

[git pull] IDE updates #1

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2008-10/msg04044.html>

- *From:* Bartłomiej Zolnierkiewicz <bzolnier@xxxxxxxxx>
 - *Date:* Fri, 10 Oct 2008 23:19:48 +0200
-

Inside:

- * ide-generic host driver fix to not claim by default ports for which there is a better driver available. (Borislav Petkov)
- * Proper PCI Power Management support.
- * More work on generic ATAPI support. (Borislav Petkov & me)
- * Use/add generic helpers for device identity handling.
- * Re-work of IDE settings infrastructure.
- * Use special driver requests for handling IDE settings. (Elias Oltmanns)
- * Misc fixes/improvements. (Mark de Wever, FUJITA Tomonori, Linus Torvalds, me)

Thanks to all contributors!

Also thanks to all people reviewing patches and testing linux-next (+ reporting bugs) since without their help it wouldn't be possible to keep the usual high-quality of IDE updates.

Linus, please pull from:

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

to receive the following updates:

```
drivers/ata/libata-scsi.c | 2 +-  
drivers/block/hd.c | 9 +-  
drivers/ide/Kconfig | 23 --  
drivers/ide/Makefile | 7 +-  
drivers/ide/arm/icside.c | 5 +-  
drivers/ide/arm/palm_bk3710.c | 8 +-  
...
```

[git pull] IDE updates #1

drivers/ide/ide-acpi.c | 6 +-
drivers/ide/ide-atapi.c | 236 ++++++-----
drivers/ide/ide-cd.c | 72 ++----
drivers/ide/ide-disk.c | 340 ++++++-----
drivers/ide/ide-dma.c | 52 +---
drivers/ide/ide-floppy.c | 668 ++++++-----
drivers/ide/ide-floppy.h | 63 ++++
drivers/ide/ide-floppy_ioctl.c | 243 ++++++-----
drivers/ide/ide-generic.c | 55 +++-
drivers/ide/ide-io.c | 117 +++++--
drivers/ide/ide-ioctls.c | 290 ++++++-----
drivers/ide/ide-iops.c | 225 ++++-----
drivers/ide/ide-lib.c | 73 +++--
drivers/ide/ide-probe.c | 252 ++++++-----
drivers/ide/ide-proc.c | 306 ++++++-----
drivers/ide/ide-tape.c | 478 ++++++-----
drivers/ide/ide-taskfile.c | 156 +------
drivers/ide/ide-timings.c | 22 +-
drivers/ide/ide.c | 238 +------
drivers/ide/legacy/ali14xx.c | 1 -
drivers/ide/legacy/buddha.c | 1 -
drivers/ide/legacy/dtc2278.c | 1 -
drivers/ide/legacy/falconide.c | 1 -
drivers/ide/legacy/gayle.c | 1 -
drivers/ide/legacy/ht6560b.c | 1 -
drivers/ide/legacy/ide-cs.c | 1 -
drivers/ide/legacy/macide.c | 1 -
drivers/ide/legacy/q40ide.c | 2 -
drivers/ide/legacy/qd65xx.c | 23 +-
drivers/ide/legacy/umc8672.c | 1 -
drivers/ide/pci/aec62xx.c | 5 +-
drivers/ide/pci/alim15x3.c | 9 +-
drivers/ide/pci/amd74xx.c | 8 +-
drivers/ide/pci/atiixp.c | 3 +-
drivers/ide/pci/cmd640.c | 43 +---
drivers/ide/pci/cmd64x.c | 5 +-
drivers/ide/pci/cs5520.c | 3 +-
drivers/ide/pci/cs5530.c | 19 +-
drivers/ide/pci/cs5535.c | 14 +-
drivers/ide/pci/cy82c693.c | 4 +-
drivers/ide/pci/delkin_cb.c | 1 -
drivers/ide/pci/generic.c | 3 +-
drivers/ide/pci/hpt34x.c | 5 +-
drivers/ide/pci/hpt366.c | 77 +++--
drivers/ide/pci/it8213.c | 3 +-
drivers/ide/pci/it821x.c | 58 +---
drivers/ide/pci/jmicron.c | 3 +-
drivers/ide/pci/ns87415.c | 9 +-
drivers/ide/pci/opti621.c | 7 +-
drivers/ide/pci/pdc202xx_new.c | 15 +-
drivers/ide/pci/pdc202xx_old.c | 11 +-

[git pull] IDE updates #1

```
drivers/ide/pci/piix.c | 5 +-
drivers/ide/pci/rz1000.c | 1 -
drivers/ide/pci/sc1200.c | 15 +-
drivers/ide/pci/scc_pata.c | 5 +-
drivers/ide/pci/serverworks.c | 9 +-
drivers/ide/pci/sgiioc4.c | 1 -
drivers/ide/pci/siimage.c | 17 +-
drivers/ide/pci/sis5513.c | 5 +-
drivers/ide/pci/sl82c105.c | 7 +-
drivers/ide/pci/slc90e66.c | 3 +-
drivers/ide/pci/triflex.c | 3 +-
drivers/ide/pci/trm290.c | 1 -
drivers/ide/pci/via82cxxx.c | 10 +-
drivers/ide/ppc/pmac.c | 6 +-
drivers/ide/setup-pci.c | 33 ++
drivers/scsi/ide-scsi.c | 120 +++-----
include/linux/ata.h | 112 ++++++--
include/linux/ide.h | 316 ++++++++-----
75 files changed, 2388 insertions(+), 2566 deletions(-)
create mode 100644 drivers/ide/ide-floppy.h
create mode 100644 drivers/ide/ide-floppy_ioctl.c
create mode 100644 drivers/ide/ide-ioctls.c
```

Bartłomiej Zolnierkiewicz (65):

- ide: remove superfluous check from ide_disk_special()
- ide: cleanup ide_disk_init_mult_count()
- ide: cleanup ide_fix_driveid()
- ide: make drive->id an union (take 2)
- ide: remove drive->driveid
- ide: use ata_id_current_chs_valid()
- ide-disk: use ata_id_wcache_enabled()
- ide-disk: use ata_id_hpa_enabled()
- libata: WIN_* -> ATA_CMD_*
- ide: WIN_* -> ATA_CMD_*
- hd: WIN_* -> ATA_CMD_*
- ide: use ATA_* defines instead of *_STAT and *_ERR ones
- ide: remove no longer needed ide_drive_t fields
- ide: fix EXABYTENEST handling in probe_for_drive()
- ide: enhance ide_busy_sleep()
- ide: remove no longer needed BUG_ON()-s from init_irq()
- ide: remove IDE_CHIPSET_* macros
- ide: remove unused _IDE_C and _IDE_DISK defines
- ide: remove needless drive->present checks from device drivers
- ide: check drive->present in ide_get_paired_drive()
- ide: remove CONFIG_IDEDISK_MULTI_MODE
- ide: include <linux/hdreg.h> only when needed
- ide: call ide_proc_register_driver() later
- ide: preparations for /proc/ide/hd*/settings rework
- ide: /proc/ide/hd*/settings rework
- ide: remove SECTOR_WORDS define

[git pull] IDE updates #1

cmd640: add __set_prefetch_mode()
ide: remove read-only ->atapi_overlap field from ide_drive_t
ide: remove ->supports_dsc_overlap field from ide_driver_t
ide: factor out HDIO_*_NICE ioctl handling to ide_*_nice_ioctl()
ide: ide_dev_has_iordy() -> ata_id_has_iordy()
ide: ide_dev_is_sata() -> ata_id_is_sata()
ide: idedisk_supports_lba48() -> ata_id_lba48_enabled()
ide: check only for CACHE FLUSH command support in ide_id_has_flush_cache()
ide: ide_id_has_flush_cache() -> ata_id_flush_enabled()
ide: ide_id_has_flush_cache_ext() -> ata_id_flush_ext_enabled()
ide: use ata_id_is_cfa()
ide: ide_id_to_hd_driveid() -> ata_id_to_hd_driveid()
ide: lba_capacity_is_ok() -> ata_id_is_lba_capacity_ok()
hpt366: add hpt3xx_disable_fast_irq() helper
ide: add proper PCI PM support (v2)
ide: remove ->bus_state field from ide_hwif_t
ide: add ide_setting_ioctl() helper
ide: cleanup generic_ide_ioctl()
ide: move ioctls handling to ide-ioctls.c
ide: add ide_check_atapi_device() helper
ide-floppy: remove needless parens
ide-floppy: add ide_floppy_format_ioctl() helper
ide-tape: remove idetape_init_rq()
ide-{floppy,tape}: remove request stack
ide-{floppy,tape}: remove packet command stack
ide-floppy: remove MODE_SENSE_* defines
ide-scsi: cleanup ide_scsi_io_buffers()
ide: add ide_io_buffers() helper
ide-floppy: add ide_floppy_set_media_lock() helper
ide-tape: add ide_tape_set_media_lock() helper
ide: add ide_init_pc() helper
ide: add ide_queue_pc_head() helper
ide: add ide_queue_pc_tail() helper
ide-floppy: ->{srfp,wp} -> IDE_AFLAG_{SRFP,WP}
ide-floppy: move floppy ioctls handling to ide-floppy_ioctl.c
ide: add ide_set_media_lock() helper
ide: add ide_do_start_stop() helper
ide: add ide_do_test_unit_ready() helper
ide: move IDE{FLOPPY,TAPE}_WAIT_CMD defines to <linux/ide.h>

Borislav Petkov (2):

ide-floppy: use scatterlists for pio transfers
ide-generic: handle probing of legacy io-ports v5

Elias Oltmanns (1):

ide: Remove ide_spin_wait_hwgroup() and use special requests instead

FUJITA Tomonori (1):

ide-cd: use the new object_is_in_stack() helper

Linus Torvalds (1):

