

[PATCH 476] HP300 fb

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2004-10/10222.html>

From: Geert Uytterhoeven (geert_at_linux-m68k.org)

Date: 10/31/04

Date: Sun, 31 Oct 2004 11:03:35 +0100

To: Linus Torvalds <torvalds@osdl.org>, Andrew Morton <akpm@osdl.org>, Antonino Daplas <[HP300 frame buffer device updates from Kars de Jong:](mailto:adaplas@p</p></div><div data-bbox=)

- Updated to use the new DIO semantics
- Added support for DIO-II boards
- Added support for 8 bit Catseye boards
- Fixed colour map support
- Added fb_blank() implementation
- Added fb_fillrect() implementation
- Added fb_sync() implementation

Signed-off-by: Kars de Jong <jongk@linux-m68k.org>

Signed-off-by: Geert Uytterhoeven <geert@linux-m68k.org>

```
--- linux-2.6.10-rc1/drivers/video/hpfb.c 2003-05-05 19:44:39.000000000 +0200
+++ linux-m68k-2.6.10-rc1/drivers/video/hpfb.c 2004-07-14 13:19:21.000000000 +0200
@@ -1,6 +1,8 @@
/*
 * HP300 Topcat framebuffer support (derived from macfb of all things)
 * Phil Blundell <philb@gnu.org> 1998
+ * DIO-II, colour map and Catseye support by
+ * Kars de Jong <jongk@linux-m68k.org>, May 2004.
 */
```

```
#include <linux/module.h>
@@ -15,19 +17,36 @@
#include <linux/init.h>
#include <linux/fb.h>
#include <linux/dio.h>
+
#include <asm/io.h>
-#include <asm/blinken.h>
-#include <asm/hwtest.h>
+#include <asm/uaccess.h>

-static struct fb_info fb_info;
+static struct fb_info fb_info = {
+ .fix = {
+ .id = "HP300 ",
```

```

+ .type = FB_TYPE_PACKED_PIXELS,
+ .visual = FB_VISUAL_PSEUDOCOLOR,
+ .accel = FB_ACCEL_NONE,
+ }
+};

-unsigned long fb_regs;
-unsigned char fb_bitmask;
+static unsigned long fb_regs;
+static unsigned char fb_bitmask;

+#define TC_NBLANK 0x4080
#define TC_WEN 0x4088
#define TC_REN 0x408c
#define TC_FBEN 0x4090
-#define TC_NBLANK 0x4080
+#define TC_PRR 0x40ea
+
+/* These defines match the X window system */
+#define RR_CLEAR 0x0
+#define RR_COPY 0x3
+#define RR_NOOP 0x5
+#define RR_XOR 0x6
+#define RR_INVERT 0xa
+#define RR_COPYINVERTED 0xc
+#define RR_SET 0xf

/* blitter regs */
#define BUSY 0x4044
@@ -40,129 +59,255 @@
#define WWIDTH 0x4102
#define WMOVE 0x409c

-static struct fb_fix_screeninfo hpfb_fix __initdata = {
- .id = "HP300 Topcat",
- .smem_len = 1024*768,
- .type = FB_TYPE_PACKED_PIXELS,
- .visual = FB_VISUAL_PSEUDOCOLOR,
- .line_length = 1024,
- .accel = FB_ACCEL_NONE,
-};
-
- static struct fb_var_screeninfo hpfb_defined = {
- .xres = 1024,
- .yres = 768,
- .xres_virtual = 1024,
- .yres_virtual = 786,
- .bits_per_pixel = 1,
- .red = {0,2,0}, /* R */
- .green = {0,2,0}, /* G */
- .blue = {0,2,0}, /* B */

