Ubuntu 系统 16.04 与 18.10 版本对比

行首无符号表示代码一样，行首 - 号表示是 16.04 版本的代码；行首 + 号表示是 18.10 版本的代码

1. hm610\_dvb.c 文件

第 163 行

break;

case SEC\_VOLTAGE\_OFF:

default:

- //GPIO\_VALUE |= ~HM610\_GPIO\_PIN(rf\_in, 0);

+

+ GPIO\_VALUE |= ~HM610\_GPIO\_PIN(rf\_in, 0);

break;

}

1. hm610\_i2c.c 文件

第 99 行

bytes\_in\_fifo = pci\_read(HM610\_I2C\_BASE, HM610\_RFO\_REG\_OFFSET) + 1;

+ dev\_dbg(i2c->i2c\_adap.dev.parent,

+ "%s entry, bytes in fifo: %d, msg: %d, SR: 0x%x, CR: 0x%x\n",

+ \_\_func\_\_, bytes\_in\_fifo, xiic\_rx\_space(i2c),

+ pci\_read(HM610\_I2C\_BASE, HM610\_SR\_REG\_OFFSET),

+ pci\_read(HM610\_I2C\_BASE, HM610\_CR\_REG\_OFFSET));

+

if (bytes\_in\_fifo > xiic\_rx\_space(i2c))

bytes\_in\_fifo = xiic\_rx\_space(i2c);

第 128 行

len = (len > fifo\_space) ? fifo\_space : len;

+ dev\_dbg(i2c->i2c\_adap.dev.parent, "%s entry, len: %d, fifo space: %d\n", \_\_func\_\_, len, fifo\_space);

+

while (len--) {

u16 data = i2c->tx\_msg->buf[i2c->tx\_pos++];

if ((xiic\_tx\_space(i2c) == 0) && (i2c->nmsgs == 1)) {

data |= HM610\_TX\_DYN\_STOP\_MASK;

+ dev\_dbg(i2c->i2c\_adap.dev.parent, "%s TX STOP\n", \_\_func\_\_);

}

pci\_write(HM610\_I2C\_BASE, HM610\_DTR\_REG\_OFFSET, data);

第 154 行

u32 pend, isr, ier;

u32 clr = 0;

- u32 cr, sr;

- u32 rx\_fifo\_ocy, rx\_fifo\_pirq;

-

+ mutex\_lock(&i2c->lock);

isr = pci\_read(HM610\_I2C\_BASE, HM610\_IISR\_OFFSET);

ier = pci\_read(HM610\_I2C\_BASE, HM610\_IIER\_OFFSET);

- mutex\_lock(&i2c->lock);

-

pend = isr & ier;

+ dev\_dbg(i2c->i2c\_adap.dev.parent, "%s: IER: 0x%x, ISR: 0x%x, pend: 0x%x\n",

+ \_\_func\_\_, ier, isr, pend);

+ dev\_dbg(i2c->i2c\_adap.dev.parent, "%s: SR: 0x%x, msg: %p, nmsgs: %d\n",

+ \_\_func\_\_, pci\_read(HM610\_I2C\_BASE, HM610\_SR\_REG\_OFFSET),

+ i2c->tx\_msg, i2c->nmsgs);

+

if ((pend & HM610\_INTR\_ARB\_LOST\_MASK) ||

((pend & HM610\_INTR\_TX\_ERROR\_MASK) &&

!(pend & HM610\_INTR\_RX\_FULL\_MASK))) {

- xiic\_reinit(i2c);

+ dev\_dbg(i2c->i2c\_adap.dev.parent, "%s error\n", \_\_func\_\_);

+ xiic\_reinit(i2c);

if (i2c->rx\_msg)

- {

xiic\_wakeup(i2c, STATE\_ERROR);

- printk(KERN\_INFO "pend1 = %d\n", pend);

- printk(KERN\_INFO "ier = %d\n", ier);

- printk(KERN\_INFO "isr = %d\n", isr);

- printk(KERN\_INFO "cr = %d\n", pci\_read(HM610\_I2C\_BASE, HM610\_CR\_REG\_OFFSET));

- printk(KERN\_INFO "sr = %d\n", pci\_read(HM610\_I2C\_BASE, HM610\_SR\_REG\_OFFSET));

-

- printk(KERN\_INFO "rx\_fifo\_ocy = %d\n", pci\_read(HM610\_I2C\_BASE, HM610\_RFO\_REG\_OFFSET));

- printk(KERN\_INFO "rx\_fifo\_pirq = %d\n", pci\_read(HM610\_I2C\_BASE, HM610\_RFD\_REG\_OFFSET));

- }

if (i2c->tx\_msg)

- {

xiic\_wakeup(i2c, STATE\_ERROR);

- printk(KERN\_INFO "pend2 = %d\n", pend);

- }

}

if (pend & HM610\_INTR\_RX\_FULL\_MASK) {

第 194 行

clr |= HM610\_INTR\_RX\_FULL\_MASK;

if (!i2c->rx\_msg) {

+ dev\_dbg(i2c->i2c\_adap.dev.parent,

+ "%s unexpexted RX IRQ\n", \_\_func\_\_);

xiic\_clear\_rx\_fifo(i2c);

goto out;

}

第 205 行

clr |= (isr & HM610\_INTR\_TX\_ERROR\_MASK);

