Práctica 6: Memorias

Arnold T. Maldonado

*Instituto Politécnico Nacional, Unidad Profesional Interdisciplinaria de Ingeniería campus Zacatecas, Zacatecas, México*

tm02arnold@gmail.com

***Resumen*— En esta práctica el objetivo fue diseñar e implementar en VHDL una memoria RAM de 16 direcciones, una memoria ROM de 16 direcciones con un ancho de palabra de 8 bits y una memoria ROM de 8 direcciones con un ancho de palabra de 4 bits. Aquí se plasman lo conceptos básicos y razonamientos que hicieron posible completar esta tarea.**

# introducción

La memoria es la parte de un sistema que almacena datos binarios en grandes cantidades. Las memorias semiconductoras están formadas por matrices de elementos de almacenamiento que pueden ser latches o condensadores. Como regla general, las memorias almacenan datos en unidades que tienen de uno a ocho bits. La unidad menor de datos binarios es, como ya sabemos, el bit. En muchas aplicaciones, se tratan los datos en unidades de 8 bits, denominados bytes o en múltiplos de unidades de 8 bits. El byte se puede dividir en dos unidades de 4 bits, que reciben el nombre de nibbles.

Cada elemento de almacenamiento en una memoria puede almacenar un 1 o un 0 y se denomina celda. Las memorias están formadas por matrices de celdas, cada bloque de la matriz de memoria representa una celda de almacenamiento y su situación se puede especificar mediante una fila y

una columna. La capacidad de una memoria es el número total de unidades de datos que puede almacenar, la cual depende del número de celdas involucradas en la matriz de memoria.

Las dos principales categorías de memorias semiconductoras son las memorias RAM y ROM. La memoria RAM (Random Access Memory, memoria de acceso aleatorio) es un tipo de memoria en la que se tarda lo mismo en acceder a cualquier dirección de memoria y éstas se pueden seleccionar en cualquier orden, tanto en una operación de lectura como de escritura.

Todas las RAM poseen la capacidad de lectura y escritura. Debido a que las memorias RAM pierden los datos almacenados cuando se desconecta la alimentación, reciben el nombre de memorias volátiles.

La memoria ROM (Read Only Memory, memoria de sólo lectura) es un tipo de memoria en la que los datos se almacenan de forma permanente o semipermanente. Los datos se pueden leer de una ROM, pero no existe la operación de escritura como en las RAM. La ROM, al igual que la RAM, es

una memoria de acceso aleatorio, pero tradicionalmente el término RAM se reserva para las memorias de acceso aleatorio de lectura/escritura.

# desarrollo de la práctica

***a) Diseño de la RAM de 16 direcciones***

Primero se completó el código en VHDL que venía en el guión de la práctica (se muestra en el Apéndice A). Se completó lo siguiente:

1. A la entidad SRAM se le añadieron los valores:
   1. **w: integer:=4** para representar el ancho de palabra.
   2. **d: integer:=4** para representar el número de palabras.
   3. **a: integer:=2** para representar el ancho de dirección.
2. Se declaró la señal **tmp\_ram** como de tipo RAM.

Después se cambiaron algunos valores. El valor de **w** es 8 y el de **a** es 2^4 = 16 y el de **d** se mantuvo en 4. El código completo se muestra en el Apéndice B.

***b) Diseño de la ROM de 16 direcciones***

Al código de las dos primeras RAM se le modificó lo siguiente:

1. El valor de **a** ahora fue 4, para las d = 2^4 = 16 direcciones requeridas.
2. Se quitaron las entradas escribir y **data\_in**.
3. Se quitó el proceso escribir.
4. Se declaró la señal **tmp\_rom** y se le asignaron los valores que yo elegí (ver Tabla 1).
5. Se añadió el proceso reloj para que visualizar el contenido de la memoria automáticamente.

El código se muestra en el Apéndice C.

|  |  |
| --- | --- |
| *Dirección* | *En memoria* |
| 0000 | 10001000 |
| 0001 | 00010001 |
| 0010 | 11001100 |
| 0011 | 00110011 |
| 0100 | 10101010 |
| 0101 | 01010101 |
| 0110 | 01110111 |
| 0111 | 10001000 |
| 1000 | 00100010 |
| 1001 | 11001100 |
| 1010 | 01000100 |
| 1011 | 10001000 |
| 1100 | 00000000 |
| 1101 | 00100010 |
| 1110 | 10001000 |
| 1111 | 00000000 |

Tabla 1: Organización de la ROM de 16 direcciones

***c) Diseño de la ROM de 8 direcciones***

Los requerimientos de la memoria fueron:

1. ROM de 8 direcciones con un ancho de palabra de 4 bits.
2. Debía implementar un contador para acceder a las direcciones de memoria.
3. Una entrada será un botón que conectará a la tarjeta donde podrá activar o desactivar el contador
4. Una salida para indicar si el conteo es ascendente o descendente.
5. La frecuencia de reloj debía ser de 1⁄2 Hz. La señal de reloj de la memoria debía ser la misma del contador.

