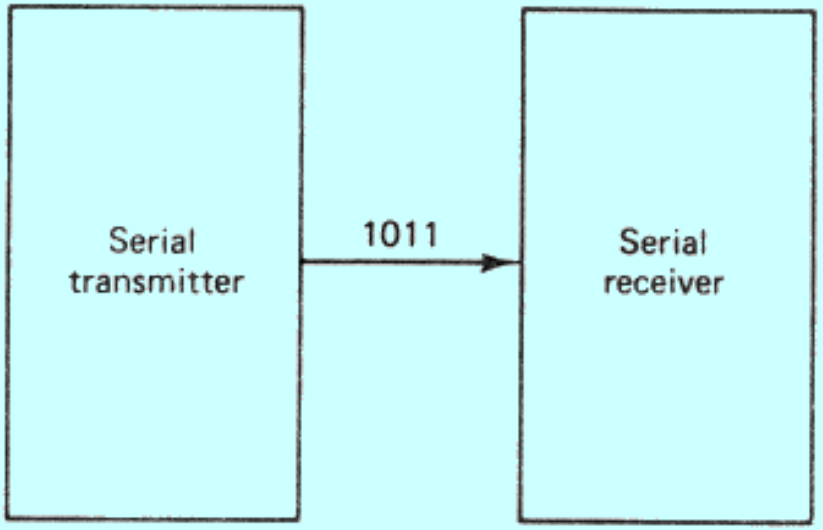
C9 SERIAL PORT (COM:)

Outline

SERIAL COMMUNICATIONS

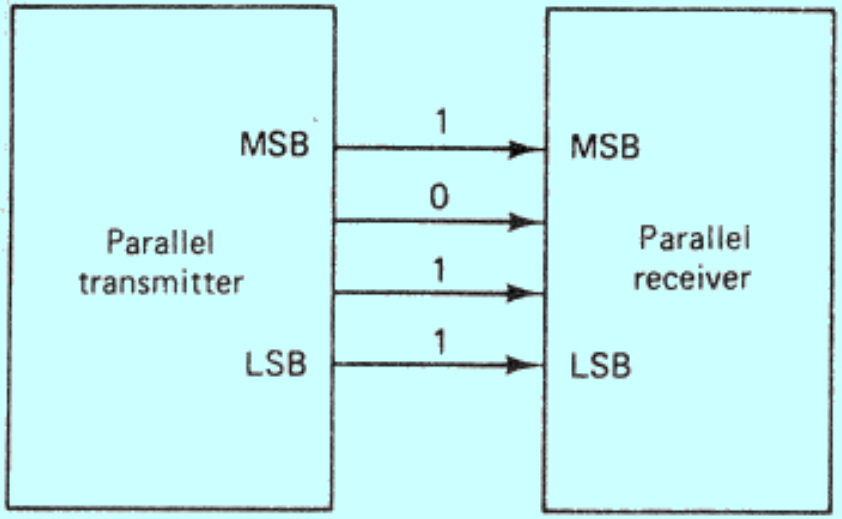
1. RS232/EIA 232 SIGNALS
2. PC UART's Roadmap
3. UART ARCHITECTURE. PINS
4. PROGRAMMING. REGISTERS
5. BIOS FUNCTIONS FOR SERIAL PORT (INT 14h)
6. APPLICATIONS

- 4 clock for 4 bit transfer



Parallel vs. Serial

- 1 clock for 4 bit transfer



- At first glance, it would seem that parallel ports should be able to send data much faster than serial ports.

REMARKS.

- However, parallel ports suffer extremely from **inter-symbol interference** (ISI) and noise, and therefore the data can be corrupted over long distances.
- Also, because the wires in a parallel system have small amounts of capacitance and mutual inductance, ***the bandwidth of parallel wires is much lower than the bandwidth of serial wires***.
- an increased bandwidth leads to a better bit rate.
- less noise in the channel means we can successfully transmit data reliably with a lower Signal-to-Noise Ratio, SNR.
- If, however, we bump up the power in a serial connection by using a differential signal with 2 wires (one with a positive voltage, and one with a negative voltage), we can use the same amount of power, have twice the SNR, and reach an even higher bitrate without suffering the effects of noise.
- USB cables, for instance, use shielded, differential serial communications, and the USB 2.0 standard is capable of data transmission rates of 480Mbits/sec!
- In addition, because of the increased potential for noise and interference, parallel wires need to be far shorter than serial wires.
- Consider the standard parallel port wire to connect the PC to a printer: those wires are between 3 and 4 feet long, and the longest commercially available is typically 25 feet. Now consider ethernet wires (which are serial, and typically unshielded twisted pair): they can be bought in lengths of 100 -300 foot

RS 232

Defined in early 1960s
Serial, Asynchronous, Full-duplex,
Voltage-based, point-to-point, 100 ft+ cables

+12V
+3V } SPACE = 0

−3V
−12V } MARK = 1

Interface	Format	Number of Devices (maximum)	Distance (maximum, ft)	Speed (maximum, bps)	Typical Use
RS-232 (TIA-232)	asynchronous serial	2	50-100	20k (faster with some hardware)	Modem, basic communications
RS-485 (TIA-485)	asynchronous serial	32 unit loads (up to 256 devices with some hardware)	4000	10M	Data acquisition and control systems
Ethernet	serial	1024	1600	10G	PC network communications
IEEE-1394b (FireWire 800)	serial	64	300	3.2G	Video, mass storage
IEEE-488 (GPIB)	parallel	15	60	8M	Instrumentation
I^2C	synchronous serial	40	18	3.4M	Microcontroller communications
Microwire	synchronous serial	8	10	2M	Microcontroller communications
MIDI	serial current loop	2 (more with flow-through mode)	50	31.5k	Music, show control
Parallel Printer Port	parallel	2 (8 with daisy-chain support)	10–30	8M	Printer
SPI	synchronous serial	8	10	2.1M	Microcontroller communications
USB	asynchronous serial	127	16 (up to 98 ft with 5 hubs)	1.5M, 12M, 480M	PC peripherals

Basics of serial communications

- **Symbol period (T_s)** – the period when the signal features don 't change (may be 1 bit or combinations of bits)

- **Modulation speed (S_m)** – nr. Symbols/ sec.

$$s_m = 1/T_s \text{ (baud)}$$

- **Binary rate (D)** – nr. of binary symbols/sec - standardized #

$$D=1/T \text{ (bps) ; T=bit duration}$$

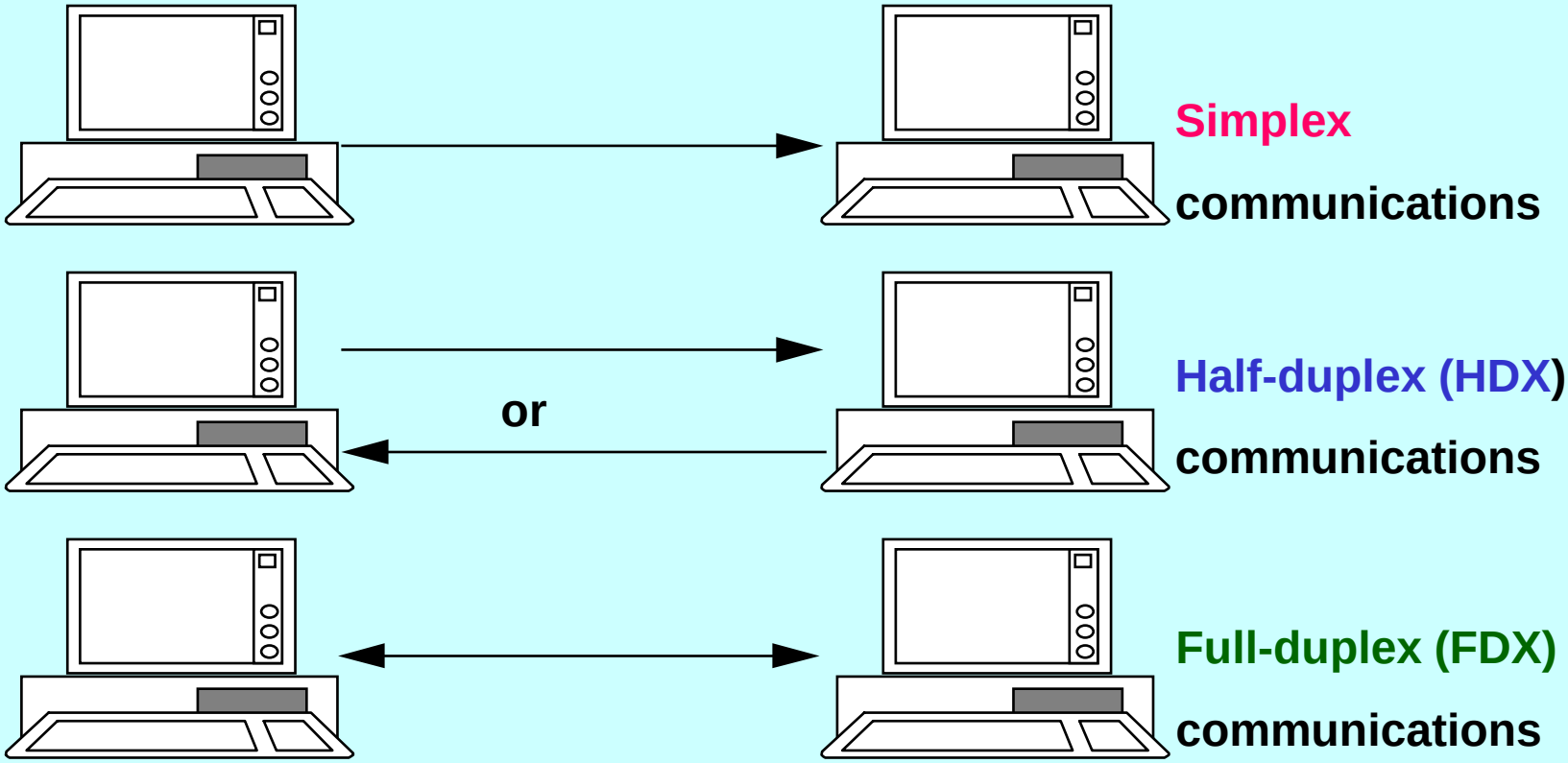
-using some tricks more bits may be defined in every T_S by fixing more features for signal and more levels for each feature, so if we have n – states for the modem :

