

[git patches] IDE updates (part 4)

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-10/msg06628.html>

- *From:* Bartłomiej Zolnierkiewicz <bzolnier@xxxxxxxxx>
 - *Date:* Fri, 19 Oct 2007 00:53:55 +0200
-

Hi,

* fix cs5535 and siimage 2.6.23 regression resulting in out-of-bound array access if there are drives on the secondary interface (Benjamin Herrenschmidt <benh@xxxxxxxxxxxxxxxxxxxxxx>)

* more host drivers are switched to always tune PIO: alim15x3, cs5520, cy82c693, opti621 and triflex

* a lot of small cleanups/improvements (which accumulated result in ~450 LOC less in the IDE land)

PS There is still one more update to go.

Please pull from:

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

to receive the following updates:

```
drivers/ide/arm/icside.c | 1 -
drivers/ide/cris/ide-cris.c | 1 +
drivers/ide/ide-disk.c | 8 +-
drivers/ide/ide-dma.c | 12 +-
drivers/ide/ide-io.c | 3 +-
drivers/ide/ide-probe.c | 3 +-
drivers/ide/ide.c | 4 -
drivers/ide/mips/au1xxx-ide.c | 3 -
drivers/ide/pci/aec62xx.c | 97 ++++-----
drivers/ide/pci/alim15x3.c | 30 +++--
drivers/ide/pci/amd74xx.c | 31 +++--
drivers/ide/pci/atiixp.c | 21 +--
drivers/ide/pci/cmd64x.c | 85 +++-----
drivers/ide/pci/cs5520.c | 31 +----
drivers/ide/pci/cs5530.c | 18 +--
drivers/ide/pci/cs5535.c | 15 +-
drivers/ide/pci/cy82c693.c | 18 +--
```

[git patches] IDE updates (part 4)

```
drivers/ide/pci/generic.c | 148 +++++-----
drivers/ide/pci/hpt34x.c | 56 +++-----
drivers/ide/pci/hpt366.c | 376 ++++++-----
drivers/ide/pci/it8213.c | 16 +--
drivers/ide/pci/it821x.c | 22 +--
drivers/ide/pci/jmicron.c | 12 +-
drivers/ide/pci/ns87415.c | 6 +-
drivers/ide/pci/opti621.c | 26 +--
drivers/ide/pci/pdc202xx_new.c | 201 +++++-----
drivers/ide/pci/pdc202xx_old.c | 135 +++++-----
drivers/ide/pci/piix.c | 167 +++++-----
drivers/ide/pci/rz1000.c | 3 +-
drivers/ide/pci/sc1200.c | 19 +--
drivers/ide/pci/scc_pata.c | 26 +--
drivers/ide/pci/serverworks.c | 86 +++++-----
drivers/ide/pci/sgiioc4.c | 11 +-
drivers/ide/pci/siimage.c | 35 +-----
drivers/ide/pci/sis5513.c | 14 +--
drivers/ide/pci/sl82c105.c | 12 +-
drivers/ide/pci/slc90e66.c | 17 +--
drivers/ide/pci/tc86c001.c | 26 +--
drivers/ide/pci/triflex.c | 12 +-
drivers/ide/pci/trm290.c | 9 +-
drivers/ide/pci/via82cxxx.c | 28 +--
drivers/ide/ppc/pmac.c | 1 -
drivers/ide/setup-pci.c | 68 +++++-----
include/linux/ide.h | 58 +++++-----
44 files changed, 760 insertions(+), 1211 deletions(-)
```

Auke Kok (2):

amd74xx: Omit PCI_REVISION_ID read

cmd64x: Use dev->revision

Bartlomiej Zolnierkiewicz (32):

siimage: bump driver version

ide: add DECLARE_GENERIC_PCI_DEV() macro to generic IDE PCI host driver

ide: add IDE_HFLAG_NO_ATAPI_DMA host flag

ide: add IDE_HFLAG_BOOTABLE host flag

ide: add IDE_HFLAG_NO_{DMA,AUTODMA} host flags

ide: remove ->init_setup_dma from ide_pci_device_t (take 2)

ide: add IDE_HFLAG_NO_LBA48 and IDE_HFLAG_NO_LBA48_DMA host flags

pdc202xx_old: remove broken SWDMA support

ide: add ->mwdma_mask and ->swdma_mask to ide_pci_device_t (take 2)

ide: use pci_dev->revision

ide: use I/O ops directly part #2 (take 2)

aec62xx: remove ->init_setup

cmd64x: remove ->init_setup

hpt366: remove ->init_setup

pdc202xx_new: remove ->init_setup

pdc202xx_old: remove ->init_setup

[git patches] IDE updates (part 4)

[git patches] IDE updates (part 4)

scc_pata: remove ->init_setup
serverworks: remove ->init_setup
ide: remove .init_setup from ide_pci_device_t
aec62xx: no need to disable UDMA in ->init_hwif method for ATP850UF
pdc202xx_new: add DECLARE_PDCNEW_DEV() macro
pdc202xx_old: add DECLARE_PDC2026X_DEV() macro
piix: add DECLARE_ICH_DEV() macro
ide: add IDE_HFLAG_ERROR_STOPS_FIFO host flag
ide: add IDE_HFLAG_SERIALIZE host flag
ide: add IDE_HFLAG_LEGACY_IRQS host flag
alim15x3: always tune PIO
cs5520: always tune PIO
cy82c693: always tune PIO
opti621: always tune PIO
triflex: always tune PIO
ide: set drive->autotune in ide_pci_setup_ports()

Benjamin Herrenschmidt (3):

ide: Add ide_get_paired_drive() helper
ide: Fix siimage driver accessing beyond array boundary
ide: Fix cs5535 driver accessing beyond array boundary

```
diff --git a/drivers/ide/arm/icside.c b/drivers/ide/arm/icside.c
index e4875ce..3af33fb 100644
--- a/drivers/ide/arm/icside.c
+++ b/drivers/ide/arm/icside.c
@@ -415,7 +415,6 @@ static void icside_dma_lost_irq(ide_drive_t *drive)
```

```
static void icside_dma_init(ide_hwif_t *hwif)
{
- hwif->atapi_dma = 1;
hwif->mwdma_mask = 7; /* MW0..2 */
hwif->swdma_mask = 7; /* SW0..2 */
```

```
diff --git a/drivers/ide/cris/ide-cris.c b/drivers/ide/cris/ide-cris.c
index 06c75f1..9a96a10 100644
--- a/drivers/ide/cris/ide-cris.c
+++ b/drivers/ide/cris/ide-cris.c
@@ -805,6 +805,7 @@ init_e100_ide (void)
```

```
hwif->dma_host_on = &cris_dma_on;
hwif->dma_off_quietly = &cris_dma_off;
hwif->cbl = ATA_CBL_PATA40;
+ hwif->host_flags |= IDE_HFLAG_NO_ATAPI_DMA;
hwif->pio_mask = ATA_PIO4,
hwif->drives[0].autotune = 1;
hwif->drives[1].autotune = 1;
```

```
diff --git a/drivers/ide/ide-disk.c b/drivers/ide/ide-disk.c
index 92177ca..2722d91 100644
--- a/drivers/ide/ide-disk.c
+++ b/drivers/ide/ide-disk.c
```

[git patches] IDE updates (part 4)

```
@@ -169,7 +169,7 @@ static ide_startstop_t __ide_do_rw_disk(ide_drive_t *drive, struct request *rq,
nsectors.all = (u16) rq->nr_sectors;

- if (hwif->no_lba48_dma && lba48 && dma) {
+ if ((hwif->host_flags & IDE_HFLAG_NO_LBA48_DMA) && lba48 && dma) {
if (block + rq->nr_sectors > 1ULL << 28)
dma = 0;
else
@@ -856,7 +856,7 @@ static int set_lba_addressing(ide_drive_t *drive, int arg)

drive->addressing = 0;

- if (HWIF(drive)->no_lba48)
+ if (drive->hwif->host_flags & IDE_HFLAG_NO_LBA48)
return 0;

if (!idedisk_supports_lba48(drive->id))
@@ -889,6 +889,7 @@ static inline void idedisk_add_settings(ide_drive_t *drive) { ; }

static void idedisk_setup (ide_drive_t *drive)
{
+ ide_hwif_t *hwif = drive->hwif;
struct hd_driveid *id = drive->id;
unsigned long long capacity;

@@ -909,7 +910,6 @@ static void idedisk_setup (ide_drive_t *drive)
(void)set_lba_addressing(drive, 1);

if (drive->addressing == 1) {
- ide_hwif_t *hwif = HWIF(drive);
int max_s = 2048;

if (max_s > hwif->rsize)
@@ -932,7 +932,7 @@ static void idedisk_setup (ide_drive_t *drive)
drive->capacity64 = 1ULL << 28;
}

- if (drive->hwif->no_lba48_dma && drive->addressing) {
+ if ((hwif->host_flags & IDE_HFLAG_NO_LBA48_DMA) && drive->addressing) {
if (drive->capacity64 > 1ULL << 28) {
printk(KERN_INFO "%s: cannot use LBA48 DMA - PIO mode will"
" be used for accessing sectors > %u\n",
diff --git a/drivers/ide/ide-dma.c b/drivers/ide/ide-dma.c
index bc57ce6..80b4f17 100644
--- a/drivers/ide/ide-dma.c
+++ b/drivers/ide/ide-dma.c
@@ -338,8 +338,10 @@ static int config_drive_for_dma (ide_drive_t *drive)
ide_hwif_t *hwif = drive->hwif;
struct hd_driveid *id = drive->id;
```

