Instituto Politécnico Nacional

Escuela Superior de Cómputo

Departamento de Ciencias e Ingeniería de la Computación

Sistemas en Chip

Práctica 18

"Practica 18 "Pantalla LCD 16x2""

Profesor: Fernando Aguilar Sánchez

Grupo: 6CM1

Equipo 3

Alumnos:

Ocampo Téllez Rodolfo

Patlani Mauricio Adriana

Ruvalcaba Flores Martha Catalina

Sandoval Hernández Eduardo

Fecha de entrega: 15/01/2023

Objetivo

Al término de la sesión, los integrantes del equipo contaran con la habilidad para manejar una pantalla LCD.

Introducción Teórica

El microcontrolador ATMEGA8535 fabricante ATMEL. En la figura 1 se muestra el microcontrolador ATMEGA8535 la cual maneja datos de 8 bits es decir su bus de datos de 8 bits. Aunque este contiene tres registros los cuales son el x, y y z, los cuales manejan datos de 16 bits.

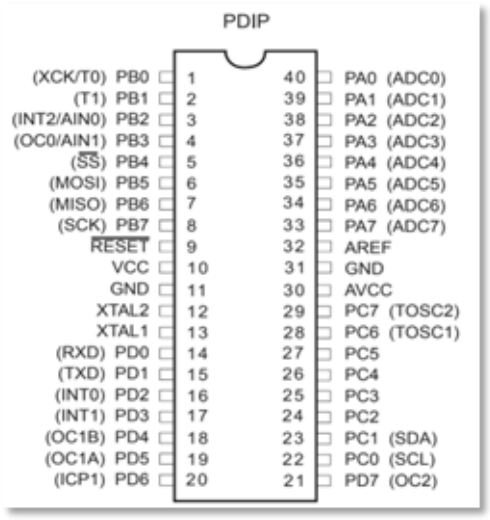

Figura 1. Configuración de pines ATmega8535

El microcontrolador utiliza una arquitectura cerrada, es decir, aquel que es inmodificable por los programadores ajenos a la compañía propietaria del código fuente. Por lo tanto, a este sistema no se le pueden colocar dispositivos periféricos, solo se usa el hardware de la compañía propietaria ya que los dispositivos ajenos a dicha compañía no son compatibles.

El microcontrolador ATMEGA8535 utiliza un encapsulado DIP-40, común en la construcción de circuitos integrados que consiste en un bloque con dos hileras paralelas de pines, observar la figura 2.

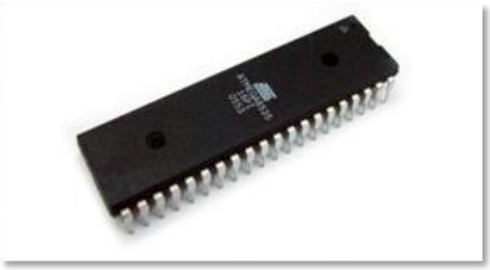

Figura 2. Microcontrolador atmega853.

La comunicación interna del microcontrolador se categoriza en 4 puertos, en la figura 1 se puede analizar la etiquetación de los puertos PA0 al PA7, PB0 al PB7, PC0 al PC7 y PD0 al PD7. Cabe recordar que maneja datos de 8 bits.

El microcontrolador se alimenta de las terminales 10 y 11 como lo muestra la figura 1, los cuales son el VCC (5 V y una tolerancia de ± 0.5V) y GND. Sin embargo, el convertidor se alimenta de forma externa en la terminal 31 y 32, los cuales son GND y AREF.

Posee un oscilador interno de 1MHz, sin embargo, como es un oscilador RC, es susceptible a variar la frecuencia con respecto a las variaciones de temperatura. Por otro lado, puede conectarse un oscilador de 0 a 8 MHz o de 0 a 16 MHz.

LCD 16x2

Las pantallas LCD (pantallas de cristal líquido) se utilizan para mostrar el estado o los parámetros en sistemas integrados. LCD 16x2 es un dispositivo de 16 pines que tiene 8 pines de datos (D0-D7) y 3 pines de control (RS, RW, EN). Los 5 pines restantes son para suministro y retroiluminación para la pantalla LCD.

