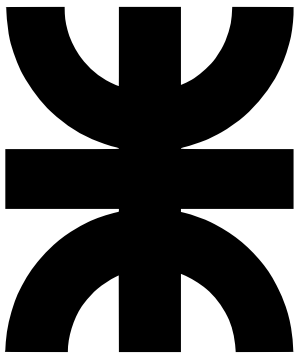
Plataforma de Hardware Reconfigurable



Luis Alberto Guanuco

Departamento de Ingeniería Electrónica

Universidad Tecnológica Nacional – Facultad Regional Córdoba

Este trabajo otorga el grado de

Ingeniero Electrónico

Cátedra Proyecto Final 2011

A toda mi familia y quienes aportaron desinteresadamente en mi crecimiento personal.

Agreadecimientos

Se agradece principalmente a la comunidad de *Hardware y Software Libre* ya que sin el aporte cooperativo y desinteresado de ellos sería imposible acceder a un sin fin de herramientas informáticas que son de gran importancia en el desarrollo territorial de cualquier Comunidad. También a los investigadores del *Centro Universitario de Desarrollo en Automoción y Robótica (CUDAR)*, quienes con su basta experiencia han brindando soporte técnico de gran manera el desarrollo del proyecto. Agradecemos el apoyo recibido por el *Departamento de Ingeniería Electrónica* de la *Universidad Tecnológica Nacional – Facultad Regional Córdoba*. Como así también a el *Laboratorio de Técnicas Digitales e Informática (LTDI)* por participar del proyecto, ofreciendo sus recursos humanos y físicos para la recepción del proyecto. A la *Agencia para el Desarrollo Económico de la ciudad de Córdoba (ADEC)* quienes nos han permitido participar de su programa *Córdoba Innovadora - Desarrollo Territorial en el Área Metropolitana de Córdoba*. En donde se ha podido adquirir experiencia en el manejo de proyecto y sobre todo la posibilidad de solventar económicamente el proyecto para su transferencia al LTDI. Y por último agradecer a todas las personas que en el transcurso del desarrollo han participado institucionalmente como así también en forma desinteresada.

Resumen

El proyecto denominado *Plataforma de Hardware Reconfigurable*, ofrecer los recursos de *hardware* y *software* necesarios en el diseño de sistemas digitales reconfigurables de alta complejidad. El objetivo principal del proyecto es su implementación en el ámbito educativo, pero su aplicación puede extenderse a la industria ya que posee flexibilidad y recursos variados para la generación de diseños prototipos.
El desarrollo se encuentra publicado bajo condiciones legales que permite el *uso* y *modificación* de todo el proyecto por cualquier diseñador que así lo desee, lo que asegura la total *libertad* a quienes se estén formando en el diseño de sistemas electrónicos digitales. La Plataforma de *Hardware* reconfigurable ofrece un desarrollo alternativo en las plataformas digitales comerciales ya que se encuentre adaptada a las necesidades requeridas por el ámbito educativo regional. En el desarrollo han participado activamente docentes e investigadores, quienes aportaron información que en parte han definido el perfil técnico y académico del proyecto.

Contenidos

Lista de Figuras

Lista de Tablas

Nomenclatura

Acrónimos / Abreviaturas

3D-IC *Three-Dimensional Integrated Circuit*

CLB *Configurable Logic Block*

CPLD *Complex Programmable Logic Device*

DLL *Delay-Locked Loop*

DSP *Digital Signal Processor*

EEPROM *Electrically Erasable Programmable Read-Only Memory*

F2F *Face-to-face*, en microelectrónica, F2F representa la disposición "cara a cara" de las uniones entre capas del diseño en la tecnología 3D-IC

FF flip-flop, circuito que tiene dos estados estables y puede ser usado para almacenar información

FPGA *Field Programmable Gate Array*

GNU *GNU's Not Unix*

HDL *Hardware Description Language*

IC *Integrated Circuit*

JTAG *Joint Test Action Group*

MSI *Medium Scale Integration*

PALCE *PAL CMOS Electrically erasable/programmable*

PAL *Programmable Array Logic*

PCI *Peripheral Component Interconnect*

PHR Plataforma de Hardware Reconfigurable

PLD *Programmable Logic Device*

PLL *Phase-Locked Loop*

SEM *Scanning Electron Microscope*

SRAM *Static-RAM*

SSI *Small Scale Integration*

TSV *Through Silicon Via*

Capítulo 1

Introducción

1.1 Motivación y objetivos

En el proceso de aprendizaje de las *Técnicas Digitales* se tiene un eslabón importante que es la implementación de estos sistemas a la práctica. Teorías como el *Álgebra de Bool* con operaciones digitales simples, hasta la síntesis de *microcontroladores* son prácticas comunes en la formación del profesional en el área de la Ingeniería Electrónica y resulta fundamental su ejercitación para concluir el ciclo de enseñanza. Actualmente existen nuevas herramientas al alcance de la mano que pueden facilitar en la formación de estudiante de ingeniería, sobre todo las denominadas *Plataformas o Kit Educativos* que cuentan con una enorme complejidad y recursos en su diseño pero no así para el usuario final. Situación tan distinta en décadas pasadas donde la industria e instituciones académicas de países desarrollados eran quiénes contaba con dichas plataformas.

En el avance tecnológico exponencial que se dio en las últimas décadas del siglo XX se podrían destacar varios logros pero quizá el que toma gran relevancia es el acceso a la información (*Internet*). Esta herramienta permite que regiones en desarrollo puedan llevar adelante estudios de nuevas tecnologías, comparando lo dicho con años anteriores donde no se contaba con la masificación de la información. En la mayoría de los casos las instituciones académicas resultan llevar la bandera en estas búsquedas del conocimiento. En búsqueda de lograr nuestro objetivo principal, ofrecer una herramienta personalizado a las necesidades de los estudiantes que se *inician* en el área del diseño de sistemas digitales basados en Dispositivos Lógicos Programables (*Programable Logic Devices*), se ha trabajado tomando como referencias proyectos universitarios en nuestra Facultad como también de otros países además de los perfiles

de diseño que tienen las placas comerciales.

1.2 Transferencia del proyecto

Como se dijo en la sección anterior, el objetivo del proyecto es diseñar una plataforma que pueda ser útil para los estudiantes iniciales en el diseño de sistemas digitales. En el análisis de la implementación del proyecto en el ámbito académico se optó en coordinar un trabajo conjunto con el *Departamento de Ingeniería Electrónica* y el *Laboratorio de Técnicas Digitales e Informática (LTDI)*, ambas instituciones pertenecientes a la *Universidad Tecnológica Nacional – Facultad Regional Córdoba*. El LTDI es el laboratorio informático principal con el que cuentan los estudiantes de Ingeniería Electrónica y cuenta con recursos físicos y humanos necesarios para cubrir la demanda de las distintas Cátedras que allí se dictan. Miembros del LTDI han formado parte de la generación de documentación necesaria para el proyecto y la intensión a futuro es que dicho personal sea quienes definan mejoras/modificaciones a versiones futuras de la *Plataforma de Hardware Reconfigurable*.

1.3 *Hardware* y *Software*

Las herramientas de *hardware* y *software* son comunes en el campo laboral de un Ingeniero Electrónico. Si bien el profesional Electrónico puede ejercer su actividad en diferentes ámbitos industriales, siempre requerirá el conocimientos de diseño de sistemas físicos como también interactuar con programas informáticos. Seguramente el desenvolvimiento del Ingeniero Electrónico de décadas anteriores contaba con otras herramientas y lograba desarrollarse con éxito en su profesión pues así lo requería la Industria. Hoy la situación es distinta y quizá exista mucha bibliografía que detalle con mayor claridad esta observación, no es nuestra intensión comparar, simplemente poner en contexto la importancia del vínculo entre el mundo físico y los sistemas informáticos.

