53B-ATZ PC/386 INSTRUMENT SYSTEM

OPERATING MANUAL

Printed in U.S.A.

03/17/92 9104-01-A
through
9107-01-D

WARRANTY

Colorado Data Systems, Inc. (CDS) products (hardware and firmware) are warranted against defects in materials and workmanship, and are warranted to meet the performance specifications as listed in the current catalog and/or data sheet for the specific product being warranted. This warranty applies for three (3) years following the date of shipment. CDS will, at its option, repair or replace, at no cost to the customer, products which prove to be defective during the warranty period, provided the defect or failure is not due to misuse or abuse of the product. The customer is responsible for shipment of the defective product to the CDS factory. Software products are supplied on a site license basis subject to the same performance warranty provisions; the materials and distribution provision applies to the distribution media only. NO OTHER WARRANTY IS EXPRESSED OR IMPLIED, INCLUDING WARRANTY FOR FITNESS OF PURPOSE. CDS SHALL, IN NO CASE, BE LIABLE FOR CONSEQUENTIAL DAMAGES.

TRADEMARK NOTIFICATION

53B-ATZ PC/386 INSTRUMENT SYSTEM

OPERATING MANUAL

DESCRIPTION 1

CONTROLS AND INDICATORS
- IEEE-488 INTERFACE PORT CONFIGURATION SWITCH 4
- JUMPERS 4
- FUSES 5
- DISK LEDs 5

SPECIFICATIONS 6

OPERATION
- OVERVIEW 9
- BASIC PROGRAMMING 10
  - Addressing Cards 11
  - Handling Interrupts 11
  - Compiled BASIC 11
  - Driver Descriptions 12
- C PROGRAMMING 20
  - Addressing Cards 20
  - Handling Interrupts 21
  - Driver Descriptions 22
- PASCAL PROGRAMMING 30
  - Addressing Cards 30
  - Handling Interrupts 30
  - PASCAL Procedures 31

INSTALLATION
- INSTALLATION REQUIREMENTS AND CAUTIONS 38
- INSTALLATION PROCEDURE 38
  - Mechanical Installation 38
  - Installing Option 31 - LCD Display 39
  - Installing GPIB Cables on the Butch Plate 40
  - Final Installation and Configuration 41
  - Adding an IBM AT-Compatible Expansion Card 41
  - Installing the Drivers 42
  - Setup Program/Changing Configuration 43

APPENDIX A
- 53/63 SERIES SYSTEM COMMANDS 45

APPENDIX B
- LCD FLAT PANEL DISPLAY 46

APPENDIX C
- CMOS MEMORY SETUP 53

53B-ATZ PC/386 INSTRUMENT SYSTEM

DESCRIPTION

The 53B-ATZ combines a complete ten slot CDSbus Instrument System and an embedded Controller based on an IBM compatible PC-AT/20 MHz 386 computer, including one full sized expansion slot, into a single chassis. This unique combination forms one of the fastest and most powerful card modular systems in the marketplace today, in many cases outperforming systems costing two and three times as much. The 53B-ATZ can operate either as a stand alone, self-contained CDSbus system, eliminating the need for an external system controller or computer, or as an intelligent node in a larger computer automated test (CAT) system.

The computer incorporated into the 53B-ATZ is optimized for test instrumentation and data acquisition applications. It includes:

- a 20 MHz 80386-based CPU,
- an IEEE-488 interface port for control of external instruments,
- optimized CDSbus interface drivers that support test program development in C, QuickBASIC, interpreted BASIC, and PASCAL,
- one parallel printer port and two serial interface ports (only one serial port with Option 31 LCD Display),
- a 40 Mbyte hard disk (a 119 MByte hard disk optional),
- 4 Mbytes of 16-bit wide zero wait state memory,
- a 1.44 Mbyte 3.5 inch floppy drive,
- a VGA graphics adapter (Option 31 is a flat panel LCD display),
- an optional math co-processor,
- an AT-style keyboard, and
- a full length PC-AT expansion slot.

All of this is packaged in a 53B-003 CDSbus instrument Card Cage together with:

- a 53A-171 Control Card,
- a CDSbus communications and chaining card (the chaining function allows the user to add up to nine additional CDSbus cages),
- all required power supplies and cables,
- rack ears, slides, and mounting hardware, and
- a rear mounted butch plate for convenient access to external peripherals and the UUT/DUT.

After initial installation, the embedded System Controller may be operated either with or without a keyboard or display, making it ideal for use in a factory floor environment. When used with the optional LCD Display Unit (Option 31), it gives the user the full functionality of a complete computer system, including keyboard and monitor, and a complete test system with up to ten full function instruments, packaged as a stand-alone system in a unit just ten and a quarter inches high.

The 53B-ATZ performs all of the computer related functions critical to the proper operation of a test station:

1) it can be the test station's system controller, providing a full development environment as well as execution capability for application programs, or an intelligent node in a larger test strategy;
2) it is an IEEE-488 compatible system controller; and
3) it is the CDSbus system controller.

It uses a 20 MHz 80386 based computer subsystem CPU coupled with an MS-DOS compatible operating system to provide a complete development and application program capability contained entirely within a CDSbus cage. The computer subsystem includes the CPU, a socket for an optional floating point processor, hard drive, floppy drive, memory capacity, interface ports, expansion slot, and the IEEE-488 interface normally associated with a full function instrument controller.

MS-DOS compatibility allows test development to be done directly on the 53B-ATZ or on any PC-compatible computer. It also allows use of the full set of software tools available in the MS-DOS workplace, including editors, compilers, assemblers, language processors, windows packages, etc. Several of these packages are available as standard options to the 53B-ATZ, including Windows™, Tektronix' TekTMS Test Management System, and Microsoft's C programming language. A fully licensed copy of the MS-DOS package is included as part of the 53B-ATZ. Tek/CDS developed Interpreted BASIC, QuickBASIC, C, and PASCAL CDSbus interface drivers, optimized for CDSbus performance, are also included.

The IEEE-488 interface port consists of hardware and software that fully implement the IEEE-488 standard, also known as GPIB. The interface is based on Capital Equipment Corporation's PC<>488 Card and is fully compatible with all CEC software. It is also fully compatible with National Instruments' software. (The IEEE-488 interface is an international standard that allows the system controller to communicate with over 2000 instruments made by over 200 manufacturers.) The IEEE-488 port can be used with applications programs in any of the popular programming languages used for instrument control applications. It can also be used without any programming to connect GPIB peripheral devices such as printers and plotters to the PC.

The IEEE-488 interface port's key features include:

- implementation of the entire IEEE-488 standard. The port can operate as a system controller or a device.

- on-board firmware (software in read-only memory), which provides GPIB extensions for popular programming languages. Firmware does not need any special installation steps, and never needs to be loaded from disk.

- a choice of programming methods: simply use the PRINT and INPUT statements of your programming language, or use CALLs directly to the firmware routines for maximum speed and flexibility.

- support of GPIB printers and plotters for use with MS-DOS, word processing, spreadsheet, and graphics programs.

The CDSbus interface port consists of hardware and software that allows the user's application program to communicate with CDSbus instruments in the 53B-003 Card Cage. In addition to controlling instruments located in the system controller's card cage, up to ten card cages may be chained together and controlled by the system controller in the main card cage.

CONTROLS AND INDICATORS

The following controls and indicators are provided to select and display the functions of the 53B-ATZ's operating environment. See Figure 2 for their physical locations.

IEEE-488 INTERFACE PORT CONFIGURATION SWITCH

The 152SA1 has a Configuration switch which is accessed through the rear of the 53B-003 Card Cage once the computer sub-assembly is removed. Refer to the Installation section of this manual for instructions on how to access the 152SA1 PCB.

This red six-position paddle switch is located near the center front of the 152SA1 board. Its settings control the base address of the driver PROM and whether the unit will be a system controller or a simple controller. The settings of the six paddles are as follows:

1 - System Controller:

This switch is normally open, making the IEEE-488 interface port a system controller. If this switch is closed, the IEEE-488 interface port is a simple controller. (IEEE-488 provides protocol to allow multiple controllers in a system to take turns as controller-in-charge. However, one controller, defined as the System Controller, retains control of the IFC and REN lines.)

2 through 6 - PROM Base Address:

These switches are set at the factory for a base address of 0C8000h. This setting should not be changed without consulting Tek/CDS at 1-800-CDS-ATE-1.

Paddle	Setting
2	open
3	closed
4	closed
5	open
6	open

JUMPERS

There are four jumper fields on the 152SA1 for selecting DMA request, DMA acknowledge and interrupt settings for both the CDSbus and IEEE-488 interfaces. These jumper fields are located on the lower front side of the 152SA1 (see Figure 2).

The CDSbus interface is set at the factory to use DMA channel 1 (DRQ1 and DACK1) and interrupt 10. The IEEE-488 interface port is not set at the factory for DMA or interrupts since it is not known what kind of expansion board may be plugged into the 53B-ATZ expansion slot. If there is no expansion board being used or if the expansion board is not using DMA channel 3, then the IEEE-488 interface may be jumpered to DMA channel 3 for higher performance.

Refer to the Installation section for information on using the expansion slot.

FUSES

The Comm/Chain Card has one 2 amp fuse located at the upper rear side of the board. If any fuse opens, remove the fault before replacing the fuse. Replacement fuse information is given in the Hardware Specifications section of this manual.

DISK LEDs

When lit, these two LEDs indicate disk activity. The floppy disk LED is visible on the floppy disk drive's bezel. The hard disk access LED is located on the top of the Comm/Chain Card.

