DISEÑO DIGITAL MODERNO

MODERN DIGITAL DESIGN

Por Jaime Orlando Ruiz Pazos
Ingeniero Electrónico, Universidad del Cauca
MSc. Sistemas Digitales, Instituto Superior Politécnico José Antonio Echeverría, La Habana
Profesor Asistente Facultad de Ingeniería, Programa de Ingeniería Electrónica.
Universidad de Nariño
jaimeruiz@udenar.edu.co

Fecha de recepción: 8 de octubre de 2009
Fecha de aprobación: 21 de octubre de 2009

RESUMEN

El impresionante desarrollo de las tecnologías microelectrónicas conduce a extraordinarios niveles de integración. La capacidad de integración en el silicio se duplica cada 18 meses aproximadamente; esto permite a los fabricantes realizar el diseño de sistemas más complejos en un solo chip de silicio y en un área mucho más reducida.

No obstante, la capacidad de desarrollar estos sistemas más complejos en un período de tiempo razonable, disminuye con el incremento de la complejidad. La capacidad de diseño en un circuito integrado crece a una razón de 21% anual mientras que la capacidad de integración lo hace a 58% anual.

Las técnicas tradicionales de diseño digital no son capaces de aprovechar las ventajas de integración actuales y, como consecuencia, la diferencia entre la capacidad de integración y la productividad del diseño parece incrementarse aún más, tendiendo a disminuir el ritmo de crecimiento de la industria de los semiconductores, si no se recurre a técnicas de diseño innovadoras.

PALABRAS CLAVES

Diseño digital moderno, Hardware reconfigurable, FPGA.

ABSTRACT

The impressive development of microelectronic technologies leads to extraordinary levels of integration. The capacity of integration in the silicon doubles every 18 months approximately. It allows the manufacturers design the most complex systems in a single silicon chip and in much more limited area.

Nevertheless, the capacity of developing these more complex systems in a reasonable period of time diminishes because of the increase of the complexity. The capacity of design in an integrated circuit grows to a reason of 21 % per year whereas the integration capacity does it to 58 % per year.

The traditional technologies of digital design don't take advantage of the current integration technology, as a consequence, the difference between the integration capacity and the productivity of the design seems to increase even tending to diminish the semiconductors industry growth, if it does not resort to innovative design techniques.

KEY WORDS

Modern digital design, reconfigurable hardware, FPGA.

INTRODUCCIÓN

El diseño lógico tradicional tiene varias alternativas de implementación que, por lo general, presenta los siguientes inconvenientes:

- Gran número de CIS.
- Baja fiabilidad
- Dificultad de mantenimiento
- Elevado consumo de área y potencia
- Alto costo (directo e indirecto)

Además, los métodos tradicionales enfrentan problemas relacionados con la situación actual del diseño digital, como:

- Implementación de sistemas más complejos.
- Esfuerzo creciente en el diseño.
- Difícil verificación.
- Reducido tiempo para la introducción de un producto al mercado.
- Ciclos de vida reducidos.

La incapacidad de las técnicas tradicionales de responder a la complejidad creciente de los sistemas digitales, la necesidad de nuevos métodos que elevaran la productividad de los diseñadores y las ventajas ofrecidas por la alta capacidad de alta integración, dieron lugar a la aparición de escenarios modernos de diseño de sistemas digitales.

Estos nuevos escenarios de diseño digital incluyen un conjunto de técnicas, herramientas y metodologías que, utilizadas de manera combinada y adecuada, permitirán incrementar la productividad de los diseñadores, valiéndose de las ventajas de la creciente capacidad de integración.

Dentro de las nuevas técnicas y herramientas para el diseño digital se incluye:

1. Lenguajes de descripción de hardware.
2. Dispositivos programables.
3. Sistemas empotrados.
4. Reusabilidad y Módulos de Propiedad Intelectual (IP).
5. Codiseño hardware/software (HW/SW).

1. LENGUAJES DE DESCRIPCIÓN DE HARDWARE (HDL)

En electrónica, un Lenguaje de Descripción de Hardware o HDL es cualquier lenguaje que realice la descripción software de componentes hardware. Puede describir la operación del circuito y su diseño y probarlo para su verificación por medio de la simulación. Estos lenguajes fueron desarrollados originalmente para descripción y simulación y después, para síntesis. Las ventajas que brinda el uso de HDL son:

- El diseñador tiene la posibilidad de simular, modelar y probar un componente hardware descrito mediante un HDL antes de ser creado físicamente.
- La utilización de los HDL permite un alto grado de abstracción a la hora de describir un elemento hardware. Esto posibilita a los diseñadores concentrarse en la descripción del comportamiento de dicho componente, obviando cómo será su implementación y estructura interna a nivel de compuertas lógicas.
- Facilidad de reusabilidad del hardware descrito, debido a que en el proceso de diseño se utiliza tecnologías genéricas, lo que no fija la tecnología a utilizar, hasta pasos posteriores en el proceso.

