

[git pull] IDE updates #3

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2008-10/msg06848.html>

- *From:* Bartłomiej Zolnierkiewicz <bzolnier@xxxxxxxxx>
 - *Date:* Fri, 17 Oct 2008 18:46:29 +0200
-

* Bugfix for IRQ masking prior to setting transfer mode on the device.
(Sergei Shtylyov)

* PM support for delkin_cb host driver (fixes bugzilla bug #11735).

* ide-cd fixups and debug log enhancements. (Borislav Petkov)

* Fix compile warning in hpt366 (Sergei/me) and scc_pata (Sergei).

* Generic ATA/ATAPI disk driver (ide-gd) replacing ide-{disk,floppy} drivers. This makes the code more maintainable and easier to extend.

There are also some immediate benefits:

- driver specific debugging support for ATA devices
- extended device numbers support for ATAPI devices
- driver specific PM support for ATAPI devices

* Removal of the broken/dead hpt34x host driver (proper support for HPT343/345 is provided by pata_hpt3x3 host driver from Alan Cox).

* Misc fixes/cleanups.

Linus, please pull from:

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

to receive the following updates:

```
arch/arm/mach-sa1100/include/mach/ide.h | 75 -----
drivers/ide/Kconfig | 88 +++-----
drivers/ide/Makefile | 19 +-
drivers/ide/ide-atapi.c | 2 +-
drivers/ide/ide-cd.c | 61 +++--
drivers/ide/ide-cd_ioctl.c | 8 +-
drivers/ide/ide-disk.c | 382 +++++-----
drivers/ide/ide-disk.h | 21 +-
drivers/ide/ide-disk_ioctl.c | 4 +-

```

[git pull] IDE updates #3

```
drivers/ide/ide-disk_proc.c | 2 +-
drivers/ide/ide-dma-sff.c | 2 +-
drivers/ide/ide-floppy.c | 357 +++++-----
drivers/ide/ide-floppy.h | 41 +----
drivers/ide/ide-floppy_ioctl.c | 23 +-
drivers/ide/ide-floppy_proc.c | 2 +-
drivers/ide/ide-gd.c | 398 ++++++
drivers/ide/ide-gd.h | 44 +++++
drivers/ide/ide-iops.c | 2 +-
drivers/ide/ide-probe.c | 1 +
drivers/ide/ide-proc.c | 6 +-
drivers/ide/ide-tape.c | 16 +-
drivers/ide/pci/Makefile | 1 -
drivers/ide/pci/delkin_cb.c | 63 +++++-
drivers/ide/pci/hpt34x.c | 193 -----
drivers/ide/pci/hpt366.c | 35 +---
drivers/ide/pci/scc_pata.c | 4 -
drivers/ide/pci/sgiioc4.c | 49 +---
drivers/leds/Kconfig | 2 +-
drivers/scsi/ide-scsi.c | 26 +-
include/asm-frv/ide.h | 10 +-
include/asm-m68k/ide.h | 9 -
include/asm-parisc/ide.h | 4 -
include/linux/ide.h | 36 +-
33 files changed, 815 insertions(+), 1171 deletions(-)
delete mode 100644 arch/arm/mach-sa1100/include/mach/ide.h
create mode 100644 drivers/ide/ide-gd.c
create mode 100644 drivers/ide/ide-gd.h
delete mode 100644 drivers/ide/pci/hpt34x.c
```

Bartłomiej Zolnierkiewicz (27):

- ide-disk: fix IDE_DFLAG_LBA48 handling on resume
- ide-disk: lock media before checking for media change
- ide-floppy: use alloc_disk_node()
- ide-disk: use to_ide_drv() and ide_drv_g()
- ide-disk: move IDE_DFLAG_DOORLOCKING flag handling to idedisk_set_doorlock()
- ide-{disk,floppy}: set IDE_DFLAG_ATTACH in *_setup()
- ide-floppy: drop 'floppy' argument from idefloppy_setup()
- ide-floppy: use drive->capacity64 for caching current capacity
- ide: IDE_AFLAG_MEDIA_CHANGED -> IDE_DFLAG_MEDIA_CHANGED
- ide: IDE_AFLAG_WP -> IDE_DFLAG_WP
- ide: IDE_AFLAG_FORMAT_IN_PROGRESS -> IDE_DFLAG_FORMAT_IN_PROGRESS
- ide: remove IDE_AFLAG_NO_DOORLOCKING
- ide-disk: factor out generic disk handling code to ide-gd.c
- ide-disk: use IDE_DFLAG_MEDIA_CHANGED
- ide-floppy: factor out generic disk handling code to ide-gd-floppy.c
- ide: prepare for merging ide-gd-floppy.c with ide-gd.c
- ide: allow device drivers to specify per-device type /proc settings
- ide: add generic ATA/ATAPI disk driver
- ide: fix support for IDE PCI controllers using MMIO on frv

[git pull] IDE updates #3

ide: remove dead <asm-arm/arch-sa1100/ide.h>
ide: remove M68K_IDE_SWAPW define from <asm-m68k/ide.h>
ide: remove unused macros from <asm-parisc/ide.h>
hpt366: fix compile warning
ide-floppy: remove idefloppy_floppy_t typedef
ide: remove broken hpt34x driver
delkin_cb: add PM support
ide: re-add TRM290 fix lost during ide_build_dmatable() cleanup

Borislav Petkov (3):

ide-cd: debug log enhancements
ide-cd: small drive type print fix
ide-cd: remove stale comment

Sergei Shtylyov (6):

ide: mask interrupt in ide_config_drive_speed()
hpt366: cleanup maskproc() method
sgiioc4: remove maskproc() method
sgiioc4: kill useless address checks
sgiioc4: kill duplicate ioremap()
scc_pata: kill unused variables

```
diff --git a/arch/arm/mach-sa1100/include/mach/ide.h b/arch/arm/mach-sa1100/include/mach/ide.h
deleted file mode 100644
index 4c99c8f..0000000
--- a/arch/arm/mach-sa1100/include/mach/ide.h
+++ /dev/null
@@ -1,75 +0,0 @@
-/*
- * arch/arm/mach-sa1100/include/mach/ide.h
- *
- * Copyright (c) 1998 Hugo Fiennes & Nicolas Pitre
- *
- * 18-aug-2000: Cleanup by Erik Mouw (J.A.K.Mouw@xxxxxxxxxxxxxxxx)
- * Get rid of the special ide_init_hwif_ports() functions
- * and make a generalised function that can be used by all
- * architectures.
- */
-
-#include <asm/irq.h>
-#include <mach/hardware.h>
-#include <asm/mach-types.h>
-
-#error "This code is broken and needs update to match with current ide support"
-
-/*
- * Set up a hw structure for a specified data port, control port and IRQ.
- * This should follow whatever the default interface uses.
- */
```

[git pull] IDE updates #3

```
-static inline void ide_init_hwif_ports(hw_regs_t *hw, unsigned long data_port,
- unsigned long ctrl_port, int *irq)
-{
- unsigned long reg = data_port;
- int i;
- int regincr = 1;
-
- /* The Empeg board has the first two address lines unused */
- if (machine_is_empeg())
- regincr = 1 << 2;
-
- /* The LART doesn't use A0 for IDE */
- if (machine_is_lart())
- regincr = 1 << 1;
-
- memset(hw, 0, sizeof(*hw));
-
- for (i = 0; i <= 7; i++) {
- hw->io_ports_array[i] = reg;
- reg += regincr;
- }
-
- hw->io_ports.ctl_addr = ctrl_port;
-
- if (irq)
- *irq = 0;
-}
-
-/*
- * This registers the standard ports for this architecture with the IDE
- * driver.
- */
-static __inline__ void
-ide_init_default_hwifs(void)
-{
- if (machine_is_lart()) {
- #ifdef CONFIG_SA1100_LART
- hw_regs_t hw;
-
- /* Enable GPIO as interrupt line */
- GPDR &= ~LART_GPIO_IDE;
- set_irq_type(LART_IRQ_IDE, IRQ_TYPE_EDGE_RISING);
-
- /* set PCMCIA interface timing */
- MECR = 0x00060006;
-
- /* init the interface */
- ide_init_hwif_ports(&hw, PCMCIA_IO_0_BASE + 0x0000, PCMCIA_IO_0_BASE + 0x1000, NULL);
- hw.irq = LART_IRQ_IDE;
- ide_register_hw(&hw);
- #endif
-}
-}
-*/
```

```
- }
-}
diff --git a/drivers/ide/Kconfig b/drivers/ide/Kconfig
index 74a369a..a820ca6 100644
--- a/drivers/ide/Kconfig
+++ b/drivers/ide/Kconfig
@@ -84,21 +84,40 @@ config BLK_DEV_IDE_SATA
```

If unsure, say N.

```
-config BLK_DEV_IDEDISK
- tristate "Include IDE/ATA-2 DISK support"
- ---help---
- This will include enhanced support for MFM/RLL/IDE hard disks. If
- you have a MFM/RLL/IDE disk, and there is no special reason to use
- the old hard disk driver instead, say Y. If you have an SCSI-only
- system, you can say N here.
+config IDE_GD
+ tristate "generic ATA/ATAPI disk support"
+ default y
+ help
+ Support for ATA/ATAPI disks (including ATAPI floppy drives).
```

```
- To compile this driver as a module, choose M here: the
- module will be called ide-disk.
- Do not compile this driver as a module if your root file system
- (the one containing the directory /) is located on the IDE disk.
+ To compile this driver as a module, choose M here.
+ The module will be called ide-gd_mod.
```

```
+
+ If unsure, say Y.
+
+config IDE_GD_ATA
+ bool "ATA disk support"
+ depends on IDE_GD
+ default y
+ help
+ This will include support for ATA hard disks.
```

If unsure, say Y.

```
+config IDE_GD_ATAPI
+ bool "ATAPI floppy support"
+ depends on IDE_GD
+ select IDE_ATAPI
+ help
+ This will include support for ATAPI floppy drives
+ (i.e. Iomega ZIP or MKE LS-120).
+
+ For information about jumper settings and the question
+ of when a ZIP drive uses a partition table, see
```

[git pull] IDE updates #3

+ <<http://www.win.tue.nl/~aeb/linux/zip/zip-1.html>>.

