UNIVERSIDAD VERACRUZANA

Instituto de Ingeniería

"Controlador de posición para un motor que acciona una válvula en forma remota"

TESIS

QUE PARA OBTENER EL GRADO DE

MAESTRO EN

INGENIERÍA ELÉCTRICA OPCION CONTROL

Presenta

Carlos Roberto González Escarpeta

ASESOR

M.I. Alberto P. Lorandi Medina

H. VERACRUZ, VER. MARZO 2002.

UNIVERSIDAD VERACRUZANA
INSTITUTO DE INGENIERIA

H. Veracruz, Ver., a 7 de Marzo del 2002
DI044/02

Al Candidato al Grado:
Ing. Carlos Roberto González Escarpeta
P r e s e n t e

En atención a su solicitud relativa, me es grato transcribir a Usted a continuación el tema que aprobado por esta Dirección propuso el M.I. Alberto P. Lorandi Medina, para que lo desarrolle como tesis, para obtener el Grado de Maestro en Ingeniería Eléctrica Opción Control.

TEMA:

"CONTROLADOR DE POSICIÓN PARA UN MOTOR QUE ACCIONA UNA VÁLVULA EN FORMA REMOTA"

Introducción
I .- Controladores
II .- Arquitectura del Microcontrolador PIC16F84
III .- Adquisición y Procesamiento de Variables
IV .- Lectura e Interpretación de Variables
Conclusión

Sin otro particular, me es grato reiterarle la seguridad de mi más atenta y distinguida consideración.

A t e n t a m e n t e
"Lis de Veracruz: Arte, Ciencia, Luz"

B. Peña P.

Dr. Bonifacio C. A. Peña Pardo
Director

BPP/apm*.

Dedicatorias

- A **Dios** quien ha hecho posible este momento, mi gratitud y respeto por siempre.
- A mi complemento, mi amada esposa **Blanca Nieves**, por su apoyo y comprensión para mi desarrollo profesional y personal.
- A mis padres **Mario Jesús y Reyna**, a quienes amo y admiro por su gran labor realizada por la educación de sus hijos.
- A mis hermanos **Mario, Oscar, Jessica, Selene y José**, con amor
- A mi abuelo **Jesús** (Q.P.D.), por ser mi guía; el ser humano que mayor influencia ha tenido en mi vida.....un ejemplo a seguir.
- A mis Abuelas **Juana, Cristina y Agustina**, quienes siempre han entregado cariño y comprensión a mi vida

Agradecimientos

- A mi honorable Jurado, por su valiosa contribución en el desarrollo del trabajo profesional:
  M.I. Alberto Pedro Lorandi Medina
  M.C. Enrique Ladrón de Guevara Durán
  M.C. Evaristo Hernández Marcelis
- **Al Instituto de Ingeniería**, mi casa de estudios de posgrado.
- **Al Instituto Tecnológico de Veracruz**, por las facilidades y apoyo que me otorgó para la realización de mi Tesis.
- A los Ingenieros que tuvieron aportaciones durante el desarrollo del trabajo:
  Martha G. González Rogel
  Arturo Hermida Huerta
  Mario Dominguez Carballo
  Guillermo Hermida Saba
- A todas las personas e instituciones que de alguna forma intervinieron para la obtención de mi objetivo.

INDICE

INTRODUCCION

El control ha jugado un papel vital en el avance de la ciencia y de la ingeniería. Este control es de gran importancia en el control numérico de las máquinas y herramientas en las industrias manufactureras. También resulta esencial en operaciones industriales como el control de presión, temperatura, humedad y viscosidad, y flujo en las industrias de transformación.

Como los avances en la teoría y práctica del control, brindan medios para lograr el funcionamiento óptimo de sistemas dinámicos, mejorar la productividad, liberar la acción de muchas operaciones manuales rutinarias y repetitivas.

La ingeniería de control se dedica principalmente al estudio y análisis de los sistemas de control. Los sistemas de control están diseñados para realizar una acción, ya que son una combinación de componentes que actúan conjuntamente y cumplen determinado objetivo. Un sistema no está limitado a objetos físicos. Los sistemas de control en una clasificación general se dividen en Sistema de Control de Lazo Abierto y Sistemas de Control de Lazo Cerrado.

Analogías:

Considerando que cuando se traslada por cualquier medio de un lugar a otro, nuestra mente tiene presente algunas variables – tiempo, distancia, color, mensajes, etc. – que al ser identificadas sabrá que ha llegado a su destino y procederá a detener nuestra acción de translado. Si esto lo explicamos de forma analítica, podemos decir que para dirigirse de un punto θ_0 a un punto θ_1, se requiere recorrer una distancia igual a θ_1-θ_0, si a esta diferencia le denominamos error (e) podemos deducir que entre mayor sea e más alejado se está del destino y que cuando e tiende a cero, se estará más cerca del destino; esto es:

$$\theta_1 = \theta_0 + e$$

Bajo este último concepto, tienen que operar los sistemas de control, ya que éstos carecen de raciocinio, entonces un sistema de control de lazo cerrado es el que realiza principalmente las siguientes acciones: medir una variable de proceso que será controlada (temperatura, presión, razón de flujo de fluido, concentración química, humedad, viscosidad, posición mecánica, velocidad mecánica, etc.), y se retroalimenta a un comparador - el comparador puede ser mecánico, eléctrico o neumático – éste lleva a cabo una comparación entre el valor medio de la variable y el punto de ajuste, que representa el valor deseado de la variable.

El comparador que realiza la diferencia entre el valor real de la salida del sistema con la entrada de referencia (valor deseado), determina el error, y produce una señal que al pasar por un controlador, reducirá el error a cero, o a un valor más pequeño.

La forma como el sistema produce la señal de control, se denomina **acción de control**. Un buen sistema de lazo cerrado reduce a cero, o casi, la señal de error. La diferencia final entre el valor medido y valor actual que el sistema permite (que ya no puede corregir) es llamado *offset* o *nivel*.

Es posible diseñar sistemas con un offset bajo y una alta velocidad de respuesta, pero a veces tiende a ser inestables. Inestable significa que el sistema propicia variaciones violentas y grandes en el valor de la variable controlada a medida que "busca" la salida deseada del controlador. Esto ocurre porque el sistema reacciona de manera excesiva a los errores, causando por tanto un error aún mayor en la dirección opuesta. Por lo que sistema oscila. Las oscilaciones generalmente acaban por desaparecer, y el sistema se estabiliza en el valor correcto de variable de control. Mientras tanto, el proceso ha estado fuera de control, y podrían darse consecuencias.

Como puede verse entonces, un buen sistema es aquel que es estable. Debido a las características de los sistemas, será diferente el tipo de control que requerirá, para así poder propiciar una estabilidad.

En función de lo descrito se puede entonces considerar factible el diseño de un CONTROLADOR DE POSICIÓN PARA UN MOTOR QUE ACCIONE UNA VÁLVULA EN FORMA REMOTA.

En la presente tesina se demostró este proceso auxiliándose de un microcontrolador que operaró por medio de un algoritmo PID (proporcional integral y derivativo) para realizar en control en lazo cerrado.

Las nuevas generaciones de microcontroladores como lo son los PIC 16F84 presentan fabulosas ventajas respecto a su tamaño programación y velocidad de procesamiento por citar un ejemplo la velocidad de procesamiento es 20 veces mayor a la de microcontrolador 68HC11 y su forma de programación es más sencilla con tan sólo 33 instrucciones, con esto se puede visualizar que en muchos de los procesos modernos se encuentran aplicaciones con este tipo de microcontroladores, así también se cuenta actualmente con una gran cantidad de simuladores tanto en microcontroladores como en controles PID´s, que nos permiten simular sistemas de control retroalimentados y manipular las variables a través del software, hasta encontrar las óptimas.

Se desea entonces controlar el desplazamiento angular de un motor de DC (corriente continua) que accionará un actuador de una válvula y la señal de control será dada en forma remota desde un lugar seguro para el operador.

El mecanismo de control se ubicó cerca de la válvula y por medio de un cableado sencillo hacia una computadora se logró su control en forma remota. La variable se envió en forma serial con instrucciones de ocho bits que indican el ángulo de desplazamiento del motor.

El motor es de corriente directa de 12 Volts y uan corriente máxima de 0.5 Ampers que se alimenta a través de un circuito regulador tipo puente H con Mosfet´s (transistores de efecto de campo) de potencia. El regualdor recibe un control PWM (modulación por ancho de pulso) que se genera por un microcontrolador PIC16F84.

El motor acciona a la válvula y el desplazamiento de ésta fue medido en grados, el motor cuenta con un sensor de corriente que desactiva el sistema si se exceden los parametros de seguridad.

La forma en que se desarrolló el presente trabajo profesional, se dio con la evaluación y puesta en marcha de cuatro etapas.

1ª Interfaces de comunicación
2ª Etapa de procesamiento
3ª Interfaz de potencia
4ª Protección

En el primer capítulo se da a conocer el marco teórico sobre controladores PID, en el segundo capítulo se explica la arquitectura del microprocesador utilizado y en el resto de los capitulos se expone el desarrollo de las cuatro etapas mencionadas, así como sus pruebas, resultados y conclusiones.

CAPITULO I
CONTROLADORES

1.1 MODOS DE CONTROL

Un controlador lineal puede describirse simplemente como un dispositivo que contiene componentes como sumadores, diferenciadores, integradores, etc. Dependiendo de cuál de estos componentes se usan y cuáles deben de ser los valores de sus parámetros; será el modo de control que se efectúe. Por ejemplo uno de los controladores más comunes en la práctica son los PID y el problema de diseño consiste en determinar los valores de las constante Kp, Ki y Kd.

1.2 MODOS DE CONTROL EN LOS SISTEMAS INDUSTRIALES EN LAZO CERRADO.

La manera como reacciona un controlador a una señal de error es un indicación del modo de control. De los cuales se encuentran clasificados en cuatro básicos.

1.- Controladores de dos posiciones, o intermitentes (encendido o pagado), on/off.
2.- Controladores Proporcionales.
3.- Controladores Proporcional- Integral.
4.- *Controladores tipo Proporcional- Integral- Derivativo.*

La mayoría de los controladores analógicos industriales utilizan electricidad o algún fluido, como aceite o aire a presión, a modo de fuentes de potencia. Los controladores analógicos también se pueden clasificar según el tipo de potencia que utilizan en su operación; como neumáticos, hidráulicos o electrónicos. La clase de controlador a usar se decide en base a la naturaleza del sistema y las condiciones de operación, incluyendo consideraciones tales como seguridad, costo-disponibilidad, confiabilidad, exactitud, peso y tamaño.

1.2.1 Control encendido apagado (on/off).

En el modo de control de encendido-apagado, el dispositivo corrector final sólo tiene dos posiciones, o estados de operación. Si la señal de error es positiva, el controlador envía al dispositivo corrector a una de sus dos posiciones. En este tipo de controlador no existen puntos medio de la variable controlada o es encendida o apagada, por lo que para variables que requieran un todo y nada, es muy adecuada.

1.2.2 Control proporcional (P)

El dispositivo corrector final tiene una rango continuo de posiciones posibles. La posición exacta que toma es proporcional a la señal de error. En otras palabras, la salida del bloque controlador es proporcional a su entrada.

Este tipo de controlador responde a la cantidad de error. Entre más grande sea el error, más drástica es la acción de corrección.

La palabra proporcional es aplicada correctamente para la cantidad de corrección introducida esta en proporción con la cantidad de error.

En general un determinado cambio porcentual en el error ocasiona un correspondiente cambio porcentual en la posición en el cierre o apertura de una válvula.

La mayoría de los controladores proporcionales cuentan con una banda proporcional ajustable, la cual es el porcentaje del rango total del controlador en el cual el valor medido cambiaría en orden de producir que el dispositivo de corrección cambie en un 100% por lo cual esta comprendida entre poco porcentaje.

Se hablará de los efectos de control utilizando un modo de control proporcional. Como era de esperarse, elimina la oscilación permanente que siempre acompaña al control todo o nada. Podría haber una oscilación temporal hasta que el controlador se acomode en la posición final de control, pero eventualmente la oscilación desaparece si se ajusta apropiadamente la banda proporcional.

Sin embargo, si la banda proporcional se escoge muy pequeña de todos modos pueden ocurrir oscilaciones, porque una banda proporcional muy pequeña hace que el control proporcional opera de la misma forma que un control todo o nada.

Este control elimina la constante de oscilación alrededor del valor de referencia. Con esto proporciona un control más preciso , y reduce el desgaste y rotura de la válvula. La válvula de posición variable se mueve solamente cuando sucede algún tipo de disturbio al proceso, y siempre se mueve de una forma menos violenta que una válvula de acción rápida.

Por último se menciona como dato importante que para hacer uso exclusivo de un control proporcional es necesario tener cambios pequeños y lentos en la carga y una variación pequeña en el valor de referencia.

1.2.3 Control proporcional integral (PI).

Para las situaciones de procesos más comunes, en los cuales los cambios en la carga son grandes y rápidos, y el valor de referencia puede variar considerablemente, el modo de control proporcional integral es el más apto para utilizarse en estos casos.

Al control proporcional integral también se le denomina control proporcional-reposicionador.

En el control proporcional integral, la posición de la válvula de control esta determinada por dos factores:

a) La magnitud de la señal de error. La cual es la parte proporcional.
b) La integral con respecto al tiempo de la señal de error; en otras palabras, la magnitud del error multiplicada por el tiempo que ha permanecido. Esta va hacer la parte integral.

Dado que la válvula puede responder a la integral con respecto al tiempo de error, cualquier variación del error que resulte del control proporcional sólo es corregido eventualmente, a medida que pase el tiempo la parte de control proporcional posiciona la válvula en proporción al error que existe, por consiguiente ayuda a reducir el desbalance (*offset*). Entre mayor tiempo persista el error, mayor distancia se mueve la válvula. En algún momento, el error se reducirá a cero, y el movimiento de la válvula cesará. A medida que pasa el tiempo, la integral de tiempo del error ya no aumenta, debido a que el error ahora es cero.

Es importante mencionar que en la mayoría de los controladores industriales, la constante de tiempo de integración no se utiliza como referencia. En lugar de ello, se habla del reciproco de la constante de tiempo de integración. Esta variable se denomina razón de reposición. Como dato importante se tiene que cuando la razón de reposición es baja (constante de tiempo grande) la parte integral es lenta en producir el efecto buscado por el proceso. Cuando la razón de reposición es alta (constante de tiempo pequeña) la parte integral del control produce rápidamente el efecto buscado por el proceso.

El modo de control proporcional integral se ajusta a la mayoría de las situaciones de control. Puede controlar bastante bien grandes cambios en la carga y grandes variaciones en el valor de referencia, esto sin oscilaciones prolongadas, ni desbalances permanentes, y una rápida recuperación después de un disturbio.

1.2.4 Control proporcional integral derivativo (PID).

Aún cuando el control proporcional integral es adecuado para la mayoría de las situaciones. Hay algunos procesos que presentan problemas de control muy difíciles que no pueden manejarse por un control proporciona integral. Específicamente, aquí se tienen dos características de proceso que presentan dichos problemas difíciles de control para los cuales no es suficiente un control proporcional integral:

a) Cambios muy rápidos en la carga.
b) Retardos de tiempo grandes entre la aplicación de la acción correctora y la comparación de los resultados de dicha acción en la variables medida.

En los casos donde uno o cualquiera de estos dos problemas prevalezca, la solución puede ser un control proporcional integral derivativo. El término control derivativo se denomina también razón de control. En el control proporcional integral derivativo la acción correctora (la posición de la válvula) es determinada por tres factores:

a) La magnitud del error. Esta es la parte proporcional.
b) La integral con respecto al tiempo del error o la magnitud del error multiplicada por el tiempo que ha permanecido. Esta es la parte integral.
c) La razón de tiempo de cambio del error; un rápido cambio en el error produce una acción correctora. Esta es la parte derivativa.

En un sentido intuitivo, la parte derivativa del controlador intenta "mirar adelante" y prevé que el proceso sufrirá un gran cambio basándose en las medidas actuales. Es decir, si la variable medida está cambiando muy rápidamente, es seguro que tratará de cambiar en una gran cantidad. Siendo éste el caso, el controlador trata de "anticiparse" al proceso y aplica mayor acción correctora que la que aplicaría un control proporcional integral.

Sin embargo, si la banda proporcional se escoge muy pequeña, pueden ocurrir oscilaciones, porque una banda proporcional muy pequeña hace que el control proporcional opere de la misma forma que un control TODO o NADA.

Se puede ver que el modo de contro0l proporcional tiene una ventaja importante sobre el control TODO o NADA. Elimina la constante oscilación alrededor del valor de referencia. Con esto proporciona un control más preciso y reduce el desgaste y rotura de la válvula. La válvula de posición variable se mueve solamente cuando sucede algún tipo de disturbio en el proceso y siempre se mueve de una forma menos violenta que una válvula de acción rápida.

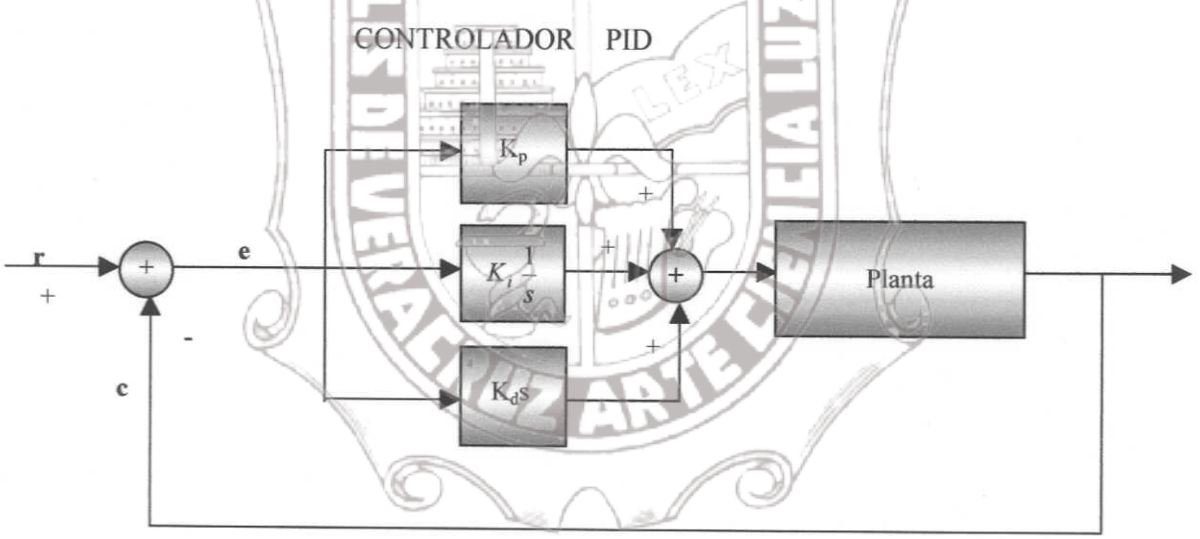


Figura 1.1 . Sistema de control PID

Dentro del funcionamiento del sistema de control, se consideró la variable de entrada ***r*** como POSD, la variable de retroalimentación ***c*** como POSR y la variable ***e*** como ERROR. Desarrollando con éstas el algoritmo principal del sistema PID dentro del controlador de posición; el cual se muestra a continuación.

1.2.5 Algoritmo del PID utilizado en el programa.

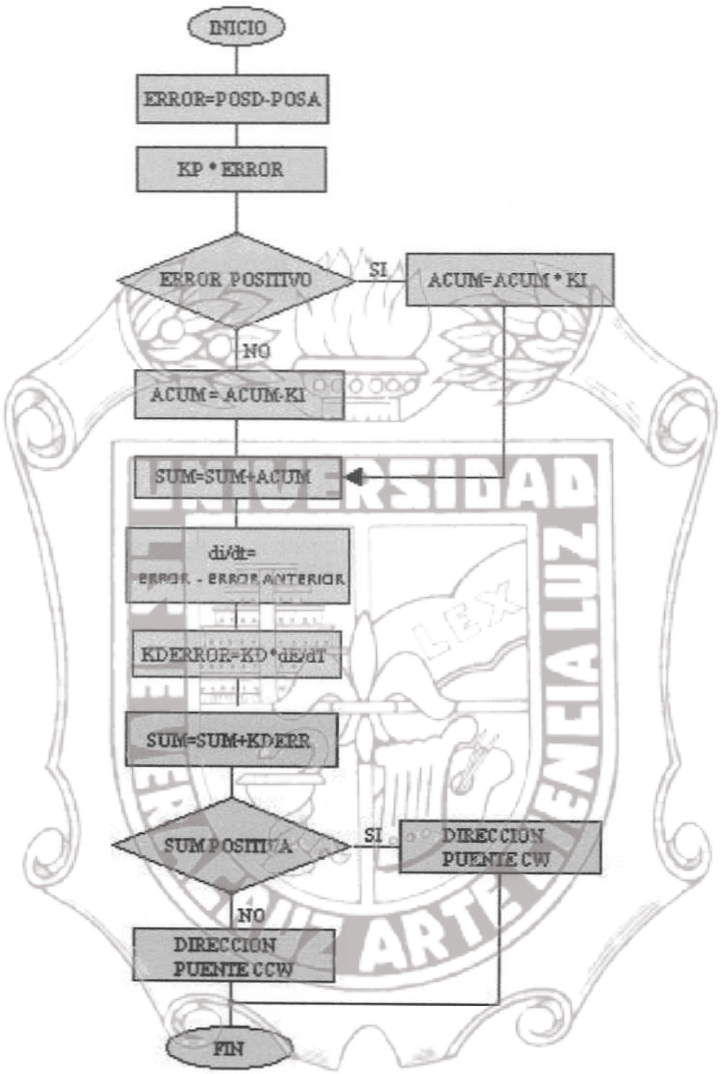


Fig. 1.2. Diagrama de flujo de la función del PID en el controlador de posición

El desarrollo del algoritmo mostrado en la figura anterior fue basado en la siguiente función:

$$\mathbf{G}(s) = K_p + K_i \frac{1}{s} + K_d s$$

BIBLIOGRAFÍA

- Kuo Benjamín C., ***Automatic Control System,*** Ed. Prentice Hall, New Jersey, 1993.
- M. M. Gupta, N. K. Sinha, ***Intelligent Control Systems, theory and applications***, IEEE PRESS.

