|  |  |  |
| --- | --- | --- |
| http://www.clep-cedep.org/sites/default/files/logo_ipn.jpg | **INSTITUTO POLITÉCNICO NACIONAL** ESCUELA SUPERIOR DE CÓMPUTO  **PROGRAMA ACADÉMICO** INGENIERÍA EN SISTEMAS COMPUTACIONALES | Resultado de imagen para logo escom |

**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

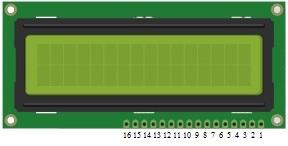
**OBJETIVO**

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.



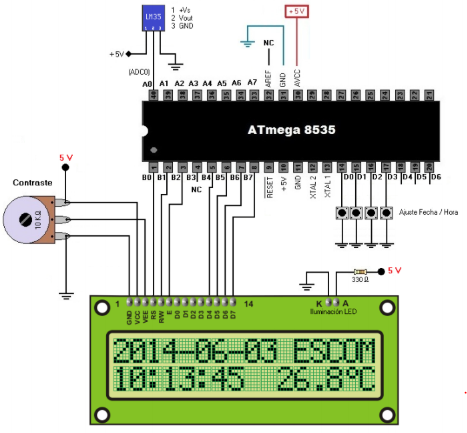
Importantes factores que se deben considerar al evaluar una pantalla LCD:

* 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
8. Project :
9. Version :
10. Date    :
11. Author  :
12. Company :
13. Comments:

16. Chip type               : ATmega8535L
17. Program type            : Application
18. AVR Core Clock frequency: 1,000000 MHz
19. Memory model            : Small
20. External RAM size       : 0
21. Data Stack size         : 128
22. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
24. #include <mega8535.h>
26. #include <delay.h>
28. // Alphanumeric LCD functions
29. #include <alcd.h>
31. #define cambio PIND.0
32. #define ha PIND.1
33. #define mm PIND.2
34. #define sd PIND.3
36. **float** cel;
37. **int** tem;
38. **int** desplz;
39. **int** cont\_antidelay,time\_antidelay;
40. bit btnp,btna;
41. unsigned **char** unidades,decenas,decimas,cn,seg=0,min=0,hor=0,dia=25,mes=10,change;
42. unsigned **short** ye=19,ar=97;
43. **const** **char** car=48; //codigo ascii
45. // Declare your global variables here
47. #define ADC\_VREF\_TYPE ((0<<REFS1) | (1<<REFS0) | (1<<ADLAR))
49. // Read the 8 most significant bits
50. // of the AD conversion result
51. unsigned **char** read\_adc(unsigned **char** adc\_input)
52. {
53. ADMUX=adc\_input | ADC\_VREF\_TYPE;
54. // Delay needed for the stabilization of the ADC input voltage
55. delay\_us(10);
56. // Start the AD conversion
57. ADCSRA|=(1<<ADSC);
58. // Wait for the AD conversion to complete
59. **while** ((ADCSRA & (1<<ADIF))==0);
60. ADCSRA|=(1<<ADIF);
61. **return** ADCH;
62. }
64. **void** main(**void**)
65. {
66. // Declare your local variables here
68. // Input/Output Ports initialization
69. // Port A initialization
70. // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
71. DDRA=(0<<DDA7) | (0<<DDA6) | (0<<DDA5) | (0<<DDA4) | (0<<DDA3) | (0<<DDA2) | (0<<DDA1) | (0<<DDA0);
72. // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
73. PORTA=(0<<PORTA7) | (0<<PORTA6) | (0<<PORTA5) | (0<<PORTA4) | (0<<PORTA3) | (0<<PORTA2) | (0<<PORTA1) | (0<<PORTA0);
75. // Port B initialization
76. // Function: Bit7=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out Bit1=Out Bit0=Out
77. DDRB=(1<<DDB7) | (1<<DDB6) | (1<<DDB5) | (1<<DDB4) | (1<<DDB3) | (1<<DDB2) | (1<<DDB1) | (1<<DDB0);
78. // State: Bit7=0 Bit6=0 Bit5=0 Bit4=0 Bit3=0 Bit2=0 Bit1=0 Bit0=0
79. PORTB=(0<<PORTB7) | (0<<PORTB6) | (0<<PORTB5) | (0<<PORTB4) | (0<<PORTB3) | (0<<PORTB2) | (0<<PORTB1) | (0<<PORTB0);
81. // Port C initialization
82. // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
83. DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (0<<DDC3) | (0<<DDC2) | (0<<DDC1) | (0<<DDC0);
84. // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
85. PORTC=(0<<PORTC7) | (0<<PORTC6) | (0<<PORTC5) | (0<<PORTC4) | (0<<PORTC3) | (0<<PORTC2) | (0<<PORTC1) | (0<<PORTC0);
87. // Port D initialization
88. // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
89. DDRD=(0<<DDD7) | (0<<DDD6) | (0<<DDD5) | (0<<DDD4) | (0<<DDD3) | (0<<DDD2) | (0<<DDD1) | (0<<DDD0);
90. // State: Bit7=P Bit6=P Bit5=P Bit4=P Bit3=P Bit2=P Bit1=P Bit0=P
91. PORTD=(1<<PORTD7) | (1<<PORTD6) | (1<<PORTD5) | (1<<PORTD4) | (1<<PORTD3) | (1<<PORTD2) | (1<<PORTD1) | (1<<PORTD0);
93. // Timer/Counter 0 initialization
94. // Clock source: System Clock
95. // Clock value: Timer 0 Stopped
96. // Mode: Normal top=0xFF
97. // OC0 output: Disconnected
98. TCCR0=(0<<WGM00) | (0<<COM01) | (0<<COM00) | (0<<WGM01) | (0<<CS02) | (0<<CS01) | (0<<CS00);
99. TCNT0=0x00;
100. OCR0=0x00;
102. // Timer/Counter 1 initialization
103. // Clock source: System Clock
104. // Clock value: Timer1 Stopped
105. // Mode: Normal top=0xFFFF
106. // OC1A output: Disconnected
107. // OC1B output: Disconnected
108. // Noise Canceler: Off
109. // Input Capture on Falling Edge
110. // Timer1 Overflow Interrupt: Off
111. // Input Capture Interrupt: Off
112. // Compare A Match Interrupt: Off
113. // Compare B Match Interrupt: Off
114. TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) | (0<<COM1B0) | (0<<WGM11) | (0<<WGM10);
115. TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<WGM13) | (0<<WGM12) | (0<<CS12) | (0<<CS11) | (0<<CS10);
116. TCNT1H=0x00;
117. TCNT1L=0x00;
118. ICR1H=0x00;
119. ICR1L=0x00;
120. OCR1AH=0x00;
121. OCR1AL=0x00;
122. OCR1BH=0x00;
123. OCR1BL=0x00;
125. // Timer/Counter 2 initialization
126. // Clock source: System Clock
127. // Clock value: Timer2 Stopped
128. // Mode: Normal top=0xFF
129. // OC2 output: Disconnected
130. ASSR=0<<AS2;
131. TCCR2=(0<<WGM20) | (0<<COM21) | (0<<COM20) | (0<<WGM21) | (0<<CS22) | (0<<CS21) | (0<<CS20);
132. TCNT2=0x00;
133. OCR2=0x00;
135. // Timer(s)/Counter(s) Interrupt(s) initialization
136. TIMSK=(0<<OCIE2) | (0<<TOIE2) | (0<<TICIE1) | (0<<OCIE1A) | (0<<OCIE1B) | (0<<TOIE1) | (0<<OCIE0) | (0<<TOIE0);
138. // External Interrupt(s) initialization
139. // INT0: Off
140. // INT1: Off
141. // INT2: Off
142. MCUCR=(0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);
143. MCUCSR=(0<<ISC2);
145. // USART initialization
146. // USART disabled
147. UCSRB=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) | (0<<RXEN) | (0<<TXEN) | (0<<UCSZ2) | (0<<RXB8) | (0<<TXB8);
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<<ACIS0);
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);
164. // SPI initialization
165. // SPI disabled
166. SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) | (0<<CPOL) | (0<<CPHA) | (0<<SPR1) | (0<<SPR0);
168. // TWI initialization
169. // TWI disabled
170. TWCR=(0<<TWEA) | (0<<TWSTA) | (0<<TWSTO) | (0<<TWEN) | (0<<TWIE);
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)
192. btna=0;
193. **else**
194. btna=1;
195. **if**((btnp==1)&&(btna==0)){
197. **if**(change==0){
199. change=1;
200. }
201. **else**{
202. change=0;
203. }
204. }
205. btnp=btna;



