UTILIZACIÓN DE SMARTMODELS PARA LA VERIFICACIÓN DE SISTEMAS ELECTRÓNICOS DIGITALES.

P.P. CARBALLO[1], M. MARRERO[2], M.C. MARTIN[2] y A.M. ESCUELA[2]
Departamento de Ingeniería Electrónica y Automática. [1]Escuela Técnica Superior de Ingenieros de Telecomunicación. [2]Escuela Universitaria de Ingeniería Técnica de Telecomunicación. Universidad de Las Palmas de Gran Canaria. 35017 – Las Palmas de Gran Canaria. España.

Durante años, la verificación de placas de circuitos impresos (PCBs) se realizaba directamente sobre los prototipos diseñados, sin una fase previa de simulación. Por lo que el éxito del diseño realizado dependía en mayor medida de la experiencia del diseñador y en muchos casos, pequeños o incluso, grandes errores hacía necesario el diseño de otra PCB con la consiguiente pérdida de tiempo y dinero. Con la incorporación al mercado de modelos de comportamiento ya creados de miles de componentes, se hace más ventajoso el uso de una fase de simulación previa del diseño que confirme su correcto funcionamiento antes de enviar a fabricación dicha PCB.

1. Introducción

El principal problema de prototipado con el que se encuentra un diseñador, en relación con la puesta en el mercado de un producto electrónico lo más rápidamente posible, es la disminución del número de iteraciones entre el diseño y la fabricación de la PCB hasta que se verifique funcionalmente el sistema electrónico, ajustándose éste, a las especificaciones iniciales, de tal forma que el número de errores en la PCB sea nulo o mínimo.

Esta metodología no está fuertemente arraigada en el diseño de sistemas electrónicos a nivel de PCB, sin embargo, la influencia de las metodologías orientadas al diseño de circuitos integrados son altamente positivas en este sentido.

En el presente trabajo se intenta dar una visión general sobre las librerías de modelos de comportamiento, concretamente *SmartModels* de *Synopsys*, y proponer una metodología de diseño que los incorpore y que asegure el correcto funcionamiento del sistema electrónico antes de pasar a la fase de prototipado. De esta forma, se reduce el tiempo de desarrollo y el coste del producto ya que no se fabricarán prototipos innecesarios. También pueden emplearse para la verificación efectiva de ASICs y FPGAs.

Estas librerías incluyen ficheros temporales que modelan los parámetros dados por el fabricante. De esta manera, permiten el control automático de errores, indicando posibles infracciones funcionales y temporales, tales como, violaciones de los tiempos de *hold* y *setup*. Se pueden utilizar en la gran mayoría de herramientas de simulación debido a la presencia de

la interfaz SWIFT (*SoftWare InterFace Technology*). Esta interfaz de simulación a nivel de evento es un estándar EDA desarrollado por *Synopsys*, en colaboración con otros fabricantes de simuladores, que permite que múltiples simuladores, con distintos características , usar la misma librería *SmartModel*. Ello elimina la laguna que existía entre la disponibilidad de un modelo y la posibilidad de usarlo en el simulador elegido.

2. *SmartModels*

SmartModels representa un estándar industrial en cuanto a librería de modelos de comportamiento utilizados en la simulación, soportando más de 14.000 dispositivos diferentes. Estos modelos incluyen microprocesadores, controladores, buses y periféricos, FPGAs, CPLDs, memorias y lógica de propósito general de los más populares fabricantes de semiconductores.

Los *SmartModels* modelan los circuitos integrados como "cajas negras" que aceptan estímulos de entrada y responden con el apropiado comportamiento de salida, como si del dispositivo físico se tratase, puesto que describen el comportamiento del dispositivo, incluyendo un chequeo de errores automático: errores de I/O, frecuencia de reloj fuera de rango, registros no inicializados, etc. Además de proporcionar una información completa que permite soportar simulaciones lógicas detalladas, incluyendo retardos pin a pin (mínimo, típico, máximo) y chequeos de tiempo (*setup*, *hold*, ancho de pulso, tiempo de ciclo), que informaran al usuario de condiciones de tiempo ilegales, tales como *glitches*, que podrían retrasar el tiempo total de diseño.

