|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| logowydzialu | Instytut Informatyki Politechniki Śląskiej  Zespół Mikroinformatyki i Teorii Automatów Cyfrowych  **Laboratorium SMiW** | | logoii | |
| **Rok akademicki** | **Rodzaj studiów\*: SSI/NSI/NSM** | **Numer ćwiczenia:** | **Grupa** | **Sekcja** |
| **2018/2019** | **SSI** | **7 i 9** | **5** | **9** |
| **Data i godzina planowana ćwiczenia:**  dd/mm/rrrr - gg:mm | **2018-12-10 – 13:15-14:45** | **Prowadzący**:  OA/JP/KT/GD/BSz/GB | **JP** | |
| **Data i godzina wykonania ćwiczenia:**  dd/mm/rrrr - gg:mm | **2018-12-10 – 13:15-14:45** |
| ***Sprawozdanie*** | | | | |
| **Temat ćwiczenia:**  Mikrokontrolery z serii AVR cz. 1 i 2 | | | | |
| **Skład sekcji:** | 1. Drabińska Martyna  2. Krasoń Bartłomiej  3. Miciak Michał | | | |

# 1.:Opis zadania

Na kontrolerze AVR ATmega2560 zrealizować program realizujący następujące założenie:

Podanie sygnału SOS na diodę LED, połączonej do PORTB7, z przerwą 0.3s.

Zadanie zrealizować w języku asemblera.

# 2.IKod programu w języku assemblerowym

//////////////////////////////////////////////////////////////////////////////

// Laboratory AVR Microcontrollers Part2

// Program template for lab 9

// Please fill in this information before starting coding

// Authors:

// Martyna Drabinska

// Michal Miciak

// Bartlomiej Krason

// Group: 5

// Section: 9

//

// Task: E

//

// Todo:

// sos signal

//

// Version: 5.0

//////////////////////////////////////////////////////////////////////////////

.nolist;quartz assumption 4Mhz

.include"m2560def.inc"

;//////////////////////////////////////////////////////////////////////////////

.list

.equ xlength =100

;//////////////////////////////////////////////////////////////////////////////

; EEPROM - data non volatile memory segment

.ESEG

;//////////////////////////////////////////////////////////////////////////////

; StaticRAM - data memory.segment

.DSEG

.ORG0x200; may be omitted this is default value

; Destination table (xlengthx bytes).

; Replace "xlengthx" with correct value

TAB\_RAM:.BYTE xlength

;//////////////////////////////////////////////////////////////////////////////

; CODE - Program memory segment

; Please Remember that it is "word" address space

;

