PROCESADOR A MEDIDA EN FPGA PARA LA NAVEGACIÓN DE UN DISPOSITIVO MÓVIL AUTÓNOMO

N. Acosta, D. Simonelli, G. Sutter, C. Lazarte y G. Martinez

INTIA/INCA – Departamento de Computación y Sistemas - Facultad de Ciencias Exactas
Universidad Nacional del Centro de la Provincia de Buenos Aires
Campus Universitario - Paraje Arroyo Seco s/n – (7000) Tandil – Argentina

Palabras clave: FPGA, vehículos móviles, procesadores a medida de la aplicación

ABSTRACT

El objetivo de este trabajo es diseñar y construir un controlador a medida para un Dispositivo Móvil Autónomo (DMA) implementado en una FPGA XC4010E de Xilinx [Xil01]. Este dispositivo debe ser capaz de recorrer un camino previamente establecido con la suficiente autonomía como para decidir, en ciertos puntos del recorrido, el camino a seguir. Dada la generalidad y amplitud del tema, se restringe el prototipo a un sistema de navegación por marcas artificiales, utilizado para representar ciertos recorridos con (re)abastecimientos múltiples. Todo esto, en entornos aptos para el desenvolvimiento del dispositivo.

1. TÉCNICAS DE POSICIONAMIENTO

Se define navegación como la acción de conducir de un lugar a otro un vehículo y de determinar su posición en cualquier instante. No se utiliza odometría ni navegación inercial debido a la acumulación de errores. En su lugar se utilizan medidas absolutas respecto a un mapa o ruta. En este caso la posición del vehículo se calcula a partir de las medidas de los sensores respecto a elementos del entorno Así, los sensores son utilizados para medir marcas artificiales, que son elementos fácilmente distinguibles los cuales, instalados explícitamente en el entorno, hacen las veces de balizas pasivas. [Lev87]

La ventaja de las marcas artificiales es que son baratas y requieren mínimo o nulo mantenimiento. Son diseñadas para que sean fácilmente detectadas aún en condiciones ambientales adversas. Las marcas suelen ser formas geométricas características, y pueden incluir información adicional como, por ejemplo, códigos de barras.

De hecho los pocos sistemas comerciales existentes se basan en alguno de estos métodos. Por ejemplo:

a) Caterpillar Industrial ha desarrollado un vehículo, que utiliza un láser rotativo para localizarse respecto a un conjunto de marcas de posición conocida. Las marcas incluyen códigos de barras que permiten identificarlas sin ambigüedad.
b) HelpMate utiliza marcas reflectantes montadas en el techo y localizaciones fácilmente visibles.
c) Komatsu utiliza marcas metálicas en forma de Z instaladas en el suelo sobre la trayectoria del vehículo junto con odometría y giroscopio. La forma de las marcas ha sido diseñada para facilitar la corrección de la posición del robot. El método es utilizable tanto en entornos interiores como exteriores, pero limita las trayectorias del móvil al recorrido marcado.

Por otra parte, muchos de los móviles de proyectos de I+D o académicos también se basan en marcas artificales. Por ejemplo el Club de robótica y mecatrónica de la UAM[1] ha presentado un gran conjunto de móviles, la mayoría de ellos basados en marcas artificiales [Uam01] [Lp90] [Mar94].

Realizar una tarea de navegación para un robot móvil significa recorrer un camino que lo conduzca desde una posición inicial hasta otra final, pasando por ciertas posiciones intermedias o metas temporales (submeta). El problema de la navegación se divide en las siguientes cuatro etapas:

- *Percepción del mund*o: Mediante el uso de sensores externos, creación de un mapa o modelo del entorno donde se desarrollará la tarea de navegación [Gon93].

- *Planificación de la rut*a: Crea una secuencia ordenada de objetivos o submetas que deben ser alcanzadas por el vehículo. Esta secuencia se calcula utilizando el modelo o mapa de entorno, la descripción de la tarea que debe realizar y algún tipo de procedimiento estratégico.

- *Generación del camino*: En primer lugar define una función continua que interpola la secuencia de objetivos construida por el planificador. Posteriormente procede a la discretización de la misma a fin de generar el camino.

- *Seguimiento del camin*o: Efectúa el desplazamiento del vehículo, según el camino generado mediante el adecuado control de los actuadores del vehículo [Mar94].

