8º Congreso Nacional de Mecatrónica
Noviembre 26 y 27, 2009. Veracruz, Veracruz.

Tarjeta Controladora de Motores de Pasos Unipolares Basada en un Modelo de Autómata Finito Determinista Implementado en Microcontroladores

Alberto Ocotitla Hernández, Tania Jaramillo Torres

Unidad Profesional Interdisciplinaria de Ingeniería y Ciencias Sociales y Administrativas
Instituto Politécnico Nacional
albertoo@cromodesign.com.mx tjaramillot0400@ipn.mx

Resumen

En el presente artículo se propone la sinergia de las ciencias computacionales con el diseño electrónico para imitar las funcionalidades de circuitos integrados especializados y de difícil adquisición en algunos casos, permitiéndonos adaptar sus bondades a nuestras necesidades específicas, en concreto se aborda el diseño de una tarjeta controladora de motores de pasos unipolares, cuya principal cualidad es la unificación de las etapas de interface y control en un solo circuito integrado, lo que permite reducir el costo y tamaño de la tarjeta, pues no se precisa un controlador comercial como el L297 para cada motor debido a que la interface con la computadora y la lógica de activación de las bobinas se realiza mediante un Microcontrolador cuyo programa está basado en un modelo de Autómata Finito Determinista, dicho modelo permite secuencias de movimiento en Full Step y Half Step, sin embargo puede modificarse fácilmente para la secuencia de movimiento Wave Drive e incluso para motores de pasos bipolares. El modelo es independiente del hardware y puede ajustarse a los requerimientos de cada aplicación específica, en el caso particular el modelo se ha implementado en un PIC16F88 con una etapa de potencia de bajo costo que brinda la posibilidad de controlar N motores desde un solo puerto RS232.

Palabras clave: Motor de Pasos, Autómata Finito Determinista, Microcontrolador

1. Introducción

En la actualidad es frecuente encontrar aplicaciones que integren 2 disciplinas que a primera vista pueden parecer distantes, tal es el caso de las ciencias computacionales y el diseño electrónico, sin embargo el resultado de la sinergia de estas disciplinas deriva en diversas aplicaciones interesantes de entre las cuales destaca el diseño de circuitos integrados altamente especializados, por ejemplo las redes neuronales utilizadas en el reconocimiento de rostros y procesamiento de imágenes en cámaras digitales [1] y los algoritmos de criptografía implementados en Arreglos de Compuertas Programables en Campo [2] ó FPGAs por sus siglas en inglés.

En este contexto la propuesta consiste en utilizar un modelo de cómputo fácil de implementar en Dispositivos de Lógica Programable y Microcontroladores, que nos permita encapsular el mayor número de etapas en un solo circuito integrado con el fin de no requerir controladores comerciales para la construcción una tarjeta controladora de N motores de pasos cuyo diseño sea robusto, de reducidas dimensiones y de bajo costo.

A continuación se hará una breve revisión de los motores de pasos, su constitución interna y formas de control, posteriormente se abordará el diseño del modelo computacional y su implementación en un microcontrolador, finalmente se mostrarán los resultados obtenidos y las conclusiones del proyecto.

2. Motores de Pasos

Los motores de pasos son actuadores eléctricos utilizados en dispositivos que requieren precisión en el movimiento, generalmente se les puede encontrar en maquinas herramienta, fotocopiadoras, impresoras y robots entre otras aplicaciones, su principal cualidad es que a diferencia de lo que ocurre con los motores DC, el eje no gira libremente ni de forma continua cuando son energizados, por el contrario su eje puede realizar desplazamientos angulares discretos llamados “pasos” y permanece enclavado en su posición ofreciendo una ligera resistencia al cambio, estos motores pueden girar únicamente un paso por vez y

lo hacen mediante la correcta combinación en el energizado de sus bobinas, tal combinación está relacionada a la forma en que está constituida la estructura interna del motor, siendo los motores de pasos de imán permanente los más utilizados y por ende los que analizaremos a continuación.

