Lab 7: Masauso Lungu

Link to my GitHub repository:

https://github.com/Masauso-L/Digital-electronis-2/tree/main/Labs/07-uart

Analog-to-Digital Conversion

1. Complete table with voltage divider, calculated, and measured ADC values for all five push buttons.

$$ADC = \frac{V_{PC0[A0]}}{V_{ref}} \cdot (2^n - 1) =$$

Push button	PC0[A0] voltage	ADC value (calculated)	ADC value (measured)
Right	0 V	0	0
Up	0.495 V	101	99
Down	1.202 V	246	256
Left	1.97 V	403	409
Select	3.18 V	651	639
none	5 V	1023	1023

2. Code listing of ACD interrupt service routine for sending data to the LCD/UART and identification of the pressed button. Always use syntax highlighting and meaningful comments:

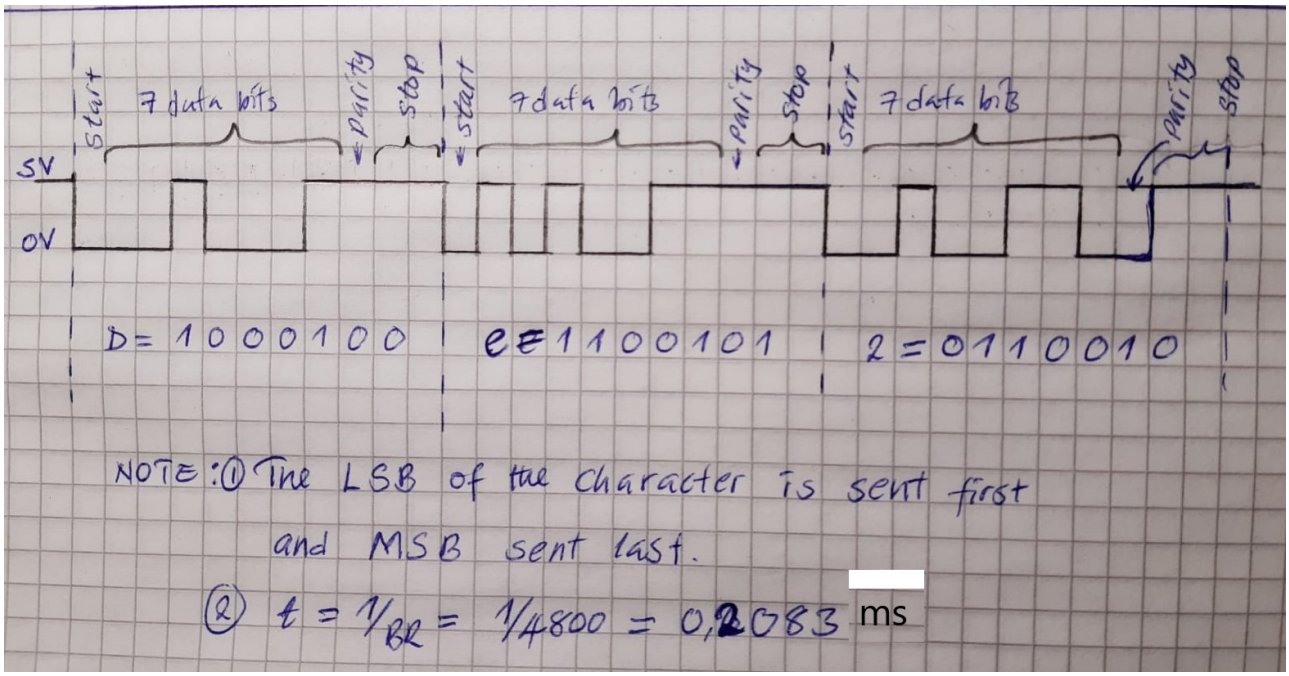
```
/* Interrupt service routines ----------------------------------------*/
/************************************************************************
 * Function: Timer/Counter1 overflow interrupt
 * Purpose:  Use single conversion mode and start conversion four times
 *           per second.
 ************************************************************************/
ISR(TIMER1_OVF_vect)
{
    // Start ADC conversion
    ADCSRA |= (1 << ADSC);


}

/************************************************************************
 * Function: ADC complete interrupt
 * Purpose:  Display value on LCD and send it to UART.
 ************************************************************************/
ISR(ADC_vect)
```

```
{
    uint16_t value = 0;
    char lcd_string[8];

    value = ADC;                    // Copy ADC result to 16-bit variable
    itoa(value, lcd_string, 10);  // Convert decimal value to string

    // WRITE YOUR CODE HERE
    lcd_gotoxy(7,0);
    lcd_puts("     ");
    lcd_gotoxy(7,0);
    lcd_puts(lcd_string);

    // Send data through UART
    if (value < 700)
    {

        uart_puts("ADC value in decimal: ");
        uart_puts(lcd_string);
        uart_puts("\r\n");   // \n means put cursor to next line, \r to the
beginning of the current line
    }

    // Convert to string in hex
    itoa(value, lcd_string, 16);
    lcd_gotoxy(13,0);
    lcd_puts("     ");
    lcd_gotoxy(13,0);
    lcd_puts(lcd_string);

    //Displaying the identity of the button pressed
    if(value >= 1016)
    {
        lcd_gotoxy(7, 1);
        lcd_puts("      ");
        lcd_gotoxy(7, 1);
        lcd_puts("None");
    }
    else if(value == 0)
    {
        lcd_gotoxy(7, 1);
        lcd_puts("     ");
        lcd_gotoxy(7, 1);
        lcd_puts("Right");
    }
    else if(value == 99)
    {
        lcd_gotoxy(7, 1);
        lcd_puts("     ");
        lcd_gotoxy(7, 1);
        lcd_puts("Up");
    }
    else if(value == 256 )
    {
```