CAPITULO II
ARQUITECTURA DEL MICROCONTROLADOR PIC16F84

2.1 ¿QUÉ ES UN MICROCONTROLADOR?

Es un circuito integrado programable que contiene todos los componentes de un computador. Se emplea para controlar el funcionamiento de una tarea determinada y, debido a su reducido tamaño, suele ir incorporado en el propio dispositivo que controla. El microcontrolador se puede definir como un sistema dedicado, en su memoria sólo recide un programa destinado a realizar una aplicación determinada; sus líneas entrada/salida soportan el conexionado de los sensores y actuadores del dispositivo a controlar. Una vez programado y configurado el microcontrolador sólo ejecutará la tarea asignada

2.1.1 Arquitectura General del PIC 16F84

Uno de los pilares en los que se basa la organización de los PIC es la **Arquitectura Harvard**. Con ella, el CPU accede de manera simultánea e independiente a la memoria de datos y a la de instrucciones. Este aislamiento entre datos e instrucciones permite que cada uno tenga el tamaño mas adecuado. Así, los datos tienen una longitud de 8 bits, mientras que las instrucciones la tienen de 14 bits. Figura 2.1.

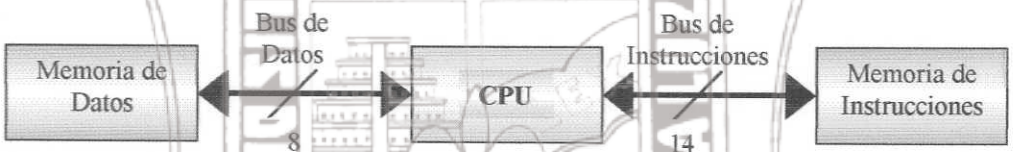


Figura 2.1. Arquitectura HARVARD.

Con la estructura segmentada se pueden realizar simultáneamente las dos fases en que se descompone cada instrucción. Al mismo tiempo que se está desarrollando la fase de ejecución de una instrucción se realiza la fase de búsqueda de la siguiente.

Otro de los recursos que propician el manejo intensivo de la arquitectura Harvard es el del "banco de registros", que participa de manera muy flexible en la ejecución de las instrucciones. Como se muestra en la figura 2.2, la ALU realiza sus operaciones lógico-aritméticas con dos operandos, uno que recibe desde el registro W, que hace las veces de Acumulador de los microprocesadores convencionales, y otro que puede provenir de cualquier registro interno. El resultado de la operación se puede depositar en cualquier registro. Esta funcionalidad da un carácter completamente ortogonal a las instrucciones, haciendo posible que los operandos fuente y destino estén ubicados en cualquier registro.

Los microcontroladores PIC reúnen todas las condiciones necesarias para pertenecer al grupo de procesadores RISC y que se citan a continuación.

1ª. Juego reducido de instrucciones

✓ La gama media de los PIC responde a un conjunto de 35 instrucciones.

2ª. Idéntico formato de las instrucciones

✓ Todas las instrucciones de la gama media tienen una longitud de código de 14 bits.

3ª. Ortogonalidad

✓ Las instrucciones pueden utilizar cualquier objeto como operando fuente o destino.

4ª. Ejecución de instrucciones simples en un ciclo

✓ Todas las instrucciones de los PIC se ejecutan en un ciclo, menos las de salto que tardan dos.

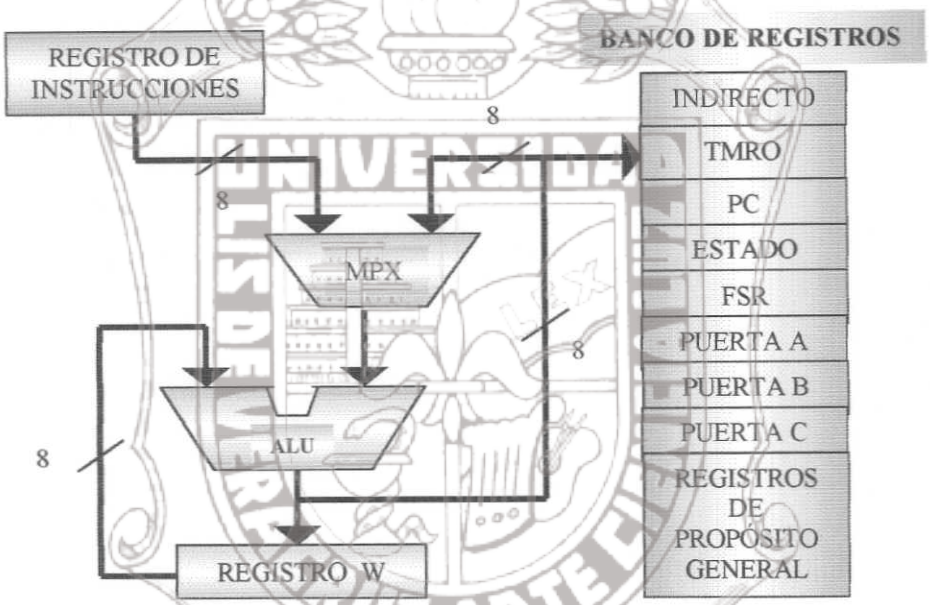


Figura 2.2. La ALU recibe los operandos desde el registro W y de cualquier otro del banco general de registros.

Las instrucciones disponen de tres modos de direccionamiento, directo, indirecto y relativo. El stack que guarda los retornos al programa principal sólo dispone de 8 niveles en los componentes de la gama media.

La tecnología CMOS con la que se fabrican los PIC permite que la tensión de alimentación pueda oscilar entre 2 y 6.25 V con un consumo reducido de potencia. A 5 V y 4 MHz., el consumo típico es menor que 2 mA , a 3 V y 32 kHz., es inferior a 15 µA. Cuando el microcontrolador funciona en "modo de reposo o espera" y con el temporizador Perro guardián (watch dog)* desactivado, la potencia necesaria es menor de 3 µA. Todas estas características dan la posibilidad a la alimentación de los sistemas con pilas convencionales.

* Temporizador utilizado por el microcontrolador, se describe posteriormente

La arquitectura interna del PIC16F84 se presenta en la figura 2.3 y consta de 7 bloques fundamentales.

1. Memoria de programa Flash de 1 K x 14 bits
2. Memoria de datos formada por dos áreas. Una RAM donde se alojan 22 registros de propósito específico (SFR) y 36 de propósito general (GPR), y otra del tipo EEPROM de 64 bytes.

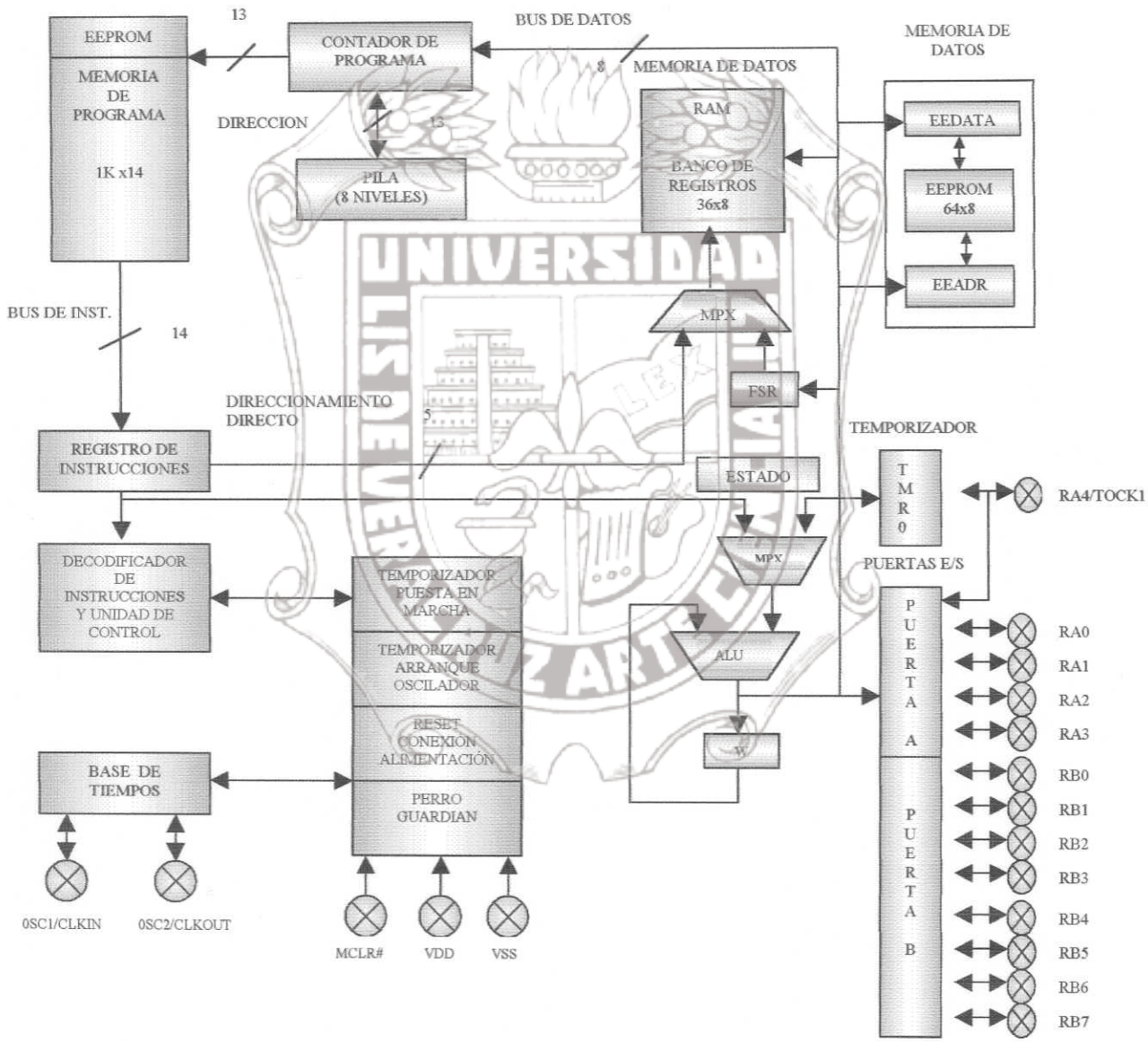


Figura 2.3 Arquitectura interna del PIC16F84

3. Camino de datos con una ALU de 8 bits y un registro de trabajo W del que normalmente recibe un operando y envía resultado. El otro operando puede provenir del bus de datos o del propio código de instrucciones (literal).

4. Diversos recursos conectados al bus de datos tales como puertas de entrada/salida temporizador TMR0, etc.
5. Base de tiempos y circuitos auxiliares.
6. Direccionamiento de la memoria del programa en base al contactor del programa ligado a una pila de 8 niveles de profundidad.
7. Direccionamiento directo e indirecto de la memoria RAM.

2.1.2 Memoria de Programa

La arquitectura de los PIC de la gama media admite un mapa de memoria de programa capaz de contener 8,192 instrucciones de 14 bits cada una. Este mapa se divide en páginas de 2,048 posiciones. Para direccionar 8 K posiciones se necesitan 13 bits, que es la longitud que tiene el Contador del Programa (PC). Sin embargo, el PIC16F84 sólo tiene implementadas 1 K posiciones, por lo que ignora los 3 bits de más peso del PC.

La principal novedad del PIC16F84 es de disponer de memoria FLASH para contener el programa, además de 64 bytes de memoria EEPROM para datos. Su gran ventaja (que le hace uno de los dispositivos más empleados en los laboratorios de diseño) es su fácil reprogramabilidad pues los dispositivos Flash son borrables eléctricamente y no requieren un proceso de borrado con rayos ultravioleta que retrasa y complica su nuevo uso. Las memorias Flash pueden ser regrabables cuantas veces se quiera directamente desde el grabador, de la misma forma que se graban.

2.1.3 El Contador del Programa y la Pila

El PC (Contador del programa) consta de 13 bits con los que se puede direccionar una memoria de código con una capacidad de hasta 8k palabras de 14 bits cada una. La memoria se organiza en paginas de 2k de tamaño.

Los PIC16X84 tienen 1 K palabras de 14 bits en la memoria de programa y aunque el PC dispone de 13 bits, en direccionamiento de la misma sólo los 10 de menos peso.

Cuando se escribe el Contador del Programa como resultado de una operación de la ALU, el byte de menos peso del PC se corresponde con el contenido del registro PCL ubicado en la posición 02h del banco 0. Los 5 bits de más peso del PC se corresponden con los 5 bits de menos peso del registro PCLATH en la posición 08h del banco 0. Los bits de más peso del PC sólo se pueden escribir a través del registro PCLATH.

En las instrucciones de salto relativo, el resultado de la misma afecta sólo a los 8 bits de menos peso del PC. Los 5 bits de mas peso se suministran desde PCLATH <4:0>. Figura 2.4. En las instrucciones GOTO y CALL los 11 bits de menos peso del PC se suministran desde el código OP (objeto). Los dos bits de más peso del PC se cargan con los bits <4:3> del registro PCLATH (Figura 2.4). Como la memoria de programa se organiza en páginas de 2k, la posición la seleccionan los 11 bits de menos peso, mientras que con los 2 bits de más peso

del PC se elige la página.Un reset pone a 0 todos los bits del PCL y PCLATH, obligando a que la dirección de reinicio sea 0000h.

La Pila es una zona aislada de las memorias de instrucciones y datos. Tiene una estructura LIFO (Ultimo en entrar primero en salir, -Last In First Out-), en la que el último valor guardado es el primero que sale. Tiene 8 niveles de profundidad cada uno con 13 bits.

Funciona como un buffer circular, de manera que el valor que se obtiene al realizar el noveno desempilado (pop) es igual al que se obtuvo primero.

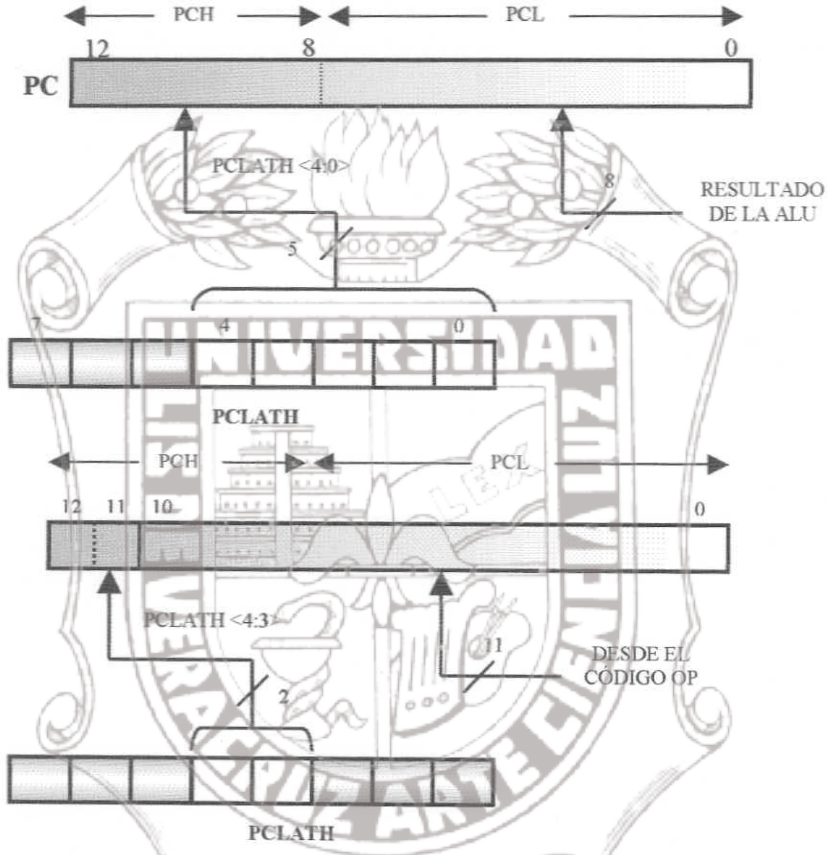


Figura 2.4. Carga del PC en situaciones diferentes

La instrucción CALL y las interrupciones originan la carga del contenido del CP en el nivel superior o *cima* de la pila. El contenido del nivel superior se saca de la pila al ejecutar las instrucciones RETURN, RETLW y RETFIE. El contenido del registro PCLATH no es afectado por la entrada o salida de información de la Pila.

2.1.4 Memoria de Datos RAM

La memoria de datos del PIC16F84 dispone de dos zonas diferentes:

1ª **Area de RAM estática o SRAM** , donde reside el Banco de Registros Específicos (SFR) y el banco de Registros de propósito General (GPR). El primer banco tiene 24 posiciones de tamaño de un byte, aunque dos de ellas no son operativas, y el segundo 36.

2ª **Area EEPROM** de 64 bytes donde, opcionalmente, se pueden almacenar datos que no se pierden al desconectar la alimentación.

La zona de memoria RAM se halla dividida en dos bancos (banco 0 y banco 1) de 128 bytes cada uno. En el PIC16F84 sólo se hallan fisicamente las 48 primeras posiciones de cada banco, de las cuales las 12 primeras están reservadas a los registros de Propósito Específico (SFR), que son los encargados del control del procesador y sus recursos. Algunos de dichos registros se hallan repetidos en la misma dirección de los dos bancos, para simplificar su acceso (INDF, ESTADO, FSR, PCLATH e INTCON).

La posición apuntada por la dirección 7H y la apuntada por la 87H no son operativas. Los 36 registros restantes de cada banco se destinan a Registros de Propósito General y en realidad solo son operativos los 36 del banco 0 porque los del banco 1 se mapean sobre el banco 0.(ver figura 2.5)

Para seleccionar el banco a acceder hay que manipular el bit 5 (RP0) del registro ESTADO. Si RP0 = 1 se accede al banco 1 y si RP0 = 0 se accede al banco 0. Tras un reset se accede automáticamente al banco 0. En el direccionamiento directo a los registros GPR se ignora el bit de más peso, que identifica el banco y sus direcciones están comprendidas entre el valor 0x0c y 0x2f en hexadecimal.

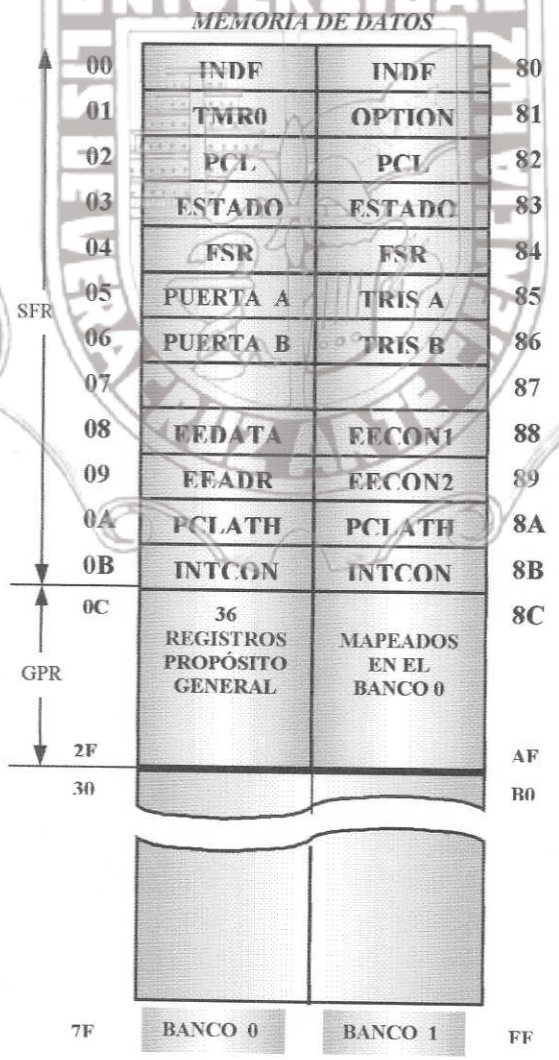


Figura 2.5 Memoria RAM

2.1.5 Direccionamiento de los Datos

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

1ª. Direccionamiento directo

Los siete bits de menos peso del código OP de la instrucción proporcionan la dirección de la posición de un banco. Los bits RP1 y RP0 del Registro de ESTADO <6:5>, seleccionan el banco. Figura 2.6.

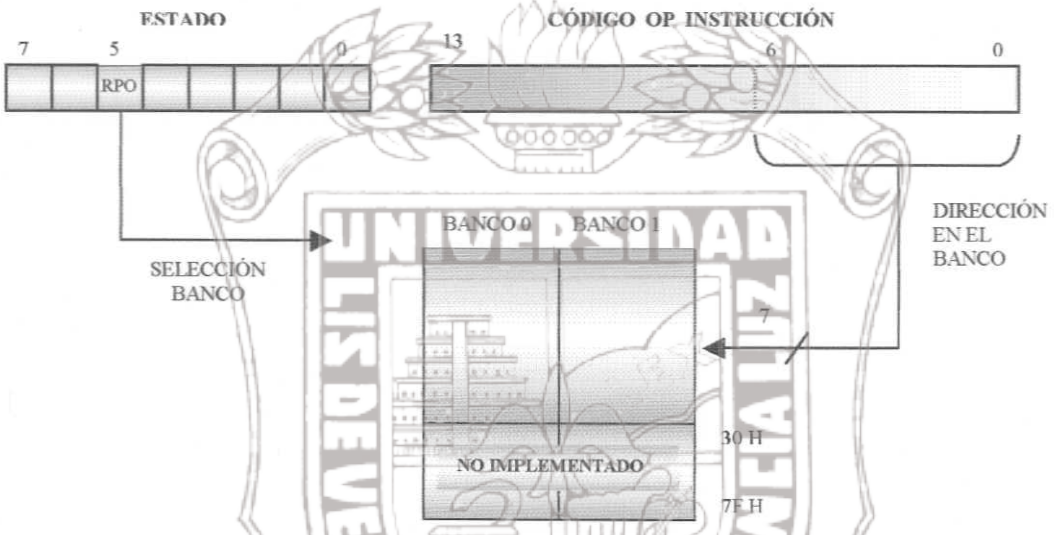


Figura 2.6. Direccionamiento Directo

2ª. Direccionamiento indirecto

En este caso el operando de la instrucción hace referencia al registro INDF, que ocupa la posición 0 del área de datos. Se accede a la posición que apunta el registro FSR, que se halla situado en la posición 4 del banco 0. Los 7 bits de menos peso de FSR seleccionan la posición y su bit de mas peso junto con el bit IRP del Registro de estado <7>, seleccionan el banco. Como sólo hay dos bancos en el PIC16F84 en este modo de direccionamiento, el bit IRP = 0 siempre. Figura 2.7.

