Índice general

Capítulo 1

Metodología Para la Enseñanza de Sistemas Embebidos

1.1. Introducción

La disponibilidad de personal calificado que absorba, asimile y aplique los conocimientos asociados a una tecnología es, a nuestro modo de ver, el punto más importante en el proceso de transferencia tecnológica; como observamos anteriormente, los canales tradicionales de transferencia han demostrado ser poco efectivos en el área bajo estudio; podemos concluir sin temor a equivocarnos que la causa principal del atraso de la industria electrónica (relacionada con el diseño digital) es la poca oferta de personal calificado que permita generar un cambio y pasar de importadores a generadores de soluciones.

Analizando el caso del Departamento de Ingeniería Eléctrica y Electrónica (DIEE) de la Universidad Nacional de Colombia, obervamos que los estudiantes son muy hábiles para analizar y para aplicar sus conocimientos en la solución a problemas académicos; sin embargo, la mayoría presenta una gran dificultad al momento de implementar físicamente estas soluciones; esto, porque en la mayoría de las asignaturas no se llega a este punto ya que el proceso termina en la simulación. La falta de contacto de muchos docentes con la industria hace que los programas de estas asignaturas no incluyan como parte de la metodología la experimentación práctica; algunos docentes argumentan que la implementación práctica no es labor de un ingeniero y por esta razón ignoran este proceso; aunque esta opinión es respetable, a nuestro modo de ver, en un país como Colombia donde no existe una oferta considerable de bienes y servicios relacionados con el diseño de sistemas electrónicos (comparado con países de la región) se le deben proporcionar al estudiante las herramientas necesarias para innovar y crear nuevos productos que den solución precisa a problemas locales. La falta de contacto por parte de los estudiantes a la solución de problemas reales origina una desconfianza por parte de la industria hacia los poductos generados localmente.

Es importante que los estudiantes utilicen herramientas adecuadas cuando se enfrentan a la implementación física. Hasta hace poco en el DIEE se encontraban trabajos de grado que utilizaban placas de prototipos; se proporcionaban soluciones basadas en herramientas comerciales las cuales eran conseguidas de forma ilegal o se utilizaban demos. Este tipo de soluciones están muy lejos de una solución comercial real, primero por que no proporciona una plataforma física robusta y segundo porque es encesario adquirir software muy costoso. Adicionalmente, no es ético utilizar herramientas comerciales en la enseñanza si no se cuentan con las licencias necesarias; es muy común ignorar este tipo de violaciones a los derechos de autor cuando se trabaja en aplicaciones académicas, aunque el fin del uso sea muy loable no deja de ser

ilegal; por esta razón, una de las premisas de nuestra metodología de enseñanza será el uso de herramientas abiertas que produzcan resultados comparables con las herramientas comerciales.

La metodología que proponemos adopta los conocimientos generados en el proceso de transferencia descrito en el capítulo **??** y se constituye en su aplicación a la academia, con esto esperamos que los estudiantes tomen conciencia de la importancia del uso de esta tecnología y utilicen esta información para crear productos novedosos que den solución a problemas locales. El contenido del curso y metodología se ajustará a la iniciativa CDIO. El Departamento de Ingeniería Eléctrica y Electrónica de la Universidad Nacional de Colombia está realizando el proceso de adaptar la Iniciativa CDIO a su programa académico, y en la mayoría de las asignaturas no se contemplan acitividades que ayuden a mejorar las habilidades en la implementación de sistemas, lo que confirma una vez más que la falta de docentes con experiencia en la industria afecta la generación de habilidades necesarias para crear nuevos productos.

1.1.1. Objetivos de la iniciativa CDIO

La iniciativa CDIO [1] ha sido desarrollada por el MIT con ayuda de academicos, industriales, ingenieros y estudiantes[1] como respuesta a los caminos diferentes que están tomando la educación de la ingeniería y las demandas del mundo real [2] Esta iniciativa ha sido adoptada por un creciente número de instituciones académicas a lo largo del mundo. Hacer parte de este esfuerzo mundial nos ayuda a mantener nuestros planes académicos actualizados con los cambios que se realizan en países más industrializados. En este capítulo mostraremos como esta iniciativa se adapta perfectamemente a la metodología propuesta en este trabajo ya que adiciona dos componentes importantísimos para la aplicación de la tecnología en la creación de nuevos productos: la implementación y la operación.

La iniciativa CDIO se basa en la suposición de que los egresados de los centros de formación en ingeniería deben ser capaces de: **C**oncebir, **D**iseñar, **I**mplementar y **O**perar sistemas funcionales en el mundo real. Como mencionamos anteriormente, en Colombia, la mayoría de los centros de formación solo tienen en cuenta la concepción y el diseño, descuidando por completo la implementación y la operación, esto, impide que se generen las habilidades necesarias tenga una estrecha relación con la industria, la cual, requiere productos que pueda comercializar o soluciones a sus necesidades. La frase *en el mundo real* resalta la importancia de trabajar en la solución de problemas que pueden encontrarse en el ejercicio profesional, lo que es muy difícil de determinar cuando los docentes no han tenido contacto con el mundo real. La iniciativa CDIO se enfoca en preparar a los estudiantes con los conocimientos habilidades y aptitudes para ser ingenieros líder; y sus principales objetivos son: [1]:

- Educar a los estudiantes para dominar un conocimiento más profundo de los fundamentos técnicos.
- Educar a los ingenieros para liderar la creación y operación de nuevos productos y sistemas.
- Educar futuros investigadores para entender la importancia estratégica y el valor de su trabajo.

Estos objetivos se adaptan a los requerimientos que se exige a la plataforma tecnológica de un país para que pueda realizar una adecuada absorción del conocimiento transferido para posteriormente transformar ese conocimiento en nuevos productos adaptados a las necesidades del país. Las premisas que capturan la visión, objetivos y fundamentos pedagógicos de la iniciativa son:

- Es posible cumplir las necesidades propias de la profesión mientras se realiza el proceso de concebir, diseñar, implementar y operar sistemas en el contexto de los sistemas de ingeniería.

[1] http://www.cdio.org

[2] lo que se aplica perfectamemente al estado de la industria electrónica (y a otras industrias) en Colombia, donde la enseñanza en ingeniería se queda a medio camino y no se hace implementantación física.

- Los resultados de la formación deben ser fijados por los sectores interesados (academia, industria, gobierno) y deben formar una secuencia de experiencias de aprendizaje, algunas de las cuales son experimentales, es decir, deben enfrentar a los estudiantes a situaciones que encontrarán en el ejercicio de su profesión.

- La adecuada construcción de esta cadena de actividades tendrán un doble impacto en la formación de los estudiantes, por un lado facilitará el aprendizaje de habilidades críticas e inter-personales y fortalecerá las habilidades de construcción de sistemas, productos y procesos, mientras se mejora el aprendizaje de los conceptos fundamentales.

Emprendimiento y liderazgo

La situación actual por la que atraviesa la Industria electrónica nacional, requiere que los profesionales en el área tengan las capacidades de emprendimiento y liderazgo necesarias para la creación de nuevas empresas o para la creación de nuevos productos, la Figura 1.1 muestra la relación entre emprendimiento [3], liderazgo y las habilidades CDIO.

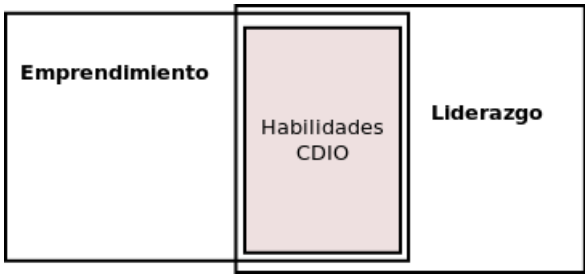

Figura 1.1:

fuente:[2]

1.1.2. Estructura del plan de estudios CDIO

La figura 1.2 muestra los bloques constructores del plan de estudios CDIO, en el primer nivel podemos observar que todo individuo interesado en obtener habilidades técnicas posee *habilidades personales y profesionales*, las cuales son fundamentales para la práctica. Para se capaces de desarrollar sistemas complejos en ingeniería, los estudiantes deben dominar los fundamentos del *razonamiento y conocimiento técnico*; para trabajar en un entorno moderno basado en grupos de trabajo los estudiantes deben desarrollar *habilidades interpersonales* de comunicación y trabajo en equipo; finalmente para ser capaz de crear y operar productos y sistemas un estudiante debe entender el concepto de *concebir, diseñar, implementar y operar sistemas en el Contexto social y empresarial*[3]

[3] El concepto clásico de emprendimiento involucra el re-direccionamiento y movilización de capital y recursos humanos para crear una nueva actividad económica; actualmente, el emprendimiento esta asociado a la creación de una nueva empresa con una nueva línea de negocios.

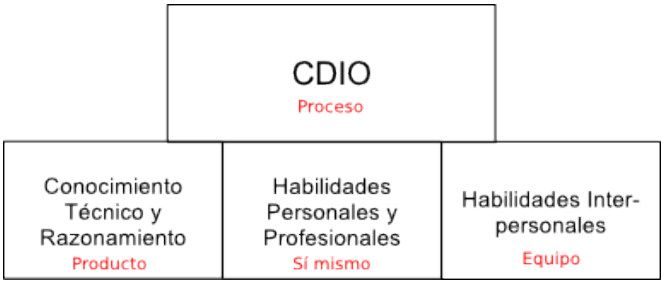

Figura 1.2: Bloques constructores de conocimiento, habilidades y actitudes necesarias para concebir, diseñar, implementar y operar sistemas en el contexto social y empresarial
fuente:[3]

Nivel 1: Razonamiento y conocimiento técnico

Los componentes del primer nivel *razonamiento y conocimiento técnico* son comúnes a los planes de estudio de las ingenierías modernas y son:

1. Fundamentos avanzados de ingeniería.
2. Fundamentos del núcleo de ingeniería.
3. Conocimiento científico.

La razón de colocar este bloque constructor en el primer nivel es solo para recordar que el objetivo primordial de cualquier programa de pregrado es el desarrollo de un conocimiento profundo de fundamentos técnicos. En este trabajo no se cambiará este componente ya que para hacerlo es necesario un consenso con las demás carreras de la facultad de ingeniería, labor que puede tomar varios años.

Habilidades personales, profesionales e interpersonales

Los niveles 2 y 3 se centran en las habilidades personales que debe poseer un individo para que pueda cumplir con el objetivo de la iniciativa CDIO. El nivel 2 esta compuesto por:

1. Las habilidades profesionales que representan las tres formas de pensar más practicadas por los ingenieros: resolución de problemas; descubrimiento de conocimiento y pensamiento sistémico.
2. Actitudes que incluyen integridad y comportamiento profesional así como las necesarias para planear la profesión.

Las habilidades que no hacen parte del contexto profesional ni del inter-personal son llamadas *habilidades y actitudes personales*, incluyen el carácter; iniciativa; perseverancia; formas de pensar más genéricas como pensamiento crítico, creativo; y habilidades propias como curiosidad, aprendizaje continuo y manejo del tiempo.

Las habilidades inter-personales, son un subconjunto de las habilidades personales y se dividen en dos grupos (que se traslapan) llamados: equipo de trabajo y comunicaciones. El equipo de trabajo hace

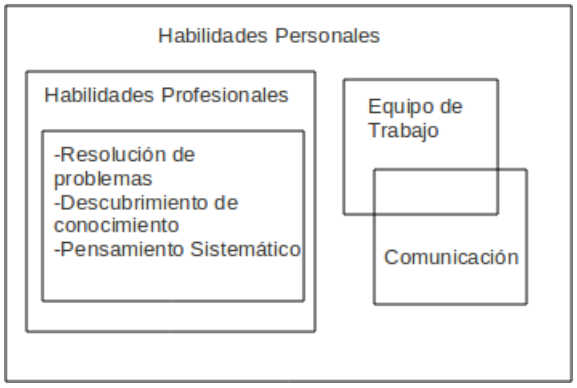

Figura 1.3: Relación entre las habilidades personales, profesionales e interpersonales fuente:[3]

referencia a las habilidades necesarias para formar, operar, fortalecer y liderar un equipo con habilidades específicas de un equipo de trabajo técnico. La comunicación se compone de habilidades para idear estrategias de comunicación y aquellas que utilizan los medios orales, escritos, electrónicos y gráficos y en el caso colombiano el uso del idioma Inglés. La Figura 1.3 muestra la relación entre las habilidades de nivel 2 (personales y profesionales) y nivel 3 (interpersonales).

Habiidades CDIO

Habilidades necesarias parea concebir, diseñar, implementar y operar sistemas en el contexto social y empresarial; estos cuatro componentes son necesarios para que los egresados de las carreras de ingeniería eléctrica y electrónica sean capaces de absorber los conocimientos que las nuevas tecnologías proporcionan, adaptarlos a la situación tecnológica y al contexto social del país para generar productos que resuelvan necesidades locales. Para satisfacer una necesidad de la sociedad es necesario conocer la dinámica empresarial, los principios que la rigen y como se debe actuar en una empresa de cualquier tipo y tamaño.

