Universidad Nacional Autónoma de México

Facultad de Ingeniería

Sistema de posicionamiento de placas fotosensitivas aplicado al proceso de revelado y ataque para el desarrollo de mascarillas

Tesis
Para obtener el título de
Ingeniero Eléctrico Electrónico
Área Control y Automatización

Presenta:
Cristina Toscano Coahuila

Director:
M. en I. Raúl Ruvalcaba Morales

Ciudad Universitaria, México, Abril 2013

Temario

Capítulo 1 Introducción

Algunos autores definen a la Microlitografía como un sistema de copiado, que imprime el diseño de circuitos de una mascarilla sobre un sustrato con película delgada depositada. Esta técnica es utilizada en la producción de máscaras, moldes, o guías para procesos posteriores de micro fabricación o para aplicaciones en líneas de investigación como: óptica, foto física, opto acústica, fotónica de microondas, etc.

En el grupo académico Fotónica de Microondas del Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET) de la Universidad Nacional Autónoma de México (UNAM), la técnica de microlitografía que se utiliza para la realización de micro dispositivos, es la microlitografía óptica. Para llevar a cabo esta tarea, este grupo académico cuenta con la siguiente infraestructura: Un cuarto limpio clase 100000, un Generador de Patrones, una lámpara de rayos UV, un equipo para depositar resina (spin coating), una tina de ultrasonido, una parrilla, agua destilada, aire comprimido y nitrógeno comprimido.

El Proceso para realizar una mascarilla para el desarrollo de microcircuitos en el CCADET es el siguiente:

- Diseñar el dispositivo.
- Determinar tanto el tipo de material del sustrato como el de la película.
- Dibujar el dispositivo en AutoCad y guardarlo con formato DXF (formato que reconoce el Generador de Patrones Mod. EM-5009B).
- Generar la mascarilla del dispositivo en el Generador de Patrones, como sigue:
    - Cargar el archivo DXF en el sistema de Control del Generador de Patrones.
    - Convertir el archivo DXF a un archivo MUL (con el software del Sistema de Control del Generador de Patrones).
    - Imprimir el dibujo en la Placa Foto Sensitiva (vidrio con óxido de hierro y resina).
- Pre hornear entre 80°C y 100 °C por un tiempo entre 5 y 10 minutos.
- Revelar la placa foto sensitiva.
- Enjuagar la placa foto sensitiva con agua destilada.
- Observar en un microscopio la placa revelada, (si se obtiene un buen resultado se pasa al siguiente paso, en caso contrario se puede volver a revelar).
- Hornear entre 80°C y 100 °C por un tiempo entre 5 y 10 minutos.
- Atacar con ácidos la Placa Foto Sensitiva.
- Enjuagar con agua tridestilada.
- Verificar con un microscopio el resultado de la mascarilla.

El presente trabajo tiene como objetivo mostrar el diseño e implementación de un sistema para la automatización del proceso de microlitografía en las etapas de revelado-enjuague y ataque-enjuague.

Con el sistema desarrollado se pretende lograr un mejor control de los tiempos de revelado, una menor manipulación de las placas por el usuario y mayor protección contra agentes

externos, con esta propuesta se harán más eficientes y de fácil manipulación los procesos de revelado y ataque.

La propuesta de diseño tiene las siguientes etapas:

- Sistemas de control (microcontrolador PIC18F452), controla el tiempo y el menú del equipo).
- Fuentes de alimentación lineales (5Vdc, 10Vdc).
- Sistema mecánico (porta placa, soportes, espiga, balero, etc.).
- Sistema de depósitos de sustancias (Recipientes de vidrio para el revelado y el ataque).
- Estructura de acrílico para aislar las partículas del medio ambiente a la placa y para evitar la dispersión los gases tóxicos que se desprenden del revelador y/o de mezcla de soluciones ácidas.

El diseño del sistema por bloques es mostrado en la Figura 1.1

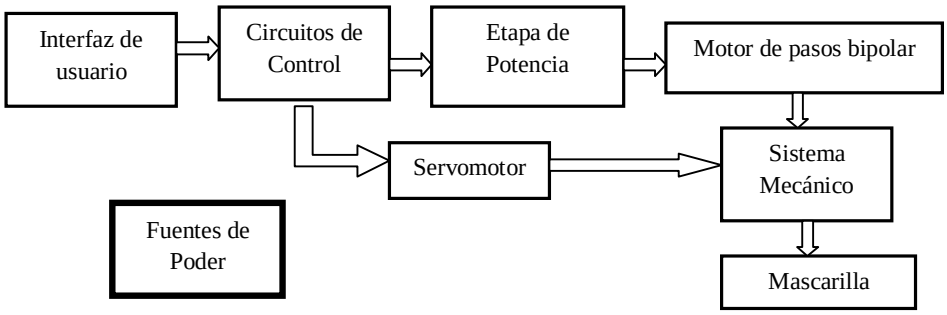


Figura 1.1

El porta placas del sistema mecánico, esta fabricado en teflón, esto debido a su alta resistencia a la corrosión, mientras que las demás piezas están fabricadas en aluminio y latón.

El sistema de control realiza los desplazamientos verticales y horizontales de la placa para pasar de una etapa a otra (revelado-enjuague o ataque-enjuague). Los movimientos horizontales son realizados con un servomotor analógico que mueve la estructura con ayuda de un balero, este servomotor es controlado con un PWM (pulse with modulation). Por otra parte, para el movimiento vertical, se realiza por medio de un motor a pasos bipolar que, acoplado con una espiga y tomando como guía cuatro postes, mueve la placa hacia arriba o hacia abajo, este motor tiene una secuencia de pasos establecida para moverse, pero su velocidad la determina el tiempo en el que está dada cada secuencia. Las señales para el control de ambos motores, están dadas por el microcontrolador PIC18F452.

La Figura 1.2 muestra el diagrama de flujo del programa que se desarrollo para el equipo de revelado-ataque.

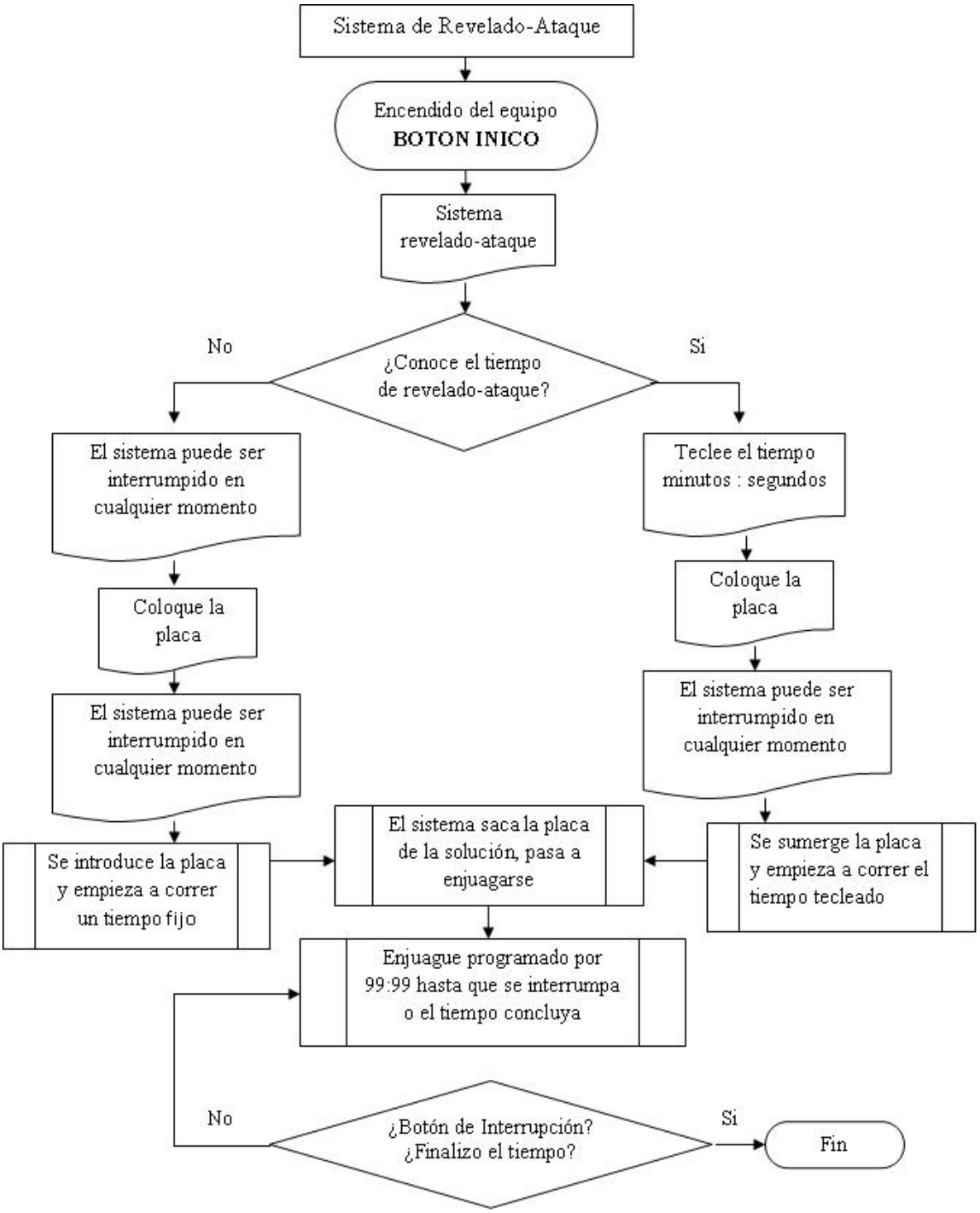


Figura 1.2

Capítulo 2. Antecedentes

2.1 Microlitografía

La microlitografía permite el grabado de figuras y/o patrones en un material fotosensible, cambiando sus propiedades químicas al ser expuesto ante radiación ultravioleta, esta técnica es utilizada en la industria de los semiconductores.

Algunas técnicas de grabado litográfico son:

- El sistema de nanolitografía por dip-pen (escritura directa) se compone de un microscopio de fuerza atómica (Atomic Force Microscope, AFM), una cabina de control ambiental (temperatura y humedad relativa) y el software de control adecuado para realizar nanolitografía. El proceso de dip-pen permite depositar compuestos como biomoléculas, resinas, polímeros y otros materiales con resolución nanométrica y sin necesidad de mascarillas.

- El proceso de hot embossing (estampado caliente) se utiliza para replicar estructuras micrométricas sobre polímeros (Figura 2.1). Se realiza en una prensa en la que la temperatura y la presión aplicadas se controlan.

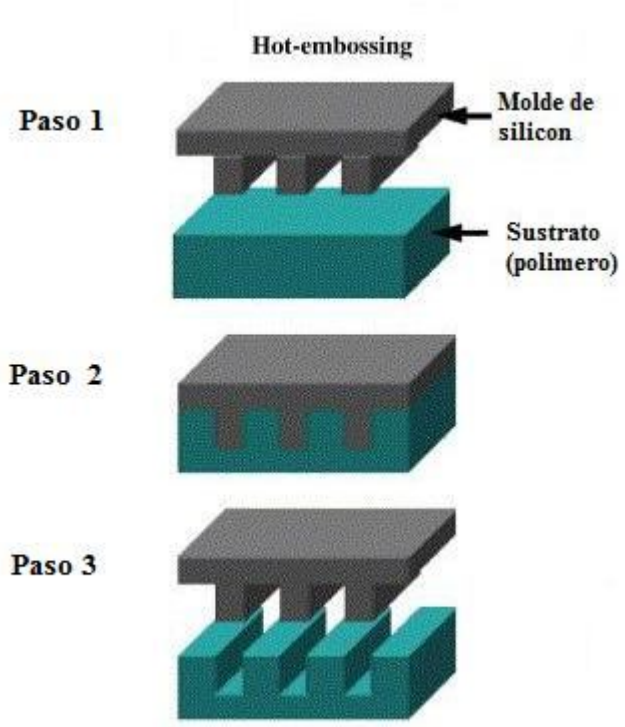


Figura 2.1

- Litografía óptica, actualmente esta técnica ha sido llevada al extremo, se está utilizando luz ultravioleta en el rango de λ=248nm para imprimir características de 150-12nm. En los siguientes años se tiene planeado hacer chips con características de (100-70)nm usando luz ultravioleta profunda (deep ultraviolet, DUV) con λ= (193-157) nm, ver Figura 2.2. Para hacer grabados más pequeños se requerirá una longitud de onda en el rango del ultravioleta

extremo (Extreme ultraviolet EUV), la luz a estas longitudes de onda es absorbida en lugar de transmitida por las lentes convencionales, el resultado sería no luz y no imagen. Pronto se tendrá que decidir cómo hacer las siguientes generaciones que micro circuitos, algunas posibilidades son descritas a continuación.

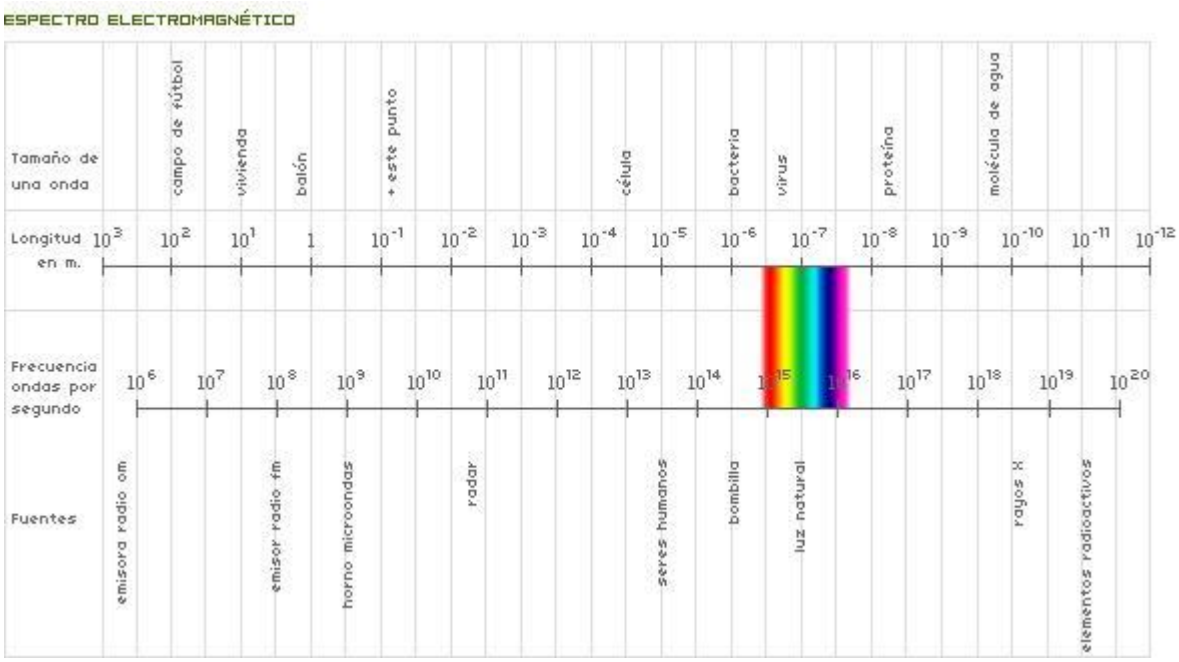


Figura 2.2.

- Litografía de Ultravioleta extremo (Extrem Ultraviolet EUV), tiene una longitud de onda λ=13.5nm, la fuente de este ultravioleta es un sistema de Laser Produce Plasma (LPP). LPP utiliza un laser de alta potencia para crear un plasma de alta energía que emite luz de longitud de onda corta en el interior de una cámara de vacío. Utiliza espejos con revestimiento llamados multilayer mirrors (MLM), pero aun estos espejos especializados absorben alrededor del 30% de luz, por lo que es recomendable no utilizar muchos espejos.

- X-Ray Litografía utiliza longitud de onda de (0.4-4)nm, tiene una resolución muy alta, las características más pequeñas que se consiguen son de (10-20)nm, puede utilizar resina positiva o negativa, no presenta difracción significativa y el tiempo de vida de una mascarilla realizada con esta técnica es mayor (Figura 2.3). Sin embargo esta técnica es muy costosa, ya requiere sustratos de oro o tugsteno con recubrimiento de carburo de silicio o diamante.

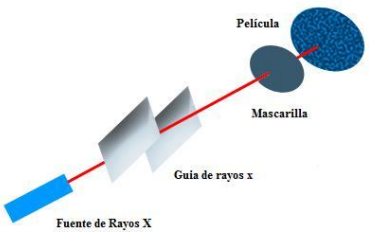


Figura 2.3

- Electron-Beam Litografía es una técnica que emplea un haz enfocado de electrones para una extrema precisión en la generación de patrones. El patrón es grabado directamente sobre la resina electro sensitiva (no usa mascarilla). La característica más pequeña que puede generar es de 5nm. Toma alrededor de 10 horas grabar sobre una oblea, un sistema de Electron-beam cuesta entre 5 y 10 millones de dólares.

- Ion Beam Litografia, consiste en el bombardeo del material con iones que generalmente son de galio, hidrógeno o helio para realizar grabados y depósitos en la superficie del material, los iones son acelerados por un voltaje que está entre 0.5 a 50 kV generando de esta forma una corriente de haz sobre la muestra de 1pA a 20nA. Los iones de galio son disparados con una energía de 1 a 3.5 Mev, para penetrar o pulverizar el material, los protones que tienen energía de 3.5 Mev pueden penetrar 160 μm del material, a demás de esto se inyecta un gas el cual es un atacante químico del material (fotoresina), disminuyendo de esta forma el tiempo del proceso.

- En el Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET) la técnica que se utiliza para realizar microcircuitos es la litografía óptica, con un laser en el rango ultravioleta, descrito con mayor detalle a continuación.

2.2 Microlitografía óptica en el CCADET

La litografía óptica es un proceso en el cual se transfiere luz a través de una mascarilla con el patrón del circuito deseado, sobre un material con película fotosensible que reacciona químicamente cuando es expuesto a radiación ultravioleta.

La resolución máxima posible con esta técnica está limitada por la naturaleza (λ) de la luz. Para obtener detalles de menor tamaño, se deben utilizar longitudes de onda más pequeñas, lo que incrementa el costo de fabricación. Además, la luz tiene un límite de difracción que restringe el grado de miniaturización del espacio sobre el que se puede enfocar la luz.

2.2.1 Mascarilla

El primer paso en la técnica de microlitografía óptica es realizar la mascarilla, que servirá como molde para futuras replicas y en distintos materiales. La mascarilla es grabada en una placa de vidrio con película de óxido de hierro y deposito de resina fotosensible positiva o negativa, según el diseño requerido.

La mascarilla en el CCADET es grabada con un Generador de Patrones EM-5009B marca Plasma (Figura 2.4) que tiene un láser con una longitud de onda de λ=0.337nm (ultravioleta), con una potencia de 240-260mW. El laboratorio donde opera el Generador de Patrones tiene un sistema de extracción de gases, para reducir la densidad de ozono liberado durante la operación del láser, un sistema de enfriamiento a través de agua y aire comprimido y un filtro rojo para evitar que las mascarillas se revelen. El generador de patrones es un sistema con las siguientes unidades funcionales:

- Sistema de control
- Sistema de posicionamiento
- Sistema óptico

- Sistema de iluminación (laser)
- Fuentes de alimentación

Figura 2.4

Una vez terminado el proceso de grabado, la mascarilla es guardada en una caja negra y transportada al cuarto limpio para proceder a revelarla.

Para revelar la mascarilla se debe utilizar el revelador de acuerdo a la resina que se utilizó, es decir revelador positivo o negativo. En el CCADET se utiliza la marca alemana "micro resist tecnology".

Se introduce totalmente la mascarilla en el revelador y se van observando los cambios de luz que va presentando el grabado para saber cuándo retirar la mascarilla. Es muy importante tener habilidad en este proceso porque al ver que la mascarilla esta lista se debe retirar inmediatamente y enguada en agua tridestilada, para su posterior revisión en el microscopio, si se considera que aun falta revelar se puede volver a introducir la mascarilla en el revelador. El proceso de revelado está determinado por el espesor inicial de la resina, las condiciones del pre horneado, composición química del revelador y por el tiempo de revelado.

Después del revelado se pre hornea la mascarilla por 10 minutos a una temperatura entre 80 y 100 °C, este pre horneado se hace en una parrilla con una superficie plana, para que el pre horneado sea uniforme. Este proceso es necesario para polimerizar, esto es, endurecer la fotoresina, mejorar su adherencia y para que la resina resista el ataque ácido. El horneado elimina trazas que quedan de la resina o revelador e introduce tensiones en la fotoresina. Mayor temperatura hace que la resina se elimine con mucha mayor dificultad.

El siguiente paso es atacar la mascarilla, en el caso de una placa de vidrio con óxido de hierro se utiliza se utiliza la siguiente mezcla:

- 20g de cloruro férrico $FeCl_3$
- 300ml de ácido clorhídrico *HCL*
- 20g de ioduro de potasio *KI*
- Agua Regia = ácido nítrico + ácido sulfúrico = HNO_3+H_2SO_4

El tiempo de ataque por lo general es menor al tiempo de revelado. Al terminar el ataque la mascarilla se enjuagada con agua tridestilada y secada con aire comprimido. De esta forma queda terminada la mascarilla y se puede proceder a hacer la transferencia del patrón sobre distintos materiales con recubrimiento de fotoresina.

2.2.2 Transferencia de la mascarilla

Una vez seleccionado el sustrato donde se transferirá la mascarilla, se debe realizar una limpieza del sustrato para eliminar los contaminantes de la superficie.

Los contaminantes pueden suponer una imperfección en el diseño por muy pequeños que sean estos, pueden afectar en la deposición de la resina o en la transferencia. Algunos de los contaminantes pueden ser:

- Partículas de sistemas mecanizados.
- Partículas del ambiente (uso de Cuarto Limpio).
- Hilos de paños.
- Residuos de fotoresina.
- Residuos de solventes, agua, reveladores, aceite, etc.

Para limpiar grasa (por mínima que sea) u otros contaminantes es necesario limpiar el sustrato en una tina de ultrasonido con acetona, etanol y agua destilada, 5 minutos por cada una de las sustancias. Y posteriormente secar con aire comprimido (de preferencia nitrógeno comprimido), para poder depositar la resina.

El depósito de fotoresina sobre el sustrato se lleva a cabo con un equipo de depósito por giro (Spin Coatig), el proceso consiste en la deposición de una pequeña cantidad de resina en el centro del sustrato con ayuda de una jeringa y posteriormente hacer girar la placa o sustrato (Figura 2.5). La cantidad de resina depositada dependerá de la viscosidad de la resina, del tamaño de la placa o sustrato y de la velocidad de giro.

La aceleración centrifuga hace que la resina se extienda y se obtenga una película delgada de sustrato. El espesor de la película y otras propiedades dependen de la naturaleza de la resina (viscosidad, velocidad de secado, etc.) y de los parámetros programados durante el proceso de giro. Factores como la velocidad de rotación, aceleración y vapores contribuyen a la definición de las propiedades de la película depositada.

Figura 2.5

Altas velocidades y grandes tiempos de giros dan como resultado películas depositadas más delgadas.

Los componentes principales de una fotoresina son tres: un polímero (resina base), un sensibilizador (también llamado inhibidor) y un solvente. El polímero cambia su estructura cuando es expuesto a una radiación, el solvente permite su aplicación y formación de una fina película sobre la superficie de la placa o sustrato, los sensibilizadores controlan las reacciones fotoquímicas en la fase polimérica. La fotoresina define la zona en que actuará la luz UV. Existen dos tipos de fotoresina positiva y negativa (Figura 2.6), explicadas a continuación.

- Positiva: La resina se ablandará-dañará en las zonas expuestas a la radiación utilizada en el proceso litográfico.
  Las regiones expuestas resultarán más solubles y se eliminan en el proceso de revelado.
- Negativa: La resina se endurecerá-estabilizará en la zonas expuestas a la radiación en el proceso litográfico.
  Las resinas negativas son polímeros (solventes orgánico: hidrocarburos) combinados con un compuesto fotosensible. La absorción de luz se transforma en energía química para iniciar la reacción de polimerización (encadenado de las moléculas en todas direcciones), lo que la hace insoluble en el revelador.

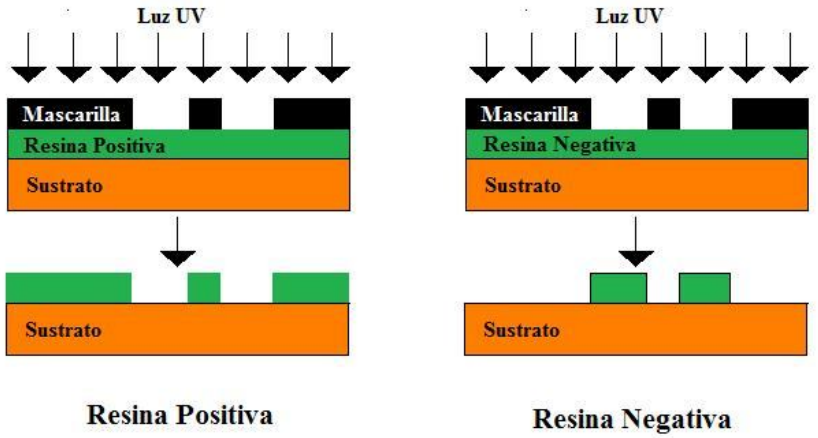


Figura 2.6

Diferencias entre resina positiva y resina negativa son mostradas en la Tabla 2.1.

Características	Tipo de Resina	
	Positiva	Negativa
Adhesión al Silicio (Si)	Regular	Excelente
Tiempo de exposición	Lenta (10-15 seg)	Rápida (2-3 seg)
Revelador	A base de agua	Solvente orgánico
Influencia del oxigeno	No	Si
Característica mínima	0.5µm y menor	±2µm
Resistencia química a la humedad	Buena	Regular
Costo del material	Costoso	Menos costoso

Tabla 2.1

La transferencia se hace colocando la mascarilla sobre el sustrato y exponiéndola a luz UV, el equipo donde se realiza la transferencia se llama alineadora de UV, es cual se muestra en la Figura 2.7.

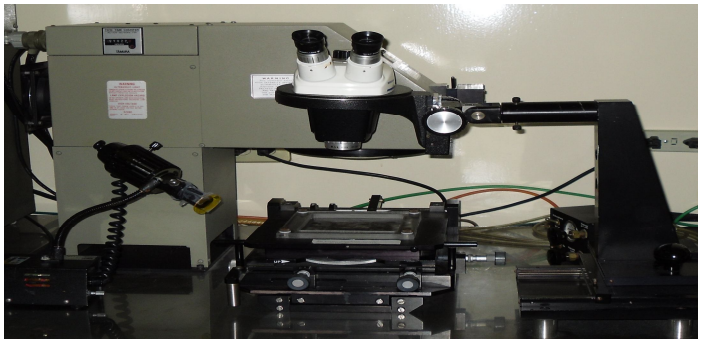


Figura 2.7

Con esto se concluye la técnica de microlitografía óptica, se ha transferido la mascarilla al sustrato deseado y se podrán realizar más replicas de la mascarilla en otro tipo de sustratos.

2.3 Programación

En la programación se utilizó la secuencia mostrada en la Figura 2.8 y a continuación se explican las herramientas mencionadas.

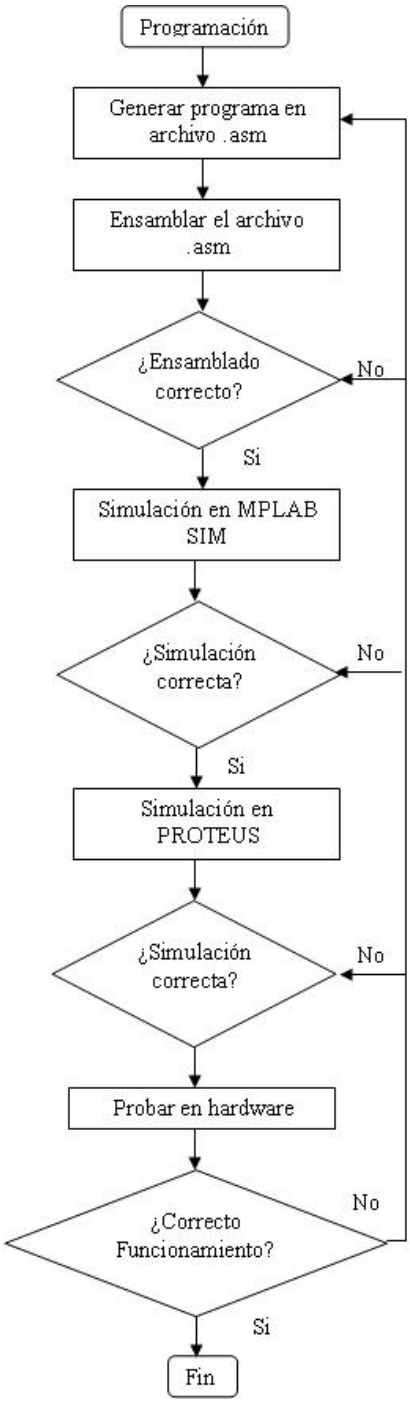


Figura 2.8

2.3.1 MPLAB IDE

La programación del PIC se realizo con MPLAB IDE que es una herramienta software de entorno de desarrollo integrado (Integrated Development Environment, IDE) que se ejecuta en Windows. Con este software se desarrollan aplicaciones para microcontroladores PIC.

