The contents of this manual and the specifications herein may change without notice.

TRADEMARKS

CAMBASIC II™and PC SmartLINK™are trademarks of Octagon Systems Corporation.

Microsoft® BASIC is a trademark of Microsoft Corporation.

Remote Processing Corporation
7975 E. Harvard Ave.
Denver, Co 80231
Ph: (303) 690 1588
Fax: (303) 690 1875
www.rp3.com

NOTICE TO USER

The information contained in this manual is believed to be correct. However, Remote Processing assumes no responsibility for any of the circuits described herein, conveys no license under any patent or other right, and make no representations that the circuits are free from patent infringement. Remote Processing makes no representation or warranty that such applications will be suitable for the use specified without further testing or modification. The user must make the final determination as to fitness for a particular use.

Remote Processing Corporation's general policy does not recommend the use of its products in life support or applications where the failure or malfunction of a board may threaten life or injury. Install redundant or backup safety systems as appropriate to the application.

FCC AND EMI NOTICE

The SBS-2300 is intended as an OEM product in an industrial environment. It was not tested for EMI radiation. When operated outside a suitable enclosure, the board and any cables coming from the board will radiate harmful signals which interfere with consuer and industrial radio frequencies. It is your responsibility to properly shield the RPC-2300 and cables coming from it to prevent such interference.

P/N 1196
Revision: 1.2

TABLE OF CONTENTS

CHAPTER 1 OVERVIEW

DESCRIPTION

The SBS-2300 is an embedded controller with a built in Basic language. Several features make it suitable as a stand alone unit:

- Built in CAMBASIC II programming language autoruns at power up. On card flash EPROM programmer saves programs to 30K.
- Eight single ended or 4 differential analog inputs convert voltage inputs to a digital value using a 12 - bit (4096 count) A/D converter. Two analog outputs are available.
- Keypad port for operator interface. The 16 position keypad is automatically scanned and is read using the KEYPAD command.
- Two RS-232 serial ports are programmable for baud rate, parity, length, and stop bits. Both inputs and outputs have a 256 byte buffer.
- Watchdog timer resets the card if the program "crashes". The timer is enabled and disabled by software.
- 48 general purpose digital I/O lines, 8 of which are high current outputs. These lines can connect to another opto rack.
- 128K of RAM is standard, with a 512K RAM optionally available. RAM is optionally battery backed using a DS-1213D.
- Built in EPROM programmer save programs for autorun on power up or reset.

The SBS-2300 uses a 64180 CPU operating at 9 Mhz. It operates stand alone or on a network using RS-485 adapter. Its 4.5" x 8" size makes it easy to mount in a NEMA box.

CAMBASIC II programming language is standard. This language was adapted for the SBS-2300 for control and data acquisition applications. A complete description of CAMBASIC II commands is in the *CAMBASIC II Programming Guide.*

Program development can take place on your PC, using your word processor, or on the SBS-2300. Programs from your PC can be downloaded using PC SmartLINK or other serial communication program.

MANUAL ORGANIZATION

This manual provides all the information required to install, configure, and use the features on the SBS-2300.

This manual assumes you are familiar with some type of BASIC programming software. The syntax used by CAMBASIC II is similar to Microsoft's GW or QuickBASIC. If you are not experienced with BASIC software, you may want to refer to books and training programs available through your local software store. The *CAMBASIC II Programming Manual* has information and examples for all commands.

NOTE: The SBS-2300 uses a Hitachi Z180 processor. Additional information can be obtained from Hitachi or a local representative. Order hardware manual #U77, software manual #U92.

MANUAL CONVENTIONS

Information appearing on your screen is shown in a different type.

Example:

```
CAMBASIC II (tm) V1.00
(c) 1985-94 Octagon Systems Corporation
(c) 1994  Remote Processing Corporation
All rights reserved - free 29434
```

Symbols and Terminology

NOTE: Text under this heading is helpful information. It is intended to act as a reminder of some interaction with another part of the manual or device that may not be obvious.

WARNING:

Information under this heading warns you of situations which might cause catastrophic or irreversible damage.

W[-] Denotes jumper block pins.

< xxx> Paired angle brackets are used to indicate a specific key on your keyboard. For example < esc> means the escape key.

BASIC uses decimal convention for designating addresses and data. There are times, however, when hexadecimal notation is more convenient to use. The hexadecimal notation used in this manual and by CAMBASIC II is the ampersand character (&) before the number. A &8C stands for 8C hexadecimal.

TECHNICAL SUPPORT

If you have a question about the SBS-2300 or CAMBASIC II used on it and can't find it in this manual, call us and ask for technical support.

When you call, please have your SBS-2300 and CAMBASIC II manuals ready. Sometimes it is helpful to know what the SBS-2300 is used for, so please be ready to describe its application as well as the problem.

Phone: 303-690-1588
FAX: 303-690-1875

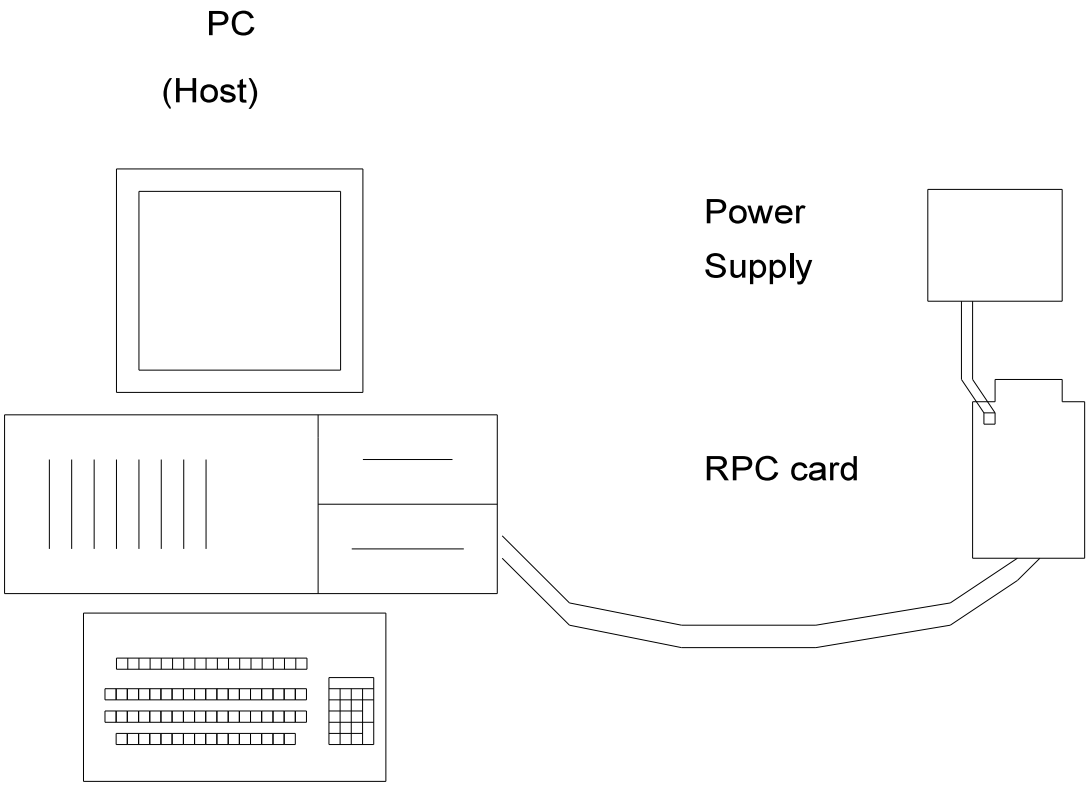


Figure 1-1 System layout

CHAPTER 2 SETUP AND OPERATION

INTRODUCTION

The SBS-2300 is ready to program as soon as you connect it to a terminal or PC and apply power. This chapter describes what is needed to get a sign- on message and begin programming.

Requirements for uploading and downloading programs is discussed. A "Where to go from here" section directs you to the chapters to read in order to use the various capabilities of the SBS-2300. Finally, a troubleshooting section helps out on the most common problems.

OPERATING PRECAUTIONS

The SBS-2300 is designed to handle a wide variety of temperature ranges and operating conditions. These characteristics require using CMOS components. CMOS is static sensitive. To avoid damaging these components, observe the following precautions before handling the SBS-2300.

1. Ground yourself before handling the SBS-2300 or plugging in cables. Static electricity can easily arc through cables and to the card. Simply touching a metal part on your PC can greatly reduce the amount of static.

2. Do not insert or remove components when power is applied. While the card is a + 5 volt only system, other voltages are generated on the card. Applying them in the wrong sequence can destroy a component.

