SIE: Plataforma Hardware *copyleft* para la Enseñanza de Sistemas Digitales

Carlos Iván Camargo – Universidad Nacional de Colombia
cicamargoba@unal.edu.co

***Abstract*—El gran avance de las técnicas de fabricación de Circuitos Integrados ha permitido que los sistemas embebidos sean parte fundamental de nuestra vida, aún sin darnos cuenta diariamente interactuamos con decenas de ellos; esto unido a la disponibilidad de herramientas software de desarrollo gratuitas (compiladores, simuladores, librerías, Sistemas Operativos, herramientas CAD) abre grandes posibilidades comerciales para paises en vía de desarrollo ya que no son necesarias grandes inversiones de capital para la concepción, diseño, y fabricación de estos sistemas.**

Los dispositivos *hardware copyleft* suministran la información necesaria para reproducir, modificar y construir el dispositivo físico, son el complemento ideal del software libre y del código abierto; la unión software libre y hardware copyleft constituye una herramienta muy poderosa para la transferencia tecnológica y de conocimientos en el diseño de sistemas digitales ya que pueden ser utilizadas como referencia para nuevos productos o como material de estudio. En este artículo se presentan las características de una plataforma *hardware copyleft* y su aplicación en la enseñanza del diseño digital.

***Index Terms*—Educación en Ingeniería, Sistemas Embebidos, Hardware copyleft, Co-diseño HW/SW.**

I. Introducción

En la actualidad estamos presenciando una tendencia global a delegar las tareas de manufactura de sistemas digitales a paises asiáticos, donde la mano de obra calificada es abundante y barata; se presentan casos donde los creadores de una determinada tecnología no la desarrollan y dejan que estos paises se beneficien de sus descubrimientos [1]. Esta situación se agrava a medida que las grandes empresas manufactureras asiáticas como Foxconn capturan la producción de los grandes diseñadores como Apple, Nokia, DELL, HP y Microsoft, lo que genera el cierre de empresas manufactureras a lo largo del mundo, con la consecuencia de pérdida de empleo masivo. En la actualidad según *Bureau of Labor Statistics* y *Thomson Financial Extel Company Report* Foxconn emplea a más personas que Apple, Dell, Micorsoft, HP, Intel y Sony combinados; esta situación es más grave en paises en vía de desarrollo, donde no existe la plataforma tecnológica para fabricar dispositivos digitales. Lo anterior lleva a preguntarse por la función y la situación de un profesional en áreas afines a la ingeniería electrónica en países donde no existe la capacidad de concepción y diseño.

La tendencia moderna en los programas académicos a la utilización de herramientas de alto nivel para la enseñanza en areas afines al desarrollo de dispositivos digitales ocasiona que los profesionales no adquieran las habilidades necesarias para completar la cadena concepción - diseño - implementación y operación, en la mayoría de los casos se generan habilidades para la concepción y el diseño a alto nivel y dejan los otros pasos en manos de herramientas especializadas y/o a empresas asiáticas. Esta situación resulta la más atractiva desde el punto de vista económico, ya que no es necesario adquirir maquinaria costosa ni contratar personal calificado para operarlas; sin embargo, limita la generación de empleo local a personas con un nivel de formación alto [1] generando desempleo en las personas menos capacitadas. Según Jon Hall presidente y CEO de Linux International " algunas facultades preparan a la gente en el uso de productos en vez de tecnologías de nivel básico" [2]. Esta situación unida al abandono de la implementación hace que la dependencia con las empresas manufactureras asiáticas aumente cada vez más.

Según el ex-director ejecutivo de Intel Andy Grove [1] la solución está en hacer de la creación de empleo la política económica gubernamental más importante y hacer que las demás giren en torno a ella; además, es necesario volver a la producción interna con el fín de generar nuevos empleos, y volver a adoptar medidas que protejan la producción interna de los productos asiáticos. Sin embargo, para lograrlo es necesario crear en los profesionales las habilidades necesarias para implementar productos comercializables. Aunque la situación latinoamericana es diferente, podemos aprender de estas experiencias y seguir el camino trazado por Brasil, Argentina y Chile en la búsqueda de independencia tecnológica y generación de empleo.

