Índice

1. Objetivos

Se diseñará e implementará una pulsera térmica que regulará la temperatura corporal. Se utilizará un módulo termoeléctrico para enviar variaciones de calor o frío a la muñeca del usuario para modificar la percepción térmica del cuerpo.

Su función es generar pulsos de frío o calor, de manera de generar una sensación de confort para una persona en condiciones donde la temperatura es muy alta o muy baja respectivamente. Está basado en el proyecto *Wristify* [1] ganador del concurso de intel *Make It Wearable* [2].

2. Especificaciones

El dispositivo utilizará una celda Peltier para enviar pulsos de calor o frío. De forma que se logre una diferencia de temperatura mayor a $0,4\,^{o}C$/seg. durante 5 segundos y durante los siguientes 10 segundos entrará en estado de espera, para luego volver a iniciar el ciclo.

Deberá contar con un sensor de temperatura para medir la temperatura ambiente y analizar si deberá enviar o recibir calor.

Finalmente deberá controlar que se cumpla el ciclo en base a la corriente que circulará por la celda Peltier.

2.1. Componentes

Deberá contar con los siguientes componentes:

- Celda Peltier: Generará los pulsos de calor en la muñeca del usuario.
- Circuito regulador de corriente: Regulará la corriente suministrada a la celda peltier.
- Disipador: La celda Peltier contará con un disipador para evitar fijar la temperatura de una de sus placas.
- Termistores: Contará con dos termistores. Uno para medir la temperatura ambiente y en base a esta decidir el modo de trabajo, frío o calor. El segundo termistor medirá la temperatura del disipador conectado a la celda Peltier para poder realizar una estimación de la temperatura de la celda.
- Salida de puerto serie: Servirá para poder monitorear en una computadora la temperatura de la placa.
- Fuente: Suministrará la corriente necesaria a la celda Peltier y proporcionará alimentación a todos los dispositivos utilizados.
- Interruptor: Para poder invertir el estado de trabajo, de frío a calor y viceversa.
- Controlador: Se utilizara un microcontrolador AVR. Es el encargado de obtener las temperaturas de los termistores para definir el modo de trabajo y autoregular la corriente de la celda Peltier mediante el circuito regulador de corriente.

