EXPERIMENT NO. – 9

TITLE: Interfacing serial port with PC both side communication.

NAME OF THE STUDENT: Sanket Arun Adsule

ROLL NO: 3041068

DATE OF PERFORMANCE: 26/9/22

DATE OF SUBMISSION:

SIGNATURE:

EXPERIMENT NO.-9

TITLE: Interfacing serial port with PC both side communication.

AIM: Write a program in Embedded C to transfer the message and receive serially at 9600 baud, 8 bit data, 1 stop bit.

OBJECTIVE:

1. To study the RS232 standard for serial communication
2. To Study the on-chip USART of PIC Microcontroller

3. To calculate the desired baud rate.
4. To interface MAX232 with PIC Microcontroller for serial communication.

THEORY:

1. Serial Communication:

The microcontroller is parallel device that transfers eight bits of data simultaneously over eight data lines to parallel I/O devices. However, in many situations, parallel data transfer is impractical. For example, parallel data transfer over a long is very expensive. Hence, a serial communication is widely used in long distance communication.

In serial data communication, 8-bit data is converted to serial bits using a parallel in serial out shift register and then it is transmitted over a single data line. The data byte is always transmitted with least significant bit first.

Serial ports are a type of computer interface for serial communication that complies with the RS-232 standard. They are 9-pin connectors that relay information, incoming or outgoing, one byte at a time. Each byte is broken up into a series of eight bits, hence the term serial port. Serial ports are controlled by a special chip call a UART (Universal Asynchronous Receiver Transmitter).

1.1 Communication links:

Serial communication is classified into three types of communication.

a. **Simplex communication link:** In simplex transmission, the line is dedicated for transmission. The transmitter sends and the receiver receives the data.

b. **Half duplex communication link:** In half duplex, the communication link can be used for either transmission or reception. Data is transmitted in only one direction at a time.

c. **Full duplex communication link:** If the data is transmitted in both ways at the same time, it is a full duplex i.e. transmission and reception can proceed simultaneously. This communication link requires two wires for data, one for transmission and one for reception.

1.2 Types of Serial communication:

Serial data communication uses two types of communication.

- **Synchronous serial data communication:** In this transmitter and receiver are synchronized. It uses a common clack to synchronize the receiver and the transmitter. First the synch character is sent and then the data is transmitted. This format is generally used for high speed transmission. In Synchronous serial data communication a block of data is transmitted at a time.

- **Asynchronous Serial data transmission:** In this, different clock sources are used for transmitter and receiver. In this mode, data is transmitted with start and stop bits. A transmission begins with start bit, followed by data and then stop bit. For error checking purpose parity bit is included just prior to stop bit. In Asynchronous serial data communication a single byte is transmitted at a time.

1.3 Baud rate:

The rate at which the bits are transmitted is called baud or transfer rate. The baud rate is the reciprocal of the time to send one bit. In asynchronous transmission, baud rate is not equal to number of bits per second. This is because; each byte is preceded by a start bit and followed by parity and stop bit. For example, in synchronous transmission, if data is transmitted with 9600 baud, it means that 9600 bits are transmitted in one second. For bit transmission time = 1 second/ 9600 = 0.104 ms.

2. RS-232 standards:

To allow compatibility among data communication equipment made by various manufactures, an interfacing standard called Rs232 was set by the Electronics Industries Association (EIA) in 1960. In 1963 it was modified and called RS232A. RS232B and RS232C were issued in 1965 and 1969, respectively. Today Rs232is the most widely used serial I/O interfacingstandard. This standard is used in PCs and numerous equipments. However, since the standard was set long before the advent of logic family, its input and output voltage levels are not TTL compatible. In RS232, a logic one (1) is represented by -3 to -25V and referred as MARK while logic zero (0) is represented by +3 to +25V and referred as SPACE. For this reason to connect any RS232 to a microcontroller system we must use voltage converters such as MAX232 to convert the TTL logic level to RS232 voltage levels and vice-versa. MAX232 IC chips are commonly referred as line drivers.

2.1 Serial Data Format:

The serial data format includes one start bit, between five and eight data bits, and one stop bit. A parity bit and an additional stop bit might be included in the format as well. The figure 4.1 below illustrates the serial data format.

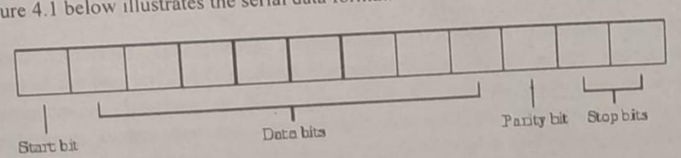


Figure 1.1: Serial data format

The format for serial port data is often expressed using the following notation: Number of data bits - parity type - number of stop bits.