[git pull] IDE updates #1

[git pull] IDE updates #1

ide: re-code ide_fixstring() loop to be less evil

Mark de Wever (1):

ide-tape: Buildfix when IDETAPE_DEBUG_LOG is set to 1.

```
diff --git a/drivers/ata/libata-scsi.c b/drivers/ata/libata-scsi.c
index 59fe051..5d312dc 100644
--- a/drivers/ata/libata-scsi.c
+++ b/drivers/ata/libata-scsi.c
@@ -503,7 +503,7 @@ int ata_cmd_ioctl(struct scsi_device *scsidev, void __user *arg)
scsi_cmd[0] = ATA_16;

scsi_cmd[4] = args[2];
- if (args[0] == WIN_SMART) { /* hack -- ide driver does this too... */
+ if (args[0] == ATA_CMD_SMART) { /* hack -- ide driver does this too */
scsi_cmd[6] = args[3];
scsi_cmd[8] = args[1];
scsi_cmd[10] = 0x4f;
diff --git a/drivers/block/hd.c b/drivers/block/hd.c
index 682243b..482c0c4 100644
--- a/drivers/block/hd.c
+++ b/drivers/block/hd.c
@@ -39,6 +39,7 @@
#include <linux/ioport.h>
#include <linux/init.h>
#include <linux/blkpg.h>
+#include <linux/ata.h>
#include <linux/hdreg.h>

#define REALLY_SLOW_IO
@@ -370,7 +371,7 @@ repeat:
struct hd_i_struct *disk = &hd_info[i];
disk->special_op = disk->recalibrate = 1;
hd_out(disk, disk->sect, disk->sect, disk->head-1,
- disk->cyl, WIN_SPECIFY, &reset_hd);
+ disk->cyl, ATA_CMD_INIT_DEV_PARAMS, &reset_hd);
if (reset)
goto repeat;
} else
@@ -558,7 +559,7 @@ static int do_special_op(struct hd_i_struct *disk, struct request *req)
{
if (disk->recalibrate) {
disk->recalibrate = 0;
- hd_out(disk, disk->sect, 0, 0, 0, WIN_RESTORE, &recal_intr);
+ hd_out(disk, disk->sect, 0, 0, 0, ATA_CMD_RESTORE, &recal_intr);
return reset;
}
if (disk->head > 16) {
@@ -631,13 +632,13 @@ repeat:
if (blk_fs_request(req)) {
```

[git pull] IDE updates #1

[git pull] IDE updates #1

```
switch (rq_data_dir(req)) {
case READ:
- hd_out(disk, nsect, sec, head, cyl, WIN_READ,
+ hd_out(disk, nsect, sec, head, cyl, ATA_CMD_PIO_READ,
&read_intr);
if (reset)
goto repeat;
break;
case WRITE:
- hd_out(disk, nsect, sec, head, cyl, WIN_WRITE,
+ hd_out(disk, nsect, sec, head, cyl, ATA_CMD_PIO_WRITE,
&write_intr);
if (reset)
goto repeat;
diff --git a/drivers/ide/Kconfig b/drivers/ide/Kconfig
index 052879a..b50b5da 100644
--- a/drivers/ide/Kconfig
+++ b/drivers/ide/Kconfig
@@ -131,29 +131,6 @@ config BLK_DEV_IDEDISK
```

If unsure, say Y.

```
-config IDEDISK_MULTI_MODE
- bool "Use multiple sector mode for Programmed Input/Output by default"
- help
- This setting is irrelevant for most IDE disks, with direct memory
- access, to which multiple sector mode does not apply. Multiple sector
- mode is a feature of most modern IDE hard drives, permitting the
- transfer of multiple sectors per Programmed Input/Output interrupt,
- rather than the usual one sector per interrupt. When this feature is
- enabled, it can reduce operating system overhead for disk Programmed
- Input/Output. On some systems, it also can increase the data
- throughput of Programmed Input/Output. Some drives, however, seemed
- to run slower with multiple sector mode enabled. Some drives claimed
- to support multiple sector mode, but lost data at some settings.
- Under rare circumstances, such failures could result in massive
- filesystem corruption.
-
- If you get the following error, try to say Y here:
-
- hda: set_multmode: status=0x51 { DriveReady SeekComplete Error }
- hda: set_multmode: error=0x04 { DriveStatusError }
-
- If in doubt, say N.
-
config BLK_DEV_IDECS
tristate "PCMCIA IDE support"
depends on PCMCIA
diff --git a/drivers/ide/Makefile b/drivers/ide/Makefile
index 64e0ecd..308b8a1 100644
--- a/drivers/ide/Makefile
```

[git pull] IDE updates #1

[git pull] IDE updates #1

```
+++ b/drivers/ide/Makefile
@@ -4,8 +4,8 @@

EXTRA_CFLAGS += -Idrivers/ide

-ide-core-y += ide.o ide-io.o ide-iops.o ide-lib.o ide-probe.o ide-taskfile.o \
- ide-pio-blacklist.o
+ide-core-y += ide.o ide-ioctls.o ide-io.o ide-iops.o ide-lib.o ide-probe.o \
+ ide-taskfile.o ide-pio-blacklist.o

# core IDE code
ide-core-$(CONFIG_IDE_TIMINGS) += ide-timings.o
@@ -37,11 +37,12 @@ obj-$(CONFIG_IDE_GENERIC) += ide-generic.o
obj-$(CONFIG_BLK_DEV_IDEPNP) += ide-pnp.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

obj-$(CONFIG_BLK_DEV_IDEDISK) += ide-disk.o
obj-$(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
-obj-$(CONFIG_BLK_DEV_IDEFLOPPY) += ide-floppy.o

ifeq ($(CONFIG_BLK_DEV_IDECS), y)
ide-cs-core-y += legacy/ide-cs.o
diff --git a/drivers/ide/arm/icside.c b/drivers/ide/arm/icside.c
index df4af40..70f5b16 100644
--- a/drivers/ide/arm/icside.c
+++ b/drivers/ide/arm/icside.c
@@ -10,7 +10,6 @@
#include <linux/slab.h>
#include <linux/blkdev.h>
#include <linux/errno.h>
-#include <linux/hdreg.h>
#include <linux/ide.h>
#include <linux/dma-mapping.h>
#include <linux/device.h>
@@ -265,8 +264,8 @@ static void icside_set_dma_mode(ide_drive_t *drive, const u8 xfer_mode)
* If we're going to be doing MW_DMA_1 or MW_DMA_2, we should
* take care to note the values in the ID...
*/
- if (use_dma_info && drive->id->eide_dma_time > cycle_time)
- cycle_time = drive->id->eide_dma_time;
+ if (use_dma_info && drive->id[ATA_ID_EIDE_DMA_TIME] > cycle_time)
+ cycle_time = drive->id[ATA_ID_EIDE_DMA_TIME];

drive->drive_data = cycle_time;

diff --git a/drivers/ide/arm/palm_bk3710.c b/drivers/ide/arm/palm_bk3710.c
index 4fd91dc..122ed3c 100644
```

[git pull] IDE updates #1

[git pull] IDE updates #1

```
--- a/drivers/ide/arm/palm_bk3710.c
+++ b/drivers/ide/arm/palm_bk3710.c
@@ -27,7 +27,6 @@
#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>
@@ -180,7 +179,7 @@ static void palm_bk3710_setpiomode(void __iomem *base, ide_drive_t *mate,
val32 |= (t2i << (dev ? 8 : 0));
writel(val32, base + BK3710_DATRCVR);

- if (mate && mate->present) {
+ if (mate) {
u8 mode2 = ide_get_best_pio_mode(mate, 255, 4);

if (mode2 < mode)
@@ -213,7 +212,8 @@ static void palm_bk3710_set_dma_mode(ide_drive_t *drive, u8 xferspeed)
palm_bk3710_setudmamode(base, is_slave,
xferspeed - XFER_UDMA_0);
} else {
- palm_bk3710_setdmamode(base, is_slave, drive->id->eide_dma_min,
+ palm_bk3710_setdmamode(base, is_slave,
+ drive->id[ATA_ID_EIDE_DMA_MIN],
xferspeed);
}
}
@@ -229,7 +229,7 @@ static void palm_bk3710_set_pio_mode(ide_drive_t *drive, u8 pio)
* Obtain the drive PIO data for tuning the Palm Chip registers
*/
cycle_time = ide_pio_cycle_time(drive, pio);
- mate = ide_get_paired_drive(drive);
+ mate = ide_get_pair_dev(drive);
palm_bk3710_setpiomode(base, mate, is_slave, cycle_time, pio);
}

diff --git a/drivers/ide/ide-acpi.c b/drivers/ide/ide-acpi.c
index 6f70462..2427c38 100644
--- a/drivers/ide/ide-acpi.c
+++ b/drivers/ide/ide-acpi.c
@@ -584,7 +584,7 @@ void ide_acpi_get_timing(ide_hwif_t *hwif)
* This function executes the _STM ACPI method for the target channel.
*
* _STM requires Identify Drive data, which has to passed as an argument.
- * Unfortunately hd_driveid is a mangled version which we can't readily
+ * Unfortunately drive->id is a mangled version which we can't readily
* use; hence we'll get the information afresh.
*/
void ide_acpi_push_timing(ide_hwif_t *hwif)
```