2.2. Diagrama de Flujo

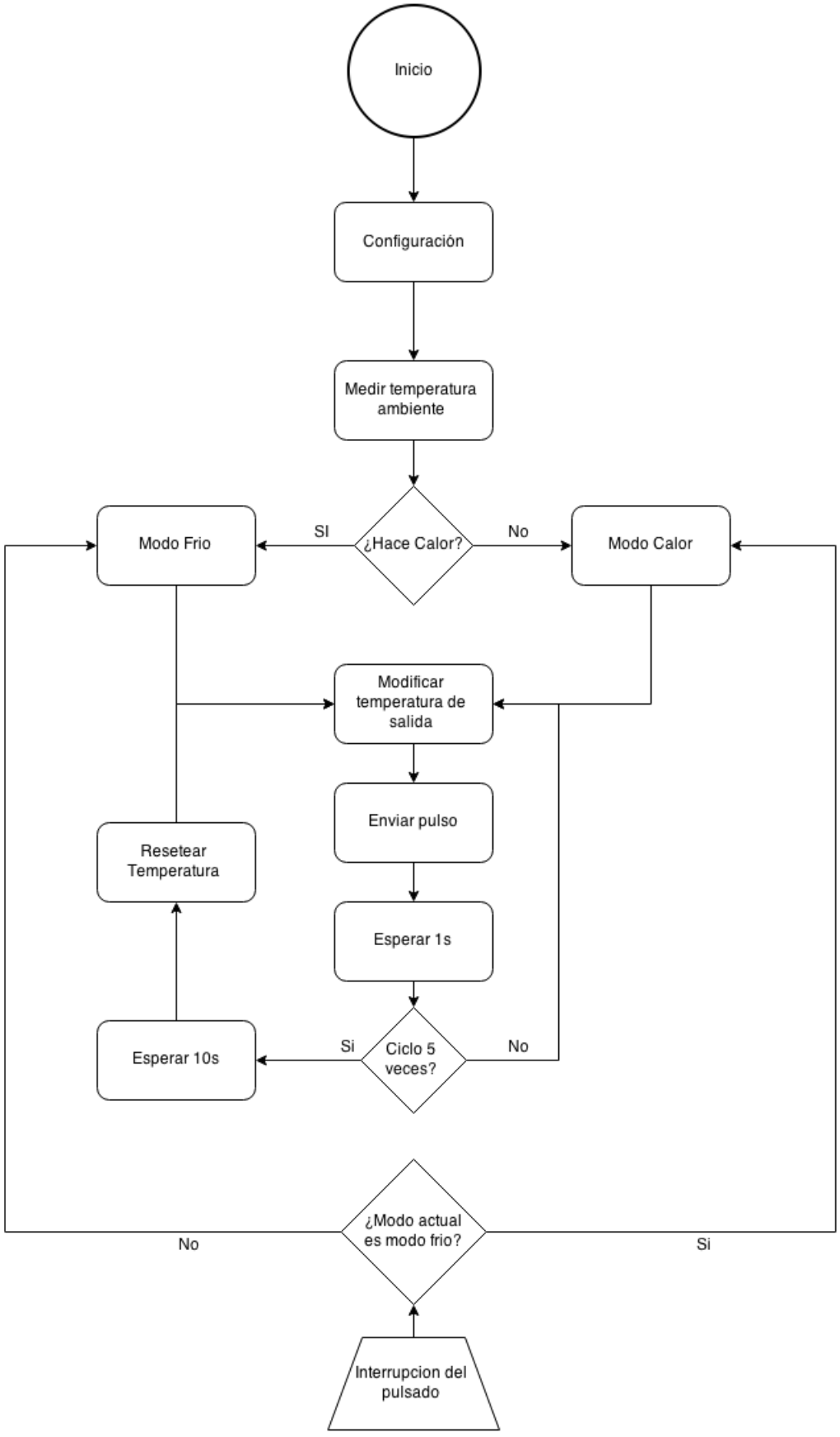

Figura 1: Diagrama de flujo del proceso

2.3. Diagrama de Bloques

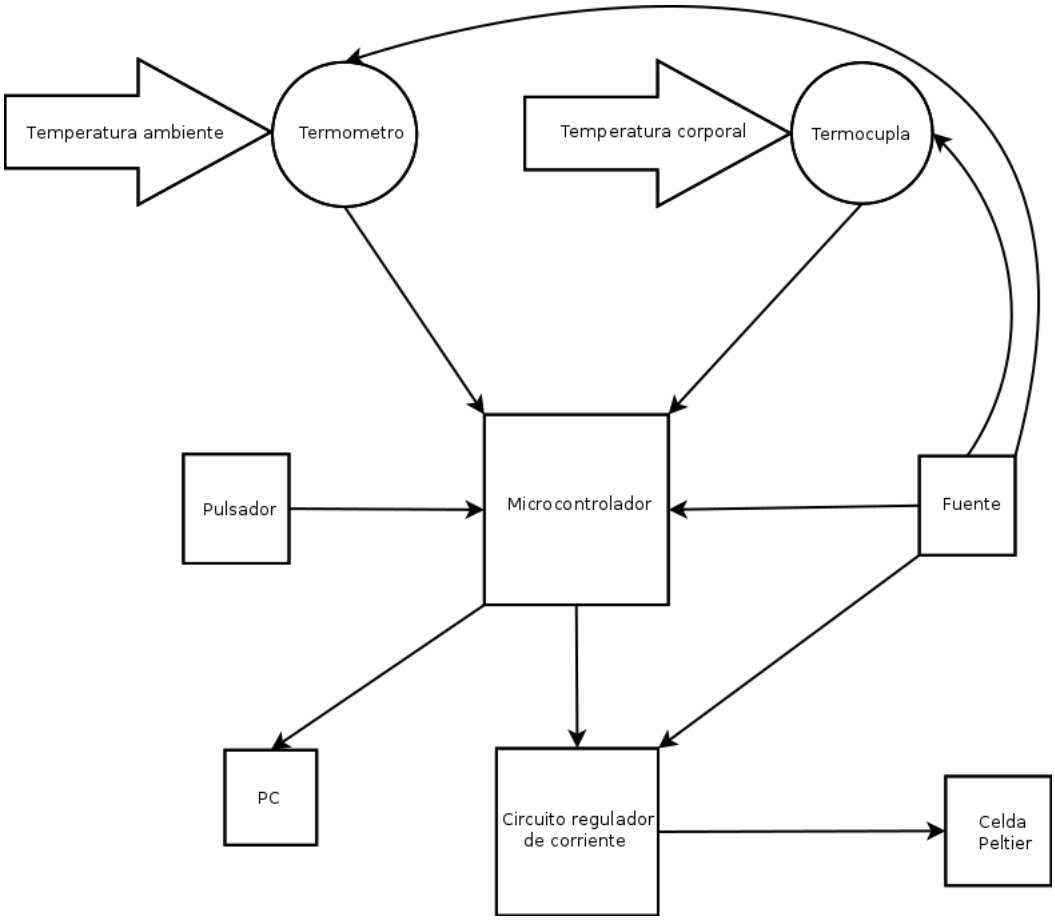

Figura 2: Diagrama de bloques

3. Diseño

3.1. Esquematico

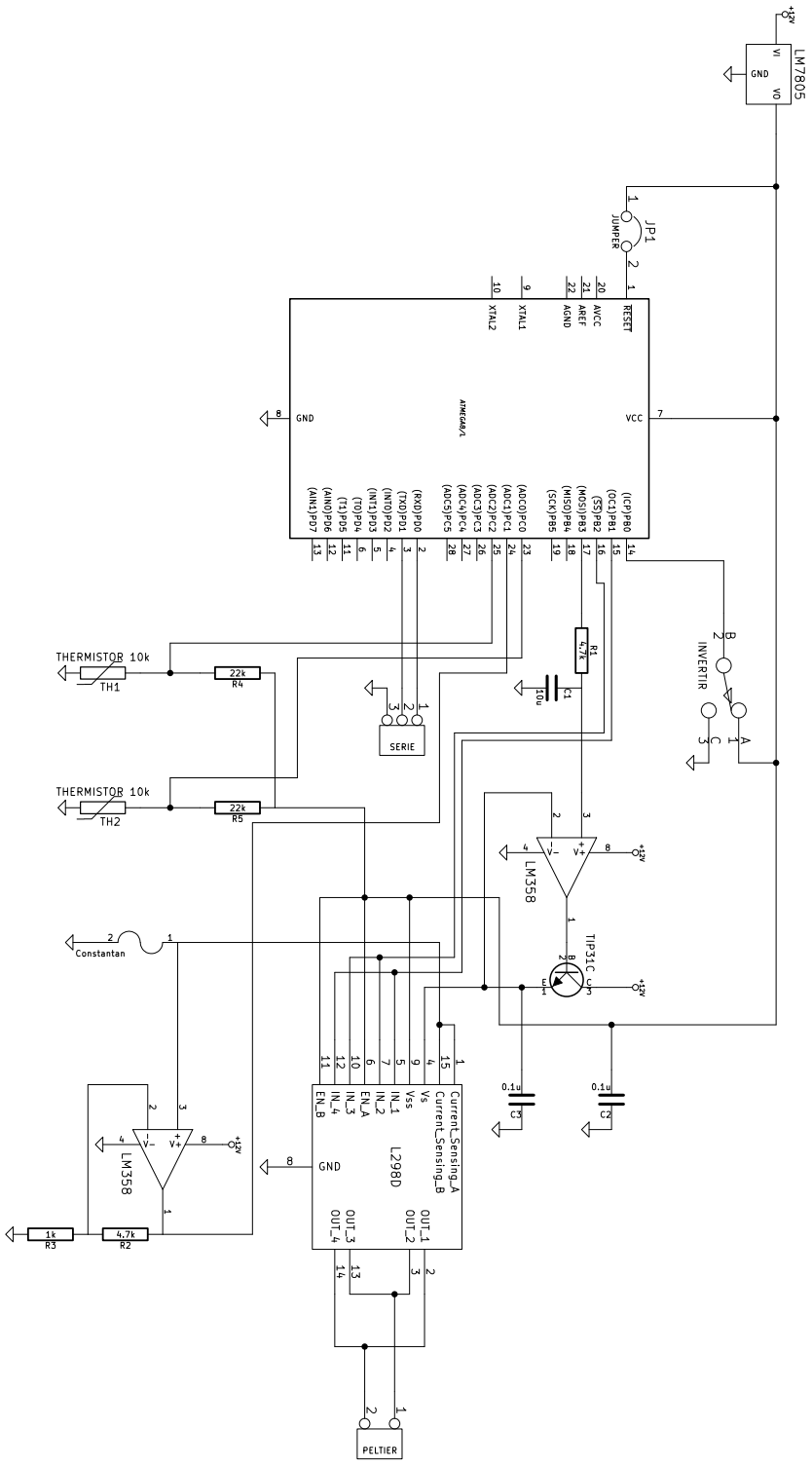

Figura 3: diagrama esquemático

3.2. Componentes

- Celda Peltier: Celda peltier de 10 W y 15x15 mm
- LM7805: Regulador de tensión para habilitar el puente H, alimentar el microcontrolador y suministrarle tensión constante a las resistencias conectadas en serie a los termistores.
- Interruptor: Interruptor para activar la inversión de la polaridad.
- Resistencias:
  - Dos reiststencias de 4,7 kΩ
  - Dos reiststencias de 22,0 kΩ
  - Una resistencia de 1,0 kΩ
- Capacitores:
  - 1 capacitor de 10 μF para generar tensión constante del PWM recibido.
  - 2 capacitores de 0.1 μF Conectados en paralelo a las alimentaciones del puente H, recomendados por el fabricante.
