Dual Serial Board (DSB)

For the Commodore Amiga

Hardware, Software, and Manual
by
ASDG, Incorporated

July 6, 1992
Second Printing

ASDG, Incorporated ("ASDG") makes no warranties, either expressed or implied, with respect to the program described herein, its quality, performance, merchantability, or fitness for any particular purpose. This program is sold "as is." The entire risk as to its quality and performance is with the buyer. Should the program prove defective following its purchase, the buyer (and not ASDG, their distributors or their retailers) assumes the entire cost of all necessary repair, correction, or servicing. In no event will ASDG be liable for direct, indirect, incidental or consequential damage, or damages resulting from loss of use or loss of anticipated profits resulting from any defect in the program even if it has been advised of the possibility of such damage. Some laws do not allow the exclusion or limitation of implied warranties or liabilities for incidental or consequential damages, so the above limitations or exclusion may not apply.

You have the non-exclusive right to use the enclosed program only on a single computer. You may physically transfer the program from one computer to another provided that the program is used on only one computer at a time. However, you may not electronically transfer the program from one computer to another over a network. You may not distribute copies of the program or the accompanying documentation to others either for a fee or without charge. You may not modify or translate the program or documentation. You may not disassemble the program or allow it to be disassembled into its constituent source codes. Your use of the program indicates your acceptance of these terms and conditions. If you do not agree to these conditions, return the program, documentation, and associated peripherals to the vendor from whom this software was purchased.

This license agreement shall be governed by the laws of the United States of America, State of Wisconsin and shall inure to the benefit of ASDG, Incorporated or its assigns.

In accordance with the Computer Software Rental Act of 1990, this software may not be rented, lent or leased.

This software and its documentation may not be provided by a "backup service" or any other vendor which does not provide an original package as composed by ASDG, Incorporated including but not limited to all original documentation, insertions, registration cards, and software.

Printed in the USA.

Contents

1 **FCC Compliance** 1

2 **Specifications** 3

2.1 Electrical Specifications 3

2.2 Environmental Specifications 3

2.3 Mechanical Specifications 3

2.4 Connector Pinouts 4

3 **Installation** 5

3.1 Installing the Board 5

3.2 Installing the Software 6

3.3 Confirming Board Installation 6

3.4 Connecting to an Amiga Serial Port 6

3.5 Converting from DSB Connector to Standard Amiga Pinouts 6

4 **Software Support Levels** 9

4.1 Exec Level Support 9

4.2 Additional Features, Notes, and Limitations . . . 11

4.2.1 Modifying the Control Line Status 11

4.2.2 Error Reporting 13

4.2.3 Known Limitations 13

4.3 DOS Level Support 13

4.3.1 Setting Serial Parameters on The Command Line 14

4.3.2 Setting MountList Entry Defaults 16

4.3.3 MountList Entries 18

4.4 The Serial DisPatcher 20

4.4.1 Installing the SDP 20

4.4.2 Using the SDP 21

4.4.3 Not Using the SDP 21

5 Troubleshooting **23**

5.1 Technical Support 23

5.2 Service and Repair Information 23

6 Extras **25**

6.1 Support of Higher Baud Rates 25

6.2 Other Software on the Distribution Disk 25

6.3 Using High Speed PostScript Printers 26

6.3.1 Before You Start 26

6.3.2 Setting the Printer Speed 27

6.3.3 Configuring a High Speed Port 29

6.3.4 Using the High Speed Port 30

Chapter 1

FCC Compliance

WARNING: This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instructions manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

If you suspect interference, you can test the product by turning the Amiga off and on with this product installed and not installed. If this product does cause interference, try the following:

1. Reorient the antenna or AC plug on the affected equipment.
2. Change the relative positions of the Amiga and the affected equipment.
3. Move the Amiga farther away from the affected equipment.
4. Plug either the Amiga or the affected equipment into a different outlet so that the Amiga and the affected equipment

are on different circuits.

Use only shield-grounded cables when connecting peripherals to the Amiga.

All peripherals must be labeled to comply with the FCC emissions requirements. Class B certified devices will usually have lower emissions than Class A devices.

Operation with unlabeled peripherals is likely to result in interference.

Use this equipment only with three-pronged type (AC ground) AC wall recepticals.

If necessary, consult your dealer or an experienced Radio Frequency Interference technician for additional suggestions. You may find the FCC booklet "How to Identify and Resolve Radio-TV Interference Problems" helpful. It is available from the U.S. Government Printing Office, Washington, D.C. 20402, stock no. 004-000-00345-4.

