Re: [PATCH 2.6.18 V6] drivers: add lcd display support

Source: http://linux.derkeiler.com/Mailing−Lists/Kernel/2006−09/msg09136.html

- *From*: Randy Dunlap <rdunlap@xxxxxxxxxxxx>
- *Date*: Sat, 30 Sep 2006 12:35:47 −0700

On Sat, 30 Sep 2006 13:22:53 +0000 Miguel Ojeda Sandonis wrote:

> Patched files Index
> −−−−−−−−−−−−−−−−−−

This list should be done with 'diffstat −p1 −w70 patch_file_name'
so that we can see the files and the patch sizes.

> patching file drivers/Kconfig
> patching file drivers/lcddisplay/cfag12864b.c
> patching file drivers/lcddisplay/cfag12864b_image.h
> patching file drivers/lcddisplay/Kconfig
> patching file drivers/lcddisplay/ks0108.c
> patching file drivers/lcddisplay/lcddisplay.c
> patching file drivers/lcddisplay/Makefile
> patching file drivers/Makefile
> patching file include/linux/cfag12864b.h
> patching file include/linux/ks0108.h
> patching file include/linux/lcddisplay.h
> patching file Documentation/lcddisplay/cfag12864b
> patching file Documentation/lcddisplay/lcddisplay
> patching file Documentation/ioctl−number.txt
> patching file CREDITS
> patching file MAINTAINERS
>
> −−−
> diff −uprN −X dontdiff linux−2.6.18−vanilla/drivers/lcddisplay/cfag12864b.c
> linux−2.6.18/drivers/lcddisplay/cfag12864b.c
> −−− linux−2.6.18−vanilla/drivers/lcddisplay/cfag12864b.c 1970−01−01 00:00:00.000000000
> +0000
> +++ linux−2.6.18/drivers/lcddisplay/cfag12864b.c 2006−09−30 10:56:32.000000000 +0000
> @@ −0,0 +1,592 @@
> +
> +#define CFAG12864B_NAME "cfag12864b"
> +

```
+/*
+ * Device
+ */
+
+static const unsigned int cfag12864b_firstminor;
+static const unsigned int cfag12864b_ndevices = 1;
```

This driver only supports one device, right?
Is that documented somewhere? I probably missed it.

```
+static int cfag12864b_major;
+static int cfag12864b_minor;
+static dev_t cfag12864b_device;
+struct cdev cfag12864b_chardevice;
+DECLARE_MUTEX(cfag12864b_mutex);
+
+/*
+ * cfag12864b Commands
+ */
+
+#define bit(n) (((unsigned char)1)<<(n))
+
+static unsigned char cfag12864b_state;
+
+static void cfag12864b_set(void)
+{
+ ks0108_writecontrol(cfag12864b_state);
+}
+
+static void cfag12864b_setbit(unsigned char state, unsigned char n)
+{
+ if (state)
+ cfag12864b_state |= bit(n);
+ else
+ cfag12864b_state &= ~bit(n);
+ cfag12864b_set();
+}
+
+static void cfag12864b_e(unsigned char state)
+{
+ cfag12864b_setbit(state, 0);
```

bit defintions? (here and below)

```
+}
+
+static void cfag12864b_cs1(unsigned char state)
```

```
+{
+ cfag12864b_setbit(state, 2);
+}
+
+static void cfag12864b_cs2(unsigned char state)
+{
+ cfag12864b_setbit(state, 1);
+}
+
+static void cfag12864b_di(unsigned char state)
+{
+ cfag12864b_setbit(state, 3);
+}
+
+static void cfag12864b_setcontrollers(unsigned char first, unsigned char second)
+{
+ if (first)
+ cfag12864b_cs1(0);
+ else
+ cfag12864b_cs1(1);
+
+ if (second)
+ cfag12864b_cs2(0);
+ else
+ cfag12864b_cs2(1);
+}
+
+static void cfag12864b_controller(unsigned char which)
+{
+ if (which == 0)
+ cfag12864b_setcontrollers(1, 0);
+ else if (which == 1)
+ cfag12864b_setcontrollers(0, 1);
+}
+
+/*
+ * Auxiliary
+ */
+
+static void normalizeoffset(unsigned int * offset)
```

