ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

"Comprobador de circuitos electrónicos digitales tipo DIP con programa embebido en un microcontrolador, presentación de opciones y resultados en una GLCD"

TESINA DE SEMINARIO

PREVIO A LA OBTENCIÓN DEL TÍTULO DE:

INGENIERO EN ELECTRONICA Y TELECOMUNICACIONES

PRESENTADO POR:

Consuelo Alexandra Cerna Pila

Andrea Guadalupe Malla Rodríguez

GUAYAQUIL – ECUADOR

2010

AGRADECIMIENTO

A Dios por acompañarme siempre.

A mi hermana por su apoyo incondicional y a mi familia por darme la fortaleza necesaria para trabajar y alcanzar mis metas.

A la ESPOL y a sus profesores por transmitirme sus conocimientos y brindarme su ayuda.

Consuelo Cerna

DEDICATORIA

Le dedico el presente trabajo a Dios, a mi familia y a mi madre quien desde el cielo me ha guiado para seguir adelante y cumplir con mis objetivos.

Consuelo Cerna

AGRADECIMIENTO

A Dios por guiarme e iluminarme para culminar esta etapa de mi vida, a mis padres por su apoyo incondicional y por brindarme la oportunidad de una excelente educación.

A mis hermanos y a mi novio por darme la fortaleza necesaria para seguir adelante. A los profesores quienes me brindaron sus conocimientos y ayuda.

Andrea G. Malla Rodríguez

DEDICATORIA

A Dios, a mis padres, a mi abuelita, a mi novio y hermanos quienes con su ejemplo de valentía supieron darme la fortaleza necesaria para seguir siempre adelante y cumplir con mis metas.

Andrea G. Malla Rodríguez

TRIBUNAL DE SUSTENTACIÓN

Ing. Carlos Enrique Valdivieso A.
PROFESOR DEL SEMINARIO DE GRADUACIÓN

Ing. Hugo Villavicencio V.
DELEGADO DEL DECANO

DECLARACIÓN EXPRESA

“La responsabilidad del contenido de esta Tesina, nos corresponde Exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL”.

(Reglamento de exámenes y títulos profesionales de la ESPOL).

Consuelo Alexandra Cerna Pila

Andrea Guadalupe Malla Rodríguez

RESUMEN

El comprobador de Circuitos Integrados (C.I) de tipo DIP es utilizado en laboratorios en los que se trabaje con el tipo de CHIPS de la familia TTL y CMOS.

En este proyecto que a continuación se describe la implementación de un prototipo de comprobador con una limitada lista de circuitos integrados con ayuda de dos microcontroladores, y con la pantalla GLCD como interfaz visual.

Este prototipo es de ámbito educacional e investigativo de uso puntual aprovechando la facilidad del MicroProC for PIC y el manejo de microcontroladores como lo son el 18f458 y el 18f4520 que controla la pantalla GLCD y comprueba el correcto funcionamiento de los Circuitos Integrados a analizar respectivamente.

ABREVIACIONES

GLCD Graphic Liquid Crystal Display (Pantalla de cristal líquido)

C.I . Circuito integrado

TTL Transistor Transistor Lógica

DC Direct Current (Corriente directa)

DIP Dual in-line package (Empaque de doble línea)

FIEC Facultad de Ingeniería de Electricidad y Computación

RAM Random-access Memory (Memoria de Acceso Aleatorio)

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE TABLAS

INTRODUCCIÓN

El presente proyecto tiene como finalidad la construcción de un comprobador de C.I., en el cual, el ingreso de la serie del integrado se efectuará a través un teclado matricial, cuyos resultados y el menú de este proyecto serán visualizados en la pantalla GLCD.

La implementación se la realizará con dos microcontroladores de la serie 18f458 y 18f4520, una pantalla GLCD como interfaz visual, y el teclado matricial.

En el primer capítulo, se detalla los antecedentes, la descripción del proyecto, planteamiento del problema, aplicaciones, limitaciones y el alcance que tiene, así como también la estrategia implementada y los productos similares en el mercado.

En el segundo capítulo, se describen las herramientas de hardware utilizadas, que incluyen equipos y materiales adicionales en la construcción del comparador de circuitos integrados. Además se detalla el software utilizado, MikroPro C, que es la principal herramienta de programación del PIC 18F458 y PIC18F4520, por lo que se da una breve descripción de las funciones utilizadas para desarrollar este proyecto.

Además se utilizó Proteus versión 7.6, un potente software que nos permite la simulación del proyecto y así ir modificando sentencias hasta llegar a nuestro objetivo.

En el tercer capítulo, se detalla el diseño general, empezando con la implementación física y pasando luego a la programación de los dispositivos.

En el cuarto y último capítulo encontramos todas las simulaciones realizadas, detalles de la implementación y datos obtenidos en diferentes pruebas.

Finalmente se exponen las conclusiones y recomendaciones.

CAPÍTULO 1

1. DESCRIPCIÓN GENERAL DEL PROYECTO

1.1 Antecedentes

En muchos casos, el determinar manualmente el correcto funcionamiento de un C.I. es trabajoso, exhausto puesto que algunos de estos integrados son de funcionamiento complejo y requerirían de ciertas pruebas de elevada complejidad, además de un gran tiempo empleado. Estos problemas son más evidentes en casas de estudio o en cualquier otro lugar donde el uso de circuitos integrados lógicos es de uso frecuente.

Considerando las necesidades y los requerimientos del laboratorio de digitales de la FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN (FIEC), que cuenta con un solo analizador de dispositivos integrados mediante software que tiene sus limitantes, como la dependencia de una computadora para ser operable,

además la fuente de alimentación es convencional, al no haber energía eléctrica no se lo podrá utilizar, por esta y muchas razones, se planteó como solución el desarrollo de un módulo portátil y autónomo de fácil manipulación, facultando al estudiante su operabilidad sin asistencia del ayudante encargado, el cual agilitará y ahorrará tiempo.

1.2 Descripción del proyecto

Este proyecto analizará el correcto funcionamiento de circuitos integrados de empaquetamiento tipo DIP de la familia TTL y CMOS cuya lista se especificará más adelante. (Ver ANEXO).

En el sistema se utilizará dos microcontroladores 18F4520 y 18F458, una pantalla GLCD, un teclado matricial 4X4 y un zócalo de 24 pines.

El usuario ingresará el tipo de CHIP colocado en el zócalo mediante teclado, se podrá observar en la pantalla GLCD la serie del mismo en caso de que este exista en el comprobador continuará con el proceso, arrojando como resultado “OK”, si el C.I. está en buen estado, “ERROR”, si está defectuoso, caso contrario, se observará un mensaje “ NO EXISTE ”, el sistema se reiniciara y partirá del principio, en cualquier caso.

1.2.1 Estrategia de implementado

El Microcontrolador 18f458 es muy versátil y de capacidad de memoria de programa (32kB), será utilizado para la interacción entre el usuario, mediante la pantalla GLCD y el teclado matricial 4X4 el microcontrolador generará un código de 8 bits que será recibido por el Microcontrolador 18f4520, este identificará que tipo de integrado seleccionó el usuario, al verificar su existencia enviará datos a las entradas del CHIP que estamos evaluando, se verificará y comprobará la respuesta que debe dar según los datos enviados, tomando en consideración que este funcionando correctamente, con las respuestas que arroja las salidas del integrado si coinciden se considera que esta correcto, caso contrario, se encuentra dañado el CHIP el segundo microcontrolador (18f4520) enviará una señal compuesta de dos bits al primero (18f458), según sea el código se observará en la pantalla GLCD el mensaje de “OK”, de buen estado, “ERROR” , si está defectuoso.

1.2.2 Limitaciones del Proyecto

Una de las limitaciones del proyecto es la cantidad de integrados a ser analizados que en nuestro caso es ciento treinta y dos, cabe recalcar que los integrados de tipo DIP se refiere a un tipo de empaquetamiento y existen una gran cantidad en el mercado, se impuso este alcance considerado los integrados utilizados en el laboratorio de digitales, además por la capacidad de memoria limitante en los microcontroladores utilizados, dentro de este grupo que se analizará no se consideran integrados de más de veinticuatro pines. Por otro lado, al ser su fuente de alimentación una batería, hace que al descargarse deje de funcionar.

Debido a que no tiene interfaces con otros dispositivos como sería una computadora, en caso de dañarse la pantalla GLCD no podrá existir interacción con el usuario.

1.3 Análisis de Soluciones existentes en el Mercado

Existen varios analizadores de integrados tipo DIP en el mercado de mayor capacidad realizados por empresas especializadas en la medición tanto de integrados como elementos digitales y electrónicos.

También existen otros analizadores mediante software con otro tipo de interfaz y de mayor alcance como Setup Electrónica CIs analógicos y digitales, Probador de C.I 570ª, Probador de C.I 575A , etc

1.3.1 Probador de C.I 570ª

Fig. 1. 1 Probador de C.I. 570A

El modelo analógico 570A Probador de IC incorporado en la colección de prueba incluye todos los CI común Devices incluyendo amplificadores, comparadores, reguladores de voltaje, referencias de tensión, interruptor analógico y multicines, aisladores y opto-acopladores, y circuitos integrados de audio.

- Modo de identificación automática.
- Prueba incondicional modo bucle.
- Muestra información de diagnóstico hasta los pines de componentes individuales.

- Robusto, de mano y con operador de baterías.

1.3.2 Probador de C.I. 575

El BK Precisión 575A es capaz de localizar fallas intermitentes, fallas relacionadas con temperatura mediante el modo de uso de pruebas de lazo condicionales o incondicionales. La identificación de dispositivos desconocidos se hace fácilmente seleccionando el botón de SEARCH (búsqueda) del menú, solo seleccione el número de terminales en el dispositivo y active el modo de búsqueda. El 575A encontrara e identificara de su libraría el dispositivo, mostrando los equivalentes funcionales para remplazo. Como parte de la prueba al IC, marca el modelo especifico de IC, su descripción funcional y el estatus de las terminales que no funcionan correctamente a través de un display propio que se desplaza o rolla información.

- Librería de dispositivos que cubre lógica TTL, CMOS, IC's de memoria y dispositivos de interface.
- Socket de 40 pines con capacidad para (compuertas NAND o CPUs).
- Identifica dispositivos sin marca o codificados en casa.
- Detecta fallas intermitentes o relacionadas con temperatura
- Muestra información de diagnostico por pin individual.

- Operación a baterías.
- Útil en escuelas que prestan integrados y desean inspeccionar el estado de entrega y recepción de los integrados.

CAPÍTULO 2

2. Fundamento Teórico

El presente capítulo se trata fundamentos teóricos en los que se basa el proyecto, los cuales involucran aspectos de programación, lógica y de implementación.

2.1. Familia lógica TTL

Las características de la tecnología utilizada, en la familia TTL (Transistor, Transistor Logic), condiciona los parámetros que se describen en sus hojas de características según el fabricante, (aunque es estándar)

Su tensión de alimentación característica se halla comprendida entre los 4,75V y los 5,25V como se ve un rango muy estrecho debido a esto, los niveles lógicos vienen definidos por el rango de tensión comprendida entre 0,2V y 0,8V para el estado **L** y los 2,4V y Vcc para el estado **H**.

La velocidad de transmisión entre los estados lógicos es su mejor base, ciertamente esta característica le hace aumentar su consumo siendo su

mayor enemigo. Motivo por el cual han aparecido diferentes versiones de TTL como FAST, SL, S, etc. y últimamente los TTL: HC, HCT y HCTLS. En algunos casos puede alcanzar poco más de los 250Mhz.

Esta familia es la primera que surge y aún todavía se utiliza en aplicaciones que requieren dispositivos SSI y MSI. El circuito lógico TTL básico es la compuerta NAND. La familia TTL utiliza como componente principal el transistor bipolar, mediante un arreglo de estos transistores se logran crear distintos circuitos de lógica digital.

Característica	Mínimo	Típico	Máximo	Unidad
V_{CC} (Voltaje de alimentación)	4.75	5.00	5.25	V
V_{OL} (Voltaje de salida lógica 0)	–	–	0.40	V
V_{OH} (Voltaje de salida lógica 1)	2.40	–	–	V
V_{IL} (Voltaje de entrada lógica 0)	–	–	0.80	V
V_{IH} (Voltaje de entrada lógica 1)	2.00	–	–	V
P_D (Potencia disipada promedio/comp.)	2.00	10.0	–	mW
T_{pd} (Tiempo de propagación)	–	15	–	ns
Fan out	–	10	–	compuertas
Margen de ruido	–	0.40	–	V

Tabla 2.1 Características de la familia TTL

2.2 Familia lógica CMOS

Existen varias series en la familia CMOS de circuitos integrados digitales. La serie 4000 que fue introducida por RCA y la serie 14000 por

Motorola, estas fueron las primeras series CMOS. La serie 74C que su característica principal es que es compatible terminal por terminal y función por función con los dispositivos TTL. Esto hace posibles remplazar algunos circuitos TTL por un diseño equivalente CMOS. La serie 74HC son los CMOS de alta velocidad, tienen un aumento de 10 veces la velocidad de conmutación. La serie 74HCT es también de alta velocidad, y también es compatible en lo que respecta a los voltajes con los dispositivos TTL.

