Newsgroups: unix-pc.general,comp.sys.att
Subject: Those LEDs in the UNIXpc, one last time; (HwNote01)
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: Just me and my computer, Columbus Ohio
Keywords: 7300 LED drivers kernel

This is the first in a series of interesting hardware notes.

Yes, there are LEDs in the UNIXpc. They located on the left side of the system near the front. There are four of them, and this is what they really do while unix is running:

(left to right)

0 RED:	This is the "user LED". It can be turned on and off with the syslocal(2) call. It is not used by any existing applications.
1 GREEN:	This is the one most people get wrong. This LED toggles everytime there is a process context change, and is cleared on the whole second
2 YELLOW:	This is the idle LED. When it is on, there are no processes in the ready to run state.
3 RED:	Heart beat LED. This is toggled on the whole second.

In the I/O scheme of things, they are the low 4 bits of the Micellaneous Control Register (MCR), at address $4A0000, word access. See the file /usr/include/sys/hardware.h for more detail.

The user LED comes in handy when goofing around with drivers and when eprintf would be too expensive. The problem comes with the other bits of the MCR. Since the MCR is write only, you can't read it to get the current state. The kernel takes care of this by having a "last written value" variable, in this case called mcr_save:

```
/*      mcr_save must be written to whenever the real MCR is written to. */
extern ushort   mcr_save;
#define led_on(x)       *MCR_ADDR = (mcr_save &= ~(ushort)(x))
#define led_off(x)      *MCR_ADDR = (mcr_save |= (ushort)(x))
#define led_toggle(x)   *MCR_ADDR = (mcr_save ^= (ushort)(x))
```

As as an example, "led_on(LED0)" produces the code:

```
        and.w   &65279,mcr_save
        mov.w   mcr_save,4849664
```

Note that since there is no locking here, an interrupt routine that uses the MCR could run between these two instructions. Knowing which interrupt routines use the MCR would provide the right kind of information to use the spl stuff <sys/spl.h> correctly to avoid trouble.

Stay tuned.

John
--
Newsgroups: unix-pc.general,comp.sys.att
Subject: MCR2 and hard drive upgrade; HwNote02
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: Just me and my computer, Columbus Ohio
Keywords: 7300 HD upgrade kernel

This is the first in a series of interesting hardware notes.

But wait, there's more!

There have been several major hardware revisions of the UNIXpc. I have seen some evidence of a P1, and the kernel seems to still support a P2, even though I have never seen a P2 running. Most boards out there are P3, P4 or P5. These last three are mostly the same, and it shows. When UNIX is booted, it prints the message "Main board is P3...P5". Some of us lucky folks have a drive with more than 8 heads. The UNIXpc up the rev P5 was only designed for 3 bits of hard disk head select, so 8 heads is the maximum. Convergent came up with a hardware patch which brings the board rev up to P5.1. It consists of a PAL, a bypass cap and 9 wires. This patch is supported in UNIX 3.5 and later. When UNIX is booted with this patch installed, it will show "Main board is P5.1"

The extra bit needed for all this came from an as yet unused port, now called the Miscellaneous Control Register 2 (MCR2), at address $E20000. The low order bit of this register is the 4th hard disk head select bit. This means the kernel has to go through some gyrations to select heads: put three bits over here and the 4th one over there.

Well now; that's all good and fine and most you are yawning about now. Please read along in <sys/hardware.h>. Right after the line:

```
#define HDSEL3  0x0001                  /* Hard disk head select bit 3 */
```

is the line:

```
#define DDRIVE1 0x0002                  /* Hard disk drive select bit 1 */
```

Hmm. Does this mean it's the drive select bit for the first drive? No, the 4th bit of the register that has the three head select bits is DDRIVE0. That must mean that this is the select bit for a SECOND hard disk drive. Everything else in this register is new, so that makes sense. Those of us who have been reading these news groups for while have heard the rumors of another upgrade from Convergent for a second hard drive. The other goodies, namely an internal floppy tape and an internal 2400 baud modem must be what Convergent calls a P6 machine. I wonder if this was going to be a UNIXpc+, or if it is a completely different machine, the miniframe perhaps. I haven't seen specs for Convergent's other machines, so I don't know.

Well, what would it take to add a second hard drive? Each hard drive has two cables going to it. One of them is a 34 wire control signal cable. This is bused to each hard drive, with a termination resistor in the last drive on the cable. Each hard drive has a DIP switch or punch-out pack to pick the select number for that drive. The second cable is the data cable. It has two differential data signals on it, one for read data, one for write data, 11 grounds, and 5 others for a total of 20 pins. This cable can not be bused, there has to be one for each hard drive. The drivers and receivers for this cable also have to provided. There is another problem. How does one select which of the two incoming data streams will be presented to the hard disk data separation circuitry? All this is starting to sound like a lot of serious hardware hacking.

What about the potentially nastiest problem of software? What would have to be changed to get the gd disk driver to control the second drive? One could format the second drive by temporarily switching it over to the drive 0 spot and then switching it back to drive 1. Well, if you will now cd over to your /dev directory, you will find some interesting things. There are two kinds of disk drive special files: character and block. The character are rfpxxx, and the block are fpxxx. A hard drive can have at most 16 partitions, by default the first partition is the VHB/BBT, the second is the swap, and the third, covering the rest of the disk is the user data partition. These correspond to the fp000, fp001, fp002 files. The floppy drive only has two paritions: VHB and user data. WHAT? NO FLOP SWAP? Sorry. That means that [r]fp00[0-9a-f] are for the hard drive, [r]fp02[01] are for the floppy. What are [r]fp01[0-9a-f] used for? Sixteen partitions for a second hard drive? Hmm.

If one could upgrade a system to handle two hard drives, would the P5.1 patch need to be installed? In theory, no it wouldn't need to be. It would be just fine to have two hard drives with <=8 heads. If fact this might be the most common application. Lots of people can dig up two 20M drives, but maybe not a 40M drive; or two 40M, but not an 80M, etc. Then there are the POWER users who want to get TWO 190M Maxtors on there.

A second drive might present another opportunity. The UNIXpc power supply is
already overloaded. If one were to take the internal hard drive out and put
it with another in an external box with an external power supply, the load
would be taken off the UNIXpc. The whole noisy mess could be shoved in a drawer
somewhere too.

The machine across the table from me shows this for a "mount" command:

```
/ on /dev/fp002 read/write on Mon Oct 10 20:01:01 1988
/mnt on /dev/fp012 read/write on Mon Oct 10 20:03:21 1988
```

Now wait a minute...

John
--
Newsgroups: unix-pc.general,comp.sys.att
Subject: I have a second hard drive on my UNIXpc; HwNote03
Keywords: 7300 3b1 HD upgrade
Message-ID: <359@uncle.UUCP>
Date: 11 Oct 88 07:05:29 GMT
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: Just me and my computer, Columbus Ohio
Lines: 84
Xref: uncle unix-pc.general:1644 comp.sys.att:3518

```
/mnt on /dev/fp012 read/write on Mon Oct 10 20:03:21 1988
                ^
```

** YOU BET! **

I see you're back. Yes, that output from my mount command was correct. I DO
have two (2) II hard drives working on my system.
I can do an mkfs on it.
I can run fsck on it.
I can mount it.
I can copy files from drive to drive.

That bit in MCR2 is indeed a select line for the second hard disk. The problem
of switching data streams turned out to be trivial. It is already supported
in the "Hard Disk Data Separator PAL". It has two pins which are grounded:
DDRIVE0* and DATA1. The nasty, messy part was what I did for the driver and
receiver. There are three spare receivers and two or zero spare drivers
depending on the version of your mother board. Some mother boards have 74LS240s
driving the Tx and Reset lines to the keyboard, later boards use two of the
26LS31 differential drivers for this. The machine I have has '240s for the
keyboard, so the two drivers are free. The problem is that the inputs for the
spares are grounded to prevent gray-state power consuption and damage. I had to
pull out three pins on each chip to break the ground connection.

The software turned out to be almost no problem at all. It turns out that the
P5.1 upgrade is needed to access the second drive. At least the gd driver
thinks it's needed. Since the P5.1 is completely invisible with the PAL out
of the socket, I tried taking it out and the accessing the second drive. No go.
It turns out that there is a word variable in the kernel "revlev" which tells
various drivers what the board revision is. It must be 0 for <=P2, 1 for
P3...P5, and 2 for >P5. You can find out your revlev with adb:

```
# adb /unix /dev/kmem
revlev/d
^D
```

You can also change your revlev to something else:

```
# adb -w /unix /dev/kmem
revlev/w 2
^D
```

It works just fine after the system is already booted. With the P5.1 PAL out

and after the adb session above, I can use the second drive just like with the
PAL in.

While I was goofing around testing this, I did get one error:

```
#HDERR ST:51 EF:10 CL:42C3 CH:4201 SN:4200 SC:4202 SDH:4221 DMACNT:FFFF DCRREG:91 MCRREG: 8B00
#drv:0 part:2 blk:4192 rpts:1

panic: gfpwrite F_HD_DMA set
```

Has anyone else ever had this one with only one drive?

I haven't even started to consider all the yucky physical stuff, namely an
external box to put all this stuff in. My next step is to try to change two
chips for one in the select circuit. After that, I am going to try out this
patch on another, hopefully different machine. No, I don't need beta testers
for this right now.

Yeah yeah yeah. How can I get one? Well, that raises an interesting question.
Assuming Convergent does have an upgrade for a second drive out. I guess it
would be like the P5.1 patch, that is, something licensed to a VAR. Now for
the really important question: Since I came up with this patch on my own, using
widely available information, do I have the right to do whatever I want with it?
Would publishing schematics with a Copyright help or hurt? Does Convergent
really give damn about the old S/50 anymore?

What will all this do to the future market for the overpriced $350.00 SCSI
board from IDT? Oh, well, gee sorry guys. I guess you took too long getting
it to market. I have not yet decided what I want to do with this. The thought
of selling it, well did sort of cross my mind. I was thinking about a daughter
board with three or four chips and a connector.

Now you've got something to do with all those full height 10M disks! Mount
the second drive on in /usr/spool/news normally. When you want to back up
your system, go single user, dismount news, drag out a stack of 10M drives,
back the system up, mount news back on, go back to multi-user.

Wow, maybe someone will even take another try at working on X!

John
--
Newsgroups: unix-pc.general,comp.sys.att
Subject: Well, maybe not so easy...; HwNote04
Reply-To: jbm@uncle.UUCP (John B. Milton)
Followup-To: 359@uncle.UUCP
Organization: Just me and my computer, Columbus Ohio
Keywords: 7300 3b1 HD upgrade

Another in a series, so far about adding a second hard disk drive the UNIXpc

Well, I got one UNIXpc upgraded. It seems the .5M machine I got from DDS is
older than the one I'm working on now. Now that I look at the schematics, I
find that the older machine (the one I got working with two hard drives) must
not match the schematics. If it did, it wouldn't have worked. The machine I'm
working on now does seem to match the schematics, and is not working, just like
I would expect. Once I get ahold of the data sheet for the WD1010, I should
be able to figure out how to get this one working.

I am now very much in favor of a small daughter board for this upgrade. I
managed to fry the 26LS32 receiver chip on the second machine. This daughter
board would fit into the socket at 14M. The PAL in that socket would be put
on the duaghter board. For all of you who have or would be will to open up
your UNIXpc, please check this location (14M), and see if the chip is indeed
socketed. I need to know as soon as possible if anyone has this PAL soldered in.
Convergent seems to be good about socketing all the expensive chips and the PALs

on all the boards I've seen so far.

So far, the daughter board will have: 26LS31, 26LS32, 74LS175, the PAL from 14M, a 20 pin DIL male connector for the data cable to the second hard drive, a 100 ohm resistor, four bypass caps. Another reason for the daughter board is a firm mounting place for the connector.

Gary Sanders (gws@n8emr) had a neat idea which I think could be implemented on the daughter board. There is at least one problem with the current release of UNIX on the UNIXpc. That is the "slow down and lock up bug". With this bug, the system gets slower and slower and finally locks up. It could be some program on the clock tick callout that eventually takes up more the 1/60th of a second. My guess for culprits there would be the window driver. The symptom after it has already locked up is: The mouse pointer still moves around just fine. The time in the status manager's window is the time it locked up. Typing a key will cause the flashing of the cursor to skip a beat, but no characters will get echoed.

