INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

PROGRAMA ACADÉMICO
INGENIERÍA EN SISTEMAS COMPUTACIONALES

INTRODUCCIÓN A LOS MICROCONTROLADORES

Práctica #18

"Pantalla LCD 16x2"

Vídeo explicativo:

https://youtu.be/8Hiew8ZTYXU

Alumno:

Rojas Alvarado Luis Enrique

Grupo:
3CM6

Profesor:
Fernando Aguilar Sánchez

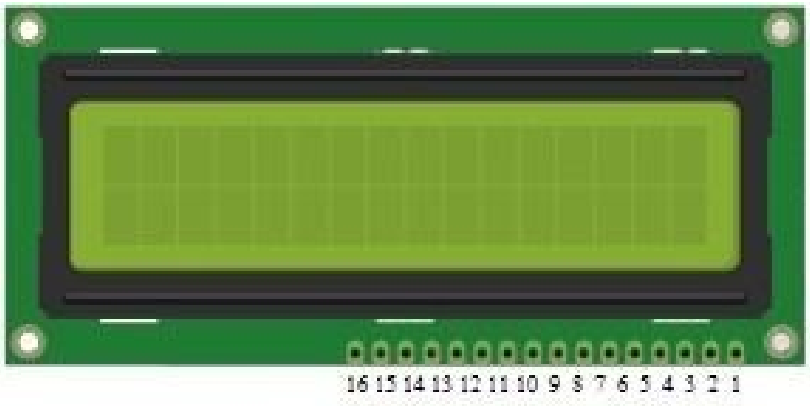
OBJETIVO

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

INTRODUCCIÓN

Una pantalla LCD (liquid crystal display: 'pantalla de cristal líquido' por sus siglas en inglés) es una pantalla delgada y plana formada por un número de píxeles en color o monocromos colocados delante de una fuente de luz o reflectora. A menudo se utiliza en dispositivos electrónicos de pilas, ya que utiliza cantidades muy pequeñas de energía eléctrica.

Está presente en un sinnúmero de aparatos, desde la limitada imagen que muestra una calculadora de bolsillo hasta televisores de 50 o más pulgadas. Estas pantallas se componen de miles de pequeños cristales líquidos, que no son sólidos ni Líquidos en realidad, sino un estado intermedio.



Importantes factores que se deben considerar al evaluar una pantalla LCD:

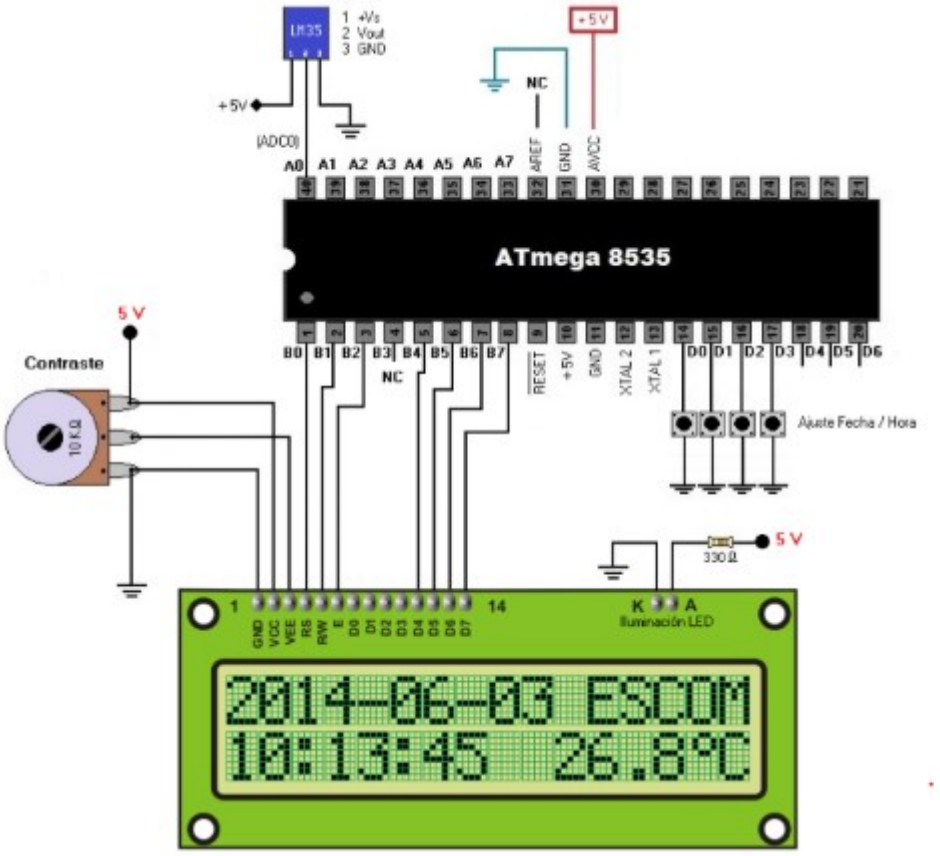