Kernel style is:
static void normalizeoffset(unsigned int *offset)

```
+{
+ if (*offset >= CFAG12864B_PAGES*CFAG12864B_ADDRESSES)
+ *offset −= CFAG12864B_PAGES*CFAG12864B_ADDRESSES;
+}
+
```

```
+static unsigned char calcaddress(unsigned int offset)
+{
+ normalizeoffset(&offset);
+ return offset%CFAG12864B_ADDRESSES;
```

spaces around '%'

```
+}
+
+static unsigned char calccontroller(unsigned int offset)
+{
+ if (offset < CFAG12864B_PAGES*CFAG12864B_ADDRESSES)
+ return 0;
+ return 1;
+}
+
+static unsigned char calcpage(unsigned int offset)
+{
+ normalizeoffset(&offset);
+ return offset/CFAG12864B_ADDRESSES;
```

spaces around '/'

```
+}
+
+void cfag12864b_format_nolock(unsigned char *src)
+{
+ unsigned short i,j,k,n;
```

spaces after commas

```
+ unsigned char *dest;
+
+ dest = kmalloc(sizeof(unsigned char) * CFAG12864B_SIZE, GFP_KERNEL);
```

Are there places where sizeof(unsigned char) is not 1?

```
+ if (dest == NULL) {
+ printk(KERN_ERR CFAG12864B_NAME ": " "format: ERROR: "
+ "can't alloc memory %i bytes\n",
```

use "%zd" to print sizeof() values, otherwise gcc says:

drivers/lcddisplay/cfag12864b.c:271: warning: format '%i' expects type 'int', but argument 2 has type 'long unsigned int'

```
+ sizeof(unsigned char) * CFAG12864B_SIZE);
+ return;
+ }
+
+ for (i = 0; i < CFAG12864B_CONTROLLERS; i++)
+ for (j = 0; j < CFAG12864B_PAGES; j++)
+ for (k = 0; k < CFAG12864B_ADDRESSES; k++) {
```

questionable indentation

```
+ dest[(i * CFAG12864B_PAGES + j) * CFAG12864B_ADDRESSES + k] = 0;
+ for (n=0; n < 8; n++)
```

spaces around '='

```
+ if (src[i * CFAG12864B_ADDRESSES + k + (j * 8 + n) * CFAG12864B_WIDTH])
+ dest[(i * CFAG12864B_PAGES + j) * CFAG12864B_ADDRESSES + k] |= bit(n);
+ }
+
+ cfag12864b_write_nolock(0, dest, CFAG12864B_SIZE);
+
+ kfree(dest);
+}
+
+/*
+ * cfag12864b Exported Commands (do lock)
+ */
+
+void cfag12864b_on(void)
+{
+ if(down_interruptible(&cfag12864b_mutex))
```

space after "if"

```
+ return;
+
+ cfag12864b_on_nolock();
+
+ up(&cfag12864b_mutex);
+}
```

```
+
+void cfag12864b_off(void)
+{
+ if(down_interruptible(&cfag12864b_mutex))
```

space after "if"

```
+ return;
+
+ cfag12864b_off_nolock();
+
+ up(&cfag12864b_mutex);
+}
+
+void cfag12864b_clear(void)
+{
+ if(down_interruptible(&cfag12864b_mutex))
```

ditto

```
+ return;
+
+ cfag12864b_clear_nolock();
+
+ up(&cfag12864b_mutex);
+}
+
+void cfag12864b_write(unsigned short offset, const unsigned char *buffer,
+ unsigned short count)
+{
+ if(down_interruptible(&cfag12864b_mutex))
```

ditto

```
+ return;
+
+ cfag12864b_write_nolock(offset,buffer,count);
+
+ up(&cfag12864b_mutex);
+}
+
+void cfag12864b_format(unsigned char *src)
+{
+ if(down_interruptible(&cfag12864b_mutex))
```

ditto

```
+ return;
+
+ cfag12864b_format_nolock(src);
+
+ up(&cfag12864b_mutex);
+}
+
+EXPORT_SYMBOL_GPL(cfag12864b_on);
+EXPORT_SYMBOL_GPL(cfag12864b_off);
+EXPORT_SYMBOL_GPL(cfag12864b_clear);
+EXPORT_SYMBOL_GPL(cfag12864b_write);
+EXPORT_SYMBOL_GPL(cfag12864b_format);
+
+/*
+ * cfag12864b ioctls (don't lock because ioctl fop do)
+ */
+
+static int cfag12864b_fopioctlformat(void __user * arg)
```

use "*arg" without space

```
+{
+ int result;
+ int ret = −ENOTTY;
+ unsigned char *tmpbuffer;
+
+ tmpbuffer = kmalloc(
+ sizeof(unsigned char)*CFAG12864B_MATRIXSIZE,GFP_KERNEL);
```

