IMPLEMENTACIÓN GRADUAL DE PRUEBAS DE LABORATORIO, PARA LA PROGRAMACIÓN DE TRES DIFERENTES VERSIONES DE CUENTARREVOLUCIONES, MEDIANTE UN MICROCONTROLADOR PIC16F84

Jaime Gómez Gil[1], Jose Fernando Díez Higuera[2], Francisco Javier Díaz Pernas[3], Ignacio de Miguel Jimenez [4]

[1]Universidad de Valladolid. jgomez@tel.uva.es
[2]Universidad de Valladolid. josdie@tel.uva.es
[3]Universidad de Valladolid. pacper@tel.uva.es
[4]Universidad de Valladolid. ignmig@tel.uva.es

RESUMEN

Este artículo muestra una práctica completa de laboratorio con el microcontrolador PIC16F84, en la que se programan diferentes tipos de cuentarrevoluciones. La práctica está organizada con pruebas intermedias hasta llegar a los resultados finales. Esto facilita la labor al alumno a la hora de depurar errores, y le motiva, ya que puede comprobar resultados tangibles parciales antes de llegar al resultado final.

En esta práctica, el alumno comprueba el funcionamiento de las puertas de E/S y TIMER del microcontrolador, así como el de los diodos *led*, *displays* de 7 segmentos y motores servo de posición.

1. INTRODUCCIÓN

La enseñanza de los microcontroladores está cada vez más presente en las ingenierías actuales. Los motivos son varios. Por una parte, a la hora de comprender el funcionamiento del sistema digital por excelencia que es el microprocesador, si queremos hacerlo con un producto que esté actualmente en el mercado, es más fácil mediante un sencillo microcontrolador que mediante un complejo microprocesador. Por otra parte, con los nuevos microcontroladores RISC, es mucho más sencillo programar en ensamblador un microcontrolador que un microprocesador. Además, es muy poco probable que en el posterior trabajo profesional, el alumno tenga que programar en ensamblador un microprocesador, mientras que sí hay ingenieros que trabajan programando en ensamblador microcontroladores. Y por último, los alumnos se motivan más cuando una aplicación real basada en microcontrolador es el objeto de su trabajo, que cuando lo es la simulación en un ordenador de un programa escrito en ensamblador.

Debido a esto, en este artículo se presenta una práctica de laboratorio basada en el microcontrolador PIC16F84. Se pretende medir ópticamente la velocidad de giro de un eje mediante un detector de corte a través de cual pasa una rueda dentada, y presentar la información de dicha velocidad de tres formas diferentes:

- Versión VISUAL: Gráficamente mediante un conjunto de *leds*, de forma que el número de *leds* encendidos es proporcional a la velocidad.
- Versión PRECISA: Numéricamente mediante varios *displays* de 7 segmentos.
- Versión CLÁSICA: Con un segmento circular escalado sobre el que gira una aguja que es movida por un motor servo de posición.

Esta práctica está enfocada como la primera toma de contacto de los alumnos con los microcontroladores. Partiendo de unos simples conocimientos presumiblemente impartidos en clases teóricas, al finalizar esta práctica, el alumno deberá ser capaz de diseñar y programar con soltura cualquier aplicación simple basada en el microcontrolador PIC16F84.

2. ORGANIZACIÓN EN EL LABORATORIO

El material que requieren poseer los alumnos es un disquete y un microcontrolador PIC16F84A-04I/P [1]. El laboratorio donde se desarrollan estas prácticas consta de 15 ordenadores PIV 800MHz en los que está instalado MPLAB IDE v6.4 [3]. En estos ordenadores los alumnos en grupos de dos, escriben y ensamblan el código de cada una de las pruebas.

Figura 1: a) Los alumnos en grupos de dos desarrollan el código de la prueba. b) Puesto de grabación con grabador SCHAER. c) Los alumnos insertan el microcontrolador en el circuito. d) Los alumnos accionan la manivela que hace girar el eje. e) Prueba de cuentarrevoluciones en versión VISUAL. f) Prueba del cuentarrevoluciones en versión PRECISA. g) Prueba de cuentarrevoluciones en versión CLÁSICA.

Cuando un grupo ha desarrollado el código de una prueba, graban el ejecutable *.hex en el disquete (*Figura 1.a*), y se dirige al puesto de grabación que consta de otro PC con un rápido grabador SCHAER. En dicho PC introducen el disquete, en el grabador introducen su microcontrolador, y mediante el software IC-Prog v1.4 en menos de un minuto tienen programado el microcontrolador con su prueba (*Figura 1.b*),.