[git patches] IDE updates (part 4)

```
- if (drive->media != ide_disk && hwif->ataapi_dma == 0)
- return 0;
+ if (drive->media != ide_disk) {
+ if (hwif->host_flags & IDE_HFLAG_NO_ATAPI_DMA)
+ return -1;
+ }

/*
* Enable DMA on any drive that has
@@ -726,8 +728,10 @@ u8 ide_find_dma_mode(ide_drive_t *drive, u8 req_mode)
int x, i;
u8 mode = 0;

- if (drive->media != ide_disk && hwif->ataapi_dma == 0)
- return 0;
+ if (drive->media != ide_disk) {
+ if (hwif->host_flags & IDE_HFLAG_NO_ATAPI_DMA)
+ return 0;
+ }

for (i = 0; i < ARRAY_SIZE(xfer_mode_bases); i++) {
if (req_mode < xfer_mode_bases[i])
diff --git a/drivers/ide/ide-io.c b/drivers/ide/ide-io.c
index ec835e3..5c8b008 100644
--- a/drivers/ide/ide-io.c
+++ b/drivers/ide/ide-io.c
@@ -484,7 +484,8 @@ static ide_startstop_t ide_ata_error(ide_drive_t *drive, struct request *rq, u8
}
}

- if ((stat & DRQ_STAT) && rq_data_dir(rq) == READ && hwif->err_stops_fifo == 0)
+ if ((stat & DRQ_STAT) && rq_data_dir(rq) == READ &&
+ (hwif->host_flags & IDE_HFLAG_ERROR_STOPS_FIFO) == 0)
try_to_flush_leftover_data(drive);

if (rq->errors >= ERROR_MAX || blk_noretry_request(rq)) {
diff --git a/drivers/ide/ide-probe.c b/drivers/ide/ide-probe.c
index 3c945d6..e294c74 100644
--- a/drivers/ide/ide-probe.c
+++ b/drivers/ide/ide-probe.c
@@ -951,7 +951,8 @@ static int ide_init_queue(ide_drive_t *drive)
blk_queue_segment_boundary(q, 0xffff);

if (!hwif->rqsizes) {
- if (hwif->no_lba48 || hwif->no_lba48_dma)
+ if ((hwif->host_flags & IDE_HFLAG_NO_LBA48) ||
+ (hwif->host_flags & IDE_HFLAG_NO_LBA48_DMA))
hwif->rqsizes = 256;
else
hwif->rqsizes = 65536;
diff --git a/drivers/ide/ide.c b/drivers/ide/ide.c
```

[git patches] IDE updates (part 4)

```
index 5b09066..961e6c8 100644
--- a/drivers/ide/ide.c
+++ b/drivers/ide/ide.c
@@ -134,8 +134,6 @@ static void init_hwif_data(ide_hwif_t *hwif, unsigned int index)

hwif->bus_state = BUSSTATE_ON;

- hwif->atapi_dma = 0; /* disable all atapi dma */
-
init_completion(&hwif->gendev_rel_comp);

default_hwif_iops(hwif);
@@ -379,7 +377,6 @@ static void ide_hwif_restore(ide_hwif_t *hwif, ide_hwif_t *tmp_hwif)

hwif->pio_mask = tmp_hwif->pio_mask;

- hwif->atapi_dma = tmp_hwif->atapi_dma;
hwif->ultra_mask = tmp_hwif->ultra_mask;
hwif->mwdma_mask = tmp_hwif->mwdma_mask;
hwif->swdma_mask = tmp_hwif->swdma_mask;
@@ -440,7 +437,6 @@ static void ide_hwif_restore(ide_hwif_t *hwif, ide_hwif_t *tmp_hwif)

hwif->mmio = tmp_hwif->mmio;
hwif->rsize = tmp_hwif->rsize;
- hwif->no_lba48 = tmp_hwif->no_lba48;

#ifdef CONFIG_BLK_DEV_IDECS
hwif->irq = tmp_hwif->irq;
diff --git a/drivers/ide/mips/au1xxx-ide.c b/drivers/ide/mips/au1xxx-ide.c
index 47c035a..2f322d7 100644
--- a/drivers/ide/mips/au1xxx-ide.c
+++ b/drivers/ide/mips/au1xxx-ide.c
@@ -699,9 +699,6 @@ static int au_ide_probe(struct device *dev)
hwif->dma_host_on = &auide_dma_host_on;
hwif->dma_lost_irq = &auide_dma_lost_irq;
hwif->ide_dma_on = &auide_dma_on;
-
- hwif->atapi_dma = 1;
-
#else /* !CONFIG_BLK_DEV_IDE_AU1XXX_MDMA2_DBDMA */
hwif->channel = 0;
hwif->hold = 1;
diff --git a/drivers/ide/pci/aec62xx.c b/drivers/ide/pci/aec62xx.c
index 3a4c2c2..b3dc12a 100644
--- a/drivers/ide/pci/aec62xx.c
+++ b/drivers/ide/pci/aec62xx.c
@@ -1,5 +1,5 @@
/*
- * linux/drivers/ide/pci/aec62xx.c Version 0.25 Aug 1, 2007
+ * linux/drivers/ide/pci/aec62xx.c Version 0.26 Sep 1, 2007
*

```

[git patches] IDE updates (part 4)

```
* Copyright (C) 1999–2002 Andre Hedrick <andre@xxxxxxxxxxxxxx>
* Copyright (C) 2007 MontaVista Software, Inc. <source@xxxxxxxxxxxx>
@@ -184,34 +184,23 @@ static unsigned int __devinit init_chipset_aec62xx(struct pci_dev *dev, const ch
static void __devinit init_hwif_aec62xx(ide_hwif_t *hwif)
{
struct pci_dev *dev = hwif->pci_dev;
- u8 reg54 = 0, mask = hwif->channel ? 0xf0 : 0x0f;
- unsigned long flags;

hwif->set_pio_mode = &aec_set_pio_mode;

- if (dev->device == PCI_DEVICE_ID_ARTOP_ATP850UF) {
- if(hwif->mate)
- hwif->mate->serialized = hwif->serialized = 1;
+ if (dev->device == PCI_DEVICE_ID_ARTOP_ATP850UF)
hwif->set_dma_mode = &aec6210_set_mode;
- } else
+ else
hwif->set_dma_mode = &aec6260_set_mode;

- hwif->drives[0].autotune = hwif->drives[1].autotune = 1;
-
if (hwif->dma_base == 0)
return;

- hwif->ultra_mask = hwif->cds->udma_mask;
- hwif->mwdma_mask = 0x07;
-
hwif->dma_lost_irq = &aec62xx_dma_lost_irq;

- if (dev->device == PCI_DEVICE_ID_ARTOP_ATP850UF) {
- spin_lock_irqsave(&ide_lock, flags);
- pci_read_config_byte (dev, 0x54, &reg54);
- pci_write_config_byte(dev, 0x54, (reg54 & ~mask));
- spin_unlock_irqrestore(&ide_lock, flags);
- } else if (hwif->cbl != ATA_CBL_PATA40_SHORT) {
+ if (dev->device == PCI_DEVICE_ID_ARTOP_ATP850UF)
+ return;
+
+ if (hwif->cbl != ATA_CBL_PATA40_SHORT) {
u8 ata66 = 0, mask = hwif->channel ? 0x02 : 0x01;

pci_read_config_byte(hwif->pci_dev, 0x49, &ata66);
@@ -220,73 +209,53 @@ static void __devinit init_hwif_aec62xx(ide_hwif_t *hwif)
}
}

-static int __devinit init_setup_aec62xx(struct pci_dev *dev, ide_pci_device_t *d)
-{
- return ide_setup_pci_device(dev, d);
-}

```

[git patches] IDE updates (part 4)

```
-
-static int __devinit init_setup_aec6x80(struct pci_dev *dev, ide_pci_device_t *d)
-{
- unsigned long dma_base = pci_resource_start(dev, 4);
-
- if (inb(dma_base + 2) & 0x10) {
- d->name = (dev->device == PCI_DEVICE_ID_ARTOP_ATP865R) ?
- "AEC6880R" : "AEC6880";
- d->udma_mask = 0x7f; /* udma0-6 */
- }
-
- return ide_setup_pci_device(dev, d);
-}
-
static ide_pci_device_t aec62xx_chipsets[] __devinitdata = {
 { /* 0 */
 .name = "AEC6210",
 - .init_setup = init_setup_aec62xx,
 .init_chipset = init_chipset_aec62xx,
 .init_hwif = init_hwif_aec62xx,
 - .autodma = AUTODMA,
 .enablebits = {{0x4a,0x02,0x02}, {0x4a,0x04,0x04}},
 - .bootable = OFF_BOARD,
 + .host_flags = IDE_HFLAG_SERIALIZE |
 + IDE_HFLAG_NO_ATAPI_DMA |
 + IDE_HFLAG_OFF_BOARD,
 .pio_mask = ATA_PIO4,
 - .udma_mask = 0x07, /* udma0-2 */
 + .mwdma_mask = ATA_MWDMA2,
 + .udma_mask = ATA_UDMA2,
 }, { /* 1 */
 .name = "AEC6260",
 - .init_setup = init_setup_aec62xx,
 .init_chipset = init_chipset_aec62xx,
 .init_hwif = init_hwif_aec62xx,
 - .autodma = NOAUTODMA,
 - .bootable = OFF_BOARD,
 + .host_flags = IDE_HFLAG_NO_ATAPI_DMA | IDE_HFLAG_NO_AUTODMA |
 + IDE_HFLAG_OFF_BOARD,
 .pio_mask = ATA_PIO4,
 - .udma_mask = 0x1f, /* udma0-4 */
 + .mwdma_mask = ATA_MWDMA2,
 + .udma_mask = ATA_UDMA4,
 }, { /* 2 */
 .name = "AEC6260R",
 - .init_setup = init_setup_aec62xx,
 .init_chipset = init_chipset_aec62xx,
 .init_hwif = init_hwif_aec62xx,
 - .autodma = AUTODMA,
 .enablebits = {{0x4a,0x02,0x02}, {0x4a,0x04,0x04}},
 - .bootable = NEVER_BOARD,

