DISEÑO DEL MICROCONTROLADOR 8051 CON MÓDULO ENSAMBLADOR- GENERADOR DE ROM EN LENGUAJE VHDL

Ezequiel Franco[1], Fernando Montero[1], Enrique Ostúa[2], Manuel J. Bellido[2], Paulino Ruiz- de- Clavijo[2], Alejandro Millán[2], David Guerrero[2], Jorge Juan Chico[2]

[1] ETS Ingeniería Informática, Universidad de Sevilla
[2]Dpto. de Tecnología Electrónica, Universidad de Sevilla
[2]Instituto de Microelectrónica de Sevilla, Centro Nacional de Microelectrónica, CSIC
{ ostua, bellido, paulino, amillan, guerre, jjchico}@dte.us.es

RESUMEN

En este trabajo se presenta el resultado de un Proyecto Fin de Carrera en el que se ha diseñado el microcontrlador 8051 en lenguaje VHDL. Para llevar a cabo este trabajo se ha empleado el entorno ISE, de Xilinx, así como la herramienta de simulación de VHDL Modelsim, de Mentor Graphics. Uno de los aspectos más interesantes del trabajo es el desarrollo de un módulo ensamblador- generador de ROM, realizado también en código VHDL, lo que sirve para mostrar la versatilidad de este lenguaje de descripción de hardware.

1. INTRODUCCIÓN

Dentro del panorama actual de la tecnología electrónica nos encontramos con todo un mundo de interesantes posibilidades de aplicacion industrial en los llamados *System on Chip*. Fundamentalmente su tecnología nos permite incluir dentro de un único chip lo que hasta ahora era un sistema electrónico de control completo. Básicamente un SoC es un sistema implementado sobre una única pastilla de silicio, lo cual significa disponer de sistemas que tienen un tamaño muy reducido y, por tanto, son facilmente empotrables en cualquier sistema de control industrial. Dentro del SoC se implementan todos los componentes del sistema electrónico. El control de la funcionalidad del sistema se lleva a cabo a través de un microcontrolador incorporado al SoC.

Uno de los microcontroladores más utilizados en los sistemas de control industrial ha sido el 8051, del que existen diferentes implementaciones físicas desarrolladas por distintos fabricantes. Los circuitos integrados que implementan al 8051 se emplean en sistemas de control diseñados sobre PCBs. Cuando se pretende incorporar el sistema completo en un SoC es necesario disponer del diseño del microcontrolador para poder incorporarlo al chip. Un aspecto muy interesante en este tipo de diseños es que deberia ser en la medida de lo posible, independiente de la tecnología sobre la que se va a fabricar el SoC. Una alternativa para lograr esta independencia tecnológica es realizar el diseño empleando un lenguaje de descripción de hardware. De esta forma se puede disponer de un código que puede emplearse tanto para la simulación del sistema como para realizar su síntesis sobre una determinada tecnología empleando las herramientas adecuadas.

El propósito de este documento es presentar el diseño del microcontrolador 8051 [1] en lenguaje VHDL [2] adecuado para empotrarlo en aplicaciones *System on Chip* de sistemas de control industrial, y su posterior implementación sobre una FPGA. El

estudio ha sido realizado como un Proyecto de Fin de Carrera de un estudiante de la titulación de Ingeniería Informática de la Universidad de Sevilla. Para el trabajo han utilizado los entornos de CAD de Diseño Electrónico de Xilinx ISE [3] y Modelsim de Mentor Graphics [4].

El objetivo principal del proyecto ha sido el diseño y verificación en lenguaje VHDL del microcontrolador 8051 partiendo de su especificación a nivel ISA (Instruction Set Architecture). Se ha realizado una verificación exhaustiva del código generado empleando Modelsim como simulador de VHDL y, además, para comprobar que es código sintetizable se ha realizado su implementación sobre FPGA empleando el entorno ISE de Xilinx. Un aspecto que hay que resolver en el diseño en código VHDL de un microcontrolador como el 8051 es el programa que se va a ejecutar en el sistema de control. En la tecnología de PCB's los microcontroladores incluyen una EPROM que es programable con el dispositivo adecuado. En cambio, en la tecnología SoC por una parte nos encontramos con que no vamos a poder incluir memorias programables (por ejemplo en FPGA's) o bien puede resultar muy costoso (en tecnología ASIC incluyendo memorias FLASH EPROM); y por otra parte, en un SoC, el programa implementado no va a cambiar.

