Memorias

Autor: Díaz Gerardo Agustín

Carrera: Ingeniería Electrónica

Materia: Electrónica II

Año: 2025

Resumen

En esta actividad se estudiaron y aplicaron los conceptos fundamentales de las memorias semiconductoras en sistemas digitales, diferenciando los tipos volátiles (SRAM y DRAM) y no volátiles (ROM, PROM, EEPROM y FLASH).

Asimismo, se implementaron memorias sintetizables en VHDL, utilizando los bloques de memoria interna (EBR) de la FPGA iCE40, para almacenar y recuperar datos de manera controlada.

Las simulaciones y pruebas demostraron el correcto funcionamiento de las operaciones de lectura y escritura, verificando la correspondencia entre los valores programados y los datos obtenidos en salida.

Este trabajo permitió comprender la importancia de las memorias en el diseño digital y su papel esencial como elementos de almacenamiento, retención y transferencia de información en dispositivos programables.

Introducción

Desarrollo Teorico

Las memorias semiconductoras son dispositivos diseñados para almacenar información binaria de manera temporal o permanente dentro de un sistema digital. Cada celda de memoria puede guardar un bit, y su acceso se realiza mediante direcciones únicas dentro de un arreglo organizado en filas y columnas (Floyd, 2015).

Existen dos grandes familias: memorias no volátiles, que conservan los datos sin alimentación eléctrica, y memorias volátiles, que pierden la información al apagarse el sistema.

1. **Memorias no volátiles (ROM, PROM, EEPROM, FLASH)**

- Las **ROM (Read Only Memory)** son memorias de solo lectura que contienen información fija grabada durante la fabricación del chip. Se utilizan principalmente para almacenar programas de inicio o firmware, como el BIOS en computadoras personales (Floyd, 2015).

- La **PROM (Programmable ROM)** permite al usuario grabar datos una única vez mediante un proceso de programación eléctrica que quema internamente fusibles, definiendo así de forma permanente los “1” y “0” en las celdas de memoria (Floyd, 2015).

- Las **EEPROM (Electrically Erasable Programmable ROM)** son memorias de solo lectura que pueden borrarse y reprogramarse eléctricamente, sin necesidad de retirar el chip del circuito. Cada celda de memoria puede modificarse individualmente aplicando impulsos eléctricos, lo que permite actualizar datos o configuraciones de forma flexible (Floyd, 2015).

- La **FLASH** es una variante moderna de la EEPROM con arquitectura de bloque, lo que permite borrar y reescribir grandes sectores de memoria simultáneamente, reduciendo costos y consumo. Por su alta densidad y velocidad, es ampliamente usada en dispositivos portátiles, memorias USB y sistemas embebidos (Floyd, 2015).

2. **Memorias volátiles (SRAM y DRAM)**

- La **SRAM (Static Random-Access Memory)** utiliza flip-flops como elementos de almacenamiento para mantener cada bit de información mientras haya energía eléctrica.
  Este tipo de memoria no necesita ciclos de refresco, por lo que ofrece alta velocidad de acceso y un diseño más simple, aunque con menor densidad y mayor costo por bit.
  Floyd la describe como ideal para implementar memorias caché y registros internos en los procesadores debido a su rapidez y estabilidad.

- La **DRAM (Dynamic Random-Access Memory)** almacena cada bit en un pequeño condensador, cuya carga eléctrica representa un "1" lógico.
  Dado que estos condensadores se descargan con el tiempo, la DRAM necesita ser refrescada constantemente mediante circuitos de control.
  A cambio, su estructura más simple permite una mayor densidad de integración y menor costo, siendo utilizada como memoria principal en la mayoría de los sistemas de cómputo (Floyd, 2015).

Aplicaciones de los arreglos de memoria en los sistemas de cómputo

En un sistema digital moderno, los arreglos de memoria se organizan en una jerarquía que equilibra latencia, ancho de banda, capacidad y costo por bit. Desde los conjuntos de registros (los más rápidos y pequeños) hasta el almacenamiento masivo (el más lento y económico), cada nivel cumple un rol específico y se apoya en el principio de localidad (temporal y espacial) para acelerar el acceso a datos e instrucciones (Harris & Harris, 2019).