.CSEG

.org0x0000; may be omitted this is default value

jmpRESET; Reset Handler

; Interrupts vector table / change to your procedure only when needed

jmpEXT\_INT0; IRQ0 Handler

jmpEXT\_INT1; IRQ1 Handler

jmpEXT\_INT2; IRQ2 Handler

jmpEXT\_INT3; IRQ3 Handler

jmpEXT\_INT4; IRQ4 Handler

jmpEXT\_INT5; IRQ5 Handler

jmpEXT\_INT6; IRQ6 Handler

jmpEXT\_INT7; IRQ7 Handler

jmpHPCINT0; PCINT0 Handler

jmpHPCINT1; PCINT1 Handler

jmpHPCINT2; PCINT2 Handler

jmpWDT; WDT Handler

jmpTIM2\_COMPA; Timer2 CompareA Handler

jmpTIM2\_COMPB; Timer2 CompareB Handler

jmpTIM2\_OVF; Timer2 Overflow Handler

jmpTIM1\_CAPT; Timer1 Capture Handler

jmpTIM1\_C0MPA; Timer1 CompareA Handler

jmpTIM1\_C0MPB; Timer1 CompareB Handler

jmpTIM1\_COMPC; Timer1 CompareC Handler

jmpTIM1\_0VF; Timer1 Overflow Handler

jmpTIM0\_COMPA; Timer0 CompareA Handler

jmpTIM0\_COMPB; Timer0 CompareB Handler,

jmpTIM0\_OVF; Timer0 Overflow Handler

jmpSPI\_STC; SPI Transfer Complete Handler

jmpUSART0\_RXC; USART0 RX Complete Handler

jmpUSART0\_UDRE; USART0,UDR Empty Handler

jmpUSART0\_TXC; USART0 TX Complete Handler

jmpANA\_COMP; Analog COmparator Handler

jmpHADC; ADC Conversion Complete Handler

jmpEE\_RDY; EEPROM Ready Handler

jmpTIM3\_CAPT; Timer3 Capture Handler

jmpTIM3\_COMPA; Timer3 CompareA Handler

jmpTIM3\_COMPB; Timer3 CompareB Handler

jmpTIM3\_COMPC; Timer3 CompareC Handler

jmpTIM3\_OVF; Timer3 Overflow Handler

jmpUSART1\_RXC; USART1 RX Complete Handler

jmpUSART1\_UDRE; USART1,UDR Empty Handler

jmpUSART1\_TXC; USART1 TX Complete Handler

jmpTWI; Two-wire Serial Interface Interrupt Handler

jmpSPM\_RDY; SPM Ready Handler

jmpTIM4\_CAPT; Timer4 Capture Handler

jmpTIM4\_COMPA; Timer4 CompareA Handler

jmpTIM4\_COMPB; Timer4 CompareB Handler

jmpTIM4\_COMPC; Timer4 CompareC Handler

jmpTIM4\_OVF; Timer4 Overlflow Handler

jmpTIM5\_CAPT; Timer5 Capture Handler

jmpTIM5\_COMPA; Timer5 CompareA Handler

jmpTIM5\_COMPB; Timer5 CompareB Handler

jmpTIM5\_COMPC; Timer5 CompareC Handler

jmpTIM5\_OVF; Timer5 Overlflow Handler

jmpUSART2\_RXC; USART2 RX Complete Handler

jmpUSART2\_UDRE; USART2,UDR Empty Handler

jmpUSART2\_TXC; USART2 TX Complete Handler

jmpUSART3\_RXC; USART3 RX Complete Handler

jmpUSART3\_UDRE; USART3,UDR Empty Handler

jmpUSART3\_TXC; USART3 TX Complete Handler

//////////////////////////////////////////////////////////////////////////////

EXT\_INT0:; IRQ0 Handler

EXT\_INT1:; IRQ1 Handler

EXT\_INT2:; IRQ2 Handler

EXT\_INT3:; IRQ3 Handler

EXT\_INT4:; IRQ4 Handler

EXT\_INT5:; IRQ5 Handler

EXT\_INT6:; IRQ6 Handler

EXT\_INT7:; IRQ7 Handler

HPCINT0:; PCINT0 Handler

HPCINT1:; PCINT1 Handler

HPCINT2:; PCINT2 Handler

WDT:; WDT Handler

TIM2\_COMPA:; Timer2 CompareA Handler

TIM2\_COMPB:; Timer2 CompareB Handler

TIM2\_OVF:; Timer2 Overflow Handler

TIM1\_CAPT:; Timer1 Capture Handler

TIM1\_C0MPA:; Timer1 CompareA Handler

TIM1\_C0MPB:; Timer1 CompareB Handler

TIM1\_COMPC:; Timer1 CompareC Handler

TIM1\_0VF:; Timer1 Overflow Handler

subir25,0x01 ;odjecie 1 od licznika

cpir25,0x00; sprawdzenie czy koniec tablicy

breqTABLE\_S; jeżeli koniec tablicy ustaw wskaźnik na nowo

elpmR16,Z+ ; pobranie kolejnej wartości tablicy do r16

out PORTB,r16 ; przekazanie wartości na port diody

rjmpCOUNTER\_SET ;skok do ponownego ustawienia licznika

TIM0\_COMPA:; Timer0 CompareA Handler

TIM0\_COMPB:; Timer0 CompareB Handler

TIM0\_OVF:; Timer0 Overflow Handler

SPI\_STC:; SPI Transfer Complete Handler

