



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



Universidad Nacional
Autónoma de México



UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Temario

Capítulo 1 Introducción	1
Capítulo 2. Antecedentes.....	4
2.1 Microlitografía	4
2.2 Microlitografía óptica en el CCADET	6
2.2.1 <i>Mascarilla</i>	6
2.2.2 <i>Transferencia de la mascarilla</i>	8
2.3 Programación	11
2.3.1 <i>MPLAB IDE</i>	12
2.3.2 <i>Lenguaje Ensamblador</i>	12
2.3.3 <i>PICSTART Plus</i>	13
2.3.4 <i>Proteus de Labcenter Electronics</i>	14
2.4 PIC18F452	14
2.4.1 <i>Table Read/Table Write (TLBRD/TLBWT)</i>	17
2.4.2 <i>Timer</i>	18
2.4.3 <i>Interrupciones</i>	20
2.4.4 <i>PWM</i>	21
Capítulo 3 Diseño de Circuitos y Programación	22
3.1 Fuentes lineales de voltaje positivo	22
3.2 Etapa de Potencia-Motor a Pasos Bipolar	22
3.2.1 <i>L297</i>	23
3.2.2 <i>L298</i>	25
3.3 Servomotor PWM	27
3.4 Bibliotecas de funciones	28
3.4.1 <i>Display LCD</i>	29
3.4.2 <i>Teclado 4x4</i>	32
3.4.3 <i>BinarioBCD</i>	36
3.4.4 <i>Librería de Retardos</i>	37

3.5 Programas de cada etapa	39
3.5.1 <i>Mensajes del menú</i>	41
3.5.3 <i>Programación para motores</i>	46
3.6 PCBs	46
Capítulo 4 Diseño Mecánico	49
4.1 Motores	49
4.2 Motor a pasos	49
4.2.1 <i>Desplazamiento vertical</i>	50
4.3 Desplazamiento horizontal	54
4.3.1 <i>PWM</i>	55
Conclusiones	58

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, foto 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

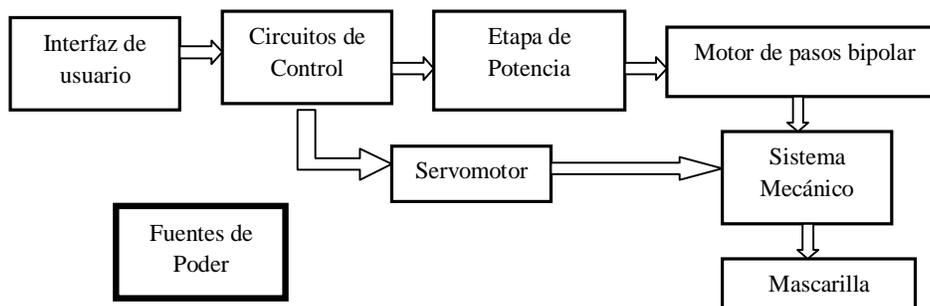


Figura 1.1

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.

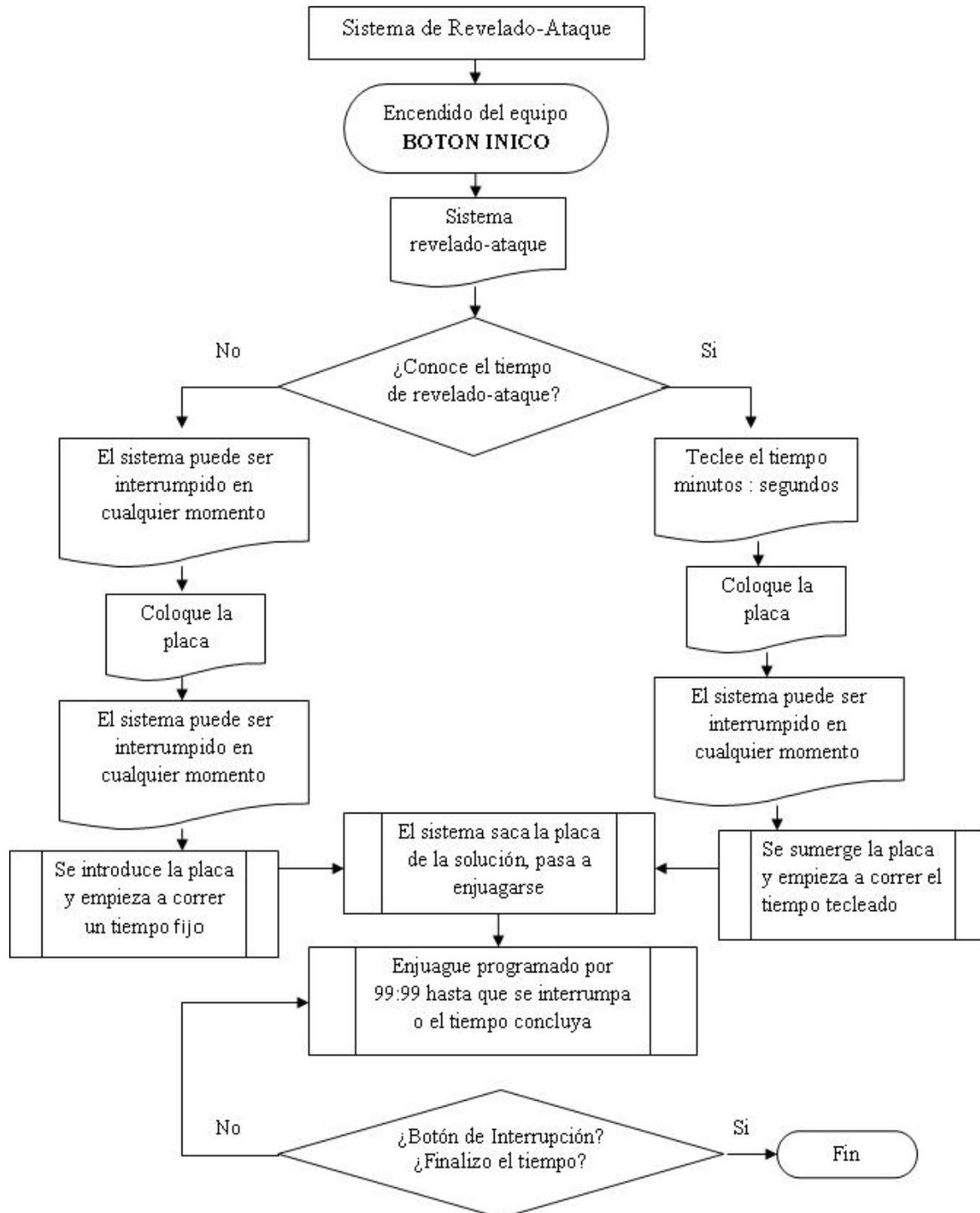


Figura 1.2

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.

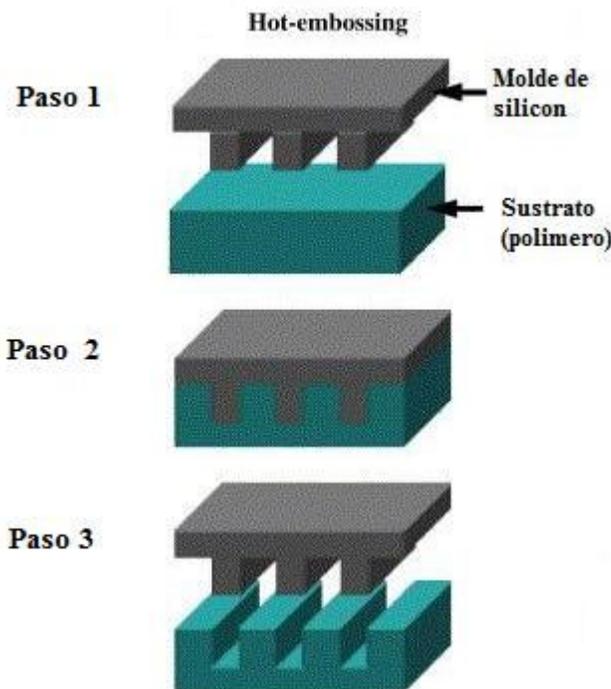


Figura 2.1

- Litografía óptica, actualmente esta técnica ha sido llevada al extremo, se está utilizando luz ultravioleta en el rango de $\lambda=248\text{nm}$ 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 $\lambda= (193-157) \text{ nm}$, 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.

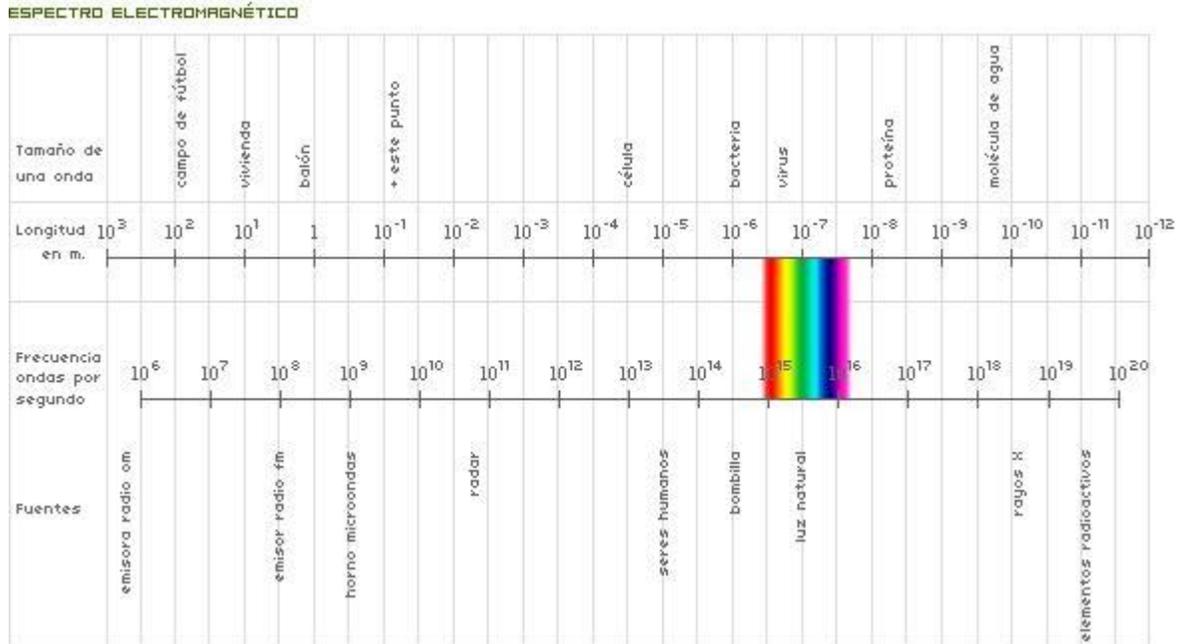


Figura 2.2.

