Instituto Politécnico Nacional

Escuela Superior de Computo

Materia:

Introducción a los microcontroladores.

Profesor:

Sanchez Aguilar Fernando

Alumnos:

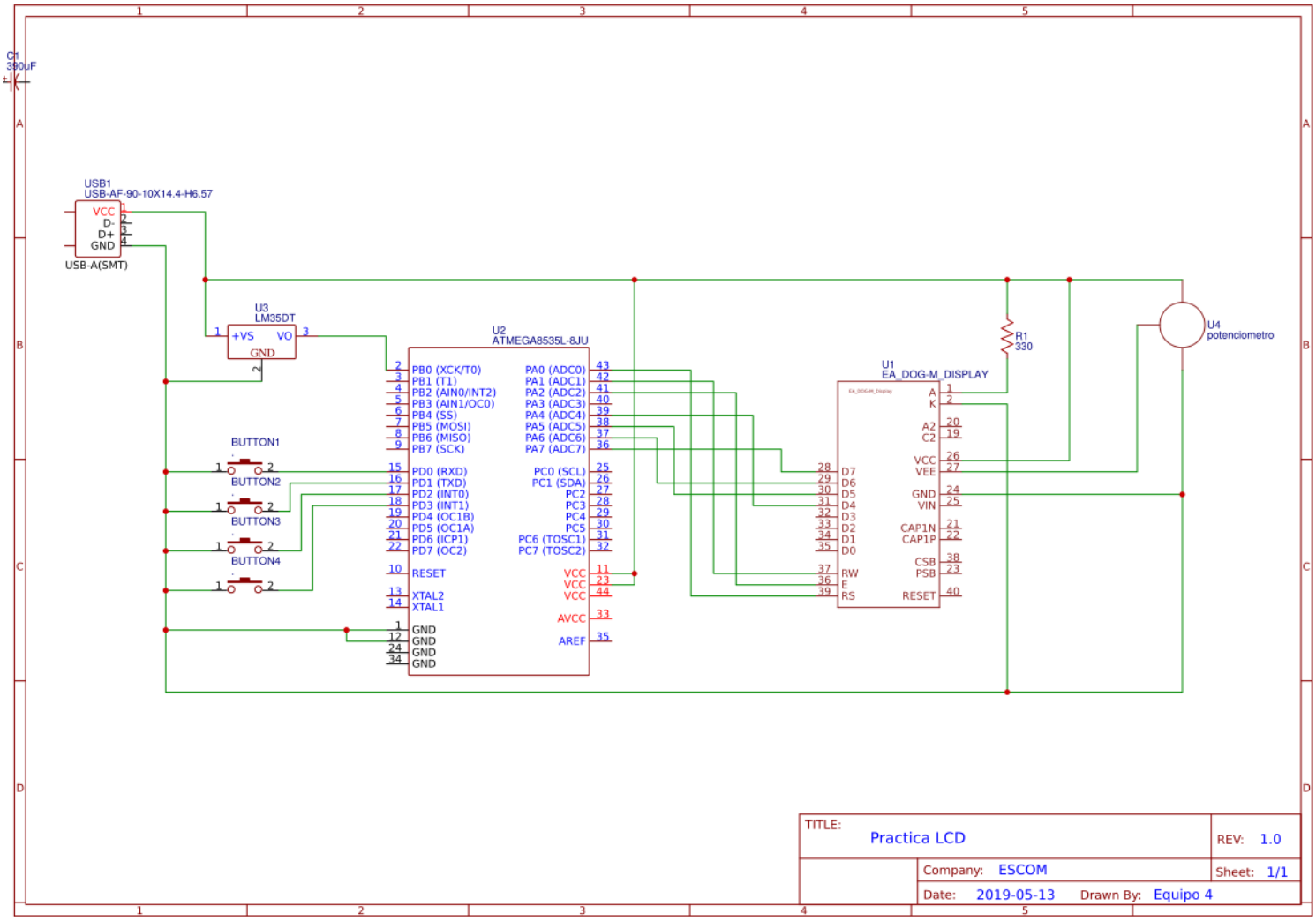
Aldavera Gallaga Iván

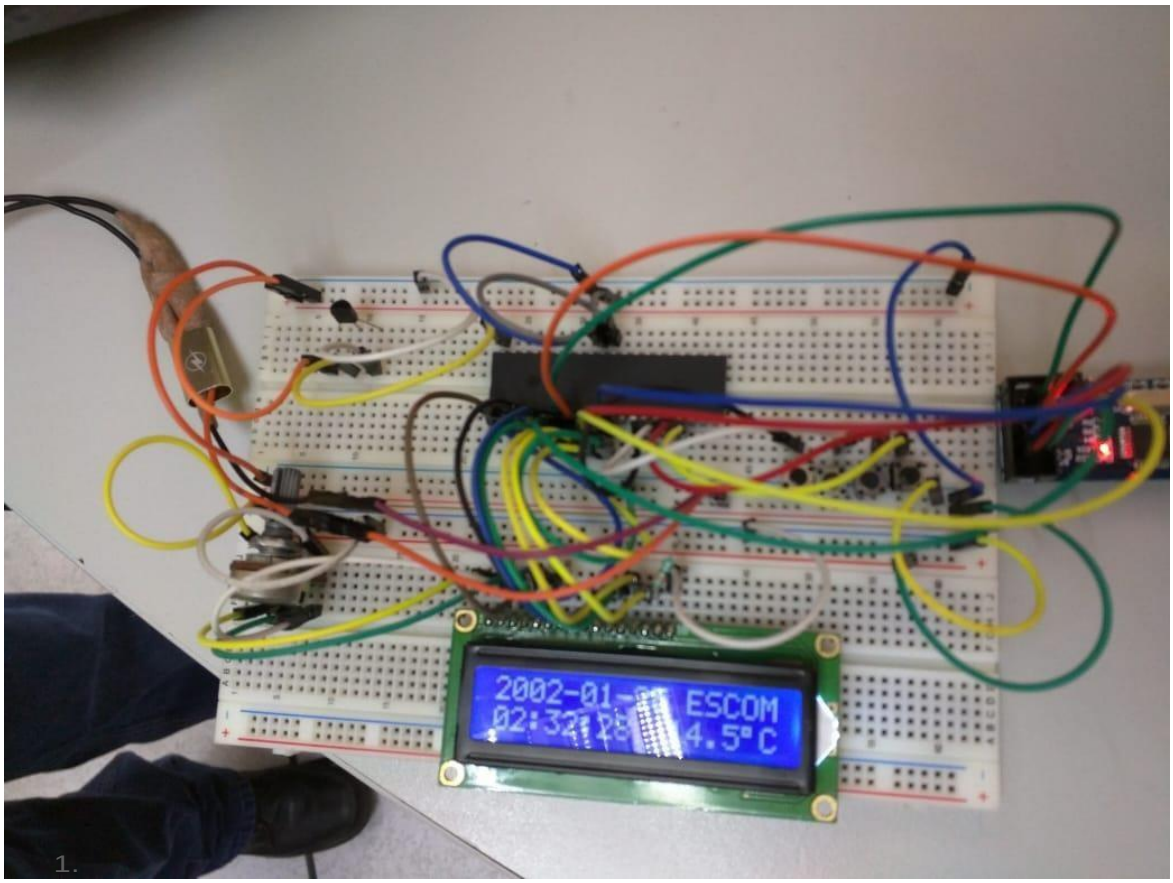
Lara Soto Rubén Jair

Morales Castellanos Adolfo Erik

Practica N°18

LCD


C1
330uF
USB1
USB-AF-90-10X14.4-H6.57
VCC
D-
D+
GND
USB-A(SMT)
U3
LM35DT
+VS
VO
GND
U2
ATMEGA8535L-8JU
PB0 (XCK/T0)
PB1 (T1)
PB2 (AIN0/INT2)
PB3 (AIN1/OC0)
PB4 (SS)
PB5 (MOSI)
PB6 (MISO)
PB7 (SCK)
PD0 (RXD)
PD1 (TXD)
PD2 (INT0)
PD3 (INT1)
PD4 (OC1B)
PD5 (OC1A)
PD6 (ICP1)
PD7 (OC2)
RESET
XTAL2
XTAL1
GND
GND
GND
GND
PA0 (ADC0)
PA1 (ADC1)
PA2 (ADC2)
PA3 (ADC3)
PA4 (ADC4)
PA5 (ADC5)
PA6 (ADC6)
PA7 (ADC7)
PC0 (SCL)
PC1 (SDA)
PC2
PC3
PC4
PC5
PC6 (TOSC1)
PC7 (TOSC2)
VCC
VCC
VCC
AVCC
AREF
BUTTON1
BUTTON2
BUTTON3
BUTTON4
U1
EA_DOG-M_DISPLAY
A
K
A2
C2
VCC
VEE
GND
VIN
CAP1N
CAP1P
CSB
PSB
RESET
D7
D6
D5
D4
D3
D2
D1
D0
RW
E
RS
R1
330
U4
potenciometro
TITLE:
Practica LCD
REV: 1.0
Company: ESCOM
Sheet: 1/1
Date: 2019-05-13
Drawn By: Equipo 4