MPLAB permite editar el archivo fuente del proyecto, además de ensamblarlo y simularlo en pantalla para comprobar cómo evoluciona la memoria de datos RAM, memoria de programa ROM registros SFE. MPLAB incluye un editor de texto, ensamblador MPSAM y un simulador de proyectos MPLAB SIM. Estos son gratuitos en la página de microchip. En la Figura 2.9 se muestra el entorno de desarrollo en MPLAB.

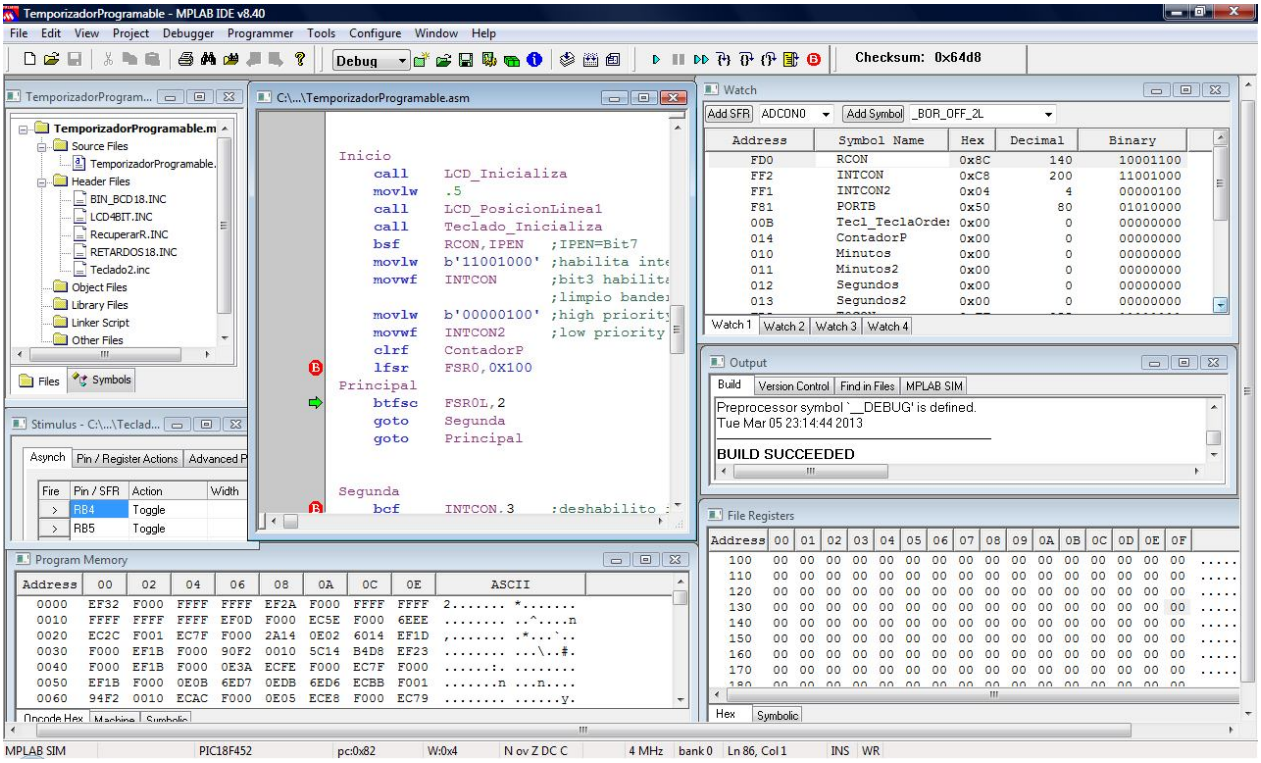


Figura 2.9

En MPLAB se puede programar en ensamblador o lenguaje C, en este caso se utilizo la programación en ensamblador, porque me pareció más organizado, de fácil entendimiento para cualquiera que quiera ver el funcionamiento del programa, ocupa menos memoria y facilidad para encontrar ejemplos claros de programación.

2.3.2 Lenguaje Ensamblador

El lenguaje ensamblador es una secuencia lógica de sentencias pertenecientes a: una línea de comentario, una instrucción ejecutable o una directiva de ensamblado. El lenguaje máquina es un programa o secuencia de instrucciones que viene dado por una secuencia de códigos binarios, como la escritura en lenguaje maquina es tediosa y proclive a errores, el lenguaje ensamblador facilita la tarea de escritura sin perder la cercanía a la máquina. Un programa en ensamblador traduce el

lenguaje ensamblador al lenguaje máquina del CPU. Un programa escrito en lenguaje ensamblador tiene la extensión *.asm, al ser traducido a código máquina tiene la extensión *.hex.

El formato básico de una sentencia en lenguaje ensamblador es:

Etiqueta Opcode Operando(s) Comentario

El campo de etiqueta es opcional, facilita la programación de secuencias.

Opcode contiene el mnemónico de la instrucción o la directiva de ensamblador. Estos están definidos por el fabricante y se encuentran en el set de instrucciones.

Operando(s), son los operandos (1 o 2 o ninguno) de la instrucción o directiva. Cuando hay 2 o más operandos, estos se separan por comas. El operando es a lo que se aplica la instrucción de opcode.

De forma opcional, el último campo es el comentario, antes de escribir un comentario se debe poner ;. Los comentarios son muy útiles para recordar la secuencia de programación y para facilitar a otra persona el entendimiento del código.

Otros códigos ocupados en la programación son las directivas, estas son órdenes para el ensamblador y no tienen traducción a código máquina, como:

ORG inicializa el contador de programa con el valor del operando.

EQU asigna al símbolo contenido en el campo etiqueta el valor de la expresión en campo operando.

__ CONFIG indica la configuración elegida para el proceso de grabación del microcontrolador. Ejemplo:

__CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC

Esto quiere decir:

- No hay protección de código (_CP_OFF)
- No se habilita el watchdog (_WDT_OFF)
- Se habilita el reset mediante Power-Up Timer (_PWRTE_ON)
- Se utiliza el oscilador por cristal de cuarzo (_XT_OSC)

LIST P=PIC18F452 indica el tipo de procesador utilizado

INCLUDE <P18F452.INC> señala el fichero donde se localizan las etiquetas que nombran los registros y el valor de cada uno.

2.3.3 PICSTART Plus

El PIC dispone de una memoria de programa interna donde se almacena el programa que lo controla, este consiste en una serie de números hexadecimales.

El programa de control se graba en la memoria de programa mediante un equipo físico denominado programador, que es conectado a través de un puerto serie, un puerto paralelo o un puerto USB,

mediante el cable de conexión adecuado. En la computadora se ejecuta un software que controla la grabación de la memoria de programa del microcontrolador.

En este proyecto se utilizo el programador PICSTART Plus de Microchip. Este tiene un espacio para colocar el PIC a grabar, es conectado a la computadora por un puerto serie y necesita alimentación de 120 V, este programador es mostrado en la Figura 2.10.

Figura 2.10

2.3.4 Proteus de Labcenter Electronics

Proteus es una compilación de programas de diseño y simulación electrónica, desarrollado por Labcenter Electronics. Consta de los programas Ares e Isis, y de los módulos VSM y Electra.

ISIS (Intelligent Schematic Input System), permite diseñar el plano eléctrico del circuito con componentes que van desde resistencias hasta algunos microcontroladores, incluyendo fuentes de alimentación y generador de señales. Integrado a ISIS esta el módulo VSM (Virtual System Modelling) con el que se puede simular en tiempo real un microcontrodor con diversas conexiones a este.

ARES (Advanced Routing and Editing Software) es una herramienta de enrutado, ubicación y edición de componentes, para la fabricación de circuitos impresos, permite editar las capas de la superficie y de la de soldadura, incluso se pueden hacer circuitos de varias capas. Con el módulo Electra se trazarán automáticamente las pistas de la manera más óptima.

2.4 PIC18F452

Un microcontrolador es un circuito integrado programable que contiene todos los componentes necesarios para controlar el funcionamiento de una tarea determinada. Los microcontroladores PIC (Peripheral Interface Controller) son una familia de microcontroladores de bajo precio, consumo reducido, calidad, fiabilidad, y sencillez de utilización.

Las características del PIC18F452 son mostradas en la Tabla 2.2

Características	**PIC18F452**
Frecuencia de operación	100Hz-40 MHz
Memoria de Programa (Bytes)	32K
Memoria de Programa (instrucciones)	16384

Memoria de Datos (Bytes)	1536
Memoria EEPROM (Datos)	256
Fuentes de Interrupción	18
I/O (entrada/salidas)Puertos	A, B, C, D, E
Timers	4
Captura/Comparación/PWM	2
Comunicación Serial	MSSP (Master Synchronous Port), direccionable USART (Universal Synchronous Asynchronous Receiver Transmitter)
Comunicación Paralela	PSP
10-bit Modulo Analógico Digital	8 canales de entrada
Resets y Delays	POR, BOR, RESET Instrucción, Stack Full,
Programable con detección de bajo voltaje	si
Número de instrucciones	75
Empaquetados	40pin DIP, 44pin PLCC, 44pin TQFP

Tabla 2.2

El PIC18f452 utiliza arquitectura Harvard que dispone de dos memorias independientes a las que se conecta mediante dos grupos de buses separados:

- Memoria de datos RAM
- Memoria de programa ROM

El PIC18F452 tiene un procesador RISC (Reduced Instruction Ser Computer), es decir que tiene un repertorio reducido de instrucciones. Las instrucciones son muy simples y suelen ejecutarse en un ciclo máquina.

Microchip diseña sus microcontroladores PIC con procesador RISC optimizado para ejecutar a muy alta velocidad un reducido número de instrucciones, solo las más frecuentemente utilizadas. En los microcontroladores RISC las instrucciones complejas se obtienen ejecutando un conjunto de instrucciones disponibles, en lugar de una única instrucción.

El procesador RISC tiene una estructura Pipeline. Un procesador segmentado o Pipeline realiza simultáneamente la ejecución de una instrucción y la búsqueda de código de la siguiente, de esta manera, se puede ejecutar una instrucción en un ciclo máquina que está constituido por cuatro ciclos de reloj.

Su formato de instrucciones tiene una arquitectura ortogonal, una instrucción puede utilizar cualquier elemento de la arquitectura como fuente o destino. En los microcontroladores PIC la salida de la ALU (Unidad Lógica Aritmética) va al registro W (Work register) y también a la memoria de datos, así el resultado puede guardarse en cualquiera de los dos destinos. La gran ventaja de esta arquitectura es que permite un gran ahorro de instrucciones ya que el resultado de cualquier instrucción que opere con la memoria puede dejarse en misma posición de memoria o en el registro W.

En la memoria de datos de los PICs se encuentran ubicados casi todos los registros de control del microcontrolador y sus periféricos de entrada/salida, así como las posiciones de memoria de usos generales.

El contador de programa o PC (Program Counter) es un registro interno de 21 bits, permite direccionar 2 Mbytes de memoria de programa. Este registro contiene la dirección de la próxima dirección a ejecutar y se incrementa automáticamente. Cuando el PIC se conecta alimentación o cuando ocurre un reset, el contador de programa se pone a cero forzando que la dirección de inicio sea 000h, la primera instrucción ejecutada será la que este guardada en esta posición. Algunas instrucciones cambian el contenidao del PC alterando la secuencia lineal de ejecución. Dentro de estas instrucciones se encuentran el GOTO y el CALL que permiten cargar de 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 saltos condicionales, que producen un incremento adicional del PC si se cumple una condición específica, haciendo que el programa salte sin ejecutar la instrucción siguiente.

La memoria de programa puede direccionar hasta 32 KB (flash) la memoria de datos hasta 1.5 KB y una EEPROM de 256 bytes.

En la memoria interna de datos (archivos de registros), cada registro de la memoria tiene una dirección de 12 bits (capacidad de 4096 bytes). La memoria de datos está dividida en dos áreas. Una de ellas corresponde a Registros de Propósito General (GPR) y la otra dedicada a loa Registros de Funciones Especiales (SFR) que controlan los recursos y periféricos del microcontrolador

Watchdog (perro guardián) consiste en un temporizador que, cuando se desborda y pasa por 0, provoca un reset automáticamente en el sistema. Se debe diseñar el programa de trabajo que controle la tarea de forma que actualice o inicialice el watchdog antes que provoque el reset.

El PIC18F452 es un microcontrolador de 40 pines (Figura 2.8). Tiene 5 puertos, 4 timers, 2 módulos generadores de señales PWM y un set de instrucciones de 75 palabras. Tiene 5 puertos que pueden ser configurados como entradas/salidas, los puertos A, B, C y D son de 8 bits, mientras que el puerto E es de sólo 3 bits.

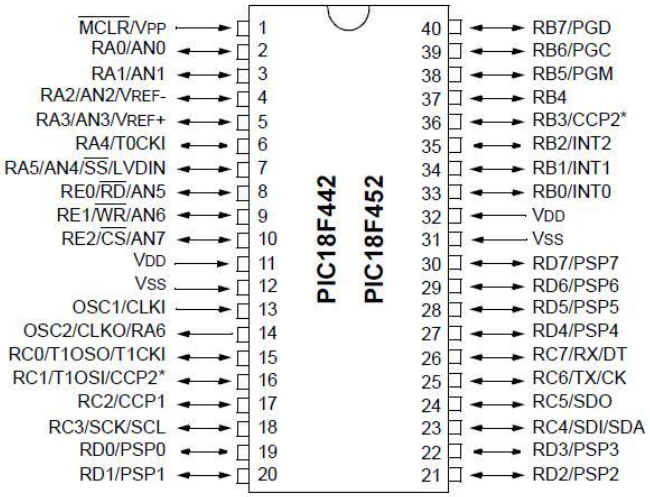


Figura 2.8

El pin 1 corresponde a MCLR/Vpp, donde MCLR es el master clear, siendo activado al nivel lógico bajo, cuando recibe este nivel resetea el PIC, también es el pin del voltaje cuando actúa en este modo.

En los pines 11 y 32 se encuentra la alimentación positiva del integrado (Vdd), en niveles de 5V, mientras que en los pines 12 y 31 se debe conectar a la señal de alimentación de nivel lógico 0V.

Los pines 13 y 14 corresponden a las señales de reloj externas, que en este caso provienen de un cristal de cuarzo de 4 MHz, con capacitores de 22pF conectados a tierra.
El PIC18F452 dispone de los siguientes puertos:

- Puerto A puede ser configurado como entrada/salida, o convertidor analógico digital.
- Puerto B está formado por 8 entradas/ salidas y por software puede ser programado como entradas con resitencias pull-up, algunos de los pines de este puerto sirven como interrupciones externas, control de voltaje o para comunicaciones seriales.
- Puerto C es para entradas/salidas o para funciones alternas específicas de cada pin, como timer, dos generadores de señales PWM, ingresos de captura y comunicaciones en serie (12C, SPI y USART). Puerto D es para entradas/salidas o para ser un puerto de comunicación esclavo paralelo.
- Puerto E con 3 bits para entradas/salidas o control del puerto de comunicaciones paralelo (lectura, escritura y habilitador).

Las características del PIC18F452 que se utilizaron en este trabajo, serán detalladas a continuación.

2.4.1 Table Read/Table Write (TLBRD/TLBWT)

Las tablas de lecturas/escrituras son una forma de almacenar datos en la memoria del programa, permite almacenar 2 bytes por palabra de instrucción

TBLPRT	Registro puntero de memoria de programa, especifica la dirección del byte
TABLAT	Registro que contiene el byte que fue leído o el que será escrito (8 bits). Los datos son transferidos a/de memoria de programa, byte por byte.
TBLRD	Instrucción de lectura de tablas de datos, Figura 2.9.
TBLWT	Instrucción de escritura de tablas de datos, Figura 2.10.

La memoria de programa es de 16 bits, mientras que la RAM es de 8 bits. Las instrucciones TBLRD/TBLWT mueven datos entre estas dos zonas de memoria.

El registro TBLPRT, está compuesto por tres registros:

TBLPTRU	Bits 16-21
TBLPTRH	Bits 8-15
TBLPTRL	Bits 0-7

Los 21 primeros bits direccionan hasta 2Mbytes de memoria de programa. El bit 22 se usa para acceder a los bits de configuración.

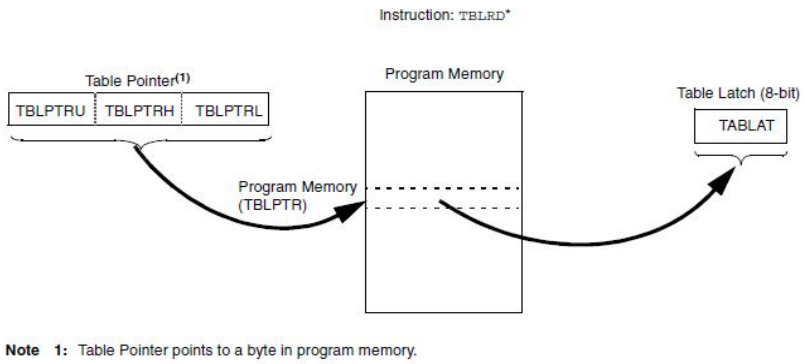


Figura 2.9

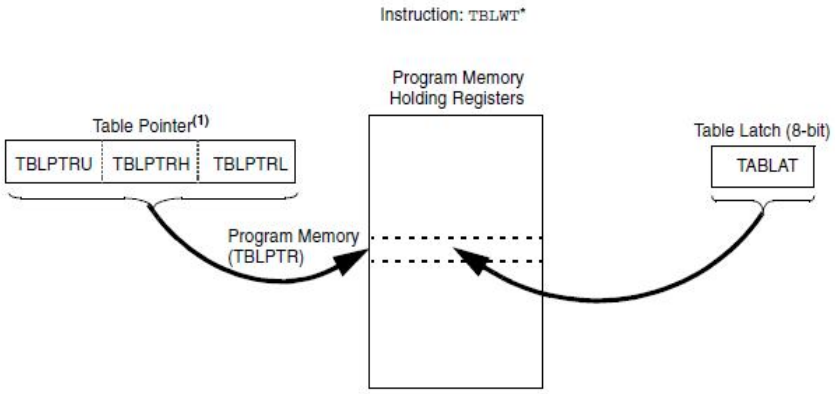


Figura 2.10

2.4.2 Timer

El timer se implementa por medio de un contador que determina un tiempo preciso entre el momento en que el valor es cargado y el instante en el que se produce su desbordamiento. Consiste en un contador ascendente o descendente que, una vez inicializado con un valor, su contenido se incrementa con cada pulso de entrada hasta lograr su valor máximo, desbordándose y volviendo a comenzar en cero.

Los impulsos aplicados pueden provenir de pulsos aplicados como entradas o de la señal de reloj interna, lo que permite al timer actuar de dos formas:

- Como contador de impulsos que llegan por un pin especifico del exterior. Su misión es contar el número de acontecimientos externos.
- Como temporizador, se utiliza para determinar intervalos de tiempos concretos.

A continuación se explica el Timer0 como temporizador, ya que se ocupó para registrar el tiempo de revelado o ataque.

T0CON: TIMER0 Registro de Control

TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PSO

Bit 0

Bit 7 TMR0ON Bit de control del Timer0 On/Off

1= Habilita Timer0
0= Detiene Timer0

Bit 6 T08BIT Bit de control Timer0 8-bit/16-bit
1 Timer 0 configurado como 8-bit timer/counter
0 Timer 0 configurado como 16-bit timer/counter

Bit 5 T0CS Timer0 fuente del reloj
1 reloj externo
0 fuente interna de reloj

Bit 4 Si se selecciono reloj externo en Bit 5, se define incrementar de
1 incrementar en transición de alto a bajo
0 incrementar en transición de bajo a alto
Bit 3 Timer0 bit para asignar prescaler
1.1.1.1 prescaler no es asignado
0 prescaler asignado

Bit 2-0 Timer0 Prescale (Divisor de frecuencia)
111=1:256 valor prescale
110=1:128 valor prescale
101=1:64 valor prescale
100=1:32 valor prescale
011=1:16 valor prescale
010=1:8 valor prescale
001=1:4 valor prescale
000=1:2 valor prescale

INTCON: Registro

GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF

Bit 0

Los bits que se deben configurar para temporización son:

Bit 5 TMR0IE: TMR0 bandera de habilitación de interrupción por desbordamiento
1 habilita interrupción de desbordamiento de TMR0
0 deshabilita interrupción de desbordamiento TMR0

Bit 2 TMR0IF: TMR0 bandera de interrupción de desbordamiento
1 el registro TMR0 se ha desbordado (se debe limpiar por software antes de salir de la interrupcion).
0 no se ha desbordado el registro TMR0

Para un contador de 16 bits los valores de desbordamiento son almacenados en TMR0H y TMR0L. El incremento es desde 0000h hasta FFFFh (65535d) y al llegar a este último valor avisa a TMR0IF.

$$\text{DesbordamientTimer0} = \frac{4}{\text{Fosc}} \times \text{Prescaler} \times \left[65535 - (\text{TMR0H - TMR0L})\right]$$

$$(\text{TMR0H - TMR0L}) = 65535 - \frac{\text{DesbordamientoTimer0} \cdot \text{Fosc}}{4 \cdot \text{Prescaler}}$$

Ambas fórmulas están en segundos.

Fosc= 4MHz valor del reloj de cuarzo.

Para un desbordamiento cada 5 segundos, con un reloj de cuarzo de 4MHz y prescaler de 256

$$(\text{TMR0H - TMR0L}) = 65535 - \frac{5 \cdot 4000000}{4 \cdot 256} \approx 46004\text{d} = \text{B3B4h}$$

Por lo que TMR0H=B3 y TMROL= B4

2.4.3 Interrupciones

Consiste en un mecanismo por el cual un evento interno o externo, asíncrono respecto al programa, puede interrumpir la ejecución de éste produciendo automáticamente un salto a una subrutina de atención, de manera que pueda atender inmediatamente el evento y retomar después la ejecución del programa exactamente en donde estaba en el momento de ser interrumpido. Este mecanismo es muy útil para el manejo de timers o rutinas que deben producirse periódicamente (actualizar un display, anti rebote del teclado), detección de pulsos externos, recepción de datos.

El PIC18F452 tienen múltiples fuentes de interrupción tanto internas como externas, los vectores de interrupción son:

- 000008h Vector de interrupción para interrupciones de alta prioridad
- 000018h Vector de interrupción para interrupciones de baja prioridad

Se puede definir por software que interrupciones serán de Alta/baja prioridad, para ello el bit RCON<IPEN> debe ser puesto en 1 o 0 y habilitar INTCON<GIEH> para interrupciones de alta prioridad o INTCON<GIEL> para interrupciones de baja prioridad.

Los registros asociados a las interrupciones son:

- RCON PIR1, PIR2, PIR3. Estos registros más INTCON/3 contienen las banderas de las interrupciones.
- INTCON PIE1, PIE2, PIE3. Estos registros más los de INTCON habilitan/deshabilitan las interrupciones particulares.
- INTCON2/3 IRP1, IRP2, IRP3. Estos registros más INTCON2/3 definen que interrupciones serán de alta o baja prioridad.

Cada fuente de interrupción tiene asociado tres bits de control: bit de Bandera (Flag) indica que una interrupción ha ocurrido, Bit de Habilitación y Bit de prioridad.

Una vez reconocido el tipo de interrupción, la dirección de retorno se respaldada en el SP y el PC es cargado con el Vector de Interrupción correspondiente. La bandera de interrupción debe ser limpiada por software antes de salir de la rutina de interrupción, la instrucción *RETFIE* restituye el estado del bit GIE y del PC rescata la dirección de retorno.

2.4.4 PWM

Cada módulo CCP está compuesto por un registro de 16 bits que puede funcionar modo captura o comparador o generador de señales PWM.

En modo PWM, cada módulo CCP puede generar señales moduladas por ancho de pulso, con hasta 10 bits de resolución, la frecuencia, tiempo en alto y ciclo de trabajo son configurados por software, una vez habilitado el PWM y definidos los parámetros se tiene un funcionamiento automático, hace uso del timer 2.

Los pasos para definir un PWM son:

1. Definir el periodo del PWM y escribirlo en PR2. Los valores de periodo (PR2) pueden ser de de 0 a 256.

$$PeridoPWM = 4 \bullet (PR2+1) \bullet ValorPreescaler \bullet Tosc$$

$$PR2 = \frac{PeriodoPWM}{4 \cdot Tosc \cdot ValorPrescaler} - 1$$

Tosc=1/Focs=1/4MHz =250ns, 4MHz frecuencia del cristal, ValorPrescaler (divisor de frecuencia)=1 o 4 o 16.

2. Definir el ciclo de trabajo y escribirlo en CCPR1L

$$CicloTrabajo = (CCPR1L:CCPICON<5:4>) \cdot Tosc \cdot (TMR2Prescaler)$$

$$CCPR1L = \frac{CicloTrabajo}{Tosc \cdot TMR2Prescaler}$$

3. Definir el CCP1 o CCP2 en el puerto C como salida.

bcf TRISC, 2 o bcf TRISC, 1

4. Definir el valor de prescaler (divisor de frecuencia) y habilitar el Timer 2, escribiendo en T2CON

T2CON: TIMER CONTROL Registro de Control

-	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0

Bit 0

Bit 2 TMR2ON
1= Timer 2 on
0= Timer 2 off

Bit 1-0 T2CKPS1:T2CKPS0 Timer 2 Prescaler o Divisor de Frecuencia (ValorPrecaler)
00= Prescaler 1
01= Prescaler 4
1x= Prescaler 16

5. Configurar el modulo CCP1 o CCP2 como operación PWM.

Registro CCP1CON/CCP2CON

-	-	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0

Bit 0

Bit 5-4 DCxB1:DCxB0: PWM ciclo de trabajo bit 1 y bit 0

Bit 3-0 CCPxM3: CCPx:M0: bits de selección
11xx= Modo PWM

Capítulo 3 Diseño de Circuitos y Programación

3.1 Fuentes lineales de voltaje positivo

Las fuentes de alimentación del sistema (Figura 3.1) parten de un transformador de 16V a 3A del que se obtienen los voltajes para la fuente de 5V y para la fuente de 10V. Los voltajes fijos se obtienen con los integrados LM7805 para la fuente de 5V y LM710 para la fuente de 10V. Ambos integrados llevan disipadores de calor, para evitar que se dañen, pero LM705 lleva un disipador más grande debido a que la caída de voltaje es mayor.

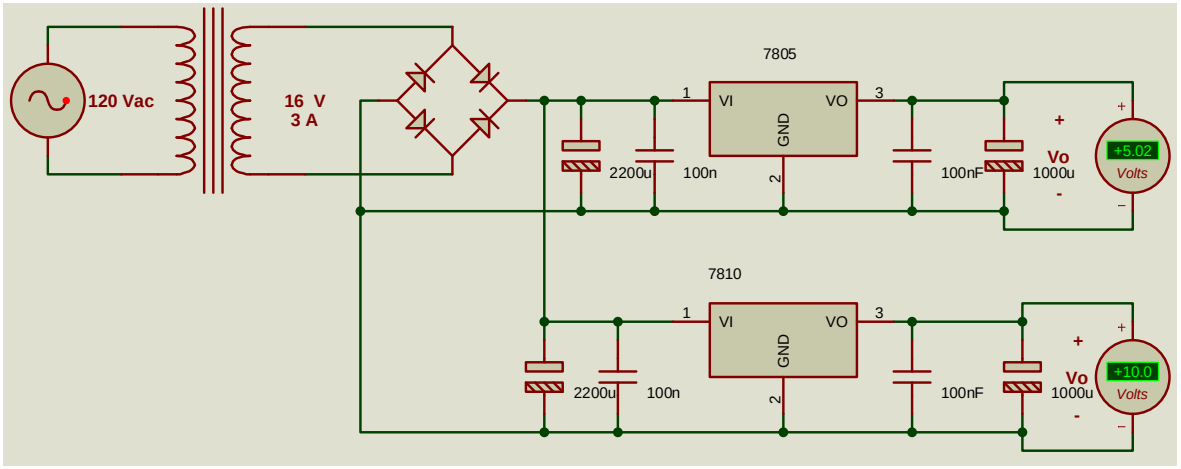


Figura 3.1

El funcionamiento general una fuente positiva fija parte de un transformador que en la parte primaria es conectado a la alimentación de 120V a 60Hz, la parte secundaria del transformador (sin derivación central) es conectado a un puente rectificador de onda completa cuya función es que el voltaje sea solo positivo, esta señal pasa por un filtro tratando de eliminar el voltaje de rizo para después pasar por el rectificador que bajara la amplitud de la señal de voltaje al valor que se requiera, y finalmente otra etapa de filtrado para el ruido.

3.2 Etapa de Potencia-Motor a Pasos Bipolar

Para realizar el movimiento vertical se utiliza un motor de pasos bipolar, este requiere una etapa de potencia, ya que el PIC18F452 solo proporciona 25mA.

En un motor a pasos bipolar las bobinas del estator se conectan formando dos grupos, como se muestra en la Figura 3.2.

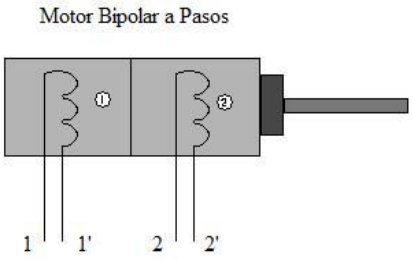


Figura 3.2

El diseño de la etapa de potencia está formado por los circuitos integrados L297 y L298. El integrado L297 realiza la generación de fases 1, 1', 2, 2' y el cambio en la dirección de la corriente que circula por los devanados, mientras que el circuito L298 maneja la potencia dos puentes H.