El nombre del proyecto, *Plataforma de Hardware Reconfigurable*, hace referencia al diseño de placas electrónicas que presentan la posibilidad de reconfigurar su estructura interna y así sintetizar diferentes arquitecturas diseñadas por el usuario, aquí es donde toma importancia en definirla *Hardware Reconfigurable*. Pero a medida que se avance en la lectura del presente informe se podrá notar la importancia que representa el *software* en el proyecto.

Actualmente en el mercado de las plataformas educativas de sistemas embebidos existe verdaderamente una enorme variedad de excelentes productos. Cada uno de estos desarrollos se encuentran orientados a un determinado grupo de usuarios pero la mayoría de estos diseños tienen recursos de *hardware* en común:

- Dispositivo principal de proceso (*hardware*)
- Puerto de programación y depuración (*debugging*)
- Periféricos
- Herramientas de *software*

A la hora de determinar que plataforma se pretende adquirir se debe realizar un análisis de los requerimientos de la implementación y obviamente tener con que presupuesto se dispone. La mayoría de las plataformas comerciales son adquiridas con la finalidad de realizar *prototipos* que permitan clarificar y definir un desarrollo final funcional. Persiguiendo el mismo fin pero en el ámbito académico, el perfil de la plataforma debe ofrecer los recursos físicos estratégicamente necesarios para el avance tecnológico de la región. De esta forma permitirá que los profesionales formados puedan implementar nuevas tecnologías en la industria local.

1.4 PHR (Plataforma de Hardware Reconfigurable)

La plataforma PHR se presenta como una herramienta para las prácticas en las Cátedras del área de Técnicas Digitales. Su estructura está basada en la caracterización planteada en la Sección 1.3. Es decir, la PHR presenta básicamente un Dispositivo Lógico Programable (PLD, siglas en inglés) al cual se tendrá acceso para sintetizar arquitecturas digitales implementadas mediante el uso de Lenguajes Descriptivos de Hardware (HDL, siglas en inglés). Anexo a este dispositivo central se dispone de una memoria de programación donde se almacenará la arquitectura implementada. Estos se accede mediante un puerto de programación estándar denominado JTAG. Además se ofrece dispositivos periféricos que permiten comunicarse al exterior de la placa mediante un puerto de comunicación serial, pulsadores, llaves, indicadores LEDs y display de 7 segmentos. Toda la energía necesaria es proporcionada por una fuente de alimentación capaz de ofrecer tanto los niveles de potencia necesarios para el dispositivo central y sus periféricos, como así también controlar los requerimientos de encendido

del dispositivo central (cuestión que se verá en detalle en Capítulos posteriores). La principal herramienta de *software* que complementa a la plataforma PHR es el encargado de reconfigurar el dispositivo central mediante el puerto JTAG. Se presentaron varias alternativas de *software* que pudieran ser útiles en el proceso de programación tanto libres como privativos. Más adelante se describirá con detalles cada una de estas variantes y se justificará la seleccionada.

1.5 Licencia Libre

El proyecto se encuentra distribuido en con licencias Libres, para ser más exactos *GNU General Public License Version 3 (GPLv3)*. Lo que permitirá que cualquier persona pueda acceder a la documentación como así también los esquemáticos y códigos (*scripts*) que se ha desarrollado. La mayoría de las referencias utilizadas en el desarrollo de la plataforma PHR han sido de otros proyectos que cuentan con licencias similares a la GPLv3. Esta decisión se toma principalmente por una cuestión legal pero también para darle un valor y divulgación a estos diseños que tienen objetivos similares a lo que aquí se pretende lograr con la PHR. De esta manera el proyecto formará parte de otros diseños similares que se encuentran distribuidos en la comunidad electrónica en forma libre o abiertas. Si bien hay desarrollos similares en otras Universidades del mundo, se pretende incentivar a que los estudiantes de esta región no solo tengan acceso a esta tecnología, sino también que se atrevan a realizar modificaciones y aportes a la plataforma PHR.

Capítulo 2

Fundamentos Básicos

2.1 Dispositivos Lógicos Programables

Los *Dispositivos Lógicos Programables* (PLDs) fueron introducidos a medidos de 1970s. La idea era construir circuitos lógicos combinacionales que fueran *programables*. Contrariamente a los microprocesadores, los cuales pueden *correr* un programa sobre un hardware *fijo*, la programabilidad de los PLDs hace referencia a niveles de *hardware*. En otras palabras, un PLD es un chip de *propósitos generales* cuyo *hardware* puede ser reconfigurado dependiendo de especificaciones particulares del programador.

EL primer PLD se llamaba PAL (*Programmable Array Logic*). Estos dispositivos disponían solo de compuertas lógicas (no tenían flip-flop), por lo que solo permitía la implementación de circuitos *combinacionales*. Para salvar este problema, Los *registered* PLDs fueron lanzados pocos después, los cuales incluían un flip-flop por cada salida del circuito. Con esta versión de los PAL, se podría implementar funciones *secuenciales* simples.

En el comienzo de 1980s, se agregaba más circuitos lógicos adicionales a la salida de los PLD. Este circuito de salida se lo identificaba como celda, llamado también *Microcelda*, que contenía (además de flip-flop) compuertas lógicas y multiplexores. Por otra parte, la celda era reprogramable, permitiendo varios modos de operación. Además, se podía proveer una señal de retorno (*feedback*) desde la salida del circuito a la lógico principal de la PAL, lo que le daba mayor flexibilidad a estos dispositivos reprogramables. Esta nueva estructura era llamada *generic PAL* o GAL. Una arquitectura de dispositivo similar fue conocido como PALCE (*PAL CMOS Electrically erasable/programmable*).

Todos estos chips (PAL, *registered* PLD, y GAL/PALCE) son ahora categorizados

como SPLDs (*Simple* PLDs). Los dispositivos GAL/PALCE son los únicos fabricados aún en una encapsulado independiente.

Luego, varios dispositivos GAL fueron fabricados en un solo chip, usando un esquema de direccionamiento más sofisticado, mayor tecnología en su fabricación, y varias características adicionales (como soporte JTAG e interfaces para varios estándares lógicos). Esta nueva propuesta se la conoció como CPLD (*Complex* PLD). Los CPLDs son actualmente muy populares debido a su alta densidad, funcionalidad, y bajo costo.

Finalmente, a mediados de 1980s, las FPGA (*Field Programmable Gate Array*) fueron introducidos al mercado de los IC. Las FPGAs diferían de los CPLDs en su arquitectura. tecnología, recursos internos, y costo. Estos dispositivos tenían como principal objetivo su implementación en diseños de gran requerimientos en recursos de hardware como así también un alto rendimiento.

Un pequeño resumen de los diferentes dispositivos PLDs se puede observar en la Tabla 2.1.

PLDs	Sample PLD (SPLD)	PAL Registered PAL GAL
	Complex PLD (CPLD)	
	FPGA	

Tabla 2.1 Evolución de los PLDs.

Por último, todos los PLDs (*simple* o *complex*) son no volátiles. Estos puede ser OTP (*One-Time Programmable*), en la que pequeños fusibles electrónicos son usados para la reprogramación, de igual forma que las EEPROM o memorias Flash. Las FPGAs, por otra lado, son en su mayoría volátiles. Para estas últimas se deben usar dispositivos externos para cargar las conexiones.

2.2 SPLDs

Como se mencionó anteriormente, los dispositivos PAL, PLA y GAL se clasifican como los *Simple* PLD (SPLDs). Una descripción de las arquitecturas de cada uno de estos dispositivos se presenta a continuación.