spacebar helps readability. add spaces around * and after comma.

```
+ if (tmpbuffer == NULL) {
+ printk(KERN_ERR CFAG12864B_NAME ": " "FOP ioctl: ERROR: "
+ "can't alloc memory %i bytes\n",
```

use "%zd" to print sizeof() values, otherwise gcc complains.

```
+ sizeof(unsigned char)*CFAG12864B_MATRIXSIZE);
```

space around '*'

```
+ goto none;
+ }
+
+ result = copy_from_user(tmpbuffer, arg,
+ sizeof(unsigned char) * CFAG12864B_MATRIXSIZE);
+ if (result != 0) {
+ printk(KERN_ERR CFAG12864B_NAME ": " "FOP ioctl: ERROR: "
+ "can't copy memory from user\n");
```

I don't think that we want a printk on every copy_from_user() error.
(here and elsewhere)

```
+ goto bufferalloced;
+ }
+
+ cfag12864b_format_nolock(tmpbuffer);
+
+ ret = 0;
+
+bufferalloced:
+ kfree(tmpbuffer);
+
+none:
+ return ret;
+}
+
+/*
+ * cfag12864b_fops (do lock)
+ */
+
+static loff_t cfag12864b_fopseek(struct file *filp, loff_t offset, int whence)
+{
+ loff_t ret = −EINVAL;
+
+ if(down_interruptible(&cfag12864b_mutex))
```

space after "if"

```
+ return −ERESTARTSYS;
+
+ switch(whence) {
+ case SEEK_SET:
+ ret = offset;
+ break;
+ case SEEK_CUR:
```

```
+ ret = filp−>f_pos + offset;
+ break;
+ case SEEK_END:
+ ret = CFAG12864B_SIZE + offset;
+ break;
+ }
+
+ if (ret < 0) {
+ ret = −EINVAL;
+ goto none;
+ }
+
+ filp−>f_pos = ret;
+
+none:
+ up(&cfag12864b_mutex);
+ return ret;
+}
+
+
+static ssize_t cfag12864b_fopwrite(struct file *filp,
+ const char __user *buffer, size_t count, loff_t *offset)
+{
+ int ret = −EINVAL;
+ int result;
+ unsigned char *tmpbuffer;
+
+ if(down_interruptible(&cfag12864b_mutex))
```

space

```
+ return −ERESTARTSYS;
+
+ if (*offset > CFAG12864B_SIZE) {
+ ret = 0;
+ goto none;
+ }
+ if (*offset + count > CFAG12864B_SIZE)
+ count = CFAG12864B_SIZE−*offset;
+
+ tmpbuffer = kmalloc(count, GFP_KERNEL);
```

I would be very tempted to allocate a buffer at (open or init?) time,
of size CFAG12864B_SIZE. It could be used here for fopwrite and
in format_nolock without having to call kmalloc() repeatedly
for those operations. However, fopioctlformat would still need
to allocate its larger buffer.

```
+ if (tmpbuffer == NULL) {
+ printk(KERN_ERR CFAG12864B_NAME ": " "FOP write: ERROR: "
+ "can't alloc memory %i bytes\n",count);
```

use "%zd" to print sizeof values, otherwise gcc complains.
use space after comma.

```
+ ret = −ENOMEM;
+ goto none;
+ }
+
+ result = copy_from_user(tmpbuffer, buffer, count);
+ if (result != 0) {
+ printk(KERN_ERR CFAG12864B_NAME ": " "FOP write: ERROR: "
+ "can't copy memory from user\n");
+ ret = −EFAULT;
+ goto bufferalloced;
+ }
+
+ cfag12864b_write_nolock(*offset, tmpbuffer, count);
+
+ *offset += count;
+ ret = count;
+
+bufferalloced:
+ kfree(tmpbuffer);
+
+none:
+ up(&cfag12864b_mutex);
+ return ret;
+}
+
+static int cfag12864b_fopioctl(struct inode * inode, struct file * filp,
+ unsigned int cmd, unsigned long arg)
+{
+ int ret = −ENOTTY;
+
+ if(down_interruptible(&cfag12864b_mutex))
```