- Resolución: Las dimensiones horizontal y vertical son expresadas en píxeles. Las pantallas HD tienen una resolucion nativa de 1366 x 768 píxeles (720p) y la resolucion nativa en las Full HD es de 1920 x 1080 píxeles (1080p).
- Ancho de punto: La distancia entre los centros de dos píxeles adyacentes. Cuanto menor sea el ancho de punto, tanto menor granularidad tendrá la imagen. El ancho de punto puede ser el mismo en sentido vertical y horizontal, o bien diferente (menos frecuente).
- Tamaño: El tamaño de un panel LCD se mide a lo largo de su diagonal, generalmente expresado en pulgadas (coloquialmente llamada área de visualización activa).
- Tiempo de respuesta: Es el tiempo que demora un píxel en cambiar de un color a otro
- Tipo de matriz: Activa, pasiva y reactiva.
- Ángulo de visión: Es el máximo ángulo en el que un usuario puede mirar el LCD, es estando desplazado de su centro, sin que se pierda calidad de imagen.Las nuevas pantallas vienen con un angulo de vision de 178 grados
- Soporte de color: Cantidad de colores soportados. Coloquialmente conocida como gama de colores.
- Brillo: La cantidad de luz emitida desde la pantalla; también se conoce como luminosidad
- Contraste: La relación entre la intensidad más brillante y la más oscura.

