

[git patches] IDE updates part #5

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2008-02/msg02415.html>

- *From:* Bartłomiej Zolnierkiewicz <bzolnier@xxxxxxxxx>
 - *Date:* Wed, 6 Feb 2008 03:11:02 +0100
-

* new IDE host driver for Palmchip BK3710 chipset
(Anton Salnikov@MontaVista)

* second part of ide-tape redux patches
(Borislav Petkov)

* ide-generic bugfix

* minor fixups/cleanups

Linus, please pull from:

master.kernel.org:/pub/scm/linux/kernel/git/bart/ide-2.6.git/

to receive the following updates:

Documentation/kernel-parameters.txt | 3 +
drivers/ide/Kconfig | 9 +
drivers/ide/arm/Makefile | 1 +
drivers/ide/arm/icside.c | 2 +-
drivers/ide/arm/palm_bk3710.c | 395 ++++++
drivers/ide/cris/ide-cris.c | 33 +-
drivers/ide/ide-acpi.c | 2 +-
drivers/ide/ide-cd.c | 9 +-
drivers/ide/ide-dma.c | 3 +-
drivers/ide/ide-floppy.c | 6 +-
drivers/ide/ide-generic.c | 10 +-
drivers/ide/ide-io.c | 14 +-
drivers/ide/ide-iops.c | 45 +-
drivers/ide/ide-lib.c | 2 +-
drivers/ide/ide-probe.c | 47 +-
drivers/ide/ide-proc.c | 1 +
drivers/ide/ide-tape.c | 2400 ++++++-----
drivers/ide/ide-taskfile.c | 35 +-
drivers/ide/ide.c | 54 -
drivers/ide/legacy/buddha.c | 72 +-
drivers/ide/legacy/falconide.c | 42 +-

[git patches] IDE updates part #5

drivers/ide/legacy/gayle.c | 39 +-
drivers/ide/legacy/hd.c | 9 +-
drivers/ide/legacy/macide.c | 57 +-
drivers/ide/legacy/q40ide.c | 9 +-
drivers/ide/pci/Makefile | 3 +-
drivers/ide/pci/generic.c | 13 -
drivers/ide/pci/siimage.c | 3 -
drivers/macintosh/mediabay.c | 46 +-
drivers/scsi/ide-scsi.c | 4 +-
include/asm-powerpc/mediabay.h | 8 +-
include/linux/ide.h | 42 +-
32 files changed, 1825 insertions(+), 1593 deletions(-)
create mode 100644 drivers/ide/arm/palm_bk3710.c

Andrew Morton (2):

drivers/ide/ide-acpi.c: fix uninitialized var warning
drivers/ide/legacy/hd.c: fix uninitialized var warning

Anton Salnikov (1):

Palmchip BK3710 IDE driver

Bartłomiej Zolnierkiewicz (7):

ide-generic: probing bugfix
ppc: fix #ifdef-s in mediabay driver (take 2)
ide: remove write-only ->sata_misc[] from ide_hwif_t
ide: remove redundant BUG_ON() from [ata_pi_]reset_pollfunc()
ide: remove ide_setup_ports()
ide: add ide_read_[alt]status() inline helpers
ide: add ide_read_error() inline helper

Borislav Petkov (18):

ide-tape: refactor the debug logging facility
ide-tape: remove struct idetape_read_position_result_t
ide-tape: remove unreachable code chunk
ide-tape: simplify code branching in the interrupt handler
ide-tape: remove typedef idetape_chrdev_direction_t
ide-tape: struct idetape_tape_t: remove unused members
ide-tape: struct idetape_tape_t: shorten member names v2
ide-tape: remove idetape_increase_max_pipeline_stages()
ide-tape: shorten some function names
ide-tape: cleanup and fix comments
ide-tape: remove struct idetape_id_gcw
ide-tape: remove unused "length" arg from idetape_create_read_buffer_cmd()
ide-tape: include proper headers
ide-tape: collect module-related macro calls at the end
ide-tape: remove leftover OnStream support warning
ide-tape: fix syntax error in idetape_identify_device()
ide-tape: cleanup the remaining codestyle issues
ide-tape: bump minor driver version

[git patches] IDE updates part #5

Denis Cheng (1):

ide-pci-generic: kill the unused ifdef/endif/MODULE code

```
diff --git a/Documentation/kernel-parameters.txt b/Documentation/kernel-parameters.txt
index 9ad4e6f..8fd5aa4 100644
```

```
--- a/Documentation/kernel-parameters.txt
```

```
+++ b/Documentation/kernel-parameters.txt
```

```
@@ -780,6 +780,9 @@ and is between 256 and 4096 characters. It is defined in the file
loop use the MONITOR/MWAIT idle loop anyways. Performance should be the same
as idle=poll.
```

```
+ ide-pci-generic.all-generic-ide [HW] (E)IDE subsystem
```

```
+ Claim all unknown PCI IDE storage controllers.
```

```
+
```

```
ignore_loglevel [KNL]
```

```
Ignore loglevel setting - this will print /all/
```

```
kernel messages to the console. Useful for debugging.
```

```
diff --git a/drivers/ide/Kconfig b/drivers/ide/Kconfig
```

```
index 45b26ed..ab8fb25 100644
```

```
--- a/drivers/ide/Kconfig
```

```
+++ b/drivers/ide/Kconfig
```

```
@@ -1009,6 +1009,15 @@ config BLK_DEV_Q40IDE
```

```
normally be on; disable it only if you are running a custom hard
drive subsystem through an expansion card.
```

```
+config BLK_DEV_PALMCHIP_BK3710
```

```
+ tristate "Palmchip bk3710 IDE controller support"
```

```
+ depends on ARCH_DAVINCI
```

```
+ select BLK_DEV_IDEDMA_PCI
```

```
+ help
```

```
+ Say Y here if you want to support the onchip IDE controller on the
```

```
+ TI DaVinci SoC
```

```
+
```

```
+
```

```
config BLK_DEV_MPC8xx_IDE
```

```
tristate "MPC8xx IDE support"
```

```
depends on 8xx && (LWMON || IVMS8 || IVML24 || TQM8xxL) && IDE=y && BLK_DEV_IDE=y &&
```

```
!PPC_MERGE
```

```
diff --git a/drivers/ide/arm/Makefile b/drivers/ide/arm/Makefile
```

```
index 5f63ad2..936e7b0 100644
```

```
--- a/drivers/ide/arm/Makefile
```

```
+++ b/drivers/ide/arm/Makefile
```

```
@@ -2,6 +2,7 @@
```

```
obj-$(CONFIG_BLK_DEV_IDE_ICSIDE) += icside.o
```

```
obj-$(CONFIG_BLK_DEV_IDE_RAPIDE) += rapide.o
```

```
obj-$(CONFIG_BLK_DEV_IDE_BAST) += bast-ide.o
```

```
+obj-$(CONFIG_BLK_DEV_PALMCHIP_BK3710) += palm_bk3710.o
```

```
ifeq ($(CONFIG_IDE_ARM), m)
```

```
obj-m += ide_arm.o
```

[git patches] IDE updates part #5

[git patches] IDE updates part #5

```
diff --git a/drivers/ide/arm/icside.c b/drivers/ide/arm/icside.c
index fb00f38..e816b0f 100644
--- a/drivers/ide/arm/icside.c
+++ b/drivers/ide/arm/icside.c
@@ -365,7 +365,7 @@ static void icside_dma_timeout(ide_drive_t *drive)
if (icside_dma_test_irq(drive))
return;

- ide_dump_status(drive, "DMA timeout", HWIF(drive)->INB(IDE_STATUS_REG));
+ ide_dump_status(drive, "DMA timeout", ide_read_status(drive));

icside_dma_end(drive);
}
diff --git a/drivers/ide/arm/palm_bk3710.c b/drivers/ide/arm/palm_bk3710.c
new file mode 100644
index 0000000..c306997
--- /dev/null
+++ b/drivers/ide/arm/palm_bk3710.c
@@ -0,0 +1,395 @@
+/*
+ * Palmchip bk3710 IDE controller
+ *
+ * Copyright (C) 2006 Texas Instruments.
+ * Copyright (C) 2007 MontaVista Software, Inc., <source@xxxxxxxxxx>
+ *
+ *
+-----
+ *
+ * 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.
+ *
+-----
+ *
+ */
+
+#include <linux/types.h>
+#include <linux/module.h>
+#include <linux/kernel.h>
+#include <linux/ioport.h>
+#include <linux/hdreg.h>
```

```

+#include <linux/ide.h>
+#include <linux/delay.h>
+#include <linux/init.h>
+#include <linux/clock.h>
+#include <linux/platform_device.h>
+
+/* Offset of the primary interface registers */
+#define IDE_PALM_ATA_PRI_REG_OFFSET 0x1F0
+
+/* Primary Control Offset */
+#define IDE_PALM_ATA_PRI_CTL_OFFSET 0x3F6
+
+/*
+ * PalmChip 3710 IDE Controller UDMA timing structure Definition
+ */
+struct palm_bk3710_udmatiming {
+ unsigned int rptime; /* Ready to pause time */
+ unsigned int cycletime; /* Cycle Time */
+};
+
+#define BK3710_BMICP 0x00
+#define BK3710_BMISP 0x02
+#define BK3710_BMIDTP 0x04
+#define BK3710_BMICS 0x08
+#define BK3710_BMISS 0x0A
+#define BK3710_BMIDTS 0x0C
+#define BK3710_IDETIMP 0x40
+#define BK3710_IDETIMS 0x42
+#define BK3710_SIDETIM 0x44
+#define BK3710_SLEWCTL 0x45
+#define BK3710_IDESTATUS 0x47
+#define BK3710_UDMACTL 0x48
+#define BK3710_UDMATIM 0x4A
+#define BK3710_MISCCTL 0x50
+#define BK3710_REGSTB 0x54
+#define BK3710_REGRCVR 0x58
+#define BK3710_DATSTB 0x5C
+#define BK3710_DATRCVR 0x60
+#define BK3710_DMASTB 0x64
+#define BK3710_DMARCVR 0x68
+#define BK3710_UDMASTB 0x6C
+#define BK3710_UDMATRP 0x70
+#define BK3710_UDMAENV 0x74
+#define BK3710_IORDYTMP 0x78
+#define BK3710_IORDYTMS 0x7C
+
+#include "../ide-timing.h"
+
+static long ide_palm_clk;
+
+static const struct palm_bk3710_udmatiming palm_bk3710_udmatimings[6] = {