+ dev\_dbg(i2c->i2c\_adap.dev.parent,

+ "%s end of message, nmsgs: %d\n",

+ \_\_func\_\_, i2c->nmsgs);

+

+

if (i2c->nmsgs > 1) {

i2c->nmsgs--;

i2c->tx\_msg++;

+ dev\_dbg(i2c->i2c\_adap.dev.parent,

+ "%s will start next...\n", \_\_func\_\_);

+

\_\_xiic\_start\_xfer(i2c);

}

}

第 227 行

xiic\_tx\_space(i2c) == 0)

xiic\_wakeup(i2c, STATE\_DONE);

else

- {

xiic\_wakeup(i2c, STATE\_ERROR);

- printk(KERN\_INFO "pend3 = %d\n", pend);

- printk(KERN\_INFO "i2c->nmsgs = %d, xiic\_tx\_space(i2c)= %d\n", i2c->nmsgs, xiic\_tx\_space(i2c));

- }

}

if (pend & (HM610\_INTR\_TX\_EMPTY\_MASK | HM610\_INTR\_TX\_HALF\_MASK)) {

第 240 行

(HM610\_INTR\_TX\_EMPTY\_MASK | HM610\_INTR\_TX\_HALF\_MASK));

if (!i2c->tx\_msg) {

+ dev\_dbg(i2c->i2c\_adap.dev.parent,

+ "%s unexpexted TX IRQ\n", \_\_func\_\_);

goto out;

}

第 248 行

if (!xiic\_tx\_space(i2c) && xiic\_tx\_fifo\_space(i2c) >= 2) {

+ dev\_dbg(i2c->i2c\_adap.dev.parent,

+ "%s end of message sent, nmsgs: %d\n",

+ \_\_func\_\_, i2c->nmsgs);

+

if (i2c->nmsgs > 1) {

i2c->nmsgs--;

i2c->tx\_msg++;

+

\_\_xiic\_start\_xfer(i2c);

} else {

xiic\_irq\_dis(i2c, HM610\_INTR\_TX\_HALF\_MASK);

+

+ dev\_dbg(i2c->i2c\_adap.dev.parent,

+ "%s Got TX IRQ but no more to do...\n",

+ \_\_func\_\_);

}

} else if (!xiic\_tx\_space(i2c) && (i2c->nmsgs == 1)){

xiic\_irq\_dis(i2c, HM610\_INTR\_TX\_HALF\_MASK);

第 261 行

}

out:

+ dev\_dbg(i2c->i2c\_adap.dev.parent, "%s clr: 0x%x\n", \_\_func\_\_, clr);

+

pci\_write(HM610\_I2C\_BASE, HM610\_IISR\_OFFSET, clr);

+

mutex\_unlock(&i2c->lock);

return 0;

}

第 320 行

xiic\_irq\_clr(i2c, HM610\_INTR\_TX\_ERROR\_MASK);

+ dev\_dbg(i2c->i2c\_adap.dev.parent, "%s entry, msg: %p, len: %d",

+ \_\_func\_\_, msg, msg->len);

+ dev\_dbg(i2c->i2c\_adap.dev.parent, "%s entry, ISR: 0x%x, CR: 0x%x\n",

+ \_\_func\_\_, pci\_read(HM610\_I2C\_BASE, HM610\_IISR\_OFFSET),

+ pci\_read(HM610\_I2C\_BASE, HM610\_CR\_REG\_OFFSET));

+

if (!(msg->flags & I2C\_M\_NOSTART)) {

u16 data = ((msg->addr << 1) & 0xfe) | HM610\_WRITE\_OPERATION |

HM610\_TX\_DYN\_START\_MASK;

第 340 行

int first = 1;

int fifo\_space = xiic\_tx\_fifo\_space(i2c);

+ dev\_dbg(i2c->i2c\_adap.dev.parent, "%s entry, msg: %p, fifos space: %d\n",

+ \_\_func\_\_, i2c->tx\_msg, fifo\_space);

+

if (!i2c->tx\_msg)

return;

第 391 行

struct hm610\_dev \*dev = i2c->dev;

int err;

+ dev\_dbg(i2c\_adap->dev.parent, "%s entry SR: 0x%x\n", \_\_func\_\_,

+ pci\_read(HM610\_I2C\_BASE, HM610\_SR\_REG\_OFFSET));

+

+

err = xiic\_busy(i2c);

if (err)

goto out;

第 404 行

if (wait\_event\_timeout(i2c->wq, (i2c->state == STATE\_ERROR) ||

(i2c->state == STATE\_DONE), HZ)) {

err = (i2c->state == STATE\_DONE) ? num : -EIO;

- if(i2c -> state == STATE\_ERROR)

- pr\_err("STATE\_ERROR i2c error!\n");

} else {

i2c->tx\_msg = NULL;

i2c->rx\_msg = NULL;

i2c->nmsgs = 0;

err = -ETIMEDOUT;

- pr\_err("xiic xfer i2c read error 2\n");

+ pr\_err("i2c read error 2\n");

}

out:

+

return err;

}

1. hm610.h 文件一致
2. hm610\_cards.c 文件一致
3. hm610\_core.c 文件一致
4. hm610\_dma.c 文件一致
5. hm610\_dma.h 文件一致
6. hm610\_regs.h 文件一致