EQUIPMENT

You will need the following equipment to begin using the SBS-2300:

SBS-2300 embedded controller
PC with a serial port and communications program (such as PC SmartLINK)

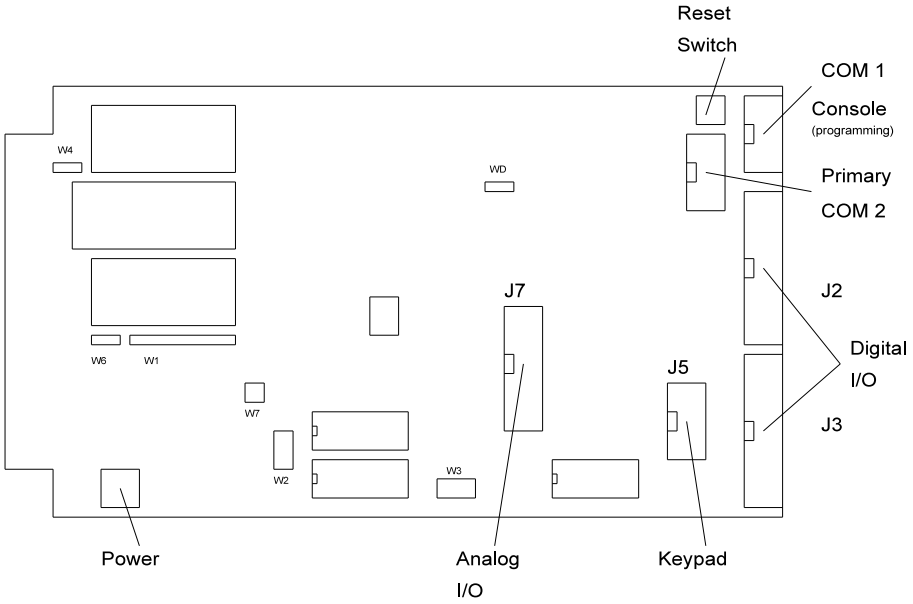


Figure 2-1 Connector Locations

(Equipment continued)
VTC-10 serial cable
+ 5, 300 ma power supply
The *CAMBASIC II Programming Manual* is strongly recommended. Refer to *Chapter 4 Serial Ports* for wiring information to make your own cable.

FIRST TIME OPERATION

Become familiar with the locations of the connectors before getting started. See Figure 2-1.

SBS-2300 jumpers have been set at the factory to operate the system immediately. For first time operation, do not install any connectors or parts unless specified below. Jumpers should be kept in default positions.

1. The SBS-2300 needs + 5 ±0.25 volts at less than 300 ma. Any well regulated supply that supplies this will work. Be careful when using "switching" power supplies. Some of these supplies do not regulate properly unless they are adequately loaded.

   Make sure power is off. Connect the power supply to the appropriately marked terminals on the SBS-2300.

2. You can use either a PC or CRT terminal to program the SBS-2300. Connect one end of the VTC-10 connector to then 10 pin COM1 (console) port on the SBS-2300. Refer to Figure 2-1 for connector location.

Using a PC

Connect the VTC-10 serial cable to the PC's COM1 or COM2 port. You may need a 9 pin male to 25 pin female adapter. The VTC-10 is designed to plug directly into the 25 pin serial port connector on a PC.

Start up your serial communication program (PC SmartLINK or other). Set communication parameters to 19.2K baud, 8 data bits, no parity, 1 stop.

Using a Terminal

Follow your terminal instructions to set the baud rate to 19.2K baud, 8 data bits, no parity, and 1 stop. You may need a 9 pin male to 25 pin female adapter to connect the VTC-10.

PC or Terminal

The SBS-2300 does not send a CTS signal to the PC or terminal. If your terminal or communications software requires this or other signals (DCD, DSR), you may have to tie them to the appropriate levels. You may be able to ignore these lines in software.

3. Turn on your power supply. On power up a copyright message is printed.

```
CAMBASIC II (tm) V1.00
(c) 1985-94 Octagon Systems Corporation
(c) 1994  Remote Processing Corporation
All rights reserved - free 29434
```

   If a nonsense message appears, your terminal or PC may not be set to the appropriate communication parameters. If the system still does not respond, refer to "TROUBLESHOOTING" later in this chapter.

4. The system is now in the "immediate mode" and is ready for you to start programming. Type the following program (in upper or lower case:

```
10 FOR X = 0 TO 2
20 PRINT "   Hello ";
30 NEXT
40 PRINT
```

   Now type `RUN`

   The system will display:

```
Hello Hello Hello
```

UPLOADING AND DOWNLOADING PROGRAMS

Downloading programs means transferring them from your PC (or terminal) to RAM on the SBS-2300. Uploading means transferring programs from RAM back to the PC. This section explains how to do both of these procedures using PC SmartLink. Generalized instructions for other terminal programs are given at the end of this section.

Uploading programs

In the previous section, you wrote a test program. To upload that program to a PC and save it to disk:

1. Press the < F1> key. A window with the main menu will appear.
2. Press the letter U (upper or lower case). Your program will begin to transfer from RAM to the

PC. When menu appears.

3. To save a program to disk, type the letter S. You are prompted for a file name. Enter the file name you want the program saved under.

4. Press < F2> to return to the immediate mode.

NOTE: Some versions of PC SmartLINK have pull down menus or will operate differently. Refer to the SmartLINK manual for the version you are using.

Downloading programs

To practice downloading a program, type

```
NEW< return>
```

Perform the following when using PC SmartLINK:

1. Press the < F1> key to view the main menu.

2. SmartLINK has a buffer which is used to temporarily store the program. If you followed these instructions without exiting SmartLINK, the previously uploaded program is in the buffer and may be downloaded. However, lets assume you just started SmartLINK. Press the L key to get the program from the disk.

3. Enter the filename to get the file.

4. Press D to download the program.

5. Press the < F2> key to return to the program. You can list the program by typing:

```
list
```

or

```
/
```

Other communications software

The following is general information when using another terminal emulation program (Procomm, Windows Terminal, etc.).

When uploading or downloading files, select ASCII text format. Other formats are not used.

CAMBASIC II does not know when you are typing in a program or if something else (laptop or mainframe) is sending it characters. The upload and download file does not contain any special control codes, it is simply ASCII characters.

Uploading programs is simply a process of receiving an ASCII file. You or your program simply needs to send "LIST" to receive the entire program. The default baud rate (19200) is rather high. Make sure your PC and communications software can work at these baud rates. PROCOMM was tested on a 12 Mhz 286 PC and it worked fine. Windows Terminal on the same PC had problems at much slower baud rates.

Downloading a program requires transmitting an ASCII file. CAMBASIC II is an incremental line compiler. As you type in (or download) a line, CAMBASIC II compiles that line. The time to compile a line depends upon its complexity and how many line of code have been entered.

CAMBASIC II must finish compiling a line before starting the next one. When a line is compiled, a "> " character is sent by the card. This should be your terminal programs pacing character when downloading a program.

If your communications program cannot look for a pacing prompt, set it to delay transmission after each line is sent. A 100 ms delay is usually adequate, but your CAMBASIC II program may be long and complex and require more time. A result of a short delay time is missing or garbled program lines.

CAMBASIC II sends out escape sequences to clear the screen. This sequence may appear as < -; on your screen. Usually this will not be a problem.

COM1 on the SBS-2300 does not recognize the CTS or RTS lines. The CTS line is pulled high on the SBS-2300. The effect of not recognizing these lines is your PC or terminal cannot hold off the SBS-2300's transmission. Converse, the SBS-2300 cannot hold off the host from sending it data.

Editing programs and programming hints

Files uploaded or downloaded are simply ASCII DOS text files. No special characters or control codes are used. You may create and edit programs using your favorite word processor or editor. Just be sure to save files in DOS text format.

A technique used to further program documentation and reduce code space is the use of comments in a downloaded file. For example, you could have the following in a file written on your editor:

```
'Check VAT temperature

'Read the output from the RTD and
' calculate the temperature

2200 a = ain(0) :'Get temp
```

The first 3 comments downloaded to the SBS-2300 would be ignored. Similarly, the empty lines between comments are also ignored. Line 2200, with its comment, is a part of the program and could be listed. The major penalty by writing a program this way is increased download time.

NOTE: Some versions of PC SmartLINK may optionally strip comments before downloading. Check your manual to see if this option is available.

Notice that you can write a program in lower case characters. CAMBASIC II translates them to upper case.

Some programmers put "NEW" as the first line in the file. During debugging, it is common to insert "temporary" lines. Adding NEW ensures that these lines are gone. Downloading time is increased when the old program is still present.

If you like to write programs in separate modules, you can download them separately. Modules are assigned blocks of line numbers. Start up code might be from 1 to 999. Interrupt handling (keypad, serial ports) might be from lines 1000 to 1499. Display output might be from 1500 to 2500. The programmer must determine the number of lines required for each section.

When replacing a program or section, downloading time is increased. Blocks of line numbers cannot be renumbered by CAMBASIC II when other parts of the program are installed. However, if a particular section is the only program downloaded, then line renumbering in that range is possible. Refer to the CAMBASIC II RENUM command.

CAMBASIC II automatically formats a line for minimum code space and increased readability. For example, you could download the following line of code:

```
10 fora=0to5
```

When you listed this line, it would appear as:

```
10 FOR A = 0 TO 5
```

Spaces are initially displayed but not stored. The following line:

```
10 for a   =    0    to    5
```

would be compressed and displayed as in the second example above. Spaces are removed.

The *CAMBASIC II Programming Manual* has more information about increasing program speed and editing options.

Instead of uploading and downloading programs, you can save them to the on card flash EPROM. This is useful if you are using a terminal to write programs. Make sure the 'AUTORUN' jumper is installed (See *Chapter 3 SAVING PROGRAMS*). To prevent automatic program execution on power up, insert the STOP statement at the beginning of the program (such as line 1). When you power up the SBS-2300, the program is transferred into RAM and executed. Delete the program line with the STOP statement to normally start programs. When saving programs, be sure to reenter the STOP statement with its line number.

WHERE TO GO FROM HERE

If you want to do this:	Turn to Chapter
Save a program	3
Autorun a program	3
Know more about serial ports	4
Install a different RAM memory chip	5
Battery backing up RAM	5
Using RAM to save variables	5
Configure digital I/O lines	6
Get switch status	6
Use high current outputs	6
Connect an external opto rack	6
Installing calendar/clock module	7
Analog I/O	8
Keypad port	9
Watchdog timer	11

Also, refer to the table of contents for a listing of major functions.

