Fiuba
Facultad de Ingeniería
Universidad de Buenos Aires

(6609) LABORATORIO DE MICROCOMPUTADORAS

Proyecto: *"Placa de aprendizaje"*

Profesor:	Ing. Guillermo Campiglio
Cuatrimestre / Año:	**1ro 2016**
Turno de clases prácticas:	**Miércoles, 19:00 a 22:00 hs**
Jefe de Trabajos Prácticos:	**Ing. Ricardo Arias**
Docente guía:	
Colaborador externo:	Ing. Pablo Marino (6602)

Autor:	**Seguimiento del proyecto**								
Alan Decurnex (85409)									

Observaciones:

Fecha de aprobación		

Firma J.T.P.

COLOQUIO	
Nota final	
Firma Profesor	

Índice

Contenido

Introducción 2
Planteo del proyecto 2
Definición de umbrales y reglas de control básicas 2
Interfaces de conexión 2
Construcción ad hoc de diferentes partes 2
Resumen de los diferentes sensados y controles: 3
Plataforma 3
Desarrollo 4
Especificaciones: 4
Sensado de tensión: 4
Sensado de temperatura: 5
Conexión con computadora: 7
Programador: 7
Diagrama en bloques 9
Esquemático 10
Diagrama de flujo 12
Código fuente documentado 13
Resultados obtenidos 21
Conclusiones 22
Hojas de datos 23

Introducción

Se realizó un dispositivo didáctico demostrativo de sensado y control de magnitudes de tensión y temperatura, las mismas son controladas por un microcontrolador AVR validando que se encuentren dentro de parámetros preestablecidos. A su vez cada medición dispone de una salida actuadora que se podrá conectar según lo que se requiera. El circuito se conecta a una computadora mediante conexión serie y se puede visualizar en pantalla los valores medidos.

Este proyecto servirá para demostrar una aplicación concreta de un microcontrolador a los estudiantes de ingeniería que cursen la materia Laboratorio (66.02), abarcando tareas básicas de medición de distintas magnitudes que brindan una introducción cualitativa al concepto a modo de motivación y representa una base para desarrollar futuros proyectos similares de medición y control.

Planteo del proyecto

El presente proyecto implementa un medidor de 8 tensiones analógicas del puerto F. Se realizaron dos mediciones demostrativas, la primera es una medición de tensión sobre una placa de prueba que emula un circuito genérico ya que tiene una salida variable (configurable manualmente) entre 0 y 5 V, esta medición permite ver como a medida que se varia la entrada se mide correctamente con el microcontrolador y se visualiza en la computadora, además dispone de un indicador que me informa mediante un led si la tensión medida se encuentra dentro de un rango de valores (ventana de tensión), para demostrar una posible acción frente a la medición obtenida. Debido a la naturaleza de la placa de prueba a sensar, se medirán en principio tensiones de hasta 5 V de continua.

La segunda medición consta de un circuito de prueba compuesto por un componente dañado al que se le conecta un sensor que informa la temperatura actual del componente y en caso de superar cierto límite acciona un relé que alimenta un circuito actuador, en este caso se optó por conectar un ventilador para observar cómo responde el circuito frente a un incremento o disminución de la temperatura.

Definición de umbrales y reglas de control básicas

Para los parámetros a sensar se propone establecer rangos y acciones o reglas pre-establecidas a seguir en base al sensado. Esto permite simular situaciones en los cuales se tienen una variable que se va fuera de un determinado rango y el microcontrolador rápidamente debe alertar y/o tomar una decisión (activando o desactivando una salida conectada a algún actuador).

Interfaces de conexión

Conexión con computadora a través de RS232, para visualización de las mediciones obtenidas.

Construcción ad hoc de diferentes partes

1. Placa con salida de tensión variable entre 0 y 5 V.
   Fuente regulada de 5 V (alimentación del punto 1)
2. Fuente regulada de 5 V con un 7805 intercambiable, bajo prueba de sensado de temperatura.

Circuito con relé actuador para la salida del circuito anterior, mediante un ventilador disminuye la temperatura.

Resumen de los diferentes sensados y controles:

1. Sensado analógico DC: tensión continua. Se medirá la tensión de salida del circuito 1 mediante el ADC del uC y se validara que se encuentre dentro de un umbral seguro (límite superior e inferior), encendiendo un led en caso de encontrarse fuera del mismo.
2. Sensado analógico DC: tensión continua. Se medirá la tensión de salida de un sensor de temperatura y se interpretara su valor, en caso de superar un límite preestablecido, se accionara con un ventilador hasta volver a los valores seguros. Este circuito se manejara con un relé.

Plataforma

Luego de investigar características de varios microcontroladores del mercado, y debido al proyecto en particular, se decidio utilizar uno que tenga una cantidad importante de entradas analógicas para poder luego escalar el proyecto a mas mediciones según lo que se requiera. No hay grandes restricciones debido a las demás funcionalidades. Se obtuvo para este fin el microcontrolador ATmega2560[1] (provisto por la plataforma Arduino MEGA[2]), que cuenta con 16 entradas analógicas, entre otras especificaciones.

[1] Datasheet ATmega2560: http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf
[2] Caracteristicas Arduino MEGA: https://www.arduino.cc/en/Main/ArduinoBoardMega2560

Desarrollo

Especificaciones:

Se miden tensiones de 0 hasta 5 V de continua por los 8 pines del puerto F, El ADC interno del microcontrolador es de 10 bits, con lo cual se tiene una resolución: Vcc/1024. En nuestro caso 5V / 1024 = 5 mV aproximadamente.

Para las mediciones se debe unir las referencias de tensión de todos los circuitos.

Sensado de tensión:

Esta parte consta de dos elementos, el primero es una fuente regulada de 5 V (alimentada por un transformador genérico de 12V DC) implementada mediante un LM7805 junto con sus respectivos filtros y led indicador de funcionamiento, y el segundo es un circuito divisor resistivo compuesto por dos potenciómetros de 1 MΩ que permiten obtener a la salida una tensión configurable manualmente entre cero y la tensión de alimentación, en este caso entre 0 y 5 V.

La placa divisora resistiva tiene una doble finalidad, en principio en la presentación didáctica del proyecto permite variar la tensión, visualizándola en la pantalla y pudiendo corroborarla mediante un voltímetro conectado en serie, a su vez se visualiza el led indicador de ventana a medida que variamos la tensión. Luego puede utilizarse para poder medir tensiones superiores a 5 V, escalando estas tensiones a los 5 V máximos de la entrada del microcontrolador[3] (en este caso la resolución disminuirá).

En esta última placa se optó por agregar en serie a los dos potenciómetros una resistencia pequeña de 330 Ω para limitar la intensidad de corriente a aproximadamente 15 mA, en caso de cortocircuitar ambos potenciómetros (el circuito de alimentación también posee un led de control, pero se optó por agregar esta resistencia de todas formas para preveer el caso de querer alimentar el circuito directamente con el micro).

