

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

```
+static unsigned short normal_addr[] = { 0, I2C_CLIENT_END };
```

Bad idea. If no platform data is found, the driver will attempt to probe I2C address 0, which is a broadcast address. I can predict interesting results ;)

So, if you don't want the driver to do anything in the absence of platform data, you should define normal_addr the following way:

If there is no platform data, m41t00_platform_probe() should return -ENODEV and make m41t00_init() causing the driver to not become active. I've tested that, actually, and it seems to work.

```
static unsigned short normal_addr[] = { I2C_CLIENT_END, I2C_CLIENT_END };
```

Hrm, that's a good idea. I'll do that, even though it shouldn't be necessary.

```
+struct m41t00_chip_info {  
+ u16 type;  
+ u16 read_limit;  
+ u8 sec; /* Offsets for chip regs */  
+ u8 min;  
+ u8 hour;  
+ u8 day;  
+ u8 mon;  
+ u8 year;  
+ u8 alarm_mon;  
+ u8 alarm_hour;  
+ u8 sqw;  
+ u32 sqw_freq;  
+};
```

u16 is probably overkill for .type and .read_limit.

Yeah, probably. I'll change them to u8.

```
+static struct m41t00_chip_info m41t00_chip_info_tbl[] = {  
+ { M41T00_TYPE_M41T00, 5, 0, 1, 2, 4, 5, 6, 0, 0, 0, 0 },  
+ { M41T00_TYPE_M41T81, 1, 1, 2, 3, 5, 6, 7, 0xa, 0xc, 0x13, 0 },  
+ { M41T00_TYPE_M41T85, 1, 1, 2, 3, 5, 6, 7, 0xa, 0xc, 0x13, 0 },  
+};
```

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

C99-style initialization, please? It'll take much more lines, granted, but is also way more robust to future changes, and will be more readable too.

Well, the struct is right there so I figured the chances of changing the struct and not seeing that the compile-time init code needs to be changed too was small. I guess its better to be safe than sorry and it does make it more readable so I'll change it.

May I ask why you define separate types for the M41T81 and the M41T85, when you handle both exactly the same way?

The only reason is so that its obvious that both the t81 and t85 are supported. If I make it M41T81 only then I can easily see someone grepping around looking for M41T85, not finding it, and writing their own driver. I don't see any harm in having both, do you?

```
+ sec = buf[m41t00_chip->sec] & 0x7f;
+ min = buf[m41t00_chip->min] & 0x7f;
+ hour = buf[m41t00_chip->hour] & 0x3f;
+ day = buf[m41t00_chip->day] & 0x3f;
+ mon = buf[m41t00_chip->mon] & 0x1f;
+ year = buf[m41t00_chip->year] & 0xff;
```

The year masking is a no-op, you could omit it.

Yes.

```
@@ -169,24 +246,100 @@ m41t00_probe(struct i2c_adapter *adap, i
{
struct i2c_client *client;
int rc;
+ u8 rbuf[1], wbuf[2], msgbuf[1] = { 0 }; /* offset into rtc's regs */
+ struct i2c_msg msgs[] = {
+ {
+ .addr = 0,
+ .flags = 0,
+ .len = 1,
+ .buf = msgbuf,
+ },
+ {
+ .addr = 0,
+ .flags = I2C_M_RD,
```

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

```
+ .len = 1,  
+ .buf = rbuf,  
+ },  
+ };
```

Why don't you initialize both .addr right away?

Actually, I already init the .addr field right away so I'll remove them from them from the code above.

```
client = kzalloc(sizeof(struct i2c_client), GFP_KERNEL);  
if (!client)  
return -ENOMEM;  
  
- strcpy(client->name, M41T00_DRV_NAME, I2C_NAME_SIZE);  
+ strcpy(client->name, m41t00_driver.driver.name, I2C_NAME_SIZE);
```

The driver name may not be suitable here, the client name should instead reflect what chip it is.

Ah, good one.

```
+ if (m41t00_chip->type != M41T00_TYPE_M41T00) {  
+ /* If asked, set SQW frequency & enable */  
+ if (m41t00_chip->sqw_freq) {  
+ msgbuf[0] = m41t00_chip->alarm_mon;  
+ if ((rc = i2c_transfer(client->adapter, msgs, 2)) < 0)  
+ goto sqw_err;
```

You did not check whether the adapter you are attaching to supports this kind of transaction! You need to call `i2c_check_functionality`, see how other i2c chip drivers (for example `ds1337`) are doing. For `i2c_transfer` and `i2c_master_send`, you want to check for `I2C_FUNC_I2C`.