- Litografía de Ultravioleta extremo (Extrem Ultraviolet EUV), tiene una longitud de onda $\lambda=13.5\text{nm}$, 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 tungsteno con recubrimiento de carburo de silicio o diamante.

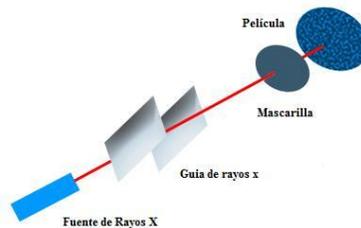


Figura 2.3

- 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 sensible (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 Litografía, 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, además de esto se inyecta un gas el cual es un atacante químico del material (fotorelina), 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 depósito 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 $\lambda=0.337\text{nm}$ (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 technology”.

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 enjuaga 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 Coating), 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 centrífuga 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 las 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.

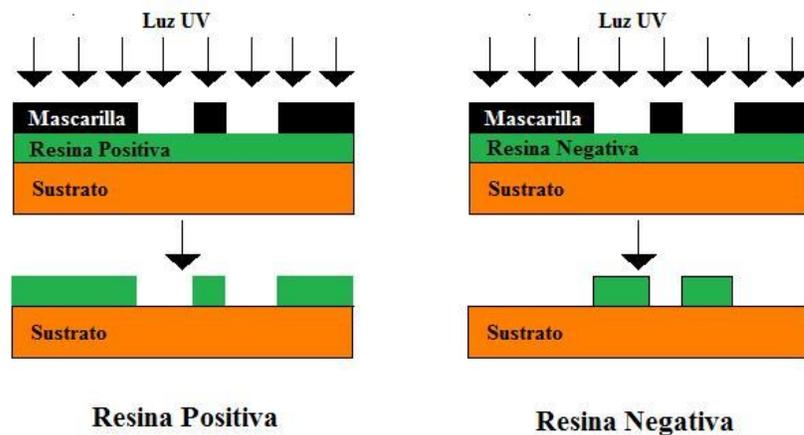


Figura 2.6

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 oxígeno	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.

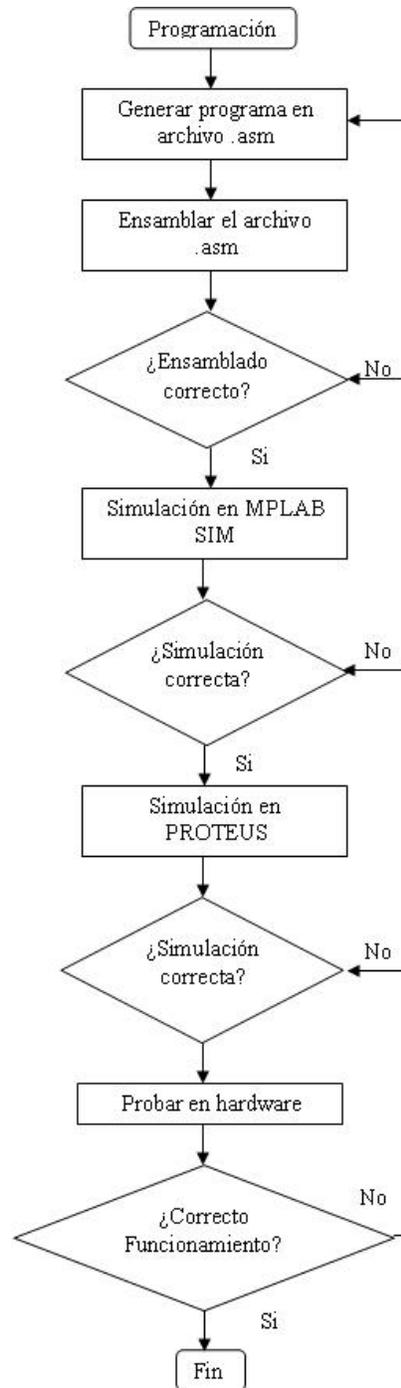


Figura 2.8

2.3.1 MPLAB IDE

La programación del PIC se realizó 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.

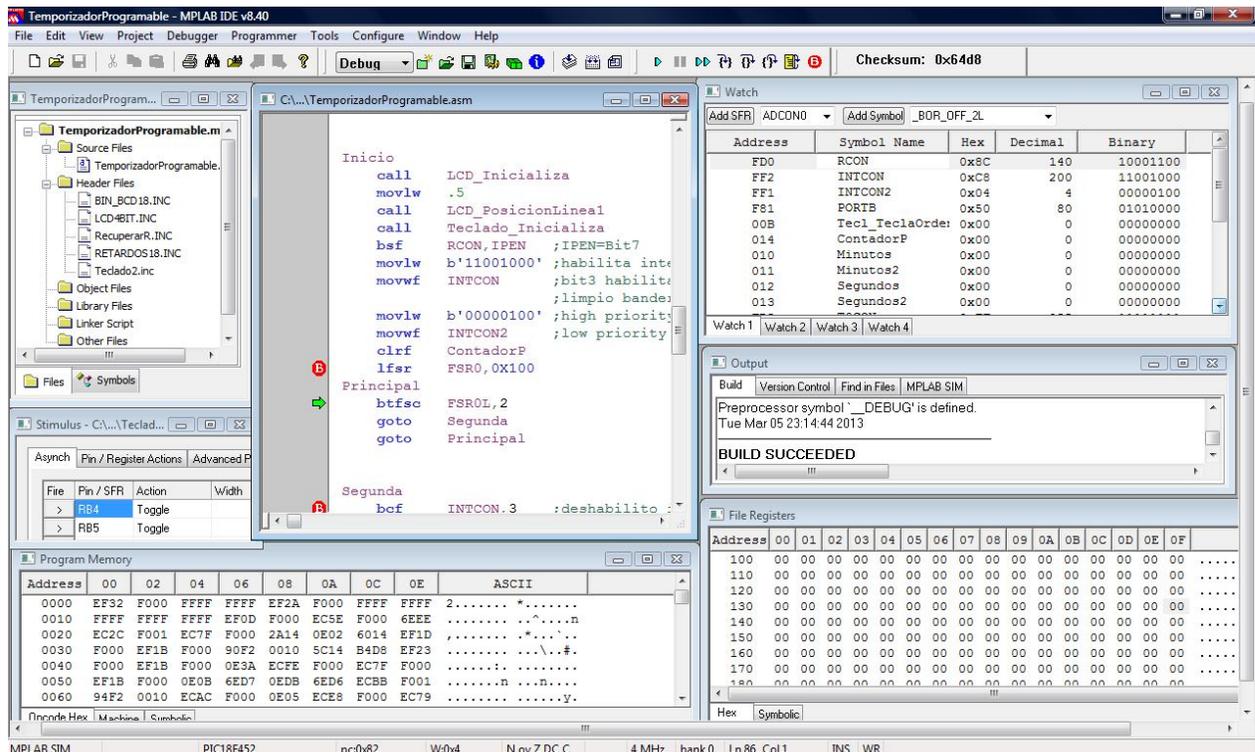


Figura 2.9

En MPLAB se puede programar en ensamblador o lenguaje C, en este caso se utilizó 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 máquina es tediosa y propiamente 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 utilizó 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 está el módulo VSM (Virtual System Modelling) con el que se puede simular en tiempo real un microcontrolador 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 Set 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 contenido 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 los 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.

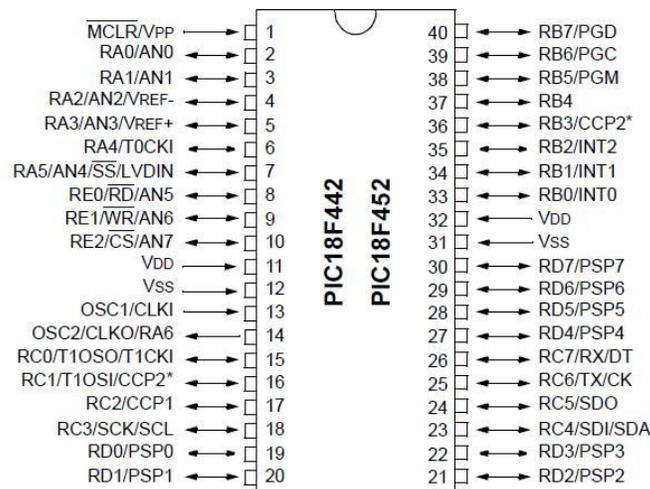


Figura 2.8

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 resistencias 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 (I2C, 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

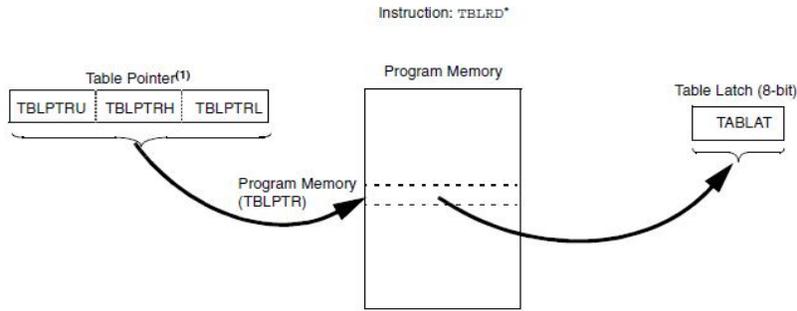
TBLPRT	Registro puntero de memoria de programa, especifica la dirección del byte
TABLAT	Registro que contiene el byte que fue leído o el que será escrito (8 bits). Los datos son transferidos a/de memoria de programa, byte por byte.
TBLRD	Instrucción de lectura de tablas de datos, Figura 2.9.
TBLWT	Instrucción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



Note 1: Table Pointer points to a byte in program memory.