Hay dos tipos básicos de *SmartModel*: los *Full-Functional Models* (FFM) y los *Bus-Functional Models* (BFM). Los primeros, simulan el rango completo del comportamiento del dispositivo, mientras que los segundos, sólo simulan los ciclos del bus. Dentro de la segunda categoría existen dos tecnologías diferentes: los *Hardware Verification Models* (HV) y los *FlexModels*. La diferencia entre ambas es el lenguaje que se utiliza para controlarlos. Mientras los modelos HV usan un lenguaje específico, parecido al C, llamado ***P**rocessor **C**ontrol **L**anguage* (PCL), los *FlexModels* usan los lenguajes de descripción hardware *Verilog* y *VHDL*, ó C. Atendiendo a la función que desempeña y al dispositivo que modelan, los *SmartModels* también se pueden clasificar en: *Memory Models (MM)*, modelan dispositivos de memoria o dispositivos que la contengan; *processor Models (PM)*, modelan microcontroladores y microprocesadores, *PLD Models (PLDM)*, modelan dispositivos de lógica programable, PLDs y PALs; y *smartCircuits Models (SCM)*, modelan CPLDs y FPGAs, así como buses (Standards/Buses) y lógica de propósito general.

3. Flujo de diseño

El flujo adoptado es el que se muestra en la figura 1. Una vez concebida la idea, elegida la metodología y el tipo de implementación, se seleccionan los modelos de verificación que representen a los componentes. Los *SmartModels* se presentan como primitivas que se pueden usar para crear un esquemático; antes de colocar la *instance* del *SmartModel* en el esquemático, se verán sus características, como se usa y configura en el esquemático, como se realiza la simulación con ellos, y que los hacen distintos de otros modelos de simulación. En líneas generales, una vez que se ha diseñado el sistema, se crean los ficheros específicos

de cada modelo, según el tipo de *SmartModels*, y se configura el medio de usuario, para posteriormente capturar el diseño y crear los estímulos que se utilizarán en la fase de simulación. Cuando los resultados son los esperados, se pasa a la fase de implementación del prototipo en PCB. Antes de crear los ficheros que configuran la función del modelo, se debe crear las características específicas que se requieran. Por ejemplo, se puede personalizar los *SmartModels* para que sus características temporales se ajusten a unos valores propios del diseñador, mediante el uso de UDT (*User Defined Timing*), o en el caso de necesitar de una FPGA ó CPLD, que no se encuentre disponible en los *SmartCircuit Models*, se puede modificar una existente para que la soporte. Los ficheros de configuración dependen del tipo de *SmartModels:* ficheros MIF (*Memory Image File*) para los modelos MM, ficheros PCL (*Processor Control Language*) para procesadores y microcontroladores (PM), ficheros JEDEC para PLDs y PALs (PLDM) y ficheros MCF (*Model Command File*) para SCM.

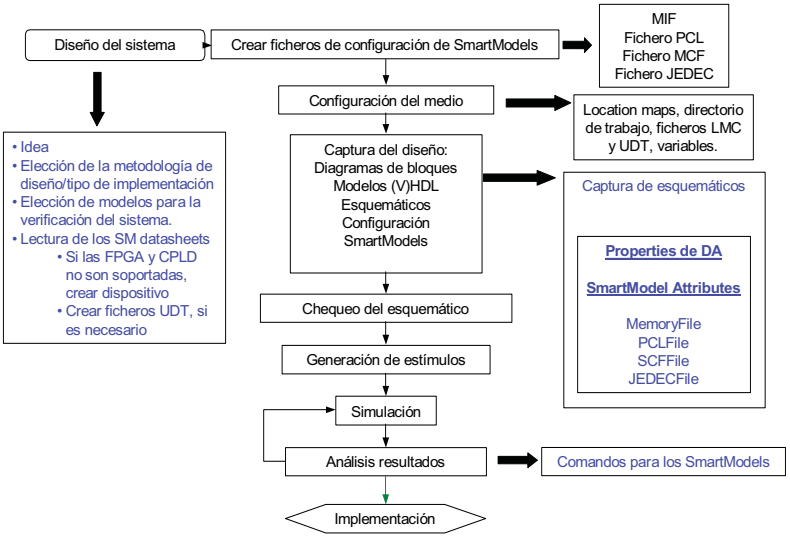

Figura 1. Flujo de diseño adoptado basado en Mentor Graphics

Planteado el modelo software del sistema, se configura el medio del usuario y variables de entorno. Se crea el directorio de trabajo, los ficheros con las versiones de los *SmartModels* y el esquemático. Generados los estímulos se pasa a simular el sistema y a analizar los resultados, hasta que se obtenga el esperado.