Los pines de control nos ayudan a configurar la pantalla LCD en modo comando o modo datos. También ayudan a configurar el modo de lectura o el modo de escritura y también cuándo leer o escribir. LCD 16x2 se puede utilizar en modo de 4 bits o modo de 8 bits dependiendo de los requisitos de la aplicación. Para poder utilizarlo necesitamos enviar ciertos comandos a la LCD en modo comando y una vez configurada la LCD según nuestra necesidad, podemos enviar los datos requeridos en modo datos.

Descripción del LCD 16x2

El display LCD (LiquidCrystalDysplay) podemos visualizar diferentes caracteres o números. Las terminales del display son vss, vdd, vee, rs, rw, en, d0, d1, d2, d3, d4, d5, d6 y d7.

- Vss.- Gnd
- Vdd.- +5v
- Vee.- Contrast Control, se ajusta el contraste con un potenciómetro de 10k ohms conectado entre vdd y vee.
- Rs.- RegisterSelect, en estado bajo se mandan instrucciones al display, en estado alto se mandan datos.
- R/W.- Read/Write, en estado bajo se escribe en el display, en estado alto se lee un dato desde el display. (En nuestro ejemplo no leeremos datos del LCD por eso se conecta directamente al Gnd).
- E.- Enable, habilita la lectura o escritura al LCD.
- D0..D7.- Data Pin, mediante estos pines recibimos lectura desde el LCD o escribimos hacia el LCD. Se pueden usar 8 bits (del D0 al D7) o se pueden usar 4 bits (del D4 al D7).

LM35

Los sensores son dispositivos ampliamente usados en multitud de circuitos. Los hay de temperatura, de humedad, de humo, de luz, y un largo etc. Son elementos que permiten medir alguna magnitud y transformarla en una respuesta de tensión. La señal analógica de salida se puede transformar a digital de forma sencilla y así poder usar este tipo de sensores con circuitos digitales, pantallas LCD, una placa Arduino, etc.

El LM35 es uno de los sensores más populares y utilizados de todos, ya que es un circuito electrónico sensor que puede medir temperatura. Su salida es analógica, es decir, te proporciona un voltaje proporcional a la temperatura. El sensor tiene un rango desde −55°C a 150°C. Su popularidad se debe a la facilidad con la que se puede medir la temperatura. Incluso no es necesario de un microprocesador o microcontrolador para medir la temperatura. Dado que el sensor LM35 es analógico, basta con medir con un multímetro, el voltaje a salida del sensor.

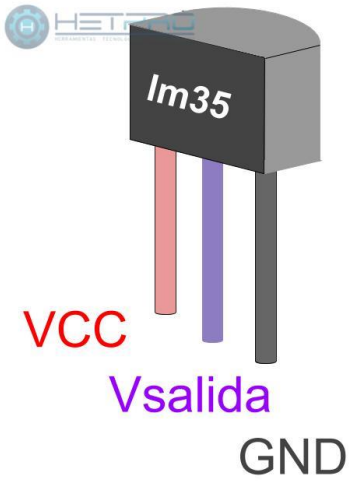

Figura 3. Sensor LM35 en empaquetado TO-92.

Para convertir el voltaje a la temperatura, el LM35 proporciona 10mV por cada grado centígrado. También cabe señalar que ese sensor se puede usar sin offset, es decir que si medimos 20mV a la salida, estaremos midiendo 2°C. Sus características principales son:

- Resolución: 10mV por cada grado centígrado.
- Voltaje de alimentación. Por ejemplo, esté sensor se puede alimentar desde 4Vdc hasta 20Vdc.
- Tipo de medición. Salida analógica.
- Numero de pines: 3 pines, GND, VCC y VSalida.
- No requiere calibración.
- Tiene una precisión de ±¼°C.
- Esta calibrado para medir °C.
- Consumo de corriente: 60 µA

Materiales y Equipo empleado

- CodeVision AVR
- AVR Studio 4
- Microcontrolador ATmega 8535
- 1 Pantalla LCD 16x2
- 1 Potenciómetro 10KΩ
- 1 Resistor de 330Ω

Desarrollo Experimental

1.- Con la información que a continuación se menciona, diseñe un programa para visualizar en una pantalla LCD 16x2 la fecha, la hora y la temperatura actual, tal y como se muestra en la figura 4. Los botones son para el ajuste de fecha y hora.

Figura 4. Asignación de pines de la pantalla LCD 16x2.

