

## [PATCH] esp: Make driver SMP-correct

**Source:** <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-12/6345.html>

---

**From:** James Nelson ([james4765\\_at\\_verizon.net](mailto:james4765_at_verizon.net))

**Date:** 12/31/04

To: [kernel-janitors@lists.osdl.org](mailto:kernel-janitors@lists.osdl.org), [linux-kernel@vger.kernel.org](mailto:linux-kernel@vger.kernel.org)

Date: Thu, 30 Dec 2004 19:43:44 -0600

This is an attempt to make the esp serial driver SMP-correct. It also removes some cruft left over from the serial\_write() conversion.

Diffstat output:

```
drivers/char/Kconfig | 2
drivers/char/esp.c | 299 ++++++-----
include/linux/hayesesp.h | 1
3 files changed, 160 insertions(+), 142 deletions(-)
```

Signed-off-by: James Nelson <[james4765@gmail.com](mailto:james4765@gmail.com)>

```
diff -urN --exclude='*~' linux-2.6.10-original/drivers/char/esp.c linux-2.6.10/drivers/char/esp.c
--- linux-2.6.10-original/drivers/char/esp.c 2004-12-24 16:33:48.000000000 -0500
+++ linux-2.6.10/drivers/char/esp.c 2004-12-30 20:33:46.243770922 -0500
@@ -152,18 +152,6 @@
```

```
/* Standard COM flags (except for COM4, because of the 8514 problem) */
#define STD_COM_FLAGS (ASYNC_BOOT_AUTOCONF | ASYNC_SKIP_TEST)
```

```
_/
- * tmp_buf is used as a temporary buffer by serial_write. We need to
- * lock it in case the memcopy_fromfs blocks while swapping in a page,
- * and some other program tries to do a serial write at the same time.
- * Since the lock will only come under contention when the system is
- * swapping and available memory is low, it makes sense to share one
- * buffer across all the serial ports, since it significantly saves
- * memory if large numbers of serial ports are open.
- */
```

```
-static unsigned char *tmp_buf;
-static DECLARE_MUTEX(tmp_buf_sem);
```

```
-
static inline int serial_paranoia_check(struct esp_struct *info,
char *name, const char *routine)
```

```
{
@@ -209,34 +197,38 @@
struct esp_struct *info = (struct esp_struct *)tty->driver_data;
unsigned long flags;
```

## Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```
- if (serial_paranoia_check(info, tty->name, "rs_stop"))
- return;
+ spin_lock_irqsave(&info->irq_lock, flags);
+
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
+ goto out;

- save_flags(flags); cli();
  if (info->IER & UART_IER_THRI) {
    info->IER &= ~UART_IER_THRI;
    serial_out(info, UART_ESI_CMD1, ESI_SET_SRV_MASK);
    serial_out(info, UART_ESI_CMD2, info->IER);
  }

- restore_flags(flags);
+out:
+ spin_unlock_irqrestore(&info->irq_lock, flags);
}

static void rs_start(struct tty_struct *tty)
{
  struct esp_struct *info = (struct esp_struct *)tty->driver_data;
  unsigned long flags;

-
- if (serial_paranoia_check(info, tty->name, "rs_start"))
- return;
-
- save_flags(flags); cli();
+
+ spin_lock_irqsave(&info->irq_lock, flags);
+
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
+ goto out;
+
  if (info->xmit_cnt && info->xmit_buf && !(info->IER & UART_IER_THRI)) {
    info->IER |= UART_IER_THRI;
    serial_out(info, UART_ESI_CMD1, ESI_SET_SRV_MASK);
    serial_out(info, UART_ESI_CMD2, info->IER);
  }
- restore_flags(flags);
+out:
+ spin_unlock_irqrestore(&info->irq_lock, flags);
}