$$D= s_m \log_2 n \text{ [bps]}$$

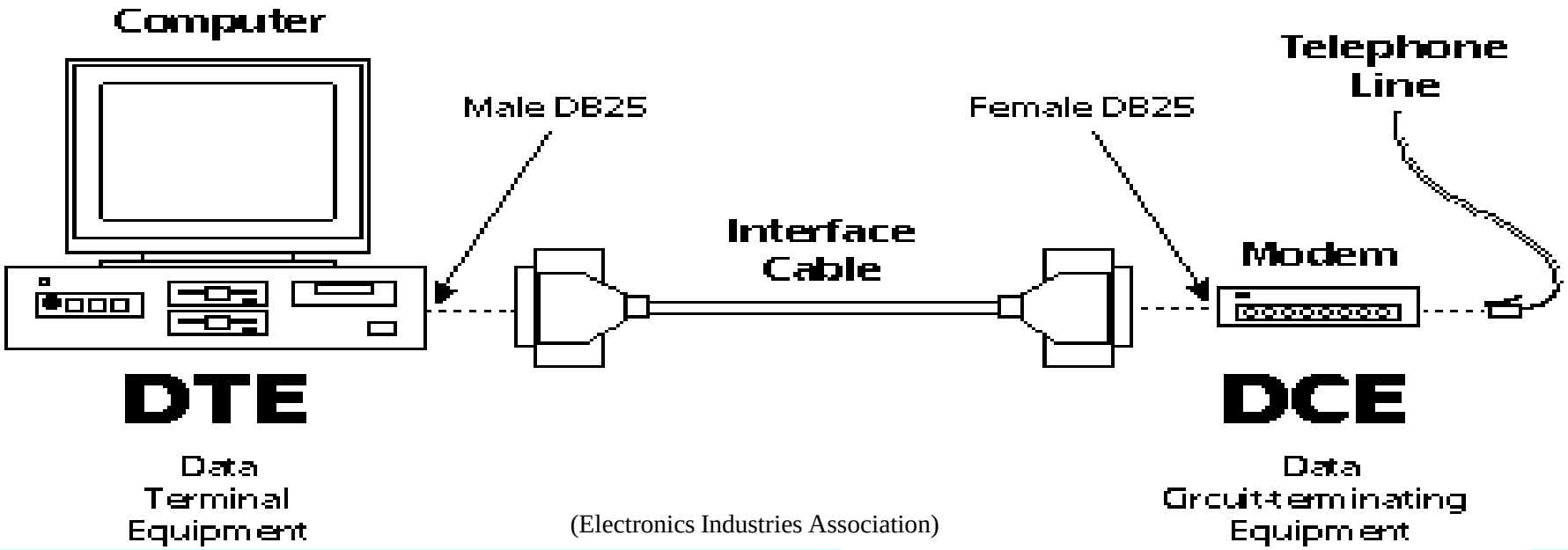
$n=2 \rightarrow D = s_m$

Ex. FAX (amplitude, phase) – more values for every feature, it may code more bits in every symbol period : D=9600 bps , sm = 1200 baud → n = 2^8 #

