El código VHDL usado para ilustrar las explicaciones teóricas, así como los diseños y bancos de prueba propuestos como solución de los ejercicios de autocomprobación, están disponibles en la URL siguiente: **https://canal.uned.es/series/64149eb43056d535ce6fe473**

ÍNDICE

Tema 1

FUNDAMENTOS DEL DISEÑO DEL HARDWARE DIGITAL

1.1. INTRODUCCIÓN

Los sistemas digitales se han ido haciendo más y más complejos durante las pasadas décadas. Este incremento en la complejidad responde, a grandes rasgos, a la Ley de Moore, según la cual el avance tecnológico posibilita que *cada aproximadamente 18 meses se doble el número de transistores que es posible alojar en un circuito integrado.*

De esta forma, en la década de 1970 un circuito integrado típico contenía decenas de miles de transistores. En la década de 1980, la capacidad aumentó a cientos de miles de transistores, y en la década de 1990 fue del orden de decenas de millones. En la década de 2000, la capacidad de los circuitos integrados es del orden de miles de millones de transistores.

Cuando se fabricaba un circuito integrado en la década de 1970, se documentaba su funcionamiento empleando una combinación de esquemáticos (representación gráfica de los componentes del circuito), diagramas de transición de estados y lenguaje natural (por ejemplo, inglés). Esta documentación podía consistir en varios cientos de páginas. Los ingenieros, que compraban el circuito integrado para usarlo en sus propios diseños, tenían que leer esta documentación para entender el funcionamiento del circuito integrado. Como puede imaginarse, leer cientos de páginas no era tarea fácil. Además, en ocasiones la documentación contenía errores o ambigüedades. La consecuencia de ello era que frecuentemente los ingenieros tenían problemas para emplear los circuitos integrados en el desarrollo de sus propios sistemas.

Debido a esta situación, el Departamento de Defensa de EE.UU. buscó un procedimiento mediante el cual los fabricantes de circuitos integrados pudieran especificar de forma precisa el funcionamiento de los circuitos. Con esta motivación, el Departamento de Defensa de EE.UU. inició el desarrollo de un *lenguaje para la descripción del hardware*, para lo cual estableció un grupo de trabajo compuesto por expertos de varias disciplinas, pertenecientes a diferentes compañías.

Un *lenguaje para la descripción del hardware* o HDL (siglas que provienen del inglés: *Hardware Description Language*) es un lenguaje, legible tanto por las máquinas como por los seres humanos, ideado para describir tanto el comportamiento como la estructura del hardware. El HDL permite describir de forma precisa y rigurosa el funcionamiento del circuito digital, el cual puede ser simulado en un ordenador con el fin de reproducir exactamente el funcionamiento del circuito. La simulación por ordenador permite obtener el valor de las señales de salida del circuito para una determinada secuencia de señales de entrada.

El HDL que el Departamento de Defensa de EE.UU. creó en los años 80 se llamó VHDL. Las siglas VHDL provienen de VHSIC Hardware Description Language.

VHSIC es un acrónimo de Very High Speed Integrated Circuit, que fue el nombre del proyecto llevado a cabo por el Departamento de Defensa de EE.UU.

La sintaxis de VHDL es muy similar a la del lenguaje de programación ADA. En 1987, el Institute of Electrical and Electronics Engineers (IEEE) adoptó VHDL como el estándar número 1076. El establecimiento de un estándar del lenguaje tiene una ventaja fundamental: las compañías desarrolladoras de software de simulación tienen una definición claramente establecida del lenguaje al que deben dar soporte.

1.2. LENGUAJES PARA LA DESCRIPCIÓN DE HARDWARE

En la actualidad, la casi totalidad de los diseñadores de circuitos digitales de cierta complejidad usan para realizar sus diseños *lenguajes para la descripción del hardware*. El empleo de HDL presenta ventajas respecto al empleo de descripciones basadas en esquemáticos. Algunas de ellas son las siguientes:

1. Puesto que una descripción HDL es simplemente un fichero de texto, es mucho más portable que un diseño esquemático, que debe ser visualizado y editado empleando la herramienta gráfica específica del entorno de CAD (*Computer-Aided Design* - Diseño asistido por ordenador) con el que se ha creado.

