

Aplicación de una FPGA en la reingeniería de una memoria RAM de una computadora naval

Application of a FPGA in the reengineering of a RAM memory of a naval computer

CHRISTIAN L. GALASSO, MIGUEL A. BANCHIERI, ADRIÁN H.

LAIUPPA, EDUARDO ZITO Y MARTÍN E. PAZ

Escuela de Oficiales de la Armada, Facultad de la Armada,
Universidad de la Defensa Nacional, Argentina.

Departamento de Ingeniería Electrónica, Universidad Tecnológica
Nacional, Facultad Regional Bahía Blanca, Argentina.

Servicio de Análisis Operativos, Armas, y Guerra Electrónica,
Armada Argentina, Base Naval Puerto Belgrano, Argentina.

christian_galasso81@yahoo.com.ar

En el presente trabajo, se describe la implementación de una FPGA en el marco de la reingeniería de una memoria RAM de una computadora de propósito específico. Esta tiene capacidad de 16 K palabras de 28 bits (24 bits de dato + 4 bits de paridad). Su tiempo de acceso es de 750 ns. El diseño original está basado en lógica discreta TTL 5 [V] y en 28 circuitos integrados de memoria RAM dinámica, que le brindan al dispositivo total versatilidad en la forma de escritura de la memoria.

1. Introducción

Se abordó el trabajo de reingeniería de una placa de memoria RAM de 16 K palabras de 28 bits, realizada mediante lógica discreta que requiere ser sustituida por una placa que tenga la misma funcionalidad pero con los mismos temporizados. El desarrollo se encargó mediante una FPGA SPARTAN 3 y dado que la RAM interna de la FPGA no cubre la capacidad requerida, se utilizó un integrado de memoria RAM basada (SRAM) externa. Este tipo de control permite acceder a un byte, control que un acceso a la memoria RAM que se desea replicar implica cuatro accesos a la SRAM (32 bits).

Entre las opciones, se pensó en usar cuatro integrados para realizar un único acceso a la SRAM por cada acceso a la memoria, pero se descartó por la excesiva necesidad de PINES I/O que esta solución requería. También se evaluó utilizar una FPGA con mayor capacidad de memoria interna, para no requerir un integrado externo. Se encontró que el dispositivo con la capacidad mínima necesaria (mayor a 458.752 bits) tenía un costo que triplicaba el costo del dispositivo elegido. Asimismo, los encapsulados disponibles se reducían solo a BGA, con lo cual el proceso de soldadura aumentaba en complejidad, mientras que en el dispositivo anterior se podía conseguir con encapsulado SMD.

Entre las características deseables del prototipo, también está el hecho de que debe soportar condiciones de operación exigentes de programación a temperaturas elevadas, vibraciones de baja frecuencia, interferencia electromagnética de potencia, y ser capaz de ocupar el espacio físico programado para el montaje de los aditivos. En los componentes del sistema la parte de observación, por lo cual se requiere de un modo de uso de dispositivos de forma determinada. Esto abre un abanico de posibilidades en áreas aun no exploradas. Por ejemplo, el hecho de reconocer un rostro o el habla de una persona para otro ser humano es relativamente sencillo. Sin embargo, si

2. Fundamentos

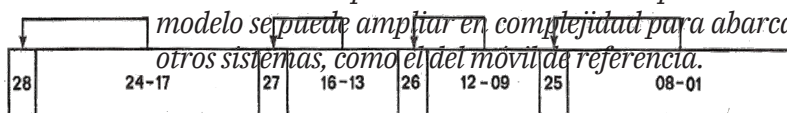
Las especificaciones particulares del hardware de memoria, a reemplazar

a) Chequeo de paridad: escribir un código para resolver esto, sería

muuy complicado. El aprendizaje automático zanja este problema sin decirle a la aplicación que hacer, solo con mostrarle ejemplos, puede obtener resultados satisfactorios. Esto puede extender las posibilidades de crear sistemas aún más complejos. La aplicación de las redes neuronales en la navegación autónoma ha sido empleada con asiduidad por la empresa Tesla en sus modelos de automóviles autónomos que ya han salido al mercado para aprender de su experiencia en las rutas y caminos. El

FIGURA 1 Este artículo es a los efectos de mostrar el algoritmo de una red neuronal sencilla

ciñéndose a un tipo de vehículo subacuo disponible. Este modelo se puede ampliar en complejidad para abarcar