/*
@@ -311,7 +303,7 @@
  return;
}

- sti();
+ spin_unlock_irq(&info->irq_lock);
```

## Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```
status_mask = (info->read_status_mask >> 2) & 0x07;

@@ -329,7 +321,7 @@
    (serial_in(info, UART_ESI_RWS) >> 3) & status_mask;
}

- cli();
+ spin_lock_irq(&info->irq_lock);

    /* make sure everything is still ok since interrupts were enabled */
    tty = info->tty;
@@ -478,7 +470,7 @@
    info->xmit_tail = (info->xmit_tail + space_avail) &
        (ESP_XMIT_SIZE - 1);

- sti();
+ spin_unlock_irq(&info->irq_lock);

    for (i = 0; i < space_avail - 1; i += 2) {
        outw(*((unsigned short *) (pio_buf->data + i)),
@@ -489,7 +481,7 @@
        serial_out(info, UART_ESI_TX,
            pio_buf->data[space_avail - 1]);

- cli();
+ spin_lock_irq(&info->irq_lock);

        if (info->xmit_cnt) {
            serial_out(info, UART_ESI_CMD1, ESI_NO_COMMAND);
@@ -654,10 +646,10 @@
            err_status = 0;
            scratch = serial_in(info, UART_ESI_SID);

- cli();
+ spin_lock_irq(&info->irq_lock);

        if (!info->tty) {
- sti();
+ spin_unlock_irq(&info->irq_lock);
            return IRQ_NONE;
        }

@@ -740,7 +732,7 @@
#ifdef SERIAL_DEBUG_INTR
    printk("end.\n");
#endif
- sti();
+ spin_unlock_irq(&info->irq_lock);
    return IRQ_HANDLED;
}
```

## Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```
@@ -859,7 +851,7 @@
    int retval=0;
    unsigned int num_chars;

- save_flags(flags); cli();
+ spin_lock_irqsave(&info->irq_lock, flags);

    if (info->flags & ASYNC_INITIALIZED)
        goto out;
@@ -973,7 +965,7 @@

    info->flags |= ASYNC_INITIALIZED;
    retval = 0;
-out: restore_flags(flags);
+out: spin_unlock_irqrestore(&info->irq_lock, flags);
    return retval;
}

@@ -993,7 +985,7 @@
    info->irq);
#endif

- save_flags(flags); cli(); /* Disable interrupts */
+ spin_lock_irqsave(&info->irq_lock, flags); /* Disable interrupts */

    /*
     * clear delta_msr_wait queue to avoid mem leaks: we may free the irq
@@ -1058,7 +1050,7 @@
    set_bit(TTY_IO_ERROR, &info->tty->flags);

    info->flags &= ~ASYNC_INITIALIZED;
- restore_flags(flags);
+ spin_unlock_irqrestore(&info->irq_lock, flags);
}

/*
@@ -1172,7 +1164,7 @@
    if (I_IXOFF(info->tty))
        flow1 |= 0x81;

- save_flags(flags); cli();
+ spin_lock_irqsave(&info->irq_lock, flags);
    /* set baud */
    serial_out(info, UART_ESI_CMD1, ESI_SET_BAUD);
    serial_out(info, UART_ESI_CMD2, quot >> 8);
@@ -1219,7 +1211,7 @@
    serial_out(info, UART_ESI_CMD2, info->config.flow_on >> 8);
    serial_out(info, UART_ESI_CMD2, info->config.flow_on);

- restore_flags(flags);
```

## Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```
+ spin_unlock_irqrestore(&info->irq_lock, flags);
}

static void rs_put_char(struct tty_struct *tty, unsigned char ch)
@@ -1227,42 +1219,44 @@
    struct esp_struct *info = (struct esp_struct *)tty->driver_data;
    unsigned long flags;

- if (serial_paranoia_check(info, tty->name, "rs_put_char"))
- return;
+ spin_lock_irqsave(&info->irq_lock, flags);
+
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
+ goto out;

    if (!tty || !info->xmit_buf)
- return;
+ goto out;

- save_flags(flags); cli();
- if (info->xmit_cnt >= ESP_XMIT_SIZE - 1) {
- restore_flags(flags);
- return;
- }
+ if (info->xmit_cnt >= ESP_XMIT_SIZE - 1)
+ goto out;

    info->xmit_buf[info->xmit_head++] = ch;
    info->xmit_head &= ESP_XMIT_SIZE-1;
    info->xmit_cnt++;
- restore_flags(flags);
+out:
+ spin_unlock_irqrestore(&info->irq_lock, flags);
}

static void rs_flush_chars(struct tty_struct *tty)
{
    struct esp_struct *info = (struct esp_struct *)tty->driver_data;
    unsigned long flags;

-
- if (serial_paranoia_check(info, tty->name, "rs_flush_chars"))
- return;
+
+ spin_lock_irqsave(&info->irq_lock, flags);
+
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
+ goto out;

    if (info->xmit_cnt <= 0 || tty->stopped || !info->xmit_buf)
- return;
+ goto out;
```

## Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```
- save_flags(flags); cli();
  if (!(info->IER & UART_IER_THRI)) {
    info->IER |= UART_IER_THRI;
    serial_out(info, UART_ESI_CMD1, ESI_SET_SRV_MASK);
    serial_out(info, UART_ESI_CMD2, info->IER);
  }
- restore_flags(flags);
+out:
+ spin_unlock_irqrestore(&info->irq_lock, flags);
}

static int rs_write(struct tty_struct * tty,
@@ -1272,16 +1266,25 @@
    struct esp_struct *info = (struct esp_struct *)tty->driver_data;
    unsigned long flags;

- if (serial_paranoia_check(info, tty->name, "rs_write"))
- return 0;
+ spin_lock_irqsave(&info->irq_lock, flags);

- if (!tty || !info->xmit_buf || !tmp_buf)
- return 0;
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
+ goto out;
+
+ if (!tty || !info->xmit_buf)
+ goto out;

    while (1) {
        /* Thanks to R. Wolff for suggesting how to do this with */
        /* interrupts enabled */

+ /*
+ * Sorry about the interrupt-enabled trick, but it is not
+ * SMP-safe, since this is no longer protected by the 2.4-era
+ * write buffer semaphore. It needs to be done with the
+ * interrupt spinlock aquired - Jim Nelson
+ */
+
        c = count;
        t = ESP_XMIT_SIZE - info->xmit_cnt - 1;

@@ -1305,15 +1308,14 @@
        ret += c;
    }

- save_flags(flags); cli();
-
    if (info->xmit_cnt && !tty->stopped && !(info->IER & UART_IER_THRI)) {
        info->IER |= UART_IER_THRI;
```

Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

        serial_out(info, UART_ESI_CMD1, ESI_SET_SRV_MASK);
        serial_out(info, UART_ESI_CMD2, info->IER);
    }

- restore_flags(flags);
+out:
+ spin_unlock_irqrestore(&info->irq_lock, flags);
    return ret;
}

@@ -1322,7 +1324,7 @@
    struct esp_struct *info = (struct esp_struct *)tty->driver_data;
    int ret;

- if (serial_paranoia_check(info, tty->name, "rs_write_room"))
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
    return 0;
    ret = ESP_XMIT_SIZE - info->xmit_cnt - 1;
    if (ret < 0)
@@ -1334,7 +1336,7 @@
    {
        struct esp_struct *info = (struct esp_struct *)tty->driver_data;

- if (serial_paranoia_check(info, tty->name, "rs_chars_in_buffer"))
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
    return 0;
    return info->xmit_cnt;
}
@@ -1342,12 +1344,15 @@
static void rs_flush_buffer(struct tty_struct *tty)
{
    struct esp_struct *info = (struct esp_struct *)tty->driver_data;
-
- if (serial_paranoia_check(info, tty->name, "rs_flush_buffer"))
+ unsigned long flags;
+
+ spin_lock_irqsave(&info->irq_lock, flags);
+ if (serial_paranoia_check(info, tty->name, __FUNCTION__)) {
+ spin_unlock_irqrestore(&info->irq_lock, flags);
    return;
- cli();
+ }
    info->xmit_cnt = info->xmit_head = info->xmit_tail = 0;
- sti();
+ spin_unlock_irqrestore(&info->irq_lock, flags);
    tty_wakeup(tty);
}

@@ -1362,6 +1367,7 @@
static void rs_throttle(struct tty_struct * tty)
{

