE

The primary use of multiple listeners is for all devices to receive controller commands, since this is usually the only time that all devices on the bus will understand the message sent.

5 BUS-MANAGEMENT lines

ATN - Asserted by the controller to put all other devices into a mode to receive commands.

SRQ - Asserted by the controlled devices to request service from the controller.

IFC - Asserted by the System Controller to take control of the bus under any circumstance.

REN - Remote/Local (RL) enable asserted by System Controller to place RL capable devices into remote.

EOI - End or Identify (EOI) asserted by current talker as end of data message, or asserted synchronously with ATN by controller to assert the identify message for parallel poll.

Each device on the bus must have a switch which can be used to determine the address of the device. Before a bus system can work, the switches must be set so that each device has a unique address somewhere between 0 and 30 (inclusive). Controller software must know which device is at which address.

The controller then directs activities on the bus via addressed commands (the address of the device is included in the command) and universal commands (all devices respond). The MTA command is an addressed command with bit pattern X10PPPPP where "X" is irrelevant, and "PPPPP" is the address of the device commanded, in this case to talk. Device Clear (DCL) with a bit pattern of X0011000, or 24 decimal, tells <u>all</u> devices on the bus to enter the device-defined clear state.

Bus activities of interest will be sending data to a device (address it to listen and output the data), getting data from a device (address it to talk and input the data), sending special IEEE commands such as Device Clear (usually to put the device into a known state), responding to SRQs and conducting a Serial Poll of devices to see which one sent an SRQ. These are all activities precisely defined by the standard.

The remainder of what happens to the devices and what the controller has to do is determined by the Device Defined meanings of the data bytes that are transferred in and out of a device.

Input and Output Schemes

The IEEE standard does not define what data byte DAB will be. Each device implemented on the interface will have its own protocol for the meaning of the bytes it inputs and outputs.

ASCII/BINARY

Each byte transferred across the interface will consist of eight bits. The bit pattern:

01000001

can be interpreted as either the character "A", as the numerical value 65 (decimal), or as simply a pattern of bits.

A good example of the difference between ASCII and BINARY expressions is the difference between sending the number 10 as 16-bit word and as a 4-character expression.

The bit patterns seen would be as follows:

Binary (low-byte-first)

00001010
00000000

Binary (high-byte-first)

00000000
00001010

ASCII (space leader)

00100000	space
00100000	space
00110001	character 1
00110000	character 0

ASCII (zero leader)

00110000	character 0
00110000	character 0
00110001	character 1
00110000	character 0

One of the more important variations on the IEEE interface is whether or not the device designers chose to treat the data bus as an 8-bit parallel numerical encoded interface, as a character interface, or some mixture of both. This decision will determine the meaning of the information encoded in the bytes sent to and received from a device.

Another important aspect of the device implementation will be data identification. Many devices have more that one type of information to input and output. Data stream identification may be by numbers (first byte is a number identifying the data to follow) or by mnemonics (character identification). This applies to both input and output streams.

Finally, there is the determination of what constitutes the end of a stream of data. Some applications will incorporate a special character (carriage return, semi-colon, line feed, etc), some will use fixed length data streams (byte count determines the end), and others will use the END message of the IEEE interface (EOI asserted by the talker).

Controllers are frequently provided with capabilities to read bytes as binary values or as characters, to determine transfer characteristics such as end on count, character, EOI, etc. Special data handling instructions will be incorporated to allow the programmer to handle data with bit-mapped, numerical or ASCII information encoding.

Service Request and Serial Poll

Service Request (SRQ) and Serial Poll are linked capabilities on the 488 interface. SRQ allows a non-controlling device on the interface to request service from the controller. This bus management line is output by all devices except the controller-in-charge, which inputs it. The line is a "float-false" line, meaning any one, all or any combination of devices can assert the signal. Once the controller senses and acknowledges an SRQ, it must choose its response. One of the factors determining the response will of course be which device(s) asserted the SRQ.

Upon execution of a Serial Poll, the controller will assert ATN, pass the SERIAL POLL ENABLE message across the bus, and sequentially address various devices to talk. Each device, as it is addressed, will output the STB (Serial Poll STATUS BYTE). Bit six in this byte is reserved for the RQS (Request Service) message. When bit six is asserted in the STB, the polled device is asserting SRQ. Once polled, the device releases the SRQ line. This allows the controller to stop polling when it sees SRQ go false, since that would indicate that all requesting devices have been polled. The remaining bits in the STB are device defined and are frequently used to identify the cause of the SRQ.

A-1.3 SD385 Implementation of the IEEE

Input and Output Procedures

The SD385 incorporates an ASCII mnemonic scheme for input/output data identification. Each stream of data will be preceded by a six character mnemonic identifying the data that is being sent. Mnemonics ending with a question mark (?) identify the data that should be output by the SD385 the next time it is addressed to talk. The following is an example:

> "TIME 08:20:00" sent to the SD385 will set the time-of-day to 8:20 AM.
>
> "TIME? " sent to the SD385 will cause the SD385 to output the time-of-day when it is addressed to talk. If addressed to talk again, it will output the time-of-day again, until some other data is identified.

For example if the time were 10:22:14, the SD385 would output:

> "TIME 10:22:14"

This duplication of the mnemonic in the output stream allows the listener to identify data being received. The exclusion of the question mark makes the data suitable "as is" to be sent back to the SD385 as input data. Though not particularly useful for a time-of-day example, this technique will be valuable for other kinds of data.

The SD385 will handshake, but ignore, any data bytes sent after the last data byte it expects in a stream. This helps to prevent interface lockups due to more data than the SD385 expects to receive. In order for the SD385 to read bytes as being part of another data-identification mnemonic, it must be addressed again.

Example: let's make up a syntax that outputs to the SD385. The statement:

```
OUTPUT @SD385; "......."
```

causes the controller to address the SD385 (send it its MLA, drop ATN and proceed to send the "......" characters)

The statement:

```
OUTPUT @SD385; "TIME  01:01:01 TIME 02:02:02"
```

would result in a time-of-day of 01:01:01, since the then second set of characters would be handshaked but ignored.

The statements:

```
OUTPUT @SD385; "TIME  01:01:01"
OUTPUT @SD385; "TIME  02:02:02"
```

would result in a time-of-day of 02:02:02 since the second set of characters was preceded with an address command. The first example output the second set of characters as a continuing part of the data stream without an address command in front of them.

There are 66 different mnemonics identifying 66 different kinds of data to input to the SD385 and 74 mnemonics identifying 74 kinds of data to be output. However, most of the input mnemonics are matched by an output mnemonic that is identical except for the substitution of a question mark (?) for a space in the mnemonic. Many of the mnemonics are variations on a two character expression with A or B, or 1 or 2 added since this is a dual-trace instrument that can also have two channels.

The net result is that there are 32 kinds of data handled by the SD385's GPIB. These 32 kinds of data will from now on be referred to as "data groups".

Data Groups

The 32 data groups are subdivided in the documentation into four main categories: Configuration Command, Control Operations, Ensemble and Numeric Output. The Configuration Command and Control Operations data groups give the GPIB programmer the ability to configure, operate and read operating status information from the instrument.