Gary's idea is to have a long duration watch dog timer on the reset line. The idea is to set the timer for, say 20 minutes. A program "petdog" would be run out of cron on 5 minute intervals. This program would attempt so operations such as disk or window, then reset the watchdog timer. As long as the watchdog keeps being reset, the system stays up. If the watchdog does not get reset, the system gets reset. Unfortunately, the XRST line on the expansion bus is driven by a gate, so an expansion card can not reset the system. This means the line must be driven from inside the machine.

Gary Sanders has the board I already got working. He will soon be stress testing a setup with two drives. He has the second drive exteral in a PC box. He also tells me some interesting news that Western Digital has a new hard disk controller chip that does RLL and is almost pin compatable with the WD1010. He'll post the details when he gets the data sheet.

Jan Isley (jan@bagend) says he will be looking into adding a 1.2M floppy soon. Right now he's having problems getting postings out onto the net. The 1.2M floppy looks like it might be possible because the Western Digital Floppy Disk Controller has a pin on it to switch from 5.25" to 8" disk drives. The high density floppy drives look just like 8" drives I'm told. There is also no problem with changeing the pin under software control. I have an S-100 Z-80 system that has another FDC from the 279x series, the 2793. I have used the S-100 with both 5.25" and 8" at the same time. Of course having two floppies would at least require a new gd driver.

One more note while I'm thinking about it. As most of all of you know, if you enter "s4test" or "S4TEST" to the 7 choice menu in the diagnostics, you will be put in "expert" mode. Mostly, this mode allows you to run just one test, or run it many times. The ? command at the "expert>" prompt displays a full screen of help. The list of commands has the hard disk test routines listed as #6. When you enter 6 or 6,0 it will run a default list of tests. When you enter the command as 6,n you can run a specific sub-test. The know subtests are: 1:recal, 2:format, 4:read sectors across disk, 5: random seek and read id, 6: non-destructive read, 8: spare a sector, 12: print VHB and BBT, 16: go to interactive device test mode ("i>" prompt, exit with "q"), 23: DESTRUCTIVE surface test. DO NOT RUN THIS EXCEPT ON NEW DISKS!!, 24: park head. The part they don't tell you here is that all these tests can be run on the second hard drive by using the test number 2 instead of 6. So, 2,12 will display information on the second hard drive. Note that the diags also check to see if you mother board is at least rev P5.1 before it will access the second hard drive. I can't think of any way to fudge the revlev in the diags. One would have to swap the second drive to the first for formatting and testing

BTW, the diagnostics are bootable just like /unix. The diagnostic disk is a mountable file system. If you copy the s4diag file from the diagnostic floppy onto you hard disk, you can boot the system with diags without using the floppy drive. The trick to doing this is to get the verbose loader onto the hard drive.

The way to do this is to format a floppy with the fdfmt.vl command, then copy the verbose loader from the floppy to the hard disk with the /etc/ldrcpy program

The next time the system is booted, it will stop and ask which drive you want to boot from, then which file you want to load from. It needs the beginning / for the file name, e.g. "/unix". There is one very obvious problem with using the verbose loader, of course. If you system resets from a power failure, it will not reboot, it will wait asking questions. This may be a feature at some installations.

Sorry for this one being so boring.

John
--
Newsgroups: unix-pc.general,comp.sys.att
Subject: All right! and 1.2M floppies; HwNote05
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: Just me and my computer, Columbus Ohio

I see you're all still out there.

First thing's first. If you have ANY ideas for hardware enhancements, e-mail them off to me. Let me know about any special deals for hardware that give you the idea and all possible uses you can think of. You will get credit for your idea in the form of recognition, but that's all! If I sell it and make $1M, tough luck. No flames, please. I doubt if anyone would send me any earth shaking ideas. What I'm looking for are things like "Why don't you make a board using the new XYZ graphics/hard-disk/tape/floppy/short-wave chip"

Well, I figured out how to get the newer generation UNIXpc mother board working with a second hard drive. The problem is this discrepancy in the schematics. The reference manual has two copies of the schematics, with assorted changes between the two. The older of the two is not as old as the board I first did the upgrade on.

The problem is the DRUN pin going to the WD1010-05. This pin needs to have a signal from an external one-shot which is triggered whenever a pair of one or zero bits (the same in MFM) comes in from the drive. The one shot does this by triggering on the data signal from the 26LS32 being low for 250ns. This separate detector is needed because the PLL (Phase Locked Loop) data separator is not turned on yet. Once the HDC detects 8 cycles on the DRUN pin, it thinks it has found an address mark and drives the RG (Read Gate) pin to activate the external PLL. Once it does that, it can start interpreting the data from the drive and start looking for sector headers.

Apparently on the old board the data signal sent to the one-shot came from after the PAL, so it got there no matter which drive is active. The newer board I just upgraded DOES match the schematics. The data signal goes straight from the 26LS32 to the one-shot. They must have had some kind of delay or stability problem going through the PAL and switched over to the raw data.
To fix the problem I had to find some way to switch which drive data is being sent to the one-shot. It turns out there is a spare 3 input NAND gate at 13M (pins 3,4,5,6). That must be it, 'cause it works!

The important question is whether or not there are any more incompatibilties waiting out there. I guess the next thing is to put together a prototype daughter board so I can get all the details nailed down. I am still considering putting the watchdog circuit on the daughter board.

On the 1.2M floppy. Convergent cheaped out here and went with the 2797 rather than the 2793. The reason being that they got a Side Select Output pin. What they traded was an internal clock divide pin. This means that the clock to the 2797 has to be externally changed from 1MHz to 2MHz when going to high density. The VCO filter circuit (on the PUMP pin) also needs to be affected. There are several little triangle notes around the FD about not using the

2797, but that would present another bunch of problems, ending in software. Fortunately there is a 2MHz clock available. I think a flip flop, an open collector buffer, a resistor and a latch should do it. The latch is already in my current design for the second hard drive upgrade...

An interesting note: I just looked at the schematics for the floppy tape board. They run the WD2797 in high density mode. The analog stuff on RPW, WPW, VCO and PUMP is different. They do have a 2 MHz clock. Hmm.

I suppose what one could do would be to permanently modify the FDC to do high density by just copying the stuff from the floppy tape. As long as you still have access to a system with a low density setup you could access low density disks. With a 1.2M disk you could mount the floppy on /usr/spool/news and expire once an hour :)

Next time: 4M machine with a 1.5M Combo card (maybe)

John
--
Newsgroups: unix-pc.general,comp.sys.att
Subject: Memory boards this time; HwNote06
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: Just me and my computer, Columbus Ohio
Keywords: More memory hard-ware upgrades UNIXpc 7300 HwNote

Hey folks,

The 2nd HD upgrade is proceeding. I am going to get the board layed out soon.

This time the subject is what to do with that old clunky 512k memory board you have laying around that you can't use now that you have a 1.5M Combo Card. I know, I know everyone says you can't put it in because the Combo card "covers up" all three slots. Well, warm up your soldering iron, because that's just not true if you change a few things...

Expansion memory in the UNIXpc starts at address $200000. Expansion I/O starts at $C00000. Memory gets 512k per slot, I/O gets 256k per slot. See table below for a list of slot number vs address. There is a good explanation of all this on page 3 of the floppy tape section of the Reference Manual. Slot detection is done by the board in a relatively neet way. There are three pins provided to each card that do NOT come from the 100 pin connector on the mother board. They are ID0, ID1 and ID2, pins 30, 29, and 28 respectively.

```
Slot    I/O     Memory
----    ------- -------
   0    $C00000 $200000
   1    $C40000 $280000
   2    $C80000 $300000
   3    $CC0000 $380000
   4    $D00000
   5    $D40000
   6    $D80000
   7    $DC0000
```

Each slot has a different bit pattern encoded on these pins. The Combo card uses the ID pins by feeding them into a 74F85 4 bit magnitude comparator. With this chip you put two four bit numbers in, and get a < = > output on three pins. For the serial chip on the Combo card, it compares address 18, 19, 20 and GND to the slot ID 0, 1, 2 and the XIOEN* pin. The XIOEN* pin is asserted whenever address pins A23=1, A22=1, A21=0 and FC2=1 (supervisor sccess). This means that the "=" pin on the comparator will be driven any time I/O to the 256k chunk for the slot the board is plugged into is accessed.

Each board has an ID "register" in the last four even bytes at the end of it's I/O space, so that the kind of board in each slot can be determined. The ID

code for a floppy tape board is 01 10 FF F0. If this address is read and there
is not card in that slot, the CPU will get a bus error. With all this info,
the machine can be acurately dynamically configured at start up time without
the need for dip switches or having to run a configuration program. It was
a real big hype situation when the Amiga came out with this brand new old idea.

A memory only card does not need an ID code because when memory is found, you
know what it is. The Combo card is nasty because it does not pay attension to
the slot ID for it's memory. Each of three banks of 512k on the Cmobo card
takes up memory slots 0, 1 and 2 no matter which slot the card is plugged into.
This means that if you have a 1.5M Combo card, there is no slot ID 3 for you
to plug your 512k memory card into. The 512k card IS well behaved and uses the
slot ID to affect the base address of the memory on it.

By now you have probably figured out where all this leads. If the 512k board
is lied to about the slot its in, it could appear at a memory slot other than
the physical one its plugged into. There is only one slot where memory can go
that there is no physical slot, number 3. If the 512k card sees ID0 (pin 30)=1
and ID1 (pin 29)=1, It will be in slot 3. I see your hand is already reaching
for the solding iron...

*** PROCEDURE TO FORCE A 512K MEMORY CARD INTO SLOT #3 ***

```
 1. Get a working 512k board.
 2. Shutdown you machine, take the Combo card out.
 3. Put the 512k card in.
 4. Boot diagnostics. It should say that you have 524227 bytes Expansion memory.
    It is not telling the truth! you really have 524228 bytes!
 5. Enter s4test at the prompt.
 6. Type 15 at the expert> prompt to test memory.
 7. It will test the main board memory first, and then the expansion. Notice
    what address the memory is at. Should be ${20,28,30}0000, for slot {0,1,2}
 8. Power down and take out the 512k card.
 9. Examine the area around the connector. Pins 1 and 33 are marked.
10. Solder suck pins 30 and 29, break them loose and pull them up a little. This
    can be tricky since the pin is the spring stuff the connector contacts are
    made out of.
11. Insulate the two connector pins. If they bend back down to the solder pads
    after you're done, IT WILL SHORT +5 TO GND AND BURN OUT YOUR POWER SUPPLY!
12. With a piece of wire, connect the two board holes where the pins were to +5
    There are two pullup resistors right there. Where they come together at the
    far corner of the board is +5.
14. Put some kind of sticker on the board down at the other end so some other
    fool will know. Something like "HARD ADDRESSED TO MEMORY SLOT 3 ($380000)"
13. Put the board back in, power up, boot diagnostics, test memory. You should
    see the address of the expansion memory at $380000.
```

Now for some advice. The power supply in this machine is wimpy. If you have a
3b1, then you have a "3b1 power supply", which means you have a cable for the
hard drive power coming directly from the power supply. AND THAT IS ALL IT
MEANS!! The amount of power available is the same for both the 7300 and the 3b1
power supply. I can't seem to find any guide as to which boards draw how much
power. Put the board in sometime when you're going to be around so you can
keep an eye on the machine.

If you have a 1.5M Combo board, you're already in the 2M to 3.5M range. Adding
another 512k isn't going to make a whole lot of difference unless you have a
lot of users or you're running some program (KCL?) that is a big memory hog.

As always, e-mail me if you have any trouble or my instructions differ from
what you see.