210. lcd\_gotoxy(11,0);
211. lcd\_putsf("ESCOM");

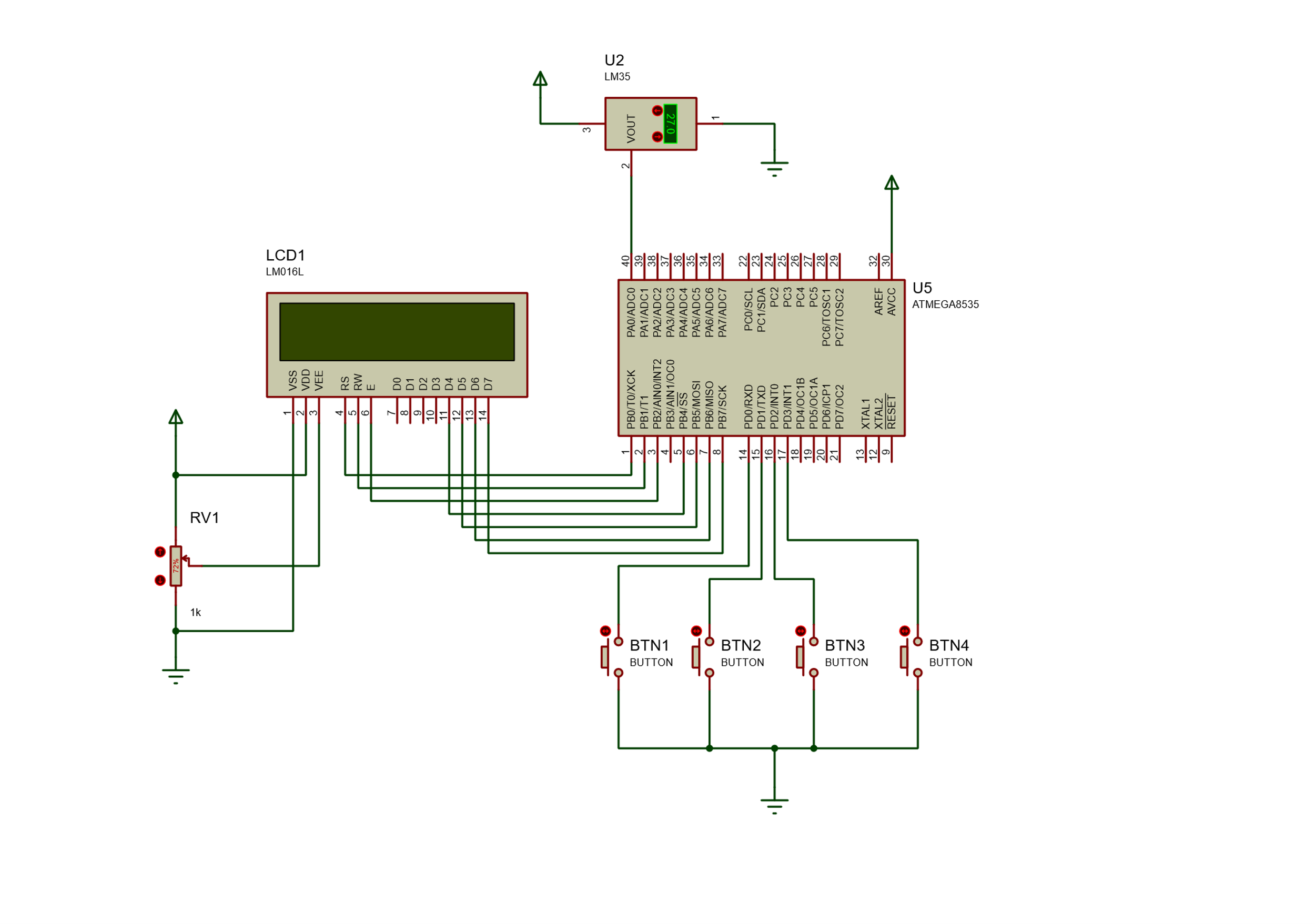
214. cn=read\_adc(0);
215. cel=cn\*1.45;
216. **if**(cel>99)
217. cel=99;
218. tem=cel\*10;
219. decenas=tem/100;
220. tem%=100;
221. decimas=tem%10;
222. unidades=tem/10;

