# Diseño de Circuitos Lógicos en base a la tecnología FPGA: Un ejemplo de aplicación (Compendio)

*M. en C. Juan C. González Robles*
*Jefe del Departamento de Producción y Adecuación de Tecnologías del CINTEC-IPN.*
*Ing. Eduardo Vega Alvarado*
*Jefe del Departamento de Laboratorios Ligeros del CINTEC-IPN.*

Continuando con la serie de artículos sobre el diseño en base a la lógica programable, el presente trabajo pretende describir una aplicación diseñada y construida con un arreglo de compuertas programables en campo (Field Programmable Gate Array, FPGA).

## *Introducción*

Para el presente desarrollo servirá como base teórica el artículo "Arreglos de Compuertas Programables en Campo, FPGA's (compendio)" **[1]**, en el cual se describen las principales variantes comerciales en relación a estos dispositivos.

Si bien el dispositivo seleccionado es del tipo de Arreglo de Celdas Lógicas (Logic Cell Array, LCA), los criterios de diseño y las herramientas de programación aplicadas son, en lo general, similares a los correspondientes a otras familias de dispositivos FPGA. La razón de esta selección es que los LCA tienen una gran aceptación como alternativa práctica de los tradicionales PLDs, y en algunos casos, como reemplazo para arreglos de compuertas simples. Para este diseño, primeramente se hará una breve descripción de los dispositivos LCA, para posteriormente manejar el tema de implantación.

## *Arquitecturas LCA*

Existen tres familias de LCAs de uso común, que son las series 2000, 3000 y 4000 de Xilinx Incorporation; así mismo, se cuenta con dispositivos similares provenientes de segundas fuentes , tales como AT&T.

### La serie 2000 de Xilinx

El Xilinx XC2064 es el dispositivo más simple de esta serie; contiene un total de 64 CLBs distribuidos en una matriz de ocho por ocho. En los LCAs de la serie 2000, cada CLB contiene una sección de lógica combinacional que es capaz de producir cualquier función de hasta cuatro variables, contando para ello con cuatro entradas de propósito general: A, B, C, y D, y una entrada especial de reloj (K). La sección lógica combinacional del CLB se maneja como una tabla de consulta, similar a una PROM.

El retardo a través de la celda lógica es constante, sin importar la función lógica que se maneje. Además , el CLB puede ser configurado para usarse como dos funciones lógicas de tres entradas, o en aplicaciones de multinivel, incluyendo algunas funciones de hasta cinco entradas mediante una operación de multiplexaje; ambos modos se

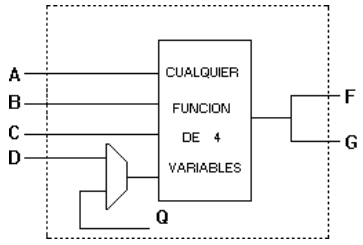


**Figura 1**
Función con una sola salida

muestran en las **figuras 1 y 2** :

El flip-flop de cada CLB puede

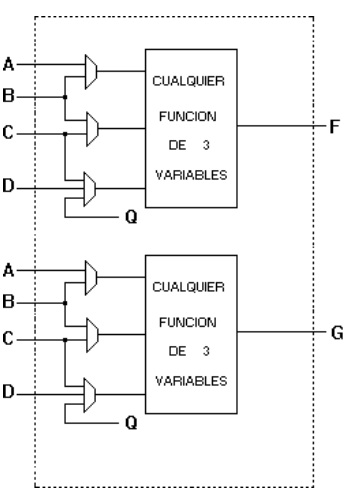


**Figura 2**
Función con dos salidas

simular un tipo D disparado por flanco o un latch por nivel; cuando no se usa como elemento de alma-

cenamiento se deshabilita. Cada elemento incluye funciones no sincrónicas de puesta y limpiado (set y reset, respectivamente).

También se cuenta con 58 bloques de Entrada/Salida ("Input/Output", I/O) para el XC2064, colocados en la periferia del dispositivo; estos bloques proveen la interface entre las terminales externas y la lógica interna. Cada bloque puede configurarse para operaciones como entrada, salida, salida en tercer estado, o buffer bidireccional. En la **figura 3** se muestra la arquitectura general de los bloques de I/O; la entrada contiene un circuito de detección de umbral para trasladar señales externas o niveles lógicos internos; este circuito permite