[git pull] IDE updates #1

```
@@ -614,10 +614,10 @@ void ide_acpi_push_timing(ide_hwif_t *hwif)
in_params[0].buffer.length = sizeof(struct GTM_buffer);
in_params[0].buffer.pointer = (u8 *)&hwif->acpdata->gtm;
in_params[1].type = ACPI_TYPE_BUFFER;
- in_params[1].buffer.length = sizeof(struct hd_driveid);
+ in_params[1].buffer.length = sizeof(ATA_ID_WORDS * 2);
in_params[1].buffer.pointer = (u8 *)&master->idbuff;
in_params[2].type = ACPI_TYPE_BUFFER;
- in_params[2].buffer.length = sizeof(struct hd_driveid);
+ in_params[2].buffer.length = sizeof(ATA_ID_WORDS * 2);
in_params[2].buffer.pointer = (u8 *)&slave->idbuff;
/* Output buffer: _STM has no output */
```

```
diff --git a/drivers/ide/ide-atapi.c b/drivers/ide/ide-atapi.c
```

```
index adf04f9..608c5ba 100644
```

```
--- a/drivers/ide/ide-atapi.c
```

```
+++ b/drivers/ide/ide-atapi.c
```

```
@@ -14,12 +14,201 @@
```

```
#define debug_log(fmt, args...) do { } while (0)
```

```
#endif
```

```
/*
```

```
+ * Check whether we can support a device,  
+ * based on the ATAPI IDENTIFY command results.
```

```
+ */
```

```
+int ide_check_atapi_device(ide_drive_t *drive, const char *s)
```

```
{
```

```
+ u16 *id = drive->id;
```

```
+ u8 gcw[2], protocol, device_type, removable, drq_type, packet_size;
```

```
+
```

```
+ *((u16 *)&gcw) = id[ATA_ID_CONFIG];
```

```
+
```

```
+ protocol = (gcw[1] & 0xC0) >> 6;
```

```
+ device_type = gcw[1] & 0x1F;
```

```
+ removable = (gcw[0] & 0x80) >> 7;
```

```
+ drq_type = (gcw[0] & 0x60) >> 5;
```

```
+ packet_size = gcw[0] & 0x03;
```

```
+
```

```
+#ifdef CONFIG_PPC
```

```
+ /* kludge for Apple PowerBook internal zip */
```

```
+ if (drive->media == ide_floppy && device_type == 5 &&
```

```
+ !strstr((char *)&id[ATA_ID_PROD], "CD-ROM") &&
```

```
+ strstr((char *)&id[ATA_ID_PROD], "ZIP"))
```

```
+ device_type = 0;
```

```
+#endif
```

```
+
```

```
+ if (protocol != 2)
```

```
+ printk(KERN_ERR "%s: %s: protocol (0x%02x) is not ATAPI\n",
```

```
+ s, drive->name, protocol);
```

```
+ else if ((drive->media == ide_floppy && device_type != 0) ||
```

```
+ (drive->media == ide_tape && device_type != 1))
```

[git pull] IDE updates #1

```
+ printk(KERN_ERR "%s: %s: invalid device type (0x%02x)\n",
+ s, drive->name, device_type);
+ else if (removable == 0)
+ printk(KERN_ERR "%s: %s: the removable flag is not set\n",
+ s, drive->name);
+ else if (drive->media == ide_floppy && drq_type == 3)
+ printk(KERN_ERR "%s: %s: sorry, DRQ type (0x%02x) not "
+ "supported\n", s, drive->name, drq_type);
+ else if (packet_size != 0)
+ printk(KERN_ERR "%s: %s: packet size (0x%02x) is not 12 "
+ "bytes\n", s, drive->name, packet_size);
+ else
+ return 1;
+ return 0;
+}
+EXPORT_SYMBOL_GPL(ide_check_atapi_device);
+
+/* PIO data transfer routine using the scatter gather table. */
+int ide_io_buffers(ide_drive_t *drive, struct ide_atapi_pc *pc,
+ unsigned int bcount, int write)
+{
+ ide_hwif_t *hwif = drive->hwif;
+ const struct ide_tp_ops *tp_ops = hwif->tp_ops;
+ xfer_func_t *xf = write ? tp_ops->output_data : tp_ops->input_data;
+ struct scatterlist *sg = pc->sg;
+ char *buf;
+ int count, done = 0;
+
+ while (bcount) {
+ count = min(sg->length - pc->b_count, bcount);
+
+ if (PageHighMem(sg_page(sg))) {
+ unsigned long flags;
+
+ local_irq_save(flags);
+ buf = kmap_atomic(sg_page(sg), KM_IRQ0) + sg->offset;
+ xf(drive, NULL, buf + pc->b_count, count);
+ kunmap_atomic(buf - sg->offset, KM_IRQ0);
+ local_irq_restore(flags);
+ } else {
+ buf = sg_virt(sg);
+ xf(drive, NULL, buf + pc->b_count, count);
+ }
+
+ bcount -= count;
+ pc->b_count += count;
+ done += count;
+
+ if (pc->b_count == sg->length) {
+ if (!--pc->sg_cnt)
+ break;
+ }
```

```

+ pc->sg = sg = sg_next(sg);
+ pc->b_count = 0;
+ }
+ }
+
+ if (bcount) {
+ printk(KERN_ERR "%s: %d leftover bytes, %s\n", drive->name,
+ bcount, write ? "padding with zeros"
+ : "discarding data");
+ ide_pad_transfer(drive, write, bcount);
+ }
+
+ return done;
+}
+EXPORT_SYMBOL_GPL(ide_io_buffers);
+
+void ide_init_pc(struct ide_atapi_pc *pc)
+{
+ memset(pc, 0, sizeof(*pc));
+ pc->buf = pc->pc_buf;
+ pc->buf_size = IDE_PC_BUFFER_SIZE;
+}
+EXPORT_SYMBOL_GPL(ide_init_pc);
+
+/*
+ * Generate a new packet command request in front of the request queue, before
+ * the current request, so that it will be processed immediately, on the next
+ * pass through the driver.
+ */
+void ide_queue_pc_head(ide_drive_t *drive, struct gendisk *disk,
+ struct ide_atapi_pc *pc, struct request *rq)
+{
+ blk_rq_init(NULL, rq);
+ rq->cmd_type = REQ_TYPE_SPECIAL;
+ rq->cmd_flags |= REQ_PREEMPT;
+ rq->buffer = (char *)pc;
+ rq->rq_disk = disk;
+ memcpy(rq->cmd, pc->c, 12);
+ if (drive->media == ide_tape)
+ rq->cmd[13] = REQ_IDETAPE_PC1;
+ ide_do_drive_cmd(drive, rq);
+}
+EXPORT_SYMBOL_GPL(ide_queue_pc_head);
+
+/*
+ * Add a special packet command request to the tail of the request queue,
+ * and wait for it to be serviced.
+ */
+int ide_queue_pc_tail(ide_drive_t *drive, struct gendisk *disk,
+ struct ide_atapi_pc *pc)
+{

```

[git pull] IDE updates #1

```
+ struct request *rq;
+ int error;
+
+ rq = blk_get_request(drive->queue, READ, __GFP_WAIT);
+ rq->cmd_type = REQ_TYPE_SPECIAL;
+ rq->buffer = (char *)pc;
+ memcpy(rq->cmd, pc->c, 12);
+ if (drive->media == ide_tape)
+ rq->cmd[13] = REQ_IDETAPE_PC1;
+ error = blk_execute_rq(drive->queue, disk, rq, 0);
+ blk_put_request(rq);
+
+ return error;
+}
+EXPORT_SYMBOL_GPL(ide_queue_pc_tail);
+
+int ide_do_test_unit_ready(ide_drive_t *drive, struct gendisk *disk)
+{
+ struct ide_atapi_pc pc;
+
+ ide_init_pc(&pc);
+ pc.c[0] = TEST_UNIT_READY;
+
+ return ide_queue_pc_tail(drive, disk, &pc);
+}
+EXPORT_SYMBOL_GPL(ide_do_test_unit_ready);
+
+int ide_do_start_stop(ide_drive_t *drive, struct gendisk *disk, int start)
+{
+ struct ide_atapi_pc pc;
+
+ ide_init_pc(&pc);
+ pc.c[0] = START_STOP;
+ pc.c[4] = start;
+
+ if (drive->media == ide_tape)
+ pc.flags |= PC_FLAG_WAIT_FOR_DSC;
+
+ return ide_queue_pc_tail(drive, disk, &pc);
+}
+EXPORT_SYMBOL_GPL(ide_do_start_stop);
+
+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)
+ return 0;
+
+ ide_init_pc(&pc);
+ pc.c[0] = ALLOW_MEDIUM_REMOVAL;
```

[git pull] IDE updates #1

```
+ pc.c[4] = on;
+
+ return ide_queue_pc_tail(drive, disk, &pc);
+}
+EXPORT_SYMBOL_GPL(ide_set_media_lock);
+
+/* TODO: unify the code thus making some arguments go away */
ide_startstop_t ide_pc_intr(ide_drive_t *drive, struct ide_atapi_pc *pc,
ide_handler_t *handler, unsigned int timeout, ide_expiry_t *expiry,
void (*update_buffers)(ide_drive_t *, struct ide_atapi_pc *),
void (*retry_pc)(ide_drive_t *), void (*dsc_handle)(ide_drive_t *),
- void (*io_buffers)(ide_drive_t *, struct ide_atapi_pc *, unsigned, int))
+ int (*io_buffers)(ide_drive_t *, struct ide_atapi_pc *, unsigned, int))
{
ide_hwif_t *hwif = drive->hwif;
struct request *rq = hwif->hwgroup->rq;
@@ -41,7 +230,7 @@ ide_startstop_t ide_pc_intr(ide_drive_t *drive, struct ide_atapi_pc *pc,

if (pc->flags & PC_FLAG_DMA_IN_PROGRESS) {
if (hwif->dma_ops->dma_end(drive) ||
- (drive->media == ide_tape && !scsi && (stat & ERR_STAT))) {
+ (drive->media == ide_tape && !scsi && (stat & ATA_ERR))) {
if (drive->media == ide_floppy && !scsi)
printk(KERN_ERR "%s: DMA %s error\n",
drive->name, rq_data_dir(pc->rq)
@@ -56,7 +245,7 @@ ide_startstop_t ide_pc_intr(ide_drive_t *drive, struct ide_atapi_pc *pc,
}

/* No more interrupts */
- if ((stat & DRQ_STAT) == 0) {
+ if ((stat & ATA_DRQ) == 0) {
debug_log("Packet command completed, %d bytes transferred\n",
pc->xferred);

@@ -65,10 +254,10 @@ ide_startstop_t ide_pc_intr(ide_drive_t *drive, struct ide_atapi_pc *pc,
local_irq_enable_in_hardirq();

if (drive->media == ide_tape && !scsi &&
- (stat & ERR_STAT) && rq->cmd[0] == REQUEST_SENSE)
- stat &= ~ERR_STAT;
+ (stat & ATA_ERR) && rq->cmd[0] == REQUEST_SENSE)
+ stat &= ~ATA_ERR;

- if ((stat & ERR_STAT) || (pc->flags & PC_FLAG_DMA_ERROR)) {
+ if ((stat & ATA_ERR) || (pc->flags & PC_FLAG_DMA_ERROR)) {
/* Error detected */
debug_log("%s: I/O error\n", drive->name);

@@ -95,7 +284,7 @@ ide_startstop_t ide_pc_intr(ide_drive_t *drive, struct ide_atapi_pc *pc,
cmd_finished:
pc->error = 0;
```

