Topic 5 : UART Serial Port



Problem 1 : An embedded systems using ATMEGA32 (working at 7.372800Mhz) communicate with Hyperterminal by UART0 serial interface. The system has 4 LEDs connected to PORTD (PD4.PD5,PD5,PD7). When a system receiving new character from the UART serial port the system will display this character to PORTC, and toggle the relevant LED( for example the character received is ‘0’ LED0 will be toggled, the character received is ‘1’ LED1 will be toggled and so on … the character received is ‘3’ LED3 will be toggled ). The system using Interupt method and UART setting with baud rate 19200, 8 bit, 1 stop bit, no parity. Write C program to control the system.

Problem **2** : An embedded systems using ATMEGA32 (working at 11.0592Mhz) communicate with Hyperterminal by UART0 serial interface. The system has 4 LEDs connected to PORTB (PB0.PB1,PB2,PB3). When a system receiving new character from the UART serial port the system will display this character to PORTD, and toggle the relevant LED ( for example the character received is ‘0’ LED0 will be toggled, the character received is ‘1’ LED1 will be toggled and so on … the character received is ‘3’ LED3 will be toggled ). The system using Interupt method and UART setting with baud rate 38400, 8 bit, 1 stop bit, no parity. Write C program to control the system.

Certainly! Let’s break down the solutions for both problems:

## **Problem 1: ATMEGA32 with 7.372800 MHz Clock Frequency**

### **Requirements:**

* ATMEGA32 microcontroller running at 7.372800 MHz.
* UART0 serial interface for communication with Hyperterminal.
* 4 LEDs connected to PORTD (PD4, PD5, PD6, PD7).
* Interrupt-based method for handling UART communication.
* UART settings: Baud rate 19200, 8-bit data, 1 stop bit, no parity.

### **Solution:**

Below is a C program that achieves the desired functionality:

#include <avr/io.h>

#include <avr/interrupt.h>

// Initialize UART

void UART\_Init() {

// Set baud rate to 19200

UBRRH = 0;

UBRRL = 51;

// Enable receiver and receiver interrupt

UCSRB = (1 << RXEN) | (1 << RXCIE);

// Set frame format: 8 data bits, 1 stop bit, no parity

UCSRC = (1 << URSEL) | (1 << UCSZ1) | (1 << UCSZ0);

}

// UART receive interrupt handler

ISR(USART\_RXC\_vect) {

char receivedChar = UDR; // Read received character

// Display character on PORTC

PORTC = receivedChar;

// Toggle relevant LED on PORTD

if (receivedChar >= '0' && receivedChar <= '3') {

PORTD ^= (1 << (receivedChar - '0' + 4));

}

}

int main() {

// Initialize ports

DDRC = 0xFF; // PORTC as output

DDRD = 0xF0; // PD4-PD7 as output

// Initialize UART

UART\_Init();

// Enable global interrupts

sei();

while (1) {

// Main loop (other tasks if needed)

}

return 0;

}

## **Problem 2: ATMEGA32 with 11.0592 MHz Clock Frequency**

### **Requirements:**

* ATMEGA32 microcontroller running at 11.0592 MHz.
* UART0 serial interface for communication with Hyperterminal.
* 4 LEDs connected to PORTB (PB0, PB1, PB2, PB3).
* Interrupt-based method for handling UART communication.
* UART settings: Baud rate 38400, 8-bit data, 1 stop bit, no parity.

### **Solution:**

Below is a C program for the second problem:

#include <avr/io.h>

#include <avr/interrupt.h>

// Initialize UART

void UART\_Init() {

// Set baud rate to 38400

UBRRH = 0;

UBRRL = 16;

// Enable receiver and receiver interrupt

UCSRB = (1 << RXEN) | (1 << RXCIE);

// Set frame format: 8 data bits, 1 stop bit, no parity

UCSRC = (1 << URSEL) | (1 << UCSZ1) | (1 << UCSZ0);

}

// UART receive interrupt handler

ISR(USART\_RXC\_vect) {

char receivedChar = UDR; // Read received character

// Display character on PORTD

PORTD = receivedChar;

// Toggle relevant LED on PORTB

if (receivedChar >= '0' && receivedChar <= '3') {

PORTB ^= (1 << (receivedChar - '0'));

}

}

int main() {

// Initialize ports

DDRD = 0xFF; // PORTD as output

DDRB = 0x0F; // PB0-PB3 as output

// Initialize UART

UART\_Init();

// Enable global interrupts

sei();

while (1) {

// Main loop (other tasks if needed)

}

return 0;

}

Make sure to adjust the pin connections and other settings according to your hardware setup. These programs should handle UART communication and LED toggling as specified in the problems. 😊👍