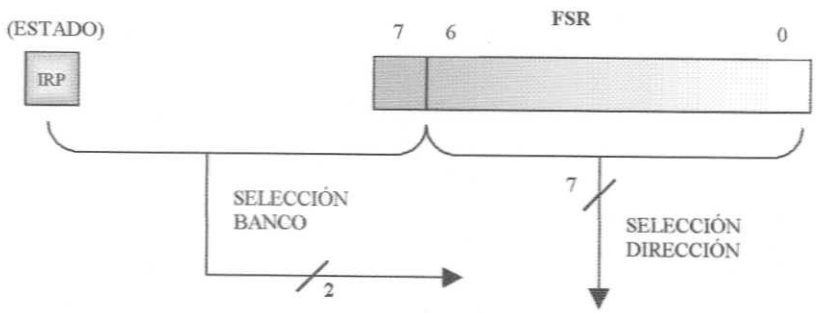


Figura 2.7 Direccionamiento Indirecto

2.1.6 El Registro de Estado

Es un registro de 8 bits en el que se reflejan circunstancias de interés sobre el estado del procesador. Ocupa la dirección 3 del banco 0 como del banco 1 de la memoria de datos RAM. Sus bits tienen tres misiones distintas.

1ª Se encargan de avisar las incidencias del resultado de la ALU (C, DC y Z).
2ª Indican el estado de Reset (TO# y PD#)
3ª Seleccionan el banco a acceder en la memoria de datos (IRP, RP0 y RP1).

En la figura 2.8 se muestra el diagrama de distribución de los bits del registro ESTADO. Los bits TO# y PD# indican el estado del procesador en algunas condiciones y no se pueden escribir. A continuación se describen los bits del registro ESTADO.

C: Acarreo

1: [^] Se a producido acarreo en el bit de más peso del resultado al ejecutar las instrucciones *addwf* y *addlw*.
0: [^] No se ha producido acarreo.

C también actúa como señalizador de llevada en el caso de la instrucción de resta, como *sublf* y *sublw*. En este caso la correspondencia es inversa (si vale 1 no hay llevada y si vale 0 sí).

DC: Acarreo en el 4° bit.

Igual significado que C pero refiriéndose al 4° bit. De interés en operaciones en BCD.

7 **ESTADO** 0

IRP	RP1	RP0	TO#	PD#	Z	DC	C
R/W	**R/W**	**R/W**	**R**	**R**	**R/W**	**R/W**	**R/W**

Figura. 2.8 Estructura interna del registro ESTADO.

Z: Cero

1: El resultado de una instrucción lógico-aritmética ha sido 0.
0: El resultado de una instrucción lógico-aritmética no ha sido 0.

[^] Valor que toma la localidad del registro en la posición indicada , se especifica así en lo sucesivo .

PD#: <<Power Down>>

1: Se pone a uno después de la conexión de la alimentación al microcontrolador o al ejecutar la instrucción *clrwdt*.
0: Se pone automáticamente a 0 mediante la ejecución de la instrucción *slepp*.

TO#:<<Time Out>>

1: Se pone a 1 después de la conexión de la alimentación o al ejecutarse las instrucciones *clrwdt* y *sleep*.
0: Se pone a 0 cuando se produce el desbordamiento del Perro Guardián (*Watchdog*).

RP1 -- RP0: Selección de banco en direccionamiento directo.

Como el PIC16F84 sólo tiene dos bancos, únicamente emplea el bit RP0, de forma que cuando vale 1 se accede al banco 1 y cuando vale 0 se accede al banco 0. Después de un reset RP0 = 0.

IRP: Selección del banco en direccionamiento indirecto

Este bit con el de más peso del registro FSR sirven para determinar el banco de la memoria de datos seleccionado. En el PIC16F84 al disponer de dos bancos no se usa este bit y debe programarse como 0.

2.1.7 Control de Tiempos

Una exigencia en las aplicaciones de control es la regulación estricta de los tiempos que duran las diversas acciones que realiza el sistema. El dispositivo típico destinado a gobernar los tiempos recibe el nombre de temporizador o "timer" y, básicamente, consiste en un contador ascendente o descendente que determina un tiempo entre el valor que se le carga y el momento en que se produce su desbordamiento o paso por 0. Figura 2.9.

En este caso se trata de un contador ascendente, que, una vez cargado con un valor, se incrementa al ritmo de los impulsos de reloj hasta que llega al desbordamiento.

Figura 2.9 Esquema simplificado de un temporizador.

En la gama media los microcontroladores PIC sólo disponen de dos temporizadores. Uno de ellos actúa como Principal y sobre él recae el control de tiempos de las operaciones del sistema. El otro recibe el nombre de Perro Guardián o "Watchdog" (WDT).

El Perro Guardián vigila que el programa no se "cuelgue" y dejen de ejecutarse las instrucciones secuenciales del mismo tal como lo ha previsto el diseñador. Para realizar esta labor de vigilancia, el Perro guardián da un paseo por el CPU cada cierto tiempo y comprueba si el programa se ejecuta normalmente; en caso contrario, por ejemplo si el control está detenido en un bucle infinito o a la espera de algún acontecimiento que no se produce, el Temporizador se activa y provoca un reset, reinicializando todo el sistema.

Tanto el Temporizador Principal, TMR0, como el Perro Guardián, a veces precisan controlar tiempos largos y aumentar la duración de los impulsos de reloj que les incrementan o decrementan. Para cubrir esta necesidad, se dispone de un circuito programable llamado Divisor de Frecuencia que divide la frecuencia utilizada por diversos rangos para poder realizar temporizaciones más largas. Para regular el comportamiento del Temporizador principal., el Perro Guardián y el Divisor de Frecuencia , se emplean algunos bits de la Palabra de configuración y del Registro de Opciones (OPTION).

El Divisor de Frecuencia puede aplicarse a uno de los dos temporizadores, al TMR0 o al WDT. Con el Temporizador Principal actúa en primer lugar, esto es que los impulsos pasan primero por el Divisor de Frecuencia y, una vez aumentada la duración de los últimos, se aplican a TMR0, actuando como Divisor Previo o "Prescaler". Con el Perro Guardián, el Divisor de Frecuencia actúa después ("Post-scaler").

El Divisor de Frecuencia puede actuar al ritmo de una señal externa aplicada sobre la terminal T0CKI, o bien, con la señal de reloj interna del microcontrolador CLKOUT, procedente del oscilador propio. Mediante algunos bits del Registro de Opciones y la Palabra de configuración se controla el trabajo del Divisor de Frecuencia sobre el TMR0 o el WDT.

2.1.8 El temporizador principal, TMR0

Se trata de un contador ascendente de 8 bits que puede actuar de dos formas.

a) Contador

Se le introducen los impulsos desde el exterior por el pin T0CKI. Su misión es "contar" el número de acontecimientos externos.

b) Temporizador

Trabaja y cuenta los impulsos de reloj del oscilador interno (CLKOUT), Se usa para determinar un tiempo fijo. Estos impulsos tienen una duración conocida que es la de un ciclo de instrucción cuya frecuencia es la cuarta parte del oscilador principal (Fosc/4).

El TMR0 se comporta como un registro de propósito especial ubicado en la posición 1 del área de datos. Puede ser leído y escrito al estar conectado directamente al bus de datos. Como se trata de un contador ascendente, conviene cargarle con el valor de los impulsos que se desean contar pero en forma de complemento a 2, Así, si se quieren contar cuatro impulsos de reloj se carga al TMR0 con el complemento a 2 de 4, lo que significa cargarle con -4. De

esta manera, con la llegada de cuatro impulsos se alcanza el valor 0, que determina el tiempo a controlar.

Para trabajar con TMR0 se pueden utilizar las siguientes fórmulas en el caso que los impulsos de reloj provengan del oscilador interno con un periodo de Tosc.

Temporización = 4 · Tosc · (Valor cargado en TMR0) · (Rango del Divisor)
Valor a cargar en TMR0 = (temporización / 4 · Tosc) · (Rango del Divisor)

Para conocer el estado en que va la cuenta del TMR0 se le puede leer en cualquier momento. Cuando se escribe un nuevo valor sobre TMR0 para iniciar una nueva temporización, el incremento del mismo se retrasa durante los dos ciclos de reloj posteriores.

En la figura 2.10 se ofrece el esquema de funcionamiento del Temporizador principal. Obsérvese que existe un bloque que retrasa dos ciclos y cuya misión consiste en sincronizar el momento del incremento producido por la señal T0CKI con el que producen los impulsos del reloj interno. Cuando no se usa el Divisor de frecuencia, la entrada de la señal de reloj externa es la misma que la salida de dicho Divisor.

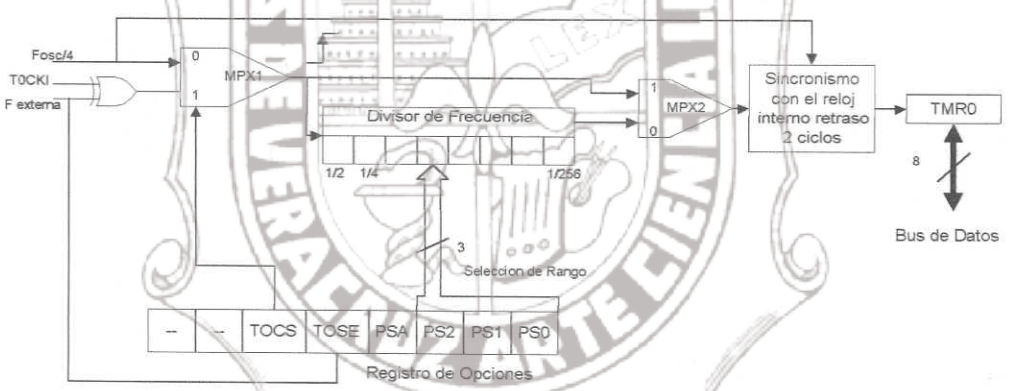


Figura 2.10 Esquema de funcionamiento del temporizador Principal TMR0.

2.1.9 El Registro de opciones (OPTION)

La misión principal de este registro es gobernar el comportamiento del Temporizador principal. Su bit T0CS (Timer 0 External Clock Edge Select) elige en el multiplexor MPX1 la procedencia de los impulsos de reloj, que pueden ser los del oscilador externo (Fosc/4) o los que se aplican desde el exterior por la terminal T0CKI. El bit. T0SE (timer 0 Clock Source Select) selecciona el tipo de flanco que es activo para la frecuencia externa. Si TOSE = 1, el flanco activo es el descendente y si T0SE = 0, es el ascendente.

Como se deduce de la figura 2.10, el bit PSA del Registro de opciones tiene la función de asignar el Divisor de frecuencia al TMR0 o al WDT. Si PSA = 0 la salida del visor se aplica al TMR0, pero si PSA = 1 el Divisor se destina al Perro guardián.

Finalmente, los 3 bits de menos peso del Registro de opciones (PSA2, PSA1 y A0) seleccionan el rango por el que el Divisor de frecuencia va a dividir los impulsos que se le apliquen. En la figura 2.11 se ofrece la distribución y asignación de los bits del Registro de opciones.

7			OPTION				0
RBPO#	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0

Figura 2.11 Distribución y asignación de los bits del Registro de opciones.

PS2:PS0 Valor con el que actúa el Divisor de frecuencia

PS2	PS1	PS0	Divisor del TMR0	Divisor del WDT
0	0	0	1:2	1:1
0	0	1	1:4	1:2
0	1	0	1:8	1:4
0	1	1	1:16	1:8
1	0	0	1:32	1:16
1	0	1	1:64	1:32
1	1	0	1:128	1:64
1	1	1	1:256	1:128

PSA: Asignación del Divisor de frecuencias

1 = El divisor de frecuencia se le asigna al WDT
0 = El divisor de frecuencia se le asigna al TMR0

TOSE: Tipo de flanco en TOCK1

1 = Incremento de TMR0 cada flanco descendente
0 = Incremento de TMR0 cada flanco ascendente

TOCS: Tipo de reloj para el TMR0

1 = Pulsos introducidos a través de TOCK1 (contador)
0 = Pulsos de reloj interno Fosc/4 (temporizador)

INTEDG: Flanco activo interrupción externa

1 = Flanco ascendente
0 = Flanco descendente

RBPO#: Resistencias Pull-up Puerta B

1 = Desactivadas
0 = Activadas

2.1.10 El Perro Guardián (WDT)

También se trata de un contador de 8 bits que actúa como temporizador y tiene el objetivo de generar un reset a todo el sistema cuando se desborda su valor. Su control de tiempos es independiente del oscilador principal y se basa en una red RC.

La temporización nominal con la que se halla programado el Perro guardián es de 18 ms, pero puede aumentarse utilizando el Divisor de frecuencia, con el cual, trabajando en el rango mayor, puede alcanzar hasta 2,3 segundos.

Para evitar que se desborde el WDT y genere un reset, hay que recargar o refrescar su cuenta antes de que llegue el desbordamiento. Este refresco, que en realidad consiste en ponerle a 0 para iniciar la temporización, se consigue por software con las instrucciones CLRWDT y SLEEP. El diseñador debe analizar el programa de trabajo y situar alguna de estas dos instrucciones en sitios estratégicos por los que pasa el flujo de control antes que transcurra el tiempo que controla el Perro guardián. De esta manera, si el programa se "cuelga" (bucle infinito, espera de acontecimiento que no se produce, etc.), no se refresca a tiempo al Perro guardián y se produce una reinicialización.

La instrucción CLRWDT borra simplemente el valor de WDT, reiniciando la cuenta. Sin embargo, la instrucción SLEEP, además de borrar el WDT, detiene a todo el sistema entrando en un modo de trabajo en el que el consumo es mínimo (modo de reposo o de bajo consumo). Si no se desactiva al Perro guardián cuando se entra en el modo de reposo, al acabar su conteo provocará un reset y se saldrá de dicho modo. Otra forma de salir del modo de reposo es activando en la terminal MCLR. Para desactivar al Perro Guardián, hay que poner un 0 en el bit 2 (WDTE) de la Palabra de Configuración (Configuration Word).

Se refleja la posibilidad de que el Divisor de frecuencia opere con el TMR0 o con el WDT, según el valor que tenga el bit PSA. Los impulsos de conteo pasan por el Divisor antes de aplicarse al TMR0 (Predivisor). Por el contrario, los impulsos pasan primero por el WDT y luego por el Divisor (Post-divisor).

2.1.11 Las Puertas de Entrada/Salida

Los PIC16F84 sólo disponen de 2 puertas de E/S. La Puerta A posee 5 líneas, RA0-RA4, y una de ellas soporta dos funciones multiplexadas. La Puerta B tiene 8 Líneas, RB0-RB7, y también tiene una con funciones multiplexadas.

Cada línea de E/S puede configurase independientemente como entrada o como salida, según se ponga a 1 a 0, respectivamente, el bit asociado del registro de configuración de cada Puerta (TRISA y TRISB). Se llaman PUERTAA y PUERTAB los registros que guardan la información que entra o sale por la puerta y ocupan las direcciones 5 y 6 del banco 0 de la memoria de datos. Los registros de configuración TRISA y TRISB ocupan las mismas direcciones pero en el banco 1.

Puerta A

Las líneas RA3-RA0 admiten niveles de entrada TTL y de salida CMOS. La Línea RA4/TOCK1 dispone de un circuito Schmitt Trigger que proporciona una buena inmunidad al ruido y la salida tiene drenador abierto. RA4 multiplexa su función de E/S con la de entrada de impulsos externos para el TMR0. Los bits del registro TRISA configuran a las líneas de la Puerta A como entradas si están a 1 y como salidas si están a 0.

Al reinicializarse el PIC todos los bits del registro TRIS quedan a 1, con lo que las líneas de las puertas quedan configuradas como entradas.

Cada línea de salida puede suministrar una corriente máxima de 20 mA y si es entrada puede absorber hasta 25 mA. Al existir una limitación en la disipación máxima de la potencia del chip se restringe la corriente máxima de absorción de la puerta A a 80 mA y la de suministro a 50 mA. La puerta B puede absorber un máximo de 150 mA y suministrar un total de 100 mA.

Puerta B

Consta de 8 líneas bidireccionales de E/S, RB7-RB0, cuya información se almacena en el registro de configuración TRISB ocupa la misma dirección en el banco1.

La línea RB0/INT tiene dos funciones multiplexadas. Además de la terminal de E/S, actúa una entrada de interrupción externa. A todas las líneas de esta puerta se les permite conectar unas resistencias pull-up de elevado valor con el positivo de la alimentación. Para este fin hay que programar en el registro OPTION el bit RBPU# = 0, afectando la conexión de la resistencia a todas las líneas. Con el Reset todas las líneas quedan configuradas como entradas y se desactivan las resistencias pull-up.

Las 4 líneas de más peso, RB7-RB4, pueden programarse para soportar una misión especial. Cuando las 4 líneas actúan como entradas se les puede programar para generar una interrupción si alguna de ellas cambia su estado lógico.

El estado de las terminales RB7-RB4 en modo entrada se compara con el valor antiguo que tenían y que se había sostenido durante la última lectura de la puerta B.

La línea RB6 también se utiliza para la grabación en serie de la memoria de programa y sirve para soportar la señal de reloj. La línea RB7 constituye la entrada de los datos en serie.

2.1.12 Palabra de Configuración

Se trata de un dirección reservada de la memoria situada en la dirección 2007H y accesible durante el proceso de grabación. Al escribirse el programa de la aplicación es necesario grabar el contenido de esta posición de acuerdo con las características del sistema.

13													0
CP	CP	CP	CP	CP	CP	CP	CP	CP	CP	PWRTE#	WDTE	FOSC1	FOSC0

Figura 2.12 Distribución de los bits de la palabra de configuración del PIC16X84.

CP: Bits de Protección de la Memoria de Código

1: No protegida
0: Protegida. El programa no se puede leer, evitando copias. Tampoco se puede sobreescribir. Además evita que pueda ser accedida la EEPROM de datos y finalmente, si se modifica el CP de 0 a 1, se borra completamente la EEPROM.

PWRTE: Activación del Temporizador <<*POWER-UP*>>

El temporizador <<power-up>> retrasa 72ms la puesta en marcha o Reset que se produce al conectar la alimentación al microcontrolador , para garantizar la estabilidad de la tensión aplicada.
1: Desactivado
0: Activado

WDTE: Activación del Perro Guardián

1: Activado el WDT
0: Desactivado

FOSC1-FOSC0: Selección Del Oscilador Utilizado

1-1: Oscilador RC
1-0: Oscilador HS
0-1: Oscilador XT
0-0: Oscilador LP

Nota: *este proceso de selección del oscilador se puede hacer al momento de la grabación del micocontrolador.*

2.1.13 Palabras de Identificación

Son 4 posiciones reservadas de la memoria de programa ubicadas en las direcciones 2000H-2003H que no son accesibles en el funcionamiento normal del microcontrolador y sólo pueden ser leídas y escritas durante el proceso de grabación. Sólo se utiliza los 4 bits de menos peso de cada palabra de identificación (ID).

2.2 MEMORIA EEPROM DE DATOS

2.2.1 Propiedades de la EEPROM

Los PIC16F84 tienen 64 bytes de memoria EEPROM de datos, donde se pueden almacenar datos y variables que interesa que no se pierdan cuando se desconecta la alimentación al sistema.

La memoria EEPROM no está mapeada en la zona de la memoria de datos donde se ubican los registros SFR Falta definir y GPR. Para poder leerla y escribirla durante el funcionamiento normal del microcontrolador hay que utilizar 4 registros del banco SFR:

- ✓ **EEDATA**
- ✓ **EEADR**
- ✓ **EECON1**
- ✓ **EECON2**

En el registro EEADR, ubicado en la dirección 9 del banco 0, se carga la dirección a acceder de la EEPROM de datos.

En el registro EEDATA, ubicado en la dirección 8 del banco 0, se depositan los datos que se leen o se escriben.

El registro EECON1, que ocupa la dirección 88H de la memoria de datos, o la dirección 8 H del banco 1, tiene misiones de control de las operaciones en la EEPROM y la distribución de sus bits se presenta en la figura 2.13 mientras que la misión de cada uno se explica continuación:

EECON1

---	---	---	**EEIF**	**WRERR**	**WREN**	**WR**	**RD**

Figura 2.13 Distribución de los bits del registro EECON1

RD: Lectura

1: Se pone a 1 cuando se va a realizar un ciclo de lectura de la EEPROM. Luego pasa a 0 automáticamente.

WR: Escritura

1: Se pone a 1 cuando se inicia un ciclo de escritura de la EEPROM. Cuando se completa el ciclo pasa a 0 automáticamente.

WREN: Permiso de Escritura

1: Permite la escritura de la EEPROM
0: Prohibe la escritura

WRERR: Señalizador de error en Escritura

1: Se pone a 1 cuando una operación de escritura ha terminado prematuramente.
0: La operación de escritura se ha completado correctamente.

EEIF: Señalizador de final de operación de Escritura

1: Indica que la operación de escritura se ha completado con éxito. Se pone a 0 por programa.
0: La operación de escritura no se ha completado.

Un ciclo de escritura de una posición de la EEPROM de datos tiene una duración típica de 10 ms, que resulta muy larga para la velocidad del procesador. Por este motivo existen varios bits en el registro EECON1 para supervisar la completa y correcta terminación del mismo. El registro EECON2 en realidad no está constituido físicamente.

2.2.2 Proceso de Lectura

Se inicia un ciclo de lectura colocando la dirección a acceder en el registro EEADR y poniendo el bit RD = 1 en el registro EECON1. El dato leído estará disponible en el registro EEDATA en el siguiente ciclo y permanecerá en él hasta que se realice una nueva lectura o escritura en la EEPROM.

2.2.3 Proceso de Escritura

Para escribir una posición de la EEPROM de datos se debe seguir una determinada secuencia de instrucciones en las que participa el registro EECON2. Este registro, que en realidad no se halla implementado físicamente, sólo asume funciones de seguridad en el proceso, cargándose en él dos valores concretos: 55 H y AA H. La duración típica de un ciclo de escritura es de 10 ms, que es notablemente larga en comparación con la velocidad del PIC. El ciclo de escritura comi8enza cargando en EEADR la dirección de la posición a escribir y en el registro EEDATA el valor a grabar.

