Modem to Terminal or Host: Software Flow Control

Host Side			
DB-9	**DB-25**	**Modem**	**Signal**
3	2	2	TxD
2	3	3	RxD
5	7	7	gnd
1	8	8	DCD
4	20	20	DTR

Host to Host: Software Flow Control

Host Side			
DB-9	**DB-25**	**Switch**	**Signal**
3	2	3	TxD
2	3	2	RxD
5	7	7	gnd
1	8	20	DCD/DTR
4	20	8	DTR/DCD

Host to Host: Hardware Flow Control

Host Side			
DB-9	**DB-25**	**Switch**	**Signal**
3	2	3	TxD
2	3	2	RxD
7	4	5	RTS
8	5	4	CTS
5	7	7	gnd
1	8	20	DCD/DTR
4	20	8	DTR/DCD

If the hosts do not emulate the same kind of RS-232 device - one is DCE and the other is DTE - then the TxD and RxD cross-wiring must be reversed, and the DTR/DCD cross-wiring must also be straightened out. In this case, the connection boils down to a straight-thru cables.

Appendix B
Cabling Schematics

The tables in this section show the pin-to-pin connections necessary to create a number of typical cables.

Software flow control cables assume the DTE/DCE pair will use XON/XOFF for flow control. The cables designed for hardware flow control route the RTS/CTS signals between devices.

Cables listed for "Host to Host" connections are used for wiring port selectors, such as Micom Turrets or Gandalf switches, to Sun serial line inputs. These cables can also be used for hardwired UUCP connections.

Terminal/Printer to Host: Software Flow Control

Host Side			
DB-9	**DB-25**	**Device**	**Signal**
3	2	3	TxD
2	3	2	RxD
5	7	7	gnd
4	8	20	DCD/DTR

Jumper pins 6-8 on the host side when using an ALM-1 multiplexor. It is often necessary to buss together pins 5, 6, 8 and 20 on the printer or terminal side to pull the inbound control lines high. Noise or floating inputs can cause intermittent failures if the inputs are not properly driven according to the device specifications.

Terminal/Printer to Host: Hardware Flow Control

Host Side			
DB-9	**DB-25**	**Device**	**Signal**
3	2	3	TxD
2	3	2	RxD
7	4	5	RTS
8	5	4	CTS
5	7	7	gnd
4	8	20	DCD/DTR

Jumper pins 6-8 on the host side when using an ALM-1 multiplexor. As above, device-side jumpers may be necessary to pull inputs high.

Appendix A
Sun System Connector Table

System Type	# Ports	Connector Type	Standard
Sun 3/50	2	DB-25S	RS-423
Sun 3/60	2	DB-25S	RS-423
Sun 3/80	1	DB-25S	both
	1	DB-9S	both
Sun 3/1x0	2	DB-25S	RS-423
Sun 3/2x0	2	DB-25S	RS-423
Sun 4/60	2	DB-25S	RS-423
Sun 4/1x0	2	DB-25S	RS-423
Sun 4/2x0	2	DB-25S	RS-423
Sun 4/3x0	1	DB-25S	both
	3	DB-9S	both
Sun 386i	1	DS-25P	RS-423
ALM-1	16	DB-25S	RS-232
ALM-2	4	DB-25S	RS-232, RTS/CTS
	12	DB-25S	RS-232, no RTS/CTS

[1] All DB-25S connectors are female gender and have clocking signals present necessary to run a synchronous serial line (for X.25, for example).

[2] The DB-25S connector on the 386i is a male connector

[3] Only the first 4 ports on the ALM-2 have RTS/CTS support; all others use software flow control only.

[4] The ALM-1 uses pin 6 for DCD instead of DSR, and a jumper must be installed for hardware carrier to be sensed properly.

5
Further Reading

[a] **The SunOS System Administration Guide.** In a docu-box near you. A chunk of this guide is based on this document. The UUCP section is particularly good, and explains why `USERFILE` is the way it is.

[b] **Nutshell Guide to UUCP** by Nutshell Books, O'Reilly and Associates, Sebastopol, CA. The definitive guide to UUCP and its care and feeding. For under $20 you can learn to decipher C. files like a trained professional.

[c] **UNIX System Administration Handbook**, by Evi Nemeth, Garth Snyder and Scott Seebass. These folk are, as they say in the Boston vernacular, wicked smart. Chapters 9 and 16 cover this material in depth. The entire book is great.

[d] **The Design and Implementation of the 4.3 BSD UNIX Operating System**, by Leffler, McKusick, Karels and Quaterman. Read it if you have a burning desire to know how the tty driver works. In general, this is a good book and a solid history of some Berkleyisms that wandered into SunOS.

