Microprocessors 1

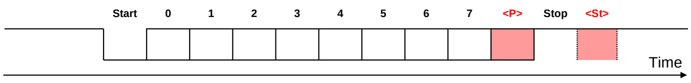
MCS-51 Serial Port

Introduction to Serial Communications

- Serial vs. Parallel transfer of data
- Simplex, Duplex and half-Duplex modes
- Synchronous, Asynchronous
  - UART – Universal Asynchronous Receiver/Transmitter.
    - Handles all issues related to the asynchronous transmission of byte sized data.
  - USART – Universal Synchronous/Asynchronous Receiver/Transmitter.
- Data transfer rate
  - Bps, baud.

Framing

- An 8-bit message needs to be “framed” so that the receiver can detect correctly its beginning and end.
- Standard framing:
  - Start bit – always 0, Stop bit – always 1.
  - Optional parity bit
  - Stop bit can be one or two bits
  - The message now becomes:
    - Start bit (1→0), LSB, …, MSB, <parity bit>, Stop bit (0→1), <2nd stop bit (1)>



RS-232 Protocol

- Serial communication standard for small computing systems.
  - Original intent was for communication between small computer. Mostly used for communication with slow peripherals.
    - The cable connecting the PC to the Kit in the lab follows this standard.
- Defines many signals – about 25 – however, only three are used in most cases.
  - RxD – Received Data
  - TxD – Transmitted Data
  - GND – Common Ground

RS-232 Line Driver

- RS-232 requires non-TTL compatible voltage levels
  - -3 to –25 for 1 and +3 to +25 for 0
- Therefore, we need an interface to connect to standard TTL chips.
  - MAX 232 and MAX 233 chips.
    - Accept standard TTL logic levels and produce RS-232 levels.
    - Utilize a normal +5 V supply.

MCS-51 Serial Port

- MCS-51 has a full-duplex serial port that can be used as a normal serial interface (non-framed) or as an internal UART (framed).
  - This serial port controls the RxD and TxD dual functions for pins P3.0 and P3.1.
- The MCS-51 serial port is controlled using the SCON SFR (98H).
- The MCS-51 serial port communicates with the rest of the chip using the SBUF SFR (99H).

The SBUF Register

- SBUF is actually two separate registers at the same address.
  - Write-only transmit register.
  - Read-only receive register.
    - Cannot read back what was sent for transmission.
- The byte to be transmitted on the serial port is “written” into SBUF.
  - Serial transmission starts immediately.
- The byte received from the serial port will be stored in SBUF once the last bit is received.
  - This is called “double buffering”.
    - Received data is buffered in the serial port itself until the full byte is received. This allows a little more time to deal with the previous data before its over-written with the new one.
    - HOWEVER, the previous data must be read before the new byte completes. Otherwise, the old data will be lost.

The SCON Register

MSB							LSB
SM0	SM1	SM2	REN	TB8	RB8	TI	RI

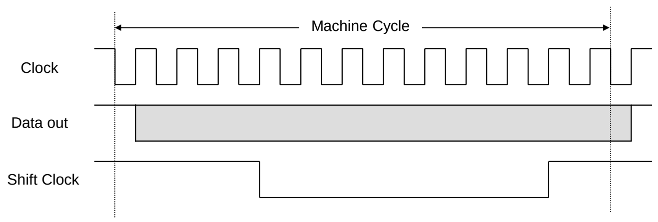
Bit	Name	Description
SCON.7	SM0	Serial Port Mode bit 0
SCON.6	SM1	Serial Port Mode bit 1
SCON.5	SM2	Multiprocessor Communication Enable
SCON.4	REN	Receive Enable Set to enable reception. CLR to disable reception.
SCON.3	TB8	Bit 8 of message to transmit Used to transmit optional parity bit
SCON.2	RB8	Bit 8 of received message Receives optional parity bit
SCON.1	TI	Transmit Interrupt Flag Set when Byte in SBUF is completely transmitted.
SCON.0	RI	Receive Interrupt Flag Set when a valid byte is received into SBUF

Modes of the Serial Port

- Mode 0 – SM0 = SM1 = 0
  - Half Duplex Synchronous Operation.
    - Data is sent and received (not simultaneously) using the RxD pin.
    - The TxD pin carries "the shift clock" during both receiving and transmitting. (Reference clock for synchronization).
    - Data is sent in 8-bit un-framed packets.
      - LSB first.
    - Data rate is set to 1/12 clock frequency.
      - Machine Cycle Frequency.
      - Same as the shift clock.