1. Introducción

La ecuación del cálculo de paridad está conformada únicamente por compuertas XOR entre los distintos bits que pertenecen a cada grupo y un pin que ingresa externamente. Este abarca la percepción de señales de diversos dispositivos que dependiendo su estado, determina si la paridad es par o que, en base al plan establecido (teoría de navegación), impar. En cada escritura de memoria, la paridad es calculada y transmiten las ordenes a los sistemas de gobierno y propulsión y almacenada, y en cada lectura se compara el valor de los bits que forman en conjunto el control del vehículo. La manera en que navegue el dispositivo hará que este pueda alcanzar a cumplimentar el/los objetivo/s buscado/s. Todo esto, por supuesto, supervisado por el/los operador/es humano/s ubicado/s en una/s estación/es de trabajo que analizarán y comandarán las acciones de la embarcación. Asimismo,

b) Lectura y escritura
existirá un conjunto de submódulos relacionados que

de la memoria RAM se empleará para guardar los datos de los sensores para formar la palabra completa de 24 bits sin embargo, para el control del dispositivo se usará un subconjunto de bits para representar diferentes formas como se especifica en la Tabla I. La palabra que utiliza la computadora es de 24 Bits y puede dividirse en dos medias palabras de 12 Bits o en tres adecuadas las ordenes mencionadas podría emplear una red neuronal que obtenga la adecuada señal a cada motor

TABLA 1
Tipos de escritura y lectura de la memoria RAM.

TIPOS DE L/E	WE4	WE3	WE2	WE1
Escribe palabra completa	0	0	0	0
Escribe mitad derecha de la palabra	1	1	0	0
Escribe mitad izquierda de la palabra	0	0	1	1
Escribe carácter de la izquierda	0	1	1	1
Escribe carácter de la derecha	1	1	1	0
Escribe carácter del medio	1	0	0	1
Lee palabra completa	1	1	1	1

3. Desarrollo

Hardware propuesto para la solución

a) La FPGA

En el caso de la FPGA, se optó por un integrado de la familia de Xilinx SPARTAN 3, la misma posee dispositivos de grado industrial que pueden aplicar a las condiciones ambientales extremas a las que estará sometido el diseño en condiciones normales de operación. Algunos miembros de esta familia poseen encapsulado SMD, con lo cual se cumple la premisa de mantener baja la complejidad de armado y posterior reparación. En el caso del número de pines I/O, los dispositivos de la familia que poseen encapsulado SMD cuentan con la cantidad suficiente de pines como para manejar todas las

entradas y salidas del diseño y además poder direccionar de a un byte una memoria de hasta 4 Mbit. Esta limitación en la cantidad de pines, que se tradujo en una limitación en el tamaño de la palabra que podía leerse o escribirse en memoria al mismo tiempo, fue determinante para el planteo de la solución.

b) La memoria RAM estática

En el caso del circuito integrado de memoria RAM, se optó por los dispositivos del tipo RAM estática asíncrona, para eliminar la necesidad de refresco. De todos los encontrados, se tomó uno de 1-Mbit ($128\text{ K} \times 8$), con tiempos de acceso de 10 ns y rango de temperatura de trabajo para uso industrial. Se impuso que el tiempo de acceso no superara los 25 ns, para poder manejar la lectura y escritura mediante una máquina de estados con un *clock* de 40 MHz.

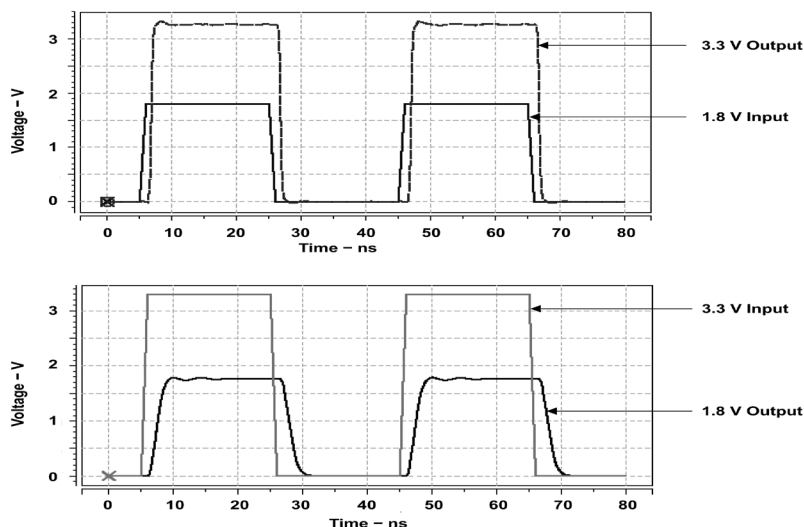
c) Las adaptaciones de entrada-salida

El bus paralelo de E/S tiene niveles de tensión de 5 V (TTL), mientras que la FPGA maneja niveles de 3,3 V (LVTTTL). Esto determinó la necesidad de utilizar circuitos integrados de adaptación de nivel. En cuanto a los buffers bidireccionales, de los diversos tipos de adaptación de niveles que pueden utilizarse, el análisis se centró en dos, los dispositivos de adaptación de doble fuente y los dispositivos tolerantes a sobretensión.