Figura 2.9

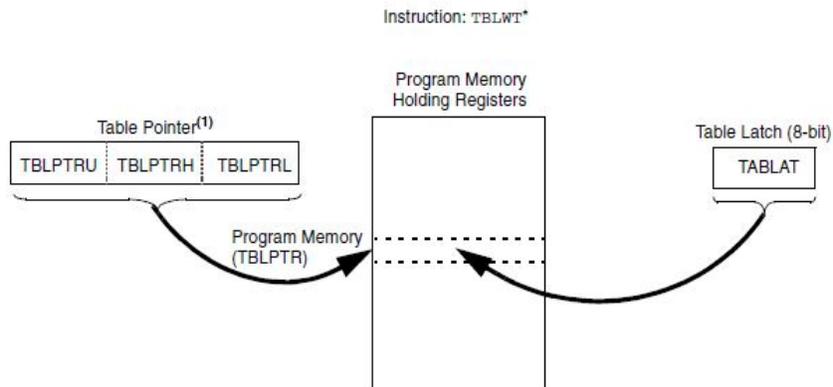


Figura 2.10

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 específico 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	T0PS0
--------	--------	------	------	-----	-------	-------	-------

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.

$$\text{DesbordamientoTimer0} = \frac{4}{F_{osc}} \times \text{Prescaler} \times [65535 - (\text{TMR0H} - \text{TMR0L})]$$

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

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

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

Por lo que TMR0H=B3 y TMR0L= 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.

$$\text{PeriodoPWM} = 4 \cdot (\text{PR2} + 1) \cdot \text{ValorPreescaler} \cdot \text{Tosc}$$

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

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

2. Definir el ciclo de trabajo y escribirlo en CCPR1L

$$\text{CicloTrabajo} = (\text{CCPR1L:CCPICON} \langle 5:4 \rangle) \cdot \text{Tosc} \cdot (\text{TMR2Prescaler})$$

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

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 LM7810 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.

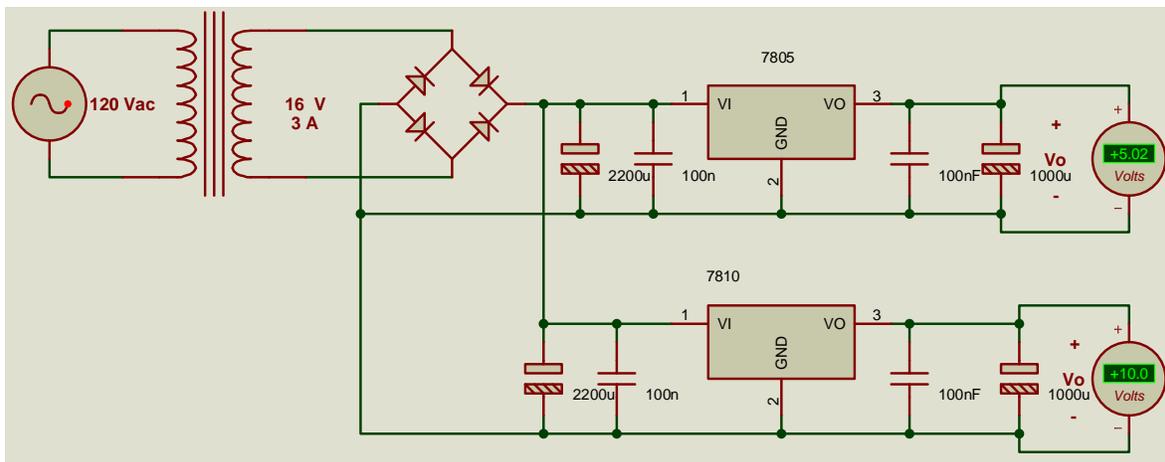


Figura 3.1

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.

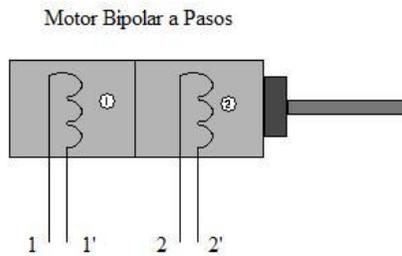


Figura 3.2

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 SO20 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

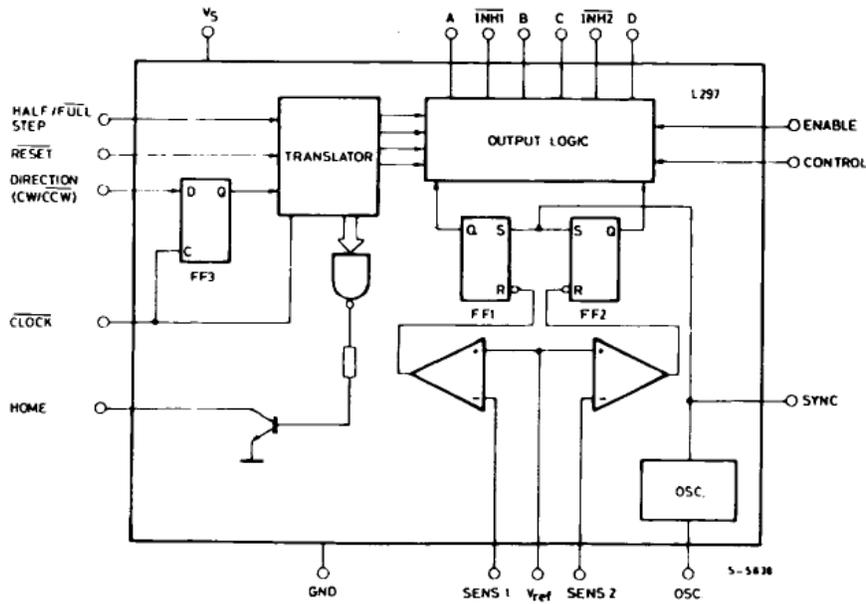


Figura 3.3

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	INH1	Pin 6 de L298
6	B	Pin 7 de L298
7	C	Pin 10 de L298
8	INH2	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/CCW	Giro, 5 V anti horario, 0 V horario
18	CLOCK	Señal TTL
19	HALF/FULL	Paso completo 0 V, medio paso 5 V
20	RESET	Figura 3.4(c)

Tabla 3.2

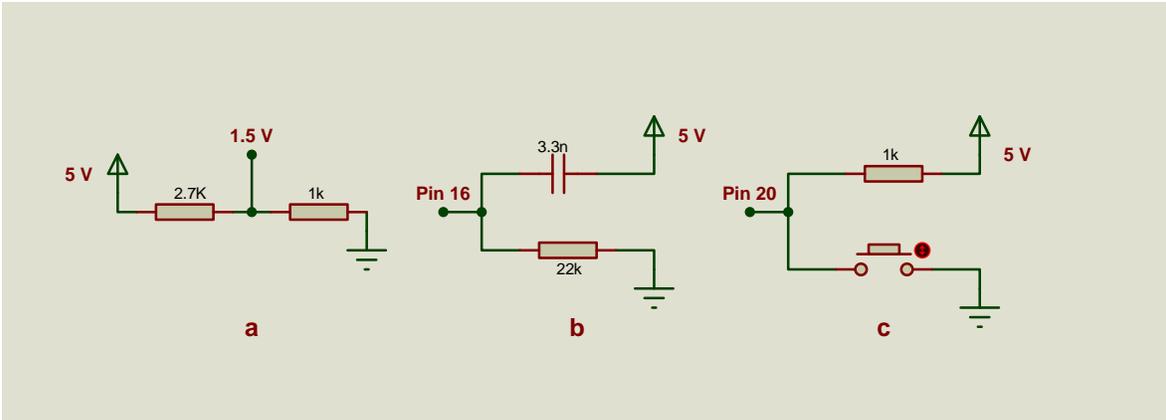


Figura 3.4

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.

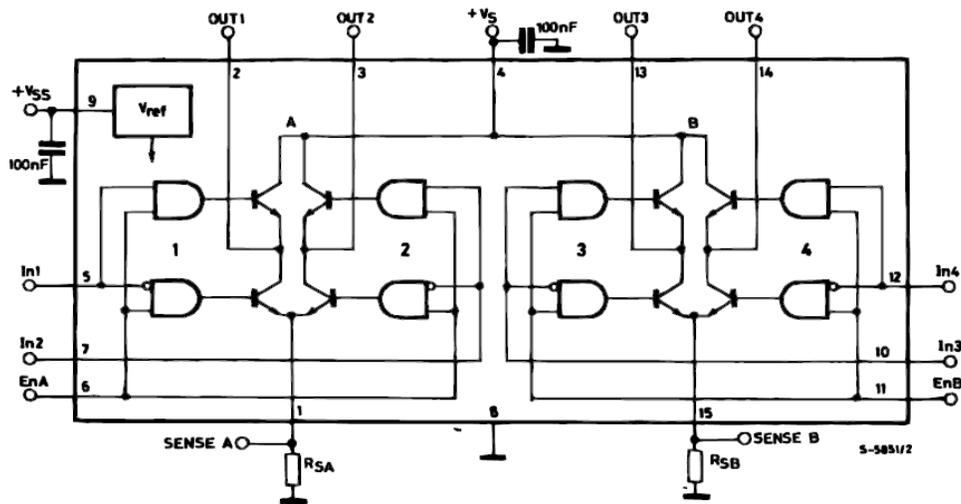


Figura 3.5

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 $R_{SA} = 0.5 \Omega$ a tierra
2	Output 1	Bobina 1
3	Output 2	Bobina 12
4	Supply Voltage V_s	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 V_{ss}	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 $R_{SB} = 0.5 \Omega$ a tierra