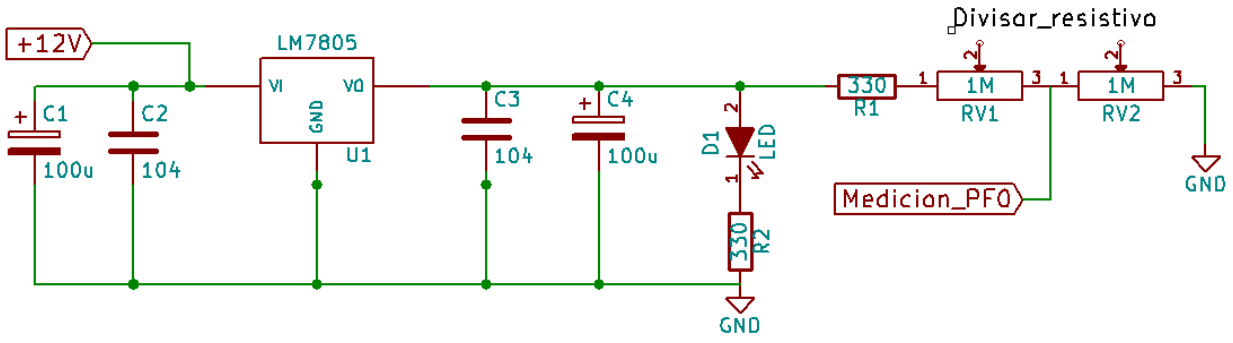


Ilustración 1 - Esquemático de la primera medición.

[3] Para configurarlo conectamos el valor máximo a medir en la entrada del divisor y variamos los potenciómetros hasta que en la salida (midiendo mediante un voltímetro) obtengamos 5V, luego de esa calibración podemos conectarlo al microcontrolador.

Para la alimentación del circuito se utiliza un transformador de 12V DC. En la etapa del regulador de tensión 7805 se utilizaron componentes recomendados de un circuito clásico implementado con este integrado, filtros a la entrada y salida, y un led que indica que está dando tensión a la salida, con su correspondiente resistor. Por último, para la etapa del divisor resistivo se utilizaron dos potenciómetros de 1MΩ, si RV2 está en corto y RV1 en el máximo, tendremos a la salida 0V, y a la inversa tendremos aproximadamente 5V a la salida. Se agregó un resistor de 330Ω para proteger el micro en caso de poner en corto ambos potenciómetros, que limita la intensidad de corriente máxima a aproximadamente 15 mA.

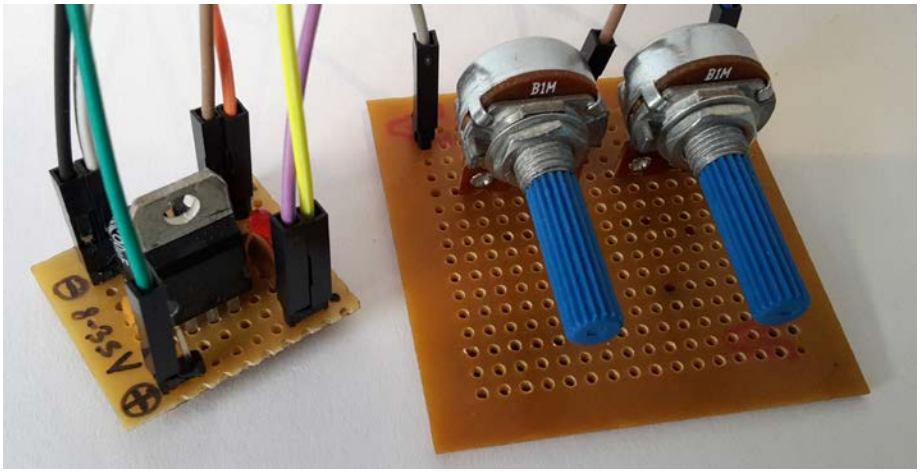


Foto 1 - Circuito de la primera medición.

Para este sensado se utilizó la primera entrada analógica del microcontrolador (marcada como A0 en la plataforma arduino, corresponde al pin 0 del puerto F).

La salida de esta medición es a través del pin 7 del puerto B, que posee un led indicador incluido en la plataforma de Arduino.

Si el valor medido se encuentra afuera de una ventana de control se activa la salida, visualizando de manera directa el resultado de la ventana de control. Por tanto, este circuito me permite variar los potenciómetros obteniendo una señal entre 0 y 5 V, visualizar dicha señal en la pantalla del ordenador y ver el led indicador de estado. Ventana utilizada: (1,7 ; 3,3) V

El divisor resistivo puede utilizarse también para escalar la entrada y así poder medir tensiones superiores a los 5 V que maneja el microcontrolador, claro que en este caso disminuiría la resolución puesto que el ADC es fijo.

Sensado de temperatura:

El circuito para esta medición está compuesto por una fuente regulada de 5 V implementada también con un 7805 pero esta vez intercambiable, bajo prueba de sensado de temperatura. Se utiliza el sensor LM35 cuya salida es de 0 mV + 10mV / °C.

Por otro lado posee un relé conectado a la salida (actuador) de la medición que activa un ventilador que disminuye la temperatura. El ventilador es alimentado por una fuente genérica de 12V.

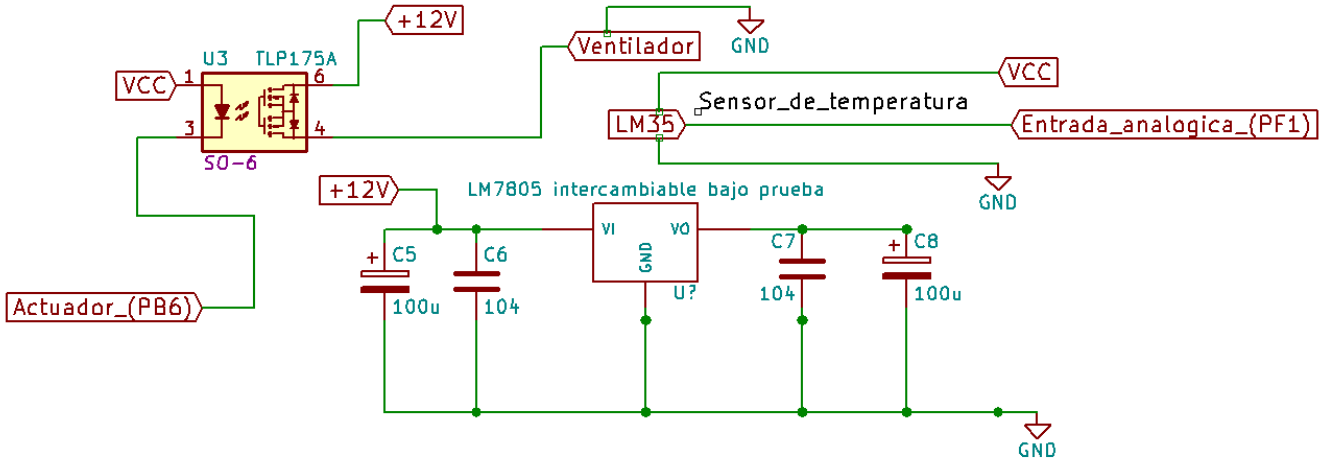


Ilustración 2 - Esquemático de la segunda medición.

Para la parte del relé se obtuvo un módulo que incluye dos leds (uno indicador de alimentación correcta y otro indicador del estado del relé) que se activa por nivel bajo. Para la alimentación del circuito se utiliza un transformador de 12V DC. Y la etapa del regulador es idéntica a la medición anterior.