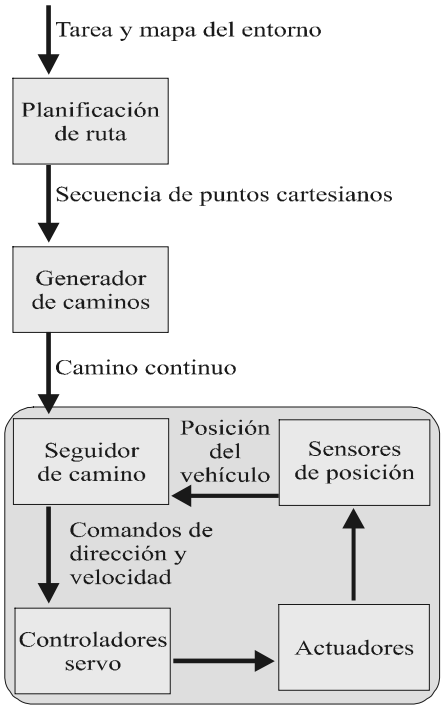

Fig. 1 – Estructura de control de navegación básica

Estas tareas pueden llevarse a cabo de forma separada, aunque en el orden especificado. La interrelación existente entre cada una de estas tareas conforma la estructura de control de navegación básica en un robot móvil y se muestra en la Fig. 1. En este esquema se parte de un mapa de entorno y de las especificaciones de la tarea de navegación. De estos datos se realiza la planificación de un conjunto de objetivos representados como una secuencia de puntos cartesianos dispersos que definen la ruta. Dicho conjunto cumple los requisitos de la tarea impuesta asegurándose de que la ruta asociada está libre de obstáculos. Mediante el uso del generador del camino se construye la referencia que utilizará el seguidor de caminos para generar los comandos de direccionamiento y velocidad que actuarán sobre los controladores del vehículo. Por último, mediante el uso de los sensores internos del vehículo (sensores de posición) en conjunción con técnicas odométricas, se produce una estimación de la posición actual, la cual será (re)alimentada al seguidor de caminos.

[1] - Universidad Autónoma de Madrid

Hay otros esquemas propuestos para a resolver esta problemática, entre ellos se puede mencionar a Levi, Brooks y Watanabe, entre otros. El esquema presentado en [Lev87] permite adaptarse a diversos entornos aunque no se posea un conocimiento exhaustivo del mismo, y se basa en dividir la planificación en dos secciones: global y local. Se habla en este caso de una descomposición jerárquica en módulos funcionales que encadenados en forma de ciclo realizan la denominada navegación estratégica. Mediante el uso de la odometría del vehículo se puede retroalimentar información, pero debido a la naturaleza del método y a las características de los sensores utilizados, la estimación efectuada se ve afectada por errores acumulativos [Lee00]. Cuando dichos errores alcanzan niveles indeseables se hace necesario eliminarlos mediante la utilización de algoritmos de estimación de la posición basados en referencias externas [Gon93]. La navegación estratégica tiene sus limitaciones en entornos dinámicos no conocidos, ya que requiere un completo conocimiento de la dinámica de los posibles obstáculos móviles, además de una adecuada actualización del mapa de entorno.

2. DISEÑO DEL NAVEGADOR

Supongamos tener un depósito central (nodo raíz), donde se almacena algún tipo de carga, una serie de puestos de trabajo donde requieran esta carga (nodos terminales u hojas) y un DMA capaz de trasladar esta carga y detectar cuando se ha vaciado (objetivo). La tarea del DMA consiste en recorrer todos los puestos de trabajo en forma sistemática (recorrido en-orden) dejando en cada puesto la carga requerida cuando el DMA queda sin carga, debe volver al deposito por más carga y es deseable que lo haga por el camino más corto posible (ahorrando tiempo y energía). Al llegar al depósito se recarga el DMA y reanuda el reparto en el puesto de trabajo donde se agotó la carga y de este modo debe continuar con la descarga por los sucesivos puestos de trabajo.

Cabe destacar que el dispositivo construido es meramente un prototipo que dista bastante de poder repartir cualquier tipo de carga, pero sí es capaz de resolver en esencia este problema.

Las consideraciones de diseño de la estructura básica del control de navegación incluyen la división en varios módulos (Fig. 2). El objetivo de tal organización es simplificar la organización del sistema utilizando módulos.

El módulo *percepción* es el encargado de percibir a través de un sistema sensorial las distintas señales que se reconocen, las cuales identifican un lugar en el espacio. Además, se encarga de la lógica de seguimiento del camino, reconocimiento de bifurcaciones y búsqueda de objetivos. Las señales (de los módulos *camino*, *fin de camino*, *choque* y *bifurcación*) son enviadas al *dispatcher* a través del módulo *Proceso de señales sensoriales*, indicando la posición del móvil sobre el camino.

El cerebro del sistema es el módulo *dispatcher* quien procesa las señales recibidas del módulo *percepción*, decide los próximos movimientos dependiendo de la actual situación y envía las señales de control al módulo *locomoción*. Al recibir la señal de posición del modulo percepción el *dispatcher* planifica el camino a seguir utilizando él modulo de *ida* o *vuelta* según corresponda.

