8251 USART and RS232C

Presented By:

Diwakar Yagaysen

Asst. Prof. CSE BBDNITM, Lucknow

Objectives

- Upon completion of this chapter, you will be able to:
- List the advantages of serial communication over parallel communication
- Explain the difference between synchronous and asynchronous communication
- Define the terms simplex, half duplex, and full duplex and diagram their implementation in serial communication
- Describe how start and stop bits frame data for serial communication
- Compare the measures baud rate and bps (bits per second)
- Describe the RS232 standard
- Compare DTE (data terminal) versus DCE (data communication) equipment
- Describe the purpose of handshaking signals such as DTR, RTS, and CTS
- Describe the operation of a USART and use and 8251 IC

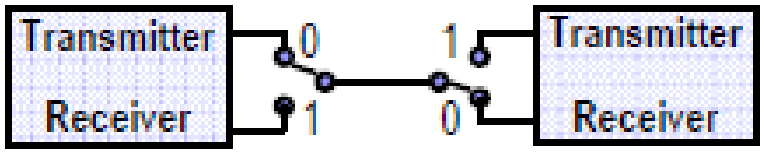
Basics of Serial Communication

- Microprocessors are based mostly on 8-bit registers. Thus, their fastest I/O is 8-bit parallel ports.
- But, wiring cost of a long distance communication is very expensive if you carry 8-wires.
- Remedy is to transfer data serially in bits instead of in bytes or words.

Simplex vs Duplex

- In Simplex Mode data flow is in one direction only.

- In Half Duplex Mode data flows in one direction, at a given time, A protocol and switches connect the devices both to receive and also to transmit.



- In Full Duplex mode, transmitted data and received data goes simultaneously through two channels.

Asynchronous vs Synchronous

How can the receiver get each data bit when it is delivered by the transmitter?

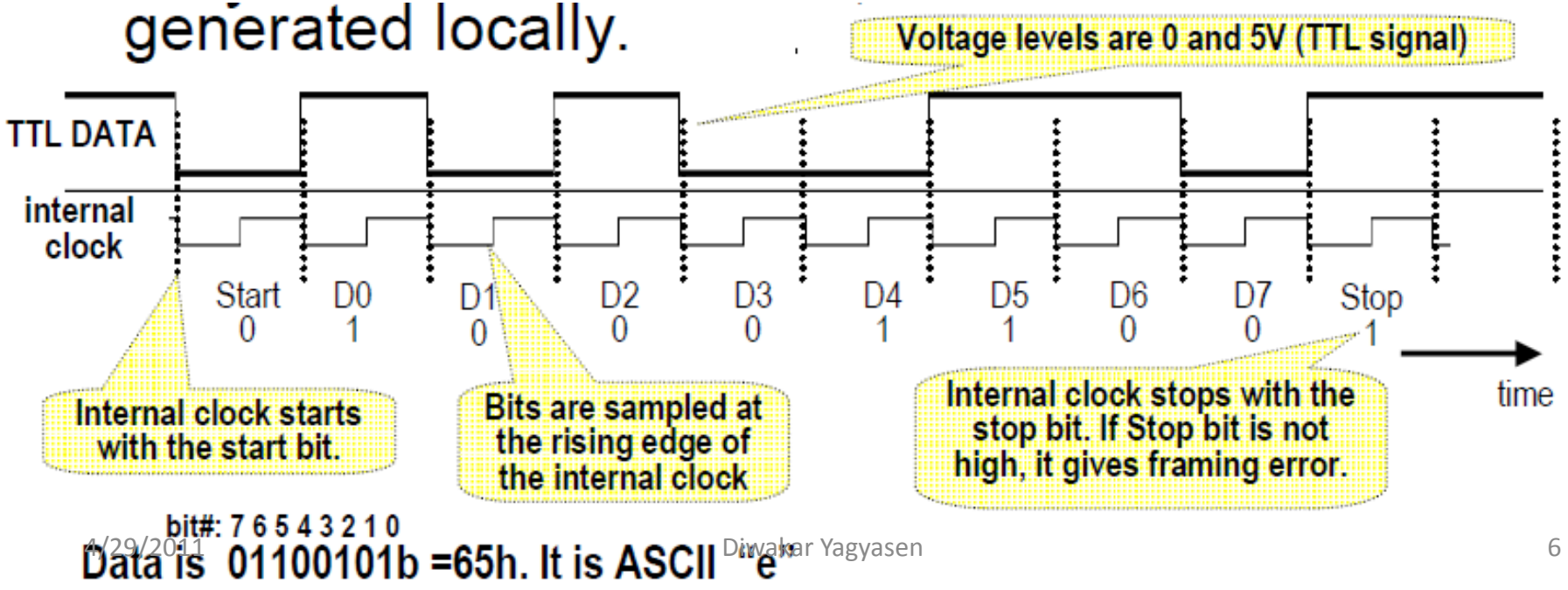
- In synchronous transmission an explicit clock signal describes the instants of valid data.
  - A single data bit is sent at each clock.
  - Minimum three signal lines required for full duplex,

