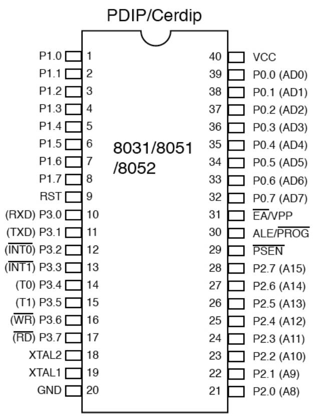
8051 Microcontroller: Serial Communication



Virendra Singh

Computer Architecture and Dependable Systems Lab

Department of Electrical Engineering

Indian Institute of Technology Bombay

http://www.ee.iitb.ac.in/~viren/

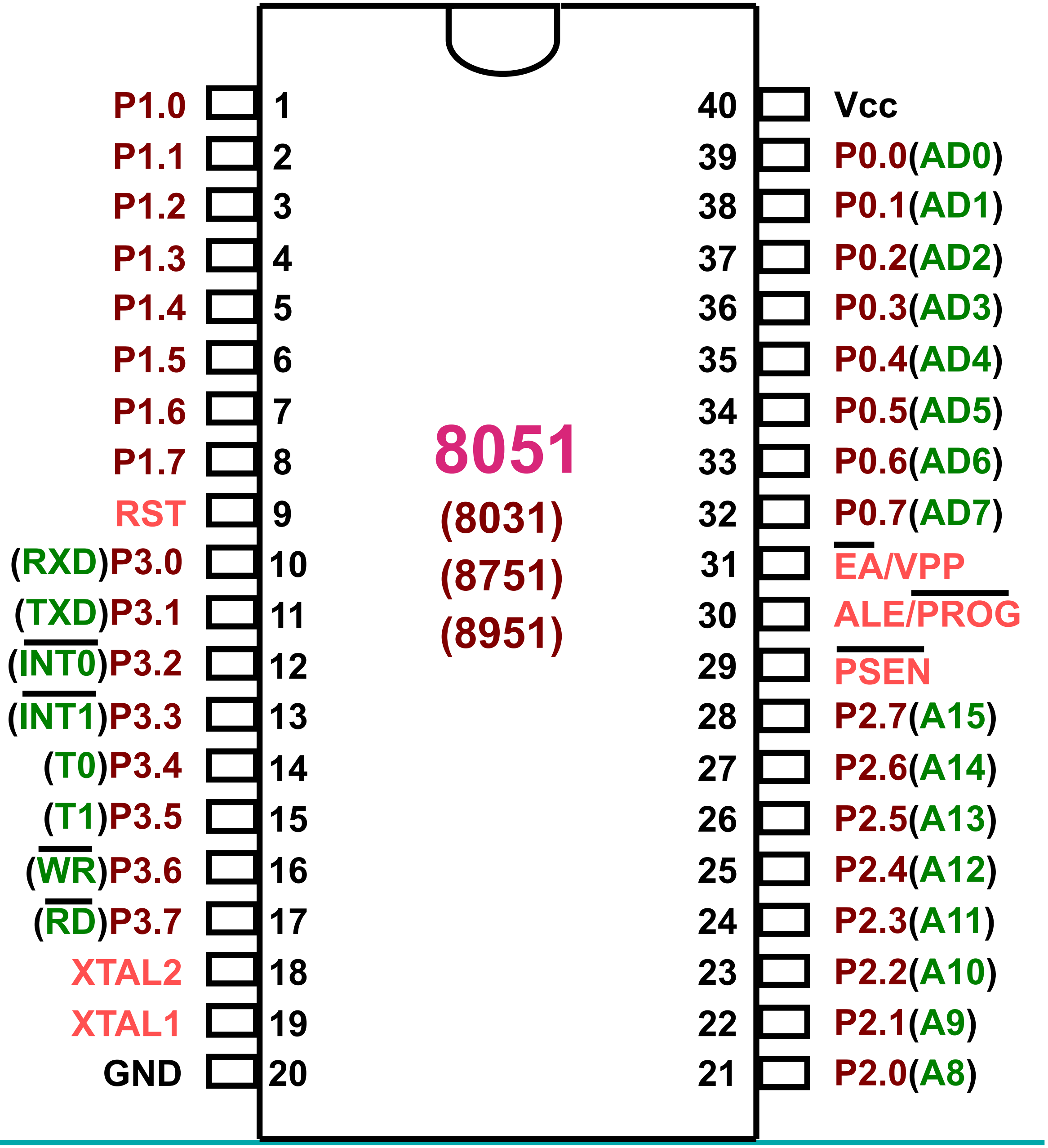
E-mail: viren@ee.iitb.ac.in

EE-309: Microprocessors

Lecture 15 (20 Aug 2015)