Mode 0 – Transmission

- Transmission starts as soon as byte is written into SBUF.
- During transmission, each bit stays valid on the RxD pin for one complete machine cycle.
  - The shift clock goes low in the middle of the cycle and returns high right before the end.
- The TI flag is set when the 8^{th} bit is done transmitting.



Mode 0 – Reception

- Reception is initiated as soon as REN bit is set to 1 and the receive interrupt (RI) bit is cleared.
  - Usually, REN is set at the beginning of the program to initialize the serial port, then RI is cleared to start a data input operation.
- As soon as RI is cleared, the shift clock will be produced on the TxD pin.
  - At the beginning of the following machine cycle, data will be clocked in from the RxD line.
    - The clocking occurs on the rising edge of the TxD line.
  - After the 8^{th} clocking cycle, the data is copied to SBUF and the RI bit is set.

Mode 1

- In mode 1, the 8051 serial port operates an 8-bit UART with variable baud rate.
  - The essential operation of a UART is parallel-to-serial conversion of output data and serial-to-parallel conversion of input data.
- 10 bits are transmitted on TxD or received on RxD.
  - Start bit, 8 data bits, 1 stop bit.
- The baud rate is set by the Timer 1 overflow rate.

Mode 1 Transmission

- Transmission starts when anything is written into SBUF.
- The period for each bit is the reciprocal of the baud rate.
- The transmit interrupt (TI) flag is set as soon as the stop bit appears on TxD.

Mode 1 Reception

- Reception is initiated by a 1-to-0 transition on the RxD line (assuming REN is 1).
  - A divide-by-16 counter is immediately started. The next bit is expected to arrive at the roll-over of this counter.
  - Bits are sampled at the 8^{th} count of this counter.
- "False Start Bit Detection"
  - 8 counts after the 1-to-0 transition, the RxD line is samples again. If it is not 0, then we have a false start bit. The receiver is reset and waits for the next 1-to-0 transition.

Mode 1 Reception

- Assuming that a valid start bit was detected:
  - The start bit is skipped.
  - Eight data bits are clocked into the serial port's register (NOT SBUF).
  - When all eight bits are received:
    - The ninth bit (the stop bit) is clocked into RB8
    - SBUF is loaded with the right data bits
    - The receiver interrupt flag (RI) is set.
  - The above three steps only occur if RI was 0 to start with.
    - Do not overwrite the previous data if it has not been read.

Mode 2

- The serial port operates as a 9-bit UART with a fixed baud rate.
- 11 bits are transmitted:
  - The start bit
  - The 8 data bits from SBUF
  - A 9^{th} data bit from TB8
  - The stop bit
- On reception, the 9^{th} data bit will be placed in RB8.
- The baud rate is fixed at either 1/32 or 1/64 of the oscillator frequency.

Mode 3

- 9-Bit UART with Variable Baud Rate.
  - Combination of modes 1 and 2.

The Baud Rates

- In Mode 0, the baud rate is fixed at the clock frequency divided by 12.
- By default, the baud rate in mode 2 is set to 1/64 of the clock frequency.
  - However, bit 7 of the PCON (Power Control) Register – known as SMOD – doubles the baud rate if it is set to 1.
  - So, if SMOD = 1, the baud rate for mode 2 is 1/32 of the clock frequency.

The Baud Rates (Contd.)

- In modes 1 and 3, the baud rate is set by the overflow rate of Timer 1.
  - However, that rate is too high. So, it is divided by 32 (or 16 if SMOD = 1) to generate the real baud rate.
  - You can think about as if Timer 1 will be clocked at XTAL / (12 * 32) = 28,800Hz (when SMOD = 0) or XTAL / (12 * 16) = 57,600Hz (when SMOD = 1)
- Transm=k x freq/(32*12*[256-TH1])
  - TH1=256-((k*freq)/(32*12*transmirate))

Setting Timer 1 to Generate Baud Rate

- How do we produce a baud rate of 1200 using an 8051 with a clock frequency of 12 MHz?
  - Baud Rate = K* FREC/(32*12*[256-TH1])
  - To produce 1200 baud, we need to set timer 1 to count for: 23 counts.
    - Set Timer 1 to operate in mode 2 (auto-reload) and set TH1 to 0E6H (-23).