A. Flujo de diseño de sistemas embebidos

Los Sistemas Embebidos son sistemas heterogéneos que contienen componentes Software (microcontroladores, microprocesadores o DSPs) y Hardware (funciones implementadas en Dispositivos lógicos programables PLDs); por este motivo, es necesario adquirir habilidades en la utilización de lenguajes de programación como C o C++ para implementar las funciones software y Verilog o VHDL para la implementación de las tareas hardware; adicionalmente, deben conocer las diferentes formas de comunicación entre estos dos tipos de funciones.

En la figura 1 se muestran los conceptos que deben dominar los diseñadores de sistemas embebidos, y las tareas que deben realizarse para la concepción, diseño e implementación de un sistema embebido. En una gran parte de los programas académicos se estudian únicamente los temas relacionados con la concepción y diseño centrándose en las especificaciones funcionales del sistema, utilizando herramientas comerciales

o COTS (Commercial off-the-shelf) para su implementación. Nuestra propuesta se basa en la utilización de herramientas abiertas tanto hardware como software que permitan recorrer todo el proceso de concepción, diseño e implementación y de esta forma obtener un entendimiento integral del proceso sin generar dependencia a productos comerciales.

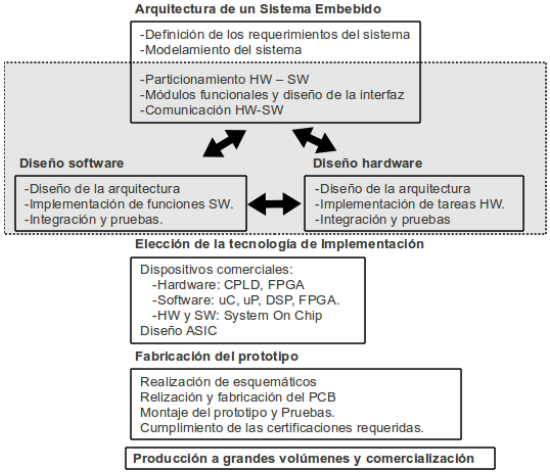

Fig. 1. Educación de sistemas embebidos. Tomada de: [3] y modificada

1) Dominios de Diseño y Niveles de abstracción: Existen tres dominios en los que se puede describir un sistema digital *Funcional*: Describe el comportamiento funcional y temporal del sistema *Estructural*: Describe su composición a partir de bloques básicos y *Físico* relacionado con la estructura física del sistema a nivel de circuito integrado o placa de circuito impreso [4]. Cada uno de estos dominios puede ser descrito utilizando diferentes niveles de abstracción. En el mercado existen herramientas que permiten entradas de diseño funcional utilizando especificaciones y algoritmos y de forma automática y optimizada generan representaciones en diferentes niveles de abstracción en los dominios estructural y físico. Desde el punto de vista comercial esto es muy útil ya que permite reducir el tiempo de diseño y los costos asociados. Sin embargo, no son muy adecuadas en la enseñanza de técnicas de diseño ya que proporcionan capas de abstracción que realizan ajustes a plataformas o a dispositivos comerciales ocultando el proceso completo al usuario; adicionalmente, su costo es muy elevado y en agunas ocasiones no se pueden adquirir de forma directa.

2) Plataformas de Desarrollo Comerciales: Existen en el mercado una gran variedad de plataformas de desarrollo que pueden ser utilizadas en la enseñanza de sistemas digitales, muchas de ellas proporcionan una gran variedad de recursos hardware y notas de aplicación con código fuente para ilustrar su programación y forma de uso. Sin embargo, presentan alguno de los siguientes inconvenientes:

- No es posible modificarlas: Lo que dificulta su uso en productos comercializables ya que es necesario construir placas sencillas adicionales para poder adaptarlas a un determinado problema.
- Los archivos de diseño de los esquemáticos y de la placa de circuito impreso no están disponibles, o se utilizan herramientas CAD muy costosas.
- Algunos módulos son propietarios y no se suministra ningún tipo de información (esquemático, referencia de los componentes)
- No se suministra el códgo fuente: En el caso de los dispositivos lógicos programables se proporciona un netlist en formato propietario, en el caso de los procesadores se proporcionan archivos compilados.
- Utilizan componentes difíciles de conseguir o sujetos a acuerdos de confidencialidad.
- No se proporcionan contactos con las empresas manufactureras para fabricación de plataformas modificadas.