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);
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. }
281. }
282. **if**(sd==0){
283. **if**(cont\_antidelay>time\_antidelay){
284. cont\_antidelay=0;
285. dia++;
286. }**else**{
287. cont\_antidelay++;
288. }
289. }
290. }

293. **if**(desplz>49){
294. desplz=0;seg++;
295. }**else**{
296. desplz++;
297. }
298. **if**(seg>59){
300. min++;
301. seg=0;
302. }
303. **if**(min>59){
305. hor++;
306. min=0;
307. seg=0;
309. }
310. **if**(hor>23){
312. dia++;
313. hor=0;
314. seg=0;
315. min=0;
316. }
318. **if**(dia>31){
319. mes++;
320. dia=0;
321. }
322. **if**(mes>12){
323. ar++;
324. mes=0;
325. **if**(ar>99){
326. ye++;
327. ar=0;
328. }
329. }
330. ///////////////////////////////////////////////hora//////////////////////////
331. lcd\_gotoxy(0,1);
332. lcd\_putchar(hor/10+car);
333. lcd\_gotoxy(1,1);
334. lcd\_putchar(hor%10+car);
336. lcd\_gotoxy(2,1);
337. lcd\_putchar(':');
339. lcd\_gotoxy(3,1);
340. lcd\_putchar(min/10+car);
341. lcd\_gotoxy(4,1);
342. lcd\_putchar(min%10+car);
344. lcd\_gotoxy(5,1);
345. lcd\_putchar(':');
347. lcd\_gotoxy(6,1);
348. lcd\_putchar(seg/10+car);
349. lcd\_gotoxy(7,1);
350. lcd\_putchar(seg%10+car);
352. ////////////////////////////////////////////fecha///////////////////////////////////////////
354. lcd\_gotoxy(0,0);
355. lcd\_putchar(ye/10+car);
356. lcd\_gotoxy(1,0);
357. lcd\_putchar(ye%10+car);
358. lcd\_gotoxy(2,0);
359. lcd\_putchar(ar/10+car);
360. lcd\_gotoxy(3,0);
361. lcd\_putchar(ar%10+car);


365. lcd\_gotoxy(4,0);
366. lcd\_putchar('-');
368. lcd\_gotoxy(5,0);
369. lcd\_putchar(mes/10+car);
370. lcd\_gotoxy(6,0);
371. lcd\_putchar(mes%10+car);
373. lcd\_gotoxy(7,0);
374. lcd\_putchar('-');
376. lcd\_gotoxy(8,0);
377. lcd\_putchar(dia/10+car);
378. lcd\_gotoxy(9,0);
379. lcd\_putchar(dia%10+car);
380. }
381. }

**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.. |