Experiment N^{o} 13

13 PC Interfacing through the Serial Ports

Introduction

In general, most PCs have two or more serial ports used to interface the PC to some peripherals, printer, modem, mouse, …etc. In this experiment you will be introduced to the interfacing of two PC's through their serial ports. You will also transfer data between the two PC's. This experiment would be of interest to students who are taking COE 342 "Data Communication" course.

Objectives

1- Serial data transmission
2- Use of Interrupt 14H
3- Port configuration.
4- Sending and receiving data through the serial ports.

13.1 Serial Port Programming

The PC serial ports may be accessed using two different ways, either through direct programming, or using INT 14H. Programming directly the serial ports requires a deep understanding of the serial port hardware. Table 13.1 gives the addresses of the different PC serial ports, and the associated interrupt numbers.

Name	Address	IRQ
COM 1	3F8 H	4
COM 2	2F8 H	3
COM 3	3E8 H	4
COM 4	2E8 H	3

Table 13.1: Standard Port Addresses

However, in this lab we are going to restrict ourselves to the use of INT 14H to access the serial port.

13.2 Interrupt 14H

INT 14h supports the four sub-functions, shown in table 13.2.

Function	Input in	Output in	Effect
00	AH	No output	Initialize Serial Port
01	AH, Character in AL	Status in AX	Write a Character to the Serial Port
02	AH	Status in AX	Read a character from the Serial Port
03	AH	Status in AX	Get status of the serial port

Table 13.2: Interrupt 14H

The serial port number (0 to 3) is specified in the DX register (0=COM1:, 1=COM2:, etc.). INT 14h expects and returns other data in the AL or AX register. The different uses of the functions of INT 14H are summarized in the following sections.

13.2.1 Serial Port Initialization

Sub-function zero initializes a serial port. This call lets you set the baud rate, select parity modes, select the number of stop bits, and the number of bits transmitted over the serial line. These parameters are all specified by the value in the AL register using the following bit encoding:

Bits	Function
0,1	Character Size
2	Stop Bits
3,4	Parity Bits
5,6,7	Baud Rate

Table 13.3.a: Control Register

Bit 1	Bit 0	Character Size
1	0	7 bits
1	1	8 bits

Table 13.3.b: Character Size Control Bits

Bit 4	Bit 3	Parity
0	0	No parity
0	1	Odd parity
1	0	No parity
1	1	Even parity

Table 13.3.c: Parity Control Bits

Bit 2	Stop bits
0	One stop bit
1	Two stop bits

Table 13.3.d: Stop Bit Control Bits

Bit 7	Bit 6	Bit 5	Baud Rate
0	0	0	110
0	0	1	150
0	1	0	300
0	1	1	600
1	0	0	1200
1	0	1	2400
1	1	0	4800
1	1	1	9600

Table 13.3.e: Baud Rate Control Bits

Example: Initialize COM1: to 2400 baud, no parity, eight bit data, and two stop bits.

```
MOV AH, 0                  ;INITIALIZE OPCODE
MOV AL, 10100111B  ;PARAMETER DATA.
MOV DX, 0                  ;COM1: PORT.
INT 14H
```

After the call to the initialization code, the serial port status is returned in AX (see Serial Port Status, AH = 3, below).

13.2.2 Transmitting a Character to the Serial Port

This function transmits the character in the AL register through the serial port specified in the DX register. On return, if AH contains zero then the character was transmitted properly. If bit 7 of AH contains one, upon return, then some sort of error occurred. The remaining seven bits contain all the error statuses returned by the GetStatus call except time out error (which is returned in bit seven). If an error is reported, you should use sub-function three to get the actual error values from the serial port hardware.

Example: Transmit a character through the COM1 port

```
MOV DX, 0          ; SELECT COM1:
MOV AL, 'A'        ; CHARACTER TO TRANSMIT
MOV AH, 1          ; TRANSMIT OPCODE
INT 14H
TEST AH, 80H       ; CHECK FOR ERROR
JNZ SERIALERROR
```

This function will wait until the serial port finishes transmitting the last character (if any) and then it will store the character into the transmit register.

13.2.3 Receive a Character from the Serial Port

Sub-function two is used to read a character from the serial port. On entry, DX contains the serial port number. On exit, AL contains the character read from the

serial port and bit 7 of AH contains the error status. When this routine is called, it does not return to the caller until a character is received at the serial port.

Example: Reading a character from the COM1 port

```
MOV DX, 00          ; SELECT COM1:
MOV AH, 02          ; RECEIVE OPCODE
INT 14H
TEST AH, 80H        ; CHECK FOR ERROR
JNZ SERIALERROR
<RECEIVED CHARACTER IS NOW IN AL>
```

13.2.4 Serial Port Status

This call returns status information about the serial port including whether or not an error has occurred, if a character has been received in the receive buffer, if the transmit buffer is empty, and other pieces of useful information. On entry into this routine, the DX register contains the serial port number. On exit, the AX register content is shown in table 13.4.

AX	Bit Meaning
0	Clear to send
1	Data set ready
2	Trailing edge ring detector
3	Receive line signal detect
4	Clear to send (CTS)
5	Data set ready (DSR)
6	Ring indicator
7	Receive line signal detect
8	Data available
9	Overrun error
10	Parity error
11	Framing error
12	Break detection error
13	Transmitter holding register empty
14	Transmitter shift register empty
15	Time out error

Table 13.4: Getting the status of a serial port

There are a couple of useful bits, not pertaining to errors, returned in the status information. If the data available bit is set (bit #8), then the serial port has received data and you should read it from the serial port. The Transmitter holding register empty bit (bit #13) tells you if the transmit operation will be delayed while waiting for the current character to be transmitted or if the next character will be immediately transmitted. By testing these two bits, you can perform other operations while waiting for the transmit register to become available or for the receive register to contain a character.