3) Importancia del diseño e impementación de placas de circuito impreso: El uso de plataformas comerciales para la implementación de dispositivos digitales en la academia hace que el estudiante no se peocupe por conocer, entender y realizar el proceso de concepción, diseño e implementación de una placa de circuito impreso, lo cual resulta muy importante para la aplicación de una serie de conceptos (capacidad de corriente, interferencia electromagnética, aislamiento, ruido, señales mixtas, velocidad de operación, niveles lógicos, etc); otro inconveniente que se presenta es que el estudiante se convierte en un usuario de tecnologías; una de estas habilidades es la ”lectura” de esquemáticos, entender y comprender la función que realiza cada componente, como debe conectarse y si es compatible con otros componentes.

Observando la forma en que los estudiantes utilizan estas plataformas de desarrollo notamos que existe un desconocimiento parcial de su funcionamiento, cuando se ven obligados a realizar conexiones de dispositivos externos se cometen errores originados por el desconocimiento de parámetros de funcionamiento como velocidad o capacidad de manejo de corriente; incorrecta arquitectura o conexión; falta de comprensión de las hojas de especificaciones suministradas por los fabricantes.

Por lo anterior podemos decir que es muy importante para un estudiante realizar el ejercicio de concebir, diseñar e implementar una placa de circuito impreso, ya que esto ayuda a la creación de habilidades necesarias en su formación y ayudan a formar profesionales con un perfil de diseñador.

II. PLATAFORMA HARDWARE COPYLEFT SIE

Antes de entrar a describir la estructura de la plataforma SIE se realizará una breve descripción del concepto *hardware copyleft*, que nace como una proyección al hardware del proyecto FOSS (Free and Open Source Software) [5], el cual constituye hoy en día la organización auto-gobernada más grande del mundo y es la responsable de la creación de una gran cantidad de herramientas software como el servidor web *Apache*, la suite para oficina *Openoffice*, el navegador *Mozilla* y el sistema operativo *Linux*.

En desarrollo hardware no existe un proyecto de estas dimensiones que permita utilizar desarrollos ya existentes para entender su principio de funcionamiento y realizar modificaciones para adaptarlas a la solución de problemas locales. En

la actualidad existen proyectos aislados como *Arduino*, *Beagle Board*, *chumby* y *NanoNote [6]* que son presentados como *open-source*, todos ellos proporcionan: Esquemáticos detallados en formato PDF; código fuente del software necesario para su operación; listas de discusión para soporte; tutoriales sobre su funcionamiento y programación; software de desarrollo; archivos de diseño: esquemático, layout; archivos para la fabricación de las placas de circuito impreso.

Salvo la plataforma *Arduino*, los costos necesarios para reproducir una de estas plataformas es muy elevado, esto se debe a las herramientas propietarias utilizadas en el diseño (10000 USD); las características de la placa de circuito impreso, 6 u 8 capas, vías de .2mm, (400 USD), el costo de los componentes (400 USD), y el montaje de encapsuldos BGA, TQFP, SMD, y QFN (100). Esto ocasiona que los proyectos *open-source hardware* no presenten una gran dinámica en desarrollo hardware y las contribuciones se limitan a la fabricación de tarjetas de expansión para propósitos específicos. Desde el punto de vista comercial, estos proyectos son muy atractivos ya que permiten modificar un diseño validado, ahorrando mucho tiempo y dinero, desde el punto de vista académico, tener acceso a los archivos de diseño, permite estudiar las técnicas utilizadas en su diseño para poder aplicarlas en proyectos propios.

Del análisis anterior podemos notar que el mayor costo se produce al utilizar herramientas comerciales para el diseño, por esta razón una de las características del *hardware copyleft* es la utilización de software libre o gratuito durante todo el proceso de diseño; esto permite que muchas universidades, empresas de todo tipo y personas interesadas puedan beneficiarse del proyecto. Para que un dispositivo sea considerado *hardware copyleft* es necesario que permita su distribución, modificación y fabricación bajo un esquema de licencias que conserve la propiedad intelectual y obligue a sus productos derivados a utilizar el mismo tipo de licenciamiento.

La principal diferencia entre el hardware y el software es la implementación física, por este motivo, todo proyecto *hardware copyleft* debe proporcionar servicios de manufactura y suministro de componentes, por lo que debe estar asociado con empresas manufactureras que permitan la producción a diferentes niveles.