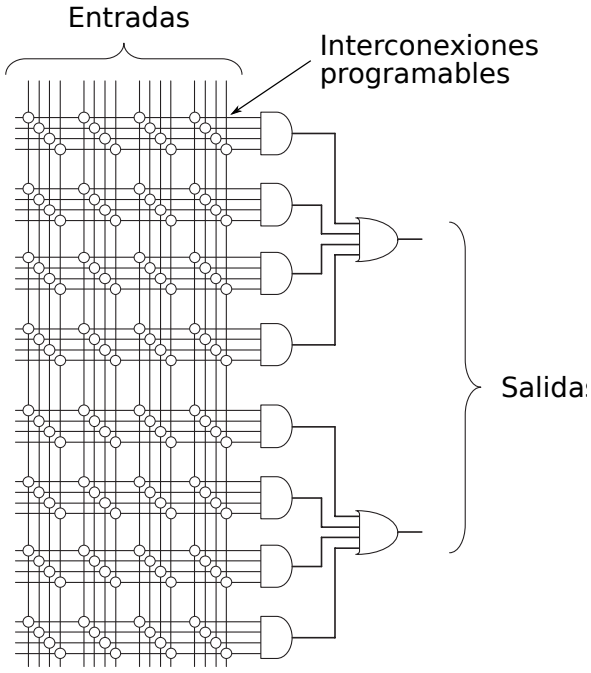


Figura 2.1 Arquitectura básica de una PAL.

2.2.1 PALs

Los *Programmable Array Logic* (PAL) son introducidos por Monolithic Memories Inc. a mediados de 1970. Su arquitectura básica se ilustra en la Figura 2.1, donde se representa con un pequeño círculo las conexiones programables. Como puede verse, el circuito esta compuesto de un arreglo de compuertas AND *programables*, seguido por un arreglo *fijo* de compuertas OR.

La implementación de la Figura 2.1 se basa en que cualquier función combinacional puede ser representada como una Suma de Productos (SOP); es decir, si a_1, a_2, $\cdots$, a_N son las entradas lógicas, entonces cualquier salida combinacional x puede ser compuesta como

$$x = m_1 + m_2 + \cdots + m_M \ , \tag{2.1}$$

donde $m_i = f_1\left(a_1, a_2, \cdots, a_N\right)$ son los términos mínimos de la función x. Por ejemplo

$$x = a_1\bar{a}_2 + a_2a_3\bar{a}_4 + \bar{a}_1\bar{a}_2a_3a_4\bar{a}_5 \tag{2.2}$$

Por lo tanto, el producto (términos mínimos) puede ser obtenido por medio de las compuertas AND, cuya salidas están conectadas a una compuerta OR para calcular su suma.

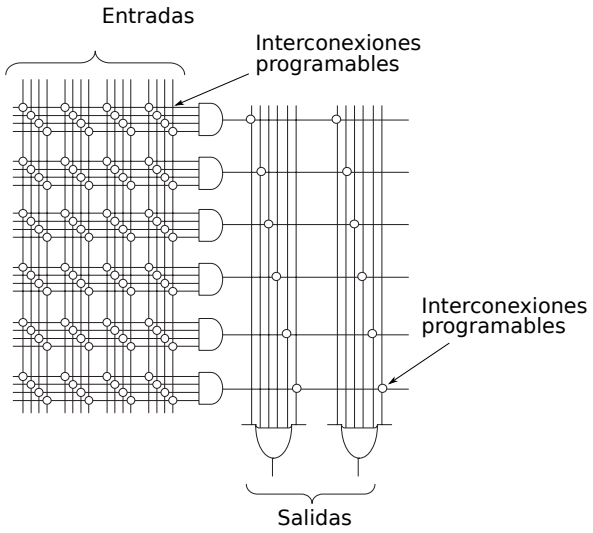


Figura 2.2 Arquitectura básica de una PLA.

La principal limitación de esta arquitectura es el hecho de que solo permite la implementación de funciones combinacionales solamente. Para solucionar este problema, las *registered* PALs fueron lanzadas a fines de la década de 1970s. Estas incluían un flip-flop en cada salida (luego de la compuerta OR en la Figura 2.1), de esta manera permitió la implementación de funciones secuenciales (aunque muy simples).

La primeras tecnologías empleadas en la fabricación de los dispositivos PALs fue bipolar, con una tensión de alimentación de 5V y un consumo de corriente al rededor de 200mA. La máxima frecuencia rondaba los 100Mhz, y las celdas programables eran de PROM (*fuse links*) o EPROM (con un tiemp de borrado de 20min. UV).

2.2.2 PLAs

Los PLA (*Programmable Logic Array*) fueron introducidos a mediados de 1970s (por Signetics Inc.). La arquitectura básica de un PLA se ilustra simbólicamente en la Figura 2.2. Si comparamos esta arquitectura con la Figura 2.1, se observa que la única diferencia fundamental entre estos es que mientras una PAL tiene compuertas AND programables y otras compuertas OR fijas, en el caso de las PLA *ambas* (las compuertas AND y OR) son programables. De esta manera se logra una ventaja en la flexibilidad del diseño. Sin embargo, se presentan elevados tiempos de retardos en los nodos de conexión internos que reducen la velocidad de funcionamiento del circuito.

La tecnología que se empleó en la fabricación de las PLAs fue la misma que en el caso de las PALs. Aunque las PLAs se encuentran obsoletas actualmente, estos han

reaparecido como parte de las arquitecturas de las primeras familias de los CPLDs de baja potencia, como por ejemplo la familia de los *CoolRunner* (de Xilinx Inc.).

2.2.3 GALs

La arquitectura de las GAL (*Generic* PAL) fueron introducidas por Lattice Inc. en los comienzos de 1980s. Este contenía varias mejoras sobre los primeros dispositivos PALs:

1. Se construyeron sealidas más sofisticadas de las celdas (*Macrocell*), las que incluían, además de flip-flop, varias compuertas y multiplexores.

2. Las Macrocell eran programables, permitiendo varios modos de operación.

3. Una señal de "retorno" desde la salida a la Marocell al arreglo reprogramable se incluyó, confiriendo al circuito mayor versatilidad.

4. Se utilizaron EEPROM en lugar de la PROM o EPROM.

Como se mencionó, la GAL es el único SPLD que todavía es fabricado en un encapsulado estándar. Además, éste también sirvió como parte en la construcción de la mayoría de los CPLDs.

La Figura 2.3 muestra un ejemplo de un dispositivo GAL, el GAL16V8. Este circuito cuenta con 16 entradas y 8 salidas, en un *package* de 20 pines. En cada salida hay una Macrocell (luego de la compuerta OR), que contiene, además del flip-flop, compuertas lógicas y multiplexores. Las interconexiones programables son representadas por pequeños círculos. Una señal de realimentación desde la Macrocell al arreglo programable puede también ser observado. Notar que esta arquitectura se asemeja directamente a la de la PAL (Figura 2.1), excepto por la presencia de una macrocell en cada salida y la señal de realimentación.

Actualmente los dispositivos GALs usan tecnología CMOS, alimentados a 3.3V, tecnología EEPROM o Flash, y alcanzan frecuencias máximas que rondan los 250Mhz. Varias compañías fabrican estos dispositivos (Lattice, Atmel, Texas Instruments, etc.).