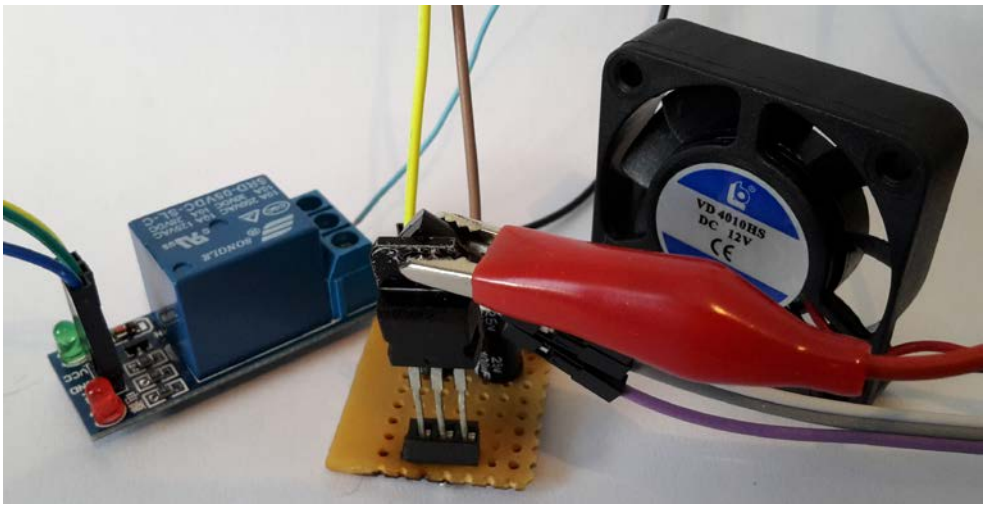


Foto 2 - Circuito de la segunda medición.

Para este sensado se utilizó la segunda entrada analógica del microcontrolador (marcada como A1 en la plataforma Arduino, corresponde al pin 1 del puerto F).

La salida de esta medición es a través del pin 6 del puerto B, que se conecta a un relé.

Actuador en forma de ventana de histéresis. Si el valor medido supera cierto límite superior se activa la salida y esta enciende el ventilador, que permanecerá encendido hasta que el valor disminuya a menos de otro límite inferior.

Ventana utilizada: (24,92 ; 26,39) °C (se utilizó una ventana sensible configurada el día de la presentación para poder demostrar el funcionamiento de la histéresis)

Conexión con computadora:

La conexión a la computadora se realizó por medio de las salidas serie del microcontrolador (RX y TX), conectadas invertidas (RX → TX, TX → RX) a un conversor serie-USB desarrollado con el integrado CP2102. Se configuro el Baud Rate a 250k.

Desde la computadora se recibe la información mediante un programa desarrollado en Python, que simplemente visualiza las mediciones por pantalla.

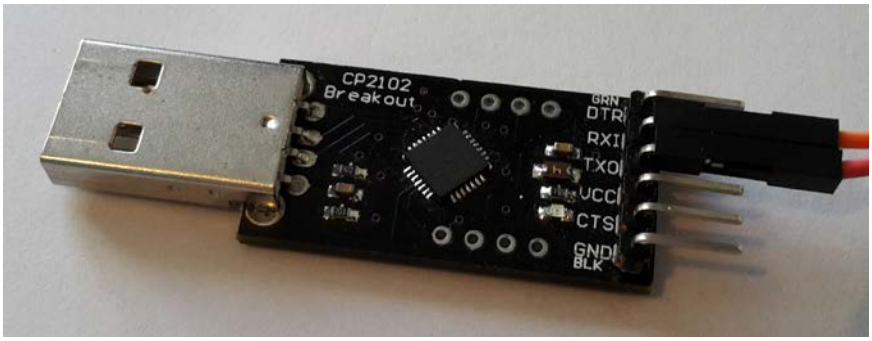


Foto 3 - Placa de conexion serie-USB

Programador:

Como programador para el microcontrolador se utilizó el provisto por el LABI para el Club de Robótica de nuestra facultad [4]. Me entregaron todos los componentes para poder soldarlo sin ningún inconveniente.

Para poder quemar el código dentro del micro se utilizó el programa avrdude v6.1.
Se modificaron los FUSES originales de la plataforma Arduino, y se eliminó el Bootloader, para poder utilizar el micro solo.

FUSES actuales del micro:
Low Fuse 0xFF
High Fuse 0xD8

[4] http://labi.fi.uba.ar/labi/programador

Extended Fuse 0xFF

Comando para ver por pantalla los FUSES actuales:
avrdude.exe -c usbtiny -p m2560 -U efuse:r:-:h -U hfuse:r:-:h -U lfuse:r:-:h

Comando para modificar los FUSES:
avrdude -c usbasp -p m2560 -P usb -b 115200 -e -u -U lock:w:0x3F:m -U lfuse:w:0xFF:m -U hfuse:w:0xD8:m -U efuse:w:0xFF:m

Comando para grabar programa en el micro:
avrdude.exe -c usbtiny -p m2560 -U flash:w:Ejercicio1.hex:i

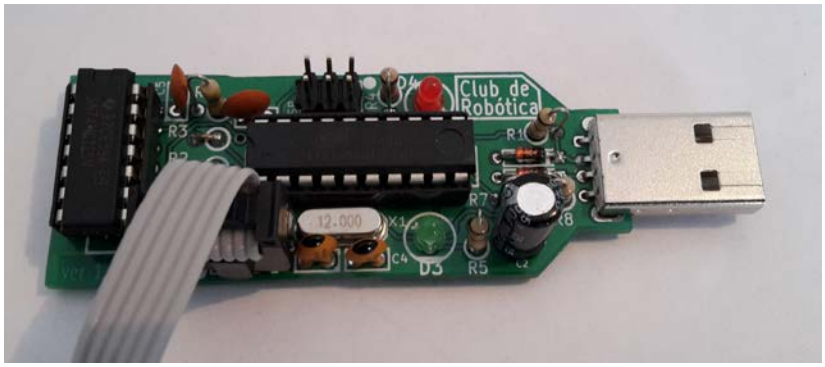


Foto 4 - Programador terminado.