1.1.3. Implementación del Plan de Estudios CDIO

La Figura 1.4 muestra los componentes que deben ser especificados para implementar el plan de estudios CDIO al currículo de las asignaturas del área de electrónica digital; en primer lugar se encuentran los resultados esperados del proceso de aprendizaje, esto es, ¿Qué deben saber y qué deben ser capaces de hacer los estudiantes al final del curso? Para contestar a esta pregunta es necesario definir las **habilidades** que serán reforzadas o desarrolladas y los *objetivos* de cada asignatura.

Figura 1.4: Objetivos, actividades, y evaluación:

Para alcanzar los objetivos definidos en el primer paso, es necesario generar una serie de **actividades** que le permitan al estudiante retener nuevos conocimientos y habilidades y desarrollar las competencias deseadas; el número de actividades debe ser tal que cubran todas las habilidades que se quieran desarrollar o reforzar. Finalmente, se deben desarrollar métodos de evaluación que permitan conocer el nivel de competencia de los estudiantes, y de esta forma ajustar las actividades para obtener los resultados esperados.

1.2. Definición e Identificación de las Habilidades CDIO

El primer paso en la implementación del plan de estudios CDIO es definir e identificar las habilidades requeridas en una determinada área del plan de estudios, en este caso en las asignaturas del área de electrónica digital. En el DIEE de la UNAL el área de electrónica digital esta compuesta por tres asignaturas para la carrera de ingeniería electrónica: Electrónica Digital 1, Electrónica Digital 2 y Sistemas Embebidos; para la carrera de ingeniería eléctrica está compuesta por Electrónica Digital 1 (la misma en las dos carreras) únicamente.

1.2.1. Introducir, Enseñar y Usar

Para transladar esta lista de habilidades a objetivos de aprendizaje es necesario determinar el grado de competencia que se espera que el profesional adquiera en cada una de las asignaturas; por supuesto, algunas de estas habilidades no pueden obtenerse solo en una asignatura y es necesario que todo el plan académico contribuya a generarla, lo que requiere un consenso del personal académico. Los niveles de competencia seleccionados para indicar el grado en que debe ser apropiada una determinada habilidad son:

1. Introducir: Introduce pero no evalúa.
2. Enseñar: Enseña y evalúa.
3. Utilizar: Utiliza, puede ser evaluado o no.

1.2.2. Habilidades CDIO

A continuación se listan las habilidades CDIO que deben desarrollarse o reforzarse en el área de electrónica digital, algunas de las habilidades como la comunicación oral y escrita en inglés es común a la mayoría de las asignaturas, mientras que otras como la integración software - hardware es exclusiva de esta línea.

2 **Aptitudes personales y profesionales**

1 Razonamiento analítico y resolución de problemas
   1) Identificación y formulación del problema.
   2) Modelamiento.
   3) Solución y recomendación.

2 Experimentación investigación y descubrimiento de conocimiento
   1) Formulación de hipótesis.
   2) Investigación experimental.

3 Pensamiento sistemático
   1) Pensamiento global.
   2) Surgimiento e interacciones.

4 Pensamiento crítico y creativo y habilidades y actitudes personales
   1) Pensamiento creativo.
   2) Pensamiento crítico.
   3) Toma de conciencia de conocimientos propios y metaconocimiento.
   4) Aprendizaje permanente y educar a otros.

5 Ética, responsabilidad profesional, equidad y otros valores personales.
   1) Ética, integridad y responsabilidad social.
   2) Comportamiento profesional y responsabilidad.
   3) Confianza y lealtad.

Las aptitudes personales y profesionales que se desean resaltar en esta metdología son:

- Pensamiento sistemático: Los conocimientos adquiridos en esta asignatura son utilizados para contribuir a una solucion a un problema de la sociedad. Esta contribución está representada en un sistema digital que utiliza tecnología de punta y para su elaboración se aplicó una metodología de diseño que optimiza recursos económicos proporcionando la funcionalidad requerida.
- Toma de conciencia de conocimientos propios: El estudiante es el responsable de adquirir el conocimiento con la guía del profesor; por lo tanto, debe ser capaz de identificar temas que no ha asimilado completamente y debe realizar actividades que ayuden en la asimilación de nuevos conocimientos.
- Aprendizaje permanente y educar a otros: El aprendizaje debe ser visto por el estudiante como un proceso continuo que no termina al momento de presentar las evaluaciones, o al finalizar los ciclos de formación; la metodología en estos cursos debe forzar al estudiante a buscar fuentes de información para mejorar la asimilación de conocimientos. El trabajo en equipo ayuda a que esta búsqueda sea compartida, de tal forma que el aprendizaje sea colectivo.

Estas habilidads le permitirán al futuro profesional asimilar cambios en las tecnologías, aplicando técnicas de auto-aprendizaje, lo que hace innecesario que un profesional se vea obligado a tomar cursos de extensión para entender un cambio o una nueva característica de la tecnología que utiliza. A nuestro modo de ver los ingenieros deben estar en la capacidad de adaptarse a estos cambios por si mismos.

3 **Habilidades interpersonales, trabajo en equipo y comunicación**

1 Equipo de trabajo
   1) Formar grupos efectivos.
   2) Equipo de liderazgo.
   3) Equipo técnico y multi-disciplinario.

2 Comunicaciones estructuradas
   1) Estrategia de comunicación.
   2) Estructura de la comunicación.
   3) Comunicación escrita.
   4) Comunicación electrónica.
   5) Presentación oral.

3 Comunicación en idioma extranjero
   1) Inglés.

4 Comunicaciones informales: Relacionarse con los demás
   1) Preguntar, escuchar y dialogar.
   2) Negociación, compromiso y resolución de conflictos.
   3) Establecimiento de conexiones.

Con las habilidades interpersonales se busca formar profesionales que puedan formar grupos de trabajo eficientes, identificando su papel dentro del mismo, facilitando el desarrollo de las actividades de los demás miembros y buscar soluciones conjuntas al problema a resolver y a problemas que surjan del desarrollo del proyecto. Esto es muy importante para el desarrollo de dispositivos digitales, ya que para su elaboración intervienen profesionales de diferentes áreas: diseñadores industriales, programadores, desarrolladores de hardware, personal de mercadeo, ventas y soporte. Por esto, es de suma importancia aprender a utilizar las diferentes formas de comunicación identificando cual es la más eficiente en un determinado caso; el uso de plataformas que permita a miembros de un equipo que se encuentran en diferentes puntos geográficos intercambiar información y discutir temas propios del diseño simplifica el trabajo y ahorra tiempo y dinero.

Por otro lado, es importante que el estudiante adquiera habilidades que le permitan documentar de forma adecuada los procesos de diseño y que el material que desarrolle pueda ser utilizado por otros miembros del equipo de forma rápida; pora esto, es importante utilizar herramientas que faciliten la generación de documentos que permitan el acceso a múltiples usuarios.

Las comunicaciones informales son la mayor fuente de información en los proyectos abiertos, usuarios con un determinado problema escriben a una lista de correo o a canales *irc* (internet relay chat)[4] que tiene como miembros a usuarios más avanzados que pueden contestar todo tipo de preguntas, y al mismo tiempo proporcionan una base de datos con todas las comunicaciones anteriores permitiendo la búsqueda de conversaciones relacionadas con un problema determinado. Estas listas de correo y canales irc también pueden ser utilizados para establecer relaciones profesionales que pueden terminar en nuevos proyectos, ofertas de empleo o colaboración.

[4]protocolo de comunicaciones en tiempo real basado en texto, permite conversaciones entre un grupo de personas; todos los usuarios que se encuentran en un canal pueden comunicarse entre sí

4 **Concebir, diseñar, implementar y operar sistemas en el contexto social y empresarial**

1 Contexto externo, social, económico y ambiental
   1) Rol y responsabilidad de los ingenieros.
   2) Impacto sobre la sociedad y el medio ambiente.
   3) Cuestiones y valores actuales.
   4) Sostenibilidad y necesidad de un desarrollo sostenible.

2 Empresa y contexto empresarial
   1) Interesados en la empresa, metas y objetivos.
   2) Espíritu empresarial técnico.
   3) Trabajo en organizaciones.
   4) Finanzas y economía de los proyectos de ingeniería

3 Concepción y administración de sistemas en ingeniería.
   1) Entender las necesidades y establecer las metas.
   2) Definir la función, concepto y arquitectura.

4 Diseño
   1) Proceso de Diseño.
   2) Fases del proceso de Diseño y enfoques.
   3) Utilización de conocimiento científico en el diseño.
   4) Diseño específico.
   5) Diseño multi-disciplinario.

5 Implementación
   1) Proceso de fabricación hardware.
   2) Proceso de Implementación de software.
   3) Integración software - hardware.
   4) Pruebas, verificación, validación y certificación.

6 Liderar esfuerzos en ingeniería
   1) Pensar creativamente e imaginar posibilidades.
   2) Definir la solución.
   3) Crear nuevas formas de solución.
   4) Construir y liderar una organización y una organización extendida.
   5) Planear y administrar un proyecto hasta su finalización.
   6) Innovar - la concepción, diseño e introducción de nuevos bienes y servicios.
   7) Innovar - el desarrollo de nuevos dispositivos, materiales o procesos que permitan nuevos bienes y servicios.

7 Emprendimiento en ingeniería
   1) Creación, formulación y organización de una empresa.
   2) Desarrollo del plan de negocios.
   3) Finanzas y capitalización.
   4) Mercadeo de productos innovadores.
   5) Cencepción de productos y servicios alrededor de nuevas tecnologías.
   6) Sistema de innovación, redes, infraestructura y servicios.

7) Construyendo el equipo e iniciando el proceso de ingeniería.
8) Manejo de la propiedad intelectual.

Este grupo de habilidades son las que marcan diferencia con las otras asignaturas del plan de estudio ya que como se mencionó anteriormente, la mayoría de los cursos no contemplan la implementación de sistemas reales donde se apliquen los conocimientos que se intentan transmitir. Debido a esto, es importante recalcar el papel de los ingenieros en la sociedad, la facultad de ingeniería de la universidad nacional tiene como parte de sus objetivos:

- Crear y asimilar críticamente el conocimiento en los campos avanzados de las ciencias, la técnica, la tecnología, el arte y la filosofía.
- Estudiar y analizar los problemas nacionales y proponer, con independencia, formulaciones y soluciones pertinentes, convirtiéndose así en conciencia crítica de la nación.
- Hacer partícipes de los beneficios de su actividad académica e investigativa a los sectores sociales que conforman la nación Colombiana.

Como establecimos en el primer capítulo de este trabajo, el problema actual de la industria electrónica (relacionada con el diseño digital) en Colombia es el atraso y desactualización de las metodologías de diseño y fabricación, originado por la falta de personal caificado en el manejo de nuevas tecnologías y metodologías. Por esta razón, es importante que los estudiantes conozcan la realidad de este sector y sean concientes de su responsabilidad en la solución de sus problemas.

Como estudiantes pueden contribuir a difundir los resultados de su actividad académica proporcionando información que pueda ser utilizada por quien este interesado en adquirir conocimientos relacionados con el diseño digital; por esto, una de las actividades que se desarrollan en estas asignaturas es la documentación del proceso de diseño de un problema específico que deben resolver grupos de trabajo, esta metodología y esta información se almacena en el portal público *linuxencaja* y puede ser utilizada por quien esté interesado.

Una parte vital en el proceso de enseñanza es el uso de metodologías de diseño para la creación de soluciones a problemas específicos, por esta razón se trabaja con la misma metodología en los tres cursos y se explican los beneficios de usar todas sus etapas; esto con el objetivo de crear buenos habitos de diseño en elos estudiantes. A diferencia de otras asignaturas en esta línea es necesario implementar físicamente las soluciones propuestas, se resalta que la simualción es un paso importante del proceso de diseño pero no es punto final. El uso de plataformas con recursos limitados para la implementación física hace que los estudiantes optimicen la arquitectura de sus diseños y aprendan a buscar alternativas más económicas pero que cumplan con los requerimientos del sistema. El problema a solucionar es formulado por el equipo de trabajo y ellos deben proporcionar las especificaciones del sistema que resolverá este problema, los estudiantes poseen libertad total en la funcionalidad del mismo, lo que ayuda a la creación de habilidades en la definición de funciones y arquitecturas.

Con esto se espera que el estudiante sea conciente de la situación del país y que sienta la necesidad de crear nuevos productos que ayuden a solucionar un problema específico y de paso ayudar a la creación de empleo creando y comercializando nuevos productos diseñados localmente.

1.2.3. Competencias de las Habilidades CDIO 2 y 3

La tabla 1.1 muestra las competencias IEU para las *aptitudes personales y profesionales* de las tres asignaturas del área de electrónica digital.