```

```

+ .red = {
+ .length = 8,
+ },
+ .green = {
+ .length = 8,
+ },
+ .blue = {
+ .length = 8,
+ },
    .activate = FB_ACTIVATE_NOW,
- .height = 274,
- .width = 195, /* 14" monitor */
- .accel_flags = FB_ACCEL_NONE,
+ .height = -1,
+ .width = -1,
    .vmode = FB_VMODE_NONINTERLACED,
};

-/*
- * Set the palette. This may not work on all boards but only experimentation
- * will tell.
- * XXX Doesn't work at all.
- */
static int hpfb_setcolreg(unsigned regno, unsigned red, unsigned green,
- unsigned blue, unsigned transp,
- struct fb_info *info)
+ unsigned blue, unsigned transp,
+ struct fb_info *info)
{
+ /* use MSBs */
+ unsigned char _red = red >> 8;
+ unsigned char _green = green >> 8;
+ unsigned char _blue = blue >> 8;
+ unsigned char _regno = regno;
+
+ /*
+ * Set a single color register. The values supplied are
+ * already rounded down to the hardware's capabilities
+ * (according to the entries in the `var' structure). Return
+ * != 0 for invalid regno.
+ */
+
+ if (regno >= info->cmap.len)
+ return 1;
+
    while (in_be16(fb_regs + 0x6002) & 0x4) udelay(1);
- out_be16(fb_regs + 0x60f0, 0);
- out_be16(fb_regs + 0x60b8, regno);
- out_be16(fb_regs + 0x60b2, red);
- out_be16(fb_regs + 0x60b4, green);
- out_be16(fb_regs + 0x60b6, blue);

```

```

+
+ out_be16(fb_regs + 0x60ba, 0xff);
+
+ out_be16(fb_regs + 0x60b2, _red);
+ out_be16(fb_regs + 0x60b4, _green);
+ out_be16(fb_regs + 0x60b6, _blue);
+ out_be16(fb_regs + 0x60b8, ~_regno);
+   out_be16(fb_regs + 0x60f0, 0xff);
+
+   udelay(100);
- out_be16(fb_regs + 0x60ba, 0xffff);
+
+ while (in_be16(fb_regs + 0x6002) & 0x4) udelay(1);
+ out_be16(fb_regs + 0x60b2, 0);
+ out_be16(fb_regs + 0x60b4, 0);
+ out_be16(fb_regs + 0x60b6, 0);
+ out_be16(fb_regs + 0x60b8, 0);
+
+ return 0;
+}
+
+/* 0 unblank, 1 blank, 2 no vsync, 3 no hsync, 4 off */
+
+static int hpfb_blank(int blank, struct fb_info *info)
+{
+ out_8(fb_regs + TC_NBLANK, (blank ? 0x00 : fb_bitmask));
+
+   return 0;
+ }

-void hpfb_copyarea(struct fb_info *info, const struct fb_copyarea *area)
+static void topcat_blit(int x0, int y0, int x1, int y1, int w, int h, int rr)
+ {
- while (in_8(fb_regs + BUSY) & fb_bitmask);
- out_8(fb_regs + WMRR, 0x3);
- out_be16(fb_regs + SOURCE_X, area->sx);
- out_be16(fb_regs + SOURCE_Y, area->sy);
- out_be16(fb_regs + DEST_X, area->dx);
- out_be16(fb_regs + DEST_Y, area->dy);
- out_be16(fb_regs + WHEIGHT, area->height);
- out_be16(fb_regs + WWIDTH, area->width);
+ if (rr >= 0)
+ {
+ while (in_8(fb_regs + BUSY) & fb_bitmask)
+ ;
+ }
+ out_8(fb_regs + TC_FBEN, fb_bitmask);
+ if (rr >= 0)
+ {
+ out_8(fb_regs + TC_WEN, fb_bitmask);
+ out_8(fb_regs + WMRR, rr);

```

```

+ }
+ out_be16(fb_regs + SOURCE_X, x0);
+ out_be16(fb_regs + SOURCE_Y, y0);
+ out_be16(fb_regs + DEST_X, x1);
+ out_be16(fb_regs + DEST_Y, y1);
+ out_be16(fb_regs + WWIDTH, w);
+ out_be16(fb_regs + WHEIGHT, h);
+ out_8(fb_regs + WMOVE, fb_bitmask);
+ }

+static void hpfb_copyarea(struct fb_info *info, const struct fb_copyarea *area)
+{
+ topcat_blit(area->sx, area->sy, area->dx, area->dy, area->width, area->height, RR_COPY);
+}
+
+static void hpfb_fillrect(struct fb_info *p, const struct fb_fillrect *region)
+{
+ u8 clr;
+
+ clr = region->color & 0xff;
+
+ while (in_8(fb_regs + BUSY) & fb_bitmask)
+ ;
+
+ /* Foreground */
+ out_8(fb_regs + TC_WEN, fb_bitmask & clr);
+ out_8(fb_regs + WMRR, (region->rop == ROP_COPY ? RR_SET : RR_INVERT));
+
+ /* Background */
+ out_8(fb_regs + TC_WEN, fb_bitmask & ~clr);
+ out_8(fb_regs + WMRR, (region->rop == ROP_COPY ? RR_CLEAR : RR_NOOP));
+
+ topcat_blit(region->dx, region->dy, region->dx, region->dy, region->width, region->height, -1);
+}
+
+static int hpfb_sync(struct fb_info *info)
+{
+ /*
+ * Since we also access the framebuffer directly, we have to wait
+ * until the block mover is finished
+ */
+ while (in_8(fb_regs + BUSY) & fb_bitmask)
+ ;
+
+ out_8(fb_regs + TC_WEN, fb_bitmask);
+ out_8(fb_regs + TC_PRR, RR_COPY);
+ out_8(fb_regs + TC_FBEN, fb_bitmask);
+
+ return 0;
+}
+

