Subject: Multicomp 6809 and the Serial Ports
Posted by coolbear on Wed, 27 Feb 2019 19:58:00 GMT
View Forum Message <> Reply to Message

I recently assembled a Multicomp II board and have it configured as a 6809. My interests are with Nitros9. I'm using the first serial port as the console and right now, I'm trying to get the second serial port working as..., well as a second serial port.

First question - I see that there are three serial port device descriptors included in the Nitros9 image. Am I correct in assuming that the console is /TERM and serial B is /T0?

Second question - I'm not seeing anything coming out of the second serial port (Serial B). I see that the status register on all three ports (0xFFD0/0xFFD2/0xFFD4) is 0x02, which means that TDRE is set and I should be able to transmit. If I write a hex value to 0XFFD1, I see that character appear on the console. If I write to 0XFFD3 or 0XFFD5, I get nothing on the second serial port.

I'm wondering if that as because while there are three 6850s defined in the FPGA but perhaps the I/O from the second or third 6850 is not brought out to the connector. Clearly, one of the three is not brought out because there are only two serial presences on the PCB.

Any ideas?

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by nealcrook on Thu, 28 Feb 2019 00:05:14 GMT
View Forum Message <> Reply to Message

Hi Coolbear,
if you are using my 6809 FPGA images then there are 3 serial ports:

* One is associated with the PS/2 and VGA "terminal"
* Two more are associated with 6850 UARTs in the RTL. One UART connects to the D connector on the board, and you need to populate the appropriate RS232 driver. The other UART connects to the header strip on the board, and is designed to connect to a TTL serial to USB adaptor.

The VDUFFD0 jumper on the board determines whether NITROS uses the "terminal" or the serial port as its boot console. What it does is swap the hardware decode between the "terminal" and UART!/Serail A: https://github.com/nealcrook/multicomp6809/wiki/VDUFFD0-Jump er

So, if you are using the Serial A as the boot console, it will be on /TERM and so the "terminal" will be on /T0 and hopefully Serial B will be on /T1.

I must confess, though, that I have probably never tested it... but I think Computerdoc aka Kip Koon had it running at one point (using my images) -- maybe with the NitrOS-9 level 1 image.

Neal.

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by nealcrook on Thu, 28 Feb 2019 19:42:01 GMT
View Forum Message <> Reply to Message

Hi Coolbear,

I did a search through my old email and found these instructions from Kip (this was for the L1 port so it might be best to try with this first)

Connect the I/O port up to a db9 serial port on a PC, and run putty. Configure putty for the correct com port. Then in Nitros9 run the "shell i=/t0&" where the /t0 could be any serial I/O port descriptor. This in fact is very much like what we are doing now on the Multicomp. It would create another user that is independent from the first user. I typically run 2 users on my multicomp09 often using this very technique.

..as I said yesterday, you would probably need to use /t1

Let me know how you get on

Neal

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by coolbear on Thu, 28 Feb 2019 20:40:21 GMT
View Forum Message <> Reply to Message

Thanks Neal,

No joy.

Here's what the setup looks like.

I have the Serial (3v) to USB converter attached to MC board on SER B. I have CTS grounded just to eliminate that as a potential problem. RTS is already at 0V. I read the state of the status port on the 6850s and all three report a 0x02, so I should be able to transmit and receive. I'm assuming that the port runs at 115k just like SER A but I have tried a few other speeds. I use one instance of Teraterm to connect to the console and a second to connect to the Windows virtual com port, which in turn connects to SER B. When I type on the second instance of teraterm, I see the RD light on the converter blinking so teraterm is sending out bits.

The following works as expected:
{Term|02}/DD:echo hello >/term
hello

I would expect the following to print to the second instance of Teraterm:
{Term|02}/DD:echo hello >/t1

...but I get nothing. Same for /t0

I also tried:

tsmon /t1&

...and then I tried running the shell as you suggested. None of that worked.

I was thinking about firing up the oscope and verifying that the received bits are getting to the MC board and hence to the Cyclone II pin - but I haven't gotten around to that yet.

Trying to fall back to Level 1 is an interesting idea though. I don't remember where I downloaded the image for Level 2 but I'm sure I have notes on that somewhere. That might be easier than digging out the scope.

Something else that I'm looking at a bit is enlarging /dd. It feels pretty small - maybe something more like 5M-10M would be better.