1) **Conjuntos de registros (Register Files)**

Un registro es una pequeña celda de almacenamiento dentro del procesador; un conjunto de registros es un arreglo multi-entrada/multi-salida que permite varias lecturas y/o escrituras por ciclo. En CPU tipo RISC, es habitual disponer de 2 puertos de lectura y 1 de escritura para sostener el caudal de una instrucción por ciclo en el datapath. Su función es mantener operandos inmediatos para la ALU, direcciones efectivas y resultados intermedios sin salir del núcleo. (Harris & Harris, 2019).

2) **Memoria caché**

La caché es una SRAM de alta velocidad situada entre la CPU y la memoria principal. Explota la localidad temporal y espacial almacenando líneas para que accesos futuros sean de baja latencia (Harris & Harris, 2019).

3) **Memoria principal (DRAM)**

La memoria principal usa DRAM, cuya celda es un condensador que requiere refresco periódico. A cambio, ofrece alta densidad y buen costo por bit, por lo que aloja programas y datos activos (Floyd, 2015).

4) **Memoria de almacenamiento**

El almacenamiento no volátil conserva los datos sin energía: FLASH, SSD y HDD son los más comunes.

FLASH/SSD: basados en celdas NAND, con borrado por bloques y gestión de desgaste. Aportan baja latencia comparados con HDD y altas operaciones de entrada/salida por segundo; ideales para arranque del sistema, bases de datos y contenido estático (Floyd, 2015).

Memorias en FPGA – Familia iCE40

Las FPGA iCE40 de Lattice Semiconductor incluyen bloques de memoria integrados llamados EBR (Embedded Block RAM), que permiten implementar memorias ROM y RAM sintetizables directamente en el dispositivo.

Cada bloque EBR puede configurarse en distintas organizaciones, como:

256×16, 512×8, 1024×4 o 2048×2, y soportan modos de lectura y escritura dual, máscaras de escritura por byte y preinicialización desde archivos de datos (Lattice Semiconductor, 2020. Memory Usage Guide for iCE40 Devices (FPGA-TN-02002-1.7). Lattice Semiconductor Corporation.).

Esto facilita la creación de memorias como ROM de 512×32 bit o RAM de dos puertos para sistemas embebidos y controladores implementados en FPGA.

Especificación de la Memoria RAM 16×4

La memoria RAM 16×4 desarrollada corresponde a un arreglo de almacenamiento de 16 palabras de 4 bits, implementada en VHDL y sintetizable en la FPGA EDU-CIAA. Se trata de una memoria síncrona de un único puerto, que permite realizar operaciones de lectura y escritura en una misma dirección, controladas por una señal de reloj.

A continuación, se describen sus principales características funcionales y estructurales:

• Capacidad y organización:
La memoria está compuesta por 16 posiciones de almacenamiento, cada una de 4 bits, lo que proporciona una capacidad total de 64 bits. El bus de direcciones posee 4 bits (addr(3 downto 0)), permitiendo direccionar las posiciones desde 0 hasta 15.

• Interfaz del módulo:
El diseño incluye las siguientes señales:

- clk: señal de reloj. Todas las operaciones se sincronizan en el flanco ascendente.
- we: señal de habilitación de escritura (*write enable*).
- addr: dirección de la palabra a leer o escribir.
- din: dato de entrada (4 bits) para escritura.
- dout: dato de salida (4 bits) correspondiente al contenido almacenado en la dirección seleccionada.

• Comportamiento de escritura:
La operación de escritura es síncrona. Cuando we = '1', en el flanco ascendente de clk el dato presentado en din se almacena en la dirección indicada por addr. Si we = '0', no ocurre ninguna modificación en la memoria.

• Comportamiento de lectura:
La lectura también es síncrona y registrada. En cada flanco ascendente del reloj, el valor almacenado en la dirección seleccionada se carga en la salida dout. Esto implica que cualquier cambio en la dirección addr se verá reflejado en la salida un ciclo de reloj después, lo cual coincide con el comportamiento típico de las memorias síncronas utilizadas en FPGA.

• Inicialización desde archivo externo:
El módulo incorpora un mecanismo de inicialización mediante el genérico init_file, que permite cargar el contenido de la memoria desde un archivo de texto ASCII. Cada línea del archivo contiene un valor hexadecimal de 4 bits, el cual se asigna secuencialmente a cada dirección de memoria. En caso de no especificarse un archivo, la memoria se inicializa automáticamente en cero.

