UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

CONTROL DE UN MOTOR PASO A PASO UTILIZANDO UN MICROCONTROLADOR

INFORME DE SUFICIENCIA

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ELECTRÓNICO

PRESENTADO POR:

PERCY OSWALDO TAIPE CORONADO

PROMOCIÓN
2004-I

LIMA-PERÚ
2011

CONTROL DE UN MOTOR PASO A PASO UTILIZANDO UN MICROCONTROLADOR

A mis padres
A mi esposa y a mi hijo,
el sentido de mi vida

SUMARIO

En el presente informe de suficiencia se describe la utilización de microcontroladores y herramientas de programación y simulación mediante un caso de estudio sencillo que servirá de referencia para una comprensión rápida y eficiente de las técnicas y alcances del uso de los microcontroladores para múltiples aplicaciones.

La oferta de herramientas de software, para programación y simulación de aplicaciones, es variada. En este informe se centra en el uso del MPLAB y del PROTEUS, para la programación y simulación de un circuito controlador de velocidad de un motor paso a paso (Full Step).

Se determina una velocidad máxima de rotación basada en un cristal de 20 MHz. Para una vuelta completa el motor paso a paso requiere de cuatro pasos, cada paso se establece a 5ms, por ello la vuelta completa es de 20ms y la velocidad máxima de giro es de 3000 RPM.

Dado que se utiliza un control binario de 1 a 255, la velocidad mínima para la aplicación es de 3000/255, es decir 11.7 RPM.

El control es realizado por software aplicado a un PIC 16F877, al cual se le añade elementos tri-state (para excitar a los devanados del motor) y pulsadores, para la selección de la velocidad, el cambio de giro, e inicio y parada.

También se hace una modificación en el código para torque fuerte (dos devanados excitados) con la correspondiente creación de una interfaz de de potencia.

ÍNDICE

INTRODUCCIÓN

Para este informe de suficiencia se ha seleccionado un proyecto sencillo cuyo principal propósito sea la de servir de base didáctica para la realización de cualquier otro proyecto.

Se trata de ser lo más detallado posible en la información proporcionada, sin embargo el lector deberá recurrir al material bibliográfico al que se hace referencia (Data Sheet), a los manuales de programación MPLAB y al de usuario del PROTEUS para su explotación máxima.

Se alcanzará material detallado sobre el PIC a utilizar para el desarrollo de la aplicación a fin de comprender específicamente los alcances y limitaciones del elemento seleccionado.

Se alcanzará el código fuente (en ensamblador) del cual se crea el archivo HEX (el ejecutable) además de las subrutinas que se encuentran en archivos separados (.asm), así cómo el esquema del circuito. Finalmente se describirá el funcionamiento del sistema desarrollado.

De las muchas tecnologías disponibles para el uso de microcontroladores se elige a la empresa Microchip al proporcionar herramientas de sencilla utilización, y que permite una rápida depuración del programa que se desarrolla.

El presente informe se divide en tres capítulos principales:

- Planteamiento de ingeniería del problema: En el cual se describe el problema, se plantea el objetivo, se evalúa el problema, se determina el alcance del trabajo (requerimientos) y finalmente se hace una síntesis del mismo.
- Marco Teórico: En tal capítulo se exponen las bases teóricas conceptuales más importantes para la comprensión del sistema descrito en el presente informe. Se explicará la teoría de los PaP (Motores paso a paso), los aspectos esenciales de los microcontroladores, los conceptos de programación básicos para el desarrollo, y finalmente se describen a las principales herramientas de simulación y programación.
- Metodología para la solución: Este consta de tres partes delimitadas. 1) La selección del tipo de PIC (Incluye descripción de características, programación y carga de instrucciones). La determinación de los controles de entrada (velocidades n1, n2, etc. ; sentido de giro, e inicio o parada) y del motor a usar, 2) Implementar el algoritmo (El algoritmo, su diagrama y explicación, además del código de programación), 3) La

simulación del diseño en el Proteus (construcción del modelo y valores de los dispositivos electrónicos y eléctricos; carga y ensamblaje del código fuente; simulación y verificación de valores)

CAPÍTULO I
PLANTEAMIENTO DE INGENIERÍA DEL PROBLEMA

El capítulo describe la importancia del motor paso a paso para las diversas aplicaciones de la industria. Así mismo la utilidad de los programas de simulación para evaluar el desempeño del circuito diseñado, cómo etapa previa a su construcción.

1.1 Descripción del Problema

Requerimiento de mecanismos con movimientos muy precisos.

El motor de paso a paso resuelve esta necesidad para las siguientes aplicaciones ejemplos: Disk-drive, impresoras, plotters, brazo y robots completos, patrón mecánico de velocidad angular, registradores XY, relojes eléctricos, control remoto, manipuladores, posicionamiento de válvulas en controles industriales, posicionamiento de piezas en general. Se aplica tanto a posicionamiento cómo a velocidad.

1.2 Objetivos del trabajo

Diseñar un circuito de control de un motor paso a paso (step-motor o PaP) mediante la utilización del PIC 16F877.

El circuito será diseñado y simulado en la herramienta PROTEUS, con fines de comprobación.

El informe pretende servir de una guía didáctica al interesado en la formulación, diseño y emulación de un proyecto que usa los elementos descritos.

1.3 Evaluación del problema

La evolución de la electrónica y la informática ha permitido rápidos diseños mediante los sistemas CAD (Diseño asistido por computador).

Recién a fines de los 70 (inicios de 1980), el centro de cómputo de la UNI dejó de utilizar las tarjetas perforadas o sombreadas. En esa época se enseñaba el FORTRAN y se tenía que esperar un día para la compilación y ejecución de un programa de pocas líneas. Años después llegaron los primeros MicroProffesors a la facultad y el aprendizaje se tornaba mucho más rápido e interactivo.

Actualmente existen muchas herramientas que ayudan al ingeniero con su propósito, es decir, diseñar un circuito y probarlo antes de implementarlo, pudiendo detectar en tiempo record cualquier falla, así cómo realizar cualquier modificación o mejora.

Los circuitos integrados especializados fueron también parte de la evolución. Los

PICs son una familia de microcontroladores tipo RISC (reduced instruction set computer) fabricados por Microchip Technology Inc. y derivados del PIC1650, originalmente desarrollado por la división de microelectrónica de General Instrument. PIC en realidad es PICmicro, sin embargo generalmente se utiliza cómo Peripheral Interface Controller (controlador de interfaz periférico).

Respecto a las aplicaciones que requieren mecanismos con movimientos muy precisos, hacer que un motor común de corriente continua gire una fracción de vuelta o un preciso número de vueltas es difícil o casi imposible.

Aún si se controlara con gran precisión la corriente necesaria, con la finalidad de fijar con exactitud el tiempo de arranque y parada del motor, sucede que al cortar la corriente la armadura no se detendrá, esto porque continúa moviéndose por inercia, y la inercia tendrá un valor muy difícil de determinar, porque depende de diversos factores: peso del rotor, fricción del eje sobre sus cojinetes, temperatura de las bobinas, núcleos de hierro, imanes e incluso y la del propio ambiente, además de otras variables del entorno y de la construcción.

La solución para aplicaciones de movimientos y posiciones precisas se logra mediante la utilización de los motores paso a paso, step motor o PaP, como son conocidos.

Actualmente la utilización de estas tres tecnologías (PaP, PIC y simuladores) son de gran difusión y utilización para los estudiantes e ingenieros de electromecánica y carreras afines

1.4 Alcance del trabajo

El trabajo se limita al control de velocidad de un motor realizado por la programación respectiva de un PIC 16F877, cuyo diseño es simulado en la herramienta PROTEUS.

Se establece que el sistema debe tener las siguientes características:

- Selección de velocidad.
- Selección de sentido de giro: horaria y antihoraria
- Selección de inicio y parada.

1.5 Síntesis del trabajo

El trabajo se sintetiza en lo siguiente:

1) Definir la topología del circuito.

- La selección del tipo de PIC.- Incluye descripción de características, programación y carga de instrucciones.
- La determinación de los controles de entrada (velocidades n1, n2, etc. ; sentido de giro, e inicio o parada) y del PaP a usar

2) implementar el algoritmo

- El algoritmo, su diagrama y explicación.
- El código de programación.- las instrucciones utilizadas para la implementación del algoritmo

3) Simular el diseño en el Proteus

- construcción del modelo.- Valores de los dispositivos electrónicos y eléctricos.
- Carga y ensamblaje del código fuente.
- Simulación y verificación de valores.

CAPÍTULO II
MARCO TEÓRICO CONCEPTUAL

En este capítulo se exponen las bases teóricas conceptuales más importantes para la comprensión del sistema descrito en el presente informe: PaP (Motores paso a paso), Microcontroladores, conceptos de programación y herramientas de simulación y programación.

2.1 Motores paso a paso (PaP)

Para hacer más didáctica la explicación de esta sección, será dividida en cinco subsecciones: aspectos generales, funcionamiento, tipos, parámetros, control.

2.1.1 Aspectos generales

Su aplicación es esencial cuando se exige un posicionamiento con un gran grado de exactitud o una excelente regulación de la velocidad. Ejemplos de aplicación son la robótica y tecnología aeroespacial, también el control de discos duros, flexibles, unidades de CD-ROM o de DVD e impresoras, además de en sistemas informáticos y para manipular y posicionar herramientas y piezas en general. Estos dispositivos son adecuados para la construcción de mecanismos en donde se requieren movimientos de gran exactitud.

La principal característica de los PaP es que se les puede mover un paso por vez por cada pulso que se le aplique. Este paso puede variar desde 90° hasta pequeños movimientos de tan solo 1.8° (incluso menos), esto quiere decir que se necesitarán 4 pasos en el primer caso (90°) y 200 para el segundo caso (1.8°), para completar un giro completo de 360°.

Los PaP poseen la capacidad de poder quedar fijos en una posición o bien totalmente libres. Si una o más de sus bobinas están energizadas, el motor estará fijo en la posición correspondiente y si no circula corriente por ninguna de sus bobinas quedará completamente libre.

El PaP posee dos partes:

- El estator, que está construido en base a cavidades en las que van depositadas las bobinas (excitadas convenientemente formarán los polos norte-sur de forma que se cree un campo magnético giratorio).
- Un rotor, construido por un imán permanente, con igual número de pares de polos que

el contenido en una sección de la bobina del estator; va montado sobre un eje soportado por dos cojinetes que le permiten girar libremente.

Por el medio que sea, al excitar el estator creando los polos N-S, y variando dicha excitación, de modo que el campo magnético formado efectúe un movimiento giratorio, el rotor seguirá el movimiento de dicho campo, produciéndose de este modo el giro del motor.

Se puede concluir que un motor PaP es un dispositivo que transforma los impulsos eléctricos en movimientos de giro controlados. Es posible hacer girar al motor en el sentido que se desee y la cantidad de vueltas y grados que sean determinados.

2.1.2 Funcionamiento

El funcionamiento de los motores eléctricos se basa en la fuerza ejercida por un campo electromagnético y que es creada cuando circula una corriente eléctrica a través de las bobinas. Si dicha bobina (estator) es mantenida en una posición mecánica fija y en su interior, influenciada por el campo electromagnético, es colocada otra bobina (rotor) recorrida por una corriente y capaz de girar sobre su eje, tenderá a buscar la posición de **equilibrio magnético**, esto quiere decir que orientará sus polos N-S hacia los polos S-N del estator, respectivamente.

Cuando el rotor alcanza la posición de equilibrio, el estator cambia la orientación de sus polos, tratará de buscar la nueva posición de equilibrio; manteniendo dicha situación de manera continua, se logrará un movimiento giratorio y continuo del rotor y simultáneamente la transformación de la energía eléctrica en mecánica, traducida cómo un movimiento giratorio.

La Figura 2.1 ilustra el funcionamiento de un motor PaP; se hace el supuesto que las bobinas L1 y L2 poseen un núcleo de hierro capaz de magnetizarse cuando ambas bobinas son recorridas por una corriente eléctrica. Por otra parte, el imán M puede girar libremente sobre el eje de sujeción central.

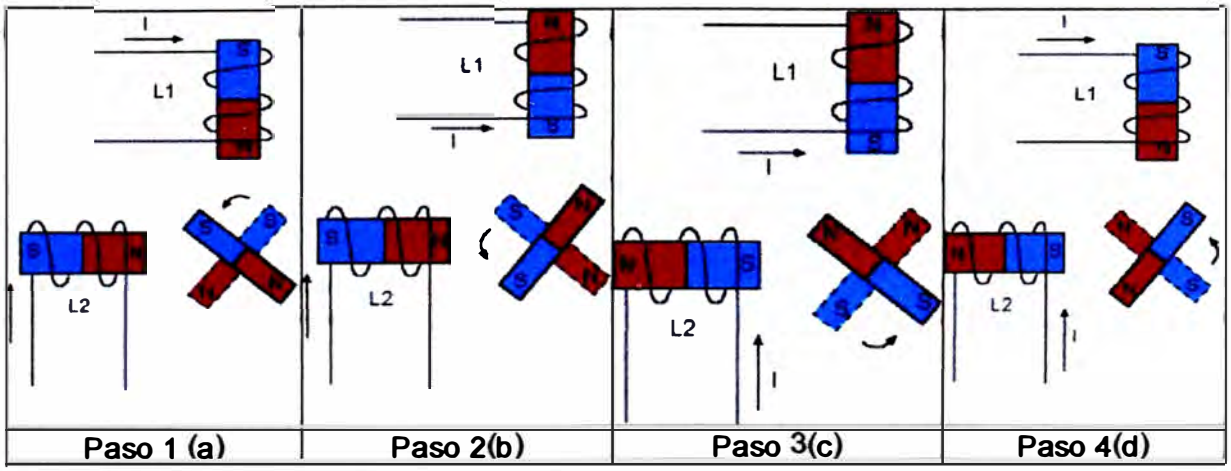


Figura 2.1 Principio de funcionamiento de un motor paso a paso bipolar

Al principio, al no aplicarse ninguna corriente a las bobinas (o fases) y con M en una posición cualquiera, el imán permanece en reposo al no someterle una fuerza externa.

Al circular corriente por ambas fases (Figura 2.1-a), se crearán dos polos magnéticos NORTE en la parte interna, bajo cuya influencia M se desplazará hasta la posición indicada en dicha figura.

Al invertir la polaridad de la corriente que circula por L1 se obtiene lo mostrado en la Figura 2.1-b y M se desplaza hasta la nueva posición de equilibrio, es decir, girado 90 grados en sentido contrario a las agujas del reloj.

Al invertir la polaridad de la corriente en L2, se obtiene lo mostrado en la Figura 2.1-c habiendo girado M otros 90 grados. Finalmente, al invertir otra vez el sentido de la corriente en L1, M gira otros 90 grados y se obtiene una revolución completa de dicho imán en cuatro pasos de 90 grados.

En consecuencia, si se mantiene la secuencia de excitación expuesta para L1 y L2, y dichas corrientes son aplicadas en forma de pulsos, el rotor avanzará pasos de 90 grados por cada pulso aplicado.

Se puede concluir que un motor PaP es un dispositivo electromecánico que convierte impulsos eléctricos en un movimiento rotacional constante y finito dependiendo de las características propias del motor.

Para conseguir motores PaP de paso mas fino, se debe aumentar el número de bobinas del estator, lo que aumentaría el costo y el volumen, con la consecuente pérdida del rendimiento del motor, lo que no hace inviable. Para lograr la solución más adecuada, se opta por la mecanización de los núcleos de las bobinas y el rotor en forma de hendiduras o dientes, creándose así micropolos magnéticos, tantos como dientes y estableciendo las situaciones de equilibrio magnéticos con avances angulares mucho menores, siendo posible conseguir motores de hasta de 500 pasos.

2.1.3 Tipos

Son dos tipos básicos (Figura 2.2):

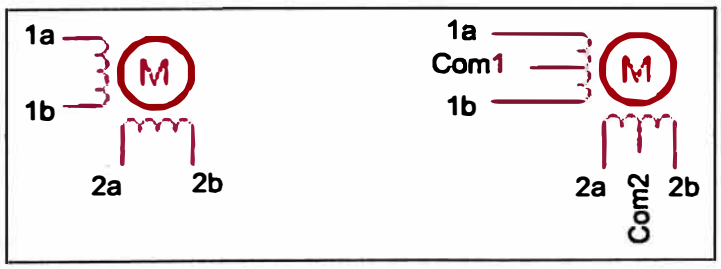


Figura 2.2 Bipolar y Unipolar

- Bipolares.- compuestos por bobinas. Solo tienen cuatro conexiones dos para cada bobina. La corriente fluye en ambos sentidos
- Unipolares.- compuestos por cuatro bobinas. Normalmente presentan seis cables, dos

para cada bobina y otro para alimentación de cada par de éstas, aunque en algunos casos se puede encontrar motores unipolares con cinco cables, básicamente es lo mismo, solo que el cable de alimentación es común para los dos pares de bobinas. Las corrientes fluyen en un solo sentido.

a. Motores bipolares

Las bobinas del estator se conectan en serie formando solamente dos grupos, los cuales se montan sobre dos estatores, tal y como se muestra en la Figura 2.3.

De este motor salen cuatro hilos que se conectan, al circuito de control, que realiza la función de cuatro interruptores electrónicos dobles, que permiten variar la polaridad de la alimentación de las bobinas. Con la activación y desactivación adecuada de dichos interruptores dobles, se obtiene secuencias adecuadas para que el motor pueda girar en un sentido o en otro.

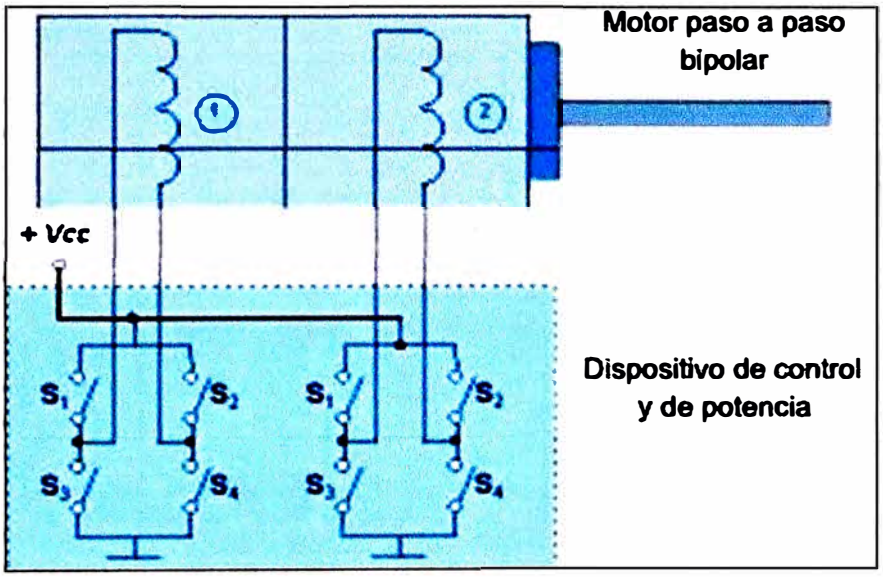


Figura 2.3 Control de motor bipolar

La existencia de varios bobinados en el estator de los motores de imán permanente, da lugar a varias formas de agrupar dichos bobinados, para que sean alimentados adecuadamente.

b. Motores Unipolares

En este tipo de motores, todas las bobinas del estator están conectadas en serie formando cuatro grupos. Se conectan dos a dos, también en serie, montándose sobre dos estatores diferentes, tal y como se aprecia en la Figura 2.4.

Del motor paso a paso salen dos grupos de tres cables, uno de los cuales es común a dos bobinados. Los seis terminales que parten del motor, deben ser conectados al circuito de control, el cual, se comporta como cuatro conmutadores electrónicos que, al ser activados o desactivados, producen la alimentación de los cuatro grupos de bobinas con que está formado el estator. Si se genera una secuencia adecuada de

funcionamiento de estos interruptores, se puede producir saltos de un paso en el número y sentido que se desee.

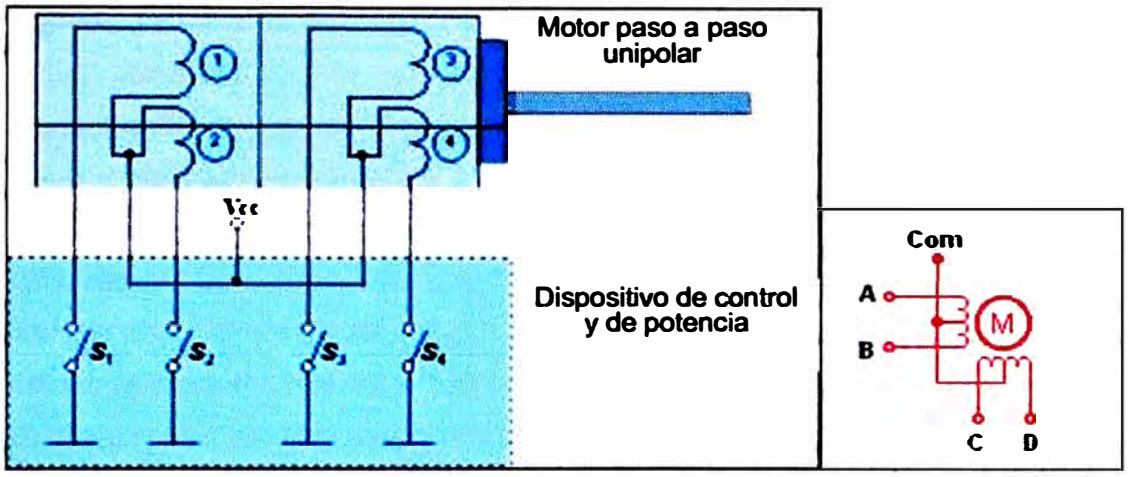


Figura 2.4 Control de motor Unipolar

La Figura 2.5 muestra algunas formas de conexión incluyendo al motor bipolar.

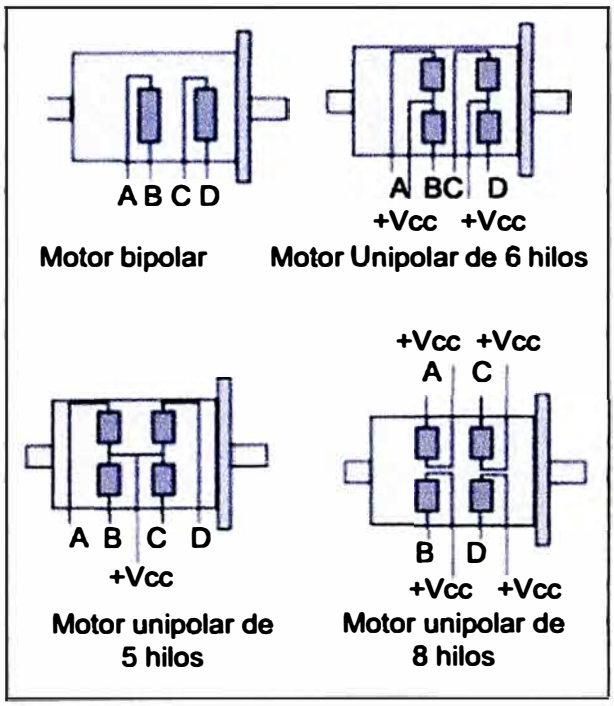


Figura 2.5 Formas de conexión

2.1.4 Parámetros

Se refiere a las principales características y parámetros que se definen sobre un motor paso a paso:

a. Par dinámico de trabajo (Working Torque)

Depende de sus características dinámicas y es el máximo momento angular que el motor es capaz de desarrollar sin perder paso, es decir, sin dejar de responder a algún impulso de excitación del estator y dependiendo, evidentemente, de la carga.

El fabricante proporciona curvas denominadas de arranque sin error (pull-in) y que

relaciona el par en función el número de pasos.

Se debe tener en cuenta que, cuando la velocidad de giro del motor aumenta, se produce un aumento de la f.c.e.m. (fuerza contraelectromotriz) en él generada y, por tanto, una disminución de la corriente absorbida por los bobinados del estator, como consecuencia de todo ello, disminuye el par motor.

b. Par de mantenimiento (Holding Torque)

Es el par requerido para desviar, en régimen de excitación, un paso el rotor cuando la posición anterior es estable; es mayor que el par dinámico y actúa como freno para mantener el rotor en una posición estable dada

c. Para de detención (Detention Torque)

Es una par de freno que siendo propio de los motores de imán permanente, es debida a la acción del rotor cuando los devanados del estator están desactivados.

d. Angulo de paso (Step angle)

Se define como el avance angular que se produce en el motor por cada impulso de excitación. Se mide en grados, siendo los pasos estándar más importantes los mostrados en la Tabla 2.1:

Tabla 2.1 Step Angle

Grados por impulso de excitación	Nº de pasos por vuelta
0,72º	500
1,8º	200
3,75º	96
7,5º	48
15º	24

e. Número de pasos por vuelta

Es la cantidad de pasos que ha de efectuar el rotor para realizar una revolución completa; evidentemente es: NP= 360/ α; donde NP es el número de pasos y α el ángulo de paso.

f. Frecuencia de paso máximo (Maximum pull-in/out)

Se define como el máximo número de pasos por segundo que puede recibir el motor funcionando adecuadamente.

g. Momento de inercia del rotor

Es su momento de inercia asociado que se expresa en gramos por centímetro cuadrado.

h. Par de mantenimiento, de detención y dinámico

Definidos anteriormente y expresados en miliNewton por metro.

2.1.5 Control

Para realizar el control de los motores paso a paso, se debe generar una secuencia

determinada de impulsos. También es necesario que estos impulsos sean capaces de entregar la corriente necesaria para que las bobinas del motor se exciten.

La Figura 2.6 muestra el diagrama de bloques de un sistema con motores paso a paso.