En el asistente de CodeVision habilite la pestaña de la LCD dándo clic como se muestra en la siguiente figura, si desea colocar la LCD en otro puerto diferente, sólo indíquele donde quiere conectarlo y el compilador le dirá donde conectar los pines y él se encargará de inicializar la LCD.

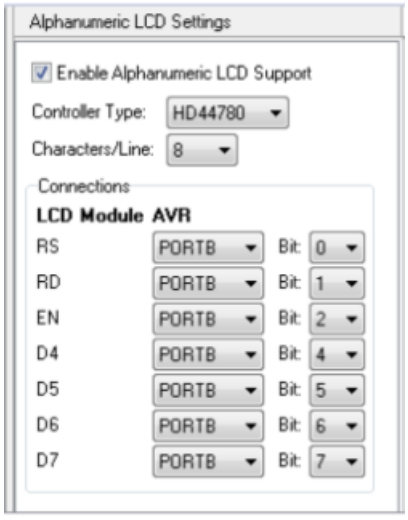

Figura 5. Asignación de pines del Puerto a la pantalla LCD 16x2.

A continuación, en la tabla 1, se muestra la asignación y conexión de los pines de la pantalla LCD de 16x2. Y en la tabla 2, tomar en cuenta que las siguientes funciones sólo se pueden usar si se inicializó la LCD en el CodeVision y en la tabla 3, las posiciones de la LCD al usar la función "lcd_gotoxy(x,y)".

LCD PIN #	DESCRIPCIÓN	CONEXIÓN
1	VSS	GND
2	VDD	+5V
3	VO	Terminal de ajuste de contraste
4	Register Select (RS)	Conecte al bit del Puerto como indica CodeVision
5	Read/Write (R/W)	Conecte al bit del Puerto como indica CodeVision
6	Clock Enable (E)	Conecte al bit del Puerto como indica CodeVision
7	Data Bit 0	No Conectar
8	Data Bit 1	No Conectar
9	Data Bit 2	No Conectar
10	Data Bit 3	No Conectar
11	Data Bit 4	Conecte al bit del Puerto como indica CodeVision
12	Data Bit 5	Conecte al bit del Puerto como indica CodeVision
13	Data Bit 6	Conecte al bit del Puerto como indica CodeVision
14	Data Bit 7	Conecte al bit del Puerto como indica CodeVision
15	Backlight Anode (+)	+5V
16	Backlight Cathode (-)	GND

Tabla 1. Asignación y conexión de pines de la pantalla LCD 16x2.

FUNCIÓN	DESCRIPCIÓN
lcd_gotoxy(x,y);	Esta función permite indicarle donde colocar el cursor, que es donde empezará a escribir el mensaje. x es la columna, y es la fila.
lcd_putsf("mensaje");	Con la función anterior le indicamos el mensaje a escribir en la LCD.
lcd_clear();	Esta función borra el mensaje en el display.
lcd_putchar('A');	Esta función escribe en pantalla una sóla letra o un carácter.
lcd_putchar(0x40)	Esta función escribe en pantalla una sóla letra o un carácter en código ASCII.
_lcd_ready(); _lcd_write_data(0xFF);	Los bits para controlar el cursor y la pantalla son: Dato=0xFF=00001DCB D=1 Enciende pantalla, D=0 coloca en stand by la pantalla (bajo consumo) C=1 Cursor on, C=0 Cursor Off B=1 Cursor parpadea, B=0 Cursor fijo
_lcd_ready(); _lcd_write_data(0xFF);	Los bits para controlar el desplazamiento del cursor y la pantalla son: Dato=0001 S/C R/L 00 Y los bits S/C y R/L tienen la siguiente descripción
	S/C R/L Descripción 0 0 Desplaza el cursor a la izquierda 0 1 Desplaza el cursor a la derecha 1 0 Desplaza la pantalla y cursor a la izquierda 1 1 Desplaza la pantalla y cursor a la derecha

Tabla 2. Funciones para la LCD reconocidas en CodeVision.