En el caso de los adaptadores doble fuente, son circuitos integrados diseñados para la comunicación asíncrona entre dos buses o dispositivos que utilizan diferente alimentación. Poseen una alimentación denominada VCCA para el puerto A y una denominada VCCB para el puerto B. La forma de onda que puede obtenerse durante una traslación de nivel se ve en la Figura 2.

FIGURA 2

Forma de onda de un dispositivo doble fuente durante una traslación de 1,8 V a 3,3 V y de 3,3 V a 1,8 V ($CL = 15 \text{ pF}$, $RL = 2 \text{ k}\Omega$).



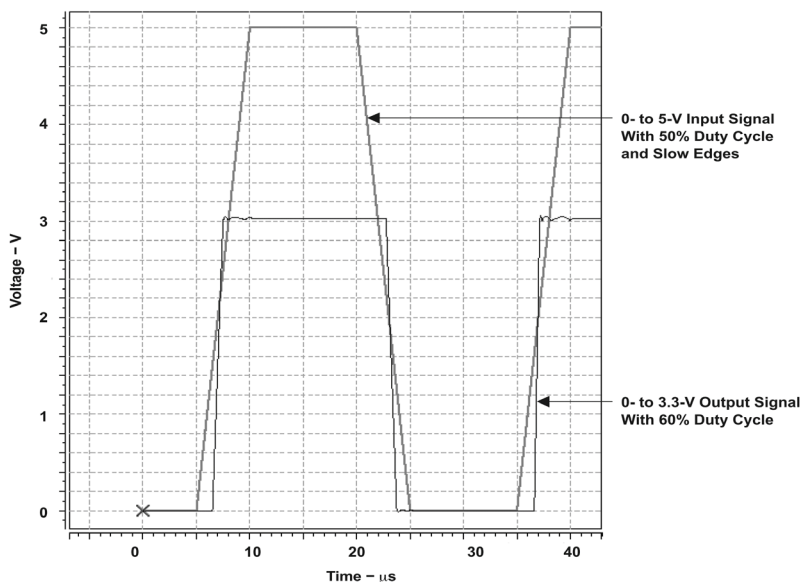
Los dispositivos con E/S tolerantes a sobretensiones se alimentan con una única VCC, que para el caso de LVTTTL serían 3V3. Están diseñados con óxidos de compuerta más gruesos, que toleran tensiones superiores a la utilizada para alimentar el circuito integrado, y eliminan el diodo de entrada que fija la tensión a VCC. Dentro de sus familias, existen algunos que llegan a manejar niveles de corriente de 64 mA, mientras que los dispositivos doble fuente no superan los 32 mA. Este mayor manejo de corriente era una característica deseable para la aplicación en desarrollo. Como contrapartida, si las señales de entrada tienen un bajo *slew rate*, el ciclo de trabajo de la señal de salida podría verse afectado. Un ejemplo de esto puede verse en la Figura 3, donde se aplica a las entradas niveles de 0 a 5 V. El dispositivo de traslación, al estar alimentado con VCC = 3V3, cambia a niveles de umbral de lógica LVTTTL. La señal de entrada tiene tiempos de subida y bajada lentos, similares a los de la aplicación en cuestión. El resultado obtenido es un

cambio en el ciclo de trabajo de salida.

Dado que al momento de la compra de los componentes todavía no se había definido si se utilizaría el reloj provisto por el bus o un cristal interno, se descartó como opción de traslación de nivel a los dispositivos tolerantes a sobretensiones, por no ser la solución de traducción ideal en aplicaciones donde el ciclo de trabajo de salida es crítico, como por ejemplo una señal de reloj.

FIGURA 3

Cambio en el ciclo de trabajo de la señal cuando se utilizan integrados tolerantes a sobretensión para la adaptación de niveles TTL 5 V a LVTTTL 3V3.



d) Descripción en VHDL

Bloques

Para emular el *handshake* del dispositivo en cuestión, se

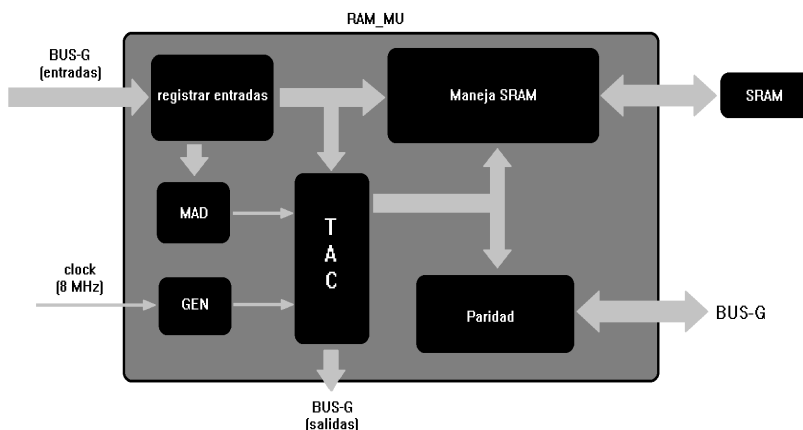
desarrolló un conjunto de diferentes bloques, para favorecer el orden, la claridad de comprensión y el testeado del diseño sobre la FPGA. Luego de analizar la lógica de funcionamiento y los temporizados del circuito de la RAM original, se pudo definir la cantidad mínima de bloques, cada uno con ciertas funciones particulares, que, al ser vinculados en un bloque integral, permiten establecer la correcta comunicación del sistema con la memoria RAM.