space after "if"

```
+ return −ERESTARTSYS;
+
+ if (_IOC_TYPE(cmd) != CFAG12864B_IOC_MAGIC)
+ goto none;
+ if (_IOC_NR(cmd) > CFAG12864B_IOC_MAXNR)
```

```
+ goto none;
+
+ switch(cmd) {
+ case CFAG12864B_IOCON:
+ cfag12864b_on_nolock();
+ ret = 0;
+ break;
+ case CFAG12864B_IOCOFF:
+ cfag12864b_off_nolock();
+ ret = 0;
+ break;
+ case CFAG12864B_IOCCLEAR:
+ cfag12864b_clear_nolock();
+ ret = 0;
+ break;
+ case CFAG12864B_IOCFORMAT:
+ ret = cfag12864b_fopioctlformat((void __user *)arg);
+ }
+
+none:
+ up(&cfag12864b_mutex);
+ return ret;
+}
+
+static const struct file_operations cfag12864b_fops =
+{
+ .owner = THIS_MODULE,
+ .llseek = cfag12864b_fopseek,
+ .write = cfag12864b_fopwrite,
+ .ioctl = cfag12864b_fopioctl,
+};
+
+/*
+ * Module Init & Exit
+ */
+
+static int __init cfag12864b_init(void)
+{
+
+#include "cfag12864b_image.h"
+
+ int result;
+ int ret = −EINVAL;
+
+ result = alloc_chrdev_region(&cfag12864b_device, cfag12864b_firstminor,
+ cfag12864b_ndevices, CFAG12864B_NAME);
+ if (result < 0) {
+ printk(KERN_ERR CFAG12864B_NAME ": " "ERROR: "
+ "can't alloc the char device region\n");
+ ret = result;
+ goto none;
```

```
+ }
+
+ cfag12864b_major = MAJOR(cfag12864b_device);
+ cfag12864b_minor = cfag12864b_firstminor;
+ cfag12864b_device = MKDEV(cfag12864b_major, cfag12864b_minor);
+
+ cfag12864b_clear_nolock();
+ cfag12864b_on_nolock();
+ cfag12864b_write_nolock(0, cfag12864b_image, CFAG12864B_SIZE);
+
+ cdev_init(&cfag12864b_chardevice,&cfag12864b_fops);
+ cfag12864b_chardevice.owner = THIS_MODULE;
+ cfag12864b_chardevice.ops = &cfag12864b_fops;
+ result = cdev_add(&cfag12864b_chardevice, cfag12864b_device,
+ cfag12864b_ndevices);
+ if (result < 0) {
+ printk(KERN_ERR CFAG12864B_NAME ": " "ERROR: "
+ "unable to add a new char device\n");
+ ret = result;
+ goto regionalloced;
+ }
+
+ if(class_device_create(lcddisplay_class, NULL, cfag12864b_device, NULL,
```

space after "if"

```
+ "cfag12864b%d", cfag12864b_minor) == NULL) {
+ printk(KERN_ERR CFAG12864B_NAME ": " "ERROR: "
+ "unable to create a device class\n");
+ ret = −EINVAL;
+ goto cdevadded;
+ }
+
+ printk(KERN_INFO CFAG12864B_NAME ": " "Inited\n");
```

KERN_DEBUG ?

```
+
+ return 0;
+
+cdevadded:
+ cdev_del(&cfag12864b_chardevice);
+
+regionalloced:
+ unregister_chrdev_region(cfag12864b_device, cfag12864b_ndevices);
+
+none:
```

```
+ return ret;
+}
+
+static void __exit cfag12864b_exit(void)
+{
+ cfag12864b_off_nolock();
+
+ class_device_destroy(lcddisplay_class, cfag12864b_device);
+ cdev_del(&cfag12864b_chardevice);
+ unregister_chrdev_region(cfag12864b_device, cfag12864b_ndevices);
+
+ printk(KERN_INFO CFAG12864B_NAME ": " "Exited\n");
```

KERN_DEBUG ?

```
+}
+
+module_init(cfag12864b_init);
+module_exit(cfag12864b_exit);
+
+MODULE_LICENSE("GPL");
+MODULE_AUTHOR("Miguel Ojeda Sandonis <maxextreme@xxxxxxxxx>");
+MODULE_DESCRIPTION("cfag12864b");
```

That's not a useful MODULE_DESCRIPTION.

```
diff −uprN −X dontdiff linux−2.6.18−vanilla/drivers/lcddisplay/cfag12864b_image.h
linux−2.6.18/drivers/lcddisplay/cfag12864b_image.h
−−− linux−2.6.18−vanilla/drivers/lcddisplay/cfag12864b_image.h 1970−01−01
00:00:00.000000000 +0000
+++ linux−2.6.18/drivers/lcddisplay/cfag12864b_image.h 2006−09−28 19:59:11.000000000
+0000
@@ −0,0 +1,95 @@
+#ifndef _CFAG12864B_IMAGE_H_
+#define _CFAG12864B_IMAGE_H_
+
+const unsigned char cfag12864b_image[] = {
```

What is in this image array? and what format is it in?
It's not just advertising/logo material, is it?
Is it really needed in cfag12864b_init()?