A. Proyecto SIE

El proyecto SIE [7] fué creado para satisfacer las necesidades de los desarrolladores de hardware permitiendo la creación de aplicaciones bajo la licencia Creative Commons BY - SA [8] la que permite la distribución y modificación del diseño (incluso para aplicaciones comerciales), con el único requisito de que los productos derivados deben tener la misma licencia y deben dar crédito al autor del trabajo original. Lo que constituye la base de los productos *hardware copyleft*.

Al ser inspirado en el movimiento FOSS, los dispositivos *hardware copyleft* comparten la misma filosofía [5], y son el complemento perfecto del software libre. Para que un dispositivo HW sea reproducible y modificable es necesario: suministrar los archivos necesarios para la fabricación, es decir, los esquemáticos y los archivos de la placa de circuito impreso (preferiblemente para herramientas abiertas como Kicad o geda); la cadena de herramientas de compilación y depuración para desarrollo de aplicaciones; el código fuente de: el programa que inicializa la plataforma (*bootloader*), la herramienta que carga dicho programa en la memoria no volátil (*usbboot*), el sistema de archivos y aplicaciones (*openwrt*); documentación completa que indique como fué diseñada, construida, como utilizarla, desarrollar aplicaciones y tutoriales que expliquen el funcionamiento de los diferentes componentes. (Todo esto puede ser descargado del sitio del proyecto: http://projects.qi-hardware.com/index.php/p/nn-usb-fpga/). Adicionalmente, se debe contar con la posibilidad de fabricación y montaje, lo que constituye la principal diferencia entre el software y el hardware libre. Esto contrasta fuertemente con el movimiento de software libre, en donde no se requiere inversión de capital para modificar un proyecto existente. Por esta razón, deben existir varios niveles de libertad, un proyecto que utilice componentes costosos y de difícil consecución limitará su alcance a un sector determinado. El nivel de libertad más alto requiere bajas sumas de dinero para replicar y/o modificar el hardware, mientras los niveles de libertad bajos requieren grandes sumas de dinero; SIE fue diseñado para que pueda ser fabricado fácilmente por el que este interesado.

1) Arquitectura: La Figura 2 muestra el diagrama de bloques de la plataforma SIE, en ella podemos encontrar un procesador que posee periféricos para comunicación serial (UART), memorias micro-SD, un puerto I2C, un LCD a color de 3 pulgadas, 2 entradas y salidas de audio stereo, 2 entradas análogas; una FPGA que proporciona 25 señales de entrada/salida digitales de propósito general (GPIOs) y controla un conversor análogo digital de 8 canales. Existen dos canales de comunicación entre la FPGA y el procesador: uno para controlar el puerto JTAG, lo que permite la configuración de la FPGA desde el procesador (lo que elimina la necesidad de cables de programación); y otro que proporciona el bus de datos, dirección y control para comunicarse con las tareas HW o periféricos implementadas en la FPGA. El procesador utilizado es un Ingenic JZ4725 (MIPS) corriendo a 400MHz, se dispone de una memoria NAND de 2GB para almacenamiento de datos y programas, así como de una memoria SDRAM de 32 MB, lo que permite la ejecución de una gran variedad de aplicaciones Linux.

2) Comunicación con el usuario y programación: SIE proporciona un canal de comunicación y alimentación a través del puerto USB-device, y es configurado para ser utilizado como una interfaz de red (*usb0*), permitiendo la transferencia de archivos y ejecución de una consola remota utilizando el protocolo *ssh*; este canal de comunicación también se utiliza para programar la memoria NAND no volátil, por lo que para realizar la programación completa de los componentes de la plataforma solo es necesario un cable USB. SIE posee un sistema de archivos basado en el proyecto *openwrt* y dispone de una gran cantidad de aplicaciones y librerías que pueden ser compiladas en un computador tradicional, siguiendo las instrucciones contenidas en la wiki del proyecto. Las aplica-