Los voltajes de alimentación en la familia CMOS tiene un rango muy amplio, estos valores van de 3 a 15 V para los 4000 y los 74C. De 2 a 6 V para los 74HC y 74HCT.

Los requerimientos de voltaje en la entrada para los dos estados lógicos se expresa como un porcentaje del voltaje de alimentación. Tenemos entonces: VOL (máx.) = 0 V, VOH (min) = VDD, VIL (máx.) = 30%VDDVIH(min) = 70% VDD.

Los CMOS pueden ser utilizados en medios con mucho más ruido. Los márgenes de ruido pueden hacerse todavía mejores si aumentamos el valor de VDD ya que es un porcentaje de este.

En lo que a la disipación de potencia concierne tenemos un consumo de potencia de sólo 2.5 nW cuando VDD = 5 V y cuando VDD = 10 V la potencia consumida aumenta a sólo 10 nW. Sin embargo tenemos que la disipación de potencia será baja mientras estemos trabajando con corriente directa. La potencia crece en proporción con la frecuencia. Una compuerta CMOS tiene la misma potencia de disipación en promedio con un 74LS en frecuencia alrededor de 2 a 3 MHz

Ya que los CMOS tienen una resistencia de entrada extremadamente grande (1012————————————) que casi no consume corriente. Pero debido a su capacitancia de entrada se limita el número de entradas CMOS que se pueden manejar con una sola salida CMOS. Así pues, el factor de carga de CMOS depende del máximo retardo permisible en la propagación. Comúnmente este factor de carga es de 50 para bajas frecuencias, para altas frecuencias el factor de carga disminuye.

Los valores de velocidad de conmutación dependen del voltaje de alimentación que se emplee, por ejemplo en una 4000 el tiempo de propagación es de 50 ns para VDD = 5 V y 25ns para VDD = 10 V. Como

podemos ver mientras VDD sea mayor podemos operar en frecuencias más elevadas.

Característica	Mínimo	Típico	Máximo	Unidad
V_{DD} (Voltaje de alimentación)	3.00	–	15	V
P_D (Potencia disipada promedio/comp.)	–	0.10	–	mW
T_{pd} (Tiempo de propagación)	–	25	–	ns
Fan out	–	50	–	compuertas
Margen de ruido	–	3.00	–	V

Tabla 2.2 Características de la familia CMOS

2.3 Diferencia entre CMOS y TTL

En esta tabla se observa cada una de las características principales de las familias lógicas TTL y CMOS, como es la velocidad de propagación, la disipación de potencia, y la excitación de salida que es diferente en cada integrado.

	Bipolar (TTL)			BiCMOS	CMOS					
					5 V			3,3 V		
	F	LS	ALS	ABT	HC	AC	AHC	LV	LVC	ALVC
Velocidad Retardo de propagación de puerta, t_p (ns)	3,3	10	7	3,2	7	5	3,7	9	4,3	3
Frecuencia máxima de reloj (MHz)	145	33	45	150	50	160	170	90	100	150
Disipación de potencia/puerta Bipolar: 50% dc (mW) CMOS: reposo (µW)	6	2,2	1,4	17	2,75	0,55	2,75	1,6	0,8	0,8
Excitación de salida I_{OL} (mA)	20	8	8	64	4	24	8	12	24	24

Tabla 2.3 TTL vs CMOS

2.4 Principales Características del PIC18F4520 – I/P

Este microcontrolador pertenece a la familia de microcontroladores de 8-bits del tipo flash. Su capacidad de memoria de programa es de 32kB, su memoria RAM 1,536 Bytes, soporta los protocolos de comunicación EUSART (USART mejorado Compatible con los estándares RS232 y RS485), SPI e I2C.

Tiene dos módulos PWM, oscilador interno de hasta 8MHz y Convertidor A/D de 8 canales. De los 40 pines que posee, 35 pueden ser usados como pines de entradas o salidas.

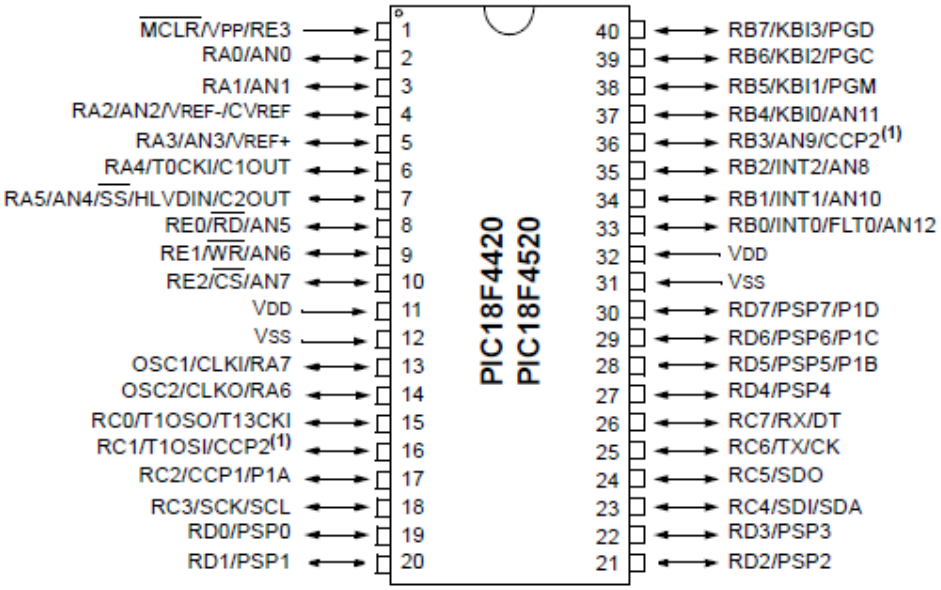


Fig. 2.1 Configuración de pines del PIC18F452

Parameter Name	Value
Program Memory Type	Flash
Program Memory (KB)	32
CPU Speed (MIPS)	10
RAM Bytes	1,536
Data EEPROM (bytes)	256
Digital Communication Peripherals	1-A/E/USART, 1-MSSP(SPI/I2C)
Capture/Compare/PWM Peripherals	2 CCP
Timers	1 x 8-bit, 3 x 16-bit
ADC	8 ch, 10-bit
Temperature Range (C)	-40 to 125
Operating Voltage Range (V)	2 to 5.5
Pin Count	40

Tabla 2.4 Parámetros técnicos del PIC18F452

2.5 Características de la Pantalla GLCD

Es una pantalla plana formada por una matriz de píxeles monocromos colocados delante de una fuente de luz o reflectora.

Dispone de una memoria RAM interna del mismo tamaño de la capacidad que dispone la pantalla.

Su iluminación de fondo está entre verde-amarillo cuando se enciende. Fácil manejo con microprocesadores de 8-Bits. Bajo poder de consumo y contiene dos controladores internos un KS0108B y KS0107B.

PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL
1	/CS1	10	DB1	19	LED_R+
2	/CS2	11	DB2	20	LED_G+
3	VSS	12	DB3	21	LED_B+
4	VDD	13	DB4	22	Y-
5	V0	14	DB5	23	X-
6	D/I	15	DB6	24	Y+
7	R/W	16	DB7	25	X+
8	E	17	/RST		
9	DB0	18	VEE		

Tabla 2.5 Configuración de pines de la pantalla GLCD

2.6 Manejo de Teclado Matricial 4X4

Un teclado matricial es un simple arreglo de botones conectados en filas y columnas, de modo que se pueden leer varios botones con el mínimo número de pines requeridos. Un teclado matricial 4×4 solamente ocupa 4

líneas de un puerto para las filas y otras 4 líneas para las colúmnas, de este modo se pueden leer 16 teclas utilizando solamente 8 líneas de un microcontrolador. Si asumimos que todas las columnas y filas inicialmente están en alto (1 lógico), la pulsación de un botón se puede detectar al poner cada fila a en bajo (0 lógico) y checar cada columna en busca de un cero, si ninguna columna está en bajo entonces el 0 de las filas se recorre hacia la siguiente y así secuencialmente.

2.7 Conceptos Básicos de Programación

En esta sección se realizará una breve descripción de los conceptos de Programación usada en el presente trabajo que se aplicaron para una mejor organización y compresión en el desarrollo del proyecto.

2.7.1 Variables Globales

Una variable global es en informática, una variable accesible en todos los ámbitos de un programa. Los mecanismos de interacción con variables globales se denominan mecanismos de entorno global. El concepto de entorno global contrasta con el de entorno local donde todas las variables son locales sin memoria compartida.

2.7.2 Variables Locales

Es la variable a la que se le otorga un ámbito local. Tales variables sólo pueden accederse desde la función o bloque de instrucciones en donde se declaran. Las variables locales se contraponen a las variables globales. En la mayoría de lenguajes de Programación las variables locales son variables automáticas almacenadas directamente en la pila de llamadas. De esta forma las variables con este ámbito se pueden declarar, reescribir y leer sin riesgo de efectos secundarios para los procesos fuera del bloque en el que son declarados.

2.7.3 Funciones

Llamadas también subprogramas o subrutinas, se presenta como un sub-algoritmo que forma parte del algoritmo principal, el cual permite resolver una tarea específica y devuelve un valor.

2.7.4 Procedimientos

Los procedimientos se asemejan mucho a las funciones, con la única diferencia que no devuelve un valor. Son utilizados para el procesamiento de información sobre las variables de ámbito global.

2.7.5 MikroProC for PIC.

Es un compilador avanzado para microcontroladores PIC. Su plataforma de programación es en C. La versión Pro incluye un conjunto de librerías y ejemplos destinados a facilitar el desarrollo de aplicaciones.

2.7.6 Proteus 7.6

Es un paquete de software para el diseño de circuitos electrónicos que incluye captura de los esquemas, simulación analógica y digital combinada, además posee una herramienta ARES que se utiliza para el diseño de circuitos impresos. Proteus es un entorno integrado diseñado para la realización completa de proyectos de construcción de equipos electrónicos en todas sus etapas: diseño, simulación, depuración y construcción. El paquete está compuesto por dos programas: ISIS, para la captura y simulación de circuitos; y ARES, para el diseño de PCB's. También permite simular y depurar el funcionamiento de todo el sistema ejecutando el software paso a paso, insertando puntos de ruptura (breakpoints, que también pueden ser generados por el hardware), mirando el contenido de registros y posiciones de memoria, etc. y comprobando si la respuesta del hardware es la correcta. También se simulan herramientas electrónicas, como osciloscopios, analizadores lógicos, voltímetros etc.

CAPÍTULO 3

3. Diseño del proyecto

En el presente capítulo se pone de manifiesto todas las etapas de Diseño, implementación y pruebas necesarias para la elaboración de este proyecto.

3.1 Diseño general

En el bloque del Teclado Matricial es utilizado para ingresar la serie del circuito integrado.

En el bloque de la Pantalla GLCD es utilizado para visualizar loas datos ingresados y los resultados presentados.

En el bloque del Microcontrolador 1 es utilizado para asignarle una código a la serie del C.I. para que el microcontrolador 2 lo interprete, además se encarga del control del teclado matricial y de la pantalla GLCD.

En el bloque del Microcontrolador 2 se encarga de la comprobación del C.I.

El bloque del CHIP es un zócalo donde se coloca el C.I. este abarca hasta 24 pines

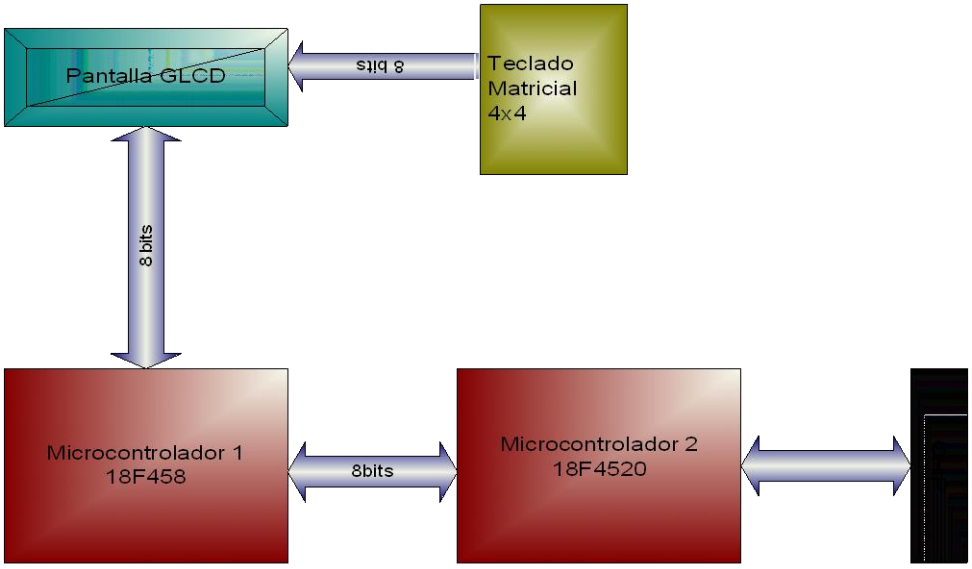


Fig. 3.1 Diagrama de Bloque del proyecto

3. 2 Diseño de componentes de software

3.2.1 Código de programación del Microcontrolador 1

En este tópico se detallará la programación de la pantalla GLCD, el teclado Matricial y microcontrolador 1