Competencias de las habilidades CDIO nivel 2 y 3			
APTITUDES PERSONALES Y PROFESIONALES	Nivel 1		
	E. Dig1	E. Dig2	Sist. Emb.
Planteamiento y resolución de problemas de ingeniería	EU		
1 Identificación y formulación del problema	EU		
2 Modelamiento	EU		
3 Solución y recomendación	EU		
Experimentación y descubrimiento de conocimiento	U		
4 Formulación de hipótesis	U		
5 Investigación experimental	U		
Pensamiento sistemático	EU		
6 Pensamiento global	U		
7 Surgimiento e interacciones	U		
Habilidades y actitudes personales	U		
8 Pensamiento creativo	IEU		
9 Pensamiento crítico	IEU		
10 Toma de conciencia de conocimientos propios	IEU		
11 Curiosidad y aprendizaje permanente ***Habilidades y actitudes profesionales***	U		
12 Ética profesional, integridad, responsabilidad	U		
13 Comportamiento profesional	U		
39 Confianza y lealtad	IEU		
HABILIDADES INTERPERSONALES	Nivel 1		
	E. Digital1	E. Digital1	Sist. Emb.
Equipo de trabajo	EU		
14 Formar grupos efectivos	EU	U	U
15 Equipo de liderazgo	EU	U	U
40 Equipo Técnico y Multi-disciplinario	EU	U	U
Comunicaciones estructuradas	EU		
16 Estrategia de comunicación	EU	U	U
17 Estructura de la comunicación	EU	U	U
18 Comunicación Escrita	EU	U	U
19 Comunicación Electrónica	EU	U	U
20 Presentación Oral	EU	U	U
Comunicación en Idioma Extranjero	U		
21 Inglés	U		
Comunicaciones Informales: Relacionarse con los demás	U		
41 Preguntar, Escuchar y Dialogar	EU	U	U
42 Negociación, compromiso y resolución de conflictos	EU	U	U
43 Establecimiento de conexiones	IEU	U	U

Cuadro 1.1: Competencias para los niveles 2 y 3 CDIO

1.2.4. Competencias de las Habilidades C.D.I.O. Sistemas en el contexto Empresarial, Social y Ambiental - Innovación

La tabla 1.2 muestra las competencias IEU para las *C.D.I.O. Sistemas en el contexto Empresarial, Social y Ambiental - Innovación* de las tres asignaturas del área de electrónica digital.

HABILIDADES CDIO	Nivel 1		
	E. Digital1	E. Digital1	Sist. Emb.
Contexto Externo, Social, Económico y Ambiental	IEU		
22 Rol y responsabilidad de los Ingenieros	IEU		
23 Impacto sobre la sociedad y el medio ambiente	IEU		
24 Cuestiones y valores actuales	IEU		
44 Sostenibilidad y necesidad de un desarrollo sostenible	IE	IE	IE
Empresa y contexto empresarial	EU		
25 Interesados en la empresa, metas y objetivos	I		
26 Espíritu Empresarial Técnico	I		
27 Trabajo exitoso en organizaciones	I		
45 Finanzas y Economía de los Proyectos de Ingeniería	IE	IE	IE
Concepción y Administración de Sistemas en Ingeniería.	IEU		
28 Entender las necesidades y establecer las metas	IEU	EU	U
29 Definir la función, concepto y arquitectura	IEU	EU	U
Diseño	IEU		
30 Proceso de Diseño	IEU	EU	U
31 Fases del proceso de Diseño y enfoques	IEU	EU	U
32 Utilización de conocimiento científico en el diseño	IEU	EU	U
33 Diseño específico	IEU	EU	U
34 Diseño multi-disciplinario	I	E	U
Implementación	EU		
35 Proceso de fabricación Hardware	IE	EU	U
36 Proceso de Implementación de Software	I	EU	U
37 Integración Software - Hardware	I	EU	U
38 Pruebas, verificación, validación y certificación	IE	EU	U

Cuadro 1.2: Competencias para CDIO

Como puede verse en la tabla 1.2 el componente distintivo del plan de estudios propuesto es el nivel de competencias de las habilidades relacionadas con diseño e implementación; en casi todas ellas se utilizan estas habilidades, lo que significa que se realizarán actividades que impliquen diseño e implementación de sistemas digitales utilizando tecnología de punta y técnicas de diseño y fabricación modernas; lo que es imprescindible si se espera que nuestros profesionales lideren esfuerzos que deriven en la creación de empresas y de nuevos productos bienes y servicios.

1.3. Integración de las Habilidades CDIO al Plan de Estudios

1.3.1. Metodología de diseño

El proceso de diseño de un Sistema Embebido comienza con la *especificación del sistema* (ver figura 1.5), en este punto se describe la funcionalidad y se definen las restricciones físicas, eléctricas y económicas. Esta especificación debe ser muy general y no deben existir dependencias (tecnológicas, metodológicas) de ningún tipo, se suele utilizar lenguajes de alto nivel, como UML, C++ para realizar la descripción. La especificación puede ser verificada a través de una serie de pasos de análisis cuyo objetivo es determinar la validez de los algorítmos seleccionados, por ejemplo, determinar si el algoritmo siempre termina, los resultados satisfacen las especificaciones. Desde el punto de vista de la re-utilización, algunas partes del funcionamiento global deben tomarse de una librería de algorítmos existentes.

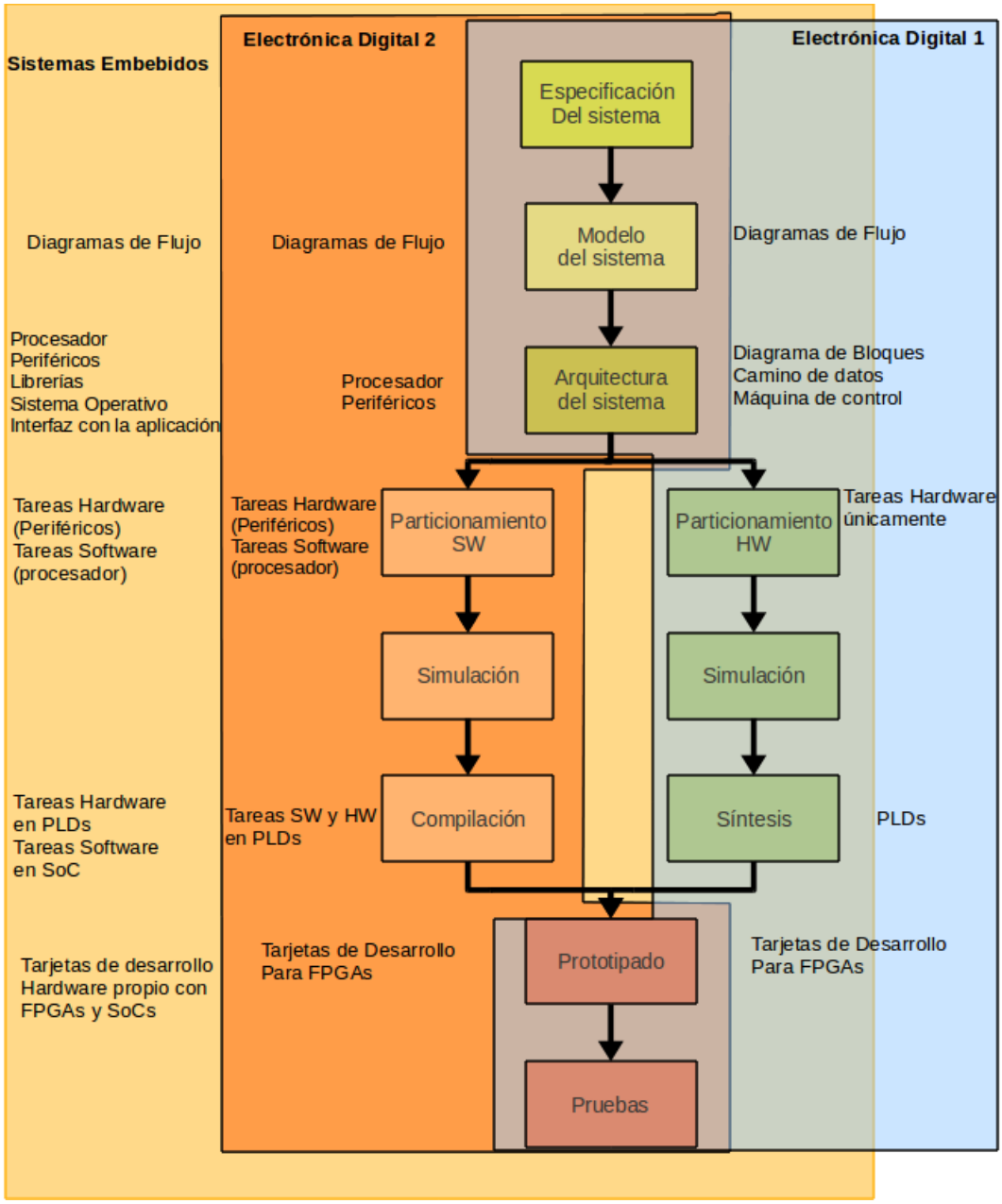

Figura 1.5: Metodología de Diseño para el área de Sistemas Digitales

Una vez definidas las especificaciones del sistema se debe realizar un modelamiento que permita extraer de estas la funcionalidad. El modelamiento es crucial en el diseño ya que de él depende el paso existoso de la especificación a la implementación. Es importante definir que modelo matemático debe soportar el entorno de diseño. Los modelos más utilizados son: máquinas de estados finitos, diagramas de flujos de datos, sistemas de eventos discretos y redes de petri. Cada modelo posee propiedades matemáticas que

pueden explotarse de forma eficiente para responder preguntas sobre la funcionalidad del sistema sin llevar a cabo dispendiosas tareas de verificación. Todo modelo obtenido debe ser verificado para comprobar que cumple con las restricciones del sistema.

Una vez se ha obtenido el modelo del sistema se procede a determinar su *arquitectura*, esto es, el número y tipo de componentes y su inter-conexión. Este paso no es más que una exploración del espacio de diseño en búsqueda de soluciones que permitan la implementación de una funcionalidad dada, y puede realizarse con varios criterios en mente: costos, confiabilidad, viabilidad comercial. Las opciones de implementacion que se estudiarán son:

- Componente HW y SW integrado en un dispositivo semiconductor (SoC): En la actualidad existen muchas compañías que fabrican procesadores de 32 bits integrados a una gran variedad de periféricos, lo cual simplifica el diseño y reduce costos. Este tipo de implementación es muy popular en los dispositivos de consumo masivo (reproductores de MP3, consolas de juego, etc), debido a los grandes niveles de producción resulta más económico contar con un dispositivo que integre el mayor número de funcionalidades, esto disminuye el costo de componentes, el área de circuito impreso.
- Componente SW en un SoC y componente HW en un dispositivo lógico programable (PLD): Cuando no existen en el mercado un SoC con la cantidad de periféricos requerida para una determinada aplicación, o con una funcionalidad específica, es necesario recurrir a la utilización de dispositivos comerciales que implementen dicha operación; en algunas ocaciones el periférico puede realizar funciones poco comúnes y no se proporciona comercialmente, por lo que es necesario implementar esta funcionalidades en una FPGA y conectarla al SoC de tal forma que este la vea como uno más de sus periféricos.
- Componente SW y HW en una FPGA: Esta es tal vez la opción más flexible, pero la de menor desempeño, ya que al utilizar los recursos lógicos de la FPGA para la implementación del procesador (softcore) la longitud de los caminos de interconexión entre los bloques lógicos aumentan el retardo de las señales, lo cual disminuye la máxima velocidad de funcionamiento. Los procesadores *softcore* más populares en la actualidad son:
  - Microblaze y Picoblaze de Xilinx.
  - NIOS de Altera.
  - Leon de Gaisler Research.
  - LatticeMico32 (LM32) de Lattice Semiconductors.
  - OpenRisc de opencores.

  De los anteriores procesadores el Leon, el LM32 y el Openrisc proporcionan el código fuente en un lenguaje de descripción de hardware, los de las empresas Xilinx y Altera entregan un código a nivel de compuertas que no permite su estudio y modificación; por esats razones y debido a que no se necesita una FPGA con mucha capacidad lógica en estos cursos se trabajará con el LM32 de Lattice.

Utilizando como base la arquitectura obtenida en el paso anterior las tareas del modelo del sistemas son mapeadas dentro de los componentes. Esto es, asignación de funciones a los componentes de la arquitectura. Existen dos opciones a la hora de implementar las tareas o procesos:

1 Implementación software: La tarea se va a ejecutar en un procesador ya sea el del SoC o el softcore LM32.

2 Implementación hardware: La tarea se va a ejecutar en un sistema digital dedicado implementado en un PLD.

Para cumplir las especificaciones del sistema algunas tareas deben ser implementadas en hardware, esto con el fín de no ocupar al procesador en tareas cíclicas, un ejemplo típico de estas tareas es la generación de bases de tiempos. La decisión de que tareas se implementan en SW y que tareas se implementan en HW recibe el nombre de *particionamiento*, esta selección es fuertemente dependiente de restricciones económicas y temporales.

Las tareas software deben compartir los recursos que existan en el sistema (procesador y memoria), por lo tanto se deben hacer decisiones sobre el orden de ejecución y la prioridad de estas. Este proceso recibe el nombre de *planificación*. En este punto del diseño el modelo debe incluir información sobre el mapeo, el particionamiento y la planificación del sistema.

Las siguientes fases corresponden a la implementación del modelo, para esto las tareas hardware deben ser llevadas al dispositivo elegido (ASIC o FPGA) y se debe obtener el ″ejecutable″ de las tareas software, este proceso recibe el nombre de *síntesis* HW y SW respectivamente, así mismo se deben sintetizar los mecanismos de comunicación.

El proceso de prototipado consiste en la realización física del sistema, esto es, la fabricación de la placa de circuito impreso, el montaje de los componentes y la fabricación de la caja. Finalmente, el sistema físico debe someterse a pruebas para verificar que se cumplen con las especificaciones iniciales y con las normas propias de su aplicación. Como puede verse en el flujo de diseño existen realimentaciones que permiten depurar el resultado de pasos anteriores en el caso de no cumplirse con las especificaciones iniciales.