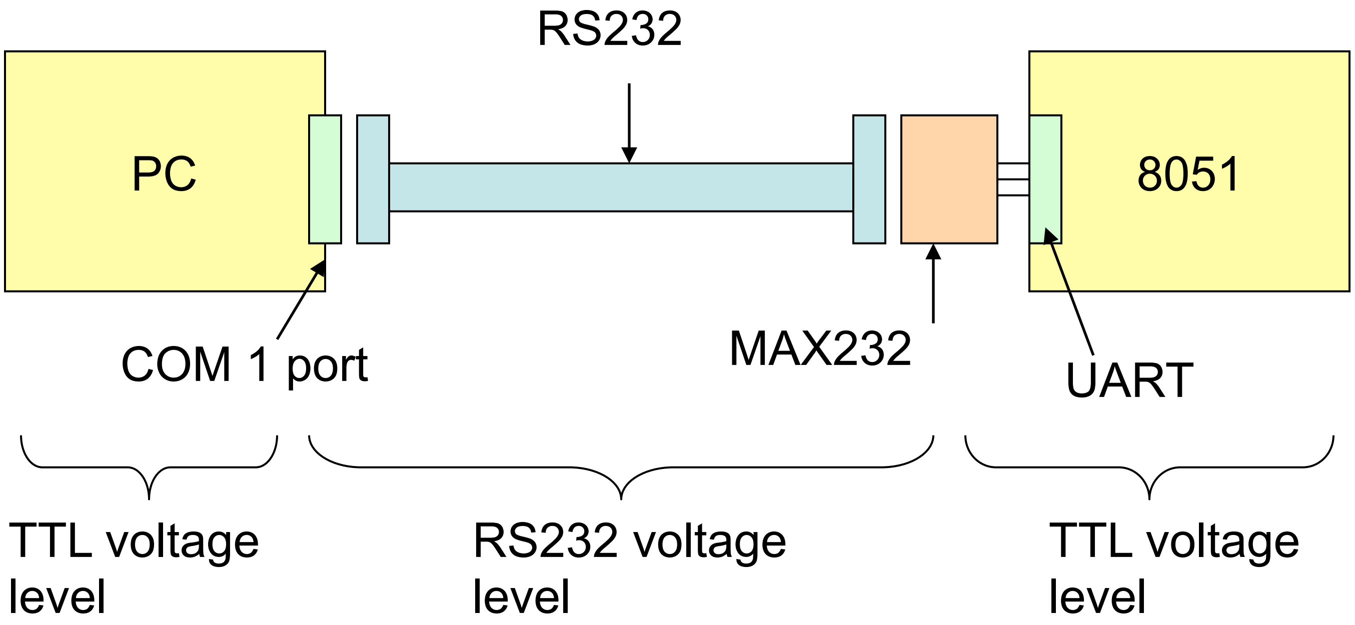
CADSL

8051 Pin Diagram

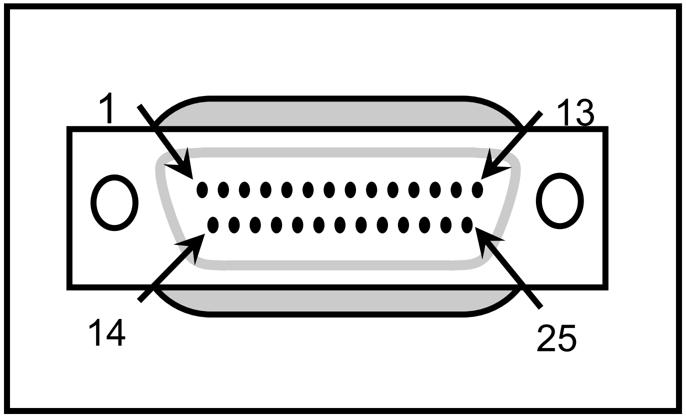


MAX232

- MAX232 IC chips are commonly referred to as line drivers.