Tabla 3.3

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

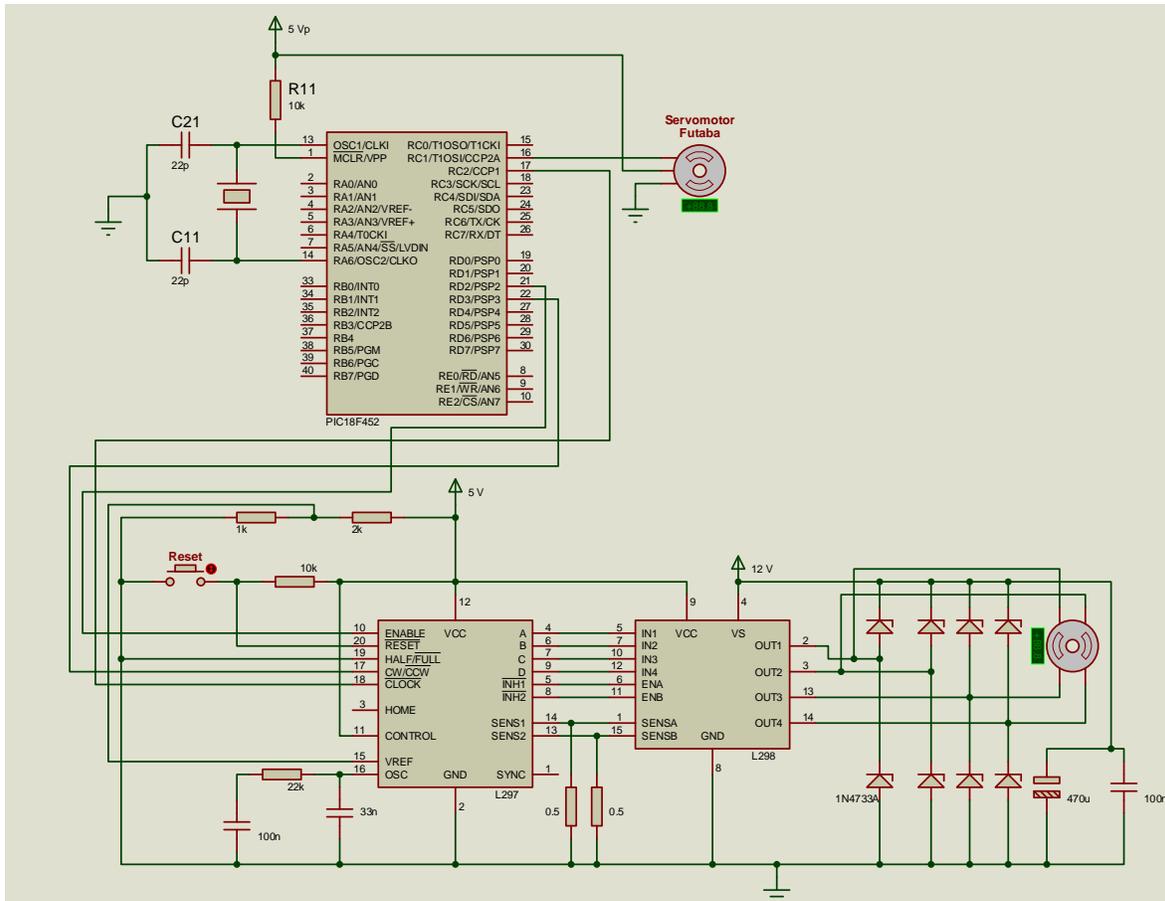


Figura 3.6

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 generó 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 programó 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 .inc 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.

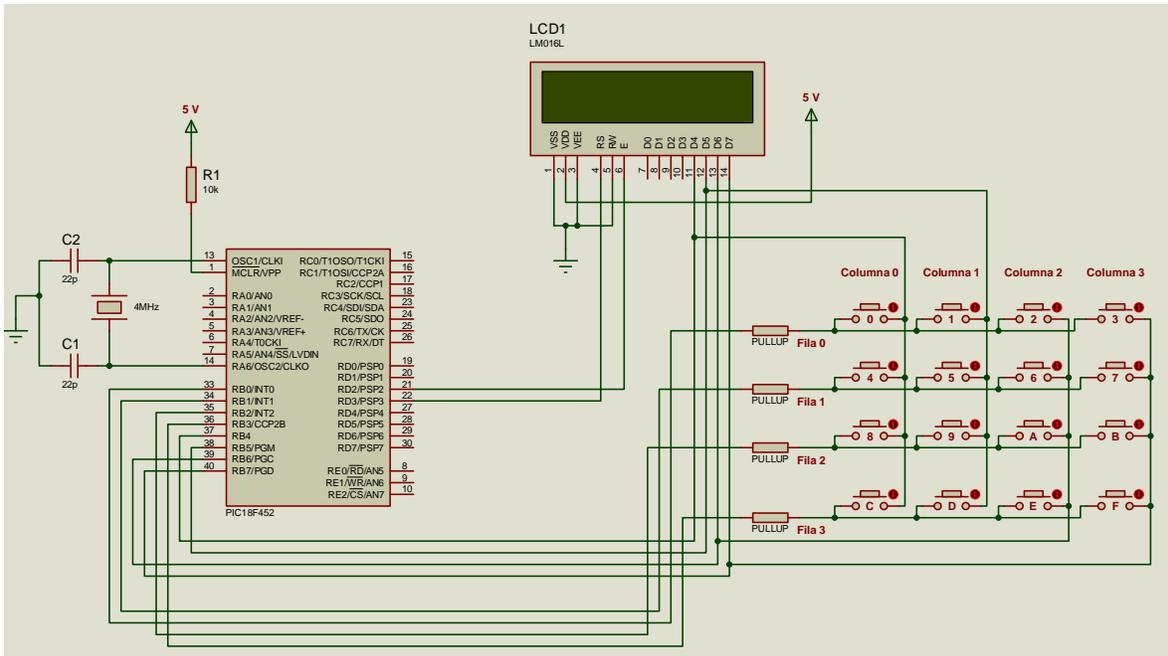


Figura 3.8

3.4.1 Display LCD

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

No. Pin	Nombre	Conexión
1	V _{SS}	0 V
2	V _{CC}	+5 V
3	V _{EE}	Contraste del LCD 0 V
4	R _s	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.

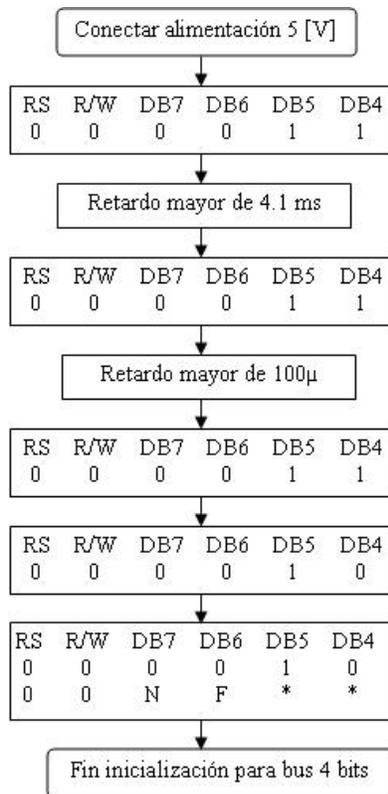


Figura 3.9

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_Pi nRS PORTD, 3
#DEFINE LCD_Pi nRW PORTD, 2
#DEFINE LCD_Pi nEnable 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_Pi nRS ; Activa el Modo Comando poniendo RS=0.
bcf LCD_Pi nRW ; En caso de que esté conectado le indica que se va a
; escribir en el LCD. NO LO TENGO CONECTADO
bcf LCD_Pi nEnable ; 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
  
```

```

movl w b' 00110000' ; SEGUNDA INSTRUCCIÓN
call LCD_Escri beLCD
call Retardo_200micros
movl w b' 00110000' ; TERCERA INSTRUCCIÓN
call LCD_Escri beLCD
movl w b' 00100000' ; Interface de 4 bits.
call LCD_Escri beLCD

; 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_Escri beLCD
andl w 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
andl w 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
movl w 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_Pi nEnable; Permite funcionamiento del LCD mediante un pequeño
bcf LCD_Pi nEnable ; 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.
movl w b' 0000110'
goto LCD_EnviaComando

LCD_Linea1 ; Cursor al principio de la Línea 1.
movl w b' 10000000' ; Dirección 00h de la DDRAM
goto LCD_EnviaComando

LCD_Linea2 ; Cursor al principio de la Línea 2.
movl w b' 11000000' ; Dirección 40h de la DDRAM
goto LCD_EnviaComando

LCD_PosicionLinea1 ; Cursor a posición de la Línea 1, a partir de la
iorl w 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
iorl w b' 11000000' ; dirección 40h de la DDRAM más el valor del
goto LCD_EnviaComando ; registro W.

LCD_OFF ; Pantalla apagada.
movl w b' 00001000'
goto LCD_EnviaComando

LCD_CursorON ; Pantalla encendida y cursor encendido.
movl w b' 00001110'
goto LCD_EnviaComando

LCD_CursorOFF ; Pantalla encendida y cursor apagado.
movl w b' 00001100'
goto LCD_EnviaComando

LCD_Borra ; Borra toda la pantalla, memoria DDRAM y pone el
movl w b' 00000001' ; cursor a principio de la línea 1.
goto LCD_EnviaComando

LCD_2Lineas4Bits5x7 ; Define la pantalla de 2 líneas, con caracteres
movl w b' 00101000' ; de 5x7 puntos y conexión al PIC mediante bus

LCD_EnviaComando
bcf LCD_Pi nRS ; Activa el Modo Comando, poniendo RS=0.

```

```

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

LCD_LineaEnBlanco
    movl w LCD_CaracteresPorLinea
    goto LCD_EnviaBlanco

LCD_UnEspacioBlanco
    movl w .1
    goto LCD_EnviaBlanco

LCD_DosEspaciosBlanco
    movl w .2
    goto LCD_EnviaBlanco

LCD_TresEspaciosBlanco
    movl w .3
    goto LCD_EnviaBlanco

LCD_EnviaBlanco
    movwf LCD_Auxiliar1 ; (LCD_Auxiliar1) se utiliza como contador.
LCD_EnviaOtrosBlanco
    movl w .1 ; 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_EnviaOtrosBlanco
    return

LCD_Byte
    movwf LCD_Auxiliar2 ; Guarda el valor de entrada.
    andl w 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.
    movl w ' ' ; 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
    andl w b' 00001111' ; Se queda con la parte baja.
    movwf LCD_Auxiliar1 ; Lo guarda.
    subl w 0x09 ; Comprueba si hay que representarlo con letra.
    btfss STATUS, C
    goto LCD_EnviaByteLetra
    movf LCD_Auxiliar1, W
    addl w '0' ; El número se pasa a carácter ASCII sumándole
    goto LCD_FinVisualizaDigitos ; el ASCII del cero y lo visualiza.
LCD_EnviaByteLetra
    movf LCD_Auxiliar1, W
    addl w 'A'-0x0A ; Sí, por tanto, se le suma el ASCII de la 'A'.
LCD_FinVisualizaDigitos
    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 diseñó 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