    Receive-DATA, Transmit-DATA, and CLOCK.
- In asynchronous transmission clock is derived using a-priory parameters and a start bit.
  - Transfer rate known, internal clock starts to pulse with the start bit, at the known transfer rate.
  - Ony two signal lines required for full duplex

    Receive-DATA and Transmit-DATA

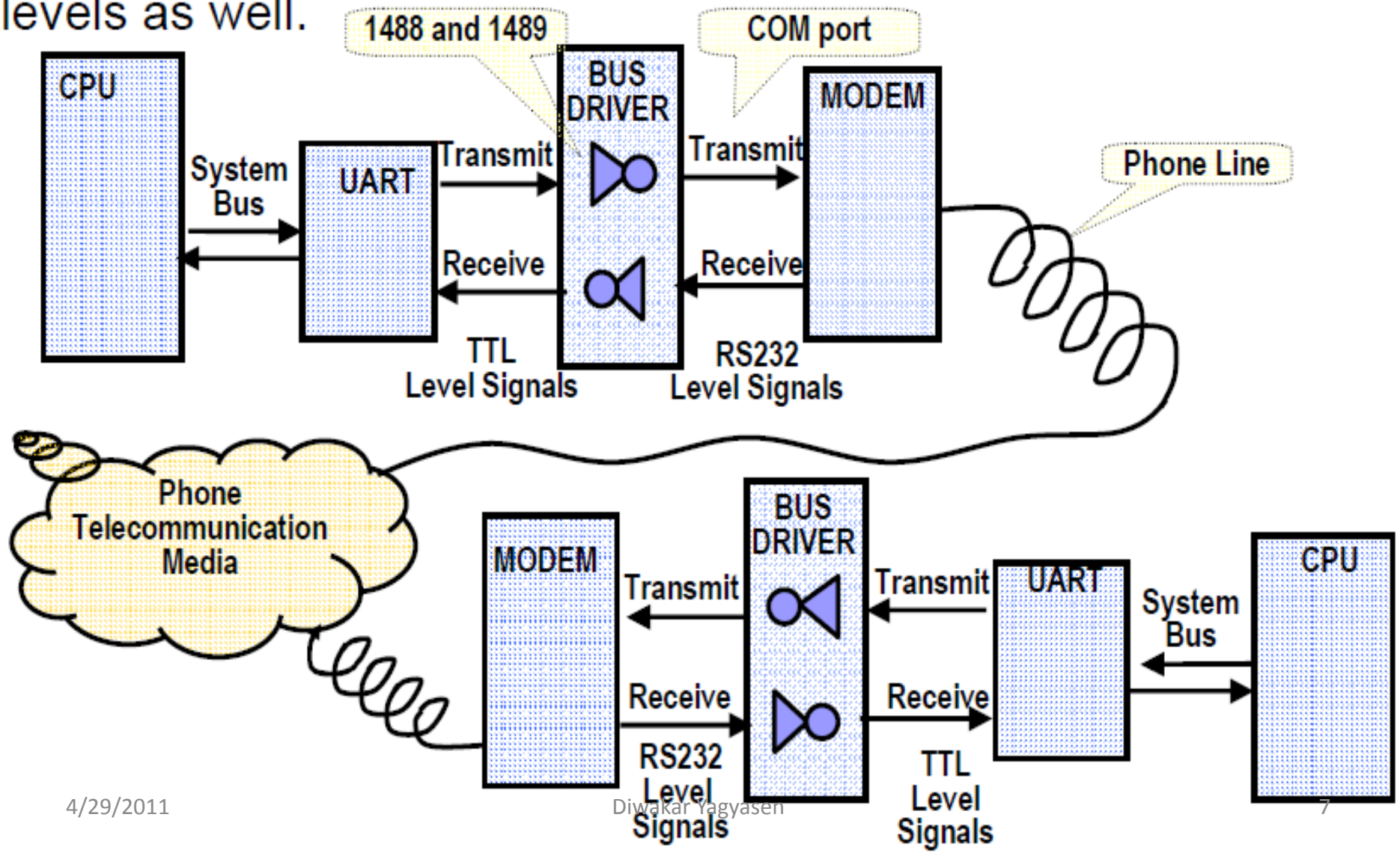
Asynchronous Transmission Start and Stop Bits

- Start bit is required to synchronize the internal clock of receiver.
- Stop bit is required to test the clock frequency.
- Only DATA is transmitted, internal clock is generated locally.