TROUBLESHOOTING

You probably turned to this section because you could not get the sign on message. If you are getting a sign on message but can't enter characters, then read section 5. The following are troubleshooting hints:

1. Check the power source. If it is below 4.65 volts, the SBS-2300 will be reset. Power is 5 ±0.25 volts. Make sure it is a clean 5 volt source. If it dips intermittently to 4.65 volts (due to switching noise or ripple), the card will reset for about 100 ms. If the noise is frequent enough, the card will be in permanent reset. Check U14, pin 6. If it is low (about 0 volts), then it is in reset. This line should be high (about + 5 volts).

2. Check the COM1 port. COM1 is also known as console port J1. Remove the connector from COM1. Refer to the outline drawing earlier in this chapter. Connect an oscilloscope (preferred) or a voltmeter to pin 3 (Txd) and ground. Pin 3 should be -6 volts or more negative. (Pin 1 is designated by the ^ symbol on the connector. Pin 3 is next to it, nearer the key opening.) If you have -6 volts or more, press the reset switch. If you have a scope attached, you should see a burst of activity. If you have a volt meter, you should see a change in voltage. Using a Fluke 8060A set to measure AC, you should see a momentary reading above 2 volts. Press reset several times to make sure it captures it.

3. Install the cable and make sure the voltages and output activity are still there. Output is from pin 3 on the VTC-10. Check to make sure something is not shorting the output.

4. Check the serial parameters on your PC or terminal. They should be set to:

   19200 baud, no parity, 8 data bits, 1 stop

If all of this fails, call technical support listed at the front of the book.

INTRODUCTION

Programs are stored in socket U3. An optional real time clock module, a DS-1216EM may also be installed in U3. See *Chapter 7* for calendar/clock installation and operation.

You can store one program up to a maximum size of about 28K bytes. A general rule to determine program storage requirements is one line requires 40 bytes. 28K bytes would store over 700 lines of code. Your application could be significantly more or less, depending upon the number of commands / line, comments, and print statements. Another indication of program size is to use the file length as saved on a PC disk.

Despite the fact you may have 128K or 512K RAM installed, the maximum program size CAMBASIC II can run is about 28K (leaving room for variable storage). Only one program can be stored on the EPROM. Programs cannot be chained.

A flash EPROM is non-volatile (retaining data even when power is disconnected), having an unlimited number of read cycles and a limited number of write cycles (about 1,000). A program is not run from EPROM. It is transferred to RAM and run from there. Programs in RAM are run and can be modified. They can be saved to EPROM for auto execution later.

The SBS-2300 can be set to autorun on power up or reset by installing a jumper (W1[9-10]). When autorun is on, the program in EPROM is loaded into RAM and begins to execute immediately.

The EPROM is write-protected with a software lock, so accidental writes on power-on or -off are almost impossible. You cannot disable the lock except when executing the SAVE command. To save parameters, you must use battery backed RAM.

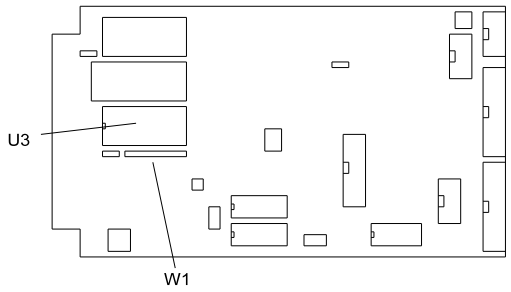


Figure 3-1 Autorun jumper

SAVING A PROGRAM

To save a program, set jumper W1[9-10]. You may set the jumper even if the power is on. Remember to discharge any static electricity before installing or removing the jumper. For this example, assume you wanted to save the following program:

```
10 FOR N = 0 TO 2
20 PRINT "Hello ";
30 NEXT
40 PRINT
```

If this program is not already in, type it in now (or, if you prefer, use your own program).

Type in the following command:

```
SAVE
```

CAMBASIC II will compile the program, program the EPROM, and verify its contents.

```
Compile...Write...Verify
```

The time it takes to do all of this depends upon the length and complexity of the program. Generally, it will be from 1 to 20 seconds. The ready prompt (>) will appear when the program has been successfully saved to the EPROM. If the program does not write to the EPROM, an error message will appear:

```
Fail @ xxxx
```

Saving a program overwrites the previous one. There is no way to recover it since both occupy the same space.

AUTORUNNING

To autorun a program:

1. Make sure there is a program in EPROM (from above).

2. Make sure the autorun jumper W1 [9-10] is installed.

If you push the reset button, the program should autoexecute. If there are any errors, the program will stop (assuming you have not trapped them with ON ERROR) and display the error message.

PREVENTING AUTORUN

When troubleshooting a program, it's not always convenient for an autoexecute file to run. This is especially true if the program has been configured to ignore the < ESC> key. To prevent autorun, remove jumper W1[9-10].

Later, if you wish to SAVE or LOAD a program, reinstall this jumper. You may do so even if the power is on and a program is running. Remember to discharge any static electricity before installing or removing the jumper.

LOADING AN EPROM PROGRAM

There are times when you may wish to temporarily modify or otherwise test out a change to a program. Since the program is loaded into RAM, modifications can be made without affecting the program in EPROM. If you find out that modifications are not desirable or did not work, you can restore the original program to RAM using the LOAD command.

SAVING DATA TO EPROM

Additional data, such as strings and constants, can be saved to the EPROM in U3. An external EPROM programmer is used to save data to the EPROM. Data cannot be saved to U3 through CAMBASIC II.

Data is saved at the top of the EPROM memory. The upper 2K bytes are always available as RAM memory size absolutely limits the program to 30K.

If you need more than 2K bytes of data, you can save data "on top" of the CAMBASIC II program in U3. A good way to determine how much memory is available is to perform a PRINT SYS(0) in the immediate mode (program not running). Subtracting 31900 from the number of bytes returned will tell you the approximate number of bytes available for data.

The best way to make sure your data will not write over the program is to perform the following steps:

First, put a remark statement that you can recognize. One suggestion is "End of the program". Next, save your program to the EPROM using the CAMBASIC II SAVE command.

Remove the EPROM and read it from your EPROM programmer. Using your programmer, go into the mode where you can examine data. Look for the remark statement at the end of the program. Instruct the programmer to put your data starting at the next even page boundary (for example 5000H, 5100H, and so on).

COMMANDS

The following is a list of CAMBASIC II commands used for saving and loading programs.

Command	Function
LOAD	Transferrs program from U3 to RAM for editing or running.
SAVE	Saves a program from RAM to U3 for autorun.

DESCRIPTION

The SBS-2300 has two serial ports that can be used for interfacing to a printer, terminal, or other serial devices. This chapter describes their characteristics and how to use them. Frequent references are made to commands listed in the *CAMBASIC II Programming Manual*. Please refer to this manual for more information.

Serial ports are numbered COM1 and COM2. COM1 is used for program development. During run time, it can be used for other functions. COM2 is a general purpose RS-232 port.

Both ports support XON/XOFF protocol to control data transmission. Each port has a 256 character interrupt driven input and output buffer. This allows characters to be sent out (using PRINT) without slowing down program execution. However, if the PRINT buffer fills, program execution is suspended until the buffer empties. Both ports have a 256 character input buffer. When more than 256 characters have been received, excess ones are ignored.

The baud rate, parity, data length, and stop bit length are changed using the CONFIG BAUD command.

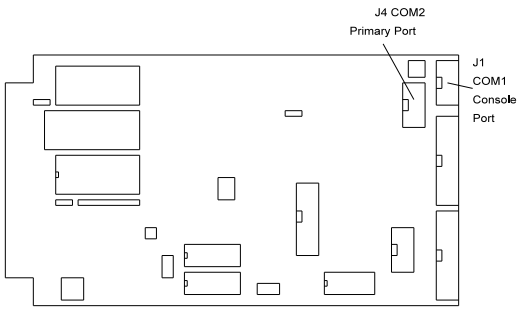


Figure 4-1 Serial ports

COM1 SERIAL PORT

COM1 is J1 and is called the Console port on the card. This port uses a VTC-10 serial cable to connect external serial devices to the port. The cable consists of a 10 pin IDC connector wired to a DB-9 connector. The connector plugs directly into a 9 pin serial port connector on a PC.

This port is normally used for programming. During run time it may be used as a general purpose serial port. When used for programming or with the INPUT statement, it will accept ASCII character values from 0 to 127. When used with the INKEY$ and COM$ functions, it will return ASCII values from 0 to 255.

COM2 SERIAL PORT

COM2 is an RS-232 port. It also uses a VTC-10 serial cable, described above. COM2 is identical to COM1 except that COM2 has 2 hardware handshaking lines, CTS and RTS. When RTS goes low, the SBS-2300 is held off from transmitting out COM2. The status of this port is read by the BIT statement. The example below returns the status of the RTS line:

```
100 B = BIT(130,5)
```

If B = 1, transmission is held off.

The CTS line may be set high or low to hold off communication. Line 400 sets CTS low and 500 sets it high.

```
400 BIT 128,4,1
500 BIT 128,4,0
```

The CONFIG BAUD statement sets both serial ports for baud rate and data type.

ACCESSING SERIAL BUFFERS

You can access COM1 and COM2 buffers in three ways:

1. INPUT statement. This removes all characters in the buffer up to the terminator character and puts them into a variable.

   When using the INPUT statement, program execution is suspended until a < cr> (Enter key) is received. Whether this is a problem depends on your particular application.

   INPUT strips bit 7 on the COM1 port. This means ASCII characters from 0 to 127 are received. The INPUT statement can return a maximum string length of about 150 characters.

2. INKEY$(n) function. Characters are removed one at a time. A null string is returned when the buffer is empty.

In this mode, you have access to the full 256 bytes. If you don't read the buffer and the buffer fills, all subsequent characters are discarded. INKEY$(n) may be used anywhere in the program.