Existen dos tipos de motores a pasos de imán permanente, los unipolares y los bipolares, ambos requieren de diferentes combinaciones para efectuar el movimiento de su eje, por sencillez este trabajo está enfocado a los motores unipolares, sin embargo como se mostrará a lo largo del artículo, es factible aplicar el modelo en motores bipolares con ligeros cambios a nivel de código y en la etapa de potencia, en la figura 1 muestra la estructura interna de un motor de pasos unipolar.

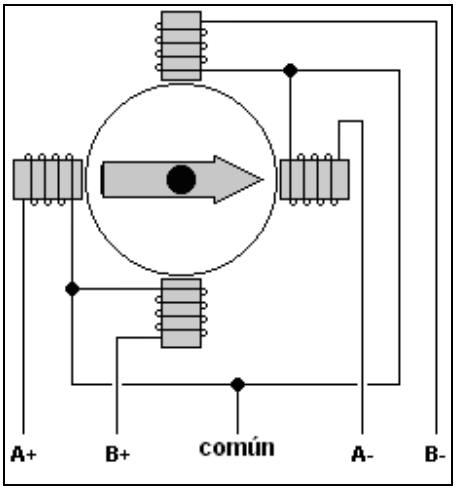

Fig. 1. Estructura interna de un motor a pasos unipolar

El motor a pasos unipolar está constituido por un imán permanente en su rotor y una serie de embobinados en su estator, generalmente tienen 5 o 6 cables, un cable para cada bobina y 1 o 2 cables comunes; como su nombre lo indica sus bobinas solo requieren de una polaridad, permitiendo utilizar una etapa de potencia más sencilla y barata en comparación a la que requieren los motores bipolares, de esta forma podemos alimentar con masa el cable común y aplicar una diferencia de potencial en las bobinas o viceversa.

Existen 4 secuencias de movimiento para los motores unipolares, las 3 más conocidas son: ***Full Step***, ***Half Step*** y ***Wave Drive***, la primera secuencia energiza 2 bobinas por paso logrando el máximo torque aunque genera cierta vibraciones y ruido en algunos casos, la Wave Drive energiza solo una bobina por paso con lo que se logra un movimiento más suave pero de menor fuerza, de la combinación de ambas secuencias surge la ***Half Step*** que energiza 1 bobina después de haber energizado 2, esto hace que la secuencia completa se efectúe en 8 pasos en vez de 4 con lo que se logra mayor resolución como se puede apreciar en la figura 2.

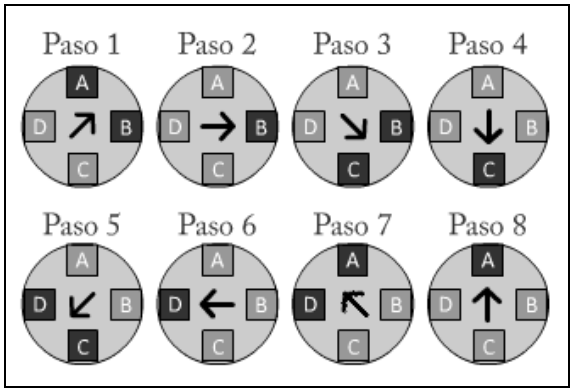

Fig. 2. Secuencia de movimiento Half Step

Existe una cuarta secuencia de movimiento, llamada Micropasos [3] que está fuera del alcance de este trabajo debido a que su naturaleza de control no es digital sino analógica, a grandes rasgos hace uso del control de la corriente aplicada a cada bobina para generar multitud de pasos intermedios entre embobinados.

Como se puede apreciar generar la combinación adecuada en los cables del motor puede resultar tedioso cuando se trabaja con varios motores controlados por computadora, por ello cualquier aplicación que requiera de un control fino sobre el número de pasos hace necesaria la consideración de un controlador comercial como el L297 [4], este circuito integrado facilita enormemente el control pues permite indicar el número de pasos que debe ejecutar el motor mediante pulsos TTL, además de contar con otras terminales para indicar mediante bits el sentido de giro y la cantidad de bobinas energizadas por paso, las señales de control se entregan en paralelo por lo que se requieren al menos 3 líneas para controlar el sentido de giro, la velocidad y la secuencia de movimiento del motor.

