Problema (Tiempo Estimado:1h 30min. 4,5 puntos).

1.- A partir de los ángulos de medida necesarios, indique los intervalos de tiempo que es necesario generar con el Microcontrolador para posicionar el servomotor en el que se coloca el elemento sensor. Justifique cada uno de ellos y explique cómo genera dichos intervalos con el bloque de Temporizadores.

Para posicionar el servo, es necesario generar una onda cuadrada como la que se representa en la Figura 1, cuyo periodo es 20ms y el tiempo activo (Ton) determina la posición del servo.

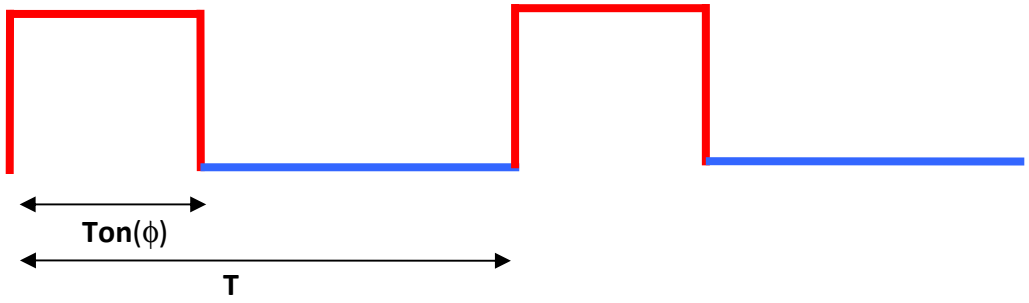


Figura 1: Onda cuadrada para el Servomotor

Generación de Ton

Las características del servomotor indican que para 0º, el ancho del pulso Ton=1,25ms y para 180º, el ancho del pulso Ton=1,75ms. Aplicando una regla de proporcionalidad lineal, y siendo ϕ l ángulo de posicionamiento en grados:

$$Ton(\phi) = Ton(0) + \phi \cdot (0.5ms / 180º)$$

Dado que los intervalos angulares que se han definido son de 30º, se obtiene la tabla siguiente:

meas_angle	Posición angular	Ton
0	0	Ton(0) = 1,25ms
1	30	1,33ms
2	60	1,41ms
3	90	1.50ms
. . . .	. . . .	. . . .
6	180	1,75 ms

Tabla 1

Generación de T

Además de estas temporizaciones, necesarias para posicionar el servo, es necesario generar el periodo de esta señal indicada, de 20ms.

Implementación de las temporizaciones

Una vez determinados los intervalos de tiempo que es necesario generar, podemos definir cómo usar los bloques de temporizadores del microcontrolador para gestionarlos.

Bloque PWM: El bloque más indicado para generar una señal cuadrada del tipo representado en la Figura 1 es el Bloque de Modulación de Ancho de Pulso (Bloque PWM).

Vamos a calcular los valores de los registros asociados, asumiendo un reloj de 8MHz, empezando por el registro del periodo de la señal. Usando la fórmula del manual, ***PWM period = (PR2 + 1) · 4 · Tosc · Prescale****, obtenemos que PR2 = 40.000 cuentas unidad. Como PR2 es un registro de 8 bits, esto no es posible, y tenemos dos opciones:*

1. *Probar con PR2, recalcular el periodo resultante, y ver si el Servomotor funciona.*
2. *Utilizar otro esquema de temporizadores, que es la solución por la que aquí optamos.*

Bloque CCP: Esta es la opción más adecuada una vez visto que el módulo PWM no nos permite escalar las temporizaciones. El módulo CCP tiene interesantes opciones que nos permiten descargar gran parte de las acciones sobre este módulo temporizador. El diagrama de bloques del módulo CCP es el siguiente:

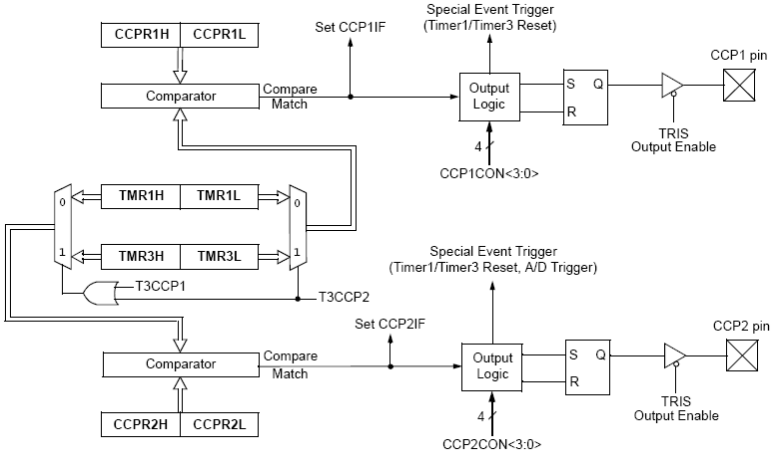


Figura 2: Módulo CCP

La idea es usar un contador (TMR1 o TMR3), de manera que cuente indefinidamente, asociándole los dos registros de comparación, CCPR. Mientras que uno de los registros CCPR almacena el número de cuentas unidad de Ton (de forma que cuando TMR=CCPR, se pone a cero el valor del pin de salida), el otro almacena el número de cuentas unidad del periodo, T (de forma que cuando TMR=CCPR, se reinicia el proceso). Esquemáticamente, lo representamos en la Figura 3.

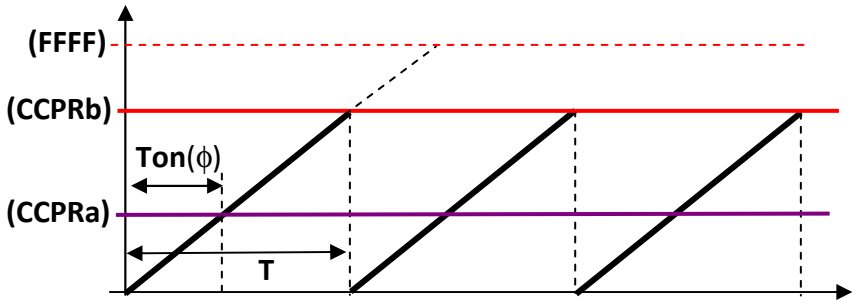


Figura 3

Sólo tenemos que elegir apropiadamente el modo de operación del módulo CCP (contenido de T3CON), y sobre qué registro (CCPR1 y CCPR2) definimos CCPRa y CCPRb. Tenga en cuenta para esto los Special Trigger Event del módulo.

Una vez que el sistema ha barrido secuencialmente todos los ángulos de medida, tomando en cada uno de ellos la medida de distancia correspondiente, éste debe reiniciar la toma de medidas volviendo al primer ángulo de medida, y comenzando de nuevo el proceso. Cada ángulo de medida se identifica mediante un número natural asignado en función del orden en que se barre y se almacena en una posición de memoria RAM (denominada **meas_angle)**.

2.- Escriba el diagrama de bloques de las subrutinas que ejecuta el microcontrolador para posicionar el sensor en el ángulo de medida que indica **meas_angle**. (Identifique cuáles de ellas son rutinas de atención a interrupción).

El programa de medida de distancias debería ser del estilo:

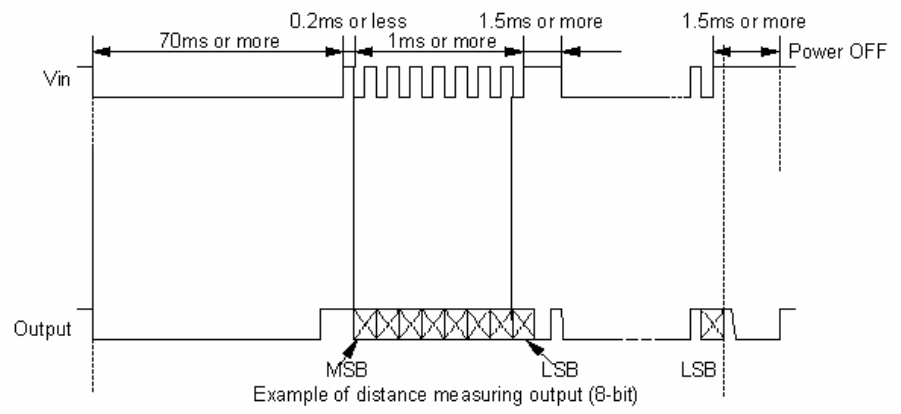
```
while(1) {
    posiciona(meas_angle);
    data = medida();
    meas_angle += 1;

    if(meas_angle == 7)
            meas_angle = 0;
}
```