3. COM$(n) retrieves all characters in the buffer, including < cr> 's and other control codes. This function is commonly used with ON COM$ multitasking statement. You can retrieve 128 of the 256 bytes in the serial buffer at one time.

SERIAL PORT FILE NUMBERS

CAMBASIC II references the serial I/O ports by file numbers, similar to DOS. The following table shows the corresponding file number to serial I/O port and how they are used with the various ports.

Description	File	Examples
COM1	1	PRINT "Hello" PRINT #1,"Hello" INPUT A$ INPUT #1,A$ A$ = INKEY$(1)
COM2	2	PRINT #2,"Hello" INPUT #2,A$ A$ = INKEY$(2)

COM1 is J1, the console port. COM2 is J4, the primary port.

COMMANDS

The following is a list of CAMBASIC II commands used for serial I/O. Variations for many commands not listed here. These commands and functions are explained in the *CAMBASIC II Programming Manual.*

Command	Function
CLEAR COM$	Clears serial input buffer
COM$	Returns string from buffer
CONFIG BAUD	Sets serial port parameters
CONFIG COM$	Configures port for ON COM$(n) interrupt
INKEY$	Returns a character from the serial buffer
INPUT	Receives string from port
LIST	Outputs program listing
ON COM$	Calls subroutine on serial input
PRINT	Outputs data in various formats
TAB	Tabs to predetermined positions

SERIAL CABLE PIN OUT

The following is the pin out between the IDC connector for the SBS-2300 and the DB-9 connector to the PC or terminal.

IDC	DB-9	Description
1	4	DCD
2	3	RXD
3	2	TXD
4	1	DTR
5	5	Ground
6	n/c	DSR
7	8	CTS out
8	7	RTS in
9	n/c	+ 5 V
10	n/c	RI

Not all pins/functions on the VTC-10 are used by the SBS-2300. See *Technical Information* for specific J1 and J4 pin outs.

INTRODUCTION

The SBS-2300 is available with 128K of RAM. A 512K RAM may be installed at any time. RAM is in socket U2. RAM may be battery backed by installing a DS-1213D in socket U2. RAM is installed on top of U2.

Battery life will depend upon RAM size, its power consumption, and amount of time the board is operating. Generally, a battery life from 5 to 10 years can be expected.

This chapter discusses changing RAM, installing a battery backup for RAM, saving and retrieving variables, and running assembly language programs. Figure 5-1 shows the location of U2 and jumpers W4 and W7 (for RAM size change).

Increasing RAM size does not increase the program size CAMBASIC II can handle. Maximum program and variable size is 30K. Additional RAM does increase the amount of variable and string storage available using the PEEK and POKE commands.

Due to the memory mapping scheme, the additional amount of memory available when a 128K or 512K RAM is installed is 32K less than the memory size. Thus, a 128K RAM provides 96K of program and data memory and a 512K provides 480K.

If program and data are battery backed, the UNNEW command may be used to restore the program. Variables used by the Basic program are cleared, however. Certain variables are preserved and data POKEd into RAM is saved.

There is an interaction between the speaker and when 512K of RAM is installed. This is discussed below under CHANGING MEMORY.

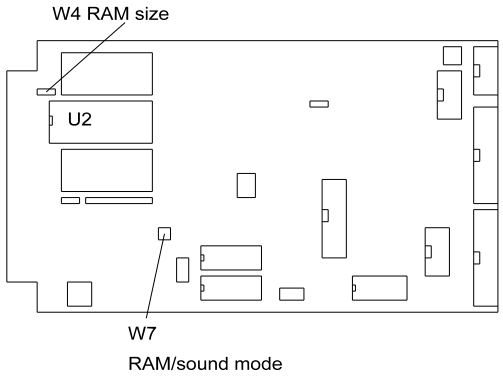


Figure 5-1 Jumpers W4 and W7

CHANGING MEMORY

Different types of memory can be installed at any time. SBS-2300 models come with 128K of RAM installed. Up to 512K can be installed.

To change a memory chip, you need to remove the original chip, install the new one, and set jumpers W4[2-3] and W7[1-2][3-4].

To install a new memory chip:

1. Turn off power to the SBS-2300.
2. Remove the memory chip from U2.
3. Orient the chip so pin 1 is closest to the card edge. Install the new chip into the socket.
4. Check and change, as necessary, jumper W4 and W7 to conform to the new memory.

RAM size	Jumper W4	Jumper W7
128K	[1-2]	[1-3][2-4]
512K	[2-3]	[1-2][3-4]

NOTE: The speaker is disabled when a 512K RAM is installed.

BATTERY BACKUP

A Dallas Semiconductor DS-1213D is used to battery

backup RAM when power is off. Battery life will depend upon RAM size, type, temperature, and time the SBS-2300 has power applied to it. You can expect the battery to last between 5 to 10 years for operation at 25°C. At 50°C, life is about 1/2 as much.

To install a DS-1213D, remove the RAM chip in U2, install the DS-1213D, and install the RAM chip on top of the module.

STORING VARIABLES IN RAM

The term "variables" in this context includes numbers, strings, arrays, recipes, and formulas as applied to your application.

Programs and CAMBASIC II variables reside in segment 0. Your variables are generally stored in segment 1 and higher. 32K of RAM is available for your program and variable storage in segment 0. The program and basic variables (A, B(15), C$, etc.) always reside in segment 0 and are cleared on reset. Variables you peek and poke to usually reside in segment 1 and above. Variables referenced by peek and poke statements. Each segment has an address range from 0 to 65535.

PEEK and POKE commands store and retrieve values from memory. For example:

```
20 POKE 12,A,1
```

puts the value of A into segment 1, address 12.

Use the PEEK statement to retrieve the variable:

```
50 B = PEEK(12, 1)
```

You can store and retrieve arrays, strings, and variables in this way. There are many variations of PEEK and POKE statements. Refer to the *CAMBASIC II Programming Manual* for additional information and examples. A list of commands appears at the end of this chapter.

CORRUPTED VARIABLES

RAM may be battery backed using a Dallas Semiconductor DS-1213D Smartsocket. When your application must rely on the accuracy of data after power up, corrupted variables becomes a possibility.

The nature of RAM is it is easily written to. Any POKE'd data is susceptible to corruption. This is especially true when the board is powered down. The DS-1213C monitors the supply voltage and turns off writing when it is below about 4.65 volts. However, when POKEing long data, such as strings and floating point numbers, a power down could interrupt a saving process. The result is information is corrupted. A scenario is explained below.

A program is running and POKEing data into RAM. At the same time it is poking, a reset occurs. A reset can occur due to power loss, someone pushing the reset button, or a watchdog timer time out.

If the program was POKEing a string (POKE$), floating point number (FPOKE), double byte (DPOKE), or array while the reset occurred, the data became corrupted. This is because the complete value was not saved.

Since it is impossible to predict or delay a reset, a work around is to duplicate or triplicate POKEd values. That is, you would have to save the same information in two or three different places. For purposes of discussion, POKEd variables are called sets because data can consist of a mixture of variables, strings and arrays.

On power up, your program would compare values from one set to the other one or two. If the two (or three) agreed, then there was no corruption and the program can reliably use the values. In practice, you would read information from set 1, but would save data to all two or three.

The use of duplicate or triplicate sets depends upon what the system must or can do if data is corrupted. When using a duplicate set, a corrupted set indicates that default values (from the program) should be used, since it is uncertain if the first or second set is corrupted. Both data sets would then be re-initialized.

A triplicate set is used to recover the last set or indicate that the data in the first set is valid. The procedure and logic is as follows.

Data is written to each element in a set in a specific and consistent order (data to an entire set does not have to be written to, just that element). For example, a calibration constant is saved (POKE'd) in three different places. Assume that the constant was assigned address 0, 100, and 200 in segment 1. The data is POKEd to address 0 first, then 100, then 200.

Upon reset, the calibration value is checked. If the value at address 0 agrees with address 100 and 200, then no corruption occurred. When address 0 and 100 agree but not 200, then this indicates that a reset occurred while updating the third set. The first data set can be trusted. The third data set simply needs to be updated.

When the first two sets do not agree, then you know that the first data is corrupted. If the second and third set agree, then, depending upon the system requirements, the first set could be "corrected" using the old data. The user or other device could be alerted that a calibration (or other) must be performed again. When all three sets disagree, then you must take action appropriate to the situation.

Another technique to check for valid memory is checksums. Simply write a program to add the values in RAM and compare it against a number is a good check. However, you cannot tell which data element was corrupted.

Instances of data corruption are rare. They do increase as the board power is cycled or reset.

ASSEMBLY LANGUAGE INTERFACE

Assembly language programs (including compiled C) must start from segment 0. Use the CAMBASIC II CALL statement to execute an assembly language program.

A specific area of RAM should be reserved for the program. This is to prevent strings and variables from corrupting that area of RAM. Use the SYS(1) and SYS(2) statements to do this. SYS(1) returns the low_wt memory location while SYS(2) returns the upper location. Run the program first to make sure variable memory has been allocated before running these SYS commands. Failure to do so may result in address returned that are not really free for assembly language programs.

There are several ways to put a program in memory, depending upon your application.

1. Use DATA statements and POKE the code into segment 0 RAM.

2. Write a program to download code. Some applications are connected to a larger system which "initializes" its systems. Using INKEY$ or COM$, code is received and then poked into memory using POKE$.

3. Read the code from the EPROM (U3) (using INP) and transfer it to RAM (using POKE).

4. Some space is available in the CAMBASIC II ROM. Space from about 6C00H to 7FFFH is available in version 1.0. The starting address will probably change in the future with different CAMBASIC II versions. You may burn your assembly language program in U1 and CALL in from BASIC.