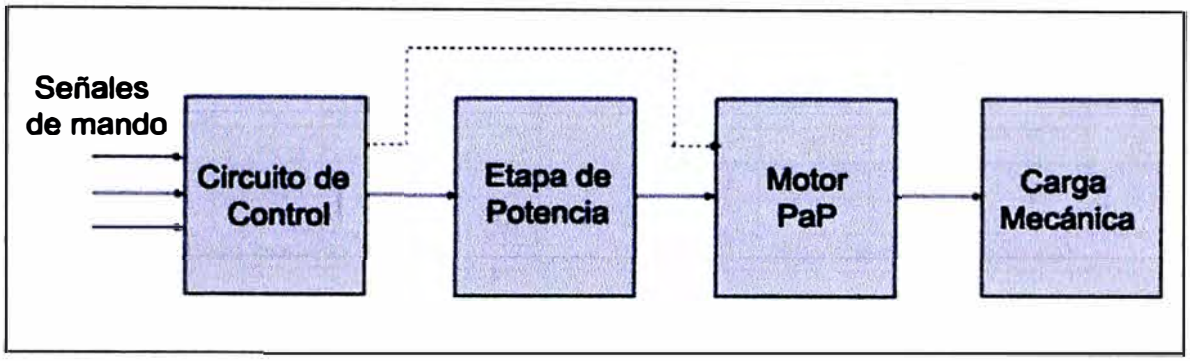


Figura 2.6 Diagrama de bloques de un sistema con motor paso a paso

Existen dos formas básicas de hacer funcionar los motores PaP, atendiendo al avance del rotor bajo cada impulso de excitación.

a. Paso completo (full step)

El rotor avanza un paso completo por cada pulso de excitación y para ello su secuencia ha de ser la correspondiente a la Tabla 2.2 para ambos sentidos de giro, las X indican los interruptores que deben estar cerrados (interruptores en ON), mientras que la ausencia de X indica interruptor abierto (interruptores en OFF).

Tabla 2.2 Secuencia de excitación de un motor paso a paso completo

Paso	S1	S2	S3	S4
1	X			X
2			X	X
3		X	X	
4	X	X		
1	X			X
Sentido horario (a)				

Paso	S1	S2	S3	S4
1	X	X		
2		X	X	
3			X	X
4	X			X
1	X	X		
Sentido antihorario (b)				

b. Medio paso (Half step)

Con este modo de funcionamiento el rotor avanza medio paso por cada pulso de excitación, presentando como principal ventaja una mayor resolución de paso, ya que disminuye el avance angular (la mitad que en el modo de paso completo). Para conseguir tal cometido, el modo de excitación consiste en hacerlo alternativamente sobre dos bobinas y sobre una sola de ellas, según se muestra en la Tabla 2.3 para ambos sentidos de giro.

Al excitar dos bobinas consecutivas del estator simultáneamente, el rotor se alinea con la bisectriz de ambos campos magnéticos; cuando desaparece la excitación de una de ellas, extinguiéndose el campo magnético inducido por dicha bobina, el rotor queda bajo la acción del único campo existente, dando lugar a un desplazamiento mitad.

Tabla 2.3 Secuencia de excitación de un motor Paso a Paso en medio paso

Paso	Excitación de Bobinas			
	S1	S2	S3	S4
1	X			X
2				X
3			X	X
4			X	
5		X	X	
6		X		
7	X	X		
8	X			
1	X			X
Sentido horario (a)				

Paso	Excitación de Bobinas			
	S1	S2	S3	S4
1	X	X		
2		X		
3		X	X	
4			X	
5			X	X
6				X
7	X			X
8	X			
1	X	X		
Sentido antihorario (b)				

Para la explicación, se usa la secuencia b (sentido antihorario) presentada en la Tabla 2.3. En el primer paso están excitadas las bobinas L1 y L2 mediante la acción de S1 y S2, el rotor así se sitúa en la posición indicada en la Figura 2 7.

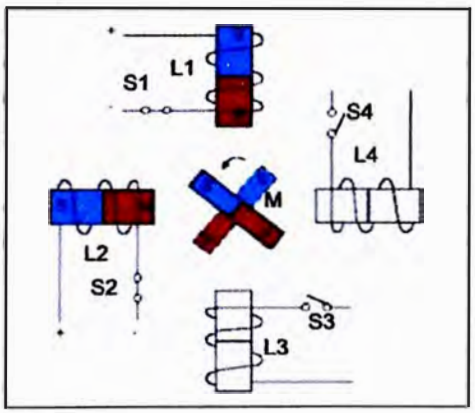


Figura 2.7 Paso 1

En el segundo paso, S1 se abre, con lo que solamente permanece excitada L2 y el rotor gira hasta alinear su polo sur con el norte generado por L2. Supuesto que este motor tenía un paso de 90 grados, en este caso sólo ha avanzado 45 grados.

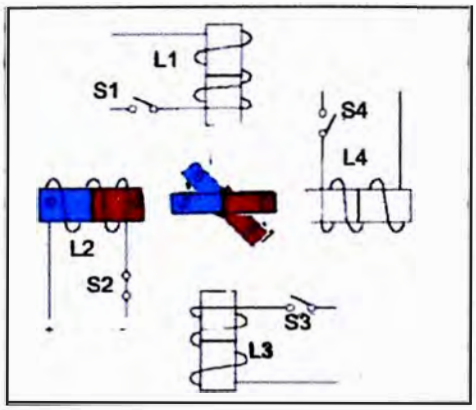


Figura 2.8 Paso 2

En el paso 3, se cierra S3, situación representada en la Figura 2 8, con lo que el rotor ha vuelto a avanzar otros 45 grados. En definitiva, los desplazamientos, siguiendo dicha secuencia, son de medio paso.

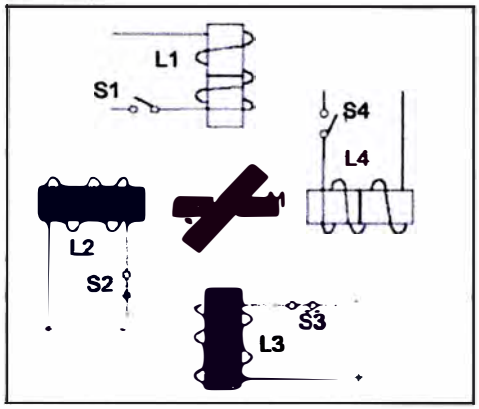


Figura 2.9 Paso 3

La forma de conseguir estas secuencias puede ser a través de un circuito lógico secuencial, con circuitos especializados o con un microcontrolador.

Debido a que el microcontrolador no es capaz de generar la corriente suficiente para excitar las bobinas del motor paso a paso, se ve la necesidad de adaptar un circuito de potencia o algún integrado (L293 Driver Push-Pull cuatro canales).

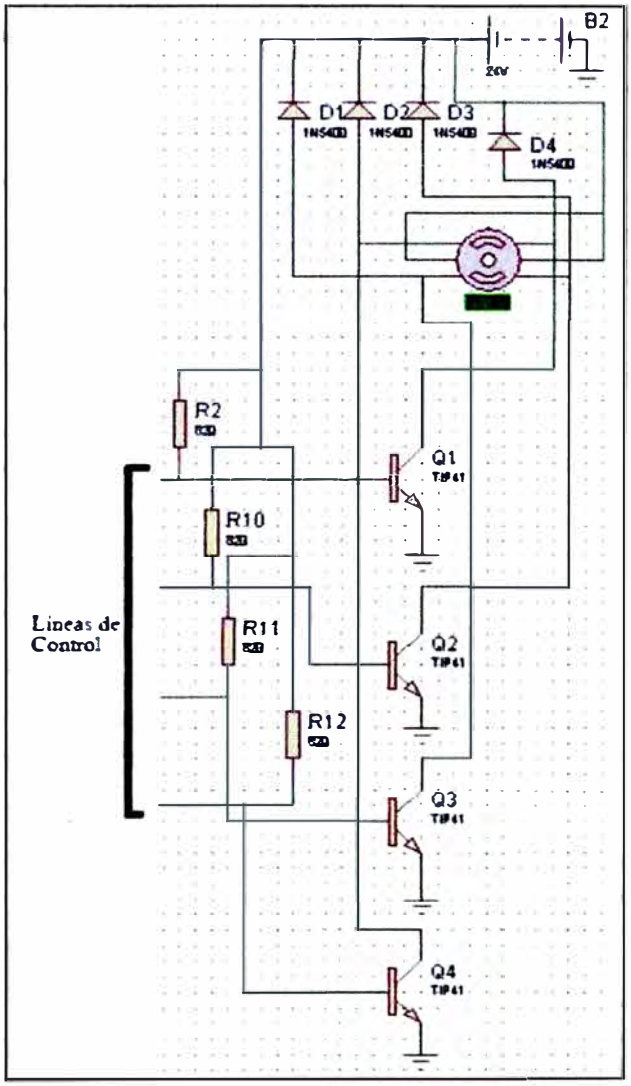


Figura 2.10 Circuito de potencia

La Figura 2.10 muestra un diseño realizado para el circuito de potencia, desarrollado en el software de simulación Proteus. También se observa el motor paso a paso con sus dos terminales comunes conectados a una fuente de 24V, y los cuatro pines correspondientes a los devanados internos del motor, conectados cada uno al colector de un transistor NPN TIP 41. Las resistencias R2, R10, R11 y R12. El ejemplo muestra a los transistores NPN TIP41 de silicio, los cuales soportan corrientes hasta de 2A, y son ideales para aplicaciones de conmutación

2.2 Microcontrolador

El Microcontrolador es un circuito integrado programable que contiene todos los elementos necesarios para controlar un sistema. Hoy en día es común encontrar microcontroladores en infinitas aplicaciones, cuyo único límite es la imaginación. La posibilidad de manejar señales de entrada y salida, así como su capacidad para procesar datos y tomar decisiones, convierten al microcontrolador en uno de los elementos más versátiles que existen actualmente.

Se considera que un microprocesador es un sistema abierto debido a que su configuración es variable de acuerdo con la aplicación a la que se destine, mientras que el microcontrolador es un sistema cerrado; todas las partes del computador están contenidas en su interior y sólo salen al exterior las líneas que controlan los periféricos

Cada fabricante de microcontroladores ofrece un gran número de distintos modelos, desde los más sencillos hasta los más potentes. Es factible seleccionar la capacidad de las memorias, el número de líneas de E/S, la cantidad y potencia de los elementos complementarios, la velocidad de funcionamiento, etc. Por todo ello, la selección del microcontrolador a utilizar. Es un aspecto muy destacado del diseño.

2.2.1 Ventajas y desventajas

Las ventajas del microcontrolador sobre el microprocesador se pueden resumir en lo siguiente:

- El circuito impreso es mucho más pequeño ya que muchos componentes se encuentran dentro el circuito integrado.
- El costo de sistema total es mucho menor, al reducir el número de componentes.
- El consumo de potencia total es mucho menor (Stand By).
- Los problemas de ruido que pueden afectar el bus de comunicación externo de los sistemas con microprocesador se eliminan, debido a que todo el sistema principal se encuentra en un solo encapsulado.
- El tiempo de desarrollo de un sistema se reduce notablemente.
- Sistema robusto: porque al estar todo el sistema en su solo integrado puede funcionar en ambientes de alta temperatura, vibración, etc.

Por otra parte las ventajas del microprocesador sobre el microcontrolador son:

- Mayor velocidad de ejecución y procesamiento. (GHz)
- Se pueden implementar programas de mayor complejidad de procesamiento (MATLAB).
- Las aplicaciones tienen una mejor HMI (interfaz gráfica hombre máquina, sistemas SCADA)
- Mayor memoria RAM (GBytes)
- Mayor memoria EEPROM (disco duro GBytes)
- Facilidad de programación y edición de código fuente (Lenguajes gráficos y de alto nivel)

2.2.2 Aplicaciones y tipos

Algunos fabricantes de microcontroladores exceden el millón de unidades, de un modelo determinado, producidas por semana. Esto proporciona una idea del enorme uso de estos componentes. Los microcontroladores están siendo empleados en infinidad de sistemas presentes en la vida diaria (juguetes, horno microondas, frigoríficos, televisores, computadoras, impresoras, módems, el sistema de arranque de nuestro coche, etc.) y otras aplicaciones tal cómo instrumentación electrónica.

Una aplicación típica podría usar varios microcontroladores para controlar individualmente cada parte del sistema. Estos pequeños controladores podrían comunicarse entre si y también con un procesador central, mucho más potente, para compartir la información y realizar la coordinación de tareas.

Existen muchos tipos de microcontroladores. La clasificación más importante es por el número de bits (4, 8, 16 ó 32 bits). Aunque las prestaciones de los microcontroladores de 16 y 32 bits son superiores a los de 4 y 8 bits, la verdad es que los microcontroladores de 8 bits son de mayor dominio en el mercado y los de 4 bits aun continúan utilizándose. La razón de esta situación es que los microcontroladores de 4 y 8 bits son apropiados para la enorme mayoría de las aplicaciones, lo que hace inadecuada la utilización de microcontroladores más potentes y por ende más caros.

Uno de los sectores que más utiliza los microcontroladores es el mercado automovilístico y militar. Algunas familias de microcontroladores actuales se desarrollaron pensando en estos sectores, siendo modificadas posteriormente para adaptarse a sistemas más genéricos. Los microprocesadores de los sectores mencionados deben soportar condiciones extremas de vibraciones, choques, ruido, etc. y seguir siendo fiables ya que el fallo de cualquier componente puede ser el origen de un accidente.

La totalidad de los microcontroladores actuales se fabrican con tecnología CMOS (Complementary Metal Oxide Semiconductor). Esta tecnología supera a las técnicas

anteriores por su bajo consumo y alta inmunidad al ruido. Pese a estar diseñados con tecnología CMOS, los µC (microcontroladores) no son vulnerables a las descargas electrostásticas (ESD) pues sus terminales están protegidos por diodos zener.

2.2.3 Parámetros de selección

Al seleccionar un microcontrolador para un diseño concreto hay que tener en cuenta diversos factores, tal cómo la documentación y herramientas de desarrollo disponibles además de su precio, y las características del microcontrolador (tipo de memoria de programa, número de temporizadores, interrupciones, etc.):

- **Costo**.- Los fabricantes de microcontroladores compiten duramente para vender sus productos. Si el fabricante desea reducir costes debe tener en cuenta las herramientas de apoyo con que va a contar: emuladores, simuladores, ensambladores, compiladores, etc. Es común que los usuarios se oriente a usar siempre la misma familia de microcontroladores al reducirse la curva de aprendizaje.
- **Aplicación**.- Para la selección de un microcontrolador es indispensable analizar los requisitos de la aplicación.
- **Procesamiento de Datos**.- Si fuera necesario que el microcontrolador realice cálculos críticos en un tiempo limitado, se debe seleccionar un dispositivo suficientemente rápido. También se debe tener en cuenta la precisión de los datos a manejar. Si no es suficiente un microcontrolador de 8 bits, se debe optar por microcontroladores de 16 ó 32 bits, o incluso un hardware de coma flotante. La alternativa más económica y suficiente es usar librerías para manejar los datos de alta precisión.
- **Entrada/Salida**.- para determinar las necesidades de Entrada/Salida del sistema es conveniente dibujar un esquema de bloques del mismo, para que así sea sencillo identificar la cantidad y tipo de señales a controlar. Luego de este análisis puede ser necesario añadir periféricos hardware externos o cambiar a otro microcontrolador más adecuado a ese sistema.
- **Consumo**.- algunos productos que incorporan µC están alimentados con baterías y su funcionamiento puede ser tan vital como activar una alarma. Lo más conveniente en un caso como éste puede ser el uso de un microcontrolador que permanezca en estado de bajo consumo pero que despierte ante la activación de una señal (una interrupción) y ejecute el programa adecuado para procesarla.
- **Memoria**.- para detectar las necesidades de memoria de la aplicación se debe analizar la memoria volátil (RAM), la memoria no volátil (ROM, EEPROM, etc.) y la memoria no volátil modificable (EEPROM). Este último tipo de memoria puede ser útil para incluir información específica de la aplicación como un número de serie o parámetros de calibración. Para determinar la cantidad de memoria necesaria es recomendable realizar

una versión preliminar (en pseudo-código) de la aplicación y a partir de ella hacer una estimación de cuánta memoria volátil y no volátil es necesaria y si es conveniente disponer de memoria no volátil modificable.

- **Ancho de palabra comando o instrucción (Wide Instructions)**: el criterio de diseño debe ser la selección de un microcontrolador de menor ancho de palabra pero que satisfaga los requerimientos de la aplicación. Usar un microcontrolador de 4 bits supone una reducción en los costos, mientras que uno de 8 bits puede ser el más adecuado si el ancho de los datos es de un byte. Los microcontroladores de 16 y 32 bits, debido a su relativo alto costo deben reservarse para aplicaciones que requieran sus altas prestaciones (Entrada/Salida potente o espacio de direccionamiento muy elevado).
- **Diseño de la placa**: la selección de un microcontrolador concreto condiciona el diseño de la placa de circuitos. Se debe tener en cuenta que si se usa un microcontrolador de gama baja, y se necesita una conversión análogo/digital, se deba comprar un dispositivo aparte, encareciendo el diseño.

2.2.4 Ventaja de los microcontroladores Microchip

Los PIC son una familia de microcontroladores tipo RISC fabricados por Microchip Technology Inc. y derivados del PIC1650, originalmente desarrollado por la división de microelectrónica de General Instrument.

La elección de una versión adecuada de PIC es la mejor solución; sin embargo, dado su carácter general, otras familias de microcontroladores son más eficaces en aplicaciones específicas, especialmente si en ellas predomina una característica concreta, que puede estar muy desarrollada en otra familia. Los detalles más importantes que vuelven atraen a los profesionales de la microelectrónica y microinformática y las razones de la excelente acogida que tienen los PIC son los siguientes:

- Sencillez de manejo: Tienen un juego de instrucciones reducido; 33 gama baja, 35 en la gama media, 75 gama alta.
- Buena información, fácil de conseguir y económica.
- Precio: Su coste es comparativamente inferior al de sus competidores.
- Poseen una elevada velocidad de funcionamiento. Buen promedio de parámetros: velocidad, consumo, tamaño, alimentación, código compacto, etc.
- Existe una gran variedad de herramientas hardware que permiten grabar, depurar, borrar y comprobar el comportamiento de los PIC.
- Diseño rápido.
- Herramientas de desarrollo fáciles y baratas. Muchas herramientas software se pueden recoger libremente a través de Internet desde Microchip (http://www.microchip.com).
- La gran variedad de modelos de PIC permite elegir el que mejor responde a los

requerimientos de la aplicación.

- Una de las razones del éxito de los PIC se basa en su utilización. Cuando se aprende a manejar uno de ellos, conociendo su arquitectura y su repertorio de instrucciones, es muy fácil emplear otro modelo.
- Diversidad de modelos de microcontroladores con prestaciones y recursos diferentes. La gran variedad de modelos de microcontroladores PIC permite que el usuario pueda seleccionar el más conveniente para su proyecto.
- Herramientas de soporte potentes y económicas. La empresa Microchip y otras que utilizan los PIC ponen a disposición de los usuarios numerosas herramientas para desarrollar hardware y software. Son muy abundantes los programadores, los simuladores software, los emuladores en tiempo real, Ensambladores, Compiladores C, Intérpretes y Compiladores BASIC, etc.
- La arquitectura Harvard y la técnica de segmentación son los principales recursos en los que se apoya el elevado rendimiento que caracteriza estos dispositivos programables, mejorando dos características esenciales:1) Velocidad de ejecución y 2) Eficiencia en la compactación del código.
- Líneas de E/S de alta corriente. Las líneas de E/S de los PIC pueden proporcionar o absorber una corriente de salida de hasta 25 mA, capaz de excitar directamente ciertos periféricos como LED o microrelés tipo REED.

2.2.5 Recursos comunes

Al estar todos los microcontroladores integrados en un chip, su estructura fundamental y sus características básicas son muy parecidas. Todos deben disponer de los bloques esenciales Procesador, memoria de datos y de instrucciones, líneas de E/S, oscilador de reloj y módulos controladores de periféricos.

Sin embargo, cada fabricante intenta enfatizar los recursos más idóneos para las aplicaciones a las que se destinan preferentemente. Los recursos que se hallan en todos los microcontroladores son:

a. Arquitectura básica

En la actualidad se impone la arquitectura Harvard frente a la Von Neumann. La arquitectura de von Neumann se caracteriza por disponer de una sola memoria principal donde se almacenan datos e instrucciones de forma indistinta. A dicha memoria se accede a través de un sistema de buses único (direcciones, datos y control).

La arquitectura Harvard dispone de dos memorias independientes una, que contiene sólo instrucciones y otra, sólo datos. Ambas disponen de sus respectivos sistemas de buses de acceso y es posible realizar operaciones de acceso (lectura o escritura) simultáneamente en ambas memorias.

b. El CPU

Es el elemento más importante del microcontrolador y determina sus principales características, tanto a nivel hardware como software. Se encarga de direccionar la memoria de instrucciones, recibir el código OP de la instrucción en curso, su decodificación y la ejecución de la operación que implica la instrucción, así como la búsqueda de los operandos y el almacenamiento del resultado.

c. Puertos de Entrada / Salida (Input / Output)

La principal utilidad de los pines que posee la cápsula que contiene un microcontrolador es soportar las líneas de E/S que comunican al computador interno con los periféricos exteriores. Según los controladores de periféricos que posea cada modelo de microcontrolador, las líneas de E/S se destinan a proporcionar el soporte a las señales de entrada, salida y control. Se conocen como puerto A,B,C,D,E.

d. Reloj principal

Todos los microcontroladores disponen de un circuito oscilador que genera una onda cuadrada de alta frecuencia, que configura los impulsos de reloj usados en la sincronización de todas las operaciones del sistema. Generalmente, el circuito de reloj está incorporado en el microcontrolador y sólo se necesitan unos pocos componentes exteriores para seleccionar y estabilizar la frecuencia de trabajo. Dichos componentes suelen consistir en un cristal de cuarzo junto a elementos pasivos o bien un resonador cerámico o una red R-C.

Aumentar la frecuencia de reloj supone disminuir el tiempo en que se ejecutan las instrucciones pero también implica un incremento del consumo de energía. Algunos µC (18F2550 por ejemplo) disponen de oscilador interno programable desde 31KHz hasta 8 MHz, es decir, no se requiere comprar cristal, la desventaja es que no suelen ser tan preciso como un cristal externo.

e. Memoria

En los microcontroladores la memoria de instrucciones o programa y la memoria de datos RAM está integrada en el propio chip. Una parte debe ser no volátil, tipo PROM ó EEPROM, y se destina a contener el programa de instrucciones que gobierna la aplicación. Otra parte de memoria será tipo RAM, volátil, y se destina a guardar las variables y los datos. Hay dos peculiaridades que diferencian a los microcontroladores de los computadores personales: 1) No existen sistemas de almacenamiento masivo como disco duro o disquetes; 2) Como el microcontrolador sólo se destina a una tarea en la memoria de programa, sólo hay que almacenar un único programa de trabajo (no es multitarea como windows, linux u otro sistema operativo).

Los tipos de memoria disponibles para los microcontroladores se describen a

continuación (ROM, EEPROM FLASH de programa, EEPROM de dato, memoria de dato RAM-S para usuario, memoria de dato RAM-S, memoria de pila RAM-S)

- Memoria ROM: (memoria de solo lectura) Esta memoria no es programada por el usuario, viene lista de fábrica y contiene la identificación (ID) o referencia del uC, los datos e instrucciones básicas del microcontrolador, podría compararse con el BIOS SETUP de una computadora personal (PC).

- Memoria de programa EEPROM FLASH: (memoria de instrucciones) En esta memoria se cargan las instrucciones o renglones de todo el programa. La Memoria de programa de la gama alta puede llegar hasta 32KBytes=16KWord, considerando que cada instrucción es de 16 bit (2 bytes) sólo se dispone de 16384 renglones de programa. Es decir, cada instrucción (de las 75 disponibles) consume 2 bytes, excepto GOTO, CALL, MOVFF, LFSR que consumen 4 bytes.

La memoria Flash EEPROM es una mejora con respecto a su predecesora EEPROM, representa mayor velocidad en los ciclos lectura/escritura, es decir, menor tiempo de acceso para leer o escribir un dato. Soporta hasta 100.000 ciclos de erase/write (borrado escritura) , es decir, se pueden cargar hasta 100.000 programas diferentes mediante el hardware cargador de código. La familia 18FXXXX dispone máximo DE (32768 direcciones x 8 bit de datos) es decir: (32KByte = 1KWORD = 512DWORD)

- MEMORIA DE DATO EEPROM: Cuando se guarda un dato en una memoria RAM-S (Memory Access Random Static) y se retira la alimentación del circuito este dato se pierde automáticamente, esto puede representar un serio problema en diversas aplicaciones, por ejemplo al guardar una clave de seguridad. Como solución a este inconveniente, los PIC's disponen de una memoria EEPROM de dato (independiente de la memoria EEPROM FLASH de programa) de 256 bytes (para la familia 16F87X, 18FXXXX, el 16F84 sólo tiene 64 bytes de EEPROM de dato).

La principal desventaja de este tipo de memoria es su relativa baja velocidad (en comparación con la memoria RAM-S), dificultad a nivel de programación para leer o escribir un dato y el número finito de ciclos erase/write (borrado/escritura) que usualmente es 1'000.000 para la familia gama alta de microchip. Las memorias EEPROM de dato del PIC suelen tener un período de retención garantizado por microchip mayor a 40 años. La familia 18FXXXX dispone máximo de (256 direcciones x 8 bit de datos) es decir: (256 bytes)

- Memoria de dato RAM-S para usuario; (registros de propósito general GPR) Son los registros o variables tipo byte (0 to 255) que tiene disponible el µC para el programa del usuario. El PIC 18F452 por ejemplo dispone de 1536 variables tipo byte distribuidas en 6 bancos. El PIC 18F4455 dispone de 2048 variables tipo byte disponibles en 8 bancos. El

PIC 16F877A dispone de 368 variables tipo byte distribuidas en 4 bancos. El PIC 16F84A dispone de 68 variables tipo byte en 1 banco.

El PIC 16F62XA dispone de 224 variables tipo byte en 3 bancos. La principal ventaja de la memoria RAM de datos versus la memoria EEPROM de dato es su alta velocidad y facilidad de acceso en programación. Su principal desventaja es su volatilidad (no conserva los datos después de un reset o el retiro de alimentación del circuito). La familia 18FXX2 dispone máximo de (1536 direcciones x 8 bit de datos en 6 bancos).

- Memoria de dato RAM-S para configuración interna del microcontrolador (registros de funcion especial SFR): Son registros o variables de 8 bit de uso privativo del µC, en estos registros se guarda información del estado (registro STATUS, INTCON, ETC por ejemplo) y funcionamiento integral del µC. El usuario no debería guardar sus datos en estas variables pues alteraría la configuración de alguna función específica. La familia 18F452 dispone máximo de (256 direcciones x 8 bit de datos en 1 banco).
- Memoria de pila RAM-S: (Stack Memory): La pila (Stack) es una zona de memoria RAM independiente de la memoria de datos y de la memoria de programa del µC. Su estructura es del tipo LIFO (Last In First Out) por lo que el último dato que se guarda es el primero que sale. La pila se carga con cada instrucción CALL o con la generación de una interrupción, se descarga con cada instrucción RETURN o RETFIE.