[e] **EIA RS-232 Standard**. Only for the die-hard or truly lonely, but available from:

Electronics Industries Association
2001 I Street NW
Washington, DC 20006
202 457 4900

- You do not see a message of the day but also do not get an `Shere` message. The remote host may not have `uucico` as the shell for your UUCP account. The actual exchange of messages during startup is covered in extreme detail in the Nutshell Books on UUCP.
- You attempt to send mail to a UUCP site, but the mail is returned to you. This is typically caused by an incorrect `sendmail.cf` file. The `sendmail` configuration should either blindly pass off UUCP-style addresses to `uux`, or it should build a class of host names that are connected via UUCP from the `L.sys` file. If you are delivering mail on both a LAN/WAN and via UUCP and have name conflicts, you should include some disambiguation rules that define a class of machines for which UUCP is the default rather than the ethernet mailer.
- UUCP mail is returned to you with a `uux` failure indicated. Check the permissions and names of directories in `/usr/spool/uucp`.

4.4 UUCP with Trailblazer Modems

To use a Trailblazer with a Sun running SunOS 4.0.3 or earlier, you must first install the serial line patches which enable 19200 baud operation. You also need a copy of UUCP which has a Trailblazer dialer in it and will recognize the token `ACUTELEBIT` in the `L.sys` file. A modified version of `uucico` for the Sun-3 and Sun-4 is available for anonymous ftp on `uunet`.

Set up the `/etc/uucp/L-devices` file with information about the calling units created several pages ago. If you want to use the same modem for 1200 and 2400 baud connections, make two entries for it:

```
ACUHAYES cua2 cua2 2400
ACUHAYES cua2 cua2 1200
ACUHAYES cua1 cua1 2400
ACUHAYES cua1 cua1 1200
```

These devices usually correspond to those in `/etc/remote`, although the two files are independent. You can use a line exclusively for outbound UUCP but remove it from the list of availbe `tip` devices.

The last step is to set up the system file entry with a chat script for logging in to the remote system:

```
princeton Any ACUHAYES 2400 16095551212
              "" \d\r login:-EOT-login: Usunne sword: foo
```

The first entry is the remote system name, followed by the time to call. The modem type and baud rate follow (they are used as look up keys in `L-devices`), and the configuration information ends with the remote site telephone number. The remainder of the systems file entry is a chat script that includes the UUCP login and password.

Chat scripts are set up as pairs of strings. The first string is expected to be sent by the remote host, and the second string is sent by the calling system. All of our chat scripts use the null string as the first expected data so that we immediately send a delay and return, which wakes up almost all modems and provides a character for the autobauding circuit to consume.

4.3 Debugging Tips

The best way to learn about UUCP and debug a session is to invoke UUCP in debugging mode:

```
# /usr/lib/uucp/uucico -r1 -sprinceton -x9
```

You will see the exact interchange of strings in the chat script, and can tweak it based on observations of the login sequences.

There are many, many ways for UUCP to break:

- You are receiving files via UUCP, but no mail is being delivered - coupled with `XQT DENIED` messages in the UUCP `LOGFILE` - then the `USERFILE` is too restrictive or does not have enough lines. The `USERFILE` must have two or more lines.
- Connections are made reliably, but the remote system is producing an `/etc/motd` banner when `uucico` starts up. Verify (on the remote system) that `uucp`'s home directory has a `.hushlogin` file, which suppresses the message of the day.

4

UUCP

UUCP is a gnarly creature and a major source of administrative woes. There are a few distinctions to be made: SunOS uses a fairly vanilla UUCP, Berkeley's 4.x BSD uses a more robust UUCP, and AT&T sells the Basic Networking Utilities (BNU, more commonly referred to as Honey DanBer UUCP - capitalization is important, as the name is the concatenation of the three author's logins). HoneyDanBer will be shipped with SunOS 4.1. For a thorough discussion of this material, check out the references in the next chapter.

4.1 Incoming UUCP

First, rename your UUCP data spooling directory to include the name of your system:

```
# cd /usr/spool/uucp
# mv D.noname D.myname
```

where *myname* is the hostname of your system. If you aren't sure what your officially announced UUCP name is or think it might get truncated to 8 characters, try `uuname -l`.

Next, ensure that `/etc/uucp/USERFILE` has at least two lines. The `USERFILE` shipped with SunOS is adequate, but people change them to effect tighter security.

