Las máquinas de tiempo como base del procesamiento en tiempo real con pequeños procesadores

Aitzol Zuloaga, Jaime Jiménez, Jesús Lázaro, Carlos Cuadrado, Unai Bidarte
Departamento de Tecnología Electrónica
Universidad del País Vasco (UPV/EHU)
Bilbao, España
Email: {aitzol.zuloaga, jaime.jimenez, jesus.lazaro, carlos.cuadrado, unai.bidarte}@ehu.es

***Resumen*—Habitualmente las máquinas de estados son estructuras fundamentales para desarrollar sistemas basados en pequeños procesadores y forman la base conceptual de los sistemas operativos. Sin embargo, las máquinas de estado no aportan la capacidad de procesamiento en tiempo real que requieren algunos sistemas. Para ello es necesario recurrir a otra estructura: las máquinas de tiempo. El concepto fundamental de las máquinas de tiempo permite dotar a pequeños procesadores con otro nivel de capacidad. Por esto último, el tema es tratado en el curso de "Sistemas Digitales" del Grado en Ingeniería en Tecnología de Telecomunicación, como una ampliación del tema de máquinas de estado, eventos y acciones.**

Keywords*—*Máquinas de tiempo, máquinas de estado, procesadores, funcionamiento robusto, método de diseño.

I. Introduccion

En este trabajo se introduce la máquina de tiempo como una estructura de software que, junto a las máquinas de estados y eventos, permite dotar a los pequeños procesadores de una gran capacidad para aquellas aplicaciones donde las condiciones temporales son estrictas. Además permite el máximo aprovechamiento de la capacidad de procesamiento del dispositivo.

El tema de las máquinas de tiempo se enmarca en la asignatura "Sistemas Digitales" que se imparte para la titulación de Grado en Ingeniería en Tecnología de Telecomunicación en la Escuela de Ingeniería de Bilbao [1]. Entre los principales objetivos establecidos para la docencia de la asignatura de Sistemas digitales están los siguientes:

- Conocer los principios básicos de funcionamiento interno de un procesador desde el punto de vista de máquina electrónica.
- Interiorizar la necesidad de utilizar estructuras profesionales para el desarrollo de programas para el diseño de sistemas de funcionamiento eficiente y robusto.
- Conocer el funcionamiento de una máquina de tiempo.

En la literatura las máquinas de tiempo, en sus diferentes variantes, son denominadas habitualmente como "Planificadores" [2], "Secuenciadores" [3] o "*Time Schedulers*" [4]. Aquí hemos preferido la denominación de "máquina de tiempo" porque hace que el estudiante lo relacione con el concepto de "máquina de estados" con el que guarda cierta relación.

Las máquinas de tiempo son el concepto fundamental que da pie a los sistemas operativos de tiempo real [2,4,5]. Por ello, este tema constituye un puente entre la enseñanza de los pequeños microcontroladores en los cursos de Sistemas Digitales y la enseñanza de los sistemas operativos más propia de los cursos de informática avanzada.

La gran proliferación en el mercado de pequeños sistemas de desarrollo y la gran accesibilidad de programas a través de Internet, ha propiciado que muchos jóvenes incursionen en el mundo de los microcontroladores. Sin embargo, la falta de guía adecuada crea vicios en su proceder que deben ser corregidos en las aulas con el fin de lograr un desempeño profesional en los futuros ingenieros. Es por ello que en la asignatura de "Sistemas Digitales" nos hemos propuesto a enseñar cómo utilizar estructuras de programación más eficientes y robustas. En este sentido, dentro de la temática, se abordan estructuras tales como las máquinas de estados, las máquinas de eventos y las máquinas de tiempo [6,7]. Esta temática no es extensamente abordada en las publicaciones ni en los cursos de las universidades, sin embargo, son herramientas utilizadas en el ámbito profesional.

Por otro lado, las máquinas de estado, las máquinas de eventos y las máquinas de tiempo, al ser estructuras muy concretas, permiten al profesorado una evaluación más sistemática y realista del desempeño del estudiante.

El estudio de la máquina de tiempo parte de la definición de los conceptos de "período de repetición", "ranuras" y "tareas". Aunque la estructura de las máquinas de tiempo tiene semejanza a las estructuras de las máquinas de estados y las máquinas de eventos, es necesario redefinir conceptos para una cabal comprensión de su utilidad y aplicación.

Es necesario señalar que el uso de las máquinas de tiempo no excluye el uso de las máquinas de estado y eventos. Debe entenderse que son estructuras que pueden utilizarse simultáneamente en un sistema sin interferirse mutuamente, más bien todo lo contrario.

Para la enseñanza práctica de la asignatura de Sistemas Digitales en la Escuela de Ingeniería de Bilbao, se utiliza el lenguaje ensamblador de los procesadores PIC16 dado que

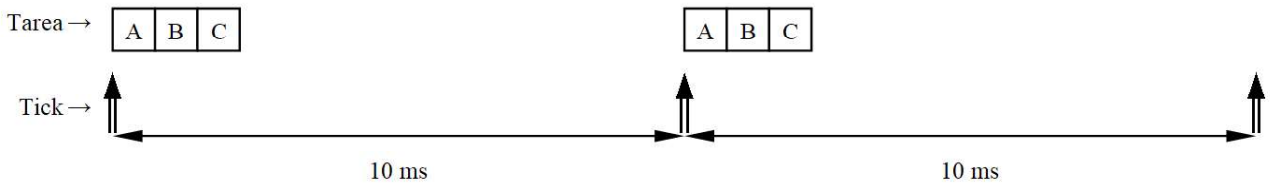

Fig. 1. Realización de 3 tareas en cada interrupción de tiempo (tick)

