

Microcontroladores

Laboratorio Sesión 9

Semestre: 2021-1

Profesor: Kalun José Lau Gan

1

Preguntas previas:

- ¿Cómo soldar bien?

- Tener buenas herramientas: cautín con control de temperatura, pinzas, pelacables, desarmadores, **pasta para soldar (flux)**, soldadura de buena calidad, esponja humedecida con agua, alcohol isopropílico.
- Practicar y desarrollar la habilidad (mirar videos instructionales)
- Verificar que lo soldado tenga una forma cónica, si esta en circunferencia significa que es un soldado frío.

- ¿Empresas que desarrollan PCBs?

- Jlcpcb.com
- Pcbway.com

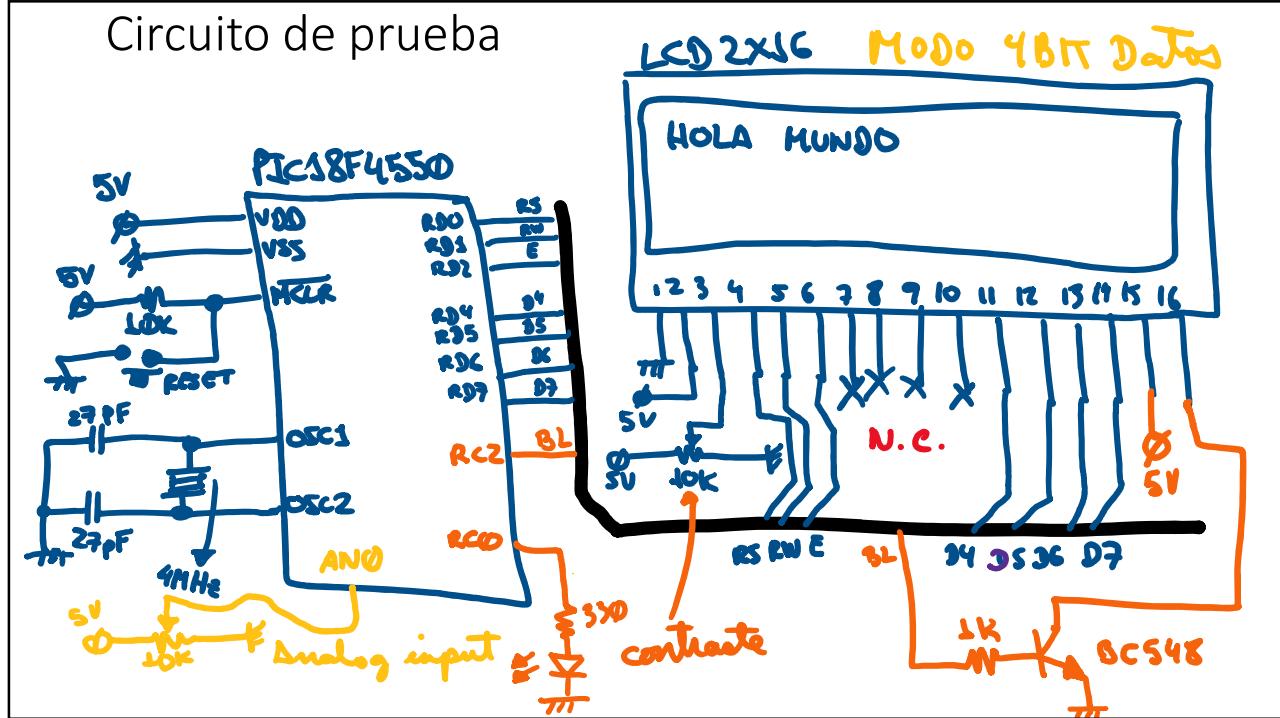
2

Agenda:

- Aspectos introductorios del XC8
- El display LCD alfanumérico HD44780
- Librería LCD para XC8
- El módulo conversor A/D en XC8