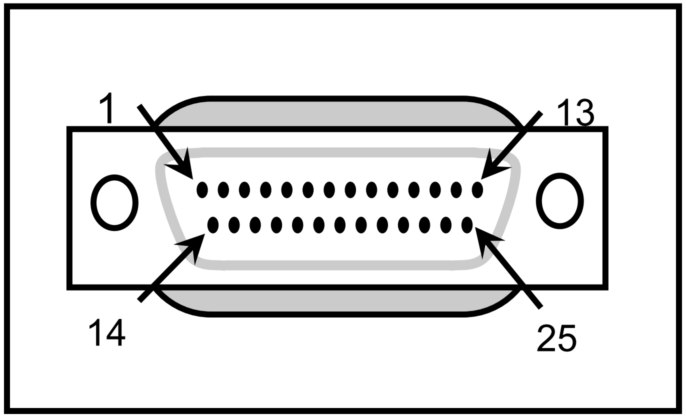


RS232 Pins for DTE



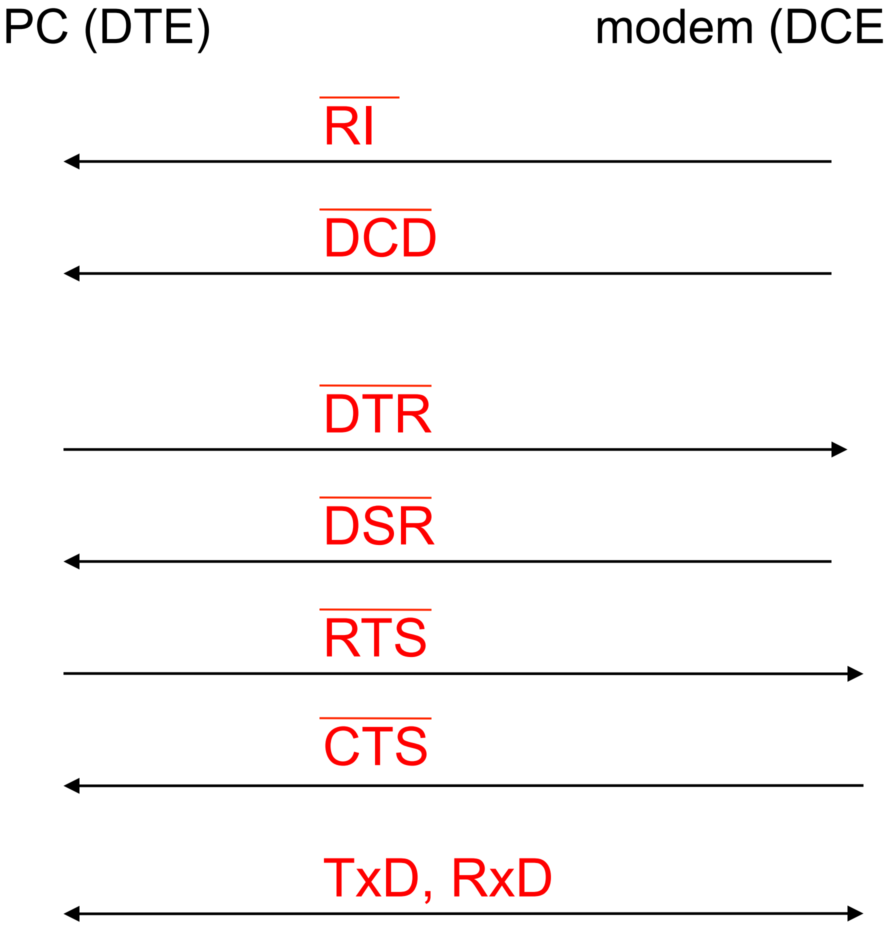
Pin	Description
1	Protective ground
2	Transmitted data (TxD)
3	Received data (RxD)
4	Request to send (RTS)
5	Clear to send (CTS)
6	Data set ready (DSR)
7	Signal ground (GND)
8	Data carrier detect (DCD)
9/10	Reserved for data testing
11	Unassigned
12	Secondary data carrier detect
13	Secondary clear to send