- Bloque registrar_entradas: este bloque se encarga de registrar todas las entradas para eliminar problemas de meta-estabilidad, que se dan cuando se manejan señales que provienen de dominios de relojes distintos. Además, genera las señales WE (WE1 a WE4), que permiten determinar la operación a realizar sobre la memoria.
- Bloque CLOCK_BUS: Dado que finalmente se decidió que el reloj del bus denominado CLOCK_BUS (8 MHz) era demasiado lento para manejar la máquina de estados que se planteó como solución, se diseñó un bloque que sincronizara el flanco de dicha señal con el del reloj interno (40 MHz).
- Bloque Decodificador de Dirección de Memoria (DDM): Este bloque selecciona el módulo de memoria a utilizar, debido a que la computadora de propósito específico tiene cuatro o más unidades de memoria RAM de 16 K.
- Bloque Temporizado y Control (TyC): Lleva a cabo el temporizado y control de las señales requeridas para iniciar los ciclos de memoria.
- Bloque de Paridad: realiza un control de paridad de los datos. Además, si hay una falla referida a esto, presenta el correspondiente aviso al procesador del sistema.
- Bloque Maneja_SRAM: se encarga del manejo del integrado de memoria SRAM y de la comunicación de este con la FPGA.

- Bloque RAM_MU: Es la estructura que engloba todos los otros módulos previamente descriptos. Se encarga, además, de implementar el temporizado y habilitaciones de los puertos E/S de: La FPGA, la SRAM y los BUFFERS de traslación de nivel. Recibe, a través de estos últimos, la E/S del bus del sistema y las asocia con los respectivos bloques.

FIGURA 4

Diagrama en bloques del VHDL.



El Bloque Maneja SRAM

Dadas las diferencias entre el hardware de la memoria RAM original y el del prototipo desarrollado, se presentaron dificultades en el diseño del VHDL para lograr que los distintos tipos de escrituras parciales funcionaran correctamente. Para tal fin, se diseñó la estructura Maneja_SRAM, la misma se describió a su vez en dos partes: El bloque FSM y otro bloque (HW_REGISTROS_IO_SRAM) que contiene toda la lógica necesaria para realizar la descomposición de la palabra de 32 bits (24 bits de dato + 4 bits de paridad + 4 bits en '0').

Maneja_SRAM, recibe los 4 bits (WE1 a WE4) que le indican el formato de la palabra que se va a escribir o leer

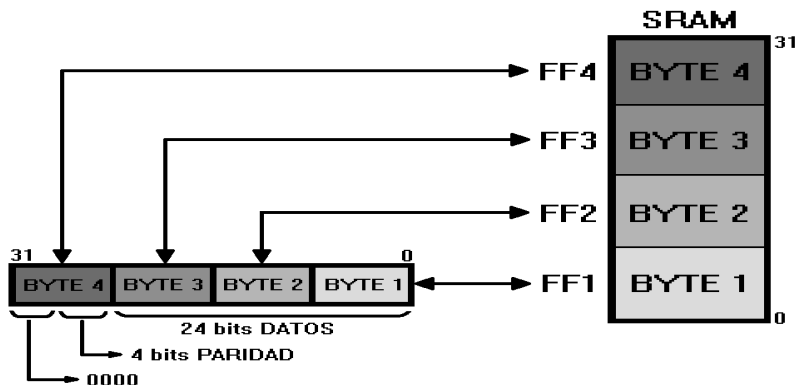
de la memoria. Recibe, además, la señal de inicio del bloque TyC para comenzar un ciclo de memoria, junto con otra señal de validación que indica el momento exacto donde el valor que está en el bus del sistema es el correcto para ser leído y guardado en memoria (ciclo de escritura). Del bloque PARIDAD se reciben los 28 bits que vienen del bus del sistema junto con la paridad generada sobre ellos. Por el contrario, si el ciclo es de lectura de memoria, el bloque Maneja_SRAM es el encargado de componer la palabra de 28 bits para entregarla al bloque de PARIDAD para que la genere nuevamente y compare los bits de paridad generados con los previamente guardados.

La FSM y el bloque HW_REGISTROS_IO_SRAM

El ciclo de lectura fue relativamente sencillo de implementar, mediante una FSM que tomaba los 4 bytes almacenados en memoria y los colocaba por orden en un registro de 32 bits para, posteriormente presentar el dato en el bus y viceversa para la escritura (completa), como puede apreciarse en la Figura 5.

