Technika Mikroprocesorowa

Sprawozdanie z Laboratorium 4

Maksym Pervov, grupa 4.7/13

1. Zadanie 1

   Disassembly:

```
--- C:\Users\Maksym\OneDrive\[illegible]\Techniki microprocesorowe\Lab4\Zadanie1\Zadanie1\Debug/.././main.c
    12: {
    13:     uint8_t x, sk;      //inicjalizacja zmiennych x i sk
    14:     uint8_t dt = 0;     //zmienna sluzy do obslugi nacisniec wiecej niz jeden przycisk
    15:     for (uint8_t i = 4; i < 8; i++)
00000036 34.e0                LDI R19,0x04      Load immediate
    14:     uint8_t dt = 0;     //zmienna sluzy do obslugi nacisniec wiecej niz jeden przycisk
00000037 60.e0                LDI R22,0x00      Load immediate
    15:     for (uint8_t i = 4; i < 8; i++)
00000038 4e.c0                RJMP PC+0x004F        Relative jump
--- C:\Users\Maksym\OneDrive\[illegible]\Techniki microprocesorowe\Lab4\Zadanie1\Zadanie1\Debug/.././main.c
    17:         PORTA= ~_BV(i); // wprowadz 0 na i-tej pozycji i 1 na innych pozycjach
00000039 81.e0                LDI R24,0x01      Load immediate
0000003A 90.e0                LDI R25,0x00      Load immediate
0000003B 03.2e                MOV R0,R19        Copy register
0000003C 02.c0                RJMP PC+0x0003        Relative jump
0000003D 88.0f                LSL R24       Logical Shift Left
0000003E 99.1f                ROL R25       Rotate Left Through Carry
0000003F 0a.94                DEC R0        Decrement
00000040 e2.f7                BRPL PC-0x03      Branch if plus
00000041 80.95                COM R24       One's complement
00000042 8b.bb                OUT 0x1B,R24      Out to I/O location
--- c:\program files (x86)\atmel\studio\7.0\toolchain\avr8\avr8-gnu-toolchain\avr\include\util/delay.h
   187:     __builtin_avr_delay_cycles(__ticks_dc);
00000043 89.ef                LDI R24,0xF9      Load immediate
00000044 90.e0                LDI R25,0x00      Load immediate
00000045 01.97                SBIW R24,0x01     Subtract immediate from word
00000046 f1.f7                BRNE PC-0x01      Branch if not equal
00000047 00.c0                RJMP PC+0x0001        Relative jump
00000048 00.00                NOP       No operation
--- C:\Users\Maksym\OneDrive\[illegible]\Techniki microprocesorowe\Lab4\Zadanie1\Zadanie1\Debug/.././main.c
    19:         sk = PINA;      //odczyt stanu klawiatury
00000049 89.b3                IN R24,0x19       In from I/O location
    20:         if(sk == 0xFF)  //sprawdzanie stanu klawiatury
0000004A 8f.3f                CPI R24,0xFF      Compare with immediate
0000004B 11.f4                BRNE PC+0x03      Branch if not equal
    22:             PORTB = 0xFF;   //jesli nacisnieto 2 i wiecej przyciskow niech wszystkie diody zaswieca
0000004C 88.bb                OUT 0x18,R24      Out to I/O location
--- C:\Users\Maksym\OneDrive\[illegible]\Techniki microprocesorowe\Lab4\Zadanie1\Zadanie1\Debug/.././main.c
    23:             break;
0000004D 08.95                RET       Subroutine return
--- c:\program files (x86)\atmel\studio\7.0\toolchain\avr8\avr8-gnu-toolchain\avr\include\util/delay.h
   187:     __builtin_avr_delay_cycles(__ticks_dc);
0000004E 89.ef                LDI R24,0xF9      Load immediate
0000004F 90.e0                LDI R25,0x00      Load immediate
00000050 01.97                SBIW R24,0x01     Subtract immediate from word
00000051 f1.f7                BRNE PC-0x01      Branch if not equal
00000052 00.c0                RJMP PC+0x0001        Relative jump
00000053 00.00                NOP       No operation
--- C:\Users\Maksym\OneDrive\[illegible]\Techniki microprocesorowe\Lab4\Zadanie1\Zadanie1\Debug/.././main.c
    26:         x=PINA&0x0F;    // odczyt i zapamietowanie stanu klawiatury
00000054 29.b3                IN R18,0x19       In from I/O location
00000055 2f.70                ANDI R18,0x0F     Logical AND with immediate
    27:         if (x == (PINA&0x0F))//petla if eliminujaca mozliwosc wystapienia drgania stykow
00000056 42.2f                MOV R20,R18       Copy register
00000057 50.e0                LDI R21,0x00      Load immediate
00000058 89.b3                IN R24,0x19       In from I/O location
00000059 8f.70                ANDI R24,0x0F     Logical AND with immediate
0000005A 90.e0                LDI R25,0x00      Load immediate
0000005B 48.17                CP R20,R24        Compare
0000005C 59.07                CPC R21,R25       Compare with carry
0000005D 41.f5                BRNE PC+0x29      Branch if not equal
```

```
    29:             switch(x)   //petla switch zalezna od zmiennej x, w ktorej znajduje sie odczytany stan klawiatury
0000005E 2d.30                  CPI R18,0x0D      Compare with immediate
0000005F 89.f0                  BREQ PC+0x12      Branch if equal
00000060 28.f4                  BRCC PC+0x06      Branch if carry cleared
00000061 27.30                  CPI R18,0x07      Compare with immediate
00000062 c1.f0                  BREQ PC+0x19      Branch if equal
00000063 2b.30                  CPI R18,0x0B      Compare with immediate