```

## Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```
    struct esp_struct *info = (struct esp_struct *)tty->driver_data;
+ unsigned long flags;
#ifdef SERIAL_DEBUG_THROTTLE
    char buf[64];

@@ -1369,38 +1375,43 @@
    tty->ldisc.chars_in_buffer(tty);
#endif

- if (serial_paranoia_check(info, tty->name, "rs_throttle"))
- return;
+ spin_lock_irqsave(&info->irq_lock, flags);
+
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
+ goto out;

- cli();
  info->IER &= ~UART_IER_RDI;
  serial_out(info, UART_ESI_CMD1, ESI_SET_SRV_MASK);
  serial_out(info, UART_ESI_CMD2, info->IER);
  serial_out(info, UART_ESI_CMD1, ESI_SET_RX_TIMEOUT);
  serial_out(info, UART_ESI_CMD2, 0x00);
- sti();
+out:
+ spin_unlock_irqrestore(&info->irq_lock, flags);
}

static void rs_unthrottle(struct tty_struct * tty)
{
    struct esp_struct *info = (struct esp_struct *)tty->driver_data;
+ unsigned long flags;
#ifdef SERIAL_DEBUG_THROTTLE
    char buf[64];

    printk("unthrottle %s: %d...\n", tty_name(tty, buf),
        tty->ldisc.chars_in_buffer(tty));
#endif
+
+ spin_lock_irqsave(&info->irq_lock, flags);

- if (serial_paranoia_check(info, tty->name, "rs_unthrottle"))
- return;
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
+ goto out;

- cli();
  info->IER |= UART_IER_RDI;
  serial_out(info, UART_ESI_CMD1, ESI_SET_SRV_MASK);
  serial_out(info, UART_ESI_CMD2, info->IER);
  serial_out(info, UART_ESI_CMD1, ESI_SET_RX_TIMEOUT);
  serial_out(info, UART_ESI_CMD2, info->config.rx_timeout);
```

## Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

- sti();
+out:
+ spin_unlock_irqrestore(&info->irq_lock, flags);
}

/*
@@ -1573,6 +1584,7 @@
    unsigned int change_dma;
    int retval = 0;
    struct esp_struct *current_async;
+ unsigned long flags;

    /* Perhaps a non-sysadmin user should be able to do some of these */
    /* operations. I haven't decided yet. */
@@ -1650,26 +1662,24 @@

    if ((new_config.flow_off != info->config.flow_off) ||
        (new_config.flow_on != info->config.flow_on)) {
- unsigned long flags;

+ spin_lock_irqsave(&info->irq_lock, flags);
    info->config.flow_off = new_config.flow_off;
    info->config.flow_on = new_config.flow_on;
- save_flags(flags); cli();
    serial_out(info, UART_ESI_CMD1, ESI_SET_FLOW_LVL);
    serial_out(info, UART_ESI_CMD2, new_config.flow_off >> 8);
    serial_out(info, UART_ESI_CMD2, new_config.flow_off);
    serial_out(info, UART_ESI_CMD2, new_config.flow_on >> 8);
    serial_out(info, UART_ESI_CMD2, new_config.flow_on);
- restore_flags(flags);
+ spin_unlock_irqrestore(&info->irq_lock, flags);
    }

    if ((new_config.rx_trigger != info->config.rx_trigger) ||
        (new_config.tx_trigger != info->config.tx_trigger)) {
- unsigned long flags;

+ spin_lock_irqsave(&info->irq_lock, flags);
    info->config.rx_trigger = new_config.rx_trigger;
    info->config.tx_trigger = new_config.tx_trigger;
- save_flags(flags); cli();
    serial_out(info, UART_ESI_CMD1, ESI_SET_TRIGGER);
    serial_out(info, UART_ESI_CMD2,
                new_config.rx_trigger >> 8);
@@ -1677,14 +1687,13 @@
    serial_out(info, UART_ESI_CMD2,
                new_config.tx_trigger >> 8);
    serial_out(info, UART_ESI_CMD2, new_config.tx_trigger);
- restore_flags(flags);
+ spin_unlock_irqrestore(&info->irq_lock, flags);
    }

