Fiuba
Facultad de Ingeniería
Universidad de Buenos Aires

(6609) LABORATORIO DE MICROCOMPUTADORAS

Proyecto:
Máquina para hacer pochoclos

Profesor:	**Ing. Guillermo Campiglio**
Cuatrimestre / Año:	**2c 2019**
Turno de clases prácticas:	**Miércoles**
Jefe de Trabajos Prácticos:	
Docente guía:	

Autores			**Seguimiento del proyecto**								
Nombre	**Apellido**	**Padrón**									
Nicolas	**Direnzo**	**98582**									
Victoria	**De Maio**	**99232**									

Observaciones:

Fecha de aprobación		

Firma J.T.P.

COLOQUIO	
Nota final	
Firma Profesor	

Índice

1. Objetivos

El principal objetivo de este proyecto es aplicar los contenidos adquiridos a lo largo del curso en un dispositivo de cocción de maíz. Para el mismo, se buscará lograr la correcta comunicación entre el microcontrolador y determinados periféricos que se detallaran en profundidad mas adelante.

2. Descripción del proyecto

El presente trabajo tiene como objetivo el diseño y la implementación de una maquina para hacer pochoclos. Para su implementación se hará uso de una placa $ArduinoMEGA$, que contiene adjuntado un micocontrolador $Atmega2560$, y el lenguaje de programación de bajo nivel $Assembler$ el cual nos permitirá programar el microcontrolador.
Como fue mencionado anteriormente, se utilizarán cuatro periféricos, en principio, que serán conectados al microcontrolar y permitirán el correcto funcionamiento de la máquina para hacer pochoclos. A continuación se detallan los mismos y sus implementaciones:

2.1. Resistencia o calentador eléctrico

Para hacer explotar el maíz es necesario elevar su temperatura hasta al menos 170 grados. Esto se logrará a través de contacto directo con una resistencia de cerámica que genera calor y la olla que contendrá el maíz.
Esta resistencia se conectará a 220v y se le agregara una etapa de control de potencia para regular su prendido y apagado de tal manera de mantener la temperatura y no quemar el maíz.

Figura 1: Resistencia ilustrativa

2.2. Termocupla

Este periférico es el encargado de medir la temperatura de la olla donde será colocado el maíz. El sensor que se va a utilizar genera una señal pequeña, por lo que será necesario el uso de una tarjeta de control MAX6675 que la amplificará, la convertirá a digital y enviara el valor de temperatura mediante la comunicación SPI.
Las conexiones de la misma son mediante los pines $MISO$, SCK y CS. El primero de estos pines se utiliza para enviar los datos desde la termocupla hacia el microcontrolador, el segundo de estos pines es el $Clock$, encargado de marcar el tiempo de envió de datos. Por ultimo, el CS llamado comúnmente $ChipSelect$ es el encargado de activarla.

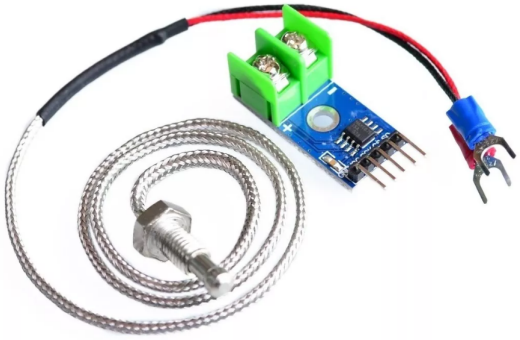

Figura 2: Termocupla ilustrativa

2.3. Motor

Se utilizó un motor bipolar paso a paso de 12v para mantener en movimiento los maíces en el fondo de la cacerola, la idea es lograr que los mismos no se peguen ni se quemen. Para esto se utilizará una varilla en forma de T.
Para controlar el motor, se hará uso del driver $PololuA4988$, el cual mantiene la conexión con el microcontrolador a partir de sus pines $step$ y dir los cuales corresponden al paso del motor y a la dirección con la que va a girar. El mismo también dispone de los pines Msk los cuales son para configurar el tipo de paso del motor, en nuestro caso querremos un paso completo para generar mayor torque.

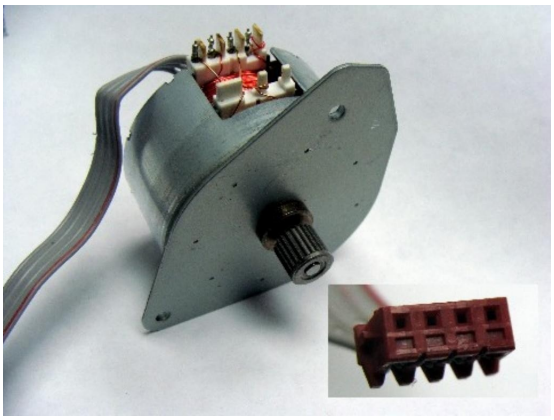

Figura 3: Motor bipolar ilustrativo.