3. Desarrollo del Modelo

Bajo el enfoque tradicional de control de motores de pasos vía puerto serie y utilizando el driver L297 se requiere de una interfaz para acoplar los voltajes RS232 con los TTL del Microcontrolador, generalmente se utiliza un

convertidor como el MAX232 [5], posteriormente el Microcontrolador convierte los datos serie a señales en paralelo que se entregan al driver L297, precisando un controlador por cada motor.

El esquema propuesto consiste en integrar las etapas vecinas en un solo circuito integrado programable, ambos esquemas se muestran en la figura 3.

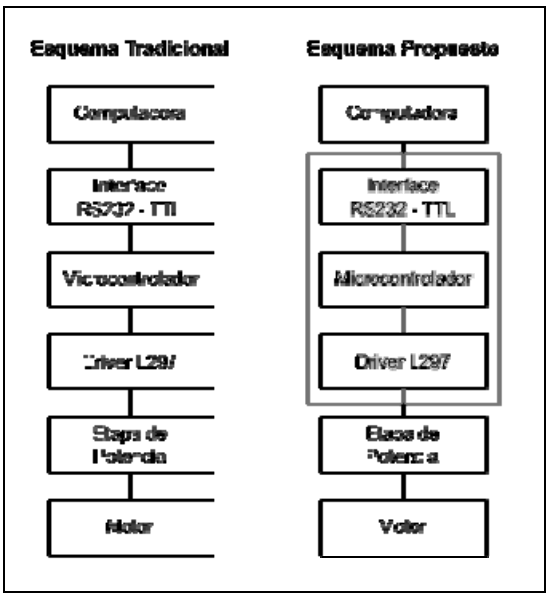

Fig. 4. Esquema tradicional vs esquema propuesto

El primer objetivo que se plantea consiste en utilizar un modelo computacional discreto que permita disponer de las bondades del circuito L297, esto es un control fino del número de pasos, la posibilidad de indicar la secuencia de movimiento y el sentido de giro del motor, dicho modelo debe ser fácil de implementar en un dispositivo programable de reducidas dimensiones y menor costo en comparación al L297.

El modelado de sistemas discretos permite verificar hipótesis, efectuar predicciones y hacer simulaciones sin tener que actuar sobre el sistema real, en particular los autómatas finitos [6] son de gran utilidad para modelar los fenómenos en los cuales no se conoce por completo la entrada del sistema, por el contrario, la entrada llega por partes, sin intervalos de tiempo preestablecidos, ninguna secuencia programada y cada estado "recuerda" el estado anterior.

El autómata finito determinista tiene 5 elementos que definen su comportamiento: estado inicial ***s***, alfabeto de entrada $\sum$, conjunto de estados ***S***, estados finales ***F*** y función de transición **δ**.

El primer paso en el modelado del autómata consiste en identificar todos los posibles estados del sistema, en nuestro caso son 8, que corresponden a los 8 pasos que ejecuta el motor en la secuencia ***Half Step***, se ha elegido esta secuencia de movimiento pues incluye dentro de si a la secuencia ***Full Step***, por lo tanto ***S = {S1,S2,S3,S4,S5,S6,S7,S8)***

El alfabeto de entrada del autómata está dado por las entradas del sistema, en nuestro caso las entradas serán 4 comandos los cuales se muestran en la tabla 1.

Comando	Acción sobre el sistema
R	Ejecuta un paso a la derecha en secuencia Full Step
L	Ejecuta un paso a la izquierda en secuencia Full Step
r	Ejecuta un paso a la derecha en secuencia Half Step
l	Ejecuta un paso a la izquierda en secuencia Half Step

Tabla 1. Comandos del controlador

De la tabla 1 determinamos que $\sum$ = ***{R,L,r,l}*** que son el conjunto de caracteres o alfabeto que reconoce el autómata, estos caracteres fueron elegidos tomando como base las palabras *Right* y *Left* del idioma inglés, también se ha simbolizado con mayúsculas a los comandos que se refieren a la secuencia de mayor torque (2 bobinas) en el motor.