Se dirigen entonces a la zona de pruebas en la que localizan el entrenador que tiene montada su prueba, insertan en el zócalo de fuerza de inserción nula su microcontrolador (*Figura 1.c*), y prueban el funcionamiento utilizando una maqueta (*Figura 1.d*) que internamente tiene el eje giratorio con una rueda dentada. El giro del eje con la rueda dentada es proporcional al giro de la rueda de la maqueta, y a la máxima velocidad que prudencialmente podemos hacer girar la rueda, conseguimos velocidades del eje comprendidas entre 0 y 200 vueltas por segundo.

3. ENUNCIADO DE LA PRÁCTICA PROPORCIONADA A LOS ALUMNOS

Disponemos de un eje que gira con velocidades comprendidas entre 0 y 200 vueltas por segundo, y del que queremos medir dicha velocidad.

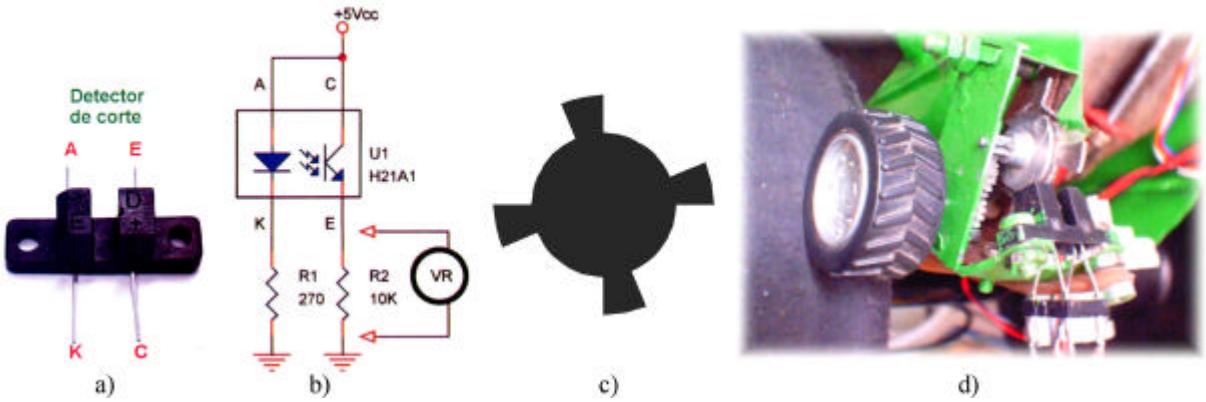

Figura 2: a) Detector de corte H21A1. b) Conexión recomendada para este detector de corte. c) Geometría de la rueda dentada. d) Disposición del detector de corte y rueda dentada en nuestra implementación.

A dicho eje le acoplamos una rueda con cuatro dientes, que los haremos pasar a través de un detector de corte (*Figura 2.a*), que consiste en un componente electrónico que posee un diodo *led* enfrentado a un fototransistor (*Figura 2.b*). Pretendemos que un microcontrolador esté continuamente midiendo la velocidad a la que gira ese eje, y que presente la información medida de tres formas diferentes:

Versión ***VISUAL***

El microcontrolador ofrece información de la velocidad de giro mediante una barra de *leds*. Si todos los *leds* están apagados, es que el eje está parado. Cuando el eje gira con velocidades pequeñas se encenderán unos pocos *leds*, y cuando el eje gira con velocidades elevadas, se encenderán muchos *leds*. La velocidad de giro será proporcional al número de *leds* encendidos.

Versión ***PRECISA***

El microcontrolador ofrece información numérica de la velocidad de giro mediante *displays* de 7 segmentos. (*Figura 1.f*).

Versión ***CLÁSICA***

El microcontrolador ofrece información similar a la de un cuentarrevoluciones de aguja analógico de un automóvil. (*Figura 1.g*).

Un posible algoritmo mediría el número de pulsos detectados durante 1 segundo, y posteriormente, dividiendo ese número entre 4, se obtendrían las revoluciones por segundo. Pero esto plantearía el siguiente problema; el número de pulsos detectados en un segundo sería un número entre 0 y 800 (200vueltas×4pulsos/vuelta) y ese número no se puede almacenar en un único registro de 8 bits como los que posee el PIC16F84A. Por ello, se muestrearán los pulsos entrantes durante la cuarta parte de un segundo, y como el número de dientes de la rueda es 4, ese número serán las revoluciones por segundo.

Tanto para facilitar como para hacer más ameno y gratificante el trabajo al alumno, esta práctica se divide en pequeñas y sencillas pruebas, hasta llegar a los diseños definitivos del cuentarrevoluciones digital.