Los lenguajes HDL son usados en el diseño de dos tipos de sistemas. Primero, son usados para diseñar circuitos integrados dedicados, tales como procesadores u otro tipo de circuito lógico digital. En este caso, el HDL especifica un modelo para el comportamiento esperado del circuito, el cual permite la simulación y puesta a punto del circuito antes de ser diseñado finalmente y construido. El resultado final es un chip de silicio.

El segundo uso de los HDL involucra la programación de dispositivos lógicos programables, como los PLDs, CPLDs, FPGAs (Field Programable Gate Array), etc. El

código HDL es procesado por algún compilador lógico y el fichero de salida es descargado en el dispositivo. La característica fundamental de este proceso y de la lógica programable en general, es que permite alterar el código muchas veces, recompilarlo y cargarlo en el mismo dispositivo para probar.

Los HDLs más difundidos y con más soporte son:

- **VHDL** (Very high speed integrated circuit HDL), sintaxis similar a ADA.
- **Verilog,** sintaxis similar a C.

Ambos HDLs son estándares IEEE.

2. DISPOSITIVOS PROGRAMABLES

Se entiende por dispositivo programable aquel circuito de propósito general que posee una estructura interna que puede ser modificada por el usuario final (o a petición suya por el fabricante) para implementar una amplia gama de aplicaciones. Los PLDs fueron muy populares desde su aparición por las nuevas ventajas que incorporaban al diseño digital. En la actualidad representan a uno de los sectores que más rápido crecimiento experimenta en la industria de los semiconductores.

Inicialmente un PLD no tiene una función definida, por lo tanto antes de que se pueda utilizar en un circuito, debe ser programado; es decir, establecer las conexiones internas que determinarán una función lógica deseada.

Las características principales de un PLD son:

- Dispositivo con diversos recursos lógicos incorporados.
- Configurable por el usuario.
- Interconexiones programables.
- Requieren herramientas CAD.

Los dispositivos lógicos programables (PLD) forman parte de los circuitos de alto nivel de integración de lógica reconfigurable, que incluye entre otros, los PLDs simples (SPLD), los PLDs complejos (CPLD) y los arreglos de compuertas programables por campo (FPGA). Lo que ha marcado la pauta en el avance de los PLDs es su unidad básica de programación, la cual muestra una construcción de mayor complejidad y mejores prestaciones funcionales a medida que han ido apareciendo.

Dispositivos Lógicos Programables Simple (SPLD).

Se podría decir que los SPLDs son pequeños Circuitos Integrados de Aplicación Específica (ASIC) configurables por el usuario, capaces de realizar una determinada función lógica. Están conformados por una matriz de compuertas AND seguida de otra matriz de compuertas, OR; una de ellas o ambas, pueden ser programadas para implementar cualquier función lógica como suma de términos productos. En este caso la unidad básica de programación es la matriz de compuertas programable.

Dispositivos Lógicos Programables Complejos (CPLD).

Un CPLD está formado por varios bloques de PLDs simples, llamados macroceldas con un sistema de interconexión entre los mismos que permite aprovechar mejor los recursos disponibles. Estos dispositivos brindan al diseñador un número mucho mayor de compuertas lógicas, por tanto, mayor cantidad de términos productos que permiten el diseño de sistemas digitales más complejos, tanto combinacionales como secuenciales.

La unidad básica de programación es la macrocelda y cada una de ellas contiene un biestable, un multiplexor y un buffer de tres estados. El flip-flop almacena el valor de salida de la compuerta, el buffer de tres estados actúa con la estructura de interconexión, la cual está conformada por un conjunto de conmutadores programables, posibilitando que un pin pueda actuar como entrada o como salida en dependencia de lo requerido por el diseñador.

Arreglos de Compuertas Programables Por Campo (FPGA).

Los FPGAs contienen bloques lógicos relativamente independientes entre sí, con una complejidad similar a un PLD de tamaño medio, que pueden interconectarse mediante conexiones programables para formar circui-

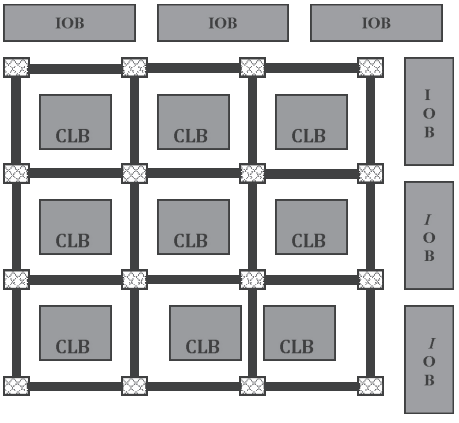
tos mayores. Existen FPGAs que utilizan pocos bloques grandes (Pluslogic, Altera y AMD) y otros que utilizan muchos bloques pequeños (Xilinx, AT&T, Plessey, Actel).