Finalmente la función de transición se obtiene al relacionar cada estado con las posibles entradas del sistema, la función se muestra en la figura 5, esto también se puede representar con un diagrama de estados el cual se muestra en la figura 6.

```
δ = {((S1,R),S3),((S1,r),S2),((S1,L),S7),((S1,l),S8)
     ((S2,R),S3),((S2,r),S3),((S2,L),S1),((S2,l),S1)
     ((S3,R),S5),((S3,r),S4),((S3,L),S1),((S3,l),S2)
     ((S4,R),S5),((S4,r),S4),((S4,L),S3),((S4,l),S3)
     ((S5,R),S7),((S5,r),S6),((S5,L),S3),((S5,l),S4)
     ((S6,R),S7),((S6,r),S7),((S6,L),S5),((S6,l),S5)
     ((S7,R),S1),((S7,r),S8),((S7,L),S5),((S7,l),S6)
     ((S8,R),S1),((S8,r),S1),((S8,L),S7),((S8,l),S7)}
```

Fig. 5. Función de transición del autómata

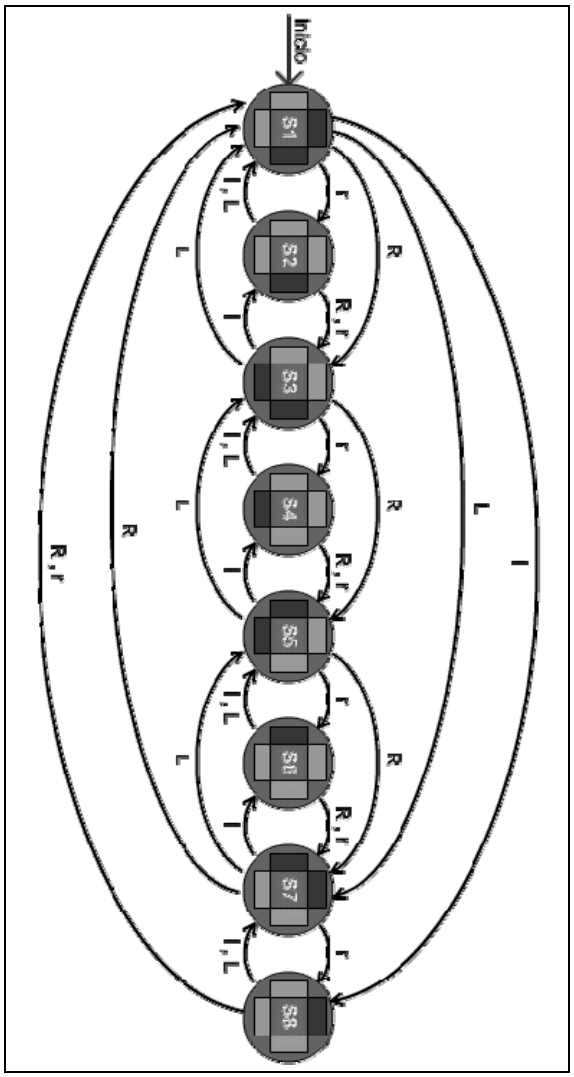

Fig. 6. Diagrama de estados del autómata

Cabe destacar que todos los estados del autómata son estados finales o de aceptación, estos suelen representarse con un doble círculo, sin embargo se ha omitido para simplificar el diagrama y que sea más legible.

Este modelo sirve para controlar un motor de pasos desde una terminal capaz de enviar caracteres en serie, sin embargo para controlar *N* motores utilizando la misma terminal se debe incluir un identificador antes de cada comando, es decir el controlador en vez de recibir un solo caracter (el comando) recibe 2 caracteres (ID y Comando), por ejemplo; la cadena ***XR,XR,XR,XL*** le indica al controlador del motor ***X*** que debe dar 3 pasos a la derecha y después 1 a la izquierda, todos en modo *Full Step*, si por el contrario una terminal serie le envía al controlador la cadena ***Yr,Yr,Yr,Xr,Xr*** le estaría indicando al controlador del motor ***Y*** que debe ejecutar 3 pasos a la derecha en modo *Half Step* y después el controlador del motor ***X*** debe ejecutar 2 pasos a la derecha en modo *Half Step*.