2.3 CPLDs

La estructura fundamental en la arquitectura de los CPLDs se ilustra en la Figura 2.4. Como se puede ver, este consiste en varios PLDs (en general del tipo GAL) con

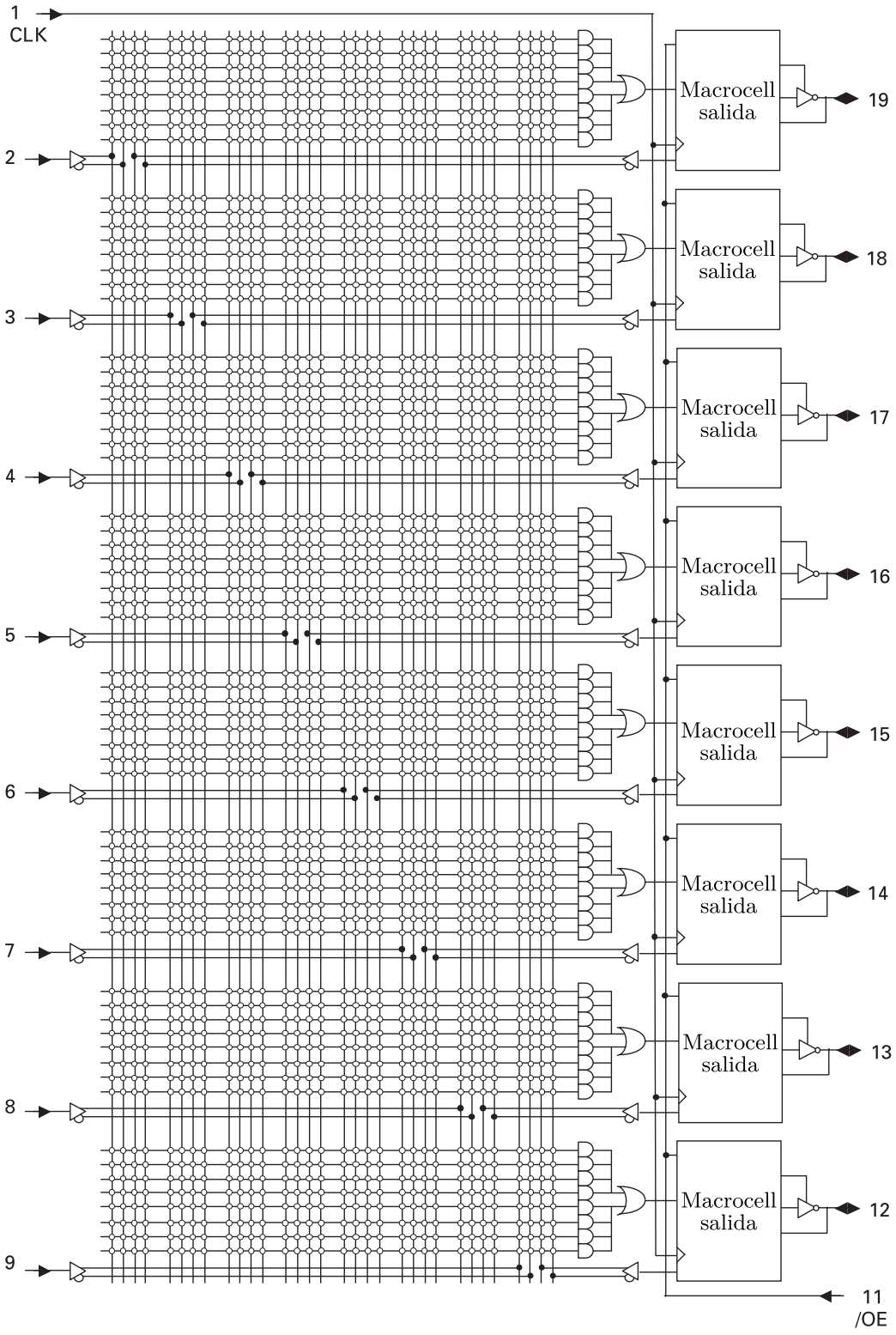


Figura 2.3 Dispositivo GAL 16V8.

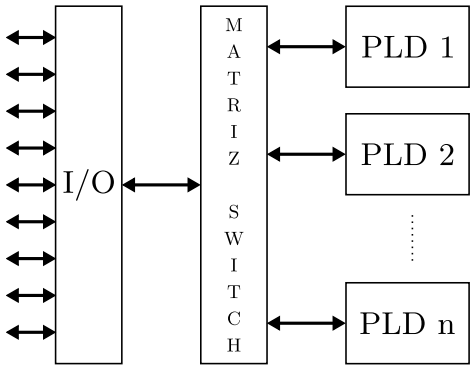


Figura 2.4 Arquitectura básica de un CPLD.

Familia	**XC9500 (XVm, XL)**	**CoolRunner XPLA3**	**CoolRunner II**
Macrocell	36 – 288	32 – 512	32 – 512
System gates	800 – 6,400	750 – 12,000	750 – 12,000
Pines I/O	34 –192	36 – 260	33 – 270
Frec. máxima interna	222 Mhz	213 Mhz	350 Mhz
Building Block	GAL 54V18 (XV, XL) GAL 36V18 (–)	Bloques PLA	Bloques PLA
Voltaje	2.5 V (XV), 3.3 V (XL), 5 V	3.3 V	1.8 V
Interconexiones	Flash	EEPROM	
Tecnología	0.35 μ CMOS	0.35 μ CMOS	0.18 μ CMOS
Corriente estática	11 – 500 mA	$<$ 0.1 mA	22 μA – 1 mA

Tabla 2.2 Características de los CPLDs de Xilinx.

una matriz de *switches* programables usadas para conectarlos todos juntos a al bloque de entrada y salida. Además, los CPLDs contiene normalmente otras características, como soporte JTAG e interfaz a otros estándares lógicos (1.8V, 2.5V, 5V, etc.).

Son varias las compañías que fabrican CPLDs, entre las más reconocidas tenemos Xilinx, Altera, Lattice, Atmel, Cypress, etc. En las Tablas 2.2 y 2.3 se disponen de las características de dos CPLDs, Xilinx y Altera. Como puede verse, más ade 500 Macrocells y más de 10000 compuertas pueden encontrarse en estos dispositivos.

Familia	**MAX7000 (B, AE, S)**	**MAX3000 (A)**	**MAX II (G)**
Macrocell / LUTs	32 – 512 macrocells	32 – 512 macrocells	192 – 1,700 macrocells 240 – 2,210 LUTs
System gates	600 – 10,000	600 – 10,000	
Pines I/O	32 –512	34 – 208	80 – 272
Frec. máxima interna	303 Mhz	227 Mhz	304 Mhz
Voltaje	2.5 V (B), 3.3 V (AE), 5 V (S)	3.3 V	1.8 V (G), 2.5 V, 3.3 V
Interconexiones	EEPROM	EEPROM	Flash + SRAM
Tecnología	0.22 μ CMOS EEPROM 4 capas de metal (7000 B)	0.3 μ 4 capas de metal	0.18 μ 6 capas de metal
Corriente estática	9 – 450 mA	9 – 150 mA	2 –50 mA

Tabla 2.3 Características de los CPLDs de Altera.

2.4 FPGAs

Las FPGAs fueron introducidas al mercado por la empresa Xilinx Inc. a mediados de 1980s. Estos dispositivos se diferencian de los CPLDs en su arquitectura, tecnología de almacenamiento, funcionalidades integradas, y costo, y además están orientadas a la implementación de altos rendimientos y grandes tamaños en lo que se refiere a recursos de hardware.

La arquitectura básica de una FPGA se ilustra en la Figura 2.5. Esta consiste de una matriz de *CLBs* (*Configurable Logic Blocks*), interconectados por un arreglo de matrices de conmutadores (*Switch Matrix*). Para caracterizar con más detalle estos dispositivo se debe recurrir a la información de los fabricantes, donde además se puede disponer de un interfaz JTAG a diversos niveles lógicos, otra funcionalidad como memorias SRAM, multiplicadores de clock (PLL o DLL), interfaz PCI, etc. Algunos chips también incluyen bloques dedicados como multiplicadores, DPSs, y microprocesadores.