POSICIÓN	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
FILA 01	(4,0)	(5,0)	(6,0)	(7,0)	(8,0)	(9,0)	(10,0)	(11,0)	(12,0)	(13,0)	(14,0)	(15,0)	(16,1)	(17,1)	(18,1)	(19,1)
FILA 02	(4,1)	(5,1)	(6,1)	(7,1)	(8,1)	(9,1)	(10,1)	(11,1)	(12,1)	(13,1)	(14,1)	(15,1)	(16,2)	(17,2)	(18,2)	(19,2)

Tabla 3. Posiciones de la LCD al usar la función "lcd_gotoxy(x,y)".

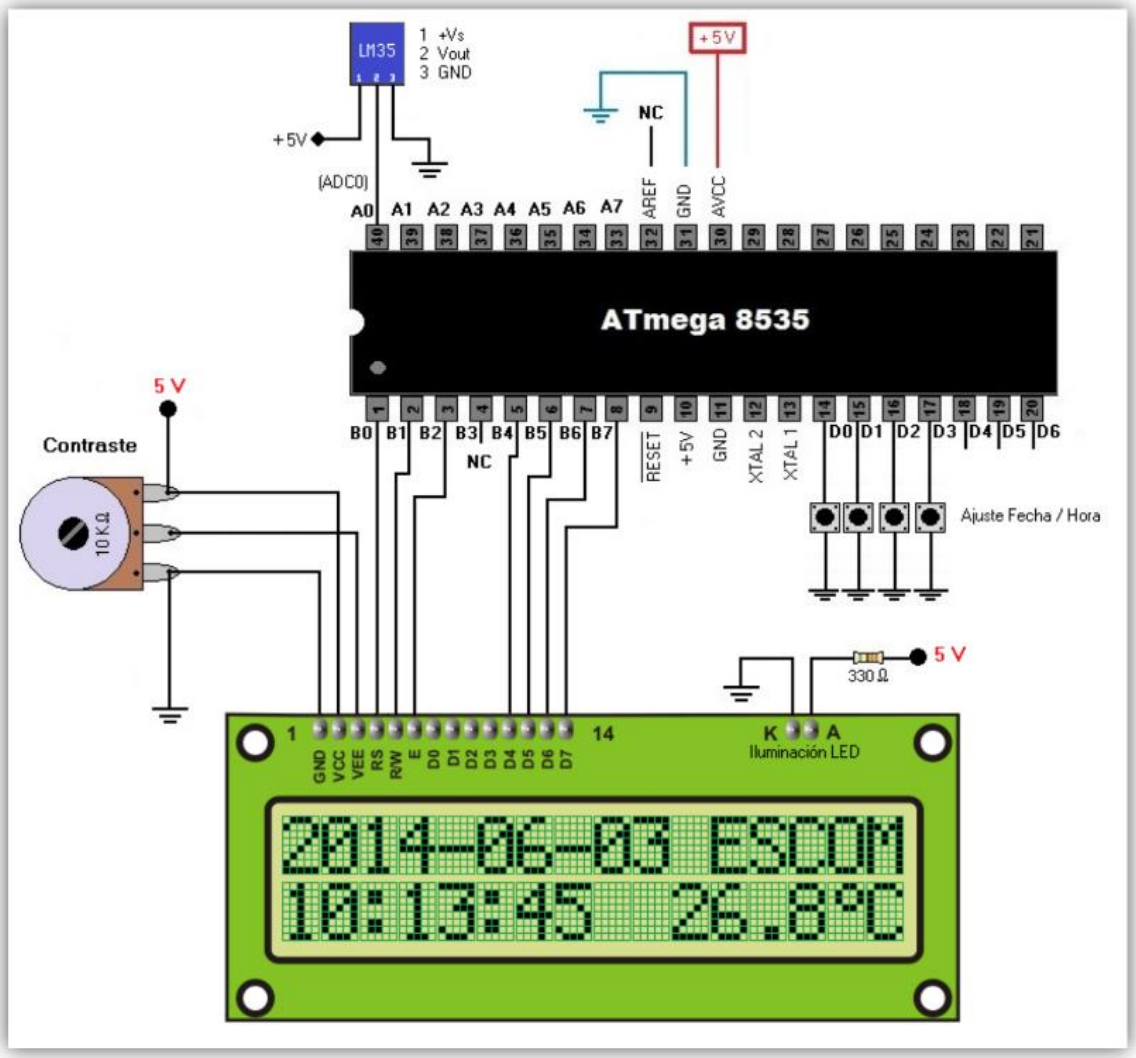

Figura 6. Circuito conectando la LCD en el Puerto B.