4. Ejemplo de aplicación

Como prototipo de evaluación de la metodología propuesta, se ha diseñado un pequeño sistema que incluye un modelo de cada uno de los *SmartModels* estudiados, esto es: el microcontrolador **MC68HC11A8FN1** de Motorola (*Processor Models*), una memoria EEPROM **X2864** de Xicor de 8kbytes (*Memory Models*), un reloj en tiempo real (RTC) **ds1307** de Dallas Semiconductor (*Full Funtional Model)* y una CPLD **XC9536XL** de Xilinx (*SmartCircuit Model*). Se trata de un sistema simple pero lo suficientemente completo como para mostrar en detalle el potencial del entorno de diseño y de simulación.

El sistema diseñado (figura 2) comprueba que el valor de frecuencia de una señal de entrada, está entre dos valores prefijados (mín. y máx.), además de guardar en memoria la frecuencia medida que no se encontrase en dicho intervalo y el tiempo en que ésta se ha producido.

La simulación de los diferentes módulos del diseño se realizó en diferentes pasos, de tal forma que inicialmente se comprobó la correcta escritura y lectura del microcontrolador sobre los puertos, para posteriormente, simular el microcontrolador con el RTC y verificar la correcta comunicación entre ellos Se finalizó con la simulación global del sistema.

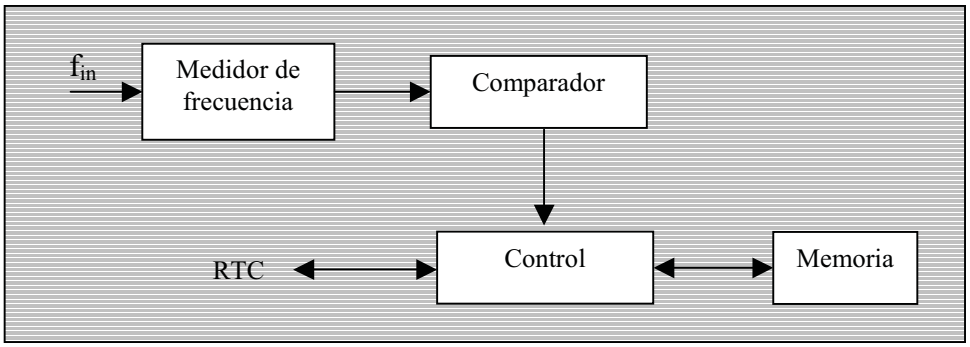

Figura 2. Diagrama de bloques del sistema diseñado.

Durante la simulación se produjeron varios *glitches* que fueron anulados modificando la lógica de generación de la señal de reloj de entrada al RTC. En la figura 3 se muestra una de las múltiples simulaciones realizadas correspondiente a la frecuencia medida y escrita en memoria.

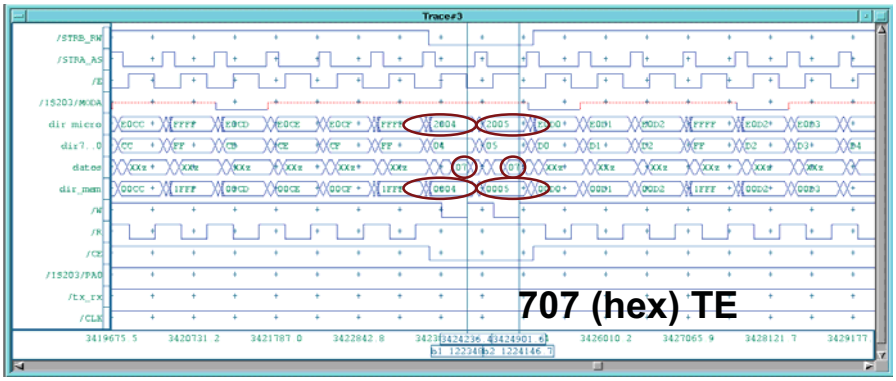

Figura 3. Resultados de la simulación

5. Conclusiones

Esté trabajo plantea el uso de modelos de simulación (*SmartModels*) en el desarrollo de sistemas electrónicos digitales basados en PCB, como fase previa de simulación. Su uso permita el ahorro de tiempo de desarrollo y de coste del producto puesto que evita la fabricación de prototipos innecesarios al verificar completamente su correcto funcionamiento.

Referencias

[1] Axel Jantsch. *System Modelling: Model of Concurrency and their Applications.* RIT 2001
[2] SmartModel Library User´s Manual. Synopsys. 2000.
[3] SmartModel Products Application Notes Manual. Synopsys. 2000.
[4] Design Architect User´s Manual. Mentor Graphics.