```

[git patches] IDE updates part #5

```
+ {160, 240}, /* UDMA Mode 0 */
+ {125, 160}, /* UDMA Mode 1 */
+ {100, 120}, /* UDMA Mode 2 */
+ {100, 90}, /* UDMA Mode 3 */
+ {85, 60}, /* UDMA Mode 4 */
+};
+
+static struct clk *ideclkp;
+
+static void palm_bk3710_setudmamode(void __iomem *base, unsigned int dev,
+ unsigned int mode)
+{
+ u8 tenv, trp, t0;
+ u32 val32;
+ u16 val16;
+
+ /* DMA Data Setup */
+ t0 = (palm_bk3710_udmatimings[mode].cycletime + ide_palm_clk - 1)
+ / ide_palm_clk - 1;
+ tenv = (20 + ide_palm_clk - 1) / ide_palm_clk - 1;
+ trp = (palm_bk3710_udmatimings[mode].rptime + ide_palm_clk - 1)
+ / ide_palm_clk - 1;
+
+ /* udmatim Register */
+ val16 = readw(base + BK3710_UDMATIM) & (dev ? 0xFF0F : 0xFFF0);
+ val16 |= (mode << (dev ? 4 : 0));
+ writew(val16, base + BK3710_UDMATIM);
+
+ /* udmastb Ultra DMA Access Strobe Width */
+ val32 = readl(base + BK3710_UDMASTB) & (0xFF << (dev ? 0 : 8));
+ val32 |= (t0 << (dev ? 8 : 0));
+ writel(val32, base + BK3710_UDMASTB);
+
+ /* udmatrp Ultra DMA Ready to Pause Time */
+ val32 = readl(base + BK3710_UDMATRP) & (0xFF << (dev ? 0 : 8));
+ val32 |= (trp << (dev ? 8 : 0));
+ writel(val32, base + BK3710_UDMATRP);
+
+ /* udmaenv Ultra DMA envelop Time */
+ val32 = readl(base + BK3710_UDMAENV) & (0xFF << (dev ? 0 : 8));
+ val32 |= (tenv << (dev ? 8 : 0));
+ writel(val32, base + BK3710_UDMAENV);
+
+ /* Enable UDMA for Device */
+ val16 = readw(base + BK3710_UDMACTL) | (1 << dev);
+ writew(val16, base + BK3710_UDMACTL);
+}
+
+static void palm_bk3710_setdmamode(void __iomem *base, unsigned int dev,
+ unsigned short min_cycle,
+ unsigned int mode)
```

```

+{
+ u8 td, tkw, t0;
+ u32 val32;
+ u16 val16;
+ struct ide_timing *t;
+ int cycletime;
+
+ t = ide_timing_find_mode(mode);
+ cycletime = max_t(int, t->cycle, min_cycle);
+
+ /* DMA Data Setup */
+ t0 = (cycletime + ide_palm_clk - 1) / ide_palm_clk;
+ td = (t->active + ide_palm_clk - 1) / ide_palm_clk;
+ tkw = t0 - td - 1;
+ td -= 1;
+
+ val32 = readl(base + BK3710_DMASTB) & (0xFF << (dev ? 0 : 8));
+ val32 |= (td << (dev ? 8 : 0));
+ writel(val32, base + BK3710_DMASTB);
+
+ val32 = readl(base + BK3710_DMARCVR) & (0xFF << (dev ? 0 : 8));
+ val32 |= (tkw << (dev ? 8 : 0));
+ writel(val32, base + BK3710_DMARCVR);
+
+ /* Disable UDMA for Device */
+ val16 = readw(base + BK3710_UDMACTL) & ~(1 << dev);
+ writew(val16, base + BK3710_UDMACTL);
+}
+
+static void palm_bk3710_setpiomode(void __iomem *base, ide_drive_t *mate,
+ unsigned int dev, unsigned int cycletime,
+ unsigned int mode)
+{
+ u8 t2, t2i, t0;
+ u32 val32;
+ struct ide_timing *t;
+
+ /* PIO Data Setup */
+ t0 = (cycletime + ide_palm_clk - 1) / ide_palm_clk;
+ t2 = (ide_timing_find_mode(XFER_PIO_0 + mode)->active +
+ ide_palm_clk - 1) / ide_palm_clk;
+
+ t2i = t0 - t2 - 1;
+ t2 -= 1;
+
+ val32 = readl(base + BK3710_DATSTB) & (0xFF << (dev ? 0 : 8));
+ val32 |= (t2 << (dev ? 8 : 0));
+ writel(val32, base + BK3710_DATSTB);
+
+ val32 = readl(base + BK3710_DATRCVR) & (0xFF << (dev ? 0 : 8));
+ val32 |= (t2i << (dev ? 8 : 0));

```

[git patches] IDE updates part #5

```
+ writel(val32, base + BK3710_DATRCVR);
+
+ if (mate && mate->present) {
+ u8 mode2 = ide_get_best_pio_mode(mate, 255, 4);
+
+ if (mode2 < mode)
+ mode = mode2;
+ }
+
+ /* TASKFILE Setup */
+ t = ide_timing_find_mode(XFER_PIO_0 + mode);
+ t0 = (t->cyc8b + ide_palm_clk - 1) / ide_palm_clk;
+ t2 = (t->act8b + ide_palm_clk - 1) / ide_palm_clk;
+
+ t2i = t0 - t2 - 1;
+ t2 -= 1;
+
+ val32 = readl(base + BK3710_REGSTB) & (0xFF << (dev ? 0 : 8));
+ val32 |= (t2 << (dev ? 8 : 0));
+ writel(val32, base + BK3710_REGSTB);
+
+ val32 = readl(base + BK3710_REGRCVR) & (0xFF << (dev ? 0 : 8));
+ val32 |= (t2i << (dev ? 8 : 0));
+ writel(val32, base + BK3710_REGRCVR);
+}
+
+static void palm_bk3710_set_dma_mode(ide_drive_t *drive, u8 xferspeed)
+{
+ int is_slave = drive->dn & 1;
+ void __iomem *base = (void *)drive->hwif->dma_base;
+
+ if (xferspeed >= XFER_UDMA_0) {
+ palm_bk3710_setudmamode(base, is_slave,
+ xferspeed - XFER_UDMA_0);
+ } else {
+ palm_bk3710_setdmamode(base, is_slave, drive->id->eide_dma_min,
+ xferspeed);
+ }
+}
+
+static void palm_bk3710_set_pio_mode(ide_drive_t *drive, u8 pio)
+{
+ unsigned int cycle_time;
+ int is_slave = drive->dn & 1;
+ ide_drive_t *mate;
+ void __iomem *base = (void *)drive->hwif->dma_base;
+
+ /*
+ * Obtain the drive PIO data for tuning the Palm Chip registers
+ */
+ cycle_time = ide_pio_cycle_time(drive, pio);
```