Chapter 2

Specifications

2.1 Electrical Specifications

Power Requirements:	+4.75 to 5.25 Vdc at 1.37 A Max. +11.4 to +12.6 Vdc at 50 mA Max. -12.6 to -11.4 Vdc at 50 mA Max.

2.2 Environmental Specifications

The Dual Serial Board is rated for the following environmental conditions.

Storage Temperature:	-40 to +70	degrees C
Operating Temperature:	0 to +55	degress C
Relative Humidity:	5 to 85%	(non-condensing)

2.3 Mechanical Specifications

Physical dimensions:	Maximum 13.85"l x 4.45"h x 0.85"w (including rear panel bracket)

Connector Pin Number	Signal Name and Abbreviation	Signal Direction
1	Data Carrier Detect (DCD)	Input
2	Receive Data (RxD)	Input
3	Transmit Data (TxD)	Output
4	Data Terminal Ready (DTR)	Output
5	Signal Ground (GND)	—
6	Data Set Ready (DSR)	Input
7	Request To Send (RTS)	Output
8	Clear To Send (CTS)	Input
9	Ring Indicator (RI)	Input

Table 2.1: Dual Serial Board Connector Pinouts

2.4 Connector Pinouts

The connectors for both port 0 and port 1 are male 9 pin subminiature D style connectors with IBM PC-AT standard configuration pinouts. The signal name and direction for each pin are shown in Table 1.

Chapter 3

Installation

3.1 Installing the Board

A small to medium size Phillips screw driver is required to install a Dual Serial Board in an Amiga 2000 or 3000.

Remove the screws along the bottom of the sides of the Amiga. There are two screws along the lower edge of each side of the machine.

Remove the screw in the top center of the back of the Amiga.

Grasp the cover on both sides and slide it toward the front of the Amiga and up.

Unscrew the bracket for the slot the Dual Serial Board will be mounted in and remove it from the system. Install the Dual Serial Board in the slot you have chosen, making sure that the board seats firmly in the connector. Re-use the screw you just removed to screw the Dual Serial Board bracket in place.

Re-install the cover of the Amiga by reversing the steps you used to remove it.

3.2 Installing the Software

The Dual Serial Board software driver is installed by copying `siosbx.device` to your `DEVS:` directory. This can be accomplished by double clicking on the `Install` icon in the root directory of the distribution diskette.

The `Install` program will also copy the DOS Handlers (which allow CLI users to access the serial expansion ports) into your `L:` directory.

The `Install` program will also install ASDG's Serial DisPatcher if you so choose. The Serial DisPatcher is described in a later section of this manual, which you should read before running the `Install` program.

3.3 Confirming Board Installation

Double clicking on the `ICheck` icon in the root directory of the distribution diskette will confirm that the DSB has properly auto-configured and will be found by the software drivers.

3.4 Connecting to an Amiga Serial Port

Table 3.1 defines the pinouts of a cable which will allow a DSB serial port to connect to the Amiga serial port. Both the 9 and 25 pin connectors in this cable must be female to mate with the connectors on the DSB board and the Amiga.

3.5 Converting from DSB Connector to Standard Amiga Pinouts

Table 3 defines the pinouts for a cable which will adapt a DSB port to look like the standard Amiga serial port. In other words,

9 Pin Connector Pin Number	Signal Abbreviation	25 Pin Connector Pin Number
1	DCD	-
2	RxD	2
3	TxD	3
4	DTR	6
5	GND	7
6	DSR	20
7	RTS	5
8	CTS	4
9	RI	-

Table 3.1: DSB to Amiga Cable Pinout

if you use this cable, you can plug any cables which would normally be plugged into the Amiga port into the 25 pin end of this cable. In this case the 9 pin connector must be female and the 25 pin connector must be a male.

9 Pin Connector Pin Number	Signal Abbreviation	25 Pin Connector Pin Number
1	DCD	-
2	RxD	3
3	TxD	2
4	DTR	20
5	GND	7
6	DSR	6
7	RTS	4
8	CTS	5
9	RI	-

Table 3.2: DSB to Amiga Pinout Adapter

Chapter 4

Software Support Levels

The Amiga personal computer comes with a single RS-232 style interface. The interface is supported by the operating system on two levels. These are the Exec level and the DOS level.