Create a login for each remote site that will call you. The most common convention is to use *Uremote* as a UUCP login, where *remote* is the name of the remote site. Give the account a password, and make sure the user id and group id are the same as that of the generic `uucp` login.

Put a bare-bones entry in the UUCP systems file `/etc/uucp/L.sys` to indicate that this site is a UUCP neighbor (*princeton* will be used an example remote site):

```
princeton None(Princeton University)
```

This entry allows *princeton* to dial in to your system, and will be seen by `uucp` and `uux` as a valid destination for outgoing UUCP traffic. The `None` keyword indicates that you never call *princeton*; they poll you. If you want to have many sites call you, but you don't want to bother (or pay for) dialing out, your `L.sys` file will remain amazingly simple. You can use the systems file to track remote site contact names and telephone numbers, since anything following `None` is ignored by UUCP.

4.2 Outgoing UUCP

After completing the configuration steps for incoming UUCP, there are a few additional pieces required to get outgoing UUCP working. First of all, contact your neighbors and get a UUCP login and password, also determine what flavor of modems they are using.

```
cua-1200:\
     :el=^D^U^C^S^Q^O@:at=hayes:ie=#$%:oe=^D:\
     :br#1200:tc=dialers:
tip0:\
     tc=UNIX-1200
UNIX-1200:\
     :el=^D^U^C^S^Q^O@:du:at=hayes:ie=#$%:oe=^D:\
     :br#1200:tc=dialers:
UNIX-2400:\
     :el=^D^U^C^S^Q^O@:du:at=hayes:ie=#$%:oe=^D:\
     :br#2400:tc=dialers:
dialers:\
     :dv=/dev/cua3,/dev/cua2,/dev/cua1:
```

The first entry defines the telephone number and baud rate for calling system *apple*; the next two entries allow users to talk directly to a modem at either 1200 or 2400. Each of these descriptions uses the generic dialer descriptions that follow. Be sure that the command set (`at` setting) is `hayes`. The `du` entry in the `UNIX` descriptions indicates that the remote is on a dial up line and causes `tip` to expect a phone number. The direct modem entries do not use `du`.

The `tip0` entry is used by `tip` when it is given a telephone number on the command line. In the example above, `tip 5551212` will dial at 1200 baud (change the `tc` field of the `tip0` entry to dial at 2400 baud).

The final entry specifies which dialers to use if none is specified when `tip` is invoked. Notice that `cua0` doesn't appear in the file - in this example, that line is reserved for incoming UUCP calls and its exclusion keeps users from tying it up. `tip` searches for a free device in the in which they order specified.

To dial out, there are a number of approaches:

call *apple*	`% tip apple`
talk to modem	`% tip cua-2400`
explicit phone #	`% tip 12015551212`

The modem switch settings outlined above provide the most UUCP compatibliity at the expense of interactive user friendliness. When talking directly to the modem, start by issuing the command

```
AT E1 V1
```

to turn on command echo verbose result codes (such as OK).

The error message `cannot synchronize with ventel` is caused by the incorrect dialer type in `/etc/remote`; make sure you have `hayes` in the modem descriptions. The equally obscure `cannot synchronize with hayes` indicates that the modem is not echoing return codes - check the switch settings.

In SunOS 4.0 and later, the `open()` of the tty device was moved into the `getty` process.

To turn on hardware carrier detect, change the value of `flags` in the kernel configuration line for the serial device, using a 0 for each port that is to use hardware carrier detect:

```
device zs0 at obio ? csr 0x20000 flags 1 priority 3
device mcp0 at vme32d32 ? csr 0x01000000 flags 0x10fff
          priority 4
```

The first example shows port B on the CPU board configured for hardware carrier, and the second example shows an ALM-2 with ports C-F set up for modems.

On a SparcStation 1, hardware carrier detect is controlled by the NVRAM, and can be changed using `eeprom(8)`:

```
# eeprom "ttya-ignore-cd=false"
```

The SparcStation 1 also provides control of the RTS and DTR signals through the NVRAM. Normally, these signals are asserted by the SparcStation 1, but they can be turned off with another invocation of `eeprom`:

```
# eeprom "ttya-rts-dtr-off=true"
```

3.4 Dial Out Configuration

As mentioned above, be sure to turn on hardware carrier detect if you want a modem to be bidirectional. The next step is to create a calling device. The formula for determining calling device minor numbers is to add 128 to the minor number of the tty device being used. Example: if `ttyd0` is device 44,12 (formerly ttyhc on an ALM-2) then the corresponding calling unit will be device 44,140. To create the devices, use `mknod`:

```
# cd /dev
# ls -l ttyd0
crw--w--w- 1 root 44, 12 Aug 17 20:36 ttyd0
# mknod cua0 c 44 140
# chmod 666 cua0
# ls -l cua0
crw-rw-rw- 1 root 44, 140 Aug 10 00:24 cua0
```