2. Una descripción esquemática únicamente describe el diseño de manera estructural, mostrando los módulos y la conexión entre ellos. Por el contrario, la descripción del circuito usando un HDL puede realizarse bien mostrando la estructura, o bien describiendo el comportamiento. Es decir, los HDL permiten describir el comportamiento que se desea que tenga el circuito, sin hacer ninguna referencia a su estructura. Las *herramientas de síntesis* permiten generar automáticamente la estructura del circuito lógico a partir de la descripción de su comportamiento.

3. El mismo HDL que se ha usado para la descripción del circuito, puede emplearse para describir los *vectores de test* y los resultados esperados del test. Los *vectores de test* son los valores de las señales que se aplicarán a los pines de entrada del circuito con la finalidad de comprobar si el funcionamiento del circuito es correcto. Así pues, pueden realizarse los *programas de test* (vectores de test e instantes en los cuales son aplicados) del circuito a medida que se diseña el propio circuito, pudiéndose con ello ir realizando diferentes pruebas a medida que se avanza en el diseño. Como ventajas añadidas, la descripción de los programas de test usando HDL es altamente portable y repetible.

1.2.1. Usos de un programa HDL

Resumiendo lo anteriormente explicado, los programas escritos en HDL pueden tener las tres aplicaciones siguientes:

- *Documentación formal.* Un programa HDL puede ser usado como una especificación formal de un sistema digital. Se trata de un tipo de documentación explícita y precisa acerca del sistema, que es intercambiable entre diferentes diseñadores y usuarios.
- *Entrada a un simulador.* La simulación por ordenador del circuito permite estudiar y comprobar su operación, sin necesidad de tener que construir físicamente el circuito. La entrada al simulador es el programa HDL que describe el circuito, así como la descripción, también en el HDL, de los vectores de test que deben aplicarse al circuito. Durante la ejecución de la simulación, el simulador interpreta el código HDL y genera las respuestas del circuito.
- *Entrada a una herramienta de síntesis.* El flujo de diseño del hardware digital se basa en un proceso de refinamiento, que convierte gradualmente una descripción de alto nivel del sistema a una descripción estructural de bajo nivel. Algunos de estos pasos de refinamiento pueden ser realizados por las herramientas software de síntesis.

  Las herramientas software de síntesis aceptan como entrada la descripción HDL del circuito y componen el correspondiente circuito empleando para ello los componentes proporcionados en una librería. La salida de la herramienta de síntesis es un nuevo programa HDL con la descripción estructural del circuito sintetizado.

1.2.2. HDL más ampliamente usados

En la actualidad, los HDL más ampliamente usados son Verilog HDL y VHDL. Ambos son lenguajes estándar de IEEE para el modelado y simulación de hardware.

- **Verilog** se creó, a principios de los años 80, como un lenguaje propiedad de la compañía Philip Moorby, compañía que años más tarde fue adquirida por Cadence Design Systems. Posteriormente, Verilog se hizo de dominio público y se promovió como un estándar de IEEE en el año 1995, denominado IEEE 1364.
- Como se ha explicado anteriormente, **VHDL** fue desarrollado en 1983 por el Departamento de Defensa de los EE.UU. con la finalidad de servir como

lenguaje estándar para la descripción de hardware. En el año 1987 se convirtió en un estándar de IEEE (IEEE 1067-1987).

Posteriormente se han ido incorporando mejoras en el lenguaje, pero la mayor parte de las nuevas funcionalidades y cambios no son sintetizables. Suelen consistir en proporcionar nuevas facilidades para el diseño de los bancos de pruebas. Cabe destacar las actualizaciones del estándar llevadas a cabo en los años 1993 (IEEE 1076-1993) y 2001 (IEEE 1076-2001). La última versión del lenguaje en el momento de escritura de este texto es IEEE 1076-2019.

A principios del año 2000 se desarrolló otro HDL denominado **SystemC**, el cual consiste en un conjunto de librerías en C++. SystemC se convirtió en el estándar 1666 de IEEE en el año 2005.

1.3. CICLO DE DISEÑO DE LOS CIRCUITOS DIGITALES

El empleo de HDL es práctica habitual en las diferentes fases del ciclo de diseño de circuitos digitales. En la Figura 1.1 se muestran las actividades que típicamente se realizan durante el ciclo de diseño e implementación de los circuitos digitales.