RS232 Pins for DTE



Pin	Description
14	Secondary transmitted data
15	Transmit signal element timing
16	Secondary received data
17	Receive signal element timing
18	Unassigned
19	Secondary request to sent
20	Data terminal ready (DTR)
21	Signal quality detector
22	Ring indicator
23	Data signal rate select
24	Transmit signal element timing
25	Unassigned

Communication Flow



PC (DTE)

modem (DCE)

$\overline{\text{RI}}$ Telephone is ringing

$\overline{\text{DCD}}$ Connection between two modems is set

$\overline{\text{DTR}}$ PC is ready

$\overline{\text{DSR}}$ modem is ready

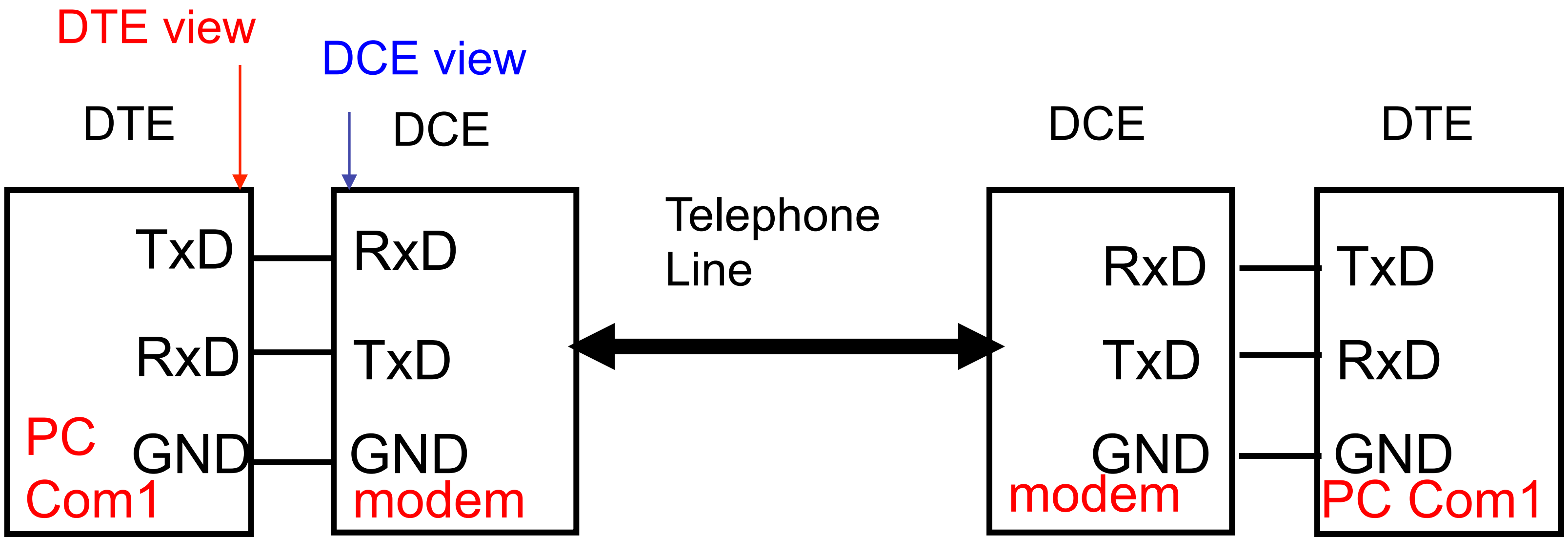
$\overline{\text{RTS}}$ PC wants to sent data

