

## Re: Exar ST16C2550 rev A2 bug

**Source:** <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-11/8000.html>

---

**From:** Russell King ([rmk+lkml\\_at\\_arm.linux.org.uk](mailto:rmk+lkml_at_arm.linux.org.uk))

**Date:** 11/28/04

Date: Sun, 28 Nov 2004 11:10:47 +0000  
To: Alex Williamson <[alex.williamson@hp.com](mailto:alex.williamson@hp.com)>

On Wed, Nov 17, 2004 at 11:26:47AM -0700, Alex Williamson wrote:

> *There seem to be an increasing number of the above UARTs floating*  
> *around and I'm wondering if we can do something to better detect and*  
> *work around their flaw. Exar has documented the problem and their*  
> *proposed serial driver changes to work around the issue here:*  
>  
> [http://www.exar.com/info.php?pdf=dan180\\_oct2004.pdf](http://www.exar.com/info.php?pdf=dan180_oct2004.pdf)

Can you check whether this patch solves it? I'd rather not rely on `size_fifo()` to do the right thing in every circumstance.

Essentially, if we find an EFR, we check whether the UART reports DVID/DREV values corresponding to the known problem scenario, and if so, we essentially ignore the EFR.

What I don't know is whether these DVID/DREV values correspond to a real device which does have an EFR. Maybe Exar people can shed some light on this?

(Also, one has to wonder what information Exar has about what we're working on, which we don't know ourselves about... check out the above link, FAQ question 6. 8))

```
===== drivers/serial/8250.c 1.92 vs edited =====
--- 1.92/drivers/serial/8250.c 2004-11-19 07:03:10 +00:00
+++ edited/drivers/serial/8250.c 2004-11-28 11:02:32 +00:00
@@ -479,6 +479,34 @@
 }

/*
+ * Read UART ID using the divisor method - set DLL and DLM to zero
+ * and the revision will be in DLL and device type in DLM. We
+ * preserve the device state across this.
+ */
+static unsigned int autoconfig_read_divisor_id(struct uart_8250_port *p)
+{
+ unsigned char old_dll, old_dlm, old_lcr;
```

```

+ unsigned int id;
+
+ old_lcr = serial_inp(p, UART_LCR);
+ serial_outp(p, UART_LCR, UART_LCR_DLAB);
+
+ old_dll = serial_inp(p, UART_DLL);
+ old_dlm = serial_inp(p, UART_DLM);
+
+ serial_outp(p, UART_DLL, 0);
+ serial_outp(p, UART_DLM, 0);
+
+ id = serial_inp(p, UART_DLL) | serial_inp(p, UART_DLM) << 8;
+
+ serial_outp(p, UART_DLL, old_dll);
+ serial_outp(p, UART_DLM, old_dlm);
+ serial_outp(p, UART_LCR, old_lcr);
+
+ return id;
+}
+
+/*
+ * This is a helper routine to autodetect StarTech/Exar/Oxsemi UART's.
+ * When this function is called we know it is at least a StarTech
+ * 16650 V2, but it might be one of several StarTech UARTs, or one of
+ @@ -490,7 +518,7 @@
+ */
static void autoconfig_has_efr(struct uart_8250_port *up)
{
- unsigned char id1, id2, id3, rev, saved_dll, saved_dlm;
+ unsigned int id1, id2, id3, rev;

    /*
     * Everything with an EFR has SLEEP
     @@ -540,21 +568,13 @@
     * 0x12 - XR16C2850.
     * 0x14 - XR16C854.
     */
- serial_outp(up, UART_LCR, UART_LCR_DLAB);
- saved_dll = serial_inp(up, UART_DLL);
- saved_dlm = serial_inp(up, UART_DLM);
- serial_outp(up, UART_DLL, 0);
- serial_outp(up, UART_DLM, 0);
- id2 = serial_inp(up, UART_DLL);
- id1 = serial_inp(up, UART_DLM);
- serial_outp(up, UART_DLL, saved_dll);
- serial_outp(up, UART_DLM, saved_dlm);
-
- DEBUG_AUTOCONF("850id=%02x:%02x ", id1, id2);
-
- if (id1 == 0x10 || id1 == 0x12 || id1 == 0x14) {
- if (id1 == 0x10)

```