The Exec level software support consists of the Exec device driver, `serial.device`, and is the programmer's principal interface to the serial port.

The DOS level software support is provided by the "port-handler" and is called `SER:`. This is the CLI user's principal interface to the serial port.

The DSB provides both the Exec and DOS levels of support for up to 32 additional serial ports. Exec level support is provided by the program `siosbx.device`, which should reside in your `DEVS:` directory. DOS level support is provided by `SERX-Handler` (and `SERXI-Handler`) which should reside in your `L:` directory.

4.1 Exec Level Support

Programming the `siosbx.device` is performed in exactly the same way as programming the standard `serial.device`. Only two minor differences (both in the call to OpenDevice()) are to be noted.

First some background. OpenDevice() takes the following form:

```
error = OpenDevice( DevName, UnitNum, IOReq, Flags )
```

where:

DevName is a pointer to a string containing the name of the device.

UnitNum is an integer containing the unit number to be opened.

IOReq is a pointer to an **IORequest** block which will be filled in the device driver and used to communicate with the device.

Flags contains additional driver specific information. Usually set to zero.

With this in mind, a call to OpenDevice() for the Amiga serial device usually looks like this:

```
if (OpenDevice( "serial.device", 0, IOB, 0 )) ...
```

The two differences are:

1. Rather than passing a pointer to a string constant containing **"serial.device"**, make this argument a string pointer variable which can point to either **serial.device** or **siosbx.device**.

2. Instead of passing the constant 0 as the unit number, make this argument an integer variable. This argument will be used to select between the possibly many (**siosbx.device**) ports you might have on your system.

So, some example code might look like:

```
char *AmigaSerial = "serial.device";
char *ASDGSerial  = "siosbx.device";

DevName = AskUserWhichDevice() ? ASDGSerial
                               : AmigaSerial;
if (DevName == ASDGSerial)
    UnitNum = AskUserWhichUnit();
else UnitNum = 0;

Result = OpenDevice( DevName, UnitNum, IOB, 0 );
```

Where **AskUserWhichDevice()** and **AskUserWhichUnit()** are hypothetical routines that allow the user to specify which device and unit to use.

Other than the **OpenDevice()** call, no other changes are required. You may use the Amiga Rom Kernal Manual for your programming documentation.

4.2 Additional Features, Notes, and Limitations

These are some of the additional features that the DSB's Exec level driver provides beyond Commodore's standard driver. Also listed is the DSB driver's one limitation.

4.2.1 Modifying the Control Line Status

Several users have requested the ability to directly manipulate the status of two control lines: RTS and DTR. Normally, the DTR line can be lowered by closing the device. When the device is reopened, the DTR line will again be raised. The Amiga serial device provides no way to manipulate the RTS line at all or affect the DTR line beyond what was just stated.

To facilitate the manipulation of these lines while the device is still open, we provide the following non-standard command:

Name SetCtrlLines — Allows setting of various control lines.

`Function` This function allows the state of various control lines to be affected without closing the expansion serial device. The effect of the command will take place immediately.

`IO Request` NOTE: Actually defining the mnemonics is left to the programmer. Their suggested names are supplied here with their intended values.

`io_Command` `SIOCMD_SETCTRLLINES` (16 decimal or 0x10 in hexadecimal).

`io_Offset` A bit mask containing a one in each bit position which will be changed. Any bit positions with a zero value will be unchanged irrespective of the value contained in `io_Length`.

`io_Length` The values each affected bit should be set to. Any bit position which has a zero in the corresponding position in `io_Offset` will be ignored.

The valid bit positions (masks) are:

`SIO_SET_RTS` Bit position 0.

`SIO_SET_DTR` Bit position 1.

Thus, to turn off RTS, you would perform:

```
IOB->io_Command = 0x10; /* or 16 or define own */
                        /*    mnemonic         */
IOB->io_Offset  = 1;    /* specify bit 0 be    */
                        /*    changed          */
IOB->io_Length  = 0;    /* it will be changed  */
                        /*    to a 0           */
DoIO( IOB );
```

Or, to turn RTS off and DTR on at the same time, you would perform:

```
IOB->io_Command = 0x10; /* or 16 or define own */
                        /*     mnemonic        */
IOB->io_Offset  = 3;    /* specify bits 0 & 1  */
                        /*     be changed      */
IOB->io_Length  = 2;    /* bit 1 set to 1,     */
                        /* bit 0 set to 0      */
DoIO( IOB );
```

4.2.2 Error Reporting

The current Amiga Exec Level support (embodied in serial.device) removed documented support for various error status messages, such as detecting invalid baud rates. The DSB Exec Level driver reports all error messages correctly as documented in the Rom Kernel Manual (even if the Amiga serial device does not.)