Ante el inconveniente que supone no poder contar con memoria EPROM, pero con la ventaja que supone el que el programa que se va a utilizar es fijo para cada SoC, la alternativa de diseño que nosotros hemos elegido es diseñar un módulo ensamblador que, a partir del código de un determinado programa escrito en lenguaje ensamblador del 8051, genere el código VHDL sintetizable de la ROM del 8051 para ese programa. Una vez que se dispone de ese código, se puede realizar la simulación del sistema 8051 completo incluyendo el programa que se va a ejecutar.

El resto de este artículo está dedicado a introducir los puntos principales del desarrollo del proyecto presentado, comenzando con una breve presentación de las características del 8501; posteriormente se analiza la metodología de diseño empleada a lo largo del proyecto, destacando los aspectos más importantes del diseño, con especial interés en el módulo ensamblador- generador de ROM; se presenta la realización de pruebas de simulación realizadas sobre el diseño VHDL del microprocesador y los resultados de la implementación sobre la FPGA, terminando con un resumen de las principales conclusiones del trabajo realizado.

2. DESCRIPCION BASICA DEL MICROCONTROLADOR INTEL 8051

Como ya hemos mencionado, el microcontrolador 8051 de Intel ([5] y [6]) es el objeto del proyecto realizado. Esta basado en los microprocesadores de 8 bits de Intel, como el 8086. Básicamente contiene una CPU de 8 bits y dispone de 3 puertos de entrada y salida paralelos y un puerto de control. Puede direccionar 64K de programa y 64K de datos separadamente, es decir un total de 128Kbytes. Ademas cuenta con una memoria RAM interna de 128 bytes.

La memoria de sistema del 8051 se clasifica en tres partes fundamentales: la primera, llamada memoria de programa, una ROM de 4 Kbytes (ampliables a 64K con una ROM

externa), es donde se encuentran todas las instrucciones que van a ser ejecutadas por el 8051, es decir, el programa a ejecutar; el segundo espacio es la memoria de datos, que podría ser una RAM externa de hasta 64 Kbytes, accedida durante las operaciones de

CARACTERÍSTICAS DEL MICROCONTROLADOR 8051	
REGISTROS	**32 REGISTROS DE PROPÓSITO GENERAL DE 8 BITS REGISTRO DE FUNCIONES ESPECIALES (SFR): ACUMULADOR (ACC), REGISTRO B, STACK POINTER (SP), DATA POINTER (DPTR), PUERTOS 0, 1, 2 Y 3 (P0, P1, P2, P3), PALABRA DE ESTADO DEL PROGRAMA (PSW)**
MEMORIA	**RAM INTERNA DE 128 BYTES. ROM INTERNA DE 4KBYTES (AMPLIABLE A 64K – ROM EXTERNA). PUEDE DIRECCIONAR 64K DE PROGRAMA Y 64K DE DATOS**
UNIDAD ARITMÉTICO-LÓGICA (ALU)	**OPERACIONES ARITMÉTICAS: SUMAS, RESTAS, DIV, MULT. OPERACIONES LÓGICAS: AND, OR, NOT, XOR. DESPLAZAMIENTOS: RL, RLC, RR, RRC. AJUSTE DECIMAL (BDC)**
CONJUNTO DE INSTRUCCIONES	**INSTRUCCIONES: DE ACCESO A MEMORIA (INTERNA Y EXTERNA), ARITMÉTICAS, LÓGICAS, DESPLAZAMIENTO, SALTOS (CONDICIONALES E INCONDICIONALES), MOVIMIENTO DE TABLAS Y DE CONTROL.**

Tabla 1: Características Básicas del 8051.

lectura y escritura de datos (no se puede ejecutar ninguna instruccion que se encuentre almacenada en este espacio de memoria); y por ultimo tenemos tambien un espacio de memoria RAM interna, de 256 bytes, conteniendo multitud de registros internos del procesador y para funciones especiales.

Desde el punto de vista del programador del microcontrolador del 8051 será preciso conocer por una parte todos los registros internos, por otra el juego completo de instrucciones que incluye junto con los diferentes modos de direccionamiento que permite utilizar.