3.1 Prueba primera

En el circuito real, el detector de corte está conexionado tal y como se muestra en la *Figura 2.a*. Identificar el punto E y mediante un voltímetro, medir la tensión en ese punto cuando un diente corta el haz de luz infrarroja y cuando el diente no le corta.
Anotar y justificar los valores medidos.

3.2 Prueba segunda

Dado el circuito de la *Figura 3*, realizar el organigrama y un programa en ensamblador MPASM que copie el contenido de la entrada RA4 en RB0.

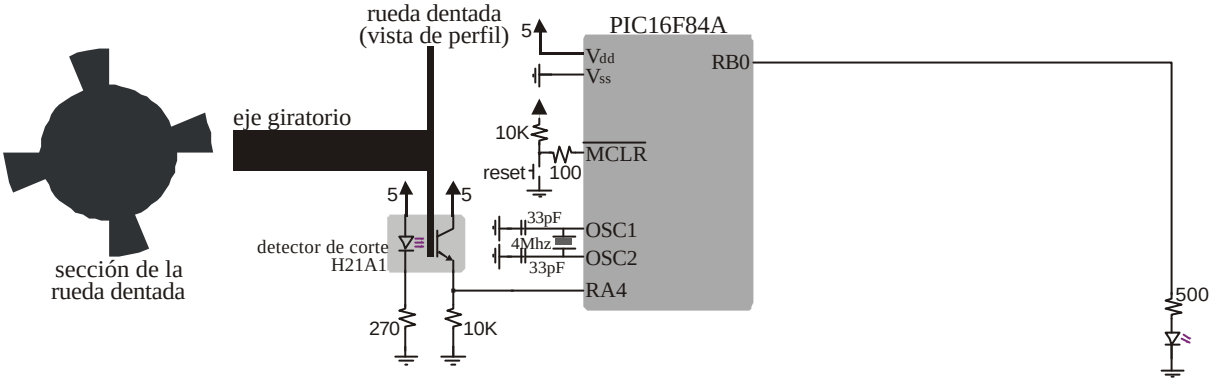

Figura 3

Girando despacio el eje, comprobar que el diodo conectado a RB0 se enciende y se apaga según la rueda dentada corta el haz infrarrojo en el detector de corte.

3.3 Prueba tercera

Disponemos del circuito mostrado en la *Figura 4*.

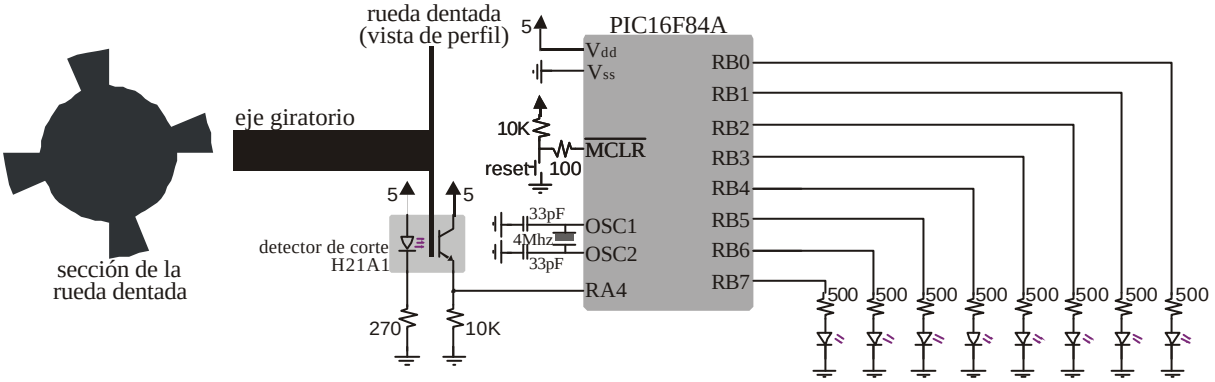

Figura 4.

Realizar el organigrama y un programa en ensamblador MPASM para que los diodos *led* se iluminen según la siguiente secuencia, a cada paso de un diente de la rueda.

00000000
00000001
00000011
00000111
00001111
00011111
00111111
01111111
11111111

11111110
11111100
11111000
11110000
11100000
11000000
10000000
vuelta al primer estado

Es decir, inicialmente los ocho *leds* están apagados hasta que el primer diente corta el detector, momento en el que se enciende el *led* menos significativo conectado a RB0. Cuando un segundo diente corta el haz, se deberán encender los *led* conectados a RB0 y RB1, y así sucesivamente.