Las FPGAs puede ser muy sofisticadas. La fabricación de chips con una tecnología CMOS de 90 nm., con nueve capas ed cobre y mas de 1000 pines de I/O, se encuentran actualmente disponible en el mercado. Algunos ejemplos de los empaquetados (*package*) de las FPGAs son ilustrados en la Figura 2.6, en los cuales se puede apreciar

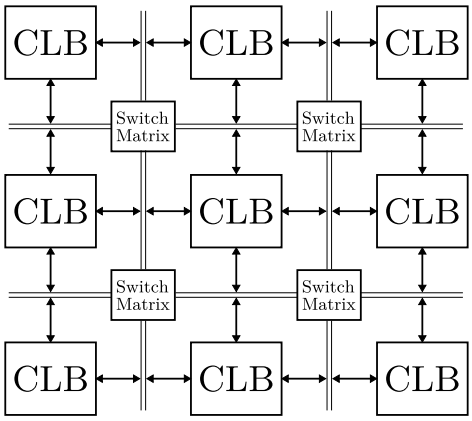
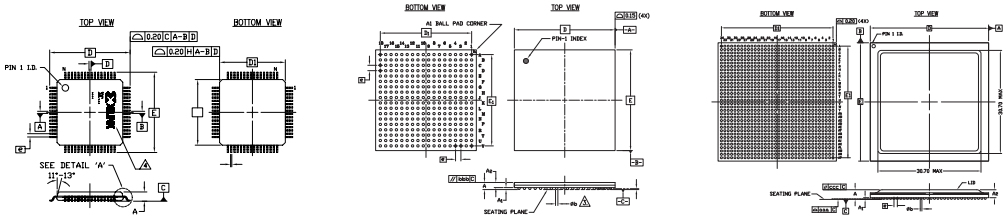


Figura 2.5 Arquitectura básica de una FPGA.



(a) *Package* VQ100 (b) *Package* CSG324 (c) *Package* FFG1156

Figura 2.6 Diferentes *package* de las FPGAs comerciales.

uno de los *package* más pequeños (Fig. 2.6a) con 100 pines, un *package* de tamaño mediano (Fig. 2.6b) de 324 pines, y uno de los grandes *package* con 1156 pines (Fig. 2.6c.

Varias compañías fabrican FPGAs, como Xilinx., Actel, Altera, QuickLogic, Atmel, etc. Ejemplo de dos fabricantes (Xilinx y Actel) se disponen en las Tablas 2.4 y 2.5. Como puede verse, estos dispositivos pueden contener miles de flip-flops y varios millones de compuertas lógicas.

Nótese que todas las FPGAs de Xilinx usan SRAM para almacenar las interconexiones, por lo que son reprogramables, pero volátiles (es así que requieren de una ROM externa). en cambio, las FPGAs de Actel son no-volátiles (estos usan fusibles electrónicos), pero no son reprogramables (excepto una familia, la cual usa memoria *Flash*). Ya que cada enfoque tiene sus propias ventajas y desventajas, la aplicación real dictará cual arquitectura de chip es la apropiada.

Familia	**Virtex II Pro**	**Virtex II**	**Virtex E**	**Virtex**	**Spartan 3**	**Spartan IIE**	**Spartan II**
CLBs	352 – 11.024	64 – 11.648	384 – 16.224	384 – 6.144	192 – 8.320	384 – 3.456	96 – 1.176
Celdas Lógicas	3.168 – 125.136	576 – 104.882	1.728 – 73.008	1.728 – 27.648	1.728 – 74.880	1.728 – 15.552	432 – 5.292
System gates		40k – 8M	72k – 4M	58k – 1.1M	50k – 5M	23k – 600k	15k – 200k
Pines de I/O	204 – 1200	88 – 1108	176 – 804	180 – 512	124 – 784	182 – 514	86 – 284
Flip-flops	2.816 – 88.192	512 – 93.184	1.392 – 64.896	1.392 – 24.576	1.536 – 66.560	1.536 – 13.824	384 – 4.704
Frec. máxima interna	547 MHz	420 MHz	240 MHz	200 MHz	326 MHz	200 MHz	200 MHz
Voltaje	1.5 V	1.5 V	1.8 V	2.5 V	1.2 V	1.8 V	2.5 V
Inter-conexiones	SRAM	SRAM	SRAM	SRAM	SRAM	SRAM	SRAM
Tecnología	0.13 μm 9 capas de cobre CMOS	.15 μm 8 capas de metal CMOS	0.18 μm 6 capas de metal CMOS	0.22 μm 5 capas de metal CMOS	0.09 μm 8 capas de metal CMOS		
SRAM bits (Bloques de RAM)	216k – 8M	72k – 3M	64k – 832k	32k – 128k	72k – 1.8M	32k – 288k	16k – 56k

Tabla 2.4 Características de FPGAs fabricadas por Xilinx.

Familia	**Accelerator**	**ProASIC**	**MX**	**SX**	**eX**
Módulos lógicos	2.016 – 32.256	5.376 – 56.320	295 – 2.438	768 – 6.036	192 – 768
System gates	125k – 2M	75k – 1M	3k – 54k	12k – 108k	3k – 12k
Pines de I/O	168 – 684	204 – 712	57 – 202	130 – 360	84 – 132
Flip-flops	1.344 – 21.504	5.376 – 26.880	147 – 1.822	512 – 4.024	128 – 512
Frec. máxima interna	500 MHz	250 MHz	250 MHz	350 MHz	350 MHz
Voltaje	1.5 V	2.5 V, 3.3 V	3.3 V, 5 V	2.5 V, 3.3 V, 5 V	2.5 V, 3.3 V, 5 V
Interconexiones	*Antifuse*	*Flash*	*Antifuse*	*Antifuse*	*Antifuse*
Tecnología	0.15 μm 7 capas de metal CMOS	.22 μm 4 capas de metal CMOS	0.45 μm 3 capas de metal CMOS	0.22 μm CMOS	0.22 μm CMOS
SRAM bits	29 k – 339 k	14 k – 198 k	2.56 k	n.a.	n.a.

Tabla 2.5 Características de FPGAs fabricadas por Actel.

2.5 Lenguajes Descriptivos de *Hardware*

La forma tradicional de diseñar circuitos digitales es dibujar diagramas lógicos que contengan compuertas (SSI) y funciones lógicos (MSI). Sin embargo, a fines de 1980s y comienzo de 1990s este proceso de diseño presentaba limitaciones como así algunos problemas. *¿Como se puede dibujar diagramas esquemáticos que contienen cientos de miles o millones de compuertas?* Con la disponibilidad de los dispositivos lógicos programables para reemplazar sistemas donde se utilizaban integrados como los TTL, un nuevo enfoque para el diseño digital fue necesario. Las herramientas asistidas por computadoras son esenciales para diseñar circuitos digitales en la actualidad. Es claro que en las últimas décadas los ingenieros digitales de hoy diseñan sistemas digitales mediante la utilización de *software*! Esto es un importante cambio de paradigma del tradicional método empleado para el diseño de sistemas digitales.

Actualmente los diseñadores digitales usan *Lenguajes Descriptivos de Hardware* (HDLs) para diseñar sistemas digitales. Los lenguajes más utilizados son *VHDL* y *Verilog*. Ambos lenguajes descriptivos permiten al usuario diseñar sistemas digitales mediante la escritura de código que describen el comportamiento de un circuito digital. Este código puede ser utilizado tanto para *simular* la operación del circuito y

sintetizar también implementarse dicho circuito en un CPLD, una FPGA o en un circuito integrado de aplicaciones específica (ASCI).

