

Re: OT;W98 popularity

Source: <http://sci.tech--archive.net/Archive/sci.electronics.design/2007-09/msg00284.html>

- *From:* Chris Jones <luginut808@xxxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 03 Sep 2007 00:29:54 +0100
-

Jonathan Kirwan wrote:

On Tue, 28 Aug 2007 11:36:29 -0700, "Joel Kolstad"
<JKolstad71HatesSpam@xxxxxxxxxx> wrote:

"Jonathan Kirwan" <jkirwan@xxxxxxxxxxxxxxxx> wrote in message
news:37p8d394keofp30uui746sg30nlfhmu5o1@xxxxxxxxxx

Still, have you LOOKED at the USB 2 manual? I have it
here, nicely
printed out, plus the HID stuff. Whew!

Not USB 2, but I did do a full-speed USB device mostly from scratch
(writing all the microcontroller routines to receive, send and parse data
packets ---
but I did have the manufacturer's example code to start with). It was
time-consuming and convinced me that, unless you really high-performance,
chips like those from FTDI are the way to go --- at least for anything
that's relatively low-volume.

There is just soooooo much more to worry about with USB. And a lot of
it is software. Let's say I want to develop my own 16 dig I/O and 4
channel analog DAC and 2 channel analog ADC board for some custom job.
In USB, I need to worry about which method I'm going to use with USB
(will it be HID or CDC class or whatnot) which means I have to first
KNOW something about it, which means reading A THICK BOOK!! Then, I
have to find out what comes standard with Windows. And there is NO
WAY it is going to work under a DOS system, where USB support is thin
at best. And that's before I get around to having to work out the
microcontroller and extra ICs I'll need and then the software on the
micro-side, if it includes a USB pinset, or yet another manual on the
FTDI chip, I suppose, if I use one. And then....

Well, ISA was dead trivial, by comparison. Solari's book was all I
needed. Everything else is trivial.

Re: OT;W98 popularity

I'm doing some USB now. But I definitely think ISA was a lot easier and NO software involved except for the PC side, which was much easier and works under DOS just fine where I have COMPLETE control over things.

This is something of an illusion: You never had control of how those ISA signals were generated on the PC side (and as I recall, there were some discrepancies from vendor to vendor in the exact timing of ISA on occasion...

Never noticed a problem on the PCs I used. 8-bit transfers were 6 cycles (old XT style), 16-bit transfers could be 3-cycles if you wanted (AT.) Plenty of time for things, easy to do, no-brainer. There were NO variations in the older boards, with the really old ones clocking the bus directly with the CPU. With the advent of faster cpus, they decoupled the cpu from the bus, but the cycle times were reliable on it and the faster cpu easily could fill the bus capability.

The new systems with the south bridge involved did have their problems. For example, ISA DMA cannot be properly driven over as PCI transactions, so the south bridge needs sideband signals to the main chipset to tell it what in the hell is going on and to avoid interrupting smoe transfer. These sideband signals were/are a source of most of the chipset bugs.

Doing the work in DOS and using assembly I could watch a quadrature encoder with 10k counts/sec running at 250RPM, including the +/- variations on the encoder with means my minimum timing was quite a bit less than the 24us that might suggest, and do it quite reliably with a minimum of 5 reads per level before a transition.

And there was NO mystery.

although I'd agree that if you just doing a card using port I/O -- no memory read/writes or fancy DMA -- it really was dead simple to make it work). With USB (and PCI, to a lesser extent), you can just plop down someone else's chunk of silicon and still have the same thing. Sure, you're trusting that silicon to "do the right thing," but is that really that much worse than trusting the CPU or north or south bridge ICs in the PC itself to do so?

Re: OT;W98 popularity

Over USB, I'd not only have all the information overload problems in such a quadrature encoder design but then I'd also have to add a micro, one of those FTDI chips, do the decoding on the remote micro, and work out a whole set of protocols of my own. And probably a whole lot of other stuff, to boot.

I guess you could say that I'm not all that happy with suggestions to "go use USB." It's not a panacea, for one thing. And it is a LOT MORE work to master, for another. And it forces a shift of certain responsibilities, in many cases where I'd rather have options than being forced.

Jon

I agree totally.

The only solution I can see is to basically ignore the PC for timing-critical tasks and just treat the PC as some kind of flaky dumb terminal. Then put all of the data processing and real time IO in a well documented microcontroller or FPGA, hopefully one that won't be obsoleted next year. At least you can buy a 32 bit microcontroller that is as powerful as the old PCs that we used to use. One thing that annoys me is that none of the cheap microcontrollers has more than a few kB of SRAM on chip, in spite of the fact that a 1Mbyte SRAM chip is cheap these days. For some reason I guess the marketing people want you to place two separate chips and wire up all of the bus lines, which slows down the access time due to the signals having to go off chip.

Chris