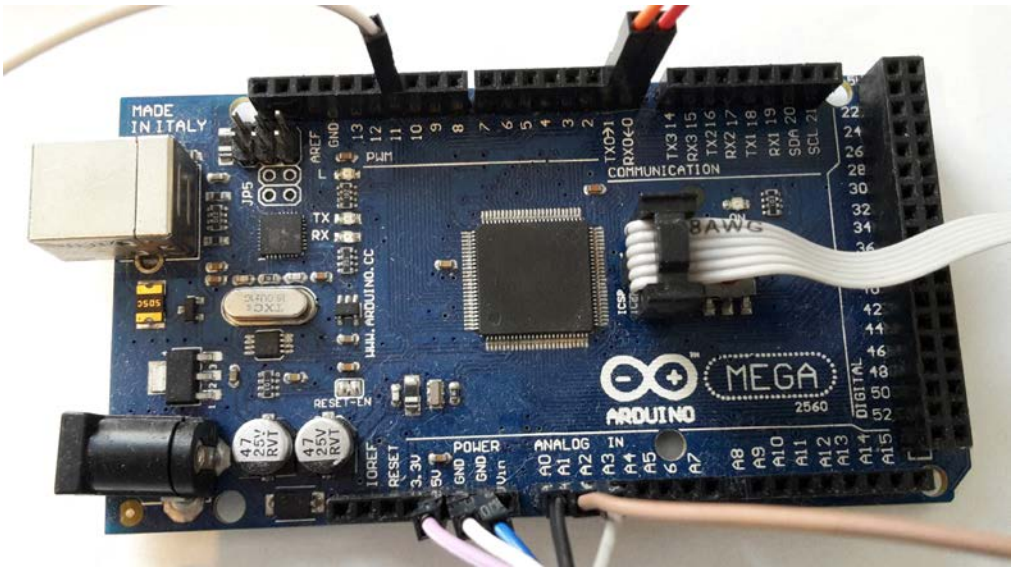
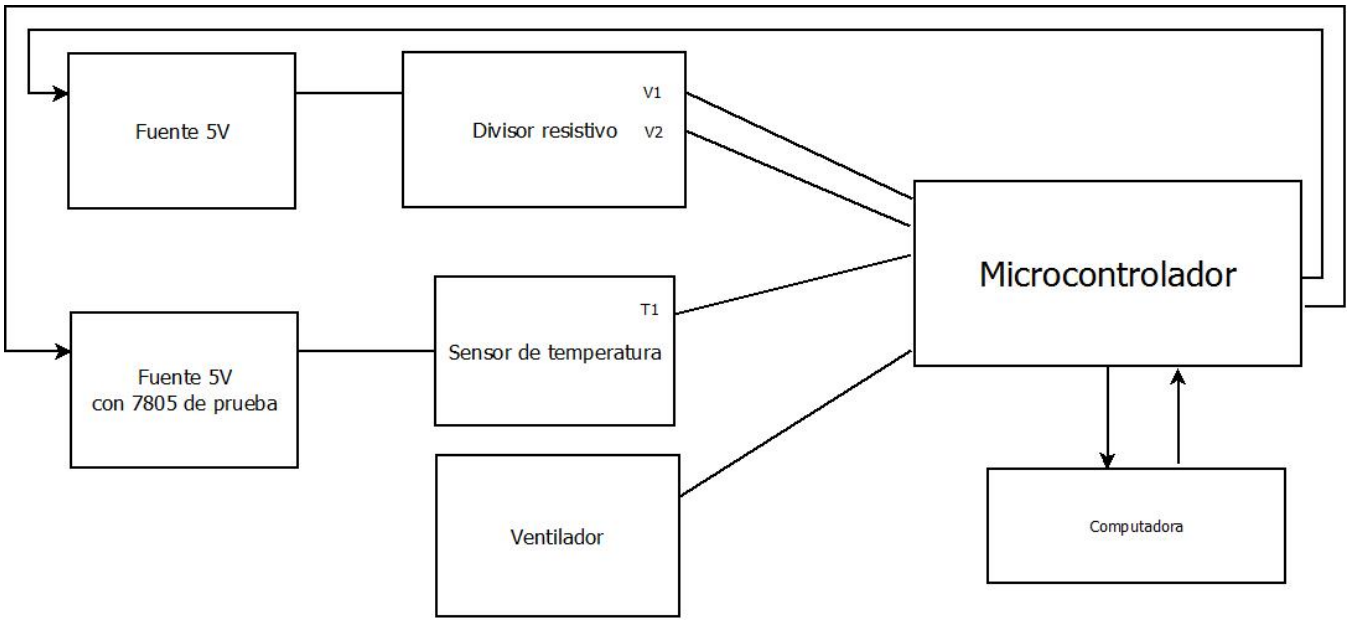


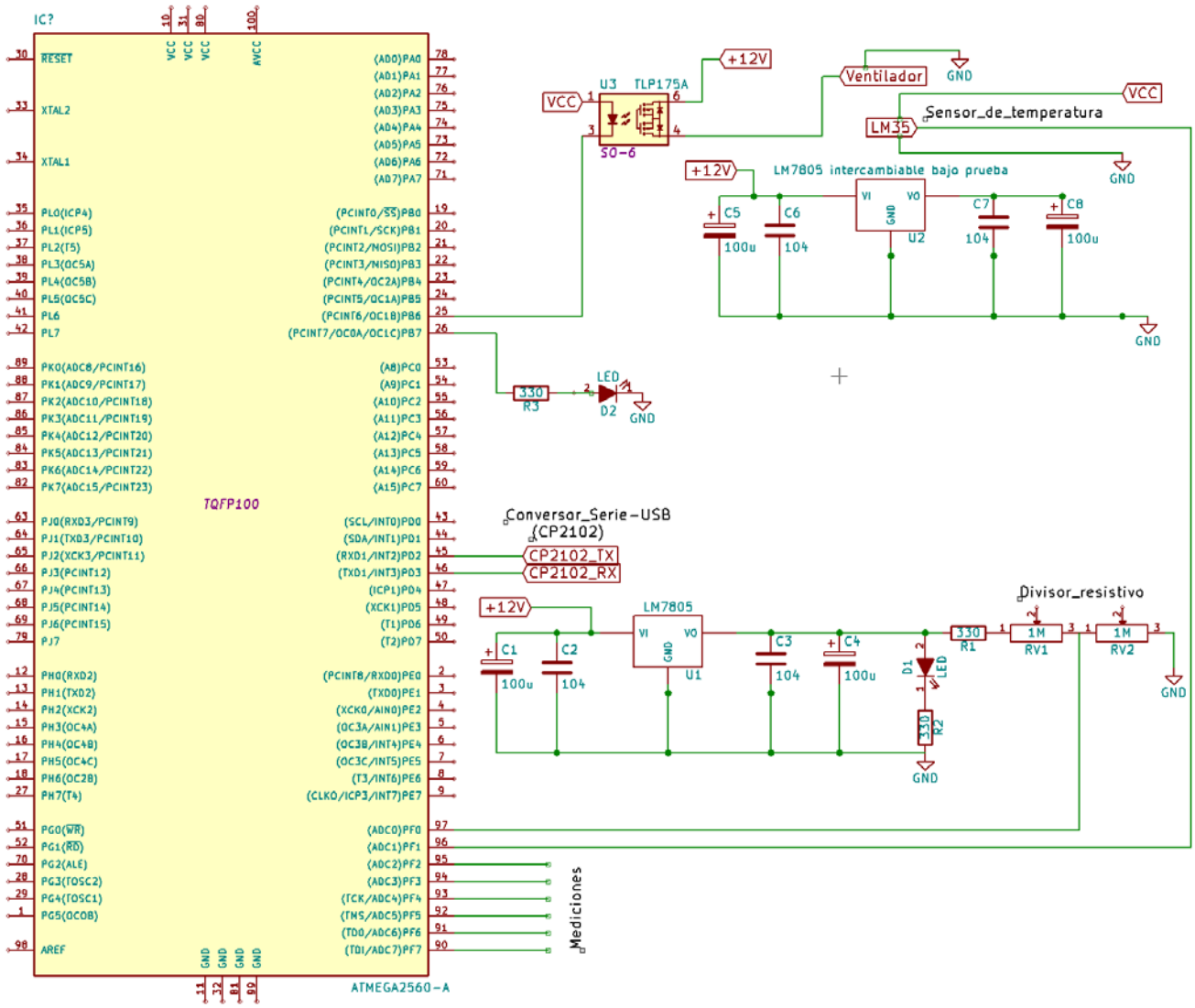
Foto 5 - Placa de Arduino MEGA utilizada para el proyecto

Diagrama en bloques



Esquemático

En principio, presentamos el esquemático del circuito de medición realizado en KiCad, luego se observa el esquemático de la placa de Arduino, que incluye la posibilidad de alimentarse por USB o a través del programador, además posee un led incorporado en la salida alta del puerto B, utilizado para el actuador de la primera medición. También se utilizó el botón de reset con su configuración, y los capacitores necesarios para el funcionamiento del microcontrolador y el ADC, el resto de la plataforma Arduino no se utilizó. La alimentación del micro es a través del programador.