```

[git patches] IDE updates (part 4)

```
+ .host_flags = IDE_HFLAG_NO_ATAPI_DMA,
.pio_mask = ATA_PIO4,
- .udma_mask = 0x1f, /* udma0-4 */
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA4,
},{ /* 3 */
.name = "AEC6280",
- .init_setup = init_setup_aec6x80,
.init_chipset = init_chipset_aec62xx,
.init_hwif = init_hwif_aec62xx,
- .autodma = AUTODMA,
- .bootable = OFF_BOARD,
+ .host_flags = IDE_HFLAG_NO_ATAPI_DMA | IDE_HFLAG_OFF_BOARD,
.pio_mask = ATA_PIO4,
- .udma_mask = 0x3f, /* udma0-5 */
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA5,
},{ /* 4 */
.name = "AEC6280R",
- .init_setup = init_setup_aec6x80,
.init_chipset = init_chipset_aec62xx,
.init_hwif = init_hwif_aec62xx,
- .autodma = AUTODMA,
.enablebits = {{0x4a,0x02,0x02}, {0x4a,0x04,0x04}},
- .bootable = OFF_BOARD,
+ .host_flags = IDE_HFLAG_NO_ATAPI_DMA | IDE_HFLAG_OFF_BOARD,
.pio_mask = ATA_PIO4,
- .udma_mask = 0x3f, /* udma0-5 */
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA5,
}
};
```

```
@@ -304,9 +273,21 @@ static ide_pci_device_t aec62xx_chipsets[] __devinitdata = {

static int __devinit aec62xx_init_one(struct pci_dev *dev, const struct pci_device_id *id)
{
- ide_pci_device_t d = aec62xx_chipsets[id->driver_data];
+ ide_pci_device_t d;
+ u8 idx = id->driver_data;
+
+ d = aec62xx_chipsets[idx];
+
+ if (idx == 3 || idx == 4) {
+ unsigned long dma_base = pci_resource_start(dev, 4);
+
+ if (inb(dma_base + 2) & 0x10) {
+ d.name = (idx == 4) ? "AEC6880R" : "AEC6880";
+ d.udma_mask = ATA_UDMA6;
+ }
+ }
```

[git patches] IDE updates (part 4)

```
- return d.init_setup(dev, &d);
+ return ide_setup_pci_device(dev, &d);
}

static const struct pci_device_id aec62xx_pci_tbl[] = {
diff --git a/drivers/ide/pci/alim15x3.c b/drivers/ide/pci/alim15x3.c
index 31d4e50..8ee2b48 100644
--- a/drivers/ide/pci/alim15x3.c
+++ b/drivers/ide/pci/alim15x3.c
@@ -1,5 +1,5 @@
/*
- * linux/drivers/ide/pci/alim15x3.c Version 0.26 Jul 14 2007
+ * linux/drivers/ide/pci/alim15x3.c Version 0.27 Aug 27 2007
*
* Copyright (C) 1998-2000 Michel Aubry, Maintainer
* Copyright (C) 1998-2000 Andrzej Krzysztofowicz, Maintainer
@@ -665,34 +665,29 @@ static void __devinit init_hwif_common_ali15x3 (ide_hwif_t *hwif)
hwif->udma_filter = &ali_udma_filter;

/* don't use LBA48 DMA on ALi devices before rev 0xC5 */
- hwif->no_lba48_dma = (m5229_revision <= 0xC4) ? 1 : 0;
+ if (m5229_revision <= 0xC4)
+ hwif->host_flags |= IDE_HFLAG_NO_LBA48_DMA;

- if (!hwif->dma_base) {
- hwif->drives[0].autotune = 1;
- hwif->drives[1].autotune = 1;
+ if (hwif->dma_base == 0)
return;
- }

/*
* check in ->init_dma guarantees m5229_revision >= 0x20 here
*/

- if (m5229_revision > 0x20)
- hwif->atapi_dma = 1;
+ if (m5229_revision == 0x20)
+ hwif->host_flags |= IDE_HFLAG_NO_ATAPI_DMA;

if (m5229_revision <= 0x20)
hwif->ultra_mask = 0x00; /* no udma */
else if (m5229_revision < 0xC2)
- hwif->ultra_mask = 0x07; /* udma0-2 */
+ hwif->ultra_mask = ATA_UDMA2;
else if (m5229_revision == 0xC2 || m5229_revision == 0xC3)
- hwif->ultra_mask = 0x1f; /* udma0-4 */
+ hwif->ultra_mask = ATA_UDMA4;
else if (m5229_revision == 0xC4)
- hwif->ultra_mask = 0x3f; /* udma0-5 */
```

[git patches] IDE updates (part 4)

```
+ hwif->ultra_mask = ATA_UDMA5;
else
- hwif->ultra_mask = 0x7f; /* udma0-6 */
-
- hwif->mwdma_mask = 0x07;
- hwif->swdma_mask = 0x07;
+ hwif->ultra_mask = ATA_UDMA6;

hwif->dma_setup = &ali15x3_dma_setup;

@@ -776,9 +771,10 @@ static ide_pci_device_t ali15x3_chipset __devinitdata = {
.init_chipset = init_chipset_ali15x3,
.init_hwif = init_hwif_ali15x3,
.init_dma = init_dma_ali15x3,
- .autodma = AUTODMA,
- .bootable = ON_BOARD,
+ .host_flags = IDE_HFLAG_BOOTABLE,
.pio_mask = ATA_PIO5,
+ .swdma_mask = ATA_SWDMA2,
+ .mwdma_mask = ATA_MWDMA2,
};

/**
diff --git a/drivers/ide/pci/amd74xx.c b/drivers/ide/pci/amd74xx.c
index 3bf3d93..7cafefb 100644
--- a/drivers/ide/pci/amd74xx.c
+++ b/drivers/ide/pci/amd74xx.c
@@ -233,7 +233,6 @@ static unsigned int __devinit init_chipset_amd74xx(struct pci_dev *dev, const ch
* Print the boot message.
*/

- pci_read_config_byte(dev, PCI_REVISION_ID, &t);
printk(KERN_INFO "%s: %s (rev %02x) UDMA%s controller\n",
amd_chipset->name, pci_name(dev), dev->revision,
amd_dma[fls(amd_config->udma_mask) - 1]);
@@ -254,18 +253,14 @@ static void __devinit init_hwif_amd74xx(ide_hwif_t *hwif)
for (i = 0; i < 2; i++) {
hwif->drives[i].io_32bit = 1;
hwif->drives[i].unmask = 1;
- hwif->drives[i].autotune = 1;
}

if (!hwif->dma_base)
return;

- hwif->atapi_dma = 1;
-
hwif->ultra_mask = amd_config->udma_mask;
- hwif->mwdma_mask = 0x07;
- if ((amd_config->flags & AMD_BAD_SWDMA) == 0)
- hwif->swdma_mask = 0x07;
```

[git patches] IDE updates (part 4)

```
+ if (amd_config->flags & AMD_BAD_SWDMA)
+ hwif->swdma_mask = 0x00;

if (hwif->cbl != ATA_CBL_PATA40_SHORT) {
if ((amd_80w >> hwif->channel) & 1)
@@ -280,13 +275,14 @@ static void __devinit init_hwif_amd74xx(ide_hwif_t *hwif)
.name = name_str, \
.init_chipset = init_chipset_amd74xx, \
.init_hwif = init_hwif_amd74xx, \
- .autodma = AUTODMA, \
.enablebits = {{0x40,0x02,0x02}, {0x40,0x01,0x01}}, \
- .bootable = ON_BOARD, \
- .host_flags = IDE_HFLAG_PIO_NO_BLACKLIST \
- | IDE_HFLAG_PIO_NO_DOWNGRADE \
- | IDE_HFLAG_POST_SET_MODE, \
+ .host_flags = IDE_HFLAG_PIO_NO_BLACKLIST | \
+ IDE_HFLAG_PIO_NO_DOWNGRADE | \
+ IDE_HFLAG_POST_SET_MODE | \
+ IDE_HFLAG_BOOTABLE, \
.pio_mask = ATA_PIO5, \
+ .swdma_mask = ATA_SWDMA2, \
+ .mwdma_mask = ATA_MWDMA2, \
}

#define DECLARE_NV_DEV(name_str) \
@@ -294,13 +290,14 @@ static void __devinit init_hwif_amd74xx(ide_hwif_t *hwif)
.name = name_str, \
.init_chipset = init_chipset_amd74xx, \
.init_hwif = init_hwif_amd74xx, \
- .autodma = AUTODMA, \
.enablebits = {{0x50,0x02,0x02}, {0x50,0x01,0x01}}, \
- .bootable = ON_BOARD, \
- .host_flags = IDE_HFLAG_PIO_NO_BLACKLIST \
- | IDE_HFLAG_PIO_NO_DOWNGRADE \
- | IDE_HFLAG_POST_SET_MODE, \
+ .host_flags = IDE_HFLAG_PIO_NO_BLACKLIST | \
+ IDE_HFLAG_PIO_NO_DOWNGRADE | \
+ IDE_HFLAG_POST_SET_MODE | \
+ IDE_HFLAG_BOOTABLE, \
.pio_mask = ATA_PIO5, \
+ .swdma_mask = ATA_SWDMA2, \
+ .mwdma_mask = ATA_MWDMA2, \
}

static ide_pci_device_t amd74xx_chipsets[] __devinitdata = {
diff --git a/drivers/ide/pci/atiixp.c b/drivers/ide/pci/atiixp.c
index 446900d..3078430 100644
--- a/drivers/ide/pci/atiixp.c
+++ b/drivers/ide/pci/atiixp.c
@@ -172,21 +172,12 @@ static void __devinit init_hwif_atiixp(ide_hwif_t *hwif)
u8 ch = hwif->channel;
```