En el módulo *control de camino* se provee el control necesario para llevar un registro que se constituye como mapa del camino recorrido. Es el encargado de realizar las operaciones necesarias para almacenar el recorrido y para hacer de interfaz con el módulo *dispatcher*. El almacenamiento del mapa es óptimo pues se elimina toda parte del recorrido que no conduce a un objetivo.

El módulo *control de locomoción* controla la dirección y velocidad de los motores. En él se puede observar como, a través del control de locomoción, el vehículo puede ser dirigido utilizando los (sub)módulos de locomoción.

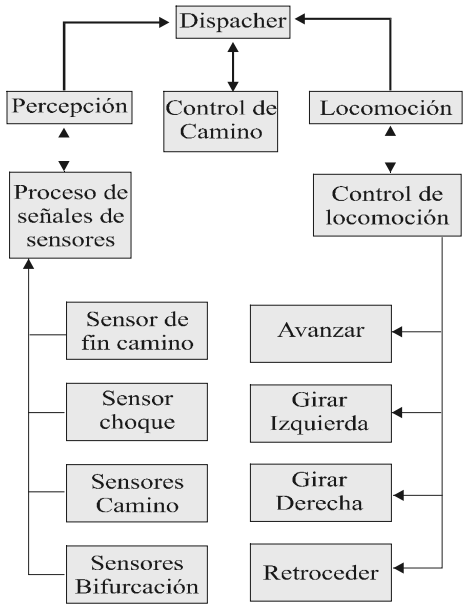

Fig. 2 – Arquitectura del controlador

3. DISEÑO DEL SISTEMA

Recorrer un camino en búsqueda de objetivos implica varias cuestiones a resolver. Se debe sistematizar el itinerario utilizando alguno de los recorridos para estas estructuras. Como se desea regresar al origen una vez encontrado el objetivo, se debe recordar que camino se ha seguido. Por otra parte, si se quiere que este regreso sea óptimo, se tiene que implementar algún tipo de algoritmo para "podar" aquellas rutas que no llevan al objetivo buscado.

El entorno donde el DMA desarrolla su actividad se compone de un camino de líneas negras (sobre fondo claro). Estas líneas están dispuestas de tal manera que forman un camino con bifurcaciones binarias ortogonales. En su camino además hay objetos sólidos contra los cuales choca el dispositivo, determinando el hallazgo de un objetivo o un fin de camino.

- Fin de camino: Indica al DMA que ha llegado a un destino (nodo terminal u hoja). Se determina por medio de una señal adquirida desde un sensor de choque.
- Objetivo: Esta marca se encuentra determinada por un objeto contra el cual el vehículo pueda "chocar", activando de esta forma un sensor de choque, que indique que se ha encontrado uno de los objetivos buscados, para reanudar el camino hacia el origen o raíz.

El algoritmo propuesto utiliza cinco sensores ópticos de los cuales dos son para seguir el borde del camino y los otros tres se utilizan para reconocimiento de bifurcaciones (Fig. 3). Y otros sensores de choque para reconocer los objetos constituidos como objetivos y finales de camino.

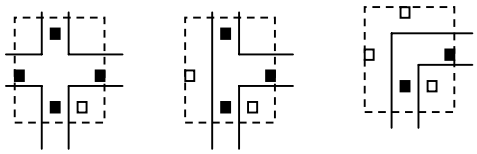
Fig. 3 - Disposición de los cinco sensores

Para memorizar el camino recorrido se parte del hecho de que por ser binario, para el camino sólo es necesario registrar la dirección tomada en cada bifurcación usando un bit ('0' para izquierda y '1' para derecha). La estructura de datos natural para almacenar las bifurcaciones es una pila.

El camino más corto es el camino directo que une el origen con el objetivo en cuestión. Todos aquellos caminos que no conducen directamente al objetivo deberían ser "podados" (habría que quitar de la pila todas las decisiones "erróneas"). Cuando se llega a un fin de camino y en éste no se halla ubicado el objetivo, se puede afirmar que la última decisión tomada es errónea. El algoritmo para corregir esta decisión se basa en quitar del registro la última decisión y cambiarla por la opción contraria. Esta forma de operar se repite cada vez que se regresa sin haber encontrado el objetivo buscado; cuando se llega al objetivo se tiene almacenada en la pila la secuencia óptima (camino directo desde el origen hasta el objetivo).

Para regresar al origen se utilizan los datos de la pila. En cada bifurcación se debe seguir por el lado que indique el tope de la misma, y realizar un POP (desapilado) para acceder al

dato anterior. Este procedimiento se repite hasta que la pila se vacíe, entonces se ha llegado al origen.

Aquí se plantea un problema con la forma de utilización de la pila, dado que si se hiciera un uso estándar de esta estructura al efectuar una operación POP el dato en cuestión se perdería definitivamente y de esta forma sería imposible retomar el recorrido desde el origen hasta la ubicación del objetivo previamente encontrado. Para solucionar este problema se agrega nueva funcionalidad a la pila para que permita recuperar nuevamente la información.