The `tip` dial-out utility uses `/etc/remote` to find a free calling device. You can get fairly elaborate in `/etc/remote`, including telephone numbers for remote sites; you may prefer to provide some common remote site information and entries for talking directly to the modems at either 1200 or 2400 baud:

```
apple:pn=12125551212:\
     :tc=UNIX-2400:
cua-2400:\
     :el=^D^U^C^S^Q^O@:at=hayes:ie=#$%:oe=^D:\
     :br#2400:tc=dialers:
```

To dial-in to a Microcom modem from another Microcom, set the switches as above, but change front panel switch 5 to "down" to force the modem to raise DCD at all times. Also, explicitly set the interface speed:

```
AT \N2  for 9600 baud
AT \N0  for 2400 baud
```

3.2 Sun Device Configuration

First, rename the tty device with the modem on it to `ttyd`*n*, where *n* starts at 0. For example, if you have modems on the first 3 ports of an ALM-2:

```
# cd /dev
# mv ttyh0 ttyd0
# mv ttyh1 ttyd1
# mv ttyh2 ttyd2
```

Change the `/etc/ttytab` file to use the proper getty definition for the dial-up line. If you have 2400 baud modems but will have users (or other systems) dialing in at 1200 and 2400 baud, be sure to use a "rotary" getty. You may want to remove the 300 baud entry from the getty cycle, however, some 1200 baud modems will not synchronize correctly with a getty that switches between 1200 and 2400 baud only. The 300 baud entry provides enough of a delay for the slower modem to properly frame characters.

Typical entries for dialup lines are:

`/etc/ttytab` excerpt:

```
ttyd0   "/usr/etc/getty Fast-Dial-2400"dialup on
ttyd1   "/usr/etc/getty Fast-Dial-2400"dialup on
```

`/etc/gettytab` excerpt:

```
D2400|Fast-Dial-2400:\
     :nx=D1200:tc=2400-baud:
3|D1200|Fast-Dial-1200:\
     :nx=D2400:fd@:tc=1200-baud:
```

3.3 Hardware Carrier Detect

You will need to turn on hardware carrier detect for the ports with modems on them so that the same line may be used for dial-in and dial-out. When hardware carrier detect is enabled, any call to `open()` on device will block until the modem supplies a signal on DCD.

In SunOS 3.5 and earler releases, `init` and `getty` don't handle hardware carrier in a clean fashion. `init` will fork a new process which opens the tty device speficied in the `/etc/ttys` file, and hand the open file descriptor to `getty` through an `exec`(). If hardware carrier detect is used, `init` 's child will block in the `open`() call until the modem asserts DCD. Because the `init` has not completed the `exec`(), `ps`(1) shows two (or more) copies of init running, which is typically cause for panic.

modems set to assert DCD, so any `getty(8)` process will believe the modem has picked up an incoming call. Tip to the modem using a "hardwire" entry in `/etc/remote` and set the parameters from the command line:

```
AT &F S51=254 S52=2 S54=3 S110=1 S111=30 S45=0
     S58=0 S92=0 Q6 &W <return>
```

&F	restores the factory defaults
S51=254	enables speed matching
S52=2	drops the connection when DTR to the modem is dropped (by the host), and performs a software reset when DTR is toggled.
S54=3	passes BREAK through the modem
S110=1	enables PEP mode packet compression
S111=1	enables UUCP "g" protocol handling
S45=0	disables remote access
S58=0	disables flow control, which is handled by the user-level processes
S92=0	sets up proper DTR/DCD handling
Q6	turns off result codes on incoming calls, preventing `getty` from fighting with the `RING` and `CONNECT` responses from the modem when a call is received.

When enabling a getty for the Trailblazer, be sure to create a 19200 baud entry in the `gettytab` file which cycles through the lower speeds as well.

