ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL
Facultad de Ingeniería EN ELECTRICIDAD Y COMPUTACIÓN

Diseño e Implementación de una Balanza Electrónica en Base al Microcontrolador PIC 16LF877

TESIS DE GRADO
Previo a la obtención del Título de:
INGENIERO EN ELECTRICIDAD ESPECIALIZACIÓN: ELECTRÓNICA INDUSTRIAL

Presentado por:
SERGIO TRUJILLO SÁNCHEZ

GUAYAQUIL-ECUADOR
2004

AGRADECIMIENTO

A todas las personas que de uno u otro modo colaboraron en la realización de este trabajo y especialmente en el Ing. Hugo Villavicencio Director de Tesis, por su invaluable ayuda.

DEDICATORIA

MIS PADRES

A MIS HERMANOS

A UN AMIGO

TRIBUNAL DE GRADUACIÓN

_____	_____
Ing. Miguel Yapur SUBDECANO DE LA FIEC PRESIDENTE	Ing. Hugo Villavicencio DIRECTOR DE TESIS
_____	_____
Ing. Alberto Larco VOCAL	Ing. Francisco Novillo VOCAL

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta Tesis de Grado, me corresponde exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL"

(Reglamento de Graduación de la ESPOL)

_____

Sergio Trujillo Sánchez

RESUMEN

El proyecto de tesis se divide en tres partes para una mejor comprensión.

Brevemente me referiré al funcionamiento de la balanza y algunos conceptos básicos sobre los microcontroladores y las galgas extensiométricas.

En el capítulo 1 se detalla el manejo de los microcontroladores, sus características relevantes, su arquitectura interna, los diferentes tipos de registros, diferentes puertos y el convertidor analógico digital que posee; además, los distintos periféricos como: la interfase con el LCD, las instrucciones y comandos del despliegue visual, diagramas de tiempos de la fase de lectura y escritura, el sensor, el Puente de Wheatstone y el acondicionamiento de señal.

El capítulo 2 constatará de los diferentes programas en lenguaje ensamblador para la comunicación del micro PIC con los distintos periféricos.

El capítulo 3 consta de las recomendaciones para una mejora de posteriores balanzas, las observaciones, las conclusiones y anexos, como tablas de los registros del PIC, un manual del usuario de la balanza, bibliografía.

INDICE GENERAL

RESUMEN VI

INDICE GENERAL VII

INDICE DE FIGURAS XI

INDICE DE TABLAS XIII

INTRODUCCIÓN 1

CAPITULO 1

1. PARTICIÓN FUNCIONAL DE LA BALANZA 3

1.1. Controlador de la Balanza 4

1.1.1. Registros Especiales del Área de Datos 5

1.1.2. Arquitectura Interna y Organización de la Memoria 6

1.1.3. La Memoria de Datos 8

1.1.4. El PC 10

1.1.4.1. La Pila 11

1.1.4.2. Direccionamiento de Datos 12

1.1.5. Registros de Funciones Especiales 13

1.1.5.1. Registro de Estado 13

1.1.5.2. Registro de Opciones 13

1.1.5.3. Registro de Interrupciones INTCON 14

1.1.5.4. Otros Registros 14

1.1.6. Recursos Comunes de los PICS 15

1.1.6.1. El Oscilador Principal 15

1.1.6.2. Perro Guardián WDT 16

1.1.6.3. Temporizador TMR0 16

1.1.6.4. Reinicialización o Reset 17

1.1.6.5. Modo de Reposo SLEEP 18
1.1.7. Interrupciones 19
1.1.7.1. Causas de la Interrupción 20
1.1.8. Puertos de Entrada Salida 22
1.1.8.1. Puerto A 23
1.1.8.2. Puerto B 23
1.1.8.3. Puerto C 23
1.1.8.4. Puerto D 24
1.1.8.5. Puerto E 25
1.1.9. Puerto Serie Síncrono (SSP) 25
1.1.9.1. Modo SPI 26
1.1.9.2. Modo I^2C 28
1.1.10. Interfase de Comunicación Serie (SCI) 29
1.1.11. Convertidor Analógico Digital 31
1.2. El Sensor 34
1.2.1. Las Galgas Extensiométricas 34
1.2.2. Control de Ruido de las Galgas 35
1.2.2.1. Técnicas de Reducción de Ruido 36
1.2.3. El Puente de Wheatstone 36
1.3. Acondicionamiento de Señal 39
1.3.1. Amplificación 40
1.3.1.1. Amplificadores de Instrumentación 40
1.3.1.2. Codificación Circuital del Amplificador de Instrumentación 44
1.3.2. Excitación 46
1.3.3. Filtrado 47
1.3.3.1. Filtros Activos 47

1.3.3.1.1. Filtro Pasa Bajo 50
1.4. LCD – HD – 44780 51
1.4.1. Comunicación entre el LCD y el PIC 54
1.4.2. Secuencia de Inicialización 55
1.4.3. Diagramas de Tiempo 56
1.5. El Teclado 57
1.5.1. Comunicación entre el Teclado y el PIC 59
1.6. Comunicación Serie RS232 63
1.6.1. Puerto Serie 64
1.6.2. Pines del DB – 9 67
1.7. Fuente de Alimentación 68
1.8. Esquemático de la Balanza 70

CAPITULO 2

2. DESARROLLO DEL SOFTWARE 73
2.1. Programación en Lenguaje Ensamblador 74
2.2. El Programa Principal 80
2.2.1. Diagrama de Flujo 80
2.2.2. Codificación en Lenguaje Ensamblador 82
2.3. Subrutina de Interfase con el LCD 83
2.3.1. Diagrama de Flujo 83
2.3.2. Codificación en Lenguaje Ensamblador 85
2.4. Subrutina de Interfase con el Teclado 86
2.4.1. Diagrama de Flujo 86
2.4.2. Codificación en Lenguaje Ensamblador 87
2.5. Programación para la Conversión Analógica 88

2.5.1. Diagrama de Flujo 89
2.5.2. Codificación en Lenguaje Ensamblador 90
2.5.3. Conversión a BCD 90
2.6. Subrutina de Comunicación con el USART 91
2.6.1. Codificación en Lenguaje Ensamblador 92
2.7. Módulo de EEPROM (Lectura/Escritura) 92
2.7.1. Codificación en Lenguaje Ensamblador 94
2.8. Interfase de Comunicación con la PC 95
2.8.1. Codificación en Visual Basic 96

CAPITULO 3

3. CONCLUSIONES Y RECOMENDACIONES 97
3.1. Sobre los Circuitos Impresos 97
3.2. Mejoras a la Balanza 98
3.3. Observaciones 99
3.4. Conclusiones 101
ANEXOS 103
Tablas de los Registros del PIC 104
Diagramas de flujo de Rutinas del Programa 111
Manual del Usuario 116
Diagramas Esquemáticos de la Balanza 121
Fotos del Proyecto 125
Simulación en Proteus 128
Datos Técnicos de la Balanza 134
BIBLIOGRAFIA 135

INDICE DE FIGURAS

Figura 1: Partición Funcional de la Balanza 4

Figura 2: Mapa de Memoria y Pila 7

Figura 3: Arquitectura Interna de los PICS 16F87X 8

Figura 4: Memoria de Datos 9

Figura 5: Modos de Direccionamiento 13

Figura 6: Organigrama de las Operaciones de una Interrupción 21

Figura 7: Diagrama de Pines del PIC 16F877 22

Figura 8: Modo SPI 27

Figura 9: Modo I^2C 29

Figura 10: Puente de Wheatstone 37

Figura 11: Esquema Básico de Medición 41

Figura 12: Filtro de Corriente Alterna a la Entrada 44

Figura 13: Amplificador de Instrumentación 45

Figura 14: Respuesta de Frecuencia de los Filtros Pasa Bajo 50

Figura 15: Filtro Pasa Bajo 51

Figura 16: Diagramas de Tiempo del Ciclo de Escritura 57

Figura 17: Teclado Matricial 58

Figura 18: Nivel Lógico 0 60

Figura 19: Nivel Lógico 1 61

Figura 20: Comunicación Serie RS232 63

Figura 21: Fuente de Alimentación 69

Figura 22: Esquemático del Microcontrolador y la Fuente de Poder 70

Figura 23: Esquemático del Teclado, LCD y Max 232 71

Figura 24: Esquemático del Acondicionamiento de Señal 71

Figura 25: Esquemático de Entrada y Salida de Señales 72

Figura 26: Pantalla del Programa en Visual Basic 95

INDICE DE TABLAS

Tabla I: Ganancias Según el Orden del Filtro 49

Tabla II: Descripción de los pines del Display LCD 52

Tabla III: Principales Instrucciones del Display LCD 53

Tabla IV: Señales de Tiempo del Ciclo de Escritura 57

Tabla V: Patillaje del DB – 9 68

Tabla VI: Instrucciones de Transferencia 75

Tabla VII: Instrucciones Aritméticas 76

Tabla VIII: Instrucciones Lógicas 77

Tabla IX: Instrucciones de Puestas a Cero 77

Tabla X: Instrucciones de Salto 78

Tabla XI: Instrucciones para Manipulación de Bits 79

Tabla XII: Instrucciones Especiales 79

INTRODUCCION

Los microcontroladores están conquistando el mundo, están presentes en nuestro trabajo, en nuestra casa y en nuestra vida en general. Se pueden encontrar controlando el funcionamiento de los ratones y teclados de las computadoras, en los teléfonos, en los hornos microondas y los televisores de nuestro hogar. Pero la invasión acaba de comenzar y el nacimiento del siglo XXI será testigo de la conquista masiva de estos diminutos computadores, que gobernarán la mayor parte de los aparatos que fabriquemos y usemos los seres humanos.

El objetivo principal de la tesis es el diseño e implementación de una balanza electrónica en base a un microcontrolador, así como el estudio de los diferentes recursos del mismo, ventajas y desventajas existentes sobre los clásicos microprocesadores.

En la actualidad muchos dispositivos electrónicos incluyendo balanzas y dispositivos de control contienen numerosos chips tanto para su funcionamiento como para manejar periféricos como teclados, el despliegue visual, etc., esto hace que el espacio físico y su costo sean muy elevados.

Al utilizar un microcontrolador se reducirán costos, recursos y espacio ya que en este chip al contar con recursos técnicos incluidos en el mismo, hace más fácil su manejo y la implementación de distintos proyectos.

Se planteó la idea de construir una balanza electrónica para demostrar que con la ayuda de este pequeño pero eficiente chip se puede realizar circuitos de mediana y gran envergadura; esto me impulsó a buscar un microcontrolador que satisfaga nuestras necesidades de diseño, encontrándose que el más adecuado era el PIC 16F877 por poseer abundante

documentación, grandes recursos en cuanto a memoria, cantidad de puertos disponibles y sobre todo lo más importante un convertidor analógico digital incorporado que abarata el costo del diseño.

El proyecto consta de dos partes, una de software y otra de hardware. En el software se implementará rutinas en lenguaje ensamblador para la comunicación del microcontrolador con su mundo exterior como son el teclado, el despliegue visual de cristal líquido LCD y sobre todo el sensor de peso o celda de carga. El hardware vendrá complementado por las diferentes conexiones que tiene el microcontrolador con los periféricos que posee la balanza y el acondicionamiento de señal para el sensor.

El presente trabajo ha sido fruto de un concienzudo análisis tecnológico, el mismo que contribuirá para que otros continúen superando pequeños desajustes que posiblemente lo he pasado por alto y sobre todo para que nuevos proyectos que tengan un microcontrolador como base encuentren en este un dispositivo que satisfaga múltiples necesidades como recursos y facilidades de acceso.

CAPÍTULO 1

1. PARTICIÓN FUNCIONAL DE LA BALANZA

En la figura 1 se puede observar todas las señales de entrada y salida que posee la balanza así como el controlador de la misma que para nuestro caso será el PIC 16F877 que se detallará a continuación.

También se detallan los distintos periféricos tales como el despliegue de cristal líquido o LCD, el teclado, el MAX 232 que va ha servir para la comunicación serial con el computador y el circuito acondicionador de señal, que para tales fines se muestra como una caja negra debido a que es un circuito un tanto complejo.

El tipo de balanza que se escogió básicamente es la que se utilizaría en cualquier tienda o despensa. Es del tipo plataforma con una capacidad máxima de 10Kg o 22lb que más o menos son los pesos que se manejan en dichos lugares. También se escogió este tipo de balanza por que se pueden realizar operaciones de cálculo de precios y almacenamiento de dichas transacciones en su memoria interna.

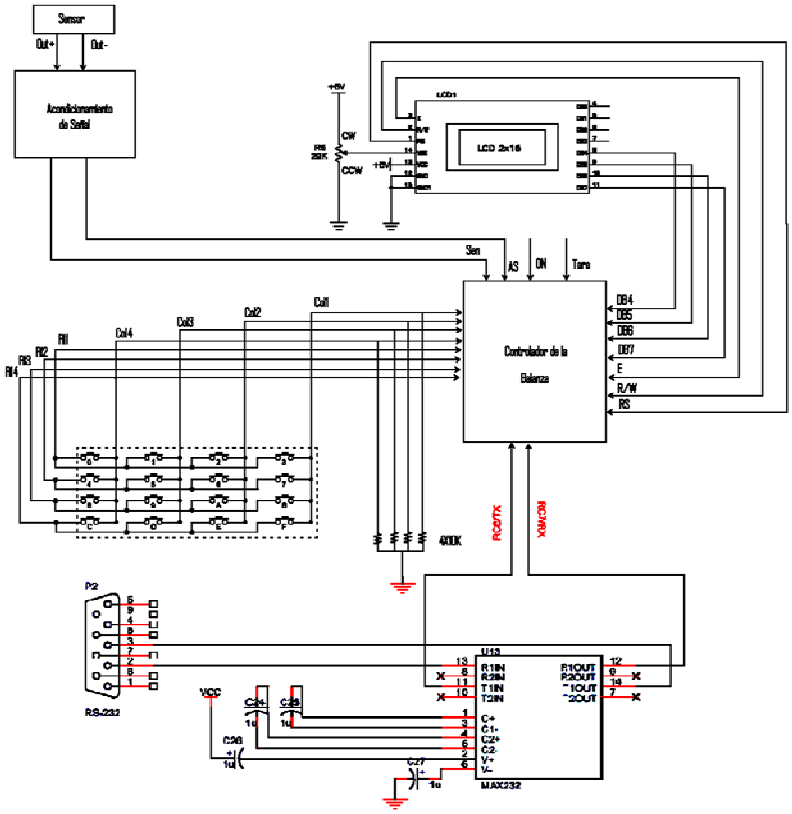


Figura 1: Partición funcional de la balanza

1.1. CONTROLADOR DE LA BALANZA

Para la elección del controlador del proyecto se decidió por un microcontrolador, las principales variables que se tomaron en cuenta para dicha elección son el tamaño de la memoria de programa y de datos, el número de canales para la conversión analógica, el tipo de memorias, número de interrupciones y puertos de entrada y salida de datos. Otra variable de suma importancia es el tipo de encapsulado.

Se decidió emplear un microcontrolador **PIC 16F877** de Microchip, el cual cuenta con características adecuadas para las distintas etapas del proyecto, pues es muy eficiente para la comunicación serial y debido a la velocidad con la que opera puede ser empleado para control. La cantidad de memoria es suficiente para todas las etapas del proyecto. La memoria es de tecnología FLASH, lo cual nos permite modificar parámetros de operación en forma rápida y eficiente.

A continuación se da una breve descripción del microcontrolador 16F877 con su arquitectura interna así como la memoria de datos y de programa, seguidamente se describe los puertos y algunos periféricos que posee.

Se puede conseguir distintos tipos de encapsulado para este dispositivo como QFP, PLCC o PDIP. Nosotros empleamos un tipo DIP dual line por la disposición del equipo para su programación.

1.1.1. REGISTROS ESPECIALES DEL AREA DE DATOS

Los Registros de Función Especial son registros usados por el CPU y los Módulos periféricos para controlar las operaciones deseadas de los dispositivos. Estos registros son implementados como una RAM estática.

Los registros de función especial pueden ser clasificados en dos grupos; Núcleo (CPU) y periféricos. Estas tablas de registros se detallan en los anexos.

1.1.2. ARQUITECTURA INTERNA Y ORGANIZACIÓN DE LA MEMORIA

Para soportar los nuevos recursos de la gama media Microchip tuvo que ampliar y mejorar la estructura de la CPU y de la memoria, aunque manteniendo la arquitectura Harvard que no es más que un CPU que maneja dos buses separados para controlar dos memorias. La primera contiene las instrucciones del programa y es llamada Memoria del Programa y la otra contiene los datos y se la nombra Memoria de Datos, además maneja el concepto RISC o Set de Instrucciones Reducido. También tuvo que añadir instrucciones y modificar algunas de las existentes.

En la **figura 2** se representa la organización de la memoria del programa. Hay tres bloques de memoria en cada PIC de la familia 16F87X que son: El PC, la Pila y la Memoria del Programa propiamente dicha.

Los PIC 16F87X poseen un contador de programa de 13 bits capaz de direccionar un espacio de 8K X 14 palabras de memoria de programa FLASH. El vector reset se encuentra ubicado en la posición 0000H y el vector de interrupción en la 0004H.

La diferencia entre los PIC 16F877/876 y 16F874/873 está dada en la capacidad de su memoria: los unos poseen una capacidad de 8K mientras que los otros solamente tiene una capacidad de 4K. Como podemos observar en la figura la Memoria de programa está paginada en bloques de 2K cada una. Los PIC 16F877/876 poseen cuatro páginas mientras que los otros dos únicamente poseen dos páginas cada uno esto hace la diferencia en la capacidad de memoria de dichos dispositivos.

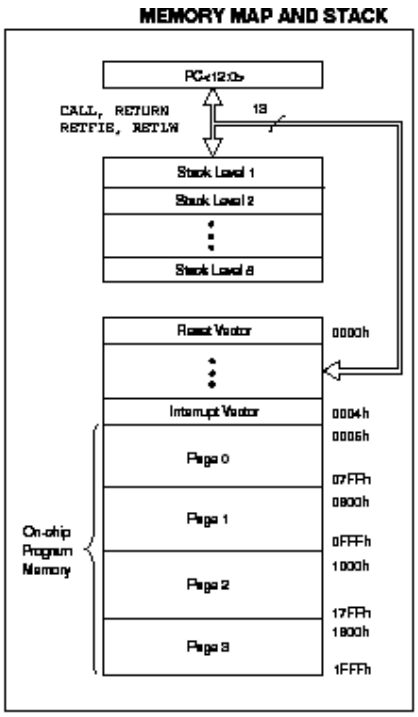


Figura 2: Mapa de Memoria y Pila

La arquitectura interna de estos dispositivos es muy similar a los de la gama media la diferencia radica en la cantidad de periféricos que poseen. Las líneas de E/S de los PIC de 40 pines corresponden a 33 patitas, siendo comunes las restantes en todos los modelos de estos tipos de PIC la alimentación, osciladores y el reset. A través del bus de datos se controlan los periféricos del microcontrolador que comunican con el mundo exterior por las patitas de los puertos.

Como se puede observar en la figura 3 tenemos el diagrama de bloques de un microcontrolador 16F877 que es igual al de 16F874 ambos poseen 40 pines en su diseño.

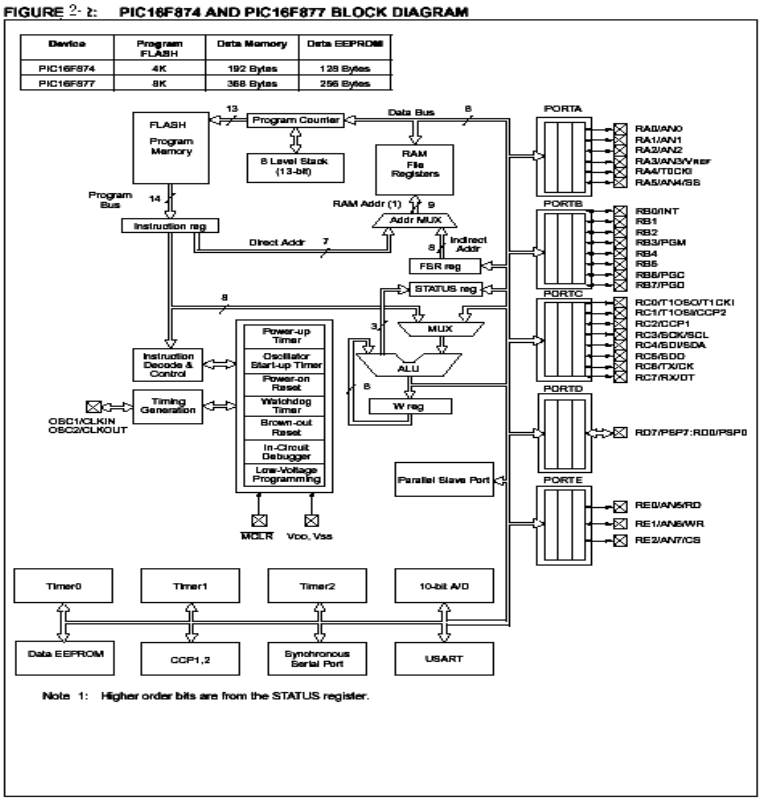


Figura 3: Arquitectura Interna de los PIC 16F87X

1.1.3. LA MEMORIA DE DATOS

La memoria interna de datos, también llamada archivo de registros (register file), esta dividida en dos grupos: los registros específicos, y los registros de propósito generales. Se organiza en cuatro bancos con 128 posiciones cada uno como máximo, la mayoría de los registros específicos se encuentran en las primeras posiciones de los bancos. Los registros de propósito general ocupan las posiciones comprendidas entre la 20H y la 7FH del banco 0 con 96 bytes, en los bancos del 1 al 3 tenemos 80 bytes para propósito general.

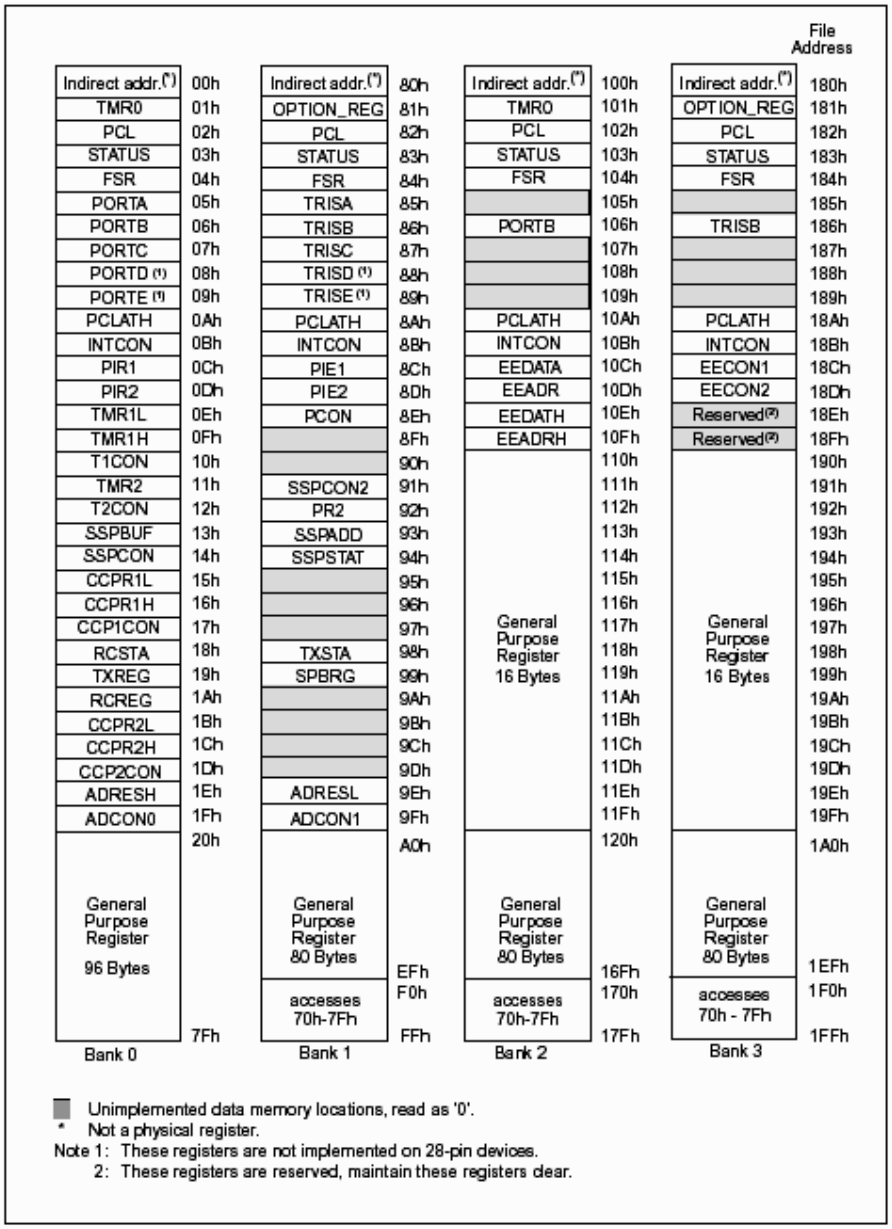


Figura 4: Memoria de Datos

1.1.4. EL PC

El Contador del Programa comúnmente denominado PC, es totalmente equivalente al de todos los microprocesadores, contiene la dirección de la próxima instrucción a ejecutar. Se incrementa automáticamente al ejecutar cada instrucción, de manera que la secuencia normal del programa es lineal una instrucción después de la otra. Al resetearse el microprocesador, todos los bits del PC toman valor 1, de manera que la dirección de arranque del programa es siempre la última posición de memoria de programa. En esta posición se deberá poner una instrucción de salto al punto donde verdaderamente se inicia el programa.

Todos los PIC 16F87X pueden direccional hasta 8K de memoria de programa organizándose en páginas de 2K cada uno, esto corresponden a 13 bits del PC, el byte de menos peso del PC se ubica en el registro PCL, este ocupa la posición 0x02 del banco 0 de la memoria de datos. Los 5 bits de más peso del PC se corresponden con los 5 bits de menos peso del registro PCLATH en la posición 0x0A del banco 0. Los bits de más peso del PC se pueden escribir a través del registro PCLATH.

Algunas instrucciones que llamaremos de control, cambian el contenido del PC alterando la secuencia lineal de ejecución. Dentro de estas instrucciones se encuentran el GOTO y el CALL que permiten cargar en forma directa un valor constante en el PC haciendo que el programa salte a cualquier posición de la memoria. Otras instrucciones de control son los SKIP o “salteos” condicionales, que producen un incremento adicional del PC si se cumple

una condición específica, haciendo que el programa salte, sin ejecutar, la instrucción siguiente.

1.1.4.1. LA PILA (STACK)

En los microcontroladores PIC la pila es una memoria interna de tamaño limitado, separada de las memorias de datos y de programa, inaccesible al programador, y organizada en forma de pila, que es utilizada solamente, y en forma automática, para guardar las direcciones de retorno de subrutinas e interrupciones. Cada posición es de 13 bits y permite guardar una copia completa del PC. Como toda memoria tipo pila, los datos son accedidos de manera tal que el primero que entra es el último que sale.

En los PICs 16F87X la pila es de 8 posiciones y en la gama 17 es de 16. Esto representa, en cierta medida, una limitación de estos microcontroladores, ya que no permite hacer uso intensivo del anidamiento de subrutinas. Esto de hecho representa una traba para el programador y además parece impedir o dificultar la programación estructurada, sin embargo es una buena solución de compromiso ya que estos microcontroladores están diseñados para aplicaciones de alta velocidad en tiempo real, en las que demoras adicionales que ocasionan un excesivo anidamiento de subrutinas es inaceptable.

Como ya se mencionó anteriormente, la pila y el puntero interno que lo direcciona, son invisibles para el programador, solo se los accede automáticamente para guardar o rescatar las direcciones del programa cuando se ejecutan las instrucciones de llamada o retorno de subrutinas, o cuando se produce una interrupción o se ejecuta una instrucción de retorno de ella.

1.1.4.2. DIRECCIONAMIENTO DE LOS DATOS

Como habíamos dicho la memoria de datos se organiza en 4 bancos de 128 posiciones de tamaño byte que contienen los Registros de Propósitos General y los Registros de Funciones Específicas. Los bits RP1 y RP0 del registro de estado son los encargados de seleccionar los bancos.

RP1	RP0	
= 00		Banco0
= 01	→	Banco1
= 10	→	Banco2
= 11	→	Banco3

Para direccionar la memoria de datos que contienen los registros de propósito específico y los de propósito general, existen dos modos de direccionamiento.

1ª. Direccionamiento Directo

Los siete bits de menos peso del código OP de la instrucción proporcionan la dirección de la posición de un banco. Los bits RP1 y RP0 del Registro de Estado <6:5>, seleccionan el banco.

2ª. Direccionamiento Indirecto

En este caso el operando de la instrucción hace referencia al registro INDF, que ocupa la posición 0 del área de datos. Se accede a la posición que apunta el registro FSR, que se halla situado en la posición cuatro del banco 0. Los 7 bits de menos peso de FSR seleccionan la posición y su bit de más peso junto con el bit IRP del Registro de Estado <7>, se selecciona el banco.

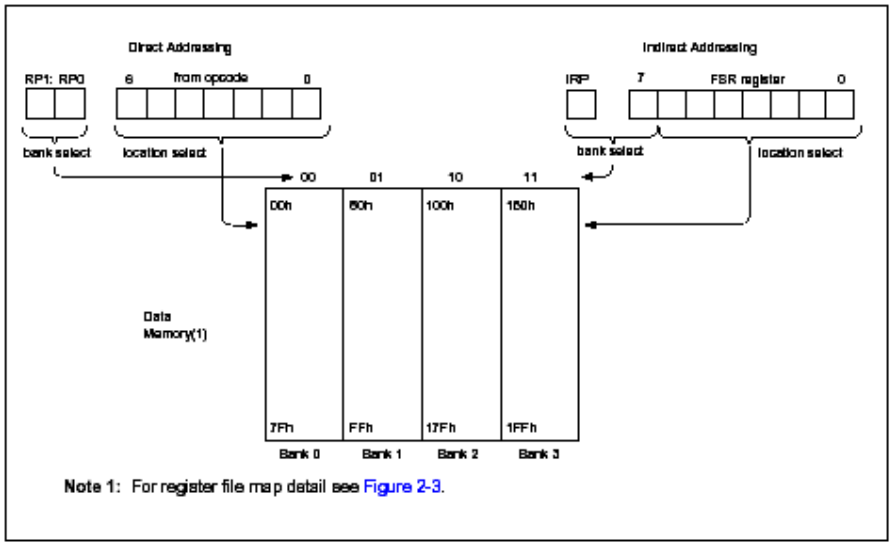


Figura 5: Modos de direccionamiento

1.1.5. REGISTROS DE FUNCIONES ESPECIALES

Todas las tablas se encuentran en los anexos lo que se muestra es una breve descripción de los bits de cada registro.

1.1.5.1. REGISTRO DE ESTADO

En el Registro de Estado se encuentra ubicado en la posición 03H del banco 0. En este se muestra los resultados que se han producido en la Unidad Lógica Aritmética (ALU) como son el bit del cero, el de acarreo total y parcial; así como también los bits de selección del banco y los de estado de reset.