3.2.1.1 Declaración de variables

```
//Declaracion de variables para la GLCD //
 char GLCD_DataPort at PORTD;
 sbit GLCD_CS1 at RC0_bit;
```

```
sbit GLCD_CS2 at RC1_bit;
sbit GLCD_RS  at RC2_bit;
sbit GLCD_RW  at RC3_bit;
sbit GLCD_EN  at RC4_bit;
sbit GLCD_RST at RC5_bit;
sbit GLCD_CS1_Direction at TRISC0_bit;
sbit GLCD_CS2_Direction at TRISC1_bit;
sbit GLCD_RS_Direction  at TRISC2_bit;
sbit GLCD_RW_Direction  at TRISC3_bit;
sbit GLCD_EN_Direction  at TRISC4_bit;
sbit GLCD_RST_Direction at TRISC5_bit;
```

3.2.1.2 Funciones y Procedimientos

```
/*Ejecuta retardo*/
void delay2S()
// ------------------------------------------------------
// GLCD Picture name: Espol.bmp
// GLCD Model: KS0108 128x64
// ------------------------------------------------------
unsigned char const Espol_bmp[1024]
/*Limpia pantalla*/
unsigned char const Clean_bmp[1024]
/*Figura del menu*/
unsigned char const Menu_bmp[1024]
/*Figura de buen estado*/
unsigned char const Buen_estado_bmp[1024]
/*Figura de mal estado*/
unsigned char const Mal_estado_bmp[1024]
```

```
/*Figura del menu de la familia CMOS*/
unsigned char const CMOS_bmp[1024]
/*Inicializa la pantalla GLCD*/
 void inicio()
/*Escribe en la pantalla el estado del integrado*/
 void comprobacion ()
/*Envia el codigo correspondiente de la familia CMOS 40XXX
 desde el  microcontrolador 1 al microcontalador 2*/
 void codigoCMOS40(unsigned short yy,unsigned short zy,unsigned short nn)

/*Envia el codigo correspondiente de la familia CMOS 74XXX
 desde el  microcontrolador 1 al microcontalador 2*/
 void codigoCMOS74(unsigned short yy,unsigned short zy,unsigned short nn)
/*Envia el codigo correspondiente de la familia TTL 74LSXXX
 desde el  microcontrolador 1 al microcontalador 2*/
void codigoTTL(unsigned short yy,unsigned short zy,unsigned short nn)
/*Evita rebotes al presioar el teclado*/
void ANTIRREBOTE()

/*Revisa que tecla a sido presionada y retorna una posición  de la tecla*/
unsigned short TECLADO()
/*Revisa que columna fue presionada y retorna esa columna*/
unsigned short CHECA_COL()
/*Retorna el valor en ascii de la tecla presionada*/
unsigned short numero(unsigned short kp)
```

3.2.1.3 Programa Principal (Main)

```
void main()
{
        unsigned short  cnt;
        while(1)
        {
         // Declaracion de variables
         int ll,ww,senal1,senal2,senal3 ;
         unsigned short kp;
         unsigned short ii,yy,zz,nn,co,zy;

        ADCON1 = 0b00001100;
        ADCON2 = 0b00111110;
        OSCCON = 126;                   //CONFIGURACION DEL CLOCK EN 8MHZ
        INTCON=0b00000000;
        INTCON2=0b00000000;
        TRISA=0b11000000;
        PORTA=0b00000000;
        TRISE=0b10000000;
        PORTE=0b11111000;
        TRISC=0b11000000;
        TRISB=0b11110000;
        PORTB=255;
        kp=0;


        Glcd_Init();        // Initialize GLCD
```

```
        Glcd_Fill(0x00);
        Glcd_Image(Espol_bmp);
        Delay_ms(2000);

         // Inicializacion de varaiables
EMPEZAR:
         ADCON1 = 0b00001100;
         ADCON2 = 0b00111110;
         TRISA=0b11000000;
         PORTA=0b00000000;
         TRISE=0b10000000;
         PORTE=0b11111000;
         TRISC=0b11000000;
         PORTB=0XFF;
         RC6_bit=0;
         RC7_bit=0;
         ii = 0;
         yy=0;
         zz=0;
         ww=0;
         nn=0;
         zy=0;
         kp=0;
        co=0;
        ll=0;
        senal1=0;
        senal2=0;
```

```
senal3=0;
Glcd_Image(Clean_bmp);
Glcd_Init();     // Initialize GLCD
Glcd_Fill(0x00);
 Delay_ms(500);
#ifdef COMPLETE_EXAMPLE
  Glcd_Set_Font(Character8x7, 8, 14, 63);// Choose font, see __Lib_GLCDFonts.c in Uses
folder
#endif

 while(ll==0)
 {
  Glcd_Image(Menu_bmp);
     do {

          kp = TECLADO();

        } while (!kp);

   cnt=numero(kp);
   if (cnt==49)
      {
         Glcd_Image(Clean_bmp);
         Glcd_Fill(0x00);
         Glcd_Write_Text("       ",0, 0, 1);
         Glcd_Write_Text("74LS",27, 3, 1);
         ll=1;
```

```
   senal1=1;
}
if (cnt==50)
{
   Glcd_Image(Clean_bmp);
   kp=0;
   Glcd_Image(CMOS_bmp);
   do {
    kp = TECLADO();
   } while (!kp);
   cnt=numero(kp);
   Delay_ms(50);
   if  (cnt==49)
    {
        Glcd_Image(Clean_bmp);
        Glcd_Fill(0x00);
        Glcd_Write_Text("          ", 63, 0, 1);
        Glcd_Write_Text("          ", 63, 2, 1);
        Glcd_Write_Text(" 74", 27, 3, 1);
        senal2=1;
    }
    if  (cnt==50)
    {
        Glcd_Image(Clean_bmp);
        Glcd_Fill(0x00);
        Glcd_Write_Text("          ", 63, 0, 1);
        Glcd_Write_Text("          ", 63, 2, 1);
```

```
            Glcd_Write_Text("   4", 27, 3 , 1);
            senal3=1;
          }
          ll=1;
       }
  }
do{
  kp=0;
  do {
     kp = TECLADO();
   } while (!kp);
    cnt=numero(kp);
if(ii==0)
 {
 yy=cnt;
 Glcd_Write_Char(yy,55, 3, 1);
 }
 if(ii==1)
 {
 Delay_ms(50);
 zz=cnt;
 zy=cnt;
 Glcd_Write_Char(zz,62 , 3, 1);
 }
  if(ii==2)
  {
  Delay_ms(50);
```

```
        nn=cnt;
        Glcd_Write_Char(nn, 69, 3, 1);
        Glcd_Write_Text("PRESIONE  Y/N: ",13, 5, 1);   // Write string
      }
      if(ii==3)
      {
       Delay_ms(50);

        Glcd_Write_Char(cnt, 105, 5, 1);
        co=cnt;
      }
   ii++;
  } while (ii<4);
if(co==89)
{
          if(senal1==1)
          {
             codigoTTL(yy,zy,nn);
              inicio() ;
          }
          if(senal2==1)
          {
            codigoCMOS74(yy,zy,nn);
             inicio() ;
          }
           if(senal3==1)
          {
```

```
                codigoCMOS40(yy,zy,nn);
                 inicio();
               }
             }
             else
               {
                  inicio();


               }


          goto EMPEZAR;
       }
```

3.2.2 Código de programación del Microcontrolador 2

En este tópico se detallará la programación utilizada para la verificación del estado del integrado y la programación del microcontrolador 2.

3.2.2.1 Declaración de Variables

```
// Inicializacion de variables

int tipo_chip;  // para ver el integrado que el usuario elija
int res_chip;   // para enviar el resultado al micro2
int cont;       //ayuda en algunos procedimientos para el conteo
```