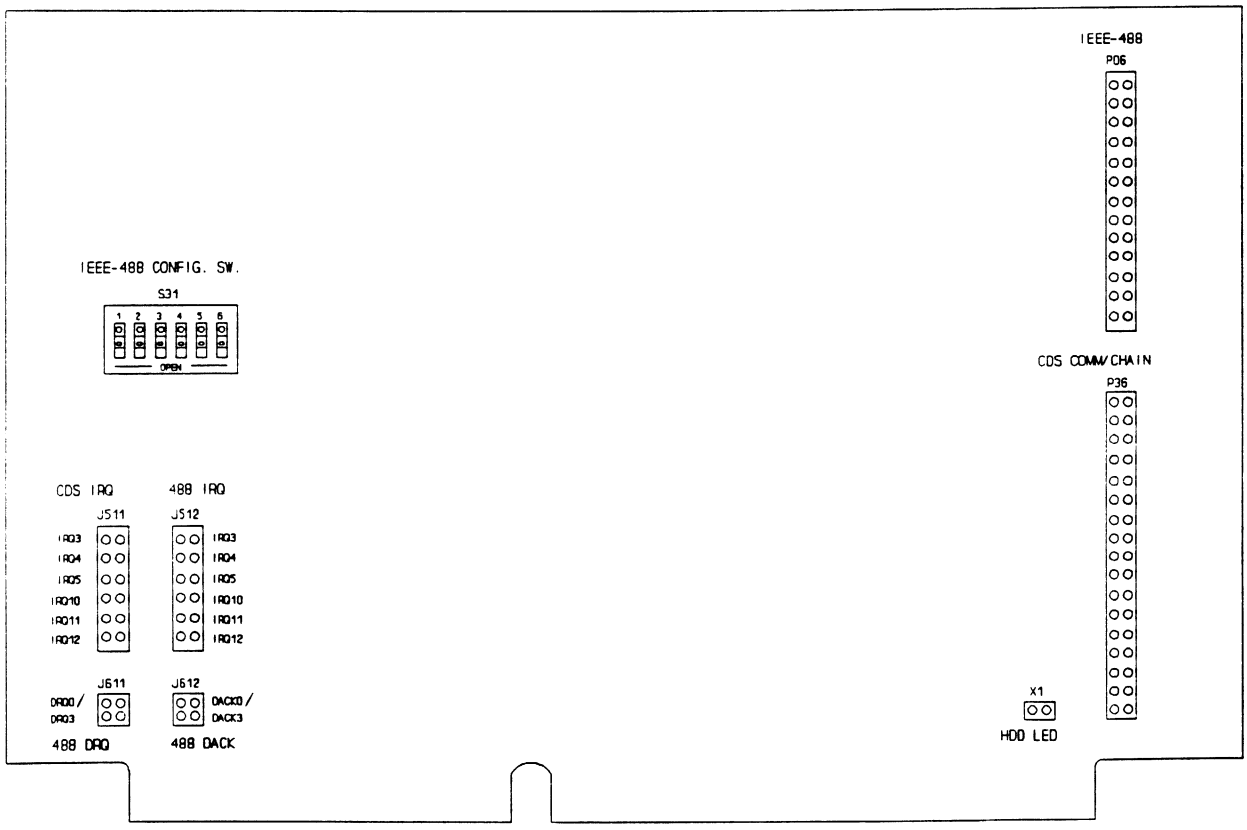


Figure 1: 53B-ATZ Card Block Diagram

SPECIFICATIONS

Function:	Embedded 386-20, CDSbus interface, and IEEE-488 System Controller.
Embedded PC-AT	
Manufacturer, P/N:	Ampro Computer SSB386.
CPU:	80386.
Clock:	20 MHz.
Printer Port:	One available (LPT1).
Serial Port:	Two available. Both accessible through a connector on the butch plate (COM1 and COM2). (COM2 is not available with Option 31 LCD Display.)
Memory:	80 nsec. dynamic RAM 4 Mbyte.
Wait States:	0 for on-board memory.
DMA Channels:	0 - 7.
Video Port:	VGA Adapter (not available with Option 31 LCD Display).
Floppy Disk:	1.44 Mbyte 3.5 inch floppy.
Hard Disk:	3.5 inch 40 Mbyte Winchester hard disk, 28 msec or better average access. 119 Mbyte hard disk available.
Hard Disk Interface:	IDE (ATA).
Co-processor:	Socket for optional 80387 math co-processor.
BIOS:	Award BIOS V3.03 or later.
DOS:	Microsoft V4.01 or later.
Expansion Slot:	PC-AT compatible. Selectable for 8 MHz or 3 MHz operation using the supplied setup software. The current available for an expansion card is 3.5 amps of 5 volts.
CDSbus Interface	
I/O Address:	200h through 207h.

IEEE-488 Interface

Design Type:	Compatible with Capital Equipment Corp. PC<>488 interface port.
I/O Address:	2B8h through 2BFh.
Firmware Address:	C8000h.
IEEE-488.1 Subsets Supported:	SH1 Source handshake. AH1 Acceptor handshake. TE5 Talker extended. LE3 Listener extended. SR1 Service request. DC1 Device clear. DT1 Device trigger. RL1 Remote/local. PP0 Parallel poll. C1-5 System controller. E1/2 Auto switch open collector / tri-state.
Addressing Capability:	Single primary/multiple secondary or multiple primary.
Power Requirements:	5V and ±15V dc power is provided by the internal Power Supply in the 53/63 Series card cage.
Voltage (5V Supply):	4.75 V dc to 5.25 V dc.
Current (5V Supply):	5 A, peak.
Voltage (±15V Supplies):	+14.64 Vdc to +15.6 V dc. -14.5 V dc to -15.5 V dc.
Current (±15V Supplies):	1 A, peak.
Replacement Fuse:	+5V: Littlefuse P/N 273005; CDS P/N 42202-73050.
Cooling:	Provided by the fans in the card cage.
Temperature:	5°C to +45°C, operating (assumes ambient temperature of 35° and airflow to assure less than 10°C temperature rise). -40°C to +85°C, storage.
Humidity:	Less than 95% R.H. non-condensing, -10°C to +30°C. Less than 75% R.H. non-condensing, +31°C to +40°C. Less than 45% R.H. non-condensing, +41°C to +55°C.
Dimensions:	The 53A-171 and 53A-153 Cards are shipped separately. 419.1 mm x 26.7 mm x 469.9 mm. (16.75 in x 10.5 in x 18.5 in).

Weight:	16.65 kg (37 lbs.)
Weight, Shipping:	The 53A-171 and 53A-153 Cards are shipped separately. 20.25 kg (45 lbs.)
Mounting Position:	Vertical position preferred, to minimize wear on the disk bearings.
Butch Plate Signal Connectors:	Video Port Socket (not available with Option 31 LCD Display). DE-9P Serial Port Plug (2). DB-25S Printer Port Socket. 5 pin DIN Keyboard Connector. 24 pin IEEE-488 Standard Connector.
Equipment Supplied:	1 - 53B-153 PC/386 Embedded Computer. 1 - DOS 4.01 (Part # 46999-40100) or DOS 5.0 (Part # 46999-50000). 1 - Backup Utility Files (Part # 41551-16020). 1 - 46999-00933 AT-compatible Keyboard. 1 - 53A-171 Control Card. 1 - 53B-003 Card Cage. 1 - 53B Power Supply Assembly (Part # 41550-54001).
Manuals Supplied:	1 - 53B-ATZ Operating Manual (Part # 00000-10031). 1 - 53B-ATZ Service Manual (Part # 00000-20031). 1 - Ampro SSB386 Technical Manual. 1 - 53A-171 Operating Manual (Part # 00000-11710). 1 - 53A-171 Service Manual (Part # 00000-21710). 1 - Ampro VGA Minimodule Technical Manual (or LCD Technical Manual for Option 31). 1 - CEC Programming Manual/Disk (Part # 47060-00004). 1 - Set of MS-DOS Manuals.
Optional Peripheral Equipment:	Flat Panel LCD Display. Ampro LCD Mini-Module Technical Manual.
Options:	Option 01: Math Co-processor. Option 02: 40 Mbyte Hard Disk. Option 03: 119 MByte Hard Disk. Option 14: 4 MByte RAM. Option 31: Flat Panel LCD Display. Option 30: Delete Hard Disk. Option 34: Windows™ Software * Option 35: TekTMS. Option 36: TekTMS, Windows™, and C * * available in U.S. only.

OPERATION

OVERVIEW

The 53B-ATZ System Controller is a CDSbus compatible embedded IBM PC-AT compatible computer with a high-speed CDSbus interface, an IEEE-488 system controller interface and a PC-AT compatible expansion slot. The 53B-ATZ is delivered with software I/O drivers to control the CDSbus interface and the IEEE-488 interface.

The PC-AT compatible processor is an Ampro Computer SSB386. Refer to the Ampro manual for technical information about the SSB386.

The 53B-ATZ is delivered with MS-DOS Version 4.01 or later installed on hard disk. All DOS utilities and commands are in a directory named 'DOS'. A set of MS-DOS Version 4.01 or later floppy disks with manuals is also included with the 53B-ATZ. The manuals describe the operation of MS-DOS, including how to load MS-DOS onto the hard disk.

NOTE: **Before any communication can take place on the CDSbus or the IEEE-488 bus the hardware must be initialized.** To do this, execute the program CDSINIT.EXE located in the CDSFILES directory on the hard disk. As shipped from the factory, the CDSINIT.EXE program is executed from the AUTOEXEC.BAT batch file, thereby initializing the CDSbus and IEEE-488 bus every time the computer is booted.

Devices on the CDS bus are accessed by the application program making calls to subroutines in the CDS drivers. The subroutines CSEND and CSENDF are used for writing data to a CDS instrument. CENTER and CENTERF are used for reading data from a CDS instrument. CTRANSMIT is used for changing system parameters such as timeouts and EOS (end of string) character. The CDS driver package used depends on the type of language performing the calls: interpreted BASIC, compiled BASIC, compiled C, or Turbo PASCAL.

With the exception of interpreted BASIC, the drivers are supplied as object modules which are linked to the user's application object module to generate a DOS executable program. For interpreted BASIC, the drivers are supplied on disk as a BLOADable file, named BIOCDS.M.

