38

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES
CUAUTITLAN

ARQUITECTURA DE LOS FPGA's Y APLICACIONES PARA USO DIDACTICO EN LOS LABORATORIOS DE ELECTRONICA.

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO MECANICO ELECTRICISTA

P R E S E N T A :

FERNANDO GARCIA RODRIGUEZ

ASESOR: ING. ADRIANA SANDOVAL GARCIA

CUAUTITLAN IZCALLI, EDO. DE MEX. 2000

VNIVERSIDAD NACIONAL
AVTNOMA DE
MEXICO

FACULTAD DE ESTUDIOS SUPERIORES CUAUTITLAN
UNIDAD DE LA ADMINISTRACION ESCOLAR
DEPARTAMENTO DE EXAMENES PROFESIONALES

ASUNTO: VOTOS APROBATORIOS

DR. JUAN ANTONIO MONTARAZ CRESPO
DIRECTOR DE LA FES CUAUTITLAN
P R E S E N T E

ATN: Q. Ma. del Carmen García Mijares
Jefe del Departamento de Exámenes
Profesionales de la FES Cuautitlán

Con base en el art. 28 del Reglamento General de Exámenes, nos permitimos comunicar a usted que revisamos la TESIS: "Arquitectura de los FPGA's y Aplicaciones para uso didáctico en los Laboratorios de Electrónica"

que presenta el pasante: Fernando García Rodríguez
con número de cuenta: 9137604-8 para obtener el TITULO de:
Ingeniero Mecánico Electricista.

Considerando que dicha tesis reúne los requisitos necesarios para ser discutida en el EXAMEN PROFESIONAL correspondiente, otorgamos nuestro VOTO APROBATORIO

A T E N T A M E N T E.
"POR MI RAZA HABLARÁ EL ESPÍRITU"
Cuautitlán Izcalli, Edo. de Méx., a 17 de Mayo de 2000

PRESIDENTE	Ing.Nicolás Calva Tapia	
VOCAL	Ing.Petra Medel Ortega	
SECRETARIO	Ing.Adriana Sandoval García	
PRIMER SUPLENTE	Ing.Blanca G.De la Peña Valencia	
SEGUNDO SUPLENTE	Ing.Jorge Buendía Gómez	

DEDICATORIA.

A mi papá

Fernando García Hernández.

Y a mi mamá

Clotilde Rodríguez González.

Sabiendo que no existe en esta vida la forma de agradecer su gran cariño, apoyo y comprensión.

AGRADECIMIENTOS.

A la Ing. Corina Adriana Sandoval García
por su amistad, conocimientos y apoyo
brindado.

Al Ing. Jorgue Buendía Gómez
por su gran apoyo.

A:
Gabriela García Rodríguez
Ricardo Fagoaga Vazques
por su comprensión y sabios consejos.

A:
Humberto López
Rebeca Rodríguez de López
Daniel Fabila
M. de J. Silvia García de Fabila
por su gran ayuda y compresión.

A todos mis primos y amigos
por su ayuda.

A todos ellos, Gracias.

INDICE

INTRODUCCION

Los circuitos digitales programables por el usuario ocupan un lugar muy importante en los sistemas electrónicos modernos. Se encuentran, por supuesto, en forma de memorias en los sistemas basados en microprocesadores; pero también son muy utilizados como circuitos lógicos de un solo chip, desplazando a los antiguos módulos que en el pasado utilizaban decenas y hasta centenas de componentes lógicos convencionales.

La propuesta de los fabricantes de CI's en este ámbito es amplia y muy diversa, y se ha pasado de chip's que solo contenían algunas decenas de compuertas (o menos) a circuitos que integran sin problema miles de compuertas lógicas. Sin embargo, a pesar de esto, aún quedan circuitos programables para sistemas de desarrollo simples, fáciles de usar y al alcance de todos.

Por otro lado, estos dispositivos están disponibles en distintas tecnologías que cubren una amplia gama de necesidades: circuitos programables una sola vez, circuitos programables eléctricamente y borrables por rayos ultravioletas, circuitos programables y borrables eléctricamente, así como circuitos cuya configuración inicial puede ser modificada en el momento de ser utilizados en el propio sistema.

Dentro de esta amplia gama de dispositivos lógicos se tiene como preámbulo, primeramente, a las memorias programables designadas genéricamente como ROM (y todas sus variantes de arquitectura como lo son PROM, EPROM, EEPROM, etc.).

Posteriormente, los dispositivos lógicos programables se clasificaron de una manera un poco más delicada que la de las ROM; debido a que los fabricantes de estos circuitos han llegado a escoger nombres para sus productos que no siempre se ajustan a una lógica o tecnología de fabricación rigurosamente.

Los dispositivos FPLA's (Arreglos Lógicos Programables de Campo) fueron los siguientes en hacer su aparición, pero no tuvieron gran éxito ya que para muchos diseñadores eran difíciles de usar

debido a la nueva tecnología de programación, y por lo tanto eran poco fiables. Sin embargo, los circuitos lógicos programables realmente más conocidos y generalizados se llaman PAL (Lógica de Arreglo Programable) de posterior aparición; cuyo nombre se trata de una marca registrada por su inventor (MMI, actualmente AMD). Estos circuitos son programables por fusibles, como las PROM. Los PAL's se dividen en dos tipos: simples, que son en realidad los PAL's que se acaban de mencionar, y los de registro que contienen registros en su arquitectura y por lo tanto pueden hacer intervenir en su funcionamiento la noción de tiempo.

Mientras que los circuitos que se han mencionado hasta este momento existen desde hace veinte o veinticinco años, en algunos casos, avanzando en el recorrido por los circuitos lógicos ahora tenemos a las GAL's (Lógica de Arreglo Genérico) de aparición posterior a los PAL's las cuales son de tecnología mucho más reciente ya que entraron al mercado hace 10 años aproximadamente. Fueron comercializadas por primera vez por Lattice Semiconductor que registró la marca. Las GAL's en realidad son PAL's de tecnología CMOS programables y borrables eléctricamente.

A continuación de estas y tan solo unos años después, aparecen los primeros CPLD's (Dispositivos Lógicos Programables Complejos) que no son más que otra variedad más de las PAL CMOS pero la estructura interna de estos y los tipos de macroceldas y redes de interconexión que se emplean aquí son completamente diferentes.

Actualmente, mientras los CPLD's más grandes disponibles integran por el orden de las 6,400 compuertas útiles, los FPGA's (Arreglos de Compuestas Programables de Campo) de reciente aparición que serán abordados como tema de investigación en esta tesis pueden alcanzar hasta las 130,000 compuertas útiles.

A pesar de esta extrema densidad de integración los FPGA's siguen siendo circuitos programables por el usuario con medios técnicos al alcance de un laboratorio de electrónica digital o de una pequeña empresa.

Existen actualmente dos tecnologías concurrentes de FPGA's. Estas dos permiten que el mismo usuario programe las redes de compuertas integradas, pero con métodos diferentes. Estos son, los FPGA's de tipo RAM que se basan en circuitos programables y borrables eléctricamente por el usuario, y los FPGA's de tipo PROM que se basan en circuitos programables por el usuario, pero no son borrables. Para este caso serán estudiados los de tipo RAM.

Existen diversos fabricantes de FPGA's pero específicamente, serán estudiados los FPGA's de la Serie XC4000 de Xilinx los cuales pertenecen a la Tercera Generación de Arreglos de Compuertas Programables de Campo.

Para ello, en el primer capítulo se hará un recorrido más amplio a través de la historia de los PLD's; y posteriormente en este mismo, serán establecidas las principales características arquitectónicas de los FPGA's de la Serie XC4000 de Xilinx, así como de sus antecesores los CPLD's (específicamente, la Serie XC9500 también de Xilinx), con el fin de tener una idea más clara de lo que son estos dispositivos.

A continuación, en el segundo capítulo, será analizada en detalle la arquitectura de los CPLD's de la Serie XC9500; ya que las nociones aquí presentadas resultan de gran ayuda para así comprender mejor la arquitectura de los FPGA's. Enseguida, en el tercer capítulo, se presenta la arquitectura de estos (los FPGA's de la Serie XC4000) de una manera mucho más completa, con lo cual se tendrá un panorama general de su estructura y funcionamiento.

Posteriormente, en el capítulo siguiente, se darán los métodos de programación más comunes para poder implementar algunos diseños con estos circuitos a fin de comprender de forma práctica los métodos de programación y su funcionamiento; así como del proceso de implementación del diseño.

Todo lo anterior, con el objetivo de aportar un material de apoyo didáctico a los Laboratorios de Electrónica en su sección de Electrónica Digital.

CAPITULO 1

Dispositivos Lógicos Programables.

1.1 INTRODUCCION.

Los circuitos digitales programables por el usuario ocupan un lugar primordial en los sistemas electrónicos modernos, por lo que es importante saber cual fue el origen de los primeros dispositivos y como estos han evolucionado a través de los años hasta la actualidad.

Ya que durante este proceso de desarrollo han intervenido diferentes fabricantes y diversas tecnologías asociadas, es básico establecer perfectamente cuales han sido los dispositivos desarrollados y los principales fabricantes de cada uno de estos. Para lo cual se hará un recorrido por la historia de los dispositivos lógicos programables (PLD's). Y a continuación, serán analizados en particular los dispositivos de Xilinx (CPLD's y FPGA's).

1.2 HISTORIA DE LOS PLD's.

A continuación se hará un recorrido a través de la historia de todos los PLD's existentes, partiendo de los dispositivos PROM (Memoria de Sólo Lectura Programable una sola vez), hasta llegar a los FPGA's (Arreglos de Compuertas Programables de Campo); actualmente los más recientes dispositivos de la Lógica Programable.