En primer lugar, el diseñador debe establecer la especificación del diseño. Dicha especificación consiste en establecer *qué* se espera que haga el circuito y qué *restricciones* deben satisfacerse (frecuencia de reloj, retardos, tamaño, etc.).

A continuación, el diseñador debe crear un diseño de alto nivel del circuito, para lo cual puede emplear un lenguaje para la descripción de hardware (HDL), por ejemplo VHDL o Verilog. Seguidamente, debe desarrollar un conjunto de programas de test (usando también VHDL o Verilog) y, si es posible, usar estos programas para testear el diseño de alto nivel, usando para ello una herramienta de simulación (*verificación funcional*). En función de los resultados de la simulación de los tests, puede ser preciso modificar el diseño de alto nivel, repitiéndose los pasos anteriores tantas veces como sea preciso.

Una vez el diseño de alto nivel funciona adecuadamente, debe traducirse al nivel de puertas lógicas o de transistores. Este proceso se denomina *síntesis*. Síntesis es la generación automática del diseño del circuito a partir de la descripción de su comportamiento. El resultado obtenido de la síntesis, denominado *netlist*, es una descripción de todas las conexiones y componentes que deben componer el circuito.

La descripción al nivel de puertas o transistores, obtenida a partir de una descripción en HDL, depende de la forma en que se ha programado el modelo en HDL y de la herramienta de síntesis empleada. Las herramientas de síntesis proporcionan numerosas opciones que permiten al diseñador especificar cómo debe realizarse la

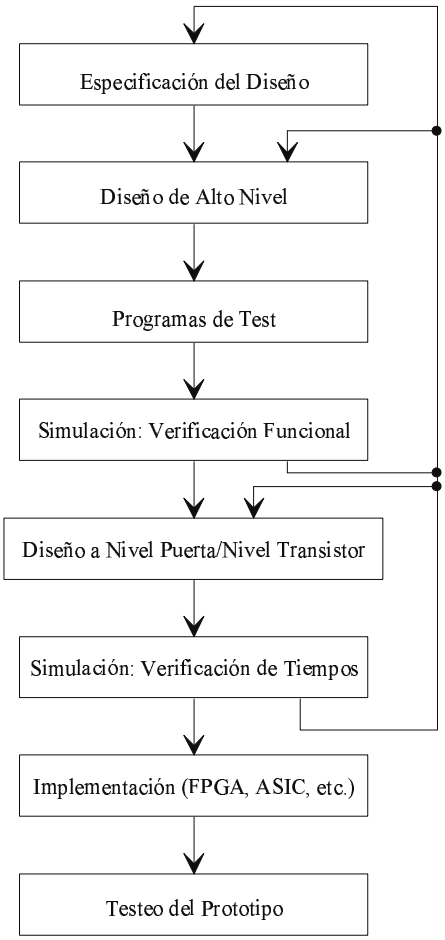

Figura 1.1: Ciclo de diseño del hardware digital.

síntesis. En particular, permiten especificar el nivel de esfuerzo a emplear por la herramienta en la optimización automática del circuito, tanto en lo que respecta a la reducción del área del circuito como en lo que respecta a sus prestaciones. Asimismo, las herramientas de síntesis permiten especificar qué módulos del circuito no deben ser optimizados.

El diseño a nivel de puertas o transistores debe ser vuelto a testear mediante simulación (*verificación de tiempos*), usando, si es posible, el mismo conjunto de tests que se realizaron sobre el diseño de alto nivel. El objetivo es estudiar si el diseño se comporta como debe y si satisface todas las restricciones que se impusieron en la fase de especificación. Si se detecta un problema a este nivel, debe volverse al correspondiente paso del ciclo de diseño.

Una vez el diseño ha superado estos tests, puede implementarse en la plataforma hardware seleccionada: PLD (*Programmable Logic Device*), FPGA (*Field-Programmable Gate Array*), ASIC (*Application-Specific Integrated Circuit*), etc. Se

emplean herramientas software para fabricar (en el caso de los ASIC) o programar (en el caso de los FPGA) el circuito integrado a partir de la *netlist*.