Esta metodología de diseño se utilizará en todos los cursos de la línea, pero se utilizarán diferentes opciones de implementación en los diferentes cursos, como se puede apreciar en la figura 1.5; en electrónica digital 1 solo se implementarán tareas hardware en una FPGA, esto con el fín de enseñar al estudiante los conceptos básicos de la electrónica digital, y el uso de máquinas de estado algorítmicas para la implementación de tareas hardware; en electrónica digital 2 se estudiará la arquitectura de los SoC (utilizando el LM32 de lattice) se utilizarán las herramientas de compilación GNU para escribir las tareas software y se crearán periféricos que implementen tareas hardware y finalmente se estudiarán las diferentes formas de comunicación entre las tareas hardware y software; finalmente en el curso sistemas embebidos, se utilizará un SoC comercial junto con la cadena de herramientas GNU y el sistema operativo Linux para estudiar la arquitectura y programación de los sistemas embebidos modernos.

1.3.2. Objetivo General

Generar en el estudiante las habilidades necesarias para concebir, diseñar, implementar y operar sistemas digitales complejos que satisfagan necesidades de la sociedad y proporcionar un canal para la transferencia de tecnología y conocimiento a la Industria Colombiana.

Electrónica Digital 1

Concebir y definir las especificaciones y requerimientos de un sistema digital, modelar su funcionamiento, y realizar la implementación siguiendo la metodología de diseño de sistemas embebidos utilizando únicamente tareas hardware.

Electrónica Digital 2

Concebir, definir las especificaciones, modelar, diseñar un sistema digital siguiendo la metodología de diseño de sistemas embebidos y realizar su implementación óptima utilizando tareas hardware (implementadas en un PLD) y tareas software (que se ejecutan en un procesador).

Sistemas Embebidos

Concebir, diseñar, e implementar un sistema digital complejo utilizando la metodología de diseño de sistemas embebidos, un SoC comercial y un sistema operativo para su implementación.

1.3.3. Objetivos Específicos

1.3.4. Ojbetivos comúnes

- Identificar las especificaciones funcionales del sistema, su arquitectura de alto nivel y definir su descomposición en elementos
- Explicar las actividades en las etapas del proceso de diseño,
- Desarrollar el pensamiento sistémico.
- Modelar funcionalmente sistemas digitales.
- Diseñar pruebas para comprobar el correcto funcionamiento de los sistemas implementados.
- Leer y entender material técnico escrito en inglés.
- Implementar un sistemas embebido (hardware o hardware/software) para cumplir una tarea determinada que cumpla con una necesidad real (Obtener e interpretar las necesidades del consumidor) utilizando técnicas, herramientas y procesos adecuados.
- Estudiar y aplicar el concepto de la re-utilización de código.
- Desarrollar trabajo en equipo incluyendo presentaciones, describiendo los diversos roles y responsabilidades.
- Documentar los diseños realizados para crear una base de datos que contribuya a la difusión del conocimiento adquirido.

Electrónica Digital 1

- Estudiar las fases de la metodología de diseño para sistemas embebidos.
- Estudiar los dominios de diseño estructural, funcional y físico.
- Estudiar los lenguajes de descripción de hardware.
- Estudiar los componentes básicos de la lógica combinatoria y secuencial.
- Estudio de las máquinas de estado algorítmicas.

Electrónica Digital 2

- Estudiar los requisitos para un particionamiento Hardware / Software óptimo.
- Estudiar la arquitectura de un procesador, micro-arquitectura, set de instrucciones, interrupciones, direccionamiento, memorias.
- Estudiar el proceso de implementación de tareas software.
- Estudiar la integración Software-Hardware.
- Diseñar pruebas para comprobar el correcto funcionamiento de los sistemas implementados.

Sistemas Embebidos

- Realizar aplicaciones que requieran diseño multi-disciplinario.
- Estudiar y realizar el proceso de fabricación hardware.
- Estudiar el principio básico de los sistemas operativos.
- Describir la integración de software en hardware electrónico
- Entender diagramas de circuitos electrónicos de sistemas digitales, identifcar sus componentes y su función.
- Estudiar diseños software y hardware existentes para entender su funcionamiento, arquitectura y adquirir experiencia en el diseño.
- Hacer parte de listas de discusión de temas técnicos que usen el inglés como lenguaje.
- Utilizar medios electrónicos para documentar procesos de diseño.

Con estos objetivos se busca que al cursar todas las asignaturas del área el estudiante adquiera los conocimientos que se resumen en la figura 1.6.

1.3.5. Metodología

Todas las actividades que se realizarán en estos cursos están encamindas a generar habilidades necesarias para concebir, diseñar, e implementar sistemas digitales complejos, y están articuladas alrededor de una única metodología de diseño. Los tres cursos tienen un carácter teórico-práctico, el componente teórico tratará los diferentes temas de forma general, con el fín de no crear dependencia con las herramientas utilizadas (lo que permitirá realizar actualizaciónes de forma fácil). En el componente práctico se tratarán temas específicos de manejo de las herramientas (lenguajes de descripción de hardware, lenguajes de programación, manejo de plataformas de desarrollo) y como estas se relacionan con la metodología de diseño utilizada.

El estudiante debe estudiar, profundizar y comprobar algunos temas tratados en clase y debe leer previamente la documentación que se encuentra disponible en el sitio web de los cursos. Adicionalmente, debe formar grupos de trabajo para realizar actividades a lo largo del semestre.

Durante el semestre se trabajará para definir las especificaciones, diseñar e implementar un dispositivo que resuelva una determinada necesidad (con la complejidad adecuada para cada curso), en la sesión

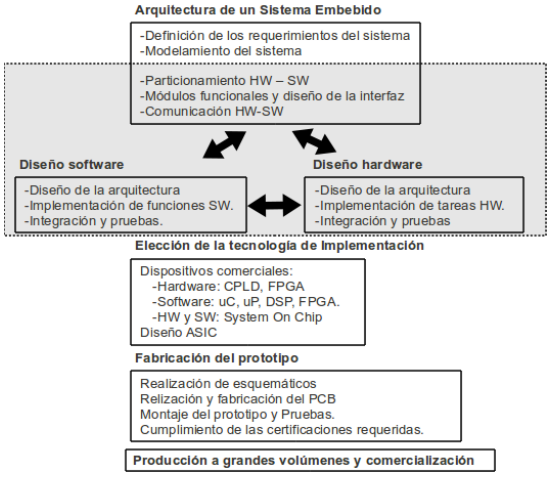

Figura 1.6: Educación de sistemas embebidos. Tomada de:[4] y modificada

teórica se tratarán aspectos relacionados con la concepción, diseño, identificación y definición de las funciones de los componentes del sistema, mientras que en el laboratorio se tratarán temas relacionados con la implementación de dichos componentes sobre PLDs o SoC. Se deben realizar presentaciones del avance, indicando las razones que se tuvieron en cuenta en cada decisión y somo se resolvieron los problemas encontrados, todo este proceso debe documentarse en la wiki del portal *linuxencaja*, esto último para formar un banco de proyectos que pueda ser utilizado como referencia por quien este interesado.

SIE: Plataforma abierta para el desarrollo de sistemas embebidos

En el mercado existe una gran variedad de plataformas que pueden ser utilizadas en el estudio de sistemas embebidos, sin embargo, no todas son adecuadas para la implementación del método que proponemos ya que la plataforma que se utilice debe proporcionar toda la información necesaria para poder entenderla, programarla, replicarla y modificarla; esto con el fin de proporcionar al estudiante una herramienta que pueda ser utilizada en el desarrollo de nuevos productos comerciales, para que se cumplan estas condiciones se requiere: acceso a los esquemáticos y a los archivos de fabricación del PCB con posibilidad de modificación; acceso a la documentación completa del proceso de fabricación; acceso a la cadena de producción; utilización de herramientas abiertas para su programación; un PLD para la implementación de tareas HW; un procesador para la implementación de tareas SW; un canal de comunicación entre el procesador y el PLD; y una comunidad que desarrolle aplicaciones para dicha plataforma y que proporcione medios para el intercambio de información a través de listas de correo y wikis.

Después de una búsqueda minuciosa no se encontraron plataformas que cumplieran con estas condiciones, en especial con las relacionadas con el proceso de diseño y de producción, esto es normal, ya que la mayoría de las empresas no quieren que se fabriquen sus plataformas y los proyectos individuales no poseen la infraestructura necesaria para la producción masiva. Por este motivo, se decidió crear una plataforma que cumpliera con los requerimientos (plataforma *SIE*), pudiera ser utilizada en los tres cursos del área y e convirtiera en una herramienta más para la difusión de los conocimientos adquiridos en este estudio a quien se encuentre interesado.

La Figura 1.7 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.

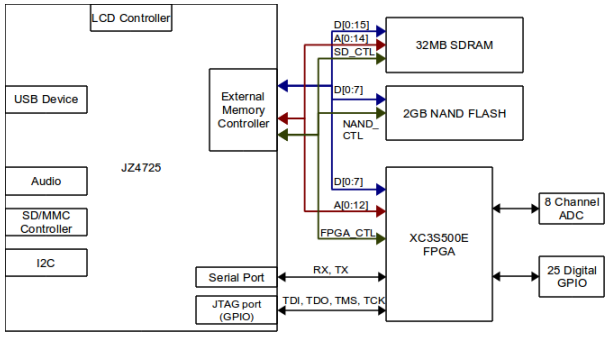

Figura 1.7: Estructura de la plataforma de desarrollo SIE

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.

1.3.6. Integración de SIE con los cursos de la línea de electrónica digital

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, 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 portal *linuxencaja*, 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 la herramienta abierta Kicad) con la funcionalidad que se le desea dar a la plataforma SIE.

Electrónica digital 1

En el primer curso del área de diseño digital en la UNAL se realiza el estudio, diseño e implementación de máquinas de estado algorítmicas utilizando la metodología de diseño presentada anterior-

mente y la herramienta gratuita de Xilinx *Webpack*, los estudiantes implementarán sus diseños utilizando lenguajes de descripción de hardware (VHDL, verilog), como resultado de este proceso se obtendrán 3 archivos uno con extensión **.bit** que se utiliza para configurar a la FPGA con la funcionalidad deseada; un archivo con extensión **pad.csv** que indica la función asignada por el diseñador a todos los pines de la FPGA; y un archivo con extensión **.vcd** que contiene los resultados de la simulación del sistema [5]. SIE proporciona un canal de comunicación entre el procesador y el puerto JTAG de la FPGA que puede ser utilizado para:

- Configuración: Carga del archivo con extensión **.bit** con la funcionalidad deseada a la FPGA; este archivo puede ser transferido a la FPGA utilizando el cable usb y el protocolo de comunicaciones ssh, cuando este archivo de configiración se encuentre en el sistema de archivos de SIE se puede utilizar la aplicación *xc3sprog* para configurar la FPGA.
- Prueba a baja frecuencia: Como mencionamos anteriormente, el protocolo JTAG permite la aplicación de vectores de prueba a un dispositivo semiconductor y la recolección de la respuesta a estos estímulos utilizando 4 señales (TDI, TDO, TMS, TCK).

Una aplicación (propia y abierta) que tiene como entrada los archivos que contienen la información de los pines (pad.csv) y los resultados de la simulación (ver figura 1.8) extrae la información necesaria para aplicar los vectores de prueba al circuito implementado en la FPGA y capturar su respuesta a estos estímulos [6], utilizando la instrucción INTEST del protocolo JTAG. Los resultados de esta prueba son graficados en el LCD de SIE y pueden ser exportados a un archivo tipo imágen; esta herramienta puede verse como una combinación de un analizador lógico y un generador de vectores de prueba de bajo costo. Con esto, se proporcionan las herramientas necesarias para realizar todas las etapas del flujo de diseño con solo una plataforma.

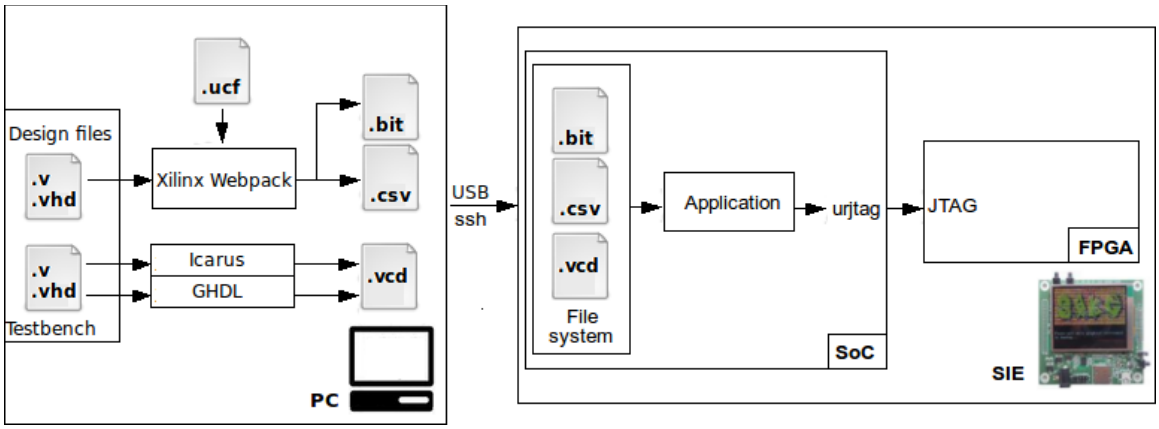

Figura 1.8: Flujo de diseño hardware

De forma intensional SIE no cuenta con memorias, displays de 7 segmentoso conectores conectados a la FPGA; experiencias con el manejo de las plataformas de desarrollo que poseen una gran cantidad de periféricos indican que los estudiantes no adquieren habilidades necesarias para la lectura de esquemáticos ni de hojas de especificaciones de componentes al utilizarlas, ya que normalmente se suministran ejemplos funcionales que son utilizados por los estudiantes; la falta de este tipo de recursos en SIE obliga a los

[5] Este archivo lo generan las herramientas de simulación abiertas *icarus* y *ghdl*
[6] para el control del puerto JTAG se modificó la herramienta *urjtag*

estudiantes a investigar sobre la adecuada conexión de estos componentes y a la construcción de una tarjeta hija que implemente la funcionalidad deseada.