00000064 89.f0                  BREQ PC+0x12      Branch if equal
00000065 1a.c0                  RJMP PC+0x001B        Relative jump
00000066 2e.30                  CPI R18,0x0E      Compare with immediate
00000067 21.f0                  BREQ PC+0x05      Branch if equal
00000068 2f.30                  CPI R18,0x0F      Compare with immediate
00000069 b1.f4                  BRNE PC+0x17      Branch if not equal
    33:                     PORTB=0;        //zaden diod nie swieci
0000006A 18.ba                  OUT 0x18,R1       Out to I/O location
    34:                     break;
0000006B 16.c0                  RJMP PC+0x0017        Relative jump
    39:                     dt++;           //inkrementacja zmiennej dt
0000006C 6f.5f                  SUBI R22,0xFF     Subtract immediate
    40:                     PORTB = i-3;    //zaswiecenie numeru przyciska w postaci binarnej (od 1 do 4)
0000006D 8d.ef                  LDI R24,0xFD      Load immediate
0000006E 83.0f                  ADD R24,R19       Add without carry
0000006F 88.bb                  OUT 0x18,R24      Out to I/O location
    41:                     break;
00000070 11.c0                  RJMP PC+0x0012        Relative jump
    46:                     dt++;           //inkrementacja zmiennej dt
00000071 6f.5f                  SUBI R22,0xFF     Subtract immediate
    47:                     PORTB = i+1;    //zaswiecenie numeru przyciska w postaci binarnej (od 5 do 8)
00000072 81.e0                  LDI R24,0x01      Load immediate
00000073 83.0f                  ADD R24,R19       Add without carry
00000074 88.bb                  OUT 0x18,R24      Out to I/O location
    48:                     break;
```

```
00000075 0c.c0                  RJMP PC+0x000D        Relative jump
    53:                     dt++;           //inkrementacja zmiennej dt
00000076 6f.5f                  SUBI R22,0xFF     Subtract immediate
    54:                     PORTB = i+5;    //zaswiecenie numeru przyciska w postaci binarnej (od 9 do 12)
00000077 85.e0                  LDI R24,0x05      Load immediate
00000078 83.0f                  ADD R24,R19       Add without carry
00000079 88.bb                  OUT 0x18,R24      Out to I/O location
    55:                     break;
0000007A 07.c0                  RJMP PC+0x0008        Relative jump
    60:                     dt++;           //inkrementacja zmiennej dt
0000007B 6f.5f                  SUBI R22,0xFF     Subtract immediate
    61:                     PORTB = i+9;    //zaswiecenie numeru przyciska w postaci binarnej (od 13 do 16)
0000007C 89.e0                  LDI R24,0x09      Load immediate
0000007D 83.0f                  ADD R24,R19       Add without carry
0000007E 88.bb                  OUT 0x18,R24      Out to I/O location
--- C:\Users\Maksym\OneDrive\◆◆◆◆◆◆◆◆◆◆\Techniki microprocesorowe\Lab4\Zadanie1\Zadanie1\Debug/.././main.c
    62:                     break;
0000007F 02.c0                  RJMP PC+0x0003        Relative jump
    67:                     PORTB = 0xFF;   //zaswiecenie wszystkich diod
00000080 8f.ef                  SER R24       Set Register
00000081 88.bb                  OUT 0x18,R24      Out to I/O location
    71:             if (dt > 1)             // jesli podczas wykonania instrukcji case wartosc dt jest wiecej niz 1, znaczy to, ze wcisnieto dwa i wiecej przyciski
00000082 62.30                  CPI R22,0x02      Compare with immediate
00000083 10.f0                  BRCS PC+0x03      Branch if carry set
    73:                 PORTB = 0xFF;       //zaswiecenie wszystkich diod
00000084 8f.ef                  SER R24       Set Register
00000085 88.bb                  OUT 0x18,R24      Out to I/O location
    15:     for (uint8_t i = 4; i < 8; i++)
00000086 3f.5f                  SUBI R19,0xFF     Subtract immediate
--- No source file ---------------------------------------------------------------
```

```
--- C:\Users\Maksym\OneDrive\◆◆◆◆◆◆◆◆◆◆\Techniki microprocesorowe\Lab4\Zadanie1\Zadanie1\Debug/.././main.c
    80: {
    81:     DDRA = 0xF0;    //ustawienie 4 najmlodszych bitow na wejscia i 4 najstarszych na wyjscia
0000008B 80.ef                  LDI R24,0xF0      Load immediate
0000008C 8a.bb                  OUT 0x1A,R24      Out to I/O location
    82:     PORTA = 0x0F;   //ustawienie 4 najmlodszych bitow na wejscia podciagniete i 4 najstarszych na 0
0000008D 8f.e0                  LDI R24,0x0F      Load immediate
0000008E 8b.bb                  OUT 0x1B,R24      Out to I/O location
    84:     DDRB = 0xFF;    //ustawianie kierunku danych dla diod na wyjscie
0000008F 8f.ef                  SER R24       Set Register
00000090 87.bb                  OUT 0x17,R24      Out to I/O location
    85:     PORTB = 0x00;   //ustawianie wszystkich diod na 0
00000091 18.ba                  OUT 0x18,R1       Out to I/O location
    89:         getkey();   // wywolanie funkcji getkey()
00000092 0e.94.36.00            CALL 0x00000036       Call subroutine
00000094 fd.cf                  RJMP PC-0x0002        Relative jump
```