Como solución se plantea almacenar los datos en una estructura con características cíclicas que permita dos modos de operación.

- Operación efectiva: En la operación efectiva la estructura se comporta según el uso habitual de una pila, es decir cada vez que se hace una operación POP, el dato es eliminado del tope de la estructura en forma permanente; y cada vez que se realiza una operación PUSH (apilar) el dato es almacenado en el tope de la pila, desplazando hacia la parte inferior de la misma los datos previos.
- Falsa operación: Con este tipo de operación los datos no son realmente eliminados cada vez que se hace un POP, ni se ingresa un nuevo dato cada vez que se realiza una operación PUSH, sino que los mismos se reciclan dentro de la estructura, permitiendo que sean posteriormente recuperados. Para conseguir esta funcionalidad en la pila se crean dos nuevas operaciones denominadas *PopX* y *PushX* estas operaciones emulan el funcionamiento de las operaciones POP y PUSH, respectivamente, pero en realidad cada vez que se ejecuta una instrucción *PopX* el dato es movido a la última posición de la pila desplazando al resto de los valores hacia la parte alta de la pila. Además se marca cada dato almacenado en la estructura con un bit para indicar la validez del dato, es decir que un dato movido a la parte inferior de la pila por medio de una instrucción *PopX* tendrá su bit de validez en 1, indicando que ese dato ha sido virtualmente eliminado. Con respecto a la instrucción *PushX* el dato que ocupará la posición del tope de la pila, no es un dato externo, sino que se agregará el dato que ocupa la parte inferior de la pila, que coincide con el último dato pasado al final por la operación *PopX*. Para realizar una operación *PushX* es necesario que el dato que ocupa la última posición de la pila tenga su bit de validez en 1, y para realizar una operación *PopX* es condición necesaria que el tope de la pila tenga su bit de validez en 1.

Cualquiera sea el modo de operación de la pila los datos son corridos en uno u otro sentido dependiendo del tipo de instrucción ejecutada. Se debe tener en cuenta que el tamaño de la pila tiene que ser lo suficientemente grande como para almacenar el recorrido. Dicho tamaño está determinado por el nivel del árbol binario dado que es la máxima secuencia desde el origen (raíz) hasta la hoja más lejana. La Fig. 4 presenta un esquema de la estructura de la pila utilizada, mientras que la Fig. 5 muestra el seudo código del sistema de navegación.

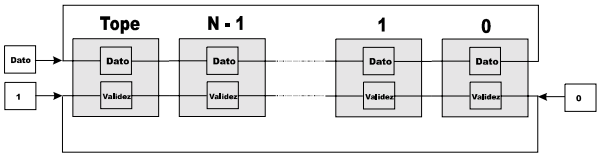

Fig. 4 - Estructura de pila circular

```
if not encontro_objetivo then
    if bifurcacion then
        if not fin then pila.push(0)
        elsif pila.tope = 1 then pila.pop
            else
                pila.push(1)
                fin = true
        end if
    endif
else
```

```
    if pila.vacia then origen = true end if
    if bifurcacion then
        if not origen then
            pila.popx
            if pila.tope = 1 then
seguirLineaIzquierda
                else seguirLineaDerecha
            end if
        else
            pila.pushx
            if pila.tope = 1 then
seguirLineaDerecha
                else seguirLineaIzquierda
            end if
        end if
end if
```

Fig. 5 – Seudo-código de navegación

6. PLATAFORMA

A la hora de implementar el diseño propuesto se consideraron diferentes alternativas. Se analizó la posibilidad de recurrir a microprocesadores genéricos para el soporte de hardware. Para ello se analizó la factibilidad de descargar la responsabilidad sobre motherboards estándares basadas en microprocesadores Pentium. Sin embargo se descartó este enfoque pues la dimensión del proyecto evidenció que no se justifica tamaña plataforma.
También se analizaron dos familias de microcontroladores: Motorola y Microchip, arribándose a la conclusión de que por costos y curva de aprendizaje la primera no resultaba conveniente teniendo en cuenta los objetivos planteados con el trabajo. De esta manera, se adoptó el microcontrolador PIC 16C56 de Microchip [Mic01] como base y se realizó un primer prototipo. El microcontrolador evidenció tiempos de respuesta elevados y escasa memoria por lo que se encaró el diseño de un sistema a medida basado en FPGA.