Cuando el diseñador realiza más de 31 llamados de subrutina (CALL) sin regresar (sin uso del RETURN) el puntero de pila (STACK POINTER) se desborda y se presenta el fenómeno denominado OVERFLOW STACK (desbordamiento de pila), lo cual es un error de programación pues el puntero de programa (Pointer Program) salta a un lugar inesperado. Lo deseable en un µC es que disponga de una gran memoria de pila independiente de la memoria de datos. La familia 18F dispone máximo de (31 direcciones x 21 bit de datos) la familia 16F dispone máximo de (8 direcciones x 16 bit de datos)

2.2.6 Recursos especiales

Cada fabricante oferta numerosas versiones de una arquitectura básica de microcontrolador. En algunas amplía las capacidades de las memorias, en otras incorpora nuevos recursos, en otras reduce las prestaciones al mínimo para aplicaciones muy simples, etc. La labor del diseñador es encontrar el modelo mínimo que satisfaga todos los requerimientos de su aplicación. De esta forma, minimizará el coste, el hardware y el software.

Los principales recursos específicos que incorporan los microcontroladores son:

a. Temporizadores o "timers":

Se emplean para controlar periodos de tiempo (temporizadores) y para llevar la cuenta de acontecimientos que suceden en el exterior (contadores). Para la medida de

tiempos se carga un registro con el valor adecuado y a continuación dicho valor se va incrementando o decrementando al ritmo de los impulsos de reloj o algún múltiplo hasta que se desborde y llegue a 0, momento en el que se produce un aviso.

Cuando se desean contar acontecimientos que se materializan por cambios de nivel o flancos en alguna de los pines del microcontrolador, el mencionado registro se va incrementando o decrementando al ritmo de dichos impulsos.

b. Perro guardián "watchdog timer" wdt

Cuando el computador personal se bloquea por un fallo del software u otra causa, se pulsa el botón del reset y se reinicia el sistema. Pero un microcontrolador funciona sin el control de un supervisor y de forma continua las 24 horas del día.

El Perro guardián consiste en un temporizador (2 ms hasta 131 seg dependiendo del PIC) que, cuando se desborda y pasa por 0, provoca un reset automáticamente en el sistema. Se debe diseñar el programa que controla la tarea de forma que refresque o inicialice al Perro guardián antes de que provoque el reset. Si falla el programa o se bloquea, no se refrescará al Perro guardián y, al completar su temporización, provocará el reset.

c. Estado de reposo o bajo consumo sleep, stand by o power saving

Son abundantes las situaciones reales de trabajo en que el microcontrolador debe esperar, sin hacer nada a que se produzca algún acontecimiento externo que le active de nuevo en funcionamiento (interrupción).

Para ahorrar energía, (factor clave en los aparatos portátiles), los microcontroladores disponen de una instrucción especial (SLEEP en los PIC), que les pasa al estado de reposo o de bajo consumo en el cual los requerimientos de potencia son mínimos.

En dicho estado se detiene el reloj u oscilador principal y se "congelan" sus circuitos asociados, quedando sumido en un profundo "sueño" el microcontrolador. Al activarse una interrupción ocasionada por el acontecimiento esperado, el microcontrolador se despierta y reanuda su trabajo.

En este estado el uC consume menos de 200nA y sólo despertará (wake up) mediante una interrupción externa. El modo sleep es muy necesario en aplicaciones que se alimentarán a partir de baterías

d. Protección ante Fallo de alimentación o "Brown Out Reset" BOR

Se trata de un circuito que resetea al microcontrolador cuando el voltaje de alimentación (VDD) es inferior a un voltaje mínimo ("brown out").

Mientras el voltaje de alimentación sea inferior al de brown out el dispositivo se mantiene reseteado, comenzando a funcionar normalmente cuando sobrepasa dicho valor. Útil Para aplicaciones que funcionarán con baterías, cuando la batería esté baja de

carga el PIC no funcionará por protección.

e. Reset de encendido ("Power on Reset") POR

Todos los PIC tienen la facultad de generar una autoreinicialización o autoreset al conectarles la alimentación.

f. Temporizacion de encendido ("Power up Timer") PWRT

Es una opción para que el µC espere un tiempo (aproximadamente 75mS) antes de empezar a ejecutar instrucciones después de alimentar el PIC. Es útil para evitar funcionamientos erráticos del µC por ruido o rebotes al conectar la alimentación.

g. Código de protección (Code Protect) CP

Cuando se procede a realizar la grabación del programa mediante el hardware cargador (loadder code), el PIC puede protegerse para evitar su lectura. También disponen los PIC de posiciones reservadas para registrar números de serie, códigos de identificación, prueba, etc.

h. interrupción mediante pines externos

Los pines de interrupción externa constituyen uno de los aspectos más importantes en un microcontrolador pues ofrece la posibilidad de interactuar de una manera óptima con periféricos externos de entrada tales como: teclados de computadora, receptores IR, RF, USB, teclados, pulsadores, etc. Los pines de interrupción externa más comunes son RB0, RB1, RB2, RB4, RB5, RB6, RB7 para gama alta.

i. Prioridad de interrupciones

La gama alta permite programar prioridad LOW OR HIGH a cada interrupción dependiendo de la importancia y urgencia en el circuito y/o aplicación.

j. Interrupciones vectorizadas:

Algunos µC poseen una dirección única (vector de interrupción independiente) para cada evento de interrupción, esto optimiza el tiempo en detección y facilita el manejo de cada interrupción. Los µC microchip no disponen de esta opción, sin embargo, algunos µC motorola si permiten dicha función.

k. Pull up / pull down interno

Los µC PIC tiene Pull UP interno en el puerto B (B0 a B7), no disponen de pull down. La resistencia interna de Pull UP es alrededor de 3KΩ. El puerto B del PIC es el único que tiene la opción de Pull UP interno, es decir, por software se puede garantizar que el PIC interpretará como 1 lógico todas los pines del puerto B configurados como entrada y que estén al aire (input float).

Esto es práctico cuando se debe conectar pulsadores o teclados matriciales a un µC y no se desea comprar resistores externos de PULL UP. Los microcontroladores son fabricados con tecnología CMOS (Complementary Metal Oxide Semiconductor) lo cual

implica que tienen alta impedancia de entrada (sus entradas requieren muy baja corriente, en el orden de µA), si una entrada se deja al aire (input float) el µC lo interpretará como ruido (debido al ruido electromagnético EMI presente en el ambiente o efecto antena y se puede inducir una tensión fantasma en los pines de entrada que estén flotando).

l. Pull up/ pull down externo

El pull up es una resistencia externa que se conecta de una entrada del microcontrolador hacia +VCC. Un pull down es lo mismo pero conectada hacia tierra. La función de estas resistencias (del orden 1KΩ a 100KΩ) es garantizarle un estado a los pines al aire o input float del PIC configurados como entrada.

m. Capacidad de corriente: modo sink (iol), modo source (ioh)

Para la gama media y alta de microchip, la corriente máxima de salida en modo sink (sumidero) o "cero lógico" es de 25 mA y la corriente máxima de salida en modo source (fuente) o "uno lógico" es de 25 mA. Este aspecto es de singular importancia pues indica la potencia que puede transmitir el PIC a los periféricos de salida tales como Relés, Led, Motores, etc.

El abanico de entrada y salida (Fan In / Fan Out) de un Circuito integrado está relacionado con las impedancias de entrada y salida del mismo. Las corrientes en un PIC alimentado a 5V son:

- IOL = 25mA (modo sink o sumidero) corriente de salida en 0 lógico
- IOH = 25mA (modo source o fuente) corriente de salida en 1 lógico
- IIL = 1µA (Corriente de entrada en cero lógico)
- IIH = 1µA (Corriente de entrada en uno lógico)

Considerando que un led (diodo emisor de luz) requiere para encender una tensión de 2V mínimo y una corriente entre 5 a 50 mA, se puede afirmar que un PIC puede encender directamente hasta 5 led por un solo terminal. Obviamente si la carga requiere más corriente (un relé convencional exige 30mA aproximadamente) se debe conectar un buffer (impulsador de corriente) o un transistor en emisor o colector común.

Por ser un dispositivo construido con tecnología CMOS (Complementary Metal Oxide semiconductor) los PIC presentan una alta impedancia de entrada, esto implica que la corriente de entrada por cada pin está en el orden de los microamperios, es decir, se puede aplicar a un µC la salida directa de cualquier sensor sin necesidad de una etapa previa de acondicionamiento de corriente (amplificador seguidor de voltaje o buffer amplificador de corriente).

n. Conversor A/D (CAD)

Los microcontroladores que incorporan un Conversor A/D (Analógico/Digital) pueden

procesar señales analógicas, tan abundantes en las aplicaciones. Suelen disponer de un multiplexor que permite aplicar a la entrada del CAD diversas señales analógicas desde los pines del circuito integrado. La familia 16F87X y 18FXX2 disponen de un conversor A/D de 8 canales a 10 bit de resolución y la familia 18FXXXX disponen de un conversor A/D de 13 canales a 10 bit de resolución.

ñ. Conversor D/A (DAC):

Transforma los datos digitales obtenidos del procesamiento del computador en su correspondiente señal analógica que saca al exterior por una de los pines de la cápsula. No todos los µC traen esta opción en su interior. Los PIC's gama enana, baja, media y alta no disponen de esta opción.

o. Comparador analógico

Algunos modelos de microcontroladores (16F62X) disponen internamente de un Amplificador Operacional que actúa como comparador entre una señal fija de referencia y otra variable que se aplica por una de los pines de la cápsula. La salida del comparador proporciona un nivel lógico 1 ó 0 según una señal sea mayor o menor que la otra. También hay modelos de microcontroladores con un módulo de tensión de referencia que proporciona diversas tensiones de referencia que se pueden aplicar en los comparadores.

p. Puertas de Entrada/Salida digitales

Todos los microcontroladores destinan algunas de sus pines a soportar líneas de E/S digitales. Por lo general, estas líneas se agrupan de ocho en ocho formando Puertas. Las líneas digitales de las Puertas pueden configurarse como Entrada o como Salida cargando un 1 ó un 0 en el bit correspondiente de un registro destinado a su configuración.

q. Puertos de comunicación USART, I C, PARALELO, SPI, USB

Con objeto de dotar al microcontrolador de la posibilidad de comunicarse con otros dispositivos externos, otros buses de microprocesadores, buses de sistemas, buses de redes y poder adaptarlos con otros elementos bajo otras normas y protocolos. Algunos modelos disponen de recursos que permiten directamente esta tarea, entre los que destacan:

- UART (Universal Asyncronous Reception Transmision).
- USART (Universal Syncronous Asyncronous Reception Transmision)
- PARALLEL PORT:Puerto paralelo esclavo para poder conectarse con los buses de otros microprocesadores.
- USB (Universal Serial Bus), que es un moderno bus serie para los PC de ala velocidad.
- Bus I2C (Interfaz De Circuitos Integrados), que es un interfaz serie de dos hilos

desarrollado por Philips.

- CAN (Controller Area Network), para permitir la adaptación con redes de conexionado multiplexado desarrollado conjuntamente por Bosch e Intel para el cableado de dispositivos en automóviles. En EE.UU. se usa el J185O.
- SPI: Interfaz de Periféricos Serial
- MSSP: Master Synchronous Serial Port

r. PROGRAMMING™ (ICSP™) VIA TWO PINS:

Reprogramación in Circuit, es decir, no es necesario retirar el PIC del circuito, pues los datos se cargan a la memoria de programa mediante el puerto serial UART RS232 Tx y Rx.

s. DEPURACIÓN DEL PROGRAMA IN CIRCUIT: IN-CIRCUIT DEBUG (ICD) VIA TWO PINS

La gama alta permite realizar (mediante un hardware específico) una depuración (debugger= delete bug = eliminar bichos) de un código fuente. Esta depuración se realiza con la PC conectada en tiempo real, el programa se irá ejecutando línea por línea en el editor (MPLAB SIMULATOR) y en el circuito hardware mediante el puerto serial RS232 del PIC y del PC.

t. Hardware Multiplicador con una instrucción (8X8 single-cycle hardware multiplier)

La gama alta incluye una instrucción (MULLW) para multiplicar 1 byte x 1 byte y ofrece la respuesta en 2 bytes.

u. Módulos captura comparación PWM CCP:

Captura una trama de datos serial y luego la compara con registros internos, útil en aplicaciones de sensores ultrasónicos. El módulo PWM (PULSE WIDE MODULATION) son circuitos que proporcionan en su salida pulsos de ciclo útil (tiempo en '1' lógico) variable y periodo constante, que se ofrecen al exterior a través de los pines del encapsulado. Suelen emplearse para el control de velocidad de motores de DC y Servomotores.

v. Level stack (niveles de pila)

La pila (Stack) es una zona de memoria RAM independiente de la memoria de datos y de la memoria de programa del µC. Su estructura es del tipo LIFO (Last In First Out) por lo que el último dato que se guarda es el primero que sale. La pila se carga con cada instrucción CALL o con la generación de una interrupción, se descarga con cada instrucción RETURN o RETFIE. Cuando el diseñador realiza más de 31 llamados de subrutina (CALL) sin regresar (sin uso del RETURN) el puntero de pila (STACK POINTER) se desborda y se presenta el fenómeno denominado OVERFLOW STACK

(desbordamiento de pila), lo cual es un error de programación pues el puntero de programa (Pointer Program) salta a un lugar inesperado. Lo deseable en un µC es que disponga de una gran memoria de pila independiente de la memoria de datos.

w. Memoria EEPROM de dato

Es una memoria relativamente pequeña (64 a 256 bytes) en la cual se pueden guardar los datos del programador y no se pierden incluso al desconectar la alimentación del circuito.

x. Low Voltage Detect (LVD):

Detecta si un voltaje aplicado al uC es menor de un nivel programado. Es útil para detectar si la batería de alimentación está agotada.

y. Selección de oscilador

Dependiendo de la frecuencia natural del cristal, se debe seleccionar el modo de oscilación y los condensadores de estabilización. En gama alta 18FXX2 si se desea una frecuencia superior a 25MHz, se debe activar el modo PLL (phase locked loop) multiplicador de frecuencia del cristal por 4. Por ejemplo, para trabajar a 40MHz (frecuencia máxima de la familia 18FXX2) lo correcto es conectar un cristal de 10MHz con condensadores de 27pF y activar el circuito PLL del µC. La frecuencia máxima de la familia 16F87X es 20MHz y la frecuencia máxima de la familia 18FXXXX es 48MHz. La familia 18FXXXX tiene la opción de trabajar con un oscilador interno configurable desde 31KHz hasta 8MHz.

El oscilador de la familia 18FXXXX se puede configurar de las siguientes maneras:

- **XT** : Crystal/Resonator
- **XTPLL** : crystal/resonator con PLL habilitado
- **HS** : High-Speed Crystal/Resonator
- **HSPLL:** High-Speed Crystal/Resonator con PLL habilitado
- **EC:** Reloj externo con salida FOSC/4
- **ECIO:** Reloj externo con Entrada/salida sobre RA6
- **ECPLL:** Reloj externo con PLL habilitado y salida FOSC/4 sobre RA6
- **ECPIO:** Reloj externo con PLL habilitado, y Entrada/Salida sonre RA6
- **INTHS:** Oscilador interno usado como reloj del microcontrolador, el oscilador HS es usado como reloj del USB.
- **INTXT:** Oscilador interno usado como reloj del microcontrolador, oscilador XT es usado como reloj del USB.
- **INTIO:** Oscilador interno usado como reloj del microcontrolador, oscilador EC usado como reloj del USB, Entradas/salidas digitales sobre RA6
- **INTCKO:** Oscilador interno usado como reloj del microcontrolador, oscilador EC usado

como reloj del USB, salidas FOSC/4 sobre RA6

2.2.7 Gamas de PICs

Un objetivo del diseñador es reducir los costos del circuito electrónico, para ello se debe seleccionar adecuadamente el microcontrolador, que cumpla con suficiencia las necesidades del proyecto.

Las aplicaciones sencillas requieren pocos recursos, las grandes aplicaciones requieren recursos numerosos y de gran potencia. La gama de microcontroladores va desde los sencillos y baratos (aplicaciones simples) hasta los complejos y costosos (aplicaciones complejas).

Microchip ha establecido cuatro familias de microcontroladores para adaptarse a las necesidades de la mayoría de los clientes potenciales. La llamada gama enana es en realidad una subfamilia formada por componentes pertenecientes a las otras gamas. Los PIC enanos son muy solicitados en aplicaciones de control de personal, en sistemas de seguridad y en dispositivos de bajo consumo que gestionan receptores y transmisores de señales.

Su pequeño tamaño los hace ideales en muchos proyectos donde esta cualidad es fundamental.

a. Gama enana

PIC12Cxxx de 8 pines con instrucciones de 12 /14 bit. Se caracteriza por su reducido tamaño, se alimentan con voltajes comprendidos entre 2,5 V y 5,5 V, consumiendo menos de 2 mA cuando trabajan a 5 V y 4 MHz. El formato de sus instrucciones puede ser de 12 o de 14 bits y su repertorio es de 33 o 35 instrucciones, respectivamente. En la Figura 2.11 se muestra el diagrama de conexionado de uno de estos PIC [1].

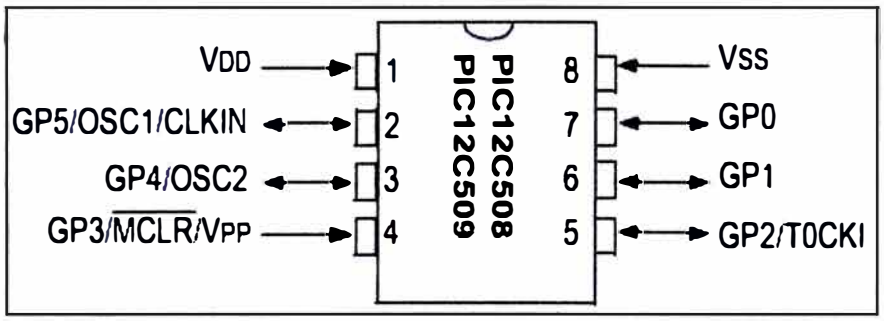


Figura 2.11 PIC12C508, 509

Aunque los PIC enanos sólo tienen 8 pines, pueden destinar hasta 6 como líneas de E/S para los periféricos porque disponen de un oscilador interno R-C.

En la Tabla 3.4 se presentan las diferencias entre los modelos de esta subfamilia. Los modelos 12C5xx pertenecen a la gama baja, siendo el tamaño de las instrucciones de 12 bits; mientras que los 12C6xx son de la gama media y sus instrucciones tienen 14 bits. Los modelos 12F6xx poseen memoria Flash para el programa y EEPROM para los datos.

Tabla 3.4 Diferencias subfamilia 12C5xx

Dispositivo	Rango de Voltaje	EPROM	Data RAM	Oscilador 2 Calibración (Bits)
PIC12C508A	3.0-5.5	512 x 12	25	6
PIC12LC508A	2.5-5.5	512 x 12	25	6
PIC12C508	2.5-5.5	512 x 12	25	4
PIC12C509A	3.0-5.5	1024 x 12	41	6
PIC12LC509A	2.5-5.5	1024 x 12	41	6
PIC12C509	2.5-5.5	1024 x 12	41	4
PIC12CR509A	2.5-5.5	1024 x 12	41	6
PIC12CE518	3.0-5.5	512 x 12	25	6
PIC12LCE518	2.5-5.5	512 x 12	25	6
PIC12CE519	3.0-5.5	1024 x 12	41	6

Nota: Oscilador Interno de 4 MHz RC oscillator con calibración programable

b. Gama baja o básica

PIC16C5X con instrucciones de 12 bits [2]. Sus versiones están encapsuladas con 18 y 28 pines y pueden alimentarse a partir de una tensión de 2,5 V, lo que les hace ideales en las aplicaciones que funcionan con pilas teniendo en cuenta su bajo consumo (menos de 2 mA a 5 V y 4 MHz). Tienen un repertorio de 33 instrucciones cuyo formato consta de 12 bits. No admiten ningún tipo de interrupción y la Pila sólo dispone de dos niveles. La Tabla 3.5 muestra las diferencias de la subfamilia y la Figura una de las presentaciones.

Tabla 3.5 Diferencias subfamilia 12C5xx

Dispositivo	Pines	I/O	EPROM/ROM	RAM
PIC16C54	18	12	512	25
PIC16C54A	18	12	512	25
PIC16C54C	18	12	512	25
PIC16CR54A	18	12	512	25
PIC16CR54C	18	12	512	25
PIC16C55	28	20	512	24
PIC16C55A	28	20	512	24
PIC16C56	18	12	1K	25
PIC16C56A	18	12	1K	25
PIC16CR56A	18	12	1K	25
PIC16C57	28	20	2K	72
PIC16C57C	28	20	2K	72
PIC16CR57C	28	20	2K	72
PIC16C58B	18	12	2K	73

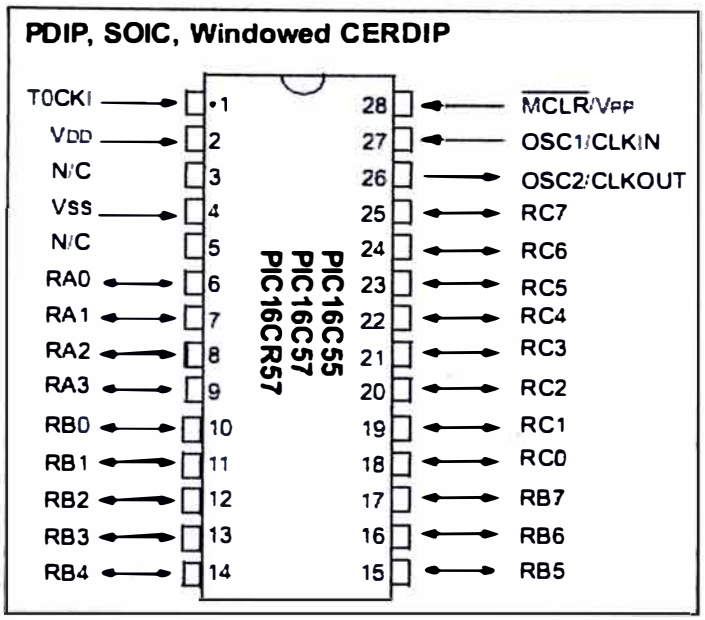


Figura 2.12 PIC16C55, 57, CR57

Nota: La pila sólo dispone de dos niveles lo que supone no poder encadenar más de dos subrutinas. Los microcontroladores de la gama baja no admiten interrupciones.

c. Gama media

PIC16FXXX con instrucciones de 14 bits [3]. Es la gama más variada y completa de los PIC. Abarca modelos con encapsulado desde 18 pines hasta 68, cubriendo varias opciones que integran abundantes periféricos. Dentro de esta gama se halla el «fabuloso PIC16X84» y sus variantes (Figura 2.13).

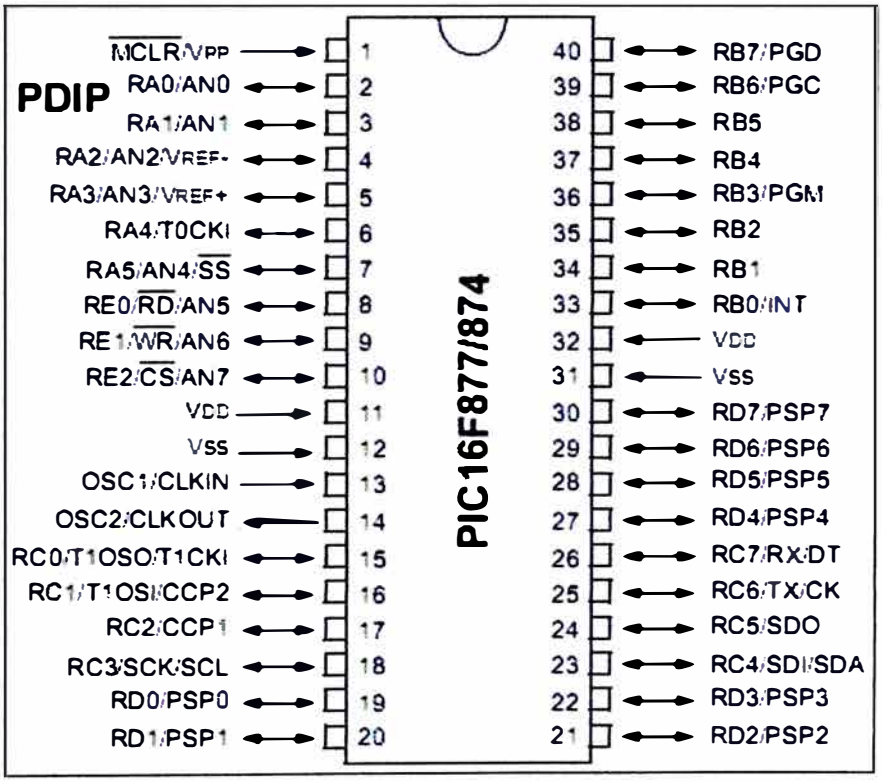


Figura 2.13 PIC16F877, PIC16F874

El 16F877 y 16F84A son uno de los modelos más representativos de la gama media. En esta gama sus componentes añaden nuevas prestaciones a las que poseían los de la gama baja, haciéndoles más adecuados en las aplicaciones complejas. Admiten interrupciones, poseen comparadores de magnitudes analógicas, convertidores A/D, puertos serie y diversos temporizadores. El repertorio de instrucciones es de 35, de 14 bits cada una y compatible con el de la gama baja. Sus distintos modelos contienen todos los recursos que se precisan en las aplicaciones de los microcontroladores de 8 bits. También dispone de interrupciones y una pila de 8 niveles que permite el anidamiento de subrutinas. En la Tabla 3.6 se presentan las principales características de los modelos de esta familia.

