INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

COMPUTER SYSTEMS LABORATORY

Fall 2011

Stage 2

Synopsis

During this stage you will develop the final system, running on a bare PC (i.e. without an operating system).

This final step requires you to develop code to access the hard disk in order to read the bitmap files from the Minix file system.

At the end of this stage, you should know:

- how to control a hard disk;
- how to deal with a Minix file system in a more generic way;
- incidentally, what is the internal organization of a specific type of BMP file;

Dealing with the Minix file system

As in this stage we have the processor set to long mode, we're now able to develop the C code for dealing with the Minix file system in a more favorable environment than in Stage 0. In particular, we will develop all the code on top of a readSectors function, which initially will read directly from the virtual hard disk file.

Exercise 1

Create a C module that gives read access to a hard disk partition, by direct access to a virtual hard disk file, with operations: openPartition, readSectors, closePartition.

Exercise 2

On top of the previous operations, develop a C module to read directories and files from a Minix partition.

Reading the hard disk

Mass storage devices on a PC, including hard disks and CD-ROMs, were frequently connected to Parallel ATA channels. Each Parallel ATA channel supported two devices, labeled *master* and *slave*. Many PCs had two ATA channels, usually referred to as primary and secondary.

Our system has a Parallel ATA hard disk, configured as master on the primary ATA channel. The ports used to the access devices on this channel are indicated in the following table.

Port	Width	Read	Write
0x01F0	16	Data Register	
0x01F1	8	Error Register	Features Register
0x01F2	8	Sector Count Register	
0x01F3	8	LBA 7-0 Register	
0x01F4	8	LBA 15-8 Register	
0x01F5	8	LBA 23-16 Register	
0x01F6	8	Drive / LBA 27-24 Register	
0x01F7	8	Status Register	Command Register
0x03F6	8	Alternate Status Register	Device Control Register

The bits in the status register and in the alternate status register are organized as follows:

7	6	5	4	3	2	1	0
BSY	RDY	DF		DRQ			ERR

BSY	Operation in course
RDY	Device not ready
DF	Device fault
DRQ	Data transfer may start
ERR	Error in previous command

7	6	5	4	3	2	1	0
					SRST	nIEN	

nIEN	Interrupt Enable (negated)
SRST	Reset master and slave devices

We will poll the device to detect status changes, so nIEN should be set to 1 at program start.

To read sector from the master device:

1. Wait for BSY == 0
2. Write 0xE0 (LBA mode, Master dev.) ored with 27:24 bits of the LBA address to port 6
3. Pause for 400ns
4. Write 23:16 bits of the LBA address to port 5
5. Write 15:8 bits of the LBA address to port 4
6. Write 7:0 bits of the LBA address to port 3
7. Write the number of sectors to read to port 2 (note: 0 means 256)
8. Write 0 (PIO mode) to port 1
9. Write 0x20 to port 7
10. Pause for 400ns
11. Wait for BSY == 0
12. Wait for ERR == 1, DF == 1, or DRQ == 1
13. If ERR == 1 or DF ==1 the operation failed; else if DRQ == 1 data is ready to be read
14. Use rep insw to read 1 sector (256 words) from port 0
15. If there all sector are read, exit; else goto 11

Final Exercise

Build a system to show photos with an interval between them. The photos will be read from 800x600 24bpp BMP files present in the Minix file system.