Estructura del programa

Código de la configuración de los periféricos utilizados y código del programa principal en C, proporcionados por el IDE de CodeVision.

```
#include <mega8535.h>
#include <delay.h>
// Alphanumeric LCD functions
#include <alcd.h>
#define boton PIND.0
#define ha PIND.1
#define mm PIND.2
#define sd PIND.3
#define ADC_VREF_TYPE ((0<<REFS1) | (1<<REFS0) | (1<<ADLAR))
```

```
// Read the 8 most significant bits
// of the AD conversion result
unsigned char read_adc(unsigned char adc_input)
{
ADMUX=adc_input | ADC_VREF_TYPE;
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=(1<<ADSC);
// Wait for the AD conversion to complete
while ((ADCSRA & (1<<ADIF))==0);
ADCSRA|=(1<<ADIF);
return ADCH;
}

float valorFinal;
int temperatura;
int desplz;
int cont_antidelay,time_antidelay;
bit botonp,botona,cambio = 0;
unsigned char unidades,decenas,decimas,valorADC,seg=0,min=0,hor=0,dia=30,mes=12;
unsigned short anio1=19,anio2=99;
const char ascii=48;

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
DDRA=(0<<DDA7) | (0<<DDA6) | (0<<DDA5) | (0<<DDA4) | (0<<DDA3) | (0<<DDA2) | (0<<DDA1)
| (0<<DDA0);
PORTA=(0<<PORTA7) | (0<<PORTA6) | (0<<PORTA5) | (0<<PORTA4) | (0<<PORTA3) | (0<<PORTA2)
| (0<<PORTA1) | (0<<PORTA0);
// Port B initialization
DDRB=(1<<DDB7) | (1<<DDB6) | (1<<DDB5) | (1<<DDB4) | (1<<DDB3) | (1<<DDB2) | (1<<DDB1)
| (1<<DDB0);
PORTB=(0<<PORTB7) | (0<<PORTB6) | (0<<PORTB5) | (0<<PORTB4) | (0<<PORTB3) | (0<<PORTB2)
| (0<<PORTB1) | (0<<PORTB0);
// Port C initialization
DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (0<<DDC3) | (0<<DDC2) | (0<<DDC1)
| (0<<DDC0);
PORTC=(0<<PORTC7) | (0<<PORTC6) | (0<<PORTC5) | (0<<PORTC4) | (0<<PORTC3) | (0<<PORTC2)
| (0<<PORTC1) | (0<<PORTC0);
// Port D initialization
```

```
DDRD=(0<<DDD7) | (0<<DDD6) | (0<<DDD5) | (0<<DDD4) | (0<<DDD3) | (0<<DDD2) | (0<<DDD1)
| (0<<DDD0);
PORTD=(1<<PORTD7) | (1<<PORTD6) | (1<<PORTD5) | (1<<PORTD4) | (1<<PORTD3) | (1<<PORTD2)
| (1<<PORTD1) | (1<<PORTD0);
…
…
…
// ADC initialization
// ADC Clock frequency: 500.000 kHz
// ADC Voltage Reference: AVCC pin
ADMUX=ADC_VREF_TYPE;
ADCSRA=(1<<ADEN) | (0<<ADSC) | (0<<ADATE) | (0<<ADIF) | (0<<ADIE) | (0<<ADPS2) |
(0<<ADPS1) | (1<<ADPS0);
SFIOR=(1<<ADHSM) | (0<<ADTS2) | (0<<ADTS1) | (0<<ADTS0);

// Alphanumeric LCD initialization
// Characters/line: 16
lcd_init(16);
desplz=0;
cont_antidelay=0;
time_antidelay=20;

while (1)
        {
        delay_ms(50);

        if(boton==0)
            botona=0;
        else
            botona=1;

        if((botonp==1) && (botona==0)) {
            cambio = ~cambio;
            delay_ms(10);
        }

        botonp=botona;

        lcd_gotoxy(11,0);
        lcd_putsf("ESCOM");
        valorADC=10*read_adc(0);
        valorFinal=50 * valorADC / 255;