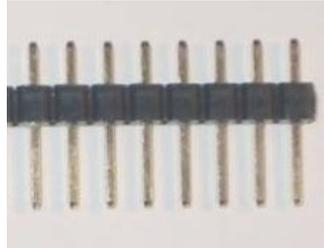


Figura 3.11

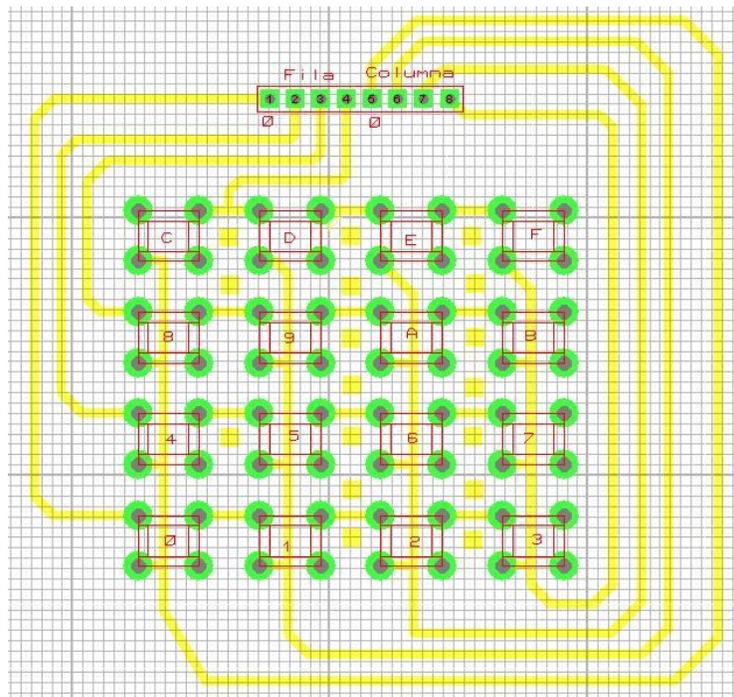


Figura 3.12

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

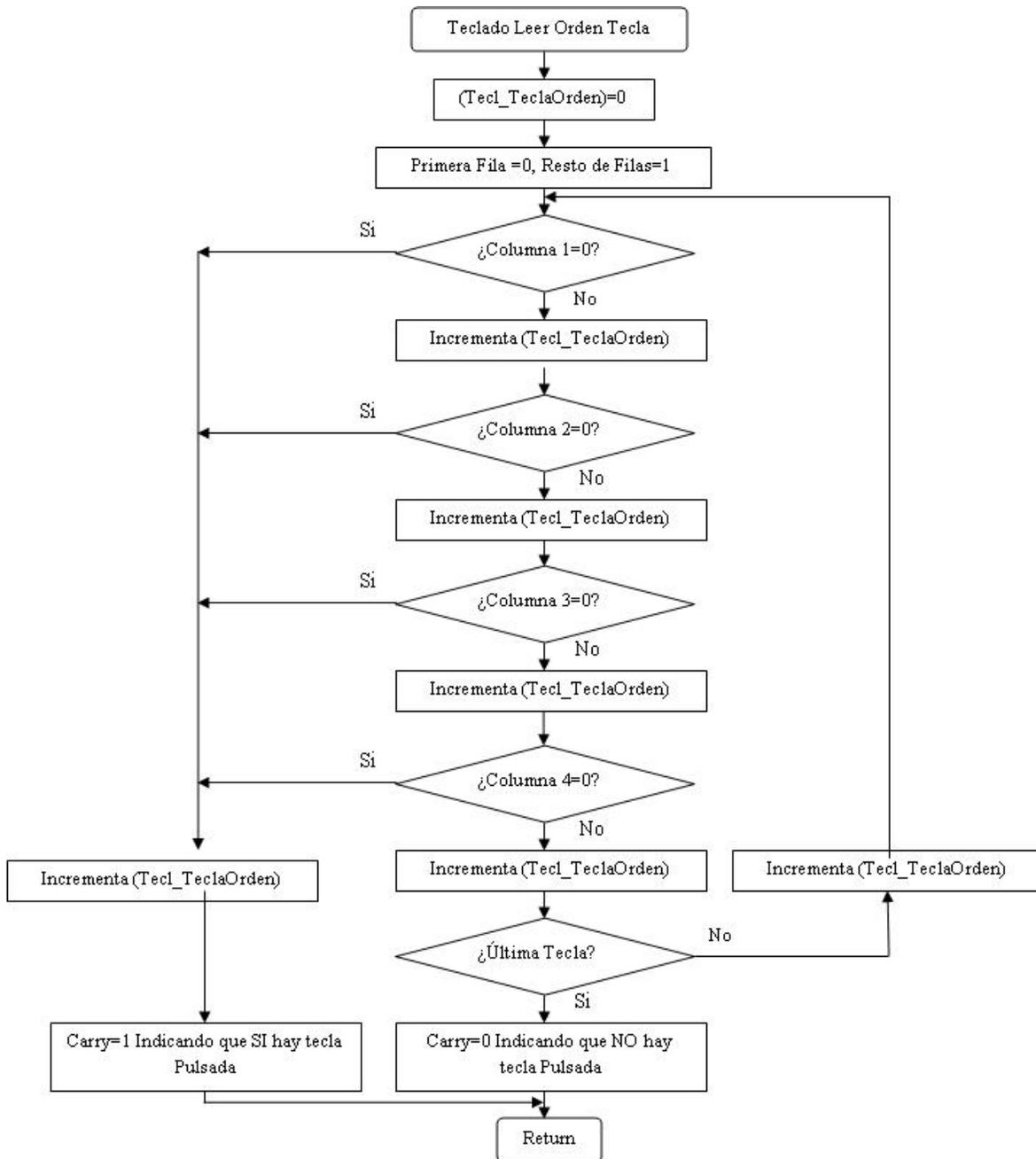


Figura 3.13

Librería para leer la tecla presionada según su orden.

```

CBLOCK 0x00B
Tecl_TeclaOrden
ENDC
Tecl_ÚltimaTecla EQU d'9' ;valor última tecla utilizada
Teclado_LeerHex
call Teclado_LeerOrdenTecla ;Leer Orden de la tecla Pulsada
btfs STATUS, C
goto Tecl_FinLeerHex
call Tecl_ConvertirteOrdenHex
bsf STATUS, C
  
```

```

Tecl _Fi nLeeHex
    return
Tecl _Convi erteOrdenHex
    rlcw   WREG
    addwf  PCL
    retlw  0h
    retlw  1h
    retlw  2h
    retlw  3h
    retlw  4h
    retlw  5h
    retlw  6h
    retlw  7h
    retlw  8h
    retlw  9h
Tecl ado_Fi nTabl aHex
Tecl ado_Ini ci al i za
    bcf    INTCON2, NOT_RBPU    ; Habilita resistencias pull-up del PuertoB
    movlw  b' 11110000'        ; 1=entrada
    movwf  TRISB
    call   Tecl ado_EsperaDej ePul sar
    return
Tecl ado_Comprobaci on          EQU    b' 11110000'
Tecl ado_EsperaDej ePul sar
    movlw  Tecl ado_Comprobaci on
    movwf  PORTB
Tecl ado_Si gueEsperando          ; Comprobaci3n de que la tecla es pul sada
    call   Retardo_20ms
    movf   PORTB, W
    sublw  Tecl ado_Comprobaci on
    btfss  STATUS, Z
    goto   Tecl ado_Si gueEsperando
    return
Tecl ado_LeeOrdenTecl a
    clrw   Tecl _Tecl aOrden
    movlw  b' 11111110'
Tecl _ChecaFi l a
    movwf  PORTB
    call   Retardo_1ms
Tecl _Col umna1
    btfss  PORTB, 4
    goto   Tecl _GuardaVal or
    incf   Tecl _Tecl aOrden, F
Tecl _Col umna2
    btfss  PORTB, 5
    goto   Tecl _GuardaVal or
    incf   Tecl _Tecl aOrden, F
Tecl _Col umna3
    btfss  PORTB, 6
    goto   Tecl _GuardaVal or
    incf   Tecl _Tecl aOrden, F
Tecl _Col umna4
    btfss  PORTB, 7
    goto   Tecl _GuardaVal or
    incf   Tecl _Tecl aOrden, F
Tecl _TerminaCol umnas
    movlw  Tecl _UI ti maTecl a
    subwf  Tecl _Tecl aOrden, W
    btfsc  STATUS, C
    goto   Tecl _NoPul sada
    rlcw   PORTB, W
    goto   Tecl _ChecaFi l a
Tecl _NoPul sada
    bcf    STATUS, C
    goto   Tecl _Fi nTecl adoLee
Tecl _GuardaVal or
    movf   Tecl _Tecl aOrden, W
    bsf   STATUS, C
Tecl _Fi nTecl adoLee
    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.

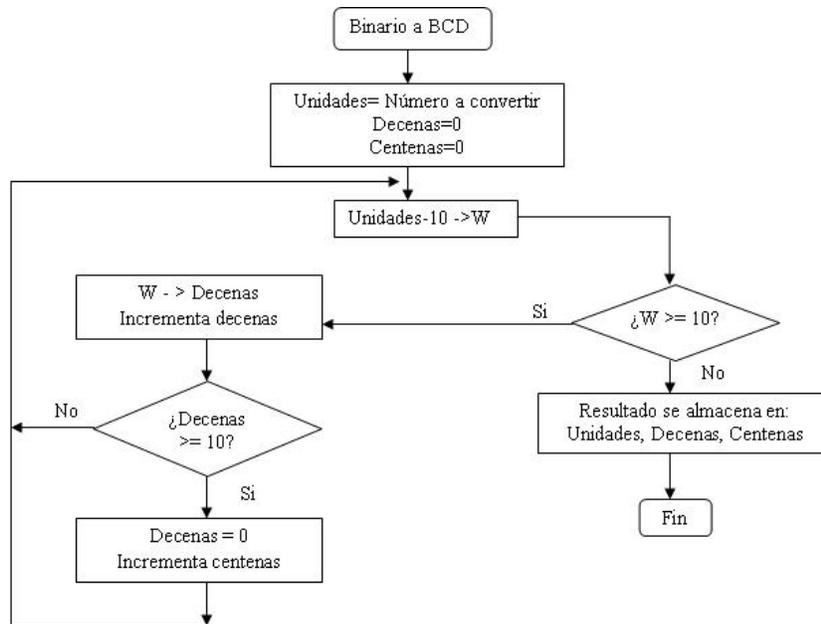


Figura 3.14

; 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