[git pull] IDE updates #1

```
if ((pc->flags & PC_FLAG_WAIT_FOR_DSC) &&
- (stat & SEEK_STAT) == 0) {
+ (stat & ATA_DSC) == 0) {
dsc_handle(drive);
return ide_stopped;
}
@@ -117,17 +306,18 @@ cmd_finished:
/* Get the number of bytes to transfer on this interrupt. */
ide_read_bcount_and_ireason(drive, &bcount, &ireason);

- if (ireason & CD) {
+ if (ireason & ATAPI_COD) {
printk(KERN_ERR "%s: CoD != 0 in %s\n", drive->name, __func__);
return ide_do_reset(drive);
}

- if (((ireason & IO) == IO) == !(pc->flags & PC_FLAG_WRITING)) {
+ if (((ireason & ATAPI_IO) == ATAPI_IO) ==
+ !(pc->flags & PC_FLAG_WRITING)) {
/* Hopefully, we will never get here */
printk(KERN_ERR "%s: We wanted to %s, but the device wants us "
"to %s!\n", drive->name,
- (ireason & IO) ? "Write" : "Read",
- (ireason & IO) ? "Read" : "Write");
+ (ireason & ATAPI_IO) ? "Write" : "Read",
+ (ireason & ATAPI_IO) ? "Read" : "Write");
return ide_do_reset(drive);
}

@@ -171,9 +361,14 @@ cmd_finished:

if ((drive->media == ide_floppy && !scsi && !pc->buf) ||
(drive->media == ide_tape && !scsi && pc->bh) ||
- (scsi && pc->sg))
- io_buffers(drive, pc, bcount, !(pc->flags & PC_FLAG_WRITING));
- else
+ (scsi && pc->sg)) {
+ int done = io_buffers(drive, pc, bcount,
+ !(pc->flags & PC_FLAG_WRITING));
+
+ /* FIXME: don't do partial completions */
+ if (drive->media == ide_floppy && !scsi)
+ ide_end_request(drive, 1, done >> 9);
+ } else
xferfunc(drive, NULL, pc->cur_pos, bcount);

/* Update the current position */
@@ -205,7 +400,8 @@ static u8 ide_wait_ireason(ide_drive_t *drive, u8 ireason)
{
int retries = 100;
```

[git pull] IDE updates #1

```
- while (retries-- && ((ireason & CD) == 0 || (ireason & IO))) {
+ while (retries-- && ((ireason & ATAPI_COD) == 0 ||
+ (ireason & ATAPI_IO))) {
printk(KERN_ERR "%s: (IO,CoD != (0,1) while issuing "
"a packet command, retrying\n", drive->name);
udelay(100);
@@ -214,8 +410,8 @@ static u8 ide_wait_ireason(ide_drive_t *drive, u8 ireason)
printk(KERN_ERR "%s: (IO,CoD != (0,1) while issuing "
"a packet command, ignoring\n",
drive->name);
- ireason |= CD;
- ireason &= ~IO;
+ ireason |= ATAPI_COD;
+ ireason &= ~ATAPI_IO;
}
}

@@ -231,7 +427,7 @@ ide_startstop_t ide_transfer_pc(ide_drive_t *drive, struct ide_atapi_pc *pc,
ide_startstop_t startstop;
u8 ireason;

- if (ide_wait_stat(&startstop, drive, DRQ_STAT, BUSY_STAT, WAIT_READY)) {
+ if (ide_wait_stat(&startstop, drive, ATA_DRQ, ATA_BUSY, WAIT_READY)) {
printk(KERN_ERR "%s: Strange, packet command initiated yet "
"DRQ isn't asserted\n", drive->name);
return startstop;
@@ -241,7 +437,7 @@ ide_startstop_t ide_transfer_pc(ide_drive_t *drive, struct ide_atapi_pc *pc,
if (drive->media == ide_tape && !drive->scsi)
ireason = ide_wait_ireason(drive, ireason);

- if ((ireason & CD) == 0 || (ireason & IO)) {
+ if ((ireason & ATAPI_COD) == 0 || (ireason & ATAPI_IO)) {
printk(KERN_ERR "%s: (IO,CoD) != (0,1) while issuing "
"a packet command\n", drive->name);
return ide_do_reset(drive);
@@ -303,7 +499,7 @@ ide_startstop_t ide_issue_pc(ide_drive_t *drive, struct ide_atapi_pc *pc,

/* Issue the packet command */
if (drive->atapi_flags & IDE_AFLAG_DRQ_INTERRUPT) {
- ide_execute_command(drive, WIN_PACKETCMD, handler,
+ ide_execute_command(drive, ATA_CMD_PACKET, handler,
timeout, NULL);
return ide_started;
} else {
diff --git a/drivers/ide/ide-cd.c b/drivers/ide/ide-cd.c
index 03c2cb6..465a92c 100644
--- a/drivers/ide/ide-cd.c
+++ b/drivers/ide/ide-cd.c
@@ -436,7 +436,7 @@ static int cdrom_decode_status(ide_drive_t *drive, int good_stat, int *stat_ret)
ide_dump_status_no_sense(drive, "media error (blank)",
stat);
```

[git pull] IDE updates #1

```
do_end_request = 1;
- } else if ((err & ~ABRT_ERR) != 0) {
+ } else if ((err & ~ATA_ABORTED) != 0) {
/* go to the default handler for other errors */
ide_error(drive, "cdrom_decode_status", stat);
return 1;
@@ -457,7 +457,7 @@ static int cdrom_decode_status(ide_drive_t *drive, int good_stat, int *stat_ret)
* If we got a CHECK_CONDITION status, queue
* a request sense command.
*/
- if (stat & ERR_STAT)
+ if (stat & ATA_ERR)
cdrom_queue_request_sense(drive, NULL, NULL);
} else {
blk_dump_rq_flags(rq, "ide-cd: bad rq");
@@ -468,7 +468,7 @@ static int cdrom_decode_status(ide_drive_t *drive, int good_stat, int *stat_ret)
return 1;

end_request:
- if (stat & ERR_STAT) {
+ if (stat & ATA_ERR) {
unsigned long flags;

spin_lock_irqsave(&ide_lock, flags);
@@ -541,7 +541,7 @@ static ide_startstop_t cdrom_start_packet_command(ide_drive_t *drive,
drive->waiting_for_dma = 0;

/* packet command */
- ide_execute_command(drive, WIN_PACKETCMD, handler,
+ ide_execute_command(drive, ATA_CMD_PACKET, handler,
ATAPI_WAIT_PC, cdrom_timer_expiry);
return ide_started;
} else {
@@ -574,7 +574,7 @@ static ide_startstop_t cdrom_transfer_packet_command(ide_drive_t *drive,
*/

/* check for errors */
- if (cdrom_decode_status(drive, DRQ_STAT, NULL))
+ if (cdrom_decode_status(drive, ATA_DRQ, NULL))
return ide_stopped;

/* ok, next interrupt will be DMA interrupt */
@@ -582,8 +582,8 @@ static ide_startstop_t cdrom_transfer_packet_command(ide_drive_t *drive,
drive->waiting_for_dma = 1;
} else {
/* otherwise, we must wait for DRQ to get set */
- if (ide_wait_stat(&startstop, drive, DRQ_STAT,
- BUSY_STAT, WAIT_READY))
+ if (ide_wait_stat(&startstop, drive, ATA_DRQ,
+ ATA_BUSY, WAIT_READY))
return startstop;
```