Asynchronous Communication System

Computers may be connected to eachother at COM port or TTL signal levels as well.



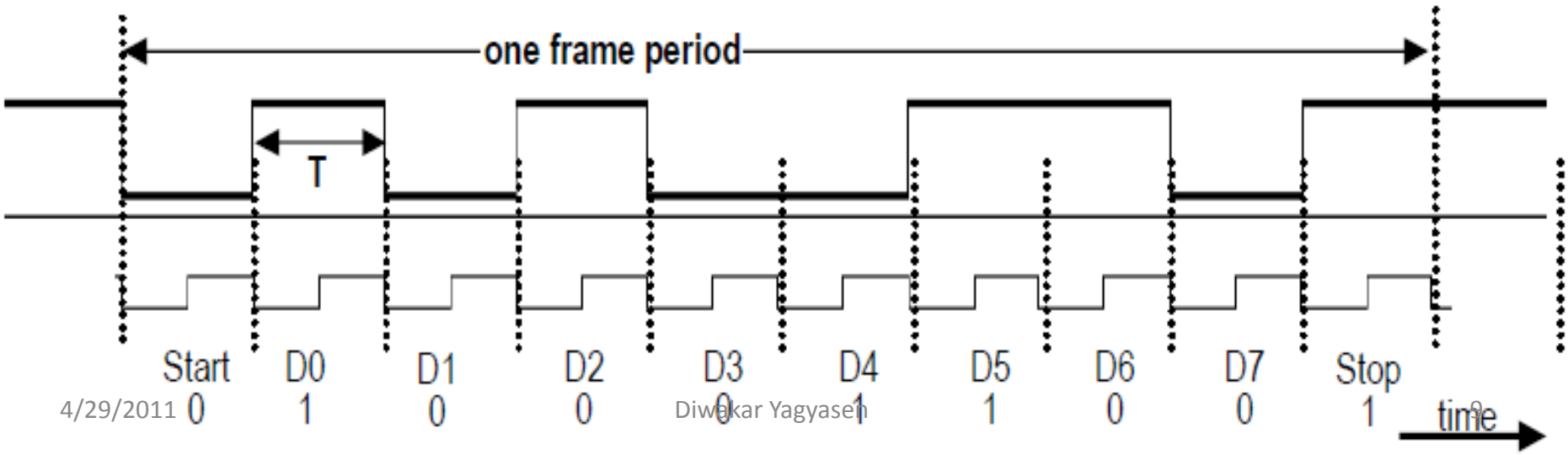
Data Transfer Rate

- Both devices shall know the data transfer rate of the communication to synchronize the internal clocks correctly.
- Data transfer rate is measured in BAUD
  - Baud = bit/second

  (including start, stop, data, parity etc.)
  - kilo Baud = 1000 Baud. (not 1024 Baud)
  - Mega Baud = 1000 000 Baud

Baud Rate Calculations

- If each bit takes T seconds, the baud rate is B= 1/T.
  - Standard Baudrates are 150, 300, 600, 1200, 2400, 4800, 9600, 19200, etc.
  - Baudrate tolerance for a10-bit frame is 5%.
- Example: T=209μs ➔ B= 1/T = 4785 Baud
  - it is 4800 Baud within 5% tolerance.