A diferencia de los SPLD, no utilizan arquitectura de matriz de compuertas AND seguida de la matriz de compuertas OR; consisten de una matriz bidimensional de bloques configurables que pueden ser conectados mediante recursos generales de interconexión que incluyen segmentos de pista de diferentes longitudes, más unos conmutadores programables para enlazar bloques a pistas o pistas entre sí. En realidad, lo que se programa en un FPGA son los conmutadores que sirven para realizar las conexiones entre los diferentes bloques, más la configuración de los bloques.

En general la complejidad de un FPGA es muy superior a la de un CPLD. La densidad de compuertas lógicas de un CPLD varía desde el equivalente de varios miles hasta decenas de miles de compuertas lógicas, mientras los FPGAs típicamente varían entre decenas de miles a varios millones.

Antes de abordar este tema es conveniente aclarar que, debido a la gran variedad de ofertas existentes en el mercado, es de suponer que parte de sus diferencias radica en su estructura interna, de tal suerte que algunas de las cosas aquí comentadas se refieren a FPGAs de Xilinx.

Figura 1 Estructura interna de un FPGA.

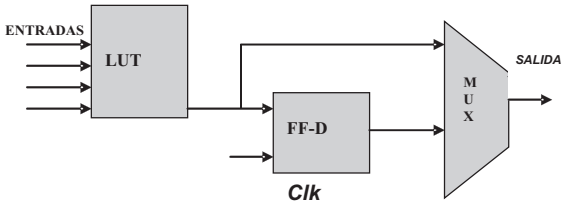


En la figura 1 se muestra la estructura interna básica de un FPGA. La conforman los siguientes elementos:

- Bloques lógicos configurables (CLB).
- Bloques de entrada/salida configurables (IOB).
- Matrices de interconexión programable.

La celda lógica básica de un FPGA consiste en una tabla de búsqueda (lookup table, LUT) de varias entradas, un biestable que actúa como almacenador y un multiplexor, como se muestra en la figura 2. Ésta tiene una sola salida, seleccionada mediante el multiplexor y puede ser la salida almacenada o no, de la memoria (LUT). La celda lógica tiene entradas para la LUT y una entrada de reloj. Dado que las señales de reloj son enrutadas normalmente a través de conexiones dedicadas de propósito especial en FPGAs comerciales, éstas son consideradas independientemente de otras señales.

Figura 2. Celda básica típica de un FPGA.



El nivel inmediato superior de las celdas básicas de los FPGAs de Xilinx es el SLICE; generalmente un *slice* contiene dos celdas básicas como lo muestra la figura 2. Entre los fabricantes de FPGAs e incluso entre las familias de un mismo fabricante, el modelo de los CLBs no es estándar. Pueden variar el número de *slices* por CLB, la distribución y el número de multiplexores dentro de estos.

Los bloques dedicados a la entrada/salida en los FPGAs son denominados IOB (Input/Output Block), soportan una amplia variedad de señalización I/O estándar como PCI, AGP 2X y de entrada/salida diferenciales. Esta característica brinda grandes posibilidades de conexión porque permite incluir una implementación basada en FPGA en muchos tipos de arquitecturas sin hacer cambios sustanciales. El elevado número de pines de entrada/salida de los FPGAs los hace ideales para aplicaciones que requieran el procesamiento de muchas señales digitales.

Los FPGAs actuales incluyen otros elementos que están integrados y que pueden ser utilizados en cualquier momento, entre ellos se puede mencionar:

- Bloques de memoria RAM (BRAMs). Son bloques de memoria dedicados que se usan sin sacrificar la memoria RAM distribuida o los recursos lógicos para conformarla. Según el modelo del FPGA se puede tener más, o menos, de estos bloques.

- Circuitos manipuladores de reloj. Son elementos muy importantes que permiten eliminar las demoras de propagación de las señales de reloj y disminuir el desfase entre las señales de reloj de salida distribuidas en el dispositivo. Con estos elementos también es posible obtener otras frecuencias, ya que son capaces de multiplicar o dividir la frecuencia del reloj de entrada.

- Multiplicadores. Son recursos para realizar operaciones de multiplicación de forma muy rápida y eficiente.

- Procesadores. Algunos de los FPGAs más avanzados tienen procesadores empotrados, lo cual los hace unos dispositivos muy potentes.