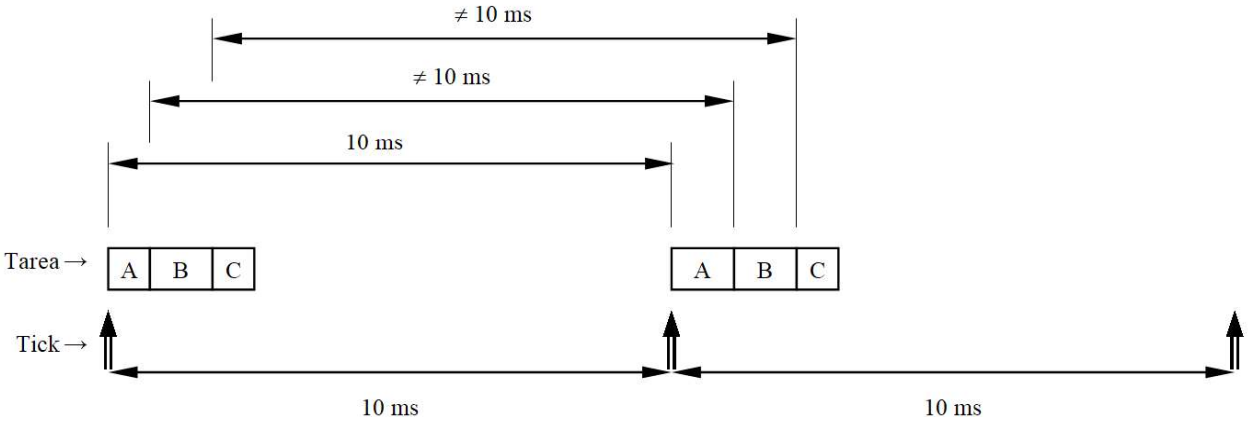

Fig. 2. Problema del incumplimiento de tiempo entre tareas

permite entender mejor el procesador desde una perspectiva de máquina electrónica [8,9,10]. Es cierto que la mayoría de los autores consideran que conceptos como la máquina de tiempo pertenecen al campo de los sistemas operativos y que éstos escapan de las posibilidades de los pequeños microcontroladores de 8 bits y en especial si se trabaja con lenguajes de bajo nivel [5, 11, 12]. Sin embargo, los conceptos básicos de los sistemas operativos se desarrollaron en procesadores de 8 bits y en lenguajes de muy bajo nivel. Además, en un lenguaje de bajo nivel se observa con una claridad especial la simplicidad de estas estructuras.

En sintonía con las prácticas, en este trabajo se exponen ejemplos de aplicaciones prácticas con procesadores PIC16, donde el uso de la máquina de tiempo permite superar, con creces, las limitaciones de los periféricos específicos del dispositivo en aplicaciones como el control de motores o servomotores.

II. Ejecución a tiempo

En los sistemas basados en procesadores se requiere con frecuencia atender ciertos dispositivos periódicamente con una relativa exactitud. Entre ellos se pueden citar los generadores de PWM, los generadores de sonido, los marcadores telefónicos.

Para atender este tipo de dispositivos, la mayoría de los microcontroladores cuenta con uno o más temporizadores y a veces con periféricos adicionales para algunos dispositivos en particular como pueden ser los generadores de PWM. En el caso de plataformas basadas en microprocesadores también es posible agregar periféricos adicionales. Sin embargo, no siempre es posible disponer de todos los periféricos requeridos o exactamente adaptados a la aplicación.

Muchas de estas aplicaciones pueden ser atendidas utilizando un temporizador y un programa debidamente desarrollado.

Hoy día los procesadores en general operan a elevadas frecuencias de reloj, por lo que son capaces de atender las exigencias que anteriormente se realizaban exclusivamente por circuitos específicamente diseñados. El mayor problema es el cumplir con la exactitud los requisitos de tiempo en que deben ejecutarse los programas.

Vamos a tratar de ilustrar los problemas y las soluciones por medio de ejemplos. Supongamos que se desean ejecutar 3 operaciones o tareas cada 10 ms. Estas tareas se denominarán *Tarea A, Tarea B y Tarea C*. Básicamente cada tarea es una subrutina. Entre la ejecución de dos *Tareas A* debe guardarse 10 ms de espacio de tiempo, al igual que entre dos *Tareas B* y dos *Tareas C*. Pero, sin embargo, la separación en tiempo entre las *Tarea A*, *B* y *C* no tiene porqué ser rigurosa.

Para cumplir con el requisito de tiempo es necesario que el procesador tenga asociado un temporizador que genere continuamente una interrupción cada cierto intervalo fijo de tiempo. Esta interrupción de tiempo, la principal del sistema, recibe el nombre de *tick*.

En el caso del ejemplo, una solución puede ser la de tener un *tick* de tiempo de 10 ms y ejecutar todas las tareas, una tras otra, tal y como se muestra en la fig. 1. El problema es que la duración de las tareas no siempre suele ser la misma, con lo cual se presenta el problema de que no existe un intervalo