[git pull] IDE updates #1

```
}

@@ -938,7 +938,7 @@ static ide_startstop_t cdrom_newpc_intr(ide_drive_t *drive)
thislen = len;

/* If DRQ is clear, the command has completed. */
- if ((stat & DRQ_STAT) == 0) {
+ if ((stat & ATA_DRQ) == 0) {
if (blk_fs_request(rq)) {
/*
* If we're not done reading/writing, complain.
@@ -1164,13 +1164,12 @@ static void cdrom_do_block_pc(ide_drive_t *drive, struct request *rq)
if (rq->bio || ((rq->cmd_type == REQ_TYPE_ATA_PC) && rq->data_len)) {
struct request_queue *q = drive->queue;
unsigned int alignment;
- unsigned long addr;
- unsigned long stack_mask = ~(THREAD_SIZE - 1);
+ char *buf;

if (rq->bio)
- addr = (unsigned long)bio_data(rq->bio);
+ buf = bio_data(rq->bio);
else
- addr = (unsigned long)rq->data;
+ buf = rq->data;

info->dma = drive->using_dma;

@@ -1181,11 +1180,8 @@ static void cdrom_do_block_pc(ide_drive_t *drive, struct request *rq)
* separate masks.
*/
alignment = queue_dma_alignment(q) | q->dma_pad_mask;
- if (addr & alignment || rq->data_len & alignment)
- info->dma = 0;
-
- if (!(addr & stack_mask) ^
- ((unsigned long)current->stack & stack_mask))
+ if ((unsigned long)buf & alignment || rq->data_len & alignment
+ || object_is_on_stack(buf))
info->dma = 0;
}
}
@@ -1206,7 +1202,7 @@ static ide_startstop_t ide_cd_do_request(ide_drive_t *drive, struct request *rq,
unsigned long elapsed = jiffies - info->start_seek;
int stat = hwif->tp_ops->read_status(hwif);

- if ((stat & SEEK_STAT) != SEEK_STAT) {
+ if ((stat & ATA_DSC) != ATA_DSC) {
if (elapsed < IDECD_SEEK_TIMEOUT) {
ide_stall_queue(drive,
IDECD_SEEK_TIMER);
```

[git pull] IDE updates #1

```
@@ -1813,13 +1809,12 @@ static ide_proc_entry_t idecd_proc[] = {
{ NULL, 0, NULL, NULL }
};

-static void ide_cdrom_add_settings(ide_drive_t *drive)
-{
- ide_add_setting(drive, "dsc_overlap", SETTING_RW, TYPE_BYTE, 0, 1, 1, 1,
- &drive->dsc_overlap, NULL);
-}
-#else
-static inline void ide_cdrom_add_settings(ide_drive_t *drive) { ; }
+ide_devset_rw_field(dsc_overlap, dsc_overlap);
+
+static const struct ide_proc_devset idecd_settings[] = {
+ IDE_PROC_DEVSET(dsc_overlap, 0, 1),
+ { 0 },
+};
#endif

static const struct cd_list_entry ide_cd_quirks_list[] = {
@@ -1866,14 +1861,14 @@ static const struct cd_list_entry ide_cd_quirks_list[] = {
{ NULL, NULL, 0 }
};

-static unsigned int ide_cd_flags(struct hd_driveid *id)
+static unsigned int ide_cd_flags(u16 *id)
{
const struct cd_list_entry *cle = ide_cd_quirks_list;

while (cle->id_model) {
- if (strcmp(cle->id_model, id->model) == 0 &&
+ if (strcmp(cle->id_model, (char *)&id[ATA_ID_PROD]) == 0 &&
(cle->id_firmware == NULL ||
- strstr(id->fw_rev, cle->id_firmware)))
+ strstr((char *)&id[ATA_ID_FW_REV], cle->id_firmware)))
return cle->cd_flags;
cle++;
}
@@ -1885,7 +1880,8 @@ static int ide_cdrom_setup(ide_drive_t *drive)
{
struct cdrom_info *cd = drive->driver_data;
struct cdrom_device_info *cdi = &cd->devinfo;
- struct hd_driveid *id = drive->id;
+ u16 *id = drive->id;
+ char *fw_rev = (char *)&id[ATA_ID_FW_REV];
int nslots;

blk_queue_prep_rq(drive->queue, ide_cdrom_prep_fn);
@@ -1900,15 +1896,15 @@ static int ide_cdrom_setup(ide_drive_t *drive)
drive->atapi_flags = IDE_AFLAG_MEDIA_CHANGED | IDE_AFLAG_NO_EJECT |
ide_cd_flags(id);
```

```

- if ((id->config & 0x0060) == 0x20)
+ if ((id[ATA_ID_CONFIG] & 0x0060) == 0x20)
drive->atapi_flags |= IDE_AFLAG_DRQ_INTERRUPT;

if ((drive->atapi_flags & IDE_AFLAG_VERTOS_300_SSD) &&
- id->fw_rev[4] == '1' && id->fw_rev[6] <= '2')
+ fw_rev[4] == '1' && fw_rev[6] <= '2')
drive->atapi_flags |= (IDE_AFLAG_TOCTRACKS_AS_BCD |
IDE_AFLAG_TOCADDR_AS_BCD);
else if ((drive->atapi_flags & IDE_AFLAG_VERTOS_600_ESD) &&
- id->fw_rev[4] == '1' && id->fw_rev[6] <= '2')
+ fw_rev[4] == '1' && fw_rev[6] <= '2')
drive->atapi_flags |= IDE_AFLAG_TOCTRACKS_AS_BCD;
else if (drive->atapi_flags & IDE_AFLAG_SANYO_3CD)
/* 3 => use CD in slot 0 */
@@ -1927,7 +1923,8 @@ static int ide_cdrom_setup(ide_drive_t *drive)
cd->devinfo.handle = NULL;
return 1;
}
- ide_cdrom_add_settings(drive);
+
+ ide_proc_register_driver(drive, cd->driver);
return 0;
}

@@ -1972,12 +1969,12 @@ static ide_driver_t ide_cdrom_driver = {
.remove = ide_cd_remove,
.version = IDECD_VERSION,
.media = ide_cdrom,
- .supports_dsc_overlap = 1,
.do_request = ide_cd_do_request,
.end_request = ide_end_request,
.error = __ide_error,
#ifdef CONFIG_IDE_PROC_FS
.proc = idecd_proc,
+ .settings = idecd_settings,
#endif
};

@@ -2112,10 +2109,10 @@ static int ide_cd_probe(ide_drive_t *drive)

if (!strstr("ide-cdrom", drive->driver_req))
goto failed;
- if (!drive->present)
- goto failed;
+
if (drive->media != ide_cdrom && drive->media != ide_optical)
goto failed;
+
/* skip drives that we were told to ignore */

```

[git pull] IDE updates #1

```
if (ignore != NULL) {
if (strstr(ignore, drive->name)) {
@@ -2137,8 +2134,6 @@ static int ide_cd_probe(ide_drive_t *drive)

ide_init_disk(g, drive);

- ide_proc_register_driver(drive, &ide_cdrom_driver);
-
kref_init(&info->kref);

info->drive = drive;
@@ -2153,7 +2148,6 @@ static int ide_cd_probe(ide_drive_t *drive)
g->driverfs_dev = &drive->gendev;
g->flags = GENHD_FL_CD | GENHD_FL_REMOVABLE;
if (ide_cdrom_setup(drive)) {
- ide_proc_unregister_driver(drive, &ide_cdrom_driver);
ide_cd_release(&info->kref);
goto failed;
}
diff --git a/drivers/ide/ide-disk.c b/drivers/ide/ide-disk.c
index 33ea8c0..01846f2 100644
--- a/drivers/ide/ide-disk.c
+++ b/drivers/ide/ide-disk.c
@@ -30,10 +30,8 @@
#include <linux/delay.h>
#include <linux/mutex.h>
#include <linux/leds.h>
-
-#define _IDE_DISK
-
#include <linux/ide.h>
+#include <linux/hdreg.h>

#include <asm/byteorder.h>
#include <asm/irq.h>
@@ -90,68 +88,19 @@ static void ide_disk_put(struct ide_disk_obj *idkp)
mutex_unlock(&idedisk_ref_mutex);
}

-/*
- * lba_capacity_is_ok() performs a sanity check on the claimed "lba_capacity"
- * value for this drive (from its reported identification information).
- *
- * Returns: 1 if lba_capacity looks sensible
- * 0 otherwise
- *
- * It is called only once for each drive.
- */
-static int lba_capacity_is_ok(struct hd_driveid *id)
-{
- unsigned long lba_sects, chs_sects, head, tail;
```

```

-
- /* No non-LBA info .. so valid! */
- if (id->cyls == 0)
- return 1;
-
- /*
- * The ATA spec tells large drives to return
- * C/H/S = 16383/16/63 independent of their size.
- * Some drives can be jumpered to use 15 heads instead of 16.
- * Some drives can be jumpered to use 4092 cyls instead of 16383.
- */
- if ((id->cyls == 16383
- || (id->cyls == 4092 && id->cur_cyls == 16383)) &&
- id->sectors == 63 &&
- (id->heads == 15 || id->heads == 16) &&
- (id->lba_capacity >= 16383*63*id->heads))
- return 1;
-
- lba_sects = id->lba_capacity;
- chs_sects = id->cyls * id->heads * id->sectors;
-
- /* perform a rough sanity check on lba_sects: within 10% is OK */
- if ((lba_sects - chs_sects) < chs_sects/10)
- return 1;
-
- /* some drives have the word order reversed */
- head = ((lba_sects >> 16) & 0xffff);
- tail = (lba_sects & 0xffff);
- lba_sects = (head | (tail << 16));
- if ((lba_sects - chs_sects) < chs_sects/10) {
- id->lba_capacity = lba_sects;
- return 1; /* lba_capacity is (now) good */
- }
-
- return 0; /* lba_capacity value may be bad */
-}
-
static const u8 ide_rw_cmds[] = {
- WIN_MULTREAD,
- WIN_MULTWRITE,
- WIN_MULTREAD_EXT,
- WIN_MULTWRITE_EXT,
- WIN_READ,
- WIN_WRITE,
- WIN_READ_EXT,
- WIN_WRITE_EXT,
- WIN_READDMA,
- WIN_WRITEDMA,
- WIN_READDMA_EXT,
- WIN_WRITEDMA_EXT,
+ ATA_CMD_READ_MULTI,