The Numeric Output data groups allow the user to read the measurement-result numbers from the instrument. The Ensemble data groups allow the user to read from and write to the large array memories of the instrument that hold the source acquired and processed information taken by the instrument.

Data Formats and Terminators

The problem with data encoding is, as previously mentioned, the bit-map, byte value, ASCII trade-off. The GBIB on the SD385 incorporates all three. Shorter input commands will be in ASCII for simplicity and to accomodate readability of user programs. All but one of the short-output groups will be ASCII as well. This one short-output group is a bit-map status word that needs to be as quick as possible.

Binary data is not delimited, as it used on a byte basis. Termination of input and output data streams wlll occur on byte count or EOI; no termination characters are used. The first twelve bytes of each binary data block has a similar format to allow for some general decoding. Bytes 1-6 are the command echo, bytes 7-8 are the number of data bytes sent and bytes 9-12 are currently spare.

ASCII data is delimited by commas. Spacing for input commands is not fixed (free format); output spacing is fixed. Numbers must be unsigned integers (no radix). Nulls, leading and trailing spaces and parity are ignored. Lower case letters are converted to upper case. Either a termination character or EOI is used.

Binary values will be: byte, word (two bytes), or floating point (four bytes). These will occur in large arrays where ASCII data encoding would require much larger data buffers (with a 6000 point, floating point array, ASCII encoding would require 48000 bytes of buffer space, vice 24000 for the binary encoding.

Conventions: BYTES - Express selections out of a list. A numerical encoding of a choice.

WORDS - For numerical values which are basically integers.

FLOATING POINT - Used for values which have a large range and resolution.

ASCII - Used when appropriate for user convenience. Expresses any kind of numerical value, or the byte-choice type of code (as a number: in ASCII there is no visible difference between a byte-choice and a word except for permissable range). Used for character text.

In all cases where ASCII is used to encode a number, the output will be Fixed Field; i.e., the characters which express the number will always occur in the same place in the output stream. The input is more flexible, the GPIB scans the data looking for field delimiters.

ASCII "SWITCH" - These are a special form of ASCII information used for what would be bit-mapped information in binary. The only characters permitted are: one (1), zero (0) and asterisk (*). Input streams which should be of this type will be rejected (not processed for intended function) if any other value occurs in the stream.

PACKED - some data streams will be "packed"; i.e., combinations of byte and ASCII values, or byte, word, and floating point values. This will be necessary when the application of the stream requires a variety of information.

Data Termination.

When a data stream is sent to a GPIB device, one of the important considerations is how the device will know when the stream has ended. It is at that point, usually, that the device will begin to process the information encoded. The same is true for the controlling computer. Typical program statements first input the data then do something with it. The input statement must finish execution before the "do something with it" process statement(s) can be executed. If the program "hangs up" because the input statement never executed, it is usually because the sending device and the computer's interpretation of the input statement disagree about what constitutes a finished input; i.e, the input program is waiting for data that the sending device will never send out.

A typical example would be a computer input structure that expects a CR/LF (carriage return/line feed) at the end of a stream to indicate the end, and a sending device that sends only the carriage return. The input statement hangs up waiting for the line-feed that will never come, even though every other character in the data stream has been loaded.

Input and output terminators are used when the data is exclusively ASCII. They become important when the length is variant (e.g., it is possible to send 10 or 100 and the terminator <u>must</u> be used to tell the GPIB the data stream is finished).

Due to the high usage of PC computers as word processors, many manufacturers incorporate word processing functions into the basic operation of the computer. Since word processing is the creation of lines and sentences on a printed page, the data streams are lines, or paragraphs of lines, terminated with a carriage return (CR) or a carriage return/line feed (CR/LF). Because of this, the PC's default terminator may be an ASCII CR or CR/LF. An ASCII data stream being received from the SD385 might contain CR or CR/LF characters. The PC, upon receipt of these characters, assumes the data is terminated and halts the data transfer.

To prevent this from occuring, the programmer must insure the data does not get halted by the default termination character. This may be accomplished by implementing EOI as the termination character.

Two settings in the IEEE Setup Page govern the ASCII terminations used by the SD385 GPIB. These are input and output terminators. Via these selections, you can determine what sort of terminator the SD385 will input or output: EOI (accept byte with EOI set as last byte), CR (accept carriage return as last byte), LF (accept line-feed as last byte) or ETX (accept end-of-text as last byte). EOI is <u>always</u> sent on the output.

Some of the data streams that the SD385 will input or output will be fixed block/binary streams (especially the ensemble data groups). Since binary values could easily contain bit patterns that match the bit patterns for CR, LF and ETX, these data groups do not use the terminator setting. Instead, they run completely off byte count.

Another aspect of Data Format that is important is how bytes are used to encode floating point information. Binary-floating-point numerical encoding is, basically, a power-of-2 exponent mantissa encoding. Two values are passed; the exponent and the mantissa. With such an expression the equivalent number is mantissa X 2^exponent. The mantissa is an expression that is actually a fraction, where the most significant bit is 2^-1, the next is 2^-2, etc. The exponent, on the other hand, is an integer. Hence the value 10.0 would be expressed as follows:

```
exp:00100 (4)

mantissa:   .101000000000
            ^
            |
            |__ binary decimal point.
```

This represents (2^-1 + 2^-3) x 2^4, which yields 2^3 + 2^1 = 8 + 2 = 10.

Other bits, values, in the expression will handle the sign of the exponent and the overall value.

One technique used to encode the sign is "excess". The actual value of the number (for example the exponent) is encoded value-128. In this fashion the programmer does not need to provide logic (testing and branching) to his conversion programs. The value can be achieved by simple subtraction.

The SD385 is provided with two floating point binary formats, selectable on Setup Page 8: BYTE and DEC. Byte format is optimized for the GPIB. A simple formulae on the four bytes output for a value will yield the desired number. DEC format is directly compatible with the internal floating point format used on PDP-11 and LSI-11 microcomputers, specifically the format used with the MACRO-11 language. This allows reading GPIB data directly to a floating point array without data conversion math.

Formulae for BYTE Format:

The four bytes for the value will be output in sequence. The formulae for the value expressed is given in terms of byte sequence: B1, B2, B3, B4.

For any value expressed by this format the value can be determined by using either of the following two formulas:

VALUE = {[(256*B1 + B4)-8192]/8192}*2^(B2-128)

or

VALUE = {[(B1/32)+(B4/8192)]-1}*2^(B2-128)

Device Address and Talk Only

Upon initial application of power, the device address for the SD385 is determined by DIP switch settings 1-5 on side-panel switch S3. All other switches on S3 should, at all times, be set to zero (0).

The device address can also be set on Setup Page 7. This selection will take priority over the rear panel setting until the unit is power-cycled or SYS RESET is pressed, at which time the switch setting will be used again. All of the other settings made on Setup Page 7 will be "remembered" and preserved by the unit through power-cycles and resets. The priority of the switch over the page selection is per the standard which requires a switch control for the device address. The address displayed on the setup page always reflects the one actually in use. Consequently, after power-up or reset this will be the switch setting rather than any previously entered value.