FIGURA 5

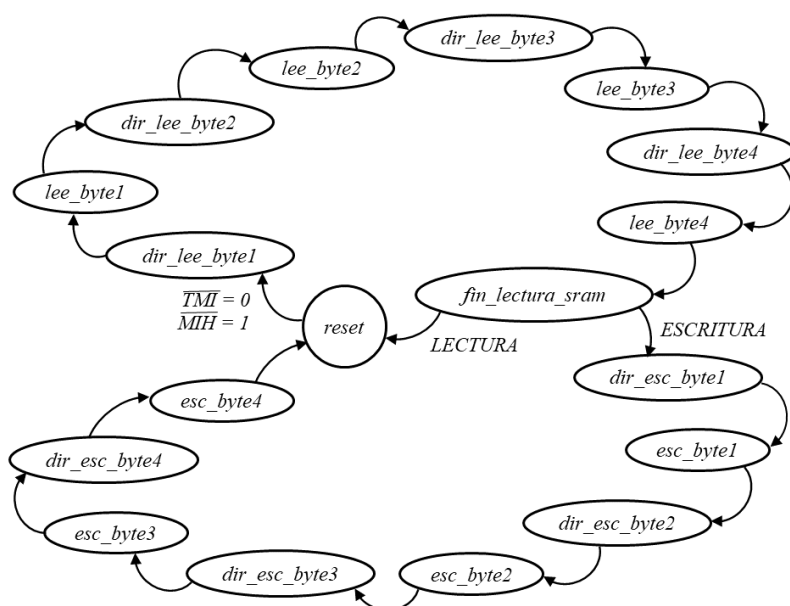
Orden de la escritura/lectura de la palabra de 32 bits en el chip de la SRAM.



La FSM [3] implementada para este fin dentro del bloque se denominó FSM_LEEYESCRIBE4B, la misma se diseñó con 18 estados como puede verse en la Figura 6.

FIGURA 6

Grafo de la máquina de estado propuesta.



La FSM está normalmente en estado “reset”, y sale de allí mediante las señales que se corresponden a un inicio de ciclo de memoria. Se determinó que, tanto para la lectura como para la escritura, se necesitaba leer la palabra guardada en la memoria SRAM, por lo que al inicio de un ciclo de memoria (cualquiera sea) se comienza automáticamente con un ciclo de lectura sobre la SRAM.

Para cumplir con el *handshake* del integrado de SRAM, se dividieron en dos partes los procesos, tanto de lectura como de escritura. En un ciclo de FSM se coloca la dirección y en

el siguiente se realiza la acción (lectura o escritura), de forma de asegurar los tiempos mínimos de establecimiento de las direcciones y los datos en los puertos correspondientes de la SRAM. Asimismo, se armó la señalización de los pines CE, OE y WE para garantizar que hubiese una separación de varios nanosegundos entre el momento en que el puerto de la FPGA se pone en lectura y el momento en que la SRAM se pone en escritura, y viceversa. Así, se asegura que nunca se dará la situación de que ambos puertos estén en escritura en simultáneo. Como puede observarse en la Figura 7, se utilizó la escritura en memoria SRAM controlada por CE. Y, como se muestra en la Figura 8, se utilizó el ciclo de lectura controlado por la transición de la dirección del dato.

FIGURA 7

Resultado de la simulación mediante Test Bench del handshake de las CE OE y WE de la SRAM durante la escritura.

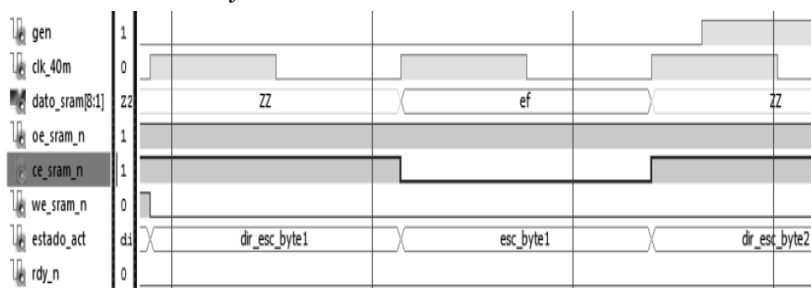
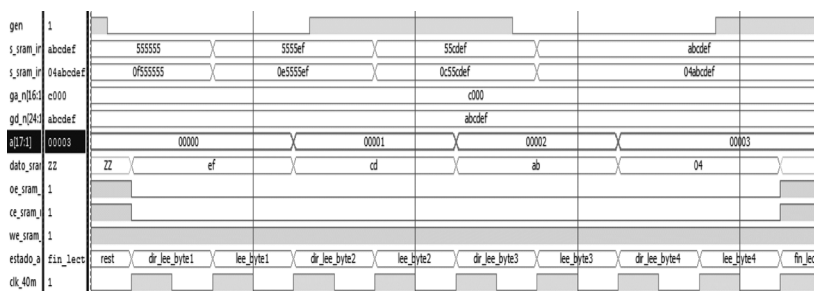


FIGURA 8

Resultado de la simulación mediante Test Bench del handshake de las CE OE y WE de la SRAM durante la lectura.