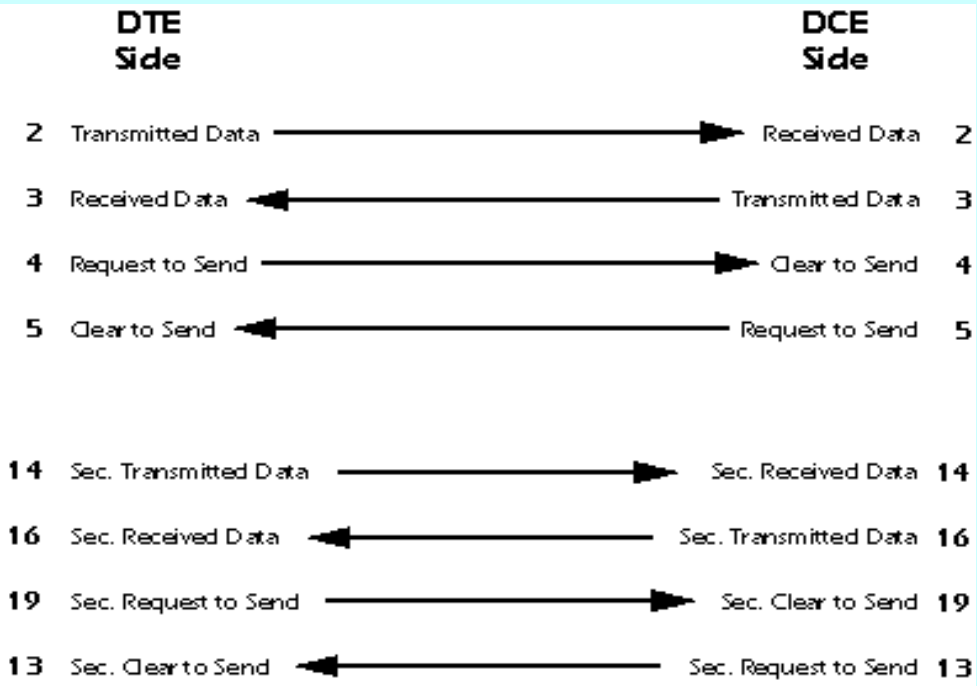
Modes of Communication

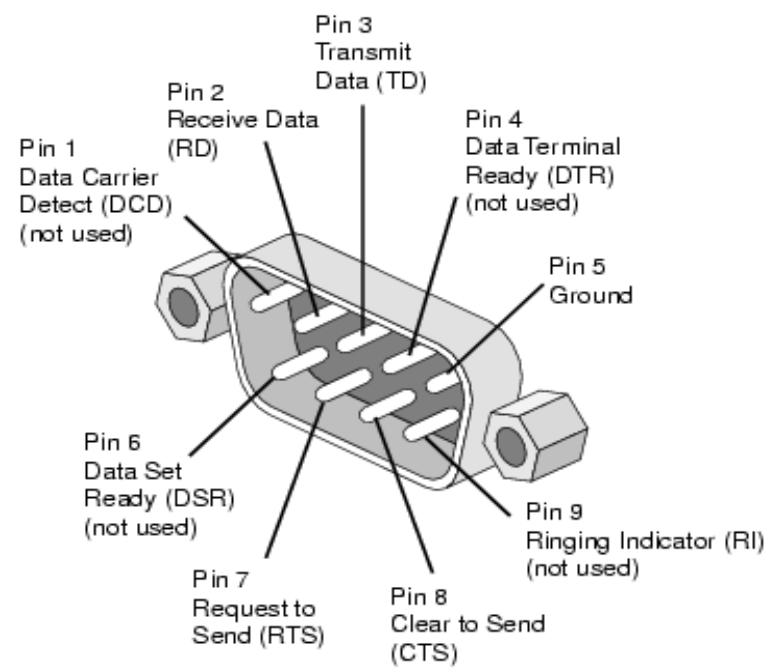


1. RS 232/EIA 232 interface signals description

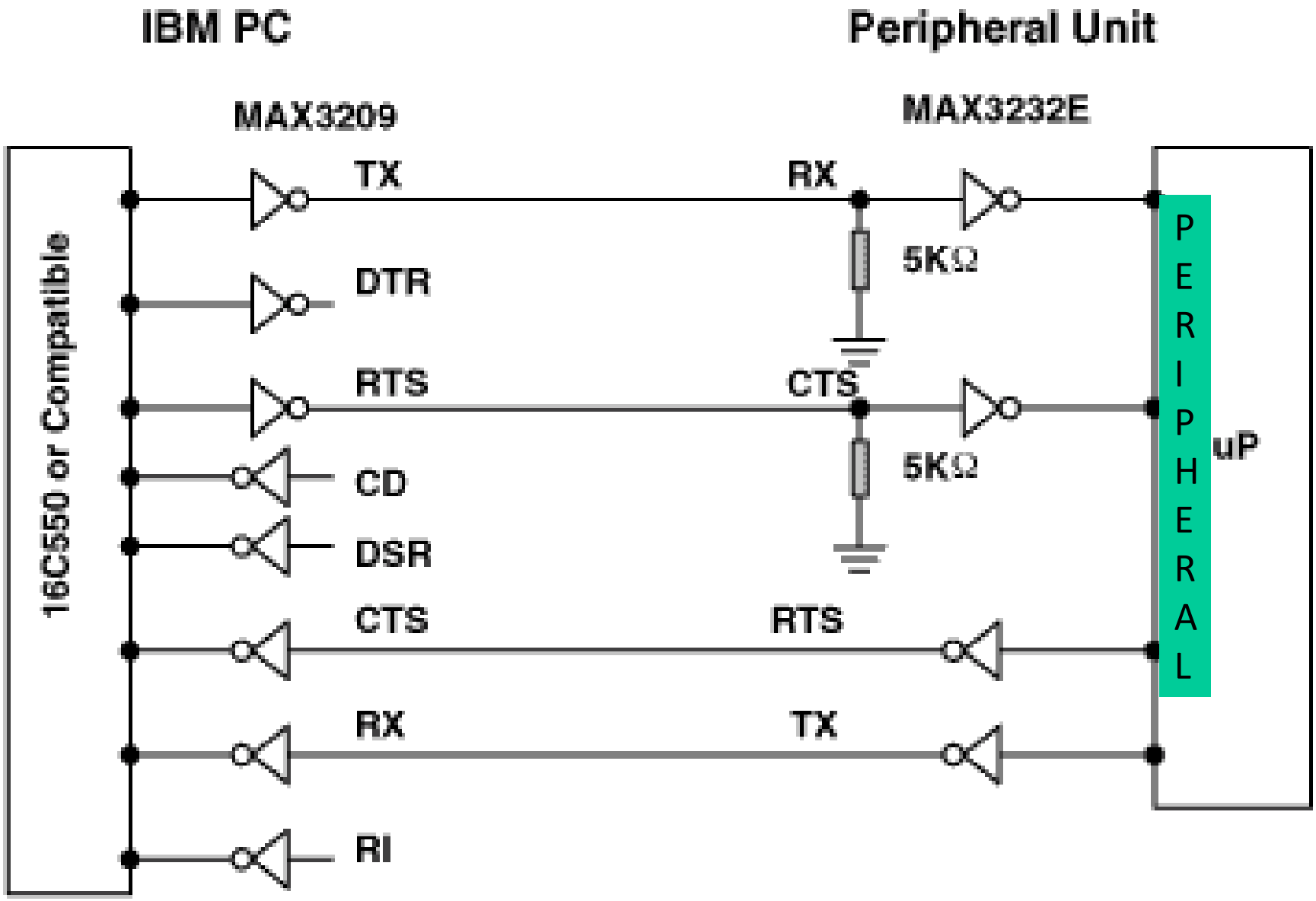


(Electronics Industries Association)





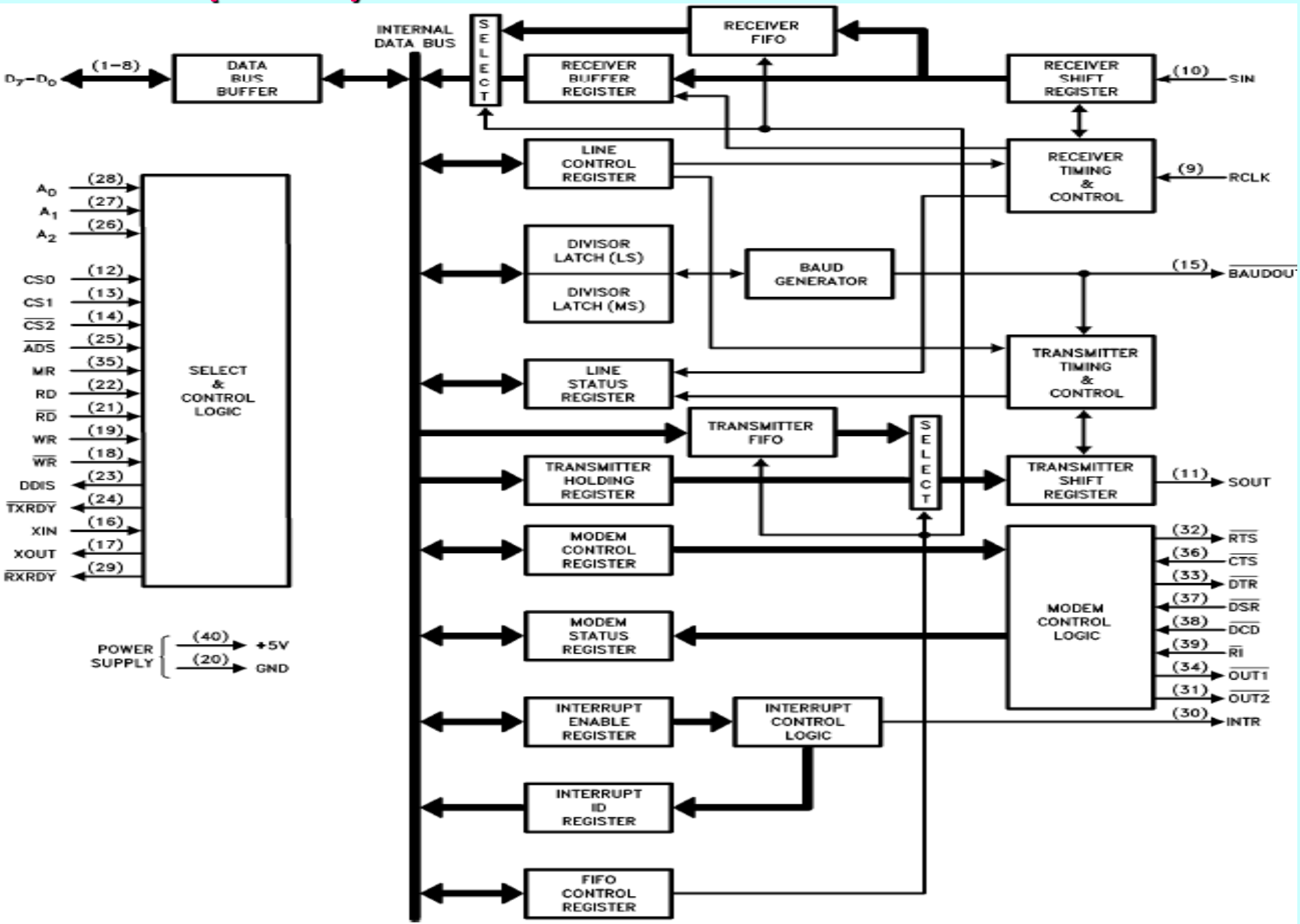
Signal Pin (DB25/DB9)	Name	Function
TxD (2/3)	Transmit Data (Transmisie date)	Data output (TXD) -OUT
RxD (3/2)	Receive Data (Receptie date)	Data input (RXD)-IN
CTS (5/8)	Clear to Send (Modem gata de emisie)	Modem (DCE) is ready to change data-IN
DCD (8/1)	Data Carrier Detect (Detectare purtatoare de semnal)	Carrier detect in modem line -IN
DSR (6/6)	Data Set Ready (Modem operational)	Modem is ready for communication-IN
DTR (20/4)	Data Terminal Ready (Terminal de date gata)	UART is ready for connection -OUT
RTS (4/7)	Request To Send (Cerere de emisie)	DTE is ready to change data - OUT
RI (22/9)	Ring Indicator (Indicator apel)	RI=1 when DCE detect a ring signal from PSTN. - IN (public switched telephone network)


IBM PC
Peripheral Unit
MAX3209
MAX3232E
16C550 or Compatible
TX
DTR
RTS
CD
DSR
CTS
RX
RI
RX
5KΩ
CTS
5KΩ
RTS
TX
PERIPHERAL
uP