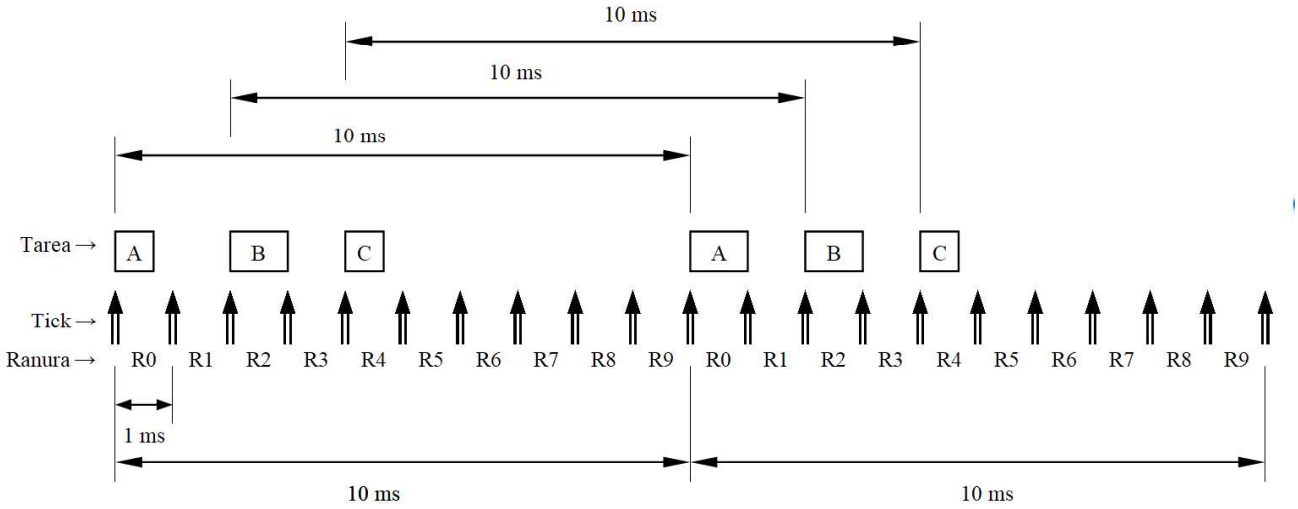

Fig. 3. Distribución de 3 tareas en una máquina de tiempo de 10 ranuras.

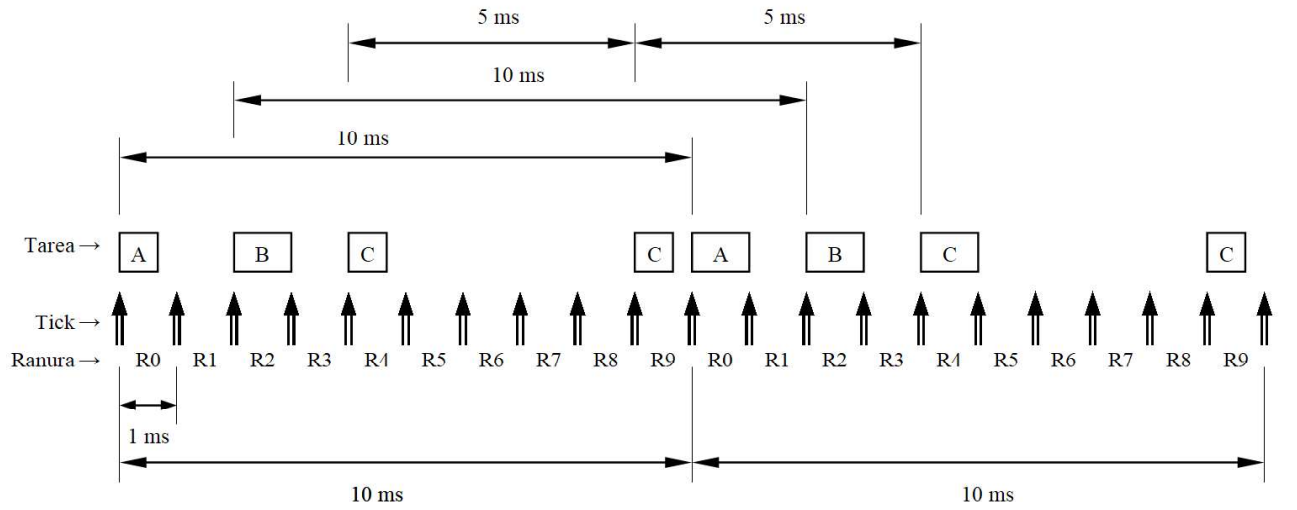

Fig. 4. Distribución de 3 tareas en una máquina de tiempo de 10 ranuras con la tarea C a distinto período.

de tiempo determinado y constante entre las *Tareas B* y entre las *Tareas C*, tal y como se muestra en fig. 2.

III. Máquinas de tiempo

La máquina de tiempo es una estructura de programación que permite la administración de tareas que deben ser ejecutadas en intervalos de tiempo precisos. Se utiliza cuando es necesario efectuar varias operaciones periódicas con cierto grado de precisión en el tiempo.

En el concepto de máquina de tiempo, se utilizan *ticks* de mayor frecuencia y en ellos se distribuyen las tareas. Siguiendo con el ejemplo, el período de repetición de 10 ms se trocea en 10 intervalos de 1 ms cada uno. A estos intervalos los denominaremos "*ranuras de tiempo*". Esto se traduce en hacer que las interrupciones de tiempo sean cada 1 ms y atender esas interrupciones por un administrador: la máquina de tiempo. De esta manera cada una de las tareas se puede realizar en una ranura de tiempo específica, tal y como se muestra en la fig. 3. En la misma figura se observa que ahora sí se cumplen los requisitos temporales entre las tareas.

Para este ejemplo se ha decidido dividir el período de repetición en 10 ranuras, y por tanto pudieran ubicarse hasta 10 tareas en las distintas. Obviamente, mientras mayor sea la densidad de ranuras por período de repetición, mayor será la carga sobre el procesador y se restará capacidad de procesamiento al lazo principal.

