

Re: general purpose driver for Printer Port?

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.development.device.drivers/2008-06/msg00374.html>

- *From:* "anonym" <anonym@xxxxxxxxxx>
 - *Date:* Mon, 16 Jun 2008 20:08:01 GMT
-

Good points thanks!

If I have computer running this funky proposed "parallel port" driver, it would'nt need any support for conventional printer port operation anymore. I don't want to "replace" the windows printer port driver for sure! If windows tries to interfere, I could just delete window's own parport.sys driver, as a worst case.

USB is attractive and more modern way to go but, it's more expensive on hardware and I've yet to find anything that lets actual USB connected hardware, directly generate an interrupt?

Since you mention it though, I notice (on my computer at least) that USB shows up at I/O ports 1820-183F and using IRQ19. I imagine though, that different motherboards may use different I/O mapping.

Where could I find some info on what "hardware" actually resides in that I/O space and how to interact with it? Is it possible, that WITHOUT any PnP detected/loaded driver or device ID or enumeration, one could directly command a USB port to send/receive a packet? Is there a common chip-set used or something like that, where I could read up on the topic?

I want to stay low-level as possible, since this gives the highest flexibility and shortest latency! This is what makes the good old parallel port so attractive, in that it's just a data register for 8-bits of direct I/O.

~anonym

"David Craig" <drivers@xxxxxxxxxx> wrote in message
news:ukHnLW#zIHA.4848@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

I would consider using a filter on the standard parallel port driver since it is rather difficult to replace the Microsoft released version. Let the standard version handle the hardware and have the filter maintain a request for data it places into its buffers holding on to them until requested. You could grab the port exclusive and keep other programs from accessing it directly if you had a root enumerated driver of your own that your software opened. It would not be a filter in that case, but a pseudo parallel port driver that only your programs would know about.

I think that covers the two possible options I can think of and may give you what you need. Switching your hardware to use USB may be the best solution overall since it is a much faster buss and can handle PnP much easier. I think there is a parallel port enumerator to create a form of PnP, but it is legacy hardware and doesn't really support true hardware based PnP.

Re: general purpose driver for Printer Port?

"anonym" <anonym@xxxxxxxxxxx> wrote in message
[news:3tx5k.33446\\$gc5.3065@xxxxxxxxxxx](mailto:news:3tx5k.33446$gc5.3065@xxxxxxxxxxx)

Has anyone ever seen/heard of a general purpose driver for PC Parallel (ie. Printer) Port?

It would do some or all of the following:

1. Respond to interrupts (ACK line on pin 10) so that synchronously with the outside world, it would pass data/status register values to/from a memory buffer. Separate buffers for INPUT and OUTPUT of course.
2. In addition to memory buffers for register values, another set of similar buffers holds timestamp values (performance counter) captured along with each interrupt. Separate timestamp buffers for INPUT and OUTPUT of course.
3. Also on the wish-list is to provide some event notification when buffers were full, 1/2 full, empty; or alternately it could be a circular buffer, in which case events can be set-up to notify at regular buffer index step multiples.
4. The timestamp buffer values can either be directly synchronous to interrupt trigger; or they can be asynchronous, in which case the driver actual waits the time period specified by the buffer value, before it either outputs or inputs port data values. A "mode selection" makes this available.
5. Could these buffers be available to both kernel driver and user application transparently, so that program interaction approaches real-time?

This would be quite useful for doing direct hardware interfaces and/or control applications with a PC running windows, as it would bypass the usual requirement to purchase expensive DAQ hardware/software. For data stream say up to 1Mhz, simple acquisition/generation of both Time and Frequency domain signals become possible. Perhaps even some simple control loop processing too!

Is this too much to ask from a Device Driver? Any pit-falls or gotchas that could arise?

I would even settle for a Driver that captured INPUT only as long as it was an interrupt driven sequence of data and timestamp values dumped into a buffer.

If this wonderful general purpose Driver does not exist, does it have to be kilo \$\$\$ to contract someone to write such a driver for 32bit Windows XP ?

Does the "Device Driver" community post/share various examples anywhere? Maybe I can find something close to what I need and modify it myself.

Re: general purpose driver for Printer Port?

~anonym