# Diseño Digital Utilizando Lógica Programable: Aplicaciones a la Enseñanza

Juan Pablo Oliver

Tesis de Maestría en Ingeniería Eléctrica

Director de tesis y director académico: Gregory Randall

Instituto de Ingeniería Eléctrica
Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay, 2007

ISSN : 1510-7264 - Reporte Técnico Nº

# Resumen

Los dispositivos lógicos programables son circuitos integrados que contienen una gran cantidad de celdas básicas, específicamente compuertas y registros, cuyas interconexiones pueden ser configuradas por el usuario para dar lugar a un diseño determinado. Estos dispositivos se han transformado en componentes esenciales de cualquier diseño electrónico digital, desplazando en gran medida a los componentes discretos.

La tecnología de la lógica programable ha significado un cambio de paradigma en el diseño electrónico: un circuito que puede modificarse vía software; ofreciendo una gran cantidad de ventajas y posibilidades. Este cambio de paradigma en la forma de diseñar también ha producido importantes transformaciones en la forma de enseñar.

En esta tesis se presentan una serie de experiencias innovadoras en la enseñanza de ingeniería electrónica utilizando lógica programable. Se plantea el estudio de un tema tecnológico como es la lógica programable, eligiendo como campo de aplicación la utilización de esta tecnología en la enseñanza de diseño electrónico digital.

Se comienza estudiando diferentes recomendaciones en la enseñanza de la ingeniería, profundizando los aspectos prácticos, de diseño y de laboratorio. Luego se realiza una puesta al día en profundidad de la lógica programable, incluyendo los dispositivos, el proceso de diseño y las herramientas utilizadas. Por último se presentan una serie de experiencias e investigaciones en metodologías de enseñanza de diseño electrónico digital.

Estas experiencias están divididas en dos grupos, en una primer instancia la mejora del curso introductorio de diseño lógico con una nueva e innovadora metodología de laboratorio; y posteriormente el desarrollo de plataformas reconfigurables realizadas por estudiantes avanzados como proyectos de fin de carrera. En ambos casos se muestran los resultados obtenidos, tanto desde el punto de vista educativo como tecnológico.

# Tabla de Contenido

# Lista de Figuras

# Capítulo 1

# Introducción

Si bien dentro de las actividades universitarias la enseñanza juega un rol muy importante, la gran mayoría de las tesis de maestría en ingeniería eléctrica tratan de temas de investigación en áreas técnicas pero no involucrando a la enseñanza. Lo usual es mostrar el estado del arte en un tema y aplicarlo por medio de algún desarrollo, en el cual pueden intervenir algoritmos, modelos, software o hardware.

El enfoque de esta tesis es presentar el estado del arte del diseño digital utilizando electrónica programable, y las "aplicaciones" o los "desarrollos" realizados fueron innovaciones en la enseñanza, que transformaron la forma de enseñar electrónica en la Universidad de la República. Se plantea entonces el estudio de un tema tecnológico como es la lógica programable, pero gran parte del trabajo se enfoca en la utilización de esta tecnología en la enseñanza de diseño electrónico digital.

Esta opción implicó un desafío, por un lado dominar una disciplina y una determinada tecnología, y por otro combinar ese conocimiento con metodologías modernas de enseñanza y aprendizaje para generar innovaciones educativas que vienen siendo aplicadas a cursos reales desde hace más de tres años.

Los aportes principales que caracterizan este trabajo son:

- la puesta al día en la lógica programable, incluyendo dispositivos, el proceso de diseño y las herramientas utilizadas.
- el estudio sobre diferentes recomendaciones en la enseñanza de la ingeniería haciendo hincapié en las actividades prácticas, de diseño y de laboratorio
- la recopilación de las experiencias de varios años de trabajo, investigación y experimentación en metodologías de la enseñanza de diseño electrónico digital

Como resultados directos más importantes podemos mencionar:

- la mejora del curso introductorio de diseño electrónico digital con una nueva metodología de laboratorio
- la incorporación de *know-how* en el área de diseño electrónico digital dentro del Instituto de Ingeniería Eléctrica de la Universidad de la República

Los dispositivos lógicos programables ya tienen más de 25 años de existencia, y hoy en día sería inconcebible pensar en diseños digitales que no los utilicen. En sus comienzos estos dispositivos fueron utilizados para sustituir diseños realizados con lógica discreta y posteriormente para sustituir diseños *full-custom* cuando los volúmenes de producción son bajos. Pero una de las mayores ventajas de estos dispositivos, que en sus orígenes no fue adecuadamente valorada, es la capacidad de ser reprogramados. Es en esta flexibilidad en donde se encuentra el mayor atractivo de esta tecnología, permitiendo pasar un diseño de la idea al silicio en tiempos muy cortos.

La lógica programable permite obtener velocidades hardware con flexibilidad software. La posibilidad de reutilización del hardware programable abarata su costo ya que puede utilizarse exactamente el mismo hardware para varias aplicaciones cambiando exclusivamente su programación interna [52].

La tecnología de la lógica programable o reconfigurable nos ofrece entonces un cambio de paradigma: hardware que puede modificarse vía software. De la misma manera que una computadora puede escribir datos en una memoria, la misma computadora puede grabar un determinado circuito dentro de un chip, y cambiarlo tantas veces como se quiera[1]. El circuito se modifica internamente, sin la necesidad de que haya cambios físicos externos.

[1] Existen algunos dispositivos en los cuales esto no se cumple ya que no permiten cambiar su programación, ver detalles en la sección 3.1

El objetivo del trabajo expuesto en esta tesis fue explorar las distintas alternativas que ofrece este nuevo paradigma, incursionando en diversos aspectos del diseño con lógica programable aplicados en la mejora de la enseñanza de diseño digital.

Las experiencias de enseñanza fueron realizadas como parte del currículo de la carrera de Ingeniería Eléctrica en la Universidad de la República, Uruguay. El trabajo puede dividirse claramente en dos partes: aplicaciones a enseñanza básica y aplicaciones avanzadas. En la parte básica se presenta la reestructuración del primer curso de diseño digital (Diseño Lógico), con la introducción de una nueva modalidad de laboratorios. En la parte avanzada se presentan experiencias realizadas como proyecto de grado (Proyecto de fin de carrera), en las que se desarrollaron plataformas reconfigurables con interfaz PCI[2] y aplicaciones realizadas con dichas plataformas.

## 1.1 Objetivos

El objetivo de este trabajo fue explorar las distintas alternativas que ofrece el nuevo paradigma de la lógica programable y utilizarlas para realizar mejoras en la enseñanza de diseño digital. Un cambio profundo en la tecnología siempre implica cambios en la enseñanza, pero en este caso el cambio tecnológico se adapta muy bien a incorporar innovaciones educativas. Los dispositivos lógicos programables se impusieron a nivel comercial por presentar varias ventajas y muchas de esas ventajas son también aplicables en el campo de la enseñanza.

Mucho se ha avanzado en los últimos años en la industria de los dispositivos lógicos programables debido a la evolución de varios aspectos críticos: aumento de capacidad y velocidad de los chips, disminución de costos, facilidad en los métodos de reprogramación, mejora sustancial de las herramientas de diseño y disponibilidad de diseños pre-hechos o bloques reutilizables (Intellectual property cores o IP cores).

[2] La descripción de las siglas utilizadas puede verse en el Glosario, al final del documento, página 124

Analizaremos cada uno de estos aspectos por separado:

1. Aumento de la capacidad de los chips de lógica programable

El aumento en el tamaño de los circuitos integrados de lógica programable ha acompañado el desarrollo general de la microelectrónica, esto en el caso de los circuitos programables ha redundado en una ampliación drástica del campo de aplicación de los mismos. Inicialmente los circuitos programables sólo podían implementar pequeñas funciones lógicas, decodificaciones o máquinas de estado, sustituyendo diseños realizables con lógica discreta. Actualmente se pueden realizar diseños muy complejos que incluyen unidades de punto fijo o punto flotante, filtros digitales, algoritmos completos, e incluso microprocesadores, ya sea de propósito general o dedicados. El incremento en la velocidad de los circuitos programables también acompaña el aumento de tamaño, pudiéndose por ejemplo, implementar multiplicadores con frecuencias de 450MHz-500MHz [14, 115].

El aumento de capacidad de los circuitos programables ha sido una de las claves en las cuales se basa el salto cualitativo del campo de aplicación de los mismos.

2. Reducción de costos

Los dispositivos lógicos programables adquiridos en grandes cantidades tienen costos similares a los microprocesadores comerciales [9]; además las compañías poseen líneas de productos de bajo costo con muy buena performance (familias Cyclone en Altera y Spartan de Xilinx).

3. Mejora en las herramientas de diseño

Otro de los factores con un alto impacto para abordar diseños complejos en dispositivos programables es el avance en el desarrollo de herramientas de diseño asistido por computador; y aquí debemos mencionar no solo la ampliación de las prestaciones de los diversos programas de diseño, sino también el abaratamiento de los mismos, tanto en el costo propio del software cómo su posibilidad de correr en plataformas baratas (PCs)

4. Métodos de reprogramación o configuración

Los primeros dispositivos requerían de programadores especiales para ser configurados, e incluso el borrado de la programación era con luz ultravioleta. Actualmente existen dos alternativas, chips con programación volátil (RAM), o chips

con programación no volátil (FLASH, EEPROM, Antifuse); pero ambas se realizan directamente desde una computadora a través de un cable.

5. Disponibilidad de bloques pre-diseñados (Intellectual property cores o IP-cores)

El último factor de gran impacto en el diseño de aplicaciones en chips programables es la posibilidad de contar con bibliotecas de bloques reutilizables pre-diseñados. De esta forma se logra, en cierta medida, paliar la granularidad fina inherente a las diversas arquitecturas de FPGAs. Si bien las FPGAs siguen estando basadas en estructuras básicas elementales, la utilización de ladrillos de mayor tamaño hace que esta limitación sea parcialmente superada. Cabe destacar que la utilización de IP-cores trae aparejados nuevos problemas como la interconexión de los mismos, o el aumento de la dificultad en la verificación de la funcionalidad del sistema completo.

Todos estos factores, que son presentados a nivel general, ofrecen también ventajas a la hora de utilizar dispositivos lógicos programables en enseñanza, haciéndolos totalmente adecuados para la implementación de plataformas de experimentación para estudiantes.

## 1.2 Introducción a los Dispositivos Lógicos Programables

Dada la importancia de estos dispositivos en el contexto de esta tesis se hará aquí una brevísima introducción a los mismos, dejando su estudio en profundidad para más adelante (ver Capítulo 3).

Los circuitos lógicos programables en sus varias formas (Field Programable Gate Arrays (FPGA), Programable Logic Devices (PLD)), son circuitos electrónicos que contienen una gran cantidad de elementos lógicos básicos: compuertas y celdas de memoria. Especificando las interconexiones entre esos elementos básicos se personaliza al dispositivo para realizar una función determinada. En lógica

programable, esta personalización es realizada por el diseñador del sistema, no por el fabricante del circuito integrado. Esto permite bajar costos fijos y hacer accesible esta tecnología para volúmenes de producción mucho más bajos.

No es fácil establecer una terminología exacta así como una clasificación de los dispositivos lógicos programables. Diversos autores así como fabricantes muchas veces redefinen o introducen términos nuevos.

Los primeros dispositivos lógicos programables son desarrollados en la segunda mitad de la década del ’70, por Monolithic Memories, y eran básicamente simples estructuras de compuertas AND–OR ordenadas para realizar funciones combinatorias de dos niveles. Los primeros nombres utilizados fueron: PALs (Programmable Arrays Logic) y PLAs (Programmable Logic Arrays); debían ser programados fuera del sistema, no podían ser programados en el circuito; y principalmente fueron utilizados para realizar aplicaciones del tipo “glue logic”. El paso inmediato fue el agregado de elementos de memoria (Flip-Flops) conectables a la salida de la función combinatoria, con lo cual el campo de aplicaciones se amplió a máquinas secuenciales simples.

Los siguientes nombres de dispositivos que se utilizaron fueron PLDs (Programmable Logic Devices), que según su complejidad y tamaño se pueden subclasificar en SPLDs (Simple Programmable Logic Devices), y CPLDs (Complex Programmable Logic Devices). En 1985 aparece la primer FPGA (Field Programmable Gate Array) introducida por Xilinx [27]. El mercado de dispositivos lógicos programables está dominado por Xilinx y Altera, existiendo otras compañías que producen dispositivos como Actel, Atmel, Cypress y Lattice.

Básicamente los dispositivos lógicos programables están compuestos por un gran conjunto de bloques lógicos cuya función es programable y una red de interconexiones de estos bloques también programable (ver Fig. 1-1). La red de interconexiones programable puede ser jerárquica o plana, y con caminos de diversa

longitud entre bloques. Los bloques lógicos tienen una parte combinatoria y un elemento de memoria que puede o no ser utilizado.

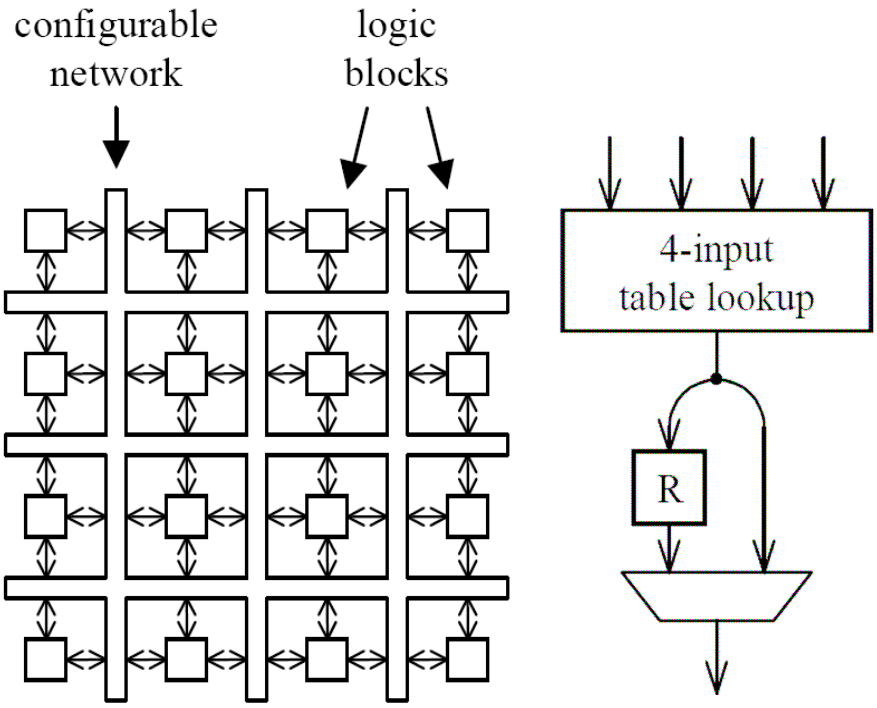


**Fig. 1-1 Estructura interna de un dispositivo programable y un bloque lógico (extraído de [59])**

Hoy en día los nombres utilizados son (C)PLDs y FPGAs y aunque las fronteras entre ambos son cada vez más borrosas en principio los PLDs están basados en estructuras de dos niveles lógicos, mientras las FPGAs típicamente usan lookup-tables (LUTs) programables de 4 entradas, ambas estructuras con un registro opcional a la salida. Otra diferencia es el modo de programación, mientras en las FPGAs la programación es volátil y está basada en una SRAM (Static Random Access Memory), los PLDs típicamente utilizan programación no volátil basada en tecnología EPROM o EEPROM ((Electrically) Erasable Programmable Read Only Memory). Los retardos internos son otra de las principales diferencias entre FPGAs y PLDs, siendo en las primeras variable y dependientes del enrutamiento de las señales, mientras que en los PLDs los retardos son generalmente fijos y más fáciles de estimar.

Tradicionalmente los dispositivos lógicos programables son personalizados fuera del sistema, o cargan su personalización durante la inicialización. Se les puede cambiar su programación a medida que se va refinando el diseño de un prototipo. Esto da a los diseños con FPGAs suficiente flexibilidad para adaptarse a cambios en los requerimientos en etapas avanzadas del diseño, lo que los hace muy atractivos para realizar implementaciones tempranas de estándares no totalmente definidos.

Un enfoque más reciente en la utilización de las FPGAs aprovecha el hecho de que estos dispositivos permiten "infinitas" reprogramaciones en forma dinámica y que el tiempo que lleva la reconfiguración de los chips es muy pequeño. De esta manera se puede redefinir el hardware en tiempo real. Este nuevo enfoque ha dado en llamarse "Lógica Reconfigurable" o "configurable". Las mismas celdas lógicas pueden ser en determinado momento un contador, luego una ALU o un filtro digital. Cada compuerta puede realizar una función durante un tiempo y cuando esa función ya no es necesaria ser utilizada para realizar otra [34, 105].

La tecnología disponible posibilita la realización de funciones de gran tamaño dentro de una FPGA, o más aún la implementación de un algoritmo completo o parte de él dentro de un único integrado. Resulta entonces atractivo integrar una computadora de propósito general con dispositivos programables como forma de obtener una arquitectura híbrida que permita que parte de un algoritmo se ejecute en el procesador central, y parte en hardware programable y de esta forma obtener un aumento significativo en la velocidad de ejecución.

## 1.3 Propuestas de la Tesis

Partiendo de la premisa que los dispositivos lógicos programables son componentes indiscutibles a la hora de implementar un diseño digital, y de su gran utilidad tanto a nivel industrial como educativo; esta tesis propone su utilización para mejorar aspectos prácticos de la enseñanza de la ingeniería eléctrica.

La primer propuesta es entonces una alternativa para los cursos introductorios de diseño digital, en la cual se plantea sustituir los laboratorios clásicos por una nueva modalidad, sin perder calidad de enseñanza, o incluso mejorándola, y manteniendo un fuerte vínculo entre la teoría y la práctica del diseño electrónico. Se propone un sistema en el cual la realización de la práctica es fuera del aula y se expone lo realizado en una defensa. Para esto fue necesario diseñar y construir una gran cantidad de kits hardware de bajo costo, para ser entregados en préstamo a los estudiantes durante todo el semestre. En estos kits los grupos de estudiantes pudieron llevar a la práctica sus diseños y mostrarlos a los docentes en una defensa oral obteniendo así una calificación.

Esta experiencia fue realizada en la carrera de Ingeniería Eléctrica de la Universidad de la República, pero creemos que puede ser trasladada a otros contextos.

La segunda propuesta involucra estudiantes avanzados realizando el diseño de plataformas reconfigurables con interfaz PCI como proyecto de grado. Se presentan ejemplos de cómo estudiantes de grado pueden llevar adelante proyectos de gran complejidad y calidad basados en dispositivos lógicos programables de gran tamaño.

## 1.4 Sinopsis de la Tesis

Describiremos brevemente la estructura del documento.

En el Capítulo 2 se presenta el problema de la enseñanza con laboratorios y de enseñar a diseñar. Luego se hace un paralelismo entre la lógica programable y el software, y posteriormente se analizan las alternativas para enseñar diseño digital; para finalizar con los antecedentes de la lógica programable en la Universidad de la República.

En el Capítulo 3 se profundiza en los dispositivos lógicos programables. Es un capítulo recopilatorio y bien documentado de estos dispositivos, y puede saltearse si el

lector está familiarizado con los mismos. Se hace una revisión exhaustiva de los métodos de configuración, los diferentes tipos de dispositivos existentes, así como de su evolución histórica. Para finalizar el capítulo se describen las diferentes herramientas de software que se utilizan para diseñar con este tipo de circuitos.

El Capítulo 4 presenta la primer experiencia de enseñanza. Aquí se describe todo el proceso de rediseño del curso de Diseño Lógico, los kits utilizados y la nueva metodología de enseñanza. Finaliza con los resultados obtenidos de los dos primeros dictados del nuevo curso, resultados académicos y resultados obtenidos a través de una encuesta estudiantil. En el Apéndice puede verse el resultado de dicha encuesta.

El Capítulo 5 describe la segunda experiencia en enseñanza, que en realidad son varios proyectos escalonados realizados por estudiantes como proyecto de fin de carrera o tesis de grado. El tema es el desarrollo de plataformas reconfigurables con interfaz PCI. Si bien se describen las plataformas desarrolladas, el capítulo está enfocado a analizar la experiencia desde el punto de vista didáctico. Como forma de validar los resultados se presentan brevemente otros proyectos de estudiantes que utilizan las plataformas diseñadas.

Por último en el Capítulo 6 se dan algunas conclusiones, se analiza el trabajo realizado y se plantean líneas de trabajo para el futuro.

# Capítulo 2

# Antecedentes

Es bien conocida la importancia que tienen los laboratorios y las experiencias de diseño en la enseñanza de la ingeniería. A continuación traducimos y transcribimos una serie de citas de Wankat & Oreovicz extraídas del libro "Teaching Engineering"[111], porque nos parecen un buen punto de partida para plantear estos temas. Los autores comienzan modificando una cita de Eastlake [37] en la que ponen a la par los laboratorios y las experiencias de diseño durante el estudio de ingeniería:

> "La ingeniería sin laboratorios [y diseño] es una disciplina diferente. Si eliminamos los laboratorios [y el diseño] deberíamos renombrar la carrera y llamarle Matemática Aplicada" (Eastlake, 1986).
>
> "Nosotros concordamos con esta cita modificada, que el diseño y las sesiones de laboratorio son el corazón de una educación en ingeniería. No hay nada malo en una carrera en matemática aplicada, pero esto no es el título que los estudiantes y las compañías creen que ellos están recibiendo." [[111] p. 168]

En cuanto a cómo enseñar a diseñar en ingeniería los mismos autores dicen:

> "La forma más usual de enseñar a diseñar y de tener una significativa experiencia en diseño es con proyectos de diseño. A los estudiantes, usualmente en grupos, se les da un problema de diseño y se les dice que lo diseñen. Puesto que los ingenieros aprenden a diseñar diseñando, este es ciertamente un procedimiento apropiado. Más aún, la gente recuerda las cosas que hizo. Nosotros podemos recordar nuestros proyectos de diseño (y nuestros laboratorios) veinticinco años después, pero no podemos recordar los detalles de las clases que recibimos." [[111], p. 173]

En cuanto a las herramientas de software a utilizar: "Un curso que no utilice software profesional está preparando a nuestros estudiantes para un tipo de trabajo que ya no

existe"[93]. Creemos que esto mismo puede aplicarse tanto a las herramientas de software como hardware.

Los cursos con laboratorios son caros, tanto en equipamiento como en tiempo de dedicación docente, veremos qué dicen al respecto los mismos autores:

> "Desde el punto de vista de un departamento la excelencia en los laboratorios son una fuente de orgullo. Si usted no cree esto, visite un departamento con un excelente laboratorio de grado y note la actitud del profesor que lo guía a través del mismo. Excelentes laboratorios también ayudan a producir ingenieros bien preparados. Por supuesto que los departamentos obtienen aquello por lo que pagan. Laboratorios de excelencia requieren dinero para su equipamiento, mantenimiento, técnicos y profesores dedicados, los cuales mantendrán su dedicación solamente si son adecuadamente remunerados. Aquellos departamentos que utilicen sus laboratorios como una forma de ahorrar dinero cuando el presupuesto es ajustado pagarán el precio de una pérdida de excelencia bastante rápidamente." [[111], p. 184]

En algunos casos la utilización de ciertas tecnologías permiten la realización de laboratorios y de experiencias de diseño con costos menores. Este es el caso del software por ejemplo, ya que con una computadora se pueden realizar una gran cantidad de experiencias diferentes. Un caso similar se plantea con los dispositivos lógicos programables, ya que la capacidad de reprogramación de los mismos hace que una misma plataforma pueda utilizarse para realizar múltiples experiencias. En el próximo apartado comenzaremos realizando una comparación entre hardware y software en el contexto de ingeniería eléctrica.

## 2.1 La lógica programable y los microprocesadores

El hardware programable tiene ciertas diferencias y ciertas similitudes con la arquitectura tradicional de un sistema basado en un microprocesador ejecutando instrucciones de programa leídas de una memoria. La similitud es que también tiene una memoria, escondida para el usuario, en la cual se debe cargar el código de configuración. El código que se carga en dicha memoria es intrínsecamente diferente del software. El software está diseñado en base a instrucciones que se ejecutan

secuencialmente una tras otra, en cambio el código de configuración, a ser cargado en un chip reconfigurable, no contiene instrucciones a ser ejecutadas, sino que indica cómo deben interconectarse las celdas que se encuentran dentro de una FPGA. Es decir que es una descripción de un determinado circuito electrónico. Al no ejecutar instrucciones en forma secuencial se puede aprovechar el paralelismo de todo el hardware disponible, pudiendo obtenerse importantes mejoras en velocidad.

¿Qué sucede cuando el sistema contiene elementos configurables? Aquí el concepto de software no puede aplicarse tal cual, ya que no hay un set de instrucciones a ejecutar. Pero sí hay una memoria en la que se almacena la configuración del circuito a implementar. Podría interpretarse como una instrucción muy larga que determina las interconexiones del hardware configurable. Algunos autores llaman a este flujo de datos de configuración *configware*. [58]

Otro punto importante es la reconfiguración del hardware, esto significa flexibilidad, y permite desarrollar múltiples aplicaciones con una misma plataforma, y con una misma inversión inicial. Puede hacerse una analogía al desarrollo del software, al abaratarse las computadoras personales y los compiladores o los ambientes de desarrollo de software el mismo tuvo un crecimiento explosivo. Salvando las distancias, la posibilidad de contar con plataformas reconfigurables de bajo costo, y herramientas de diseño hardware, han producido un crecimiento importante en diseños reutilizables ofrecidos a terceros (ver por ejemplo: www.opencores.org) en un proceso que tiene un interesante paralelismo con el fenómeno del *opensource* en el software, pero que es de aparición mucho más reciente. Hay autores que justifican esta demora en la aparición, y posterior proliferación, de los microprocesadores en los años ’70, ya que durante varios años los microprocesadores pudieron satisfacer la demanda de cálculo para la gran mayoría de las aplicaciones. Tradennick [107] va más allá aún y dice que los microprocesadores “*...retrasaron el desarrollo de la lógica programable al menos en 10 años”*. Obviamente esta afirmación no es comprobable ni refutable, pero sí puede verse que en la década del ’70 disminuyeron los esfuerzos de investigación en arquitecturas reconfigurables, aunque las ideas fundamentales ya habían sido publicadas.

El desarrollo de herramientas de diseño asistido ha llevado a incorporar una gran cantidad de conceptos de software en el diseño hardware, y se puede realizar una analogía entre los procesos. En el caso de un programa escrito en un lenguaje de alto nivel existe una compilación y posteriormente una optimización del mismo que lleva a obtener instrucciones en lenguaje de máquina o lenguaje ensamblador. Análogamente en el diseño de un circuito, que parte de una especificación en un lenguaje de descripción hardware, se realiza un proceso de compilación del mismo y luego un proceso de síntesis y optimización que dan como resultado un circuito eléctrico como un listado de componentes e interconexiones. Es necesario entonces que el ingeniero electrónico que se mueve en esta frontera conozca ambos procesos, tanto en sus semejanzas como en sus diferencias.

¿Dónde poner entonces la frontera entre hardware y software? En general, para resolver un determinado problema, existen variadas soluciones que van desde un sistema "puramente" software hasta soluciones cien por ciento hardware. La frontera puede moverse de acuerdo a criterios de ingeniería que involucran velocidad, consumo, confiabilidad o costos. Podríamos decir que existe una relación dialéctica entre el hardware y el software.

## 2.2 Alternativas para enseñar diseño digital enmarcadas en los nuevos criterios en la enseñanza de la ingeniería

Veremos una reseña de las alternativas que existen para la enseñanza de diseño electrónico digital en el contexto de las nuevas tendencias en la enseñanza de la ingeniería, haciendo hincapié en las experiencias prácticas y de laboratorio. Para esto analizaremos documentos de recomendaciones acerca de programas de ingeniería de organismos de acreditación y ejemplos de otras universidades.

A nivel internacional los cursos básicos de electrónica digital son obligatorios en las carreras de Ingeniería Eléctrica e Ingeniería en Computación, siendo generalmente cursos compartidos por ambas carreras.

En Estados Unidos, por ejemplo, la IEEE Computer Society y la Association for Computing Machinery (ACM) han elaborado en conjunto una serie de reportes que establecen recomendaciones para los programas de grado de las diferentes carreras o especialidades que involucran computación. En particular el reporte sobre la carrera de Ingeniería en Computación: *Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering, elaborado por The Joint Task Force on Computing Curricula IEEE Computer Society Association for Computing Machinery*, el 12 de diciembre de 2004 [28] tiene recomendaciones que también son aplicables a la Ingeniería Eléctrica ya que el área de diseño digital es común a ambas orientaciones. A continuación se hace un breve resumen de este documento en el tema de diseño digital, laboratorios y actividades prácticas.

En dicho informe, en su Capítulo 5, se establece la importancia de integrar actividades prácticas y actividades de diseño distribuidas a lo largo de la carrera. También se destaca la importancia de una experiencia de diseño final, en la forma de proyecto de fin de carrera. Las experiencias de laboratorio se establecen como una parte esencial en la formación de un ingeniero, permitiendo a los estudiantes observar y manipular dispositivos o sistemas reales. En los laboratorios se recomienda incluir actividades de diseño, implementación, testeo, generación de documentación; así como el diseño de experimentos de adquisición de datos, interpretación de los mismos y, cuando corresponda, la utilización de esos datos para mejorar o corregir el diseño inicial.

Se admiten de igual forma experiencias de laboratorios formando parte integral de un curso o como cursos independientes.

Se recomienda que los laboratorios incluyan implementaciones físicas de circuitos, y que también se utilicen herramientas de simulación.

Se afirma que en las etapas formativas de su educación, los estudiantes a menudo son motivados por la naturaleza práctica de la ingeniería. Las experiencias de laboratorio capitalizan este interés y ayudan a desarrollar en los estudiantes confianza en sus habilidades técnicas. Las experiencias de laboratorio promueven experticia en construir nuevos dispositivos y ayudan a apreciar el importante rol del personal técnico, los equipos de trabajo y los profesionales de otras disciplinas.

Se destaca la importancia de utilizar herramientas de ingeniería tales como paquetes de diseño asistido por computadora, lenguajes de descripción hardware, simuladores, emuladores, modelos, instrumentos reales y virtuales, herramientas de depuración o *debugging*, etc.

Otro punto mencionado es la necesidad que los estudiantes adquieran capacidades de comunicación oral y escrita; y también la capacidad de trabajar en equipo, siendo importante que haya experiencias de este tipo a lo largo de la carrera.

En la sección 8.3 se advierte la necesidad de contar con recursos adecuados para los laboratorios y sobre los costos, usualmente elevados, que esto trae aparejado, tanto en recursos humanos como materiales.

En el Apéndice A, donde se describe el conjunto de conocimientos de toda la carrera, se incluyen en secciones obligatorias los dispositivos lógicos programables; y se espera que el estudiante sea capaz de utilizar estos dispositivos (FPGAs y PLDs) para implementar diseños de sistemas digitales. Esto muestra que este tipo de dispositivos están totalmente afianzados en la enseñanza de la electrónica digital desplazando completamente a los dispositivos de lógica discreta tradicionales de las familias TTL y CMOS.

En el Apéndice B se dan cuatro ejemplos de programas que muestran diferentes formas de implementar la carrera. En todos los casos se incluyen cursos obligatorios de lógica digital, con experiencias de laboratorio. La carga horaria del laboratorio en los primeros tres ejemplos es de 42 horas (3 horas semanales para un curso de 14

semanas). También se incluye un proyecto de diseño al final de la carrera con una carga de 84 horas totales.

La difusión de la educación a distancia ha llevado a repensar el tema de las experiencias prácticas de laboratorio, las alternativas son variadas, y van desde meros tutoriales o simuladores que corren localmente, hasta verdaderos laboratorios remotos con experimentos controlados a distancia. Siguiendo con el caso de Estados Unidos, ABET Inc. que es la institución encargada de acreditar las carreras de ingeniería en dicho país realizó en el año 2002 un coloquio [39] para definir el rol de los laboratorios en la educación del ingeniero, e intentar responder a las preguntas de si la educación a distancia puede lograr los mismos objetivos que los laboratorios reales.

Partiendo de la base que una de las características principales de la educación en ingeniería son los laboratorios reales y las experiencias de diseño, se generan una serie de cuestionamientos si un programa de educación a distancia no incluye clases reales ni experiencias de laboratorio. Como parte de este proceso se comienza preguntándose cuáles son los resultados de esas experiencias prácticas en el currículum, y si se pueden definir los atributos de los graduados en ingeniería que fueron formados con laboratorios tradicionales; para por último llegar a conocer si estos mismos atributos pueden ser logrados a través de programas de educación a distancia. Los participantes del coloquio fueron docentes de vasta experiencia en educación en ingeniería, principalmente en el área de laboratorios.

Uno de los resultados fue la definición amplia de las experiencias de laboratorio como una “interacción personal con equipamiento/herramientas que lleva a la acumulación de conocimiento y habilidades requeridas en una profesión orientada a la práctica”.

En consenso entre los participantes del coloquio se definieron los objetivos de aprendizaje para los laboratorios de ingeniería, que se citan textualmente (traducción libre del autor):

“Todos los objetivos comienzan de la siguiente forma: “Completando los laboratorios durante la carrera de ingeniería el estudiante podrá....”

Objetivo 1: Instrumentación

Aplicar sensores apropiados, instrumentación, y/o herramientas de software para realizar medidas de cantidades físicas.

Objetivo 2: Modelos

Identificar las fortalezas y limitaciones de los modelos teóricos como predictores del comportamientos del mundo real. Esto puede incluir evaluar si una teoría describe adecuadamente un acontecimiento físico y establecer o validar una relación entre los datos medidos y los principios físicos subyacentes.

Objetivo 3: Experimento

Idear un acercamiento experimental, especificar el equipo apropiado y los procedimientos, poner estos procedimientos en ejecución, e interpretar los datos resultantes para caracterizar un material, un componente, o un sistema de ingeniería.

Objetivo 4: Análisis de datos

Demostrar la capacidad de recoger, analizar, e interpretar datos, y de sacar y apoyar conclusiones. Poder hacer juicios basados en órdenes de magnitud, y conocer los sistemas de unidades y sus conversiones.

Objetivo 5: Diseño

Diseñar, construir, o montar una parte, un producto, o un sistema, incluyendo el uso de metodologías específicas, equipamientos, o materiales; alcanzando los requerimientos del cliente; desarrollando especificaciones del sistema a partir de los requerimientos; y probando y eliminando errores de un prototipo, de un sistema, o de un proceso usando las herramientas apropiadas para satisfacer los requisitos.

Objetivo 6: Aprender de las fallas

Reconocer los fracasos debidos a equipos defectuosos, partes, código, construcción, proceso, o diseño, y entonces realizar un proceso de reingeniería para obtener soluciones efectivas.

Objetivo 7: Creatividad

Demostrar niveles apropiados de pensamiento independiente, creatividad, y capacidad de solucionar problemas del mundo real.

Objetivo 8: Psicomotor

Demostrar la capacidad de selección, modificación, y operación de herramientas y recursos apropiados de la ingeniería.

Objetivo 9: Seguridad

Reconocer los temas de salud, seguridad y ambientales relacionados con los procesos y las actividades tecnológicas, y manejarse con ellos responsablemente.

Objetivo 10: Comunicación

Comunicarse con eficacia acerca del trabajo de laboratorio con una audiencia específica, tanto en forma oral como escrita, a niveles que van desde resúmenes ejecutivos a informes técnicos completos.

Objetivo 11: Trabajo en equipo

Trabajar eficazmente en equipo, incluyendo la estructuración de responsabilidades individuales y colectivas; asignar roles, responsabilidades, y tareas; supervisar el progreso; cumplir con los plazos; e integrar las contribuciones individuales en el entregable final.

Objetivo 12:La ética en el laboratorio

Comportarse con los estándares éticos más altos, incluyendo la divulgación de información en forma objetiva e interactuando con integridad.

Objetivo 13: Conocimiento sensorial

Utilizar los sentidos humanos para recopilar información y hacer juicios sólidos de ingeniería al formular conclusiones sobre problemas del mundo real."

El coloquio no tuvo conclusiones sobre la educación a distancia, pero definió una serie de acciones para evaluar su aplicación en la ingeniería.

Los defensores de los laboratorios virtuales o remotos señalan que varios de estos objetivos pueden alcanzarse fuera de los laboratorios reales, las excepciones son los objetivos de instrumentación, psicomotor y de conocimiento sensorial. Además clasifican estos objetivos en tres rangos según su importancia [20, 49]: ética, análisis de datos, comunicación y trabajo en equipo son considerados esenciales. Modelos, experimentos, instrumentación y seguridad son considerados muy importantes. Conocimiento sensorial, psicomotor, aprendizaje de los fallos y diseño son considerados importantes. Con esta clasificación dos de los objetivos no alcanzables quedan en la categoría intermedia y uno en la baja. Nosotros no coincidimos con esta clasificación, ya que por experiencia personal podemos advertir que tanto las experiencias de diseño aplicadas en los laboratorios, como el aprendizaje de los errores allí cometidos y su corrección son aportes valiosísimos en la formación de los estudiantes.

Faisel y Rosa [40] analizan el rol de los laboratorios en las carreras de ingeniería, en un artículo muy completo y bien documentado publicado en 2005. Allí destacan la importancia de los laboratorios en la formación del ingeniero, realizan una reseña histórica de los mismos y señalan la falta de definición de sus objetivos pedagógicos hasta el coloquio del año 2002 organizado por ABET [39]. Plantean la necesidad de profundizar los 13 objetivos del coloquio dándole una "calibración"; definen además

varias líneas futuras de investigación, pronosticando un crecimiento en la investigación pedagógica sobre los laboratorios.

En forma más general ABET Inc. ha definido una serie de criterios para acreditar las carreras de ingeniería, comúnmente llamados criterios 3.a-k [7] que citamos textualmente (traducción libre del autor):

"Los programas de ingeniería deben demostrar que sus estudiantes logran:

a) capacidad para aplicar conocimientos de matemáticas, ciencia e ingeniería
b) capacidad para diseñar y llevar a cabo experimentos, así como analizar e interpretar datos
c) capacidad para diseñar un sistema, componentes, o procesos para alcanzar las necesidades requeridas dentro de restricciones reales tales como económicas, ambientales, sociales, políticas, éticas, de salud y seguridad, de posibilidad de fabricación y que sea sostenible
d) capacidad para trabajar en equipos multidisciplinarios
e) capacidad para identificar, formular, y resolver problemas de ingeniería
f) comprensión de las responsabilidades profesionales y éticas
g) capacidad para comunicarse en forma efectiva
h) amplitud de educación necesaria para comprender el impacto de las soluciones de la ingeniería en un contexto global, económico, ambiental y social
i) el reconocer la necesidad y la habilidad de enrolarse en un proceso de aprendizaje continuo
j) conocimiento de temas contemporáneos
k) capacidad de usar técnicas, destrezas, y herramientas modernas de ingeniería necesarias para la práctica de la profesión"

Estos criterios son logrados a través de los cursos, pero varios de ellos son específicos de actividades prácticas (b) y de diseño (c).