[bits deleted]

> diff −uprN −X dontdiff linux−2.6.18−vanilla/drivers/lcddisplay/Kconfig
> linux−2.6.18/drivers/lcddisplay/Kconfig
> −−− linux−2.6.18−vanilla/drivers/lcddisplay/Kconfig 1970−01−01 00:00:00.000000000
> +0000
> +++ linux−2.6.18/drivers/lcddisplay/Kconfig 2006−09−30 02:52:24.000000000 +0000
> @@ −0,0 +1,110 @@
> +#
> +# For a description of the syntax of this configuration file,
> +# see Documentation/kbuild/kconfig−language.txt.
> +#
> +# LCD Display drivers configuration.
> +#
> +# Maintainer: Miguel Ojeda Sandonis <maxextreme@xxxxxxxxx>
> +#

Don't need any of those comments above.

> +menu "LCD Display support"
> +
> +config LCDDISPLAY
> + tristate "LCD Display support"
> + default n
> + −−−help−−−
> + If you have a LCD display, say Y.
> +
> + To compile this as a module, choose M here:
> + module will be called lcddisplay.

"the module will be..."

> + Most LCD drivers use a I/O port (like the parallel port)

use an I/O port

> + so you will need to say Y or M at them if you want to see

say Y or M for them

> + more options in this menu.
> +
> + If unsure, say N.
> +
> + Maintainer: Miguel Ojeda Sandonis <maxextreme@xxxxxxxxx>

Just in the MAINTAINERS file.

> +comment "Parallel port dependent:"
> +
> +config KS0108
> + tristate "KS0108 LCD Controller"
> + depends on LCDDISPLAY && PARPORT
> + default n
> + −−−help−−−
> + If you have a LCD display controlled by one or more KS0108

have an LCD display

> + controllers, say Y. You will need also another more specific
> + driver for your LCD.
> +
> + Depends on Parallel Port support. If you say Y at
> + parport, you will be able to compile this as a module (M)
> + and built−in as well (Y). If you said M at parport,

s/and/or/

> + you will be able only to compile this as a module (M).

However, that is true everywhere, so we usually don't say it.

> + To compile this as a module, choose M here:
> + module will be called ks0108.

"the module will be..."

> + If unsure, say N.
> +
> + Maintainer: Miguel Ojeda Sandonis <maxextreme@xxxxxxxxx>

Just in MAINTAINERS file.

> +config KS0108_PORT
> + hex "Parallel port where the LCD is connected to"

...where the LCD is connected"

+ depends on KS0108
+ default 0x378
+ −−−help−−−
+ The address of the parallel port where the LCD is connected to.

... where the LCD is connected.

+ The first standard parallel port address is 0x378.
+ The second standard parallel port address is 0x278.
+ The third standard parallel port address is 0x3BC.
+
+ You can specify a different address if you need.
+
+ If you don't know what I'm talking about, load the parport module,
+ and execute "dmesg". You can see there how many parallel ports

comma after "parallel ports"

+ where detected and which address everyone has.

add comma, change wording, like: "where they are connected,"
"and which address each one has."

+ Usually you only need to use 0x378.
+
+ If you compile this as a module, you can still override this
+ using the module parameters.
+
+config KS0108_DELAY
+ int "Delay between each control writing (microseconds)"
+ depends on KS0108
+ default "2"
+ −−−help−−−
+ Amount of time the ks0108 should wait between each control write
+ to the parallel port.
+
+ If your driver seems to miss random writings, increment this.
+
+ If you don't know what I'm talking about, ignore it.
+
+ If you compile this as a module, you can still override this

+ using the module parameters.
+
+config CFAG12864B
+ tristate "CFAG12864B LCD Display"
+ depends on KS0108
+ default n
+ −−−help−−−
+ If you have a Crystalfontz 128x64 2−color LCD display,
+ cfag12864b Series, say Y. You also need the ks0108 LCD
+ Controller driver.
+
+ For help about how to wire your LCD to the parallel port,
+ check this image: http://www.skippari.net/lcd/sekalaista
+ /crystalfontz_cfag12864B−TMI−V.png

check this image: <newline and put URL all on one line>

+ To compile this as a module, choose M here:
+ module will be called cfag12864b.

