Subject: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by will on Sat, 04 Mar 2017 21:36:36 GMT
View Forum Message <> Reply to Message

Hi everyone

I am pleased to announce a new ECB expansion board that I have been working on for the last few weeks. It provides a nice fast interface to a PC over USB. I am measuring sustained receive and transmit rates of 690KB/sec (5.65Mbit/sec) using this board with my Mark-IV SBC (Z180 at 2x18.432MHz, 2 I/O wait states).

The board is based around the FTDI FT232H chip, configured in asynchronous 8-bit wide FIFO mode. To avoid having to solder 0.5mm pitch connections I have chosen to use the Adafruit FT232H Breakout Board (product 2264) which costs about $15 and is in stock at Mouser (part 485-2264), Digikey (part 1528-1449-ND), and no doubt various other distributors.

The connection is presented to the PC as a USB serial device, so it can be treated just like a normal serial port. The baud rate and other serial settings configured on the Virtual COM Port are ignored, it just provides an 8-bit clean path at whatever rate you can pump data in or out of it. The FT232H has large (1KB) FIFOs for both receive and transmit.

The board decodes 4 I/O addresses on the ECB bus. The A2-A7 bits can be freely configured so it can be located anywhere in the I/O address space. One I/O address is read/written to transfer data to/from the FIFOs. One I/O address is a status register, with the important receive and transmit FIFO status bits at the top and bottom so RLA/RRA can shift these bits into the carry flag, permitting tight loops. A third I/O address can be written to signal "send immediate" to the FT232H, this causes any data in the transmit FIFO to be sent over USB at the earliest opportunity (data is also sent automatically after a 17ms timeout expires or when the FIFO reaches half full).

The board supports generating interrupts on either (or both) of two conditions: data waiting in receive FIFO, or space available in transmit FIFO. The interrupt can be routed to ECB lines NMI, INT, IR0--IR7. Interrupts are controlled by writing to the status register.

In addition to the 8-bit datapath through the FIFO, there are two general purpose I/O lines, one in each direction, connected to the FT232H. I have not tested these (or come up with any use for them!) yet, but they were "free" to add to the design. They are controlled by reading/writing the status register.

To keep PCB size (and cost) down, the board is smaller than a normal ECB board. It is exactly 100mm wide (0.076mm narrower than a normal board, only 3 thousandths of an inch, you won't notice) and only 87mm high (about half as tall as a normal board). This means it fits within the 100x100mm envelope where some PCB manufacturing services offer a cheaper "prototyping" service.

The FT232H chip's FIFO mode is slightly odd (arguably designed incorrectly) in that it clocks in data on the falling edge of its /WR input, instead of the standard rising edge. ECB-USB-FIFO handles this by delaying the /WR signal by approx 32ns which easily meets the 5ns setup and 5ns hold time requirements for the FT232H. There is provision on the board to allow for longer or shorter delays if required, although I doubt adjusting this should be required.

The lack of configuration parameters means the board is very easy to program. I have a working interrupt-driven TTY driver for FUZIX written up, it's just a few lines of code and worked on the first try. It was also very quick to write a fast file transfer program called FIFOPIPE based on my earlier FATPIPE program.

I am planning to write a driver for FUZIX to enable remote disk and network access using the DriveWire protocol. I also plan to write a driver for Linux on KISS-68030 to use this as a TTY for a fast PPP link.

I am going to do a manufacturing run of these boards. If you are interested please post here or PM me to register your interest. Note that the production run boards will be made with green soldermask, not red as shown in the photos. I'm not sure how much the boards will cost, depends on volume, probably about $10.

The schematic for the board is attached. Photos of the bare and assembled board are below.

Feedback welcomed. My thanks to John and to Wayne for their advice on the design.

File Attachments

1) ECB_USB_FIFO.pdf, downloaded 350 times

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by denis2342 on Sun, 05 Mar 2017 20:05:43 GMT
View Forum Message <> Reply to Message

Hi will,

if these boards are working on the KISS68030 I'm in for four boards ;) Maybe you can send them together.

What else are you planning? You very active now with the ECB stuff.

btw, what is the 0-ohm resistor for?

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by will on Sun, 05 Mar 2017 22:39:15 GMT
View Forum Message <> Reply to Message

I've noted your order. I am planning to post out all FIFO and Backplane boards together.