```

```

+ ATA_CMD_WRITE_MULTI,
+ ATA_CMD_READ_MULTI_EXT,
+ ATA_CMD_WRITE_MULTI_EXT,
+ ATA_CMD_PIO_READ,
+ ATA_CMD_PIO_WRITE,
+ ATA_CMD_PIO_READ_EXT,
+ ATA_CMD_PIO_WRITE_EXT,
+ ATA_CMD_READ,
+ ATA_CMD_WRITE,
+ ATA_CMD_READ_EXT,
+ ATA_CMD_WRITE_EXT,
};

static const u8 ide_data_phases[] = {
@@ -322,9 +271,9 @@ static u64 idedisk_read_native_max_address(ide_drive_t *drive, int lba48)
/* Create IDE/ATA command request structure */
memset(&args, 0, sizeof(ide_task_t));
if (lba48)
- tf->command = WIN_READ_NATIVE_MAX_EXT;
+ tf->command = ATA_CMD_READ_NATIVE_MAX_EXT;
else
- tf->command = WIN_READ_NATIVE_MAX;
+ tf->command = ATA_CMD_READ_NATIVE_MAX;
tf->device = ATA_LBA;
args.tf_flags = IDE_TFLAG_TF | IDE_TFLAG_DEVICE;
if (lba48)
@@ -359,10 +308,10 @@ static u64 idedisk_set_max_address(ide_drive_t *drive, u64 addr_req, int lba48)
tf->hob_lbal = (addr_req >>= 8) & 0xff;
tf->hob_lbam = (addr_req >>= 8) & 0xff;
tf->hob_lbah = (addr_req >>= 8) & 0xff;
- tf->command = WIN_SET_MAX_EXT;
+ tf->command = ATA_CMD_SET_MAX_EXT;
} else {
tf->device = (addr_req >>= 8) & 0x0f;
- tf->command = WIN_SET_MAX;
+ tf->command = ATA_CMD_SET_MAX;
}
tf->device |= ATA_LBA;
args.tf_flags = IDE_TFLAG_TF | IDE_TFLAG_DEVICE;
@@ -385,25 +334,6 @@ static unsigned long long sectors_to_MB(unsigned long long n)
}

/*
- * Bits 10 of command_set_1 and cfs_enable_1 must be equal,
- * so on non-buggy drives we need test only one.
- * However, we should also check whether these fields are valid.
- */
-static inline int idedisk_supports_hpa(const struct hd_driveid *id)
-{
- return (id->command_set_1 & 0x0400) && (id->cfs_enable_1 & 0x0400);
-}

```

[git pull] IDE updates #1

```
-
-/*
- * The same here.
- */
-static inline int idedisk_supports_lba48(const struct hd_driveid *id)
-{
- return (id->command_set_2 & 0x0400) && (id->cfs_enable_2 & 0x0400)
- && id->lba_capacity_2;
-}
-
-/*
- * Some disks report total number of sectors instead of
- * maximum sector address. We list them here.
- */
@@ -417,7 +347,7 @@ static const struct drive_list_entry hpa_list[] = {
static void idedisk_check_hpa(ide_drive_t *drive)
{
unsigned long long capacity, set_max;
- int lba48 = idedisk_supports_lba48(drive->id);
+ int lba48 = ata_id_lba48_enabled(drive->id);

capacity = drive->capacity64;

@@ -453,23 +383,23 @@ static void idedisk_check_hpa(ide_drive_t *drive)

static void init_idedisk_capacity(ide_drive_t *drive)
{
- struct hd_driveid *id = drive->id;
+ u16 *id = drive->id;
-/*
- * If this drive supports the Host Protected Area feature set,
- * then we may need to change our opinion about the drive's capacity.
- */
- int hpa = idedisk_supports_hpa(id);
+ int hpa = ata_id_hpa_enabled(id);

- if (idedisk_supports_lba48(id)) {
+ if (ata_id_lba48_enabled(id)) {
/* drive speaks 48-bit LBA */
drive->select.b.lba = 1;
- drive->capacity64 = id->lba_capacity_2;
+ drive->capacity64 = ata_id_u64(id, ATA_ID_LBA_CAPACITY_2);
if (hpa)
idedisk_check_hpa(drive);
- } else if ((id->capability & 2) && lba_capacity_is_ok(id)) {
+ } else if (ata_id_has_lba(id) && ata_id_is_lba_capacity_ok(id)) {
/* drive speaks 28-bit LBA */
drive->select.b.lba = 1;
- drive->capacity64 = id->lba_capacity;
+ drive->capacity64 = ata_id_u32(id, ATA_ID_LBA_CAPACITY);
if (hpa)
```

[git pull] IDE updates #1

```
idedisk_check_hpa(drive);
} else {
@@ -480,7 +410,7 @@ static void init_idedisk_capacity(ide_drive_t *drive)

static sector_t idedisk_capacity(ide_drive_t *drive)
{
- return drive->capacity64 - drive->sect0;
+ return drive->capacity64;
}

#ifdef CONFIG_IDE_PROC_FS
@@ -490,10 +420,10 @@ static int smart_enable(ide_drive_t *drive)
struct ide_taskfile *tf = &args.tf;

memset(&args, 0, sizeof(ide_task_t));
- tf->feature = SMART_ENABLE;
- tf->lbam = SMART_LCYL_PASS;
- tf->lbah = SMART_HCYL_PASS;
- tf->command = WIN_SMART;
+ tf->feature = ATA_SMART_ENABLE;
+ tf->lbam = ATA_SMART_LBAM_PASS;
+ tf->lbah = ATA_SMART_LBAH_PASS;
+ tf->command = ATA_CMD_SMART;
args.tf_flags = IDE_TFLAG_TF | IDE_TFLAG_DEVICE;
return ide_no_data_taskfile(drive, &args);
}
@@ -506,9 +436,9 @@ static int get_smart_data(ide_drive_t *drive, u8 *buf, u8 sub_cmd)
memset(&args, 0, sizeof(ide_task_t));
tf->feature = sub_cmd;
tf->nsect = 0x01;
- tf->lbam = SMART_LCYL_PASS;
- tf->lbah = SMART_HCYL_PASS;
- tf->command = WIN_SMART;
+ tf->lbam = ATA_SMART_LBAM_PASS;
+ tf->lbah = ATA_SMART_LBAH_PASS;
+ tf->command = ATA_CMD_SMART;
args.tf_flags = IDE_TFLAG_TF | IDE_TFLAG_DEVICE;
args.data_phase = TASKFILE_IN;
(void) smart_enable(drive);
@@ -523,7 +453,7 @@ static int proc_idedisk_read_cache
int len;

if (drive->id_read)
- len = sprintf(out, "%i\n", drive->id->buf_size / 2);
+ len = sprintf(out, "%i\n", drive->id[ATA_ID_BUF_SIZE] / 2);
else
len = sprintf(out, "(none)\n");

@@ -549,13 +479,14 @@ static int proc_idedisk_read_smart(char *page, char **start, off_t off,

if (get_smart_data(drive, page, sub_cmd) == 0) {
```

```

unsigned short *val = (unsigned short *) page;
- char *out = ((char *)val) + (SECTOR_WORDS * 4);
+ char *out = (char *)val + SECTOR_SIZE;
+
page = out;
do {
out += sprintf(out, "%04x%c", le16_to_cpu(*val),
(++i & 7) ? ' ' : '\n');
val += 1;
- } while (i < (SECTOR_WORDS * 2));
+ } while (i < SECTOR_SIZE / 2);
len = out - page;
}

@@ -566,14 +497,14 @@ static int proc_idedisk_read_sv
(char *page, char **start, off_t off, int count, int *eof, void *data)
{
return proc_idedisk_read_smart(page, start, off, count, eof, data,
- SMART_READ_VALUES);
+ ATA_SMART_READ_VALUES);
}

static int proc_idedisk_read_st
(char *page, char **start, off_t off, int count, int *eof, void *data)
{
return proc_idedisk_read_smart(page, start, off, count, eof, data,
- SMART_READ_THRESHOLDS);
+ ATA_SMART_READ_THRESHOLDS);
}

static ide_proc_entry_t idedisk_proc[] = {
@@ -595,11 +526,11 @@ static void idedisk_prepare_flush(struct request_queue *q, struct request *rq)
BUG_ON(task == NULL);

memset(task, 0, sizeof(*task));
- if (ide_id_has_flush_cache_ext(drive->id) &&
+ if (ata_id_flush_ext_enabled(drive->id) &&
(drive->capacity64 >= (1UL << 28)))
- task->tf.command = WIN_FLUSH_CACHE_EXT;
+ task->tf.command = ATA_CMD_FLUSH_EXT;
else
- task->tf.command = WIN_FLUSH_CACHE;
+ task->tf.command = ATA_CMD_FLUSH;
task->tf.flags = IDE_TFLAG_OUT_TF | IDE_TFLAG_OUT_DEVICE |
IDE_TFLAG_DYN;
task->data_phase = TASKFILE_NO_DATA;
@@ -609,6 +540,8 @@ static void idedisk_prepare_flush(struct request_queue *q, struct request *rq)
rq->special = task;
}

+ide_devset_get(multcount, mult_count);

