Dept. of Computer Engineering, METU
CENG336 Laboratory Session # 2
Reference Document

Assistants
Mehmet Tan, Sertan Girgin

1. Before the Lab

1.1 Useful Resources

1. Microcomputer and Interfacing, Douglas V. Hall, Chapters 9 and 10
2. The Intel 32-bit Microprocessors, Barry B. Brey
3. HELPPC (can be downloaded from ftp://ftp.simtel.net/pub/simtelnet/msdos/info/helppc21.zip or you can read it online at http://members.tripod.com/~oldboard/assembly/)
4. Art of Assembly, (available from http://cs.smith.edu/~thiebaut/ArtOfAssembly/artofasm.html (in html format), or from http://zoo.cs.yale.edu/classes/cs421/papers/art-of-asm/pdf/ (in PDF format)

1.2 Materials and Equipment

1. Serial port of a PC
2. Serial null modem cable (will be available in the lab).

1.3 Objective

At the end of this experiment, you will be able to

- Override interrupt handlers
- Read/Write data from/to a file.
- Implement serial communications between PCs

1.4 Lab Rules:

1. Those without any preliminary work will not be allowed to attend the lab.
2. Those who have cheated in preliminary, quiz or other lab work get zero for both labs.
3. Lab groups (i.e. your partner) are assigned at random and announced just before the lab session.
4. Those who are more than 20 minutes late will not be allowed to attend the lab session.
5. Make-up labs require written official excuse. (e.g. from Health Center)
6. Collaboration between lab groups is forbidden.
7. The assistant may ask questions to each individual about the details of the experiment, which are also graded.
8. At the end of every lab session, there'll be a closed-book-and-notes lab quiz, which is strictly ten (10) minutes long. Getting less than 50% from the quiz implies the lab grade to be reduced to 50%.
9. Eating, drinking and smoking are forbidden.

10. Students are responsible from leaving the lab clean. Cleanliness and tidiness is also subject to grading.

Background

In this lab, you will implement simple file transfer between two computers over serial connection. The programming background information is contained in *Art of Assembly*, Chapter 13 (File I/O) and Chapter 22 (The PC Serial Ports). You also have to program the 8259-compatible interrupt controller on the computer board to enable serial port interrupts.

2. Preliminary Work

2.1 Initializing the serial port controller

Write an assembly program to initialize COM1 to 9600baud, 7 data bits, 2 stop bits and even parity.

2.2 Implementing serial communications

Write an assembly program which sends the character read from keyboard to COM1 and displays any character received from the serial port.

2.3 Implementing simple file transfer

Write two assembly programs (client and server), which together used to transfer files from one computer to another. The client will send commands (when necessary), and the server will send responses. Client program must interact with the user and implement the following commands:

lpwd

Display current local directory (*client side*).

Example:
```
> lpwd
a:\
```

BONUS	**pwd**
	Display current remote directory (*server side*).
	
	Example:
	`> pwd`
	`a:\`

lcd *directory*

Change current local directory to *directory.*

Example:
```
> lcd misc
```

```
> lcd ..\subdir
```

BONUS	**cd *directory***
	Change current remote directory to *directory*.

	Example:
	`> cd misc`
	`> cd ..\subdir`

ldir

Display contents of the current local directory in the following format:

Filename Size [*Date Time*]

Date (YYYYMMDD) and *Time* (HH:MM) fields are optional (bonus).

Example:
```
> ldir
kernel.sys 45815 20040417 21:19
command.com 66673 20040330 01:03
himem.exe 6838 20040409 15:18
.
.
.
```

dir

Display contents of the current remote directory in the following format:

Filename Size [*Date Time*]

Date (YYYYMMDD) and *Time* (HH:MM) fields are optional (bonus).

Example:
```
> dir
nasm.exe 259638 20030912 20:36
edit.exe 62123 20040418 22:05
.
.
.
```

get *filename*

Copy file named *filename* from current remote directory to current local directory. If *filename* does not exist in the current remote directory, you must display “Not found.” message. If *filename* already exists in the current local directory, you must overwrite it. (**Bonus**: Ask “Overwrite existing file [Y/N]?”. Overwrite existing file if answer is Y, otherwise cancel *get* operation).

Example
```
> get nasm.exe
```

```
> get win.com
Not found.
> get nasm.exe
Overwrite existing file [Y/N]? Y
```

BONUS	**put *filename***
	
	 Copy file named *filename* from current local directory to current remote directory. If *filename* does not exist in the current local directory, you must display "Not found." message. If *filename* already exists in the current remote directory, you must overwrite it. (**Bonus**: Ask "Overwrite existing file [Y/N]?". Overwrite existing file if answer is Y, otherwise cancel *put* operation).
	
	 Example
	 `> put command.com`
	 `> put win.com`
	 `Not found.`
	 `> put command.com`
	 `Overwrite existing file [Y/N]? Y`

exit

Exit program.

Server program must respond to the requests from the client program. You are free to define communication protocol used between the client and the server. Server program must display a log of messages sent from the client program.

Example:

```
Changing current working directory to misc.
Sending nasm.exe (259638 bytes).
Receiving command.com (66673 bytes).
.
.
.
```

2.3.1 Restrictions

- You must initialize COM1 to 14400 baud, 8 data bits, 1 stop bits and odd parity.

3. Lab Work

3.1 Part 1

Run your program that you've implemented for 3.1

3.2 Part 2

Run your program that you've implemented for 3.2

3.3 Part 3

Run your program that you've implemented for 3.3.