Al acabar el proceso de escritura el bit WR pasa a valer 0 automáticamente, mientras que el señalizador EEIF se pone a 1. Este último bit hay que ponerlo a 0 posteriormente mediante software.

2.3 INTERRUPCIONES Y RESET

2.3.1 Causas de Interrupción

En los PIC16F84 el Vector de Interrupción se halla situado en la dirección 0004H, en donde comienza la Rutina de Servicio a la Interrupción (RSI). En general, en dicho Vector se suele colocar una instrucción de salto incondicional (GOTO), que traslada el flujo de control a la zona de la memoria de código destinada a contener la rutina de atención a la interrupción.

Los PIC16F84 pueden ser interrumpidos por 4 causas diferentes, pero todas ellas desvían el flujo de control a la dirección 0004H, por lo que otra de las operaciones iniciales de la RSI es averiguar cuál de las posibles causas ha sido la responsable de la interrupción en curso. Para ello se exploran los señalizadores de las fuentes de interrupción.

Otro detalle importante en la RSI de los PIC16F84 es que estos microcontroladores poseen un bit GIE (Global Interrupt Enable) que cuando vale 0 prohibe todas las interrupciones. Al comenzar RSI dicho bit GIE se pone automáticamente a 0, con objeto de no atender nuevas interrupciones hasta que se termine la que ha comenzado. En el retorno final de la interrupción, GIE pasa a valer automáticamente 1 para volver a tener en cuenta las interrupciones. Dicho retorno de interrupción se realiza mediante la instrucción RETFIE.

Los PIC16X84 tienen 4 causas o fuentes posibles de interrupción:

1ª Activación del pin RB0/INT.
2ª Desbordamiento del temporizador TMR0.
3ª Cambio de estado en unos de los 4 pines de más peso (RB7:RB4) de la Puerta B.
4ª Finalización de la escritura de la EEPROM de datos.

Cuando ocurre cualquiera de los 4 sucesos indicados se originan una petición de interrupción, que si acepta y se atiende depositando el valor del Contador del Programa actual en la Pila, poniendo el bit GIE = 0 y cargando en el Contador de Programa el Valor 0004H, que es el Vector de Interrupción donde se desvía el flujo de control.

Cada fuente de interrupción dispone de un señalizador o "flag" (bandera), que es un bit que se pone automáticamente a 1 cuando se produce. Además cada fuente de interrupción tiene otro bit de permiso, que según su valor permite o prohíbe la realización de dicha interrupción.

2.3.2 Registro INTCON (Control de Interrupciones)

La mayor parte de los señalizadores y bits de permiso de las fuentes de interrupción en los PIC16X84 están implementados sobre los bits del registro INTCON, que ocupa la dirección 0B H del banco 0, hallándose duplicado en el banco 1. (ver figura 2.14).

GIE: Permiso Global de Interrupciones

1: Permite la ejecución de todas las interrupciones.
0: Prohíbe todas las interrupciones.

INTCON

GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF

Figura 214 Registro de Control de Interrupciones

EEIE: Permiso de Interrupción por fin de la escritura en la EEPROM

1: Permite que se origina la interrupción
0: Prohíbe esta interrupción

TOIE: Permiso de Interrupción por sobrepasamiento del TMR0N

1: Permite la Interrupción al desbordarse el TMR0
0: Prohíbe la Interrupción

INTE: Permiso de Interrupción por activación del pin RB0/INT

1: Permite la Interrupción al activarse RBO/INT
0: Prohíbe la Interrupción

RBIE: Permiso de Interrupción por cambio de estado lógico en RB7:RB4

1: Permite esta Interrupción
0: Prohíbe esta Interrupción

TOIF: Señal de sobrepasamiento del TMR0

1: Se pone a 1 cuando ocurre el sobrepasamiento
0: Indica que el TMR0 no se ha desbordado

INTF: Señal de activación del pin RB0/INT

1: Se pone a 1 cuando se activa RB0/INT
0: Indica que RB0/INT no se ha activado.

RBIF: Señal de cambio de estado lógico en los pines RB7:RB4

1: Pasa a 1 cuando cambia el estado lógico de alguna de estas líneas
0: No a cambiado el estado lógico en RB7:RB4

Cuando GIE = 0 no se acepta ninguna de las interrupciones. Si GIE = 1 sólo se aceptan aquellas fuentes de interrupción cuyo bit de permiso se lo permita. Los señalizadores deben ponerse a 0 por programa antes del retorno de la interrupción y son operativos aunque la interrupción esté prohibida con su bit de permiso correspondiente. En la figura 2.15 se muestra el esquema de la lógica de control que origina la interrupción.

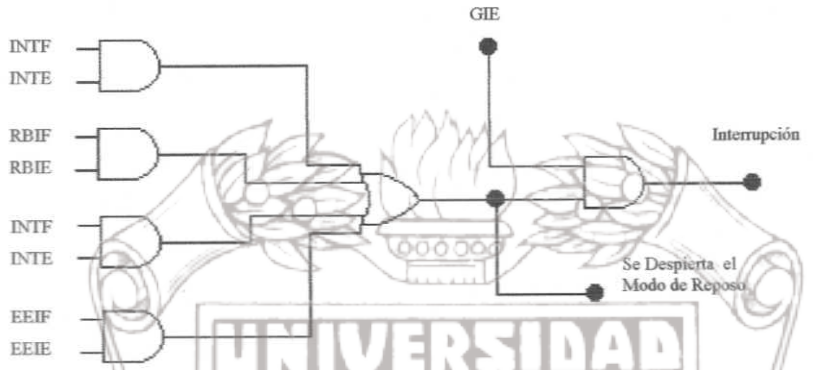


Figura 2.15 Lógica de Control para la generación de una Interrupción.

2.3.3 Interrupción Externa INT

Esta fuente de interrupción es sumamente importante para atender acontecimientos externos en tiempo real. Cuando ocurre alguno uno de ellos activa el pin RB0/INT y se hace una petición de interrupción. Entonces, de forma automática, el bit INTF = 1 y, si el bit de permiso INTE = 1y GIE = 1, se autoriza el desarrollo de la interrupción. Antes de regresar al programa principal hay que borrar INTF, puesto que en caso contrario al ejecutarse la instrucción de retorno RETFIE se volvería a desarrollar el mismo proceso de interrupción.

Mediante el bit 6, llamado INTDEG, del registro OPTION se puede seleccionar cuál será el flanco activo en RB0/INT. Si se desea que sea el ascendente se escribe un 1 en dicho bit, y si se desea que sea el descendente se escribe 0.

2.3.4 Interrupción por Desbordamiento del TMR0

Cuando TMR0 se desborda y pasa del valor FF H al 00H, el señalizador TOIF se pone automáticamente a 1. Sí, además. El bit de permiso de la interrupción del TMR0 = 1y el bit de Permiso Global de Interrupciones GIE = 1, se produce una interrupción.

Si no se recarga el TMR0 cuando se desborda, sigue contando desde 00 H a FF H. En cualquier momento se puede leer y escribir este registro, pero cada vez que se escribe se pierden dos ciclos de reloj para la sincronización.

Cuando se carga inicialmente TMR0 con el valor N_{10}, cuenta 256 – N impulsos, siendo el tiempo que tarda en hacerlo el que expresa la siguiente fórmula:

$$\text{Temporización} = (4)(T_{OSC})(256\text{-}N)(\text{ Rango de frecuencia})$$

2.3.5 Interrupción por Cambio de Estado Lógico en las Líneas RB7:RB4 de la Puerta B

Esta interrupción está diseñada específicamente para detectar la pulsación de una tecla correspondiente a un teclado matricial que se explora con 4 líneas de E/S. Para esta función se destinan las líneas RB7:RB4 de la Puerta B, cada vez que cambia el estado lógico de una de ellas fuerza al señalizador RBIF a ponerse a 1, y si los bits de permiso RBIE = GIE= 1 se autoriza la interrupción.

2.3.6 Interrupción por Finalización de la Escritura en la EEPROM de Datos

El tiempo típico que tarda en desarrollarse una operación de escritura en la EEPROM de datos de los PIC16X84 es de 10 ms, que es considerable comparado con la con la velocidad a la que el procesador ejecuta instrucciones. Para asegurarse que se ha completado la escritura y puede continuarse con el flujo de control del programa es aconsejable manejar la interrupción que se origina al finalizar la escritura, que pone automáticamente el señalizador EEIF a 1, y se autoriza siempre que los bits de permiso EEIE = GIE = 1.

2.3.7 Reinicialización o Reset

Los PIC16F84 tienen 5 causas que provocan la reinicialización del sistema, que consiste en cargar al Contador del Programa con el valor 000H (Vector de Reset) y poner el estado de los bits de los registros específicos (SFR) con un valor conocido.

1ª Conexión de la alimentación. POR (" Power on Reset ")
2ª Activación del pin MCLR# (" Master Clear Reset ") en funcionamiento normal
3ª Activación del pin MCLR# en estado de reposo
4ª Desbordamiento del Perro Guardián en funcionamiento normal
5ª Desbordamiento del Perro Guardián en estado de reposo

En la tabla 2.1 se presenta el estado lógico que adquieren los bits de los registros SFR de la memoria de datos cuando se provoca un reset por una de las 5 causas posibles. El pin MCLR# dispone de un filtro interno para eliminar los ruidos y los impulsos muy pequeños.

En el registro ESTADO hay 2 bits que indican las condiciones en las que se ha originado el Reset. Se trata de TO# (Timer Out) y PD# (Power Down). Figura 2.16.

REGISTRO	DIRECCIÒN	CONEXIÓN DE LA ALIMENTACIÒN	DESBORDAMIENTO PERRO GUARDIÁN MODO NORMAL	DESBORDAMIENTO PERRO GUARDIÁN MODO REPOSO	MCLR# MODO NORMAL	MCLR# MODO REPOSO
W	-------	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
INDIR	00h	-------	------	-------	-------	------
TRC	01h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PC	02h	0000 h	0000 h	PC + 1	0000 h	0000 h
STATUS	03h	0001 1xxx	0000 1uuu	uuu0 0uuu	000u uuuu	0001 0uuu
FSR	04h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PORT A	05h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PORT B	06h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TRIS A	85h	---1 1111	---1 1111	---u uuuu	---1 1111	---1 1111
TRIS B	86h	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
OPTION	81h	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
EEDATA	08h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
EEADR	09h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ECON. 1	88h	---0 0000	---0 ?000	---u uuuu	---0 ?000	---0 ?000
EECON 2	89h	-------	------	-----	----	------
PCLATH	0Ah	---0 0000	---0 0000	---u uuuu	---0 0000	---0 0000
INTCON	0Bh	0000 000x	0000 000u	uuuu uuuu	0000 0000	0000 0000

u = No cambia x = indeterminado ---- = No indeterminado ? = Depende de otras condiciones

Tabla 2.1 Valor que adquieren los registros SFR

TO#	PD#	Condición de Reset
1	1	POR (Reset por conexión V_{DD})
0	1	Desbordamiento WDT en funcionamiento normal
1	0	Desbordamiento WDT en estado de reposo
1	1	Activación MCLR# en funcionamiento normal
1	0	Activación MCLR# en reposo

Figura 2.16 Condiciones de Reset reflejadas por el estado de los bits TO# y PD# del registro ESTADO

2.3.8 Modo de Reposo (SLEEP)

Este modo de funcionamiento de los PIC está caracterizado por el reducido consumo de energía que requiere y está muy recomendado en aquellas aplicaciones en las que hay largos periodos de espera hasta que se produzca algún suceso asíncrono, como la pulsación de una tecla. En dichos periodos el procesador está inactivo.

El consumo típico del PIC es de 2 mA, aproximadamente, reduciéndose a menos de 10 uA en el modo de Reposo, lo que permite alimentarle con una pequeña pila durante casi dos años.

Para entrar en el modo de Reposo hay que ejecutar la instrucción SLEEP y se produce una situación muy especial de funcionamiento que parece como si el sistema se ha congelado, requiriendo el mínimo suministro de energía para mantener el estado del procesador sin ninguna actividad.

Para salir del estado de Reposo existen tres alternativas:

a) Activación externa de MCLR# para provocar un reset.
b) Desbordamiento del Perro Guardián si quedó en el modo de Reposo.
c) Generación de una interrupción.

2.4 MANEJO DE INSTRUCCIONES

2.4.1 Características Generales

Tal como corresponde a un procesador RISC, las instrucciones de los microcontroladores PIC cumplen las siguientes condiciones:

a) Juego reducido
- ✓ Sólo existen 35 instrucciones en la gama media

b) Sencillas y rápidas
- ✓ La mayoría se ejecuta en un ciclo de instrucción y sólo las de salto precisan dos ciclos. El ciclo de instrucción consta de cuatro periodos del reloj principal.

c) Ortogonalidad
- ✓ La ubicación de los operandos que manejan las instrucciones es muy flexible. Cualquier objeto del procesador puede actuar como fuente o como destino.

d) Formato uniforme de las instrucciones
- ✓ Todas las instrucciones de los modelos PIC16F84 tienen una longitud fija de 14 bits. Esta característica significa un notable ahorro de la memoria de código y una facilidad en la construcción de compiladores.

e) Formato uniforme de los datos
- ✓ En general, los datos y operandos tienen una longitud de 8 bits.

Respecto a los direccionamientos de la memoria de código, es el Contador de Programa (PC) el que establece el flujo de control. En la mayoría de las instrucciones el PC se incrementa automáticamente para apuntar la siguiente instrucción del programa. En las instrucciones de salto, cuando es del tipo directo, el valor que se carga en el PC proviene de una parte de los bits del código OP (código objeto) de la propia instrucción. En los saltos relativos, la ALU (unidad aritmética lógica) suma al valor actual del PC el del salto, almacenando el resultado en el PC.

Para direccionar los datos u operandos que manejan las instrucciones existen tres modos:

1°. Inmediato: El valor del dato está contenido en el código OP de la propia instrucción.

2°. Directo: La dirección del área de la memoria de datos donde se halla el operando está contenida en el código OP de la instrucción.

3°. Indirecto: La dirección de la memoria de datos que guarda el operando está contenida en un registro.

2.4.2 Definiciones y Abreviaturas

Para manejar el juego de instrucciones máquina de un procesador se precisa conocer la estructura y el funcionamiento de los componentes del mismo. Cuando se describe la operatividad de las instrucciones, se hace referencia a los objetos que dispone el procesador y al efecto que producen sobre ellos.

Para simplificar y facilitar la descripción de las instrucciones de la gama baja se utilizan bastantes abreviaturas que hay que conocer, habiéndose elegido las que emplea el fabricante del producto, Microchip[♦].

ABREVIATURA	DESCRIPCIÓN
PC	Contador de Programa que direcciona la memoria de instrucciones. Tiene un tamaño de 13 bits en la gama media, de los cuales los ocho de menos peso configuran el registro PCL que ocupa la dirección 02 del área de datos.
TOS	Cima de la Pila, que tiene 8 niveles en la gama media
WDT	Perro Guardián (Watchdog)
W	Registro W, similar al Acumulador
f	Suele ser un campo de 5 bits (fffff) que contiene la dirección del Banco de registros, que ocupa el banco 0 del área de datos.
d	Direcciona uno de dichos registros. Bit del código OP de la instrucción, que selecciona el destino. Si d = 0, el destino es W y si d = 1, el destino es f.
dest	Destino (Registro W o f)
TO	Bit "Time Out" del Registro de Estado
PD	Bit "Power Down" del Registro de Estado
b	Suele ser un campo de 3 bits (bbb) que determinan la posición de un bit dentro de un registro de 8 bits
k	Se trata, normalmente, de un campo de 8 bits (kkkk kkkk) que representa un dato inmediato. También puede constar de 9 bits en las direcciones de salto que cargan al PC
x	Valor indeterminado (puede ser un 1 ó 0)
label	Nombre de una etiqueta
[ ]	Opciones
( )	Contenido
→	Se asigna a
< >	Campos de bits de un registro
∈	En el conjunto de
Z	Señalizador de Cero en el Registro de Estado
C	Señalizador de Acarreo en el octavo bit del Registro de Estado
DC	Señalizador de Acarreo en el cuarto bit del Registro de Estado
Itálicas	Términos definidos por el usuario

♦ Fabricante del PIC 16F84

2.4.3 Instrucciones de Transferencia

Es el grupo de instrucciones más empleado y su misión fundamental es transferir información a y desde los registros del área de datos.

NEMONICO	DESCRIPCION	CODIGO OP	BANDERAS
MOVF f,d	Mueve el contenido del registro f al destino. Si d=0 el destino es W y si d=1, es f	0010 00df ffff	Z
MOVWF f	Mueve el contenido del registro W al registro f	0010 001f ffff	---- ----
MOVLW k	Carga un inmediato (literal k) en W	1100 kkkk kkkk	---- ----
SWAPF f,d	Se intercambian los 4 bits de mas peso con los de menos de f y el resultado se almacena en el destino (W si d=0 y f si d=1)	0011 10df ffff	---- ----

Tabla 2.2 Principales características de las instrucciones de transferencia correspondientes a los PIC16F84

2.4.4 Instrucciones Aritméticas

Las instrucciones de este grupo realizan diferentes operaciones aritméticas a través de la ALU: sumas, restas, complementos, incrementos, decrementos y rotaciones. Tabla 2.3.

NEMONICO	DESCRIPCION	CODIGO OP	BANDERAS
ADDWF f,d	Suma el contenido de W con el de f. Si d=0 el destino es W y si d=1, es f	0001 11df ffff	C, DC, Z
ADDLW	Suma el valor de la literal con W y lo deposita en éste.	111x kkkk kkkk	C,DC,Z
SUBLW	Resta W a la literal y lo deposita en W	110x kkkk kkkk	C,DC,Z
SUBWF f,d	Resta W de f	0000 10df ffff	C, DC, Z
INCF f,d	Incrementa f	0010 10df ffff	Z
DECF f,d	Decrementa f	0000 11df ffff	Z
COMF f,d	Complementa f	0010 01df ffff	Z
RLF f,d	Rota el registro f a la izquierda a través del acarreo (C)	0011 01df ffff	C
RRF f,d	Rota el registro f a la derecha a través del acarreo (C)	0011 00df ffff	C

Tabla 2.3 Instrucciones aritméticas

2.4.5 Instrucciones Lógicas

En la tabla 2.4 se presentan las instrucciones que efectúan sobre los operandos, las tres instrucciones lógicas típicas: AND, OR y XOR.

NEMONICO	DESCRIPCION	CODIGO OP	BANDERAS
ANDWF f,d	Operación lógica AND entre W y f	0001 11df ffff	C, DC, Z
IORWF f,d	Operación lógica OR entre W y f	0001 00df ffff	C, DC, Z
XORWF f,d	Operación lógica XOR entre W y f	0001 10df ffff	Z
ANDLW k	AND de W con un inmediato	1110 kkkk kkkk	Z
IORLW k	OR de W con un inmediato	1101 kkkk kkkk	Z
XORLW k	XOR de W con un inmediato	1111 kkkk kkkk	Z

Tabla 2.4. Instrucciones lógicas

2.4.6 Instrucciones de Puesta a Cero

Son tres instrucciones que borran o ponen a cero algún registro. Tabla 2.5.

NEMONICOS	DESCRIPCION	CODIGO OP	BANDERAS
CLRF	Borra el registro f	0000 011f ffff	Z
CLRW	Borra el registro W	0000 1000 0000	Z
CLRWDT	Borra o refresca el WDT	0000 0000 0100	TO y PD

Tabla 2.5. Instrucciones para el borrado de registros

2.4.7 Instrucciones de Salto

Este grupo de instrucciones rompe la secuencia normal del flujo de instrucciones del programa, provocando saltos. Afectan el contenido del PC. En la tabla 2.6 no se han incluido dos instrucciones especiales de salto que comprueban en un registro si un bit esta a 1 o a 0 y, según el caso, producen o no un salto de una instrucción ("skip").

NEMONICOS	DESCRIPCION	CODIGO OP	BANDERAS
CALL k	Salto a subrutina	1001 kkkk kkkk	---- ----
RETLW k	Retorno de subrutina (PC←TOS) y el registro W se carga con el inmediato k	1000 kkkk kkkk	---- ----
RETFIE	Retorna de una interrupción	0000 0000 1001	---- ----
RETURN	Retorna de una subrutina	0000 0000 1000	---- ----
GOTO k	Salto incondicional	101k kkkk kkkk	---- ----
DECFSZ	Se decrementa f. Si el resultado es 0 se salta la siguiente instrucción	0010 11df ffff	---- ----
INCFZ	Se incrementa f. Si el resultado es 0 se salta la siguiente instrucción	0011 11df ffff	---- ----

Tabla 2.6. Instrucciones de salto que rompen la secuencia normal del programa (TOS representa el contenido de la cima de la Pila)

2.4.8 Instrucciones para la Manipulación de Bits

Se trata de un pequeño conjunto de instrucciones que se encargan de verificar el valor de un bit particular de un registro, de ponerlo a 1 ó 0 e incluso de producir salto de una instrucción según se cumpla o no la condición establecida. Tabla 2.7.

NEMONICOS	DESCRIPCIÓN	CODIGO OP2	BANDERAS	NOTAS
BCF f,b	Borra el bit b del registro f	0100 bbbf ffff	---- ----	2 y 4
BSF f,b	Pone a 1 el bit b del registro f	0101 bbbf ffff	---- ----	2 y 4
BTFSC f,b	Verifica el bit b del registro f y si vale 0 salta una instrucción	0110 bbbf ffff	---- ----	---- ----
BTFSS f,b	Verifica el bit b del registro f y si vale 1 salta una instrucción	0111 bbbf ffff	---- ----	---- ----

Tabla 2.7 Instrucciones para el manejo de un bit de un registro.

2.4.9 Instrucciones Especiales

Son dos instrucciones que realizan operaciones muy específicas.

NEMONICOS	DESCRIPCION	CODIGO OP	BANDERAS	NOTAS
NOP	No operación (Pausa)	0000 0000 0000	---- ----	---- ----
SLEEP	Pasa al procesador al modo de reposo o bajo consumo	0000 0000 0011	TO y PD	---- ----

Tabla 2.8. Instrucciones especiales.