$\overline{\text{CTS}}$ modem is ready to receive

TxD, RxD transmit data

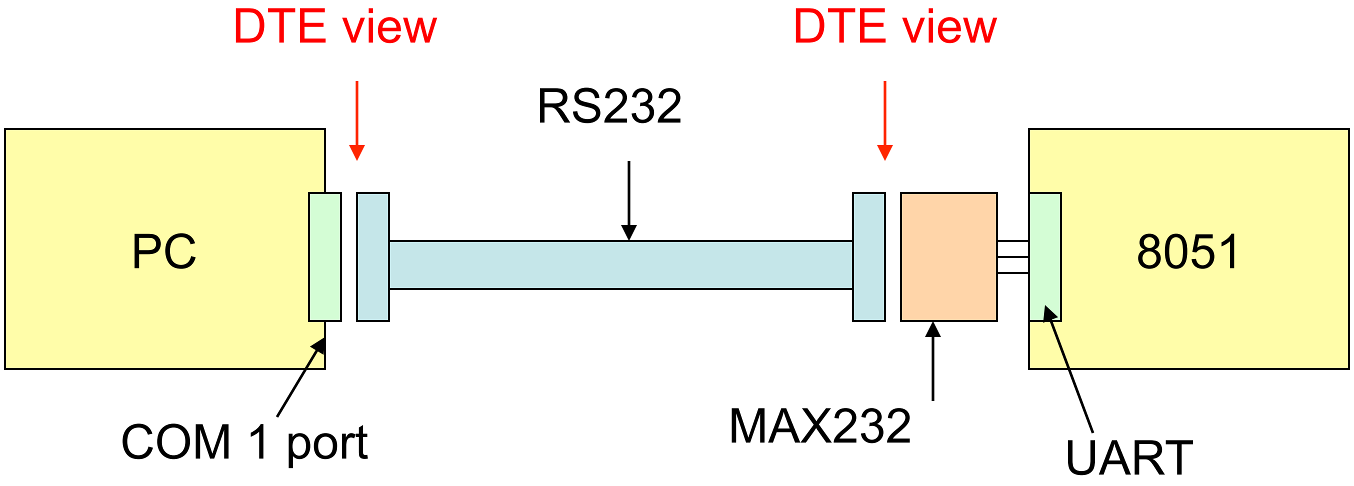
DTE and DCE

- Communication Equipments are classified as
  - DTE (data terminal equipment)
    - Terminals and computers that send and receive data
  - DCE (data communication equipment)
    - Communication equipment (only for transfer data), modem



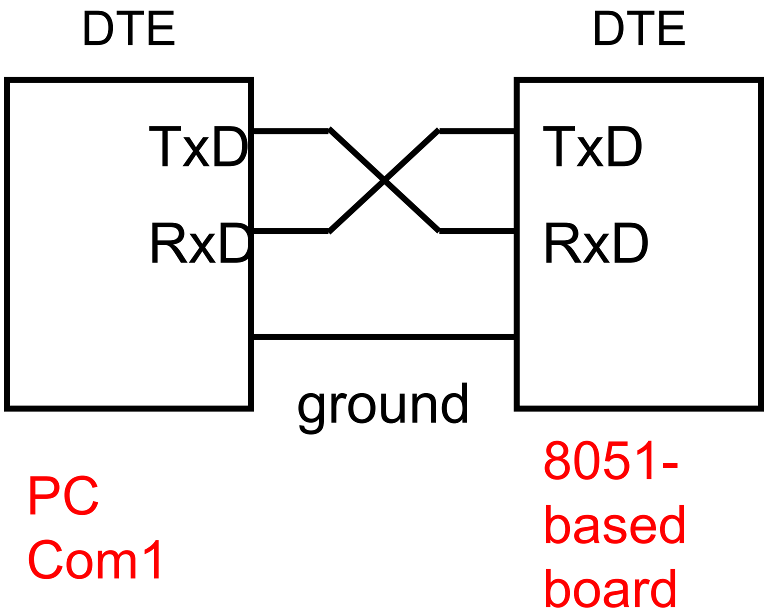
IBM PC/compatible COM ports

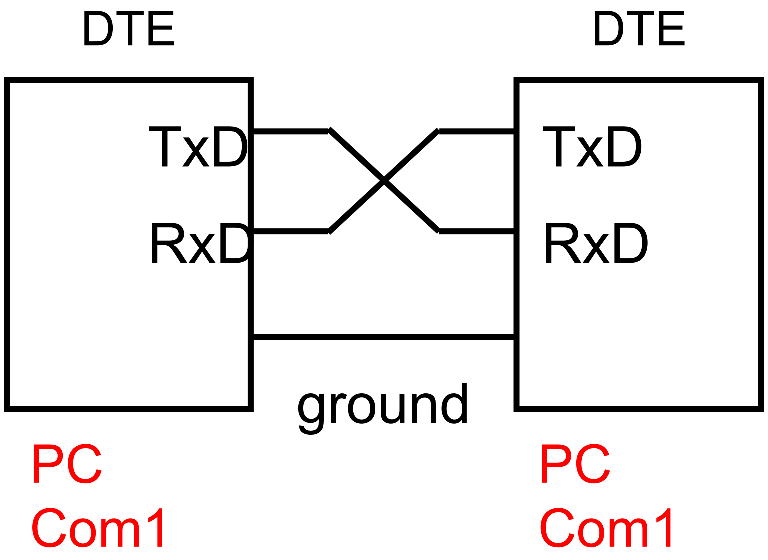
- IBM PC had 2 COM ports.
  - Both COM ports have RS232-type connectors.
  - For mouse, modem