Did I mention just how amazingly cool this thing is? I mean I now have a 6809 running at 25MHz (at least I think it is. Hmm, I'll have to check on that). Pretty neat stuff!

- David

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by nealcrook on Fri, 01 Mar 2019 20:45:51 GMT
View Forum Message <> Reply to Message

I think that, to enlarge DD, you would either need to rebuild from source or use the toolshed tools to create a larger empty disk image and then copy everything from the existing boot disk onto it.

Next, you need to change the script that creates the SDcard image so that it will insert this new, larger image without overwriting the other disk images

Finally, you need to change the Nitros drive "descriptors" to understand the new SDcard block start address associated with the new (more widely-spaced) disks. You can hack them or rebuild from source. From level2/mc09l2/modules/makefile:

```
# SDOFFSET is the high 16 bits of the 24-bit block address at
# which the disk image starts on the SDcard. It must match
# the value used in the create_sd_image script.
dds0_80d.dd: mc09sdcdesc.asm
	$(AS) $< $(ASOUT)$@ $(AFLAGS) $(DSDD80) -DDNum=0 -DSDOFFSET=0x0280 -DDD=1

s0_80d.dd: mc09sdcdesc.asm
	$(AS) $< $(ASOUT)$@ $(AFLAGS) $(DSDD80) -DDNum=0 -DSDOFFSET=0x0280
```

```
s1_80d.dd: mc09sdcdesc.asm
	$(AS) $< $(ASOUT)$@ $(AFLAGS) $(DSDD80) -DDNum=1 -DSDOFFSET=0x0290

s2_80d.dd: mc09sdcdesc.asm
	$(AS) $< $(ASOUT)$@ $(AFLAGS) $(DSDD80) -DDNum=2 -DSDOFFSET=0x02A0

s3_80d.dd: mc09sdcdesc.asm
	$(AS) $< $(ASOUT)$@ $(AFLAGS) $(DSDD80) -DDNum=3 -DSDOFFSET=0x02B0
```

I don't know your level of familiarity with all this stuff so let me know if you need help or more detailed pointers. A simpler (though not so convenient) approach is to make the 4th disk larger - you can do this without trampling on any of the others or changing and start addresses.

Using the L2 image there is at least some indication that the serial ports should work. Specifically, in my emulator, which is using the "vdu" as /term, I confirmed that you should be seeing output on /t1. I assume that you're using (level2) from my SDcard image on the WIKI?

```
$ ./mc09-run-int
Reading symbols from '../6809M.map'...

02:0x1F7E 8606                LDA   #$06

(dbg) c

6809 CamelForth v1.1  20 Mar 16



OK NITROS9 NITROS9 BOOTKREL Boot Krn
tb0.....................................................................................bKrnP2 IOMan Init RBF mc09sd DD
                 D0 D1 D2 D3 SCF mc6850 Term T0 T1 PipeMan Piper Pipe Clock Clock2 Shell
Date DeIniz Echo Iniz Link Load Save Unlink i2xo[uart0 stat wr: PC=0xc59d, addr=0x0000, w
          data=0x82]

CNitrOS-9/6809 Level 2 V3.3.0

Multicomp09

(C) 2014 The NitrOS-9 Project

**   DEVELOPMENT BUILD   **

** NOT FOR DISTRIBUTION! **

Fri Jun  9 20:13:12 2017
```

http://www.nitros9.org

* Welcome to NitrOS-9 Level 2 on the Multicomp09 *

yyyy/mm/dd hh:mm:ss
Time ?
June 09, 2017 20:16:42

Shell+ v2.2a 17/06/09 20:16:58

{Term|02}/DD:echo hello >/term
hello

{Term|02}/DD:echo hello >/t0
[uart1 stat rd addr=0x00000002]
[uart1 stat wr: PC=0xc59d, addr=0x0002, wdata=0x83]
[uart1 stat rd addr=0x00000002]
[uart1 data wr of 0x68]
[uart1 stat rd addr=0x00000002]
[uart1 data wr of 0x65]
[uart1 stat rd addr=0x00000002]
[uart1 data wr of 0x6c]
[uart1 stat rd addr=0x00000002]
[uart1 data wr of 0x6c]
[uart1 stat rd addr=0x00000002]
[uart1 data wr of 0x6f]
[uart1 stat rd addr=0x00000002]
[uart1 data wr of 0x20]
[uart1 stat rd addr=0x00000002]
[uart1 data wr of 0x0d]
[uart1 stat rd addr=0x00000002]
[uart1 data wr of 0x0a]
[uart1 stat rd addr=0x00000002]