A partir de la adopción de estos criterios se produjeron una serie de cambios en las carreras de ingeniería de las universidades norteamericanas cuyo impacto fue posteriormente evaluado exhaustivamente [72]. Los nuevos criterios mantienen ciertas características de los estándares anteriores pero incluyen además el desarrollo de habilidades de comunicación, la capacidad de identificar y resolver problemas, el trabajo en equipo, priorizando además criterios éticos y contextuales.

La adopción de los criterios 3.a-k llevó a un cambio importante en los métodos de enseñanza como puede verse en la Fig. 2-1, incrementando el uso de métodos de enseñanza activa, tales como proyectos de diseño, trabajos en grupos, estudios de caso, ejercicios de aplicación y la utilización de simuladores.

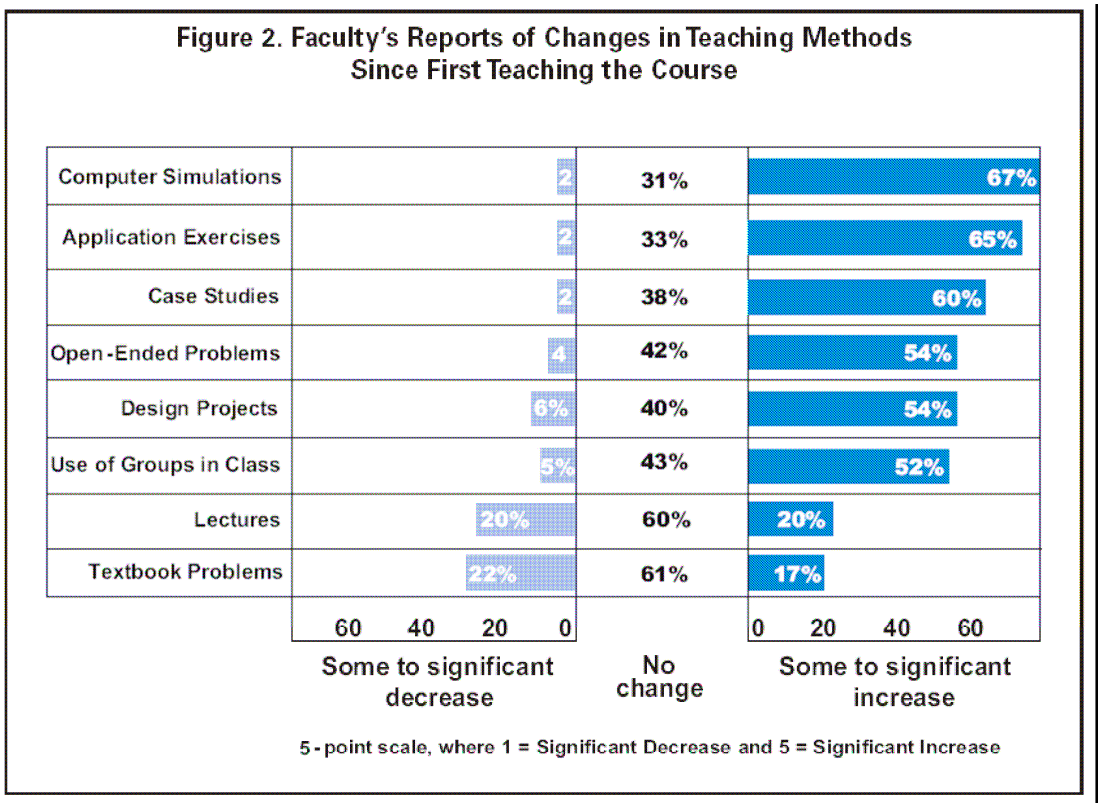


**Fig. 2-1 Cambios en los métodos de enseñanza a partir de los nuevos criterios 3.a-k (extractado de [72])**

Por su parte la Unión Europea se encuentra embarcada en un proceso de cambio de gran magnitud que ha sido llamado el Proceso de Bologna, que comienza con la declaración de la Sorbonne en 1998 y posteriormente con la declaración de Bologna en 1999. Allí se establece un plazo hasta 2010 para la realización del Espacio Europeo de Educación Superior (EHEA), con fases bienales. La primera fase tuvo lugar en Praga en 2001, la segunda fue en Berlín en 2003, la siguiente tuvo lugar en Bergen (Noruega) en mayo de 2005 y la próxima se realizará en Londres en 2007.

Las bases para este proceso son:

- La adopción de un sistema de grados fácilmente entendible y comparable
- La adopción de un sistema esencialmente basado en dos ciclos (grado y posgrado)
- Establecimiento de un sistema homogéneo de créditos (ECTS) y calificaciones
- Promoción de la movilidad
- Promoción de cooperación para asegurar la calidad
- Promoción de la dimensión europea en educación superior
- Aprendizaje continuo
- Participación de las instituciones de educación superior y de los estudiantes
- Promoción del atractivo del Area de Educación Superior Europea y colaboración con el resto del mundo

En la definición del sistema de créditos se establece lo siguiente:

> "El Sistema europeo de transferencia y acumulación de créditos (ECTS) es un sistema centrado en el estudiante, que se basa en la carga de trabajo del estudiante necesaria para la consecución de los objetivos de un programa. Estos objetivos se especifican preferiblemente en términos de los resultados del aprendizaje y de las competencias que se han de adquirir."[33]

Este movimiento ha llevado también al rediseño de los programas de varias carreras, cambiando el sistema de enseñanza tradicional en el cual la asignatura y el profesor eran el centro del proceso educativo, por una metodología donde el alumno y el desarrollo de sus capacidades pasan a ocupar el lugar central. En este nuevo marco cobran sentido los denominados métodos de enseñanza activos, como pueden ser el estudio de casos o el aprendizaje basado en proyectos [77].

Para tener una idea del impacto de este proceso, en el Congreso Tecnologías Aplicadas a la Enseñanza de la Electrónica realizado en Madrid en julio de 2006, hubo cinco sesiones especialmente dedicadas al Espacio Europeo de Educación Superior; y de las 155 presentaciones europeas, 41 hacen referencia al tema.

Dentro del MERCOSUR, se implementó un Mecanismo Experimental de Acreditación de Carreras de Grado Universitario. En el área de ingeniería el documento de la Comisión Consultiva [32] define una serie de requisitos para la experiencia práctica y de laboratorio.

En la sección de Características Académicas de las carreras se pide que se expliciten las horas de laboratorios y prácticas como porcentaje de las horas totales. Si bien en esta oportunidad experimental no se establecen mínimos para dichas actividades y solo se expresa lo siguiente:

> "Las actividades académicas deben contemplar aulas teóricas, prácticas y experiencias de laboratorio y su distribución debe fijarse en porcentaje."[32]

En versiones preliminares de los estándares a cumplir se incluía:

> "Se propone una distribución variable con un mínimo de aulas teóricas del 50% y prácticas y experiencias de laboratorio del 20% de la carga horaria total."[3]

Los laboratorios se nombran y se jerarquizan a lo largo de todo el documento, desde el diseño de las asignaturas, las actividades de aulas, como parte del proceso enseñanza-aprendizaje, y como actividades orientadas a desarrollar capacidades específicas declaradas en el perfil del egresado.

Hay un ítem específico (4.3) en el que se exige la existencia de laboratorios e instalaciones especiales; adecuadamente equipadas según las necesidades y exigencias del plan de estudios y al número de estudiantes, en las cuales los alumnos puedan participar en forma activa.

### 2.2.1 Análisis

Analizando el contexto expuesto anteriormente podemos ver que la realidad actual en la enseñanza de la ingeniería es muy compleja y está en un proceso de cambios importantes. Las tecnologías favorecen la diversidad de propuestas, y las herramientas

[3] Extractado de una versión preliminar no aprobada.

teóricas y metodológicas aún no tienen respuestas para algunos temas como por ejemplo la evaluación y certificación de los laboratorios a distancia.

El abanico de posibilidades es muy grande, intentando realizar una taxonomía de las alternativas actuales a las experiencias de laboratorio en ingeniería en general, y a laboratorios de diseño electrónico digital en particular podríamos clasificarlas en:

*Laboratorios reales*

Esta es la alternativa tradicional en la cual los cursos cuentan con laboratorios donde los estudiantes concurren a realizar las prácticas de diseño. Esto puede formar parte del mismo curso o bien puede haber cursos o módulos de laboratorios separados.

*Simuladores*

Si se cuenta con computadoras, una alternativa es utilizar alguna herramienta software de diseño y llegar hasta la simulación de los mismos. Pensamos que esto sin lugar a dudas es mejor que la realización exclusiva de los diseños en papel, pero que no permite tener una visión completa del proceso de diseño, y podemos afirmar por experiencia propia que muchos problemas interesantes se dan a la hora de llevar un diseño a hardware. Con el uso de simuladores es posible depurar un diseño hasta cierto punto, pero no permiten realizar una depuración completa del mismo. La depuración o *debugging* del diseño es una tarea esencial en ingeniería.

*Laboratorios reales controlados a distancia, laboratorios remotos*

Recientemente se ha presentado otra alternativa que es la realización de laboratorios a distancia o remotos, esto puede o no ir acompañado de un curso a distancia; encontrándose una gran cantidad de publicaciones en el tema [74, 75, 82, 108]. Una breve historia de los laboratorios remotos puede consultarse en [84]. Esta alterativa no es fácil de implementar ya que se requieren laboratorios equipados con hardware que pueda ser configurado en forma remota. Además si bien se estaría cumpliendo con la premisa que el diseño realizado por el estudiante es llevado a hardware, el resultado del mismo sólo se está viendo también en forma remota. Pueden probarse con vectores de test, y obtener los vectores de resultados asociados, pero todo se hace a

través de computadoras y redes y el efecto "caja negra" es muy grande para el estudiante que está alejado manejando una computadora. En definitiva nos parece que, si bien en algunas áreas de ingeniería estos laboratorios pueden ser muy útiles, por ejemplo en aquellas que involucran elementos costosos, robots o plantas a controlar, en electrónica digital no se gana demasiado en comparación con la simulación, la mayor diferencia es que hay que aprender a utilizar herramientas de configuración y test de hardware vía redes.

*Laboratorios virtuales*

Es una alternativa similar a la simulación, pero con paquetes de software que dan la sensación de estar armando un experimento, ya que en la pantalla de la computadora se pueden ver símiles de instrumentos. Nuevamente pensamos que en el caso de electrónica digital esta alternativa no ofrece ventajas significativas.

Creemos que la alternativa que se presenta en el Capítulo 4 mantiene las características de experimentación con hardware real y a su vez tiene ciertas características de educación a distancia, sin requerir el montaje de un laboratorio tradicional, con los costos que esto trae aparejados; aunque sí consume una buena cantidad de horas docentes.

## 2.3 Antecedentes de los dispositivos lógicos programables en la Universidad de la República

En el año 1992 el IIE se propuso formar un ambiente de desarrollo y un equipo humano capacitado en la utilización de dispositivos lógicos programables para el diseño de electrónica digital y su posterior difusión al medio local. En ese sentido se seleccionó y adquirió el equipamiento necesario para comenzar a trabajar y se elaboró

un proyecto de desarrollo tecnológico que fue aprobado por la Comisión Sectorial de Investigación Científica de la Universidad.

En diciembre de 1992 se grabó el primer integrado y se realizó una presentación y debate sobre el tema con asistencia de profesionales del medio. En este marco se han realizado actividades de asesoramiento a empresas de electrónica del medio y se han incorporado dispositivos lógicos programables en proyectos de fin de carrera culminados con éxito.

En 1994 se trasladó esta nueva tecnología a la enseñanza de grado incorporando el uso de dispositivos lógicos programables en el laboratorio del curso de grado Diseño Lógico, y en diciembre de ese mismo año se realizó un curso de actualización y de grado en el IIE, el cual volvió a dictarse en 1995, 1996 y de 1998 en adelante. Este curso tiene un enfoque fuertemente práctico, y está estructurado en tres partes: introducción teórica, laboratorio y proyecto final [4].

La utilización de lógica programable en laboratorios ha demostrado ser muy efectiva, ya que ahorra mucho tiempo con problemas tecnológicos y de cableado, y permite que un diseño se lleve rápidamente del papel al circuito.

El hecho que Diseño Lógico sea el primer curso donde el estudiante de Ingeniería Eléctrica toma contacto con circuitos electrónicos genera gran entusiasmo y motivación en los estudiantes, que ven funcionar sus diseños.

Se apunta a una enseñanza de calidad a partir de la propuesta de cursos en donde el estudiante tenga un rol activo en su aprendizaje. Los laboratorios son una herramienta para promover ese rol. Se sostiene, a nivel conceptual, que cuanto más activo sea el aprendizaje, tendrá un mayor grado de significación para el estudiante y por tanto será mejor comprendido y obtendrá un conocimiento más duradero.

# Capítulo 3

# Dispositivos Lógicos Programables

Los dispositivos lógicos programables pueden ser definidos como circuitos lógicos digitales en los cuales el usuario puede grabar el circuito a implementar.

En este capítulo se presentará una completa descripción de estos dispositivos así como su evolución histórica desde los primeros dispositivos programables pensados para simplificar diseños digitales realizados con compuertas discretas, hasta los modernos dispositivos reconfigurables en los cuales se puede implementar una aplicación completa (SoPC, System on a Programmable Chip).

Los dispositivos lógicos programables pueden clasificarse de acuerdo a la tecnología utilizada en su fabricación, que estará directamente asociado a la forma de programarse, y de acuerdo a su complejidad. Estas clasificaciones no son ortogonales ya que la tecnología utilizada además de determinar el tipo de programación posible determinará, junto con la densidad y la arquitectura interna, el grado de complejidad del dispositivo.

La primer idea de implementación de un dispositivo lógico reprogramable, es decir con programación dinámica, es atribuida a Wahlstrom que en 1967 [109] propone un circuito con estas características. La tecnología de integración de circuitos utilizada en esa época no permitía incluir gran cantidad de transistores en un chip y por lo tanto la idea de Wahlstrom no tiene demasiada repercusión, ya que se debían utilizar una gran cantidad de transistores adicionales solo para programar el dispositivo. Pasarían casi

dos décadas hasta que un dispositivo conceptualmente similar fuera desarrollado por Xilinx en 1985 [27] y patentado por Freeman [46]. Para esa época la patente de Wahlstrom ya estaba caduca [107].

Entre estos dos hitos históricos aparecen los dispositivos programables con programación no volátil, y que en principio tienen una evolución propia, tanto en el modo de programación como en la estructura interna, que mayoritariamente están basados en lógica en dos niveles. Una completa historia de los PLDs puede verse en Pellerin and Holley [94] y en Pellerin and Thibault [95]

Hoy en día la frontera entre CPLDs y FPGAs es mucho más borrosa, ya que por ejemplo una de las últimas familias de CPLDs desarrolladas por Altera, la familia MAX, si bien tiene programación no volátil, la misma está almacenada en una memoria Flash interna al chip, y se carga en una SRAM de configuración al inicializar el dispositivo.

En la Tabla 3-1 pueden verse los distintos tipos de FPGAs comerciales y sus características principales.

Los precios de la tabla fueron obtenidos directamente de los sitios web de los distribuidores: Avnet Memec, Arrow, Mouser y Future Electronics. Se eligieron los dispositivos de menor y mayor precio, y salvo las excepciones indicadas es el precio unitario.

La frecuencia máxima es sacada directamente de las hojas de datos de los fabricantes o en su defecto la calculada para un contador binario de 16 bits.

En los bloques hardware sólo se mencionan aquellos que pueden usarse para aumentar la capacidad lógica del dispositivo, no se incluyen PLLs, manejadores de relojes, etc.

| Manufacturer Family Type | Type | | Eq. System gates (k) | RAM bits (k) | User I/O | HW blocks | Freq. Max. | U$S Min. | U$S Max. |
|---|---|---|---|---|---|---|---|---|---|
| **Actel** | | | | | | | | | |
| Axcelerator | FPGA | antifuse | 125-2000 | 29-338 | 168-684 | -- | 500MHz | 9,9[c] | 145,1[c] |
| ProASICPLUS | FPGA | flash | 75-1000 | 27-198 | 158-712 | -- | 150MHz | 4,9[c] | 140,2[c] |
| Fusion | FPGA | flash | 90-1500 | 27-270 | 75-252 | A/D, Analog Quad | 350MHz | | |
| IGLOO | FPGA | flash, low power | 30-1000 | 0-144 | 81-300 | -- | | 1,5 | [b] |
| IGLOO/e | FPGA | flash, low power | 600-3000 | 108-504 | 270-616 | -- | | [b] | [b] |
| **Altera** | | | | | | | | | |
| MAX 3000A | CPLD | eeprom | 0.6-10 | -- | 34-208 | -- | 227MHz | 1,2 | 70 |
| MAX II | CPLD | flash, low power | 3.6-32[a] | -- | 80-272 | -- | 212MHz | 6 | 92,8 |
| Acex[d] | FPGA | sram | 10-100 | 12-49 | 136-333 | -- | 285MHz | 52 | 103 |
| Cyclone II | FPGA | sram | 180-2800[a] | 120-1152 | 158-622 | 13-150 mult 18x18 | 445MHz | 13,9 | 384 |
| Stratix | FPGA | sram | 422-3300[a] | 920-7428 | 426-1203 | 24-88 mult 18x18 | 420MHz | 205 | 10280 |
| Stratix II | FPGA | sram | 600-8000[a] | 419-9383 | 366-1170 | 12-96 mult 18x18 | 450MHz | 190 | 10688 |
| **Atmel** | | | | | | | | | |
| ATF15 | CPLD | eeprom | 1.5-12 | -- | 36-212 | -- | 200MHz | 1,8 | 22,47 |
| AT40 | FPGA | sram | 5-50 | 2-18 | 128-384 | -- | 100MHz | 13 | 84 |
| AT6000[d] | FPGA | sram | 6-30 | -- | 96-124 | -- | 70MHz | 21,5 | 180 |
| **Cypress** | | | | | | | | | |
| Delta39K | CPLD | sram o flash | 30-200 | 80-480 | 174-428 | -- | 233MHz | 59 | 190 |
| Quantum38[d] | CPLD | sram | 30-100 | 16-48 | 174-302 | -- | 125MHz | | |
| Ultra37000 | CPLD | eeprom | 0,96-15 | -- | 37-269 | -- | 200MHz | 3,7 | 136 |
| **Lattice** | | | | | | | | | |
| ispXPLD | CPLD | eeprom + sram | 75-300 | 128-512 | 141-381 | -- | 300MHz | 40,77 | 238,5 |
| ispMATCH | CPLD | eeprom | | -- | 32-208 | -- | 400MHz | 2,63 | 73,44 |
| LatticeSC | FPGA | sram | | 1030-7800 | 300-942 | ASIC blocks | 370MHz | | |
| ispXGPA | FPGA | eeprom | 139-1250 | 92-414 | 176-496 | -- | 290MHz | 25,5c | 383c |
| LatticeECP-DSP | FPGA | sram | 400-3000 | 117-709 | 224-576 | 0-32 mult 18x18 | 414MHz | 8,6 | 139,8 |
| MachXO | CPLD | flash + sram | | 9.2-27.6 | 78-271 | -- | 388MHz | 6,7 | 44,1 |
| **Quicklogic** | | | | | | | | | |
| Eclipse | FPGA | antifuse | 248-583 | 46-83 | 250-347 | -- | 450MHz | 33 | 387 |
| Eclipse II | FPGA | antifuse | 47-321 | 9-55 | 92-310 | 0-12 mult 8x8, 16 bit adder | 250MHz | 8 | 191 |
| PolarPro | FPGA | antifuse, low power | 75-1000 | 37-221 | 172-652 | -- | 200MHz | 14 | 177 |
| **Xilinx** | | | | | | | | | |
| CoolRunner II | CPLD | eeprom | 0,75-12 | -- | 33-270 | -- | 323MHZ | 1,4 | 86 |
| Spartan II | FPGA | sram | 50-600 | 32-288 | 182-514 | -- | 200MHz | 8,45 | 67,8 |
| Spartan-3E | FPGA | sram | 100-1600 | 72-648 | 108-376 | 4-36 mult 18x18 | 230MHz | 7,9 | 63 |
| Spartan-3L | FPGA | sram | 4000 | 1728 | 391-712 | 24-96 mult 18x18 | 250MHz | 55 | 281 |
| Spartan-3 | FPGA | sram | 50-5000 | 1872 | 124-784 | 4-104 mult 18x18 | 250MHz | 15,5 | 448 |
| Virtex II | FPGA | sram | 40-8000 | 72-3024 | 88-1108 | 4-168 mult 18x18 | 420MHz | 85,36 | 4177 |
| Virtex II Pro | FPGA | sram | ?-8000 | 216-7992 | 204-1144 | 12-444 mult 18x18 0-2 Power PC | 309MHz | 61 | 12223 |
| Virtex 4 | FPGA | sram | 960-15300 | 648-9936 | 320-960 | 32-512 mult 18x18, adder, accum 0-2 PowerPC | 500MHz | 118 | 9809 |
| Virtex 5 | FPGA | sram | | 1152-10368 | 400-1200 | 32-192 mult 25x18 | 550MHz | [b] | [b] |

[a] **System Gates estimadas en base a dispositivos similares de [106]**

[b] Precios obtenidos de [36], 10.000 unidades
[c] Dispositivo recién salido al mercado, no pudieron obtenerse precios
[d] Dispositivos maduros no recomendados para nuevos diseños

**Tabla 3-1 Principales Dispositivos Lógicos Programables comerciales y sus características**

En algunos casos fue imposible calcular las *system gates* equivalentes para un determinado dispositivo, particularmente en aquellos casos en que hay bloques hardware.

En las secciones siguientes de este capítulo se analizarán diferentes tipos de dispositivos lógicos programables.

## 3.1 Almacenamiento de la configuración

La configuración de los dispositivos lógicos programables debe almacenarse de algún modo dentro del dispositivo, si analizamos las diferentes formas de almacenamiento de la configuración veremos que se utilizan las mismas que para las memorias de almacenamiento de datos, y estas son:

PROM (Programmable Read-Only Memory), o también llamados dispositivos OTP (One Time Programmable). Estos dispositivos solo pueden programarse una vez. Tradicionalmente son dispositivos de bajo costo que se utilizan en producción una vez que el diseño a programar está terminado y probado.

EPROM (Erasable Programmable Read-Only Memory). Estos dispositivos permiten múltiples programaciones permitiendo el borrado con luz ultra-violeta.

EEPROM (Electrically Erasable Programmable Read-Only Memory). Similares a los anteriores, pero el borrado se realiza en forma eléctrica. Los dispositivos más primitivos se borran utilizando tensiones especiales más elevadas que los niveles lógicos utilizados por el dispositivo en funcionamiento. Las versiones más nuevas ISP

(In System Programability) permiten el borrado y la nueva programación utilizando niveles lógicos de voltaje de modo que no es necesario extraer el integrado del sistema y ponerlo en un programador, sino que se puede programar en el propio sistema en el cual es utilizado.

PLICE (Programmable Low-Impedance Circuit Element) Antifuse [25], esta tecnología es propietaria de Atmel y actúa al revés que un fusible: cuando no es programado es un aislante, cuando se programa se genera un camino de baja resistencia.

ViaLink Antifuse es la tecnología utilizada por Quicklogic, funciona igual que el PLICE Antifuse, pero presenta menor resistencia y menor capacidad parásita. (300 ohms para el PLICE y 50 ohms para Vialink)

Flash SRAM (Static Random Access Memory), las memorias Flash tienen la ventaja de ser eléctricamente borrables como las EEPROM pero utilizan celdas de menor tamaño parecidas a las utilizadas por las EPROM.

SRAM (Static Random Access Memory). La programación se escribe exactamente igual que como se escriben datos en una RAM estática. Estos dispositivos son los que introducen el cambio cualitativo que los hace reconfigurables. Pueden ser reconfigurados repetidas veces en forma muy rápida, sin alterar la vida útil del dispositivo.

Los dispositivos basados en SRAM son los únicos que permiten una configuración dinámica. Si bien en los demás dispositivos (excepto los basados en PROM y Antifuse), existe la posibilidad de configurarlos múltiples veces, la cantidad de configuraciones que aceptan está limitada a unas 10.000 y su configuración es lenta. Por el contrario los dispositivos basados en SRAM permiten “infinitas” reprogramaciones y por lo tanto son los únicos que pueden ser utilizados para aplicaciones de lógica reconfigurable, tanto en aplicaciones clásicas como en aplicaciones que requieran reconfiguración en tiempo de corrida (run-time).

Algunos de los dispositivos basados en SRAM incorporan una característica extra que es la reconfiguración parcial. Esto quiere decir que se puede configurar una parte del dispositivo mientras el resto del mismo sigue en operación normal. La reconfiguración parcial tiene múltiples ventajas: permite tener varios diseños en un mismo chip y cambiarlos por partes, permite reducir el tiempo de programación de cada parte, y por lo tanto simplifica su utilización para reprogramaciones en tiempo de corrida.

Los tiempos de configuración dependen de la cantidad de bits de configuración de cada chip y de la máxima frecuencia de reloj con la cual cada dispositivo puede ser programado. Para poner algún ejemplo cargar la configuración completa en un Xilinx XCV1000E, con unas 27.500 celdas lógicas, lleva aproximadamente 12,5 ms [119].

Las últimas innovaciones en configuración de dispositivos incluyen la compresión/descompresión del flujo de datos de configuración como forma de ahorrar tiempo y espacio de almacenamiento; y el encriptado de los datos para proteger los diseños. También se han incluido formas de programación remota para poder actualizar versiones en sistemas ya diseñados.

## 3.2 Arquitecturas de dispositivos programables

Veremos primero tres tipos de dispositivos basados en estructuras AND-OR, o más genéricamente diseños en dos niveles, que pueden resumirse en la siguiente figura:

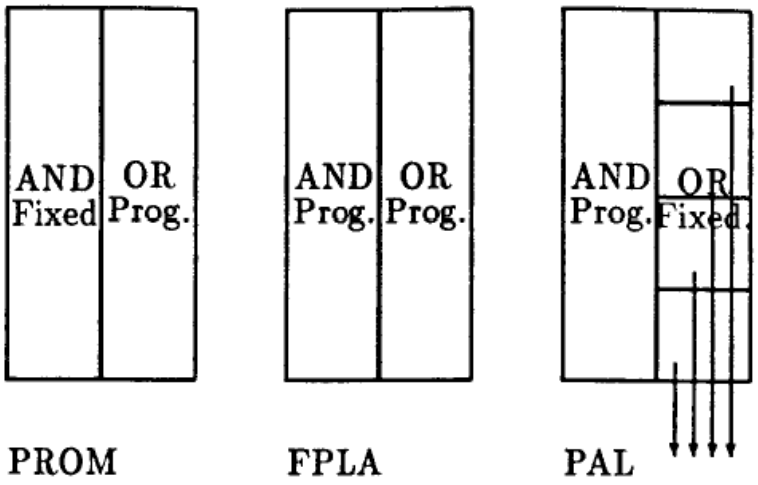


**Fig. 3-1 Diferentes implementaciones en dos niveles AND-OR (extraído de [69])**

### 3.2.1 Programmable Read Only Memory PROM

Las memorias usualmente se utilizan para almacenamiento de datos, pero pueden ser vistos como circuitos digitales que pueden implementar funciones booleanas.

Desde el punto de vista lógico una memoria PROM de N bits de direcciones está constituida por un primer nivel de compuertas ANDs completo (todos los minitérminos posibles), y un segundo nivel de compuertas OR (tantas como salidas) programable. Con esta configuración es posible implementar cualquier función lógica de N variables.

Las primeras PROMs fueron desarrolladas por Harris en 1970, y en 1971 Intel introduce las EPROMs borrables con luz ultravioleta.

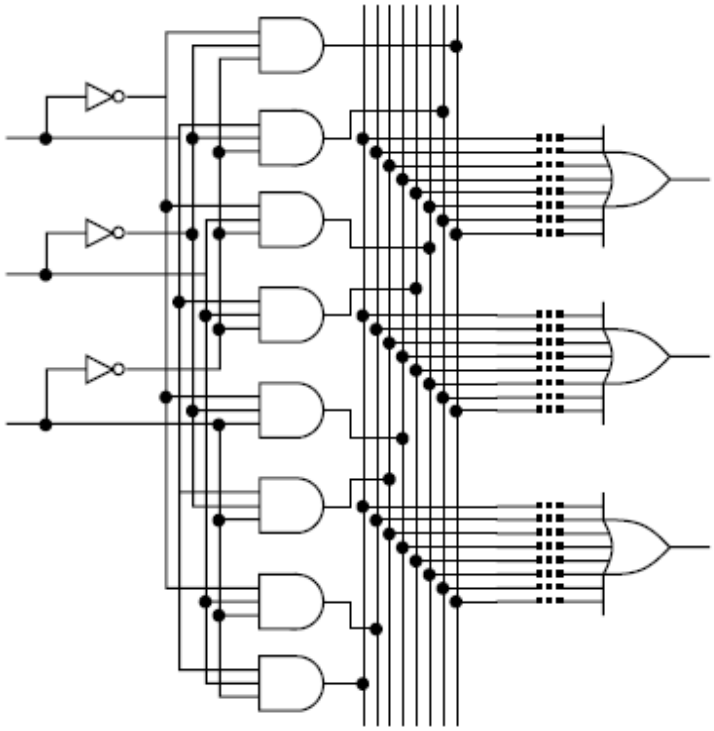

**Fig. 3-2 Esquema lógico en dos niveles de una PROM (extraído de [52])**

El problema con la utilización de PROMs para la implementación de funciones booleanas es que usualmente son dispositivos de baja velocidad.

### 3.2.2 Programmable Logic Array PLA

Los primeros dispositivos lógicos programables comerciales fueron las PALs. Estos dispositivos estában pensados para implementar funciones lógicas en dos niveles AND-OR, pero no tenían el plano AND completo, sino programable. De esta forma no es posible implementar cualquier función lógica de un determinado número de entradas sino sólo aquellas funciones que sean minimizables y puedan ser representadas con la cantidad de términos producto disponibles dentro del chip. El segundo nivel o plano OR también es programable.

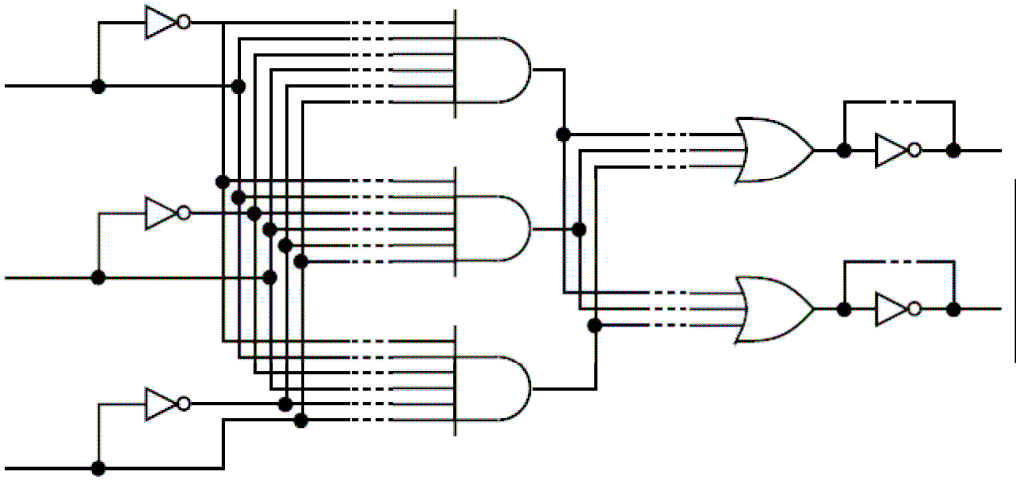

**Fig. 3-3 Esquema de una PLA (extraído de [52])**

Estos dispositivos fueron comercializados a principios de los ‘70s pero no tuvieron mayor éxito debido a su baja performance, a su alto costo y a que no existían herramientas CAD que corrieran en computadores de bajo costo.

### 3.2.3 PALs (Programmable Array Logic)

Los PALs introducidos en 1978 por Monolithic Memories Inc (MMI) [94], fueron la segunda generación de dispositivos desarrollados y básicamente son una simplificación de los PLAs. Este tipo de dispositivos mantiene la estructura de dos planos AND-OR, pero mientras el plano AND es programable, el plano OR se simplifica y pasa a ser fijo. Al reducir las interconexiones programables se logra mayor velocidad.

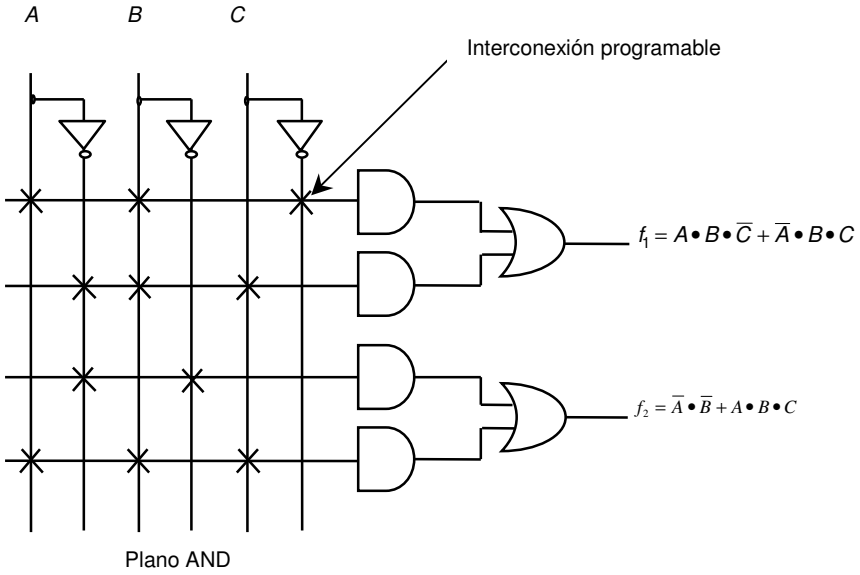


**Fig. 3-4 Celda básica con plano AND programable y OR fijo, a la salida de cada función lógica se incorpora un FF (adaptado de [53])**

Estos dispositivos tuvieron mayor éxito comercial que los anteriores y se utilizaron ampliamente en la sustitución de diseños realizados con lógica discreta del tipo "glue-logic".

Además de poder realizar funciones booleanas se agrega un flip-flop por cada compuerta OR, de modo de poder realizar circuitos secuenciales. Así aparece el concepto de celda, macrocelda (macrocell) o CLB (Configurable Logic Block), un bloque programable capaz de implementar una determinada función lógica más una celda de memoria o flip-flop.

### 3.2.4 Cellular Arrays

Otro tipo de dispositivos programables que presenta un enfoque diferente de los vistos anteriormente, están formados por un conjunto de elementos simples interconectados entre sí, pero en ellos las interconexiones son básicamente fijas, siendo programable la función que realiza cada elemento.

Antecedentes de cellular arrays pueden encontrarse a fines de los años ’60 y principios de los ’70 (Kautz [67], Minnick [81], Shoup 1970 [[102]]), pero sus primeras versiones comerciales datan de mucho tiempo después, a fines de los ‘80 Algotronix desarrolla el dispositivo llamado Cellular Array Logic or CAL [68, 69].

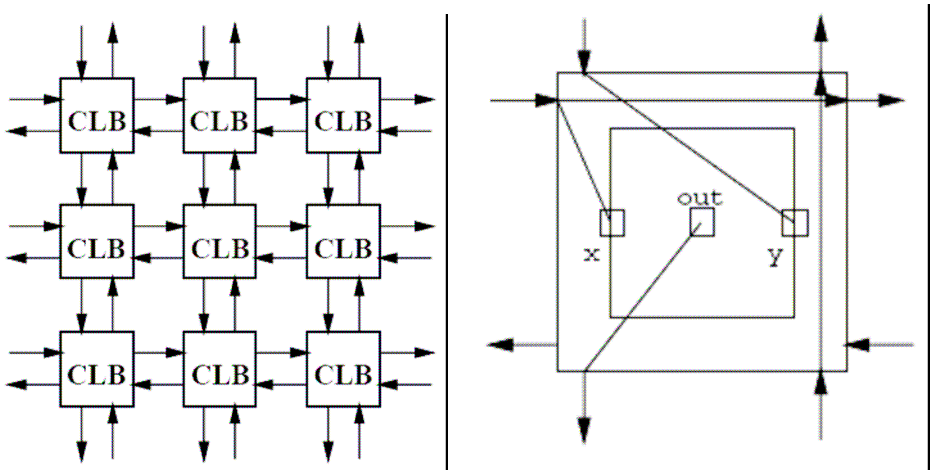


**Fig. 3-5 Arquitectura y celda lógica del Algotronix CAL (extraído de [52])**

En las figuras puede verse el arreglo bidimensional de bloques, llamados Configurable Logic Block, or CLBs. Un CLB puede realizar cualquier operación booleana de dos entradas e interconectarse con los bloques vecinos.

La configuración del dispositivo era dinámica, basada en RAM.

### 3.2.5 PLDs (Programmable Logic Device) y CPLDs (Complex Programmable Logic Device)

Los CPLDs son herederos directos de las PALs y podrían clasificarse en la misma categoría. La celda básica es igual y está realizada en dos niveles AND-OR, con el primer nivel AND programable y el nivel OR fijo; y con el elemento de memoria o Flip-Flop a la salida del OR. El almacenamiento de la programación es por EEPROM o Flash.

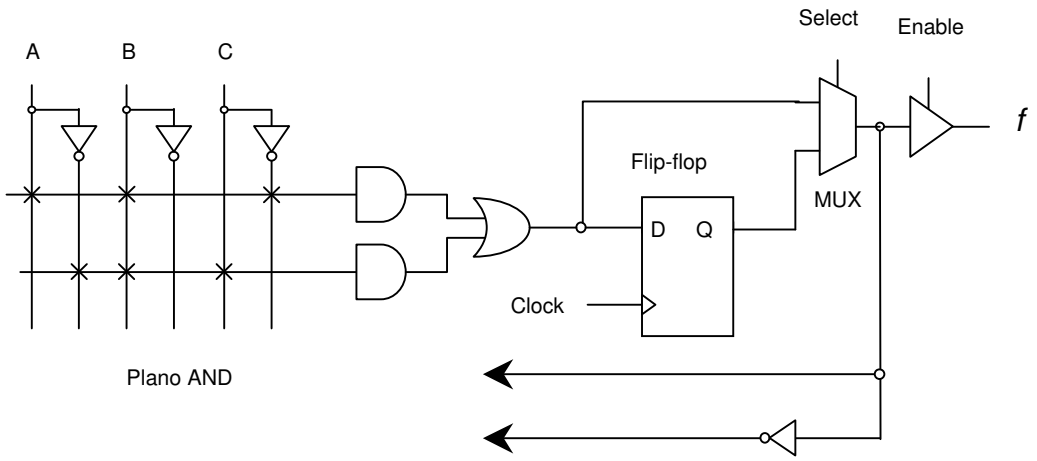


**Fig. 3-6 Esquema genérico de un PLD, plano AND variable, plano OR fijo, FF y realimentación (adaptado de [53])**

Permiten la realimentación de la salida de una celda hacia la matriz de interconexión; de esta forma se puede lograr sintetizar funciones lógicas complejas de más de dos niveles o máquinas de estados si se realimentan las salidas de los FF.