1.1.5.2. REGISTRO DE OPCIONES

Este Registro se encuentra ubicado en la posición 81H del banco 1. Es un registro que se puede leer y escribir, contiene los bits de control para configurar el escalar del temporizador

TMR0 y del perro guardián WDT así como también los bits para habilitar las resistencias de pull-ups del Puerto B e interrupción externa INT

1.1.5.3. REGISTRO DE INTERRUPCIONES INTCON

Este se encuentra ubicado en la posición 0BH del banco 0 Puesto que los microcontroladores PIC de la gama media admiten interrupciones, requieren un registro encargado de su regulación.

La operatividad de sus bits se entenderá mejor cuando se explique la operatividad de las interrupciones. Algunos bits actúan como señalizadotes del estado de un módulo y otros como bits de permiso o autorización para que se pueda generar la interrupción.

1.1.5.4. OTROS REGISTROS

PIE1 Este registro contiene los bits de habilitación para interrupciones periféricas.

PIR1 Este registro contiene los bits de banderas para las interrupciones periféricas.

PIE2 Este registro contiene los bits de habilitación individual de los CCP2 interrupción periférica, el bus SSP y la interrupción para de escritura de la EEPROM.

PIR2 Este registro contiene los bits de banderas para el CCP2, el bus SSP y la escritura de la EEPROM.

PCON El registro de control de poder contiene dos bits con los que se puede diferenciar cuando el Reset se ha originado por un fallo en la alimentación (BO) o por la conexión de la misma (POR).

1.1.6. RECURSOS COMUNES DE LOS PICS

En la gama media existen muchos recursos que los comparten la mayoría de los modelos y que son conocidos muchos de ellos por la gama baja.

1.1.6.1. EL OSCILADOR PRINCIPAL

Este es un parámetro fundamental a la hora de establecer la velocidad de ejecución de las instrucciones y el consumo de energía. Un ciclo de instrucción se corresponde con 4 ciclos de reloj. Todas las instrucciones de PIC se ejecutan en un ciclo de instrucción salvo las de salto que necesitan dos. Así pues, funcionando a 10 MHz un ciclo de reloj son 100 ns y por lo tanto un ciclo de instrucción corresponde a 400 ns.

La frecuencia de trabajo viene marcada por el oscilador externo. Los PIC admiten cuatro tipos de osciladores:

- **Oscilador RC:** Oscilador de bajo coste formado por una resistencia y un condensador, cuyos valores determinan la frecuencia de oscilación. Proporcionan una estabilidad mediocre.
- **Oscilador HS:** Basado en un cristal de cuarzo, alcanza una velocidad entre 4 y 10 MHz.
- **Oscilador XT:** Oscilador de cristal o resonador para frecuencias entre 100 KHz y 4 MHz.

- **Oscilador LP:** Oscilador de bajo consumo, con cristal o resonador para frecuencias entre 35 y 200 KHz.

1.1.6.2. PERRO GUARDIAN WDT

El perro guardián (WDT) es un contador interno de 8 bits que reinicializa al micro (es decir provoca un Reset) cuando se desborda. Su control de tiempo es independiente del TMR0 y puede bloquearse para que no funcione programando un bit WDTE de la palabra de configuración.

Para evitar que se desborde hay que refrescarle, poniéndole a cero con las instrucciones *clrwdt y sleep.* Se debe analizar las instrucciones de la tarea y situar alguna de estas dos instrucciones en sitios estratégicos por los que pase el flujo del programa, antes de que transcurra el tiempo asignado al WDT. De esta forma si el programa se cuelga no se refresca el perro guardián y se produce la reinicialización del sistema.

1.1.6.3. TEMPORIZADOR TMR0

El dispositivo destinado a controlar los tiempos recibe el nombre de temporizador *(timer)* y básicamente consiste en un contador ascendente o descendente que determina el tiempo trascurrido entre el valor que se carga y el momento en el que se produce su desbordamiento o pasa por cero. El PIC 16F877 posee un temporizador/contador de 8 bits llamado TMR0 que puede actuar de dos maneras distintas.

- Como **contador** de los impulsos que se introducen por la pata RA4/TOCKI; en este caso su misión es contar el número de ciertos acontecimientos externos al micro. Al

llegar al valor FFH se desborda y pasa a 00H, activando un señalizador y/o produciendo una interrupción.

- Como **temporizador**, cuenta los impulsos de reloj del oscilador interno; en este caso se utiliza para determinar un tiempo fijo, ya que los impulsos tienen una duración conocida que equivale a un ciclo de reloj de instrucción (o lo que es lo mismo una cuarta parte del ciclo de reloj). Igual que en el caso anterior cuando se desborda pasa de FFH a 00H, poniendo a 1 un bit señalizador y/o provocando una interrupción.

El TMR0 se comporta como un registro de propósito especial (SFR) ubicado en la dirección 1 del banco 0 de la memoria de datos. Puede ser leído y escrito en cualquier momento al estar conectado con el bus de datos. Cuando funciona como temporizador hay que cargarle con el número de impulsos que se quieren contar pero expresados en complemento a 2. De esta forma, cuando transcurra el número de impulsos deseado el contador valdrá 00H activando de esta forma el señalizador TOIF y/o produciendo una interrupción.

Para programar el comportamiento del temporizador TMR0, el perro guardián y el divisor de frecuencias se utiliza algunos bits del registro OPTION y de la palabra de configuración.

1.1.6.4. REINICIALIZACIÓN O RESET

El PIC 16F877 tiene cinco causas por las que se provoca la reinicialización o reset del micro. Al producirse un reset se carga el valor 0000H (Vector de Reset) en el PC y los registros específicos toman un valor conocido. Las causas que motivan esta situación son las siguientes.

- Conexión de la alimentación
- Activación de la pata MCLR# en funcionamiento normal
- Activación de la pata MCLR# en estado de reposo
- Desbordamiento del perro guardián en funcionamiento normal
- Desbordamiento del perro guardián en el estado de reposo

1.1.6.5. MODO DE REPOSO SLEEP

Este modo de funcionamiento se caracteriza porque el micro consume muy poca energía, y por lo tanto está muy recomendado para aplicaciones en las cuales hay una larga espera, hasta que se produzca algún suceso asíncrono como la pulsación de una tecla (Ej: un cajero automático).

El consumo típico del PIC es de 2 mA aproximadamente, reduciéndose a 10 uA en estado de bajo consumo, lo que permite alimentarle con una pequeña pila durante dos años. Para entrar en estado de reposo hay que ejecutar la instrucción SLEEP, y el micro se queda como si estuviera congelado. El TMR0 se detiene y se elimina la entrada de impulsos externos por TOCKI. Como no hay impulsos el micro deja de ejecutar instrucciones hasta que se despierte y salga de ese estado.

Si el perro guardián continúa activo en el estado de reposo, al entrar en él se borra pero sigue funcionando. Los bits PD y T0 del registro de ESTADO toman los valores 0 y 1 respectivamente.

Para salir del estado de reposo existen tres alternativas:

- Activación externa de MCLR para provocar el reset
- Desbordamiento del perro guardián si quedó operativo en el estado de reposo
- Generación de una interrupción.

Cuando se despierta el PIC se pasa a ejecutar la siguiente instrucción a sleep. Los bits T0 y PD del registro de estado se emplean para conocer la causa del reset que despierta el sistema.

1.1.7. INTERRUPCIONES

Las interrupciones son desviaciones asíncronas del flujo del control del programa originada por diversos sucesos, que no están bajo el control de las instrucciones del programa. Estos sucesos pueden ser internos o externos al sistema y en diseños industriales son un recurso muy importante para atender acontecimientos físicos en tiempo real. Cuando se produce una interrupción se detiene la ejecución de un programa en curso, se salva la dirección actual de la pila y se carga el PC con la dirección del Vector de Interrupción que es una dirección reservada de la memoria de código.

En el PIC 16F877 el Vector de Interrupción está situado en la dirección 00004H de la memoria de programa, donde empieza la Rutina de Servicio de la Interrupción. En general, en dicha dirección se suele colocar una instrucción de salto incondicional GOTO que salta directamente a la dirección del programa en la que comienza la rutina de servicio de la interrupción.

La rutina de servicio comienza guardando en la memoria de datos los registros específicos que va a emplear y por lo tanto pueden alterar su contenido. Antes del retorno al programa

principal se recupera los valores guardados y se restaura el estado del procesador. Otra cuestión importante es averiguar cual de las posibles causas ha motivado la interrupción en este caso. Para ello se explora los señalizadores de las fuentes de interrupción que están almacenados en el registro INTCON.

El PIC 16F877 posee un bit GIE (*Global Interrupt Enable)* que cuando vale 0 prohíbe todas las interrupciones. Lo primero que debe hacer la rutina de servicio es poner a 0 este bit para evitar que se aniden interrupciones. En el retorno final de la interrupción (Instrucción RETFIE), GIE pasa a valer 1 automáticamente para permitir de nuevo las interrupciones. Sin embargo, los señalizadores de interrupción no se borran de forma automática y hay que desactivarlos por programa (de no hacerlo así en la siguiente interrupción tendríamos problemas para establecer su origen).

1.1.7.1. CAUSAS DE LA INTERRUPCIÓN

Son cuatro las posibles causas de interrupción:

- Activación de la pata RB0/INT
- Desbordamiento del temporizador TMR0
- Cambio de estado de una de las entradas RB4 – RB7 de la puerta B
- Finalización de la escritura en la EEPROM de datos

Estas solicitudes se atienden siempre que el bit GIE tenga valor 1. Cada una de las fuentes de interrupción anteriores dispone de un señalizador o *flag* que es un bit que se pone automáticamente a 1 cuando se produce la interrupción. Además cada interrupción tiene también un bit de permiso, que permite o prohíbe la solicitud de esa interrupción.

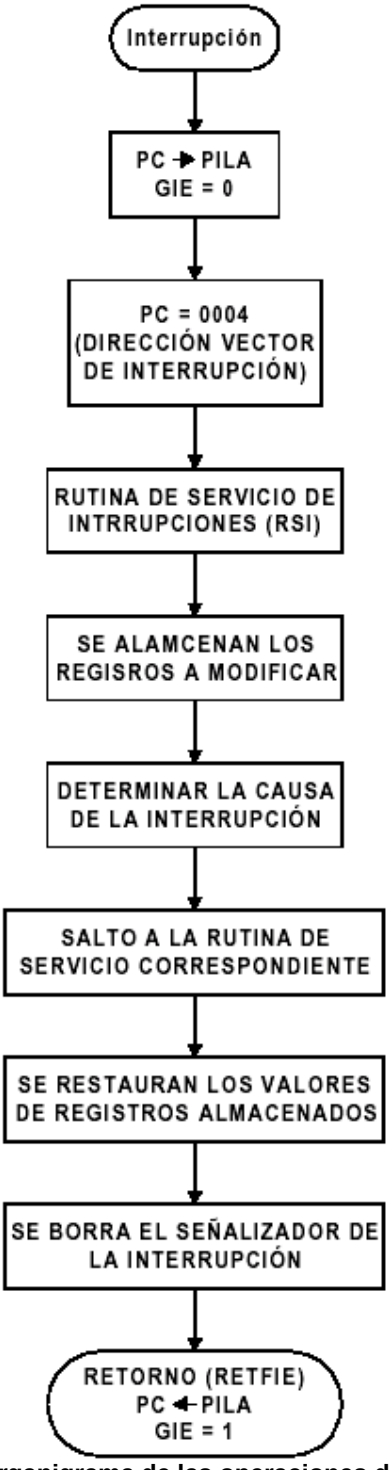


Figura 6: Organigrama de las operaciones de una interrupción

1.1.8. PUERTOS DE ENTRADA SALIDA

Como en este grupo de microcontroladores existen cuatro modelos (73, 74, 76, 77) se hace referencia al más completo, que es el PIC 16F877 de 40 patitas, que contiene más puertos y más periféricos, los cuales se citan a continuación:

- Recursos comunes (TMR0, Perro Guardián, Modo de reposo, Reset de conexión de alimentación, interrupciones, etc)
- Convertidor A/D de 10 bits con 8 canales de entrada
- 2 Módulos de Captura/Comparación/PWM(CCP)
- Puerto Síncrono Serie (SSP)
- Interfaz de Comunicación Serie (SCI)
- Puerto Paralelo Esclavo

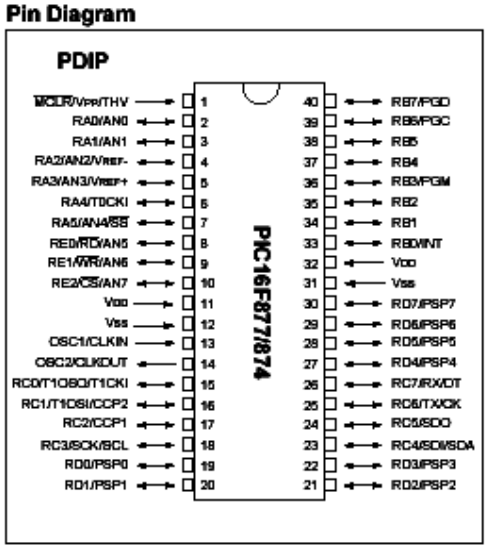


Figura 7: Diagrama de pines del PIC 16F877

1.1.8.1. PUERTO A

Consta de seis patitas o líneas (RA0 a RA5). Todas, menos RA4, pueden actuar como E/S digitales o como canales de entrada para el convertidor A/D. La patita RA4, además de entrada salida digital puede funcionar como entrada de reloj externo para el TMR0.

1.1.8.2. PUERTO B

Las cuatro líneas de menos peso del Puerto B (RB<3:0>) actúan como E/S digitales, según la programación del registro TRISB. Además pueden disponer de una carga pull-up interna si se programa la línea como entrada si el bit <7> (RBPO) del registro OPTION vale 0.

Las líneas RB<7:4> funcionan como las anteriores, pero además pueden provocar una interrupción si se programa como entradas y se produce el cambio de nivel lógico en alguna de ellas. En tal caso se activa el bit <0> (RBIF) de INTCON o al hacer una nueva lectura del Puerto B.

1.1.8.3. PUERTO C

Es un puerto bidireccional de 8 bits disponibles en los modelos 73, 74 y 76. Cada patita actúa como E/S digital, según la programación de TRISC. Además, también puede actuar como entrada o salida de diversos periféricos internos.

A continuación se describen la nomenclatura de cada patita y sus funciones:

RC0/OSO/T1CKI: E/S digital. Salida para la conexión del cristal de reglamentación del oscilador externo para el TMR1. Entrada de reloj para el TMR1.

RC1/OSI/CCP2: E/S digital. Entrada para la conexión de cristal del oscilador externo del TMR1. E/S del módulo 2 para la Captura/Comparación/PWM

RC2/CCP1: E/S digital. E/S del módulo 1 de Captura/Comparación/PWM

RC3/SCK/SCL: E/S digital. Reloj síncrono para los modos SPI e I^2C del puerto serie

RC4/SDI/SDA: E/S digital. Entrada de datos serie en el modo SPI. E/S serie en el modo I^2C.

RC5/SDO: E/S digital. Salida de datos serie en el modo SPI

RC6/Tx/CK: E/S digital. Línea de transmisión asíncrona del canal serie. Reloj síncrono del canal serie

RC7/Rx/DT: E/S digital. Línea de recepción asíncrona del canal serie. Línea de datos del canal serie síncrono

1.1.8.4. PUERTO D

Solo esta dispone en los PIC 74 y 77 y consta de 8 líneas bidireccionales. Cada patita puede configurarse como entrada y salida digital según la programación del TRIS D.

También puede funcionar como Puerto Paralelo Esclavo, para soportar la interacción directa con el bus de datos de 8 bits de otro microprocesador. Para funcionar en este modo hay que poner a 1 el bit <4> (PSMODE) de TRISE. En tal caso, las líneas RE <2:0> del Puerto E pasan a soportar las líneas de control CS, WR y RD, entre el Puerto D y el bus del microprocesador. Cada vez que el microprocesador realiza un ciclo de lectura o escritura sobre el Puerto D el bit <7> (PSPIF) del registro PIR1 se pone a 1.

1.1.8.5. PUERTO E

Las tres patitas pueden funcionar como E/S digitales, según la programación de los tres bits de menos peso del registro TRISE. También puede actuar como señales de control (RD, WR y CS) para el flujo de datos entre un microprocesador y el Puerto D, cuando está programada en el modo Esclavo. Deben programarse como entradas. Finalmente, también pueden realizar estas tres patitas la función de canales de entradas analógicas para el Conversor A/D, según la programación del registro ADCON1.

En el registro TRISE, sirve para configurar las líneas de entrada salida digitales del Puerto E, o bien, para actuar como Registro de Estado cuando el Puerto D funciona como Puerto Paralelo Esclavo.

1.1.9. PUERTO SERIE SINCRONO (SSP)

Se trata de un periférico diseñado para soportar una interfaz serie síncrono que resulta muy eficiente para la comunicación del microcontrolador con dispositivos tales como displays, EEPROM, ADC, etc.

Tiene dos modos de trabajo:

1º. Interfaz Serie de Periféricos (SPI)

2º. Interfaz Inter-Circuitos (I^2C)

1.1.9.1. MODO SPI

Sirve para conectar varios microcontroladores de la misma o diferente familia, bajo el formato “maestro-esclavo”, siempre que dispongan de un interfaz compatible

En este modo se pueden emplear tres o cuatro señales de control: Salida de Datos (SDO), Entrada de Datos (SDI), Reloj (SCK) y Selección de Esclavo (SS). Dichas señales se corresponden con las patitas RC5, RC4, RC3 y RA5 respectivamente. Cada una de las señales debe programarse como entrada o salida según su condición, utilizando los bits de los registros TRIS. Cualquier función del modo SPI queda anulada poniendo con el valor opuesto a su condición el bit correspondiente de TRIS. Por ejemplo, si solo se quiere recibir datos, se programa la patita que soporta a SDO como entrada y así se anula su función.

Con el registro de control SSPCON se eligen las diferentes opciones de trabajo:
Modo Master (SCK es salida), Modo Esclavo (SCK es entrada), tipo de flanco de reloj, velocidad de SCK en Modo Master, etc.

Cuando se recibe un dato útil, este se introduce en serie en SSPSR y pasa a SSPBUF en paralelo. El dato a transmitirse deposita en SSPBUF y de aquí pasa a SSPSR. Se puede recibir y transmitir datos simultáneamente. SSPSR es un registro de desplazamiento que funciona serie/paralelo/serie.

Cuando se acaba de transmitir o recibir un dato completo se activa el bit BF (Buffer Lleno) del registro SSPSTAT. También lo hace el señalizador SSPIF y si el bit de permiso esta activado se genera una interrupción.

Cuando se recibe un dato durante una transmisión se ignora y se activa el bit WCOL que indica que se ha producido una “colisión”.

En el caso que se reciba un nuevo dato en SSPSBUF sin haber leído el anterior se genera un error de desbordamiento.

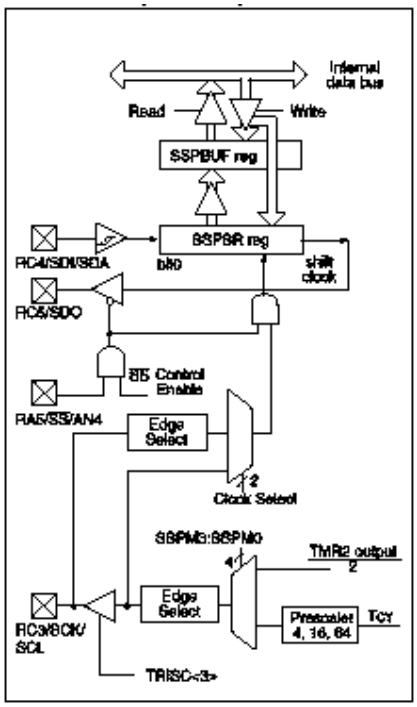


Figura 8: Modo SPI

1.1.9.2. MODO I^2C

Este tipo de interfaz serie ha sido desarrollado por Phillips y utiliza solo dos hilos trenzados y una masa común para la interconexión de los diversos dispositivos, que han tenido que ser diseñados para soportar este protocolo, asegurando una gran fiabilidad en la comunicación que llega a tolerar una velocidad máxima de 400 Kbps. Es capaz de interconectar hasta 128 dispositivos situados a gran distancia, por lo que resulta muy usado en edificios inteligentes, control de distribuciones de electricidad, agua y gas, piscifactorías, etc.

El master es el que inicia y termina la transferencia general y provee de la señal de reloj. El esclavo ("slave") es el dispositivo direccionado por el master, mediante 7 bits, lo que limita el número de componentes a 128.

El inicio de la transmisión se determina con el bit de inicio (S) y el final con otro bit de stop (P). El bus serie de 2 hilos utiliza uno de ellos para transferir datos (SDA) y el otro para la señal de reloj (SCL).

En el protocolo I^2C cada dispositivo tiene asignada una dirección de 7 o de 10 bits que envía el master cuando comienza la transferencia con uno de ellos. Tras la dirección se añade el bit de recepción/transmisión o lectura/escritura (R/W). Los datos se transmiten con longitud de un byte y al finalizar cada uno se inserta un bit de reconocimiento ACK. Debe existir un modulo de arbitraje que gestione que solo hay un maestro en cada instante sobre el bus compartido.

SSPBUF es el registro donde se almacena el byte a transmitir o el que se recibe. SSPSR es el registro desplazamiento serie de la línea E/S. SSPADD es el registro de direcciones que

identifica el dispositivo (modo esclavo) o que lo direcciona (modo master). El registro de control SSPCON selecciona las diversas funciones del modo I^2C.

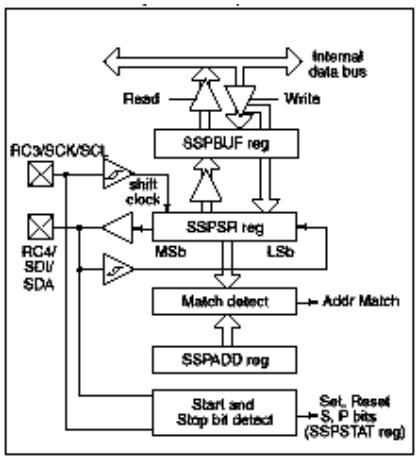


Figura 9: Modo I^2C

El registro SSPCON se muestra en los anexos este sirve para programar al puerto serie.

1.1.10. INTERFASE DE COMUNICACIONES SERIE (SCI)

Esta interfaz proporciona las mismas prestaciones que un UART programable: Se lo puede programar de dos modos diferentes:

Asíncrono (full-duplex)

La comunicación es bidireccional. La patita RC6/Tx/CK actúa como línea de transmisión y la RC7/Rx/DT como línea de recepción. Cada dato lleva un bit de inicio y otro de stop.

Síncrono (semiduplex)

Comunicación unidireccional. Una sola línea para los datos que se implementan sobre la patita RC7/Rx/DT. En el modo master la señal de reloj sale por la patita RC6/Tx/CK. En el modo esclavo (“slave”) entra por ella

En ambos modo los datos pueden ser de 8 o 9 bits, pudiendo emplear el noveno como bit de paridad, transmitiéndose o recibiéndose por el bit <0> de RXSTA y/o RCSTA.

El registro específico TXSTA actúa como registro de estado y control del transmisor y el RCSTA hace lo mismo para el receptor.

Los baudios se establecen por el valor cargado en el registro SPBRG y el bit BRGH del registro TXSTA, con el que se puede elegir la velocidad alta (1) o baja (0) en el modo asíncrono.

BAUDIOS = F_{OSC} / (n(x + 1))

n = 4 en el modo síncrono
n = 16 en el modo asíncrono de alta velocidad
n = 64 en el modo asíncrono de baja velocidad
x = valor cargado en el registro SPBRG

Siendo x = (F_{OSC} / Baudios) / (n - 1)

Mediante la programación de los bits del registro TXSTA y RCSTA se configura el modo de trabajo. Así, SPEN configura RC7/Rx y RC6/Tx como líneas de comunicación serie. El transmisor se activa con el bit TxEN. El dato a transmitir se carga en TxREG y luego pasa al registro transmisor TSR, cuando se haya transmitido el bit de stop del dato anterior.

Entonces se activa el señalizador TxIF y si el bit de permiso esta activado se produce una interrupción.

Activando Tx8/9 se inserta el noveno bit almacenado en el bit <0> (TxD8) de TXSTA. El bit TRMT indica si el transmisor esta vacío o no. El dato se recibe por RSR y cuando se completa se pasa al registro RCREG para su posterior lectura, activándose el señalizador RCIF y si acaso la interrupción.

Si se activa el bit RC8/9 del RCSTA el noveno bit se deposita en el bit <0> (RCD8) del RCSTA. Los bits OERR y FERR indican error de desbordamiento y de trama, respectivamente

En modo síncrono el SCI trabaja en half duplex, no pudiendo emitir y transmitir a la vez. La señal de reloj la envía el transmisor (maestro) conjuntamente con los datos. Los principios y el funcionamiento de la emisión y la recepción síncronas son similares al modo asíncrono y únicamente hay que seleccionar esta forma de trabajo cargando adecuadamente los registros TXSTA y RCSTA.

Para la transmisión serial de la balanza con el computador se trabajo en modo asíncrono a una velocidad de 9600 baudios sin bit de paridad y a alta velocidad.

1.1.11. CONVERTIDOR ANALÓGICO DIGITAL

El módulo convertidor A/D tiene 8 canales. La entrada analógica carga a un capacitor de muestreo, que es la entrada del convertidor. El convertidor genera un resultado digital de su nivel analógico vía aproximaciones sucesivas. El resultado es un número de 10 bits. Se

cuenta con una referencia alta y baja de voltaje de entrada que se puede seleccionar con una combinación de Vdd, Vss, RA2 o RA3.

El convertidor puede operar aún en modo “Sleep”. El reloj debe ser derivado del oscilador interno RC del A/D.

Registros:

(ADRESH): A/D Result High Register
(ADRESL): A/D Result Low Register
(ADCON0): A/D Control Register 0
(ADCON1): A/D Control Register 1

El registro de control 0 controla la operación del módulo A/D. El registro de control 1 configura las funciones de los pines del puerto. Se pueden configurar como entradas analógicas o como entradas/salidas digitales.

Los registros de resultados altos y bajos, contienen el resultado de la conversión. Cuando esta se ha completado, el resultado es cargado en dichos registros, el bit GO/DONE es limpiado y la bandera de interrupción ADIF es puesta a 1.

Pasos para la conversión:

1.- Configuración del módulo A/D

- Configuración de los pines analógico, las referencias de voltaje y las salidas entradas digitales (ADCON1)
- Selección de un canal de entrada A/D (ADCON0)

- Selección de un reloj de conversión (ADCON0)
- Activar el módulo (ADCON0)

2.- Configurar las interrupciones A/D si se desea

- Limpiar el bit ADIF
- Poner en uno el bit ADIE
- Poner en uno el bit GIE

3.- Esperar el tiempo requerido

4.- Empezar la conversión

- Poner en uno la bandera GO/DONE (ADCON0)

5.- Esperar que la conversión esté completa

- Verificando hasta que la bandera GO/DONE sea limpiada
- Esperando a la interrupción A/D

6.- Leer el resultado en ADRESH; ADRESL. Limpiar ADIF si se requiere

7.- Para otra conversión se repiten los pasos 1 y 2. TAD es la conversión por bit. Se requiere 2TAD antes de iniciar la siguiente captura de datos.

La impedancia de la fuente recomendada es de 10 k, para que el capacitor se cargue correctamente.

Para el tiempo de conversión se requiere de por lo menos 12 TAD para una conversión de 10 bits.

Opciones para el reloj:

2 TOSC

8 TOSC

32 TOSC

RC interno

El reloj de conversión debe seleccionarse para asegurar un TAD de 1.6 microsegundos mínimos.

Con 32 TOSC el dispositivo alcanza una frecuencia máxima de 20 MHz

Si durante la conversión limpiamos el bit GO/DONE se interrumpe la conversión.(Manual de Microcontroladores Universidad de Guadalajara; Introducción a los Microcontroladores José Adolfo González; Microcontroladores PIC diseño de Aplicaciones Angulo)

1.2. EL SENSOR

El sensor que se utilizó en la construcción de la balanza electrónica es denominado celda de carga, la misma que como elemento transductor posee Galgas Extensiométricas en formación de un Puente de Wheatstone. Los mismos que se detallará a continución.

1.2.1. LAS GALGAS EXTENSIOMÉTRICAS

Las células de carga y las galgas extensiométricas son elementos metálicos que cuando se someten a un esfuerzo sufren una deformación del material, y por lo tanto una variación de su resistencia interna.

Las galgas extensiométricos es un dispositivo comúnmente usado en pruebas y mediciones mecánicas. La más usada es la galga extensiométrica de resistencia, que consiste en una matriz de bobinas o cable muy fino, el cual varía su resistencia linealmente dependiendo de la carga aplicada al dispositivo. Cuando usted usa una galga extensiométrica, usted pega la galga directamente al dispositivo bajo prueba, aplica fuerza y mide la carga detectando los cambios en resistencia. Las galgas extensiométricas también son usadas en sensores que detectan fuerza, aceleración, presión y vibración.

1.2.2.CONTROL DE RUIDO EN LAS GALGAS

Prácticamente cualquier dispositivo eléctrico que genera, consume o transmite energía es una fuente potencial de ruido para circuitos con galgas. Las siguientes es una lista de los más comunes:

- Líneas eléctrica
- Motores
- Transformadores
- Relevadores
- Lámparas fluorescentes
- Transmisores de radio
- Tormentas eléctricas
- Cautines eléctricos

El ruido se puede categorizar en dos tipos: ruido electrostático y ruido electromagnético

1.2.2.1. TECNICAS DE REDUCCION DE RUIDO

La protección más simple y efectiva contra el ruido electrostático es una red conductora (blindaje) que rodea a los cables de la señal de interés, también llamada “Jaula de Faraday”.

Si no se provee de una trayectoria de baja resistividad para drenar las cargas acumuladas, se crea un acoplamiento capacitivo a la señal con el blindaje.

La técnica más efectiva para reducir el ruido electromagnético no es tratando de proteger los conductores sino tratar de que ambas entradas del amplificador lleven el mismo nivel de ruido.

El amplificador debe mostrar buenas características de CMRR y se debe de cuidar el manejo de los cables. Se recomienda trenzar los cables de señal y en casos graves, trenzar los cables de las líneas eléctricas

1.2.3. EL PUENTE DE WHEATSTONE

Ya que las mediciones de carga requieren detectar cambios muy pequeños de resistencia, el circuito de Puente de Wheatstone se usa predominantemente. El circuito de Puente de Wheatstone consiste de cuatro elementos resistivos con excitación de voltaje aplicado en las puntas del puente. Las galgas extensiométricas pueden ocupar uno, dos o cuatro brazos del puente, completando con resistencias fijas los brazos que sobran.