Se usó el código de la memoria ROM anterior, se le modificaron las siguientes cosas:

1. Se añadió el proceso **reloj** que establece la frecuencia del reloj especificada.
2. Se añadió la entrada **funciona**  para controlar el cuándo funciona del contador.
3. Se añadió la salida **X** para indicar el funcionamiento del contador

La ROM contiene los valores de las direcciones de la Tabla 2.

El código completo se muestra el código en el Apéndice D.

|  |  |
| --- | --- |
| *Dirección* | *En memoria* |
| 000 | 1000 |
| 001 | 0001 |
| 010 | 1100 |
| 011 | 0011 |
| 100 | 1010 |
| 101 | 0101 |
| 110 | 0111 |
| 111 | 1000 |

Tabla 2: Organización de la ROM de 8 direcciones

# Simulaciones

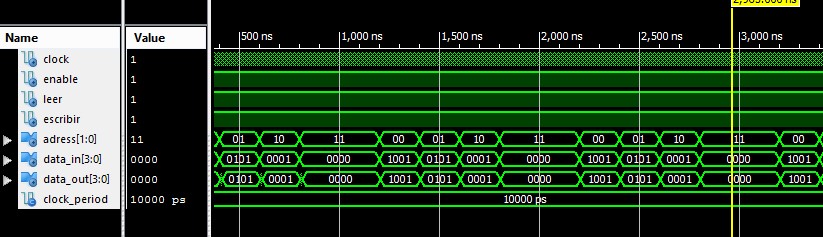


Figura 1: Simulación de la RAM de 4 direcciones

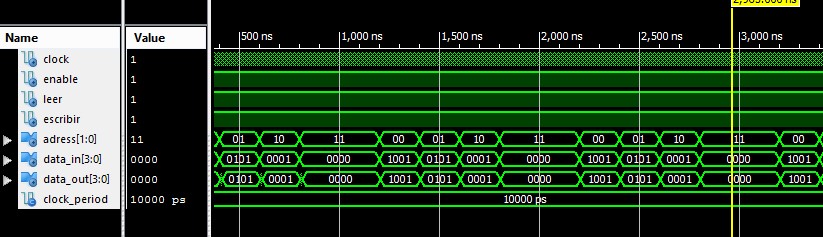
**

Figura 2: Simulación de la RAM de 16 direcciones

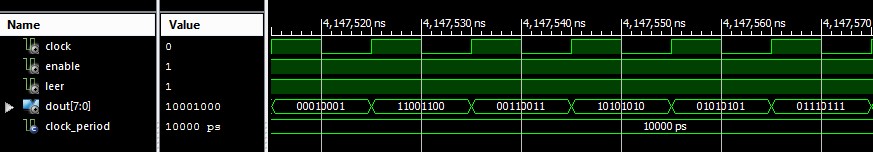
**

Figura 3: Simulación de la ROM de 16 direcciones con un ancho de palabra de 8 bits

# Implementaciones en el FPGA

Un circuito electrónico

Descripción generada automáticamente con confianza baja

Figura 4: RAM de 16 direcciones



Figura 5: ROM de 16 direcciones



Figura 6: ROM de 8x4 con el contador ascendente



Figura 7: ROM de 8x4 con el contador descendente

Conclusiones

Dado que las memorias RAM tienen tantos usos resultó importante para mí el conocer cómo funciona y cómo se organiza una, así como verla en acción usando los LED de FPGA.

Aquí se ven reflejados la gran mayoría los conceptos que vimos en el curso y me agradó poder diseñar las memorias usándolos.