BIBLIOGRAFÍA

Angulo M. Ignacio, Angulo U. José M., ***"Microcontroladores PIC, Diseño práctico de aplicaciones"***, Ed. Mc Graw Hill, Madrid, España 1997.

CAPITULO III
ADQUISICION Y PROCESAMIENTO DE VARIABLES

3.1 ADQUISICIÓN DE DATOS

El objetivo de un sistema de control consiste en controlar la salida ***c*** de una manera predeterminada, por medio de la entrada ***u*** y aplicando los elementos del sistema de control. A las entradas del sistema también de les llama señales de control y a las salidas variables controladas. En el sistema de control que se diseñó, se definió la entrada como posición deseada POSD y la salida, como posición actual POSA. Benjamin C. Kuo, Sistemas automáticos de control,segunda ed. Een español, editorial continental, s.a. de c.v., mexico 1995.

3.1.1 Posición deseada

Antecedentes:

En el capitulo II se describió que el puerto A del PIC16F84 tiene 5 líneas de entrada (RA0–RA4) y el puerto B 8 líneas (RB0– RB7).

Para transmitir y almacenar la posición deseada (POSR) se requiere enviar (desde un lugar diferente a donde se ubicará el controlador) un dato de tamaño de palabra de 8 bits el cual representará a la misma; esta variable de entrada generará un desplazamiento angular en la flecha del motor de DC que se utilizará con propósitos de abrir o cerrar una válvula.

Si se considera que el dato de entrada consta de ocho bits, éste puede tener 256 posibles combinaciones -equivalente a 256 posiciones-. Para abrir, cerrar o posicionar una válvula (en este caso tipo compuerta) es necesario como máximo un desplazamiento angular de 0°-90°. Con la finalidad de tener versatilidad en el controlador se propone contar con un desplazamiento de aproximadamente 360°.

* ***Mínimo angulo de desplazamiento = 360° /256 posiciones = 1.4°***

La adquisición del dato de 8 bits por el puerto A del microcontrolador implica que este deberá accesarse en forma serial –por la condición del numero de terminales del mismo- de ahí que es necesario realizar una conversión del dato de paralelo-serie como se muestra

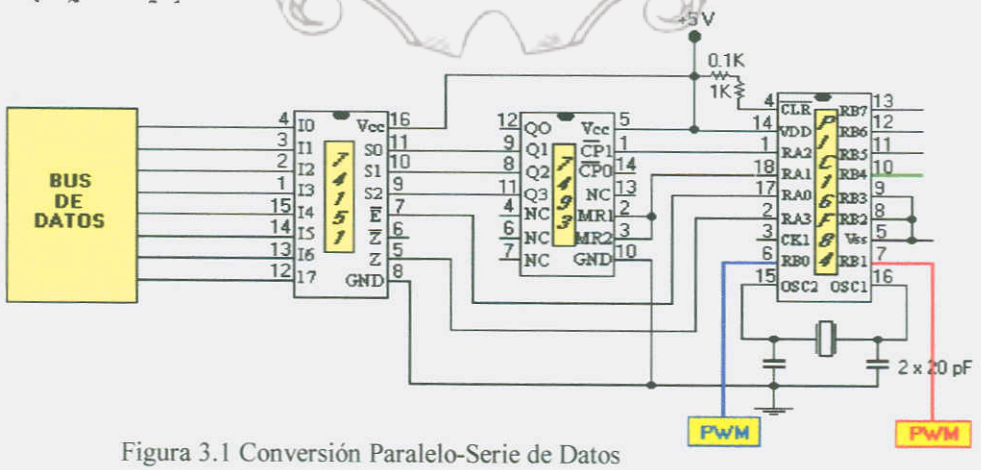


Figura 3.1 Conversión Paralelo-Serie de Datos

* Condiciones ideales considerando que no existan huelgos entre los engranes del sistema.

De acuerdo a lo apreciado en la figura 3.1, se define entonces que las líneas de entrada del puerto A, RA3 se utiliza como entrada serial y las otras tres para el control, vía software, de la conversión paralelo-serie. Para completar la transferencia de la posición deseada se requiere proporcionar 7 pulsos de reloj al contador que activa las entradas de control del multiplexor 8x1; el dato se almacena en un registro denominado "POSR" el cual ocupa la dirección 0Ch del microcontrolador.

3.1.2 Posición actual

La posición actual (POSA) del sistema se mide directamente de la flecha del motor, la cual tiene acoplado un sensor de posición (full 360° smart position sensor by spectrol) el cual nos proporcionara una señal analógica de 0 V a 5 V ± 10% según lo muestra la siguiente figura.

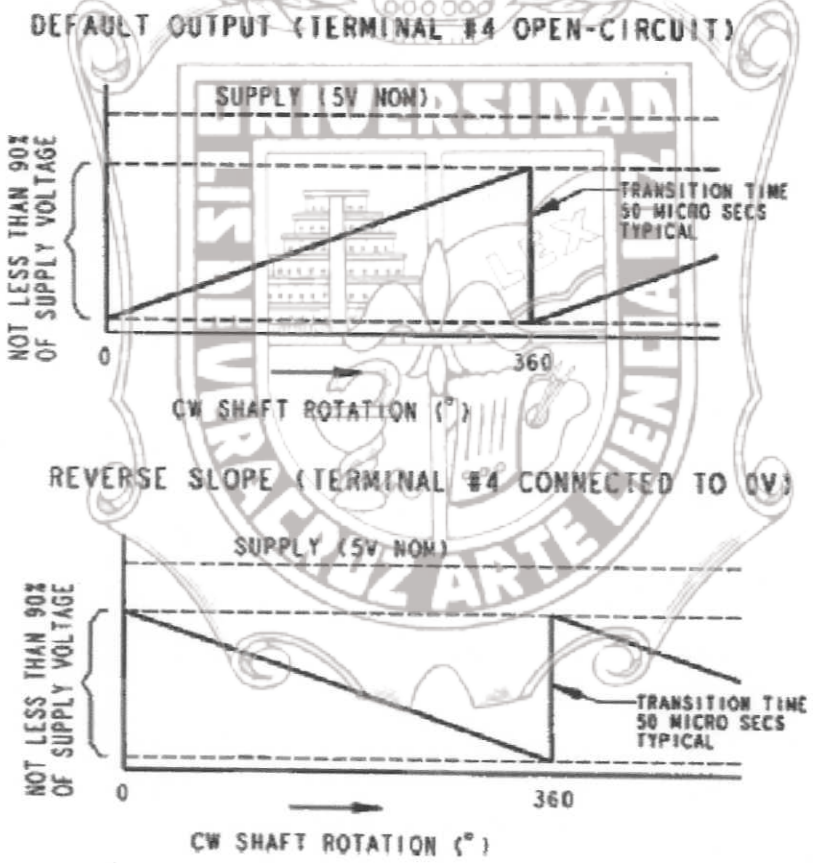


Figura 3.2 Señal analógica proporcionada por el sensor de posición *

Dicha señal analógica que representa el desplazamiento angular real en la flecha del motor, deberá procesarse a través de un convertidor análogo-digital (CI ADC0832) para

* Información tomada de la hoja de especificaciones del sensor smart position de Spectrol.

que de esta manera se obtenga un valor binario de ocho bits , equivalente al valor analógico proporcionado por el sensor.

De igual forma que en el proceso de transferencia y almacenamiento de la posición deseada; en el caso de la posición actual, ésta deberá ser transmitida en forma serial hacia el microprocesador; lo anterior se muestra en la figura 3.3.

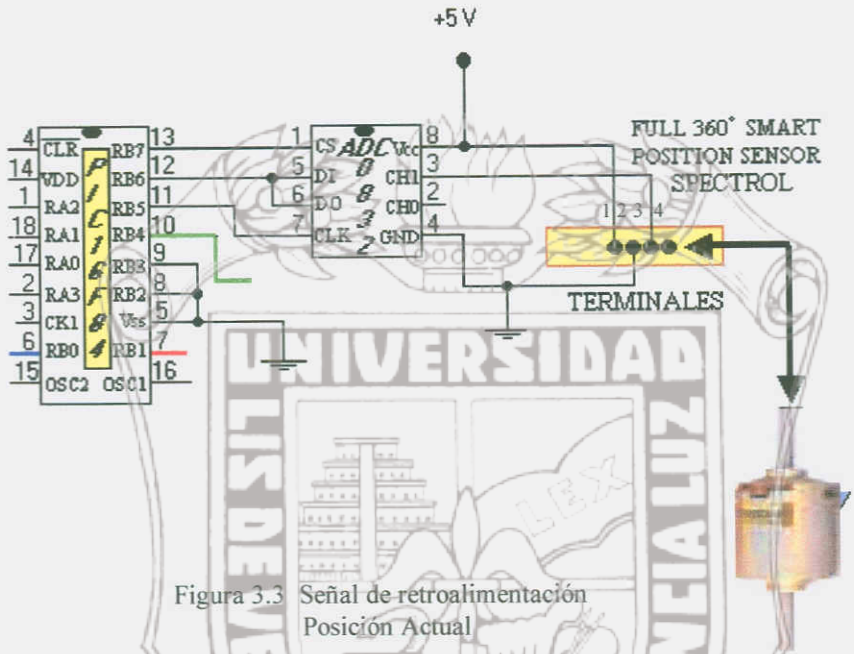


Figura 3.3 Señal de retroalimentación Posición Actual

De donde la posición deseada en su valor analógico se introduce en la terminal CH0 del convertidor y se transfiere en forma binaria por la terminal D0, hacia el microprocesador, utilizando para esto un algoritmo de adquisición de datos que controla las terminales CS, CLK y D1 del mismo convertidor. Asimismo el dato de ocho bits que representa a la posición actual; se almacena en un registro denominado "POSA" el cual ocupa la dirección 0Dh.

3.2 VARIABLE ERROR

En todo sistema de control de lazo cerrado la variable error es de suma importancia ya que en base a la variación de ésta, el sistema sabrá si se encuentra lejos o cerca de su set point.

Dada la importancia de la misma, es necesario conocer la magnitud del error y su signo; lo anterior para dar inicio a la construcción del controlador PID.

Haciendo uso de sencillos algoritmos matemáticos se manipulan los datos POSR y POSA para conocer la magnitud del error (POSR – POSA) y su signo, el cual se monitorea en el bit CARRY del registro status. De acuerdo al comportamiento del mismo si POSD > POSA entonces C=1 o bien si POSD < POSA entonces C=0. El signo del error permitirá

seleccionar el sentido de giro del motor, el cual es proporcionado por la activación del regulador tipo H a través de las señales CW o CCW que genera el microcontrolador. Esto se representa en la siguiente figura.

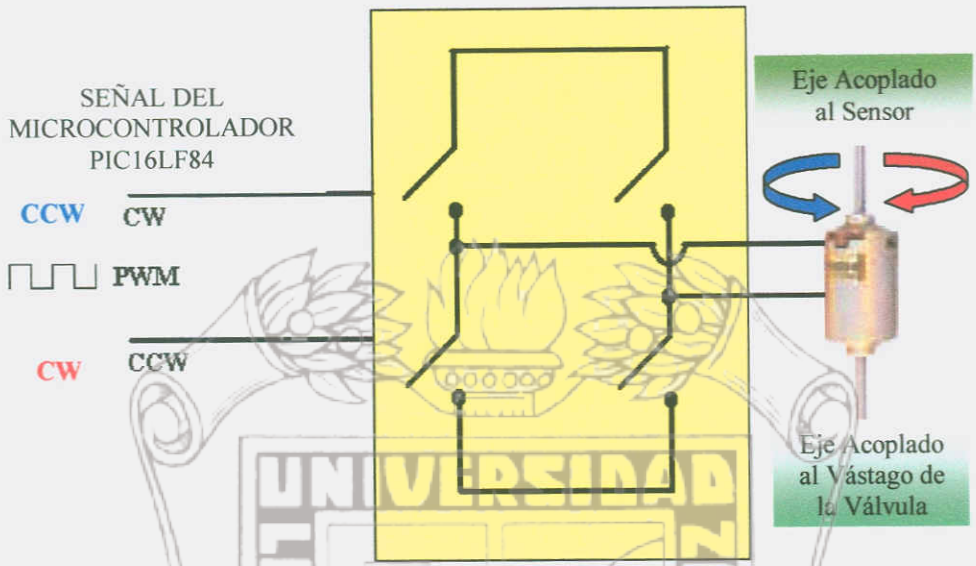


Figura 3.4 Puente tipo H, activado en cualquiera de los dos sentidos

3.3 SENSOR DE CORRIENTE

Hoy en día y con los avances de la tecnología de una manera sorprendente, la mayoría de los sistemas electrónicos cuentan con dispositivos de seguridad que los hacen ser más confiables y versátiles. Esto es con la idea de ofrecer al usuario la oportunidad de tener un sistema con la garantía ante posibles daños ocasionados por diversos factores, hablando en términos eléctricos, como lo son los picos de tensión o corriente.

Sin embargo, muchas veces estos picos de corriente o tensión no son lo suficientemente dañinos para un sistema, cuando no sobrepasa un tiempo o parámetros determinados en el cual el sistema se puede dañar, bajo estas condiciones el sistema debe seguir funcionando de manera normal.

Cuando un sistema arranca (un refrigerador, un aire acondicionado, etcétera; por ejemplo) demanda una cantidad considerable de corriente. Ante esta necesidad el sistema debe continuar funcionando de manera normal.

Para este proyecto se diseñó un sistema de protección de corriente, basado en las características eléctricas del motor, el cual soporta 0.5 Amperes como corriente máxima en un periodo no mayor a 7 segundos, los cuales pueden ser demandados por la carga al ser acoplada.

La idea desde un principio de este proyecto era controlar la posición de cierre o apertura de una válvula de tipo industrial, sin embargo el costo de la misma es muy elevado, por lo que no se establecieron las características de la misma, en cuestión de carga para el motor. Como alternativa y para fines didácticos o demostrativos, se simuló el cuerpo de la válvula con una caja de reducción de engranes.

En la primera parte (etapa analógica, figura 3.5) de este sensor, se toma la corriente que circula a través del puente que energiza al motor, para determinar el momento en que se está excediendo el límite de corriente. Esto se realiza mediante un comparador de voltaje. De esta manera, mientras la caída de voltaje a través de la resistencia (por el flujo de corriente) no supere el voltaje de referencia, la salida del integrado lm311, estará en un nivel bajo de voltaje, cuando el voltaje de referencia sea superado por exceso de corriente, esta salida se pondrá en un nivel alto (5 V).

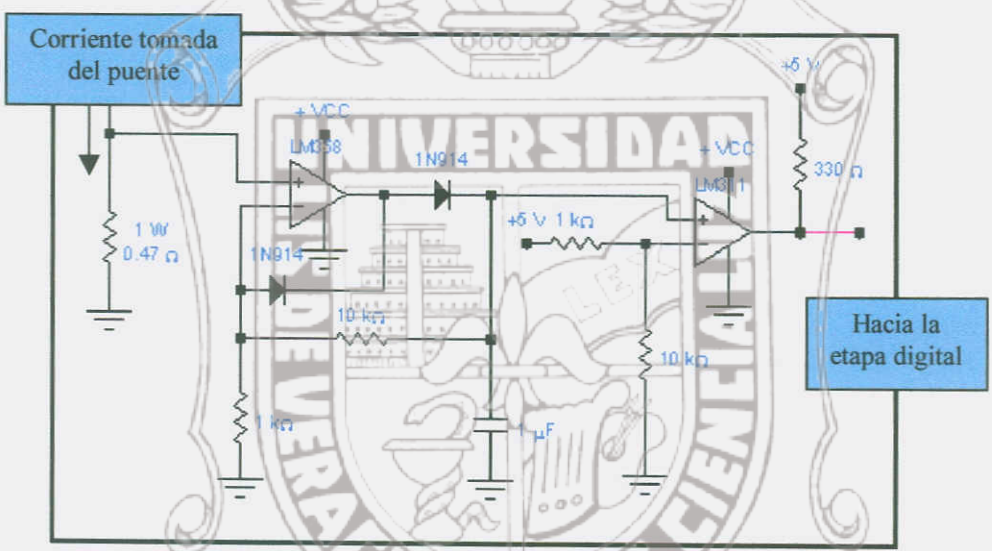
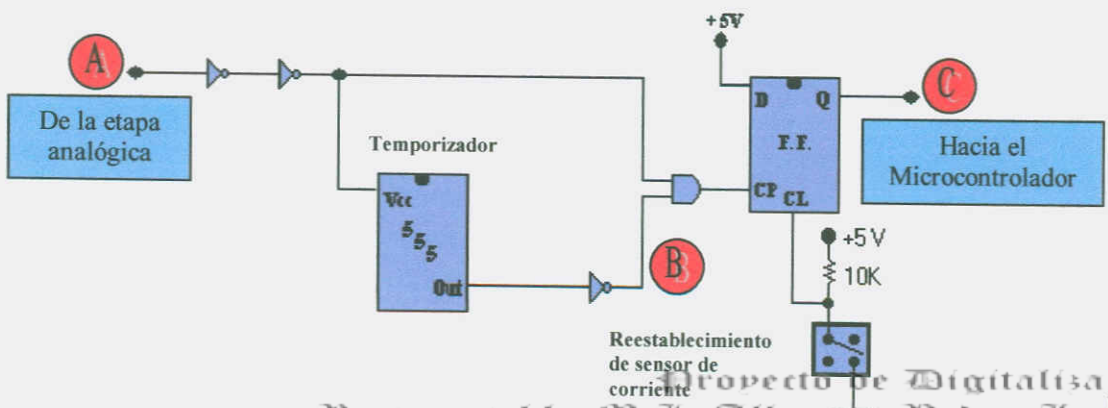


Figura 3.5 Parte analógica del sensor de corriente

Sin embargo, esa etapa no lo es todo, ya que si es sólo un pico de corriente, el sistema se apagaría y volvería a encenderse cuando la corriente se estabilice o estaría oscilando entre encendido y apagado debido a los arranques del motor. Por lo tanto surge la necesidad de almacenar este dato en una etapa digital, donde esta señal se comparará con una señal dada por un temporizador. La señal del temporizador estará condicionada por la señal misma de la etapa analógica (Figura 3.6).



Como se puede observar, la misma señal de la etapa analógica sirve como alimentación (Vcc) para el temporizador.

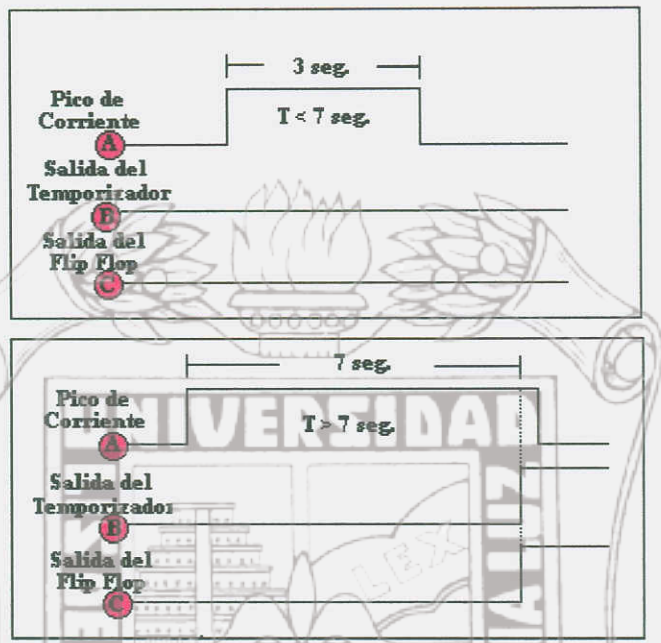


Figura 3.7 Diagrama de tiempos del sensor de corriente

La idea, como se muestra en la figura 3.7, es que mientras no pasen más de 7 segundos, el sistema se mantenga funcionando, de lo contrario que el mismo se apague.

3.4 PROGRAMA DEL CONTROLADOR

El programa se generó en lenguaje ensamblador (exclusivo para el PIC 16F84), para esto se consideraron cuatro etapas:

1. Diseño de algoritmos
2. Definición de variables
3. Construcción de subrutinas
4. Diseño del programa principal

- Diseño de algoritmos

El algoritmo principal se muestra en la figura 3.8:

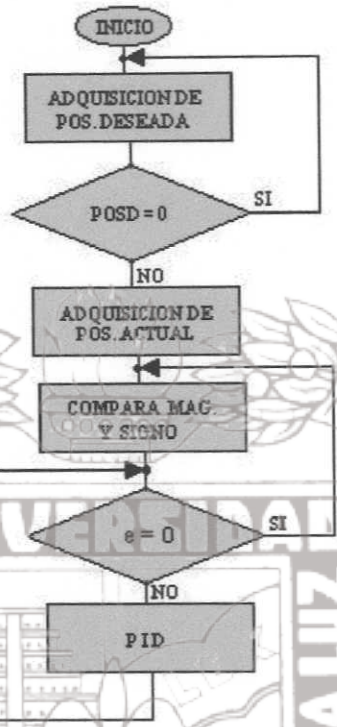


Figura 3.8 Algoritmo del Programa

Primeramente el sistema debe ser capaz de estar disponible para que una posición deseada sea introducida, si la posición es cero entonces el sistema esperará a que se de una posición.

Cuando existe una posición diferente de cero, el sistema comparará la misma, con la posición actual dada a través de la retroalimentación y determinará la magnitud y signo del error generado.

Una vez determinados estos valores, serán procesados a través de un controlador PID (Proporcional-Integrado-Derivativo), de donde resultará el sentido que los pulsos PWM deberán tomar para activar el puente tipo H que energizará el motor.

