HOWTO.5

Adding one or more serial cards to a Linux Box and avoiding IRQ and memory address conflicts.

The information below can be used as general guidelines for the installation of any serial card; however the suggested recommendations were tested with the SIIG HighSpeed/Hi-IRQ I/O, Model IO1812 and SIIG CyberPro QuadI/O, Model IO1825 with RedHat Linux 5.2 installed.

If additional high-speed serial ports are required, one or more serial ports have to be added. It is important to set the jumpers to interrupts(IRQs) which will not conflict with other computer resources using the same IRQs. To use more than two serial devices requires reassigning one or more interrupts. This must be done on the serial card itself as well as through software commands. The standard interrupt requests seen by Linux for COM1 and COM2 are 3 and 4. Therefore, it is necessary to select other interrupts for additional ports. Interrupts already in use can be seen by looking at the file `/proc/interrupts`. A good choice is to reassign an interrupt form the secondary parallel port (IRQ 5) which is rarely used. If the main (primary) parallel port (IRQ 7) is not used at all , it is even possible to assign its IRQ to a 4th serial port. One caveat: RedHat Linux does not necessarily recognize the new IRQ hardware settings at boot time, and will typically assign the same IRQs (3 and 4 in succession) to additional COM ports detected during boot. Therefore, it is necessary to force the kernel to recognize the hardware settings by creating a file called `rc.serial` in /`etc/rc.d` with the following lines

```
#!/bin/csh
/bin/setserial /dev/ttyS# irq #
```

where `ttyS#` is the number of the new comm port and `irq #` is the IRQ number matching that of the dip switch set on the serial card for that port. Another `setserial` command is required for a 4th port.

Make sure to give execute permission to `rc.serial` by issuing the command
`chmod a+x /etc/rc.d/rc.serial`
For each reboot , the file `/etc/rc.d/rc.sysinit` will look for `rc.serial` and implement the new IRQ settings.

If more than 4 ports are required, additional work is required. As mentioned above, the total number of ports that can be used is limited by the number of interrupts and port I/O addresses available. This is not a Linux limitation but a limitation of the PC bus. For a typical computer there are generally not enough IRQs available to install more than 4 independent ports. So, additional serial ports will have to share IRQs. This is not a problem as long as the devices attached to the ports sharing the same IRQs are used at different times. However, even devices that share IRQs *must* have different I/O port addresses to avoid confusing the kernel as to which port the signal should go to. All IRQs do is to interrupt the kernel when the device they are assigned to needs attention. Be very careful of devices that occasionally query the port when you don't expect it. For example, dial-out modems should not be a problem; however, dial-in modems could potentially create an IRQ conflict condition. Another limitation is that the RedHat distribution of Linux creates only 4 devices (character files) `/dev/ttyS0` to `/dev/ttyS3` during its installation. Additional devices must be created with the `MAKEDEV` command. For each additional port beyond 4, issue the command:

```
./MAKEDEV ttyS#
```

from the /`dev` directory. These files should have read/write permission for everyone.

Because of the above limitation, Linux will not detect the additional serial ports at boot time. Therefore, the hardware setting must be explicitly specified during boot. These settings concern port I/O address, speed, and IRQ settings. They should be added to the file `rc.serial`. Below are hardware settings that will

work for 6 serial ports with shared IRQs of 5 and 7 (Four independent serial ports if only 4 serial devices are used, 2 independent ports and 4 with shared resources for 6 serial devices).

```
#!/bin/csh
/bin/setserial /dev/ttyS2 irq 5
/bin/setserial /dev/ttyS3 irq 7
/bin/setserial /dev/ttyS4 port 0x2f0 irq 5 uart 16550A ^Fourport
/bin/setserial /dev/ttyS5 port 0x3e0 irq 7 uart 16550A ^Fourport
```

Troubleshooting tips:

The best way to test each port individually is to connect a modem and use the Linux communication program `cu` to try to connect to it and then list the register settings. If you get a message such as “device busy” or “access denied” when starting the communication program, make sure that permissions are set properly on the device file you are using for the connection (i.e. `/dev/ttyS#`) . From root, issue the command `chmod a+rw /dev/ttyS#`.

If the `cu` command is successful but no response is received from the modem (i.e. you cannot list the register settings), there is a port addressing or interrupt problem. This is not necessarily a conflict. Some interrupts just do not seem to work well with Linux. An example is IRQ11 even though it may be listed as available. Also, make sure you do not use an interrupt already required by another device such as a hard disk controller (typically 14 for the hard disk, and 15 for the zip drive), or the floppy drive. Interrupts 2 or 9 will not work either since 2 is required for cascading IRQs from 9 through 15. Check the file `/proc/interrupts` once more. To see I/O port addresses in use after Linux boots, look at `/proc/ioports` .