```
/*Interfacing Bluetooth HC05 Modul with PIC 18F25k22
 * The serial bluetooth App from Kai morich will be used
 * File:   HC05_Bluetooth.c
 * Author: Lasaros Goumas
 *Created on April 16, 2026, 1:32 PM
 */


/* Includes
 ******************************************************************************/
#include <xc.h>
#include <p18cxxx.h>            //PIC 18F25k22 Controller


/*Configuration
 ******************************************************************************/
#pragma config FOSC = INTIO67
#pragma config PWRTEN = ON
#pragma config WDTEN = OFF
#pragma config PBADEN = OFF
#pragma config LVP = OFF
#pragma config CP1 = OFF
#pragma config CPB = OFF
#pragma config WRT0 = OFF
#pragma config WRTC = OFF
#pragma config EBTR0 = OFF
#pragma config EBTRB = OFF


/*Declarations
 ******************************************************************************/
#define _XTAL_FREQ 8000000     // Fosc frequency for _delay() library#include
<xc.h>
#define Baud_rate 9600
#define LED_info PORTB
#define Blue_LED PORTCbits.RC0
#define Yellow_LED PORTCbits.RC1


/*Functions
 ******************************************************************************/
void init_PIC (void){
    TRISB = 0x00;
    ANSELB = 0x00;          //RB PINs are outputs and digital
    LATB = 0x00;
    ANSELC = 0x00;          //RC Pins sind digital
    LATC = 0x00;
    OSCCON = 0b01101110;    //8MHz; Internal Oscilator; stable
}

void init_EUSART (void){
    TRISC = 0b11000000;     //RC<7:6> are RX1,TX1 Ports
    INTCONbits.GIE = 0;
    INTCONbits.PEIE = 0;
```

```
    PIE1bits.RC1IE = 0;     //Global,peripheral,Receive Interrupts disabled
    TXSTA1bits.SYNC = 0;    //Asynchronous mode
    SPBRG1 = 51;
    SPBRGH1  = 0;           // for high baud_rate
    TXSTA1bits.BRGH = 1;    //High baud rate select bit
    BAUDCON1bits.BRG16 = 0; //8-bit Baud rate is used
    RCSTA1bits.SPEN = 1;    //TX1; RX1 serial port pins enabled
    RCSTA1bits.CREN = 1;    //Enable continuous reception
    RCSTA1bits.RX9 = 0;     //8-bit reception enabled
    TXSTA1bits.TXEN = 1;    //8-bit transmision enabled
}

void EUSART_Write(char data){
    while(!PIR1bits.TX1IF);   // warten bis TX frei
    TXREG1 = data;
}

void EUSART_Write_Text(const char *text){     //String senden
    while(*text){
        EUSART_Write(*text++);
    }
}


/*Main Routine
 *********************************************************************/
void main(void){
    init_PIC ();
    init_EUSART();
    EUSART_Write_Text("Start OK\r\n");

 while(1){
        if(PIR1bits.RC1IF){                 //Wait for interrupt flag
            if(RCSTA1bits.OERR){
                RCSTA1bits.CREN = 0;
                RCSTA1bits.CREN = 1;        //Clear possible overrun error
            }
            LED_info = RCREG1;              // Read The Received Data Buffer
            EUSART_Write_Text("Empfangen: ");

            if(LED_info == 0x31){            //ASCII "1"
                Blue_LED = 1;               // Blue LED ON
            EUSART_Write_Text("Blue ON\r\n");
            }
            else if(LED_info == 0x32){       //ASCII "2"
                Blue_LED = 0;               // Blue LED OFF
            EUSART_Write_Text("Blue OFF\r\n");
            }
            else if(LED_info == 0x33){       //ASCII "3"
                Yellow_LED = !Yellow_LED;   // Toggle Yellow LED
            EUSART_Write_Text("Yellow Toggle\r\n");
            }
        }
        }
```

```
}
```