+

+ If unsure, say N.

+

config BLK_DEV_IDECS

tristate "PCMCIA IDE support"

depends on PCMCIA

@@ -163,29 +182,6 @@ config BLK_DEV_IDETAPE

To compile this driver as a module, choose M here: the module will be called ide-tape.

-config BLK_DEV_IDEFLOPPY

- tristate "Include IDE/ATAPI FLOPPY support"

- select IDE_ATAPI

- ---help---

- If you have an IDE floppy drive which uses the ATAPI protocol,

- answer Y. ATAPI is a newer protocol used by IDE CD-ROM/tape/floppy

- drives, similar to the SCSI protocol.

-

- The LS-120 and the IDE/ATAPI Iomega ZIP drive are also supported by

- this driver. For information about jumper settings and the question

- of when a ZIP drive uses a partition table, see

- <<http://www.win.tue.nl/~aeb/linux/zip/zip-1.html>>.

- (ATAPI PD-CD/CDR drives are not supported by this driver; support

- for PD-CD/CDR drives is available if you answer Y to

- "SCSI emulation support", below).

-

- If you say Y here, the FLOPPY drive will be identified along with

- other IDE devices, as "hdb" or "hdc", or something similar (check

- the boot messages with dmesg).

-

- To compile this driver as a module, choose M here: the

- module will be called ide-floppy.

-

config BLK_DEV_IDESCSI

tristate "SCSI emulation support (DEPRECATED)"

depends on SCSI

@@ -332,7 +328,7 @@ config IDEPCI_PCIBUS_ORDER

TODO: split it on per host driver config options (or module parameters)

config BLK_DEV_OFFBOARD

bool "Boot off-board chipsets first support (DEPRECATED)"

- depends on BLK_DEV_IDEPCI && (BLK_DEV_AEC62XX || BLK_DEV_GENERIC ||

BLK_DEV_HPT34X || BLK_DEV_HPT366 || BLK_DEV_PDC202XX_NEW ||

BLK_DEV_PDC202XX_OLD || BLK_DEV_TC86C001)

+ depends on BLK_DEV_IDEPCI && (BLK_DEV_AEC62XX || BLK_DEV_GENERIC ||

BLK_DEV_HPT366 || BLK_DEV_PDC202XX_NEW || BLK_DEV_PDC202XX_OLD ||

BLK_DEV_TC86C001)

help

Normally, IDE controllers built into the motherboard (on-board

controllers) are assigned to ide0 and ide1 while those on add-in PCI

@@ -482,28 +478,6 @@ config BLK_DEV_CS5535

It is safe to say Y to this question.

```
-config BLK_DEV_HPT34X
- tristate "HPT34X chipset support"
- depends on BROKEN
- select BLK_DEV_IDEDMA_PCI
- help
- This driver adds up to 4 more EIDE devices sharing a single
- interrupt. The HPT343 chipset in its current form is a non-bootable
- controller; the HPT345/HPT363 chipset is a bootable (needs BIOS FIX)
- PCI UDMA controllers. This driver requires dynamic tuning of the
- chipset during the ide-probe at boot time. It is reported to support
- DVD II drives, by the manufacturer.
```

```
-
-
- config HPT34X_AUTODMA
- bool "HPT34X AUTODMA support (EXPERIMENTAL)"
- depends on BLK_DEV_HPT34X && EXPERIMENTAL
- help
- This is a dangerous thing to attempt currently! Please read the
- comments at the top of <file:drivers/ide/pci/hpt34x.c>. If you say Y
- here, then say Y to "Use DMA by default when available" as well.
-
- If unsure, say N.
```

```
config BLK_DEV_HPT366
tristate "HPT36X/37X chipset support"
select BLK_DEV_IDEDMA_PCI
diff --git a/drivers/ide/Makefile b/drivers/ide/Makefile
index ceaf779..093d324 100644
--- a/drivers/ide/Makefile
+++ b/drivers/ide/Makefile
@@ -37,18 +37,25 @@ obj-$(CONFIG_IDE_H8300) += h8300/
obj-$(CONFIG_IDE_GENERIC) += ide-generic.o
obj-$(CONFIG_BLK_DEV_IDEPNP) += ide-pnp.o
```

```
-ide-disk_mod-y += ide-disk.o ide-disk_ioctl.o
+ide-gd_mod-y += ide-gd.o
ide-cd_mod-y += ide-cd.o ide-cd_ioctl.o ide-cd_verbose.o
-ide-floppy_mod-y += ide-floppy.o ide-floppy_ioctl.o
```

```
+ifeq ($(CONFIG_IDE_GD_ATA), y)
+ ide-gd_mod-y += ide-disk.o ide-disk_ioctl.o
ifeq ($(CONFIG_IDE_PROC_FS), y)
- ide-disk_mod-y += ide-disk_proc.o
- ide-floppy_mod-y += ide-floppy_proc.o
+ ide-gd_mod-y += ide-disk_proc.o
+endif
+endif
+
+ifeq ($(CONFIG_IDE_GD_ATAPI), y)
```

[git pull] IDE updates #3

```
+ ide-gd_mod-y += ide-floppy.o ide-floppy_ioctl.o
+ifeq ($(CONFIG_IDE_PROC_FS), y)
+ ide-gd_mod-y += ide-floppy_proc.o
+endif
endif

-obj-$(CONFIG_BLK_DEV_IDEDISK) += ide-disk_mod.o
+obj-$(CONFIG_IDE_GD) += ide-gd_mod.o
obj-$(CONFIG_BLK_DEV_IDECD) += ide-cd_mod.o
-obj-$(CONFIG_BLK_DEV_IDEFLOPPY) += ide-floppy_mod.o
obj-$(CONFIG_BLK_DEV_IDETAPE) += ide-tape.o

ifeq ($(CONFIG_BLK_DEV_IDECS), y)
diff --git a/drivers/ide/ide-atapi.c b/drivers/ide/ide-atapi.c
index 2e30571..4e58b9e 100644
--- a/drivers/ide/ide-atapi.c
+++ b/drivers/ide/ide-atapi.c
@@ -191,7 +191,7 @@ int ide_set_media_lock(ide_drive_t *drive, struct gendisk *disk, int on)
{
struct ide_atapi_pc pc;

- if (drive->atapi_flags & IDE_AFLAG_NO_DOORLOCK)
+ if ((drive->dev_flags & IDE_DFLAG_DOORLOCKING) == 0)
return 0;

ide_init_pc(&pc);
diff --git a/drivers/ide/ide-cd.c b/drivers/ide/ide-cd.c
index 3308b1c..13265a8 100644
--- a/drivers/ide/ide-cd.c
+++ b/drivers/ide/ide-cd.c
@@ -99,7 +99,7 @@ static void ide_cd_put(struct cdrom_info *cd)
/* Mark that we've seen a media change and invalidate our internal buffers. */
static void cdrom_saw_media_change(ide_drive_t *drive)
{
- drive->atapi_flags |= IDE_AFLAG_MEDIA_CHANGED;
+ drive->dev_flags |= IDE_DFLAG_MEDIA_CHANGED;
drive->atapi_flags &= ~IDE_AFLAG_TOC_VALID;
}

@@ -340,8 +340,8 @@ static int cdrom_decode_status(ide_drive_t *drive, int good_stat, int *stat_ret)
}

ide_debug_log(IDE_DBG_RQ, "%s: stat: 0x%x, good_stat: 0x%x, "
- "rq->cmd_type: 0x%x, err: 0x%x\n", __func__, stat,
- good_stat, rq->cmd_type, err);
+ "rq->cmd[0]: 0x%x, rq->cmd_type: 0x%x, err: 0x%x\n",
+ __func__, stat, good_stat, rq->cmd[0], rq->cmd_type, err);

if (blk_sense_request(rq)) {
/*
@@ -843,13 +843,10 @@ static void ide_cd_restore_request(ide_drive_t *drive, struct request *rq)
```

[git pull] IDE updates #3

```
rq->q->prep_rq_fn(rq->q, rq);
}

-/*
- * All other packet commands.
- */
static void ide_cd_request_sense_fixup(ide_drive_t *drive, struct request *rq)
{
-
- ide_debug_log(IDE_DBG_FUNC, "Call %s\n", __func__);
+ ide_debug_log(IDE_DBG_FUNC, "Call %s, rq->cmd[0]: 0x%x\n",
+ __func__, rq->cmd[0]);

/*
* Some of the trailing request sense fields are optional,
@@ -876,7 +873,7 @@ int ide_cd_queue_pc(ide_drive_t *drive, const unsigned char *cmd,
if (!sense)
sense = &local_sense;

- ide_debug_log(IDE_DBG_PC, "Call %s, rq->cmd[0]: 0x%x, write: 0x%x, "
+ ide_debug_log(IDE_DBG_PC, "Call %s, cmd[0]: 0x%x, write: 0x%x, "
"timeout: %d, cmd_flags: 0x%x\n", __func__, cmd[0], write,
timeout, cmd_flags);

@@ -1177,8 +1174,9 @@ static ide_startstop_t cdrom_start_rw(ide_drive_t *drive, struct request *rq)
unsigned short sectors_per_frame =
queue_hardsect_size(drive->queue) >> SECTOR_BITS;

- ide_debug_log(IDE_DBG_RQ, "Call %s, write: 0x%x, secs_per_frame: %u\n",
- __func__, write, sectors_per_frame);
+ ide_debug_log(IDE_DBG_RQ, "Call %s, rq->cmd[0]: 0x%x, write: 0x%x, "
+ "secs_per_frame: %u\n",
+ __func__, rq->cmd[0], write, sectors_per_frame);

if (write) {
/* disk has become write protected */
@@ -1221,7 +1219,8 @@ static ide_startstop_t cdrom_do_newpc_cont(ide_drive_t *drive)
static void cdrom_do_block_pc(ide_drive_t *drive, struct request *rq)
{

- ide_debug_log(IDE_DBG_PC, "Call %s, rq->cmd_type: 0x%x\n", __func__,
+ ide_debug_log(IDE_DBG_PC, "Call %s, rq->cmd[0]: 0x%x, "
+ "rq->cmd_type: 0x%x\n", __func__, rq->cmd[0],
rq->cmd_type);

if (blk_pc_request(rq))
@@ -1257,9 +1256,6 @@ static void cdrom_do_block_pc(ide_drive_t *drive, struct request *rq)
}
}

-/*
```