```

Linux–Kernel: [PATCH 476] HP300 fb

```

static struct fb_ops hpfb_ops = {
    .owner = THIS_MODULE,
    .fb_setcolreg = hpfb_setcolreg,
- .fb_fillrect = cfb_fillrect,
+ .fb_blank = hpfb_blank,
+ .fb_fillrect = hpfb_fillrect,
    .fb_copyarea = hpfb_copyarea,
    .fb_imageblit = cfb_imageblit,
    .fb_cursor = soft_cursor,
+ .fb_sync = hpfb_sync,
};

-#define TOPCAT_FBOMSB 0x5d
-#define TOPCAT_FBOLSB 0x5f
+/* Common to all HP framebuffer */
+#define HPFB_FBWMSB 0x05 /* Frame buffer width */
+#define HPFB_FBWLSB 0x07
+#define HPFB_FBHMSB 0x09 /* Frame buffer height */
+#define HPFB_FBHLSB 0x0b
+#define HPFB_DWMSB 0x0d /* Display width */
+#define HPFB_DWLSB 0x0f
+#define HPFB_DHMSB 0x11 /* Display height */
+#define HPFB_DHLSB 0x13
+#define HPFB_NUMPLANES 0x5b /* Number of colour planes */
+#define HPFB_FBOMSB 0x5d /* Frame buffer offset */
+#define HPFB_FBOLSB 0x5f

-int __init hpfb_init_one(unsigned long base)
+static int __init hpfb_init_one(unsigned long phys_base, unsigned long virt_base)
{
- unsigned long fboff;
+ unsigned long fboff, fb_width, fb_height, fb_start;
+
+ fb_regs = virt_base;
+ fboff = (in_8(fb_regs + HPFB_FBOMSB) << 8) | in_8(fb_regs + HPFB_FBOLSB);
+
+ fb_info.fix.smem_start = (in_8(fb_regs + fboff) << 16);
+
+ if (phys_base >= DIOII_BASE)
+ {
+ fb_info.fix.smem_start += phys_base;
+ }

- if (fb_get_options("hpfb", NULL))
- return -ENODEV;
-
- fboff = (in_8(base + TOPCAT_FBOMSB) << 8) | in_8(base + TOPCAT_FBOLSB);
+ if (DIO_SECID(fb_regs) != DIO_ID2_TOPCAT)
+ {
+ /* This is the magic incantation the HP X server uses to make Catseye boards work. */
+ while (in_be16(fb_regs+0x4800) & 1)

