**Listing 1. Kod programu przypominacza**

//Miniaturowy przypominacz AVT

//Michał Kurzela @ 2018

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#include <avr/sleep.h>

//definicja długości odmierzanych czasów [w minutach]

#define CZAS\_1 15

#define CZAS\_2 30

#define CZAS\_3 60

#define CZAS\_4 120

//progi ADC do wykrycia zworek

#define PROG\_4 700 //nominalnie 512

#define PROG\_3 380 //nominalnie 327

#define PROG\_2 250 //nominalnie 185

#define PROG\_1 130 //nominalnie 93

#define LADDER\_ON PORTB|=(1<<PORTB1)

#define LADDER\_OFF PORTB&=~(1<<PORTB1)

#define BUZZER\_ON PORTB|=(1<<PORTB3)

#define BUZZER\_OFF PORTB&=~(1<<PORTB3)

#define LED\_ON PORTB|=(1<<PORTB4)

#define LED\_OFF PORTB&=~(1<<PORTB4)

#define LICZBA\_BLYSKOW 4 //liczba błysków w jednym cyklu

#define PRZERWA\_KROTKA 50 //długość krótkiej przerwy w generowaniu błysków

#define PRZERWA\_DLUGA 250 //długość przerw między seriami błysków

volatile uint8\_t sekundy **=** 0**;** //odliczone sekundy

volatile uint8\_t minuty **=** 0**;** //odliczone minuty

volatile uint8\_t czy\_odlicza **=** 0**;** //0 - pozostaje uśpiony, 1 - odlicza, 2 - wszczyna alarm

volatile uint8\_t czas\_zadany**;** //zadany do odliczenia czas w minutach

// Przerwanie od przetwornika A/C

ISR**(**ADC\_vect**)**

**{**

**if((**ADCL **|** **(**ADCH**<<**8**))** **<=** PROG\_1**){** czas\_zadany **=** CZAS\_1**;** **}**

**if((**ADCL **|** **(**ADCH**<<**8**))** **>** PROG\_1 **&&** **(**ADCL **|** **(**ADCH**<<**8**))** **<=** PROG\_2**){** czas\_zadany **=** CZAS\_2**;** **}**

**if((**ADCL **|** **(**ADCH**<<**8**))** **>** PROG\_2 **&&** **(**ADCL **|** **(**ADCH**<<**8**))** **<=** PROG\_3**){** czas\_zadany **=** CZAS\_3**;** **}**

**if((**ADCL **|** **(**ADCH**<<**8**))** **>** PROG\_3 **&&** **(**ADCL **|** **(**ADCH**<<**8**))** **<=** PROG\_4**){** czas\_zadany **=** CZAS\_4**;** **}**

**if((**ADCL **|** **(**ADCH**<<**8**))** **>** PROG\_4**){** czas\_zadany **=** 0**;** czy\_odlicza **=** 0**;}** //przejdź w stan uśpienia, jeżeli nie ma załączonego DIP switcha

ADCSRA **=** 0**;** LADDER\_OFF**;** //wyłącz ADC i drabinkę

**if(**czy\_odlicza**)**

**{** //jeżeli odliczanie nie zostało zatrzymane

sekundy **=** 0**;** minuty **=** 0**;** //przygotowanie do odliczania

WDTCR **=** **(**1 **<<** WDTIE**)** **|** **(**1 **<<** WDP1**)** **|** **(**1 **<<** WDP2**);** //uruchomienie przerwań od watchdoga + timeout 1s

**}**

LED\_OFF**;** //wyłącz diody LED - użytkownik otrzyma mignięcie

**}**

// Przerwanie od przycisku SW1

ISR**(**PCINT0\_vect**)**

**{**

**if(**bit\_is\_clear**(**PINB**,**PINB0**))**

**{**

**if(**czy\_odlicza **==** 0**)**

**{** //jeżeli nie odliczał

LADDER\_ON**;** //załączenie drabinki rezystorowej

LED\_ON**;** //załączenie diod LED dla uzyskania mignięcia potwierdzającego start

czy\_odlicza **=** 1**;** //trwa odliczanie

ADMUX **=** **(**1 **<<** MUX0**);** //wejście ADC1

ADCSRA **=** **(**1 **<<** ADEN**)** **|** **(**1 **<<** ADATE**)** **|** **(**1 **<<** ADIE**)** **|** **(**1 **<<** ADPS2**);** //uruchomienie ADC, wielokrotne powtarzanie, uruchomienie przerwania, preskaler przez 16

ADCSRA **|=** **(**1 **<<** ADSC**);** //rozpoczęcie konwersji

\_delay\_ms**(**200**);** //aby wyeliminować drganie styków

**}**

**if(**czy\_odlicza **==** 2**)**

**{** //jeżeli został wszczęty alarm po odliczeniu czasu

czy\_odlicza **=** 0**;** //przejście w stan uśpienia

WDTCR **=** 0**;** //watchdog jest już niepotrzebny

LED\_OFF**;** BUZZER\_OFF**;** //wyłącz diody i buzzer

\_delay\_ms**(**1000**);** //aby wyeliminować drganie styków

**}**

**}**

**}**

// Przerwanie od Watchdog

**/**SR**(**WDT\_vect**)**

**{**

**if(**czy\_odlicza **==** 1**)**

**{**

sekundy**++;**

LED\_ON**;**

\_delay\_ms**(**2**);**

LED\_OFF**;**

**}**

**if(**sekundy **>=** 60**){**minuty**++;** sekundy **=** 0**;** **}**

**if(**minuty **>=** czas\_zadany**){** czy\_odlicza **=** 2**;}** //rozpocznij alarm

**}**

// GLÓWNA FUNKCJA PROGRAMU

int main**(**void**)**

**{**

//konfiguracja wyprowadzeń

DDRB **=** **(**1 **<<** DDB4**)** **|** **(**1 **<<** DDB3**)** **|** **(**1 **<<** DDB1**);** //wyjścia LED, BUZZER i LADDER\_ON

PORTB **=** **(**1 **<<** PORTB0**);** //wejście SWITCH w stanie wysokim

//uruchomienie przerwań od przycisku

GIMSK **=** **(**1 **<<** PCIE**);** //aktywacja systemu przerwań PCINT

PCMSK **=** **(**1 **<<** PCINT0**);** //przerwanie od pinu PINB0

sei**();** //uruchomienie systemu przerwań

**while(**1**)**

**{**

**while(**czy\_odlicza **==** 2**){** //w stanie alarmu

**for(**uint8\_t i **=** 0 **;** i **<** LICZBA\_BLYSKOW **;** i**++)**

**{**

LED\_ON**;** BUZZER\_ON**;**

\_delay\_ms**(**PRZERWA\_KROTKA**);**

LED\_OFF**;** BUZZER\_OFF**;**

\_delay\_ms**(**PRZERWA\_KROTKA**);**

**}**

\_delay\_ms**(**PRZERWA\_DLUGA**);**

**}**

set\_sleep\_mode**(**SLEEP\_MODE\_PWR\_DOWN**);**

sleep\_enable**();**

sleep\_cpu**();**

**}**

sleep\_disable**();**

**}**