Al final de la lectura de los 4 bytes se agregó un ciclo denominado “fin_lectura_sram” con la finalidad de evitar problemas en la lectura del cuarto y último byte que se lee de la memoria. Por temporizado, cuando se llega a este estado ya se tiene establecida la operación a realizar con la memoria (lectura o escritura). Si la operación es de lectura, se comienza con el protocolo para pedir al microprocesador habilitación para escribir sobre el bus bidireccional. Si es de escritura, el dato válido a escribir ya se encuentra en el bus, por lo que el mismo es almacenado en registros internos para ser procesados posteriormente por la FSM.

e) Hardware para resolver las escrituras parciales

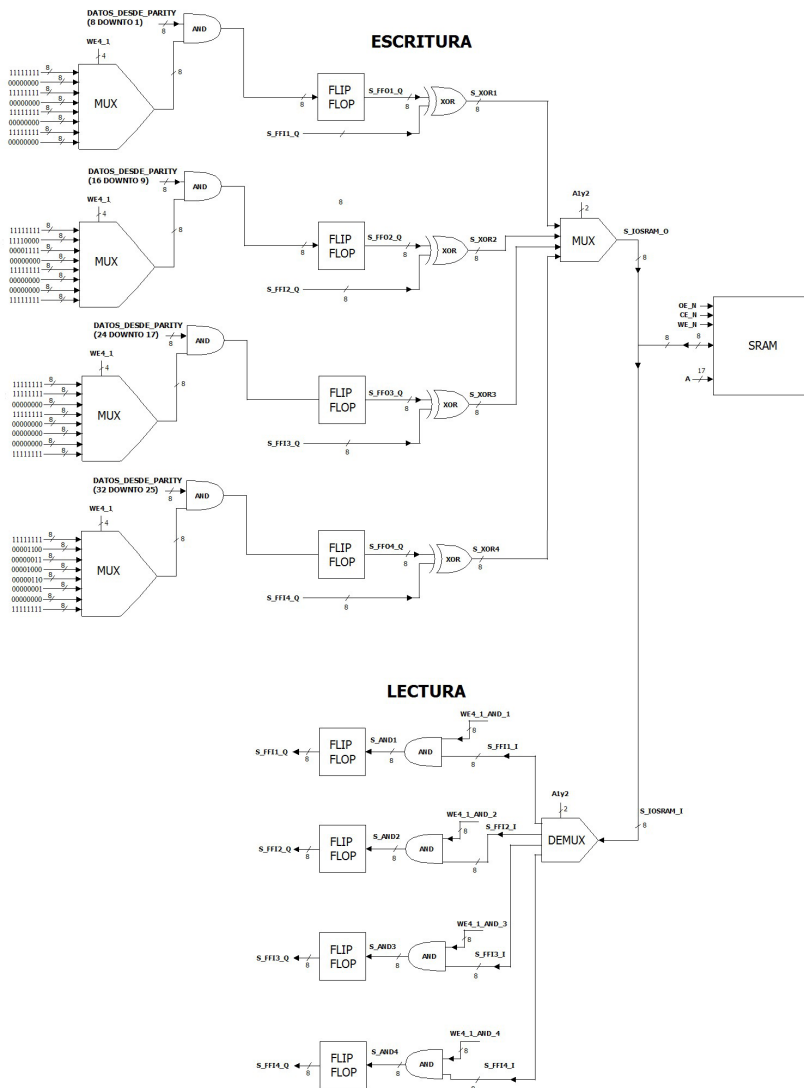
Uno de los desafíos de esta implementación fue describir un hardware que modificara solo aquellos bits de la palabra que se querían escribir y dejar intactos el resto. Especialmente el último byte almacenado en la memoria SRAM que contenía los 4 bits de paridad y los 4 ceros lógicos para completar; donde, dependiendo el tipo de escritura parcial, debían modificarse uno o dos de los bits de paridad, manteniendo el resto.

En primer lugar, se implementaron compuertas AND bit a bit, entre cada uno de los cuatro bytes extraídos de la SRAM (durante el ciclo de lectura), con un valor binario particular seteado en cada caso, dependiendo del tipo de escritura a realizar (completa, media palabra o carácter). El valor con el que se realiza la AND es un ‘0’ lógico en la posición de los bits que se desean sobrescribir y un ‘1’ en aquellos que se desean mantener como se han leído de la SRAM. Se implementaron, también, otra serie de compuertas AND con los datos que ingresan desde el bus del sistema, donde los valores ‘1’ y ‘0’ lógicos están invertidos con respecto a las AND que se realizan con los datos extraídos de la SRAM. Finalmente se infirió una serie de compuertas XOR entre las salidas de ambas AND para obtener el valor correcto a escribir en memoria. La Figura 9 muestra un esquema de cómo quedaría la síntesis del VHDL

implementado.