En la memoria RAM interna de 256 bytes que incluye el 8051 nos encontramos dos bloques de 128 bytes bien diferenciados: en la parte baja tenemos 4 bancos de 8 registros de propósito general, (R0- R7), un espacio de direccionamiento bit a bit (de 16 bytes) y un espacio libre (de 80 bytes); en la parte alta se encuentran los registros de funciones especiales (SFR's), donde nos encontramos el Acumulador (ACC), el Registro B (B), la palabra de estado de programa (PSW), el Stack Pointer (SP), el Data Pointer (DPTR) y los cuatro puertos de entrada/salida (P0, P1, P2, P3). A su vez en el registro de estado de programa (PSW) se refleja el estado de la CPU en ese determinado instante de tiempo, con una serie de banderas como la de Carry, el Carry Auxiliar, el Selector del Banco de Registros, la de Overflow, la de Paridad y una bandera de usos generales.

En cuanto al juego de instrucciones, el 8051 tiene instrucciones booleanas (movimiento, limpieza, establecimiento, complementación y operaciones AND y OR), instrucciones de saltos (condicionales e incondicionales) e instrucciones de transferencia de datos (con la RAM interna, con una RAM externa y de movimiento de arrays). Los modos de direccionamiento que tiene el 8051 son los siguientes: directo, indirecto, inmediato, indexado y por registro.

En la Tabla 1 se muestra un resumen de las principales características del microcontrolador.

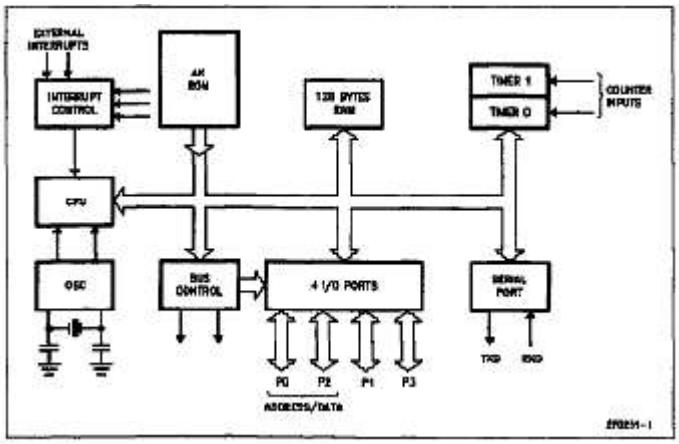


Figura 1. Diagrama de Bloques del 8051

3. METODOLOGÍA DE DISEÑO

Para la realización de este proyecto de diseño del 8051 se parte de las especificaciones del fabricante, Intel [2], que son, sobre todo, especificaciones funcionales, de comportamiento, por lo que se dice poco en ellas sobre la arquitectura interna de cada uno de los bloques que componen el microcontrolador.

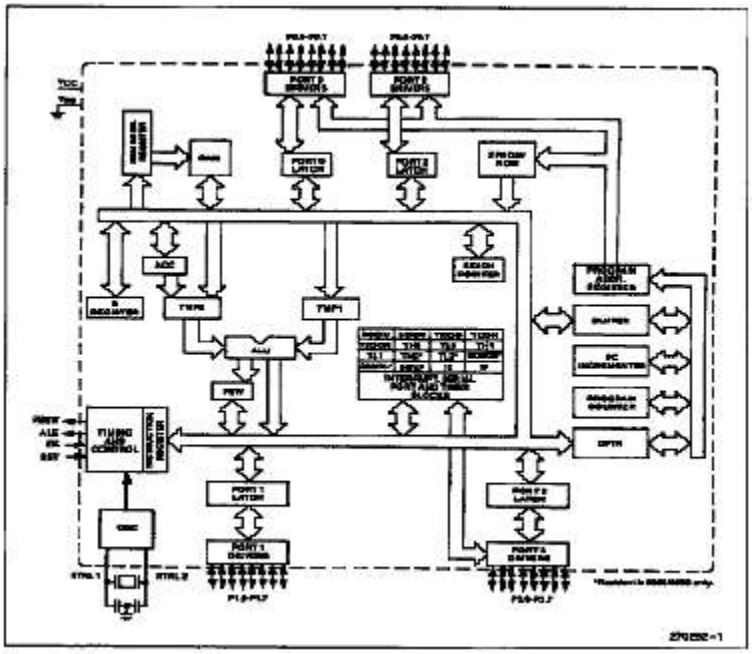


Figura 2. Arquitectura Interna del 8051

En la figura 1 se muestra el único esquema que aparece en esta documentación que se ha manejado sobre el microcontrolador 8051. En ella podemos apreciar la CPU, la ROM, la RAM, los puertos de entrada/salida y otros elementos del microcontrolador

como timers, oscilador, etc... Por otra parte, en la descripción arquitectural interna de la CPU de la figura 2, vemos que la información que proporciona el fabricante es algo confusa, ya que no se identifican claramente los bloques por los que está compuesta la CPU.Por esto, al plantearse la metodología de trabajo que se llevaría en este proyecto, se decide descartar la idea de realizar una descripción estructural frente a una descripción funcional o de comportamiento. Así pues, la metología de diseño utilizada atiende a una estructura Top-Down, siguiendo el esquema tradicional que se muestra en la figura 3.