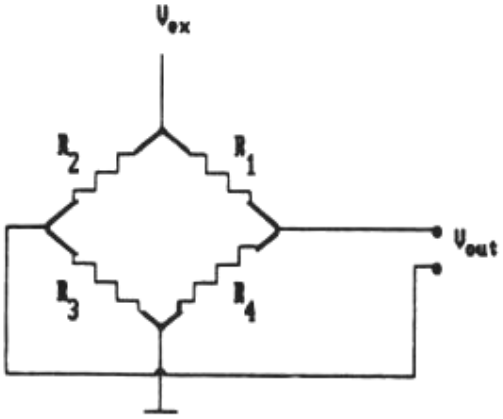


FIGURA 5.- PUENTE SIMPLE. LA SALIDA SERA CERO SI SE CUMPLE $R_1/R_4 = R_2/R_3$

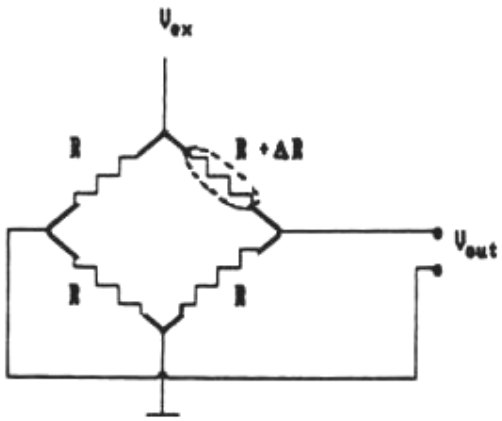


FIGURA 6a.- PUENTE CON UN ELEMENTO VARIABLE. LA SENSIBILIDAD ES 1/4.

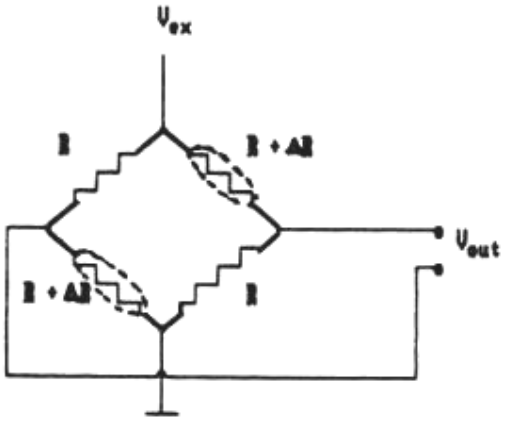


FIGURA 6b.- DOS ELEMENTOS VARIABLES CUYOS CAMBIOS LO HACEN EN LA MISMA DIRECCION. LA SENSIBILIDAD ES 1/2

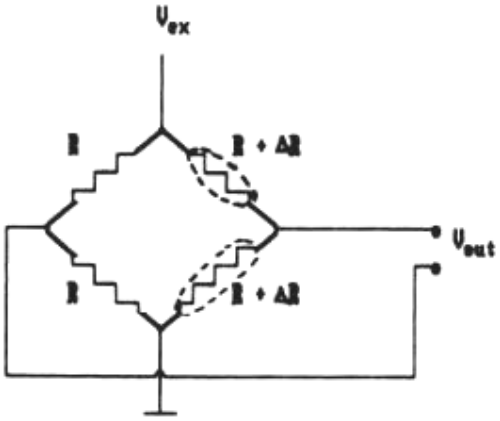


FIGURA 6c.- DOS ELEMENTOS VARIABLES CUYOS CAMBIOS LO HACEN EN DIRECCION OPUESTA. LA SENSIBILIDAD ES 1/2.

Figura 10: Puente de Wheatstone

El valor de salida es la diferencia entre los dos divisores de tensión;

$$Vout = \frac{R1}{R1+R4}Vext - \frac{R2}{R2+R3}Vext$$

V_{OUT} será nulo si R1/R4 = R2/R3.

La configuración de las resistencias del transductor determina la sensibilidad total frente a los pequeños cambios en la resistencia del transductor ΔR, la que está definida por la relación V_{out}.*R/V_{ex}.* ΔR.

El puente más simple consiste de un único elemento variable, cuya resistencia es R + ΔR, y tres ramas cuyas resistencias son R La razón entre V_{out}/V_{ex}. es ΔR/ (4R+2 * ΔR) Si ΔR«R, la sensibilidad es 1/4; un rango de ±10 Ω en un transductor de 1 KΩ (1% de plena escala) producirán un cambio de ±0.025 Ven a la salida cuando el puente sea excitado por 10 V (0.25%). La importancia del puente, sin embargo, es que la salida es directamente proporcional a la diferencia entre resistencias en lugar de pequeños cambios en una gran señal.

La configuración de dos elementos variables, los elementos variables cambian de forma idéntica y se colocan en ramas opuestas. La relación V_{out}/V_{ex}. es Δ R/(2R + ΔR). Si ΔR«2R, la sensibilidad es casi 1/2.

En algunos casos, los transductores resistivos se pueden instalar de manera que la resistencia de uno disminuya en el mismo grado que la del otro sube; las resistencias se ponen en ramas adyacentes. La sensibilidad, cuando ΔR<<2R, se reduce a casi 1/2.

Aunque la configuración anterior, no es más sensible que las otras, tiene una ventaja adicional. Si las dos ramas variables cambian en la misma dirección (debido a la temperatura, por ejemplo, cuando se mide la presión), ese cambio se cancela.

Cuando se colocan elementos activos en todas las ramas, con un par respondiendo en dirección opuesta a la del otro, la sensibilidad es exactamente 1. La señal de salida relativa a la tensión de excitación es igual al cambio relativo de voltaje. La salida del puente requiere un amplificador con entrada diferencial, debido a que ambos extremos de la salida diagonal serán diferentes de tierra. Las tensiones de salida generalmente también son muy pequeñas, por lo tanto se deben usar amplificadores de instrumentación con entradas diferenciales. Cuando la ganancia es grande se pueden hacer medidas muy sensibles con estas configuraciones. .(www.measurementsgroup.com)

1.3. ACONDICIONAMIENTO DE SEÑAL

En el acondicionamiento de señal podemos encontrar estas etapas, aunque no todas están siempre presentes:

- Amplificación
- Excitación
- Filtrado
- Multiplexado
- Aislamiento
- Linealización

En mi caso como el sensor es lineal la etapa de linealización no la tuve que realizar. La etapa de multiplexado se la usa para la conmutación de diferentes entradas del convertidor, de modo que con un solo convertidor podemos medir los datos de diferentes canales de entrada. En el caso de la balanza esto no se realizó ya que al tener solo una entrada analógica no se tuvo que multiplexar la señal. La señal entró directamente al canal 1 del convertidor incorporado en el microcontrolador.

Vamos a describir algunas etapas del acondicionamiento de señal.(Tutorial Acondicionamiento de Señales National Instruments)

1.3.1. AMPLIFICACIÓN

Es el tipo más común de acondicionamiento. Para conseguir la mayor precisión posible la señal de entrada debe ser amplificada de modo que su máximo nivel coincida con la máxima tensión que el convertidor pueda leer.

En este caso como el sensor de peso tiene una señal diferencial muy pequeña con una salida máxima de 2 mV/V esto equivale que con una alimentación de 10 V al máximo peso tendremos una salida de 20 mV tuve la necesidad de diseñar un Amplificador de Instrumentación.

1.3.1.1. AMPLIFICADORES DE INSTRUMENTACIÓN

Existen equipos en la industria, en equipos de electromedicina, y en equipos en otras muchas aplicaciones, la necesidad de medir señales muy pequeñas del orden de los micro voltios o pocos mili voltios en la presencia de comparativamente grandes señales de ruido provenientes de distintas fuentes. Para realizar las mediciones estos deberán utilizar en su entrada Amplificadores de Instrumentación con una adecuada Relación Rechazo de Modo Común (CMRR).

El esquema clásico de un amplificador de este tipo se muestra en la figura 11 para una entrada con un Puente de Wheatstone que es el caso del que nos ocupa para la celda de carga.

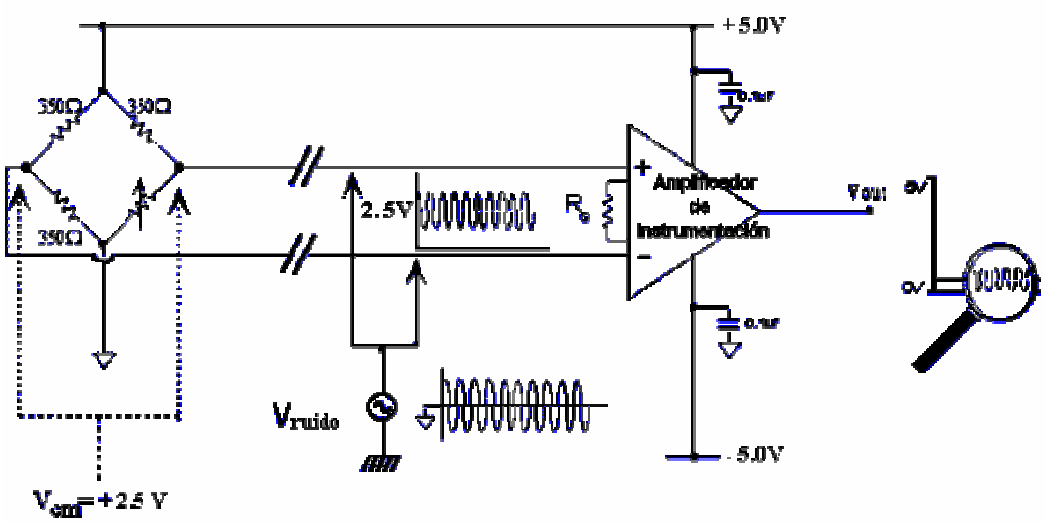


Figura 11: Esquema básico de medición

Al amplificador de instrumentación ingresan dos señales de modo común: una de c.c. de +2.5V provenientes del puente de resistencias y la otra de c.a. Vruido inducida sobre los cables de entrada al amplificador.

Rechazo de Modo Común.- Los Amplificadores de Instrumentación amplifican la diferencia entre dos señales. Estas señales diferenciales en la práctica provienen de sensores como son termocuplas, fotosensores, puentes de medición resistivos, etc. En la figura de arriba se ve que de un puente resistivo, en estado de equilibrio sin señal, en la mitad de las ramas del puente existe una señal de 2.5 V respecto a masa. Esta señal de corriente continua es común a ambas entradas por lo cual es llamada Voltaje de Modo Común de la señal diferencial. Se pueden ver que estas señales no contienen información útil en lo que se quiere medir y como el amplificador amplificará la diferencia de ambas, al ser iguales, se restan y a la salida el resultado será cero o sea idealmente no están contribuyendo a la

información de salida. También se ve que se inducen señales de corriente alterna en ambas entradas a la vez y que serán rechazadas como en el caso de la continua. Pero al producirse un desbalance del equilibrio del puente por la variación de una de sus resistencias se producirá una señal que será aplicada entre ambas entradas y será amplificada. Por lo expuesto, es que se justifica la utilización de amplificadores de instrumentación para rechazar señales que entran en modo común, o sea en las dos entradas se presenta la misma señal.

En la práctica, las señales de modo común nunca serán rechazadas completamente, de manera que una pequeña parte de la señal indeseada contribuirá a la salida.

Para cuantificar la calidad del Amplificador de Instrumentación, se especifica la llamada Relación de Rechazo de Modo Común (CMRR) que matemáticamente se expresa como:

$$CMRR[db] = 20 \times \log\left(\frac{Ad}{Acm}\right)$$

Siendo:

- Ad= Amplificación Diferencial
- Ad= Vout/Vin diferencial
- Acm= Amplificación Modo Común
- Vcm= Voltaje de modo común en la entrada
- Acm= Vout/Vcm
- Vout= Voltaje de salida

De la última fórmula podemos obtener la Vout como:

$$Vout = \frac{Ad}{\log^{-1}\left(\frac{CMRR}{20}\right)} \times Vcm$$

De las hojas de datos de los Amplificadores de Instrumentación podemos obtener por ejemplo:

- CMRR= 100 db
- Ad= 10
- De la figura, Vcm de modo común es de 2.5 Volt.

De donde:

Vout= 250 uV para el caso de la figura anterior.(Eamon Nash A Practical Review of Common Mode and In-Amps)

Como se ve en la figura 11, y como se dijo, se presentan a las entradas diferenciales, señales de corriente continua y de corriente alterna y al no ser infinito el CMRR, una cierta cantidad de ambas estarán presentes a la salida, además de la señal diferencial deseada. La componente indeseada de corriente continua puede considerarse como un offset y es sencillo de ajustarlo externamente. La componente indeseada de corriente alterna es más complicada de disminuir a la entrada, y se hace principalmente utilizando filtros de corriente alterna colocados a la entrada, diminuyendo el ancho de banda de utilización del amplificador.

En nuestro caso el filtro de corriente alterna no es más que una celda R-C con resistencias iguales y de valor de 1KΩ y con capacitores cerámicos de 15pF con una precisión del 5% para atenuar diferentes señales de ruido de corriente alterna como focos fluorescentes,

motores o cualquier dispositivo que genere este tipo de interferencia. El capacitor de 0.047μF sirve únicamente para fijar la señal que está siendo producto de análisis.

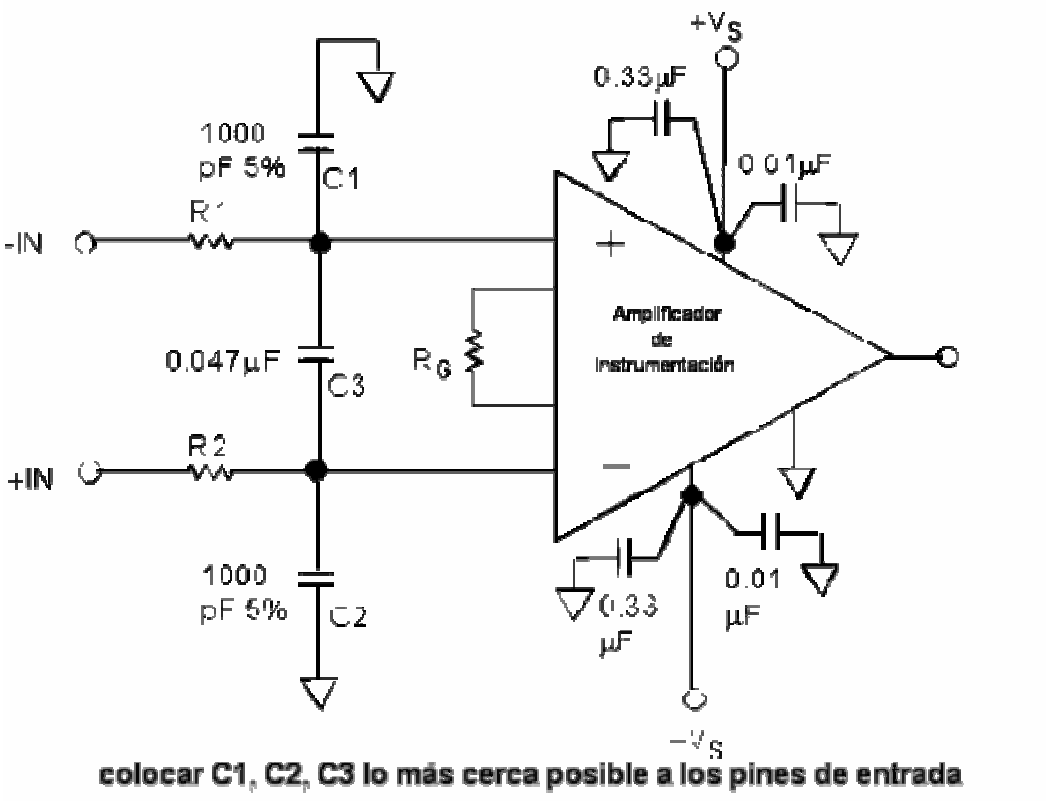


Figura 12: Filtro de corriente alterna a la entrada

1.3.1.2. CONFIGURACIÓN CIRCUITAL DEL AMPLIFICADOR DE INSTRUMENTACIÓN

El circuito esquemático que se muestra en la figura 13 es la configuración que se implementó en la tarjeta que acondiciona la señal de entrada al PIC. Esta está compuesta por tres amplificadores operacionales LM 324, resistencias de 10KΩ (R1, R2, R3 y R4) y 20KΩ (R5 y R6) del 1% de precisión y un potenciómetro de 400Ω que es la que da la ganancia al circuito que es este caso es Rg.

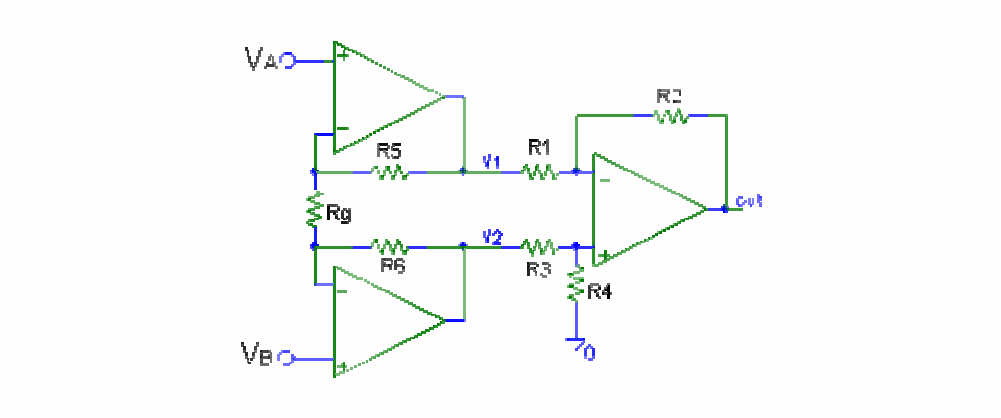


Figura 13: Amplificador de Instrumentación

En este amplificador se acostumbra a hacer R5=R6=R; R1=R3 y R2=R4 y como ya vimos la amplificación diferencial será:

$$Ad = \left(\frac{Vo}{Vb - Va}\right) = \left(\frac{R2}{R1}\right) \times \left(1 + \frac{2 \times R}{Rg}\right)$$

Considerando Va = Vb = Vcm y como los amplificadores de entrada están en configuración simétrica, la misma tensión aparece en V1 y V2, de manera que de las ecuaciones vistas anteriormente en amplificadores de diferencia surge que:

$$\frac{Vout}{Vcm} = \frac{R4}{R3 + R4} \times \frac{R1 + R2}{R1} - \frac{R2}{R1}$$

Y el CMRR será

$$CMRR = 20 \times \log\left[\frac{Ad}{\left(\frac{R4}{R3+R4} \times \frac{R1+R2}{R1} - \frac{R2}{R1}\right)}\right]$$

Nuevamente el CMRR depende de la Ad y del cuidado en seleccionar los valores de las resistencias, ya sea para que sean lo más iguales posibles o sus relaciones de unas a otras sean lo más exactas posibles.

1.3.2. EXCITACIÓN

La etapa de acondicionamiento de señal a veces hay que generar algún tipo de excitación para algunos transductores, como por ejemplo las galgas extensiométricas, termistores o RTD, que necesitan de la misma, bien por su constitución interna, (como el termistor, que es una resistencia variable con la temperatura) o bien por la configuración que se conectan (como el caso de las galgas, que se suelen montar en un puente de Wheatstone).

En el caso de las celdas de carga el voltaje de excitación fluctúa entre 10 a 15 voltios sea este alterno o continuo.

El voltaje recomendado por el fabricante es de 10 Voltios continuos siendo el máximo de 15 Voltios. Mientras mayor sea el voltaje de excitación mayor será el rango de salida a plena carga ya que depende de la relación mV/V de la celda.

Para el diseño de la balanza se tomó como voltaje de excitación el recomendado por el fabricante esto es de 10 V continuo lo que me da un valor máximo de salida a plena carga de 20 mV.

1.3.3. FILTRADO

El fin del filtro es eliminar las señales no deseadas de la señal que estamos observando. Por ejemplo, en las señales cuasi-continuas, (como la temperatura) se usa un filtro de ruido de unos 4 Hz, que eliminará interferencias, incluidos los 50/60 Hz. de la red eléctrica.

Las señales alternas, tales como la vibración, necesitan un tipo distinto de filtro antialiasing, que es un filtro pasabajo pero con un corte muy brusco, que elimina totalmente las señales de mayor frecuencia que la máxima a medir, ya que si no se eliminasen aparecerían superpuestas a la señal medida, con el consiguiente error.

Para la implementación del filtro me decidí por un filtro activo pasa bajos debido a que tenemos que eliminar la frecuencia de la red eléctrica que afecta al correcto funcionamiento del sensor.

1.3.3.1. FILTROS ACTIVOS

Los filtros son circuitos capaces de controlar las frecuencias permitiendo o no el paso de estas dependiendo de su valor.

Un filtro es un amplificador selectivo con las frecuencias. Se llaman activos ya que constan de elementos pasivos (células R-C) y elementos activos como el OP-AMP.

El funcionamiento de las células se basa principalmente en su actuación como divisor de tensión. Al aumentar la frecuencia de la señal, la reactancia del condensador disminuirá y entrará más o menos tensión al OP-AMP, dependiendo de si es pasa altos o pasa bajos respectivamente.

Para cualquier filtro se emplean las siguientes definiciones:

- **Frecuencia de corte.** Es aquella en la que la ganancia del circuito cae a -3 db por debajo de la máxima ganancia alcanzada. En los filtros pasa y elimina banda existen dos: una superior y otra inferior.
- **Banda pasante.** Conjunto de frecuencias de ganancia superior a la de corte en un margen menor o igual a 3 db.
- **Calidad.** Especifica la eficacia del filtro, es decir, la idealidad de su respuesta. Se mide en db/octava; db/década. Lo ideal sería que tomara un valor infinito

Hay gran variedad de estructuras en filtros. Cada una suele llevar el nombre de su inventor. Existen gran número de fórmulas deducibles para lo cual se logra el correcto funcionamiento del filtro, pero para que no nos resulte muy complicado de entender nos limitaremos a mencionar las más importantes.

1.- Valor de la frecuencia de corte, a partir de esta ecuación se deducirán todas las demás:

$$fo = \frac{1}{(2 \bullet \pi \bullet R \bullet C)}$$

2.- Tanto para montar un filtro de orden 1 como de orden 2 conocida la frecuencia central o de corte se debe fijar los valores de C1 = C2 = C para pasar a obtener los valores de las resistencias del circuito R1 = R2:

$$R1 = R2 = \frac{1}{(2 \bullet \pi \bullet C \bullet fo)}$$

3.- Ahora fijamos el valor de R3 y calculamos el valor de P para lograr la ganancia correcta del filtro:

$$P = R3 \bullet (Av - 1)$$

La ganancia de cada etapa es importante ajustarla para compensar el consumo de las células R-C y no afecte a la ganancia total del filtro. Dicha ganancia para cada orden de filtro viene dado por la siguiente tabla:

	A_{v0}	A_{v1}	A_{v2}	A_{v3}	A_{v4}
n = 1	1				
n = 2		1,586			
n = 3	1	2			
n = 4		2,235	1,152		
n = 5	1	2,382	1,382		
n = 6		2,482	1,586	1,068	
n = 7	1	2,555	1,753	1,198	
n = 8		2,610	1,889	1,337	1,038

Tabla I: Ganancias según el orden del filtro

Se pueden construir filtros mucho más selectivos con las frecuencias encadenando varios filtros de dichos tipos. Así encadenando un filtro de orden 1 y otro de orden 2, se obtiene un nuevo filtro de orden 3. Para lograr esto se debe usar siempre el mayor número posible de

filtros de orden 2 y en primer lugar el de orden 1, dependiendo del orden de filtro a construir. De este modo se logra que la curva de respuesta sea mucho más vertical y más próxima a la frecuencia central acercándose a la respuesta ideal. Pero esta construcción también es más cara y no siempre merece la pena emplearla.

1.3.3.1.1. FILTRO PASA BAJO

Se trata de un filtro que permite el paso de las frecuencias inferiores a una frecuencia conocida llamada frecuencia central (Fc) atenuando enormemente las frecuencias superiores a dicha frecuencia central. Su respuesta no es ideal, en los gráficos aportados se puede observar dicha diferencia creadas por las limitaciones de la electrónica

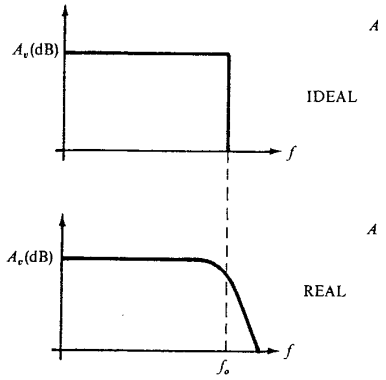


Figura 14: Respuesta de frecuencia de los filtros pasa bajo

La repuesta en frecuencia es

$$H(s) = \frac{K}{R1R2C1C2s^2 + \left[R1C2 + R2C2 + (1-K)R2C1\right]s + 1}$$

En el siguiente montaje se puede observar el filtro pasa bajo de orden 2. Obsérvese que el número de orden del montaje coincide con el número de células R-C. El diseño es igual que para el filtro pasa altos pero intercambiado las resistencias por los condensadores.(A Sanz y J. I. Artigas Diseño de filtros Activos)

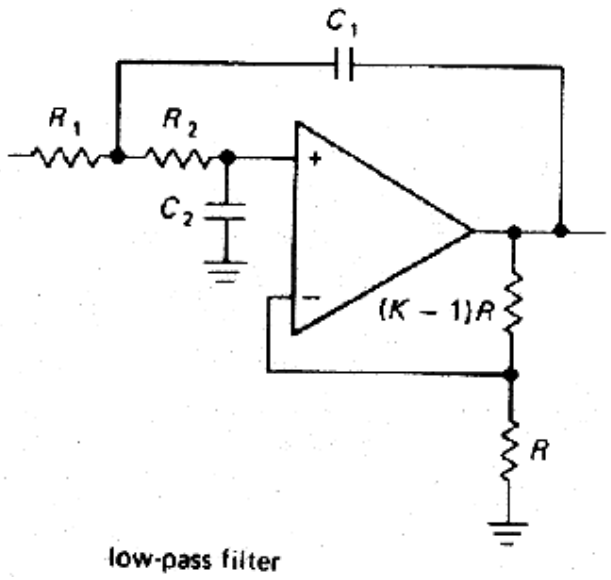


Figura 15: Filtro pasa bajo

En la etapa de filtrado de la señal se utilizó un filtro pasa bajo como el que se puede observar en la figura 15 se lo construyó para que la ganancia sea unitaria y así no afecte a la ganancia propia del amplificador de instrumentación.

Estas son las tres etapas más importantes del acondicionamiento de señal y son las que están implementadas en el circuito acondicionador.

1.4. LCD HD-44780

El LCD HD-44780 es el display utilizado por la balanza electrónica. Se trata de un módulo microcontrolado capaz de representar 2 líneas de 16 caracteres cada una. A través de 8 líneas de datos se le envía el carácter ASCII que se desea visualizar así como ciertos códigos de control que permiten realizar diferentes efectos de visualización. Igualmente mediante estas líneas de datos el módulo devuelve información de su estado interno.

Con otras tres señales adicionales se controla el flujo de información entre el LCD y el equipo de informática que lo gestiona.

A continuación se presenta la descripción de señales empleadas por el módulo LCD, así como el número de patilla a la que corresponden.

NUMERO DE PIN	DESCRIPCIÓN
1	GND
2	VDD
3	Vo
4	RS
5	R/W
6	E
7	DB0
8	DB1
9	DB2
10	DB3
11	DB4
12	DB5
13	DB6
14	DB7

Tabla II: Descripción de los pines del display LCD

Este display posee 2 tipos de memorias RAM, las cuales son: La RAM de datos de visualización (DDRAM), en donde la posición de cada caracter se encuentra asociado a una dirección de esta memoria y la RAM generadora de caracteres (CGRAM) la cual le permite al usuario definir por programa hasta 8 patrones de caracteres en un formato de 5x7 puntos. También posee una ROM generadora de caracteres (CGROM) en la cual se encuentra almacenado 192 caracteres definidos. Al poseer circuitería interna el dispositivo solo necesita recibir instrucciones para trabajar. Las principales de este módulo LCD se resumen en la siguiente tabla.

Nombre constante	Valor	Significado
LCDLinea1	0x80	Coloca cursor en la posición 1 línea 1
LCDLinea2	0x0C	Coloca el cursor en la posición 1 línea 2
LCDCLR	0x01	Borra la pantalla + LCDLinea1
LCDCasa	0x02	Como LCDLinea1
LCDInc	0x06	El cursor incrementa su posición tras cada carácter
LCDDec	0x04	El cursor decrementa su posición tras cada carácter
LCDOn	0x0C	Enciende la pantalla
LCDOff	0x08	Apaga la pantalla
CursOn	0x0E	Enciende pantalla mas cursor
CursOff	0x0C	Apaga pantalla mas cursor
CursBlink	0x0F	Enciende la pantalla con cursor parpadeando
LCDIzda	0x18	Desplaza los caracteres mostrados a la izquierda
LCDDecha	0x1C	Desplaza los caracteres mostrados a la derecha
CursIzda	0x10	Mueve el cursor una posición a la izquierda
CursDecha	0x14	Mueve el cursor una posición a la derecha
LCDFuncion	0x38	Programa una interface 8 bits, pantalla 2 líneas, fuente 5x7 pixeles
LCDCGRAM	0x40	Programa el generador de caracteres del usuario RAM

Tabla III: Principales instrucciones del despliegue LCD

La DDRAM puede almacenar hasta 80 caracteres, al energizar el despliegue solo se verá la mitad izquierda (8 caracteres al izquierda), al configurarlo con dos líneas se logra encender todos los 16 caracteres, en donde las posiciones de 0 a 7 son visibles y las posiciones de 8 a 39 son invisibles para la primera línea, lo mismo ocurre en la segunda línea de la 40 a la 47 son visibles y de la 48 a la 80 se muestran invisibles. Esto es importante tomar en cuenta al momento de la programación del despliegue.

Este módulo LCD tiene dos registros seleccionables. El registro de instrucciones (IR), en el que se almacena las instrucciones o las direcciones de las dos memorias RAM (DDRAM o CGRAM) a las que se desee acceder, y el registro de datos (DR) en el cual, se almacenan o reciben lo que se envía entre el microprocesador y la DDRAM o CGRAM. Estos registros pueden ser seleccionados mediante la línea selectora RS (Register Select) y son los únicos que pueden ser controlados por el microprocesador.

1.4.1. COMUNICACIÓN ENTRE EL LCD Y EL PIC

La comunicación entre este módulo LCD y el microcontrolador se realiza de la siguiente manera:

Se configura al LCD para que la comunicación sea a 4 bits, sin uso de retardos vía software (por lo tanto el reloj puede ser alterado sin problemas), se usa el puerto D del microcontrolador. Todas las rutinas de envío de datos o comandos se pueden realizar cuando el BUSY FLAG del LCD indica que está libre.

Esquema de conexión:

- **RD0:** Este pin es configurado como entrada, más puede ser usado como salida si lo necesitamos de ese modo ya que este no es utilizado en la comunicación.
- **RD1:** Se conecta con la señal de control R/S. Sacando un nivel lógico “0” por ella, se selecciona el registro de control del módulo. Sacando un nivel lógico “1” se selecciona el registro de datos. Esta línea debe programarse como salida.
- **RD2:** Se conecta con la señal R/W. Sacando un nivel lógico “0” por ella, el módulo es escrito con la información presente en ese momento en el puerto D alto que

deberá actuar como salida. Sacando un “1” se lee el estado interno del módulo LCD. Dicho estado se recibe a través del mismo puerto D que deberá ser programado como entrada. La línea RD2 debe programarse como salida.