        valorFinal= valorFinal + 7;
        if(valorFinal>99) valorFinal=99;

        temperatura=valorFinal*10;
```

```
        decenas=temperatura/100;
        temperatura%=100;
        decimas=temperatura%10;
        unidades=temperatura/10;
        lcd_gotoxy(10,1);
        lcd_putchar(decenas+ascii);
        lcd_gotoxy(11,1);
        lcd_putchar(unidades+ascii);
        lcd_gotoxy(12,1);
        lcd_putchar('.');
        lcd_gotoxy(13,1);
        lcd_putchar(decimas+ascii);
        lcd_gotoxy(14,1);
        lcd_putchar(ascii+175);
        lcd_gotoxy(15,1);
        lcd_putchar('C');

        if(cambio==1) {
            if(ha==0){
                hor++;
                delay_ms(50);
            }

            if(mm==0){
                min++;
                delay_ms(50);
            }

            if(sd==0){
                seg++;
                delay_ms(50);
            }
        } else {
            if(ha==0){
                anio2++;
                if(anio2>99){
                    anio1++; anio2=0;
                }
                delay_ms(50);
            }

            if(mm==0){
                mes++;
                delay_ms(50);
            }
```

```
            if(sd==0){
                dia++;
                delay_ms(50);
            }
        }

        if(desplz>49){
            desplz=0;
            seg++;
        } else {
            desplz++;
        }

        if(seg>59){ min++; seg=0; }
        if(min>59){ hor++; min=0; seg=0; }
        if(hor>23){ dia++; hor=0; seg=0; min=0; }
        if(dia>31){ mes++; dia=0; }
        if(mes>12){
            anio2++;
            mes=0;
            if(anio2>99){ anio1++; anio2=0; }
        }

        lcd_gotoxy(0,1);
        lcd_putchar(hor/10+ascii);
        lcd_gotoxy(1,1);
        lcd_putchar(hor%10+ascii);
        lcd_gotoxy(2,1);
        lcd_putchar(':');
        lcd_gotoxy(3,1);
        lcd_putchar(min/10+ascii);
        lcd_gotoxy(4,1);
        lcd_putchar(min%10+ascii);
        lcd_gotoxy(5,1);
        lcd_putchar(':');
        lcd_gotoxy(6,1);
        lcd_putchar(seg/10+ascii);
        lcd_gotoxy(7,1);
        lcd_putchar(seg%10+ascii);

        lcd_gotoxy(0,0);
        lcd_putchar(anio1/10+ascii);
        lcd_gotoxy(1,0);
        lcd_putchar(anio1%10+ascii);
        lcd_gotoxy(2,0);
        lcd_putchar(anio2/10+ascii);
```

```
        lcd_gotoxy(3,0);
        lcd_putchar(anio2%10+ascii);
        lcd_gotoxy(4,0);
        lcd_putchar('-');
        lcd_gotoxy(5,0);
        lcd_putchar(mes/10+ascii);
        lcd_gotoxy(6,0);
        lcd_putchar(mes%10+ascii);
        lcd_gotoxy(7,0);
        lcd_putchar('-');
        lcd_gotoxy(8,0);
        lcd_putchar(dia/10+ascii);
        lcd_gotoxy(9,0);
        lcd_putchar(dia%10+ascii);

        delay_ms(50);

      }
}
```