```

Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

    if (new_config.rx_timeout != info->config.rx_timeout) {
- unsigned long flags;

+ spin_lock_irqsave(&info->irq_lock, flags);
    info->config.rx_timeout = new_config.rx_timeout;
- save_flags(flags); cli();

        if (info->IER & UART_IER_RDI) {
            serial_out(info, UART_ESI_CMD1,
@@ -1693,7 +1702,7 @@
                new_config.rx_timeout);
        }

- restore_flags(flags);
+ spin_unlock_irqrestore(&info->irq_lock, flags);
    }

    if (!(info->flags & ASYNC_INITIALIZED))
@@ -1716,11 +1725,12 @@
    {
        unsigned char status;
        unsigned int result;
+ unsigned long flags;

- cli();
+ spin_lock_irqsave(&info->irq_lock, flags);
        serial_out(info, UART_ESI_CMD1, ESI_GET_UART_STAT);
        status = serial_in(info, UART_ESI_STAT1);
- sti();
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        result = ((status & UART_LSR_TEMT) ? TIOCSER_TEMT : 0);
        return put_user(result, value);
    }
@@ -1730,17 +1740,23 @@
    {
        struct esp_struct * info = (struct esp_struct *)tty->driver_data;
        unsigned char control, status;
+ unsigned long flags;

- if (serial_paranoia_check(info, tty->name, __FUNCTION__))
+ spin_lock_irqsave(&info->irq_lock, flags);
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__))) {
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        return -ENODEV;
- if (tty->flags & (1 << TTY_IO_ERROR))
+ }
+ if (unlikely(tty->flags & (1 << TTY_IO_ERROR))) {
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        return -EIO;
+ }

```

Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

        control = info->MCR;
- cli();
+
        serial_out(info, UART_ESI_CMD1, ESI_GET_UART_STAT);
        status = serial_in(info, UART_ESI_STAT2);
- sti();
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        return ((control & UART_MCR_RTS) ? TIOCM_RTS : 0)
                | ((control & UART_MCR_DTR) ? TIOCM_DTR : 0)
                | ((status & UART_MSR_DCD) ? TIOCM_CAR : 0)
@@ -1753,13 +1769,17 @@
        unsigned int set, unsigned int clear)
{
        struct esp_struct * info = (struct esp_struct *)tty->driver_data;
+ unsigned long flags;

- if (serial_paranoia_check(info, tty->name, __FUNCTION__))
+ spin_lock_irqsave(&info->irq_lock, flags);
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__))) {
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        return -ENODEV;
- if (tty->flags & (1 << TTY_IO_ERROR))
+ }
+ if (unlikely(tty->flags & (1 << TTY_IO_ERROR))) {
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        return -EIO;
-
- cli();
+ }

        if (set & TIOCM_RTS)
            info->MCR |= UART_MCR_RTS;
@@ -1774,7 +1794,7 @@
        serial_out(info, UART_ESI_CMD1, ESI_WRITE_UART);
        serial_out(info, UART_ESI_CMD2, UART_MCR);
        serial_out(info, UART_ESI_CMD2, info->MCR);
- sti();
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        return 0;
}

@@ -1785,21 +1805,25 @@
{
        struct esp_struct * info = (struct esp_struct *)tty->driver_data;
        unsigned long flags;
-
- if (serial_paranoia_check(info, tty->name, "esp_break"))
+
+ spin_lock_irqsave(&info->irq_lock, flags);
+

```

Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

+ if (unlikely(serial_paranoid_check(info, tty->name, __FUNCTION__))) {
+ spin_unlock_irqrestore(&info->irq_lock, flags);
+     return;
+ }

- save_flags(flags); cli();
+     if (break_state == -1) {
+         serial_out(info, UART_ESI_CMD1, ESI_ISSUE_BREAK);
+         serial_out(info, UART_ESI_CMD2, 0x01);
+ spin_unlock_irqrestore(&info->irq_lock, flags);