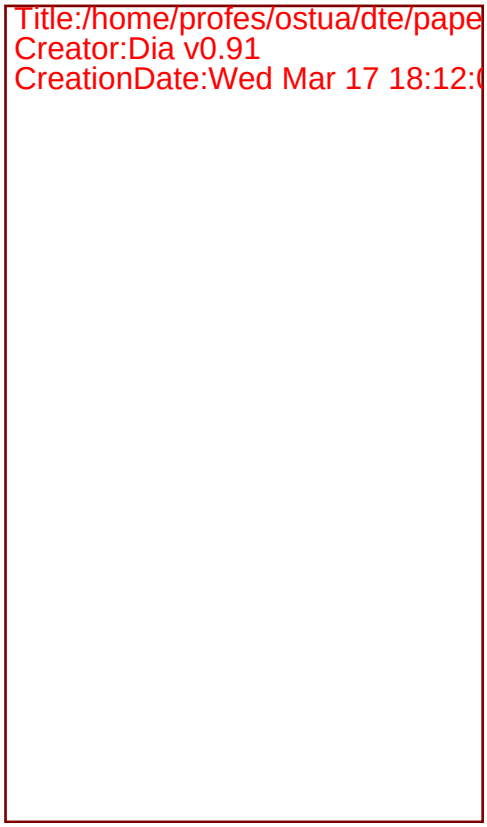


Figura 3. Metodología de Diseño del 8051

Trás un profundo análisis de toda la documentación disponible se han identificado cuatro grandes bloques sobre los que se fundamenta todo el diseño, a saber: Unidad de Control, Memoria RAM, Memoria ROM y la ALU. Si bien el diseño de la RAM, ROM y ALU no presentan problemas dignos de reseñar, con la Unidad de Control se presenta el problema de que no se conocen los estados internos por los que discurre el flujo de ejecución de instrucciones del microcontrolador, por lo que se opta por recurrir a los estados típicos de un microprocesador, que son los estados de RESET o iniciación, INTERRUPCIÓN, BÚSQUEDA y EJECUCIÓN. Sin embargo, aunque la realización de los dos primeros estados es directa, en los dos últimos nos volvemos a encontrar nuevos problemas de diseño, motivados porque en el 8051 no tienen un formato de instrucciones único sino muy variable, incluso pudiendo estar compuesta por uno, dos y hasta tres bytes, en función del modo de direccionamiento usado en la instrucción. Para resolver el problema que se plantea en la fase de búsqueda, ya que, a priori se desconoce si la instrucción consta de 1, 2 o 3 palabras de memoria, se ha decidido incluir un decodificador del primer byte de la instrucción. Este decodificar se encargará de dos funciones principales: la traducción del formato 'no uniforme' nativo del 8051, a un

formato uniforme con sólo 7 bits que identifica unívocamente tanto a la instrucción como el modo de direccionamiento; y por otra parte identificará, mediante 2 nuevos bits, el tamaño en bytes de la instrucción, indicando la necesidad de captar un segundo o incluso tercer byte en la fase de búsqueda.

En la fase de ejecución se ha descompuesto cada una de las instrucciones del 8051 en microinstrucciones que operan ciclo de reloj a ciclo de reloj. Una vez realizado el análisis de los estados en los que se va a encontrar el procesador se puede proceder al diseño del mismo.

4. DESARROLLO DEL PROCESO DE DISEÑO LLEVADO A CABO PARA EL MICROCONTROLADOR 8051.

Existen dos alternativas básicas para llevar a cabo el diseño: una pasaría por realizar un diseño para una tecnología específica mediante, por ejemplo, captura de esquemáticos [7]. El inconveniente de esta alternativa es que el resultado final es tecnológicamente dependiente, lo cual significa que para otra tecnología habrá que realizar un diseño nuevo completo.