The programmer can choose to deal with these error conditions as desired. Supporting these conditions will provide more robust functionality when operating on a DSB port and makes no difference when using the Amiga's serial port.

4.2.3 Known Limitations

The DSB driver cannot provide mark or space parity. Setting mark or space parity through the DSB will set no parity checking. Odd and even parity function properly.

4.3 DOS Level Support

The DSB comes with two DOS handlers, `SERX-Handler` and `SERXI-Handler`. The former is similar to the standard DOS handler `SER:`, while the latter is similar to the standard DOS handler `AUX:`. That is, SERX-Handler provides buffered input/output while SERXI-Handler provides unbuffered (or interactive) input/output.

The DSB DOS handlers provide considerably greater flexibility than their standard DOS counterparts in that they provide the ability to dynamically change serial parameters at will.

This is accomplished via three stages of control.

1. Serial parameters can be specified on the command line allowing the CLI user to have total control over serial line characteristics. This is accomplished by specifying the serial parameters as if they were directory and file names on the serial device.
2. Any parameters not specified on the command line can be set to defaults specified in the `DEVS:MountList` file.
3. Any parameters not specified either on the command line or in the `DEVS:MountList` file will default to the values specified in the **`Serial Preferences`** editor for the Amiga's standard serial port.

That is, serial parameters can be set on a command by command basis or via two levels of default settings.

4.3.1 Setting Serial Parameters on The Command Line

The syntax of command line specification of serial parameters is given as follows:

device name:baud/data format/buffer size/handshaking

where:

devicename Is any name you desire. This must correspond to an entry in your `DEVS:MountList` file. NOTE: Unit differentiation is done in the `DEVS:MountList` file and, consequently, is reflected in the device's name.

baud Is any number between 110 and 292000. This will be used as the baud rate of the serial line. If this argument is missing, the baud rate will be chosen from the handler's two levels of defaults.

dataformat Is a 3 character string of the form:

nbits parity nstop

where:

nbits Is the number of bits which will be sent in each byte. This number may be between 5 and 8.

parity Is the parity setting. This may be the letters "N" for none, "E" for even, or "O" for odd. This is **not** case sensitive.

nstop This is the numver of stop bits which will be sent after each byte. This may be 1 or 2. If *nbits* is 8, then *nstop* must be 1.
If this parameter is missing, data length, parity, and number of stop bits will be chosen from the handler's two levels of defaults.

buffersize Is any number larger than 64. This represents the number of bytes which will be requested as the size of the serial receive buffer. Obviously, this number is limited by the availability of memory. One reasonable rule of thumb is "the higher the baud rate, the larger the buffer." If this parameter is missing, a value will be chosen from the handler's two levels of defaults.

handshaking Is a number between 0 and 3 (inclusive), which specifies what type of handshaking is to be used. If missing, the handler chooses handshaking from its two levels of defaults. The values represent:

0	No handshaking
1	XON/XOFF handshaking
2	CTS/RTS handshaking
3	Both XON/XOFF and CTS/RTS handshaking.

Notice that any missing argument will cause the handler to choose a default setting either from the `DEVS:MountList` file or from the `Serial Preferences`.

For example, the following command line will copy the named file to an expansion serial line set entirely to default settings:

```
copy S:startup-sequence Ser3:
```

The following command line will copy the named file to an expansion serial line set to 9600 baud with a 20000 byte buffer. All other parameters will be set to the defaults:

```
copy S:startup-sequence Line2:9600//20000
```

The following command line will copy the named files to an expansion serial line set to 19200 baud with 7 data bits, no parity, 2 stop bits, a large buffer, and XON/XOFF handshaking.

```
copy S:startup-sequence Micky:19200/7N2/30000/1
```

In the above examples, note that the expansion serial port name is entirely dependent upon a corresponding `DEVS:MountList` entry.