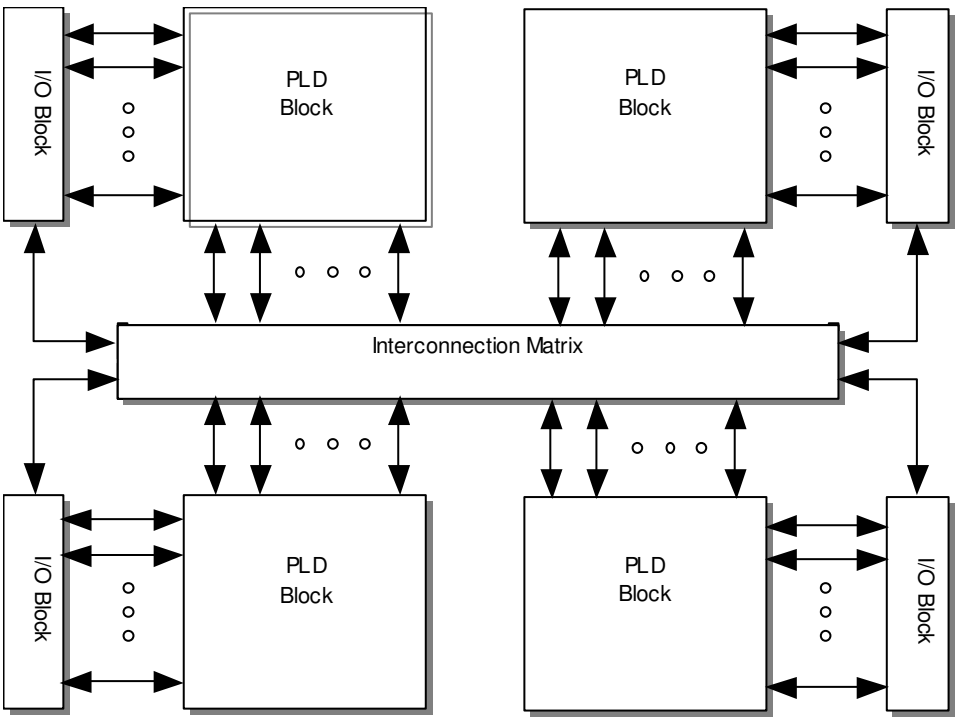


**Fig. 3-7 Interconexión de bloques dentro de un PLD (adaptado de [53])**

Poseen matrices de interconexiones programables, y usualmente las celdas están agrupadas en una estructura jerárquica, con conexiones rápidas entre vecinos. Hay una diferenciación de bloques de entrada-salida.

Como ejemplo de arquitectura comercial veremos la serie MAX 3000A de Altera

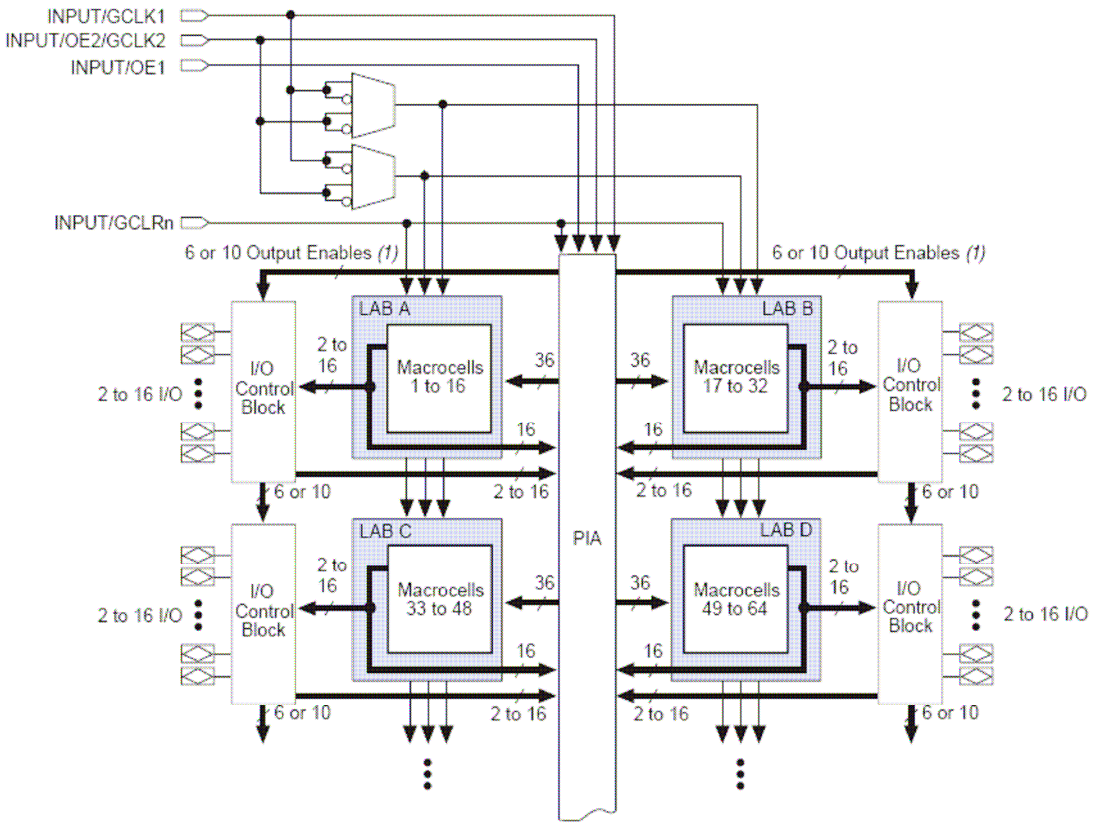


**Fig. 3-8 Altera serie MAX 3000A, estructura de bloques [12]**

Diagrama de bloques de la serie MAX 3000A [12], el dispositivo está compuesto de LABs (Logic Array Blocks), y cada LAB está formado por 16 Macrocells

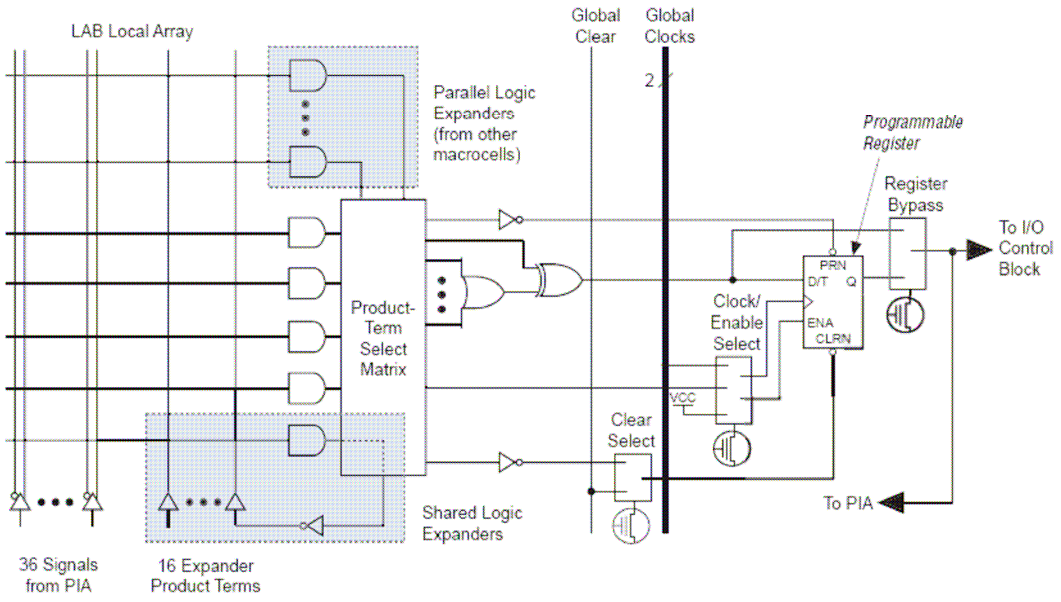


**Fig. 3-9 Altera MAX 3000A, estructura de celda [12]**

Cada Macrocell está formada por una matriz de selección de términos producto, y pueden seleccionarse hasta 5 términos producto. Existen dos tipos de Expanders que pueden ser compartidos entre Macrocells vecinas. Cada Macrocell dispone de un Flip-Flop que puede configurarse como D, JK, SR, o T. Dicho Flip-Flop puede ser salteado para implementaciones puramente combinatorias. Se proveen señales globales de reloj y habilitación de tri-state de las salidas.

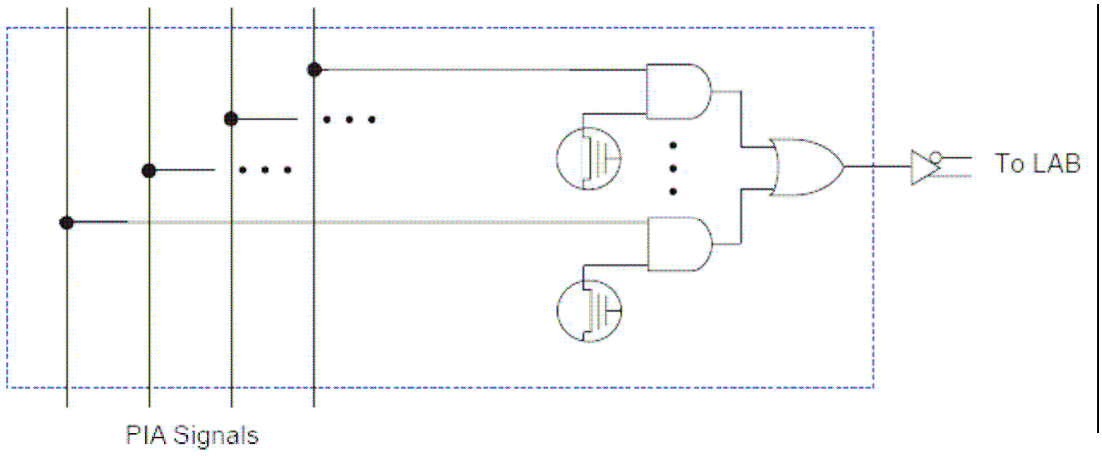


**Fig. 3-10 Altera MAX 3000A, interconexiones [12]**

El Programmable Interconnect Array (PIA) es un bus global que permite conectar cualquier entrada a cualquier salida del array.

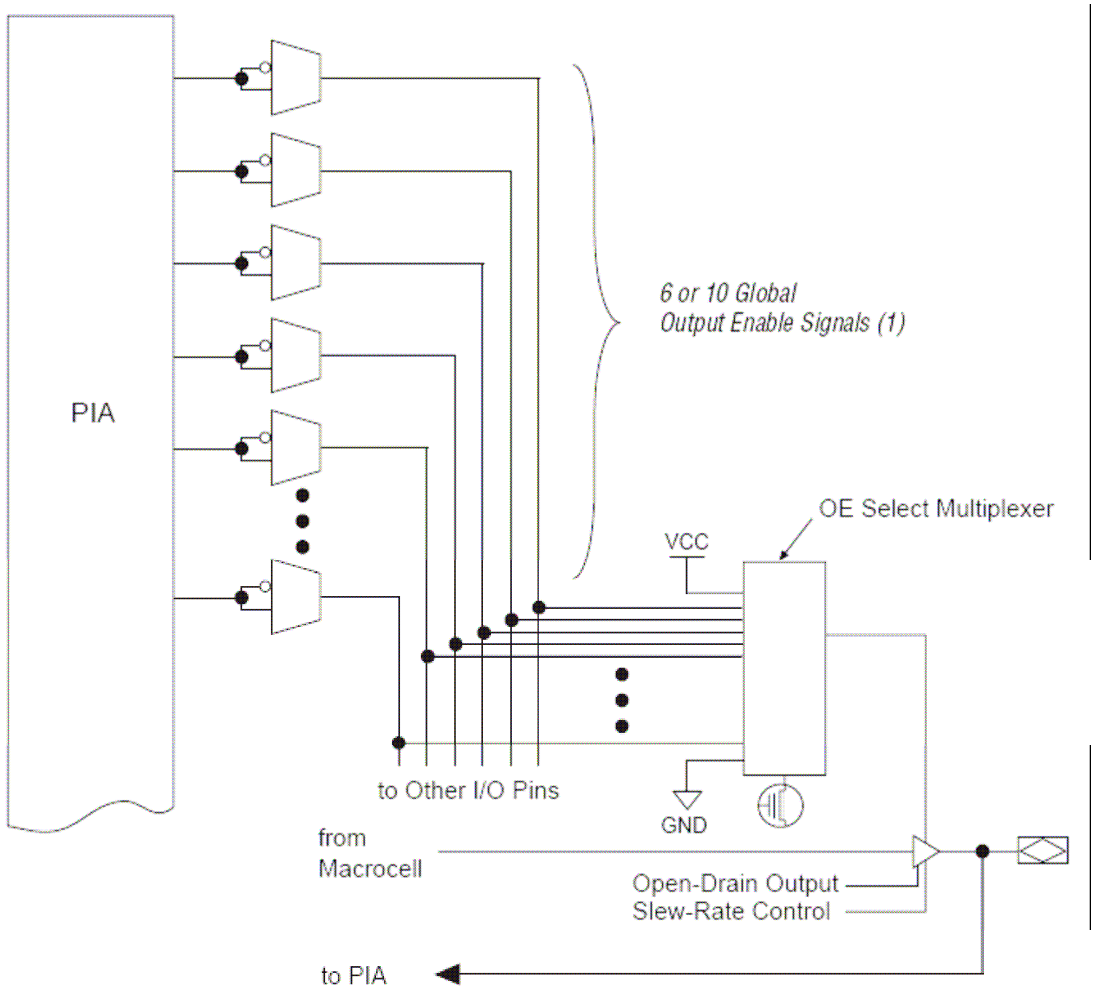


**Fig. 3-11 Altera MAX3000A, esquema de entrada-salida [12]**

Las señales del PIA pueden conectarse a los bloques de entrada/salida. Cada bloque de IO permite su configuración como entrada, como salida, o en forma bidireccional. A la salida hay un buffer tri-state configurable.

### 3.2.6 FPGAs (Field Programmable Gate Array)

En 1984 Xilinx desarrolla un nuevo dispositivo que denomina Logic Cell Array (LCA), basado en un concepto diferente a los PLDs. Los LCAs están compuestos de una gran cantidad de celdas lógicas cuya función es programable, dichas celdas pueden ser interconectadas mediante conexiones programables de varios tipos. Estos dispositivos y sus desarrollos posteriores dan origen a las hoy llamadas FPGAs.

Las FPGAs están basados en una estructura regular de bloques de procesamiento e interconexiones programables, rodeados de bloques dedicados a entrada salida (ver Fig. 3-12)

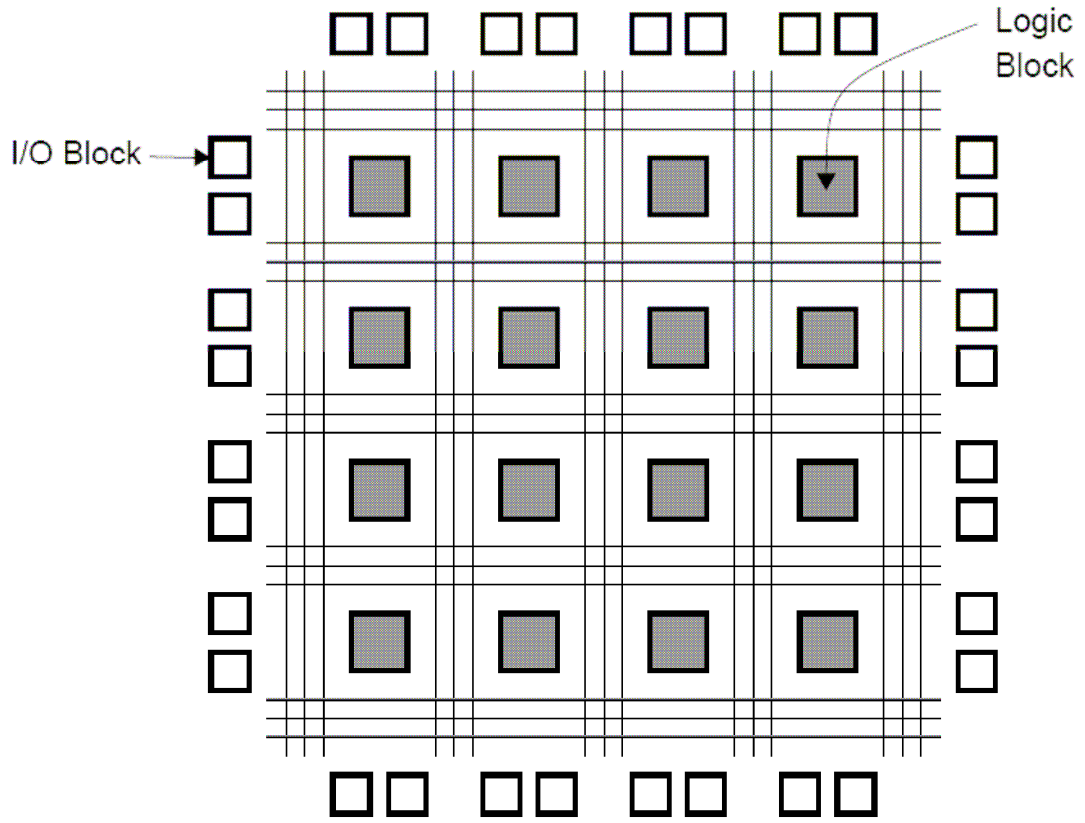


**Fig. 3-12 Esquema interno de una FPGA (extractado de [25])**

Las interconexiones usualmente están organizadas en forma de una malla jerárquica, disponiéndose de caminos rápidos entre bloques contiguos, caminos verticales y caminos horizontales. De esta forma los elementos de procesamiento forman una *isla* rodeada de líneas de interconexión.

Los elementos de procesamiento pueden realizar funciones simples de un bajo número de entradas para dar como resultado una única salida. Esos bloques o celdas internamente están compuestos por una Look-Up Table (LUT) más algún elemento de memoria o flip-flop. El tamaño de la LUT más utilizado es de cuatro entradas, que

permite implementar cualquier función lógica de cuatro entradas, o lo que es lo mismo una tabla de verdad de 16 renglones.

Hay varias investigaciones que muestran que las LUTs de cuatro entradas dan buenos resultados en cuanto a su eficiencia en área y velocidad [96, 97]; y este resultado ha sido utilizado comercialmente.

Los bloques de procesamiento de las FPGAs comerciales usualmente son un poco más complejos que una LUT más un FF, permitiendo mayor flexibilidad. Algunas de la características buscadas son la posibilidad de realizar funciones combinatorias de mayor número de entradas, la posibilidad de realizar bloques aritméticos con acarreo, la posibilidad de aprovechar un bloque que ha sido parcialmente utilizado, permitiendo por ejemplo usar por un lado la salida de la función combinatoria y el elemento de memoria por separado, o la incorporación de un mayor número de elementos de memoria por bloque.

Usualmente estos bloques de procesamiento se agrupan, los bloques que pertenecen a un mismo grupo o *cluster* tienen interconexiones locales, esto redunda en una mayor velocidad de interconexión y en el ahorro de recursos globales. El tamaño de estos *clusters* es una característica que influye en la performance de una FPGA. En dispositivos comerciales es usual ver clusters grandes, compuestos por entre 8 y 10 elementos lógicos. Se han hecho varios estudios sobre el efecto del tamaño de los clusters en el área y la velocidad de una FPGA, así como su interacción con el tamaño de las LUTs, los resultados muestran que los valores óptimos están entre 4 y 10 [8, 76].

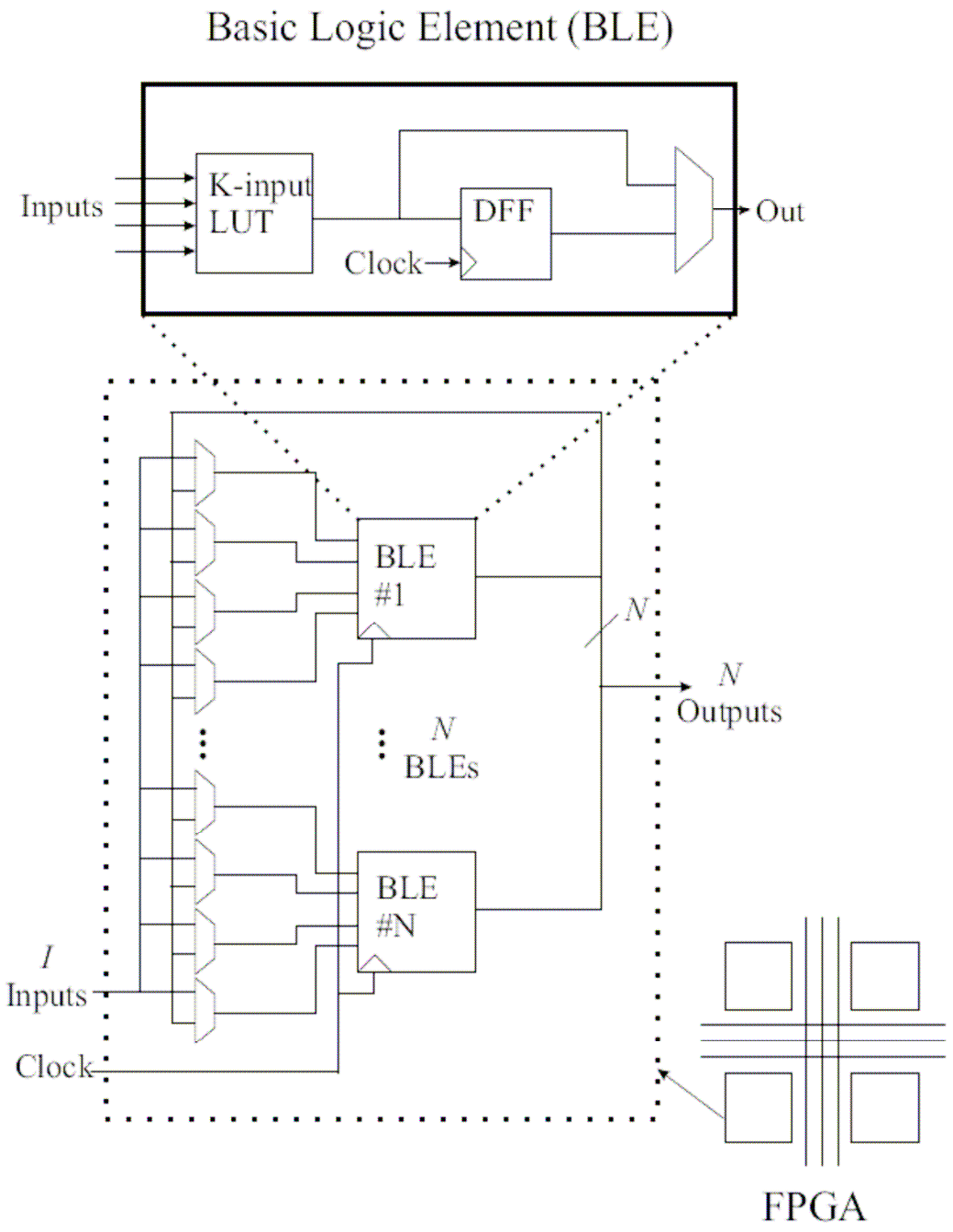


**Fig. 3-13 Estructura de un bloque de procesamiento (Basic Logic Element, BLE) y un Cluster (extraído de [8])**

El tamaño de los clusters también influye en las herramientas de CAD, el tiempo de compilación de un diseño aumenta con clusters pequeños [8, 76].

Las interconexiones ocupan un lugar muy importante dentro de los integrados, ya sea en términos de área como en los retardos producidos [22]. Se puede estimar que el porcentaje del área utilizado para las interconexiones está entre 70 y 90% del área total del chip [97]. Las relaciones de área entre lógica, interconexiones y memoria de configuración pueden verse en la Fig. 3-14

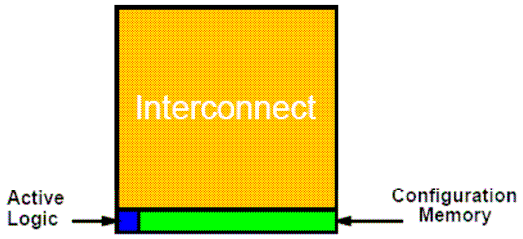


**Fig. 3-14 Relación interna de áreas dentro de un FPGA [extractado de [35]]**

Si bien las primeras FPGAs tenían una estructura simétrica tipo isla, con bloques lógicos y caminos de interconexión verticales y horizontales rodeándolos, las FPGAs actuales tienen estructuras jerárquicas tanto en los bloques lógicos, que se agrupan en clusters como ya fue mencionado, como en las interconexiones que están organizadas en caminos de distinta longitud y retardo.

Un ejemplo del uso de interconexiones programables controladas por celdas de SRAM puede verse en la Fig. 3-15. Se utilizan bits de SRAM para controlar el estado de transistores de paso y líneas de control de multiplexores [25].

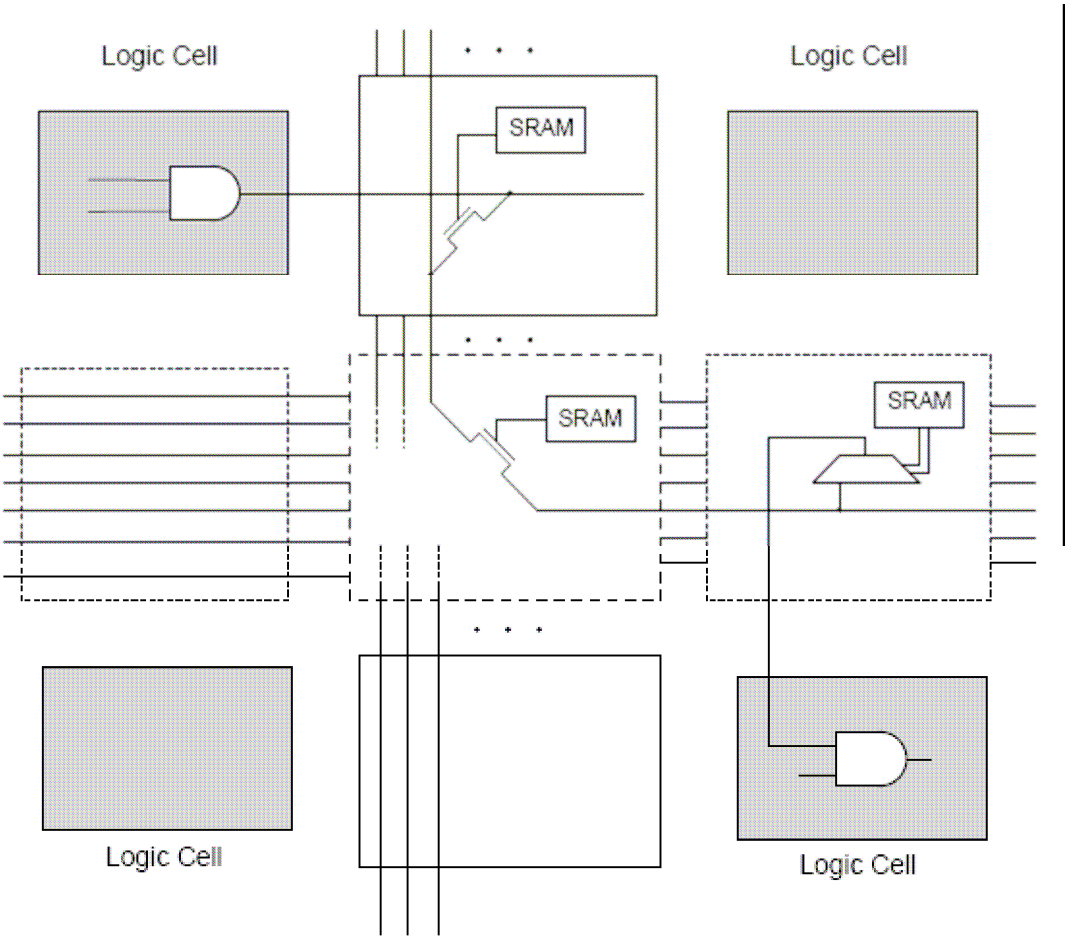


**Fig. 3-15 Interconexiones programables en una FPGA (extractado de [25])**

Los bloques de interconexión (switch blocks) se intercalan entre los bloques lógicos.

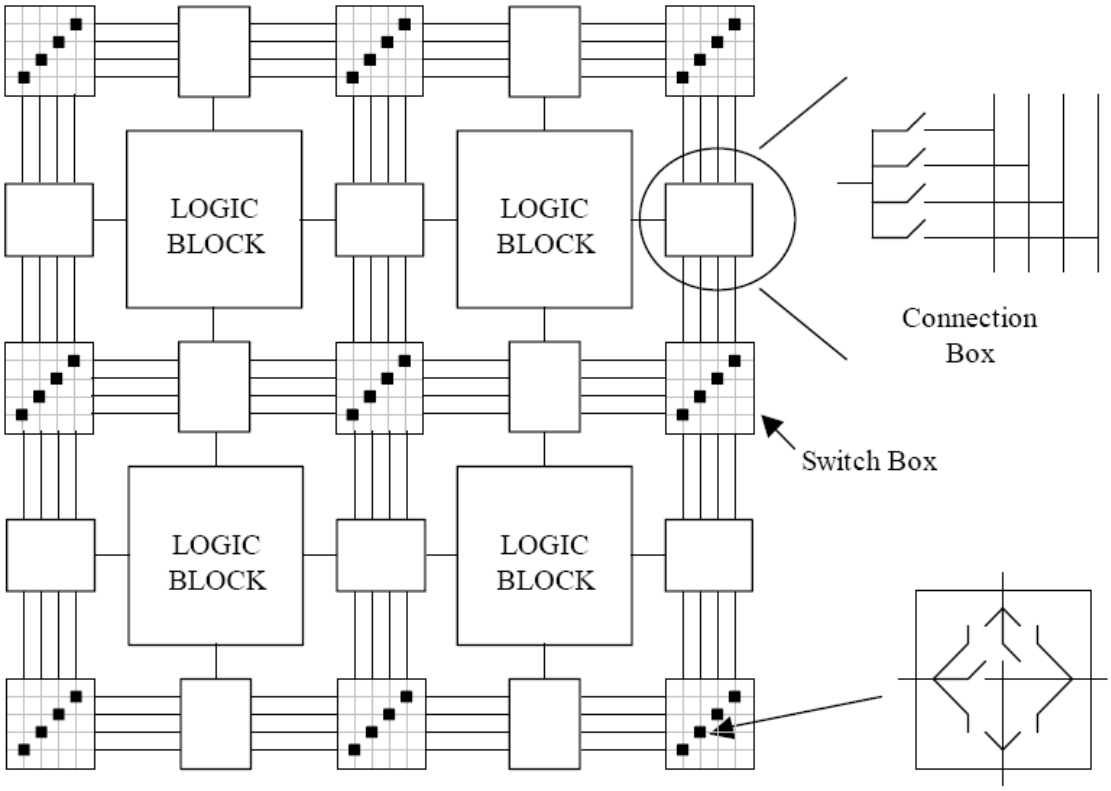


**Fig. 3-16 Estructura de bloques de interconexión y bloques lógicos (extractado de [48])**

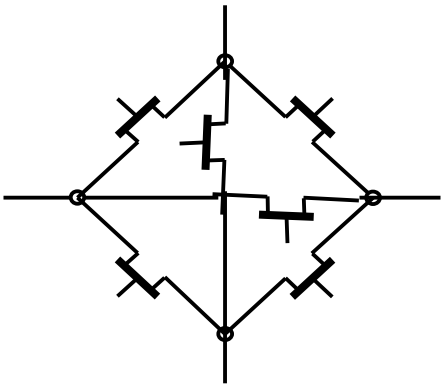

**Fig. 3-17 Punto de interconexión formado por 6 transistores de paso (adaptado de [53])**

Seis transistores de paso conforman el punto de interconexión, dichos transistores son manejados por las celdas de memoria de configuración.

Existen diversas arquitecturas de interconexión, siendo usual contar con caminos de diferente longitud

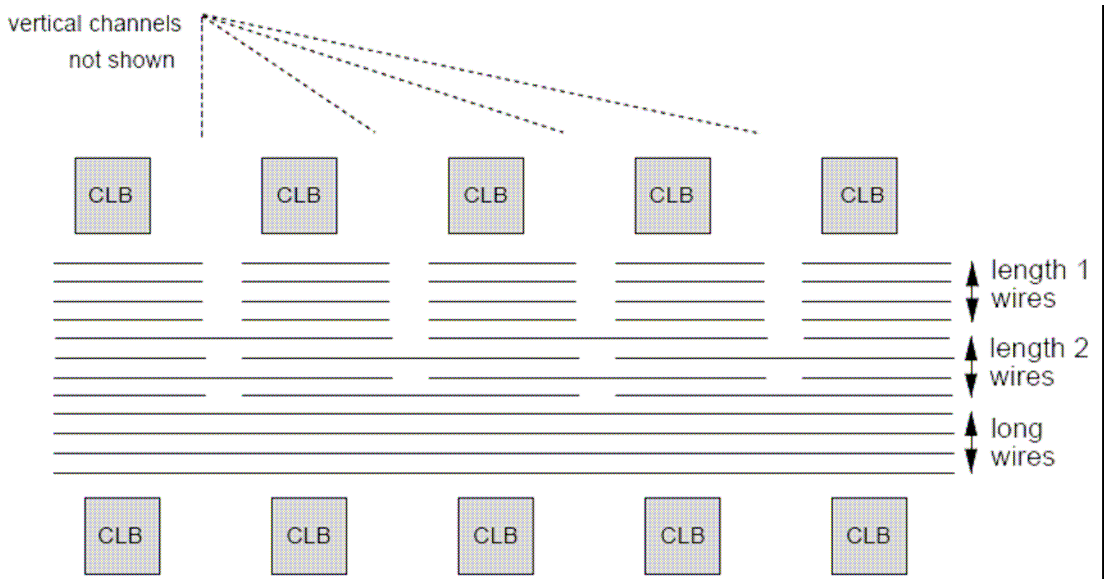


**Fig. 3-18 Segmentos de interconexión (Xilinx XC4000) [25]**

Usualmente hay tres tipos diferentes de interconexiones: Interconexiones directas entre CLBs, interconexiones de propósito general que atraviesan el chip en direcciones horizontales y verticales, y líneas largas que son reservadas para distribuir señales críticas, típicamente señales de reloj.

