Bài 7 : Khảo sát bộ truyền nhận nối tiếp USART

Nội dung:

Nd1. Khảo sát bộ truyền nối tiếp USART của PIC.

Nd2. Trao đổi thông tin giữa 2 PICDEM PIC18 với nhau, sử dụng cáp chéo.

Nd3. Trao đổi thông tin giữa PICDEM PIC18 với PC qua cổng COM (RS-232C), phần mềm Terminal, sử dụng cáp thẳng.

Yêu cầu:

Yc1. Viết chương trình sử dụng khối truyền nhận nối tiếp USART của PIC. Lập trình chọn chế độ hoạt động, truyền trực tiếp, nhận qua ngắt quãng và hiện ra LCD.

Yc2. Thực hiện truyền nhận ký tự giữa 2 PICDEM PIC18 với nhau.

Yc3. Thực hiện truyền nhận ký tự giữa PICDEM PIC18 với PC.

7.1 Thực hiện yêu cầu 1:

Bước 1. Tạo dự án mới Tn07, tập tin nguồn Tn07_Usart.c.

Bước 2. Khởi động LCD, nút nhấn RA5.

Bước 3. Tạo module Usart.c chứa các hàm:

- **`Usart_init()`** : khởi động USART.
- **`Usart_isr()`** : xử lý ngắt quãng ưu tiên thấp cho giao tiếp Usart, kiểm tra xác nhận có ngắt quãng, kiểm tra xem quá trình nhận ký tự có lỗi xảy ra hay không. Nếu nhận không có lỗi thì gọi hàm **`Usart_process()`** (xử lý nhận ký tự và hiện ra màn hình LCD). Ngược lại, gọi hàm **`Rcerr_process()`** để xử lý lỗi. Các hàm **`Usart_process()`** và **`Rcerr_process()`** được viết trong module chính.
- **`Send_char()`** : truyền một ký tự ra cổng USART.

Bước 4. Khởi động khối EUSART1 để hoạt động ở chế độ sau:

- Chế độ truyền bất đồng bộ (Asynchronous), 8 bit, High speed bằng thanh ghi TXSTA1.

REGISTER 20-1: TXSTAx: TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

- Cấu hình yêu cầu có TXSTA1 = B'00100100' (sinh viên kiểm tra lại).
- Cấu hình chế độ nhận bằng thanh ghi RCSTA1.

REGISTER 20-2: RCSTAx: RECEIVE STATUS AND 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

- Cấu hình yêu cầu có RCSTA1 = B'10010000' (sinh viên kiểm tra lại).
- Tốc độ truyền nhận 9600 bit/s. Sinh viên tra bảng số liệu có sẵn trong data sheet (table 20-3, p.255) để chọn giá trị cho bộ phát xung SPBRG.
- Cấu hình ngắt nhận dữ liệu với mức ưu tiên thấp.
- Định nghĩa một số hằng

```
#define     TX1_IO      TRISCbits.RC6
#define     RX1_IO      TRISCbits.RC7
#define     TX1_CFG     0b00100100  //TXEN=1,BRGH=1
#define     RC1_CFG     0b10010000  //SPEN=1,CREN=1
```

- Hàm **Usart_init()**

```
void Usart_init()
{   TXSTA1=TX1_CFG;      //Chon TXSTA1
    SPBRG1=BAUDRATE;
    RCSTA1=RC1_CFG;      //Chon RCSTA1
    RCONbits.IPEN=1;
    IPR1bits.RC1IP=0;    //ngat nhan ky tu uu tien thap
    PIR1bits.RC1IF=0;    //Xoa co ngat nhan
    PIE1bits.RC1IE=1;    //Cho phep dung ngat nhan
    INTCONbits.GIEH=1;   //Cho phep ngat toan cuc
    INTCONbits.GIEL=1;
    TX1_IO=XUAT;RX1_IO=NHAP;
}
```

Bước 5. Hàm **Usart_isr()** thực hiện các việc sau:

- Kiểm tra xác nhận có ngắt nhận dữ liệu nối tiếp (cờ RC1IF), nếu có thì xoá cờ ngắt và làm tiếp các bước kế.
- Kiểm tra 2 bit lỗi FERR và OERR trong thanh ghi RCSTA1. Nếu không có lỗi thì gọi hàm **Usart_process()** để nhận ký tự, ngược lại gọi hàm **Rcerr_process()** để xóa lỗi.

```
void interrupt low_priority Usart_isr()
{   if ((PIE1bits.RC1IE==1)&&(PIR1bits.RC1IF==1))
    {   PIR1bits.RC1IF=0;
        if (RCSTA1 & RCERROR == 0)
            Usart_process();
        else
            Rcerr_process();
    }
}
```

Bước 6. Hàm **Send_char()** thực hiện truyền 1 ký tự :

```
void Send_char(char c)
{
    TXREG1=c;
    while (PIR1bits.TX1IF==0);
}
```

Bước 7. Trong module chính Tn07_Usart.c, viết hàm **Usart_process()** thực hiện :

- Nhận ký tự từ RCREG1.
- Hiện ký tự nhận được ra LCD ở hàng 2, từ trái sang phải, đến cuối thì quay trở về đầu hàng.
- Xuất ký tự ra LED.

```
char    kytu_tr,kytu_nh,cot_tr,cot_nh;
#define HANG_NH     1
void Usart_process()
{   kytu_nh=RCREG1;
    lcd_gotoxy(HANG_NH,cot_nh);putchar(kytu_nh);
    if ((++cot_nh)==16) cot_nh=0;
}
```