2.4. Bluetooth

Para posibilitar la comunicación entre el usuario y la maquina se optó por utilizar una conexión bluetooth. Se genero una aplicación de celular Android mediante la pagina $Mit\ app\ inventor$ la cual se utilizará para el prendido y apagado de la pochoclera y además nos permitirá visualizar en tiempo real la temperatura alcanzada.
Para la conexión de la misma se utilizaran los pines Tx y Rx conectados en forma cruzada a los del microcontrolador de manera de recibir los datos mediante la comunicación $USART$.

Figura 4: Modulo bluetooth ilustrativo.

2.5. Relé de estado solido

Se utilizó un relé de estado solido para poder vincular el microcontrolador y la resistencia. El mismo cuenta con un pin negado de indicación de estado bajo (0v) o estado alto (5v) el cual permite prender o apagar la resistencia en el caso que sea menor o mayor a los limites correspondientes.

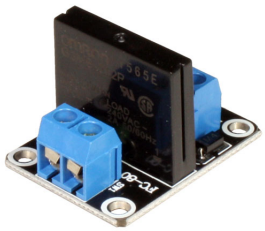

Figura 5: Relé de estado solido ilustrativo.

3. Diagrama en bloque

A continuación se detalla el diagrama de bloques del proyecto, en el mismo se muestra el microcontrolador, los periféricos, sus interconexiones y sus alimentaciones.

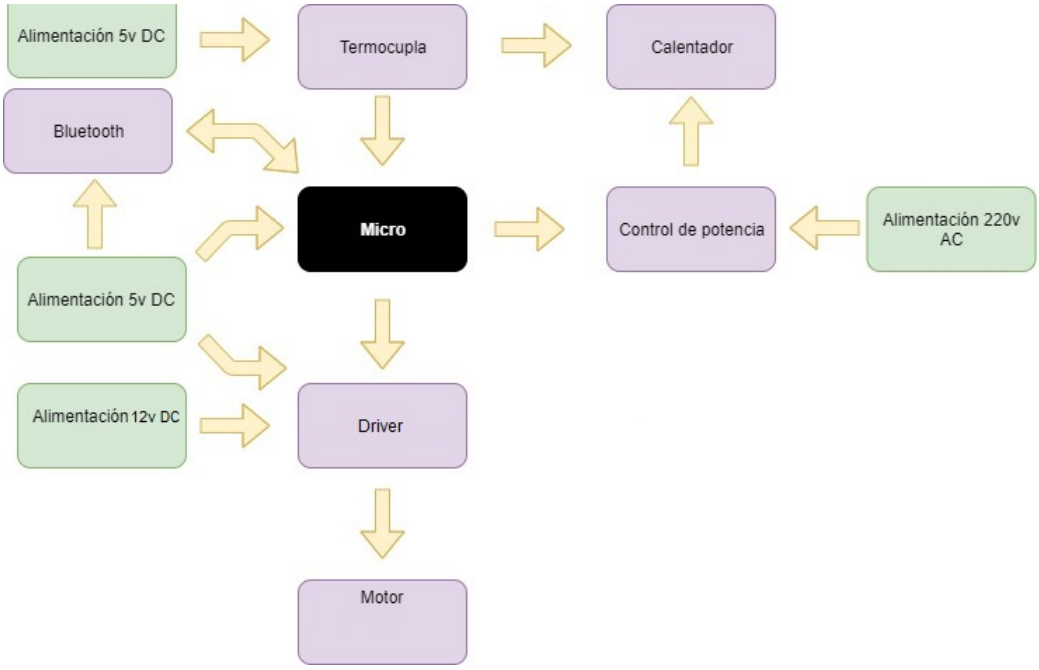

Figura 6: Diagrama en bloques del proyecto.

4. Circuito esquemático

Haciendo uso del software $Kicad$ realizamos el siguiente esquemático del proyecto en donde se observan todas las conexiones realizadas:

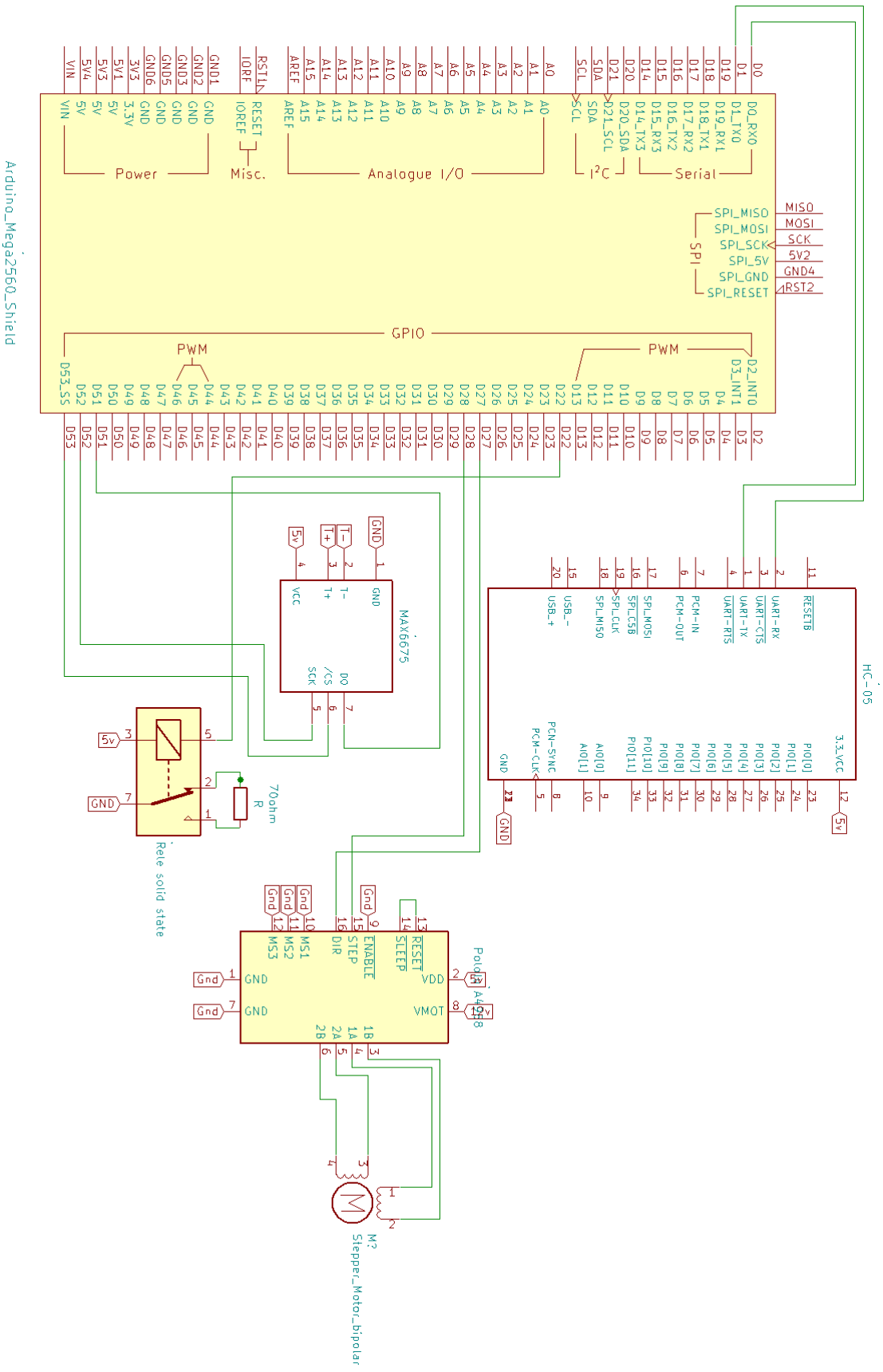

Figura 7: Esquemático circuital del proyecto.

5. Listado de Componentes

- Modulo Driver $Pololu$ A4988
- 1 placa Arduino MEGA 2560.
- 1 Termocupla tipo K.
- 1 Modulo MAX6675.
- 1 Módulo Bluetooth Arduino HC05
- Relé de estado solido.
- Fuente continua de 5v y 12v.
- Fuente alterna de 220v.
- Cables tipo banana - coco y coco - coco.
- Resistencia calentadora.

5.1. Presupuesto

- $150 Modulo Driver $Pololu$ A4988
- $750 Termocupla tipo K + Modulo MAX6675
- $100 Cables arduino
- $400 1 Módulo Bluetooth Arduino HC05
- $300 Relé de estado solido
- $140 Pines
- $1300 Arduino mega 2560
- $250 Placa experimental
- $200 Resistencia
- $350 Cacerola
- $300 Motor paso a paso

En total se llevó un presupuesto total de 4240 pesos.

6. Software - Diagrama de flujo o seudocódigo

Mediante la utilización del Software libre Dia, se realizó un diagrama de flujo principal seguido por ramificaciones de flujo del proyecto.