[git pull] IDE updates #3

```
- * cdrom driver request routine.
- */
static ide_startstop_t ide_cd_do_request(ide_drive_t *drive, struct request *rq,
sector_t block)
{
@@ -1267,8 +1263,10 @@ static ide_startstop_t ide_cd_do_request(ide_drive_t *drive, struct request *rq,
ide_handler_t *fn;
int xferlen;

- ide_debug_log(IDE_DBG_RQ, "Call %s, rq->cmd_type: 0x%x, block: %llu\n",
- __func__, rq->cmd_type, (unsigned long long)block);
+ ide_debug_log(IDE_DBG_RQ, "Call %s, rq->cmd[0]: 0x%x, "
+ "rq->cmd_type: 0x%x, block: %llu\n",
+ __func__, rq->cmd[0], rq->cmd_type,
+ (unsigned long long)block);

if (blk_fs_request(rq)) {
if (drive->atapi_flags & IDE_AFLAG_SEEKING) {
@@ -1412,6 +1410,10 @@ static int cdrom_read_capacity(ide_drive_t *drive, unsigned long *capacity,

*capacity = 1 + be32_to_cpu(capbuf.lba);
*sectors_per_frame = blocklen >> SECTOR_BITS;
+
+ ide_debug_log(IDE_DBG_PROBE, "%s: cap: %lu, sectors_per_frame: %lu\n",
+ __func__, *capacity, *sectors_per_frame);
+
return 0;
}

@@ -1643,6 +1645,9 @@ void ide_cdrom_update_speed(ide_drive_t *drive, u8 *buf)
maxspeed = be16_to_cpup((__be16 *)&buf[8 + 8]);
}

+ ide_debug_log(IDE_DBG_PROBE, "%s: curspeed: %u, maxspeed: %u\n",
+ __func__, curspeed, maxspeed);
+
cd->current_speed = (curspeed + (176/2)) / 176;
cd->max_speed = (maxspeed + (176/2)) / 176;
}
@@ -1732,7 +1737,7 @@ static int ide_cdrom_probe_capabilities(ide_drive_t *drive)
return 0;

if ((buf[8 + 6] & 0x01) == 0)
- drive->atapi_flags |= IDE_AFLAG_NO_DOORLOCK;
+ drive->dev_flags &= ~IDE_DFLAG_DOORLOCKING;
if (buf[8 + 6] & 0x08)
drive->atapi_flags &= ~IDE_AFLAG_NO_EJECT;
if (buf[8 + 3] & 0x01)
@@ -1777,7 +1782,7 @@ static int ide_cdrom_probe_capabilities(ide_drive_t *drive)
if ((cdi->mask & CDC_DVD_R) == 0 || (cdi->mask & CDC_DVD_RAM) == 0)
printk(KERN_CONT " DVD%s%s",
```

[git pull] IDE updates #3

```
(cdi->mask & CDC_DVD_R) ? "" : "-R",
- (cdi->mask & CDC_DVD_RAM) ? "" : "-RAM");
+ (cdi->mask & CDC_DVD_RAM) ? "" : "/RAM");

if ((cdi->mask & CDC_CD_R) == 0 || (cdi->mask & CDC_CD_RW) == 0)
printk(KERN_CONT " CD%s%s",
@@ -1908,6 +1913,16 @@ static const struct ide_proc_devset idecd_settings[] = {
IDE_PROC_DEVSET(dsc_overlap, 0, 1),
{ 0 },
};
+
+static ide_proc_entry_t *ide_cd_proc_entries(ide_drive_t *drive)
+{
+ return idecd_proc;
+}
+
+static const struct ide_proc_devset *ide_cd_proc_devsets(ide_drive_t *drive)
+{
+ return idecd_settings;
+}
#endif

static const struct cd_list_entry ide_cd_quirks_list[] = {
@@ -1986,8 +2001,8 @@ static int ide_cdrom_setup(ide_drive_t *drive)
if (!drive->queue->unplug_delay)
drive->queue->unplug_delay = 1;

- drive->atapi_flags = IDE_AFLAG_MEDIA_CHANGED | IDE_AFLAG_NO_EJECT |
- ide_cd_flags(id);
+ drive->dev_flags |= IDE_DFLAG_MEDIA_CHANGED;
+ drive->atapi_flags = IDE_AFLAG_NO_EJECT | ide_cd_flags(id);

if ((drive->atapi_flags & IDE_AFLAG_VERTOS_300_SSD) &&
fw_rev[4] == '1' && fw_rev[6] <= '2')
@@ -2069,8 +2084,8 @@ static ide_driver_t ide_cdrom_driver = {
.end_request = ide_end_request,
.error = __ide_error,
#ifdef CONFIG_IDE_PROC_FS
- .proc = idecd_proc,
- .settings = idecd_settings,
+ .proc_entries = ide_cd_proc_entries,
+ .proc_devsets = ide_cd_proc_devsets,
#endif
};

diff --git a/drivers/ide/ide-cd_ioctl.c b/drivers/ide/ide-cd_ioctl.c
index 74231b4..df3df00 100644
--- a/drivers/ide/ide-cd_ioctl.c
+++ b/drivers/ide/ide-cd_ioctl.c
@@ -86,8 +86,8 @@ int ide_cdrom_check_media_change_real(struct cdrom_device_info *cdi,
```

[git pull] IDE updates #3

```
if (slot_nr == CDSL_CURRENT) {
(void) cdrom_check_status(drive, NULL);
- retval = (drive->atapi_flags & IDE_AFLAG_MEDIA_CHANGED) ? 1 : 0;
- drive->atapi_flags &= ~IDE_AFLAG_MEDIA_CHANGED;
+ retval = (drive->dev_flags & IDE_DFLAG_MEDIA_CHANGED) ? 1 : 0;
+ drive->dev_flags &= ~IDE_DFLAG_MEDIA_CHANGED;
return retval;
} else {
return -EINVAL;
@@ -136,7 +136,7 @@ int ide_cd_lockdoor(ide_drive_t *drive, int lockflag,
sense = &my_sense;

/* If the drive cannot lock the door, just pretend. */
- if (drive->atapi_flags & IDE_AFLAG_NO_DOORLOCK) {
+ if ((drive->dev_flags & IDE_DFLAG_DOORLOCKING) == 0) {
stat = 0;
} else {
unsigned char cmd[BLK_MAX_CDB];
@@ -157,7 +157,7 @@ int ide_cd_lockdoor(ide_drive_t *drive, int lockflag,
(sense->asc == 0x24 || sense->asc == 0x20)) {
printk(KERN_ERR "%s: door locking not supported\n",
drive->name);
- drive->atapi_flags |= IDE_AFLAG_NO_DOORLOCK;
+ drive->dev_flags &= ~IDE_DFLAG_DOORLOCKING;
stat = 0;
}

diff --git a/drivers/ide/ide-disk.c b/drivers/ide/ide-disk.c
index 3853bde..223750c 100644
--- a/drivers/ide/ide-disk.c
+++ b/drivers/ide/ide-disk.c
@@ -14,9 +14,6 @@
* This is the IDE/ATA disk driver, as evolved from hd.c and ide.c.
*/

-#define IDEDISK_VERSION "1.18"
-
-#include <linux/module.h>
#include <linux/types.h>
#include <linux/string.h>
#include <linux/kernel.h>
@@ -39,46 +36,8 @@
#include <asm/io.h>
#include <asm/div64.h>

-#if !defined(CONFIG_DEBUG_BLOCK_EXT_DEVT)
-#define IDE_DISK_MINORS (1 << PARTN_BITS)
-#else
-#define IDE_DISK_MINORS 0
-#endif
-
```

[git pull] IDE updates #3

```
#include "ide-disk.h"

-static DEFINE_MUTEX(idedisk_ref_mutex);
-
-#define to_ide_disk(obj) container_of(obj, struct ide_disk_obj, kref)
-
-static void ide_disk_release(struct kref *);
-
-static struct ide_disk_obj *ide_disk_get(struct gendisk *disk)
-{
- struct ide_disk_obj *idkp = NULL;
-
- mutex_lock(&idedisk_ref_mutex);
- idkp = ide_disk_g(disk);
- if (idkp) {
- if (ide_device_get(idkp->drive))
- idkp = NULL;
- else
- kref_get(&idkp->kref);
- }
- mutex_unlock(&idedisk_ref_mutex);
- return idkp;
-}
-
-static void ide_disk_put(struct ide_disk_obj *idkp)
-{
- ide_drive_t *drive = idkp->drive;
-
- mutex_lock(&idedisk_ref_mutex);
- kref_put(&idkp->kref, ide_disk_release);
- ide_device_put(drive);
- mutex_unlock(&idedisk_ref_mutex);
-}
-
static const u8 ide_rw_cmds[] = {
ATA_CMD_READ_MULTI,
ATA_CMD_WRITE_MULTI,
@@ -374,7 +333,7 @@ static void idedisk_check_hpa(ide_drive_t *drive)
}
}

-static void init_idedisk_capacity(ide_drive_t *drive)
+static int ide_disk_get_capacity(ide_drive_t *drive)
{
u16 *id = drive->id;
int lba;
@@ -403,11 +362,28 @@ static void init_idedisk_capacity(ide_drive_t *drive)
if (ata_id_hpa_enabled(id))
idedisk_check_hpa(drive);
}
-}