Okay.

```
+  
+ wbuf[0] = m41t00_chip->alarm_mon;  
+ wbuf[1] = rbuf[0] & ~0x40;  
+ if ((rc = i2c_master_send(client, wbuf, 2)) < 0)  
+ goto sqw_err;
```

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

```
+
+ wbuf[0] = m41t00_chip->sqw;
+ wbuf[1] = m41t00_chip->sqw_freq;
+ if ((rc = i2c_master_send(client, wbuf, 2)) < 0)
+ goto sqw_err;
+
+ wbuf[0] = m41t00_chip->alarm_mon;
+ wbuf[1] = rbuf[0] | 0x40;
+ if ((rc = i2c_master_send(client, wbuf, 2)) < 0)
+ goto sqw_err;
+ }
+
+ /* Make sure HT (Halt Update) bit is cleared */
+ msgbuf[0] = m41t00_chip->alarm_hour;
+ if ((rc = i2c_transfer(client->adapter, msgbuf, 2)) < 0)
+ goto ht_err;
+
+ if (rbuf[0] & 0x40) {
+ wbuf[0] = m41t00_chip->alarm_hour;
+ wbuf[1] = rbuf[0] & ~0x40;
+ if ((rc = i2c_master_send(client, wbuf, 2)) < 0)
+ goto ht_err;
+ }
+ }
+
+ /* Make sure ST (stop) bit is cleared */
+ msgbuf[0] = m41t00_chip->sec;
+ if ((rc = i2c_transfer(client->adapter, msgbuf, 2)) < 0)
+ goto st_err;
+
+ if (rbuf[0] & 0x80) {
+ wbuf[0] = m41t00_chip->sec;
+ wbuf[1] = rbuf[0] & ~0x80;
+ if ((rc = i2c_master_send(client, wbuf, 2)) < 0)
+ goto st_err;
+ }
}
```

What you do here are basically SMBus Read Byte and SMBus Write Byte transactions. The code would be much more simple if you were using the `i2c_smbus_read_byte_data` and `i2c_smbus_write_byte_data` functions, which take care of all the technical details.

That's true but I assumed that since I was using `i2c_transfer` earlier, I should use it here. Is that a bad assumption?
I do see that `ds1337.c` uses both types.

```
+st_err:
+ dev_err(&client->dev, "m41t00_probe: Can't clear ST bit\n");
```

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

```
+ goto attach_err;
+ht_err:
+ dev_err(&client->dev, "m41t00_probe: Can't clear HT bit\n");
+ goto attach_err;
+sqw_err:
+ dev_err(&client->dev, "m41t00_probe: Can't set SQW Frequency\n");
+attach_err:
+ kfree(client);
+ return rc;
}
```

Mrghh, this isn't exactly elegant... You should be able to make it better if you switch to i2c_smbus_read/write_byte_data as suggested.

Yep.

```
diff -Nurp linux-2.6.16-mm1-cleanup/include/linux/m41t00.h
linux-2.6.16-mm1-newsupp/include/linux/m41t00.h
--- linux-2.6.16-mm1-cleanup/include/linux/m41t00.h 1969-12-31
17:00:00.000000000 -0700
+++ linux-2.6.16-mm1-newsupp/include/linux/m41t00.h 2006-03-27
16:25:51.000000000 -0700
@@ -0,0 +1,50 @@
+/*
+ * Definitions for the ST M41T00 family of i2c rtc chips.
+ *
+ * Author: Mark A. Greer <mgreer@xxxxxxxxxxx>
+ *
+ * 2005 (c) MontaVista Software, Inc. This file is licensed under
```

We're in year 2006 now :)

Yes, but my understanding is that I should leave the 2005 there b/c that file was already copyrighted with that year. I could do a "2005, 2006" if you like.

```
+ * the terms of the GNU General Public License version 2. This program
+ * is licensed "as is" without any warranty of any kind, whether express
+ * or implied.
+ */
+
+ #ifndef _M41T00_H
+ #define _M41T00_H
+
+ #define M41T00_DRV_NAME "m41t00"
```

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

Why do you need to export this?

Because the code that passes the platform_data needs to put that name in the data so that the driver can find it. That's the value that the driver uses to search for its platform_data. I'll attach the patch to the platform code that I used to test it so you can see what I mean. Its at the end of this email.

```
+#define M41T00_I2C_ADDR 0x68
```

Not used anywhere?