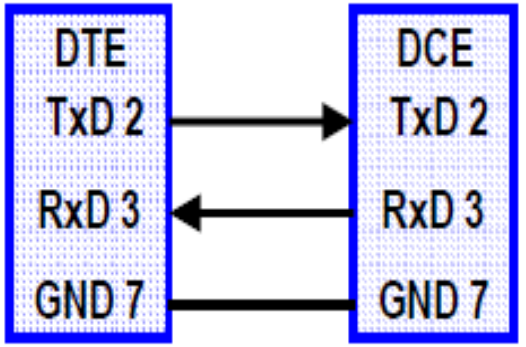


Packet transmission time

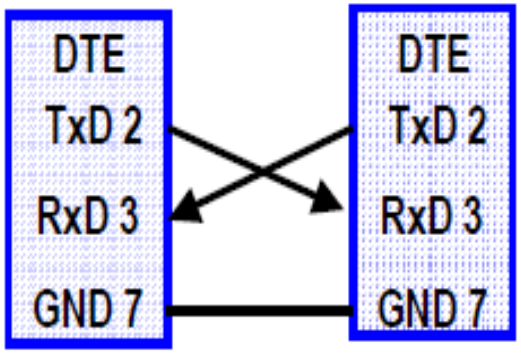
- If a system sends a packet of 50 bytes at 1200 Baud, using 8-data, no-parity, one stop bits, what is the transmission time of the whole packet:
  - 1-byte frame is 1-start + 8-data + 1-stop bit = 10 bits/byte.
  - packet is transferred by 50 x 10 bits = 500 bits on the serial communication line.
  - T_{packet} = 500 bits / 1200 Baud = 0.417sec. = 417 milliseconds.

RS232 Socket Pins

- Recommended standard describe two kind of sockets.
  - DTE: Device Terminal Equipment.
    - Computers, Terminals, etc.
  - DCE: Device Communication Equipment
    - Modem (modulator-demodulator).



- **Equipment connections:**
  - DTE is connected to DCE without crossing.
  - DTE is connected to DTE cross-wired.



Handshaking Signals

Modem-Terminal handshaking signals:

- Device status signals
  - DTR: Data terminal ready (DTE is ok.).
  - DSR: Data Set ready (DCE is ok.)
- Flow control signals
  - RTS: Request to Send (DTE sends char.)
  - CTS: Clear to Send (DCE accepts RTS)

UART 8251

- A processor may transmit/receive data in serial format without any extra hardware.
  - But it costs to the processing time of the processor.
- A UART (Universal Asynchronous Receiver Transmitter) is a hardware device that shifts out data bits to transmit a data byte, and also shifts-in data bits to receive a data byte.

8251 UART Device

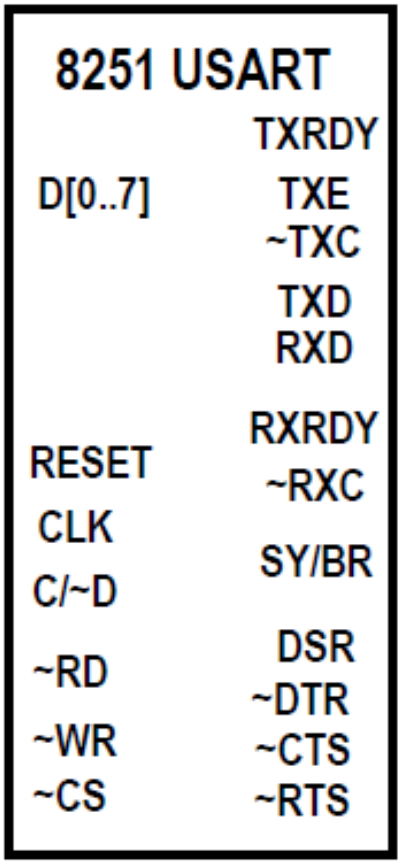
Register addressing

~CS C/~D=00 (data)