Null Modem Connection

- The simplest connection between a PC and microcontroller requires a minimum of three pins, TxD, RxD, and GND.
  - null modem connection





RS422 & RS485

- By using RS232, the limit distance between two PCs is about 15m.
- It works well even the distance=30m.
- If you want to transfer data with long distance (ex: 300m), you can use RS422 or RS485.

TxD and RxD pins in the 8051

- In 8051, the data is received from or transmitted to
  - RxD: received data (Pin 10, P3.0)
  - TxD: transmitted data (Pin 11, P3.1)
- TxD and RxD of the 8051 are TTL compatible.
- The 8051 requires a line driver to make them RS232 compatible.
  - One such line driver is the MAX232 chip.

PC Baud Rates

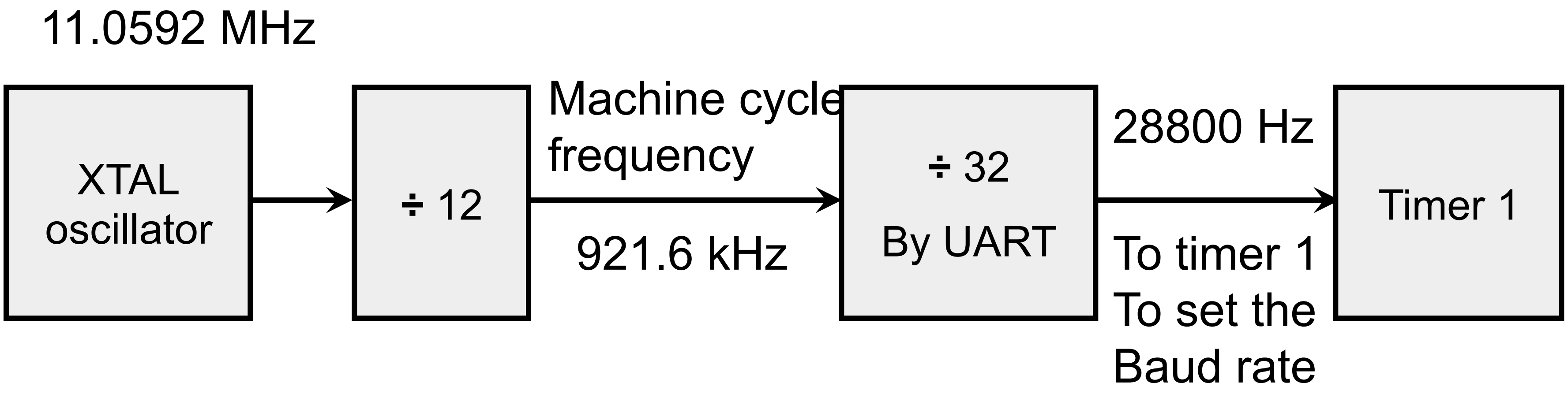
- PC supports several baud rates.
  - You can use netterm, terminal.exe, stty, ptty to send/receive data.
- Hyperterminal supports baud rates much higher than the ones list in the Table.

110 bps
150
300
600
1200
2400
4800
9600 (default)
19200

Note: Baud rates supported by 486/Pentium IBM PC BIOS.