Electrónica digital 2

Como se mencionó anteriormente, en este curso se busca que el estudiante entienda la diferencia entre tareas hardware y software, y los canales de comunicación entre ellas; para esto se implementarán las tareas software en el procesador *softcore* LM32 de Lattice (o cualquiera con el código fuente disponible); para esto, es necesario estudiar la arquitectura de la unidad de procesamiento, su set de instrucciones, manejo de interrupciones, comunicación con las memorias de datos y de instrucciones y su forma de programación. El código fuente facilita el estudio de las operaciones internas del procesador al permitir el seguimiento de señales específicas, a manera de ejemplo en la figura 1.9 se muestra el acceso a una memoria externa para tipos de datos *unsigned char*

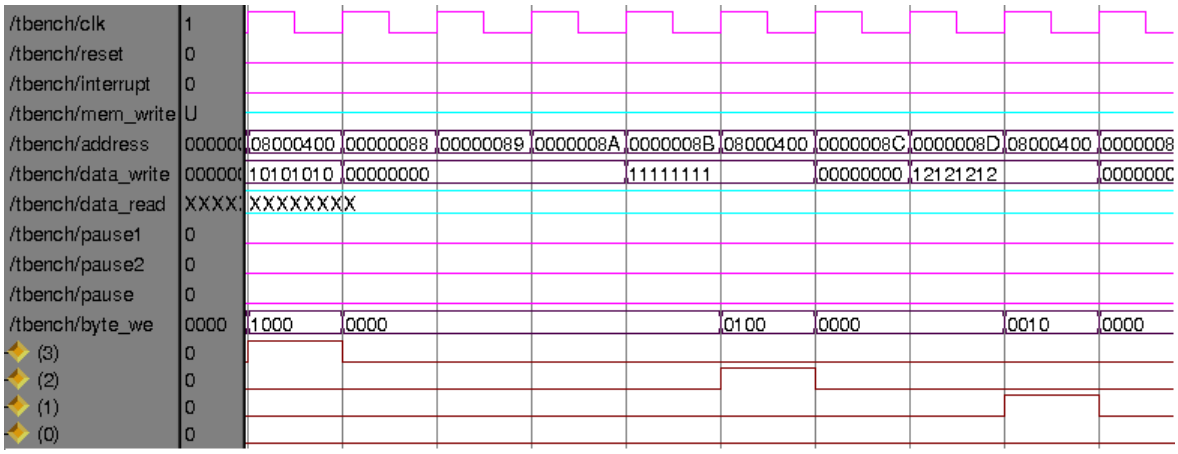

Figura 1.9: Simualción del acceso de datos a memaria externa en un procesador mips

Este tipo de simualciones permiten entender perfectamente conceptos relacionados con el funcionamiento del sistema (diferencia de tipos de datos en este ejemplo) lo que no es posible realizar si se utilizan dispositivos comerciales en el aprendizaje de la arquitectura de computadores ya que los simuladores existentes se enfocan en el contenido del banco de registros y en la memoria y no pueden mostrar (hasta el momento) las señales internas. Pero, ¿Porqué no implementar un procesador propio para la realización de este estudio? esta metodologia se utilizó en el pasado en muchos cursos de arquitectura de computadores y es utilizada en la actualidad, su principal desventaja radica en la falta de herramientas para realizar la programación de estas nuevas arquitecturas, por lo que es necesario programarlas en lenguaje de máquina lo que resulta ser muy tedioso aún para aplicaciones sencillas.

La metodología que proponemos aquí permite utilizar la cadena de herramientas GNU existente para el procesador LM32, esta cadena de herramientas permite trabajar con lenguaje ensamblador, C y C++ y proporciona un flujo de diseño software que puede ser utilizado otros procesadores comerciales. En la figura 1.10 se muestra el flujo de diseño al utilizar un procesador *softcore*; con él, el estudiante puede entender que el flujo de diseño software se utiliza para generar el contenido de la memoria de programa del SoC y que la estructura del procesador se describe en un lenguaje de descripción de hardware al que se le aplica el mismo flujo de diseño que aplicó en la asignatura anterior.

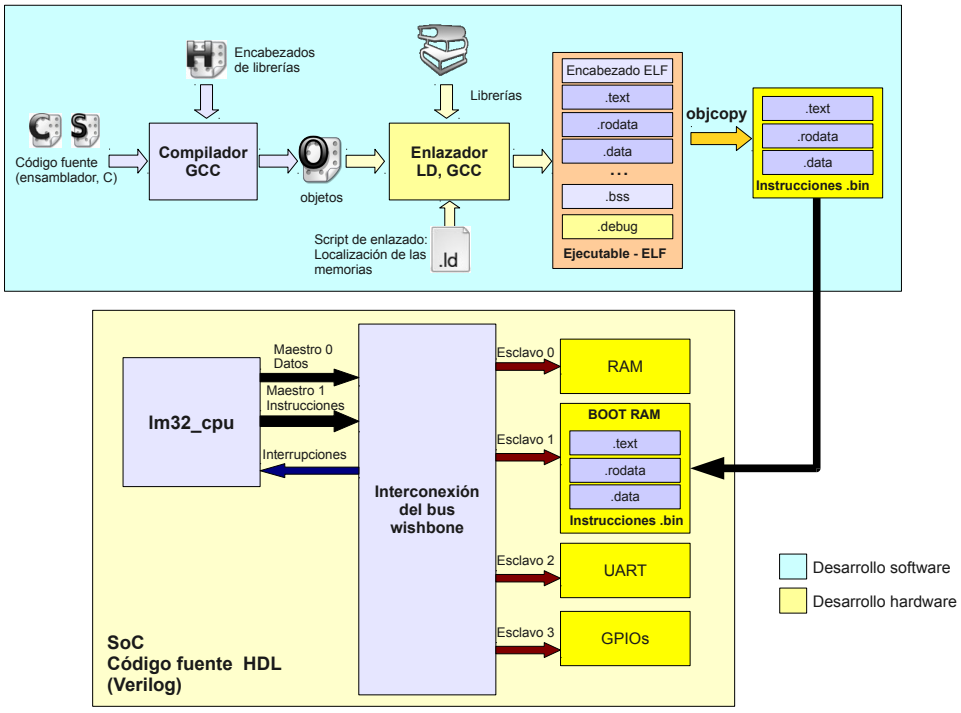

Figura 1.10: Flujo de diseño hardware/software al utilizar un procesador *softcore*

En este curso se implementarán tareas hardware en forma de periféricos utilizando los conocimientos en la asignatura electrónica digital 1, se estudiarán las diferentes opciones de comunicación entre la CPU y los periféricos y se darán las bases del particionamiento óptimo Hardware - software.

Una pregunta válida en este punto (y ya ha sido realizada por los estudiantes) es: ¿Cuando se enseñará el manejo de microcontroladores y microprocesadores comerciales? La pregunta es muy importante ya que estos dispositivos son muy utilizados y en una gran gama de aplicaciones no es necesario utilizar complejos SoC. La respuesta a la pregunta es que no es necesario enseñar el manejo de estos dispositivos ya que con los conocimientos adquiridos en este curso se cuenta con la información necesaria para que los estudiantes puedan hacerlo por ellos mismos; experiencias con los estudiantes del DIEE de la UNAL muestran que gracias a que la cadena de herramientas de compilación GNU soportan una gran variedad de procesadores de 8, 16 y 32 bits, es posible cambiar de CPU de forma fácil y si el código fué escrito de forma adecuada, utilizarlo con pequeñas modificaciones en diferentes procesadores.

En el repositorio del proyecto SIE [7] se puede obtener el código fuente 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*, lo que permite realizar modificaciones en la arquitectura del procesador, permitiendo 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. 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

[7] git clone http://projects.linuxencaja.net/SIE

los estudiantes a optimizarlos y a buscar en el mercado circuitos integrados que les permita cumplir con la funcionalidad requerida, construir una tarjeta hija y montar estos componentes.

En este curso el procesador ingenic de la plataforma SIE es utilizado como herramienta de configuración del PLD, 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*. Es posible comunicar el procesador LM32 implementado en la FPGA con el procesador ingenic utilizando sus puertos seriales, lo que proporciona un canal de depuración para las aplicaciones que se ejecutan en la FPGA; suministrando, todas las herramientas necesarias para la realización de las actividades.

Tanto la CPU, los periféricos, la memoria ram y la memoria de programa estarán implementados en la FPGA de SIE. Durante todo el periodo académico se desarrollará un proyecto de complejidad media que involucre el uso de tareas HW y SW, (en el portal *linuxencaja* se pueden observar los mejores proyectos realizados hasta el momento); para esto se formarán equipos de trabajo de 3 personas que deben hacer una propuesta inicial (concepción y especificaciones), realizar la descripción funcional del sistema utilizando algoritmos (diseño y modelo del sistema), realizar el particionamiento en funciones HW y SW, implementar estas funciones y diseñar una placa de circuito impreso (utilizando *kicad*) con lo necesario para implementar la funcionalidad requerida (implementación). Se realizarán entregas periódicas para verificar el cumplimiento del cronograma propuesto por el equipo de trabajo y el proceso de diseño debe ser documentado en la página wiki del curso. Esta actividad generará habilidades de trabajo en equipo, elaboración de esquemáticos, fabricación de PCBs, escritura de documentos técnicos e implementación de sistemas digitales.

1.3.7. Sistemas Embebidos

Una vez asimilados los conceptos de arquitectura de SoCs e implementación de tareas haedware y software se utilizará un SoC comercial, para que los estudiantes entiendan las diferencias entre los procesadores soft-core y hard-core y conozcan las herramientas más utilizadas en la implementación de sistemas digitales modernos. Asi mismo, se utilizará el sistema operativo Linux para ilustrar la diferencia entre las aplicaciones *standalone* y las que utilizan sistemas operativos, se utiliza el procesador ingenic para ejecutar tareas de visualización, comunicación, control e interfaz con el usuario, 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.

QI hardware [8], 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*; desarrollo software utilizando librerías gráficas como QT o GTK. *openwrt*.

La figura 1.11 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), 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

[8] http://projects.qi-hardware.com/

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.

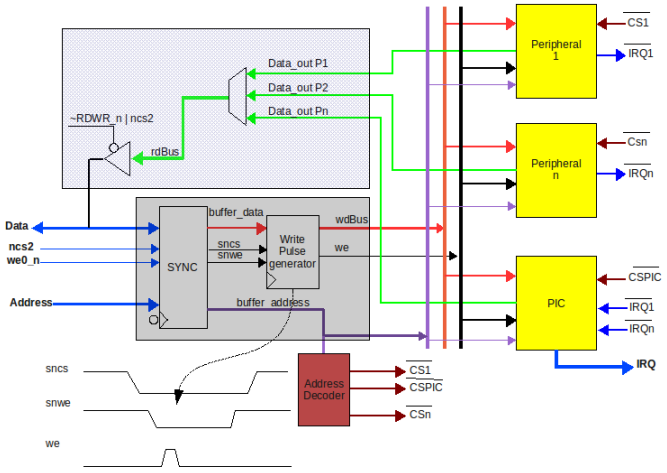

Figura 1.11: Interfaz HW-SW en SIE

Durante el periodo académico los estudiantes deben diseñar, implementar y operar un sistema embebido concebido por un equipo de trabajo de tres personas, utilizando como base la plataforma SIE, diseñarán una tarjeta hija (utilizando kicad) que implemente la funcionalidad deseada, todos los proyectos deben integrar tareas HW en la FPGA con módulos del kernel para su control. Los proyectos realizados hasta el momento pueden encontrarse en el portal *linuxencaja*.

1.3.8. 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, los archivos necesarios para replicar y/o modificarlo, esto es una consecuencia de la licencia CC-BY-SA que establece que todo proyecto derivado de un producto con este esquema de licencias debe proporcionar los archivos necesarios para entneder, reproducir y modificar el proyecto y que se aplique la licencia CC-BY-SA a esta modificación.

La experiencia del proyecto FOSS indica que muchos miembros de estas comunidades ingresan para suplir necesidades, pero muchos de ellos continuan creando código y prestando servicios 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 [5]. 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 copylef harware, 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.