4.3.2 Setting MountList Entry Defaults

As described above, any serial parameters not specified on the command line will be set according to two levels of defaults. The first level of default is any settings you may have made in the `DEVS:MountList` file. These settings are communicated to the handler by setting an appropriate value on a `DEVS:MountList` entry `Startup` line. This takes the form of:

```
Startup = 0xNNNNNNNN
```

The eight digit hexadecimal number is composed of fields which will be described below. Note that any fields specified to be 0 mean that the parameter will be set according to the handler's second means of default—`Serial Preferences`. Also note that the `Startup` number is parsed right to left by the handler.

The two rightmost digits specify the unit number. This determines which DSB expansion serial port will respond to this `DEVS:MountList` entry.

- If you are willing to accept the handler's second level of default (`Serial Preferences`), for all serial parameter settings, then all that needs be set is the unit number.
- The third digit from the right specifies the default baud rate. A value of 0 causes the handler to use the default baud rate specified by `Serial Preferences`. The other legal values are:

1	110 Baud
2	300 Baud
3	1200 Baud
4	2400 Baud
5	4800 Baud
6	9600 Baud
7	19200 Baud
9	38400 Baud
A	57600 Baud
B	76800 Baud

- The fourth digit from the right specifies the data byte size. This may be a value between 5 and 8, inclusive. If 8 bit data bytes are chosen, then the number of stop bits must be 1. If this parameter is 0, then the value set in `Serial Preferences` will be used.
- The fifth digit from the right specifies the type of parity check which will be used. Legal values are 1 through 3, inclusive. They represent no parity, even parity, and odd parity, respectively. If this digit is 0, the value set in `Serial Preferences` will be used.
- The sixth digit from the right specifies the number of stop bits which will be used on the serial line. If the number of data bits is chosen to be 8, then the number of stop bits must be 1. Legal values are 1 and 2. If this digit is 0, the value set in `Serial Preferences` will be used.
- The seventh digit from the right specifies the size of the serial receive buffer. These sizes were chosen to correspond to

the sizes which can be specified in `Serial Preferences`. If this digit is 0 or not present, the serial receive buffer size set in `Serial Preferences` will be used. The legal values are:

1	512 Bytes
2	1024 Bytes
3	2048 Bytes
4	4096 Bytes
5	8000 Bytes
6	16000 Bytes

- The eighth digit from the right specifies the type of handshaking which will be used on the serial line. Legal values are 1 through 4, inclusive. If zero is specified, then handshaking style will be set to match that specified by `Serial Preferences`. The legal values are:

1	XON/XOFF Handshaking
2	CTS/RTS Handshaking
3	Both XON/XOFF and CTS/RTS Handshaking
4	No Handshaking

When specifying the value of `Startup`, leading zeros may be deleted.

4.3.3 MountList Entries

An entry in `DEVS:MountList` is necessary to associate a device name (such as `Line5:`) with a handler and unit number. As described in the previous section, the `DEVS:MountList` entry also specifies defaults for various serial line parameters.

The following lines are required for buffered serial lines (similar to `SER:`):

```
devname:  Handler = L:SERX-Handler
          Stacksize = 2000
          Priority = 5
          GlobVec = -1
#
```

The following lines are required for unbuffered serial lines (similar to `AUX:`):

```
devname:    Handler = L:SERXI-Handler
            Stacksize = 2000
            Priority = 5
            GlobVec = -1
#
```

In both cases, <**devname**> refers to any name you wish to associate with the serial line. **NOTE:** The trailing # must be present!

If a **Startup** line is not specified, the unit referred to will default to unit 0.

For example, to set up both an interactive (unbuffered) and a buffered serial device on unit 3 running at 9600 baud with XON/XOFF handshaking and no parity, one would specify:

```
Line3:       Handler = L:SERX-Handler
             Stacksize = 2000
             Priority = 5
             GlobVec = -1
             Startup = 0x10010603
#

MikesTerm:   Handler = L:SERXI-Handler
             Stacksize = 2000
             Priority = 5
             GlobVec = -1
             Startup = 0x10010603
#
```

A reference to **Line3:** would result in communications over the buffered line. Referring to **MikesTerm:** would result in communications over the unbuffered (or interactive) line. References to both devices may be made, although not at the same time. In order to use these devices, the following lines would have to be executed from some CLI (like the **startup-sequence**, for example):

```
mount Line3:
mount MikesTerm:
```

4.4 The Serial DisPatcher

Provided on the DSB distribution disk is an exclusive software utility which we call the Serial DisPatcher (SDP). Using the SDP, you will be able to use any communications software (which uses the Amiga's standard serial device) on any DSB serial port with no modifications to the executable program.