Apéndice A

código de la RAM de 4 direcciones

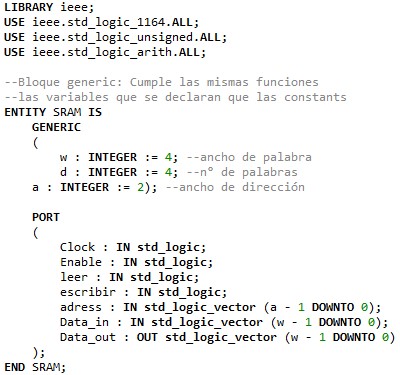


Figura 8: RAM con 4 direcciones (1/2)

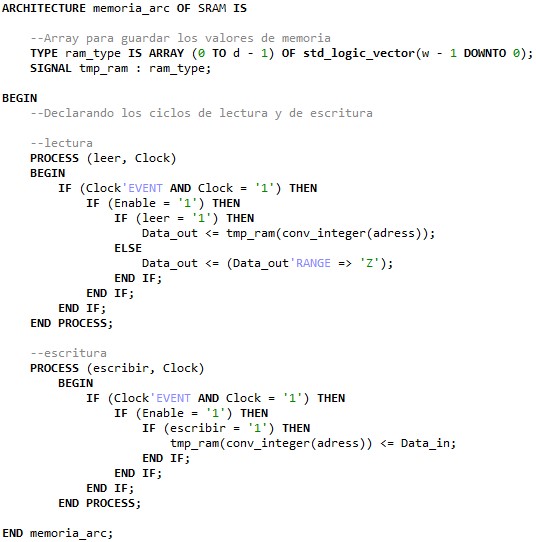


Figura 9: RAM con 4 direcciones (2/2)

Apéndice B

código de la RAM de 16 direcciones

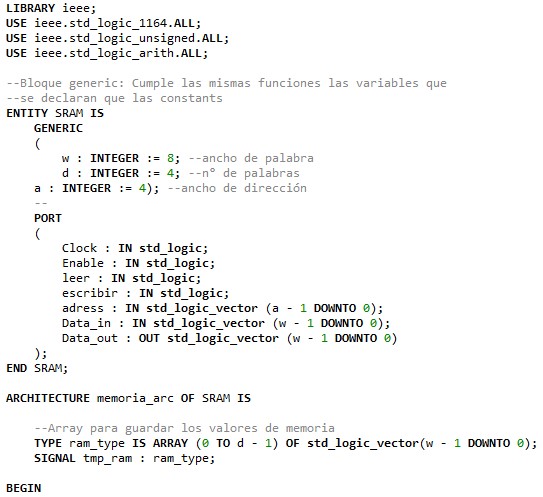


Figura 10: RAM con 16 direcciones (1/2)

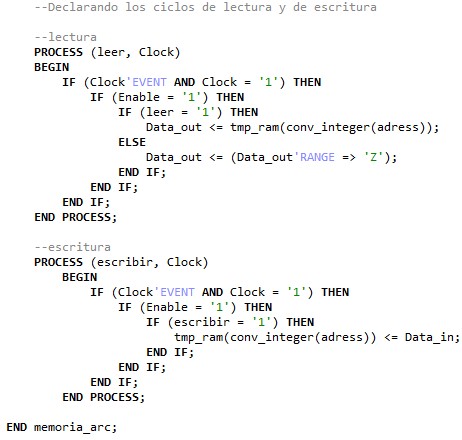


Figura 11: RAM con 16 direcciones (2/2)

Apéndice C

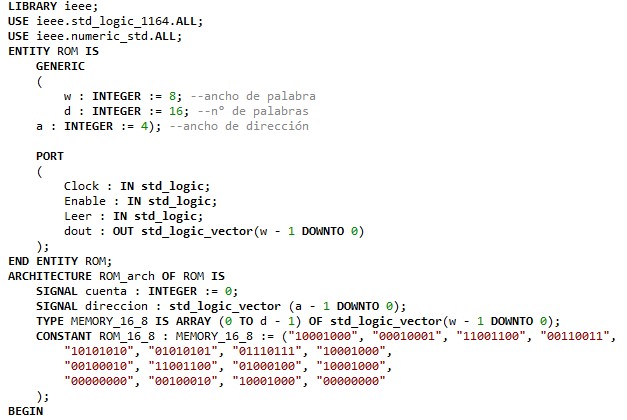
código de la ROM de 16 direcciones

Figura 12: ROM con 16 direcciones (1/4)

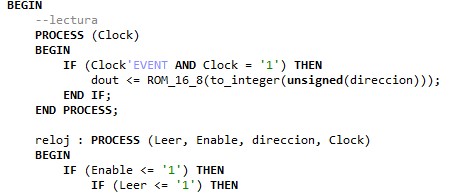


Figura 13: ROM con 16 direcciones (2/4)