En esta estructura de ranuras del ejemplo, pudiera existir una tarea que se deseara ejecutar con una mayor periodicidad. Supongamos, por ejemplo, que la *Tarea C* se quiera ejecutar cada 5 ms. Para ello se asignará dicha tarea en dos de las 10 ranuras como se observa en la fig. 4.

La construcción de una máquina de tiempo por programa es muy parecida a una máquina de estados y se ejecuta como rutina de atención en la interrupción del temporizador fig. 5. El diagrama de flujo de la máquina de tiempo para el ejemplo que se viene utilizando, se muestra en la fig.

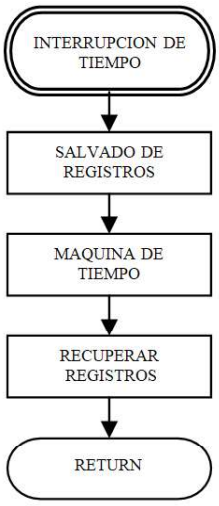

Fig. 5. La máquina de tiempo se ubica en la rutina de atención de interrupciones del temporizador.

6. Obsérvese la semejanza con una *máquina de estados*, la diferencia fundamental está en que el concepto de "estado" se transfigura en el concepto de "ranura de tiempo" y que el número de la ranura de tiempo se actualiza dentro de la misma máquina de tiempo.

La codificación de las máquinas de tiempo varía de un procesador a otro o de un lenguaje de programación a otro. Sin embargo, la implementación puede hacerse con gran eficiencia en la mayoría de los procesadores. Para mantener una perspectiva electrónica en las aulas, nosotros utilizamos el lenguaje ensamblador de los procesadores PIC16 y la codificación de la máquina de tiempo en ensamblador sobre estos procesadores se muestra en la fig. 7.

Este mismo ejemplo nos sirve para hacer un cálculo aproximado de la carga que tiene la máquina de tiempo sobre el procesador. Suponiendo un PIC16 con un ciclo de máquina de 1 us, la máquina de tiempo se ejecuta en aproximadamente 10 us. El salvado y recuperación de registros por interrupción supone, en conjunto, unos 30 us. Esto significa que con una interrupción de 1 ms, la máquina de tiempo representa una carga de un 4%.

IV. Control de motores por PWM

Para seguir ilustrando la aplicación de las máquinas de tiempo, vamos a suponer que se desea controlar independientemente la velocidad de cuatro motores DC con un procesador PIC16F887 (Fig. 8). Este procesador dispone de dos periféricos específicos para generar hasta 2 señales PWM con una resolución de 10 bits, por lo que no podríamos gobernar los cuatro motores independientemente. El uso de la máquina de tiempo permite controlar varios motores mediante PWM utilizando salidas digitales convencionales (GPIO).

En nuestro caso, por simplicidad, vamos a suponer que se controlan los motores a 4 niveles de velocidad, correspondientes a una modulación PWM al 0%, al 33.3%, al 66.7% y al 100%. El período de la señal de control a utilizar será de 1.5 ms (667 Hz).

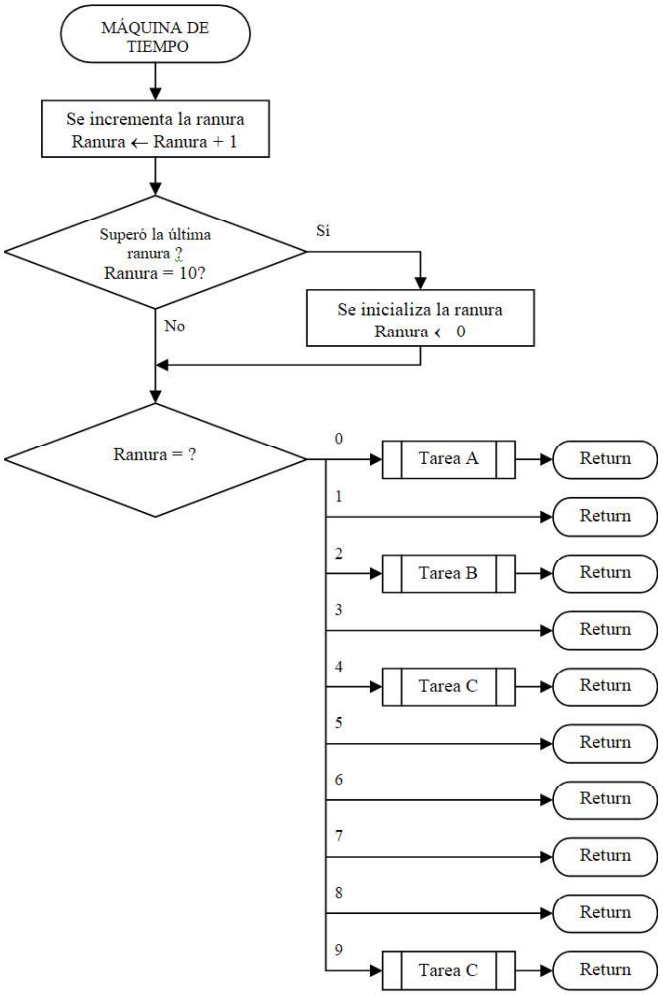

Fig. 6. Diagrama de flujo de una máquina de tiempo.

El número mínimo de ranuras que se requieren para controlar los motores viene dado por la ecuación:

N_Ranuras = N_Motores x (N_Niveles - 1)

Dado que para este ejemplo se tienen 4 motores y 4 niveles de modulación, la máquina de tiempo tiene que tener un mínimo de 12 ranuras que serán barridas en un período de repetición de 1.5 ms. En la fig. 9 se pueden observar las señales para los cuatro motores, suponiendo diferentes modulaciones en cada uno de ellos. Obsérvese cómo existe un desfase en el inicio de las señales debido al posicionamiento de las tareas en las ranuras.