Also, using the SDP, you can reroute a serial printer to any DSB serial port with total transparency to the rest of the system.

The SDP accomplishes this by impersonating the Amiga's Exec level driver, `serial.device`. Commodore's device driver is renamed `cbmser.device` and is opened by the SDP should you actually want to use the Amiga's serial port.

It is not possible for the SDP to decrease throughput or affect system performance in any detrimental way (so don't even think it).

4.4.1 Installing the SDP

The installation of the SDP is accomplished by answering `y` to the question `Install Serial DisPatcher? (y|n)` asked while running the `Install` program.

Answering in the affirmative will cause the `Install` program to replace your `serial.device` with the SDP. A slightly modified version of Commodore's serial.device will be copied into your `DEVS:` directory under the name `cbmser.device`. Your original Commodore supplied device driver will be renamed `original.serial.device`, should you wish to keep it available as a backup.

The slightly modified version of Commodore's serial device driver contained on the distribution disk is the V1.3 `serial.device`. The only modification is to replace the internally occuring string "serial.device" with the string "cbmser.device". This allows the program to refer to itself properly.

4.4.2 Using the SDP

After installing the SDP you may find it necessary to reboot your system if you had already been using the Amiga's serial device driver since the last time you rebooted.

After having done so, if needed, each time any program attempts to open `serial.device`, the SDP will present a requester asking you if you wish to use the standard Amiga serial port. If you click on the **Yes** gadget, your program will operate as if nothing had happened using the standard Amiga serial port.

If you click on the **No** gadget, you will be asked to specify the unit number of the DSB port you wish your program to use. The calling program will never be aware that it is not using the standard Amiga serial port.

Should any internal error occur, such as not being able to open the requester due to a memory shortage, the SDP is programmed to default to using the Amiga's standard serial port.

Note that some programs open the serial device several times (perhaps once for reading and once, separately, for writing). In that case, the SDP will ask you where to direct the serial activity each time. Simply answer the SDP's requester the same way each time.

Note that the SDP will timeout after fifteen seconds. After a timeout, it will default to using the Amiga's standard serial port.

4.4.3 Not Using the SDP

Should you elect not to use the services of the SDP, you may still use any program written to use `serial.device` on the DSB. This can be accomplished by replacing all internally occurring strings having the value of "serial.device" with the string "siosbx.device" in the executable version of each program.

This can be done using any well featured text editor (like ASDG's CygnusEd Professional) or any binary file zapper (like John Hodgson's NewZap).

Note that this will allow you to use only DSB port 0, as the unit number to be opened is encoded in the actual instructions of the program. Changing the unit number (without having the source code to the program) is not straightforward. This is one of the chief advantages of using the Serial DisPatcher over the file zapping method (i.e., any DSB port can be used).

When using a binary file capable editor (like CygnusEd Professional) be careful not to alter the size of the file in any way or to make changes to any bytes other than the device name. Doing so will surely cause trouble.

Chapter 5

Troubleshooting

5.1 Technical Support

ASDG provides technical support via telephone at (608) 273-6585 (Monday through Friday, 10 A.M. to 5 P.M. Central Time) and upon Portal (**go asdg**), BIX (**join asdg**), and CompuServe (**go amigav**, then section 2). ASDG representatives occasionally stop in on GEnie and Usenet as well.

Additionally, you may send us your questions or comments in a letter to our business address:

ASDG, Incorporated
925 Stewart Street
Madison, WI 53713
USA

5.2 Service and Repair Information

Service and repair assistance can be obtained from ASDG, Incorporated by calling the number listed in Section 5.1 (Technical Support).

Always contact ASDG before returning a product for service.

Please have the following information available when you call:

1. Product name, serial number, and revision number.
2. Your shipping and billing address.
3. Your contact name and telephone number.

Ship products back in the same container they came in, if at all possible. If the original container is not available, take the following precautions:

1. Place boards in anti-static bags.
2. Allow room for padding material.
3. Send the product, a description of the problem, and the information from your phone conversation with the ASDG service staff to the address listed in Section 5.1 (Technical Support). Make it attention to "Service Department".

Chapter 6

Extras

6.1 Support of Higher Baud Rates

In addition to the baud rates listed in Section 4.3.2, the DSB hardware can also support a rate of 115200, although not through our software. You can use this rate by setting it through a terminal program. Our software does not directly support it.