[git patches] IDE updates part #5

```
+ mate = ide_get_paired_drive(drive);
+ palm_bk3710_setpiomode(base, mate, is_slave, cycle_time, pio);
+}
+
+static void __devinit palm_bk3710_chipinit(void __iomem *base)
+{
+ /*
+ * enable the reset_en of ATA controller so that when ata signals
+ * are brought out, by writing into device config. at that
+ * time por_n signal should not be 'Z' and have a stable value.
+ */
+ writel(0x0300, base + BK3710_MISCCTL);
+
+ /* wait for some time and deassert the reset of ATA Device. */
+ mdelay(100);
+
+ /* Deassert the Reset */
+ writel(0x0200, base + BK3710_MISCCTL);
+
+ /*
+ * Program the IDETIMP Register Value based on the following assumptions
+ *
+ * (ATA_IDETIMP_IDEEN , ENABLE ) |
+ * (ATA_IDETIMP_SLVTIMEN , DISABLE) |
+ * (ATA_IDETIMP_RDYSMPL , 70NS) |
+ * (ATA_IDETIMP_RDYRCVRY , 50NS) |
+ * (ATA_IDETIMP_DMAFTIM1 , PIOCAMP) |
+ * (ATA_IDETIMP_PREPOST1 , DISABLE) |
+ * (ATA_IDETIMP_RDYSEN1 , DISABLE) |
+ * (ATA_IDETIMP_PIOFTIM1 , DISABLE) |
+ * (ATA_IDETIMP_DMAFTIM0 , PIOCAMP) |
+ * (ATA_IDETIMP_PREPOST0 , DISABLE) |
+ * (ATA_IDETIMP_RDYSEN0 , DISABLE) |
+ * (ATA_IDETIMP_PIOFTIM0 , DISABLE)
+ */
+ writew(0xB388, base + BK3710_IDETIMP);
+
+ /*
+ * Configure SIDETIM Register
+ * (ATA_SIDETIM_RDYSMPS1 ,120NS ) |
+ * (ATA_SIDETIM_RDYRCYS1 ,120NS )
+ */
+ writeb(0, base + BK3710_SIDETIM);
+
+ /*
+ * UDMACTL Ultra-ATA DMA Control
+ * (ATA_UDMACTL_UDMAP1 , 0 ) |
+ * (ATA_UDMACTL_UDMAP0 , 0 )
+ *
+ */
+ writew(0, base + BK3710_UDMACTL);
```

```

+
+ /*
+ * MISCCTL Miscellaneous Control Register
+ * (ATA_MISCCTL_RSTMODEP , 1) |
+ * (ATA_MISCCTL_RESETP , 0) |
+ * (ATA_MISCCTL_TIMORIDE , 1)
+ */
+ writel(0x201, base + BK3710_MISCCTL);
+
+ /*
+ * IORDYTMP IORDY Timer for Primary Register
+ * (ATA_IORDYTMP_IORDYTMP , 0xffff )
+ */
+ writel(0xFFFF, base + BK3710_IORDYTMP);
+
+ /*
+ * Configure BMISP Register
+ * (ATA_BMISP_DMAEN1 , DISABLE ) |
+ * (ATA_BMISP_DMAEN0 , DISABLE ) |
+ * (ATA_BMISP_IORDYINT , CLEAR) |
+ * (ATA_BMISP_INTRSTAT , CLEAR) |
+ * (ATA_BMISP_DMAERROR , CLEAR)
+ */
+ writew(0, base + BK3710_BMISP);
+
+ palm_bk3710_setpiomode(base, NULL, 0, 600, 0);
+ palm_bk3710_setpiomode(base, NULL, 1, 600, 0);
+ }
+static int __devinit palm_bk3710_probe(struct platform_device *pdev)
+{
+ hw_regs_t ide_ctrl_info;
+ int index = 0;
+ int pribase;
+ struct clk *clkp;
+ struct resource *mem, *irq;
+ ide_hwif_t *hwif;
+ void __iomem *base;
+
+ clkp = clk_get(NULL, "IDECLK");
+ if (IS_ERR(clkp))
+ return -ENODEV;
+
+ ideclkp = clkp;
+ clk_enable(ideclkp);
+ ide_palm_clk = clk_get_rate(ideclkp)/100000;
+ ide_palm_clk = (10000/ide_palm_clk) + 1;
+ /* Register the IDE interface with Linux ATA Interface */
+ memset(&ide_ctrl_info, 0, sizeof(ide_ctrl_info));
+
+ mem = platform_get_resource(pdev, IORESOURCE_MEM, 0);
+ if (mem == NULL) {

```

[git patches] IDE updates part #5

```
+ printk(KERN_ERR "failed to get memory region resource\n");
+ return -ENODEV;
+ }
+ irq = platform_get_resource(pdev, IORESOURCE_IRQ, 0);
+ if (irq == NULL) {
+ printk(KERN_ERR "failed to get IRQ resource\n");
+ return -ENODEV;
+ }
+
+ base = (void *)mem->start;
+
+ /* Configure the Palm Chip controller */
+ palm_bk3710_chipinit(base);
+
+ pribase = mem->start + IDE_PALM_ATA_PRI_REG_OFFSET;
+ for (index = 0; index < IDE_NR_PORTS - 2; index++)
+ ide_ctrl_info.io_ports[index] = pribase + index;
+ ide_ctrl_info.io_ports[IDE_CONTROL_OFFSET] = mem->start +
+ IDE_PALM_ATA_PRI_CTL_OFFSET;
+ ide_ctrl_info.irq = irq->start;
+ ide_ctrl_info.chipset = ide_palm3710;
+
+ if (ide_register_hw(&ide_ctrl_info, NULL, &hwif) < 0) {
+ printk(KERN_WARNING "Palm Chip BK3710 IDE Register Fail\n");
+ return -ENODEV;
+ }
+
+ hwif->set_pio_mode = &palm_bk3710_set_pio_mode;
+ hwif->set_dma_mode = &palm_bk3710_set_dma_mode;
+ hwif->mmio = 1;
+ default_hwif_mmiops(hwif);
+ hwif->cbl = ATA_CBL_PATA80;
+ hwif->ultra_mask = 0x1f; /* Ultra DMA Mode 4 Max
+ (input clk 99MHz) */
+ hwif->mwdma_mask = 0x7;
+ hwif->drives[0].autotune = 1;
+ hwif->drives[1].autotune = 1;
+
+ ide_setup_dma(hwif, mem->start);
+
+ return 0;
+}
+
+static struct platform_driver platform_bk_driver = {
+ .driver = {
+ .name = "palm_bk3710",
+ },
+ .probe = palm_bk3710_probe,
+ .remove = NULL,
+};
+
```

[git patches] IDE updates part #5

```

+static int __init palm_bk3710_init(void)
+{
+ return platform_driver_register(&platform_bk_driver);
+}
+
+module_init(palm_bk3710_init);
+MODULE_LICENSE("GPL");
+
diff --git a/drivers/ide/cris/ide-cris.c b/drivers/ide/cris/ide-cris.c
index 00587a8..e79bf8f 100644
--- a/drivers/ide/cris/ide-cris.c
+++ b/drivers/ide/cris/ide-cris.c
@@ -753,6 +753,25 @@ static void cris_set_dma_mode(ide_drive_t *drive, const u8 speed)
cris_ide_set_speed(TYPE_DMA, 0, strobe, hold);
}

+static void __init cris_setup_ports(hw_regs_t *hw, unsigned long base)
+{
+ int i;
+
+ memset(hw, 0, sizeof(*hw));
+
+ for (i = 0; i <= 7; i++)
+ hw->io_ports[i] = base + cris_ide_reg_addr(i, 0, 1);
+
+ /*
+ * the IDE control register is at ATA address 6,
+ * with CS1 active instead of CS0
+ */
+ hw->io_ports[IDE_CONTROL_OFFSET] = base + cris_ide_reg_addr(6, 1, 0);
+
+ hw->irq = ide_default_irq(0);
+ hw->ack_intr = cris_ide_ack_intr;
+}
+
static const struct ide_port_info cris_port_info __initdata = {
.chipset = ide_etrax100,
.host_flags = IDE_HFLAG_NO_ATAPI_DMA |
@@ -765,24 +784,16 @@ static const struct ide_port_info cris_port_info __initdata = {
static int __init init_e100_ide(void)
{
hw_regs_t hw;
- int ide_offsets[IDE_NR_PORTS], h, i;
+ int h;
u8 idx[4] = { 0xff, 0xff, 0xff, 0xff };

printk("ide: ETRAX FS built-in ATA DMA controller\n");

- for (i = IDE_DATA_OFFSET; i <= IDE_STATUS_OFFSET; i++)
- ide_offsets[i] = cris_ide_reg_addr(i, 0, 1);
-

```

[git patches] IDE updates part #5

```

- /* the IDE control register is at ATA address 6, with CS1 active instead of CS0 */
- ide_offsets[IDE_CONTROL_OFFSET] = cris_ide_reg_addr(6, 1, 0);
-
for (h = 0; h < 4; h++) {
ide_hwif_t *hwif = NULL;

- ide_setup_ports(&hw, cris_ide_base_address(h),
- ide_offsets,
- 0, 0, cris_ide_ack_intr,
- ide_default_irq(0));
+ cris_setup_ports(&hw, cris_ide_base_address(h));
+
hwif = ide_find_port(hw.io_ports[IDE_DATA_OFFSET]);
if (hwif == NULL)
continue;
diff --git a/drivers/ide/ide-acpi.c b/drivers/ide/ide-acpi.c
index 25aaeae..e07b189 100644
--- a/drivers/ide/ide-acpi.c
+++ b/drivers/ide/ide-acpi.c
@@ -171,7 +171,7 @@ err:
static acpi_handle ide_acpi_hwif_get_handle(ide_hwif_t *hwif)
{
struct device *dev = hwif->gendev.parent;
- acpi_handle dev_handle;
+ acpi_handle uninitialized_var(dev_handle);
acpi_integer pcidevfn;
acpi_handle chan_handle;
int err;
diff --git a/drivers/ide/ide-cd.c b/drivers/ide/ide-cd.c
index ee4d458..5e42c19 100644
--- a/drivers/ide/ide-cd.c
+++ b/drivers/ide/ide-cd.c
@@ -295,7 +295,8 @@ static int cdrom_decode_status(ide_drive_t *drive, int good_stat, int *stat_ret)
int stat, err, sense_key;

/* Check for errors. */
- stat = HWIF(drive)->INB(IDE_STATUS_REG);
+ stat = ide_read_status(drive);
+
if (stat_ret)
*stat_ret = stat;

@@ -303,7 +304,7 @@ static int cdrom_decode_status(ide_drive_t *drive, int good_stat, int *stat_ret)
return 0;

/* Get the IDE error register. */
- err = HWIF(drive)->INB(IDE_ERROR_REG);
+ err = ide_read_error(drive);
sense_key = err >> 4;

if (rq == NULL) {