2.5.1 VHDL

El lenguaje VHDL surgió como parte de un programa norteamericano denominado *Very High Speed Integrated Circuits* (VHSIC), a comienzos de 1980. En el desarrollo de la ejecución de este programa surgió la necesidad de contar con un lenguaje que permita describir la estructura y funciones para los circuitos integrados (ICs). Es así que el VHSIC *Hardware Description Language* (VHDL) fue desarrollado. Luego la IEEE adoptaría como un lennguaje estándar en los Estados Unidos.

VHDL fue diseñado para cubrir necesidades el proceso de diseño. Primero, este lenguaje permite la descripción de la estructura de un diseño, de esta forma se puede descomponer en sub-diseños, y a la vez como estos sub-diseños se interconectan entre sí. Segundo, VHDL permite la especificación de la función de los diseños usando las formas del lenguaje de programación similares a otros lenguajes familiares. Tercero, permite a un diseño ser simulado antes de ser fabricado, por lo que los diseñadores puede rápidamente compara alternativas y probar correciones sin el retardo y espera de los prototipos en *hardware*.

2.5.2 Verilog

Verilog esta basado en el lenguaje de programación C en la estructura de la sintaxis pero la manera en la que se comporta es diferentes pues es un lenguaje descriptivo. Este formato permitió una rápida aceptación por parte de los diseñadores de *hardware*.

Con el incremento en el éxito de VHDL, Cadence decidió hacer el lenguaje abierto y disponible para estandarización. Cadence transfirió Verilog al dominio público a través de Open Verilog International, actualmente conocida como Accellera. Verilog fue después enviado a la IEEE que lo convirtió en el estándar IEEE 1364-1995, habitualmente referido como Verilog 95.

2.6 Diseño de sistemas digitales

En el proceso de enseñanza de los sistemas digitales se requiere de recursos físicos que complementen el contenido teórico. En la carrera de ingeniería electrónica de nuestra casa de estudio, las técnicas digitales se clasifican en cuatro niveles:

Técnicas Digitales I el contenido comprende conceptos desde el Álgebra de Boole, funciones lógicas, sistema de numeración, codificadores/decodificadores, circuitos secuenciales, manejo de lenguajes descriptivos de *hardware*.

Técnicas Digitales II Métodos de discretización, convertidores AD/DA, microprocesadores, microcontroladores.

Técnicas Digitales III Instrumentación virtual, adquisición y acondicionamiento de señales (DAQ), redes de computadoras, DSP, sistemas lineales (convolución, correlación/autocorrelación y Fourier), interpolación/decimación (ventanas: rectangular, Hanning, Hamming, Blackman, Kaiser), filtros digitales.

Técnicas Digitales IV Arquitecturas de lógicas programables, sistemas de diseño para PLDs, procesado y mecanismos de simulación del lenguajes VHDL, síntesis, modelado con VHDL.

La tecnología lógica programable (PLD) es desarrollada en dos de las cuatro cátedras del área de *técnicas digitales*. A mediados de la década del 2000, se comenzó a introducir fuertemente la posibilidad de implementar los diseños digitales sobre dispositivos PLD. Lo que ha requerido la capacitación de los docentes sobre esta tecnología. El Centro Universitario de Desarrollo en Automoción y Robótica fue quién innovó sobre la formación de recursos humanos para la transferencia de conocimientos y desarrollos para los laboratorios. Tal es así la inserción y actualización del área que la cátedra electiva *Técnicas Digitales IV* surgió no hace más de cuatro años. Estos recursos tecnológicos fueron paulatinamente incluidos en las cátedras.

2.7 Influencia de la Programabilidad

En muchos textos la ley de Moore es usada para destacar la evolución de la tecnología de silicio en la industria de los dispositivos semiconductores. Poro hay otro interesante punto de vista particularmente para los dispositivos PLDs, la *onda de Makimoto* que fue publicada por primera vez en Enero de 1991 por la revista *Electronics Weekly*. Este concepto se basa en la observación de Tsugio Makimoto quién notó que la tecnología se desplazaba entre la *estandarización* y la *personalización* (véase la Figura 2.7). En el comienzo de la década de 1960s, un número de componentes estándares fueron desarrollados, llamados series lógicas 7400 (por Texas Instruments). Dispositivos que servían para crear diversas aplicaciones digitales. Entrada la década de 1970s, la época

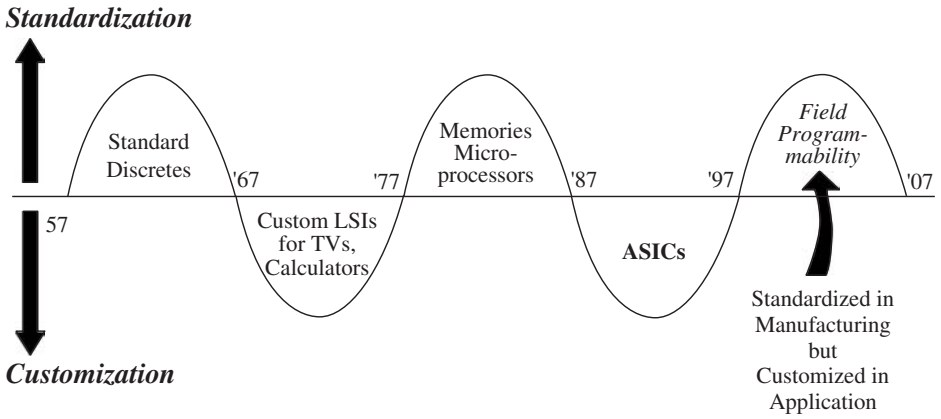


Figura 2.7 Onda de Makimoto.

de los dispositivos personalizados (LSI, siglas en inglés de *Low-Scape Integration*) comenzó a desarrollarse donde los chips eran creados para aplicaciones específicas como ser una calculadora. El chip fue incrementando su nivel de integración y así fue que nació el termino integración a media escala (MSI, siglas en inglés de *Medium-Scale Integration*). La evolución de los microprocesadores en la década de 1970s llevó a la estandarización de chips que fueran usados para un amplio rango de aplicaciones. Es entonces que en 1980s nació el ASIC (*Application-Specific Integrated Circuit*) donde el diseñador podría superar la limitación de la secuencialidad de los microprocesadores, quienes poseían varias limitaciones en aplicaciones en DSP (*Digital Signal Processing*) donde se requería un mayor nivel de cálculos. La aparición de la FPGA como un dispositivo con la capacidad de proporcionar recursos lógicos necesarios para conectar varios componentes entre sí llevo a que se conviertan en dispositivos populares.

Se podría considerar la existencia de dos épocas de la *programabilidad* donde la *primera* época ocurre con la aparición del microprocesador en la década de 1970s, donde los programadores desarrollan soluciones programables basados sobre estos *hardware* fijos. El gran reto en esta época fue en entorno de *software*; los desarrolladores trabajaban con lenguajes *assembly* e incluso cuando los compiladores y ensambladores surgieron para el lenguaje C, pues se obtenían mejores rendimientos con la codificación manual. Se comenzaron a obtener librerías que proporcionaban funciones básicas, permitiendo al diseñador concentrarse en la programación de la aplicación. Estas funciones actualmente son fácilmente accedidas desde los compiladores y ensambladores comerciales/libres. Actualmente hay una gran demanda de lenguajes de programación de alto-nivel como C y Java. Tal es así la abstracción del lenguaje que incluso entornos de desarrollos de alto nivel como UML están siendo

implementados.

