Subject: Re: [PATCH] Add kernel/notifier.c
Posted by Alexey Dobriyan on Fri, 20 Jul 2007 07:22:42 GMT
View Forum Message <> Reply to Message

On Thu, Jul 19, 2007 at 02:48:59PM -0700, Andrew Morton wrote:
> On Thu, 19 Jul 2007 20:46:11 +0400
> Alexey Dobriyan <adobriyan@sw.ru> wrote:
>
> > There is separate notifier header, but no separate notifier .c file.
> >
> > Extract notifier code out of kernel/sys.c which will remain for
> > misc syscalls I hope. Merge kernel/die_notifier.c into kernel/notifier.c.
>
> If you were running checkpatch (I hope you are)

:)

> then you'd find that we
> copied over a whole pile of cruft. We might as well fix that up while
> we're moving the code around.
>
> Also, your patch tried to add some trailing whitespace, but checkpatch
> failed to notice that.

It notices here:

```
 $ ./scripts/checkpatch.pl ../notifier.patch | grep ERROR
 ERROR: trailing whitespace
 ERROR: trailing whitespace
 ERROR: trailing whitespace
 ERROR: trailing whitespace
 ERROR: trailing whitespace
 ERROR: "foo * bar" should be "foo *bar"
 ERROR: trailing whitespace
 ERROR: "foo * bar" should be "foo *bar"
```

> (The code motion and the cleanups should really be separate patches, and
> indeed they are, but I'll end up joining them before it hits git)