- LM358: Dos amplificadores operacionales. Uno para suministrar corriente a la base del NPN y el segundo para amplificar la tensión leída del constantán.
- TIP31C: Transistor de potencia NPN, utilizado para regular la corriente.
- L298D: Puente H utilizado para invertir la polaridad de la celda Peltier
- Constantán: alambre utilizado para sensar la corriente generada.
- Bateria: de 12 V y 2,9 Ah
- Pines:
  - 3 pines para el puerto serie.
  - 2 pines para el reseteo del microcontrolador.
  - 2 pines para conectar la celda peltier al circuito.
- Termistores: Dos termistores NTC de 10 kΩ

3.3. Regulador de Corriente

Se utilizó un regulador de corriente controlado por un PWM como se muestra en la figura 4

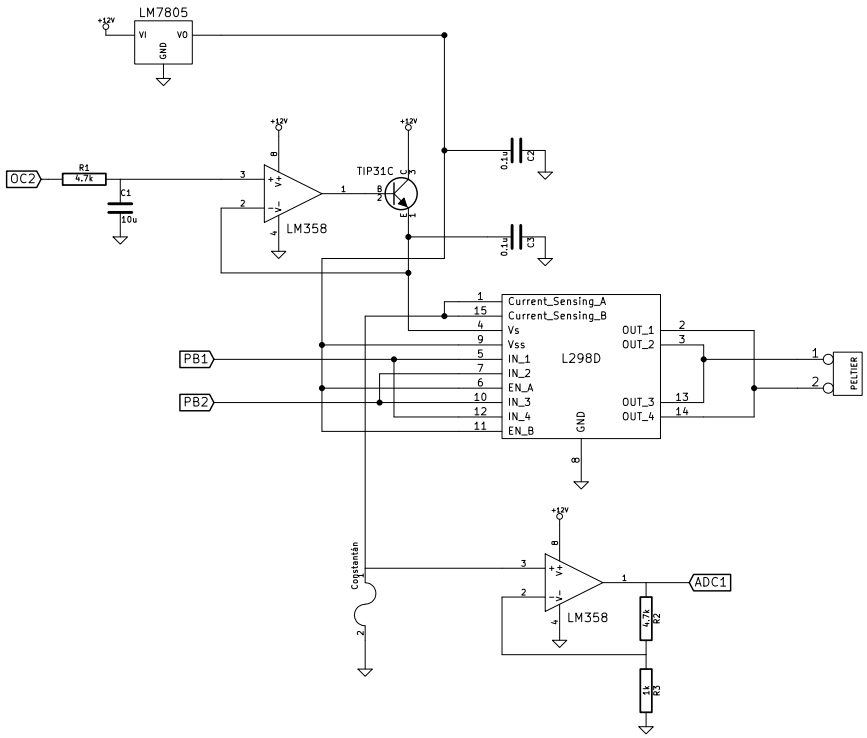

Figura 4: Regulador de Corriente

El circuito RC generará una tensión constante del PWM generado por el microcontrolador. Dicha tensión regulará la corriente suministrada a la base del NPN por el amplificador operacional, generando una corriente constnte entre el colector y el emisor del transistor.