El programa que posee el microcontrolador, resultado del algoritmo mostrado en la figura 3.8 se muestra a continuación.

```
LIST P = 16LF84

W        EQU     0x00    ; ACUMULADOR (REGISTRO)
PORTA    EQU     0x05    ; PUERTO A DEL MICROCONTROLADOR
PORTB    EQU     0x06    ; PUERTO B DEL MICROCONTROLADOR
LRESULT  EQU     0x0E    ; REGISTRO PARA ADQUSICION DE DATOS
HRESULT  EQU     0x0F    ; REGISTRO PARA ADQUSICION DE DATOS
POSD     EQU     0x0C    ; POSICION DESEADA
POSA     EQU     0x0D    ; POSICION ACTUAL
DATAMUX  EQU     d'3'    ; DATO DE ENTRADA AL MUX
CLKMUX   EQU     d'2'    ; RELOJ DEL MUX
MR       EQU     d'1'    ; MASTER RESET DEL MUX
EMUX     EQU     d'0'    ; ENABLE DEL MUX

;******************* EQUIVALENCIAS ***********************

SWR      EQU     0x03    ; STATUS WORD REGISTER
                         ; 0 = ACARREO (CARRY)
                         ; 1 = DC
                         ; 2 = Z, SE ACTIVA SI EL RESULTADO ES CERO

CARRY    EQU     d'0'    ; BIT DE ACARREO DEL SWR
Z        EQU     d'2'    ; BIT CERO DEL SWR
DC       EQU     d'1'
SW4      EQU     d'4'    ; SWITCHES PARA INICIO
SW3      EQU     d'3'
SW2      EQU     d'2'
CSN      EQU     d'7'    ; CHIP SELECT PARA A/D
BV       EQU     d'6'    ; LINEA DE DATOS DEL A/D
CK       EQU     d'5'    ; RELOJ PARA EL A/D

;***************** EQUIVALENCIAS PARA ERROR ************************

FLAGS    EQU     0x19     ; SE UTILIZA ESTA LOCACION DE VARIABLE COMO BANDERAS
                          ; EL BIT 0 SIGNIFICA QUE EL ERROR 1 ES NEGATIVO
                          ; EL BIT 1 SIGNIFICA EL ACCUMULADOR DE ERROR
                          ; EL BIT 2 SIGNIFICA EL TERMINO  DE/DT
                          ; EL BIT 3 ES LA DIRECCION 0 REPRESENTA CW
                          ; EL BIT 4 SIGNIFICA EL VIEJO ERROR

STATUS   EQU     0x03
F        EQU     1
HI       EQU     07H      ; NUMERO DE MICROSEGUNDOS ACTIVO
LO       EQU     08H      ; NUMERO DE MICROSEGUNDOS INACTIVO
PCNT     EQU     09H      ; PORCENTAJE DE CICLO TRABAJO REQUERIDO
HI_T     EQU     0AH      ; CONTADOR PARA TIEMPO ACTIVO
LO_T     EQU     0BH      ; CONTADOR PARA TIEMPO INACTIVO
ERR1     EQU     0x10     ; MANTIENE EL ERROR DE POSICION ESTE TIENE
                          ; UNA  MAGNITUD DE 8 BITS CON EL SIGNO
SUMLO    EQU     0x11     ; SUMA PROGRESIVA DE  LOS TERMINOS PID
ACCUM    EQU     0x12     ; ACUMULADOR DEL ERROR
ERR_O    EQU     0x13     ; ERROR ANTERIOR USADO PARA EL de/dt
CYCLES   EQU     14H      ; CONTADOR PARA CICLOS DE SALIDA
mulcnd   EQU     15H      ; MUTIPLICANDO DE 8 BITS
```

```
ACCaLO  EQU     15H     ; MISMA LOCALIDAD USADA PARA LA RUTINA DE SUMA
mulplr  EQU     16H     ; MULTIPLICADOR DE 8 BIT
ACCbLO  EQU     16H     ; MISMA LOCALIDAD USADA PARA LA RUTINA DE SUMA
H_byte  EQU     17H     ; BYTE ALTO DEL RESULTADO DE 16 BITS
ACCaHI  EQU     17H     ; MISMA LOCALIDAD USADA PARA LA RUTINA DE SUMA
L_byte  EQU     18H     ; BYTE BAJO DEL RESULTADO DE 16 BITS
ACCbHI  EQU     18H     ; MISMA LOCALIDAD USADA PARA LA RUTINA DE SUMA
count   EQU     1AH     ; CONTADOR DE LAZO
SUMHI   EQU     1BH     ; BYTE ALTO DE LA SUMA DE LAZO

;*************** ASIGNACION DE PUERTOS Y CONSTANTES *********************

PWMCW   EQU     0       ; BIT DE SALIDA DE PWM CW (IZQ/DER)
PWMCCW  EQU     1       ; BIT DE SALIDA DE PWM CCW (DER/IZQ)
Same    EQU     1
ER_SGN  EQU     d'0'    ; BIT DE SIGNO PARA EL ERROR EN EL REGISTRO
BANDERA
AC_SGN  EQU     d'1'    ; BIT DE SIGNO PARA EL ACUMULADOR DE ERROR
DE_SGN  EQU     d'2'    ; BIT DE SIGNO PARA  DE/DT
OER_SGN EQU     d'4'    ; BIT DE SIGNO PARA EL ERROR ANTERIOR
KP      EQU     90      ; GANANCIA PROPORCIONAL
KI      EQU     40      ; GANANCIA INTEGRAL
KD      EQU     10      ; GANANCIA DIFERENCIAL
DIR     EQU     3       ; BANDERA DE DIRECCION

PNTR    EQU     00H     ; CONTENIDO DEL PUNTERO

;*****************************************************************
        ORG     0       ; POSICION DE INICIO DEL PROGRAMA EN LA PILA
        GOTO    BEGIN   ; INICIO

;******************************* MACROS ********************************
;***********************************************************************

CLKUP   MACRO           ; RELOJ DEL ADC
        BSF     PORTB,CK  ; ADQUISICION DE DATOS DEL A/D
        NOP             ; RETARDO
        ENDM

CLKDN   MACRO           ; RELOJ DEL ADC
        BCF     PORTB,CK  ; ADQUISICION DE DATOS DEL A/D
        NOP             ; RETARDO
        ENDM

GET_BIT MACRO           ; OBTENER LOS DATOS DE A/D
        BCF     SWR,CARRY ; PONE EL ACARREO EN CERO
        BSF     PORTB,CK  ; SELECCIONA RELOJ EN ALTO
        BTFSC   PORTB,BV  ; OBSERVA EL DATO QUE ENTRA
        BSF     SWR,CARRY ; SELECCIONA EL ACARREO EN 1
        RLF     POSA,1    ; ROTA A LA IZQUIERDA EL REGISTRO W
        BCF     PORTB,CK  ; SELECCIONA RELOJ EN BAJO
        NOP             ; RETARDO
        ENDM

;*********************  RUTINAS MATEMATICAS  ************************
;*******************************************************************
```

```
;********************  MULTIPLICACION DE 8 BIT ************************
;********** COMIENZO DE LA RUTINA DE MULTIPLICACION ********************

mpy_S   CLRF    H_byte
        CLRF    L_byte
        MOVLW   8
        MOVWF   count
        MOVF    mulcnd,W
        BCF     STATUS,CARRY
loop    RRF     mulplr, F
        BTFSC   STATUS,CARRY
        ADDWF   H_byte,Same
        RRF     H_byte,Same
        RRF     L_byte,Same
        DECFSZ  count, F
        GOTO    loop
        RETLW   0


;****** SUMA Y RESTA  DE DOBLE PRESICION ( ACCb-ACCa->ACCb ) ***********
;*************************************************************************

D_sub   CALL    neg_A

;********** SUMA DE DOBLE PREESICION ( ACCb+ACCa->ACCb ) ***************
;*************************************************************************

D_add   MOVF    ACCaLO,W
        ADDWF   ACCbLO, F
        BTFSC   STATUS,CARRY
        INCF    ACCbHI, F
        MOVF    ACCaHI,W
        ADDWF   ACCbHI, F
        RETLW   00


neg_A   COMF    ACCaLO, F
        INCF    ACCaLO, F
        BTFSC   STATUS,Z
        DECF    ACCaHI, F
        COMF    ACCaHI, F
        RETLW   00

;******DIVIDE ENTRE 16 Y LIMITA A 100 (DECIMAL)************
;***********************************************************

SHIFT   MACRO
        BCF     SWR,CARRY
        RRF     L_byte, F
        BCF     SWR,CARRY
        RRF     H_byte, F
        BTFSC   SWR,CARRY
        BSF     L_byte,7
        ENDM

DIV_LMT
        SHIFT
```

```
        SHIFT
        SHIFT
        SHIFT

LMT100
        MOVLW   1H
        SUBWF   H_byte,0
        BTFSS   SWR,CARRY
        GOTO    L8_E
        MOVLW   64H
        MOVWF   L_byte
        GOTO    LMT_EXIT
L8_E
        MOVLW   64H


        SUBWF   L_byte,0
        BTFSS   SWR,CARRY
        GOTO    LMT_EXIT
        MOVLW   64H
        MOVWF   L_byte
LMT_EXIT
        RETLW   00

;*************************************************************
; ESTA RUTINA CALCULA EL TIEMPO DE LA MANERA SIGUIENTE:
; PCNT = CICLO DE TRABAJO  % 100 - PCNT --> LO  Y
; PCNT --> HI. LOS VALORES DE CERO EN LO O HI SON FORZADOS A 1.
;*************************************************************

CALCTIMES
        MOVF    PCNT,W          ; PONE PORCENTAJE REQUERIDO EN EL REGISTRO W
        MOVWF   HI
        MOVLW   64H
        MOVWF   LO
        MOVF    PCNT,0
        SUBWF   LO,1
        MOVF    HI,0
        BTFSC   SWR,2
        INCF    HI,1
        MOVF    LO,0
        BTFSC   SWR,2
        INCF    LO,1
        RETLW   00



;***************** PROGRAMA DE POSICION DESEADA *********************
;********************************************************************

BEGIN
        CLRW                    ; LIMPIA O PONE A CERO TODAS LAS LOCALIDADES
        CLRF    PORTB           ; MENCIONADAS (CLEAR)
        CLRF    PORTA
        CLRF    LRESULT
        CLRF    HRESULT
        CLRF    POSD
```

```
CLRF    POSA

MOVLW   0x08
TRIS    PORTA

BSF     PORTA,EMUX
BSF     PORTA,MR        ; BORRA AL CONTADOR
BCF     PORTA,EMUX
BCF     PORTA,MR        ; DESHABILITA AL MR GENERAL

MOVF    PORTA,0
ANDLW   0x08

MOVWF   POSD
RRF     POSD,1          ; SE RECORRE TRES VECES EL DATO
RRF     POSD,1          ; EN EL REGISTRO POSD
RRF     POSD,1

BSF     PORTA,CLKMUX    ; SE ACTIVA EL RELOJ PARA EL CONTADOR
BCF     PORTA,CLKMUX

CLRW

MOVF    PORTA,0
ANDLW   0x08


MOVWF   LRESULT
RRF     LRESULT,1       ; SE RECORRE EL DATO DOS VECES EN EL REGISTRO
RRF     LRESULT,1       ; REGISTRO LRESULT

CLRW

MOVF    LRESULT,0       ; SE MUEVE EL DATO LRESULT A W Y SE REALIZA UNA
IORWF   POSD,1          ; SUMA CON EL DATO DE POSR Y SE DEJA EN POSD.

BSF     PORTA,CLKMUX    ; SE ACTIVA EL RELOJ PARA EL CONTADOR
BCF     PORTA,CLKMUX

CLRW


MOVF    PORTA,0
ANDLW   0x08


MOVWF   LRESULT
RRF     LRESULT,1

CLRW


MOVF    LRESULT,0
IORWF   POSD,1

BSF     PORTA,CLKMUX
BCF     PORTA,CLKMUX
```

```
CLRW

MOVF    PORTA,0
ANDLW   0x08


MOVWF   LRESULT

CLRW


MOVF    LRESULT,0
IORWF   POSD,0

ANDLW   0x0F            ; PRUEBA PARA QUE NO PASEN DATOS SIGNIFICATIVOS.

MOVWF   POSD

MOVF    PORTA,0
ANDLW   0x08

CLRF    LRESULT
CLRW

MOVF    POSD,0

MOVWF   LRESULT

BSF     PORTA,CLKMUX
BCF     PORTA,CLKMUX

CLRW

CLRF    POSD

MOVF    PORTA,0
ANDLW   0x08

MOVWF   HRESULT         ; SE RECORRE EL DATO UNA VEZ EN EL
RLF     HRESULT,1       ; REGISTRO HRESULT

CLRW

MOVF    HRESULT,0       ; SE MUEVE EL DATO HRESULT A W Y SE REALIZA
IORWF   POSD,1          ; UNA SUMA CON EL DATO POSD Y SE DEJA EN POSD.

BSF     PORTA,CLKMUX
BCF     PORTA,CLKMUX

CLRW

MOVF    PORTA,0
ANDLW   0x08

MOVWF   HRESULT         ; SE RECORRE EL DATO DOS VECES EN EL
```

```
RLF     HRESULT,1       ; REGISTRO HRESULT

RLF     HRESULT,1

CLRW

MOVF    HRESULT,0       ; SE MUEVE EL DATO HRESULT A WY SE REALIZA
IORWF   POSD,1          ; UNA SUMA CON EL DATO POSD Y SE DEJA EN POSD.

BSF     PORTA,CLKMUX
BCF     PORTA,CLKMUX

CLRW

MOVF    PORTA,0
ANDLW   0x08

MOVWF   HRESULT         ; SE RECORRE EL DATO TRES VECES EN EL
RLF     HRESULT,1       ; REGISTRO HRESULT
RLF     HRESULT,1
RLF     HRESULT,1

CLRW

MOVF    HRESULT,0       ; SE MUEVE EL DATO HRESULT A W Y SE REALIZA
IORWF   POSD,1          ; UNA SUMA CON EL DATO POSD Y SE DEJA EN POSD.

BSF     PORTA,CLKMUX
BCF     PORTA,CLKMUX

CLRW

MOVF    PORTA,0
ANDLW   0x08

MOVWF   HRESULT         ; SE RECORRE EL DATO CUATRO VECES EN EL
RLF     HRESULT,1       ; REGISTRO HRESULT
RLF     HRESULT,1
RLF     HRESULT,1
RLF     HRESULT,1

CLRW

MOVF    HRESULT,0       ; SE MUEVE EL DATO HRESULT A WY SE REALIZA
IORWF   POSD,0          ; UNA SUMA CON EL DATO DE POSD Y SE DEJA EN W.

ANDLW   0xF0

MOVWF   HRESULT

IORWF   LRESULT,0

MOVWF   POSD

BSF     PORTA,EMUX
```

```
;****************** CHECA LA POSICION DESEADA INVALIDA *******************

; LA POSICION 0 ES CONSIDERADA COMO UN REQUERIMIENTO QUE NO MANEJA EL SISTEMA.
; EL POSICIONADOR SE RESETEARA A UN ESTADO SEGURO DONDE NO SE TRATE DE MOVER
; EL SISTEMA A ALGUNA POSICION ARBITRARIA.

MOVE
        MOVF    POSD,W          ; CHECAR LA POSICION DESEADA
        BTFSC   SWR,Z           ; SI ES CERO ESPERA HASTA QUE NO HAYA OTRA
        GOTO    BEGIN


;********************** PROGRAMA DEL CONVERTIDOR ANALOGICO *************
;*****************************************************************

GOTO    CLRREG1

;** CHECA LOS LIMIT SWITCH PARA COMENZAR LA CONVERSION ANALOGA/DIGITAL **
;*********************************************************************

CLRREG1
        MOVLW   0x1C
        TRIS    PORTB
        CLRW
        OPTION
SW_TRAP
        CLRWDT
        BTFSC   PORTB,SW4
        GOTO    SW_TRAP
        BTFSC   PORTB,SW3
        GOTO    SW_TRAP

        BSF     PORTB,CSN
        GOTO    READ_POS

;********************** REQUISICION DE POSICION ************************
;*****************************************************************

;********************** LEYENDO LOS VALORES DEL A/D *******************

READ_POS
        CLRF    POSA            ; BORRA LA POSICION MANTENIDA ACTUALMENTE
        BCF     PORTB,CSN       ; PONE EL CHIP SELECT BAJO PARA A/D
        NOP
        MOVLW   1CH             ; PONE LA LINEA DE DATOS COMO SALIDA
        TRIS    PORTB           ; PARA MANDAR PONE ARRIBA LOS BITS
        BSF     PORTB,BV        ; PONE PARA "INICIO" BIT
        NOP
 CLKUP                          ; RELOJ EN EL BIT DE INICIO
 CLKDN
 CLKUP
 CLKDN
 CLKUP                          ; RELOJ EN CANAL 1
 CLKDN                          ; A EL MUX
        MOVLW   5CH             ; PONE LA LINEA DE DATOS COMO ENTRADA
        TRIS    PORTB           ; PARA RECIBIR DATOS DESDE A/D
 CLKUP
```

```
CLKDN
GET_BIT                         ; OBTIENE BIT 7
GET_BIT                         ; OBTIENE BIT 6
GET_BIT                         ; OBTIENE BIT 5
GET_BIT                         ; OBTIENE BIT 4
GET_BIT                         ; OBTIENE BIT 3
GET_BIT                         ; OBTIENE BIT 2
GET_BIT                         ; OBTIENE BIT 1
GET_BIT                         ; OBTIENE BIT 0
        BSF     PORTB,CSN


;****ESTE PROGRAMA COMPARTE LOS PROGRAMAS DE POSICION DESEADA Y DE******
;****POSICION ACTUAL POR LO QUE SE REALIZA EL CALCULO DE ERROR**********

;****************** CALCULOS DE LOS TERMINOS PID  **********************

;************************ CALCULO DEL ERROR ******************************

; EL ERROR ES MUY SIMPLE DE OBTENER YA QUE ES LA DIFERENCIA ENTRE DONDE SE
; ENCUENTRA EL SISTEMA Y DONDE DEBERIA DE ESTAR EN UN INSTANTE EN
; PARTICULAR. ESTA FORMADO POR LA RESTA DE LA POSICION DESEADA  DE
; LA POSICION ACTUAL (Posicion deseada - Posicion actual). ESTA DIFERENCIA
; ES USA PARA DETERMINAR LOS TERMINOS PROPORCIONAL, INTEGRAL Y DIFERENCIAL
; QUE CONTRIBUYEN A LA SALIDA.

C_ERR
        MOVF    POSA,0          ; CARGAR LA POSICION ACTUAL EN W
        SUBWF   POSD,0          ; RESTARLA DE LA POSICION REQUERIDA
        BTFSC   SWR,CARRY       ; CHECAR EL BIT CARRY PARA DETERMINAR EL SIGNO
        GOTO    PLS_ER          ; SI ES POSITIVO  (POSD>POSA)
        GOTO    MNS_ER          ; SI ES NEGATIVO  (POSA>POSD)

PLS_ER
        MOVWF   ERR1            ; SALVAR LA DIFERENCIA EN "ERROR"
        BCF     FLAGS,ER_SGN    ; PONER A 0 LA BANDERA DE SIGNO PARA INDICAR
                                ; QUE ES POSITIVO

        GOTO    CE_EXIT

MNS_ER
        MOVF    POSD,0          ; REALIZAR DE NUEVO LA RESTA
        SUBWF   POSA,0          ; ACTUAL - DESEADA
        MOVWF   ERR1            ; COLOCAR LA DIFERENCIA EN "ERROR"
        BSF     FLAGS,ER_SGN    ; PONER A 1 LA BANDERA DE SIGNO  PARA INDICAR
                                ; QUE ES NEGATIVO

CE_EXIT
        CLRF    SUMLO           ; LIMPIA VIEJOS VALORES DE SALIDA PARA PREPARARLOS
        CLRF    SUMHI           ; PARA LA SUMA DE ESTOS CICLOS


;****************CALCULO DEL TERMINO PROPORCIONAL ********************

; ESTE TERMINO REDUCE EL TIEMPO DE SUBIDA, PERO NO ELIMINA NUNCA EL ERROR
; EN REGIMEN PERMANENTE, EL TERMINO PROPORCIONAL ES EL ERROR DE TIEMPO DE
; LA GANANCIA PROPORCIONAL.
```

```
; EL TERMINO GANANCIA PROPORCIONAL ES UN TERMINO SEÑALADO ENTRE -100 Y 100.

C_PROP
        MOVF    ERR1,0          ; CARGA EL TERMINO DE ERROR EN W
        MOVWF   mulcnd          ; MULTIPLICA ESTE POR LA GANANCIA PROPORCIONAL
        MOVLW   KP
        MOVWF   mulplr
        CALL    mpy_s
        CALL    DIV_LMT

RESTORE_SGN
        BTFSS   FLAGS,ER_SGN    ; SI EL SIGNO DEL ERROR ES NEGATIVO ENTONCES
        GOTO    ADDPROP         ; PONE EL SIGNO EN EL BYTE BAJO
        COMF    L_byte,1
        INCF    L_byte,1


ADDPROP
        MOVF    L_byte,W        ; SALVA LA PARTE PROPORCIONAL
        ADDWF   SUMLO,1         ; EN LA SUMA
        BTFSC   SWR,CARRY       ; SI TUVO ACARREO DE SALIDA ENTONCES
        INCF    SUMHI,1         ; INCREMENTA EL BYTE ALTO
        MOVLW   0               ; ENTONCES
        BTFSC   SUMLO,7         ; EL SIGNO SE EXTIENDE HASTA
        MOVLW   0FF             ; EL BYTE SUPERIOR
        ADDWF   SUMHI,1



;****************CALCULO DEL TERMINO INTEGRAL*************************

; ELIMINA EL ERROR EN REGIMEN PERMANENTE, SIN EMBARGO EMPEORA LA RESPUESTA
; TRANSITORIA DEL SISTEMA. ESTE CORRIGE EL ERROR AÑADIENDO UN NUMERO PEQUEÑO
; EN EL ACUMULADOR EN CADA CICLO DE EJECUCION DEL PROGRAMA.

C_INT
        MOVF    ERR1,W          ; COLOCA EL ERROR EN EL REGISTRO W
        BTFSC   SWR,Z           ; Y CHECA PARA VER SI ESTA ES CERO
        GOTO    ADDINT          ; SI NO ES ASI ENTONCES CAMBIA EL ACUMULADOR
        BTFSC   FLAGS,ER_SGN    ; PRUEBA BANDERAS PARA ENCONTRAR LA POLARIDAD
        GOTO    MNS_1           ; DEL ERROR .. 0 POSITIVO 1 NEGATIVO
PLS_1
        MOVLW   KI              ; SI ES POSITIVO SUMA UNO A
        ADDWF   ACCUM,1         ; ACUMULADOR DE ERROR
        GOTO    LMTACM          ; LIMITANDO ESTE A +/-100
MNS_1
        MOVLW   KI              ; SI ES NEGATIVO RESTA UNO
        SUBWF   ACCUM,1         ; DEL ACUMULADOR DE ERROR
LMTACM
        BTFSC   ACCUM,7         ; CHECA EL SIGNO DEL BIT DEL ERROR EN EL ACUMULADOR
        GOTO    M_LMT           ; Y HACE AL "LIMIT" POSITIVO O NEGATIVO
P_LMT
        MOVLW   9CH             ; PARA UN "LIMIT" POSITIVO SUMA 156 AL
        ADDWF   ACCUM,0         ; NUMERO Y VE SI GENERA UN ACARRE0
        BTFSS   SWR,CARRY       ; CHECA LA BANDERA DE ACARREO SI ES ACTIVA
        GOTO    ADDINT          ; SI NO ES ASI ENTONCES ESTA BIEN
        MOVLW   64H             ; DE LO CONTRARIO, FORZA AL ACUMULADOR A
```

```
        MOVWF   ACCUM           ; 100 DECIMAL
        GOTO    ADDINT
M_LMT
        MOVLW   9CH             ; PARA UN "LIMIT" NEGATIVO RESTA 156 DE
        SUBWF   ACCUM,0         ; EL NUMERO Y VE SI GENERA UNA CONDICION
        BTFSC   SWR,CARRY       ; DE NO ACARREO, INDICANDO UN "ROLL-OVER"
        GOTO    ADDINT
        MOVLW   9CH
        MOVWF   ACCUM

ADDINT
        MOVF    ACCUM,W
        ADDWF   SUMLO,1
        BTFSC   SWR,CARRY
        INCF    SUMHI,1
        MOVLW   0
        BTFSC   ACCUM,7
        COMF    W,W
        ADDWF   SUMHI,1


U_DEXIT


;****************** CALCULO DEL TERMINO DIFERENCIAL ********************

; ESTE TERMINO INCREMENTA LA ESTABILIDAD, REDUCE EL SOBREIMPULSO
; Y MEJORA LA RESPUESTA TRANSITORIA EN EL SISTEMA.
; EL TERMINO DIFERENCIAL EXAMINA EL ERROR Y DETERMINA QUE TANTO
; HA CAMBIANDO DESDE EL ULTIMO CICLO. LO HACE RESTANDO EL ERROR ANTERIOR
; DEL NUEVO ERROR. ENTONCES EL TIEMPO DEL CICLO ES RELATIVAMENTE MODIFICADO,
; Y PODEMOS USAR ESTE COMO EL "DT" DEL DESEADO "DE/DT". ESTE DERIVADO DEL
; ERROR ES ENTONCES MULTIPLICADO POR EL TERMINO DE GANANCIA DIFERENCIAL KD Y
; SE CONVIERTE EL TERMINO DIFERENCIAL EN UNA CONTRIBUCION PARA LA SUMA FINAL.

; PRIMERO, CREA EL TERMINO "DE" HACIENDO UNA SUBSTRACCION DEL NUEVO ERROR
; MENOS EL ERROR ANTERIOR. (nuevo error - anterior error).

C_DIFF
        MOVF    ERR1,W          ; CARGA EL NUEVO ERROR EN EL REGISTRO W
        BTFSS   FLAGS,ER_SGN
        GOTO    LO_BYTE
        COMF    ERR1,1          ; CORRIGE EL VALOR PARA SER DE 16 BIT
        INCF    ERR1,W
        COMF    ERR1,1          ; RESTAURA ESTE PARA FUTURO USO A MAG. DE 8 BIT
LO_BYTE
        MOVWF   ACCbLO          ; PARA SUBSTRACCION
        MOVLW   00
        BTFSC   FLAGS,ER_SGN    ; SIGNO SE EXTIENDE AL BYTE ALTO
        MOVLW   0FF
        MOVWF   ACCbHI
        MOVF    ERR_O,W         ; CARGA EL ERROR ANTERIOR EN OTRO REGISTRO
        BTFSS   FLAGS,OER_SGN
        GOTO    LO_BYTEO
        COMF    ERR_O,1         ; CORRIGE EL VALOR A 16 BIT
        INCF    ERR_O,W
LO_BYTEO
```

```
        MOVWF   ACCaLO          ; PARA SUBSTRACCION
        MOVLW   00
        BTFSC   FLAGS,OER_SGN   ; SIGNO SE EXTIENDE AL BYTE ALTO
        MOVLW   0FF
        MOVWF   ACCaHI
        CALL    D_sub           ; SE REALIZA LA SUBSTRACCION

STRIP_SGN
        BTFSC   ACCbHI,7        ; PRUEBA EL SIGNO DEL RESULTADO
        GOTO    NEG_ABS
        GOTO    POS_ABS
NEG_ABS
        BSF     FLAGS,DE_SGN    ; SI ES NEGATIVO ACTIVA LA BANDERA Y
        COMF    ACCbLO,1        ; COMPLEMENTA EL VALOR
        INCF    ACCbLO,W
        MOVWF   ERR_O
        GOTO    MULT_KD
POS_ABS
        BCF     FLAGS,DE_SGN    ; SI  ES POSITIVO RESETEA LA BANDERA
        MOVF    ACCbLO,W        ; Y GUARDA EL VALOR
        MOVWF   ERR_O


MULT_KD
        MOVF    ERR_O,W
        MOVWF   mulcnd          ; MUEVE EL TERMINO DE/DT AL REGISTRO "MULCND"
        MOVLW   KD              ; MUEVE EL TERMINO GANANCIA DIFERECIAL AL
        MOVWF   mulplr          ; REGISTRO "MULPLR" PARA MULTIPLICAR EL DE/DT
        CALL    mpy_S
        CALL    DIV_LMT

RE_SGN
        BTFSS   FLAGS,DE_SGN    ; SI EL SIGNO DE "DE" ES NEGATIVO
ENTONCES
        GOTO    SAVE_DIFF       ; PONE EL SIGNO EN EL BYTE BAJO.
        COMF    L_byte,1
        INCF    L_byte,1

SAVE_DIFF
        MOVF    L_byte,W
        BTFSC   SWR,Z
        GOTO    ROLL_ER
        MOVWF   ERR_O


ADDDIF
        MOVLW   00
        BTFSC   FLAGS,DE_SGN    ; PONE EL TERMINO KD*(DE/DT)  EN
        MOVLW   0FF             ; LOS REGISTROS CORRESPONDIENTES. Y EL
        MOVWF   ACCbHI          ; SIGNO SE EXTIENDE AL BYTE ALTO.
        MOVF    ERR_O,W
        MOVWF   ACCbLO
        MOVF    SUMLO,W
        MOVWF   ACCaLO
        MOVF    SUMHI,W
        MOVWF   ACCaHI
        CALL    D_add
```

```
        MOVF    ACCbLO,W
        MOVWF   SUMLO
        MOVF    ACCbHI,W
        MOVWF   SUMHI

ROLL_ER
        MOVF    ERR1,W
        MOVWF   ERR_O
        BCF     FLAGS,OER_SGN
        BTFSC   FLAGS,ER_SGN
        BSF     FLAGS,OER_SGN


;************ DETERMINACION DE LA DIRECCION DEL PUENTE*********************
;
; DESPUES DE QUE LA SUMA DE TODOS LOS COMPONENTES HA SIDO REALIZADA, EL SIGNO
; DE LA SUMA DETERMINARA DE QUE MANERA EL PUENTE DEBERA SER ACTIVADO.
; SI LA SUMA ES NEGATIVA EL PUENTE NECESITA SER PUESTO EN LA DIRECCION CCW; SI
; LA SUMA ES POSITIVA ENTONCES EL PUENTE NECESITA SER PUESTO EN LA DIRECCION CW.
; ESTO ES PURAMENTE UNA CONVENCION Y DEPENDE DE LA POLARIDAD DEL MOTOR Y EL
; ELEMENTO DE RETROALIMENTACION.

SET_DIR
        BCF     FLAGS,DIR
        BTFSC   SUMHI,7
        BSF     FLAGS,DIR


;*************** ESCALA DE NUMERO ENTRE 0 Y 100% ********************

; DESPUES QUE LA DIRECCION ES PUESTA LA PETICION PARA EL CICLO DE TRABAJO
; ES LIMITADO ENTRE 0 Y 100 PORCIENTO. ESTE VALOR ES PASADO POR LA RUTINA
; CORRESPONDIENTE Y CARGADO EN LA VARIABLE "PCNT"

L_SUMM
        BTFSS   SUMHI,7         ; CHECA PARA VER SI ESTE ES NEGATIVO
        GOTO    POS_LM
        COMF    SUMHI,1
        COMF    SUMLO,1
        INCF    SUMLO,1

POS_LM
        MOVLW   1H
        SUBWF   SUMHI,0
        BTFSS   SWR,CARRY
        GOTO    LB_L
        MOVLW   64H
        MOVWF   SUMLO
        GOTO    LP_EXIT
LB_L
        MOVLW   64H
        SUBWF   SUMLO,0
        BTFSS   SWR,CARRY
        GOTO    LP_EXIT
        MOVLW   64H
        MOVWF   SUMLO
```

```
LP_EXIT
        MOVF    SUMLO,W         ; ALMACENA EL VALOR REQUERIDO DE
        MOVWF   PCNT            ; CICLO DE TRABAJO PARA PWM

;*********************************************************************
;****************** RUTINA DE GENERACION DE PWM **********************
;
; LO IMPORTANTE AQUI ES NO TENER QUE REALIZAR DEMASIADAS DESICIONES O
; CALCULOS MIENTRAS ESTAS GENERANDO LOS 100 O MAS PULSOS. ESTO TOMARIA
; TIEMPO Y LIMITARIA EL MINIMO O MAXIMO DE CICLO DE TRABAJO.

WHICH_DIR
        BTFSC   FLAGS,DIR       ; CHECA LA DIRECCION DE LA BANDERA
        GOTO    GOCCW           ; PARA DETERMINAR EL SENTIDO DEL PWM
        GOTO    GOCW            ; PARA EL PUENTE

GOCW
        BCF     PORTB,PWMCCW    ; PONE EL PUENTE EN LA DIRECCION CW
        MOVLW   64H
        MOVWF   CYCLES
        CALL    CALCTIMES

RLDCW
        MOVF    HI,0
        MOVWF   HI_T
        MOVF    LO,0
        MOVWF   LO_T
        CLRWDT

CWHI
        BSF     PORTB,PWMCW
        DECFSZ  HI_T,1
        GOTO    CWHI
CWLO
        BCF     PORTB,PWMCW
        DECFSZ  LO_T,1
        GOTO    CWLO
        DECFSZ  CYCLES,1
        GOTO    RLDCW
        GOTO    BEGIN


GOCCW
        BCF     PORTB,PWMCW     ; PONE EL PUENTE EN LA DIRECCION CCW
        MOVLW   64H
        MOVWF   CYCLES
        CALL    CALCTIMES
RLDCCW
        MOVF    HI,0
        MOVWF   HI_T
        MOVF    LO,0
        MOVWF   LO_T
        CLRWDT

CCWHI
        BSF     PORTB,PWMCCW
        DECFSZ  HI_T,1
```

```
        GOTO    CCWHI
CCWLO
        BCF     PORTB,PWMCCW
        DECFSZ  LO_T,1
        GOTO    CCWLO
        DECFSZ  CYCLES,1
        GOTO    RLDCCW
        GOTO    BEGIN

        END                     ; FIN DEL PROGRAMA
```

