

Re: [PATCH] i2c: adds support for i2c bus on 8xx

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-05/msg04880.html>

- *From:* Vitaly Bordug <vitb@xxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Thu, 10 May 2007 14:35:54 +0400
-

On Thu, 10 May 2007 11:28:35 +0200
Jean Delvare wrote:

Hi Vitaly,

There is a mailing list dedicated to Linux I2C development (check MAINTAINERS), so why don't you use it instead of the already cluttered LKML?

because it's both arch-specific and drivers/ thing. I'll Cc that ML next time.

On Tue, 08 May 2007 10:31:31 +0400, Vitaly Bordug wrote:

Utilized devicetree to store I2C data, ported i2c-algo-8xx.c from 2.4 approach(which remains nearly intact), refined i2c-rpx.c. I2C functionality has been validated on mpc885ads with EEPROM access.

Interface has been reworked to of_device as well as other feedback addressed. Repeated start ability is missing from the implementation.

Did you check the hardware documentation? Is is a hardware limitation? It is likely to cause problems, so it should be investigated, and if it really cannot be fixed, then this must be clearly documented (in Kconfig and in the driver) and the advertised functionalities of the driver should be reduced accordingly (most SMBus transactions include a repeated start.)

Yes, I've checked RM about I2C controller, and cannot find anything about repeated start. There are a few words about multi-master considerations, that SoC features bus arbitration, but to avoid collisions, "higher-level handshake protocol must be used" (MPC885 RM, p. 32-6).

How the driver should advertise this case, can you suggest?

Re: [PATCH] i2c: adds support for i2c bus on 8xx

Signed-off-by: Vitaly Bordug <vitb@xxxxxxxxxxxxxxxxxxxxxx>

```
arch/powerpc/platforms/8xx/mpc885ads_setup.c | 15 +
arch/powerpc/sysdev/fsl_soc.c | 15 +
drivers/i2c/algos/Kconfig | 2
drivers/i2c/algos/Makefile | 1
drivers/i2c/algos/i2c-algo-8xx.c | 520
+++++
drivers/i2c/busses/Kconfig | 2
drivers/i2c/busses/i2c-rpx.c | 143 +++++-
include/linux/i2c-algo-8xx.h | 29 + 8 files
changed, 683 insertions(+), 44 deletions(-)
```

After my original review, we had agreed that having a separate algorithm driver wasn't needed. You didn't change that though. Why? I will not consider applying this patch until that change is done (or a good reason not to do it is given.)

yes, that's correct, missed this point...

```
diff --git a/arch/powerpc/platforms/8xx/mpc885ads_setup.c
b/arch/powerpc/platforms/8xx/mpc885ads_setup.c index
a57b577..40d12a2 100644 ---
a/arch/powerpc/platforms/8xx/mpc885ads_setup.c +++
b/arch/powerpc/platforms/8xx/mpc885ads_setup.c @@ -51,6 +51,17 @@
static void init_smc1_uart_ioports(struct fs_uart_platform_info*
fpi); static void init_smc2_uart_ioports(struct
fs_uart_platform_info* fpi); static void init_scc3_ioports(struct
fs_platform_info* ptr); #ifdef CONFIG_I2C_RPXLITE
+static void init_i2c_ioports()
+{
+ cpm8xx_t *cp = (cpm8xx_t *)immr_map(im_cpm);
+
+ setbits32(&cp->cp_pbpar, 0x00000030);
+ setbits32(&cp->cp_pbdir, 0x00000030);
+ setbits16(&cp->cp_pbodr, 0x0030);
+}
+ #endif
+
void __init mpc885ads_board_setup(void)
{
cpm8xx_t *cp;
@@ -115,6 +126,10 @@ void __init mpc885ads_board_setup(void)
immr_unmap(io_port);
```