Desired Device Address	Switch Settings 54321 (sw #)
0	00000
1	00001
2	00010
3	00011
4	00100
5	00101
6	00110
7	00111
8	01000
9	01001
10	01010
11	01011
12	01100
13	01101
14	01110
15	01111
16	10000
17	10001
18	10010
19	10011
20	10100
21	10101
22	10110
23	10111
24	11000
25	11001
26	11010
27	11011
28	11100
29	11101
30	11110
31	11111 <------ if set will cause SD385 to power-up in talk only mode.

Table A-1. Device Address Switch Settings

Service Request (SRQ)

The SD385 can be set to generate Service Request on the following events:

Event	STB When Polled
AVERAGE COMPLETE	65
CHAN. A OVERLOAD	66
CHAN. B OVERLOAD	67
SPARE	68
SPARE	69
SPARE	70
WF MEM LOAD COMPLETE	71
ERROR	72
PLOT START PUSHED	73
PLOT COMPLETE	74
INPUT MEMORY SINGLE TRIGGER	
ACQUISITION COMPLETE	75
FRONT PANEL BUTTON PUSHED	76
PRINT START	77
PRINT COMPLETE	78

The SD385 will "queue up" events and output a stream of up to 17 STB responses to a Serial Poll. This allows user software to detect more than one event occurring between polls.

If an excess of events (>17) has occurred, without Serial Polls to clear some of them from the "queue" the STB will have a value of 127.

The SD385 will issue <u>no</u> Service Requests unless it has been told to via the Service Request Generation ("SRQGN ") data group.

Summary

The SD385 GPIB is an implementation of IEEE-488('78) : Standard Digital Interface for Programmable Instrumentation. It incorporates the basic talker/listener functions with Service Request, Serial Poll, Device Clear and talk-only capabilities.

The SD385 GPIB uses a mnemonic data group identification scheme with 32 basic data groups giving the controller/computer programmer the ability to remotely configure and operate the instrument and read control-status information, numerical readouts and ensemble/array signal source and processed data.

The data handled by the SD385 will be formatted as ASCII, binary or packed combinations of the two for very complex functions. Flexibility in binary floating point format, and terminators is included.

A-1.4 SD385 DATA GROUPS

CONFIGURATION COMMAND GROUPS

Write Mnemonic	Read Mnemonic	Description
"MCNFG "	"MCNFG?"	Machine Config. - packed byte/ASCII complete configuration of instrument. Read copy will reproduce configuration when sent in.
"PRCON "	"PRCON?"	Panel Recall Control - allows programmer to tell GPIB to ignore parts of MCNFG when sent in (panel recall).
"SCNFG "	"SCNFG?"	Single Config. - single setting configure read setting or write control.
"SRQGN "	"SRQGN?"	Service Request Gen. - allow user to designate events upon which to generate SRQ. Read allows check.

CONTROL OPERATION GROUPS

Write Mnemonic	Read Mnemonic	Description
"EASEL "		Erase Screen - clear screen in prep for....
	"ERROR?"	Error Reading - read error identification code.
"FPKEY "	"FPKEY?"	Front Panel Key - push button remotely, poll last button pushed.
"FPLOK "	"FPLOK?"	Front Panel Lock - lock out front panel, unlock, check condition
	"IDENT?"	Identify - read 380 ID (options, revision, etc.)
"SOUND "		Sound Beep - sound instrument "beep" (attention)
	"STTUS?"	General Status - read important instrument status.
"TEXT "	"TEXT? "	Text Entry - read or write the user text
"TIME "	"TIME? "	Time of Day - read or set
"VECTR "		Draw a Vector - draw vectors on screen
"PLANE "	"PLANE?"	Set or reads plane used for VECTR drawing
"RFDSP "		Forces screen annotation update
"STEST "	"STEST?"	Run self test, get results of self test

ENSEMBLE DATA GROUPS

Write Mnemonic	Read Mnemonic	Description
"AV*** " (Note 1)	"AV***?"	Average Memory Data - content of Average (AV) Memory by channel (Gaa or Gbb) or XPRD avg cross-channel (Gba, real or imaginary)
(Note 2)	"HIRM*?"	High Resolution Memory Data - content of trace 1, 2, 3 or 4 in high resolution (floating point)

Note 1 - * = channel identification characters: GAA, GBB; or or cross products identification characters: BAR or BAI

Note 2 - * = trace identification character: 1 or 2

ENSEMBLE DATA GROUPS (cont.)

Write Mnemonic	Read Mnemonic	Description
"INPM* " (Note 2)	"INPM*?"	Input Memory Data - Input Memory A or B data
(Note 3)	"LORM*?"	Low Resolution Memory Data - contents of trace 1 or 2 in low res. (byte integers).
"ST*** " (Note 1)	"ST***?"	Storage Mem. Data - content of Storage (ST) Memory by channel (Gaa or Gbb) or XPRD avg cross-channel (Gba real or imaginary).
"WFLC* " (Note 4)	"WFLC*?"	Waterfall Memory Calib. - Waterfall file 1, 2... or 8 calibration block
"WFLD* " (Note 4)	"WFLD*?"	Waterfall Mem. Data - Waterfall file 1, 2... or 8 data
"WFLRS "		Waterfall Mem. Reset - reset Waterfall read pointer
"WFLSZ "	"WFLSZ?"	Waterfall Transfer Size - set or read transfer size for Waterfall data

Note 1 - * = channel identification characters: GAA or GBB, or cross products identification characters: BAR or BAI

Note 2 - * = channel identification character: A or B; e.g., "INPMA?" or "INPMB?"

Note 3 - * = trace identification character: 1 or 2

Note 4 - * = file identification character: 1, 2, 3, 4, 5, 6, 7 or 8

NUMERIC OUTPUT GROUPS

Write Mnemonic	Read Mnemonic	Description
"CURSR "	"CURSR?"	Cursor Information - copy of cursor readouts, or positioning control.
	"LIST? "	List Readout - copy of harmonic, octave or peak list.
"MARKS "	"MARKS?"	Marks - copy of mark readouts, or mark position control.
	"PLOT? "	Plot Data Request - HP-GL instructions that copy screen data
	"STATS?"	Statistics - copy of special statistics readouts.
	"RPMDT?"	RPM Data Readout - request the current RPM.
	"RMSPC?"	Read channel rms signal levels as as percent of full scale.
	"PRINT?"	Print data request - HP THINKJET instructions for a copy of the screen.

A-2 PROGRAMMING COMMENTS

A-2.1 General Information

For any particular computer with IEEE capability there will be "syntax" implemented to achieve communication. Two basic techniques will be employed: language extensions and accessory software.

Typically language extensions include statements that will output commands and data on the GPIB and input data. Frequently the input statements will automatically issue the commands that cause the computer to listen and the device to talk.