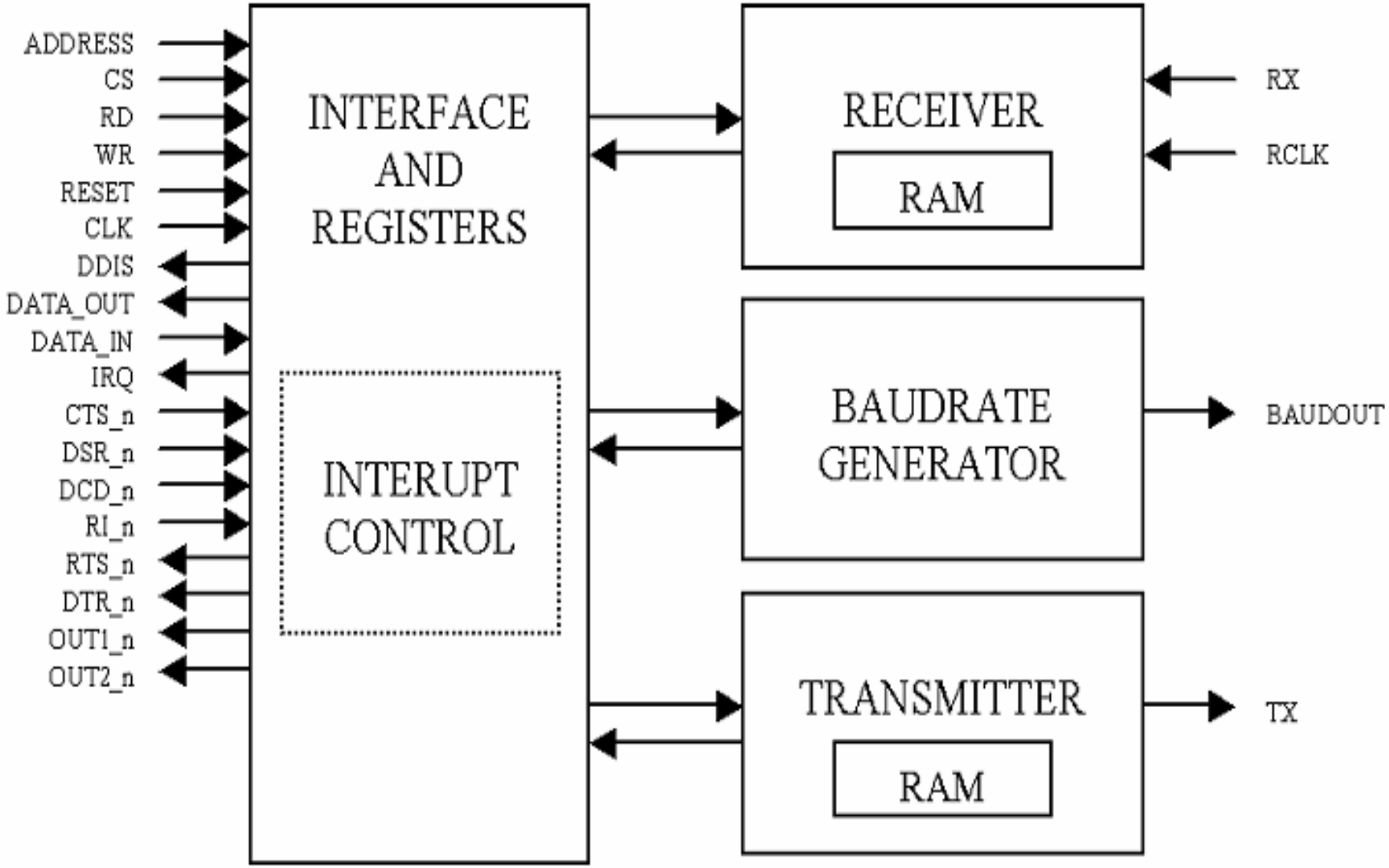
2. PC UART's Roadmap

- **8250 UART** was the original UART and was able of speeds up to 9600 bps with a 1-byte FIFO
- **8250A UART** was a revised version of the 8250 with an additional register that allowed software to verify it
- **16450 UART** Slightly faster then earlier UARTs.
- **16550 UART** has a 16-byte FIFO (FIFO bug)
- **16550A UART** had same features as previous 16550 UART with bug fixed
- **16650 UART** has a 32-byte FIFO 460.8 Kbps
- **16750 UART** has a 64-byte FIFO 921.6 Kbps
- **16950 UART** has a 128-byte FIFO
- **Hayes ESP (Enhances Serial Ports)** - 8 bit card with 2 UART's 16550A and a processor that allows DMA transfer

3. UART (16550) ARCHITECTURE



UART


ADDRESS
CS
RD
WR
RESET
CLK
DDIS
DATA_OUT
DATA_IN
IRQ
CTS_n
DSR_n
DCD_n
RI_n
RTS_n
DTR_n
OUT1_n
OUT2_n
INTERFACE AND REGISTERS
INTERUPT CONTROL
RECEIVER
RAM
RX
RCLK
BAUDRATE GENERATOR
BAUDOUT
TRANSMITTER
RAM
TX

Select & Control Block -This block does the interfacing tasks

Line Control & Status Register Block - This block includes the Line Control Register (LCR) and the Line Status Registers (LSR).

Modem Control Block - The Modem Control Block incorporates the Modem Control Register(MCR), the Modem Status Register(MSR) and the Scratch Pad Register(SPR)

Interrupt Control Block- This block generate the external interrupt signal (intr)

FIFO Control Block - This block includes the FIFO Control Register

Baud Rate Generator Block - This block includes the lower(dl low) and the upper (dl high) divisor latches

Transmitter Block & Transmit FIFO- The transmitter block transmits the data onto the external Serial out signal

Receiver Block & FIFO - The receiver section receives the data onto the serial input signal

Factory Test Register - The Factory Test Register is provided to test the Baud Rate generator, to check the parity error and to check the framing error.

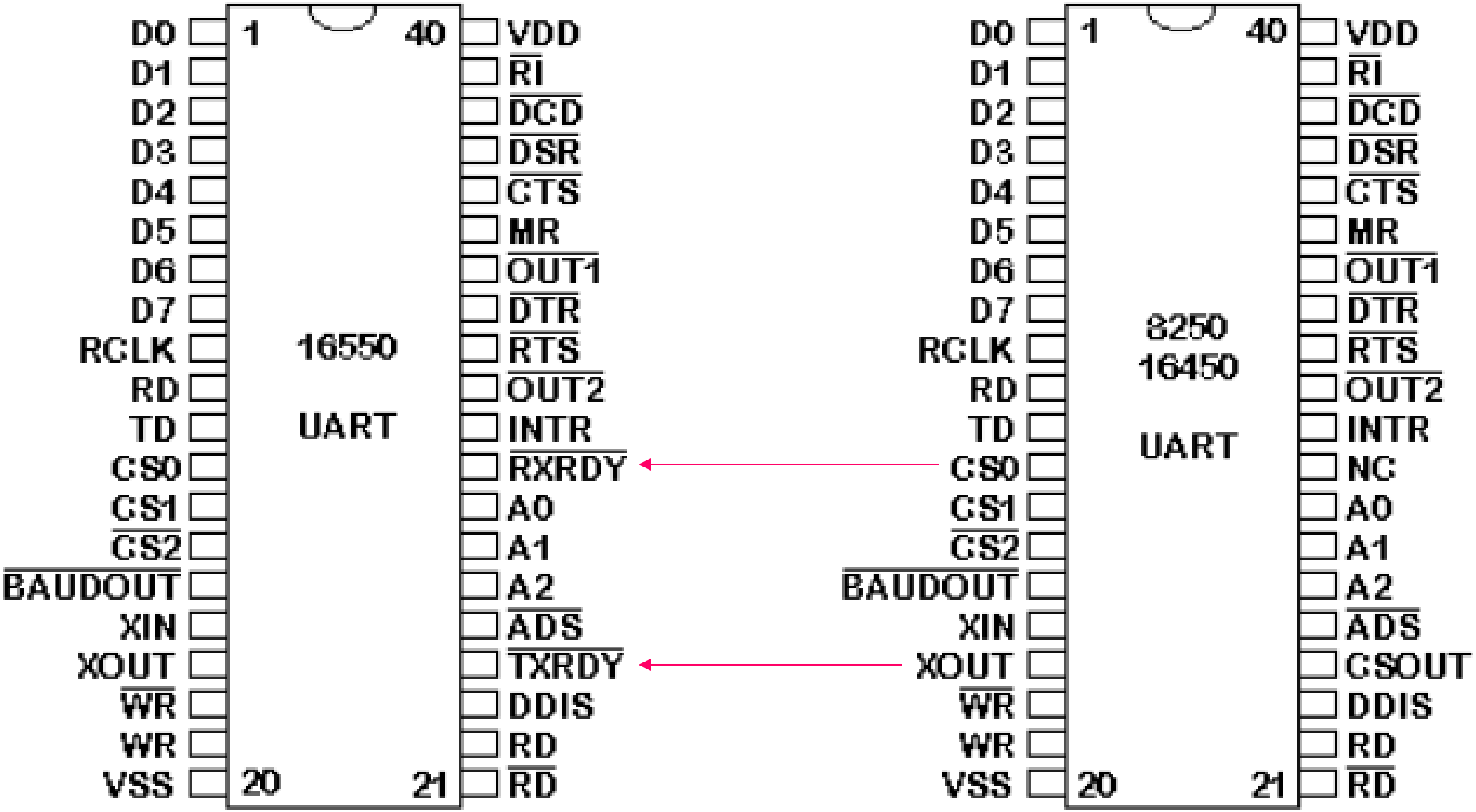
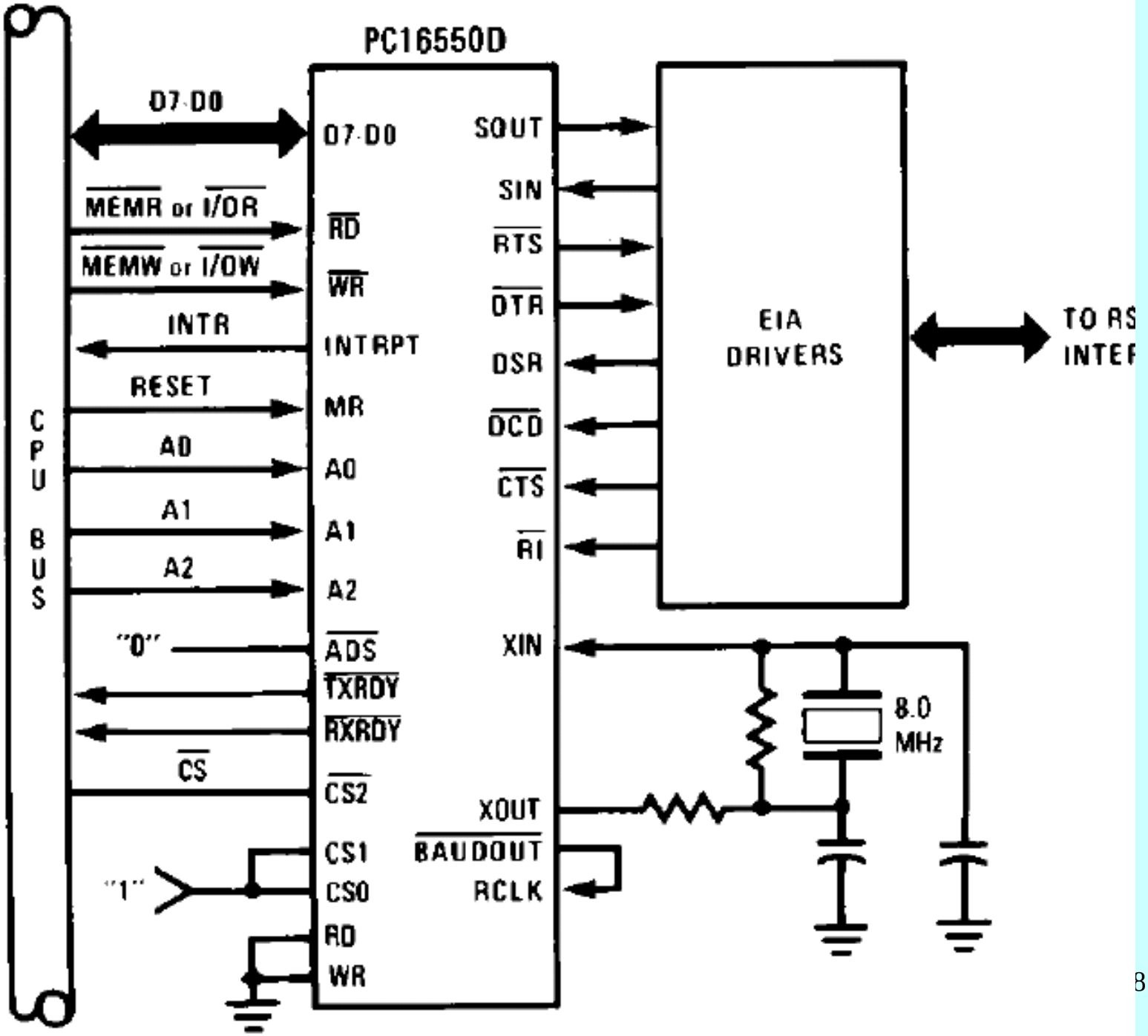