3.2.2.2 Funciones y Procedimientos

```
// Procedimiento para retardo
void retardo(){
    delay_ms(2000);
}
```

```
////////////////////////////////////////////////////////////
// PROCEDIMIENTOS DE CADA UNO DE LOS INTEGRADOS A COMPROBAR //
////////////////////////////////////////////////////////////

//Cada uno de los integrados a comprobar tendrán su respectivo procedimiento el
//cual se lo realiza dependiendo de las especificaciones de cada integrado
//comparando la tabla de verdad

 //******* //
//  TTL   //
//*******//
void nand(){
    TRISC=9;
    PORTC=191;
    TRISD=72;
    PORTD=255;

    if ( RD3_bit==0 && RC3_bit==0 && RD6_bit==0 && RC0_bit==0 ){
       res_chip=1;
    }
     else{
        res_chip=2;
     }
}
void inv(){
    TRISC=42;
    PORTC=63;
    TRISD=42;
    PORTD=255;
    if ( RC1_bit==0 && RC3_bit==0 && RC5_bit==0 && RD1_bit==0 && RD3_bit==0 &&
        RD5_bit==0 ){
        res_chip=1;
    }
    else{
    res_chip=2;
    }
}
void and11(){
    TRISC=32;
    PORTC=63;
    TRISD=34;
    PORTD=255;
    cont=0;
    if ( RD1_bit==1 && RC5_bit==1 && RD5_bit==1 ){
        cont++;
        PORTC=0;
        PORTD=128;
        if ( RD1_bit==0 && RC5_bit==0 && RD5_bit==0 ){
          cont++;
        }
    }
    if (cont==2){
```

```
        res_chip=1;
    }
    else{
    res_chip=2;
    }
    cont=0;
}
void nand13(){
    TRISC=32;
    PORTC=59;
    TRISD=2;
    PORTD=238;
    if ( RD1_bit==0 && RC5_bit==0 ){
        cont++;
        PORTC=0;
        PORTD=128;
        if ( RD1_bit==1 && RC5_bit==1 ){
          cont++;
        }
    }
     if (cont==2){
       res_chip=1;
    }
    else{
    res_chip=2;
    }
    cont=0;
}
void ninv(){
    TRISC=42;
    PORTC=63;
    TRISD=42;
    PORTD=254;
    if ( RC1_bit==1 && RC3_bit==1 && RC5_bit==1 && RD1_bit==1 && RD3_bit==1 &&
        RD5_bit==1 ){
        cont++;
        PORTC=0;
        PORTD=128;
        if ( RC1_bit==0 && RC3_bit==0 && RC5_bit==0 && RD1_bit==0 &&
            RD3_bit==0 && RD5_bit==0 ){
            cont++;
        }
    }
    if (cont==2){
       res_chip=1;
    }
    else{
    res_chip=2;
    }
    cont=0;
}
void or32(){
    TRISC=36;
    PORTC=0;
```

```
    TRISD=18;
    PORTD=128;
    if ( RD4_bit==0 && RC2_bit==0 && RD1_bit==0 && RC5_bit==0 ){
        res_chip=1;
        retardo();
    }
    else{
        res_chip=2;
        retardo();
    }
 }

void decoder42(){
    TRISC=127;
    RC7_bit=0;
    TRISD=7;
    PORTD=128;
    RD7_bit=1;
    //retardo();
    //PARA EL CERO
    if ( PORTC==126 && PORTD==135 ){
        cont++;
        RD6_bit=1;
        retardo();
    }
    else{
        res_chip=2;
        goto FIN;
    }
    //PARA EL 1

    if ( PORTC==125 && PORTD==199 ){
        cont++;
         RD5_bit=1;
         RD4_bit=1;
         retardo();
    }
    else{
        res_chip=2;
        goto FIN;
    }
    //PARA EL 7

    if ( PORTC==127 && PORTD==246 ){
        cont++;
        RD3_bit=1 ;
        retardo();
    }
    else{
        res_chip=2;
        goto FIN;
    }

    // default
```

```
    if ( PORTC==127 && PORTD==255 ){
        cont++;
    }
    else{
        res_chip=2;
        goto FIN;
    }
    if (cont==4)
    {
        res_chip=1;
    }
    else
    {
        res_chip=2;
    }

    FIN: cont=0;

}

 void elemento51(){
    TRISC=32;
    PORTC=0;
    TRISD=2;
    RD7_bit=1;
    retardo();
    cont=0;
    if (RD1_bit==1 && RC5_bit==1){
       cont++;
       PORTC=63;
       PORTD=254;
       if (RD1_bit==0 && RC5_bit==0){
          cont++;
       }

    }
    if (cont==2)
    {
       res_chip=1;
    }
    else{
       res_chip=2;
    }
   cont=0;
}

 void flipflop73(){
    TRISC=0;
    PORTC=106;
    TRISD=102;
    PORTD=128;
    delay_ms(100);
    RC0_bit=1;
```

```
    RC4_bit=1;
    delay_ms(100);
    RC0_bit=0;
    RC4_bit=0;
    delay_ms(100);
    if (RD1_bit==0 && RD2_bit==1 && RD5_bit==1 && RD6_bit==0){
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    retardo();
    PORTC=46;
    PORTD=8;
    delay_ms(100);
    RC0_bit=1;
    RC4_bit=1;
    delay_ms(100);
    RC0_bit=0;
    RC4_bit=0;
    delay_ms(100);
    if (RD1_bit==1 && RD2_bit==0 && RD5_bit==0 && RD6_bit==1){
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
     if (cont==2){
      res_chip=1;
    }
    else{
      res_chip=2;
    }
    FIN:cont=0;


}


void lanch75(){
    TRISC=129;
    PORTC=126;
    TRISD=231;
    PORTD=16;
    retardo();
    if (RD7_bit==1 && RD6_bit==1 && RD5_bit==0 && RD2_bit==0 && RD1_bit==1
        && RD0_bit==1 && RC7_bit==0 && RC0_bit==0){
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
```

```
    }
    PORTC=24;
     if (RD7_bit==0 && RD6_bit==0 && RD5_bit==1 && RD2_bit==1 && RD1_bit==0
        && RD0_bit==0 && RC7_bit==1 && RC0_bit==1){
       cont++;
    }
    else{
       res_chip=2;
       goto FIN;
    }
    if (cont==2){
       res_chip=1;
    }
    else{
       res_chip=2;
    }
    FIN:cont=0;

 }

void suma83(){
  //0+15=15
    TRISC=34;
    PORTC=88;
    TRISD=97;
    PORTD=132;
    delay_ms(100);
    if (RC1_bit==1 && RC5_bit==1 && RD0_bit==1 && RD5_bit==0 && RD6_bit==1)
    {
       cont++;
    }
    else{
       res_chip=2;
       goto FIN;
    }
    //9+8=17
    retardo();
    PORTC=17;
    PORTD=162;
    delay_ms(100);
    if (RC1_bit==0 && RC5_bit==0 && RD0_bit==1 && RD5_bit==1 && RD6_bit==0){
       cont++;
    }
    else{
       res_chip=2;
       goto FIN;
    }
    if (cont==2){
       res_chip=1;
    }
    else{
       res_chip=2;
    }
    FIN:cont=0;
```

```
}
 void shift91(){
    TRISC=0;
    PORTC=16;
    TRISD=192;
    PORTD=48;
    delay_ms(100);
    RD2_bit=1;
    delay_ms(100);
    RD2_bit=0;
    delay_ms(100);
    if (RD6_bit==1 && RD7_bit==0)
    {
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    retardo();
    RD2_bit=1;
    delay_ms(100);
    RD2_bit=0;
    delay_ms(100);
   if (RD6_bit==0 && RD7_bit==1)
    {
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    if (cont==2){
      res_chip=1;
    }
    else{
      res_chip=2;
    }
    FIN:cont=0;
}
 void shift95()
{
    int ww,kp,ii,yy,suma,ant,yz,suma1,ant1,suma2,jj;

    TRISC=0b00000000;
    TRISD=0b01111000;
    TRISE=0b10000000;

    PORTC=0b00011111;
    PORTD=0b10000100;
    PORTE=0b10000000;
    ww=0;
    kp=1;
    ii=0;
```

```
    jj=0;
    suma=0;
     ant=0;
     suma1=0;
     ant1=0;
    while(ww==0)
    {
     if(kp==1)
     {
       Delay_ms(500);

       PORTC=0b00011111;
       RD2_bit=1;
       RD7_bit=1;

       kp=0;
       Delay_ms(500);
       PORTC=0b00011111;
       RD2_bit=0;
       RD7_bit=1;

       yy= RD3_bit+RD4_bit+RD5_bit+RD6_bit ;

        suma=yy+ant;
        ant=suma;

       ii++;

     }
     kp=1;

     if(ii==4)
     {
       ww=1;
      }

  }


  if(suma==10)
  {
     res_chip=1;
  }
 else
  {
    res_chip=2;
  }
}

 void buffer125(){
    TRISC=36;
    PORTC=0;
    TRISD=18;
    PORTD=128;
```

```
    delay_ms(100);
    if (RC2_bit==0 && RD1_bit==0 && RC5_bit==0 && RD4_bit==0){
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    retardo();
    PORTC=18;
    PORTD=164;
    delay_ms(100);
    if (RC2_bit==1 && RD1_bit==1 && RC5_bit==1 && RD4_bit==1){
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    if (cont==2){
      res_chip=1;
    }
    else{
      res_chip=2;
    }
    FIN:cont=0;
 }
void decoder137(){
    TRISC=64;
    PORTC=32;
    TRISD=127;
    PORTD=128;
    delay_ms(100);
    if (RC6_bit==1 && RD0_bit==1 && RD1_bit==1 && RD2_bit==1 && RD3_bit==1
       && RD4_bit==1 && RD5_bit==1 && RD6_bit==0) {
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    retardo();
    PORTC=39;
    PORTD=128;
    delay_ms(200);
    if (RC6_bit==0 && RD0_bit==1 && RD1_bit==1 && RD2_bit==1 && RD3_bit==1
       && RD4_bit==1 && RD5_bit==1 && RD6_bit==1) {
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    retardo();
```

```
    PORTC=16;
    PORTD=128;
    delay_ms(200);
    if (RC6_bit==1 && RD0_bit==1 && RD1_bit==1 && RD2_bit==1 && RD3_bit==1
        && RD4_bit==1 && RD5_bit==1 && RD6_bit==1) {
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    if (cont==3){
      res_chip=1;
    }
    else{
      res_chip=2;
    }
    FIN:cont=0;
 }
void multiplex151(){
    TRISC=48;
    PORTC=8;
    TRISD=0;
    PORTD=128;
    delay_ms(100);
    //0
    if (RC4_bit==1 && RC5_bit==0) {
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    //3
    retardo();
    PORTC=1;
    PORTD=134;
    delay_ms(200);
     if (RC4_bit==1 && RC5_bit==0) {
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    //7
    retardo();
    PORTC=0;
    PORTD=143;
    delay_ms(200);
    if (RC4_bit==1 && RC5_bit==0) {
      cont++;
    }
    else{
```

```
        res_chip=2;
        goto FIN;
      }
      //default
      retardo();
      PORTC=64;
      PORTD=128;
      delay_ms(200);
      if (RC4_bit==0 && RC5_bit==1) {
        cont++;
      }
      else{
        res_chip=2;
        goto FIN;
      }
      if (cont==4){
        res_chip=1;
      }
      else{
        res_chip=2;
      }
      FIN:cont=0;
 }

void shift164(){
      TRISC=60;
      PORTC=3;
      TRISD=120;
      PORTD=132;
      delay_ms(800);
      RD1_bit=1;
      delay_ms(800);
      if (RC2_bit==1) {
        RD1_bit=0;
        RD1_bit=1;
        delay_ms(800);
        cont++;
        if (RC3_bit==1){
            RD1_bit=0;
            RD1_bit=1;
            delay_ms(800);
            cont++;
            if (RC4_bit==1){
              RD1_bit=0;
              RD1_bit=1;
              delay_ms(800);
              cont++;
              if (RC5_bit==1){
                RD1_bit=0;
                RD1_bit=1;
                delay_ms(800);
                cont++;
                if (RD3_bit==1){
                  RD1_bit=0;
```

```
            RD1_bit=1;
            delay_ms(800);
            cont++;
            if (RC4_bit==1){
              RD1_bit=0;
              RD1_bit=1;
              delay_ms(800);
              cont++;
               if (RD5_bit==1){
                 RD1_bit=0;
                 RD1_bit=1;
                 delay_ms(800);
                 cont++;
                 if (RD6_bit==1){
                   RD1_bit=0;
                   RD1_bit=1;
                   delay_ms(800);
                   cont++;
                 }
               }
             }

          }
        }
    }
  }
}
PORTC=0;
PORTD=133;
delay_ms(800);
RD1_bit=1;
delay_ms(800);
if (RC2_bit==0) {
  RD1_bit=0;
  RD1_bit=1;
  delay_ms(800);
  cont++;
  if (RC3_bit==0){
     RD1_bit=0;
     RD1_bit=1;
     delay_ms(800);
     cont++;
     if (RC4_bit==0){
       RD1_bit=0;
       RD1_bit=1;
       delay_ms(800);
       cont++;
       if (RC5_bit==0){
         RD1_bit=0;
         RD1_bit=1;
         delay_ms(800);
         cont++;
         if (RD3_bit==0){
           RD1_bit=0;
```

```
                RD1_bit=1;
                delay_ms(800);
                cont++;
                if (RD4_bit==0){
                  RD1_bit=0;
                  RD1_bit=1;
                  delay_ms(800);
                  cont++;
                  if (RD5_bit==0){
                    RD1_bit=0;
                    RD1_bit=1;
                    delay_ms(800);
                    cont++;
                    if (RD6_bit==0){
                      RD1_bit=0;
                      RD1_bit=1;
                      delay_ms(800);
                      cont++;
                    }
                  }

                }
              }
            }
          }
        }
      }

      if (cont==16){
        res_chip=1;
      }
      else{
        res_chip=2;
      }
      FIN:cont=0;
}

 void contador169(){
    //Load
    TRISC=0;
    PORTC=12;
    TRISD=124;
    PORTD=128;
    delay_ms(100);
    RC1_bit=1;
    delay_ms(100);
    if (RD2_bit==0 && RD3_bit==0 && RD4_bit==1 && RD5_bit==1 && RD6_bit==1) {
      cont++;
    }
    else{
      res_chip=2;
      goto FIN;
    }
    //HOLD
```

```
retardo();
PORTC=84;
PORTD=131;
delay_ms(100);
RC1_bit=1;
delay_ms(100);
if (RD2_bit==0 && RD3_bit==0 && RD4_bit==1 && RD5_bit==1 && RD6_bit==1) {
  cont++;
}
else{
  res_chip=2;
  goto FIN;
}
//CONTADOR DOWN
retardo();
PORTC=0;
PORTD=129;
delay_ms(100);
RC1_bit=1;
delay_ms(100);
if (RD2_bit==0 && RD3_bit==0 && RD4_bit==1 && RD5_bit==0 && RD6_bit==1){
   RC1_bit=0;
   RC1_bit=1;
   delay_ms(100);
   cont++;
   if (RD2_bit==0 && RD3_bit==0 && RD4_bit==0 && RD5_bit==1 &&
       RD6_bit==1){
       RC1_bit=0;
       RC1_bit=1;
       delay_ms(100);
       cont++;
       if (RD2_bit==0 && RD3_bit==0 && RD4_bit==0 && RD5_bit==0 &&
           RD6_bit==0){
           cont++;
       }
   }
}
//Load
retardo();
PORTC=12;
PORTD=128;
delay_ms(100);
RC1_bit=1;
delay_ms(100);
if (RD2_bit==0 && RD3_bit==0 && RD4_bit==1 && RD5_bit==1 && RD6_bit==1) {
  cont++;
}
else{
  res_chip=2;
  goto FIN;
}
//CONTADOR UP
retardo();
PORTC=1;
```

```
PORTD=129;
delay_ms(100);
RC1_bit=1;
delay_ms(100);
if (RD2_bit==0 && RD3_bit==1 && RD4_bit==0 && RD5_bit==0 && RD6_bit==1){
   RC1_bit=0;
   RC1_bit=1;
   delay_ms(100);
   cont++;
   if (RD2_bit==0 && RD3_bit==1 && RD4_bit==0 && RD5_bit==1
      && RD6_bit==1){
      RC1_bit=0;
      RC1_bit=1;
      delay_ms(100);
      cont++;
      if (RD2_bit==0 && RD3_bit==1 && RD4_bit==1 && RD5_bit==0 &&
         RD6_bit==1){
         RC1_bit=0;
         RC1_bit=1;
         delay_ms(100);
         cont++;
         if (RD2_bit==0 && RD3_bit==1 && RD4_bit==1 && RD5_bit==1 &&
            RD6_bit==1){
            RC1_bit=0;
            RC1_bit=1;
            delay_ms(100);
            cont++;
            if (RD2_bit==1 && RD3_bit==0 && RD4_bit==0 && RD5_bit==0
               && RD6_bit==1){
               RC1_bit=0;
               RC1_bit=1;
               delay_ms(100);
               cont++;
               if (RD2_bit==1 && RD3_bit==0 && RD4_bit==0 &&
                  RD5_bit==1 && RD6_bit==1){
                  RC1_bit=0;
                  RC1_bit=1;
                  delay_ms(100);
                  cont++;
                  if (RD2_bit==1 && RD3_bit==0 && RD4_bit==1 &&
                     RD5_bit==0 && RD6_bit==1){
                     RC1_bit=0;
                     RC1_bit=1;
                     delay_ms(100);
                     cont++;
                     if (RD2_bit==1 && RD3_bit==0 && RD4_bit==1 &&
                        RD5_bit==1 && RD6_bit==1){
                        RC1_bit=0;
                        RC1_bit=1;
                        delay_ms(100);
                        cont++;
                        if (RD2_bit==1 && RD3_bit==1 &&
                           RD4_bit==0 && RD5_bit==0 &&
                           RD6_bit==1){
```

```
                                RC1_bit=0;
                                RC1_bit=1;
                                delay_ms(100);
                                cont++;
                                if (RD2_bit==1 && RD3_bit==1 &&
                                   RD4_bit==0 && RD5_bit==1 &&
                                   RD6_bit==1){
                                   RC1_bit=0;
                                   RC1_bit=1;
                                   delay_ms(100);
                                   cont++;
                                   if (RD2_bit==1 && RD3_bit==1 &&
                                      RD4_bit==1 && RD5_bit==0 &&
                                      RD6_bit==1){
                                      RC1_bit=0;
                                      RC1_bit=1;
                                      delay_ms(100);
                                      cont++;
                                      if (RD2_bit==1 &&
                                         RD3_bit==1 &&
                                         RD4_bit==1 &&
                                         RD5_bit==1 &&
                                         RD6_bit==0){
                                         cont++;
                                      }
                                   }
                                }
                             }
                          }
                       }
                    }
                 }
              }
           }
        }
     }
  }

if (cont==18){
   res_chip=1;
}
else{
   res_chip=2;
}
FIN:cont=0;
}

 //////////
 // CMOS //
 /////////
  void Cnor00(){
     TRISC=32;
     PORTC=0;
     TRISD=12;
     PORTD=144;
```

```
    if ( RD2_bit==1 && RD3_bit==0 && RC5_bit==1 ){
        res_chip=1;
    }
    else{
        res_chip=2;
    }
 }
void Cdecada17(){
    TRISC=127;
    PORTC=0;
    TRISD=15;
    PORTD=128;
    delay_ms(100);
    //RD5_bit=1;
    delay_ms(100);
    if (RC0_bit==0 && RC1_bit==0 && RC2_bit==1 && RC3_bit==0 && RC4_bit==0 &&
       RC5_bit==0 && RC6_bit==0 && RD0_bit==0 && RD1_bit==0 && RD2_bit==0 &&
       RD3_bit==1) {
       RD5_bit=0;
       RD5_bit=1;
       delay_ms(100);
       cont++;
       if (RC0_bit==0 && RC1_bit==1 && RC2_bit==0 && RC3_bit==0 && RC4_bit==0
          && RC5_bit==0 && RC6_bit==0 && RD0_bit==0 && RD1_bit==0 &&
          RD2_bit==0 && RD3_bit==1) {
          RD5_bit=0;
          RD5_bit=1;
          delay_ms(100);
          cont++;
          if (RC0_bit==0 && RC1_bit==0 && RC2_bit==0 && RC3_bit==1 &&
             RC4_bit==0 && RC5_bit==0 && RC6_bit==0 && RD0_bit==0 &&
             RD1_bit==0 && RD2_bit==0 && RD3_bit==1) {
             RD5_bit=0;
             RD5_bit=1;
             delay_ms(100);
             cont++;
             if (RC0_bit==0 && RC1_bit==0 && RC2_bit==0 && RC3_bit==0 &&
                RC4_bit==0 && RC5_bit==0 && RC6_bit==1 && RD0_bit==0 &&
                RD1_bit==0 && RD2_bit==0 && RD3_bit==1) {
                RD5_bit=0;
                RD5_bit=1;
                delay_ms(100);
                cont++;
                if (RC0_bit==0 && RC1_bit==0 && RC2_bit==0 && RC3_bit==0 &&
                   RC4_bit==0 && RC5_bit==0 && RC6_bit==0 && RD0_bit==0 &&
                   RD1_bit==1 && RD2_bit==0 && RD3_bit==1) {
                   RD5_bit=0;
                   RD5_bit=1;
                   delay_ms(100);
                   cont++;
                   if (RC0_bit==1 && RC1_bit==0 && RC2_bit==0 &&
                      RC3_bit==0 && RC4_bit==0 && RC5_bit==0 &&
                      RC6_bit==0 && RD0_bit==0 && RD1_bit==0 &&
                      RD2_bit==0 && RD3_bit==0) {
```

```
                RD5_bit=0;
                RD5_bit=1;
                delay_ms(100);
                cont++;
                if (RC0_bit==0 && RC1_bit==0 && RC2_bit==0 &&
                   RC3_bit==0 && RC4_bit==1 && RC5_bit==0 &&
                   RC6_bit==0 && RD0_bit==0 && RD1_bit==0 &&
                   RD2_bit==0 && RD3_bit==0) {
                   RD5_bit=0;
                   RD5_bit=1;
                   delay_ms(100);
                   cont++;
                   if (RC0_bit==0 && RC1_bit==0 && RC2_bit==0 &&
                      RC3_bit==0 && RC4_bit==0 && RC5_bit==1 &&
                      RC6_bit==0 && RD0_bit==0 && RD1_bit==0 &&
                      RD2_bit==0 && RD3_bit==0) {
                      RD5_bit=0;
                      RD5_bit=1;
                      delay_ms(100);
                      cont++;
                      if (RC0_bit==0 && RC1_bit==0 && RC2_bit==0
                         && RC3_bit==0 && RC4_bit==0 &&
                         RC5_bit==0 && RC6_bit==0 &&
                         RD0_bit==1 && RD1_bit==0 &&
                         RD2_bit==0 && RD3_bit==0) {
                         RD5_bit=0;
                         RD5_bit=1;
                         delay_ms(100);
                         cont++;
                         if (RC0_bit==0 && RC1_bit==0 &&
                            RC2_bit==0 && RC3_bit==0 &&
                            RC4_bit==0 && RC5_bit==0 &&
                            RC6_bit==0 && RD0_bit==0 &&
                            RD1_bit==0 && RD2_bit==1 &&
                            RD3_bit==0) {
                            RD5_bit=0;
                            RD5_bit=1;
                            delay_ms(100);
                            cont++;
                            if (RC0_bit==0 && RC1_bit==0 &&
                               RC2_bit==1 && RC3_bit==0 &&
                               RC4_bit==0 && RC5_bit==0 &&
                               RC6_bit==0 && RD0_bit==0 &&
                               RD1_bit==0 && RD2_bit==0 &&
                               RD3_bit==1) {
                               cont++;

                            }
                         }
                      }
                   }
                }
             }
          }
       }
    }
```

```
            }
          }
        }
  }

if (cont==11){
   res_chip=1;
}
else{
   res_chip=2;
}
FIN:cont=0;
}
void Ccontador20(){
    TRISC=127;
    PORTC=0;
    TRISD=121;
    PORTD=130;
    while( cont < 16381 ){
      delay_us(10);
      RD1_bit=0;
      delay_us(10);
      RD1_bit=1;
      cont++;
    }
    retardo();
    if (RC0_bit==1 && RC1_bit==1 && RC2_bit==1 && RC3_bit==1 && RC4_bit==1 &&
       RC5_bit==1 && RC6_bit==1 && RD0_bit==1 && RD3_bit==1 && RD4_bit==1 &&
       RD5_bit==1 && RD6_bit==1){
       res_chip=1;
    }
    else{
       res_chip=2;
    }
    cont=0;

}
void Cnand23(){
    TRISC=32;
    PORTC=31;
    TRISD=12;
    PORTD=242;
    if ( RD2_bit==0 && RC5_bit==0 && RD3_bit==0 ){
        res_chip=1;
    }
    else{
    res_chip=2;
    }
}
void Ccontador24(){
    TRISC=60;
    PORTC=0;
    TRISD=52;
    PORTD=128;
```

```
    while( cont < 128 ){
      delay_us(1000);
      RC0_bit=0;
      delay_us(1000);
      RC0_bit=1;
      cont++;
    }
   if ( RC2_bit==1 && RC3_bit==1 && RC4_bit==1 && RC5_bit==1  && RD2_bit==1
       && RD4_bit==1 && RD5_bit==1 ){
       res_chip=1;
    }
    else{
       res_chip=2;
    }
    cont=0;

}
void Cor72(){
    TRISC=1;
    PORTC=0;
    TRISD=64;
    PORTD=128;
    if ( RC0_bit==0 && RD6_bit==0 ){
        cont++;
        delay_ms(100);
        PORTC=30;
        PORTD=188;
        if ( RC0_bit==1 && RD6_bit==1){
            cont++;
            delay_ms(100);
        }
    }
    if(cont==2){
        res_chip=1;
    }
    else{
    res_chip=2;
    }
}
void Cand73(){
    TRISC=32;
    PORTC=59;
    TRISD=12;
    PORTD=242;
    if ( RD2_bit==1 && RC5_bit==0 && RD3_bit==1 ){
        res_chip=1;
    }
    else{
    res_chip=2;
    }
}
void Cor75(){
    TRISC=32;
    PORTC=8;
```

```
    TRISD=12;
    PORTD=128;
    if ( RD2_bit==0 && RC5_bit==1 && RD3_bit==0 ){
        cont++;
        delay_ms(100);
        PORTC=31;
        PORTD=242;
        if ( RD2_bit==1 && RC5_bit==1 && RD3_bit==1){
            cont++;
            delay_ms(100);
        }
    }
    if(cont==2){
        res_chip=1;
    }
    else{
    res_chip=2;
    }
}
        //Analiza el funcionamiento de un REGISTRO  de  la  serie 74LS195//

        void shift195()

        //Analiza el funcionamiento de un REGISTRO  de  la  serie 74LS198//

        void shift198()

        //Analiza el funcionamiento de un BUFFER  de  la  serie 74LS240//

        void buffer240()

        //Analiza el funcionamiento de un BUFFER  de  la  serie 74LS240//

        void buffer244()

        //Analiza el funcionamiento de un BUS  de  la  serie 74LS245//

        void bus245()

        {

            TRISC=254;

            PORTC=0;

            TRISA=1;

            PORTA=12;

            TRISD=0;

            PORTD=150;

            retardo();
```

```
if (RC1_bit==0 && RC2_bit==1 && RC3_bit==0 && RC4_bit==1 && RC5_bit==1
    && RC6_bit==0 && RC7_bit==1 && RA0_bit==1){
     cont++;
}
else{
    res_chip=2;
    goto FIN;
}
TRISC=0;
PORTC=107;
TRISA=12;
PORTA=1;
TRISD=63;
PORTD=128;
retardo();
 if (RA2_bit==1 && RA3_bit==0 && RD0_bit==1 && RD1_bit==1 && RD2_bit==0
     && RD3_bit==1 && RD4_bit==0 && RD5_bit==1){
      cont++;
 }
 else{
     res_chip=2;
     goto FIN;
 }
if (cont==2){
   res_chip=1;
}
else{
```

```
            res_chip=2;
        }
        FIN: cont=0;
    }

////////////////////////////
// INICIALIZACION DEL CHIP //
////////////////////////////
void InitMain() {
  ADCON1 = 255;
  OSCCON = 126;                  //CONFIGURACION DEL CLOCK EN 8MHZ
  TRISA = 0;                   // configure trisA pins as input
  PORTA=0;
  PORTB = 0;                     // set PORTB to 0
  TRISB = 255;                    // designate PORTB pins as INput
  PORTC = 0;                     // set PORTC to 0
  TRISC = 0;                    // designate PORTC pins as output
  PORTD = 0;
  TRISD = 0;
  PORTE = 0;
  TRISE = 0;
  RE3_bit=1;
  cont=0;
  tipo_chip=0;
  res_chip= 0;
}
```