```

Linux-Kernel: [PATCH 476] HP300 fb

```

+ ;
+ out_be16(fb_regs+0x4800, 0); /* Catseye status */
+ out_be16(fb_regs+0x4510, 0); /* VB */
+ out_be16(fb_regs+0x4512, 0); /* TCNTRL */
+ out_be16(fb_regs+0x4514, 0); /* ACNTRL */
+ out_be16(fb_regs+0x4516, 0); /* PNCNTRL */
+ out_be16(fb_regs+0x4206, 0x90); /* RUG Command/Status */
+ out_be16(fb_regs+0x60a2, 0); /* Overlay Mask */
+ out_be16(fb_regs+0x60bc, 0); /* Ram Select */
+ }
+
+ /*
+ * Fill in the available video resolution
+ */
+ fb_width = (in_8(fb_regs + HPFB_FBWMSB) << 8) | in_8(fb_regs + HPFB_FBWLSB);
+ fb_info.fix.line_length = fb_width;
+ fb_height = (in_8(fb_regs + HPFB_FBHMSB) << 8) | in_8(fb_regs + HPFB_FBHLSB);
+ fb_info.fix.smem_len = fb_width * fb_height;
+ fb_start = (unsigned long)ioremap_writethrough(fb_info.fix.smem_start,
+ fb_info.fix.smem_len);
+ hpfb_defined.xres = (in_8(fb_regs + HPFB_DWMSB) << 8) | in_8(fb_regs + HPFB_DWLSB);
+ hpfb_defined.yres = (in_8(fb_regs + HPFB_DHMSB) << 8) | in_8(fb_regs + HPFB_DHLSB);
+ hpfb_defined.xres_virtual = hpfb_defined.xres;
+ hpfb_defined.yres_virtual = hpfb_defined.yres;
+ hpfb_defined.bits_per_pixel = in_8(fb_regs + HPFB_NUMPLANES);
+
+ printk(KERN_INFO "hpfb: framebuffer at 0x%lx, mapped to 0x%lx, size %dk\n",
+ fb_info.fix.smem_start, fb_start, fb_info.fix.smem_len/1024);
+ printk(KERN_INFO "hpfb: mode is %dx%dx%d, linelength=%d\n",
+ hpfb_defined.xres, hpfb_defined.yres, hpfb_defined.bits_per_pixel, fb_info.fix.line_length);

- hpfb_fix.smem_start = 0xf0000000 | (in_8(base + fboff) << 16);
- fb_regs = base;
-
-#if 0
- /* This is the magic incantation NetBSD uses to make Catseye boards work. */
- out_8(base+0x4800, 0);
- out_8(base+0x4510, 0);
- out_8(base+0x4512, 0);
- out_8(base+0x4514, 0);
- out_8(base+0x4516, 0);
- out_8(base+0x4206, 0x90);
-#endif
- /*
+ /*
+     * Give the hardware a bit of a prod and work out how many bits per
+     * pixel are supported.
+ */
-
- out_8(base + TC_WEN, 0xff);
- out_8(base + TC_FBEN, 0xff);

```

```

- out_8(hpfb_fix.smem_start, 0xff);
- fb_bitmask = in_8(hpfb_fix.smem_start);
+ out_8(fb_regs + TC_WEN, 0xff);
+ out_8(fb_regs + TC_PRR, RR_COPY);
+ out_8(fb_regs + TC_FBEN, 0xff);
+ out_8(fb_start, 0xff);
+ fb_bitmask = in_8(fb_start);
+ out_8(fb_start, 0);

    /*
     * Enable reading/writing of all the planes.
     */
- out_8(base + TC_WEN, fb_bitmask);
- out_8(base + TC_REN, fb_bitmask);
- out_8(base + TC_FBEN, fb_bitmask);
- out_8(base + TC_NBLANK, 0x1);
+ out_8(fb_regs + TC_WEN, fb_bitmask);
+ out_8(fb_regs + TC_PRR, RR_COPY);
+ out_8(fb_regs + TC_REN, fb_bitmask);
+ out_8(fb_regs + TC_FBEN, fb_bitmask);
+
+ /*
+ * Clear the screen.
+ */
+ topcat_blit(0, 0, 0, 0, fb_width, fb_height, RR_CLEAR);

    /*
     * Let there be consoles..
     */
+ if (DIO_SECID(fb_regs) == DIO_ID2_TOPCAT)
+ strcat(fb_info.fix.id, "Topcat");
+ else
+ strcat(fb_info.fix.id, "Catseye");
    fb_info.fbops = &hpfb_ops;
    fb_info.flags = FBINFO_DEFAULT;
    fb_info.var = hpfb_defined;
- fb_info.fix = hpfb_fix;
- fb_info.screen_base = (char *)hpfb_fix.smem_start; // FIXME
+ fb_info.screen_base = (char *)fb_start;

- fb_alloc_cmap(&fb_info.cmap, 256, 0);
+ fb_alloc_cmap(&fb_info.cmap, 1 << hpfb_defined.bits_per_pixel, 0);

    if (register_framebuffer(&fb_info) < 0)
+ {
+ fb_dealloc_cmap(&fb_info.cmap);
    return 1;
+ }
+
+ printk(KERN_INFO "fb%d: %s frame buffer device\n",
+ fb_info.node, fb_info.fix.id);

