The K Project

ATAPI Driver

LSE Team

EPITA

May 17, 2019

Introduction

- **ATA** (*AT Attachment*) original interface standard for hard drives
- Originally named IDE (*Integrated Drive Electronics*)
- Replaced by **SATA** (*Serial ATA*) in 2003

ATAPI

- *ATA Packet Interface*
- Designed to use ATA for other devices than hard disks
- In our case: CD-ROM
- Part of the ATA standard
- **SCSI** commands and responses through the ATA interface

ATA - IBM PC

- 2 ATA controllers (buses)
- IRQ14 and IRQ15 of slave PIC
- Maximum 2 drives per ATA bus
- Each controller has a set of I/O ports

ATA Registers

- Primary registers:
  - 1st controller: 0x1f0
  - 2nd controller: 0x170
- Offsets from primary register:
  - +0: Data Register
  - +1: Error Register (R)
  - +1: Features Register (W)
  - +2: Sector Count Register
  - +3: LBA Low Register
  - +4: LBA Mid Register
  - +5: LBA High Register
  - +6: Drive/Head Register
  - +7: Status Register (R)
  - +7: Command Register (W)
- Device Control Register (DCR): 0x3f6 and 0x376

Status Register

- Bit 0: Error (ERR)
- Bit 1: Index Mark (IDX)
- Bit 2: Data Corrected (CORR)
- Bit 3: Data Transfer Requested (DRQ)
- Bit 4: Seek complete (DSC)
- Bit 5: Device Fault (DF)
- Bit 6: Device Ready (DRDY)
- Bit 7: Busy (BSY)

Discover ATAPI drive

For each ATA bus:

- Send 'Software Reset' to the controller's DCR
- Send 'Disable IRQ' to the controller's DCR

```
outb(DCR(port), SOFTWARE_RESET)
outb(DCR(port), DISABLE_IRQ)
```

Discover ATAPI drive


```
/* reg == PRIMARY_REG || reg == SECONDARY_REG */
for drive in (ATA_PORT_MASTER, ATA_PORT_SLAVE):
    /* Select current drive */
    outb(DRIVE_REG(reg), drive);
    /* Approx. 4 * inb() */
    wait_400ns();
    /* Look for ATAPI signature */
    sig[0] = inb(SECTOR_COUNT_REG(port))
    sig[1] = inb(LBA_LO_REG(port))
    sig[2] = inb(LBA_MI_REG(port))
    sig[3] = inb(LBA_HI_REG(port))
    /*
     * memcmp(sig, ...)
     * If it matches: saves (reg, drive)
     */
```

Read block

- Send SCSI packet with 'READ 12' commmand
- To send a packet to the drive:

```
busy_wait();
outb(FEATURES_REG(drive), 0); /* No overlap/no DM
outb(SECTOR_CNT_REG(drive), 0);     /* No queuing
outb(LBA_MI_REG(drive), CDROM_BLK_SIZE);
outb(LBA_HI_REG(drive), CDROM_BLK_SIZE >> 8);
outb(COMMAND_REG(drive), 0xa0); /* PACKET */
wait_packet_req();
```

Read block

- To wait for a packet to be requested:
  - Poll the status register until BSY is cleared and DRQ is set
- Write the SCSI packet to the Data Register
  - One word at a time (you must use `outw()`)
- Read Sector Count Reg while it doesn't return DATA_TRANSMIT (0x2)

Read block

Once the SCSI packet has been sent:

- Read CDROM_BLK_SIZE word by word:
  `inw(DATA_REGISTER(port))`
- Read Sector Count Register while it does not return PACKET_COMMAND_COMPLETE (0x3)

Write 'waiting' helper functions:

```
void busy_wait(u16 drive);
void wait_device_selection(u16 drive);
void wait_packet_request(u16 drive);
```

These functions should only be doing `inb()` calls and status checking.

Recommanded methodology

Write functions to discover the ATAPI device:

```
void select_drive(u16 bus, u8 slave);
bool is_atapi_drive(u16 bus, u8 slave);
void discover_atapi_drive();
```

Recommanded methodology

Write functions to read data on the drive:

```
int send_packet(struct SCSI_packet *pkt, u16 drive,
                u16 size);
void *read_block(size_t lba);
```

Feel free to modify the proposed function prototypes.

Introduction

- On the ATAPI drive, there will be an ISO 9660 filesystem.
- You should already know how it works.
- The header you had in `myreadiso` is given.

File System Interface


```
int open(const char *pathname, int flags);
ssize_t read(int fd, void *buf, size_t count);
off_t seek(int fd, off_t offset, int whence);
int close(int fd);
```

Open and Close

Open:

- locate the file and store the infos you need to retrieve it quickly
- register an fd in your fd table

Close:

- free the fd

Read and Seek

Seek:

- modify the file offset according to `whence`

Read:

- use the informations you stored in `open()`
- find the blocks of data to read and copy it
- modify the file offset

Contact

- k[at]lse.epita.fr
- labos.lse with [K] tag
- #k (irc.rezosup.org)
- guillaume.pagnoux[at]lse.epita.fr
- tom.decrette[at]lse.epita.fr