```

[git pull] IDE updates #3

```
-sector_t ide_disk_capacity(ide_drive_t *drive)
- {
- return drive->capacity64;
+ /* limit drive capacity to 137GB if LBA48 cannot be used */
+ if ((drive->dev_flags & IDE_DFLAG_LBA48) == 0 &&
+ drive->capacity64 > 1ULL << 28) {
+ printk(KERN_WARNING "%s: cannot use LBA48 - full capacity "
+ "%llu sectors (%llu MB)\n",
+ drive->name, (unsigned long long)drive->capacity64,
+ sectors_to_MB(drive->capacity64));
+ drive->capacity64 = 1ULL << 28;
+ }
+
+ if ((drive->hwif->host_flags & IDE_HFLAG_NO_LBA48_DMA) &&
+ (drive->dev_flags & IDE_DFLAG_LBA48)) {
+ if (drive->capacity64 > 1ULL << 28) {
+ printk(KERN_INFO "%s: cannot use LBA48 DMA - PIO mode"
+ " will be used for accessing sectors "
+ "> %u\n", drive->name, 1 << 28);
+ } else
+ drive->dev_flags &= ~IDE_DFLAG_LBA48;
+ }
+
+ return 0;
}

static void idedisk_prepare_flush(struct request_queue *q, struct request *rq)
@@ -508,7 +484,7 @@ static void update_ordered(ide_drive_t *drive)
* time we have trimmed the drive capacity if LBA48 is
* not available so we don't need to recheck that.
*/
- capacity = ide_disk_capacity(drive);
+ capacity = ide_gd_capacity(drive);
barrier = ata_id_flush_enabled(id) &&
(drive->dev_flags & IDE_DFLAG_NOFLUSH) == 0 &&
((drive->dev_flags & IDE_DFLAG_LBA48) == 0 ||
@@ -616,7 +592,12 @@ ide_ext_devset_rw(wcache, wcache);

ide_ext_devset_rw_sync(nowerr, nowerr);

-static void idedisk_setup(ide_drive_t *drive)
+static int ide_disk_check(ide_drive_t *drive, const char *s)
+ {
+ return 1;
+ }
+
+static void ide_disk_setup(ide_drive_t *drive)
{
struct ide_disk_obj *idkp = drive->driver_data;
ide_hwif_t *hwif = drive->hwif;
```

[git pull] IDE updates #3

```
@@ -652,33 +633,13 @@ static void idedisk_setup(ide_drive_t *drive)
drive->queue->max_sectors / 2);

/* calculate drive capacity, and select LBA if possible */
- init_idedisk_capacity(drive);
-
- /* limit drive capacity to 137GB if LBA48 cannot be used */
- if ((drive->dev_flags & IDE_DFLAG_LBA48) == 0 &&
- drive->capacity64 > 1ULL << 28) {
- printk(KERN_WARNING "%s: cannot use LBA48 - full capacity "
- "%llu sectors (%llu MB)\n",
- drive->name, (unsigned long long)drive->capacity64,
- sectors_to_MB(drive->capacity64));
- drive->capacity64 = 1ULL << 28;
- }
-
- if ((hwif->host_flags & IDE_HFLAG_NO_LBA48_DMA) &&
- (drive->dev_flags & IDE_DFLAG_LBA48)) {
- if (drive->capacity64 > 1ULL << 28) {
- printk(KERN_INFO "%s: cannot use LBA48 DMA - PIO mode"
- " will be used for accessing sectors "
- "> %u\n", drive->name, 1 << 28);
- } else
- drive->dev_flags &= ~IDE_DFLAG_LBA48;
- }
+ ide_disk_get_capacity(drive);

/*
* if possible, give fdisk access to more of the drive,
* by correcting bios_cyls:
*/
- capacity = ide_disk_capacity(drive);
+ capacity = ide_gd_capacity(drive);

if ((drive->dev_flags & IDE_DFLAG_FORCED_GEOM) == 0) {
if (ata_id_lba48_enabled(drive->id)) {
@@ -718,9 +679,17 @@ static void idedisk_setup(ide_drive_t *drive)
drive->dev_flags |= IDE_DFLAG_WCACHE;

set_wcache(drive, 1);
+
+ if ((drive->dev_flags & IDE_DFLAG_LBA) == 0 &&
+ (drive->head == 0 || drive->head > 16)) {
+ printk(KERN_ERR "%s: invalid geometry: %d physical heads?\n",
+ drive->name, drive->head);
+ drive->dev_flags &= ~IDE_DFLAG_ATTACH;
+ } else
+ drive->dev_flags |= IDE_DFLAG_ATTACH;
+ }

-static void ide_cacheflush_p(ide_drive_t *drive)
```

[git pull] IDE updates #3

```
+static void ide_disk_flush(ide_drive_t *drive)
{
if (ata_id_flush_enabled(drive->id) == 0 ||
(drive->dev_flags & IDE_DFLAG_WCACHE) == 0)
@@ -730,267 +699,40 @@ static void ide_cacheflush_p(ide_drive_t *drive)
printk(KERN_INFO "%s: wcache flush failed!\n", drive->name);
}

-static void ide_disk_remove(ide_drive_t *drive)
-{-
- struct ide_disk_obj *idkp = drive->driver_data;
- struct gendisk *g = idkp->disk;
-
- ide_proc_unregister_driver(drive, idkp->driver);
-
- del_gendisk(g);
-
- ide_cacheflush_p(drive);
-
- ide_disk_put(idkp);
-}
-
-static void ide_disk_release(struct kref *kref)
-{-
- struct ide_disk_obj *idkp = to_ide_disk(kref);
- ide_drive_t *drive = idkp->drive;
- struct gendisk *g = idkp->disk;
-
- drive->driver_data = NULL;
- g->private_data = NULL;
- put_disk(g);
- kfree(idkp);
-}
-
-static int ide_disk_probe(ide_drive_t *drive);
-
-/*
- * On HPA drives the capacity needs to be
- * reinitialized on resume otherwise the disk
- * can not be used and a hard reset is required
- */
-static void ide_disk_resume(ide_drive_t *drive)
+static int ide_disk_init_media(ide_drive_t *drive, struct gendisk *disk)
{
- if (ata_id_hpa_enabled(drive->id))
- init_idedisk_capacity(drive);
-}
-
-static void ide_device_shutdown(ide_drive_t *drive)
-{-
-#ifdef CONFIG_ALPHA
```

[git pull] IDE updates #3

```
- /* On Alpha, halt(8) doesn't actually turn the machine off,
- it puts you into the sort of firmware monitor. Typically,
- it's used to boot another kernel image, so it's not much
- different from reboot(8). Therefore, we don't need to
- spin down the disk in this case, especially since Alpha
- firmware doesn't handle disks in standby mode properly.
- On the other hand, it's reasonably safe to turn the power
- off when the shutdown process reaches the firmware prompt,
- as the firmware initialization takes rather long time -
- at least 10 seconds, which should be sufficient for
- the disk to expire its write cache. */
- if (system_state != SYSTEM_POWER_OFF) {
-#else
- if (system_state == SYSTEM_RESTART) {
-#endif
- ide_cacheflush_p(drive);
- return;
- }
-
- printk(KERN_INFO "Shutdown: %s\n", drive->name);
-
- drive->gendev.bus->suspend(&drive->gendev, PMSG_SUSPEND);
+ return 0;
}

-static ide_driver_t idedisk_driver = {
- .gen_driver = {
- .owner = THIS_MODULE,
- .name = "ide-disk",
- .bus = &ide_bus_type,
- },
- .probe = ide_disk_probe,
- .remove = ide_disk_remove,
- .resume = ide_disk_resume,
- .shutdown = ide_device_shutdown,
- .version = IDEDISK_VERSION,
- .do_request = ide_do_rw_disk,
- .end_request = ide_end_request,
- .error = __ide_error,
-#ifdef CONFIG_IDE_PROC_FS
- .proc = ide_disk_proc,
- .settings = ide_disk_settings,
-#endif
-};
-
-static int idedisk_set_doorlock(ide_drive_t *drive, int on)
+static int ide_disk_set_doorlock(ide_drive_t *drive, struct gendisk *disk,
+ int on)
{
ide_task_t task;
+ int ret;
```