- **RD3:** Se conecta con la señal E. Cuando se aplica un nivel “1” el módulo queda habilitado y es posible por lo tanto la transferencia de información entre el puerto y las líneas de datos. Aplicando un “0” el módulo queda desconectado y sus líneas de datos en alta impedancia. RD3 debe programarse también como salida.
- **RD4-RD7:** Están conectados a las líneas de datos D4-D7 del módulo. A través del puerto D, se envían códigos ASCII o de control al módulo o, se recibe por parte de este, el estado interno del mismo. Habrá que programarlo en forma vi. direccional, como salida si se quiere enviar códigos o comandos y como entrada si se desea conocer el estado interno del mismo.

1.4.2. SECUENCIA DE INICIALIZACIÓN

El módulo LCD ejecuta automáticamente una secuencia de inicio interna en el instante de aplicarle la tensión de alimentación si se cumplen los requisitos de alimentación expuestos en el siguiente cronograma:

Dichos requisitos consisten en que el tiempo que tarda en estabilizarse la tensión desde 0.2V hasta los 4.5V mínimos necesarios sea entre 0.1mS y 10mS. Igualmente el tiempo de desconexión debe ser como mínimo de 1mS antes de volver a conectar.

La secuencia de inicio ejecutada es la siguiente:

1.- Se ejecuta el comando **CLEAR DISPLAY** borrando la pantalla. La bandera **BUSY** se mantiene a “1” (ocupado) durante 15 mS hasta que finaliza la inicialización.

2.- Se ejecuta el comando **FUNCTION SET**, que establece el interfaz con el bus de datos. Se elige por defecto el tamaño de bus de datos a 8 bits (DL=1) y el número de reglones del display en 1 (N=0)

3.- Se ejecuta el comando **DISPLAY ON/OFF CONTROL**, que hace que el display quede en OFF (D=0); también cursor en OFF (C=0) y sin parpadeo del cursor en (B=0).

4.- Se ejecuta el comando **ENTRY MODE SET**, que establece la dirección de movimiento del cursor con auto incremento del cursor (I/D=1) y modo normal, no desplazamiento, del display (S=0).

Si la conexión de alimentación no reúne las condiciones que exige el módulo LCD, habría que realizar la secuencia de inicialización por software. En cualquier caso, es importante enviar al LCD la primera instrucción de trabajo después de que hayan transcurrido 15 mS, para completar dicha secuencia de inicialización.

1.4.3. DIAGRAMAS DE TIEMPOS

Como ya hemos visto la interfase que se está desarrollan es la que se implementa a 4 bits. En la figura 16 se muestra los diagramas de tiempo para este tipo de interfase como se puede observar en este el pin de la habilitación (enable) captura los datos en el flanco de bajada los pines RS y RW se mantienen en bajo hasta que la palabra enviada se escriba en el LCD. Si nosotros queremos leer la bandera busy flan esta tiene que ser hecha con el pin RW en alto y también en el flanco de bajada del pin de habilitación.(LCD HD 44780 www.hitachi.com)

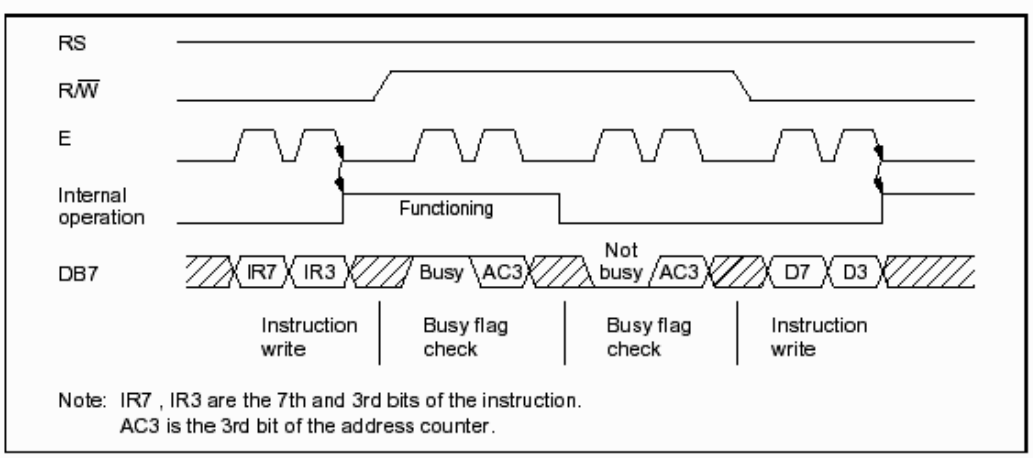


Figura 16: Diagramas de tiempo del ciclo de escritura

Write-Cycle	**V_{DD}**	2.7 - 4.5 V (2)	4.5 - 5.5 V (2)		2.7 - 4.5 V (2)	4.5 - 5.5 V (2)	
Parameter	**Symbol**	**Min(1)**		**Typ(1)**	**Max(1)**		**Unit**
Enable Cycle Time	t_c	1000	500	-	-	-	ns
Enable Pulse Width (High)	t_w	450	230	-	-	-	ns
Enable Rise/Fall Time	t_r, t_f	-	-	-	25	20	ns
Address Setup Time	t_{as}	60	40	-	-	-	ns
Address Hold Time	t_{ah}	20	10	-	-	-	ns
Data Setup Time	t_{ds}	195	80	-	-	-	ns
Data Hold Time	t_h	10	10	-	-	-	ns

Tabla IV: Señales de tiempo del ciclo de escritura

1.5. EL TECLADO

El prototipo que estoy desarrollando requiere de un pequeño teclado matricial para diferentes procesos que se van implementado.

Frecuentemente en nuestros proyectos necesitamos poder entrar datos para determinado uso. Esto se lo puede hacer con un simple interruptor, o con una interfase con la PC u otro tipo de equipo.

Un dispositivo común para esto es el teclado matricial, este usualmente consiste de 12 o 16 teclas arregladas en una matriz de 3x4 o 4x4 respectivamente.

Para la entrada de datos de la balanza contaremos con un teclado de 16 teclas que tienen un arreglo de 4 filas y 4 columnas. Los interruptores de conexión están alambrados en una matriz que tiene un tipo de arreglo que mantiene al conector un voltaje tan bajo como es posible.

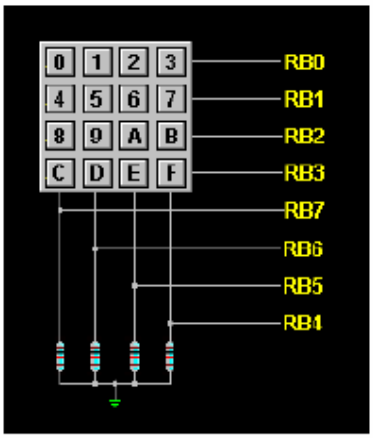


Figura 17: Teclado matricial

La operación de este teclado es bastante fácil ya que cada tecla se puede representar como un simple interruptor.

Si la tecla “0” es presionada, esta crea un corto entre la fila 1 y la columna 1

Si la tecla “1” es presionada, esta crea un corto entre la fila 1 y la columna 2

Si la tecla “2” es presionada, esta crea un corto entre la fila 1 y la columna 3

Si la tecla “3” es presionada, esta crea un corto entre la fila 1 y la columna4

Si la tecla “4” es presionada, esta crea un corto entre la fila 2 y la columna 1

Si la tecla “F” es presionada, esta crea un corto entre la fila 4 y la columna 4

Los pines RB0 a RB3 están conectados a las filas y los pines RB4-RB7 están conectados a las columnas.

La tarea que tenemos en manos es como leer este teclado con un PIC.

1.5.1. COMUNICACIÓN ENTRE EL TECLADO Y EL PIC

La solución es razonablemente simple e involucra escribir un software que pueda detectar si una tecla es presionada. Esta acción puede ser llamada “examinar el teclado”.

Si nosotros pusiéramos los pines del puerto B conectados a las columnas como entradas, y llevadas a tierra vía resistores de 10 K, ahora leemos el puerto B, veríamos que RB4 a RB7 estarían en un nivel lógico de “0”.

Ahora suponemos que los pines RB0 a RB3 están puestas como salidas y están todas a un nivel lógico “0”. En este estado ¿los pines de entrada se leerían diferentes si una tecla es presionada? No, ellas no se leerían diferente si la tecla es presionada, siempre se leería en un nivel lógico “0”. Sin presionar también se leería “0”.

Como podemos observar en la figura 18 el nivel lógico no cambia de cero en las entradas RB4 a RB7 si presionamos o no la tecla debido a que como se mencionó el nivel lógico de la salida es cero.

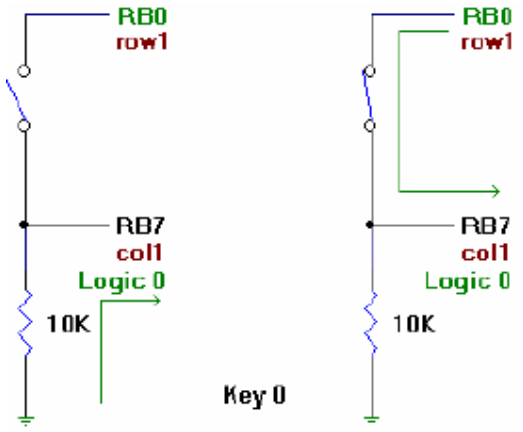


Figura 18: Nivel lógico “0”

Qué pasaría si nosotros pusiéramos el pin RB0 como una salida en lógica “1”. Ponemos 5V en el teclado a lo largo de la fila 1. Cuando no es presionada ninguna tecla, los pines RB4 a RB7 todavía estarían con un nivel lógico “0”, ahora, suponemos que la tecla 0 es presionada, esto causaría que el pin RB7 se lea con un nivel lógico de “1”.

Si la tecla 1 es presionada, RB6 se leerá con un nivel lógico “1”

Si la tecla 2 es presionada, RB5 se leerá con un nivel lógico “1”

Si la tecla 3 es presionada, RB4 se leerá con un nivel lógico “1”

Si la tecla 4 es presionada, RB7 se leerá con un nivel lógico “0”

Esta situación vemos en la figura 19 el nivel lógico cambia de cero a uno en las entradas que tengan las teclas 1 al 3 debido que toda la fila tiene un nivel lógico alto, en cambio las demás filas se mantendrían en bajo. El software diseñado hace que los pines que están siendo filas roten cambiando su nivel lógico de bajo a alto continuamente esperando que sea presionada cualquier tecla para a fin de que el microcontrolador pueda leer este cambio de estado poniendo las columnas del teclado como entradas del mismo.

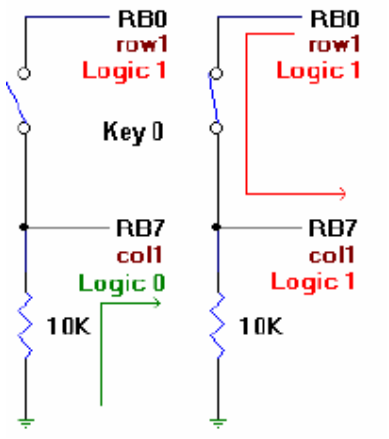


Figura 19: Nivel lógico 1

¿Por qué RB7 no se lee con un nivel lógico de “1” cuando la tecla 4 fue presionada?

Esto es porque nosotros solo pusimos a RB0 en un nivel lógico “1” y RB1 está todavía en un nivel “0”, por consiguiente la fila 2 está a un nivel lógico “0”. Cuando la tecla 4 es presionada se está cambiando a un nivel lógico “0” a RB7.

Se ha notado que cuando una fila es puesta a 5V y una tecla es presionada en esa misma fila, nosotros podemos leer el nivel lógico que cambia en la correspondiente columna de entrada.

Ahora tenemos que examinar el teclado, para esto a cada fila secuencialmente colocamos un “1” lógico y miramos en la columna para ver si está en un nivel lógico “1”. Si una columna está en nivel “1” una tecla ha sido presionada, si todas las columnas tienen un nivel “0” ninguna tecla ha sido presionada.

La primera cosa que hay que hacer es decidir cuan frecuentemente y cuan rápido se desea examinar el teclado. El tiempo puede variar dependiendo de los requerimientos del proyecto,

nosotros examinaremos el teclado cada 65 milisegundos o alrededor de 15 veces por segundo.

Para esto utilizaremos el temporizador propio del PIC llamado Timer 0 o TMR0 como recordaremos se encuentra localizado en la dirección 1H de la RAM. Sabemos que para operar el TMR0 se tienen que configurar el OPTION REGISTER, las primeras instrucciones que se tienen que hacer es poner al TMR0 que se incremente con el reloj interno del PIC (bit 5), el preescalar es asignado al TMR0 (bit 3), y por último el preescalar es puesto para una división por 256 (bits 2-0).

El TMR0 es incrementado en uno cada vez que el reloj interno del PIC se ha incrementado en 256. Este tiempo es causado por el preescalar que está asignado, cuando es puesto por una división de reloj de radio 256.

Lo segundo es examinar el teclado para esto hacemos RB0 a RB3 ir a un nivel lógico de “1” en giro. Esta secuencia pone a 5V cada una de las filas del teclado. Si uno de los pines RB4-RB7 están en un nivel lógico “1”cuando una tecla ha sido presionada tiene un valor de 0 a 15 que es retornada en una variable llamada **KEYVAL.**

Si no hay teclas presionadas se retorna un valor de 255 (0xFF).

La variable **Rows** mantendrá el valor que es usado para poner cada fila a un nivel lógico “1” en la rotación. La variable **Columns** es una variable que mantiene el valor cuando los bits de las columnas RB4-RB7 son leídos.

1.6. COMUNICACIÓN SERIE RS232

Una de las partes más importantes de los sistemas basados en microcontroladores es la comunicación con el mundo exterior. Tanto es así que la mayoría de los PICs modernos incluyen internamente una UART para la comunicación serie. Esta UART implementa de forma sencilla el protocolo de comunicación serie que utiliza la conexión RS232 por ejemplo.

Pero para poder conectar el PIC mediante una conexión RS232 no solo basta la UART (que resuelve el problema lógico), sino que es también necesario adaptar las señales eléctricas al estándar RS232. Esto lo realiza el chip ST232 junto con algún condensador adicional.

Se han elegido los puertos RC6 (TX) y RC7 (RX) para conectarlos directamente al MAX232 y realizar la comunicación serie mediante dos hilos. Esta elección se ha hecho así porque los PIC que tienen integrada la UART utilizan estas dos señales como transmisión y recepción.

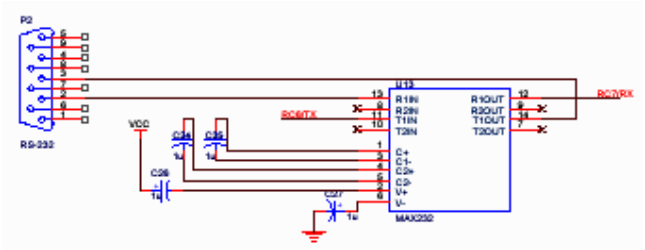


Figura 20: Comunicación serie RS232

1.6.1. PUERTO SERIE

Las comunicaciones serie se utilizan para enviar datos a través de largas distancias, ya que las comunicaciones en paralelo exigen demasiado cableado para ser operativas. Los datos serie recibidos desde un MODEM y otros dispositivos son convertidos a paralelo gracias a lo cual pueden ser manejados por el bus del PC.

Los equipos de comunicaciones serie se pueden dividir entre simplex, half-duplex y full-duplex. Una comunicación simplex envía información en una sola dirección (p.e. una emisora de radio comercial). Half-duplex significa que los datos pueden ser enviados en ambas direcciones entre dos sistemas, pero una sola dirección al mismo tiempo. En una transmisión full-duplex cada sistema pude enviar y recibir datos al mismo tiempo.

Hay dos tipos de comunicaciones: síncronas y asíncronas. En una transmisión síncrona los datos son enviados en bloques, el transmisor y el receptor son sincronizados por uno o más caracteres especiales llamados caracteres sync

El puerto serie del PC es un dispositivo asíncrono, en una transmisión asíncrona, un bit identifica su bit de comienzo y 1 o 2 bits identifican su final, no es necesario ningún carácter de sincronismo. Los bits de datos son enviados al receptor después del bit de inicio. El bit de menos peso es transmitido primero. Un carácter de datos suele consistir en 7 u 8 bits. Dependiendo de la configuración de la transmisión un bit de paridad es enviado después de cada bit de datos. Se utiliza para corregir errores en los caracteres de datos. Finalmente 1 o 2 bits de parada son enviados.

Descripción del Puerto: El puerto serie del PC es compatible con el estándar RS-232C. Este estándar fue diseñado en los 60s para comunicar un equipo terminal de datos o DTE (Data Terminal Equipment, el PC en este caso) y un solo equipo de comunicación de datos o DCE (Data Comunication Equipment, habitualmente un modem)

El estándar especifica 25 pines de señal, y que el conector DTE debe ser macho y el conector DCE hembra. Los conectores más usados son el DB-25 macho, pero muchos de los pines no son necesarios. Por esta razón en muchos PC modernos se utiliza los DB-9 macho. Los voltajes para un nivel lógico alto están entre -3V y -15V. Un nivel lógico bajo tendrá un voltaje entre +3V y +15V. Los voltajes más usados son +12V y -12V.

Las señales más utilizadas se listan a continuación:

DTR (Data Terminal Ready): El PC indica al MODEM que está encendido y listo para enviar datos.
DSR (Data Set Ready): El MODEM indica al PC que está encendido y listo para transmitir o recibir datos.
RTS (Request To Send): El PC pone esta señal a 1 cuando tiene un carácter listo para ser enviado.
CD (Carrier Detect): El MODEM pone esta señal a 1 cuando ha detectado el ordenador.
CTS (Clear To Send): El MODEM está preparado para transmitir datos. El ordenador empezará a enviar datos al MODEM.
TxD: El MODEM recibe datos desde la PC.
RxD: El MODEM transmite datos al PC.

El circuito integrado que convierte los datos de paralelo a serie y viceversa se llama UART (Universal Asynchonous Receiver Transmitter). La UART típica para un PC es el Intel 8251ª,

este circuito integrado puede ser programado para realizar comunicaciones serie síncronas o asíncronas.

Ocho bits de datos (D0-D7) conectan al 8251ª al bus de datos del PC. La entrada de chip select (CS) habilita el circuito integrado cuando es seleccionado por el bus de control del PC. Este circuito integrado tiene dos direcciones internas, una dirección de control y una de datos. La dirección de control queda seleccionada cuando la entrada C-/D está seleccionada a un nivel alto. La dirección de datos queda seleccionada cuando la entrada C-/D está a nivel bajo. La señal de RESET resetea el circuito integrado. Cuando RD está a nivel bajo el ordenador lee un byte de control o un byte de datos. La señal WR es habilitada por el PC para escribir un byte. Las dos señales están conectadas a las señales de control del sistema con los mismos nombres.

El UART incluye cuatro registros internos:

THR: Registro temporal de salida.
TSR: Registro de salida.
RDR: Registro de entrada.
RSR: Registro temporal de entrada.

Cada carácter a transmitir es almacenado en el registro THR. La UART añade los bits de inicio y paro. Luego copia todos los bits (datos, start y stop) al registro TSR. Para acabar el proceso los bits son enviados a la línea a través de la señal TD.

Cada carácter recibido desde la línea RD es almacenada en el registro RSR. Los bits de inicio y paro son eliminados y la UART escribe el carácter en el registro RDR. Para acabar el proceso el carácter es leído por la PC.

Direccionando el Puerto: Hay dos maneras de direccionar el puerto serie, a través de la interrupción 14H de la BIOS o a través de la interrupción 21H del DOS.

La interrupción 14H de la BIOS utiliza cuatro funciones para programar el puerto serie. Cada función es seleccionada asignando un valor al registro AH del microprocesador. Las cuatro funciones son listadas a continuación:

Función 00H: Inicializa el puerto serie y selecciona la velocidad, el número de bits de datos de start y de stop y los parámetros de paridad.
Función 01H: Envía un carácter al puerto serie especificado.
Función 02H: Lee un carácter desde el puerto serie especificado.
Función 03H: Devuelve el estado del puerto serie especificado.

Hay varias funciones de la interrupción 21H del DOS relacionadas a la operación del puerto serie:

Función 03H: Lee un carácter desde el puerto COM1
Función 04H: Escribe un carácter desde el puerto COM1
Función 40H: Esta función envía un número de bytes desde un buffer a un dispositivo especificado.

1.6.2. PINES DEL DB-9

En la siguiente tabla se muestra todas las señales del puerto serie así como el número de pines que posee el DB-9.

Pin	Name	Dir	Description
1	CD	←	Carrier Detect
2	RXD	←	Receive Data
3	TXD	→	Transmit Data
4	DTR	→	Data Terminal Ready
5	GND	—	System Ground
6	DSR	←	Data Set Ready
7	RTS	→	Request to Send
8	CTS	←	Clear to Send
9	RI	←	Ring Indicator

Tabla V: Patillaje del DB-9

(www.ipn.mx)

1.7. FUENTE DE ALIMENTACIÓN

Esta fuente fue diseñada para alimentar tanto al microcontrolador como para los diferentes periféricos que esta posee.

Esta fuente entrega voltajes de 5, 10 y -10 voltios. Como podemos apreciar en la figura 21 tenemos un transformador que baja la tensión de 120 a 24 voltios con un amperaje máximo de 1 Amperio con una derivación central para darnos el voltaje negativo. El puente rectificador esta compuesto por cuatro diodos 1N4001 agrupados de dos en dos; la primera pareja rectifica la parte positiva y la segunda la parte negativa. Los filtros rectificadores se escogieron de tal forma que dieran un voltaje de rizado no superior a tres voltios siendo su valor más aproximado a 1000μF ya que es una carga pesada.

El regulador KA 7810 entrega una corriente máxima de 1 Amperio con una potencia de salida máxima de 15 Vatios a 25ºC y 10V de regulación; con estas características de regulación y de corriente se procedió a escoger este regulador para que sea el que entregue

el voltaje a todo el circuito y especialmente a la celda de carga, debido a que el fabricante de la celda recomienda que sea 10V de corriente continua su alimentación. La máxima corriente que demanda el circuito no llega más allá de 400 mA y como se ve esto no excede la capacidad de entrega de corriente del 7810 SIENDO EL QUE MÁS CONSUME EL MICROCONTROLADOR. El 7910 solamente está para el voltaje negativo que requiere el LM324 de acuerdo al diseño de la tarjeta acondicionadora de tal forma que el 7910 podría suministrar hasta 500 mA como máximo y esto no excede en lo más mínimo el consumo del LM324 (www.datasheetarchive.com).

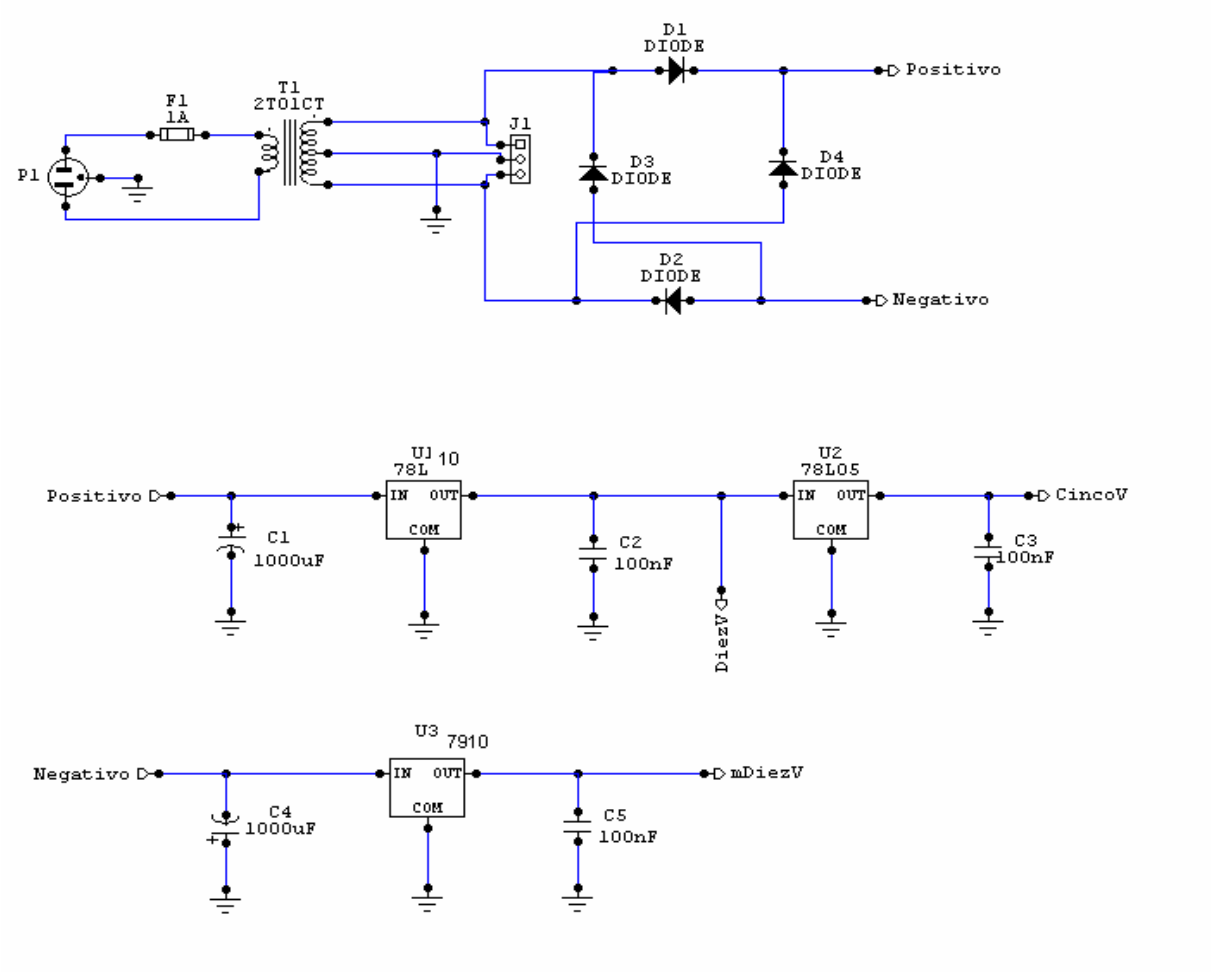


Figura 21: Fuente de alimentación

1.8. ESQUEMÁTICO DE LA BALANZA

En los siguientes gráficos tenemos los esquemáticos de la balanza con todas sus conexiones entre el microcontrolador y sus distintos periféricos.