For example, 8-N-1 is interpreted as eight data bits, no parity bit, and one stop bit, while 7-E-2 is interpreted as seven data bits, even parity, and two stop bits.

The data bits are often referred to as a character because these bits usually represent an ASCII character. The remaining bits are called framing bits because they frame the data bits.

2.2. Serial Port Pin Function

DB-9 Pin No.	Abbreviation	Full Name	Function
Pin 3	TXD	Transmit Data	Serial Data Output
Pin 2	RXD	Receive Data	Serial Data Input
Pin 7	RTS	Request To Send	This line informs the Modem that the UART is ready to exchange data.
Pin 8	CTS	Clear To Send	This line indicates that the Modem is ready to exchange data.
Pin 6	DSR	Data Set Ready	This tells the UART that the modem is ready to establish a link.
Pin 5	SG	Signal Ground	Ground
Pin 1	DCD	Data Carrier Detect	When the modem detects a "Carrier" from the modem at the other end of the phone line, this Line becomes active.
Pin 4	DTR	Data Terminal Ready	This is the opposite to DSR. This tells the Modem that the UART is ready to link.
Pin 9	RI	Ring Indicator	Goes active when modem detects a ringing signal from the PSTN.

Table 1.1: D Type 9 Pin and Serial Port Pin Functions

3. EUSART Module in PIC Microcontroller

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is one of the two serial I/O modules. (Generically, the USART is also known as a Serial Communications Interface or SCI.) The EUSART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also be configured as a halfduplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc. The Enhanced USART module implements additional features, including automatic baud rate detection and calibration, automatic wake-up on Sync Break reception and

12-bit Break character transmit. These make it ideally suited for use in Local Interconnect Network bus (LIN bus) systems. The EUSART can be configured in the following modes:

- Asynchronous (full-duplex) with:
  - Auto-wake-up on Break signal
  - Auto-baud calibration
  - 12-bit Break character transmission
- Synchronous – Master (half-duplex) with
- selectable clock polarity
- Synchronous – Slave (half-duplex) with selectable
- clock polarity

3.1 PIN DESCRIPTION

Signal	Pin No.	Symbol
TXD	25	RC6/TX/CK
RXD	26	RC7/RX/DT/SDO

The pins of the Enhanced USART are multiplexed withPORTC. In order to configure RC6/TX/CK and RC7/RX/DT/SDO as an EUSART:

- bit SPEN (RCSTA<7>) must be set (= 1)
- bit TRISC<7> must be set (= 1)
- bit TRISC<6> must be set (= 0)

3.2 EUSART Register Map:

The operation of the Enhanced USART module is controlled through three registers:

- Transmit Status and Control (TXSTA)
- Receive Status and Control (RCSTA)
- Baud Rate Control (BAUDCON)

SFR	Description	Access	Reset Value	Address
TXSTA	Transmit Status & Control Register	Read/Write	0x02	0xFAC
RCSTA	Receive Status & Control Register	Read/Write	0x00	0xFAB
BAUDCON	Baud rate control register	Read/Write	0x00	0xFB8
SPBRGH	Baud rate generator register higher byte	Read/Write	0x00	0xFB0
SPBRG	Baud rate generator register lower byte	Read/Write	0x00	0xFAF
TXREG	Transmission Register	Write	0x00	0xFAD
RCREG	Receive Register	Read	0x00	0xFAE

3.3 EUASRT Register Description

3.3.1 Transmit Status & Control Register

Bit No.	Control Bit	Description
Bit 7	CSRC	Clock Source Select bit Asynchronous mode: Don't care. Synchronous mode: 1 = Master mode; 0 = Slave mode
Bit 6	TX9	9-Bit Transmit Enable bit 1 = Selects 9-bit Tx; 0 = Selects 8-bit Tx
Bit 5	**TXEN**	**Transmit Enable bit** **1 = Transmit enabled ; 0 = Transmit disabled**
Bit 4	**SYNC**	**EUSART Mode Select bit** **1 = Synchronous mode; 0 = Asynchronous mode**
Bit 3	SENDB	Asynchronous mode: Send Break Character bit Synchronous mode: Don't care.
Bit 2	**BRGH**	**High Baud Rate Select bit** **Asynchronous mode: 1 = High speed; 0 = Low speed** **Synchronous mode: Unused in this mode.**
Bit 1	**TRMT**	**Transmit Shift Register Status bit** **1 = TSR empty; 0 = TSR full**
Bit 0	TX9D	9th bit of Transmit Data Can be address/data bit or a parity bit.