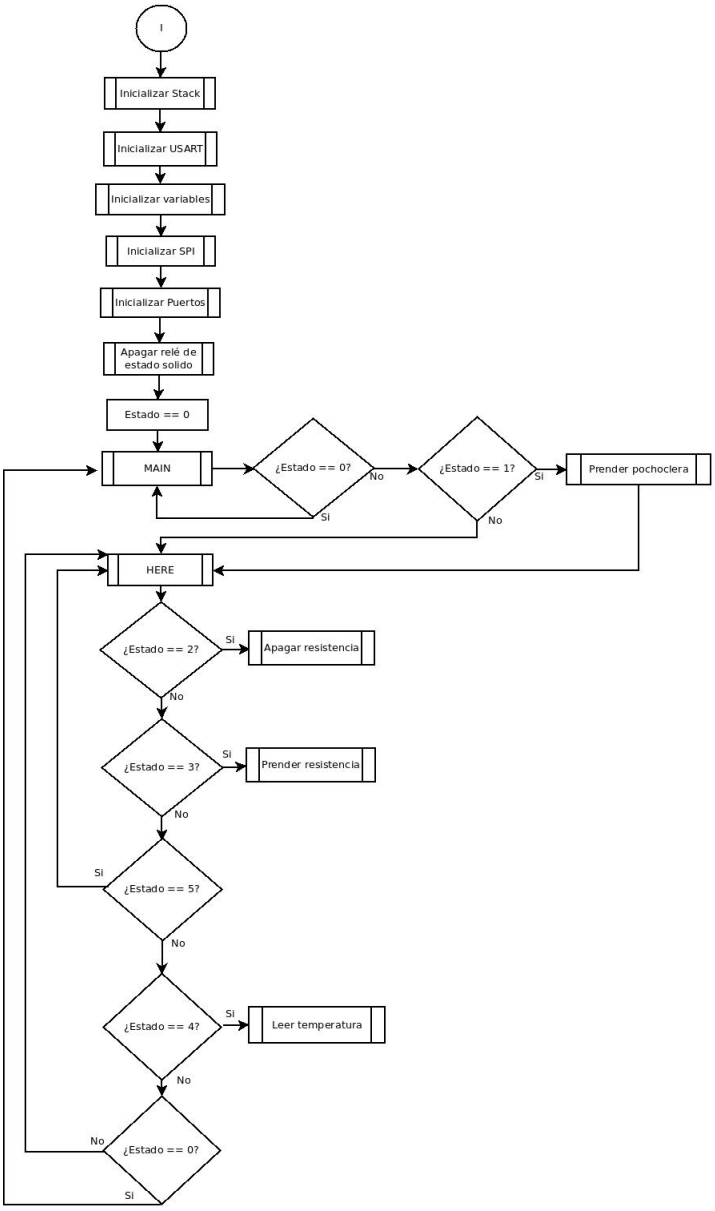

Figura 8: Diagrama de flujo principal del proyecto.

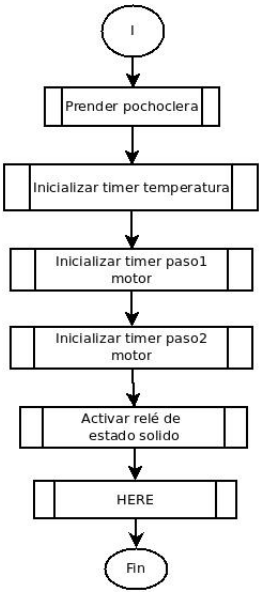

Figura 9: Diagrama de flujo de prender pochoclera del proyecto.

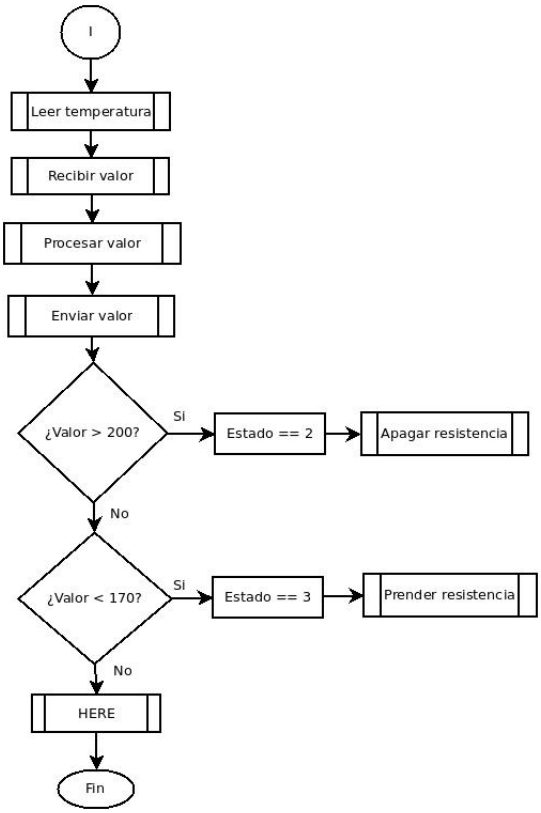

Figura 10: Diagrama de flujo de leer temperatura del proyecto.

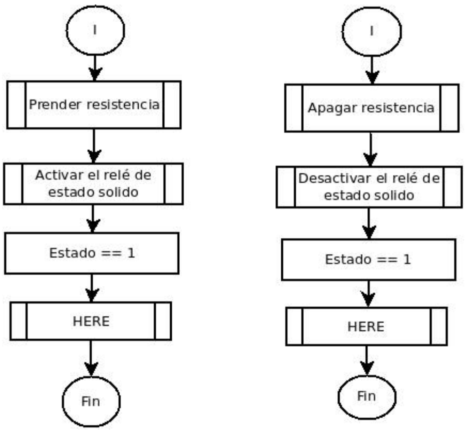

Figura 11: Diagrama de flujo prender y apagar resistencia del proyecto.