[git pull] IDE updates #3

```
+
+ if ((drive->dev_flags & IDE_DFLAG_DOORLOCKING) == 0)
+ return 0;

memset(&task, 0, sizeof(task));
task.tf.command = on ? ATA_CMD_MEDIA_LOCK : ATA_CMD_MEDIA_UNLOCK;
task.tf.flags = IDE_TFLAG_TF | IDE_TFLAG_DEVICE;

- return ide_no_data_taskfile(drive, &task);
-}
-
-static int idedisk_open(struct inode *inode, struct file *filp)
-{
- struct gendisk *disk = inode->i_bdev->bd_disk;
- struct ide_disk_obj *idkp;
- ide_drive_t *drive;
-
- idkp = ide_disk_get(disk);
- if (idkp == NULL)
- return -ENXIO;
-
- drive = idkp->drive;
-
- idkp->openers++;
-
- if ((drive->dev_flags & IDE_DFLAG_REMOVABLE) && idkp->openers == 1) {
- check_disk_change(inode->i_bdev);
- /*
- * Ignore the return code from door_lock,
- * since the open() has already succeeded,
- * and the door_lock is irrelevant at this point.
- */
- if ((drive->dev_flags & IDE_DFLAG_DOORLOCKING) &&
- idedisk_set_doorlock(drive, 1))
- drive->dev_flags &= ~IDE_DFLAG_DOORLOCKING;
- }
- return 0;
-}
-
-static int idedisk_release(struct inode *inode, struct file *filp)
-{
- struct gendisk *disk = inode->i_bdev->bd_disk;
- struct ide_disk_obj *idkp = ide_disk_g(disk);
- ide_drive_t *drive = idkp->drive;
-
- if (idkp->openers == 1)
- ide_cacheflush_p(drive);
-
- if ((drive->dev_flags & IDE_DFLAG_REMOVABLE) && idkp->openers == 1) {
- if ((drive->dev_flags & IDE_DFLAG_DOORLOCKING) &&
- idedisk_set_doorlock(drive, 0))
```

[git pull] IDE updates #3

```
- drive->dev_flags &= ~IDE_DFLAG_DOORLOCKING;
- }
+ ret = ide_no_data_taskfile(drive, &task);

- idkp->openers--;
+ if (ret)
+ drive->dev_flags &= ~IDE_DFLAG_DOORLOCKING;

- ide_disk_put(idkp);
-
- return 0;
-}
-
-static int idedisk_getgeo(struct block_device *bdev, struct hd_geometry *geo)
-{
- struct ide_disk_obj *idkp = ide_disk_g(bdev->bd_disk);
- ide_drive_t *drive = idkp->drive;
-
- geo->heads = drive->bios_head;
- geo->sectors = drive->bios_sect;
- geo->cylinders = (u16)drive->bios_cyl; /* truncate */
- return 0;
+ return ret;
}

-static int idedisk_media_changed(struct gendisk *disk)
-{
- struct ide_disk_obj *idkp = ide_disk_g(disk);
- ide_drive_t *drive = idkp->drive;
-
- /* do not scan partitions twice if this is a removable device */
- if (drive->dev_flags & IDE_DFLAG_ATTACH) {
- drive->dev_flags &= ~IDE_DFLAG_ATTACH;
- return 0;
- }
-
- /* if removable, always assume it was changed */
- return !(drive->dev_flags & IDE_DFLAG_REMOVABLE);
-}
-
-static int idedisk_revalidate_disk(struct gendisk *disk)
-{
- struct ide_disk_obj *idkp = ide_disk_g(disk);
- set_capacity(disk, ide_disk_capacity(idkp->drive));
- return 0;
-}
-
-static struct block_device_operations idedisk_ops = {
- .owner = THIS_MODULE,
- .open = idedisk_open,
- .release = idedisk_release,
```

```

- .ioctl = ide_disk_ioctl,
- .getgeo = idedisk_getgeo,
- .media_changed = idedisk_media_changed,
- .revalidate_disk = idedisk_revalidate_disk
+const struct ide_disk_ops ide_ata_disk_ops = {
+ .check = ide_disk_check,
+ .get_capacity = ide_disk_get_capacity,
+ .setup = ide_disk_setup,
+ .flush = ide_disk_flush,
+ .init_media = ide_disk_init_media,
+ .set_doorlock = ide_disk_set_doorlock,
+ .do_request = ide_do_rw_disk,
+ .end_request = ide_end_request,
+ .ioctl = ide_disk_ioctl,
};
-
-MODULE_DESCRIPTION("ATA DISK Driver");
-
-static int ide_disk_probe(ide_drive_t *drive)
-{
- struct ide_disk_obj *idkp;
- struct gendisk *g;
-
- /* strstr("foo", "") is non-NULL */
- if (!strstr("ide-disk", drive->driver_req))
- goto failed;
-
- if (drive->media != ide_disk)
- goto failed;
-
- idkp = kzalloc(sizeof(*idkp), GFP_KERNEL);
- if (!idkp)
- goto failed;
-
- g = alloc_disk_node(IDE_DISK_MINORS, hwif_to_node(drive->hwif));
- if (!g)
- goto out_free_idkp;
-
- ide_init_disk(g, drive);
-
- kref_init(&idkp->kref);
-
- idkp->drive = drive;
- idkp->driver = &idedisk_driver;
- idkp->disk = g;
-
- g->private_data = &idkp->driver;
-
- drive->driver_data = idkp;
-
- idedisk_setup(drive);

```

[git pull] IDE updates #3

```
- if ((drive->dev_flags & IDE_DFLAG_LBA) == 0 &&
- (drive->head == 0 || drive->head > 16)) {
- printk(KERN_ERR "%s: INVALID GEOMETRY: %d PHYSICAL HEADS?\n",
- drive->name, drive->head);
- drive->dev_flags &= ~IDE_DFLAG_ATTACH;
- } else
- drive->dev_flags |= IDE_DFLAG_ATTACH;
-
- g->minors = IDE_DISK_MINORS;
- g->driverfs_dev = &drive->gendev;
- g->flags |= GENHD_FL_EXT_DEVT;
- if (drive->dev_flags & IDE_DFLAG_REMOVABLE)
- g->flags = GENHD_FL_REMOVABLE;
- set_capacity(g, ide_disk_capacity(drive));
- g->fops = &idedisk_ops;
- add_disk(g);
- return 0;
-
-out_free_idkp:
- kfree(idkp);
-failed:
- return -ENODEV;
-}
-
-static void __exit idedisk_exit(void)
-{
- driver_unregister(&idedisk_driver.gen_driver);
-}
-
-static int __init idedisk_init(void)
-{
- return driver_register(&idedisk_driver.gen_driver);
-}
-
-MODULE_ALIAS("ide:*m-disk*");
-MODULE_ALIAS("ide-disk");
-module_init(idedisk_init);
-module_exit(idedisk_exit);
-MODULE_LICENSE("GPL");
diff --git a/drivers/ide/ide-disk.h b/drivers/ide/ide-disk.h
index a82fa43..b234b0f 100644
--- a/drivers/ide/ide-disk.h
+++ b/drivers/ide/ide-disk.h
@@ -1,19 +1,11 @@
#ifdef __IDE_DISK_H
#define __IDE_DISK_H

-struct ide_disk_obj {
- ide_drive_t *drive;
- ide_driver_t *driver;
- struct gendisk *disk;

```

[git pull] IDE updates #3

```
- struct kref kref;
- unsigned int openers; /* protected by BKL for now */
-};
-
-#define ide_disk_g(disk) \
- container_of((disk)->private_data, struct ide_disk_obj, driver)
+#include "ide-gd.h"

+#ifdef CONFIG_IDE_GD_ATA
/* ide-disk.c */
- sector_t ide_disk_capacity(ide_drive_t *);
+ extern const struct ide_disk_ops ide_ata_disk_ops;
ide_decl_devset(address);
ide_decl_devset(multcount);
ide_decl_devset(nowerr);
@@ -21,12 +13,17 @@ ide_decl_devset(wcache);
ide_decl_devset(acoustic);

/* ide-disk_ioctl.c */
- int ide_disk_ioctl(struct inode *, struct file *, unsigned int, unsigned long);
+ int ide_disk_ioctl(ide_drive_t *, struct inode *, struct file *, unsigned int,
+ unsigned long);

#ifdef CONFIG_IDE_PROC_FS
/* ide-disk_proc.c */
extern ide_proc_entry_t ide_disk_proc[];
extern const struct ide_proc_devset ide_disk_settings[];
#endif
+#else
+#define ide_disk_proc NULL
+#define ide_disk_settings NULL
+#endif

#endif /* __IDE_DISK_H */
diff --git a/drivers/ide/ide-disk_ioctl.c b/drivers/ide/ide-disk_ioctl.c
index a6cf1a0..a49698b 100644
--- a/drivers/ide/ide-disk_ioctl.c
+++ b/drivers/ide/ide-disk_ioctl.c
@@ -13,12 +13,10 @@ static const struct ide_ioctl_devset ide_disk_ioctl_settings[] = {
{ 0 }
};

- int ide_disk_ioctl(struct inode *inode, struct file *file,
+ int ide_disk_ioctl(ide_drive_t *drive, struct inode *inode, struct file *file,
unsigned int cmd, unsigned long arg)
{
struct block_device *bdev = inode->i_bdev;
- struct ide_disk_obj *idkp = ide_disk_g(bdev->bd_disk);
- ide_drive_t *drive = idkp->drive;
int err;
```