Las características tenidas en cuenta para la toma de tal decisión fueron la alta velocidad del ciclo de desarrollo y la facilidad de testeo. La implementación de los módulos del sistema a partir de código VHDL aceleraron enormemente el proceso debido, en parte, a la simplicidad conceptual que presenta el hecho de escribir código fuente en un lenguaje de alto nivel para definir el comportamiento de los subsistemas y, por otra parte, a la potencia que presentan las herramientas de síntesis para trasladar la especificación funcional a una FPGA. [Add01] [Emb97] [Ter97]

En este sentido, la familia adoptada fue la serie 4000E de Xilinx y todo el diseño de la unidad de control se centró en FPGA Express y Xilinx Foundation Series. El soporte físico lo constituye una FPGA 4010EPC84 montada sobre una XS40 Board [Xes01] [Xs01]. El control de movimiento se basó en dos motores paso a paso controlados por drivers de potencia UCN-5814B (Fig. 6).

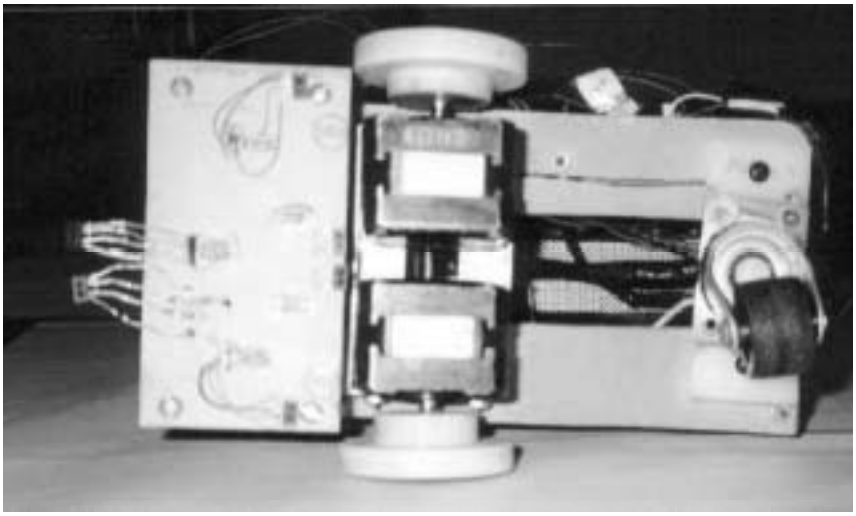

Fig. 6 – Control de movimientos del DMA

6.1. Implementación

Todas las actividades concernientes al DMA pueden ser implementadas en la FPGA utilizada, incluyendo el control de los motores, la toma de señales desde los sensores y el manejo de la memoria, donde se debe recordar la secuencia utilizada parar recorrer el camino desde el origen hasta el objetivo.
Para implementar el control en una FPGA se realiza una serie de componentes en VHDL cada uno de los cuales tiene una funcionalidad específica, a continuación se hace una reseña brindando detalles a cerca de cada uno de estos componentes.

6.2. Programación de la FPGA

Una vez exportado el archivo XNF desde FPGA Express se puede verificar el correcto funcionamiento del proyecto mediante el módulo de simulación [Dvb99]. El programa realiza la simulación funcional del proyecto por medio de la generación de estímulos sobre las entradas de la misma. En este ejemplo se puede observar la simulación del modulo principal. En las filas superiores, se observan las salidas correspondientes al control de los motores y a continuacion todas las entradas como necesarias, reset, reloj y sensores.

En la fig. 7 se muestra un recorrido junto a la secuencia de señales que se generaron. El ejemplo contempla todos los casos posibles de estados.

Al iniciar el recorrido, el DMA recorre el circuito siguiendo el borde derecho de la línea (señal inversa en bajo), en ciertos momentos es necesario cambiar la lógica de recorrido al borde izquierdo (inversa en alto), las señales el_tope y es_valido muestran el tope de la pila y su bit de validez mientras que spop, spush, spopush, spopx y spushx representan las señales de operación de la pila. Los sensores 1 y 2 son los encargados de seguir la linea, mientras que los sensores 3, 4, 4B, y 5 sirven para detectar las bifurcaciones. La señal alta corresponde a la lectura del color blanco y la baja al color negro. Las entradas sensor choque y sensor fin, corresponden a los microswitches que indican la presencia de un objetivo y un final de camino, respectivamente.

- Mainreset corresponde a la entrada de reset, mainclk al reloj de la FPGA, y reloj_lento a la señal de reloj normalizada.
- MotorDer y MotorIzq corresponden a las salidas que se conectan a los drivers de potencia de los motores paso a paso, estas señales representan la secuencias de 4