• Aplicación en la plataforma EDU-CIAA-FPGA:
Para la etapa de demostración práctica indicada en la actividad, la memoria se conectó a:

- Interruptores físicos de la placa para ingresar direcciones y datos.
- Un pulsador para controlar we.
- Un display de 8 segmentos para visualizar el dato contenido en dout.

Materiales y Métodos

Análisis preliminar y definición de la arquitectura del arreglo de memoria

El primer paso consiste en determinar el tipo de memoria, sus dimensiones y comportamiento temporal.

Siguiendo a Harris & Harris, una memoria digital se organiza como un conjunto de palabras direccionables, donde cada acceso puede involucrar lectura, escritura o ambas según los puertos disponibles. Esta estructura se formalizó mediante:

- Profundidad (cantidad de palabras): 16, 32, 512 según el módulo.
- Ancho de palabra: 4, 32 bits según la consigna.
- Cantidad de puertos:
  1. Un puerto de lectura/escritura para RAM simples.
  2. Dos puertos de lectura + uno de escritura para el register file 32×32

Se estableció además que todas las memorias debían comportarse como dispositivos síncronos, de modo que lectura y/o escritura ocurran en el flanco ascendente del reloj, tal como recomiendan los textos de diseño digital para facilitar la síntesis y preservar tiempos de acceso deterministas.

Modelado del arreglo de memoria en VHDL

Siguiendo el Dally en el capítulo de memoria y estructuras de datos sintéticas en VHDL, el arreglo interno de memoria se modeló como un array de std_logic_vector, por ejemplo:

type ram_type is array (0 to N-1) of std_logic_vector(W-1 downto 0);

Esto permite al sintetizador inferir automáticamente recursos de memoria dedicados como BRAM o LUT-RAM, un proceso consistente con lo descrito por Lattice en la *Memory Usage Guide for iCE40 Devices*.

Cada memoria se implementó como:

- **RAM simple**: un puerto con señales addr, din, dout, we, clk.

- **ROM**: misma estructura pero sin señal de escritura.
- **Registro**: dos caminos combinacionales de lectura y un camino síncrono de escritura, siguiendo el patrón para unidades de registros de CPU.

Descripción del comportamiento síncrono

El comportamiento básico de la memoria se describe empleando un proceso secuencial controlado por flanco ascendente:

```
-------------------------------------------------------
process(clk)
begin
  if rising_edge(clk) then
    if we = '1' then
      mem(addr) <= din;
    end if;
    dout <= mem(addr);-- lectura síncrona
  end if;
end process;
-------------------------------------------------------
```

Este patrón, recomendado por Dally como forma estándar de modelar memorias en VHDL, garantiza que:

1. La lectura es registrada, lo que facilita el cumplimiento de tiempos en FPGA.
2. La escritura se realiza de manera determinista en el mismo flanco de reloj.
3. El sintetizador infiere bloques de memoria reales, tal como describe Lattice en la guía técnica para dispositivos iCE40.

Inicialización desde archivo externo

Como exige la consigna, se incorporó un mecanismo de inicialización mediante archivos ASCII con valores hexadecimales. Para ello se utilizó la librería textio y la función hread, siguiendo ejemplos de memoria propuestos en Dally:

```
-------------------------------------------------------------
impure function init_ram return ram_type is
  file ram_file : text;
  ...
  hread(line_content, mem(addr_index), valid);
-------------------------------------------------------------
```

La función se ejecuta al inicio de la simulación o síntesis, permitiendo cargar contenido predeterminado sin modificar el código VHDL. Esto es especialmente útil en memorias ROM y bancos de pruebas automáticos.

Verificación mediante simulación

Para cada módulo se desarrolló un testbench que verifica:

- Escrituras correctas cuando we='1'.
- Lecturas correctas con latencia de un ciclo.
- Comportamiento en acceso simultaneo a la misma dirección.
- Inicialización correcta desde archivos externos.

- En el caso del registro, correcto funcionamiento de los puertos dobles de lectura y escritura.

Resultados y Discusión

Se diseñaron tres módulos de memoria principales (rom_512x32, ram_512x32_dp, reg_32x32) y un módulo adicional (ram_16x4) para implementar en la FPGA. Cada uno de estos módulos fue validado funcionalmente mediante un banco de prueba dedicado, confirmando que su comportamiento se ajusta a las especificaciones de la actividad.