6.1. Código

```
.include "m2560def.inc"

.def temp      = r16
.def receive1   = r17
.def receive2   = r18
.def cant_shift = r19
.def cont1     = r20
.def cont2     = r21
.def temp2     = r22
.def estado    = r23


.cseg

.org 0x00
    jmp Init

.org URXC0addr            ; Interrupcion de usart (Bluetooth)
    rjmp  Handler_Int_URXC0

.org OC3Aaddr           ; Timer 3 = Lo uso para el paso (prendido)
    rjmp  Paso_ON

.org OC1Aaddr           ; Timer 1 = Lo  uso para la medicion de temperatura
    rjmp  Timer_1

.org OC4Aaddr           ; Timer 4 = Lo uso para el paso (apagado)
    rjmp  Paso_OFF

.org INT_VECTORS_SIZE


;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_;
;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_;
;-----------------------------------------------------------------------------------------------;
;-----------------------------------------------------------------------------------------------;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;PROGRAMA PRINCIPAL;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;-----------------------------------------------------------------------------------------------;
;-----------------------------------------------------------------------------------------------;
;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_;
;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_;



Init:

    ldi temp, low(RAMEND)
    out spl, temp
    ldi temp, high(RAMEND)
    out sph, temp
```

```
    call Usart_init       ; Inicializo la USART
    call Var_init       ; Inicializo variables varias
    call SPI_MasterInit     ; Inicializo SPI
    call Port_init        ; Inicializo puertos varios
    call Rele_solido_OFF    ; Inicializo el rele apagado
    call Estado_0       ; Inicializo el estado en apagado


    sei             ; Habilitacion global de interrupciones



MAIN:

    cpi estado, 0       ; Si el estado es 0 (pochoclera apagada) me quedo en el main
    breq MAIN
    cpi estado, 1       ; Si el estado es 1 (cambia por interrupcion) voy a prender la pochoclera y
    breq Pochoclera       ; luego al HERE


HERE:

    cpi estado, 2       ; Si el estado es 2 debo apagar la resistencia ya que paso el limite
    breq Apagar_resistencia   ; superior de temperatura
    cpi estado, 3       ; Si el estado es 3 debo prender la resistencia ya que la temperatura es
    breq Prender_resistencia  ; menor al limite inferior
    cpi estado, 5       ; Si el estado es 5 significa que lei un valor de temperatura
    breq HERE
    cpi estado, 4       ; Si el estado es 4 significa que debo leer un valor de temperatura
    breq Leer_temperatura
    cpi estado, 0       ; Si el estado es 0 debo ir al MAIN y esperar (apagaron la pochoclera)
    breq MAIN

    JMP HERE

;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_;
;-----------------------------------------------------------------------------------------------;
;-----------------------------------------------------------------------------------------------;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;,SUBRUTINAS;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;-----------------------------------------------------------------------------------------------;
;-----------------------------------------------------------------------------------------------;
;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_;



Apagar_resistencia:

    call Rele_solido_OFF    ; Desactivo la resistencia
    ldi estado, 1       ; Pongo el estado en prendido.

    jmp HERE

Prender_resistencia:

    call Rele_solido_ON2    ; Activo la resistencia
    ldi estado, 1       ; Pongo el estado prendido

    jmp HERE


Pochoclera:

    call Timer_init       ; Inicializo el timer que habilita la lectura de temperatura
    call Timer_Motor_init0    ; Inicializo el timer de paso 1
    call Timer_Motor_init2    ; Inicializo el timer de paso 2
    call Rele_solido_ON2    ; Prendo la resistencia


    jmp HERE          ; Vuelvo al ciclo principal a esperar que me cambien el estado
```

```
124 Rele_solido_ON:
125
126     cbi portA, 0
127     ldi estado, 1
128
129
130     jmp HERE
131
132 Rele_solido_OFF:
133
134     sbi portA, 0
135     ldi estado, 1
136
137
138     ret
139
140 Rele_solido_ON2:
141
142     cbi portA, 0
143     ldi estado, 1
144
145
146     ret
147
148 Usart_Transmit:
149
150     lds temp, UCSR0A
151     sbrs temp,UDRE0
152     rjmp Usart_Transmit     ; Espero hasta que el buffer de transmision este vacio
153
154     sts UDR0, receive1      ; Envio el dato de la temperatura (los 8 bits que representan la
155                   ;temperatura del 0 al 255)
156
157     ret
158
159
160 Temperature_extract:
161
162     cbr receive2, 7       ; Limpio los ultimos 3bits
163
164     clc             ; Limpio el Carry
165
166     lsl receive2        ; Corro hacia la izquierda
167     rol receive1        ; Corro hacia la izquiera con carry anterior.
168     clc
169
170     lsl receive2
171     rol receive1
172     clc
173
174     lsl receive2
175     rol receive1        ; Tengo en receive1 los 8 bits que me representan la temp de 0 a 255
176     clc
177
178     ret
179
180 Delay_1ms:
181
182     ldi cont2, 20
183     dec cont1
184     brne loop
185
186
187     ret
188
189 loop:
190     dec cont2
191     brne loop
192     jmp Delay_1ms
193
194 Leer_temperatura:
195
196     cbi portb, 0        ; Pongo en alto el #CS
```

```
197
198       call SPI_MasterReceive    ; Me voy a esperar datos de temperatura
199     call Temperature_extract  ; Pongo los datos en un registro
200
201     ldi estado,5        ; Cambio el estado a "temperatura leida"
202
203     call Usart_Transmit     ; Los envio por Bluetooth
204