3.2.1 L297

El integrado L297 es un controlador de motor a pasos, genera un controlador de cuatro señales de fases para un motor a pasos bipolar de dos fases o un motor a pasos unipolar de cuatro fases, para aplicaciones con microcontroladores. Puede controlar el motor en paso completo, medio paso, y micropasos, permitiendo el control de cambio en la corriente de los devanados. Este integrado requiere señal de reloj, dirección y modo de entrada de las señales. Como las fases son generadas por el integrado L297 la carga del microcontrolador se reduce. Tiene empaque en DIP20 y SO2O y puede acoplado con L298N, L293E o con transistores.

El integrado L297 requiere entradas del Pic para el reloj, seleccionar el tipo de paso (completo o medio paso), seleccionar el sentido de giro y reset, que fueron dadas con el PIC18F452. Con esto el circuito genera la secuencia de fases mostrado en la Tabla 3.1.

	1	**1'**	**2**	**2'**		**1**	**1'**	**2**	**2'**
Paso Completo Sentido Horario	0	1	1	0	**Paso Completo Sentido Anti Horario**	1	0	1	0
	0	1	0	1		1	0	0	1
	1	0	0	1		0	1	0	1
	1	0	1	0		0	1	1	0

Tabla 3.1

El diagrama del integrado L297 se encuentra en la Figura 3.3

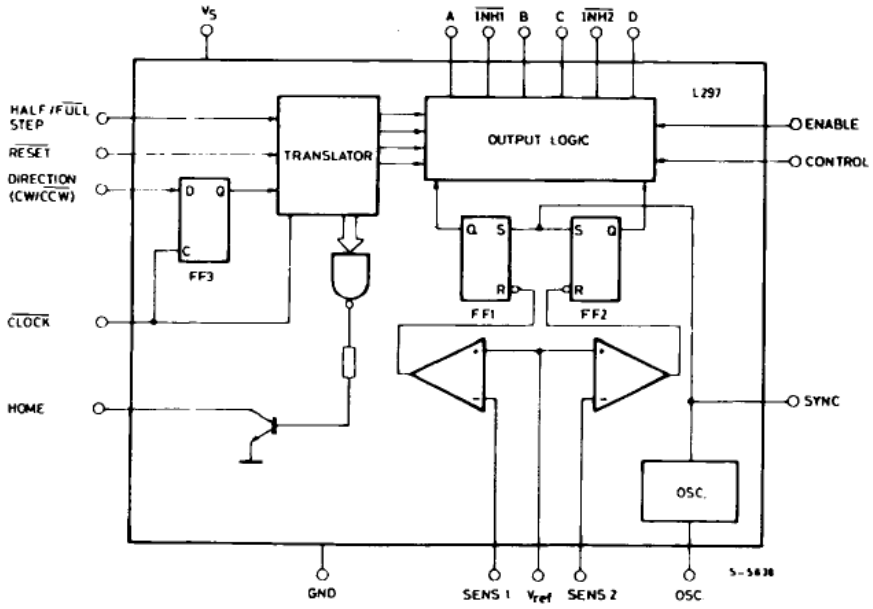


Figura 3.3

La conexión del integrado L297 para el motor a pasos se encuentra en la Tabla 3.2.

No. Pin	Nombre	Conexión
1	SYNC	Nada
2	GND	Tierra
3	HOME	Nada
4	A	Pin 5 de L298
5	$\overline{\text{INH}}1$	Pin 6 de L298
6	B	Pin 7 de L298
7	C	Pin 10 de L298
8	$\overline{\text{INH}}2$	Pin 11 de L298
9	D	Pin 12 de L298
10	ENABLE	1 iniciar, 0 detener
11	CONTROL	5 V
12	Vs	5V
13	SENS2	Pin 15 de L298
14	SENS1	Pin 1 de L298
15	Vref	1.5 V. Figura 3.4 (a)
16	OSC	Ver Figura (b)
17	CW/$\overline{\text{CCW}}$	Giro, 5 V anti horario, 0 V horario
18	$\overline{\text{CLOCK}}$	Señal TTL
19	HALF/$\overline{\text{FULL}}$	Paso completo 0 V, medio paso 5 V
20	$\overline{\text{RESET}}$	Figura 3.4(c)

Tabla 3.2

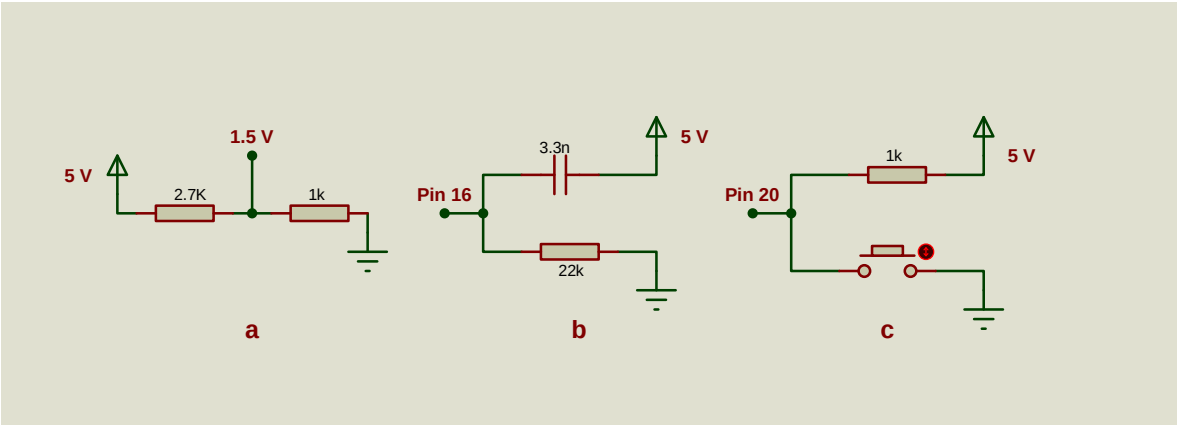


Figura 3.4

3.2.2 L298

El L298 es un circuito integrado monolítico empaquetado en 15-lead Multiwatt y PowerSO20. Maneja alto voltaje y alta corriente con un doble puente H diseñado para aceptar niveles lógicos TTL y manejar cargas inductivas como relays, solenoides, DC y motores a pasos. Dos entradas activan o desactivan el integrado, independientemente de las señales de entrada. Los emisores de los transistores inferiores del puente H están conectados juntos y la correspondiente terminal de salida puede ser utilizada para una resistencia de detección externa. Tiene una entrada adicional de voltaje, por lo que la lógica funciona a bajo voltaje.

El L298 integra dos etapas de potencia (A, B), es una configuración de puente H y sus salidas son para una carga inductiva, estos puentes H dependen del estado de las entradas. La corriente que fluye a través de la carga sale del puente H a las resistencias de medición externas RSA y RSB que permiten detectar la intensidad de la corriente.

La etapa de entrada de cada puente H son accionadas por cuatro compuertas, las entradas de estas son: In 1 e In2, Enable A y In 3, y In 4 y Enable B. Las entradas establecen el estado del puente H cuando Enable está en alto y si Enable está en bajo se inhabilita el puente. Todas las entradas son compatibles TTL.

Este circuito es capaz de suministrar una corriente de hasta 2A y diodos conectados al final de este establecen caminos de recirculación de corriente, en la Figura 3.5 se muestra su constitución interna.

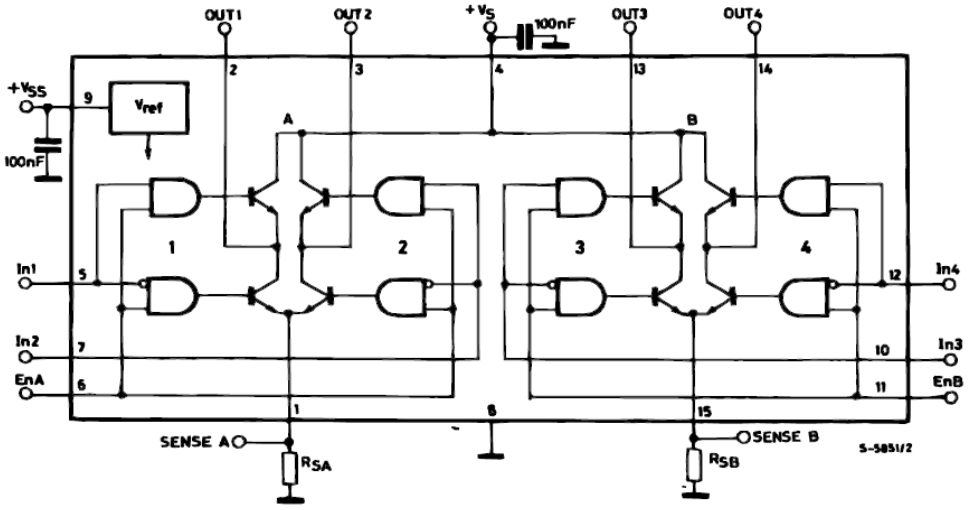


Figura 3.5

La conexión del L298 para esta aplicación se encuentra en la Tabla 3.3

No. Pin	Nombre	Conexión
1	Current sensing A	Pin 4 de L297 y RSA= 0.5 Ω a tierra
2	Output 1	Bobina 1
3	Output 2	Bobina 12
4	Supply Voltage Vs	Alimentación de Motor 10 V
5	Input 1	Pin 4 de L297
6	Enable A	Pin 5 de L297
7	Input 2	Pin 6 de L297
8	GND	Tierra
9	Logic Supply Voltage Vss	5 V
10	Input 3	Pin 7 de L297
11	Enable B	Pin 8 de L297
12	Input 4	Pin 9 de L297
13	Output 3	Bobina 2
14	Output 4	Bobina 2
15	Current Sensing B	Pin 1 de L297 y RSB=0.5 Ω a tierra

Tabla 3.3

En la Figura 3.6 se muestra el diagrama de conexión del PIC y los integrados L297 y L298.

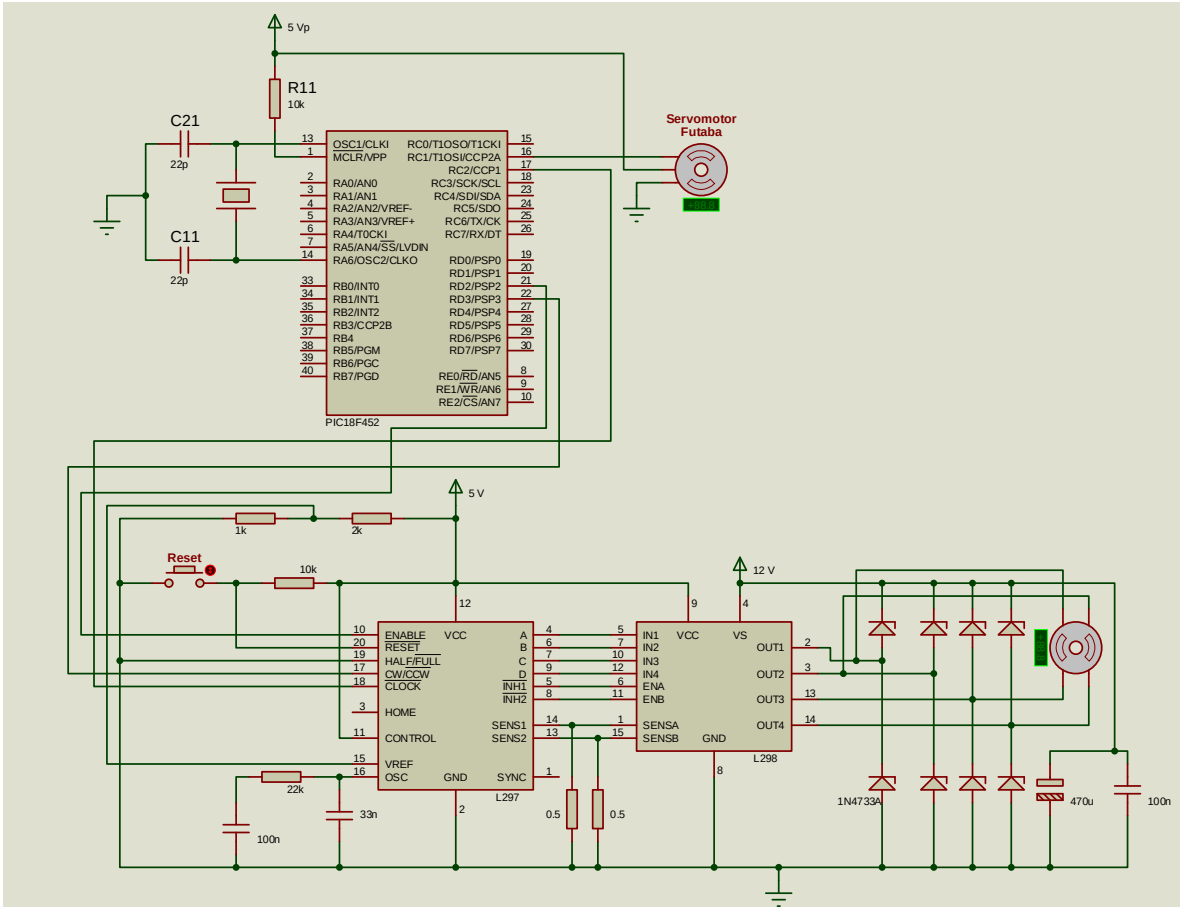


Figura 3.6

3.3 Servomotor PWM

Para realizar los movimientos del revelador al enjuague o del atacante al enjuague, se utilizó un servomotor analógico Futaba FP-S148. Este requiere conexión a alimentación (5 V), tierra y la señal de control PWM.

El control del servomotor se limita a indicar en qué posición se debe situar, mediante una señal cuadrada TTL modulada por ancho de pulso PWM. La duración del nivel alto de la señal indica la posición donde se quiere poner el eje del motor. La duración de los pulsos indica el ángulo de giro del motor.

Para esta aplicación se genero el PWM con uno de los puertos para PWM del PIC, con este puerto especial se requiere introducir el periodo y el ciclo de trabajo. La frecuencia del servomotor se programó en 244.14Hz, y su movimiento fue definido cambiando el ciclo de trabajo para mover el servomotor lo más cerca posible de 0° y 180°, estos valores no fueron exactos porque al tratar de forzar a que el servomotor se posicionara en estos extremos, el servomotor presentaba un zumbido y vibraba. Los valores con que se programo son mostrados en la Tabla 3.4 y sus gráficas aparecen en la Figura 3.7.

Posición	Periodo [ms]	PR2	Ciclo de trabajo [ms]	CCPR2L
0°	4.096	255	0.883	55

180°	4.096	255	4.015	250

Tabla 3.4

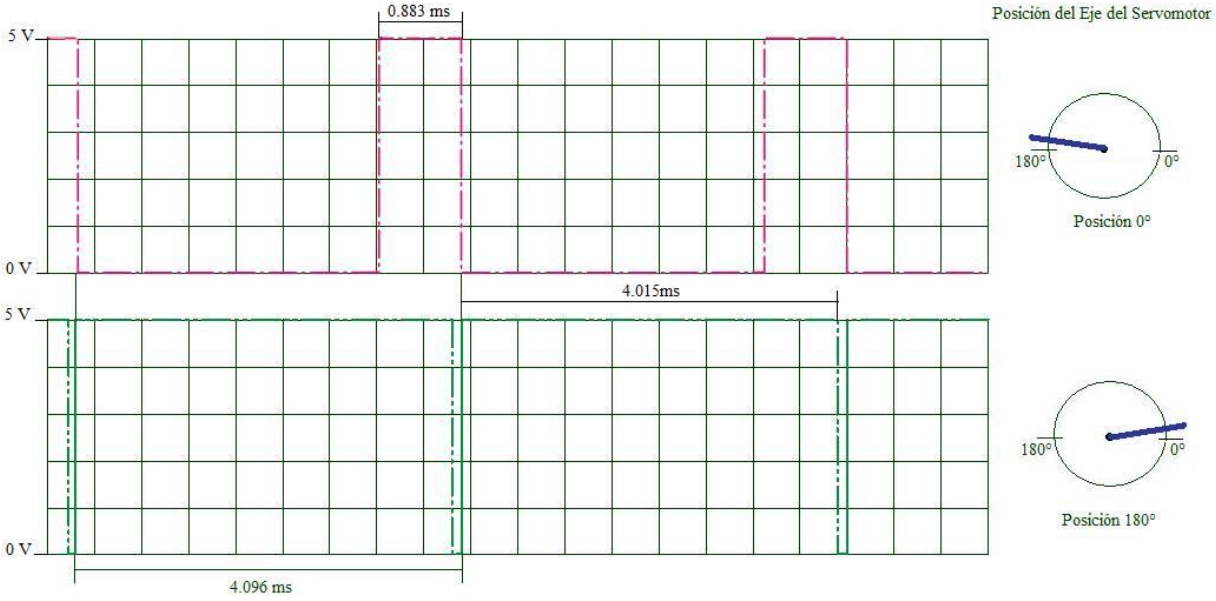


Figura 3.7

3.4 Bibliotecas de funciones

Las bibliotecas de funciones o librerías son programas que se pueden usar repetidamente en otros programas. Para generar una librería la programación se debe guardar en un archivo con extensión .icn y para agregarla a un programa basta con incluir su nombre.inc al final del programa principal. El programa principal es guardado con extensión .asm.

La interacción hombre-máquina tiene un display LCD 16x2 y un teclado matricial de 4x4, a continuación se muestra los diagramas de flujo de estos procesos y la programación de manera de librerías. En la Figura 3.8 se muestra la conexión del LCD, teclado matricial y el PIC18F452.

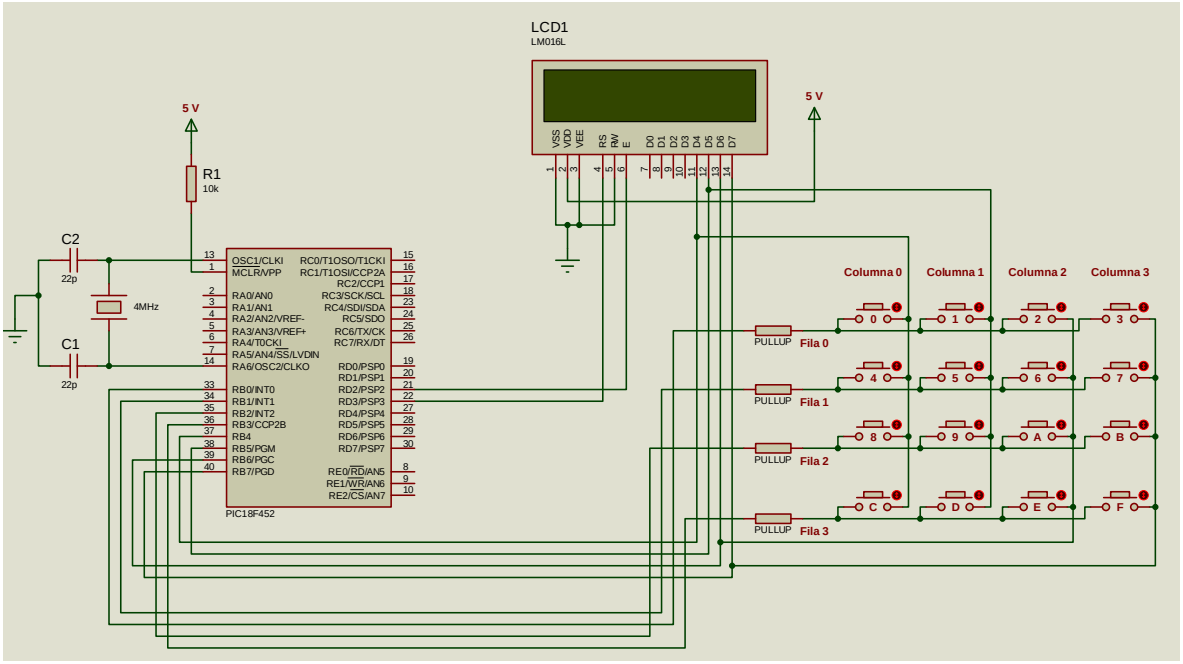


Figura 3.8

3.4.1 Display LCD

Se utilizo el display LCD 16x2 modelo JHD-162ASTN, que se conectó para esta aplicación como indica la Tabla 3.4.

No. Pin	Nombre	Conexión
1	Vss	0 V
2	Vcc	+5 V
3	V_{EE}	Contraste del LCD 0 V
4	Rs	Instrucción Registro/Dato PORTD 3
5	R/W	Escribir 0/Leer 1, PORTD 2
6	E	Enable, 0 V o PORTD 1
7	DB0	Dato de Entrada/Salida no se utiliza para conexión con 4 bits
8	DB1	Dato de Entrada/Salida no se utiliza para conexión con 4 bits
9	DB2	Dato de Entrada/Salida no se utiliza para conexión con 4 bits
10	DB3	Dato de Entrada/Salida no se utiliza para conexión con 4 bits
11	DB4	Dato de Entrada/Salida, PORTB 4
12	DB5	Dato de Entrada/Salida, PORTB 5
13	DB6	Dato de Entrada/Salida, PORTB 6
14	DB7	Dato de Entrada/Salida, PORTB 7
15	LED -	Voltaje para LED
16	LED +	Voltaje para LED

Tabla 3.4

El diagrama de flujo para inicializar el display LCD 16x2 con 4 bits se muestra en la Figura 3.9.

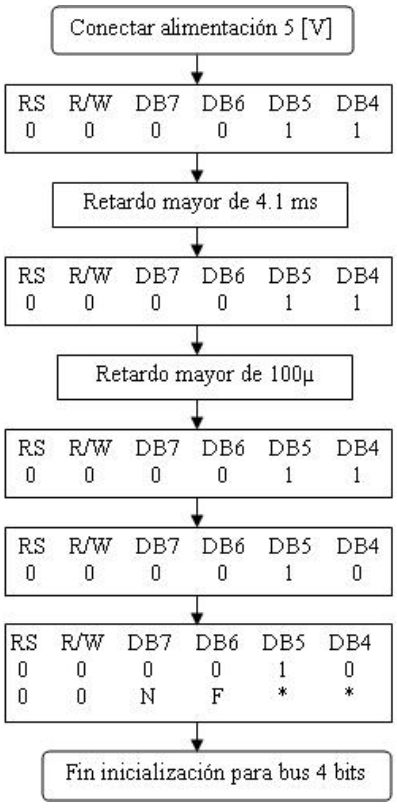


Figura 3.9

Librería para el display LCD 16x2

```
        CBLOCK 0x003
        LCD_Dato
        LCD_GuardaDato
        LCD_GuardaTRISB
        LCD_Auxiliar1
        LCD_Auxiliar2
        ENDC

LCD_CaracteresPorLinea    EQU.16 ;Número de caracteres por línea del LCD

#DEFINE      LCD_PinRS          PORTD,3
#DEFINE      LCD_PinRW          PORTD,2
#DEFINE      LCD_PinEnable      PORTD,1
#DEFINE      LCD_BusDatos       PORTB

LCD_Inicializa     ;Configuración de las líneas conectadas a pines RS, R/W y E

        bcf    TRISD,3      ;DEFINICIÓN DEL PUERTO D 1,2 y 3 como Salidas
        bcf    TRISD,2
        bcf    TRISD,1
        movlw  b'00001111'
        movwf  TRISB
        bcf    LCD_PinRS    ;Activa el Modo Comando poniendo RS=0.
        bcf    LCD_PinRW    ;En caso de que esté conectado le indica que se va a
                            ;escribir en el LCD. NO LO TENGO CONECTADO
        bcf    LCD_PinEnable;Impide funcionamiento del LCD poniendo E=0.

        call   Retardo_20ms       ;EMPIEZA
        movlw  b'00110000'        ;PRIMERA INSTRUCCIÓN
        call   LCD_EscribeLCD     ; Escribe el dato en el LCD.
        call   Retardo_5ms
```

```
        movlw  b'00110000'         ;SEGUNDA INSTRUCCIÓN
        call   LCD_EscribeLCD
        call   Retardo_200micros
        movlw  b'00110000'         ;TERCERA INSTRUCCIÓN
        call   LCD_EscribeLCD
        movlw  b'00100000'         ; Interface de 4 bits.
        call   LCD_EscribeLCD

                                   ;Ahora configura el resto de los parámetros:
        call   LCD_2Lineas4Bits5x7;LCD de 2 líneas y caracteres de 5x7 puntos.
        call   LCD_Borra     ;Pantalla encendida y limpia. Cursor al principio
        ;call  LCD_CursorOFF       ; de la línea 1. Cursor apagado.
        call   LCD_CursorON
        call   LCD_CursorIncr      ; Cursor en modo incrementar.
        return

LCD_EscribeLCD
        andlw  b'11110000'         ; Se queda con el nibble alto del dato que es el
        movwf  LCD_Dato            ; que hay que enviar y lo guarda.
        movf   LCD_BusDatos,W      ; Lee la información actual de la parte baja
        andlw  b'00001111'         ; del Puerto B, que no se debe alterar.
        iorwf  LCD_Dato,F          ; Enviará la parte alta del dato de entrada
                                   ; y en la parte baja lo que había antes.
        movf   TRISB,W             ; Guarda la configuración que tenía antes TRISB.
        movwf  LCD_GuardaTRISB
        movlw  b'00001111'         ; Las 4 líneas inferiores del Puerto B se dejan
        andwf  TRISB,F             ; como estaban y las 4 superiores como salida.
        movf   LCD_Dato,W          ; Recupera el dato a enviar.
        movwf  LCD_BusDatos        ; Envía el dato al módulo LCD.
        bsf    LCD_PinEnable;Permite funcionamiento del LCD mediante un pequeño
        bcf    LCD_PinEnable       ;pulso y termina impidiendo el funcionamiento de
                                   ;LCD.
        movf   LCD_GuardaTRISB,W   ; la configuración del Puerto B.
        movwf  TRISB               ; Realmente es TRISB.

        return


LCD_CursorIncr                     ; Cursor en modo incrementar.
        movlw  b'00000110'
        goto   LCD_EnviaComando
LCD_Linea1                         ; Cursor al principio de la Línea 1.
        movlw  b'10000000'         ; Dirección 00h de la DDRAM
        goto   LCD_EnviaComando
LCD_Linea2                         ; Cursor al principio de la Línea 2.
        movlw  b'11000000'         ; Dirección 40h de la DDRAM
        goto   LCD_EnviaComando
LCD_PosicionLinea1                 ;Cursor a posición de la Línea 1, a partir de la
        iorlw  b'10000000'         ; dirección 00h de la DDRAM más el valor del
        goto   LCD_EnviaComando    ; registro W.
LCD_PosicionLinea2                 ;Cursor a posición de la Línea 2, a partir de la
        iorlw  b'11000000'         ; dirección 40h de la DDRAM más el valor del
        goto   LCD_EnviaComando    ; registro W.
LCD_OFF                            ; Pantalla apagada.
        movlw  b'00001000'
        goto   LCD_EnviaComando
LCD_CursorON                       ; Pantalla encendida y cursor encendido.
        movlw  b'00001110'
        goto   LCD_EnviaComando
LCD_CursorOFF                      ; Pantalla encendida y cursor apagado.
        movlw  b'00001100'
        goto   LCD_EnviaComando
LCD_Borra                          ;Borra toda la pantalla, memoria DDRAM y pone el
        movlw  b'00000001'         ;cursor a principio de la línea 1.
        goto   LCD_EnviaComando
LCD_2Lineas4Bits5x7                ; Define la pantalla de 2 líneas, con caracteres
        movlw  b'00101000'         ; de 5x7 puntos y conexión al PIC mediante bus

LCD_EnviaComando
        bcf    LCD_PinRS           ; Activa el Modo Comando, poniendo RS=0.
```

```
       goto   LCD_Envia
LCD_Caracter
       bsf    LCD_PinRS           ; Activa el "Modo Dato", poniendo RS=1.

LCD_LineaEnBlanco
       movlw  LCD_CaracteresPorLinea
       goto   LCD_EnviaBlancos
LCD_UnEspacioBlanco
       movlw  .1
       goto   LCD_EnviaBlancos
LCD_DosEspaciosBlancos
       movlw  .2
       goto   LCD_EnviaBlancos
LCD_TresEspaciosBlancos
       movlw  .3
LCD_EnviaBlancos
       movwf  LCD_Auxiliar1       ; (LCD_Auxiliar1) se utiliza como contador.
LCD_EnviaOtroBlanco
       movlw  ' '                 ; Esto es un espacio en blanco.
       call   LCD_Caracter        ; Visualiza tanto espacios en blanco como se
       decfsz LCD_Auxiliar1,F     ; haya cargado en (LCD_Auxiliar1).
       goto   LCD_EnviaOtroBlanco
       return

LCD_Byte
       movwf  LCD_Auxiliar2       ; Guarda el valor de entrada.
       andlw  b'11110000'         ; Analiza si el nibble alto es cero.
       btfss  STATUS,Z            ; Si es cero lo apaga.
       goto   LCD_VisualizaAlto   ; No es cero y lo visualiza.
       movlw  ' '                 ; Visualiza un espacio en blanco.
       call   LCD_Caracter
       goto   LCD_VisualizaBajo

LCD_ByteCompleto
       movwf  LCD_Auxiliar2       ; Guarda el valor de entrada.
LCD_VisualizaAlto
       swapf  LCD_Auxiliar2,W     ; Pone el nibble alto en la parte baja.
       call   LCD_Nibble          ; Lo visualiza.
LCD_VisualizaBajo
       movf   LCD_Auxiliar2,W     ; Repite el proceso con el nibble bajo.

LCD_Nibble
       andlw  b'00001111'         ; Se queda con la parte baja.
       movwf  LCD_Auxiliar1       ; Lo guarda.
       sublw  0x09                ; Comprueba si hay que representarlo con letra.
       btfss  STATUS,C
       goto   LCD_EnviaByteLetra
       movf   LCD_Auxiliar1,W
       addlw  '0'                 ; El número se pasa a carácter ASCII sumándole
       goto   LCD_FinVisualizaDigito    ; el ASCII del cero y lo visualiza.
LCD_EnviaByteLetra
       movf   LCD_Auxiliar1,W
       addlw  'A'-0x0A            ; Sí, por tanto, se le suma el ASCII de la 'A'.
LCD_FinVisualizaDigito
       goto   LCD_Caracter        ; Y visualiza el carácter. Se hace con un "goto"
                                  ; para no sobrecargar la pila.
```

3.4.2 Teclado 4x4

El teclado de 4x4 se diseño con 16 Push-bottons (Figura 3.10) y 8 pines de conexión para controlar el teclado (Figura 3.11), la conexión de estos se muestra en la Figura 3.12.