Dado que las 12 ranuras deben ser distribuidas en un período de 1,5 ms, las ranuras deben ser ubicadas cada 125 us, es decir, la interrupción del temporizador que llama a la máquina de tiempo deberá ser de 125 us.

La rutina para fijar la velocidad de cada motor (MOTOR_n), mostrada en la fig. 10, se utilizará en el programa principal, fuera de la interrupción. Con esa rutina se asigna la velocidad (0, 1, 2 o 3) a la variable correspondiente.

Las tareas que deben instalarse en la máquina de tiempo

```
;*****************************************************************
; Máquina de tiempo

MAQTIE:
        INCF    Ranura,F    ; Se actualiza ranura de tiempo
        MOVLW   d'10'       ; Se verifica que no haya superado el
        XORWF   Ranura,W    ;  valor máximo de ranuras
        BTFSC   STATUS,Z    ;
        CLRF    Ranura      ;  en caso contrario se inicializa
        MOVF    Ranura,W    ; Se toma el número de ranura
        ADDWF   PCL,F       ; Se salta a la ranura correspondiente
        GOTO    TAREA_A     ; Ranura 0: Tarea A
        RETURN              ; Ranura 1: Ranura vacia
        GOTO    TAREA_B     ; Ranura 2: Tarea B
        RETURN              ; Ranura 3: Ranura vacia
        GOTO    TAREA_C     ; Ranura 4: Tarea C
        RETURN              ; Ranura 5: Ranura vacia
        RETURN              ; Ranura 6: Ranura vacia
        RETURN              ; Ranura 7: Ranura vacia
        RETURN              ; Ranura 8: Ranura vacia
        GOTO    TAREA_C     ; Ranura 9: Tarea C

TAREA_A:                    ; Tarea A
              ; Programa de tarea A
        RETURN

TAREA_B:                    ; Tarea B
              ; Programa de tarea B
        RETURN

TAREA_C:                    ; Tarea C
              ; Programa de tarea C
        RETURN
```

Fig. 7. Codificación de una máquina de tiempo para procesadores PIC16.

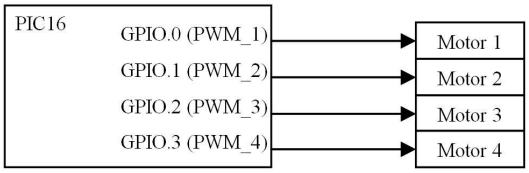

Fig. 8. Procesador PIC16 con los 4 motores a controlar.

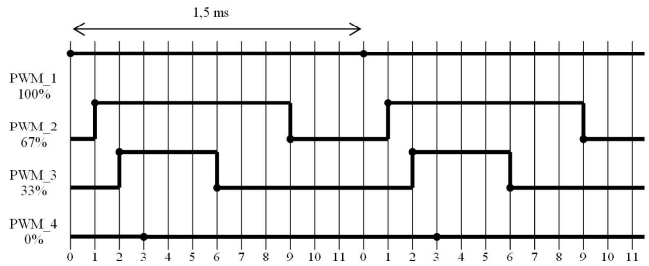

Fig. 9. Dos períodos de las señales de control de los motores (La actuación de la máquina de tiempos se simboliza con un punto).

(GENMI_n y GENM_n) son de la forma presentada en los diagramas de flujo de la fig. 11. La primera tarea (GENMI_n) reinicia el valor del contador de ancho de pulso para cada período de repetición. El diagrama de la máquina de tiempo se visualiza en la fig. 12.

Suponiendo nuevamente un PIC16 con un ciclo de máquina de 1.0 us, la máquina de tiempo con las respectivas tareas de control de motores representa una carga de aproximadamente un 50%. Esta carga de trabajo depende de la complejidad de las tareas ubicadas en la máquina de tiempo, aunque como se observa son tareas relativamente simples. Por ello, la variable

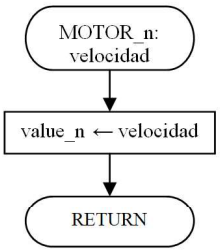

Fig. 10. Rutina de fijación de velocidad para el motor "n".

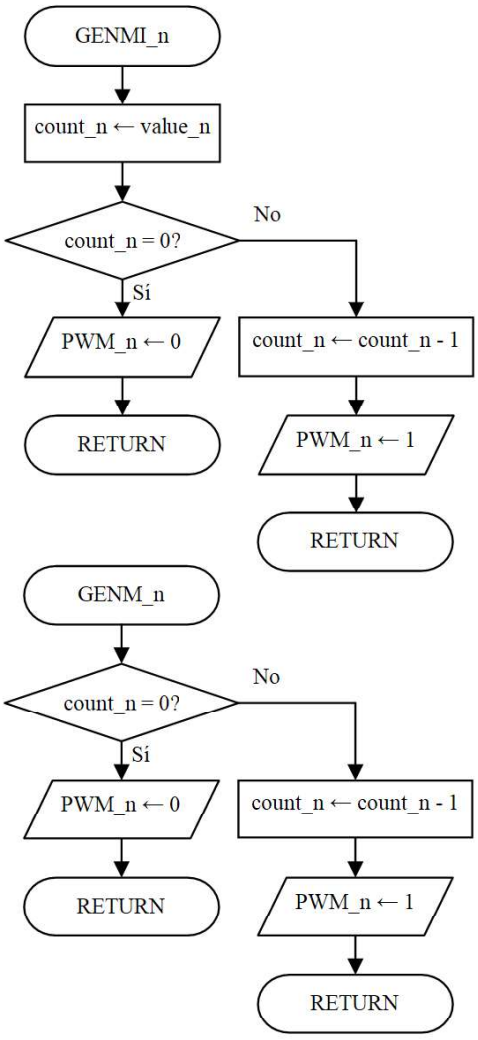

Fig. 11. Diagrama de flujo de las tareas que deben ser llamadas desde la máquina de tiempo de control de los motores.

que influye más significativamente en la carga de trabajo es la duración del período de cada ranura, en este caso, 125 us.