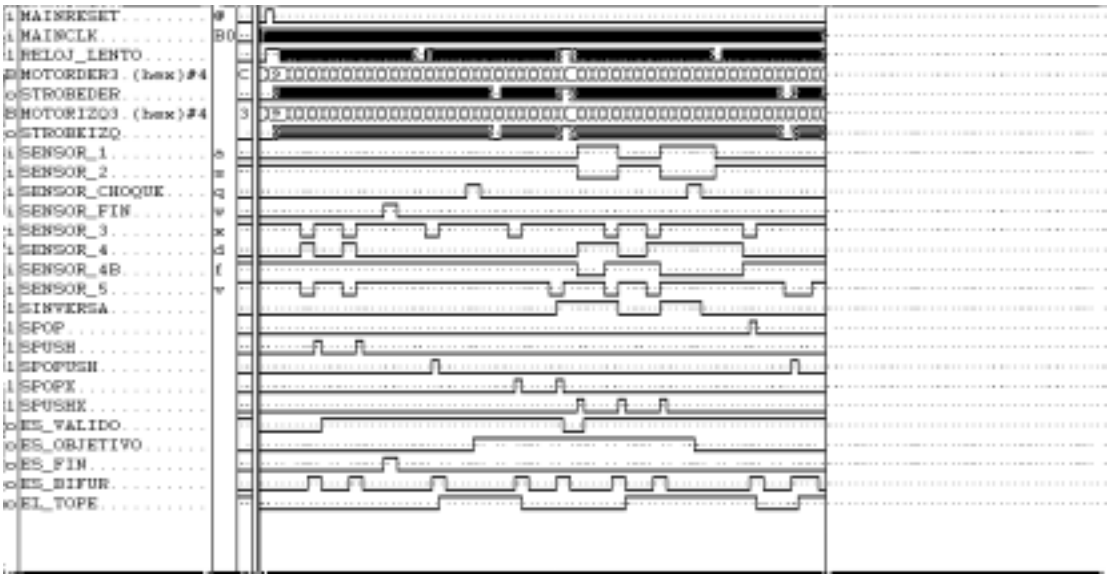

Fig. 7 - Ciclo completo de simulación

bits. Finalmente, las señales de strobeizq y strobeder corresponden a las habilitaciones de cada motor.

La Fig. 8 muestra un esquema del diseño de todo el sistema.

7. PRUEBAS

Para la realización de pruebas se utilizaron diversas configuraciones de pistas con el propósito de comprobar el correcto funcionamiento del DMA. Los resultados obtenidos en las pruebas fueron satisfactorios: el Dispositivo Móvil Autónomo reaccionó correctamente ante las distintas situaciones propuestas.

La Fig. 9 muestra tres fotos del DMA materializado (vista lateral, superior e inferior). La Fig. 10 muestra una secuencia en una de las pistas en las cuales se realizaron las pruebas. La Fig. 11 muestra un diagrama de eventos que se producen al simular el sistema. Las señales en el gráfico son las siguientes:

- Bifurcación: indica la detección de una bifurcación en el recorrido, y es determinado por medio de la lectura de señal baja (cero) en dos de los sensores destinados a la detección de bifurcaciones.
- Fin de Camino: indica la detección de un fin de camino en el recorrido, y es determinado por medio de la lectura de una señal alta (uno) en el sensor de fin de camino.
- Origen: indica que el vehículo se encuentra en el origen del recorrido, y es determinado por medio de la lectura de tope de pila vacío.
- Objetivo: indica la detección de un objetivo, y es determinado por medio de la lectura de señal alta (uno) en el sensor de choque destinado a la detección de objetivos.

8. CONCLUSIONES

A pesar de los condicionantes tecnológicos, se logró desarrollar un prototipo, capaz de resolver en esencia el problema de la búsqueda de objetivos en un laberinto ortogonal, conformando una aplicación completamente dedicada sobre un dispositivo Hardware.

Si bien la estructura electromecánica del DMA es precaria, el prototipo permitió la realización de pruebas y el análisis de los casos de la aplicación. Por otra parte, es una primer experiencia para el desarrollo de un dispositivo capaz de realizar esta tarea en entornos reales.

La velocidad promedio en las pruebas fue de 4,54 cm/seg. El consumo tiene 2 valores distintos: durante la configuración es de 1.32A y en funcionamiento es de 0.72A.

Como trabajos factibles de realizar tomando como base esta experiencia se presentan las siguientes variantes:

a) Permitir al DMA evitar obstáculos durante el recorrido (utilizando sensores ópticos o de sonido para medir la distancia).
b) Utilización de las técnicas empleadas para vehículos futuros. En este momento se está diseñando un ‘cadete electrónico’, donde el DMA es el mensajero entre oficinas.
c) Análisis de técnicas de reducción de consumo para DMA, con el propósito de reducir el tamaño y peso de la batería.