{Term|02}/DD:echo hello >/t1
[uart2 stat rd addr=0x00000004]
[uart2 stat wr: PC=0xc59d, addr=0x0004, wdata=0x83]
[uart2 stat rd addr=0x00000004]
[uart2 data wr of 0x68]
[uart2 stat rd addr=0x00000004]
[uart2 data wr of 0x65]
[uart2 stat rd addr=0x00000004]
[uart2 data wr of 0x6c]
[uart2 stat rd addr=0x00000004]
[uart2 data wr of 0x6c]
[uart2 stat rd addr=0x00000004]
[uart2 data wr of 0x6f]

[uart2 stat rd addr=0x00000004]
[uart2 data wr of 0x20]
[uart2 stat rd addr=0x00000004]
[uart2 data wr of 0x0d]
[uart2 stat rd addr=0x00000004]
[uart2 data wr of 0x0a]
[uart2 stat rd addr=0x00000004]

{Term|02}/DD:

Although I did both the Level 1 and Level 2 OS ports I actually have very limited expertese in NitrOS-9 so if you want to create an application disk image or some user notes or point out some bugs I'd welcome your contribution. It might even motivate me to tidy up some of the documentation and build notes on the WIKI and my Github WIKI

Yes, 25MHz 6809 is neat! Others have this design going even faster

Neal.

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by coolbear on Sat, 02 Mar 2019 07:09:20 GMT
View Forum Message <> Reply to Message

OK, I've resolved the serial port issue - and I have to confess that I feel like a complete idiot. I was assuming that TxD on the MC II board would connect to TxD on the USB adapter. I mean why not. Well, such is not the case, you need to build in the null modem between the adapter and the MC. What a dummy I am. So that's working now. Thanks for your kindness and apologies if I've wasted your time.

On to tweeking /dd.

BTW, I don't think that there is an x/y/zmodem implementation in your standard utilities. I don't have xmodem, but I do have kermit. Please let me know if you would like a copy of that to add to the utility set.

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by nealcrook on Sat, 02 Mar 2019 09:29:19 GMT
View Forum Message <> Reply to Message

David,

glad to hear you found and fixed the problem. I'm sure I went through that same learning process. I've updated the WIKI to highlight this detail.

https://www.retrobrewcomputers.org/doku.php?id=boards:sbc:multicomp:cycloneii-c:start#bringup_debug_of_full_system

yes, I'd be happy to add any utilities. You could also contribute it to the nitros9 code base over on sourceforge.

Neal.

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by just4fun on Sat, 13 Apr 2019 10:25:21 GMT
View Forum Message <> Reply to Message

wrong post... ignore it.

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by computerdoc on Sun, 10 May 2020 05:32:38 GMT
View Forum Message <> Reply to Message

Hi Multicomp09 Enthusiasts,

I have a version of NitrOS9 6809 L2 v03.00.00 for the Coco 3 that Gary Becker adapted to boot from an SDHC Card on his Coco3 FPGA design for the Altera
Terasic DE1 FPGA Board. Many features of the Coco 3 are implemented on this platform. However the one thing I have done inspired by a fellow Coco
Community member that I think you may be interested in is I adapted the device descriptors to make it possible to create 4 - 1GB partitions on an 8GB
SDHC card. If you are interested in how I did this, I had created a setup guide for other Coco3FPGA users for adapting a freshly compiled NitrOS9 boot
image file to run on the Coco3FPGA platform with 4 - 1GB partitions that you could adapt for the Multicomp09 that I would be most happy to share with you.

Hmmm, that reminds me, I need to check the online storage situation on my Multicomp09 boards. I don't remember if they can even use the larger SDHC
cards that 4 - 1GB partitions would require. The Multicomp09 was originally designed to use 2GB SD cards. I'll need to see if the Multicomp09 was ever
upgraded to use the larger SDHC cards. If the Multicomp09 can use larger SDHC cards, I think I just might have to upgrade the NitrOS9 running
on my Multicomp09s to utilize this enlarged online storage system.

Neal, do you remember if the compiled VHDL source code for the Multicomp09 can read SDHC cards?