Tabla 3.6 Variantes de la familia PIC16F87X

	PIC16F873	**PIC16F874**	**PIC16F876**	**PIC16F877**
Frecuencia de operación	DC - 20 MHZ	DC - 20 MHZ	DC - 20 MHZ	DC - 20 MHZ
RESETS (y Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
FLASH Program Memory (14-bit words)	4K	4K	8K	8K
Data Memory (bytes)	192	192	368	368
EEPROM	128	128	256	256
Interrupciones	13	14	13	14
Puertos E/S	Puertos A, B, C	Puertos A, B, C, D, E	Puertos A, B, C	Puertos A, B, C, D, E
Timers	3	3	3	3
Módulos Capture/Compare/PWM	2	2	2	2
Comunicaciones seriales	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Comunicaciones paralelas	-	PSP	-	PSP
10-bit Analog-to-Digital Module (canales de entrada)	5	8	5	8
Conjunto de instrucciones	35	35	35	35

Encuadrado en la gama media también se halla la versión PIC14C000, que soporta el diseño de controladores inteligentes para cargadores de baterías, pilas pequeñas, fuentes de alimentación ininterrumpibles y cualquier sistema de adquisición y procesamiento de señales que requiera gestión de la energía de alimentación. Los PIC 14C000 admiten cualquier tecnología de las baterías como Li-Ion, NiMH, ect, Ph y Zinc. El temporizador TMR1 que hay en esta gama tiene un circuito oscilador que puede trabajar asincrónicamente y que puede incrementarse aunque el microcontrolador se halle en el modo de reposo ("sleep"), posibilitando la implementación de un reloj en

tiempo real. Las líneas de E/S presentan una carga "pull-up" activada por software.

d. Gama Alta

PIC17CXXX, PIC18FXXXX con instrucciones de 16 bits [4]. Esta gama alcanza hasta las 75 instrucciones de 16 bits de ancho. Sus modelos disponen de un sistema de gestión de interrupciones sectorizadas; esto quiere decir que cada interrupción tiene una dirección única y priorizada muy potente (Figura 2.14 PIC18C4X2, Figura 2.15 PIC18C2X2).

La gama alta también incluyen diversos controladores de periféricos, puerto USB 2.0 de alta y baja velocidad, puertas de comunicación serie y paralelo con elementos externos, un multiplicador hardware de gran velocidad y mayores capacidades de memoria, que alcanza más de 32 KBytes en la memoria de instrucciones y hasta 2048 bytes en la memoria de datos.

La característica más destacable de los componentes de esta gama (17CXXX) es su arquitectura abierta. Esto consiste en la posibilidad de ampliación del microcontrolador con elementos externos.

Para la ampliación los pines pueden ofrecer al exterior las líneas de los buses de datos, direcciones y control, a las que se conectan memorias o controladores de periféricos. Sin embardo esta característica obliga a estos componentes a tener un elevado número de pines comprendido entre 40 y 44. Esta era una filosofía de construcción del sistema que se empleaban en los microprocesadores y no suele ser una práctica habitual cuando se emplean microcontroladores.

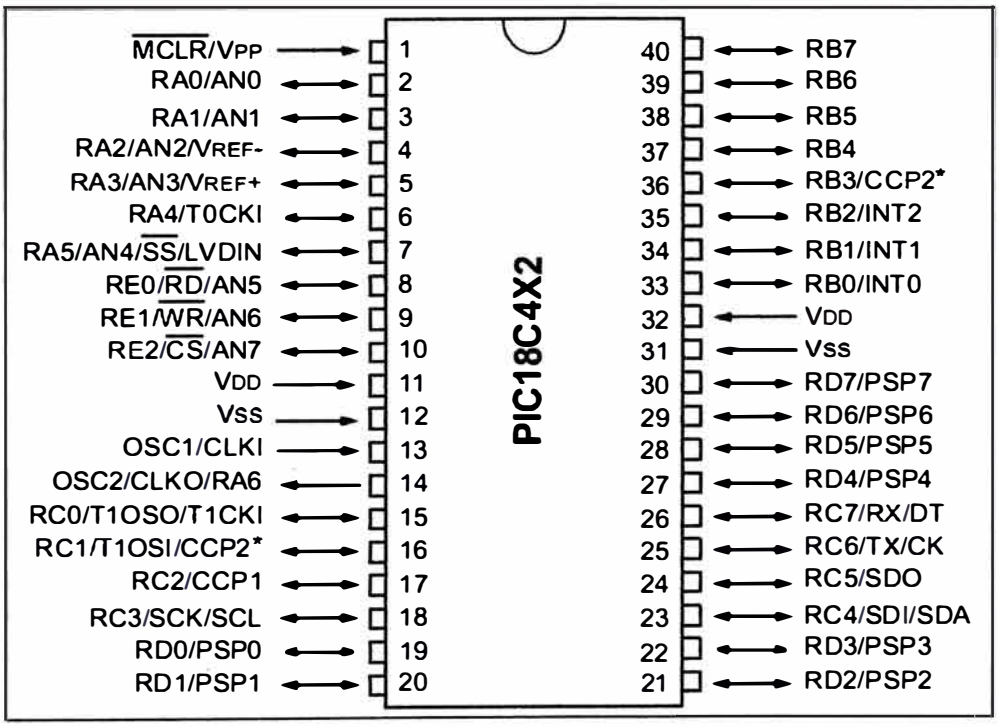


Figura 2.14 PIC18C4X2

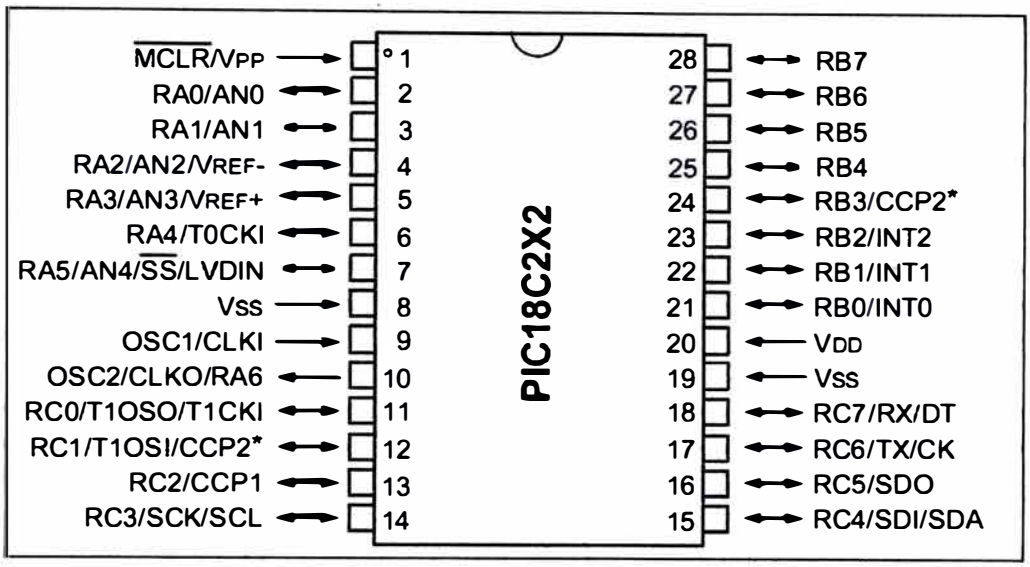


Figura 2.15 PIC18C2X2

En la Tabla 3.7 se presentan las principales características de los modelos de esta familia de microcontroladores del fabricante Microchip.

Tabla 3.7 Variantes de la familia PIC18C4X2

	PIC18C242	**PIC18C252**	**PIC18C442**	**PIC18C452**
Frecuencia de operación	DC - 40 MHZ	DC - 40 MHZ	DC - 40 MHZ	DC - 40 MHZ
RESETS y Delays) POR, BOR, Reset Instruction Stack full, Stack underflow (PWRT, OST)	Si	Si	Si	Si
Program Memory (instructions 16-bit words)	8192	16384	8192	16384
Data Memory (bytes)	512	1536	512	1536
Interrupciones	16	16	17	17
Puertos E/S	Puertos A, B, C	Puertos A, B, C	Puertos A, B, C, D, E	Puertos A, B, C, D, E
Timers	4	4	4	4
Módulos Capture/Compare/PWM	2	2	2	2
Comunicaciones seriales	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Comunicaciones paralelas	-	-	PSP	PSP
10-bit Analog-to-Digital Module (canales de entrada)	5	5	85	8
Conjunto de instrucciones	75	75	75	75

2.3 Conceptos de programación de Microcontroladores

La programación de microcontroladores requiere de conceptos claros y precisos para establecer un comportamiento adecuado acorde con los requerimientos del sistema a desarrollar.

2.3.1 Lenguaje de máquina y lenguaje ensamblador

El lenguaje de máquina es el único lenguaje que entienden los µC es el formado por los ceros (0) y unos (1) del sistema binario. A este lenguaje se le denomina lenguaje de máquina. Los códigos de este lenguaje que forman las instrucciones se llaman códigos de máquina.

Por ejemplo, cuando el µC lee el código de máquina "11111000111010" está recibiendo la instrucción de suma 58 al registro de trabajo W y guarda el resultado en W. La codificación hexadecimal es una manera más comprensible de la codificación binaria, así por ejemplo, el código de máquina "11111000111010" se podría expresar en hexadecimal como 3E3A.

El lenguaje ensamblador (de bajo nivel) es la forma de expresar las instrucciones de una manera más natural al hombre y que, sin embargo, es muy cercana al µC porque cada una de sus instrucciones corresponde con otra en código de máquina que el µC es capaz de interpretar". Utiliza nemónicos (instrucciones o comandos) (33 para la gama baja, 35 para la gama media y 75 para la gama alta) que son grupos de caracteres alfanuméricos que simbolizan las órdenes o tareas a realizar con cada instrucción.

Por ejemplo, para ordenar al PIC: "suma 58 al registro de trabajo W y guarda el resultado en este mismo registro W", en lenguaje ensamblador para gama media es "ADDLW .58" que es mucho más comprensible para un ser humano que el código de máquina "11111000111010" ó 3E3A en hexadecimal.

La principal ventaja del lenguaje ensamblador no es precisamente su facilidad, por el contrario se han desarrollado lenguajes de alto nivel para obviar el uso de assembler, no obstante, el lenguaje assembler presenta dos importantes ventajas que son:

- Optimización de Código: Si el usuario desarrolla destrezas en este tipo de programación entonces puede generar el código de mínimo tamaño en la memoria de programa, lo cual es muy importante al implementar programas largos y complejos en microcontroladores con escasa memoria de programa como gama baja y media de microchip.
- Velocidad de Ejecución: Como consecuencia de la optimización de código, ejecuta un grupo o set de instrucciones con mayor rapidez.

El lenguaje ensamblador necesita un programa para edición y compilación (MPASM ensamblador de Microchip), el cual es un software que se encarga de traducir los

nemónicos y símbolos alfanuméricos del programa escrito en ensamblador por el usuario a código de máquina comprensible para el µC. El programa escrito en lenguaje ensamblador recibe el nombre de código fuente, archivo fuente o fichero fuente. Suele tener la extensión *.asm.

El archivo fuente debe ser traducido a código de máquina, de lo cual se encarga el programa ensamblador. La mayoría de los ensambladores proporcional a su salida un archivo que suele tener la extensión *.hex (hexadecimal) (Motorola emplea la extensión .s19). El ensamblador más utilizado por los µC microchip PIC es el MPASM que viene con el paquete MPLAB IDE, el cual será descrito en la siguiente sección

2.3.2 Lenguaje de alto nivel, programación estructurada y modular

El lenguaje ensamblador representa dificultad en el momento de la programación por ser un lenguaje más cercano al µC o µP que al usuario humano. Debido a esto se crearon los lenguajes de alto nivel (nivel de los humanos) tales como: Basic, C, Turbo C, C++, PASCAL, FORTRAN, PROLOG, LPM2, etc. (Lenguajes de alto nivel).

El modelo de programación modular y estructurada por macros de assembler, compilado condicional de subrutinas optimizadas y gestor de interrupciones (handler of interruptions) automático de interrupciones es un software libre que persigue proporcionar a programadores de microcontroladores PIC, expertos y/o novatos, herramientas prácticas y económicas que permitirán el desarrollo rápido, fácil, optimizado y funcional de aplicaciones electrónicas.

Con la implementación de potentes macros y subrutinas de assembler, se ofrece una amplia gama de posibilidades a nivel de hardware y software que sin duda corroborará al desarrollo de innumerables proyectos.

El sistema de programación modular está habilitado para los microcontroladores de la gama media y alta de la familia microchip:, tales como PIC18F452, 16F84A, 16F627A, 16F628A.

La programación estructurada es un importante concepto en el diseño de proyectos. Se entiende como la división del programa principal en módulos o procedimientos que realizan una determinada tarea dentro del programa.

La principal ventaja de la programación estructurada y modular por macros versus los lenguajes de programación de alto nivel convencionales (PIC Basic, Microbasic, compilador C, etc) radica en el control automático de interrupciones de periféricos externos e internos.

- Simplifica el tiempo de desarrollo de cada parte del algoritmo por separado, permitiendo concentrar la atención en los detalles de la aplicación.
- Produce códigos más fiables, fáciles de entender, documentar y modificar.

- Control automático de interrupción de alta y baja prioridad lo cual potencia las prestaciones del microcontrolador al poder atender simultáneos procesos y/o periféricos de hardware.
- Proporciona al programador acceso al código fuente de las macros, librerías y subrutinas de Assembler, lo cual ofrece la posibilidad de crear, editar o mejorar las funciones o macros existentes de LPM2.

2.3.3 Cargador de código y cargador de arranque

El Loader code (cargador de código) es un hardware compuesto por muy pocos dispositivos electrónicos (JDM por ejemplo) que se conecta por el puerto paralelo, USB o serial y que descarga el código de máquina desde la PC hacia la memoria de programa del microcontrolador.

El Bootloader (cargador de arranque) es un hardware que se conecta al puerto serial y que permite cargar el código de máquina desde una PC hacia la memoria del µC, la diferencia con un cargador de código convencional (JDM por ejemplo) es que la carga de datos se lleva a cabo por el puerto USART del µC (Tx y Rx) y no por los pines convencionales.

Esto ofrece la ventaja de no tener necesidad de retirar el µC del circuito al cual está conectado (reprogramación In Circuit). Para que un BOOTLOADER pueda funcionar se requiere cargar un pequeño código previamente en la memoria de programa del PIC mediante el JDM por ejemplo. Esta opción la tiene la familia gama alta de microchip.

2.3.4 Macros y subrutinas

El macro de assembler facilita la elaboración de programas. Consiste en una serie de instrucciones y directivas que se agrupan en una sola orden mayor de forma que se comporta como una única instrucción cuando es invocada. Suele utilizarse para automatizar el uso de grupos de instrucciones usadas con frecuencia. Las macros pueden aceptar argumentos, lo que las hace muy flexibles.

Antes de que una macro pueda ser invocada en una línea del programa fuente como si se tratase de una instrucción, debe ser definida por el diseñador con una respectiva sintaxis.

Una diferencia sustancial entre una macro y una subrutina o función, es que la macro al ser invocada "pega" en la memoria de programa el bloque de código que la conforma. Esto suele representar una desventaja con respecto a una subrutina en el sentido que las macros consumen mayor memoria de programa.

Los Argumentos de una macro son los parámetros o datos de entrada que requiere dicha macro para procesar una función específica, por ejemplo el siguiente código declara una macro específica que realiza dos instrucciones.

```
ANADE   MACRO     VAR1, VAR2
        MOVF      VAR1, W
        ADDWF     VAR2, W
 ENDM
```

En el ejemplo anterior el nombre de la Macro es "ANADE", en realidad es cualquier etiqueta que quiera el programador, los argumentos son llamados VAR1, VAR2. Las directivas MACRO Y ENDM definen el inicio y final de la Macro. Para este ejemplo en particular la macro ANADE exige dos parámetros o argumentos de entrada los cuales deben ser dos variables

Respecto a las subrutinas, también llamadas procedimientos, se puede acotar que algunas veces el mismo grupo de instrucciones es ejecutado en diferentes partes de un programa, para optimizar el tamaño de código generado en la memoria de programa, se recomienda agrupar dichas instrucciones en un formato llamado **subrutina o procedimiento**, de la siguiente manera:

```
LIMPIA_PUERTOS  CLRF PORTA
                CLRF PORTB
                CLRF PORTC
                CLRF PORTD
                CLRF PORTE
                RETURN
```

Cada vez que en el programa se requiera limpiar los puertos (ponerlos en cero) solamente se invoca la etiqueta de la subrutina mediante la instrucción **CALL**, ejemplo: CALL LIMPIA_PUERTOS. Usualmente las subrutinas más requeridas se guardan en una librería.

Las subrutinas de assembler no aceptan parámetros de entrada. Cuando una subrutina de assembler es invocada el valor del puntero de programa es guardado en la pila y el nuevo valor del puntero de programa es la dirección de la etiqueta de la subrutina. Cuando el puntero de programa encuentra la instrucción RETURN entonces saca de la pila el último valor guardado y actualiza el puntero de programa de manera que la próxima instrucción a ejecutar corresponderá al siguiente renglón de la instrucción de llamada (CALL o evento de interrupción).

Se puede dar el caso en que una subrutina llama a otra subrutina. Esto es conocido cómo anidamiento de subrutinas, es decir, emplear la instrucción CALL repetidas veces sin que intervenga la instrucción RETURN. El nivel de anidamiento (número de anidamiento de subrutinas permitido) de cada microcontrolador se denomina level stack ó niveles de pila y varía dependiendo de la gama, por ejemplo para la gama baja es de 2

niveles, la gama media (16F84A, 16F877, etc) es de 8 niveles, para la gama alta es de 31 niveles.

Es frecuente necesitar más de una subrutina en los programas. También es habitual que algunas subrutinas se utilicen en varios programas. En estos casos es conveniente disponer de bibliotecas (library) de subrutinas denominadas librerías. El programa ensamblador de microchip MPASM dispone de una directiva denominada INCLUDE que realiza esta función pegando el archivo de referencia en el programa. Dicho archivo se inserta en el código durante el proceso de ensamblado. Por ejemplo: #INCLUDE mislibrerias.inc

2.3.5 Función, instrucción, etiquetas, tipos de variables, directivas

La función es una subrutina que recibe (parámetros o argumentos de entrada) y devuelve (parámetros o argumentos de salida), por ejemplo, una función llamada DIVISION podría recibir dos parámetros (dividendo y divisor) y puede devolver dos parámetros (cociente y residuo). Las macros son una especie de función, la diferencia es que las macros de assembler sólo aceptan parámetros de entrada. El lenguaje assembler para microcontroladores no tiene implementado el concepto de función.

Instrucción (comando o nmemónico) es una palabra reservada (que no puede modificarse) que implica una orden dada al microcontrolador en un lenguaje determinado, por ejemplo, en programación C una instrucción o comando puede ser FOR, IF, WHILE, etc, en programación Assembler una instrucción o comando es MOVF, BCF, RETURN, CALL, etc.

Etiqueta (label) es una expresión compuesta por una palabra alfanumérica que designa a una subrutina o a un renglón del programa, dicha palabra es escogida por el diseñador y usualmente su significado está relacionado con la función de la subrutina o sección de código donde se encuentra. En LPM2 las etiquetas son de color rojo carmesí y siempre están el la primera columna del editor.

Variable GPR (general purpose register) o registro de propósito general, es un lugar de la memoria RAM de dato del microcontrolador en la cual pueden leerse o escribirse datos del programa de usuario. Para poder usar una variable es necesario declararla previamente. Considerando que el PIC 18F452 tiene 1536 byte de memoria RAM de datos, entonces se pueden declarar hasta 1536 variables tipo byte (0 a 255) o 768 variables tipo Word (0 a 65535). Dependiendo del lenguaje de programación existen diversos tipos o formatos de variables: (FLOAT, INTEGER, LONG, DOUBLE, SINGLE, STRING, CHAR, BOOLEANA, BYTE, WORD, DOUBLE WORD, VARIANT, DATE, etc).

Variable SFR (special function register) o registro de proposito especial. A pesar que es una variable se la llama **REGISTRO** para aludir a las posiciones de memoria RAM

reservadas (uso específico) del microcontrolador. Por ejemplo, el puerto B del PIC puede verse como una variable en el sentido que puede leerse o escribirse en él, no obstante, es más apropiado considerarlo como un registro de propósito especial (SFR).

La constante es una expresión que no cambia su valor en todo el código fuente de un programa. Puede tener tantos formatos como las variables.

La directiva de assembler es una palabra reservada interpretada por el compilador MPASM de microchip. Las directivas no generan código de máquina, por lo tanto, no ocuparán espacio en la memoria de dato o programa del µC. Algunas directivas son:

INCLUDE archivo.inc ; Incluye una librería al código general del programa.
DT "MOTOR" : Representa una lista de datos en una tabla
ERROR "WARN" : Genera un mensaje de error en el compilador.
IF A=0 : Compilado condicional

ENDIF
MESSG "Mensaje" : Genera un mensaje en el compilador
CBLOCK 10 : Declaración de variables
ENDC
ORG 4 : Origen de vector de interrupción
END : Fin del programa

2.3.6 Técnicas de interacción con eventos externos

Existen dos técnicas: Polling o sondeo, y el manejo de interrupciones

a. Polling

Es un método poco eficiente de verificar el estado de una entrada digital del microcontrolador mediante un ciclo infinito de testeo por programa, es decir, para saber si se ha presionado un interruptor conectado a tierra en PTA0 el código por técnica polling sería, que ejecutar alguna funcion si se presiona el pulsar en PTA0:

```
LABEL1    BTFSC    PORTC,0
GOTO      LABEL1
```

La principal desventaja de esta técnica es el alto consumo en los recursos del microcontrolador, pues entre más entradas se requiera testear, más ocupado estará el µC, dicho tiempo podría ser requerido para otras aplicaciones, tales como procesamiento, operaciones aritméticas, entradas análogas, etc. En conclusión, no es una buena técnica de programación, lo ideal es que las entradas de un microcontrolador se verifiquen mediante interrupciones programadas.

b. Interrupciones

Una interrupción es un mecanismo mediante el cual un evento interno (fin de

conversión análogo digital, envío de un dato USART, fin de escritura en EEPROM, desbordamiento de un Timer) o externo (presión de un pulsador en entrada digital del puerto PTB0, PTB1, PTB2, PTB4, PTB5, PTB6, PTB7, nótese que PTB3 no genera interrupción, un teclado matricial conectado al puerto B, un teclado de computador, UN SENSOR IR, etc) puede interrumpir la ejecución de un programa principal (Main Program) en cualquier momento.

A partir de entonces se produce automáticamente un salto a una subrutina de atención a la interrupción también conocida como VECTOR DE INTERRUPCIONES, el cual puede ser de alta o baja prioridad (High or Low priority).

Cuando el puntero de programa (pointer program) salta al vector de interrupción (vector número 8 para high priority o vector número 24 para low priority) atiende el conjunto de instrucciones escritas a partir del vector de interrupción, cuando encuentra la instrucción RETFIE (retorno de interrupción) sale de la subrutina de interrupción y continua con la instrucción del programa principal que estaba ejecutando en el momento que se presentó la interrupción.

La interrupción tiene la característica de la inmediatez, nace de la necesidad de ejecutar una subrutina en el instante preciso y, por tanto, se considera su intervención urgente.

Este método es más eficaz que la técnica Polling dado que el µC no perderá tiempo preguntando al pin de entrada para saber el estado, sino que únicamente atenderá al periférico (cualquier dispositivo externo que se pueda conectar al PIC, por ejemplo un pulsador, teclado, sensor, bumpers, etc) cuando éste se lo pida mediante una solicitud de interrupción.

Las interrupciones constituyen el mecanismo más óptimo para la conexión del PIC con el exterior ya que sincroniza la ejecución de programas con los acontecimientos externos.

Esto es muy útil, para el manejo de dispositivos de entrada que requieran una atención inmediata, tales como detección de pulsos externos en un sensor infrarrojo (IR) o un receptor de datos seriales por radiofrecuencia (RF), detección de pulsadores, teclados de computadora, teclados matricial, sensores magnéticos de puertas y ventanas, bumpers en robótica, etc.

El funcionamiento de las interrupciones es similar al de una subrutina invocada por la instrucción CALL, salvo que las interrupciones no son invocadas por una línea de código sino por un evento externo o interno al µC.

Indudablemente una de las principales ventajas de atender un periférico de entrada mediante interrupciones y no por técnica polling es la posibilidad de activar el modo Sleep

(Dormir) o Stand By del µC para la función Power Saving o ahorro de energía, mediante el cual el PIC consume menos de 0.2uA (<200nA) en la gama alta.

b.1 Interrupciones sectorizadas

Algunos µC (Motorola por ejemplo) poseen un Gestor de interrupciones vectorizadas, es decir, cada interrupción tiene una dirección única o vector independiente, esto representa una ventaja en el programa pues facilita el control y detección de las interrupciones.

La familia microchip gama media y alta no dispone de esta opción, por el contrario, para detectar una interrupción es menester verificar (check) cada una de las banderas implicadas en dicha interrupción, este proceso puede ser poco eficiente (tarda más tiempo en detectar la interrupción) para algunas aplicaciones con periféricos high speed (alta velocidad).

b.2 Prioridad de interrupciones

Es un recurso de la gama alta de microchip mediante el cual por software se pueden "priorizar" los eventos o interrupciones internas o externas de un µC. Un ejemplo podría ser un botón de parada de emergencia (Hongo de Seguridad) en un proceso industrial automatizado. Al presionar dicho pulsador el µC debe "obedecer inmediatamente" esta orden y detener el proceso. En el código de esta aplicación se debe configurar al pulsador como (High priority) y al resto de periféricos de entrada como (Low priority).

2.3.7 Programa principal, bancos de memoria, instrucciones

El programa principal es una sección del código fuente que se caracteriza por no atender peticiones de interrupción, en esta parte del código fuente reposa el puntero de programa (pointer program) mientras no se reporte un evento de interrupción interno o externo. Cuando se emplea la técnica Polling usualmente es en main program donde se realiza el testeo cíclico de entradas.

La gama alta de microchip 18FXX2 puede tener hasta 6 bancos de memoria (0 to 7), cada banco contiene 256 registros o variables de 8 bit. El usuario puede acceder a dichos registros o variables mediante el registro especial BSR (Register Selec Bank).

La familia 18FXXXX consta de 75 instrucciones (mientras la gama media sólo tiene 35). Todas las instrucciones consumen 2 bytes de memoria (un renglón de los 16384 disponibles) excepto GOTO, CALL, MOVFF, LFSR que consumen 4 bytes (2 renglones). Todas las instrucciones emplean un ciclo de máquina, excepto las instrucciones ramificadas que emplean 2 ciclos de máquina.