[git pull] IDE updates #3

```
err = ide_setting_ioctl(drive, bdev, cmd, arg, ide_disk_ioctl_settings);
diff --git a/drivers/ide/ide-disk_proc.c b/drivers/ide/ide-disk_proc.c
index 4724976..1146f42 100644
--- a/drivers/ide/ide-disk_proc.c
+++ b/drivers/ide/ide-disk_proc.c
@@ -56,7 +56,7 @@ static int proc_idedisk_read_capacity
ide_drive_t*drive = (ide_drive_t *)data;
int len;

- len = sprintf(page, "%llu\n", (long long)ide_disk_capacity(drive));
+ len = sprintf(page, "%llu\n", (long long)ide_gd_capacity(drive));

PROC_IDE_READ_RETURN(page, start, off, count, eof, len);
}
diff --git a/drivers/ide/ide-dma-sff.c b/drivers/ide/ide-dma-sff.c
index 0903782..cac431f 100644
--- a/drivers/ide/ide-dma-sff.c
+++ b/drivers/ide/ide-dma-sff.c
@@ -130,7 +130,7 @@ int ide_build_dmatable(ide_drive_t *drive, struct request *rq)
xcount = bcount & 0xffff;
if (is_trm290)
xcount = ((xcount >> 2) - 1) << 16;
- if (xcount == 0x0000) {
+ else if (xcount == 0x0000) {
if (count++ >= PRD_ENTRIES)
goto use_pio_instead;
*table++ = cpu_to_le32(0x8000);
diff --git a/drivers/ide/ide-floppy.c b/drivers/ide/ide-floppy.c
index cf0aa25..aeb1ad7 100644
--- a/drivers/ide/ide-floppy.c
+++ b/drivers/ide/ide-floppy.c
@@ -15,12 +15,6 @@
* Documentation/ide/ChangeLog.ide-floppy.1996-2002
*/

-#define DRV_NAME "ide-floppy"
-#define PFX DRV_NAME ": "
-
-#define IDEFLOPPY_VERSION "1.00"
-
-#include <linux/module.h>
#include <linux/types.h>
#include <linux/string.h>
#include <linux/kernel.h>
@@ -49,19 +43,6 @@

#include "ide-floppy.h"

-/* module parameters */
-static unsigned long debug_mask;
-module_param(debug_mask, ulong, 0644);
```

[git pull] IDE updates #3

```
-
-/* define to see debug info */
-#define IDEFLOPPY_DEBUG_LOG 0
-
-#if IDEFLOPPY_DEBUG_LOG
-#define ide_debug_log(lvl, fmt, args...) __ide_debug_log(lvl, fmt, args)
-#else
-#define ide_debug_log(lvl, fmt, args...) do { } while (0)
-#endif
-
-/*
 * After each failed packet command we issue a request sense command and retry
 * the packet command IDEFLOPPY_MAX_PC_RETRIES times.
 @@ -83,43 +64,13 @@ module_param(debug_mask, ulong, 0644);
 /* Error code returned in rq->errors to the higher part of the driver. */
 #define IDEFLOPPY_ERROR_GENERAL 101

-static DEFINE_MUTEX(idefloppy_ref_mutex);
-
-static void idefloppy_cleanup_obj(struct kref *);
-
-static struct ide_floppy_obj *ide_floppy_get(struct gendisk *disk)
-{
- struct ide_floppy_obj *floppy = NULL;
-
- mutex_lock(&idefloppy_ref_mutex);
- floppy = ide_drv_g(disk, ide_floppy_obj);
- if (floppy) {
- if (ide_device_get(floppy->drive))
- floppy = NULL;
- else
- kref_get(&floppy->kref);
- }
- mutex_unlock(&idefloppy_ref_mutex);
- return floppy;
-}
-
-static void ide_floppy_put(struct ide_floppy_obj *floppy)
-{
- ide_drive_t *drive = floppy->drive;
-
- mutex_lock(&idefloppy_ref_mutex);
- kref_put(&floppy->kref, idefloppy_cleanup_obj);
- ide_device_put(drive);
- mutex_unlock(&idefloppy_ref_mutex);
-}
-
-/*
 * Used to finish servicing a request. For read/write requests, we will call
 * ide_end_request to pass to the next buffer.
 */
```

[git pull] IDE updates #3

```
-static int idefloppy_end_request(ide_drive_t *drive, int uptodate, int nsecs)
+static int ide_floppy_end_request(ide_drive_t *drive, int uptodate, int nsecs)
{
- idefloppy_floppy_t *floppy = drive->driver_data;
+ struct ide_disk_obj *floppy = drive->driver_data;
struct request *rq = HWGROUP(drive)->rq;
int error;

@@ -161,12 +112,12 @@ static void idefloppy_update_buffers(ide_drive_t *drive,
struct bio *bio = rq->bio;

while ((bio = rq->bio) != NULL)
- idefloppy_end_request(drive, 1, 0);
+ ide_floppy_end_request(drive, 1, 0);
}

static void ide_floppy_callback(ide_drive_t *drive, int dsc)
{
- idefloppy_floppy_t *floppy = drive->driver_data;
+ struct ide_disk_obj *floppy = drive->driver_data;
struct ide_atapi_pc *pc = drive->pc;
int uptodate = pc->error ? 0 : 1;

@@ -200,10 +151,10 @@ static void ide_floppy_callback(ide_drive_t *drive, int dsc)
"Aborting request!\n");
}

- idefloppy_end_request(drive, uptodate, 0);
+ ide_floppy_end_request(drive, uptodate, 0);
}

-static void ide_floppy_report_error(idefloppy_floppy_t *floppy,
+static void ide_floppy_report_error(struct ide_disk_obj *floppy,
struct ide_atapi_pc *pc)
{
/* suppress error messages resulting from Medium not present */
@@ -222,7 +173,7 @@ static void ide_floppy_report_error(idefloppy_floppy_t *floppy,
static ide_startstop_t idefloppy_issue_pc(ide_drive_t *drive,
struct ide_atapi_pc *pc)
{
- idefloppy_floppy_t *floppy = drive->driver_data;
+ struct ide_disk_obj *floppy = drive->driver_data;

if (floppy->failed_pc == NULL &&
pc->c[0] != GPCMD_REQUEST_SENSE)
@@ -286,7 +237,7 @@ static void idefloppy_create_rw_cmd(ide_drive_t *drive,
struct ide_atapi_pc *pc, struct request *rq,
unsigned long sector)
{
- idefloppy_floppy_t *floppy = drive->driver_data;
+ struct ide_disk_obj *floppy = drive->driver_data;
```

[git pull] IDE updates #3

```
int block = sector / floppy->bs_factor;
int blocks = rq->nr_sectors / floppy->bs_factor;
int cmd = rq_data_dir(rq);
@@ -310,7 +261,7 @@ static void idefloppy_create_rw_cmd(ide_drive_t *drive,
pc->flags |= PC_FLAG_DMA_OK;
}

-static void idefloppy_blockpc_cmd(idefloppy_floppy_t *floppy,
+static void idefloppy_blockpc_cmd(struct ide_disk_obj *floppy,
struct ide_atapi_pc *pc, struct request *rq)
{
ide_init_pc(pc);
@@ -329,13 +280,12 @@ static void idefloppy_blockpc_cmd(idefloppy_floppy_t *floppy,
pc->req_xfer = pc->buf_size = rq->data_len;
}

-static ide_startstop_t idefloppy_do_request(ide_drive_t *drive,
- struct request *rq, sector_t block_s)
+static ide_startstop_t ide_floppy_do_request(ide_drive_t *drive,
+ struct request *rq, sector_t block)
{
- idefloppy_floppy_t *floppy = drive->driver_data;
+ struct ide_disk_obj *floppy = drive->driver_data;
ide_hwif_t *hwif = drive->hwif;
struct ide_atapi_pc *pc;
- unsigned long block = (unsigned long)block_s;

ide_debug_log(IDE_DBG_FUNC, "%s: dev: %s, cmd: 0x%x, cmd_type: %x, "
"errors: %d\n",
@@ -353,7 +303,7 @@ static ide_startstop_t idefloppy_do_request(ide_drive_t *drive,
else
printk(KERN_ERR PFX "%s: I/O error\n", drive->name);

- idefloppy_end_request(drive, 0, 0);
+ ide_floppy_end_request(drive, 0, 0);
return ide_stopped;
}
if (blk_fs_request(rq)) {
@@ -361,11 +311,11 @@ static ide_startstop_t idefloppy_do_request(ide_drive_t *drive,
(rq->nr_sectors % floppy->bs_factor)) {
printk(KERN_ERR PFX "%s: unsupported r/w rq size\n",
drive->name);
- idefloppy_end_request(drive, 0, 0);
+ ide_floppy_end_request(drive, 0, 0);
return ide_stopped;
}
pc = &floppy->queued_pc;
- idefloppy_create_rw_cmd(drive, pc, rq, block);
+ idefloppy_create_rw_cmd(drive, pc, rq, (unsigned long)block);
} else if (blk_special_request(rq)) {
pc = (struct ide_atapi_pc *) rq->buffer;
```

[git pull] IDE updates #3

```
} else if (blk_pc_request(rq)) {
@@ -373,7 +323,7 @@ static ide_startstop_t idefloppy_do_request(ide_drive_t *drive,
idefloppy_blockpc_cmd(floppy, pc, rq);
} else {
blk_dump_rq_flags(rq, PFX "unsupported command in queue");
- idefloppy_end_request(drive, 0, 0);
+ ide_floppy_end_request(drive, 0, 0);
return ide_stopped;
}

@@ -394,7 +344,7 @@ static ide_startstop_t idefloppy_do_request(ide_drive_t *drive,
*/
static int ide_floppy_get_flexible_disk_page(ide_drive_t *drive)
{
- idefloppy_floppy_t *floppy = drive->driver_data;
+ struct ide_disk_obj *floppy = drive->driver_data;
struct gendisk *disk = floppy->disk;
struct ide_atapi_pc pc;
u8 *page;
@@ -410,11 +360,11 @@ static int ide_floppy_get_flexible_disk_page(ide_drive_t *drive)
}

if (pc.buf[3] & 0x80)
- drive->atapi_flags |= IDE_AFLAG_WP;
+ drive->dev_flags |= IDE_DFLAG_WP;
else
- drive->atapi_flags &= ~IDE_AFLAG_WP;
+ drive->dev_flags &= ~IDE_DFLAG_WP;

- set_disk_ro(disk, !(drive->atapi_flags & IDE_AFLAG_WP));
+ set_disk_ro(disk, !(drive->dev_flags & IDE_DFLAG_WP));

page = &pc.buf[8];

@@ -445,7 +395,9 @@ static int ide_floppy_get_flexible_disk_page(ide_drive_t *drive)
drive->name, lba_capacity, capacity);
floppy->blocks = floppy->block_size ?
capacity / floppy->block_size : 0;
+ drive->capacity64 = floppy->blocks * floppy->bs_factor;
}
+
return 0;
}

@@ -455,7 +407,7 @@ static int ide_floppy_get_flexible_disk_page(ide_drive_t *drive)
*/
static int ide_floppy_get_capacity(ide_drive_t *drive)
{
- idefloppy_floppy_t *floppy = drive->driver_data;
+ struct ide_disk_obj *floppy = drive->driver_data;
struct gendisk *disk = floppy->disk;
```