Steps to Transmit a Byte

1. Program T1 for Mode2 (TMOD ← 0x20)
2. Load TH1 with the initial value (baud rate dependant) (TH1 ← FD / FA / F4 / E8)
3. Program SCON for Mode1 (SCON ← 0x50)
4. Start Timer1 (SETB TR1)
5. Clear TI
6. Load SBUF with the byte to be transferred (SBUF ← byte)
7. Wait until TI becomes 1 (JNB TI, not_done)
8. Go back to Step5 for next byte

Examples: Transmit a character

- Transfer ASCII "A" serially at 9600 baud continuously

```
START:  MOV TMOD, #20H     ;Put T1 in mode2
        MOV TH1, #-3       ;9600 baud
        MOV SCON, #50H     ;8b, 1stop, 1start, REN enabled
        SETB TR1           ;start timer T1
AGAIN:  CLR TI             ;ready to transmit
        MOV SBUF, #'A'     ;letter A is to be transmitted
HERE:   JNB TI, HERE       ;poll TI until all the bits are transmitted
        SJMP AGAIN         ;while(1) loop (forever loop)
```


Steps to Receive a Byte

1. Program T1 for Mode2 (TMOD ← 0x20)
2. Load TH1 with the initial value (baud rate dependant) (TH1 ← FD / FA / F4 / E8)
3. Program SCON for Mode1 (SCON ← 0x50)
4. Start Timer1 (setb TR1)
5. Clear RI
6. Wait until RI becomes 1 (jnb RI, not_done)
7. Store SBUF (A ← SBUF)
8. Go back to Step5 for next byte

Example: Receive Data

- Receive bytes serially and display them on P1, continuously.

```
START:  MOV TMOD, #20H     ;T1 in mode 2
        MOV TH1, #-3       ;9600 baud
        MOV SCON, #50H     ;8b, 1start, 1stop
        SETB TR1           ;start T1
AGAIN:  CLR RI             ;ready to receive a byte
HERE:   JNB RI, HERE       ;wait until one byte is Rx-ed
        MOV A, SBUF        ;read the received byte from SBUF
        MOV P1, A          ;display on P1
        SJMP AGAIN         ;while (1)
```

Serial Ports with Interrupts

- Using serial port with interrupts is THE way it was intended to be used.
- Both the RI and TI flags raise the Serial interrupt (S0), if S0 is enabled in IE.
  - ISR for S0 is at 0x0023
- Simple Case
  - Transmit is polling based (Poll TI flag) and Receive is interrupt driven
  - Transmit is interrupt driven and Receive is polling based (Poll RI flag)
- In these cases, the ISR of S0 will check for the appropriate flag and either copy data to or from SBUF

Serial Ports with Interrupts

- General Case
  - 8051 is in full duplex mode, I.e. receives and transmits data continuously
  - Both Transmit and Receive is interrupt driven
- Write the ISR for S0 such that
  - ISR must first check which one of RI and TI raised the S0 interrupt
  - If RI is set, then read data from SBUF to a safe place and clear RI
  - If TI is set, then copy the next character to be transmitted onto SBUF and clear TI.

Example : Simple case

- 8051 gets data from P1 and sends it to P2 continuously while receiving from Serial port. Serial port data is to be displayed on P0

```
        ORG 0
        LJMP MAIN           ; avoid the IVT

        ORG 23H
        LJMP SERIAL         ; serial port ISR

        ORG 30H
MAIN:   MOV P1, #0FFH       ; P1 as input port
        MOV TMOD, #20       ; T1 in mode 2
        MOV TH1, #-3        ; 9600 baud
        MOV SCON, #50H      ; 8b, 1start, 1stop
        MOV IE, #10010000B  ; enable S0 interrupt
        SETB TR1            ; enable T1
BACK:   MOV A, P1
        MOV P2, A
        SJMP BACK
```

```
        ORG 100H
SERIAL: JB TI, TRANS
        MOV A, SBUF         ;copy received data
        MOV P0, A           ;display it on P0
        CLR RI              ;clear RI
        RETI
TRANS:  CLR TI              ;do nothing
        RETI                ;ISR does not handle TX
        end
```