Se optó por utilizar un L298D para el puente H ya que cuenta en un mismo integrado dos puentes H que soportan 2 A de corriente. Conectados en paralelo como se muestra en la figura 4 se puede duplicar dicha corriente máxima para que soporte hasta 4 A de corriente.

Al final del circuito se sensará la corriente generada mediante la tensión en el alambre constatán que es amplificada por el amplificador operacional. Para que la tensión de salida varíe entre 0 V y 2,56 V y sea leído por el microcontrolador.

Las resistencias del amplificador se obtuvieron considerando que para la corriente maxima registrada, la salida no supere los 2,56 V. Se registró una corriente maxima de 1,75 A y se midió una resistencia de 0,25 Ω.

La tensión de salida se obtiene mediante:

$$V_{ADC1} = R_{constantan} I_{MAX} \frac{R_3 + R_2}{R_3} \tag{1}$$

Luego fijando $R_3 = 1\,\text{k}\Omega$ y $R_2 = 4{,}7\,\text{k}\Omega$ se verificó que la tensión no supere los 2,56 V:

$$V_{ADC1} = 0{,}25\,\Omega\ 1{,}75\,\text{A} \frac{1\,\text{k}\Omega + 4{,}7\,\text{k}\Omega}{1\,\text{k}\Omega} = 2{,}49\text{V}$$

3.4. Medición de la temperatura

Para medir la temperatura se utilizó un divisor resistivo utilizando termistores para medir su tensión y poder estimar la temperatura. Se obtuvieron las resistencias a conectar en serie con los termistores de forma que la tensión maxima no supere los 2,56 V

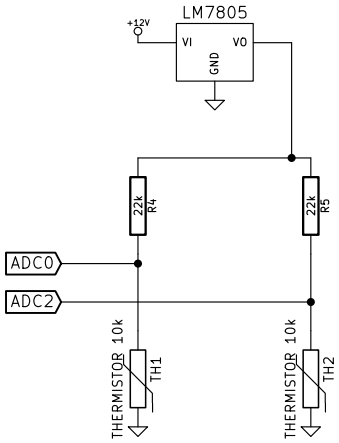

Figura 5: Divisor de tensión de los termistores

$$V_{termistor} = 5\,\mathrm{V}\frac{R_{termistor}}{R_{termistor} + R_{serie}} \tag{2}$$

Finalmente se eligió una resistencia de $R_{serie} = 22\,\mathrm{k\Omega}$ para R_4 y R_5 verificando que la tensión en los termistores no supere la tensión de referencia del ADC del microcontrolador para la resistencia maxima registrada en los termistores a $R_{0\,^\circ\mathrm{C}} = 15\,\mathrm{k\Omega}$:

$$V_{termistor} = 5\,\mathrm{V}\frac{15\,\mathrm{k\Omega}}{15\,\mathrm{k\Omega} + 22\,\mathrm{k\Omega}} = 2\,\mathrm{V}$$

4. Especificaciones del microcontrolador

4.1. Microcontrolador

Para este proyecto se utilizo un microcontrolador `Atmega8L`. El datasheet del mismo se puede obtener en la pagina de Atmel[4]

4.2. Configuraciones

4.2.1. Clock

El clock del microcontrolador fue establecido en 8MHz.