```

movl w    .10
subwf    BCD_Unidades, W
btfss    STATUS, C
goto     BIN_BCD_Fin
  
```

BCD_IncrementaDecenas

```

movwf    BCD_Unidades
incf     BCD_Decenas, F
movl w    .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:

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

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 utilizó 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:

$$2+1+1+(k \times 1)+(k-1) \times 1+2+(k-1) \times 2+2=5+4k$$

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

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

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_500mi cros      ; La llamada "call" aporta 2 ciclos máquina.
  nop                  ; Aporta 1 ciclo máquina sin hacer ninguna operación.
  movl w .164          ; Aporta 1 ciclo máquina.
  goto RetardoMi cros; Aporta 2 ciclos máquina.
Retardo_200mi cros
  nop
  movl w .64
  goto RetardoMi cros
Retardo_100mi cros
  movl w .31
  goto RetardoMi cros
Retardo_50mi cros
  nop
  movl w .14
  goto RetardoMi cros
Retardo_20mi cros
  movl w .5
RetardoMi cros
  movwf R_ContA
Rmi cros_Bucle
  decfsz R_ContA, F   ; (K-1)x1 cm (cuando no salta) + 2 cm (al saltar).
  goto Rmi cros_Bucle; Aporta (K-1)x2 ciclos máquina.
  return              ; El salto del retorno aporta 2 ciclos máquina.
  movl w .200
  goto Retardos_ms
Retardo_100ms
  movl w .100
  goto Retardos_ms
Retardo_50ms
  movl w .50
  goto Retardos_ms
Retardo_20ms
  movl w .20
  goto Retardos_ms
Retardo_10ms
  movl w .10
  goto Retardos_ms
Retardo_5ms
  movl w .5
  goto Retardos_ms
Retardo_2ms
  movl w .2
  goto Retardos_ms
Retardo_1ms
  movl w .1

Retardos_ms
  movwf R_ContB
R1ms_BucleExterno
  movl w .249
  movwf R_ContA

```

```

R1ms_Bucl eInterno
    nop
    decfsz R_ContA, F
    goto R1ms_Bucl eInterno
    decfsz R_ContB, F
    goto R1ms_Bucl eExterno
    return
Retardo_20s
    movl w . 200
    goto Retardo_1Deci ma
Retardo_10s
    movl w . 100
    goto Retardo_1Deci ma
Retardo_5s
    movl w . 50
    goto Retardo_1Deci ma
Retardo_2s
    movl w . 20
    goto Retardo_1Deci ma
Retardo_1s
    movl w . 10
    goto Retardo_1Deci ma
Retardo_500ms
    movl w . 5
Retardo_1Deci ma
    movwf R_ContC
R1Deci ma_Bucl eExterno2
    movl w . 100
    movwf R_ContB
R1Deci ma_Bucl eExterno
    movl w . 249
    movwf R_ContA
R1Deci ma_Bucl eInterno
    nop
    decfsz R_ContA, F
    goto R1Deci ma_Bucl eInterno
    decfsz R_ContB, F
    goto R1Deci ma_Bucl eExterno
    decfsz R_ContC, F
    goto R1Deci ma_Bucl eExterno2
    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.

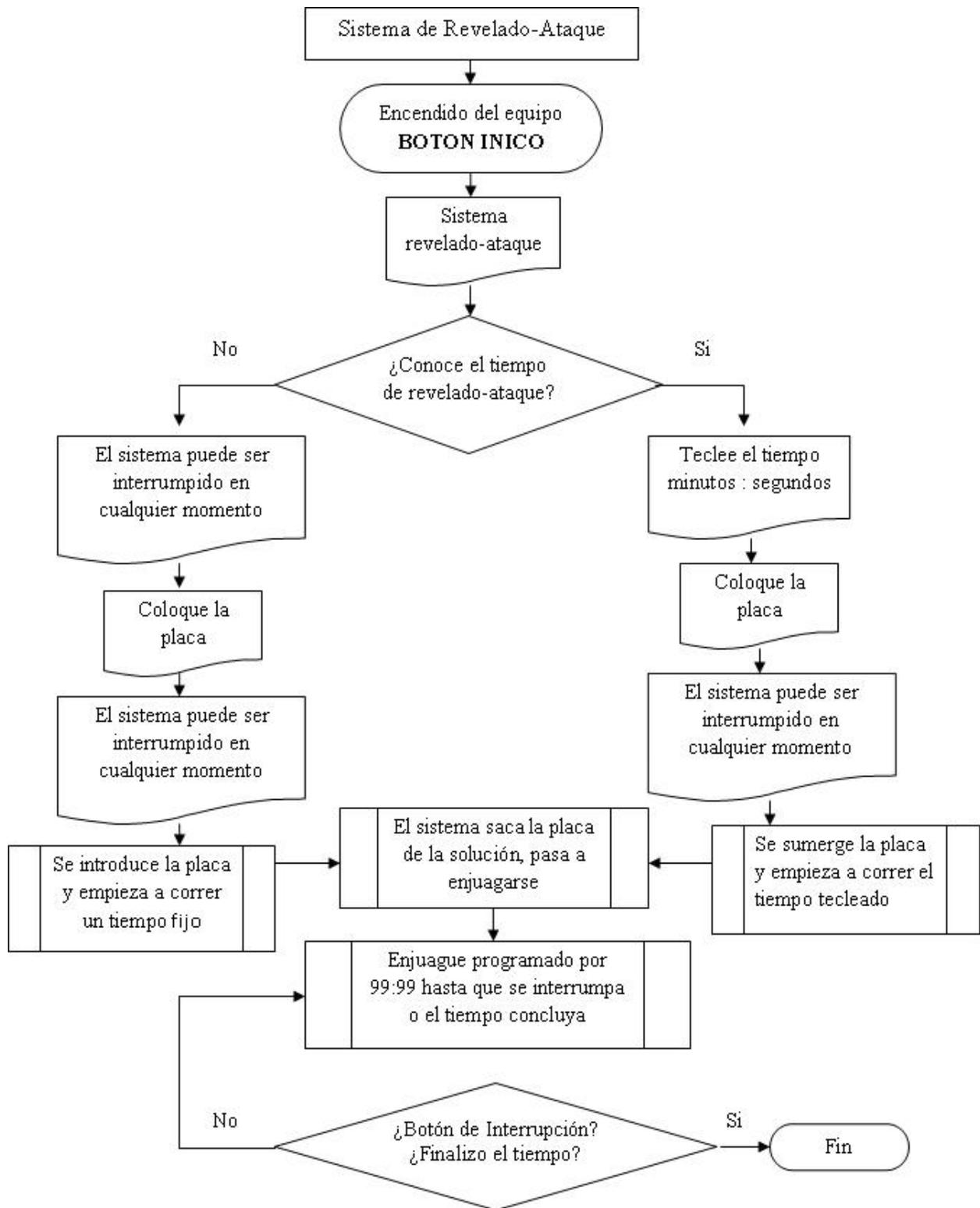


Figura 3.15

3.5.1 Mensajes del menú

```
LCD1
    clr f    TBLPTRU
    clr f    TBLPTRH
    movl w   0x06
    movwf   TBLPTRH
    movl w   0x00
    movwf   TBLPTRL
LCD11 ; ****"Revelado-Ataque"***
    TBLRD*+
    movf    TABLAT, W
    call    LCD_Character
    btfss   TBLPTRL, 4
    goto    LCD11
LCD11 i nea2 ; ****"R-A"****
    call    LCD_Linea2
    movl w   0x06
    movwf   TBLPTRH
    movl w   0x10
    movwf   TBLPTRL
LCD12
    TBLRD*+
    movf    TABLAT, W
    call    LCD_Character
    btfsc   TBLPTRL, 4
    goto    LCD12
    return
LCD2
    call    LCD_Borra
    call    LCD_Linea1
    movl w   0x06
    movwf   TBLPTRH
    movl w   0x20
    movwf   TBLPTRL
LCD21 ; ****"Colocar"****
    TBLRD*+
    movf    TABLAT, W
    call    LCD_Character
    btfss   TBLPTRL, 4
    goto    LCD21
LCD21 i nea2
    call    LCD_Linea2
    movl w   0x06
    movwf   TBLPTRH
    movl w   0x30
    movwf   TBLPTRL
LCD22 ; ****"Sustancias"***
    TBLRD*+
    movf    TABLAT, W
    call    LCD_Character
    btfsc   TBLPTRL, 4
    goto    LCD22
    return
LCD3
    call    LCD_Borra
    call    LCD_Linea1
    movl w   0x06
    movwf   TBLPTRH
    movl w   0x40
    movwf   TBLPTRL
LCD31 ; ****"Colocar placa"****
    TBLRD*+
    movf    TABLAT, W
    call    LCD_Character
    btfss   TBLPTRL, 4
    goto    LCD31
    return
LCD4
```

```

        call LCD_Borra
        call LCD_Li nea1
        movl w 0x06
        movwf TBLPTRH
        movl w 0x50
        movwf TBLPTRL
LCD41  ;"Interrupci on del "
        TBLRD*+,
        movf TABLAT, W
        call LCD_Caracter
        btfsc TBLPTRL, 4
        goto LCD41
LCD41 i nea2
        call LCD_Li nea2
        movl w 0x06
        movwf TBLPTRH
        movl w 0x60
        movwf TBLPTRL
LCD42 ;****"equi po con Al to"****
        TBLRD*+,
        movf TABLAT, W
        call LCD_Caracter
        btfss TBLPTRL, 4
        goto LCD42
        return
LCD6
        call LCD_Borra
        call LCD_Li nea1
        movl w 0x06
        movwf TBLPTRH
        movl w 0x90
        movwf TBLPTRL
LCD61 ;****"Tecl ear ti empo"****
        TBLRD*+,
        movf TABLAT, W
        call LCD_Caracter
        btfsc TBLPTRL, 4
        goto LCD61
LCD61 i nea2
        call LCD_Li nea2
        movl w 0x06
        movwf TBLPTRH
        movl w 0xA0
        movwf TBLPTRL
LCD62 ;****"mi nutos: segundos"****
        TBLRD*+,
        movf TABLAT, W
        call LCD_Caracter
        btfss TBLPTRL, 4
        goto LCD62
        return
LCD7
        call LCD_Borra
        call LCD_Li nea1
        movl w 0x06
        movwf TBLPTRH
        movl w 0xB0
        movwf TBLPTRL
LCD71 ;****"Fin"****
        TBLRD*+,
        movf TABLAT, W
        call LCD_Caracter
        btfsc TBLPTRL, 4
        goto LCD71
        return

org 0x600 ;Memori a donde se guardan l os mensajes
db "Revel ado-Ataque " ;1
db " (R-A) " ;1.2
db " Col ocar " ;2