```

[git patches] IDE updates part #5

```
@@ -692,7 +693,7 @@ int ide_cd_check_ireason(ide_drive_t *drive, int len, int ireason, int rw)
/* Some drives (ASUS) seem to tell us that status
* info is available. just get it and ignore.
*/
- (void) HWIF(drive)->INB(IDE_STATUS_REG);
+ (void)ide_read_status(drive);
return 0;
} else {
/* Drive wants a command packet, or invalid ireason... */
@@ -1326,7 +1327,7 @@ @ ide_do_rw_cdrom (ide_drive_t *drive, struct request *rq, sector_t block)
if (blk_fs_request(rq)) {
if (info->cd_flags & IDE_CD_FLAG_SEEKING) {
unsigned long elapsed = jiffies - info->start_seek;
- int stat = HWIF(drive)->INB(IDE_STATUS_REG);
+ int stat = ide_read_status(drive);

if ((stat & SEEK_STAT) != SEEK_STAT) {
if (elapsed < IDECD_SEEK_TIMEOUT) {
diff --git a/drivers/ide/ide-dma.c b/drivers/ide/ide-dma.c
index 3cf59f2..a4bb328 100644
--- a/drivers/ide/ide-dma.c
+++ b/drivers/ide/ide-dma.c
@@ -147,7 +147,8 @@ @ ide_startstop_t ide_dma_intr (ide_drive_t *drive)
u8 stat = 0, dma_stat = 0;

dma_stat = HWIF(drive)->ide_dma_end(drive);
- stat = HWIF(drive)->INB(IDE_STATUS_REG); /* get drive status */
+ stat = ide_read_status(drive);
+
if (OK_STAT(stat,DRIVE_READY,drive->bad_wstat|DRQ_STAT)) {
if (!dma_stat) {
struct request *rq = HWGROUP(drive)->rq;
diff --git a/drivers/ide/ide-floppy.c b/drivers/ide/ide-floppy.c
index f8fe6ee..faf22d7 100644
--- a/drivers/ide/ide-floppy.c
+++ b/drivers/ide/ide-floppy.c
@@ -465,7 +465,7 @@ @ static void idefloppy_retry_pc(ide_drive_t *drive)
idefloppy_pc_t *pc;
struct request *rq;

- (void)drive->hwif->INB(IDE_ERROR_REG);
+ (void)ide_read_error(drive);
pc = idefloppy_next_pc_storage(drive);
rq = idefloppy_next_rq_storage(drive);
idefloppy_create_request_sense_cmd(pc);
@@ -501,7 +501,7 @@ @ static ide_startstop_t idefloppy_pc_intr (ide_drive_t *drive)
}

/* Clear the interrupt */
- stat = drive->hwif->INB(IDE_STATUS_REG);
+ stat = ide_read_status(drive);
```

[git patches] IDE updates part #5

```
/* No more interrupts */
if ((stat & DRQ_STAT) == 0) {
@@ -1246,7 +1246,7 @@ static int idefloppy_get_format_progress(ide_drive_t *drive, int __user *arg)
u8 stat;

local_irq_save(flags);
- stat = drive->hwif->INB(IDE_STATUS_REG);
+ stat = ide_read_status(drive);
local_irq_restore(flags);

progress_indication = ((stat & SEEK_STAT) == 0) ? 0 : 0x10000;
diff --git a/drivers/ide/ide-generic.c b/drivers/ide/ide-generic.c
index be469db..709b9e4 100644
--- a/drivers/ide/ide-generic.c
+++ b/drivers/ide/ide-generic.c
@@ -20,8 +20,14 @@ static int __init ide_generic_init(void)
if (ide_hwifs[0].io_ports[IDE_DATA_OFFSET])
ide_get_lock(NULL, NULL); /* for atari only */

- for (i = 0; i < MAX_HWIFS; i++)
- idx[i] = ide_hwifs[i].present ? 0xff : i;
+ for (i = 0; i < MAX_HWIFS; i++) {
+ ide_hwif_t *hwif = &ide_hwifs[i];
+
+ if (hwif->io_ports[IDE_DATA_OFFSET] && !hwif->present)
+ idx[i] = i;
+ else
+ idx[i] = 0xff;
+ }

ide_device_add_all(idx, NULL);

diff --git a/drivers/ide/ide-io.c b/drivers/ide/ide-io.c
index 4bddef0..3addbe4 100644
--- a/drivers/ide/ide-io.c
+++ b/drivers/ide/ide-io.c
@@ -466,7 +466,7 @@ static ide_startstop_t ide_ata_error(ide_drive_t *drive, struct request *rq, u8
return ide_stopped;
}

- if (hwif->INB(IDE_STATUS_REG) & (BUSY_STAT|DRQ_STAT))
+ if (ide_read_status(drive) & (BUSY_STAT | DRQ_STAT))
rq->errors |= ERROR_RESET;

if ((rq->errors & ERROR_RESET) == ERROR_RESET) {
@@ -493,7 +493,7 @@ static ide_startstop_t ide_atapi_error(ide_drive_t *drive, struct request *rq, u
/* add decoding error stuff */
}

- if (hwif->INB(IDE_STATUS_REG) & (BUSY_STAT|DRQ_STAT))
```

[git patches] IDE updates part #5

```
+ if (ide_read_status(drive) & (BUSY_STAT | DRQ_STAT))
/* force an abort */
hwif->OUTB(WIN_IDLEIMMEDIATE, IDE_COMMAND_REG);

@@ -821,9 +821,8 @@ static ide_startstop_t execute_drive_cmd (ide_drive_t *drive,
#ifdef DEBUG
printk("%s: DRIVE_CMD (null)\n", drive->name);
#endif
- ide_end_drive_cmd(drive,
- hwif->INB(IDE_STATUS_REG),
- hwif->INB(IDE_ERROR_REG));
+ ide_end_drive_cmd(drive, ide_read_status(drive), ide_read_error(drive));
+
return ide_stopped;
}

@@ -1231,7 +1230,7 @@ static ide_startstop_t ide_dma_timeout_retry(ide_drive_t *drive, int error)
printk(KERN_WARNING "%s: DMA timeout error\n", drive->name);
(void)HWIF(drive)->ide_dma_end(drive);
ret = ide_error(drive, "dma timeout error",
- hwif->INB(IDE_STATUS_REG));
+ ide_read_status(drive));
} else {
printk(KERN_WARNING "%s: DMA timeout retry\n", drive->name);
hwif->dma_timeout(drive);
@@ -1355,7 +1354,8 @@ void ide_timer_expiry (unsigned long data)
startstop = ide_dma_timeout_retry(drive, wait);
} else
startstop =
- ide_error(drive, "irq timeout", hwif->INB(IDE_STATUS_REG));
+ ide_error(drive, "irq timeout",
+ ide_read_status(drive));
}
drive->service_time = jiffies - drive->service_start;
spin_lock_irq(&ide_lock);
diff --git a/drivers/ide/ide-iops.c b/drivers/ide/ide-iops.c
index a95178f..c32e759 100644
--- a/drivers/ide/ide-iops.c
+++ b/drivers/ide/ide-iops.c
@@ -430,10 +430,10 @@ int drive_is_ready (ide_drive_t *drive)
* about possible isa-pnp and pci-pnp issues yet.
*/
if (IDE_CONTROL_REG)
- stat = hwif->INB(IDE_ALTSTATUS_REG);
+ stat = ide_read_altstatus(drive);
else
/* Note: this may clear a pending IRQ!! */
- stat = hwif->INB(IDE_STATUS_REG);
+ stat = ide_read_status(drive);

if (stat & BUSY_STAT)
```

[git patches] IDE updates part #5

```
/* drive busy: definitely not interrupting */
@@ -458,23 +458,24 @@ EXPORT_SYMBOL(drive_is_ready);
*/
static int __ide_wait_stat(ide_drive_t *drive, u8 good, u8 bad, unsigned long timeout, u8 *rstat)
{
- ide_hwif_t *hwif = drive->hwif;
unsigned long flags;
int i;
u8 stat;

udelay(1); /* spec allows drive 400ns to assert "BUSY" */
- if ((stat = hwif->INB(IDE_STATUS_REG)) & BUSY_STAT) {
+ stat = ide_read_status(drive);
+
+ if (stat & BUSY_STAT) {
local_irq_set(flags);
timeout += jiffies;
- while ((stat = hwif->INB(IDE_STATUS_REG)) & BUSY_STAT) {
+ while ((stat = ide_read_status(drive)) & BUSY_STAT) {
if (time_after(jiffies, timeout)) {
/*
* One last read after the timeout in case
* heavy interrupt load made us not make any
* progress during the timeout..
*/
- stat = hwif->INB(IDE_STATUS_REG);
+ stat = ide_read_status(drive);
if (!(stat & BUSY_STAT))
break;

@@ -494,7 +495,9 @@ static int __ide_wait_stat(ide_drive_t *drive, u8 good, u8 bad, unsigned long ti
*/
for (i = 0; i < 10; i++) {
udelay(1);
- if (OK_STAT((stat = hwif->INB(IDE_STATUS_REG)), good, bad)) {
+ stat = ide_read_status(drive);
+
+ if (OK_STAT(stat, good, bad)) {
*rstat = stat;
return 0;
}
@@ -617,6 +620,7 @@ int ide_driveid_update(ide_drive_t *drive)
ide_hwif_t *hwif = drive->hwif;
struct hd_driveid *id;
unsigned long timeout, flags;
+ u8 stat;

/*
* Re-read drive->id for possible DMA mode
@@ -633,10 +637,15 @@ int ide_driveid_update(ide_drive_t *drive)
SELECT_MASK(drive, 0);
```