Simulaciones

La simulación se realizó en el software Proteus Design Suite, previamente cargado el programa en formato ".hex" obtenido del software CodeVision AVR, el cual es un software para el microcontrolador Microchip AVR y sus contrapartes XMEGA.

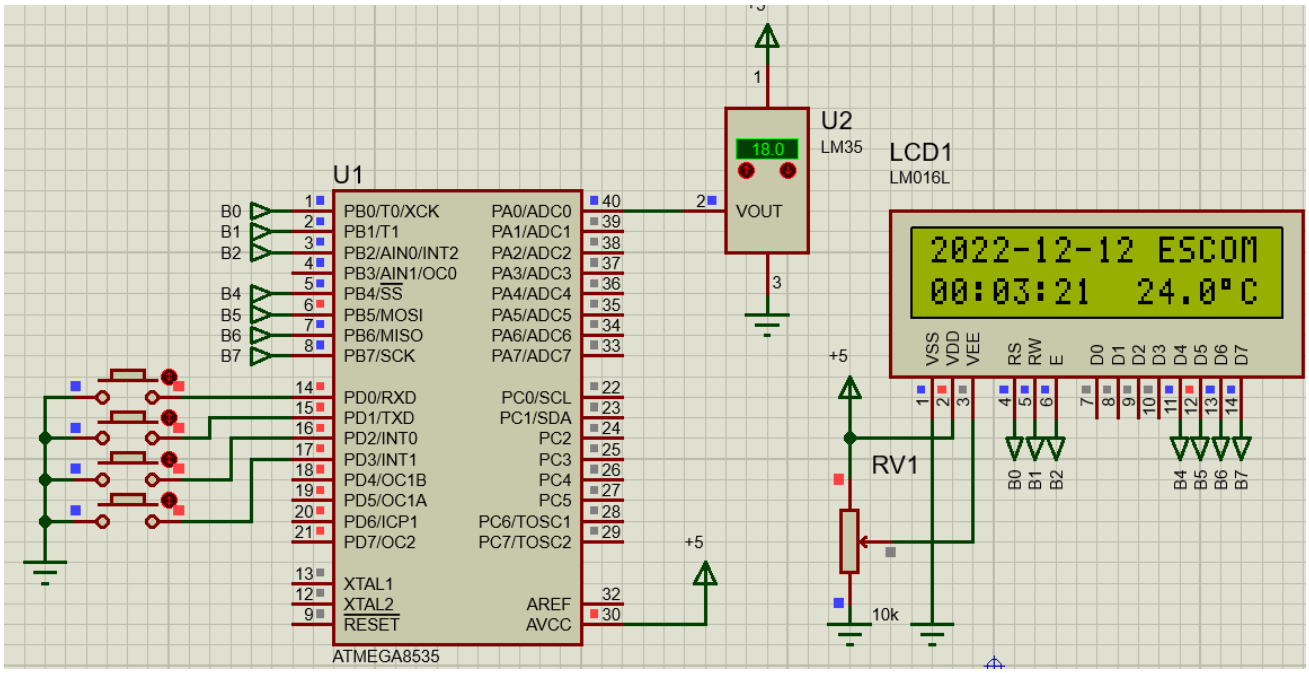

Figura 7. Simulación de la práctica 18.

Fotografía del circuito armado

A continuación, se muestra la figura 9 la evidencia sobre la realización y prueba de la practica 18.

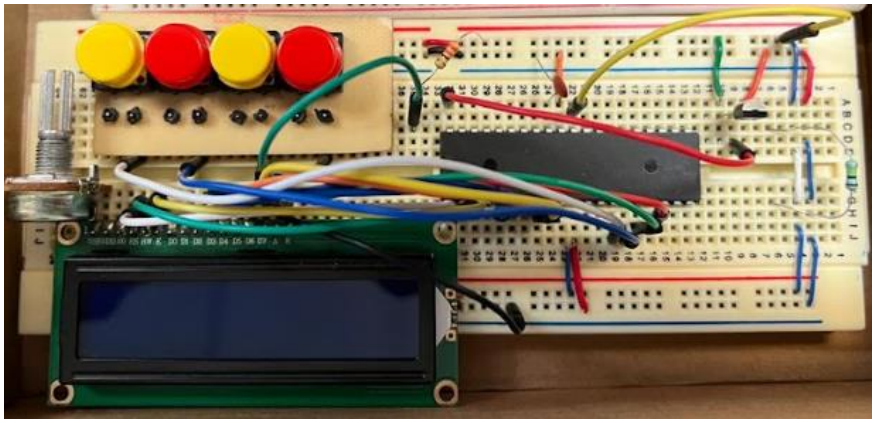

Figura 8. Circuito correspondiente a la práctica 18.