Figure 8 : Pin Diagrams for 16550, 16450 & 8250 UARTs

Pin No.	Name	Notes
Pin 1:8	D0:D7	Data Bus
Pin 9	RCLK	Receiver Clock Input. The frequency of this input should equal the receivers baud rate * 16
Pin 10	RD	Receive Data
Pin 11	TD	Transmit Data
Pin 12	CS0	Chip Select 0 - Active High
Pin 13	CS1	Chip Select 1 - Active High
Pin 14	nCS2	Chip Select 2 - Active Low
Pin 15	nBAUDOUT	Baud Output - Output from Programmable Baud Rate Generator. Frequency = (Baud Rate x 16)
Pin 16	XIN	External Crystal Input - Used for Baud Rate Generator Oscillator
Pin 17	XOUT	External Crystal Output
Pin 18	nWR	Write Line - Inverted
Pin 19	WR	Write Line - Not Inverted
Pin 20	VSS	Connected to Common Ground
Pin 21	RD	Read Line - Inverted
Pin 22	nRD	Read Line - Not Inverted
Pin 23	DDIS	Driver Disable. This pin goes low when CPU is reading from UART. Can be connected to Bus Transceiver in case of high capacity data bus.
Pin 24	nTXRDY	Transmit Ready
Pin 25	nADS	Address Strobe. Used if signals are not stable during read or write cycle
Pin 26	A2	Address Bit 2
Pin 27	A1	Address Bit 1
Pin 28	A0	Address Bit 0

Pin 29	nRXRDY	Receive Ready
Pin 30	INTR	Interrupt Output
Pin 31	nOUT2	User Output 2
Pin 32	nRTS	Request to Send
Pin 33	nDTR	Data Terminal Ready
Pin 34	nOUT1	User Output 1
Pin 35	MR	Master Reset
Pin 36	nCTS	Clear To Send
Pin 37	nDSR	Data Set Ready
Pin 38	nDCD	Data Carrier Detect
Pin 39	nRI	Ring Indicator
Pin 40	VDD	+ 5 Volts

Table 2 : Pin Assignments for 16550A UART


PC16550D
D7-D0
D7-D0
MEMR or I/OR
RD
MEMW or I/OW
WR
INTR
INTRPT
RESET
MR
A0
A0
A1
A1
A2
A2
"0"
ADS
TXRDY
RXRDY
CS
CS2
CS1
"1"
CS0
RD
WR
SOUT
SIN
RTS
DTR
DSR
DCD
CTS
RI
XIN
XOUT
BAUDOUT
RCLK
CPU BUS
EIA DRIVERS
TO RS
INTER
8.0 MHz

ADDRESSES & INTERRUPTS RESERVED FOR SERIAL PORTS

Name	Base addr.	Intr. (IRQ)	COM addr. in BIOS area
COM 1	3F8	4	0000:0400
COM 2	2F8	3	0000:0402
COM 3	3E8	4	0000:0404
COM 4	2E8	3	0000:0406

```
#include <stdio.h>
 #include <dos.h>
    void main(void)
    {
     unsigned int far *indadr;  /* Indicator catre locatia adresei portului */
     unsigned int adr_port;       /* Adresa portului */
     int i;
     indadr=(unsigned int far *)0x00000400;
for (i = 0; i <  4; i++)
    {
     adr_port = *indadr;
  if (adr_port == 0)
            printf("Nu s-a gasit nici un port pentru COM%d \n",i+1);
     else
printf("Adresa alocata pentru COM%d is %Xh\n",i+1,adr_port);

    *indadr++;
        }
    }
```

4. UART PROGRAMMING. 8250/16550 Registers

Address COM1 (COM2)	R/W	Abr.	Register name
3F8h (2F8h)	Write	**THR**	Transmit Holding Buffer (Buffer de transmisie)
	Read	**RBR**	Receive Buffer (Buffer de receptie)
	Read/ Write	**-**	Divisor Latch Low Byte (Registru divizor, octetul „low”, dacă b7 al registrului de control al liniei este 1)
3F9h (2F9h)	Read/ Write	**IER**	Interrupt Enable Register (Registru de validare a întreruperilor)
	Read/ Write	**-**	Divisor Latch High Byte (Registru divizor, octetul “high”, dacă b7 al registrului de control al liniei este 1)
3FAh (2FAh)	Read	**IIR**	Interrupt Identification Register (Registru de identificare a întreruperilor)
	Write	**FCR**	FIFO Control Register (Registru de control FIFO)-numai la 16550
3FBh (2FBh)	Read/ Write	**LCR**	Line Control Register (Registru de control linie)
3FCh (2FCh)	Read/ Write	**MCR**	Modem Control Register (Registru de control modem)
3FDh (2FDh)	Read	**LSR**	Line Status Register (Registru de stare linie)
3FEh (2FEh)	Read	**MSR**	Modem Status Register (Registru de stare modem)
3FFh (2FEh)	Read/ Write	**SPR**	Scratchpad Register – only for 16550

THR – Transmit Holding Register (R/W)
TSR – Transmit Shift Register
RBR – Receiver Buffer Register (R/W)

D7	D6	D5	D4	D3	D2	D1	D0

IER – Interrupt Enable Register (R/W)

0	0	LPM*	SM*	MODEM STATUS CHANGE	RECEIVE LINE STATUS CHANGE	TRANSMIT HOLDING REG. EMPTY	RECEIVE DATA AVAILABLE

Bit	Description
7	-
6	-
5	Enable Low Power Mode (16750)
4	Enable Sleep Mode (16750)
3	Interrupt gen. To MSR changing
2	Enable Interrupt gen. To LSR changing
1	Enable Interrupt gen. To THR empty
0	Enable Interrupt gen. when character is received

Divisor Latch Register LSB, MSB (R/W) **Divisor = 1,843,200 / (Binary Debit* 16)**

Binary debit (bps)	Divisor	Binary debit(bps)	Divisor
150	300h	4800	18h
300	180h	7200	10h
600	C0h	9600	0Ch
1200	60h	19200	06h
2400	30h	38400	03h
3600	20h	115200	01h

IIR – Interrupt Idendification Register (R)