[git patches] IDE updates part #5

```

return 0; /* drive timed-out */
}
+
msleep(50); /* give drive a breather */
- } while (hwif->INB(IDE_ALTSTATUS_REG) & BUSY_STAT);
+ stat = ide_read_altstatus(drive);
+ } while (stat & BUSY_STAT);
+
msleep(50); /* wait for IRQ and DRQ_STAT */
- if (!OK_STAT(hwif->INB(IDE_STATUS_REG),DRQ_STAT,BAD_R_STAT)) {
+ stat = ide_read_status(drive);
+
+ if (!OK_STAT(stat, DRQ_STAT, BAD_R_STAT)) {
SELECT_MASK(drive, 0);
printk("%s: CHECK for good STATUS\n", drive->name);
return 0;
@@ -649,7 +658,7 @@ int ide_driveid_update(ide_drive_t *drive)
return 0;
}
ata_input_data(drive, id, SECTOR_WORDS);
- (void) hwif->INB(IDE_STATUS_REG); /* clear drive IRQ */
+ (void)ide_read_status(drive); /* clear drive IRQ */
local_irq_enable();
local_irq_restore(flags);
ide_fix_driveid(id);
@@ -850,17 +859,16 @@ static ide_startstop_t do_reset1 (ide_drive_t *, int);
static ide_startstop_t atapi_reset_pollfunc (ide_drive_t *drive)
{
ide_hwgroup_t *hwgroup = HWGROUP(drive);
- ide_hwif_t *hwif = HWIF(drive);
u8 stat;

SELECT_DRIVE(drive);
udelay (10);
+ stat = ide_read_status(drive);

- if (OK_STAT(stat = hwif->INB(IDE_STATUS_REG), 0, BUSY_STAT)) {
+ if (OK_STAT(stat, 0, BUSY_STAT))
printk("%s: ATAPI reset complete\n", drive->name);
- } else {
+ else {
if (time_before(jiffies, hwgroup->poll_timeout)) {
- BUG_ON(HWGROUP(drive)->handler != NULL);
ide_set_handler(drive, &atapi_reset_pollfunc, HZ/20, NULL);
/* continue polling */
return ide_started;
@@ -898,9 +906,10 @@ static ide_startstop_t reset_pollfunc (ide_drive_t *drive)
}
}

- if (!OK_STAT(tmp = hwif->INB(IDE_STATUS_REG), 0, BUSY_STAT)) {

```

[git patches] IDE updates part #5

```
+ tmp = ide_read_status(drive);
+
+ if (!OK_STAT(tmp, 0, BUSY_STAT)) {
if (time_before(jiffies, hwgroup->poll_timeout)) {
- BUG_ON(HWGROUPE(drive)->handler != NULL);
ide_set_handler(drive, &reset_pollfunc, HZ/20, NULL);
/* continue polling */
return ide_started;
@@ -909,7 +918,9 @@ static ide_startstop_t reset_pollfunc (ide_drive_t *drive)
drive->failures++;
} else {
printk("%s: reset: ", hwif->name);
- if ((tmp = hwif->INB(IDE_ERROR_REG)) == 1) {
+ tmp = ide_read_error(drive);
+
+ if (tmp == 1) {
printk("success\n");
drive->failures = 0;
} else {
diff --git a/drivers/ide/ide-lib.c b/drivers/ide/ide-lib.c
index b42940d..1ff676c 100644
--- a/drivers/ide/ide-lib.c
+++ b/drivers/ide/ide-lib.c
@@ -578,7 +578,7 @@ u8 ide_dump_status(ide_drive_t *drive, const char *msg, u8 stat)
}
printk("}\n");
if ((stat & (BUSY_STAT|ERR_STAT)) == ERR_STAT) {
- err = drive->hwif->INB(IDE_ERROR_REG);
+ err = ide_read_error(drive);
printk("%s: %s: error=0x%02x ", drive->name, msg, err);
if (drive->media == ide_disk)
ide_dump_ata_error(drive, err);
diff --git a/drivers/ide/ide-probe.c b/drivers/ide/ide-probe.c
index 9c07bdb..fd0ef82 100644
--- a/drivers/ide/ide-probe.c
+++ b/drivers/ide/ide-probe.c
@@ -264,8 +264,7 @@ err_misc:
static int actual_try_to_identify (ide_drive_t *drive, u8 cmd)
{
ide_hwif_t *hwif = HWIF(drive);
- int rc;
- unsigned long hd_status;
+ int use_altstatus = 0, rc;
unsigned long timeout;
u8 s = 0, a = 0;

@@ -273,19 +272,17 @@ static int actual_try_to_identify (ide_drive_t *drive, u8 cmd)
msleep(50);

if (IDE_CONTROL_REG) {
- a = hwif->INB(IDE_ALTSTATUS_REG);
```

[git patches] IDE updates part #5

```
- s = hwif->INB(IDE_STATUS_REG);
- if ((a ^ s) & ~INDEX_STAT) {
- printk(KERN_INFO "%s: probing with STATUS(0x%02x) instead of "
- "ALTSTATUS(0x%02x)\n", drive->name, s, a);
+ a = ide_read_altstatus(drive);
+ s = ide_read_status(drive);
+ if ((a ^ s) & ~INDEX_STAT)
/* ancient Seagate drives, broken interfaces */
- hd_status = IDE_STATUS_REG;
- } else {
+ printk(KERN_INFO "%s: probing with STATUS(0x%02x) "
+ "instead of ALTSTATUS(0x%02x)\n",
+ drive->name, s, a);
+ else
/* use non-intrusive polling */
- hd_status = IDE_ALTSTATUS_REG;
- }
- } else
- hd_status = IDE_STATUS_REG;
+ use_altstatus = 1;
+ }

/* set features register for atapi
* identify command to be sure of reply
@@ -306,11 +303,15 @@ static int actual_try_to_identify (ide_drive_t *drive, u8 cmd)
}
/* give drive a breather */
msleep(50);
- } while ((hwif->INB(hd_status)) & BUSY_STAT);
+ s = use_altstatus ? ide_read_altstatus(drive)
+ : ide_read_status(drive);
+ } while (s & BUSY_STAT);

/* wait for IRQ and DRQ_STAT */
msleep(50);
- if (OK_STAT((hwif->INB(IDE_STATUS_REG)), DRQ_STAT, BAD_R_STAT)) {
+ s = ide_read_status(drive);
+
+ if (OK_STAT(s, DRQ_STAT, BAD_R_STAT)) {
unsigned long flags;

/* local CPU only; some systems need this */
@@ -320,7 +321,7 @@ static int actual_try_to_identify (ide_drive_t *drive, u8 cmd)
/* drive responded with ID */
rc = 0;
/* clear drive IRQ */
- (void) hwif->INB(IDE_STATUS_REG);
+ (void)ide_read_status(drive);
local_irq_restore(flags);
} else {
/* drive refused ID */
```

[git patches] IDE updates part #5

```
@@ -367,7 +368,7 @@ static int try_to_identify (ide_drive_t *drive, u8 cmd)

ide_set_irq(drive, 0);
/* clear drive IRQ */
- (void) hwif->INB(IDE_STATUS_REG);
+ (void)ide_read_status(drive);
udelay(5);
irq = probe_irq_off(cookie);
if (!hwif->irq) {
@@ -455,7 +456,9 @@ static int do_probe (ide_drive_t *drive, u8 cmd)
return 3;
}

- if (OK_STAT((hwif->INB(IDE_STATUS_REG)), READY_STAT, BUSY_STAT) ||
+ stat = ide_read_status(drive);
+
+ if (OK_STAT(stat, READY_STAT, BUSY_STAT) ||
drive->present || cmd == WIN_PIDENTIFY) {
/* send cmd and wait */
if ((rc = try_to_identify(drive, cmd))) {
@@ -463,7 +466,7 @@ static int do_probe (ide_drive_t *drive, u8 cmd)
rc = try_to_identify(drive,cmd);
}

- stat = hwif->INB(IDE_STATUS_REG);
+ stat = ide_read_status(drive);

if (stat == (BUSY_STAT | READY_STAT))
return 4;
@@ -482,7 +485,7 @@ static int do_probe (ide_drive_t *drive, u8 cmd)
}

/* ensure drive IRQ is clear */
- stat = hwif->INB(IDE_STATUS_REG);
+ stat = ide_read_status(drive);

if (rc == 1)
printk(KERN_ERR "%s: no response (status = 0x%02x)\n",
@@ -496,7 +499,7 @@ static int do_probe (ide_drive_t *drive, u8 cmd)
SELECT_DRIVE(&hwif->drives[0]);
msleep(50);
/* ensure drive irq is clear */
- (void) hwif->INB(IDE_STATUS_REG);
+ (void)ide_read_status(drive);
}
return rc;
}
@@ -521,7 +524,7 @@ static void enable_nest (ide_drive_t *drive)

msleep(50);
```