Re: [PATCH] i2c: adds support for i2c bus on 8xx

```
#endif
+
+#ifdef CONFIG_I2C_RPXLITE
+ init_i2c_ioports();
+#endif
}

diff --git a/arch/powerpc/sysdev/fsl_soc.c
b/arch/powerpc/sysdev/fsl_soc.c index 8a123c7..be60db7 100644
--- a/arch/powerpc/sysdev/fsl_soc.c
+++ b/arch/powerpc/sysdev/fsl_soc.c
@@ -33,6 +33,7 @@
#include <asm/irq.h>
#include <asm/time.h>
#include <asm/prom.h>
+#include <asm/of_platform.h>
#include <sysdev/fsl_soc.h>
#include <mm/mmu_decl.h>
#include <asm/cpm2.h>
@@ -1102,4 +1103,18 @@ err:

arch_initcall(cpm_smc_uart_of_init);

+static int __init fsl_i2c_cpm_of_init(void)
+{
+ struct device_node *np = NULL;
+ /*
+ * Register all the devices which type is "i2c-cpm"
+ */
+ while ((np = of_find_compatible_node(np, "i2c",
+ "fsl,i2c-cpm")) != NULL)
+ of_platform_device_create(np, "fsl-i2c-cpm", NULL);
+ return 0;
+}
+
+arch_initcall(fsl_i2c_cpm_of_init);
+
+
+#endif /* CONFIG_8xx */
diff --git a/drivers/i2c/algos/Kconfig b/drivers/i2c/algos/Kconfig
index af02034..11e37ff 100644
--- a/drivers/i2c/algos/Kconfig
+++ b/drivers/i2c/algos/Kconfig
@@ -41,6 +41,8 @@ config I2C_ALGOPCA
config I2C_ALGO8XX
tristate "MPC8xx CPM I2C interface"
depends on 8xx && I2C
+ help
+ 8xx I2C Algorithm,supports the CPM I2C interface for
```

Re: [PATCH] i2c: adds support for i2c bus on 8xx

```
mpc8xx CPUs.
config I2C_ALGO_SGI
tristate "I2C SGI interfaces"
diff --git a/drivers/i2c/algos/Makefile b/drivers/i2c/algos/Makefile
index cac1051..1bd3b37 100644
--- a/drivers/i2c/algos/Makefile
+++ b/drivers/i2c/algos/Makefile
@@ -6,6 +6,7 @@ obj-$(CONFIG_I2C_ALGOBIT) += i2c-algo-bit.o
obj-$(CONFIG_I2C_ALGOPCF) += i2c-algo-pcf.o
obj-$(CONFIG_I2C_ALGOPCA) += i2c-algo-pca.o
obj-$(CONFIG_I2C_ALGO_SGI) += i2c-algo-sgi.o
+obj-$(CONFIG_I2C_ALGO8XX) += i2c-algo-8xx.o

ifeq ($(CONFIG_I2C_DEBUG_ALGO),y)
EXTRA_CFLAGS += -DDEBUG
diff --git a/drivers/i2c/algos/i2c-algo-8xx.c
b/drivers/i2c/algos/i2c-algo-8xx.c new file mode 100644
index 0000000..01ffcee
--- /dev/null
+++ b/drivers/i2c/algos/i2c-algo-8xx.c
@@ -0,0 +1,520 @@
+/*
+ * i2c-algo-8xx.c i2x driver algorithms for MPC8XX CPM
+ * Copyright (c) 1999 Dan Malek (dmalek@xxxxxxx).
+ *
+ * This program is free software; you can redistribute it and/or
+ * modify
+ * it under the terms of the GNU General Public License as
+ * published by
+ * the Free Software Foundation; either version 2 of the License,
+ * or
+ * (at your option) any later version.
+
+ * This program is distributed in the hope that it will be useful,
+ * but WITHOUT ANY WARRANTY; without even the implied warranty of
+ * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
+ * See the
+ * GNU General Public License for more details.
+
+ * You should have received a copy of the GNU General Public
+ * License
+ * along with this program; if not, write to the Free Software
+ * Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
+ *
+ * moved into proper i2c interface; separated out platform specific
+ * parts into i2c-rpx.c
+ * Brad Parker (brad@xxxxxxxxxxxx)
+ *
+ * (C) 2007 Montavista Software, Inc.
+ * Vitaly Bordug <vitb@xxxxxxxxxxxxxxxxxxxxxx>
+ */
```