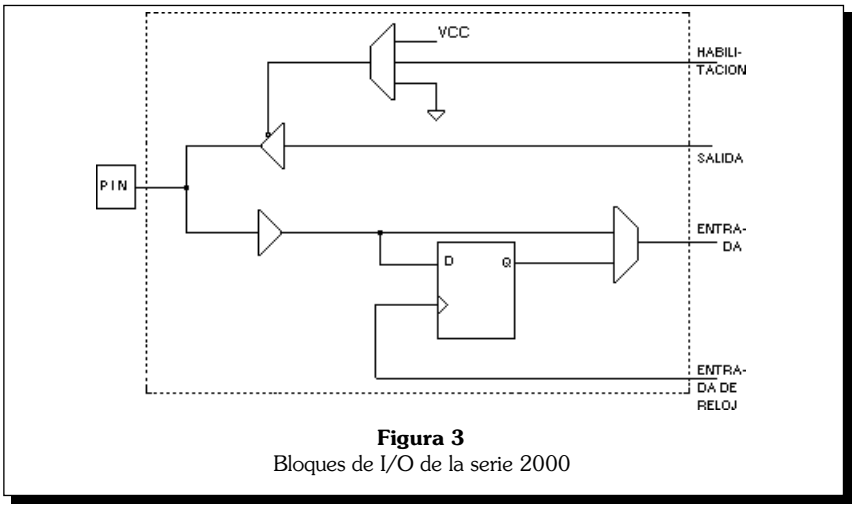


**Figura 3**
Bloques de I/O de la serie 2000

al dispositivo operaciones con niveles TTL o CMOS.

Los canales de interconexión programables del LCA proveen la trayectoria para conectar entradas y salidas de los bloques de I/O y las celdas lógicas en toda la configuración. Estos canales son programados en los puntos de cruce llamados *puntos de interconexión programable,* clasificándose en tres tipos: a) interconexiones de propósito general, b) líneas distantes, c) interconexiones directas.

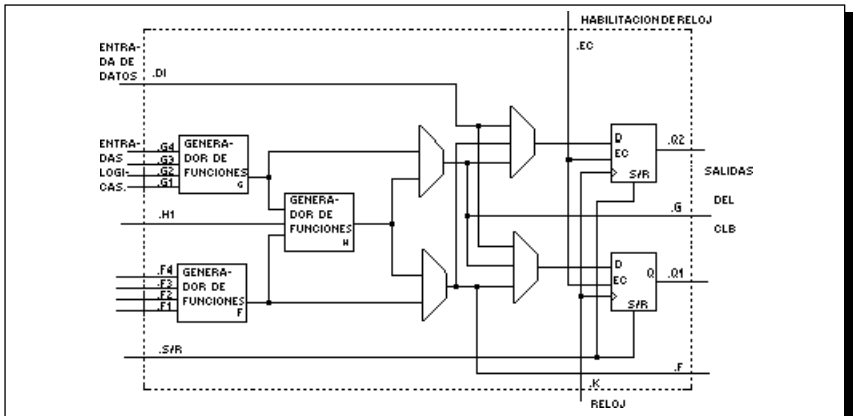


**Figura 4**
CLB de la serie 4000

**La serie 3000 de LCAs.**

Esta serie difiere de la serie 2000 en la complejidad de sus CLBs e IOBs, así como por la adición de un bus interno (especializado en líneas de gran longitud) para rutas de señales multiplexadas a través del dispositivo. El CLB contiene dos flip-flops, y tiene más líneas de entrada que las presentes en la serie 2000; cinco de estas entradas pueden aplicarse como entradas combinacionales, mientras las restantes se emplean para el reset y el reloj de los flip-flops, y para funciones de carga de datos. Los bloques de I/O incluyen flip-flops tipo D como registros de salida, que pueden trabajar alternativamente como latches de entrada.

**La serie 4000 de LCAs.**

Esta serie fue anunciada en 1989, está basada en RAM estática conteniendo CLBs y varios tipos de recursos de interconexión. Se considera que las mejoras en la arquitectura de ruteo y en el proceso incrementan la velocidad de operación de las serie 4000 al doble de la disponible en la serie 3000. El uso de un número significativamente mayor de recursos de interconexión contribuye, a su vez, a una mayor eficiencia en el ruteo de los dispositivos de esta serie.

Los CLBs en esta serie son similares a los de la serie 3000 (**figura 4**); la diferencia es que cuenta con dos bloques de lógica combinacional con entradas separadas, con lo que un solo CLB puede implementar funciones lógicas de hasta nueve entradas.