USART0\_RXC:; USART0 RX Complete Handler

USART0\_UDRE:; USART0,UDR Empty Handler

USART0\_TXC:; USART0 TX Complete Handler

ANA\_COMP:; Analog COmparator Handler

HADC:; ADC Conversion Complete Handler

EE\_RDY:; EEPROM Ready Handler

TIM3\_CAPT:; Timer3 Capture Handler

TIM3\_COMPA:; Timer3 CompareA Handler

TIM3\_COMPB:; Timer3 CompareB Handler

TIM3\_COMPC:; Timer3 CompareC Handler

TIM3\_OVF:; Timer3 Overflow Handler

USART1\_RXC:; USART1 RX Complete Handler

USART1\_UDRE:; USART1,UDR Empty Handler

USART1\_TXC:; USART1 TX Complete Handler

TWI:; Two-wire Serial Interface Interrupt Handler

SPM\_RDY:; SPM Ready Handler

TIM4\_CAPT:; Timer4 Capture Handler

TIM4\_COMPA:; Timer4 CompareA Handler

TIM4\_COMPB:; Timer4 CompareB Handler

TIM4\_COMPC:; Timer4 CompareC Handler

TIM4\_OVF:; Timer4 Overlflow Handler

TIM5\_CAPT:; Timer5 Capture Handler

TIM5\_COMPA:; Timer5 CompareA Handler

TIM5\_COMPB:; Timer5 CompareB Handler

TIM5\_COMPC:; Timer5 CompareC Handler

TIM5\_OVF:; Timer5 Overlflow Handler

USART2\_RXC:; USART2 RX Complete Handler

USART2\_UDRE:; USART2,UDR Empty Handler

USART2\_TXC:; USART2 TX Complete Handler

USART3\_RXC:; USART3 RX Comlplete Handler

USART3\_UDRE:; USART3,UDR Empty Handler

USART3\_TXC:; USART3 TX Complete Handler

reti; return from all no used

;//////////////////////////////////////////////////////////////////////////////

; Program start

RESET:

ldir16,0xFF

out DDRB,r16 ;ustawienie portu B jako output

;ustawienie tablicy

    TABLE\_S:

ldir25,0x1C ;28 – rozmiar tablicy do r25

ldizl,low(TAB\_ROM\*2) ; ładujemy adres tablicy do Z

ldizh,high(TAB\_ROM\*2)

ldir16,byte3(TAB\_ROM\*2)

out RAMPZ,r16

;ustawienie licznika

    COUNTER\_SET:

cli ;wyłączenie przerwań

LDIR20,(1<< TOIE1)

STS TIMSK1,R20

;// prescaler na 1024

ldir20,0x05

STS TCCR1B,r20

;//ustawienie wartości początkowej licznika

ldir20,0xfb

sts TCNT1H,r20

ldir20,0x61

sts TCNT1L,r20

sei

loop:

rjmploop

//------------------------------------------------------------------------------

// Program end - Ending loop

//------------------------------------------------------------------------------

End:

rjmpEND

//------------------------------------------------------------------------------

// Table Declaration - place here test values

// Test with different table values and different begin addresses of table (also above 0x8000)

//

//.org0x8000

TAB\_ROM:.db0x80,0x00,0x80,0x00,0x80,0x00,0x00,0x80,0x80,0x80,0x00,0x80,0x80,0x80,0x00,0x80

.db0x80,0x80,0x00,0x00,0x80,0x00,0x80,0x00,0x80,0x00,0x00,0x00

.EXIT

//------------------------------------------------------------------------------

# 3.IPodsumowanie

Aby rozwiązać zadanie musieliśmy skorzystać z przerwania generowanego przez Timer1 układu:

TIM1\_0VF:; Timer1 Overflow Handler

Dodatkowo w programie musieliśmy odpowiednio ustawić częstotliwość generowania przerwania, aby spełniała ona założenia zadania.

Generowanie sygnału SOS (miganie diody) zrealizowaliśmy wpisując w odpowiedniej kolejności wartości sygnału do tablicy TAB\_ROM, z której kolejne wartości następnie przekazywaliśmy na port B.7, tak aby dioda imitowała nadawanie sygnału SOS.

Realizację zadania rozpoczęliśmy od pisania programu w środowisku Atmel Studio 7 oraz przetestowaniu go w symulatorze środowiska. Po ukończeniu programu, wgraliśmy go na kontroler za pomocą środowiska. Program działał prawidłowo zarówno w symulatorze jak i na kontrolerze.