```

[git pull] IDE updates #1

```
+
/*
 * This is tightly woven into the driver->do_special can not touch.
 * DON'T do it again until a total personality rewrite is committed.
@@ -618,7 +551,7 @@ static int set_multcount(ide_drive_t *drive, int arg)
struct request *rq;
int error;

- if (arg < 0 || arg > drive->id->max_multsect)
+ if (arg < 0 || arg > (drive->id[ATA_ID_MAX_MULTSECT] & 0xff))
return -EINVAL;

if (drive->special.b.set_multmode)
@@ -635,22 +568,21 @@ static int set_multcount(ide_drive_t *drive, int arg)
return (drive->mult_count == arg) ? 0 : -EIO;
}

+ide_devset_get(nowerr, nowerr);
+
static int set_nowerr(ide_drive_t *drive, int arg)
{
if (arg < 0 || arg > 1)
return -EINVAL;

- if (ide_spin_wait_hwgroup(drive))
- return -EBUSY;
drive->nowerr = arg;
drive->bad_wstat = arg ? BAD_R_STAT : BAD_W_STAT;
- spin_unlock_irq(&ide_lock);
return 0;
}

static void update_ordered(ide_drive_t *drive)
{
- struct hd_driveid *id = drive->id;
+ u16 *id = drive->id;
unsigned ordered = QUEUE_ORDERED_NONE;
prepare_flush_fn *prep_fn = NULL;

@@ -666,9 +598,9 @@ static void update_ordered(ide_drive_t *drive)
 * not available so we don't need to recheck that.
 */
capacity = idedisk_capacity(drive);
- barrier = ide_id_has_flush_cache(id) && !drive->noflush &&
+ barrier = ata_id_flush_enabled(id) && !drive->noflush &&
(drive->addressing == 0 || capacity <= (1ULL << 28) ||
- ide_id_has_flush_cache_ext(id));
+ ata_id_flush_ext_enabled(id));

printk(KERN_INFO "%s: cache flushes %ssupported\n",
drive->name, barrier ? "" : "not ");
```

[git pull] IDE updates #1

```
@@ -683,7 +615,9 @@ static void update_ordered(ide_drive_t *drive)
blk_queue_ordered(drive->queue, ordered, prep_fn);
}

-static int write_cache(ide_drive_t *drive, int arg)
+ide_devset_get(wcache, wcache);
+
+static int set_wcache(ide_drive_t *drive, int arg)
{
ide_task_t args;
int err = 1;
@@ -691,11 +625,11 @@ static int write_cache(ide_drive_t *drive, int arg)
if (arg < 0 || arg > 1)
return -EINVAL;

- if (ide_id_has_flush_cache(drive->id)) {
+ if (ata_id_flush_enabled(drive->id)) {
memset(&args, 0, sizeof(ide_task_t));
args.tf.feature = arg ?
- SETFEATURES_EN_WCACHE : SETFEATURES_DIS_WCACHE;
- args.tf.command = WIN_SETFEATURES;
+ SETFEATURES_WC_ON : SETFEATURES_WC_OFF;
+ args.tf.command = ATA_CMD_SET_FEATURES;
args.tf.flags = IDE_TFLAG_TF | IDE_TFLAG_DEVICE;
err = ide_no_data_taskfile(drive, &args);
if (err == 0)
@@ -712,14 +646,16 @@ static int do_idedisk_flushcache(ide_drive_t *drive)
ide_task_t args;

memset(&args, 0, sizeof(ide_task_t));
- if (ide_id_has_flush_cache_ext(drive->id))
- args.tf.command = WIN_FLUSH_CACHE_EXT;
+ if (ata_id_flush_ext_enabled(drive->id))
+ args.tf.command = ATA_CMD_FLUSH_EXT;
else
- args.tf.command = WIN_FLUSH_CACHE;
+ args.tf.command = ATA_CMD_FLUSH;
args.tf.flags = IDE_TFLAG_TF | IDE_TFLAG_DEVICE;
return ide_no_data_taskfile(drive, &args);
}

+ide_devset_get(acoustic, acoustic);
+
static int set_acoustic(ide_drive_t *drive, int arg)
{
ide_task_t args;
@@ -728,22 +664,24 @@ static int set_acoustic(ide_drive_t *drive, int arg)
return -EINVAL;

memset(&args, 0, sizeof(ide_task_t));
- args.tf.feature = arg ? SETFEATURES_EN_AAM : SETFEATURES_DIS_AAM;
```

[git pull] IDE updates #1

```
+ args.tf.feature = arg ? SETFEATURES_AAM_ON : SETFEATURES_AAM_OFF;
args.tf.nsect = arg;
- args.tf.command = WIN_SETFEATURES;
+ args.tf.command = ATA_CMD_SET_FEATURES;
args.tf.flags = IDE_TFLAG_TF | IDE_TFLAG_DEVICE;
ide_no_data_taskfile(drive, &args);
drive->acoustic = arg;
return 0;
}

+ide_devset_get(addressing, addressing);
+
+/*
+ * drive->addressing:
+ * 0: 28-bit
+ * 1: 48-bit
+ * 2: 48-bit capable doing 28-bit
+ */
-static int set_lba_addressing(ide_drive_t *drive, int arg)
+static int set_addressing(ide_drive_t *drive, int arg)
{
if (arg < 0 || arg > 2)
return -EINVAL;
@@ -753,52 +691,54 @@ static int set_lba_addressing(ide_drive_t *drive, int arg)
if (drive->hwif->host_flags & IDE_HFLAG_NO_LBA48)
return 0;

- if (!idedisk_supports_lba48(drive->id))
+ if (ata_id_lba48_enabled(drive->id) == 0)
return -EIO;
+
drive->addressing = arg;
+
return 0;
}

+ide_devset_rw(acoustic, acoustic);
+ide_devset_rw(address, addressing);
+ide_devset_rw(multcount, multcount);
+ide_devset_rw(wcache, wcache);
+
+ide_devset_rw_sync(nowerr, nowerr);
+
#ifdef CONFIG_IDE_PROC_FS
-static void idedisk_add_settings(ide_drive_t *drive)
-{
- struct hd_driveid *id = drive->id;
-
- ide_add_setting(drive, "bios_cyl", SETTING_RW, TYPE_INT, 0, 65535, 1, 1,
- &drive->bios_cyl, NULL);
- ide_add_setting(drive, "bios_head", SETTING_RW, TYPE_BYTE, 0, 255, 1, 1,
```

[git pull] IDE updates #1

```
- &drive->bios_head, NULL);
- ide_add_setting(drive, "bios_sect", SETTING_RW, TYPE_BYTE, 0, 63, 1, 1,
- &drive->bios_sect, NULL);
- ide_add_setting(drive, "address", SETTING_RW, TYPE_BYTE, 0, 2, 1, 1,
- &drive->addressing, set_lba_addressing);
- ide_add_setting(drive, "multcount", SETTING_RW, TYPE_BYTE, 0,
- id->max_multsect, 1, 1, &drive->mult_count,
- set_multcount);
- ide_add_setting(drive, "nowerr", SETTING_RW, TYPE_BYTE, 0, 1, 1, 1,
- &drive->nowerr, set_nowerr);
- ide_add_setting(drive, "lun", SETTING_RW, TYPE_INT, 0, 7, 1, 1,
- &drive->lun, NULL);
- ide_add_setting(drive, "wcache", SETTING_RW, TYPE_BYTE, 0, 1, 1, 1,
- &drive->wcache, write_cache);
- ide_add_setting(drive, "acoustic", SETTING_RW, TYPE_BYTE, 0, 254, 1, 1,
- &drive->acoustic, set_acoustic);
- ide_add_setting(drive, "failures", SETTING_RW, TYPE_INT, 0, 65535, 1, 1,
- &drive->failures, NULL);
- ide_add_setting(drive, "max_failures", SETTING_RW, TYPE_INT, 0, 65535,
- 1, 1, &drive->max_failures, NULL);
-}
-#else
-static inline void idedisk_add_settings(ide_drive_t *drive) { ; }
+ide_devset_rw_field(bios_cyl, bios_cyl);
+ide_devset_rw_field(bios_head, bios_head);
+ide_devset_rw_field(bios_sect, bios_sect);
+ide_devset_rw_field(failures, failures);
+ide_devset_rw_field(lun, lun);
+ide_devset_rw_field(max_failures, max_failures);
+
+static const struct ide_proc_devset idedisk_settings[] = {
+ IDE_PROC_DEVSET(acoustic, 0, 254),
+ IDE_PROC_DEVSET(address, 0, 2),
+ IDE_PROC_DEVSET(bios_cyl, 0, 65535),
+ IDE_PROC_DEVSET(bios_head, 0, 255),
+ IDE_PROC_DEVSET(bios_sect, 0, 63),
+ IDE_PROC_DEVSET(failures, 0, 65535),
+ IDE_PROC_DEVSET(lun, 0, 7),
+ IDE_PROC_DEVSET(max_failures, 0, 65535),
+ IDE_PROC_DEVSET(multcount, 0, 16),
+ IDE_PROC_DEVSET(nowerr, 0, 1),
+ IDE_PROC_DEVSET(wcache, 0, 1),
+ { 0 },
+};
#endif

static void idedisk_setup(ide_drive_t *drive)
{
+ struct ide_disk_obj *idkp = drive->driver_data;
ide_hwif_t *hwif = drive->hwif;
- struct hd_driveid *id = drive->id;
```

[git pull] IDE updates #1

```
+ u16 *id = drive->id;
+ char *m = (char *)&id[ATA_ID_PROD];
unsigned long long capacity;

- idedisk_add_settings(drive);
+ ide_proc_register_driver(drive, idkp->driver);

if (drive->id_read == 0)
return;
@@ -807,11 +747,11 @@ static void idedisk_setup(ide_drive_t *drive)
/*
* Removable disks (eg. SYQUEST); ignore 'WD' drives
*/
- if (id->model[0] != 'W' || id->model[1] != 'D')
+ if (m[0] != 'W' || m[1] != 'D')
drive->doorlocking = 1;
}

- (void)set_lba_addressing(drive, 1);
+ (void)set_addressing(drive, 1);

if (drive->addressing == 1) {
int max_s = 2048;
@@ -853,8 +793,7 @@ static void idedisk_setup(ide_drive_t *drive)
capacity = idedisk_capacity(drive);

if (!drive->forced_geom) {
-
- if (idedisk_supports_lba48(drive->id)) {
+ if (ata_id_lba48_enabled(drive->id)) {
/* compatibility */
drive->bios_sect = 63;
drive->bios_head = 255;
@@ -880,22 +819,22 @@ static void idedisk_setup(ide_drive_t *drive)
drive->name, capacity, sectors_to_MB(capacity));

/* Only print cache size when it was specified */
- if (id->buf_size)
- printk(KERN_CONT " w/%dKiB Cache", id->buf_size / 2);
+ if (id[ATA_ID_BUF_SIZE])
+ printk(KERN_CONT " w/%dKiB Cache", id[ATA_ID_BUF_SIZE] / 2);

printk(KERN_CONT ", CHS=%d/%d/%d\n",
drive->bios_cyl, drive->bios_head, drive->bios_sect);

/* write cache enabled? */
- if ((id->csfo & 1) || (id->cfs_enable_1 & (1 << 5)))
+ if ((id[ATA_ID_CSFO] & 1) || ata_id_wcache_enabled(id))
drive->wcache = 1;

- write_cache(drive, 1);
```