## *Proceso de diseño y programación del FPGA*

Conceptualmente hablando, el transformar un diseño en un circui-

to basado en LCA que funcione apropiadamente es un proceso considerablemente más largo y laborioso que el desarrollo equivalente utilizando PLDs. La primera consideración es determinar que tan adaptable resulta el diseño inicial a una implementación con LCAs. Una vez tomada la determinación de aplicar esta tecnología, la siguiente etapa involucra el empleo de una herramienta de diseño proporcionada por el fabricante. Para el caso de Xilinx, este sistema recibe el nombre de Xact, llevándose a cabo los siguientes pasos (los cuales se muestran en la **figura 5**):

1.- A partir del diagrama esquemático, se obtienen los archivos de formato de la lista de conexiones (XNF). Este formato es una simple representación de los diseños y no involucra aún el uso de un dispositivo específico.

2.- El formato XNF generado se maneja para crear un circuito optimizado dirigido al LCA en particular que se desea utilizar. En esta etapa es posible combinar varios archivos XNF para generar un solo circuito, así como efectuar varias optimizaciones al mismo. Finalmente se obtiene un archivo denominado de diseño LCA, el cual en realidad consiste en la descripción detallada para el Xact de lo que se desea implementar.

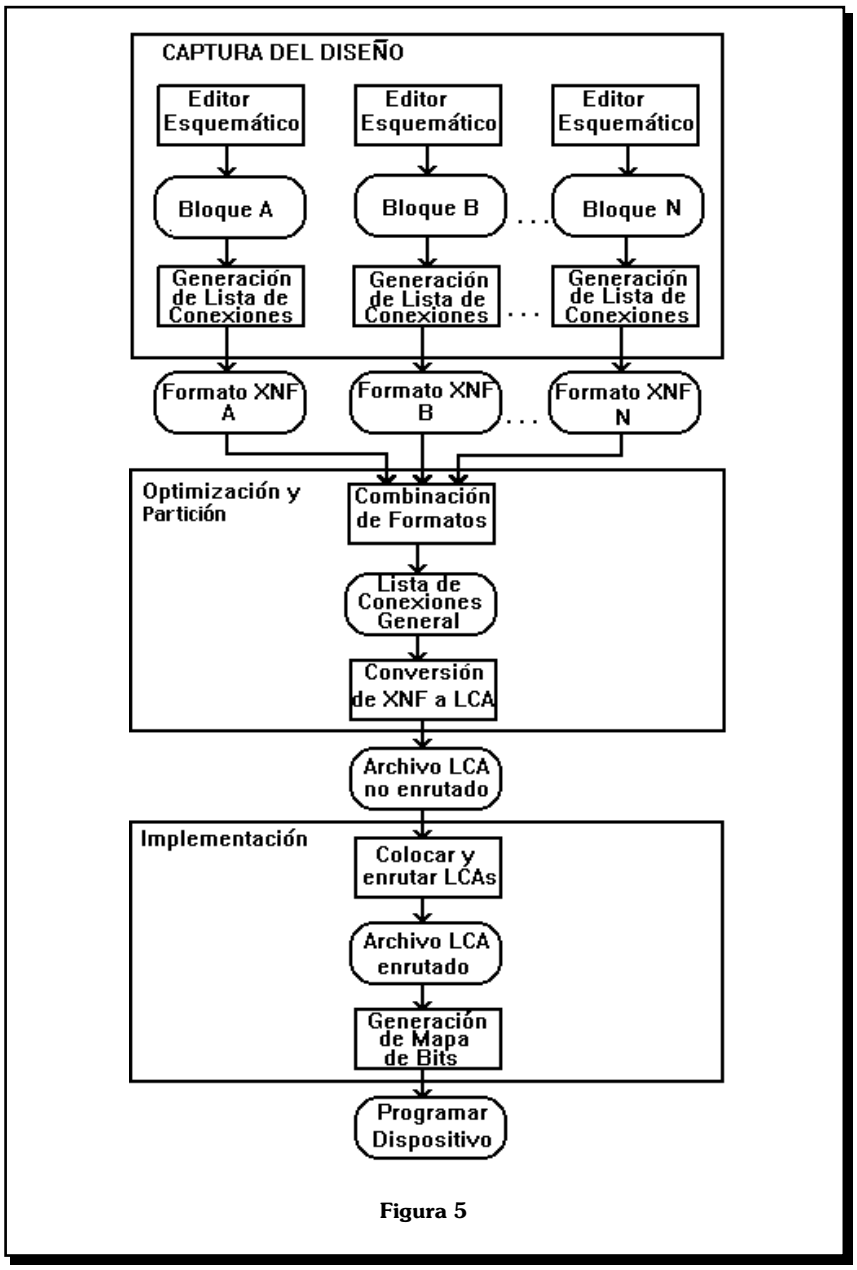


**Figura 5**

3.- Finalmente se pasa a la fase de implantación, en la cual el archivo de diseño para LCA se convierte en un conjunto de bits para configuración del dispositivo seleccionado.