Switch Settings: Microcom High-Speed Modem

Microcom makes a series of high-speed modems that support autobaud speed matching and hardware flow control (modem to modem). To use this modem for dial-in at both 9600 and 2400 baud, set the switches as follows:

Front Panel		Rear Panel	
Switch	**Setting**	**Switch**	**Setting**
1	up	1,2	up, up
2	down	3,4	down, up
3	up	5	up
4	up	6	up
5	up	7	down
6	up		
7	up		
8	down		

Software Switches

Setting	Function
AT \J1	enable autobaud
AT \N3	enable speed switching
AT \X1	pass through XON/XOFF

bundles two modems together on a single card. The cards plug into a rackmount backplane, with redundant power supplies. There are some LEDs and toggle switches on the front:

LED	Indicator
HS	High Speed, lit when modem is operating at 2400 baud
OH	Off Hook, telephone is "picked up"
RI	Ring Indicator. Call is incident to modem
MR	Modem Ready. What it is
TR	Data Terminal Ready. Lit when DTR is raised

Switch	Function
AL/DL	Loopback, Analog and Digital. Leave in center-off position
RDL/BSY	Busy-out. Normally left in center-off position, but flip to BSY to busy-out a line for testing

If a modem is pulled out of the chassis, the line is automatically busied out. The DIP switches are located near the back of the card, and should be set as follows:

Switch	Setting	Function
1	off	DTR normal
2	on	numeric result codes
3	on	enable result codes
4	off	enable command echo
5	off	answer on first ring
6	off	assert DCD when carrier is sensed
7	on	use AT command set
8	on	retain connection after +++
9	off	originate at 2400 baud
10	off	hold CTS high

If these modems are wired to a hunt group, it is possible to debug specific lines in the hunt group by busying-out the well-behaved lines and forcing incoming calls to use the suspect circuits.

Switch Settings: Telebit Trailblazer

The Telebit Trailblazer modem is ideal for UUCP since it has hardware support for the UUCP "g" protocol. It also does packet compression with other Trailblazers, offsetting some of the overhead of UUCP. As another nice feature, there are no external switches with the exception of the "A/B" switch (used to load default values), the "T/D" switch (used for manual dialing) and a small reset switch located next to the RS-232 connector. Depressing the reset switch will reload the factory defaults.

When run at 19200 baud, with packet compression, effective throughput of 12000-13000 baud between like systems is the norm.

To set up the modem for dial-in/out and UUCP usage, connect it to the system and make sure there is not a `getty(8)` process running on that line. By default, Telebit ships their

3
Modems

3.1 Dial-In Configuration

This section covers the basic switch settings for a variety of modems, and the Sun kernel and device configurations that are required to use a modem for dial-in.

Switch Settings: US Robotics Courier

Here is a set of UUCP compatible, dial-in/out modem switch settings for a US Robotics Courier modem:

Switch	Setting	Function
1	up	causes modem to follow DTR
2	down	default is numeric result codes
3	down	default is to display result codes
4	down	default is to suppress command echo
5	up	answer on first ring
6	up	assert DCD when carrier is sensed
7	up	single phone connection
8	down	enable AT command set
9	down	retain connection after +++
10	up	unused

Using the settings above, commands you type to the modem will return numeric codes, such as "0" for a successful command. The carrier sense settings will cooperate with the Sun's hardware carrier detect. If your modem is not answering incoming calls, be sure that switch 5 is up and that the AA light is on. The quad switch is left in the "normal" position unless you have the wrong cable type and need to un-cross pins 2 and 3.

If you have a US Robotics 2400E Error Correcting modem and will be communicating with modems without the error correction protocol, turn off error correction so that the two devices can properly negotiate a connection. To suppress error correction, `tip` to the modem and set register M to 0:

```
AT &M 0 &W
```

If you are connecting the modem to a terminal for dial-in, change DIP switch 6 to "down", which will force DCD high at all times. This allows you to see the characters typed on the terminal when talking directly to the modem.

Switch Settings: US Robotics Rack Mount

US Robotics also ships its 2400 baud Courier modems in a rackmount configuration, which

modem or worse. Similarly, not all RS-232 connectors map the telephone line conductors to signal pins the same way; some use red and green for pins 2 and 3 while other mappings use the yellow/black pair for the data lines. The moral is to check the point-to-point wiring after assembling a cable using telephone components.

2.6 Printers

Printers behave like terminals. They are wired using null modem cables, although some printers expect a signal on DSR or DCD. The classic LaserWriter problem is to wire the printer, and watch the "processing" LED begin to flicker while nothing is printing or queued to be printed. This indicates that the printer is wired with the wrong kind of cable.