[git patches] IDE updates part #5

```
- stat = hwif->INB(IDE_STATUS_REG);
+ stat = ide_read_status(drive);

if (!OK_STAT(stat, 0, BAD_STAT))
printk(KERN_CONT "failed (status = 0x%02x)\n", stat);
diff --git a/drivers/ide/ide-proc.c b/drivers/ide/ide-proc.c
index 975c0ff..bab88ca 100644
--- a/drivers/ide/ide-proc.c
+++ b/drivers/ide/ide-proc.c
@@ -65,6 +65,7 @@ static int proc_ide_read_imodel
case ide_4drives: name = "4drives"; break;
case ide_pmac: name = "mac-io"; break;
case ide_aulxxx: name = "aulxxx"; break;
+ case ide_palm3710: name = "palm3710"; break;
case ide_etrax100: name = "etrax100"; break;
case ide_acorn: name = "acorn"; break;
default: name = "(unknown)"; break;
diff --git a/drivers/ide/ide-tape.c b/drivers/ide/ide-tape.c
index bf40d8c..49dd2e7 100644
--- a/drivers/ide/ide-tape.c
+++ b/drivers/ide/ide-tape.c
@@ -15,7 +15,7 @@
* Documentation/ide/ChangeLog.ide-tape.1995-2002
*/

-#define IDETAPE_VERSION "1.19"
+#define IDETAPE_VERSION "1.20"

#include <linux/module.h>
#include <linux/types.h>
@@ -39,63 +39,70 @@
#include <scsi/scsi.h>

#include <asm/byteorder.h>
-#include <asm/irq.h>
-#include <asm/uaccess.h>
-#include <asm/io.h>
+#include <linux/irq.h>
+#include <linux/uaccess.h>
+#include <linux/io.h>
#include <asm/unaligned.h>
#include <linux/mtio.h>

+enum {
+ /* output errors only */
+ DBG_ERR = (1 << 0),
+ /* output all sense key/asc */
+ DBG_SENSE = (1 << 1),
+ /* info regarding all chrdev-related procedures */
+ DBG_CHRDEV = (1 << 2),
+ /* all remaining procedures */
```

[git patches] IDE updates part #5

```
+ DBG_PROCS = (1 << 3),
+ /* buffer alloc info (pc_stack & rq_stack) */
+ DBG_PCRQ_STACK = (1 << 4),
+};
+
+/* define to see debug info */
+#define IDETAPE_DEBUG_LOG 0
+
+#if IDETAPE_DEBUG_LOG
+#define debug_log(lvl, fmt, args...) \
+{ \
+ if (tape->debug_mask & lvl) \
+ printk(KERN_INFO "ide-tape: " fmt, ## args); \
+}
+#else
+#define debug_log(lvl, fmt, args...) do { } while (0)
+#endif
+
+/****** Tunable parameters *****/

/*
- * Pipelined mode parameters.
+ * Pipelined mode parameters.
*
- * We try to use the minimum number of stages which is enough to
- * keep the tape constantly streaming. To accomplish that, we implement
- * a feedback loop around the maximum number of stages:
+ * We try to use the minimum number of stages which is enough to keep the tape
+ * constantly streaming. To accomplish that, we implement a feedback loop around
+ * the maximum number of stages:
*
- * We start from MIN maximum stages (we will not even use MIN stages
- * if we don't need them), increment it by RATE*(MAX-MIN)
- * whenever we sense that the pipeline is empty, until we reach
- * the optimum value or until we reach MAX.
+ * We start from MIN maximum stages (we will not even use MIN stages if we don't
+ * need them), increment it by RATE*(MAX-MIN) whenever we sense that the
+ * pipeline is empty, until we reach the optimum value or until we reach MAX.
*
- * Setting the following parameter to 0 is illegal: the pipelined mode
- * cannot be disabled (calculate_speeds() divides by tape->max_stages.)
+ * Setting the following parameter to 0 is illegal: the pipelined mode cannot be
+ * disabled (idetape_calculate_speeds() divides by tape->max_stages.)
*/
#define IDETAPE_MIN_PIPELINE_STAGES 1
#define IDETAPE_MAX_PIPELINE_STAGES 400
#define IDETAPE_INCREASE_STAGES_RATE 20

/*
- * The following are used to debug the driver:
```

[git patches] IDE updates part #5

```

- *
- * Setting IDETAPE_DEBUG_LOG to 1 will log driver flow control.
+ * After each failed packet command we issue a request sense command and retry
+ * the packet command IDETAPE_MAX_PC_RETRIES times.
*
- * Setting them to 0 will restore normal operation mode:
- *
- * 1. Disable logging normal successful operations.
- * 2. Disable self-sanity checks.
- * 3. Errors will still be logged, of course.
- *
- * All the #if DEBUG code will be removed some day, when the driver
- * is verified to be stable enough. This will make it much more
- * esthetic.
- */
-#define IDETAPE_DEBUG_LOG 0
-
-/*
- * After each failed packet command we issue a request sense command
- * and retry the packet command IDETAPE_MAX_PC_RETRIES times.
- *
- * Setting IDETAPE_MAX_PC_RETRIES to 0 will disable retries.
+ * Setting IDETAPE_MAX_PC_RETRIES to 0 will disable retries.
*/
#define IDETAPE_MAX_PC_RETRIES 3

/*
- * With each packet command, we allocate a buffer of
- * IDETAPE_PC_BUFFER_SIZE bytes. This is used for several packet
- * commands (Not for READ/WRITE commands).
+ * With each packet command, we allocate a buffer of IDETAPE_PC_BUFFER_SIZE
+ * bytes. This is used for several packet commands (Not for READ/WRITE commands)
*/
#define IDETAPE_PC_BUFFER_SIZE 256

@@ -114,48 +121,39 @@
#define IDETAPE_WAIT_CMD (900*HZ)

/*
- * The following parameter is used to select the point in the internal
- * tape fifo in which we will start to refill the buffer. Decreasing
- * the following parameter will improve the system's latency and
- * interactive response, while using a high value might improve system
- * throughput.
+ * The following parameter is used to select the point in the internal tape fifo
+ * in which we will start to refill the buffer. Decreasing the following
+ * parameter will improve the system's latency and interactive response, while
+ * using a high value might improve system throughput.
*/
-#define IDETAPE_FIFO_THRESHOLD 2
+#define IDETAPE_FIFO_THRESHOLD 2

```

```
/*
- * DSC polling parameters.
- *
- * Polling for DSC (a single bit in the status register) is a very
- * important function in ide-tape. There are two cases in which we
- * poll for DSC:
+ * DSC polling parameters.
*
- * 1. Before a read/write packet command, to ensure that we
- * can transfer data from/to the tape's data buffers, without
- * causing an actual media access. In case the tape is not
- * ready yet, we take out our request from the device
- * request queue, so that ide.c will service requests from
- * the other device on the same interface meanwhile.
+ * Polling for DSC (a single bit in the status register) is a very important
+ * function in ide-tape. There are two cases in which we poll for DSC:
*
- * 2. After the successful initialization of a "media access
- * packet command", which is a command which can take a long
- * time to complete (it can be several seconds or even an hour).
+ * 1. Before a read/write packet command, to ensure that we can transfer data
+ * from/to the tape's data buffers, without causing an actual media access.
+ * In case the tape is not ready yet, we take out our request from the device
+ * request queue, so that ide.c could service requests from the other device
+ * on the same interface in the meantime.
*
- * Again, we postpone our request in the middle to free the bus
- * for the other device. The polling frequency here should be
- * lower than the read/write frequency since those media access
- * commands are slow. We start from a "fast" frequency -
- * IDETAPE_DSC_MA_FAST (one second), and if we don't receive DSC
- * after IDETAPE_DSC_MA_THRESHOLD (5 minutes), we switch it to a
- * lower frequency - IDETAPE_DSC_MA_SLOW (1 minute).
+ * 2. After the successful initialization of a "media access packet command",
+ * which is a command that can take a long time to complete (the interval can
+ * range from several seconds to even an hour). Again, we postpone our request
+ * in the middle to free the bus for the other device. The polling frequency
+ * here should be lower than the read/write frequency since those media access
+ * commands are slow. We start from a "fast" frequency - IDETAPE_DSC_MA_FAST
+ * (1 second), and if we don't receive DSC after IDETAPE_DSC_MA_THRESHOLD
+ * (5 min), we switch it to a lower frequency - IDETAPE_DSC_MA_SLOW (1 min).
*
- * We also set a timeout for the timer, in case something goes wrong.
- * The timeout should be longer then the maximum execution time of a
- * tape operation.
- */
-
-/*
- * DSC timings.
+ * We also set a timeout for the timer, in case something goes wrong. The
```