3.2.2.3 Programa Principal (Main)

```
void main()
{
    EMPEZAR:  InitMain();


    NUEVO:    delay_ms(1000);
            while(PORTB!=0)
            {
                tipo_chip=PORTB;
                delay_ms(1000);
                    //case para escoger el cada uno de los integrados
```

```
switch (tipo_chip)
{
   case 1:nand();break;          //7401
   case 2:nor();break;           //7402
   case 3:nand03();break;        //7403
   case 4:inv();break;           //7404
   case 5:inv();break;           //7405
   case 6:inv();break;           //7406
   case 7:and09();break;         //7409
   case 8:nand10();break;        //7410
   case 9:and11();break;         //7411
   case 10:nand13();break;       //7413
   case 11:inv();break;          //7414
   case 12:inv();break;          //7416
   case 13:ninv();break;         //7417
   case 14:nand13();break;       //7420
   case 15:and21();break;        //7421
   case 16:nand13();break;       //7422
   case 17:nor25();break;        //7425
   case 18:nor22();break;        //7427
   case 19:nor();break;          //7428
   case 20:nand30();break;       //7430
   case 21:or32();break;         //7432
   case 22:nor();break;          //7433
   case 23:nand03();break;       //7437
   case 24:nand03();break;       //7438
```

```
case 25:nand13();break;        //7440
case 26:decoder42();break;     //7442
case 27:decoder42();break;     //7445
case 28:decoder47();break;     //7447
case 29:decoder48();break;     //7448
case 30:decoder49();break;     //7449
case 31:elemento51();break;    //7451
case 32:contador169();break;   //74169
case 33:paridad180();break;    //74180
case 34:elemento54();break;    //7454
case 35:elemento54();break;    //7455
case 36:shift194();break;      //74194
case 37:ninv();break;          //7471
case 38:shift195();break;      //74195
case 39:flipflop73();break;    //7473
case 40:flipflop74();break;    //7474
case 41:lanch75();break;       //7475
case 42:flipflop76();break;    //7476
case 43:flipflop377();break;   //74377
case 44:flipflop373();break;   //74373
case 45:suma83();break;        //7483
case 46:buffer367();break;     //74367
case 47:buffer368();break;     //74368
case 48:contador90();break;    //7490
case 49:shift91();break;       //7491
case 50:contador92();break;    //7492
case 51:contador93();break;    //7493
```

```
case 52:sumador283();break;    //74283
case 53:shift95();break;       //patita 3 mal
case 54:shift96();break;       //7496
case 55:selector257();break;   //74257
case 56:selector258();break;   //74258
case 57:flipflop273();break;   //74273
case 58:flipflop107();break;   //74107
case 59:flipflop109();break;   //74109
case 60:flipflop112();break;   //74112
case 61:flipflop122();break;   //74122
case 62:bus245();break;        //74245
case 63:buffer125();break;     //74125
case 64:buffer126();break;     //74126
case 65:multiplexer251();break;//74251
case 66:nand03();break;        //74132
case 67:nand133();break;       //74133
case 68:exor136();break;       //74136
case 69:decoder137();break;    //74137
case 70:decoder137();break;    //74138
case 71:decoder139();break;    //74139
case 72:buffer244();break;     //74244
case 73:buffer240();break;     //74240
case 74:shift179();break;      //74179
case 75:decoder42();break;     //74145
case 76:decoder147();break;    //74147
case 77:decoder148();break;    //74148
case 78:flipflop175();break;   //74175
```

```
case 79:multiplex151();break;  //74151
case 80:multiplex153();break;  //74153
case 81:decoder154();break;    //74154
case 82:demultiplex155();break;//74155
case 83:demultiplex155();break;//74155
case 84:multiplex157();break;  //74157
case 85:multiplex158();break;  //74158
case 86:contador160();break;   //74160
case 87:contador161();break;   //74161
case 88:contador163();break;   //74163
case 89:shift164();break;      //74164
case 90:shift165();break;      //74165
case 91:shift166();break;      //74166
case 92:shift173();break;      //74173
case 93:shift174();break;      //74174
case 94:ram170();break;        //74170
case 95:sum181();break;        //74181
case 96:contador190();break;   //74190 -
case 97:contador191();break;   //74191 -
case 98:contador192();break;   //74192 -
case 99:contador193();break;   //74193-
case 100:shift198();break;     //74198
case 102:shift299();break;     //74299
case 103:contador390();break;  //74390 -
case 104:contador393();break;  //no sale

case 105:Cnor00();break;       //4000
```

```
case 106:Cnor01();break;       //4001
case 107:Cnor02();break;       //4002
case 108:Cnand11();break;      //4011
case 109:Cnand12();break;      //4012
case 110:Cdecada17();break;    //4017
case 111:Ccontador20();break;  //4020
case 112:Cnand23();break;      //4023
case 113:Ccontador24();break;  //4024
case 114:Cnor25();break;       //4025
case 115:Cdecoder26();break;   //4026
case 116:Cdecoder28();break;   //4028
case 117:Ccontador29();break;  //4029
case 118:Cexor30();break;      //4030
case 119:Ccontador40();break;  //4040
case 120:Cinv49();break;       //4049
case 121:Cninv50();break;      //4050
case 122:Cnand68();break;      //4068
case 123:Cinv69();break;       //4069
case 124:Cexor30();break;      //4070
case 125:Cor71();break;        //4071
case 126:Cor72();break;        //4072
case 127:Cand73();break;       //4073
case 128:Cor75();break;        //4075
case 129:Cexor30();break;      //4077
case 130:Cand81();break;       //4081
case 131:Cand82();break;       //4082
case 132:Cnand11();break;      //4093
```

```
                default: ("PIN NO ENCONTRADO!");
              }
              goto SALIR;
            }
            goto NUEVO;
 SALIR:   delay_ms(100);
          PORTE=res_chip;
          retardo();
          retardo();
          goto   EMPEZAR;
}
end;
```