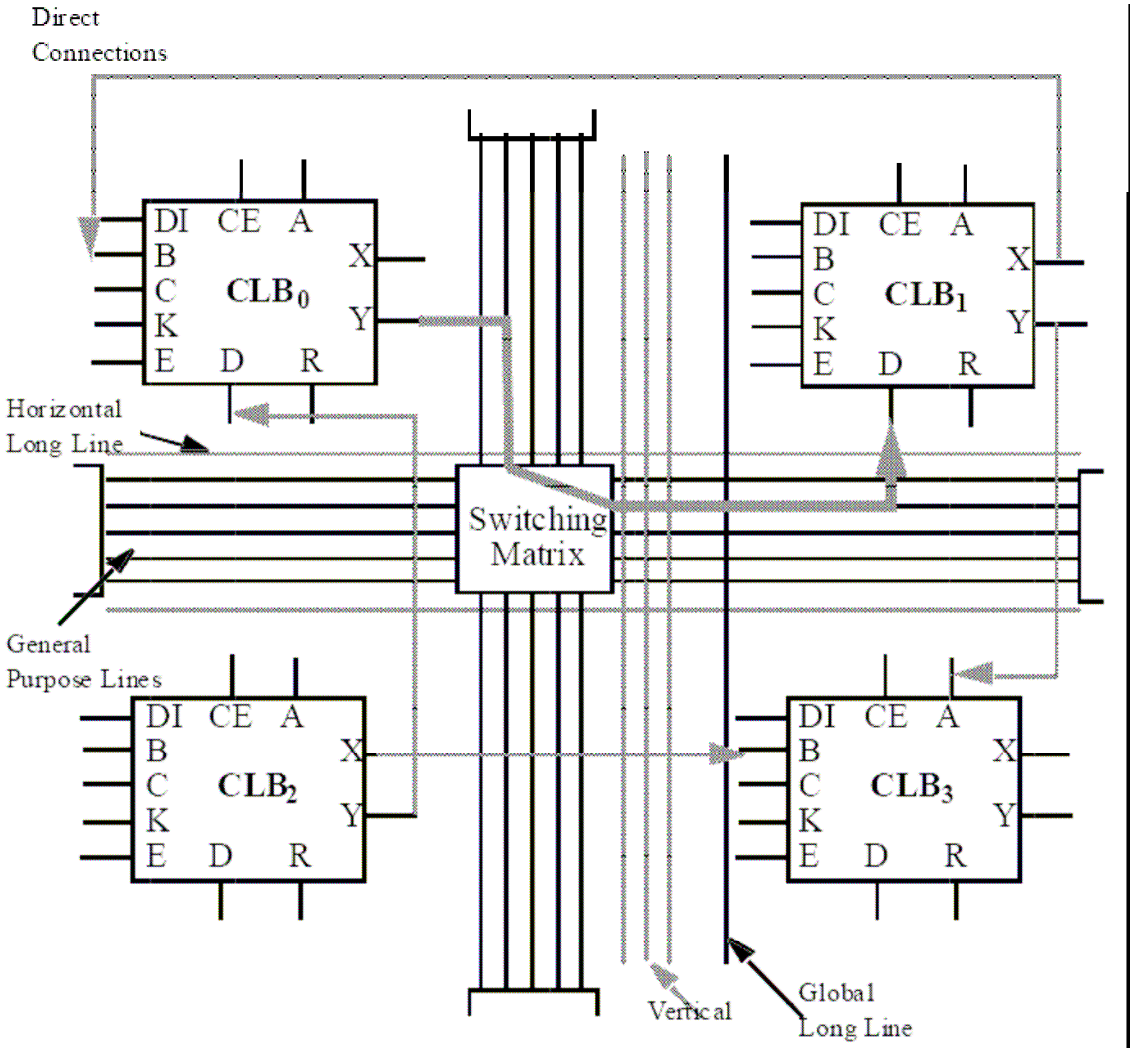


**Fig. 3-19 Detalle de los diferentes tipos de interconexiones (extractado de [30])**

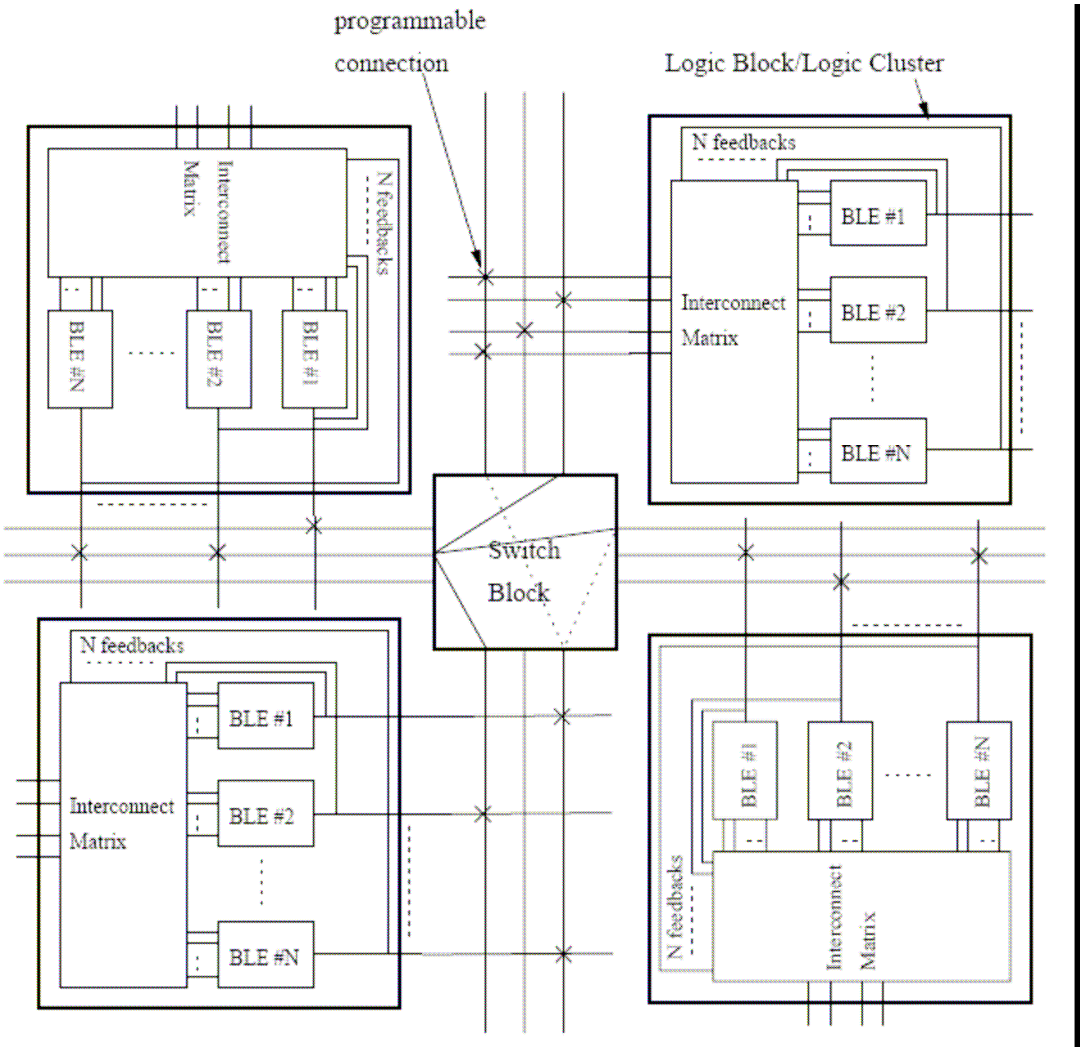


**Fig. 3-20 Detalle de las interconexiones de una FPGA (extractado de [78])**

### Nuevas incorporaciones a la arquitectura básica de FPGAs

El desarrollo de las FPGAs ha sido constante desde sus inicios hasta hoy; los fabricantes han incorporado innovaciones que manteniendo la generalidad de estos componentes, los hacen adecuados para un rango cada vez más amplio de aplicaciones. A continuación veremos una descripción de las diversas características que incorporan las FPGAs modernas.

- Memoria

Quizá la primer incorporación a la estructura básica de las FPGAs es la inclusión de bloques de memoria RAM. Estos bloques están disponibles para el diseñador, y son configurables en el tamaño y el ancho de palabra. Pueden ser utilizados para almacenamiento de datos o para implementar funciones combinatorias complejas, aunque son más lentos que las celdas estándar. De acuerdo a la configuración pueden ser utilizados como RAM dual port, FIFOs, o RAM sincrónica.

- Bloques aritméticos

  El incremento en aplicaciones de procesamiento digital de señales llevó a la incorporación en hardware de bloques aritméticos. Estos bloques implementan funciones multiplicador-acumulador (MAC) con enteros.

- Microprocesadores

  Cuando se utilizan FPGAs para realizar funciones de cálculo es usual que trabajen en conjunto con un microprocesador compartiendo las tareas. Existen varios ejemplos de FPGAs que incorporan microprocesadores internamente. Estos microprocesadores pueden estar en hardware como hardcores, es decir que dentro del chip hay un bloque de silicio específico para el microprocesador; o bien como parte de los circuitos programados en la FPGA como softcores o IP cores. Ejemplos de hardcores pueden verse en FPGAs de Xilinx Virtex II Pro [117] y Virtex 4 [114-116], que incorporan hasta cuatro cores de PowerPC 405 [113], o en la familia Excalibur de Altera [10] (aunque Altera ya no está promoviendo el uso de estos dispositivos para nuevos diseños).

  Hay una enorme cantidad de ejemplos de softcores, pero para seguir en la línea de los fabricantes de chips mencionaremos los Pico y MicroBlaze de Xilinx y el NIOS II de Altera [13].

- Menejo de relojes

  Al aumentar el tamaño y la complejidad de los chips se hace necesario proveer una buena distribución interna de las líneas de reloj globales que no introduzcan diferencias de retardo entre distintas partes del dispositivo. Es así que se incluyen líneas especiales rápidas para la distribución de los relojes,

bloques específicos de control de señales de reloj y PLLs para generar internamente diferentes frecuencias a partir de una señal externa.

- Entrada-salidas específicas

  Para que las FPGAs puedan manejar directamente líneas de alta velocidad sin necesidad de transceivers externos se incorporan a los bloques de IO transceivers programables que cumplen con varios de los estándares usados, ya sea en single-ended o diferenciales, los mismos llegan a manejar señales de varios giga bits por segundo.

- Conversores serie-paralelo de alta velocidad

  Asociado con el ítem anterior, para poder trabajar con señales de alta frecuencia, es necesario incorporar serializadores o conversores serie paralelo de alta velocidad.

- Facilidades de test on-chip

  Existen diversas estrategias que facilitan la prueba y el debugging de los diseños, estas van desde la posibilidad de la lectura o escritura de los registros y de las memorias internas vía JTAG, hasta la incorporación de analizadores lógicos integrados en el chip.

**Ejemplos de FPGAs comerciales**

**Como ejemplos finales de FPGAs modernas veremos esquemas de las estructuras internas de Stratix II de Altera [73] y las series Virtex II y Virtex 4 de Xilinx.**

## Altera Stratix II

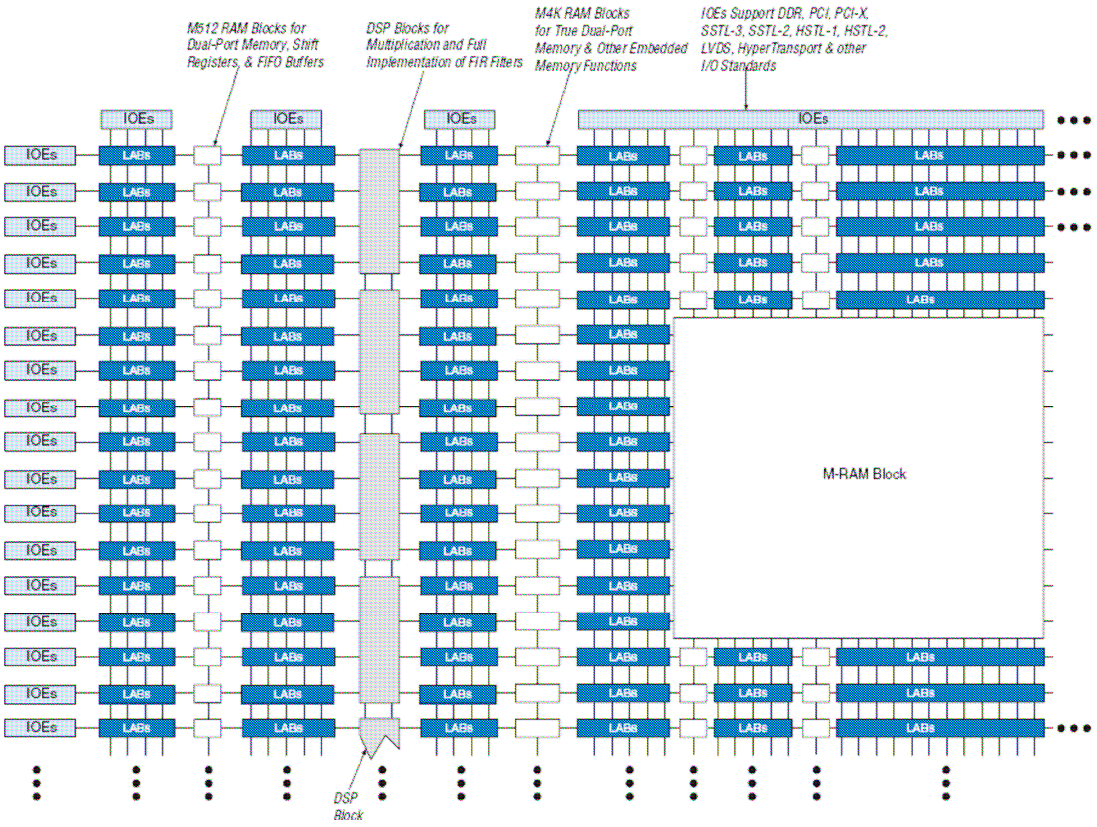


**Fig. 3-21 Diagrama de bloques del Stratix II (extraído de [14])**

En el diagrama de bloques puede verse la estructura del Stratix II que incluye diferentes niveles de memorias intercalados con los LABs, y bloques DSP.

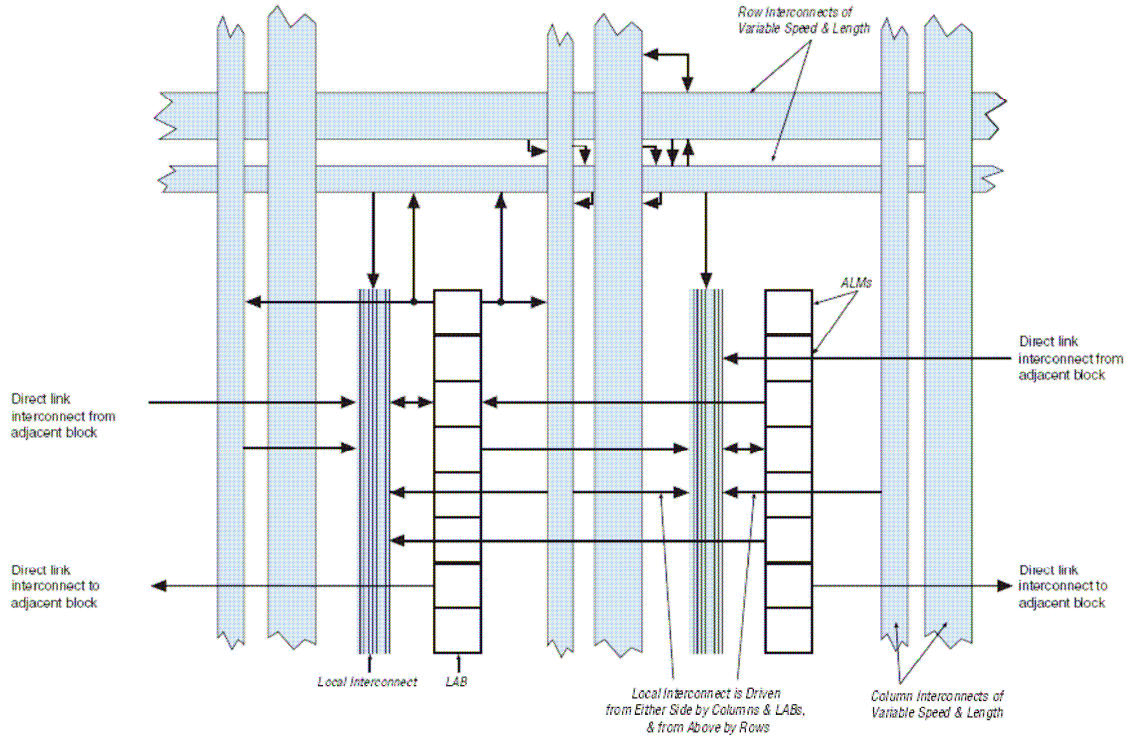


**Fig. 3-22 Estructura de un LAB del Stratix II (extraído de [14])**

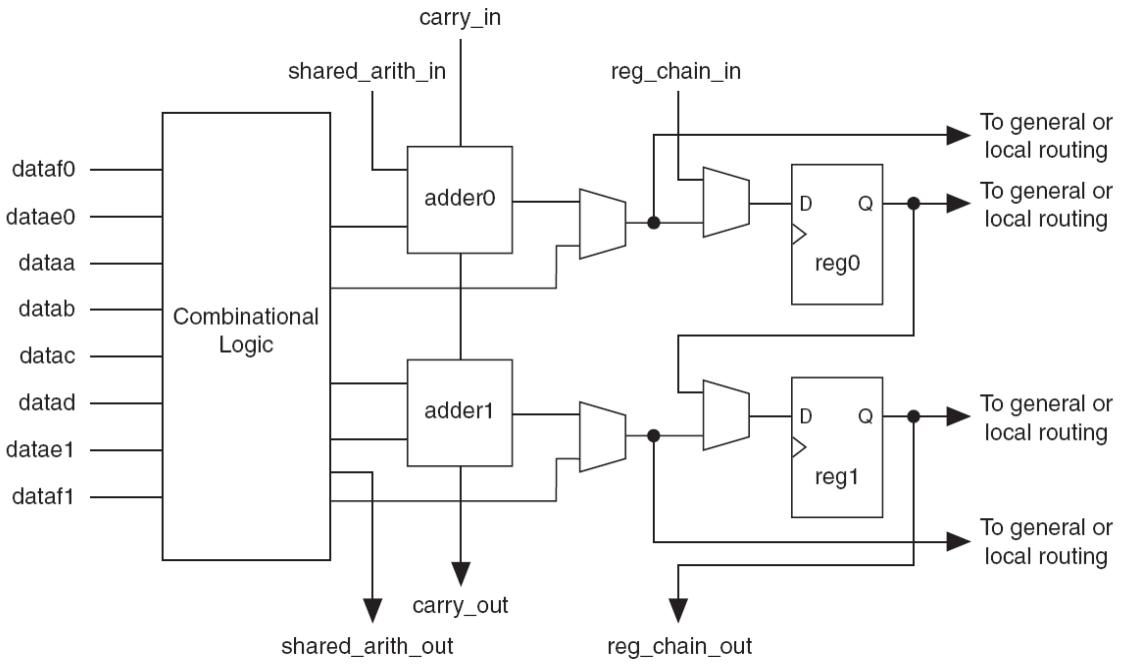


**Fig. 3-23 Diagrama de un Adaptive Logic Module (ALM) del Stratix II (extraído de [14])**

Lo más novedoso del Stratix II es que su arquitectura está basada en los llamados Adaptive Logic Modules (ALM), los cuales están formados por una LUTs adaptable, dos sumadores, segmentos de carry-chain, dos flip-flops, y lógica adicional. La principal característica de estos elementos es que la LUT puede adquirir diversas configuraciones que van desde una única tabla de 7 entradas a dos LUTs de cuatro entradas trabajando en forma independiente [26], como puede verse en el esquema de la Fig. 3-24

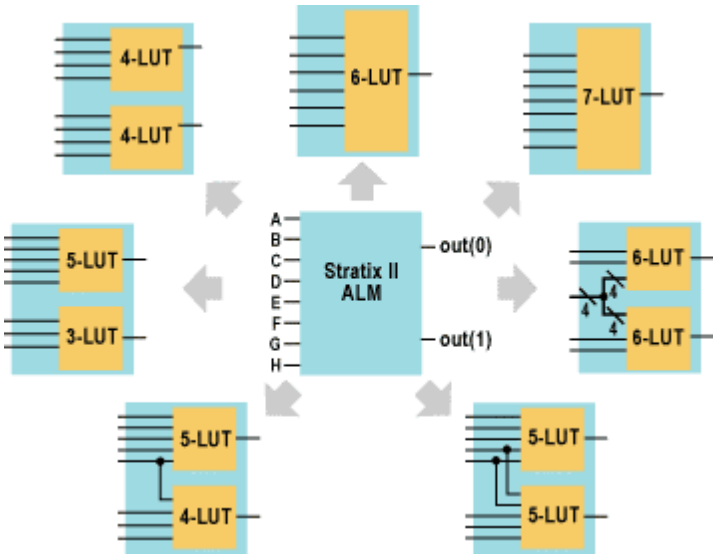


**Fig. 3-24 Diferentes configuraciones de un ALM (extraído de [15])**

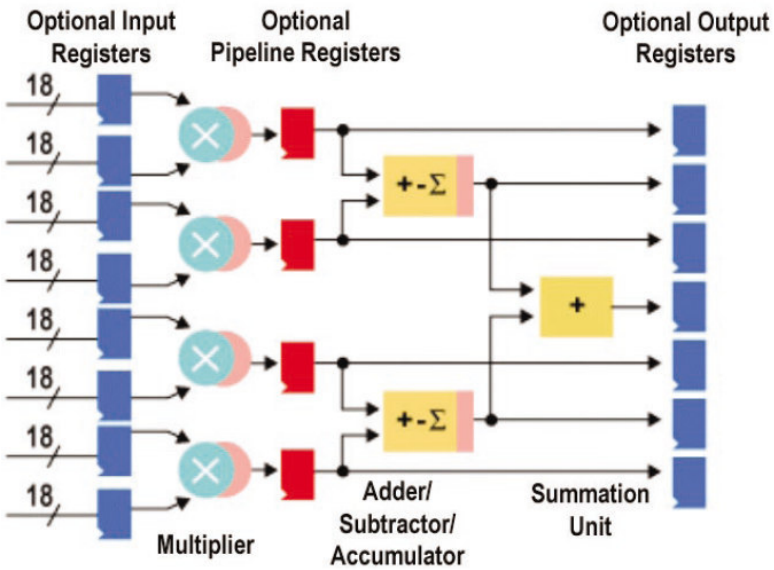


**Fig. 3-25 Bloque lógico de DSP (extraído de [15])**

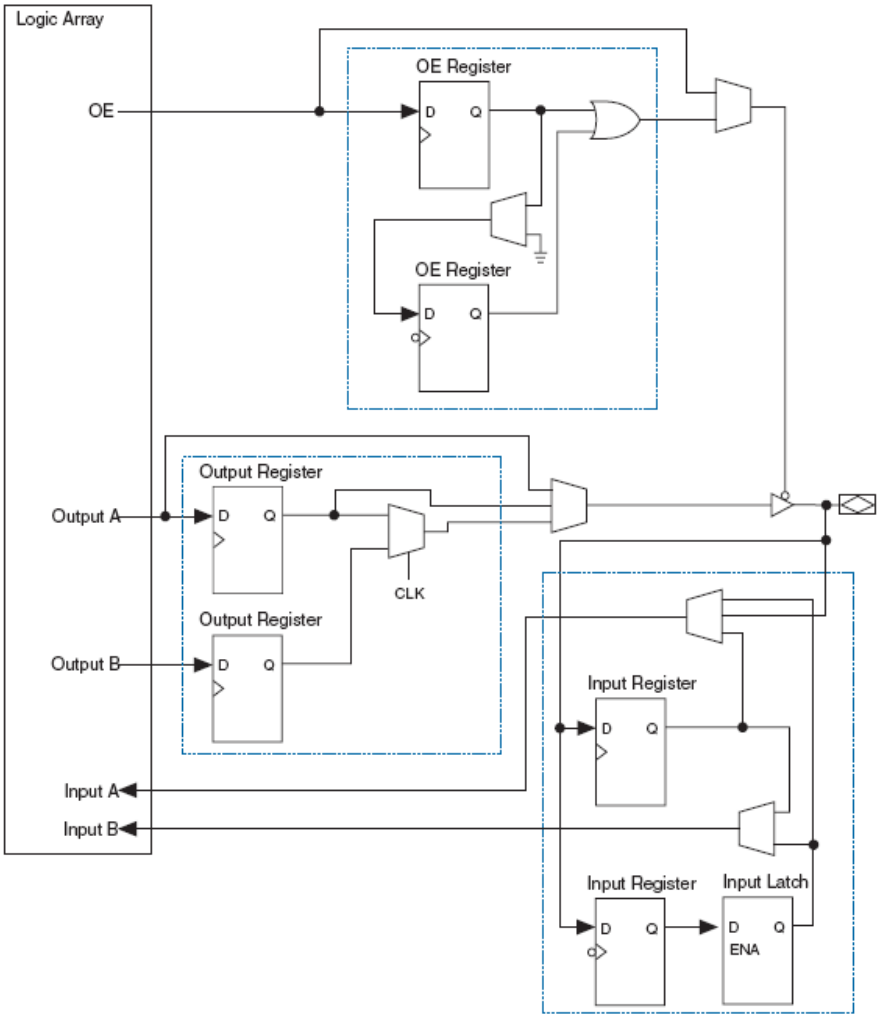


**Fig. 3-26 Elemento de IO del Stratix II (extraído de [14])**

## Xilinx Virtex II y Virtex 4

A continuación se muestran los diagramas de bloques de dos integrados de la familia Virtex, dado que estos chips tienen arquitecturas muy similares serán analizados en conjunto. Comenzamos viendo la figura Fig. 3-27 que muy claramente muestra la jerarquía de la estructura del Virtex II.

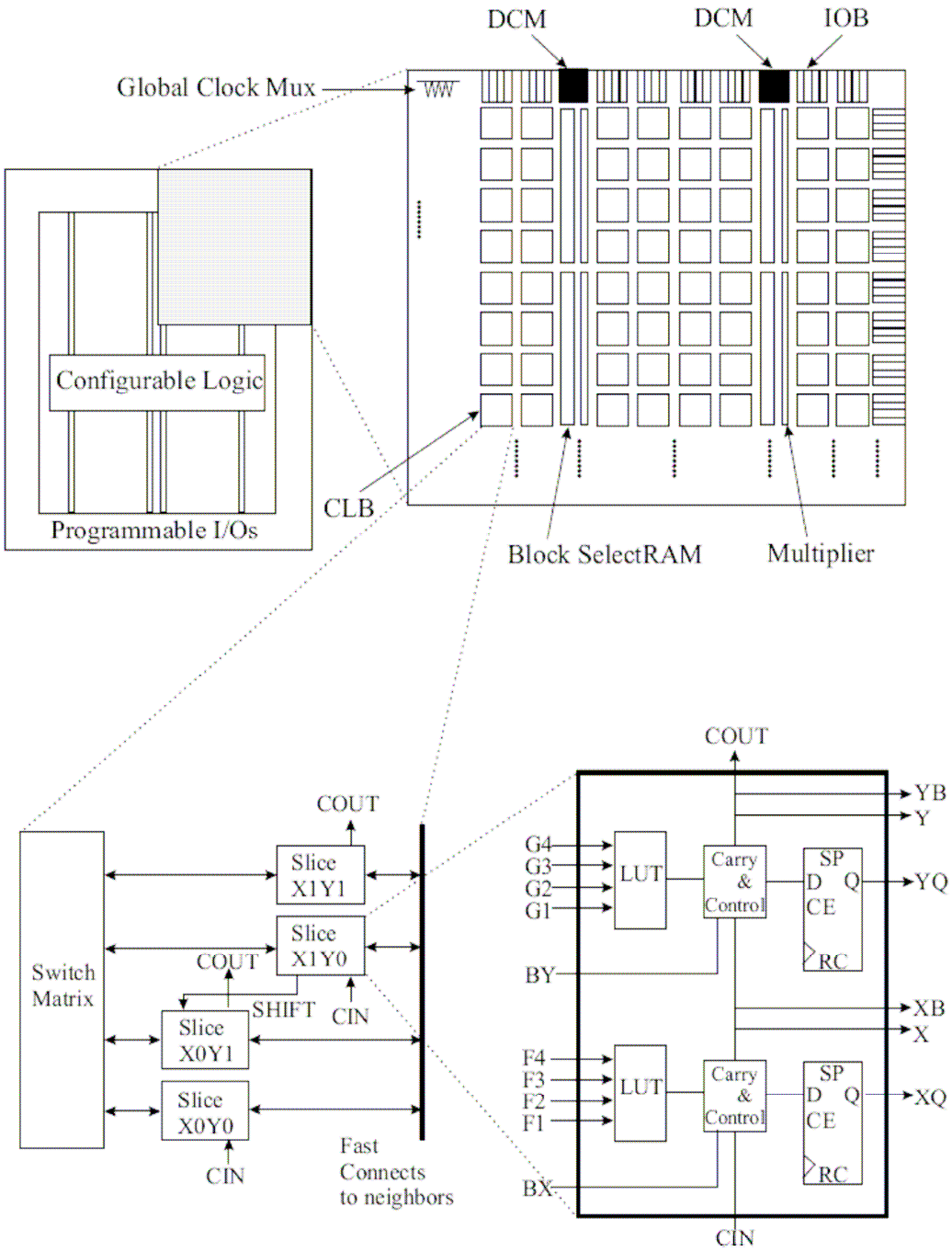


**Fig. 3-27 Arquitectura jerárquica de una Virtex II (extraído de [106])**

Cada slice del Virtex II posee dos LUTs, lógica de control de acarreo y dos Flip-Flops. En la figura siguiente puede verse que las LUTs pueden configurarse para representar funciones lógicas, cómo memorias o cómo shift-registers.

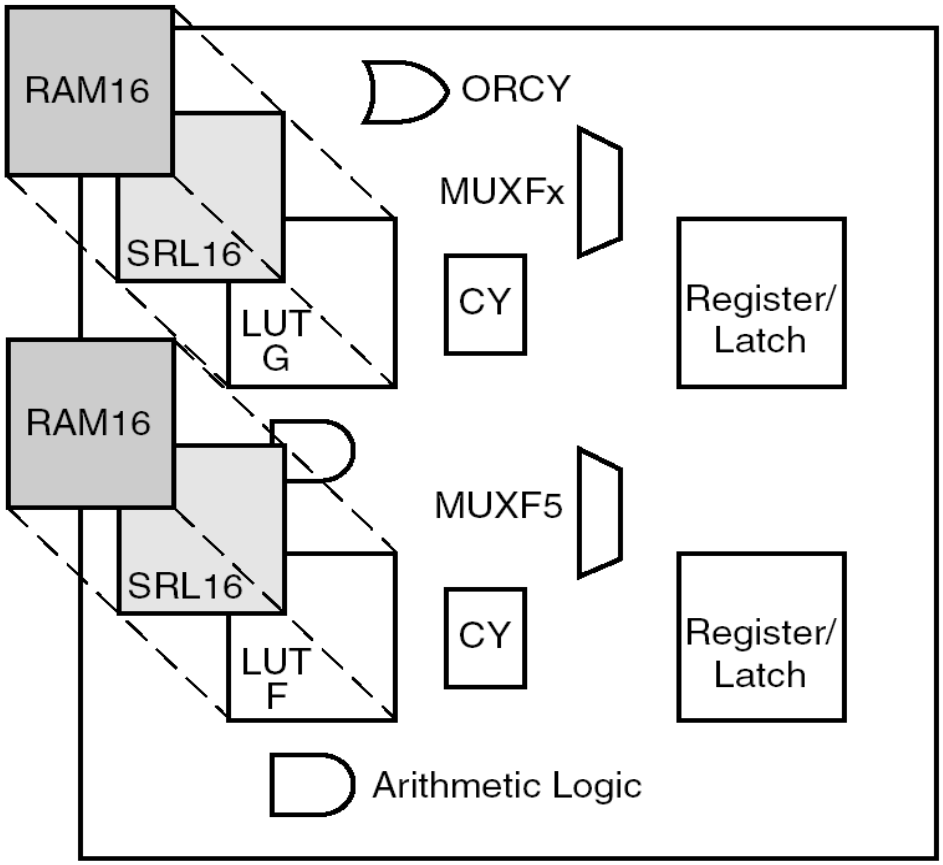


**Fig. 3-28 Configuración de un Slice de un Virtex II (extraído de [118])**

Esta es una diferencia entre el Virtex II y el Virtex 4; en el Virtex 4 la arquitectura pasa a tener dos tipos de Slices, los que pueden alternar entre función combinatoria o memoria; y aquellos que sólo pueden ser utilizados como funciones lógicas.

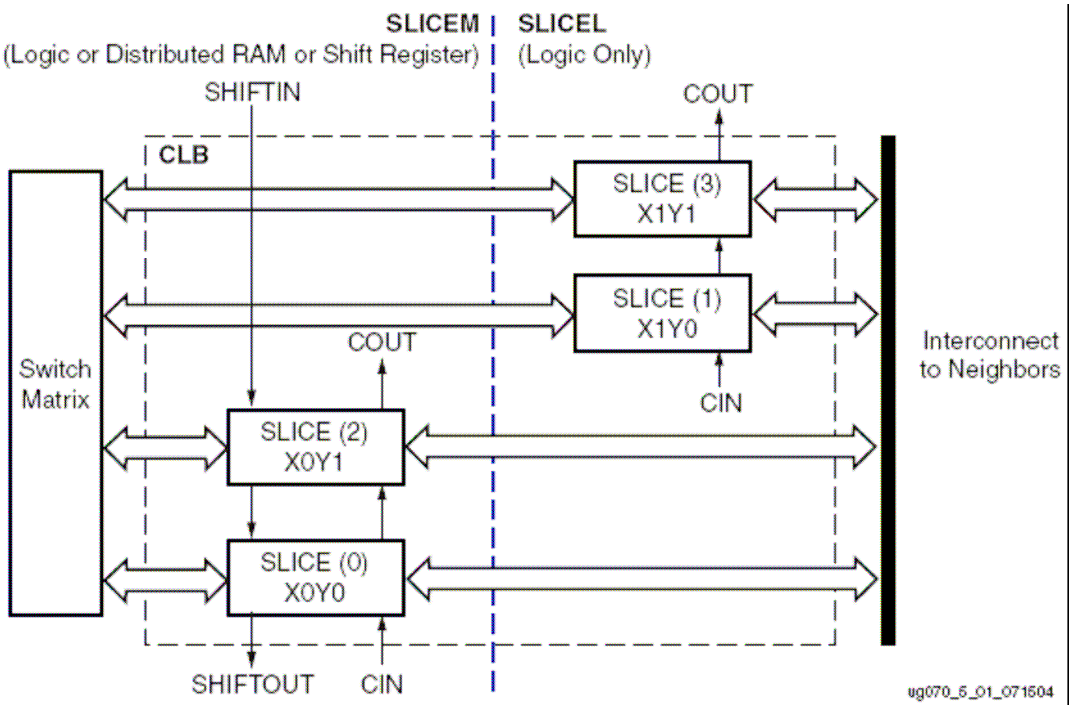


**Fig. 3-29 Diferentes tipos de slices en el Virtex 4 (extraído de [116])**

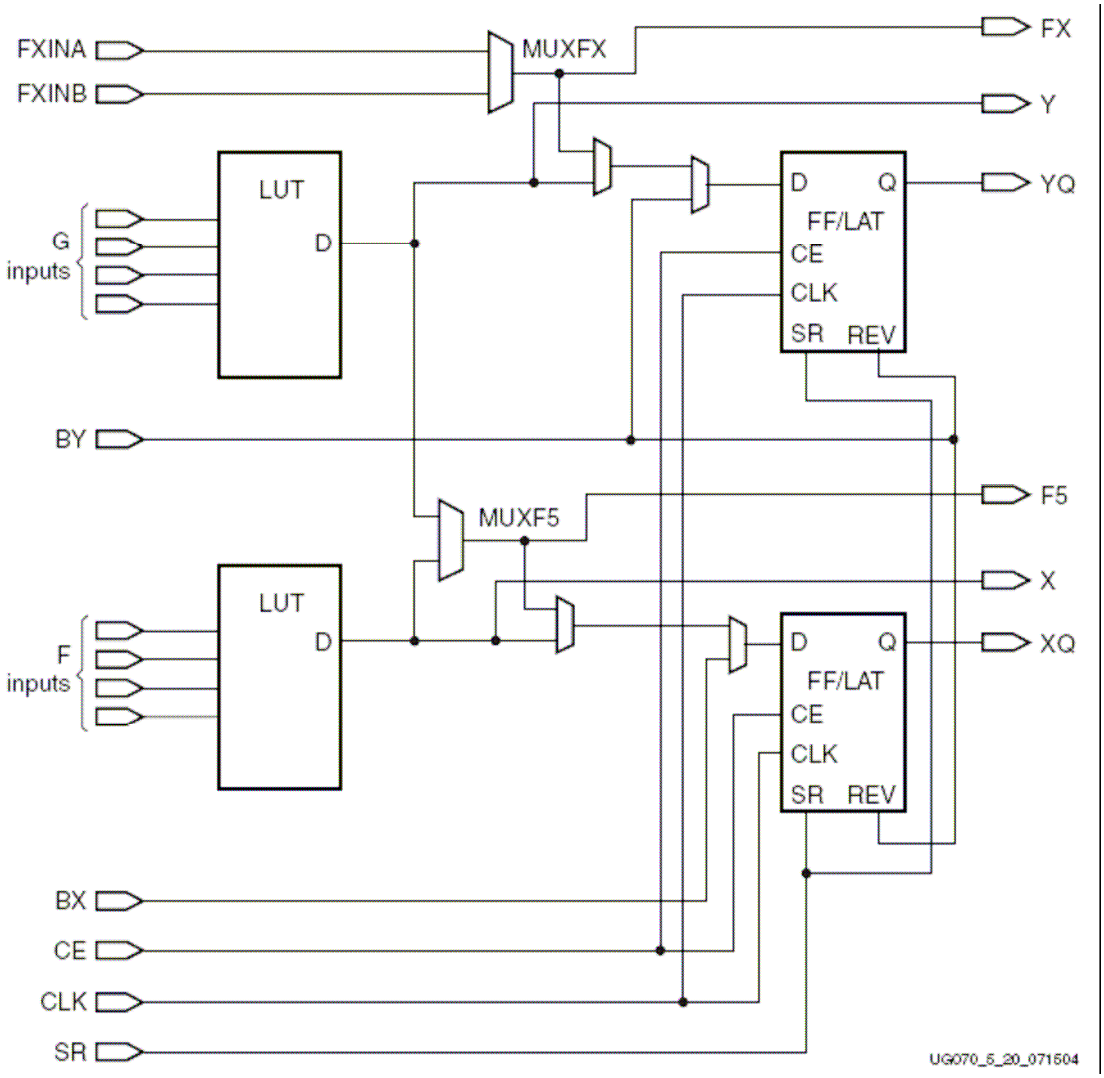


**Fig. 3-30 Virtex 4 vista simplificada de un slice (extraído de [116])**

### 3.2.7 ASICS Estructurados (Structured ASICS)

Tanto Altera [11] como Xilinx [121] ofrecen una nueva alternativa a sus FPGAs, y es la de los Structured ASICS. Estos dispositivos no son programables en campo, por lo tanto estrictamente no entrarían en la reseña que estamos realizando, pero por sus características haremos una brevísima descripción de los mismos.

La idea que proponen los fabricantes es contar con una línea de ASICS que se asocia a una familia de FPGAs, para producción en grandes volúmenes. Una vez que el desarrollo está realizado diseñado, prototipado y verificado en una FPGA, se puede

realizar una rápida migración a dispositivos no programables, pero con arquitecturas similares a los programables.

Si bien se pierden todas las características asociadas a la programabilidad, se logra bajar mucho los costos en volúmenes de producción elevados [120].

### 3.2.8 PIDs (Programmable Interconnect Devices)

Para completar el espectro de los dispositivos lógicos programables haremos una breve descripción de dispositivos programables de interconexión. Es decir dispositivos que no poseen capacidad lógica y sólo permiten realizar conexiones programables entre sus pines.

Estos dispositivos conocidos como Field-Programmable Interconnect Chips (FPIC), o también Field-Programmable Interconnect Devices (FPID), pueden verse como una gran matriz de interconexiones programable, o dispositivos de cableado programable; que generalmente almacenan la información de configuración en una SRAM (aunque también existieron versiones de antifuse).

Los FPIDs o FPICs [79] usualmente poseen un gran número de pines (del orden de 1000) y pueden ser reconfigurados dinámicamente.

Hoy en día estos dispositivos casi no se comercializan como componentes, de los dos fabricantes principales I-Cube y Aptix, el primero ya no existe, y el segundo no comercializa los chips sino en sus propios productos. Aptix se dedica a test y verificación de diseños y ha desarrollado placas de interconexiones reconfigurables basadas en FPICs [17] (FPIC es marca registrada de Aptix Corp, San Jose, CA.), permite interconectar cualquier punto de la placa con cualquier otro sin la necesidad de impresos o cableados y con la posibilidad de modificaciones en cualquier etapa del diseño.

Lattice posee una familia de dispositivos de este tipo orientados a bus switching ispGDX2 [71], son dispositivos un poco más complejos ya que agregan varias características que los hacen adecuados para el manejo de señales de alta velocidad. Incluyen varias interfaces de entrada-salida de alta velocidad, single-ended y diferenciales, FIFOs en las entradas, serializadores-deserializadores de alta velocidad, y bloques de manejo de relojes con PLLs. En la siguiente figura puede verse un diagrama de bloques de la familia ispGDX2.