3.4 Prueba cuarta

Resolver el mismo problema que en la prueba anterior, pero ahora utilizando una tabla de 32 columnas, es decir, un conjunto de 32 instrucciones en una subrutina del tipo *retlw '00001111'*, precedidas por una instrucción del tipo *addwf PCL,1*. Evidentemente, a la subrutina se la llamará con una instrucción *call subrutina.* Anotar en la memoria tanto el organigrama como el programa realizado.

3.5 Prueba quinta

En la prueba anterior es necesario girar la manivela muy despacio para poder percibir el movimiento de las luces en los *leds*. Modificar ligeramente el organigrama y programa anterior para que los cambios de estado se realicen con el paso de 5 dientes en vez de con uno.

3.6 Prueba sexta

Se dispone del circuito mostrado en la *Figura 5*.

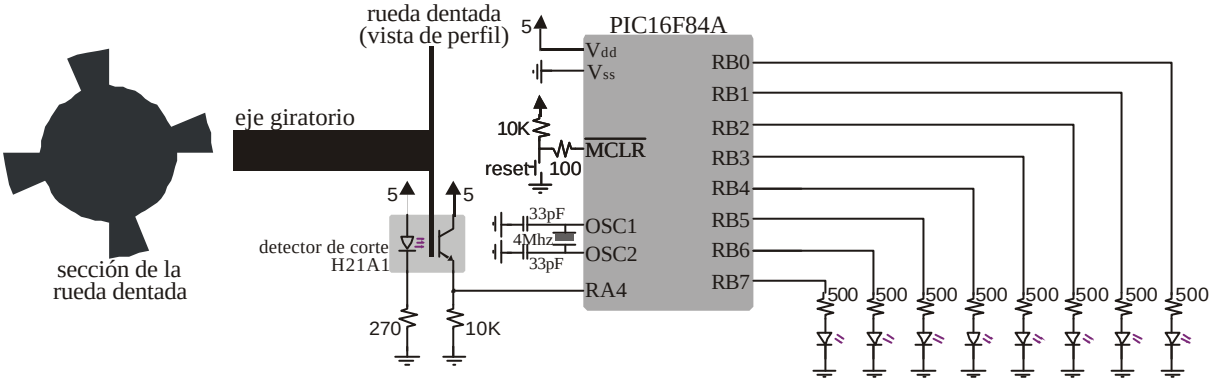

Figura 5

Realizar el organigrama y un programa en ensamblador MPASM que inicialmente configure como entrada al TIMER los impulsos recibidos por la patilla RA4 sin ningún tipo de preescalado, y que posteriormente se mantenga en un bucle infinito copiando el contenido del registro TMR0 en el registro PORB. ¿Qué se observa? ¿por qué?

3.7 Prueba séptima

En la prueba anterior es necesario girar el mando muy despacio para poder percibir el movimiento de las luces en los *leds*. Modificar el organigrama y programa anterior, asignando a TMR0 el *prescaler* con una división de frecuencia de 16:1. ¿Qué se observa? ¿por qué?

3.8 Prueba octava

Se dispone del circuito mostrado en la *Figura 6*.

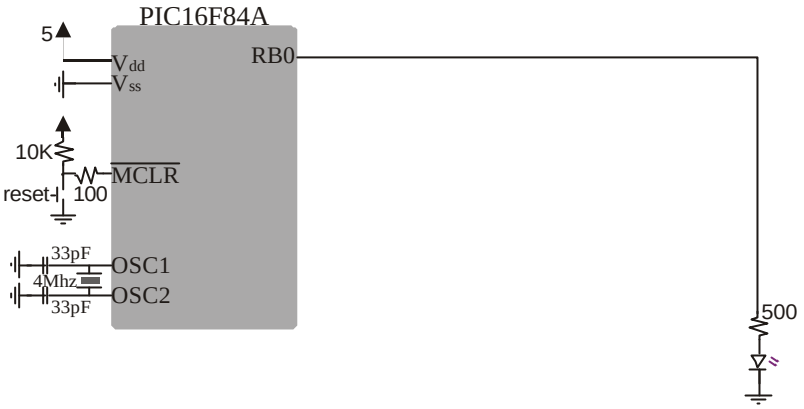

Figura 6

Realizar el organigrama y un programa en ensamblador MPASM que haga encenderse y apagarse el *led* cada 0’25 segundos (se permite un error del 1%). La temporización se hará mediante la llamada a una subrutina que contendrá bucles anidados, es decir, no se realizará con ayuda del TIMER.

3.9 Prueba novena

Se dispone del circuito mostrado en la *Figura 7*.