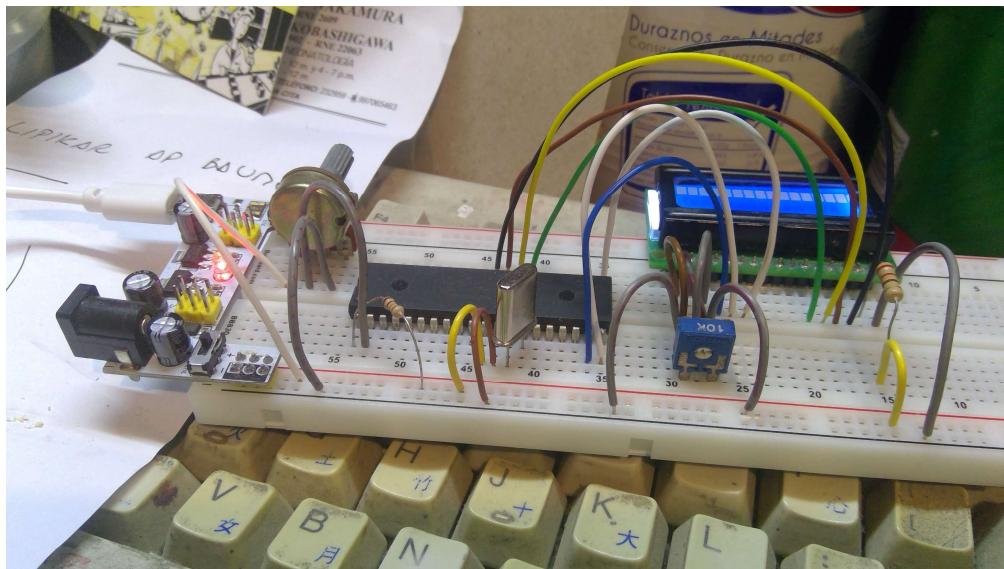
3

Circuito de prueba



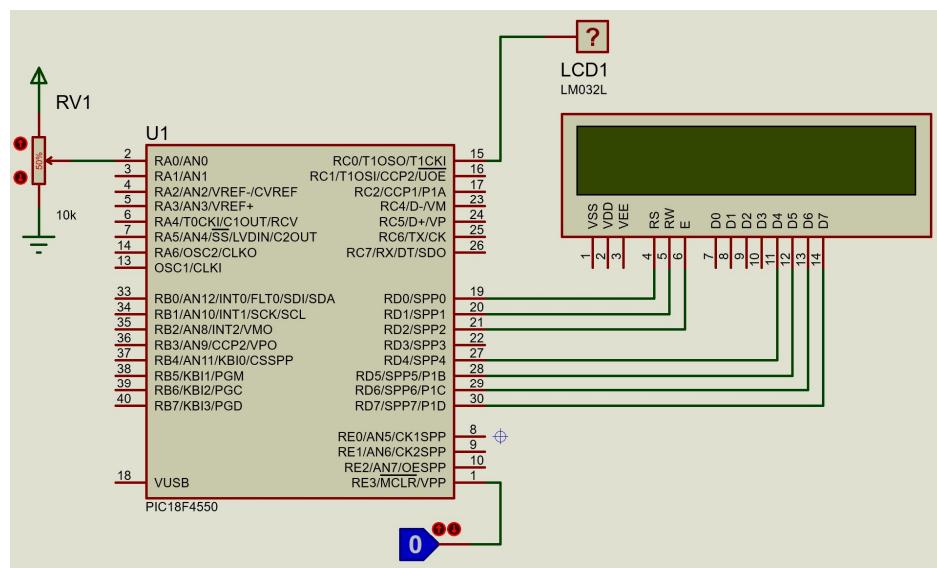
4

Implementación física



5

Circuito simulado en Proteus



6

Primer ejemplo en XC8: titilador de un bit por RC0

Diagrama de flujo:



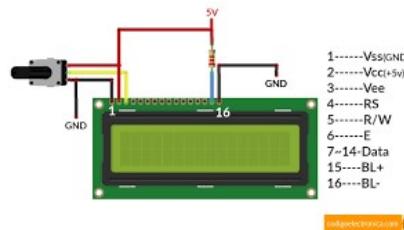
```

4 #pragma config PLLDIV = 1           // PLL Prescaler Selection
5 #pragma config CPUDIV = OSC1_PLL2 // System Clock Post
6 #pragma config FOSC = XTPLL_XT   // Oscillator Selection
7 #pragma config PWRT = ON          // Power-up Timer Enable
8 #pragma config BOR = OFF          // Brown-out Reset Enable
9 #pragma config WDT = OFF          // Watchdog Timer Enable
10 #pragma config CCP2MX = ON         // CCP2 MUX bit (CCP2)
11 #pragma config PBADEN = OFF        // PORTB A/D Enable bit
12 #pragma config MCLEN = ON          // MCLR Pin Enable bit
13 #pragma config LVP = OFF          // Single-Supply ICS1
14
15 #include <xc.h>
16 #define _XTAL_FREQ 48000000UL
17 /*El microcontrolador tiene funcionando el PLL con 48MHz*/
18
19 void main(void) {
20     TRISCbits.RC0 = 0;           //RC0 salida
21     while(1){
22         LATCbits.LC0 = 1;        //Encendemos el LED
23         __delay_ms(100);        //retardo de 100ms
24         LATCbits.LC0 = 0;        //Apagamos el LED
25         __delay_ms(100);        //retardo de 100ms
26     }
27 }
28 }
```

7

El LCD alfanumérico HD44780

- Basado en el controlador Hitachi HD44780A
- Diferentes tamaños, desde 1x8 hasta 4x40
- Interface paralela de datos (4 ó 8 bits)
- Tiene control de contraste y luz de fondo
- Posee un ROM de caracteres predefinidos



8

El LCD alfanumérico HD44780

- ROM de caracteres:
 - Muy similar al código ASCII en 7 bits
 - El símbolo de grado (°) en ASCII es Alt+0167, en el ROM de caracteres del HD44780 es 0xDF
 - El símbolo “ñ” en ASCII es Alt+164, en el ROM de caracteres del HD44780 es 0xEE
 - Capacidad de ocho caracteres personalizados (CGRAM 0x00-0x07)

Upper Bit 4	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
Lower Bit 4	00000	00001	00010	00011	01000	01001	01010	01011	10000	10001	10010	10011	11000	11001	11100	11101	11110
	0@P°F								-T@&P								
xxxx0000	(1)																
xxxx0001	(2)	! 1AQ@a							■ 7#4&q								
xxxx0010	(3)	"ZBRbr							「イツ×βθ								
xxxx0011	(4)	#3CScs							」ウテモε								
xxxx0100	(5)	\$4DTdt							、エトトμο								
xxxx0101	(6)	%5EUeu							・オナユσÜ								
xxxx0110	(7)	&6FUVfv							ヲカニヨρΣ								
xxxx0111	(8)	'7GWgwg							アキラガπ								
xxxx1000	(1)	(8Hxhx							イクネリゞ								
xxxx1001	(2))9IYiy							タケル"y								
xxxx1010	(3)	*:JZjz							エコハレj								
xxxx1011	(4)	+;K[k							オサヒロ*								
xxxx1100	(5)	,<L¥11							ヤシフワ¢								
xxxx1101	(6)	-=M]m}							ユスヘンモ								
xxxx1110	(7)	.>N^n>							ヨテホ^n								
xxxx1111	(8)	/?O_o<							ツツマ"ö								