John
--
Newsgroups: unix-pc.general,comp.sys.att,u3b.tech

```
Subject: Upgrade to 512k memory board upgrade; HwNote06.01
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: U.N.C.L.E.
Keywords: memory 512k upgrade HwNote hard-ware UNIXpc 7300

Hey folks,

Yes, I am still out here. YES, the second hard drive is STILL GOING TO HAPPEN!

I have had questions from people who couldn't quite figure out how to do the
hardare patch to the 512k memory boards, so this is as upgrade to HwNote06.
I will call this HwNote06.01. Keep the old 06 around, just append this to it.

Once again, this concerns using the 512k (0.5M) RAM ONLY BOARD with a FULLY
populated EIA/Combo board. If you don't have both of these ingredients, you
can't make this batch of cookies. According to the manuals, there is no way to
have both a fully populated (i.e. 1.5M) Combo Card in your system AND a 512k
memory board, which would give you a complete 2M of expansion RAM. Well, as I
pointed out in HwNote06, this is not quite true, if you're willing to get out
the solding iron...

If you want the long version, with all the details, see HwNote06.

*** PROCEDURE TO FORCE A 512K MEMORY ONLY CARD TO WORK WITH A 1.5M COMBO ***

 1. Read all this through first! Print it out, as it's tough to read when the
    system is powered down.
 2. Contemplate whether you think you can really do this. If you think you
    might be able to, but you're not sure, get a hardware person to be around
    while YOU do it.
 3. Get a working 512k board, a low wattage soldering iron, rosin core solder,
    a sponge to wipe the soldering iron tip on, some wire wrap wire or other
    fine wire, a copy of the diagnostic disk, some anti-static stuff, one of
    those wrist ground straps (if you're into anti-static bondage), and a
    "solder sucker". No, this last item does not have to do with the one before
    it, rather it is a device for quickly removing molten solder from a hot
    solder joint. I prefer one of these, rather than solder wick. If you have
    not used a solder sucker before, get a scrap PC board to practice on. Any
    board from an IBM-PC will do :).
 4. Shutdown your machine from the silly menus or with the "shutdown" command.
    You have to be logged in as root to do this.
 5. Re-boot the system with the diagnostic disk and park the hard disk heads if
    you have to. Some hard disks need it, some do it themselves, thank you.
 6. Turn the POWER OFF, and wait for the hard disk to completely stop spinning
    before you slam the machine around.
 7. Take the Combo card out. There should be two little phillips head screws
    holding it in. Set it aside on some anti-static stuff.
 8. Get out the 512k card and put it in any slot.
 9. Wonder why you didn't use the static bag the 512k card was in from step 5.
10. Turn on the soldering iron, get the sponge wet, put it out of the way.
11. Boot diagnostics. It should say that you have 524227 bytes Expansion memory.
    It's lying! You really have 524228 bytes.
12. Type "s4test" at the prompt.
13. Type "15" at the expert> prompt to test memory.
14. It will test the main board memory first, and then the expansion. Notice
    what the address of the EXPANSION memory is. It should be ${20,28,30}0000
    for slot {0,1,2}.
15. Don't walk away while it's testing memory, hot soldering iron.
16. Park the heads, power down and take the 512k card out.
17. Examine the area around the connector. See figure 1 below.
18. Solder suck pins 30 and 29 on the connector. Break them loose and pull them
    all the way out of the holes. This can be tricky since the pin is the
    spring stuff the connector contacts are made out of. It takes a
    surprisingly large amount of force to get them out.
19. Insulate these two connector pins with tape. If they bend back down to the
```

```
    solder pads after you're done, IT WILL SHORT +5 TO GND AND MAY BURN OUT
    YOUR POWER SUPPLY!
20. With a piece of wire, connect the two board HOLES where the pins WERE to +5
    You could use any +5 on the board. I used the leg of resistor R6 closest to
    resistor R2. (see figure 2 below)
21. All done with the dangerous stuff. Unplug the soldering iron.
22. Put some kind of sticker on the board down at the other end of the board,
    so some other fool will know why this board behaves so weird. Something
    like "HARD ADDRESSED TO MEMORY SLOT 3, address $380000"
23. Put the board back in, CAREFULLY power up. Lean over and listen as you
    click the power on. If the phone relays go click click click click click
    real fast, OR NO RELAYS CLICK, you've got those pins I told you to be
    careful with shorted! TURN THE POWER OFF IMMEDIATELY. Check pins 29 and 30
    and make VERY sure they're no touching anything. If your hard disk does
    not spin up, you may have ONE OF THOSE drives where the brake melts to the
    spindle. Take all the cards out and try to power it up again. If the hard
    drive still won't spin up, ****LIGHTLY**** spin the spindle shaft to
    break it loose.
24. Do 9 to 12 above. You should see that the address of the expansion memory
    has changed from what it was to $380000. It won't matter what slot you put
    the modified 512k card in anymore.
25. Park, power down, add the Combo Board, do 9 to 12 above. It will take
    quite a while to check out all the memory, especially if this brings you up
    to 4M. When you boot the diags, you should see a total of 2M expansion RAM.
26. Put the cover plate on over the 512k RAM card.
27. Re-boot UNIX and make sure the memory amounts UNIX reports are right.
28. Did you really turn the soldering off?
29. Pat yourself on the back. (if you need it)
30. Watch your system over the next few hours to make sure it's not overheating.

Summary:
Shut down, power down, lift and insulate pins 29 & 30, connect where they used
to be to +5, label it, put it in any slot, test with diags, re-boot.


---------------------------------------------------------------------------

       ___________________________________
  _____|_________________________________|_____
  |  |o|'''''''''''''''''''''''''''''''''|o|  |
  |  +-----------------------------------+    |
  | *   ^                               ^     |
  |     `-- 33                          `- 1  |
  |                                           |

Figure 1.
---------------------------------------------------------------------------



---------------------------------------------------------------------------
                       R2
Either of these -->  --===--
                    |
                    # R6
                    |

Figure 2, enlargement of the * in figure 1.
---------------------------------------------------------------------------


A repeat of SOME of my advice from HwNote06:

Now for some advice. The power supply in this machine is wimpy. Adding one more
card to your machine may push it over the edge. After all this, you may have to
take the 512k card out and forget about it. I can't seem to find any guide as
to which boards draw how much power, so you'll just have to guess it out. If
```

your hard drive won't spin up with the 512k board in, that's a good sign you're near the limit.

I have heard conflicting information from many different sources about the 3b1 power supplies. Some people swear up and down that the 3b1 power supply is exactly the same as the 7300 power supply except for the drive cable. Other people say that the 3b1 power supply is definately a different, heftier critter. Would someone who has access to both please tell the rest of us the real story?

If you have a 1.5M Combo board, you're already in the 2M to 3.5M range. Adding another 512k isn't going to make a whole lot of difference unless you have a lot of users or you're running some program (KCL?) that's a big memory hog.

As always, e-mail me if you have any trouble or my instructions differ from what you see.

Stay tuned.

John
--
Newsgroups: unix-pc.general,comp.sys.att
Subject: Could a 68881 Math co-processor go fast?; HwNote07
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: Just me and my computer, Columbus Ohio
Keywords: HwNote fpa 68881 UNIXpc hard-ware 7300

Parts of this article are copied from mail to David Wexelblat (dwex@mtgzz.att.com), who suggested a 68881 when I asked for hw ideas. The rest of you just don't seem to have any ideas :-!

I was thinking about that one a while ago. AT&T actually made a 68881 board, they even have a part number, 105160253. I talked to some guy at AT&T, and he says the board was never release due to poor performance. Since the 68010 does not have a coprocessor interface, the 68881 would have to operate as an I/O device, so it would work fine as an expansion card. David tells me that the software interface was implemented using an F-line exception handler. While this is clean from the compiler point of view (you just generate 68881 instructions), it is expensive:
1. Execute instruction
2. Get F-line exception
3. Stack everything away
4. Start running the common kernel exception code
5. Pull all the parameters from the user PC and talk to the 68881 through I/O
6. Defer to another process while the instruction runs on the 68881
7. Get an interrupt, get the status
8. Call the kernel code to return from exception.

Steps 6&7 could be replaced with
6. Loop polling the 68881 for completion
7. Get the status

I don't know what they did about having an expanded process context for procs using the 68881.

It seems the only way to get decent speed is to have the program using the 68881 be the only one using it and to access it directly. David's idea is to modify GNU C to generate the proper I/O directly. The only code that could directly access I/O would be kernel code, like a loadable driver. It wouldn't be so fun to write all your floating point programs as loadable drivers. Hmm, you would have unlimited CPU... Nah

On the other hand, there is a messy solution. The user only has access to the user memory mapped to his process. There is a routine plock(2) which might be used to lock a page so that the physical address of the page would never change and would always be there. I have never tried to use this on the UNIXpc, I just checked to make sure it's in the manual. The user address could be passed to a /dev/fpa driver via ioctl(2). The driver coulld take this address and find the physical address corresponding to it. This address could be passed from the driver to some I/O which would map the 68881 in every time there is an access to this page. When the file to this driver is closed, the mapping is disabled. The fpa device could only be opened once. The program would have to run suid to call plock(), but it could do a setuid(getuid()) once it's running, which would ba a reasonable compromise.

The hardware could not be on the expansion bus for this idea, because you could not be sure that a given user page is on an expansion memory board. There could also be nasty problems if the page is used for anything else while mapped to the 68881. Some good news is that the 68881 could be run asyncronously, so it could have it's own clock. So yes, it could be a 33MHz 68882 (if they're that fast yet?). Yes, it would cost more than the machine. I don't know what the current price is for the 12.5 MHz 68881. Assuming I could whip together something that would work. How many people would be inter-ested in buying a board without the 6888[12]? Please e-mail.

Ok, tell me how much of this I got wrong. Also send me any other ideas. Since this would all be done in I/O, I suppose it could be ANY over priced FPA. I know what you're thinking now, how about a DSP instead? Sure, I think all this would apply.

John
--
Newsgroups: unix-pc.general,comp.sys.att
Subject: Bus expansion (one way or another); HwNote08
References: <367@uncle.UUCP> <733@bacchus.UUCP>
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: Just me and my computer, Columbus Ohio

In article <733@bacchus.UUCP> darren@bacchus.UUCP (Darren Friedlein) writes:
>OK - here's another question. What's involved in building a card cage for
>the UNIXpc if it was possible to get these connectors? Could it be done
>with like ribbon cable? Since this machine IS based on a back-plane, this
>would appear to be possible. Any comments?

It is not strictly a bus like S-100. 100 pins come off the mother board, and 99 pin Eurocard-ish connectors are used in the bus. Each slot has 3 pins for slot ID, which are of course different for each slot. Other than that, the rest of the signals (as far as I know) are bused. The little 3 slot bus board in the UNIXpc is an 8 layer board.

There is an "Expansion Unit" out there, Comcode 405176785, I've seen a price of $1695 for it. It adds 5 slots and has a board that plugs into one of the slots. You can't put memory in it. I have thought about building a bigger bus. My thought was to take the little three slot bus out and use a new stacked 8 slot bus. Turn the whole mess into a vertical "tower" arrangement:

```
     +------ mother board
     v
 [==]|   <-- floppy
 $$$$|   <-- hard drive(s)
 $$$$|   <
 ----|   \
 ----|   |
 ----|   |
 ----|   |-- 8 slots
 ----|   |
 ----|   |
 ----|   |
 ----|   /
 %%%%    <-- Power supply
 %%%%
```

The problem is, who would buy it? A new bus PC would have to layed out. Probably even re-designed to be properly balance for all 8 slots. The power supply would have to be out-and-out changed. New cables: hard disk (3), floppy (2), video, keyboard. You would need some kind of base for the monitor. Few people would even look into it without a proper case, and on and on. How many people out there can afford 8 cards for this beastie anyway?

The better idea seems to be a big project looming in the future. What everyone wants is lots of cheap expansion. Nothing for this machine is cheap compared to the PClones. You guessed it, why not build an IBM-PC bus bridge? A SCSI card for the PCs only costs $45. I think they're putting serial cards in Cracker Jack boxes these days. There are lots of other tantelizing boards out there: Hard and floppy disk, xGA, serial, parallel, IEEE-488, gaggles of co-processors. I don't even want to think about DMA just yet. The part that scares me most is noise trouble. My friend James Nugen (jcn@uncle) has had an IBM hard disk controller working on an Amiga, so this is close along those lines.

The software for this idea would be a nightmare. The IBM-PC BIOS pieces parts are on cards, a lot of good that does us. I don't think it would be very fun to run an 8086 interpreter in the device driver. This all leads to the same problems the MINIX folks are having with all these boards. It also means we have a good source for drivers, hard disk in particular.