El ciclo de maquina es e tiempo que tarda en ejecutarse un programa; depende de la frecuencia del oscilador conectado al µC y del número de ciclos de máquina ejecutados.

Un ciclo de máquina es la unidad básica de tiempo del µC. Para los PIC un ciclo de

máquina equivale a 4 ciclos del cristal oscilador, es decir, para un cristal de 20MHz el tiempo de un ciclo de máquina será 4/20MHz = 200nS. En efecto, el PIC tarda 1 CM en ejecutar cualquier instrucción (renglón de programa) excepto para aquellas instrucciones ramificadas (Branches instructions) que consume 2 Ciclos de máquina tales como: GOTO, BRA, CALL, BTFSS, BTFSC, RETURN, RETFIE, etc.

Cada instrucción de un µC PIC consume 4 ciclos de reloj, es decir, con un cristal de 40 MHz el PIC ejecutará hasta 10 MIPS (millones de instrucciones por segundo).

Este factor de división (entre 4 para los PIC) varía dependiendo de la arquitectura empleada, por ejemplo existen µC (Motorola, Intel) que consumen hasta 8 ó más ciclos de reloj por cada instrucción.

Nota: MIPS (Millones de instrucciones por segundo). Es el número de instrucciones que ejecuta el PIC en un segundo expresando en millones. Por ejEmplo un PIC con cristal de 20MHz opera a 5 MIPS.

2.4 Herramientas para desarrollo de aplicaciones y simulación

En la presente sección se describen a las principales herramientas para el desarrollo de aplicaciones y simulación.

2.4.1 Aspectos generales

Los simuladores cuentan con características principales que facilitan la programación de los PIC. Estas son:

- editor de ensamblador,
- compilador de alto nivel,
- depuración (debugger in circuit),
- simulador (simulator),
- placas de evaluación,
- emuladores in circuit.

a. Editor de ensamblador

La programación en lenguaje ensamblador es bastante dificultosa pues para ello requiere un relativo dominio, sin embargo permite desarrollar programas muy eficientes al proporcionar al programador el control total del sistema que está desarrollando.

b. Compilador de alto nivel

Las alternativas de programación al lenguaje ensamblado son los lenguajes de alto nivel (Lenguaje C, Basic, Pascal ó LPM2). El MPLAB IDE posee lo que se llama el Project Manager cuyo diagrama de procesos es mostrado en la Figura 2.17.

El propósito es obtener un archivo ejecutable (executable file) desde el archivo fuente (source files), ver Figura 2.18. La programación en un lenguaje de alto nivel reduce el tiempo de desarrollo de una aplicación. El código resultante podría ser mucho más ineficiente que el programado en ensamblador si no se programa con cuidado.

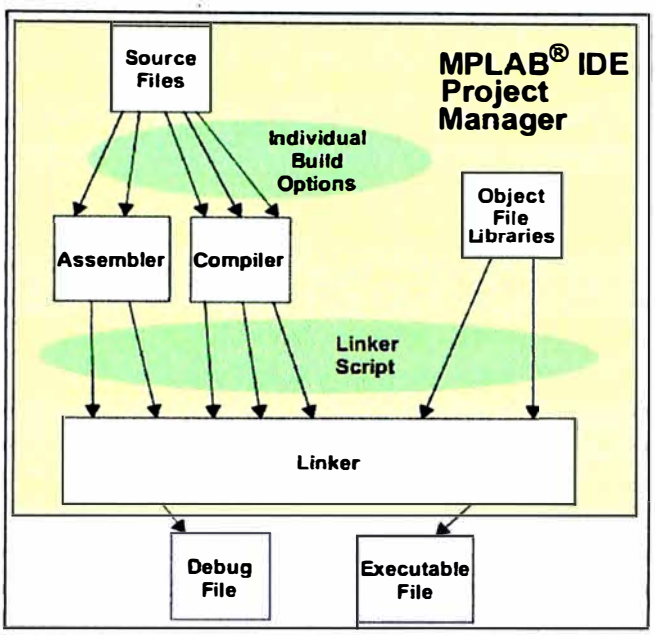


Figura 2.17 Project Manager

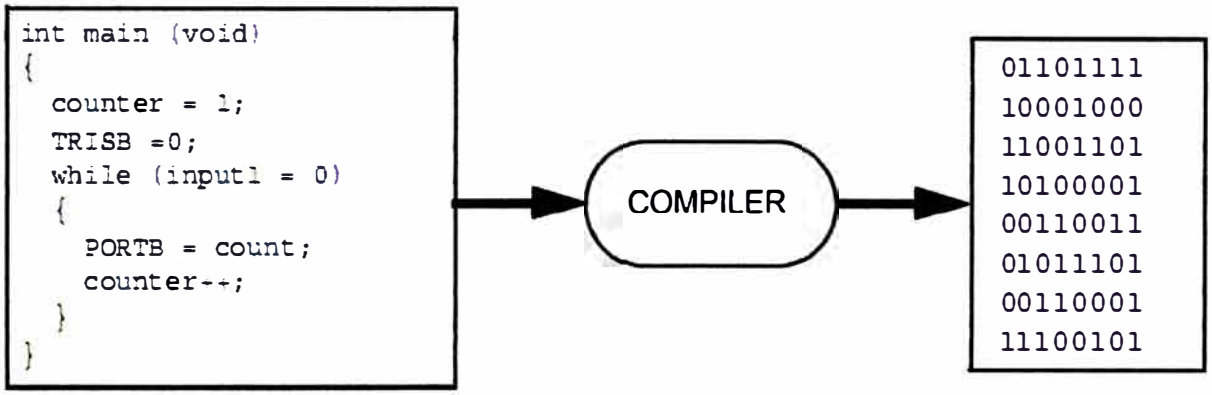


Figura 2.18 Project Manager

c. Depuración (debugger in circuit)

Los microcontroladores programados van a controlar dispositivos físicos (reales), por ellos los desarrolladores necesitan herramientas que les permitan comprobar el buen funcionamiento del microcontrolador cuando es conectado al resto de circuitos. El depurador se encarga de esta tarea.

d. Simulador (simulator)

Son aplicativos software capaces de ejecutar en un PC programas realizados para el microcontrolador. Los simuladores permiten tener un control absoluto sobre la ejecución de un programa, siendo ideales para la depuración de los mismos. Su gran inconveniente es que es difícil simular la entrada y salida de datos del microcontrolador. Tampoco cuentan con los posibles ruidos en las entradas, pero, al menos, permiten el paso físico de la implementación de un modo más seguro y menos costoso, puesto que ahorraremos en grabaciones de chips para la prueba in-situ. Entre los simuladores más comunes está el MP SIM de Microchip y el ISIS Proteus.

e. Placas de evaluación

Se trata de pequeños sistemas con un microcontrolador ya montado y que suelen conectarse a un PC desde el que se cargan los programas que se ejecutan en el microcontrolador.

Las placas suelen incluir visualizadores LCD, teclados, LED's, fácil acceso a los pines de E/S, etc. El sistema operativo de la placa recibe el nombre de programa monitor. El programa monitor de algunas placas de evaluación, aparte de permitir cargar programas y datos en la memoria del microcontrolador, puede permitir en cualquier momento realizar ejecución paso a paso, monitorizar el estado del microcontrolador o modificar los valores almacenados los registros o en la memoria.

f. Emuladores in circuit

Se trata de un instrumento que se coloca entre el PC anfitrión y el zócalo de la tarjeta de circuito impreso donde se alojará el microcontrolador definitivo. El programa es ejecutado desde el PC, pero para la tarjeta de aplicación es como si lo hiciese el mismo microcontrolador que luego irá en el zócalo. Presenta en pantalla toda la información tal y como luego sucederá cuando se conecte el uC real.

2.4.2 MPLAB IDE

MPLAB [5] IDE (Integrated Development Environment) es un conjunto de herramientas integradas gratuitas para el desarrollo de aplicaciones embebidas basadas en los microcontroladores PIC® y dsPIC® de la empresa Microchip. Ver Figura 2.16

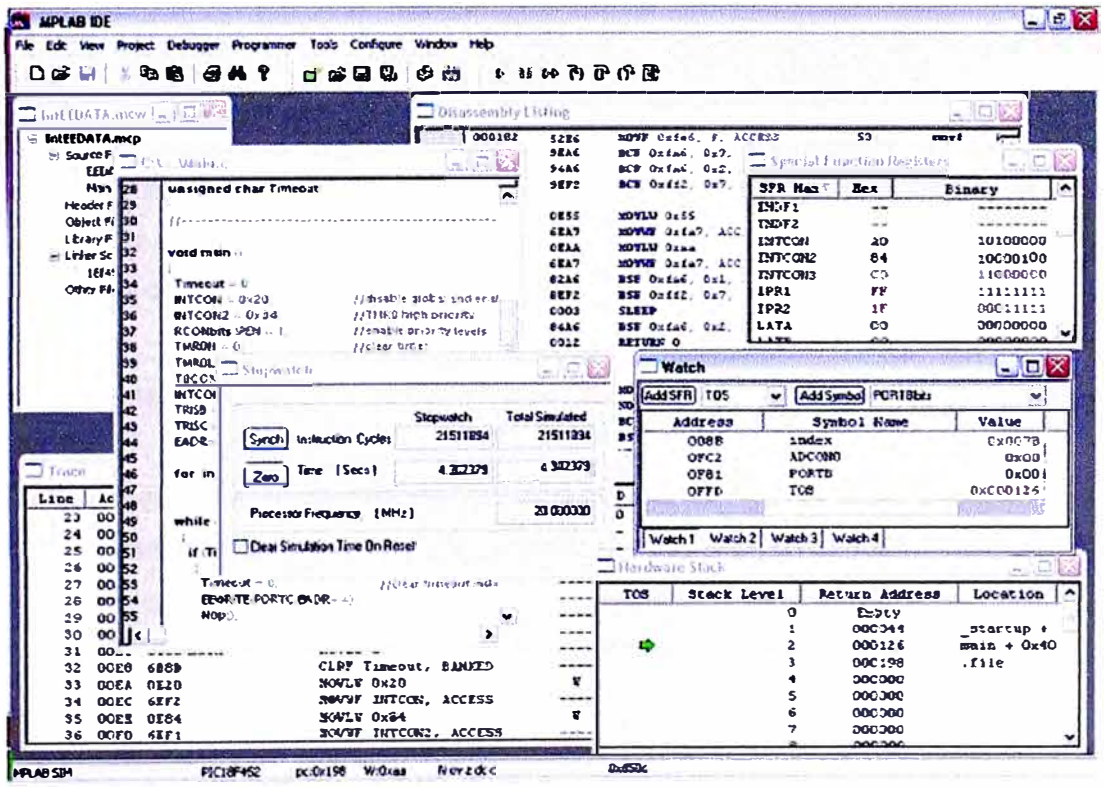


Figura 2.16 IGU del MPLAB

MPLAB IDE Se ejecuta sobre una aplicación de 32 bits sobre el MS Windows®, esta aplicación es fácil de usar e incluye una cantidad de componentes de software gratuitos con la finalidad de que el desarrollo y depuración sea rápida. Funciona cómo una IGU (interfaz gráfica de usuario) unificada sencilla para herramientas de desarrollo de software y hardware adicional (Microchip y third-party).

2.4.3 SPICE

Es un acrónimo inglés de Simulation Program with Integrated Circuits Emphasis (Programa de simulación con énfasis en circuitos integrados). Fue desarrollado por la Universidad de California, Berkeley en 1975 por Donald Pederson [6].

Es un estándar internacional cuyo objetivo es simular circuitos electrónicos analógicos compuestos por resistencias, condensadores, diodos, transistores, etc. Para ello hay que describir los componentes, describir el circuito y luego elegir el tipo de simulación (temporal, en frecuencia, en continua, paramétrico, Montecarlo).

2.4.4 PROTEUS

PROTEUS [7], es un entorno integrado diseñado para la realización completa de proyectos de construcción de equipos electrónicos en todas sus etapas: diseño, simulación, depuración y construcción. La suite se compone de cuatro elementos, perfectamente integrados entre sí:

a. ISIS

Herramienta para la elaboración avanzada de esquemas electrónicos, que incorpora una librería de más de 6.000 modelos de dispositivos digitales y analógicos (Figura 2.17).

b. ARES

Herramienta para la elaboración de placas de circuito impreso con posicionador automático de elementos y generación automática de pistas, que permite el uso de hasta 16 capas. Con ARES el trabajo duro de la realización de placas electrónicas recáe sobre el PC en lugar de sobre el diseñador.

c. PROSPICE

Herramienta de simulación de circuitos según el estándar industrial SPICE3F5.

d. VSM

Permite incluir en la simulación de circuitos el comportamiento completo de los microcontroladores más conocidos del mercado. PROTEUS es capaz de leer los ficheros con el código ensamblado para los microprocesadores de las familias PIC, AVR, 8051, HC11, ARM/LPC200 y BASIC STAMP y simular perfectamente su comportamiento.

Incluso se puede ver su propio código interactuar en tiempo real con su propio hardware pudiendo usar modelos de periféricos animados tales como displays LED o LCD, teclados, terminales RS232, simuladores de protocolos I2C, etc. Proteus es capaz

de trabajar con los principales compiladores y ensambladores del mercado.

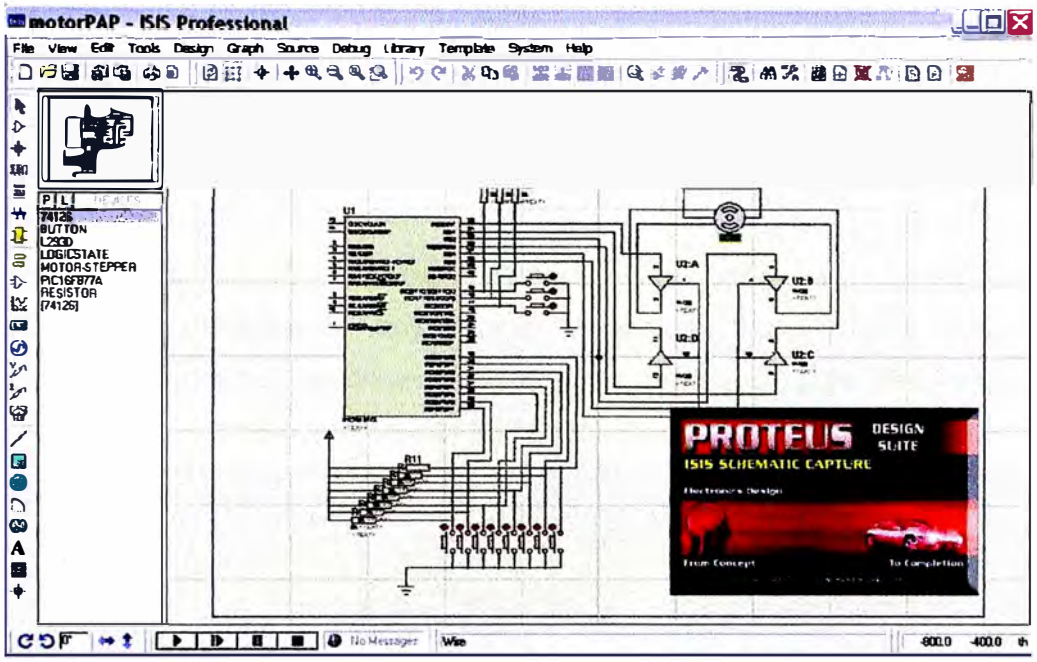


Figura 2.17 ISIS de Proteus

2.4.5 NI Multisim

Multisim de Nacional Instruments [8] es una herramienta ampliamente usada en el campo profesional, gracias a su potente entorno interactivo, no requiere un conocimiento amplio en simulación SPICE ni una amplia experiencia en el uso de herramientas de simulación (Figura 2.18).

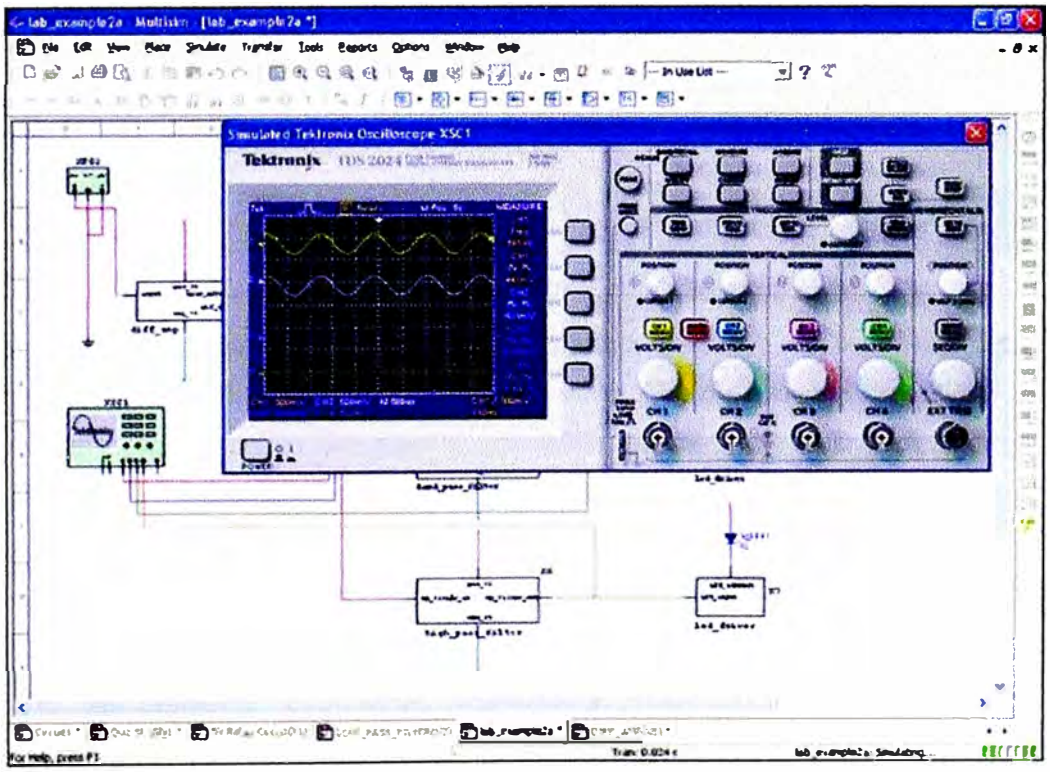


Figura 2.18 Entorno de desarrollo Multisim

Con NI Multisim se realiza la captura, simulación y postprocesado en el mismo entorno. Esto elimina los altos costos de realizar el diseño en múltiples etapas repetitivas, asegurando la alta calidad en la simulación y medidas en el diseño. La familia de productos a nivel profesional de NI Multisim (Base, Full y Power Pro) ofrece un amplio conjunto de herramientas para los diseñadores profesionales:

2.4.6 ORCAD PSpice

El programa SPICE [9], Simulation Program with Integrated Circuits Emphasis, proporciona una herramienta muy útil e interesante para poder determinar el funcionamiento de circuitos eléctricos y electrónicos tanto analógicos como digitales, sin necesidad de tener que recurrir a su montaje en laboratorio (Figura 2.19).

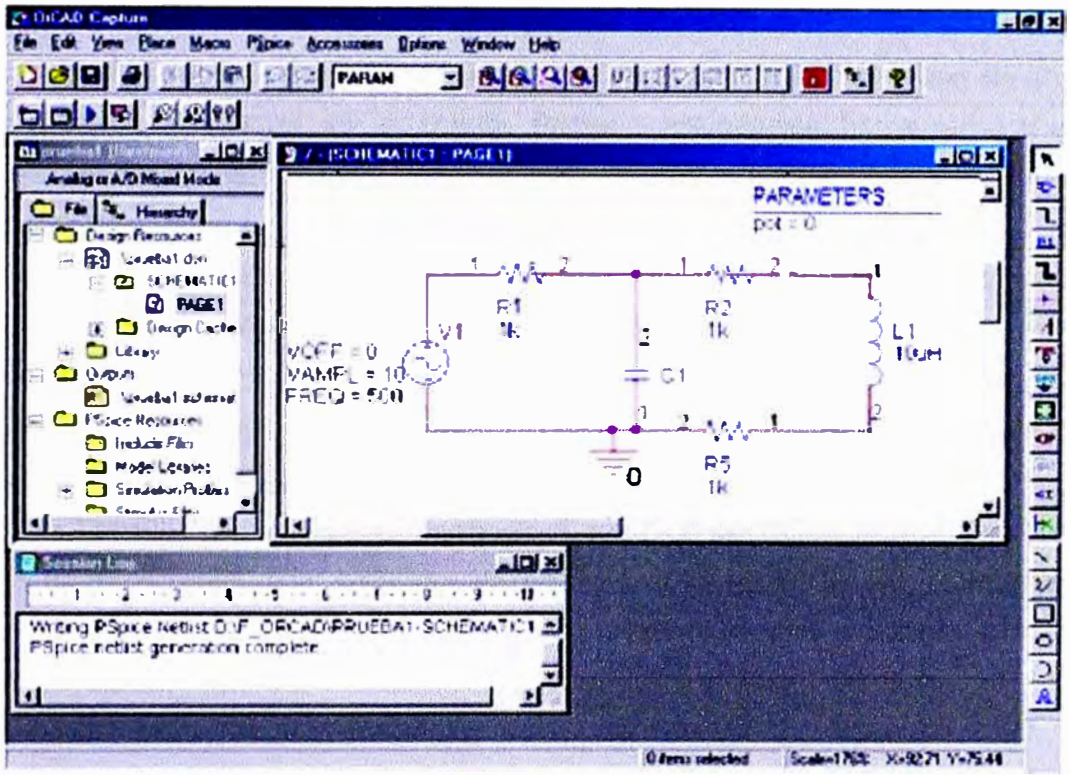


Figura 2.19 Entorno de desarrollo ORCAD PSpice

Fue desarrollado en la Universidad de California-Berkeley, en los años setenta, fue mejorándose hasta aparecer en 1984 el programa PSpice, versión disponible para PC. La absorción de la empresa Microsim Corporation, propietario del programa PSpice, por parte de OrCAD, ha supuesto lograr la unión en una sola aplicación de uno de los programas más potentes de simulación analógica y digital.

El resultado es el programa llamado OrCAD PSpice release 9.1 Demo versión, que se adjunta en un CD-ROM de difusión gratuita y en inglés, que permitirá el diseño y simulación de circuitos analógicos, digitales y mixtos. Hay nuevas versiones ya desarrolladas, como la 10.0, puesto que los equipos de las empresas nunca cesan en su trabajo. Todo ello se puede encontrar en Internet, en la dirección www.orcad.com.

Las versiones modernas de incluyen varias partes, cada cual con su misión específica, si bien son tres las que se considerarán imprescindibles dentro del manejo del programa: Capture, PSpice A/D y Layout.

1. Módulo Capture / Capture Cis. Es un capturador de esquemas que permite dibujar circuitos o modificar los ya creados. Además, dará la posibilidad de editar componentes, seleccionar el tipo de análisis, realizar chequeos eléctricos, etc. Puede considerarse que Capture es el entorno principal de trabajo, porque desde el mismo pueden ejecutarse otros programas auxiliares para configurar señales de estímulos, módulo Editor o procesar gráficamente los resultados de la simulación, módulo PSpice.
2. Módulo PSpice A/D. Es la parte del paquete OrCAD PSpice, encargada de realizar al simulación del comportamiento del circuito para el análisis seleccionado. Cuenta con un capturador de ondas que posibilita visualizar los resultados de la simulación de un modo gráfico a través del monitor del ordenador. Posee como principal herramienta el uso de cursores que determinarán el valor exacto de las coordenadas de un punto cualquiera de la señal representada.
3. Módulo Pspice Model Editor. Es un programa que permitirá modelar cualquier elemento de una librería o incluso diseñar elementos propios a partir de sus características físicas.
4. Módulo Pspice Stimulus Editor. Permitirá generar diferentes tipos de señales, con la posibilidad de visualizarlas a la vez que se están diseñando.
5. Módulo Layout Plus. Permite el diseño PCB ayudándose de los módulos anteriores o ficheros de otros programas: creación del circuito con sus componentes, su ubicación o emplazamiento sobre la placa, la interconexión, la generación de máscaras y finalmente la documentación.

CAPÍTULO III
METODOLOGÍA PARA LA SOLUCIÓN DEL PROBLEMA

En el presente capítulo se exponen los aspectos esenciales del proyecto propuesto.

3.1 Topología del circuito

Se ha establecido que el sistema debe tener las siguientes características:

- Manejo de un motor paso a paso
- Selección de velocidad.
- Selección de sentido de giro: horaria y antihoraria
- Selección de inicio y parada.
- Determinación de PIC

3.1.1 Diagrama de bloques del sistema

El sistema tendría la configuración de bloques mostrada en la Figura 3.1.

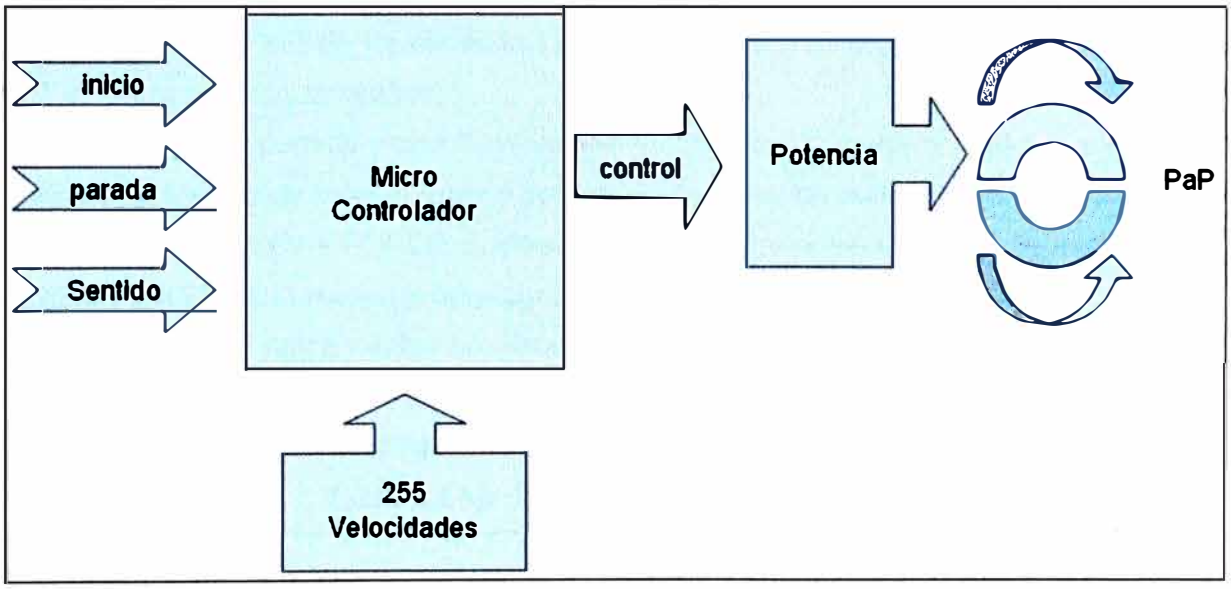


Figura 3.1 Esquema a nivel de bloques

Se establece que:

- Para el arranque y parada del motor, se usarán pulsadores independientes, que recién al soltarlos se reconozca la orden o comando.
- Para el cambio de giro, se usará un botón de enclavamiento, presionado este el PaP gira en un sentido, y liberado en el otro sentido.
- Para determinar la velocidad, se usarán 8 botones, a fin de asegurar 255 velocidades. El valor 01H (00000001) es para la máxima velocidad, mientras que el valor FFH (todos

los botones presionados) para la menor velocidad (Vmax/255).