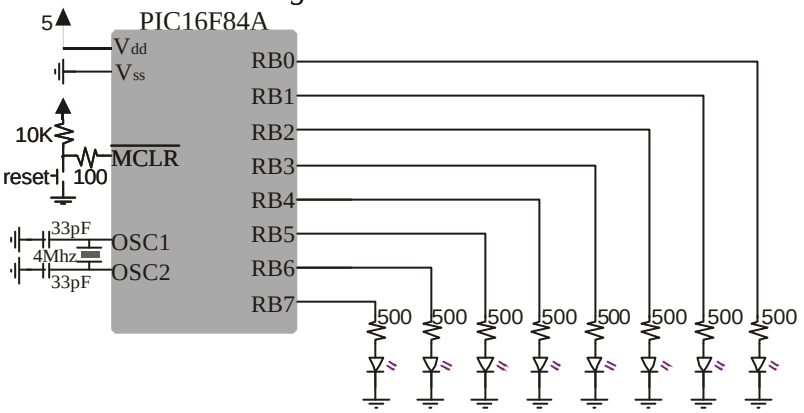

Figura 7

Realizar el organigrama y un programa en ensamblador MPASM que realice un encendido de estos *leds* con la animación del coche fantástico y el retardo en el encendido de luces del programa anterior (0’25segundos). Comprobar que la luz tarda en desplazarse de un extremo a otro dos segundos (para ello, los extremos deben encenderse dos veces, es decir, se deben completar cada ciclo en ocho fases de 0’25 segundos).

3.10 Prueba décima

Realizar la prueba anterior, pero ahora ayudándonos en la temporización por el registro TIMER.

3.11 Prueba undécima

Se dispone del circuito mostrado en la *Figura 8*.

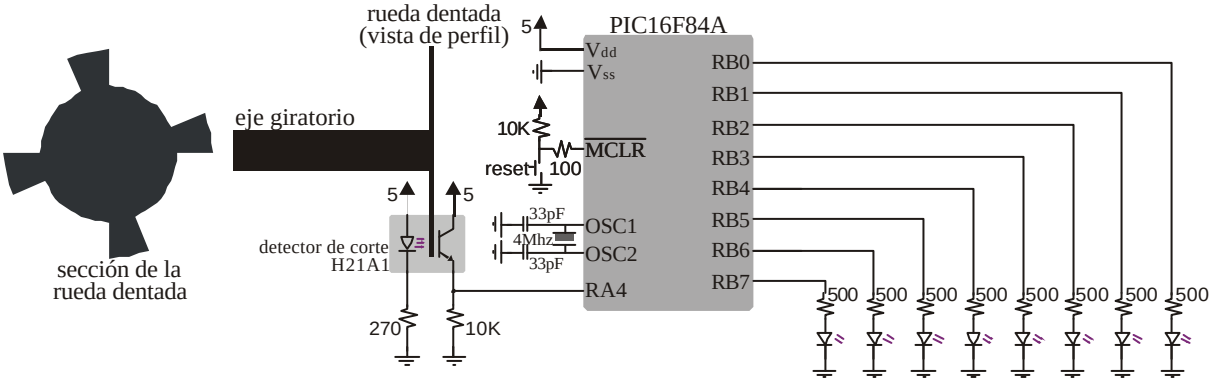

Figura 8

Realizar el organigrama y un programa en ensamblador MPASM que haga lo siguiente:

1. Configurar a la entrada de TMR0 los impulsos generados en la patilla RA4 sin escalado.
2. Poner el TMR0 a cero.
3. Temporizar 0’25 segundos.
4. Mostrar en PORTB el contenido de TMR0.
5. Volver al paso 2.

Nota importante: Debido a que el PIC16F84A sólo posee un TIMER y lo utilizamos para contar los impulsos entrantes por RA4, la temporización se deberá hacer sin ayuda del TIMER.

3.12 Prueba duodécima

Se dispone del circuito mostrado en la *Figura 9*.