7	6	5	4	3	2	1	0
a	a	a	a	a	a	a	a
0	0	0	0	0	0	0	0

4.2.2. PWM

El PWM

5. Codigo Proyecto

```
#include <avr/io.h>

.section .data
.org 0x000

Temp_Ambiente:   .byte 0
Temp_Disipador:  .byte 0
Temp_Peltier:    .byte 0
Tension_Salida:  .byte 0

Modo_Operacion:  .byte 0

Modo_Standby:    .byte 0

PWM:             .byte 0
Tension_min:     .byte 0
Tension_max:     .byte 0
Iterador:        .byte 0

.section .text
.org 0x0
.global main
rjmp    main

#define low(x)   lo8(x)
#define high(x)  hi8(x)

;Regs:
;r16: Temporal, pasaje de parametro y de retorno
#define Reg_Temporal r16

;r20: Contador
#define Contador r20

;Constantes:
;Tipos de dato para mandar por serial

#define Dato_Tempe_Ambiente      'A'
#define Dato_Tempe_Disipador     'T'
#define Dato_Num_Iteracion       'I'
#define Dato_Tension_Salida      'P'
#define Dato_Tempe_Peltier       'X'
#define Dato_PWM                 'W'
#define Dato_max                 'M'
#define Dato_min                 'N'

#define Incremento_Pulso_Calor    3
#define Incremento_Pulso_Frio    10
#define Incremento_Regulacion     1
#define PWM_inicial              95


#define Eeprom_Inicio_Calor  0x64
#define Eeprom_Inicio_Frio   0x94


;Saltea el vector de interrupcion
.org 0x0020
```

```
59  main:

61  STACK_Init:
        ldi     Reg_Temporal,       low(RAMEND)
63      out     _SFR_IO_ADDR(SPL),  Reg_Temporal
        ldi     Reg_Temporal,       high(RAMEND)
65      out     _SFR_IO_ADDR(SPH),  Reg_Temporal

67      rcall   PWM_Init
        rcall   PUENTE_H_Init
69      rcall   USART_Init

71  LOOP:
        ;Iteraciones, se usa para saber la diferencia que se debe obtener
73      clr     Contador

75      rcall   GET_MODE                            ;obtengo el modo de operacion

77      ldi     r29,                PWM_inicial     ;PWM
        sts     PWM,                r29
79      rcall   RESET_PWM

81  REDUCIR_LOOP:

83      mov     Reg_Temporal,       Contador

85      ldi     r27,                10
        rcall   ESPERA
87      rcall   AUMENTAR_PULSO


89
        inc     Contador                            ;Cumpli una vuelta
91      ldi     Reg_Temporal,       6
        cpse    Contador,           Reg_Temporal    ;No saltar si ya ejecuto 5 vueltas
93
        rjmp    REDUCIR_LOOP
95
        cbi     _SFR_IO_ADDR(DDRB), 3               ;(OC2) para salida
97      sbi     _SFR_IO_ADDR(DDRB), 3               ;(OC2) para salida
        cbi     _SFR_IO_ADDR(PORTB),3
99
;       rcall STANDBY
101     rjmp LOOP

103 ;-------------------------------------Funciones-------------------------------------
    RESET_PWM:
105     ;8 bits mas significativos de la direccion de la eeprom a leer
        clr     r17
107     lds     r31,                Modo_Operacion
        cpi     r31,                0
109     breq    TABLA_FRIO
    TABLA_CALOR:
111     ldi     Reg_Temporal,       Eeprom_Inicio_Calor
        rjmp    CARGAR_TABLA
113 TABLA_FRIO:
        ldi     Reg_Temporal,       Eeprom_Inicio_Frio
115 CARGAR_TABLA:
        inc     Reg_Temporal
117     inc     Reg_Temporal
        sts     Iterador,           Reg_Temporal
```

```
119     rcall   LEER_EEPROM
        sts     Tension_min, r25
121
        inc     Reg_Temporal
123     inc     Reg_Temporal
        rcall   LEER_EEPROM
125     sts     Tension_max, r25

127     lds     r29, PWM
        out     _SFR_IO_ADDR(OCR2), r29
129
        ret
131 ;-----------------------------------------------------------------------
    ; espera r27 * 100mseg
133 ESPERA:
        rcall   SET_PWM
135     rcall   TRANSMITIR_DATOS
        rcall   DEMORA
137


139     dec     r27
        clr     Reg_Temporal
141     cpse    r27, Reg_Temporal
        rjmp    ESPERA
143     ret

145 ;-----------------------------------------------------------------------
    ; Lee de eeprom en la direccion indicada en los registros r17 para los 8 bits
147 ; mas significativos y r16 para los 8 bits menos significativos. Guarda el contenido
    ; en el registro r25
149 LEER_EEPROM:
        ;Espera hasta que la ultima escritura este terminada
151     sbic    _SFR_IO_ADDR(EECR), EEWE
        rjmp    LEER_EEPROM
153
        ;r17 elige la tabla
155     out     _SFR_IO_ADDR(EEARH),r17
        ;r16 el campo
157     out     _SFR_IO_ADDR(EEARL),r16

159     ;habilita el modo lectura
        sbi     _SFR_IO_ADDR(EECR), EERE
161     ;guarda el contenido de la ódireccin antes cargada en r25
        in      r25, _SFR_IO_ADDR(EEDR)
163     ret

165 ;-----------------------------------------------------------------------
    ; Inicializa el puente H seteando los pines 1 y 2 del puerto B como salida.
167 PUENTE_H_Init:
        sbi     _SFR_IO_ADDR(DDRB), 2
169     sbi     _SFR_IO_ADDR(DDRB), 1
        ret
171
    MODO_FRIO:
173     sbi     _SFR_IO_ADDR(PORTB),1
        cbi     _SFR_IO_ADDR(PORTB),2
175     ret

177 MODO_CALOR:
        cbi     _SFR_IO_ADDR(PORTB),1
```

```
179     sbi     _SFR_IO_ADDR(PORTB),2
        ret
181 ;-----------------------------------------------------------------------