- El motor a usar será un PaP unipolar de 6 hilos con secuencia de control para torque débil (un solo devanado activo) y para torque fuerte (dos devanados activos).
- Se usará la técnica polling. No se usará interrupciones
- Se determina el uso del PIC 16F877 o el 16F877A [10].

3.1.2 PIC16F877X

Para el desarrollo de la aplicación es necesario tener claro varios aspectos del microcontrolador a usar.

El microcontrolador PIC16F877 de Microchip pertenece a una gran familia de microcontroladores de 8 bits (bus de datos) que tienen las siguientes características generales que los distinguen de otras familias:

- Arquitectura Harvard
- Tecnología RISC
- Tecnología CMOS

Estas características se conjugan para lograr un dispositivo altamente eficiente en el uso de la memoria de datos y programa y por lo tanto en la velocidad de ejecución. El 16F877 es de la gama media y tiene 35 instrucciones de 14 bits de longitud.

Los rangos de voltaje de alimentación van de 4.5 a 6 voltios para el modo estándar (2.5 a 6 para el modo extendido)

El PIC16F877 permite hasta 8 diferentes modos para el oscilador. El usuario puede seleccionar alguno de estos 8 modos programando 2 bits de configuración del dispositivo denominados: FOSC1 y FOSC0, ubicados en un registro especial de configuración en la localidad 2007H de la memoria de programa:

En algunos de estos modos el usuario puede indicar que se genere o no una salida del oscilador (CLKOUT) a través de una patita de Entrada/Salida. Los modos de operación se muestran en la Tabla 3.1:

Tabla 3.1 Modo de operación del oscilador

FOSC1	FOSC0	Modo de operación del oscilador
0	0	LP Baja frecuencia (y bajo consumo de potencia)
0	1	XT Cristal / Resonador cerámico externos, (Media frecuencia)
1	0	HS Alta velocidad (y alta potencia) Cristal/resonador
1	1	RC Resistencia / capacitor externos

Los tres modos LP, XT y HS usan un cristal o resonador externo, la diferencia sin embargo es la ganancia de los drivers internos, lo cual se ve reflejado en el rango de frecuencia admitido y la potencia consumida.

En la Tabla 3.2 se muestran los rangos de frecuencia así como los capacitores recomendados para un oscilador en base a cristal.

Tabla 3.2 Modo de operación del oscilador

Modo	Frecuencia típica	Capacitores recomendados	Modo
		C1	C2
LP	32 khz 200 khz	68 a 100 pf 15 a 30 pf	68 a 100 pf 15 a 30 pf
XT	100 khz 2 Mhz 4 Mhz	68 a 150 pf 15 a 30 pf 15 a 30 pf	150 a 200 pf 15 a 30 pf 15 a 30 pf
HS	8 Mhz 10 Mhz 20 Mhz	15 a 30 pf 15 a 30 pf 15 a 30 pf	15 a 30 pf 15 a 30 pf 15 a 30 pf

En los tres modos mostrados en la tabla anterior se puede usar un cristal o resonador cerámico extemo. En la Figura 3.2 se muestra la conexión de un cristal a las patitas OSC1 y OS2 del PIC.

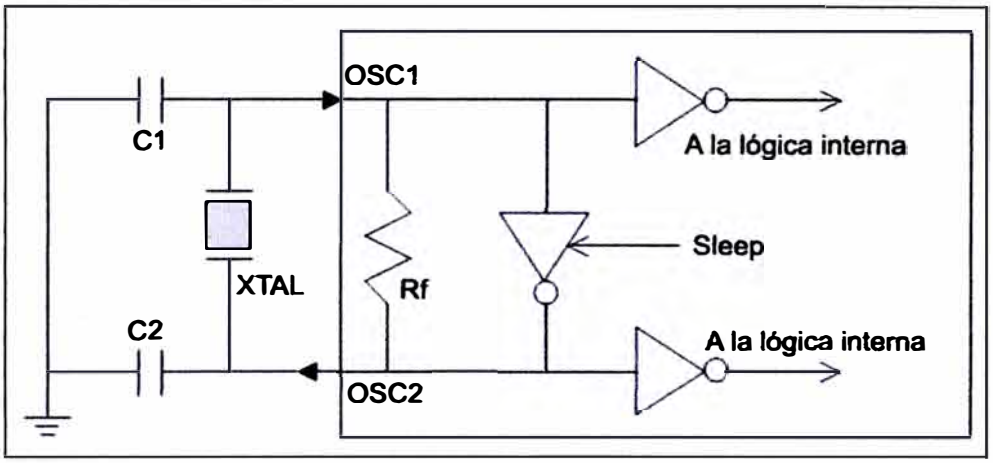


Figura 3.2 Uso de cristal

En los modos RC y EXTRC el PIC puede generar su señal oscilatoria basado en un arreglo RC externo conectado a la patita OSC1 como se muestra en la Figura 3.3. Este modo sólo se recomienda cuando la aplicación no requiera una gran precisión en la medición de tiempos.

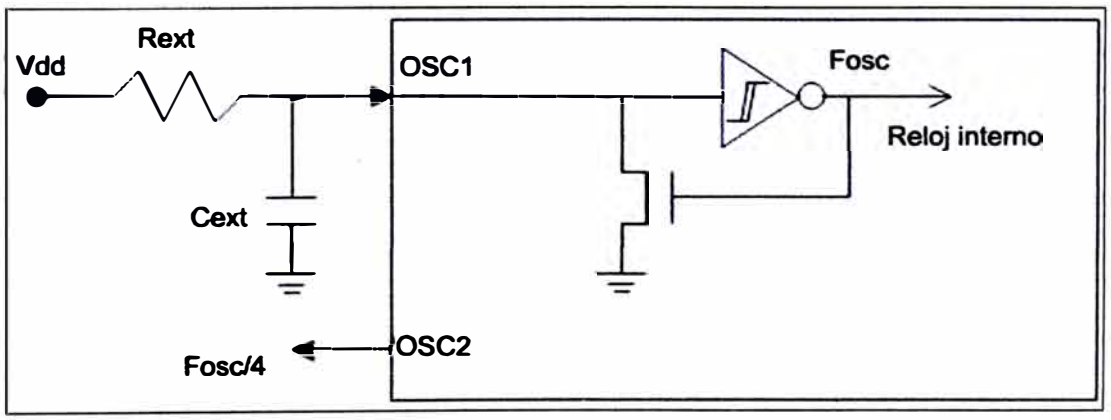


Figura 3.3 Uso de resistencia

Para este caso, la frecuencia de oscilación depende no sólo de los valores de Rext y Cext, sino también del voltaje de la fuente Vdd. Los rangos admisibles para resistencia y capacitor son: Rext: de 3 a 100 Kohms, y Cext: mayor de 20 pf

También es posible conectar una señal de reloj generada mediante un oscilador externo a la patita OSC1 del PIC. Para ello el PIC deberá estar en uno de los tres modos que admiten cristal (LP, XT o HS). La conexión se muestra en la Figura 3.4.

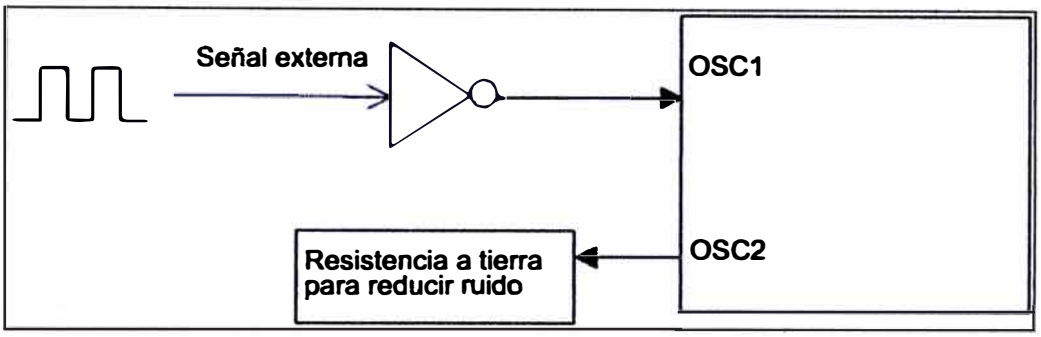


Figura 3.4 Oscilador externo

En los PIC's que poseen el modo de oscilación INTRC, el PIC usa un arreglo RC interno que genera una frecuencia de 4 Mhz con un rango de error calibrable de ±1.5%. Para calibrar el error de oscilación se usan los bits CAL3, CAL2, CAL1 Y CAL0 del registro OSCCAL.

El fabricante ha colocado un valor de calibración para estos bits en la última dirección de la memoria de programa. Este dato ha sido guardado en la forma de una instrucción RETLW XX.

El CPU de esta familia posee tecnología RISC. Posee sólo 35 instrucciones, todas se ejecutan en un ciclo de reloj, excepto los saltos que requieren dos. La frecuencia de operación es de 0 a 20 MHz (200 nseg de ciclo de instrucción).

La memoria del PIC es

- Hasta 8k x 14 bits de memoria Flash de programa
- Hasta 368 bytes de memoria de datos (RAM)
- Hasta 256 bytes de memoria de datos EEPROM
- Lectura/escritura de la CPU a la memoria flash de programa
- Protección programable de código
- Stack de hardware de 8 niveles

Reset e interrupciones: Hasta 14 fuentes de interrupción, Reset de encendido (POR), Timer de encendido (PWRT), Timer de arranque del oscilador (OST), Sistema de vigilancia Watchdog timer,

Otras características: Modo SLEEP de bajo consumo de energía, programación y depuración serie "In-Circuit" (ICSP) a través de dos patitas, rango de voltaje de operación de 2.0 a 5.5 volts, alta disipación de corriente de la fuente: 25mA, rangos de temperatura:

Comercial, Industrial y Extendido, bajo consumo de potencia (Menos de 0.6mA a 3V 4 Mhz; 20 µA a 3V 32 Khz; menos de 1µA corriente de standby o modo SLEEP).

La Figura 3.5 muestra el diagrama de bloques funcional del PIC a utilizar.

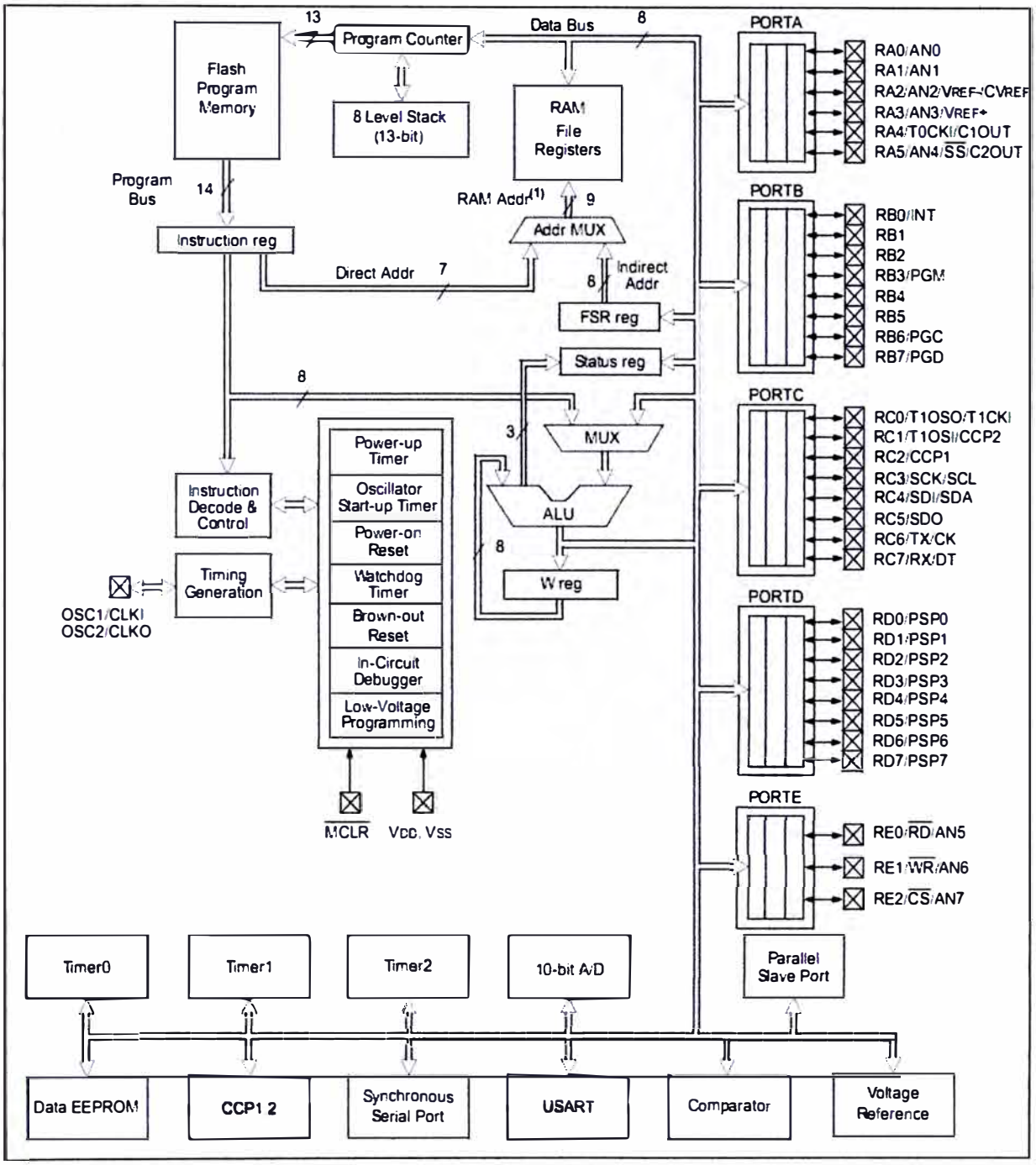


Figura 3.5 Diagrama de bloques del PIC 16F877A

Los registros de la CPU son:

Registro PC.- Registro de 13 bits que siempre apunta a la siguiente instrucción a ejecutarse. En la siguiente sección se dan mayores detalles en el manejo de este registro.

- Registro de Instrucción.- Registro de 14 bits. Todas las instrucciones se colocan en él

para ser decodificadas por la CPU antes de ejecutarlas.

- Registro W.- Registro de 8 bits que guarda resultados temporales de las operaciones realizadas por la ALU.

- Registro STATUS.- Registro de 8 bits, cada uno de sus bits (denominados Banderas) es un indicador de estado de la CPU o del resultado de la última operación como se indica en la Figura 3.6. (R= Bit leíble; W= Bit Escribible; U= No implementado, se lee como 0; -n= Valor después del Reset de encendido)

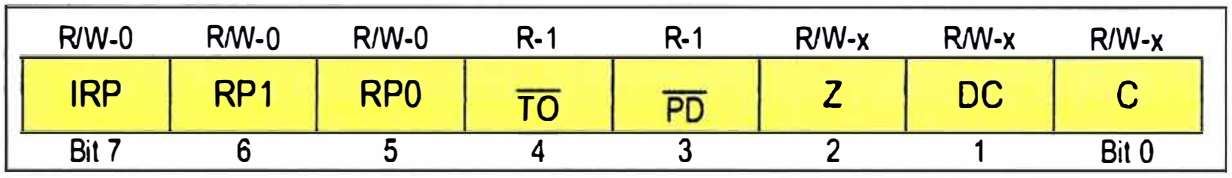


Figura 3.6 Registro Status

Z.- Este bit se pone (=1) para indicar que el resultado de la última operación fue cero, de lo contrario se limpia (=0)

C.- Bit de acarreo/préstamo' de la última operación aritmética (en el caso de préstamo (resta), el bit se invierte antes de guardarse)

DC.- Acarreo/ préstamo proveniente del cuarto bit menos significativo. Funciona igual que el bit C, pero para operaciones de 4 bits.

El conjunto de instrucciones del PIC16 es altamente ortogonal y se clasifica en tres categorías:

- Operaciones orientadas a bytes
- Operaciones orientadas a bits
- Operaciones literales y de control

Cada instrucción es una palabra de 14-bit divida en un opcode el cual especifica el tipo de instrucción y uno o más operandos el cual posteriormente especifica la operación de la instrucción. Los formatos para cada categoría es presentada en la Figura 3.7, mientras que los campos de los opcodes están resumidas en la Tabla 3.3.

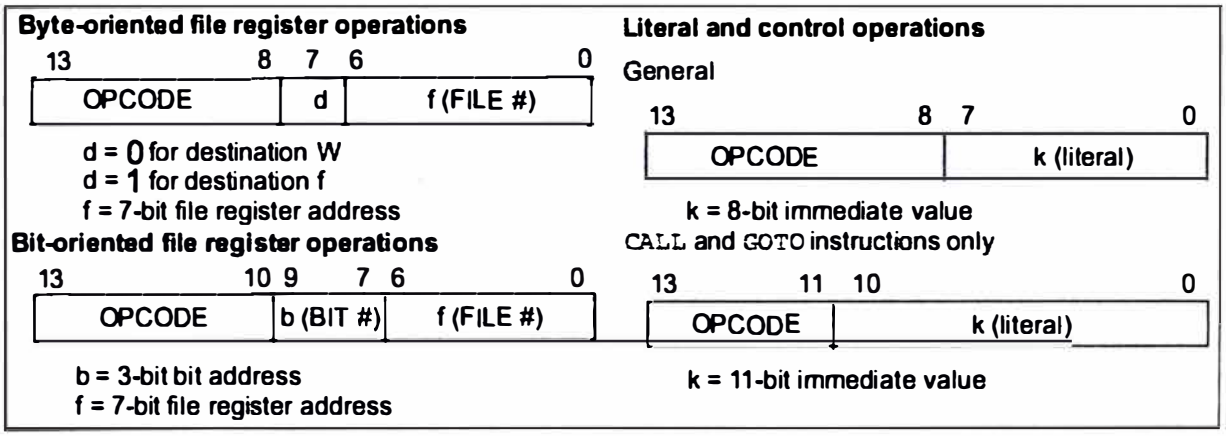


Figura 3.7 Formatos de instrucciones

Tabla 3.3 Descripción de datos de opcode

Campo	Descripción
f	Dirección del archivo de registro (0x00 to 0x7F) RAM
W	Registro de trabajo (acumulador)
b	Dirección del bit dentro de un registro f de 8 bits
k	Campo literal, constante o rótulo
x	Irrelevante (= 0 o 1). El ensamblador generará código con x = 0. Es la forma recomendada para compatibilidad con todas las herramientas de software Microchip.
d	Selección de destino; d = 0: se almacena en W, d = 1: se almacena en f. El valor por defecto es d = 1.
PC	Contador de programa
TO	Time out bit
PD	Power down bit

La Tabla 3.4, 3.5 y 3.6 lista las instrucciones reconocidas por el MPASM™ Assembler. Una completa descripción de cada instrucción es también disponible en el documento PICmicro® Mid-Range MCU Family Reference Manual (DS33023) [11].

En las Instrucciones orientadas a byte, 'f' representa la dirección del archivo de registro que será usado por la instrucción. 'd' representa el destino del resultado; Si d es cero, el resultado es colocado en el registro W, si d es uno, el resultado es colocado en el registro de archivo especificado en la instrucción.

Tabla 3.4 Instrucciones orientadas a byte

Mnemónico	Descripción	Ciclos	Código de Máquina	Banderas afectadas
ADDWF f,d	Suma f + W	1	00 0111 dfff ffff	C,DC,Z
ANDWF f,d	W AND f	1	00 0101 dfff ffff	Z
CLRF f	Limpia f	1	00 0001 1fff ffff	Z
CLRW	Limpia W	1	00 0001 0xxx xxxx	Z
COMF f,d	Complementa los bits de f	1	00 1001 dfff ffff	Z
DECF f,d	Decrementa f en 1	1	00 0011 dfff ffff	Z
DECFSZ f,d	Decrementa f, escapa si 0	1(2)	00 1011 dfff ffff	
INCF f,d	Incrementa f en 1	1	00 1010 dfff ffff	Z
INCFSZ f,d	Incrementa f, escapa si 0	1(2)	00 1111 dfff ffff	
IORWF f,d	W OR f	1	00 0100 dfff ffff	Z
MOVF f,d	Copia el contenido de f	1	00 1000 dfff ffff	Z
MOVWF f	Copia contenido de W en f	1	00 0000 1fff ffff	
NOP	No operación	1	00 0000 0xx0 0000	
RLF f,d	Rota f a la izquierda	1	00 1101 dfff ffff	C
RRF f,d	Rota f a la derecha	1	00 1100 dfff ffff	C
SUBWF f,d	Resta f – W	1	00 0010 dfff ffff	C,DC,Z
SWAPF f,d	Intercambia nibbles de f	1	00 1110 dfff ffff	
XORWF f,d	W EXOR f	1	00 0110 dfff ffff	Z

Para instrucciones orientadas a bit, 'b' representa el campo de bit que selecciona el bit afectado por la operación, mientras que 'f' representa la dirección del archivo de registro en el cual el bit se localiza.

Tabla 3.5 Instrucciones orientadas a bit

Mnemónico	Descripción	Ciclos	Código de Máquina	Banderas afectadas
BCF f,b	Limpia bit b en f	1	01 00bb bfff ffff	
BSF f,b	Pone bit b en f	1	01 01bb bfff ffff	
BTFSC f,b	Prueba bit b en f, escapa si 0	1(2)	01 10bb bfff ffff	
BTFSS f,b	Prueba bit b en f, escapa si 1	1(2)	01 11bb bfff ffff	

Para operaciones con literales y de control, 'k' representa una constante de 8 u once bits, o un valor literal.

Tabla 3.6 Instrucciones con literales o de control

Mnemónico	Descripción	Ciclos	Código de Máquina	Banderas afectadas
ADDLW k	Suma literal k + W→ W	1	11 111x kkkk kkkk	C,DC,Z
ANDLW k	k AND W→ W	1	11 1001 kkkk kkkk	Z
CALL k	Llamado a subrutina	2	10 0kkk kkkk kkkk	
CLRWDT	Limpia timer del watchdog	1	00 0000 0110 0100	*TO , PD*
GOTO k	Salto a la dirección k	2	10 1kkk kkkk kkkk	
IORLW k	k OR W→ W	1	11 0000 kkkk kkkk	Z
MOVLW k	Copia literal a W	1	11 00xx kkkk kkkk	
RETFIE	Retoma de interrupción	2	00 0000 0000 1001	
RETLW k	Retoma con literal k en W	2	11 01xx kkkk kkkk	
RETURN	Retoma de subrutina	2	00 0000 0000 1000	
SLEEP	Activa Modo standby	1	00 0000 0110 0011	*TO , PD*
SUBLW k	Resta k – W→ W	1	11 110x kkkk kkkk	C,CD,Z
XORLW k	k EXOR W→ W	1	11 1010 kkkk kkkk	Z

Un ciclo de instrucción consiste de cuatro periodos de oscilador; para una frecuencia de 4 MHz, este proporciona un tiempo de ejecución normal de 1 µs. Todas las instrucciones son ejecutadas dentro de un solo ciclo de instrucción, a menos que un test condicional sea cierto, o el contador de programa es cambiado cómo resultado de una instrucción. Cuando esto ocurre, la ejecución toma dos ciclos de instrucción con un segundo ciclo ejecutado como NOP.

