```
/*****************************************
 * Comunicador hospitalar_C_ADC_INT_PWM.c
 *
 * Created: 12/11/2019 08:26:31
 * Author : Ana Watanabe
 *****************************************/

#define F_CPU 16000000UL
#include <avr/io.h> //definições do componente especificado
#include <avr/interrupt.h>

#define set_bit(Y,bit_x) (Y|=(1<<bit_x))   /*ativa o bit */
#define clr_bit(Y,bit_x) (Y&=~(1<<bit_x))  /*limpa o bit */
#define cpl_bit(Y,bit_x) (Y^=(1<<bit_x))   /*troca o bit */
#define tst_bit(Y,bit_x) (Y&(1<<bit_x))    /*testa o bit */

// protótipos

ISR(INT0_vect); // Protótipo da Int. ext INT0. Tabela de vetores pag. 158
ISR(TIMER1_OVF_vect); // Protótipo da Interrupção TIMER1. (16 bits)

char tempo = 0;  // tempo de intermintência do buzzer
char flag_tempo = 0;     // permite contar tempo
char acionador = 0;      // controle do paciente apertar acionador
char segundo = 0;
char duty = 0;
char passou = 0;

int main(void)
{
     // configuração de E/S digitais
     DDRB = 0xff; // pinos PB0 à 7: saidas (leds)
     DDRD = 0x88; // pinos: PD3(buzzer de PWM) e PD7(buzzer): saídas,
                  // PD2(acionador) e PD5(seleção de tempo): entradas
     PORTD = 0x24; // pull up: PD2 e PD5.

     //Desliga buzzer
     clr_bit(PORTD, PD7);  //DESLIGA O BUZZER INTERMITENTE
     // Desliga todos leds
     PORTB = 0x00; // Apaga todos leds

     // Configuração do ADC
     DIDR0 = 0x10;//  entrada analógica no PC4
     ADCSRA = 0x87; // ADC habilitado, prescaler = 128
     ADMUX = 0x44; // Tensão AVCC, alinhado a direita (10 bits), canal 4

     // configuração da interrupção INT0 (PD2) pg.32
     EICRA = 0b00000010;   // borda de descida em INT0 pg. 163
     EIMSK = 0x01;   // Ativa INT0.   pg. 164

   // configuração da interrupção de timer no TC1(16 bits)
    TCCR1A = 0; // timer para oper.normal OC1A e OC1B desconect.
    TCCR1B = 0; //limpa registrador
    TCCR1B |= (1<<CS10)|(1 << CS12); // configura prescaler 1024
    TCNT1 = 0xC2F7;  //valor para que estouro ocorra em 1 segundo
     // 65536-(16MHz/1024/1Hz) =  65536 - 15.625 = 49911(0xC2F7)
    TIMSK1 |= (1 << TOIE1);
```

```
// configuração do PWM no PD3 => OCR2B   // pg. 32
    TCCR2A = 0b00100001; //PWM com fase corrigida, saida OC2B não invertida,
modo 1 e TOP = 0xff   pg.203
// 488,28Hz = 16000000/2 prescaler (255) => prescaler = 64
    TCCR2B = 0x04; //PWM fase corrigida,  modo 1 e prescaler =  64

// desliga buzzer (PWM)
   OCR2B = 0;   // duty cycle = 0

    sei(); // Liga a chave geral de interrupções.

// acende um led
     leds = 0x01;    //acende o led do PB0

     while(1) //laço infinito
      {
       // verifica temperatura
       set_bit(ADCSRA, ADSC);
       while (tst_bit(ADCSRA, ADSC))  // aguarda leitura
           ;
       if(ADC >= 641) // Se temperatura maior que 38 graus
           duty = 178;   // duty cycle = 70%
       if(ADC <= 589) // Se temperatura menor  que 35 graus
           duty = 76;   // duty cycle = 30%
       if(ADC > 589 && ADC < 641)   // temperature normal
           duty = 0;    // duty cycle = 0% sem buzzer
       OCR2B = duty;  // atualiza o registrador do OC2B

       if(!tst_bit(PIND, PD5))    // leitura da chave PD5
              tempo = 1;//  PD5 = 0, então tempo =1
       else
              tempo = 3; // PD5 = 1, então tempo =3

       if(acionador ==1)    //se foi apertado uma vez
           {
           flag_tempo = 1;  //  considera contagem de  tempo
           PORTB = leds;
           if (segundo >= tempo)
              {
              leds <<=1;
              if(leds == 0)
                leds = 0x01;
              segundo = 0;   //inicializa a contagem
              }
           flag_tempo = 0;  //  considera sem contagem de  tempo
           }
       // led escolhido fica aceso e toca a buzina a cada 1s
       if(acionador ==2)
         {
         flag_tempo = 1;  //  considera contagem de  tempo
          PORTB = leds;    // acende só o led escolhido
         if (passou ==1)
                {
                cpl_bit(PORTD,PD7);  // liga ou desliga o buzzer
                passou = 0;          // reseta o flag
                }
```

```
          flag_tempo = 0;  //  considera sem contagem de  tempo
          }
        }  //fim do while
    }     // fim do main


//*********************************************
//   Rotina de tratamento de Interrupção Externa
acionador:
    0   - inicio
    1   - piscando
    2   - escolhido
//*********************************************

ISR(INT1_vect){
acionador++;
if (acionador ==3)
   acionador =1;
}

//*****************************************************
//   Rotina de tratamento de Timer (1s)- clock interno
//*****************************************************

ISR(TIMER1_OVF_vect){
// verifica flag_tempo, se igual a zero, sai da rotina
if(flag_tempo !=0)
   {
   TCNT1=49911;/*Inicializa o timer */
   passou = 1;
   segundo++;
   }
}
```