- writes to the transmit buffer
- reads from the receive buffer

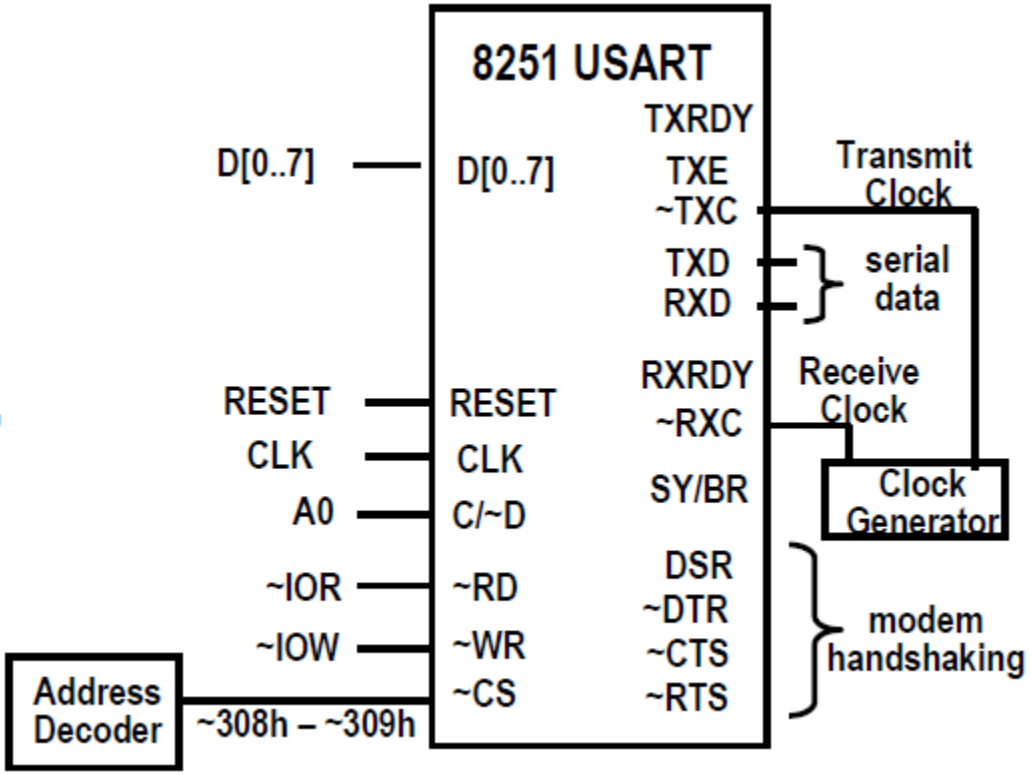
~CS C/~D=01 (control)

- writes to the
  - mode register right after a reset.
  - command register after mode is written.
- reads from the status register.



8251 system bus connection

- Data buffer address is 308h
- Control/Status address is 309h,
- TXRDY and RXRDY are for interrupted operation.



8251 Clock Signals

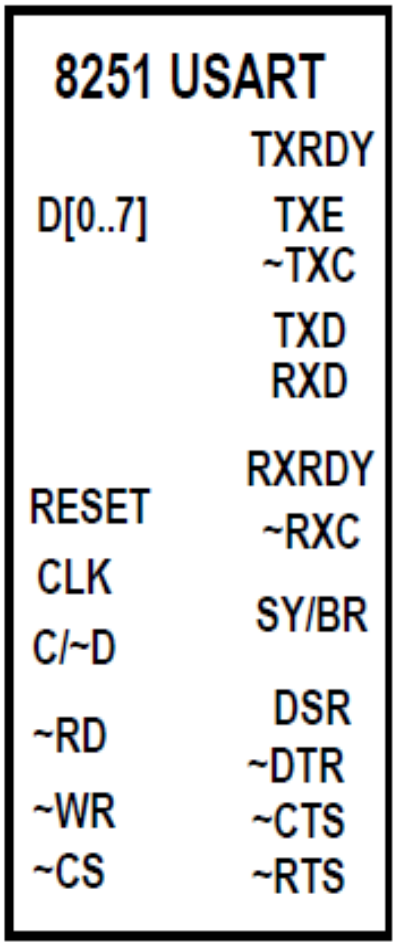
- CLK is system clock input
- TXC and RXC are transmit-receive clock inputs.
  - There are three baudrate factors
    - divide by 1, 16 and 64.

Example. Find RXC oscillator frequency for 1200 Baud operation with baudrate factor 1/64.

Solution: f_{RXC} =1200*64 Hz = 76.8 kHz.

Example: What shall be the baudrate factor for 4800 Baud operation if RXC is connected to 19.2 kHz ?

Solution: 19.2kHz/4.8kHz = 4, → 1/4



8251 Reset Sequence

- 8251 reset sequence is
  - **write three successive zeros** to control address to assure writing a reset to the command register.
  - **write command 40h** to reset (reset chip)
- After the reset, 8251 expects mode settings.
  - write the mode settings to control address
- There after 8251 needs command settings.
  - write command for command settings.
- Now the device is ready for transmit and receive operations

8251 mode/control settings and status bits

8251 USART

Mode Register format for asynchronous mode:

b7 b6 = { S2S1: nr.of stop bits **00:** invalid / **01:**1stop / **10:** 1.5stop / **11:** 2stop },
b5 = { EP: parity type **0:** odd / **1:** even },
b4 = { PEN: parity enable **0:** no-parity-bits / **1:** parity-bits-present },
b3 b2 = { L2L1: nr.of data bits **00:** 5-bit / **01:** 6-bit / **10:** 7-bit / **11:** 8-bit },
b1 b0 = { B2B1: baud rate factor **00:** sync-mode / **01:** /1 / **10:** /16 / **11:** /64 }