Figura 3.10

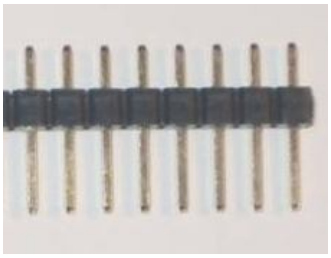


Figura 3.11

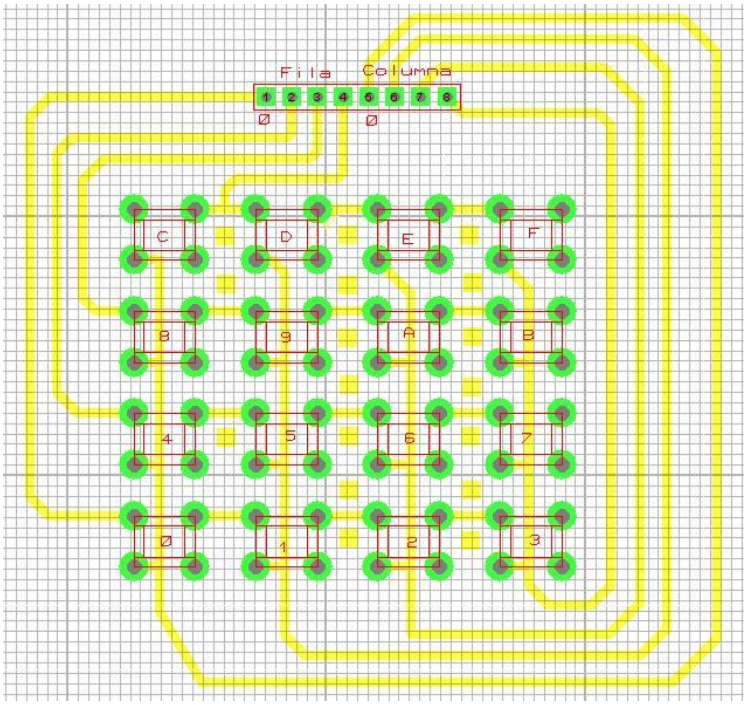


Figura 3.12

Diagrama de flujo para leer el orden la tecla presionada en el teclado matricial se muestra en la Figura 3.13.

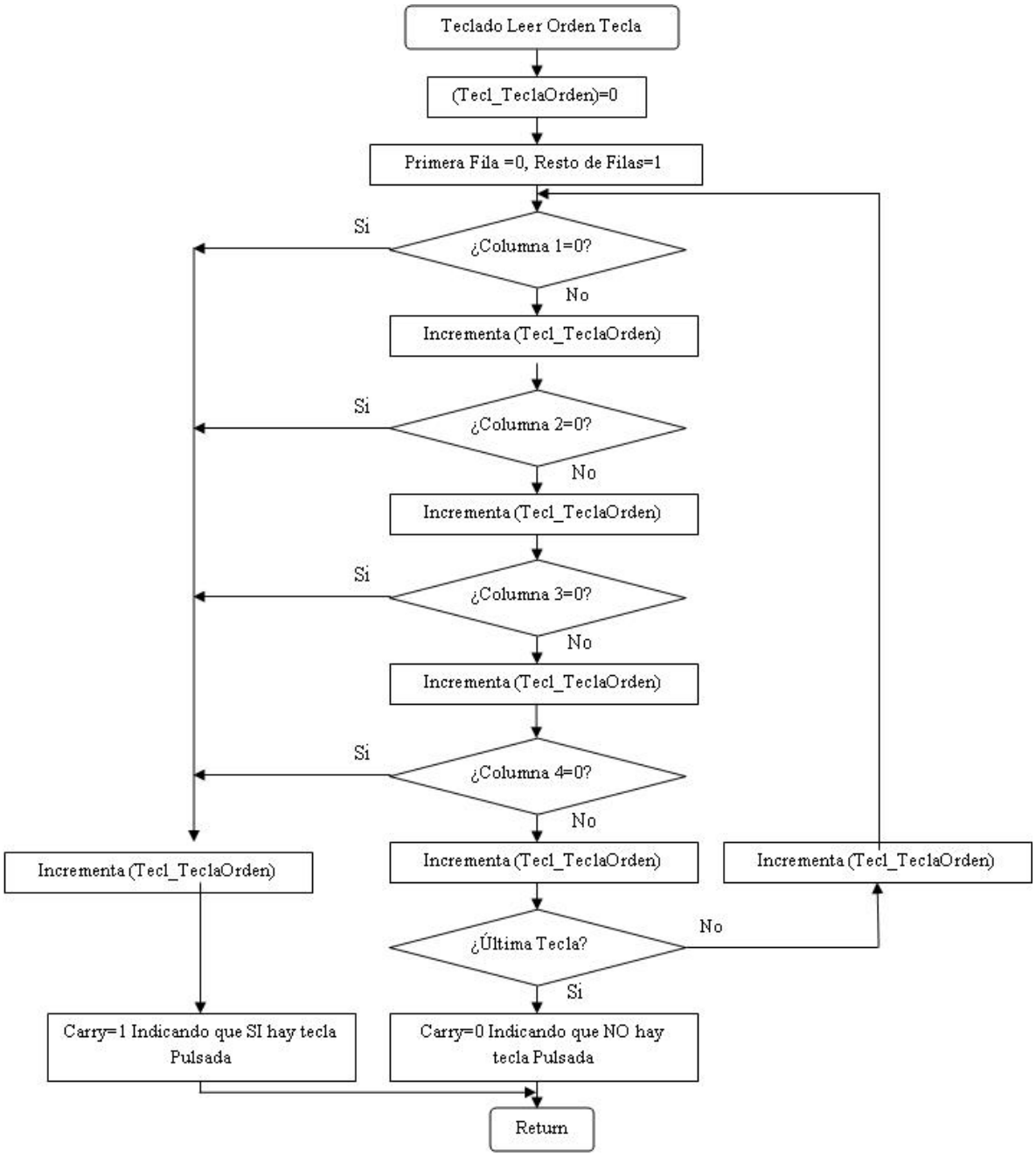


Figura 3.13

Librería para leer la tecla presionada según su orden.

```
            CBLOCK 0x00B
            Tecl_TeclaOrden
            ENDC
Tecl_UltimaTecla    EQU    d'9'                ;valor última tecla utilizada
Teclado_LeeHex
            call   Teclado_LeeOrdenTecla      ;leer Orden de la tecla Pulsada
            btfss  STATUS,C
            goto   Tecl_FinLeeHex
            call   Tecl_ConvierteOrdenHex
            bsf    STATUS,C
```

```
Tecl_FinLeeHex
            return
Tecl_ConvierteOrdenHex
            rlcf   WREG
            addwf  PCL
            retlw  0h
            retlw  1h
            retlw  2h
            retlw  3h
            retlw  4h
            retlw  5h
            retlw  6h
            retlw  7h
            retlw  8h
            retlw  9h
Teclado_FinTablaHex
Teclado_Inicializa
            bcf    INTCON2,NOT_RBPU    ;Habilita resistencias pull-up del PuertoB
            movlw  b'11110000'         ;1=entrada
            movwf  TRISB
            call   Teclado_EsperaDejePulsar
            return
Teclado_Comprobacion      EQU    b'11110000'
Teclado_EsperaDejePulsar
            movlw  Teclado_Comprobacion
            movwf  PORTB
Teclado_SigueEsperando                 ;Comprobación de que la tecla es pulsada
            call   Retardo_20ms
            movf   PORTB,W
            sublw  Teclado_Comprobacion
            btfss  STATUS,Z
            goto   Teclado_SigueEsperando
            return
Teclado_LeeOrdenTecla
            clrf   Tecl_TeclaOrden
            movlw  b'11111110'
Tecl_ChecaFila
            movwf  PORTB
            call   Retardo_1ms
Tecl_Columna1
            btfss  PORTB,4
            goto   Tecl_GuardaValor
            incf   Tecl_TeclaOrden,F
Tecl_Columna2
            btfss  PORTB,5
            goto   Tecl_GuardaValor
            incf   Tecl_TeclaOrden,F
Tecl_Columna3
            btfss  PORTB,6
            goto   Tecl_GuardaValor
            incf   Tecl_TeclaOrden,F
Tecl_Columna4
            btfss  PORTB,7
            goto   Tecl_GuardaValor
            incf   Tecl_TeclaOrden,F
Tecl_TerminaColumnas
            movlw  Tecl_UltimaTecla
            subwf  Tecl_TeclaOrden,W
            btfsc  STATUS,C
            goto   Tecl_NoPulsada
            rlcf   PORTB,W
            goto   Tecl_ChecaFila
Tecl_NoPulsada
            bcf    STATUS,C
            goto   Tecl_FinTecladoLee
Tecl_GuardaValor
            movf   Tecl_TeclaOrden,W
            bsf    STATUS,C
Tecl_FinTecladoLee
            return
```

A continuación se muestran las librerías auxiliares para las librerías ya mencionadas y para la programación que se mostrará después.

3.4.3 BinarioBCD

La conversión de un número en binario natural a BCD es por ejemplo el valor 124 expresado en binario natural sería 01111100, para expresarla en BCD hay que separar las centenas, decenas y unidades quedando 0001 0010 0100, la Figura 3.14 muestra el diagrama de flujo.

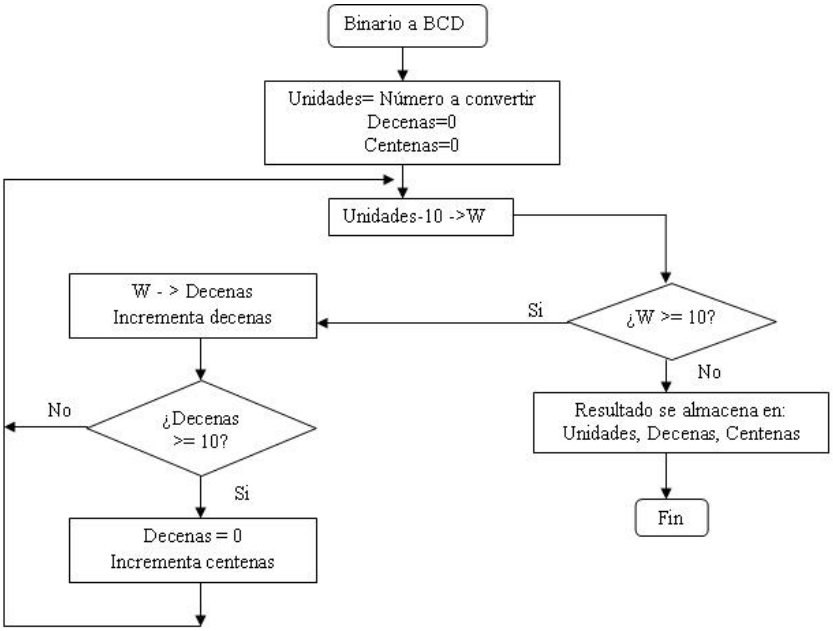


Figura 3.14

```
;un número binario natural de 8 bits es convertido a BCD
;el resultado es guardado en 3 posiciones de memoria BCD_Centenas
;BCD_Decenas, BCD_Unidades
;Entrada: en el registro W el número binario natural a convertir
;Salidas: en (BCD_Centenas)(BCD_Decenas)(BCD_Unidades)

                    CBLOCK 0x007
                    BCD_Centenas
                    BCD_Decenas
                    BCD_Unidades
                    ENDC

BIN_a_BCD
                    clrf        BCD_Centenas
                    clrf        BCD_Decenas
                    movwf       BCD_Unidades

BCD_Resta10
                    movlw       .10
                    subwf       BCD_Unidades,W
                    btfss       STATUS,C
                    goto        BIN_BCD_Fin
BCD_IncrementaDecenas
                    movwf       BCD_Unidades
                    incf        BCD_Decenas,F
                    movlw       .10
```

```
                    subwf       BCD_Decenas,W
                    btfss       STATUS,C
                    goto        BCD_Resta10
BCD_IncrementeCentenas
                    clrf        BCD_Decenas
                    incf        BCD_Centenas,F
                    goto        BCD_Resta10
BIN_BCD_Fin
                    swapf       BCD_Decenas,W
                    addwf       BCD_Unidades,W
                    return
```

3.4.4 Librería de Retardos

El tiempo que tarda en ejecutarse un programa depende de la frecuencia del oscilador conectado al microcontrolador y del número de ciclos máquina ejecutados. Un ciclo máquina es la unidad básica de tiempo que utiliza el microcontrolador. Para el PIC18F452 el ciclo máquina equivale a 4 ciclos de reloj, el tiempo que tarda en producirse un ciclo máquina es igual a cuatro veces el oscilador. Las instrucciones en el microcontrolador PIC18F452 necesitan un ciclo máquina para ejecutarse, excepto las de salto (goto, call, return, etc.) que necesitan dos ciclos máquina.

El tiempo que tarda el microcontrolador en ejecutar una tarea, está fijado por la fórmula 3.1:

$$Tiempo = 4\frac{1}{f}cm \qquad (3.1)$$

Siendo

- *f* es la frecuencia del oscilador
- *cm*, es el número de ciclos máquina que tarda en ejecutar una tarea

En este proyecto se utilizó un oscilador de cristal de cuarzo de 4MHz por lo que el ciclo máquina tiene una duración de 1µs.

Para generar tiempos de espera o retardos, se utilizaron subrutinas de retardo, basadas en instrucciones que se repiten las veces que sean necesarias, hasta conseguir el retardo requerido. Como se sabe el tiempo de ejecución de cada instrucción, se utilizo el registro R_ContA, que es el contador del número de iteraciones para obtener el tiempo de retardo deseado.

La subrutina que se va a ocupar está formada por las siguientes instrucciones:

```
Retardo_Xms                     ;call aporta 2 ciclos máquina
        movlw d'k'              ;Aporta un ciclo máquina
        movwf  R_ContA          ; Aporta un ciclo máquina
RXms_BucleInterno
        nop                     ;Aporta kx1 ciclos máquina
        decfsz  R_ContA,F       ;(k-1)x1cm (cuando no salta)+2cm (al saltar)
        goto    RXms_BucleInterno ;Aporta (k-1)x2 ciclos máquina
        return                  ;Salto de retorno aporta 2 cm
```

En total la subrutina tarda un tiempo de:

2+1+1+(kx1)+(k-1)x1+2+(k-1)2+2=5+4k

Para calcular el valor de k con que se cargará el registro R_ContA se utiliza la fórmula 3.2:

$$k = \frac{Tiempo - 5}{4}[\mu s] \qquad (3.2)$$

Esta es el razonamiento que se siguió para realizar la librería de retardos e incluso se anidaron subrutinas para retardos mas grandes (en segundos).

```
       CBLOCK 0x000
       R_ContA
       R_ContB
       R_ContC
       ENDC

Retardo_500micros          ; La llamada "call" aporta 2 ciclos máquina.
       nop                 ; Aporta 1 ciclo máquina sin hacer ninguna operación.
       movlw  .164         ; Aporta 1 ciclo máquina.
       goto   RetardoMicros; Aporta 2 ciclos máquina.
Retardo_200micros
       nop
       movlw  .64
       goto   RetardoMicros
Retardo_100micros
       movlw  .31
       goto   RetardoMicros
Retardo_50micros
       nop
       movlw  .14
       goto   RetardoMicros
Retardo_20micros
       movlw  .5
RetardoMicros
       movwf  R_ContA
Rmicros_Bucle
       decfsz R_ContA,F    ; (K-1)x1 cm (cuando no salta) + 2 cm (al saltar).
       goto   Rmicros_Bucle; Aporta (K-1)x2 ciclos máquina.
       return              ; El salto del retorno aporta 2 ciclos máquina.
       movlw  .200
       goto   Retardos_ms
Retardo_100ms
       movlw  .100
       goto   Retardos_ms
Retardo_50ms
       movlw  .50
       goto   Retardos_ms
Retardo_20ms
       movlw  .20
       goto   Retardos_ms
Retardo_10ms
       movlw  .10
       goto   Retardos_ms
Retardo_5ms
       movlw  .5
       goto   Retardos_ms
Retardo_2ms
       movlw  .2
       goto   Retardos_ms
Retardo_1ms
       movlw  .1

Retardos_ms
       movwf  R_ContB
R1ms_BucleExterno
       movlw  .249
       movwf  R_ContA
```

```
R1ms_BucleInterno
       nop
       decfsz R_ContA,F
       goto   R1ms_BucleInterno
       decfsz R_ContB,F
       goto   R1ms_BucleExterno
       return
Retardo_20s
       movlw  .200
       goto   Retardo_1Decima
Retardo_10s
       movlw  .100
       goto   Retardo_1Decima
Retardo_5s
       movlw  .50
       goto   Retardo_1Decima
Retardo_2s
       movlw  .20
       goto   Retardo_1Decima
Retardo_1s
       movlw  .10
       goto   Retardo_1Decima
Retardo_500ms
       movlw  .5
Retardo_1Decima
       movwf  R_ContC
R1Decima_BucleExterno2
       movlw  .100
       movwf  R_ContB
R1Decima_BucleExterno
       movlw  .249
       movwf  R_ContA
R1Decima_BucleInterno
       nop
       decfsz R_ContA,F
       goto   R1Decima_BucleInterno
       decfsz R_ContB,F
       goto   R1Decima_BucleExterno
       decfsz R_ContC,F
       goto   R1Decima_BucleExterno2
       return
```

3.5 Programas de cada etapa

Ya que se ha hecho referencia a las librerías en las que se apoyan los programas principales, a continuación se muestran los programas para cada etapa del sistema.

En la Figura 3.15 se encuentra el diagrama de flujo del funcionamiento general del sistema.

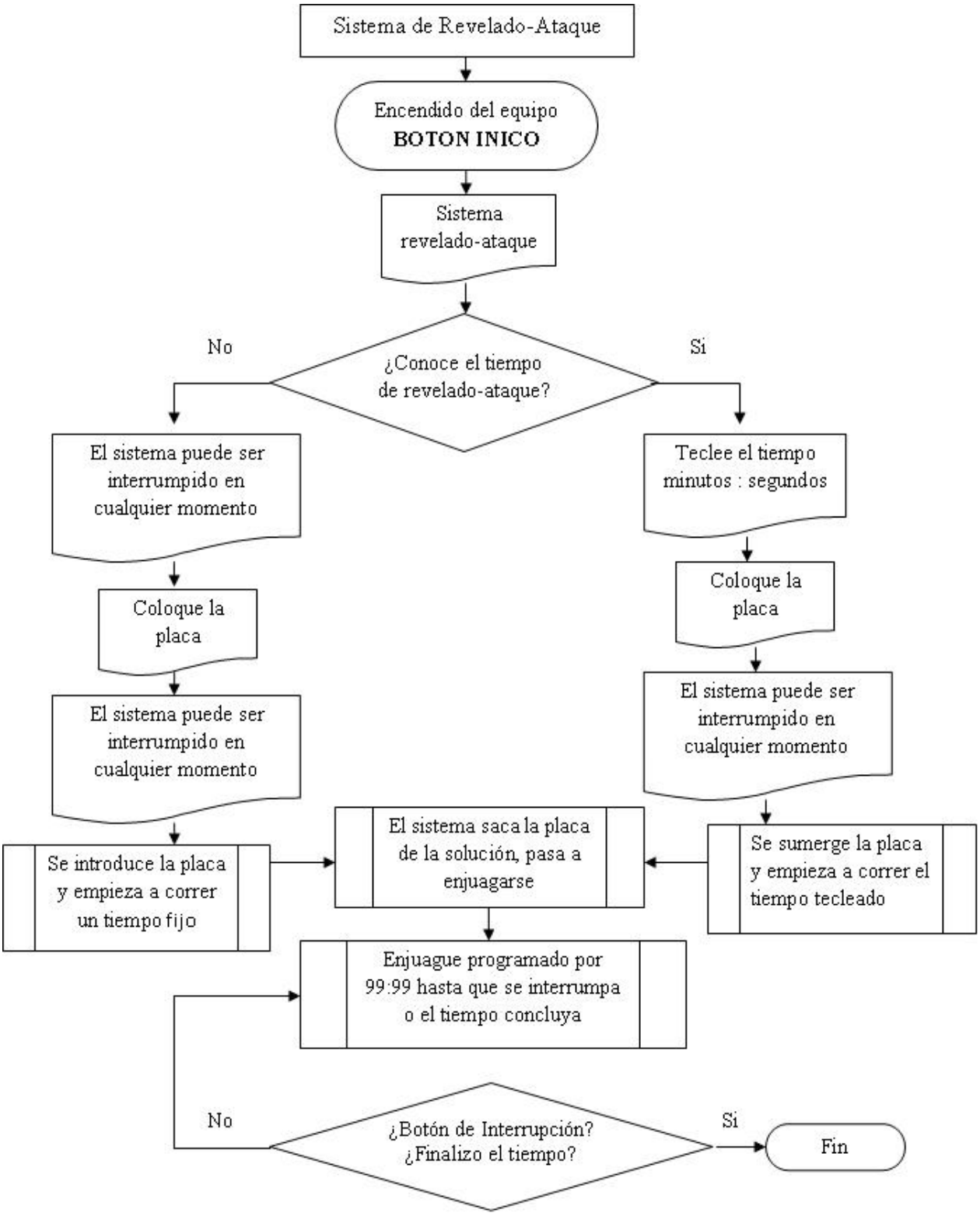

Sistema de Revelado-Ataque
Encendido del equipo
BOTON INICO
Sistema
revelado-ataque
¿Conoce el tiempo
de revelado-ataque?
No
Si
El sistema puede ser
interrumpido en
cualquier momento
Coloque la
placa
El sistema puede ser
interrumpido en
cualquier momento
Se introduce la placa
y empieza a correr
un tiempo fijo
Teclee el tiempo
minutos : segundos
Coloque la
placa
El sistema puede ser
interrumpido en
cualquier momento
Se sumerge la placa
y empieza a correr el
tiempo tecleado
El sistema saca la placa
de la solución, pasa a
enjuagarse
Enjuague programado por
99:99 hasta que se interrumpa
o el tiempo concluya
¿Botón de Interrupción?
¿Finalizo el tiempo?
No
Si
Fin

Figura 3.15

3.5.1Mensajes del menú

```
LCD1
      clrf   TBLPTRU
      clrf   TBLPTRH
      movlw  0x06
      movwf  TBLPTRH
      movlw  0x00
      movwf  TBLPTRL
LCD11 ;***"Revelado-Ataque"***
      TBLRD*+
      movf   TABLAT,W
      call   LCD_Caracter
      btfss  TBLPTRL,4
      goto   LCD11
LCD1linea2   ;***"R-A"****
      call   LCD_Linea2
      movlw  0x06
      movwf  TBLPTRH
      movlw  0x10
      movwf  TBLPTRL
LCD12
      TBLRD*+
      movf   TABLAT,W
      call   LCD_Caracter
      btfsc  TBLPTRL,4
      goto   LCD12
      return
LCD2
      call LCD_Borra
      call LCD_Linea1
      movlw  0x06
      movwf  TBLPTRH
      movlw  0x20
      movwf  TBLPTRL
LCD21 ;***"Colocar"****
      TBLRD*+
      movf   TABLAT,W
      call   LCD_Caracter
      btfss  TBLPTRL,4
      goto   LCD21
LCD2linea2
      call   LCD_Linea2
      movlw  0x06
      movwf  TBLPTRH
      movlw  0x30
      movwf  TBLPTRL
LCD22 ;***"Sustancias"***
      TBLRD*+
      movf   TABLAT,W
      call   LCD_Caracter
      btfsc  TBLPTRL,4
      goto   LCD22
      return
LCD3
      call LCD_Borra
      call LCD_Linea1
      movlw  0x06
      movwf  TBLPTRH
      movlw  0x40
      movwf  TBLPTRL
LCD31 ;***"Colocar placa "****
      TBLRD*+
      movf   TABLAT,W
      call   LCD_Caracter
      btfss  TBLPTRL,4
      goto   LCD31
      return
LCD4
```

```
       call   LCD_Borra
       call   LCD_Linea1
       movlw  0x06
       movwf  TBLPTRH
       movlw  0x50
       movwf  TBLPTRL
LCD41  ;"Interrupcion del"
       TBLRD*+
       movf   TABLAT,W
       call   LCD_Caracter
       btfsc  TBLPTRL,4
       goto   LCD41
LCD4linea2
       call   LCD_Linea2
       movlw  0x06
       movwf  TBLPTRH
       movlw  0x60
       movwf  TBLPTRL
LCD42  ;***"equipo con Alto"***
       TBLRD*+
       movf   TABLAT,W
       call   LCD_Caracter
       btfss  TBLPTRL,4
       goto   LCD42
       return
LCD6
       call LCD_Borra
       call LCD_Linea1
       movlw  0x06
       movwf  TBLPTRH
       movlw  0x90
       movwf  TBLPTRL
LCD61  ;***"Teclear tiempo"****
       TBLRD*+
       movf   TABLAT,W
       call   LCD_Caracter
       btfsc  TBLPTRL,4
       goto   LCD61
LCD6linea2
       call   LCD_Linea2
       movlw  0x06
       movwf  TBLPTRH
       movlw  0xA0
       movwf  TBLPTRL
LCD62  ;***"minutos:segundos"***
       TBLRD*+
       movf   TABLAT,W
       call   LCD_Caracter
       btfss  TBLPTRL,4
       goto   LCD62
       return
LCD7
       call   LCD_Borra
       call   LCD_Linea1
       movlw  0x06
       movwf  TBLPTRH
       movlw  0xB0
       movwf  TBLPTRL
LCD71  ;***"Fin"***
       TBLRD*+
       movf   TABLAT,W
       call   LCD_Caracter
       btfsc  TBLPTRL,4
       goto   LCD71
       return

       org    0x600              ;Memoria donde se guardan los mensajes
       db     "Revelado-Ataque "  ;1
       db     "     (R-A)      "  ;1.2
       db     "     Colocar    "  ;2
```

```
       db     "   Sustancias   "  ;2.2
       db     " Colocar placa  "  ;3.1
       db     "Interrupcion del"  ;4.1
       db     " equipo con Alto"  ;4.2
       db     " Teclear tiempo "  ;6.1
       db     "minutos:segundos"  ;6.2
       db     "      Fin       "  ;7
```