La velocidad del giro del motor estará en función de la velocidad con la que lleguen los caracteres desde la terminal y las limitaciones mecánicas propias del motor.

4. Implementación Física del Modelo

Al momento de elegir el dispositivo se consideraron PLDs y Microcontroladores, se optó por la segunda opción debido a que muchos modelos incluyen oscilador interno y módulos USART por lo que la implementación sería rápida y muy sencilla, en el caso particular se ha tenido experiencia previa con el microcontrolador PIC16F88 por ello se decidió utilizarlo, este integrado de 18 pines, cuenta con oscilador interno de 8 MHz, 2 puertos de E/S, un ADC de 8 canales y 10 bits de resolución así como una unidad USART, en la figura 7 se puede ver en detalle la distribución de las terminales del integrado.

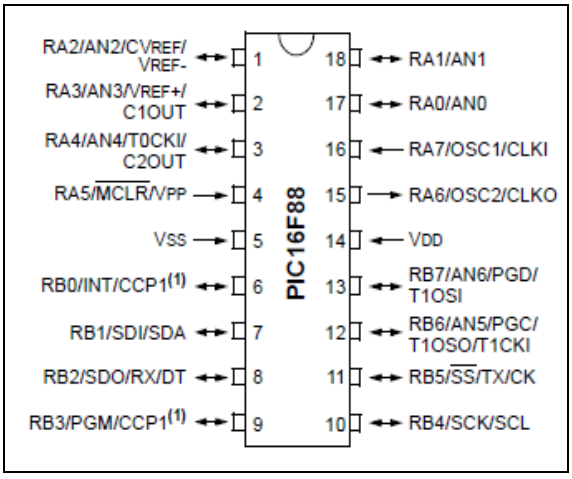

Fig. 8. Distribución de los pines del PIC16F88

Debido a sus 2 puertos de 8 bits se decidió implementar 2 autómatas, uno con el ID *X* y otro con *Y*, cada uno con salida a los puertos B y A respectivamente, a nivel de código solo debió de tenerse precaución en desactivar los periféricos analógicos, tales como el ADC mediante el registro ANSEL que debe igualarse a 0 para trabajar en modo digital, otro registro importante es el OSCCON que determina la velocidad del reloj interno, por lo que dicho registro tuvo que igualarse a 126 para trabajar a una frecuencia de 8 MHz.

Antes de llevar a cabo la construcción se elaboró un prototipo virtual, en la figura 8 se muestra la simulación del circuito en Proteus ISIS, se trata de un diagrama simplificado en el que se ha utilizado una barra de LEDs para verificar el correcto funcionamiento, la terminal virtual simula cualquier terminal serie, en el caso concreto se estableció una velocidad de 2400 baudios, esto debido a que algunos módulos de RF trabajan a baja velocidad, de esta forma se asegura la compatibilidad con la PC y con módulos de RF.

