Subject: Re: [PATCH/RFC] kthread API conversion for dvb_frontend and av7110
Posted by Cedric Le Goater on Thu, 14 Sep 2006 21:07:49 GMT
View Forum Message <> Reply to Message

Herbert Poetzl wrote:
> Okay, as I promised, I had a first shot at the
> dvb kernel_thread to kthread API port, and here
> is the result, which is running fine here since
> yesterday, including module load/unload and
> software suspend (which doesn't work as expected
> with or without this patch :),

So you have such an hardware ?

[...]

> @@ -600,7 +595,6 @@ static int dvb_frontend_thread(void *dat
>
> static void dvb_frontend_stop(struct dvb_frontend *fe)
> {
> - unsigned long ret;
> struct dvb_frontend_private *fepriv = fe->frontend_priv;
>
> dprintk ("%s\n", __FUNCTION__);
> @@ -608,33 +602,17 @@ static void dvb_frontend_stop(struct dvb
> fepriv->exit = 1;

do we still need the ->exit flag now that we are using kthread_stop() ?
same question for ->wakeup ?

> mb();
>
> - if (!fepriv->thread_pid)
> - return;
> -
> - /* check if the thread is really alive */
> - if (kill_proc(fepriv->thread_pid, 0, 1) == -ESRCH) {
> - printk("dvb_frontend_stop: thread PID %d already died\n",
> - fepriv->thread_pid);
> - /* make sure the mutex was not held by the thread */
> - init_MUTEX (&fepriv->sem);
> + if (!fepriv->thread)
> return;
> - }
> -
> - /* wake up the frontend thread, so it notices that fe->exit == 1 */
> - dvb_frontend_wakeup(fe);
>

> - /* wait until the frontend thread has exited */
> - ret = wait_event_interruptible(fepriv->wait_queue,0 == fepriv->thread_pid);
> - if (-ERESTARTSYS != ret) {
> - fepriv->state = FESTATE_IDLE;
> - return;
> - }
> + kthread_stop(fepriv->thread);
> + init_MUTEX (&fepriv->sem);

the use of the semaphore to synchronise the thread is complex. It will
require extra care to avoid deadlocks.

>   fepriv->state = FESTATE_IDLE;
>
>   /* paranoia check in case a signal arrived */
> - if (fepriv->thread_pid)
> - printk("dvb_frontend_stop: warning: thread PID %d won't exit\n",
> -   fepriv->thread_pid);
> + if (fepriv->thread)
> + printk("dvb_frontend_stop: warning: thread %p won't exit\n",
> +   fepriv->thread);

kthread_stop uses a completion already. so the above is real paranoia :)

> }
>
> s32 timeval_usec_diff(struct timeval lasttime, struct timeval curtime)
> @@ -684,10 +662,11 @@ static int dvb_frontend_start(struct dvb
> {
>   int ret;
>   struct dvb_frontend_private *fepriv = fe->frontend_priv;
> + struct task_struct *fe_thread;
>
>   dprintk ("%s\n", __FUNCTION__);
>
> - if (fepriv->thread_pid) {
> + if (fepriv->thread) {
>   if (!fepriv->exit)
>   return 0;
>   else
> @@ -701,18 +680,18 @@ static int dvb_frontend_start(struct dvb
>
>   fepriv->state = FESTATE_IDLE;
>   fepriv->exit = 0;
> - fepriv->thread_pid = 0;
> + fepriv->thread = NULL;
>   mb();
>

> - ret = kernel_thread (dvb_frontend_thread, fe, 0);
> -
> - if (ret < 0) {
> - printk("dvb_frontend_start: failed to start kernel_thread (%d)\n", ret);
> + fe_thread = kthread_run(dvb_frontend_thread, fe,
> + "kdvb-fe-%i", fe->dvb->num);
> + if (IS_ERR(fe_thread)) {
> + ret = PTR_ERR(fe_thread);

ret could be local.

[...]

_____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers