

[PATCH 1/4] media-radio: Pci probing for maestro radio

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2005-12/msg08655.html>

- *From:* "Jiri Slaby" <xslaby@xxxxxxxxxx>
 - *Date:* Sat, 31 Dec 2005 17:16:35 +0100
-

Pci probing for maestro radio

Pci probing functions added, some functions were rewritten.
Use PCI_DEVICE macro.
dev_* used for printing when pci_dev available.
some static variables changed to dynamic to allow operation with multiple cards.
Deleted macros for DEVICE_IDS, they are in pci_ids.h yet.

Signed-off-by: Jiri Slaby <jirislaby@xxxxxxxxxx>

commit 91d2ae25f82de5c70a8051cf579bf28d46d6bd59
tree a4501a344df183a6b851bbe223f4d40cc3400f38
parent 035d4d0665e725eb326dc3f5b6e96a53b5c559e3
author <ku@bellona.(none)> Sat, 31 Dec 2005 16:54:57 +0100
committer <ku@bellona.(none)> Sat, 31 Dec 2005 16:54:57 +0100

drivers/media/radio/radio-maestro.c | 173 ++++++-----
1 files changed, 109 insertions(+), 64 deletions(-)

diff --git a/drivers/media/radio/radio-maestro.c b/drivers/media/radio/radio-maestro.c

```
--- a/drivers/media/radio/radio-maestro.c
+++ b/drivers/media/radio/radio-maestro.c
@@ -27,11 +27,7 @@
#include <linux/pci.h>
#include <linux/videodev.h>

-#define DRIVER_VERSION "0.04"
-
-#define PCI_VENDOR_ESS 0x125D
-#define PCI_DEVICE_ID_ESS_ESS1968 0x1968 /* Maestro 2 */
-#define PCI_DEVICE_ID_ESS_ESS1978 0x1978 /* Maestro 2E */
+#define DRIVER_VERSION "0.05"
```

```
#define GPIO_DATA 0x60 /* port offset from ESS_IO_BASE */
```

```
@@ -66,6 +62,26 @@ module_param(radio_nr, int, 0);
```

[PATCH 1/4] media-radio: Pci probing for maestro radio

```
static int radio_ioctl(struct inode *inode, struct file *file,
unsigned int cmd, unsigned long arg);
+static int maestro_probe(struct pci_dev *pdev, const struct pci_device_id *ent);
+static void maestro_remove(struct pci_dev *pdev);
+
+static struct pci_device_id maestro_r_pci_tbl[] = {
+ { PCI_DEVICE(PCI_VENDOR_ID_ESS, PCI_DEVICE_ID_ESS_ESS1968),
+ .class = PCI_CLASS_MULTIMEDIA_AUDIO << 8,
+ .class_mask = 0xffff00 },
+ { PCI_DEVICE(PCI_VENDOR_ID_ESS, PCI_DEVICE_ID_ESS_ESS1978),
+ .class = PCI_CLASS_MULTIMEDIA_AUDIO << 8,
+ .class_mask = 0xffff00 },
+ { 0 }
+};
+MODULE_DEVICE_TABLE(pci, maestro_r_pci_tbl);
+
+static struct pci_driver maestro_r_driver = {
+ .name = "maestro_radio",
+ .id_table = maestro_r_pci_tbl,
+ .probe = maestro_probe,
+ .remove = __devexit_p(maestro_remove),
+};

static struct file_operations maestro_fops = {
.owner = THIS_MODULE,
@@ -85,14 +101,14 @@ static struct video_device maestro_radio
.fops = &maestro_fops,
};

-static struct radio_device
+struct radio_device
{
__u16 io, /* base of Maestro card radio io (GPIO_DATA)*/
muted, /* VIDEO_AUDIO_MUTE */
stereo, /* VIDEO_TUNER_STEREO_ON */
tuned; /* signal strength (0 or 0xffff) */
struct semaphore lock;
-} radio_unit = {0, 0, 0, 0, };
+};

static __u32 radio_bits_get(struct radio_device *dev)
{
@@ -251,40 +267,7 @@ static int radio_ioctl(struct inode *ino
return ret;
}

-static __u16 radio_install(struct pci_dev *pcidev);
-
-MODULE_AUTHOR("Adam Tlalka, atlka@xxxxxxxx");
-MODULE_DESCRIPTION("Radio driver for the Maestro PCI sound card radio.");
```

[PATCH 1/4] media-radio: Pci probing for maestro radio

```
-MODULE_LICENSE("GPL");
-
-static void __exit maestro_radio_exit(void)
-{
- video_unregister_device(&maestro_radio);
-}
-
-static int __init maestro_radio_init(void)
-{
- register __u16 found=0;
- struct pci_dev *pcidev = NULL;
- while(!found && (pcidev = pci_find_device(PCI_VENDOR_ESS,
- PCI_DEVICE_ID_ESS_ESS1968,
- pcidev)))
- found |= radio_install(pcidev);
- while(!found && (pcidev = pci_find_device(PCI_VENDOR_ESS,
- PCI_DEVICE_ID_ESS_ESS1978,
- pcidev)))
- found |= radio_install(pcidev);
- if(!found) {
- printk(KERN_INFO "radio-maestro: no devices found.\n");
- return -ENODEV;
- }
- return 0;
-}
-
-module_init(maestro_radio_init);
-module_exit(maestro_radio_exit);
-
-static inline __u16 radio_power_on(struct radio_device *dev)
+static __u16 __devinit radio_power_on(struct radio_device *dev)
{
register __u16 io=dev->io;
register __u32 ofreq;
@@ -305,29 +288,91 @@ static inline __u16 radio_power_on(struc
return (ofreq == radio_bits_get(dev));
}

-static __u16 radio_install(struct pci_dev *pcidev)
+static int __devinit maestro_probe(struct pci_dev *pdev,
+ const struct pci_device_id *ent)
{
- if(((pcidev->class >> 8) & 0xffff) != PCI_CLASS_MULTIMEDIA_AUDIO)
- return 0;
-
- radio_unit.io = pcidev->resource[0].start + GPIO_DATA;
- maestro_radio.priv = &radio_unit;
- init_MUTEX(&radio_unit.lock);
-
- if(radio_power_on(&radio_unit)) {
- if(video_register_device(&maestro_radio, VFL_TYPE_RADIO, radio_nr)==-1) {
```

[PATCH 1/4] media-radio: Pci probing for maestro radio

```
- printk("radio-maestro: can't register device!");
- return 0;
- }
- printk(KERN_INFO "radio-maestro: version "
- DRIVER_VERSION
- " time "
- __TIME__ " "
- __DATE__
- "\n");
- printk(KERN_INFO "radio-maestro: radio chip initialized\n");
- return 1;
- } else
- return 0;
+ struct radio_device *radio_unit;
+ struct video_device *maestro_radio_inst;
+ int retval;
+
+ retval = pci_enable_device(pdev);
+ if (retval) {
+ dev_err(&pdev->dev, "enabling pci device failed!\n");
+ goto err;
+ }
+
+ retval = -ENOMEM;
+
+ radio_unit = kzalloc(sizeof(*radio_unit), GFP_KERNEL);
+ if (radio_unit == NULL) {
+ dev_err(&pdev->dev, "not enough memory\n");
+ goto err;
+ }
+
+ radio_unit->io = pci_resource_start(pdev, 0) + GPIO_DATA;
+ init_MUTEX(&radio_unit->lock);
+
+ maestro_radio_inst = video_device_alloc();
+ if (maestro_radio_inst == NULL) {
+ dev_err(&pdev->dev, "not enough memory\n");
+ goto errfr;
+ }
+
+ memcpy(maestro_radio_inst, &maestro_radio, sizeof(maestro_radio));
+ video_set_drvdata(maestro_radio_inst, radio_unit);
+ pci_set_drvdata(pdev, maestro_radio_inst);
+
+ retval = video_register_device(maestro_radio_inst, VFL_TYPE_RADIO,
+ radio_nr);
+ if (retval) {
+ printk(KERN_ERR "can't register video device!\n");
+ goto errfr1;
+ }
+
+ 
```

[PATCH 1/4] media-radio: Pci probing for maestro radio

```
+ if (!radio_power_on(radio_unit)) {
+   retval = -EIO;
+   goto errfr1;
+ }
+
+ dev_info(&pdev->dev, "version " DRIVER_VERSION " time " __TIME__ " "
+   __DATE__ "\n");
+ dev_info(&pdev->dev, "radio chip initialized\n");
+
+ return 0;
+errfr1:
+ kfree(maestro_radio_inst);
+errfr:
+ kfree(radio_unit);
+err:
+ return retval;
+
+}
+
+static void __devexit maestro_remove(struct pci_dev *pdev)
+{
+ struct video_device *vdev = pci_get_drvdata(pdev);
+
+ video_unregister_device(vdev);
+}

+MODULE_AUTHOR("Adam Tlalka, atlka@xxxxxxxx");
+MODULE_DESCRIPTION("Radio driver for the Maestro PCI sound card radio.");
+MODULE_LICENSE("GPL");
+
+static int __init maestro_radio_init(void)
+{
+ int retval = pci_register_driver(&maestro_r_driver);
+
+ if (retval)
+ printk(KERN_ERR "error during registration pci driver\n");
+
+ return retval;
+}
+
+static void __exit maestro_radio_exit(void)
+{
+ pci_unregister_driver(&maestro_r_driver);
+}
+
+module_init(maestro_radio_init);
+module_exit(maestro_radio_exit);
-
```

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

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

- **References:**

- ◆ **[PATCH 0/4] maestro radio driver rewritten to 2.6 pci api**
 - ◇ *From: Jiri Slaby*

- Prev by Date: **[PATCH 0/4] maestro radio driver rewritten to 2.6 pci api**
- Next by Date: **[PATCH 4/4] media-radio: Maestro avoid accessing private structures directly**
- Previous by thread: **[PATCH 3/4] media-radio: Maestro types change**
- Next by thread: **[PATCH 4/4] media-radio: Maestro avoid accessing private structures directly**
- Index(es):
 - ◆ **Date**
 - ◆ **Thread**