I would envision a new twist to get this attached to the UNIXpc. A lot of people are already maxed out to 3 cards, so they don't have a slot to spare. For something like a bridge, which needs everything on the bus, I am wondering if a flexible PC edge connector tap might work. I thought this one up talking to Gary Sanders (gws@n8emr) on the phone. The idea is to design a flexible PC board which matches the fingers on the mother board. You detach the mother board from the bus connector, fold the flexible PC up the right way, wrap it around the mother board fingers, and then carefully plug the whole mess back into the bus edge connector. The flexible PC could be layed out with interleaved grounds and heavy traces for power. It could even be folded between the UNIXpc and a PC box for easier Advanced Cable Management. The problems would be cost and mechanical repeatability of the connector end. Has anyone ever seen this done before?

Just thought I'd put one MORE iron in the fire.

John
--
Newsgroups: unix-pc.general,comp.sys.att
Subject: Convergent tells all, more HD stuff; HwNote09
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: Just me and my computer, Columbus Ohio
Keywords: UNIXpc 7300 hardware upgrade Convergent floppy tape

Hello again everyone,

What follows is, as they say from the horses mouth. The horse in this case is Convergent Technology (CT), makers of the UNIXpc/7300/3b1/S-50 for those of you tuning in late. This information comes from someone who actually worked on this machine. His name is in /usr/installed/.list. I spoke with him today on the phone, but Jan Isley (jan@bagend) tracked him down. I thank him a great deal for spending time with me on the phone.

The UNIXpc was based on the THEN EXISTING miniframe. Both are 68010, same virtual memory, etc. As for how many machines there are: about 200 level P2s, 5-10k P3s, 20-30k P4s, the rest P5s, for a total of about 70000 machines. The next rev of the machine, P6 was in the works, as a manufacturing cost reduction. The P6 was a reality internally at CT in the prototype stage. There were never any production machines made because AT&T was backing out. The rest, as they say is history. The differences in the P6 are:

1. Another head select bit for the hard drive (known outside as the P5.1 upgrade)
2. Another hard drive select and data connector.
3. A new one-chip data separator to replace the dicrete one.
4. Internal floppy tape drive.
5. Internal 2400 baud modem.
6. Change in the Hard disk controller from WD1010-05 to WD2010-05. (maybe)

Several of us have the P5.1 upgrade. It is supported in UNIX 3.5 and later.

I have reverse engineered the second hard drive upgrade, which will become a three drive upgrade (for a total of four hard drives).

There are lots of hard disk data separators out there, so switching from the discrete one would be a good thing. I don't think existing machine really need to be upgraded because it does work.

The internal floppy tape was implemented using the existing FD controller. It does not disable the regular floppy drive. CT could not get a stable circuit to switch back and forth between 250kb and 500kb, so they elected to use a different flavor of the Cipher 525 that uses 250kb. It gets the same amount of data on the same kind of tape as the external floppy tape drive. Now wait a minute, that's right. The trick is that this other kind of drive runs the tape half as fast! The problem is that this other flavor of floppy tape drive is hard to come by. The support bits for the internal floppy tape are part of the MCR2 register, see <sys/hardware.h>.

Ah yes. Everyone who has read <sys/hardware.h> and <sys/modem.h> has wondered what all the 2400 baud stuff is about. Many people have posted stuff saying it can't be done. Well... almost. The machine as it sits CAN send 2400 baud data, the problem is that it can't receive it. This part needs a proprietary AT&T DSP chip. I don't know what the part number is, but if you open up an AT&T 2400 baud modem, you'd might see it.

The 2010 was never supported correctly, and there could be an operation problem, although I have never heard of one. The difference on the outside between the 1010 and the 2010 is pin 4, which is a No Connect on the 1010 and SDHLE* on the 2010. This pin is used to drive an external latch to hold the head and drive select bits when the internal SDH register is written to, saving external decode cicuitry. In the UNIXpc this is already done. This feature would be for someone designing a new board. The 2010 does provide some support for ECC in the form of generation and detection, but not correction. The data sheet is very weak here, assuming the reader knows all about ECC correction.

Now the fun part. There WAS an upgrade piggy back board made for the head, drive and floppy tape parts. This is the board some people have heard about. Some machines were upgraded within CT. I just got a FAX copy of the schematic, parts list and parts placement from Jan Isley, who got it from CT. AT&T decided not to use the upgrade for a variety of reasons, so they did not purchase it from CT. CT does not care about the UNIXpc or the upgrade. VARs were never given rights to the upgrade.

What's on the board:

A 74273 to latch the floppy tape signals, a 74273 to latch the drive signals, a 74240 to drive the floppy tape and hard drive signals, a 26LS31 to drive the second hard drive data, a 26LS32 to receive same, a 7402,04,08 for the rest. There is also a circuit to mask the floppy interrupt under software control. This is bit 10 if MCR2 (F_MASK). This was wanted by the software folks, but was apparently never implemented in the gd driver.

The method used for the second drive differs from the one I came up with in that it was meant to be compatible with the P[2345] and the new P6 data separator. They are essentially the same. The data bus part is merged with the head stuff. The internal floppy tape was really working at CT under 3.51, but no one knows whether it made it into the 3.51 distribution. Everything I can find says no.

Supposedly some very late model 3b1s have a place for a connector for the internal floppy tape drive. That is, the tape drive is internally controlled, but not internally mounted. Everyone with 3b1s look around your left fan and e-mail me if you see anything that looks like a place for a connector.

They do (or did) have some extender boards. One is straight out, and one has a right angle bend. They are two layer boards, and they had problems with them in that the memory boards would not work right on them. They had a kluged up bus card that faced out the back of the machine that would work.

Expansion cards: CT made the Combo board, some of the memory boards, the floppy tape, and the Ethernet. Alloy made the DOS-73 coprocessor, the expansion box and maybe the 68881 board. CT was going to do the Star Lan board, but AT&T did it instead. I don't know about the Voice Power (Lenny?). I was wrong about the expansion box and bus masters in there, but I was right about the memory, it can't go there. The expansion box has 6 slots in it, because it passes the third slot, for a total of 8 slots.

Those $%#*&@&* connectors: They did initially use the 96 pin standard connector, but they needed 3 more pins. At that time the 99 pin connector was a standard part. Some of the extender cards they used did have 96 pin connectors on them. I don't know how they managed that one, because on one end there is: 1:GND, 34: EXP2BG, 67:XMA16 and at the other 33:EXP1BG, 66:XMA17, 99:GND.

That's about it. I'm working from the notes I took today on the phone, so there are probably things I forgot.

Oh, you noticed I'm shooting for ALL FOUR HARD DRIVES. Well, yes I am. I think I can come up with a loadable gd driver to replace the one in the kernel. I'm looking in that direction now because it looks like the only way to do the floppy tape, a high density floppy and four hard drives. Some of you may be saying FOUR HARD DRIVES, COME ON! Well, the reasons I gave before are still valid now. A lot of us just can't afford to buy one big drive, but we might be able to come up with a bunch of little ones.

The part about a loadable gd driver is the fact that the one in the kernel is in use. I don't know if I can transfer the running to drive context to my driver completely. Overlaying an open driver doesn't bother me since Ford's kbd driver does it. Maybe if I code it in assembler I can patch it over the one in the kernel. By looking at /etc/lddrv/unix.sym, I see that the gd driver seems to span from gdqrdblk at 0x16988 to gdwrprot at 0x2548c, 60164 bytes. That should be more than enough. I'll just hack it out before bed...

Other problems: <stand.h> says NMOUNT is 4 and not accessable through ktune(7), so that would be the largest number of partitions mountable at once. With four HDs, you couldn't mount a floppy, you'd have to umount one of the hard drives first. Oh well. That raises some other problems with pre-defined <sys/space.h>.

These are extracted from <sys/space.h>

```
-> NMOUNT=4, HDMAXCYL=1400
struct mount mount[NMOUNT];
char gdhdbbq[HDMAXCYL], gdsybbq[HDMAXCYL];
struct bbmcell gdhdbb[HDMAXBADBLK], gdsybb[HDMAXBADBLK];
struct gdsw gdsw[] = {
        {       0,{"WIN", 1023, 4, 17, 68,0, 0, 512},
                -1,0,16, 1023*16,40, ghdintr, ghdstart,
                HDMAXCYL, HDMAXBADBLK, gdhdbbq, gdhdbb,0},
        {       0,{"WIN1", 1023, 4, 17, 68,0, 0, 512},
                -1,0,16, 1023*16,40, ghdintr, ghdstart,
                HDMAXCYL, HDMAXBADBLK, gdsybbq, gdsybb,0},
        {       0,{"FD", 80, 2, 8, 16,2, 0, 512},
                -1,0,8, 80*8,60, gfpintr, gfpstart,
                FPMAXCYL, FPMAXBADBLK, NULL, NULL,0}
};
struct iobuf    gdutab[DISKS];
```

Hmm. I don't know if I can just allocate the rest of the space for the
additional drives in the replacement driver or not. I suppose it's about time
to get ahold of a copy of The Loadable Device Driver Writer's Guide. Well, I'm
up 173 lines already.

Stay tuned. Same BAT time, same BAT channel. Ooops, sorry Robin.

John
--
Newsgroups: unix-pc.general,comp.sys.att
Subject: Re: Second Hard Drive for UNIX-pc; HwNote10
References: <413@limbic.UUCP>
Reply-To: jbm@uncle.UUCP (John B. Milton)
Distribution: world
Organization: U.N.C.L.E.
Keywords: unix-pc hard-disk upgrade toes crunch ouch HwNote

In article <413@limbic.UUCP> gil@limbic.UUCP (Gil Kloepfer Jr.) writes:
[describes merciless feet stomping and how he got the 2nd hard drive going]

I'm glad to see the schematic you got worked ok.

Yeah, yeah, I've been slackin' lately. My design is now complete except for
some board layout details. Now for the grubby details...

Unlike the P5.1 upgrade, this upgrade CAN NOT be passive.
Some wires just HAVE TO BE CUT FOR IT TO WORK!

```
For older mother boards:
  13N-1 must be clipped loose and connected to grounded. 13N-2 must be clipped
  loose and connected to 14M-3. This is actually an upgrade done by Convergent
  on later mother boards.
For all mother boards:
  13K-3 must be clipped loose from the board.

Parts list so far:
 1 double sided PC board
 1 26LS31 differential line driver
 1 26LS32 differential line receiver
 1 16R8 PAL
 2 16 pin sockets
 1 20 pin socket
 4 20 pin headers
 2 34 pin headers
 3 bypass caps
 4 100 ohm resistors

Other parts needed:
 1 long 34 pin daisy chain cable with one header connector and at least two
   drive connectors
 1 an internal 34 pin cable to go from the mother board header to the header on
   my board
 ? 20 pin data cables for each hard drive
 1 external box and power supply

Connections inside the UNIXpc:
 1. Data bus line D1
 2. Data bus line D2
 3. Data bus line D3
 4. The register select line MCR2SEL*
 5. volts.
 6. The hard disk read data line to the hard disk data separator PAL, 14M-3
 7. The hard disk write data line, 16M-6
 8. Maybe the RST* line.

It will:
 1. Add select lines for 3 more drives to the 34 pin cable. (total of 4 drives)
 2. Provide a 34 pin header.
 3. Provide active data connectors (20 pin) for 4 drives.
 4. Work no matter how you mount it. (within reason)
 5. Make you the envy of all your friends.

It will not:
 1. Provide P5.1 board version feedback.
 2. Provide additional 4th head select bit.
 3. Provide floppy tape support.
 4. Provide a watch dog circuit.
 5. Force you to take out your fan if you don't want to.
 6. Cost too much. A price of $50.00 is what I'm shooting for now.
 7. Affect the price of tea in China.

Since the 3 drive selects have to be added to the 34 pin cable, the ground and
DDRIVE0* line will be picked off of the 34 pin cable from the mother board.
This will save two patch wires.

I am assuming that your machine will have the "P5.1 FIELD UPGRADE" already
installed. I will make that available as a SEPARATE kit. As I have mentioned
in the HwNotes, you do not HAVE to have the P5.1 to use two drives. You DO have
to have the P5.1 to format the second drive without swapping cables around.
I plan to have a patch for the diag disk (3.5 and 3.51) by the time I start
selling. The reason I am not providing the P5.1 on my board is that a lot of
people already have the P5.1 installed. I also wanted to keep the number of
wires going to my board down. The P5.1 part can be installed and tested
separately, thus easing the upgrade process.

I will be providing: the board, schematics, parts list, board layout,
background, theory of operation, patch wiring instructions, formatting
instructions, and probably a disk with HwNotes and other stuff. The text part
of the instructions will be posted for review.

Work is progressing on modifying the hard disk driver to USE all four hard
drives. It will not be available with the initial release of the board, but
will be posted in source form if and when I get it done. Until that is
available, you will be restricted to only two hard drives.

As far as the fan wars go, I have designed the board so that the cable headers
will match up with the fan grates. I will recommend in my instructions that
the fan be removed, the board be placed in that space, and that the existing
internal hard drive be mounted in the external enclosure with the second drive.
There will, of course be a melt down disclamer :)

As always, it's never too late to get your opinions and advice in on the
design of this board. If there are bugs found later, I would expect at least
50% of them fixable with a PAL change. Any of what I have described here may
change between now and when I get evrything working.

John
--
Newsgroups: comp.sys.att,unix-pc.general
Subject: Re: Can the 7300's modem go 2400 Baud?; HwNote11
References: <426@polyof.UUCP>
Reply-To: jbm@uncle.UUCP (John B. Milton)
Distribution: world
Organization: U.N.C.L.E.
Keywords: HwNote P6 2400 floppy tape second hard drive
```

```
In article <426@polyof.UUCP> steve@polyof.UUCP (A2 Steve Weiss ) writes:
>
>       While browsing through /usr/include/sys I found several
>things which would lead one to believe that the on board modem on
>the 7300 can go 2400 baud.
Here we go again...

>       Those things are as follows:
>
>hardware.h:/* D5-7 are used for 2400 baud modem */
>hardware.h:#define BAUD2400    0x0020                  /* Select 2400 baud on modem */
>modem.h:#define Baud2400_SCM_Wr1       0x10  /* also need R8b3 to be set  */
>modem.h:#define at2400Baud_SCM_Rr2     0x0C
>modem.h:#define SetSynchMode_SCM_Wr4           0x80  /* 1200 and 2400 baud only  */
>modem.h:#define Set2400Baud_SCM_Wr8            0x08  /* R1b4 should also be set  */
>
>       Can the modem go 2400 Baud?  If not, why were these #define's made?
```

Those 2400 things are there BECAUSE...

Convergent Technologies, the OEM of the S4 (safari 4) a.k.a UNIXpc, was working on the next model of the machine. This was to be mother board revision P6. Things to be added:

1. 2400 baud internal modem.
2. Provisions for a second hard drive.
3. Addition of a 4th hard disk head select bit.
4. Addition of a floppy tape drive, simlar to the external one, but slower!
5. Addition of a some cicuitry so that the kernel could sense a P6 mother board and react properly.

Some of the software needed to make all this work was completed and put into the standard distribution. Notably: support for the 4th head select bit and the second hard drive (/dev/*fp01?). The 2400 baud modem stuff was only put in as stubs. There is no real code to drive the 2400 baud modem if it was there. The 2400 baud modem DID work on machines inside Convergent. It used a proprietary AT&T 2400 DSP modem chip. This is the sort of thing AT&T uses to make their own modems. It is not something the rest of us could get ahold of for reasonable prices. These days, there is no point with 2400 baud modems coming down in price.

The internal floppy tape is an interesting story. The external tape drive for the UNIXpc is also a "floppy tape". Floppy tape is a very appropriate name, as the connectors and pin definitions for the connectors are almost identical to a regular 5 1/4" floppy drive. It can also be controlled with a regular floppy disk controller chip. The idea for the P6 was to use the existing internal floppy disk controller. The problem here is this: the floppy drive is the regular double density drive. It has a bit transfer speed of 250k. The external tape drive has a bit rate of 500k, just like 8" floppy drives and AT 1.2M 5 1/4" floppy drives. The Floppy Disk Controller (FDC) chip used in the UNIXpc, the Western Digital 2797 can deal with both bit rates.

The trick is that the clock it uses must be able to shift from 1MHz to 2MHz, and the yucky analog stuff for the phase locked loop must also "shift". The external tape drive has it's own controller board, with its own FDC, tuned to the 500k data rate. Well, to make a long story short, the guys at Convergent could not get this to work. To get around the problem, they used a slightly different model of the tape drive used externally. This other tape drive could write a tape IN THE SAME FORMAT, and yet use a 250k data rate. How, you ask? This other drive moves the tape past the head TWICE AS SLOW. Crude, but it works. Since these drives are targeted to existing floppy systems, it is understandable. For those of you have seen the UNIXpc external tape in action, you can imagine just how slow it would be going twice as slow. I imagine Convergent would have worked out the problems to use the high speed drive, but the P6, and indeed the whole machine was shot down.

I looked at adding the circuitry to do the floppy tape stuff to the board I'm making, but I decided against. The software to support the internal floppy tape drive IS NOT provided with UNIX. I am looking at adding/substituting a 1.2M floppy for my next project.

The 2,3,4 above are going to be on the board I'm building. I have the PAL done and simulated, but not burned. I was going to use the "P5.1" PAL to do the 4th head select bit and the version feedback, and then a new PAL to do the 2nd hard drive select and hard disk data multiplexing. Oh, you noticed I said "was". Yes, I changed the design again. I have integrated both sets of functions on ONE PAL. This makes the design a total of 3 chips (PAL, driver receiver), which is as low as it can get without going full custom or macro cell. The PAL I have chosen is a relatively new, somewhat more expensive one, the 22V10. This is a WONDERFUL PAL to work with. I have barely scratched the surface with my needs as far as what this PAL can do. I mostly went with this PAL because I need lots of I/O pins, and output configurability. That "V" stands for versatile, and it means it. If I goof something up, there should be lots of slack with this PAL to fix things.

As has been said before, if you don't want to use the 4th head select stuff (you already have the P5.1 PAL installed), you simply don't connect up all the wires. I like Gil's idea of using a socket on one of the spares. The logical way to have the best of both worlds is to use the P5.1 wiring pattern. This will work, since there are some NC pins that can be used WHEN DUMPING THE ON BOARD P5.1 stuff. It's 2:30, time to go to bed and think.

It is looking more and more like I won't be re-writing the gd driver. It would probably be a small loadable driver that would use almost all of the gd driver in /unix. The extra arrays needed to support two more (for a total of 4) drives could be allocated by the driver when it is loaded. Basically, the loadable driver would provide another character, and another blocked major device. These would mostly act as drive select front ends. Unfortunately, there are a lot of details to be worked out. I may have to do a drive table swapping kind of deal to make it all work.

Now for the important part. As lenny and others have suggested:
All of this would be much easier if some of the tables in /unix were increased in size. A very few things could be tweeked before UNIX is compiled for the next fix disk. This would make life MUCH easier for some of us.

```
I suggest:
  gdisk.h:
    DISKS from 3 to 7 (4 HDs, 3 floppys) (from 2 bits to 3)
    (see DISK_n, and gdsw[])
    HDMAXCYL from 1400 to 2048 (might be a disk format change)
    HDMAXBADBLK from 128 to 512 ""
  usr/src/UTS/kern/cf/conf.c: (guess)
    NMOUNT from 4 to 16 (lots of partitions on lots of drives)
    NLBDRV from 4 to 8 (blocked drivers)
    NLCDRV from 10 to 20 (character drivers)
```

I probably missed a few, and some of this is based on how other UNIX systems do configuration. I guess my dream would be to have a set of libs and .o files for UNIX, JUST LIKE EVERY OTHER UNIX system I've worked on. For those of you who don't know, most UNIX systems are shipped with a set of lib???.a, *.o and some *.c files. The system is configured by modifying .h or .c files, compiling them and re-linking the kernal. With our system, they chose to do system re-config through the ktune(7) stuff, and only provide a subset of the changable stuff. If ktune were expanded, that would work too, and might be quiker and easier to implement. It would require an extra line for each parameter here and there, a new ktune program, and a new man page. Anyone who can pass this through to AT&T, please do. If you have a contract whereby you can INSIST on

```
these sorts of things, do that. PLEASE!

I'm looking to have Beta boards ready to roll by the end of the month, or
early Feb. I don't want to pay the nasty prices for rush PC board work.

As always I appriciate any comments. I may not appriciate it too much after
I have boards made...

The price of tea in China remains unaffected.

John
--
Newsgroups: comp.sys.att,unix-pc.general
Subject: Well, I'm almost there...; HwNote12
Reply-To: jbm@uncle.UUCP (John B. Milton)
Distribution: world
Organization: U.N.C.L.E.
Keywords: HwNote hard drive PAL board upgrade

Hey all,

Well time for an update. I have finished laying out the board for the second
hard drive/P5.1 upgrade. I have also finished the definition for the PAL.
This is how it's gonna go (so far):

One PC board with 3 chips: 26LS31, 26LS32 and PAL22V10. The PAL incorporates
all the functionality of the "P5.1 Field upgrade PAL", that is, it will:

1. Latch and drive the 4th hard disk head select bit, HDSEL3*
2. Latch D4 when MCR2 is written to
3. Drive the latched MCR2 D4 onto the data bus when PHSTAT* is asserted.

Items 2 and 3 provide "revision level information", used to set revlev and
change your mother board from "P3...P5" to "P5.1". New functions provided by
the PAL22V10:

4. Latch DDRIVE1*, DDRIVE2*, DDRIVE3* so that four hard drives can be selected.
   Note that the bits for DDRIVE2* and DDRIVE3* are not defined in the include
   <sys/hardware.h>. I just picked the next two bits up from HDSEL3 and
   DDRIVE1*. They WOULD be defined thus:

#define DDRIVE2 0x0004                  /* Hard disk drive select bit 2 */
#define DDRIVE3 0x0008                  /* Hard disk drive select bit 3 */

5. Multiplexing of the hard disk data streams (all four)
6. An extra inverter, in case I need it later.

I will be using Gil's idea of getting signals from the existing P5.1 socket.
I know, "but what if I don't have the P5.1 installed?". No problem, you just
install the part of it my board will be using. My board needs additional
signals that are not on the P5.1 socket (you DID put a scoket in didn't you)

These are the signals from the old P5.1 socket used:
 2. PHSTAT*
 4. D4
 7. D0
 8. MCR2SEL*
 9. RST*
10. GND
17. HDSEL3 (not used, I drive it from my board)

The P5.1 socket has a number of unused pins, which I will be using:
 3. D3
 5. D2
 6. D1
```

```
13. GND (not connected on the UNIXpc side)
14. HDMUX (multiplexed hard disk data from the PAL)
15. GND (not connected on the UNIXpc side)
16. HDWRDAT (hard disk write data)
18. (left unused)
20. VCC
```

Yes, my board will be getting power from the P5.1 socket. The GND from the P5.1 socket is connected to the hard drive signal cable ground on my board.

Once the mother board modifications have been done to your machine, all you have to do is hook up two cables: The hard disk signal cable (34 pin) and a 20 dip header to 20 pin dip head ribbon cable from the P5.1 to a similar socket on my board. Since my board will not be permanently attached, if it goes bad, all you have to do is unplug it. All three chips will be socketed, so if one goes bad it should be easy to replace. This is the current parts list:

U1	PAL22V10	Pre-programmed PLD
U2	26LS31	Quad differential line driver
U2	26LS32	Quad differential line receiver
C1-3	0.1uf	Bypass, or decoupling capacitors
C4	50-100uf	Local surge supply.
R1-4	100 ohm	Line termination for 26LS32
JD1-4	20 pin header	For connection to four hard drives
JS1-2	34 pin header	One to the mother board, one to the external drives
JH1	20 pin socket	For connection to expanded host "P5.1" socket

Misc: 2-16 pin sockets, 1-24 pin socket, mounting hardware.

As I think I've already said, this board will work with or without the P5.1 already installed. If you have the P5.1 in, you've already got a lot of the work done. If you don't have the P5.1 installed, or don't know what that is, don't worry. As far as physical placement of drives goes, I have designed my board to work best having ALL HARD DRIVES MOUNTED EXTERNALLY. Since there is absolutely no way you can run a second hard drive inside, unless both drives are super low power 3.5", I have assumed that since one drive HAS to go outside, they should BOTH go outside. I have seen the UNIXpc sitting on a regular IBM-PC clone box, and that should be the most cost effective solution, since PC boxes and power supplies can be had very cheap. I have been told from many different sources that the cables to the external hard drives can be as long as THREE FEET, so you can have your hard drives sit in a corner and whine.