Devices on the IEEE-488 bus are accessed by the application program making calls to subroutines in the CEC-488 drivers. The subroutines SEND and SENDF are used for writing data to an IEEE-488 instrument. ENTER and ENTERF are used for reading data from an IEEE-488 instrument. With the exception of interpreted BASIC, the drivers require files on disk to create a DOS executable program. For interpreted BASIC, the drivers reside in read-only-memory (ROM) at physical address 0C8000 hex. Refer to the supplied CEC PC<>488 Programming Manual for information on using the IEEE-488 drivers.

BASIC PROGRAMMING

Driver files relevant to BASICA:

BIOCDS.M	BLOADable version of CDS drivers.
GPICDS.BAS	A sample program which demonstrates how to initialize and call the drivers from BASICA.

The descriptions and examples in this section assume the user is already experienced in BASIC programming. This section is not intended as a tutorial on BASIC syntax or programming, and the novice user is referred to the many excellent books available on the subject.

When operating in an interpreted basic environment (BASICA), the drivers are loaded into the computer's memory using the "BLOAD" command in one of two ways.

NOTE: When working in the Quick Basic environment, ARRAYS must be forced to remain STATIC whinth the Quick Basic data space. To accomplish this simply add the common command following the array dimension command.

Example:

DIM SHARED A% (500): COMMON A%().

The first example loads the drivers into BASICA's working space:

```
10 CLEAR ,50000!
     Protect memory above 50000 for CDS drivers.
20 IODRIVER = 50000!
30 BLOAD "BIOCDS.M", IODRIVER
     Load subroutines.
40 CINIT = IODRIVER
     Establish offsets for subroutines.
50 CSEND = IODRIVER + 3
60 CSENDF = IODRIVER + 6
70 CENTER = IODRIVER + 9
80 CENTERF = IODRIVER + 12
90 CTRANSMIT = IODRIVER + 15
100 CALL CINIT
     Initialize drivers.
110 user program code . . .
```

If the drivers are to be loaded into an absolute memory location (not in BASICA's 64K space), a DEF SEG command must first be executed to inform BASICA where to load the drivers. The following example shows how to load and initialize the drivers:

```
10 DEF SEG = &H5000
     Memory segment to load subroutine (typical example)
20 BLOAD "BIOCDS.M",0
     Load subroutine
30 CINIT = 0
     Establish offsets for subroutines
40 CSEND = 3
50 CSENDF = 6
60 CENTER = 9
70 CENTERF = 12
80 CTRANSMIT = 15
```

```
90 CALL CINIT
     Initialize drivers
100 user program code...
```

CAUTION:

This location of BIOCDS.M in memory may interfere with other TSR (Terminate and Stay Resident) programs. Be aware of what programs are loaded and their locations. If BIOCDS.M is loaded on top of a TSR program, unpredictable results will occur. The problem may not appear until BASIC is exited.

Addressing Cards

Before communication can take place between the application program and the CDS function cards, the function card must be addressed. To address a function card for the first time, the system command "@XY" must be issued. X is the card cage address (0-9) selected on the 53A-171 card for that cage. Y is the card address (0-9) within that card cage. The card will remain addressed until a new "@XY" command is issued. Use the CSEND command to address the function card:

```
110 INPUT "ENTER ADDRESS OF CDS FUNCTION CARD -> ";ADDR$
120 WRT$ = "@" + ADDR$
130 CALL CSEND(WRT$,ER%)
140 IF ER% <> 0 THEN PRINT "ERROR #";ER%;" RETURNED FROM DRIVERS"
```

In the above example, if 02 is entered at the prompt for an address, then cage 0 function card 2 would be addressed.

Handling Interrupts

When an instrument in a CDSbus system requires the attention of the application program, it will generate an RFI (Request For Interrupt) signal on the CDSbus backplane. This interrupt will in turn emulate a button press on the game port of the embedded PC/AT. This allows the user's program to take advantage of BASIC's STRIG function for trapping RFI interrupts on the CDSbus.

The files on disk named GPICDS.BAS (for Interpreted BASIC) and GPBCDS.BAS (for Compiled BASIC) give examples of how to use BASIC's STRIG function in conjunction with the CDS drivers for interrupting a BASIC program.

Compiled BASIC

Driver files relevant to Compiled BASIC:

BIOCDS.OBJ	CDS drivers object module that is linked with the user's application programs object module to create an executable file.
GPBCDS.BAS	A sample program demonstrating how to initialize and call the drivers from Compiled BASIC.
GPBCDS.EXE	Executable version of sample program.
BLDB.BAT	Batch file for creating the .EXE file.

Detailed descriptions of the drivers are in alphabetical order on the following pages.

Driver: CENTER (read data)

Syntax: CALL CENTER (RD$, RDLEN%, STATUS%)

Purpose: Reads data FROM a CDS instrument TO a string variable.

Description:

RD$ String variable which will contain the data read from the CDS instrument. RD$ must be initialized to a string containing at least as many characters as will be received.

RDLEN% This variable is returned to the caller and contains the actual number of characters read. It can be used to trim RD$ to the correct length.

STATUS% Argument returned to caller. It will contain one of the following values:

- 0 = no faults or errors
- 1 - 3 = interface problem
- 5 = invalid parameter in CENTER command
- 8 = timeout error on the CDS device
- 9 = string is full before EOS character was received. More input may be required.

CENTER will terminate reception of data on any of the following:

1) the string is full.
2) an EOS character is received with the BINARY line on the CDSbus inactive (see EOS character under CTRANSMIT).
3) the timeout limit to transfer the entire string is exceeded.

Example:

```
1010 RD$ = SPACE$(20)
1020 CALL CENTER (RD$,RDLEN%,STATUS%)
1030 IF STATUS% <> 0 THEN GOSUB 2000
1040 PRINT LEFT$(RD$,RDLEN%)
```

This example sets up a string variable (RD$) of length 20.

CENTER is then called. Two error checks are performed on return from CENTER. The first checks STATUS%. If the integer is any value except zero, the error routine at line 2000 is called. The error routine will parse STATUS% to determine the nature of the error.

With error checking completed, the program formats the data using the LEFT$ command and prints the data to the display.

Driver: CENTERF (read data and write to a file)

Syntax: CALL CENTERF (FLNAME$, STATUS%)

Purpose: Reads data (ASCII or binary) FROM a CDS device TO a file on disk.

Description:

FLNAME$ Indicates the filename where data will be written. FLNAME$ may be up to 50 characters long, including a drive and path designation.

STATUS% Argument returned to caller. It will contain one of the following values:

0 = no faults or errors
1 - 3 = interface problem
5 = invalid parameter in CENTERF command
8 = timeout error
13 = path not found
14 = no handles available
15 = file access denied
16 = invalid file handle
20 = insufficient disk space
22 = invalid access code for file

The CENTERF command has extensive file management capabilities. The command will search for the defined filename (FLNAME$). If found, that file will be opened and the new data read from the CDS device will be written to the file.

Note that CENTERF does not operate in an append mode. That is, data will not be appended to an existing file. If an existing file is specified, the data in that file will be overwritten.

If the file is not found, CENTERF will create the file and open it. The program must not open the file to be used; CENTERF will open the file. Once the data transfer is finished, CENTERF will close the file.

CENTERF will terminate on any of the following:

1) an error is detected
2) an EOS character is received with the BINARY line on the CDSbus inactive (if this option is enabled)
3) the time out limit is exceeded for reading back all data

Example:

```
100  RD$ = SPACE $ (255)
110  FILENAME$ = "a:\TESTDATA\TEST1.DAT"
120  CALL CENTERF (FILENAME$,STATUS%)
130  IF STATUS% >10 THEN GOSUB 2000
140  IF STATUS% <>0 THEN GOSUB 3000
150  ...
```

In this example, a file called TEST1.DAT on the A: disk drive in subdirectory \TESTDATA will be opened. If the subdirectory is not found, CENTERF will stop and display error code 13. If the subdirectory is found but the file is not found, CENTERF will create the file and proceed with no errors.

The status byte check divides the errors into two groups: file errors (line 2000) and interface errors (line 3000).

Driver: CINIT (initialize drivers)

Syntax: CALL CINIT

Purpose: CINIT needs to be called at the beginning of the program to initialize the drivers.

Description: CINIT initializes the following to their default state:

timeout = 2 seconds
EOS character = line feed

Example: 1000 CALL CINIT

Driver: CSEND (write data)

Syntax: CALL CSEND (WRT$, STATUS%)

Purpose: Sends data FROM a string variable TO a CDS device.

Description: WRT$ Data string to be sent.

STATUS% Argument returned to caller. It will contain one of the following values:

0 = no faults or errors
1 - 3 = interface problem
5 = invalid parameter in CSEND command
8 = timeout error on the CDS device

Example:

```
1000 WRT$ = "R1" + CHR$(13) + CHR$(10)
1020 CALL CSEND (WRT%,STATUS%)
1030 IF STATUS% <> 0 THEN GOSUB 2000
```

In this example, the string "R1" is sent to the presently addressed function card. Note that <CR><LF> must be specifically appended to the string to be sent, using the CHR$( ) command.

When CSEND returns, a STATUS <> 0 will branch to the error routine at line 2000.

Driver: CSENDF (send data from a file)

Syntax: CALL CSENDF (FLNAME$, STATUS%)

Purpose: Sends data (ASCII or binary) FROM a file on disk TO a CDS device.

Description:

FLNAME$ Indicates the filename from which data will be read. FLNAME$ may be up to 50 characters long, including a drive and path designation. If the file does not exist, a status error will be reported.

STATUS% Argument returned to caller. It will contain one of the following values:

0	= no faults or errors
1 - 3	= interface problem
5	= invalid parameter in CSENDF command
8	= timeout error on the CDS device (for the entire write operation)
12	= file not found
13	= path not found
14	= no handles available
15	= file access denied
16	= invalid file handle
20	= insufficient disk space
22	= invalid access code for file

CSENDF will terminate on an error, including any timeout error, or EOF.