Kip

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by nealcrook on Sun, 10 May 2020 15:47:54 GMT
View Forum Message <> Reply to Message

Hi Kip,
yes, the SD controller in my design supports SDHC cards. The default setup for NitrOS-9 supports 4, 2MByte disk images. You can use a bigger image for the last disk with no changes (but it may stop you from using that same SDcard for running Fuzix). To put in bigger images for other disks you need to do 2 things for the 2nd/3rd/4th disks

1/ you need to change the start offset of the disk image in the disk descriptor
2/ you need to change the script that generates the SD card image, to put the images at the correct/corresponding start offsets.

regards,

Neal.

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by coolbear on Sun, 10 May 2020 18:28:42 GMT
View Forum Message <> Reply to Message

The setup guide certainly sounds interesting, I'd like to see that.

Subject: Re: Multicomp 6809 and the Serial Ports
Posted by computerdoc on Mon, 11 May 2020 00:41:47 GMT
View Forum Message <> Reply to Message

Hi David,

Neal just reminded me what needs to be done to increase the online storage on the SD Card for NitrOS9. Then I realized what I did to
create the 4 - 1GB partitions for the Coco3FPGA which is very different from what is needed on the Multicomp09 v1.10 board SD card
setup which is the last version of the Multicomp09 board that I built.

The SD Card setup for the Coco3FPGA has only 1 NitrOS9 Operating system with 4 - 2MB disk

drives so there are no other software systems
to be concerned about getting overwritten when enlarging the 4 disk descriptors to create larger disk drive images on the SDHC card.

Whereas, the Multicomp09 SD card has 7 different software packages / operating systems specifically placed on that SD card which are
Camelforth, Microsoft Extended Color Basic, Flex09, Buggy - a 6809 monitor with among other things an inline assembler / disassembler,
Cubix, NitrOS9 and Fuzix. These 7 software systems are located on the SD Card in that specific order.

As you know, the Multicomp09 at power-on boots into Camelforth 1st. From Camelforth, the other 6 software packages / operating systems
can be called using 1 of 6 key words specifically written to transfer control of the Multicomp09 to 1 of the other 6 software systems.

To enlarge the 4 Disk Drive Images in NitrOS9, new locations for the 2nd, 3rd and 4th drives must be calculated so the 2nd, 3rd and 4th
Disk Descriptors and the build scripts both can be updated with the new calculated locations. In addition, new updated and resized disk
drive image files have to be created corresponding to each of the 4 disk descriptors.

So what I need to do, is revisit the Multicomp09 build scripts to make sure I'm remembering all this correctly and begin making the
calculations. Many steps are needed to be updated to make sure the larger disk images are in the correct places on the SDHC card where
the updated Disk Descriptors expect them to be.

In addition to all that, Fuzix has to be moved further into the SD card so it does not overwrite the new resized NitrOS9 disk drive
images previously written by the build scripts as Fuzix is the last operating system to be written to the SDHC card when the build
scripts are executed. Remember, the 4 resized disk drive image files have to be created for NitrOS9 before the build scripts can be run.

Please give me some time to work this long process into my daily life which has also changed considerably since my Multicomp09 days.

I will need to know what new size you desire for the 4 disk drives in NitrOS9. I may need other information as the process progresses.

In answer to your request, the last version of "Preparing Terasic DE-1 & SD Card for Self-booting NitrOS-9.pdf" for the Coco3FPGA I wrote
is attached for you to peruse and enjoy.

Note: Please do NOT use this setup guide unless you have a fully functional Coco3FPGA platform.

This is just to give you some idea of what is involved with a portion of the process to enlarge the 4 NitrOS9 disk drives. Once the new SD
card image is built, I will get back to you.

Oh, by the way, I will need to know what additional software you wish to have on the 2nd, 3rd and 4th disk drives in NitrOS9. As there is no
way to connect to the internet in this version of NitrOS9, to add, change or update any of the software packages on the 4 disk drive images
as of the last time I used my Multicomp09 boards, is to recreate the 4 disk drives' image files and rebuild the SD card image file.

I remember having a lot of fun working with Neal during the development process and testing of the Multicomp09 hardware as well the software.
Fell free to ask questions any time. Well, that is all I have for the moment. Take care my friend!

Kip

File Attachments

1) Preparing Terasic DE-1 & SD Card for Self-booting NitrOS-9.pdf, downloaded 459 times