La otra alternativa es utilizar un lenguaje de descripción de hardware, como VHDL [8] y [9], lo cual permitiría que al cambiar la tecnología, la mayor parte del diseño (o incluso en muchas ocasiones en su totalidad) se podría reutilizar.

Por otra parte, con un LDH es posible realizar una descripción a nivel estructural o bien una descripción de comportamiento. Esta última, al ser una descripción de más alto nivel tiene como ventajas que facilita la independencia tecnológica y, además, permite realizar modificaciones al diseño de una forma mucho más simple que las descripciones estructurales. El inconveniente principal está en que al efectuar la síntesis sobre una tecnología el resultado va a ser peor que el de la descripción estructural. Nosotros hemos elegido realizar una descripción de comportamiento para implementar el 8051.

En la figura 4 se muestra un diagrama de la máquina de estados que se ha implementado.

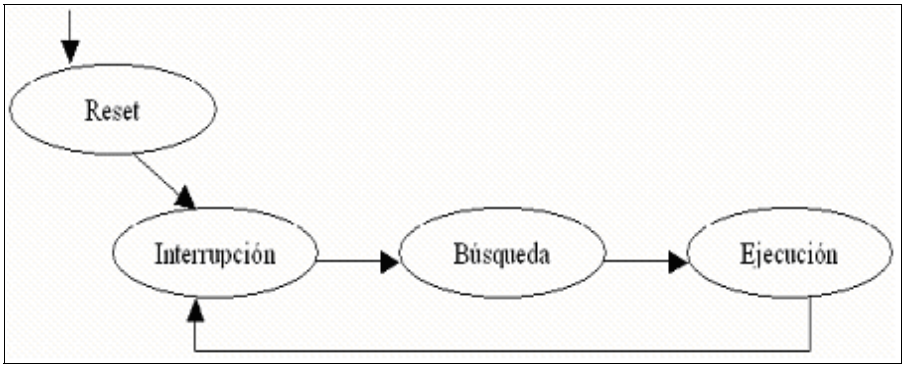


Figura 4. Máquina de estados del 8051

4.1 Ensamblador- Generador de ROM para el microcontrolador 8051.

Como ya mencionamos en la introducción, uno de los aspectos que hay que tener en cuenta al desarrollar un diseño de un microcontrolador empotrable en un SoC es cómo se va a incorporar la ROM donde se almacene el programa de control que debe ejecutar el micro. Una alternativa podría ser incluir una memoria Flash- Eprom, pero el gran inconveniente está en que la mayoría de las tecnologías no disponen de este tipo de memorias y, además, en aquellas en que si es factible resulta muy costoso. En cambio, la ventaja que posee un SoC es que es un sistema de aplicación específica donde el programa que se va a ejecutar es fijo. Esto permite una alternativa de diseño que consiste en diseñar el bloque específico de ROM que contiene el programa que va a ejecutarse, cada vez que se va a implementar un SoC.