V. Control de servos por PWM

Otra aplicación muy idónea y popular para la máquina de tiempo es el control de servomotores o servos [13]. Estos dispositivos son muy utilizados en robots y su control se lleva

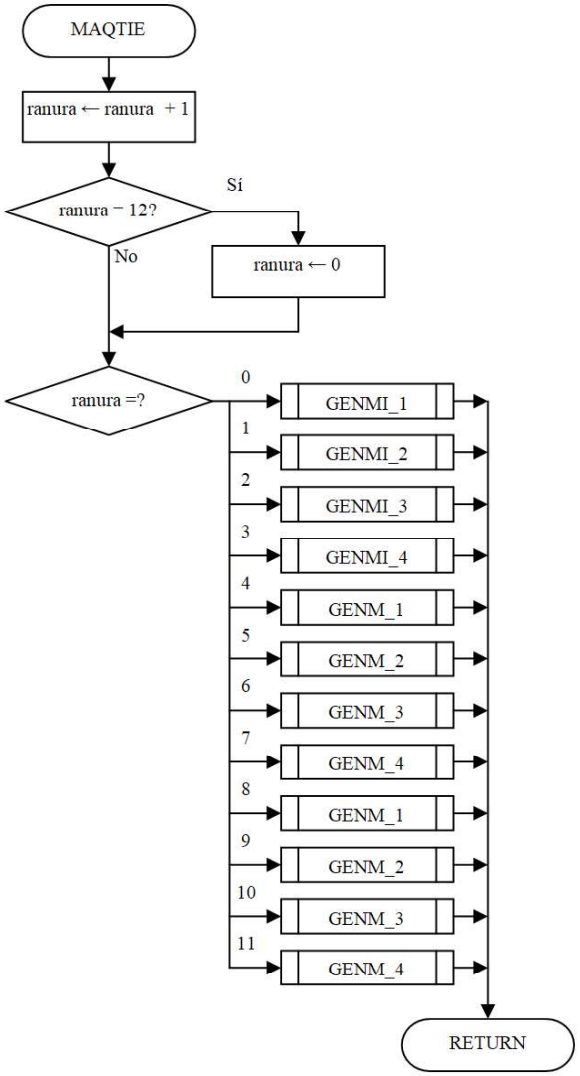

Fig. 12. Diagrama de flujo de la máquina de tiempo de control de motores.

a cabo con una señal PWM. Un coche-robot puede requerir un par de servos, un avión radio-controlado unos 5 o 6, y un robot humanoide unos 8. Por lo que un microcontrolador como el PIC16F887 comentado anteriormente, pudiera resultar descartado si se recurre a sus dos periféricos PWM. El uso de la máquina de tiempo en el mismo microcontrolador puede controlar sin problemas más de 8 servos.

Estos servomotores son controlados por una señal PWM con un período de 20 ms. El ancho del pulso puede variar desde 1 ms (brazo a 0º) hasta 2 ms (brazo a 180º) tal y como se representa en la fig. 13. Señales de estas características no pueden ser generadas por los periféricos específicos para PWM en los procesadores PIC16F887. Por ello, la única solución es el uso de la máquina de tiempo.

En este caso, el número de ranuras requeridas debe calcularse de manera diferente porque la señal a generar tiene requisitos temporales muy específicos. Es necesario partir de la resolución requerida en el movimiento de los servomotores. Supongamos que se desea una resolución de 18º en el giro

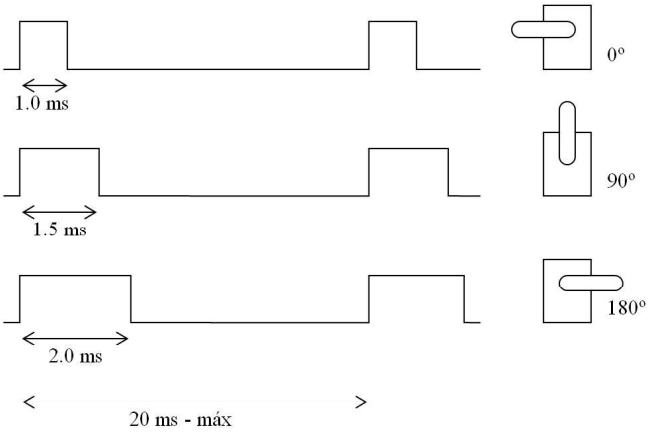

Fig. 13. Señal de control de servomotores.

de los mismos. El tiempo de cada ranura para obtener tal resolución es:

T_Ranura =

(Tmax_Pulso - Tmin_Pulso) x Resolución / 180º

T_Ranura = (2ms - 1ms) x 18º / 180º = 100us

El número total de ranuras es ahora dependiente del período total de la señal y del tiempo de ranura:

N_Ranuras = T_Total / T_Ranura

N_Ranuras = 20ms / 100us = 200 ranuras

A diferencia del caso anterior del control de motores, para los servos es imperativo que cada 20 ms se presente un pulso de por lo menos 1 ms. Por ello, la tarea de inicio del pulso GENSI_n es distinta (fig. 14).