The boards appear to work fine with the KISS-68030 -- it's I/O cycles have to be slower than my Z180 to suit the the NS32202. Poking at card from the monitor works fine; I have not written a driver for Linux yet. This means interrupts on the 68030 are untested, but in principle they should work fine.

>btw, what is the 0-ohm resistor for?

If you look at the schematic, lower left corner, you'll find C12 and R3. They form an LC circuit which can be used to further delay the /WR signal to the FT232H if required. I don't anticipate it will be needed so the builder should omit C12 (hence "0uF" value) and fit a wire jumper in place of R3 (0 ohms). The silkscreen on the final boards will have a little text to inform the builder to read the notes before populating those components.

It's also possible to bypass some of the gate delays by running a patch from R3 to pins 5 or 7 on U3 to reduce the delay to the /WR signal should the builder require this -- again I don't anticipate this will be needed, but this way we have the option without needing to cut any traces.

>What else are you planning? You very active now with the ECB stuff.

I'm not working on any more ECB boards at this time, but open to suggestions!

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by ab0tj on Mon, 06 Mar 2017 00:16:07 GMT

View Forum Message <> Reply to Message

Hi Will,

I would be interested in one of these boards. A high speed PPP link on the kiss system without messing with custom baud rates definitely interests me.

Thanks,
Alex

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by will on Mon, 06 Mar 2017 01:05:57 GMT

View Forum Message <> Reply to Message

Thanks, I have noted your order.

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by denis2342 on Mon, 06 Mar 2017 14:07:32 GMT

View Forum Message <> Reply to Message

I assume that the Z180 is slower than the 68030 and that the 68030 would get a higher throughput rate with the FTDI chip?

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by will on Mon, 06 Mar 2017 20:11:20 GMT

View Forum Message <> Reply to Message

Just did a quick transmit speed test on the 68030. I let gcc generate the code. To make it comparable I had it read data out of memory (I used the full text of Alice in Wonderland, about 73KB) and sent that repeatedly. I measured time to transmit 1MB as 1.595sec which is 642KByte/sec. I can probably improve on this a bit by hand-tuning the transmit loop, but I wouldn't expect to get too far above 700KB/sec.

700KB/sec is about twice the speed we can read data off the IDE interface on the MF/PIC.

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by pbirkel on Fri, 10 Mar 2017 20:37:45 GMT

View Forum Message <> Reply to Message

Will: Please sign-me-up for three boards. THX!

will wrote on Sat, 04 March 2017 13:36Hi everyone

I am pleased to announce a new ECB expansion board that I have been working on for the last few weeks. It provides a nice fast interface to a PC over USB. I am measuring sustained receive and transmit rates of 690KB/sec (5.65Mbit/sec) using this board with my Mark-IV SBC (Z180 at 2x18.432MHz, 2 I/O wait states).

To keep PCB size (and cost) down, the board is smaller than a normal ECB board. It is exactly 100mm wide (0.076mm narrower than a normal board, only 3 thousandths of an inch, you won't notice) and only 87mm high (about half as tall as a normal board). This means it fits within the 100x100mm envelope where some PCB manufacturing services offer a cheaper "prototyping" service.

I am going to do a manufacturing run of these boards. If you are interested please post here or PM me to register your interest. Note that the production run boards will be made with green soldermask, not red as shown in the photos. I'm not sure how much the boards will cost, depends on volume, probably about $10.

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines

Posted by ajacocks on Wed, 15 Mar 2017 21:36:57 GMT
View Forum Message <> Reply to Message

I'm interested in a board, as well.

- Alex

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by davetypeguy on Thu, 16 Mar 2017 00:10:39 GMT
View Forum Message <> Reply to Message

Will,

I'd like two boards.

Thanks!

Dave

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by will on Fri, 17 Mar 2017 15:09:01 GMT
View Forum Message <> Reply to Message

PCBs have arrived, I've assembled one and it works well.

I will wait for the Mini Backplane boards to arrive (expected any day now) before I get in touch with everyone who has ordered boards.

I have sufficient stock to meet all orders placed to date, plus a few spares.

Edit: Well the postman just dropped off the Backplane boards, and they work too, so I'll be writing to everyone over the weekend to get details for shipping!