Crear estos hábitos ayuda a que los jóvenes sean conscientes de su papel dentro de la 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 garantiza su crecimiento, la labor de los estudiantes es vital para el uso del recurso común y pueden surgir 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 conocimientos a los países en vía de desarrollo donde la plataforma tecnológica no se ha desarrolado lo suficiente para asimilar estos conocimientos.

1.3.9. Actividades

A continuación se enumerarán las actividades que se desarrollarán durante las tres asignaturas, indicando las habilidades (ver tablas 1.1 y 1.2) que se quieren reforzar o desarrollar.

Lectura de material del curso 10, 11

Con la lectura previa de los temas, el estudiante adquiere la capacidad de absorber conocimiento (11), identificar sus preferenicas, deficiencias y buscar ayuda para suplirlas (10), lo cual ayuda al mejoramiento de las habilidades para el auto-aprendizaje, uno de los problemas detectados en los estudiantes es la necesidad de una autoridad que le proporcione la información que necesita para resolver un problema o tomar una decisión, lo que crea una dependencia impidiendo que busquen la información por ellos mismos.

Lectura de material técnico en inglés 10, 11, 6, 30, 33, 21

La mayor parte de la documentación de los componentes electrónicos esta escrita en inglés técnico, es necesario que el estudiante aprenda a entender este tipo de escritura y se familiarice con su estructura. Esto le permite identificar el funcionamiento de un componente del sistema (6,30), determinar que componente se adapta mejor a sus necesidades (33) y mejorar sus habilidades para comunicarse en inglés (21).

Utilización de metodologías de diseño 1, 2, 3, 6, 7, 9, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38

La metodología de diseño (30,31) de sistemas embebidos requiere identificar un problema (1, 28), plantear una solución (3,29,32) lógica (9) de alto nivel (9), modelarla (2) a nivel de sistema(6), verificar el cumplimiento de los requerimientos(33,38); proporciona métodos para determinar su arquitectura óptima (particionamiento HW/SW) y definir la función e interacción (37,7) de sus componentes software (36) y hardware (35).

Implementación de sistemas digitales sencillos 3, 14, 29, 30, 35, 36, 17, 18, 19

La realización de prácticas de laboratorio en las que grupos de trabajo (14) implementan diseños de baja o media complejidad le permite al estudiante: Formular recomendaciones (3) para que no se repitan errores en experiencias futuras. Utilizar sistemas de desarrollo (30) para la implementación de tareas HW y SW a bajo nivel (36). Con el fin de mejorar la capacidad de comunicación escrita (18, 19) se deben presentar informes que refuercen las habilidades generadas en la utilización de la metodología de diseño, por lo que se deben tener la siguiente estructura (17):

- Un diagrama de caja negra que indique las entradas y salidas del sistema.
- Una descripción de alto nivel del algoritmo que implementa la solución (29).
- Un diagrama de bloques que indique el particionamiento y la interconexión entre sus componentes (30).
- Descripciones de alto nivel de cada uno de los componentes (31).
- La implementación y simulación de cada componente y del sistema completo (35), donde se muestre que el sistema cumple con las especificaciones funcionales (38).

Proyecto del curso 1,2,3, 14, 15, 30, 31, 32, 33, 34, 35, 22, 23, 24, 25, 27

Durante el semestre se trabajará para definir las especificaciones (1,2,3), diseñar (30,31,32,33,34) e implementar un dispositivo que resuelva una necesidad (con la complejidad adecuada para cada curso) de la sociedad (22), en la sesión teórica se tratarán aspectos relacionados con la concepción, diseño, identificación y definición de las funciones de los componentes del sistema, mientras que en el componente práctico, los relacionados con la implementación de dichos componentes sobre PLDs o SoC.

A los estudiantes se les hace una descripción funcional de alto nivel del sistema, ellos deben organizarse en grupos de trabajo (14,15), definir la función de cada uno de sus integrantes (27,14,31), establecer estrategias de comunicación (16,31), realizar y/o cumplir un cronograma de actividades (25,31) que permita resolver la necesidad en el tiempo especificado (22). Una de las estrategias de comunicación es la realización de presentaciones orales (20), en las que cada equipo de trabajo expondrá el estado de su sub-proyecto, indicando las razones que se tuvieron en cuenta en cada decisión y como se resolvieron los problemas encontrados (24). Adicionalmente, todo este proceso debe documentarse en el sitio web del curso (wiki del portal *linuxencaja*) con el objetivo de crear una base de proyectos que permitan a futuros estudiantes utilizar la experiencia obtenida (23) y en un determinado caso dar continuidad al proyecto.

El estudiante debe diseñar y construir placas de circuito impreso con los circuitos necesarios para su aplicación (35) siguiendo las normas de diseño establecidas por el fabricante (resolución, número de capas, costo) y las restricciones del circuito (Capacidad de corriente, niveles de ruido, compatibilidad electromagnética, etc). Vale la pena aclarar que durante el primer curso los estudiantes no poseen la experiencia necesaria para realizar (sin asistencia) labores como la división de tareas, generación de un cronograma de actividades y fijar la estrategia de comunicación, razón por la cual el docente debe acompañar este proceso.

Participación en listas de discución 21

Con el objeto de aumentar las capacidades en la comunicación en idioma extranjero, se alentará a los estudiantes a que hagan parte de listas de discusión en diferentes temas técnicos, algunos problemas que

encontrarán en la realización de las diferentes prácticas deben ser consultados en estas listas para encontrar una forma de solución

Hardware y Software Copy-Left

El conocimiento debe ser considerado un bien común y se debe garantizar el acceso a todo el mundo. Por esta razón SAKC proporciona la documentación necesaria para:

- Estudiar. entender, y reproducir o modificar su Arquitectura.
- Conocer su proceso de fabricación.
- Entender su funcionamiento global y la interacción de sus componentes.
- Estudiar tutoriales que explican su programación.
- Descargar, estudiar y modificar el código fuente de todas las aplicaciones existentes actualmente.
- Realizar consultas con los creadores de las aplicaciones y de las plataforma de desarrollo.
- Contribuir a mejorar la calidad de la documentación y crear nueva información.

1.4. Desarrollo de métodos de Evaluación

La naturaleza de esta metodología hace poco eficiente el método tradicional de evaluación ya que el proceso de diseño no se puede limitar a las 2 o 4 horas que puede durar una prueba escrita, tampoco se puede aislar al estudiante de las fuentes de información, ni de la consulta con otros estudiantes. La forma de evaluación debe simular un entorno laboral, donde se trabaja en equipo y cada miembro del equipo es responsable de una tarea, en la evaluación se tendrá en cuenta el resultado final, pero cada miembro del equipo debe sustentar de forma individual su aporte y sus conocimientos; esto con el fin de asegurar que todos los miembros del equipo trabajen y todos estén al corriente de las actividades que realizan sus compañeros.

La aplicación de la forma de evaluación tradicional a asignaturas en las que se busca crear en el estudiante habilidades que le permitan realizar el flujo de diseño completo de un sistema digital, contemplaba una marcada división entre el componente práctico y el componente teórico, antes de aplicar este programa académico, las prácticas de laboratorio y el tema que se trataba en clase no estaban sincronizados y las metodologías utilizadas eran diferentes, lo que originaba confusión en los estudiantes; existía un proyecto final que debía ser elaborado junto con las prácticas de laboratorio, lo que creaba una sobrecarga de trabajo que se traducía en proyectos de muy baja calidad. Por otro lado, no se puede medir la capacidad de diseño de un estudiante en un exámen convencional como los que se utilizan en otras asignaturas, primero porque el tiempo requerido para entender y asimilar el problema varía dependiendo de cada persona, y limitar este tiempo favorecería a los más rápidos pero no necesariamente a las mejores soluciones. El diseño es un proceso de creación personal y las soluciones dadas a un problema muestran rasgos de la personalidad que no pueden y no deben ser estandarizadas; lo que si se debe hacer es señalar como se pueden optimizar las soluciones.

Como se mencionó anteriormente, durante todo el semestre se realizará un proyecto que busca dar solución a un determinado problema; se realizarán tres avances para determinar su estado, las fechas y contenido de estos avances están sincronizadas con un cronograma que permite a los estudiantes aplicar

los conocimienos adquiridos. De esta forma, los estudiantes estarán realizando actividades durante todo el semestre, lo que los obliga a estar revisando y aplicando constantemente la información obtenida en el componente teórico; durante las horas de práctica los estudiantes trabajarán en el mismo proyecto y darán solución a problemas de implementacion con ayuda del profesor auxiliar. Esta forma de evaluación elimina la costumbre de estudiar una semana o un día antes de a prueba, buscar evaluaciones de semestres anteriores y memorizar la forma de solucionarlos. Asimismo, se proporciona al estudiante el tiempo suficiente para realizar el proceso de diseño completo lo que hace que esta experiencia se aproxime mucho a una situación que va a encontrar en el ejercicio de su profesion; adicionalmente, se pretende reducir la presión y el estrés que generan las pruebas escritas en los estudiantes, repartiendola en un equipo de trabajo que colabora para la realización de una meta común.

Contenido de las entregas

En cada entrega el equipo de trabajo debe realizar una serie de actividades que le ayuden a crear o mejorar habilidades en: la generación de documentos técnicos; presentación oral y escrita de proyectos en ingeniería; aplicación de metodologías de diseño; uso de la tecnología para resover problemas locales; trabajo en equipo y mejoramiento de técnicas de auto-aprendizaje. Por esta razón, se evaluará el contendio del informe, su presentación oral y el avance de la solución. Para la elaboración de los informes se suministran las herramientas web (*wiki*) que permiten la edición de documentos por múltiples usuarios, llevar un historial de cambios que les permita conocer los últimos aportes y pubicar de forma fácil imagenes y videos; lo que facilita la realización del documento, permitiendo que sea al mismo tiempo editado desde diferentes lugares al tiempo que lo coloca a disposición de quien esté interesado. La calificación obtenida en cada entrega será asignada a todos los miembros del equipo

Sustentación individual

Con las entregas parciales se obtiene una calificación que refleja el trabajo realizado por el grupo de trabajo; sinembargo, es necesario determinar la contribución de cada miembro del equipo y evaluar la asimilación de conocimiento de forma individual; adicionalmente, se pretende evitar que existan miembros del equipo que no realicen aportes. Para detrminar este nivel de asimilación, se aplica una prueba oral a cada uno de los integrantes, con el fin de determinar si existen o no vacios conceptuales y si se conoce el trabajo realizado en la elaboración del proyecto; de esta prueba se obtendrá un coeficiente entre **0** y **1**, el cual será aplicado a la calificación obtenida por el grupo y de esta forma obtener la calificación indicidual.

1.5. Objetivos de aprendizaje

Los objetivos de aprendizaje están formados por la superposición de los dominios congnitivo, afectivo y psicomotor [6]

1.5.1. Dominio cognitivo

El dominio congnitivo involucra conocimiento y desarrollo de habilidades iintelectuales. Incluye el reconocimiento de hechos específicos, procedimientos, y conceptos que ayudan en el desarrollo de habilidades y capacidades intelectuales. Bloon identificó las siguientes seis categorías, las que están ordenadas desde el comportamiento más simple al más complejo. Las categorías pueden ser considerados como grados de dificultades. Es decir, se deben dominar las primeros antes para poder desarrollar las otras.

1 Conocimiento: definido aquí como la acción de recordar información aprendida anteriormente.

- Objetivos generales
  - *Identifica* las diferentes etapas del proceso de diseño
  -
  -
  -
  -
  -
- Electrónica digital 1
  - *Identifica* la diferencia entre circuito lógico y secuencial.
  -
  -
  -
  -
  -
- Electrónica digital 2
  - *Identifica*
  -
  -
  -
  -
  -
- Sistemas embebidos
  - *Identifica* las diferentes etapas del proceso de diseño
  -
  -
  -
  -
  -

defines; describes; enumerates; identifies; labels; lists; matches; names; reads; records; reproduces; selects; states; views; writes;.

2 Comprensión: relacionado con los objetivos, comportamientos o respuestas que representan el entendimiento de un mensaje contenido en una comunicación, sin referirse a otro material. Para llegar a este entendimiento el estudiante puede cambiar la comunicación en su mente para reflejar una forma paralela más significativa para él.

- Objetivos generales
  - *Identifica* las diferentes etapas del proceso de diseño
  -
  -
  -

-
-

- Electrónica digital 1
  - *Identifica* la diferencia entre circuito lógico y secuencial.
  -
  -
  -
  -
  -
- Electrónica digital 2
  - *Identifica*
  -
  -
  -
  -
  -
- Sistemas embebidos
  - *Identifica* las diferentes etapas del proceso de diseño
  -
  -
  -
  -
  -

classifies; cites; converts; describes; discusses; estimates; explains; generalizes; gives examples; illustrates; makes sense out of; paraphrases; restates (in own words); summarizes; traces; understands.

3 Aplicación: habilidad de utilizar información aprendida previamente en nuevas situaciones para resolver problemas con una única o mejor solución.

acts; administers; applies; articulates; assesses; charts; collects; computes; constructs; contributes; controls; demonstrates; determines; develops; discovers; establishes; extends; implements; includes; informs; instructs; operationalizes; participates; predicts; prepares; preserves; produces; projects; provides; relates; reports; shows; solves; teaches; transfers; uses; utilizes.