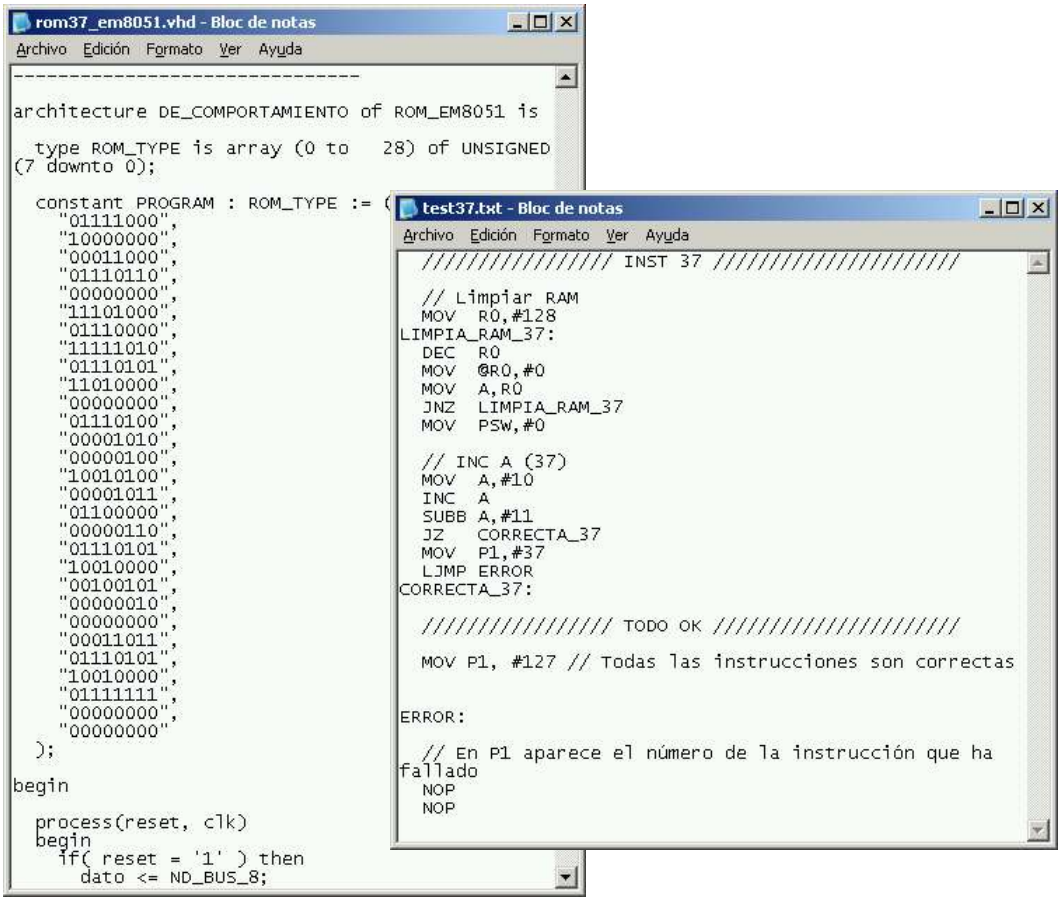
Es evidente que para que esta alternativa sea eficiente es necesario poder generar esta ROM de forma lo más automática posible, a partir del programa escrito en algún lenguaje de programación. Por esto es necesario desarrollar algún tipo de 'compilador'. En este proyecto se ha desarrollado un módulo ensamblador- generador de ROM, cuyo modo de operación básico consiste en generar el código de la ROM en lenguaje VHDL a partir del código en lenguaje ensamblador del programa.

El generar la ROM en lenguaje VHDL permite la simulación del sistema completo, incluida la ejecución del programa de control. Además, al tener la ROM en VHDL es posible realizar su síntesis junto con el resto de componentes del microcontrolador.

Uno de los aspectos más interesantes de este módulo ensamblador- generador de ROM es que ha sido implementado en VHDL. La ventaja principal que supone tener el propio ensamblador- generador de ROM en este lenguaje es que a la hora de realizar el proceso de depuración del programa de control del sistema que se está diseñando no es necesario cambiar de entorno de trabajo para hacer cambios. Esto es, el propio simulador de VHDL (en nuestro caso Modelsim de Mentor Graphics [4]) es el que se utiliza para generar el código VHDL de la ROM que, automáticamente se añade al sistema completo con el programa que va a ejecutarse. La modificación del programa se realiza sobre el código en ensamblador e inmediatamente es posible relanzar la simulación para volver a testar el funcionamiento del sistema. En las figura 5 se muestra un ejemplo de un programa en ensamblador y el código VHDL de la ROM generada por nuestro módulo ensamblador- generador de ROM.

5. VERIFICACIÓN E IMPLEMENTACIÓN DEL DISEÑO DEL 8051

Una de las grandes ventajas que ha supuesto el disponer del módulo ensamblador- generador de ROM ha sido el facilitar enormemente la realización de una verificación exhaustiva del comportamiento del diseño del microcontrolador 8051. Efectivamente, este módulo permite generar el código ROM de un programa para el 8051 y la herramienta de simulación permite realizar una verificación de la ejecución de la ejecución de dicho programa. Teniendo en mente esto y para verificar lo más exhaustivamente el diseño, el procedimiento que se ha ideado ha consistido en preparar unos programas en ensamblador, cada uno con el fin de testar una determinada



Figuras 5. Código Ensamblador para testar una instrucción del 8051 (INC A) y memoria ROM en VHDL obtenida con el módulo ensamblador- generador de ROM.

instrucción, y que ofrezcan un resultado final simplemente indicando si la ejecución de la instrucción ha dado un resultado correcto o no correcto (esto se indica colocando en un puerto de salida un valor u otro al finalizar la ejecución de ese programa). Estos programas de testeo tienen todos una estructura bastante similar, en la que es posible modificar la instrucción que se está verificando de tal forma que es posible realizar una comprobación de todos y cada una de las instrucciones con unos mínimos cambios.