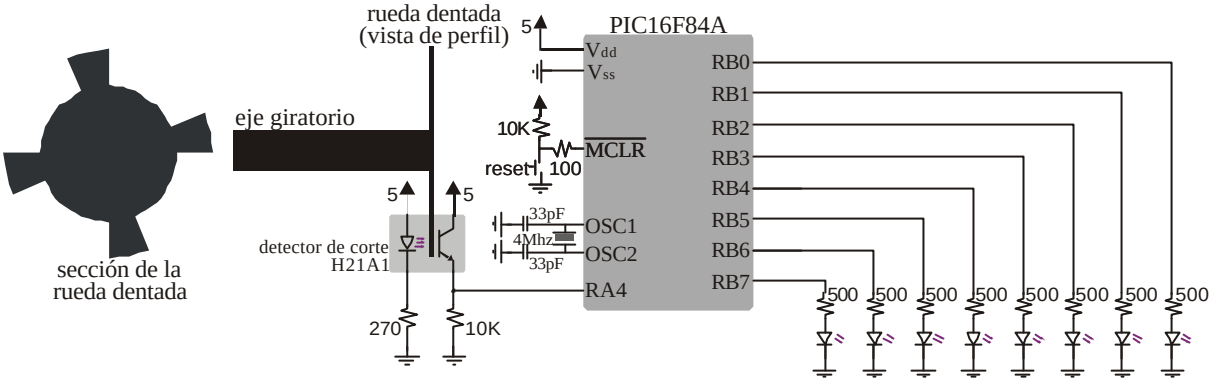

Figura 9

Realizar el organigrama y un programa en ensamblador MPASM que haga lo siguiente:

1. Configurar a la entrada de TMR0 los impulsos generados en la patilla RA4 sin escalado.
2. Poner el TMR0 a cero
3. Temporizar 0’25 segundos.
4. Según el contenido de TMR0 mostrar en PORTB lo siguiente:
    a. Si TMR0=0 entonces PORTB=00000000
    b. Si 1≤TMR0≤25 entonces PORTB=00000001
    c. Si 26≤TMR0≤50 entonces PORTB=00000011
    d. Si 51≤TMR0≤75 entonces PORTB=00000111
    e. Si 76≤TMR0≤100 entonces PORTB=00001111
    f. Si 101≤TMR0≤125 entonces PORTB=00011111
    g. Si 126≤TMR0≤150 entonces PORTB=00111111
    h. Si 151≤TMR0≤175 entonces PORTB=01111111
    i. Si 176≤TMR0 entonces PORTB=11111111
5. Volver al paso 2.

3.13 Prueba decimotercera

Se dispone del circuito mostrado en la *Figura 10*.

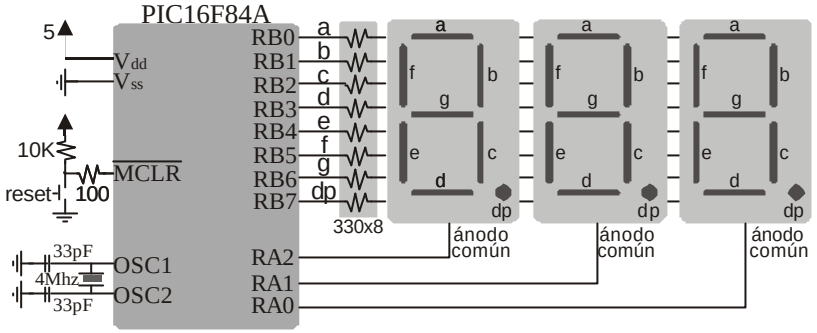

Figura 10

Realizar el organigrama y un programa en ensamblador MPASM que presente un 2 en el *display* central.

3.14 Prueba decimocuarta

Realizar la prueba anterior, pero ahora presentando un 222 (un 2 en los tres *displays*).

3.15 Prueba decimoquinta

Se dispone del circuito mostrado en la *Figura 11*.

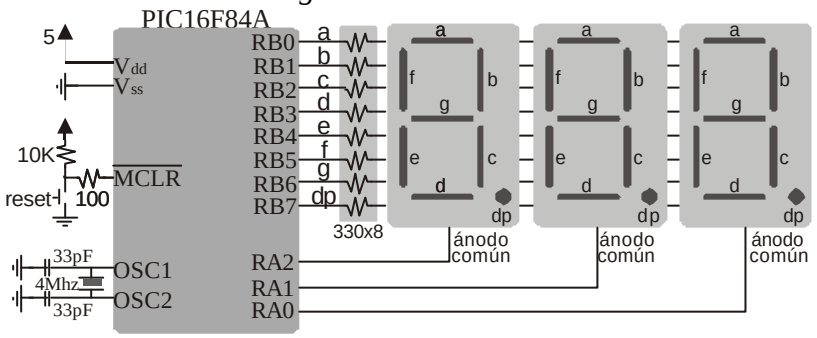

Figura 11

Realizar el organigrama y un programa en ensamblador MPASM que presente un 123 en los *displays*. Para conseguirlo se entra en un bucle en el que:

1. Se saca 1 en el dígito más significativo.
2. Retardo de 1ms.
3. Se saca 2 en el dígito central.
4. Retardo de 1ms.
5. Se saca 3 en el dígito menos significativo.
6. Retardo de 1 ms.
7. Vuelta al paso 1.