Una vez implementado, el circuito integrado puede ser testeado con ayuda de un generador de patrones (para generar los vectores de test), y un analizador lógico u osciloscopio (para medir las salidas).

En este texto únicamente se explican las primeras fases del ciclo de diseño, en las cuales se establece la especificación funcional del circuito, se realiza el diseño de alto nivel del circuito, se elabora el programa de test, y se simula el banco de pruebas con el fin de realizar una verificación funcional del circuito. La especificación de las restricciones que debe satisfacer el circuito, así como las fases de síntesis, verificación de tiempos, implementación y testeo del prototipo, quedan fuera del alcance de este texto.

1.4. TECNOLOGÍAS DE CIRCUITOS INTEGRADOS

La implementación hardware del circuito digital puede realizarse empleando circuitos integrados de diferentes *tecnologías*. Una característica fundamental que diferencia cada tecnología de circuitos integrados es la manera de adaptar el circuito a una determinada aplicación.

En algunas tecnologías, la estructura del circuito integrado está completamente predefinida y el usuario debe particularizarlo para su aplicación "programándolo". Es decir, definiendo un patrón de conexiones en la estructura del circuito integrado, bien descargando dicha estructura en la memoria interna del circuito integrado o mediante el fundido selectivo de los fusibles internos del circuito integrado.

En el otro extremo, las tecnologías ASIC (*Application Specific Integrated Circuit*) requieren del diseño de *máscaras de fotolitografía* propias para cada uno de los pasos del proceso de fabricación del circuito integrado.

Así pues, las tecnologías non-ASIC permiten particularizar el circuito en campo, mientras que los circuitos de las tecnologías ASIC deben ser particularizados a su aplicación en la fábrica de circuitos integrados.

1.4.1. Clasificación de las tecnologías

Es posible clasificar las tecnologías de circuitos integrados digitales de la manera mostrada a continuación. Obsérvese que se ha conservado la terminología en lengua inglesa en aquellos casos en que es ésta la usada habitualmente.

- Full-custom ASIC
- Standard-cell ASIC
- Gate array ASIC
- Dispositivos complejos programables en campo (FPGA y CPLD)
- Dispositivos sencillos programables en campo (PROM, PAL y PLA)
- Componentes estándar de pequeño y medio tamaño

A continuación, se describen algunas de las características más relevantes de cada una de estas tecnologías.

Full-custom ASIC

En la tecnología *full custom* todos los aspectos del circuito integrado son diseñados específicamente para una determinada aplicación, especificándose las características de cada uno de los dispositivos electrónicos (transistores, diodos, capacitores, etc.) que componen el circuito. En consecuencia, el circuito resultante estará completamente optimizado y tendrá el mejor comportamiento posible.

Sin embargo, realizar el diseño al nivel de los dispositivos electrónicos es complejo y sólo resulta práctico hacerlo para circuitos con un número de dispositivos relativamente bajo. No resulta práctico usar este procedimiento para diseñar sistemas completos, los cuales en la actualidad pueden constar de decenas e incluso cientos de millones de transistores.

La aplicación fundamental de la tecnología *full custom* es el diseño de pequeños circuitos digitales básicos, denominados "*celdas estándar*", que posteriormente serán usados de manera modular para componer circuitos de mayor tamaño. Un tipo importante de este tipo de módulo son los circuitos básicos a partir de los cuales, por repetición, se construyen celdas estándar. Dos ejemplos son el circuito que implementa un bit de memoria y el circuito que realiza la suma de dos números de un bit, que son usados de manera repetitiva para el diseño de una memoria y un sumador de dos números de n bits, respectivamente.

Standard-cell ASIC

En la tecnología *standard cell* el circuito integrado es construido conectando entre sí celdas estándar predefinidas, que ya han sido previamente diseñadas y testeadas.

Esta tecnología permite trabajar al nivel de puertas lógicas, en lugar de al nivel de transistores, lo cual simplifica bastante el proceso de diseño.

Los fabricantes de dispositivos normalmente proporcionan librerías de celdas estándar que implementan los bloques constitutivos básicos, tales como puertas lógicas, componentes combinacionales simples (and-or-inverter, MUX 2:1, sumador completo de 1 bit, etc.), elementos básicos de memoria (latch-D, flip-flop D, etc.), memorias RAM, etc.