3.3 Diseño de componentes de Hardware

A continuación se especificara las características más relevantes de los módulos físicos y su utilización en el proyecto.

3.3.1 El Microcontrolador

El PIC18F452-I/P es un microcontrolador de 8 bits del tipo flash, con memoria de programa de 32kB y una RAM de 1536 Bytes. Este último dato fue muy importante para la selección del micro debido a que la mayoría de los dispositivos disponibles tenían tan solo la mitad de Memoria RAM. Además tiene 40 pines, de los cuales 35 pueden ser

entradas o salidas en el proyecto se utilizo todos estos pines en los dos Microcontroladores.

Figura 3.1 El microcontrolador

3.3.2 Pantalla Gráfica GLCD 128X64

El menú y el ingreso de datos es mostrada a través de una pantalla G LCD, está graficá resolución de 128X64 pixeles La pantalla es monocromática RGB es decir puede mostrar la información en cualquiera de los tono que se pueda obtener combinando los colores rojo, verde y azul. Su controlador es un chip Samsung S6B0108 (KS0108).

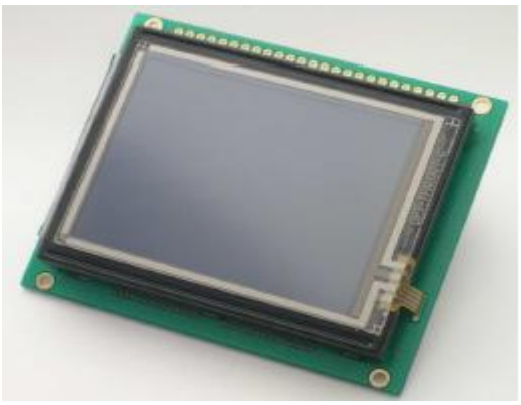


Figura 3.2 Pantalla Gráfica GLCD 128X64

CAPÍTULO 4

4. Simulación y Pruebas Experimentales

4.1 Presentación de menú en la pantalla GLCD

Al inicio se observa la presentación del menú de la figura 4.1 que tiene como opción la familia TTL y CMOS.

Al presionar la segunda opción en la figura 4.1 entrará al segundo menú de la figura 4.2 en donde se observan las dos opciones de las series que se pueden verificar en la familia CMOS.

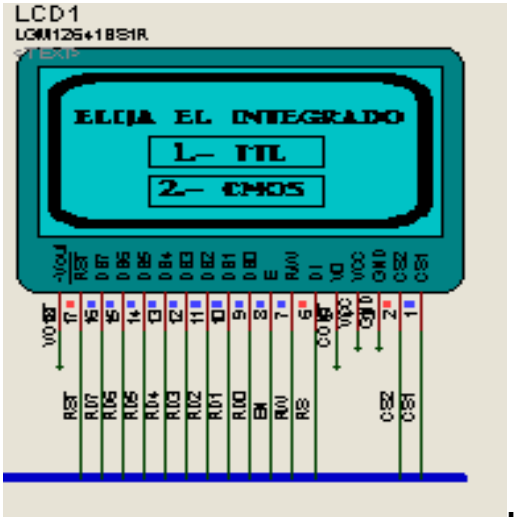


Figura 4.1 Simulación MENU1

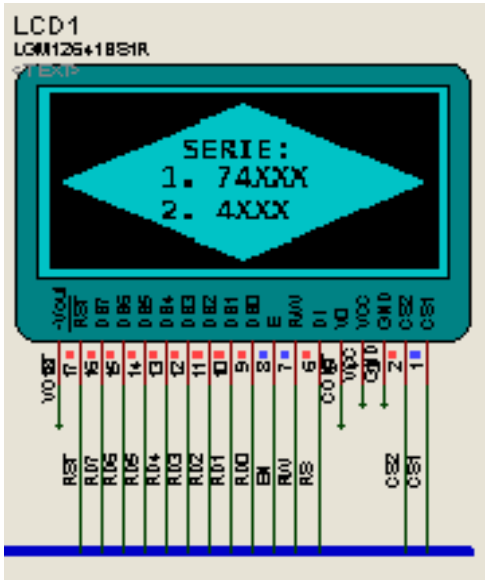


Figura 4.2 Simulación MENU2

4.2 Ingreso de la serie del integrado

En la figura 4.3 se observa el circuito completo y además el ingreso de un integrado mediante teclado para ser analizado

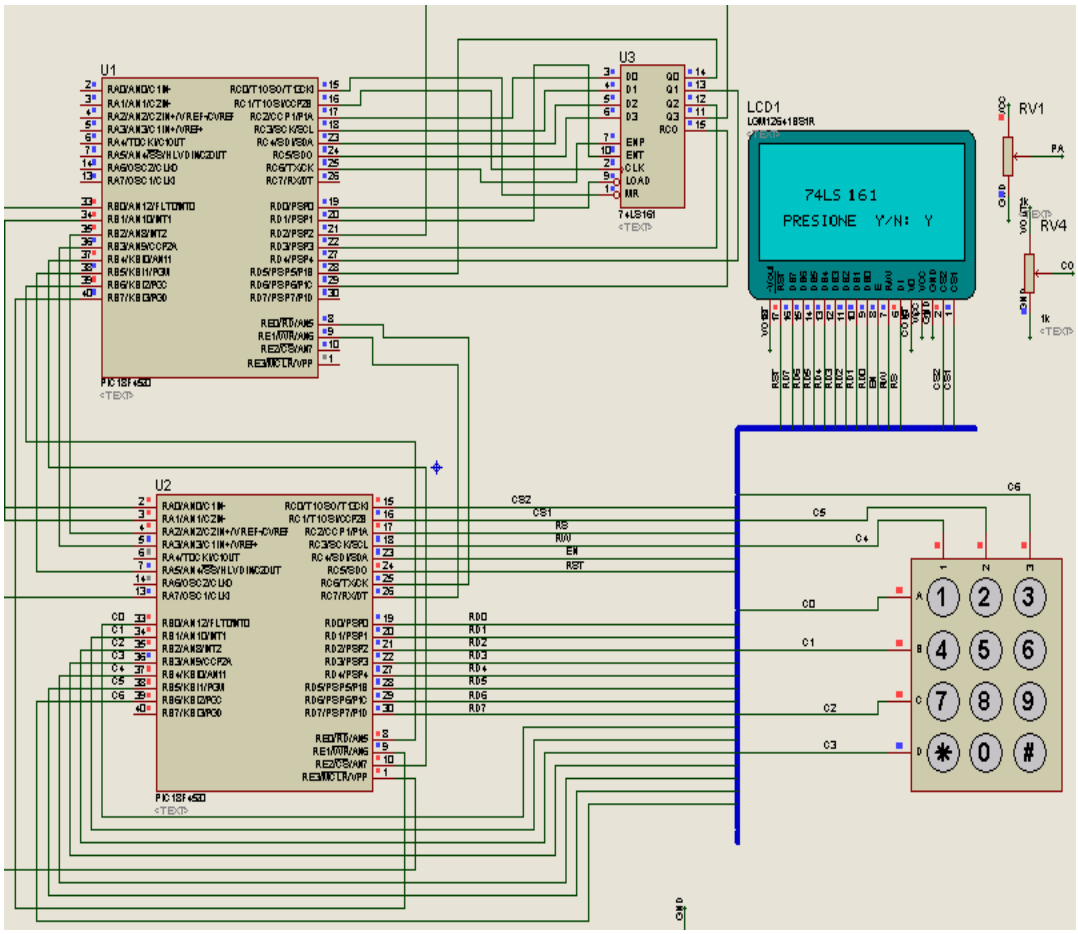


Figura 4.3 Simulación del ingreso del C.I.

4.3 Integrado en buen estado

En la figura 4.4 se observa el mensaje en la pantalla después del analices que en este caso el circuito integrado se encuentra en buen estado.

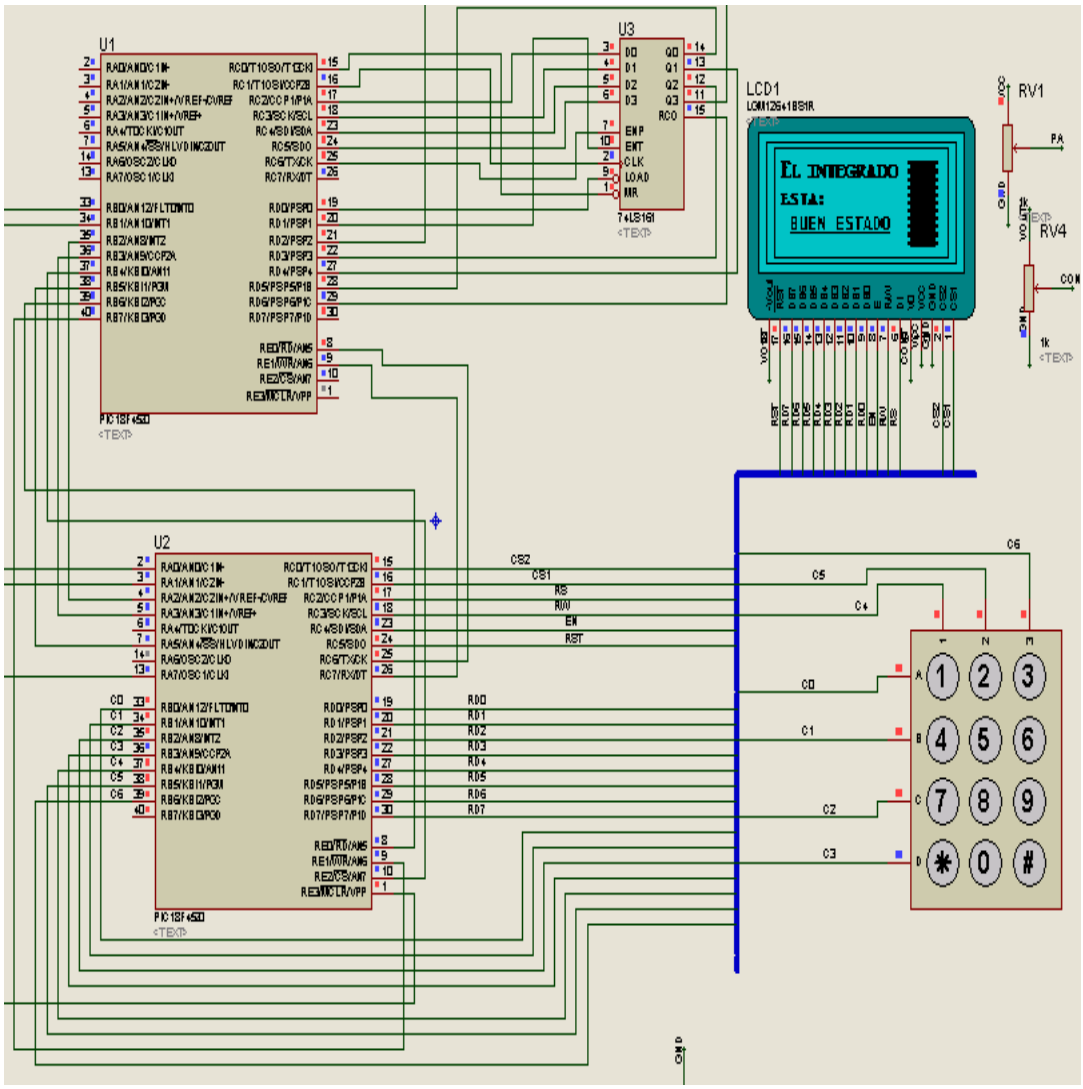


Figura 4.4 Simulación del C.I. en buen estado

4.4 Integrado en mal estado

En la figura 4.4 se observa el mensaje en la pantalla después del analices que en este caso el circuito integrado se encuentra en mal estado