Re: [PATCH] i2c: adds support for i2c bus on 8xx

```
+
+#include <linux/kernel.h>
+#include <linux/module.h>
+#include <linux/delay.h>
+#include <linux/slab.h>
+#include <linux/init.h>
+#include <linux/interrupt.h>
+#include <linux/errno.h>
+#include <linux/sched.h>
+#include <linux/i2c.h>
+#include <linux/i2c-algo-8xx.h>
+#include <asm/io.h>
+#include <asm/cacheflush.h>
+#include <asm/time.h>
+#include <asm/mpc8xx.h>
+
+/* Try to define this if you have an older CPU (earlier than rev
D4) */ + #undef I2C_CHIP_ERRATA
+
+static wait_queue_head_t iic_wait;
+static ushort r_tbase, r_rbase;
+
+static int cpm_debug;
+
+static irqreturn_t cpm_iic_interrupt(int irq, void *dev_id)
+{
+ i2c8xx_t *i2c = (i2c8xx_t *) dev_id;
+
+ #if 0
+ /* Chip errata, clear enable. This is not needed on rev D4
CPUs */
+ /* This should probably be removed and replaced by
I2C_CHIP_ERRATA stuff */
+ /* Someone with a buggy CPU needs to confirm that */
+ out_8(&i2c->i2c_i2mod, in_8(&i2c->i2c_i2mod) | ~1);
+ #endif
+ /* Clear interrupt.
+ */
+ out_8(&i2c->i2c_i2cer, 0xff);
+
+ /* Get 'me going again.
+ */
+ wake_up_interruptible(&iic_wait);
+
+ return IRQ_HANDLED;
+}
+
+static int cpm_iic_init(struct i2c_adapter *adap)
+{
+ struct i2c_algo_8xx_data *cpm = adap->algo_data;
+ iic_t *iip = cpm->iip;
```

Re: [PATCH] i2c: adds support for i2c bus on 8xx

```
+ i2c8xx_t *i2c = cpm->i2c;
+ unsigned char brg;
+
+ if (cpm_debug)
+ dev_dbg(adap->dev, "cpm_iic_init()\n");
```

Should be "&adap->dev". Same for all other calls. This means you never tried to compile your code with debugging enabled. Please do (there are options in the I2C Support menu for this.)

OK

```
+
+ init_waitqueue_head(&iic_wait);
+
+ /* Initialize the parameter ram.
+ * We need to make sure many things are initialized to
+ zero,
+ * especially in the case of a microcode patch.
+ */
+ iip->iic_rstate = 0;
+ iip->iic_rdp = 0;
+ iip->iic_rbptr = 0;
+ iip->iic_rbc = 0;
+ iip->iic_rxtmp = 0;
+ iip->iic_tstate = 0;
+ iip->iic_tdp = 0;
+ iip->iic_tbptr = 0;
+ iip->iic_tbc = 0;
+ iip->iic_txtmp = 0;
+
+ /* Set up the IIC parameters in the parameter ram.
+ */
+ iip->iic_tbase = r_tbase = cpm->dp_addr;
+ iip->iic_rbase = r_rbase = cpm->dp_addr + sizeof(cbd_t) *
2; +
+ if (cpm_debug) {
+ dev_dbg(adap->dev, "iip %p, dp_addr 0x%x\n",
cpm->iip,
+ cpm->dp_addr);
+ dev_dbg(adap->dev, "iic_tbase %d, r_tbase %d\n",
iip->iic_tbase,
+ r_tbase);
+ }
+
+ iip->iic_tfcr = SMC_EB;
+ iip->iic_rfcr = SMC_EB;
+
```

Re: [PATCH] i2c: adds support for i2c bus on 8xx

```
+ /* Set maximum receive size.
+ */
+ iip->iic_mrblr = CPM_MAX_READ;
+
+ /* Initialize Tx/Rx parameters.
+ */
+ if (cpm->reloc == 0) {
+ cpm8xx_t *cp = cpm->cp;
+ int res;
+
+ u16 v = mk_cr_cmd(CPM_CR_CH_I2C, CPM_CR_INIT_TRX)
+ | CPM_CR_FLG; +
+ out_be16(&cp->cp_cpcr, v);
+ res = wait_event_timeout(iic_wait,
+ !(in_be16(&cp->cp_cpcr) &
+ CPM_CR_FLG),
+ HZ * 10);
```

This is a pretty long timeout. Can it realistically take that long?