Example:

```
100  FILENAME$ = "C:\THISWEEK\TESTDATA\TEST5.DAT"
120  CALL CSENDF (FILENAME$, STATUS%)
130  IF STATUS% <> 0 THEN GOTO 2000
140  ...
```

In this example, the file will be opened if not already open, read until an interface error occurs or until the file EOF mark is found. Error 12 is generated if the file is not found; this command will not create a new file.

Driver: CTRANSMIT (send command to interface)

Syntax: CALL CTRANSMIT (COMMAND%, DAT%, RESPONSE%, STATUS%)

Purpose: Allows parameters such as EOS character and Timeout to be defined by the caller.

Description: COMMAND% This variable contains a number which specifies the command to be executed:

1 = VERSION - Returns the version number of the PC-AT drivers in the RESPONSE% variable as an unformatted integer. For example, Version 1.00 is returned as 100; V3.11 is returned as 311.

2 = TIMEOUT - Sets the timeout with the value passed in DAT%. DAT% = the number of 0.1 second ticks that can occur before the transfer is aborted. The default timeout is 2 seconds (or timeout = 20) if this command is never executed. For multiple array transfer commands, the timeout value applies to each array. If the timeout value is exceeded before the completion of CSEND, CENTER, CSENDF, or CENTERF, then a timeout status will be returned to the calling program. The maximum allowed timeout value is 32,767, which is about 1 hour.

8 = EOS - Sets the EOS terminator with the value passed in DAT%. The default EOS terminator is an ASCII Linefeed (0Ah) if this command is never executed. The EOS terminator will cause readback (CENTER or CENTERF) to stop only if the BINARY line on the CDSbus is inactive.

DAT% Refer to the individual command descriptions for the functions affected by this parameter.

RESPONSE% The Response parameter is only valid for the VERSION command. However, this field must be included in all calls to CTRANSMIT in order to keep the parameters in the proper order.

STATUS% - Argument returned to caller. It will contain one of the following values:

0 = no faults or errors
1 - 3 = interface problem
5 = invalid parameter in CTRANSMIT command
8 = timeout error on the CDS device

Examples:

```
170 COMMAND%=1
180 CALL CTRANSMIT(COMMAND%,DAT%,RESPONSE%,STATUS%)
190 PRINT "USING CDS DRIVERS VERSION,#.##";(RESPONSE%/100)
200 COMMAND%=8
210 DAT%=13
220 CALL CTRANSMIT(COMMAND%,DAT%,RESPONSE%,STATUS%)
230 IF STATUS%<>0 THEN GOTO 2000
```

In this example, the program is verifying and initializing the interface to the desired operating parameters. The version of the drivers will be printed on the display. The EOS character is changed to a carriage return <CR>.

C PROGRAMMING

Driver files relevant to C:

CIOCDS.OBJ	CDS drivers object module that is linked with the user's application programs object module to create an executable file.
CINT.TXT	Example of how to handle RFI interrupts in C.
GPCCDS.C	A sample program demonstrating how to initialize and call the drivers from Compiled C.
GPCCDS.EXE	Executable version of sample program.
BLD.BAT	Batch file for creating the .EXE file

Detailed descriptions of these drivers are given in alphabetical order on the following pages.

C compilers compatible with Microsoft C version 5.0 or later can be used with the CDS drivers. The application program should be compiled as a Large memory model to generate the application object module. The C link program is then used to combine the application object module with the driver object module (CIOCDS.OBJ) in order to create a DOS executable program.

The descriptions and examples in this section assume the user is already experienced in C programming. This section is not intended as a tutorial on C syntax or programming, and the novice user is referred to the many excellent books available on the subject.

Addressing Cards

Before communication can take place between the users application program and the CDS function cards, the function card must be addressed. To address a function card for the first time, the system command "@XY" must be issued. X is the card cage address (0-9) selected on the 53A-171 card for that cage. Y is the card address (0-9) within that card cage. The card will remain addressed until a new "@XY" command is issued. Use the CSEND command to address the function card:

```
strcpy(wrtstr,"@02") ;
status = csend(wrtstr);
if (status) cerror(status);
```

The above example will address card 2 in cage 0.

Handling Interrupts

When an instrument in a CDSbus system requires the attention of the application program, it will generate an RFI on the CDSbus backplane. This interrupt will in turn generate an interrupt to the embedded PC/AT computer on hardware interrupt line 10. This is similar to the serial port generating a hardware interrupt on line 4 when COM1 needs service.

The file on disk named CINT.TXT gives an example of how to develop and install an interrupt handler in the embedded PC/AT to handle hardware interrupt 10.

Driver Descriptions

Driver: CENTER

Syntax: int center(rdstr,bufsize(rdstr),&count) ;

Purpose: Reads data FROM a CDSbus device TO a string.

Description:

char rdstr[xxx] storage buffer which will contain the data read from the CDSbus device.

int bufsize size of storage buffer, in bytes.

int count count is returned to the caller and contains the actual number of characters read.

CENTER will terminate on any of the following:

1) the storage buffer is full
2) an EOS character is received with the BINARY line on the CDSbus inactive (see EOS character under CTRANSMIT)
3) the timeout limit to transfer the entire string is exceeded

CENTER will add a null to the end of the storage buffer, provided it was not full when the last character was written. A variable set equal to the function CENTER will return the status of the transfer. The status value returned is described at the end of this section.

Example:

```
char rdstr[300] ;
int count ;
int status ;
int addr ;

main ()
{
  .
  .
  status = center(rdstr, sizeof(rdstr), &count) ;
  if (status) cerror(status) ;
  else printf("%s",rdstr) ;
  .
  .
}
```

The example above reads a data string from the presently addressed CDS device into the storage buffer rdstr[xxx]. The status of the transfer is returned to the variable status. If the status is non-zero, the function cerror is executed based on the value of status . An example showing the function cerror is provided at the end of the section.

Driver: CENTERF

Syntax: int centerf((char far *) flname) ;

Purpose: Reads data (ASCII or binary) FROM a CDSbus device TO a file on disk.

Description: char flname[xx] indicates the filename to which data will be written. flname may be up to 50 characters long, including a drive and path designation.

The CENTERF command has extensive file management capabilities. The command will search for the defined filename (FLNAME$). If found, that file will be opened and the new data read from the CDSbus device will be written to the file.

Note that CENTERF does not operate in an append mode. That is, data will not be appended to an existing file. If an existing file is specified, the data in that file will be overwritten.

If the file is not found, CENTERF will create the file and open it. The program must not open the file to be used; CENTERF will open the file. Once the data transfer is finished, CENTERF will close the file.

CENTERF will terminate on any of the following:

1) an error is detected
2) an EOS character is received with the BINARY line on the CDSbus inactive.
3) the timeout limit is exceeded for the entire read operation

Example:

```
char *flname  ;
int status ;
int addr ;

main ()
{
  .
  .
  status = centerf((char far *) flname)  ;
  if (status) cerror(status) ;
  .
  .
}
```

The example above reads data from the presently addressed CDSbus device and stores it in the file flname . The status of the transfer is returned to the variable status. If the status is non-zero, the function verror is executed based on the value of status . An example showing the function cerror is provided at the end of the section.

Driver: CINIT

Syntax: void cinit() ;

Purpose: CINIT needs to be called at the beginning of the program to initialize the drivers.

Description: CINIT will initialize the timeout for two seconds, and set the EOS character to a line feed.

Example:

```
main ()
{
   .
   .
   cinit();
   .
   .
}
```

Driver: CSEND

Syntax: int csend((char far *) wrtstr) ;

Purpose: Sends data FROM a string TO a CDSbus device.

Description: char wrtstr[xxx] data string to be sent.

CSEND will determine the string length by searching for a null.

A variable set equal to the function CSEND will return the status of the transfer. The status value returned is described at the end of the section.

Example:

```
char wrtstr[100] ;
int status ;

main ()
{
  .
  .
  status = csend((char far *) wrtstr)  ;
  if (status) cerror(status) ;
  .
  .
}
```

The example above sends the data string wrtstr to the presently addressed CDSbus device. The status of the transfer is returned into the variable status . If the status is non-zero, the function cerror is executed based on the value of status .

Driver: CSENDF

Syntax: int csendf((char far *) flname) ;

Purpose: Sends data (ASCII or binary) FROM a file on disk TO the presently addressed CDSbus device.

Description:

char flname[xx]	indicates the filename from which data will be read. flname may be up to 50 characters long, including a drive and path Designation.

A variable set equal to the function CSENDF will return the status of the transfer. The status value returned is described at the end of the section.

Example:

```
char *flname  ;
int status ;

main ()
{
  .
  .
  status = csendf((char far *) flname)  ;
  if (status) cerror(status) ;
  .
  .
}
```

The example above sends data from the file flname to the presently addressed CDSbus device. The status of the transfer is returned into the variable status . If the status is non-zero, the function cerror is executed based on the value of status .

Driver: CTARRAY

Syntax: int ctarray ((char far *) wrtstr, count, end) ;

Purpose: Sends binary data FROM an integer array TO the presently addressed CDSbus device.

Description:

char wrtstr[xxx]	data string to be sent.
int count	specifies the number of bytes to be transmitted.

The size of the char data string is limited to 65535 bytes. If the CDSbus device is expecting more than that number, the data can be stored in two or more buffers.

A variable set equal to the function CTARRAY will return the status of the transfer. The status value returned is described at the end of the section.

Example:

```
char data [1000] ;
int count ;

count = 1000 ;

status = ctarray(data,count) ;
if(status) cerror(status) ;
```

The example above sends data from the integer array data to the presently addressed CDSbus device The status of the transfer is returned into the variable status . If the status is non-zero, the function verror is executed based on the value of status .