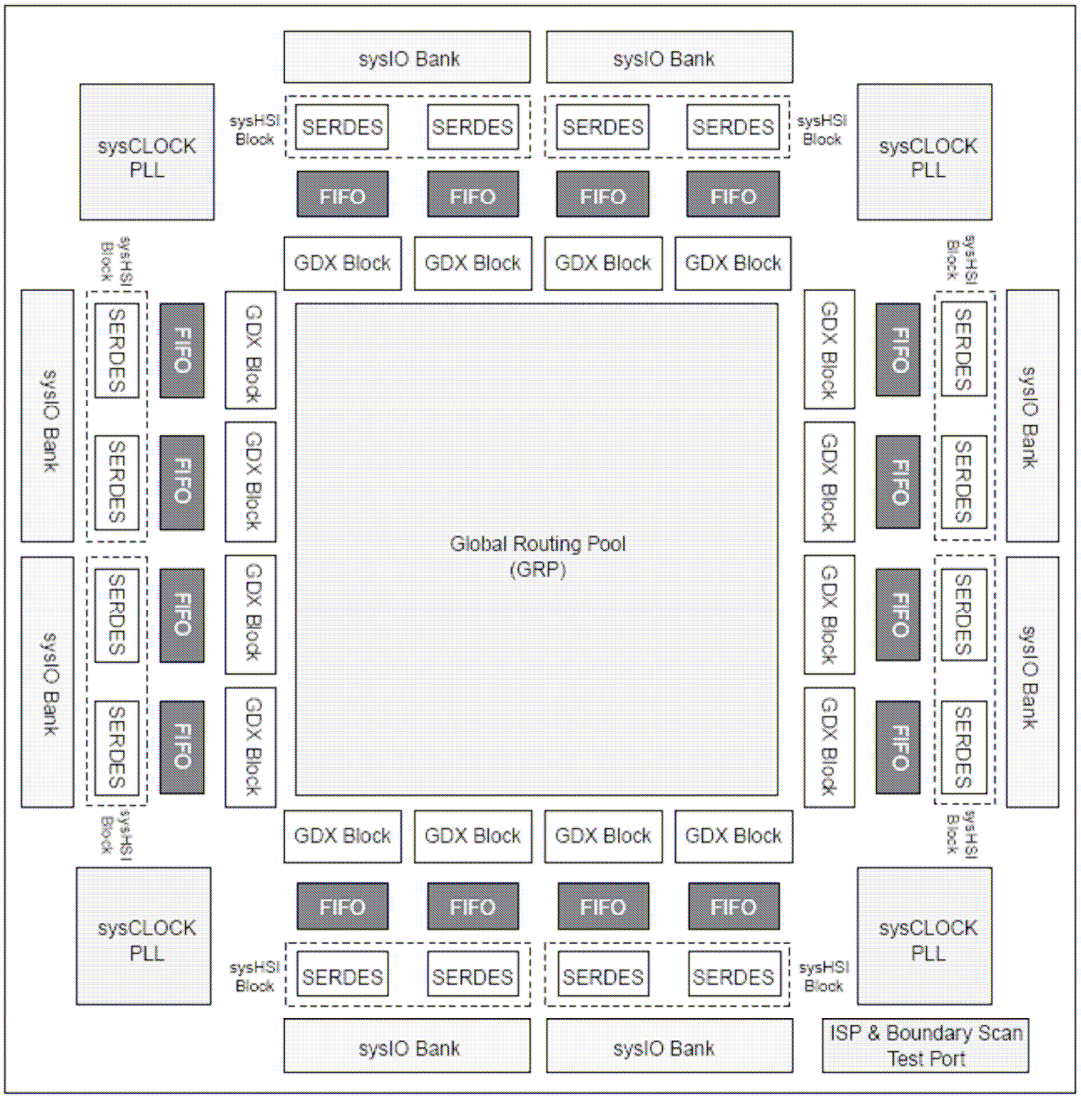


**Fig. 3-31 Diagrama de bloques del ispGDX2 de Lattice (extraído de [71])**

## 3.3 Herramientas de software (EDA, CAE, CAD)

Primero un poco de terminología, las herramientas CAD (Computer-Aided Design) se definen como aquellas que realizan funciones de *place and route*, y *layout*; usualmente el término CAE (Computer-Aided Engineering) se aplica mayormente a simulación, síntesis y análisis de tiempos; aunque es usual utilizar ambos términos en forma intercambiable. El término EDA (Electronic Design Automation) engloba las dos definiciones anteriores [92]

Para diseñar con dispositivos lógicos programables es absolutamente necesario el uso de un conjunto de herramientas software de buena calidad, confiables y de fácil utilización. Estas herramientas son esenciales, tanto a nivel industrial como en la formación del ingeniero, en la medida que la integración progresa y ya no se puede acceder a los elementos básicos del hardware.

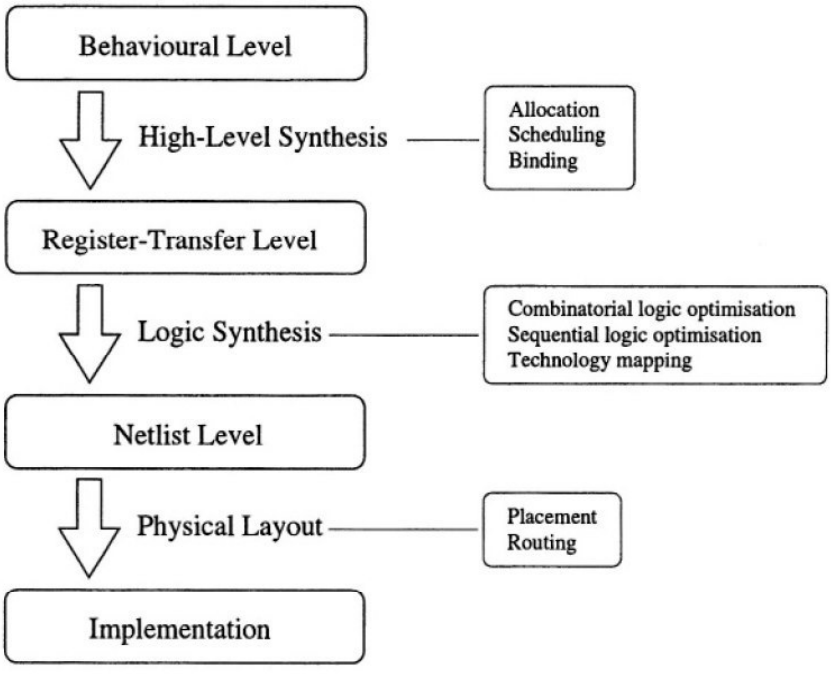


**Fig. 3-32 Niveles de especificación de un diseño y los diferentes procesos involucrados (extractado de [101])**

Las etapas de un proceso típico de diseño, sin las realimentaciones de simulación y verificación, pueden verse en la Fig. 3-32

La síntesis de alto nivel es el proceso de compilar una descripción comportamental dada en un lenguaje de alto nivel en una descripción estructural dada a nivel de transferencia de registros (RTL *register-transfer level*). Este proceso generalmente se hace en forma manual, pero es incipiente el desarrollo de herramientas y lenguajes de alto nivel, que veremos con más detalle en la sección 3.3.3.

La síntesis lógica es el proceso de convertir una descripción RTL en un *netlist* optimizado. El *netlist* es una descripción del circuito a nivel de compuertas y registros y las interconexiones entre esos elementos. Las tareas realizadas en el proceso de síntesis incluyen la optimización de la lógica combinatoria y la optimización de la lógica secuencial. Dichas optimizaciones pueden incluir diferentes requerimientos cómo el área del chip utilizada, la velocidad o el consumo.

El mapeo tecnológico se refiere a la mapeo de un determinado *netlist* genérico en las primitivas o bloques disponibles en una determinada tecnología. En el caso de FPGAs, por ejemplo, se deberán mapear las funciones lógicas en las LUTs disponibles en la familia utilizada. Debe hacerse algo similar para los registros, pasando de un registro genérico a los tipos de registro disponibles en la tecnología.

Las herramientas utilizadas deben contar también con la capacidad de simular y verificar el diseño en diferentes niveles y etapas del mismo. Si se trabaja con lenguajes de especificación hardware, por ejemplo VHDL, la forma usual de simular es construir un *test bench* o banco de pruebas. El *test bench* se especifica también en VHDL e incluye como un componente el diseño en cuestión, generándole entradas y chequeando sus salidas. Un simulador VHDL es capaz de ejecutar *test-benches* disponiendo, normalmente, de un conjunto de utilidades que facilitan la depuración de los modelos y la revisión de resultados.

Las primeras etapas pueden hacerse con herramientas genéricas independientes de los fabricantes de dispositivos programables, pero las etapas de *placement* y *routing* requieren la utilización de herramientas propietarias.

### 3.3.1 Evolución histórica

La evolución de las herramientas de EDA se dio en conjunto con la evolución de los dispositivos lógicos programables y de las computadoras.

Las primeras formas de trabajar con PLDs implicaban la traducción de un diseño a un formato de tabla de verdad llamado H&L [94]. Posteriormente aparece el PALASM, un programa escrito en FORTRAN que permitía convertir un diseño descripto en ecuaciones boleanas, en archivos de programación de dispositivos PAL de Monolithics Memories (MMI).

En esta época no solo eran importantes las herramientas software, sino también los equipos de programación de dispositivos, que recién se estaban generalizando. Un salto importante fue la introducción del estándar 3 de JEDEC, propuesto en 1980. Este estándar define un formato de archivo común independiente del fabricante del dispositivo y del fabricante del programador.

En los ochentas aparecen dos lenguajes que permiten un mayor nivel de abstracción en la expresión de los diseños, y son independientes de los fabricantes de chips, estos son el ABEL (Advanced Boolean Expression Language) y el CUPL (Common Universal tool for Programmable Logic). Estas herramientas además de definir un lenguaje para la entrada de diseños, disponían de paquetes de minimización booleana, síntesis de máquinas de estados y simulación.

A mediados de los ochentas aparecen los programas de diseño de Altera A+PLUS y un conjunto de herramientas sumistradas por Xilinx para sus nuevos dispositivos. Como los dispositivos de Xilinx tenían una arquitectura diferente sus herramientas incluían place & route automático para un eficiente uso de los recursos disponibles.

La siguiente generación de herramientas software incluyó la posibilidad de especificar diseños mediante entrada esquemática, haciendo más fácil la migración de diseños realizados previamente con circuitos lógicos de las familias TTL y CMOS.

### 3.3.2 Los lenguajes de especificación hardware (HDL Hardware Description Language)

Si bien ABEL y CUPL ofrecían una herramienta de diseño independiente de los fabricantes de circuitos, estos lenguajes estaban muy atados a la implementación del diseño en un PLD. A principios de los ochentas aparecen dos lenguajes de alto nivel pensados para especificar el comportamiento del hardware digital, estos son VHDL y Verilog. La importancia de estos lenguajes es su estandarización por el IEEE, VHDL tiene su primer estándar en 1987 [62], y luego es modificado en 1993, 2000 y 2002 [61-63] y Verilog varios años después, recién pasa a ser estándar en 1995 y es revisado en 2001 [64, 65].

VHDL es un lenguaje diseñado para describir sistemas electrónicos digitales cuyo origen está en un programa del gobierno de Estados Unidos Very High Speed Integrated Circuits (VHSIC) iniciado en 1980. A lo largo de ese proyecto resultó clara la necesidad de contar con un lenguaje estándar que pudiera describir la estructura y la funcionalidad de un circuito digital y el resultado fue el desarrollo del VHSIC Hardware Description Language (VHDL) [18].

Verilog HDL tuvo un origen diferente, arrancó en sus comienzos como un lenguaje propietario desarrollado en 1983 por Gateway Design Automation, compañía que posteriormente fue adquirida por Cadence Design System. A comienzos de los ’90 Cadence abre la especificación del lenguaje y promueve su estandarización por el IEEE que se logra en 1995 [104].

Ambos lenguajes, si bien diferentes, poseen características similares. Permiten describir la estructura de un diseño y su partición en bloques jerárquicos, así como la

interconexión de dichos bloques. Además admiten describir la funcionalidad de esos bloques independientemente de su implementación y la posterior simulación de los mismos y del sistema completo antes de su implementación. Los HDLs, a diferencia de los lenguajes de programación usuales, están especialmente diseñados para permitirle a los diseñadores modelar la concurrencia de procesos inherente a los elementos hardware.

El salto cualitativo en las herramientas EDA fue a fines de los ochentas cuando aparecieron los primeros programas que incluían síntesis lógica automática. Hasta ese momento los HDLs eran utilizados para especificar, simular y verificar los diseños, pero éstos debían ser manualmente traducidos a su implementación con compuertas. La síntesis lógica cambió radicalmente la metodología de diseño, con una especificación a nivel de transferencia de registros (RTL, Register Transfer Level), hecha en algún HDL, la herramienta extrae en forma automática el detalle de compuertas, flip-flops e interconexiones. Esto permite describir circuitos digitales complejos exclusivamente en alto nivel, dejando los detalles de implementación a herramientas automatizadas. Los lenguajes pasan entonces a tener una mayor importancia en el diseño digital.

Cabe mencionar que la codificación RTL realizada en un lenguaje HDL requiere utilizar un subconjunto de dicho lenguaje, ya que como los lenguajes HDL no fueron específicamente diseñados para síntesis, sino que están pensados para especificar y simular hardware, existen en ellos varias sentencias o expresiones que no son sintetizables en hardware.

Las etapas de alto nivel de un diseño, síntesis lógica y mapeo, pueden ser realizadas con herramientas independientes del fabricante del circuito programable, pero las etapas de bajo nivel, *placement* y *routing* deben utilizar las herramientas de los fabricantes de FPGAs, ya que la información de programación de las mismas es propietaria.

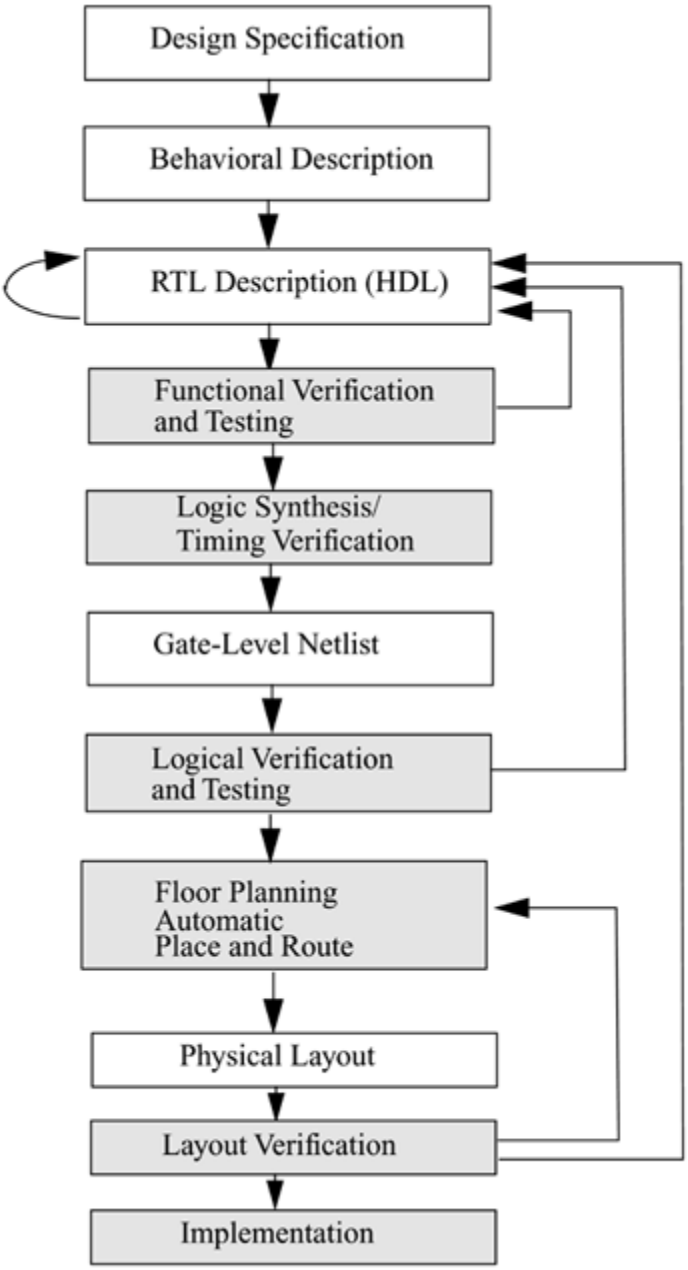


**Fig. 3-33 Proceso típico de diseño con un HDL (extractado de [92])**

La Fig. 3-33 muestra el proceso típico de diseño con sus diferentes etapas [92]. Los bloques sombreados son las etapas de procesamiento, los bloques sin sombrear representan el nivel del diseño.

Cuando el diseño va a ser implementado en una FPGA la herramienta debe realizar el mapeo en las celdas de la FPGA, esto significa que las herramientas deben convertir el diseño o las ecuaciones booleanas en el tipo de LUTs utilizadas en las FPGAs. Hay

varias investigaciones que estudian los algoritmos necesarios para realizar este mapeo en LUTs y sus optimizaciones [31, 45].

Para el caso de FPGAs modernas que incluyen bloques de RAM, bloques aritméticos u otros bloques dedicados, es necesario que las herramientas EDA los sepan utilizar adecuadamente.

Posteriormente resta realizar el *placement* y *routing* dentro de la FPGA. Esto significa elegir en qué celdas se ubica el diseño y elegir los caminos de interconexiones entre esas celdas. Investigaciones sobre algoritmos de *placement* y *routing* para FPGAs pueden verse en [23, 66, 80].

### 3.3.3 Herramientas de alto nivel

A medida que los diseños aumentan su complejidad aparece la necesidad de utilizar herramientas de mayor grado de abstracción para poder desarrollar aplicaciones cumpliendo con las exigencias de productividad. Es claro que cuando se aumenta el nivel de abstracción se gana en tiempo de diseño, pero se pierde en la optimización del mismo.

Actualmente todavía la mayoría de los diseños se realizan utilizando lenguajes de descripción hardware, principalmente VHDL y Verilog, y existen estrategias específicas para manejar diseños de gran tamaño. La principal es partir el diseño en bloques más pequeños y diseñar adecuadamente la jerarquía de estos bloques así como las interfaces de los mismos. Muchos de estos bloques pueden estar pre-hechos como IP cores, o diseñarse una vez y reutilizarse [70].

Las cosas se complican aún más porque hay que describir el comportamiento esperado del sistema para poder verificar los bloques diseñados. Si bien los HDLs permiten esto, no son los lenguajes más cómodos para ello. Si además el sistema incluye un microprocesador y por lo tanto software corriendo en él, entonces se plantea la idea de contar con un lenguaje único que sirva tanto para el hardware que

va estar implementado en una FPGA como para el software del microprocesador. Cuando el sistema incluye un microprocesador y bloques hardware hay toda una disciplina que estudia cómo particionar los diseños llamada hardware/software codesign, una buena introducción a este tema puede verse en [112].

Otro punto importante cuando se piensa en diseños que utilizan plataformas reconfigurables como aceleradores de cálculo, es que la mayoría de los algoritmos están especificados y probados en lenguajes de programación tales como C o Matlab y no en HDLs.

Ha habido varios esfuerzos en incorporar lenguajes de programación al diseño con FPGAs, el problema principal es que estos lenguajes están concebidos para implementaciones secuenciales y no para representar el paralelismo inherente al hardware.

Si bien hay una gran cantidad de lenguajes de programación que han sido utilizados para sintetizar hardware en forma más o menos automática, el más utilizado es el C. Esto se debe al grado de difusión que tiene el C, y pueden mencionarse varios ejemplos de traductores de C a HDL. Las dificultades que se presentan hacen que siempre estos traductores no incluyan la sintaxis completa del C sino un subconjunto del lenguaje, y que además introduzcan diversas modificaciones o ampliaciones al C para representar adecuadamente el paralelismo. Como ejemplos concretos podemos mencionar: HandelC [29], Trasmogrifier C [47], Streams-C [50], Impulse C [95].

Han habido varios esfuerzos para compilar directamente desde Matlab a hardware, como muchos algoritmos se especifican directamente en Matlab esta idea es muy atractiva, algunos resultados pueden verse en [19, 55, 56]. Para algunas aplicaciones los resultados del hardware generado automáticamente son cinco veces más lentos y ocupan cuatro veces más celdas que los diseños hechos manualmente; pero se presentan otros casos en donde las velocidades de los diseños son comparables, aunque se mantienen las diferencias de tamaño. Demás está decir que los tiempos de

diseño se reducen enormemente, aunque los autores de las publicaciones no utilizan herramientas comerciales.

SystemC es otra alternativa de descripción de alto nivel, está implementado como clases de C++, y permite la definición de hardware a varios niveles de abstracción [103].

Otro punto interesante es la utilización de lenguajes de programación para realizar la verificación de un diseño. En lugar de diseñar un test-bench en HDL es posible describirlo en un lenguaje de programación. Se han desarrollado módulos de software que permiten enlazar simuladores HDL con paquetes clásicos de simulación de sistemas, un ejemplo es el vínculo entre Matlab y Simulink con ModelSim (simulador HDL). Esto permite desarrollar el entorno de pruebas (test benches) en un lenguaje de alto nivel, y "conectar" entradas y salidas hacia bloques de hardware descritos en HDL. Otra aplicación es la integración de bloques hardware descritos como HDL en modelos a escala de sistema. [ver mathworks]

## 3.4 Comentarios finales

En la evolución de los dispositivos lógicos programables han jugado diversos factores, creemos que los dos principales han sido el aumento de la capacidad de integración en un chip y el desarrollo de buenas herramientas CAD.

Ha habido una interacción de ida y vuelta entre el campo de aplicaciones en el cual estos dispositivos son aplicables y el desarrollo de los mismos. Un dispositivo de mayor tamaño y con mayores prestaciones de velocidad aumenta la cantidad de aplicaciones que pueden ser resueltas con él; pero a su vez los fabricantes de dispositivos diseñan los mismos teniendo en cuenta los potenciales usos futuros, aunque manteniendo la generalidad y flexibilidad. Hay ejemplos de dispositivos que no han tenido el éxito comercial esperado, como es el caso de la familia XC6200 de

Xilinx que se utilizó mucho pero solo a nivel académico. Un ejemplo más reciente es la familia Excalibur de Altera que incluyó un hardcore de un microprocesador ARM922T en un dispositivo de la familia APEX 20K; que quedó maduro muy rápidamente y hoy la empresa promueve exclusivamente el uso del softcore NIOS II en sus dispositivos.

Podemos distinguir tres etapas en la evolución de los dispositivos y sus aplicaciones:

- Los primeros dispositivos fueron utilizados para sustituir diseños realizados con lógica discreta TTL y CMOS. Las ventajas fueron claras, ya que utilizar dispositivos programables permite incluir en un único integrado toda la lógica de un determinado sistema, ahorrando espacio, y permitiendo modificaciones del diseño sin tener que modificar el circuito impreso.

- Con la aparición de los dispositivos de mayor tamaño se amplió el campo de aplicaciones a diseños completos de circuitos secuenciales, máquinas de control o procesamientos de datos sencillos. El nicho de aplicaciones de esta etapa fue el prototipado rápido, los sistemas con especificaciones tempranas que van a evolucionar, pre-procesamiento de entradas, etc.

- Con los dispositivos disponibles hoy en día es posible implementar aplicaciones completas y complejas dentro de un mismo chip (Soc, SoPC), sustituyendo aplicaciones que usualmente son realizadas en microprocesadores o DSPs, y obteniendo importantes mejoras en velocidad. Es posible implementar aplicaciones de procesamiento de señales o microprocesadores completos dentro de una FPGA.

Resulta interesante ver que si bien las FPGAs modernas incorporan un gran número de características nuevas: PLLs, bloques de RAM, multiplicadores-acumuladores, entradas-salidas de alta velocidad, cores de microcontroladores; la estructura básica del chip es la misma en los últimos 20 años. Es decir celdas con LUTs y flip-flops; y que varias de las implementaciones hardware, como los microprocesadores o los bloques aritméticos, pueden realizarse perfectamente con la estructura básica original.

¿Dónde está entonces el gran cambio en la evolución de las FPGAs? Nuestra opinión es que el factor determinante en la evolución de los dispositivos lógicos programables no es cualitativo, ya que no se produjeron grandes innovaciones en cuanto a las arquitecturas internas de los chips, sino cuantitativo, ya que se aumenta drásticamente la cantidad de bloques lógicos. Pero este aumento de capacidad provoca un cambio cualitativo en el tipo de aplicaciones que se pueden realizar en los mismos.

Esta es una gran diferencia con respecto al mundo de los microprocesadores, en un microprocesador, aún pequeño, es posible implementar casi cualquier aplicación, y el factor determinante estará dado por el tiempo que esa aplicación consume. En una FPGA en cambio el factor determinante está dado por el tamaño, ya que si la aplicación no cabe es muy probable que ya no sea interesante implementarla.

# Capítulo 4

# Aplicaciones en Enseñanza: Curso Básico

En este capítulo se presenta la reformulación del curso de Diseño Lógico realizada en el año 2004 con financiamiento de la Comisión Sectorial de Enseñanza (Incorporación de Innovaciones en materia de Enseñanza de grado, Llamado 2002).

La reformulación del curso se plantea debido a problemas de masividad que dificultaban la realización de los laboratorios existentes.

En el nuevo curso se modificaron los laboratorios y las formas de evaluación. Se pasó de un sistema clásico en el cual un grupo de estudiantes prepara una práctica en su casa y la realiza en el laboratorio de la Facultad, a un sistema en el cual la realización de la práctica es domiciliaria y se expone lo realizado en una defensa. Para esto fue necesario diseñar y construir una gran cantidad de kits hardware de bajo costo, que son entregados en préstamo a los estudiantes durante todo el semestre. En estos kits los grupos de estudiantes llevan a la práctica sus diseños y los presentan a los docentes en una defensa oral obteniendo así una calificación.

Los resultados de esta experiencia dieron lugar a varias publicaciones en conferencias [90],[89],[88],[54].

## 4.1 Introducción

El curso de Diseño Lógico [3] es obligatorio para los estudiantes de la carrera de Ingeniería Eléctrica y opcional para Ingeniería en Computación. En diferentes versiones se dicta desde 1990, es cursado por estudiantes del 5to semestre (para Ing. Eléctrica), y es el primer curso en donde los estudiantes ven temas específicos de la opción elegida. Esto es un punto de partida importante, ya que los estudiantes no tienen experiencia previa en electrónica ni en instrumentación de laboratorio.

En el nuevo plan de estudios de 1997 este curso se semestralizó y su aprobación pasó a ser por parciales. Esta modalidad semestral se dictó por primera vez en el año 1999, con una carga horaria de 3 horas semanales de teórico, 1 hora y media semanal de ejercicios y un laboratorio de 8 horas presenciales en el semestre (16 semanas).

Uno de los objetivos del curso es que los estudiantes aprendan a diseñar un circuito digital a partir de una especificación no formal, es decir de una descripción verbal o escrita. Enseñar a diseñar implica que el estudiante aprenda no sólo lo que se da en las clases o lo que puede encontrar en la bibliografía sino que se enfrente a todas las etapas del diseño de circuitos que permiten resolver problemas reales. Para esto, como ya se mencionó en la sección 2.2, es fundamental contar con laboratorios bien equipados y con prácticas que permitan una mejor aplicación del conocimiento.

Debido al creciente aumento en la cantidad de estudiantes el dictado o agregado de nuevas prácticas de laboratorio presentaba varias dificultades:

- problemas locativos: horarios libres del laboratorio
- problemas de horarios de los estudiantes
- problemas de horarios de los docentes: para cada laboratorio presencial son necesarios tres docentes durante toda la práctica.
- problemas de coordinación del curso: una misma práctica no puede repetirse por más de tres semanas porque no habría tiempo para realizarla

Para resolver estos problemas se buscó una alternativa en la cual gran parte del trabajo de laboratorio se realice en la casa de los estudiantes. Esta alternativa no es nueva, es bien conocido el trabajo fuera de horario de clases; pero en este caso no se quería perder el contacto con la electrónica. El objetivo final del curso es aprender a diseñar un circuito electrónico, y no queríamos que el laboratorio terminara en una simulación (con algún software de diseño digital que incluyera un simulador), sino que cubriera todas las etapas hasta llegar a un circuito funcionando. Este puede parecer un punto trivial, porque puede suponerse que un circuito que se comportó bien en la simulación va a funcionar tal cual lo simulado; pero la realidad muchas veces nos dice que esto no es así. Con circuitos reales se presentan problemas no vistos o no simulados adecuadamente.

La solución adoptada fue cambiar la forma de realizar los laboratorios y las formas de evaluación, sin modificar las clases teóricas y de ejercicios, buscando poder seguir realizando laboratorios de calidad en condiciones de masividad.

La financiación de este proyecto permitió reformular el curso de Diseño Lógico incorporando más de sesenta kits de hardware que se entregan en préstamo a los estudiantes, nuevas prácticas y nuevas formas de evaluación. La realización de prácticas en la casa de los estudiantes permitió resolver varios de estos problemas e incorporar una práctica más con el mismo equipo docente. Esto se debió fundamentalmente a que se destrabaron las simultaneidades que se tienen que dar en los laboratorios presenciales: un local + tres docentes + 30 estudiantes.

## 4.2 Objetivos del proyecto

### 4.2.1 Objetivos generales

1. Mejorar la calidad de la enseñanza en el curso básico de electrónica digital, en uno de los aspectos más importantes de la misma que es la práctica en laboratorios.
2. Diseñar un laboratorio semipresencial utilizando una herramienta tecnológica que permite pasar a esta modalidad.

3. Lograr que los estudiantes apropien conocimientos con significado con la utilización práctica de los mismos en una modalidad diferente que la utilizada hasta ahora.
4. Poder afrontar un crecimiento en la matrícula sin aumento de horas docentes.

### 4.2.2 Objetivos específicos

1. Desarrollar e implementar un ciclo de laboratorios para Diseño Lógico con la premisa que gran parte del mismo pueda ser realizado en el domicilio de los estudiantes o en las salas de computadoras con las que cuenta la Facultad en el caso que el grupo de estudiantes no disponga de computador.
2. Aumentar la cantidad de las prácticas a realizar.
3. Diseñar un kit hardware de bajo costo que pueda ser entregado a cada grupo de estudiantes durante todo el semestre del curso.
4. Rediseñar las practicas de laboratorio así como documentos guía para cada una de ellas.
5. Diseñar formas de evaluación de aprendizaje y los controles de conocimiento para otorgar la aprobación del laboratorio.
6. Implementar un sistema de seguimiento y apoyo vía Internet.

## 4.3 Historia de la ejecución del proyecto

El proyecto comenzó a ejecutarse durante el año 2003 con el objetivo de dictar el curso en el primer semestre de 2004 con la nueva modalidad. Fue realizado por un equipo de dos Asistentes (Sebastián Fernández y Javier Rodríguez) y tres Ayudantes (Fiorella Haim, Pablo Rolando y Lyl Ciganda), bajo la dirección del autor. Para las etapas de evaluación se contó con la participación de varios integrantes de la Unidad de Enseñanza de la Facultad de Ingeniería (Marina Míguez, Nancy Peré y Virginia Rodés), y en la etapa de montaje de los kits con los funcionarios del taller del IIE (Sergio Beheregaray)

La primera parte del proyecto consistió en el diseño, selección de componentes y construcción de los kits hardware consistentes en: una placa de desarrollo, un manual de usuario [2] y una fuente de alimentación.

La segunda parte del proyecto comprendió el diseño de las prácticas a ser realizadas por los estudiantes. Se buscaron prácticas escalonadas, en las cuales los diseños hechos fueran reutilizados en las prácticas siguientes. De esta forma se logra construir diseños de mayor tamaño y complejidad, pero utilizando bloques que son conocidos porque fueron diseñados de antemano por los propios estudiantes.

La tercer parte fue finalmente la puesta en marcha del curso. Para esto se realizaron reuniones de coordinación entre los docentes de la asignatura y se contó con el apoyo de la unidad de enseñanza, cuyos integrantes observaron algunas instancias de evaluación de conocimientos.

Se tuvieron que resolver un sin número de detalles simples pero nuevos, como por ejemplo diseñar mecanismos de préstamo de las placas al inicio del curso y su devolución al final; diseñar un tutorial de uso del software de diseño Max+Plus II [16] y programación de las placas; implementar mecanismos ágiles de consulta y evacuación de dudas ya sea en clase o vía Internet.

A continuación pasaremos a describir en detalle la nueva metodología del curso y los elementos que la componen.

## 4.4 La nueva metodología del “Laboratorio en casa”

Como ya fue mencionado, la principal característica de esta nueva modalidad es que la mayor parte del trabajo de laboratorio es realizado por los estudiantes en sus casas (o en las salas de computadoras de la Facultad), pero trabajando con hardware real.

Al comienzo del curso se pide a los estudiantes que formen grupos de tres personas, y que se inscriban eligiendo un horario en el cual tendrán que realizar las defensas de sus prácticas. Una vez formados los grupos se reparte un kit a cada uno, que permanece en su poder durante todo el semestre hasta finalizado el curso.

Se establecen claramente las reglas del laboratorio así como las reglas de responsabilidad sobre el kit asignado. Cada grupo es responsable de su kit, debiendo devolverlo en buenas condiciones, en caso de roturas el grupo debe reponerlo o pagarlo. Dado que el kit es de muy bajo costo se supone que un grupo de tres estudiantes puede afrontar un gasto de esa magnitud sin problemas.

Para la gran mayoría de los estudiantes esta es la primera vez que se enfrentan al manejo de hardware real, y en particular a conectar un elemento no estándar a una computadora. Para prevenir accidentes o daños se les recomienda leer cuidadosamente el manual de usuario que se entrega con cada kit.

Las letras de cada práctica se entregan con dos semanas de anticipación. Es decir que los estudiantes tienen dos semanas para resolver el problema planteado y mostrar el diseño realizado. Existen dos juegos completos de prácticas, y dentro de cada uno de ellos hay pequeñas diferencias para asegurar trabajos originales. Cada práctica es realizada por los estudiantes en sus casas o en las salas de computación de la universidad. Si tienen dudas pueden realizar consultas en las clases de ejercicios o vía correo electrónico. Las consultas realizadas en la lista de correo electrónico pueden ser respondidas por docentes, o también por otros estudiantes, en cuyo caso los docentes supervisan que la respuesta sea adecuada.

Un hito importante del curso es la evaluación de las prácticas. Cada grupo debe entregar un informe escrito y realizar una presentación de su diseño frente a un docente y una demostración del circuito funcionando en hardware.

## 4.5 Los kits hardware DL-LAB

El kit hardware DL-LAB consta de una placa electrónica, una fuente de alimentación, software de diseño, un manual de usuario y un tutorial (Fig. 4-1)

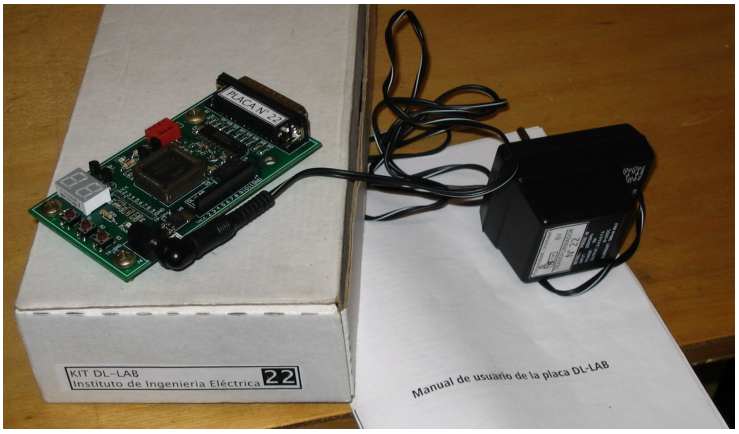


**Fig. 4-1 Kit hardware DL-LAB**

Desde hace unos cuantos años se utilizan en los laboratorios circuitos de lógica programable [57, 110] en lugar de circuitos integrados discretos. La utilización de lógica programable ha demostrado ser muy efectiva, ya que evita problemas tecnológicos y de cableado, y permite que un diseño se lleve rápidamente del papel al circuito. El objetivo entonces fue diseñar una placa (Fig. 4-2) que contara con un chip programable (PLD), y tuviera algunos elementos de entrada salida como para poder implementar allí una variada gama de diseños, pero manteniéndose en un costo muy reducido para poder fabricar una gran cantidad de estos kits, y haciendo hincapié en un diseño robusto.

El bajo costo fue un requisito fundamental por dos motivos, primero porque se iban a construir una gran cantidad de placas, y segundo porque en caso de roturas las mismas debían ser repuestas por los estudiantes, esto implicó una cuidadosa selección de los componentes. El otro criterio de diseño importante fue la robustez, ya que los usuarios de las placas son estudiantes sin experiencia en manejo de dispositivos electrónicos y

las mismas iban a ser utilizadas sin supervisión directa de docentes. Además este criterio de diseño redunda en un muy bajo costo de mantenimiento.

El diseño final consta de un dispositivo de lógica programable con 64 macroceldas, un regulador de tensión de alimentación, reloj, dos displays de 7 segmentos y leds para observar salidas, pulsadores y switches para ingresar entradas, y conectores de expansión para acceder a los pines del chip. La lógica necesaria para la programación del PLD está incluida en la placa evitando así la necesidad de utilizar cables especiales. El PLD se programa desde un PC conectando directamente la placa al puerto de impresora a través de un conector DB25.

El costo final fue de USD 28, sin contar la mano de obra de montaje, es decir que se incluyen los costos del circuito impreso, los componentes y la fuente. La fuente de alimentación es un eliminador de pilas estándar de 500mA.