1.2.1 PROM's.

En general, dispositivos lógicos programables (PLD's) es un término genérico que se refiere a cualquier tipo de circuito integrado que pueda ser configurado por el usuario para la implementación de un diseño en particular. Desde que los PLD's se programan "en el campo" por

el usuario, estos dispositivos son también llamados dispositivos lógicos programables de campo (FPLD's). Los dispositivos programables han jugado un papel importante en el diseño de hardware digital ya que proveen una gran flexibilidad en cuanto a configuración por el usuario se refiere, para una amplia variedad de aplicaciones. Uno de los dispositivos lógicos programables más comunes fue la memoria de sólo lectura programable una sola vez (PROM).

Los primeros dispositivos PROM se desarrollaron por Harris en 1970. Las PROM's combinaron la tecnología del eslabón fusible de nicromo de Harris con la estructura de arreglo simplificado que se había vuelto popular en la forma de ROM's (Memoria de Sólo Lectura) programables por máscara. Estos dispositivos eran útiles y funcionales para una variedad de programas y almacenamiento de datos, por lo que fueron aceptados rápidamente por los diseñadores de circuitos. En unos cuantos años, las PROM's se apoderaron de la popularidad de las ROM's enmascaradas, y muchos de los fabricantes de CI's empezaron a producir los nuevos dispositivos en una amplia gama de formas y tamaños.

Las primeras PROM's tenían una matriz de fusibles. Una matriz de fusibles consiste en puentes de metal propiamente dicho que se conectan al interceptarse con las señales en la matriz. Programar esta matriz implica elevar a un voltaje específico (más alto de lo normal) ciertos pines de entrada / salida, mientras se aplica una dirección e información (datos) a otros pines de entrada / salida. Cuando una secuencia específica de entradas y voltajes se aplica a los pines de un dispositivo, una corriente elevada fluye como consecuencia por el fusible designado y lo funde.

Estos patrones específicos de formas de onda de voltaje, llamados algoritmos de programación, son únicos en cada dispositivo. Puesto que cada dispositivo programable tiene requisitos únicos de programación, los usuarios de las primeras PROM's tenían que construir y mantener un circuito de programación diferente para cada dispositivo que se quisiera usar.

Los libros de datos de las primeras PROM's incluyeron descripciones de cómo construir los circuitos para los programadores, puesto que no existía ningún programador de dispositivos comercialmente disponible. El uso de estos programadores construidos por el usuario daban por resultado índices de programación pobres y dispositivos programados poco confiables.

A principios de 1970, había un número creciente de fabricantes de dispositivos PROM: Intel, Intersil, Harris, MMI (Monolitics Memories Inc.), etc. Cada uno de estos fabricantes ofrecía una gran lista de estos dispositivos. Intel introdujo una PROM borrable por luz ultravioleta (UV) en 1971, acelerando el crecimiento industrial todavía más.

1.2.2 FPLA's.

Intersil Corporation y Signetics Corporation, ambos fabricantes de PROM's, comprendieron que la estructura del dispositivo DM 7575 / DM8575 PLA programado por máscara de National era ideal como dispositivo programable de campo. Por lo que entablaron una carrera por ver quien introducía los primeros dispositivos en el mercado. Las dos compañías llamaron a sus nuevos dispositivos propuestos FPLA's (Arreglos Lógicos Programables de Campo)

El 2 de Junio de 1975, en la edición del EE Times, Intersil anunció su FPLA IM5200. Poco después, Signetics introdujo su 82S100. Estos dispositivos se adelantaron a su tiempo en muchos aspectos.

Estos primeros dispositivos FPLA's fueron bastante poderosos (los 82S100 todavía están disponibles, bajo el nuevo nombre de PLS100), debido a que proveyeron un número razonable de entradas y salidas. Eran bastante flexibles en su diseño para ser usados en una amplia variedad de aplicaciones.

Mientras que a simple vista los IM5200 y los 82S100 parecían muy similares, eran de hecho bastante diferentes en términos de su tecnología de programación. En lugar de usar la tecnología de eslabón fusible bien conocida para su nuevo dispositivo, Intersil escogió utilizar un nuevo tipo de elemento programable que había desarrollado para sus dispositivos PROM. Este elemento de programación fue diseñado para mejorar los rendimientos de programación por encima de los primeros dispositivos programables. La tecnología, llamada de migración inducida por avalancha, o AIM, utiliza una base abierta de un transistor NPN como elemento de programación. Para programar un elemento AIM, una corriente elevada se forza a pasar a través del transistor de emisor a colector. Esto causa un corto del emisor a la base que deja al transistor operando como un diodo.

Desgraciadamente, los dispositivos reales que salieron fuera de la fábrica de Intersil resultaron no ser confiables, con bajos rendimientos de programación, y no tuvieron éxito en el mercado.
El 82S100 de Signetics utilizó el eslabón fusible como tecnología de programación puesto que era más fiable y más exitosa en el mercado. Otro factor en el éxito de los dispositivos de Signetics eran los esfuerzos de Napoleone Cavian que estaba en ese momento como Gerente de Productos Avanzados de Signetics. Los nuevos dispositivos eran nada familiares a la mayoría de los diseñadores de circuitos y requerían un nivel mucho más alto de educación del usuario y promoción, a diferencia de los dispositivos antiguos.

Inclusive con un nivel de programación alto, documentación y soporte de aplicaciones, muchos de los diseñadores no quisieron usar estos PLD's, porque los dispositivos todavía eran percibidos como demasiado difíciles de usar. Incluso con la fiabilidad razonable de los dispositivos de Signetics, el FPLA tenía una velocidad de operación máxima relativamente lenta (debido a los dos arreglos programables), era caro, y tenía una pobre reputación para las pruebas. Otro factor que limitaba la aceptación de los FPLA era el encapsulado grande (28 pines con más de media pulgada de ancho).

Aunque Intersil y Signetics fueron las primeras compañías en comercializar con relativo éxito los FPLA's, esto no significa que fueron los primeros que los desarrollaron. Ya en 1971, GE (General Electric) estaba desarrollando un dispositivo lógico programable basado en la tecnología PROM. El dispositivo era desarrollado por David Greer. El dispositivo experimental de GE tenía mejoras sobre la estructura ROAM de IBM, proporcionando una trayectoria interna a las señales del plano OR para volver a entrar al plano AND directamente. Esto permitió el uso de lógica multinivel sin pérdida de pines de entrada o salida.

A fines de 1971, se completó un dispositivo lógico programable MOS experimental en el Centro de Investigación y Desarrollo de General Electric en Schenectady, Ny por Gerry Michon y Hugh Burke. El dispositivo no sólo tenía las mejoras de la lógica de los arreglos tipo FPLA mejorada, sino también utilizó tecnología de compuertas flotantes borrables por UV anunciada unos años antes por Intel. El dispositivo de GE realmente fue alguna vez el primer PLD borrable desarrollado, antecediendo a los PLD's disponibles comercialmente por más de una década. Los investigadores en GE no sólo desarrollaron los primeros FPLA's y EPLD's funcionando, sino también describieron y patentaron una estructura de arreglos plegados notablemente similar a los arreglos plegados que aparecieron después en PLD's complejos quince años más tarde.

En 1974, bajo los términos de una patente y acuerdos secretos de comercio con GE, la compañía MMI comenzó el desarrollo de un dispositivo lógico programable por máscara que incorporaría las innovaciones de GE. El dispositivo (MMI número de parte 5760 / 6760) se nombró como arreglo lógico asociado programable o PALA. El dispositivo se completó en 1976, y podía llevar a cabo circuitos secuenciales o multinivel de más de 100 compuertas equivalentes. Además, el dispositivo fue apoyado por un ambiente de diseño muy automatizado desarrollado por GE.

1.2.3 PAL's.

A pesar de que los dispositivos de Signetics tuvieron algún éxito, realmente no ganaron gran aceptación sino hasta fines de los años 70's, cuando MMI introdujo el dispositivo PAL. Después de trabajar con GE en el dispositivo PALA el primer esfuerzo de MMI para el diseño de sus dispositivos fue reproducir los FPLA's de Intersil y Signetics. Se produjeron algunos cientos de copias del Signetics 82S100 para la evaluación interna, pero estos dispositivos nunca fueron liberados. De esta experiencia, y la primera experiencia de MMI con GE, el dispositivo PAL nació. La nueva familia de dispositivos se anunció en el verano de 1978.

El proyecto para crear el dispositivo PAL fue manejado por John Birkner y el circuito del PAL propiamente dicho fue diseñado por H. T. Chua. Birkner provenía de Computer Automation, Inc., donde él había desarrollado un procesador de 16 bits que usaba 80 dispositivos de la lógica normal. Su experiencia con lógica normal lo llevó a creer que los dispositivos programables por el usuario serían más atractivos a los usuarios si estos fueran diseñados para reemplazar lógica normal. Esto significaba que los tamaños de encapsulado tenían que ser los típicos de los dispositivos existentes, y las velocidades tenían que ser mejoradas. Los nuevos dispositivos resultado de este pensamiento fueron un descubrimiento y un gran éxito en el mercado.

Los dispositivos PAL utilizaron la tecnología de fusibles PROM ahora madura y bastante fiable, los cuales contenían un solo arreglo programable. Esta combinación producía un dispositivo con funcionamiento mucho más rápido que los primeros FPLA's La programación de los dispositivos era simple puesto que estos cumplían con normas de la industria en sus encapsulados y empleaban la tecnología de fusibles PROM bien conocida.

Un factor en el éxito de los dispositivos PAL era el alto nivel de apoyo al cliente ofrecido por MMI en forma de aplicaciones y documentación para el usuario, que sirvieron para aclarar el proceso de diseño. El manual de los dispositivos PAL escrito por el propio Birkner, proporcionó un puente conceptual entre los métodos de la lógica discreta del pasado y los métodos de alto nivel del futuro.