183 TRANSMITIR_DATOS:
        rcall   LEER_AMBIENTE
185     rcall   LEER_DISIPADOR
        rcall   LEER_PELTIER
187
        ;Envio la iteracion
189     ldi     Reg_Temporal,       Dato_Num_Iteracion      ; tipo de dato a mandar
        rcall   USART_Transmit
191     mov     Reg_Temporal,       Contador
        rcall   USART_Transmit
193
        ldi     Reg_Temporal,       Dato_Tempe_Ambiente     ; tipo de dato a mandar
195     rcall   USART_Transmit
        lds     Reg_Temporal,       Temp_Ambiente
197     rcall   USART_Transmit

199     ldi     Reg_Temporal,       Dato_Tension_Salida     ; tipo de dato a mandar
        rcall   USART_Transmit
201     lds     Reg_Temporal,       Tension_Salida
        rcall   USART_Transmit
203
        ldi     Reg_Temporal,       Dato_Tempe_Disipador    ; tipo de dato a mandar
205     rcall   USART_Transmit
        lds     Reg_Temporal,       Temp_Disipador
207     rcall   USART_Transmit

209     ldi     Reg_Temporal,       Dato_Tempe_Peltier      ; tipo de dato a mandar
        rcall   USART_Transmit
211     lds     Reg_Temporal,       Temp_Peltier
        rcall   USART_Transmit
213
        ldi     Reg_Temporal,       Dato_PWM                ; tipo de dato a mandar
215     rcall   USART_Transmit
        lds     Reg_Temporal,       PWM
217     rcall   USART_Transmit

219     ldi     Reg_Temporal,       Dato_min                ; tipo de dato a mandar
        rcall   USART_Transmit
221     lds     Reg_Temporal,       Tension_min
        rcall   USART_Transmit
223
        ldi     Reg_Temporal,       Dato_max                ; tipo de dato a mandar
225     rcall   USART_Transmit
        lds     Reg_Temporal,       Tension_max
227     rcall   USART_Transmit

229     ret

231 ;-----------------------------------------------------------------------
    LEER_AMBIENTE:
233     ldi     Reg_Temporal,       0b11000000          ; canal 0 temperatura ambiente
        rcall   READ_ADC                                ; leer tension del peltier
235     rcall   TRADUCIR_TERMISTOR
        sts     Temp_Ambiente,      Reg_Temporal
237     ret
```

```
239 LEER_DISIPADOR:
        ldi     Reg_Temporal,       0b11000010          ; canal 2 temperatura disipador
241     rcall   READ_ADC                                ; leer tension del termistor
        rcall   TRADUCIR_TERMISTOR
243     sts     Temp_Disipador,     Reg_Temporal
        ret
245
    LEER_PELTIER:
247     ldi     Reg_Temporal,       0b11000001          ; canal 1 tension peltier
        rcall   READ_ADC                                ; leer tension del peltier
249     sts     Tension_Salida,     Reg_Temporal
        rcall   TRADUCIR_PELTIER
251     ret

253 ;-----------------------------------------------------------------------------
    ;Set PWM
255 ;Setea el pwm del pin OC2 con el tiempo en bajo pasado como parametro
    ;Reg_Temporal: tiempo en bajo a asignar
257
    SET_PWM:
259     lds     r30, PWM
        lds     Reg_Temporal,       Modo_Standby
261     cpi     Reg_Temporal,       1
        ;Si esta en modo standby no realiza cambios
263     breq    APLICAR_CAMBIO

265     lds     r29,                Tension_Salida
        lds     r17,                Tension_min
267     lds     r18,                Tension_max
        ldi     r26,                Incremento_Regulacion ;valor a ser restado o sumado
269
        cp      r29,                r17
271     brlo    AUMENTAR

273     cp      r29,                r18
        brlo    APLICAR_CAMBIO
275
    DISMINUIR:
277     cpi     r30,                100
        breq    APLICAR_CAMBIO
279     add     r30,                r26
        rjmp    APLICAR_CAMBIO
281 AUMENTAR:
        cpi     r30,                0
283     breq    APLICAR_CAMBIO
        sub     r30,                r26
285 APLICAR_CAMBIO:
        sts     PWM,                r30
287     out     _SFR_IO_ADDR(OCR2), r30
        ret
289 ;-----------------------------------------------------------------------------

291 AUMENTAR_PULSO:
        ;8 bits mas significativos de la direccion de la eeprom a leer
293     clr     r17

295     lds     Reg_Temporal,       Iterador
        inc     Reg_Temporal
297     inc     Reg_Temporal
        sts     Iterador,           Reg_Temporal
```

```
299
        rcall   LEER_EEPROM
301     sts     Tension_min,        r25
        inc     Reg_Temporal
303     inc     Reg_Temporal
        rcall   LEER_EEPROM
305     sts     Tension_max,        r25

307     lds     r31,                Modo_Operacion
        cpi     r31,                0
309     breq    AUMENTO_FRIO
AUMENTO_CALOR:
311     ldi     r26,                Incremento_Pulso_Calor
        rcall AUMENTAR
313     ret
AUMENTO_FRIO:
315     ldi     r26,                Incremento_Pulso_Frio
        rcall   AUMENTAR
317     ret

319 ;-----------------------------------------------------------------------
;Espera durante 100mseg
321 DEMORA:
        ldi     Reg_Temporal,       0xCF            ; Valores de los que empieza a contar