Hay diferencias también en la distribución de las tareas en la máquina de tiempo (Fig. 15). Como el ancho mínimo del pulso es de 1.0 ms, una vez inicializado el pulso no debe ser controlado hasta la ranura 9. Además como el máximo ancho de pulso debe ser de 2.0 ms, las ranuras en que debe utilizarse la rutina GENS_n son las comprendidas entre la 9 y la 19.

Como se puede observar en la fig. 15, se encuentran libres las ranuras 1 a 8 y las ranuras 20 a 199. Éstas pueden ser utilizadas para controlar otros servos adicionales. En total, con esta máquina de tiempo es posible controlar 10 servomotores con una carga de trabajo del orden de 50% para un procesador PIC16 con un ciclo de máquina de 1 us.

VI. Control de tareas mixtas

En los ejemplos anteriores se ha utilizado la máquina de tiempos para controlar un único tipo de dispositivo: motores o servos. Sin embargo, en una misma máquina de tiempo pueden convivir tareas de diferentes dispositivos. Por ejemplo: motores o luces controladas por PWM, servomotores, reloj de tiempo real, etc.

Supongamos que deseamos controlar con una máquina de tiempo dos servomotores y dos motores. Para simplificar el ejemplo se controlarán los motores a 3 niveles de velocidad

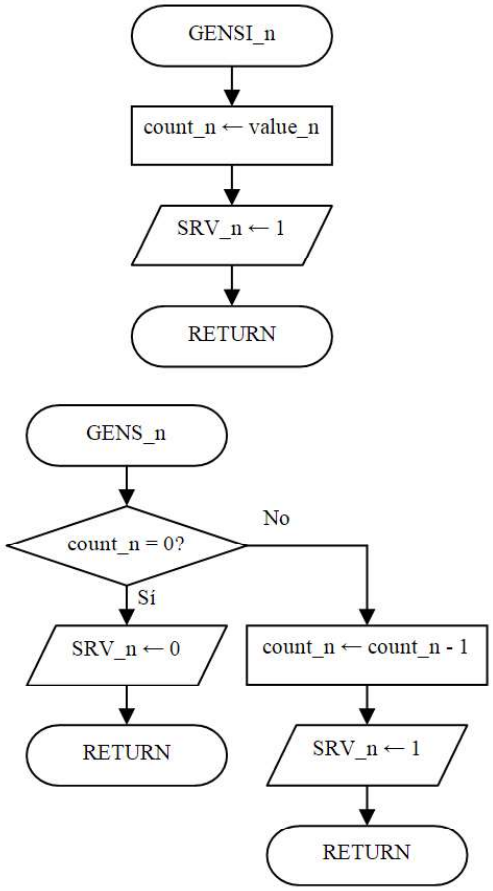

Fig. 14. Diagrama de flujo de las tareas que deben ser llamadas desde la máquina de tiempo de control de los servomotores.

(0%, 50% y 100%), es decir, se requiere el uso de 2 ranuras por período para cada motor.

La señal de período más largo es la correspondiente a los servomotores, y es la que marca el período total de la máquina de tiempo, es decir 20 ms. Esto significa que si sólo se posicionan 2 ranuras en todo el período, la frecuencia de control de los motores será de 50 Hz (20 ms). Esta frecuencia puede resultar muy baja para un control adecuado. Por ello, y dado que disponemos de muchas ranuras libres, se distribuirán uniformemente las dos ranuras de cada motor por todo el conjunto de ranuras. En este caso es posible asignar cinco pares de ranuras para cada motor por lo que la frecuencia de control aumenta a unos, más razonables, 250 Hz (5 x 50 Hz).

En la Tabla 1 se observa la distribución de tareas en la máquina de tiempo para el ejemplo mixto propuesto. De esta tabla, se deduce que aún queda espacio para el control de varios dispositivos sin afectar sus funciones. Por ejemplo, pudiera utilizarse una ranura para la actualización del reloj de tiempo real del sistema cada 20 ms.

VII. Conclusiones

La máquina de tiempo es una estructura de programación muy sencilla que permite hacer un uso muy eficiente de las posibilidades de pequeños procesadores que operan a velocidades respetables aún en sus versiones más económicas.

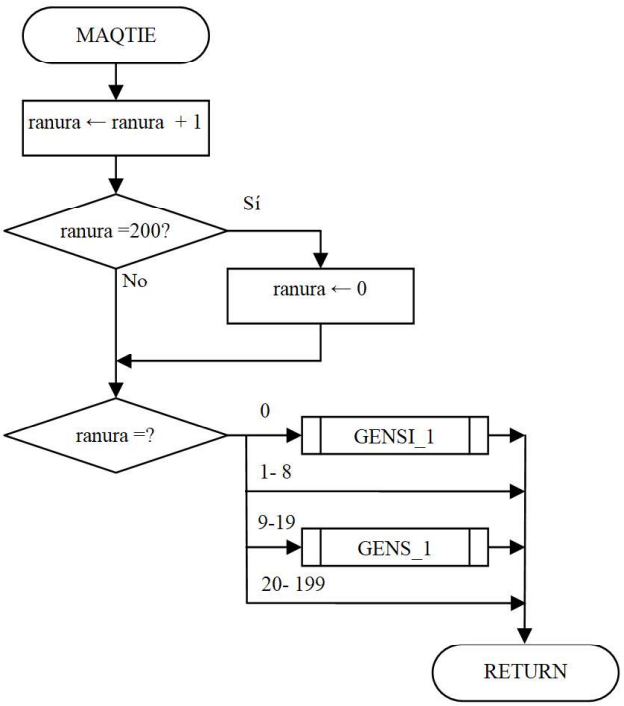

Fig. 15. Diagrama de flujo de la máquina de tiempo de control de un servomotor.