[git pull] IDE updates #3

```
struct ide_atapi_pc pc;
u8 *cap_desc;
@@ -466,7 +418,7 @@ static int ide_floppy_get_capacity(ide_drive_t *drive)
drive->bios_head = drive->bios_sect = 0;
floppy->blocks = 0;
floppy->bs_factor = 1;
- set_capacity(floppy->disk, 0);
+ drive->capacity64 = 0;

ide_floppy_create_read_capacity_cmd(&pc);
if (ide_queue_pc_tail(drive, disk, &pc)) {
@@ -523,6 +475,8 @@ static int ide_floppy_get_capacity(ide_drive_t *drive)
"non 512 bytes block size not "
"fully supported\n",
drive->name);
+ drive->capacity64 =
+ floppy->blocks * floppy->bs_factor;
rc = 0;
}
break;
@@ -547,21 +501,12 @@ static int ide_floppy_get_capacity(ide_drive_t *drive)
if (!(drive->atapi_flags & IDE_AFLAG_CLIK_DRIVE))
(void) ide_floppy_get_flexible_disk_page(drive);

- set_capacity(disk, floppy->blocks * floppy->bs_factor);
-
return rc;
}

-sector_t ide_floppy_capacity(ide_drive_t *drive)
-{
- idefloppy_floppy_t *floppy = drive->driver_data;
- unsigned long capacity = floppy->blocks * floppy->bs_factor;
-
- return capacity;
-}

-static void idefloppy_setup(ide_drive_t *drive, idefloppy_floppy_t *floppy)
+static void ide_floppy_setup(ide_drive_t *drive)
{
+ struct ide_disk_obj *floppy = drive->driver_data;
u16 *id = drive->id;

drive->pc_callback = ide_floppy_callback;
@@ -592,252 +537,42 @@ static void idefloppy_setup(ide_drive_t *drive, idefloppy_floppy_t *floppy)
blk_queue_max_sectors(drive->queue, 64);
drive->atapi_flags |= IDE_AFLAG_CLIK_DRIVE;
/* IOMEGA Clk! drives do not support lock/unlock commands */
- drive->atapi_flags |= IDE_AFLAG_NO_DOORLOCK;
+ drive->dev_flags &= ~IDE_DFLAG_DOORLOCKING;
}
}
```

```

(void) ide_floppy_get_capacity(drive);

ide_proc_register_driver(drive, floppy->driver);
-}

-static void ide_floppy_remove(ide_drive_t *drive)
-{
- idefloppy_floppy_t *floppy = drive->driver_data;
- struct gendisk *g = floppy->disk;
-
- ide_proc_unregister_driver(drive, floppy->driver);
-
- del_gendisk(g);
-
- ide_floppy_put(floppy);
+ drive->dev_flags |= IDE_DFLAG_ATTACH;
}

-static void idefloppy_cleanup_obj(struct kref *kref)
+static void ide_floppy_flush(ide_drive_t *drive)
{
- struct ide_floppy_obj *floppy = to_ide_drv(kref, ide_floppy_obj);
- ide_drive_t *drive = floppy->drive;
- struct gendisk *g = floppy->disk;
-
- drive->driver_data = NULL;
- g->private_data = NULL;
- put_disk(g);
- kfree(floppy);
}

-static int ide_floppy_probe(ide_drive_t *);
-
-static ide_driver_t idefloppy_driver = {
- .gen_driver = {
- .owner = THIS_MODULE,
- .name = "ide-floppy",
- .bus = &ide_bus_type,
- },
- .probe = ide_floppy_probe,
- .remove = ide_floppy_remove,
- .version = IDEFLOPPY_VERSION,
- .do_request = idefloppy_do_request,
- .end_request = idefloppy_end_request,
- .error = __ide_error,
-#ifdef CONFIG_IDE_PROC_FS
- .proc = ide_floppy_proc,
- .settings = ide_floppy_settings,
-#endif
-};

```

[git pull] IDE updates #3

```
-
-static int idefloppy_open(struct inode *inode, struct file *filp)
+static int ide_floppy_init_media(ide_drive_t *drive, struct gendisk *disk)
{
- struct gendisk *disk = inode->i_bdev->bd_disk;
- struct ide_floppy_obj *floppy;
- ide_drive_t *drive;
int ret = 0;

- floppy = ide_floppy_get(disk);
- if (!floppy)
- return -ENXIO;
-
- drive = floppy->drive;
-
- ide_debug_log(IDE_DBG_FUNC, "Call %s\n", __func__);
-
- floppy->openers++;
-
- if (floppy->openers == 1) {
- drive->atapi_flags &= ~IDE_AFLAG_FORMAT_IN_PROGRESS;
- /* Just in case */
-
- if (ide_do_test_unit_ready(drive, disk))
- ide_do_start_stop(drive, disk, 1);
-
- if (ide_floppy_get_capacity(drive)
- && (filp->f_flags & O_NDELAY) == 0
- /*
- * Allow O_NDELAY to open a drive without a disk, or with an
- * unreadable disk, so that we can get the format capacity
- * of the drive or begin the format - Sam
- */
- ) {
- ret = -EIO;
- goto out_put_floppy;
- }
-
- if ((drive->atapi_flags & IDE_AFLAG_WP) && (filp->f_mode & 2)) {
- ret = -EROFS;
- goto out_put_floppy;
- }
-
- drive->atapi_flags |= IDE_AFLAG_MEDIA_CHANGED;
- ide_set_media_lock(drive, disk, 1);
- check_disk_change(inode->i_bdev);
- } else if (drive->atapi_flags & IDE_AFLAG_FORMAT_IN_PROGRESS) {
- ret = -EBUSY;
- goto out_put_floppy;
- }
- return 0;

```

```

-
-out_put_floppy:
- floppy->openers--;
- ide_floppy_put(floppy);
- return ret;
-}
-
-static int idefloppy_release(struct inode *inode, struct file *filp)
-{
- struct gendisk *disk = inode->i_bdev->bd_disk;
- struct ide_floppy_obj *floppy = ide_drv_g(disk, ide_floppy_obj);
- ide_drive_t *drive = floppy->drive;
-
- ide_debug_log(IDE_DBG_FUNC, "Call %s\n", __func__);
-
- if (floppy->openers == 1) {
- ide_set_media_lock(drive, disk, 0);
- drive->atapi_flags &= ~IDE_AFLAG_FORMAT_IN_PROGRESS;
- }
-
- floppy->openers--;
-
- ide_floppy_put(floppy);
-
- return 0;
-}
-
-static int idefloppy_getgeo(struct block_device *bdev, struct hd_geometry *geo)
-{
- struct ide_floppy_obj *floppy = ide_drv_g(bdev->bd_disk,
- ide_floppy_obj);
- ide_drive_t *drive = floppy->drive;
+ if (ide_do_test_unit_ready(drive, disk))
+ ide_do_start_stop(drive, disk, 1);

- geo->heads = drive->bios_head;
- geo->sectors = drive->bios_sect;
- geo->cylinders = (u16)drive->bios_cyl; /* truncate */
- return 0;
-}
+ ret = ide_floppy_get_capacity(drive);

-static int idefloppy_media_changed(struct gendisk *disk)
-{
- struct ide_floppy_obj *floppy = ide_drv_g(disk, ide_floppy_obj);
- ide_drive_t *drive = floppy->drive;
- int ret;
+ set_capacity(disk, ide_gd_capacity(drive));

- /* do not scan partitions twice if this is a removable device */
- if (drive->dev_flags & IDE_DFLAG_ATTACH) {

```

[git pull] IDE updates #3

```
- drive->dev_flags &= ~IDE_DFLAG_ATTACH;
- return 0;
- }
- ret = !(drive->atapi_flags & IDE_AFLAG_MEDIA_CHANGED);
- drive->atapi_flags &= ~IDE_AFLAG_MEDIA_CHANGED;
return ret;
}

-static int idefloppy_revalidate_disk(struct gendisk *disk)
-{
- struct ide_floppy_obj *floppy = ide_drv_g(disk, ide_floppy_obj);
- set_capacity(disk, ide_floppy_capacity(floppy->drive));
- return 0;
-}
-
-static struct block_device_operations idefloppy_ops = {
- .owner = THIS_MODULE,
- .open = idefloppy_open,
- .release = idefloppy_release,
- .ioctl = ide_floppy_ioctl,
- .getgeo = idefloppy_getgeo,
- .media_changed = idefloppy_media_changed,
- .revalidate_disk = idefloppy_revalidate_disk
+const struct ide_disk_ops ide_atapi_disk_ops = {
+ .check = ide_check_atapi_device,
+ .get_capacity = ide_floppy_get_capacity,
+ .setup = ide_floppy_setup,
+ .flush = ide_floppy_flush,
+ .init_media = ide_floppy_init_media,
+ .set_doorlock = ide_set_media_lock,
+ .do_request = ide_floppy_do_request,
+ .end_request = ide_floppy_end_request,
+ .ioctl = ide_floppy_ioctl,
};
-
-static int ide_floppy_probe(ide_drive_t *drive)
-{
- idefloppy_floppy_t *floppy;
- struct gendisk *g;
-
- if (!strstr("ide-floppy", drive->driver_req))
- goto failed;
-
- if (drive->media != ide_floppy)
- goto failed;
-
- if (!ide_check_atapi_device(drive, DRV_NAME)) {
- printk(KERN_ERR PFX "%s: not supported by this version of "
- DRV_NAME "\n", drive->name);
- goto failed;
- }
}
```