Los PIC tienen dos tipos de memoria: Memoria de Datos y Memoria de programa, cada bloque con su propio bus: 1) Bus de datos y 2) Bus de programa; por lo cual cada bloque puede ser accedido durante un mismo ciclo de oscilación. La memoria de datos consta de dos áreas mezcladas y destinadas a funciones distintas: Registros de Propósito Especial (SFR), y Registro de Propósito General (GPR). Los SFR son

localidades asociadas específicamente a los diferentes periféricos y funciones de configuración del PIC y tienen un nombre específico asociado con su función. Mientras que los GPR son memoria RAM de uso general. Toda la memoria de datos está organizada en 4 bancos numerados 0, 1, 2 y 3. Esto es mostrado en la Figura 3.8

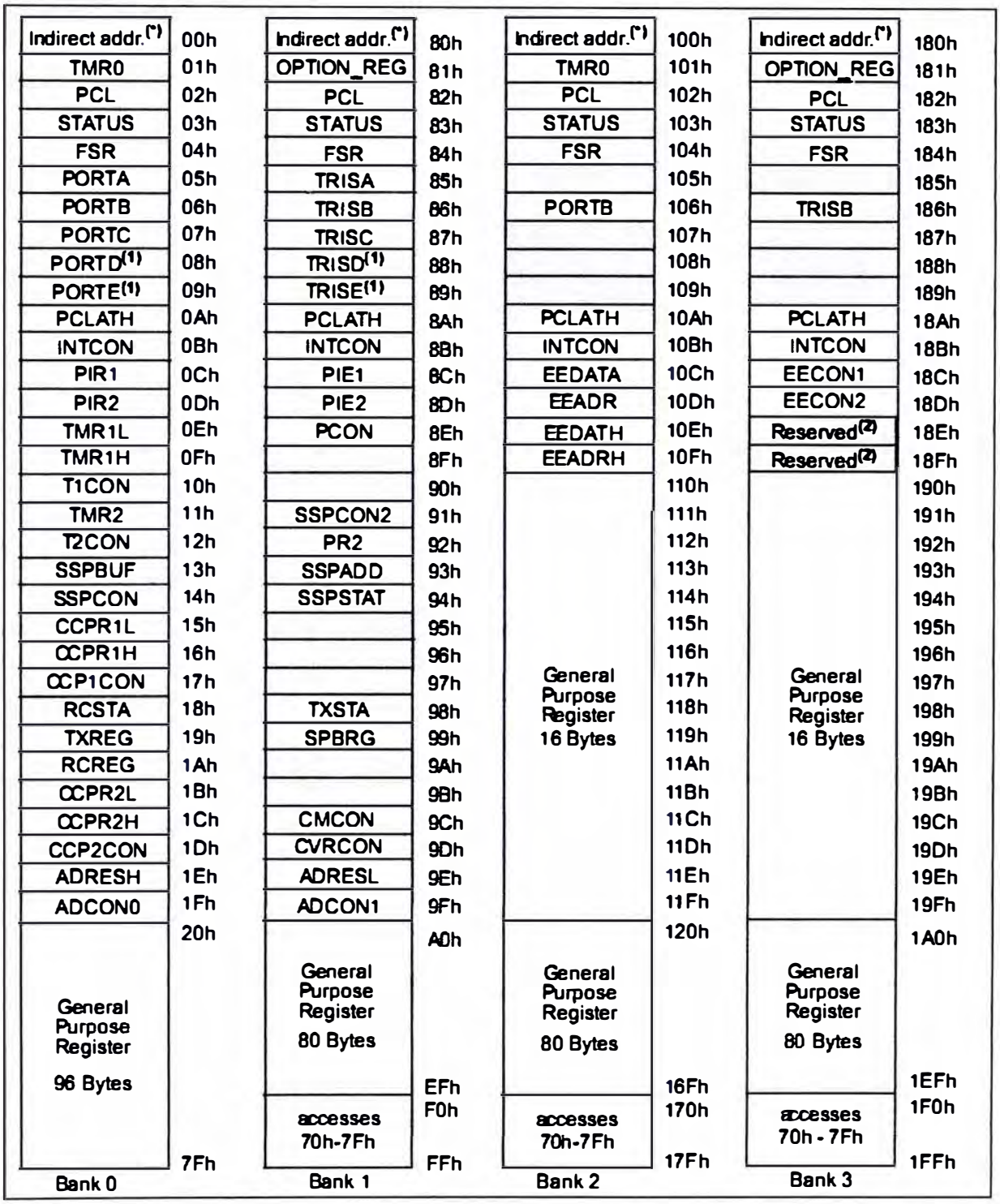


Figura 3.8 Detalle del mapa del archivo de registros y su organización

3.2 implementación de algoritmo

El algoritmo y explicación, además del código de programación son expuestos a

continuación.

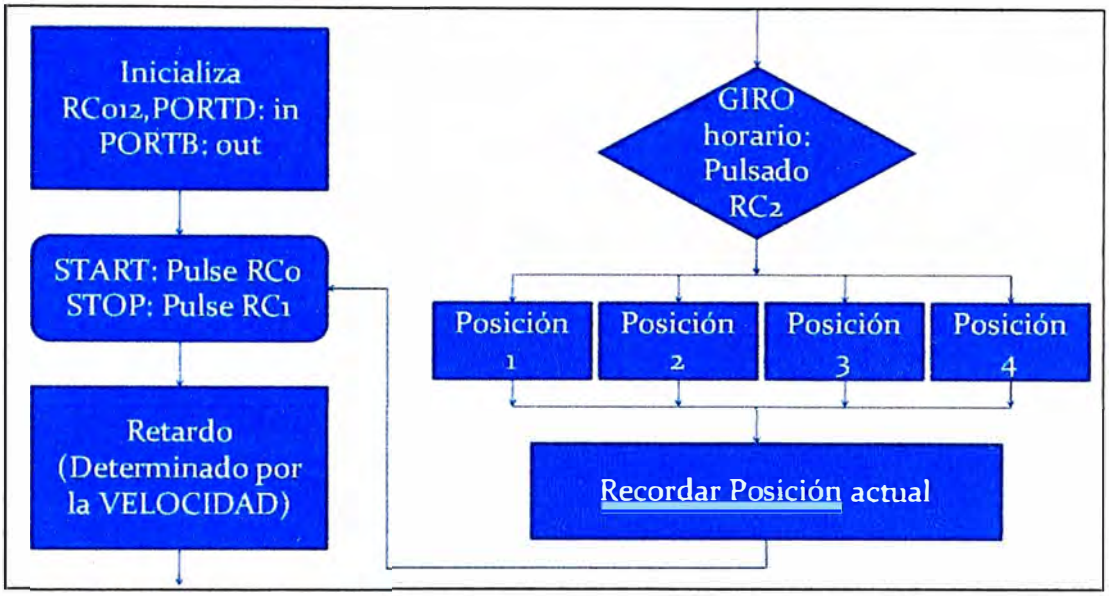


Figura 3.9 Diagrama de flujo

Según se muestra e la Figura 3.9, se inicializa los puertos RC 0,1 y 2 cómo entradas, también al Puerto D como entrada; el puerto B es configurado como salida.

Para el START se establece el pulso de subida de RC0, lo mismo que para el STOP. Se determina un tiempo de retardo (tiempo entre paso) en 5 ms para la velocidad máxima, luego esta se reduce según los valores en el puerto D. El giro es dado por un botón de enclavamiento que cambiará el giro sin importar en que posición se encuentre.

Las posiciones son 0°, 90°, 180° y 270° (torque débil). Ver Figura 3.10.

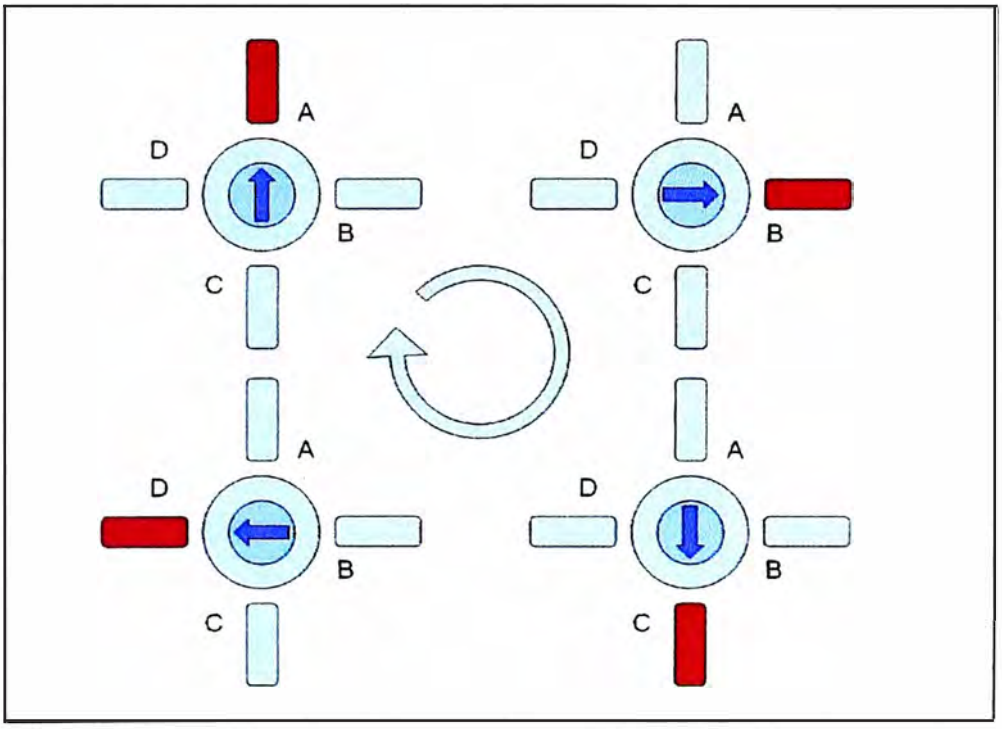


Figura 3.10 Full Step débil (un devanado excitado).

Las posiciones son 45°, 135°, 225° y 315° para el torque fuerte. Ver Figura 3.11.

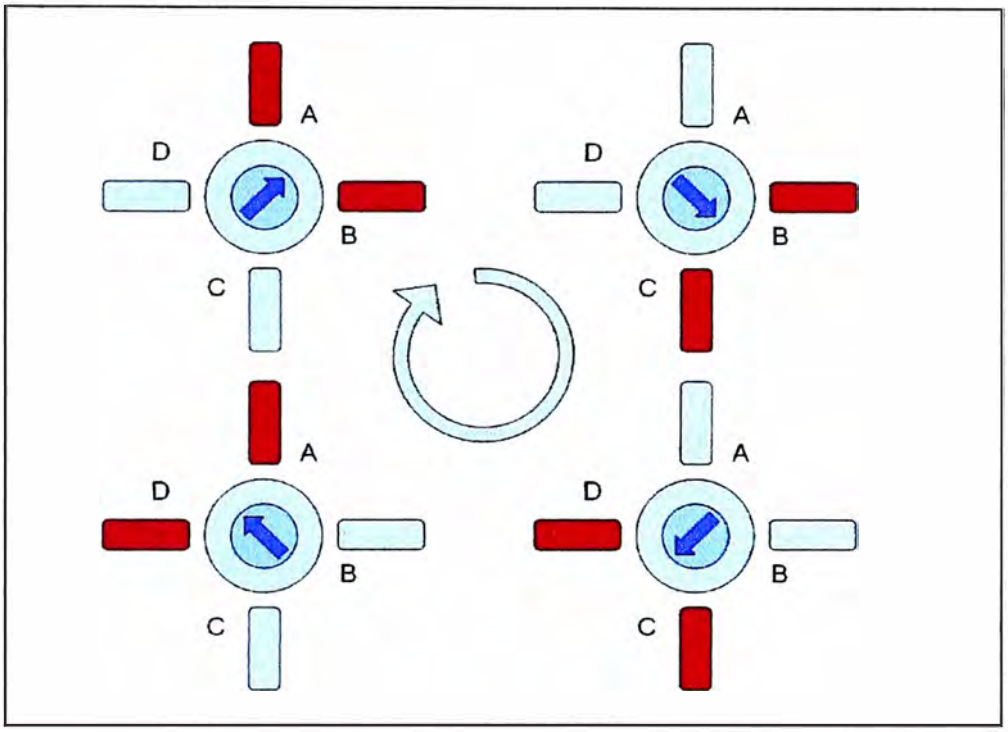


Figura 3.11 Full Step fuerte (dos devanados excitados).

La técnica para el sensado de los controles es la del polling (no se usan interrupciones). Cada botón de START y STOP es considerado con el pulso de subida, es decir que se sensa si cambia de 1 a 0, para luego esperar a que cambie de 0 a 1 para aplicar la orden de arranque o parada, según corresponda.

El archivo de programación es motor_PAPunipolar.asm. El código mostrado es el aplicado al torque débil.

```
__CONFIG _CP_OFF & _PWRTE_ON & _WDT_OFF & _LVP_OFF & _HS_OSC
;*****************************************
;motor_PAPunipolar.asm
;*****************************************
    LIST P=16F877
    INCLUDE<P16F877.INC>          ; sirve para bajar nombres ESPECIALES
    include<macropic.inc>
    CBLOCK              0X20
    ESTADO_ACTUAL                 ; variable para RECORDAR el ESTADO
    VELOX                         ; variable para determinar la Velocidad
    ENDC

    ORG             0
; INICIALIZACION
    bsf             STATUS,RP0
    clrf    TRISB                 ;
    movlw b'00000110'
    movwf ADCON1
```

```
    movlw 0xff
    movwf TRISC                 ; En el pin RC2 se invertirá el sentido de giro
    movlw 0xFF
    movwf TRISD                 ; En este puerto determinaremos la velocidad
    bcf   STATUS,RP0
    clrf  ESTADO_ACTUAL
    clrf  PORTB
START                           ;———boton de START———
    btfsc PORTC,0
    goto  $-1
    btfss PORTC,0
    goto  $-1
    goto  MAIN
;————————————————————————————
STOP                            ;———boton de STOP———
    clrw                        ; limpiamos la pila (call-return)
    btfss PORTC,1
    goto  START
    goto  MAIN
;————————————————————————————
MAIN
    bsf   PORTB,4     ; seteamos el bit RB4
    bsf   PORTB,5     ; seteamos el bit RB5
    btfss TRISC,2             ; es el bit RC2=1? si es asi salta
    goto  PRINCIPAL0          ; se va a la etiqueta PRINCIPAL0
    goto  PRINCIPAL1          ; se va a la etiqueta PRINCIPAL1

PRINCIPAL0

    call  VELOCIDAD           ; se va a la rutina de VELOCIDAD
    call  GIRO1               ; rutina que produce un giro de 90ºal motor
    movwf PORTB                       ; el motor gira 90º
    incf  ESTADO_ACTUAL       ; Voy al siguiente estado del motor PAP
    goto  STOP                ; checkeo si se preciono STOP

PRINCIPAL1

    call  VELOCIDAD           ; se va a la rutina de VELOCIDAD
    call  GIRO2               ; rutina que produce un giro de 90ºal motor
    movwf PORTB               ; el motor gira 90º
    incf  ESTADO_ACTUAL       ; Voy al siguiente estado del motor PAP
    goto  STOP                ; checkeo si se preciono STOP

;————————Giro Antihorario————————————
GIRO1
    movf  ESTADO_ACTUAL,W         ; Verificamos el estado Actual
    andlw b'00000011'         ; solo 4 estados
    addwf PCL,1               ; Extraemos el siguiente estado de la Tabla
    retlw b'00000001'
    retlw b'00000010'
    retlw b'00001000'
    retlw b'00000100'
;————————Giro Horario————————————————
```

```
GIRO2
    movf   ESTADO_ACTUAL,W        ; Verificamos el estado Actual
    andlw  b'00000011'            ; solo 4 estados
    addwf  PCL,1                  ; Extraemos el siguiente estado de la Tabla
    retlw  b'00000100'
    retlw  b'00001000'
    retlw  b'00000010'
    retlw  b'00000001'
;————Metodo para el cambio de Velocidad——
VELOCIDAD
    movf   PORTD,0                ; chequeamos la velocidad seleccionada
    addlw  .1                     ; para invertir los valores
    btfsc  STATUS,Z               ; Si la velocidad es cero el motor se parara
    goto   STOP                   ; chequeamos si pulsaron STOP
    movf   PORTD,0
    movwf VELOX                   ; El valor que queremos lo aplicamos a los retardos
LAZO_1                            ; Etiqueta de retardos para seleccionar velocidad
    delay_ms      .5              ; 5 en base 10 para los 5 ms
    decfsz VELOX
    goto   LAZO_1
    return

    include<retardo.asm>          ; incluimos la subrutina de retardos
    end
```

- _CONFIG: Es el inicio de la configuracion
- _CP_OFF: Opción de protección de código APAGADO
- _WDT_OFF: Opción de perro guardián apagada
- _LVP_OFF: Opción de MODO BAJO CONSUMO APAGADO (Low Voltage)
- _PWRTE_ON Power-up Timer ON
- _HS_OSC Oscilador en modo de alta velocidad

VELOX simplemente multiplica el retardo ya dado de 5ms.

- Si VELOX es 1 solamente una vez se ejecuta, que seria la máxima velocidad
- Si VELOX es 2 se duplica el tiempo a 10ms
- Si VELOX es 3 se triplica el tiempo a 15ms

Y así sucesivamente hasta llegar al valor máximo de VELOX que es .255 que implica 5ms x 255 que sería la mínima velocidad por paso.

Las modificaciones en el código para el control de Torque fuerte son realizadas en las secciones Giro 1 y Giro 2 del código principal, cómo se muestran a continuación

```
;————————Giro Antihorario————————————
GIRO1
    movf   ESTADO_ACTUAL,W        ; Verificamos el estado Actual
    andlw  b'00000011'            ; solo 4 estados
    addwf  PCL,1                  ; Extraemos el siguiente estado de la Tabla
    retlw  b'00000011'
    retlw  b'00001010'
    retlw  b'00001100'
```

```
    retlw   b'00000101'
;————————Giro Horario————————————————

GIRO2
    movf    ESTADO_ACTUAL,W         ; Verificamos el estado Actual
    andlw   b'00000011'             ; solo 4 estados
    addwf   PCL,1                   ; Extraemos el siguiente estado de la Tabla
    retlw   b'00000101'
    retlw   b'00001100'
    retlw   b'00001010'
    retlw   b'00000011'
```

Esto quiere decir en DCBA → AB=11, luego BC=11, CD=11, DA=11 para cada caso los demás valores están en cero

3.3 Diseño en el Proteus

Se usa el Proteus para el esquema y la simulación del sistema de control de velocidad de un motor paso a paso unipolar. El esquema mostrado en la Figura 3.15 corresponde al sistema de control de torque débil el cual, cómo ya fue explicado, es cuando sólo uno de los devanados está excitado (Figura 3.16 en simulación).

En la Figura 3.12 se muestra el detalle del puerto RD para la selección de velocidad. Se utilizan unos pulsadores de enclavamiento para definir valores numéricos entre 1 y 255, es decir la cantidad de veces por la que la velocidad máxima es dividida. Según se muestra en la figura, el pulsador presionado coloca al pin RD0 en "0" lógico para la velocidad máxima, para la mínima deben estar todos presionados (todos en "0")

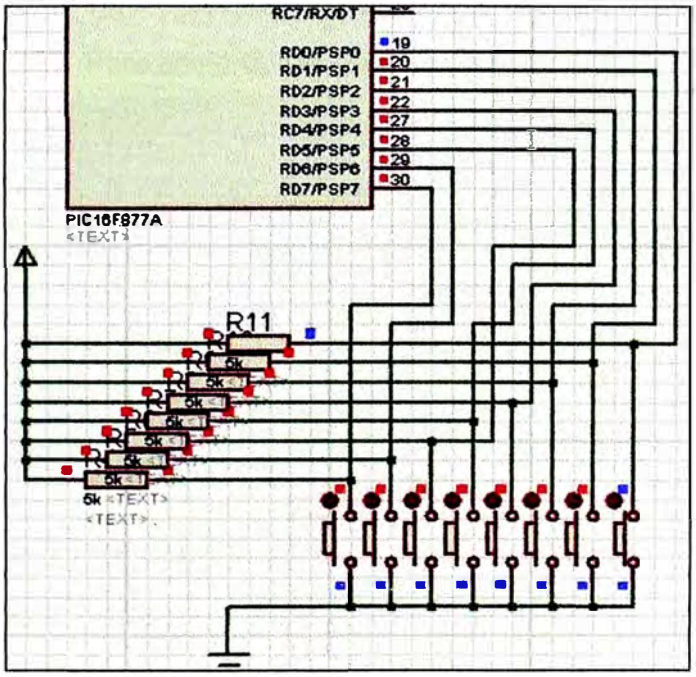


Figura 3.12 Selección de velocidad

La Figura 3.13 muestra el detalle del arranque, parada y cambio de giro del circuito (RC0, RC1 y RC2 respectivamente). Al pulsarse y soltarse el correspondiente a RC0 el

giro se inicia; al pulsar y soltar el correspondiente a RC1 el motor se detiene; al enclavarse o soltarse el pulsador correspondiente a RC2 el giro cambia de sentido,

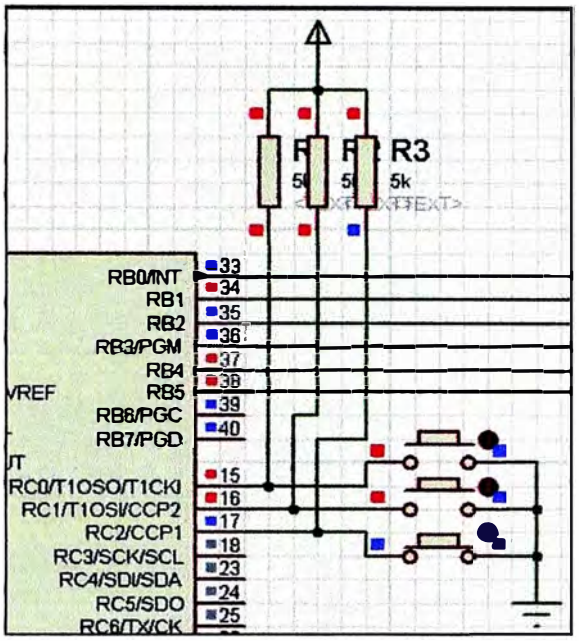


Figura 3.13 Arranque, parada y cambio de giro

La Figura 3.14 muestra la etapa de excitación del PaP. La excitación es proporcionada por el puerto RB. En colores diferentes se muestra la ruta de excitación de los devanados correspondientes. Rojo es para RB0, Azul es para RB1, Verde es para RB2, y Naranja para RB3. De acuerdo a ello el giro en un sentido corresponde a la excitación por parte de RB1→RB3→RB2→RB0, y en sentido contrario es RB0→RB2→RB1→RB3. Para ese propósito se usa un buffer 74126 [12].

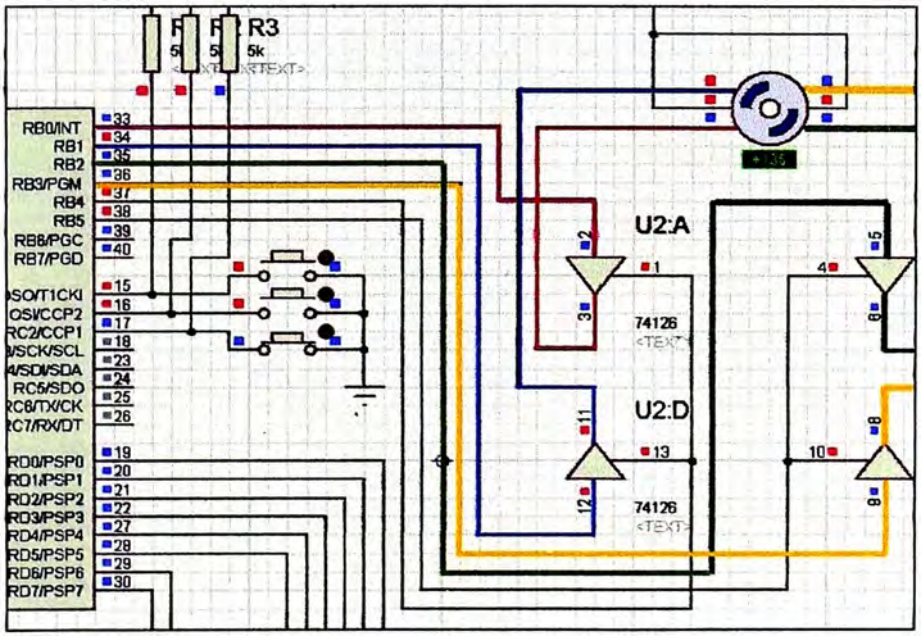


Figura 3.14 Excitación (RB1 activo)

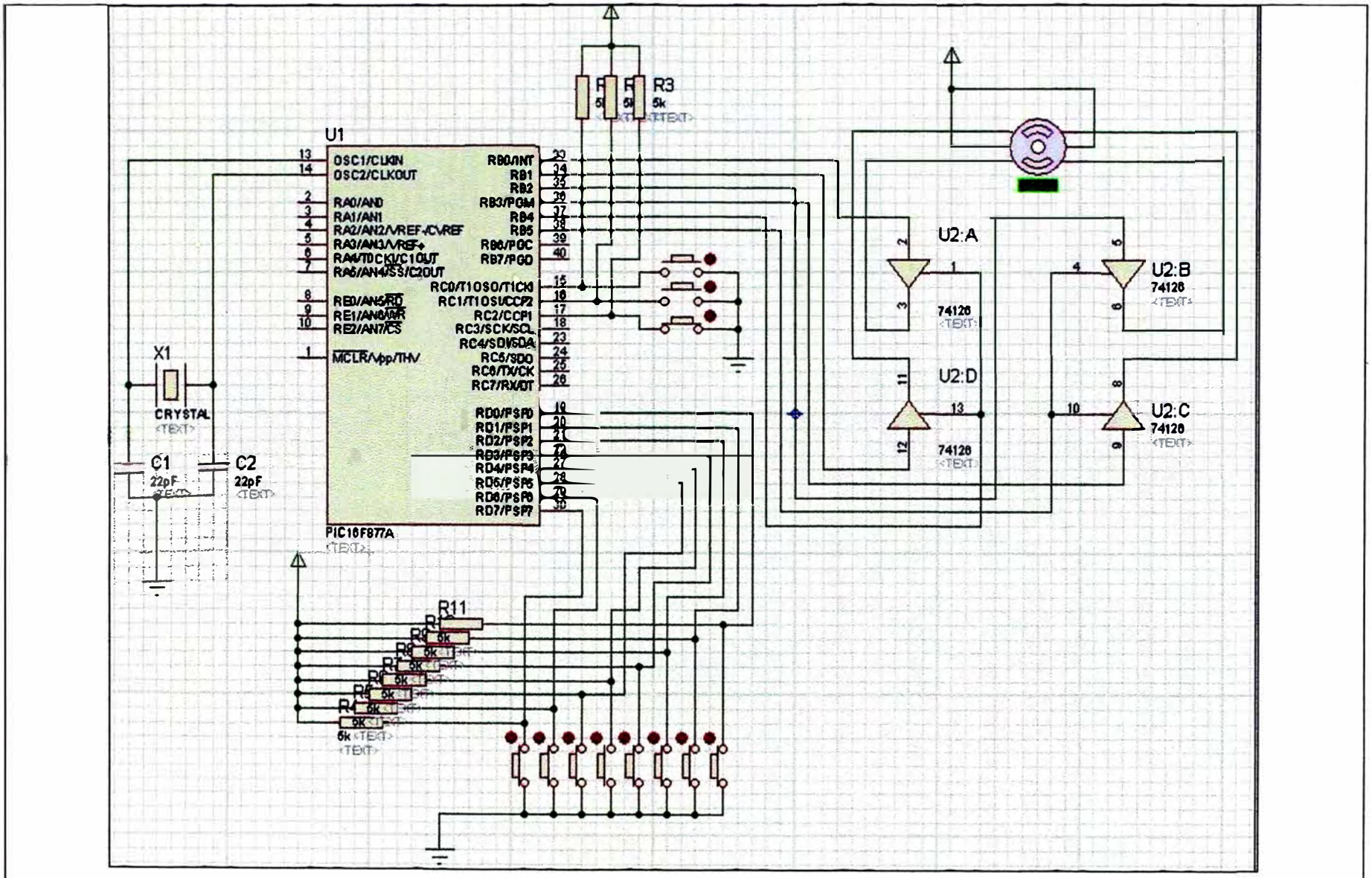


Figura 3.15 Esquema sin simulación

3.4 Modificación para torque fuerte

Para el caso del torque fuerte, la corriente de consumo del motor es muy grande comparada con la corriente que el microcontrolador puede suministrar, por tal motivo es imprescindible construir un circuito de potencia que cumpla la función de interfaz entre el microcontrolador y el PaP.