En la figura 6 se muestra un ejemplo de este programa de testeo (para la instrucción ADD A, #data). En la figura 7 se tiene el código VHDL de la ROM generada para este programa y en la figura 8 se muestra el resultado de la simulación del microcontrolador 8051 con la ejecución del programa.

Por último, se ha utilizado el entorno ISE de Xilinx para comprobar la capacidad de síntesis del código VHDL generado [10]. El resultado es positivo si bien teniendo en cuenta que la descripción empleada para el diseño ha sido realizada a un alto nivel (descripción de comportamiento del microcontrolador), el porcentaje de componentes de la FPGA que se utilizan es relativamente elevado, como se observa en la Tabla 2.

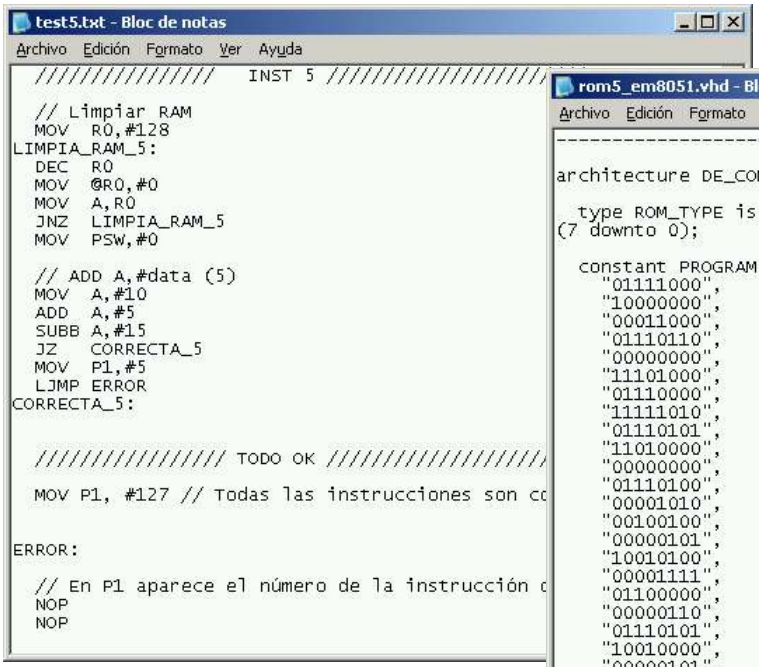


Figura 6. Código Ensamblador para probar una instrucción (ADD A, #data)

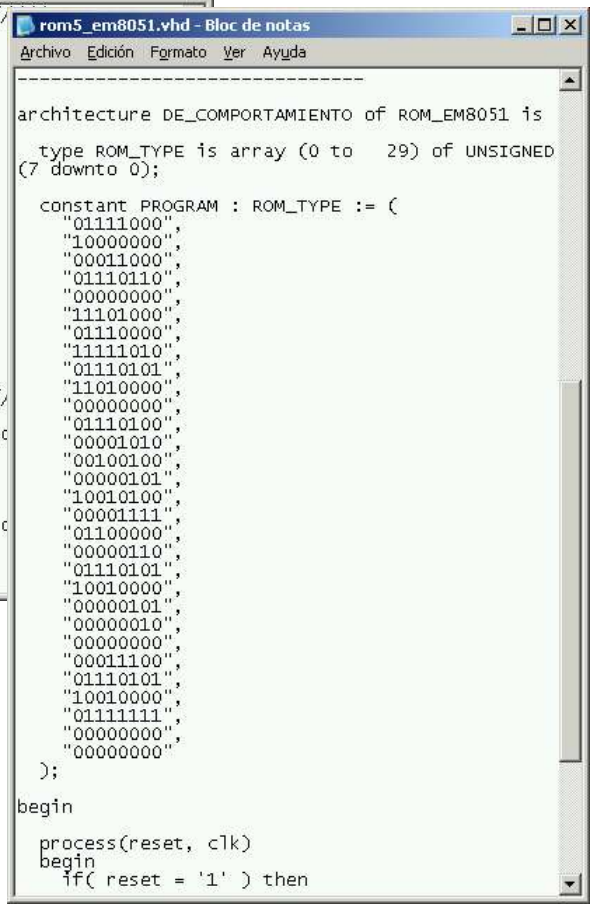


Figura 7. Memoria ROM en VHDL obtenida a partir de ese programa en ensamblador.