Una característica importante de mencionar, es que muchos FPGAs modernos soportan reconfiguración parcial o total *in-system*. Esto permite modificar los diseños dinámicamente para realizar actualizaciones del sistema o para reconfiguración dinámica como parte normal del funcionamiento del sistema. Algunos FPGAs tienen la capacidad de reconfiguración parcial, lo que permite a una parte del dispositivo ser reprogramada, mientras otras partes continúan su funcionamiento.

Programación del FPGA.

Como se mencionó anteriormente la programación del FPGA es equivalente a la configuración de la estructura de conexiones mediante la programación de sus conmutadores y la configuración de los bloques para el desempeño de una función lógica determinada; esa programación debe ser almacenada en una memoria.

Se puede utilizar una memoria SRAM o RAM estática, la cual es una memoria volátil que pierde su contenido cuando no está energizada, lo que obliga a reprogramar el dispositivo siempre que éste se energice nuevamente. También se usa con frecuencia las memorias FLASH que no son volátiles; en este caso la reprogramación sería necesaria únicamente cuando haya cambios en la configuración, pero se debe tener en cuenta que éstas pueden ser reprogramadas un número de veces limitado (entre 10^4 y 10^6).

El tipo más usado en FPGA es el basado en SRAM porque puede ser reprogramado cuantas veces se desee. De hecho, un SRAM FPGA, debe ser programado cada vez que se energiza, porque el chip contiene una memoria RAM volátil donde se guarda la información de la estructura de interconexiones internas y de la función lógica que va a ser implementada. En los diseños en los que se necesite configurar al FPGA inmediatamente cuando se energice el sistema, se utiliza una memoria PROM en serie con la configuración del FPGA almacenada y, cada vez que el sistema es energizado, se descarga la configuración desde la memoria al FPGA.

Las principales ventajas del uso de los dispositivos programables en el diseño de sistemas digitales:

- **Facilidad de diseño**: estos dispositivos facilitan el proceso de implementación de un diseño. Una vez descrito por vía esquemática y/o por medio de un HDL, el diseñador usa las herramientas de CAD para sintetizar, optimizar, simular e implementar este diseño. Esto hace que se obtenga, en muy breve tiempo, un prototipo hardware que permite empezar el proceso de verificación funcional del circuito y corrección de errores.

- **Flexibilidad**: los diseñadores cuentan con elementos de alto nivel contenidos en los PLDs como pueden ser sumadores, multiplicadores, memorias y hasta microprocesadores.

- **Bajos costos de desarrollo**: muchos de estos dispositivos son de bajo costo debido principalmente a la facilidad con la que se puede implementar diferentes diseños en un mismo dispositivo. Las herramientas de desarrollo son baratas y en algunos casos de libre distribución.

- **Ciclo de diseño más corto**: los diseños basados en estos dispositivos reducen el tiempo de introducción al mercado, debido a la facilidad de desarrollo y al mejoramiento de las herramientas de diseño.

- **Elaboración de prototipos**: es una de sus aplicaciones ideales debido al bajo costo de implementación y a la rapidez con la que se obtiene los prototipos. Cambios posteriores en el prototipo pueden ser realizados fácilmente y sin apenas costos adicionales.

- **Reconfiguraciones de hardware** in-system: constituyen un medio ideal para aquellas aplicaciones que requieren un cambio en la funcionalidad del dispositivo durante su fase de operación. Si se necesita corregir fallos o errores de diseño, los cambios son hechos vía software. El diseño puede ser reprogramado, re-implementado y probado inmediatamente.

3. SISTEMAS EMPOTRADOS

Actualmente, los diseñadores integran cada vez más funciones en un solo encapsulado. Los ASICs modernos a menudo incluyen varios elementos de alto nivel como procesadores, bloques de memoria RAM, ROM, EEPROM y Flash, así como otros tipos de módulos para formar lo que se conoce como Sistemas en un Circuito Integrado (System on Chip, SoC).

Un Sistema en un Circuito Integrado (SoC) es un diseño que contiene dos o más elementos de alto nivel como los mencionados anteriormente, desarrollado en un único circuito integrado. El objetivo de los sistemas empotrados es integrar funcionalidad diversa en un solo chip.

Los microcontroladores fueron considerados los primeros SoC, porque contenían varios elementos de alto nivel como el procesador, la memoria, los temporizadores, controlador de IT y otros elementos en un solo encapsulado.

En la actualidad las realizaciones SoC van dirigidas a satisfacer, entre otras, algunas de las siguientes necesidades:

- Integración de 2 ó más macrocomponentes utilizadas previamente como CI independientes.

- Necesidad de combinar soporte SW y HW específico en el mismo CI.

- Combinación de componentes digitales y analógicos.