El circuito se construye conectando estas celdas estándar. El tipo de celdas empleadas y la forma de conectarlas depende de la aplicación. Así pues, cada diseño de circuito integrado de esta tecnología requiere de un juego específico de máscaras de fotolitografía, con lo cual debe ser construido específicamente en la fábrica de circuitos integrados.

Gate array ASIC

En la tecnología *gate array* el circuito es construido a partir de un array predefinido de "*celdas base*". Al contrario que en la tecnología *standard cell*, el circuito integrado de la tecnología *gate array* consiste en la repetición unidimensional o bidimensional de un único circuito sencillo, llamado *celda base*, que es similar a una puerta lógica.

Asimismo, los fabricantes de gate array ASICs proporcionan librerías de componentes prediseñados, denominados "*macro celdas*", que son arrays de celdas base conectadas entre sí para formar bloques lógicos con funcionalidades básicas. Las macro celdas facilitan el diseño, que puede realizarse conectando entre sí macro celdas.

Puesto que el diseño estructural de las celdas base y su posición en el circuito integrado están predefinidos, es posible prefabricar el array de celdas base. La particularización del circuito se realiza especificando cómo deben ser conectadas entre sí las celdas base.

Así pues, los primeros niveles de fabricación de los circuitos integrados de la tecnología *gate array*, que están dedicados a la fabricación de los transistores de las celdas base, son comunes a todas las aplicaciones y por ello pueden ser fabricados independientemente de la aplicación. Sólo las máscaras de fotolitografía de los niveles de metal, que definen las conexiones entre las celdas base, deben ser diseñadas específicamente para la aplicación. Al reducirse el número de máscaras de fotolitografía que deben ser fabricadas específicamente para la aplicación en concreto, el proceso de fabricación se simplifica considerablemente.

Dispositivos complejos programables en campo

La tecnología non-ASIC más versátil es el *circuito integrado complejo programable en campo*, que consiste en un array de *celdas lógicas* genéricas y en la interconexión genérica entre ellas. Aunque las celdas lógicas y su interconexión están prefabricadas, el circuito integrado dispone de fusibles que permiten "programar" el circuito integrado, adaptándolo a su aplicación.

Este proceso de programación del circuito integrado, consistente en fundir algunos de sus fusibles, puede realizarse empleando instrumental relativamente barato y sencillo de usar. Puesto que la particularización del circuito integrado se realiza "en campo", esta tecnología se denomina "programable en campo" (*field programmable*), en oposición a las tecnologías ASIC, en las cuales el circuito integrado debe ser particularizado en la fábrica en la que es construido.

Las celdas lógicas de los dispositivos programables en campo son bastante más complejas que las celdas base de los gate array ASICs. De acuerdo a la estructura circuital de sus celdas lógicas, los dispositivos complejos programables en campo pueden clasificarse en dos tipos: CPLD (*Complex Programmable Logic Device*) y FPGA (*Field Programmable Gate Array*).

Dispositivos sencillos programables en campo

Los dispositivos sencillos programables en campo son, como su nombre indica, dispositivos programables con una estructura interna más sencilla que CPLDs y FPGAs. Históricamente, este tipo de circuitos integrados se han llamado PLDs (*Programmable Logic Devices*). Están compuestos por dos arrays, uno de puertas AND y otro de puertas OR. Pueden programarse las interconexiones en uno de estos arrays o en ambos, con el fin de adaptar el circuito a su aplicación.

- En los dispositivos PROM (*Programmable Read Only Memory*) puede programarse el array OR.
- En los dispositivos PAL (*Programmable Array Logic*) puede programarse el array AND.
- En los dispositivos PLA (*Programmable Logic Array*) pueden programarse ambos arrays.

En la actualidad, los circuitos integrados de este tipo son usados raramente, empleándose generalmente FPGAs.

Circuitos estándar de pequeña y media integración

Antes de la popularización de los dispositivos programables en campo, la única alternativa que existía a los circuitos integrados ASIC era utilizar circuitos integrados prefabricados SSI/MSI (*Small-/Medium-Scaled Integrated circuits*). Un ejemplo es la serie 7400 de la familia TTL (*Transistor-Transistor Logic*), que contiene más de 100 circuitos diferentes, que iban desde puertas NAND simples hasta unidades aritméticas de 4 bits.