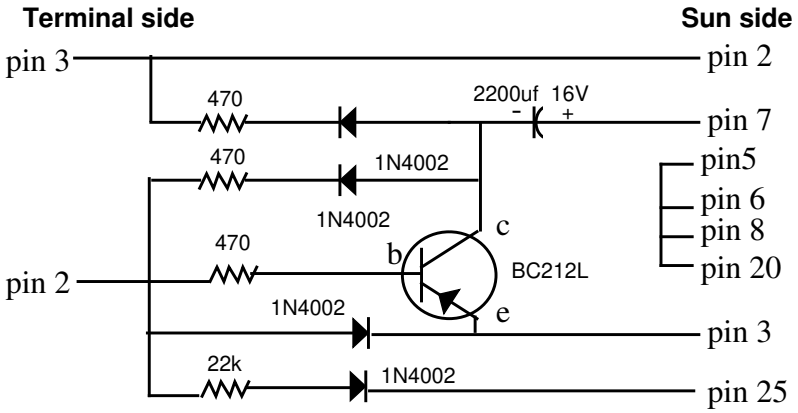
If you really want to have fun and impress your parents, set up a `tip` connection to a LaserWriter (use the "hardwire" entry in `/etc/remote` as an template). `tip` to the printer, type `executive` and hit return. You won't see this input echoed, but you will be put into the LaserWriter PostScript interpreter. You can now talk to the LaserWriter, and write PostScript code on the fly. This is also a simple way to verify a printer connection, since you can `tip` to the Laserwriter and feed it:

```
newpath 10 setlinewidth  100 100 moveto
100 0 rlineto 0 100 rlineto closepath
stroke showpage
```

If you get a triangle printed on the page, then your hardware and cabling are fine and any printing problems are restricted to `lpd`**,** `/etc/printcap` and friends.

2.7 Console Wiring

If you are using a terminal as a console, and remove the RS-232 connector, the Sun system will halt with an Abort when the connector is replaced., just as if you had hit BREAK. The following circuit suppresses the abort and allows connectors to be removed and replaced without problem:



For completeness, it is worth mentioning that you run the risk of damaging any piece of equipment, particularly keyboards and terminals, if you unplug it with power applied.

requires a carrier (the host's DTR signal routed into the terminal's DCD). Remember, both host and terminal are DTE, so they assert DTR and expect (sometimes) a signal on DCD. If your hardware is this complex, you have two options:

- Kludge DTR/DCD by bussing pins on the RS-232 connectors
- Buy Wyse terminals, which can be driven by 3-wire RJ-11 lines.

Appendix B contains cable schematics for the most common interconnect requirements.

Many devices do not handle "floating" inputs well. Printers such as the LaserWriter and older Imagen models are the classic examples of devices that require jumpers at the connector to pull unused inputs high. If the device is expecting a signal on the pin (even if the signal, like RTS, may not ever be used), then the input should not be left floating.

The converse is also true: running too many conductors to a device may confuse it, if the device doesn't expect to deal with RTS/CTS but still sees signals on those conductors, it may behave in random ways. Don't rely on pre-packaged "null modem" converters, since they may buss or cross-wire pins 6, 8 or 20. Two devices driving the same conductor can also produce unexpected results.

The vast majority of serial line problems are caused by improperly wired cables - using a null modem where a straight through would have been appropriate. To diagnose these headaches, you really need an RS-232 breakout box, which sits in between a cable and a piece of equipment. The box has a set of LEDs which show the state of all interesting RS-232 lines, and the better boxes have a patch wires to allow you to switch pins 2 and 3, or wire DCD or DTR to a high or low level. A favorite is available from JDR Microdevices for about $35 (see their ad in *Radio Electronics*).

2.5 RJ-11 and RJ-45 Wiring

A number of vendors make RS-232 connectors that are outfitted with an RJ-11 or RJ-45 telephone jack, so that standard 4- or 8-wire telephone cable can be used for interconnections. With an RJ-11 crimping tool, cables can be cut to the desired length using parts from your local Radio Shack - no soldering involved.

Entire buldings can be wired using two sets of telephone lines: one for voice and one for data. Easch office gets a duplex wall jack, and terminals can be wired into the wall. Once the buliding wiring is done, there is never a need to run another RS-232 cable from the machine room to a user's office or cube.

If such a scheme is used, then two sets of punch-down blocks make the administration of the serial line snake pit less treacherous. An "octopus" of RS-232 connectors runs from one punch-down block to the serial line controller on the host(s), the other punch-down block contains the endpoints of lines run into individual offices. Circuits and completed by wiring the feed side to the office side; when a new office is occupied the terminal can be brought on-line by punching down a new connection.