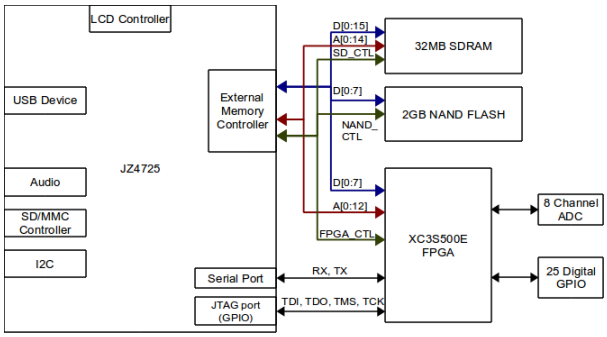

Fig. 2. Estructura de la plataforma de desarrollo SIE

ciones software se cross-compilan utilizando las herramientas suministradas por el proyecto *GNU-Linux*; para el componente hardware se utilizan las herramientas gratuitas suministradas por Xilinx. La figura 3 muestra el diagrama de flujo y las herramientas utilizadas para la implementación de las tareas software.

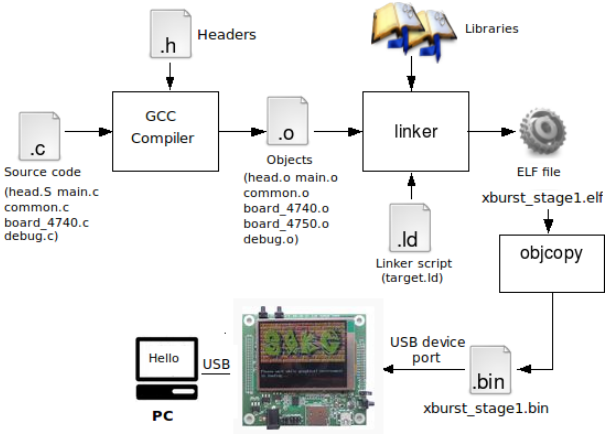

Fig. 3. Flujo de diseño software

3) Especificaciones físicas: Las dimensiones de SIE son 8cm de largo, 8 cm de ancho, 1cm de altura; su placa de circuito impreso es de dos capas, utiliza líneas de 0.2m, vías de 0.3mm de diámetro, los componentes se encuentran en una sola capa y son TQFP o SMD, no posee componentes BGA o QFN lo que facilita el montaje manual. Todo esto hace que sea posible la reproducción y modificación de esta plataforma a un precio muy bajo. El costo de fabricación de esta tarjeta se estima en 70 usd para 50 unidades. La figura 4 muestra la apariencia de la plataforma de desarrollo SIE.

B. Comunidad hardware copyleft

Una parte importante de este método de enseñanza es la filosofía del proyecto hardware copyleft, por esta razón, cada grupo debe hacer un aporte, suministrando la información completa del proceso de desarrollo y suministrando los archivos necesarios para replicar y/o modificarlo, esto es una consecuencia de la licencia CC-BY-SA.

Fig. 4. Plataforma de desarrollo SIE

La experiencia del proyecto FOSS indica muchos miembros de estas comunidades ingresan para suplir necesidades, pero muchos de ellos continuan creando código y prestando servicios a la comunidad porque disfrutan programar. Estos *aficionados* realizan un papel muy importante dentro de la comunidad encargándose de tareas como mejora de la plataforma tecnológica, re-escribiendo secciones de código, documentandolo, respondiendo preguntas, preservando o mejorando la arquitectura [9]. Las actividades de documentación además de contribuir a mejorar las habilidades de escritura de reportes técnicos ayudan a formar una comunidad que contribuye al crecimiento del proyecto copyleft hardware, los estudiantes ingresan a las listas de desarrolladores aprendiendo a utilizar una herramienta muy poderosa en la que pueden compartir sus inquietudes con miembros más experimentados y mientras participan ayudan a crear un banco de preguntas que pueden ser útiles para futuros miembros. Adicionalmente se obliga a expresarse en un idioma diferente.

Crear estos hábitos ayuda a que los jóvenes sean conscientes de su papel dentro de una comunidad y piensen que sus acciones pueden ayudarla o perjudicarla, los proyectos realizados por ellos podrán ser parte de los recursos de la comunidad (si la calidad del trabajo lo amerita) y pueden ser la continuación de un esfuerzo prolongado o el punto de partida de un nuevo conocimiento; la licencia CC-BY-SA garantiza que todos los trabajos derivados de este recurso serán parte del mismo, lo que permite su crecimiento, la labor de los estudiantes es vital para el uso del recurso común y puede crear miembros que en un futuro formularán políticas y reglas de uso del recurso. Por otro lado, participar en este tipo de proyectos permite crear reputación, la cual puede ser útil para establecer relaciones profesionales, de negocios o personales. El entorno académico es ideal para atraer nuevos miembros a la comunidad hardware copyleft, ya que se trabaja con jóvenes con deseos de ser parte de un grupo y de adquirir conocimientos. Desde el punto de vista comercial este recurso es muy atractivo ya que permite ahorrar mucho tiempo, esfuerzo y dinero para la creación de

nuevos productos. Por otro lado, el concepto de hardware copyleft es una herramienta poderosa para transferir tecnología y conocimientos a los países en vía de desarrollo donde la plataforma tecnológica no está lo suficientemente desarrollada.