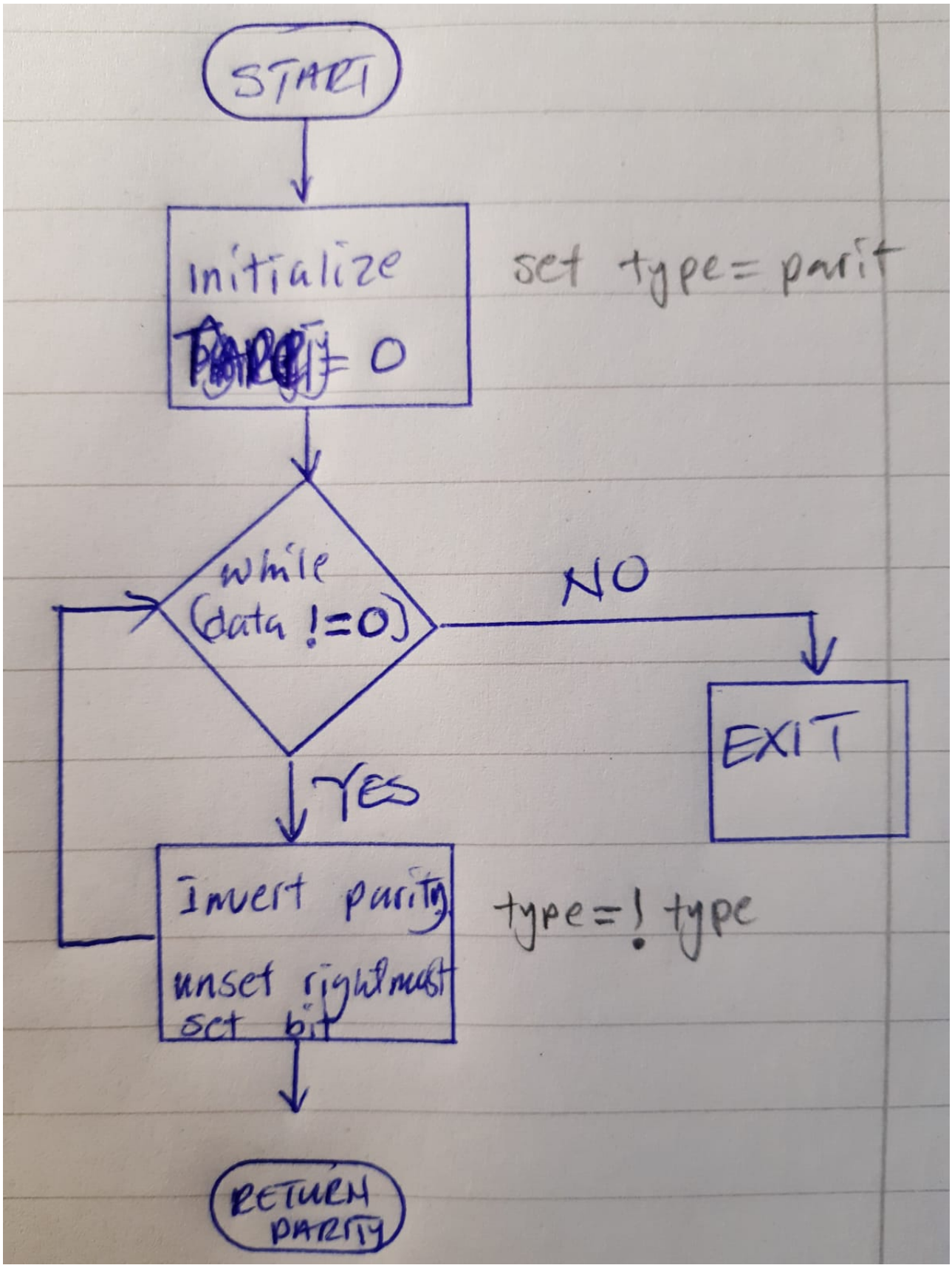
```
            lcd_gotoxy(7, 1);
            lcd_puts("     ");
            lcd_gotoxy(7, 1);
            lcd_puts("Down");
        }
        else if(value == 409)
        {
            lcd_gotoxy(7, 1);
            lcd_puts("     ");
            lcd_gotoxy(7, 1);
            lcd_puts("Left");
        }
        else if(value == 639)
        {
            lcd_gotoxy(7, 1);
            lcd_puts("      ");
            lcd_gotoxy(7, 1);
            lcd_puts("Select");
        }

    }
```

UART communication

1. (Hand-drawn) picture of UART signal when transmitting three character data `De2` in 4800 7O2 mode (7 data bits, odd parity, 2 stop bits, 4800 Bd).



2. Flowchart figure for function `uint8_t get_parity(uint8_t data, uint8_t type)` which calculates a parity bit of input 8-bit `data` according to parameter `type`. The image can be drawn on a computer or by hand. Use clear descriptions of the individual steps of the algorithms.



Temperature meter

Consider an application for temperature measurement and display. Use temperature sensor TC1046, LCD, one LED and a push button. After pressing the button, the temperature is measured, its value is displayed on the LCD and data is sent to the UART. When the temperature is too high, the LED will start blinking.

1. Scheme of temperature meter. The image can be drawn on a computer or by hand. Always name all components and their values.