ESCOM
4.5°C

```
/*****************************************************
This program was created by the
CodeWizardAVR V2.60 Evaluation
Automatic Program Generator
© Copyright 1998-2012 Pavel Haiduc, HP InfoTech S.R.L.
http://www.hpinfotech.com

Project :
Version :
Date    : 16/04/2019
Author  : Equipo 4
Company :
Comments:


Chip type               : ATmega8535L
Program type            : Application
AVR Core Clock frequency: 1,000000 MHz
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;
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);
```

```
61. return ADCH;
62. }
63.
64. void main(void)
65. {
66. // Declare your local variables here
67.
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<<D
    DA1) | (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<<PO
    RTA2) | (0<<PORTA1) | (0<<PORTA0);
74.
75. // Port B initialization
76. // Function: Bit7=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out Bit1=Out Bit0=O
    ut
77. DDRB=(1<<DDB7) | (1<<DDB6) | (1<<DDB5) | (1<<DDB4) | (1<<DDB3) | (1<<DDB2) | (1<<D
    DB1) | (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<<PO
    RTB2) | (0<<PORTB1) | (0<<PORTB0);
80.
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<<D
    DC1) | (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<<PO
    RTC2) | (0<<PORTC1) | (0<<PORTC0);
86.
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<<D
    DD1) | (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<<PO
    RTD2) | (1<<PORTD1) | (1<<PORTD0);
92.
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;
101.
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;
124.
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;
134.
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);
137.
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);
144.
145.       // USART initialization
146.       // USART disabled
147.       UCSRB=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) | (0<<RXEN) | (0<<TXEN) | (0<<U
    CSZ2) | (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<<ACIS0);
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<<AD
    PS2) | (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<<SPR0);
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.
185.      while (1)
186.            {
187.
188.                  if(cambio==0)
189.
190.                      btna=0;
191.                      else
192.                      btna=1;
193.                  if((btnp==1)&&(btna==0)){
194.
195.                  if(change==0){
196.
197.                  change=1;
198.                  }
199.                    else{
200.                    change=0;
201.                    }
202.                  }
203.                  btnp=btna;
204.
205.
206.
207.
208.                lcd_gotoxy(11,0);
209.                lcd_putsf("ESCOM");
210.
211.
212.                cn=read_adc(0);
213.                cel=cn*1.45;
214.                if(cel>99)
215.                cel=99;
216.                tem=cel*10;
217.                decenas=tem/100;
218.                tem%=100;
219.                decimas=tem%10;
220.                unidades=tem/10;
221.
222.
223.                lcd_gotoxy(10,1);
224.                lcd_putchar(decenas+car);
225.                lcd_gotoxy(11,1);
```

```
                lcd_putchar(unidades+car);
                lcd_gotoxy(12,1);
                lcd_putchar('.');
                lcd_gotoxy(13,1);
                lcd_putchar(decimas+car);

                lcd_gotoxy(14,1);
                lcd_putchar(car+175);
                lcd_gotoxy(15,1);
                lcd_putchar('C');
                ////////////////////reloj en mov////////////////////////////
                if(change==1){
                  if(ha==0){
                  hor++;
                  delay_ms(300);
                  }
                  if(mm==0){
                  min++;
                  delay_ms(300);
                  }
                  if(sd==0){
                  seg++;
                  delay_ms(300);
                  }
                }else{
                  if(ha==0){
                  ar++;
                  if(ar>99){
                  ye++;
                  ar=0;
                  }
                  delay_ms(300);
                  }
                  if(mm==0){
                  mes++;
                  delay_ms(300);
                  }
                  if(sd==0){
                  dia++;
                  delay_ms(300);
                  }
                }


                delay_ms(300);
                seg++;
                if(seg>59){

                min++;
                seg=0;
                }
                if(min>59){

                hor++;
                min=0;
                seg=0;

                }
                if(hor>23){

                dia++;
```

```
287.                hor=0;
288.                seg=0;
289.                min=0;
290.                }
291.
292.                if(dia>31){
293.                mes++;
294.                dia=0;
295.                }
296.                if(mes>12){
297.                ar++;
298.                mes=0;
299.                if(ar>99){
300.                ye++;
301.                ar=0;
302.                }
303.                }
304.                 ////////////////////////////////////////////////hora//////////
    //////////////
305.                lcd_gotoxy(0,1);
306.                lcd_putchar(hor/10+car);
307.                lcd_gotoxy(1,1);
308.                lcd_putchar(hor%10+car);
309.
310.                lcd_gotoxy(2,1);
311.                lcd_putchar(':');
312.
313.                lcd_gotoxy(3,1);
314.                lcd_putchar(min/10+car);
315.                lcd_gotoxy(4,1);
316.                lcd_putchar(min%10+car);
317.
318.                lcd_gotoxy(5,1);
319.                lcd_putchar(':');
320.
321.                lcd_gotoxy(6,1);
322.                lcd_putchar(seg/10+car);
323.                lcd_gotoxy(7,1);
324.                lcd_putchar(seg%10+car);
325.
326.                ////////////////////////////////////////////fecha///////////////
    ////////////////////////
327.
328.                lcd_gotoxy(0,0);
329.                lcd_putchar(ye/10+car);
330.                lcd_gotoxy(1,0);
331.                lcd_putchar(ye%10+car);
332.                lcd_gotoxy(2,0);
333.                lcd_putchar(ar/10+car);
334.                lcd_gotoxy(3,0);
335.                lcd_putchar(ar%10+car);
336.
337.
338.
339.                lcd_gotoxy(4,0);
340.                lcd_putchar('-');
341.
342.                lcd_gotoxy(5,0);
343.                lcd_putchar(mes/10+car);
344.                lcd_gotoxy(6,0);
345.                lcd_putchar(mes%10+car);
```

```
346.
347.                 lcd_gotoxy(7,0);
348.                 lcd_putchar('-');
349.
350.                 lcd_gotoxy(8,0);
351.                 lcd_putchar(dia/10+car);
352.                 lcd_gotoxy(9,0);
353.                 lcd_putchar(dia%10+car);
354.             }
355.     }
```