III. Integración con la Educación

En la actualidad SIE está siendo utilizada en los cursos de arquitectura de computadores y sistemas embebidos de la Universidad Nacional de Colombia sede Bogotá (cerca de 120 estudiantes utilizando 45 placas), la metodología utilizada se centra en la realización de un proyecto durante el período académico, los estudiantes deben realizar el proceso de concepción, diseño e implementación de una aplicación que da solución a un determinado problema, cada grupo integrado por tres estudiantes está encargado de llevar una *bitácora* del proyecto en la wiki servidor del proyecto QI-hardware [1], incluyendo toda la información generada durante el proceso de diseño (diagramas de flujo, diagramas de bloque, simulaciones) el único requisito es la construcción de una placa hija (diseñada en Kicad) con la funcionalidad que se le desea dar a la plataforma SIE.

A. Máquinas de Estados Algorítmicas

En el primer curso del área de diseño digital en la Universidad Nacional de Colombia, el objetivo es el estudio, diseño e implementación de Máquinas de Estado Algorítmicas utilizando metodologías de diseño como la presentada en [10]. SIE proporciona un canal de comunicación entre el procesador y el puerto JTAG de la FPGA, este canal es utilizado para su configuración y para realizar pruebas a baja frecuencia del circuito implementado en la FPGA. El proyecto SIE proporciona una herramienta basada en *urjtag* que recibe como entradas un archivo (con extensión *pad.csv* generado por la herramienta de Xilinx) con la función de cada uno de los pines de la FPGA y el resultado de la simulación en formato *vcd* (value change dump); utilizando la instrucción INTEST de la cadena JTAG, aplica vectores de prueba al circuito implementado en la FPGA, captura y grafica su respuesta en el LCD de la plataforma; esta herramienta puede verse como una combinación de un analizador lógico y un generador de vectores de prueba de bajo costo (ver figura 5).

B. Arquitectura de Computadores

En la página de descargas del proyecto SIE [2] se pueden obtener implementaciones en VHDL para el procesador MIPS MLITE y en verilog para el procesador de lattice LM32, se suministran los archivos necesarios para el desarrollo de aplicaciones software utilizando la cadena de herramientas GNU y los archivos para realizar la simulación del softcore utilizando *ghdl* e *icarus verilog*; A diferencia de Microblaze de Xilinx (suministra un formato tipo netlist que utiliza los bloques básicos de Xilinx) *plasma* y *lm32* permiten realizar modificaciones en la arquitectura del procesador, permitiendo