3.5 DIAGRAMA ESQUEMÁTICO DEL SITEMA DE CONTROL

Como resultado de la interconexión de las diferentes etapas de diseño del controlador de posición, se muestran acontinuación los diagramas esquemáticos del circuito electrónico de baja potencia y de potencia.

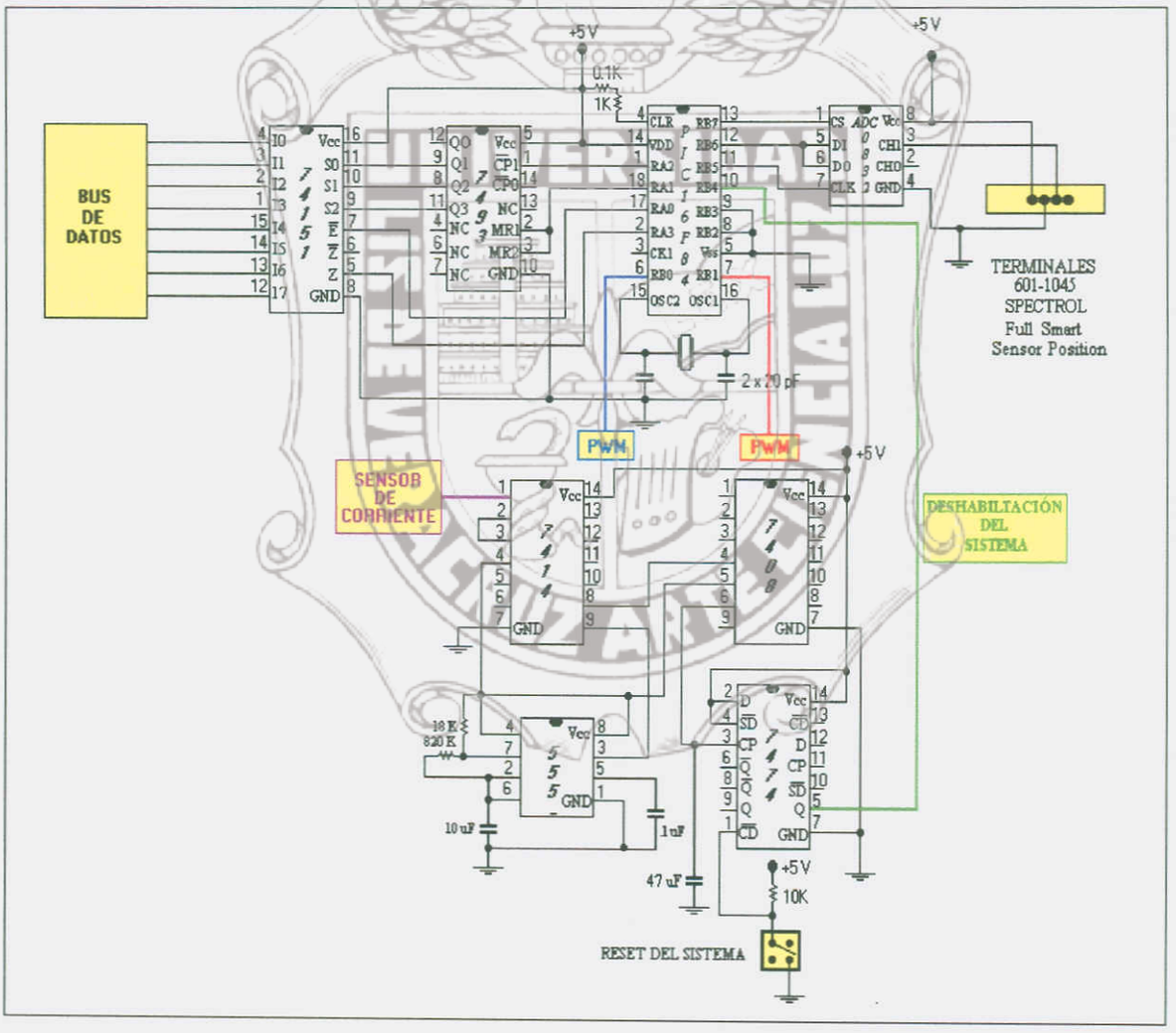


Figura 3.4 Circuito de la etapa de baja potencia, del sistema de control.

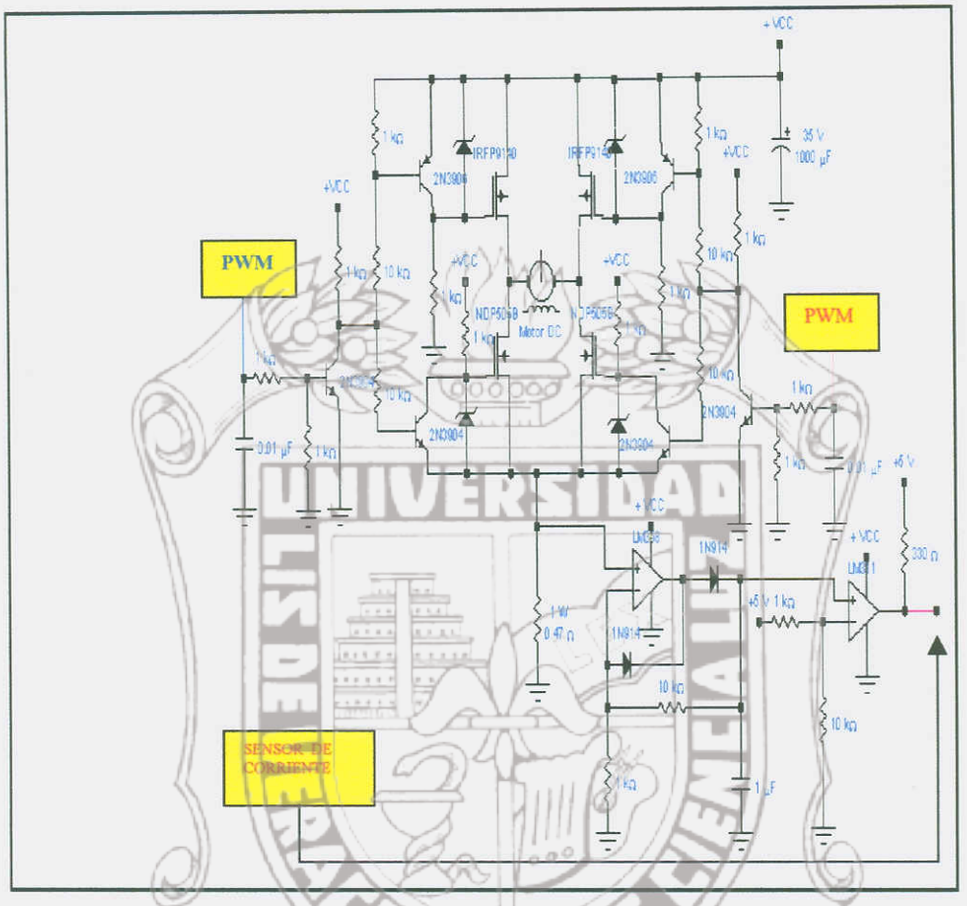


Figura 3.5 Circuito de la etapa de potencia, del sistema de control.

BIBLIOGRAFÍA

- Microchip Technology Inc., ***Microchip Data Book,*** Ed. Microchip, USA 1994.
- ***Embedded Control Handbook***, Microchip, Vol. 1.
- ***MPSIM SIMULATOR, USER´S GUIDE***, Microchip.
- ***PICSTART-16B1 DEVELOPMENT SYSTEM, USER´S GUIDE***, Microchip.
- Motorola, ***Fast and TTL Data*** Motorola, Inc., 1992.
- ***www.microchip.com***

- *www.spectrol.com*
- *www.national.com*

CAPITULO IV
LECTURA E INTERPRETACION DE VARIABLES

4.1 MEDICIÓN DE VARIABLES

La medición de variables representan el enlace entre el sistema y el mundo exterior, en la práctica no todas las variables son accesibles o medibles, las que aquí se presentan son las que pudieron ser medidas de manera física, el resto del proceso fue verificado apoyado en el simulador SIMUPIC'84.

4.1.1 Posición Deseada

La posición deseada es introducida de forma paralela (8 bits), la cual es convertida en forma serial para introducirla al puerto A del microcontrolador, en la figura 4.1 se muestra la primera señal (Gráfico verde) como posición deseada, la segunda (Gráfico amarillo) son los pulsos de reloj generados por el microcontrolador para realizar la conversión paralela-serie y la tercera señal (Gráfico morado) es la habilitación para la misma conversión, de arriba hacia abajo. Se puede observar que el dato de posición deseada es satisfactoria para ser recibida por el microcontrolador en forma serial, el cual almacena en uno de sus registros dicho dato.

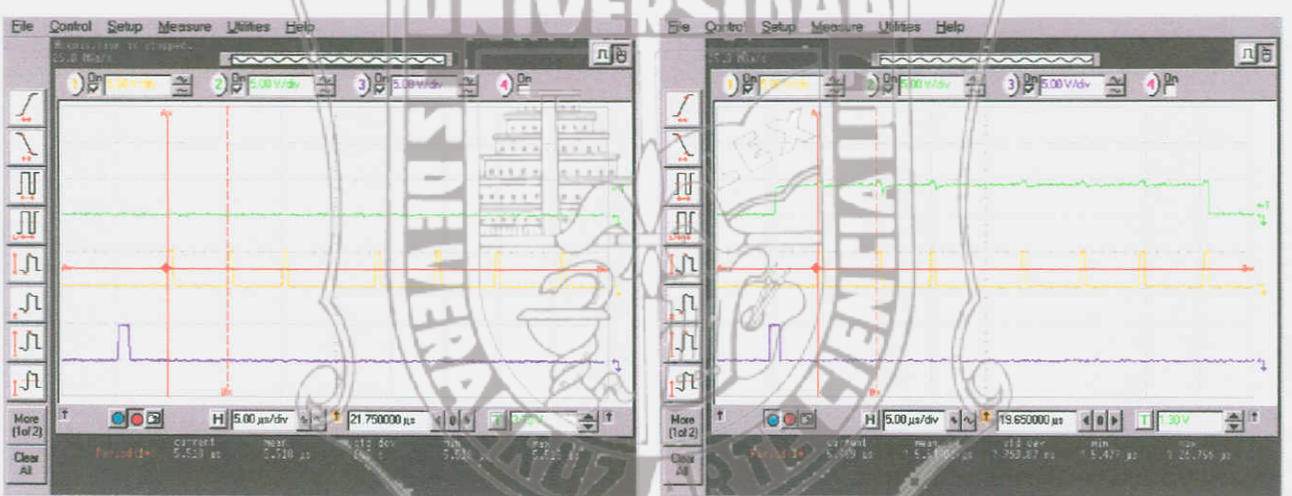


*Figura 4.1 Dos señales diferentes de dato de entrada (Posición Deseada)

***NOTA:** Todas las figuras que se ilustran fueron pruebas reales que se muestrearon con el Osciloscopio HP Infinium, directamente del circuito físico en el laboratorio.