Módulos de Memoria Descritos

Se generaron cuatro archivos VHDL para describir los componentes de hardware:

- **rom_512x32.vhd**: Describe una memoria ROM (Read-Only Memory) de 512 palabras, donde cada palabra es de 32 bits. La memoria es síncrona; la salida dout se actualiza únicamente en el flanco de subida del reloj (clk). El contenido de la memoria se inicializa al comienzo de la simulación mediante la función init_rom, que lee los datos desde un archivo de texto externo (init_file) .
- **ram_512x32_dp.vhd**: Implementa una memoria RAM (Random Access Memory) de 512x32 bits con doble puerto simple. Posee un puerto de escritura síncrono (controlado por clk1) y un puerto de lectura síncrono (controlado por clk2). El puerto de escritura incluye una máscara (mask) que permite la escritura selectiva por byte (8 bits), habilitando la modificación de partes de la palabra de 32 bits.
- **reg_32x32.vhd**: Describe un conjunto de registros de 32x32 bits. Este módulo cumple con el requisito de tres puertos: un puerto de escritura síncrono (controlado por clk y we) y dos puertos de lectura independientes (addr1/dout1, addr2/dout2). A diferencia de la RAM, las lecturas son asíncronas, lo que significa que dout1 y dout2 se actualizan instantáneamente cuando sus respectivas direcciones de entrada cambian.
- **ram_16x4.vhd**: Contiene la descripción de una RAM de 16x4 bits de un solo puerto, utilizada para el demostrador gráfico de la Figura 1. Es un módulo síncrono (clk) que utiliza la misma dirección (addr) tanto para la lectura como para la escritura, controlada por la señal we.

Bancos de Prueba y Verificación

Para cada módulo de memoria principal, se desarrolló un banco de prueba (testbench) que verifica su correcta operación mediante una serie de aserciones (assert).

- **rom_512x32_tb.vhd**: Verifica la ROM. Su proceso de estímulo comprueba que los datos cargados desde el archivo de inicialización sean correctos. Realiza lecturas en direcciones específicas (ej. 0 y 3) y verifica que los datos de salida coincidan con los valores esperados. También comprueba que las direcciones no inicializadas devuelvan el valor por defecto (0).
- **ram_512x32_dp_tb.vhd**: Es el testbench para la RAM de doble puerto. Este banco de pruebas verifica secuencialmente:
    1. **Prueba 1:** La correcta lectura del valor inicial cargado desde el archivo (.txt).

2. **Prueba 2:** La operación simultánea, asegurando que se puede leer una dirección (addr 8) sin interferencias mientras se escribe en otra (addr 9).

3. **Prueba 3:** La verificación de la escritura realizada en la prueba anterior.

4. **Pruebas 4 y 5:** La funcionalidad de la máscara de escritura. Se escribe un valor solo en los bytes superiores (mask = "1100") y se verifica que el resultado final en la memoria sea la combinación correcta del valor nuevo y el valor antiguo.

- **reg_32x32_tb.vhd**: Valida el conjunto de registros. Sus pruebas están diseñadas para verificar las características únicas de este componente:

    1. **Prueba 1:** Lectura asíncrona de los puertos, verificando que los datos de salida aparecen sin esperar un ciclo de reloj.

    2. **Pruebas 2 y 3:** Verifican la operación de escritura y su posterior lectura.

    3. **Prueba 4:** Asegura que la señal we deshabilita correctamente la escritura, impidiendo cambios en el registro.

    4. **Prueba 5:** Valida la operación de los 3 puertos a la vez, escribiendo en un registro mientras se leen otros dos simultáneamente.

Conclusiones

La descripción de hardware es el proceso de definir el comportamiento y la estructura de un circuito digital mediante un lenguaje como VHDL. No se trata de programar instrucciones secuenciales, sino de especificar cómo se conectan y funcionan componentes reales como registros, memorias o puertas lógicas.

Referencias

- Harris, S. L., & Harris, D. (2019). Digital design and computer architecture: RISC-V edition. Morgan Kaufmann.

- Floyd, T. L. (2015). Digital Fundamentals (11th ed.). Pearson Education.

- Dally, W., Harting, R., Aamodt, T. Digital Design Using VHDL (Cambridge, 2016).