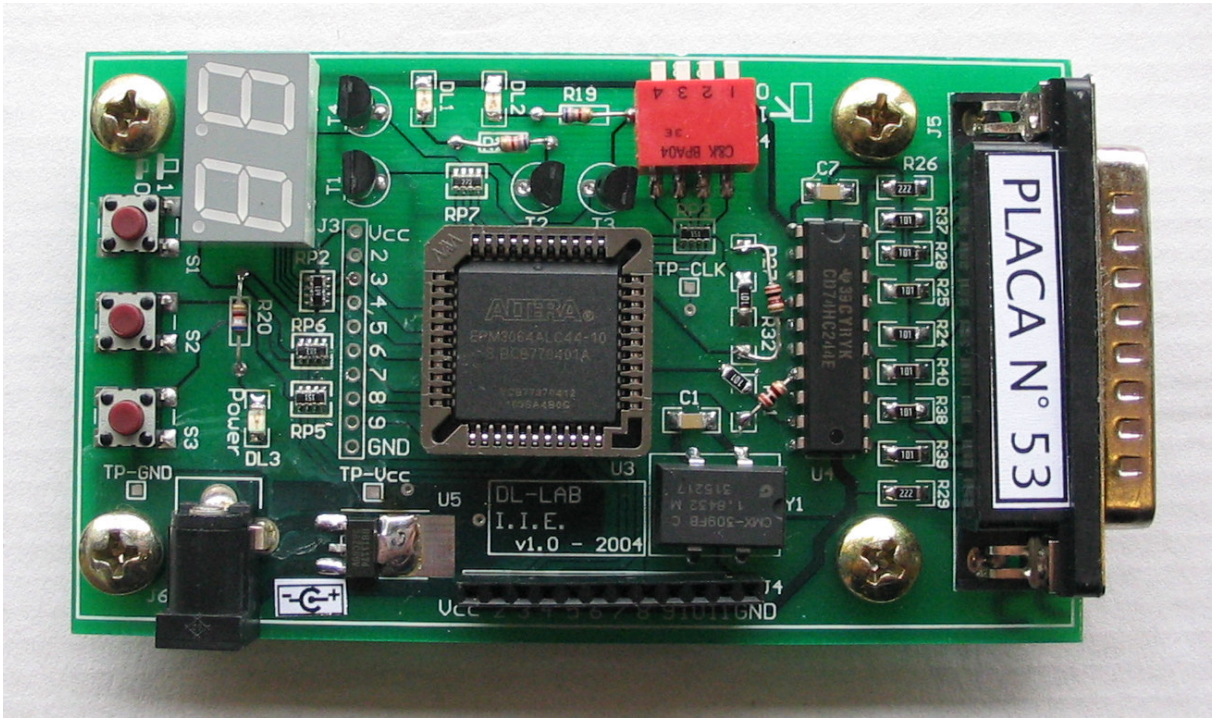


**Fig. 4-2 Placa diseñada para el curso de Diseño Lógico**

Cada kit se entrega a los alumnos en una caja que contiene la placa hardware, la fuente y un manual de usuario. El manual de usuario contiene información detallada de la placa, las precauciones que deben ser tenidas en cuenta para trabajar con ella y las instrucciones necesarias para la programación del PLD.

Asimismo, de la página web del curso se puede descargar un tutorial que indica paso a paso como realizar un primer diseño en la placa. Se aconseja a todos los estudiantes realizar el tutorial completo antes de comenzar con las prácticas de laboratorio.

El software utilizado es el Max+Plus de Altera, dicha herramienta si bien es una herramienta comercial que no fue específicamente diseñada con fines educativos es muy fácil de usar y los estudiantes pueden instalar versiones libres en sus computadoras.

Esta herramienta de software actualmente está siendo discontinuada y va a ser sustituida por el Quartus, por lo que en nuevas versiones del curso se hará una migración a la nueva herramienta.

## 4.6 Descripción de las prácticas de laboratorio

Las tareas consisten en tres prácticas escalonadas que cubren los temas principales del curso, estos son: circuitos combinatorios, circuitos secuenciales modo reloj y circuitos realizados a partir de un lenguaje de descripción hardware de tipo RTL (Register Transfer Language) [60].

Para aumentar la motivación de los estudiantes se buscaron diseños que se aproximen a aplicaciones del mundo real. Para poder lograr esto se particionó el diseño en tres etapas, realizadas como tareas separadas. Cada tarea se realiza en forma independiente, pero forma parte del diseño final, los bloques combinatorios y los circuitos secuenciales de las primeras dos prácticas son reutilizados en la última práctica para dar lugar a una aplicación concreta. De esta forma se logra un circuito

final que resuelve un determinado problema y se introduce un concepto muy importante como es el de la reutilización de bloques previamente diseñados

Al diseñar cada una de las prácticas se tuvo mucho cuidado en respetar los tiempos de dedicación de los estudiantes establecidos en el programa de la asignatura. Para esto se realizaron mediciones de tiempos con estudiantes que ya habían cursado en años anteriores.

Actualmente se cuenta con dos conjuntos diferentes de prácticas: El primero consiste en emular el control del display de un reproductor de CD, como entradas tiene las funciones básicas de PLAY, PAUSE, FF y REW, y como salida muestra en el display el track activo y qué función se está realizando. La otra práctica consiste en un juego de ruleta simplificado, en el cual se pueden realizar apuestas y se muestra la cantidad de dinero disponible.

Si bien la utilización de lógica programable en laboratorios de diseño digital posee grandes ventajas frente a la lógica discreta: se evitan problemas de cableado, introduce mayor confiabilidad, etc., se introduce un efecto "caja negra" en el proceso de diseño que involucra la utilización de herramientas de software y la programación de un chip a través de una computadora. Para lograr una mejor comprensión del proceso, por un lado en el teórico se explica en detalle cómo es la arquitectura interna del PLD utilizado, y por otro se pide que los diseños se hagan en papel hasta llegar al diagrama esquemático final del circuito. Este diagrama esquemático es ingresado directamente en el software de diseño. Experiencias previas realizadas por este equipo docente han mostrado que el ingreso de diseños por esquemáticos es la alternativa que mejor se adapta a nuestro curso, ya que el diseño realizado en papel se traslada directamente al esquemático. En ediciones previas del curso se utilizaron lenguajes de descripción hardware pero la dificultad extra de aprender el lenguaje y el efecto "caja negra" generaba mayores dificultades a los estudiantes en un curso introductorio.

## 4.7 Evaluación

Las tres prácticas de laboratorio están distribuidas a lo largo del semestre. Si bien las prácticas son realizadas por grupos de tres estudiantes que trabajan juntos durante todo el curso, al final del mismo cada alumno tendrá una nota individual por su trabajo en el laboratorio que formará parte de la nota final del curso junto con la de una prueba final.

La evaluación del trabajo de cada práctica se realiza en una presentación oral del diseño frente a un docente y una demostración del circuito funcionando en hardware. Previamente cada grupo debe entregar un informe escrito que incluye la descripción del diseño realizado y una serie de ítems establecidos en la tarea. Esto ayuda al docente a descubrir dificultades o errores cometidos, que pueden ser corregidos o explicados durante la instancia de evaluación.

Cada docente dispone de un listado de ítems a verificar y una serie de preguntas a realizar, como forma de homogeneizar las evaluaciones realizadas. Luego cada integrante del grupo responde en forma individual las preguntas y se le pide que realice pequeñas modificaciones al diseño original, como forma de evaluar su grado de comprensión del problema y el dominio de las herramientas utilizadas. De esta forma es posible evaluar individualmente a cada integrante del grupo.

Cada evaluación lleva aproximadamente una hora por grupo.

El cambio en la forma de evaluación permite que el aprendizaje se haga en forma continua, ya que para cada práctica los estudiantes deben repasar los conceptos trasmitidos en el teórico. Además, permite la interacción de estudiantes y docentes durante la evaluación siendo la instancia de evaluación también una instancia de aprendizaje. Por ejemplo se permite que el estudiante analice los errores en el momento, que aprenda de ellos, y si es posible que los corrija ahí mismo.

## 4.8 Herramientas administrativas

La nueva modalidad de laboratorios en casa implica nuevas tareas administrativas que se agregan al funcionamiento normal del curso, ya que es necesario administrar el préstamo de los kits, y coordinar los horarios de las presentaciones orales de las evaluaciones. Para esto han sido desarrolladas dos herramientas de software que agilizan y automatizan estos procesos. Primero una herramienta de registro vía web que permite realizar la asignación de horarios para las presentaciones orales; y segundo una planilla que calcula en forma automática las notas de cada alumno en función de las evaluaciones realizadas por los docentes para cada práctica y el puntaje obtenido en la prueba final.

Además se ha establecido un sistema de consultas vía correo electrónico para permitir que los estudiantes puedan evacuar sus dudas trabajando en forma remota.

## 4.9 Resultados académicos

Se realizaron medidas del tiempo utilizado por los estudiantes para realizar cada tarea, para ver si coincidía con el tiempo asignado en el programa del curso. De los datos suministrados por los propios estudiantes se obtuvo que la primer tarea les llevó un promedio de 12 horas, mientras las dos últimas unas 15 horas cada una. Esto da un total de 42 horas dedicadas al laboratorio.

Observando el proceso de aprendizaje, la primera impresión del equipo docente fue que los estudiantes que realizaron el curso en la nueva modalidad lograban adquirir mejores habilidades que aquellos que lo habían realizado en las ediciones previas con la modalidad tradicional. Esto se notaba especialmente en el manejo y dominio de las herramientas tanto hardware como software, y en las habilidades de diseño. Esta impresión que en una primera instancia fue totalmente subjetiva se confirmó después de tener los primeros resultados de las evaluaciones. Las notas obtenidas en las tres

ediciones del curso se muestran el la Fig. 4-3, siendo el promedio de 23,1 puntos sobre un total de 25.

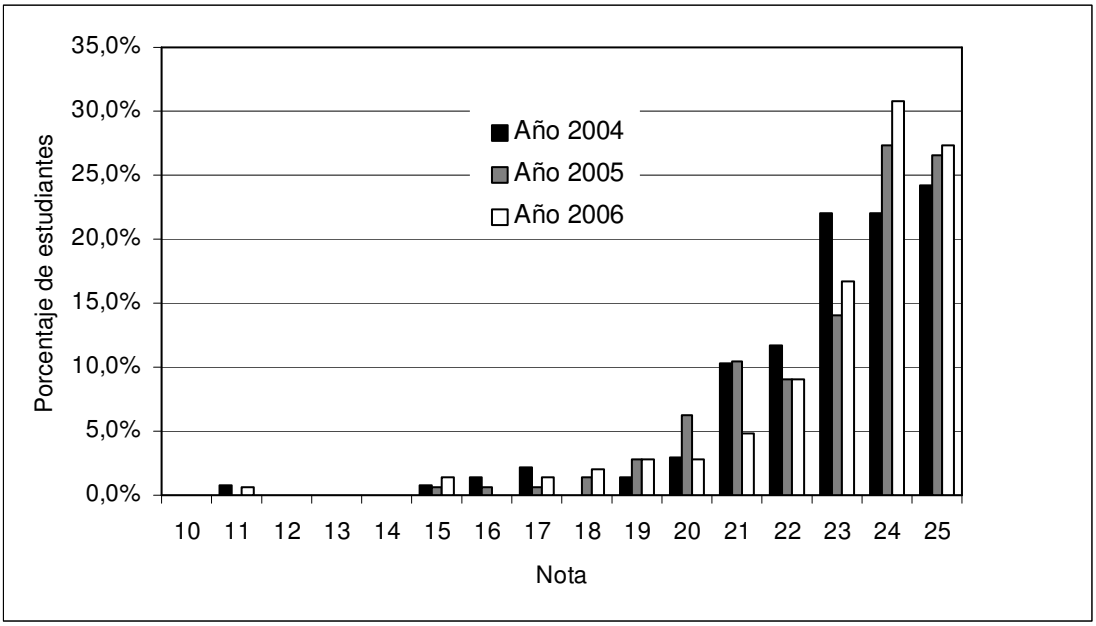


**Fig. 4-3 Notas obtenidas por los estudiantes en el laboratorio**

Si analizamos la nueva modalidad y la comparamos con los objetivos de aprendizaje para los laboratorios de ingeniería definidos por ABET y presentados en la sección 2.2, podemos ver que se cumplen casi la totalidad de los mismos. El que queda claramente incumplido es el objetivo de instrumentación, ya que la nueva metodología no incluye la interacción de los estudiantes con instrumentos de laboratorio tales como osciloscopios, fuentes, etc. Pero podemos decir que el resto de los objetivos se cumplen en forma parcial o total.

Para obtener la opinión de los estudiantes sobre la nueva modalidad del curso se realizó una encuesta diseñada por la Unidad de Enseñanza de la Facultad de Ingeniería. Esta encuesta fue realizada dos veces, la primera al final de la primera edición del curso en el año 2004, y la segunda al final de la tercera edición en el año 2006.

La primer vez que se realizó la encuesta fue respondida por unos 100 estudiantes. La primer pregunta fue: "¿Cuál es tu opinión respecto a la modalidad semipresencial del laboratorio?" Esta pregunta fue respondida por 93 estudiantes, 88 (94,6%) de los cuales tuvieron una opinión positiva, 4 (4,1%) dieron una opinión neutra, y sólo uno (1,1%) dio una opinión negativa. En la edición 2006 las respuestas a la misma pregunta se repartieron de la siguiente manera: 35 (77,8%) opiniones positivas, 6 (13,3%) neutras y 4 (8,9%) negativas. Esta vez la encuesta fue realizada por un grupo menor de estudiantes, y si bien se nota una baja en las opiniones positivas, igual la modalidad es aprobada por una amplia mayoría.

Al preguntarles si incluirían alguna modificación en la forma de organizar el laboratorio dentro de esta modalidad semipresencial, la gran mayoría dijo que no haría modificaciones a la forma propuesta del laboratorio. Muchas de las sugerencias dadas por los estudiantes fueron incorporadas en ediciones posteriores del curso.

## 4.10 Antes y después

En esta sección realizaremos una comparación entre las principales características del nuevo curso y la modalidad anterior.

- *Cantidad de prácticas*: El curso de Diseño Lógico incluía 2 sesiones de laboratorio de 4 horas cada una por grupo de estudiantes, mientras que el curso actual de Diseño Lógico incluye 3 prácticas [1] que cada grupo de 3 estudiantes puede realizar en su casa. Se establece una semana para que cada grupo muestre su diseño y lo defienda en 45 minutos ante un docente de la asignatura.
- *Material de laboratorio*: Se contaba con 9 placas para usar en el laboratorio del IIE y ahora se cuenta con más de sesenta kits de mejor calidad, más prestaciones, y que queda en poder de los estudiantes todo el semestre.
- *Temas*: En la primer sesión se realizaba una práctica de circuitos combinatorios y en la segunda una de circuitos secuenciales modo reloj. Ahora se agregó una

tercer práctica de circuitos secuenciales utilizando un lenguaje de descripción hardware a nivel RTL (Register Transfer Language).

- *Metodología*: Los grupos de 3 estudiantes tenían que resolver un problema, diseñar una solución y dibujar el circuito en el software Max+Plus II antes de concurrir al laboratorio. Ahora además tienen que programar los chips y hacer todas las pruebas de funcionamiento para después realizar una defensa de su trabajo frente a un docente.
- *Evaluación*: Una vez en el laboratorio, se hacía un mini cuestionario individual para controlar que todos hubieran hecho la práctica, y a continuación cada grupo programaba un chip. El laboratorio no tenía calificación, solo se aprobaba o reprobaba. Ahora cada una de las 3 prácticas lleva puntos individuales asignados por el docente en base al diseño, la defensa y el informe presentado.
- *Aprobación del curso*: Antes se exoneraba la asignatura si se aprobaba el laboratorio, se tenía más de 65% sumando los dos parciales y un mínimo de 50% en el segundo parcial; se ganaba el curso si se aprobaba el laboratorio y se tenía más de 25 puntos entre los dos parciales y se reprobaba el curso si se reprobaba el laboratorio y/o no se alcanzaban los 25 puntos entre los dos parciales. Ahora para exonerar hay que aprobar el laboratorio (asistir a las 3 prácticas y obtener un mínimo de 15 puntos) y obtener más de 65 puntos entre las evaluaciones de las prácticas y la evaluación escrita final; para ganar el curso hay que aprobar el laboratorio y tener un mínimo de 25 puntos sumando las evaluaciones y se recursa si se reprueba el laboratorio y/o no se alcanzan los 25 puntos sumando las evaluaciones.
- *Contacto con el material de laboratorio*: Antes cada grupo de 3 estudiantes estaba en contacto con la placa de desarrollo por menos de 8 horas en todo el curso mientras que ahora cada grupo de 3 estudiantes tiene en su casa la placa de desarrollo durante todo el semestre. Esto influye en el entusiasmo de los estudiantes (para muchos de ellos esta es la primera vez que tienen componentes electrónicos en sus manos) y en su capacidad para implementar y testear sus diseños en forma más independiente.

## 4.11 Resultados y Conclusiones

El IIE cuenta ahora con más de 65 kits de lógica programable funcionando. El grupo de electrónica aplicada adquirió a lo largo del proyecto experiencia y habilidades en temas de diseño de placas, componentes electrónicos, chips programables, montaje superficial. Los estudiantes tuvieron a su disposición durante todo el semestre una placa de desarrollo para programar sus diseños. Es una opinión generalizada de todos los docentes del curso que los alumnos que cursaron en la nueva modalidad, en promedio, adquieren un dominio de la herramienta de diseño y programación de chips cualitativamente superior al de los estudiantes que realizaron el curso en su versión anterior.

Desde el punto de vista docente el nuevo curso fue exitoso, se cumplieron ampliamente los objetivos planteados, se diseñaron y construyeron los kits, y se realizaron las primeras tres versiones del curso con la nueva modalidad.

Se pudo apreciar una muy buena receptividad por parte de los estudiantes frente a este nuevo método de dictado del curso, y la misma fue confirmada por dos encuestas realizadas por la Unidad de Enseñanza de la FI (la encuesta completa realizada en 2003 puede verse en el Anexo). Se percibió que trabajaron con independencia, demostrando un gran dominio de la herramienta de software y de la tecnología involucrada; en cuanto a las calificaciones obtenidas en los laboratorios, el promedio fue de 23,1 puntos sobre un total de 25. Un resultado no menor es que todas las placas han sido devueltas en perfecto estado.

Cabe destacar que el nuevo sistema tiene varias ventajas: requiere menor cantidad de horas docentes; no necesita un gran local con equipamiento de laboratorio; flexibiliza los horarios tanto para docentes como para los estudiantes y permite hacer más prácticas en menos tiempo. Estas propiedades hacen que el nuevo sistema sea mucho más escalable, ya que para atender más estudiantes sólo hay que escalar horas docentes y kits hardware.

Estas mismas características junto a su bajo costo comparativo hacen atractiva la idea de aplicar esta experiencia en otros contextos.

## 4.12 Portabilidad de la experiencia

Por sus características, esta experiencia educativa puede ser fácilmente trasladada o adaptada a diferentes contextos. Si bien fue pensada como una forma de lograr una buena calidad de enseñanza en cursos masivos, la misma puede ser aplicada a cursos con menor cantidad de estudiantes, o incluso en cursos no presenciales.

En esta sección se presenta, a modo de guía, el cálculo de horas docentes necesarias para implementar esta metodología de curso. Se comparan cuatro modalidades distintas, que se describen a continuación:

Caso 1 Curso sin laboratorio, con una prueba en la mitad del curso y una prueba final.
Caso 2 Curso con laboratorio tradicional con una prueba en la mitad del curso y una prueba final.
Caso 3 Curso con laboratorio tradicional y una prueba final
Caso 4 Nuestra metodología: laboratorio en casa y una prueba final

El cálculo de horas de cada caso no incluye las horas de las clases teóricas o de ejercicios, y tampoco incluye las horas necesarias para preparar y corregir la prueba final ya que éstas son comunes a todos los casos. Se presenta una comparación de horas teniendo en cuenta estas consideraciones en función del número de estudiantes (Fig. 4-4).

Para el Caso 1 (curso sin laboratorio) la curva graficada representa básicamente las horas necesarias para corrección de la prueba de mitad de curso.

En los casos que incluyen laboratorio tradicional, y basándonos en nuestra experiencia previa para cursos introductorios, asumimos que cada práctica necesita 3 docentes durante 5 horas pudiendo atender hasta 8 grupos de 3 estudiantes en un mismo salón

equipado adecuadamente. Esta modularidad de 24 estudiantes (8 grupos de 3) hace que las curvas correspondientes a los casos que incluyen laboratorios tradicionales aparezcan escalonadas.

Las horas necesarias por nuestra metodología fueron obtenidas de medidas realizadas en Diseño Lógico en las ediciones 2004 y 2005.

Se asume que en ambas modalidades de laboratorio (tradicional y en casa) se realizan tres tareas prácticas.

De esta forma, se puede estimar el costo en horas docentes que conlleva incluir esta metodología en otros tipos de curso; por ejemplo, para un curso de 160 estudiantes sin laboratorios (Caso 1) incorporar trabajo de laboratorio con nuestra metodología le implica un aumento del 47% de sus horas docentes, mientras que incorporarlo con un laboratorio tradicional le significa un incremento del 83% (Caso 3).

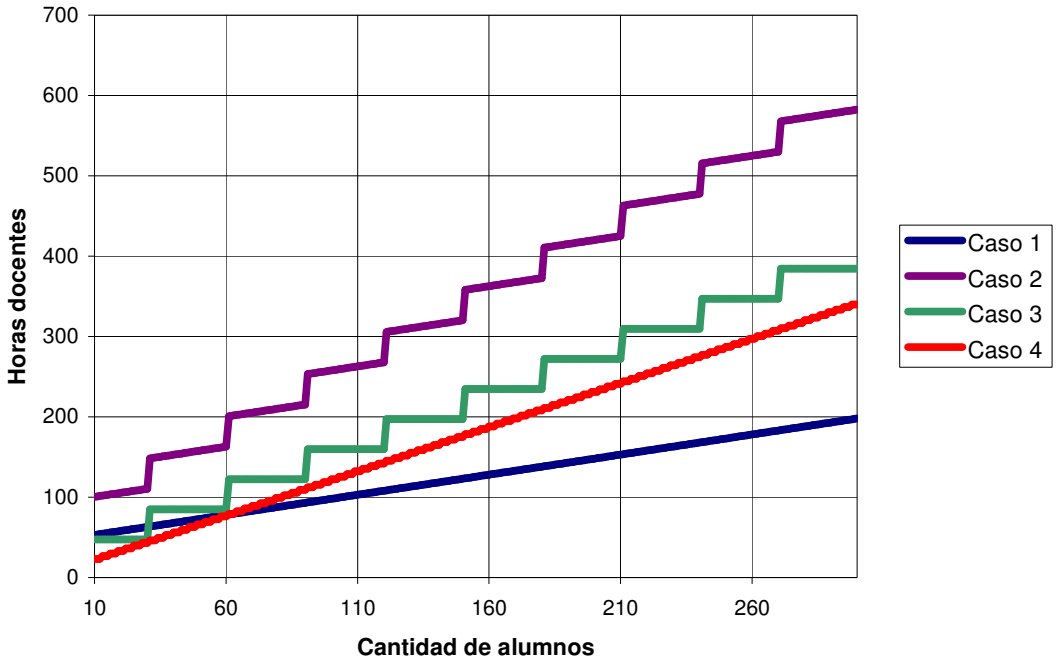


**Fig. 4-4 Comparación de horas docentes necesarias para cada caso**

En nuestro curso, pasamos de impartir un laboratorio tradicional con dos pruebas semestrales (Caso 2) a la modalidad de laboratorio en casa (Caso 4). Esto representa un ahorro del orden del 45% en horas docentes para el caso de 160 estudiantes.

Para completar el cálculo del costo de implementación de nuestra metodología, al costo de horas docentes hay que agregarle la inversión material inicial. Esta inversión consiste casi exclusivamente en el costo de las placas, siendo necesarias una cada tres estudiantes. Como ya mencionamos el costo de los componentes de cada placa asciende a USD 28 y a esto hay que agregarle el costo de la mano de obra necesaria para el armado de las mismas (el montaje de un kit insume aproximadamente unas cuatro horas).

En nuestros tres años de experiencia con esta metodología, el costo de mantenimiento de las placas ha sido prácticamente nulo: solo se detectaron dos casos de soldadura fría que no implicaron costos de materiales y fueron rápidamente resueltos.

# Capítulo 5

# Aplicaciones en Enseñanza Avanzadas

En este capítulo se presentan una serie de experiencias desarrolladas como proyecto de grado por varios grupos de estudiantes. El Proyecto de fin de carrera es una asignatura en la cual se aplican en forma práctica todos los conocimientos adquiridos previamente. Los proyectos se realizan en grupos de dos o tres estudiantes, con una dedicación estimada por estudiante de 525 horas, equivalentes a 35 créditos.

Se destacan las experiencias recogidas haciendo hincapié en cómo utilizando, la tecnología de la lógica programable, grupos de estudiantes de grado pueden realizar diseños digitales de alta complejidad y calidad en contextos de bajos recursos.

Se comienza describiendo brevemente una serie de trabajos previos que sirvieron como base para la posterior realización de los proyectos por parte de los estudiantes. Estos trabajos previos fueron el origen de una serie de proyectos relativos al desarrollo de plataformas reconfigurables y sus aplicaciones. Se diseñaron y construyeron dos plataformas con interfaz PCI de diferente grado de complejidad basadas en lógica programable. Ambos desarrollos fueron realizados por grupos de estudiantes de grado como parte de su proyecto de fin de carrera. Se describen brevemente las características de las plataformas diseñadas, las opciones de diseño elegidas y las aplicaciones que se realizaron sobre ellas por otros grupos de estudiantes como forma de validarlas.
Parte de los resultados de estos trabajos fueron presentados en [85].

## 5.1 Antecedentes

Los proyectos de estudiantes que se presentan forman parte de una línea de trabajo del Grupo de Electrónica Aplicada consistente en la utilización de plataformas reconfigurables asociadas a una computadora personal.

La alternativa elegida fue utilizar una placa con FPGAs como un coprocesador reconfigurable trabajando en conjunto con un PC. La arquitectura utilizada consta entonces de una placa reconfigurable trabajando en conjunto con el microprocesador del PC en modalidad coprocesador.

Un coprocesador reconfigurable consta de una o más FPGAs, memoria RAM, y una interfaz para poder conectarse al motherboard de una computadora personal. Generalmente el procesador de la computadora personal puede acceder a la RAM del coprocesador y enviar datos al FPGA. Usualmente las funciones que van a ser implementadas en el coprocesador son seleccionadas del software que ya fue escrito para ser ejecutado en el computador. La selección de estas funciones se hace pensando en lograr el máximo speedup, y la restricción principal es el tamaño de la FPGA [83].

La idea básica es poder partir un determinado algoritmo y realizar una parte en el microprocesador del PC y otra en la placa reconfigurable. Además el PC se encarga del reparto de las tareas y de la reprogramación de la placa, así como de toda la interfaz con el usuario.

Esta arquitectura se presenta atractiva por varios motivos. En principio una ventaja es que la placa reconfigurable es el único hardware a utilizar además del computador. Un PC es una plataforma ampliamente difundida y de bajo costo, que cuenta con varios periféricos utilizables. El PC se encarga de resolver la entrada salida de datos sin hardware adicional, de la interfaz con la aplicación, y de reconfigurar los dispositivos.

El esquema de la Fig. 5-1 muestra la arquitectura utilizada en comparación con el método clásico.

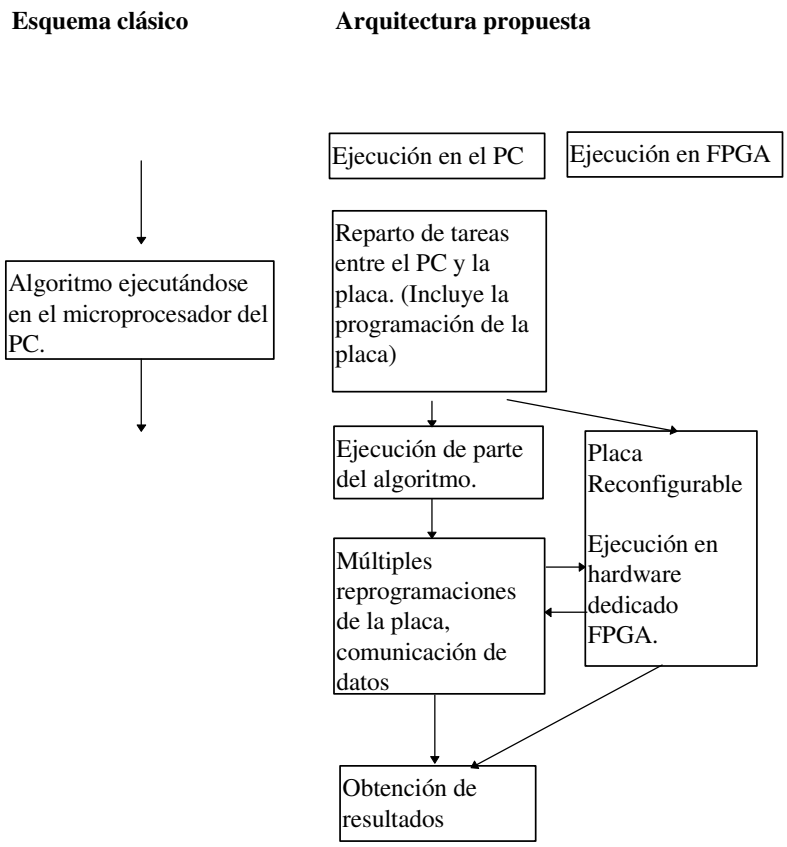


**Fig. 5-1 Comparación entre la ejecución de un algoritmo en software con una ejecución mixta hardware-software**

También existen ciertas desventajas por utilizar esta configuración. Dependiendo del tipo de aplicación los puntos críticos son la capacidad de procesamiento de la placa reconfigurable, cantidad de memoria disponible en la misma, su bus de datos o lo que es lo mismo el ancho de banda de entrada salida de datos, y la velocidad de reconfiguración.

La propia arquitectura de las computadoras personales tiene limitaciones que van a influir en los resultados obtenidos. La más importante es el cuello de botella en el ancho de banda de la trasmisión de datos entre la placa reconfigurable y el PC. Esto

va a influir tanto en los datos que van a ser procesados en la placa, como en la velocidad de reconfiguración de la misma.

La utilización de hardware dedicado dentro de un computador personal requiere el desarrollo de interfaces software que manejen este hardware. Dependiendo del sistema operativo a utilizar habrá que desarrollar diferentes drivers o módulos que realicen la capa más cercana al hardware y ofrezcan al software de aplicación una interfaz cómoda de usar.

Las primeras experiencias exitosas realizadas datan del año 1998 y fueron con una plataforma donada por la empresa Altera. Esta plataforma, la RIPP10, tenía una interfaz ISA lo que hacía muy fácil el desarrollo del software de comunicaciones con el PC. Se realizaron aplicaciones de aceleración de cálculos de redes neuronales para control y dieron lugar a varias publicaciones [86, 87, 91].

### 5.1.1 Trabajos realizados con la plataforma ARC-PCI

Los resultados exitosos hicieron posible la gestión y obtención de una nueva donación de la empresa Altera, en este caso una placa con interfaz PCI. La placa ARC-PCI (Altera Reconfigurable Computer - Peripheral Component Interconnect) contiene tres FPGAs de la familia FLEX10K, memoria RAM estática y un PLL. Uno de los FPGAs está dedicado a la interfaz PCI y los otros dos quedan libres para desarrollar las aplicaciones de usuario.

La placa estaba en estado experimental, y recién un tiempo después de haberla recibido, Altera suministró un demo muy primitivo que incluía la descripción AHDL del circuito a ser cargado en el FPGA de interfaz PCI. Este circuito permitía cargar a través del PCI a los FPGAs restantes con aplicaciones de usuario. Del lado de software el demo incluía un driver para funcionar en Windows NT, pero cerrado, ya que utilizaba el producto WinRT producido por la empresa BlueWater Systems. La interfaz diseñada era muy lenta, no permitiendo transferencias "burst" en ningún sentido, y trabajaba a una velocidad muy inferior a los 33MHz permitidos por el PCI.

Por otro lado, el software utilizado no podía modificarse ya que no se contaban con licencias del WinRT utilizado para el driver en Windows NT.

Por estos motivos se decidió mejorar la interfaz PCI para poder aprovechar mejor las capacidades de la placa. El objetivo de esta parte del trabajo fue entonces diseñar nuevamente el hardware y el software de la interfaz PCI, pero optimizando las velocidades de transferencia de datos entre el PC y la placa, ya sean datos a procesar o velocidad en la reconfiguración de los FPGAs de usuario.

El propósito fue contar con una plataforma hardware reconfigurable, fácil de usar, y que utilizara la máxima velocidad de transferencia de datos, de modo de poder diseñar aplicaciones sin tener que diseñar todas las veces la interfaz de entrada salida.

Para ello se utilizó un core PCI de Altera, y se realizó un diseño incluyendo ese core que tuviera las características antes mencionadas:

- posibilidad de escrituras y lecturas "burst" a la RAM
- posibilidad de reconfigurar los chips de usuario via bus PCI
- bits de control y status necesarios

También se escribió el software necesario para comunicarse vía PCI con la placa y realizar las operaciones de lectura escritura en RAM, y reprogramación de los chips de usuario.

**Hardware**

El hardware diseñado incluye:

- PCI Iniciator/Target (utilizando PCI core de Altera)
- Configuración de los chips de usuario (utiliza parcialmente un diseño previo de la Universidad de Waterloo [24])
- Arbitros de buses

Para dar una idea del tamaño del diseño podemos decir que se escribieron unas 1400 líneas de código AHDL.

**Software**

Se realizaron programas de pruebas para leer y escribir dispositivos mapeados en la zona PCI. Se escribieron rutinas que permiten realizar transferencias en modo "burst" y unitarias desde y hacia las memorias de la ARCPCI, así como reconfigurar los FPGAs de usuario mediante comandos enviados desde el PCI.

**Resultados**

Se alcanzó plenamente el objetivo buscado, la plataforma y los diseños realizados en ella funcionaron adecuadamente, y pudo ser utilizada tanto por docentes como por estudiantes realizando su proyecto de fin de carrera.

En cuanto a las optimizaciones de velocidad buscadas, se logró obtener la máxima velocidad de transferencias posible en un sentido: desde el PC hacia la ARCPCI. Pero se tuvo que insertar un tiempo de espera cuando se transfieren datos desde la ARCPCI hacia el PC, a continuación se resumen las velocidades de transferencia obtenidas en cada modo:

**Velocidad de transferencia**

Modo PCI Target:

- write burst sin Twait 33MHz
- read single agregando un Twait 33MHz

Modo PCI Master:

- La placa ARCPCI puede escribir directamente a la memoria del PC en modo burst agregando un Twait 33MHz

**Configuración de FPGAs**

El mecanismo elegido para configurar las FPGAs de usuario a través del bus PCI fue primero cargar el archivo de configuración en una de las memorias de la ARCPCI y luego lanzar una máquina de estados que realiza la configuración en background, dicha configuración demora 5.5 ms.

### 5.1.2 Proyectos de estudiantes realizados en la plataforma ARC-PCI

La puesta a punto de la plataforma y el ambiente de desarrollo permitió llevar a la práctica los dos primeros proyectos de estudiantes en la ARC-PCI, permitiendo así la validación de la plataforma.

#### 5.1.2.1 Utilización de Lenguaje C para especificación hardware

Este proyecto fue realizado por Daniel Gómez, Gabriel Heguy, Matías Puig, y dirigido por el autor. A continuación se hace una descripción breve de sus principales características, la documentación completa del proyecto puede verse en [51].

Se estudió la utilización de lenguaje C en el diseño de aplicaciones en hardware. Se evaluaron distintas herramientas de conversión de C a algún lenguaje del tipo HDL.

En la primer etapa se realizó una búsqueda de información sobre herramientas libres de este tipo y de las encontradas se seleccionaron dos: Handel C y Transmogrifier C. La segunda etapa fue de familiarización con las herramientas, investigando los circuitos generados a partir de diseños escritos en C. En la tercer etapa se analizaron distintos ejemplos para ambas herramientas como algoritmos de división y raíz cuadrada, un árbitro de bus, ejemplo de utilización de memorias externas e internas, un multiplicador con pipeline y ejemplos de operaciones con matrices. En todos estos ejemplos se relevaron los recursos ocupados y la frecuencia máxima alcanzada para chips de la familia Altera y en algunos casos también se hizo para chips de Xilinx. Además se hicieron comparaciones en algunos ejemplos con versiones escritas en VHDL.

Finalmente se eligió una de las herramientas (Handel C) y un algoritmo escrito en C para analizar la facilidad de traducirlo de la forma más directa posible para su implementación en hardware. El diseño seleccionado fue un filtro de imágenes en tonos de grises que se implementó en la placa ARC-PCI de Altera. Además se

realizaron comparaciones de performance entre este diseño y el filtro original (en C) corriendo en diferentes PC.

Los objetivos específicos de esta actividad fueron:

- Estudio del estado del arte en herramientas de pasaje automático de software a hardware.
- Familiarización con las herramientas y selección de dos para su evaluación.
- Análisis de funcionamiento de las herramientas elegidas.
- Comparación de éstas utilizando ejemplos sencillos.
- Elección de una de ellas e implementación de un algoritmo más complejo.

Para evaluar el funcionamiento de los compiladores se usaron ejemplos típicos de desarrollos en Hardware (árbitro de bus, memoria FIFO) y otros de Software (raíz cuadrada, producto de matrices). En algunos de estos casos se realizó también una solución en VHDL para comparar el tiempo de diseño y características como recursos del chip y frecuencia máxima.

Resultados y conclusiones

En general el diseño se realizó más rápido y fácil que por los métodos tradicionales, permitiendo de esta forma ciclos de prototipación rápidos. También se concluyó que la curva de aprendizaje de estos compiladores es mejor que para VHDL, sobre todo por Ingenieros no Eléctricos, pues se necesita solamente saber programar en C, aunque para obtener óptimos resultados además es necesario conocer la estructura de los circuitos generados y tener nociones de diseño de circuitos.

Tanto los diseños implementados en Transmogrifier C como los implementados con Handel C son de tamaño superior, consumen mayor cantidad de FF y celdas lógicas, que sus pares en VHDL. Estos compiladores también producen diseños más lentos (hasta 50% menos). Sin embargo, los tiempos de diseño en todos los casos fueron menores, y en particular con Handel C fueron aún menores. Para los ejemplos desarrollados la diferencia de horas de diseño entre VHDL y C fue de un 25% (en general) pero estos son ejemplos sencillos. Para grandes diseños o sin ir tan lejos en la

aplicación final realizada, esta diferencia puede ser de más del 50%, pues por ejemplo realizar lazos de control en C es muy sencillo (poner While) mientras que en VHDL es una tarea muy tediosa.

La codificación de un algoritmo dado presenta un grado similar de complejidad en su realización en lenguaje ANSI C y en las variantes de C definidas por Transmogrifier y Handel.

Por otro lado, en cuanto a flexibilidad, como el circuito generado está especificado en VHDL, puede ser fácilmente integrado (o instanciado) en diseños más complejos. Por ejemplo se pueden realizar las tareas tediosas de un diseño desde C y los detalles finos hacerlos en VHDL.

En cuanto a las limitaciones encontradas cabe destacar que, no es muy sencillo adaptar un programa ya escrito en C a Transmogrifier C o Handel C, depende de su complejidad (por ejemplo no se puede usar recursión) y sobre todo de los tipos utilizados (punteros, arrays, etc.). Otro factor importante para poder realizar las traducciones desde C son las operaciones realizadas pues no se cuenta con, por ejemplo, división, potencia o funciones trigonométricas. Estas herramientas sólo generan circuitos secuenciales modo reloj, con un único reloj global y todos los flip-flop sensibles al mismo flanco (de subida por defecto).

A continuación se resumen las características más relevantes de esta metodología:

- Rápido desarrollo de algoritmos.
- Facilidad para modificar el diseño.
- Ciclos de prototipación rápidos.
- Fácil programación o descripción de los circuitos desde C
- Buena integración con otros circuitos
- Amplio espectro de requerimientos cubiertos

#### 5.1.2.2 Redes neuronales en hardware

Este proyecto fue realizado por Daniel Ferrer, Roberto Fleitas y Ramiro González, y dirigido por Julio Pérez.

