

|  |
| --- |
| **Course: Embedded Electronic Devices and Programming** |
| Laboratory work № 1 |
| “Programming I/O ports of the microcontroller” |

|  |  |
| --- | --- |
| Student name and surname: | Daniil Grammatikopulo |
| Student code: | 79101 |
| Group: | 4103BDA |
| Lecturer: | Artjoms Ivanov |

|  |  |
| --- | --- |
| Date performed: | 16.03.2023 |

Riga

2023

Table of Content

[TASK FOR LABORATORY WORK 3](#_Toc129893694)

[The report should have followed structure: 3](#_Toc129893695)

[Individual variant of the task 4](#_Toc129893696)

[Microcontroller I/O diagram 5](#_Toc129893697)

[ALGORITHM 6](#_Toc129893698)

[PROGRAM CODE 7](#_Toc129893699)

[CONCLUSION 8](#_Toc129893700)

TASK FOR LABORATORY WORK

The report should have followed structure:

* to develop of the microcontroller, I/O diagram.
* to create an algorithm of main function for the ATMega48 microcontroller in accordance with an individual task.
* to create program code in Atmel Studio 7 environment.
* to debug the program in Atmel Studio 7 environment.
* to form time intervals, use functions that form delays.
* when debugging a program in the Atmel Studio 7 environment, the call of functions that generate delays must be commented out;
* the clock frequency of the ATMega48 microcontroller is 8 MHz.
* when the button is pressed on the input/output line of the port - logical "0", when the button is released on the input/output line of the port - logical "1”.
* the LED is turned on by logical "0" on the I/O line of the port and turned off by logical "1”.
* prepare a report on laboratory work.

Individual variant of the task

Initially, LED6 turns on and off at a frequency of 0.5 Hz. Each press of the SW1 button increases the LED switching period by 1 s, and each press of the SW5 button decreases the LED switching period by 0.3 s. The value of the LED switching period can vary from 0.2 s to 7.0 s.

Microcontroller I/O diagram

Изображение выглядит как диаграмма, схематичный

Автоматически созданное описание

**ATmega48 diagram 1**

ALGORITHM

Изображение выглядит как диаграмма

Автоматически созданное описание

**Flowchart algorithm 1**

PROGRAM CODE

#define *F\_CPU* 8000000UL

#include <avr/io.h>

#include <util/delay.h>

#define UPPER\_BOUND 7000 // Setting max value for delay

#define LOWER\_BOUND 200 // Setting min value for delay

#define LED PORTB

#define ISPRESSED(n) (~PIND & ( 1 << n )) // Macro for checking whether button is pressed

// Macro for decreasing/increasing delay

#define CHANGE\_TIME if (ISPRESSED(5)) {decrease\_ms(&ms); continue;} if (ISPRESSED(1)) {increase\_ms(&ms); continue;}

// Function to delay for a given number of ms

void delay(*uint16\_t* ms){

while(ms--) *\_delay\_ms*(1); // Delay for 1ms at a time

}

// Function to increase the delay time by 1000ms

void increase\_ms(*uint16\_t*\* ms) {

while (ISPRESSED(1)); // Wait for SW1 to be released

if (\*ms + 1000 >= UPPER\_BOUND) // If the new delay time would be above the upper bound, set it to the upper bound

\*ms = UPPER\_BOUND; // Assign to 7000ms

else

\*ms += 1000; // Otherwise, increase the delay by 1000ms

}

// Function to decrease the delay time by 300ms

void decrease\_ms(*uint16\_t*\* ms) {

while (ISPRESSED(5)); // Wait for switch 5 to be released

if (\*ms <= LOWER\_BOUND+300) // if to avoid the overflow problem of uint16\_t type, in case of negative 'ms'

\*ms = LOWER\_BOUND; // Assign to 200ms

else if (\*ms - 300 <= LOWER\_BOUND) // If decreasing the delay time would be below the lower bound, set it to the lower bound

\*ms = LOWER\_BOUND; // Assign to 200ms

else

\*ms -= 300; // Otherwise, decrease the delay by 300ms

}

int main(void)

{

CLKPR = (1 << CLKPCE); // Enable write access CLKPR

CLKPR = 0; // Set the clock prescaler to divide by 1

DDRB = 0x40; // Set 6th LED as output

DDRD = 0x00; // Set PORTD as input

PORTD = 0x22; // Enable pull-up resistors for SW1 & SW5

*uint16\_t* ms = 500; // Set initial delay to 500ms

while (1) // Infinite loop

{

CHANGE\_TIME // Check for button presses during delay time

LED = 0x40; // Turn on the 6th LED

CHANGE\_TIME // Check for button presses during delay time

delay(ms); // Delay for the specified amount of time

CHANGE\_TIME // Check for button presses during delay time

LED = 0x00; // Turn off the 6th LED

CHANGE\_TIME // Check for button presses during delay time

delay(ms); // Delay for the specified amount of time

CHANGE\_TIME // Check for button presses during delay time

}

}

CONCLUSION

In conclusion, this task required an understanding of microcontroller programming and implementation of various features. The development of the microcontroller and I/O diagram was followed by the creation of an algorithm for the ATMega48 microcontroller based on the individual variant of the task. The program code was created in Atmel Studio 7 environment and debugged to ensure proper functioning. Time intervals were formed using custom delay functions, because *\_ms\_delay()* accepts only constant parameter.

The main difficulty was encountered when the button was held, but it was debounced using *while(ISPRESSED(n)).* There was also an error due to carelessness that caused overflow while decreasing delay for the *uint16\_t* type if the number was negative. Last but not least, the problem was connected setting clock frequency to 8 MHz, it makes microcontroller work slower, but it was fixed by enabling write access to Clock Prescaler Register and setting the prescaler to divide by 1.

Overall, this task gave practical experience in microcontroller programming and enhanced the understanding and skills in the field.