- Objetivos generales
  - *Identifica* las diferentes etapas del proceso de diseño
  -
  -
  -
  -
  -
- Electrónica digital 1
  - *Identifica* la diferencia entre circuito lógico y secuencial.
  -

-
-
-
-

- Electrónica digital 2
  - *Identifica*
  -
  -
  -
  -
  -
- Sistemas embebidos
  - *Identifica* las diferentes etapas del proceso de diseño
  -
  -
  -
  -
  -

4 Análisis: La separación de la información en sus partes componentes, examinando y entendiendo su estructura. Distinguiendo entre hechos e inferencias.

analyzes; breaks down; categorizes; compares; contrasts; correlates; diagrams; differentiates; discriminates; distinguishes; focuses; illustrates; infers; limits; outlines; points out; prioritizes; recognizes; separates; subdivides.

5 Síntesis: definida como la acción de unir elementos y partes para conformar una estructura o patrón, enfatizando en la creación de un nuevo significado o estructura.

adapts; anticipates; collaborates; combines; communicates; compiles; composes; creates; designs; develops; devises; expresses; facilitates; formulates; generates; hypothesizes; incorporates; individualizes; initiates; integrates; intervenes; invents; models; modifies; negotiates; plans; progresses; rearranges; reconstructs; reinforces; reorganizes; revises; structures; substitutes; validates.

6 Evaluación: emitir juicios personales sobre el valor de las ideas o materiales.

appraises; compares y contrasts; concludes; criticizes; critiques; decides; defends; interprets; judges; justifies; reframes; supports.

1.5.2. Conclusiones

El método de evaluación propuesto ha demostrado su eficacia en lo relacionado con la eliminación de presiones y la generación de hábitos de continuo estudio de la asignatura; sin embargo, se ha observado que los estudiantes están acostumbrados a actuar por presión de una calificación; esto es, si no hay calificación no estudian o no realizan los ejercicios propuestos; por este motivo, se incorporó las tareas a las entregas parciales, siendo esta la única forma de hacer que la mayoría de los estudiantes leyeran con anticipación los temas que se tratarán en las clases; aunque esto no es lo recomendado, es la realidad y hasta que todas las asignaturas ayuden a que los estudiantes cambien sus habitos de estudio es necesario tomar este tipo

de medidas. Este comportamiento muestra que los estudiantes no utilizan un método de estudio adecuado, y que la universidad no proporciona los medios para que se generen; en algunas universidades del país se acostumbra a hacer pequeños exámenes para forzar a los estudiantes a leer los temas con anterioridad, es una solución efectiva pero genera una sobrecarga de trabajo para el docente y la solución final es el resultado de una medida que castiga y no de un hábito del estudiante.

Encuestas realizadas a los estudiantes durante los últimos dos años muestran que ellos perciben un grado de exigencia mucho mayor comparando con otras asignaturas; pero al mismo tiempo, que la experiencia en estos cursos es muy útil para su vida profesional y que es la única asignatura que los enfrenta a problemas reales de diseño e implementación de sistemas; entienden que es necesario dedicar tiempo por fuera de aula si se desea asimilar la información, son concientes de que la responsabilidad de adquirir este conocimiento es de ellos y manifiestan la importancia del uso de esta tecnología en la solución de problemas locales.

1.6. Software

Para la realización de las actividades descritas anteriormente se utilizará software abierto que pueda ser descargado libremente sin restricción alguna, esto con el fin de proporcionar al estudiante herramientas software legales que le permitan adquirir las habilidades descritas anteriormente; y puedan ser utilizadas en su ejercicio profesional en el desarrollo de nuevos productos. Debemos tener presente que además de proporcionar a los estudiantes los conocimientos necesarios en su profesión, también debemos hacer que ellos respeten los derechos de autor y la propiedad intelectual de las herramientas software comerciales, en muchas universidades se acostumbra a ignorar el uso de software ilegal por parte de los estudiantes ya que es necesario para el desarrollo del curso y el establecimiento no cuenta con recursos para adquir las licencias suficientes; aunque el uso de este software conseguido de forma ilegal es muy loable no deja de ser una violación a las leyes que protegen los derechos de los creadores de aplicaciones software; adicionalmente, formar a estudiantes con herramientas costosas los vuelve dependientes de ellas, lo que requerirá de grandes inversiones en software cuando ellos formen sus propias empresas, lo que puede desalentar este tipo de iniciativas. Por lo anterior se hizo una revisión y prueba de las herramientas abiertas disponibles que pueden ser utilizadas en los cursos de diseño digital y que proporcionan la funcionalidad requerida en esta propuesta académica.

1.6.1. Análisis y simulación de circuitos

En la mayoría de centros de formación se utilizan herramientas que utilizan el simulador SPICE (Simulation Program with Integrated Circuits Emphasis) el cual ha demostrado ser una herramienta muy útil en diferentes áreas (electrónica análoga, electrónica digital, diseño VLSI, electrónica de potencia, entre otras) ya que proporciona diferentes formas y niveles de simulación, permitiendo el uso de modelos para representar de la mejor forma posible un dispositivo real. SPICE fué desarrollado por el departamento de ingeniería eléctrica y ciencias de la computación de la universidad de Berkeley [9] y su código fuente es distribuido bajo la licencia BSD (Berkeley Software Distribution), la última versión disponible es la *3f5*, gracias a esto se han desarrollado una serie de herramientas que utilizan este potente simualador y proporcionan una interfaz amigable para el usuario.

En nuestros cursos se utiliza *QUCS* (Quite Universal Circuit Simulator) desarrollado en QT y permite ser ejecutado en cual sistema operativo. QUCS permite realizar análisis AC, DC, transitorio, de balance

[9] http://http://bwrc.eecs.berkeley.edu/Classes/IcBook/SPICE/

armónico y soporta dispositivos lineales, no lineales, micro-cintas, coplanares, líneas de transmisión y representados por modelos SPICE. En la figura 1.12 se muestra la captura de esta aplicación.

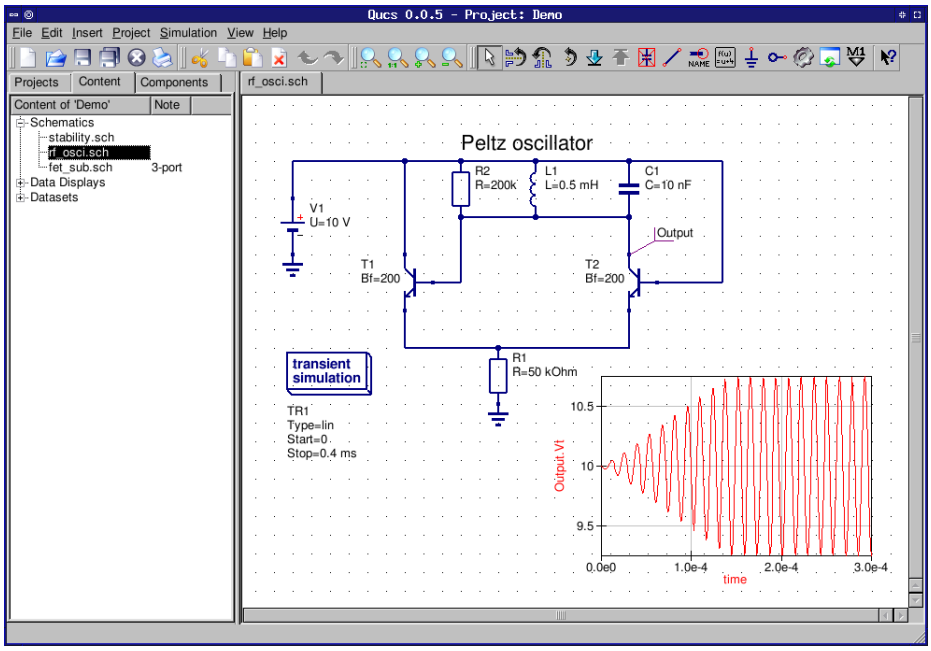

Figura 1.12: Simulador QUCS

1.6.2. Síntesis, simulación y verificación digital

Para la síntesis digital a partir de lenguajes de descripción de hardware se utilizan las herramientas gratuitas suministradas por los fabricantes de FPGAs, *webpack* de Xilinx y *Quartus* de Altera; debido a que la estructura interna de las FPGAs solo las conocen ellos[10], es obligatorio utilizar sus herramientas para obtener el archivo de configuración [11].

Para la simulación de sistemas digitales que utilizan como entrada de diseño lenguajes de descripción de hardware existen los simuladores *ICARUS* para verilog y *GHDL* para vhdl; los dos pueden ser utilizados para realizar simulaciones funcionales, post síntesis o post place & route (trabajando en conjunto con las herramientas de los fabricantes) y ambos soportan el formato de salida VCD (definido junto con el lenguaje de descripción de hardware verilog por el estándar IEEE 1364-2001). Adicionalmente, estas herramientas pueden ser utilizadas en los sistemas operativos más utilizados.

Como herramienta de simulación se utilizará *GTKWAVE*, la cual acepta como entrada archivos en formato *VCD* y puede ser ejecutada en MAC, Linux y Windows. *GTKWAVE* realiza un manejo adecuado de la jerarquía del sistema bajo análisis, permitiendo observar todas las señales de los diferentes módulos que componen la jerarquía superior, lo que es muy útil en este tipo de simulaciones; en la figura 1.13 se puede observar una captura de esta herramienta.

[10] Es posible obtener información valiosa de sus patentes por ejemplo: http://www.freepatentsonline.com/6301693.pdf

[11] En la actualidad existen proyectos que buscan proporcionar herramientas abiertas que realicen estas funciones, pero no han logrado obtener una herramienta que las pueda reemplazar

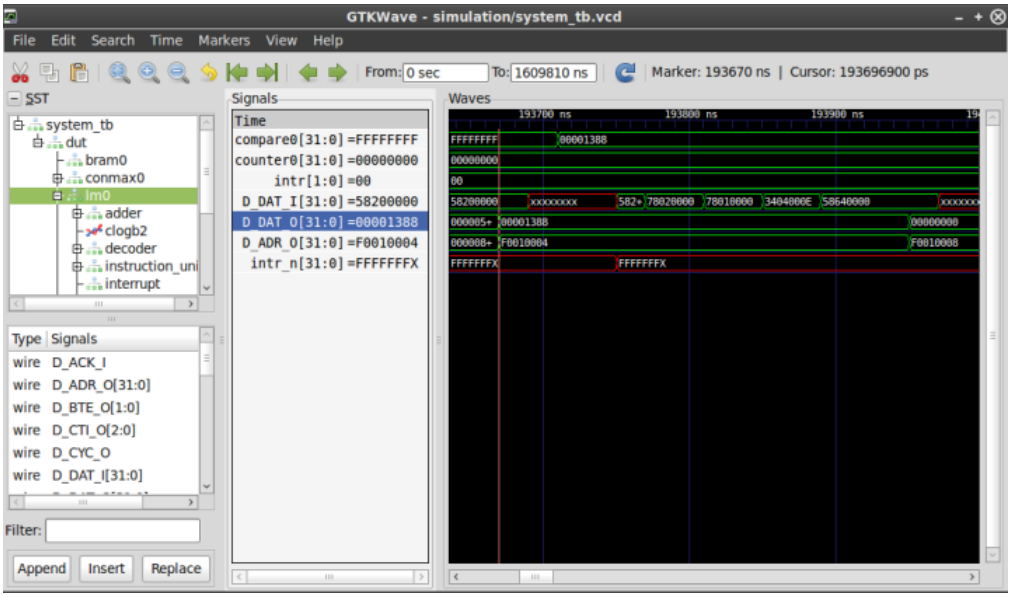

Figura 1.13: Visualizador de formas de onda *GTKWAVE*

Una vez configurada la FPGA se utiliza la herramienta *URJTAG* para verificar el correcto funcionamiento del sistema implementado en la FPGA, *URJTAG* proporciona una capa de abstracción de hardware que permite el manejo del puerto JTAG de cualquier dispositivo, proporcionando funciones de alto nivel para la aplicación de las funciones JTAG (IDCODE, INTEST, EXTEST, BYPASS, SAMPLE/PRELOAD) permitiendo la aplicación de vectores de prueba al núcleo lógico de la FPGA y la captura de la respuesta a estos estímulos; estas pruebas se realizan a baja frecuencia. *URJTAG* soporta varias interfaces físicas para control de las señales del puerto JTAG (TDI, TDO, TMS y TCK) las cuales pueden ser conectadas a diferentes puertos de un computador, o como en nuestro caso a un puerto virtual creado en el procesador MIPS.

1.6.3. Herramientas para la configuración de PLDs

Aunque con la plataforma de desarrollo SIE no es necesario utilizar herramientas ni unidades de programación adicionales para configurar su FPGA; el procesador de SIE ejecuta dos aplicaciones que son utilizadas para realizar esta función y pueden ser ejecutadas en computadores personales: *URJTAG* y *XC3SPROG*, las dos funcionan de forma similar, utilizan dispositivos conectados al puerto paralelo (conexión directa) o USB (basados en el protocolo MPSSE del chip FT2232) del computador; para ejecutar las instrucciones extendidas de la FPGA CFG_OUT, CFG_IN, JSTART y JPROGRAM (hasta el momento solo han sido probadas con las FPGAs de Xilinx).