[1] http://en.qi-hardware.com/wiki/2010-II/es

[2] http://projects.qi-hardware.com/index.php/p/nn-usb-fpga/source/tree/master/

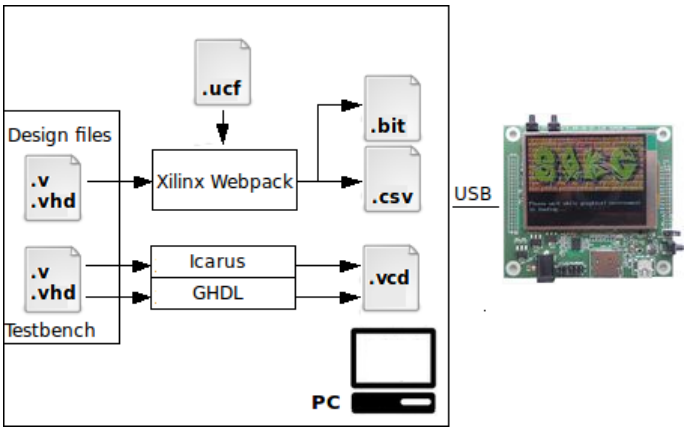

Fig. 5. Flujo de diseño hardware

la creación de nuevas instrucciones y co-procesadores. Adicionalmente, se proporcionan herramientas que inicializan las memorias internas de los procesadores softcore con el código generado por la cadena de compilación; lo que permite realizar un estudio completo del funcionamiento del procesador. En este curso se estudia la arquitectura de un sistema sobre silicio (SoC) y los estudiantes deben concebir, diseñar e implementar un periférico dedicado, conectarlo al SoC y escribir el código en C que lo controla e implementa la funcionalidad. La FPGA de SIE posee recursos limitados, lo que obliga a los estudiantes a optimizarlos y a buscar en el mercado circuitos integrados que les permita cumplir con la funcionalidad requerida.

En este curso el procesador es utilizado como herramienta de configuración de la FPGA, los archivos *.bit* son transferidos al sistema de archivos de SIE utilizando el protocolo *ssh* y desde allí son transferidos a la FPGA utilizando *xc3sprog* o *urjtag*. Ejecutando la aplicación *minicom* en el procesador de SIE, es posible utilizar el puerto serie del procesador implementado en la FPGA como canal de depuración; por lo que no es necesario utilizar programadores o aplicaciones externas.

C. Sistemas Embebidos

QI hardware, proporciona las herramientas software necesarias para el desarrollo de aplicaciones software en los procesadores MIPS de Ingenic JZ4720/25/40; en la actualidad existen aplicaciones abiertas para: programar las memorias no volátiles *usbboot*; inicializar la plataforma y cargar la imágen de linux. *u-boot* crear la imágen del kernel de Linux; crear un sistema de archivos *openwrt*; crear instaladores de un gran número de aplicaciones *opkg-openwrt*; desarrollo software utilizando librerías gráficas como QT o GTK. *openwrt*.

SIE utiliza un procesador JZ4725 y tiene una arquitectura compatible con el producto de QI-hardware *BEN NANONOTE*, por esta razón, todo el desarrollo realizado para BEN es aplicable a SIE, esto garantiza el mantenimiento, actualización y disponibilidad del software necesario para el desarrollo de aplicaciones.

En este curso se utiliza el procesador MIPS para ejecutar tareas de visualización, comunicación, control e interfaz con el usuario. Se utilizan librerías gráficas de alto nivel como QT (de Nokia) para realizar la interfaz, se desarrollan módulos del kernel y programas en espacio de usuario para el control de periféricos dedicados (implementados en la FPGA). Con esto se proporciona a los estudiantes herramientas que están siendo utilizadas en la actualidad por los grandes fabricantes de dispositivos digitales como Nokia, Dell, Hewlett Packard.

La figura 6 muestra la arquitectura de la etapa de comunicación entre el procesador y la FPGA, las señales *DATA, NCS2, we0_n, address* provenientes del procesador no se encuentran en fase con la señal de reloj de la FPGA, por lo que deben ser sincronizadas con este; (el módulo SYNC se encarga de esta tarea). Debido a que no se pueden implementar buses tri-estado en el interior de la FPGA (sólo se pueden utilizar en los pines de la FPGA), por esta razón, los periféricos deben tener dos buses de datos uno de entrada (Color Naranja) y otro de salida (color verde). Se debe proporcionar un circuito a la salida de los buses de los periféricos que permita el paso de la información del periférico seleccionado (en la figura se representa este elemento como un Multiplexor). Debido a que la FPGA es mucho más rápida que el procesador, es necesario generar un pulso con la duración de un ciclo de reloj de la FPGA para la señal de escritura *we0_n*; esto se hace para evitar que se realice más de una operación de escritura (esta tarea es realizada por el módulo Write Pulse Generator, en la figura puede verse el diagrama de tiempos que representa su funcionamiento). Un buffer tri-estado debe controlar el acceso al bus de Datos, este buffer debe presentar un estado de alta impedancia cuando no se selecciona ningún periférico o cuando se realiza una operación de escritura y debe permitir el paso de información desde el periférico seleccionado en las operaciones de lectura.