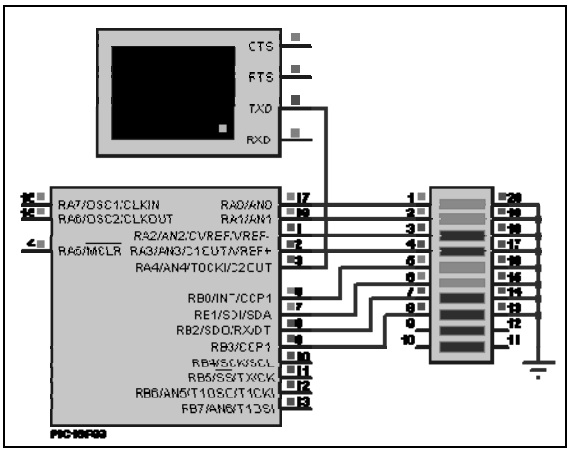

Fig. 8. Simulación del circuito en Proteus ISIS

Como se puede apreciar en la imagen se han utilizado los 4 bits menos significativos de cada puerto, los autómatas inician en el estado 1 y por ello ambos tienen en nivel alto sus 2 bits menos significativos que una vez acoplados a la etapa de potencia activarían las bobinas A y B de cada motor, una vez comprobado el correcto funcionamiento de los autómatas, se implementó físicamente todo el sistema, la interface RS232, la cual gracias a las características eléctricas del microcontrolador se puede implementar directamente con una resistencia de 22k conectada al pin 3 (TX) del conector DB9 y al pin 18 (PORTA.0) del microcontrolador, estas conexiones se muestran en la figura 9, a nivel de código ha de manejarse una lógica invertida pues los voltajes RS232 para los "1" y "0" lógicos están invertidos con respecto a los TTL.

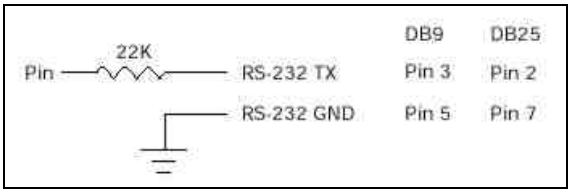

Fig. 9. Conexiones mínimas para interface RS232

La etapa de potencia es independiente del modelo, por lo que puede ser implementada con puentes H, transistores de potencia o relevadores según se requiera. En el caso particular se han utilizado 8 transistores TIP120 los cuales son NPN y soportan hasta 60 Volts y 5 Ampers máximo, en la figura 10 se muestra el diagrama empleado para energizar cada bobina, el circuito se ha de repetir 4 veces para cada motor.

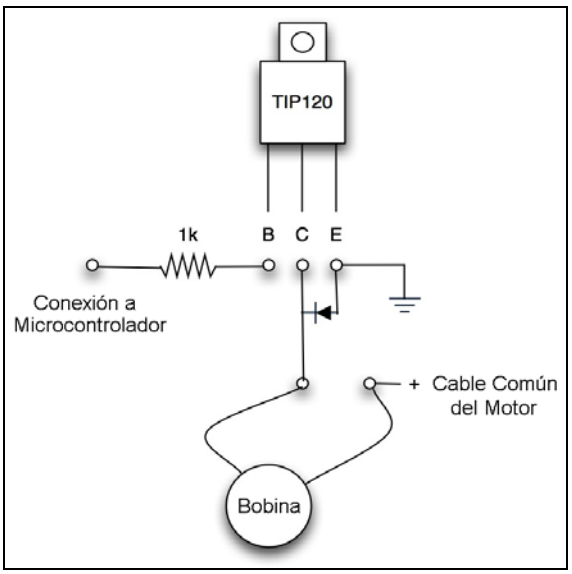

Fig. 10. Diagrama de conexiones del TIP 120

En las figura 11 se muestra el diseño de circuito impreso elaborado en Proteus Ares, en la figura 12 se muestra el prototipo en tablilla fenólica, los detalles de la implementación física (código, simulación y PCB) están disponibles bajo pedido.