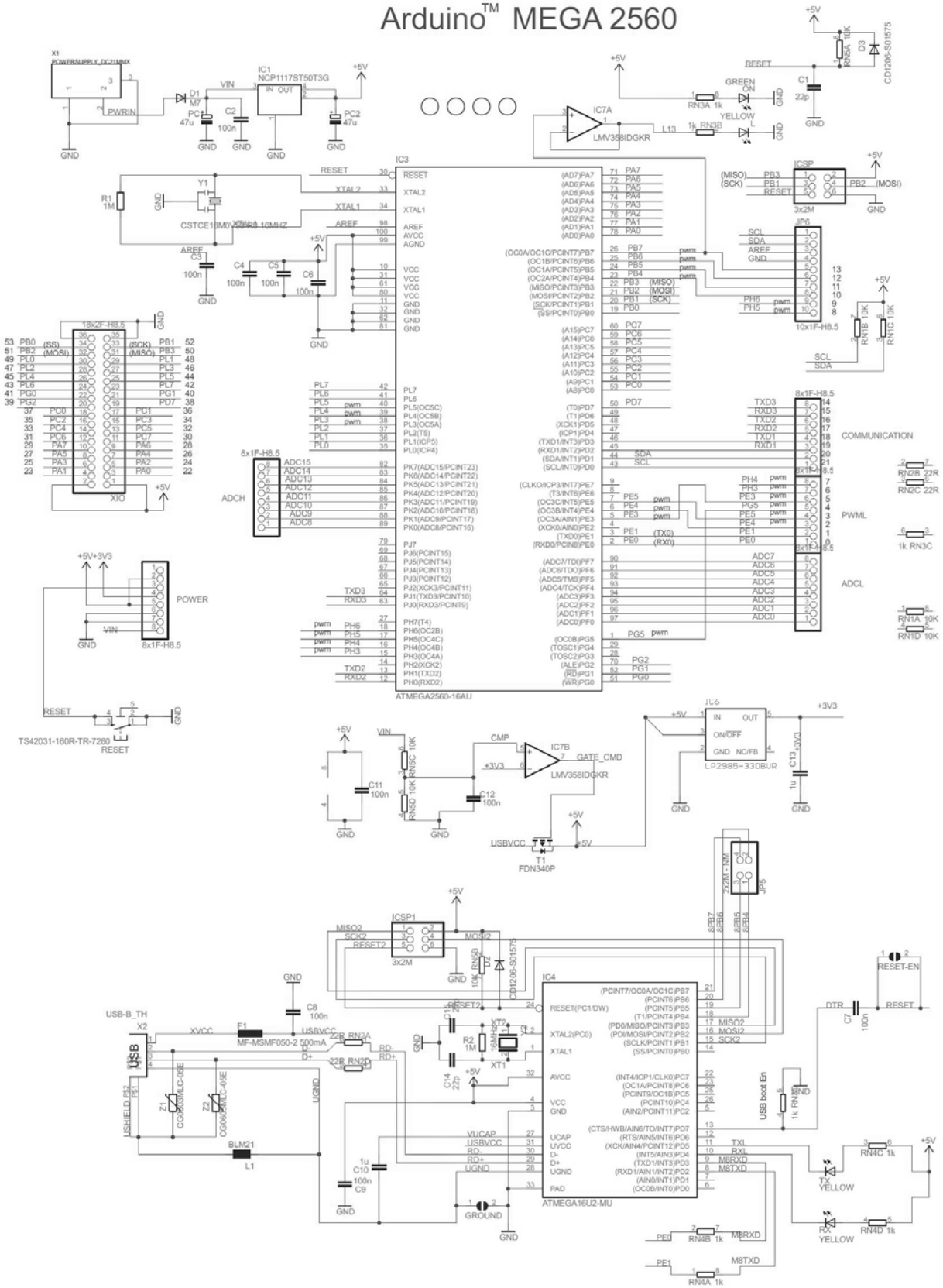
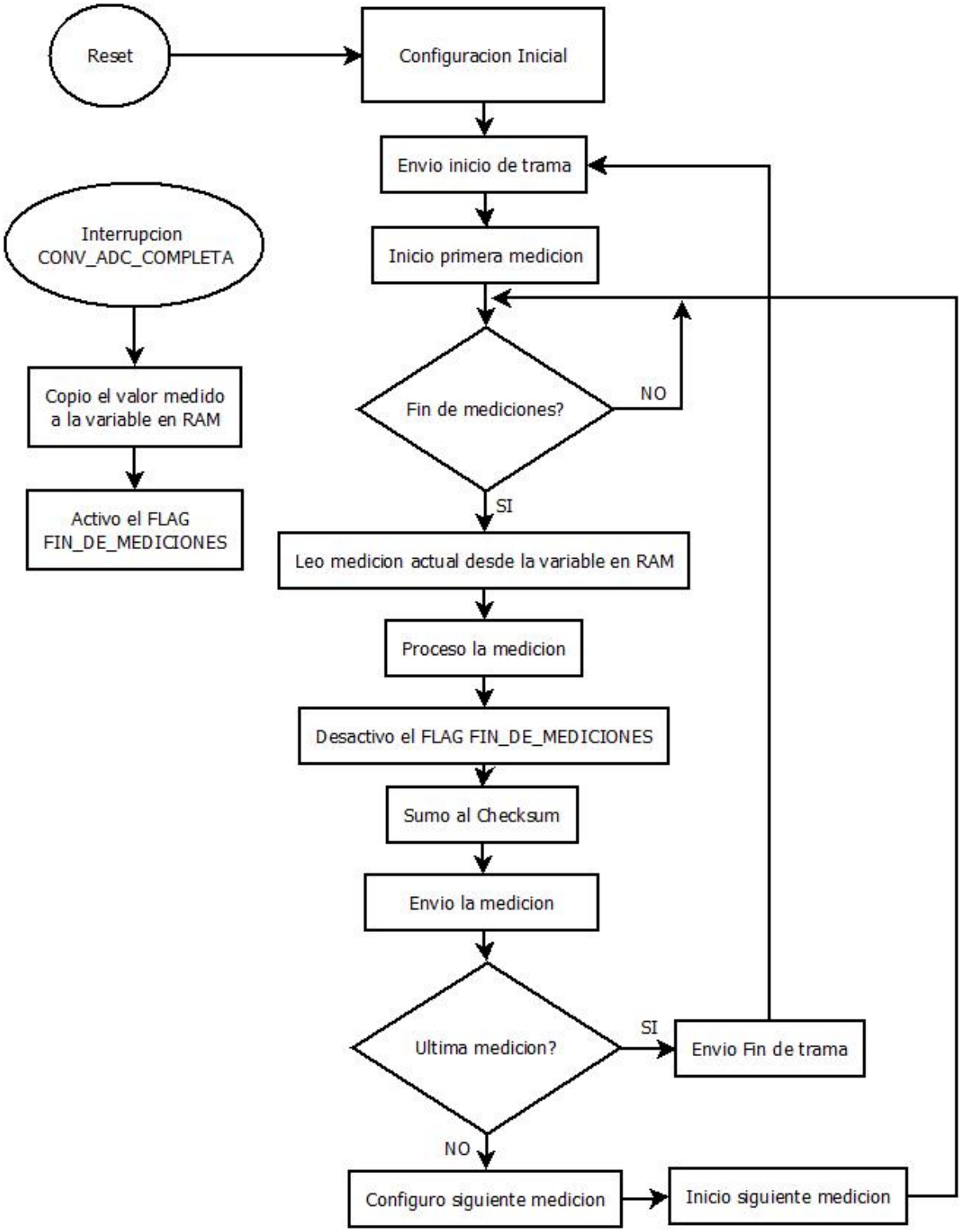