Then take this cleanup which is based on yours;
In particular, it fixes [Space][Tab][Space] places and removes lines
between kernel-doc comment and function itself.

```
--- a/kernel/notifier.c
+++ b/kernel/notifier.c
@@ -10,7 +10,6 @@
  * at shutdown. This is used to stop any idling DMA operations
```

```
 * and the like.
 */
-
BLOCKING_NOTIFIER_HEAD(reboot_notifier_list);

/*
@@ -50,13 +49,12 @@ static int notifier_chain_unregister(struct notifier_block **nl,
 * @val:  Value passed unmodified to notifier function
 * @v:  Pointer passed unmodified to notifier function
 * @nr_to_call: Number of notifier functions to be called. Don't care
- *        value of this parameter is -1.
+ *   value of this parameter is -1.
 * @nr_calls: Records the number of notifications sent. Don't care
- *      value of this field is NULL.
- *  @returns: notifier_call_chain returns the value returned by the
+ *   value of this field is NULL.
+ * @returns: notifier_call_chain returns the value returned by the
 *   last notifier function called.
 */
-
static int __kprobes notifier_call_chain(struct notifier_block **nl,
      unsigned long val, void *v,
      int nr_to_call, int *nr_calls)
@@ -95,7 +93,6 @@ static int __kprobes notifier_call_chain(struct notifier_block **nl,
 *
 * Currently always returns zero.
 */
-
int atomic_notifier_chain_register(struct atomic_notifier_head *nh,
   struct notifier_block *n)
{
@@ -107,7 +104,6 @@ int atomic_notifier_chain_register(struct atomic_notifier_head *nh,
 spin_unlock_irqrestore(&nh->lock, flags);
 return ret;
}
-
EXPORT_SYMBOL_GPL(atomic_notifier_chain_register);

/**
@@ -131,7 +127,6 @@ int atomic_notifier_chain_unregister(struct atomic_notifier_head *nh,
 synchronize_rcu();
 return ret;
}
-
EXPORT_SYMBOL_GPL(atomic_notifier_chain_unregister);

/**
@@ -153,7 +148,6 @@ EXPORT_SYMBOL_GPL(atomic_notifier_chain_unregister);
```

```
  * Otherwise the return value is the return value
  * of the last notifier function called.
  */
-
 int __kprobes __atomic_notifier_call_chain(struct atomic_notifier_head *nh,
       unsigned long val, void *v,
       int nr_to_call, int *nr_calls)
@@ -165,7 +159,6 @@ int __kprobes __atomic_notifier_call_chain(struct atomic_notifier_head
*nh,
  rcu_read_unlock();
  return ret;
 }
-
 EXPORT_SYMBOL_GPL(__atomic_notifier_call_chain);

 int __kprobes atomic_notifier_call_chain(struct atomic_notifier_head *nh,
@@ -173,8 +166,8 @@ int __kprobes atomic_notifier_call_chain(struct atomic_notifier_head *nh,
 {
  return __atomic_notifier_call_chain(nh, val, v, -1, NULL);
 }
-
 EXPORT_SYMBOL_GPL(atomic_notifier_call_chain);
+
 /*
  * Blocking notifier chain routines.  All access to the chain is
  * synchronized by an rwsem.
@@ -190,7 +183,6 @@ EXPORT_SYMBOL_GPL(atomic_notifier_call_chain);
  *
  * Currently always returns zero.
  */
-
 int blocking_notifier_chain_register(struct blocking_notifier_head *nh,
   struct notifier_block *n)
 {
@@ -209,7 +201,6 @@ int blocking_notifier_chain_register(struct blocking_notifier_head *nh,
  up_write(&nh->rwsem);
  return ret;
 }
-
 EXPORT_SYMBOL_GPL(blocking_notifier_chain_register);

 /**
@@ -240,7 +231,6 @@ int blocking_notifier_chain_unregister(struct blocking_notifier_head *nh,
  up_write(&nh->rwsem);
  return ret;
 }
-
 EXPORT_SYMBOL_GPL(blocking_notifier_chain_unregister);
```

```
 /**
@@ -261,7 +251,6 @@ EXPORT_SYMBOL_GPL(blocking_notifier_chain_unregister);
  * Otherwise the return value is the return value
  * of the last notifier function called.
  */
-
 int __blocking_notifier_call_chain(struct blocking_notifier_head *nh,
        unsigned long val, void *v,
        int nr_to_call, int *nr_calls)
@@ -305,13 +294,11 @@ EXPORT_SYMBOL_GPL(blocking_notifier_call_chain);
  *
  * Currently always returns zero.
  */
-
 int raw_notifier_chain_register(struct raw_notifier_head *nh,
   struct notifier_block *n)
 {
  return notifier_chain_register(&nh->head, n);
 }
-
 EXPORT_SYMBOL_GPL(raw_notifier_chain_register);

 /**
@@ -329,7 +316,6 @@ int raw_notifier_chain_unregister(struct raw_notifier_head *nh,
 {
  return notifier_chain_unregister(&nh->head, n);
 }
-
 EXPORT_SYMBOL_GPL(raw_notifier_chain_unregister);

 /**
@@ -351,14 +337,12 @@ EXPORT_SYMBOL_GPL(raw_notifier_chain_unregister);
  * Otherwise the return value is the return value
  * of the last notifier function called.
  */
-
 int __raw_notifier_call_chain(struct raw_notifier_head *nh,
        unsigned long val, void *v,
        int nr_to_call, int *nr_calls)
 {
  return notifier_call_chain(&nh->head, val, v, nr_to_call, nr_calls);
 }
-
 EXPORT_SYMBOL_GPL(__raw_notifier_call_chain);

 int raw_notifier_call_chain(struct raw_notifier_head *nh,
@@ -366,7 +350,6 @@ int raw_notifier_call_chain(struct raw_notifier_head *nh,
```

```
 {
  return __raw_notifier_call_chain(nh, val, v, -1, NULL);
 }
-
 EXPORT_SYMBOL_GPL(raw_notifier_call_chain);

 /*
@@ -384,7 +367,6 @@ EXPORT_SYMBOL_GPL(raw_notifier_call_chain);
  *
  * Currently always returns zero.
  */
-
 int srcu_notifier_chain_register(struct srcu_notifier_head *nh,
   struct notifier_block *n)
 {
@@ -403,7 +385,6 @@ int srcu_notifier_chain_register(struct srcu_notifier_head *nh,
  mutex_unlock(&nh->mutex);
  return ret;
 }
-
 EXPORT_SYMBOL_GPL(srcu_notifier_chain_register);

 /**
@@ -435,7 +416,6 @@ int srcu_notifier_chain_unregister(struct srcu_notifier_head *nh,
  synchronize_srcu(&nh->srcu);
  return ret;
 }
-
 EXPORT_SYMBOL_GPL(srcu_notifier_chain_unregister);

 /**
@@ -456,7 +436,6 @@ EXPORT_SYMBOL_GPL(srcu_notifier_chain_unregister);
  * Otherwise the return value is the return value
  * of the last notifier function called.
  */
-
 int __srcu_notifier_call_chain(struct srcu_notifier_head *nh,
         unsigned long val, void *v,
         int nr_to_call, int *nr_calls)
@@ -490,7 +469,6 @@ EXPORT_SYMBOL_GPL(srcu_notifier_call_chain);
  * up by calling srcu_cleanup_notifier_head().  Otherwise the head's
  * per-cpu data (used by the SRCU mechanism) will leak.
  */
-
 void srcu_init_notifier_head(struct srcu_notifier_head *nh)
 {
  mutex_init(&nh->mutex);
@@ -498,7 +476,6 @@ void srcu_init_notifier_head(struct srcu_notifier_head *nh)
```

```
   BUG();
  nh->head = NULL;
 }
-
 EXPORT_SYMBOL_GPL(srcu_init_notifier_head);

 /**
@@ -511,12 +488,10 @@ EXPORT_SYMBOL_GPL(srcu_init_notifier_head);
  * Currently always returns zero, as blocking_notifier_chain_register()
  * always returns zero.
  */
-
-int register_reboot_notifier(struct notifier_block * nb)
+int register_reboot_notifier(struct notifier_block *nb)
 {
  return blocking_notifier_chain_register(&reboot_notifier_list, nb);
 }
-
 EXPORT_SYMBOL(register_reboot_notifier);

 /**
@@ -528,30 +503,25 @@ EXPORT_SYMBOL(register_reboot_notifier);
  *
  * Returns zero on success, or %-ENOENT on failure.
  */
-
-int unregister_reboot_notifier(struct notifier_block * nb)
+int unregister_reboot_notifier(struct notifier_block *nb)
 {
  return  blocking_notifier_chain_unregister(&reboot_notifier_list , nb);
 }
-
 EXPORT_SYMBOL(unregister_reboot_notifier);

-
-
 static ATOMIC_NOTIFIER_HEAD(die_chain);

 int notify_die(enum die_val val, const char *str,
        struct pt_regs *regs, long err, int trap, int sig)
 {
  struct die_args args = {
-  .regs  = regs,
-  .str  = str,
-  .err  = err,
-  .trapnr  = trap,
-  .signr  = sig,
+  .regs = regs,
```

```
+  .str = str,
+  .err = err,
+  .trapnr = trap,
+  .signr = sig,

  };
-
  return atomic_notifier_call_chain(&die_chain, val, &args);
 }
```