I have designed the board to fit in the left fan grate, since in my opinion, it is the best place. Howevery, YOU DON'T HAVE TO if you don't want to. Remeber, in the two drive configuration there will be no less than five cables running to this board, three of which will be running outside the UNIXpc, just begging to be tripped over.

I still have to check over my PC layout with some other people to make sure what I designed can really be made. When I defined the PAL, I used a pretty heafty SIMULATION section to test it out, so I don't expect problems there. If anyone knows of a good, fast cheap (I know) PC board production company, let me know. I'm going to be scrounging up the bucks to cover the setup fee and a short run of boards. No, I don't need anymore Beta testers.

While I'm lining up a PC production facility, I'll be working on the documentation and installation instructions. Right now I'm going to bed...

If you have any comments YOU BETTER TELL ME NOW!

John
--
Newsgroups: comp.sys.att,unix-pc.general
Subject: Re: Summary: Hard disk errors on a 3b1; HwNote13
References: <388@ntvax.UUCP> <465@manta.pha.pa.us>

```
Reply-To: jbm@uncle.UUCP (John B. Milton)
Distribution: world
Organization: U.N.C.L.E.
Keywords: HDERR 3b1 disk errors Seagate HwNote
```

In article <465@manta.pha.pa.us> brant@manta.pha.pa.us (Brant Cheikes) writes:
...
>OUTSTANDING QUESTIONS I'D KILL FOR ANSWERS TO:

Well, I don't think you'll have to kill...

>1. What are the meanings of the following fields in the HDERR message:
>   ST, EF, SC, DCRREG, MCRREG.

A typical error message in /usr/adm/unix.log:

```
HDERR ST:51 EF:10 CL:FF80 CH:FF01 SN:FF00 SC:FF02 SDH:FF25 DMACNT:FFFF DCRREG:95 MCRREG:9100 Tue Aug
2 02:00:38 1988
```

All disk requests go through a generic part of the disk driver, gd. This part does not know the difference between a floppy and a hard disk, it just knows blocks. It calls various low level, disk specific routines to get it's work done. The error above comes from the low level hard disk routine, gdhd. If it were a floppy error (FDERR), it comes from the floppy low level routine, gdfp. You get one of these after each error from the WD1010 hard disk controller. If it retries 8 times, you get 8 messages. After the low level routine returns to gd, it is SUPPOSED to print one of these:

```
drv:0 part:2 blk:20294 rpts:2 Sat Jun 11 14:43:06 1988
```

The "rpts" is how many retries (?DERRs) the low level driver took to complete the request. I have no idea why SOMETIMES you get this message and most of the time you don't. If the low level driver retries <sys/gdisk.h>:GDRETRIES (15) times, you will get one of these little babies in the [!!] icon:

```
        Unrecoverable hard disk error
```

These are the real badies. Another thing you might notice when this happens, is that the hard disk will do a slow seek to track zero. This will happen when there are 5 retries left. Some drives are VERY noisy doing slow seeks, others you won't notice at all. You WILL notice a big delay just before the [!!] icon pops up if the head was WAY out on the disk.

Now to the question about the meaning of all the fields in the HDERR message. All of the values are the state of things when an interrupt occures, usually at the end of a disk operation. All of the information here comes from pages 3-8 to 3-10 of the "Storage Management Products Handbook, 1986", from Western Digital.

```
HDERR ST:51 EF:10 CL:FF80 CH:FF01 SN:FF00 SC:FF02 SDH:FF25 DMACNT:FFFF DCRREG:95 MCRREG:9100 Tue Aug
2 02:00:38 1988
```

```
ST: The "status register" from the WD1010
  Bit 7 Busy. Set when the controller is accessing the disk.
  Bit 6 Ready. Reflects DRDY, pin 28; DREADY*, pin 22 from the HD
  Bit 5 Write Fault. Reflects WF, pin 30; DFAULT*, pin 12 from the HD
  Bit 4 Seek Complete. Reflects SC, pin 32; DSCOMPL*, pin 8 from the HD
  Bit 3 Data Request. Reflects DBRQ, pin 36. Driven by the WD1010, NC.
  Bit 2 Reserved. Always 0
  Bit 1 Command in Progress.
  Bit 0 Unrecoverable error. Indicates the error register should be checked.


EF: The "error register" from the WD1010
  Bit 7 Bad Block Detect. From what I can tell about how things are done on
```

```
    our systems, this feature is not used. We use a direct mapping method where
    the position of bad blocks is determined by the bad block table. If this
    gets turned on, it is some kind of glitch on the disk.
  Bit 6 CRC Data Field. This one deserves a direct quote:
      "This bit is set when a CRC error occures in the data
       field. With Retry enabled, ten more attempts are made
       to read the sector correctly. If none of these attempts
       are successful, the Error Status is set also (bit 0 in
       the Status Register). If one of the attempts is suc-
       cessful, this bit remains set to inform the Host that
       a marginal condition exists. However, the Error Status
       bit is not set. Even if errors exist, the data can be read."
    On our machiones, if bits 7, 5, 1 or 0 are set or if the error register is
    not zero!, or if there was DMA trouble, an HDERR message will be printed.
    This is extremely good. It means every time there is the slightest flicker
    in the data, you will get an error message. If you get only one, the error
    is probably transient and does not mean anything. You should NOT try to
    lock out the block! If you get a bunch of CRC errors, but a good read,
    this is probably a weak spot and should be locked out.
  Bit 5 Reserved. Always zero.
  Bit 4 ID not found. Like CRC, this bit is set when the ID field for the
    requested sector can not be found, or has a bad CRC.
  Bit 3 Reserved. Always zero.
  Bit 2 Aborted Command. Should never happen on our system. If you get it, it
    probably means BAD power line trouble.
  Bit 1 Track Zero Error. This is very bad, and usually indicates a very bad
    hardware failure in the drive, so you'll never see it until you get a
    second hard drive on your system :)
  Bit 0 Data Address Mark Not Found. Yet another thing not found.

Our driver DOES NOT use the built-in retry feature of the WD1010. This means
that when the driver retries 15 times on a CRC error, you only get 15, not
150 retries. Retries can apparently be DISABLED for both the hard disk and
the floppy disk through an ioctl(f,GDRETRY,1). I could see doing this for
floppies, where you would want to discard any marginal disks, but I don't see
much practical use for the hard drive.

The values CL, CH, SN, SC, SDH are also registers in the WD1010. They are
printed as simple %x, but are only significant in the low order 8 bits. The
top 8 bits are just bus garbage, and will vary with the machine, whether, time
of day, or phase of the moon.

CL: Cylinder low. This register hold the least significant 8 bits of the of the
  cylinder number.
CH: You guessed it, Cylinder High. It contains the high order two (2) bits of
  the cylinder. Also as you might have guessed, it is the high order three (3)
  bits if you have a WD2010.
SN: Sector number. 'nough said.
SC: Sector count. Our driver DOES use this feature to transfer multiple sectors.
SDH: This is a catch all. It contains ESSDDHHH: Extension bit, Sector size,
  Drive select, and Head select. These values compare with the sector header
  on the disk. The sector size and head are written on the disk, but the drive
  number is NOT. This is why you can format any drive at any select code, and
  then move it to another select code without problems. The extension bit is
  not used on our machine. When turned on, it extends the place for the CRC
  from two bytes to seven bytes for externaly generated ECC codes.

The DMACNT is a register in our custom DMA circuitry. From <sys/iohw.h>:
#define DMA_CNT_MASK            0x3fff  /* Bits 13...0 holds dma count */
#define DMA_ERROR               0x8000  /* dma error bit mask, 0 = error */

The DCRREG is a miscellaneous register for disk stuff (top of page 11 in the
schematics). This register can not be read, so what is shown is the value
of dcr_save. From <sys/iohw.h>:
#define NOT_FDRST               0x80    /* 0 = reset, 1 = not reset */
```

```
#define FDR0                     0x40    /* 1 = floppy selected */
#define FDMTR                    0x20    /* 1 = floppy motor on */
#define NOT_HDRST                0x10    /* 0 = hdc reset, 1 = hdc not reset */
#define HDR0                     0x08    /* 1 = hard disk 0 selected */
#define HDSEL                    0x07    /* Head select mask */
```

```
The MCRREG is another Miscellaneous Control Register for some of everything.
It is on the bottom right of page 15 of the schematics. The only bit of any
importance here is DMA_READ. As with DCRREG, this register can not be read.
The value printed comes from mcr_save. From <sys/hardware.h>:
#define CLRSINT                  0x8000  /*  CLRSINT- toggle from 1 to 0 and
                                         back to 1 to dismiss level 6, 60
                                         hertz interrupt */
#define DMA_READ                 0x4000  /* DMAR/W- 0 = disk DMA write
                                                    1 = disk DMA read */
#define LPSTB                    0x2000  /* LPSTB+  toggle from 0 to 1 and back
                                         to 0 to strobe data to line printer */
#define MCKSEL                   0x1000  /* MCKSEL- 0 = modem RX & TX selected
                                                    1 = programmable Baud Rate
                                                         generator is selected */
#define LED3                     0x800   /* LED3-  0 = on, 1 = off */
#define LED2                     0x400   /* LED2-  0 = on, 1 = off */
#define LED1                     0x200   /* LED1-  0 = on, 1 = off */
#define LED0                     0x100   /* LED0-  0 = on, 1 = off */

As you can see from the bit definitions above, only the low order 8 bits of
DCRREG are used, and only the high order 8 bits of MCDRREG are used. You can
print out the current values of dcr_save and mcr_save with:
$ adb /unix /dev/kmem
dcr_save/x
mcr_save/x
^D

Well, that should do that for a while (until the flames roll in :)

>
>2. Does anyone know the specs for the WD1010 controller chip?
Ahhhhhhhhhhhhhh yup.

>                                                                 In
>   particular, I have been told by a tech at Seagate that the ST-4096
>   will recal if step pulses are spaced more than 7ns apart.  Does the
>   controller chip meet this requirement?
If your drive can not handle slow seeks, its junk. In this case I think you have
bad information. It is possible that the ST-4096 has a bad resonance problem
around 7ms. Our driver tells the WD1010 to seek the drives as fast as it can
all the time. From <sys/space.h>:------------+
struct gdsw gdsw[] = {                      V
        {       0,{"WIN", 1023, 4, 17, 68,0, 0, 512},
                -1,0,16, 1023*16,40, ghdintr, ghdstart,
                HDMAXCYL, HDMAXBADBLK, gdhdbbq, gdhdbb,0},

The zero here means a 35us seek time. This leaves it up to the drive to seek
however fast it can. The WD1010 then wait for the drive to signal when it has
completed the seek. This is fantastic for voice coil drives. When a recal is
done when there's an error, the driver sends a RESTORE command. This is also
done at power up. With a restore command, the WD1010 waits for the drive to
signal seek complete BETWEEN EVERY STEP! This is why some drives are so very
noisy when the recal. The head has just stoped, the drive said it's done, then
the WD1010 tells it to start moving again! GO STOP GO STOP GO STOP...

>3. Is there anyone who has a ST-4096 in their UNIXpc and is having no
>   difficulty?
Try jan@bagend. I have heard of others.
```

Lastly, if you want to convert that nasty HDERR to a block number so you
can use the diagnostics to lock it out:

```
----
HDERR ST:51 EF:40 CL:4260 CH:4201 SN:420C SC:4202 SDH:4222 DMACNT:FFFF DCRREG:92 MCRREG:9F00 Sat Jun
11 14:43:05 1988

HDERR ST:51 EF:40 CL:4260 CH:4201 SN:420C SC:4202 SDH:4222 DMACNT:FFFF DCRREG:92 MCRREG:9F00 Sat Jun
11 14:43:05 1988

drv:0 part:2 blk:20294 rpts:2 Sat Jun 11 14:43:06 1988
----
```

This is a real example from my unix.log. Let's run down everything here.