NOTE: Only highlighted bits are applicable to this exercise.

1.3.2 Receive Status & Control Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

Bit No.	Control Bit	Description
Bit 7	SPEN	**Serial Port Enable bit** **1 = Serial port enabled (configures RX/DT and TX/CK pins as serial port pins)** **0 = Serial port disabled (held in Reset)**
Bit 6	RX9	9-Bit Receive Enable bit 1 = Selects 9-bit reception; 0 = Selects 8-bit reception
Bit 5	SREN	Single Receive Enable bit Asynchronous mode: Don't care. Synchronous mode – Master: 1 = Enables single receive 0 = Disables single receive This bit is cleared after reception is complete. Synchronous mode – Slave: Don't care.
Bit 4	**CREN**	**Continuous Receive Enable bit** **Asynchronous mode:** **1 = Enables receiver** **0 = Disables receiver** **Synchronous mode:** **1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)** **0 = Disables continuous receive**
Bit 3	ADDEN	Address Detect Enable bit Asynchronous mode 9-bit (RX9 = 1): 1 = Enables address detection, enables interrupt and loads the receive buffer when RSR<8> is set 0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit Asynchronous mode 9-bit (RX9 = 0): Don't care.
Bit 2	FERR	Framing Error bit 1 = Framing error (can be updated by reading RCREG register and receiving next valid byte) 0 = No framing error
Bit 1	OERR	Overrun Error bit 1 = Overrun error (can be cleared by clearing bit CREN) 0 = No overrun error
Bit 0	RX9D	9th bit of Received Data This can be address/data bit or a parity bit and must be calculated by user firmware.

NOTE: Only highlighted bits are applicable to this exercise.

1.3.3 Baud rate control register

R/W-0	R-1	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	RXDTP	TXCKP	BRG16		WUE	ABDEN
bit 7							bit 0

the baud rate. In Synchronous mode, BRGH is ignored. Table 1.1 shows the formula for computation of the baud rate for different EUSART modes which only apply in Master Mode (internally generated clock). Given the desired baud rate and FOSC, the nearest integer value for the SPBRGH:SPBRGregisters can be calculated using the formulas in Table 1.1. From this, the error in baud rate can be determined. It may be advantageous to use the high baud rate (BRGH = 1), or the 16-bit BRG to reduce the baud rate error, or achieve a slow baud rate for a fast oscillator frequency. Writing a new value to the SPBRGH:SPBRG registers causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

Configuration Bits			BRG Mode / EUSART Mode	Baud Rate Formula [n= SPBRGH:SPBRG]
SYNC	BRG16	BRGH		
0	0	0	8-bit / Asynchronous	BR = Fosc/[64(n+1)]
0	0	1	8-bit / Asynchronous	BR = Fosc /[16(n+1)]
0	1	0	16-bit / Asynchronous	
0	1	1	16-bit / Asynchronous	BR = Fosc/[4(n+1)]
1	0	x	8-bit / Synchronous	
1	1	x	16-bit / Synchronous	

Table 1.1: Baud rate formulae

3.5 Baud rate Calculation

Mode selection

a. BRG Mode = 8-bit by setting BRG16 bit in BAUDCON Register
b. EUSART mode = Asynchronous by clearing SYNC bit in TXSTA Register
c. Formulae Desired **Buad Rate = Fosc / [4(SPBRGH:SPBRG+1)]** selected by setting BRG16 bit in TXSTA Register

For Fosc = 20MHz, Baud rate = 9600

SPBRGH:SPBRG = [Fosc/4(Baud rate)]-1

$= [20x10^6/(4x9600)] - 1$

$= [519]_{10} = [0207]_{16}$

% Error = (Calculated baud rate - Desired Baud Rate)/Desired Baud Rate
= (Fosc / [4(SPBRGH:SPBRG+1)] – 9600)/Desired Baud Rate
= ([20x10^6/[4x(520+1)]] – 9600)/Desired Baud Rate
= (9615 – 9600)/9600

% Error = 0.16 %

Therefore to reduce error we have selected **SPBRGH = 0x02** and **SPBRG = 0x08**

For Fosc = 48MHz, Baud rate = 9600

SPBRGH:SPBRG = [Fosc/4(Baud rate)]-1
= $[48x10^6/(4x9600)] - 1$
= $[1249]_{10} = [04E1]_{16}$
Therefore SPBRGH = 0x04 and SPBRG = 0xE1