In fact, it can. Especially when CPM is loaded with Ethernet/UART/Etc transactions. If you feel it is overkill and I2C shouldn't do that, I'll trim it down though.

```
+ if (!res)
+ return -EIO;
+
+ } else {
+ iip->iic_rbptr = iip->iic_rbase;
+ iip->iic_tbptr = iip->iic_tbase;
+ iip->iic_rstate = 0;
+ iip->iic_tstate = 0;
+ }
+
+ /* Select an arbitrary address. Just make sure it is
+ unique.
+ */
+ out_8(&i2c->i2c_i2add, 0xfe);
+
+ /* Make clock run at 60 kHz.
+ */
+ brg = ppc_proc_freq / (32 * 2 * 60000) - 3;
+ out_8(&i2c->i2c_i2brg, brg);
+
+ out_8(&i2c->i2c_i2mod, 0x00);
+ out_8(&i2c->i2c_i2com, 0x01); /* Master mode */
+
+ /* Disable interrupts.
+ */
```

Re: [PATCH] i2c: adds support for i2c bus on 8xx

Re: [PATCH] i2c: adds support for i2c bus on 8xx

```
+ out_8(&i2c->i2c_i2cmr, 0);
+ out_8(&i2c->i2c_i2cer, 0xff);
+
+ /* Install interrupt handler.
+ */
+ request_irq(cpm->irq, cpm_iic_interrupt, 0, "8xx_i2c",
i2c); +
+ return 0;
+}
+
+static int cpm_iic_shutdown(struct i2c_algo_8xx_data *cpm)
+{
+ i2c8xx_t *i2c = cpm->i2c;
+
+ /* Shut down IIC.
+ */
+ out_8(&i2c->i2c_i2mod, in_8(&i2c->i2c_i2mod) | ~1);
+ out_8(&i2c->i2c_i2cmr, 0);
+ out_8(&i2c->i2c_i2cer, 0xff);
+
+ return (0);
+}
```

Should return void, it cannot fail.

OK

```
+
+static void cpm_reset_iic_params(iic_t * iip)
+{
+ iip->iic_tbase = r_tbase;
+ iip->iic_rbase = r_rbase;
+
+ iip->iic_tfcr = SMC_EB;
+ iip->iic_rfcr = SMC_EB;
+
+ iip->iic_mrblr = CPM_MAX_READ;
+
+ iip->iic_rstate = 0;
+ iip->iic_rdp = 0;
+ iip->iic_rbptr = iip->iic_rbase;
+ iip->iic_rbc = 0;
+ iip->iic_rxtmp = 0;
+ iip->iic_tstate = 0;
+ iip->iic_tdp = 0;
+ iip->iic_tbptr = iip->iic_tbase;
+ iip->iic_tbc = 0;
+ iip->iic_txtmp = 0;
+}
```