+         interruptible_sleep_on(&info->break_wait);
+     } else {
+         serial_out(info, UART_ESI_CMD1, ESI_ISSUE_BREAK);
+         serial_out(info, UART_ESI_CMD2, 0x00);
+ spin_unlock_irqrestore(&info->irq_lock, flags);
+     }
- restore_flags(flags);
+ }

static int rs_ioctl(struct tty_struct *tty, struct file * file,
@@ -1809,8 +1833,9 @@
+     struct async_icount cprev, cnow; /* kernel counter temps */
+     struct serial_icounter_struct __user *p_cuser; /* user space */
+     void __user *argp = (void __user *)arg;
+ unsigned long flags;

- if (serial_paranoid_check(info, tty->name, "rs_ioctl"))
+ if (unlikely(serial_paranoid_check(info, tty->name, __FUNCTION__)))
+     return -ENODEV;

+     if ((cmd != TIOCGSERIAL) && (cmd != TIOCSSERIAL) &&
@@ -1849,17 +1874,17 @@
+         * Caller should use TIOCGICOUNT to see which one it was
+         */
+         case TIOCMWAIT:
- cli();
+ spin_lock_irqsave(&info->irq_lock, flags);
+         cprev = info->icount; /* note the counters on entry */
- sti();
+ spin_unlock_irqrestore(&info->irq_lock, flags);
+         while (1) {
+             interruptible_sleep_on(&info->delta_msr_wait);
+             /* see if a signal did it */
+             if (signal_pending(current))
+                 return -ERESTARTSYS;
- cli();
+ spin_lock_irqsave(&info->irq_lock, flags);
+         cnow = info->icount; /* atomic copy */
- sti();
+ spin_unlock_irqrestore(&info->irq_lock, flags);

```

Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

        if (cnow.rng == cprev.rng &&
            cnow.dsr == cprev.dsr &&
            cnow.dcd == cprev.dcd &&
@@ -1886,9 +1911,9 @@
        * RI where only 0->1 is counted.
        */
        case TIOCGICOUNT:
- cli();
+ spin_lock_irqsave(&info->irq_lock, flags);
        cnow = info->icount;
- sti();
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        p_cuser = argp;
        if (put_user(cnow.cts, &p_cuser->cts) ||
            put_user(cnow.dsr, &p_cuser->dsr) ||
@@ -1911,6 +1936,7 @@
static void rs_set_termios(struct tty_struct *tty, struct termios *old_termios)
{
    struct esp_struct *info = (struct esp_struct *)tty->driver_data;
+ unsigned long flags;

    if ( (tty->termios->c_cflag == old_termios->c_cflag)
        && (RELEVANT_IFLAG(tty->termios->c_iflag)
@@ -1922,23 +1948,23 @@
        /* Handle transition to B0 status */
        if ((old_termios->c_cflag & CBAUD) &&
            !(tty->termios->c_cflag & CBAUD)) {
+ spin_lock_irqsave(&info->irq_lock, flags);
        info->MCR &= ~(UART_MCR_DTR|UART_MCR_RTS);
- cli();
        serial_out(info, UART_ESI_CMD1, ESI_WRITE_UART);
        serial_out(info, UART_ESI_CMD2, UART_MCR);
        serial_out(info, UART_ESI_CMD2, info->MCR);
- sti();
+ spin_unlock_irqrestore(&info->irq_lock, flags);
    }

    /* Handle transition away from B0 status */
    if (!(old_termios->c_cflag & CBAUD) &&
        (tty->termios->c_cflag & CBAUD)) {
+ spin_lock_irqsave(&info->irq_lock, flags);
        info->MCR |= (UART_MCR_DTR | UART_MCR_RTS);
- cli();
        serial_out(info, UART_ESI_CMD1, ESI_WRITE_UART);
        serial_out(info, UART_ESI_CMD2, UART_MCR);
        serial_out(info, UART_ESI_CMD2, info->MCR);
- sti();
+ spin_unlock_irqrestore(&info->irq_lock, flags);
    }

    /* Handle turning of CRTSCTS */