Source code:

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>

void getkey()
{
	uint8_t x, sk;			//inicjalizacja zmiennych x i sk
	uint8_t dt = 0;			//zmienna sluzy do obslugi nacisniec wiecej niz jeden
					przycisk
```

```
for (uint8_t i = 4; i < 8; i++)
{
	PORTA= ~_BV(i);		// wprowadz 0 na i-tej pozycji i 1 na innych pozycjach
	_delay_ms(1);		//opoznienie co 1 ms
	sk = PINA;		//odczyt stanu klawiatury
	if(sk == 0xFF)		//sprawdzanie stanu klawiatury
	{
		PORTB = 0xFF; //jesli nacisnieto 2 i wiecej przyciskow niech
				wszystkie diody zaswieca
		break;
	}
	_delay_ms(1);		//opoznienie co 1 ms
	x=PINA&0x0F;		// odczyt i zapamietowanie stanu klawiatury
	if (x == (PINA&0x0F))//petla if eliminujaca mozliwosc wystapienia drgania
				stykow
	{
		switch(x)	//petla switch zalezna od zmiennej x, w ktorej znajduje
				sie odczytany stan klawiatury
		{
			case 0b00001111:	//jesli zaden przycisk nie jest wcisniety
			{
				PORTB=0;		//zaden diod nie swieci
				break;
			}

			case 0b00001110:	//jesli wcisnieto przycisk z wiersza 1
			{
				dt++;			//inkrementacja zmiennej dt
				PORTB = i-3;	//zaswiecenie numeru przyciska w postaci
						binarnej (od 1 do 4)
				break;
			}

			case 0b00001101:	//jesli wcisnieto przycisk z wiersza 2
			{
				dt++;			//inkrementacja zmiennej dt
				PORTB = i+1;	//zaswiecenie numeru przyciska w postaci
						binarnej (od 5 do 8)
				break;
			}