3.5.2 Introducir el tiempo de revelado o ataque

Programación para introducir el tiempo a programar ser mostrado en el LCD e iniciar el temporizador.

```
   CBLOCK    0x014
   ContadorP
   ENDC

#DEFINE      siguiente    PORTD,4      ;Pin 27 ->
#DEFINE      no           PORTD,5      ;Pin 28 No

       ORG          0x0000
       goto         Inicio
       ORG          0x0008       ; Vector de Interrupciones de  Prioridad Alta
       goto   ISegunda
       ORG    0X0016
       goto   IPrimera
IPrimera
       call   Teclado_LeeHex
       movwf  POSTINC0
       call   LCD_Nibble
       call   Teclado_EsperaDejePulsar
       incf   ContadorP,F
       movlw  .2
       cpfslt ContadorP
       goto   Puntitos
       goto   FinInterrupcion
FinInterrupcion
       bcf    INTCON,0           ;limpia la bandera de interrupcioón del PuertoB
       retfie
Puntitos
       subwf  ContadorP,0,0
       btfsc  STATUS,Z
       goto   Manda
       call   Teclado_EsperaDejePulsar
       goto   FinInterrupcion
Manda
       movlw  ':'
       call   LCD_Caracter
ISegunda
       movlw   0x0B              ;1 seg
       movwf   TMR0H
       movlw   0xDB              ;1 seg
       movwf   TMR0L             ;cargo nuevamente 3035 (0x0BDB)
       call   MuestraRE
       bcf          INTCON,TMR0IF;Bit3 Borro la bandera
       retfie
Inicio
       call   LCD_Inicializa
       bsf    TRISD,4            ; siguiente
       bsf    TRISD,5            ; no
       call   LCD1               ;***"Revelado-Ataque"***
                                 ;***"R-A"****
       call   Retardo_1s
       call   LCD2               ;***"Colocar"****
                                 ;***"Sustancias"***
       call   Retardo_1s
YA     btfss  siguiente;(->)
```

```
       goto   YA
       call   LCD3                ;***"Colocar placa "****
       call   Retardo_1s
YA1    btfss  siguiente           ; (->)
       goto   YA1
       call   LCD4                ;***"Interrupcion del"**
                                  ;***"equipo con Alto"***
       call   Retardo_1s
YA2    btfss  siguiente
       goto   YA2
       call   LCD5                ;***"¿Tiempo R-A?"****
                                  ;***"Si(->)  No(No)"***
       call   Retardo_1s

pregunta
       btfsc  siguiente
       goto   PedirTiempo
       btfsc  no
       goto   carga
       goto   pregunta

PedirTiempo
       call   LCD6
       call   Retardo_1s
ya     btfsc  siguiente
       goto   Listo1
       goto   ya

carga
       movlw  .0
       movf   Segundos2
       movlw  .60
       movf   Minutos2
       goto   Inicio2

Listo1
       call   Teclado_Inicializa
       bsf    RCON,IPEN           ;IPEN=Bit7
       movlw  b'11001000'         ;habilita interrupciones bit 7 y 6
       movwf  INTCON              ;bit3 habilita interrupciones PuertoB
                                  ;limpio banderas TMR0IF-Bit2,INRBIF;Bit0
       movlw  b'00000100'         ;high priority bit2 Timer0
       movwf  INTCON2             ;low priority bit0 PuertoB
       clrf   ContadorP
       call   LCD_Borra
       movlw  .5
       call   LCD_PosicionLinea1
       lfsr   FSR0,0X100

Principal
       btfsc  FSR0L,2
       goto   Segunda
       goto   Principal

Segunda
    bcf       INTCON,3           ;deshabilito interrupcion por el PuertoB
    call      RecuperarRe
    call      Retardo_1s

Inicio2                           ;Desbordamiento
       movlw  b'00000011'         ;prescale 1:16 modo 16 Bits para 1seg
       movwf  T0CON
       movlw  b'10100000'         ;habilta las interruciones y la del Timer0
       movwf  INTCON
       bsf    INTCON2,TMR0IP      ;Bit2 Interrupcion alta prioridad
InicioTemporizador
       btfsc  PORTD,4
       goto   InicioTimer0
       goto   InicioTemporizador
InicioTimer0
```

```
      bsf    T0CON,TMR0ON       ;habilita Timer0 Bit7
      movlw  0x0B               ;1 seg
      movwf  TMR0H              ;65535-62500=3035(0x0BDB)   para 1 segundo
      movlw  0xDB               ;1 seg
      movwf  TMR0L              ;valores para que en 1 segundo se interrupa

Bucla
      movlw .0
      cpfsgt Minutos2
      goto   PuedeSerFin
      goto   Bucla
PuedeSerFin
      movlw .0
      cpfsgt Segundos2
      goto   Fin
      goto   Bucla
Fin
      bcf    T0CON,TMR0ON       ;deshabilita Timer0 Bit7
      sifin
      btfsc  siguiente
      call   LCD7
      goto   sifin
      goto   Inicio

   CBLOCK    0x010
    Minutos
    Minutos2
    Segundos
    Segundos2
    ENDC

RecuperarRe
      lfsr   FSR0,0X100         ;Recupero valores de la memoria RAM
      movf   POSTINC0,WREG
      movwf  Minutos
      movf   POSTINC0,WREG
      movwf  Minutos2
      movf   POSTINC0,WREG
      movwf  Segundos
      movf   POSTINC0,WREG
      movwf  Segundos2
      movf   Minutos,WREG       ;Convierto a Minutos
      mullw  .10                ;Multiplico por 10 y sumo
      movf   PRODL,WREG
      movwf  Minutos
      addwf  Minutos2,F
      movf   Segundos,WREG      ;Convierto a Segundo
      mullw  .10                ;Multiplico por 10 y sumo
      movf   PRODL,WREG
      movwf  Segundos
      addwf  Segundos2,F
      return

MuestraRE                       ;mostrar reloj
      movlw .0
      cpfseq Segundos2          ;Compare Segundos2 with 0, skip =
      goto   NoEsCero
      goto   SegundoEsCero
NoEsCero
      movf   Segundos2,WREG     ;Segundos->WREG
      sublw  .99                ;99-(wreg)->wreg
      btfss  STATUS,C           ;Si C=0      ResultadoNegativo o Cero
      goto   SegundoEsCero      ;Si C=1      ResultadoPositivo
Decr
   decfsz    Segundos2,F        ;decrement Segundos2 skip if Segundos2=0
   goto      SalirI
SegundoEsCero
      movlw .0
      cpfseq Minutos2           ;Compare Minutos2 with 0, skip=0
      goto   Continua1
```

```
       goto   SalirI
Continua1
       decf   Minutos2,F
       movlw  .59
       movwf  Segundos2
SalirI
       movlw  .5
       call   LCD_PosicionLinea2
       movf   Minutos2,WREG
       call   BIN_a_BCD
       call   LCD_ByteCompleto
       movlw  ':'
       call   LCD_Caracter
       movf   Segundos2,WREG
       call   BIN_a_BCD
       call   LCD_ByteCompleto
       return
```

3.5.3 Programación para motores

```
PWM1   ;para motor a pasos
       movlw  .88          ;1) PWM periodo
       movwf  PR2
       movlw  .44          ;2) duty cycle
       movwf  CCPR1L
       bcf    TRISC,2      ; 3)configuración como salida
       movlw  b'00000110'  ;4) Bit 2 Timer 2 on, bit 1y0 preescaler=16
       movwf  T2CON
       movlw  b'00101100'  ;5) nunca cambia configuración (44)
       movwf  CCP1CON
       call   Retardo_2s
       bsf    TRISC,2      ;desactivar movimiento del motor a pasos
PWM2   ;para servomotor P=4.096ms f=244.14Hz
PrimeraP
       movlw  .255         ;1) PWM periodo
       movwf  PR2
       movlw  .55          ;2) duty cycle, va a cambiar para posicionar servomotor
       movwf  CCPR2L
       bcf    TRISC,1      ; 3)configuración como salida
       movlw  b'00000110'  ;4) Bit 2 Timer 2 on, bit 1y0 preescaler=16
       movwf  T2CON
       movlw  b'00101100'  ;5) nunca cambia configuración (44)
       movwf  CCP2CON
       call   Retardo_1s
       movlw  .180
       movwf  CCPR2L
       call   Retardo_1s
       bcf    T2CON,2
```

3.6 PCBs

Los circuitos fueron hechos en placas fenolicas para preservar su funcionamiento y hacer más compacto el sistema, los circuitos PCB's se hicieron en las siguientes etapas:

- Fuentes de Poder de 5V y 10V (Figura 3.16)

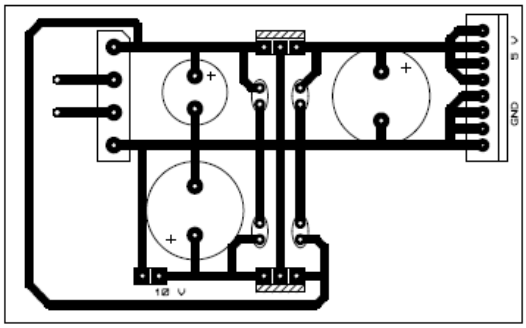


Figura 3.16

- Tarjeta para PIC18F452 (Figura 3.17)

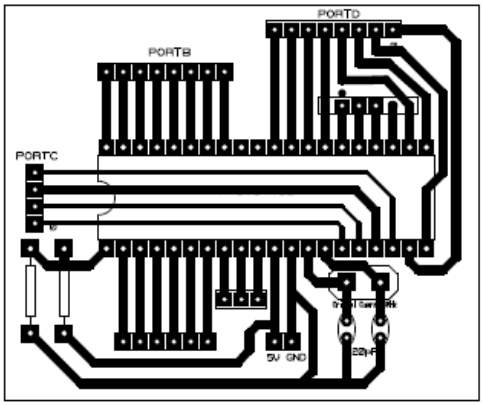


Figura 3.17

- Etapa de potencia (para motor a pasos y servomotor)

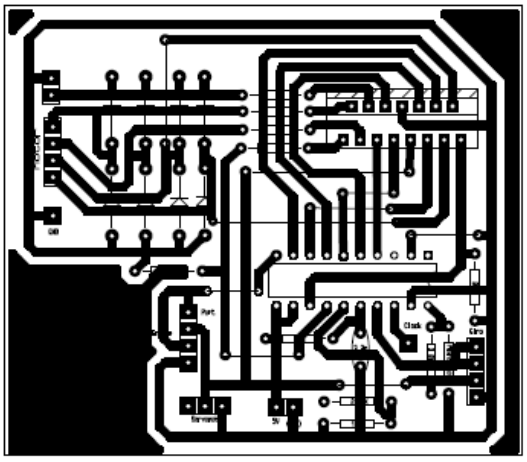


Figura 3.18

- Placa para conexión de Display LCD 16x2 y teclado matricial (Figura 3.19)

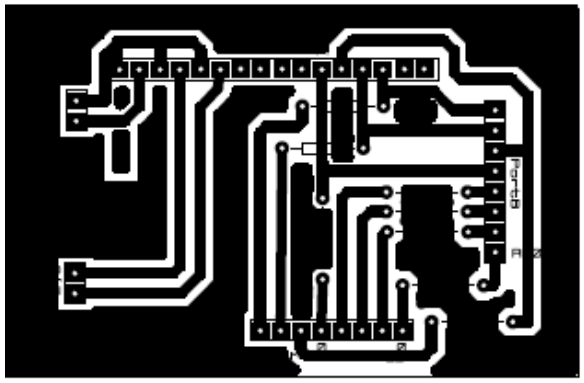


Figura 3.19

- Teclado matricial 4x4 (Figura 3.20)

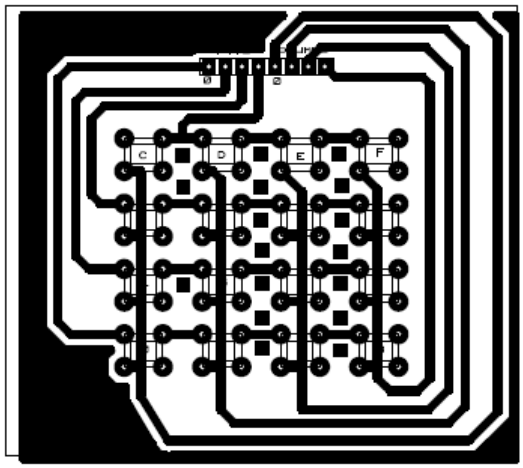


Figura 3.20

Capítulo 4 Diseño Mecánico

4.1 Motores

Un motor es la parte de una máquina capaz de hacer una acción, transformando algún tipo de energía (eléctrica, de combustibles fósiles, etc.), en energía mecánica capaz de realizar un trabajo.

Los motores eléctricos, realizan trabajo a partir de una corriente eléctrica, utilizan la inducción electromagnética para producir movimiento, según sea la constitución del motor: núcleo con cable arrollado, sin cable arrollado, monofásico, trifásico, con imanes permanentes o sin ellos; la potencia depende del calibre del alambre, las vueltas del alambre y la tensión eléctrica aplicada.

La Figura 4.1 muestra de modo esquemático las partes principales de un motor de corriente continua.

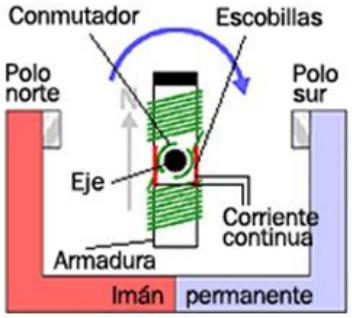


Figura 4.1

El elemento situado en el centro es la parte del motor que genera el movimiento, se llama rotor, y consiste en un electroimán que puede girar libremente entorno a un eje, está rodeado por un imán permanente, cuyo campo magnético permanece fijo.

El electroimán recibe la corriente a través del contacto establecido entre las escobillas y el conmutador, las escobillas permanecen fijas, mientras que el conmutador puede girar libremente entre ellas siguiendo el movimiento del rotor.

Algunos parámetros para saber la eficiencia de un motor son:

- Rendimiento: cociente entre la potencia útil que genera y la potencia absorbida (η).
- Velocidad nominal: número de revoluciones por minuto (rpm o RPM) a las que gire (n).
- Potencia: trabajo que el motor es capaz de realizar en un tiempo y velocidad de giro determinadas, se mide en caballos de vapor (CV).
- Par motor: momento de rotación que actúa sobre el eje del motor y determina su giro. Se mide en newtons-metro.

4.2 Motor a pasos

Los motores a pasos tienen muchas aplicaciones, debido a la gran precisión que tienen. Estos motores se pueden mover desde un paso hasta el número de pasos que se le indiquen, dependiendo de la cantidad de pulsos que se apliquen, estos pasos pueden ser desde 1.8° hasta 90°. Estos motores se pueden quedar en una posición, si una o más de sus bobinas están energizada o totalmente libres

si no hay corriente circulando por estas. Están constituidos por un rotor sobre el que se aplican distintos imanes permanentes y un cierto número de bobinas excitadoras bobinadas en su estator. Las bobinas son parte del estator y el rotor es un imán permanente. La conmutación o excitación de las bobinas debe ser manejada externamente por un sistema de control.

En la figura 4.2 se muestran los tipos de motores a pasos que existen.

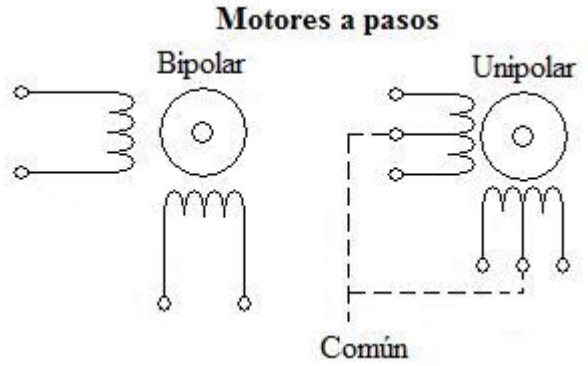


Figura 4.2

Los motores unipolares suelen tener 6 o 5 cables de salida, 4 cables reciben los pulsos que indican la secuencia y duración de los pasos y los otros 2 son para alimentación. Hay tres secuencias de movimiento para estos motores: un paso por vez para torque alto, un paso por vez torque bajo y medio paso.

Los motores Bipolares generalmente tienen 4 cables de salida, necesitan inversión de la corriente que circula en sus bobinas en una secuencia determinada. Cada inversión de la polaridad provoca el movimiento del eje en un paso, cuyo sentido esta determinad por la secuencia seguida.

Para identificar las bobinas de un motor, se puede medir la resistencia entre los cables, los cables que tengan alguna resistencia conforman una bobina y los cables que tengan resistencia infinita no son una bobina.

4.2.1 Desplazamiento vertical

El motor utilizado para el desplazamiento vertical es un motor a pasos bipolar, ver Figura 4.3, este fue elegido por la alta frecuencia a la que responde y porque tiene el torque necesario para realizar el movimiento vertical del porta placa.

Figura 4.3

Este motor a pasos tiene las especificaciones eléctricas mostradas en la Tabla 4.1 y su diagrama eléctrico se muestra en la Figura 4.4.

Modelo	Paso en ángulos	Longitud	Voltaje nominal	Corriente nominal	Resistencia de fase	Inductancia de fase	Par de mantenimiento	No. de conductores	Inercia del rotor	Par de retención	Peso
	°	mm	V	A	Ω	mH	g-cm		g-cm²	g-cm	kg
42BYG HW811	1.8	48	3.1	2.5	1.25	1.8	4800	4	68	280	0.38

Tabla 4.1

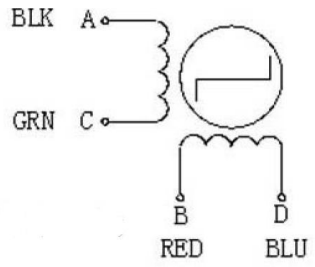


Figura 4.4

En la Figura 4.5 se muestra el diagrama de bloques del sistema mecánico del desplazamiento vertical.

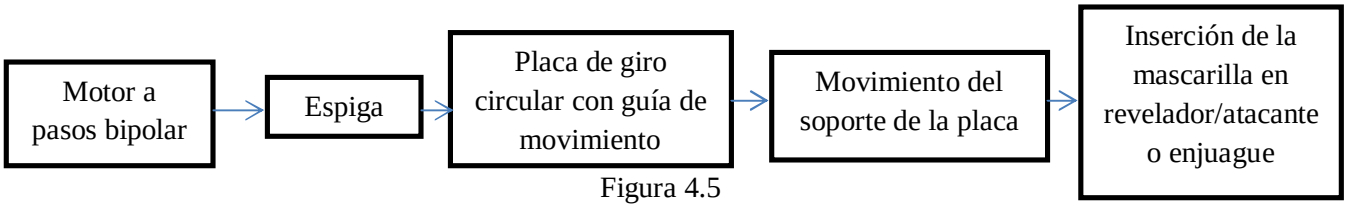


Figura 4.5

Las piezas que conforman el sistema mecánico para este desplazamiento son mostradas en la Figura 4.6.

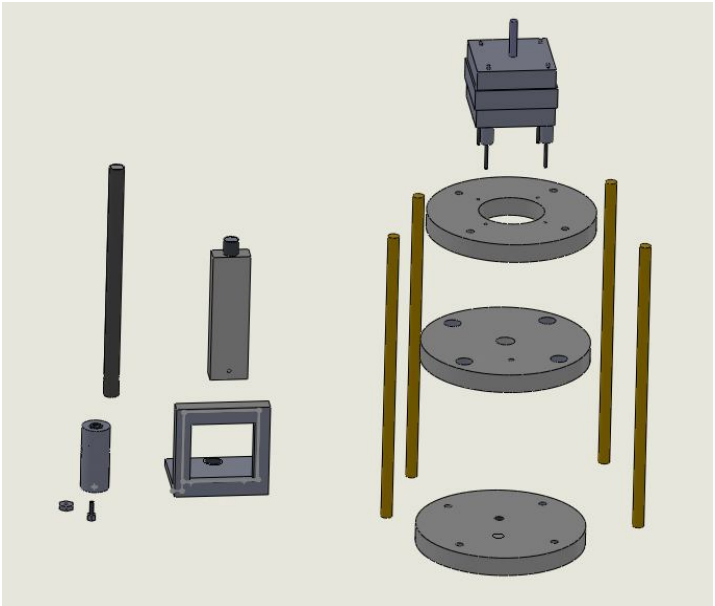


Figura 4.6

El motor a pasos bipolar esta acoplando al eje del motor con la espiga (eje de propulsión), este acoplamiento consiste en un cilindro de aluminio de 3.5 cm de largo, perforado de un lado al diámetro del eje del motor y del otro lado perforado al diámetro de la espiga y machuelado (para el enroscamiento interno) con el machuelo 3/8 que es el correspondiente a la rosca que tiene la espiga. El cilindro de acoplamiento se colocó a presión en el eje del motor. Para la sujeción de este acoplamiento con la espiga se requirió perforar perpendicularmente al eje del motor, de lado a lado, para sujetar con un tornillo que a su vez es sujetado por una tuerca, ver Figura 4.7.Esto fue requerido porque la espiga solía separarse del acoplamiento después haber realizado varios cambios de desplazamiento, hacia arriba y hacia abajo.

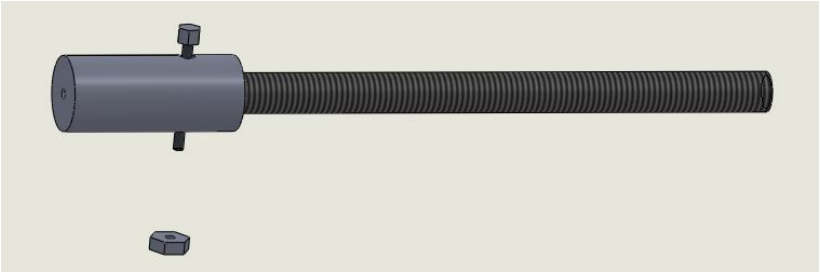


Figura 4.7

El motor a pasos fue sujeto en la parte alta, en un cilindro de teflón que acondicionado para sujetar tanto al motor como a los 4 postes de guía. La sujeción del motor se realizó por sus cuatro extremos con tornillos y amortiguadores antivibratorios, ver Figura 4.8.

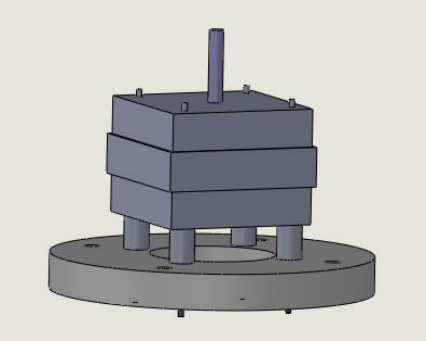


Figura 4.8

Los amortiguadores antivibratorios (Figura 4.9) se utilizan para la atenuación de vibraciones internas de una máquina (motores, bombas, grupos hidráulicos) o bien para permitir la vibración de partes de una máquina y que esta vibración no se transmita al exterior o se transmita lo menos posible, en este caso su aplicación se refiere al segundo uso, es decir, al permitir que el motor se esté moviendo por encima del circulo se evita que la espiga acoplada al eje del motor cabeceé y se genere menos fricción con los postes guía; permitiendo que el motor no se force y el movimiento sea rápido y sin dificultad.

Figura 4.9

La siguiente pieza es otro cilindro (Figura 4.10), también fabricada en teflón, tiene un orificio en el centro, con el diámetro de la rosca que tiene la espiga para que se transmita el movimiento. Además tiene otros cuatro orificios en los extremos para los postes, esto es para que esta placa no gire alrededor de la espiga, por ultimo un orificio lateral con rosca para sujetar al porta placa.

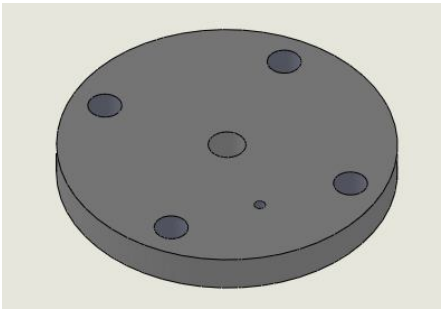


Figura 4.10

El porta placa para la mascarilla fue fabricado en teflón, por ser un material muy difícil de corroer y que resiste altas temperaturas. Las medidas de éste están de acuerdo a una placa para mascarilla de tamaño de 5x5 cm. Pero el porta placa es desprendible, por lo que en caso de una placa más pequeña o más grande se puede cambiar, ver Figura 4.11.

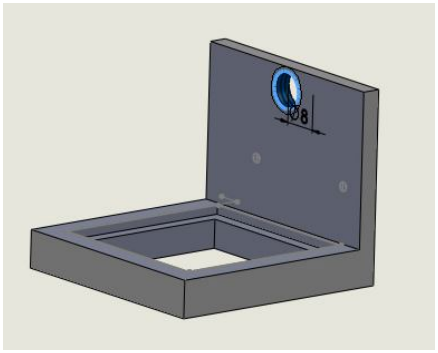


Figura 4.11

La Figura 4.12 muestra el sistema para el desplazamiento vertical completo.

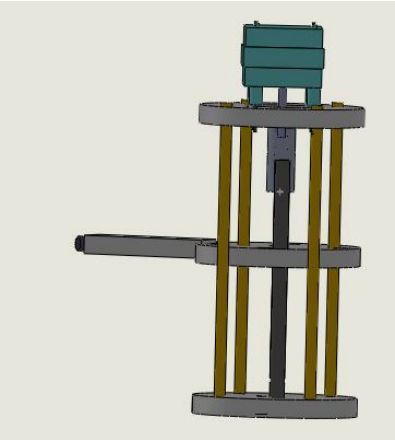


Figura 4.12

4.3 Desplazamiento horizontal

Para el desplazamiento horizontal, se utilizó un servomotor analógico. Es un motor de corriente continua que tiene la capacidad de controlar su posición por medio de un PWM. Es capaz de ubicarse en cualquier posición dentro de un rango de operación de 180º y mantenerse estable en dicha posición.

En general, los servos suelen estar compuestos por 4 elementos fundamentales:

Motor de corriente continua (DC): Elemento que le brinda movilidad al servo.
Engranajes reductores: Tren de engranajes que se encarga de reducir la alta velocidad de giro del motor DC para acrecentar su capacidad de torque o par motor.
Sensor de desplazamiento: potenciómetro colocado en el eje de salida del servo que se utiliza para conocer la posición angular del motor.
Circuito de control: Es una placa electrónica que implementa un control de posición por realimentación. Este circuito compara la señal de entrada de referencia (posición deseada) con la posición actual medida por el potenciómetro. La diferencia entre la posición actual y la deseada es amplificada y utilizada para mover el motor en la dirección necesaria para reducir el error.

Los servos tienen tres cables (Figura 4.13), 2 de alimentación y un cable de control que indica la posición deseada al circuito de control mediante señales PWM (Pulse Width Modulation).

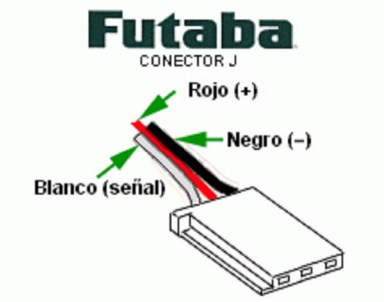


Figura 4.13

Las señales PWM utilizadas para controlar los servos están formadas por pulsos positivos cuya duración es proporcional a la posición deseada del servo y que se repiten a un periodo

recomendable de 20ms (50Hz). Se pueden mover en un rango extendido de 180º y los pulsos de control varían entre 0.5 y 2.5ms. Para mantener fijo un servomotor en una posición habrá que enviar periódicamente el pulso correspondiente, si no recibe señales, el eje del servomotor quedará libre y se podrá mover.

En este trabajo se ocupa el servomotor Futaba FP-S148 (Figura 4.14) con las siguientes especificaciones:

Control con PWM, a un periodo recomendado de 1.52ms
Motor de 3-poles
Alimentación 4.8V o 6V
Corriente a 6V de 8mA
Torque: 33 oz-in (2.4 kg-cm) @ 4.8V
42 oz-in (3.0 kg-cm) @ 6V
Tiempo de transmisión: 0.28 sec/60° @ 4.8V
0.22 sec/60° @ 6V
Dimensiones: (40 x 20 x 36mm)
Peso: 44g

Figura4.14

4.3.1 PWM

La generación del control PWM fue realizada a una frecuencia de 700 Hz, para las posiciones de 0 a 180°. Esta señal se implementó con el PIC18F452 a través de uno de sus puertos.

Las piezas que conforman el sistema mecánico para el desplazamiento horizontal son mostradas en la Figura 4.15.

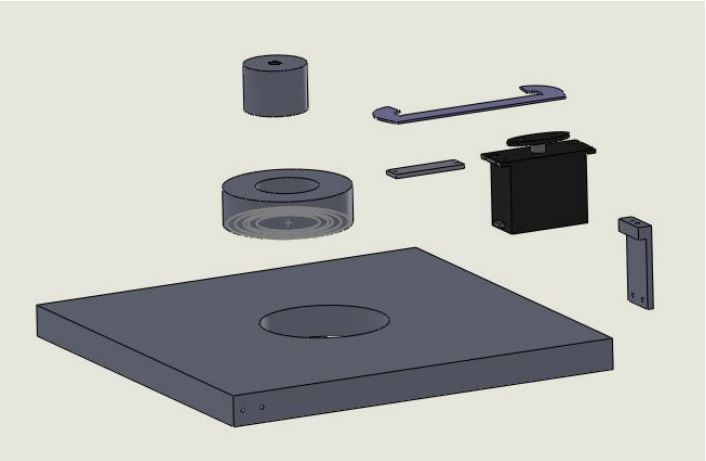


Figura 4.15

La manera en que el servomotor es acoplado con las piezas mecánicas, se muestra en el diagrama de bloques de la Figura 4.16.

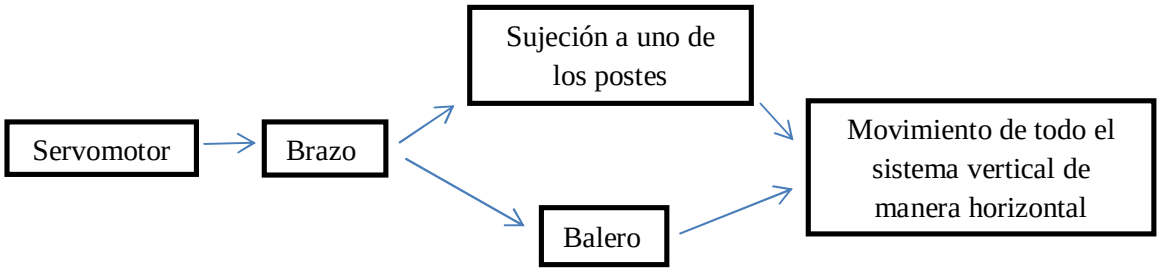


Figura 4.16

El brazo del servomotor fue extendido con una placa de aluminio para que el alcance del movimiento fuera mayor y pudiera acoplarse con el sistema de desplazamiento vertical, esta unión es a través de uno de los postes que sirven como guía y son parte del sistema vertical.

El sistema vertical fue colocado sobre un balero de 6.1 cm de diámetro externo. Lo que hace, que con un giro del servomotor todo el sistema se mueva de manera muy rápida de izquierda a derecha, el uso del balero también permitió agregar una etapa de potencia para el servomotor.

El balero fue colocado a presión en el centro de una placa de aluminio de 20 x 20 cm, esta placa sirve también de soporte a todo el sistema. Para acoplar el balero con el sistema vertical fue metido a presión un cilindro de aluminio de 3 cm de largo en el centro del balero, a este cilindro se le hizo cuerda en la parte superior para que con un tornillo se sujete el sistema vertical. Como la longitud del cilindro es solo 0.4 cm mayor que el que la placa de aluminio el sistema no se tambalea, quedando muy estable.