Otro factor en el éxito de los dispositivos PAL fue PALASM que significa ensamblador PAL (PAL Assambler) que es un ensamblador simbólico. PALASM era un programa para computadora escrito por John Birkner que convertía descripciones del diseño compuestas de ecuaciones Booleanas directamente en datos para la programación de un dispositivo PAL específico.

Se han usado ecuaciones Booleanas desde los primeros días del diseño lógico (mucho tiempo antes de que el circuito integrado se concibiera) para expresar funciones lógicas. El álgebra Booleana está en el núcleo de cualquier curso de lógica digital, de manera que es una forma de representación con la que están más familiarizados los usuarios en vez de las formas tabulares.

El PALASM era un programa de computadora simple escrito en FORTRAN. El código fuente del programa entero requirió sólo seis páginas de FORTRAN y se publicó en el libro de datos de los dispositivos PAL.

La simplicidad del lenguaje PALASM hizo posible que los fabricantes de programadores implementaran el lenguaje directamente en el hardware para programación, como se hizo en el Data I / O LogicPak, el programador de Diseño Estructurado, y otros programadores para PAL similares.

Una indicación del éxito del PAL es que, diez años después de la introducción de los primeros PAL's 16L8 y 16R8, estos dispositivos constituyeron la mayoria de todos los PLD's usados aunque había más de 200 tipos de dispositivos PLD disponibles en ese momento.

Otro factor en el éxito de los PAL fue el nivel de apoyo para la programación. El primer FPLA's había padecido de los mismos problemas de programación que atormentaron a las primeras PROM's; los programadores construidos por los usuarios no confiables y los algoritmos de programación poco comprensibles.

John Birkner y su equipo determinaron hacer un éxito a los PAL's, de modo que trabajaron muy estrechamente con compañías de programadores para asegurar una programación confiable garantizada a los usuarios del dispositivo. El primer programador de PAL's se desarrolló como un esfuerzo de la acción colectiva entre MMI y Data I / O, y de hecho se utilizaron dos PAL's como parte de su construcción. Esto significo que el primer programador prototipo de PAL's tenía que ser inicializado emulando la función de sus propios PAL's con PROM's y algunos dispositivos TTL adicionales.

Por sus esfuerzos, MMI premió a John Birkner y H. T. Chua con una paga extraordinaria, y automóviles nuevos durante todos los años siguientes.

Posteriormente, el apoyo para los programadores fue mejorado dramáticamente con la introducción de JEDEC estándar 3. Esta norma proporciona un formato de intercambio de datos

común a los usuarios para la programación del dispositivo. El formato estándar se propuso por primera vez en 1980, y define un formato de archivo de computadora que hace posible transferir diseños y datos de prueba entre dispositivos, y programadores de dispositivo de diferentes fabricantes.

Después de que MMI abrió camino y patentó el PAL, otros fabricantes de dispositivos empezaron o cambiaron a dispositivos tipo PAL bajo la licencia de MMI, o bien con bastantes cambios que permitieran exigir la propiedad de los diseños. De esta manera, el PAL se volvió la base para las nomenclaturas de nuevos dispositivos y literalmente docenas de juicios y amparos por infracciones de patente (se ha dejado ver que, por algunos años, los abogados de Silicon Valley ganaron tanto dinero de la patente del PAL como la compañía que lo originó). El más grande de estos competidores era Advanced Micro Devices. AMD finalmente adquirió a MMI y las dos líneas de productos fueron unidas.

Mientras el PAL tenía marca registrada y estaba patentado, el PALASM se colocó en el dominio público y prosperó en muchos sistemas de computadoras diferentes. Además de la falta de soporte universal a dispositivos PLD's. PALASM padeció de una completa falta de flexibilidad en métodos de captura de diseños.

Los usuarios estaban exigiendo un ambiente de diseño mejor, uno que apoyara a todos los PLD's disponibles hasta ese momento y permitiera más flexibilidad en la especificación y procesado de diseños secuenciales grandes y complejos típicos de los más nuevos dispositivos. La investigación de dicha herramienta de diseño empezó con Data I / O en la primavera de 1981. Inicialmente, los únicos resultados de esta investigación eran algunos vagos requisitos de producto y la selección de un nombre para el producto, ABEL (Lenguaje de Expresión Booleana Avanzado).

El proyecto de ABEL se anunció a varios fabricantes de PLD's en 1982, y fue bien recibido. Y en ese momento, los líderes de proyecto de Data I / O se encuentran con Assisted Technology, y con la ayuda de muchos otros, comienzan a trabajar en un producto de software llamado CUPL (o la herramienta universal común para la lógica programable). En marzo de 1983, Assisted technology liberó la versión beta de CUPL y a fines de este mismo año, Data I / O liberó la versión beta de ABEL. Ambas soportaban casi a todos los dispositivos disponibles de ese momento.

1.2.4 GAL's.

Después del tremendo éxito de los PAL, los nuevos esfuerzos de los dispositivos de MMI se apuntaron principalmente a reforzar y ajustar al dispositivo PAL básico. El ejemplo más notable de este hecho fue el dispositivo MegaPAL 64R32. Este enorme dispositivo (84 pines) era el equivalente a cuatro PAL's normales y consumía una cantidad grande de energía.

Entretanto, AMD produjo una serie de dispositivos PAL que extendieron la estructura PAL básica proporcionando al usuario características de configuración adicionales, como la programación de las trayectorias de los registros y la realimentación de los 22V10. Esta tendencia continuó con nuevos dispositivos que generalmente no se distinguieron por la reducción de pines o tamaños del encapsulado, sino por la configuración y programabilidad de la circuitería. Durante varios años, la tecnología en uso predominante para estos dispositivos seguía siendo la tecnología PROM bipolar, hasta que la revolución de los CMOS alcanza a los PLD's.

La entrada de Altera en el negocio de la lógica programable en 1984 trajo un nuevo nivel de sofisticación a los dispositivos y a las herramientas de diseño. Antes de Altera, los fabricantes de dispositivos generalmente estaban satisfechos con permitir que otras compañías cuidaran del diseño y comercialización del hardware de programación y de las herramientas de diseño avanzadas para PLD. Algunos fabricantes del dispositivo, de hecho, estaban considerando suspender el apoyo de sus propias herramientas de diseño. Altera, sin embargo, estaba desarrollando una serie de dispositivos completamente nueva que era, en la opinión de Altera, de suficiente complejidad para lo cual muchas de las herramientas de diseño deberían de requerir herramientas diseñadas específicamente para esos dispositivos.

Al contrario de los PLD's anteriores, los dispositivos de Altera estaban basados en la tecnología de EPROM CMOS. La naturaleza borrable de estos dispositivos los hizo ideales para los ambientes de diseño y también útil en el ambiente de producción debido a su incrementado nivel de prueba inherente. El uso de tecnología CMOS le permitió a Altera producir dispositivos de muy alta densidad y complejidad, este aumento en la complejidad hizo impráctico el uso de técnicas de diseño tradicionales, más que cualquier tipo anterior de PLD. Los usuarios de los dispositivos de Altera ciertamente requerían herramientas de diseño de alto nivel.

Además de Altera, muchas otras pequeñas compañías de dispositivos se han unido en la pelea. Estos pequeños diseñadores de dispositivos han producido PLD's basados en CMOS, y a menudo

confían en suministrar la herramienta de apoyo para sus PLD's como una forma de asegurar el éxito de sus dispositivos.

Una de estas compañías fue Cypress Semiconductor. Los primeros dispositivos producidos por Cypress fueron las versiones CMOS de alta velocidad de las populares PAL's producidas bajo un acuerdo con MMI.

Otra compañía que se especializa en la tecnología CMOS borrable fue Lattice semiconductor que se fundó en 1983. Lattice desarrollo una familia de dispositivos de tipo PAL eléctricamente borrables que llamó Lógica de Arreglo Genérico, o dispositivos GAL. Estos dispositivos iniciaron una cadena de pleitos en la que MMI casi arruina a Lattice. Cuando la tormenta se había disipado, MMI tenía el derecho de producir los dispositivos GAL (aunque no bajo las siglas de GAL). Después de sobrevivir los pleitos de MMI, Lattice siguió su desarrollo y liberó un PLD más avanzado, que se situó por arriba de la estructura PAL normal.

Dos de los jugadores importantes de Lattice fueron Dave Rutledge y Dean Suhr, ambos habían trabajado previamente para Harris. Harris Semiconductor había trabajado estrechamente con IBM en la primera investigación sobre PLD's pero no habían llevado a cabo un PLD como producto de línea en los primeros días de los PDL's. Si la primera investigación de Harris / IBM hubiera sido publicada, muchos de los dolores de cabeza de los pleitos de Lattice contra MMI se podrían haber evitado.

Desde el principio de su existencia, Lattice trabajó estrechamente con proveedores de herramientas de diseño PLD para asegurar que sus dispositivos fueran soportados totalmente por software en el momento de que los dispositivos estuvieran disponibles a los clientes. Un disco tutorial producido como un esfuerzo compartido entre Lattice y Assisted Technology ayudó a educar a los nuevos usuarios de PLD's sobre los beneficios de los dispositivos y técnicas de diseño de alto nivel.

1.2.5 CPLD's.

Al poco tiempo, Xilinx Corporation anunció que ellos liberarían pronto un concepto completamente nuevo en dispositivos programables por el usuario. Los dispositivos de Xilinx, Dispositivos Lógicos Programables Complejos (CPLD's), eran una desviación radical respecto a los primeros CI

programables ya que la estructura interna de los CPLD's y los tipos de macroceldas y redes de interconexión que se emplean aquí son totalmente diferentes.