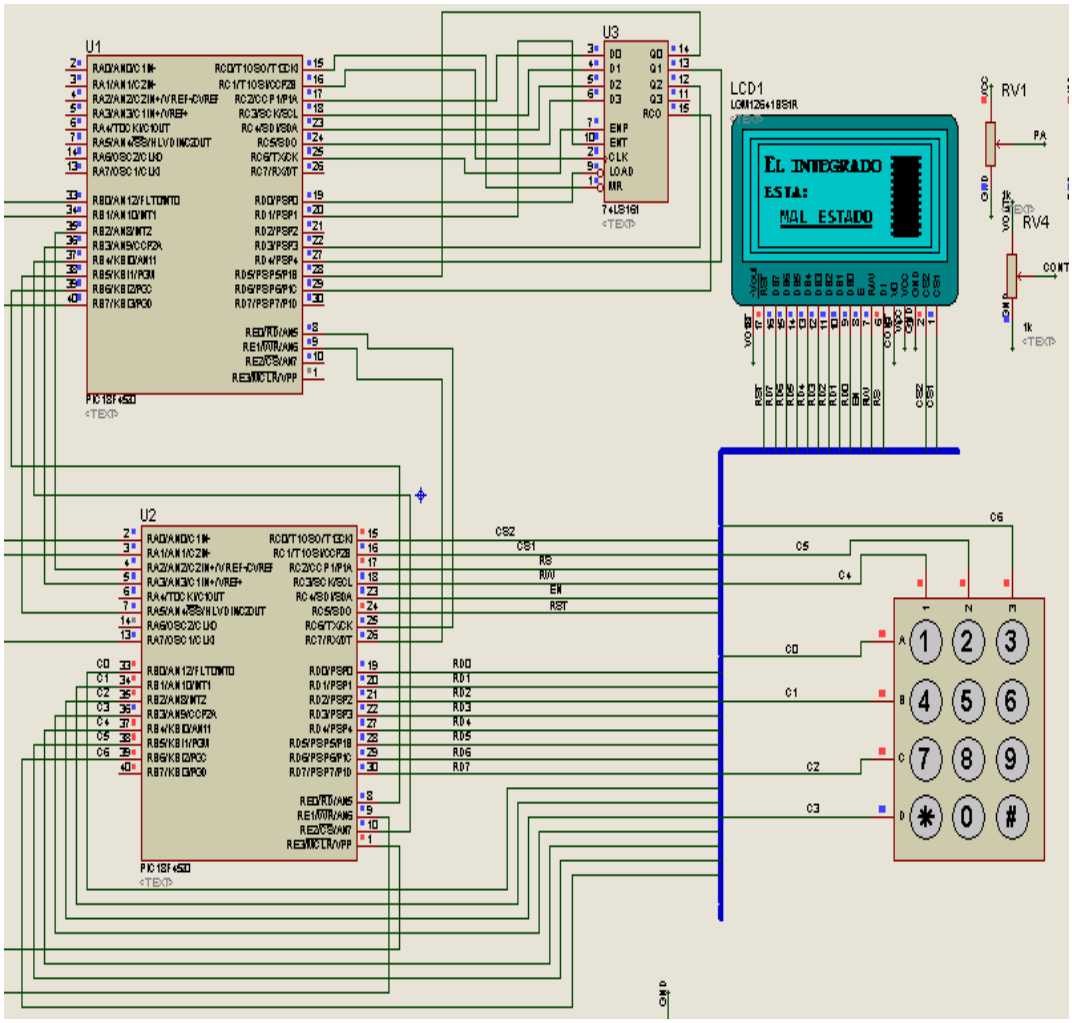


Figura 4.5 Simulación del circuito en mal estado

Conclusiones y Recomendaciones

Las conclusiones son:

1. En este proyecto jugó un papel muy importante la velocidad del µC, debido a que fue de gran ventaja al momento de realizar las pruebas para la verificación del funcionamiento de cada integrado, dado que estas se completan con gran rapidez en la mayoría de chips, cabe recalcar que el tiempo de comprobación de cada integrado es distinto, esto depende de su funcionalidad.

2. Para la verificación del integrado se consideró su funcionalidad, considerando la hoja de datos de cada uno de ellos. De esta forma se realiza las pruebas de una manera eficiente asegurando un resultado óptimo.

3. Este comprobador analiza la mayoría de integrados utilizados en el Laboratorio de Digitales de la Facultad de Ingeniería en Electricidad y Computación, además, se incluyó integrados de la familia CMOS que no son utilizados en el laboratorio anteriormente especificado, debido a que

estos integrados son más sensibles que los de la familia TTL. También, se consideró la compatibilidad entre estas dos familias de circuitos integrados.

4. Este equipo comprobador de chips es muy útil, considerando que es práctico para establecer el estado de trabajo de diversos chips integrados TTL y CMOS. Además, ofrece características que lo hace sencillo al momento de la manipulación, como la movilidad y el fácil uso. Cabe recalcar que para utilizarlo se debe conocer la serie del integrado.

5. Debido, a la cantidad de pruebas que se debe realizar para cada integrado, la memoria de programación del microcontrolador no debe ser menor a 32Kb para la cantidad de integrados que se programó. Se puede incrementar el número de chips con microcontroladores de mayor capacidad de memoria, o bien con el mismo habilitando la memoria externa.

Las recomendaciones son:

1. Revisar y entender el manual de especificaciones del microcontrolador y de los circuitos integrados para su buen funcionamiento y de esta manera no cometer errores en la conexión de sus pines. En el caso del microcontrolador 18F4520, si no se utiliza el reloj externo configurar sus pines como salida, así evitaremos que ingrese datos erróneos y falle la programación.

2. Calibrar el potenciómetro que controla el contraste en la pantalla GLCD para visualizar las letras o gráficos, caso contrario no podrá haber interacción con el usuario, ni proceder a la comprobación del chip.

3. Para comprobar el funcionamiento de un Chip se debe saber la serie del mismo en caso de colocar en el zócalo un integrado diferente al ingresado por teclado se mostrará datos erróneos en la pantalla GLCD. Asegurarse de colocar de forma correcta el circuito integrado en el zócalo para evitar fallas en la comprobación y daños en el mismo.

4. El programa MikroC Pro for PIC resulta una herramienta muy práctica para el manejo de las pantallas GLCD y el teclado, para realizar una buena programación se debe consultar con las librerías existentes en la opción HELP.

ANEXOS

Anexo 1

Lista de los integrados

Integrados	Características	Integrados	Características
74ls01	NAND	74ls136	EXOR
74ls02	NOR	74ls137	DECODER/MUX
74ls03	NAND	74ls138	MULTIPLEXOR
74ls04	INVERSOR	74ls139	DECODER MULTIPLE
74ls05	INVERSOR	74ls141	DECODER BCD - DECIMAL
74ls06	INVERSOR	74ls142	CONTADOR,DRIVER,DECODER
74ls09	AND	74ls143	CONTADOR
74ls10	NAND	74ls145	DECODER BCD - DECIMAL
74ls11	AND	74ls147	DECODER DECIMAL -BCD
74ls13	NAND 4entradas	74ls148	8 a 3 ENCODER
74ls14	INVERSOR	74ls150	DATA SELECTOR MULTIPLEXOR
74ls16	INVERSOR	74ls151	SELECTOR MULTIPLEXOR
74ls17	BUFFER	74ls153	MULTIPLEXOR 4 a 2
74ls20	NAND 4entradas	74ls154	DEMULTIPLEXADOR
74ls21	AND 4etradas	74ls155	MULTIPLEXADOR
74ls22	NAND 4entradas	74ls156	MULTIPLEXADOR
74ls25	NOR	74ls157	SELECTOR MULTIPLEXOR
74ls27	NOR 3entradas	74ls158	SELECTOR, MUX
74ls28	NOR 2entradas	74ls160	CONTADOR DECADA
74ls30	NAND 8puertos	74ls161	CONTADOR BINARIO
74ls32	OR 2entradas	74ls163	CONTADOR BINARIO
74ls33	NOR buffer	74ls164	SHIFT REGISTER
74ls37	NAND 2entradas	74ls165	SHIFT REGISTER
74ls38	NAND buffer	74ls166	SHIFT REGISTER
74ls40	NAND buffer 4 puertos	74ls167	LACHE DECADA RATE
74ls42	DECO. Bcd-decimal	74ls168	CONTADOR UP-DOWN
74ls45	DECO. Bcd-decimal	74ls169	CONTADOR 4 bits
74ls47	DECO. Bcd-7seg.	74ls170	REGISTRO FILAS
74ls48	DECO. Bcd-7seg.	74ls173	BITS REGISTRO
74ls49	DECO.	74ls174	FLIPFLOP

Integrados	Características	Integrados	Características
74ls74	FLIK POT DUAL	74ls192	CONTADOR BINARIO UP-DOWN
74ls75	4 BIT PLATCH	74ls193	CONTADOR BINARIO UP-DOWN
74ls76	JK	74ls194	REGISTRO
74ls80	ADREES	74ls195	SHIFT REGISTER
74ls82	FULL ADRESS	74ls198	SHIFT REGISTER
74ls83	SUMATORIO	74ls201	RAM 256
74ls84	RAM	74ls221	DUAL SHORT
74ls89	REGISTRO	74ls240	INVERSOR BUFFER
74ls90	R/W MEMORIES	74ls97	MULTIPLICADOR BINARIO
74ls91	SHIFT REGISTER	74ls98	SELECTOR REGISTRO
74ls92	DIVISOR CONTADORES	74ls100	BISTABLE LACTHES
74ls93	CONTADOR	74ls107	DUAL FF con CLR
74ls94	REGISTRO	74ls109	DUAL FF con CLR
74ls95	REGISTRO	74ls112	JK TRIGER
74ls96	REGISTRO	74ls122	REGISTRO MONOSTABLE
74ls124	CONTROLLER OSCILETOR	74ls390	
74ls125	BUFFER	74ls393	CONTADOR BINARIO
74ls126	BUFFER 3 estados	74ls51	CONTADOR
74ls128	LINE DRIVERS	74ls52	DECOR.
74ls132	NAND SCHMITT TRIGER	74ls53	PARIDAD
74ls133	NAND	74ls54	AND-OR inversor
74ls136	EXOR	74ls55	AND-OR inversor 4ent.
		74ls70	AND JK
		74ls71	SUMADOR
		74ls72	AND disparador JK
		74ls73	AND 2JK

Integrados	Características	Integrados	Características
4000	NOR	4050	NORINVERSOR
4001	NOR	4068	NAND
4002	NOR	4069	INVERSOR
4011	NAND	4070	EXOR
4012	NAND	4071	OR
4017	DECADA	4072	OR
4020	CONTADOR	4073	AND
4023	NAND	4075	OR
4024	CONTADOR	4077	EXOR
4025	NOR	4081	AND
4026	DECODER	4082	AND
4028	DECODER	4093	NAND
4029	CONTADOR		
4030	EXOR		
4040	CONTADOR		
4049	INVERSOR		