[git pull] IDE updates #3

```
- floppy = kzalloc(sizeof(idefloppy_floppy_t), GFP_KERNEL);
- if (!floppy) {
- printk(KERN_ERR PFX "%s: Can't allocate a floppy structure\n",
- drive->name);
- goto failed;
- }
-
- g = alloc_disk(1 << PARTN_BITS);
- if (!g)
- goto out_free_floppy;
-
- ide_init_disk(g, drive);
-
- kref_init(&floppy->kref);
-
- floppy->drive = drive;
- floppy->driver = &idefloppy_driver;
- floppy->disk = g;
-
- g->private_data = &floppy->driver;
-
- drive->driver_data = floppy;
-
- drive->debug_mask = debug_mask;
-
- idefloppy_setup(drive, floppy);
- drive->dev_flags |= IDE_DFLAG_ATTACH;
-
- g->minors = 1 << PARTN_BITS;
- g->driverfs_dev = &drive->gendev;
- if (drive->dev_flags & IDE_DFLAG_REMOVABLE)
- g->flags = GENHD_FL_REMOVABLE;
- g->fops = &idefloppy_ops;
- add_disk(g);
- return 0;
-
-out_free_floppy:
- kfree(floppy);
-failed:
- return -ENODEV;
-}
-
-static void __exit idefloppy_exit(void)
-{
- driver_unregister(&idefloppy_driver.gen_driver);
-}
-
-static int __init idefloppy_init(void)
-{
- printk(KERN_INFO DRV_NAME " driver " IDEFLOPPY_VERSION "\n");
- return driver_register(&idefloppy_driver.gen_driver);
-}
```

```

-}
-
-MODULE_ALIAS("ide:*m-floppy*");
-MODULE_ALIAS("ide-floppy");
-module_init(idfloppy_init);
-module_exit(idfloppy_exit);
-MODULE_LICENSE("GPL");
-MODULE_DESCRIPTION("ATAPI FLOPPY Driver");
-
diff --git a/drivers/ide/ide-floppy.h b/drivers/ide/ide-floppy.h
index 17cf865..c17124d 100644
--- a/drivers/ide/ide-floppy.h
+++ b/drivers/ide/ide-floppy.h
@@ -1,37 +1,9 @@
#ifdef __IDE_FLOPPY_H
#define __IDE_FLOPPY_H

_/*
- * 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
- * idfloppy_floppy_t, defined below.
- */
-typedef struct ide_floppy_obj {
- ide_drive_t *drive;
- ide_driver_t *driver;
- struct gendisk *disk;
- struct kref kref;
- unsigned int openers; /* protected by BKL for now */
-
- /* Last failed packet command */
- struct ide_atapi_pc *failed_pc;
- /* used for blk_{fs,pc}_request() requests */
- struct ide_atapi_pc queued_pc;
-
- /* Last error information */
- u8 sense_key, asc, ascq;
-
- int progress_indication;
-
- /* Device information */
- /* Current format */
- int blocks, block_size, bs_factor;
- /* Last format capacity descriptor */
- u8 cap_desc[8];
- /* Copy of the flexible disk page */
- u8 flexible_disk_page[32];
-} idfloppy_floppy_t;
#include "ide-gd.h"

#ifdef CONFIG_IDE_GD_ATAPI
_/*

```

[git pull] IDE updates #3

* Pages of the SELECT SENSE / MODE SENSE packet commands.

* See SFF-8070i spec.

```
@@ -46,17 +18,22 @@ typedef struct ide_floppy_obj {  
#define IDEFLOPPY_IOCTL_FORMAT_GET_PROGRESS 0x4603
```

```
/* ide-floppy.c */
```

```
+extern const struct ide_disk_ops ide_atapi_disk_ops;  
void ide_floppy_create_mode_sense_cmd(struct ide_atapi_pc *, u8);  
void ide_floppy_create_read_capacity_cmd(struct ide_atapi_pc *);  
-sector_t ide_floppy_capacity(ide_drive_t *);
```

```
/* ide-floppy_ioctl.c */
```

```
-int ide_floppy_ioctl(struct inode *, struct file *, unsigned, unsigned long);  
+int ide_floppy_ioctl(ide_drive_t *, struct inode *, struct file *, unsigned int,  
+ unsigned long);
```

```
#ifdef CONFIG_IDE_PROC_FS
```

```
/* ide-floppy_proc.c */
```

```
extern ide_proc_entry_t ide_floppy_proc[];  
extern const struct ide_proc_devset ide_floppy_settings[];  
#endif -  
- pci_read_config_byte(dev, 0x5a, &scr1);  
- if (((scr1 & 0x10) >> 4) != mask) {  
- if (mask)  
- scr1 |= 0x10;  
- else  
- scr1 &= ~0x10;  
- pci_write_config_byte(dev, 0x5a, scr1);  
- }  
- } else {  
+ if (drive->quirk_list == 0)  
+ return;  
+  
+ if (info->chip_type >= HPT370) {  
+ u8 scr1 = 0;  
+  
+ pci_read_config_byte(dev, 0x5a, &scr1);  
+ if (((scr1 & 0x10) >> 4) != mask) {  
if (mask)  
- disable_irq(hwif->irq);  
+ scr1 |= 0x10;  
else  
- enable_irq (hwif->irq);  
+ scr1 &= ~0x10;  
+ pci_write_config_byte(dev, 0x5a, scr1);  
}  
- } else  
- outb(ATA_DEVCTL_OBS | (mask ? 2 : 0), hwif->io_ports.ctl_addr);  
+ } else if (mask)  
+ disable_irq(hwif->irq);  
+ else
```

[git pull] IDE updates #3

```
+ enable_irq(hwif->irq);
}

/*
@@ -1289,7 +1287,6 @@ static u8 hpt3xx_cable_detect(ide_hwif_t *hwif)

static void __devinit init_hwif_hpt366(ide_hwif_t *hwif)
{
- struct pci_dev *dev = to_pci_dev(hwif->dev);
struct hpt_info *info = hpt3xx_get_info(hwif->dev);
int serialize = HPT_SERIALIZE_IO;
u8 chip_type = info->chip_type;
diff --git a/drivers/ide/pci/scc_pata.c b/drivers/ide/pci/scc_pata.c
index 9ce1d80..49f163a 100644
--- a/drivers/ide/pci/scc_pata.c
+++ b/drivers/ide/pci/scc_pata.c
@@ -617,7 +617,6 @@ static int __devinit init_setup_scc(struct pci_dev *dev,
unsigned long intmask_port;
unsigned long mode_port;
unsigned long ecmode_port;
- unsigned long dma_status_port;
u32 reg = 0;
struct scc_ports *ports;
int rc;
@@ -637,7 +636,6 @@ static int __devinit init_setup_scc(struct pci_dev *dev,
intmask_port = dma_base + 0x010;
mode_port = ctl_base + 0x024;
ecmode_port = ctl_base + 0xf00;
- dma_status_port = dma_base + 0x004;

/* controller initialization */
reg = 0;
@@ -843,8 +841,6 @@ static u8 scc_cable_detect(ide_hwif_t *hwif)

static void __devinit init_hwif_scc(ide_hwif_t *hwif)
{
- struct scc_ports *ports = ide_get_hwifdata(hwif);
-
/* PTERADD */
out_be32((void __iomem *) (hwif->dma_base + 0x018), hwif->dmatable_dma);

diff --git a/drivers/ide/pci/sgiioc4.c b/drivers/ide/pci/sgiioc4.c
index dd63454..8af9b23 100644
--- a/drivers/ide/pci/sgiioc4.c
+++ b/drivers/ide/pci/sgiioc4.c
@@ -101,18 +101,8 @@ sgioc4_init_hwif_ports(hw_regs_t * hw, unsigned long data_port,
for (i = 0; i <= 7; i++)
hw->io_ports_array[i] = reg + i * 4;

- if (ctrl_port)
- hw->io_ports.ctrl_addr = ctrl_port;
```

```

-
- if (irq_port)
- hw->io_ports.irq_addr = irq_port;
-}
-
-static void
-sgioc4_maskproc(ide_drive_t * drive, int mask)
-{
- writeb(ATA_DEVCTL_OBS | (mask ? 2 : 0),
- (void __iomem *)drive->hwif->io_ports.ctl_addr);
+ hw->io_ports.ctl_addr = ctrl_port;
+ hw->io_ports.irq_addr = irq_port;
}

static int
@@ -310,16 +300,14 @@ static u8 sgioc4_read_status(ide_hwif_t *hwif)
unsigned long port = hwif->io_ports.status_addr;
u8 reg = (u8) readb((void __iomem *) port);

- if ((port & 0xFFF) == 0x11C) { /* Status register of IOC4 */
- if (!(reg & ATA_BUSY)) { /* Not busy... check for interrupt */
- unsigned long other_ir = port - 0x110;
- unsigned int intr_reg = (u32) readl((void __iomem *) other_ir);
+ if (!(reg & ATA_BUSY)) { /* Not busy... check for interrupt */
+ unsigned long other_ir = port - 0x110;
+ unsigned int intr_reg = (u32) readl((void __iomem *) other_ir);

- /* Clear the Interrupt, Error bits on the IOC4 */
- if (intr_reg & 0x03) {
- writel(0x03, (void __iomem *) other_ir);
- intr_reg = (u32) readl((void __iomem *) other_ir);
- }
+ /* Clear the Interrupt, Error bits on the IOC4 */
+

```