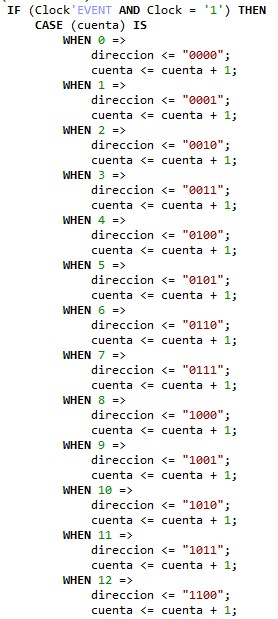


Figura 14: ROM con 16 direcciones (3/4)

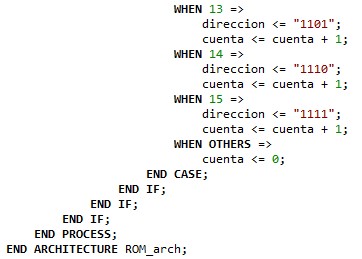


Figura 15: ROM con 16 direcciones (4/4)

Apéndice D

Código de la ROM de 8 direcciones

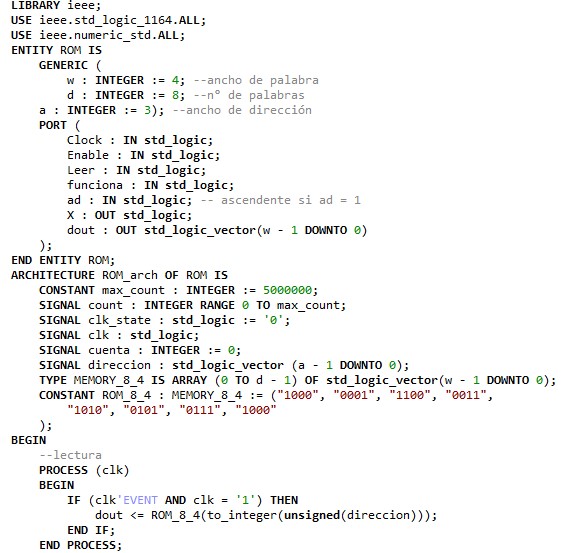


Figura 16: ROM con 8 direcciones (1/5)

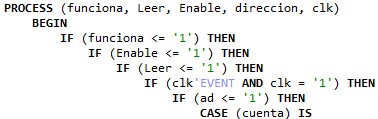


Figura 17: ROM con 8 direcciones (2/5)

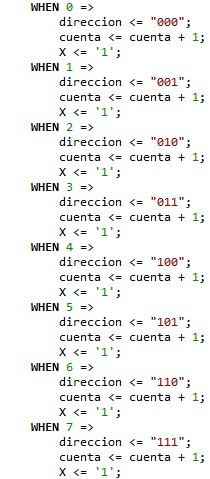


Figura 18: ROM con 8 direcciones (3/5)

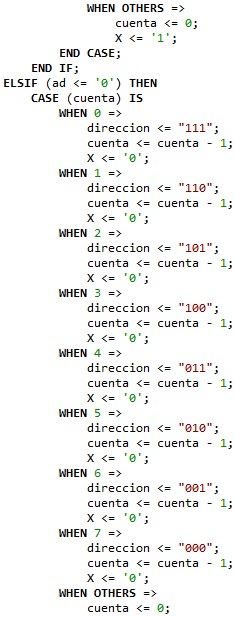


Figura 19: ROM con 8 direcciones (4/5)

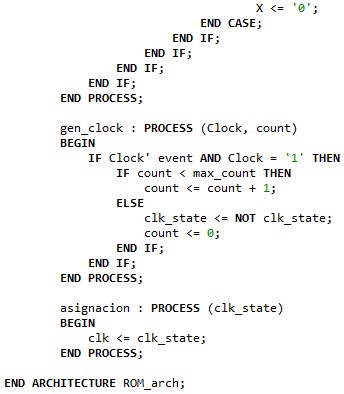


Figura 20: ROM con 8 direcciones (5/5)

referencias

1. M. M. Morris, *Diseño Digital* (3ª Ed.). Naucalpan de Juárez: Pearson Educación, 2003.
2. T. L Floyd, *Fundamentos de Sistemas* Digitales (9ª Ed.). Pearson Educación, 2007.
3. D. D. Gajski y P. G. Carlos, *Principios de Diseño Digital*. Prentice Hall, 2000.