205     cpi receive1,200      ; Si el valor de temperatura es mayor o igual a 200 debo cambiar el estado
206     brsh Estado_2       ; para apagar la resistencia
207
208     cpi receive1, 170     ; Si el valor de temperatura es menor a 200 debo cambiar el estado para
209     brlo Estado_3       ; prender la resistencia
210
211
212     jmp HERE
213
214 Estado_2:
215
216     ldi estado, 2       ; Cambio a estado de resistencia apagada.
217
218
219     jmp HERE
220
221
222 Estado_3:
223
224     ldi estado, 3       ; Cambio a estado de resistencia prendida
225
226
227     jmp HERE
228
229 SPI_MasterReceive:
230
231     cbi portb,0         ; Habilito la lectura poniendo en 0 #CS (negado)
232
233     ldi cont1, 243        ; Contador para delay
234     call delay_1ms        ; Espero a que se estabilice
235
236     ldi cant_shift, 7
237     call  SPI_Read
238     mov receive1,xh       ; Guardo el dato que vuelve en R17
239
240     ldi cant_shift, 7
241     ldi xh, 0
242     call  SPI_Read      ; Voy otra vez porque son 16 bits
243     mov receive2,xh       ; Guardo el dato que vuelve en R18
244
245     sbi portb,0         ; Deshabilito la lectura poniendo en 1 #CS (negado)
246
247     call delay_1ms        ; Espero a que se estabilice
248     call delay_1ms
249     call delay_1ms
250     call delay_1ms
251     call delay_1ms
252     call delay_1ms
253
254     ret
255
256 SPI_Read:
257
258     cbi portb, 1        ; Pono en bajo el clock
259
260     ldi cont1, 243        ; Contador para delay
261     call Delay_1ms        ; Espero 1ms
262
263     sbic pinb, 2        ; Salto de linea si en miso me llego un 0
264     set             ; Seteo el flag T
265
266     bld xh, 0         ; Coloco lo que tengo en flag T en el bit 0 de receive1
267     lsl xh            ; Shifteo
268
269     sbi portb, 1        ; Pongo en alto el clock
```

```

    ldi cont1, 243        ; Contador para delay
    call Delay_1ms        ; Espero 1ms

    dec cant_shift        ; Contador, lo quiero hacer 7 veces
    clt
    brne SPI_read


    ret

Pochoclera_OFF:

    clr temp
    sts TIMSK3, temp      ; Deshabilito el timer 3
    sts TIMSK4, temp      ; Deshabilito el timer 4
    sts TIMSK1, temp      ; Deshabilito el timer 1
    cbi porta, 6        ; Apago el paso
    call Rele_Solido_OFF    ; Apago la resistencia
    call Estado_0       ; Cambio el estado a "pochoclera apagada"


    reti

Estado_1:

    sbi PORTB,  7
    ldi estado, 1       ; Cambio estado a prendido: Tengo que prender el motor, la resistencia y
                  ; comenzar a medir

    reti
Estado_0:

    cbi PORTB,7
    ldi estado, 0       ; Cambio estado a apagado: Apaga todo.


    reti


;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_--_;
;-----------------------------------------------------------------------------------------------;
;-----------------------------------------------------------------------------------------------;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;,INTERRUPCIONES;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;-----------------------------------------------------------------------------------------------;
;-----------------------------------------------------------------------------------------------;
;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-;


;-----------------------------------------------------------------------------------------------;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;USART;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;-----------------------------------------------------------------------------------------------;
Handler_Int_URXC0:

    lds temp2, UDR0       ; Leo lo que recibi
    cpi temp2, 1        ; Lo comparo con un 1
    breq Estado_1       ; Si es un 1 voy a cambiar el estado a prendido
    cpi temp2, 0        ; Lo comparo con un 0
    breq Pochoclera_OFF     ; Si es un 0 voy a apagar la pochoclera


    reti            ; Fin de la interrupcion

;-----------------------------------------------------------------------------------------------;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;MEDICION DE TEMPERATURA;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;-----------------------------------------------------------------------------------------------;

Timer_1:

    ldi estado, 4       ; Cambio estado a "leer temperatura"

        ldi temp, 0x00        ; Habilito el timer
    sts TCNT1L, temp
```

```
343     ldi temp, 0x00
344     sts TCNT1H, temp      ; El timer comienza en 0000
345
346
347     reti
348
349 ;-----------------------------------------------------------------------------------------------;
350 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;PASO DEL MOTOR 1;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
351 ;-----------------------------------------------------------------------------------------------;
352
353 Paso_ON:
354
355     sbi porta, 6        ; Prendo el paso
356
357     ldi temp, 0x00        ; Deshablito la interrupcion. Se volvera a iniciar cuando salte PASO_OFF,
358     sts TIMSK3, temp      ; generando que el prendido y apagado del paso sea equiespaciado en tiempo
359
360
361     reti
362
363 ;-----------------------------------------------------------------------------------------------;
364 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;PASO DEL MOTOR 2;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
365 ;-----------------------------------------------------------------------------------------------;
366
367 Paso_OFF:
368
369     ldi temp, 0x00
370     sts TCNT3L, temp
371     ldi temp, 0x80
372     sts TCNT3H, temp
373
374     ldi temp, (1<<OCIE3A)   ; Hablito interrupcion de timer 3(quedo deshabilitada cuando salto