Arduino™ MEGA 2560
ATMEGA2560-16AU
ATMEGA16U2-MU
IC3
IC4
IC1
NCP1117ST50T3G
IC7A
IC7B
LMV358IDGKR
LP2985-330BUR
CSTCE16M0V53-R0
TS42031-160R-TR-7260
FDN340P
MF-MSMF050-2
CD1206-S01575
10x1F-H8.5
8x1F-H8.5
18X2F-H8.5
3x2M
USB-B_TH
BLM21
COMMUNICATION
PWML
ADCL
POWER
RESET-EN
GROUND
USB boot En

Diagrama de flujo



Código fuente documentado

```
/*
 * Placa-de-aprendizaje.asm
 * Trabajo practico de Laboratorio de microprocesadores 86.07
 * 1er cuatrimestre de 2016 - FIUBA
 * Autor: Alan Decurnex (85409)
 */

.INCLUDE "m2560def.inc"                //Incluye definición archivos ATMEGA2560

; ***** DEFINICION DE CONSTANTES ************************************************************
; se mide desde ADMUX_INICIO hasta ADMUX_FIN, 8 pines en total si mido desde A0 hasta A7
.equ   ADMUX_INICIO = 0x00 ; (codigo implememtado para 5 bits de MUX) desde A0
.equ   ADMUX_FIN      = 0x07 ; hasta A7

.equ   MASCARA_ADMUX = 0x17 ; mascara para leer los 5 bits del MUX
.equ   INICIO_DE_TRAMA = 0x5A ; inicio de trama del codigo de checksum

; constantes para la medicion desde A0
.equ   A0_limite_superior_H = 0x02 ; superior 3.3 V, 675, 0x02A3
.equ   A0_limite_superior_L = 0xA3
.equ   A0_limite_inferior_H = 0x01 ; inferior 1.7 V, 348, 0x015C
.equ   A0_limite_inferior_L = 0x5C

; constantes para la medicion desde A1
.equ   A1_limite_superior_H = 0x00 ; superior 0x0036; decimal 54; 26,39 grados centigrados
.equ   A1_limite_superior_L = 0x36 ;
(54*5*100/1023)
.equ   A1_limite_inferior_H = 0x00 ; inferior 0x0033; decimal 51; 24,92 grados centigrados
.equ   A1_limite_inferior_L = 0x33 ;
(51*5*100/1023)

; constantes para la medicion desde A2
.equ   A2_maximo_H = 0x00 ;  3.3 V, 675, 0x02A3
.equ   A2_maximo_L = 0x33


; ***** DEFINICION DE REGISTROS *************************************************************
.def   checksum      = r24; Cyclic Redundancy Check

; ESTADO - registro encargado de registrar el estado actual del programa mediante flags
.def   ESTADO = r25
; bit 0 = Flag que se activa al final de las mediciones

; ***** BIT DEFINITIONS *********************************************************************
.equ   FIN_DE_MEDICIONES    = 0    ; Flag que se activa al final de las mediciones

; definiciones de los puertos de los actuadores de las mediciones
; salen todas por el puerto B

.equ   Pin_A0 = 7 ; se plantea de forma inversa para aprovechar el led del arduino
.equ   Pin_A1 = 6
.equ   Pin_A2 = 5
.equ   Pin_A3 = 4
.equ   Pin_A4 = 3
```

```
.equ   Pin_A5 = 2
.equ   Pin_A6 = 1
.equ   Pin_A7 = 0

.equ   PORT_A0 = PORTB
.equ   PORT_A1 = PORTB
.equ   PORT_A2 = PORTB
.equ   PORT_A3 = PORTB
.equ   PORT_A4 = PORTB
.equ   PORT_A5 = PORTB
.equ   PORT_A6 = PORTB
.equ   PORT_A7 = PORTB

.equ   DDR_A0 = DDRB
.equ   DDR_A1 = DDRB
.equ   DDR_A2 = DDRB
.equ   DDR_A3 = DDRB
.equ   DDR_A4 = DDRB
.equ   DDR_A5 = DDRB
.equ   DDR_A6 = DDRB
.equ   DDR_A7 = DDRB


; ***** PRINCIPAL ***************************************************************************
;Se abre un segmento de datos para definir variables
.dseg
.org   SRAM_START // 0x200

; Definición de variables en la zona de memoria de uso general
muestreo:     .byte  2 ; Variable que guarda el muestreo de la medicion actual (2 bytes)

;Se abre un segmente de código flash para escribir instrucciones o tablas.
.cseg
.ORG $00
RJMP RESET                  // Reset

; interrupcion por conversion ADC completa
.org   ADCCaddr             ; definido en m2560def.inc (0x003a)
rjmp   ISR_CONV_ADC_COMPLETA

.ORG INT_VECTORS_SIZE // salteo el vector de interrupciones

RESET:
       // mueve el stack pointer al final de la RAM para maximizar el espacio de STACK
       LDI R16,LOW(RAMEND)
       OUT SPL,R16
       LDI R16,HIGH(RAMEND)
       OUT SPH,R16

       rcall inicializar_puertos
       rcall inicializar_USART
       clr ESTADO ; inicializa estado

reinicio_mediciones:
       ldi checksum, INICIO_DE_TRAMA ; sumo el inicio de trama al checksum

       ldi r16, INICIO_DE_TRAMA
```

```
	rcall USART_Transmit ; transmito el inicio de trama
	rcall ADC_Primera_Medicion ; configuro y comienzo primera medicion

principal:
	sbrs ESTADO, FIN_DE_MEDICIONES
	rjmp principal

	; leo la medicion actual
	lds		r20, muestreo
	lds		r16, muestreo+1

	rcall procesar_medicion ; proceso la medicion

	andi ESTADO, ~(1<<FIN_DE_MEDICIONES) ; limpio el flag de FIN_DE_MEDICIONES

	; sumo la medicion actual al checksum
	add checksum, r20
	add checksum, r16

	; envio la medicion actual
	rcall USART_Transmit
	mov r16, r20
	rcall USART_Transmit

	; me fijo si es la ultima medicion
	lds    r16,ADMUX
	ldi r17, MASCARA_ADMUX
	and r17, r16 ; hago una mascara para solo quedarme con el MUX4:0
	cpi r17, ADMUX_FIN   ; ultimo pin a leer
	breq enviar_fin_de_trama

	inc r16                    ; convierto desde el siguiente pin
	sts    ADMUX,r16     ; escribe reg. ADMUX de configuración del ADC

	; envio a medir la siguiente conversion
	lds    r16,ADCSRA
	ori r16,(1<<ADSC); escribo ADSC en uno para iniciar siguiente conversion.
	sts ADCSRA,r16

	rjmp principal

enviar_fin_de_trama:
	neg checksum
	mov r16, checksum
	rcall USART_Transmit
	rjmp reinicio_mediciones

; inicializa los puertos utilizados, el puerto F como entradas analogicas y el B como salidas de
los actuadores, con todas sus salidas en cero.
inicializar_puertos:
	ldi r16,0x00
	; se mide desde el puerto F
	out DDRF,r16 ;configura todo el puerto F como entradas (analogicas)

	; control de salidas correspondientes a los valores medidos
	ldi r16,0xFF
```

```
	; configura los pines de salida de los actuadores de las mediciones
	sbi DDR_A0, Pin_A0
	sbi DDR_A1, Pin_A1
	sbi DDR_A2, Pin_A2
	sbi DDR_A3, Pin_A3
	sbi DDR_A4, Pin_A4
	sbi DDR_A5, Pin_A5
	sbi DDR_A6, Pin_A6
	sbi DDR_A7, Pin_A7

	; coloco por defecto las salidas en cero
	cbi PORT_A0, Pin_A0
	cbi PORT_A1, Pin_A1
	cbi PORT_A2, Pin_A2
	cbi PORT_A3, Pin_A3
	cbi PORT_A4, Pin_A4
	cbi PORT_A5, Pin_A5
	cbi PORT_A6, Pin_A6
	cbi PORT_A7, Pin_A7

; rutina de inicializacion del USART, carga valores para un Baud Rate correspondiente y llama a
la rutina USART_Init
inicializar_USART:
	; configuro baud rate de 250k (ver tabla pagina 226 de la datasheet)  error = 0%
	clr r17
	ldi r16, 7
	rcall USART_Init ; inicializo USART
	ret

; rutina que configura la primera medicion del ADC segun el valor de ADMUX_INICIO, y envia a
medir.
ADC_Primera_Medicion:
	clr r26
	cli ; SREG<I>=0 interrupciones globales deshabilitadas
	; importante, la seleccion del MUX siempre debe hacerse ANTES de iniciar la conversion
	; (puede hacerse durante la conversion anterior despues de que baje ADIF)