Figura 4.18 muestra el sistema para el desplazamiento horizontal.

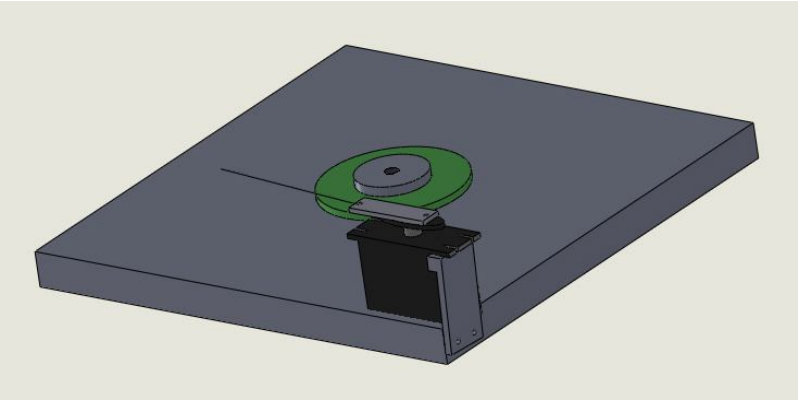


Figura 4.18

Todas las piezas fueron manufacturadas en el CCADET. A excepción de la placa para el balero, las otras piezas fueron hechas en la sección mecánica del grupo académico fotónica de microondas, que cuenta con una fresadora, un torno, una fresadora pequeña, un tornillo mecánico, brocas, buriles, machuelos, maneral y herramientas de uso general.

La placa para el balero fue hecha en el taller mecánico de la sección de prototipos del CCADET por ser la pieza más grande y requerir un sostén de fresadora más grande para sostener la placa mientras se trabajaba.

Cuando se empezó a manufacturas de las piezas se tenía una idea de cómo iba a ser el sistema, sin embargo, sobre la marcha se fueron realizando muchas modificaciones para acoplar los motores, la parte electrónica, los materiales disponibles y para mejorar la calidad de las piezas.

La Figura 4.19 muestra al sistema completo.

Figura 4.19.

Conclusiones

He adquirido experiencia en el desarrollo de un sistema que es implementado desde cero, al principio pensaba que si desarrollaban todas las etapas que iba a necesitar y luego las probaba juntas, funcionarían, ahora sé que se requiere un acoplamiento entre las partes. En el caso de desarrollar un proyecto pequeño, como es este, creo que lo mejor sería ir desarrollando todo en cascada, es decir empezar con alguna parte e ir uniendo las demás hasta que todo funcione.

Cuando comencé este proyecto tenía una idea muy distinta de cómo iba a ser, a como es ahora, pensé que la programación del pic iba a ser muy poca, se convirtió en demasiada y tuvé que cambiar a un pic con mayor capacidad, pero ahora que he aprendido más de los pics, sé que podría regresar al pic con que hice las primeras pruebas. Esto sucedió porque me fue muy complicado reunir suficiente información para la programación de pics, y aun más difícil encontrar ejemplos de programación, mucho de lo que tuve que desarrollar fue sin una base previa, lo que hizo que me llevara mucho tiempo pero fue una buena experiencia para saber que es desarrollar algo desde cero y supuso un reto porque nunca he sido hábil para programar.

Respecto a las partes de la estructura mecánica, estas cambiaron por completo la primera estructura era muy rígida y se termino con una estructura más flexible para que pudiera mantenerse sobre el eje de la espiga y sin hacer vibrar al porta placa.

Para la realización de la estructura tuve que aprender a usar de manera básica el torno, la fresadora y las herramientas que estas requieren, pero me di cuenta de la importancia de que estas máquinas se encuentren calibradas y del gran papel que juega la experiencia, ya que el ajuste no va a ser perfecto pero se puede corregir mucho el maquinado si se han hecho otras piezas con anterioridad. Las piezas de la estructura realizada no es perfecta pero funciona adecuadamente gracias a los amortiguadores antivibratorios y a que las demás piezas están lo más simétricas que se pudo maquinar.

La rapidez del sistema para desplazar la placa dentro del revelador o el atacante y después al enjuague es igual a la que un usuario con experiencia le tomaría hacer estos desplazamientos. El sistema tiene como ventajas:

- Fija la posición de los recipientes con el revelador o atacante y el recipiente del enjuague, evitando pérdida de tiempo si estos recipientes no están cerca o a la menor distancia posible.
- Con el sistema no resbalará la placa de las manos del usuario.
- Para un usuario sin experiencia este proceso es más sencillo.
- Se puede cronometrar el tiempo de revelado o ataque.
- Se puede lograr con mayor facilidad un resultado de calidad similar para distintas placas.

Con este sistema se da un prototipo para realizar un sistema que cumpla con los requisitos necesarios para operar dentro de un cuarto limpio y también como una referencia para realizar un sistema para revelado y ataque para dispositivos en el orden de nanómetros o magnitudes inferiores para futuros desarrollos de circuitos.

De manera personal, he aprendido que la mejor forma para que aprenda algo es estudiando lo básico de teoría tratar de hacer algo práctico y si necesito realizar algo más complicado o no funciona, entonces si recurrir a la teoría, de otra forma me es más tardado comprender algo.

Referencias

Burn J. Lin, Mark A. McCord. (1997). Handbook of Microlithografhy, Micromachining and Microfabrication. Washington USA: A publication of the international Society for Optical Engineering. Editorial Bellingham.

Mandado Enrique, Mandado Yago. (2007). Sistemas electrónicos digitales. España: Marcombo.

Palacios Enrique, Remiro Enrique, López Lucas. (2009). Microcontrolador PIC16F84, Madrid, España: Alfaomega.

Peatman B. John. (2003). Embedded Design with the PIC18F452 Microcontroller. Estados Unidos: Prentice Hall.

Gonsalve, G.S. (2005, mayo). Tecnología y Procesos de Fabricación de Microsistemas. Recuperado de http://bibing.us.es/proyectos/abreproy/20018/fichero/proyecto%252FPROYECTO.pdf

Litografia. (s.f.) Recuperado en noviembre 2012, de http://ebookbrowse.com/master-t1-2-resinas-clase-4-pdf-d166032398

Introducción a la fabricación de los Circuitos Integrados. (s.f.) Recuperado en noviembre del 2012, de http://www.fdi.ucm.es/profesor/mila45/TC_04_05/Teoria/Tema10.pdf

Vaglio Alessandro, Gronheid Roel. (2011). Microelectronic Engineering. Recuperado en mayo del 2012 de: www.elsevier.com/locate/mee

Camargo Leyner, Duran Sebastian. (mayo 2012). Electron Beam & Ion Beam. Recuperado en octubre 2012 de: http://www.gmun.unal.edu.co/~ijaramilloj/cursos/tecnicas/exposiciones/Grupo%2009%20Sebastian%20Dur%C3%A1n%20Leyner%20Camargo/Resumen%20i-e%20beam.pdf

Hiroyuki Chuma. (Diciembre 2004). Increasing Complexity and Limits of Organization in the Microlithography Industry: Implications for Japanese Science-based Industries. Recuperado en septiembre del 2012, de: http://www.rieti.go.jp/en/publications/summary/05030000.html

Micro resist Technology. Recuperado en agosto de 2012, de: http://www.microresist.de/home_en.htm

Micro Litography Services. Recuperado agosto de 2012, de: http://www.microlitho.co.uk/home.html

MLI, Micro Lithography INC. Recupera agosto de 2012, de: http://www.mliusa.com/index.htm

Hojas de datos de PIC18F452, LM7805, LM7810, L297, L298, LCD. Recuperado en 2012, de: www.alldatasheet.com/

Instrucciones de Programación del PIC18F452 de microchip. Recuperado en 2012 de: www.microchip.com

Apéndices

PIC18FXX2

28/40-pin High Performance, Enhanced FLASH Microcontrollers with 10-Bit A/D

High Performance RISC CPU:

- C compiler optimized architecture/instruction set
  - Source code compatible with the PIC16 and PIC17 instruction sets
- Linear program memory addressing to 32 Kbytes
- Linear data memory addressing to 1.5 Kbytes

Device	On-Chip Program Memory		On-Chip RAM (bytes)	Data EEPROM (bytes)
	FLASH (bytes)	# Single Word Instructions		
PIC18F242	16K	8192	768	256
PIC18F252	32K	16384	1536	256
PIC18F442	16K	8192	768	256
PIC18F452	32K	16384	1536	256

- Up to 10 MIPs operation:
  - DC - 40 MHz osc./clock input
  - 4 MHz - 10 MHz osc./clock input with PLL active
- 16-bit wide instructions, 8-bit wide data path
- Priority levels for interrupts
- 8 x 8 Single Cycle Hardware Multiplier

Peripheral Features:

- High current sink/source 25 mA/25 mA
- Three external interrupt pins
- Timer0 module: 8-bit/16-bit timer/counter with 8-bit programmable prescaler
- Timer1 module: 16-bit timer/counter
- Timer2 module: 8-bit timer/counter with 8-bit period register (time-base for PWM)
- Timer3 module: 16-bit timer/counter
- Secondary oscillator clock option - Timer1/Timer3
- Two Capture/Compare/PWM (CCP) modules. CCP pins that can be configured as:
  - Capture input: capture is 16-bit, max. resolution 6.25 ns (T_{CY}/16)
  - Compare is 16-bit, max. resolution 100 ns (T_{CY})
  - PWM output: PWM resolution is 1- to 10-bit, max. PWM freq. @: 8-bit resolution = 156 kHz, 10-bit resolution = 39 kHz
- Master Synchronous Serial Port (MSSP) module, Two modes of operation:
  - 3-wire SPI™ (supports all 4 SPI modes)
  - I^2C™ Master and Slave mode

Peripheral Features (Continued):

- Addressable USART module:
  - Supports RS-485 and RS-232
- Parallel Slave Port (PSP) module

Analog Features:

- Compatible 10-bit Analog-to-Digital Converter module (A/D) with:
  - Fast sampling rate
  - Conversion available during SLEEP
  - Linearity ≤ 1 LSb
- Programmable Low Voltage Detection (PLVD)
  - Supports interrupt on-Low Voltage Detection
- Programmable Brown-out Reset (BOR)

Special Microcontroller Features:

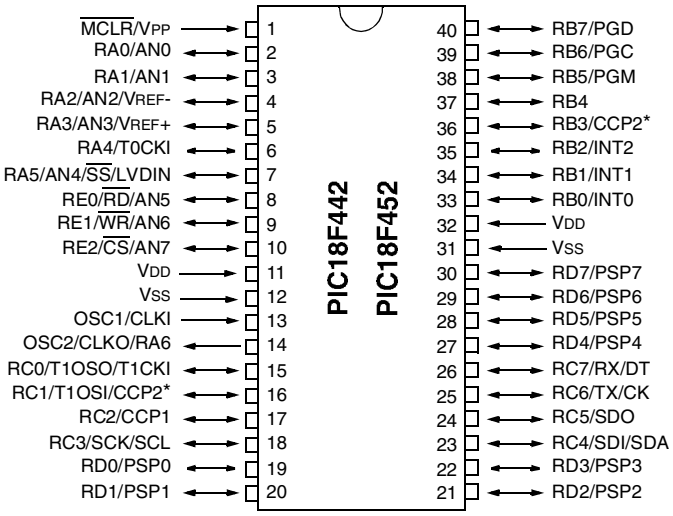
- 100,000 erase/write cycle Enhanced FLASH program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory
- FLASH/Data EEPROM Retention: > 40 years
- Self-reprogrammable under software control
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own On-Chip RC Oscillator for reliable operation
- Programmable code protection
- Power saving SLEEP mode
- Selectable oscillator options including:
  - 4X Phase Lock Loop (of primary oscillator)
  - Secondary Oscillator (32 kHz) clock input
- Single supply 5V In-Circuit Serial Programming™ (ICSP™) via two pins
- In-Circuit Debug (ICD) via two pins

CMOS Technology:

- Low power, high speed FLASH/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Industrial and Extended temperature ranges
- Low power consumption:
  - < 1.6 mA typical @ 5V, 4 MHz
  - 25 μA typical @ 3V, 32 kHz
  - < 0.2 μA typical standby current

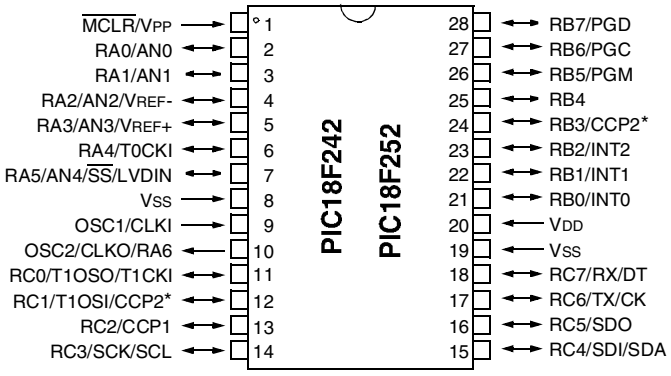
Pin Diagrams (Cont.'d)

DIP



Note: Pin compatible with 40-pin PIC16C7X devices.

DIP, SOIC



* RB3 is the alternate pin for the CCP2 pin multiplexing.

1.0 DEVICE OVERVIEW

This document contains device specific information for the following devices:

- PIC18F242
- PIC18F442
- PIC18F252
- PIC18F452

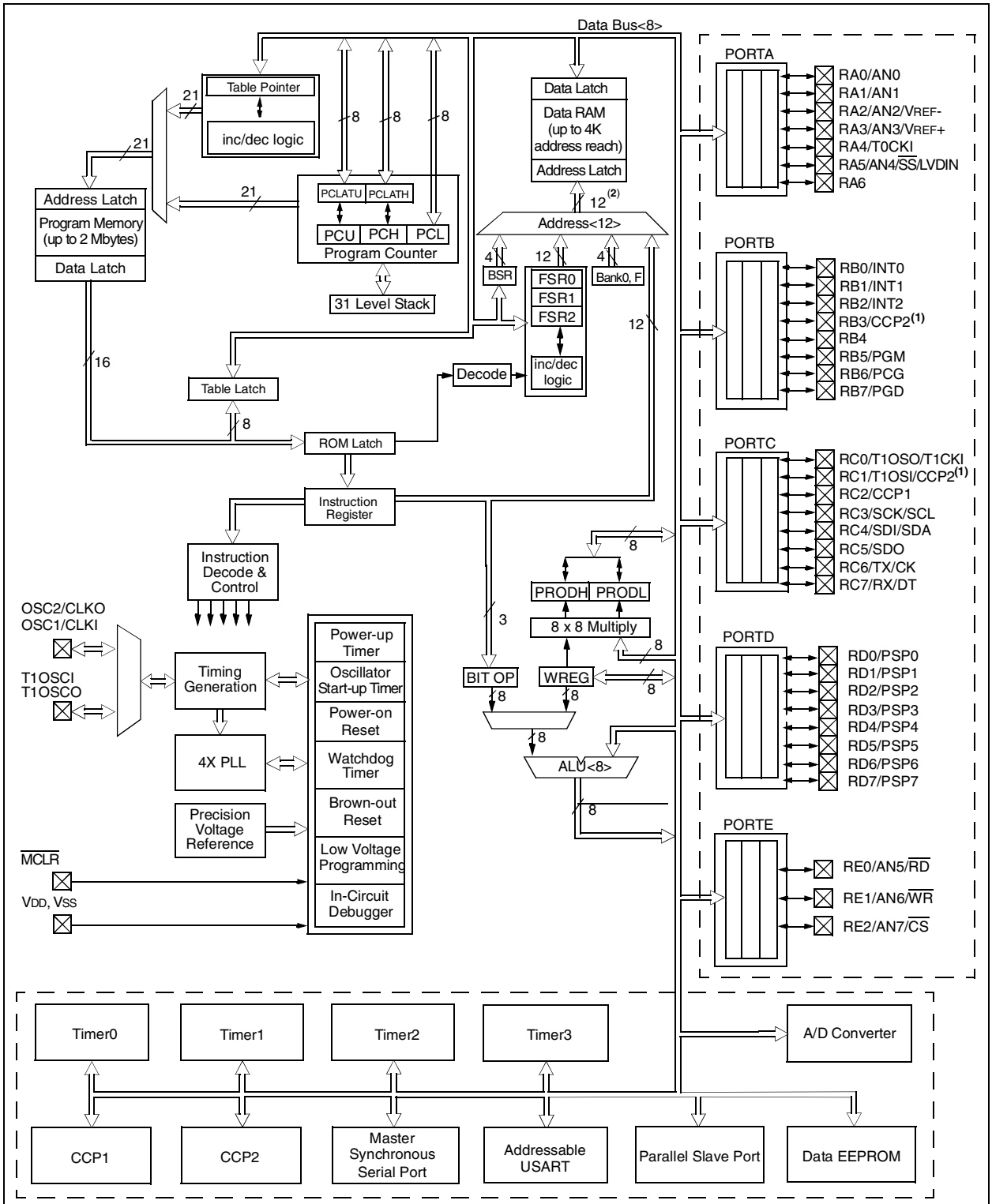
These devices come in 28-pin and 40/44-pin packages. The 28-pin devices do not have a Parallel Slave Port (PSP) implemented and the number of Analog-to-Digital (A/D) converter input channels is reduced to 5. An overview of features is shown in Table 1-1.

The following two figures are device block diagrams sorted by pin count: 28-pin for Figure 1-1 and 40/44-pin for Figure 1-2. The 28-pin and 40/44-pin pinouts are listed in Table 1-2 and Table 1-3, respectively.

TABLE 1-1: DEVICE FEATURES

Features	PIC18F242	PIC18F252	PIC18F442	PIC18F452
Operating Frequency	DC - 40 MHz	DC - 40 MHz	DC - 40 MHz	DC - 40 MHz
Program Memory (Bytes)	16K	32K	16K	32K
Program Memory (Instructions)	8192	16384	8192	16384
Data Memory (Bytes)	768	1536	768	1536
Data EEPROM Memory (Bytes)	256	256	256	256
Interrupt Sources	17	17	18	18
I/O Ports	Ports A, B, C	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C, D, E
Timers	4	4	4	4
Capture/Compare/PWM Modules	2	2	2	2
Serial Communications	MSSP, Addressable USART	MSSP, Addressable USART	MSSP, Addressable USART	MSSP, Addressable USART
Parallel Communications	—	—	PSP	PSP
10-bit Analog-to-Digital Module	5 input channels	5 input channels	8 input channels	8 input channels
RESETS (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST)
Programmable Low Voltage Detect	Yes	Yes	Yes	Yes
Programmable Brown-out Reset	Yes	Yes	Yes	Yes
Instruction Set	75 Instructions	75 Instructions	75 Instructions	75 Instructions
Packages	28-pin DIP 28-pin SOIC	28-pin DIP 28-pin SOIC	40-pin DIP 44-pin PLCC 44-pin TQFP	40-pin DIP 44-pin PLCC 44-pin TQFP

FIGURE 1-2: PIC18F4X2 BLOCK DIAGRAM



Note 1: Optional multiplexing of CCP2 input/output with RB3 is enabled by selection of configuration bit.

2: The high order bits of the Direct Address for the RAM are from the BSR register (except for the `MOVFF` instruction).

3: Many of the general purpose I/O pins are multiplexed with one or more peripheral module functions. The multiplexing combinations are device dependent.

4.0 MEMORY ORGANIZATION

There are three memory blocks in Enhanced MCU devices. These memory blocks are:

- Program Memory
- Data RAM
- Data EEPROM

Data and program memory use separate busses, which allows for concurrent access of these blocks.

Additional detailed information for FLASH program memory and Data EEPROM is provided in Section 5.0 and Section 6.0, respectively.

4.1 Program Memory Organization

A 21-bit program counter is capable of addressing the 2-Mbyte program memory space. Accessing a location between the physically implemented memory and the 2-Mbyte address will cause a read of all '0's (a `NOP` instruction).

The PIC18F252 and PIC18F452 each have 32 Kbytes of FLASH memory, while the PIC18F242 and PIC18F442 have 16 Kbytes of FLASH. This means that PIC18FX52 devices can store up to 16K of single word instructions, and PIC18FX42 devices can store up to 8K of single word instructions.

The RESET vector address is at 0000h and the interrupt vector addresses are at 0008h and 0018h.

Figure 4-1 shows the Program Memory Map for PIC18F242/442 devices and Figure 4-2 shows the Program Memory Map for PIC18F252/452 devices.

4.14 RCON Register

The Reset Control (RCON) register contains flag bits that allow differentiation between the sources of a device RESET. These flags include the $\overline{TO}$, $\overline{PD}$, $\overline{POR}$, $\overline{BOR}$ and $\overline{RI}$ bits. This register is readable and writable.

Note 1: If the BOREN configuration bit is set (Brown-out Reset enabled), the $\overline{BOR}$ bit is '1' on a Power-on Reset. After a Brown-out Reset has occurred, the BOR bit will be cleared, and must be set by firmware to indicate the occurrence of the next Brown-out Reset.

2: It is recommended that the $\overline{POR}$ bit be set after a Power-on Reset has been detected, so that subsequent Power-on Resets may be detected.

REGISTER 4-3: RCON REGISTER

R/W-0	U-0	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

bit 7 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (16CXXX Compatibility mode)

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
1 = The RESET instruction was not executed
0 = The RESET instruction was executed causing a device RESET (must be set in software after a Brown-out Reset occurs)

bit 3 **$\overline{TO}$:** Watchdog Time-out Flag bit
1 = After power-up, CLRWDT instruction, or SLEEP instruction
0 = A WDT time-out occurred

bit 2 **$\overline{PD}$:** Power-down Detection Flag bit
1 = After power-up or by the CLRWDT instruction
0 = By execution of the SLEEP instruction

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
1 = A Power-on Reset has not occurred
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
1 = A Brown-out Reset has not occurred
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

8.0 INTERRUPTS

The PIC18FXX2 devices have multiple interrupt sources and an interrupt priority feature that allows each interrupt source to be assigned a high priority level or a low priority level. The high priority interrupt vector is at 000008h and the low priority interrupt vector is at 000018h. High priority interrupt events will override any low priority interrupts that may be in progress.

There are ten registers which are used to control interrupt operation. These registers are:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2
- PIE1, PIE2
- IPR1, IPR2

It is recommended that the Microchip header files supplied with MPLAB® IDE be used for the symbolic bit names in these registers. This allows the assembler/compiler to automatically take care of the placement of these bits within the specified register.

Each interrupt source, except INT0, has three bits to control its operation. The functions of these bits are:

- Flag bit to indicate that an interrupt event occurred
- Enable bit that allows program execution to branch to the interrupt vector address when the flag bit is set
- Priority bit to select high priority or low priority

The interrupt priority feature is enabled by setting the IPEN bit (RCON<7>). When interrupt priority is enabled, there are two bits which enable interrupts globally. Setting the GIEH bit (INTCON<7>) enables all interrupts that have the priority bit set. Setting the GIEL bit (INTCON<6>) enables all interrupts that have the priority bit cleared. When the interrupt flag, enable bit and appropriate global interrupt enable bit are set, the interrupt will vector immediately to address 000008h or 000018h, depending on the priority level. Individual interrupts can be disabled through their corresponding enable bits.

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PICmicro® mid-range devices. In Compatibility mode, the interrupt priority bits for each source have no effect. INTCON<6> is the PEIE bit, which enables/disables all peripheral interrupt sources. INTCON<7> is the GIE bit, which enables/disables all interrupt sources. All interrupts branch to address 000008h in Compatibility mode.

When an interrupt is responded to, the Global Interrupt Enable bit is cleared to disable further interrupts. If the IPEN bit is cleared, this is the GIE bit. If interrupt priority levels are used, this will be either the GIEH or GIEL bit. High priority interrupt sources can interrupt a low priority interrupt.

The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (000008h or 000018h). Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

The “return from interrupt” instruction, `RETFIE`, exits the interrupt routine and sets the GIE bit (GIEH or GIEL if priority levels are used), which re-enables interrupts.

For external interrupt events, such as the INT pins or the PORTB input change interrupt, the interrupt latency will be three to four instruction cycles. The exact latency is the same for one or two-cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding enable bit or the GIE bit.

Note:	Do not use the `MOVFF` instruction to modify any of the Interrupt control registers while **any** interrupt is enabled. Doing so may cause erratic microcontroller behavior.

8.1 INTCON Registers

The INTCON Registers are readable and writable registers, which contain various enable, priority and flag bits.

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

REGISTER 8-1: INTCON REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
bit 7							bit 0

bit 7 **GIE/GIEH:** Global Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts
When IPEN = 1:
1 = Enables all high priority interrupts
0 = Disables all interrupts

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts
When IPEN = 1:
1 = Enables all low priority peripheral interrupts
0 = Disables all low priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 overflow interrupt
0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit
1 = Enables the INT0 external interrupt
0 = Disables the INT0 external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** INT0 External Interrupt Flag bit
1 = The INT0 external interrupt occurred (must be cleared in software)
0 = The INT0 external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

Note: A mismatch condition will continue to set this bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 8-2: INTCON2 REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1	U-0	R/W-1
RBPU	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$**: PORTB Pull-up Enable bit
1 = All PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG0**:External Interrupt0 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 5 **INTEDG1**: External Interrupt1 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 4 **INTEDG2**: External Interrupt2 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 3 **Unimplemented:** Read as '0'

bit 2 **TMR0IP**: TMR0 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **Unimplemented:** Read as '0'

bit 0 **RBIP**: RB Port Change Interrupt Priority bit
1 = High priority
0 = Low priority

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
- n = Value at POR	’1’ = Bit is set	’0’ = Bit is cleared	x = Bit is unknown

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

8.6 INT0 Interrupt

External interrupts on the RB0/INT0, RB1/INT1 and RB2/INT2 pins are edge triggered: either rising, if the corresponding INTEDGx bit is set in the INTCON2 register, or falling, if the INTEDGx bit is clear. When a valid edge appears on the RBx/INTx pin, the corresponding flag bit INTxF is set. This interrupt can be disabled by clearing the corresponding enable bit INTxE. Flag bit INTxF must be cleared in software in the Interrupt Service Routine before re-enabling the interrupt. All external interrupts (INT0, INT1 and INT2) can wake-up the processor from SLEEP, if bit INTxE was set prior to going into SLEEP. If the global interrupt enable bit GIE is set, the processor will branch to the interrupt vector following wake-up.

Interrupt priority for INT1 and INT2 is determined by the value contained in the interrupt priority bits, INT1IP (INTCON3<6>) and INT2IP (INTCON3<7>). There is no priority bit associated with INT0. It is always a high priority interrupt source.

8.7 TMR0 Interrupt

In 8-bit mode (which is the default), an overflow (FFh $\rightarrow$ 00h) in the TMR0 register will set flag bit TMR0IF. In 16-bit mode, an overflow (FFFFh $\rightarrow$ 0000h) in the TMR0H:TMR0L registers will set flag bit TMR0IF. The interrupt can be enabled/disabled by setting/clearing enable bit T0IE (INTCON<5>). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit TMR0IP (INTCON2<2>). See Section 10.0 for further details on the Timer0 module.

8.8 PORTB Interrupt-on-Change

An input change on PORTB<7:4> sets flag bit RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit, RBIE (INTCON<3>). Interrupt priority for PORTB interrupt-on-change is determined by the value contained in the interrupt priority bit, RBIP (INTCON2<0>).

8.9 Context Saving During Interrupts

During an interrupt, the return PC value is saved on the stack. Additionally, the WREG, STATUS and BSR registers are saved on the fast return stack. If a fast return from interrupt is not used (See Section 4.3), the user may need to save the WREG, STATUS and BSR registers in software. Depending on the user's application, other registers may also need to be saved. Equation 8-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 8-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

```
MOVWF   W_TEMP                          ; W_TEMP is in virtual bank
MOVFF   STATUS, STATUS_TEMP             ; STATUS_TEMP located anywhere
MOVFF   BSR,    BSR_TEMP                ; BSR located anywhere
;
; USER ISR CODE
;
MOVFF   BSR_TEMP,   BSR                 ; Restore BSR
MOVF    W_TEMP,     W                   ; Restore WREG
MOVFF   STATUS_TEMP,STATUS              ; Restore STATUS
```

10.0 TIMER0 MODULE

The Timer0 module has the following features:

- Software selectable as an 8-bit or 16-bit timer/counter
- Readable and writable
- Dedicated 8-bit software programmable prescaler
- Clock source selectable to be external or internal
- Interrupt-on-overflow from FFh to 00h in 8-bit mode and FFFFh to 0000h in 16-bit mode
- Edge select for external clock

Figure 10-1 shows a simplified block diagram of the Timer0 module in 8-bit mode and Figure 10-2 shows a simplified block diagram of the Timer0 module in 16-bit mode.

The T0CON register (Register 10-1) is a readable and writable register that controls all the aspects of Timer0, including the prescale selection.

REGISTER 10-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

bit 7 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

bit 6 **T08BIT**: Timer0 8-bit/16-bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

bit 5 **T0CS**: Timer0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE**: Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Timer0 Prescaler Assignment bit
1 = TImer0 prescaler is NOT assigned. Timer0 clock input bypasses prescaler.
0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

bit 2-0 **T0PS2:T0PS0**: Timer0 Prescaler Select bits
111 = 1:256 prescale value
110 = 1:128 prescale value
101 = 1:64 prescale value
100 = 1:32 prescale value
011 = 1:16 prescale value
010 = 1:8 prescale value
001 = 1:4 prescale value
000 = 1:2 prescale value

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

14.1 CCP1 Module

Capture/Compare/PWM Register 1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. All are readable and writable.