[git patches] IDE updates (part 4)

```
struct pci_dev *pdev = hwif->pci_dev;

- if (!hwif->irq)
- hwif->irq = ch ? 15 : 14;
-
hwif->set_pio_mode = &atiixp_set_pio_mode;
hwif->set_dma_mode = &atiixp_set_dma_mode;
- hwif->drives[0].autotune = 1;
- hwif->drives[1].autotune = 1;

if (!hwif->dma_base)
return;

- hwif->atapi_dma = 1;
- hwif->ultra_mask = 0x3f;
- hwif->mwdma_mask = 0x07;
-
pci_read_config_byte(pdev, ATIIXP_IDE_UDMA_MODE + ch, &udma_mode);

if ((udma_mode & 0x07) >= 0x04 || (udma_mode & 0x70) >= 0x40)
@@ -203,18 +194,20 @@ static ide_pci_device_t atiixp_pci_info[] __devinitdata = {
{ /* 0 */
.name = "ATIIXP",
.init_hwif = init_hwif_atiixp,
- .autodma = AUTODMA,
.enablebits = {{0x48,0x01,0x00}, {0x48,0x08,0x00}},
- .bootable = ON_BOARD,
+ .host_flags = IDE_HFLAG_LEGACY_IRQS | IDE_HFLAG_BOOTABLE,
.pio_mask = ATA_PIO4,
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA5,
},{ /* 1 */
.name = "SB600_PATA",
.init_hwif = init_hwif_atiixp,
- .autodma = AUTODMA,
.enablebits = {{0x48,0x01,0x00}, {0x00,0x00,0x00}},
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_SINGLE,
+ .host_flags = IDE_HFLAG_SINGLE | IDE_HFLAG_LEGACY_IRQS |
+ IDE_HFLAG_BOOTABLE,
.pio_mask = ATA_PIO4,
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA5,
},
};

diff --git a/drivers/ide/pci/cmd64x.c b/drivers/ide/pci/cmd64x.c
index f3d3bde..adee2ef 100644
--- a/drivers/ide/pci/cmd64x.c
+++ b/drivers/ide/pci/cmd64x.c
@@ -439,11 +439,8 @@ static unsigned int __devinit init_chipset_cmd64x(struct pci_dev *dev, const cha
```

[git patches] IDE updates (part 4)

```
u8 mrdmode = 0;

if (dev->device == PCI_DEVICE_ID_CMD_646) {
- u8 rev = 0;

- pci_read_config_byte(dev, PCI_REVISION_ID, &rev);
-
- switch (rev) {
+ switch (dev->revision) {
case 0x07:
case 0x05:
printk("%s: UltraDMA capable\n", name);
@@ -505,22 +502,13 @@ static u8 __devinit ata66_cmd64x(ide_hwif_t *hwif)
static void __devinit init_hwif_cmd64x(ide_hwif_t *hwif)
{
struct pci_dev *dev = hwif->pci_dev;
- u8 rev = 0;
-
- pci_read_config_byte(dev, PCI_REVISION_ID, &rev);

hwif->set_pio_mode = &cmd64x_set_pio_mode;
hwif->set_dma_mode = &cmd64x_set_dma_mode;

- hwif->drives[0].autotune = hwif->drives[1].autotune = 1;
-
if (!hwif->dma_base)
return;

- hwif->atapi_dma = 1;
- hwif->mwdma_mask = 0x07;
- hwif->ultra_mask = hwif->cds->udma_mask;
-
/*
* UltraDMA only supported on PCI646U and PCI646U2, which
* correspond to revisions 0x03, 0x05 and 0x07 respectively.
@@ -533,7 +521,7 @@ static void __devinit init_hwif_cmd64x(ide_hwif_t *hwif)
*
* So we only do UltraDMA on revision 0x05 and 0x07 chipsets.
*/
- if (dev->device == PCI_DEVICE_ID_CMD_646 && rev < 5)
+ if (dev->device == PCI_DEVICE_ID_CMD_646 && dev->revision < 5)
hwif->ultra_mask = 0x00;

if (hwif->cbl != ATA_CBL_PATA40_SHORT)
@@ -548,10 +536,10 @@ static void __devinit init_hwif_cmd64x(ide_hwif_t *hwif)
break;
case PCI_DEVICE_ID_CMD_646:
hwif->chipset = ide_cmd646;
- if (rev == 0x01) {
+ if (dev->revision == 0x01) {
hwif->ide_dma_end = &cmd646_1_ide_dma_end;
```

[git patches] IDE updates (part 4)

```
break;
- } else if (rev >= 0x03)
+ } else if (dev->revision >= 0x03)
goto alt_irq_bits;
/* fall thru */
default:
@@ -561,80 +549,61 @@ static void __devinit init_hwif_cmd64x(ide_hwif_t *hwif)
}
}

-static int __devinit init_setup_cmd64x(struct pci_dev *dev, ide_pci_device_t *d)
-{
- return ide_setup_pci_device(dev, d);
-}
-
-static int __devinit init_setup_cmd646(struct pci_dev *dev, ide_pci_device_t *d)
-{
- /*
-  * The original PCI0646 didn't have the primary channel enable bit,
-  * it appeared starting with PCI0646U (i.e. revision ID 3).
-  */
- if (dev->revision < 3)
- d->enablebits[0].reg = 0;
-
- return ide_setup_pci_device(dev, d);
-}

static ide_pci_device_t cmd64x_chipsets[] __devinitdata = {
{ /* 0 */
.name = "CMD643",
- .init_setup = init_setup_cmd64x,
.init_chipset = init_chipset_cmd64x,
.init_hwif = init_hwif_cmd64x,
- .autodma = AUTODMA,
.enablebits = {{0x00,0x00,0x00}, {0x51,0x08,0x08}},
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_ABUSE_PREFETCH,
+ .host_flags = IDE_HFLAG_ABUSE_PREFETCH | IDE_HFLAG_BOOTABLE,
.pio_mask = ATA_PIO5,
+ .mwdma_mask = ATA_MWDMA2,
.udma_mask = 0x00, /* no udma */
}, { /* 1 */
.name = "CMD646",
- .init_setup = init_setup_cmd646,
.init_chipset = init_chipset_cmd64x,
.init_hwif = init_hwif_cmd64x,
- .autodma = AUTODMA,
.enablebits = {{0x51,0x04,0x04}, {0x51,0x08,0x08}},
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_ABUSE_PREFETCH,
+ .host_flags = IDE_HFLAG_ABUSE_PREFETCH | IDE_HFLAG_BOOTABLE,
```

[git patches] IDE updates (part 4)

```
.pio_mask = ATA_PIO5,
- .udma_mask = 0x07, /* udma0-2 */
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA2,
},{ /* 2 */
.name = "CMD648",
- .init_setup = init_setup_cmd64x,
.init_chipset = init_chipset_cmd64x,
.init_hwif = init_hwif_cmd64x,
- .autodma = AUTODMA,
.enablebits = {{0x51,0x04,0x04}, {0x51,0x08,0x08}},
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_ABUSE_PREFETCH,
+ .host_flags = IDE_HFLAG_ABUSE_PREFETCH | IDE_HFLAG_BOOTABLE,
.pio_mask = ATA_PIO5,
- .udma_mask = 0x1f, /* udma0-4 */
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA4,
},{ /* 3 */
.name = "CMD649",
- .init_setup = init_setup_cmd64x,
.init_chipset = init_chipset_cmd64x,
.init_hwif = init_hwif_cmd64x,
- .autodma = AUTODMA,
.enablebits = {{0x51,0x04,0x04}, {0x51,0x08,0x08}},
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_ABUSE_PREFETCH,
+ .host_flags = IDE_HFLAG_ABUSE_PREFETCH | IDE_HFLAG_BOOTABLE,
.pio_mask = ATA_PIO5,
- .udma_mask = 0x3f, /* udma0-5 */
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA5,
}
};

_/*
- * We may have to modify enablebits for PCI0646, so we'd better pass
- * a local copy of the ide_pci_device_t structure down the call chain...
- */
static int __devinit cmd64x_init_one(struct pci_dev *dev, const struct pci_device_id *id)
{
- ide_pci_device_t d = cmd64x_chipsets[id->driver_data];
+ ide_pci_device_t d;
+ u8 idx = id->driver_data;
+
+ d = cmd64x_chipsets[idx];
+
+ /*
+ * The original PCI0646 didn't have the primary channel enable bit,
+ * it appeared starting with PCI0646U (i.e. revision ID 3).
+ */
}
```