- Combinación con elementos mecánicos (MEMS).

Debido al desarrollo exponencial que ha experimentado el campo de la microelectrónica, la implementación de sistemas en un circuito integrado (SoCs) se ha vuelto una realidad. Su evolución se ve favorecida por el alto grado de integración que puede alcanzarse y por la disponibilidad de recursos que brindan los dispositivos modernos.

Los SoCs son desarrollados sobre circuitos integrados para aplicaciones específicas (ASICs). Como se conoce, la utilización de los ASICs no brinda mucha flexibilidad ni posibilidades de reconfiguración de los sistemas. Dados los recursos disponibles en los FPGAs actuales es posible implementar sistemas completos sobre estos dispositivos, obteniéndose así los denominados SoPC (System on Programable Chip).

En el desarrollo de ambos tipos de sistemas la etapa de depuración y simulación con las herramientas CAD es muy común. Incluso en el desarrollo de los SoC, se utiliza dispositivos programables (FPGAs) para comprobar el correcto funcionamiento del sistema, antes de comenzar la fabricación del ASIC final. Esto permite la detección de errores que pueda tener el diseño y, a su vez la corrección antes de comenzar su fabricación, porque este proceso es extremadamente costoso.

4. REUSABILIDAD Y MÓDULOS DE PROPIEDAD INTELECTUAL (IP)

Constituyen otro de los elementos que hacen parte de los nuevos escenarios del Diseño Digital. El término reusabilidad aparece como una necesidad para tratar de aliviar la antes mencionada creciente diferencia entre la capacidad de integración (tecnología) y la capacidad de diseño (diseñadores) de los sistemas digitales. Surge por las siguientes razones:

- No se puede perder tiempo en diseñar componentes ya utilizados y verificados previamente cada vez que se requiera su utilización.

- Necesidad de reutilización de componentes.

- Necesidad de descripciones SW (usualmente en HDLs) de componentes HW complejas.

Como el término lo indica, la reusabilidad es la capacidad de que, elementos o componentes desarrollados previamente, puedan ser reutilizados en posteriores diseños. La reusabilidad de un componente está directamente vinculada a su flexibilidad, lo que significa la capacidad de éste de ser insertado en distintos sistemas.

Módulos de Propiedad Intelectual (IP)

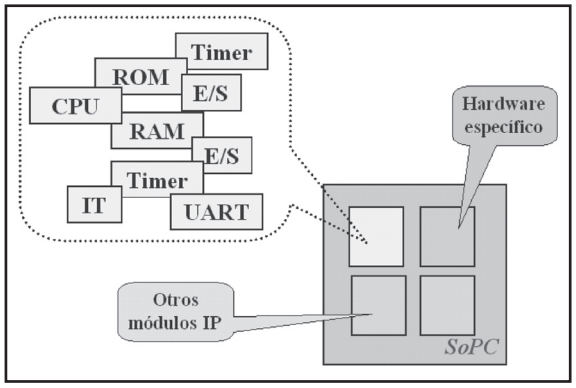
Para llevar a cabo la reusabilidad de los diseños, es preciso disponer de descripciones funcionales muy bien definidas de los diferentes componentes a utilizar. Estas descripciones son conocidas como Módulos de Propiedad Intelectual o Módulos IP.

Los módulos IP pueden ser considerados en general como una descripción software (generalmente utilizando HDL) de un componente hardware. Estos son usados por los diseñadores de sistemas (SoC ó SoPC) para lograr un producto final más complejo.

También conocidos como IP COREs o Megafunctions, los módulos IP son la base de la reusabilidad, siendo ésta una característica inherente y una de sus principales ventajas.

La creciente complejidad de la tecnología de los sistemas empotrados incrementa dramáticamente la carga de diseños y crea la necesidad del empleo de módulos IP previamente verificados para simplificar el diseño de circuitos multifuncionales más complejos. El diseño de sistemas empotrados se ve muy favorecido por el desarrollo de los módulos IP y su posibilidad de reusabilidad. La figura 3 muestra un esquema de la estructura general de un SoPC, en la cual se puede observar los distintos elementos que lo componen.

Figura 3. Representación de un sistema empotrado utilizando módulos IP



Son muchas las ventajas de la utilización de los módulos de propiedad intelectual en el desarrollo de sistemas digitales. Las más importantes son:

- **Capacidad de reutilización**. Permite que un componente, una vez diseñado y depurado, pueda ser incluido en cualquier sistema.

- **Flexibilidad en su utilización**. La mayoría de los módulos IP son configurables, lo que supera un inconveniente de los componentes hardware que describen.

- **Mayor fiabilidad**, debido a que no presentan defectos físicos de fabricación y están previamente probados y depurados.