TABLE 14-1: CCP MODE - TIMER RESOURCE

CCP Mode	Timer Resource
Capture	Timer1 or Timer3
Compare	Timer1 or Timer3
PWM	Timer2

14.2 CCP2 Module

Capture/Compare/PWM Register2 (CCPR2) is comprised of two 8-bit registers: CCPR2L (low byte) and CCPR2H (high byte). The CCP2CON register controls the operation of CCP2. All are readable and writable.

TABLE 14-2: INTERACTION OF TWO CCP MODULES

CCPx Mode	CCPy Mode	Interaction
Capture	Capture	TMR1 or TMR3 time-base. Time-base can be different for each CCP.
Capture	Compare	The compare could be configured for the special event trigger, which clears either TMR1 or TMR3 depending upon which time-base is used.
Compare	Compare	The compare(s) could be configured for the special event trigger, which clears TMR1 or TMR3 depending upon which time-base is used.
PWM	PWM	The PWMs will have the same frequency and update rate (TMR2 interrupt).
PWM	Capture	None
PWM	Compare	None

14.5 PWM Mode

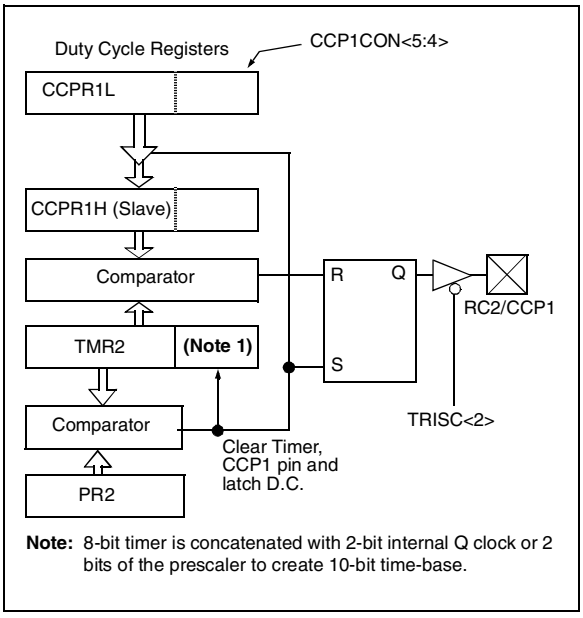
In Pulse Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISC<2> bit must be cleared to make the CCP1 pin an output.

Note:	Clearing the CCP1CON register will force the CCP1 PWM output latch to the default low level. This is not the PORTC I/O data latch.

Figure 14-3 shows a simplified block diagram of the CCP module in PWM mode.

For a step-by-step procedure on how to set up the CCP module for PWM operation, see Section 14.5.3.

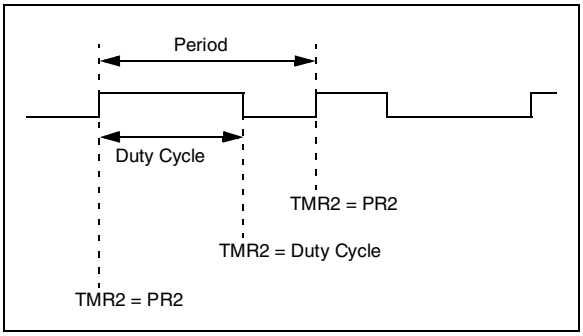
FIGURE 14-3: SIMPLIFIED PWM BLOCK DIAGRAM



Note: 8-bit timer is concatenated with 2-bit internal Q clock or 2 bits of the prescaler to create 10-bit time-base.

A PWM output (Figure 14-4) has a time-base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 14-4: PWM OUTPUT



14.5.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

$$\text{PWM period} = (PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 prescale value})$$

PWM frequency is defined as 1 / [PWM period].

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (exception: if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is latched from CCPR1L into CCPR1H

Note:	The Timer2 postscaler (see Section 12.0) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

14.5.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSbs and the CCP1CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

$$\text{PWM duty cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 prescale value})$$

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR1H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read only register.

The CCPR1H register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This double buffering is essential for glitchless PWM operation.

When the CCPR1H and 2-bit latch match TMR2 concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the equation:

$$\text{PWM Resolution (max)} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{bits}$$

Note:	If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.

TABLE 20-2: PIC18FXXX INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb			LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d, a	Add WREG and f	1	0010	01da0	ffff	ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010	0da	ffff	ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	AND WREG with f	1	0001	01da	ffff	ffff	Z, N	1,2
CLRF	f, a	Clear f	1	0110	101a	ffff	ffff	Z	2
COMF	f, d, a	Complement f	1	0001	11da	ffff	ffff	Z, N	1, 2
CPFSEQ	f, a	Compare f with WREG, skip =	1 (2 or 3)	0110	001a	ffff	ffff	None	4
CPFSGT	f, a	Compare f with WREG, skip >	1 (2 or 3)	0110	010a	ffff	ffff	None	4
CPFSLT	f, a	Compare f with WREG, skip <	1 (2 or 3)	0110	000a	ffff	ffff	None	1, 2
DECF	f, d, a	Decrement f	1	0000	01da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010	11da	ffff	ffff	None	1, 2, 3, 4
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100	11da	ffff	ffff	None	1, 2
INCF	f, d, a	Increment f	1	0010	10da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011	11da	ffff	ffff	None	4
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100	10da	ffff	ffff	None	1, 2
IORWF	f, d, a	Inclusive OR WREG with f	1	0001	00da	ffff	ffff	Z, N	1, 2
MOVF	f, d, a	Move f	1	0101	00da	ffff	ffff	Z, N	1
MOVFF	f_s, f_d	Move f_s (source) to 1st word f_d (destination) 2nd word	2	1100 1111	ffff ffff	ffff ffff	ffff ffff	None	
MOVWF	f, a	Move WREG to f	1	0110	111a	ffff	ffff	None	
MULWF	f, a	Multiply WREG with f	1	0000	001a	ffff	ffff	None	
NEGF	f, a	Negate f	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	1, 2
RLCF	f, d, a	Rotate Left f through Carry	1	0011	01da	ffff	ffff	C, Z, N	
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100	01da	ffff	ffff	Z, N	1, 2
RRCF	f, d, a	Rotate Right f through Carry	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	Set f	1	0110	100a	ffff	ffff	None	
SUBFWB	f, d, a	Subtract f from WREG with borrow	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
SUBWF	f, d, a	Subtract WREG from f	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	
SUBWFB	f, d, a	Subtract WREG from f with borrow	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	1, 2
SWAPF	f, d, a	Swap nibbles in f	1	0011	10da	ffff	ffff	None	4
TSTFSZ	f, a	Test f, skip if 0	1 (2 or 3)	0110	011a	ffff	ffff	None	1, 2
XORWF	f, d, a	Exclusive OR WREG with f	1	0001	10da	ffff	ffff	Z, N	
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b, a	Bit Clear f	1	1001	bbba	ffff	ffff	None	1, 2
BSF	f, b, a	Bit Set f	1	1000	bbba	ffff	ffff	None	1, 2
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011	bbba	ffff	ffff	None	3, 4
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010	bbba	ffff	ffff	None	3, 4
BTG	f, d, a	Bit Toggle f	1	0111	bbba	ffff	ffff	None	1, 2

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP`, unless the first word of the instruction retrieves the information embedded in these 16-bits. This ensures that all program memory locations have a valid instruction.

5: If the Table Write starts the write cycle to internal memory, the write will continue until terminated.

TABLE 20-2: PIC18FXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb			LSb	Status Affected	Notes
LITERAL OPERATIONS									
ADDLW	k	Add literal and WREG	1	0000	1111	kkkk	kkkk	C, DC, Z, OV, N	
ANDLW	k	AND literal with WREG	1	0000	1011	kkkk	kkkk	Z, N	
IORLW	k	Inclusive OR literal with WREG	1	0000	1001	kkkk	kkkk	Z, N	
LFSR	f, k	Move literal (12-bit) 2nd word to FSRx 1st word	2	1110	1110	00ff	kkkk	None	
				1111	0000	kkkk	kkkk		
MOVLB	k	Move literal to BSR<3:0>	1	0000	0001	0000	kkkk	None	
MOVLW	k	Move literal to WREG	1	0000	1110	kkkk	kkkk	None	
MULLW	k	Multiply literal with WREG	1	0000	1101	kkkk	kkkk	None	
RETLW	k	Return with literal in WREG	2	0000	1100	kkkk	kkkk	None	
SUBLW	k	Subtract WREG from literal	1	0000	1000	kkkk	kkkk	C, DC, Z, OV, N	
XORLW	k	Exclusive OR literal with WREG	1	0000	1010	kkkk	kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS									
TBLRD*		Table Read	2	0000	0000	0000	1000	None	
TBLRD*+		Table Read with post-increment		0000	0000	0000	1001	None	
TBLRD*-		Table Read with post-decrement		0000	0000	0000	1010	None	
TBLRD+*		Table Read with pre-increment		0000	0000	0000	1011	None	
TBLWT*		Table Write	2 (5)	0000	0000	0000	1100	None	
TBLWT*+		Table Write with post-increment		0000	0000	0000	1101	None	
TBLWT*-		Table Write with post-decrement		0000	0000	0000	1110	None	
TBLWT+*		Table Write with pre-increment		0000	0000	0000	1111	None	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP`, unless the first word of the instruction retrieves the information embedded in these 16-bits. This ensures that all program memory locations have a valid instruction.

5: If the Table Write starts the write cycle to internal memory, the write will continue until terminated.

2.Precautions in use of LCD Modules

(1)Avoid applying excessive shocks to the module or making any alterations or modifications to it.
(2)Don't make extra holes on the printed circuit board, modify its shape or change the components of LCD module.
(3)Don't disassemble the LCM.
(4)Don't operate it above the absolute maximum rating.
(5)Don't drop, bend or twist LCM.
(6)Soldering: only to the I/O terminals.
(7)Storage: please storage in anti-static electricity container and clean environment.
(8). Winstar have the right to change the passive components
(9). Winstar have the right to change the PCB Rev.

3.General Specification

Item	Dimension	Unit
Number of Characters	16 characters x 2 Lines	—
Module dimension	80.0 x 36.0 x 13.2(MAX)	mm
View area	66.0 x 16.0	mm
Active area	56.20 x 11.5	mm
Dot size	0.55 x 0.65	mm
Dot pitch	0.60 x 0.70	mm
Character size	2.95 x 5.55	mm
Character pitch	3.55 x 5.95	mm
LCD type	STN Positive, Yellow Green Transflective	
Duty	1/16	
View direction	6 o'clock	
Backlight Type	LED Yellow Green	

4.Absolute Maximum Ratings

Item	Symbol	Min	Typ	Max	Unit
Operating Temperature	T_{OP}	-20	—	+70	℃
Storage Temperature	T_{ST}	-30	—	+80	℃
Input Voltage	V_I	V_{SS}	—	V_{DD}	V
Supply Voltage For Logic	V_{DD}-V_{SS}	-0.3	—	7	V
Supply Voltage For LCD	V_{DD}-V_0	-0.3	—	13	V

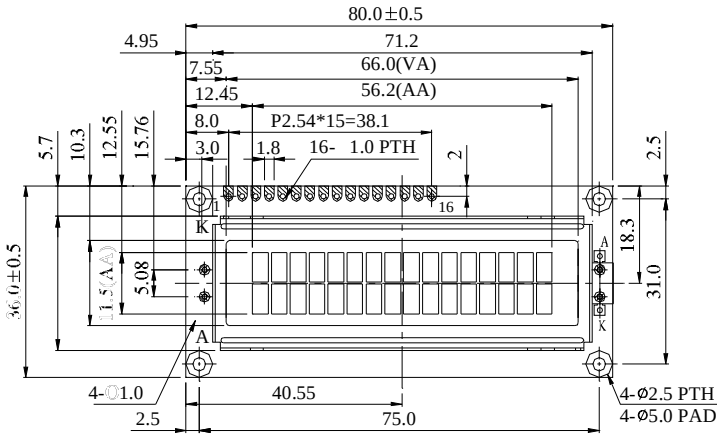
5.Electrical Characteristics

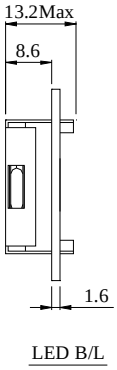
Item	Symbol	Condition	Min	Typ	Max	Unit
Supply Voltage For Logic	V_{DD}-V_{SS}	—	4.5	5.0	5.5	V
Supply Voltage For LCD	V_{DD}-V_0	Ta=-20	—		5.2	V
		Ta=25℃	—	3.7		V
		Ta=70℃	3.2	—	—	V
Input High Volt.	V_{IH}		0.7 V_{DD}		V_{DD}	V
Input Low Volt.	V_{IL}		V_{SS}	—	0.6	V
Output High Volt.	V_{OH}		3.9	—	—	V
Output Low Volt.	V_{OL}		—	—	0.4	V
Supply Current	I_{DD}	V_{DD}=5.0V	1.0	1.2	1.5	mA

7.Interface Pin Function

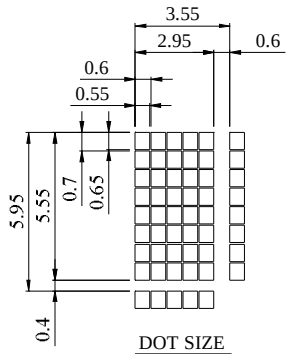
Pin No.	Symbol	Level	Description
1	V_{SS}	0V	Ground
2	V_{DD}	5.0V	Supply Voltage for logic
3	VO	(Variable)	Operating voltage for LCD
4	RS	H/L	H: DATA, L: Instruction code
5	R/W	H/L	H: Read(MPU Module) L: Write(MPU⇒Module)
6	E	H,H→L	Chip enable signal
7	DB0	H/L	Data bus line
8	DB1	H/L	Data bus line
9	DB2	H/L	Data bus line
10	DB3	H/L	Data bus line
11	DB4	H/L	Data bus line
12	DB5	H/L	Data bus line
13	DB6	H/L	Data bus line
14	DB7	H/L	Data bus line
15	A	—	LED +
16	K		LED

8.Contour Drawing &Block Diagram

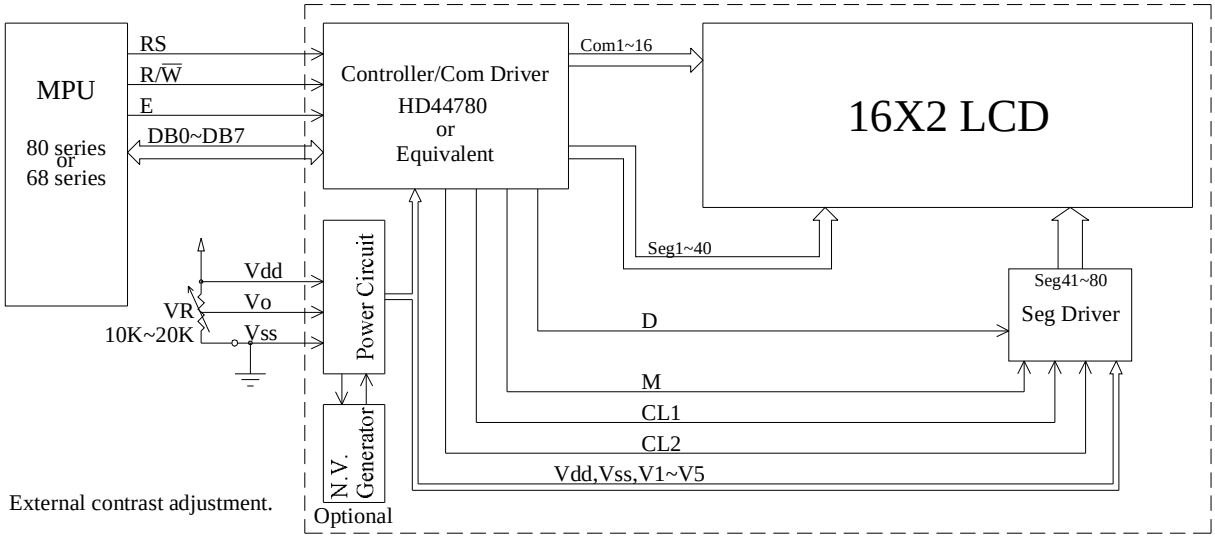




PIN NO.	SYMBOL
1	Vss
2	Vdd
3	Vo
4	RS
5	R/$\overline{W}$
6	E
7	DB0
8	DB1
9	DB2
10	DB3
11	DB4
12	DB5
13	DB6
14	DB7
15	A
16	K



The non-specified tolerance of dimension is ±0.3mm.



Character located	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DDRAM address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
DDRAM address	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F

10.Character Generator ROM Pattern

Table.2

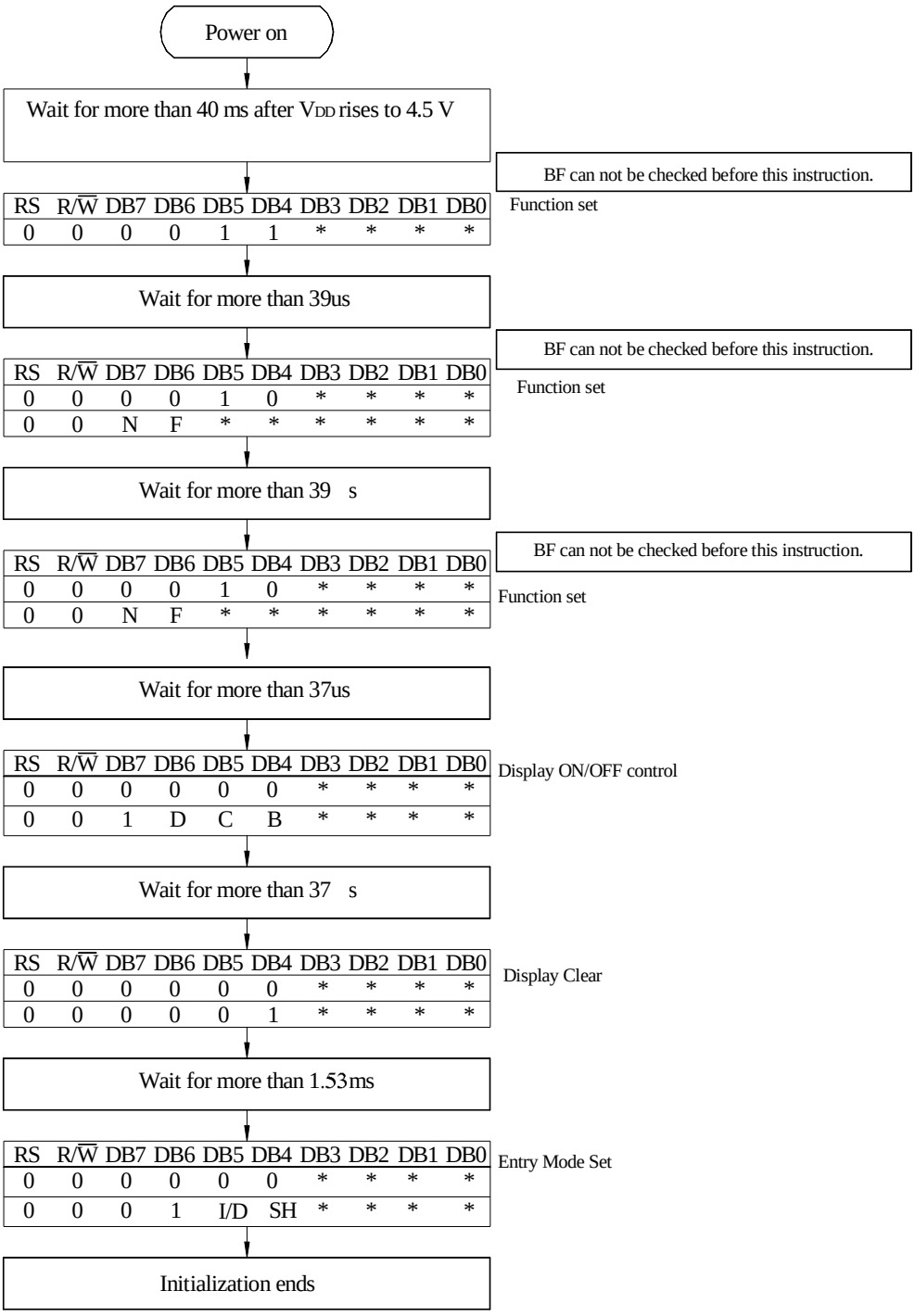
Upper 4 bit / Lower 4 bit	LLLL	LLLH	LLHL	LLHH	LHLL	LHLH	LHHL	LHHH	HLLL	HLLH	HLHL	HLHH	HHLL	HHLH	HHHL	HHHH
LLLL	CG RAM (1)																
LLLH	CG RAM (2)															
LLHL	CG RAM (3)															
LLHH	CG RAM (4)																
LHLL	CG RAM (5)																
LHLH	CG RAM (6)															
LHHL	CG RAM (7)															
LHHH	CG RAM (8)															
HLLL	CG RAM (1)															
HLLH	CG RAM (2)															
HLHL	CG RAM (3)															
HLHH	CG RAM (4)															
HHLL	CG RAM (5)															
HHLH	CG RAM (6)															
HHHL	CG RAM (7)															
HHHH	CG RAM (8)																

11.Instruction Table

Instruction	Instruction Code										Description	Execution time (fosc=270Khz)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write “00H” to DDRAM and set DDRAM address to “00H” from AC	1.53ms
Return Home	0	0	0	0	0	0	0	0	1	—	Set DDRAM address to “00H” from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.53ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	SH	Assign cursor moving direction and enable the shift of entire display.	39 μs
Display ON/OFF Control	0	0	0	0	0	0	1	D	C	B	Set display (D), cursor (C), and blinking of cursor (B) on/off control bit.	39 μs
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	—	—	Set cursor moving and display shift control bit, and the direction, without changing of DDRAM data.	39 μs
Function Set	0	0	0	0	1	DL	N	F	—	—	Set interface data length (DL:8-bit/4-bit), numbers of display line (N:2-line/1-line)and, display font type (F:5×11 dots/5×8 dots)	39 μs
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter.	39 μs
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter.	39 μs
Read Busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	0 μs
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	43 μs
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	43 μs

＊ ”—” don’t care

13.Initializing of LCM

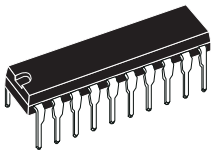


4-Bit Ineterface

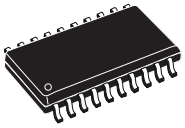
L297

STEPPER MOTOR CONTROLLERS

- NORMAL/WAVE DRIVE
- HALF/FULL STEP MODES
- CLOCKWISE/ANTICLOCKWISE DIRECTION
- SWITCHMODE LOAD CURRENT REGULATION
- PROGRAMMABLE LOAD CURRENT
- FEW EXTERNAL COMPONENTS
- RESET INPUT & HOME OUTPUT
- ENABLE INPUT



DIP20



SO20

ORDERING NUMBERS : L297/1 (DIP20)
L297D (SO20)

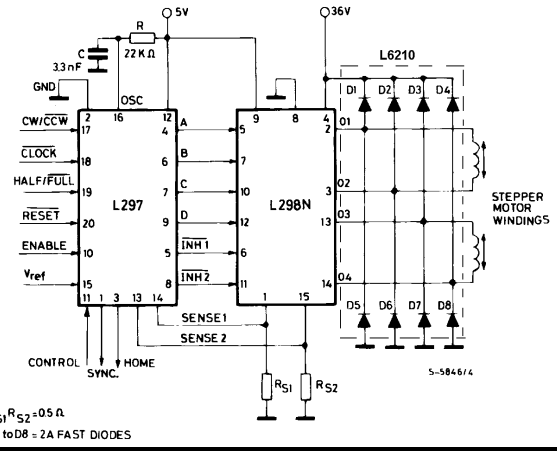
DESCRIPTION

The L297 Stepper Motor Controller IC generates four phase drive signals for two phase bipolar and four phase unipolar step motors in microcomputer-controlled applications. The motor can be driven in half step, normal and wawe drive modes and on-chip PWM chopper circuits permit switch-mode control of the current in the windings. A feature of this device is that it requires only clock, direction and mode input signals. Since the phase are generated internally the burden on the microprocessor, and the programmer, is greatly reduced. Mounted in DIP20 and SO20 packages, the L297 can be used with monolithic bridge drives such as the L298N or L293E, or with discrete transistors and darlingtons.

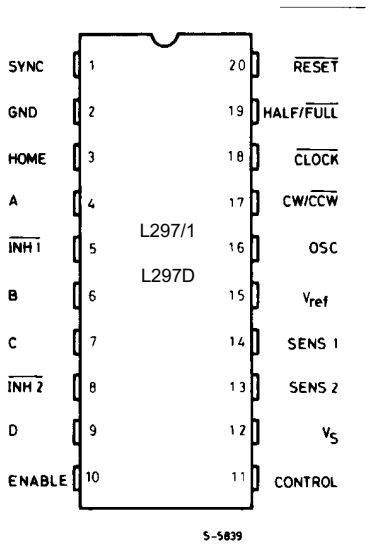
ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_s	Supply voltage	10	V
V_i	Input signals	7	V
P_{tot}	Total power dissipation (T_{amb} = 70°C)	1	W
T_{stg}, T_j	Storage and junction temperature	-40 to + 150	°C

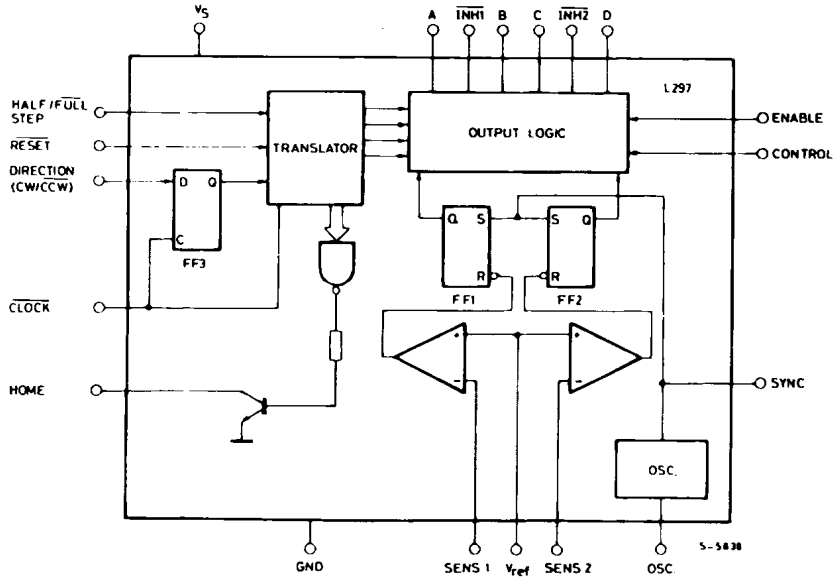
TWO PHASE BIPOLAR STEPPER MOTOR CONTROL CIRCUIT



PIN CONNECTION (Top view)



BLOCK DIAGRAM (L297/1 - L297D)



PIN FUNCTIONS - L297/1 - L297D

N°	NAME	FUNCTION
1	SYNC	Output of the on-chip chopper oscillator. The SYNC connections The SYNC connections of all L297s to be synchronized are connected together and the oscillator components are omitted on all but one. If an external clock source is used it is injected at this terminal.
2	GND	Ground connection.
3	HOME	Open collector output that indicates when the L297 is in its initial state (ABCD = 0101). The transistor is open when this signal is active.
4	A	Motor phase A drive signal for power stage.
5	$\overline{INH1}$	Active low inhibit control for driver stage of A and B phases. When a bipolar bridge is used this signal can be used to ensure fast decay of load current when a winding is de-energized. Also used by chopper to regulate load current if CONTROL input is low.
6	B	Motor phase B drive signal for power stage.
7	C	Motor phase C drive signal for power stage.
8	$\overline{INH2}$	Active low inhibit control for drive stages of C and D phases. Same functions as INH1.
9	D	Motor phase D drive signal for power stage.
10	ENABLE	Chip enable input. When low (inactive) INH1, INH2, A, B, C and D are brought low.
11	CONTROL	Control input that defines action of chopper. When low chopper acts on INH1 and INH2; when high chopper acts on phase lines ABCD.
12	V_s	5V supply input.
13	$SENS_2$	Input for load current sense voltage from power stages of phases C and D.
14	$SENS_1$	Input for load current sense voltage from power stages of phases A and B.
15	V_{ref}	Reference voltage for chopper circuit. A voltage applied to this pin determines the peak load current.
16	OSC	An RC network (R to V_{CC}, C to ground) connected to this terminal determines the chopper rate. This terminal is connected to ground on all but one device in synchronized multi - L297 configurations. f $\cong$ 1/0.69 RC
17	CW/$\overline{CCW}$	Clockwise/counterclockwise direction control input. Physical direction of motor rotation also depends on connection of windings. Synchronized internally therefore direction can be changed at any time.
18	$\overline{CLOCK}$	Step clock. An active low pulse on this input advances the motor one increment. The step occurs on the rising edge of this signal.

PIN FUNCTIONS - L297/1 - L297D (continued)

N°	NAME	FUNCTION
19	HALF/$\overline{\text{FULL}}$	Half/full step select input. When high selects half step operation, when low selects full step operation. One-phase-on full step mode is obtained by selecting FULL when the L297's translator is at an even-numbered state. Two-phase-on full step mode is set by selecting FULL when the translator is at an odd numbered position. (The home position is designate state 1).
20	$\overline{\text{RESET}}$	Reset input. An active low pulse on this input restores the translator to the home position (state 1, ABCD = 0101).