In all cases, it is best to load code into RAM from a "secure" source. Even though RAM is battery backed, over time there is the possibility it could be corrupted.

Below is an example of loading and running an assembly language program.

```
100 FOR N = &FB00 TO &FB0C
110 READ A
120 POKE N,A
130 NEXT

900 DATA &DB, 2, &47, &E6, &FE, &D3
910 DATA 2, &78, &F6, 1, &D3, 2, &C9

2000 CALL &FB00
```

Lines 100 to 130 load the program into RAM. DATA statements may be entered manually or made by the MAKEDB program included with PC SmartLINK.

Line 2000 calls the program listed below. It toggles J2 line 13.

```
IN      A,(2)
LD      B,A
AND     0FEH
OUT     (2),A
LD      A,B
OR      1
OUT     (2),A
RET
```

COMMANDS

The following is a list of CAMBASIC II commands used with RAM.

Command	Function
CALL	Calls an assembly language routine

CLEAR Clears strings and allocates string space
PEEK Returns a byte
DPEEK Returns a 16 bit value
PEEK$ Returns a string
FPEEK Returns a floating point number
POKE Stores a byte
DPOKE Stores a 16 bit value
POKE$ Stores a string
FPOKE Stores a floating point number

INTRODUCTION

Digital I/O lines are used to interface with opto-module racks, switches, low current LED's, and other TTL devices. The SBS-2300 has 48 of these lines available through J2 and J3. Eight of these lines are high current outputs, capable of sinking 75 to 200 ma. Eight lines on J3 are shared by the keypad connector, J5. When the keypad is used, 8 of the 48 lines are not available.

Eight, 16, or 24 position opto racks are connected to J2 or J3. These opto racks accept G4 series opto modules. G4 series opto modules are used to sense the presence of AC or DC voltages or switch them. Maximum switching current is 3 amperes.

WARNING:

Apply power to the SBS-2300 before applying a voltage to the digital I/O lines to prevent current from flowing in and damaging devices. If you cannot apply power to the SBS-2300 first, contact technical support for suggestions appropriate to your application.

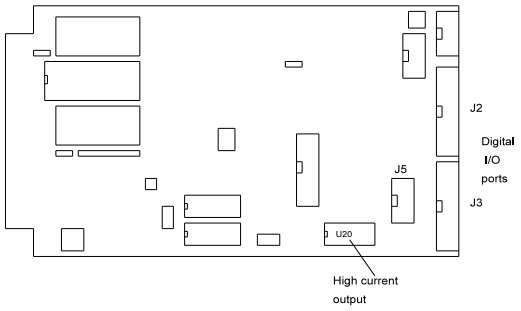


Figure 6-1 Digital I/O

DIGITAL I/O PORT

Digital I/O lines on the SBS-2300 are supplied by an 82C55 chip. The chip's lines go to connectors J2 and J3.

The lines on J2 and J3 are divided into 3 eight bit groups. Ports A and B can be configured as all inputs or outputs. Port C can be programmed as one group of 8 inputs or outputs or as two groups of four lines (upper and lower C). The four lines in upper and lower C can each be programmed as all inputs or outputs.

When a line is configured as an output, it can sink a maximum of 2.5 mA at 0.4V and can source over 2.5 mA at 2.4V. Outputs sink 15 mA at 1.0V.

J2 and J3 are accessed using CAMBASIC II LINE, OPTO, INP, and OUT statements. LINE reads or writes to a port based on the connector number. LINE is generally used with the STB-26 board. OPTO reads or writes to an opto module based on its position in an MPS opto rack. INP and OUT access a byte of data at a port.

The base I/O address for J2 is 0 and J3 is 64 when using INP, OUT, and CONFIG PIO statements. CONFIG PIO statement is used to configure the 8255 lines for inputs and outputs. Upon reset, lines are configured for inputs.

J2 and J3 are accessed using LINE or OPTO statements according to the table below.

Connector No	LINE # terminal	OPTO rack position
J2	1-25	0-23
J3	101 - 125	100 - 123

LINE #'s access that pin number on J2 or J3. LINE # 2 or 102 are not valid. Line 2 is + 5 volt supply.

J3, port A is connected to a high current sink through U20. See *High current output* later in this chapter.

J3 port C is shared with the keypad port J5. If you are using a keypad through J5, these 8 lines are not available.

Pull up resistors

Digital I/O lines at J2 and J3 are pulled up to + 5 volts through a 10K resistor pack.

These pull ups makes interfacing to switches and "open collector" TTL devices easy. See "Interfacing to Switches and other devices" below.

High current output

Eight lines at J3 can be used as high current drivers. These outputs switch loads to ground. Outputs are controlled by Port A on the 82C55. Its address is 64.

Logic outputs from this port are inverted. That is, when

a 1 is written to the high current port, the output is switched on and goes low.

The output driver chip, U20, can be replaced with a DIP shunt jumper so it is like the other lines at J3.

NOTE: Outputs at the high current lines are not compatible with TTL logic levels and should not be used to drive other logic devices.

Each of the high current outputs can sink 100 ma at 50V.

WARNING:

External supplies using the high current outputs must be tied to J3, pin 26 and NOT the power connector. Failure to do so can produce a ground loop and cause erratic operation.

The thermal time constant of U20 is very short, so the number of outputs that are on at any one time should include those that overlap even for a few milliseconds.

Incandescent lamps have a "cold" current of 11 times its operating current. Lamps requiring more than 50 ma should not be used.

Protection diodes must be used with inductive loads. Refer to figure 6-2

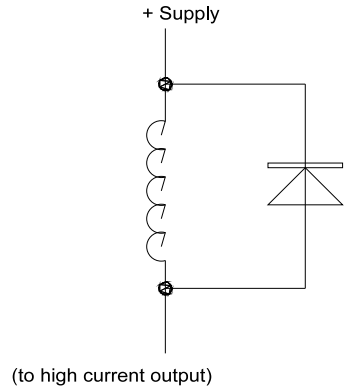


Figure 6-2 Inductive load protection

Do not parallel outputs for higher drive. This could result in damage since outputs do not share current equally.

Interfacing to an opto-module rack

J2 and J3 I/O lines interface to an MPS-8, 16, or 24 position opto module rack. Lines not going to an opto module connect to a screw terminal on the MPS-XX series boards. This feature allows you to connect switches or other TTL type devices to the digital I/O lines. The MPS-XX series boards accept G4 series modules.

Use the OPTO command to access and control G4 opto modules. The LINE command is used to access individual lines on the STB-26 or MPS-XX rack.

A CMA-26 connects J2 and J3 on the SBS-2300 to the MPS-XX board. Cable length should be less than 2 feet for the 8 position rack and 18 inches for the 16 and 24 positions. Excessive cable lengths cause a high voltage drop and consequently unreliable operation. Be sure to supply + 5V and ground to the appropriately marked terminals.

You must configure the 8255 ports for outputs before using them. Use the following table to determine the corresponding opto channel for a particular 82C55 port:

Opto channels	82C55 port	Connector	Addr.
M0-M3	Lower C	J2	2
M4-M7	Upper C	J2	2
M8-M15	A	J2	0
M16-M23	B	J2	1
M100-M103	Lower C	J3	66
M104-M107	Upper C	J3	66
M108-M115	A	J3	64
M116-M123	B	J3	65

"Opto channel" is the position as marked on the MPS-xx board. The channel number is preceded by a 'M' character on the MPS board. When connecting J3 to an opto rack, add 100 to the number on the rack. J3 has a high current output on port A (channels M8-M15). Replace U20 with a shunt jumper to operate normally.

To turn on an opto module, an output line must be low. A module is turned off by writing a '1' to a channel. The logic at J3 port A, with the high current outputs installed is just the reverse. A '1' at a line causes the module to turn ON.

High current outputs at J3 port A are optionally configurable as TTL I/O by replacing U20 with a DIP shunt jumper. This keeps logic compatible with ports B and C. If opto channels 8-15 are used as inputs, then U20 must be replaced by a DIP shunt jumper.

Configuring digital I/O lines

Lines are configured during program execution using the CONFIG PIO command. On power up or reset, all lines are inputs.

When a line is configured as an output, it can sink a maximum of 2.5 mA at 0.4V and can source a minimum of 2.5 ma at 2.4V. When driving opto modules, the outputs sink 15 mA at 1.0V.