El objetivo principal de esta actividad fue diseñar una red neuronal implementada en hardware con el propósito de comparar la performance de la misma frente a su implementación software, como forma de acelerar cálculo en hardware digital. Los objetivos específicos preveían implementar en hardware digital una estructura de red neuronal tipo perceptrón multicapa en la ARC-PCI.

Se obtuvo un diseño descrito en AHDL, que mediante la asignación de parámetros utilizando el programa Max+Plus II, permite sintetizar un circuito digital que implementa una red neuronal de topología variable.

Se cumplieron satisfactoriamente los objetivos planteados al inicio del proyecto, superando en gran medida el primero de los mismos. Se obtuvo un diseño de una red neuronal perceptrón multicapa ampliamente parametrizable con múltiples grados de libertad en su arquitectura y forma de implementación en el hardware. La madurez alcanzada en el diseño de los principales circuitos de la red neuronal a lo largo de todo el proyecto permitió obtener una gran experiencia en el diseño de hardware digital. Se puede decir además que se obtuvo una madurez en la realización de proyectos de gran magnitud. Los resultados completos pueden verse en [43] y fueron publicados en un congreso internacional [44].

## 5.2 Plataforma IIE-PCI

Este proyecto fue realizado por Sebastián Fernández y Ciro Mondueri y dirigido por el autor.

La plataforma IIE-PCI [41, 42] fue el primer desarrollo hardware realizado en el IIE utilizando la interfaz PCI [5, 100] y comenzó a principios del año 2002. En el grupo

de trabajo había experiencia en diseño de placas ISA, y en la utilización de placas PCI pero no en su diseño, por lo tanto se trató de definir una serie de objetivos muy modestos:

- realizar una placa PCI lo más sencilla posible
- la interfaz PCI debía estar realizada en lógica programable
- debía tener memoria interna
- costo menor a 250 dólares

La limitante del bajo costo y la poca experiencia en el IIE en el diseño de placas similares hizo que se optara por una placa con un solo FPGA. Esta decisión limitó las prestaciones de la placa ya que no puede ser reconfigurada a través del bus PCI.

### 5.2.3 Características Técnicas

Las principales características de la placa desarrollada son las siguientes:

- Compatible con bus PCI de 32 bit, 3.3V y 5V
- 128Mbit de memoria SDRAM on-board
- FPGA de la familia Altera ACEX EP1K100PQ208 [4]
- Expansores con señales provenientes del FPGA
    - 31 señales de propósito general
    - 32 señales compartidas con el bus de datos de la SDRAM
- Conversores DC-DC para generar las fuentes de 3.3v y 2.5v utilizadas por los integrados.
- PLL que permite regenerar y multiplicar la frecuencia del reloj PCI o de un cristal en la placa.
- El FPGA puede ser programado utilizando:
    - EPROM de programación EPC2, se autoconfigura al encenderse
    - protocolo JTAG o configuración serie pasiva (PS) mediante el cable de programación ByteBlasterMV
- Llaves y leds de propósito general
- Circuito impreso de 4 capas (señal, tierra, alimentación, señal)

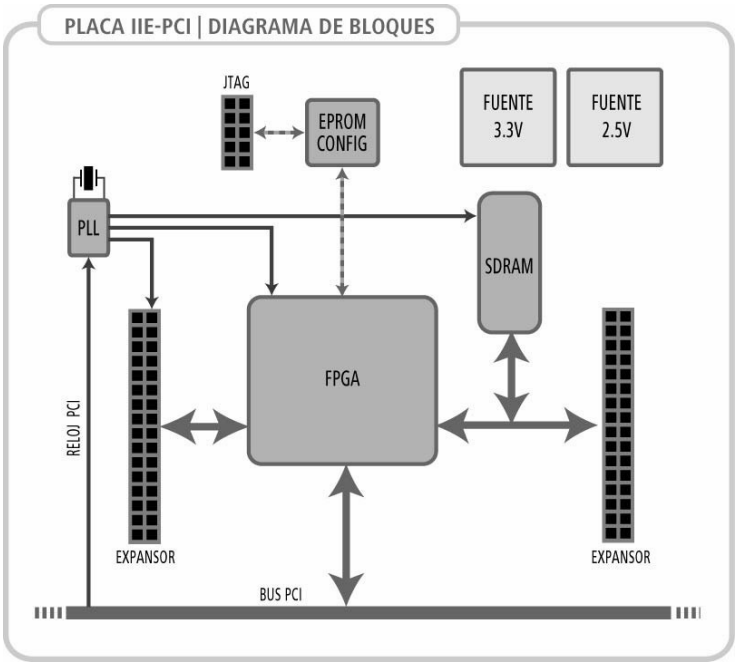


**Fig. 5-2 Diagrama de la arquitectura de la placa diseñada.**

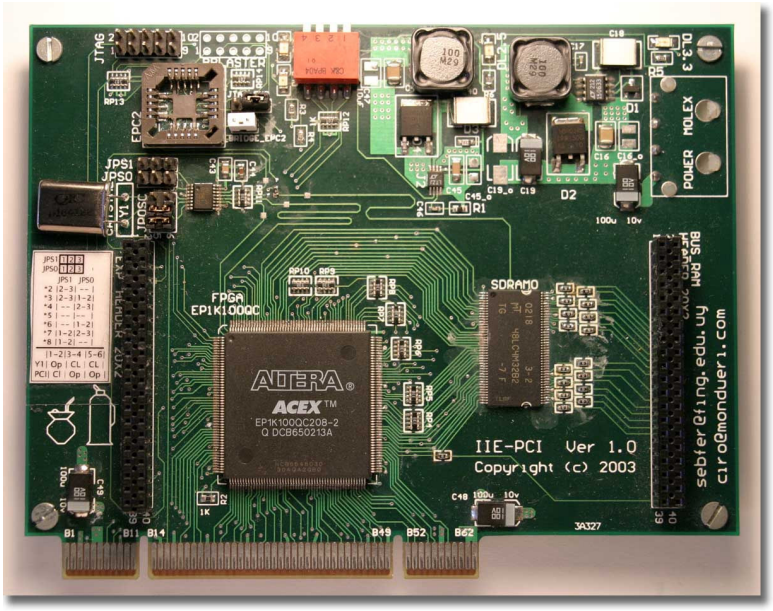


**Fig. 5-3 Vista superior de la placa**

## 5.3 Aplicaciones realizadas en la plataforma IIE-PCI

La placa ha sido utilizada por varios grupos de estudiantes de asignaturas de grado y fin de grado para diversos proyectos. Esto ha permitido validar totalmente el diseño y ha demostrado que sus características permiten el desarrollo y prueba de las más diversas aplicaciones. A continuación se describen brevemente las principales aplicaciones realizadas.

### 5.3.4 Diseño de un IP core PCI propio con un driver para Linux

La primer aplicación implementada en la placa IIE-PCI fue un IP core Target PCI con interfaz Wishbone [6] y un driver Linux para probarlo [98]. Esta primer aplicación sirvió para validar el diseño eléctrico de la interfaz PCI, y fue realizada por los mismos estudiantes que diseñaron la placa IIE-PCI.

La placa y esta aplicación fueron probadas en varias máquinas sin mayores inconvenientes.

El driver PCI para Linux incluye herramientas que permiten realizar estadística de las transferencias de datos realizadas en el bus. Esto sirvió para realizar diferentes pruebas y poder medir el desempeño del core PCI diseñado.

### 5.3.5 Diseño de un controlador de SDRAM

Como parte de una asignatura de grado, un grupo de estudiantes desarrolló un controlador para la SDRAM de la placa.

Este controlador fue sintetizado en el FPGA junto con el core PCI ya diseñado. Dado que ambos diseños contaban con la misma interfaz de aplicación, su integración no presentó ningún tipo de problemas.

Este diseño completo fue utilizado para validar el funcionamiento de la memoria interna de la placa.

### 5.3.6 Diseño de un coprocesador para ruteo IP

La placa IIE-PCI fue utilizada en una aplicación donde se sintetizó un diseño compuesto por tres cores: el core PCI, el controlador de SDRAM y un diseño que implementa el ruteo de tráfico IP en hardware. Este proyecto también fue llevado a cabo por un grupo de estudiantes Ernesto Bazzano, Diego Alcetegaray, Fabrizio Luongo y dirigido por Julio Pérez [21].

La aplicación fue probada en una máquina con sistema operativo Linux donde se modificaron las funciones de ruteo para que en lugar de realizarlas por software estas fueran realizadas en la placa conectada al bus PCI.

### 5.3.7 Filtrado de imágenes

La placa fue utilizada por varios grupos de estudiantes de una asignatura de grado para diseñar y probar algoritmos de procesamiento de imágenes. Algunos de estos proyectos fueron sintetizados junto al controlador de memoria.

## 5.4 Plataforma reconfigurable PGVirtex

El proyecto PGVirtex [99] fue realizado por Jimena Saporiti, Agustín Villavedra y Silvia Gómez y dirigido por el autor.

Consistió en el desarrollo de una segunda plataforma reconfigurable de mayor envergadura que la IIE-PCI utilizando uno de los chips de la familia Virtex-E de Xilinx (XCV1000E) donados al IIE.

Aprovechando la experiencia adquirida anteriormente y los buenos resultados obtenidos con la IIE-PCI se decidió diseñar y construir una segunda placa de mayor capacidad y complejidad. Esto significó nuevos desafíos dadas las tecnologías utilizadas: integrados con encapsulados BGAs y montaje superficial de muy alta densidad. La fabricación de los circuitos impresos así como la soldadura de los chips BGAs fue realizada en EEUU, pero todo el diseño se realizó localmente.

### 5.4.8 Características Técnicas

Las principales características de la placa desarrollada son las siguientes:

- Compatible con bus PCI de 32 bit, 3.3V y 5V
- Comunicación con bus PCI a través de integrado PCI9054 de la empresa PLX
- Dos memorias SDRAM de 128Mbit cada una.
- FPGA de la familia Virtex-E de Xilinx [119]. Este FPGA tiene poco más de 330.000 compuertas lógicas y 404 puertos de entrada/salida, distribuidos en un encapsulado BGA (Ball Grid Array) de 560 pines.
- Interfaces de entrada/salida: PCI, RS-232, USB y 5 puertos Ethernet
- Conversores DC-DC para generar las fuentes de 1.8 y 3.3v utilizadas por los integrados.
- PLL que permite regenerar y multiplicar la frecuencia del reloj PCI o de un cristal en la placa.
- Llaves y leds de propósito general
- Circuito impreso de 6 capas

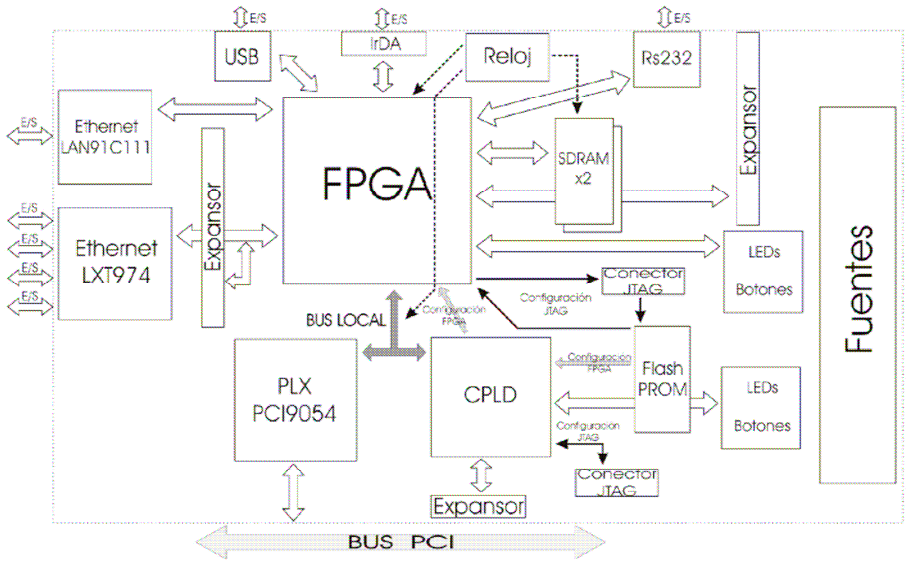


**Fig. 5-4 Diagrama de la arquitectura de la placa diseñada**

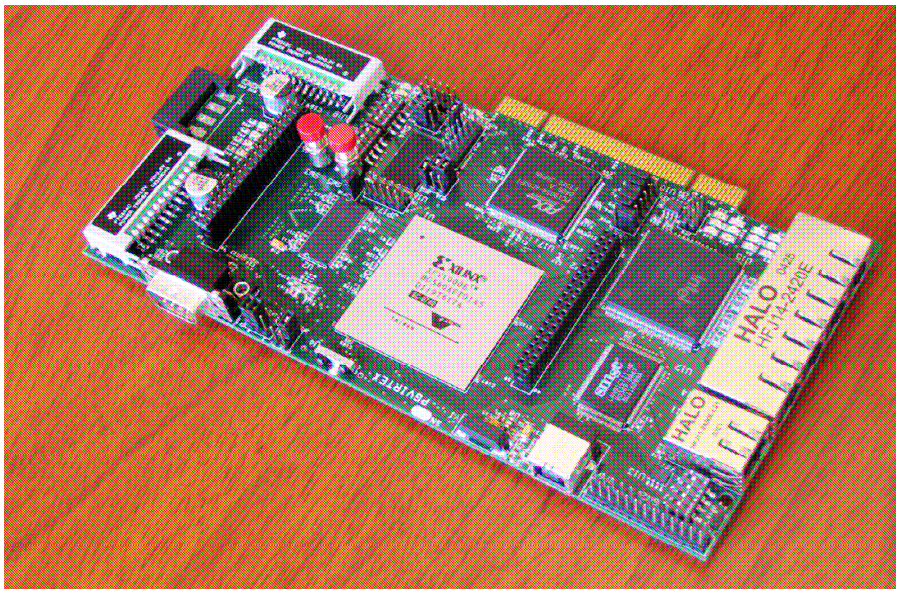

**Fig. 5-5 Vista superior de la placa**

Esta placa está siendo actualmente utilizada por otros grupos de estudiantes, también en sus proyectos de fin de carrera, pero los mismos aún no han finalizado.

## 5.5 Conclusiones y comentarios finales

Como conclusión principal podemos remarcar que en los casos presentados se cumplieron ampliamente los objetivos buscados, es decir que fue posible que estudiantes de grado realizaran el diseño y la implementación de circuitos digitales complejos como proyecto de fin de carrera con recursos muy acotados. Las plataformas reconfigurables desarrolladas han sido y están siendo utilizadas en varias aplicaciones en aceleración de algoritmos así como bancos de prueba para la validación de IP cores.

Creemos que desde el punto de vista educativo hay varios aspectos que vale la pena destacar. El primero es que los proyectos que involucran desarrollo hardware llevaron más tiempo de lo previsto, quizá no en la parte estrictamente de diseño e implementación de las plataformas, pero sí en una gran cantidad de tareas accesorias que hubo que resolver en cada caso, como por ejemplo importar todos los componentes y los circuitos impresos, trámites, compras fuera del país, etc. El desarrollo de las plataformas puede evaluarse en unos ocho meses de trabajo del equipo. Los costos se mantuvieron dentro de los rangos especificados inicialmente que fueron USD 250 para la IIE-PCI y USD 500 para la PGVirtex.

El segundo aspecto interesante es el crecimiento en complejidad entre el desarrollo de la primer plataforma y la segunda. Esto puede verse analizando los principales hitos de cada proyecto:

IIE-PCI
1. diseño multicapa
2. soldadura manual y con poco equipamiento de encapsulados de patas muy finas, se pudieron soldar integrados de 0.5mm entre patas
3. distribución de señales digitales de alta velocidad

PGVirtex
1. utilización integrados con encapsulados BGA (contratando la soldadura)

2. diseño y montaje de circuitos de comunicación (USB, Ethernet)
3. reprogramación por PCI

Esto se debió principalmente a la selección de los proyectos a realizar de modo que fueran modulares y el segundo utilizara los conocimientos adquiridos en el primero.

El tercer aspecto a destacar son las capacidades generadas dentro del IIE gracias a la realización de estos proyectos

- Diseño PCI: sofware y hardware
- Desarrollo de "device drivers" PCI para Linux
- Experiencia en diseño de impresos con integrados de alta densidad
- Diseño de aplicaciones en FPGAs (diseño digital)
- Dominio de lenguajes de especificación hardware (VHDL y Verilog)
- Manejo de herramientas software de diseño y simulación
- Manejo de herramientas software de diseño de impresos
- Realización de soldadura de montaje superficial
- Contratación de servicios de fabricación de circuitos impresos en el exterior, así como la realización de soldaduras de dispositivos BGAs

Esta experiencia confirmó nuevamente algo ya conocido: que diseñar y realizar hardware en Uruguay siempre presenta dificultades, como ser la lejanía con los proveedores o el manejo de tecnologías de punta en las que no hay experiencia. Solamente la soldadura de los circuitos integrados de alta densidad es un problema, ya que el tipo de encapsulado y la gran cantidad de pines hacen que sea necesario contratar este trabajo y localmente solo se pueden soldar hasta una cierta densidad debiendo en otros casos contratar el servicio en el exterior. La importación (y exportación-soldadura-importación) fue otro problema, debido a burocracias y altos costos de los despachantes. Muchas veces los trámites fueron hechos por los propios integrantes del equipo investigador. Todos estos inconvenientes hacen que se pierda mucho tiempo en resolver aspectos no técnicos pero que son fundamentales para la finalización en tiempo y forma de los proyectos.

Finalmente queremos decir que estamos convencidos que proponer a estudiantes de grado este tipo de proyectos, que van más allá del simple diseño de un circuito, hace que salgan preparados para enfrentar situaciones del mundo real.

# Capítulo 6

# Conclusiones

La lógica programable ofrece una excelente oportunidad para experimentar nuevas alternativas de enseñanza de diseño electrónico digital. La flexibilidad de la tecnología permite implementar diversos diseños sobre una misma plataforma dándole al estudiante la posibilidad de ensayar, equivocarse, y corregir sus errores, aprendiendo en todo el proceso.

En el caso del curso de Diseño Lógico, que es un curso introductorio y de carácter masivo se presentó una alternativa al modelo clásico de laboratorios. Esta alternativa fue muy bien recibida por los estudiantes, y es más fácil de implementar que los laboratorios clásicos, tanto en la cantidad de horas docentes como en la utilización de recursos escasos: salones, computadoras. Pensamos que esta alternativa ofrece una enseñanza de excelencia, y que incluso profundiza el contacto del estudiante con las herramientas y plataformas de diseño y experimentación; ya que el kit utilizado permanece en poder de los estudiantes durante todo el semestre.

Aquí se plantea la primer idea de trabajo futuro que es fomentar la creatividad de los estudiantes con diseños que vayan más allá de los estrictamente propuestos en el curso. Esto todavía no se ha implementado, pero ya hemos analizado varias alternativas para llevarlo adelante. Incluso hubo estudiantes que plantearon la posibilidad de adquirir el kit del curso para poder seguir implementando sus propios diseños.

Pensamos que la experiencia en el curso de Diseño Lógico es trasladable a otros contextos, principalmente a aquellos donde la gran cantidad de estudiantes hagan peligrar una buena realización de laboratorios. Si bien el kit diseñado es muy pequeño

en cuanto a la cantidad de celdas disponibles para realizar los diseños, ya contamos con una buena cantidad de problemas que se pueden implementar sin dificultades en la placa. Incluso para aquellos cursos que requieran dispositivos de mayor tamaño pueden re-diseñar la placa sin grandes incrementos del costo final.

En cuanto a la segunda experiencia, el diseño de plataformas reconfigurables con interfaz PCI, planteó desafíos muy interesantes. Allí los grupos de estudiantes tuvieron que enfrentarse a problemas de diseño reales pero alcanzables, lo que hizo que trabajaran muy motivados yendo más allá de lo estrictamente exigible en el curso de proyecto.

Dejaron un subproducto utilizable tanto por otros grupos de estudiantes como por docentes en trabajos de investigación. La elección de este tipo de plataformas fue un acierto, ya que al tener interfaz PCI pueden conectarse a un computador personal obteniendo una fácil solución para la entrada salida de datos con buen ancho de banda.

Aquí hay bastante camino recorrido, pero aún puede hacerse mucho. La aceleración de algoritmos implementados en hardware es un tema abierto de investigación, tanto a nivel teórico como en su realización práctica; y contar con plataformas de buena capacidad permitirá seguir experimentando en esa área.

Un tema nuevo a nivel general es la enseñanza de computación reconfigurable. Aprovechar que se dispone de hardware programable para ser utilizado en conjunto con un procesador y acelerar la ejecución de ciertas tareas. Esto implica romper con el paradigma de la ejecución secuencial de instrucciones y pasar a la utilización del paralelismo masivo que permiten los dispositivos lógicos programables. Esta tesis no ha abordado este tema directamente, pero deja abierto un camino a recorrer en este sentido.

# Agradecimientos

Fuentes de financiamiento:

- Beca de la CAP de maestría
- Proyecto Fondo Clemente Estable 6035 financió las placas PCI
- El trabajo para el curso de Diseño Lógico fue financiado por la Comisión Sectorial de Enseñanza de la Universidad de la República, Incorporación de Innovaciones en materia de Enseñanza de grado, Llamado 2002. Posteriormente se obtuvo un nuevo financiamiento para continuar el proyecto durante el año 2004.

Donaciones

- Altera University Program por sus múltiples donaciones de software y hardware, tanto para investigación como para enseñanza: 100 licencias de software para toda la Facultad, material de investigación: RIPP10, ARC-PCI, chips IIE-PCI, placas de enseñanza UP1 y UP2
- Xilinx: integrados Virtex, software ISE

A Marina Miguez y Nancy Peré por sus valiosos aportes durante la elaboración del proyecto CSE.

A Sergio Beheregaray del Taller del IIE por su colaboración en el montaje de las placas de Diseño Lógico.

Al todo el equipo de Electrónica Aplicada: Julio Pérez, Sebastián Fernández, Javier Rodríguez, Pablo Rolando, Fiorella Haim y Lyl Ciganda.

A Alvaro Giusto por las charlas sobre enseñanza y acreditación.

A Carlos Maciel por sus valiosos aportes en el enfoque de este documento.

A Gregory Randall (aunque Umberto Eco diga que es de mal gusto [38]) por la paciencia y su perseverancia en incentivarme a realizar esta tesis.

# Bibliografía


[1] Letras de las prácticas de Diseño Lógico. Visited on: Dec. 2005. [Online]. Available: http://iie.fing.edu.uy/ense/asign/dislog/#lab

[2] Manual de usuario de la placa de Diseño Lógico. Visited on: Dec. 2005. [Online]. Available: http://iie.fing.edu.uy/ense/asign/dislog/material/manual_dllab.pdf

[3] Página web del curso de Diseño Lógico. Visited on: Dic. 2005. [Online document]. Available: http://iie.fing.edu.uy/ense/asign/dislog/

[4] Página web del curso Diseño Lógico 2. Visited on: 2005. [Online]. Available: http://iie.fing.edu.uy/ense/asign/dlp/

[5] "PCI Local Bus Specification Rev 2.2," PCI Special Interest Group Dec. 1998.

[6] WISHBONE System-on-Chip Interconnection Architecture for Portable IP Cores Revision: B.3. Visited on. [Online]. Available: http://www.opencores.org/projects.cgi/web/wishbone/wishbone

[7] ABET Inc. Engineering Accreditation Commission, "Criteria for Accrediting Engineering Programs, 2006-2007 Accreditation Cycle," Baltimore, MD, USA October 29 2005.

[8] E. Ahmed and J. Rose, "The Effect of LUT and Cluster Size on Deep-Submicron FPGA Performance and Density," in *Proceedings of the 2000 ACM/SIGDA Eighth International Symposium on Field-Programmable Gate Arrays (FPGA'00)*, Monterey, CA, United States, Feb. 9-11, 2000, pp. 3-12.

[9] Altera, Delivering RISC Processors in an FPGA for $2.00. Visited on: Dec. 2005. Altera Corporation

[10] Altera, Excalibur Device Overview. Visited on: Sep. 2005. [Online]. Available: http://www.altera.com

[11] Altera, HardCopy Series Handbook, Volume 1. Visited on: Sep. 2005. [Online]. Available: http://www.altera.com

[12] Altera, MAX 3000A Programmable Logic Device Family Data Sheet. Visited on: Aug. 2005. [Online]. Altera Corporation

[13] Altera, Nios II Processor Reference Handbook. Visited on: Sep. 2005. [Online]. Available: http://www.altera.com

[14] Altera, Stratix II Device Family Data Sheet. Visited on: Sep. 2005. [Online]. Available: http://www.altera.com

[15] Altera, Stratix II Logic Structure. Visited on: Sep. 2005. [Online]. Available: http://www.altera.com/products/devices/stratix2/features/st2-features.html

[16] Altera Corporation, Max+Plus Software. Visited on: Dec. 2005. [Online]. Available: http://www.altera.com/products/software/pld/products/maxplus2/mp2-index.html

[17] Aptix, System Explorer. Reconfigurable System Prototyping for SoC Emulation. Visited on: Sep. 2005. Available: http://www.aptix.com/

[18] P. J. Ashenden, "The Designer's Guide to VHDL," Morgan Kaufmann Publishers.
[19] P. Banerjee, N. Shenoy, A. Choudhary, S. Hauck, C. Bachmann, M. Haldar, P. Joisha, A. Jones, A. Kanhare, A. Nayak, S. Periyacheri, S. Walkden, and D. Zaretsky, "A MATLAB Compiler for Distributed, Heterogeneous, Reconfigurable Computing Systems," in *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'00)*, Napa Valley, CA, United States, Apr. 17-19, 2000, pp. 39-48.
[20] L. M. Bartolo and D. R. Sadoway, "MIT's 3.091 and NSDL Materials Digital Library: Investigating the Role of Digital Libraries in Freshmen Introductory Science Courses with No Lab Component," in *APS Meeting*. Los Angeles, CA, USA, 2005.
[21] E. Bazzano, D. Alcetegaray, and F. Luongo, "Implementación Hardware de Algoritmos de Ruteo IP," Tesis de grado, Universidad de la República, Montevideo, Dic., 2004.
[22] V. Betz and J. Rose, "FPGA Routing Architecture: Segmentation and Buffering to Optimize Speed and Density," in *Proceedings of the 1999 ACM/SIGDA Seventh International Symposium on Field-Programmable Gate Arrays (FPGA'99)*, Monterey, CA, United States, Feb. 21-23, 1999, pp. 59-68.
[23] V. Betz and J. Rose, "VPR: A New Packing, Placement and Routing Tool for FPGA Research," in *Proceedings of the 7th International Workshop on Field-Programmable Logic and Applications (FPL'97)*, W. Luk, P. Y. K. Cheung, and M. Glesner, Eds., London, United Kingdom, Sept. 1-3, 1997, pp. 213-222.
[24] W. D. Bishop, "Configurable Computing for Mainstream Software Applications," Ph.D. dissertation, University of Waterloo, Waterloo, Ontario, Canada, 2003.
[25] S. Brown and J. Rose, "FPGA and CPLD Architectures: A Tutorial," *IEEE Design and Test of Computers*, vol. 13, no. 2, pp. 42-57, Summer 1996.
[26] D. Bursky, "Adaptive Logic Molds Faster, More Efficient FPGAs," *Electronic Design*, vol. 52, no. 6, pp. 48, Mar. 15 2004.
[27] W. S. Carter, K. Duong, R. H. Freeman, H.-C. Hsieh, J. Y. Ja, J. E. Mahoney, L. T. Ngo, and S. L. Sze, "A User Programmable Reconfigurable Logic Array," in *Proceedings of the IEEE 1986 Custom Integrated Circuits Conference*, Rochester, NY, United States, May 12-15, 1986, pp. 233-235.
[28] CE2004 Task Force, "Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering," The Computer Society of the Institute of Electrical and Electronics Engineers, Inc. and Association for Computing Machinery December 12 2004.
[29] Celoxica, HANDEL-C Language Overview. Visited on: Nov. 2004. [Online]. Available: http://www.celoxica.com/techlib/files/CEL-W0307171KDD-47.pdf
[30] M. Chang, Programmable Logic. Visited on: Sep. 2005. Available: http://www.ee.iastate.edu/~morris
[31] A. Chowdhary and J. P. Hayes, "General Technology Mapping for Field-Programmable Gate Arrays Based on Lookup Tables," *ACM Transactions on Design Automation of Electronic Systems*, vol. 7, no. 1, pp. 1-32, Jan. 2002.
[32] Comisión Consultiva de Expertos en Ingeniería, "MERCOSUR/RME - ACTA Nº1/2002, Dimensiones, Componentes, Criterios e Indicadores para la

Acreditación MERCOSUR," MERCOSUR EDUCATIVO, Buenos Aires, Argentina Junio 2002.
[33] Comisión Europea, Sistema europeo de transferencia y acumulación de créditos (ECTS). Visited on: September 13 2006. [Online]. Oficina de Publicaciones Oficiales de las Comunidades Europeas. Available: http://ec.europa.eu/education/programmes/socrates/ects/doc/ectskey_es.pdf
[34] K. Compton and S. Hauck, "Reconfigurable Computing: a Survey of Systems and Software," *ACM Computing Surveys (CSUR)*, vol. 34, no. 2, pp. 171-210, 2002.
[35] A. DeHon, "Reconfigurable Architectures for General-Purpose Computing," Ph.D. dissertation, Massachusetts Institute of Techonogy, 1996.
[36] B. Dipert, "EDN's Fourth Annual Programmable-Logic Directory," in *EDN*, 2004, pp. 49-67.
[37] C. N. Eastlake, "Tell me, I'll forget; show me, I'll remember; involve me, I'll understand (The tangible benefit of labs in the undergraduate curriculum)," in *Proceedings ASEE Annual Conference, ASEE*, Washington DC, 1986, p. 420.
[38] U. Eco, "Cómo se hace una tesis," 1982.
[39] L. D. Feisel and G. D. Peterson, "A Colloquy on Learning Objectives For Engineering Education Laboratories," in *American Society for Engineering Education Annual Conference & Exposition*, 2002.
[40] L. D. Feisel and A. J. Rosa, "The Role of the Laboratory in Undergraduate Engineering Education," *Journal of Engineering Education*, 2005.
[41] S. Fernández and C. Mondueri, "IIE-PCI una Plataforma de Desarrollo para el bus PCI," Tesis de grado, Universidad de la República, Montevideo, Dic., 2003.
[42] S. Fernández and C. Mondueri, IIE-PCI una Plataforma de Desarrollo para el bus PCI. Visited on: Dec. 2005. [Online]. Available: http://mondueri.com/iiepci
[43] D. Ferrer, R. Fleitas, and R. González, "NeuroFPGA: Implementación de Redes Neuronales en Lógica Reconfigurable," Tesis de grado, Universidad de la República, Montevideo, Uruguay, 2003.
[44] D. Ferrer, R. González, R. Fleitas, J. Pérez Acle, and R. Canetti, "NeuroFPGA - Implementing Artificial Neural Networks on Programmable Logic Devices," in *Design, Automation and Test in Europe Conference and Exhibition Designers' Forum (DATE'04)*, Paris, France, Feb., 2004.
[45] R. J. Francis, "Technology Mapping for Lookup.Table Based Field Programmable Gate Arrays," Ph.D. dissertation, University of Toronto, 1992.
[46] R. H. Freeman, Patent No. 4870302 (1984-1989).
[47] D. Galloway, "The Transmogrifier C Hardware Description Language and Compiler for FPGAs," in *Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines (FCCM'95)*, Napa Valley, CA, United States, Apr. 19-21, 1995, pp. 136-144.
[48] V. George, "Low Energy Field-Programmable Gate Array," Ph.D. dissertation, University of California, Berkeley, Fall, 2000.
[49] S. Giersch, L. M. Bartolo, C. S. Lowe, and A. C. Powell, Reusability in the Materials Digital Library. Visited on: September 12 2006. [Online]. Available: http://itsacast.blogspot.com/2005/10/sarah-giersch.html

[50] M. Gokhale, J. Stone, and J. Arnold, "Stream-Oriented FPGA Computing in the Streams-C High Level Language," in *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'00)*, Napa Valley, CA, United States, Apr. 17-19, 2000, pp. 49-56.
[51] D. Gómez, G. Heguy, and M. Puig, "Utilización de lenguaje C para diseño hardware," Tesis de grado, Universidad de la República, Montevideo, Uruguay, Diciembre, 2002.
[52] S. A. Guccione, "Programming Fine-Grained Reconfigurable Architectures," Ph.D. dissertation, University of Texas at Austin, 1995.
[53] S. Haas, Field Programmable Gate Arrays. Visited on: Oct. 2005. [Online]
[54] F. Haim, S. Fernández, J. Rodriguez, L. Ciganda, P. Rolando, and J. P. Oliver, "Laboratory at Home: Actual Circuit Design and Testing Experiences in Massive Digital Design Courses," in *International Conference on Engineering Education (ICEE)*, San Juan, Puerto Rico, July 23-28, 2006.
[55] M. Haldar, A. Nayak, A. Choudhary, and P. Banerjee, "A System for Synthesizing Optimized FPGA Hardware from Matlab," in *Proceedings of the 2001 International Conference on Computer Aided Design (ICCAD□01)*, 2001.
[56] M. Haldar, A. Nayak, N. Shenoy, A. Choudhary, and P. Banerjee, "FPGA Hardware Synthesis from MATLAB," in *Proceedings of Fourteenth International Conference on VLSI Design*, Bangalore, India, Jan. 3-7, 2001, pp. 299-304.
[57] J. O. Hamblen and M. D. Furman, "Rapid Prototyping of Digital Systems," Kluwer Academia Publishers, 2001.
[58] R. Hartenstein, "Reconfigurable Computing (RC) being Mainstream: Torpedoed by Education," in *International Conference on Microelectronic Systems Education (MSE2005)*, Anaheim, CA, USA, 2005.
[59] J. R. Hauser, "Augmenting a Microprocessor with Reconfigurable Hardware," Ph.D. dissertation, University of California, Berkeley, Berkeley, Fall, 2000.
[60] F. Hill and Peterson, "Digital Logic and Microprocessors."
[61] IEEE, "1076-1993 Standard VHDL Language Reference Manual." New York, NY, USA: The Institute of Electrical and Electronics Engineers, Inc., 1993.
[62] IEEE, "1076-2000 Standard VHDL Language Reference Manual." New York, NY, USA: The Institute of Electrical and Electronics Engineers, Inc., 2000.
[63] IEEE, "1076-2002 Standard VHDL Language Reference Manual." New York, NY, USA: The Institute of Electrical and Electronics Engineers, Inc., 2002.
[64] IEEE, "1364-1995 Standard Hardware Description Language Based on the Verilog Hardware Description Language." New York, NY, USA: The Institute of Electrical and Electronics Engineers, Inc., 1995.
[65] IEEE, "1364-2001 Standard Verilog® Hardware Description Language." New York, NY, USA: The Institute of Electrical and Electronics Engineers, Inc., 2001.
[66] P. Kannan and D. Bhatia, "Tightly Integrated Placement and Routing for FPGAs," in *Proceedings of the 11th International Workshop on Field-Programmable Logic and Applications (FPL'01)*, G. Brebner and R. Woods, Eds., Belfast northern Ireland, United Kingdom, Aug. 27-29, 2001, pp. 233-242.

[67] W. H. Kautz, K. N. Levitt, and A. Waksman, "Cellular Interconnection Arrays," *IEEE Transactions on Electronic Computers*, vol. C-17, no. 5, pp. 443--451, May 1968.
[68] T. Kean and J. Gray, "Configurable Hardware: Two Case Studies of Micro-Grain Computation," in *Systolic Array Processors*, J. McCanny and E. S. Jr, Eds.: Prentice Hall, 1989, pp. 310--319.
[69] T. A. Kean, "Configurable Logic: A Dynamically Programmable Cellular Architecture and its VLSI Implementation," Ph.D dissertation, University of Edinburgh, Edinburgh, January, 1989.
[70] M. Keating and P. Bricaud, "Reuse Methodology Manual For System-on-a-Chip Designs," Third ed., Norwell, Massachusetts, USA, Kluwer Academic Publishers, 2002.
[71] Lattice, ispGDX2 Family. High Performance Interfacing and Switching. Visited on: Sep. 2005. Available: http://www.latticesemi.com/
[72] L. R. Lattuca, P. T. Terenzini, and J. F. Volkwein, "Engineering Change: A Study of the Impact of EC2000, Executive Summary," ABET Inc., Baltimore, MD, USA March 2006.
[73] D. Lewis, E. Ahmed, G. Baeckler, V. Betz, M. Bourgeault, D. Cashman, D. Galloway, M. Hutton, C. Lane, A. Lee, P. Leventis, S. Marquardt, C. McClintock, K. Padalia, B. Pedersen, G. Powell, B. Ratchev, S. Reddy, J. Schleicher, K. Stevens, R. Yuan, R. Cliff, and J. Rose, "The Stratix II Logic and Routing Architecture," in *Proceedings of the 2005 ACM/SIGDA 13th International Symposium on Field- Programmable Gate Arrays (FPGA'05)*, Monterey, CA, United States, Feb. 20-22, 2005.
[74] E. Lindsay and M. Good, "Remote, Proximal and Simulated Access to Laboratory Hardware - A Pilot Study," in *EdMEDIA World Conf. Educational Multimedia, Hypermedia, Telecommunications*, Denver, CO, USA, Jun. 24-29, 2002.
[75] E. D. Lindsay and M. C. Good, "Effects of Laboratory Access Modes Upon Learning Outcomes," *IEEE Transactions on Education*, vol. 48, no. 4, pp. 619-631, Nov. 2005.
[76] A. Marquardt, V. Betz, and J. Rose, "Speed and Area Tradeoffs in Cluster-Based FPGA Architectures," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 1, pp. 84-93, Feb. 2000.
[77] B. Martín-del-Brío, A. Bono-Nuez, and P. Buisán, "Herramientas para la Enseñanza de Microcontroladores Orientadas al EEES," in *Tecnologías Aplicadas a la Enseñanza de la Electrónica*, Madrid, Spain, 12-14 July, 2005.
[78] M. I. Masud, "FPGA Routing Structures: A Novel Switch Block and Depopulated Interconnect Matrix Architectures," Ph.D. dissertation, University of British Columbia, Dec., 1999.
[79] C. Maxfield, Field-programmable devices. Visited on. [Online]. EDN Access. Available: http://www.edn.com/archives/1996/101096/default.asp
[80] L. McMurchie and C. Ebeling, "PathFinder: A Negotiation-Based Performance-Driven Router for FPGAs," in *Proceedings of the 1995 ACM/SIGDA Third International Symposium on Field-Programmable Gate Arrays (FPGA'95)*, Monterey, CA, United States, Feb. 12-14, 1995, pp. 111-117.