La arquitectura de estos circuitos hace el proceso de diseño para estos dispositivos bastante diferente al de los PLD's anteriores, de manera que Xilinx también creó un juego de herramientas de diseño para los usuarios. Estas herramientas de diseño incluyen la colocación automática y funciones de asignación de ruta que ayudan a utilizar el dispositivo eficazmente.

Otra compañía pequeña cuyos dispositivos tienen requisitos únicos de Software es Exel Microelectronics, fundada en 1983. Esta compañía desarrolló un dispositivo que sólo contenía un arreglo plegado que permite llevar a cabo diseños lógicos de múltiples niveles. Este dispositivo requería de las capacidades de herramientas de diseño no disponibles al momento de su desarrollo. La mejor solución al dilema de apoyo de software para Exel era producir un juego de programas de software que pudiera agregarse al producto de ABEL y por consiguiente darle la capacidad a ABEL para apoyar al nuevo dispositivo.

Altera también lanzo una serie de productos CPLD's los cuales estuvieron evidentemente soportados por cierto número de programas estándar. Sin embargo, parte de la fuerza de los productos de Altera fue la comercialización conjunta de un software de desarrollo particularmente flexible y potente. Este software ha evolucionado considerablemente y existen numerosas versiones funcionando en los distintos tipos de computadoras.

1.2.6 FPGA's.

Al momento del desarrollo de esta tesis, el segmento más reciente y dinámico del mercado de PLD's está en el área de los FPGA's (Arreglos de Compuertas Programables de Campo). Después de que Xilinx liberó sus primeros dispositivos FPGA's, otros fabricantes (la mayoría de ellos pequeñas compañías recién fundadas), empezaron a trabajar en otro tipo de FPGA's.

En 1988, Actel Corporation introdujo un FPGA muy diferente a los dispositivos de Xilinx. El ACT 1, así llamado, tiene una densidad que es comparable a los arreglos de compuertas programados por máscara y usa una tecnología de programación totalmente nueva. Como otros dispositivos, el dispositivo de Actel requiere la colocación y la asignación de ruta de las funciones lógicas para ser usado eficazmente. El software para realizar estas tareas se desarrolló en Actel en paralelo con el

desarrollo del dispositivo. En 1989, Plessey Semiconductor anunció que pronto estarían produciendo FPGA's.

Aun cuando todos los dispositivos FPGA actualmente disponibles o anunciados compartan algunos atributos similares, todos ellos tienen arquitecturas únicas, y son desviaciones mayores de los antiguos PLD's.

1.3 COMPARACION ENTRE LOS CPLD'S DE LA SERIE XC9500 Y LOS FPGA'S DE LA SERIE XC4000.

Después de haber realizado un recorrido a través de la historia de los dispositivos lógicos programables, se procederá a mostrar de una manera breve la arquitectura y características principales de los CPLD's de la Serie XC9500, así como la de los FPGA's de la Serie XC4000, ambos de Xilinx.

Lo anterior, con el propósito de establecer de una manera más precisa las principales características arquitectónicas, las ventajas, desventajas y diferencias entre unos y otros. Además, de resaltar las mejoras realizadas a los FPGA's los cuales son una generación más avanzada respecto a sus antecesores los CPLD's.

Para posteriormente ser analizadas y poder tener una noción más clara de lo que son este tipo de dispositivos (ya que estos varían desde la estructura interna, capacidad de diseño, hasta los grados físicos de integración) a partir de la comparación entre unos y otros. Y para finalizar, será establecida una comparación tomando en cuenta a los fabricantes de esta clase de dispositivos, con el objetivo de establecer cual es la mejor opción.

1.3.1 CPLD's de la Serie XC9500.

Los CPLD's de la Serie XC9500, que se analizarán en detalle en el Capítulo 2, son dispositivos constituidos por Bloques de Entrada / Salida (IOB's) que constan de la interfaz entre la lógica interna y los pines de I / O del usuario, Matrices de Interconexión y Bloques de Función (FB's) los cuales están compuestos de 18 macroceldas independientes, que en conjunto conceden la facilidad de implementar cualquier tipo de función lógica. Como se muestra en la Figura 1.1.

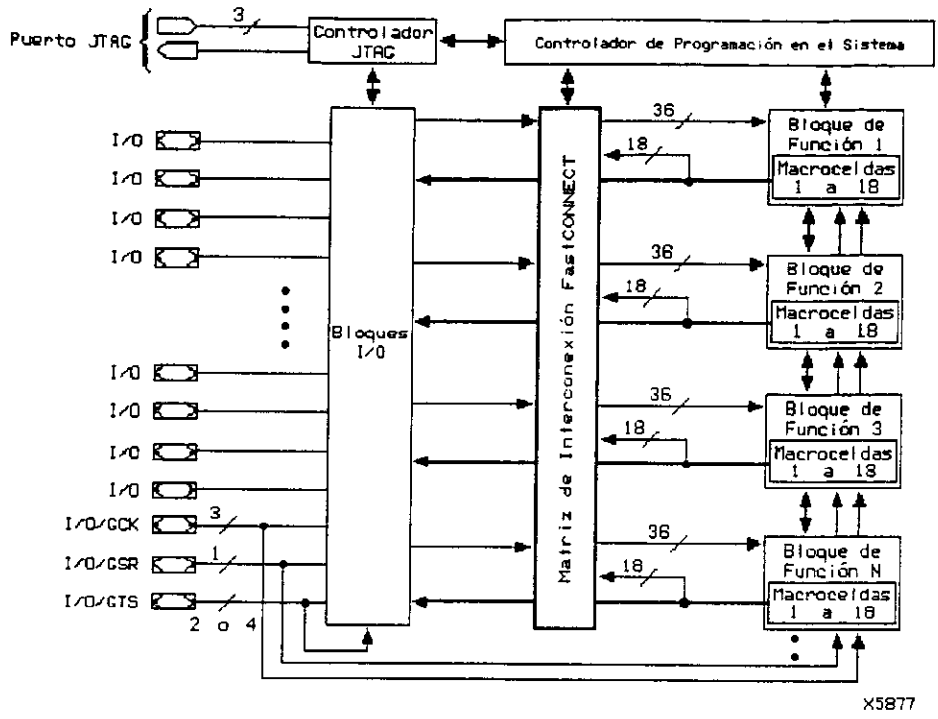


Figura 1.1: Arquitectura de los CPLD's de la Serie XC9500.

Todos los integrantes de esta serie de cuentan con las siguientes características:

- Programación en Sistema.
- Retraso pin – a – pin de 5 ns.
- Grado de integración que va desde las 36 hasta las 536 macroceldas con 800 a 12,800 compuertas lógicas utilizables.
- Un mínimo de 10,000 ciclos de programación y borrado
- Mejorada capacidad de asignación de pines durante los cambios de diseño.
- Soporte de prueba y desarrollo sobre los diseños implementados.
- Modo programable de bajo consumo de potencia.
- Salidas con control de "Slew – Rate".
- Pines programables por el usuario para tierra.
- Avanzada tecnología de FastFlash (5V. CMOS).

Los aspectos arquitectónicos de la familia XC9500 permiten a este tipo de dispositivos ser utilizados en aplicaciones tales como: sustitución de dispositivos PAL, o GAL, o combinación de PAL o de GAL, por un CPLD, Decodificadores / Codificadores, Tablas de búsqueda, y en general cualquier función de tipo combinacional o secuencial.

1.3.2 FPGA's de la Serie XC4000.

La Serie XC4000 de FPGA's de Xilinx, la cual se abordará con detalle en el Capítulo 3, consta de dispositivos desarrollados con una arquitectura programable que se compone de Bloques Lógicos Configurables (CLB's) cuyos elementos principales son Generadores de Función, Flip – Flops y Multiplexores, interconectados por poderosos recursos de enrutamiento, y rodeados por Bloques de Entrada / Salida (IOB's) que proporcionan la interfaz entre el paquete externo de pines y la lógica interna. La arquitectura de los dispositivos XC4000 se muestra en la Figura 1.2.

Las características principales de los FPGA's de la Serie XC4000 son las siguientes:

- Grado de integración que va desde las 2,000 hasta las 130,000 compuertas utilizables (Lógicas y RAM).
- Celdas de memoria RAM las cuales pueden activarse para leer o escribir datos de configuración en diversas formas.
- Generadores de Función con gran flexibilidad.
- Abundantes recursos de enrutamiento jerarquizados.
- Arquitectura de Arreglo Flexible.
- Características de Sistemas Orientados ("slew – rate" individualemte progragable, "pull – up" programable de entrada, etc.).
- Configuración mediante la carga de Archivo Binario.
- Reprogramabilidad ilimitada.
- Opción de reconfiguración ya en la práctica.
- Compatibilidad con los dispositivos de la Serie XC4000 anteriores.
- Software poderoso y sofisticado, que cubre todos los aspectos de diseño (ingreso del diseño, simulación, compilación, transmisión y prueba del mismo).