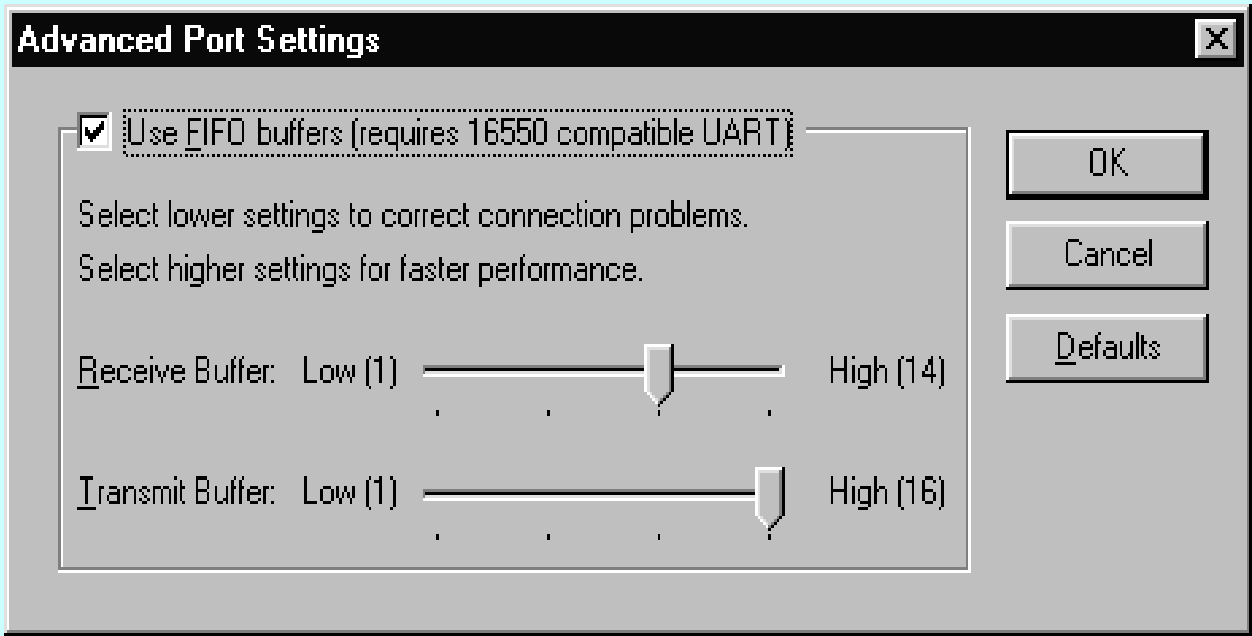
FIFO ENABLED (16550)	FIFO ENABLED (16550)	0	0	INTERRUPT ID 2 (16550)	INTERRUPT ID 1	INTERRUPT ID 0	PEND

	IID2	IID1	IID0	PEND
No Interrupt Pending	X	X	X	1
Receive Line Status Change	0	1	1	0
Receive Data Available	0	1	0	0
Receive Timeout (16550)	1	1	0	0
Transmit Holding Reg. Empty	0	0	1	0
Modem Status Change	0	0	0	0

FCR - FIFO Control Register (W)

TRIGGER 1	TRIGGER 0	0	0	DMA MODE	TXFIFO RESET	RXFIFO RESET	ENABLE

Bit	Function		
7-6	B7	B6	Interrupt triggering
	0	0	1 Byte
	0	1	4 Bytes
	1	0	8 Bytes
	1	1	14 Bytes
5	FIFO enabled on 64 bytes (only for 16750)		
4	Reserved		
3	Select DMA mode. PC not accept.		
2	Delete transmition FIFO buffer		
1	Delete reception FIFO buffer		
0	Enable FIFO		



LCR – Line Control Register (R/W)

DIVISOR LATCH ACCESS BIT	BREAK	STICK	EVEN PARITY SELECT	PARITY ENABLE	STOP BIT SELECT	WORD LENGTH SELECT 1	WORD LENGTH SELECT 0

Bit	Description			
7	1	DLAB provides access to baud rate divisor latch		
	0	Reception buf.,transmision buf. and IER reg. access		
6		Break Enable		
5-3	**B5**	**B4**	**B3**	Paritaty select
	X	X	0	No parity
	0	0	1	Odd parity
	0	1	1	Even parity
	1	0	1	Zero Parity
	1	1	1	One parity
2	Stop bit lenght			
	0	1 bit de stop		
	1	2 stop bits for characters of 6/7/8 bits or 1.5 bits for 5 bits characters		
1-0	B1	B0	Character lenght	
	0	0	5 Bits	
	0	1	6 Bits	
	1	0	7 Bits	
	1	1	8 Bits	

MCR – Modem Control Register (R/W)

0	**0**	**0**	**LOOPBACK TEST**	**OUT1**	**OUT2**	**REQUEST TO SEND**	**DATA TERMINAL READY**

Bit	Description
7	Reserved
6	Reserved
5	Reserved
4	LoopBack Mode (DTR,RTS, RI and DCD Not Connected)
3	AUX OUT2
2	AUX OUT1
1	RTS
0	DTR

LSR – Line Status Register (R)

RXFIFO HOLDS ERROR (16550)	TRANSMIT EMPTY	TRANSMIT HOLDING REG. EMPTY	BREAK DETECTED	FRAMING ERROR	PARITY ERROR	OVERRUN ERROR	RECEIVE DATA AVAILABLE

Bit	Description
0	Data available
1	Overrun error
2	Parity error
3	Framing error
4	Break signal received
5	THR is empty
6	THR is empty, and line is idle
7	Errornous data in FIFO

MSR – Modem Status Register (R)

DATA CARRIER DETECT	DATA SET READY	RING INDICATE	CLEAR TO SEND	DELTA DCD	DELTA DSR	DELTA RI	DELTA CTS

Scratchpad Register (16550) (R/W)

D7	D6	D5	D4	D3	D2	D1	D0

BAUD RATE GENERATOR PROGRAMMING TABLE All boards will typically have the 1.8432 MHZ crystal. Some boards will also allow selection of **the 7.3728 MHZ clock.**

BAUD RATE	1.8432 MHZ	7.3728 MHZ
50	2304	9216
75	1536	6144
110	1047	4188
134.5	857	3428
150	768	3072
300	384	1536
600	192	768
1200	96	384
2400	48	192
3600	32	128
4800	24	96
7200	16	64
9600	12	48
19.2K	6	24
38.4K	3	12
57.6K	2	8
115.2K	1	4
230.4K	not possible	2
460.8K	not possible	1

5. BIOS SERVICES for the SERIAL PORT (INT 14h)

AH (Line status)

Bit	Description
7	(Time-Out Error): Time-out for transmission or reception
6	(Transmitter Shift Register Empty): the character from TSR is transmitted
5	(Transmitter Holding Register Empty): the register is empty;
4	(Break Detect): the line is kept to "0"
3	(Framing Error): the stop bit is not received;
2	(Parity Error): parity error;
1	(Overrun Error): overrun error
0	(Data Ready): a character is stored in the Reception register

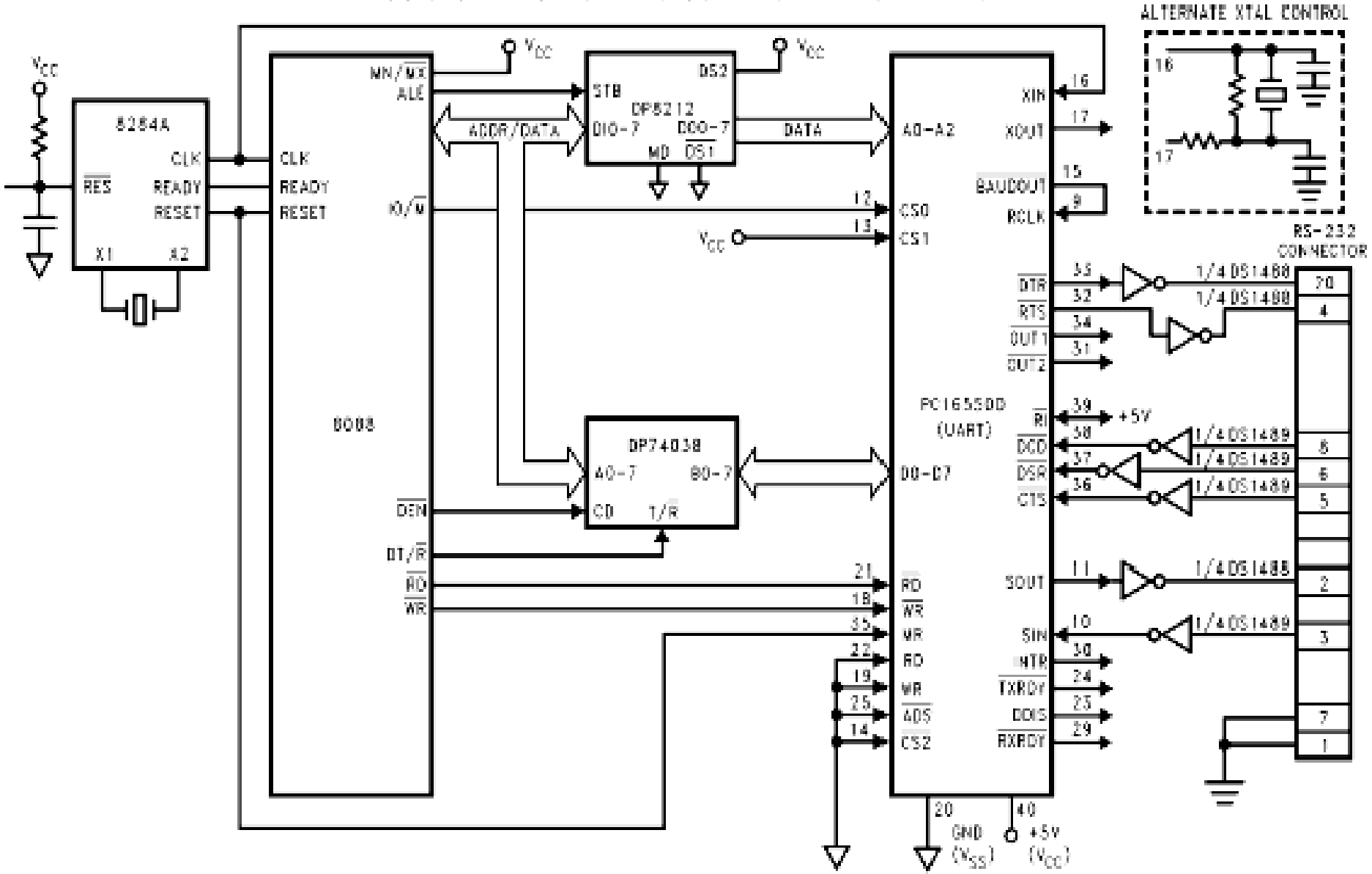
AL (Modem status)