Yes, in the platform code.

```
+#define M41T00_TYPE_M41T00 0
+#define M41T00_TYPE_M41T81 81
+#define M41T00_TYPE_M41T85 85
```

The i2c core has a facility for drivers supporting several types of devices. Check for I2C_CLIENT_INSMOD_3() in your case. The advantage if you go that way is that chip types can be set from userspace through module parameters, which may be convenient for testing when adding support for new platforms.

Okay, I'll take a look.

```
+/ * SQW output disabled, this is default value by power on */
+#define SQW_FREQ_DISABLE (0)
+
+#define SQW_FREQ_32KHZ (1<<4) /* 32.768 KHz */
+#define SQW_FREQ_8KHZ (2<<4) /* 8.192 KHz */
+#define SQW_FREQ_4KHZ (3<<4) /* 4.096 KHz */
+#define SQW_FREQ_2KHZ (4<<4) /* 2.048 KHz */
+#define SQW_FREQ_1KHZ (5<<4) /* 1.024 KHz */
+#define SQW_FREQ_512HZ (6<<4) /* 512 Hz */
+#define SQW_FREQ_256HZ (7<<4) /* 256 Hz */
+#define SQW_FREQ_128HZ (8<<4) /* 128 Hz */
+#define SQW_FREQ_64HZ (9<<4) /* 64 Hz */
+#define SQW_FREQ_32HZ (10<<4) /* 32 Hz */
+#define SQW_FREQ_16HZ (11<<4) /* 16 Hz */
+#define SQW_FREQ_8HZ (12<<4) /* 8 Hz */
+#define SQW_FREQ_4HZ (13<<4) /* 4 Hz */
```

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

```
+#define SQW_FREQ_2HZ (14<<4) /* 2 Hz */  
+#define SQW_FREQ_1HZ (15<<4) /* 1 Hz */
```

Not used anywhere either?

Platform code can pass one of these values into the driver via platform_data (if its a t81 or t85, not needed for the t00).

```
+extern ulong m41t00_get_rtc_time(void);  
+extern int m41t00_set_rtc_time(ulong nowtime);
```

Hopefully you won't have to export these anymore after you move to the new RTC subsystem. But that's a different story. In the meantime, shouldn't you have EXPORT_SYMBOL_GPL() for these functions? Else compiling m41t00 as a module won't work.

Good point. Will fix.

Below is the katana patch. Basically, it adds the platform_data required for the m41t00 rtc which the m41t00 driver later takes out. Note that the m41t00 doesn't need the sqw_freq value so you won't see it being set in this patch.

I did have a couple questions above so I'll wait for your response and then make a new patch.

Thanks for your time, Jean.

Mark

```
diff -Nurp linux-2.6.16-mm1-cleanup/arch/ppc/platforms/katana.c  
linux-2.6.16-mm1-newsupp/arch/ppc/platforms/katana.c  
--- linux-2.6.16-mm1-cleanup/arch/ppc/platforms/katana.c 2006-03-23 16:15:22.000000000 -0700  
+++ linux-2.6.16-mm1-newsupp/arch/ppc/platforms/katana.c 2006-03-23 17:37:46.000000000 -0700  
@@ -28,6 +28,7 @@  
#include <linux/mtd/physmap.h>  
#include <linux/mv643xx.h>  
#include <linux/platform_device.h>  
+#include <linux/m41t00.h>  
#ifdef CONFIG_BOOTIMG  
#include <linux/bootimg.h>  
#endif  
@@ -843,8 +844,31 @@ katana_find_end_of_memory(void)  
}
```

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

Re: [PATCH 2.6.16-mm1 3/3] rtc: add support for m41t81 & m41t85 chips to m41t00 driver

```
#if defined(CONFIG_I2C_MV64XXX) && defined(CONFIG_SENSORS_M41T00)
-extern ulong m41t00_get_rtc_time(void);
-extern int m41t00_set_rtc_time(ulong);
+static struct m41t00_platform_data katana_m41t00_pdata = {
+ .type = M41T00_TYPE_M41T00,
+ .i2c_addr = M41T00_I2C_ADDR,
+};
+
+static struct platform_device katana_m41t00_pdev = {
+ .name = M41T00_DRV_NAME,
+ .id = 0,
+ .num_resources = 0,
+ .dev = {
+ .platform_data = &katana_m41t00_pdata,
+ },
+};
+
+static int __init
+katana_add_pdata(void)
+{
+ int rc;
+
+ if ((rc = platform_device_register(&katana_m41t00_pdev)))
+ platform_device_unregister(&katana_m41t00_pdev);
+
+ return rc;
+}
+arch_initcall(katana_add_pdata);

static int __init
katana_rtc_hookup(void)
-
```

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/>