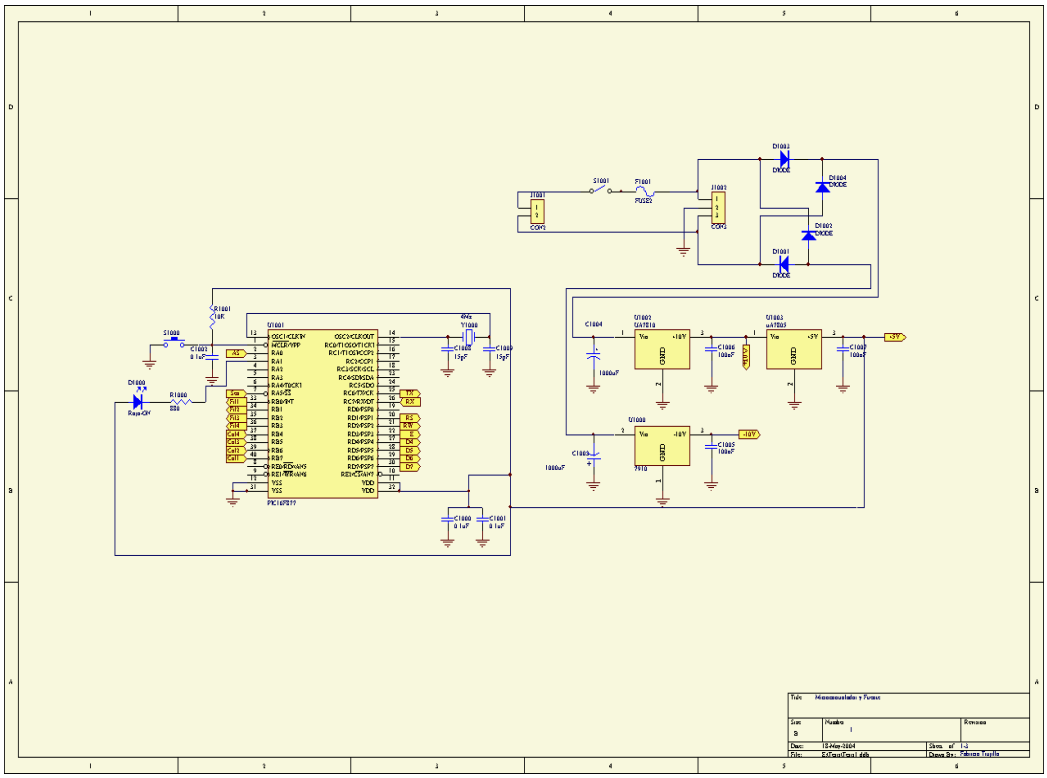


Figura 22: Esquemático del Microcontrolador y la Fuente de Poder

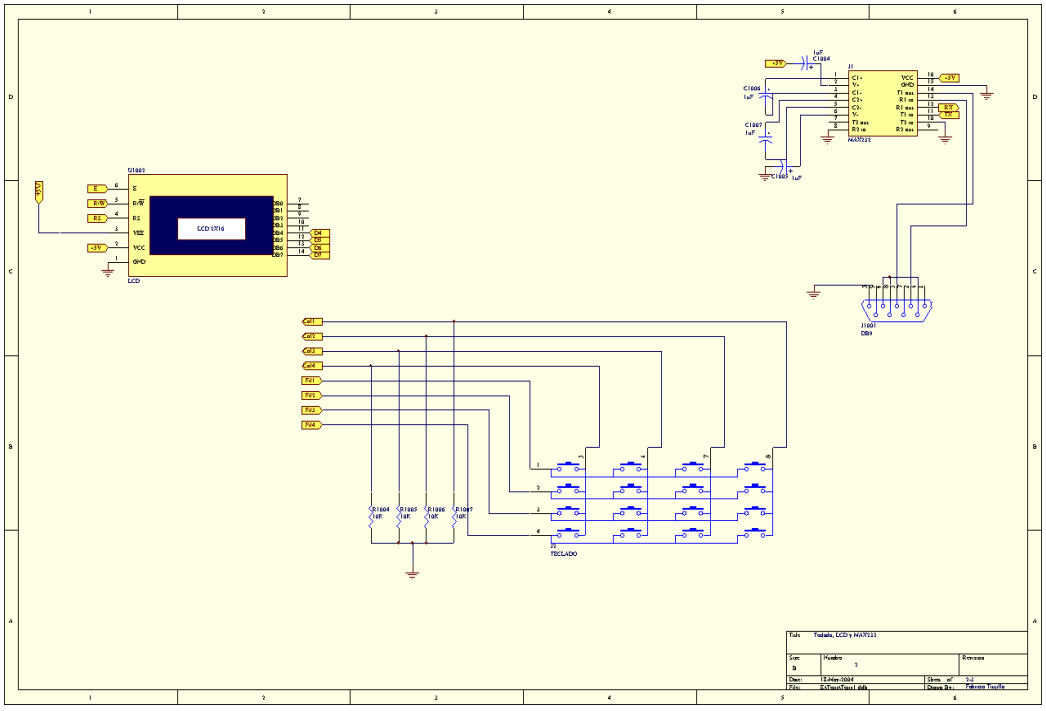


Figura 23: Esquemático del Teclado, LCD y MAX 232

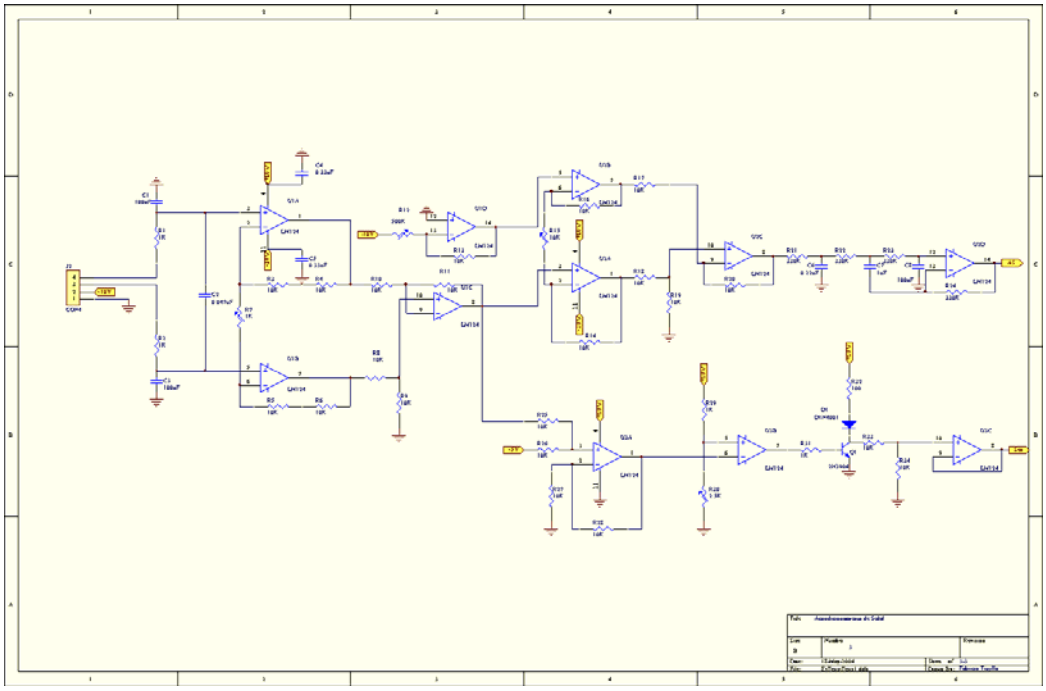


Figura 24: Esquemático del Acondicionamiento de Señal

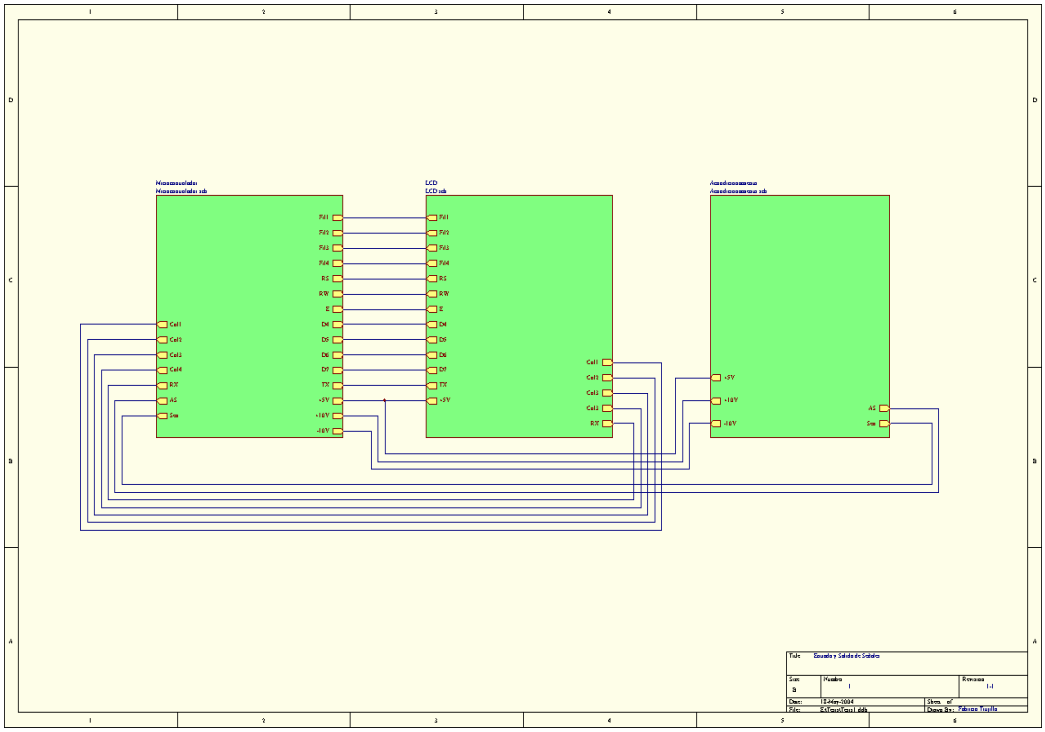


Figura 25: Esquemático de Entradas y Salidas de Señales

CAPÍTULO 2

2. DESARROLLO DEL SOFTWARE

A continuación se va a detallar el desarrollo de software tanto en lenguaje ensamblador como los respectivos diagramas de flujo del programa principal y todas las subrutinas que intervienen en el mismo.

Las variables que posee el programa van en función de la necesidad con que se desarrolló el programa, algunas de las cuales pueden estar redundantes o no se las tomó en cuenta.

Opte por poner las subrutinas de manera independiente para su mejor comprensión de esta forma se puede ir explicando mejor la realización de los mismos, pero antes se da una pequeña introducción viendo las diferentes instrucciones que tiene el ensamblador que utilizan los microcontroladores de la familia 16F87X de Microchip.

2.1. PROGRAMACIÓN EN LENGUAJE ENSAMBLADOR

El PIC 16F877 como ya se ha comentado posee una arquitectura RISC, **Computador de Juego de Instrucciones Reducido**. Esto implica no sólo que el juego de instrucciones sea pequeño (35 en el caso del micro que nos ocupa) sino que además posee las siguientes características:

- **Instrucciones simples y rápidas:** Son operaciones muy sencillas y por lo tanto pueden ejecutarse en un solo ciclo de instrucción.
- **Las instrucciones son ortogonales:** Apenas tienen restricciones en el uso de los operandos, de forma que cualquier instrucción puede usar cualquier operando.
- **La longitud de las instrucciones y datos es constante:** Todas las instrucciones tienen 14 bits y todos los datos son de un byte.

A continuación se describe el juego de instrucciones:

1.- Instrucciones de transferencia

Su principal función es la de transferir información a y desde el área de datos. En los mnemónicos utilizados la *f* indica el registro fuente, mientras que la *d* indica el registro destino.

Con estas instrucciones podemos mover datos cualesquiera para poder operar con ellos con las demás instrucciones del ensamblador. Se puede transferir tanto datos literales como datos entre registros.

INSTRUCCIONES DE TRANSFERENCIA			
SINTAXIS	**OPERACIÓN**	**CICLOS**	**SEÑALIZADORES**
MOVF f, d	Mueve el contenido del registro f al destino. Si d =0 el destino es W y si d=1 el destino es f.	1	Z
MOVWF f	Mueve el contenido del registro W al registro f.	1	----
MOVLW k	Carga un valor inmediato, el literal *k*, en el registro W.	1	----
SWAPF f, d	Se intercambian los 4 bits de más peso con los de menos peso de f, y el resultado se almacena en el destino, que será W si d = 0 y f si d = 1	1	----

Tabla VI: Instrucciones de transferencia

2.- Instrucciones Aritméticas

Las instrucciones de este grupo realizan diferentes operaciones aritméticas a través de la ALU; sumas, restas complementos, incrementos y decrementos y rotaciones.

Estas son todas las funciones matemáticas que el ensamblador del PIC 16F877 posee y con estas hay que realizar las diferentes rutinas de punto flotante que se requieren para el calculo del precio final y de la transformación a libras.

En el caso de las rutinas en punto flotante se las puede encontrar desarrolladas en la página de microchip (www.microchip.com) estas están a disposición del que desee lo único que no le dicen es como utilizarlas esto queda para el que está programando.

INSTRUCCIONES ARTIMÉTICAS			
SINTAXIS	**OPERACIÓN**	**CICLOS**	**SEÑALIZADORES**
ADDWF f, d	Suma el contenido de W con el de f. Si d = 0 el destino es W y si d = 1, es el registro f.	1	C, DC, Z
ADDLW k	Suma el valor literal *k* con el contenido del registro W.	1	C, DC, Z
SUBWF f, d	Resta el contenido del registro f menos el contenido de W. Si d = 0 el resultado se almacena en W. Si d = 1 se almacena en f. Si el resultado es negativo el acarreo se pone a 0	1	C, DC, Z
SUBLW k	Resta el contenido del registro W de un valor inmediato (k – W).	1	C, DC, Z
INCF f, d	Incrementa el contenido del registro f. Si d = 0 el resultado se almacena en W y si d = 1 se almacena en f.	1	Z
DECF f, d	Decrementa el contenido del registro f. Si d = 0 el resultado se almacena en W y si d = 1 se almacena en f.	1	Z
COMF f, d	Complementa el registro f. Si d = 0 el resultado se almacena en W y si d = 1 se almacena en f.	1	Z
RLF f, d	Rota el registro f a la izquierda a través del acarreo. Si d = 0 el resultado se almacena en W y si d = 1 se almacena en f.	1	C
RRF f, d	Rota el registro f a la derecha a través del acarreo. Si d = 0 el resultado se almacena en W y si d = 1 se almacena en f.	1	C

Tabla VII: Instrucciones aritméticas

3.- Instrucciones lógicas

Se presentan las instrucciones que efectúan las tres operaciones lógicas típicas sobre sus operandos: AND, OR y XOR.

INSTRUCCIONES LÓGICAS			
SINTAXIS	**OPERACIÓN**	**CICLOS**	**SEÑALIZADORES**
ANDWF f, d	Operación lógica AND entre los registros W y f. Si d = 0 el resultado se almacena en W y si d = 1 se almacena en f.	1	C, DC, Z
IORWF f, d	Operación lógica OR entre los registros W y f. Si d = 0 el resultado se almacena en W y si d = 1 se almacena en f.	1	C, DC, Z
XOR f, d	Operación lógica XOR entre los registros W y f. Si d = 0 el resultado se almacena en W y si d = 1 se almacena en f.	1	Z
ANDLW k	Operación AND de W con un valor inmediato.	1	Z
IORLW k	Operación OR de W con un valor inmediato.	1	Z
XORLW k	Operación XOR de W con un valor inmediato.	1	Z

Tabla VIII: Instrucciones lógicas

4.- Instrucciones de puestas a cero

Estas tres instrucciones borran, es decir ponen a cero el contenido de algún registro.

INSTRUCCIONES DE PUESTA A CERO			
SINTAXIS	**OPERACIÓN**	**CICLOS**	**SEÑALIZADORES**
CLRF	Borra el registro f, pone a cero todos sus bits.	1	Z
CLRW	Borra el registro W, pone a cero todos sus bits.	1	Z
CLRWDT	Borra o refresca el perro guardián WDT.	1	TO, PD (Registro de Control)

Tabla IX: Instrucciones de puestas a cero

5.- Instrucciones de salto

Estas instrucciones rompen la secuencia normal del flujo del programa, provocando saltos, por lo que todas afectan al contenido del PC. Las instrucciones referentes a subrutinas como son CALL, RETURN, RETLW y RETFIE, afectan también al contenido de la pila, ya que la primera introduce en la cima de la pila el contenido del PC; y las otras tres sacan la cima de la pila y la carga en el PC, de forma que esta será la siguiente instrucción que se ejecutará.

INSTRUCCIONES DE SALTO			
SINTAXIS	**OPERACIÓN**	**CICLOS**	**SEÑALIZADORES**
CALL k	Salto a una subrutina cuya primera instrucción está almacena en la dirección k.	2	TO, PD (Registro de Control)
GOTO k	Salto incondicional a la dirección k.	2	----
RETFIE	Retorno de la interrupción. (pone GIE=1 automáticamente)	2	----
RETLW k	Retorna de una subrutina y carga el valor k en el registro W.	2	----
RETURN	Retorno de una subrutina.	2	----
DECFSZ f, d	Se decrementa el contenido de f y si el resultado es 0 se salta la siguiente instrucción. Si d = 0 el resultado se almacena en W; si d = 1 en f.	1 ó 2	----
INCFZ f, d	Se incrementa el registro f, y si el resultado es 0 se salta la siguiente instrucción. Si d = 0 el resultado se almacena en W; si d = 1 en f.	1 ó 2	----

Tabla X: Instrucciones de salto

6.- Instrucciones para la manipulación de bits

Comprende un pequeño grupo de instrucciones que se encargan de verificar el valor de un bit particular de un registro, de ponerle a 1 o a 0, e incluso de producir un salto de una instrucción según se cumpla o no la condición establecida.

INSTRUCCIONES PARA LA MANIPULACIÓN DE BITS			
SINTAXIS	**OPERACIÓN**	**CICLOS**	**SEÑALIZADORES**
BCF f, b	Borra o pone a 0 el bit b del registro f.	1	----
BSF f, b	Activa o pone a 1 el bit b del registro f..	1	----
BTFSC f, b	Verifica el bit b del registro f y si vale cero se salta una instrucción.	1 ó 2	----
BTFSS f, b	Verifica el bit b del registro f y si vale 1 se salta una instrucción.	1 ó 2	----

Tabla XI: Instrucciones para la manipulación de bits

7.- Instrucciones especiales

Son dos instrucciones que realizan operaciones muy específicas y que no entran en ninguna de las categorías anteriores.

INSTRUCCIONES ESPECIALES			
SINTAXIS	**OPERACIÓN**	**CICLOS**	**SEÑALIZADORES**
NOP	No hace nada	1	----
SLEEP	Indica al procesador que debe entrar en el modo de reposo o bajo consumo.	1	TO, PD (Registro de Control)

Tabla XII: Instrucciones especiales

2.2. EL PROGRAMA PRINCIPAL

Este programa cumple con todas las funciones para las que fue diseñado originalmente. Se agruparon los módulos que fueron programados y probados en un solo programa: conversión A/D, Lectura/Escritura en la EEPROM, transmisión serial e interfase con el sensor. El programa es capaz de poder ingresar datos de precios vía teclado y poder calcular su precio total. Cuando recibe los datos del sensor de peso esta llama a la rutina de conversión analógica digital para su proceso una vez realizado esto se procede a almacenar en la EEPROM interna del microcontrolador para su transmisión posterior. Igualmente puede presentar en pantalla algunas funciones con las que fue programado como son: entrega de menús y verificación de datos en memoria.

Bits para la comunicación con el LCD

```
RS          EQU   0X01
RW          EQU   0X02
E           EQU   0X03
BUSYFLAG    EQU   0X07
        DADO
```

Variables para la comunicación con el teclado

Columns
Rows
Count
KeyFlag
KeyVal
NewKey
Tecla
Flags
Debounce

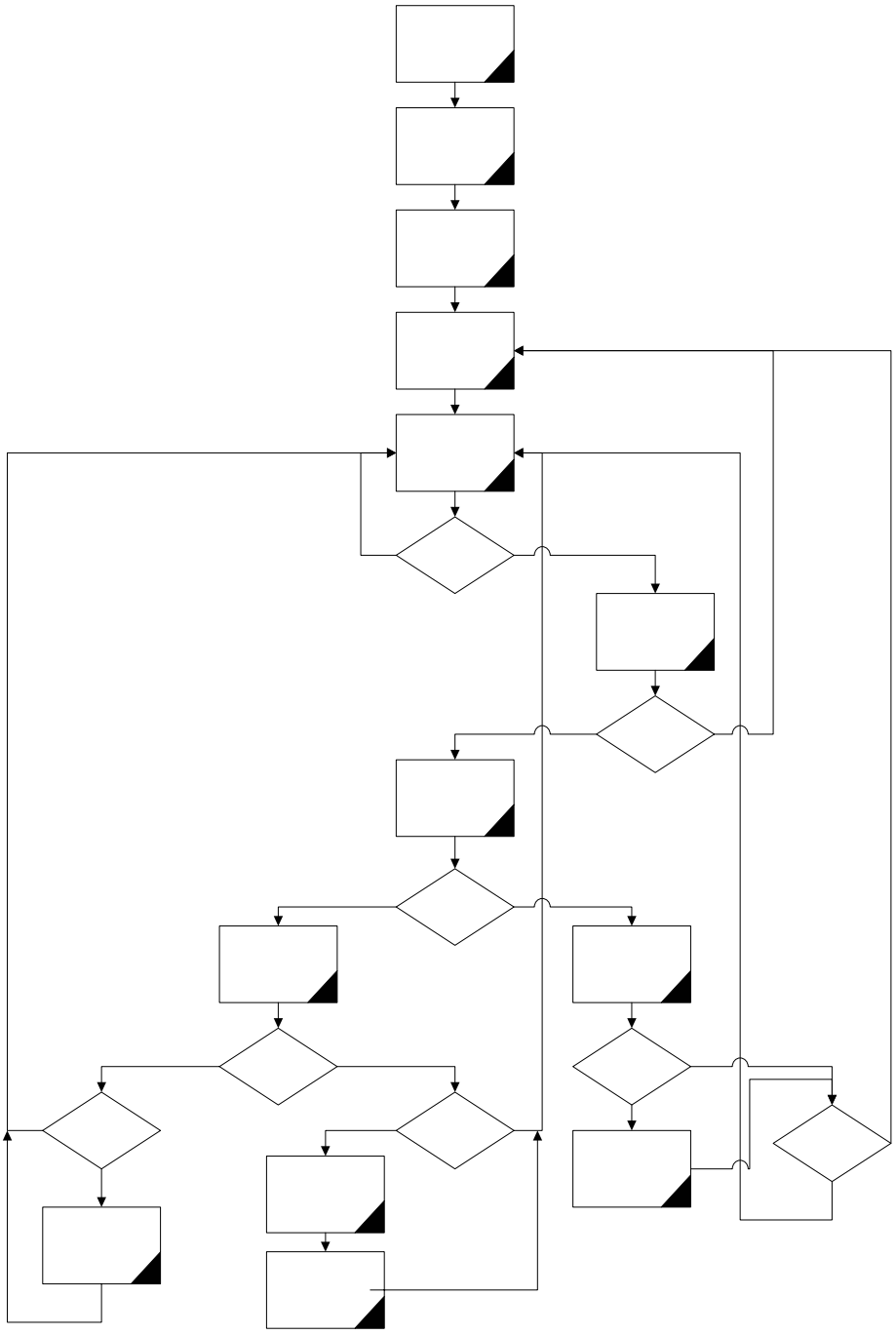
Variables para la conversión analógica

L_byte
H_byte
R0
R1
R2
DIG1
DIG2
DIG3
DIG4

Variables para la comunicación serial

dirmentrans
tempdato

2.2.1. DIAGRAMA DE FLUJO



2.2.2. CODIFICACIÓN EN LENGUAJE ENSAMBLADOR

```
Start
        CALL        InitPorts
        CALL        InitLCD
        bcf         PORTA,1
;=====================
; Inicialización de la balanza del tipo
experimental
;=====================
        clrf        Debounce
        clrf        Flags
; Estos registros se inicializan con cero
MnLoop
        Movlw       0x80
        CALL        EE
        movlw       'W'
        CALL        ED
        movlw       '='
        CALL        ED
        Movlw       0x87
        CALL        EE
        movlw       'k'
        CALL        ED
        movlw       'g'
        CALL        ED
        movlw       0xC7
        CALL        EE
        movlw       'P'
        CALL        ED
        Movlw       '='
        CALL        ED
;-----------------------------------------
        bcf         KeyActLCD
        bcf         ServAdc
MnLoop2
        movlw       0x82
        CALL        EE
        movlw       '0'
        CALL        ED
        movlw       '0'
        CALL        ED
        movlw       ','
        CALL        ED
        movlw       '0'
        CALL        ED
        movlw       '0'
        CALL        ED
        movlw       0xC9
        CALL        EE
        Movlw       '0'
        CALL        ED
        movlw       '0'
        CALL        ED
        movlw       ','
        CALL        ED
        movlw       '0'
        CALL        ED
        movlw       '0'
        CALL        ED
        movlw       '0'
        CALL        ED
        movlw       '$'
        CALL        ED

;---------------------------------------
Act
        btfss       INTCON,T0IF
        goto        Act
        bcf         INTCON,T0IF
        CALL        ServiceADC
        btfss       ServAdc
        goto        Ref
        goto        MnLoop
;---------------------------------------
Ref
        movf        Debounce,F
        btfsc       STATUS,Z
        goto        TestKP
        decf        Debounce,F
        btfsc       ServKey
        goto        ServiceKey
```

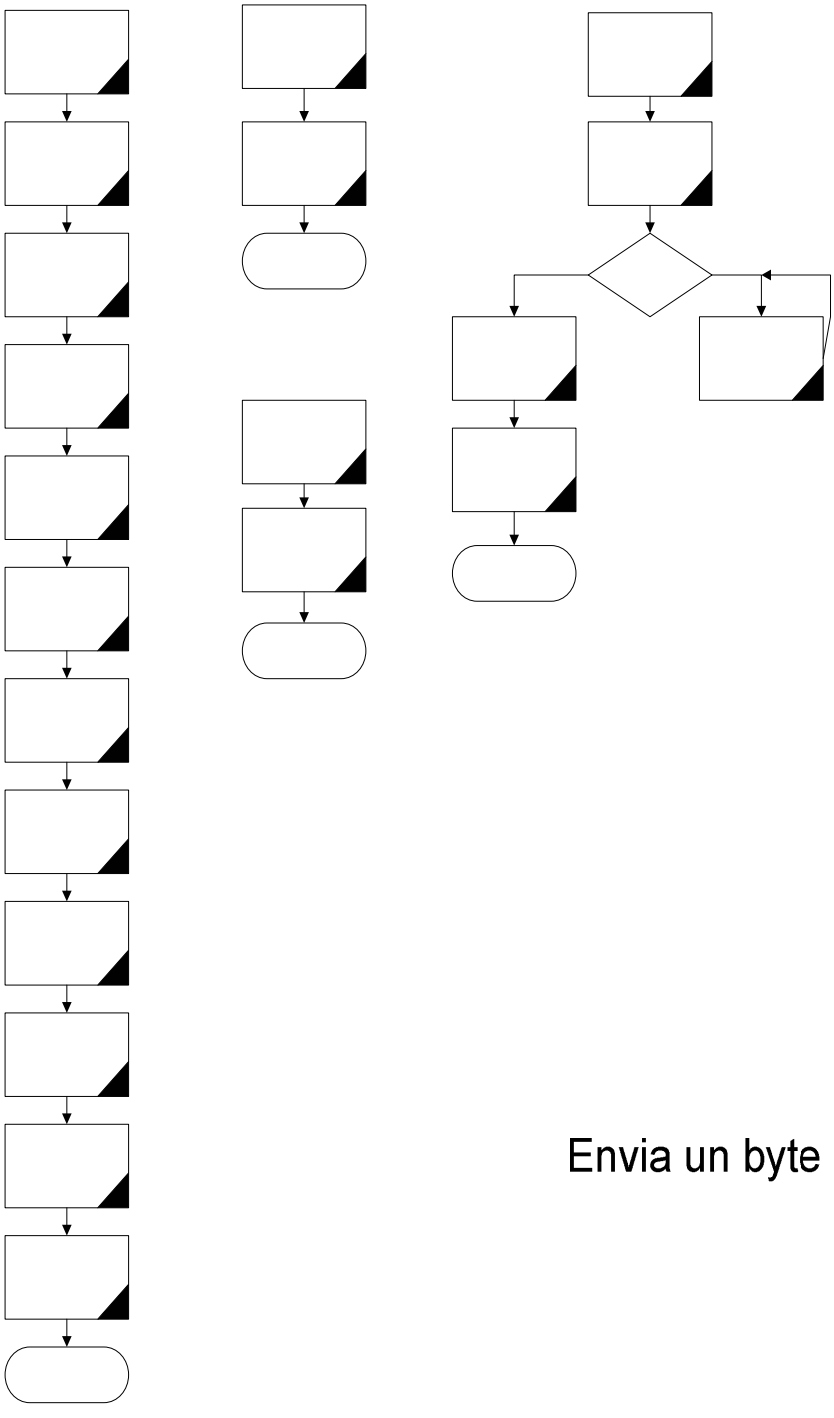
```
Ref1
        btfss       KeyActLCD
        goto        Act
        goto        MnLoop
TestKP
        call        KeyCheck
        ;Chequemos el teclado
        btfss       KeyVal,7
        goto        KeyIsDown
        btfss       Flags,Key
        goto        Act
        bcf         Flags,Key
        movlw       3h
        movwf       Debounce
        goto        Act

KeyIsDown
        btfsc       Flags,Key
        goto        Act             ;
        bsf         Flags,Key
        movlw       3h
        movwf       Debounce
        movf        KeyVal,W
        bsf         ServKey
        CALL        T_Conv
        movwf       NewKey
        goto        Act
```

2.3. SUBRUTINA DE INTERFASE CON EL LCD

Como ya hemos visto la interfase del LCD se la hace a 4 bits por consiguiente primero se escriben los cuatro bits de mayor peso para luego hacerlo con los de menor peso, no se hace uso de retardos para escribir los datos esto se logra chequeado la bandera que tiene el propio LCD, todo esto lo hace el puerto D que es el que está habilitado para la comunicación con el LCD.

2.3.1. DIAGRAMA DE FLUJO



Envia un byte

2.3.2. CODIFICACIÓN EN LENGUAJE ENSAMBLADOR

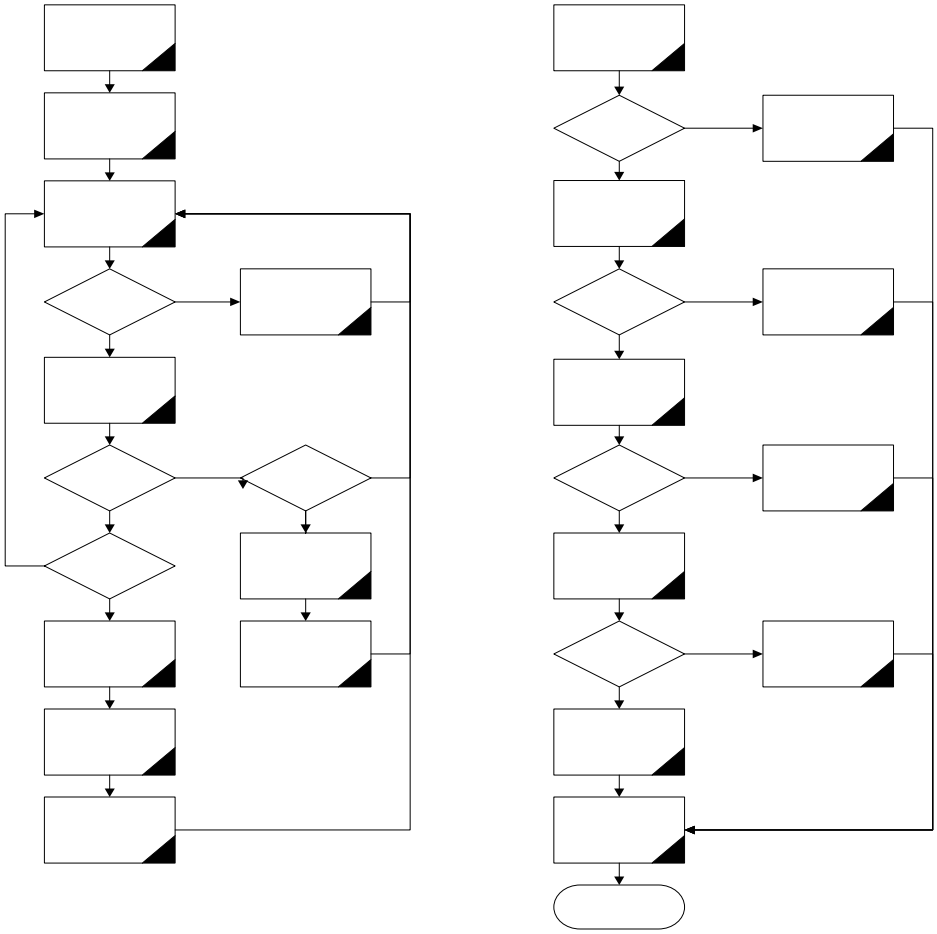
```
;*****************************
  ;***   RUTINAS DE LCD   ***
;*****************************
ED
        BSF         PORTD,RS
        BCF         PORTD,RW
ENVIABYTE
        BSF         PORTD,E
        MOVWF       DADO
        MOVLW       0X0F
        ANDWF       PORTD,F
        MOVLW       0XF0
        ANDWF       DADO,W
        IORWF       PORTD,F
        BCF         PORTD,E
        BSF         PORTD,E
        SWAPF       DADO,F
        MOVLW       0X0F
        ANDWF       PORTD,F
        MOVLW       0XF0
        ANDWF       DADO,W
        IORWF       PORTD,F
        BCF         PORTD,E
        CALL    CHECABUSYFLAG
        RETURN
*****************************
EC
        BCF         PORTD,RS
        BCF         PORTD,RW
        CALL        ENVIABYTE
        RETURN
*****************************
*****************************
EE
        BCF         PORTD,RS
        BCF         PORTD,RW
        CALL        ENVIABYTE
        RETURN
*****************************
CHECABUSYFLAG
        BSF         STATUS,RP0
        MOVLW       0XF1
        MOVWF       TRISD
        BCF         STATUS,RP0
        BCF         PORTD,RS
        BSF         PORTD,RW
        BSF         PORTD,E
        BTFSSPORTD,BUSYFLAG
        GOTO        RETORNO
        BCF         PORTD,E
        BSF         PORTD,E
        BCF         PORTD,E
        GOTO $ - 6
RETORNO
        BCF         PORTD,E
        BSF         PORTD,E
        BCF         PORTD,E
        BSF         STATUS,RP0
        MOVLW       0X01
        MOVWF       TRISD
        BCF         STATUS,RP0
        RETURN

*****************************
```

2.4. SUBRUTINA DE INTERFASE CON EL TECLADO

En caso del teclado la codificación se la hace para el Puerto B que es el escogido para este propósito. Cuando se presiona una tecla la subrutina procesa la información y devuelve el valor en ASCII de la tecla presionada sino la subrutina devuelve el valor FF.