4.1.2 Posición Actual

La posición deseada es comparada con la de posición actual, la cual se da por retroalimentación por medio del desplazamiento angular de la flecha del motor. En la figura 4.2 se muestra la señal analógica que varía de 0 a 5 V aproximadamente y que representa tal desplazamiento, ésta es introducida a un convertidor análogo-digital para ser tratada por el microcontrolador en forma digital. Al igual que el dato de posición deseada, el dato de posición actual también se almacena en un registro del microcontrolador. La señal de retroalimentación es generada por un sensor de posición (Full 360° Smart Position Sensor, de la compañía Spectrol) el cual va acoplado al eje del motor.

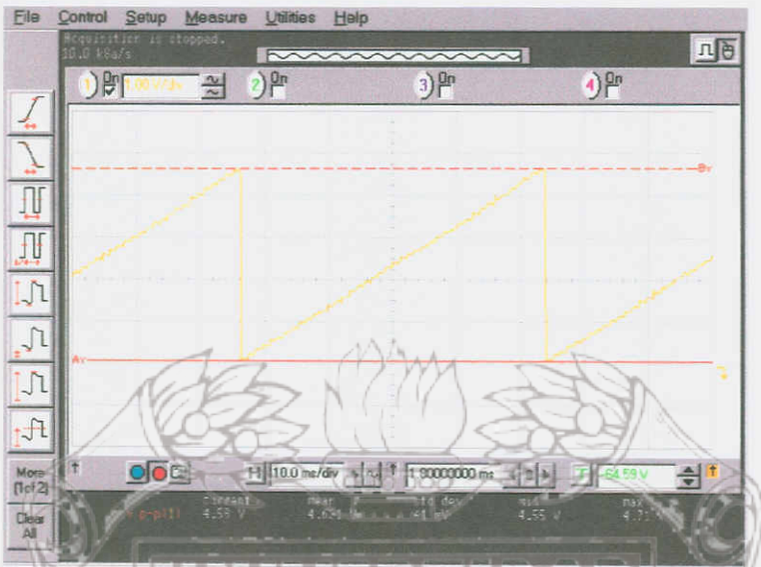


Figura 4.2. Señal medida del sensor de posición Full 360° Smart Position Sensor.

Una vez almacenados los datos de las dos posiciones (Deseada y Actual) en los registros correspondientes, estos son tomados y a través de operaciones aritméticas en el microcontrolador se calcula el error, su magnitud y signo.

A través de la diferencia o el error obtenido a partir de la comparación procesada en el microcontrolador se obtiene la señal PWM, la cual es encargada de corregir esta diferencia.

4.2 SEÑAL PWM

Una de las metas principales dentro del programa del microcontrolador, fue la de lograr la señal PWM (Modulación por Ancho de Pulso), la cual se encarga de proporcionar la energía al motor, esta señal se muestra a continuación en uno u otro sentido dependiendo de la magnitud y signo del error:

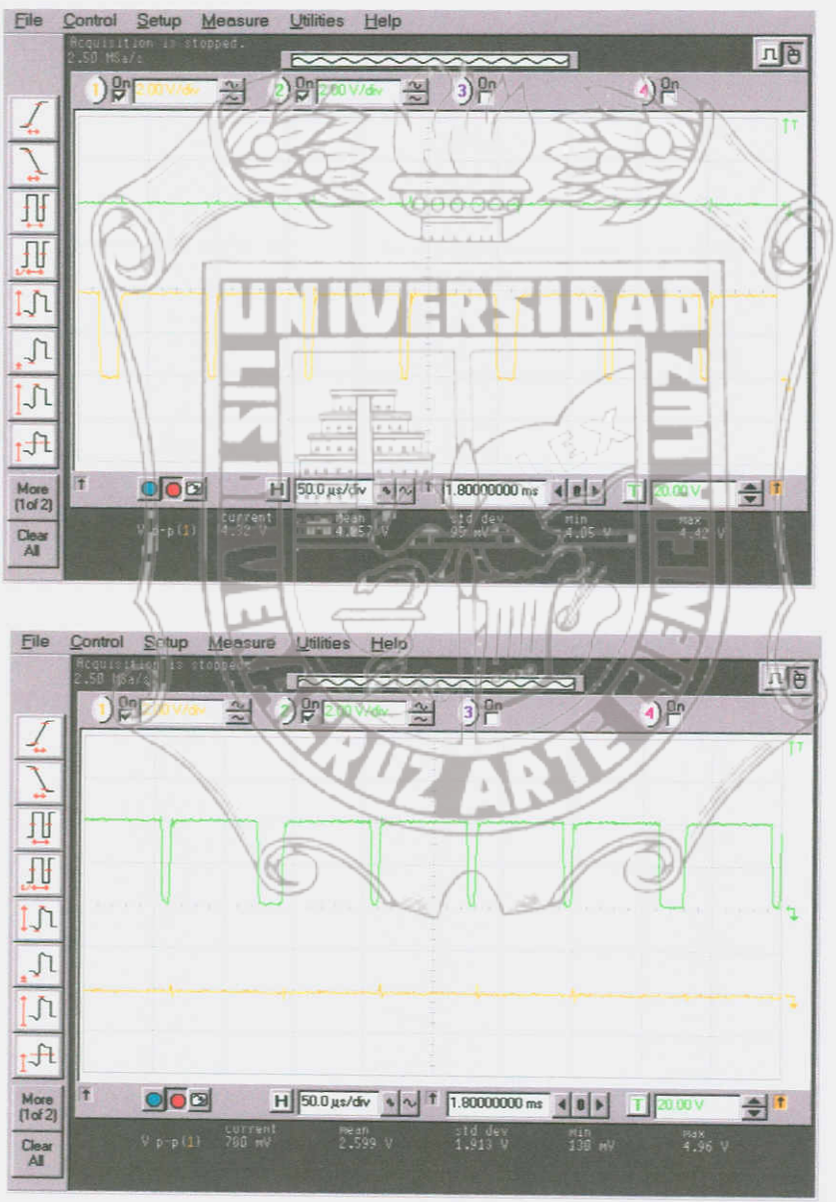


Figura 4.3 Señales medidas de los puertos de salida del microcontrolador cuando la posición deseada es mayor y menor que la posición actual respectivamente.

Cuando la posición deseada está muy alejada de la posición actual, la magnitud de la parte alta del ciclo de trabajo de la señal PWM es de casi 95% del mismo, tal y como se ve en la figura 4.3. Sin embargo, cuando la posición actual se acerca a la posición deseada (el error tiende a cero), la magnitud de la parte alta del ciclo de trabajo se hace menor. Esto indica que se necesita menos energía para el motor, ya que la posición deseada está siendo encontrada; esto se puede ver en la figura 4.4:

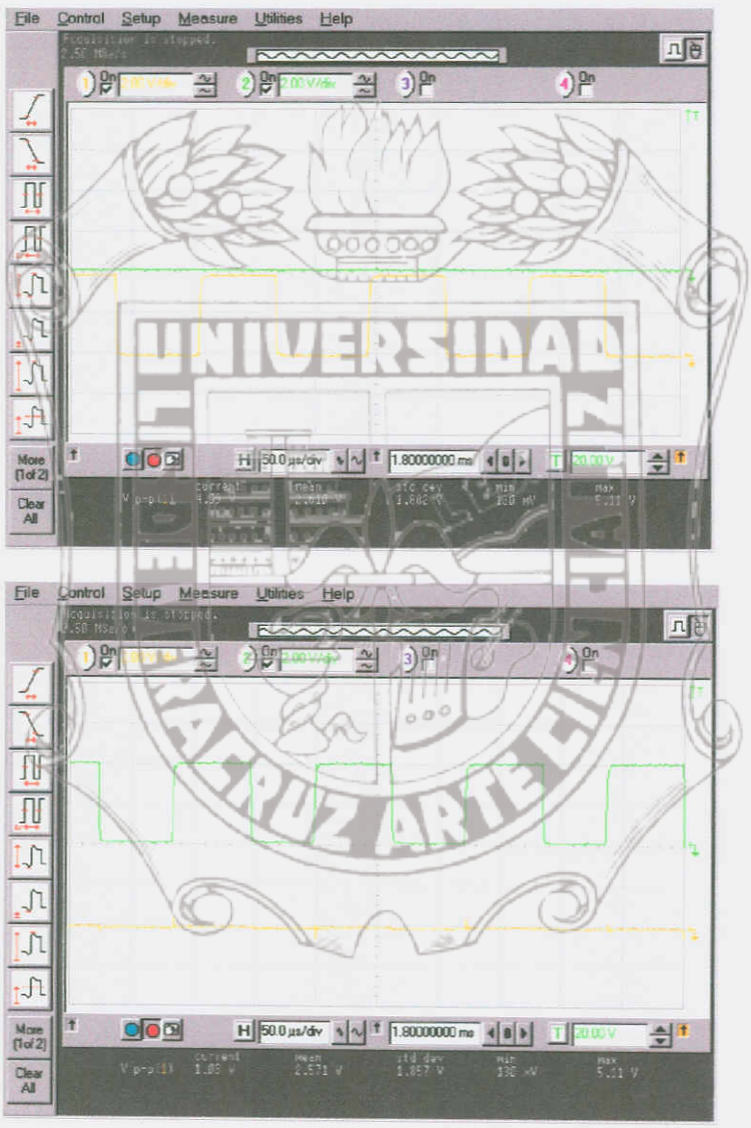


Figura 4.4 Señales medidas de los puertos de salida del microcontrolador cuando la posición actual se acerca a la posición deseada.

4.3 PROTECCIÓN DE CORRIENTE EXCESIVA

Como se puede observar en las gráficas (Figura 4.5) cuando el tiempo sobrepasa los 7 segundos (gráfico verde), se multiplica con el gráfico amarillo y el resultado es el gráfico morado, el sistema se inhibe, provocando por lo tanto que el motor sea desactivado. Esto se logra activando o poniendo una bandera a uno (gráfico morado), la cual es procesada a través del microcontrolador, el encargado de realizar dicha tarea basada en programación.

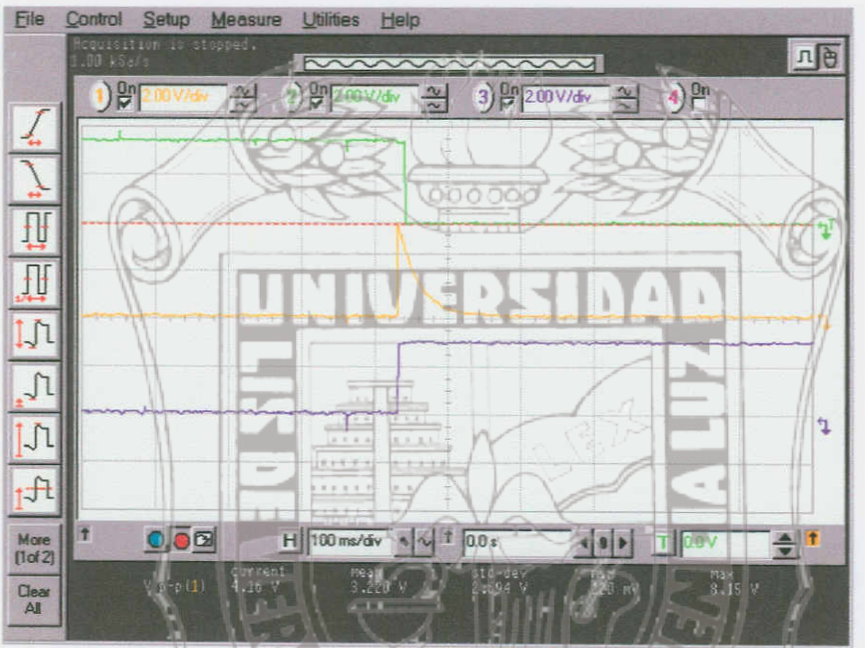
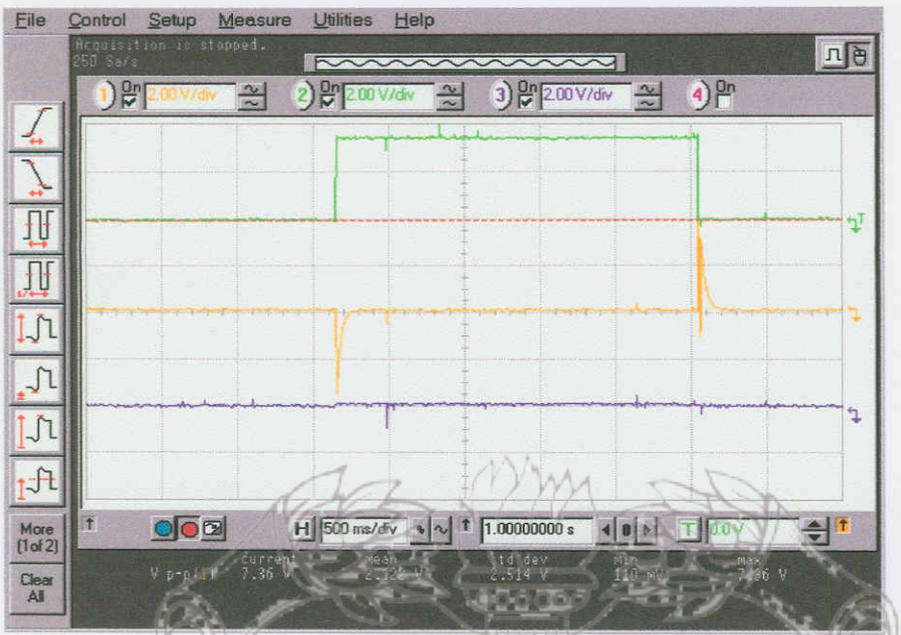


Figura 4.5 Señal medida del sensor de corriente (circuito físico)

Cuando el tiempo de sobrecorriente no es excedido, esta bandera se mantiene en cero, la misma es interpretada como una condición normal para el sistema de protección. En la figura 4.6 observamos en el gráfico verde que el pico de corriente dura aproximadamente 3 segundos, el gráfico amarillo nunca está en nivel alto dentro de ese periodo, por lo tanto, el gráfico morado que es el encargado de desactivar el sistema, se mantiene en un nivel bajo.



Después de haber deshabilitado el sistema por sobrecorriente y después de haber detectado o arreglado el problema que la causó, es necesario restablecer el sistema para que reinicie su operación. En la gráfica (Figura 4.7) podemos ver claramente como la bandera (Gráfico morado)que estaba en uno es puesta a cero a través del botón de restablecimiento del sensor de corriente.

Figura 4.7 Señal medida del sensor de corriente (circuito físico)

4.4 DIFERENTES VISTAS DEL CONTROLADOR

Con la finalidad de comprobar la comunicación en forma remota entre un periférico (computadora) y el controlador, se utilizó el software Lab View™ el cual a través de una función del programa, muestra una ventana para seleccionar o indicar la posición deseada del sistema. La comunicación con el sistema es por medio del puerto paralelo de una computadora y se logró la comunicación con un cable de aproximadamente 50 cm.

Lo anterior sólo fue establecido con fines de simulación, ya que no es costeable dedicar un equipo de computo para proporcionar con éste la posición deseada del sitema.

En las siguientes imágenes se observa el prototipo final del controlador de posición.

Foto 1.- Vista del sistema de control conectado al puerto paralelo de la computadora

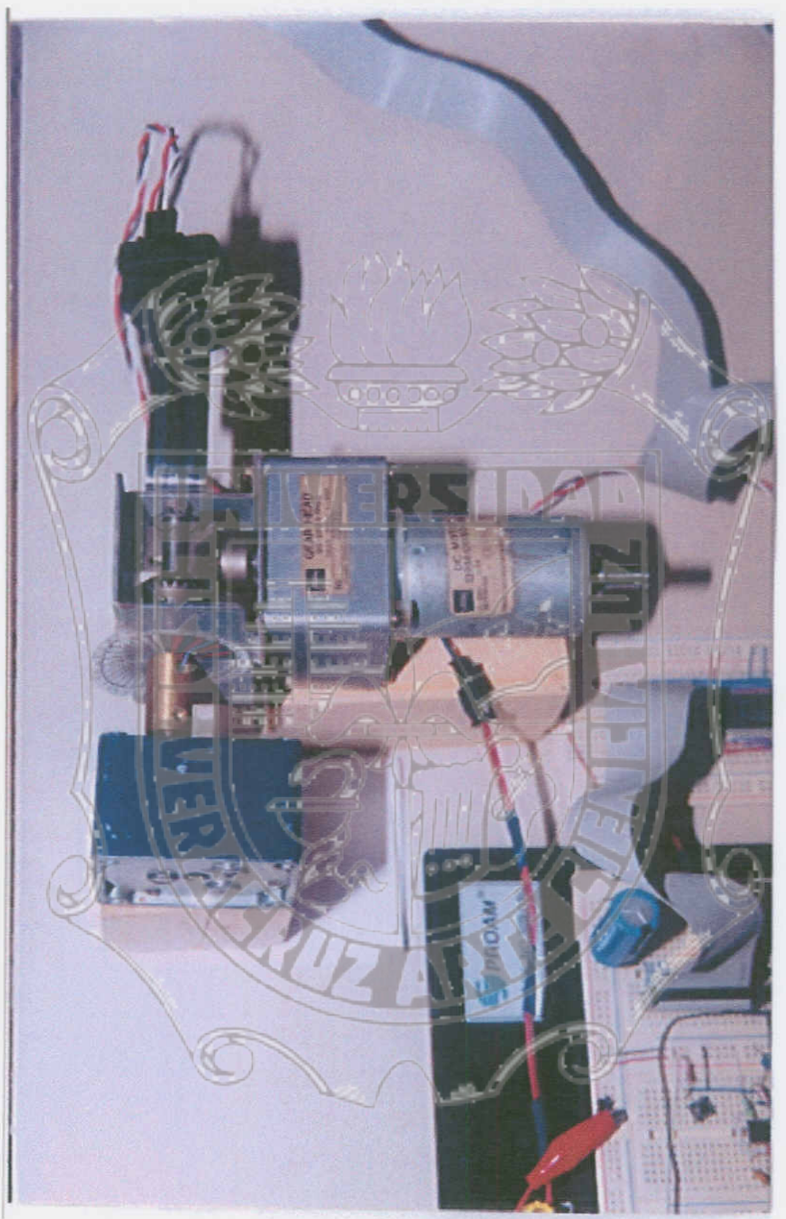


Foto 2.- Vista del motor acoplado a una caja de reducción que simula una válvula.

4.5 LOS EFECTOS DE LOS PARÁMETROS PID

Como se mencionó en el capítulo I, los controladores Proporcional, Integral y Derivativo, influyen directamente en la estabilidad de un sistema de control en lazo cerrado, cada uno de manera diferente y con lo valores adecuados, logrando de manera conjunta un sistema óptimo y aceptable para aplicaciones del tipo industrial.

En el desarrollo de este proyecto se realizaron pruebas con diferentes valores de los parámetros del controlador PID, hasta alcanzar la estabilidad adecuada del sistema. Debido a que la variable a controlar es la posición medida en grados, se presentan los datos que se obtuvieron de dicha prueba.

Ganancia	Observaciones
*KP=30	El tiempo de subida para alcanzar el set point es relativamente rápido, la oscilación que se obtiene es de aproximadamente de ±10° sobre el mismo.
*KI=2	
*KD=20	

Ganancia	Observaciones
*KP=3	El tiempo de subida para el set point se reduce ligeramente y la oscilación resultante es de aproximadamente de ±4° sobre el mismo.
*KI=15	
*KD=2	

Ganancia	Observaciones
*KP=90	El tiempo de subida para el set point se hace más lenta y la oscilación es prácticamente nula.
*KI=40	
*KD=10	

* Los valores de las constantes KP, KI y KD están expresados en formato hexadecimal.

Donde:

KP: Ganancia Proporcional
KI: Ganancia Integral
KD: Ganancia Derivativa

El tiempo de subida para alcanzar el set point, es equivalente al tiempo o velocidad de cierre, apertura o a la búsqueda de una determinada posición de la válvula. Para un sistema donde controla el flujo de vapor de agua, combustible, etc., no es recomendable que se obstruya bruscamente por la válvula, ya que podría ocasionar daños a la tubería o cualquier otro mecanismo de la misma. Por lo tanto un tiempo de subida lento nos proporcionaría la seguridad adecuada para evitar este posible accidente.

Con respecto a la oscilación, es importante remarcar que no siempre se logra tener una oscilación totalmente nula, ya que no sólo depende de los parámetros de controlador PID, sino que también jugarán un papel importante los parámetros del motor, como los huelgos en los engranes (por citar un ejemplo), incluso la misma carga podría afectar en la estabilidad u oscilación del sistema.

Con esta prueba se refleja un resultado satisfactorio para el objetivo propuesto en dicho proyecto.

CONCLUSIONES

Como consecuencia de establecer un lazo de control que midió una variable denominada Posición Actual y la comparó con otra denominada Posición Deseada, fue posible controlar el desplazamiento angular en el motor; el puerto paralelo de una computadora se utilizó para establecer comunicación a distancia (aproximadamente un metro) con el sistema. Para satisfacer esta prueba se utilizó el software LabView™ el cual fungió como interface entre la computadora y el sistema.

Se observó que el sistema cuenta con la posibilidad de controlar nuestra variable principal (desplazamiento angular) entre cero y trescientos sesenta grados pero, como se trata de un controlador para el cierre y apertura de una válvula, el sistema fue manipulado para controlar de cero a noventa grados.

En lo que respecta al control del desplazamiento angular, se observo también que éste dependió principalmente de la asignación de los parámetros Kp, Ki y Kd, para lo cual fue necesario aplicar el método de sintonización de un PID.

En la tabla comparativa 4.5 se aprecian los valores óptimos de dichos parámetros. Al establecer Kp = 90, Ki = 40 y Kd = 10 se logró efectuar un control casi perfecto con variaciones mínimas debidas al acoplamiento mecánico. Así mismo la exactitud está en función de la apreciación de las lecturas efectuadas en la carátula de medición ubicada en la flecha del motor.

En las gráficas obtenidas en las figuras 4.6 a la 4.8 se aprecia la desactivación del sistema por exceso de corriente; condición importante que se estableció también como parte de los logros del proyecto.

Con lo anterior expuesto se concluye que la investigación fue satisfactoria en lo que respecta a los objetivos establecidos.