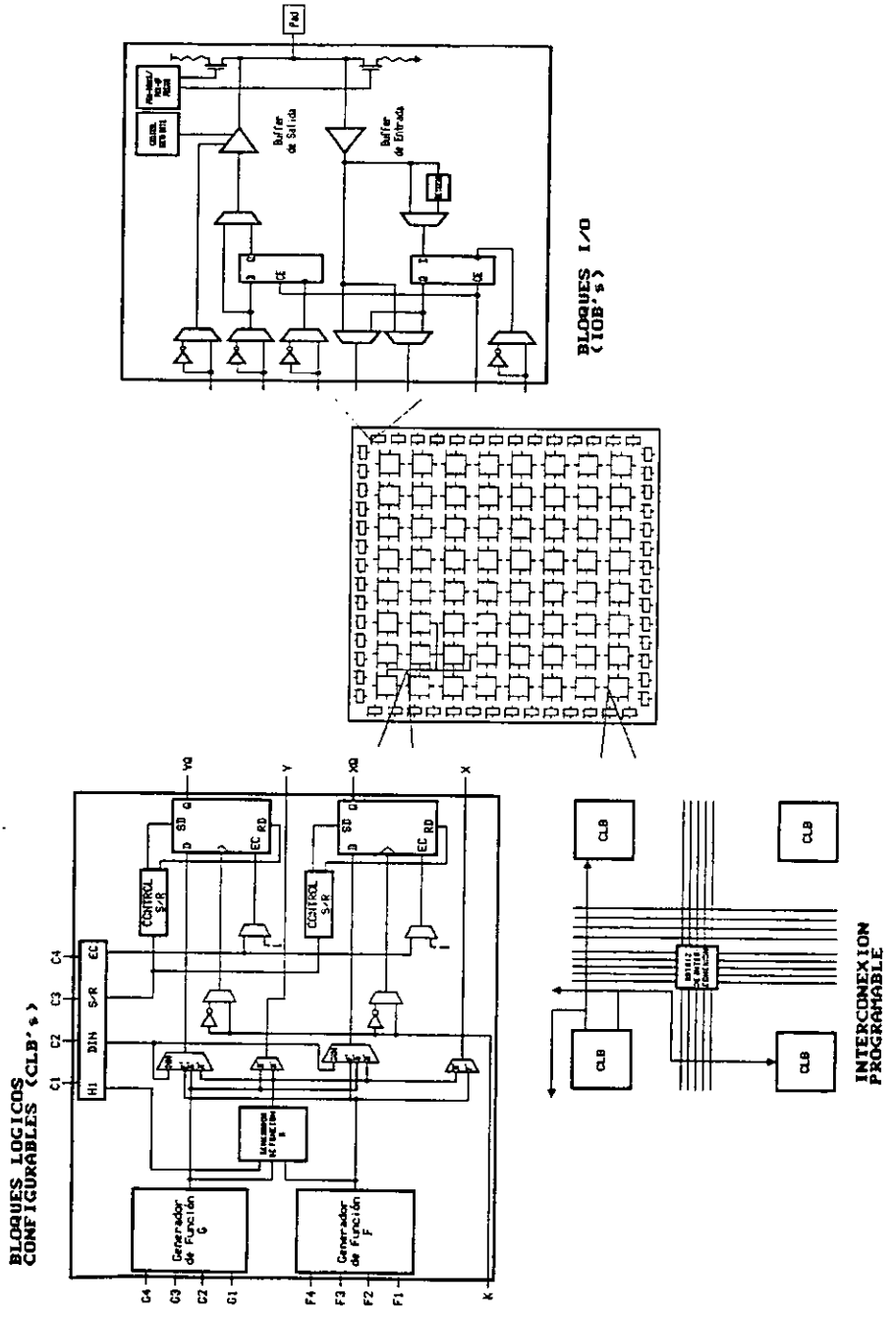


Figura 1.2: Arquitectura de los FPGA's de la Serie XC4000.

Características adicionales de los FPGA's de la Serie XC4000EX / XL.

- Velocidad del sistema incrementada.
- Capacidad de enrutamiento incrementada.
- Nueva capacidad Latch en los CLB's.
- Nueva Red de Reloj flexible de alta velocidad.
- Modo de Configuración Paralelo de alta velocidad (Express™).
- Bits adicionales de dirección en el Modo de Configuración Paralelo Maestro.

Ya que los FPGA's de Xilinx poseen un gran número características que mejoran su desempeño y capacidad de funcionamiento, pueden usarse en diseños innovadores donde el hardware se cambia dinámicamente, o donde debe adaptarse a diferentes aplicaciones. Además, debido a la lógica de este tipo de dispositivos, pueden ser usados en una gran variedad de aplicaciones tales como: aplicaciones que involucran operaciones aritméticas donde se requiere de gran eficiencia y desempeño, cálculos de desplazamiento de dirección a alta velocidad en sistemas gráficos o basados en microprocesador, procesamiento digital de señales a alta velocidad, transmisión de datos, sistemas de auto – diagnostico, o para implementar hardware de propósito general (como filtros digitales, decodificadores de dirección, comprobadores de paridad, contadores cargables, o aplicaciones FIFO).

1.3.3 Comparación entre CPLD's Y FPGA's.

Ahora se dará paso, después de analizar por separado las principales características arquitectónicas de ambas series de dispositivos y algunas de sus aplicaciones más importantes, a establecer las diferencias más significativas de ambos dispositivos lógicos programables en sistema. Con el fin de tener una idea más clara de estos dispositivos, y de sus cualidades; a partir de la comparación entre ellos.

Como se pudo observar, la Serie XC9500 posee una arquitectura ideal para la implementación de casi cualquier diseño de tipo combinacional o secuencial. Debido a que las características de estos dispositivos permiten el diseño de estas funciones lógicas en un solo chip; y por lo tanto hay una reducción de componentes utilizados, disminución en la disipación de potencia, velocidad de operación incrementada, corto tiempo de diseño al utilizar herramientas CAD, y en consecuencia, costos mínimos de Hardware / Software.

A diferencia de estos, los dispositivos de la Serie XC4000 poseen una arquitectura de más alta capacidad y rendimiento basada en memoria RAM que provee los beneficios de cualquier dispositivo VLSI pero a un bajo costo. Además, las características de esta serie permiten la implementación de diseños mucho más complejos como el procesamiento digital de señales a alta velocidad o la transmisión de datos, es decir, permiten un número mayor de funciones lógicas.

Al igual que los CPLD's de la Serie XC9500, los FPGA's XC4000 permiten reducir el número de componentes utilizados, y por consiguiente la disipación de potencia; y como las herramientas de diseño CAD son más versátiles permiten una mejor optimización del diseño al tener una mayor cantidad de opciones y menores ciclos de diseño y desarrollo. También ofrecen una solución eficiencia – costo mucho más buena para índices de producción altos (más de 5000 sistemas por mes). Para unidades de volumen alto más bajo es el costo.

Esto es, la Serie XC9500 ofrece una mejor solución para diseños pequeños de bajo costo en proyectos estudiantiles. Y la Serie XC4000, es ideal para diseños mucho más grandes de arquitectura computacional, procesamiento digital de señales y comunicaciones; a nivel escuela o industria.

La Tabla 1. 1 muestra las diferencias principales de ambos dispositivos.

CPLD's XC9500.	FPGA's XC4000.
Herramientas CAD.	Herramientas CAD más versátiles.
Costo mínimo a menor volumen.	Costo menor a mayor volumen.
Diseños Combinacionales y Secuenciales.	Diseños de arquitectura Computacional, Procesamiento Digital de Señales y Comunicaciones.
Proyectos Estudiantiles.	Nivel Escuela o Industria.

Tabla 1.1: Diferencias de los Dispositivos XC95000 y XC4000.

Comparación entre CPLD's y FPGA's Tomando en Cuenta otros Fabricantes.

Como se vio anteriormente, existen otros fabricantes de dispositivos programables por el usuario en el segmento de los CPLD's y FPGA's; además de Xilinx.

Para los CPLD's se tienen a los siguientes:

- Altera, cuya familia principal de dispositivos es la serie EP con densidades desde 2,100 hasta 7,500 compuertas con la Serie EPM.

- Exel Microelectronics, cuyo dispositivo desarrollado sólo contiene un arreglo plegado que permite llevar a cabo diseños lógicos de múltiples niveles.

- Y otros fabricantes más, pero aquí sólo son mencionados algunos de ellos, a manera de referencia.

Y para los FPGA's:

- Plessey Semiconductors, específicamente la familia ERA60K que incluye dispositivos con densidades desde 2,000 hasta 40,000 compuertas. Esta familia tiene la ventaja de ser reconfigurable eléctricamente.

- ACTEL Corporation, representado por su familia de dispositivos ACT1 con densidades desde 1,000 hasta 6,000 compuertas. Estos dispositivos no son reconfigurables.

- AMD, cuyos dispositivos tienen una densidad de 1,000 a 10,000 compuertas. Estos tienen la posibilidad de ser reconfigurados eléctricamente por medio del sistema de desarrollo.

- CMOS Technology con densidades de 1,200 a 3,000 compuertas, los cuales tienen funciones de suma de productos distribuidas en la matriz de interconexión programable. Esto es, las funciones lógicas primarias son realizadas en la matriz de interconexión y no en los bloques lógicos como se lleva acabo en los arreglos de Xilinx, Actel, etc. Las funciones generadas son

utilizadas a su vez por los bloques lógicos con propósitos combinacionales y / o secuenciales.

- Al igual que en el caso anterior, existen otros fabricantes.

La Tabla 1.2 muestra comparativamente las cualidades de ambos tipos de dispositivos tomando en cuenta a los distintos fabricantes de dispositivos. Lo anterior con el objetivo de establecer cual es la mejor opción de acuerdo a las necesidades de diseño.

CRITERIO.	CPLD's.	FPGA's.
Fabricantes y disponibilidad.	1	3
Rendimiento.	3	3
Area.	4	2
Facilidad de diseño.	1	2
Facilidad de cambios en el diseño.	1	2
Costos.	1	3
Disponibilidad en el mercado a nivel sistema.	1	3
Costos de fabricación a nivel sistema.	3	3

Nota: 1 = la mejor opción. 5 = la peor opción.

Tabla 1.2: Cualidades de Ambos Tipos de Dispositivos Tomando en Cuenta los Fabricantes.

CAPITULO 2

Arquitectura de los CPLD's de la Serie XC9500.