	ldi    r16,(0<<REFS1)|(1<<REFS0)|(0<<ADLAR)|ADMUX_INICIO ; ADMUX son los bits bajos, 4:0
		; (0<<REFS1)|(1<<REFS0): Referencia interna AVCC (5V)
		; (0<<ADLAR): Ajuste a derecha del resultado
		; (0<<MUX4)|(0<<MUX3)|(0<<MUX2)|(0<<MUX1)|(0<<MUX0): Convierto la tensión desde
PF0 (ADC0)
	sts    ADMUX,r16     ; escribe reg. ADMUX de configuración del ADC

	ldi    r16,(0<<MUX5)|(0<<ADTS2)|(0<<ADTS1)|(0<<ADTS0) ; los demas bits los dejo en cero
		; (0<<MUX5): para los canales F0:7 debe valer 0, de F8:15 debe valer 1
		; (0<<ADTS2)|(0<<ADTS1)|(0<<ADTS0): Free runing
	sts    ADCSRB,r16    ; escribe reg. B de configuración del ADC

	ldi
r16,(1<<ADEN)|(1<<ADSC)|(0<<ADATE)|(0<<ADIF)|(1<<ADIE)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)
		; (1<<ADEN): Habilita el ADC
		; (1<<ADSC): Pedido de 1er conversión (inicia)
		; (0<<ADATE):  permite el auto triggering, por ejemplo: conversion continua
		; (si esta en cero solo inicia medicion con ADSC en 1)
		; (0<<ADIF): Limpio posible interrupcion falsa
		; (1<<ADIE): Interrumpe cada vez que termina una conversión
```

```
              ; (1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0): 16MHz/128 = 125 kHz
                     ; primera conversion en (25/125k) = 200 uSeg
                     ; siguiente conversion en (13/125k) = 104 uSeg
       sts ADCSRA,r16             ; escribe reg. A de configuración del ADC / se inicia primera
conversion

       sei ; habilito interrupciones globales
       ret

; procesa la medicion actual
procesar_medicion:

       ; me fijo cual medicion es
       lds    r18,ADMUX
       ldi r17, MASCARA_ADMUX
       and r17, r18 ; hago una mascara para solo quedarme con el MUX4:0

       ; switch
       cpi r17, 0
       breq procesar_A0
       cpi r17, 1
       breq procesar_A1
       cpi r17, 2
       breq procesar_A2
       cpi r17, 3
       breq procesar_A3
       cpi r17, 4
       breq procesar_A4
       cpi r17, 5
       breq procesar_A5
       cpi r17, 6
       breq procesar_A6
       cpi r17, 7
       breq procesar_A7
       rjmp default

procesar_A0:
       ; ventana = [A0_limite_inferior, A0_limite_superior]
       ; si el valor no esta incluido en la ventana, habilita la salida.

       ; comparo con limite inferior
       ldi r17, A0_limite_inferior_H ; cargo en r17 el valor alto para poder usar cpc
       cpi r20, A0_limite_inferior_L ; comparo los bits bajos
       cpc r16, r17 ; comparo los bits altos
       brlo A0_afuera_de_ventana ; afuera por limite inferior

       ; comparo con limite superior
       ldi r17, A0_limite_superior_H
       cpi r20, A0_limite_superior_L ; comparo los bits bajos
       cpc r16, r17 ; comparo los bits altos
       brlo A0_adentro_de_ventana
       breq A0_adentro_de_ventana
       ; afuera por limite superior

A0_afuera_de_ventana:
       sbi PORT_A0, Pin_A0 ; prende el pin correspondiente
       rjmp fin_procesar_A0
```

```
A0_adentro_de_ventana:
	cbi PORT_A0, Pin_A0 ; apago el pin correspondiente

fin_procesar_A0:
	rjmp fin_switch

procesar_A1: // procesamiento de una ventana de temperatura
	; ventana  = [A1_limite_inferior, A1_limite_superior]
	; si el valor es superior a A1_limite_superior, habilito la salida
	; si el valor es inferior a A1_limite_inferior, deshabilito la salida
	; si el valor esta incluido en la ventana, no hago nada

	; comparo con limite inferior
	ldi r17, A1_limite_inferior_H
	cpi r20, A1_limite_inferior_L ; comparo los bits bajos
	cpc r16, r17 ; comparo los bits altos
	brlo A1_menor_a_ventana ; afuera por limite inferior

	; comparo con limite superior
	ldi r17, A1_limite_superior_H
	cpi r20, A1_limite_superior_L ; comparo los bits bajos
	cpc r16, r17 ; comparo los bits altos
	brlo A1_adentro_de_ventana
	breq A1_adentro_de_ventana

	; si llega aca es porque esta afuera por limite superior

A1_mayor_a_ventana:
	sbi PORT_A1, Pin_A1 ; prende el pin correspondiente
	rjmp fin_procesar_A1

A1_menor_a_ventana:
	cbi PORT_A1, Pin_A1 ; apago el pin correspondiente
	rjmp fin_procesar_A1

A1_adentro_de_ventana: ; en este caso no hago nada

fin_procesar_A1:
	rjmp fin_switch

procesar_A2:
; si el valor medido supera el maximo, se habilita la salida

	; comparo con el maximo
	ldi r17, A2_maximo_H
	cpi r20, A2_maximo_L ; comparo los bits bajos
	cpc r16, r17 ; comparo los bits altos
	brlo A2_menor_o_igual_al_maximo ; menor al maximo
	breq A2_menor_o_igual_al_maximo ; igual al maximo
	; mayor al maximo

A2_mayor_al_maximo:
	sbi PORT_A2, Pin_A2 ; prende el pin correspondiente
	rjmp fin_procesar_A2

A2_menor_o_igual_al_maximo:
```

```
	cbi PORT_A2, Pin_A2 ; apago el pin correspondiente

fin_procesar_A2:
	rjmp fin_switch

procesar_A3:
	rjmp fin_switch
procesar_A4:
	rjmp fin_switch
procesar_A5:
	rjmp fin_switch
procesar_A6:
	rjmp fin_switch
procesar_A7:
	rjmp fin_switch
default:
	/* no hago nada*/
fin_switch:
	ret


///// IMPLEMENTACION DE FUNCIONES DE USART, segun Datasheet ///////////

; inicializa el USART
USART_Init:
	; Set baud rate
	sts UBRR0H, r17
	sts UBRR0L, r16

	ldi r16, (1<<U2X0) ; dobla la velocidad de transmision en modo asincronico
	sts UCSR0A, r16

	; Enable receiver and transmitter
	ldi r16, (1<<RXEN0)|(1<<TXEN0) ;  (0<<UCSZ02)
	sts UCSR0B, r16

	; Set frame format: 8data, 1stop bit
	ldi r16, (1<<UCSZ01)|(1<<UCSZ00); asincronico, paridad disable, 1 stop bit,
	                                 ; 8 data (011, el tercer bit esta en UCSR0B)
	sts UCSR0C, r16
	ret

; envia el dato que esta en r16 por serie
USART_Transmit:
	; Wait for empty transmit buffer
	lds r17, UCSR0A
	sbrs r17, UDRE0 ; me fijo si esta vacio el buffer de salida
	rjmp USART_Transmit

	; Put data (r16) into buffer, sends the data
	sts UDR0, r16
	ret

; lee el siguiente valor serie y lo almacena en r16
USART_Receive:
	; Wait for data to be received
	lds r17, UCSR0A
```

```
	sbrs r17, RXC0 ; se fija si llego nuevo dato
	rjmp USART_Receive

	; Get and return received data from buffer
	lds r16, UDR0
	ret

////////// FIN USART ////////////////////////////////////////////////////////////


;-----------------------------------------------------------------------
; Rutina de Servicio de Interrupción (ISR) del conversor ADC
;-----------------------------------------------------------------------
ISR_CONV_ADC_COMPLETA:
; limpia automaticamente el bit ADIF (especificado en Datasheet)
; copio el resultado de la conversión a la variable "muestreo"
	push r20
	push r21
	lds		r20,ADCL		; se lee 1ro el byte bajo
	lds		r21,ADCH		; y luego el alto

	sts		muestreo,r20
	sts		muestreo+1,r21

	ori ESTADO, (1<<FIN_DE_MEDICIONES) ; seteo el flag FIN_DE_MEDICIONES

	pop r21
	pop r20
	reti
```