```

```

+     return 0;
+ }

@@ -178,36 +326,82 @@
int __init hpfb_init(void)
{
    unsigned int sid;
+ mm_segment_t fs;
+ unsigned char i;
+ int err;

    /* Topcats can be on the internal IO bus or real DIO devices.
- * The internal variant sits at 0xf0560000; it has primary
+ * The internal variant sits at 0x560000; it has primary
    * and secondary ID registers just like the DIO version.
    * So we merge the two detection routines.
    *
    * Perhaps this #define should be in a global header file:
    * I believe it's common to all internal fbs, not just topcat.
    */
-#define INTFBADDR 0xf0560000
+#define INTFBVADDR 0xf0560000
+#define INTFBPADDR 0x560000
+
+ if (!MACH_IS_HP300)
+ return -ENXIO;

+ if (fb_get_options("hpfb", NULL))
+ return -ENODEV;
+
- if (hwreg_present((void *)INTFBADDR) &&
- (DIO_ID(INTFBADDR) == DIO_ID_FBUFFER) &&
- topcat_sid_ok(sid = DIO_SECID(INTFBADDR))) {
- printk("Internal Topcat found (secondary id %02x)\n", sid);
- hpfb_init_one(INTFBADDR);
- } else {
- int sc = dio_find(DIO_ID_FBUFFER);
-
- if (sc) {
- unsigned long addr = (unsigned long)dio_scodetoviraddr(sc);
- unsigned int sid = DIO_SECID(addr);
-
- if (topcat_sid_ok(sid)) {
- printk("Topcat found at DIO select code %02x "
- "(secondary id %02x)\n", sc, sid);
- hpfb_init_one(addr);
- }
+ fs = get_fs();
+ set_fs(KERNEL_DS);
+ err = get_user(i, (unsigned char *)INTFBVADDR + DIO_IDOFF);

```

```

+ set_fs(fs);
+
+ if (!err && (i == DIO_ID_FBUFFER) && topcat_sid_ok(sid = DIO_SECID(INTFBVADDR)))
+ {
+ printk(KERN_INFO "Internal Topcat found (secondary id %02x)\n", sid);
+ if (hpfb_init_one(INTFBPADDR, INTFBVADDR))
+ {
+ return -ENOMEM;
+ }
+ }
+ else
+ {
+ int sc, size;
+ unsigned long paddr, vaddr;
+
+ if ((sc = dio_find(DIO_ENCODE_ID(DIO_ID_FBUFFER, DIO_ID2_LRCATSEYE))) < 0 &&
+ (sc = dio_find(DIO_ENCODE_ID(DIO_ID_FBUFFER, DIO_ID2_HRCCATSEYE))) < 0 &&
+ (sc = dio_find(DIO_ENCODE_ID(DIO_ID_FBUFFER, DIO_ID2_HRMCATSEYE))) < 0 &&
+ (sc = dio_find(DIO_ENCODE_ID(DIO_ID_FBUFFER, DIO_ID2_TOPCAT))) < 0)
+ {
+ return -ENXIO;
+ }
+
+ dio_config_board(sc);
+ paddr = dio_scodetophysaddr(sc);
+
+ if (sc >= DIOII_SCBASE)
+ {
+ /* To find out the real size of the device we first need to map it. */
+ vaddr = (unsigned long)ioremap(paddr, PAGE_SIZE);
+ size = DIO_SIZE(sc, vaddr);
+ iounmap((void *)vaddr);
+ vaddr = (unsigned long)ioremap(paddr, size);
+ }
+ else
+ {
+ vaddr = paddr + DIO_VIRADDRBASE;
+ size = DIO_SIZE(sc, vaddr);
+ }
+ sid = DIO_SECID(vaddr);
+
+ printk(KERN_INFO "Topcat found at DIO select code %d "
+ "(secondary id %02x)\n", sc, sid);
+ if (hpfb_init_one(paddr, vaddr))
+ {
+ if (sc >= DIOII_SCBASE)
+ iounmap((void *)vaddr);
+ dio_unconfig_board(sc);
+ return -ENOMEM;
+ }
+ }

```

Linux-Kernel: [PATCH 476] HP300 fb

```
+
+   return 0;
+ }
```

Gr{oetje,eeting}s,

Geert

```
--
Geert Uytterhoeven -- There's lots of Linux beyond ia32 -- geert@linux-m68k.org
In personal conversations with technical people, I call myself a hacker. But
when I'm talking to journalists I just say "programmer" or something like that.
-- Linus Torvalds
```

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