2.1 INTRODUCCION.

Los CPLD's de la familia XC9500 proporcionan una avanzada programación en sistema y capacidades de prueba de alto rendimiento, integrando una lógica de propósito general. Todos los dispositivos programables en sistema de esta familia cuentan como mínimo con 10,000 ciclos de programación y borrado. Se incluye también en todos los miembros de familia un extenso soporte de prueba y desarrollo sobre los diseños implementados.

Los dispositivos de la Serie XC9500 constan de Bloques de Entrada / Salida IOB's), Matrices de Interconexión y Bloques de Función (FB's) compuestos por macroceldas capaces de implementar cualquier función combinacional o de registro. En la Tabla 2.1 se muestra la densidad lógica de los dispositivos XC9500 la cuál oscila entre las 800 y las 6,400 compuertas útiles, respectivamente.

	XC9536	XC9572	XC95108	XC95144	XC95216	XC95288
Macroceldas.	36	72	108	144	216	288
Compuertas Utilizables.	800	1,600	2,400	3,200	4,800	6,400
Registros.	36	72	108	144	216	288
t_{PD} (ns.)	5	7.5	7.5	7.5	10	15
t_{SU} (ns.)	4.5	5.5	5.5	5.5	6.5	8.0
t_{CO} (ns.)	4.5	5.5	5.5	5.5	6.5	8.0
f_{CNT} (MHz.)	100	125	125	125	111	95
f_{SYSTEM} (MHz.)	100	83	83	83	67	56

Nota: f_{CNT} = frecuencia de operación para contadores de 16 – bits.

F_{SYSTEM} = frecuencia interna de operación para los sistemas de propósito general, diseñados con múltiples FB's.

Tabla 2.1: Dispositivos de la Familia XC9500.

Las múltiples opciones de empaquetamiento y las capacidades asociadas I / O se muestran en la Tabla 2.2. La familia XC9500 es totalmente compatible en sus pines, lo que permite la migración fácil del diseño a través de múltiples opciones de densidad en un determinado software de camino gráfico.

	XC9536	XC9572	XC95108	XC95144	XC95216	XC95288
44 Pin VQFP.	34					
44 Pin PLCC.	34	34				
84 Pin PLCC.		69	69			
100 Pin TQFP.		72	81	81		
100 Pin PQFP.		72	81	81		
160 Pin PQFP.			108	133	133	
208 Pin HQFP.					166	168
352 Pin BGA.					166	192

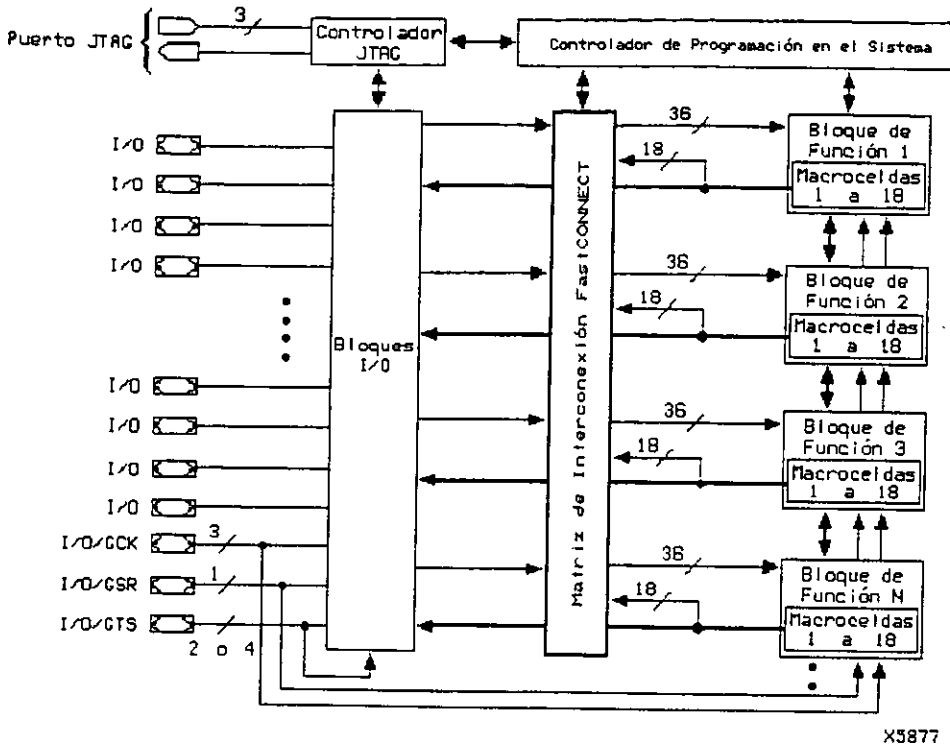
Tabla 2.2: Encapsulados Disponibles y Pines I / O de los Dispositivos (no incluye los pines dedicados JTAG).

Los aspectos arquitectónicos de la familia XC9500 cumplen con los requerimientos de programación en sistema. Además, de la mejorada capacidad de asignación de pines durante los cambios de diseño evita la costosa refabricación de placas. Un conjunto de instrucciones expandidas permite el control de la versión de los modelos de programación y la puesta a punto en el sistema. Los aspectos avanzados del sistema incluyen salidas con control de "Slew – Rate" y pines programables por el usuario para tierra, y así reducir el ruido del sistema.

2.2 DESCRIPCION DE LA ARQUITECTURA.

Cada dispositivo XC9500 es un subsistema que consta de múltiples Bloques de Función (FB's) y Bloques de I / O (IOB's) totalmente interconectados por la matriz de interconexión "FastCONNECT". El IOB provee buffers para las entradas y salidas del dispositivo. Cada FB provee la capacidad de lógica programable con 36 entradas y 18 salidas. La matriz de interconexión "FastCONNECT" conecta todas las salidas del FB y las señales de entrada a las entradas del FB. Por cada FB, existen de 12 a 18 salidas (dependiendo de la cantidad de pines del

encapsulado) y señales de habilitación de salida asociadas que manejan directamente a los IOB's. Ver la Figura 2.1.



Nota: Las Salidas del Bloque de Función (indicadas por las líneas gruesas) manejan directamente los Bloques I / O.

Figura 2.1: Arquitectura XC9500.

2.2.1 Bloque de Función.

Cada Bloque de Función, como se muestra en la Figura 2.2, está compuesto de 18 macroceldas independientes, cada una capaz de implementar una función combinacional o de registro. El FB también recibe el reloj global, la habilitación de salida, y las señales "Set / Reset". El FB genera 18 salidas que manejan a la matriz de interconexión "FastCONNECT". Estas 18 salidas y sus correspondientes señales de habilitación de salida también manejan al IOB.

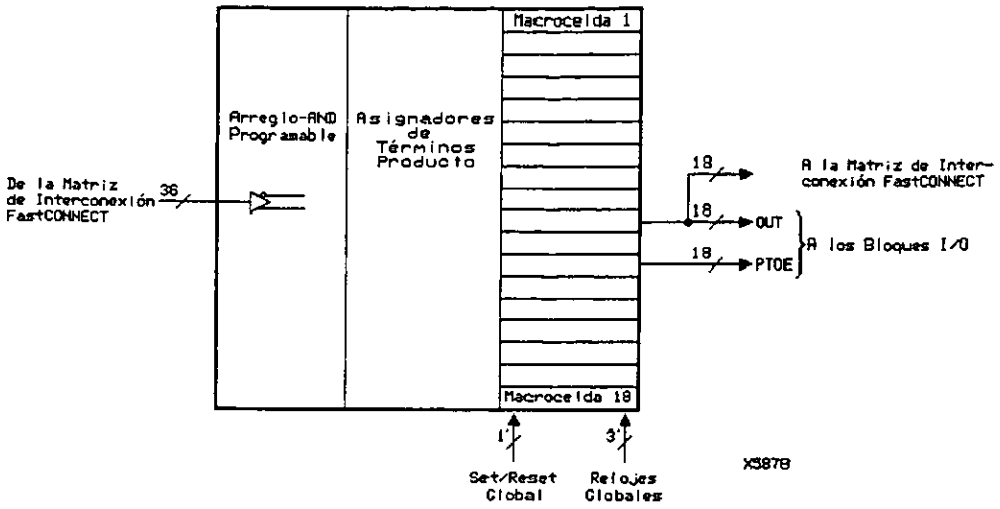


Figura 2.2: Bloque de Función XC9500.

La lógica dentro del FB se implementa usando una representación de suma de productos. Treinta y seis entradas proporcionan 72 señales verdaderas y su complemento en el arreglo AND programable para formar 90 términos producto. Cualquier número de estos términos producto término, hasta los 90 disponibles, pueden asignarse a cada macrocelda mediante el asignador de términos producto.

Cada FB (exceptuando el XC9536) soporta rutas de realimentación locales que permiten que cualquier número de salidas del FB maneje su propio arreglo AND programable sin salir del FB. Estas trayectorias se usan para crear contadores muy rápidos y máquinas de estado donde todos los estados del registro están dentro del mismo FB.

a) Macrocelda.

Cada macrocelda XC9500 puede configurarse individualmente para una función combinacional o de registro. La macrocelda y la lógica del FB asociado se muestran en la Figura 2.3.