323     out     _SFR_IO_ADDR(TCNT1H),Reg_Temporal
        ldi     Reg_Temporal,       0x2B
325     out     _SFR_IO_ADDR(TCNT1L),Reg_Temporal
        ldi     Reg_Temporal,       4               ; 0000 0100 habilita poner en 1
327     out     _SFR_IO_ADDR(TIFR), Reg_Temporal
        out     _SFR_IO_ADDR(TIMSK),Reg_Temporal    ; el bit 3 de TIRF cuando haya overflow
329     ldi     Reg_Temporal,       0b00000011      ; velocidad: clk/64
        out     _SFR_IO_ADDR(TCCR1B),Reg_Temporal
331
DEMORA_LOOP:
333     in      Reg_Temporal,       _SFR_IO_ADDR(TIFR)
        sbrs    Reg_Temporal,       2
335     rjmp    DEMORA_LOOP

337     ldi     Reg_Temporal,       1
        out     _SFR_IO_ADDR(TIFR), Reg_Temporal
339     clr     Reg_Temporal                        ; finalizo contador
        out     _SFR_IO_ADDR(TIFR), Reg_Temporal
341     out     _SFR_IO_ADDR(TCCR1B),Reg_Temporal

343     ret


345
;-----------------------------------------------------------------------
347 ;Standby
;Espera durante 10 segundos
349 STANDBY:
        ldi     Reg_Temporal,       1
351     sts     Modo_Standby,       Reg_Temporal
        ldi     r29,                255             ;PWM
353     sts     PWM,                r29
        rcall   SET_PWM
355     ldi     r27,                100
        rcall   ESPERA
357     clr     Reg_Temporal
        sts     Modo_Standby,       Reg_Temporal
```

```
359     ret

361 ;-----------------------------------------------------------------------
    ;Transmit
363 ;Transmite por el puerto paralelo el dato pasado como parametro
    ;Reg_Temporal: valor a transmitir
365
    USART_Transmit:
367     sbis    _SFR_IO_ADDR(UCSRA),UDRE            ;Espero a que se libere el UDRE
        rjmp    USART_Transmit
369
        out     _SFR_IO_ADDR(UDR),  Reg_Temporal
371     ret

373 ;-----------------------------------------------------------------------
    ;Usart init
375 ;Inicializa el USART para poder enviar datos

377 USART_Init:
        ldi     Reg_Temporal,       (1<<TXEN)    ;enable
379     out     _SFR_IO_ADDR(UCSRB),Reg_Temporal

381     ;8bits, 1bit de stop, sin bit de paridad
        ldi     Reg_Temporal,       (1<<URSEL)|(3<<UCSZ0)
383     out     _SFR_IO_ADDR(UCSRC),Reg_Temporal

385     ldi     Reg_Temporal,       0xC                 ;Baud 38400 (Clock de 8Mhz)
        out     _SFR_IO_ADDR(UBRRL),Reg_Temporal
387
        ret
389
    ;-----------------------------------------------------------------------
391 ;Read adc
    ;Lee un dato del conversor adc y lo devuelve
393 ;Reg_Temporal: canal del cual leer
    ;Reg_Temporal: valor leido devuelto
395
    READ_ADC:
397
        out     _SFR_IO_ADDR(ADMUX),Reg_Temporal    ;
399     ldi     Reg_Temporal,       0b11001111      ;
        out     _SFR_IO_ADDR(ADCSRA),Reg_Temporal   ;
401
    WAIT_ADC:
403
        in      Reg_Temporal,       _SFR_IO_ADDR(ADCSRA)
405     ;Espera a que finalice la lectura
        sbrs    Reg_Temporal,       4
407     rjmp    WAIT_ADC

409     sbi     _SFR_IO_ADDR(ADCSRA),4

411     in      Reg_Temporal,       _SFR_IO_ADDR(ADCL)  ;
        in      r17,                _SFR_IO_ADDR(ADCH)  ;
413     lsr     r17
        ror     Reg_Temporal
415     lsr     r17
        ror     Reg_Temporal
417
        ret
```

```
419
    ;-----------------------------------------------------------------------------
421 ;PWM init
    ;Inicializa los puertos de salida del pwm
423 PWM_Init:
        sbi     _SFR_IO_ADDR(DDRB), 3               ;(OC2) para salida
425     ;(01110001) Phase correct, no pre escalar, clear on match
        ldi     Reg_Temporal,       0x71
427     out     _SFR_IO_ADDR(TCCR2),Reg_Temporal
        ret
429
    ;-----------------------------------------------------------------------------
431 ;Traducir termistor
    ;Convierte el valor recibido por parametro en su temperatura equivalente
433 ;entrada: Reg_Temporal: valor leido por el ADC

435 TRADUCIR_TERMISTOR:
        mov     r18,                Reg_Temporal
437     ldi     r17,                0               ;tabla termistor
        ldi     Reg_Temporal,       0               ;indice
439 LOOP_BUSQUEDA_TERM:
        rcall   LEER_EEPROM
441     cpi     r25,                0
        breq    FIN_TABLA
443     cp      r18,                r25             ;leido vs valor tabla
        brsh    END_TERMISTOR
445     inc     Reg_Temporal
        inc     Reg_Temporal
447     rjmp    LOOP_BUSQUEDA_TERM

449 END_TERMISTOR:
        inc     Reg_Temporal
451     rcall   LEER_EEPROM
        mov     Reg_Temporal,       r25
453     ret

455 FIN_TABLA:
        dec     Reg_Temporal
457     dec     Reg_Temporal
        rjmp    END_TERMISTOR
459
    ;-----------------------------------------------------------------------------
461 ;Traducir peltier
    ;Convierte el valor recibido por parametro en su temperatura equivalente
463 ;entrada: Temp_Disipador, Tension_Salida, Modo_Operacion
    ;salida: Temp_Peltier
465
    TRADUCIR_PELTIER:
467     lds     r18,                Tension_Salida