9

El LCD alfanumérico HD44780

- Tabla de caracteres ASCII de 7 bits

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0 000	NUL	(null)	32	20 040	4#32;	Space		64	40 100	4#64;	Ø	ø	96	60 140	4#96;	~	
1	1 001	SOH	(start of heading)	33	21 041	4#33;	!	!	65	41 101	4#65;	A	ª	97	61 141	4#97;	a	
2	2 002	STX	(start of text)	34	22 042	4#34;	"	"	66	42 102	4#66;	B	ª	98	62 142	4#98;	b	
3	3 003	ETX	(end of text)	35	23 043	4#35;	#	#	67	43 103	4#67;	C	ª	99	63 143	4#99;	c	
4	4 004	EOT	(end of transmission)	36	24 044	4#36;	\$	\$	68	44 104	4#68;	D	ª	100	64 144	4#100;	d	
5	5 005	ENQ	(enquiry)	37	25 045	4#37;	%	%	69	45 105	4#69;	E	ª	101	65 145	4#101;	e	
6	6 006	ACK	(acknowledge)	38	26 046	4#38;	&	&	70	46 106	4#70;	F	ª	102	66 146	4#102;	f	
7	7 007	BEL	(bell)	39	27 047	4#39;	*	*	71	47 107	4#71;	G	ª	103	67 147	4#103;	g	
8	8 010	BS	(backspace)	40	28 050	4#40;	{	{	72	48 110	4#72;	H	ª	104	68 150	4#104;	h	
9	9 011	TAB	(horizontal tab)	41	29 051	4#41;))	73	49 111	4#73;	I	ª	105	69 151	4#105;	i	
10	A 012	LF	(NL line feed, new line)	42	2A 052	4#42;	*	*	74	4A 112	4#74;	J	ª	106	6A 152	4#106;	j	
11	B 013	VT	(vertical tab)	43	2B 053	4#43;	+	+	75	4B 113	4#75;	K	ª	107	6B 153	4#107;	k	
12	C 014	FF	(NP form feed, new page)	44	2C 054	4#44;	,	,	76	4C 114	4#76;	L	ª	108	6C 154	4#108;	l	
13	D 015	CR	(carriage return)	45	2D 055	4#45;	-	-	77	4D 115	4#77;	M	ª	109	6D 155	4#109;	m	
14	E 016	SO	(shift out)	46	2E 056	4#46;	.	.	78	4E 116	4#78;	N	ª	110	6E 156	4#110;	n	
15	F 017	SI	(shift in)	47	2F 057	4#47;	/	/	79	4F 117	4#79;	O	ª	111	6F 157	4#111;	o	
16	10 020	DLE	(data link escape)	48	30 060	4#48;	0	0	80	50 120	4#80;	P	ª	112	70 160	4#112;	p	
17	11 021	DC1	(device control 1)	49	31 061	4#49;	1	1	81	51 121	4#81;	Q	ª	113	71 161	4#113;	q	
18	12 022	DC2	(device control 2)	50	32 062	4#50;	2	2	82	52 122	4#82;	R	ª	114	72 162	4#114;	r	
19	13 023	DC3	(device control 3)	51	33 063	4#51;	3	3	83	53 123	4#83;	S	ª	115	73 163	4#115;	s	
20	14 024	DC4	(device control 4)	52	34 064	4#52;	4	4	84	54 124	4#84;	T	ª	116	74 164	4#116;	t	
21	15 025	NAK	(negative acknowledgement)	53	35 065	4#53;	5	5	85	55 125	4#85;	U	ª	117	75 165	4#117;	u	
22	16 026	SYN	(synchronous idle)	54	36 066	4#54;	6	6	86	56 126	4#86;	V	ª	118	76 166	4#118;	v	
23	17 027	ETB	(end of trans. block)	55	37 067	4#55;	7	7	87	57 127	4#87;	W	ª	119	77 167	4#119;	w	
24	18 030	CAN	(cancel)	56	38 070	4#56;	8	8	88	58 130	4#88;	X	ª	120	78 170	4#120;	x	
25	19 031	EM	(end of medium)	57	39 071	4#57;	9	9	89	59 131	4#89;	Y	ª	121	79 171	4#121;	y	
26	1A 032	SUB	(substitute)	58	3A 072	4#58;	:	:	90	5A 132	4#90;	Z	ª	122	7A 172	4#122;	z	
27	1B 033	ESC	(escape)	59	3B 073	4#59;	:	:	91	5B 133	4#91;	[ª	123	7B 173	4#123;	{	
28	1C 034	FS	(file separator)	60	3C 074	4#60;	<	<	92	5C 134	4#92;	\	ª	124	7C 174	4#124;		
29	1D 035	GS	(group separator)	61	3D 075	4#61;	=	=	93	5D 135	4#93;]	ª	125	7D 175	4#125;	}	
30	1E 036	RS	(record separator)	62	3E 076	4#62;	>	>	94	5E 136	4#94;	^	ª	126	7E 176	4#126;	~	
31	1F 037	US	(unit separator)	63	3F 077	4#63;	?	?	95	5F 137	4#95;	_	ª	127	7F 177	4#127;	DEL	