[git patches] IDE updates part #5

```
+ * timeout should be longer then the maximum execution time of a tape operation.
*/
+
+/* DSC timings. */
#define IDETAPE_DSC_RW_MIN 5*HZ/100 /* 50 msec */
#define IDETAPE_DSC_RW_MAX 40*HZ/100 /* 400 msec */
#define IDETAPE_DSC_RW_TIMEOUT 2*60*HZ /* 2 minutes */
@@ -166,19 +164,15 @@

/***** End of tunable parameters *****/

_/*
- * Read/Write error simulation
- */
+/* Read/Write error simulation */
#define SIMULATE_ERRORS 0

_/*
- * For general magnetic tape device compatibility.
- */
-#ifndef enum {
- idetape_direction_none,
- idetape_direction_read,
- idetape_direction_write
-} idetape_chrdev_direction_t;
+/* tape directions */
+enum {
+ IDETAPE_DIR_NONE = (1 << 0),
+ IDETAPE_DIR_READ = (1 << 1),
+ IDETAPE_DIR_WRITE = (1 << 2),
+};

struct idetape_bh {
u32 b_size;
@@ -187,24 +181,32 @@ struct idetape_bh {
char *b_data;
};

_/*
- * Our view of a packet command.
- */
typedef struct idetape_packet_command_s {
- u8 c[12]; /* Actual packet bytes */
- int retries; /* On each retry, we increment retries */
- int error; /* Error code */
- int request_transfer; /* Bytes to transfer */
- int actually_transferred; /* Bytes actually transferred */
- int buffer_size; /* Size of our data buffer */
+ /* Actual packet bytes */
+ u8 c[12];
+ /* On each retry, we increment retries */
```

```

+ int retries;
+ /* Error code */
+ int error;
+ /* Bytes to transfer */
+ int request_transfer;
+ /* Bytes actually transferred */
+ int actually_transferred;
+ /* Size of our data buffer */
+ int buffer_size;
struct idetape_bh *bh;
char *b_data;
int b_count;
- u8 *buffer; /* Data buffer */
- u8 *current_position; /* Pointer into the above buffer */
- ide_startstop_t (*callback) (ide_drive_t *); /* Called when this packet command is completed */
- u8 pc_buffer[IDETAPE_PC_BUFFER_SIZE]; /* Temporary buffer */
- unsigned long flags; /* Status/Action bit flags: long for set_bit */
+ /* Data buffer */
+ u8 *buffer;
+ /* Pointer into the above buffer */
+ u8 *current_position;
+ /* Called when this packet command is completed */
+ ide_startstop_t (*callback) (ide_drive_t *);
+ /* Temporary buffer */
+ u8 pc_buffer[IDETAPE_PC_BUFFER_SIZE];
+ /* Status/Action bit flags: long for set_bit */
+ unsigned long flags;
} idetape_pc_t;

/*
@@ -223,9 +225,7 @@ typedef struct idetape_packet_command_s {
/* Data direction */
#define PC_WRITING 5

-/*
- * A pipeline stage.
- */
+/* A pipeline stage. */
typedef struct idetape_stage_s {
struct request rq; /* The corresponding request */
struct idetape_bh *bh; /* The data buffers */
@@ -233,9 +233,8 @@ typedef struct idetape_stage_s {
} idetape_stage_t;

/*
- * Most of our global data which we need to save even as we leave the
- * driver due to an interrupt or a timer event is stored in a variable
- * of type idetape_tape_t, defined below.
+ * Most of our global data which we need to save even as we leave the driver due
+ * to an interrupt or a timer event is stored in the struct defined below.
*/

```

```

typedef struct ide_tape_obj {
ide_drive_t *drive;
@@ -271,15 +270,14 @@ typedef struct ide_tape_obj {
int rq_stack_index;

/*
- * DSC polling variables.
+ * DSC polling variables.
*
- * While polling for DSC we use postponed_rq to postpone the
- * current request so that ide.c will be able to service
- * pending requests on the other device. Note that at most
- * we will have only one DSC (usually data transfer) request
- * in the device request queue. Additional requests can be
- * queued in our internal pipeline, but they will be visible
- * to ide.c only one at a time.
+ * While polling for DSC we use postponed_rq to postpone the current
+ * request so that ide.c will be able to service pending requests on the
+ * other device. Note that at most we will have only one DSC (usually
+ * data transfer) request in the device request queue. Additional
+ * requests can be queued in our internal pipeline, but they will be
+ * visible to ide.c only one at a time.
*/
struct request *postponed_rq;
/* The time in which we started polling for DSC */
@@ -287,73 +285,57 @@ typedef struct ide_tape_obj {
/* Timer used to poll for dsc */
struct timer_list dsc_timer;
/* Read/Write dsc polling frequency */
- unsigned long best_dsc_rw_frequency;
- /* The current polling frequency */
- unsigned long dsc_polling_frequency;
- /* Maximum waiting time */
+ unsigned long best_dsc_rw_freq;
+ unsigned long dsc_poll_freq;
unsigned long dsc_timeout;

- /*
- * Read position information
- */
+ /* Read position information */
u8 partition;
/* Current block */
- unsigned int first_frame_position;
- unsigned int last_frame_position;
- unsigned int blocks_in_buffer;
+ unsigned int first_frame;

- /*
- * Last error information
- */

```

```
+ /* Last error information */
u8 sense_key, asc, ascq;

- /*
- * Character device operation
- */
+ /* Character device operation */
unsigned int minor;
/* device name */
char name[4];
/* Current character device data transfer direction */
- idetape_chrdev_direction_t chrdev_direction;
+ u8 chrdev_dir;

- /*
- * Device information
- */
- /* Usually 512 or 1024 bytes */
- unsigned short tape_block_size;
+ /* tape block size, usually 512 or 1024 bytes */
+ unsigned short blk_size;
int user_bs_factor;

/* Copy of the tape's Capabilities and Mechanical Page */
u8 caps[20];

/*
- * Active data transfer request parameters.
- *
- * At most, there is only one ide-tape originated data transfer
- * request in the device request queue. This allows ide.c to
- * easily service requests from the other device when we
- * postpone our active request. In the pipelined operation
- * mode, we use our internal pipeline structure to hold
- * more data requests.
+ * Active data transfer request parameters.
*
- * The data buffer size is chosen based on the tape's
- * recommendation.
+ * At most, there is only one ide-tape originated data transfer request
+ * in the device request queue. This allows ide.c to easily service
+ * requests from the other device when we postpone our active request.
+ * In the pipelined operation mode, we use our internal pipeline
+ * structure to hold more data requests. The data buffer size is chosen
+ * based on the tape's recommendation.
*/
- /* Pointer to the request which is waiting in the device request queue */
- struct request *active_data_request;
- /* Data buffer size (chosen based on the tape's recommendation */
+ /* ptr to the request which is waiting in the device request queue */
+ struct request *active_data_rq;
```

```

+ /* Data buffer size chosen based on the tape's recommendation */
int stage_size;
idetape_stage_t *merge_stage;
int merge_stage_size;
struct idetape_bh *bh;
char *b_data;
int b_count;
-
+
/*
- * Pipeline parameters.
+ * Pipeline parameters.
*
- * To accomplish non-pipelined mode, we simply set the following
- * variables to zero (or NULL, where appropriate).
+ * To accomplish non-pipelined mode, we simply set the following
+ * variables to zero (or NULL, where appropriate).
*/
/* Number of currently used stages */
int nr_stages;
@@ -378,20 +360,13 @@ typedef struct ide_tape_obj {
/* Status/Action flags: long for set_bit */
unsigned long flags;
/* protects the ide-tape queue */
- spinlock_t spinlock;
+ spinlock_t lock;

- /*
- * Measures average tape speed
- */
+ /* Measures average tape speed */
unsigned long avg_time;
int avg_size;
int avg_speed;

- char vendor_id[10];
- char product_id[18];
- char firmware_revision[6];
- int firmware_revision_num;
-
/* the door is currently locked */
int door_locked;
/* the tape hardware is write protected */
@@ -400,11 +375,9 @@ typedef struct ide_tape_obj {
char write_prot;

/*
- * Limit the number of times a request can
- * be postponed, to avoid an infinite postpone
- * deadlock.
+ * Limit the number of times a request can be postponed, to avoid an

```

```

+ * infinite postpone deadlock.
*/
- /* request postpone count limit */
int postpone_cnt;

/*
@@ -419,30 +392,19 @@ typedef struct ide_tape_obj {
int tape_head;
int last_tape_head;

- /*
- * Speed control at the tape buffers input/output
- */
+ /* Speed control at the tape buffers input/output */
unsigned long insert_time;
int insert_size;
int insert_speed;
int max_insert_speed;
int measure_insert_time;

- /*
- * Measure tape still time, in milliseconds
- */
- unsigned long tape_still_time_begin;
- int tape_still_time;
-
- /*
- * Speed regulation negative feedback loop
- */
+ /* Speed regulation negative feedback loop */
int speed_control;
int pipeline_head_speed;
int controlled_pipeline_head_speed;
int uncontrolled_pipeline_head_speed;
int controlled_last_pipeline_head;
- int uncontrolled_last_pipeline_head;
unsigned long uncontrolled_pipeline_head_time;
unsigned long controlled_pipeline_head_time;
int controlled_previous_pipeline_head;
@@ -451,18 +413,7 @@ typedef struct ide_tape_obj {
unsigned long uncontrolled_previous_head_time;
int restart_speed_control_req;

- /*
- * Debug_level determines amount of debugging output;
- * can be changed using /proc/ide/hdx/settings
- * 0 : almost no debugging output
- * 1 : 0+output errors only
- * 2 : 1+output all sensekey/asc
- * 3 : 2+follow all chrdev related procedures
- * 4 : 3+follow all procedures