        PASO_ON)
375     sts TIMSK3, temp
376
377     cbi porta, 6        ; Apago el paso
378
379     ldi temp, 0x00
380     sts TCNT4L, temp
381     ldi temp, 0x00
382     sts TCNT4H, temp      ; Reinicio el timer con 0
383
384
385     reti
386
387
388 ;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-;
389 ;-----------------------------------------------------------------------------------------------;
390 ;-----------------------------------------------------------------------------------------------;
391 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;,INICIALIZACIONES;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
392 ;-----------------------------------------------------------------------------------------------;
393 ;-----------------------------------------------------------------------------------------------;
394 ;-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-;
395
396
397
398
399 ;-----------------------------------------------------------------------------------------------;
400 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;USART;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
401 ;-----------------------------------------------------------------------------------------------;
402 Usart_init:
403                   ; Seteo velocidad de transmision 9600 b/s
404     ldi temp, high((16000000/(8*9600) - 1))
405     sts UBRR0H, temp
406     ldi temp, low((16000000/(8*9600) - 1))
407     sts UBRR0L, temp
408                   ; Doble velocidad
409     ldi temp, (1<<U2X0)
410     sts UCSR0A, temp
411                   ; Habilito receptor y transmisor
412     ldi temp, (1<<RXCIE0)|(1<<RXEN0)|(1<<TXEN0)
413     sts UCSR0B,temp
414                   ; Seteo formato: trama de 8bits, 1 bite de stop
```

```
    ldi temp, (1<<UCSZ01)|(1<<UCSZ00)
    sts UCSR0C,temp


    ret
;-----------------------------------------------------------------------------------------------;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;SPI;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;-----------------------------------------------------------------------------------------------;


SPI_MasterInit:

    ldi temp, 0x00        ;   MOSI==PB2==51   SCK==PB1==52   #CS==PB0==53
    ldi temp,(1<<DDB1)|(1<<DDB0)    ; SCK (PB1) y #CS (PB0) como salida.
    out DDRB,temp

    sbi portb, 0        ; Pongo en alto el #CS
    ldi cont1, 243        ; Contador para delay
    call delay_1ms        ; Espero

    clt             ; Limpio el flag T


    ret


;-----------------------------------------------------------------------------------------------;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;TIMER 1;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;-----------------------------------------------------------------------------------------------;

Timer_init:

    ldi temp, 0x00
    sts TCNT1L, temp
    sts TCNT1H, temp      ; El timer comienza con 0000

    ldi temp, (1<<CS12)
    sts TCCR1B, temp      ; Configuro el timer con fclk/256

    ldi temp, (1<<OCIE1A)   ; Hablito la interrupcion
    sts TIMSK1, temp


    ret

;-----------------------------------------------------------------------------------------------;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;TIMER 3;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;-----------------------------------------------------------------------------------------------;
Timer_Motor_init0:

    ldi temp, 0x00
    sts TCNT3L, temp
    ldi temp, 0x80
    sts TCNT3H, temp      ; El timer comienza con temp

    ldi temp, (1<<CS30)
    sts TCCR3B, temp      ; Configuro el timer con fclk

    sbi porta, 5        ; Fijo la direccion del paso a paso

    ldi temp, (1<<OCIE3A)   ; Hablito la interrupcion
    sts TIMSK3, temp


    ret
;-----------------------------------------------------------------------------------------------;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;TIMER 4;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;-----------------------------------------------------------------------------------------------;

Timer_Motor_init2:

    ldi temp, 0x00
    sts TCNT4L, temp
```

```
488     sts TCNT4H, temp      ; El timer comienza con 0000
489
490     ldi temp, (1<<CS40)
491     sts TCCR4B, temp      ; Configuro el timer con fclk
492
493     ldi temp, (1<<OCIE4A)   ; Hablito la interrupcion
494     sts TIMSK4, temp
495
496
497     ret
498 ;-----------------------------------------------------------------------------------------------;
499 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
500 ;-----------------------------------------------------------------------------------------------;
501
502 Port_init:              ; Inicializo puertos varios         ;
503
504     sbi DDRB, 7         ; Pongo el pin 7 del puerto B como salida (es el led)
505     sbi DDRA, 0         ; Pongo el pin 0 del puerto A como salida (rele)
506     sbi DDRA, 5         ; Pongo el pin 5 del puerto H como salida (paso a paso)
507     sbi DDRA, 6         ; Pongo el pin 6 del puerto H como salida (paso a paso)
508
509
510     ret
511 ;-----------------------------------------------------------------------------------------------;
512 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
513 ;-----------------------------------------------------------------------------------------------;
514 Var_init:             ; Inicializo variables varias
515
516     ldi receive1, 0       ; Byte 1 de temperatura
517     ldi receive2, 0       ; Byte 2 de temperatura
518     ldi temp, 0         ; Registro auziliar
519     ldi cont1, 243        ; Contador para delay
520
521
522     ret
```