        ldi     r17,                0               ;tabla termistor
469     lds     r31,                Modo_Operacion
        cpi     r31,                0
471     breq    TABLA_FRIO_PELTIER
    TABLA_CALOR_PELTIER:
473     ldi     Reg_Temporal,       Eeprom_Inicio_Calor
        rjmp    LOOP_BUSQUEDA_TERM_PELTIER
475 TABLA_FRIO_PELTIER:
        ldi     Reg_Temporal,       Eeprom_Inicio_Frio
477     rjmp    LOOP_BUSQUEDA_TERM_PELTIER
```

```
479 LOOP_BUSQUEDA_TERM_PELTIER:
        rcall   LEER_EEPROM
481     cpi     r25,            0xFF
        breq    FIN_TABLA_PELTIER
483     cp      r25,            r18             ; valor tabla vs leido
        brsh    END_PELTIER
485     inc     Reg_Temporal
        inc     Reg_Temporal
487     rjmp    LOOP_BUSQUEDA_TERM_PELTIER
    END_PELTIER:
489     inc     Reg_Temporal
        rcall   LEER_EEPROM
491     lds     r17,            Temp_Disipador
        lds     r18,            Modo_Operacion
493
        mov     Reg_Temporal,   r25
495
        sbrc    r18,            0
497     rjmp    CALCULO_CALOR
        rjmp    CALCULO_FRIO
499
    CALCULO_CALOR:
501     add     Reg_Temporal,   r17
        sts     Temp_Peltier,   Reg_Temporal
503     ret

505 CALCULO_FRIO:
        sub     r17,            Reg_Temporal
507     mov     Reg_Temporal,   r17
        sts     Temp_Peltier,   Reg_Temporal
509     ret

511 FIN_TABLA_PELTIER:
        dec     Reg_Temporal
513     dec     Reg_Temporal
        rjmp    END_PELTIER
515
    ;-----------------------------------------------------------------------------
517 ;Get mode
    ;Devuelve el valor del modo en el cual se ejecuta para la temperatura Reg_Temporal
519 ;Reg_Temporal: temperatura leida
    ;Reg_Temporal: valor leido devuelto
521 GET_MODE:
        ldi     Reg_Temporal,   0b11000000      ; canal 0 temperatura ambiente
523     rcall   READ_ADC
        rcall   TRADUCIR_TERMISTOR              ;obtengo la temperatura
525     cpi     Reg_Temporal,   50              ;temperatura arbitraria para el modo cal
        brsh    COLD_MODE
527
    HOT_MODE:
529     sbic    _SFR_IO_ADDR(PINB), 0           ; Si el pin esta en 1 cambia de modo
        rjmp    SET_COLD
531
    SET_HOT:
533     cbi _SFR_IO_ADDR(PORTB), 1
        sbi _SFR_IO_ADDR(PORTB), 2
535
        ldi     Reg_Temporal,   1               ;1 es modo calor
537     sts     Modo_Operacion, Reg_Temporal    ;Modo_Operacion modo de operacion inicia
```

```
539         ret

541 COLD_MODE:
            sbic    _SFR_IO_ADDR(PINB), 0           ; Si el pin esta en 1 cambia de modo
543         rjmp    SET_HOT

545 SET_COLD:
            sbi _SFR_IO_ADDR(PORTB),    1
547         cbi _SFR_IO_ADDR(PORTB),    2

549         ldi     Reg_Temporal,       0           ;0 es el modo frio
            sts     Modo_Operacion,     Reg_Temporal    ;Modo_Operacion modo de operacion inicia
551
            ret
553
    .section .eeprom
555 .org   0x0000

557 ;Tablas de conversion
    ;Formato: [(Tension medida, Temperatura*2),...]
559 .byte   204 ,   14
    .byte   201 ,   16
561 .byte   199 ,   18
    .byte   197 ,   20
563 .byte   195 ,   22
    .byte   194 ,   24
565 .byte   193 ,   26
    .byte   192 ,   28
567 .byte   190 ,   30
    .byte   188 ,   32
569 .byte   186 ,   34
    .byte   184 ,   36
571 .byte   182 ,   38
    .byte   181 ,   40
573 .byte   172 ,   42
    .byte   169 ,   44
575 .byte   166 ,   46
    .byte   163 ,   48
577 .byte   160 ,   49
    .byte   159 ,   50
579 .byte   158 ,   53
    .byte   157 ,   56
581 .byte   156 ,   60
    .byte   155 ,   62
583 .byte   154 ,   64
    .byte   153 ,   66
585 .byte   152 ,   68
    .byte   151 ,   70
587 .byte   147 ,   72
    .byte   145 ,   74
589 .byte   143 ,   76
    .byte   141 ,   78
591 .byte   140 ,   80
    .byte   0, 0 ;FIN DE TABLA
593
    ; TABLA PARA MODO CALOR
595 ;Formato: [Tension minima, Dif_Temperatura*2]
    .org    0x0064
597 .byte   0   ,   0
    .byte   1   ,   0
```

```
599 .byte   5    ,   1
    .byte   10   ,   2
601 .byte   15   ,   3
    .byte   20   ,   4
603 .byte   25   ,   5
    .byte   30   ,   6
605 .byte   40   ,   7
    .byte   50   ,   8
607 .byte   60   ,   10
    .byte   80   ,   12
609 .byte   90   ,   16
    .byte   100  ,   24
611 .byte   110  ,   36
    .byte   0xFF, 0xFF  ;FIN DE TABLA
613
    ; TABLA PARA MODO FRIO
615 ;Formato: [Tension minima, Dif_Temperatura*2]

617 .org    0x0094
    .byte   0    ,   0
619 .byte   1    ,   0
    .byte   5    ,   1
621 .byte   10   ,   2
    .byte   15   ,   3
623 .byte   20   ,   4
    .byte   25   ,   5
625 .byte   30   ,   6
    .byte   40   ,   7
627 .byte   45   ,   8
    .byte   50   ,   9
629 .byte   60   ,   10
    .byte   80   ,   14
631 .byte   90   ,   16
    .byte   100  ,   18
633 .byte   110  ,   20
    .byte   115  ,   22
635 .byte   0xFF, 0xFF  ;FIN DE TABLA

637 .end
```

Referencias

[1] http://www.embrlabs.com/

[2] https://youtu.be/sDZHITVfYrI

[3] https://youtu.be/kvUMCip-r4A

[4] http://www.atmel.com/images/atmel-2486-8-bit-avr-microcontroller-atmega8_l_datasheet.