2.4.1. DIAGRAMA DE FLUJO



2.4.2. CODIFICACIÓN EN LENGUAJE ENSAMBLADOR

```
*******************************
; Rutina de Test de teclado
*******************************
TestKey
        call        KeyCheck
        btfss       KeyVal,7
        goto        KeyPresionada
        btfss       Flags,Key
        return
        bcf         Flags,Key
        movlw       3h
        movwf       Debounce
        return
*******************************
KeyPresionada
        btfsc       Flags,Key
        return
        bsf         Flags,Key
        movlw       3h
        movwf       Debounce
        movf        KeyVal,W
        CALL        T_Conv
        bsf         KeyHit
        return
*******************************
; Rutina que chequea el teclado
*******************************
KeyCheck
        Movlw       b'00000001'
        Movwf       Rows
        clrf        KeyVal
RowLoop
        movf        Rows,W
        movwf       PORTB
        movlw       4h
        movwf       Count
DLoop
        decfsz      Count,F
        goto        DLoop
        movf        PORTB,W
        andlw       b'11110000'
        movwf       Columns
        btfss       STATUS,Z
        goto        ChkColm
        movlw       4h
        addwf       KeyVal,F
        goto        NextRow
ChkColm
        bcf         STATUS,C
        rlf         Columns,F
        btfsc       STATUS,C
        goto        LowRow
        incf        KeyVal,F
        movf        Columns,F
        btfss       STATUS,Z
        goto        ChkColm
NextRow
        bcf         STATUS,C
        rlf         Rows,F
        btfss       Rows,4
        goto        RowLoop
        movlw       0xFF
        movwf       KeyVal
LowRow
        clrf        PORTB
        RETURN
*******************************
;Tabla de valores de las teclas
; Col1 Col2 Col3 Col4
 ; (RB4) (RB5) (RB6) (RB7)
 ;
 ;Row1(RB0) 1 2 3 A
 ;
 ;Row2(RB1) 4 5 6 B
 ;
 ;Row3(RB2) 7 8 9 C
 ;
 ;Row4(RB3) * 0 # D
*******************************
```

```
T_Conv
        ADDWF   PCL,F
        RETLW   '*'
        RETLW   '7'
        RETLW   '4'
        RETLW   '1'
        RETLW   '0'
        RETLW   '8'
        RETLW   '5'
        RETLW   '2'
        RETLW   '#'
        RETLW   '9'
        RETLW   '6'
        RETLW   '3'
        RETLW   'D'
        RETLW   'C'
        RETLW   'B'
        RETLW   'A'
```

****

2.5. PROGRAMACIÓN PARA LA CONVERSIÓN ANALÓGICA

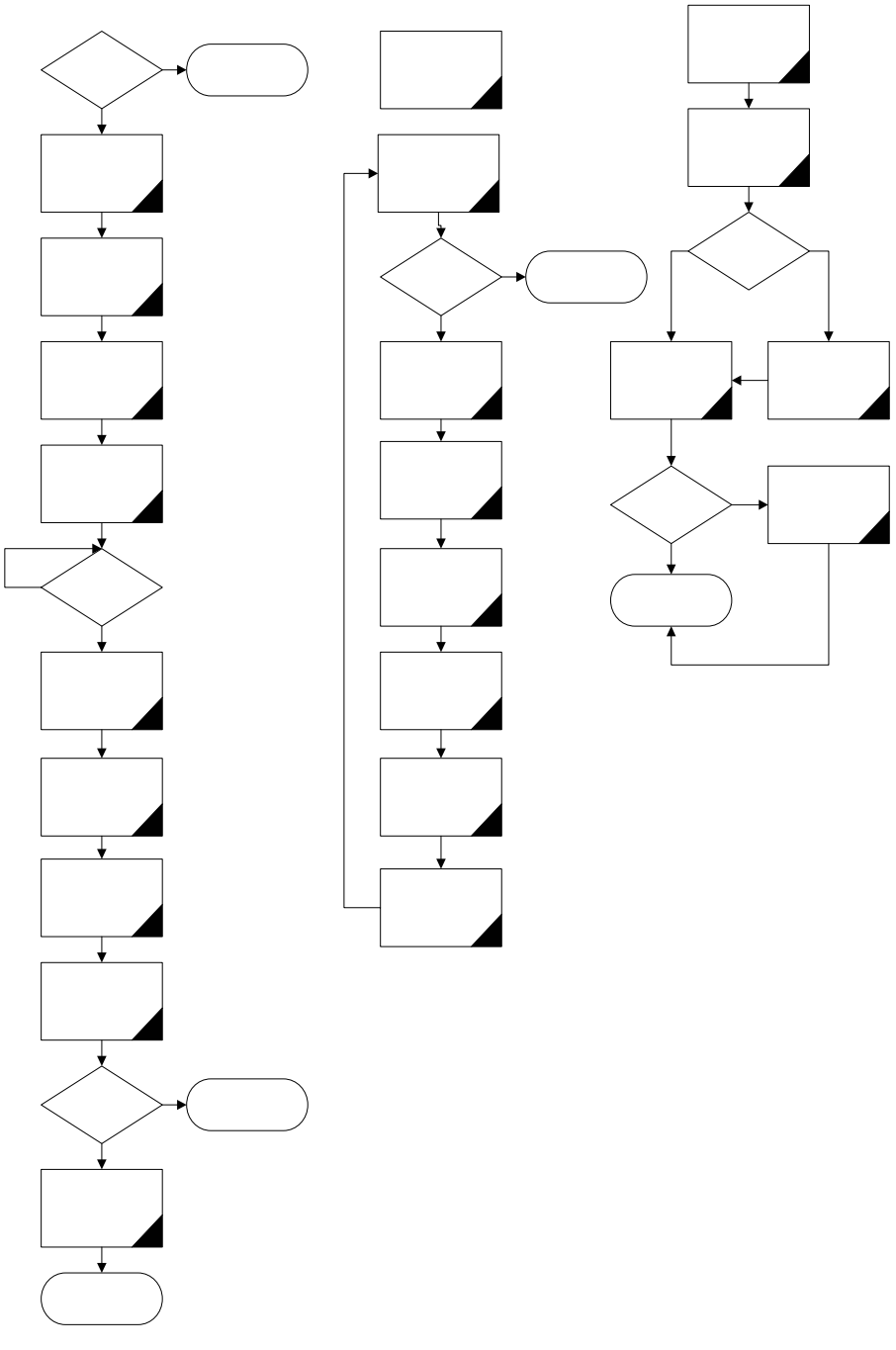
A continuación se presenta la rutina para la conversión analógica. Dicha rutina se puede encontrar en los manuales del PIC 16F877, los pasos seguidos para la conversión analógica digital se vio en el capítulo 1 lo único que se debe hacer es la codificación de los mismo.

La rutina para la conversión de los datos obtenidos en números BCD se verá a continuación con su diagrama de flujo, se utilizó el direccionamiento indirecto de los datos, como una manera más sencilla para realizarlo, el registro que se emplea para dicho propósito es el FSR que como vimos en el capítulo 1 este ayuda en la labor de direccionamiento de datos.

El convertidor que posee el microcontrolador como ya lo dijimos es de aproximaciones sucesivas esto hace que nosotros podemos conocer los datos con anterioridad ya que utiliza una fórmula conocida que es:

$Binario = \frac{2^n \times Vin}{Vref}$ Siendo n el número de bits para nuestro caso 10 y Vref el voltaje de referencia de 5V.

2.5.1. DIAGRAMA DE FLUJO



Rutina de Serv

PORTA,

V

Configuram
Puerto A

2.5.2. CODIFICACIÓN EN LENGUAJE ENSAMBLADOR

```
ServiceADC
        btfss       PORTA,5
        RETURN
;*****************************
MIDE

        bsf         STATUS,RP0
        bcf         PIE1,ADIE
        bcf         STATUS,RP0
;*****************************
        movlw       0x81
        movwf       ADCON0
        bcf         PIR1,ADIF
        bcf         INTCON,PEIE
        bcf         INTCON,GIE
        CALL        espera200
        bsf         ADCON0,GO
;*****************************
PRUEBA
        btfss       PIR1,ADIF
        goto        PRUEBA
;*****************************
        bcf         PIR1,ADIF
        bcf         ADCON0,GO
        bsf         STATUS,RP0
        movf        ADRESL,W
        bcf         STATUS,RP0
        movwf       L_byte
        movf        L_byte,W
        movwf       temp1
        movf        ADRESH,W
        movwf       H_byte
        movf        H_byte,W
        movwf       temp2
;*****************************
        CALL        espera20
        CALL        Conversión
        CALL        Actualizar
;*****************************
        movlw       0x82
        CALL        EE
        movf        DIG4,W
        CALL        ED
        movf        DIG3,W
```

2.5.3. CONVERSIÓN A BCD

```
Conversión
        bcf         STATUS,0
        movlw       .16
        movwf       contador
        clrf        R0
        clrf        R1
        clrf        R2

loop16  rlf         L_byte, F
        rlf         H_byte, F
        rlf         R2, F
        rlf         R1, F
        rlf         R0, F
        decfsz      contador, F
        goto        adjDEC
        RETLW       0

adjDEC  movlw       R2
        movwf       FSR
        call        adjBCD
        movlw       R1
        movwf       FSR
        call        adjBCD
        movlw       R0
```

```
        movwf   FSR
        call    adjBCD
        goto    loop16

adjBCD  movlw   3
        addwf   0,W
        movwf   temporal
        btfsc   temporal,3
        movwf   0
        movlw   30
        addwf   0,W
        movwf   temporal
        btfsc   temporal,7
        movwf   0
        RETLW   0
```

2.6. SUBRUTINA DE COMUNICACIÓN CON EL USART

La subrutina corresponde al módulo programado y probado de la transmisión por el puerto serial del microcontrolador.

Configuramos el puerto serial del microcontrolador de tal manera que la velocidad de transmisión se la realice a 9600 baudios entre el microcontrolador y la interfase RS-232. Esta rutina realiza la transmisión serial de cualquier dato de 8 bits que se desea a la vez. Para realizar la prueba, transmitimos los siguientes datos: b'10101010', b'01010101'. Pudimos observar que se realiza satisfactoriamente la transmisión por medio del osciloscopio.

El formato de transmisión es de 8 bits, con un bit de paro, uno de inicio y sin paridad. Cada bit se transmite en 18µs incluyendo el bit de inicio y sin bit de paro, el tiempo de transmisión es de 162µs incluyendo el bit de paro siendo el tiempo de transmisión total de 180µs.

Primero medimos el pin 25 (Tx) del microcontrolador, y pudimos observar que los datos transmitidos correspondían exactamente con lo que nosotros transmitíamos.

Esta salida es TTL (0-5V). Para poder hacer la interfase con la PC, necesitamos tener niveles RS-232 (±9V). Para esto utilizamos el DC14C232, que es una interfase 232 dual.

La conectamos al sistema mínimo (Tx). La salida del integrado (Tx-232) la enviamos al pin 2 del cable del puerto serial de la PC. Conectamos las tierras comunes y configuramos la hiperterminal para que la velocidad fuera igual a la de transmisión.

En la PC recibimos una cadena de U y * (U*U*U*……). Estos datos corresponden al código ASCII de los datos originalmente enviados por el microcontrolador. Con esta prueba confirmamos la correcta operación de nuestra interfase RS-232.

2.6.1. CODIFICACIÓN EN LENGUAJE ENSAMBLADOR

```
DRS232
        clrf        dirmentrans
Initrans
        bcf         STATUS,RP0
        bcf         STATUS,RP1
        movf        CNTaux1,W
        movwf       CNTAD
        CALL        EErd
        movwf       tempdato
        bsf         STATUS,RP0
        bsf         TRISC,6
        bsf         TRISC,7
        movlw       0x19
        movwf       SPBRG
        bsf         TXSTA,BRGH
        bcf         TXSTA,SYNC
        bcf         STATUS,RP0
        bsf         RCSTA,SPEN
        bsf         STATUS,RP0
        bcf         TXSTA,6
        bsf         TXSTA,TXEN
        bcf         STATUS,RP0
;-----------------------------------------
        movf        tempdato,W
        movwf       TXREG
        bsf         STATUS,RP0
        btfss       TXSTA,1
        goto        tesdato
        bcf     STATUS,RP0
;-----------------------------------------
        incf        CNTaux1,F
        incf        dirmentrans,F
        movf        dirmentrans,W
        xorlw       0x10
        btfss       STATUS,Z
        goto        Initrans

        RETURN
```

2.7. MÓDULO DE EEPROM (Lectura/Escritura)

Esta memoria se puede leer y borrar durante la operación normal en el rango completo de Vdd. Tiene una capacidad de 256x8 bytes. La memoria no permite un borrado masivo por parte del operador. Los datos de memoria no son mapeados directamente en los espacios de registros de archivo. Son indirectamente seleccionados a través de SFR (Special Function Registers).

Los SFR usados son los siguientes:

- EECON1
- EECON2
- EEDATA
- EEADATH
- EEADR
- EEADRH

La memoria permite hacer lecturas y escrituras de bytes. EEDATA contiene los datos de 8 bits a leer o guardar y EEADR contiene la dirección de la EEPROM a leer o escribir. El rango de direcciones va de 0H a FFH.

Esta EEPROM está diseñada para ciclos rápidos de lectura/escritura. El tiempo de escritura se controla por un temporizador interno en el microcontrolador. El tiempo de escritura varía con el voltaje y la temperatura.

Una escritura borra automáticamente la localidad y escribe los nuevos datos. La memoria de programa no puede ser acezada durante una escritura. Durante una escritura el oscilador continúa alimentando a los periféricos y continúa operando.

EECON1 es el registro de control para los accesos de memoria. EECON2 no es un registro físico y se usa exclusivamente para la secuencia de escritura de la memoria.

Los bits de control RD y WR inician las operaciones de lectura y escritura respectivamente. Sólo pueden ser limpiados o puestos a uno por software. Se limpian por hardware cuando una de estas dos operaciones concluye.

El bit WREN cuando vale 1, permitirá la operación de escritura. La bandera de interrupción EEIF en PIR2 se pone en uno cuando la lectura es completada. SE debe limpiar por software.

2.7.1. CODIFICACIÓN EN LENGUAJE ENSAMBLADOR

- **Rutina para entrar información en la EEPROM**

```
EEwr
        local   E1
        movf    CNTAD,W
        bsf     STATUS,RP1
        bcf     STATUS,RP0
        movwf   EEADR
        bcf     STATUS,RP1
        bcf     STATUS,RP0
        movf    Dato,W
        bsf     STATUS,RP1
        bcf     STATUS,RP0
        movwf   EEDATA
        bsf     STATUS,RP0
        bcf     EECON1,EEPGD
        bsf     EECON1,WREN
        bcf     INTCON,GIE
        movlw   h'55'
        movwf   EECON2
        movlw   h'AA'
        movwf   EECON2
        bsf     EECON1,WR
        bcf     STATUS,RP0
        bcf     STATUS,RP1
E1
        btfss   PIR2,4
        goto    E1
        bcf     PIR2,4
        bsf     STATUS,RP0
        bsf     STATUS,RP1
        bcf     EECON1,WREN
        bsf     INTCON,GIE
        bcf     STATUS,RP0
        bcf     STATUS,RP1
        RETURN
```

- **Rutina para sacar información de la EEPROM**

```
EErd
        movf        CNTAD,W
        bsf         STATUS,RP1
        bcf         STATUS,RP0
        movwf       EEADR
        bsf         STATUS,RP0
        bcf         EECON1,EEPGD
        bsf         EECON1,RD
        bcf         STATUS,RP0
        movf        EEDATA,W
        bcf         STATUS,RP1
        RETURN
```

2.8. INTERFASE DE COMUNICACIÓN CON LA PC

La interfase de comunicación se la ha hecho en Visual Basic 3. Básicamente consiste en un programa de captura de la información que le envía el PIC al PC. Para esto tiene un botón que captura la información y la va ordenando en sus respectivos campos como son: el peso del producto, precio por unidad, precio total y también a que producto corresponde.

Figura 26: Pantalla del programa en Visual Basic

2.8.1. CODIFICACIÓN EN VISUAL BASIC

Programa elaborado en Visual Basic 3
Tienes que poner la configuración del puerto

Código del botón "Leer Puerto Serial",

```
Sub Command1_Click ()

ReDim mem(144) As String
ReDim memo(9) As String * 16
Dim aux, peso, pu, pt As String

Dim buferentrada As String
buferentrada = Comm1.Input
txtRecibir.Text = buferentrada
'txtRecibir.Text & buferentrada

aux = ""
k = 1
For i = 0 To 8
   For j = k To k + 15
aux = aux + Mid$(txtRecibir.Text, j, 1)
   Next j
   memo(i) = aux
aux = ""
   k = 16 * (i + 1) + 1
Next i

For i = 0 To 8
   grid.Row = i + 1
   peso = peso + Mid$(memo(i), 3, 4)
   grid.Col = 1
   grid.Text = peso
   pu = pu + Mid$(memo(i), 7, 4)
   grid.Col = 2
   grid.Text = pu
   pt = pt + Mid$(memo(i), 11, 6)
   grid.Col = 3
   grid.Text = pt
   peso = ""
   pu = ""
   pt = ""
Next i


End Sub
```

CAPÍTULO 3

3. CONCLUSIONES Y RECOMENDACIONES

En este capítulo se abarca el desarrollo de los circuitos impresos de los componentes de hardware de la balanza, así como las mejoras de diversos aspectos en la balanza entre otras. Finalmente exponiendo las conclusiones y observaciones realizadas en el trabajo.

3.1 SOBRE LOS CIRCUITOS IMPRESOS

Para el desarrollo de los circuitos impresos se utilizó el programa PROTEL 99, el cual permite crear y editar PCBs (Printed Circuit Board), como una ayuda para la creación del circuito impreso se tomó la simulación realizada para el circuito la cual se la realizó en el programa Circuit Maker, con eso estábamos seguros de los componentes que teníamos que incorporar al impreso.

Cabe decir que se hace uso de la opción de creación de los circuitos en forma automática, aunque obviamente en los lugares donde el programa no podía darle buen destino a la pista esta se la realizó manualmente.

Como observación final sobre este tópico aunque no es este el caso debido a que el circuito no es lo bastante grande, hay que indicar que la fabricación de un circuito impreso a doble cara en nuestro medio o sea la ciudad de Guayaquil no se lo puede hacer, pues los lugares donde los fabrican son empíricos y no tienen los recursos técnicos para la elaboración de los mismos con todas las características que los circuitos deberían poseer. En otros lugares como en Cuenca si se los puede realizar pero por falta de tiempo como de factibilidad no se lo pudo hacer.

3.2. MEJORAS A LA BALANZA.

Entre las mejoras que se podría incorporar a mediano plazo sería la colocación de otro MAX 232 que serviría para la comunicación de la balanza con algún tipo de impresora de tiques, pero por razones de tiempo y la falta de información al respecto se me fue imposible hacerlo por el momento.

También se ha estado pensando en que se podría modificar el diseño de la fuente de alimentación, para que esta sea una fuente conmutada que permitiría mayor estabilidad del voltaje al igual que serviría para incorporara numerosas funciones de encendido y apagado de la balanza.

Por último y como mejora de la presentación de los datos, sería la incorporación de un display LCD un poco más grande para tener mejor organizado los datos que se van a

presentar. Lastimosamente por la falta de recursos tanto económicos como también en lo que concierne a que en las electrónicas no se pueden conseguir estos dispositivos no me permitió llevarlo a cabo.

Para finalizar en lo concerniente al software este podría ser más optimizado ahorrando cantidad de memoria así como tiempo de proceso. Esto se lograría si funciones que a veces están redundantes se crearan más subrutinas que hagan el mismo proceso una y otra vez, pero esto queda para posteriores modificaciones que deseen hacerle al programa, debido a la versatilidad que poseen los microcontroladores PIC.

3.3. OBSERVACIONES

Una de las observaciones más importantes que deben tener en cuenta cuando se está programando un microcontrolador PIC y sobre todo si el programa que están creando es demasiado extenso es la paginación del mismo. Como se dijo en el capítulo 1 cuando se habló de la memoria del programa, esta memoria es de 8K distribuida en páginas de 2K cada una, cuando nosotros queremos hacer un salto a una subrutina que está en una página diferente al la que nosotros estamos trabajando las instrucciones CALL o GOTO solamente suministran los 11 primeros bits al PC para el direccionamiento del programa, es decir que nosotros nos podemos mover únicamente dentro de la página que estamos trabajando, los dos bits que faltan se los tiene que suministrar mediante programación al registro PCLATH que es el encargado de darle al PC la dirección del salto.

Ahora bien como se debe programar al PCLATH con los dos bits que falta, esto se logra mediante las siguientes instrucciones que voy a detallar a continuación:

Movlw HIGH con el nombre de la rutina a la que queremos saltar;

Movwf PCLATH; con esta instrucción se le está dando al PCLATH la dirección a la cual nosotros queremos saltar, y por último saltamos a la rutina a la cual deseamos sea esta con una instrucción CALL o GOTO.

CALL Nombre de la rutina a la cual saltamos. Después de hacer esto nosotros tenemos que poner el PCLATH de acuerdo a la página que estamos trabajando para esto debemos de controlar los bits <4,3> del PCLATH.

Esto es muy importante tener en cuenta al momento de la programación ya que si no causarían que el programa se cuelgue ya que no sabe el PC a donde saltar.

Otra cosa que debemos tomar en cuenta es cuando se trabaja con rutinas en punto flotante, estas poseen sus propias variables que se encuentran en un registro denominado MATH16.inc este archivo se tiene que introducir al programa mediante una instrucción de inclusión denominada INCLUDE, este archivo debe ser guardado en la misma carpeta del MPLAB para que lo pueda compilar el MPASM. Si nosotros queremos añadir nuevas variables para trabajar con estas rutinas, debemos hacerlo colocando las nuevas variables dentro del mismo archivo o sea dentro del MATH16 no dentro del programa que estamos creando ya que corren el riesgo de que estas variables comiencen a variar sin razón alguna con lo cual dichos cálculos no van a ser correctos.

Como se habrán podido darse cuenta existen otras instrucciones que maneja el ensamblador del PIC muchas de las cuales se logró conocer mediante numerosos programas ejemplo que se halló en INTERNET en la labor de investigación para la realización del proyecto de tesis.

Lo último que podemos decir es que cuando ustedes trabajen con tablas ya sea para conversión de valores en caracteres ASCII o para un display de 7 segmentos o como en mi caso para dar valor a las teclas pulsadas en el teclado tienen que fijarse que estas deber ser puestas como comienzo de las subrutinas ya que como es el registro PCL el que maneja el desplazamiento dentro de la tabla, al poseer este solamente 8 bits tiene que necesariamente estar en valores comprendidos entre 00H y FFH, al no ser así, este registro que tiene que ver con el contador del programa va ha adquirir el valor de 00H, de tal forma que al regresar de la subrutina que invocó a la tabla esta retornará al comienzo del programa con lo que sin razón aparente iniciará otra vez la ejecución del mismo.

3.4. CONCLUSIONES

En la realización de la tesis no se contó con mucha información respecto a ciertos aspectos importantes en su desarrollo, he abarcado dos campos importantes en mi formación profesional, el manejo de circuitos electrónicos y la programación. Se ha logrado una interacción eficiente entre el hardware y el software. Cabe destacar que las ideas básicas fueron tomadas de la bibliografía, pero el desarrollo en si de la tesis involucró mucha imaginación y creatividad para resolver diferentes problemas que se encontró en el camino, habiendo encontrado respuestas a muchas de ellas, sin embargo algunas quedaron pendientes. Descubrí algunos detalles que no estaban documentados en los textos de bibliografía o que no había alcanzado a revisar por falta de tiempo pero que con la experimentación se fueron despejando las dudas.

En el desarrollo de la balanza cabe destacar la parte electrónica, entre lo más importante es el circuito acondicionador de señal, que es la suma de pasos coherentes que se describen en la información proporcionada por la bibliografía. Incluyen la etapa de amplificación,

linealización, multiplexión y filtrado. Todas estas etapas se las llevaron a cabo exceptuando algunas de ellas ya que estaban hechas en el circuito propio del sensor que como bien se mencionó es una celda de carga. En el caso del sensor tenemos que por su alto costo no se pudo conseguir uno nuevo, por lo que se optó por uno que estaba en buen estado que se lo sacó de una balanza dañada, si bien nos da la información deseada, está no es tan exacta como hubiera querido, debido a pequeños cambios en la medida del sensor debido tal vez por alguna inestabilidad de la fuente de alimentación ya que si el voltaje cambia a la entrada esto provoca que cambie la medida del sensor y el PIC que posee un convertidor analógico-digital de 10 bits que es muy sensible, al haber pequeñas variaciones a la entrada del convertidor el valor que esta midiendo cambia y este se ve reflejado en el LCD.

Pero esto carece de importancia al quedar demostrado que con la ayuda de un microcontrolador y que con circuitería adicional y pocos integrados de apoyo se logró el resultado deseado.

ANEXOS

ANEXO 1

TABLAS DE LOS REGISTROS DEL PIC

REGISTROS DE FUNCIONES ESPECIALES

REGISTRO DE ESTADO (DIRECCIÓN 03h, 83h, 103h, 183h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit7							bit0

bit7: **IRP:** Bit de Selección del Banco de Registros (usado para direccionamiento indirecto)
1= Banco 2,3 (100h-1FFh)
2= Banco 0,1 (00h-FFh)

bit6-5: **RP1:RP0:** de Selección del Banco de Registros (usado para direccionamiento directo)
11= Banco 3
10= Banco 2
01= Banco 1
00= Banco 0
Cada Banco es de 128 bytes

bit4: **TO:** Bit de tiempo fuera
1= Después de encendido, instrucción CLRWDT, o instrucción SLEEP
0= Un tiempo fuera a ocurrido WDT

bit3: **PD:** Bit power-down
1= Después de encendido o por la instrucción CLRWDT
0= Por la ejecución de una instrucción SLEEP

bit2 **Z:** Bit Cero
1= El resultado de una operación lógica o aritmética es cero
0= Si el resultado de una operación lógica o aritmética no es cero

bit1 **DC:** Bit del dígito de acarreo o préstamo
1= Si a ocurrido un acarreo en el cuarto bit de orden bajo
0= Si no a ocurrido un acarreo

bit0 **C:** Acarreo o préstamo
1= Si a ocurrido un acarreo en el bit más significativo
0= Si no ha ocurrido un acarreo

Nota: Para préstamo la polaridad es reservada. Una sustracción es ejecutada por añadidura del complemento a dos del segundo operando. Para la rotación (RRF, RLF), este bit es cargado con un bit de orden bajo o alto del registro fuente.

REGISTRO DE OPCIONES (DIRECCIÖN 81h, 181h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit7							bit0

bit7 **RBPU:** Bit de habilitación de las resistencias Pull-up del puerto B
1= Pull-up deshabilitadas
0= Pull-up habilitadas

bit6 **INTEDG:** Bit de selección de interrupción por flanco
1= Interrupción en el flanco de subida del pin RBO/INT
0= Interrupción en el flanco de bajada del pin RBO/INT

bit5 **TOCS:** Bit de selección de la fuente de reloj para el TMR0
1= Transición en pin RA4/T0CKI
0= Instrucción del ciclo interno del reloj (CLKOUT)

bit4 **TOSE:** Bit de selección del flanco de la fuente de reloj del TMR0
1= Incremento en la transición de alto a bajo en pin RA4/T0CKI
0= Incremento en la transición de bajo a alto en pin RA4/T0CKI

bit3 **PSA:** Bit para asignar el preescalar
1= Preescalar es asignado a el WDT
0= Preescalar es asignado a el TMR0

bit 2-0 **PS2:PS0:** Bits de selección del preescalar

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

REGISTRO PIE1 (DIRECCIÓN 8Ch)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit7							bit0

bit7 **PSPIE(1):** Bit de habilitación para la interrupción para el puerto paralelo esclavo
1= Habilitación al PSP para la interrupción de la lectura/escritura
0= Deshabilitado el PSP

bit6 **ADIE:** Bit de habilitación de la interrupción del convertidor analógico/digital
1= Habilitación de la interrupción
2= Deshabilitada la interrupción

bit5 **RCIE:** Bit de habilitación de la interrupción para la recepción del USART
1= Interrupción de la recepción habilitada
0= Interrupción de la recepción deshabilitada

bit4 **TXIE:** Bit de habilitación de la interrupción para la transmisión del USART
1= Interrupción de la transmisión habilitada
0= Interrupción de la transmisión deshabilitada

bit3 **SSPIE:** Bit de habilitación de la interrupción Puerto Serial Sincrónico
1= Habilitada la interrupción del SSP
0= Deshabilitada la interrupción del SSP

bit2 **CCP1IE:** Bit de habilitación del CCP1
1= Habilitada la interrupción del CCP1
0= Deshabilitada la interrupción del CCP1

bit1 **TMR2IE:** Bit de habilitación de la interrupción del TMR2 a PR2
1= Habilitación de interrupción del TMR2 a PR2
0= Deshabilitada la interrupción del TMR2 a PR2

bit0 **TMR1IE:** Bit de habilitación de la sobrecarga del TMR1
1= Habilitada la interrupción
0= Deshabilitada la interrupción

REGISTRO PIR1 (DIRECCIÓN 0Ch)

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit7							bit0

bit7 **PSPIF(1):** Bandera de la interrupción del puerto esclavo paralelo
1= Una lectura o escritura se ha producido (debe ser limpiada por software)
0= No ha ocurrido ni lectura ni escritura

bit6 **ADIF:** Bandera de la interrupción del convertidor analógico/digital
1= Conversión completa
0= Conversión no completa

bit5 **RCIF:** Bandera de la interrupción de recepción del USART
1= Buffer lleno
0= Buffer vacío

bit4 **TXIF:** Bandera de la interrupción de la transmisión del USART
1= Buffer vacío
0= Buffer lleno

bit3 **SSPIF:** Bandera de la interrupción del Puerto Serial Sincrónico
1= La condición del SSP ha ocurrido y debe ser limpiado por software antes de retornar de la rutina de servicio de interrupción. Las condiciones de seteo de este bit son:
SPI
Una transmisión/recepción ha tomado lugar
I^2C Esclavo
Una transmisión/recepción ha tomado lugar
I^2C Maestro
Una transmisión/recepción ha tomado lugar
0= La condición de interrupción no ha ocurrido

bit2 **CCP1IF:** Bandera de interrupción CCP1
Modo de Captura
1= Un registro de captura ocurrió en el TMR1 (debe ser limpiado por software)
0= Registro de captura no ocurrió
Modo de Comparación
1= Registro de comparación ocurrió en el TMR1 (debe ser limpiado por software)
0= Registro de comparación no ocurrió
Modo PWM
Usado en este modo

bit1 **TMR2IF:** Bandera de interrupción TMR2 a PR2
1= TMR2 a PR2 ocurrió (debe ser limpiado por software)
0= No ocurrió

bit0 **TMR1IF:** Bandera de interrupción de sobrecarga del TMR1
1= Registro de sobrecarga (debe ser limpiada por software)
0= Registro no hay sobrecarga

