

Re: [PATCH 09/12] i386/x86_64: EHCI usb debug port early printk support.

Re: [PATCH 09/12] i386/x86_64: EHCI usb debug port early printk support.

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-04/msg12233.html>

- *From:* ebiederm@xxxxxxxxxxxxx (Eric W. Biederman)
 - *Date:* Mon, 30 Apr 2007 14:54:18 -0600
-

Andi Kleen <ak@xxxxxxx> writes:

Thanks for writing that code. It should be an interesting alternative on boxes where firescope doesn't work.

I hope I can eventually merge early firewire support code too.

On Mon, Apr 30, 2007 at 10:32:02AM -0600, Eric W. Biederman wrote:

With legacy free systems serial ports have stopped being an option to get early boot traces and other debug information out of a machine.

This needs a CONFIG_* at least. And some documentation on how to set it up on both sides.

Besides CONFIG_EARLY_PRINTK I assume. It is enough code so there is a case for it.

This debug device can be used to replace serial ports for kgdb, kdb, and console support. And gregkh has a simple usb serial driver for it so user space applications that control serial ports should work unmodified.

But not merged yet, right? I was hoping it could be done from user space anyways.

Sorry old comment, that piece has been merged for a while. It is the usb_debug module. It creates a tty device that you can just cat to get the usb debug output.

Re: [PATCH 09/12] i386/x86_64: EHCI usb debug port early printk support.

Re: [PATCH 09/12] i386/x86_64: EHCI usb debug port early printk support.

For users the hard part looks like it will be finding cables and finding which is usb debug port 1 and realizing that there is flow control so the kernel boot will not happen if someone is not reading the serial console data.

That's nasty. Any way to work around that?

Maybe. It has been long enough since I wrote the code I need to go back and look.

```
index 92213d2..dc097aa 100644
--- a/arch/x86_64/kernel/early_printk.c
+++ b/arch/x86_64/kernel/early_printk.c
@@ -3,9 +3,19 @@
#include <linux/init.h>
#include <linux/string.h>
#include <linux/screen_info.h>
+#include <linux/usb/ch9.h>
+#include <linux/pci_regs.h>
+#include <linux/pci_ids.h>
+#include <linux/errno.h>
```

Can you put it in a separate file please?
Perhaps with a little abstraction in drivers/usb ?

```
+static void dbgp_breath(void)
+{
+ /* Sleep to give the debug port a chance to breathe */
```

But you don't?

Good point. At least this early I'm not certain there is any way I can productively do that. This is before we have calibrated the tsc's and the like so timeouts are difficult, and as I recall our default guess isn't.

This lack of a good timeout looks to be the reason ehci_wait_for_port doesn't timeout in a timely fashion because I don't timeout until I have wrapped a 32bit number.

Re: [PATCH 09/12] i386/x86_64: EHCI usb debug port early printk support.

```
+static __u32 __init find_dbgp(int ehci_num, unsigned *rbus, unsigned  
*rslot,
```

```
unsigned *rfunc)
```

This should be probably merged into the early quirks loop

```
early_console = &simnow_console;  
keep_early = 1;  
+ } else if (!strncmp(buf, "dbgp", 4)) {
```

usb would seem to be more intuitive

Could be. I was thinking usb debug port. dbgp is at least unique, and unfortunately it doesn't look like any old usb cable will do, so a straight usb I expect would be very misleading.

The truth is I don't have a big need for this. I put it together as a proof of concept to see how hard it would be, etc. I can clean it up a little but I'm really hoping I can get this into one of the development trees and people who have more use for it then I do can play with it and improve things.

One persons experience on two machines probably isn't quite enough of a sample to Document how to use this. At least beyond what I did in my changelog.

Eric

-

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxx
More majordomo info at <http://vger.kernel.org/majordomo-info.html>
Please read the FAQ at <http://www.tux.org/lkml/>