La *segunda* época de la programabilidad se encuentra marcada por las FPGAs. En la Figura 2.7, Makimoto indica que el campo de la programabilidad se estandariza para su fabricación y la personalización del diseño se encuentra en la capa de aplicación de un desarrollo con las tecnologías mencionadas. Esto puede ser considerado como lo que ofrece la programabilidad de *hardware* en el dominio del *software* donde el *hardware* permanece fijo. Esto es un reto fundamental como la mayoría de las herramientas de programación de computadora que trabajan sobre el principio de una plataforma de *hardware* fijo, lo que permite realizar optimizaciones ya que hay una orientación clara sobre la manera de mejorar el rendimiento de una representación algorítmica. Con las FPGAs, el usuario tiene plena libertad para definir la arquitectura que mejor se adapte a la aplicación. Sin embargo, esto presenta un problema en el que cada solución debe ser *hecha a mano* y todos los diseñadores de *hardware* conocen los problemas en el diseño y verificación.

Algunas de las tendencias en las dos épocas tienen similitudes. En los primeros días, el modo esquemático (*schematic capture*) fue usado para diseñar los primeros circuitos que era sinónimo con el nivel *assembly* en programación. Los lenguajes de descripción de *hardware* como el VHDL y Verilog emergieron ya que podrían ser utilizados para producir un nivel de abstracción más alto con el objetivo de contar con una herramienta basada en C como son SystemC y CataultC de Mentor Graphics como un entorno único de programación. Inicialmente como con los lenguajes de programación de *software*, había una desconfianza en la calidad de los resultados que producía el código con este nuevo enfoque. Sin embargo, con el fin de mejorar los costos de desarrollo, las herramientas de síntesis que eran equivalentes a la evolución de los compiladores de *software* eficientes para los lenguaje de alto-nivel, y también la evolución de las funciones de librería, estableció un alto grado de confianza que posteriormente llevó al uso de los lenguajes descriptivos de *hardware* (HDLs) sean comnues para la implementación en FPGA. En efecto, el surgimiento de los IP-cores refleja la evolución de librerías como son funciones programables de entradas/salidas para el flujo del *software* donde funciones comunes fueron reutilizadas donde los desarrolladores confiaban en la calidad de los resultados que producían estas librerías, especialmente en lo que las presiones para producir más código en el mismo lapso de tiempo crecieron con la evolución tecnológica. Los primeros IP-cores surgieron a partir de funciones de librerías básicas en el procesamiento de señales complejas y funciones de comunicación la mayoría de estos suministrados por los proveedores de FPGA y

diversos repositorios web de IP-cores.

Capítulo 3

Antecedentes

Una de las principales actividades de los centros y grupos de investigación es la generación de recursos y contenidos académicos. Si bien las líneas de trabajo son definidas por una secretaría a nivel regional, en muchos casos la transferencia de investigación y desarrollo (I+D) beneficia a las mismas instituciones académicas permitiendo la actualización tecnológica con los resultados arribados por dichos centros.

El *Centro Universitario de Desarrollo en Automoción y Robótica* (CUDAR) desarrolla sus actividades en la Universidad Tecnológica Nacional – Facultad Regional Córdoba. Las principales actividades y áreas del Centro se puede clasificar como,

- Investigación y Desarrollo
  - Área Electrónica
  - Área Mecánica
  - Área Informática
- Extensión y Tecnología
  - Convenios y transferencias
  - Cursos de capacitación
  - Consultoría y servicios

En la búsqueda de nuevas tecnologías es que miembros del CUDAR comienzan el estudio de la tecnología PLD en la década del 2000. La principal actividad se centra en la participación de congreso y seminarios relacionados con la lógica programable.

Con el proyecto *Robot Industrial de Arquitectura Reconfigurable Implementado con FPGA* desarrollado en el CUDAR se institucionalizó la investigación y se comenzó a realizar desarrollos relacionados con dispositivos reconfigurables como CPLD y FPGA. Estos esfuerzos en la formación y adquisición de nuevas tecnologías llevó a la actualización de la cátedra *Técnicas Digitales I*, obviamente que fue necesaria la introducción de estos nuevos conceptos debido a la posibilidades que se tenían para adquirir estos

dispositivos que anteriormente no se encontraban comercializados en forma tan masiva en nuestra región. Hace menos de cinco años el CUDAR impulsó la creación de la cátedra electiva *Técnicas Digitales IV* con el objeto de dar continuidad a los estudiantes de ingeniería electrónica interesados en especializarse sobre la implementación de sistemas complejos basados en sistemas digitales reconfigurables.

Para acompañar el proceso de formación tanto en el CUDAR como en las cátedras de grado se requirió contar con recursos de *hardware* donde implementar los sistemas digitales diseñados. Si bien se logró adquirir plataformas educativas, estratégicamente el CUDAR, desde el comienzo, dispuso el desarrollo de plataformas propias que lograran cubrir las necesidades del momento en materia de periféricos y otras cuestiones.

3.1 Placa MiniLab

En la cátedra de Técnicas Digitales I, la primera aproximación con los sistemas digitales físicos es el armado del denominado *Proyecto MiniLab*. El circuito del MiniLab es una herramienta para los trabajos prácticos de diferentes cátedras de electrónica que cursan los alumnos del tercer nivel de la carrera ingeniería electrónica. Y es que esta plataforma permite montar y testear circuitos con relativa facilidad. Dispone recursos digitales básicos con el fin de realizar prueba, tiene señales lógicas de entrada ('0' y '1') y también cuenta con indicadores LED que puede ser utilizados como salidas digitales. Además de entrada/salidas se dispone de un generadores de pulsos de reloj. En la Figura 3.1 se puede ver una fotografía del proyecto MiniLab.

Sobre esta plataforma multi-propósito se montan diferentes circuitos integrados (ICs) con compuertas (NOT, OR, AND, XOR, etc.) que permiten al estudiante implementar las diferentes funciones digitales a medida que avanza el contenido de la materia. Es aquí el primer antecedente sobre la disposición de recursos físicos vinculados con el contenido académico. Al transcurrir los años consecuentemente los avances tecnológicos demandaron, por parte de plataformas como el MiniLab, ofrecer nuevos recursos. Ya el MiniLab resultaba limitado para los estudiantes iniciales. Un factor clave resulta ser la decisión de introducir en la cátedra de Técnicas Digitales I los conceptos de lógica programable y acercar a los estudiantes a esta nueva tecnología, complementando así los recursos que dispone el MiniLab.

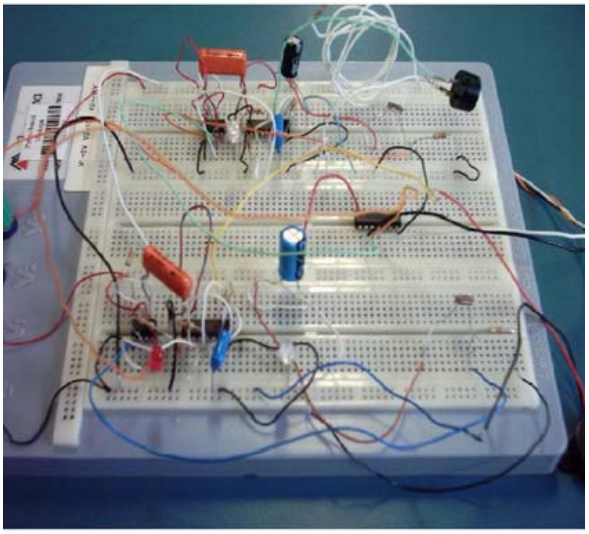
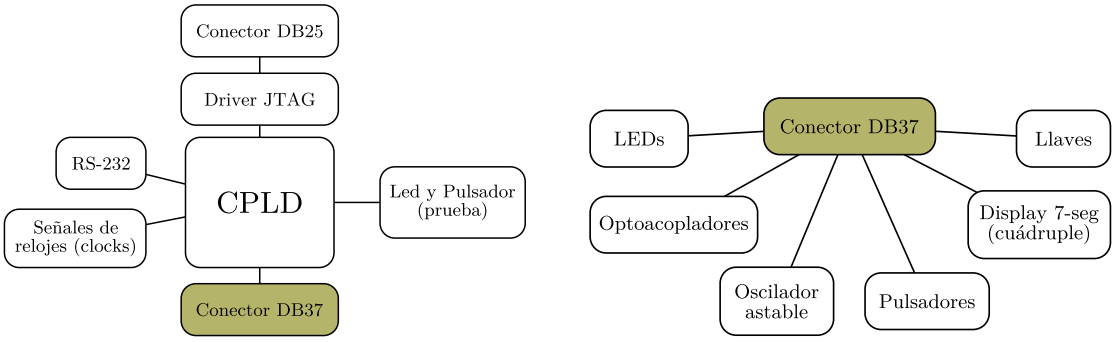


Figura 3.1 Implementación de un circuito electrónico montado sobre le *MiniLab*.