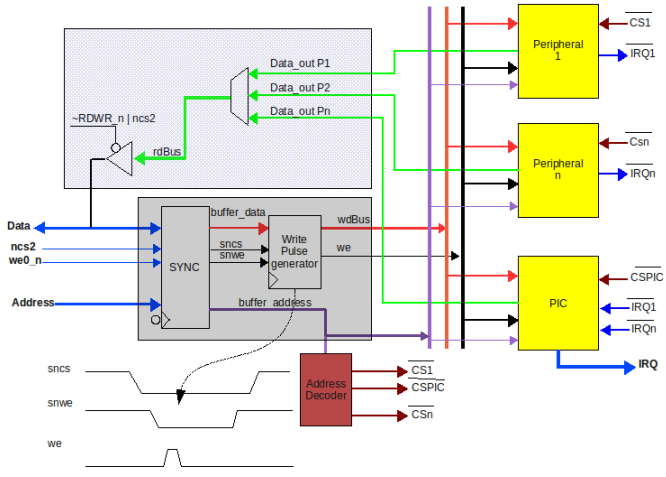

Fig. 6. Interfaz HW-SW en SIE

IV. Conclusiones

SIE es una plataforma única en el mercado de muy bajo costo que proporciona todo lo necesario para desarrollar un curso completo de diseño digital, reduciendo de forma considerable el equipo de medida necesario para la realización de las prácticas. Las actividades propuestas en las tres asignaturas del área tienen como objetivo generar en el estudiante las habilidades necesarias que le permitan diseñar sistemas digitales de diferente grado de complejidad, hasta llegar a un sistema que puede ser comercializable y satisfacer una necesidad de una determinada comunidad, con esto, se evita que el último paso en el proceso de enseñanza sea la simulación.

En este artículo se presentan los pasos necesarios para crear dispositivos comercializables (desde el punto de vista técnico), pero no se presentan los pasos necesarios para su comercialización como estudio de mercado; plan de negocios, financiación; esto será tratado en trabajos posteriores.

La utilización de herramientas abiertas permite que el estudiante conozca y controle los diferentes pasos de la metodología de diseño y sea capaz de ajustarlas para diferentes situaciones, esto hace que se adquiera un conocimiento sobre la tecnología sin crear dependencia hacia las herramientas comerciales.

La utilización de estas herramientas abiertas en los cursos del área de electrónica digital en la Universidad Nacional de Colombia han aumentado la calidad de los trabajos de grado de los estudiantes, pasando de la utilización de procesadores de 8 bits, implementación de prototipos en placas universales (perfboard) o en placas de pruebas (protoboard), lenguaje ensamblador, a aplicaciones que utilizan procesadores de 32 bits, tareas HW implementadas en FPGAs, sistemas operativos, implementadas en placas de circuito impreso con componentes de montaje superficial. Adicionalmente, se logró dar continuidad a los contenidos de las tres asignaturas haciendo que los estudiantes utilicen las herramientas adecuadas para dar solución a un determinado problema (en el pasado todo se resolvía con procesadores de 8 bits, o con FPGAs).

References

[1] A. Grove. How America Can Create Jobs. http://www.businessweek.com/magazine/content/10_28/b4186048358596.htm, May 2010.

[2] Jon Hall. POR GRANDES QUE SEAN...: ASEGURE EL FUTURO DE SU NEGOCIO. *Linux magazine,*, ISSN 1576-4079(58):92, 2009.

[3] H. Mitsui, H. Kambe, and H. Koizumi. Use of Student Experiments for Teaching Embedded Software Development Including HW/SW Co-Design. *IEEE TRANSACTIONS ON EDUCATION,*, 52(3), August 2009.

[4] Gajski D.D., Abdi S., Gerstlauer A., and Schirner G. *Embedded System Design: Modeling, Synthesis, Verification*. Springer, 2009.

[5] R. Stallman. Philosophy of the GNU project. URL: http://www.gnu.org/philosophy/, 2007.

[6] Qi Hardware. Qi Hardware Copyleft Hardware Project. URL: http://en.qi-hardware.com/.

[7] C. Camargo. Proyecto SAKC. *URL:http://en.qi-hardware.com/wiki/SAKC*.

[8] Creative Commons. Licencias Creative Commons. URL: http://creativecommons.org/licenses., 2004.

[9] S. Shah. Motivation, Governance, and the Viality of Hybrid Forms in Open Source Software Development. *Management Science*, July 2006.

[10] Luis Alejandro Cortés. *Verification and Scheduling Techniques for Real-Time Embedded Systems*. PhD thesis, Linköpings universitet Institute of Technology, 2005.