Digital I/O programming example

The following example reads a switch at port A, bit 3 (J2-25), reads an opto module at channel 1 and writes an opto module at channel 5. A LED is controlled at J2-10 (port B, bit 0).

```
200 D = BIT(0,3)     :'Read switch status, port A
210 F = OPTO(101)    :'read opto module, ch. 1
220 OPTO 103,ON      :'write module, channel 3
230 BIT 1,0,0        :'turn on led at J2-10
240 BIT 1,0,1        :'turn off led at J2-10
250 A = LINE(103)    :'Reads pin 3 at J2
```

Connector pin out - J2

Pin #	82C55	Description	Opto Channel
19	Port A, line 0		8
21	Port A, line 1		9
23	Port A, line 2		10
25	Port A, line 3		11
24	Port A, line 4		12
22	Port A, line 5		13
20	Port A, line 6		14
18	Port A, line 7		15
10	Port B, line 0		16
8	Port B, line 1		17
4	Port B, line 2		18
6	Port B, line 3		19
1	Port B, line 4		20
3	Port B, line 5		21
5	Port B, line 6		22
7	Port B, line 7		23
13	Port C, line 0	Lower C	0
16	Port C, line 1	Lower C	1
15	Port C, line 2	Lower C	2
17	Port C, line 3	Lower C	3
14	Port C, line 4	Upper C	4
11	Port C, line 5	Upper C	5
12	Port C, line 6	Upper C	6
9	Port C, line 7	Upper C	7
26			Ground
2			+ 5V

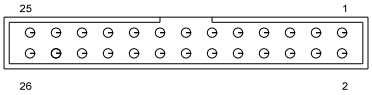


Figure 6-3 Digital I/O connector pin out (viewed from top)

Connector pin out - J3

Pin #	82C55	Description	Opto Channel
19	Port A, line 0	High current	8
21	Port A, line 1	High current	9
23	Port A, line 2	High current	10
25	Port A, line 3	High current	11
24	Port A, line 4	High current	12
22	Port A, line 5	High current	13
20	Port A, line 6	High current	14
18	Port A, line 7	High current	15
10	Port B, line 0		16
8	Port B, line 1		17
4	Port B, line 2		18
6	Port B, line 3		19
1	Port B, line 4		20
3	Port B, line 5		21
5	Port B, line 6		22
7	Port B, line 7		23
13	Port C, line 0	Shared w/J5 Keypad	0
16	Port C, line 1	Shared w/J5 Keypad	1
15	Port C, line 2	Shared w/J5 Keypad	2
17	Port C, line 3	Shared w/J5 Keypad	3
14	Port C, line 4	Shared w/J5 Keypad	4
11	Port C, line 5	Shared w/J5 keypad	5
12	Port C, line 6	Shared w/J5 keypad	6
9	Port C, line 7	Shared w/J5 keypad	7
26			Ground
2			+ 5V

COMMANDS

The following tables shows the CAMBASIC II commands used for digital I/O.

Command	Function
BIT	Function returns status of bit at an I/O address
BIT	Command sets a bit at an I/O address
CONFIG PIO	Configures J3 I/O port
INP	Returns a byte from an I/O address
LINE	Returns status of an opto line
OPTO	Sets an opto module output
OUT	Writes a byte to an I/O address
PULSE	Reads or writes a pulse at a port.

See also ON BIT, ON COUNT, ON INP, and related statements.

DESCRIPTION

The SBS-2300 has a battery backed Calendar/clock option using the DS-1216EM. When used in conjunction with the DATE$ and TIME$ commands, the current date and time can be set and read. The DS-1216EM is accurate to 1 minute/month at 25°C. Battery life is a minimum of 5 years.

INSTALLATION

The real time clock is installed in socket U3. Remove the IC in U3. Install the DS-1216EM in U3. Install the IC previously removed on top of the DS-1216EM.

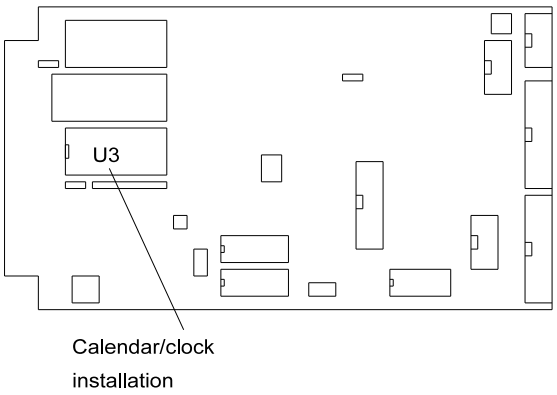


Figure 7-1 Calendar/clock installation

SETTING DATE AND TIME

The clock must be turned on before it is used. This need be done only once. To turn on the clock, type:

```
CONFIG CLOCK ON
```

The date and time can be set while running a program or in the immediate mode. Date and time are treated as strings and not numbers. To set the date and time:

```
date$="08-18-94"
time$="13:56:00"
```

To retrieve date and time as part of a program:

```
2000 DA$ = DATE$(0)
2010 TI$ = TIME$(0)
```

You can also print the date and time in the immediate mode:

```
pr time$(0)
13:56:03
```

COMMANDS

The following is a list of CAMBASIC II commands for the calendar/clock.

Command	Function
DATE$	Sets date
DATE$(0)	Returns date
TIME$	Sets time
TIME$(0)	Returns time

INTRODUCTION

The SBS-2300 has eight single ended or four differential analog input channels than can be interfaced to external analog devices. These channels can be used to measure voltages from transducers, 4-20 ma current loops, thermistors, etc. The converter reads a voltage and returns a 12 bit (4096 count) number in under a milli-second. Inputs are programmable for 0 to + 5 or ±5 volt, single ended or differential mode.

Additionally, 2 analog output channels with 12 bit accuracy are available. Output voltage is 0-5V, 0-10V, or ±5V.

Capacitors may be added to pads located adjacent to the A/D converter (U10). This will help reduce noise on analog inputs. Capacitor values are application dependent. 0.01 mfd is a good value to start from. Higher values may be used in extremely noisy environments or when time between samples is long (> 100 ms).

The AIN function is used to return a voltage while AOT commands an output.

This chapter begins with basic hook-up information, then proceeds to initialization, data reading, and calibration. Analog output option is discussed near the end.

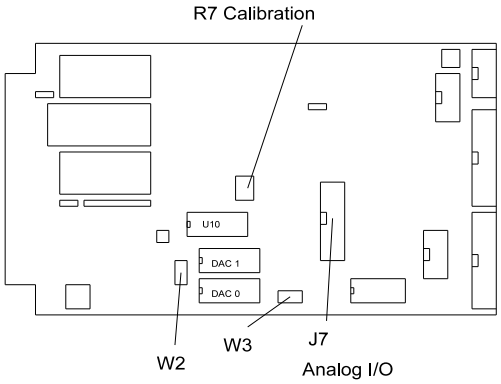


Figure 8-1 Analog I/O

CONNECTING ANALOG I/O

Analog I/O interface via J7. The STB-20 terminal board may be used to bring signals to terminal blocks.

The following table defines the signal pin out from analog I/O port J7.

J7 Pin #	Signal
1	CH0 input
2-16	Ground (even pins)
3	CH1 input
5	CH2 input
7	CH3 input
9	CH4 input
11	CH5 input
13	CH6 input
15	CH7 input
17	DAC 0 output
18	+ 12V
19	DAC 1 output
20	-12V

Initializing Inputs

The SBS-2300 can have up to eight single-ended inputs, four differential, or a mixture of single ended and differential inputs. Each analog input you intend to use must be initialized. Initialization is performed using the CONFIG AIN command. The syntax is:

```
CONFIG AIN channel, input, range
```

Where:

`channel` is from 0 to 7 for single-ended inputs or 0-6 for differential. Differential inputs require 2 lines. The channels you specify in a "mixed" application depends upon what lines are used for single ended and differential. Refer to the table below for single ended and differential channels

Polarity	+	-	+	-	+	-	+	-
J7 pin #	1	3	5	7	9	11	13	15
Differential Channel	0	0	2	2	4	4	6	6
Single ended Channel	0	1	2	3	4	5	6	7

For example, if you wanted one differential input,

channel 0 would use J7 pin numbers 1 and 3. Single ended inputs 2-7 are available.

`input` specifies single ended or differential. 0 = differential, 1 = single ended.

`range` is voltage input. 0 = ±5V and 1 = 0 to + 5V.

Below are sample syntaxes for the CONFIG AIN command:

1. Single ended mode, 0-5V input

```
CONFIG AIN chan,1,1
```

The input voltage is from 0 to 5 volts. The result from the AIN function is 0 for 0.000V and 4095 for + 4.9988V. `chan` may range from 0 to 7, if no other channels are used for differential inputs.

2. Differential mode, 0 to + 5V input

```
CONFIG AIN chan,0,1
```

`chan` can be 0, 2, 4, or 6. The input may range from 0 to + 5V. However, if the (-) input is more positive than the (+) input, the result will always be zero. The result from the AIN function is 0 for a difference of 0.000V and 4095 for a difference of 4.9988V.

3. Single ended, ±5V input

```
CONFIG AIN chan,1,0
```

The input ranges from -5V to + 5V. The result from an AIN function is 0 for -5.000V, 2048 for 0.000V, and 4095 for + 4.9988V.

4. Differential, ±5V input

```
CONFIG AIN chan,0,0
```

The input ranges from -5V to + 5V. The result is the difference of the two voltages. AIN will return 0 for a difference of -5.000V, 2048 for a difference of 0.000V, and 4095 for a difference of 4.9988V.

ACQUIRING ANALOG DATA

Once the analog input is initialized, the AIN function is used to acquire data. The syntax is:

```
S = AIN(ch)
```

Where: `ch` = channel number, 0-7

This command reads the voltage and returns a number from 0 to 4095 to the variable S. The number returned corresponds to the voltage input and the `type` the channel was configured for.

To convert the returned numbers to a voltage, use the following formulas:

5V Unipolar: A = 0.00122 * AIN(channel)
±5 Bipolar: A = 0.00244 * AIN(channel) - 5

The AIN function requires about 1 ms to convert a channel of data. Additional time is needed to store the data. Saving data to a single dimension array takes 500 micro-seconds longer than saving to a simple variable.

Datalogging on a timer tick

Some application require that data be taken at fixed intervals. The ON TICK construct can be used to take data in intervals from 0.01 to 655.35 seconds. The program below takes 100 samples on 2 channels every 10 seconds.

```
10 DIM F(100,2)
20 ON TICK 10 GOSUB 50
30 ..this is a dummy loop
40 GOTO 30
50 F(I,0) = AIN(0)
60 F(I,1) = AIN(1)
70 INC I
80 IF I = 100 THEN ON TICK 10 GOSUB
90 RETURN
```

Line 80 shuts off interrupts after 100 samples.

MEASURING HIGHER VOLTAGES

Input voltages higher than 5V are measured by placing a resistor in series with the input. Use the following formula to determine the required series resistance:

Rs = Vi * 20,000 - 100,000

Rs is the resistor value in ohms in series with the input. Vi is the maximum input voltage. If the result of your calculation is 0 or negative, a series resistor is not necessary.