Según se muestra en la Figura 3.16, la interfaz de potencia se implementa con transistores NPN TIP41 de silicio; estos soportan corrientes de hasta de 2A, y son adecuados para aplicaciones de conmutación.

Se observa que los dos terminales comunes del PaP se han conectado a una fuente de 24V. Cada devanado del motor está conectado al colector de un transistor NPN TIP 41. Las resistencias brindan corriente suficiente para que los transistores puedan operar en corte y saturación.

De esta forma, cuando en la línea de control existe un cero lógico, pondrá al transistor en corte, desactivando la bobina correspondiente del motor.

Cuando en la línea de control haya un uno lógico, la corriente pondrá al transistor en saturación, activando la bobina correspondiente.

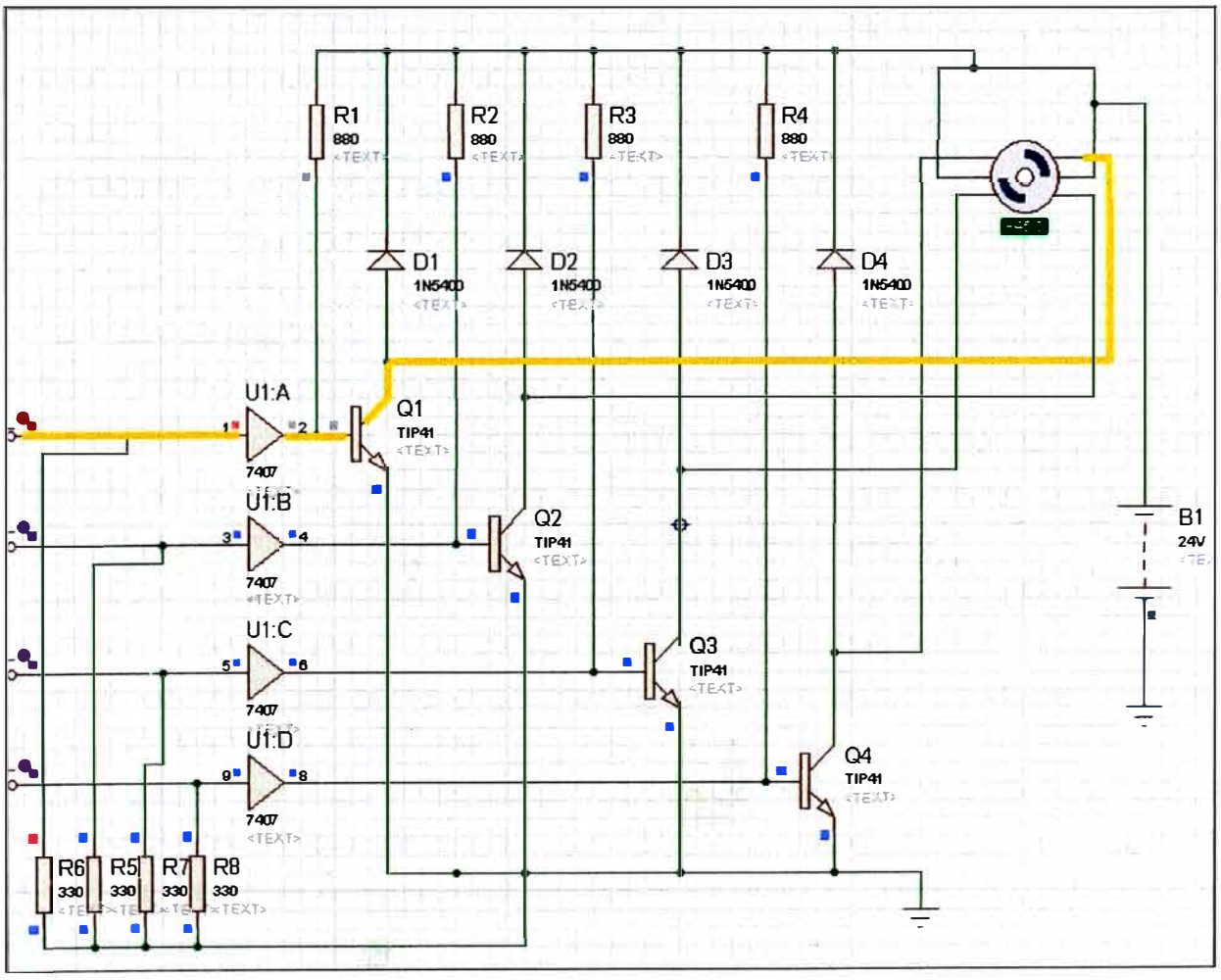


Figura 3.16 Circuito de potencia

Las tierras de las fuentes que alimentan el circuito de control (5V) y el de potencia (24V) deben estar unidas, esto con la finalidad de que cuando las líneas de control se encuentren en cero lógico, la corriente suministrada por la fuente de 24V encuentre un camino que le permita retornar a la misma a través de la conexión a tierra generada por este estado lógico.

La ventaja de este circuito es que los transistores jamás operarán con corrientes provenientes del circuito de control; éstas siempre serán proporcionadas por las resistencias. No existe riesgo de que el circuito de control (la parte más sensible del circuito a causa de la presencia del microcontrolador) sufra algún daño causado por cualquier incidente (cortos, sobrecorrientes o contracorrientes en el circuito de potencia.

Para suministrar la corriente nominal al motor y así obtener un torque ideal, se debe proporcionar la corriente nominal del motor en cada bobinado. Cuando los transistores entran en región de saturación, la corriente de colector debe ser igual a la requerida para la operación del motor.

Para el ejemplo las resistencias han sido calculadas para una corriente nominal de 1.4 A., las resistencias se pueden calcular de la siguiente forma: corriente nominal del motor i_c =1.4; β=50 ; Voltaje base-emisor en saturación=2V. De la teoría de electrónica análoga, para todo transistor BJT se debe cumplir:

$i_c = \beta i_b$

Recorriendo la malla de base a emisor para los transistores:

$-24V + i_b R_b + V_{be\text{-}sat} = 0$; donde $V_{be\text{-}sat} = 2V$;

$Rb = (24V-2V) / i_b \rightarrow$

$Rb = (24V-2V) / (i_c/\beta) \rightarrow$

$Rb = (24V-2V) / (1.4/50) = 785.71$ ohms

La resistencia comercial más cercana a esta cantidad es de 820Ω. Con este valor obligamos al transistor a entrar en región de saturación cuando la línea de control se encuentre en alta impedancia, accionando la bobina correspondiente.

Los diodos están conectados en modo de contracorriente. Estos cumplen la finalidad de proveer un camino por el cual pueda circular la corriente presente en los elementos inductivos del motor una vez se desconecte alguna bobina, evitando corrientes inversas que puedan destruir los transistores.

En el diseño se utilizan los diodos semiconductores 1N5400, los cuales son usados en aplicaciones de rectificación con una corriente máxima directa de 3A.

3.5 Características de componentes usados

En esta sección se describen las características más resaltantes de los dispositivos utilizados.

3.5.1 DM74LS126A-Quad 3-STATE Buffer

Dispositivo que contiene cuatro compuertas independientes las cuales realizan una unción no inversora de buffer de corrientes. La salida tiene la característica 3-state. Cuando es habilitada, la salida exhibe características de baja impedancia de una salida LS estándar con la capacidad adicional de permitir alimentar un bus de líneas sin resistores externos.

Cuando está deshabilitada, las salidas están en alta impedancia. De esta manera la salida no actúa ni como carga ni como driver. Para minimizar la posibilidad que dos salidas intenten tomar un bus común, el tiempo de deshabilitación es menor que el tiempo de habilitación de las salidas.

Las condiciones de operación recomendadas son mostradas en la Tabla 3.7. La funcionalidad de la misma en la Tabla 3.8.

Tabla 3.7 Condiciones de operación recomendadas

Símbolo	Parámetro	Min	Nominal	Máximo	Unidades
VCC	Voltaje fuente	4.75	5	5.25	V
VIH	Voltaje entrada Alta				V
VIL	Voltaje entrada Baja			0.8	V
IOH	Voltaje salida Alta			-2.6	mA
IOL	Voltaje salida Baja			24	mA
TA	Temperatura operación	0		70	°C

Tabla 3.8 Funcionalidades

Entrada		Salida
A	C	Y
L	H	L
H	H	H
X	L	Hi-Z

Donde:

- H = Nivel lógico alto
- L = Nivel lógico bajo
- X = alto o bajo
- Hi-Z = 3-STATE (salidas deshabilitadas)

3.5.2 SN74LS07- Driver en colector abierto

Posee seis buffers/drivers con características de alto voltaje en salidas de colector abierto para proporcionar alta corriente. Se caracterizan por usar entradas TTL. Este dispositivo tiene una tasa de salida de voltaje de 30V. La máxima corriente de drenado es de 30 mA para el SN54LS07 y de 40mA para el SN74LS07.Estos circuitos son compatibles con la mayoría de familias TTL. La potencia de disipación es 140 mW con promedio de teiempo de retardo de propagación de 12 ns.

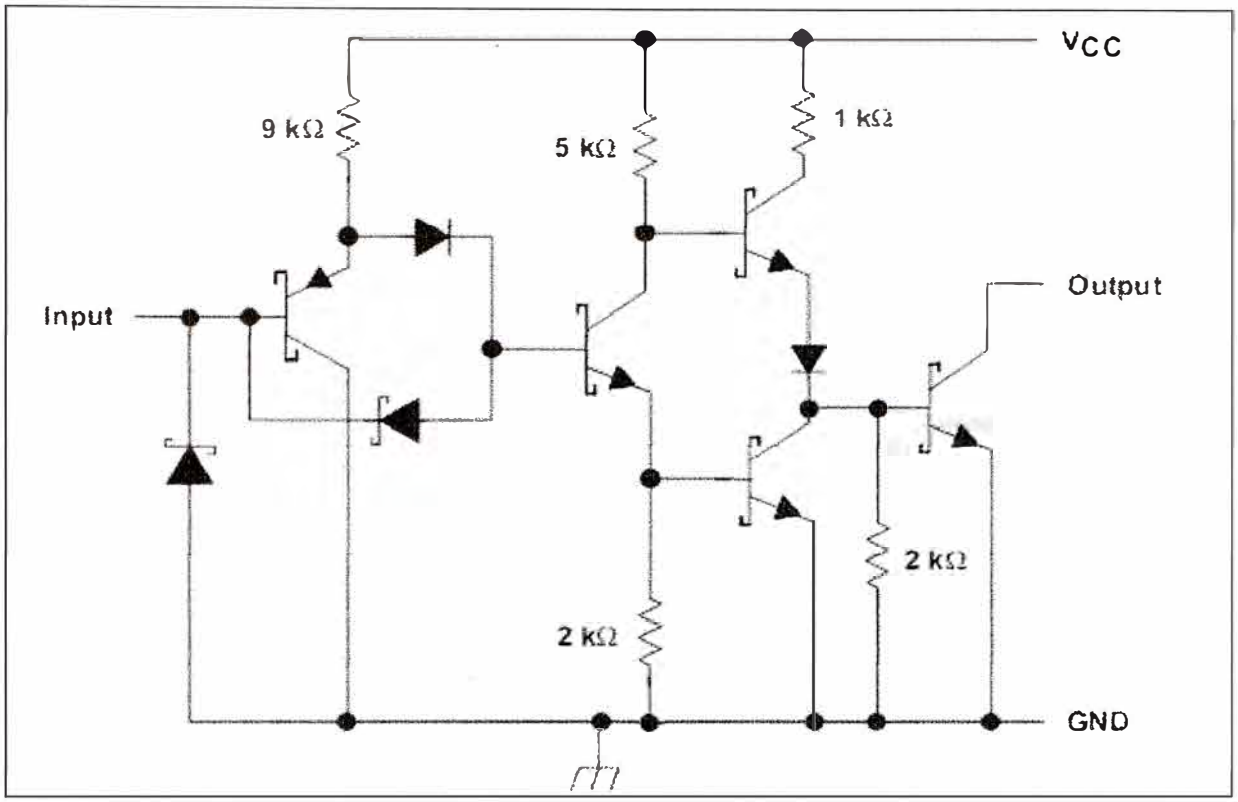


Figura 3.17 Diagrama esquemático del 7407

3.5.3 TIP41- Transistor de potencia NPN

La Tabla 3.9 muestra las características de este transistor de potencia.

Tabla 3.9 Características del transistor de potencia

	Símbolo	Valor	Unidad
Voltaje colector-base	V_{CBO}	80	V
Voltaje colector-emisor	V_{CEO}	40	V
Voltaje base-emisor	V_{EBO}	5	V
Corriente de colector	I_C	6	A
Corriente de colector pico	I_{CM}	10	A
Corriente de base	I_B	3	A
Potencia de disipación	Ptot	65	W
Temperatura de operación	Tj	-65 to +150	°C

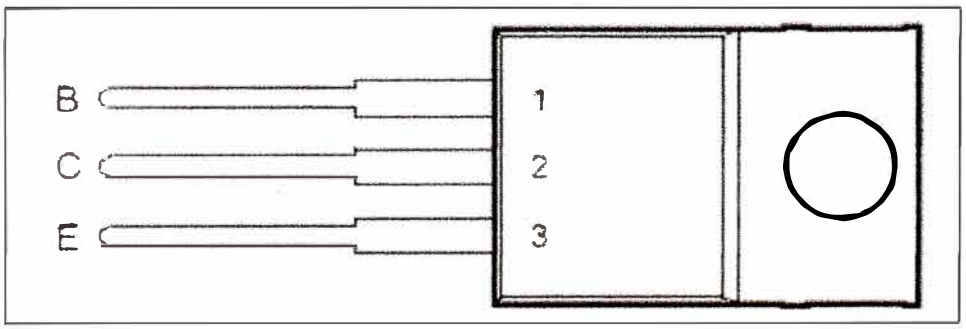


Figura 3.18 Vista superior

3.5.4 1N5400- Diodo de 3 amperios

Sus características son mostradas en la Tabla 3.10.

Tabla 3.10 1N5400

PE LUMBER		
Máximo pico de voltaje inverso	50	V
Máximo voltaje RMS	35	V
Máximo voltaje DC de bloqueo	50	V
Máxima corriente directa promedio de rectificación	3.0	A
Corriente de pico, 8.3 ms oinda sinusoidal medio ciclo	200	A
Máximo voltaje directo instantáneo a 3.0A	0.95	V
Capacitancia de junción	40.0	Pf
Temperatura de operación	-65 a 175	°C

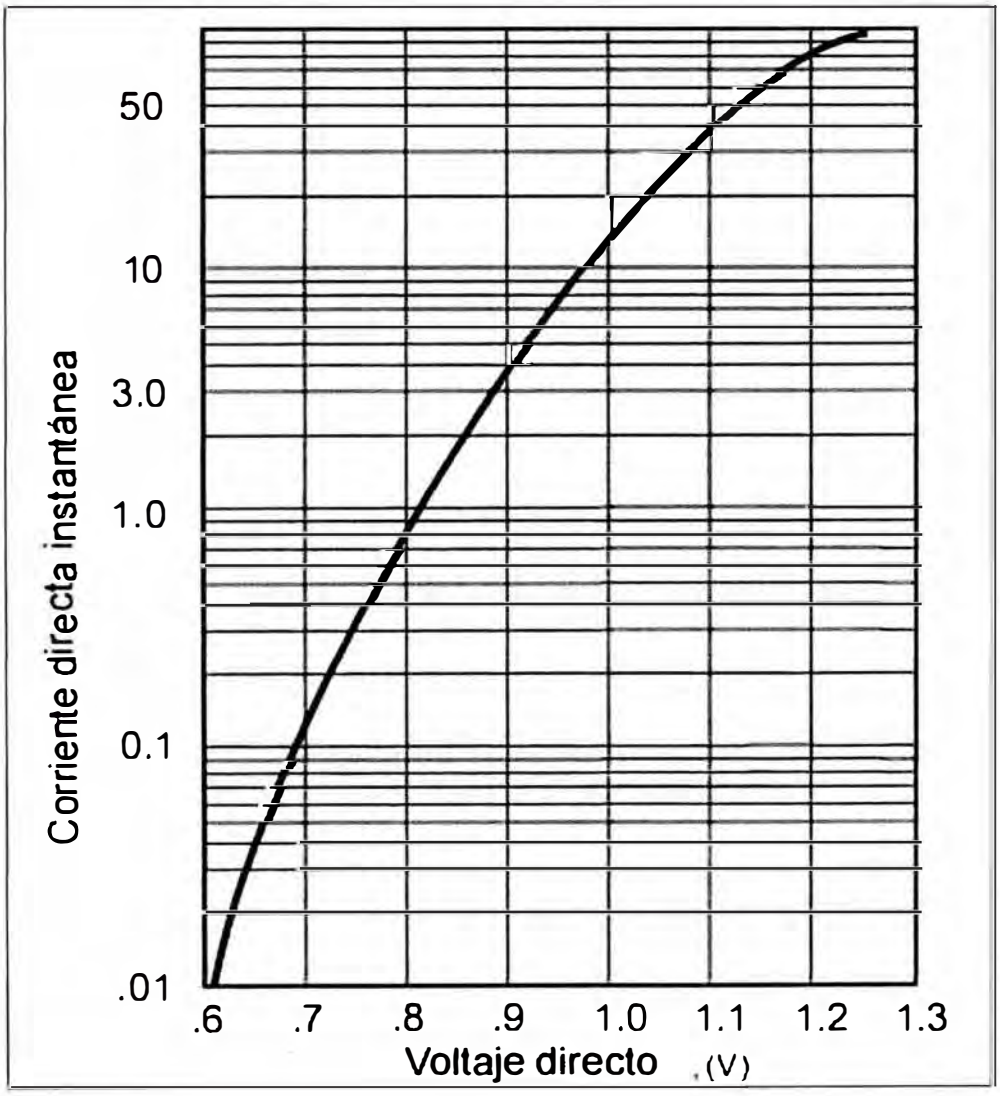


Figura 3.19 Características típicas en directo

3.5.5 Motor 17PM-K342

El modelo utilizado para los cálculos del torque fuerte es el 17PM-K342 de la marca Minebea (Figura 3.20), el cual corresponde a un motor unipolar de voltaje nominal de 24V, y una corriente de 1,4 A. (Ver Tabla 3.11)

El principio de funcionamiento de este tipo de motores consiste en alimentar de manera secuencial cada uno de los cuatro bobinados que presenta en su interior. El modelo elegido posee cinco terminales, uno de los cuales corresponde al común y va conectado a +24V. la idea es mantener uno de los cuatro terminales restantes conectado a la tierra del circuito, con los otros tres alimentados con +24V, realizando una rotación ordenada de la conexión a cero voltios. Por cada cambio en el terminal conectado a tierra, se presentara en el motor un giro o paso, el cual, para el modelo seleccionado, corresponde a 1.8°. En la tabla 3.12 se muestra la configuración de alimentación necesaria para obtener este modo de operación.

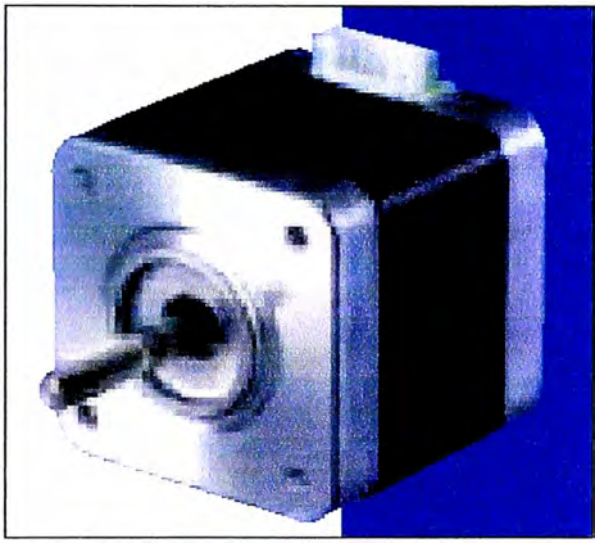
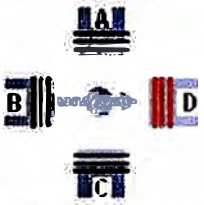


Figura 3.20 17PM-K342

Tabla 3.11 Características del 17PM-K342

Parámetro	Valores
Ángulo de paso	1.8 grados
Secuencia	UNI-POLAR
Corriente	1.4 A
Resistencia	2.0 Ohmios
Torque estático	250 mNm
Inductancia	2.5 mH
Inercia del rotor	50 $g.cm^2$
Torque de parada	11.3 mNm
Masa	250 g

Tabla 3.12. Secuencia de alimentación en la configuración.

PASO	Bobina A	Bobina B	Bobina C	Bobina D	
1	ON	OFF	OFF	OFF	
2	OFF	ON	OFF	OFF	
3	OFF	OFF	ON	OFF	
4	OFF	OFF	OFF	ON	

CONCLUSIONES Y RECOMENDACIONES

1. Con el caso de estudio presentado se ha demostrado la modularidad de los PICs para diversas aplicaciones.
2. Con el uso del simulador se ha podido evaluar la interactividad de los microcontroladores con los circuitos externos permitiendo rediseñar el proyecto ante nuevos requerimientos de funcionalidad a nivel de código, así cómo a nivel de potencia.
3. Es recomendable establecer los requerimientos del proyecto a implementar, dimensionando al microcontrolador de acuerdo a ellos.
4. Antes de aplicar el código se debe establecer el diagrama de flujo de la funcionalidad a fin de establecer las mejores alternativas para cumplir con la filosofía del proyecto.
5. Se deben seguir las indicaciones y valores recomendados por los fabricantes así cómo realizar la consulta respectiva a los documentos técnicos de cada dispositivo para así lograr optimizar su utilización.
6. Los elementos con carga reactiva pueden generar ruido por lo que se deben tomar las previsiones a nivel circuital para evitar los errores propios de estos sistemas.
7. El uso de simuladores (PROTEUS) no asegura un funcionamiento real ya que este está supeditado a factores físicos (capacidades e inductancias parásitas, etc.).
8. Se debe tener precaución en la manipulación del microcontrolador cunado se encuentra energizado, así mismo se debe garantizar las características eléctricas del dispositivo.
9. Se debe proporcionar la adecuada fuente de alimentación para los motores paso a paso; una fuente de 5 v y 20 A aisladas de la fuente de control debe ser suficiente.

ANEXO A
GLOSARIO DE TÉRMINOS

BSR	Register Selec Bank
CAD	Diseño asistido por computador.
CAN	Controller Area Network
CMOS	Complementary Metal Oxide Semiconductor
CP	Code Protect
GPR	registros de propósito general
HMI	interfaz hombre máquina
IDE	Integrated Development Environment
LIFO	Last In First Out
LVD	Low Voltage Detect
OST	Timer de arranque del oscilador
PaP	Motor paso a paso
PIC	Peripheral Interface Controller
POR	Power on Reset
PWM	Pulse Wide Modulation
PWRT	Timer de encendido
RISC	Reduced instruction set computer
SFR	registro de propósito especial
UART	Universal Asyncronous Reception Transmision)
USB	Universal Serial Bus
SPICE	Simulation Program with Integrated Circuits Emphasis

BIBLIOGRAFÍA

1. PIC12C5XX. http://ww1.microchip.com/downloads/en/DeviceDoc/40139e.pdf
2. PIC16C5XX. http://ww1.microchip.com/downloads/en/DeviceDoc/30453d.pdf
3. PIC16F877. http://ww1.microchip.com/downloads/en/DeviceDoc/41452A.pdf
4. PIC18C4X2. http://ww1.microchip.com/downloads/en/DeviceDoc/39026c.pdf
5. MPLAB IDE. Diversos catálogos http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469&part=SW007002
6. Spice. http://newton.ex.ac.uk/teaching/CDHW/Electronics2/Spice3Quickstart.html
7. Proteus de LabCenter. http://www.labcenter.com/products/pcb_overview.cfm
8. Multisim. http://www.ni.com/multisim/.
9. PSpice. http://materias.fi.uba.ar/6625/TPs/Tutoriales/9_1Tutorial%20SPICE.pdf
10. PIC16F877A. http://ww1.microchip.com/downloads/en/DeviceDoc/39582b.pdf
11. PICmicro® Mid-Range MCU Family Reference Manual (DS33023) http://ww1.microchip.com/downloads/en/DeviceDoc/33023a.pdf
12. DM74LS126A http://ltodi.est.ips.pt/lab-dee-et/datasheets/TTL/74126.pdf, Buffer 3-State
13. 74LS07 http://www.datasheetcatalog.com/datasheets_pdf/7/4/L/S/74LS07.shtml Buffer -Driver con colector abierto
14. Paul Acarnley , "Stepping Motors, a guide to theory and practice", 4ta edición, publicado por "The Institution of Egineering and Technology", London, 2007.
15. Javier Colmenares Apitz ,"Motores paso a paso", 2009, http://www.monografias.com/trabajos17/motor-paso-a-paso/motor-paso-a-paso.shtml
16. Irlenys Tersek Rodríguez, "Motores Paso a Paso. Funcionamiento y Control a Través de un PC", http://www.monografias.com/trabajos24/motores/motores.shtml
17. Dogan Ibrahim, "PIC Basic Projects: 30 Projects using PIC BASIC and PIC BASIC PRO", 2001.
18. Dogan Ibrahim, "Advanced PIC Microcontroller Projects in C: From USB to RTOS with the PIC 18F Series", 2008.
19. Minebea Stepping motor model 17PM-K data sheet, Minebea Co., Ltd, 2004.
20. Stepping Motor, Step Motor Sequence Tutorial http://www.8051projects.net/stepper-motor-interfacing/step-sequence.php