Because of this, the language extensions include statement parameters that identify the device. For example, a syntax that uses print statements, with a GPIB card internally set as port 11 outputting to an SD385 at device address 20, would look like this:

```
PRINT @11,20:"TIME 01:01:01"
```

Variations that you can expect will be "write", "wrt", "wbyte", "wtb", "output", "prt", "send", etc. Variations on input syntax will be "read", "red", "input", "rbyte", "rdb", "enter", "ent", etc.

Some language extensions will allow or require you to declare port and device address symbolically in a "device" specifier or table. Thus, our example would change to:

```
DEVICE SD385:11,20
.
.
.
PRINT @SD385:"TIME  01:01:01"
```

Some extensions do not automatically issue IEEE commands. You must figure them out and send them yourself. For this reason, this appendix includes command definitions located in Table A-3.

To get the example data stream to the SD385, the instrument (at address 20) must be sent its listen command. Previously we noted that a listen command (MLA) has a bit pattern of X01PPPPP, where "PPPPP" is the device address. The "X01" part has a value of 32 so we know that the value of the byte that encodes this command is 32 + 20 = 52.

The following example syntax incorporates an "at sign" (@) to tell the computer to assert ATN and a colon (:) to tell it to drop it. Hence:

```
PRINT @52:"TIME  01:01:01"
```

To get the data out involves similar instructions. Note that when dealing with the SD385 the user must output a data identification mnemonic before reading in a stream.

In the following examples, "A$" is a string (character) variable used to receive the data.

```
PRINT @11,20:"TIME? "
READ @11,20:A$

DEVICE SD385:11,20
.
.
.
PRINT @SD385:"TIME? "
READ @SD385:A$

PRINT @52:"TIME? "
PRINT @84:          (64+20 yields talk cmd MTA)
READ A$
```

Thus, we see some of the variations in the input/output syntax and the necessary differences in approach resulting.

In either case, the accessory software implements IEEE communication via subroutine calls. Typically these are parameterized subroutines or functions. Instructions to the subroutines are in the parameters and in some cases a device table. The following is an example:

(output)

```
A$="TIME  01:01:01"
CALL TALK_IEEE(20,A$)
```

(input TIME to A$)

```
A$="TIME? "
CALL TALK_IEEE(20,A$)
CALL LISTEN_IEEE(20,A$)
```

A-2.2 Format Statements

When reading information from the SD385 the user will have to program for the data format and the termination technique chosen/needed. A good example of this is the status word output (STTUS?) from the SD385. This is a bit-mapped binary value for quick output. The data is two bytes, with the 6 character command mnemonic in front of it.

One technique for getting binary information into a computer will be the format statement. This varies from machine to machine, but usually is achieved with a pattern of characters that tell the computer what to expect. For example:

```
FORMAT (6A)
PRINT @SD385:"STTUS?"
FORMAT (6A,2B)
READ @SD385:A$,B,C
```

The result is the status word in values B & C. The "6A" tells the computer to expect 6 characters. 2B tells it to expect 2 bytes as numbers.

Some variations that might be seen will be special characters that tell the computer to output EOI on the last byte, output carriage return, line feed, etc.

The software accessory IEEE packages sometimes provide the formatting as part of the subroutine set:

```
A$="6A,2B"
CALL IEEE_FORMAT(A$)
```

or as part of a device table structure. When this is the case, it may be necessary to architecture the table so that the SD385 shows up as several devices with the same device address, but different format requirements.

NOTE

Although the SD385 outputs/inputs much of the information as ASCII, it is entirely possible to tell the computer to take ASCII information as bytes. Since the characters are bytes, interpretation of the data as characters can be left for later program sections. This must be evaluated by the individual programmer, of course, in concert with the total capabilities and idiosyncracies of the system implementation in question.

A-2.3 Summary of Needed Information

To construct an IEEE interface program, procedures and commands to implement certain functions must be known. The programmer must know the following:

1. primary address of the host computer/controller

2. how to instruct the computer to input and/or output to a specific device on the IEEE interface bus

3. how to instruct the computer to interpret a byte as a numerical value (this may constitute a formatted read or numeric function on a buffer location)

4. how to instruct the computer to terminate a data transmission, in either direction, by termination characters; e.g., EOI, byte count, etc.
5. how to instruct the computer to issue a Device Clear
6. how to instruct the computer to sense a Service Request Generation

7. how to instruct the computer to conduct a Serial Poll

NOTE

Items 5 through 7 are application dependent.

The remainder of this appendix constitutes detailed information on the data groups; i.e., content, meaning, and programming examples. The programming examples will all be HP-85 extended BASIC.

A-3 CONFIGURATION COMMANDS

A-3.1 "MCNFG " Machine Configuration

The machine configuration command is used to set up the parameters and modes of the analyzer. The data is comprised of numbers corresponding to the actual values for the parameters and to the menu selections for the modes. Each value is explained in Table A-4.

This is a 368 byte packed ASCII & byte selector "map" of the entirety of the instrument configuration. When sent to the SD385, the configuration defined by the map settings will be assumed by the SD385. The map consists of byte values which represent selections within menus (1 is selection 1; 2 is selection 2; etc.) and ASCII fields (1 to 10 characters) that represent some number. Characters encountered in these fields will be purely numerical (the characters 0, 1,...9), hyphen (-) and decimal point (.). All of the Setup Page selections are contained in this stream, plus the cursor, reference and mark locations.

All of the Setup Page selections are contained in this stream, plus the cursor, reference and mark locations.

Each control location in the map has a number assigned which, for documentation purposes, is called the Map Location Index (MLI). The MLI number for the various controls ranges from 71 to 249.

For a summary of the entire map, refer to Table A-4.

NOTES:

1. There are three display memory selections:

   Type 1 is for single channel Spectrum, PDH and CD. The menu selections are:

   RT & AVG:
   RT & STO:
   AVG & STO:

   Type 2 is for 2 Ch Spectrum, 2 Ch Math; PDH and CD; and all Time functions. The menu selections are:

   RT: INP & RT
   AVG: INP & AVG
   STO: INP & STO

   Type 3 is for all Transfer Function, Power and IFFT functions. The menu selections are:

   AVG:
   STO:

User Notes

Making individual settings can be accomplished by reading the MCNFG map out, making some changes and then sending it back in. However, use of the Single Configuration command is recommended where only a few changes are to be made.

NOTE

> Attempting to compose a complete map in program is not recommended. The opportunity for error is large.

Except for the cursor location controls, each of the control locations in the previous map have an equivalent setup page control. The settings for the byte-selectors will be the same as menu numbers seen in the setup pages.

For example, to change the Y UNITS to VOLTS, the following program could be used:

```
OUTPUT 720 ; "MCNFG?"           ! tell 385 send map
TRANSFER 720 TO A$ FHS ; EOI    ! read to A$, end on EOI
A$[18,18]=CHR$(1)
OUTPUT 720 ; A$
```

ASCII can likewise be read, changed and sent. The following program cycles MV/EU for chan. A through settings of 100, 200,....1500.

```
DIM A$[388]                        !8 byte overhead
IOBUFFER A$                        !declare as buffer
OUTPUT 720 ;"MCNFG?"               !tell 385 send map
TRANSFER 720 TO A$ FHS ; EOI       !read to A$
FOR I=100 TO 1500 STEP 100         !loop
A$[87,91]=VAL$(I)                  !set as mv/eu A
OUTPUT 720 ; A$                    !send it over
WAIT 2000                          !see result
NEXT I                             !keep going
END                                !finis
```

Note that the map needed to be read only once.