```

```

db " Sustancias " ;2.2
db " Colocar placa " ;3.1
db " Interrupcion del " ;4.1
db " equipo con Alto" ;4.2
db " Tecl ear 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 si gui ente PORTD, 4 ; Pin 27 ->
#DEFINE no PORTD, 5 ; Pin 28 No

ORG 0x0000
goto Inicio
ORG 0x0008 ; Vector de Interrupciones de Prioridad Alta
goto I Segunda
ORG 0x0016
goto I Primera
I Primera
call Teclado_LeeHex
movwf POSTINCO
call LCD_Nibbl e
call Teclado_EsperaDej ePulsar
incf ContadorP, F
movlw .2
cpfst ContadorP
goto Puntos
goto FinInterrupcion
FinInterrupcion
bcf INTCON, 0 ; Limpia la bandera de interrupción del PuertoB
retfie
Puntos
subwf ContadorP, 0, 0
btfsc STATUS, Z
goto Manda
call Teclado_EsperaDej ePulsar
goto FinInterrupcion
Manda
movlw ':'
call LCD_Caracter
I Segunda
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 ; si gui ente
bsf TRISD, 5 ; no
call LCD1 ; ****"Revelado-Ataque"***
; ****"R-A"****
call Retardo_1s
call LCD2 ; ****"Colocar"****
; ****"Sustancias"****
YA call Retardo_1s
btfss si gui ente; (->)

```

```

        goto YA
        call LCD3
        call Retardo_1s
YA1     btfss si_guiente
        goto YA1
        call LCD4
        ; ****"Colocar placa"****
        ; (->)
        ; ****"Interrupcion del"***
        ; ****"equipo con Alto"****

YA2     call Retardo_1s
        btfss si_guiente
        goto YA2
        call LCD5
        ; ****"¿Tiempo R-A?"****
        ; ****"Si (->) No(No)"****

        call Retardo_1s

pregunta
        btfsc si_guiente
        goto PedirTiempo
        btfsc no
        goto carga
        goto pregunta

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

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

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

Principal
        btfsc FSROL, 2
        goto Segunda
        goto Principal

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

Inicio2
        movlw b'00000011'
        movwf TOCON
        movlw b'10100000'
        movwf INTCON
        bsf INTCON2, TMR0IP
        ; Desbordamiento
        ; prescale 1:16 modo 16 Bits para 1seg
        ; habilita las interrupciones y la del Timer0
        ; Bit2 Interrupcion alta prioridad

InicioTemporizador
        btfsc PORTD, 4
        goto InicioTimer0
        goto InicioTemporizador

InicioTimer0

```

```

    bsf    TOCON, TMR0ON    ; habilita Timer0 Bit7
    movl w 0x0B            ; 1 seg
    movwf TMR0H            ; 65535-62500=3035(0x0BDB) para 1 segundo
    movl w 0xDB            ; 1 seg
    movwf TMR0L            ; valores para que en 1 segundo se interrumpa

Bucla
    movl w .0
    cpfsgt Minutos2
    goto PuedeSerFin
    goto Bucla

PuedeSerFin
    movl w .0
    cpfsgt Segundos2
    goto Fin
    goto Bucla

Fin
    bcf    TOCON, TMR0ON    ; deshabilita Timer0 Bit7
    si fin
    btfsc si siguiente
    call LCD7
    goto si fin
    goto Inicio

CBLOCK 0x010
    Minutos
    Minutos2
    Segundos
    Segundos2
ENDC

RecuperarRe
    lfsr  FSR0, 0x100        ; Recupero valores de la memoria RAM
    movf  POSTNCO, WREG
    movwf Minutos
    movf  POSTNCO, WREG
    movwf Minutos2
    movf  POSTNCO, WREG
    movwf Segundos
    movf  POSTNCO, WREG
    movwf Segundos2
    movf  Minutos, WREG      ; Convierto a Minutos
    mull w .10              ; Multiplico por 10 y sumo
    movf  PRODL, WREG
    movwf Minutos
    addwf Minutos2, F
    movf  Segundos, WREG    ; Convierto a Segundo
    mull w .10              ; Multiplico por 10 y sumo
    movf  PRODL, WREG
    movwf Segundos
    addwf Segundos2, F
    return

MuestraRE
    movl w .0                ; mostrar reloj
    cpfseq Segundos2        ; Compare Segundos2 with 0, skip =
    goto NoEsCero
    goto SegundoEsCero

NoEsCero
    movf  Segundos2, WREG    ; Segundos->WREG
    subl w .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 Salir

SegundoEsCero
    movl w .0
    cpfseq Minutos2        ; Compare Minutos2 with 0, skip=0
    goto Continua1

```

```

        goto Salir1
Conti nua1
    decf  Mi nutos2, F
    movl w  .59
    movwf Segundos2
Salir1
    movl w  .5
    call  LCD_Posici onLi nea2
    movf  Mi nutos2, WREG
    call  BIN_a_BCD
    call  LCD_ByteComple to
    movl w  ':'
    call  LCD_Caracter
    movf  Segundos2, WREG
    call  BIN_a_BCD
    call  LCD_ByteComple to
    return