Anexo 2

Hoja de datos técnicos GLCD 128X64

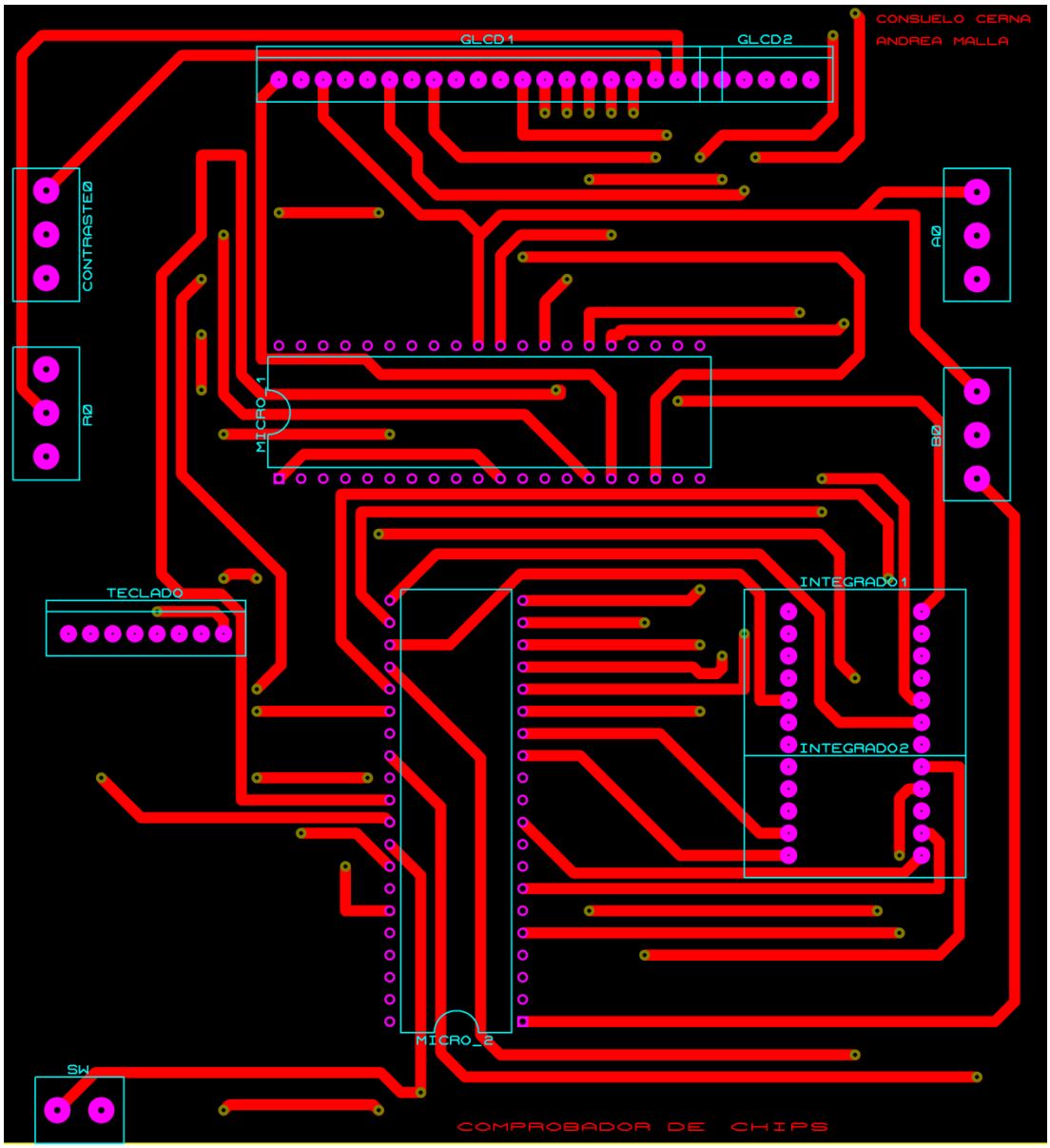
Anexo 3

Hoja de datos técnicos
PIC18F4520

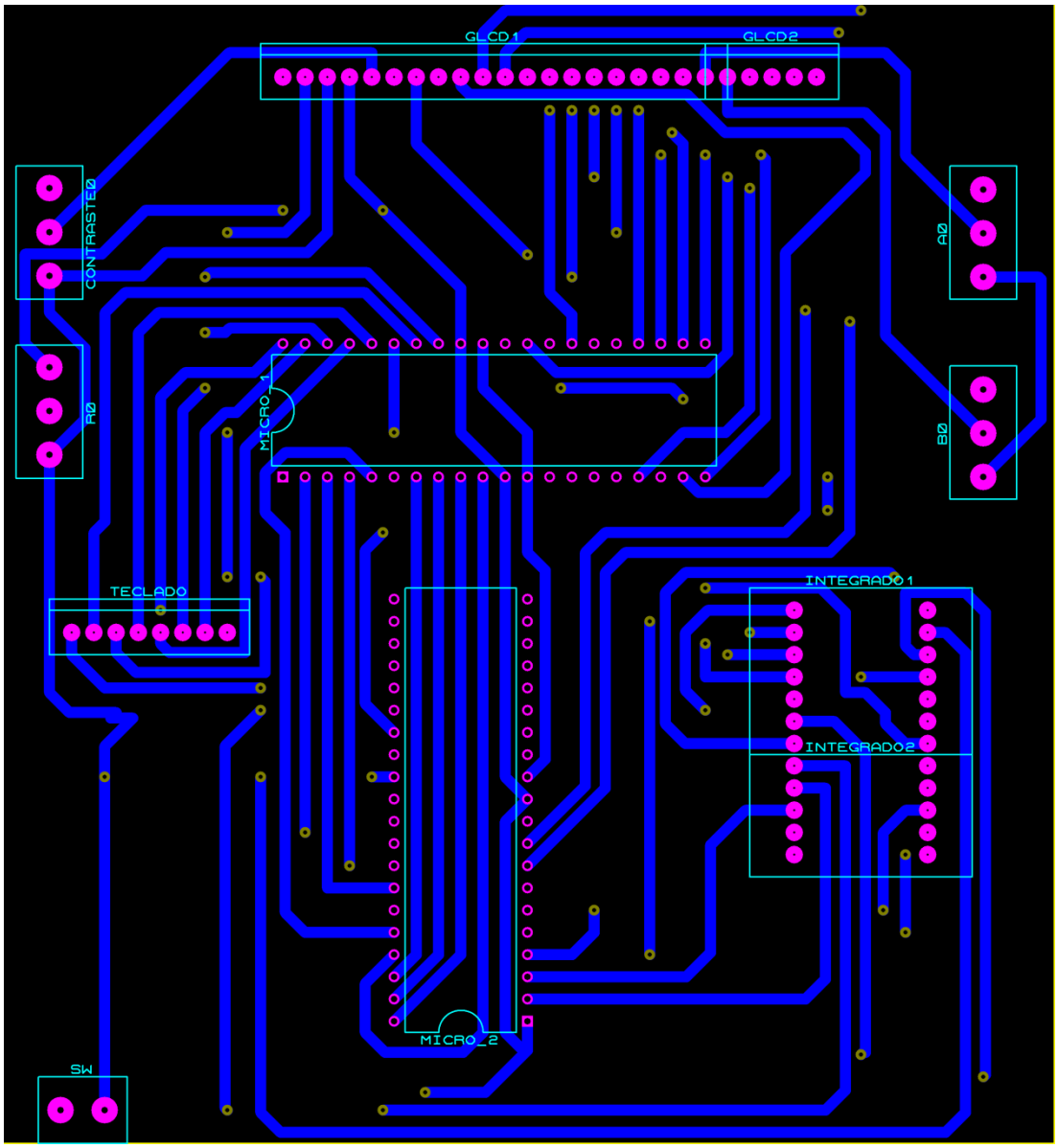
ANEXO 4

Esquemático y PCB del circuito

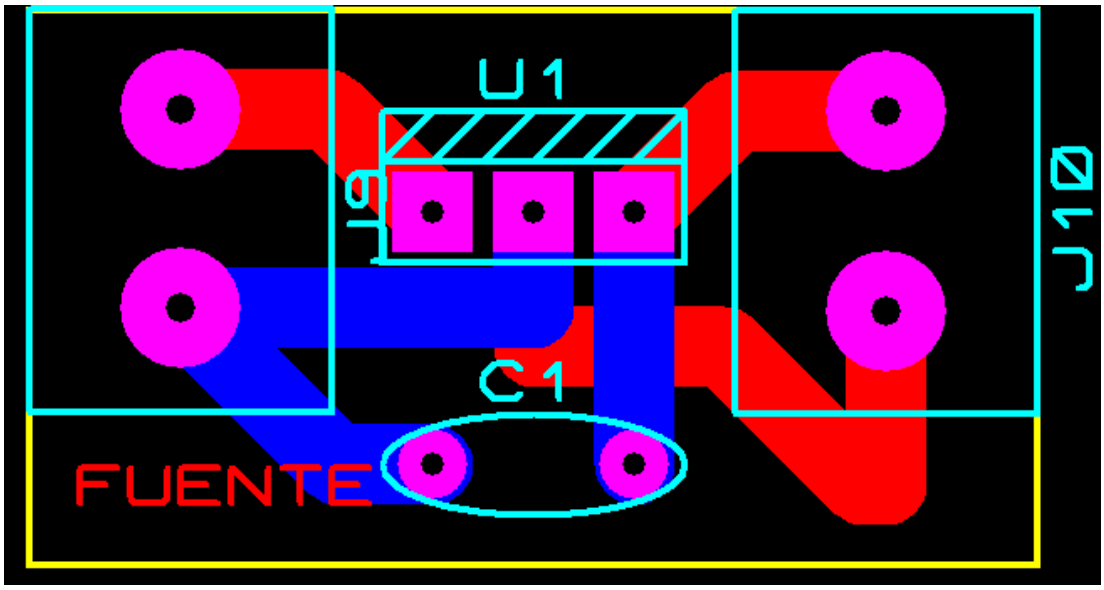
PCB Cara A


CONSUELO CERNA
ANDREA MALLA
GLCD1
GLCD2
CONTRASTE0
A0
R0
B0
MICRO_1
TECLADO
INTEGRADO1
INTEGRADO2
MICRO_2
SW
COMPROBADOR DE CHIPS

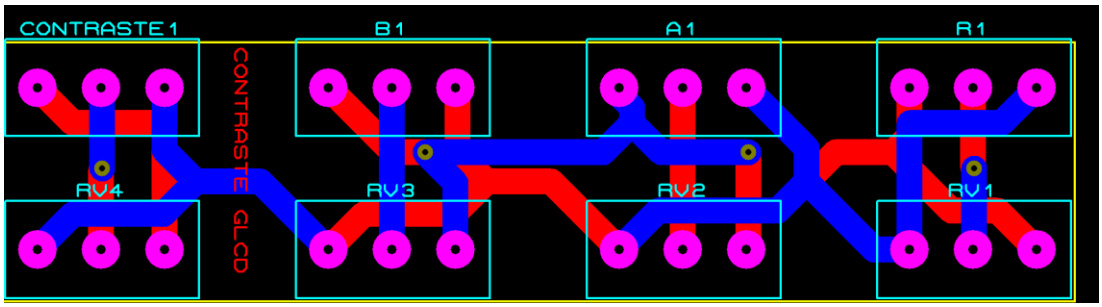
PCB Cara B


GLCD1
GLCD2
CONTRASTE0
A0
R0
MICRO_1
B0
TECLADO
INTEGRADO1
INTEGRADO2
MICRO_2
SW

FUENTE 5 V



CONTRASTE GLCD



ANEXO 5

Lista de Materiales y Presupuesto

Cant	Descripción	Precio U	Precio T
1	Pantalla GLCD 128X64 RGB + Panel táctil	$ 55,00	$ 55,00
2	PIC18F4520 microcontrolador	$ 9,00	$ 18,00
1	LM7805 Regulador de voltaje +5V	$ 0,25	$ 0,25
2	Tira de espadines	$ 0,50	$ 1,00
4	Tira de espadines hembra	$ 1,20	$ 4,80
1	Capacitor 100nf	$ 0,35	$0,35
1	Zócalo 24 pines	$ 3,50	$ 3,50
1	Switch 2 Posiciones	$ 0,35	$ 0,35
3	Fabricación PCB		$ 35
1	Caja Acrílico	$23,00	$23,00
1	Batería 9v	$ 3,50	$ 3,50
1	Teclado	$ 5,00	$ 5,00
4	Potenciómetros	$ 0,50	$2,00
		TOTAL	$ 143,25

BIBLIOGRAFÍA

[1]. Microchip, Data Sheet PIC18F4520; http://ww1.microchip.com/downloads/en/DeviceDoc/39631a.pdf ; **Fecha de consulta: 15**/Agosto/2010.

[2]. Mikroelektronica; Manual de Usuario MikroBasic Pro for PIC; http://www.mikroe.com/eng/downloads/get/37 ; **Fecha de consulta:** 20/Agostol/2010.

[3]. Mikroelektronica; Guía de Referencia MikroBasic; http://www.mikroe.com/pdf/mikroCPro ; **Fecha de consulta:** 20/Agosto/2010.

[4]. Mikroelektronica; Presentación Pantalla GLCD; http://www.mikroe.com/eng/downloads/get/468/es_mikroe_article_basic_avr_01_09.pdf ; **Fecha de consulta:** 02/Septiembre/2010.

[5]. Mikroelektronica; Presentación Pantalla GLCD; http://www.mikroe.com/eng/downloads/get/468/es_mikroe_article_basic_avr_01_09.pdf ; **Fecha de consulta:** 02/Septiembre/2010.

[6]. Datasheet; hoja de datos de circuitos integrados familia TTL; http://www.dainau.com/ttl_datasheet.htm ; **Fecha de consulta:** 05/Septiembre/2010.

[7]. Conceptos; familia TTL y CMOS; http://focus.ti.com/docs/prod/folders/print/sn7400.html; **Fecha de consulta:** 10/Septiembre/2010.