Wroclaw University Of Science And Technology

Caner Olcay 276715

Microcontroller

14.01.2025

Task 1 & 2: RS232 Echo Test and Single Character Sending

Objective:

To validate the RS232 hardware connections and ensure reliable communication between the Arduino Leonardo and the Bray Terminal. This involves:

1. Performing an echo test to verify RS232 functionality.
2. Sending a single character ('A') in continuous mode to observe consistent data transmission.

Code:

```
#define F_CPU 16000000UL //
#include <avr/io.h>
#include <util/delay.h>

// UART başlatma fonksiyonu
void uart_init(unsigned int ubrr) {
    UBRR1H = (unsigned char)(ubrr >> 8);
    UBRR1L = (unsigned char)ubrr;
    UCSR1B = (1 << RXEN1) | (1 << TXEN1);
    UCSR1C = (1 << UCSZ11) | (1 << UCSZ10);
}


void uart_transmit(unsigned char data) {
    while (!(UCSR1A & (1 << UDRE1)));
    UDR1 = data;
}

int main(void) {
    uart_init(103);

    while (1) {
        uart_transmit('A');
        _delay_ms(1000);
    }
}
```

Output:

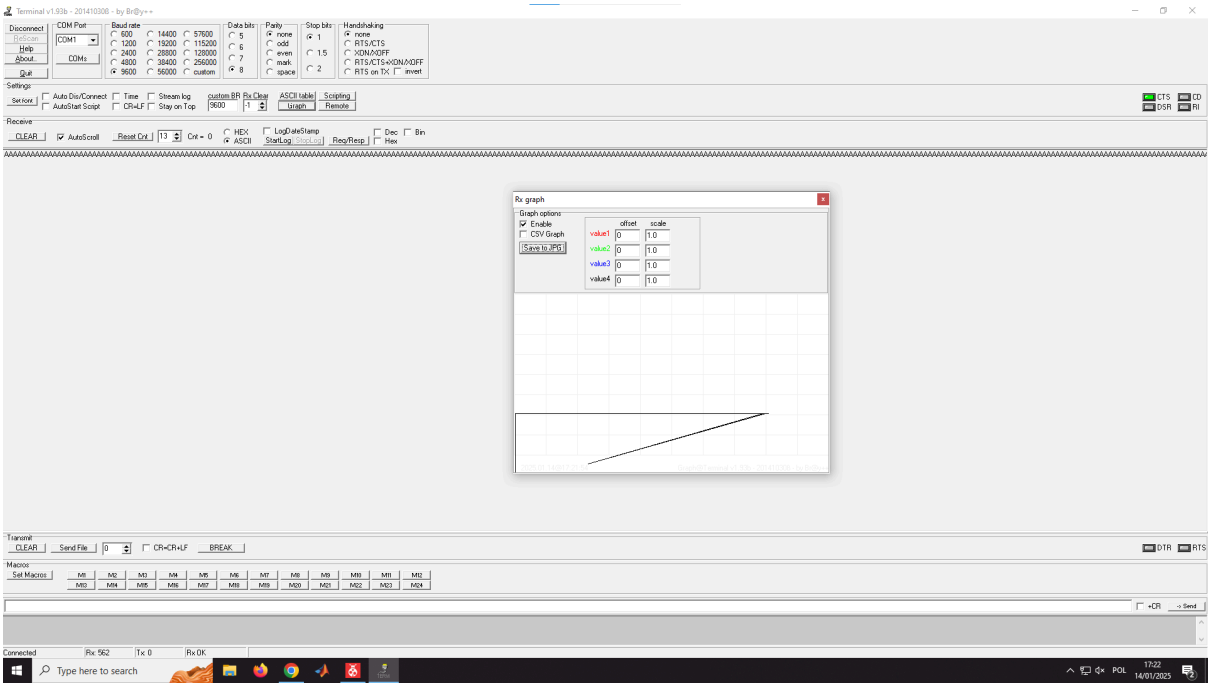


Figure 1: Bray Terminal showing continuous transmission of 'A'.

Explanation: This figure demonstrates continuous single-character sending, validating the RS232 communication between the Arduino Leonardo and the Bray Terminal.

Task 3: ADC Sampling and Data Transmission via RS232

Objective:

To implement ADC functionality on the Arduino Leonardo, read analog data from a potentiometer or sensor, and transmit the data to the Bray Terminal using RS232 communication. Additionally, visualize the ADC values using the Bray Terminal's graphing feature.