REGISTROS ASOCIADOS CON EL USART

TXSTA: REGISTRO DE ESTADO DE TRANSMISIÓN Y CONTROL

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit7							bit0

bit7 **CSRC:** Bit de selección de la fuente de reloj
Modo Asincrónico
No implementado
Modo Sincrónico
1= Modo Maestro (Reloj generado internamente del BRG)
0= Modo Esclavo (Reloj de una fuente externa)

bit6 **TX9:** 9 bits bit de habilitación de transmisión
1= Selección de 9 bits de transmisión
0= Selección de 8 bits de transmisión

bit5 **TXEN:** Bit de habilitación de transmisión
1= Habilitación de transmisión
0= Transmisión deshabilitada

bit4 **SYNC:** Bit de selección del Modo del USART
1= Modo sincrónico
0= Modo asincrónico

bit3 **No implementado:** Lee como cero

bit2 **BRGH:** Bit de selección de la velocidad de transmisión
1= Alta velocidad
0= Baja velocidad

bit1 **TRMT:** Bit del registro de estado de la transmisión
1= TSR vacío
0= TSR lleno

bit0 **TX9D:** Dato con 9 bits de transmisión. Puede ser bit de paridad

RCSTA: RGISTRO DE ESTADO DE RECEPCIÓN Y CONTROL

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit7							bit0

bit7 **SPEN:** Bit de habilitación del Puerto Serial
1= Puerto Serial habilitado (Configurar RC7/RX/DT y RC6/TX/CK)
0= Puerto Serial deshabilitado

bit6 **RX9:** 9 bits bit de habilitación de recepción
1= Selección de 9 bits de recepción
0= Selección de 8 bits de recepción

bit5 **SREN:** Bit de habilitación de simple recepción
Modo Asíncrono
No habilitado
Modo Maestro Sincrónico
1= Habilitación de simple recepción
0= Simple recepción deshabilitado
Modo Esclavo Sincrónico
No usado en este modo

bit4 **CREN:** Bit de habilitación de recepción continua
Modo Asincrónico
1= Recepción continua habilitada
0= Recepción continua deshabilitada
Modo Sincrónico
1= Recepción continua habilitada hasta que el bit habilitado CREN es limpiado
0= Recepción continua deshabilitada

bit3 **ADDEN:** Bit habilitador de detección de dirección
Modo Asincrónico 9 bits (RX9=1)
1= Habilitada la detección de dirección, habilitado la interrupción y carga del buffer de recepción donde RSR <8> es seteado
0= Deshabilitada la detección de dirección, todos los bytes son recibidos, y el noveno bit puede ser usado com bit de paridad

bit2 **FERR:** Bit de error de trama
1= Error de trama
0= No hay error de trama

bit1 **OERR:** Bit de error de desbordamiento
1= Error de desbordamiento
0= No hay desbordamiento

bit0 **RX9D:** Noveno bit de recepción de datos (Puede ser bit de paridad)

FÓRMULA DE VELOCIDAD DE TRANSMISIÓN

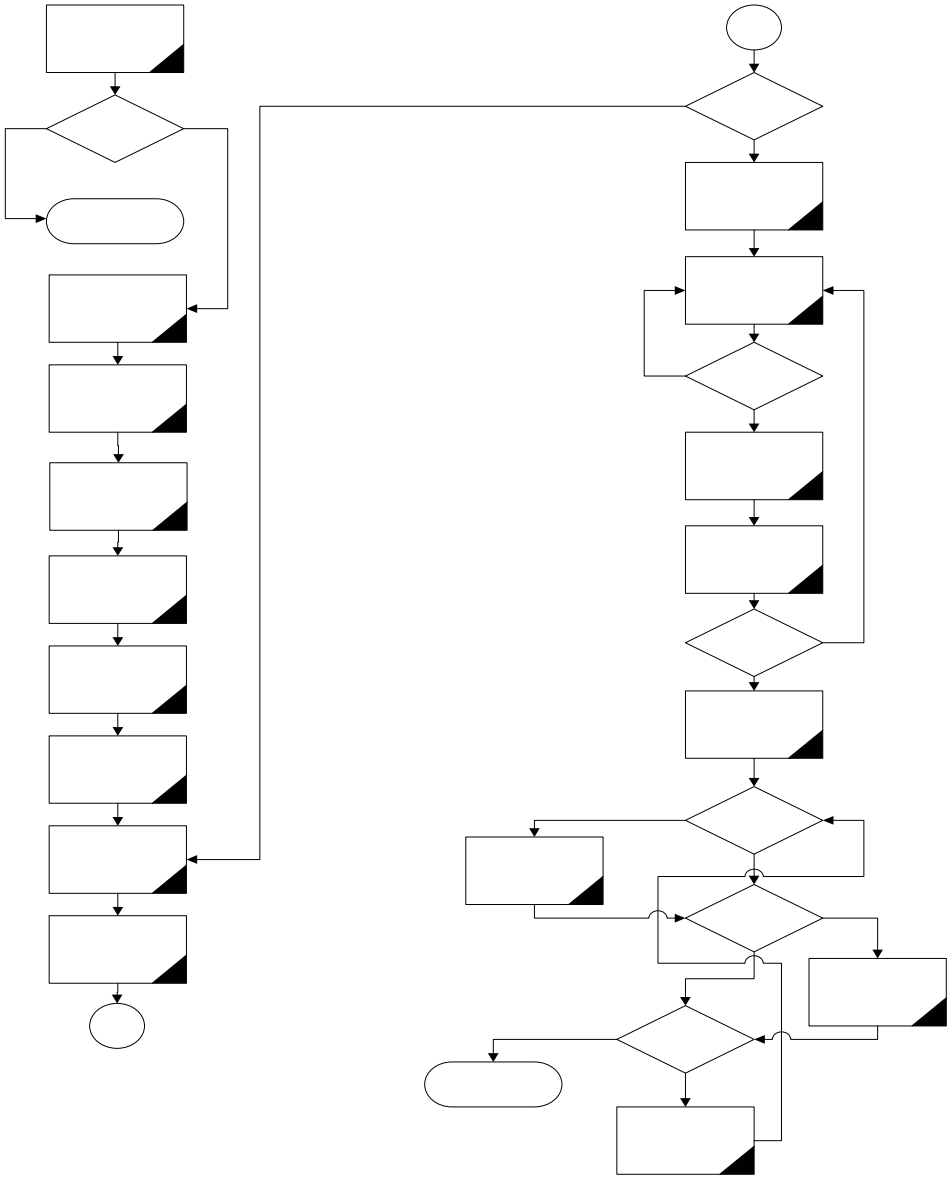
SYNC	BRGH=0 (BAJA VELOCIDAD	BRGH=1 (ALTA VELOCIDAD
0	(Asincrónico) baudios=Fosc/(64(x+1))	(Asincrónico) baudios=Fosc/(16(x+1))
1	(Sincrónico) baudios=Fosc/(4(x+1))	(Sincrónico) NA

X= valor en el registro SPBRG (0 a 255)

ANEXO 2

DIAGRAMAS DE FLUJO DE RUTINAS DEL PROGRAMA

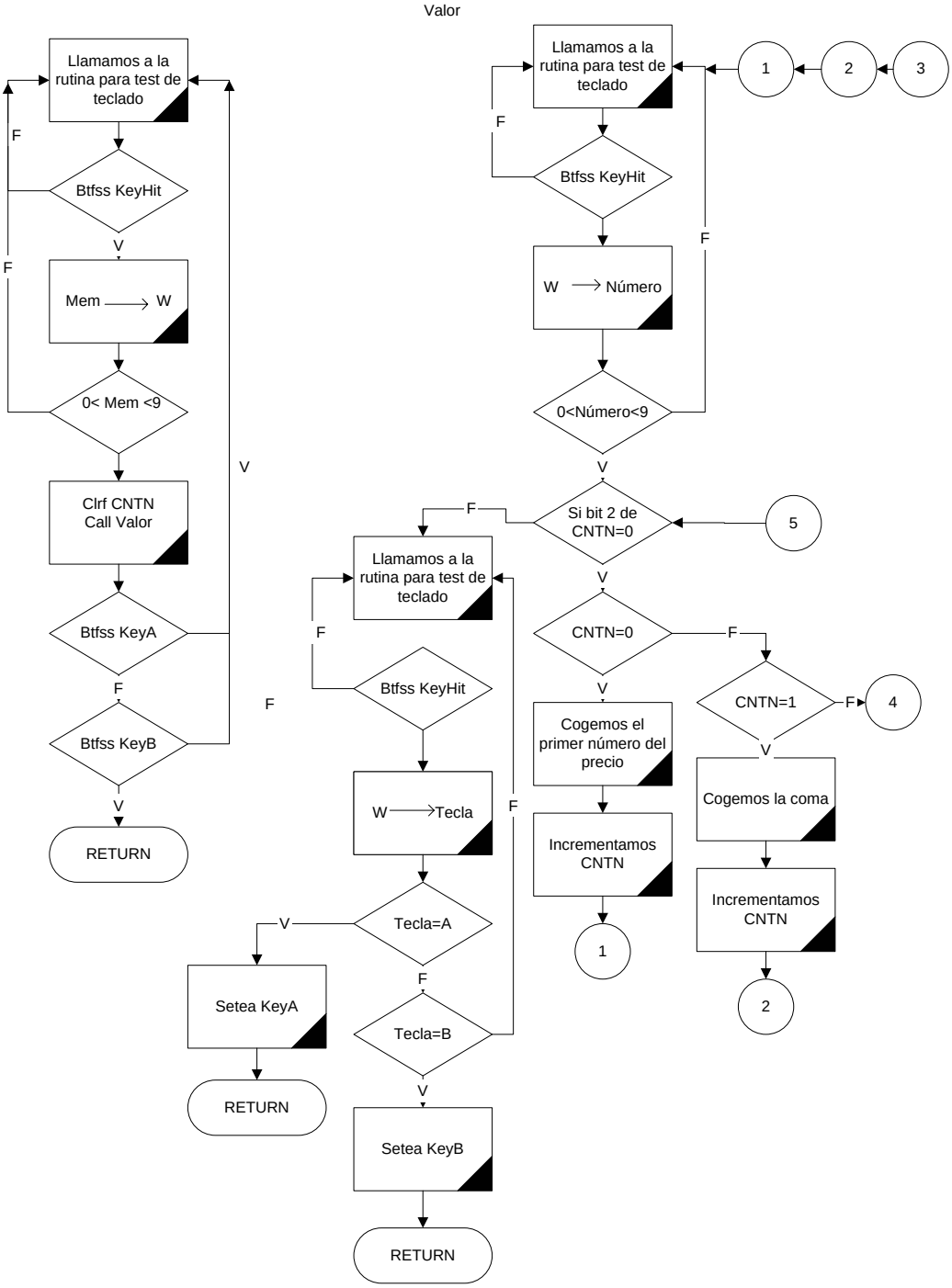
RUTINA DE PRESENTACIÓN DEL MENÚ

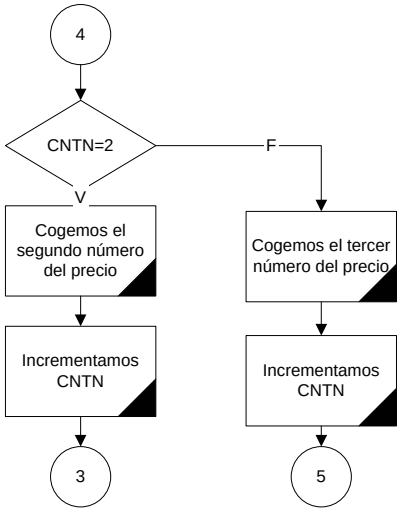


Limpia
bandera

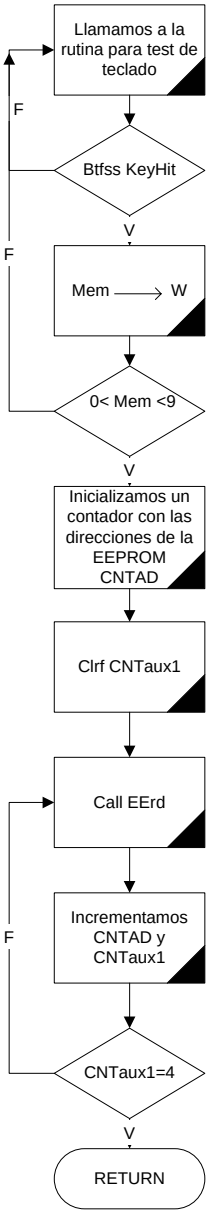
Po

RUTINA DE INTRODUCCIÓN DE PRECIOS

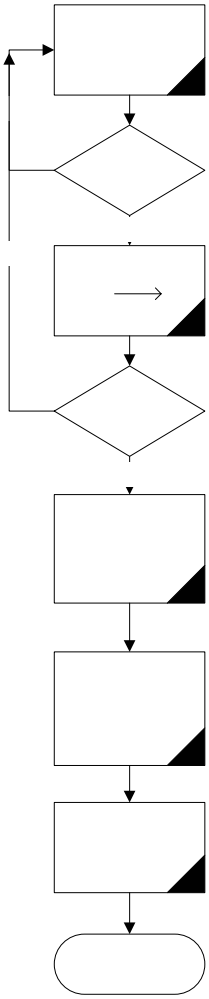



4
CNTN=2
F
V
Cogemos el segundo número del precio
Incrementamos CNTN
3
Cogemos el tercer número del precio
Incrementamos CNTN
5

RUTINA DE REVICIÓN DE LAS MEMORIAS



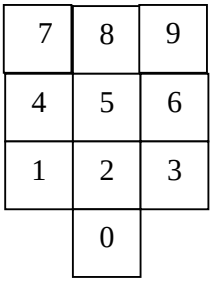
RUTINA DE CÁLCULO DEL PRECIO



ANEXO 3

MANUAL DEL USUARIO

DESCRIPCIÓN DEL TECLADO



Teclado Numérico: Sirve para la introducción de precios y memorias.

C	**Cálculo:**	Se utiliza para hacer el cálculo del precio
B	**Programación:**	Se utiliza para la programación de precios
A	**Anotación:**	Se utiliza para validar la información
D	**Serial:**	Se utiliza para mandar información vía RS232
*	**Menú:**	Sirve para la presentación de un pequeño Menú
#	**Memorias:**	Explora los precios introducidos en la memoria

INSTALACIÓN

Las balanzas de mostrador es necesario instalarlas en una superficie firme y horizontal, la puesta a nivel de la balanza, puede realizarse mediante las patas que esta posee.

Enchufar la balanza en una toma de red con conexión a tierra.

PUESTA EN MARCHA

Pulse el interruptor de red (lado derecho de la balanza) SIN PESO SOBRE EL PLATO.

Al ponerse en marcha la balanza aparecerá el indicador encerado para poder medir el peso (W=00,00 Kg) prendiéndose un LED rojo, quedando la balanza preparada para su utilización.

UTILIZACIÓN

Una vez prendida la balanza se tendrá que introducir los precios deseados en la memoria para esto se procederá de la siguiente manera:

Ejemplo: Si se desea introducir un precio de $1,50

1. Presione la tecla B para programar el precio.
2. Presione cualquier tecla del 1 al 9 para escoger la memoria en donde se programará el precio.
3. Presione el número 1
4. Presione cualquier tecla de numeral para que aparezca la coma
5. Presione el 5 y luego el 0
6. Presione A si quiere corregir el precio o si quiere guardar otro precio en otra localidad de memoria. Vuelva a repetir los pasos del 2 al 6
7. Presione B para salir de la programación de precios.

Ya programados todos los precios deseados se procede con el pesado de los productos. Coloque el producto a ser pesado en el plato de la balanza, espere un momento y saldrá el peso en el despliegue visual tanto en kilos como en libras, el LED se apagará para indicar que se puede realizar el cálculo del precio total.

Para calcular el precio total se procede de la siguiente manera:

1. Presione la tecla C para calcular el precio
2. Presione la tecla del 1 al 9 para sacar el precio de la localidad de memoria.
3. El precio total saldrá en el repliegue visual (P=00,000 $)
4. Para volver a calcular el precio sea con la misma localidad de memoria o con otra repita los pasos del 1 al 3.

Para visualizar el Menú presione la tecla "*" pero sin peso en el plato con esto se procederá a ver el menú guardado en memoria.

Para visualizar los precios guardados en memoria se procederá de la siguiente manera, esto se hará sin peso en el plato de la balanza.

1. Presione la tecla #
2. presione la tecla de 1 al 9 para ver la localidad de memoria
3. El precio saldrá en el despliegue visual
4. Espere un momento y volverá a su estado original listo para recibir otro peso.

Para la transferencia de los datos almacenados en memoria vía RS 232 se procederá de la siguiente manera:

1. Instale el cable serial tanto en la balanza como en la PC
2. Corra el programa en Visual Basic 3 en la PC
3. Escoja el puerto de comunicación con la PC (en el programa para la PC), se habilita la tecla LeerPuerto
4. Presione la tecla D en el teclado de la balanza
5. Presione la tecla A para enviar los datos
6. Presione la tecla LeerPuerto en el programa de la PC
7. Los datos se leerán en la PC.

CALIBRACIÓN

Para calibrar la balanza se procede de la siguiente manera:

1. Colocar la punta del multímetro en el pin 14 del integrado 2 de la tarjeta de acondicionamiento de señal.
2. Sin peso en el plato de la balanza primero se calibra el cero para esto se mueve la resistencia R12 (Potenciómetro de 500K) hasta conseguir que la lectura valga cero.
3. Colocando un peso conocido en el plato se mueve la resistencia R15 (Potenciómetro de 10K) hasta lograr que el voltaje sea el deseado. Para esto se da un listado de los posibles valores de voltaje para valores de pesos conocidos.

Cálculos de voltajes para el acondicionamiento de señal

Peso	V.celda	V.Entrada	R. Entrada	Binario	V. Entrada al PIC
0,0	0,0030	5,0030	350,420252	0	0,0000
0,1	0,0031	5,0031	350,438942	10	0,0488
0,3	0,0034	5,0034	350,476324	30	0,1465
0,6	0,0038	5,0038	350,532405	60	0,2930
0,9	0,0042	5,0042	350,588494	90	0,4395
1,0	0,0043	5,0043	350,607193	100	0,4883
1,2	0,0046	5,0046	350,644593	120	0,5859
1,5	0,0050	5,0050	350,700701	150	0,7324
1,8	0,0054	5,0054	350,756817	180	0,8789
2,0	0,0057	5,0057	350,794233	200	0,9766
2,1	0,0058	5,0058	350,812943	210	1,0254
2,4	0,0062	5,0062	350,869078	240	1,1719
2,7	0,0066	5,0066	350,925221	270	1,3184
3,0	0,0070	5,0070	350,981374	300	1,4648
3,3	0,0074	5,0074	351,037536	330	1,6113
3,6	0,0078	5,0078	351,093706	360	1,7578
3,9	0,0082	5,0082	351,149886	390	1,9043
4,0	0,0083	5,0083	351,168614	400	1,9531
4,2	0,0086	5,0086	351,206074	420	2,0508
4,5	0,0090	5,0090	351,262272	450	2,1973
4,8	0,0094	5,0094	351,318479	480	2,3438
5,0	0,0097	5,0097	351,355955	500	2,4414
5,1	0,0098	5,0098	351,374694	510	2,4902
5,4	0,0102	5,0102	351,430919	540	2,6367
5,7	0,0106	5,0106	351,487153	570	2,7832
6,0	0,0110	5,0110	351,543395	600	2,9297
6,3	0,0114	5,0114	351,599647	630	3,0762
6,6	0,0118	5,0118	351,655908	660	3,2227
6,9	0,0122	5,0122	351,712178	690	3,3691
7,0	0,0123	5,0123	351,730936	700	3,4180
7,2	0,0126	5,0126	351,768457	720	3,5156
7,5	0,0130	5,0130	351,824744	750	3,6621
7,8	0,0134	5,0134	351,881041	780	3,8086
8,0	0,0137	5,0137	351,918577	800	3,9063
8,1	0,0138	5,0138	351,937347	810	3,9551
8,4	0,0142	5,0142	351,993662	840	4,1016
8,7	0,0146	5,0146	352,049986	870	4,2480
9,0	0,0150	5,0150	352,106319	900	4,3945
9,3	0,0154	5,0154	352,162661	930	4,5410
9,6	0,0158	5,0158	352,219012	960	4,6875
9,9	0,0162	5,0162	352,275372	990	4,8340
10,0	0,0163	5,0163	352,294161	1000	4,8828

Donde el voltaje de la celada

$$V \cong \left(\frac{0.02}{15}\right) \times Peso + 0.03$$

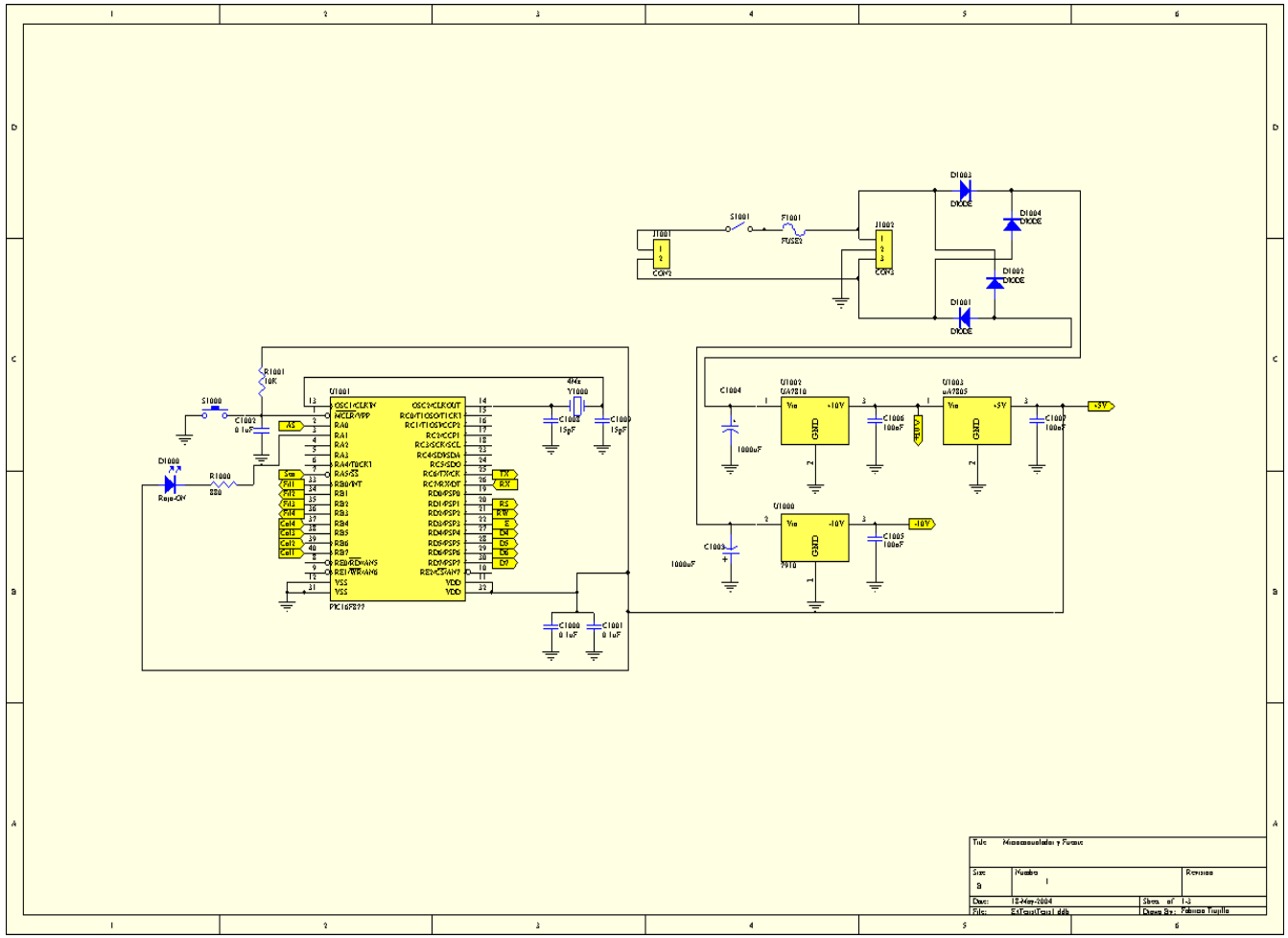
Voltaje de entrada al PIC

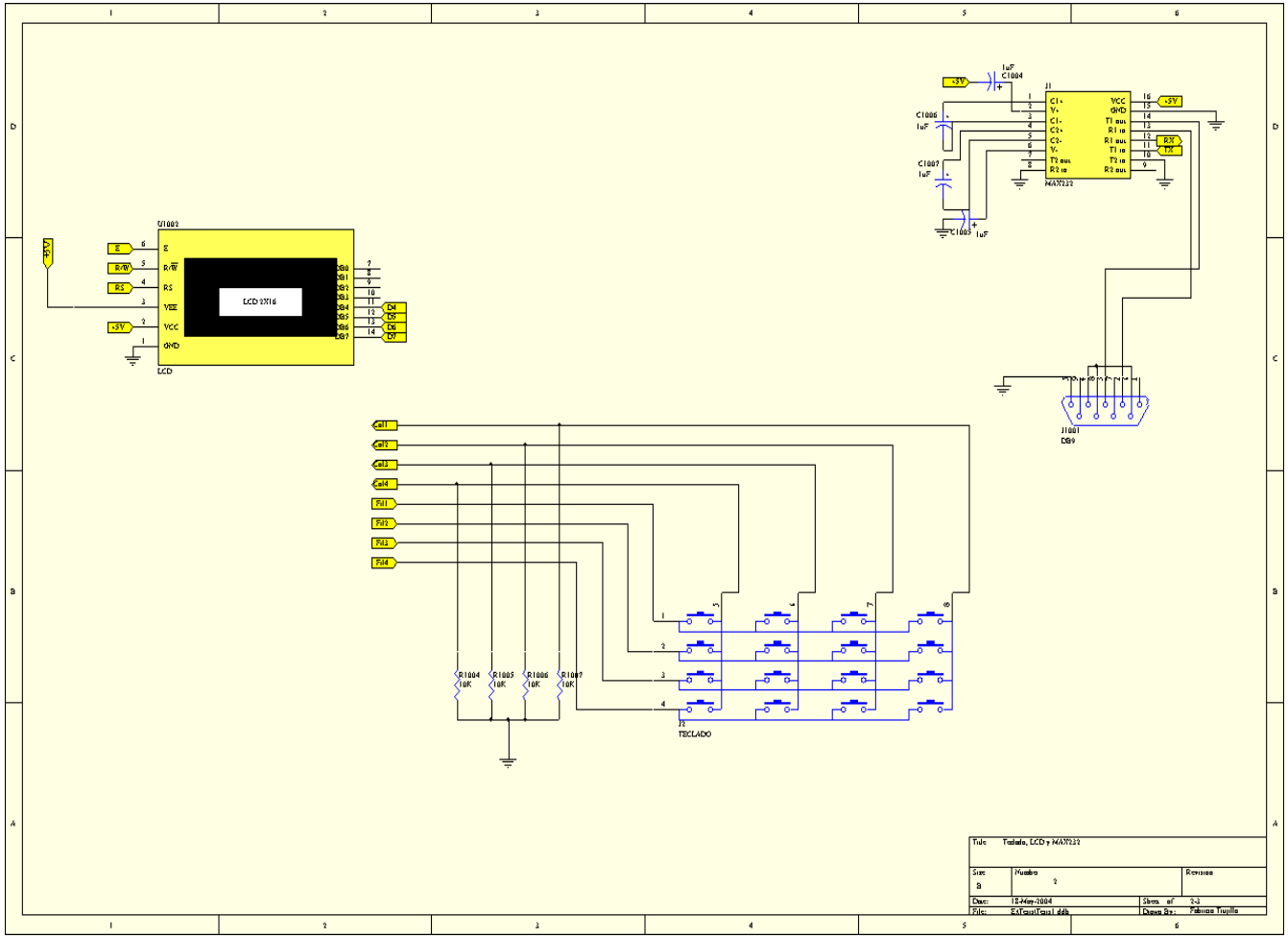
$$V \cong \frac{(Binario \times 5)}{1024}$$

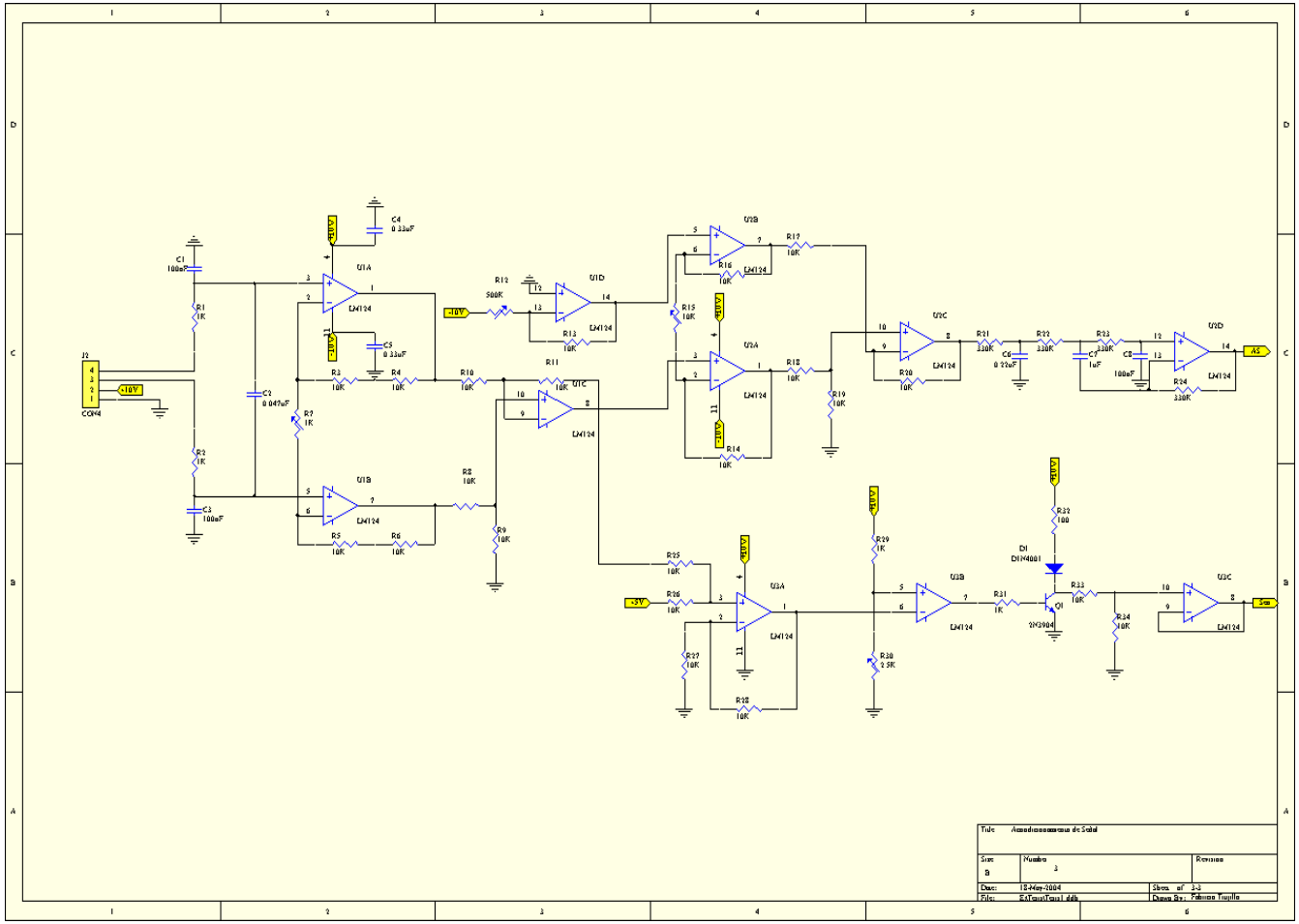
Con estas fórmulas podemos sacar valores para distintos pesos que no consten en la tabla.