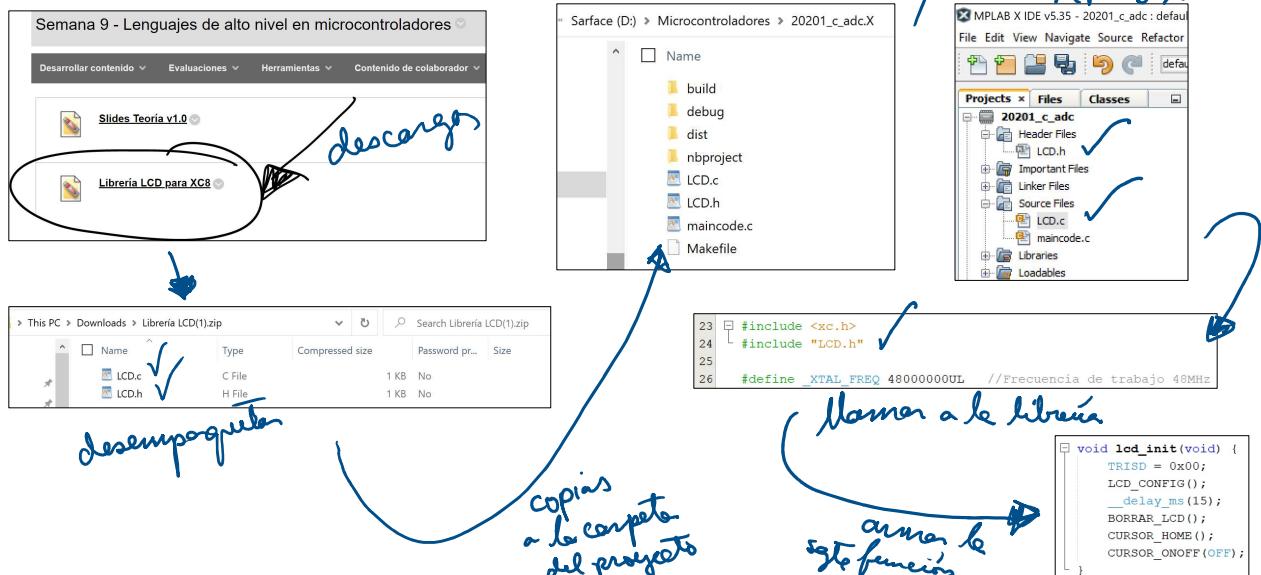
Source: www.LookupTables.com

El LCD alfanumérico HD44780

- Referencia: Hoja técnica del HD44780
 - http://academy.cba.mit.edu/classes/output_devices/44780.pdf
- Para trabajar con el display se ha creado una librería de comandos en la cual posee las siguientes características:
 - Interface de 4 bits
 - Comandos para: Limpiar pantalla, ocultar cursor, pasar de línea, caracteres personalizados, etc.
 - Puerto D empleado
 - Tener en cuenta FOSC especificado dentro de la librería (48MHz)

11

Uso del LCD en XC8 (librería S_SAL)