[81] R. C. Minnick, "A Survey of Microcellular Research," *Journal of the ACM*, vol. 14, no. 2, pp. 203--241, 1967.
[82] J. Murphy, I. Grout, J. Walsh, and T. O. Shea, "Local and Remote Laboratory User Experimentation Access using Digital Programmable Logic," in *International Journal of Online Engineering (iJOE)*, vol. 1, 2005.
[83] N. Narasimhan, V. Srinivasan, M. Vootukuru, J. Walrath, S. Govindarajan, and R. Vemuri, "Rapid Prototyping of Reconfigurable Coprocessors," in *International Conference on Application-specific Systems, Architectures and Processors*, Aug., 1996.
[84] NetLab - School of Electrical and Information Engineering at the University of South Australia, History of Remote Laboratories. Visited on: Aug. 2006. [Online]. Available: http://www.unisanet.unisa.edu.au/Resources/netlab/NetLab/NetLab%20overview/History.htm
[85] J. P. Oliver and S. Fernández, "Desarrollo de Plataformas Reconfigurables con Interfaz PCI como Proyecto de Grado," in *Southern Conference on Programmable Logic 2006*, Mar del Plata, Argentina, 2006.
[86] J. P. Oliver, A. Fonseca de Oliveira, J. Pérez Acle, and R. Canetti, "Síntesis Hardware de Redes ALN para Aplicaciones en Control," in *VIII Reunión de Trabajo en Procesamiento de la Información y Control (RPIC99)*, Mar del Plata, Argentina, 23 al 25 de setiembre, 1999, pp. 51-8 60-8.
[87] J. P. Oliver, A. Fonseca de Oliveira, J. Pérez Acle, R. J. de la Vega, and R. Canetti, "Implementation of Adaptive Logic Networks on an FPGA board," in *Proceedings of SPIE, Configurable Computing: Technology and Applications*, J. Schewel, Ed., 1998, pp. 264-273.
[88] J. P. Oliver, F. Haim, S. Fernández, J. Rodríguez, L. Ciganda, and P. Rolando, "Laboratorios en Casa: Una Nueva Alternativa para Cursos Masivos de Diseño Lógico Digital," in *Técnologías Aplicadas a la Enseñanza de la Electrónica*, Madrid, Spain, 12-14 July, 2006.
[89] J. P. Oliver, F. Haim, S. Fernandez, J. Rodríguez, and P. Rolando, "Hardware Lab at Home Possible with Ultra Low Cost Boards," in *IEEE International Conference on Microelectronic Systems Education (MSE2005)*, Anaheim, CA, USA, June 12-13, 2005.
[90] J. P. Oliver, F. Haim, S. Fernández, J. Rodríguez, and P. Rolando, "Prácticas de laboratorio no presencial en diseño electrónico digital," in *II Congreso de Enseñanza en Facultad de Ingeniería*, Montevideo, Uruguay, Oct., 2004.
[91] J. P. Oliver and J. Pérez Acle, "Utilización de FPGAs como aceleradores de cálculo," in *VII Workshop Iberchip*, Montevideo, Uruguay, 21 al 23 de marzo, 2001.
[92] S. Palnitkar, "Verilog® HDL: A Guide to Digital Design and Synthesis," Prentice Hall PTR, 2003.
[93] J. R. Paris, "Professional software in process design instruction: From why to how to beyond," in *Proceedings ASEE Annual Conference, ASEE*, Washington, DC, 1991, p. 1161.
[94] D. Pellerin and M. Holley, "Practical Design Using Programmable Logic," Englewood Cliffs, NJ, Prentice Hall, 1991.

[95] D. Pellerin and S. Thibault, "Practical FPGA Programming in C," Upper Saddle River, NJ, Prentice Hall PTR, 2005.
[96] J. Rose, R. J. Francis, D. Lewis, and P. Chow, "Architecture of Field-Programmable Gate Arrays: The Effect of Logic Block Functionality on Area Efficiency," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 5, pp. 1217-1225, Oct. 1990.
[97] J. Rose, A. E. Gamal, and A. S. Vincentelli, "Architecture of Field-Programmable Gate Arrays," *Proceedings of the IEEE*, vol. 81, no. 7, pp. 1013--1029, July 1993.
[98] A. Rubini and J. Corbet, "Linux Device Drivers," 2nd ed., Sebastopol, CA, O'Reilly & Associates, Inc., 2001.
[99] J. Saporiti, A. Villavedra, and S. Gómez, "PGVirtex: PCI Board," Tesis de grado, Universidad de la República, Montevideo, Nov., 2004.
[100] T. Shanley and D. Anderson., "PCI System Architecture," third ed., Reading, Massachussetts, Addison Wesley Publishing Company, 1995.
[101] R. Sharp, "Higher-Level Hardware Synthesis," *Lecture Notes in Computer Science*, vol. 2963, G. Goos, J. Hartmanis, and J. van Leeuwen, Eds., Berlin, Springer-Verlag, 2004.
[102] R. G. Shoup, "Programmable Cellular Logic Arrays," Ph.D. dissertation, Carnegie-Mellon University, March, 1970.
[103] SystemC, The Open SystemC Initiative. Visited on: Nov. 2004. [Online]. Available: http://www.systemc.org
[104] D. K. Tala, Verilog Tutorial. Visited on: Nov. 2004. [Online]. Available: http://www.deeps.org
[105] R. Tessier and W. Burleson, "Reconfigurable Computing for Digital Signal Processing: A Survey," *The Journal of VLSI Signal Processing*, vol. 28, no. 1, pp. 7-27, June 2001.
[106] M. Tomminska, "Application of Reprogrammability in Algorithm Acceleration," Ph.D. dissertation, Helsinki University of Technology, Espoo, Finland, 2004.
[107] N. Tredennick, "Technology and Business: Forces Driving Microprocessor Evolution," *Proceedings of the IEEE*, vol. 83, no. 12, pp. 1641-1652, Dec. 1995.
[108] J. Tuttas, K. Rütters, and B. Wagner, "Telepresent vs. Traditional Learning Environments – A Field Study," in *International Conference on Engineering Education*, Valencia, Spain, July 22-26, 2003.
[109] S. Wahlstrom, "Programmable Logic Arrays. Cheaper by the Millions," *Electronics*, vol. 40, no. 25, pp. 90-95, Dec. 1967.
[110] J. F. Wakerly, "Digital Design Principles and Practices," Third ed., Prentice-Hall, 2000.
[111] P. C. Wankat and F. S. Oreovicz, "Teaching Engineering," New York, McGraw-Hill, 1993.
[112] W. Wolf, "A Decade of Hardware/Software Codesign," *Computer*, vol. 36, no. 4, pp. 38-43, April 2003.
[113] Xilinx, PowerPC 405 Processor Block Reference Guide. Visited on: Sep. 2005. [Online]. Available: http://www.xilinx.com

[114] Xilinx, Virtex-4 Data Sheet:DC and Switching Characteristics. Visited on: Aug. 2005. [Online]. Available: http://www.xilinx.com/
[115] Xilinx, Virtex-4 Family Overview. Visited on: Aug. 2005. [Online]. Available: http://www.xilinx.com/bvdocs/publications/ds112.pdf
[116] Xilinx, Virtex-4 User Guide. Visited on: Oct. 2005. [Online]. Available: http://www.xilinx.com/bvdocs/userguides/ug070.pdf
[117] Xilinx, Virtex-II Pro and Virtex-II Pro X Platform FPGAs: Complete Data Sheet. Visited on: Aug. 2005. [Online Product Specification]. Available: http://www.xilinx.com
[118] Xilinx, Virtex-II™ Platform FPGAs: Complete Data Sheet. Visited on: May 2004. Available: http://www.xilinx.com
[119] Xilinx, Virtex™-E 1.8 V Field Programmable Gate Arrays. Visited on. Available: http://www.xilinx.com/bvdocs/publications/ds022.pdf
[120] Xilinx, Xilinx EasyPath Solutions. FPGAs Below Structured ASIC Prices. Visited on: Aug. 2005. [Online]. Available: http://www.xilinx.com/
[121] Xilinx, Xilinx Introduces Next-Generation EasyPath FPGAs. Visited on: Aug. 2005. [Online]. Available: http://www.xilinx.com/

# Glosario

| | |
|---|---|
| ARC-PCI | Placa reconfigurable donada por la empresa Altera |
| ACEX | Familia de FPGAs de la empresa Altera |
| AHDL | Altera Hardware Description Language |
| ASIC | Application Specific Integrated Circuit |
| BAR | Base Address Register |
| BIOS | Basic Input/Output System |
| BGA | Ball Grid Array |
| CAD | Computer-Aided Design |
| CAE | Computer-Aided Engineering |
| CAN | Controller Area Network |
| CPLD | Complex Programmable Logic Device |
| DSP | Digital Signal Processor |
| EDA | Electronic Design Automation |
| FBGA | Fine Ball Grid Array |
| FLEX | Familia de FPGAs de la empresa Altera |
| FPGA | Field Programmable Gate Array |
| Hardcore | Circuito prediseñado implementado directamente en silicio, el circuito no es programable por el usuario. |
| HDL | Hardware Description Language |
| IIE | Instituto de Ingeniería Eléctrica de la Facultad de Ingeniería, Universidad de la República |
| IIE-PCI | Primer placa reconfigurable realizada en el IIE |
| IP | Internet Protocol |
| IP core | Intellectual Property core, circuito prediseñado que se comercializa en forma similar a un paquete de software |
| ISA | Industry Standard Architecture |
| JTAG | Joint Test Action Group - Industry Standard — IEEE Std 1149.1-1990 |
| Max+Plus II | Software de diseño de Altera |

| | |
|---|---|
| OpenCores | http://www.opencores.org |
| PC | Personal Computer |
| PCI | Local Bus Peripheral Component Interconnect |
| PE | Process Element |
| PGVirtex | Segunda placa reconfigurable realizada en el IIE |
| PLL | Phase-Locked Loop |
| PLD | Programmable Logic Device |
| PROM | Programmable Read-Only Memory |
| RAM | Random Access Memory |
| SDRAM | Synchronous Dynamic RAM |
| SEU | Single Event Upset |
| SoC | System-on-Chip |
| Softcore | Circuito prediseñado que se comercializa en forma similar a un paquete de software, y es implementado en las celdas lógicas de un dispositivo programable. También llamado Soft IP core. |
| Thesic | Plataforma de test de circuitos desarrollada en TIMA, INPG, Grenoble, Francia |
| TQFP | Thin Quad Flat Pack |
| USB | Universal Serial Bus |
| Verilog | Lenguaje de especificación hardware |
| VHDL | Lenguaje de especificación hardware VHSIC (Very-high-speed Integrated Circuit) Hardware Description Language |
| Virtex | Familia de FPGAs de la empresa Xilinx |
| Wishbone | System-on-Chip (SoC) Interconnect Architecture for Portable IP Cores |

# Apéndice

Encuesta del curso de Diseño Lógico realizada por la Unidad de Enseñanza

UNIDAD DE ENSEÑANZA
FACULTAD DE INGENIERIA

# Informe Encuesta Diseño Lógico

## De la modalidad semipresencial del laboratorio:

**Pregunta 1**

**¿Cuál es tu opinión respecto a la modalidad semipresencial del laboratorio?**

La amplia mayoría de los estudiantes contestaron que tienen una buena (y muy buena) opinión sobre esta forma de desarrollar el laboratorio. Que esta es una forma que les permite tener una gran práctica al manipular la placa y usar el programa libremente.

Algunas frases de las encuestas:

- "Me parece buena ya que nos fuerza a adquirir los conocimientos necesarios por nuestros propios medios."
- "Flexible, más entretenida."
- "Muy buena, interesante, motivadora."
- "Fue excelente, no tuvo desperdicios."
- "Está bueno, así se hacen valer los 12 créditos de la materia (hacemos algo)."
- "Muy buena, ayuda a aprender probando."
- "El laboratorio me pareció excelente."
- "Interesante y motivadora."
- "Me gustó, es una forma de pedir un trabajo sin obligar a nadie a venir en un horario inconveniente"
- "Es buena, aporta mucho."
- "Pienso que es mejor que los laboratorios tradicionales ya que se tiene más tiempo para realizar y comprender bien la práctica."
- "Me parece positiva, obliga a un mayor esfuerzo."
- "Muy cómoda."
- "creo que es bueno, uno tiene tiempo para poder examinar mejor la placa, programa."
- "Ayuda a llevar al día el curso. Nos ponemos en contacto con la herramienta"
- "No estoy de acuerdo"
- "Un avance respecto al anterior"
- "Me parece bien porque nos da una instancia para 'jugar' con el programa y con la placa, en este caso, viendo aplicaciones reales con sus resultados, en vez de tanta teoría."
- "Buena, permitía que pudiéramos trabajar."
- "Esta bueno porque da la oportunidad de discusión entre los compañeros sin limitación de tiempos."
- "Le da un toque de responsabilidad a los estudiantes."
- "..se aprende mucho."
- "muy buena y didáctica"
- "Buena forma de trabajo."

- "complica bastante a los grupos que son formados por gente que no se conoce, pero son los menos."
- "Excelente, realizar tareas en mi casa fue motivante."
- "Me parece una muy buena idea. Otras materias deberían adoptarla. Las materias más vinculadas con la parte práctica del Ingeniero deberían de ser aprobadas en su totalidad por laboratorios."
- "...es agradable poder probar y equivocarte sin la presión de un docente al lado."
- "...permite sumar puntaje y obliga a los estudiantes a estar más o menos al día con el curso."
- "Creo que está bien. Cuánto antes nos tomen pruebas orales, mejor."

**Pregunta 2**

**¿Qué beneficios y qué inconvenientes puedes identificar en relación a la organización del laboratorio en esta modalidad?**

**Beneficios**: los principales beneficios que se plantearon fueron por el lado de un manejo más libre de los tiempos, de una mayor tranquilidad para preparar las prácticas y llegar a las defensas con más seguridad y conocimiento del programa y la placa. Se refleja una sensación de que se aprende mucho haciendo las prácticas, al igual que un nivel más alto de motivación. Otro punto que se destaca es que las prácticas hacen que no se atrasen en el curso, es decir que tienen que seguir el desarrollo de la asignatura para poder realizarlas.

- "Favorece a que cada uno pueda aprovechar el tiempo de mejor manera"
- "Contacto directo con la herramienta que usaremos"
- "es mejor porque se trabaja más en domicilio y se aprende más que antes."
- "incentiva a estudiar la materia."
- "Ningún inconveniente, es buena esta forma de organización y hace llevadera la materia."
- "Ninguno, aparte de ver como actúa un circuito en la realidad."
- "Motiva a estudiar más."
- "Rinde más el trabajo."
- "El poder manipular objetos que de otra forma no sabría cuando lo iba a hacer."
- "Ganamos experiencia y perdemos miedos, cuando nos acostumbramos a utilizar la placa (miedo me refiero a romperla)."
- "reafirma conocimientos teóricos."
- "Entiendo que si hay mucha gente no se puede implementar, pero es muy estimulante trabajar con chips y hacer diseños de circuito."
- "se llega con el material pronto."
- "Aprendés mucho con los laboratorios."
- "Incentiva a estudiar la materia."
- "Dispongo de mi tiempo."
- "Nos obliga a todos a trabajar en el laboratorio para la defensa."
- "La evaluación es justa."

- "Lo bueno es que te hace seguir por lo menos un poquito la materia para poder hacer y defender cada una de las tres prácticas."
- "Practicidad en una carrera muy teórica."
- "..aprendemos en la práctica lo que nos enseñan en el teórico."
- "Ayuda a estar al día."
- "Permite acercarnos más a nuestra vocación ya que se necesitan más tareas de este tipo en la carrera." "Más interesante."
- "Te hace estudiar y entender más los conceptos."
- "Beneficios de la mitad presencial: que se aprende de los profesores. De la mitad no presencial: que se puede experimentar solos."
- "Los beneficios son que se puede aprender mejor que en un laboratorio a distancia pues hay un seguimiento del curso."

**Inconvenientes:** el principal inconveniente que se plantea es que las prácticas llevan más tiempo, lo que resta tiempo a la realización de ejercicios de práctico o a otras asignaturas. Además se presenta como un problema el que no se puedan realizar las prácticas en las salas de máquinas de Facultad (115 y 501).

- "Hay que ajustar los tiempos de estudio demasiado."
- "Requiere dejar de hacer práctico"
- "Lleva más tiempo que antes (aunque se aprende más)."
- "Juegan mucho los nervios en esta modalidad, pero te motivan a perfeccionarte."
- "Deja poco tiempo para otras materias."
- "Inconveniente: carga horaria."
- "Inconveniente: tiempo de preparación."
- "La imposibilidad de programar las placas en los salones 115 y 501 y un horario de consulta semanal determinado."
- "No saber que pasa si hay un error."
- "Escaso tiempo entre presentación de la letra y la defensa de la misma , teniendo en cuenta la coordinación entre grupos."
- "Falta un poco más de guía en cuanto a detalles técnicos."
- "Deberían dejar utilizar las máquinas del laboratorio a aquellas personas que solo necesiten configurar la placa, los demás al 501."
- "Exige cierta disponibilidad horaria."
- "En la facultad solo se podía programar la placa en el instituto lo cual ocasionaba algunos inconvenientes de horarios para lo que no tenemos computadoras."
- "si te trancas y no te sale nada puede ser que no sepas para donde arrancar => consultas"
- "Es muy importante acompañarla con muchas consultas, lo único que puede ser malo quedarse con algunas dudas y perder tiempo aprendiendo a usar el MaxPlus."
- "Ninguna porque si alguien no hace nada, esta la evaluación oral del trabajo."
- "Lleva tiempo y no todo el mundo lo dispone."

- "Que los alumnos deben resolver los problemas solos con los conceptos recién dados en el teórico."
- "Hay que tener PC adecuado y consulta a docentes en algunos momentos no se logra (p.e. fin de semana)."

**Pregunta 3**

**¿Incluirías alguna modificación en la forma de organizar el laboratorio dentro de esta modalidad semipresencial?**

La gran mayoría dijo que no haría modificaciones a la forma propuesta del laboratorio. En los casos en que propusieron alguna, las siguientes frases son las más significativas:

- "se debería aclarar más que tipo de preguntas nos pueden hacer. Quizás los haría más cortos e incluiría uno de modo a nivel."
- "Que se coordine un profesor para cada grupo para consultas sobre el laboratorio."
- "Tal vez un poco más de tiempo entre la publicación de la letra y la fecha de la práctica."
- "Entregaría la letra con un poco más de anticipación."
- "Mantendría todo como está, salvo la desproporción en las calificaciones laboratorio-parcial."
- "Instructivo más específico acerca del Altera, sobre todo en la parte de simulaciones."
- "El porcentaje que debemos obtener en el parcial es demasiado para el trabajo que lleva el laboratorio. Con 25ptos. en el laboratorio => Exonerar con 50% del parcial."
- "Más consultas y guía sobre el programa y el problema."
- "Tener más máquinas en facultad para trabajar."

UNIDAD DE ENSEÑANZA
FACULTAD DE INGENIERIA

## De la presentación:

**Pregunta 1**

**¿Cuál es tu opinión respecto a los recursos informáticos de apoyo utilizados (sitio web, lista, material, etc.)?**

Hay distintas opiniones sobre los materias del curso. La mayoría dicen que están bien. Las mayores observaciones son con respecto al sitio web de la asignatura. El segundo lugar lo tiene la lista de correo que algunos preferirían que fuese un news o foro.

- "Muy buenos"
- "Buenos"
- "Alcanzan"
- "Aceptable"
- "Más soluciones"
- "Excelentes"
- "Útiles"
- "Solo utilicé la página para información general"
- "Alcanzaba, pero podría dedicarse alguna clase para presentar el programa, material, etc."
- "El material es muy bueno, pero me ayudaría mucho tener ejercicios con soluciones. La lista ayuda"
- "El material podría mejorarse (+parciales y exámenes) y el resto también"
- "No los usé mucho, pero la web está bien, tiene todo lo necesario. Sobre la lista tiene que haber más control."
- "Precisaríamos un solo material (no mirar 4 libros para la materia) y me parece que es mejor las NEWS de noticias, ya que si hay algún problema es más fácil consultarlas mismo desde la FACULTAD"
- "Deberían poner en la web, más exámenes con soluciones más convincentes, ya que, por ejemplo, la mayoría tienen el diagrama de estados hecho mínimo 'al toque'; podrían hacerlo un poco más grande para saber como lo pensaron y después minimizarlo"
- "Faltaría un manual con todas las herramientas necesarias del programa"
- "Estaría bueno que hubiera material teórico más accesible"
- "El sitio web debería incluir ejemplos de esta modalidad. La lista de consultas es imprescindible"
- "Estaría bueno que se publicaran soluciones de ejercicios de practico en la web. Así como (esto indispensable) de exámenes"
- "Tendría que haber más información sobre como usar el MaxPlus III"
- "Buenos, pero agregaría el material que dan en teórico a la página ya que esto ayudaría a los que no podemos venir al teórico"
- "Deberían poner news y eliminar la lista de correo"
- "Perfectos a lo sumo, que podrían poner parciales y exámenes en web (gracias)"
- "Muy buena, todo está bien organizado y a tiempo"
- "Información insuficiente"

- "Muy buenos y prácticos"
- "Muy buena, toda la información necesaria para llevar a cabo las prácticas se encontraba allí"
- "No muy buenos, debido a que había por ej. errores en distintos lugares (compilación) que no se especificaban en ningún lado"
- "Podrían haber más evaluaciones colgadas y un letrero grande de novedades en general"
- "Buena, lo único que falta son más evaluaciones (parciales y exámenes) y material teórico, el libro no sirve, sería bueno un repartido con lo que se usa en el curso"
- "Es suficiente para poder realizar las prácticas"
- "Quizás alguna clase de consulta además de la que se contaba"
- "Están bien organizados"
- "La lista: OK, poco concurrida por los estudiantes.
- Material: sería bueno poder identificar entre parciales y exámenes anteriores.
- Web: ¡no hay ni un parcial ni un examen! ¡Muy poco útil!"
- "Tenemos poco conocimiento del manejo del Max Plus II"
- "Es buena, no hace falta más recursos"
- "Creo que estuvieron bien. Quizás se podría agregar un foro donde quedaran registradas respuestas a dudas sobre el Max Plus"
- "Actualicen la página más seguido"
- "La lista es una buena idea, no se uso mucho. Tendrían que poner evaluaciones viejas"
- "Los recursos son muy pobres"
- "Buenos, lástima que solo se pueda programar la placa en el laboratorio de Dis. Lógico, porque acota posibilidades"
- "Sería bueno que pusieran las noticias también en la página web"
- "Prefiero el newsgroup en lugar de lista. Falta material de exámenes y parciales en la web"
- "faltaría un tutorial más extenso de MaxPlus"
- "Sería bueno que pongan las soluciones de los prácticos ya que al no poder hacer todos en clase poder ver cual es la solución correcta y en caso de no ser la de uno ir a clase de consulta y entender el error"
- "Me pareció bueno, quizás más bibliografía o material sobre Max Plus"
- "Bueno, lo único malo es que no se pueden usar las placas en el 501 y 115"
- "Las máquinas del laboratorio son carretas, para presentar una práctica están bien, pero para trabajar se dificultan las cosas"
- "Web -> Poco actualizado. Lista -> útil, pero sería mejor un newsgroup"
- "Mayormente de acuerdo con ellos"
- "No la usé mucho, pero cualquier apoyo extra es bienvenido"
- "Son muy útiles, en especial la lista"
- "Para el laboratorio los recursos fueron buenos, aunque el libro del curso no era bueno"
- "Deberían de hacer más clases de consulta previas"
- "Son de gran utilidad"

- "Falta material teórico, no pude estar en las últimas clases teóricas y me cuesta obtener el material"
- "Sitio web -> pocos cambios, no variaba mucho. Material -> 10 puntos. Lista? "
- "El sitio web no se actualizaba muy seguido (cartelera)"
- "Creo que fueron muy buenos, el foro sería mejor tenerlo, no por lista"
- "La placa fue buena. El altera nos daba problemas algunas veces"
- "Están muy bien la página y la lista. Respecto a la bibliografía me parece que ninguno de los libros es muy claro"
- "El programa MAX-PLUS es muy complicado para la gente que lo ve por primera vez"
- "Buena. El inconveniente por falta de lugar y PC's en cantidad, lugar y tiempo suficientes"

**Pregunta 2**

**¿Cuánto tiempo en horas promedio estimadas te llevó realizar las prácticas de laboratorio?**

Los estudiantes dedicaron un promedio de 16hs a cada práctica.

**Pregunta 2.1**

**¿Qué porcentaje de ese tiempo trabajaste solo?**

- En promedio los estudiantes trabajan un 20% del tiempo solo.
- Hay un 25% de estudiantes que no dedicaron tiempo solos a las prácticas sino que realizaron toda la práctica en grupo.

**Pregunta 3**

**¿Incluirías alguna modificación? ¿Cuál?**

La mayoría de los estudiantes dicen que no harían modificaciones en la presentación. Los cambios propuestos son los siguientes:

- "Menos RTL"
- "Que sean solo dos prácticas"
- "Si, más tiempo"
- "Más información sobre programa"
- "No, me parecieron muy buenos los laboratorios"
- "Sería bueno tener un parcial más"
- "Mala idea tener los labs los lunes"
- "Que en la presentación nos dieran más tiempo para responder las preguntas, de forma de no presionarnos llevándonos a cometer errores (cosa que me paso personalmente con un tema que yo sabía, pero con los nervios respondí mal perdiendo puntos)"
- "Dar más tiempo para realizar las prácticas. Alguna clase más de consulta"
- "Podrían comprimir la primera parte del curso (aburrida), para poner un laboratorio de modo nivel, se aprende mucho con los laboratorios (se van dudas)"

- “Que fuera el mismo profesor el responsable de evaluar a un grupo en las diferentes etapas”
- “Que se prevean los posibles errores por compilar la placa, por ejemplo”
- “No, me pareció muy bien hecho”
- “Que las prácticas se den con más anticipación para tener más tiempo para prepararlas”
- “Que los laboratorios no fueran tan extensos, ya que así se le quita mucho tiempo de estudio a otras materias”
- “Si, dar un poco más de tiempo desde que dan la letra hasta que se debe presentar”
- “Que las preguntas a la entrega fueran escritas, a todos los del grupo por igual”
- “Que nos dieran alguna instrucción en el manejo del MaxPlus II”
- “Más laboratorios, parciales escritos para cada uno y menos peso del parcial final. Con las reglas de hoy, el parcial final es casi un exámen, no hay incentivo desde el punto de vista del puntaje en los laboratorios”
- “Algunas cosas con respecto al diseño con el Max Plus. Por ejemplo, para la práctica 2 no pude entender que un bus en un FF-D es como un LATCH”
- “Daría más tiempo para preparar las entregas y más clases de consulta”
- “A la presentación no, agregaría si se pudiese (algo muy difícil) aunque sea una clase de manejo del programa”
- “Que al contestar los docentes no presionen, cada uno tiene sus propios tiempos y no por eso se sabe menos. La falta de criterio en la evaluación, y no hay forma de discutirle al docente la nota que te ponen. A veces me pareció que existía prejuicio, el docente pensaría que uno no hizo nada estando muy equivocado. Eso desalienta mucho, por más que uno trabaje e hizo todo igual no te pone la nota correspondiente, te pone menos.”
- “Exonerar con 50 o 45% con el máximo en laboratorios”
- “Más ayuda por parte de los docentes”
- “Pasos intermedios de cada práctica para no tener un solo contacto con los docentes solo al presentar (final)”

UNIDAD DE ENSEÑANZA
FACULTAD DE INGENIERIA

## De la Gestión:

**Pregunta 1**

**¿Cuál es tu opinión sobre la manera en que fue gestionado el laboratorio?**

La amplia mayoría de los que contestaron esta pregunta destacan que fue buena la gestión de las tareas de laboratorio. Algunas de las opiniones son las siguientes.

- "Perfecta"
- "Estuvo bien, salvo que no lleva tanta nota"
- "Buena, sobre todo por la dedicación de los docentes para con las problemáticas que se nos plantearon durante la tarea"
- "Creo que fue productivo y que mediante las prácticas uno aclara mucho más los conceptos"
- "Ordenada y simple"
- "En la última práctica hubo falta de tiempo"
- "En las evaluaciones se hicieron preguntas que el estudiante no tenía herramientas para responder"
- "Creo que ha sido la mejor manera de implementar un laboratorio desde que entré a la FING"
- "Regular, la nota de la defensa no fue representativa (mi compañero respondió poco y nada y tuvo la misma calificación)"
- "Faltaron disponibilidad de recursos"
- "Faltó un poquito de difusión de las tareas (no se sabía cuando estaban las letras)"
- "Mala (al principio), me enteré un día antes de la primera entrega con que grupo me tocaba"
- "se asignaba una hora de práctica a cada grupo y muchas veces no se cumplía"

**Pregunta 2**

**¿Cuál es tu opinión sobre el modo en que se desarrolló la comunicación entre alumnos y docentes, y entre alumnos entre sí?**

En general los estudiantes destacan la buena comunicación con los docentes, mientras que la comunicación entre alumnos la restringen al grupo de laboratorio.

- "Buena"
- "Agradable"
- "Bien, fluida"
- "Estoy conforme con la relación con los docentes. Nos atendió impecable en todo momento"
- "Buena, excepto los fines de semana cuando la presentación es el lunes"
- "Solo tuve contacto con los docentes, con el resto de los alumnos no discutimos sobre nada"
- "No tan buena"

- "Al ser un grupo no muy grande hubo una buena comunicación y los docentes siempre estaban accesibles"
- "Alumnos y docentes, buena. Alumnos entre sí, poca comunicación (solo la del grupo)"
- "Bien, podría haber más clases de consulta"
- "Buena, principalmente a través de la lista de correo"
- "Solo me comuniqué vía mail ya que no puedo asistir a las clases"
- "Hubo una unión entre los grupos que no se podría haber dado sin los laboratorios"
- "Con los profesores no fue tan buena, salvo en el momento mismo de la práctica"

**Pregunta 3**

**¿Incluirías alguna modificación respecto a las tutorías y/o a los recursos comunicacionales y a su forma de gestión? ¿Cuál?**

La mayoría de los estudiantes no propusieron modificaciones a la gestión del curso. Una opinión que se repitió aquí fue la solicitud de clases de consulta para los laboratorios. Algunas de las opiniones son las siguientes:

- "Clases de consulta en horario de la noche (20hs aprox.)"
- "Algunos cursillos de MaxPlus, placa, etc."
- "Publicaría las letras con más anticipación a la fecha del laboratorio"
- "Si, que los fines de semana algún docente revisara las consultas por la web"
- "Más clases de consulta"
- "News de noticias"
- "Habría que repartir mejor el tiempo según la dificultad de las prácticas (tendrían que dar más tiempo para la última)"
- "Tener asignado un docente c/ algunos estudiantes"
- "Alguna clase de consulta previa a las entregas"
- "Las consultas en la página y no por correo"
- "Si, le daría más tiempo a la realización de las prácticas ya que nos exigía una dedicación muy grande en tiempo haciendo que descuidemos otras materias"
- "Los newsgroups, más clases de consulta -> las que hubieron fueron masivas"

**Pregunta 4**

**¿Te sentiste apoyado por el equipo docente?**

La respuesta a esta pregunta fue un sí en la gran mayoría de los casos, y algunos "más o menos" o casos puntuales.

- "Si"
- "En todo momento"
- "Poco tiempo de consulta con ellos (personal) para dudas sobre la práctica"
- "Podrían haber habido más orden en clases de consulta (a veces no había nadie a quien preguntarle)"

- “En algunas consultas no”
- “Sí, realmente sí, pero dentro de los horarios en que encontramos a los docentes disponibles”
- “Sí, la verdad es que no tengo quejas”
- “Por lo general sí, aunque cuando íbamos a programar la placa por primera vez y pedimos ayuda, no ayudaron nada”
- “Algunas de las propuestas no eran muy concretas. Ej. en el tercer lab. no fueron muy claros en la parte de la modificación del contador. Parte 3”
- “Sí porque tratan de ayudar todo lo que pueden”

UNIDAD DE ENSEÑANZA
FACULTAD DE INGENIERIA

## De lo Metodológico:

Cuál es tu opinión respecto a:
**Pregunta 1**
**¿la metodología de trabajo grupal?**

Casi todos los estudiantes evalúan como muy positiva la metodología de trabajo en grupo. Destacan claramente los beneficios e incluso llegan a calificarla de "necesaria".

- "No me gustan; se pierde mucho tiempo en los trabajos, pero es bueno porque se intercambian ideas diferentes"
- "Adecuada" "Ayuda" "Muy útil" "Es necesaria"
- "Está bueno que lo impongan porque se aprende mucho"
- "Me parece muy productivo porque las dudas que tiene alguno las puede disipar otro"
- "Me simplificó mucho las cosas"
- "Conociendo a la gente del grupo es fácil. Si no tenes compañeros puede tocarte un grupo que no trabaje bien"
- "Estuvo buena, porque da pie a discutir con el grupo"
- "Sirve para poder discutir y eso ayuda a aprender y entender mejor las cosas"
- "Buena: te hace conocer gente nueva y más cabezas piensan mejor que una sola"
- "Buena, favorece en muchos aspectos"
- "Hace más llevadera la materia"
- "Es muy buena, prefiero trabajos grupales"
- "Tuvimos una compañera que abandonó al comienzo, pero igual de a 2 pudimos hacer las prácticas"
- "No todos aprenden"
- "Me parece muy buena y que debería aplicarse más en otras materias"
- "Cuando es de esta forma es muy bueno porque no hay presiones sino que es tu propio interés por la materia"
- "Sirve para aclarar problemas que uno solo no ve"

**Pregunta 2**
**¿la forma de evaluación?**

Una gran mayoría de los estudiantes están de acuerdo con la forma de evaluación. Las únicas observaciones que se encuentran apuntan a tener dos parciales o a que el laboratorio tenga más peso en el puntaje total. Algunas de las expresiones se transcriben abajo.

- "Justa"
- "Ideal"
- "Bastante equilibrada"
- "Mala por el horario"

- "A mi me gusta esta forma de evaluación"
- "Buena. El puntaje en el curso podría ser más de 25/100, por ejemplo 35/100"
- "Estaría mejor con 2 parciales"
- "La evaluación de la parte oral no me pareció muy buena ya que las preguntas no se entendían del todo bien"
- "Depende de los profesores, algunos te evalúan bien, o sea te hacen pensar lo que estas diciendo"
- "Está bien aunque para el parcial se olvida un poco la primera parte del curso"

**Pregunta 3**

**¿Incluirías alguna modificación? ¿Cuál?**

Se presentaron pocas propuestas de modificaciones, entre ellas la más mencionada fue el pedido por mantener el formato anterior de dos parciales, además de la nueva forma de laboratorio.

- "El horario"
- "Mayor peso en la nota total"
- "Sería mejor si se pudiera trabajar en Facultad, ya que hasta la mitad del tiempo de la primer tarea ni siquiera teníamos el programa"
- "La primera parte cuando se ven códigos, nros. en compl. a 2, etc. hasta que se comienza a ver mapas K, es demasiado larga y tediosa"
- "2 parciales"
- "Tener un primer parcial independientemente de tener o no los laboratorios"
- "Dar clases opcionales (dentro del laboratorio) para aprender a manejar el programa que se usa en el curso (El tutorial en la web no es claro)."
- "Falta más 'Acción' en los laboratorios"
- "Más puntos"
- "Preguntas orales más concretas"

UNIDAD DE ENSEÑANZA
FACULTAD DE INGENIERIA

## Percepción General:

Los encuestados lo perciben como un curso muy práctico, y la frase que se presentó más fue “es el primer curso práctico de verdad”.

- “Es divertido!”
- “Me gustó, fue bueno trabajar con la plaqueta”
- “Lo malo es la primera parte, esa parte me parece que le dan mucho tiempo y no sirve tanto, además es más fácil, se podrían dar repartidos y dejar que cada uno los estudie, dar un poco en clase y después dedicarse a la segunda parte que sí importa. No es proporcional el tiempo dedicado a la dificultad, la segunda parte es muy difícil y se da en poco tiempo”
- “Muy buen curso, con buenos docentes”
- “Muy buena, aprendí lo más importante del curso sin llevar al día los prácticos”
- “Muy bueno, aunque habría que darle más uso al Max Plus, en los prácticos por ej.”
- “Hasta ahora es el mejor curso de la carrera ya que se ven problemas reales y no tanta teoría física o matemática”
- “Hasta ahora ha sido la materia más ‘linda’ que he cursado, muy bien dictada por los docentes”
- “Es muy abstracto el pasaje de la letra de los problemas al ejercicio”
- “Es el primer curso en que nos enseñan algo más o menos aplicable”
- “Me pareció mejor que en los años anteriores”
- “Interesante”
- “Bastante buena”
- “Ágil, interesante”
- “Que se ha modificado para mejor”
- “Me gustó mucho, me interesó lo tratado y tuvo un muy buen desarrollo en general. Esta metodología creo que es buena no solo para D. Lógico sino como metodología”
- “El mejor curso que hice en todos estos años”
- “Muy bien dado”
- “Es un buen curso, sobre todo en lo teórico, ya que lo dictado en clase era muy buen material”
- “Mucho más interesante que en años anteriores”
- “Fue el curso más práctico que tuvimos hasta el momento”
- “Sinceramente muy bueno (aunque soy estudiante de Computación)”
- “Muy buena, aunque es la única materia del IIE que cursé y estoy acostumbrado a cursos del INCO que en general son malos”
- “Muy buena, creo que mejoró sustancialmente respecto a la anterior forma del curso”
- “Bien o Muy Bien, interesante, ahora se aprendió más con los lab. (yo recurso)”
- “Es un buen curso, el teórico podría ser más completo. El material no sirve para nada”

- “Un curso humano. Se puede trabajar y estudiar”
- “Hay ciertas preguntas que no las entiendo pero como comentario general puedo decir que fue una materia que me gustó mucho y esta bien organizado el curso”

UNIDAD DE ENSEÑANZA
FACULTAD DE INGENIERIA

## Comentarios Generales:

- “Estoy conforme con el curso”
- “No asusten tanto con las placas”
- “Sé que hay poco tiempo pero no se olviden de las consultas para ejercicios, el año pasado servían mucho”.
- “Queda por mejorar, pero lo importante es que se cambió es espíritu de dar clase a lo importante es aprender, no evaluar con rigor”

Algunas conclusiones:

- Dada la forma en que se realizó la encuesta se dio un fenómeno raro: 1/5 de los estudiantes no contestaron la segunda carilla de la encuesta.
- Algunos estudiantes **expresaron** no entender las preguntas.