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by will on Sun, 19 Mar 2017 23:59:53 GMT
View Forum Message <> Reply to Message

Hi everyone

The schematic, Bill of Materials, notes on construction, documentation of the I/O Registers and some sample code have now been posted to the Wiki. The first boards will be shipped tomorrow. I have some stock of boards left over for future orders.

Hopefully everyone will find the construction straightforward. I anticipate most problems will be with programming the serial EEPROM on the FT232H board. I've written a short program to do this under Linux, and the FT_PROG program is available for Windows (although I found it to be not exactly straightforward to use). Do post here if you have any problems with the construction.

Thanks

Will

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by will on Tue, 21 Mar 2017 22:56:24 GMT

View Forum Message <> Reply to Message

Hi everyone

Quick update:

Fitting 74AHCT139 at U6 and 74AHCT244 at U9 gives a worthwhile improvement in FIFO read timing. This change has allowed me to reduce the I/O wait states from 2 to 1 on my Mark IV SBC (Z180 @ 2x18.432=36.864MHz).

This has been tested reasonably thoroughly -- 4GB of data transferred both in and out without any error.

My recommendation is therefore to fit 74AHCT139 at U6 and 74AHCT244 at U9. I am currently using 74LS everywhere else.

Thanks

Will

Update: Now transferred 32GB data in both directions (64GB total) without problems. I think we can say it's working reliably!

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by ewout on Thu, 23 Mar 2017 13:07:13 GMT

View Forum Message <> Reply to Message

Very nice!

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines

Posted by will on Thu, 23 Mar 2017 19:53:50 GMT
View Forum Message <> Reply to Message

I've made some measurements with my oscilloscope to see how the timing changes as I replace 74LS with CMOS parts.

Channel 1 (Yellow): board select from 74LS688 ("/SELECT", U4 pin 19)
Channel 2 (Cyan): output enable on FIFO status line driver (U9 pin 1)
Channel 3 (Magenta): ECB bus line D0 (U3 pin 2, in retrospect this was not an interesting signal!)

After /SELECT (channel 1) falls the signal passes through a '139 and a '32 before ending up at U9 (channel 2).

Firstly here's the board using only 74LS parts throughout. The delay from /SELECT to U9's output enable is about 21.4ns.

I then replaced U6 and U9 with 74AHCT. The delay is reduced by 8.4ns to 13ns (-39% versus 74LS).

I then additionally replaced U7 and U11 with 74ACT32. The delay is reduced by a further 4.8ns, to 8.2ns (-62% versus 74LS). You can really see how much noise the fast CMOS introduces!

I was surprised how much ringing the CMOS introduced, I had really tried to reduce induction by using short wires, soldering them the board and wrapping around the tip and ground of my scope probe. Does anyone know of a better way to connect and ground the probes? Or is this ringing actually in the circuit and not an artefact of my poor probe technique?

Upshot is that replacing the 74LS32 with 74ACT32 is probably not necessary but does give a bit more safety margin. While my testing suggests it is not required, I am going to keep the 74ACT32 parts installed.

Subject: Re: ECB-USB-FIFO: Fast USB interface for ECB machines
Posted by etchedpixels on Fri, 24 Mar 2017 20:06:17 GMT
View Forum Message <> Reply to Message

If you have a real serial link I would avoid drivewire like the plague. Drivewire is a gloriously clever hack but it's got some really deep seated design problems
- No windowing
- Client has to poll server regularly for traffic
- No fast way to avoid a full poll if nothing is pending
- Each client poll is basically a busy wait for the latency of response of the server
- Server does not guarantee if you choose to let a given channel block (eg to wait for one of your processes) it won't bung up the entire DriveWire server.

It's also seriously lacking in error detection and recovery, despite being more complex than protocols that do handle it

Given you've got a FIFO and a slow host that I suspect needs to poll to hit the full data rate you might want to look at the link level for localtalk. This pulls a wonderful dirty trick where the header for a packet fits into the FIFO and is then followed by a small delay and the data, so the process is

Header burst
{ IRQ
{ inspect fifo - decide what to do

these two fit into the latency of the FIFO filling plus any delay you need to add on the fast PC end

poll if need for data block.

Even for disk I/O it's probably worth something like that so you don't keep locking the box up waiting fior the other end to service the request, but get to do IRQ handling until the block arrives back.