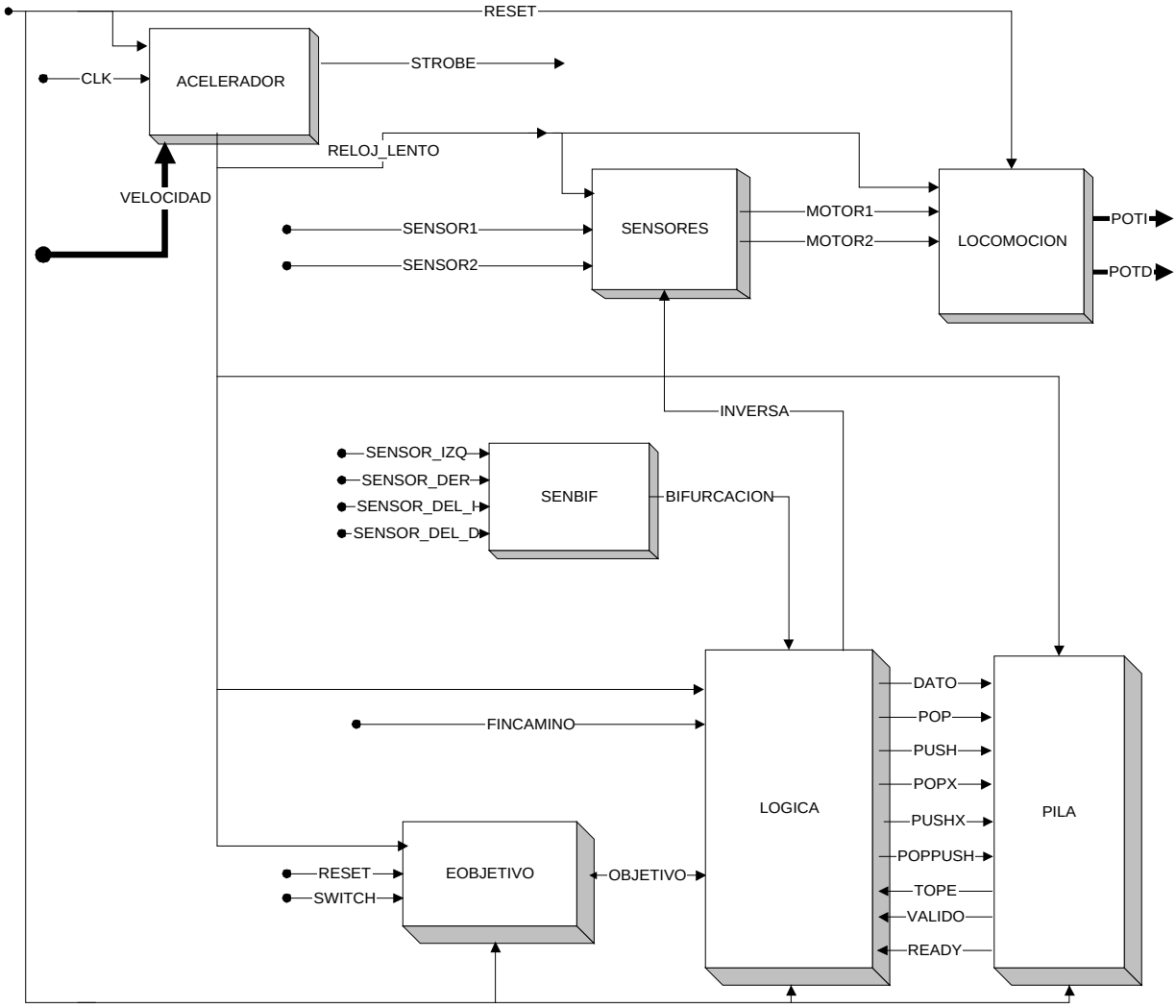

Fig. 8 - Diseño esquemático de todo el sistema

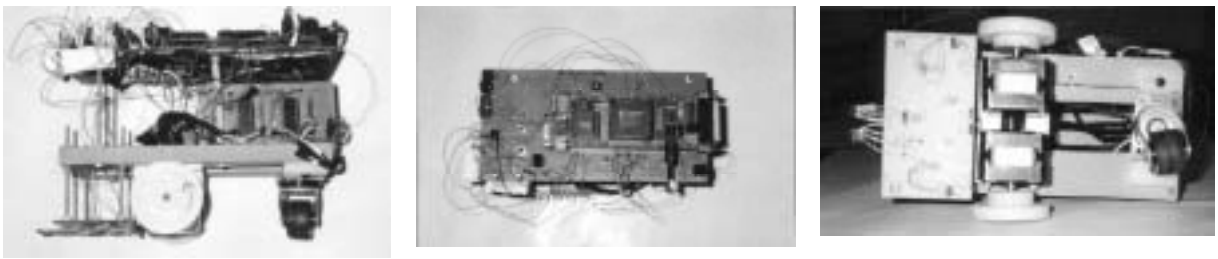

Fig. 9 - Diseño del DMA, vista lateral, superior e inferior

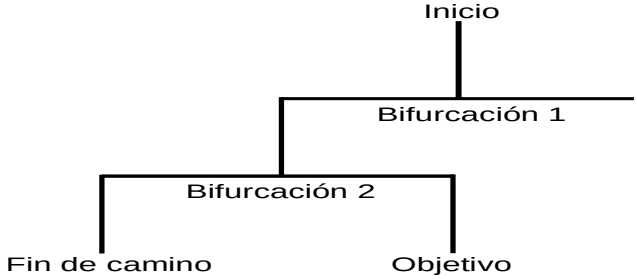

Fig. 11 – Diagrama de eventos del sistema

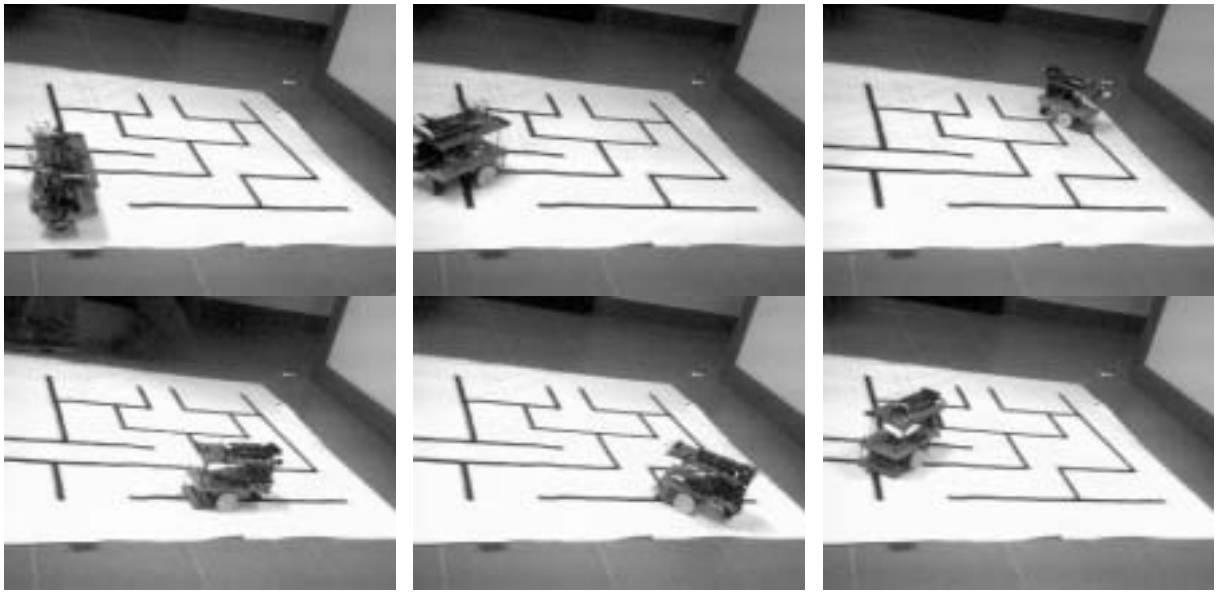

Fig. 10 – Secuencia de recorrido en una pista

9. REFERENCIAS

[Add01] Jay K. Adams, Donald E. Thomas, "The Design of Mixed Hardware/ Software Systems", Proceedings of the 33rd Design Automation Conference, 1996.

[Dvb99] Dave Van den Bout, "Quick Start Guide for Foundation F1.3", http://www.synopsis.com

[Emb97] http://www.embedded.com/97/feat9706.htm

[Gon93] González J. (1993) "Estimación de la Posición y Construcción de Mapas para un Robot Móvil Equipado con un Escáner Láser Radial". Tesis Doctoral. Universidad de Málaga.

[Lee00] Lee, C.S.G. Sensor-Based Robots: Algorithms and Architectures.