ST=51:01010001: Ready, Seek complete, Error.
EF=40:01000000: CRC data field
CH=4201, CL=4260: Cylinder is $0160=256+96=352
SN=420C: Sector number is 12
SC=4202: Sector count for this transfer is 2. We don't know what the original
  count was, and we don't care. During a multiple sector transfer, the SN is
  incremented and the SC is decremented as each sector is transfered. This
  makes it easy to pause and retry in the middle of a multi-sector transfer.
  The bottom line is that the SN is accurate.
SDH=4222:[0 01 00 010]: Extension off (good), Sector size=01 (512, good),
  Drive=0, no surprise for this machine, Head=010=2 (the third head), no
  surprise as this is a 9 head drive.
DMACNT:FFFF, Hmm. The DMA_ERROR bit is on. I don't know how to interpret the
  rest of the count register when there is an error. I don't think it matters.
DCRREG:92:10010010: FDRST* not asserted, FDDRIVE0* is asserted, FDMOTOR* is
  asserted, HDRST* is not asserted, DDRIVE0* is asserted, HDSELx=010=2=3rd.
  Well, this is interesting. It looks like the floppy drive was on when this
  hard drive error happened! The HDSELx lines DO correspond with the SDH reg,
  which is good.
MCRREG:9F00:10011111: CLSINT, DMA_READ=0=disk DMA write, LPSTB, MCKSEL, and
  now for the important one: ooh! aah! all four LEDs were off! (so what)

```
drv:0       It was the first hard drive
part:2      It was the file system partition (I only have one)
blk:20294   See below
rpts:2      This matches, there were two HDERR lines
Sat Jun 11 14:43:06 1988: Ok, so I was goofing with the floppy drive then.
```

I picked this case out of my unix.log because it DID have this gd line. Most of
my bunches of HDERR lines DO NOT have them. As you can also see the high order
8 bits of the WD1010 registers were 42 in this example, and they were FF in the
first example, like I said phase of the moon.

Now for the fun part. Lets map this cryptic shit back to the real world.

I've got two flavors, depending on how you like to think:

```
sector = ((((CH%256)*256+(CL%256))*HEADS)+(SDH%8)*SECTORS_PER_TRACK)+(SN%256)
```

or

```
sector = (((((CH&0xff)<<8)+(CL&0xff))*HEADS)+(SDH&0x07))<<4)+(SN&0xff)
```

Our "blocks" are 1024 or 2 sectors, so the block=sector/2. There are 16 data
sectors, or 8 blocks per track.

```
  (((((4201&0xff)<<8)+(4260&0xff))*9)+(4222&0x07))<<4)+(420C&0xff)
  ((256+96)*9+2)*16+12
  (392*9+2)*16+12
  (3168+2)*16+12
  3170*16+12
  50720+12
  50732  This is the sector number from the beginning of /dev/rfp000
  25366  This is the block offset
-    72  Size of my bad block table
-  5000  Size of my swap partition
=======
  20294  What do you know! it matches!
```

Now for even more fun! If this had been a recent HDERR message, I would now
run that neato-wiz-bang "bf" program Brant Cheikes just posted:

```
$ bf /dev/fp002 20294
block 20294 inode 8853

$ ncheck -i 8853
/dev/fp002:
8853    /usr/spool/news/comp/os/minix/1594
```

Ahh! Wouldn't you know it! I've got news stomping on my soft blocks!

I did not obtain permission from anyone for use of the information contained
in this article, so there. Western Digital does have good terse documentation,
just the way I like it: guts and no fluff.

Update on the second hard drive board: The board layout looks like it'll fly,
except for some cosmetic stuff. I got pricing from Saturn Electronics in MI:

2.5" x 3.0", 312 holes, no solder mask, no silk screen, double sided, plated
through holes, all holes one bit size.

```
 $200.00   Prototype quantity (about 10), including setup
 $100.00   Setup fee
   $3.75   Per board,   50 qty. ( $187.50)
   $3.00   Per board,  100 qty. ( $300.00)
   $2.76   Per board,  250 qty. ( $690.00)
   $2.09   Per board, 1000 qty. ($2090.00)
   $1.76   Per board, 5000 qty. ($8800.00)
```

If someone knows of a better price, call me. If you like these prices, call
me, and I'll give you the number of the local rep. I am in no way associated
with Saturn, I just heard of them by word of mouth.

John

Newsgroups: comp.sys.att,unix-pc.general
Subject: Re: Which chip(s) is(are) bad?; HwNote14
Keywords: HwNote bad memory
Message-ID: <593@uncle.UUCP>
Date: 15 Aug 89 22:34:27 GMT
References: <1648@naucse.UUCP>
Reply-To: jbm@uncle.UUCP (John B. Milton)
Organization: U.N.C.L.E.
Lines: 181
Xref: uncle comp.sys.att:6497 unix-pc.general:3928

In article <1648@naucse.UUCP> sbw@naucse.UUCP (Steve Wampler) writes:
[how do you map the /usr/adm/unix.log NMI parity errors to memory chips?]

>address (what does *hpte mean?) given in the error message, nor an
Hardware Page Table Entry, the physical memory mapping RAM.

```
>NMI (parity error) at 0x2FF2E6 (*hpte: 0xE24D) Tue Aug  8 17:25:24 1989
>NMI (parity error) at 0x2FF2E6 (*hpte: 0xE24D) Tue Aug  8 17:25:25 1989
>NMI (parity error) at 0x81000 (*hpte: 0x4251) Tue Aug  8 18:00:01 1989
```

```
>NMI (parity error) at 0x2FF2E6 (*hpte: 0xE24B) Fri Aug 11 04:25:27 1989
>NMI (parity error) at 0x2FF2E6 (*hpte: 0xC24B) Fri Aug 11 04:25:29 1989
>NMI (parity error) at 0x301004 (*hpte: 0xC22F) Fri Aug 11 05:26:30 1989
>NMI (parity error) at 0x301004 (*hpte: 0xC22F) Fri Aug 11 05:26:30 1989
>NMI (parity error) at 0x81000 (*hpte: 0x4238) Fri Aug 11 06:04:45 1989
```

The first number is the virtual address, the number your program sees. The *hpte means, not the address of the mapping RAM used, but the contents of the mapping RAM for the virtual page in question. The UNIXpc page size is 4k (2^12). The virtual address is useless in tracking down the memory chip. The RAM which does virtual to physical mapping on the UNIXpc is 16 bits wide. During a virtual memory cycle, what goes into the mapping RAM is address bits A12 to A21. What comes out are mapped address bit MA12 to MA21. The MA12 to MA21 together with A1 to A11 (and UDS/LDS) lines form the actual physical address used to access memory.

Ok, that's 10 bits for address mapping. That means that 1024 pages of 4k each can be mapped by the mapping hardware, for a total of 4M. This is the unusual part. The UNIXpc has a maximum physical memory limit of 4M, which is exactly the same as the maximum virtual memory limit. On most machines with virtual memory, the virtual space is quite large, typically 4G. On these machines, the maximum physical memory limit is far smaller than the maximum virtual limit. Ok, 10 bits, that leave 6 page status bits, PS0 to PS4 and WE. Only two (PS0 and PS1) of the page status bits are currently used. Hmmm. This is the encoding of PS0 and PS1:

PS0	PS1	Status of page
0	0	Page is not present (or not allowed)
0	1	Page is present, but has not been accessed
1	0	Page has been read but not written (clean)
1	1	Page has been written to (dirty)

These page status bits are what makes virtual memory work. They are automatically updated by hardware during each memory access. When an access is made to page that is not present, a fault is generated. It is then up to the UNIX kernel to decided whether the process is really allowed to access that page or not, and what to do about it. When it is time for a page belonging to a process to be removed from memory, the page status bits are checked to see what has happened to the page. If the page has been written to, it is assumed that the page was changed and is now different than the page in /dev/swap. It must therefor, be written to disk. If the page was read but not written, then the version on disk is the same, so no write is needed.

This is the mapping of the bits in the mapping RAM when read/written:

Bit	Desc	Bit	Desc	Bit	Desc	Bit	Desc
0	MA12	1	MA13	2	MA14	3	MA15
4	MA16	5	MA17	6	MA18	7	MA19
8	MA20	9	MA21	10	PS4	11	PS3
12	PS2	13	PS0	14	PS1	15	WE

We can use the above table to decode the error messages to get a physical address. Remember that the low 12 bits of the physical address are not mapped, so they are the same as the virtual address. In the first case from above, The virtual address is 0x2FF2E6, and the contents of the mapping RAM for that virtual address is 0xE24D, so the physical address is

```
(0x2FF2E6 & 0xFFF) + ((0xE24D & 0x3FF) << 12)
(0x2E6) + (24D000)
0x24D2E6
```

Well, that was the easy part! Now for the more difficult part of finding which memory chip that is. All the memory for the UNIXpc produced by Convergent, AT&T and others has all consisted of 64k or 256k dynamic memory chips, organized as

64k by 1 or 256k by 1. All memory for the UNIXpc MUST have a valid parity bit, because the UNIXpc has a scheme to TEST the parity bits. The CPU chip used in the UNIXpc is the Motorola 68010, which has a 16 bit data bus. The 68010 can also access, for read or write, wither the odd or even byte exclusively. The UNIXpc thus has memory which is organized in groups of 18 chips, 16 for data and two for parity. The trouble with transient (one shot) parity errors when running UNIX is that you can't tell whether it was the even, the odd, or both bytes with the given information.

The first thing we can tell right off the bat is that this memory location is not on the mother board. Internal memory addresses range from 0x000000 to 0x1FFFFF, and external from 0x200000 to 0x3FFFFF. Here is a sorted, uniqed table of the addresses:

Virtual	*hpte	Physical
0x301004	0xC22F	0x22F004
0x081000	0x4238	0x238000
0x2FF2E6	0xE24B	0x24B2E6
0x2FF2E6	0xE24D	0x24D2E6
0x081000	0x4251	0x251000

Hmm. They all seem fairly well clustered together. In this case we know that the expansion memory is on a Combo card. The Combo card has three sets of 18 chips, with these addresses:

Chip coords.	Address range
5A to 6K	0x200000 to 0x27FFFF
7A to 8K	0x280000 to 0x2FFFFF
9A to 10K	0x300000 to 0x37FFFF

All the addresses I got from the error messages range from 0x22F004 to 0x251000. So, given the slim information we got from the kernel, it's the first bank. Without further info, there's no way to tell which of the 18 chips may be bad. Note that I say MAY. There is no guarantee that there is anything wrong with all the memory on the board. The simple addition of the board to the system may have put the power requirements over what the power supply can handle. It may be that one (or more) of the chips in the first bank of the Combo card was the first to show the signs of insufficient power. Note that expansion cards will be the first to show signs of low voltage, since they are furthest away from the power connector. One way to get around the power problem is to clean the contact on the mother board where the power supply ribbon cable plugs in. Just pull it off and put it on a couple of times, making sure it's all the way on the last time.

The next thing I would try is to run the memory test on the diagnostics disk all night long to try to catch the bad location. You will then run into one of the more annoying things about the memory diagnostics, you can't test a sub-section of memory, you have to test all of it. This is especially bad when you have 2M on the mother board and you're looking for problems with expansion memory.

If the diagnostics can catch a memory problem, you will get a lot more information. As far as memory on the Combo boards goes, the boards were layed out so they could be repaired easily. Row K is all parity chips, alternating low high. Column A to K is D0 to D7 for odd (5, 7, 9) or D8 to D15 for even. So, if the diagnostics say bad parity chip, 0x293475 high, that would be the second bank (7A to 8K), parity (row K), high (column 8): 8K.

The layout on the mother board is much the same, the major difference being in what kind of memory chips are on you mother board. If a memory chip completely fails in the first bank of memory on the mother board, UNIX will panic and you won't be able to boot anything. What was that repair center number again?

The mother board memory is organized much the same as the Combo board. Mother board memory goes from 2A (front right) to 10H (back left). Column 10 is all parity chips, alternating low high, front to back. Column 2 to 9 is D0 to D7 for rows A, C, E, G or D8 to D15 for rows B, D, F, H.

```
Chip coords.    Address range (64k)     Address range (256k)
2G to 10H       0x000000 to 0x01FFFF    0x000000 to 0x07FFFF
2E to 10F       0x020000 to 0x03FFFF    0x080000 to 0x0FFFFF
2C to 10D       0x040000 to 0x05FFFF    0x100000 to 0x17FFFF
2A to 10B       0x060000 to 0x07FFFF    0x180000 to 0x1FFFFF
```

For example, 0x056789 would be one of the 18 chips at 2C to 10D if you have a 512k machine fully populated with 64k RAM chips, or one of the 18 chips at 2G to 10H if you have a half populated 1M mother board or a 2M mother board.