Bước 8. Viết hàm **Rcerr_process()** thực hiện xóa lỗi và xuất trị 0xFF ra LED.

```
void Rcerr_process()
{   RCSTA1bits.CREN=0;  //xoa loi FERR va OERR
    RCSTA1bits.CREN=1;
    led=0xFF;
}
```

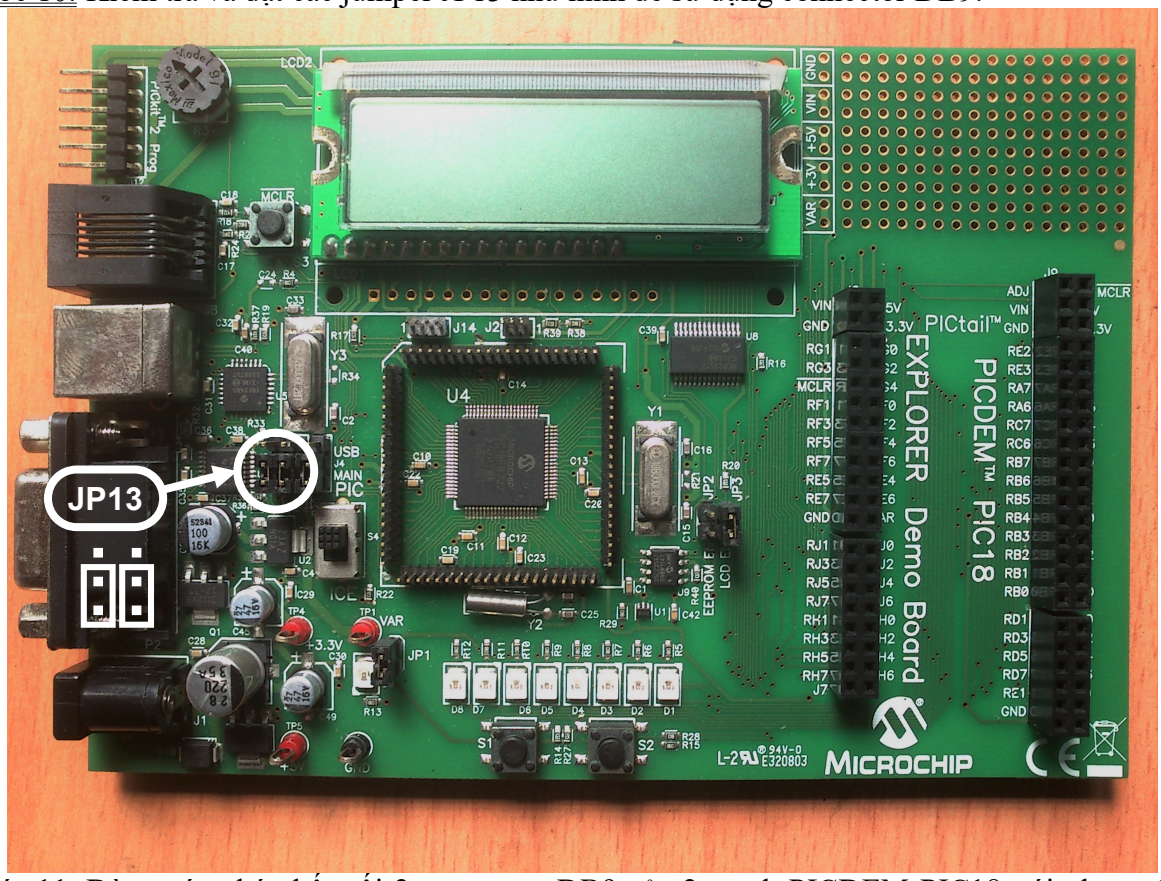
Bước 9. Kiểm tra nút RA5, thực hiện:

- Truyền ký tự trong biến **kytu_tr** (gọi hàm **Send_char()** để truyền).

- Hiện ký tự đã truyền ra LCD ở hàng 1, từ trái sang phải, đến cuối thì quay trở về đầu hàng.
- Tăng mã ký tự trong biến **kytu_tr**.

7.2 Thực hiện yêu cầu 2: truyền/nhận PICDEM PIC18 với nhau

Bước 10. Kiểm tra và đặt các jumper JP13 như hình để sử dụng connector DB9.



Bước 11. Dùng cáp chéo kết nối 2 connector DB9 của 2 mạch PICDEM PIC18 với nhau, chạy chương trình và quan sát.

7.3 Thực hiện yêu cầu 3: truyền/nhận PICDEM PIC18 với PC

Bước 12. Nếu PC có đầu kết nối DB9 thì dùng cáp thẳng nối giữa PICDEM PIC18 với PC.

Bước 13. Trên PC, chạy chương trình Terminal.exe để thực hiện truyền nhận.

Bước 14. Với PC không hỗ trợ kết nối DB9, thì việc truyền nhận sẽ được thực hiện qua cổng USB (Sinh viên làm thêm, xem tài liệu PICDemPIC18 User's guide, trang 22).

7.4 Bài tập

a) Thực hiện truyền nhận với tốc độ 1200 bit/s, 2400 bit/s.

b) Viết chương trình điều khiển PICDEM PIC18 từ xa thông qua UART từ một PICDEM PIC18 khác. Ví dụ gởi ký tự 's' thực hiện sáng led, gởi ký tự 't' thì tắt led.