Los circuitos se diseñaban seleccionando cuáles de esos circuitos integrados debían usarse y fabricando una placa impresa específica para la aplicación, con las pistas para la conexión entre los pines de los circuitos integrados, y en la cual se soldaban los circuitos integrados. En este tipo de implementación, la mayor parte de los *recursos* (potencia consumida, área y coste de fabricación) es consumida por el encapsulado y el routing, y no por el silicio, que es donde se realiza la computación. Dado que los dispositivos programables tienen mejores capacidades y son más baratos, hoy en día no se diseñan circuitos complejos usando tecnología SSI/MSI.

1.4.2. Comparación entre tecnologías

Una vez se ha decidido desarrollar hardware para una aplicación, debe escogerse qué tecnología emplear. En esta sección se compararán las tres tecnologías FPGA, *gate array* y *standard cell*, empleando para ello cuatro criterios: área, velocidad, potencia consumida y coste. Cada tecnología tiene sus puntos fuertes y débiles, de modo que para cada aplicación debe decidirse qué tecnología es más adecuada.

Área

El área del circuito integrado (o equivalentemente, su tamaño) depende de la arquitectura del circuito y de la tecnología. Los circuitos integrados pequeños requieren menos recursos, precisan de programas de test más sencillos y tienen un mejor rendimiento en fabricación, entendiendo dicho rendimiento como el porcentaje de chips buenos que se obtienen por oblea.

Frecuentemente, una misma funcionalidad puede conseguirse empleando diferentes arquitecturas, con diferentes áreas y velocidades. Por ejemplo, existen diferentes circuitos que realizan la suma de dos números de n bits, algunos de ellos son sencillos (ocupan poco área) y lentos, mientras que otros son complejos (ocupan mayor área)

y rápidos. Una vez se ha determinado la arquitectura del circuito, el área del circuito integrado depende de la tecnología.

En la tecnología *standard cell*, las celdas estándar y sus interconexiones son particularizadas para la aplicación en concreto, con lo cual no hay desperdicio de área de silicio. El chip resultante está completamente optimizado y el área es mínima.

En la tecnología *gate array*, el circuito debe ser construido a partir de celdas base cuya posición en el circuito está predefinida. Dado que la funcionalidad y posición de las celdas base no son específicas a la aplicación, el aprovechamiento del silicio no es óptimo. En consecuencia, normalmente un circuito integrado de la tecnología *gate array* necesita mayor área (aproximadamente entre un 20 % y un 100 % más) que ese mismo circuito desarrollado en la tecnología *standard cell*.

En la tecnología FPGA, una parte considerable del área del circuito integrado está dedicada a posibilitar que el circuito pueda ser programado en campo. Más aun, la funcionalidad de las celdas lógicas y las interconexiones están preestablecidas, siendo frecuente que cierto porcentaje de la capacidad no sea utilizado en la aplicación en concreto a la que se destina el FPGA. Por todo ello, puede estimarse que un circuito implementado en la tecnología FPGA tiene entre 2 y 5 veces mayor área que ese mismo circuito implementado en una tecnología ASIC.

Velocidad

La velocidad de un circuito digital es el tiempo necesario para que dicho circuito realice su función. Este tiempo se estima considerando el mayor retardo en la propagación que puede producirse en cada etapa del circuito. Normalmente, cuanto más rápido sea un circuito, mejor. Sin embargo, para poder realizar las operaciones más rápidamente, es necesario emplear circuitos con arquitecturas más complejas, que ocupan mayor área.

Si se usan arquitecturas idénticas, normalmente un circuito integrado con mayor área es más lento, debido a sus mayores capacidades parásitas.

Dado que en la tecnología *standard cell* las interconexiones y el área pueden ser optimizadas, esta tecnología es la que permite menores retardos de propagación y, consiguientemente, mayor velocidad. En el otro extremo, la tecnología FPGA es la que tiene retardos de propagación mayores.

Al igual que sucedía con el área, la diferencia que existe respecto a la velocidad entre las tecnologías ASIC *standard cell* y *gate array* es considerablemente menor que entre estas tecnologías ASIC y la tecnología FPGA.