NOTE: If an input voltage exceeds + 5.0V or is less than -5.0V, other channels will be in error.

Converting analog measurements

Inputs can be converted to engineering units of measurement by performing scaling calculations in the program. The AIN function returns values from 0 to 4095. To change these numbers into something more meaningful, use the following formula:

var = K * AIN(*n*)

n is the analog channel to read. K is the scaling constant. K is obtained by dividing the highest number in the range of units by the maximum AIN count (4095).

Example 1: To measure the results of an A/D conversion in volts and the voltage range is 0 to 5V, divided 5 by 4095 to obtain K.

K = 5/4095
K = .001221

Your program could look something like:

```
1000 C = .001221 * AIN(N)
```

Example 2: You want to measure a 0 to 200 PSI pressure transducer with a 0 to + 5V output. Divide 200 by 4095 to obtain the constant K.

K = 200 / 4095
K = .0488

The code can then look like:

```
1000 B = .0488*AIN(0)
```

CALIBRATION

The A/D converter is calibrated using an external voltmeter. For 12 bit accuracy, you must use a voltmeter with an accuracy of 0.02% or better.

To calibrate the SBS-2300:

1. Connect the digital voltmeter ground to U10, pin 11.
2. Connect the digital voltmeter '+ ' lead to U10, pin 14.
3. Adjust trim pot R7 for 5.000VDC.

You may increase the reference voltage to a higher value, up to 5.12V. This will allow you to detect if an input device is at or above its output range (5V).

ASSEMBLY LANGUAGE ACCESS

The A/D converter is serially accessed. The routines used to initialize, read and write to this chip are a part of CAMBASIC II and are not available for distribution. There are no external assembly language accesses (calls or jumps) available.

ANALOG OUTPUT

Two optional analog output channels can be independently configured for three voltage ranges. These ranges are jumpered in hardware. Refer to the following table for jumper settings.

Range	J7-17 (AOT 0)	J7-19 (AOT 1)
0 - + 5	W2[2-4]	W2[1-3]
0 - + 10	W2[8-10]	W2[7-9]
±5V	W2[6-8]	W2[5-7]

W3 is always set [1-2] and [5-6]. Channel 0 output is pin 17 and channel 1 is pin 19.

Programming voltage output

The AOT command is used to send data to an analog output. The syntax is:

AOT *channel, value*

Where:

channel specifies the analog channel to write data to and can be either 0 or 1. *channel* 0 is on pin 17 and 1 is on pin 19.

value is the value to output from 0 to 4095.

Use the following table to convert from a desired voltage to a *value*.

Range	Formula
0 - 5	*value*= V * 819

0 - 10	*value*= V * 409.5
± 5V	*value*= V * 409.5 + 2047.5

The result of the formula produces a number which can be used in place of *value*.

COMMANDS

The following is a list of CAMBASIC II commands for analog input and output.

Command	Function
CONFIG AIN	Configures analog input
AIN	Returns result of reading
AOT	Sends number to D/A converter

INTRODUCTION

16 position keypads are plugged into keypad port J5. Keys are arranged in a 4 x 4 matrix format. A key is recognized when a row and a column connect.

CAMBASIC II automatically scans and debounces the keypad every debounce time. Debounce time is fixed at 80 ms. Keypad presses may be returned either as a number from 1 to 16 or as an ASCII character. The ASCII character returned corresponds to those on Remote Processing's KP-1 keypad. Character assignments are changed using the SYS(8) function.

Keypads from Remote Processing simply plug into J5. Keypad cable length should be limited to less than 5 feet.

If the keypad port is not used, it may be used as a general purpose digital I/O port.

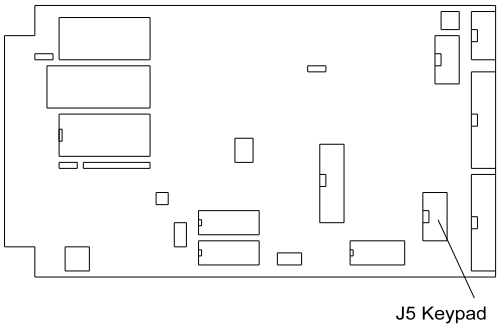


Figure 9-1 Keypad connector

PROGRAMMING EXAMPLE

The following example sets up CAMBASIC II to scan a 16 position keypad. The results are echo'ed to the display and the speaker is sounded when a key is pressed.

```
10 CONFIG PIO 1,0,1,1,0,64
20 'Optionally change keypad character 'B'
30 ' to the letter 'M'
40 POKE SYS(8)+7,77
60 ON KEYPAD$ GOSUB 500
70 PRINT " Enter a number";
100 'loop for this example
110 GOTO 100

500 A$ = KEYPAD$(0)
510 IF A$ = "C" THEN ..clear_beep
520 IF A$ = CHR$(13) THEN ..enter
530 PRINT A$;
540 B$ = B$+A$
560 RETURN

600 ..clear_beep
610 B$=""
630 DELAY .4
650 PRINT CHR$(12); "         ";CHR$(12);
660 RETURN

700 ..enter
710 FL = 1
730 RETURN
```

Program explanation

Lines 10-80 set up the parameters for the keypad. Lines 500 to 730 process the key press. If a "C" or "#" is pressed, it is an exception and is handled that way. Otherwise, the character is displayed and stored.

Lines 700 to 730 process the "enter" key. The enter flag, FL, is set to a 1 to indicate to another part of the program that B$ has complete data.

The KEYPAD$(0) function returns a single character string that has been assigned to a particular key. Characters are assigned using the SYS(8) statement.

KEYPAD PORT PIN OUT - J5

The keypad port uses port C from an 82C55. Lower port C is configured as an input. Upper port C are outputs.

The table below lists J5's pin out, 82C55 port and bit, and its intended function.

Pin	82C55 Port/bit	Function
1	C/0	Row 1
2	C/6	Column 3
3	C/5	Column 2
4	C/1	Row 2
5	C/2	Row 3
6	C/4	Column 1
7	C/7	Column 4
8	C/3	Row 4
9	nc	
10		Ground

Ground is not used.

COMMANDS

The following is a list of CAMBASIC II commands for the keypad.

Command	Function
INPUT KEYPAD$	Input data from a keypad
KEYPAD$(n)	Returns last key from keypad port
ON KEYPAD$	Causes a program branch when a key is pressed
SYS(8)	Returns keypad string address

DESCRIPTION

Pin 16 on the card edge connector is the speaker output from the 64180 CPU chip. This port can be used to drive a speaker. The SOUND command is used to generate a frequency.

Sound output is not available when a 512K RAM is installed. This is due to the fact one of the address lines is also used as a frequency output.

SOUND Syntax is:
SOUND *frequency*
SOUND *frequency,duration*

frequency is from about 20 Hz to over 20,000 Hz.
duration is from 0.01 to 325 seconds.

NOTE: When SOUND is used with a *duration* parameter, program execution is suspended until it is timed out.

Speaker output is controlled by jumper W7.

[1-2],[3-4] 512K RAM installed
[1-3],[2-4] Speaker enabled

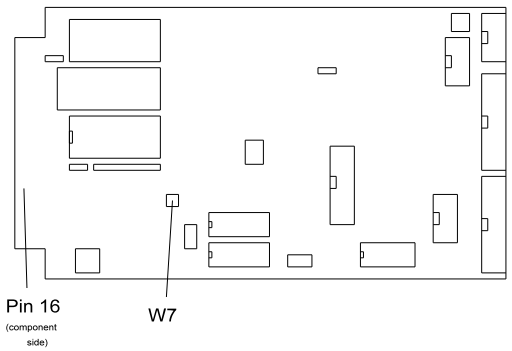


Figure 10-1 Jumper W7

Connecting a speaker

Refer to figure 10-2 below for circuit connections to a speaker. The series resistor determines the volume. the Capacitor sets the lower frequency limit. Generally, values from 100 uF to 470 uF are adequate. The speaker may be any value but those with 50 ohms or greater produce higher sound output.

WARNING:

Do not connect pin 16 directly to a speaker, ground, or + 5V, even momentarily, as damage to the CPU will result.

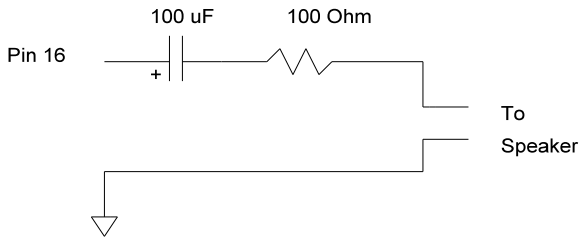


Figure 10-2 Interface schematic

Programming example

The following example produces tones from 200 to 5000 hz and back down again in 500 hz increments.

```
10 FOR N = 200 TO 5000 STEP 500
20  SOUND N,.5
30 NEXT
40 FOR N = 5000 TO 200 STEP -500
50 SOUND N,.5
60 NEXT
70 GOTO 10
```

To stop program execution, press the < esc> key.

AIN and AOT commands shut off the sound.

DESCRIPTION

The watchdog timer is used to reset the SBS-2300 if the program or CPU "crashes". When enabled, the program must write to I/O address &E8 to avoid a reset. The timeout is adjustable for 150 ms, 600 ms, or 1.2 seconds.

The watchdog should be disabled when using INPUT and DELAY statements. Also, loops which do not end quickly or are of indeterminate duration should be avoided unless a timer reset pulse is included. An example of an indeterminate loop is one that waits for a port condition to change.

The watchdog is enabled by writing a 1 to address &E4, bit 1 and disabled by writing a 0 to the same location (use the BIT statement to do this). The timer is reset by a write of any data to I/O address &E8.