3.16 Prueba decimosexta

Se dispone del circuito mostrado en la *Figura 12*.

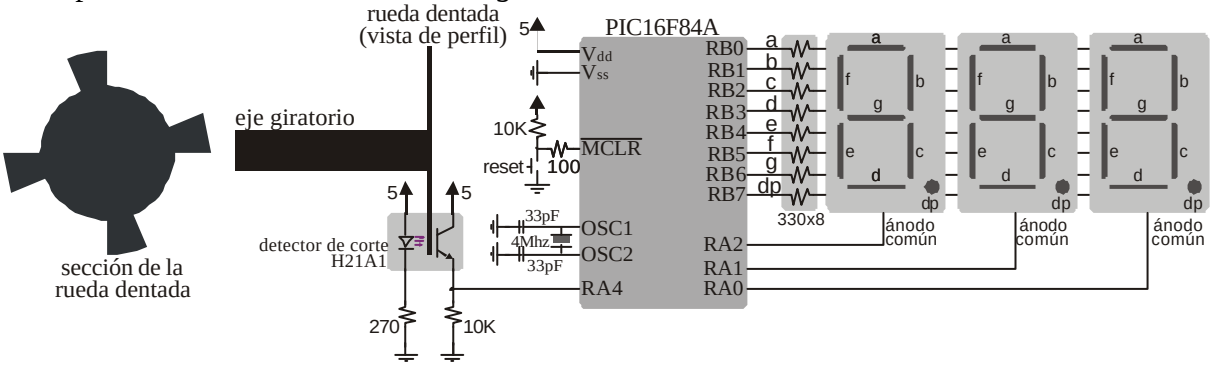

Figura 12

Realizar el organigrama y un programa en ensamblador MPASM que se mantenga en el bucle:

1. Configurar a la entrada de TMR0 los impulsos generados en la patilla RA4 sin escalado.
2. Poner el TMR0 a cero.
3. Temporizar 0'25 segundos.
4. Mostrar durante unas décimas de segundo (entre 2 y 10 décimas) el número de vueltas contenido en TMR0 en los *displays*.
5. Volver al paso 2.

3.17 Prueba decimoséptima

Con el fin de mejorar la legibilidad, se pide modificar el programa anterior para que no se presenten los ceros a la izquierda. En el caso de que la rueda esté parada se visualizará únicamente el punto del dígito menos significativo.

3.18 Prueba decimoctava

Se dispone del circuito mostrado en la *Figura 13*.

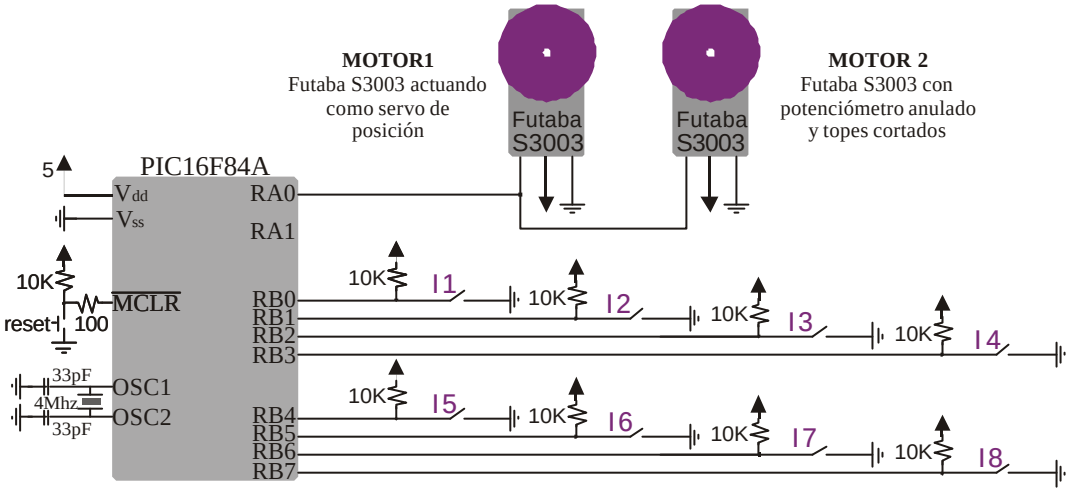

Figura 13

Se pide que el alumno se proponga una o varias pruebas para implementar con el diseño del circuito dado. El objetivo será comprender el funcionamiento de un motor servo de posición, y un servo modificado a giro continuo y gobernado por PWM. No es necesario que se utilicen todos o alguno de los interruptores, si la aplicación ideada no lo requiere. Sería interesante que el alumno emplease interrupciones a la hora de realizar la programación. En la memoria se debe explicar lo que se pretende conseguir, presentar el organigrama, y el listado del programa en ensamblador MPASM. Finalmente debe comentar los resultados obtenidos.