Driver: CTRANSMIT

Syntax: int ctransmit (command, data, &response)

Purpose: Allows parameters such as EOS character and Timeout to be defined by the caller.

Description: int data

int command This argument contains a number which specifies the command to be executed:

1 = VERSION - Returns the version number of the PC-AT drivers to the response argument.

2 = TIMEOUT - Sets the timeout with the value passed in the data argument. The data argument = the number of 0.1 second ticks that can occur before the transfer is aborted. The default timeout is 2 seconds (timeout = 20) if this command is never executed. The maximum timeout is 32.767. If the timeout value is exceeded before the completion of CSEND, CENTER, CENTERF or CTARRAY, then a timeout status will be returned to the calling program.

8 = EOS - Sets the EOS terminator with the value passed in the data argument. The default EOS terminator is an ASCII Linefeed (0Ah) if this command is never executed.

Example:

```
timeout()
{
  printf("Enter number of .1 second ticks for timeout ->") ;

  instr = cgets(inbuffer) ;
  i = atoi(instr)   ;
  status = ctransmit(2,i,&response) ;
  if (status) cerror(status) ;
}
```

This example shows how to assign a timeout value. The printf statement requests input of a timeout value. This value is connected to an integer and assigned in the vtransmit call.

Status Checking

All calls to the driver subroutines with the exception of CINIT will return the following status to the caller:

0 = everything went ok
1 - 3 = interface problem
5 = invalid parameter in CTRANSMIT command
8 = timeout error on the CDSbus device
9 = buffer was full before EOS was received. More input may be required.
12 = file not found
13 = path not found
14 = no handles available
15 = file access denied
16 = invalid file handle
20 = insufficient disk space
22 = invalid access code for file

Example:

```
int status ;
char wrtstr[100] ;
int status ;        /* returned by drivers */

main ()
{
  .
  .
  status = csend((char far *) wrtstr)  ;
  if (status) cerror(status) ;
  .
  .
}
cerror(status)
{
  switch (status)
   {
   case 1:  printf("\nerror - Interface error\n\n"); break ;
   case 2:  printf("\nerror - Interface error\n\n"); break ;
   case 3:  printf("\nerror - Interface error\n\n"); break ;
   case 5:  printf("\nerror - command error for transmit\n\n"); break;
   case 8:  printf("\nerror - timeout \n\n"); break ;
   case 12: printf("\nerror - file not found\n\n"); break ;
   case 13: printf("\nerror - path not found\n\n"); break ;
   case 14: printf("\nerror - no handles available\n\n"); break ;
   case 15: printf("\nerror - file access denied\n\n"); break ;
   case 16: printf("\nerror - invalid file handle\n\n"); break ;
   case 20: printf("\nerror - insufficient disk space\n\n"); break ;
   case 22: printf("\nerror - invalid access code for file\n\n"); break;
  default:          break  ;
   }
}
```

The example above shows a detailed handling of a status error for the csend command.

PASCAL PROGRAMMING

Driver files relevant to Turbo PASCAL:

PIOCDS.OBJ	CDS drivers object module that contains the external procedures called by the Turbo PASCAL application program. PIOCDS.OBJ is linked with the user's object module to create an executable file.
PINT.TXT	Example of how to handle ATZ interrupts in Turbo PASCAL.
GPPCDS.PAS	A sample program which demonstrates how to initialize and call the drivers from Turbo PASCAL.
GPPCDS.EXE	Executable version of sample program.

The application program should include the lines {$L PIOCDS} and {$F} to inform the Turbo compiler to link in the assembly language object file (PIOCDS.OBJ) and perform far calls to any external procedures in the application program. Refer to the program GPPCDS.PAS for an example of how the drivers are integrated into and called by an application program.

The descriptions and examples in this section assume the user is already experienced in Pascal programming. This section is not intended as a tutorial on Pascal syntax or programming, and the novice user is referred to the many excellent books available on the subject.

Addressing Cards

Before communication can take place between the user's application program and the CDS function cards, the function card must be addressed. To address a function card for the first time, the system command "@XY" must be issued. X is the card cage address (0-9) selected on the 53A-171 card for that cage. Y is the card address (0-9) within that card cage. The card will remain addressed until a new "@XY" command is issued. Use the CSEND command to address the function card:

```
WRTSTRING: = '@02';
CSEND (WRTSTRING, STATUS);
```

Handling Interrupts

When an instrument in the CDSbus system requires the attention of the application program, it will generate an RFI on the CDSbus backplane. This interrupt will in turn generate an interrupt to the embedded PC/AT computer on hardware interrupt line 10. This is similar to the serial port generating hardware interrupt line 4 when COM1 needs service.

The file on disk named PINT.TXT gives an example of how to develop and install an interrupt handler in the embedded PC/AT to handle hardware interrupt 10.

PASCAL Procedures

Driver: CINIT

Syntax: procedure Cinit;external;

Purpose: CINIT must be called at the beginning of the program to initialize the drivers, as shown in the example below.

Example:

```
procedure  cinit; external;
begin
      cinit;
      .
      .
      .
end
```

Driver: CSEND

Syntax:
```
procedure Csend(wrtstr: string ;
status : integer);external;
```

Purpose: Sends data FROM a string TO a CDSbus device.

Description:

wrtstr — Pointer to string. The first byte of string is the length, and the remaining bytes are data.

Example:
```
Write('Enter string to send -> ');
ReadLn(wrtstr);
Csend(wrtstr,cstatus);
if cstatus <> 0 then Cerror(cstatus);
```

This example prompts the user to enter data, reads the data into the wrtstr buffer and then sends the data to the presently addressed CDSbus device. On return from Csend, if the status is not equal to zero, then the error handler Cerror is called.

Driver: CSENDF

Syntax:
```
procedure Csendf(flname: string ;
status : integer);external;
```

Purpose: Sends data (ASCII or binary) FROM a file on disk TO a CDSbus device.

Description:

flname — Indicates the filename from which data will be read. flname may be up to 50 characters long, including a drive and path designation.

Csendf will terminate on any error, including a timeout error or when the file transfer is complete.

Example:
```
Write('Enter name of file to Read -> ') ;
Readln(flname);
Csendf(addr,flname,status);
if cstatus <> 0 then Cerror(cstatus);
```

This example prompts the user for a filename, reads in the filename, then sends data from the file to the presently addressed CDSbus device. On return from Csendf, if the status is not equal to zero, then the error handler Cerror is called.

Driver: CENTER

Syntax:
```
procedure Center(rdstr: string ;
rdlen: integer ;
ccount: integer ;
status: integer);external;
```

Purpose: Reads ASCII characters FROM a CDSbus device TO a string.

Description:

rdstr — Pointer to storage buffer that will contain the data read from the CDSbus device.

rdlen — Size of storage buffer.

ccount — count is returned to the caller and contains the actual number of characters read.

CENTER will terminate on any of the following:

1) the storage buffer is full.
2) an EOS character is received with the BINARY line on the CDSbus inactive (see EOS character under CTRANSMIT).
3) timeout limit to transfer the entire string is exceeded.

Example:
```
Center(rdstr,Sizeof(rdstr),ccount,cstatus);
if vstatus <> 0 then Cerror(cstatus)
else Writeln('Data received = ',rdstr);
```

This example reads a data string from the presently addressed CDSbus device into the storage buffer rdstr . On return from Center if the status is not equal to zero, then the error handler Cerror is called.

Driver: CENTERF

Syntax: procedure Centerf(flname: string ;
status : integer);external;

Purpose: Reads data (ASCII or binary) FROM a CDSbus device TO a file on disk.

Description: flname Indicates the filename from which data will be read. Flname may be up to 50 characters long, including a drive and path designation.

The CENTERF command has extensive file management capabilities. The command will search for the defined filename (FLNAME$). If found, that file will be opened and the new data read from the CDSbus device will be written to the file.

Note that CENTERF does not operate in an append mode. That is, data will not be appended to an existing file. If an existing file is specified, the data in that file will be overwritten.

If the file is not found, CENTERF will create the file and open it. The program must not open the file to be used; CENTERF will open the file. Once the data transfer is finished, CENTERF will close the file.

CENTERF will terminate on any of the following:

1) An error is detected.
2) An EOS character is received with the CDSbus Binary line inactive (see EOS character under CTRANSMIT).
3) The timeout limit is exceeded for the entire read operation.

Example:
```
Write('Enter name of file to Write -> ') ;
Readln(flname);
Centerf(flname,cstatus);
if cstatus <> 0 then Cerror(cstatus);
```

The example above reads data from the presently addressed CDSbus device and stores it in the file flname . The status of the transfer is returned to the variable status. If the status is non-zero, the function cerror is executed based on the value of status . An example showing the function cerror is provided at the end of the section.

Driver: CTRANSMIT

Syntax:
```
procedure Ctransmit(command: integer ;
data: integer ;
status : integer);external;
```

Purpose: Allows parameters such as EOS character and Timeout to be defined by the caller.

Description: command This argument contains a number which specifies the command to be executed:

1 = VERSION - Returns the version number of the CDS drivers to the response argument.

2 = TIMEOUT - Sets the timeout with the value passed in the data argument. The data argument equals the number of 0.1 second ticks that can occur before the transfer is aborted. The default timeout is two seconds if this command is never executed. If the timeout value is exceeded before the completion of CSEND, CENTER, or CENTERF, then a timeout status will be returned to the calling program.

8 = EOS - Sets the EOS terminator with the value passed in the data argument. The default EOS terminator is an ASCII linefeed (0Ah) if this command is never executed.

Example:
```
{change timeout from default of two seconds to something new}
 Write('Enter number of 0.1 second ticks for timeout -> ');
 Readln(i);
 Ctransmit(2,i,cstatus);
 if cstatus <> 0 then Cerror(cstatus);
```

This example shows how to assign a timeout value. The write statement requests input of a timeout value, and then calls ctransmit with the new timeout value.