```

3.5.3 Programación para motores

```

PWM1 ; para motor a pasos
    movl w  .88 ; 1) PWM periodo
    movwf  PR2
    movl w  .44 ; 2) duty cycle
    movwf  CCP1L
    bcf    TRISC, 2 ; 3) configuración como salida
    movl w  b' 00000110' ; 4) Bit 2 Timer 2 on, bit 1y0 preescal er=16
    movwf  T2CON
    movl w  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
Pri meraP
    movl w  .255 ; 1) PWM periodo
    movwf  PR2
    movl w  .55 ; 2) duty cycle, va a cambiar para posicionar servomotor
    movwf  CCPR2L
    bcf    TRISC, 1 ; 3) configuración como salida
    movl w  b' 00000110' ; 4) Bit 2 Timer 2 on, bit 1y0 preescal er=16
    movwf  T2CON
    movl w  b' 00101100' ; 5) nunca cambia configuración (44)
    movwf  CCP2CON
    call  Retardo_1s
    movl w  .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)

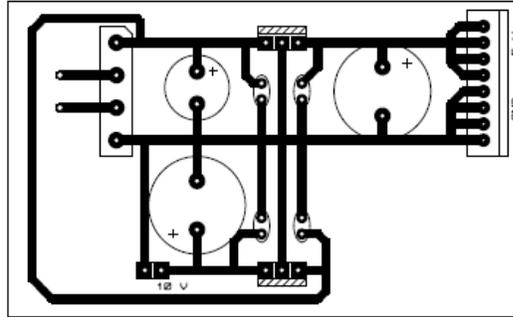


Figura 3.16

- Tarjeta para PIC18F452 (Figura 3.17)

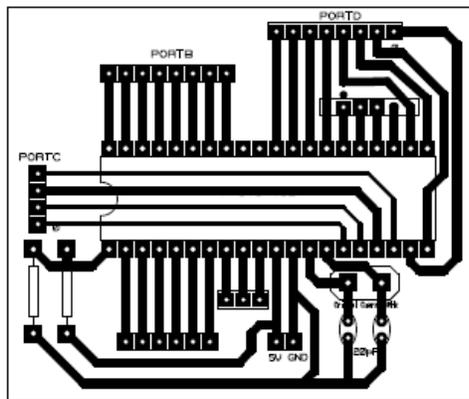


Figura 3.17

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

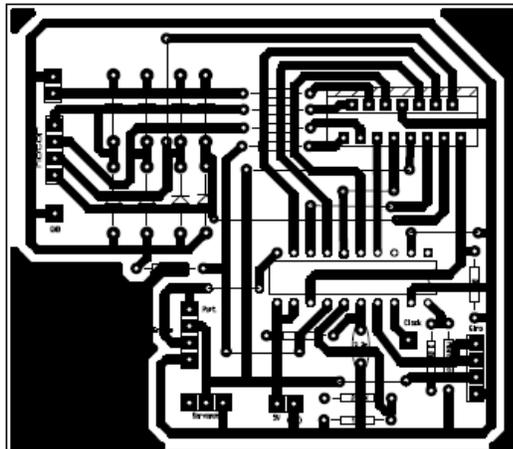


Figura 3.18

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

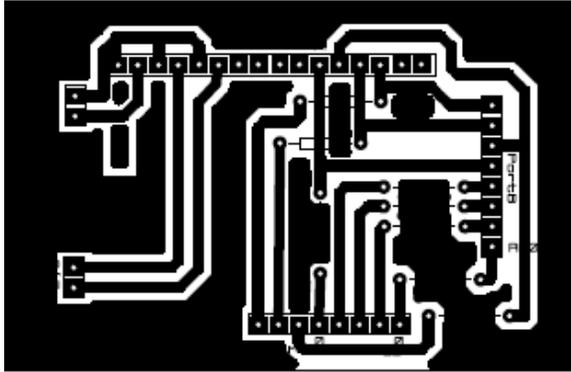


Figura 3.19

- Teclado matricial 4x4 (Figura 3.20)

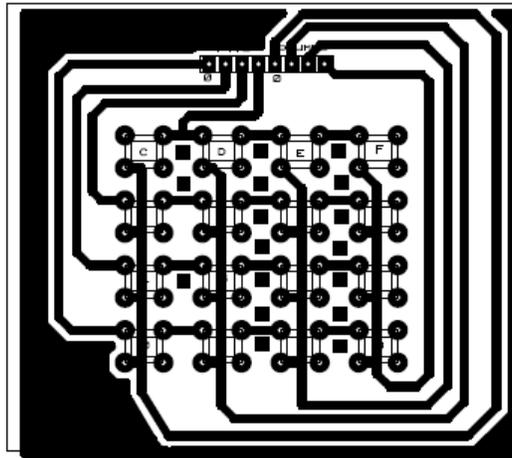


Figura 3.20

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.

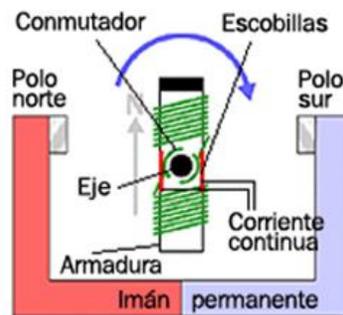


Figura 4.1

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.

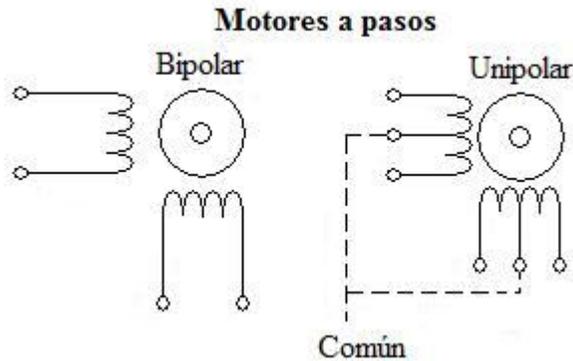


Figura 4.2

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 está determinado 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-cm ²	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

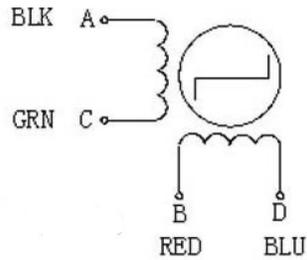


Figura 4.4

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

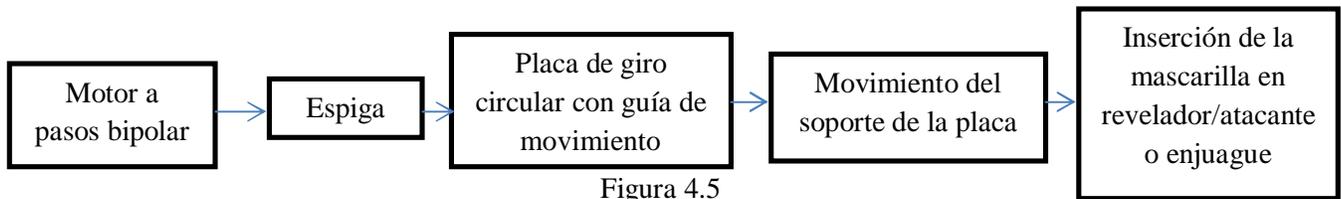


Figura 4.5

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

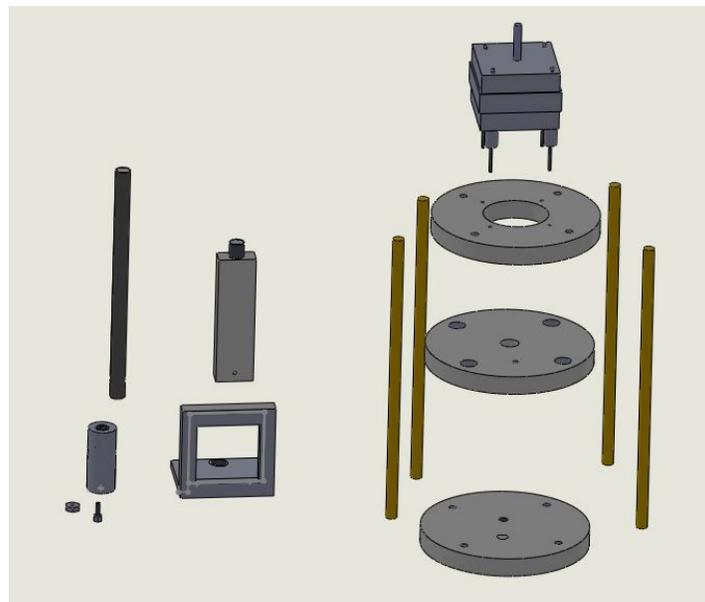


Figura 4.6

El motor a pasos bipolar esta acoplado 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.

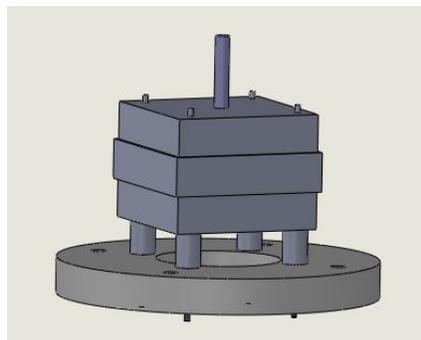


Figura 4.8

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 círculo 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 último un orificio lateral con rosca para sujetar al porta placa.

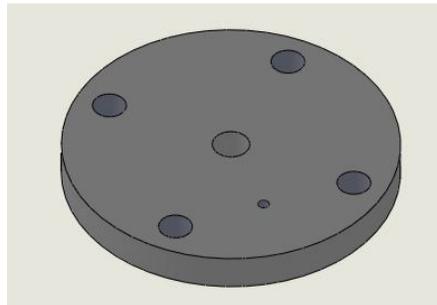


Figura 4.10

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.

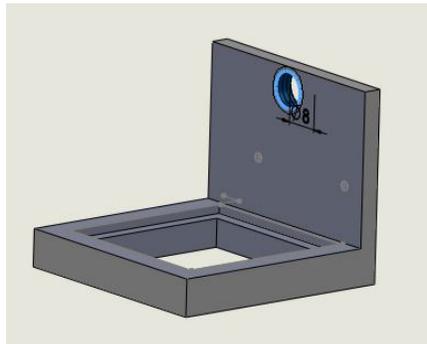


Figura 4.11

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

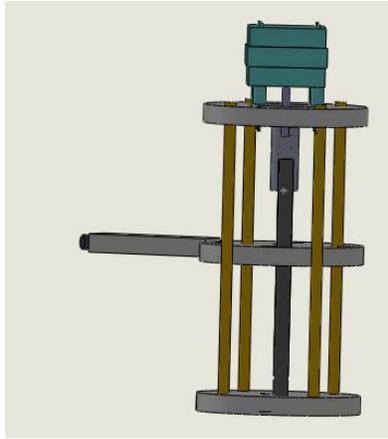


Figura 4.12

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 180° 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).

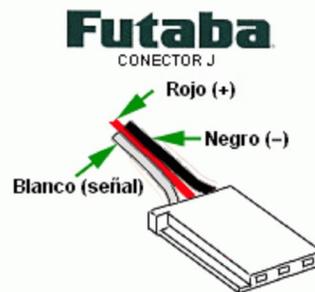


Figura 4.13

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 180° 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.

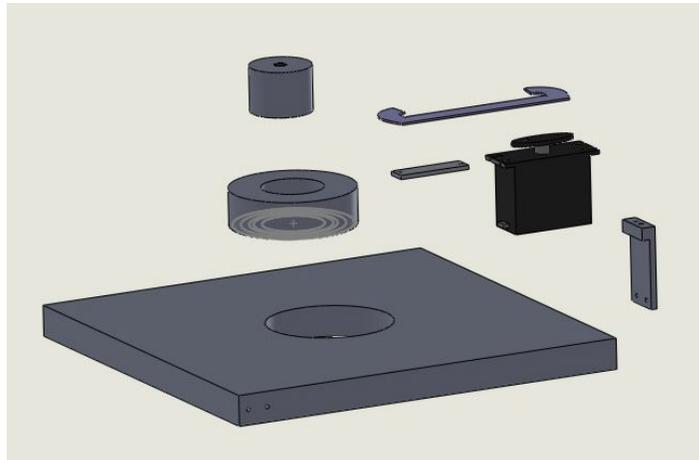


Figura 4.15

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

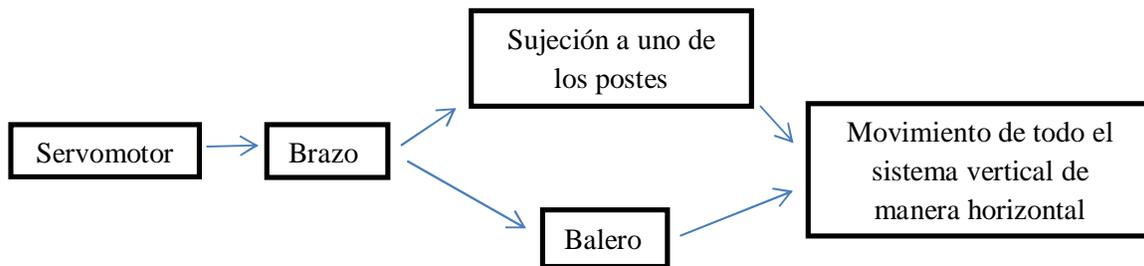


Figura 4.16

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.

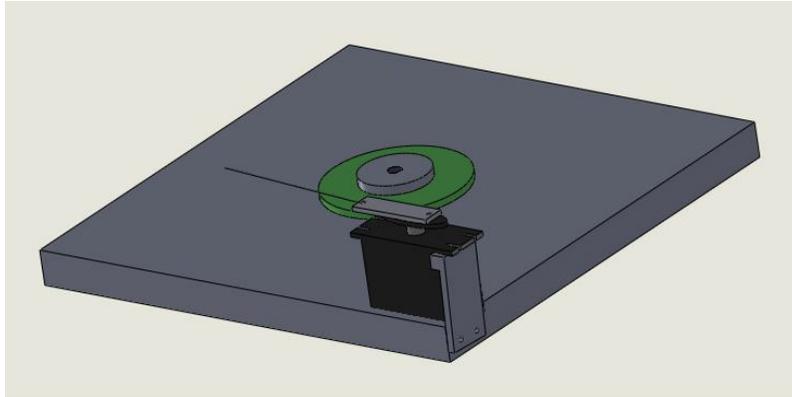


Figura 4.18

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 tuve 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 Microlithography, 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>

Litografía. (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