[git patches] IDE updates (part 4)

```
+ if (idx == 1 && dev->revision < 3)
+ d.enablebits[0].reg = 0;

- return d.init_setup(dev, &d);
+ return ide_setup_pci_device(dev, &d);
}

static const struct pci_device_id cmd64x_pci_tbl[] = {
diff --git a/drivers/ide/pci/cs5520.c b/drivers/ide/pci/cs5520.c
index a8bf494..aa98e81 100644
--- a/drivers/ide/pci/cs5520.c
+++ b/drivers/ide/pci/cs5520.c
@@ -106,18 +106,6 @@ static void cs5520_set_dma_mode(ide_drive_t *drive, const u8 speed)
}

/*
- * We provide a callback for our nonstandard DMA location
- */
-
- static void __devinit cs5520_init_setup_dma(struct pci_dev *dev, ide_pci_device_t *d, ide_hwif_t *hwif)
- {
- unsigned long bmid = pci_resource_start(dev, 2); /* Not the usual 4 */
- if (hwif->mate && hwif->mate->dma_base) /* Second channel at primary + 8 */
- bmid += 8;
- ide_setup_dma(hwif, bmid, 8);
- }
-
- /*
* We wrap the DMA activate to set the vdma flag. This is needed
* so that the IDE DMA layer issues PIO not DMA commands over the
* DMA channel
@@ -125,6 +113,7 @@ static void __devinit cs5520_init_setup_dma(struct pci_dev *dev, ide_pci_device_t

static int cs5520_dma_on(ide_drive_t *drive)
{
+ /* ATAPI is harder so leave it for now */
drive->vdma = 1;
return 0;
}
@@ -134,29 +123,21 @@ static void __devinit init_hwif_cs5520(ide_hwif_t *hwif)
hwif->set_pio_mode = &cs5520_set_pio_mode;
hwif->set_dma_mode = &cs5520_set_dma_mode;

- if (hwif->dma_base == 0) {
- hwif->drives[1].autotune = hwif->drives[0].autotune = 1;
+ if (hwif->dma_base == 0)
return;
- }

hwif->ide_dma_on = &cs5520_dma_on;
-

```

[git patches] IDE updates (part 4)

```
- /* ATAPI is harder so leave it for now */
- hwif->atapi_dma = 0;
- hwif->ultra_mask = 0;
- hwif->swdma_mask = 0;
- hwif->mwdma_mask = 0;
}
```

```
#define DECLARE_CS_DEV(name_str) \
{ \
.name = name_str, \
.init_setup_dma = cs5520_init_setup_dma, \
.init_hwif = init_hwif_cs5520, \
.autodma = AUTODMA, \
.bootable = ON_BOARD, \
.host_flags = IDE_HFLAG_ISA_PORTS | \
IDE_HFLAG_VDMA, \
+ IDE_HFLAG_CS5520 | \
+ IDE_HFLAG_VDMA | \
+ IDE_HFLAG_NO_ATAPI_DMA | \
+ IDE_HFLAG_BOOTABLE, \
.pio_mask = ATA_PIO4, \
}
```

```
diff --git a/drivers/ide/pci/cs5530.c b/drivers/ide/pci/cs5530.c
```

```
index 0d23b8a..ba0c6eb 100644
```

```
--- a/drivers/ide/pci/cs5530.c
```

```
+++ b/drivers/ide/pci/cs5530.c
```

```
@@ -245,9 +245,6 @@ static void __devinit init_hwif_cs5530 (ide_hwif_t *hwif)
unsigned long basereg;
u32 d0_timings;
```

```
- if (hwif->mate)
- hwif->serialized = hwif->mate->serialized = 1;
-
hwif->set_pio_mode = &cs5530_set_pio_mode;
hwif->set_dma_mode = &cs5530_set_dma_mode;
```

```
@@ -258,16 +255,9 @@ static void __devinit init_hwif_cs5530 (ide_hwif_t *hwif)
if (CS5530_BAD_PIO(inl(basereg + 8)))
outl(cs5530_pio_timings[(d0_timings >> 31) & 1][0], basereg + 8);
```

```
- hwif->drives[0].autotune = 1;
- hwif->drives[1].autotune = 1;
-
if (hwif->dma_base == 0)
return;
```

```
- hwif->atapi_dma = 1;
- hwif->ultra_mask = 0x07;
- hwif->mwdma_mask = 0x07;
-
```

[git patches] IDE updates (part 4)

```
hwif->udma_filter = cs5530_udma_filter;
}

@@ -275,10 +265,12 @@ static ide_pci_device_t cs5530_chipset __devinitdata = {
.name = "CS5530",
.init_chipset = init_chipset_cs5530,
.init_hwif = init_hwif_cs5530,
- .autodma = AUTODMA,
- .bootable = ON_BOARD,
+ .host_flags = IDE_HFLAG_SERIALIZE |
+ IDE_HFLAG_POST_SET_MODE |
+ IDE_HFLAG_BOOTABLE,
.pio_mask = ATA_PIO4,
- .host_flags = IDE_HFLAG_POST_SET_MODE,
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA2,
};

static int __devinit cs5530_init_one(struct pci_dev *dev, const struct pci_device_id *id)
diff --git a/drivers/ide/pci/cs5535.c b/drivers/ide/pci/cs5535.c
index e4891a1..5ac82ff 100644
--- a/drivers/ide/pci/cs5535.c
+++ b/drivers/ide/pci/cs5535.c
@@ -84,7 +84,7 @@ static void cs5535_set_speed(ide_drive_t *drive, const u8 speed)

/* Set the PIO timings */
if ((speed & XFER_MODE) == XFER_PIO) {
- ide_drive_t *pair = &drive->hwif->drives[drive->dn ^ 1];
+ ide_drive_t *pair = ide_get_paired_drive(drive);
u8 cmd, pioa;

cmd = pioa = speed - XFER_PIO_0;
@@ -180,25 +180,20 @@ static void __devinit init_hwif_cs5535(ide_hwif_t *hwif)
hwif->set_pio_mode = &cs5535_set_pio_mode;
hwif->set_dma_mode = &cs5535_set_dma_mode;

- hwif->drives[1].autotune = hwif->drives[0].autotune = 1;
-
if (hwif->dma_base == 0)
return;

- hwif->atapi_dma = 1;
- hwif->ultra_mask = 0x1F;
- hwif->mwdma_mask = 0x07;
-
hwif->cbl = cs5535_cable_detect(hwif->pci_dev);
}

static ide_pci_device_t cs5535_chipset __devinitdata = {
.name = "CS5535",
.init_hwif = init_hwif_cs5535,
```

[git patches] IDE updates (part 4)

```
- .autodma = AUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_SINGLE | IDE_HFLAG_POST_SET_MODE,
+ .host_flags = IDE_HFLAG_SINGLE | IDE_HFLAG_POST_SET_MODE |
+ IDE_HFLAG_BOOTABLE,
.pio_mask = ATA_PIO4,
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA4,
};

static int __devinit cs5535_init_one(struct pci_dev *dev,
diff --git a/drivers/ide/pci/cy82c693.c b/drivers/ide/pci/cy82c693.c
index c498ecf..efc20bd 100644
--- a/drivers/ide/pci/cy82c693.c
+++ b/drivers/ide/pci/cy82c693.c
@@ -1,5 +1,5 @@
/*
- * linux/drivers/ide/pci/cy82c693.c Version 0.40 Sep. 10, 2002
+ * linux/drivers/ide/pci/cy82c693.c Version 0.41 Aug 27, 2007
*
* Copyright (C) 1998-2000 Andreas S. Krebs (akrebs@xxxxxxxxxxxxxx), Maintainer
* Copyright (C) 1998-2002 Andre Hedrick <andre@xxxxxxxxxxxxxx>, Integrator
@@ -431,15 +431,8 @@ static void __devinit init_hwif_cy82c693(ide_hwif_t *hwif)
hwif->chipset = ide_cy82c693;
hwif->set_pio_mode = &cy82c693_set_pio_mode;

- if (!hwif->dma_base) {
- hwif->drives[0].autotune = 1;
- hwif->drives[1].autotune = 1;
+ if (hwif->dma_base == 0)
return;
- }
-
- hwif->atapi_dma = 1;
- hwif->mwdma_mask = 0x04;
- hwif->swdma_mask = 0x04;

hwif->ide_dma_on = &cy82c693_ide_dma_on;
}
@@ -461,10 +454,11 @@ static ide_pci_device_t cy82c693_chipset __devinitdata = {
.init_chipset = init_chipset_cy82c693,
.init_iops = init_iops_cy82c693,
.init_hwif = init_hwif_cy82c693,
- .autodma = AUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_SINGLE | IDE_HFLAG_TRUST_BIOS_FOR_DMA,
+ .host_flags = IDE_HFLAG_SINGLE | IDE_HFLAG_TRUST_BIOS_FOR_DMA |
+ IDE_HFLAG_BOOTABLE,
.pio_mask = ATA_PIO4,
+ .swdma_mask = ATA_SWDMA2_ONLY,
+ .mwdma_mask = ATA_MWDMA2_ONLY,
```

[git patches] IDE updates (part 4)

```
};

static int __devinit cy82c693_init_one(struct pci_dev *dev, const struct pci_device_id *id)
diff --git a/drivers/ide/pci/generic.c b/drivers/ide/pci/generic.c
index cce6311..5116583 100644
--- a/drivers/ide/pci/generic.c
+++ b/drivers/ide/pci/generic.c
@@ -65,119 +65,65 @@ static void __devinit init_hwif_generic (ide_hwif_t *hwif)
default:
break;
}
-
- if (!(hwif->dma_base))
- return;
-
- hwif->atapi_dma = 1;
- hwif->ultra_mask = 0x7f;
- hwif->mwdma_mask = 0x07;
- hwif->swdma_mask = 0x07;
}

-#if 0
- /* Logic to add back later on */
-
- if ((dev->class >> 8) == PCI_CLASS_STORAGE_IDE) {
- ide_pci_device_t *unknown = unknown_chipset;
- init_setup_unknown(dev, unknown);
- return 1;
+#define DECLARE_GENERIC_PCI_DEV(name_str, dma_setting) \
+ { \
+ .name = name_str, \
+ .init_hwif = init_hwif_generic, \
+ .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA | \
+ dma_setting | \
+ IDE_HFLAG_BOOTABLE, \
+ .swdma_mask = ATA_SWDMA2, \
+ .mwdma_mask = ATA_MWDMA2, \
+ .udma_mask = ATA_UDMA6, \
+ }
- return 0;
-#endif

static ide_pci_device_t generic_chipsets[] __devinitdata = {
- { /* 0 */
- .name = "Unknown",
- .init_hwif = init_hwif_generic,
- .autodma = AUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- }, { /* 1 */
+ /* 0 */ DECLARE_GENERIC_PCI_DEV("Unknown", 0),
```

```

+
+ { /* 1 */
.name = "NS87410",
.init_hwif = init_hwif_generic,
- .autodma = AUTODMA,
.enablebits = {{0x43,0x08,0x08}, {0x47,0x08,0x08}},
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 2 */
- .name = "SAMURAI",
- .init_hwif = init_hwif_generic,
- .autodma = AUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 3 */
- .name = "HT6565",
- .init_hwif = init_hwif_generic,
- .autodma = AUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 4 */
- .name = "UM8673F",
- .init_hwif = init_hwif_generic,
- .autodma = NODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 5 */
- .name = "UM8886A",
- .init_hwif = init_hwif_generic,
- .autodma = NODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 6 */
- .name = "UM8886BF",
- .init_hwif = init_hwif_generic,
- .autodma = NODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 7 */
- .name = "HINT_IDE",
- .init_hwif = init_hwif_generic,
- .autodma = AUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 8 */
- .name = "VIA_IDE",
- .init_hwif = init_hwif_generic,
- .autodma = NOAUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 9 */