Bit	Description
7	(Carrier Detect): CD status
6	(Ring Indicator): RI status;
5	(Data Set Ready): DSR status;
4	(Clear To Send) : CTS status;
3	Delta Carrier Detect):change of the DCD status
2	(Trailing Edge Ring Indicator):RI = 1
1	(Delta Data Set Ready): change of the DSR status
0	(Delta Clear To Send): change of the CTS status

Service	Function	Input	Output
AH=00	**Serial Port Initialization**	AL = Comunications parameters DX = serial port nr. (0-3)	AH = Line Status AL = Modem Status
AH=01	**Transmit a Character to the Serial Port**	AL = character DX = serial port nr. (0-3)	AH: b7=0 if correct b7=1 if error AH b6..0 = status
AH=02	**Receive a Character from the Serial Port**	DX = serial port nr. (0-3)	AH: b7=0 if correct b7=1 if error AH b6..0 = status AL=character
AH=03	**Serial Port Status**	DX = serial port nr. (0-3)	AH = Line Status AL = Modem Status

Bits 1-0 : caracter length (10 - 7 bits; 11 - 8 bits)
Bit 2 : stop bits (0- 1 stop bit ; 1 - 2 stop bit)
Bits 4–3 : parity control (01- p. odd, 11 – p.even, x0 no parity)
Biţi 7–5 : baud rate (000 -110 bps, ….111 -9600 bps)

This shows the basic connections of an PC16550D to an 8088 CPU

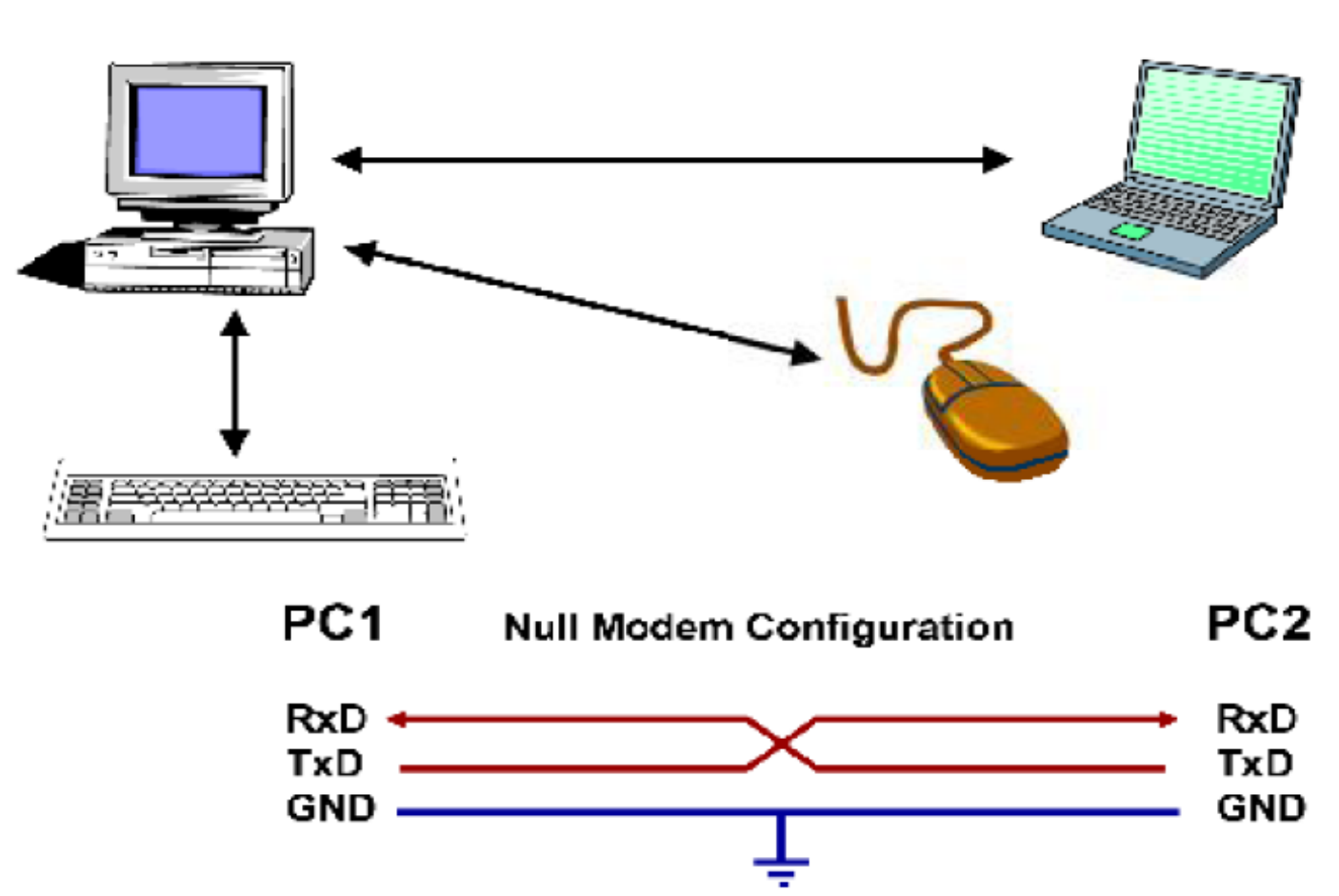


6. Applications

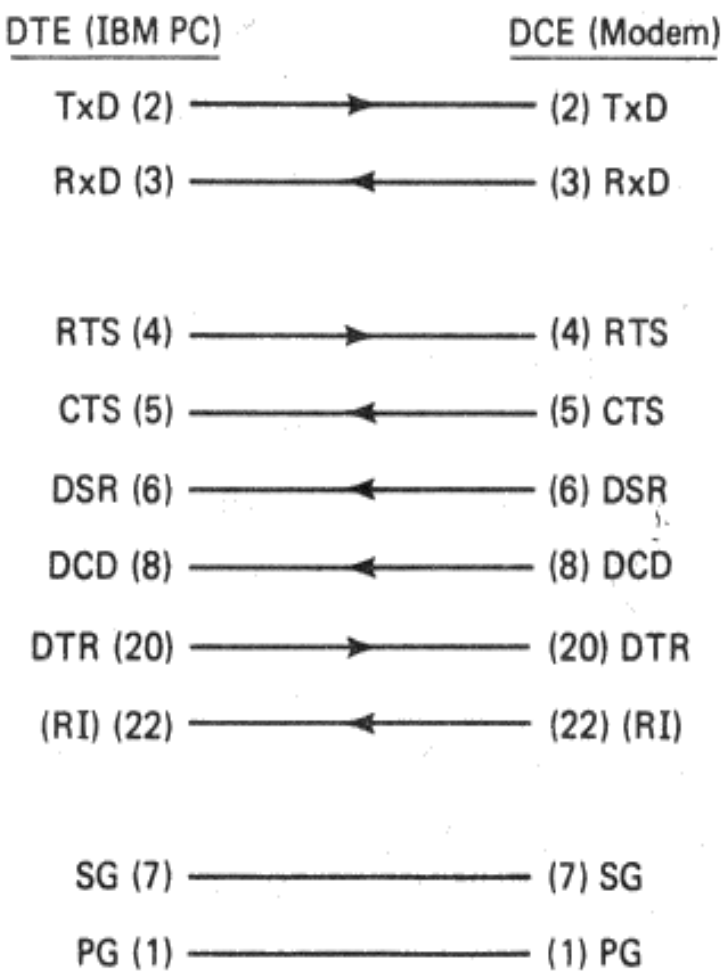
Make a serial connection between two PCs using COM1: ports. Write an application which allows to change text messages between 2 users (chat).

Data: COM1: 9600Baud, 2 STOP, No parity; Use INT 14h services.