THERMAL DATA

Symbol	Parameter		DIP20	SO20	Unit
$R_{th\text{-}j\text{-}amb}$	Thermal resistance junction-ambient	max	80	100	°C/W

CIRCUIT OPERATION

The L297 is intended for use with a dual bridge driver, quad darlington array or discrete power devices in step motor driving applications. It receives step clock, direction and mode signals from the systems controller (usually a microcomputer chip) and generates control signals for the power stage.

The principal functions are a translator, which generates the motor phase sequences, and a dual PWM chopper circuit which regulates the current in the motor windings. The translator generates three different sequences, selected by the HALF/FULL input. These are normal (two phases energised), wave drive (one phase energised) and half-step (alternately one phase energised/two phases energised). Two inhibit signals are also generated by the L297 in half step and wave drive modes. These signals, which connect directly to the L298's enable inputs, are intended to speed current decay when a winding is de-energised. When the L297 is used to drive a unipolar motor the chopper acts on these lines.

An input called CONTROL determines whether the chopper will act on the phase lines ABCD or the inhibit lines INH1 and INH2. When the phase lines are chopped the non-active phase line of each pair (AB or CD) is activated (rather than interrupting the line then active). In L297 + L298 configurations this technique reduces dissipation in the load current sense resistors.

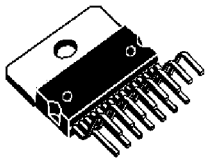
A common on-chip oscillator drives the dual chopper. It supplies pulses at the chopper rate which set the two flip-flops FF1 and FF2. When the current in a winding reaches the programmed peak value the voltage across the sense resistor (connected to one of the sense inputs $SENS_1$ or $SENS_2$) equals V_{ref} and the corresponding comparator resets its flip flop, interrupting the drive current until the next oscillator pulse arrives. The peak current for both windings is programmed by a voltage divider on the V_{ref} input.

Ground noise problems in multiple configurations can be avoided by synchronising the chopper oscillators. This is done by connecting all the SYNC pins together, mounting the oscillator RC network on one device only and grounding the OSC pin on all other devices.

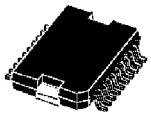
L298

DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)



Multiwatt15



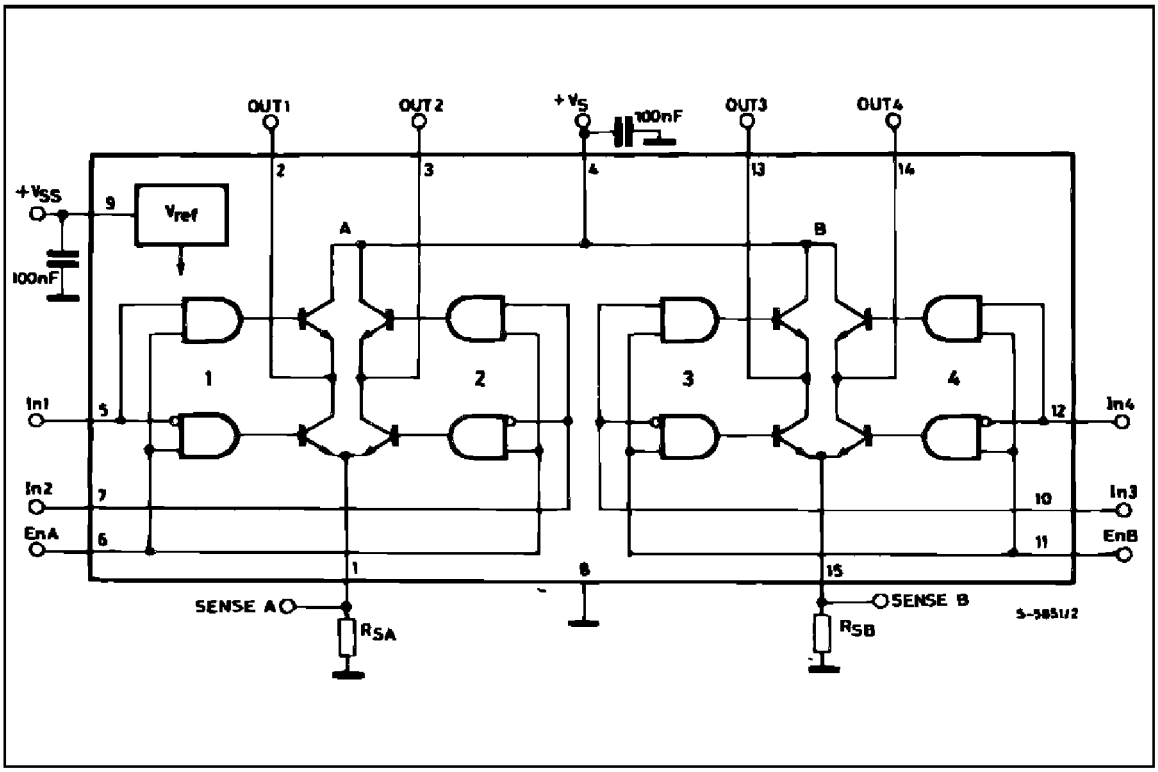
PowerSO20

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

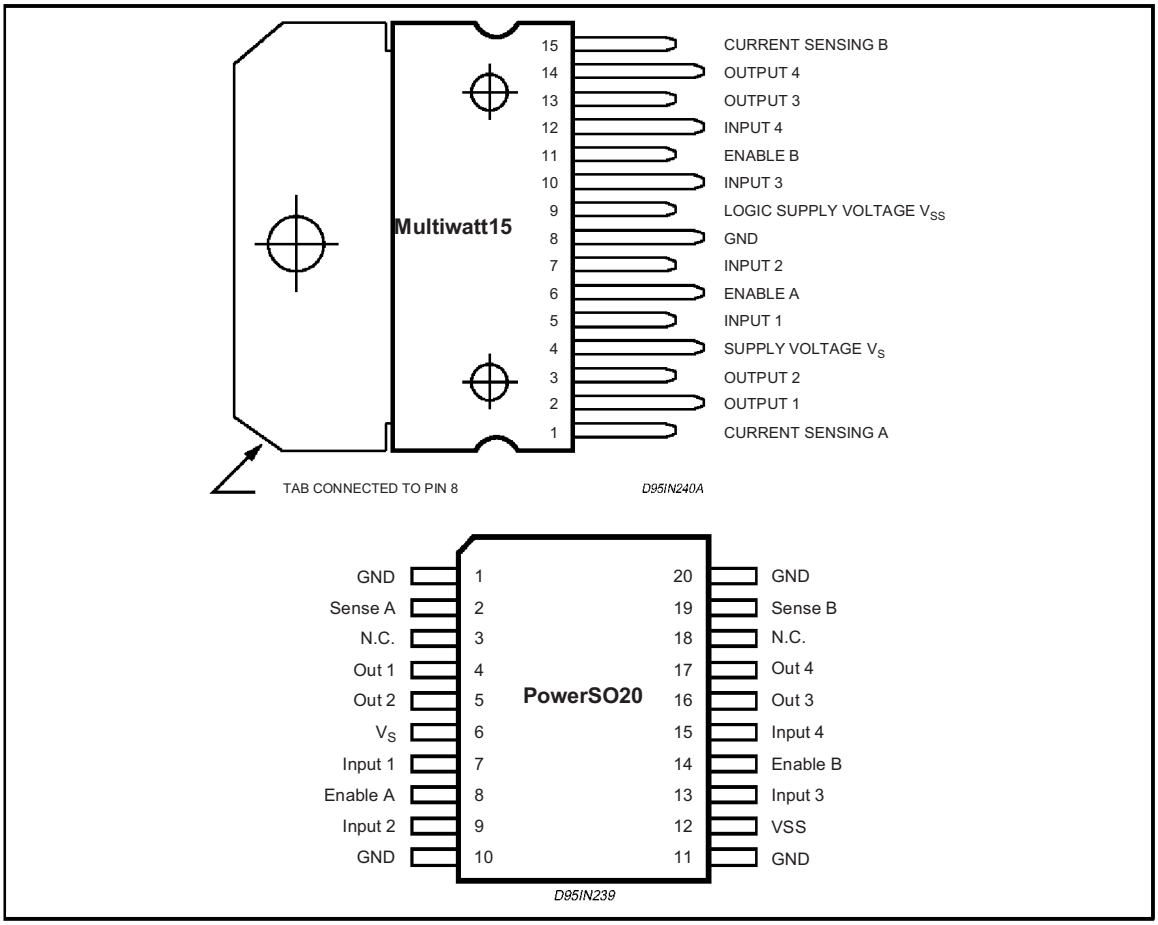
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_i,V_{en}	Input and Enable Voltage	–0.3 to 7	V
I_O	Peak Output Current (each Channel) – Non Repetitive (t = 100µs) –Repetitive (80% on –20% off; t_{on} = 10ms) –DC Operation	 3 2.5 2	 A A A
V_{sens}	Sensing Voltage	–1 to 2.3	V
P_{tot}	Total Power Dissipation (T_{case} = 75°C)	25	W
T_{op}	Junction Operating Temperature	–25 to 130	°C
T_{stg}, T_j	Storage and Junction Temperature	–40 to 150	°C

PIN CONNECTIONS (top view)



THERMAL DATA

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	Max.	–	3	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	°C/W

(*) Mounted on aluminum substrate

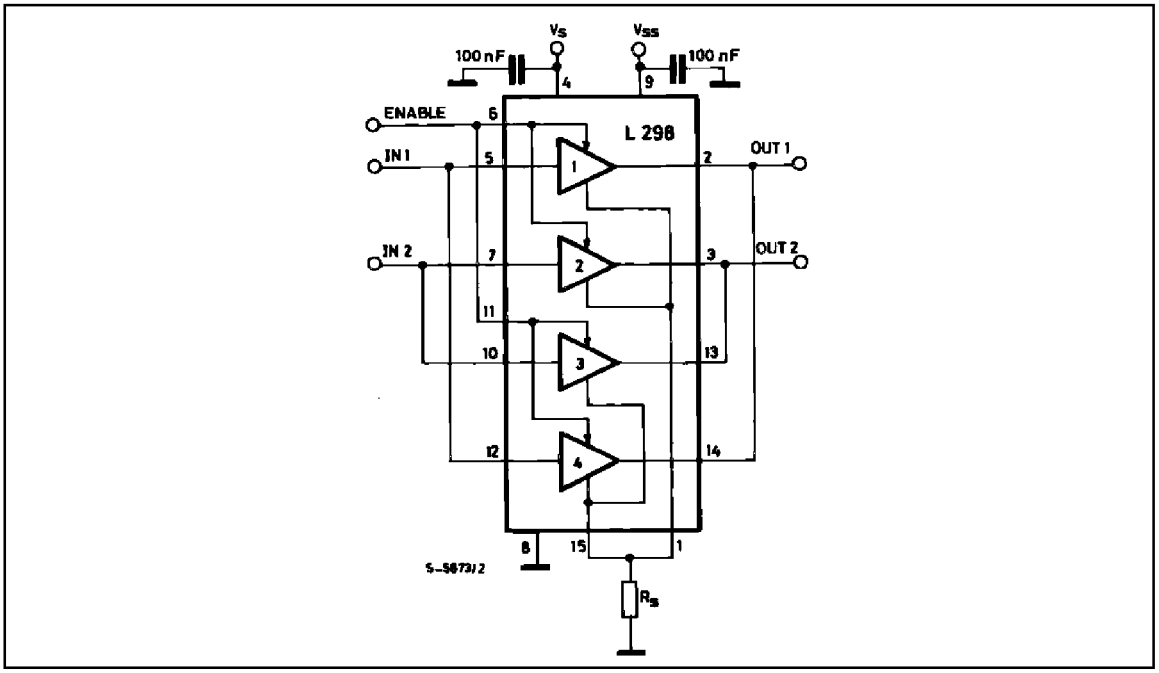
PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V_S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	VSS	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 4)	Operative Condition	V_{IH} +2.5		46	V
V_{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I_S	Quiescent Supply Current (pin 4)	V_{en} = H; I_L = 0 V_i = L V_i = H		13 50	22 70	mA mA
		V_{en} = L V_i = X			4	mA
I_{SS}	Quiescent Current from V_{SS} (pin 9)	V_{en} = H; I_L = 0 V_i = L V_i = H		24 7	36 12	mA mA
		V_{en} = L V_i = X			6	mA
V_{iL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V_{iH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		VSS	V
I_{iL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V_i = L			–10	µA
I_{iH}	High Voltage Input Current (pins 5, 7, 10, 12)	Vi = H ≤ V_{SS} –0.6V		30	100	µA
V_{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V_{en} = H	Enable High Voltage (pins 6, 11)		2.3		V_{SS}	V
I_{en} = L	Low Voltage Enable Current (pins 6, 11)	V_{en} = L			–10	µA
I_{en} = H	High Voltage Enable Current (pins 6, 11)	V_{en} = H ≤ V_{SS} –0.6V		30	100	µA
$V_{CEsat(H)}$	Source Saturation Voltage	I_L = 1A I_L = 2A	0.95	1.35 2	1.7 2.7	V V
$V_{CEsat(L)}$	Sink Saturation Voltage	I_L = 1A (5) I_L = 2A (5)	0.85	1.2 1.7	1.6 2.3	V V
V_{CEsat}	Total Drop	I_L = 1A (5) I_L = 2A (5)	1.80		3.2 4.9	V V
V_{sens}	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

Figure 7 : For higher currents, outputs can be paralleled. Take care to parallel channel 1 with channel 4 and channel 2 with channel 3.



APPLICATION INFORMATION (Refer to the block diagram)

1.1. POWER OUTPUT STAGE

The L298 integrates two power output stages (A; B). The power output stage is a bridge configuration and its outputs can drive an inductive load in common or differenzial mode, depending on the state of the inputs. The current that flows through the load comes out from the bridge at the sense output : an external resistor (R_{SA} ; R_{SB}.) allows to detect the intensity of this current.

1.2. INPUT STAGE

Each bridge is driven by means of four gates the input of which are In1 ; In2 ; EnA and In3 ; In4 ; EnB. The In inputs set the bridge state when The En input is high ; a low state of the En input inhibits the bridge. All the inputs are TTL compatible.

2. SUGGESTIONS

A non inductive capacitor, usually of 100 nF, must be foreseen between both Vs and Vss, to ground, as near as possible to GND pin. When the large capacitor of the power supply is too far from the IC, a second smaller one must be foreseen near the L298.

The sense resistor, not of a wire wound type, must be grounded near the negative pole of Vs that must be near the GND pin of the I.C.

Each input must be connected to the source of the driving signals by means of a very short path.

Turn-On and Turn-Off : Before to Turn-ON the Supply Voltage and before to Turn it OFF, the Enable input must be driven to the Low state.

3. APPLICATIONS

Fig 6 shows a bidirectional DC motor control Schematic Diagram for which only one bridge is needed. The external bridge of diodes D1 to D4 is made by four fast recovery elements (trr $\leq$ 200 nsec) that must be chosen of a VF as low as possible at the worst case of the load current.

The sense output voltage can be used to control the current amplitude by chopping the inputs, or to provide overcurrent protection by switching low the enable input.

The brake function (Fast motor stop) requires that the Absolute Maximum Rating of 2 Amps must never be overcome.

When the repetitive peak current needed from the load is higher than 2 Amps, a paralleled configuration can be chosen (See Fig.7).

An external bridge of diodes are required when inductive loads are driven and when the inputs of the IC are chopped; Shottky diodes would be preferred.

This solution can drive until 3 Amps In DC operation and until 3.5 Amps of a repetitive peak current.

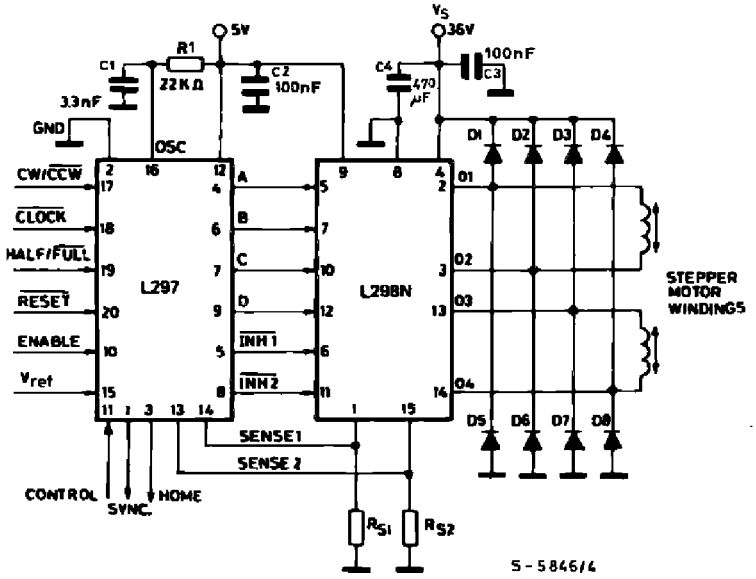
On Fig 8 it is shown the driving of a two phase bipolar stepper motor ; the needed signals to drive the inputs of the L298 are generated, in this example, from the IC L297.

Fig 9 shows an example of P.C.B. designed for the application of Fig 8.

Fig 10 shows a second two phase bipolar stepper motor control circuit where the current is controlled by the I.C. L6506.

Figure 8 : Two Phase Bipolar Stepper Motor Circuit.

This circuit drives bipolar stepper motors with winding currents up to 2 A. The diodes are fast 2 A types.



$R_{S1} = R_{S2} = 0.5\,\Omega$

D1 to D8 = 2 A Fast diodes { $V_F \leq 1.2$ V @ I = 2 A ; trr ≤ 200 ns

MC78XX/LM78XX/MC78XXA

3-Terminal 1A Positive Voltage Regulator

Features

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

Description

The MC78XX/LM78XX/MC78XXA series of three terminal positive regulators are available in the TO-220/D-PAK package and with several fixed output voltages, making them useful in a wide range of applications. Each type employs internal current limiting, thermal shut down and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.

TO-220

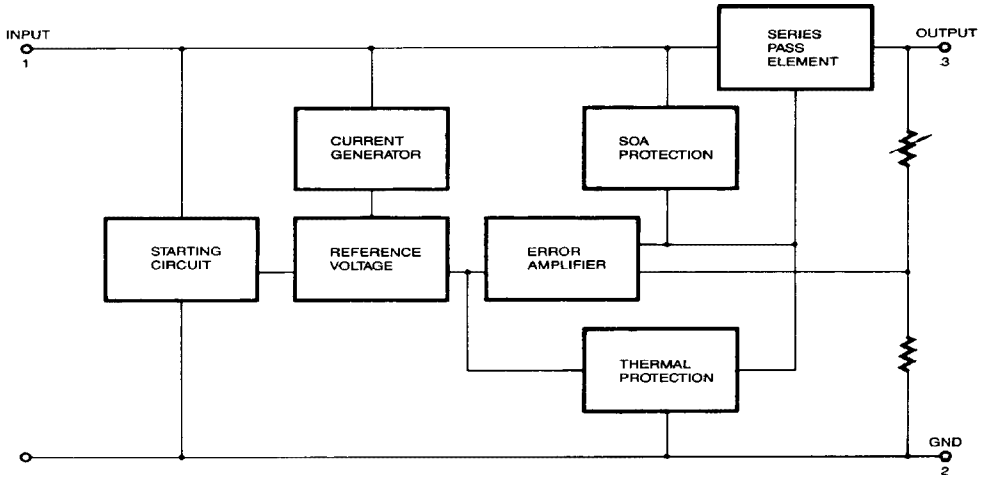
1

D-PAK

1

1. Input 2. GND 3. Output

Internal Block Digram



Rev. 1.0.1

Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Input Voltage (for V_O = 5V to 18V) (for V_O = 24V)	V_I V_I	35 40	V V
Thermal Resistance Junction-Cases (TO-220)	$R_{\theta JC}$	5	°C/W
Thermal Resistance Junction-Air (TO-220)	$R_{\theta JA}$	65	°C/W
Operating Temperature Range	T_{OPR}	0 ~ +125	°C
Storage Temperature Range	T_{STG}	-65 ~ +150	°C

Electrical Characteristics (MC7805/LM7805)

(Refer to test circuit ,0 C < T_J < 125°C, I_O = 500mA, V_I = 10V, C_I= 0.33 F, C_O= 0.1μF, unless otherwise specified)

Parameter	Symbol	Conditions		MC7805/LM7805			Unit
				Min.	Typ.	Max.	
Output Voltage	V_O	T_J =+25 °C		4.8	5.0	5.2	V
		5.0mA ≤ Io ≤ 1.0A, P_O ≤ 15W V_I = 7V to 20V		4.75	5.0	5.25	
Line Regulation (Note1)	Regline	T_J=+25 °C	V_O = 7V to 25V	-	4.0	100	mV
			V_I = 8V to 12V	-	1.6	50	
Load Regulation (Note1)	Regload	T_J=+25 °C	I_O = 5.0mA to1.5A	-	9	100	mV
			I_O =250mA to 750mA	-	4	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	8.0	mA
Quiescent Current Change	I_Q	I_O = 5mA to 1.0A		-	0.03	0.5	mA
		V_I= 7V to 25V		-	0.3	1.3	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O= 5mA		-	-0.8	-	mV/ °C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A=+25 °C		-	42	-	μV/Vo
Ripple Rejection	RR	f = 120Hz V_O = 8V to 18V		62	73	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	15	-	mΩ
Short Circuit Current	I_{SC}	V_I = 35V, T_A =+25 °C		-	230	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_o due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Typical Applications

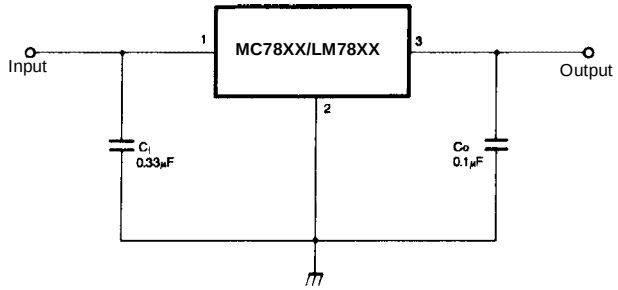


Figure 5. DC Parameters

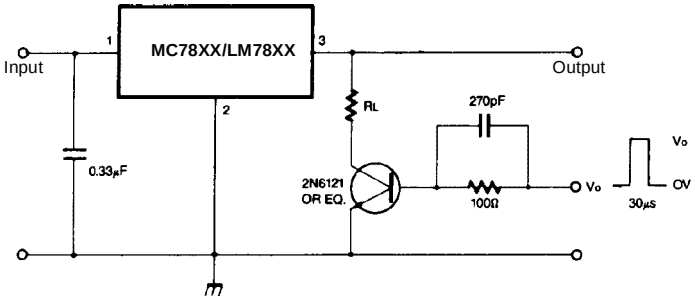


Figure 6. Load Regulation

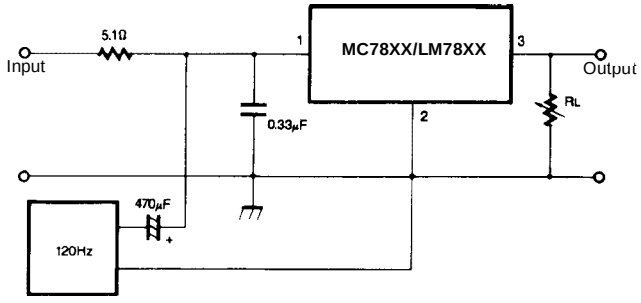


Figure 7. Ripple Rejection

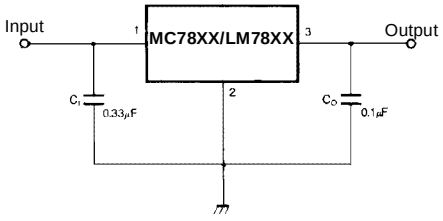
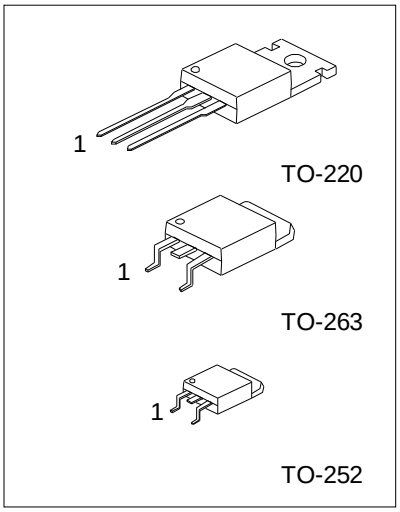


Figure 8. Fixed Output Regulator

UTC LM78XX LINEAR INTEGRATED CIRCUIT

3-TERMINAL 1A POSITIVE VOLTAGE REGULATOR



1: Input 2: GND 3: Output

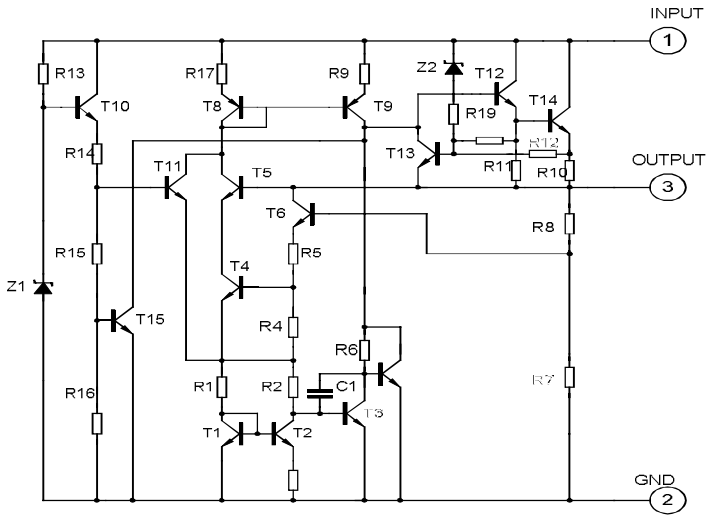
DESCRIPTION

The UTC 78XX family is monolithic fixed voltage regulator integrated circuit. They are suitable for applications that required supply current up to 1 A.

FEATURES

*Output current up to 1.5 A
*Fixed output voltage of 5V, 6V, 8V, 9V, 10V, 12V, 15V ,18V and 24V available
*Thermal overload shutdown protection
*Short circuit current limiting
*Output transistor SOA protection

TEST CIRCUIT



QW-R101-006,C

UTC LM7810 ELECTRICAL CHARACTERISTICS

(VI=16V, Io=0.5A, Tj= 0 C - 125°C, C1=0.33uF, Co=0.1uF, unless otherwise specified)(Note 1)

PARAMETER	SYMBOL	TEST CONDITIONS	MIN	TYP	MAX	UNIT
Output Voltage	Vo	Tj=25°C, Io=5mA - 1.0A	9.60	10.0	10.40	V
		VI =12.5V to 25V, Io=5mA - 1.0A,PD<=15W	9.50		10.50	V
Load Regulation	ΔVo	Tj=25°C,Io=5mA - 1.5A			100	mV
		Tj=25°C,Io=0.25A - 0.75A			50	mV
Line regulation	ΔVo	VI =13V to 25V,Tj=25°C			100	mV
		VI =13V to 25V, Tj=25°C,Io<=1A			100	mV
Quiescent Current	Iq	Tj=25°C, Io=<1A			8.0	mA
Quiescent Current Change	ΔIq	VI =12.6V to 25V			1.0	mA
	ΔIq	Io=5mA - 1.0A			0.5	mA
Output Noise Voltage	VN	10Hz<=f<=100kHz		58		μV
Temperature coefficient of Vo	Vo/△T	Io=5mA		-1.1		mV/°C
Ripple Rejection	RR	VI =13V - 23V,f=120Hz,Tj=25°C	56	72		dB
Peak Output Current	IPK	Tj=25°C		1.8		A
Short-Circuit Current	ISC	VI=35V, Tj=25°C		250		mA
Dropout Voltage	Vd	Tj=25°C		2.0		V

UTC LM7812 ELECTRICAL CHARACTERISTICS

(VI=19V, Io=0.5A, Tj= 0 C - 125°C, C1=0.33uF, Co=0.1uF, unless otherwise specified)(Note 1)

PARAMETER	SYMBOL	TEST CONDITIONS	MIN	TYP	MAX	UNIT
Output Voltage	Vo	Tj=25°C, Io=5mA - 1.0A	11.52	12.0	12.48	V
		VI =14.5V to 27V, Io=5mA - 1.0A,PD<15W	11.40		12.60	V
Load Regulation	ΔVo	Tj=25°C,Io=5mA - 1.5A			120	mV
		Tj=25°C,Io=0.25A - 0.75A			60	mV
Line regulation	ΔVo	VI =14.5V to 30V,Tj=25°C			120	mV
		VI =14.6V to 27V,Tj=25°C, Io=1A			120	mV
Quiescent Current	Iq	Tj=25°C, Io=<1A			8.0	mA
Quiescent Current Change	ΔIq	VI =14.5V to 30V			1.0	mA
	ΔIq	Io=5mA - 1.0A			0.5	mA
Output Noise Voltage	VN	10Hz<=f<=100kHz		75		μV
Temperature coefficient of Vo	Vo/△T	Io=5mA		-1.5		mV/°C
Ripple Rejection	RR	VI =15V - 25V,f=120Hz,Tj=25°C	55	72		dB
Peak Output Current	IPK	Tj=25°C		1.8		A
Short-Circuit Current	ISC	VI=35V, Tj=25°C		250		mA
Dropout Voltage	Vd	Tj=25°C		2.0		V