Re: [PATCH] i2c: adds support for i2c bus on 8xx

```
+
+ #define BD_SC_NAK ((ushort)0x0004) /* NAK –
+ did not respond */ #define BD_SC_OV
+ ((ushort)0x0002) /* OV – receive overrun */ #define
+ CPM_CR_CLOSE_RXBD ((ushort)0x0007) +
+ static void force_close(struct i2c_adapter *adap)
+ {
+ struct i2c_algo_8xx_data *cpm = adap->algo_data;
+ i2c8xx_t *i2c = cpm->i2c;
+ if (cpm->reloc == 0) { /* micro code disabled */
+ cpm8xx_t *cp = cpm->cp;
+ u16 v =
+ mk_cr_cmd(CPM_CR_CH_I2C, CPM_CR_CLOSE_RXBD) |
+ CPM_CR_FLG; +
+ if (cpm_debug)
+ printk("force_close()\n");
+
+ out_be16(&cp->cp_cpcr, v);
+ wait_event_timeout(iic_wait,
+ !(in_be16(&cp->cp_cpcr) &
+ CPM_CR_FLG),
+ HZ * 5);
+ }
+ out_8(&i2c->i2c_i2cmr, 0x00); /* Disable all
+ interrupts */
+ out_8(&i2c->i2c_i2cer, 0xff);
+ }
+
+ /* Read from IIC...
+ * abyte = address byte, with r/w flag already set
+ */
+ static int
+ cpm_iic_read(struct i2c_adapter *adap, u_char abyte, char *buf,
+ int count) + {
+ struct i2c_algo_8xx_data *cpm = adap->algo_data;
+ iic_t *iip = cpm->iip;
+ i2c8xx_t *i2c = cpm->i2c;
+ cbd_t *tbuf, *rbdf;
+ u_char *tb;
+ int res = 0;
+
+ if (count >= CPM_MAX_READ)
+ return -EINVAL;
+
+ /* check for and use a microcode relocation patch */
+ if (cpm->reloc) {
+ cpm_reset_iic_params(iip);
+ }
+
+ tbuf = (cbd_t *) cpm_dpram_addr(iip->iic_tbase);
+ rbdf = (cbd_t *) cpm_dpram_addr(iip->iic_rbase);
```

Re: [PATCH] i2c: adds support for i2c bus on 8xx

```
+
+ /* To read, we need an empty buffer of the proper length.
+ * All that is used is the first byte for address, the
remainder
+ * is just used for timing (and doesn't really have to
exist).
+ */
+ tb = cpm->temp;
+ tb = (u_char *) (((uint) tb + 15) & ~15);
+ tb[0] = abyte; /* Device address byte w/rw
flag */ +
+ flush_dcache_range((unsigned long)tb, (unsigned long)(tb +
1)); +
+ if (cpm_debug)
+ dev_dbg(adap->dev, "cpm_iic_read(abyte=0x%x)\n",
abyte); +
+ tbd->cbd_bufaddr = __pa(tb);
+ tbd->cbd_datlen = count + 1;
+ tbd->cbd_sc = BD_SC_READY | BD_SC_LAST | BD_SC_WRAP |
BD_IIC_START; +
+ iip->iic_mrblr = count + 1; /* prevent excessive
read, +1
+ is needed otherwise
will the
+ RXB interrupt come too
early */ +
+ /* flush will invalidate too. */
+ flush_dcache_range((unsigned long)buf, (unsigned long)(buf
+ count)); +
+ rbd->cbd_datlen = 0;
+ rbd->cbd_bufaddr = __pa(buf);
+ rbd->cbd_sc = BD_SC_EMPTY | BD_SC_WRAP | BD_SC_INTRPT;
+
+ /* Chip bug, set enable here */
+ out_8(&i2c->i2c_i2cmr, 0x13); /* Enable some
interupts */
+ out_8(&i2c->i2c_i2cer, 0xff);
+ out_8(&i2c->i2c_i2mod, in_8(&i2c->i2c_i2mod) |
1); /* Enable */
+ out_8(&i2c->i2c_i2com, in_8(&i2c->i2c_i2com) |
0x80); /* Begin transmission */ +
+ /* Wait for IIC transfer */
+ res = wait_event_interruptible_timeout(iic_wait, 0, 1 *
HZ); +
+ if (res < 0) {
```

The usual style is no blank line between the action and the test.

Re: [PATCH] i2c: adds support for i2c bus on 8xx

OK

[snip]

```
pmsg->len,  
+ (unsigned long)pmsg->buf);  
+  
+ addr = pmsg->addr << 1;  
+ if (pmsg->flags & I2C_M_RD)  
+ addr |= 1;  
+ if (pmsg->flags & I2C_M_REV_DIR_ADDR)  
+ addr ^= 1;
```

You didn't drop this?