The above technique may seem a little cumbersome for making individual settings of the instrument. That is not the intended primary application for the map. It is a complete definition of the configuration of the instrument. It's best use is storage/retrieval applications. The programmer has the instrument/system operator set up the configuration for a test and then stores the resultant map for later sending to the SD385 for complete restoration of an analyzer configuration.

NOTE

> Once sent in, a period of time will pass before the instrument will be completely configured to match the map. The "config in process" bit in the status word will help determine when it is proper to operate the instrument on the new settings.

A-3.2 "PRCON " Panel Recall Control (Map Masking)

The "PRCON " data group allows the user to mask out parts of the map for configuring the instrument. This is the same as the Panel Recall feature used for storing and recalling complete analyzer configurations. The effect of this is to cause parts of the input map to be ignored as far as the configuring process is concerned.

This data group is directly related to the PRS feature of the Setup Listing, and is in fact a copy of it that can be changed directly by the GPIB user. The masking of sections of the map is done on a setup page basis. Only the IEEE page is fixed on power-up to prevent inadvertent change.

This command is of the "switch" data type. A one (1) will mask the appropriate setup page, a zero (0) will allow it to be reconfigured and an asterisk (*) indicates that the current setting should be left alone.

An example of the intended application for this control would be a setup where the entered Y CALIB parameters (page 5) match the transducers currently in use. The programmer has read to A$ a previously recorded map from disk storage and wishes to configure the instrument to match without affecting the current Y CALIB parameters. This could be accomplished by:

```
OUTPUT 720 ;"PRCON 000010010"
OUTPUT 720 ; A$
```

The Y CALIB part of A$ would be ignored, and the current Y-CALIB settings of the instrument would be preserved.

A-3.3 "SCNFG " Single Configurataion

Another technique for configuring the instrument is the single configuration control group (SCNFG). This data group relates directly to the "MCNFG " stream. The MLI numbers in the map table are used to identify the control to be exercised. The input data is two ASCII fields delimitted by a comma (,). The first number is interpreted by the SD385 as the MLI to be set, the second as the desired setting. For example, the following program will cycle through all five Y-units:

```
FOR I=1 TO 5                    !loop
A$="SCNFG 82,"&VAL$(I)          !y units MLI & setting
OUTPUT 720 ; A$
WAIT 3000                       !see result
NEXT I
END
```

This is one group where input and output terminators may make a difference, since the SD385 will not process the input stream until it knows it received the last byte. For this example to work, we had to set the SD385 input terminator to CR so that the HP-85's default CR/LF terminator would work.

It is also possible to make more than one setting. For example to set Y UNITS to EU and Y OPERATION to MAG^2 to get EU^2 when Y-units is MLI 82 and Y-oper is MLI 125, send:

```
OUTPUT 720 ; "SCNFG 82,2,125,2"
```

NOTE

Making changes via the Single Configuration command may not update the annotation on the display screen. For example, the command "SCNFG 174,1" will change the Frequency Range to 20kHz, but the display will still show the previous value. If a screen update is desired, issue an "RFDSP" (Refresh Display) command.

The SCNFG data group can also be used to read individual control settings. For example, the following program reads the setting for every MLI and the print shown was the result.

```
10 DIM A$[100]
20 IOBUFFER A$
30 FOR I=71 TO 249
40 OUTPUT 720 ;"SCNFG?"&VAL$(I)
50 TRANSFER 720 TO A$ FHS ; EOI
60 PRINT A$
70 CONTROL A$,1 ; 0 ! clear A$
80 NEXT I
90 END
```

Format of output:

```
SCNFG  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
       |         | | |   | | |               |
        ---------  |  ---  |  ---------------
            |      |   |   |         |
            |      |   |   |          ----------- 12 Thru 20 = Value
            |      |   |    --------------------- 11 = Comma
            |      |    ------------------------- 8 Thru 10 = MLI #
            |       ----------------------------- 7 = Space
             ------------------------------------ 1 Thru 6 = Command
                                                            Echo
```

A-3.4 "SRQGN " Sevice Request Generation

This data group is used to tell the SD385 what events to generate SRQ upon. It is "switch" data just like the PRCON data group. A one (1) indicates that SRQ is desired on the event related to the position, a zero (0) that it is not. An asterisk (*) indicates that the current setting should be left alone. This allows a certain amount of progressive software to be written on the SRQ feature, though the user should make a habit of using an asterisk instead of zero right from the very start if he wishes to use this feature.

These program statements:

```
OUTPUT 720 ; "SRQGN?"
ENTER 720 ; A$
```

yield the following:

```
SRQGN 000000000000000
      |||||||||||||||
      |||||||||||||| -15 Spare
      ||||||||||||| --14 Print finished
      |||||||||||| ---13 Print requested
      ||||||||||| ----12 Front panel button pushed
      |||||||||| -----11 Input Memory single trigger
      ||||||||||         acquisition complete
      ||||||||| ------10 Plot finished or stopped
      |||||||| --------9 Plot requested (button pushed)
      ||||||| ---------8 Error occurred
      |||||| ----------7 Waterfall load complete
      ||||| -----------6 Spare
      |||| ------------5 Spare
      ||| -------------4 Spare
      || --------------3 Channel B overload
      | ---------------2 Channel A overload
       ----------------1 Average stop
```

NOTE

The SRQ resulting from a front panel button push will not occur when buttons are pushed remotely from the GPIB.

A-4 CONTROL OPERATIONS

A-4.1 "EASEL " Erase Screen

The command "EASEL X" erases all or part of the display screen and controls the analyzer's active screen updating; Where "X" is a command specifier number (0, 1, 2 or 3). This command can be used to prepare the screen for user graphics and text.

3 -- erases trace, grids & annotation
2 -- erases trace & grids only
1 -- erases annotation only

All three of the above specifiers turn off the appropriate area of active screen updating by the analyzer. Updating of the selected areas is accomplished with the GPIB via the TEXT and VECTR commands.

0 -- restores active screen updating by the analyzer

A-4.2 "ERROR?" Error Reading

Table A-2 defines the content meaning of the value returned by the "ERROR?" data group. The output will be "ERROR XXX". Where "XXX" is a three character ASCII field containing the error number.

TABLE A-2. SD385 Error Codes

ERROR #	SCREEN TEXT
1	**INTERNAL ERROR #1** - Attempted control request; MLI is out of range.
2	**FILE EMPTY** - Attempt to display empty WF file.
3	**FILE FULL** - Rejects new data in WF FULL mode.
4	**ILLEGAL OPERATION** - (WF) Changing settings while updating, etc.
5	**EXPRESSION OUT OF RANGE** - Attempt to enter ASCII field with value out of range.

TABLE A-2. SD385 Error Codes (cont.)