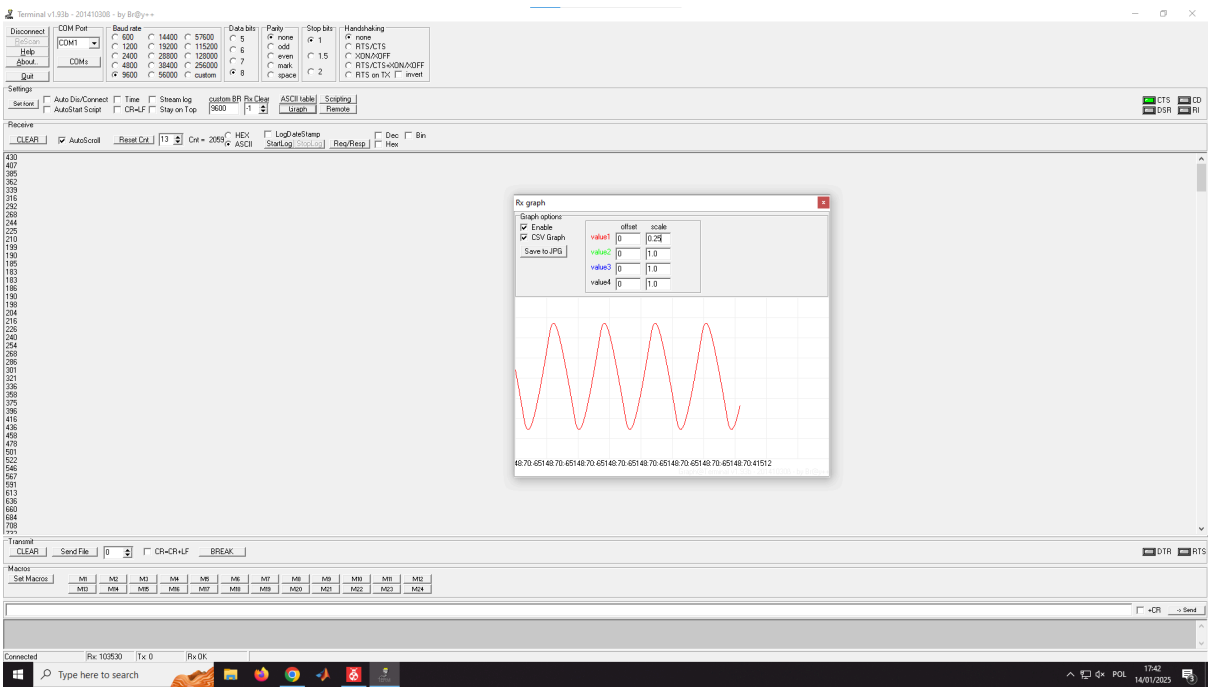
Code:

```
#include <avr/io.h>

#include <util/delay.h>

#include <stdio.h>

#define F_CPU 16000000UL

#define BAUD 9600

#define MYUBRR F_CPU/16/BAUD-1

void uart_init(unsigned int ubrr) {

    UBRR1H = (unsigned char)(ubrr >> 8);

    UBRR1L = (unsigned char)ubrr;

    UCSR1B = (1 << RXEN1) | (1 << TXEN1);

    UCSR1C = (3 << UCSZ10);

}

void uart_transmit(unsigned char data) {

    while (!(UCSR1A & (1 << UDRE1)));

    UDR1 = data;

}

void uart_send_string(char *str) {

    while (*str) {

        uart_transmit(*str++);

    }

}
```

```
void initPWM() {
    DDRB |= (1 << PB5); // Set PB5 as output
    TCCR1A |= (1 << WGM11) | (1 << COM1A1);
    TCCR1B |= (1 << WGM12) | (1 << WGM13) | (1 << CS11); // Prescaler 8
    ICR1 = 1999; // TOP for 1kHz PWM
}
void setPWM(uint16_t duty) {
    if (duty > ICR1) {
        duty = ICR1;
    }
    OCR1A = duty;
}
void initADC() {
    ADMUX = (1 << REFS0) | (1 << MUX2) | (1 << MUX1) | (1 << MUX0); // AVcc reference,
ADC7 input
    ADCSRA = (1 << ADEN) | (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0); // Enable ADC,
prescaler 128
}
uint16_t readADC() {
    ADCSRA |= (1 << ADSC); // Start conversion
    while (ADCSRA & (1 << ADSC)); // Wait for conversion to complete
    return ADC;
}
int main(void) {
    uart_init(MYUBRR);
    initPWM();
    initADC();
```

```
    while (1) {
        for (uint16_t i = 0; i <= 2000; i += 50) {
            setPWM(i);
            uint16_t adcValue = readADC(); // Read ADC value
            char buffer[10];
            sprintf(buffer, "%d\r\n", adcValue); // Format ADC value as string
            uart_send_string(buffer); // Send string over UART
            _delay_ms(100);
        }
        for (uint16_t i = 2000; i >= 50; i -= 50) {
            setPWM(i);
            uint16_t adcValue = readADC(); // Read ADC value
            char buffer[10];
            sprintf(buffer, "%d\r\n", adcValue);
            uart_send_string(buffer);
            _delay_ms(100);
        }
    }
}
```

Output:



1. **Figure 2:** Bray Terminal showing continuous ADC values.
    - Example: 512, 523, 540...
2. **Figure 3:** Graphical representation of ADC values using Bray Terminal's Graph mode.
    - Graph shows variation as the analog input changes (e.g., when a potentiometer is adjusted).

Explanation:

- The PWM signal is adjusted while reading ADC values. The ADC data is transmitted via RS232, and its real-time change is observed in both terminal text and graphical format.

Conclusion

This experiment demonstrates the successful implementation of:

1. RS232 communication via echo test and continuous character transmission.
2. Analog-to-Digital Conversion (ADC) to read analog inputs and transmit the data over RS232.
3. Visualization of ADC data in both textual and graphical formats using the Bray Terminal