Well, I've been working from schematics up to now, let me take a guess at the 512k/2M AT&T RAM Expansion card.

**** WARNING THIS IS A GUESS ****

Row K parity, odd columns low, even columns high.

```
Chip coords.    Address range (64k)     Address range (256k)
 5A to  6K      0x000000 to 0x01FFFF    0x000000 to 0x07FFFF
 7A to  8K      0x020000 to 0x03FFFF    0x080000 to 0x0FFFFF
 9A to 10K      0x040000 to 0x05FFFF    0x100000 to 0x17FFFF
11A to 12K      0x060000 to 0x07FFFF    0x180000 to 0x1FFFFF
```

Well finding a bad memory chip and replacing it are two different things. If you have the skill and go after you mother board, PUT SOCKETS IN when you start putting things back together. If you have bad 64k chips on a 512k mother board, consider replacing ALL the chips and doing an upgrade to 2M. If you have a Combo board (read sockets), replace the offending bank with new chips. If you can't, swap the chips in the bad bank with another bank (assuming you have more than one bank installed), then see if the memory problems move.

The continuity may be a little bad on this one folks; I've got a cold and this was done in three sessions.

John
--
John Bly Milton IV, jbm@uncle.UUCP, n8emr!uncle!jbm@osu-cis.cis.ohio-state.edu
(614) h:294-4823, w:785-1110; N8KSN, AMPR: 44.70.0.52; Don't FLAME, inform!

Newsgroups: unix-pc.general,comp.sys.att
From: jbm@uncle.UUCP (John B. Milton)
Subject: Wait states, why only 4M, 68012, LEDs ; HwNote15
Date: 28 Nov 89 05:59:29 GMT

"Does expansion memory run slower than motherboard memory?"

Earlier I said that external memory has an extra wait state, making it slower than internal memory. I was mearly repeating what someone else had told me. After some research, I find that it is not a yes/no answer, that is, some machine do and some don't. On page 2-23 of the Reference manual, at the top, and on 2-26 (same text), there is a statement "19F, acting as a MUX, is set up to select the A inputs because we are not in expansion memory...". According to the schematics (both versions), the select pin, pin 1 is grounded and the B inputs are NC. Hmm one says waits, one say no wait states.

I have examined 3 different motherboard vintages and found:
1. That very old motherboards (with the piggy-back board instead of custom chips) do have a 74258 at 19F, and have patch wires on the B inputs, with pin 1 (A/B select) not grounded. It appears to implement expansion memory wait states.
2. Semi-old motherboards (most of the 1M) have a 74258 at 19F, with the select

```
   pin grounded, and no patch wires on the B inputs. In this case, it appears
   that expansion memory wait states have not been enabled.
3. Very new motherboards (mostly 2M 3b1) do not even have a 74258 at 19F. The
   expansion memory wait state circuit was designed out.

My guess at all this is that the DMA prototype (piggy-back board) was not
quiet fast enough to access expansion memory with no wait states. To get
around this, they designed in the 74258 to add a wait state to expansion
memory accesses. Machines went out with the piggy-back board and the
74258 patched for expanion wait states. Later, when the custom chips were
ready, a lot of machines went out with the 74258 just taking up power and time.
When the motherboard layout was re-done, the 74258 was dropped.

What all this means is that, if you have an old machine with the piggy-back
board and expansion memory, you can get a performance increase by switching
to a newer motherboard. Well, I think that horse is dead.
---

        "Well, if the UNIXpc only has 4M for RAM and the 68010 can
        address 16M, where did the other 12M go and why can't we use it?"

What follows is a brief description of where things are in the UNIXpc address
space. This should give you an idea of how sloppy the decoding is. Sloppy
decoding is not as bad as it might seem, as it's easier to implement and
faster. In the address below, "x" means "don't care" or any value. All the
addresses are in hex, so each "x" is 16. Multiply the "x"s together to get
the size of the space each I/O register or device takes up. For descriptions
of the bits in the I/O registers, see the appropriate /usr/include/sys/*.h
files. The notation [1236-8] means 1, 2, 3, 6, 7, 8. For example,
"xx[ef][7f]xxx[13-f]" would expand to the BIT pattern:
"xxxx xxxx 111x x111 xxxx xxxx xxxx 111x"

UNIXpc memory map:

The UNIXpc memory is divided into 4 major 4 megabyte chunks:
I       xx000000 to xx3fffff    RAM memory, fast cycle access (400ns)
II      xx400000 to xx7fffff    fast cycle I/O
III     xx800000 to xxbfffff    slow cycle read (appears to also be writable)
IV      xxc00000 to xxffffff    slow cycle I/O



I
xx000000 to xx1fffff    Internal mappable RAM (400ns cycle)
xx200000 to xx3fffff    External mappable RAM (400ns cycle (or slower))

II
xx[4-7]0x[08]00 to xx40x[7f]ff Map RAM, 2k
xx[4-7]1xxxx            General Status Register
xx[4-7]20000 to xx427fff Video RAM, 32k (31320 on screen)
xx[4-7]3xxxx            Bus Status Register 0
xx[4-7]4xxxx            Bus Status Register 1
xx[4-7]5xxxx            Phone status
xx[4-7]6xxxx            DMA count register
xx[4-7]7xxxx            Line printer status register
xx[4-7]8xxxx            Real Time Clock
xx[4-7]9[08]xxx         Handset relay
xx[4-7]9[19]xxx         Line select 2
xx[4-7]9[2a]xxx         Hook relay 1
xx[4-7]9[3b]xxx         Hook relay 2
xx[4-7]9[4c]xxx         Line 1 hold
xx[4-7]9[5d]xxx         Line 2 hold
xx[4-7]9[6e]xxx         Line 1 A-lead
xx[4-7]9[7f]xxx         Line 2 A-lead
xx[4-7]axxxx            Miscellaneous Control Register
xx[4-7]bxxxx            TM/DIALWR
```

```
xx[4-7]cxxxx           CSR
xx[4-7]dxxxx           DMA, Address register
xx[4-7]exxxx           Disk Control Register
xx[4-7]fxxxx           Line printer data register

II
xx800000 to xxbfffff    Boot ROM, temporarily located at xx000000 during RESET

III
xxc00000                Expansion slot 0 I/O
xxc40000                Expansion slot 1 I/O
xxc80000                Expansion slot 2 I/O
xxcc0000                Expansion slot 3 I/O
xxd00000                Expansion slot 4 I/O
xxd40000                Expansion slot 5 I/O
xxd80000                Expansion slot 6 I/O
xxdc0000                Expansion slot 7 I/O

xx[ef][08]xxx0          WD1010 Data register
xx[ef][08]xxx2          WD1010 Error register
xx[ef][08]xxx4          WD1010 Count register
xx[ef][08]xxx6          WD1010 Sector number register
xx[ef][08]xxx8          WD1010 Cylinder number low register
xx[ef][08]xxxa          WD1010 Cylinder number high register
xx[ef][08]xxxc          WD1010 Sector drive head register
xx[ef][08]xxxe          WD1010 Status/Command register
xx[ef][08]xxxf          (ignored)

xx[ef][19]xxx0          WD2797 Status/Command register
xx[ef][19]xxx2          WD2797 Track register
xx[ef][19]xxx4          WD2797 Sector register
xx[ef][19]xxx6          WD2797 Data register
xx[ef][19]xxx[13578-f]  (unused)

xx[ef][2a]xxxx          Miscellaneous Control Register 2 (used with P5.1)

xx[ef][3b]xxxx          Real Time Clock data bits

General Control Register
xx[ef][4c][08]xxx       EE
xx[ef][4c][19]xxx       P1E
xx[ef][4c][2a]xxx       BP
xx[ef][4c][3b]xxx       ROMLMAP
xx[ef][4c][4c]xxx       L1 MODEM
xx[ef][4c][5d]xxx       L2 MODEM
xx[ef][4c][6e]xxx       D/N CONNECT
xx[ef][4c][7f]xxx       Whole screen reverse video

xx[ef][5d]xxx0          8274, Ch A data
xx[ef][5d]xxx2          8274, Ch B data
xx[ef][5d]xxx4          8274, Ch A status/command
xx[ef][5d]xxx6          8274, Ch B status/command
xx[ef][5d]xxx[1357-f]   (unused)
xx[ef][6e]0xxx          Line control
xx[ef][6e]3xxx          Relay and lamp drivers
xx[ef][6e]4xxx          Options A/S and handshake
xx[ef][6e]5xxx          Options CCITT and disconnect
xx[ef][6e]6xxx          RD, SD and chip test
xx[ef][6e]8xxx          Transceiver control 1
xx[ef][6e]9xxx          Transceiver control 2
xx[ef][6e]axxx          Transceiver status
xx[ef][6e][b-f]xxx      (undefined)

xx[ef][7f]xxx0          6850 (keyboard) Status register
xx[ef][7f]xxx2          6850 (keyboard) Data register
```

```
xx[ef][7f]xxx[13-f]     (unused)
```

Note that every "xxx" above is 4k, and every "xxxx" is 64k of address space. There are some very interesting things to note here. One very interesting thing is that the 16k of boot ROM takes up an entire 4M chunk! Note that this is up in the slow cycle half of memory (1000ns), and is not refreshed. There does not seem to be circuitry to restrict writes to this area. This area could be replaced with a ROM-disk, with the boot ROM in the first partition. Since EPROM devices are faster now than when the UNIXpc was designed, the slow cycle access could be defeated. Note that all of the I/O in the second 4M chunk (xx400000 to xx7fffff) could fit in xx400000 to xx4fffff if A20 and A21 were used. This would free up 3M of fast cycle space.
--

"I've heard of the 68000, 68008, 68010, 68020, 68030, and the 68040, but what is the 68012?"

Well, this one got me thinking the other night. What, you've never heard of the 68012? If you've got the data sheet for the 68010, it's right there on the cover: "MC68010/MC68012 16-/32-Bit Virtual Memory Microprocessors". The 68012 is EXACTLY the same as the 68010 from a software point of view. The only difference between the 68012 and the 68010 is that the 68012 has 8 additional pins: A24, A25, A26, A27, A28, A29, A31 and RMC. That's right A30 *IS NOT* provided. This allows the 68012 to access 2G of memory, in two separate 1G areas from 00000000 to 3fffffff (same as 40000000 to 7fffffff) and
from   80000000 to dfffffff (same as c0000000 to ffffffff). The 68012 is only available in an 84 pin grid array package. The RMC pin is the same as the 68020 pin by the same name, and is asserted during Read-Modify-Write bus cycles, like those produced by the TAS instruction. Word has it that chip in the 68012 is exactly the same as the 68010, it just has more wires bonded to the die. Word also has it that the chip was specially produced for one customer. Why they couldn't run out A30 when they added so many GND pins is beyond me.

All this is fine and wonderful, but there are problems. Memory above the 16M boundry would have to be limited to supervisor mode (kernel space), since there is no MMU access. Exceptions would be applications like the "vidpal". If the upper byte of any of the address registers used by the kernel is EVER set to non-zero, then this approach will be tougher to use. Since the UNIXpc mother-board and expansion bus don't support the additional address lines, the only place to use them would be on a daughter board.
--

"When the system won't boot, don't the LEDs on the side indicate the problem?"

When the machine is first started, the boot ROM goes though some tests before it tries to boot. If one of these tests fails, the machine halts with the number of the test on the LEDs. The LEDs are behind the gates on the left side of the machine, near the front.

```
LED 1 Red
LED 2 Green
LED 3 Yellow
LED 4 Red (I guess they ran out of colors)

Test  4    3    2    1  Problem
  1  off  off  off   on Failed telephone initialization
  2  off  off   on  off Failed video RAM test
  3  off  off   on   on Failed map RAM test
  4  off   on  off  off Failed to set map RAM to unity map
  5  off   on  off   on Failed dynamic RAM test
  6  off   on   on  off Failed initialization
  7  off   on   on   on Failed to find loader on disk
```

I guess I should have put this in HwNote01.

```
John
--
John Bly Milton IV, jbm@uncle.UUCP, n8emr!uncle!jbm@osu-cis.cis.ohio-state.edu
(614) h:252-8544, w:469-1990; N8KSN, AMPR: 44.70.0.52; Don't FLAME, inform!
```