3.19 Prueba decimonovena

Se dispone del circuito mostrado en la *Figura 14*.

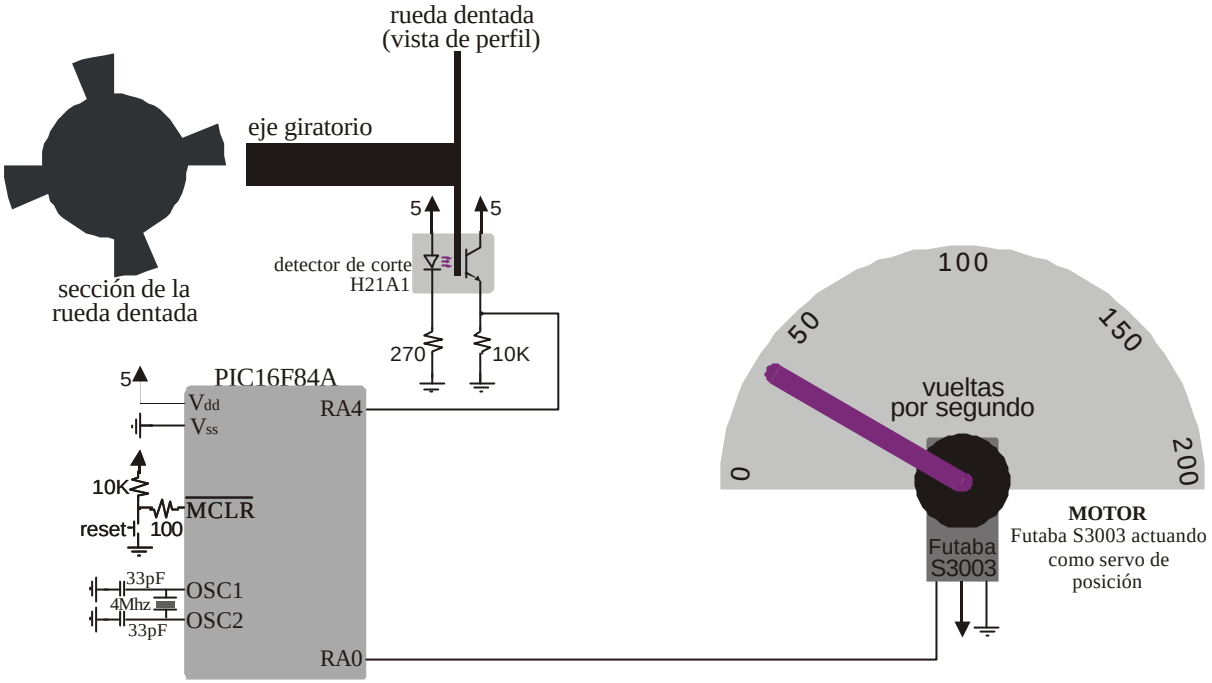

Figura 14

Realizar el organigrama y un programa en ensamblador MPASM que presente gráficamente mediante el servo, la velocidad de giro de la rueda dentada.

4. CONCLUSIONES

Esta práctica de laboratorio cumple la característica de ser gradual en su dificultad, ya que el alumno puede realizar las primeras pruebas teniendo unos conocimientos básicos del PIC16F84A, y la complejidad desde estas primeras pruebas va creciendo hasta llegar a las últimas, con la necesidad por parte del alumno de dominar la teoría de las interrupciones.

Al basarse una prueba en las anteriores, la depuración de errores se simplifica, ya que el alumno tendrá acotados los errores en la parte de código añadida respecto a la prueba anterior.

El alumno se siente motivado con la realización de esta práctica, ya que está realizando una aplicación real, y no simplemente simulando un programa en un ordenador.

5. BIBLIOGRAFÍA

[1] *PIC16F84A Data Sheet*, Microchip, 2001.
[2] *PICmicro™ Mid-Range MCU Family Reference Manual*, Microchip, 1997.
[3] *MPLAB IDE,SIMULATOR, EDITORUSER'S GUIDE*, Microchip, 2001.
[4] J.M. Angulo Usategui, E. Martín Cuenca, L. Angulo Martínez, *Microcontroladores PIC: la solución en un chip*, Paraninfo, Madrid, 1997.
[5] C. Tabernier, *Microcontroladores PIC*, Paraninfo, Madrid, 1997.