

EclaireXL - Feature #55

Replace main CPU

07/19/2017 08:14 PM - foft

Status:	New	Start date:	07/19/2017
Priority:	Normal	Due date:	
Assignee:		% Done:	0%
Category:		Estimated time:	0:00 hour
Target version:			
Description			
I really find the stack based zpu annoying to debug. Also as its setup the rom is full.			
Consider replacing with something newer, also with gcc support, such as https://github.com/SpinalHDL/VexRiscv .			

History

#1 - 09/01/2017 09:06 PM - foft

So I'm now thinking about ditching my address decoder rework and doing something more fundamental.

Using this CPU to build a complete system with a large linear address space. With SDRAM, block ram, custom chips, pbi and the 'zpu regs' (usb controller, sd card dma etc). Possible using wb_conmax or wb_conbuilder for the interconnect.

Then once that is working plugging the 6502 and antic on top as additional bus masters - with an 'original' -> linear address mapping, throttle and pbi access logic.

All just ideas right now. Despite not really writing any code for some time I've been researching/thinking about what to do.

#2 - 09/01/2017 10:59 PM - sadosp

foft wrote:

So I'm now thinking about ditching my address decoder rework and doing something more fundamental.

Using this CPU to build a complete system with a large linear address space. With SDRAM, block ram, custom chips, pbi and the 'zpu regs' (usb controller, sd card dma etc). Possible using wb_conmax or wb_conbuilder for the interconnect.

Then once that is working plugging the 6502 and antic on top as additional bus masters - with an 'original' -> linear address mapping, throttle and pbi access logic.

All just ideas right now. Despite not really writing any code for some time I've been researching/thinking about what to do.

:-)

#3 - 09/02/2017 08:58 AM - 917k

Sounds cool. I am wondering what this will do for the project, I mean what would this actually mean to the end user?

#4 - 09/02/2017 09:14 AM - 917k

Sounds cool!

#5 - 09/02/2017 11:00 AM - sadosp

917k wrote:

Sounds cool. I am wondering what this will do for the project, I mean what would this actually mean to the end user?

Beyond the technical part that I really do not fully understand, I think is an effort of Mark to stimulate the interest of some developer who is familiar with the FPGA programming! I hope some day someone to help him..... In reality to help all us...

#6 - 09/03/2017 01:37 PM - foft

There are several problems with the current setup:

- The firmware is constrained by the current ZPU setup. Its hard to debug and tricky to add more memory to it.

- I'm struggling to meet timing requirements, due to the over-complex interconnect glue.
- I've been keen to go towards running the core at 116MHz instead of the current 58MHz, which will allow 32x mode cache-less and I can use fully registered cycles. Currently you may have spotted that 32x and 16x are very close on sysinfo?

I don't see why we shouldn't expose the RISC V CPU to the end user, so it would be a large modern system. Still with 100% compatibility. Well I might stick a 2nd PIA in the 6502 memory map to allow access to the RISC V and registers.

#7 - 09/03/2017 01:40 PM - foft

To Panos' comment. Yeah I'd love some help, if someone could help me define the memory map and write the firmware for the RISC V that would be an enormous help. There are also many HDL parts that need doing if anyone fancies working on the hardware.

#8 - 09/06/2017 09:44 PM - foft

I've managed to get the spinal HDL project compiled, so now have a verilog or vhd file for the CPU. It seems to have 3 ports. Instruction master, data master and debug slave. So I will need to map them into wishbone. They are also in their own format so I need to do some simple adaptors to map it to wishbone, which I think I almost understand how to do!

This article explains a little on how the debugging via GDB can work:

<https://riscv.org/wp-content/uploads/2016/01/Tues1030-RISC-V-External-Debug.pdf>

It looks like I need to map some rom and ram in as well as the debug port. I'm not yet clear how I connect from JTAG to the system bus, to allow GDB to write to the debug port.

I grabbed the compiler from here and it seems to 'just run' which is good: <https://www.sifive.com/products/tools/>

Pretty pleased with progress, but its baby steps yet!

#9 - 09/07/2017 07:43 PM - foft

The author of the cpu kindly got back to me with some details.

The wishbone interface adaptor looks pretty easy.

Apparently the chip uses a non-standard debugging interface though. Which is good actually since its simpler. However he uses a different jtag interface so I need to do some work. He has a patched opened that can talk to the cpu at https://github.com/SpinalHDL/openocd_riscv. This can be used with GDB. It needs adapting to talk via the USB blaster though.

So I have to build the usb blaster jtag side though. Some handy reading on that:

https://www.altera.com/en_US/pdfs/literature/ug/ug_virtualjtag.pdf

<https://www.altera.com/support/training/course/ovjtag1110.html>

Possibly more useful though are some real examples:

<http://idle-logic.com/2012/04/15/talking-to-the-de0-nano-using-the-virtual-jtag-interface/>

<https://github.com/binary-logic/vj-uart>

#10 - 09/08/2017 10:06 PM - foft

Been trying out openocd locally with the cpu running in verilator. gdb connects, code loads and runs. Pretty cool! OK I just ran some commands as documented but I'm still pleased:-)

I'm very impressed with this riscv cpu project - seems like it'll be a real step up from the zpu. Not to mention that it'll be nice to be able to run code from any of the ram!

#11 - 09/08/2017 10:21 PM - sadosp

:-)

#12 - 09/10/2017 09:24 PM - foft

I read up on the 'altera virtual jtag' and think I now understand how to get this connected, at least in theory.

I guess next steps that might make sense are:

- Build a core with virtual jtag and signal tap and try to send it equivalent commands to those used by the vexriscv cpu.
- Modify the openocd code to talk to this and check commands make it to the core
- Build a core with just vexriscv and block ram and try running some code via gdb.
- Wire up the wishbone interconnect and check this still works
- Add the sdram, zpu regs, usb, sd card and atari chips as wishbone slaves.
- Write some c code to display something using gtia (antic is not yet a master!) and play audio via pokey.
- Add 6502 and antic as bus masters - with address translation layer
- Check acid still works loading from aspeq! We'll have no firmware/drive emulation/keyboard support yet. yet.
- Rewrite firmware...

This might take a while, but we will end up in a much better place I think!

#13 - 09/17/2017 02:00 PM - foft

3/3