

Re: Converting serial I/O to ethernet I/O

Source: <http://unix.derkeiler.com/Newsgroups/comp.os.vms/2007-11/msg00521.html>

- *From:* "John Wallace" <johnwallace4@xxxxxxxxxxxxxxxxxxx>
 - *Date:* Sat, 17 Nov 2007 17:26:50 -0000
-

"tadamsmar" <tadamsmar@xxxxxxxx> wrote in message
<news:96874356-8dd3-4000-81bd-fa43ca05128b@xx>

On Nov 16, 8:28 am, Bob Gezelter <gezelt...@xxxxxxxx> wrote:

On Nov 16, 8:04 am, tadamsmar <tadams...@xxxxxxxx> wrote:

I maintain a VMS real-time app. that has a serial interfaces to analyzers. The latest version of the analyzers support network communications and we want to convert from serial to network.

Any pointers on how to do it? I think it's a matter of making the right QIO calls, perhaps just revising or adding alternative QIO calls.

tadamsmar,

It may be as simple as that, or a more extensive re-write. It depends on the actual interface specifications of the analyzers.

In my career, I have seen both. I have seen devices just change from serial interfaces to say, a LAT or telnet scheme; and I have seen devices adopt completely different message formats, standards, and protocols.

In the abstract, it is not possible to answer this question from the information in your post. It would require careful reading of the documentation for both the old analyzers and the new analyzers, and a review of the source code of the application.

Re: Converting serial I/O to ethernet I/O

– Bob Gezelter, <http://www.rlgsc.com>

The manual says it uses the the same commands over RS-232 and Ethernet. Over Ethernet, it uses TCP/IP and port 9880.

Excellent. You have documentation, and have read it. That's always A Good Start.

In addition to the good advice already posted: IFF it really is as simple as the docs claim, and the command protocol really is relatively simple, you might well want to have a play with your favourite TELNET application (VAXman suggested the obvious VMS one, others are available), use it to Telnet to the analyser's port 9880, type in some commands by hand, and examine the resulting behaviour. This will give you an early idea of how the box really behaves, rather than how the documentation says it behaves. (You might want to say what kind of box it is, just in case someone round here has done this before).

You may want to give some thought to error conditions, their detection, and appropriate error recovery. With a classic serial port, there's not a huge amount to go wrong, and if something does break, it's usually obvious (to the application) reasonably quickly, partly because the VMS terminal driver conveniently supports timeouts. Introduce a network layer (and in particular a TCP/IP layer) and the possibilities for error are almost infinite, and in the wrong circumstances it can take a relatively long time for the application to detect that something's not right (JF already noted the absence of meaningful timeouts). If the application is expecting quick indication of success or failure of a command/response sequence, a simple one-for-one translation of QIOs with some changes to connection setup and teardown may be about to lose that. In some cases this may not matter. In other cases, it may require serious thought (you want the application, and anything that depends on it, to hang while the TCP/IP stack takes the time to detect that the analyser isn't actually responding?)

Would this by any chance be related the BASEstar application you mentioned here earlier in the year? My recollection (from a *long* time ago) is that BASEstar Classic comes with a thing called a "Generic DAS". You got its source code and it could be quite easily customised to talk to simple devices like printers, barcode readers, weighing machines, and other such simple devices, probably including analysers that talk Telnet. The Device Access Software (DAS) handles much of the connectivity and configuration housekeeping for you, leaving your application the simple (!) challenge of handling the actual data transfers. Your current app probably isn't using BASEstar, and changing it to use BASEstar may be a bigger change than you'd consider, but it never hurts to be aware of other options which may be relevant. Apologies in advance if this train of thought is irrelevant.

Good luck,

Re: Converting serial I/O to ethernet I/O

Re: Converting serial I/O to ethernet I/O

John Wallace

.