- **Reducción del ciclo de diseño**, lo que representa un menor tiempo para la introducción de productos al mercado.

Con la aparición de los módulos IP y su indiscutible protagonismo en las técnicas avanzadas de diseño digital, éstas han experimentado cambios:

- Se ha presentado un nuevo esquema de diseño basado en módulos IP.

- El diseño digital pasa de ser puramente hardware (interconexiones de CI) a mayoritariamente software.

- Posibilidad de desarrollos propios. Los diseñadores pueden diseñar nuevos componentes sin tener que fabricarlos físicamente (hardware).

- Requiere potentes herramientas de CAD.

5. CODISEÑO HARDWARE/SOFTWARE

Al explorar el espacio de diseño de un sistema electrónico de cierta complejidad siempre surgen dos alternativas contrapuestas. Por un lado, el empleo de componentes estándares cuya funcionalidad puede ser definida mediante programación. Por otro, la implementación de dicha funcionalidad mediante un circuito microelectrónico, específicamente construido para la aplicación. Es bien conocido que la primera alternativa (alternativa software) proporciona soluciones que presentan una gran flexibilidad a pesar de requerir consumos de área y tiempos de ejecución elevados, mientras que la segunda (alternativa hardware) optimiza los aspectos de tamaño y velocidad de operación, sacrificando la flexibilidad de la solución. A medio camino entre ambas (sistemas híbridos), las técnicas de codiseño hardware/software (HW/SW) pretenden obtener un compromiso adecuado entre las ventajas e inconvenientes de ambas aproximaciones.

Muchos de los sistemas electrónicos modernos constan de componentes de hardware dedicado y software, ejecutándose en plataformas específicas. Los sistemas híbridos no son nuevos, pero han crecido considerablemente en los últimos años debido a las posibilidades de las realizaciones empotradas SoC ó SoPC. Esta tendencia se debe principalmente a la aplicación de técnicas de diseño desde diferentes áreas para desarrollar sistemas digitales mixtos.

Alternativas tradicionales de diseño de sistemas híbridos.

Tradicionalmente, las técnicas de diseño para HW y SW han sido completamente diferentes. Históricamente, estos sistemas han sido diseñados especificando el hardware y subsecuentemente haciendo el software para él.

Cada tipo de implementación (HW o SW) tiene sus características. Éstas determinan que los diseñadores se inclinen por una u otra estructura para las diferentes tareas del sistema a diseñar. Las realizaciones híbridas HW/SW tienen las siguientes características:

- Particionado de tareas "a priori". La decisión de qué tareas serán implementadas en hardware y cuáles en software, se hace basándose en la experiencia de los diseñadores.

  Hardware: tareas poco flexibles que requieran mayor velocidad.

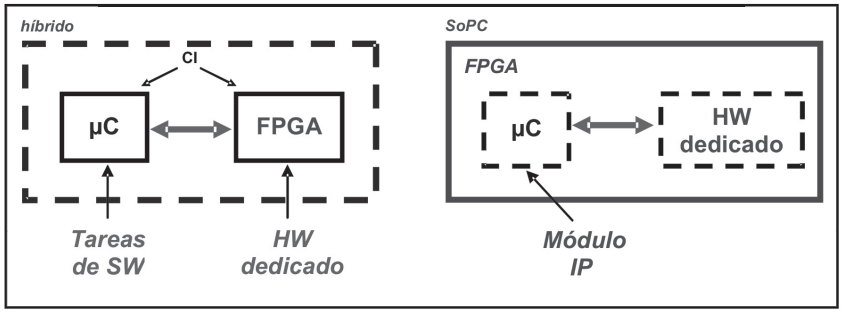
  Software: tareas que consuman muchos recursos y no requieran de altas velocidades.

- Flujos de desarrollo independientes. Una vez determinados los elementos hardware y software que integrarán al sistema, el desarrollo de los mismos se realiza mediante flujos independientes, utilizando herramientas de CAD independientes.

- Pueden ser realizaciones mediante CI independientes (μC + FPGA.) o empotradas (SoPC basado en módulos IP).

En la figura 4 se muestra ambos tipos de realizaciones:

Figura 4. Implementación de sistemas híbridos HW/SW.



Limitaciones de las realizaciones híbridas:

- Imposibilidad de realizar simulaciones de la interacción HW/SW, antes de su integración.
- Incapacidad de diseños óptimos debido a que las decisiones de partición HW/SW son tomadas por los propios diseñadores sin emplear herramientas de optimización.
- Lentitud en la obtención de realimentación en las métricas de calidad.
- Escasez de herramientas automáticas que permitan explorar, en tiempos cortos, las distintas alternativas de diseño.

Codiseño HW/SW.