FIGURA 9

Diagrama esquemático de la lógica implementada mediante VHDL para resolver las escrituras parciales en memoria.



Esta implementación requirió la inferencia de 64 flip-flops para registrar tanto la palabra que se extraía de la SRAM como la que ingresaba desde el bus del sistema. Este registro interno de la palabra a ser escrita en memoria permite además que aunque se encuentre en el ciclo más largo (el de escritura), se pueda indicar hacia el bus que el proceso está completo, mientras que la FSM continúa internamente con su ciclo de funcionamiento. Cumpliendo en todos los casos con el temporizado del ciclo de acceso a memoria.

4. Ensayos y capturas

Depuración mediante A.L. y Test Benchs

Para verificar el correcto funcionamiento del prototipo se instaló el mismo en uno de los slots correspondientes a las memorias del sistema y mediante una placa expansora se capturó el handshake de la misma mediante un Analizador Lógico. Como era de esperarse las primeras pruebas dieron resultados erróneos, pero mediante la comparación de las capturas sobre la memoria original con las capturas sobre el prototipo se encontraron los momentos en que el prototipo dejaba de responder como la memoria original. Posteriormente se utilizaron las capturas para depurar el diseño lógico mediante la generación de Tests Benchs que replicaran la señalización capturada en los momentos donde se encontraban las diferencias. Se identificaron en la simulación aquellos bloques responsables de las fallas y se realizaron las correcciones pertinentes.

Ciclo de lectura de memoria

Depurados los primeros errores, se pudo corroborar el correcto funcionamiento del prototipo. Se compararon los tiempos de lectura del prototipo con los del dispositivo original. Como pueden apreciarse en las capturas de pantalla del analizador lógico en las Figuras 10 y 11, estos fueron iguales (700 ns).

FIGURA 10

Captura, mediante analizador lógico, del handshake de la lectura de la memoria original.

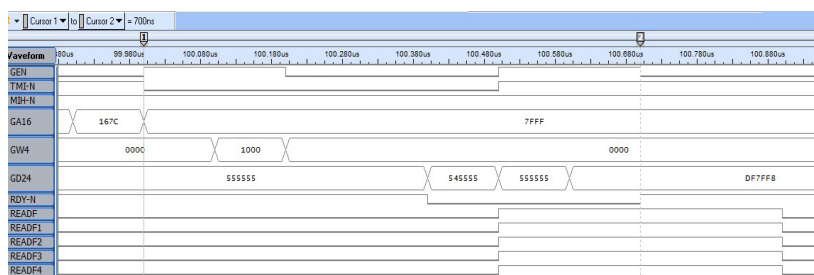
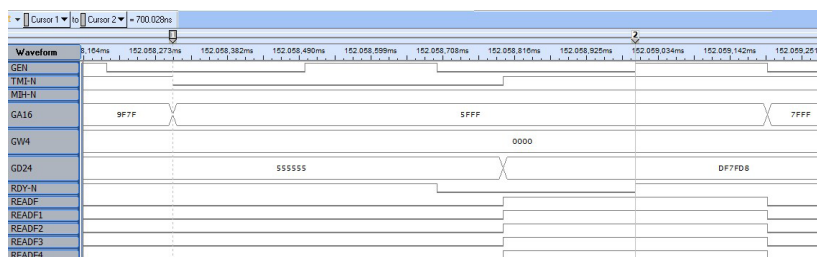


FIGURA 11

Captura, mediante analizador lógico, del handshake de la lectura del prototipo.



Se encontraron, además, diferencias significativas en los tiempos de establecimiento de los datos cuando el dispositivo escribe sobre el bus. En las Figuras 10 y 11, se puede observar que mientras la memoria original demora alrededor de 32 ns para establecer el dato correcto (xDF7FD8), el prototipo coloca el valor en el bus prácticamente de forma instantánea. Este resultado era esperable, debido a la marcada distancia entre las tecnologías intervinientes en cada placa.

Ciclo de escritura de memoria

En las capturas del test de prueba de las placas de memoria, también se pudo verificar el correcto temporizado de la

escritura, como se ve en las Figuras 12 y 13. Este mostró una diferencia de 60 ns en menos, en el prototipo, con respecto a la placa original. Las pruebas que se realizaron a posteriori sobre la computadora de propósito específico, donde se llegó a someter al prototipo hasta 48 horas de uso continuo, demostraron que esta diferencia en menos no afecta su normal desempeño. Tampoco pudo parametrizarse si hubo alguna mejora.

