Microprocessor Systems

Response Meter

Designer:

inż. Rafał Potempa

Supervisor:

dr inż. Krzysztof Taborek

Table of contents

1. Description

Task of the project was to design response meter based on Atmel AVR Atmega 32 microcontroller, utilizing two row LCD display.

The design uses:

- port A for communication and power of LCD display,
- port B for LED array,
- port C for input from tact-switch buttons.

The implementation uses interrupts of Timer 0 for control of LCD and Timer 1 for real time stopwatch, that count milliseconds between LED light on and click of proper button.

The first stage of work of program is waiting for any button input, while displaying on LCD:

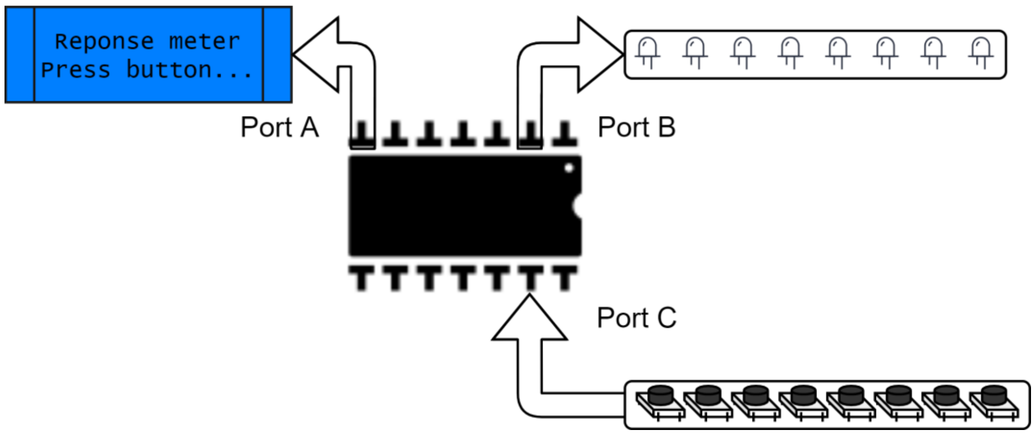
```
Response meter
Press button...
```

After detection of keyboard input, testing procedure starts. First LED in predefined sequence switches on, the global variable **`inProgress`** allows interrupt to increment value of **`totalTime`** global variable, with every millisecond. When correct button is pressed, **`inProgress`** is set to disallow the T1 interrupt counting time. The next diode in sequence turns on, and the process continues until the end of sequence.

When test is over, the program divides **`totalTime`** by number of elements is sequence and displays results in the form of:

```
Total: 10.654 s
Mean:   2.664 s
```

2. Functional Schematic



3. Listing of response-meter

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdio.h>
#include <util/delay.h>
#include "hd44780.c"

int sequence[4] = {1, 4, 5, 7};
unsigned int totalTime = 0;
int millisecond = 2000; // 16 000 000 Hz / 8 / 1000 Hz = 2 000
char buffer[16];
int inProgress = 0;

ISR (TIMER1_COMPA_vect) {
    if (inProgress)
        totalTime++;
}

void initTimer() {
    TCCR1B |= (1<<CS11)|(1<<WGM12);
    OCR1A = millisecond;
    TIMSK |= (1<<OCIE1A);
    sei();
}

void initDevices() {
    DDRC = 0x00; // keyboard
    PORTC = 0xff;
    DDRB |= 0xff; // led
    PORTB = 0xff;
}

void printResult() {
    LCD_Clear();
    LCD_GoTo(0,0);
    float average = totalTime/4.0;
    sprintf(buffer, "Total: %2d.%d s", (int)totalTime/1000, (int)totalTime%10
00);
    LCD_WriteText(buffer);
    LCD_GoTo(0,1);
    sprintf(buffer, "Mean:   %d.%d s", (int)average/1000, (int)average%1000);
    LCD_WriteText(&buffer);
}
```

```
void testing() {
    totalTime = 0;


    for (int i = 0; i < 4; i++)
    {
        inProgress = 1;
        PORTB = ~(1 << sequence[i]);
        while(PINC != PORTB); // waiting for correct button
        inProgress = 0;
    }
}

int main(void)
{
    initDevices();

    LCD_Initalize();
    initTimer();

    while(1){
        LCD_Clear();
        LCD_GoTo(1,0);
        LCD_WriteText("Response meter");
        LCD_GoTo(1,1);
        LCD_WriteText("Press button...");
        while(PINC == 0xff){};
        LCD_GoTo(1,1);
        LCD_WriteText("  Testing...    ");
        testing();
        printResult();
        break;
    }
}
```