[git pull] IDE updates #1

```
+ set_wcache(drive, 1);
}

static void ide_cacheflush_p(ide_drive_t *drive)
{
- if (!drive->wcache || !ide_id_has_flush_cache(drive->id))
+ if (!drive->wcache || ata_id_flush_enabled(drive->id) == 0)
return;

if (do_idedisk_flushcache(drive))
@@ -937,7 +876,7 @@ static int ide_disk_probe(ide_drive_t *drive);
*/
static void ide_disk_resume(ide_drive_t *drive)
{
- if (idedisk_supports_hpa(drive->id))
+ if (ata_id_hpa_enabled(drive->id))
init_idedisk_capacity(drive);
}

@@ -980,12 +919,12 @@ static ide_driver_t idedisk_driver = {
.shutdown = ide_device_shutdown,
.version = IDEDISK_VERSION,
.media = ide_disk,
- .supports_dsc_overlap = 0,
.do_request = ide_do_rw_disk,
.end_request = ide_end_request,
.error = __ide_error,
#ifdef CONFIG_IDE_PROC_FS
.proc = idedisk_proc,
+ .settings = idedisk_settings,
#endif
};

@@ -994,7 +933,7 @@ static int idedisk_set_doorlock(ide_drive_t *drive, int on)
ide_task_t task;

memset(&task, 0, sizeof(task));
- task.tf.command = on ? WIN_DOORLOCK : WIN_DOORUNLOCK;
+ 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);
@@ -1059,52 +998,28 @@ static int idedisk_getgeo(struct block_device *bdev, struct hd_geometry *geo)
return 0;
}

+static const struct ide_ioctl_devset ide_disk_ioctl_settings[] = {
+{ HDIO_GET_ADDRESS, HDIO_SET_ADDRESS, &ide_devset_address },
+{ HDIO_GET_MULTCOUNT, HDIO_SET_MULTCOUNT, &ide_devset_multcount },
+{ HDIO_GET_NOWERR, HDIO_SET_NOWERR, &ide_devset_nowerr },
+{ HDIO_GET_WCACHE, HDIO_SET_WCACHE, &ide_devset_wcache },

```

[git pull] IDE updates #1

```
+{ HDIO_GET_ACOUSTIC, HDIO_SET_ACOUSTIC, &ide_devset_acoustic },
+{ 0 }
+};
+
static int idedisk_ioctl(struct inode *inode, struct file *file,
unsigned int cmd, unsigned long arg)
{
- unsigned long flags;
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, (*setfunc)(ide_drive_t *, int);
- u8 *val;
-
- switch (cmd) {
- case HDIO_GET_ADDRESS: val = &drive->addressing; goto read_val;
- case HDIO_GET_MULTCOUNT: val = &drive->mult_count; goto read_val;
- case HDIO_GET_NOWERR: val = &drive->nowerr; goto read_val;
- case HDIO_GET_WCACHE: val = &drive->wcache; goto read_val;
- case HDIO_GET_ACOUSTIC: val = &drive->acoustic; goto read_val;
- case HDIO_SET_ADDRESS: setfunc = set_lba_addressing; goto set_val;
- case HDIO_SET_MULTCOUNT: setfunc = set_multcount; goto set_val;
- case HDIO_SET_NOWERR: setfunc = set_nowerr; goto set_val;
- case HDIO_SET_WCACHE: setfunc = write_cache; goto set_val;
- case HDIO_SET_ACOUSTIC: setfunc = set_acoustic; goto set_val;
- }
+ int err;

- return generic_ide_ioctl(drive, file, bdev, cmd, arg);
+ err = ide_setting_ioctl(drive, bdev, cmd, arg, ide_disk_ioctl_settings);
+ if (err != -EOPNOTSUPP)
+ return err;

-read_val:
- mutex_lock(&ide_setting_mtx);
- spin_lock_irqsave(&ide_lock, flags);
- err = *val;
- spin_unlock_irqrestore(&ide_lock, flags);
- mutex_unlock(&ide_setting_mtx);
- return err >= 0 ? put_user(err, (long __user *)arg) : err;
-
-set_val:
- if (bdev != bdev->bd_contains)
- err = -EINVAL;
- else {
- if (!capable(CAP_SYS_ADMIN))
- err = -EACCES - drive->name, __func__, bcount);
- ide_pad_transfer(drive, direction, bcount);
- }
-}
-
```

[git pull] IDE updates #1

```
static void idefloppy_update_buffers(ide_drive_t *drive,
struct ide_atapi_pc *pc)
{
@@ -267,43 +156,6 @@ static void idefloppy_update_buffers(ide_drive_t *drive,
idefloppy_end_request(drive, 1, 0);
}

-/*
- * Generate a new packet command request in front of the request queue, before
- * the current request so that it will be processed immediately, on the next
- * pass through the driver.
- */
-static void idefloppy_queue_pc_head(ide_drive_t *drive, struct ide_atapi_pc *pc,
- struct request *rq)
-{
- struct ide_floppy_obj *floppy = drive->driver_data;
-
- blk_rq_init(NULL, rq);
- rq->buffer = (char *) pc;
- rq->cmd_type = REQ_TYPE_SPECIAL;
- rq->cmd_flags |= REQ_PREEMPT;
- rq->rq_disk = floppy->disk;
- memcpy(rq->cmd, pc->c, 12);
- ide_do_drive_cmd(drive, rq);
-}
-
-static struct ide_atapi_pc *idefloppy_next_pc_storage(ide_drive_t *drive)
-{
- idefloppy_floppy_t *floppy = drive->driver_data;
-
- if (floppy->pc_stack_index == IDEFLOPPY_PC_STACK)
- floppy->pc_stack_index = 0;
- return (&floppy->pc_stack[floppy->pc_stack_index++]);
-}
-
-static struct request *idefloppy_next_rq_storage(ide_drive_t *drive)
-{
- idefloppy_floppy_t *floppy = drive->driver_data;
-
- if (floppy->rq_stack_index == IDEFLOPPY_PC_STACK)
- floppy->rq_stack_index = 0;
- return (&floppy->rq_stack[floppy->rq_stack_index++]);
-}
-
static void ide_floppy_callback(ide_drive_t *drive)
{
idefloppy_floppy_t *floppy = drive->driver_data;
@@ -341,16 +193,9 @@ static void ide_floppy_callback(ide_drive_t *drive)
idefloppy_end_request(drive, uptodate, 0);
}
}
```

[git pull] IDE updates #1

```
-static void idefloppy_init_pc(struct ide_atapi_pc *pc)
-{
- memset(pc, 0, sizeof(*pc));
- pc->buf = pc->pc_buf;
- pc->buf_size = IDEFLOPPY_PC_BUFFER_SIZE;
-}
-
-static void idefloppy_create_request_sense_cmd(struct ide_atapi_pc *pc)
+void ide_floppy_create_request_sense_cmd(struct ide_atapi_pc *pc)
{
- idefloppy_init_pc(pc);
+ ide_init_pc(pc);
pc->c[0] = GPCMD_REQUEST_SENSE;
pc->c[4] = 255;
pc->req_xfer = 18;
@@ -362,14 +207,13 @@ static void idefloppy_create_request_sense_cmd(struct ide_atapi_pc *pc)
*/
static void idefloppy_retry_pc(ide_drive_t *drive)
{
- struct ide_atapi_pc *pc;
- struct request *rq;
+ struct ide_floppy_obj *floppy = drive->driver_data;
+ struct request *rq = &floppy->request_sense_rq;
+ struct ide_atapi_pc *pc = &floppy->request_sense_pc;

(void)ide_read_error(drive);
- pc = idefloppy_next_pc_storage(drive);
- rq = idefloppy_next_rq_storage(drive);
- idefloppy_create_request_sense_cmd(pc);
- idefloppy_queue_pc_head(drive, pc, rq);
+ ide_floppy_create_request_sense_cmd(pc);
+ ide_queue_pc_head(drive, floppy->disk, pc, rq);
}

/* The usual interrupt handler called during a packet command. */
@@ -378,8 +222,8 @@ static ide_startstop_t idefloppy_pc_intr(ide_drive_t *drive)
idefloppy_floppy_t *floppy = drive->driver_data;

return ide_pc_intr(drive, floppy->pc, idefloppy_pc_intr,
- IDEFLOPPY_WAIT_CMD, NULL, idefloppy_update_buffers,
- idefloppy_retry_pc, NULL, ide_floppy_io_buffers);
+ WAIT_FLOPPY_CMD, NULL, idefloppy_update_buffers,
+ idefloppy_retry_pc, NULL, ide_io_buffers);
}

/*
@@ -396,10 +240,9 @@ static int idefloppy_transfer_pc(ide_drive_t *drive)
drive->hwif->tp_ops->output_data(drive, NULL, floppy->pc->c, 12);

/* Timeout for the packet command */
- return IDEFLOPPY_WAIT_CMD;
```

```
+ return WAIT_FLOPPY_CMD;  
}
```

—

```
/*
```

```
* Called as an interrupt (or directly). Wh
```