3.2 Kit CPLD

El *Kit CPLD* desarrollado por el CUDAR en el año 2005 y publicado posteriormente en el año 2006 se presenta como un *Equipo de Laboratorio basado en CPLD*. Esta plataforma permite simular y desarrollar circuitos digitales basados en lógica programable. Optimizado para reducir el costo de su fabricación, el Kit CPLD permitía introducir a los alumnos a los lenguajes de descripción de hardware (principalmente VHDL).

El diseño consta de dos bloque funcionales perfectamente acotados y cada uno de ellos está implementado sobre una placa independiente una de la otra. En la Figura 3.2 se representa en un diagrama de dos bloque el Kit CPLD. Estos bloques se denominarán de aquí en adelante como bloque programador (Fig. 3.2a) y bloque expansión I/O (Fig. 3.2b). El primer bloque indispensable en este Equipo tiene las siguientes particularidades. Alojar un CPLD con la electrónica necesaria para poder ser programarlo (JTAG), comunicación en forma serial para usos diversos y conectores que interconectan los pines físicos de entrada y salida del chip CPLD con el exterior. El otro boque se puede considerar de uso opcional, debido a que es una extensión del primer bloque y es utilizado principalmente como interfase entre el usuario y la lógica con la cual será programado el CPLD.



(a) Placa principal con el CPLD (b) Placa expansión, conector DB-37

Figura 3.2 Diagrama en bloque de la primera versión del *Kit CPLD*.

Capítulo 4

El proyecto PHR

4.1 Desarrollos de referencias

4.2 Estructura general del proyecto

4.3 Definición de estructura de las placas

4.4 Selección de dispositivos principales

4.5 Descripción de las placas

4.5.1 PHR

4.5.2 S3Power

4.5.3 OOCDLink

Capítulo 5

Implementación del Proyecto

5.1 Necesidad de plataformas de *hardware*

5.2 Antecedentes

5.3 Cátedras beneficiadas

5.4 Inserción de los lenguajes descriptivos

Capítulo 6

Costos y Financiamiento

6.1 Planificación de los gastos

6.2 Procedimientos en adquisición de materias primas

6.3 Financiación del Proyecto

6.4 ¿Costos en *Software*?

Capítulo 7

Protocolo JTAG

7.1 Características del Protocolo

7.2 Dispositivo FT2232D (USB/JTAG)

7.2.1 Controladores del dispositivo

7.3 Aplicación del protocolo JTAG

Capítulo 8

Proceso de Implementación Lenguajes HDLs

8.1 Flujo del Diseño con HDL

8.2 Lenguajes

8.3 Herramientas de *Software* de Xilinx

8.3.1 ISE

8.3.2 Impact

Capítulo 9

Programación de la PHR

9.1 Flujo del proceso de programación

9.2 Manejo de *scripts*

Capítulo 10

Proyectos *Open Hardware*

10.1 Fundamentos y principios básicos

10.2 Tipos de Licencias

10.3 Importancia en la Educación

10.4 Algunos proyectos *Libres*

10.4.1 Nacionales

10.4.2 Internacionales

Capítulo 11

Conclusiones

En principio se copian las mismas secciones que se tenían en el Capítulo 1. La copia se debe a que la conclusión debe ser una consecuencia de cada uno de las secciones planteadas al comienzo. Seguramente los nombres no serán iguales pero están relacionados.

11.1 Motivación y objetivos

11.2 Transferencia del proyecto

11.3 *Hardware* y *Software*

11.4 PHR (Plataforma de Hardware Reconfigurable)

11.5 Licencia Libre

Referencias

Anexo A

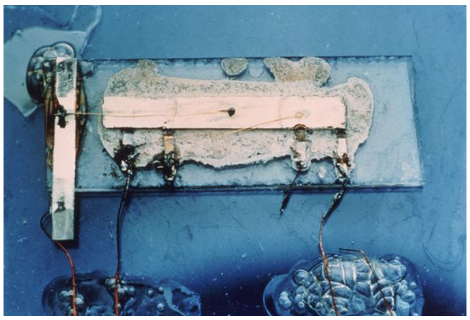
Dispositivos Electrónicos

A.1 Evolución de los Circuitos Integrados

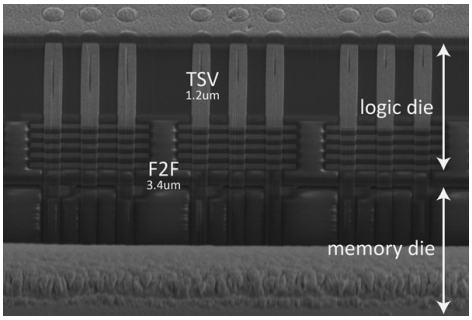
El costo total de un sistema digital se puede decir que está determinado por las placas electrónicas, sistema de alimentación, interconexiones y encapsulado, así como el diseño, testeo, y otros costos de producción que también fueron comentados en la Sección 1.3 del Capítulo introductorio. Aquí los dispositivos digitales representan una fracción del total del costo del sistema. Consecuentemente, todo diseñador pretende realizar un sistema donde sea mínimo el total de circuitos integrados utilizados, lo que en muchos casos permitirá reducir las dimensiones de la placa electrónica, requerimientos de potencia y otros costos relacionados.

En las últimas décadas, el número de transistores que forman un circuito integrado (IC, por sus siglas en inglés) se ha incremento debido al creciente nivel de integración en los procesos de fabricación de estos dispositivos electrónicos. La primera aparición pública del IC[1] fue en Mayo de 1952 por Geoffrey W. A. Dummer, quién presentó la idea de la integración en un Simposio sobre "Avances en Componentes Electrónicos de Calidad". En el año 1959 Jack Kilby fue quién diseñó el primer circuito integrado, dispositivo de germanio que *integraba* seis transistores en una misma base semiconductora para formar un oscilador de rotación de fase. Y es así que en la década de '80 se manipulaba aproximadamente un millón de transistores, a fines de la misma década billones de transistores que integran un IC. Actualmente se está trabajando en tecnologías mucho más novedosas, sobre todo en el proceso de fabricación, los Circuitos Integrados en 3 Dimensiones (3D-IC). Aquí se trabaja con dos o más capas

[1]Se conoce que Werner Jacobi, en 1949, tramito la patente de un circuito integrado pero no se registra que se haya realizo una implementación funcional.



(a) El primer IC creado por Jack Kilby en 1958. Contiene solo un transistor y componentes de soporte sobre una base de germanio.



(b) Imagen microscópica (SEM) de un 3D-IC desarrollado por la empresa Tezzaron Semiconductor. Se identifica una parte del chip, lógica, memoria, vías de conexión verticales (TSV) y terminales de unión (F2F).

Figura A.1 Evolución y comparación de los circuitos integrados

de componentes electrónicos activos que son integrados tanto en forma vertical como horizontal sobre un mismo circuito.