6.2 Other Software on the Distribution Disk

As a service to you, we have placed several public domain and/or shareware communications packages on the DSB distribution disk.

ASDG, Incorporated cannot provide support assistance for these programs, which are provided as is. Any questions or suggestions should be directed to the persons specifically mentioned with each program, **not to ASDG**.

We would encourage you to send the shareware authors a donation if you find their programs useful.

As the packages included on the DSB disk are subject to change, please double click on the **On This Disk** icon in the root directory of the distribution disk for information on the packages that you have received.

6.3 Using High Speed PostScript Printers

6.3.1 Before You Start

This document describes how to configure a PostScript printer to run at higher than normal speeds when used in conjunction with an ASDG serial communications product. Applicable products include the Dual Serial Board (DSB, in two varieties) and the SBX-Serial/2 (in two varieties).

These products all use the same serial controller chip. The varieties mentioned above allude to the two operational speeds (of serial controllers) ASDG has used in manufacturing their serial products.

The first production run of the Dual Serial Board used 6MHz serial controllers. All subsequent production runs of the DSB used 8MHz serial controllers.

On the SBX-Serial/2 module, 6MHz serial controllers are used, except for special orders for the 8MHz part.

You'll need to identify which serial controller you have. You can do this by looking at the board or module and identifying the large chip marked "Zilog". On this chip, there will be a long string of digits beginning with "Z08530". The end of this string are two digits specifying the chip's speed.

If these two digits are "08", you have an 8MHz part and can run your PostScript printer at up to 57,600 baud. If these two digits are "06", you have a 6MHz part, and you can only run your printer at up to 38,400 baud.

The large chip can be found directly on the printed circuit board if you have a Dual Serial board or on the SBX-Serial/2 module if you are using a Twin-X with that module.

For use with the Twin-X and the SBX-Serial/2, make sure your switch settings are set properly for the serial controller you have. For a 6MHz serial controller, the switch settings should be all off (0). For an 8MHz serial controller, the switch settings should all be off (0) except for switch 3 which should be on (1).

Using Twin-X-Info (again, just for Twin-X users), a 6MHz board should generate a product i.d. of 512. An 8MHz board should generate a product i.d. of 516.

Dual Serial Board owners do not need to set any switches, but do need to know which speed serial controller they have in order to get the best results.

If you cannot identify which serial controller you have, do not proceed past this point.

6.3.2 Setting the Printer Speed

First, enter a terminal program and set the baud rate to the printer's current speed setting (usually 9600 baud). Note that you'll probably want to use the program "Gonzales" (included on the ASDG serial software distribution disk) since this program will allow you to set 57,600 and 38,400 baud.

Note that the Gonzales distributed on the ASDG distribution disk is programmed to open the standard serial.device. If you are already connected to your printer over an ASDG provided serial port, you will need to have installed the Serial DisPatcher to allow Gonzales to be switched to an ASDG provided serial port.

If you do not have a terminal program which can set the speed you desire to run that printer at, you cannot complete this change and should not proceed.

Type "`executive`" (without the quotation marks) followed by a carriage return.

If you're in full duplex mode, you won't see what you type, but that's ok. The printer should respond with a header describing the version of PostScript in your printer, and give you a "PS>" prompt. Whatever you type will now be echoed and executed.

Confirm that your printer is equipped with version 38 or later of PostScript. If you do not have version 38 or later, then this procedure will not work.

Type the line "`serverdict begin 0 exitserver`" followed by a carriage return. If this is successful, the printer should respond with a message indicating that the permanent state may be changed. The '0' in the above line is a password which allows you access to the permanent settings area of the printer.

If the password has been previously changed, the above will not work, and you should replace the 0 with the current password (a number).

Note that the permanent settings area of your printer is maintained in an EEPROM (inside the printer). EEPROMs cannot be changed an unlimited number of times. While your EEPROM should be good for approximately 10,000 changes, be aware that changing the EEPROMs should be kept to a minimum.

Retype the "`executive`" followed by a carriage return. You probably won't see this echoed but that's ok.

Enter "`statusdict begin`" (without the quotes, of course) followed by a carriage return. This gives you access to the command you'll be using to set the new baud rate.

If you have an 8MHz serial controller enter the following line:

```
25 57600 0 setsccbatch <carriage return>
```

If you have a 6MHz serial controller enter the following line:

```
25 38400 0 setsccbatch <carriage return>
```

The new baud rate should take effect immediately. You are also dumped out of executive mode. To confirm that your printer is set to the new baud rate, change your Amiga's baud rate to match the new rate and type the word `executive` again, followed by a carriage return.