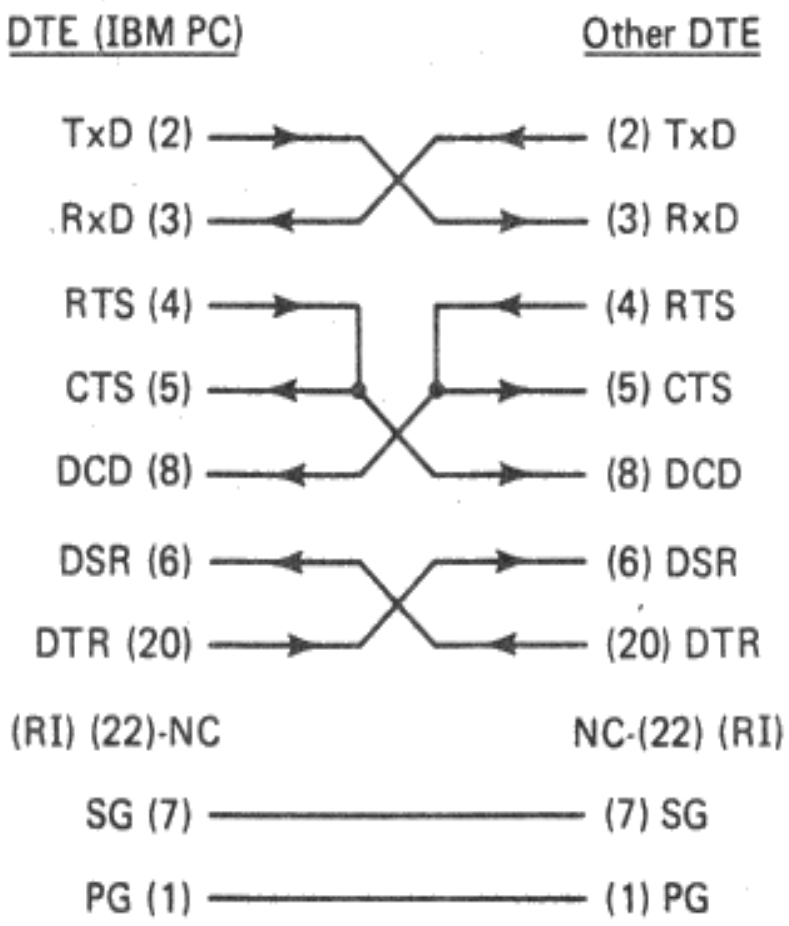
HW. Rewrite the application using the hardware interrupts.



Ideal Case



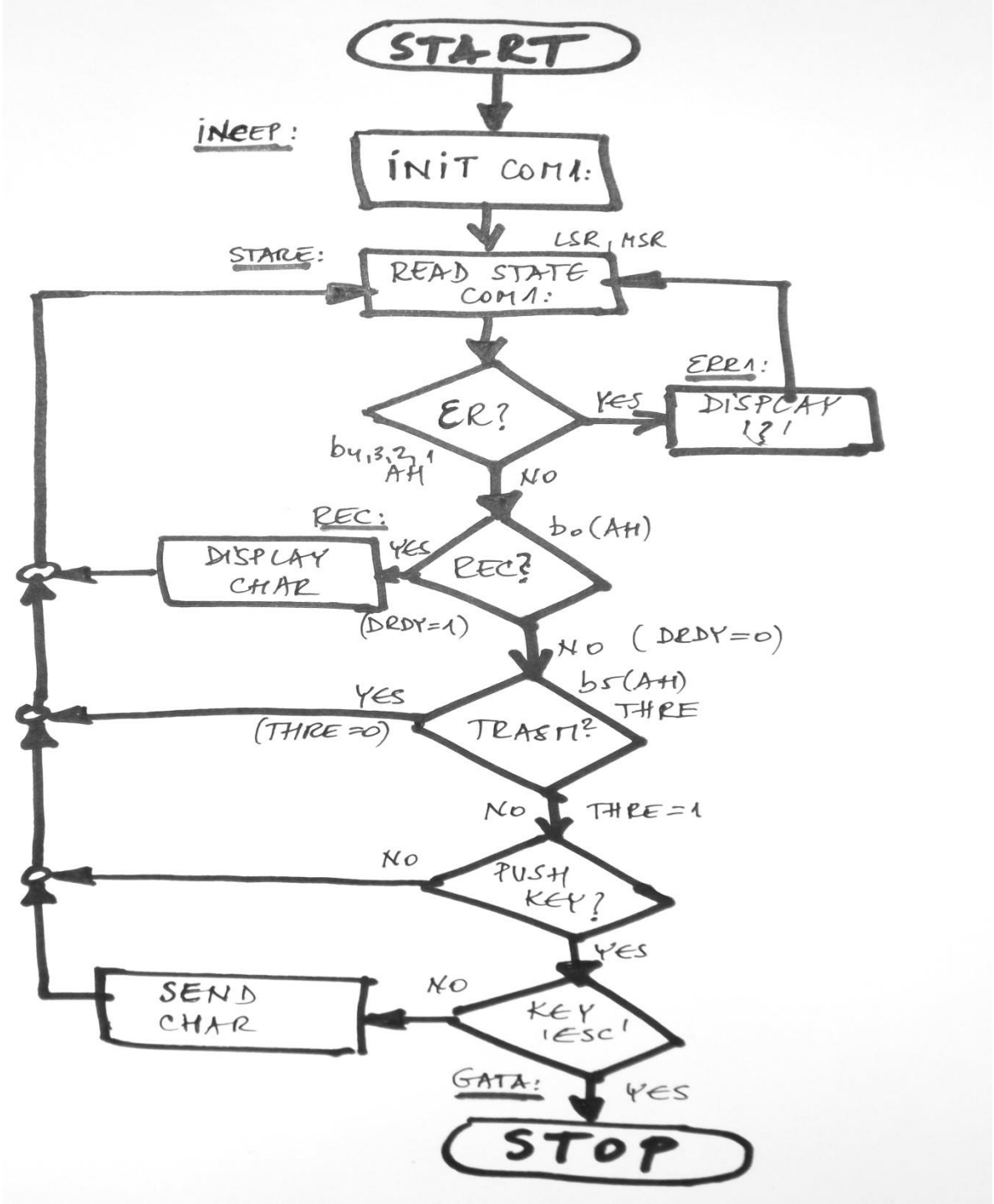
Null Modem Adapter



References:

http://www.beyondlogic.org/

http://www.hardwarebook.info/Category:Cable#Serial


START
INCEP:
INIT COM1:
STARE:
LSR, MSR
READ STATE COM1:
ER?
YES
ERR1:
DISPLAY '?'
b4,3,2,1 AH
NO
REC:
b0(AH)
REC?
YES
DISPLAY CHAR
(DRDY=1)
NO (DRDY=0)
b5(AH) THRE
YES
(THRE=0)
TRASM?
NO
THRE=1
NO
PUSH KEY?
YES
SEND CHAR
NO
KEY 'ESC'
GATA:
YES
STOP

```
title serial communication program
;using INT 14h service
code segment para 'code'
        assume cs:code,ds:code,ss:nothing
        org 100h

incep:  mov     ah,0        ;init. UART
        mov     dx,0        ;COM1
        mov     al,11100111b
;9600 baud,fara parit.,8 biti/car,2STOP
        int     14h
stare:  mov     ah,3        ;servici stare UART
        mov     dx,0        ;com1
        int     14h
        test    ah,1eh      ;test biti eroare
        jnz     err1
        test    ah,01h      ;test receptie
        jnz     receptie
        test    ah,20h      ;test buffer tr.
        jz      stare       ; THRE=1?
        mov     ah,1        ;astept tasta
        int     16h         ; z=0
        jz      stare
        mov     ah,0        ;citire tasta
        int     16h         ;al= car. tastat
        cmp     al,1bh      ;al= ESC ?
        jz      gata        ;transmit caracter
        mov     ah,1        ;emisie car. tastat
        mov     dx,0
        int     14h
        jmp     stare
receptie:
        mov     ah,2        ;serv. rec.caracter
        mov     dx,0
        int     14h
        push    ax          ;salvare caracter
        mov     bx,0        ;atrib. de afisare
        mov     ah,0eh
        int     10h         ;afisare
        pop     ax
        cmp     al,0dh      ;CR?
        jnz     stare
        mov     al,0ah      ;afisare LF
        mov     bx,0
        mov     ah,0eh
        int     10h
        jmp     stare
err1:   mov     ah,2
        mov     dx,0
        int     14h
        mov     al,'?'
        mov     bx,0
        mov     ah,0eh
        int     10h
        jmp     stare
gata:   int     20h
code    ends
        end     incep
```

HomeWork. 1. Find and study materials about the VIRTUAL COMMUNICATIONS PORT (VCP).

```
#include <bios.h>
#include <conio.h>
#define COM1      0
#define DATA_READY 0x100
#define SETTINGS ( 0x80 | 0x02 | 0x00 | 0x00)
int main(void)
{
  int in, out, status;
  bioscom(0, SETTINGS, COM1); /*initialize the port*/
  cprintf("Data sent to you:  ");
  while (1)
  {
    status = bioscom(3, 0, COM1); /*wait until get a data*/
    if (status & DATA_READY)
        if ((out = bioscom(2, 0, COM1) & 0x7F) != 0)  /*input a data*/
          putch(out);
        if (kbhit())
        {
          if ((in = getch()) == 27)   /* ASCII of Esc*/
            break;
          bioscom(1, in, COM1);   /*output a data*/
        }
  }
  return 0;
}
```

2. Analyze the above application.