```

[git patches] IDE updates (part 4)

```
- .name = "OPTI621V",
- .init_hwif = init_hwif_generic,
- .autodma = NOAUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 10 */
+ .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA |
+ IDE_HFLAG_BOOTABLE,
+ .swdma_mask = ATA_SWDMA2,
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA6,
+ },
+
+ /* 2 */ DECLARE_GENERIC_PCI_DEV("SAMURAI", 0),
+ /* 3 */ DECLARE_GENERIC_PCI_DEV("HT6565", 0),
+ /* 4 */ DECLARE_GENERIC_PCI_DEV("UM8673F", IDE_HFLAG_NO_DMA),
+ /* 5 */ DECLARE_GENERIC_PCI_DEV("UM8886A", IDE_HFLAG_NO_DMA),
+ /* 6 */ DECLARE_GENERIC_PCI_DEV("UM8886BF", IDE_HFLAG_NO_DMA),
+ /* 7 */ DECLARE_GENERIC_PCI_DEV("HINT_IDE", 0),
+ /* 8 */ DECLARE_GENERIC_PCI_DEV("VIA_IDE", IDE_HFLAG_NO_AUTODMA),
+ /* 9 */ DECLARE_GENERIC_PCI_DEV("OPTI621V", IDE_HFLAG_NO_AUTODMA),
+
+ { /* 10 */
. name = "VIA8237SATA",
. init_hwif = init_hwif_generic,
- .autodma = AUTODMA,
- .bootable = OFF_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 11 */
- .name = "Piccolo0102",
- .init_hwif = init_hwif_generic,
- .autodma = NOAUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 12 */
- .name = "Piccolo0103",
- .init_hwif = init_hwif_generic,
- .autodma = NOAUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 13 */
- .name = "Piccolo0105",
- .init_hwif = init_hwif_generic,
- .autodma = NOAUTODMA,
- .bootable = ON_BOARD,
- .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
- },{ /* 14 */
+ .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA |
+ IDE_HFLAG_OFF_BOARD,
+ .swdma_mask = ATA_SWDMA2,
+ .mwdma_mask = ATA_MWDMA2,
```

[git patches] IDE updates (part 4)

```
+ .udma_mask = ATA_UDMA6,
+ },
+
+ /* 11 */ DECLARE_GENERIC_PCI_DEV("Piccolo0102", IDE_HFLAG_NO_AUTODMA),
+ /* 12 */ DECLARE_GENERIC_PCI_DEV("Piccolo0103", IDE_HFLAG_NO_AUTODMA),
+ /* 13 */ DECLARE_GENERIC_PCI_DEV("Piccolo0105", IDE_HFLAG_NO_AUTODMA),
+
+ { /* 14 */
+ .name = "Revolution",
+ .init_hwif = init_hwif_generic,
+ .autodma = AUTODMA,
+ .bootable = OFF_BOARD,
+ .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA,
+ .host_flags = IDE_HFLAG_TRUST_BIOS_FOR_DMA |
+ IDE_HFLAG_OFF_BOARD,
+ .swdma_mask = ATA_SWDMA2,
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA6,
+ }
+};
```

```
diff --git a/drivers/ide/pci/hpt34x.c b/drivers/ide/pci/hpt34x.c
index 44ac0e2..67af1a7 100644
```

```
--- a/drivers/ide/pci/hpt34x.c
```

```
+++ b/drivers/ide/pci/hpt34x.c
```

```
@@ -125,49 +125,45 @@ static unsigned int __devinit init_chipset_hpt34x(struct pci_dev *dev, const cha
```

```
static void __devinit init_hwif_hpt34x(ide_hwif_t *hwif)
```

```
{
```

```
- u16 pcicmd = 0;
```

```
-
```

```
hwif->set_pio_mode = &hpt34x_set_pio_mode;
```

```
hwif->set_dma_mode = &hpt34x_set_mode;
```

```
+}
```

```
- hwif->drives[0].autotune = 1;
```

```
- hwif->drives[1].autotune = 1;
```

```
-
```

```
- pci_read_config_word(hwif->pci_dev, PCI_COMMAND, &pcicmd);
```

```
-
```

```
- if (!hwif->dma_base)
```

```
- return;
```

```
-
```

```
+static ide_pci_device_t hpt34x_chipsets[] __devinitdata = {
```

```
+ { /* 0 */
```

```
+ .name = "HPT343",
```

```
+ .init_chipset = init_chipset_hpt34x,
```

```
+ .init_hwif = init_hwif_hpt34x,
```

```
+ .extra = 16,
```

```
+ .host_flags = IDE_HFLAG_NO_ATAPI_DMA |
```

```
+ IDE_HFLAG_NO_AUTODMA,
```

[git patches] IDE updates (part 4)

```
+ .pio_mask = ATA_PIO5,
+ },
+ { /* 1 */
+ .name = "HPT345",
+ .init_chipset = init_chipset_hpt34x,
+ .init_hwif = init_hwif_hpt34x,
+ .extra = 16,
+ .host_flags = IDE_HFLAG_NO_ATAPI_DMA |
+ IDE_HFLAG_NO_AUTODMA |
+ IDE_HFLAG_OFF_BOARD,
+ .pio_mask = ATA_PIO5,
#ifdef CONFIG_HPT34X_AUTODMA
- if ((pcicmd & PCI_COMMAND_MEMORY) == 0)
- return;
-
- hwif->ultra_mask = 0x07;
- hwif->mwdma_mask = 0x07;
- hwif->swdma_mask = 0x07;
+ .swdma_mask = ATA_SWDMA2,
+ .mwdma_mask = ATA_MWDMA2,
+ .udma_mask = ATA_UDMA2,
#endif
-}
-
-static ide_pci_device_t hpt34x_chipset __devinitdata = {
- .name = "HPT34X",
- .init_chipset = init_chipset_hpt34x,
- .init_hwif = init_hwif_hpt34x,
- .autodma = NOAUTODMA,
- .bootable = NEVER_BOARD,
- .extra = 16,
- .pio_mask = ATA_PIO5,
+ }
};

static int __devinit hpt34x_init_one(struct pci_dev *dev, const struct pci_device_id *id)
{
- ide_pci_device_t *d = &hpt34x_chipset;
- static char *chipset_names[] = {"HPT343", "HPT345"};
+ ide_pci_device_t *d;
u16 pcicmd = 0;

pci_read_config_word(dev, PCI_COMMAND, &pcicmd);

- d->name = chipset_names[(pcicmd & PCI_COMMAND_MEMORY) ? 1 : 0];
- d->bootable = (pcicmd & PCI_COMMAND_MEMORY) ? OFF_BOARD : NEVER_BOARD;
+ d = &hpt34x_chipsets[(pcicmd & PCI_COMMAND_MEMORY) ? 1 : 0];

return ide_setup_pci_device(dev, d);
}
diff --git a/drivers/ide/pci/hpt366.c b/drivers/ide/pci/hpt366.c
```

[git patches] IDE updates (part 4)

```
index fcb21dd..18f5b7d 100644
--- a/drivers/ide/pci/hpt366.c
+++ b/drivers/ide/pci/hpt366.c
@@ -1,9 +1,10 @@
/*
- * linux/drivers/ide/pci/hpt366.c Version 1.14 Oct 1, 2007
+ * linux/drivers/ide/pci/hpt366.c Version 1.20 Oct 1, 2007
*
* Copyright (C) 1999–2003 Andre Hedrick <andre@xxxxxxxxxxxxxxx>
* Portions Copyright (C) 2001 Sun Microsystems, Inc.
* Portions Copyright (C) 2003 Red Hat Inc
+ * Portions Copyright (C) 2007 Bartlomiej Zolnierkiewicz
* Portions Copyright (C) 2005–2007 MontaVista Software, Inc.
*
* Thanks to HighPoint Technologies for their assistance, and hardware.
@@ -393,8 +394,9 @@ enum ata_clock {
*/

struct hpt_info {
+ char *chip_name; /* Chip name */
u8 chip_type; /* Chip type */
- u8 max_ultra; /* Max. UltraDMA mode allowed */
+ u8 udma_mask; /* Allowed UltraDMA modes mask. */
u8 dpll_clk; /* DPLL clock in MHz */
u8 pci_clk; /* PCI clock in MHz */
u32 **settings; /* Chipset settings table */
@@ -432,78 +434,89 @@ static u32 *hpt37x_settings[NUM_ATA_CLOCKS] = {
};

static struct hpt_info hpt36x __devinitdata = {
+ .chip_name = "HPT36x",
.chip_type = HPT36x,
- .max_ultra = HPT366_ALLOW_ATA66_3 ? (HPT366_ALLOW_ATA66_4 ? 4 : 3) : 2,
+ .udma_mask = HPT366_ALLOW_ATA66_3 ? (HPT366_ALLOW_ATA66_4 ? ATA_UDMA4 :
ATA_UDMA3) : ATA_UDMA2,
.dpll_clk = 0, /* no DPLL */
.settings = hpt36x_settings
};

static struct hpt_info hpt370 __devinitdata = {
+ .chip_name = "HPT370",
.chip_type = HPT370,
- .max_ultra = HPT370_ALLOW_ATA100_5 ? 5 : 4,
+ .udma_mask = HPT370_ALLOW_ATA100_5 ? ATA_UDMA5 : ATA_UDMA4,
.dpll_clk = 48,
.settings = hpt37x_settings
};

static struct hpt_info hpt370a __devinitdata = {
+ .chip_name = "HPT370A",
.chip_type = HPT370A,
```