TABLA I
ISTRIBUCIÓN DE TAREAS EN LA MÁQUINA DE TIEMPO PARA CONTROLAR 2 SERVOMOTORES Y 2 MOTORES.

Ranura	Tarea
0	Servo GENSI_1
1	Motor GENMI_1
2	Motor GENMI_2
9-19	Servo GENS_1
20	Servo GENSI_2
21	Motor GENM_1
22	Motor GENM_2
29-39	Servo GENS_2
41	Motor GENMI_1
42	Motor GENMI_2
61	Motor GENM_1
62	Motor GENM_2
81	Motor GENMI_1
82	Motor GENMI_2
101	Motor GENM_1
102	Motor GENM_2
121	Motor GENMI_1
122	Motor GENMI_2
141	Motor GENM_1
142	Motor GENM_2
161	Motor GENMI_1
162	Motor GENMI_2
181	Motor GENM_1
182	Motor GENM_2

Como se ha podido ver a través de los ejemplos expuestos en el presente trabajo, el uso de la máquina de tiempo permite extender la capacidad de control de los procesadores por encima de lo que se pudiera prever de sus especificaciones comerciales.

Pero quizás, la ventaja más importante desde el punto de vista docente, radica en ayudar a formar nuevos conceptos de programación que servirán de base a cursos avanzados de programación. Todo esto sin perder la perspectiva de la máquina electrónica que es un microcontrolador.

Por otro lado, para el profesor que dirige el curso, el uso de estructuras como la máquina de estados y eventos y la máquina de tiempo, permite una evaluación más concreta y objetiva del progreso de los estudiantes. Además, el conocimiento de estructuras profesionales claras, permite que los estudiantes puedan aprender a trabajar en ambientes colaborativos donde la forma de trabajo está muy reglada en el aspecto de la integración de bloques de programa. Aspecto éste que es muy remarcado con los actuales programas de estudio formulados en base a competencias [14].

AGRADECIMIENTOS

El trabajo descrito en esta publicación ha sido generado y patrocinado por el Departamento de Educación, Política Lingüística y Cultura del Gobierno Vasco en base a las ayudas para apoyar las actividades de grupos de investigación del sistema universitario vasco IT978-16 y por el Ministerio de Economía y Competitividad a través del proyecto de investigación TEC2017-84011 y los fondos FEDER.

REFERENCIAS

[1] BOPV. RESOLUCIÓN de 20 de diciembre de 2010, de la Vicerrectora de Ordenación Académica, de la Universidad del País Vasco/Euskal Herriko Unibertsitatea, por la que se procede a la publicación del plan de estudios del Grado en Ingeniería Técnica de Telecomunicación de la Universidad del País Vasco/Euskal Herriko Unibertsitatea. Boletín Oficial del País Vasco, nº 44 de 4 de marzo de 2011.

[2] "Planificador" Wikipedia, 2018. [En línea] Disponible en: https://es.wikipedia.org/wiki/Planificador [Accedido: 26/04/2018].

[3] R. Tocci, N. Widmer, *Sistemas digitales: principios y aplicaciones.* Mexico: Prentice Hall, 2003.

[4] "Scheduling (computing)" Wikipedia, 2018. [En línea] Disponible en: https://en.wikipedia.org/wiki/Scheduling_(computing) [Accedido: 26/04/2018].

[5] O. González, "Sistemas operativos en microcontroladores. Seminario de Microrrobots" Universidad de Alcalá, 2006. [En línea] Disponible en: http://www.alcabot.com/alcabot/seminario2006/SEM06_SOenMicrocontroladores.pdf [Accedido: 26/04/2018].

[6] A. Zuloaga, A. Astarloa, *Sistemas de procesamiento digital*, Madrid: Delta Publicaciones, 2008.

[7] J. Jiménez, C. Cuadrado, I. Kortabarria, J. Andreu, A. Zuloaga, "Dificultades al aprender a especificar máquinas de estados en sistemas microprogramados," , pp. 428-432, XII Congreso de Tecnologías, Aprendizaje y Enseñanza de la Electrónica (TAEE 2016), Sevilla (España), Junio 22-24, 2016.

[8] A. Zuloaga, *Laboratorio de Sistemas Digitales*, Bilbao: Publicaciones de la Escuela de Ingeniería de Bilbao, 2013

[9] A. Zuloaga, A. Astarloa, *Sistema Digital PICTOR*, Bilbao: Servicio de publicaciones de la Escuela de Ingeniería de Bilbao, 2013

[10] A. Zuloaga, J. Jiménez, I. Kortabarria, J. Andreu, " PICTOR: circuito impreso para montar un sistema digital básico," , pp. 315- 319, XI Congreso de Tecnologías, Aprendizaje y Enseñanza de la Electrónica (TAEE 2014), Sevilla (España), Junio 11-13, 2014.

[11] E. García, *Compilador C CCS y Simulador Proteus para Microcontroladores PIC*, México: Alfaomega grupo editor, 2008.

[12] T. Wilmshurst, *Designing Embedded Systems with PIC Microcontrollers.* Londres: Elsevier. 2007

[13] "Servomotor ¿Que es y como funciona?", Ingeniería mecafenix, 2018. [En línea] Disponible en: http://www.ingmecafenix.com/electricidad-industrial/servomotor/ [Accedido: 04/02/2018].

[14] BOE. Orden CIN/352/2009, de 9 de febrero, por la que se establecen los requisitos para la verificación de los títulos universitarios oficiales que habiliten para el ejercicio de la profesión de Ingeniero Técnico de Telecomunicación, 20 de febrero de 2009.