ADANAC SOFTWARE

Digiboard Serial Ports on Linux Servers

By Dirk Hart

Recently I have been working on a RedHat server installing all manner of stuff including serial ports. This customer has an 56K DDS line under contract for a few more years so we went with a Digi C/X controller to match what was in their SCO server. When I asked Digi tech support about a couple of things that weren't clear they suggested that I forget the the software and epca drivers that came with the controller and instead download and install their dgap drivers which comes with their familiar mpi software for configuring the board.
cartoon

So I did. Too bad the instructions on their support web site are incorrect. Not terribly incorrect though so I typed in rpmbuild --rebuild 40002347_A.srpm which failed since the file name really is 40002347_a..src.rpm. Well that didn't work either but for different reasons. It turns out that with RedHat Enterprise Server the kernel can't be autodetected, so you have to explicitly say what kernel you have so after a while I typed in

rpmbuild --rebuild --define DISTRO=REDHAT_ES_3 40002347_a.src.rpm

since I had RedHat Enterprise Server. This seemed to work fine, so in the next step I typed in:

cd /usr/src/redhat/RPMS/i386
rpm -ivv dgap-1.1-1.i386.rpm

and that seemed to work fine too. At this point I typed in

chkconfig --add dgap

so that the dgap drivers would always load at boot time. Next, I went ahead and ran the Digi configuration utility mpi. I did my best answering all the questions but as I had a 56K line I didn't choose the suggested defaults. After a couple of tries I ran the Digi Port Authority and found that the pod (concentrator) status was OK and that I could monitor a port on the other end, which was indeed far away.

Now, on SCO OpenServer, there is a handy utility called pcu for configuring the serial ports but there is no equivalent for Linux. Maybe there's just too many different ways of doing stuff on Linux platforms for a utility like pcu to cope with. In any case the accepted way of doing this seemed like a throwback to 15 years ago when we made our own inittab entrys. Since the first line is connected to a dumb terminal I added this line to /etc/inittab:

c01:2345:respawn:/sbin/agetty -L ttyc01 38400 ansi

and typed in init q at the shell prompt. Bingo, we had a login at the remote site.

I also installed an Equinox EPS2-MI 'serial hub' which means 'serial ports attached to a LAN'. This install went a lot smoother, but it has the same issue where you have to add entries to /etc/inittab manually. Since this widget is being used to run printers I edited /etc/inittab and added:

Q01e0:2345:respawn:/bin/bash -c "(stty 9600 -ixany -crtscts ixon; sleep 30000)< /dev/ttyQ01e0"

This clever entry conditions /dev/ttyQ01e0 to 9600 baud etc, and when 30000 seconds is up it just respawns again for 30000 seconds. Well, I was unable to make this work and for no obvious reason, until someone noticed there was an odd message on the console about some entry that was too long in /etc/inittab. Indeed. So I quickly changed the entry to read:

Qe0:2345:respawn:/bin/bash -c "(stty 9600 -ixany -crtscts ixon; sleep 30000)< /dev/ttyQ01e0"
difference is the label: Qe0 vs. Q01e0

and that worked fine. I expect that you could use the same sort of entry for printers on the Digi equipment as well.