"the module will be..."

+ If unsure, say N.
+
+ Maintainer: Miguel Ojeda Sandonis <maxextreme@xxxxxxxxx>

Only in MAINTAINERS file.

+endmenu
+
diff −uprN −X dontdiff linux−2.6.18−vanilla/drivers/lcddisplay/ks0108.c
linux−2.6.18/drivers/lcddisplay/ks0108.c
−−− linux−2.6.18−vanilla/drivers/lcddisplay/ks0108.c 1970−01−01 00:00:00.000000000
+0000
+++ linux−2.6.18/drivers/lcddisplay/ks0108.c 2006−09−30 12:41:31.000000000 +0000
@@ −0,0 +1,160 @@
+/*
+ * ks0108 Exported cmds (don't lock)
+ *
+ * you _should_ lock in the top driver, so
+ * this functions _should not_ get race conditions in any way.

these functions

```
+ * Locking for each byte here would be so slow and useless.
+ */
+
+#define bit(n) (((unsigned char)1)<<(n))
+
+void ks0108_writecontrol(unsigned char byte)
+{
+ udelay(ks0108_delay);
+ parport_write_control(ks0108_parport, byte ^ (bit(0) | bit(1) | bit(3)));
```

what does the xor do? what are the bits?

```
+}
+
+void ks0108_displaystate(unsigned char state)
+{
+ ks0108_writedata((state ? bit(0) : 0) | bit(1) | bit(2) | bit(3) | bit(4) | bit(5));
```

Are the state bits documented somewhere?
Can you use meaningful mnemonic names for the bits?

```
+}
+
+void ks0108_startline(unsigned char startline)
+{
+ ks0108_writedata(min(startline,(unsigned char)63) | bit(6) | bit(7));
```

bit definitions?

```
+}
+
+void ks0108_address(unsigned char address)
+{
+ ks0108_writedata(min(address,(unsigned char)63) | bit(6));
```

bit definition?

```
+}
+
+void ks0108_page(unsigned char page)
+{
+ ks0108_writedata(min(page,(unsigned char)7) | bit(3) | bit(4) | bit(5) | bit(7));
```

bit definitions?

```
+}
+
+static int __init ks0108_init(void)
+{
+ int result;
+ int ret = −EINVAL;
+
+ ks0108_parport = parport_find_base(ks0108_port);
+ if (ks0108_parport == NULL) {
+ printk(KERN_ERR KS0108_NAME ": " "ERROR: "
```

merge strings together, like:
printk(KERN_ERR KS0108_NAME ": ERROR: "
(throughout the source files)

```
+ "parport didn't find %i port\n",ks0108_port);
```

space after ","

```
+ goto none;
+ }
+
+ ks0108_pardevice = parport_register_device(ks0108_parport, KS0108_NAME,
+ NULL, NULL, NULL, PARPORT_DEV_EXCL, NULL);
+ if (ks0108_pardevice == NULL) {
+ printk(KERN_ERR KS0108_NAME ": " "ERROR: "
+ "parport didn't register new device\n");
+ goto none;
+ }
+
+ result = parport_claim(ks0108_pardevice);
+ if (result != 0) {
+ printk(KERN_ERR KS0108_NAME ": " "ERROR: "
+ "can't claim %i parport, maybe in use\n",ks0108_port);
```

space after comma

```
+ ret = result;
+ goto registered;
+ }
+
+ printk(KERN_INFO KS0108_NAME ": " "Inited − ks0108_port=0x%X
```

```
ks0108_delay=%i\n", ks0108_port, ks0108_delay);
+ return 0;
+
+registered:
+ parport_unregister_device(ks0108_pardevice);
+
+none:
+ return ret;
+}
+
+
+MODULE_LICENSE("GPL");
+MODULE_AUTHOR("Miguel Ojeda Sandonis <maxextreme@xxxxxxxxx>");
+MODULE_DESCRIPTION("ks0108");
```