Status Checking

All calls to the driver subroutines, with the exception of CINIT, will return the following status to the caller:

0 = no problems or errors
1 - 3 = interface error
5 = invalid parameter in CTRANSMIT command
8 = timeout error
9 = buffer full before EOS received
12 = file not found
13 = path not found
14 = no handles available
15 = file access denied
16 = invalid file handle
20 = insufficient disk space
22 = invalid access code for file

Example:

```
procedure Cerror(cstatus : integer);
 begin
     Writeln;
     case cstatus of
     1:  Write('error - interface error');
     2:  Write('error - interface error');
     3:  Write('error - interface error');
     5:  Write('error - command error for ctransmit');
     8:  Write('error - timeout ');
     9:  Write('error - buffer full , more input required');
     12: Write('error - file not found');
     13: Write('error - path not found');
     14: Write('error - no handles available');
     15: Write('error - file access denied');
     16: Write('error - invalid file handle');
     20: Write('error - insufficient disk space');
     22: Write('error - invalid access code for file');
     end;
     Writeln;
 end;

 begin
   .
   .
 Csend(wrtstr,cstatus);
 if cstatus <> 0 then Cerror(cstatus);
   .
   .
   .
 end;
```

The example above shows a detailed handling of a status error for the csend command.

INSTALLATION REQUIREMENTS AND CAUTIONS

Only qualified personnel should perform this installation.

CAUTION:

The 53B-ATZ is a piece of electronic equipment and therefore has some susceptibility to electrostatic damage (ESD). ESD precautions must be taken whenever module sub-assemblies are handled.

INSTALLATION PROCEDURE

Mechanical Installation

The left-most slot of the 53B Card Cage is the floppy disk slot. The 53B-153 Comm/Chain Card has green ejectors and goes in the second slot from the left, which has a green decal. The 53A-171 Control Card has black ejectors and goes in the third slot from the left, which has a black decal.

CAUTION:

Note that there are two ejectors on each card. Make sure the ejectors marked "53B-153" and "53A-171" are at the top.

1) Be sure power is off in the chassis.

2) Insert the Comm/Chain Card into the card guides and slide it in about half way.

3) Attach the ribbon cable to the connector on the lower front part of the card. Use a cable tie to fasten the ribbon cable to the tie base.

   NOTE: The ribbon cable should be attached with the red wire toward the bottom of the card.

4) Insert the card the rest of the way into the Comm/Chain slot.

5) Insert the 53A-171 Control Card into the control slot.

6) The 53B-ATZ requires that the video monitor be connected to the VIDEO connector on the butch plate of the 53B-ATZ. If the unit has an Option 31 LCD Display, follow the installation procedure listed below.

7) As shipped, an AT-compatible keyboard is required for an errorless power-up sequence. If the module is to be used without the keyboard, it must still be powered up the first time with a keyboard. Plug the keyboard cable into the butch plate connector labeled "KBRD". With a change in setup, the 53B-ATZ may subsequently be used without a keyboard.

If the System Controller is to be used without the keyboard, in a production test environment for example, use the Setup program described below to prevent the Ampro SSB386 from hanging up during boot-up when it detects that the keyboard is missing. The ERROR HALT field of the Setup menu may be scrolled to select NO HALT ON KEYBOARD. The Setup screen has instructions on how to change items. For CMOS memory screens, refer to Appendix C.

Installing Option 31 - LCD Display

Use the following procedure to install the LCD Display:

1. Align the LCD Display on the hinges that are attached to the top of the card cage. If necessary, loosen the nuts on the hinges to align the display.

2. Attach the LCD Display to the hinges with the supplied hinge pins. The hinge pins are inserted from the outside towards the inside of the card cage.

3. Adjust the fit of the Display to the front opening of the card cage, and tighten the nuts that secure the hinges.

4. While holding the Display in the fully open position, cut the cable tie holding the display support arm.

5. Extend the display support arm and attach the arm to the display using a supplied thumb nut.

6. Remove the two nuts on the bottom, front bar of the card cage, and mount the two supplied spring clips to the studs. The offset at the end of the spring clip should be pointing down. Replace and tighten the nuts.

7. Refer to Figure 2 (with Display in extended position or removed).

   A. Connect the Display's power connector to the panel jack on the switch plate.

   B. Connect the LCD Display cable (16 conductor ribbon) to the 16-pin header (P54), with the red wire (pin 1) towards the hex keypad. Use a supplied cable tie to secure the cable to the LCD Display.

   C. Connect the 9-pin female D-sub connector (labeled 'P30') to the 9-pin male connector P30 on the LCD Display, and connect the other end to the COM2 connector located on the butch plate.

   D. Connect the keyboard extension cable (5-pin keyboard connector) to the keyboard extension connector (P50), and connect the other end to the Keyboard Input connector on the butch plate.

   E. Use a supplied cable tie to secure both the keyboard extension cable and the hex keypad cable to the tie base on the display.

8. Lower the Display to its operating position, pushing excess cable into the cable tray, and turn on the card cage power.

9. Lift the Display slightly to adjust the contrast control, if needed.

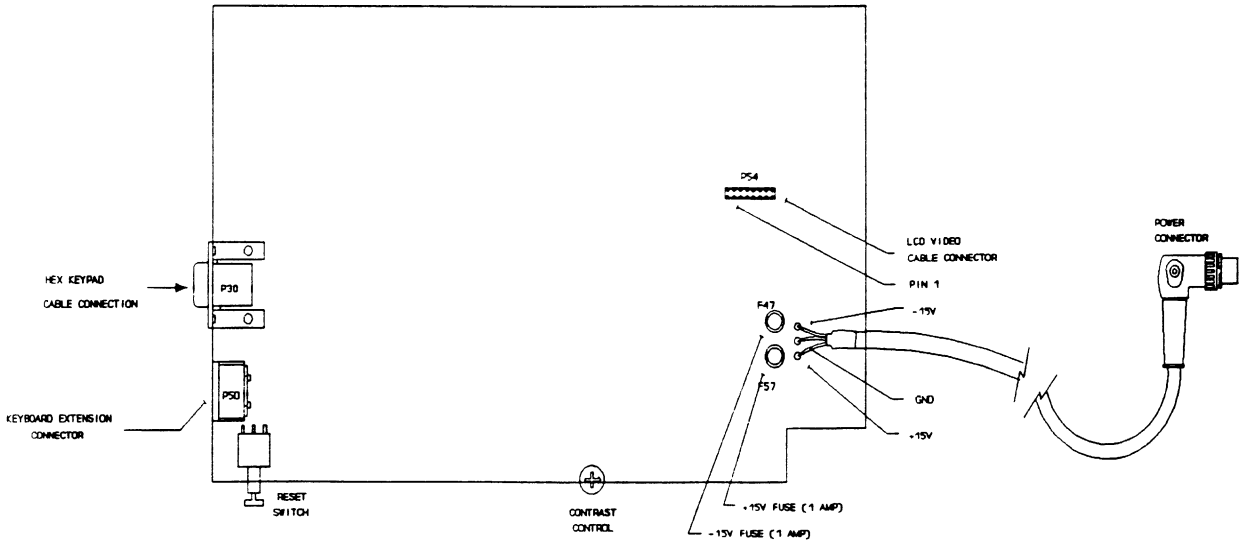


Figure 2: Controls and Indicators

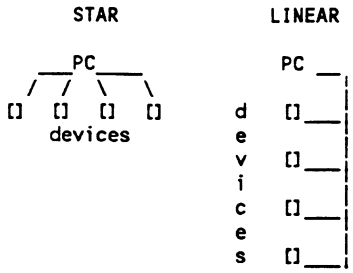
Installing GPIB Cables on the Butch Plate

The 53B-ATZ uses connecting nuts with metric threads, in accordance with the IEEE 488-1978 standard. Any cables attached to the unit must have metric threads. Cables manufactured with English threaded screws typically have bright screws, and cables manufactured with metric threads typically have dark screws.

After the cable is installed, check to see that all devices on the GPIB have different addresses. Address assignments are usually made by setting a small group of switches near the device's GPIB connector. The device address of the IEEE-488 interface port is set under program control to any valid address (0 through 30, typically 0) by calling the INITIALIZE routine.

The IEEE-488 interface port is guaranteed to drive fifteen devices with a total cable length of 20 meters, or two meters times the number of devices, whichever is less. Longer cables will usually work but are not recommended.

A STAR cabling topology minimizes worst-case transmission path lengths but concentrates the system capacitance at a single node. A LINEAR cabling topology produces longer path lengths but distributes the capacitive load. Combinations of star and linear cabling configurations are also acceptable.



Final Installation and Configuration

Turn on the card cage power and the video monitor power.

If the display is correct (legible), but an error message is displayed, check the system configuration by running the built-in Setup program.

When the 53B-ATZ boots up, it will be in the MS-DOS command mode. This is indicated by the C:> prompt. At this point, the system is ready to be configured to the user's software requirements.

Adding an IBM AT-Compatible Expansion Card

Located in the rear of the 53B-ATZ is an IBM AT-compatible, full size, 16-bit expansion slot. Use the following procedure to install an expansion card:

1. Refer to the AMPRO computer manual and make sure the expansion card does not have any conflicts with existing memory addresses, I/O addresses, DMA channels or Interrupt levels being used by the system. In addition to the above resouces used by the computer, the following resources are used by the IEEE-488 port and the CDSbus port:

    A. Memory addresses 0C8000h through 0CFFFFh. (IEEE-488 Prom)
    B. I/O addresses 02B8h through 02BFh. (IEEE-488 port registers)
    C. I/O addresses 0200h through 0207h. (CDSbus port registers)
    D. DMA channel 1. (Used by CDSbus hardware)
    E. IRQ 10. (Used by CDSbus hardware)

2. Be sure power is off in the chassis.

3. For easiest access to the expansion slot turn the chassis upside down.

4. Remove the four screws that secure the butch plate to the chassis. The butch plate will now hinge away from the chassis and be suspended by the cabling.