12

Ejemplo: Visualizar “Hola mundo UPC!” en el LCD

```

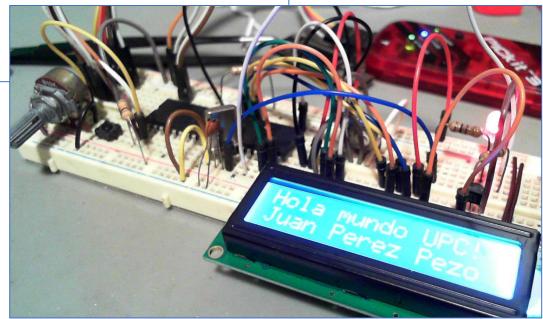
1 //Zona de bits de configuracion
2 #pragma config PLLDIV = 1           // PLL Presca
3 #pragma config CPUDIV = OSC1_PLL2// System Cl
4 #pragma config FOSC = XTPLL_XT // Oscillator
5 #pragma config PWRT = ON           // Pow
6 #pragma config BOR = OFF           // Brow
7 #pragma config BORV = 3            // Brow
8 #pragma config WDT = OFF           // Watchdog
9 #pragma config WDTPS = 32768        // Watchdog
10 #pragma config CCP2MX = ON          // CCP2
11 #pragma config PBADEN = OFF         // PORTB
12 #pragma config MCLE = ON           // MCLE
13 #pragma config LVP = OFF           // Sing
14
15 #include <xc.h>
16 #include "LCD.h"
17 #define _XTAL_FREQ 48000000UL
18
19 void init_conf(void){
20     TRISCBits.RC0 = 0;           //RC0 como salida
21     TRISCBits.RC2 = 0;           //RC2 como salida
22 }
23
24 void lcd_config(void){
25     TRISD = 0x00;
26     LCD_CONFIG();
27     _delay_ms(15);
28     Borrar_LCD();
29     Cursor_Home();
30     Cursor_OnOff(OFF);
31 }

```

```

33 void main(void) {
34     init_conf();
35     lcd_config();
36     LATCbits.LC2 = 1;           //Luz de fondo del LCD encendida
37     ESCRIBE_MENSAJE("Hola mundo UPC!",15);
38     POS_CURSOR(2,0);
39     ESCRIBE_MENSAJE("Juan Perez Pezo",15);
40     while(1){
41         LATCbits.LC0 = 1;
42         _delay_ms(250);
43         LATCbits.LC0 = 0;
44         _delay_ms(250);
45     }
46 }

```



13

Código ejemplo para configurar y leer AN0 en el A/D:

```

28     unsigned int res ad = 0;
29
30 void configuracion(void) {
31     //Aqui colocas las configuraciones iniciales
32     ADCON2 = 0xA4;           //ADFM=0 (just derecha), 8TAD, Fosc/4
33     ADCON1 = 0x0E;           //Canal AN0 habilitado
34     ADCON0 = 0x01;           //Canal AN0 seleccionado y encendemos el A/D
35
36 void main(void) {
37     configuracion();
38
39     while (1) {
40         //Tu programa de usuario
41         ADCON0bits.GODONE = 1;           //Inicio una captura de muestra en AN0
42         while(ADCON0bits.GODONE == 1);   //Espero a que termine de convertir
43         res_ad = (ADRESH << 8) + ADRESL; //Grabar el resultado en la variable res_ad
44         convierte(res_ad);             //Sacar los dígitos
45     }
46 }

```

14

Código en XC8 del ejemplo inicial:

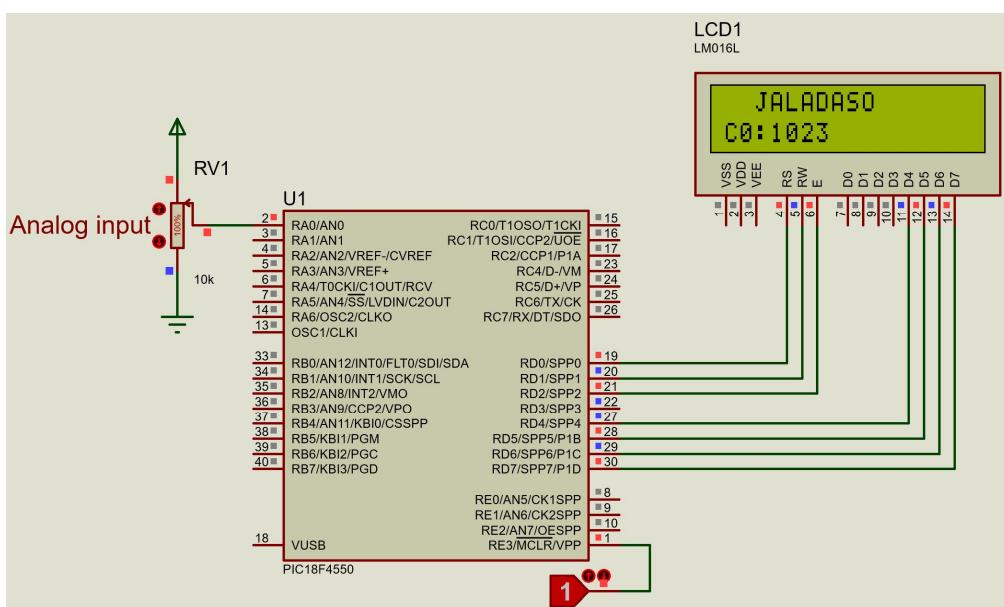
```

26 #define _XTAL_FREQ 48000000UL //Frecuencia
27
28 unsigned int res_ad = 0;
29 unsigned int millar = 0;
30 unsigned int centena = 0;
31 unsigned int decena = 0;
32 unsigned int unidad = 0;
33
34 void convierte(unsigned int numero){
35     millar = numero / 1000;
36     centena = (numero % 1000) / 100;
37     decena = (numero % 100) / 10;
38     unidad = numero % 10;
39 }
40
41 void lcd_init(void) {
42     TRISD = 0x00; //Puerto D configurado como salida
43     LCD_CONFIG();
44     __delay_ms(15);
45     BORRAR_LCD();
46     CURSOR_HOME();
47     CURSOR_ONOFF(OFF);
48 }
49
50 void configuracion(void) {
51     //Aqui colocas las configuraciones
52     ADCON2 = 0xA4; //ADFI
53
54     ADCON1 = 0x0E; //Canal 1
55     ADCON0 = 0x01; //Canal 0
56     lcd_init();
57 }
58
59 void main(void) {
60     configuracion();
61     ESCRIBE_MENSAJE("VIRTUALASO",10);
62     while (1) {
63         //Tu programa de usuario
64         ADCON0bits.GODONE = 1;
65         while(ADCON0bits.GODONE == 1);
66         res_ad = (ADRESH << 8) + ADRESL;
67         convierte(res_ad);
68         POS_CURSOR(2,0);
69         ENVIA_CHAR(millar+0x30);
70         ENVIA_CHAR(centena+0x30);
71         ENVIA_CHAR(decena+0x30);
72         ENVIA_CHAR(unidad+0x30);
73     }
74 }

```

15

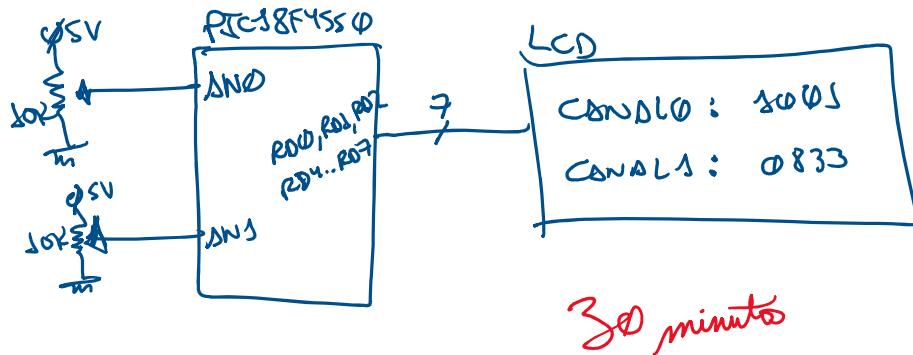
Simulación en Proteus:



16

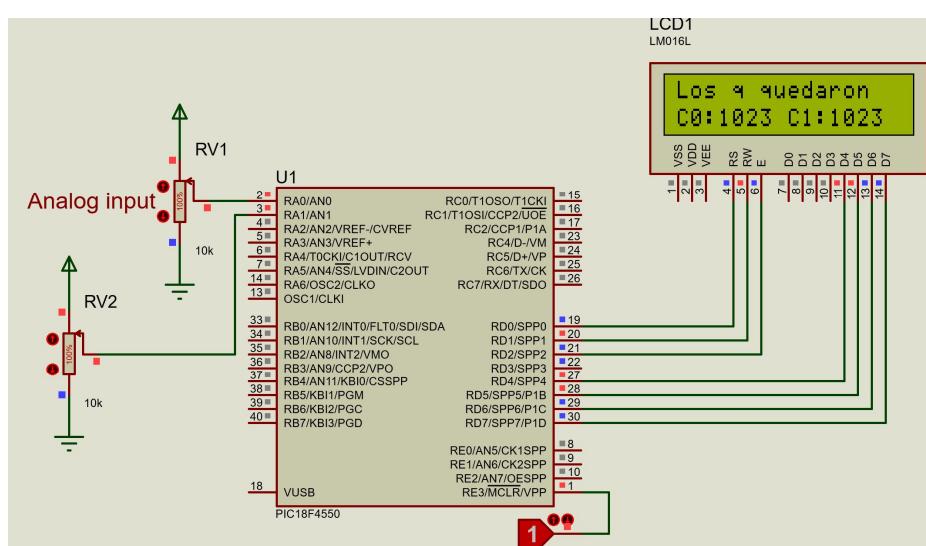
Asignación:

- Leer dos canales analógicos y mostrarlos en el LCD



17

Simulación:



18

Código en XC8:

```

27  void configuracion(void) {
28      //Aquí colocas las configuraciones inic
29      ADCON2 = 0xA4;           //ADFM=0 (
30      ADCON1 = 0x0D;          //Canal AN0
31      ADCONbits.ADON = 1;     //Encendemos
32      lcd_init();
33
34      unsigned int res_ad0 = 0;
35      unsigned int res_ad1 = 0;
36
37      unsigned int millar = 0;
38      unsigned int centena = 0;
39      unsigned int decena = 0;
40      unsigned int unidad = 0;
41
42      void convierte(unsigned int numero){
43          millar = numero /1000;
44          centena = (numero % 1000) / 100;
45          decena = (numero % 100) / 10;
46          unidad = numero % 10;
47
48      void lcd_init(void) {
49          TRISD = 0x00;           //Pue
50          LCD_CONFIG();
51          _delay_ms(15);
52          BORRAR_LCD();
53          CURSOR_HOME();
54          CURSOR_ONOFF(OFF);
55
56      void main(void) {
57          configuracion();        //Llamada a
58          ESCRIBE_MENSAJE("Los q quedaron",14);
59          while(1){
60              ADCON0 = 0x03;
61              //ADCON0bits.GODONE = 1;
62              while(ADCON0bits.GODONE == 1);    //
63              res_ad0 = (ADRESH << 8) + ADRESL;
64              ADCON0 = 0x07;                  //
65              while(ADCON0bits.GODONE == 1);    //
66              res_ad1 = (ADRESH << 8) + ADRESL;
67              POS_CURSOR(2,0);             //S
68              ESCRIBE_MENSAJE("CO:",3);
69              convierte(res_ad0);         //
70              ENVIA_CHAR(millar+0x30);    //E
71              ENVIA_CHAR(centena+0x30);   //E
72              ENVIA_CHAR(decena+0x30);    //E
73              ENVIA_CHAR(unidad+0x30);    //E
74              ESCRIBE_MENSAJE(" Cl:",4);
75              convierte(res_ad1);         //
76              ENVIA_CHAR(millar+0x30);    //E
77              ENVIA_CHAR(centena+0x30);   //E
78              ENVIA_CHAR(decena+0x30);    //E
79              ENVIA_CHAR(unidad+0x30);    //E
80
81
82
83
84
85
86
87
88      }
  
```

19

Fin de la sesión!

20