[Lev87] Levi P. (1987) "Principles of Planning and Control Concepts for Autonomous Mobile Robots". Proc. of 1987 IEEE International Conference on Robotics and Automation. pp 874-881.

[Lp90] Lozano-Pérez T. (1990) "Foreword: Mobile Robots and Robotics". Autonomous Robot Vehicles. Editores I.J. Cox y G.T. Wilfong. Springer-Verlag. pp vii-xi.

[Mar94] Martínez J.L. (1994) "Seguimiento Automático de Caminos en Robots Móviles". Tesis Doctoral. Universidad de Málaga.

[Mic01] Microchip Inc. http://www.microchip.com

[Sio01] Sandler Ben-Zion. Robotics: Designing the Mechanisms for Automated Machinery.

[Ter97] Lluís Terés/Yago Torroja/Serafín Olcorz/ Eugenio Villas, "VHDL Lenguaje Estándar de Diseño Electrónico", McGraw-Hill, 1997

[Uam01] Escuela de mecatrónica http://www.uam.es.

[Xes01] Xess Corporation - Home Page http://www.xess.com

[Xil01] Xilinx Data Book", Xilinx Inc., 2001, http://www.xilinx.com.

[Xs01] Herramientas para volcado y testeo de *bitstreams* en placas XS40, XS90, http://www.xess.com/fpga.