

[patch 1/2] ACPI: bay: remove ACPI driver struct

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-01/msg02328.html>

- *From:* Kristen Carlson Accardi <kristen.c.accardi@xxxxxxxxxx>
 - *Date:* Mon, 8 Jan 2007 16:21:10 -0800
-

The bay driver is a platform driver, and doesn't need to also be an acpi driver. Remove the acpi driver related structures and callbacks, they didn't do anything anyway. Switch to uevent for user space event notification.

Signed-off-by: Kristen Carlson Accardi <kristen.c.accardi@xxxxxxxxxx>

drivers/acpi/bay.c | 99 ++-----
1 file changed, 4 insertions(+), 95 deletions(-)

```
--- 2.6-mm.orig/drivers/acpi/bay.c
+++ 2.6-mm/drivers/acpi/bay.c
@@ -47,18 +47,6 @@ MODULE_LICENSE("GPL");
acpi_get_name(h, ACPI_FULL_PATHNAME, &buffer);\
printk(KERN_DEBUG PREFIX "%s: %s\n", prefix, s); }
static void bay_notify(acpi_handle handle, u32 event, void *data);
-static int acpi_bay_add(struct acpi_device *device);
-static int acpi_bay_remove(struct acpi_device *device, int type);
-
-static struct acpi_driver acpi_bay_driver = {
- .name = ACPI_BAY_DRIVER_NAME,
- .class = ACPI_BAY_CLASS,
- .ids = ACPI_BAY_HID,
- .ops = {
- .add = acpi_bay_add,
- .remove = acpi_bay_remove,
- },
-};

struct bay {
acpi_handle handle;
@@ -234,14 +222,6 @@ int eject_removable_drive(struct device
}
EXPORT_SYMBOL_GPL(eject_removable_drive);

-static int acpi_bay_add(struct acpi_device *device)
-{
- bay_dprintf(device->handle, "adding bay device");
- strcpy(acpi_device_name(device), "Dockable Bay");
```

[patch 1/2] ACPI: bay: remove ACPI driver struct

```
- strcpy(acpi_device_class(device), "bay");
- return 0;
-}
-
static int acpi_bay_add_fs(struct bay *bay)
{
int ret;
@@ -339,52 +319,6 @@ bay_add_err:
return -ENODEV;
}

-static int acpi_bay_remove(struct acpi_device *device, int type)
-{
- /*** FIXME: do something here */
- return 0;
-}
-
-/**
- * bay_create_acpi_device - add new devices to acpi
- * @handle - handle of the device to add
- *
- * This function will create a new acpi_device for the given
- * handle if one does not exist already. This should cause
- * acpi to scan for drivers for the given devices, and call
- * matching driver's add routine.
- *
- * Returns a pointer to the acpi_device corresponding to the handle.
- */
-static struct acpi_device * bay_create_acpi_device(acpi_handle handle)
-{
- struct acpi_device *device = NULL;
- struct acpi_device *parent_device;
- acpi_handle parent;
- int ret;
-
- bay_dprintf(handle, "Trying to get device");
- if (acpi_bus_get_device(handle, &device)) {
- /*
- * no device created for this object,
- * so we should create one.
- */
- bay_dprintf(handle, "No device for handle");
- acpi_get_parent(handle, &parent);
- if (acpi_bus_get_device(parent, &parent_device))
- parent_device = NULL;
-
- ret = acpi_bus_add(&device, parent_device, handle,
- ACPI_BUS_TYPE_DEVICE);
- if (ret) {
- pr_debug("error adding bus, %x\n",
- -ret);
```

[patch 1/2] ACPI: bay: remove ACPI driver struct

```
- return NULL;
- }
- }
- return device;
-}
-
/**
 * bay_notify - act upon an acpi bay notification
 * @handle: the bay handle
@@ -394,38 +328,19 @@ static struct acpi_device * bay_create_a
 */
static void bay_notify(acpi_handle handle, u32 event, void *data)
{
- struct acpi_device *dev;
+ struct bay *bay_dev = (struct bay *)data;
+ struct device *dev = &bay_dev->pdev->dev;

bay_dprintk(handle, "Bay event");

switch(event) {
case ACPI_NOTIFY_BUS_CHECK:
- printk("Bus Check\n");
case ACPI_NOTIFY_DEVICE_CHECK:
- printk("Device Check\n");
- dev = bay_create_acpi_device(handle);
- if (dev)
- acpi_bus_generate_event(dev, event, 0);
- else
- printk("No device for generating event\n");
- /* wouldn't it be a good idea to just rescan SATA
- * right here?
- */
- break;
case ACPI_NOTIFY_EJECT_REQUEST:
- printk("Eject request\n");
- dev = bay_create_acpi_device(handle);
- if (dev)
- acpi_bus_generate_event(dev, event, 0);
- else
- printk("No device for generating eventn");
-
- /* wouldn't it be a good idea to just call the
- * eject_device here if we were a SATA device?
- */
+ kobject_uevent(&dev->kobj, KOBJ_CHANGE);
break;
default:
- printk("unknown event %d\n", event);
+ printk(KERN_ERR PREFIX "Bay: unknown event %d\n", event);
}
}
```

[patch 1/2] ACPI: bay: remove ACPI driver struct

```
@@ -457,10 +372,6 @@ static int __init bay_init(void)
acpi_walk_namespace(ACPI_TYPE_DEVICE, ACPI_ROOT_OBJECT,
ACPI_UINT32_MAX, find_bay, &bays, NULL);
```

```
- if (bays)
- if ((acpi_bus_register_driver(&acpi_bay_driver) < 0))
- printk(KERN_ERR "Unable to register bay driver\n");
-
if (!bays)
return -ENODEV;
```

```
@@ -481,8 +392,6 @@ static void __exit bay_exit(void)
kfree(bay->name);
kfree(bay);
}
-
- acpi_bus_unregister_driver(&acpi_bay_driver);
}
```

```
postcore_initcall(bay_init);
```

```
--
-
```

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/>