ERROR #	SCREEN TEXT
6	**INVALID ANALYZER SETUP** - Data not suitable for WF load.
7	**CHANGE TO ANALYZER SETUP CLEAR FILE AND RE-UPDATE** New data for WF load does not match current WF file contents.
8	**TO CLEAR WATERFALL FILE, PRESS CLEAR AGAIN** Requires two presses of the CLEAR button to actually **(FPKEY 18,18)**.
9	**NO HIDDEN LINES IN WATERFALL SCROLL** - Attempt to enable Hidden Lines while WF scroll is on.
10	**X2,X4 WILL BE FORCED X1 IN ZOOM OR AMP DOM** - (analyzer) Attempt to set X2, X4 when not allowed.
11	**CLEAR ALL WATERFALL FILES BEFORE CHANGING FILE SIZE** Changing file size erases the WF Memory. Unit is asking to make sure.
12	**LOGX WILL BE FORCED LINX1 IF NOT FREQ DOM BASEBAND** Attempt to set LOG X when not allowed.
13	**DB UNITS ARE FORCED TO VOLTS IF BIPOLAR (TIM, RE IM) DATA** - Attempt to set DB when bipolar data is up.
14	**NO HIDDEN LINES IN X2 OR X4** - Attempt to enable Hidden Lines when in X2 or X4.
15	**NO TACH SIGNAL** - Attempt to D RPM acquire when no tach signal present.
16	**MATH/ORB WILL BE FORCED DUAL WHEN X CALIB IS UNEQUAL** Attempt to select math or orbit when X calibration of trace 1 is different from that of trace 2.
17	**CMPRSD X AXIS VALID ONLY IN TIME INPUT ELSE FORCED LINX1** - Compressed X-Axis selected when selected function is other than time domain.

TABLE A-2. SD385 Error Codes (cont.)

ERROR #	SCREEN TEXT
18	**YAXIS WILL BE FORCED LOG FOR DB UNITS** - Attempt to set LIN Y-axis when DB units are displayed.
19	**Spare**
20	**Spare**
21	**Spare**
22	**INPUT MEMORY DATA NOT ACQUIRED** - Attempt to read unloaded Input Memory.
23	**SPARE**
24	**SRA OPTION - RPM ERROR**
25	**CONTROL MUST BE ENTERED FROM KEYPAD** - Attempt to use SCROLL UP/DOWN directional buttons on an ASCII Control Field.
26	**ZOOM WILL BE SUPRESSED FOR THIS FUNCTION** - Attempt to set Zoom when not allowed.
27	**SPARE**
28	**I/O SOFTWARE FAILURE**
29	**SPARE**
30	**NO LISTENERS ON GPIB SYSTEM** - SD385 addressed to talk, found no active listeners on the bus (RFD and DAC messages asserted simultaneously).
31	**SPARE**
32	**UNRECOGNIZABLE GPIB ANALYZER COMMAND** - Mnemonic input to SD385 was not understood.

TABLE A-2. SD385 Error Codes (cont.)

ERROR #	SCREEN TEXT
33	**ILLEGAL VALUE IN I/O DATA** - Value in part of data-stream input to SD385 was either out of range, had invalid characters or caused internal errors when I/O attempted to process.
34	**SPARE**
35	**NOT ENOUGH I/O DATA RECEIVED** - I/O did not receive enough data to process (incomplete map or array data stream).
36	**INVALID CALIBRATION DATA BLOCK** - Calibration information input with data contained invalid values.
37	**BAD INPUT MEM/WF I/O BLOCK SIZE** - Block size input to I/O for transmission of WF Memory data was unusable.
38	**I/O ERROR: DATA DOESN'T EXIST** - Requested data stream does not exist.
39	**SPARE**
40	**AT LIMIT** - (WF) Attempt to scroll past end of WF file.
41	**OUT OF RANGE** - (WF) Entered value was out of permissable range.
42	**SPARE**
43	**FORMAT DISK** - Disk installed in drive requires formatting to be compatible with the SD385.
44	**SPARE**
45	**INCOMPATIBLE FILE TYPE** - The data type trying to be stored/recalled does not match the file type; i.e., loading Spectrum data into a Waterfall data file.

TABLE A-2. SD385 Error Codes (cont.)

ERROR #	SCREEN TEXT
46	**STORE/RECALL PROBLEMS** - Disk Reader detected a hardware failure. Unit may require repair.
47	**INCREASE WF FILE SIZE**
48	**FILE DOES NOT EXIST** - File number does not appear on the directory.
49	**DIRECTORY FULL** - The maximum number of files (255) have been opened.
50	**PLACE MEMORIES IN HOLD** - Memories must be in HOLD before data storage can occur.
51	**IEEE PRIORITY** - The SD385's GPIB is currently busy performing another transfer. When the GPIB is free, the requested operation causing this error will be performed.
52	**CAN'T STORE 30 1/3 OR 10 OCTAVE OF INP OR RT DATA**
53	**INVALID TRACE DATA** - Attempt to store an empty memory or Waterfall file.
54	**SELF TEST FAILURE** - Drive unit failed the self-test diagnostic. Drive unit may require service.
55	**DATA STORE MSG ERR**
56	**DRIVE ERROR** - Disk Reader detected drive unit failure. Drive unit may require service.
57	**ILLEGAL OPERATION** - Action performed is not allowed.
58	**WRITE PROTECT** - Writing attempt made to a write protected disk.

TABLE A-2. SD385 Error Codes (cont.)

ERROR #	SCREEN TEXT
59	**CONTROLLER ERROR** - Disk Reader detected drive unit failure. Drive unit may require service.
60	**BAD DISK** - Disk installed in drive unit is defective.
61	**CLEAR WF FILE** - Attempt to recall a Waterfall file when the Waterfall is already loaded with Waterfall data. This is done to prevent accidental overwrite of data.
62	**SYSTEM ERROR** - Disk Reader detected drive unit failure. Drive unit may require service.
63	**FILE EMPTY** - (WF) attempt to recall a record from an empty file.
64	**SPARE**
65	**INSERT DISK** - There is no disk inserted in the drive unit.
66	**DISK FULL** - All available data bytes have been used.
67	**RPM TOO HIGH FOR SRA TRACK**
68	**RPM TOO LOW FOR SRA TRACK**
69	**MIN RPM X PPR/FSORD LESS THAN 5HZ**
70	**MAX RPM X PPR/FSORD MORE THAN 10KHZ**

NOTE

Most of the errors relate to control settings made by the operator on the front panel. If the GPIB is used, via "SCNFG ", to make MLI settings, and the I/O sees an error return from the analyzer control, the I/O generates error 33. Other errors in response to operations, will be seen by the GPIB as is.

A-4.3 "FPKEY " Front Panel Key

This data group is used to perform operations on the analyzer. Predominately the remote key-push capability is used to operate the instrument (e.g. start an average, initiate an Input or Extended Memory update, Waterfall load, etc.)