ANEXO 4

DIAGRAMAS ESQUEMÁTICOS DE LA BALANZA

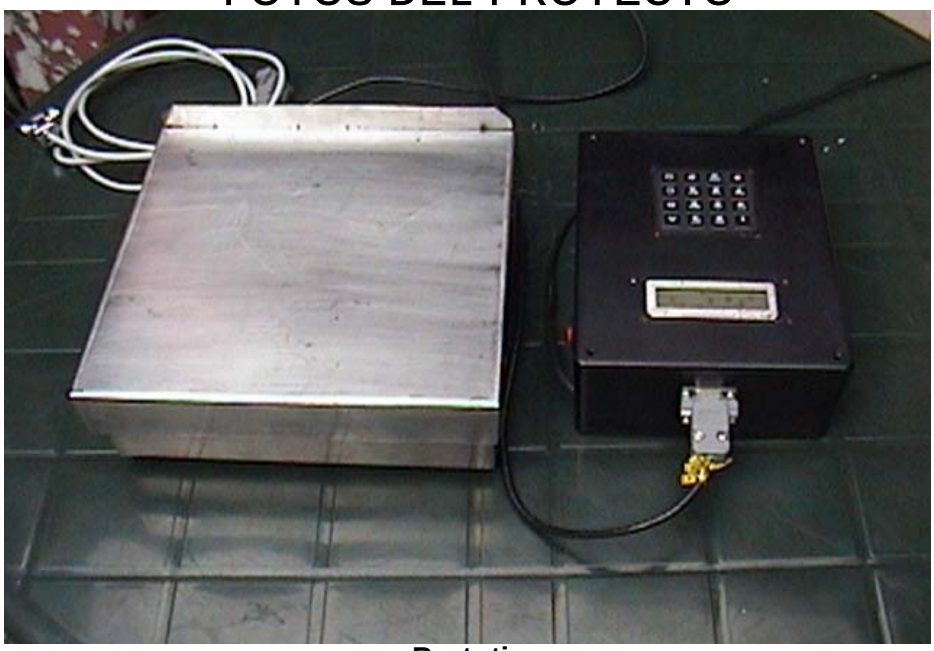

D1003
DIODE
D1004
DIODE
D1002
DIODE
D1001
DIODE
S1001
F1001
FUSE2
J1001
CON2
J1002
CON3
R1001
10K
S1000
C1002
0.1uF
D1000
Rojo-DIV
R1000
330
U1001
PIC16F877
4Mz
Y1000
C1002
15pF
C1003
15pF
C1000
0.1uF
C1001
0.1uF
C1004
1000uF
U1002
UA7810
+10V
C1006
100nF
U1003
uA7805
+5V
C1007
100nF
+5V
U1000
7910
-10V
C1005
100nF
C1003
1000uF
Title Microcontrolador y Fuente
Size B
Number 1
Revision
Date: 18-May-2004
Sheet of 1-3
File: E:\Tesis\Tesis1.ddb
Drawn By: Palomino Trujillo


U1002
LCD 1X16
LCD
+5V
E
R/W
RS
VEE
VCC
GND
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
D4
D5
D6
D7
C1004
1uF
C1006
1uF
C1007
1uF
C1005
1uF
J1
MAX232
C1+
V+
C1-
C2+
C2-
V-
T2 out
R2 in
VCC
GND
T1 out
R1 in
R1 out
T1 in
T2 in
R2 out
RX
TX
J1001
DB9
col1
col2
col3
col4
Fil1
Fil2
Fil3
Fil4
R1004
10K
R1005
10K
R1006
10K
R1007
10K
J1
TECLADO
Title
Teclado, LCD y MAX232
Size
B
Number
2
Revision
Date:
18-May-2004
Sheet of
2-3
File:
EsTeclaTest1.ddb
Drawn By:
Fabiana Trujillo


J2
CON4
-10V
C1
100nF
R1
1K
C2
0.047uF
R2
1K
C3
100nF
C4
0.33uF
U1A
LM124
C5
0.33uF
R3
10K
R4
10K
U1B
LM124
R5
10K
R6
10K
R8
10K
R9
10K
R10
10K
R11
10K
U1C
LM124
R12
500K
U1D
LM124
R13
10K
U2B
LM124
R15
10K
R16
10K
R17
10K
U2A
LM124
R14
10K
R18
10K
R19
10K
U2C
LM124
R20
10K
R21
330K
C6
0.22uF
R22
330K
R23
330K
C7
1uF
C8
100nF
U2D
LM124
R24
330K
AS
U3A
LM124
R25
10K
R26
10K
R27
10K
R28
10K
+5V
U3B
LM124
R29
1K
R30
2.5K
D1
DIN4001
R32
100
R31
1K
Q1
2N3904
R33
10K
R34
10K
U3C
LM124
Sos
Title
Acondicionamiento de Señal
Size
B
Number
3
Revision
Date:
18-May-2004
Sheet of
3-3
File:
EATemplate1.ddb
Drawn By:
Fabiana Trujillo

ANEXO 5

FOTOS DEL PROYECTO

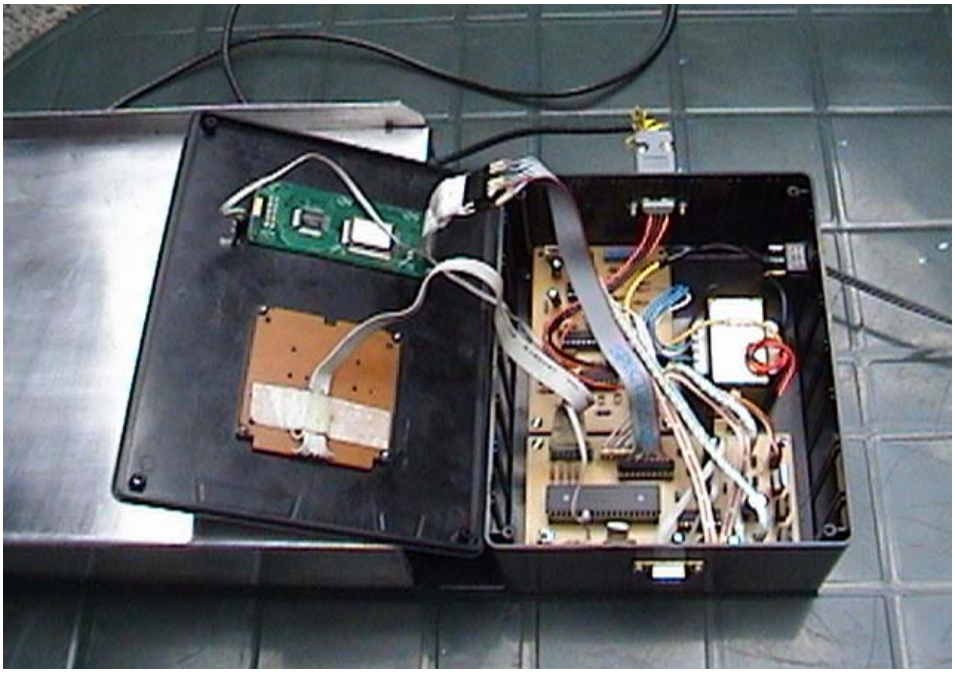


Prototipo

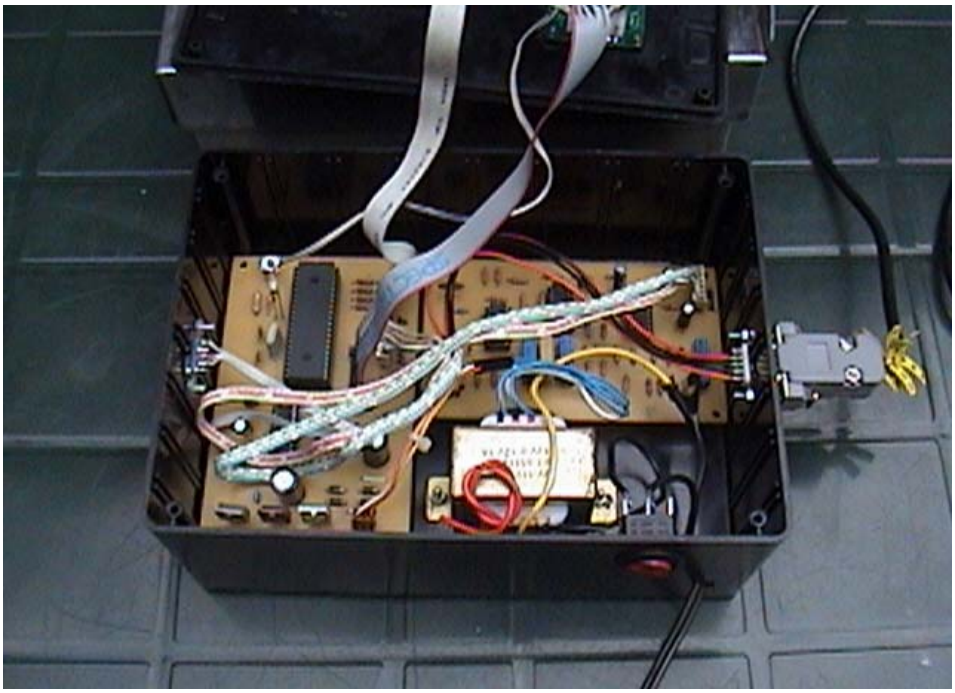
Teclado y Despliegue Visual (LCD)

Plataforma para el peso

Montaje del Sensor



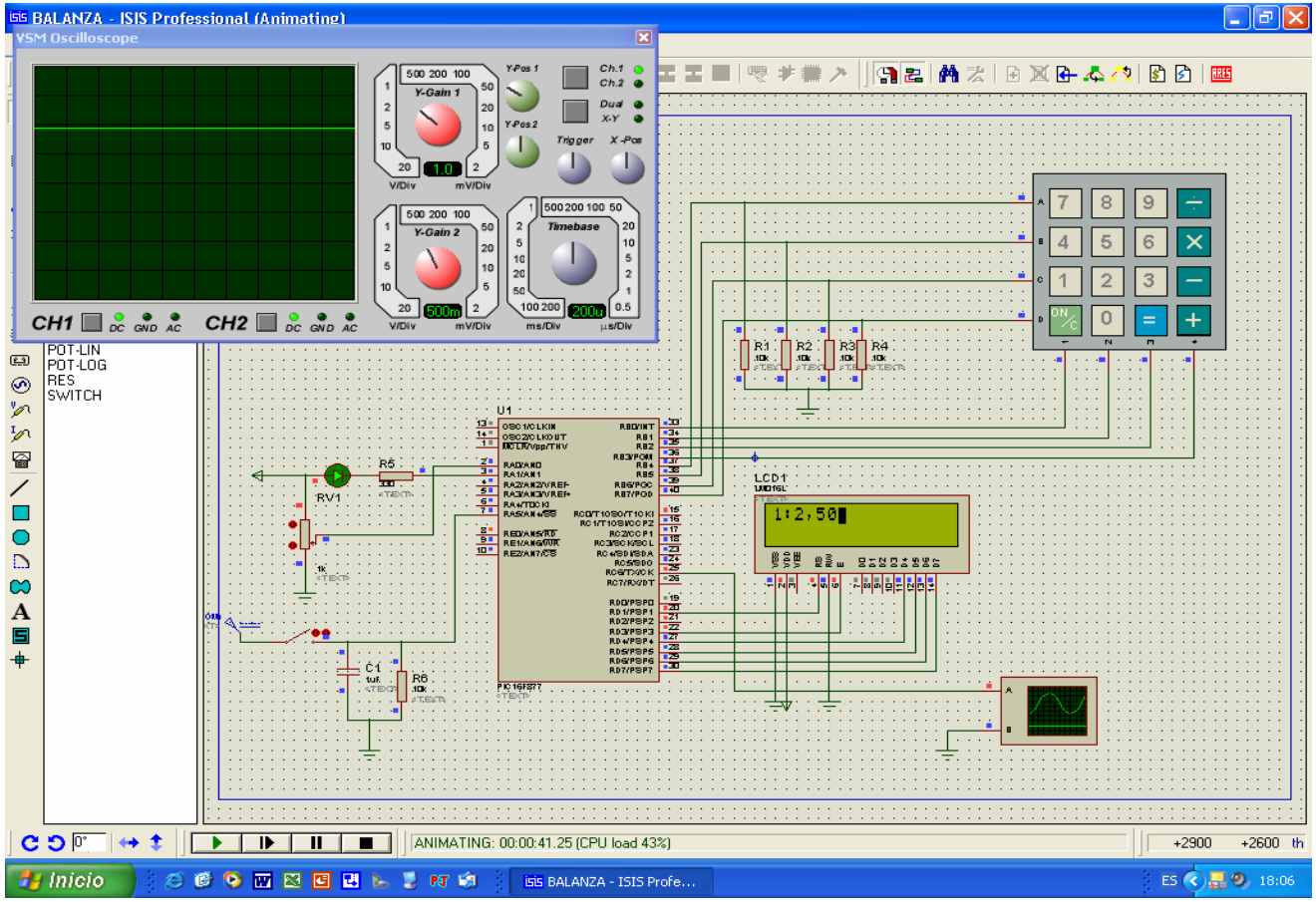
Tarjeta Principal con el Microcontrolador (PIC)



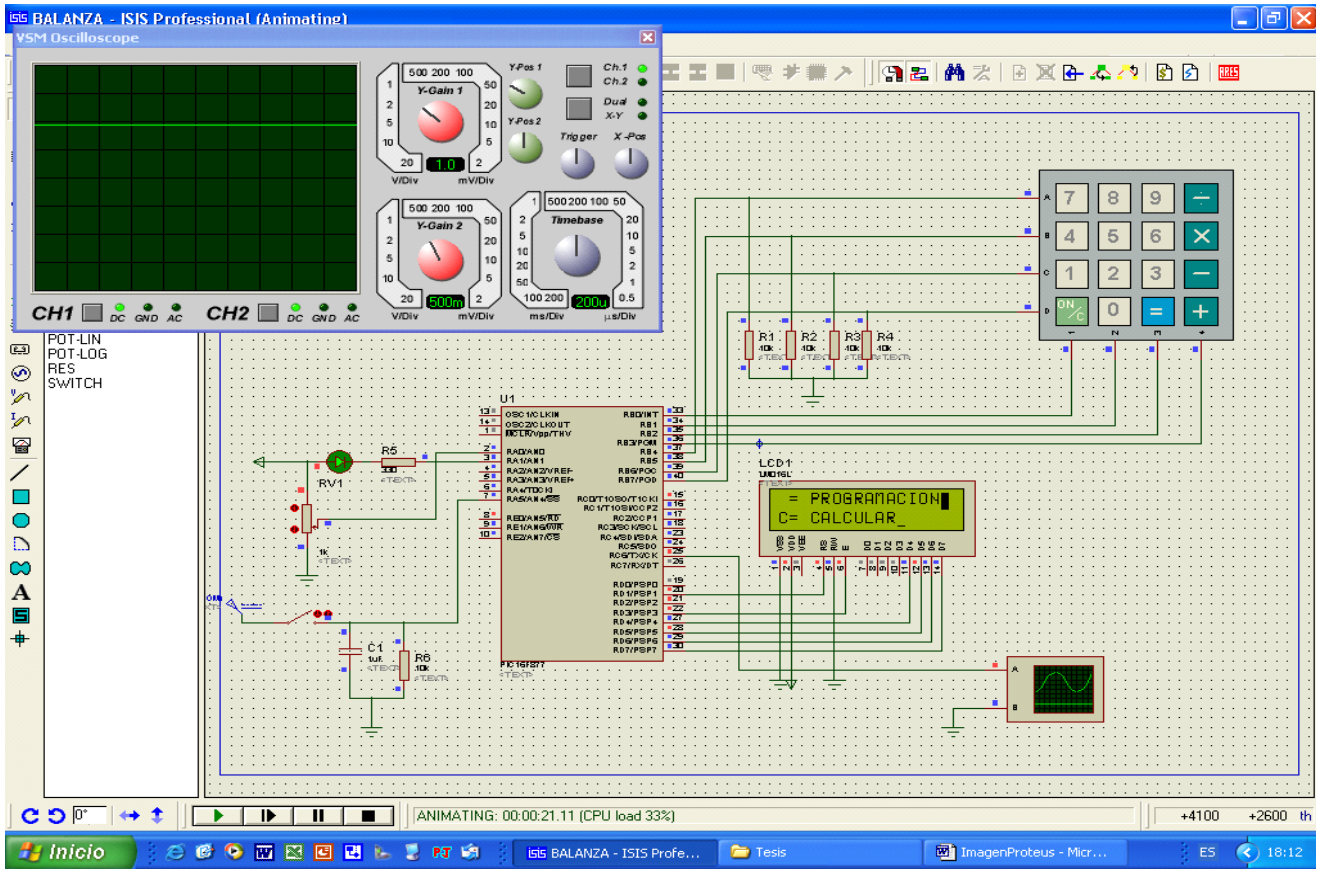
Tarjeta del Acondicionamiento de Señal

ANEXO 6

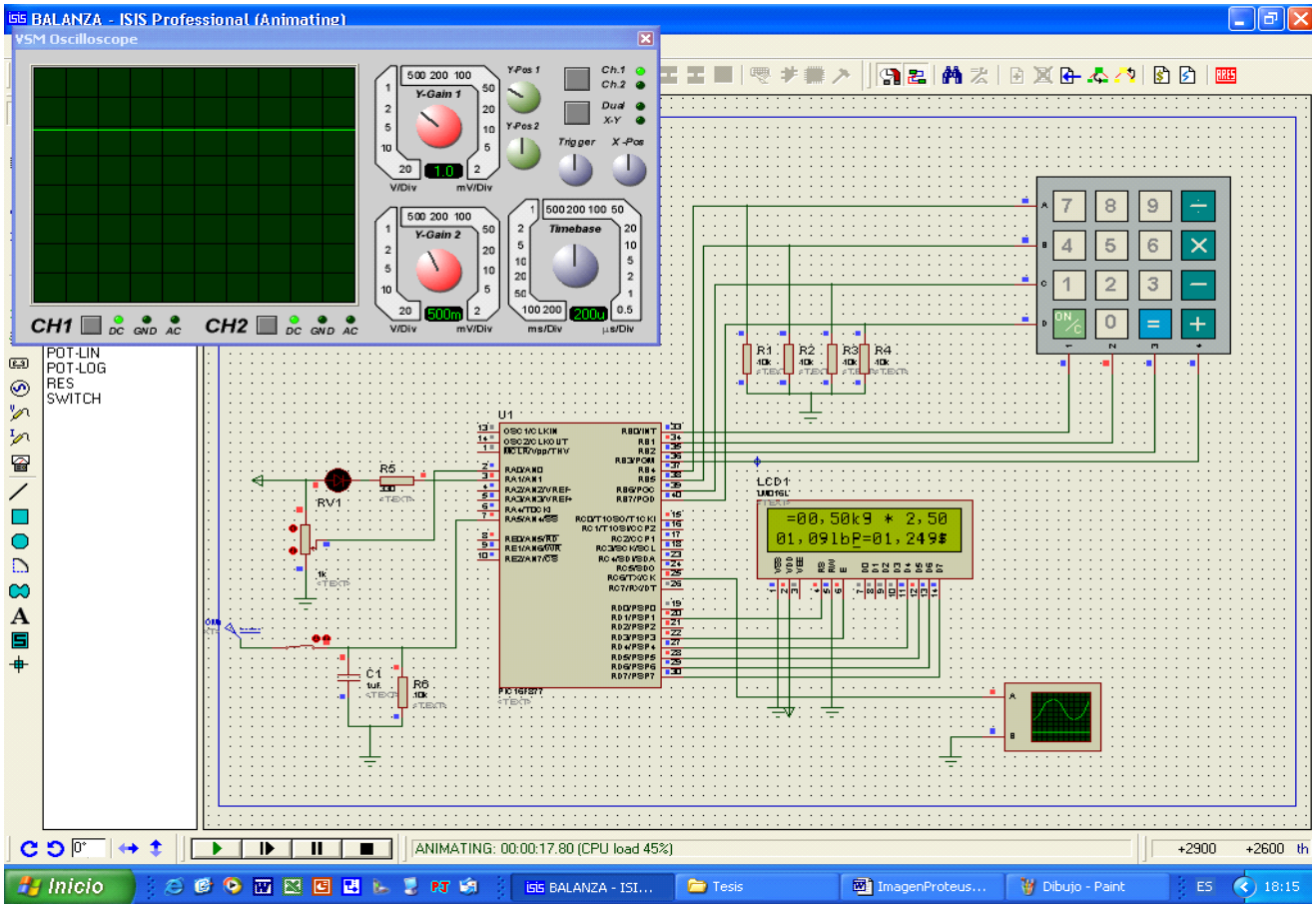
SIMULACIÓN EN PROTEUS



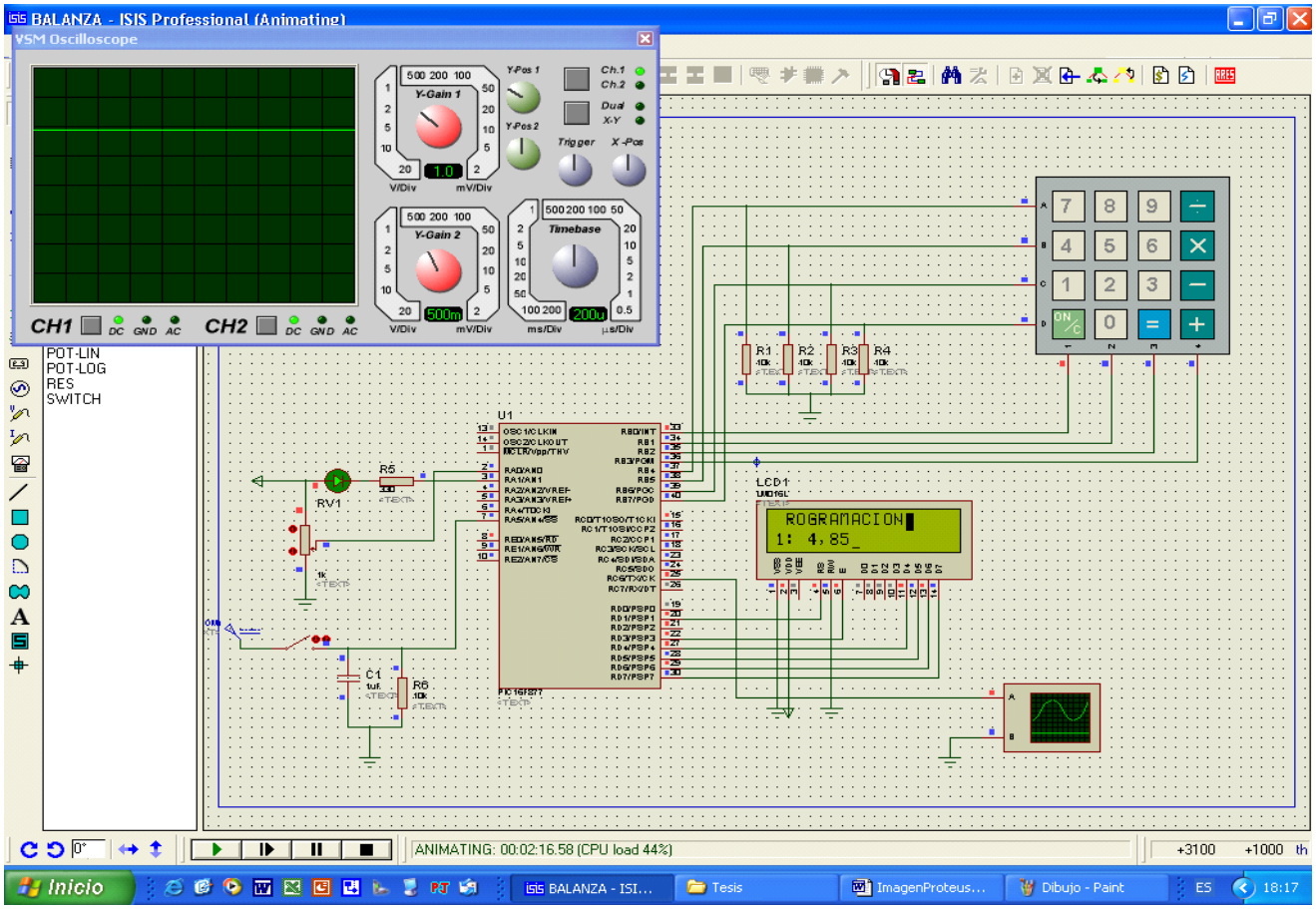
REVISANDO PRECIOS



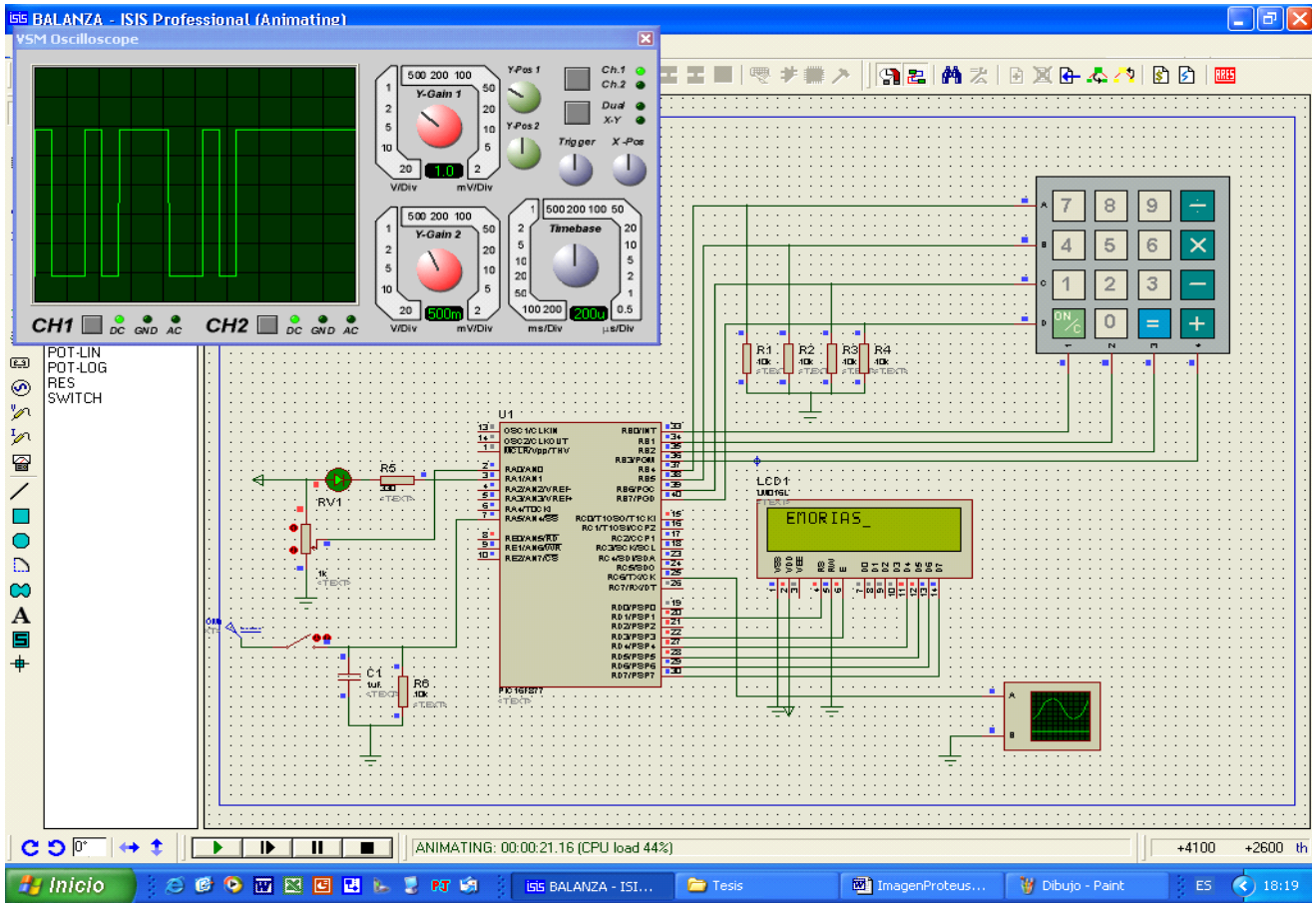
REVISANDO MENÚ



PESANDO Y CALCULANDO EL PRECIO



PROGRAMANDO UN PRECIO

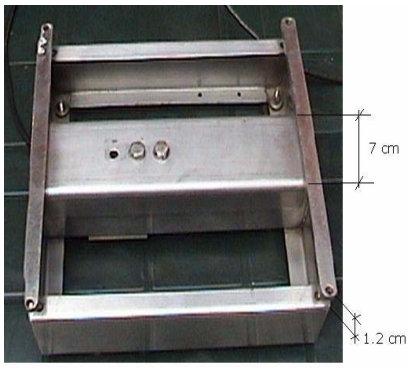


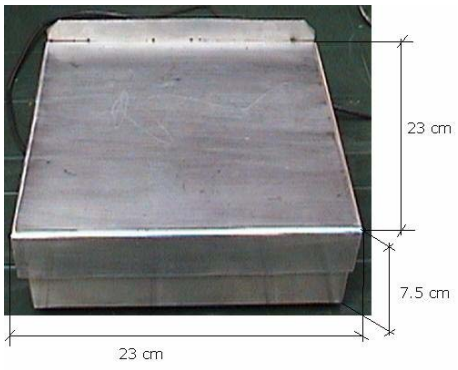
ENVIANDO INFORMACIÓN VIA MAX 232

ANEXO 7

DATOS TÉCNICOS DE LA BALNZA

Alimentación	120 Vac
Voltaje de la celda de carga	10 V dc
Voltaje del PIC	5 V dc
Voltajes de la tarjeta acondicionadora	±10 V dc
Impedancia de entrada de la celda	415 Ω
Corriente máxima del circuito	400 mA
Peso total de la Balanza	4,24 Kg





BIBLIOGRAFÍA

1. UNIVERSIDAD DE GUADALAJARA, Manual de Microcontroladores 1998
2. JOSÉ ADOLFO GONZÁLEZ, Introducción a los Microcontroladores, Grupo Editorial McGrawHill
3. ANGULO, Microcontroladores PIC Diseño de Aplicaciones, Grupo Editorial McGrawHill
4. A. SANZ Y J. I. ARTIGAS, Diseño de Filtros Activos, Universidad de Zaragoza, IEC
5. NATIONAL INSTRUMENTS, Tutorial Acondicionamiento de Señales
6. EAMAN NASH, A Practical Review of Common Mode and In-Amps, Analogic Device
7. www.microchip.com
8. www.ipn.mx
9. www.hitachi.com
10. www.measurementsgroup.com
11. www.datasheetarchives.com