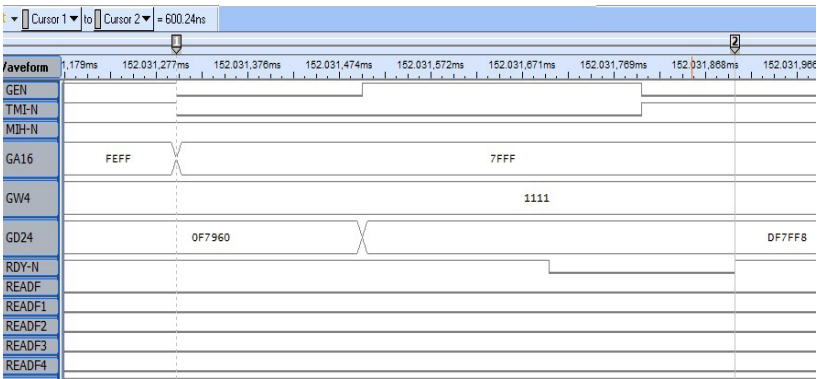
FIGURA 12

Captura, mediante analizador lógico, del handshake de la escritura de la memoria original.



FIGURA 13

Captura, mediante analizador lógico, del handshake de la escritura del prototipo.



5. Conclusiones

En esta clase de diseño, de componentes particulares con baja escala de integración, se trabaja muy cerca del hardware, con lo cual la información de las tablas de verdad, de las formas de onda y de las características dinámicas (todas presentes en las hojas de datos) de los diferentes circuitos integrados con los que interactuará la FPGA cobran significativa importancia, aún en la descripción en VHDL.

Asimismo, en el diseño lógico digital, el *test bench* resultó ser una herramienta de depuración fundamental. La generación del *test bench* a partir de las capturas realizadas sobre el dispositivo original, como sobre el prototipo bajo prueba, permitió al equipo de trabajo ir realizando modificaciones sobre el VHDL hasta obtener una versión completamente funcional.

Finalmente, se realizaron pruebas sobre la computadora de propósito específico que terminaron de validar el prototipo. Se sometió la nueva memoria RAM a usos continuos de hasta 48 horas y no presentó ninguna falla. Se procedió a realizar las pruebas y procedimientos para la homologación del prototipo, tarea que aún sigue en progreso.

A futuro, se prevé evaluar el desempeño de otros tipos de chequeo de paridad, aprovechando los 4 bits ociosos que quedan en el byte de paridad que se almacena en memoria junto con la palabra. El mecanismo de generación y chequeo de paridad que se implemente deberá contemplar que el pin PR indica un cambio de paridad de par a impar, y esto es utilizado en programas de prueba de funcionamiento de la memoria para detectar fallas.

Dado que finalmente no se utilizó el *clock* de 8 MHz provisto por el bus del sistema, se prevé desarrollar un prototipo que tenga como integrados de adaptación de nivel los dispositivos con E/S tolerantes a 5 V, para aprovechar la mayor capacidad de corriente que éstos manejan.

Se prevé, además, realizar un diseño con un integrado de memoria de mayor capacidad que permita emular, en una sola plaqueta, las cuatro plaquetas que originalmente posee el sistema. Esto sería posible porque los cuatro slots de las memorias poseen prácticamente el mismo cableado y la forma que utiliza el sistema para diferenciarlas es mediante un cambio en dos bits de la dirección.

Referencias bibliográficas

- Galasso Ch. L.; Friedrich G. R.; Burgos S. O.; Díaz G. J.; Antonini A. A. (2013). Reingeniería de las interfaces entre computadoras del sistema de comando y control. Revista Digital Del INUN N° 4. Año 2013. Dra. Lucía Alejandra Destro, Dra. Diana Fernández Calvo (Editores); pp. 59 – 98. ISSN 1853-4015. ISSN: 1852-7205. URL: http://www.ara.mil.ar/archivos/Docs/Galasso_corregido1.pdf
- Chu, Pong P. (2006). RTL hardware design using VHDL. Wiley-IEEE Press.
- “Spartan 3 Generation FPGA User Guide. Extended Spartan-3A, Spartan-3E and Spartan-3 FPGA families”. (2011). Manual de usuario UG331 versión 1.8 de la empresa Xilinx.
- Texas Instruments – Application Report SCEA035A (2004). Selecting the Right Level-Translation Solution.
- UG660 (v14.1) (2012). ISim User Guide – April 24, 2012.

Palabras clave: *Memoria RAM — sistema embebido — FPGA — reingeniería;*

Keywords: *RAM memory — embedded system — FPGA — re-engineering; Embedded System.*

Abstract

This work describes the implementation of a FPGA in the reengineering of a RAM memory of a specific purpose computer. It has a capacity of 16 K 28-bit words (24 data bits + 4 parity bits). Its access time is 750 ns. The original design is based on discrete TTL 5 [V] logic and 28 dynamic RAM integrated circuits that give the device total versatility in the way of writing memory.