% Error = (Calculated baud rate - Desired Baud Rate)/Desired Baud Rate
= (Fosc / [4(SPBRGH:SPBRG+1)] – 9600)/Desired Baud Rate
= $([48x10^6/[4x(1249+1)]] - 9600)$/Desired Baud Rate
= (9600 – 9600)/9600
% Error = 0.0 %
Therefore we have selected **SPBRGH = 0x04** and **SPBRG = 0xE1**

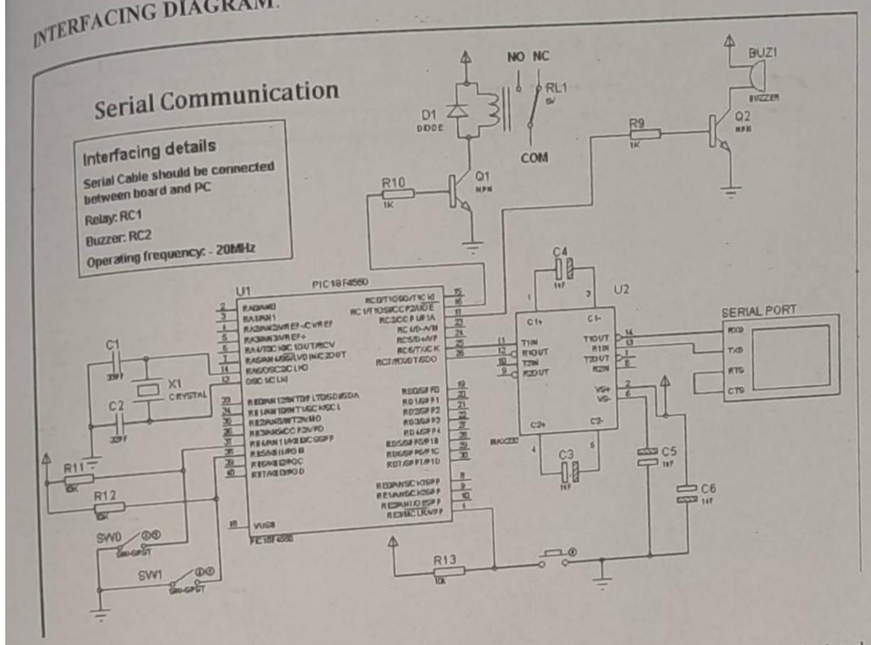
3.6 To set up an Asynchronous Transmission:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bitSYNC and setting bit SPEN.
3. If the signal from the TX pin is to be inverted, set the TXCKP bit.
4. If interrupts are desired, set enable bit TXIE.
5. If 9-bit transmission is desired, set transmit bit TX9.
6. Enable the transmission by setting bit TXEN which will also set bit TXIF.
7. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
8. Load data to the TXREG register (starts transmission).
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

3.7 To set up an Asynchronous Reception:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bitSYNC and setting bit SPEN.
3. If the signal at the RX pin is to be inverted, set the RXDTP bit.
4. If interrupts are desired, set enable bit RCIE.
5. If 9-bit reception is desired, set bit RX9.
6. Enable the reception by setting bit CREN.
7. Flag bit, RCIF, will be set when reception is complete and an interrupt will be generated if enable bit, RCIE, was set.
8. Read the RCSTA register to get the 9th bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREG register.
10. If any error occurred, clear the error by clearing enable bit CREN.
11. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

INTERFACING DIAGRAM:



ALOGORITHM

1. Calculate the SPBRGH: SPBRG value for desired baud rate. Load the calculated value in SPBRGH: SPBRG.
2. Configure Port pin RC6 as output and port pin RC7 as input.
3. Set BRG Mode = 8-bit by setting BRG16 bit in BAUDCON Register.
4. Set EUSART mode = Asynchronous by clearing SYNC bit in TXSTA Register
5. Enable the Serial Port by setting SPEN bit in RCSTA Register
6. Enable the Transmission by setting TXEN bit TXSTA Register
7. Check TRMT bit in TXSTA register for TXREG empty. Load data to the TXREG register if TRMT = 1
8. Repeat the step 6 until complete message transmitted
9. Enable the continuous reception by setting CREN bit in RCSTA register
10. Poll the RCIF bit in PIR1 register to check data is received
11. Read the 8-bit received data by reading the RCREG register
12. Do the control action as per data received.
13. Repeat the steps from 10.