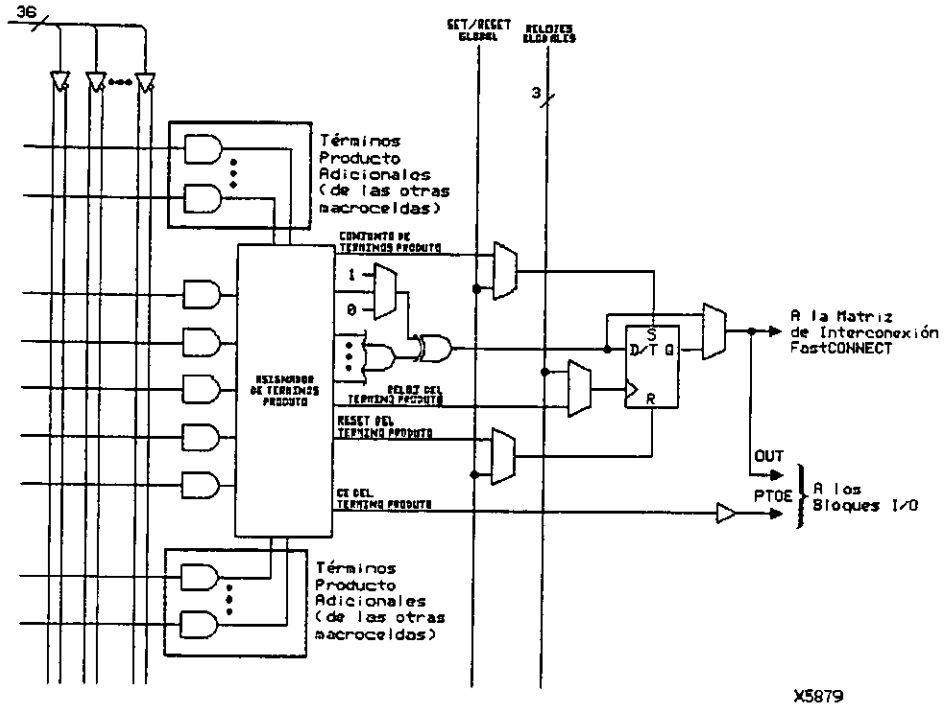


Figura 2.3: Macrocelda XC9500 Dentro del Bloque de Función.

Cinco términos producto directos desde el arreglo AND están disponibles para su uso como entradas de datos primarios (hasta las compuertas OR y XOR) para implementar funciones combinacionales, o como entradas de control incluyendo la señal de reloj, "Set / Reset" y habilitación de salida. El asignador de términos producto asociado con cada macrocelda selecciona cómo serán usados los cinco términos directos.

El registro de la macrocelda puede configurarse como un flip – flop de tipo D o tipo T, o puede ser desviado para operaciones combinacionales. Cada registro soporta ambas operaciones asíncronas "set y reset". Durante la aplicación de potencia, todos los registros del usuario son inicializados en estado de precarga, definido por el usuario (valor por omisión igual a 0, si no se especifica otro valor).

Todas las señales de control globales están disponibles en cada macrocelda individual, incluyendo la señal de reloj, "Set / Reset", y las señales de habilitación de salida. Como se muestra en la Figura 2.4, el reloj de registro de la macrocelda origina cualquiera de los tres relojes globales o el reloj del término producto. Ambas polaridades, la verdadera y su complemento del pin GCK pueden usarse dentro del dispositivo. Una entrada GSR también permite que en los registros de usuario pueda establecerse el estado del usuario.

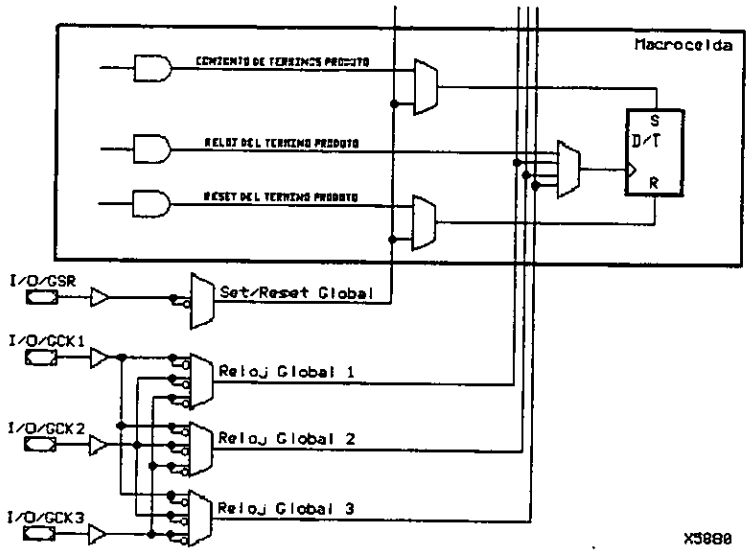


Figura 2.4: Reloj de Macrocelda y Capacidad Set / Reset.

b) Asignador de Términos Producto.

El asignador de términos producto controla de que manera los cinco términos producto directos son asignados a cada macrocelda. Por ejemplo, todos los cinco términos directos pueden manejar la función OR como se muestra en la Figura 2.5.

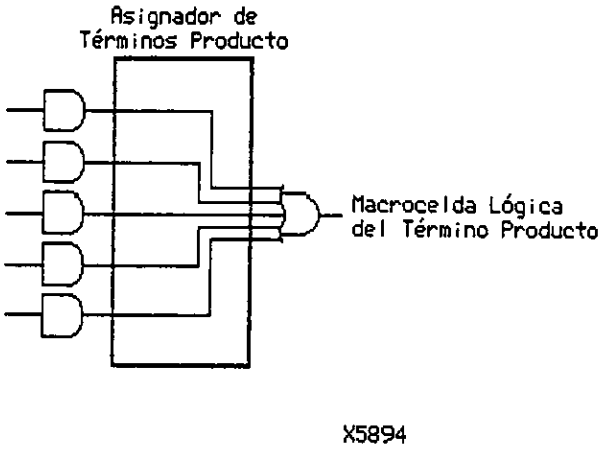


Figura 2.5: Lógica de la Macrocelda Usando el Término Producto Directo.

El asignador de términos producto puede reasignar otros términos producto dentro del FB para aumentar la capacidad lógica de una macrocelda más allá de cinco términos directos. Cualquier macrocelda que requiera términos producto adicionales puede accesar de forma no obligatoria a los términos producto en otras macroceldas dentro del FB. Hasta 15 términos producto pueden estar disponibles para una sola macrocelda con sólo un pequeño retardo incremental de t_{PTA}, como se muestra en la Figura 2.6.

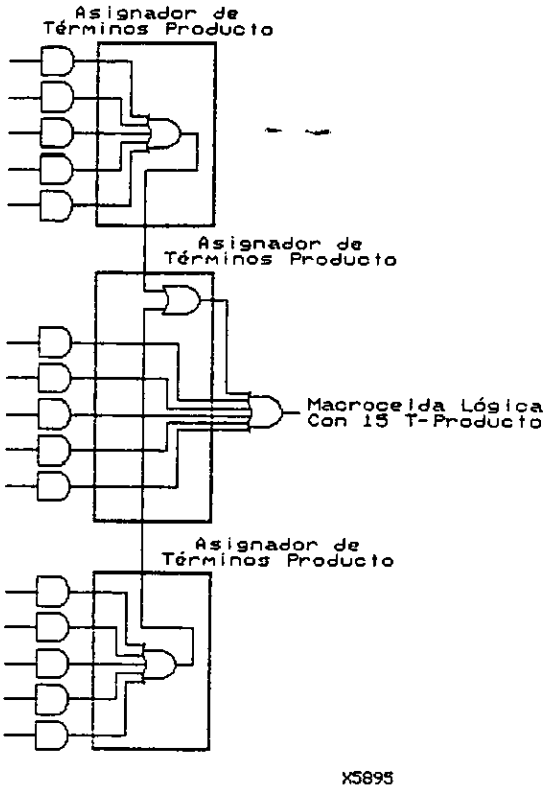


Figura 2.6: Asignador de Términos Producto con 15 Términos Producto.

Se debe notar que el retardo incremental afecta sólo a los términos producto en otras macroceldas. La temporización de los términos producto directos no cambia.

El asignador de términos producto puede reasignar los términos producto desde cualquier macrocelda dentro del FB combinando sumas parciales de productos sobre varias macroceldas, como se muestra en la Figura 2.7.

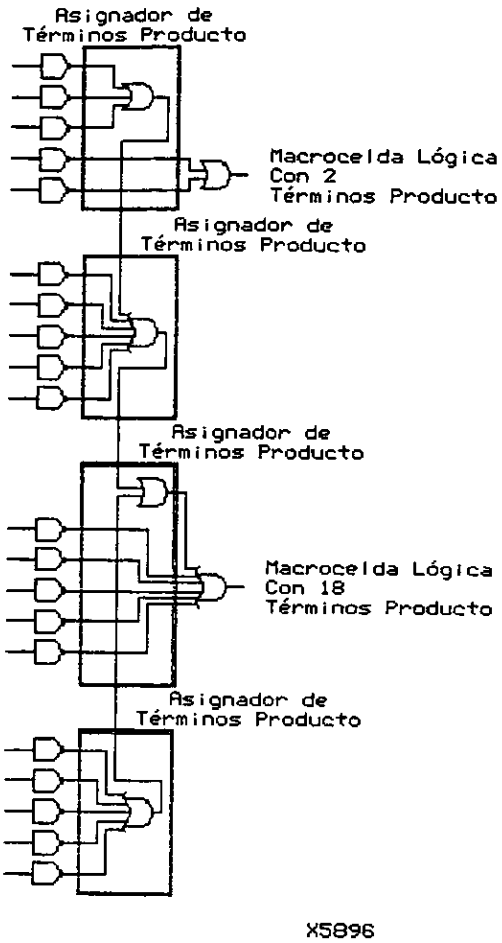


Figura 2.7: Asignador de Términos Producto sobre Varias Macroceldas.

En este ejemplo, el retardo incremental es sólo $2*t_{PTA}$. Todos los 90 términos producto están disponibles en cualquier macrocelda, con un retardo incremental máximo de $8*t_{PTA}$.

La lógica interna del asignador de términos producto se muestra en la Figura 2.8.