Correct, missed the cleanup.

```
+  
+ if (pmsg->flags & I2C_M_RD) {  
+ /* read bytes into buffer */  
+ ret = cpm_iic_read(adap, addr, pmsg->buf,  
pmsg->len);  
+ if (cpm_debug)  
+ dev_dbg(adap->dev,  
+ "i2c-algo-8xx.o: read %d  
bytes\n", ret);  
+ if (ret < pmsg->len) {  
+ return (ret < 0) ? ret :  
-EREMOTEIO;  
+ }  
+ } else {  
+ /* write bytes from buffer */  
+ ret = cpm_iic_write(adap, addr, pmsg->buf,  
pmsg->len);  
+ if (cpm_debug)  
+ dev_dbg(adap->dev,  
+ "i2c-algo-8xx.o: wrote %d\n",  
+ ret);  
+ if (ret < pmsg->len) {  
+ return (ret < 0) ? ret :  
-EREMOTEIO;  
+ }  
+ }  
+ }  
+ return (num);
```

Useless parentheses.

Re: [PATCH] i2c: adds support for i2c bus on 8xx

yeah...

```
+}
+
+static u32 cpm_func(struct i2c_adapter *adap)
+{
+ return I2C_FUNC_I2C | I2C_FUNC_SMBUS_EMUL;
+}
+
+/* -----exported algorithm data:
+----- */
+static struct i2c_algorithm cpm_algo = {
+ .master_xfer = cpm_xfer,
+ .functionality = cpm_func,
+};
+
+/*
+ * registering functions to load algorithms at runtime
+ */
+int i2c_8xx_add_bus(struct i2c_adapter *adap)
+{
+ int res;
+
+ if (cpm_debug)
+ dev_dbg(adap->dev,
+ "i2c-algo-8xx.o: hw routines for %s
+ registered.\n",
+ adap->name);
+
+ /* register new adapter to i2c module... */
+
+ adap->algo = &cpm_algo;
+
+ res = cpm_iic_init(adap);
+
+ if (res)
+ return res;
+
+ return i2c_add_adapter(adap);
+}
+
+int i2c_8xx_del_bus(struct i2c_adapter *adap)
+{
+ struct i2c_algo_8xx_data *cpm = adap->algo_data;
+
+ i2c_del_adapter(adap);
+
+ return cpm_iic_shutdown(cpm);
+}
```

Re: [PATCH] i2c: adds support for i2c bus on 8xx

Should return void.

OK.

```
+
+EXPORT_SYMBOL(i2c_8xx_add_bus);
+EXPORT_SYMBOL(i2c_8xx_del_bus);
+
+MODULE_AUTHOR("Brad Parker <brad@xxxxxxxxxxxx>");
+MODULE_DESCRIPTION("I2C-Bus MPC8XX algorithm");
+MODULE_LICENSE("GPL");
diff --git a/drivers/i2c/busses/Kconfig b/drivers/i2c/busses/Kconfig
index ece31d2..df29fb5 100644
--- a/drivers/i2c/busses/Kconfig
+++ b/drivers/i2c/busses/Kconfig
@@ -365,7 +365,7 @@ config I2C_PROSAVAGE

config I2C_RPXLITE
tristate "Embedded Planet RPX Lite/Classic support"
- depends on (RPXLITE || RPXCLASSIC) && I2C
+ depends on (RPXLITE || RPXCLASSIC || MPC86XADS ||
MPC885ADS) && I2C select I2C_ALGO8XX
```

A recent change to the i2c Kconfig files make it no longer needed to have all options depend on I2C explicitly. When you regenerate your patch on top of Linus' latest tree, you'll have to update this.

The help text for this option is still missing.

OK, good point.

[snip]

Question: once this is merged, will we possibly get rid of arch/ppc/boot/simple/iic.c?

yes of course. In fact, this layer being obsoleted by powerpc works, and may be broken in numerous ways (I am regularly asked why make fail somewhere in arch/ppc/boot, and common ans is just to use uImage that does not compile this part).

There still may be boards that neither have u-boot as firmware, nor able to update what's existing. I think declare stuff as feature to be removed would be enough for concerned folks to react. OTOH, there's good chance that nobody will even care :)

Re: [PATCH] i2c: adds support for i2c bus on 8xx

Re: [PATCH] i2c: adds support for i2c bus on 8xx

--

Sincerely, Vitaly

-

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/>