[git patches] IDE updates (part 4)

```
- .max_ultra = HPT370_ALLOW_ATA100_5 ? 5 : 4,  
+ .udma_mask = HPT370_ALLOW_ATA100_5 ? ATA_UDMA5 : ATA_UDMA4,  
.dpll_clk = 48,  
.settings = hpt37x_settings  
};
```

```
static struct hpt_info hpt374 __devinitdata = {  
+ .chip_name = "HPT374",  
.chip_type = HPT374,  
- .max_ultra = 5,  
+ .udma_mask = ATA_UDMA5,  
.dpll_clk = 48,  
.settings = hpt37x_settings  
};
```

```
static struct hpt_info hpt372 __devinitdata = {  
+ .chip_name = "HPT372",  
.chip_type = HPT372,  
- .max_ultra = HPT372_ALLOW_ATA133_6 ? 6 : 5,  
+ .udma_mask = HPT372_ALLOW_ATA133_6 ? ATA_UDMA6 : ATA_UDMA5,  
.dpll_clk = 55,  
.settings = hpt37x_settings  
};
```

```
static struct hpt_info hpt372a __devinitdata = {  
+ .chip_name = "HPT372A",  
.chip_type = HPT372A,  
- .max_ultra = HPT372_ALLOW_ATA133_6 ? 6 : 5,  
+ .udma_mask = HPT372_ALLOW_ATA133_6 ? ATA_UDMA6 : ATA_UDMA5,  
.dpll_clk = 66,  
.settings = hpt37x_settings  
};
```

```
static struct hpt_info hpt302 __devinitdata = {  
+ .chip_name = "HPT302",  
.chip_type = HPT302,  
- .max_ultra = HPT372_ALLOW_ATA133_6 ? 6 : 5,  
+ .udma_mask = HPT302_ALLOW_ATA133_6 ? ATA_UDMA6 : ATA_UDMA5,  
.dpll_clk = 66,  
.settings = hpt37x_settings  
};
```

```
static struct hpt_info hpt371 __devinitdata = {  
+ .chip_name = "HPT371",  
.chip_type = HPT371,  
- .max_ultra = HPT371_ALLOW_ATA133_6 ? 6 : 5,  
+ .udma_mask = HPT371_ALLOW_ATA133_6 ? ATA_UDMA6 : ATA_UDMA5,  
.dpll_clk = 66,  
.settings = hpt37x_settings  
};
```

[git patches] IDE updates (part 4)

```
static struct hpt_info hpt372n __devinitdata = {
+ .chip_name = "HPT372N",
. chip_type = HPT372N,
- .max_ultra = HPT372_ALLOW_ATA133_6 ? 6 : 5,
+ .udma_mask = HPT372_ALLOW_ATA133_6 ? ATA_UDMA6 : ATA_UDMA5,
. dpll_clk = 77,
. settings = hpt37x_settings
};
```

```
static struct hpt_info hpt302n __devinitdata = {
+ .chip_name = "HPT302N",
. chip_type = HPT302N,
- .max_ultra = HPT302_ALLOW_ATA133_6 ? 6 : 5,
+ .udma_mask = HPT302_ALLOW_ATA133_6 ? ATA_UDMA6 : ATA_UDMA5,
. dpll_clk = 77,
. settings = hpt37x_settings
};
```

```
static struct hpt_info hpt371n __devinitdata = {
+ .chip_name = "HPT371N",
. chip_type = HPT371N,
- .max_ultra = HPT371_ALLOW_ATA133_6 ? 6 : 5,
+ .udma_mask = HPT371_ALLOW_ATA133_6 ? ATA_UDMA6 : ATA_UDMA5,
. dpll_clk = 77,
. settings = hpt37x_settings
};
@@ -676,12 +689,11 @@ static int hpt3xx_quirkproc(ide_drive_t *drive)
```

```
static void hpt3xx_intrproc(ide_drive_t *drive)
{
- ide_hwif_t *hwif = HWIF(drive);
-
if (drive->quirk_list)
return;
+
/* drives in the quirk_list may not like intr setups/cleanups */
- hwif->OUTB(drive->ctl | 2, IDE_CONTROL_REG);
+ outb(drive->ctl | 2, IDE_CONTROL_REG);
}
```

```
static void hpt3xx_maskproc(ide_drive_t *drive, int mask)
@@ -709,8 +721,8 @@ static void hpt3xx_maskproc(ide_drive_t *drive, int mask)
enable_irq (hwif->irq);
}
} else
- hwif->OUTB(mask ? (drive->ctl | 2) : (drive->ctl & ~2),
- IDE_CONTROL_REG);
+ outb(mask ? (drive->ctl | 2) : (drive->ctl & ~2),
+ IDE_CONTROL_REG);
}
```

[git patches] IDE updates (part 4)

```
/*
@@ -750,9 +762,9 @@ static void hpt370_irq_timeout(ide_drive_t *drive)
printk(KERN_DEBUG "%s: %d bytes in FIFO\n", drive->name, bfifo & 0x1ff);

/* get DMA command mode */
- dma_cmd = hwif->INB(hwif->dma_command);
+ dma_cmd = inb(hwif->dma_command);
/* stop DMA */
- hwif->OUTB(dma_cmd & ~0x1, hwif->dma_command);
+ outb(dma_cmd & ~0x1, hwif->dma_command);
hpt370_clear_engine(drive);
}

@@ -767,12 +779,12 @@ static void hpt370_ide_dma_start(ide_drive_t *drive)
static int hpt370_ide_dma_end(ide_drive_t *drive)
{
ide_hwif_t *hwif = HWIF(drive);
- u8 dma_stat = hwif->INB(hwif->dma_status);
+ u8 dma_stat = inb(hwif->dma_status);

if (dma_stat & 0x01) {
/* wait a little */
udelay(20);
- dma_stat = hwif->INB(hwif->dma_status);
+ dma_stat = inb(hwif->dma_status);
if (dma_stat & 0x01)
hpt370_irq_timeout(drive);
}
@@ -833,34 +845,32 @@ static int hpt374_ide_dma_end(ide_drive_t *drive)

static void hpt3xxn_set_clock(ide_hwif_t *hwif, u8 mode)
{
- u8 scr2 = hwif->INB(hwif->dma_master + 0x7b);
+ u8 scr2 = inb(hwif->dma_master + 0x7b);

if ((scr2 & 0x7f) == mode)
return;

/* Tristate the bus */
- hwif->OUTB(0x80, hwif->dma_master + 0x73);
- hwif->OUTB(0x80, hwif->dma_master + 0x77);
+ outb(0x80, hwif->dma_master + 0x73);
+ outb(0x80, hwif->dma_master + 0x77);

/* Switch clock and reset channels */
- hwif->OUTB(mode, hwif->dma_master + 0x7b);
- hwif->OUTB(0xc0, hwif->dma_master + 0x79);
+ outb(mode, hwif->dma_master + 0x7b);
+ outb(0xc0, hwif->dma_master + 0x79);

/*
```

[git patches] IDE updates (part 4)

```
* Reset the state machines.
* NOTE: avoid accidentally enabling the disabled channels.
*/
- hwif->OUTB(hwif->INB(hwif->dma_master + 0x70) | 0x32,
- hwif->dma_master + 0x70);
- hwif->OUTB(hwif->INB(hwif->dma_master + 0x74) | 0x32,
- hwif->dma_master + 0x74);
+ outb(inb(hwif->dma_master + 0x70) | 0x32, hwif->dma_master + 0x70);
+ outb(inb(hwif->dma_master + 0x74) | 0x32, hwif->dma_master + 0x74);

/* Complete reset */
- hwif->OUTB(0x00, hwif->dma_master + 0x79);
+ outb(0x00, hwif->dma_master + 0x79);

/* Reconnect channels to bus */
- hwif->OUTB(0x00, hwif->dma_master + 0x73);
- hwif->OUTB(0x00, hwif->dma_master + 0x77);
+ outb(0x00, hwif->dma_master + 0x73);
+ outb(0x00, hwif->dma_master + 0x77);
}

/**
@@ -1139,7 +1149,7 @@ static unsigned int __devinit init_chipset_hpt366(struct pci_dev *dev, const cha
* Select 66 MHz DPLL clock only if UltraATA/133 mode is
* supported/enabled, use 50 MHz DPLL clock otherwise...
*/
- if (info->max_ultra == 6) {
+ if (info->udma_mask == ATA_UDMA6) {
dpll_clk = 66;
clock = ATA_CLOCK_66MHZ;
} else if (dpll_clk) { /* HPT36x chips don't have DPLL */
@@ -1291,14 +1301,9 @@ static void __devinit init_hwif_hpt366(ide_hwif_t *hwif)
if (new_mcr != old_mcr)
pci_write_config_byte(dev, hwif->select_data + 1, new_mcr);

- hwif->drives[0].autotune = hwif->drives[1].autotune = 1;
-
if (hwif->dma_base == 0)
return;

- hwif->ultra_mask = hwif->cds->udma_mask;
- hwif->mwdma_mask = 0x07;
-
/*
* The HPT37x uses the CBLID pins as outputs for MA15/MA16
* address lines to access an external EEPROM. To read valid
@@ -1354,7 +1359,7 @@ static void __devinit init_dma_hpt366(ide_hwif_t *hwif, unsigned long dmabase)
u8 dma_new = 0, dma_old = 0;
unsigned long flags;

- dma_old = hwif->INB(dmabase + 2);
```