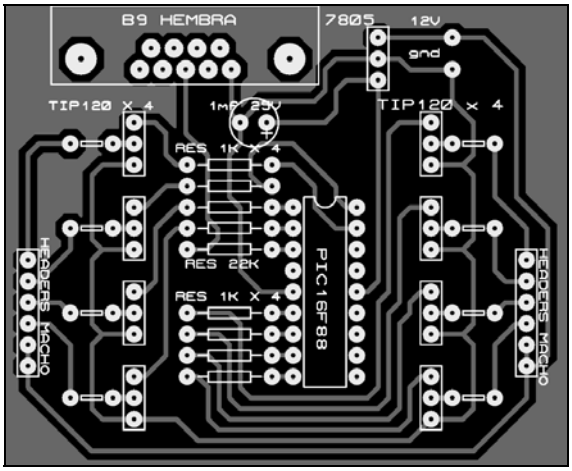

Fig. 11. Diseño PCB de la Tarjeta

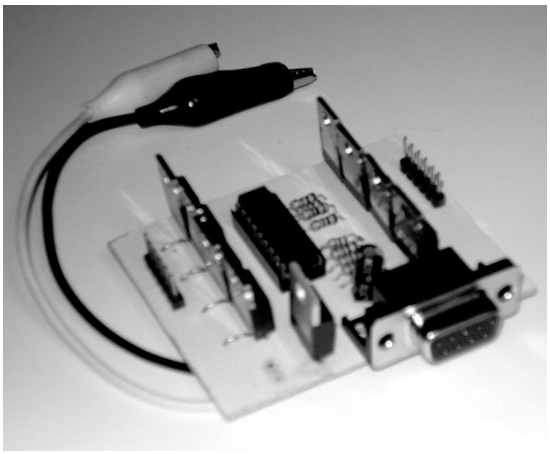

Fig. 12. Tarjeta para control de 2 motores unipolares

5. Resultados

Se logró de manera exitosa la implementación de un circuito controlador de motores de pasos unipolares, a su vez se diseñó una sencilla tarjeta que utiliza un solo microcontrolador PIC16F88, el cual es capaz de controlar dos motores de pasos unipolares de hasta 24V y 3A, en secuencias de movimiento *Full Step* y *Half Step*, sin embargo el modelo aplica para controlar *N* motores pues en el caso de requerir controlar 3 o más, solo se modificarían los *IDs* en el código de un nuevo microcontrolador, que también compartiría el bus serie.

El circuito resultante es de un tamaño reducido, es funcional y además representa un gran ahorro en el consumo de corriente pues en lugar de utilizar 4 circuitos integrados (MAX232, PIC16F88 y 2 L297) solo se utiliza uno que encapsula las 3 etapas de los otros integrados en un solo chip.

Como desventaja tenemos que no se puede hacer girar 2 o más motores al mismo tiempo salvo que tengan el mismo ID, este efecto indeseado podría reducirse al intercalar las señales por software, con tiempos pequeños podría darse el efecto de que ambos motores giran al mismo tiempo.

6. Conclusiones y Trabajo Futuro

Si bien cada autómata se encarga del control de un motor, no hay ninguna limitación lógica en cuanto a la cantidad de autómatas que se pueden implementar en el dispositivo programable, en realidad la limitación viene ligada al número de puertos con el que cuenta el dispositivo.

Una mejora podría consistir en aumentar el número de autómatas por microcontrolador ya que para efecto de este trabajo solo se ocuparon los 4 bits bajos del byte de cada puerto, pero podrían ocuparse los 4 bits altos también para implementar otro autómata y de esta forma tener 2 por puerto.

Es factible adaptar el modelo para controlar motores bipolares, pero se deben hacer cambios a nivel de código, por lo que resultaría interesante contar con 2 autómatas para unipolares y otros 2 para bipolares en el mismo integrado, se podría establecer el tipo de motores a utilizar con una señal en alguno de los bits libres de cualquiera de los puertos.

Finalmente una considerable mejora sería actualizar la interface a USB, ya que el modelo sigue siendo aplicable en ese caso.

Referencias

[1] Acuña F. *"Arquitectura FPGA para Procesamiento de Imágenes con Redes Neuronales de Función de Base Radial"*, Tesis de nivel maestría, INAOE.

[2] Watts A. y Wanio A. "*FPGA Implementation of Triple DES", 2008* Center for Systems Integration, Florida Atlantic University.

[3] Laidman R. "Stepper Motors and Control, Part IV - Microstepping of Stepper Motors", 2001. http://www.stepperworld.com/Tutorials/pgMicrostepping.htm

[4] Hoja de datos del controlador L297 en: http://www.datasheetcatalog.org/datasheet/stmicroelectronics/1334.pdf

[5] Hoja de datos del convertidor de voltajes MAX232 en: http://www.datasheetcatalog.org/datasheet/maxim/MAX220-MAX249.pdf

[6] Morales G. *"Principios de Autómatas Finitos", 2000* CINVESTAV,IPN, México. http://delta.cs.cinvestav.mx/~gmorales/ta/ta.html