La parte más importante en esta etapa es la colocación y enrutamiento de los elementos del circuito en el dispositivo. Para llevar a cabo la verificación final se cuenta con herramientas de simulación para comprobar tanto la operación del circuito LCA como su conexión física.

Como ya se mencionó, la primera etapa se puede desarrollar por medio de herramientas de propósito general, pero la segunda y tercera etapas (optimización e implementación) requieren manejar programación especial. Una vez descrito el proceso general, se describirá en detalle una aplicación real.

## *Descripción del Diseño*

Para la descripción del diseño se pueden emplear herramientas para la generación de esquemáticos, tales como Orcad SDT, HiWire, y FutureNET, o bien programas que manejen descripciones de comportamiento, tales como el OPAL y PALASM2. Como ejemplo se tomará el esquemático mostrado en la **figura 6 [2]**, en la que

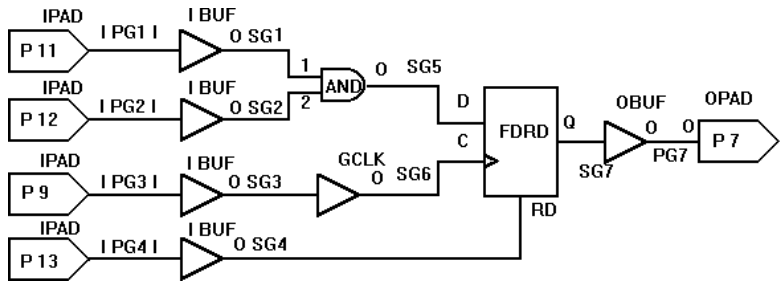


**Figura 6**

se tiene un flip-flop tipo D simple alimentado por una compuerta AND; como complemento, en el diagrama también se muestran los símbolos correspondientes al reloj interno del LCA y a los buffers de entrada y salida del mismo.

Una vez que se tiene el esquemático, se obtiene el archivo XNF. Para el ejemplo en desarrollo, dicho archivo es **[2]**:

```
LCANET, 1
PROG, PIN2XNF, 2.01, Created from ejemplo.pin
PART, 2064PC68-1
SYM, SG5, AND
PIN, O, O, SG5
PIN, 2, I, SG2
PIN, 1, I, SG1
END
SYM, SG1, IBUF
PIN, O, O, SG1
PIN, I, I, PG1
END
SYM, SG2, IBUF
PIN, O, O, SG2
PIN, I, I, PG2
END
SYM, SG4, IBUF
PIN, O, O, SG4
PIN, I, I, PG4
END
SYM, PG7, OBUF
PIN, O, O, PG7
PIN, I, I, SG7
END
SYM, SG3, IBUF
PIN, O, O, SG3
PIN, I, I, PG3
END
SYM, SG6, GCLK
PIN, O, O, SG6
PIN, I, I, SG3
END
SYM, SG7, DFF
PIN, RD, I, SG4
PIN, Q, O, SG7
PIN, D, I, SG5
PIN, C, I, SG6
END
EXT, PG1, I, , LOC=P11, BLKNM=PG1
EXT, PG2, I, , LOC=P12, BLKNM=PG2
EXT, PG3, I, , LOC=P9, BLKNM=PG3
EXT, PG4, I, , LOC=P13, BLKNM=PG4
EXT, PG7, O, , LOC=P7, BLKNM=PG7
EOF
```

A partir del archivo XNF, la siguiente etapa contempla una optimización del mismo, empleando para ello las herramientas adecuadas tales como el optimizador de circuitos de Xilinx (XNFOPT). Dado que el ejemplo es bastante simple no se hace necesario realizar esta etapa.