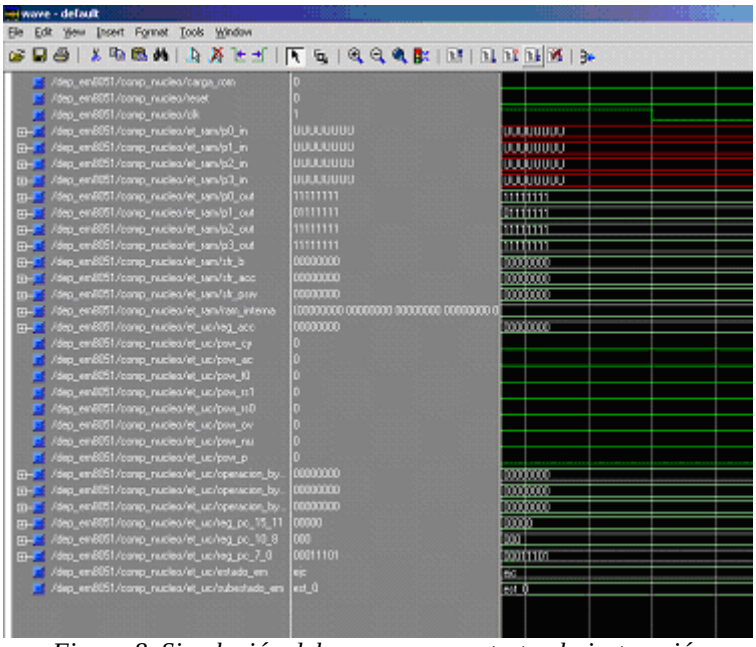


Figura 8. Simulación del programa que testea la instrucción.

SÍNTESIS	FPGA	% OCUPACIÓN
MICRO 8051	VIRTEX-E XCV300E	89%

TABLA 2. RESULTADOS DE LA SÍNTESIS DEL 8051 SOBRE FPGA.

6. CONCLUSIONES

En este artículo se ha presentado el diseño en VHDL del microcontrolador 8051y su posterior simulación y síntesis sobre una FPGA de Xilinx. Como principal resultado del proyecto desarrollado podemos destacar la adecuación del diseño realizado al campo de la enseñanza de la microelectrónica, al estar realizado el diseño en un lenguaje de descripción hardware, el VHDL, sencillo de entender desde el punto de vista del alumno y que abre unas enormes a la hora de trabajar con él de forma práctica, propiciando el poder introducir cambios o innovaciones en el microcontrolador objeto del estudio.

Otro elemento interesante que se ha descrito ha sido el módulo ensamblador del 8051, codificado en VHDL, y que nos permite obtener un bloque de memoria ROM 'ya microprogramado' con el código escrito por el programador y facilmente intercambiable en caso de necesidad. Resulta particularmente útil a la hora de la verificación y testeo del diseño del micro, siendo además un módulo independiente del mismo.

7. AGRADECIMIENTOS

Este trabajo ha sido realizado bajo la financiación de los proyectos MCYT TIC2000-1350 MODEL, TIC2001-2283 VERDI y DGV PHB2002-0018-PC del Gobierno Español.

8. REFERENCIAS

[1] F. Pardo, 'VHDL. Lenguaje para descripción y modelado de circuitos'. Univ. de Valencia.
[2] MCS 51 Microcontrollers - Intel Corporation, http://www.intel.com/design/mcs51/index.htm
[3] ISE - Xilinx Inc., http://www.xilinx.com
[4] Modelsim - Mentor Graphics, http://www.modelsim.com
[5] A. Vega, 'Manual del microcontrolador 8051' y 'Aplicaciones del 8051'.
[6] Jaka Simsic & Simon Teran, '8051 Core Specification', http://www.opencores.org
[7] M. Estevez, 'Diseño del microprocesador ARCI con DFWII', PFC de la Univ. de Sevilla, 2002.
[8] L.A. Barragán y J.I. Artigas, 'Diseño y modelado digital con VHDL y síntesis lógica'. Univ. de Zaragoza.
[9] E. Ostúa, 'Diseño del microprocesador SRC con FPGA de Xilinx'. PFC de la Univ. de Sevilla, 2003.
[10] S. Holgado, 'Tutorial ISE de Xilinx, Manejo Básico'. Univ. Autónoma de Madrid.