***posiciona**(arg): Rutina que carga los valores en CCP para posicionar el servo en un determinado ángulo. El argumento es el valor entero que selecciona el ángulo de entre las 7 posiciones posibles dadas en la tabla.*

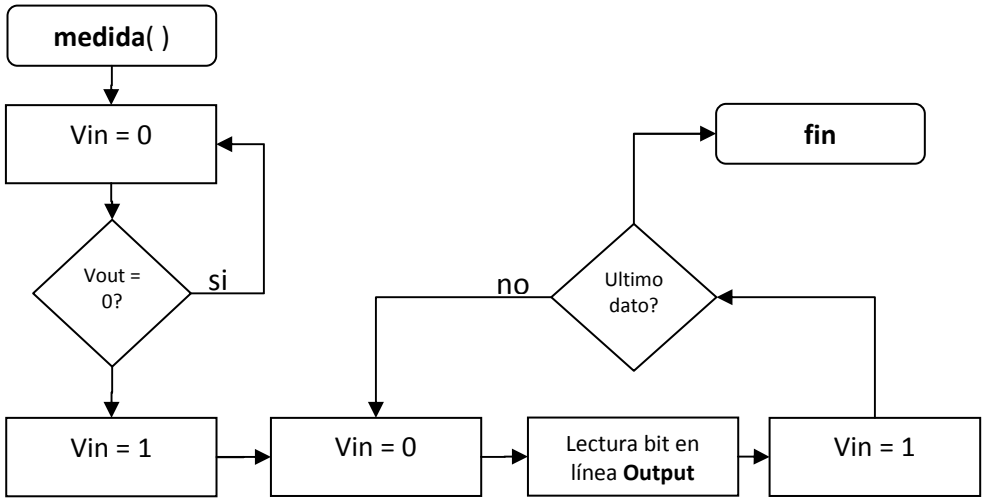
***medida**(): Rutina que obtiene un dato del sensor de distancia, y lo devuelve tras la llamada.*

3.- Una vez posicionado el sensor en el ángulo de medida anterior, el microcontrolador debe recoger la medida del sensor. Dibuje el diagrama de flujo de la subrutina que debe ejecutar el microcontrolador para disponer de la medida en el registro W de la CPU. El diagrama de flujo debe indicar claramente TODOS los pasos que debemos realizar para obtener la medida, sin ser ni muy específico ni muy general.

Tenga en cuenta para este apartado el cronograma del sensor, que reproducimos aquí para su conveniencia:



Dado que este no es una interfase Standard debido al tiempo de 70ms de espera, lo mejor es realizar una implementación software del protocolo de comunicación con el sensor.



Como última aclaración, indicar que este diagrama de flujo no contiene el tiempo de espera entre dos lecturas consecutivas, entendiéndose que es una espera que debe realizarse externamente. También se entiende que al ejecutarse las lecturas de de la línea output, el tiempo de ejecución de las instrucciones es suficiente, pero pueden incluirse bloques de espera para que el tiempo de ejecución de la lectura sea superior a 1ms como se indica.

4.- Deseamos almacenar los datos de distancias en una zona de memoria en la que cada posición contenga la última medida disponible para un ángulo de medida fijo. Escriba la secuencia de instrucciones que, a partir de la medida en el acumulador, ésta se pase a la dirección de almacenamiento.

Para este apartado, debemos disponer de una zona de memoria con tantas posiciones como ángulos en los que vamos a realizar medidas. De esta forma, consideramos que reservamos en memoria RAM la siguiente zona:

dirección	Valor (8 bits)
angle0	
angle1	
angle2	
angle3	
. . . .	
angle6	

Tabla 2

Utilizando los punteros en C, guardar un dato en la memoria requiere

```
*actual_angle = data;
```

*donde *actual_angle es un puntero a la posición donde debe almacenarse el dato de la posición angular que estamos midiendo, y data es el valor devuelto por la rutina medida(). También es recomendable indicar que dicho puntero se inicializa a angle0,*

```
*actual_angle = angle0;
```

Y que en cada iteración se actualiza su valor, hasta compararlo con angle6, momento en el que el puntero se reinicia a angle0.

5.- Realice ahora, y de forma consecuente al programa previsto en el punto 3, las conexiones pertinentes entre el dispositivo sensor de distancias y el microcontrolador.

Dado que la lectura se implementa por software, podemos utilizar cualquier pin que no esté ya ocupado (como CCP1 para el servomotor).