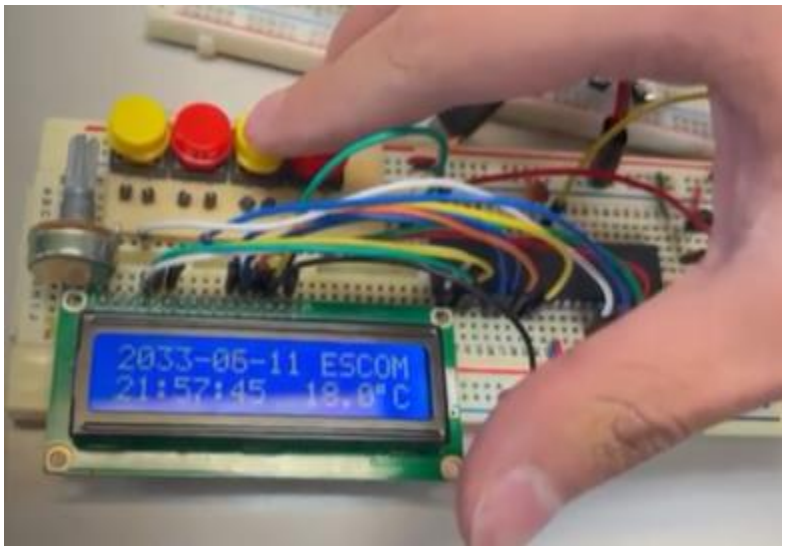

Figura 9. Circuito armado y funcionamiento de la práctica 18.

Observaciones y conclusiones Individuales

Ocampo Téllez Rodolfo

Esta práctica integró elementos que previamente se habían trabajado por lo que no presentó mayor complicación en cuanto a la realización del código, ya que incluso en materias anteriores como Arquitectura de Computadoras se había manejado el display LCD con instrucciones a un nivel más bajo, en esta práctica gracias al uso del microcontrolador Atmega 8535 y principalmente a su entorno CodeVision el manejo del LCD facilitado utilizando el asistente y las funciones disponibles. Del mismo modo se volvió a incorporar el termómetro realizado en la práctica 16, aunque en esta ocasión los valores resultantes de la conversión anlógica-digital se manejaron de forma diferente transformando el valor entero de la variable mediante un cast a uno char mediante el código ASCII.

Patlani Mauricio Adriana

Usando los conocimientos obtenidos en arquitectura de computadoras, el manejo del display LCD hizo que la realización de esta prática fuera sencilla. En esta práctica solo se mostraron datos como la fecha, la hora y un texto. Y usando lo aprendido en la práctica del termometro, solo se agregó el display LCD para mostrar los valores

Ruvalcaba Flores Martha Catalina

La realización de esta práctica no tuvo mayor complicación, puesto que el manejo del LCD 16x2 se adquirió en la materia de arquitectura de computadoras. Por lo que, la realización para mostrar la fecha, hora y un texto fue sencillo. Y para la temperatura, se utilizó un LM35 para capturarla, y para mostrarlo en el LCD, se tuvo que hacer la visualización carácter por carácter, entonces primero se capturó la temperatura y ese valor se tuvo que descomponer en caracteres.

Sandoval Hernández Eduardo

Esta práctica fue un tanto sencilla puesto que ya tenemos experiencia previa con el LCD de 16x2 ya que tuvimos una práctica similar en Arquitectura de Computadoras, la implementación del código para su uso en el ATMega es mucho más sencilla que en la tarjeta Altera ya que no solo la sintaxis del lenguaje C es más comprensible que VHDL sino que también ya se cuenta con funciones que simplifican todo el trabajo, además de que en esta práctica se reutilizó parte del código para el sensor de temperatura, lo más complicado fue interpretar la información que se quería mostrar en el LCD.

Bibliografía

- F. Aguilar. (2020, Octubre 12). 18 Práctica de Pantalla LCD 16x2. Introducción a los Microcontroladores. [Online]. Disponible en: https://youtu.be/P_g_KhGpqVI

- Atmel Corporation. (2006). 8-bit Microcontroller with 8K Bytes In-System Programmable Flash. [Online]. Disponible en: https://drive.google.com/file/d/1acpQaDlsyLHr3w3ReSgrlXRbshZ5Z-lb/view

- F. Cereijo. (2004, Diciembre 28). Microcontroladores PIC. [Online]. Disponible en: https://drive.google.com/file/d/1mAydVlkZYqtYXhFQ0Tnr_UjZyjiwIp90/view

- LCD16x2 interfacing with AVR ATmega16/ATmega32. (n.d.). Electronicwings.com. Retrieved December 11, 2022, from https://www.electronicwings.com/avr-atmega/lcd16x2-interfacing-with-atmega16-32

- Isaac. (Febrero. 2021). LM35: información completa sobre este sensor de temperatura. [Online]. Disponible en: https://www.hwlibre.com/lm35/

- R. Marmolejo. (Febr, 2022). LM35 – El sensor de temperatura más popular. [Online]. Disponible en: https://hetpro-store.com/TUTORIALES/lm35/