[git patches] IDE updates (part 4)

```
+ dma_old = inb(dmabase + 2);

local_irq_save(flags);

@@ -1365,60 +1370,26 @@ static void __devinit init_dma_hpt366(ide_hwif_t *hwif, unsigned long
dmabase)
if (masterdma & 0x30) dma_new |= 0x20;
if ( slavedma & 0x30) dma_new |= 0x40;
if (dma_new != dma_old)
- hwif->OUTB(dma_new, dmabase + 2);
+ outb(dma_new, dmabase + 2);

local_irq_restore(flags);

ide_setup_dma(hwif, dmabase, 8);
}

-static int __devinit init_setup_hpt374(struct pci_dev *dev, ide_pci_device_t *d)
+static void __devinit hpt374_init(struct pci_dev *dev, struct pci_dev *dev2)
{
- struct pci_dev *dev2;
-
- if (PCI_FUNC(dev->devfn) & 1)
- return -ENODEV;
-
- pci_set_drvdata(dev, &hpt374);
-
- if ((dev2 = pci_get_slot(dev->bus, dev->devfn + 1)) != NULL) {
- int ret;
-
- pci_set_drvdata(dev2, &hpt374);
-
- if (dev2->irq != dev->irq) {
- /* FIXME: we need a core pci_set_interrupt() */
- dev2->irq = dev->irq;
- printk(KERN_WARNING "%s: PCI config space interrupt "
- "fixed.\n", d->name);
- }
- ret = ide_setup_pci_devices(dev, dev2, d);
- if (ret < 0)
- pci_dev_put(dev2);
- return ret;
+ if (dev2->irq != dev->irq) {
+ /* FIXME: we need a core pci_set_interrupt() */
+ dev2->irq = dev->irq;
+ printk(KERN_INFO "HPT374: PCI config space interrupt fixed\n");
+ }
- return ide_setup_pci_device(dev, d);
-}
-
-static int __devinit init_setup_hpt372n(struct pci_dev *dev, ide_pci_device_t *d)
```

[git patches] IDE updates (part 4)

```
-{
- pci_set_drvdata(dev, &hpt372n);
-
- return ide_setup_pci_device(dev, d);
- }

-static int __devinit init_setup_hpt371(struct pci_dev *dev, ide_pci_device_t *d)
+static void __devinit hpt371_init(struct pci_dev *dev)
{
- struct hpt_info *info;
u8 mcr1 = 0;

- if (dev->revision > 1) {
- d->name = "HPT371N";
-
- info = &hpt371n;
- } else
- info = &hpt371;
-
-/*
* HPT371 chips physically have only one channel, the secondary one,
* but the primary channel registers do exist! Go figure...
@@ -1428,194 +1399,102 @@ static int __devinit init_setup_hpt371(struct pci_dev *dev, ide_pci_device_t
*d)
pci_read_config_byte(dev, 0x50, &mcr1);
if (mcr1 & 0x04)
pci_write_config_byte(dev, 0x50, mcr1 & ~0x04);
-
- pci_set_drvdata(dev, info);
-
- return ide_setup_pci_device(dev, d);
- }
-
-static int __devinit init_setup_hpt372a(struct pci_dev *dev, ide_pci_device_t *d)
-{
- struct hpt_info *info;
-
- if (dev->revision > 1) {
- d->name = "HPT372N";
-
- info = &hpt372n;
- } else
- info = &hpt372a;
- pci_set_drvdata(dev, info);
-
- return ide_setup_pci_device(dev, d);
- }

-static int __devinit init_setup_hpt302(struct pci_dev *dev, ide_pci_device_t *d)
+static int __devinit hpt36x_init(struct pci_dev *dev, struct pci_dev *dev2)
{
```

[git patches] IDE updates (part 4)

```
- struct hpt_info *info;
-
- if (dev->revision > 1) {
- d->name = "HPT302N";
+ u8 mcr1 = 0, pin1 = 0, pin2 = 0;

- info = &hpt302n;
- } else
- info = &hpt302;
- pci_set_drvdata(dev, info);
-
- return ide_setup_pci_device(dev, d);
-}
-
-static int __devinit init_setup_hpt366(struct pci_dev *dev, ide_pci_device_t *d)
-{
- struct pci_dev *dev2;
- u8 rev = dev->revision;
- static char *chipset_names[] = { "HPT366", "HPT366", "HPT368",
- "HPT370", "HPT370A", "HPT372",
- "HPT372N" };
- static struct hpt_info *info[] = { &hpt36x, &hpt36x, &hpt36x,
- &hpt370, &hpt370a, &hpt372,
- &hpt372n };
-
- if (PCI_FUNC(dev->devfn) & 1)
- return -ENODEV;
+ /*
+ * Now we'll have to force both channels enabled if
+ * at least one of them has been enabled by BIOS...
+ */
+ pci_read_config_byte(dev, 0x50, &mcr1);
+ if (mcr1 & 0x30)
+ pci_write_config_byte(dev, 0x50, mcr1 | 0x30);

- switch (rev) {
- case 0:
- case 1:
- case 2:
- /*
- * HPT36x chips have one channel per function and have
- * both channel enable bits located differently and visible
- * to both functions -- really stupid design decision... :-(
- * Bit 4 is for the primary channel, bit 5 for the secondary.
- */
- d->host_flags |= IDE_HFLAG_SINGLE;
- d->enablebits[0].mask = d->enablebits[0].val = 0x10;
+ pci_read_config_byte(dev, PCI_INTERRUPT_PIN, &pin1);
+ pci_read_config_byte(dev2, PCI_INTERRUPT_PIN, &pin2);

- d->udma_mask = HPT366_ALLOW_ATA66_3 ? (HPT366_ALLOW_ATA66_4 ?
```

[git patches] IDE updates (part 4)

```
- ATA_UDMA4 : ATA_UDMA3) : ATA_UDMA2;
- break;
- case 3:
- case 4:
- d->udma_mask = HPT370_ALLOW_ATA100_5 ? ATA_UDMA5 : ATA_UDMA4;
- break;
- default:
- rev = 6;
- /* fall thru */
- case 5:
- case 6:
- d->udma_mask = HPT372_ALLOW_ATA133_6 ? ATA_UDMA6 : ATA_UDMA5;
- break;
+ if (pin1 != pin2 && dev->irq == dev2->irq) {
+ printk(KERN_INFO "HPT36x: onboard version of chipset, "
+ "pin1=%d pin2=%d\n", pin1, pin2);
+ return 1;
+ }

- d->name = chipset_names[rev];
-
- pci_set_drvdata(dev, info[rev]);
-
- if (rev > 2)
- goto init_single;
-
- if ((dev2 = pci_get_slot(dev->bus, dev->devfn + 1)) != NULL) {
- u8 mcr1 = 0, pin1 = 0, pin2 = 0;
- int ret;
-
- pci_set_drvdata(dev2, info[rev]);
-
- /*
- * Now we'll have to force both channels enabled if
- * at least one of them has been enabled by BIOS...
- */
- pci_read_config_byte(dev, 0x50, &mcr1);
- if (mcr1 & 0x30)
- pci_write_config_byte(dev, 0x50, mcr1 | 0x30);
-
- pci_read_config_byte(dev, PCI_INTERRUPT_PIN, &pin1);
- pci_read_config_byte(dev2, PCI_INTERRUPT_PIN, &pin2);
- if (pin1 != pin2 && dev->irq == dev2->irq) {
- d->bootable = ON_BOARD;
- printk("%s: onboard version of chipset, pin1=%d pin2=%d\n",
- d->name, pin1, pin2);
- }
- ret = ide_setup_pci_devices(dev, dev2, d);
- if (ret < 0)
- pci_dev_put(dev2);
- return ret;
```

```
- }
-init_single:
- return ide_setup_pci_device(dev, d);
+ return 0;
}

static ide_pci_device_t hpt366_chipsets[] __devinitdata = {
{ /* 0 */
- .name = "HPT366",
- .init_setup = init_setup_hpt366,
+ .name = "HPT36x",
. init_chipset = init_chipset_hpt366,
. init_hwif = init_hwif_hpt366,
. init_dma = init_dma_hpt366,
- .autodma = AUTODMA,
- .enablebits = {{0x50,0x04,0x04}, {0x54,0x04,0x04}},
- .bootable = OFF_BOARD,
+ /*
+ * HPT36x chips have one channel per function and have
+ * both channel enable bits located differently and visible
+ * to both functions --- really stupid design decision... :-(
+ * Bit 4 is for the primary channel, bit 5 for the secondary.
+ */
+ .enablebits = {{0x50,0x10,0x10}, {0x54,0x04,0x04}},
. extra = 240,
. host_flags = IDE_HFLAG_SINGLE |
+ IDE_HFLAG_NO_ATAPI_DMA |
+ IDE_HFLAG_OFF_BOARD,
. pio_mask = ATA_PIO4,
+ .mwdma_mask = ATA_MWDMA2,
}, { /* 1 */
. name = "HPT372A",
- .init_setup = init_setup
```