

# [PATCH 11/19] ieee1394: dv1394: sem2mutex conversion

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-07/msg00450.html>

- *From:* Stefan Richter <[stefanr@xxxxxxxxxxxxxxxxxxxxx](mailto:stefanr@xxxxxxxxxxxxxxxxxxxxx)>
- *Date:* Mon, 3 Jul 2006 01:08:59 +0200 (CEST)

Signed-off-by: Stefan Richter <[stefanr@xxxxxxxxxxxxxxxxxxxxx](mailto:stefanr@xxxxxxxxxxxxxxxxxxxxx)> (not runtime-tested)

```
-----  
drivers/ieee1394/dv1394-private.h | 6 ++----  
drivers/ieee1394/dv1394.c | 31 ++++++++-----  
2 files changed, 19 insertions(+), 18 deletions(-)
```

Index: linux/drivers/ieee1394/dv1394-private.h

```
=====  
--- linux.orig/drivers/ieee1394/dv1394-private.h 2006-07-01 09:16:48.000000000 +0200  
+++ linux/drivers/ieee1394/dv1394-private.h 2006-07-01 20:50:48.000000000 +0200  
@@ -460,7 +460,7 @@ struct video_card {  
int dma_running;
```

/\*

- 3) the sleeping semaphore 'sem' - this is used from process context only,  
+ 3) the sleeping mutex 'mtx' - this is used from process context only,  
to serialize various operations on the video\_card. Even though only one  
open() is allowed, we still need to prevent multiple threads of execution  
from entering calls like read, write, ioctl, etc.

```
@@ -468,9 +468,9 @@ struct video_card {
```

I honestly can't think of a good reason to use dv1394 from several threads  
at once, but we need to serialize anyway to prevent oopses =).

- NOTE: if you need both spinlock and sem, take sem first to avoid deadlock!

+ NOTE: if you need both spinlock and mtx, take mtx first to avoid deadlock!

\*/

```
- struct semaphore sem;
```

```
+ struct mutex mtx;
```

/\* people waiting for buffer space, please form a line here... \*/

```
wait_queue_head_t waitq;
```

Index: linux/drivers/ieee1394/dv1394.c

```
=====  
--- linux.orig/drivers/ieee1394/dv1394.c 2006-07-01 17:42:38.000000000 +0200  
+++ linux/drivers/ieee1394/dv1394.c 2006-07-01 20:50:48.000000000 +0200  
@@ -95,6 +95,7 @@  
#include <linux/fs.h>  
#include <linux/poll.h>
```

[PATCH 11/19] ieee1394: dv1394: sem2mutex conversion

```
#include <linux/smp_lock.h>
+#include <linux/mutex.h>
#include <linux/bitops.h>
#include <asm/byteorder.h>
#include <asm/atomic.h>
@@ -247,7 +248,7 @@ static void frame_delete(struct frame *f
```

Frame\_prepare() must be called OUTSIDE the video->spinlock.  
However, frame\_prepare() must still be serialized, so  
- it should be called WITH the video->sem taken.  
+ it should be called WITH the video->mtx taken.  
\*/

```
static void frame_prepare(struct video_card *video, unsigned int this_frame)
@@ -1271,7 +1272,7 @@ static int dv1394_mmap(struct file *file
int retval = -EINVAL;
```

```
/* serialize mmap */
- down(&video->sem);
+ mutex_lock(&video->mtx);
```

```
if ( ! video_card_initialized(video) ) {
retval = do_dv1394_init_default(video);
@@ -1281,7 +1282,7 @@ static int dv1394_mmap(struct file *file
```

```
retval = dma_region_mmap(&video->dv_buf, file, vma);
out:
- up(&video->sem);
+ mutex_unlock(&video->mtx);
return retval;
}
```

```
@@ -1337,17 +1338,17 @@ static ssize_t dv1394_write(struct file
```

```
/* serialize this to prevent multi-threaded mayhem */
if (file->f_flags & O_NONBLOCK) {
- if (down_trylock(&video->sem))
+ if (!mutex_trylock(&video->mtx))
return -EAGAIN;
} else {
- if (down_interruptible(&video->sem))
+ if (mutex_lock_interruptible(&video->mtx))
return -ERESTARTSYS;
}
```

```
if ( !video_card_initialized(video) ) {
ret = do_dv1394_init_default(video);
if (ret) {
- up(&video->sem);
+ mutex_unlock(&video->mtx);
return ret;
}
```

```

}
}
@@ -1418,7 +1419,7 @@ static ssize_t dv1394_write(struct file

remove_wait_queue(&video->waitq, &wait);
set_current_state(TASK_RUNNING);
- up(&video->sem);
+ mutex_unlock(&video->mtx);
return ret;
}

@@ -1434,17 +1435,17 @@ static ssize_t dv1394_read(struct file *

/* serialize this to prevent multi-threaded mayhem */
if (file->f_flags & O_NONBLOCK) {
- if (down_trylock(&video->sem))
+ if (!mutex_trylock(&video->mtx))
return -EAGAIN;
} else {
- if (down_interruptible(&video->sem))
+ if (mutex_lock_interruptible(&video->mtx))
return -ERESTARTSYS;
}

if ( !video_card_initialized(video) ) {
ret = do_dv1394_init_default(video);
if (ret) {
- up(&video->sem);
+ mutex_unlock(&video->mtx);
return ret;
}
video->continuity_counter = -1;
@@ -1526,7 +1527,7 @@ static ssize_t dv1394_read(struct file *

remove_wait_queue(&video->waitq, &wait);
set_current_state(TASK_RUNNING);
- up(&video->sem);
+ mutex_unlock(&video->mtx);
return ret;
}

@@ -1547,12 +1548,12 @@ static long dv1394_ioctl(struct file *fi

/* serialize this to prevent multi-threaded mayhem */
if (file->f_flags & O_NONBLOCK) {
- if (down_trylock(&video->sem)) {
+ if (!mutex_trylock(&video->mtx)) {
unlock_kernel();
return -EAGAIN;
}
} else {

```

[PATCH 11/19] ieee1394: dv1394: sem2mutex conversion

```
- if (down_interruptible(&video->sem)) {
+ if (mutex_lock_interruptible(&video->mtx)) {
unlock_kernel();
return -ERESTARTSYS;
}
@@ -1778,7 +1779,7 @@ static long dv1394_ioctl(struct file *fi
}
```

out:

```
- up(&video->sem);
+ mutex_unlock(&video->mtx);
unlock_kernel();
return ret;
}
@@ -2253,7 +2254,7 @@ static int dv1394_init(struct ti_ohci *o
clear_bit(0, &video->open);
spin_lock_init(&video->spinlock);
video->dma_running = 0;
- init_MUTEX(&video->sem);
+ mutex_init(&video->mtx);
init_waitqueue_head(&video->waitq);
video->fasync = NULL;
```

-

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