6.1.1. Estrategias adoptadas

Como primera consideración se utilizaron variables de estado para reflejar la próxima acción a realizar por la pochoclera. Cierta medida fue tomada a raíz de simplificar la rutina de los timers, debido a que al realizar múltiples llamados a funciones dentro de sus rutinas, podrían generar un mal funcionamiento entre los mismos.Estos estados se detallan a continucacion:

- Estado 0: Pochoclera apagada.
- Estado 1: Pochoclera prendida.
- Estado 2: Resistencia apagada: Se pasó el máximo valor permitido de temperatura.
- Estado 3: Resistencia prendida: La temperatura está por debajo del mínimo valor permitido, entonces se debe calentar nuevamente.
- Estado 4: Leer temperatura: Si se activó es porque es tiempo de leer temperatura (se activa mediante timer).
- Estado 5: Temperatura leída: Luego de leer un valor se setea este estado.

Se hizo uso de tres timers de 16-bits y sus correspondientes interrupciones por $OutputCompare$ del microcontrolador $Atmega2560$. Estas interrupciones se disparan cuando el contador del timer llega a su máximo valor. Los timers utilizados se especifican a continuación en conjunto con su tarea a realizar:

- Timer 1: Se utilizó para controlar el tiempo entre dos mediciones consecutivas de temperatura. La rutina de interrupción asociada a este se encarga de cambiar el estado a "Leer temperatura"(Estado == 4) con el fin de que, en donde corresponda, se verifique el mismo y se realice la acción indicada.
- Timer 3: Es el encargado de generar un flanco de subida en el pin correspondiente al paso del motor.
- Timer 4: Es el encargado de generar un flanco de bajada que completa el pulso necesario para generar el paso del motor.

Además a la hora de recibir un dato, a modo de visualizarlo en la aplicación del smartphone se creo la rutina $Temperature\_extract$ la cual recibe el valor otorgado por la termocupla y realiza una serie de corrimientos de manera tal de quedarnos con los bits mas significativos en un total de 1 byte representado la temperatura en el rango de 0 a 255.

7. Resultados

La realización del trabajo practico final tuvo resultados satisfactorios pero llevo a muchas complicaciones a lo largo de la cursada. Entre los problemas que tuvimos se puede detallar la rotura del driver $Pololu$ debido a una mala conexión, la mala configuración de las bobinas del motor y por último tuvimos problemas para coordinar bien el tiempo de los timers a la hora de mantener el motor prendido y levantar los valores de temperatura proporcionados por la termocupla.

7.1. Futuras mejoras del trabajo practico

En consecuencia con los resultados y los problemas obtenidos, se nos hizo posible imaginar ciertas mejoras para evitar estos problemas futuros y hacer de la maquina para hacer pochoclos un proyecto mas eficiente y robusto. Entre las mejoras se encuentran:

- La soldadura de todas las conexiones, ya que era común que hubiera problema por algún cable mal conectado o falso contacto.
- Opción de uso no remoto: Es decir, que pueda utilizarse el dispositivo sin necesidad de conectarlo a un smartphone.
- Estética y mecánica: Se podría achicar la estructura, mejorar el eje para que tenga un giro mas suave, facilitar la extracción de la cacerola para su correcta limpieza, agregarle luces.
- Desarrollar un sistema que apague automáticamente la pochoclera cuándo todos los maíces hayan explotado.
- Centralizar las alimentaciones: Cuando se realizo la prueba ante los docentes del curso se utilizaron las fuentes de tensión del laboratorio, esto limita el uso domestico. Podrían colocarse reguladores de tensión para poder proveer energía a todos los periféricos.
- Independizarse del Arduino: Se podría programar el microcontrolador por separado e integrarlo a una placa junto al resto de componentes.
- Agregar contenedores/dispensers de azucar, sal y aceite que se activen desde el celular o via manual para que el usuario pueda elegir la cantidad a gusto.

8. Conclusiones

Podemos concluir que si bien la idea original del proyecto era mas ambiciosa, ya que inicialmente se quería agregar dispenser's de azúcar, sal y aceite para su completa condimentación, y que por cuestiones de falta tiempo se debieron acotar las posibilidades, fue de gran utilidad para plasmar y reflejar los contenidos aprendidos durante el desarrollo de la cursada, además de dimensionar el trabajo que supuso realizar el proyecto.
Por otro lado, pudimos ver las complicaciones del lenguaje $Assembler$ contrastándolo con los códigos del lenguaje Arduino, aunque destacamos que su uso permito un entendimiento a mas bajo nivel del microcontrolador.
La realización del trabajo practico nos permitió visualizar lo escrito en el código de forma tangible, es decir, nos hizo crear un proyecto que fue mas allá de lineas de código, ya que pudimos verlo funcionar a lo largo de la cursada. Si bien hubo complicaciones como las mencionadas anteriormente, se considera que este trabajo practico fue fructífero para el desarrollo tanto de la materia como de la carrera.

9. Apéndice

9.1. Enlaces consultados

https://appinventor.mit.edu/
https://www.pololu.com/product/1182
https://proyectoarduino.com/arduino-mega-2560/