Baud Rates in 8051

- The 8051 transfers and receives data serially at many different baud rates by using UART.
- UART divides the machine cycle frequency by 32 and sends it to Timer 1 to set the baud rate.
- Signal change for each roll over of timer 1



Baud Rates in 8051

- Timer 1, mode 2 (8-bit, auto-reload)
- Define TH1 to set the baud rate.
  - XTAL = 11.0592 MHz
  - The system frequency = 11.0592 MHz / 12 = 921.6 kHz
  - Timer 1 has 921.6 kHz/ 32 = 28,800 Hz as source.
  - TH1=FDH means that UART sends a bit every 3 timer source.
  - Baud rate = 28,800/3= 9,600 Hz

Example

With XTAL = 11.0592 MHz, find the TH1 value needed to have the following baud rates. (a) 9600 (b) 2400 (c) 1200

With XTAL = 11.0592 MHz, we have:

The frequency of system clock = 11.0592 MHz / 12 = 921.6 kHz

The frequency sent to timer 1 = 921.6 kHz/ 32 = 28,800 Hz

(a) 28,800 / 3 = 9600 where -3 = FD (hex) is loaded into TH1

(b) 28,800 / 12 = 2400 where -12 = F4 (hex) is loaded into TH1

(c) 28,800 / 24 = 1200 where -24 = E8 (hex) is loaded into TH1

Notice that dividing 1/12th of the crystal frequency by 32 is the default value upon activation of the 8051 RESET pin.

Timer 1 TH1 Register Values for Various Baud Rates

Baud Rate	TH1 (Decimal)	TH1 (Hex)
9600	-3	FD
4800	-6	FA
2400	-12	F4
1200	-24	E8

Note: XTAL = 11.0592 MHz.

Registers Used in Serial Transfer Circuit

- SUBF (Serial data buffer)
- SCON (Serial control register)
- PCON (Power control register)

SBUF Register

- Serial data register: **SBUF**

  MOV SBUF, #'A' ;put char 'A' to transmit

  MOV SBUF, A   ;send data from A

  MOV A, SUBF   ;copy received data to A

  – An 8-bit register

  – Set the usage mode for two timers

    - For a byte of data to be transferred via the TxD line, it must be placed in the SBUF.
    - SBUF holds the byte of data when it is received by the 8051;s RxD line.

  – Not bit-addressable

SCON Register

- Serial control register: **SCON**

SM0, SM1 Serial port mode specifier

REN (Receive enable) set/cleared by software to enable/disable reception.

TI Transmit interrupt flag.

RI Receive interrupt flag.

SM2 = **TB8** = **TB8** =0 (not widely used)

(MSB) (LSB)

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

* SCON is bit-addressable.

SCON Mode bits

- SM1 and SM0 determine the framing of data.
  - SCON.6 (SM1) and SCON.7 (SM0)

.

SM1	SM0	Mode	Operating Mode	Baud Rate
0	0	**0**	Shift register	Fosc./12
0	1	**1**	**8-bit UART**	**Variable by timer1**
1	0	**2**	9-bit UART	Fosc./64 or Fosc./32
1	1	**3**	9-bit UART	Variable

SM2

- SM2 enables the multiprocessor communication for mode 2 & 3.
- We make it 0 since we are not using the 8051 in a multiprocessor environment.
  - SM2=0 : Single processor environment
  - SM2=1 : multiprocessor environment

REN (Receive Enable)

- Set/cleared by software to enable/disable reception.
  - REN=1
    - It enable the 8051 to receive data on the RxD pin of the 8051.
    - If we want the 8051 to both transfer and receive data, REN must be set to 1.
    - **SETB SCON.4**
  - REN=0
    - The receiver is **disabled.**
    - The 8051 can not receive data.
    - **CLR SCON.4**

TB8 (Transfer Bit 8)

- SCON.3
- TB8 is used for serial modes 2 and 3.
- The 9th bit that will be transmitted in mode 2 & 3.
- Set/Cleared by software.

RB8 (Receive Bit 8)

- SCON.2
- In serial mode 1, RB8 gets a copy of the stop bit when an 8-bit data is received.

TI (Transmit Interrupt Flag)

- When the 8051 finishes the transfer of the 8-bit character, it raises the TI flag.
- TI is raised by hardware at the beginning of the stop bit in mode 1.
- Must be cleared by software.

RI (Receive Interrupt)

- Receive interrupt flag. Set by hardware in mode 1. Must be cleared by software.
- When the 8051 receives data serially via RxD, it gets rid of the start and stop bits and place the byte in the SBUF register.
- Then 8051 raises RI to indicate that a byte is received
- RI is raised at the beginning of the stop bit.