Resultados obtenidos

Durante la presentación final de las mediciones del trabajo, vimos que las mismas son sensibles al ruido de los potenciómetros, sin variar los mismos la señal medida variaba 1 o 2 discretizaciones del ADC.

Para el valor de temperatura medida a través del LM35 también veíamos una variación pero en términos generales estaba bastante estable. Demostrativamente hubo que modificar la ventana de histéresis para que sea más sensible respecto de la temperatura del laboratorio, es un dato a tener en cuenta ya que si no se modifican en el momento, puede pasar que no llegue a apagarse el ventilador si nunca se alcanza la temperatura de reposo (límite inferior). Se logró una medición más estable en los valores alimentando el microcontrolador por USB directamente y no a través del programador.

Por ultimo comentar que se llevó armado el prototipo y hubo que verificar las conexiones ya que no funcionaba correctamente.

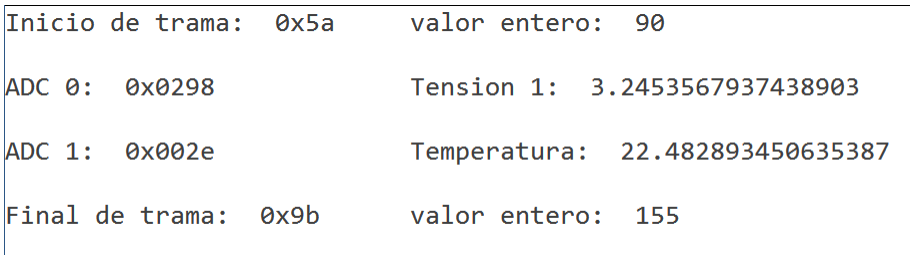


Ilustración 3 - Captura de pantalla de las mediciones de muestra.

Conclusiones

Se implementó el circuito medidor propuesto utilizando un microcontrolador AVR y los conceptos de programación en Assembler aprendido durante la cursada.

Durante el desarrollo del trabajo notamos que las configuraciones de las diferentes funcionalidades son sensibles a fallos de programación respecto al cambio de los bits de los correspondientes registros, siempre hay que estar muy atentos a las especificaciones de la hoja de datos del microcontrolador, y revisar bit por bit todos los registros involucrados para configurarlos correctamente según lo que necesitemos.

También aprendimos que para controlar circuitos de más alta tensión a la del circuito, debemos utilizar circuitos relé que nos permiten manejarlos desde el microcontrolador.

Notamos que el prototipo es sensible a los movimientos, sería mejor armarlo fijo en una madera o similar para evitar conexiones flojas que generen ruido en las mediciones.

Como mejoras para el trabajo proponemos incrementar las mediciones demostrativas, se podría medir valores de corriente, viendo si el consumo de un circuito a superior al esperado y en ese caso apagar la alimentación de los mismos. En el caso de la medición de temperatura, además de prender el ventilador, se podría tener otro límite superior a la ventana y en caso de superarlo apagar la alimentación del circuito para evitar daños.

Otra posible mejora que es fácil de agregar a la presentación es la comentada anteriormente, utilizar el divisor resistivo para escalar la entrada y así poder medir tensiones superiores a 5V, para ello deberíamos sencillamente poner en la entrada de este circuito la tensión máxima a medir y variar los potenciómetros hasta que a la salida tengamos 5V, luego conectarlo al microcontrolador.

Hojas de datos