This is an ASCII group whereby the programmer sends "FPKEY " and some digits. For example the following will push the AVERAGE START button:

```
OUTPUT 720 ; "FPKEY 9"
```

More than one button can be pushed in one stream transmission by including more button codes, delimited by commas. If a zero precedes the stream of button codes, the SD385 GPIB will "push" them silently (no beep). For example:

```
OUTPUT 720 ; "FPKEY 0,3,9"
```

will push UPDATE and AVG START without sounding the beep. "FPKEY?" will cause the SD385 to output the number of the button last pushed. This includes remotely pushed buttons. The number-button equivalence is the same as for input, and is defined in Figure A-2.

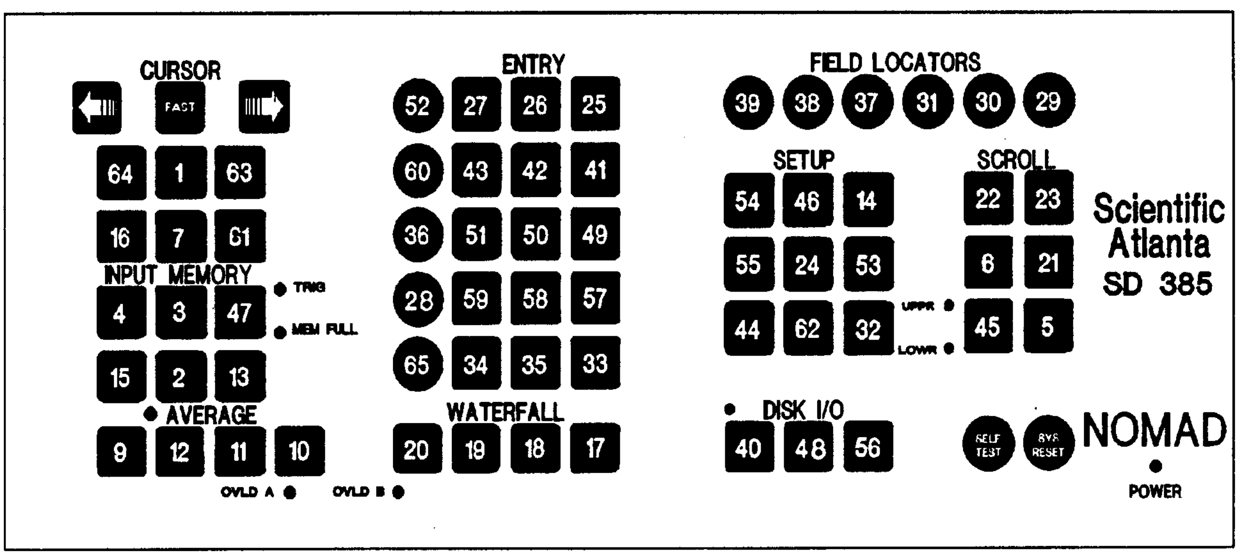


Figure A-2. Front Panel Button Key Codes

The following is a list of the buttons and their appropriate keycodes:

CODE	BUTTON	GROUP	CODE	BUTTON	GROUP
1	CURSOR UP	CURSOR	34	.	ENTRY
2	HOLD	INPUT MEM	35	0	ENTRY
3	UPDATE	INPUT MEM	36	%TH	ENTRY
4	AUTO RANGE	INPUT MEM	37	FREQ	FIELD LOC
5	WATERFALL	SCROLL	38	MEM	FIELD LOC
6	SCROLL DOWN	SCROLL	39	FUNC	FIELD LOC
7	CURSOR DOWN	CURSOR	40	DISK	DISK I/O
8			41	9	ENTRY
9	START	AVERAGE	42	8	ENTRY
10	STORE	AVERAGE	43	7	ENTRY
11	STOP	AVERAGE	44	PANEL	SETUP
12	CONT	AVERAGE	45	SEL TRACE	SCROLL
13	INP MEM RT	INPUT MEMORY	46	SETUP UP	SETUP
14	PAGE ADV	SETUP	47	SEL CHAN	INPUT MEMORY
15	INP MEM LEFT	INPUT MEMORY	48	PRINT	DISK I/O
16	LIST	CURSOR	49	6	ENTRY
17	DISP ON/OFF	WATERFALL	50	5	ENTRY
18	CLEAR	WATERFALL	51	4	ENTRY
19	HOLD	WATERFALL	52	CF	ENTRY
20	LOAD	WATERFALL	53	SETUP RIGHT	SETUP
21	ZOOM	SCROLL	54	SETUP ON/OFF	SETUP
22	SCROLL UP	SCROLL	55	SETUP LEFT	SETUP
23	MENU	SCROLL	56	PLOT	DISK I/O
24	HELP	SETUP	57	3	ENTRY
25	ENT	ENTRY	58	2	ENTRY
26	CLR	ENTRY	59	1	ENTRY
27	RCL	ENTRY	60	AVG#	ENTRY
28	FILE	ENTRY	61	SET	CURSOR
29	AVG	FIELD LOC	62	SETUP DOWN	SETUP
30	BAND	FIELD LOC	63	MARK	CURSOR
31	Y UNITS	FIELD LOC	64	RESET	CURSOR
32	TEXT	SETUP	65	TIME	ENTRY
33	+/-	ENTRY			

A-4.4 "FPLOK " Front panel locking, unlocking.

Send "FPLOK 1" to lockout the front panel from unfriendly fingers, "FPLOK 0" to unlock it.

A-4.5 "IDENT?" Identify

"IDENT?" will cause the SD385 to output a stream of codes that identify the number of channels, firmware revision, and options installed. The resultant output will look like this:

```
IDENT 385,2,X1.1,  0,  0,255,255,  0
           |  |     |   |  |   |    |
           |  |     |   |  |   |     -- Spare
           |  |     |   |  |    ------- Data Storage Option
           |  |     |   |   ----------- SRA Option
           |  |     |    -------------- Spare
           |  |      ------------------ Spare
           |   ------------------------ Revision
            --------------------------- 1 = 1 chan, 2 = 2 chan

             Option    0 = not installed
             Codes     255 = installed
```

A-4.6 "SOUND " Sound Beep

The "SOUND " mnemonic is a user convenience that sounds the analyzer's beep. This can be used to signal events in a test or to get attention from people operating the system.

A-4.7 "STTUS?" General Status

"STTUS?" will cause the SD385 to output a binary status byte that quickly identifies current instrument activities of interest. The output will be the six character command echo and two bytes of status ("STTUS b,c" where b & c are bytes that can take on any value, depending on the instrument activity).

For example:

```
OUTPUT 720 ; "STTUS?"
ENTER 720 USING "6A,#,B,B" ; A$,B,C
```

will bring the bytes to B & C for bit detection (note the format "image" specifier -- 6A for the 6 character ASCII command echo, the "#" to cause termination when the last variable in the statement has been satisfied and B for binary).