```

[git patches] IDE updates part #5

```
- * 5 : 4+include pc_stack rq_stack info
- * 6 : 5+USE_COUNT updates
- */
- int debug_level;
+ u32 debug_mask;
} idetape_tape_t;

static DEFINE_MUTEX(idetape_ref_mutex);
@@ -495,9 +446,7 @@ static void ide_tape_put(struct ide_tape_obj *tape)
mutex_unlock(&idetape_ref_mutex);
}

-/*
- * Tape door status
- */
+/* Tape door status */
#define DOOR_UNLOCKED 0
#define DOOR_LOCKED 1
#define DOOR_EXPLICITLY_LOCKED 2
@@ -517,30 +466,23 @@ static void ide_tape_put(struct ide_tape_obj *tape)
/* 0 = no tape is loaded, so we don't rewind after ejecting */
#define IDETAPE_MEDIUM_PRESENT 9

-/*
- * Some defines for the READ BUFFER command
- */
+/* A define for the READ BUFFER command */
#define IDETAPE_RETRIEVE_FAULTY_BLOCK 6

-/*
- * Some defines for the SPACE command
- */
+/* Some defines for the SPACE command */
#define IDETAPE_SPACE_OVER_FILEMARK 1
#define IDETAPE_SPACE_TO_EOD 3

-/*
- * Some defines for the LOAD UNLOAD command
- */
+/* Some defines for the LOAD UNLOAD command */
#define IDETAPE_LU_LOAD_MASK 1
#define IDETAPE_LU_RETENSION_MASK 2
#define IDETAPE_LU_EOT_MASK 4

/*
- * Special requests for our block device strategy routine.
+ * Special requests for our block device strategy routine.
*
- * In order to service a character device command, we add special
- * requests to the tail of our block device request queue and wait
- * for their completion.
```

[git patches] IDE updates part #5

```
+ * In order to service a character device command, we add special requests to
+ * the tail of our block device request queue and wait for their completion.
*/

enum {
@@ -551,55 +493,20 @@ enum {
REQ_IDETAPE_READ_BUFFER = (1 << 4),
};

-/*
- * Error codes which are returned in rq->errors to the higher part
- * of the driver.
- */
+/* Error codes returned in rq->errors to the higher part of the driver. */
#define IDETAPE_ERROR_GENERAL 101
#define IDETAPE_ERROR_FILEMARK 102
#define IDETAPE_ERROR_EOD 103

-/*
- * The following is used to format the general configuration word of
- * the ATAPI IDENTIFY DEVICE command.
- */
-struct idetape_id_gcw {
- unsigned packet_size :2; /* Packet Size */
- unsigned reserved234 :3; /* Reserved */
- unsigned drq_type :2; /* Command packet DRQ type */
- unsigned removable :1; /* Removable media */
- unsigned device_type :5; /* Device type */
- unsigned reserved13 :1; /* Reserved */
- unsigned protocol :2; /* Protocol type */
-};
-
-/*
- * READ POSITION packet command – Data Format (From Table 6–57)
- */
-typedef struct {
- unsigned reserved0_10 :2; /* Reserved */
- unsigned bpu :1; /* Block Position Unknown */
- unsigned reserved0_543 :3; /* Reserved */
- unsigned eop :1; /* End Of Partition */
- unsigned bop :1; /* Beginning Of Partition */
- u8 partition; /* Partition Number */
- u8 reserved2, reserved3; /* Reserved */
- u32 first_block; /* First Block Location */
- u32 last_block; /* Last Block Location (Optional) */
- u8 reserved12; /* Reserved */
- u8 blocks_in_buffer[3]; /* Blocks In Buffer – (Optional) */
- u32 bytes_in_buffer; /* Bytes In Buffer (Optional) */
-} idetape_read_position_result_t;
-
-/* Structures related to the SELECT SENSE / MODE SENSE packet commands. */
```

[git patches] IDE updates part #5

```
#define IDETAPE_BLOCK_DESCRIPTOR 0
#define IDETAPE_CAPABILITIES_PAGE 0x2a

/*
- * The variables below are used for the character device interface.
- * Additional state variables are defined in our ide_drive_t structure.
+ * The variables below are used for the character device interface. Additional
+ * state variables are defined in our ide_drive_t structure.
*/
-static struct ide_tape_obj * idetape_devs[MAX_HWIFS * MAX_DRIVES];
+static struct ide_tape_obj * idetape_devs[MAX_HWIFS * MAX_DRIVES];

#define ide_tape_f(file) ((file)->private_data)

@@ -616,23 +523,17 @@ static struct ide_tape_obj * ide_tape_chrdev_get(unsigned int i)
}

/*
- * Function declarations
- *
- */
-static int idetape_chrdev_release (struct inode *inode, struct file *filp);
-static void idetape_write_release (ide_drive_t *drive, unsigned int minor);
-
-/*
* Too bad. The drive wants to send us data which we are not ready to accept.
* Just throw it away.
*/
-static void idetape_discard_data (ide_drive_t *drive, unsigned int bcount)
+static void idetape_discard_data(ide_drive_t *drive, unsigned int bcount)
{
while (bcount-->0)
(void) HWIF(drive)->INB(IDE_DATA_REG);
}

-static void idetape_input_buffers (ide_drive_t *drive, idetape_pc_t *pc, unsigned int bcount)
+static void idetape_input_buffers(ide_drive_t *drive, idetape_pc_t *pc,
+ unsigned int bcount)
{
struct idetape_bh *bh = pc->bh;
int count;
@@ -644,8 +545,11 @@ static void idetape_input_buffers (ide_drive_t *drive, idetape_pc_t *pc, unsigned
idetape_discard_data(drive, bcount);
-
-/*
- * idetape_kfree_stage calls kfree to completely free a stage, along with
- * its related buffers.
- */
-static void __idetape_kfree_stage (idetape_stage_t *stage)
+/* Free a stage along with its related buffers completely. */
+static void __idetape_kfree_stage(idetape_stage_t *stage)
```

[git patches] IDE updates part #5

```
{
struct idetape_bh *prev_bh, *bh = stage->bh;
int size;
@@ -907,30 +770,29 @@ static void __idetape_kfree_stage (idetape_stage_t *stage)
kfree(stage);
}

-static void idetape_kfree_stage (idetape_tape_t *tape, idetape_stage_t *stage)
+static void idetape_kfree_stage(idetape_tape_t *tape, idetape_stage_t *stage)
{
__idetape_kfree_stage(stage);
}

/*
- * idetape_remove_stage_head removes tape->first_stage from the pipeline.
- * The caller should avoid race conditions.
+ * Remove tape->first_stage from the pipeline. The caller should avoid race
+ * conditions.
*/
-static void idetape_remove_stage_head (ide_drive_t *drive)
+static void idetape_remove_stage_head(ide_drive_t *drive)
{
idetape_tape_t *tape = drive->driver_data;
idetape_stage_t *stage;
-
-#if IDETAPE_DEBUG_LOG
- if (tape->debug_level >= 4)
- printk(KERN_INFO "ide-tape: Reached idetape_remove_stage_head\n");
-#endif /* IDETAPE_DEBUG_LOG */
+
+ debug_log(DBG_PROCS, "Enter %s\n", __func__);
+
+ if (tape->first_stage == NULL) {
+ printk(KERN_ERR "ide-tape: bug: tape->first_stage is NULL\n");
+ return;
+ }
+ if (tape->active_stage == tape->first_stage) {
+ - printk(KERN_ERR "ide-tape: bug: Trying to free our active pipeline stage\n");
+ + printk(KERN_ERR "ide-tape: bug: Trying to free our active "
+ "pipeline stage\n");
+ return;
+ }
stage = tape->first_stage;
@@ -940,9 +802,11 @@ static void idetape_remove_stage_head (ide_drive_t *drive)
if (tape->first_stage == NULL) {
tape->last_stage = NULL;
if (tape->next_stage != NULL)
- printk(KERN_ERR "ide-tape: bug: tape->next_stage != NULL\n");
+ printk(KERN_ERR "ide-tape: bug: tape->next_stage !="
+ " NULL\n");
if (tape->nr_stages)
```

[git patches] IDE updates part #5

```
- printk(KERN_ERR "ide-tape: bug: nr_stages should be 0 now\n");
+ printk(KERN_ERR "ide-tape: bug: nr_stages should be 0 "
+ "now\n");
}
}

@@ -957,10 +821,8 @@ static void idetape_abort_pipeline(ide_drive_t *drive,
idetape_stage_t *stage = new_last_stage->next;
idetape_stage_t *nstage;

-#if IDETAPE_DEBUG_LOG
- if (tape->debug_level >= 4)
- printk(KERN_INFO "ide-tape: %s: idetape_abort_pipeline called\n", tape->name);
-#endif
+ debug_log(DBG_PROCS, "%s: Enter %s\n", tape->name, __func__);
+
while (stage) {
nstage = stage->next;
idetape_kfree_stage(tape, stage);
@@ -975,8 +837,8 @@ static void idetape_abort_pipeline(ide_drive_t *drive,
}

/*
- * idetape_end_request is used to finish servicing a request, and to
- * insert a pending pipeline request into the main device queue.
+ * Finish servicing a request and insert a pending pipeline request into the
+ * main device queue.
*/
static int idetape_end_request(ide_drive_t *drive, int uptodate, int nr_sects)
{
@@ -987,15 +849,12 @@ static int idetape_end_request(ide_drive_t *drive, int uptodate, int nr_sects)
int remove_stage = 0;
idetape_stage_t *active_stage;

-#if IDETAPE_DEBUG_LOG
- if (tape->debug_level >= 4)
- printk(KERN_INFO "ide-tape: Reached idetape_end_request\n");
-#endif /* IDETAPE_DEBUG_LOG */
+ debug_log(DBG_PROCS, "Enter %s\n", __func__);

switch (uptodate) {
- case 0: error = IDETAPE_ERROR_GENERAL; break;
- case 1: error
```