The watchdog timer is part of a voltage monitor, battery backup controller, and reset chip U14.

Watchdog time is determined by jumper WD. The following table determines the timeout.

Pins	Timeout
[1-2]	150 ms
no jumper	600 ms
[2-3]	1.2 seconds

PROGRAM EXAMPLES

The following program fragments enable the watchdog timer, reset it while the program is running, and then disables it.

```
100 BIT &E4,0,1 :'Turn on watchdog
.
.
.
5000 OUT &E8,0 :'Reset timer
.
.
.
10000 BIT &E4,0,0 :'Turn off watchdog
```

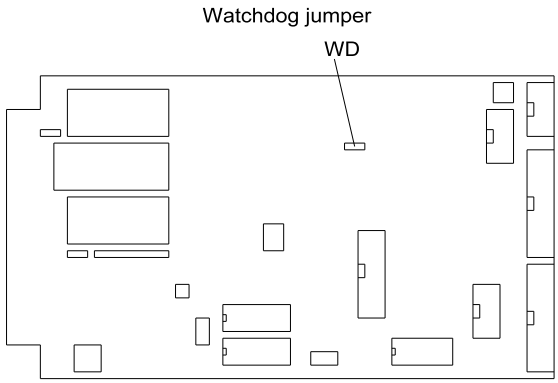


Figure 11-1 Watchdog timer

ELECTRICAL

CPU
64180, 9.216 Mhz clock

Memory
CAMBASIC II, 32K ROM

Programming and data is 128K RAM standard, 512K Optional.

Program is 32K EPROM (U3)

Memory speeds are 80 ns or faster

Digital I/O
The SBS-2300 has 48 digital I/O lines. 24 are from J2, which is a general purpose port. The other 24 are from J3. J3 has 8 high current outputs, which may be jumpered for inputs. The keypad port, J5, uses 8 of the 24 lines. All ports use an 82C55 for interfacing.

The specifications below are for all digital I/O except for the eight high current lines at J3.

Drive current	2.5 ma maximum per line, sink or source. TTL compatible.
Output low voltage	0.45V max at 2.5 mA, 1V max at 15 mA for opto rack.
Output high volts	2.4V minimum, sink or source at rated current.

All digital input lines are TTL compatible.

High current output at J3

8 of the 24 lines can drive up to 500 ma at 50V. Refer to *CHAPTER 6, DIGITAL AND OPTO PORTS* for limitations.

Keypad input
10 lines accept a 16 position matrix keypad. Scanning and debounce performed in CAMBASIC II.

Serial ports
Two RS-232D serial ports. All have RxD, TxD, and CTS lines. COM1 has only these lines. COM2 also has an RTS line. Baud rates from 600 to 38.4K, 7 or 8 data bits, parity even, odd, or none, 1 or 2 stop bits.

EPROM and programmer
Accepts 29C256 or equivalent EPROM.
Size:32K
Speed:120ns or faster.

Watchdog timer, reset
Watch dog timer resets card for 150 ms minimum when enabled. Push button reset included.

Power requirements
+ 5VDC ±5% at 300 ma.
RS-232 voltages generated on card.
Current consumption does not include any opto-modules or other accessories.

MECHANICAL

Size: 4.5" x 8.0"
Maximum height: 0.675", no cables installed.

Mounting holes: 4 each corner. Hole size is 0.156" dia. See drawing.

MEMORY AND I/O MAP

Memory

Description		Address
CAMBASIC II	U1	&00000 - &07FFF
RAM, U2	128K	&08000 - &1FFFF
	512K	&08000 - &7FFFF

I/O

Description	Address
J2 Digital	&0000 - &0003
J3 Digital	&0040 - &0043
Keypad	&0040 - &0043
Watchdog	&00E4 - &00E4
Watchdog timer reset	&00E8 - &00E8
Internal processor	&0080 - &00BF
DAC 0	&00C0 - &00CF
DAC 1	&00D0 - &00DF
Program flash EPROM U3	&0300 - &7FFF

Only I/O address &8000 to &FFFF are available off card. No memory addresses are available off card.

JUMPER DESCRIPTIONS

A * after a jumper position indicates factory default and is jumpered.

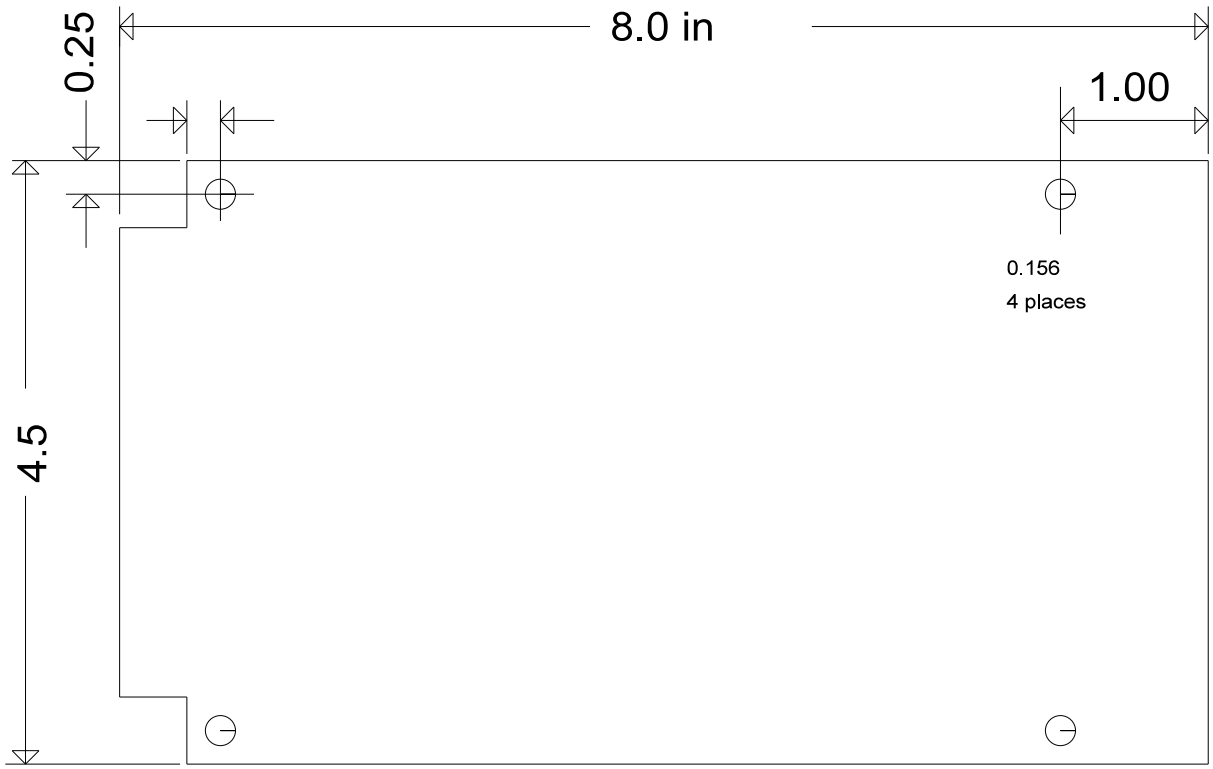
Jumper	Description
W1	U3 Select
[9-10]	Autorun enable
W2	Analog output range
[1-3]*	Ch1 0-5V
[2-4]*	Ch0 0-5V
[7-9]	Ch1 0-10V
[8-10]	Ch0 0-10V
[5-7]	Ch1 ± 5V
[6-8}	Ch0 ± 5V
W3	DAC voltage supply select
[3-4]*	Internal supply (Ch0)
[7-8]*	Internal supply (Ch1)
W4	U2 System RAM size
[1-2]*	32K or 128K RAM
[2-3]	256K or larger
W6	User memory device type select
[1-2]*	29C256 flash EPROM
no jumper	Other device
W7	RAM/Sound select
[1-2],[3-4]	512K RAM enable
[1-3],[2-4]*	Sound output enable
WD	Watchdog timer
[1-2]	150 ms
[2-3]*	1.2 seconds
no jumper	600 ms

Edge connector

Line	Signal	Description
A	+ 5	Power
B	D0	Data 0
C	D2	Data 2
D	D4	Data 4
E	D6	Data 6
F	A0	Address A0
H	A2	Address A2
J	A4	Address A4
K	A6	Address A6
L	A8	Address A8
M	A10	Address A10
N	A12	Address A12
P	A14	Address A14
R	IWR*	I/O write
S	HOLD*	CPU hold
T	-	not used
U	INT0	Interrupt 0
V	CLK	CPU clock
W	PS*	Peripheral select
X	+ 12.7	Power
Y	-	not used
Z	Gnd	Power ground
1	+ 5V	Power
2	D1	Data D1
3	D3	Data D3
4	D5	Data D5
5	D7	Data D7
6	A1	Address A1
7	A3	Address A3
8	A5	Address A5
9	A7	Address A7
10	A9	Address A9
11	A11	Address A11
12	A13	Address A13
13	A15	Address A15
14	IRD*	I/O Read strobe
15	-	not used
16	Sound	Speaker port
17	-	not used
18	RES*	Reset input
19	INT1	Interrupt input
20	-	not used
21	-12V	Power
22	Ground	Power ground

Serial Port pin out

Pin #	J1 Signal	J4 Signal
1	NC	NC
2	RxD	RxD
3	TxD	TxD
4	NC	NC
5	Gnd	Gnd
6	NC	NC
7	NC	CTS
8	RTS	RTS
9	+ 5	+ 5
10	NC	NC



Card edge connector is on .156 in centers
with .093 contact width.

SBS-2300 board outline