The following is a definition of the bit values:

FIRST BYTE (BYTE 7)

BIT	MEANING
7 (msb)	Configuration change in progress
6	Front panel in text mode
5	Overload A
4	Overload B
3	Waterfall Scroll
2	Zoom
1	Update
0 (lsb)	Average

SECOND BYTE (BYTE 8)

BIT	MEANING
7 (msb)	Input Trigger
6	Input Fill
5	Panel lock
4	Auto Range
3	Waterfall Load
2	Trace Upper
1	Trace Lower
0 (lsb)	Spare

The following sample program starts the Average, then reads the status word to test for completion of the Average.

```
10 OUTPUT 720;"FPKEY 9"          !push start
20 WAIT 1000                     !let it happen
30 OUTPUT 720;"STTUS?"
40 ENTER 720 USING "6A,#,B,B" ; A$,B,C
50 IF NOT BIT (C,0) THEN 100
60 WAIT 500
70 GOTO 40
100 BEEP @ DISP "AVERAGE DONE" @ END
```

Some of the conditions in the status word will not be set immediately in response to the push of a button. For example, in the above sample program, a one second wait was incorporated after the AVERAGE START push, before the program started sampling the status word. The instrument turns on the Average In Progress flags when, and if, it actually starts the average.

A variation on the program could account for that, with more reliability than the one second pause:

```
10 OUTPUT 720 ; "FPKEY 9"
20 OUTPUT 720 ; "STTUS?"
30 GOSUB 200
40 IF BIT (C,0) THEN 70
50 WAIT 500
60 GOTO 20
70 WAIT 500
80 GOSUB 200
90 IF BIT (C,0) THEN 70
100 BEEP @ DISP "AVG DONE" @ END
200 ENTER 720 USING "6A,#,B,B";A$,B,C
210 RETURN
```

It is best to remember that the status word yields the current status. Frequently the user really wishes to detect an event which is not indicated merely by status, but a change in status (e.g. the above sample program which "triggers" only when status changes from Average On to Average Off), SRQ's may be used to detect such events.

A-4.8 "TEXT " Text Entry

The "TEXT " command provides the ability to remotely put user text on the screen, or read such text to the controlling computer ("TEXT? "). This command accesses the same text capability as the front panel text feature, without the need for pushing buttons. Each available line on the screen is 56 characters long and has an assigned line number. Line 1 is at the top, line 20 at the bottom. This command has the following format:

"TEXT xx,text..."

where 'xx' = line number

This sample program:

```
OUTPUT 720 ; "TEXT  2,SAMPLE TEXT HERE"
OUTPUT 720 ; "TEXT  20,AND HERE"
```

results in the following display:

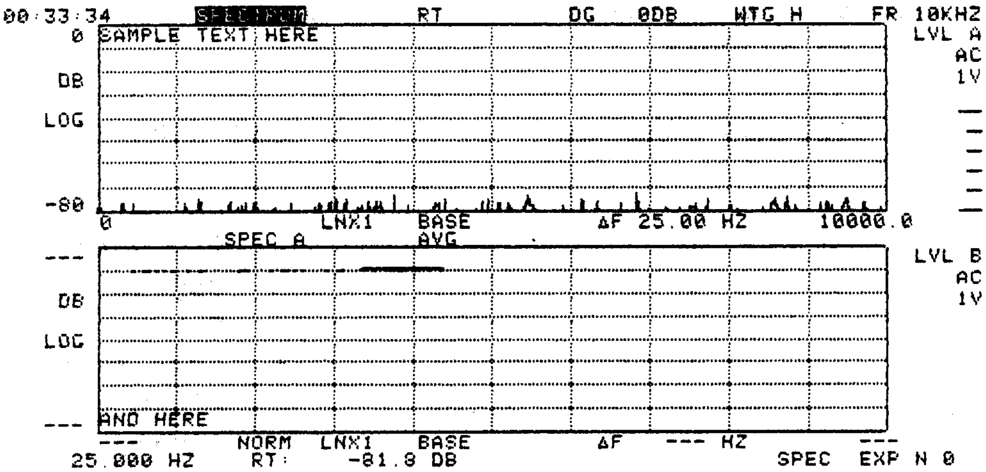


Figure A-3. TEXT ENTRY Display

If the user outputs "TEXT 2" without any input text in the stream, the GPIB will clear the line from the screen:

```
OUTPUT 720 ; "TEXT  2"
OUTPUT 720 ; "TEXT  20"
```

will clear the previous entries from the screen.

"TEXT? " will cause the SD385 to output the contents of the last line specified in a Text read command.

To read any line of text send the TEXT? command followed by the line number of the text desired to be read. For example:

```
"TEXT? 1"
```

will read the contents of line 1.

Entry of special text characters not located on the front panel may be accomplished using the GPIB. These characters are related to the user's keyboard in the following manner:

Keyboard Character	Text Character
?	Delta
$	ϕ Phase
!	Degree
;	Sigma
[	Left absolute bar
]	Right absolute bar
\	Square Root
*	Exponential 2

A-4.9 "TIME" Time of Day

We used this group in some of the earlier examples. You enter the desired time of day as three 2-digit numbers separated by colons (:) or commas (,), for example:

```
OUTPUT 720 ;  "TIME  08:30:56"
      or
OUTPUT 720 ;  "TIME  08,30,56"
```

will set the time of day to 8:30:56.

Five seconds later the following:

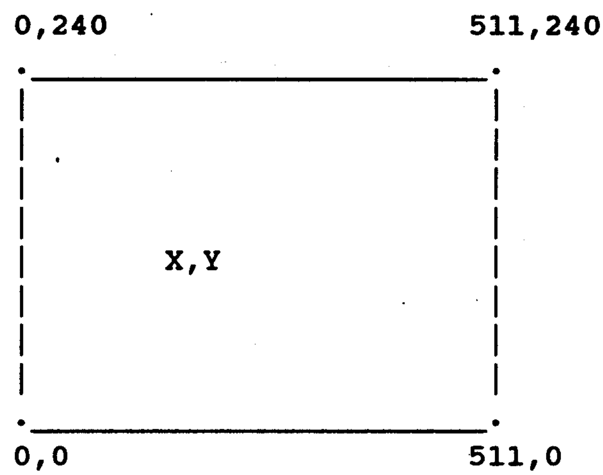
```
OUTPUT 720 ;  "TIME? "
ENTER 720 ;  A$
```

will retrieve A$ of "TIME 08:31:01".

A-4.10 "VECTR " Draw a Vector (Graphics)

The "VECTR "command allows the user to draw user-graphics on the screen.

The command is accompanied by numbers which define the X and Y points of the start of the vector and the X and Y points of the end of the vector. The screen is calibrated as shown:



The following commands:

```
"VECTR 0,0,0,240"
"VECTR 0,240,511,240"
"VECTR 511,240,511,0"
"VECTR 511,0,0,0"
```

would draw a box around the outside limits of the screen.

NOTE

> Unless the active screen update is turned off, the trace to screen update will erase some of the user graphics. Vectors can be erased by drawing to the same location again. Active screen updating is controlled with the EASEL command.

A-4.11 "PLANE" Set or Read Drawing Plane

Plane 1: Vectors will be drawn in the trace plane.

Plane 2: Vectors will be drawn in the grid plane.

The Trace Plane is being redrawn when analyzer is updating or configuration is changing.

Therefore, vectors drawn in this plane will be lost under these conditions. Vectors drawn in the grid plane will remain until a grid change occurs.