- Aspecto: La proporción de la anchura y la altura (por ejemplo, 5:4, 4:3, 16:9 y 16:10).
- Puertos de entrada: Por ejemplo: DVI, VGA, LVDS o incluso S-Video y HDMI.

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 3. Los botones son para el ajuste de fecha y hora.

Circuito



Código

```
1.  /*****************************************************
2.  This program was created by the
3.  CodeWizardAVR V2.60 Evaluation
4.  Automatic Program Generator
5.  © Copyright 1998-2012 Pavel Haiduc, HP InfoTech s.r.l.
6.  http://www.hpinfotech.com
7.
8.  Project :
9.  Version :
10. Date    :
11. Author  :
12. Company :
13. Comments:
14.
15.
16. Chip type               : ATmega8535L
17. Program type            : Application
18. AVR Core Clock frequency: 1,000000 MHz
19. Memory model            : Small
```

```
External RAM size       : 0
Data Stack size         : 128
*****************************************************/

#include <mega8535.h>

#include <delay.h>

// Alphanumeric LCD functions
#include <alcd.h>

#define cambio PIND.0
#define ha PIND.1
#define mm PIND.2
#define sd PIND.3

float cel;
int tem;
int desplz;
int cont_antidelay,time_antidelay;
bit btnp,btna;
unsigned char unidades,decenas,decimas,cn,seg=0,min=0,hor=0,dia=25,mes=10,change;
unsigned short ye=19,ar=97;
const char car=48; //codigo ascii

// Declare your global variables here

#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;
}

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

// Input/Output Ports initialization
// Port A initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
DDRA=(0<<DDA7) | (0<<DDA6) | (0<<DDA5) | (0<<DDA4) | (0<<DDA3) | (0<<DDA2) | (0<<DDA1) | (0<<DDA0);
// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
PORTA=(0<<PORTA7) | (0<<PORTA6) | (0<<PORTA5) | (0<<PORTA4) | (0<<PORTA3) | (0<<PORTA2) | (0<<PORTA1)
    | (0<<PORTA0);

// Port B initialization
// Function: Bit7=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out Bit1=Out Bit0=Out
DDRB=(1<<DDB7) | (1<<DDB6) | (1<<DDB5) | (1<<DDB4) | (1<<DDB3) | (1<<DDB2) | (1<<DDB1) | (1<<DDB0);
// State: Bit7=0 Bit6=0 Bit5=0 Bit4=0 Bit3=0 Bit2=0 Bit1=0 Bit0=0
PORTB=(0<<PORTB7) | (0<<PORTB6) | (0<<PORTB5) | (0<<PORTB4) | (0<<PORTB3) | (0<<PORTB2) | (0<<PORTB1)
    | (0<<PORTB0);

// Port C initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (0<<DDC3) | (0<<DDC2) | (0<<DDC1) | (0<<DDC0);
// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
```

```
PORTC=(0<<PORTC7) | (0<<PORTC6) | (0<<PORTC5) | (0<<PORTC4) | (0<<PORTC3) | (0<<PORTC2) | (0<<PORTC1) | (0<<PORTC0);

// Port D initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
DDRD=(0<<DDD7) | (0<<DDD6) | (0<<DDD5) | (0<<DDD4) | (0<<DDD3) | (0<<DDD2) | (0<<DDD1) | (0<<DDD0);
// State: Bit7=P Bit6=P Bit5=P Bit4=P Bit3=P Bit2=P Bit1=P Bit0=P
PORTD=(1<<PORTD7) | (1<<PORTD6) | (1<<PORTD5) | (1<<PORTD4) | (1<<PORTD3) | (1<<PORTD2) | (1<<PORTD1) | (1<<PORTD0);

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=(0<<WGM00) | (0<<COM01) | (0<<COM00) | (0<<WGM01) | (0<<CS02) | (0<<CS01) | (0<<CS00);
TCNT0=0x00;
      OCR0=0x00;

      // Timer/Counter 1 initialization
      // Clock source: System Clock
      // Clock value: Timer1 Stopped
      // Mode: Normal top=0xFFFF
      // OC1A output: Disconnected
      // OC1B output: Disconnected
      // Noise Canceler: Off
      // Input Capture on Falling Edge
      // Timer1 Overflow Interrupt: Off
      // Input Capture Interrupt: Off
      // Compare A Match Interrupt: Off
      // Compare B Match Interrupt: Off
      TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) | (0<<COM1B0) | (0<<WGM11) | (0<<WGM10);
      TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<WGM13) | (0<<WGM12) | (0<<CS12) | (0<<CS11) | (0<<CS10);

      TCNT1H=0x00;
      TCNT1L=0x00;
      ICR1H=0x00;
      ICR1L=0x00;
      OCR1AH=0x00;
      OCR1AL=0x00;
      OCR1BH=0x00;
      OCR1BL=0x00;

      // Timer/Counter 2 initialization
      // Clock source: System Clock
      // Clock value: Timer2 Stopped
      // Mode: Normal top=0xFF
      // OC2 output: Disconnected
      ASSR=0<<AS2;
      TCCR2=(0<<WGM20) | (0<<COM21) | (0<<COM20) | (0<<WGM21) | (0<<CS22) | (0<<CS21) | (0<<CS20);
      TCNT2=0x00;
      OCR2=0x00;

      // Timer(s)/Counter(s) Interrupt(s) initialization
      TIMSK=(0<<OCIE2) | (0<<TOIE2) | (0<<TICIE1) | (0<<OCIE1A) | (0<<OCIE1B) | (0<<TOIE1) | (0<<OCIE0) | (0<<TOIE0);

      // External Interrupt(s) initialization
      // INT0: Off
      // INT1: Off
      // INT2: Off
      MCUCR=(0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);
      MCUCSR=(0<<ISC2);

      // USART initialization
      // USART disabled
      UCSRB=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) | (0<<RXEN) | (0<<TXEN) | (0<<UCSZ2) | (0<<RXB8) | (0<<TXB8);
```