One side note: not all 4-wire telephone cables run the red and green conductors straight through. In some cases, the lines are reversed, turning a straight-through cable into a null

cable, depending upon the application and the equipment used. However, in the majority of all applications, the shield should only be connected to **one** end of the cable. **Wiring both ends of the cable shield can create ground loops**. On DB-25 connectors, the shield ground is connected to pin 1; on DB-9 connectors it should be attached to the outside of the connector shell.

2.3 Pin Assignments

The following table presents generic pin-outs for DB-25 and DB-9 connectors. In the discussions that follow, the DB-25 conventions will be used.

Pin # DB-9	Pin # DB-25	Signal Description
NC	1	Protective Ground
3	2	Transmit Data (TxD)
2	3	Receive Data (RxD)
7	4	Request to Send (RTS)
8	5	Clear to Send (CTS)
6	6	Data Set Ready (DSR)
5	7	Signal ground (gnd)
1	8	Data Carrier Detect (DCD)
4	20	Data Terminal Ready (DTR)

System information, including the specific number of connectors of each type, the interface standards used, and the availability of hardware flow control is presented in Appendix A.

2.4 Cabling Primer

If you are wiring any two DTE devices directly - a terminal to a serial port, for example - you need a null modem cable. This is an RS-232 cable with pins 2 and 3 cross wired. It earns its name by emulating the effect of two modems, where the modems would do the cross-wiring of transmitted and received data signals. In a modem connection, the modem originating the call sends data at one frequency and receives at another; the answering modem reverses frequencies. The data lines are crossed using modulated signals rather than physical connections.

Most terminals can be wired to hosts with just three wires: a common ground, TxD and RxD. The terminal ignores DSR, and the serial line controller ignores DCD (or supplies it in software). A 3-wire connection is easily run using RJ-11 (telephone) wiring and standard telephone punch blocks.

To wire a modem (DCE) to a terminal or to a host (DTE), you use a straight-thru serial cable: a minimum of pins 2, 3, 7, 8, and 20 are wired to their corresponding pins on each connector. To directly connect any two pieces of DTE, such as a host and a terminal, a null modem cable is used.

Some null modem cables also cross-wire pins 8 and 20 (DCD and DTR). These cables must be used in cases where the serial line controller requires a hardware carrier (which is produced by routing the terminal's DTR signal into the host's DCD input) and the terminal

2
Cabling

2.1 EIA Standards

Sun workstations use either the RS-232C or RS-423A recommended standards. Cabling specifications and signal level thresholds are available in the formal serial line interface specifications, available from the EIA (see Chapter 5).

RS-232 and RS-423 use the following voltage thresholds:

	Open Circuit Voltages		
Standard	**Minimum**	**Maximum**	**Without Damage**
RS-232C	3.0 V	25.0 V	25.0 V
RS-423A	4.0 V	6.0 V	12.0 V

There is a common operating area between 4.0 and 6.0 volts, allowing RS-232 and RS-423 to interoperate. When mixing RS-232 and RS-423 circuits, performance is limited to that of RS-232C.

RS-232C specifies that the risetime through the 3.0 volt transition not exceed 4% of the signal element duration. RS-423A requires slower rise times to reduce cross talk for operation over longer distances: signals with a higher effective AC frequency are more likely to suffer from inductive coupling when run in parallel over long distances.

The use of short cables is recommended. Typically, 50 feet is the maximum cable run for 9600 baud operation that is "within spec", but longer runs are certainly the norm. The EIA standard requires that the circuit capacitance, measured at the interface point and including the signal terminator, not exceed 2500 picofarads. Running serial line cables over standard telephone wiring extends the maximum run length up to several hundred feet.

2.2 Shield Ground (Drain)

Some interface applications require use of shielded interconnecting cables for RFI (radio Frequency Interference) suppresion or other environmental concerns. If your radio emits bursts of static each time the printer starts a new job, you are actually hearing the RFI. More severe cases involve crosstalk on analog telephone lines or inductive interference from nearby power conduits. As a general rule of thumb, don’t tie-wrap unshielded RS-232 lines and power cords together, as the line voltage (AC) noise that leaks out of the unit power supply will be picked up by the serial line. To get an idea of just how bad AC line noise can be, watch what happens to your TV image when a hair dryer is plugged into the same circuit (your mileage may vary).

When used, the cable shield can be connected to frame ground at either or both ends of the

RTS and CTS provide a hardware handshaking mechanism. Under SunOS 4.0 and later, hardware handshaking is available on the CPU board serial ports and on the first 4 ports of an ALM-2. RTS/CTS flow control can be enabled using an argument to `stty`:

```
% stty crtscts
```

RTS and CTS control the flux of data with respect to the DTE - when it is busy, it will not assert CTS. There is no flow control from the DCE side of the connection, so no device can request that a terminal stop sending data.