Control Register format for asynchronous mode:

b7 = { EH: Enter hunt mode to search sync char **1:** enable / **0:** disable }
b6 = { IR: Internal reset **1:** resets the 8251A }
b5 = { RTS: Request to send, **1:** RTS-output-forced-to-low }
b4 = { ER: Error Reset **1:** reset error flags PE,OE,FE }
b3 = { SBRK: Send break char **1:** forces TxD low }
b2 = { RxE: Receiver enable **1:** enable, **0:** disable }
b1 = { DTR: Data terminal ready **1:** DTR-output-forced-to-low }
b0 = { TxE: Transmitter enable **1:** enable, **0:** disable }

Status Register format for asynchronous mode:

b7 = {DSR **1:** DSR pin is active (low)}
b6 = {SY/BD **1:** sync-or-break char detected}
b5 = {FE **1:** Framing error occurs}
b4 = {OE **1:** Overrun error occurs}
b3 = {PE **1:** Parity error occurs}
b2 = {TxE **1:** Tx finished transmitting all data}
b1 = {RxRDY **1:** Data-in buffer is full}
b0 = {TxRDY **1:** Data-out buffer is empty}

8251 initialization

- For the circuit with ports 308h, 309h enabling 8251, write a code
- a) to initialize it for 1200 baud when TXC connected to 19.2kHz, to transmit characters to a DCE with 8-bit data, no parity, one stop bit configuration.
- b) to read port 300h, and transmit the value in serial format

Mode Register

b7 b6 = { S2S1: nr.of stop bits 00: inval./ 01:1stop / 10: 1.5stop / 11: 2stop },
b5 = { EP: parity type 0: odd / 1: even },
b4 = { PEN: parity enable 0: no-parity-bits / 1: parity-bits-present },
b3 b2 = { L2L1: nr.of data bits 00: 5-bit / 01: 6-bit / 10: 7-bit / 11: 8-bit },
b1 b0 = { B2B1: baud rate factor 00: sync-mode / 01: 1 / 10: 16 / 11: 64 }

Control Register

b7 = { EH: Enter hunt mode to search sync char 1: enable / 0: disable }
b6 = { IR: Internal reset 1: resets the 8251A }
b5 = { RTS: Request to send, 1: RTS-output-forced-to-low }
b4 = { ER: Error Reset 1: reset error flags PE,OE,FE }
b3 = { SBRK: Send break char 1: forces TxD low }
b2 = { RxE: Receiver enable 1: enable, 0: disable }
b1 = { DTR: Data terminal ready 1: DTR-output-forced-to-low }
b0 = { TxE: Transmitter enable 1: enable, 0: disable }

Mode register shall be 01001110b = 4Eh
Control register shall be 00110011b = 33h

handshaking to modem is enabled.

receiver is disabled. only transmitter enabled.

8251 coding

- For the circuit with ports 308h, 309h enabling 8251, write a code
- a) to initialize it for 1200 baud when TXC connected to 19.2kHz, to transmit characters to a DCE with 8-bit data, no parity, one stop bit configuration.
- b) to read port 300h, and transmit the value in serial format

Status Register format for asynchronous mode:

b7	= {DSR	**1**: DSR pin is active (low)}
b6	= {SY/BD	**1**: sync-or-break char detected}
b5	= {FE	**1**: Framing error occurs}
b4	= {OE	**1**: Overrun error occurs}
b3	= {PE	**1**: Parity error occurs}
b2	= {TxE	**1**: Tx finished transmitting all data}
b1	= {RxRDY	**1**: Data-in buffer is full}
b0	= {TxRDY	**1**: Data-out buffer is empty}

```
        .model small
        .code
; reset 8251
        mov dx,309h
        mov al,0
        out dx,al
        out dx,al
        out dx,al
        mov al,40h
        out dx,al
; set mode
        mov al, 4Eh
        out dx,al
; set  command
        mov al, 33h
        out dx,al
```

```
mainloop
        ....
; read the port
        mov dx,300h
        in al,dx
        mov ah,al ; save it
; send the char,
; wait if buffer is full
        mov dh,309h
wdataout
        in al,dx ; status byte
        and al,01h
        jz wdataout
; now send character
        mov al,ah
        mov dx,308h
        out  dx,al
; continue looping
        jmp mainloop
        end
```

Questions

Thank You.