			case 0b00001011:	//jesli wcisnieto przycisk z wiersza 3
			{
				dt++;			//inkrementacja zmiennej dt
				PORTB = i+5;	//zaswiecenie numeru przyciska w postaci
						binarnej (od 9 do 12)
				break;
			}

			case 0b00000111:	//jesli wcisnieto przycisk z wiersza 4
			{
				dt++;			//inkrementacja zmiennej dt
				PORTB = i+9;	//zaswiecenie numeru przyciska w postaci
						binarnej (od 13 do 16)
				break;
			}

			default:		//jesli zadna instrukcja nie byla
						wykonana, znaczy to, ze wcisnieto dwa i
						wiecej przyciski
			{
				PORTB = 0xFF; //zaswiecenie wszystkich diod
				break;
			}
		}
```

```
                    if (dt > 1)                 // jesli podczas wykonania instrukcji case
                                                wartosc dt jest wiecej niz 1, znaczy to,
                                                ze wcisnieto dwa i wiecej przyciski
                    {
                            PORTB = 0xFF;       //zaswiecenie wszystkich diod
                    }
            }
        }
}


int main(void)

{
        DDRA = 0xF0;            //ustawienie 4 najmlodszych bitow na wejscia i 4 najstarszych
                                na wyjscia
        PORTA = 0x0F;           //ustawienie 4 najmlodszych bitow na wejscia podciagniete i 4
                                najstarszych na 0

        DDRB = 0xFF;            //ustawianie kierunku danych dla diod na wyjscie
        PORTB = 0x00;           //ustawianie wszystkich diod na 0

    while (1)
    {
            getkey();       // wywolanie funkcji getkey()
    }
}
```

2. Zadanie 2 (wykonane częściowo)

Source code:

```
#define F_CPU 1000000L
#include <avr/io.h>
#include <util/delay.h>
#include <stdlib.h>

int operacja()
{
	uint8_t x, sk;
	uint8_t dt = 0;
	for (uint8_t i = 4; i < 8; i++)
	{
		PORTA= ~_BV(i);				// wprowadzanie 0 na i-tej pozycji
i 1 na innych pozycjach
		_delay_ms(1);
		sk = PINA;
		if(sk == 0xFF)
		{
			PORTB = 0xFF;
			break;
		}
		_delay_ms(1);
		x=PINA&0x0F;			// odczytanie i zapamietywanie stanu
klawiatury
		if (x == (PINA&0x0F))
		{
			switch(x)
			{
				case 0b00001111:	//zaden przycisk nie jest wcisniety
				{
					PORTB=0;
					break;
				}

				case 0b00001110:	//wybrano dodawanie
				{
					dt++;			//inkrementacja zmiennej dt
					break;
				}

				case 0b00001101:	//wybrano odejmowanie
				{
					dt++;			//inkrementacja zmiennej dt
					break;
				}

				case 0b00001011:	//wybrano mnozenie
				{
					dt++;			//inkrementacja zmiennej dt
					break;
				}

				case 0b00000111:	//wybrano dzielenie
				{
					dt++;			//inkrementacja zmiennej dt
					break;
				}

				default:			//jesli zadna instrukcja nie byla
wykonana, znaczy to, ze wcisnieto dwa i wiecej przyciski
				{
```

```
                                        PORTB = 0xFF; //zaswiecenie wszystkich diod
                                        break;
                                }
                        }
                        if (dt > 1)                         // jesli podczas wykonania
    instrukcji case wartosc dt jest wiecej niz 1, znaczy to, ze wcisnieto dwa i wiecej
    przyciski
                        {
                                PORTB = 0xFF;       //zaswiecenie wszystkich diod
                        }
                }
        }
    }

    int main(void)
    {
            DDRA = 0xF0;  //ustawienie 4 najmlodszych bitow na wejscia i 4 najstarszych na
    wyjscia
            PORTA = 0x0F; //ustawienie 4 najmlodszych bitow na wejscia podciagniete i 4
    najstarszych na 0

            DDRB = 0xFF;  //ustawianie kierunku danych dla diod na wyjscie
            PORTB = 0x00; //ustawianie wszystkich diod na 0

            DDRD=0x0F;          //konfiguracja portu dla kolumn wyswietlacza

        while (1)
        {
                operacja();
        }
}
```