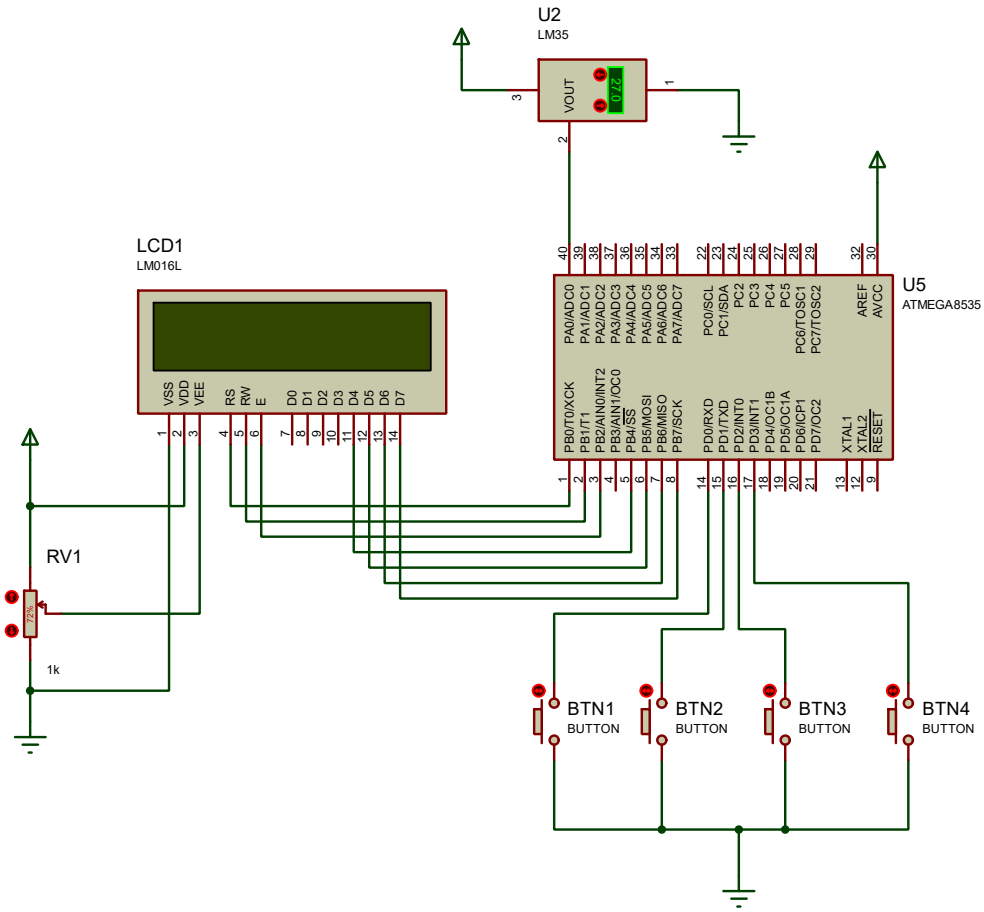
```
148.
149.       // Analog Comparator initialization
150.       // Analog Comparator: Off
151.       ACSR=(1<<ACD) | (0<<ACBG) | (0<<ACO) | (0<<ACI) | (0<<ACIE) | (0<<ACIC) | (0<<ACIS1) | (0<<ACI
   S0);
152.
153.       // ADC initialization
154.       // ADC Clock frequency: 500,000 kHz
155.       // ADC Voltage Reference: AVCC pin
156.       // ADC High Speed Mode: Off
157.       // ADC Auto Trigger Source: ADC Stopped
158.       // Only the 8 most significant bits of
159.       // the AD conversion result are used
160.       ADMUX=ADC_VREF_TYPE;
161.       ADCSRA=(1<<ADEN) | (0<<ADSC) | (0<<ADATE) | (0<<ADIF) | (0<<ADIE) | (0<<ADPS2) | (0<<ADPS1) |
   (1<<ADPS0);
162.       SFIOR=(1<<ADHSM) | (0<<ADTS2) | (0<<ADTS1) | (0<<ADTS0);
163.
164.       // SPI initialization
165.       // SPI disabled
166.       SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) | (0<<CPOL) | (0<<CPHA) | (0<<SPR1) | (0<<SP
   R0);
167.
168.       // TWI initialization
169.       // TWI disabled
170.       TWCR=(0<<TWEA) | (0<<TWSTA) | (0<<TWSTO) | (0<<TWEN) | (0<<TWIE);
171.
172.       // Alphanumeric LCD initialization
173.       // Connections are specified in the
174.       // Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
175.       // RS - PORTB Bit 0
176.       // RD - PORTB Bit 1
177.       // EN - PORTB Bit 2
178.       // D4 - PORTB Bit 4
179.       // D5 - PORTB Bit 5
180.       // D6 - PORTB Bit 6
181.       // D7 - PORTB Bit 7
182.       // Characters/line: 16
183.       lcd_init(16);
184.       desplz=0;
185.       cont_antidelay=0;
186.       time_antidelay=20;
187.       while (1)
188.             {
189.                   delay_ms(1);
190.                   if(cambio==0)
191.
192.                       btna=0;
193.                       else
194.                       btna=1;
195.                   if((btnp==1)&&(btna==0)){
196.
197.                   if(change==0){
198.
199.                   change=1;
200.                   }
201.                     else{
202.                     change=0;
203.                     }
204.                   }
205.                   btnp=btna;
206.
207.
208.
209.
210.                lcd_gotoxy(11,0);
211.                lcd_putsf("ESCOM");
212.
```

```
213.
214.                 cn=read_adc(0);
215.                 cel=cn*1.45;
216.                 if(cel>99)
217.                 cel=99;
218.                 tem=cel*10;
219.                 decenas=tem/100;
220.                 tem%=100;
221.                 decimas=tem%10;
222.                 unidades=tem/10;
223.
224.
225.                 lcd_gotoxy(10,1);
226.                 lcd_putchar(decenas+car);
227.                 lcd_gotoxy(11,1);
228.                 lcd_putchar(unidades+car);
229.                 lcd_gotoxy(12,1);
230.                 lcd_putchar('.');
231.                 lcd_gotoxy(13,1);
232.                 lcd_putchar(decimas+car);
233.
234.                 lcd_gotoxy(14,1);
235.                 lcd_putchar(car+175);
236.                 lcd_gotoxy(15,1);
237.                 lcd_putchar('C');
238.                 ////////////////////reloj en mov//////////////////////////
239.                 if(change==1){
240.                   if(ha==0){
241.                   if(cont_antidelay>time_antidelay){
242.                   cont_antidelay=0;hor++;
243.                   }else{
244.                   cont_antidelay++;
245.                   }
246.                   }
247.                   if(mm==0){
248.                   if(cont_antidelay>time_antidelay){
249.                   cont_antidelay=0;min++;
250.                   }else{