Posteriormente el archivo optimizado se convierte a un formato intermedio, conocido como MAP. Este archivo representa un circuito seccionado, donde cada una de las secciones corresponde a un CLB o a un IOB del dispositivo a emplear. Una vez que se obtiene el archivo MAP este puede convertirse a un archivo de diseño LCA. El archivo LCA no es otra cosa mas que una lista de instrucciones que dirigen al Xact durante los procesos de colocación y enrutamiento sobre el dispositivo físico. El siguiente listado muestra el archivo LCA del ejemplo **[2]**.

```
;: small.lca, MAP2LCA 2.20
Design 2064PC68
Speed -1
Addnet SG7  AA.X P7.O
Addnet SG6  AA.K CLK.AA.O
Addnet SG4  AA.D P13.I
Addnet SG3  P9.I CLK.AA.I
Addnet SG2  AA.A P12.I
Addnet SG1  AA.B P11.I
Nameblk AA SG7
Editblk AA
BASE FG
CONFIG X:Q Y:  Q:FF SET:
RES:D CLK:K:  F:A:B
EQUATE F=(A*B)
Endblk
Nameblk P9 PG3
Editblk P9
BASE IO
CONFIG I:PAD BUF:
Endblk
NameblkP7 PG7
Editblk P7
BASE IO
CONFIG I: BUF:ON
Endblk
Nameblk P13 PG4
Editblk P13
BASE IO
CONFIG I:PAD BUF:
Endblk
Nameblk P12 PG2
Editblk P12
BASE IO
CONFIG I:PAD BUF:
Endblk
Nameblk P11 PG1
Editblk P11
BASE IO
CONFIG I:PAD BUF:
Endblk
```

### *Implantación final*

El archivo de diseño LCA se alimenta al programa APR ("Automatic Place and Route"), generándose así un nuevo archivo que incluye la colocación de los bloques lógicos y el enrutamiento de señales. Como su nombre lo indica, el APR calcula automáticamente la colocación y el enrutamiento, y si estos son correctos se obtiene otro archivo formado por un patrón de bits para programar al dispositivo; en caso de que se detecten errores en el archivo de diseño, Xact permite su edición para corregir las deficiencias.

```
:  0200  0002  0000  FC
:  1000  0000  FF04  400F  FBEE  FFFF  FFFF  FFFF  FFFF  B7FD  09
:  1000  1000  FDFB  FFFB  F7F7  FFBF  FFFF  FFFF  FFFF  FFFF  4A
:  1000  2000  FFFD  FFFF  FFFF  FFFF  FFFF  EFFF  FDFB  FBFB  00
:  1000  3000  F7F7  F77F  FFFF  FFFF  FFFF  FFFF  FFFB  FFFF  6C
:  1000  4000  FFFF  FFFF  FFFF  DFFF  FFFF  FFFF  FFFF  FFFF  E0
:  1000  5000  FEFF  FFFF  FFFF  FFFF  FFF7  FFFF  FFFF  FFFF  B9
:  1000  6000  FFFF  BFFF  FFFF  FFFF  FFFF  FFFF  FDFF  FFFF  E2
:  1000  7000  FFFF  FFFF  FFEF  FFFF  FFFF  FFFF  FFFF  7FFF  20
:  1000  8000  FFFF  FFFF  FFFF  FFFF  FBFF  FFFF  FFFF  FFFF  84
:  1000  9000  FFDF  FFFF  FFFF  FFFF  FFFF  FFFE  FFFF  FFFF  91
:  1000  A000  FFFF  FFFF  F7FF  FFFF  FFFF  FFFF  FF3F  FFFF  28
:  1000  B000  FFFF  FFFF  FFFF  DFFD  FFFF  FFFF  FFFF  FFFF  72
:  1000  C000  EFFF  FFFF  FFFF  FFFF  FF7F  FFFF  FFFF  FFFF  D0
:  1000  D000  FFFF  FFFB  FFFF  FFFF  FFFF  FFFF  9FFF  FFFF  94
:  1000  E000  FFFF  FFFF  FFEF  FCFF  FFFF  FFFF  FFFF  7FF7  BB
:  1000  F000  FFFF  FFFF  FFFF  FFFF  BFFF  FFFF  FFFF  FFFF  50
```

Contando con el patrón de bits, la configuración del LCA no presenta mayores dificultades; únicamente debe tomarse en cuenta que los LCA son dispositivos de tipo volátil, por lo que si se desea dejar permanente la programación es necesario almacenar ésta en una memoria ROM conectada al LCA en cuestión. A continuación se muestra el archivo del patrón de bits del ejemplo en formato codificado para una memoria ROM **[2]**.

## Bibliografía


**[1]** González Robles Juan C., Vega Alvarado Eduardo. "Arreglos de compuertas programables en campo, FPGA's. Revista POLIBITS No. 12, Enero-Marzo 1994.

**[2]** David Pellerin/Michael Holley. PRACTICAL DESIGN USING PROGRAMMABLE LOGIC. . Prentice Hall, 1991.

**[3]** Edwards J. Mc. Cluskey. LOGIC DESIGN PRINCIPLES WHIT EMPHASIS ON TESTABLE SEMICUSTOM CIRCUITS. Prentice Hall, 1986.

**[4]** DESIGNING WHIT HIGH PERFORMANCE ASICS. Di Giacomo. Prentice Hall, 1991.