Use a real MODULE_DESCRIPTION, please.

```
diff −uprN −X dontdiff linux−2.6.18−vanilla/drivers/lcddisplay/lcddisplay.c
linux−2.6.18/drivers/lcddisplay/lcddisplay.c
−−− linux−2.6.18−vanilla/drivers/lcddisplay/lcddisplay.c 1970−01−01 00:00:00.000000000
+0000
+++ linux−2.6.18/drivers/lcddisplay/lcddisplay.c 2006−09−28 19:59:18.000000000 +0000
@@ −0,0 +1,79 @@
+
+static int __init lcddisplay_init(void)
+{
+ int ret = −EINVAL;
+
+ lcddisplay_class = class_create(THIS_MODULE, LCDDISPLAY_NAME);
+ if (IS_ERR(lcddisplay_class)) {
+ printk(KERN_ERR LCDDISPLAY_NAME ": " "ERROR: "
```

The multiple quotation marks is a bit odd & difficult to read.
How about just doing:

printk(KERN_ERR LCDDISPLAY_NAME ": ERROR: "

(in MANY places)?

```
+ "can't create %s class\n", LCDDISPLAY_NAME);
+ goto none;
+ }
+
+ printk(KERN_INFO LCDDISPLAY_NAME ": " "Inited\n");
```

printk(KERN_INFO LCDDISPLAY_NAME ": Inited\n");

or better yet,
printk(KERN_DEBUG LCDDISPLAY_NAME ": Inited\n");

```
+ return 0;
+
+none:
+ return ret;
+}
+
+static void __exit lcddisplay_exit(void)
+{
+ class_destroy(lcddisplay_class);
+
+ printk(KERN_INFO LCDDISPLAY_NAME ": " "Exited\n");
```

KERN_DEBUG & merge the strings?

```
+}
+
+module_init(lcddisplay_init);
+module_exit(lcddisplay_exit);
+
+MODULE_LICENSE("GPL");
+MODULE_AUTHOR("Miguel Ojeda Sandonis <maxextreme@xxxxxxxxx>");
+MODULE_DESCRIPTION("lcddisplay");
```

Not a useful MODULE_DESCRIPTION.

```
diff −uprN −X dontdiff linux−2.6.18−vanilla/include/linux/cfag12864b.h
linux−2.6.18/include/linux/cfag12864b.h
−−− linux−2.6.18−vanilla/include/linux/cfag12864b.h 1970−01−01 00:00:00.000000000
+0000
+++ linux−2.6.18/include/linux/cfag12864b.h 2006−09−28 19:59:57.000000000 +0000
@@ −0,0 +1,58 @@
+
+#ifndef _CFAG12864B_H_
+#define _CFAG12864B_H_
+
+#include <linux/ioctl.h>
+
+#define CFAG12864B_WIDTH 128
+#define CFAG12864B_HEIGHT 64
+#define CFAG12864B_MATRIXSIZE CFAG12864B_WIDTH*CFAG12864B_HEIGHT
```

Use parens around defined expressions (above).

```
+#define CFAG12864B_CONTROLLERS 2
+#define CFAG12864B_PAGES 8
+#define CFAG12864B_ADDRESSES 64
+#define CFAG12864B_SIZE CFAG12864B_CONTROLLERS * \
+ CFAG12864B_PAGES * \
+ CFAG12864B_ADDRESSES
```

use parens.

```
+#define CFAG12864B_IOC_MAGIC 0xFF
+#define CFAG12864B_IOC_MAXNR 0x03
+
+#define CFAG12864B_IOCOFF _IO(CFAG12864B_IOC_MAGIC,0)
+#define CFAG12864B_IOCON _IO(CFAG12864B_IOC_MAGIC,1)
+#define CFAG12864B_IOCCLEAR _IO(CFAG12864B_IOC_MAGIC,2)
+#define CFAG12864B_IOCFORMAT _IOW(CFAG12864B_IOC_MAGIC,3,void *)
+
+extern void cfag12864b_on(void);
+extern void cfag12864b_off(void);
+extern void cfag12864b_clear(void);
+extern void cfag12864b_write(unsigned short offset,
+ const unsigned char *buffer, unsigned short count);
+extern void cfag12864b_format(unsigned char *src);
```

All function prototypes that have parameters (above):
kernel style is to use type + param_name, not just type.

```
+#endif /* _CFAG12864B_H_ */
+
diff −uprN −X dontdiff linux−2.6.18−vanilla/Documentation/lcddisplay/cfag12864b
linux−2.6.18/Documentation/lcddisplay/cfag12864b
−−− linux−2.6.18−vanilla/Documentation/lcddisplay/cfag12864b 1970−01−01
00:00:00.000000000 +0000
+++ linux−2.6.18/Documentation/lcddisplay/cfag12864b 2006−09−30 12:20:28.000000000
+0000
@@ −0,0 +1,371 @@
+
+−−−−−−−−−
+2. WIRING
+−−−−−−−−−
+
+The cfag12864b LCD Display Series don't have a official wiring.
```

... don't have official wiring.