5. Insert the expansion card into the expansion slot. If the card is full size, make sure the card edge opposite the mounting panel is secured by the card guide.

6. Use a #6-32 screw to attach the mounting panel of the expansion card to the mounting bracket.

7. Re-attach the butch plate to the chassis.

Installing the Drivers

The 53B-ATZ is configured at the factory with all CDSbus I/O drivers, the CEC IEEE-488 interface port drivers, the Ampro utilities, and DOS 4.1 or later. This software is also supplied on a floppy disk, which should be archived.

The supplied drivers on the hard disk are organized in several sub-directories. Refer to the Driver File Locations table for an outline of the files on the hard disk.

If you intend to use software written for the IEEE-488 interface port by a third party manufacturer, check to insure that the software is written for the GPIB interface chip included with the 53B-ATZ, an NEC 7210.

Driver File Locations

Root Directory

CDSFILES

- Sub BASICA
  - BIOCDS.M
  - GPICDS.BAS
- Sub C
  - BLD.BAT
  - CIOCDS.M
  - GPCCDS.C
  - GPCCDS.EXE
- Sub PASCAL
  - GPPCDS.EXE
  - GPPCDS.PAS
  - PIOCDS.OBJ
- Sub QB
  - BIOCDS.OBJ
  - BLDB.BAT
  - GPBCDS.BAS
  - GPBCDS.EXE

Files
- BIOCDS.TXT
- CDSINIT.EXE
- CINT.TXT
- CIOCDS.TXT
- PIOCDS.TXT
- PINT.TXT
- README.TXT

DOS

This sub-directory contains all DOS commands, drivers, etc.

AMPRO

- Sub VGA
  This contains all utility files for MiniModule / VGA Board.
- Sub LCD
  This contains all utility files for the Ampro MiniModule LCD Board (Option 31 only)

Files
All utility files for SSB386 Board.

CEC

This directory and its sub-directories contain all GPIB drivers and support files supplied by CEC.

Setup Program/Changing Configuration

Figure 3 shows an example of the Setup screen, with the standard configurations for the 53B-ATZ. The setup program may be invoked in the short interval during boot time by pressing CTRL-ALT-ESC. Once the Ampro SSB386 has completed booting, setup is entered only by executing SETUP.COM, which is found in the Ampro subdirectory.

DATE: (mm/dd/yy)	9/4/90
TIME: (hh:mm:ss)	10:42:29 MST
1ST FLOPPY	1.4M
2ND FLOPPY	NONE
AT HDC DISK 1	TYPE 17*
AT HDC DISK 2	NONE
VIDEO	VGA
BASE MEMORY	640
EXTENDED MEMORY	3456
ERROR HALT	HALT ON ALL ERRORS
CPU SPEED	HIGH

Figure 3: Setup Screen Display (Partial)

* With the 119 meg hard drive installed, Disk 1 would show Type 39. These are IDE-type drives. Refer to the Software Configuration section of the SB386 manual and Appendix C for further information.

Accessing the 152SA1 Board

1. Remove the bottom of the card cage. Turn the cage over and remove the four screws on each side that secure the bottom cover of the cage.

2. Remove the four screws that secure the butch plate to the chassis. This will allow the butch plate to hinge away from the chassis for removal of the computer board.

3. Remove the screw from the metal bracket on the front edge of the computer board. Loosen the Allen screw on the card guide at the rear of the computer board, using a 3/32 Allen wrench.

4. Lift the computer board until its backpolane connector is disconnected from the I/O backplane. The cables attached to the computer board have enough of a service loop to allow the board to be lifted away from the 152SA1 board for access to the IEEE-488 Configuration switch and jumpers.

APPENDIX A

53/63 SERIES SYSTEM COMMANDS

Command	Description
@XY	The @XY (Address) command addresses a function card in the 53/63 Series System. @ is a delimiter used by the 53/63 Series System. X is a card cage address (0-9) defined by the Address Select switch on the 53A-171 Control Card in the addressed card cage. Y is a function-card address (0-9) defined by the Address Select switch on the function card. Once a card cage/function-card combination is addressed, it remains addressed until the 53/63 Series System detects a new @ character.
@XS	The @XS (Status) command provides the interrupt status of all function cards within the card cage defined by X. The interrupt status of all function cards in the addressed card cage is latched into the 53A-171 Control Card when the @XS command is issued. All function cards in all card cages become unaddressed after the @XS command is issued. The 53A-171 Control Card Operating Manual describes the @XS command in detail.
@XH	The @XH (Halt) command halts all function cards within the card cage defined by X. This command does not affect function cards in other card cages. How a function card reacts to the @XH command depends on the particular card. In all cases, an addressed function card (Power LED out) becomes unaddressed (Power LED lit).
STOP	The STOP command is not a string of ASCII characters. This command is hard-wired from the system controller to the 53/63 System's communications card in each card cage. When the system controller issues a STOP command, each function card reacts as if it had received the @XH command described above. How the system controller executes a STOP command depends on the communications card used. For example, when using the 53B-153 Comm Chain Card, a STOP command is executed during power-up or whenever the computer is rebooted.

APPENDIX B

LCD FLAT PANEL DISPLAY

The contents of this appendix are as follows:

Description (46)
Controls And Indicators (47)
Cables (48)
Keypad (48)
Specifications (49)
Connector Wiring (51)

Description

The LCD Flat Panel Display is an independent Liquid Crystal Display (LCD) unit designed to take the place of a VGA monitor in a test system. The LCD Display can be mounted on a 53B-ATZ Card Cage in place of the standard front panel, or it can be separated up to a distance of one foot from the card cage. Extra caution must be taken when separating the display from the cage. The display can be damaged if it tips over or falls.

The 25-line by 80-character CGA text screen (640 x 200 CGA graphics) with built-in electro-luminescent (EL) backlighting permits use of the unit in complete darkness. The display is driven by an Ampro LCD Mini-module which replaces the standard VGA Mini-module. For a complete description of the LCD Mini-module refer to the Ampro LCD Mini-module Technical Manual.

The built-in hex keypad is connected to serial port 2 (COM2) of the Ampro SB-286 Board. The CMOS RAM is changed to accept data from COM2. Refer to Appendix C to set up COM2 for the keypad input. The keypad sends the keypresses to the computer at the baud rate of 9600 bps. The exact syntax is listed in the Specifications section. When an Option 31 LCD Display is ordered, the factory sets the CMOS RAM for use with the hex keypad. For a full description of CMOS RAM setup, refer to Appendix C.

The hex keypad permits the operator to interact with a test program. (This keypad is not intended for use as a program-entry device; such use would soon become very tedious. Use the keyboard for data entry.) To enter a character shown in red, just press the desired character. To enter a blue, yellow, or green character, first press the same color Shift key, and then the desired character. The Shift LED comes on as soon as a shift key is pressed, indicating that the operator is now in a shift sequence. If the operator presses another Shift key while the Shift LED is lit, the Shift LED remains lit, but the keypad switches to the new shift color. The keypad does not have an auto-repeat capability, nor does it recognize when two keys are pressed simultaneously (n-key rollover). The membrane-type keypad is suitable for use in harsh environments.

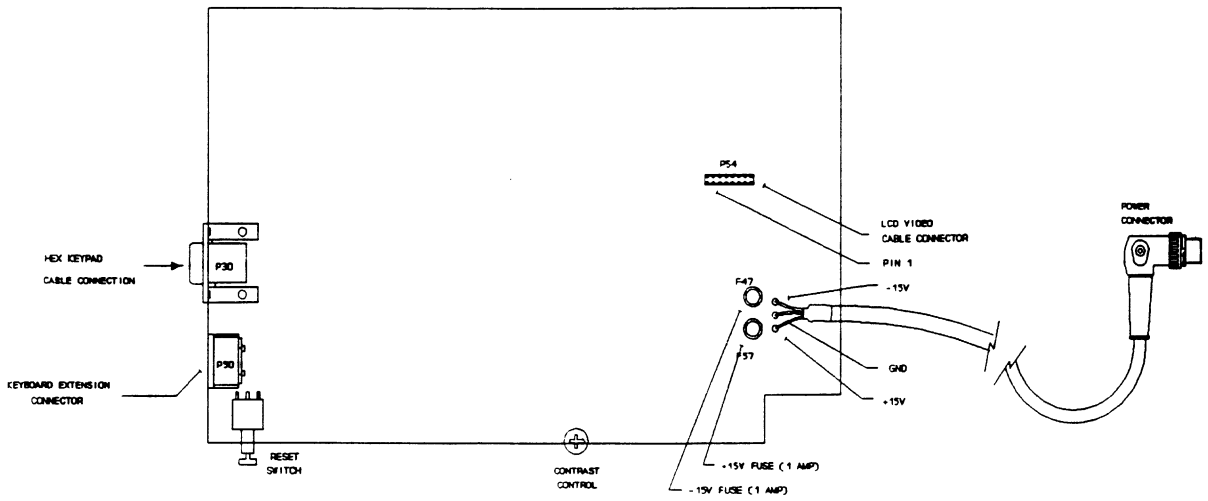


Figure 4: Controls and Indicators

Controls And Indicators

The following controls and indicators are provided to select and display the functions of the LCD Flat Panel Display's operating environment. See Figure 4 for their physical locations.

Reset Switch

This momentary switch resets the Display to its power-up default condition. The Remote and Power LEDs will be on, and the Shift LED will be off.

Fuses

The Display has two 1-amp fuses for the two input voltages of +15 V dc and -15 V dc. These fuses protect the Display as well as the card cage. If a fuse blows, correct the fault before replacing it. If a fuse blows when the Display is first being installed on the card cage, first check the power connections in the card cage.

Contrast Control

The Display has a contrast control on the lower edge of its printed circuit board. This control is a small gray wheel that extends over the edge of the bottom of the board. Turn this control to adjust the darkness of the characters for the standard display and the darkness of the background for reverse video. This control should be adjusted to suit the operator's personal preference or ambient lighting conditions.