```

## Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

@@ -1975,10 +2001,11 @@
     struct esp_struct * info = (struct esp_struct *)tty->driver_data;
     unsigned long flags;

- if (!info || serial_paranoia_check(info, tty->name, "rs_close"))
- return;
-
- save_flags(flags); cli();
+ spin_lock_irqsave(&info->irq_lock, flags);
+
+ if (unlikely(!info ||
+ serial_paranoia_check(info, tty->name, __FUNCTION__)))
+ goto out;

     if (tty_hung_up_p(filp)) {
         DBG_CNT("before DEC-hung");
@@ -2058,7 +2085,7 @@
     info->flags &= ~(ASYNC_NORMAL_ACTIVE|ASYNC_CLOSING);
     wake_up_interruptible(&info->close_wait);
out:
- restore_flags(flags);
+ spin_unlock_irqrestore(&info->irq_lock, flags);
}

static void rs_wait_until_sent(struct tty_struct *tty, int timeout)
@@ -2067,8 +2094,9 @@
     unsigned long orig_jiffies, char_time;
     unsigned long flags;

- if (serial_paranoia_check(info, tty->name, "rs_wait_until_sent"))
- return;
+ spin_lock_irqsave(&info->irq_lock, flags);
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
+ goto out;

     orig_jiffies = jiffies;
     char_time = ((info->timeout - HZ / 50) / 1024) / 5;
@@ -2076,13 +2104,15 @@
     if (!char_time)
         char_time = 1;

- save_flags(flags); cli();
     serial_out(info, UART_ESI_CMD1, ESI_NO_COMMAND);
     serial_out(info, UART_ESI_CMD1, ESI_GET_TX_AVAIL);

     while ((serial_in(info, UART_ESI_STAT1) != 0x03) ||
            (serial_in(info, UART_ESI_STAT2) != 0xff)) {
+
+ spin_unlock_irqrestore(&info->irq_lock, flags);
         msleep_interruptible(jiffies_to_msecs(char_time));
+ spin_lock_irqsave(&info->irq_lock, flags);

```

Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

        if (signal_pending(current))
            break;
@@ -2093,8 +2123,8 @@
        serial_out(info, UART_ESI_CMD1, ESI_NO_COMMAND);
        serial_out(info, UART_ESI_CMD1, ESI_GET_TX_AVAIL);
    }
-
- restore_flags(flags);
+
+out: spin_unlock_irqrestore(&info->irq_lock, flags);
        set_current_state(TASK_RUNNING);
    }

@@ -2105,7 +2135,7 @@
    {
        struct esp_struct * info = (struct esp_struct *)tty->driver_data;

- if (serial_paranoia_check(info, tty->name, "esp_hangup"))
+ if (unlikely(serial_paranoia_check(info, tty->name, __FUNCTION__)))
        return;

        rs_flush_buffer(tty);
@@ -2174,15 +2204,13 @@
        printk("block_til_ready before block: ttys%d, count = %d\n",
            info->line, info->count);
    #endif
- save_flags(flags);
- cli();
+ spin_lock_irqsave(&info->irq_lock, flags);
        if (!tty_hung_up_p(filp))
            info->count--;
- restore_flags(flags);
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        info->blocked_open++;
        while (1) {
- save_flags(flags);
- cli();
+ spin_lock_irqsave(&info->irq_lock, flags);
            if ((tty->termios->c_cflag & CBAUD)) {
                unsigned int scratch;

@@ -2194,7 +2222,7 @@
                serial_out(info, UART_ESI_CMD2,
                    scratch | UART_MCR_DTR | UART_MCR_RTS);
            }
- restore_flags(flags);
+ spin_unlock_irqrestore(&info->irq_lock, flags);
        set_current_state(TASK_INTERRUPTIBLE);
        if (tty_hung_up_p(filp) ||
            !(info->flags & ASYNC_INITIALIZED)) {

```

Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

@@ -2264,7 +2292,7 @@
     info = info->next_port;

     if (!info) {
- serial_paranoia_check(info, tty->name, "esp_open");
+ serial_paranoia_check(info, tty->name, __FUNCTION__);
         return -ENODEV;
     }