Como alternativa a las dificultades que presentan los métodos tradicionales de diseño de sistemas híbridos, aparece la metodología de codiseño. Éste se define como la metodología que aborda como un todo, el diseño de partes tanto software como hardware, de un sistema electrónico. Por consiguiente, tal metodología de diseño emplea, tanto técnicas propias del diseño hardware, como de programación, las cuales han sido tradicionalmente consideradas como disciplinas separadas, para producir un diseño flexible y funciones locales eficientes. Además, el codiseño HW/SW debe abordar la conexión de las partes hardware y software, es decir, la generación de interfaces HW/SW.

El diseño concurrente de hardware y software ha mostrado ser ventajoso cuando estos son considerados en conjunto, en lugar de entidades independientes; pero a su vez, involucrar ambos en el diseño de sistemas complejos, no es una tarea fácil.

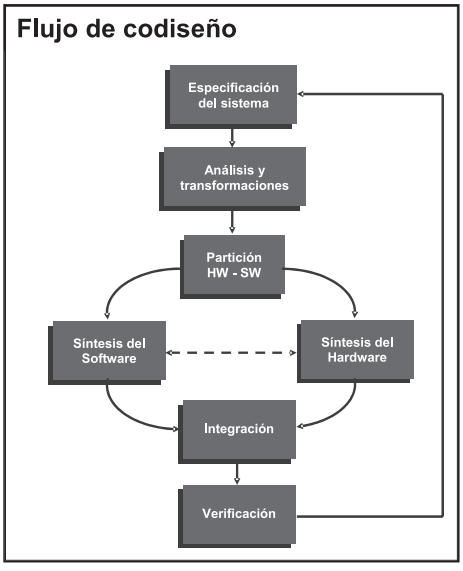
El desarrollo de esta técnica requiere de muy potentes herramientas de CAD.

La utilización de la metodología de codiseño HW/SW ha introducido objetivos y restricciones más fuertes en los diseños:

- Temporización.
- Consumo de potencia.
- Productividad.
- Integración.
- Flexibilidad.
- Fiabilidad y seguridad.
- Verificación.

Dos de las principales características de la metodología de codiseño HW/SW son, primero: la división de los elementos hardware y software que compondrán el sistema no se realiza al inicio del proceso de diseño. Segundo: el flujo de desarrollo de estos elementos no avanza de manera independiente, pues ambos flujos interactúan a lo largo del proceso de diseño con la ayuda de las herramientas de CAD.

Figura 5. Metodología de codiseño HW/SW



Especificación: en esta fase se describe la funcionalidad del sistema, se plantea las restricciones y los objetivos.

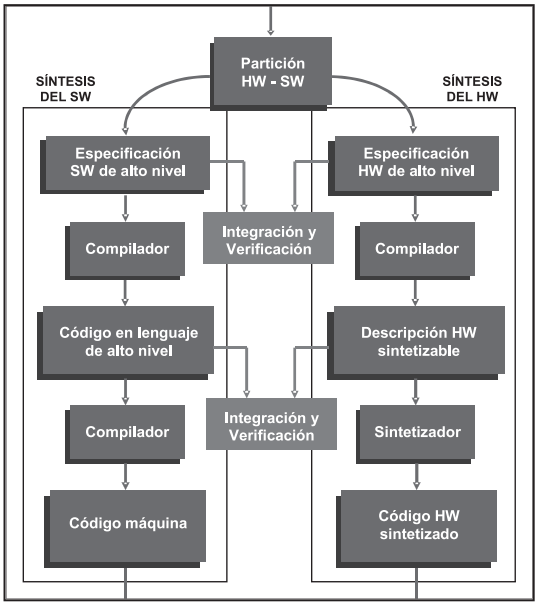
Análisis y transformaciones: aquí se distingue dos etapas: la primera es la de análisis, en la cual se realiza la verificación formal o simulación. La segunda es la de transformaciones y refinamiento, donde tiene lugar la conversión a formato intermedio.

Partición: consiste en determinar qué partes del sistema serán implementadas en hardware y cuáles en software. Tiene como objetivo conseguir el desempeño requerido del sistema dentro de sus restricciones (costo, tamaño, consumo de potencia).

Síntesis del software: esta fase se complica si se considera restricciones de tiempo real. Se realiza la planificación de tareas de manera que se satisfaga todas las restricciones temporales. Ver figura 6.

Integración HW/SW: consiste en la síntesis de la interfaz HW/SW. Tiene como objetivo minimizar las comunicaciones entre estos componentes.