Cables

The following cables are supplied with the LCD Display:

Power Cable

The Display's power cable plugs into a modified switch plate at the front of the card cage. This modified switch plate is supplied with the 53B-ATZ-31. The power cable is three feet long, and permits use of the Display detached from the card cage.

RS-232C Cable

The RS-232C cable is for the hex keypad communications. The hex keypad is connected to COM2, and the Ampro SB-286 CMOS RAM is configured to take input from the COM2 port.

Keyboard Extension Cable

The Keyboard Extension Cable connects the 53B-ATZ keyboard connector, located on the butch plate, to the LCD Display's keyboard connector.

LCD Display Cable

The LCD Display Cable is a 16-conductor cable that connects the LCD Mini-module mounted on the SB-286 Board to the LCD Flat panel Display.

Keypad

The LCD Display has a built-in, membrane-type, 16-key keypad. This color-coded keypad can generate 49 ASCII characters, using a shifting scheme. Pressing a particular shift key activates the characters that are the same color as that shift key.

The keypad also has three LEDs:

Power and Remote LEDs

These LEDs come on when the Display's internal microprocessor has been initialized properly.

Shift LED

The Shift LED comes on whenever one of the three Shift keys on the built-in keypad is pressed. When lit, the Shift LED tells the operator that the keypad is in a shift sequence. If the operator presses another shift key instead of a character key, the Shift LED remains on, and the shift color changes to that of the new Shift key.

Specifications

Function: Flat panel LCD display for 53B-ATZ Option 31.

Screen Type: Liquid Crystal Display (LCD), Sharp Corp. (Part # LM64014M).

Viewing Angle: 25° up.
35° down.
35° side to side.

Backlighting: Electroluminescent (EL), excited by internal 115 V ac, 750 Hz at 50 mA power supply.

Screen Size: CGA Graphics: 640 x 200 pixels.
CGA Text: 80 characters x 25 lines.

LCD Controller: Ampro Mini-module/LCD (VADEM V6-600).

Keypad

Type: Sealed membrane.

Characters: 0 through 9.
Uppercase A through Z.
Special characters:
/ slash
, comma
. period
" quotation mark
? question mark
\ back-slash
: colon
! exclamation point
; semicolon
- hyphen

NOTE: All characters are ASCII-encoded.

Special Keys: SHIFT 1, SHIFT 2, SHIFT 3, SP (space), BK SP (backspace), ENTER (carriage return).

Scan Rate: 5 ms.

Auto-repeat: None.

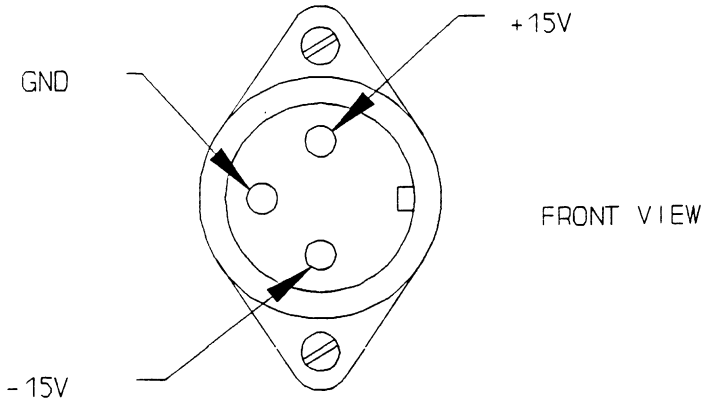
n-Key Rollover: None.

LEDs: Power, Remote, and Shift.

Keypad Controller:	6-MHz, CMOS Z80 running at 5.58 MHz.
Program Memory:	8K-byte, 150-ns EPROM.
Power Requirements	
Source:	Connects to 53B Card Cage power supplies.
Current:	500 mA of +15V dc. 200 mA of -15V dc.
Cooling:	Provided by the fan in the 53B Card Cage.
Temperature:	-10°C to +65°C, operating (assumes ambient temperature of 55° and airflow to assure less than 10°C temperature rise). -40°C to +85°C, storage.
Humidity:	Less than 95% R.H. non-condensing, -10°C to +30°C. Less than 75% R.H. non-condensing, +31°C to +40°C. Less than 45% R.H. non-condensing, +41°C to +55°C.
Dimensions:	261 mm high, 64 mm deep, 426 mm wide. (10.25 in x 2.5 in x 16.75 in).
Dimensions, Shipping:	152.4 mm high, 508 mm deep, 355.6 mm wide. (6 in x 20 in x 14 in).
Weight:	2.0 kg including switch plate. (4.4 lb).
Weight, Shipping:	2.3 kg. (5.0 lb).
Mounting Location:	Replaces 53B-003 Card Cage's Plexiglas front panel.
Input/Output Connections	
Power Connection:	3-pin DIN plug.
LCD Display Port:	16-pin 0.1" header (P54).
Keyboard Port:	5-pin DIN plug (P50).
Hex Keypad Port:	Male DB-9 (P30).

Connector Wiring

Switch Plate Power Connector Wiring



Wire Color Code

Red = +15 V dc
Black = Ground
Green = -15 V dc

LCD Display Port

16-pin 0.1" header (P54), wired as follows:

Pin 1	FRP	Pin 9	UD0
Pin 2	LIP	Pin 10	UD1
Pin 3	CLP	Pin 11	UD2
Pin 4	FRMB	Pin 12	UD3
Pin 5	N.C.	Pin 13	LD0
Pin 6	+5 V	Pin 14	LD1
Pin 7	GND	Pin 15	LD2
Pin 8	VLDC	Pin 16	LD3

Keyboard Port

5-pin DIN plug (P50), wired as follows:

Pin 1	Keyboard CLK
Pin 2	Keyboard DATA
Pin 3	GND
Pin 4	Reset
Pin 5	Keyboard Power

Hex Keypad Port

Serial, RS-232C compatible, transmit and receive.

Characteristics:	Data bits: 8. Stop bits: 1. Parity: None. Baud rate: 9600.
Signals:	TXD, RXD, CTS, DTR, RTS, and GND.
Connector Type:	DB-9 socket.

Connector Wiring Hex Keypad Port)

Pin 1	N.C.
Pin 2	RXD
Pin 3	TXD
Pin 4	DTR
Pin 5	GND
Pin 6	N.C.
Pin 7	RTS
Pin 8	CTS
Pin 9	N.C.

APPENDIX C

CMOS MEMORY SETUP

The setup program may be invoked in the short interval during boot time by pressing CTRL-ALT-ESC. Once the Ampro SSB386 has completed booting, setup is entered only by executing SETUP.COM, which is found in the Ampro subdirectory.

The following screens are displayed when the setup program is invoked. Use the arrow keys to select items which are to be changed. The asterisks shown here do not appear on the actual screens, and are used here to refer to the explanatory notes. These screens may not be identical for all Ampro boards. These are shown as guidelines only.

```
Set up:     Date - current date          Time:                    Current Date:

        1st floppy - 1.4 Meg (use side arrow keys)
        2nd floppy - None
                                  Cyls.  Heads  Sectors  Precomp.  Land. Zone
        * AT HDC Disk 1 - Type 17  977     5      17       300       977
        AT HDC Disk 2 - None

        Video - EGA/VGA

        System RAM Map - 4 Meg Memory (use side arrows to select base/ext.)
        Base Memory - 640
        Extended Memory - 3456

        Error Halt - Halt on all errors
        Shadow RAM - Disabled

        CPU Speed Select - NORMAL

        (Press PgDn key for next Setup Menu)
```

* Use the side arrow keys to change this if required.
Type 17 - 40 MByte IDE
Type 39 - 120 MByte IDE drive (optional)

Extended Option /Peripheral Configurations

Mono/Color Jumper	Color
Serial Port 1 (J3)	Enabled
Serial Port 2 (J2)	Enabled
Parallel Port	Primary
U32 Byte Wide Socket	Disabled
U33 Byte Wide Socket	Disabled
U22, 33 Socket Size	64K
System DMA Speed	Normal
CPU Speed During Floppy I/O	Full (Normal)
SB/386 Extended BIOS	Enabled
Hot Key Setup	Enabled
SCSI Interface Hardware	Enabled

(Press PgDn key for next Setup Menu)

Extended SCSI and Hard Disk Configuration

SCSI/BIOS Services	Disabled
SCSI Initiator ID	7
SCSI Disk I/O Retries	10

SCSI Disk Map	Physical Device
SCSI Disk 1	Not Active
SCSI Disk 2	Not Active
SCSI Disk 3	Not Active
SCSI Disk 4	Not Active
SCSI Disk 5	Not Active
SCSI Disk 6	Not Active
SCSI Disk 7	Not Active

DOS Disk Map	Physical Device
Default Boot Device	Hard Drive
1st Hard Disk	At Bus HDC, Disk 1
2nd Hard Disk	Not Active
3rd Hard Disk	Not Active
4th Hard Disk	Not Active
5th Hard Disk	Not Active
6th Hard Disk	Not Active
7th Hard Disk	Not Active
8th Hard Disk	Not Active

(Press PgDn key for next Setup Menu)

Extended Serial Console configuration

```
Console Output Device ..... Video
Console Input Device ..... Keyboard *

Serial Console Output Setup
    Data Length .....
    Stop Bits .....
    Parity .....
    Baud .....
    Delete from Com Port Table .....
Console Output Handshake .....

Serial Console Input Setup **
    Data Length ..... 8 bits
    Stop Bits ..... 1
    Parity ..... None
    Baud ..... 9600
    Delete from Com Port Table ..... YES
```

* Selects console input:

Keyboard	to get input from the keyboard only.
Serial 2	to get input from the keyboard and the hex keypad (Option 31).

** Selects baud rate of console input. The selections shown are for Option 31, and will appear when 'Serial 2' is selected.