251.                   cont_antidelay++;
252.                   }
253.                   }
254.                   if(sd==0){
255.                   if(cont_antidelay>time_antidelay){
256.                   cont_antidelay=0;seg++;
257.                   }else{
258.                   cont_antidelay++;
259.                   }
260.                   }
261.                 }else{
262.                   if(ha==0){
263.                   if(cont_antidelay>time_antidelay){
264.                   cont_antidelay=0;
265.                   ar++;
266.                   if(ar>99){
267.                   ye++;
268.                   ar=0;
269.                   }
270.                   }else{
271.                   cont_antidelay++;
272.                   }
273.                   }
274.                   if(mm==0){
275.                   if(cont_antidelay>time_antidelay){
276.                   cont_antidelay=0;
277.                   mes++;
278.                   }else{
279.                   cont_antidelay++;
280.                   }
```

```
                  }
                  if(sd==0){
                  if(cont_antidelay>time_antidelay){
                  cont_antidelay=0;
                  dia++;
                  }else{
                  cont_antidelay++;
                  }
                  }
                }


               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){
               ar++;
               mes=0;
               if(ar>99){
               ye++;
               ar=0;
               }
               }
                ////////////////////////////////////////////hora//////////////////////////
               lcd_gotoxy(0,1);
               lcd_putchar(hor/10+car);
               lcd_gotoxy(1,1);
               lcd_putchar(hor%10+car);

               lcd_gotoxy(2,1);
               lcd_putchar(':');

               lcd_gotoxy(3,1);
               lcd_putchar(min/10+car);
               lcd_gotoxy(4,1);
               lcd_putchar(min%10+car);

               lcd_gotoxy(5,1);
               lcd_putchar(':');

               lcd_gotoxy(6,1);
               lcd_putchar(seg/10+car);
```

```
                lcd_gotoxy(7,1);
                lcd_putchar(seg%10+car);

                ////////////////////////////////////////////fecha////////////////////////////////////
    ////////
                lcd_gotoxy(0,0);
                lcd_putchar(ye/10+car);
                lcd_gotoxy(1,0);
                lcd_putchar(ye%10+car);
                lcd_gotoxy(2,0);
                lcd_putchar(ar/10+car);
                lcd_gotoxy(3,0);
                lcd_putchar(ar%10+car);


                lcd_gotoxy(4,0);
                lcd_putchar('-');

                lcd_gotoxy(5,0);
                lcd_putchar(mes/10+car);
                lcd_gotoxy(6,0);
                lcd_putchar(mes%10+car);

                lcd_gotoxy(7,0);
                lcd_putchar('-');

                lcd_gotoxy(8,0);
                lcd_putchar(dia/10+car);
                lcd_gotoxy(9,0);
                lcd_putchar(dia%10+car);
            }
        }
```

Simulación



Conclusiones

En la realización de esta práctica adquirimos la habilidad para manejar una pantalla LCD.

Aprendí la interacción del microprocesador con los periféricos asociados a esta práctica, reforzando así mis conocimientos de la arquitectura de este así como de los funcionamientos del periférico.

Bibliografía

[1] Anónimo, «LCD (pantalla de cristal líquido),» [En línea]. Available: https://www.ecured.cu/LCD_(pantalla_de_cristal_líquido)#:~:text=Una%20pantalla%20LCD%20(liquid%20crystal,fuente%20de%20luz%20o%20reflectora..