@@ -2275,12 +2303,6 @@
     tty->driver_data = info;
     info->tty = tty;

- if (!tmp_buf) {
- tmp_buf = (unsigned char *) get_zeroed_page(GFP_KERNEL);
- if (!tmp_buf)
- return -ENOMEM;
- }
-
     /*
     * Start up serial port
     */
@@ -2335,7 +2357,7 @@
     if (!request_region(info->port, REGION_SIZE, "esp serial"))
         return -EIO;

- save_flags(flags); cli();
+ spin_lock_irqsave(&info->irq_lock, flags);

     /*
     * Check for ESP card
@@ -2369,10 +2391,11 @@
         serial_out(info, UART_ESI_CMD2, 0x00);
     }
}
+ spin_unlock_irqrestore(&info->irq_lock, flags);
+
     if (!port_detected)
         release_region(info->port, REGION_SIZE);

- restore_flags(flags);
     return (port_detected);
}

@@ -2513,6 +2536,7 @@
     init_waitqueue_head(&info->close_wait);
     init_waitqueue_head(&info->delta_msr_wait);
     init_waitqueue_head(&info->break_wait);
+ spin_lock_init(&info->irq_lock);
     ports = info;
     printk(KERN_INFO "ttyP%d at 0x%04x (irq = %d) is an ESP ",

```

Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```

        info->line, info->port, info->irq);
@@ -2563,18 +2587,14 @@

static void __exit espserial_exit(void)
{
- unsigned long flags;
    int e1;
    struct esp_struct *temp_async;
    struct esp_pio_buffer *pio_buf;

    /* printk("Unloading %s: version %s\n", serial_name, serial_version); */
- save_flags(flags);
- cli();
    if ((e1 = tty_unregister_driver(esp_driver)))
- printk("SERIAL: failed to unregister serial driver (%d)\n",
- e1);
- restore_flags(flags);
+ printk(KERN_ERR "SERIAL: failed to unregister serial driver"
+ "(%d)\n", e1);
    put_tty_driver(esp_driver);

    while (ports) {
@@ -2590,9 +2610,6 @@
        free_pages((unsigned long)dma_buffer,
            get_order(DMA_BUFFER_SZ));

- if (tmp_buf)
- free_page((unsigned long)tmp_buf);
-
    while (free_pio_buf) {
        pio_buf = free_pio_buf->next;
        kfree(free_pio_buf);
diff -urN --exclude='*~' linux-2.6.10-original/drivers/char/Kconfig linux-2.6.10/drivers/char/Kconfig
--- linux-2.6.10-original/drivers/char/Kconfig 2004-12-24 16:33:49.000000000 -0500
+++ linux-2.6.10/drivers/char/Kconfig 2004-12-30 20:25:12.945068082 -0500
@@ -170,7 +170,7 @@

config ESPSERIAL
    tristate "Hayes ESP serial port support"
- depends on SERIAL_NONSTANDARD && ISA && BROKEN_ON_SMP
+ depends on SERIAL_NONSTANDARD && ISA
    help
        This is a driver which supports Hayes ESP serial ports. Both single
        port cards and multiport cards are supported. Make sure to read
diff -urN --exclude='*~' linux-2.6.10-original/include/linux/hayesesp.h
linux-2.6.10/include/linux/hayesesp.h
--- linux-2.6.10-original/include/linux/hayesesp.h 2004-12-24 16:35:24.000000000 -0500
+++ linux-2.6.10/include/linux/hayesesp.h 2004-12-30 18:15:06.541960410 -0500
@@ -108,6 +108,7 @@
    wait_queue_head_t break_wait;
    struct async_icount icount; /* kernel counters for the 4 input interrupts */

```

## Linux-Kernel: [PATCH] esp: Make driver SMP-correct

```
    struct hayes_esp_config config; /* port configuration */  
+ spinlock_t irq_lock;  
    struct esp_struct *next_port; /* For the linked list */  
};
```

—

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@vger.kernel.org

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>