References:

1- Barry B. Brey, “Programming the 80286, 80386, 80486, and Pentium-Based Personal Computer”, Prentice Hall, (1996).
2- Randall Hyde, “The Art of Assembly Language Programming”, http://webster.cs.ucr.edu/Page_asm/ArtofAssembly/ArtofAsm.html
3- COE 342 "Data Communication" lecture notes.

13.3 Lab Work

Pre Lab Work:

1- Read and understand material related to INT 14h.
2- Write program 13.1 and understand what the program is doing.
3- Write program 13.2 and understand what the program is doing.

Lab Work:

1- To be able to run the programs, you need to connect the PC's through their serial ports COM1 or COM2, with serial cables terminated by D9 type connectors. You will be supplied with such cables in the lab. You need also to work in groups of two.
2- Run program 13.1 and notice what the program is doing.
3- Run program 13.2 and notice what the program is doing
4- Modify program 13.1 to make it send at a baud rate of 9600.
5- Modify program 13.1 to make it send a string of 3 characters.
6- Modify program 13.2 to make it receive a string of 3 characters at a baud rate of 9600.

Lab Assignment:

Develop a program that displays a menu consisting of the following choices:

- **Configure Serial Port:** prompts the user to enter one of the standard baud rate values. The program then configures the COM1 port with the given baud rate. (The rest of the configuration with default values)
- **Send String:** prompts the user to enter the length of the string and then enter the string character by character. The program sends each character as soon as it is entered.
- **Receive String:** prompts the user to enter the length of the string. It then receives the string character by character and displays it on the screen.
- **Send File (Optional):** prompts the user to enter the location of the file and then sends that file. (Needs more information about handling files. Ask the instructor for References to that information.)
- **Receive File (Optional):** receives the file. (Needs more information about handling files. Ask the instructor for References to that information.)

TITLE ‘Program 13.1’
;This program configures the serial port (COM1) and sends a character through that port.

```
.MODEL SMALL
.STACK 200

.DATA
      PROMPT      DB    ‘Enter the character to be sent’,0DH,0AH,’$’
      SENT        DB    ‘The   character   has   successfully   been
sent’,’$’
      ERROR     DB      ‘An error occurred. Please check your hardware and try
again.’,’$’

.CODE
.STARTUP

      MOV AH, 0H              ; INITIALIZE OPCODE
      MOV AL, 10100111B       ; PARAMETER DATA.
      MOV DX, 0H              ; COM1: PORT.
      INT 14H

      LEA DX, PROMPT          ; DISPLAY PROMPT ON SCREEN
      MOV AH, 09H
      INT 21H

      MOV AH, 01H             ; READ THE CHARACTER FROM KEYBOARD
      INT 21H

      MOV DX, 0H              ; SELECT COM1:
      MOV AH, 1H              ; TRANSMIT OPCODE
      INT 14H
      TEST AH, 80H            ; CHECK FOR ERROR
      JNZ PORTERROR

      LEA DX, SENT            ; DISPLAY THAT THE CHARACTER WAS SENT
      MOV AH, 09H
      INT 21H
      JMP FINISH

      PORTERROR:
      LEA DX, ERROR           ; DISPLAY THAT THERE WAS AN ERROR
      MOV AH, 09H
      INT 21H

      FINISH:

.END
```

TITLE ‘Program 13.2’
;This program configures the serial port (COM1) and receives a character through that port.

```
.MODEL SMALL
.STACK 200

.DATA
      PROMPT      DB    ‘Receiving          character.          Please
wait.’,0DH,0AH,’$’
      RECEIVED    DB    ‘The        character        has        been
received.’,0DH,0AH,’$’
      ERROR       DB    ‘An error occurred. Please check your hardware and try
again.’,’$’
      CHAR        DB    ‘The received character is: ‘,’$’

.CODE
.STARTUP

      MOV AH, 0H              ; INITIALIZE OPCODE
      MOV AL, 10100111B       ; PARAMETER DATA.
      MOV DX, 0H              ; COM1: PORT.
      INT 14H

      LEA DX, PROMPT          ; DISPLAY PROMPT ON SCREEN
      MOV AH, 09H
      INT 21H

      MOV DX, 0H              ; SELECT COM1:
      MOV AH, 02H             ; RECEIVE OPCODE
      INT 14H
      TEST AH, 80H            ; CHECK FOR ERROR
      JNZ PORTERROR

      LEA DX, RECEIVED        ;   DISPLAY   THAT   THE   CHARACTER   WAS
RECEIVED
      MOV AH, 09H
      INT 21H

      LEA DX, CHAR            ; DISPLAY CHAR MESSAGE
      MOV AH, 09H
      INT 21H

      MOV DL, AL              ; DISPLAY THE CHARACTER ON SCREEN
      MOV AH, 02H
      INT 21H
      JMP FINISH

      PORTERROR:
      LEA DX, ERROR           ; DISPLAY THAT THERE WAS AN ERROR
      MOV AH, 09H
      INT 21H

      FINISH:

.END
```