Almost all devices use software handshaking - the familiar XON/XOFF flow control. When the relatively slow DTE device cannot accept more data, it sends an XON character (control-S, although it can be set in the tty driver); when it is ready to resume data consumption it sends XOFF (control-Q).

DSR is asserted by the communications device to indicate that it is ready; it is sometimes bussed to DCD when a hardware carrier is required but none is provided due to cable conductor or hardware limitations.

DTR and DCD provide modem control. DTR is asserted by a terminal to indicate that it is ready - usually "ready" is equivalent to "powered on". It is a signal from the DTE to the modem to either connect or disconnect the telephone connection. For example, `tip` will drive DTR low when it exits so that the phone line is hung up properly. Similarly, DCD allows the modem to inform the DTE that a connection has been made or broken. If the telephone connection is dropped, a modem will drive DCD low so that the host can clean up any `rlogin` or `tip` processes.

1
RS-232 Basics

1.1 DTE and DCE

A device talking on a serial line can be one of two flavors: either Data Communications Equipment (DCE) or Data Terminal Equipment (DTE).

In general, DTE includes terminals, printers and machines with serial line controllers ; DCE is a modem or something that looks like one. This is counter-intuitive, since you would expect the host machine to be "communications equipment", however, "terminal" implies an endpoint of the data communications hardware chain. The actual hardware implementation varies by vendor, but in the Sun RS-232 world, hosts behave like DTE. Terminal servers, such as the Annex or Bridge CS series, vary in their interfaces.

In most cases, a complete RS-232 circuit is two DTE devices joined by two DCE devices or something that emulates a pair of DCE devices, like a null modem cable. A dial-in circuit is two pairs of DTE/DCE devices: terminal (DTE) connected to a remote modem (DCE), joined by a telephone line to another modem (DCE), which is cabled to the host (acting as DTE to complete the circuit).

Again, this is more of a guideline than a rule, since hosts may operate as DCE. It is imperative that you know whether your devices are DTE or DCE, as this information will drive the cabling process. The Sun ALM-1 is a perfect example; you can get stung moving serial lines off of an ALM-1 onto an ALM-2 because the ALM-1 was configured as DCE (it is jumper-selectable).

It is also important to note that the gender of connectors is independent of whether a device is DCE or DTE. The use of a particular gender DB-25 connector on a terminal does not imply any particular behavior for the device.

1.2 Signal Definitions

RS-232 communication relies on only a few interesting signals:

TxD	transmit data	RxD	receive data
DCD	carrier detect	DTR	terminal ready
RTS	request to send	CTS	clear to send
DSR	data set ready	gnd	signal ground

The definition of TxD and RxD is with regard to DTE: data is transmitted from terminal (or host) to modem along TxD, and from modem to terminal along the RxD conductor.

Introduction

This document covers the basics of asynchronous serial line communications. It is intended for use by system administrators, systems engineers and people at trade shows faced with seemingly endless combinations of cable pinouts. Some basic knowledge of DC signal terminology is assumed, although imprecisely used terms will be clearly defined for the scope of this guide.

Chapter 1 Introduces the fundamentals of RS-232 theory and terminology.

Chapter 2 Everything you need to know about cables and connectors before plugging in the soldering iron.

Chapter 3 Configuration and usage of modems for dial-in and dial-out service.

Chapter 4 Simple UUCP installation

Chapter 5 Where to go next

The appendices contain tables of configuration and cabling information specific to Sun systems.

This is not intended to be a complete guide, but merely an introduction to the most common problems and a collection of "cookbook" recipies for their resolution. Synchronous serial lines are not included in this revision but are "on the heap" for the next pass.

Sincere thanks are due to Ed Fleschute from Sun Microsystems in Milpitas, CA. Ed provided much of the electrical detail on RS-232 and all of the cable schematics in Appendix B.

Hal L. Stern
Sun Microsystems
One Cranberry Hill
Lexington, MA 02173
halstern@sun.com

Legalese: (c) 1989, 1990 Hal L. Stern. Anyone may reproduce or retransmit this document, in whole or in part, provided that the copy or republication shows the source as Hal L. Stern, and that any other use of this material reference this manual and the fact that the material is copyright by Hal L. Stern and is used by permission. Sun Microsystems assumes no responsibility or liability arising out of the use of this document.

Unix is a trademark of AT&T Technolgies, Inc. Sun Microsystems, SunOS, and Sun Workstation are trademarks of Sun Microsystems, Inc.