+The common wiring is done to the parallel port:
+
+http://www.skippari.net/lcd/sekalaista/crystalfontz_cfag12864B−TMI−V.png
+
+You can get help at Crystalfontz and LCDInfo forums.
+
+

+3.1. ioctl & 128*64 boolean matrix
+−−−
+
+This method is easier, but you have to update the entire display
+each time you want to change it.
+
+Note:
+
+ CFAG12864B_FORMATSIZE ==
+ CFAG12864B_WIDTH * CFAG12864B_HEIGHT ==
+ 128 * 64
+
+Declare the matrix and other one:

other one what? another matrix buffer?

+ unsigned char MyDrawing[CFAG12864B_WIDTH][CFAG12864B_HEIGHT];
+
+ unsigned char Buffer[CFAG12864B_FORMATSIZE];
+
+Copy the 2d matrix to the buffer , like:

to the buffer,

+
+ for(i = 0; i < CFAG12864B_WIDTH; i++)
+ for(j = 0; j < CFAG12864B_HEIGHT; j++)
+ Buffer[i + j * CFAG12864B_WIDTH] = MyDrawing[i][j];
+
+Call the ioctl:
+
+ ioctl(fdisplay, CFAG12864B_IOCFORMAT, Buffer);
+
+Voila! Your drawing should appear on the screen.
+

+
+
+3.2. Direct writing
+−−−−−−−−−−−−−−−−−−
+
+This methods allows you to change each byte of the device,

This method

+so you can achieve a higher update rate updating only the pixels
+you are going to change.

+4.2 Example BMP writer
+−−−−−−−−−−−−−−−−−−−−
+
+You can take ideas from this code and start programming. I think it is useful
+for understanding how the driver can be used. It just work, don't expect

just works;

+good BMP−related code. I chose such bitmap format because it is simple.
+
+The program reads a .bmp 128x64 2−colors file, convert it to a

"converts it to a"

+boolean [128*64] buffer and then use ioctl to display it on the screen.

"then uses"

diff −uprN −X dontdiff linux−2.6.18−vanilla/Documentation/ioctl−number.txt
linux−2.6.18/Documentation/ioctl−number.txt
−−− linux−2.6.18−vanilla/Documentation/ioctl−number.txt 2006−09−20 03:42:06.000000000
+0000
+++ linux−2.6.18/Documentation/ioctl−number.txt 2006−09−27 19:15:13.000000000 +0000
@@ −191,3 +191,5 @@ Code Seq# Include File Comments
<mailto:aherrman@xxxxxxxxxx>
0xF3 00−3F video/sisfb.h sisfb (in development)
<mailto:thomas@xxxxxxxxxxxxxxx>
+0xFF 00−1F linux/cfag12864b.h cfag12864b LCD Display Driver
+ <mailto:maxextreme@xxxxxxxxx>

Also need additions to Documenation/ABI/ ?
Please read/check Documentation/ABI/README.
and please read/check Documentation/SubmitChecklist.

> diff −uprN −X dontdiff linux−2.6.18−vanilla/MAINTAINERS linux−2.6.18/MAINTAINERS
> −−− linux−2.6.18−vanilla/MAINTAINERS 2006−09−20 03:42:06.000000000 +0000
> +++ linux−2.6.18/MAINTAINERS 2006−09−27 19:15:13.000000000 +0000
> @@ −1707,6 +1707,11 @@ M: James.Bottomley@xxxxxxxxxxxxxxxxxxxxx
> L: linux−scsi@xxxxxxxxxxxxxxx
> S: Maintained
>
> +LCD DISPLAY DRIVERS
> +P: Miguel Ojeda Sandonis
> +M: maxextreme@xxxxxxxxx
> +S: Maintained
> +

Needs a mailing list (L:) entry.

−−−
~Randy
−
To unsubscribe from this list: send the line "unsubscribe linux−kernel" in
the body of a message to majordomo@xxxxxxxxxxxxxxx
More majordomo info at http://vger.kernel.org/majordomo−info.html
Please read the FAQ at http://www.tux.org/lkml/