You may receive an error message indicating that the value you entered in the setsccbatch command is out of range. This message occurs if your particular printer cannot support the higher baud rate. Type the line:

```
25 19200 0 setsccbatch <carriage return>
```

This will set your printer to 19200 baud, which most printers can run. Check your printer manual, or call the manufacturer, to see if this is the highest baud rate the printer will support. If it is, make the necessary adjustments to the port configurations described below.

6.3.3 Configuring a High Speed Port

Your PostScript printer is now all set to be used at the new baud rate. At this point we can now configure the Amiga's software to provide access to the higher speed printer.

This is accomplished by defining a MountList entry which will associate a logical name (for example, `lino:`) with a given serial port at a given speed.

Here's a sample MountList entry to use a PostScript printer at 57600 baud on expansion serial port 0, using XON/XOFF flow control:

```
lino:   Handler = L:SERX-Handler
        Startup = 0x10000A00
        Stacksize = 2000
        Priority = 5
        GlobVec = -1
#
```

If you have a 6MHz serial controller you will only be able to communicate with your printer at 38,400 baud.

Here's a MountList entry to accomplish that, using expansion serial port 0.

```
lino:   Handler = L:SERX-Handler
        Startup = 0x10000900
        Stacksize = 2000
        Priority = 5
        GlobVec = -1
#
```

Note that the last two digits of Startup specify the port number. If you had a 6MHz serial controller and wished to run your PostScript printer on port 3 you might specify:

```
lino:   Handler = L:SERX-Handler
        Startup = 0x10000903
        Stacksize = 2000
        Piiority = 5
        GlobVec = -1
#
```

In these example MountList entries, the name lino: is used only for example purposes. You can name the high speed port almost anything you wish.

6.3.4 Using the High Speed Port

You must place a mount of the high speed port you configured in your MountList into your startup-sequence. For the above examples, you might add `mount lino:`.

Now that the high speed port is configured and mounted, you may use it anywhere you would have used `SER:`. For example, in Professional Page, instead of having the PostScript output go to `SER:`, specify `lino:` (if `lino:` is what you named the port).

You should now be ready to use your PostScript printer at either 4 or 6 times the rate at which you had previously been using it. Note that the change described in this document will increase the speed of only the data transfer portion of the PostScript printing process. The time taken by the printer to internally digest the information you have sent it will remain the same.

During high speed transfers to the printer you may notice some jitter on your Amiga display if your current front screen is interlaced. This is normal and unavoidable. What is happening is that occasionally the interrupts caused by the high speed data transfer may cause a vertical blank interrupt to be delayed long enough to pass beyond the currently displayed video frame.

This is unavoidable since the only solution to this would be to interrupt the processor at a lower priority than the vertical blank. If this course were chosen, the highest reliable throughput through the expansion serial device would be limited to approximately 9600 baud.

To be phrased another way, the ASDG serial expansion board could either interrupt the Amiga at a priority higher or lower than the vertical blank. If higher, you will occasionally see a slight jitter in interlaced screens during high speed transfers. If lower, you'd constantly lose serial data. We think we made the right choice.

4.4.4 Removing the SDP

To remove the SDP, execute the following commands from the CLI or Shell prompt (remember to press the RETURN key after each line):

```
cd DEVS:
dir
```

You should see a file named oldser.device or cbmser.device. If you do not, contact Technical Support for further help. Once you have located the file, type the following:

```
delete serial.device
rename oldser.device serial.device
```

or use cbmser.device in place of oldser.device if that is the name of the file.

6.4 Using the DSB with the Personal TBC II

To use Digital Processing Systems, Inc.'s DPS Personal TBC II with the DSB, follow these steps:

1. On the TBC II board, set DIP switch #2 to the OFF position. This will set the serial port to run at 9600 baud.
2. From the Workbench screen, change the TBC II's icon tool types to:

   ```
   SERIAL_DEVICE = siosbx.device
   BAUD = 9600
   PORT_NUMBER = 0
   ```

   Note that PORT_NUMBER could also be equal to 1, for unit 1 on the DSB.

You will need to purchase a cable to connect the two board together. This can be found at most electronic parts stores or computer dealers. The cable required is a standard IBM-AT Serial Port to RS-232 cable.