1.6.4. Diseño de placas de circuito impreso

Este tipo de herramientas han sido las más difíciles de conseguir ya que al comienzo de este estudio no existía una herramienta que permitiera realizar de forma fácil la elaboración del esquemático, asignación de *footprints*, distribución y localización de componentes en el layout, ruteo de las señales, y verificación

de reglas de diseño tanto en el esquemático como en el layout. En la actualidad el proyecto *KICAD* ha alcanzado un nivel adecuado para ser utilizado en el desarrollo de aplicaciones comercializables y fué utilizado en todos los proyectos académicos durante la aplicación del presente plan de estudios; está formado por 5 aplicaciones: el editor de esquemáticos *Eeschema*; el editor de circuitos impresos *Pcbnew*; el visor de archivos gerber *Gerbview*; la utilidad para asignar footprints *Cvpcb* y el manejador de proyectos *Kicad*.

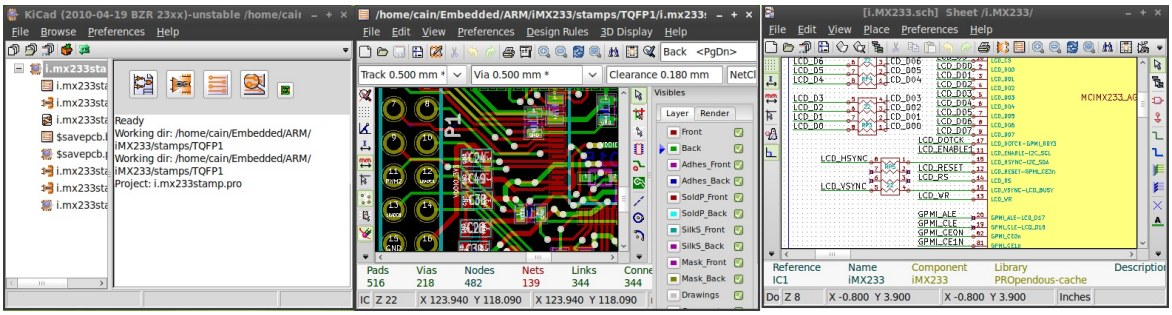

Figura 1.14: Herramienta para la elaboración de circuitos impresos *KICAD*

1.6.5. Herramientas de desarrollo para sistemas embebidos

GNU binutils

Colección de utilidades para archivos binarios y estan compuestas por:

- **addr2line** Convierte direcciones de un programa en nombres de archivos y números de línea. Dada una dirección y un ejecutable, usa la información de depuración en el ejecutabe para determinar que nombre de archivo y número de línea está asociado con la dirección dada.
- **ar** Esta utilidad crea, modifica y extrae desde ficheros; un fichero es una colección de otros archivos en una estructura que hace posible obtener los archivos individuales.
- **as** Utilidad que compila la salida del compilador de C (GCC).
- **c++filt** Este program realiza un mapeo inverso: Decodifica nombres de bajo-nivel en nombres a nivel de usuario, de tal forma que el *linker* pueda mantener estas funciones sobrecargadas (overloaded) “from clashing”.
- **gasp** GNU Assembler Macro Preprocessor
- **ld** El *linker* GNU combina un número de objetos y ficheros, re-localiza sus datos y los relaciona con referencias. Normalmente el último paso en la construcción de un nuevo programa es el llamado a ld.
- **nm** Realiza un listado de símbolos de archivos tipo objeto.
- **objcopy** Copia los contenidos de un archivo tipo objeto a otro. *objcopy* utiliza la librería GNU BFD para leer y escribir el archivo tipo objeto. Permite escribibr el archivo destino en un formato diferente al del archivo fuente.
- **objdump** Despliega información sobre archivos tipo objeto.

- **ranlib** Genera un índice de contenidos de un fichero, y lo almacena en él.
- **readelf** Interpreta encabezados de un archivo ELF.
- **size** Lista el tamaño de las secciones y el tamaño total de un archivo tipo objeto.
- **strings** Imprime las secuencias de caracteres imprimibles de almenos 4 caracteres de longitud.
- **strip** Elimina todos los símbolos de un archivo tipo objeto.

Compilador

El *GNU Compiler Collection* normalmente llamado GCC, es un grupo de compiladores de lenguajes de programación producido por el proyecto GNU. Es el compilador standard para el software libre, de los sistemas operativos basados en Unix y algunos propietarios como Mac OS de Apple. Soporta los lenguajes ADA, C, C++, Fortran, Java, Objective-C, Objective-C++ para las arquitecturas Alpha, ARM, Atmel AVR, Blackfin, H8/300, System/370, System/390, IA-32 (x86), x86-64, IA-64 i.e. the Ïtanium", Motorola 68000, Motorola 88000, MIPS, PA-RISC, PDP-11, PowerPC, SuperH, SPARC, VAX, Renesas R8C/M16C/M32C y MorphoSys. Gracias a esto puede considerarse como una herramienta universal para el desarrollo de sistemas embebidos, el código escrito en una plataforma (en un lenguaje de alto nivel) puede ser implementado en otra sin mayores cambios, esto elimina la dependencia entre el código fuente y el procesador [12], lo que es posible cuando se utiliza el lenguaje ensamblador.

GNU Debugger

El depurador oficial de GNU (GDB) al igual que GCC, soporta múltiples lenguajes y plataformas; permite monitorear y modificar las variables internas del programa y hacer llamado a funciones de forma independiente a la ejecución normal del mismo. Además, permite establecer sesiones remotas utilizando el puerto serie o TCP/IP. Aunque GDB es una aplicación que se ejecuta en consola de comandos, se han desarrollado varios front-ends como DDD o GDB/Insight.

Librerías C

Es necesario contar con las librerías standard de C: stdio, stdlib, math, etc; las más utilizadas en sistemas embebidos son:

- **glibc** Es la librería C oficial del proyecto GNU; el principal inconveniente al trabajar con esta librería en sistemas embebidos es que genera ejecutables de mayor tamaño que los generados a partir de otras librerías, lo cual no la hace muy atractiva para este tipo de aplicaciones.
- **uClibc** Es una librería diseñada especialmente para sistemas embebidos, es mucho más pequeña que **glibc**.
- **newlib** Al igual que **uClibc**, está diseñada para sistemas embebidos. El típico "Hello, world!" ocupa menos de 30k en un entorno basado en newlib, mientras que en uno basado en glibc, puede ocupar 380k.
- **diet libc** Es una versión de *libc* optimizada en tamaño, puede ser utilizada para crear ejecutables estáticamente enlazados para linux en plataformas alpha, arm, hppa, ia64, i386, mips, s390, sparc, sparc64, ppc y x86_64.

[12] Esto recibe el nombre de re-utilización de código

Sistema operativo Linux

El sistema operativo Linux es utilizado en este curso en el sistemas emebebido y en el computador personal, su carácter abierto y de código libre permte al estudiante estudiar su funcionamiento y autilizarlo en el desarrollo de aplicaciones haciendo uso de la gran variedad de aplicaciones y librerías para diversos campos de aplicación; adicionalmente, Linux soporta una gran variedad de protocoos de comunicación, sistemas de archivos y da soporte a una extensa variedad de periféricos que utilizan diferentes protocolos (USB, SPI, I2C, SD, SATA, PCI, PCI-e, etc) lo que facilita el desarrollo de aplicaciones tanto en los computadores personales como en los sistemas embebidos.

Librería Qt

Qt es un marco de trabajo portable que es utilizado para el desarrollo de aplicaciones software con interfaz de usuario gráfica (GUI); este proyecto proporciona los entornos de desarrollo *QT - designer* y *Qt - creator* los cuales, proporiconan una interfaz de programación amigable con una gran variedad de objetos gráficos que pueden ser integrados fácilmente; acelerando el tiempo de desarrollo de la interfaz y suministrando un entorno agradable al usuario final, en la figura 1.15 se puede observar un ejemplo de una interfaz gráfica creada con Qt.

Figura 1.15: Interfaz gráfica generada con *Qt*

Sistema de archivos

El sistema de archivos proporciona las herramientas necesarias por el sistema operativo para implementar una determinada funcionalidad; comunicándose con el kernel para controlar el flujo de datos desde y hacia los periféricos. Aunque es posible construir un sistema de archivos desde cero, no es nada práctico ya que es una tarea tediosa que requiere cierto nivel de experiencia; por esto, en la actualidad, existen varias

distribuciones que proporcionan herramientas de configuración que permiten generar de forma automática una gran variedad de aplicaciones, dentro de las más utilizadas se encuentran:

- Busybox: Diseñado para optimizar el tamaño y rendimiento de aplicaciones embebidas, BusyBox [13] combina en un solo ejecutable más de 70 utilidades estándares UNIX, en sus versiones ligeras. BusyBox es considerada la navaja suiza de los sistema embebidos, dado que permite sustituir la gran mayoría de utilidades que se suelen localizar en los paquetes GNU fileutils, shellutils, findutils, textutils, modutils, grep, gzip, tar, etc. Busybox hace parte de la mayoría de distribuciones de Linux para sistemas embebidos.

- Buildroot: conformado por un grupo de *scripts* y *parches* que facilita la generación de la cadena de herramientas y el sistema de archivos para un sistema embebido que usa Linux. Posee una interfaz que permite realizar de forma fácil la configuración; utiliza busybox para generar la utilidades básicas de Linux y permite adaptar software adicional de forma fácil [14].

- OpenEmbedded: Al igual que Buildroot, el proyecto openembedded proporciona un entorno que permite generar la cadena de herramientas y el sistema de atchivos para un sistema embebido, utiliza busybox y permite la creación de archivos que permiten compilar software que no se incluya en la distribución original. Adicionalmente openembedded crea archivos de instalación con un formato derivado del proyecto *handhelds ipk*, lo que permite la instalación de paquetes de forma similar a la distribución debian.

- Openwrt: Distribución que fué utilizada inicialmente para routers domésticos, es utilizada en la actualidad en otras plataformas; proporciona los mismos servicios que OpenEmbedded, ocupando menos espacio en el computador donde se realiza la compilación, el proceso de compilación es más rápido.

1.6.6. Herramientas CAD

Para la elaboración de las cajas que contendrán las tarjetas electrónicas y servirán de soporte físico a los dispositivos electrónicos, se utilizará la herramienta *QCAD* disponible para los sistemas operativos más utilizados; con esta aplicación se puede generar la forma de la placa de circuito impreso con los orificios necesarios para ajustarse a la caja y a la aplicación; este archivo puede ser exportado a la herramienta *KICAD* lo que facilita la colocación de componentes.

1.6.7. Conclusiones

Las aplicaciones mencionadas en esta sección han sido probadas por estudiantes de las universidades: UNAL, ULA, UIS y por la empresa emQbit, por esto podemos decir que este grupo de herramientas soporta todo el proceso de diseño de un sistema embebido; su caracter abierto y gratuito permite ahorrar mucho dinero si se compara con el requerido para comprar herramientas propietarias; el único inconveniente de las herramientas abiertas es que en algunos casos su interfaz no es tan amigable y no cuentan con las funcionalidades de su contraparte comercial. Una ventaja adicional es la no utilización de software ilegal, práctica que ha sido tolerada por la academia durante muchos años; adicionamente, se le enseña al estudiante una alternativa que le permite ahorrar costos de inversión en el software necesario para comenzar una nueva empresa; con lo que este dinero se puede destinar a compra de equipo necesario para el desarrollo de la actividad.

[13] http://www.busybox.net/

[14] http://buildroot.uclibc.org/buildroot.html#add_software

Aunque este grupo de herramientas pueden ejecutarse en los sistemas operativos mas populares (Linux, Mac OS y Windows) se prefiere el uso de Linux ya que es un sistema operativo gratuito y no es necesario pagar ningún tipo de licencia, lo que reduce aún más la inversión en software.

Encuestas realizadas a los estudiantes indican que: con este grupo de herramientas se pueden realizar todas las actividades propuestas en este programa academico; que son más difíciles de manejar que las herramientas comerciales; pero que se prefiere el uso de alternativas abiertas al de sofware ilegal; en el pasado no les importaba utilizar métodos ilegales para licenciar aplicaciones software, pero que ahora van a buscar primero las alternativas libres antes de recurrir al uso de software ilegal; el uso de Linux hace que se sientan identificados con él, llegando al punto de realizar un cambio total de sistema operativo. Lo anterior demuestra que los estudiantes se adaptan fácilmente a nuevas herramientas y que ellos prefieren el uso de software abierto a software ilegal, por lo que el constante uso de herramientas comerciales se puede atribuir a los docentes que no están interesados en buscar alternativas abiertas, ya sea porque cuentan con licencias o porque no les interesa que se utilice software iegal.

Bibliografía

[1] Worldwide CDIO Initiative. ”Benefits of CDIO”URL:http://www.cdio.org/benefits-cdio on November, 2009.

[2] E. Crawley, J. Malmqvist, D. Brodeur, and B. Lucas. CDIO Syllabus, Leadership and Entrepreneurship. *5th International CDIO conference*, 2009.

[3] Edward F. Crawley. The CDIO Syllabus A Statement of Goals for Undergraduate Engineering Education. *URL:http://www.cdio.org*, 2001.

[4] 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.

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

[6] Bloom B S, editor. *Taxonomy of Educational Objetives, the classification of educational goals*. Mckay.