Figura 6. Etapas de síntesis de los elementos de HW y SW

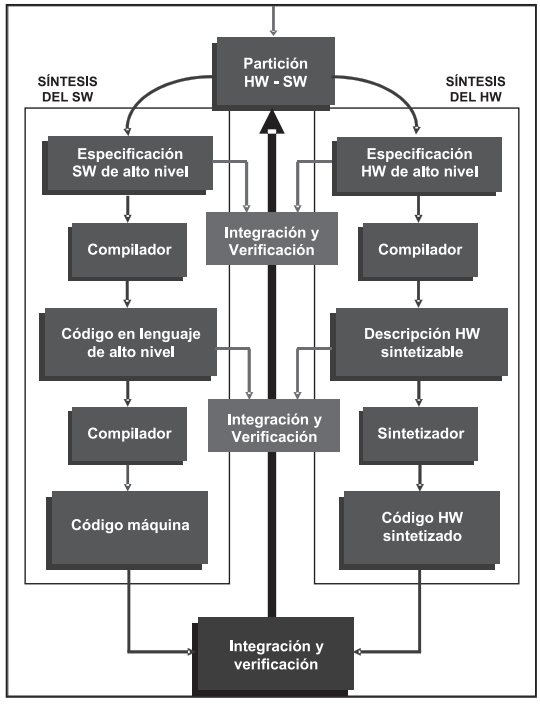


Verificación. Existen dos tipos: la formal, que emplea modelos matemáticos y la no formal, que lleva a cabo las simulaciones, utilizando estímulos. La verificación formal sobrepasa los límites de la simulación, al ser independiente de las entradas. La fase de verificación permite el cumplimiento de requisitos y restricciones más fuertes:

- Funcionales: temporales.
- No funcionales: área, consumo, número de pines, etc.

La figura 7 muestra cómo después de las etapas de integración se verifica los requerimientos iníciales del sistema.

Figura 7. Etapas de integración y verificación



Se debe aclarar que la técnica de codiseño HW/SW para el diseño y desarrollo de sistemas digitales, aunque es muy útil y sus ventajas son indiscutibles, aún no está madura, debido a la elevada complejidad que requieren las herramientas de CAD que asisten en el desarrollo de los diseños. En la actualidad, estas herramientas están en desarrollo y muchos trabajos son encaminados al perfeccionamiento de las mismas.

Es muy importante mencionar que ninguna de las técnicas, herramientas y metodologías que hacen parte de los nuevos escenarios de Diseño Digital se puede ver ni valorar de forma independiente o aislada de las demás, ya que el desarrollo o utilización de cada técnica involucra a otras de ellas. Por ejemplo, el empleo de los HDLs resultaría absurdo sin la existencia de los dispositivos lógicos programables, así como no se podría desarrollar ninguna técnica sin la utilización de los HDLs y no tendría sentido mencionar el codiseño Hw/Sw sin tener en cuenta la reusabilidad y los sistemas empotrados.

También cabe resaltar la importancia de las herramientas de CAD en la aplicación y desarrollo del diseño digital mediante los nuevos métodos de diseño.

La interacción y utilización combinada de las técnicas avanzadas permite:

- Descripciones de alto nivel con elevado grado de abstracción.
- Metodologías de diseño estructurado. Diseños jerárquicos.
- Desarrollos SoC y SoPC.
- Reducción del tiempo de desarrollo.

BIBLIOGRAFÍA

[1] Vaughn Betz and Jonathan Rose, Effect of the Prefabricated Routing Track Distribution on FPGA Area-Efficiency

[2] Coussy, P., Baganne, A., Martin, E. A design methodology for integrating IP into SoC systems, Proceedings IEEE Custom Integrated Circuits Conference, Jan. 2000.

[3] Chiang, S., Foundries and the Dawn of an Open IP Era, 2001

[4] Peter J. Ashenden,VHDL Cookbook, 1990

[5] Parnell, K., Mehta, N., Programmable Logic Design Quick Start Hand Book, Segunda edición, 2002

[6] Torres Fernández, Gildo. Análisis y Evaluación de un Módulo de Propiedad Intelectual del Microcontrolador 8051, 2006

[7] Altera Corporation. CPLDs vs. FPGAs Comparing High Capacity Programmable Logic, 1995

[8] FPGA Reuse Methodology Manual, Segunda Edición, Xilinx Inc.

[9] A Complete Design Solution for Structured ASICs, Magma Design Automation, Inc, 2005

[10] Julio A. de Oliveira Filho, de Lima, Paulo Romero Maciel, Juliana Moura, Bruno Celso, "A Fast IP-Core Integration Methodology for SoC Design", 2003.

[11] Codiseño Hardware/Software de Controladores Difusos A. Cabrera, S. Sánchez-Solano, R. Senhadji, A. Barriga, C.J. Jiménez, I. Baturone.

[12] Sánchez, Pedro Martín, Diseño de sistemas Hardware-Software aplicando técnicas de codiseño, 2003