SCON Serial Port Control Register (Bit Addressable)

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

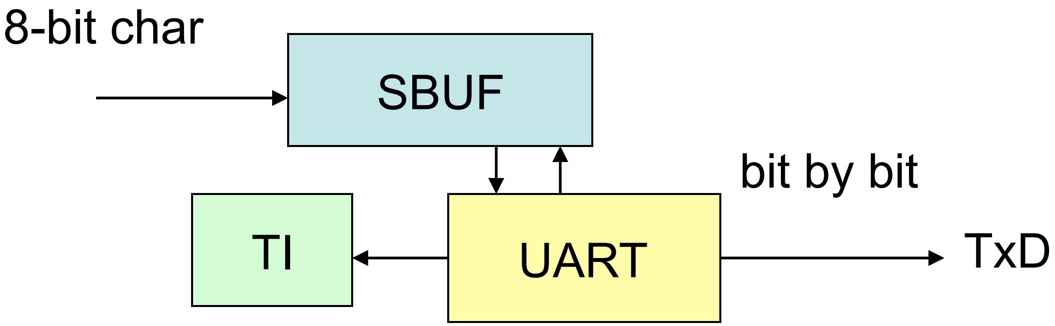
SM0	SCON.7	Serial port mode specifier
SM1	SCON.6	Serial port mode specifier
SM2	SCON.5	Used for multiprocessor communication. (Make it 0)
REN	SCON.4	Set/cleared by software to enable/disable reception.
TB8	SCON.3	Not widely used.
RB8	SCON.2	Not widely used.
TI	SCON.1	Transmit interrupt flag. Set by hardware at the beginning of the stop bit in mode 1. Must be cleared by software.
RI	SCON.0	Receive interrupt flag. Set by hardware halfway through the stop bit time in mode 1. Must be cleared by software.

Note: Make SM2, TB8, and RB8 = 0.

Transfer Data with TI Flag

- The following sequence is the steps that the 8051 goes through in transmitting a character via TxD:

1. The byte character to be transmitted is written into the SBUF register.
2. It transfers the **start bit**.
3. The **8-bit character** is transferred one bit at a time.
4. The **stop bit** is transferred.



Transfer Data with the TI flag

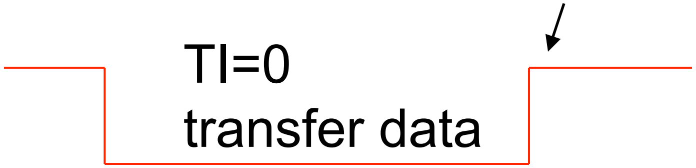
5. During the transfer of the stop bit, the 8051 raises the TI flag, indicating that the last character was transmitted and it is ready to transfer the next character.
6. By monitoring the **TI** flag, we know whether or not the 8051 is ready to transfer another byte.
   - We will not overloading the SBUF register.
   - If we write another byte into the SBUF before TI is raised, the untransmitted portion of the previous byte will be lost.
   - We can use interrupt to transfer data
7. After SBUF is loaded with a new byte, the TI flag bit must be cleared by the programmer.

Transferring Data Serially in 8051

1. Use the timer 1 in mode 2
   - `MOV TMOD,#20H`
2. Set the value TH1 to chose baud rate.
   - `MOV TH1,#FDH` **;Baud rate = 9600bps**
3. Set SCON register in mode 1.
   - `MOV SCON,#50H`
4. Start the timer.
   - `SETB TR1`

Transferring Data Serially in 8051

5. Clear TI flag.
   - `CLR TI`



raise when sending the stop bit

TI=0
transfer data

6. The character byte to be transferred serially is written into the SBUF register.
   - `MOV SBUF,#'A'`
7. Keep monitoring the Transmit Interrupt (TI) to see if it is raised.
   - `HERE:  JNB TI, HERE`
8. To transfer the next character, go to Step 5.

Example: Serial Transmission

Write a program for the 8051 to transfer letter “A” serially at 4800 baud, continuously.

```
        MOV   TMOD,#20H    ;timer 1, mode 2
        MOV   TH1,#-6      ;4800 baud rate
        MOV   SCON,#50H    ;8-bit,1 stop,REN enabled
        SETB  TR1          ;start timer 1
AGAIN:  MOV   SBUF,#”A”    ;letter “A” to be transferred
HERE:   JNB   TI,HERE      ;wait for the last bit
        CLR   TI           ;clear TI for next char
        SJMP  AGAIN        ;keep sending A
```

Thank You