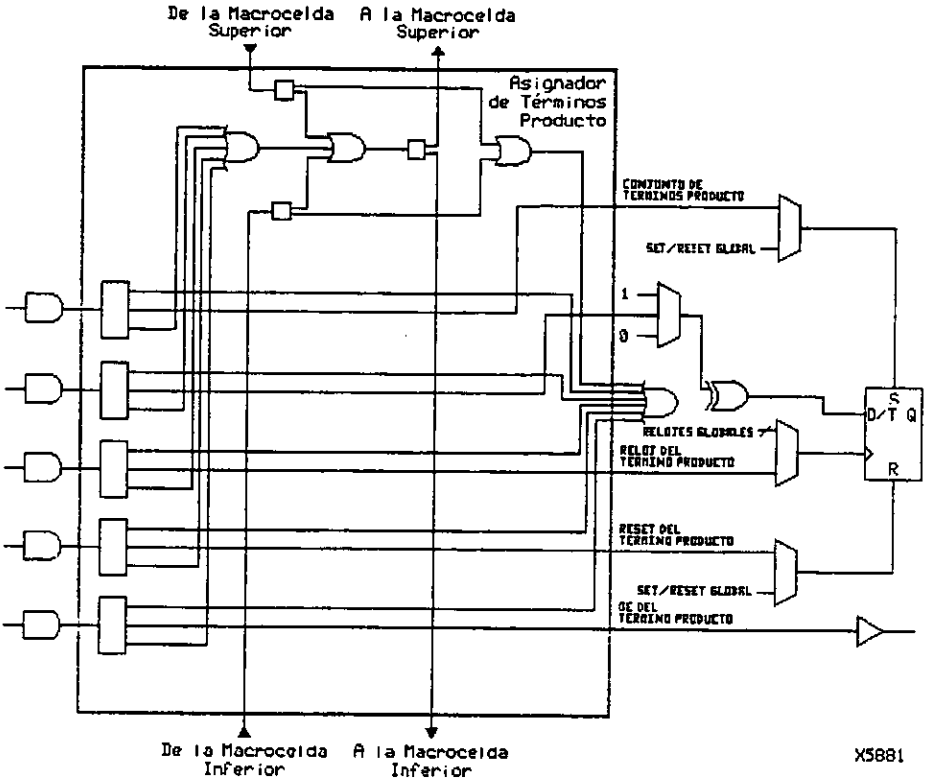


Figura 2.8: Lógica del Asignador de Términos Producto.

2.2.2 Matriz de Interconexión "FastCONNECT".

La matriz de interconexión "FastCONNECT" conecta las señales a las entradas del FB, como se muestra en la Figura 2.9. Todas las salidas del IOB (correspondiendo a los pines de entrada del usuario) y todas las salidas del FB manejan la Matriz "FastCONNECT". Cualquiera de estas (hasta una sobrecarga de entrada del FB al límite de 36) puede seleccionarse, mediante la programación del usuario, para manejar cada FB con un retardo uniforme.

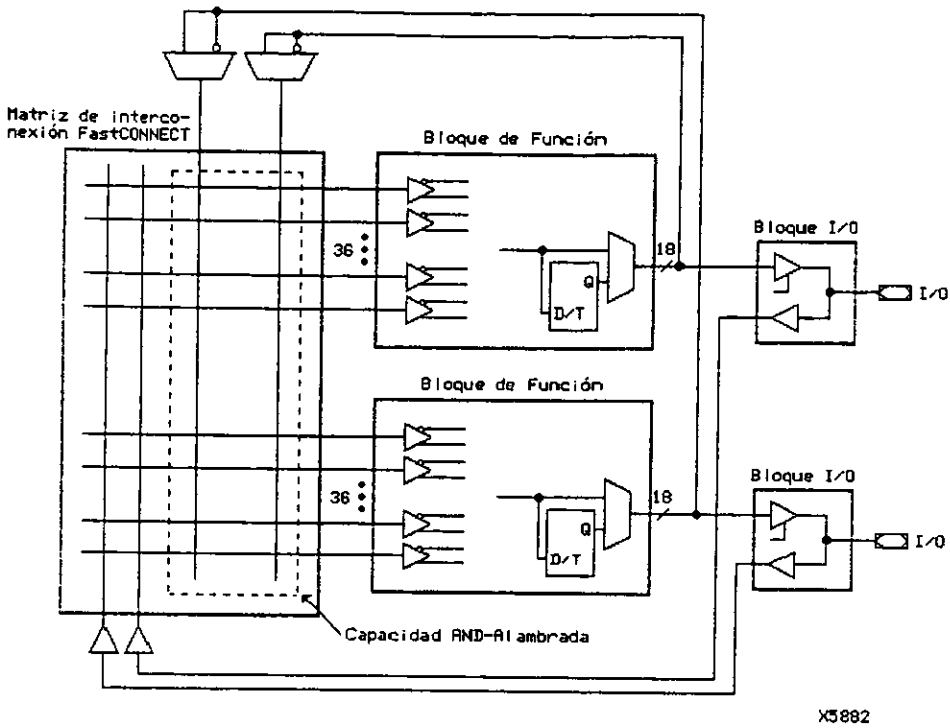


Figura 2.9: Matriz de Interconexión "FastCONNECT".

La matriz de interconexión "FastCONNECT" es capaz de combinar múltiples conexiones internas en una sola AND alambrada de salida antes de manejar al FB destino. Esto proporciona una capacidad lógica adicional y aumenta la lógica eficaz de sobrecarga de entrada del FB destino sin ningún retardo de tiempo adicional. Esta capacidad está disponible para conexiones internas que se originan sólo desde las salidas del FB. Esto es invocado automáticamente por el software de desarrollo donde sea necesaria su aplicación.

2.2.3 Bloque I / O.

El Bloque I / O (IOB) consta de la interfaz entre la lógica interna y los pines de I / O del usuario del dispositivo. Cada IOB incluye un buffer de entrada, un manejador de salida, una selección de

habilitación de salida multiplexada, y control de tierra programable por el usuario. Ver la Figura 2.10 para detalles.

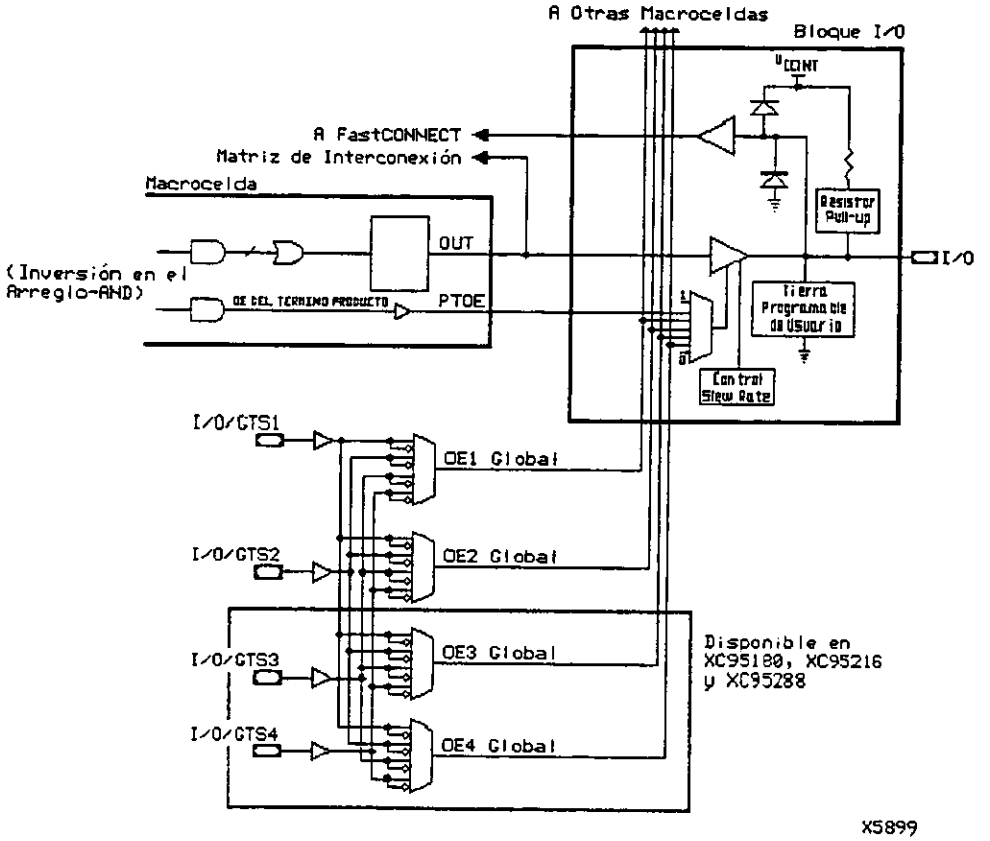


Figura 2.10: Bloque I /O y la Capacidad de Habilitación de Salida.

El buffer de entrada es compatible con el estándar 5V. CMOS, 5 V. TTL y 3.3 V. El buffer de entrada usa los 5 V. internos de suministro de voltaje (V_{CCINT}) para asegurar que los umbrales de la entrada sean constantes y no varíen con el voltaje de V_{CCIO}.

La habilitación de salida puede generarse desde una de cuatro opciones: una señal de término producto de la macrocelda, o cualquiera de las señales globales OE, siempre "1", o siempre "0". Hay dos tipos de habilitación de salida global para los dispositivos con hasta 144 macroceldas, y

cuatro tipos de habilitación de salida global para los dispositivos con 180 o más macroceldas. Ambas polaridades de cualquier control global de tercer estado (GTS) pueden usarse dentro del dispositivo.

Cada salida tiene de forma independiente su control "Slew – Rate". En la salida puede ser posible reducir los índices de ruido por debajo del límite (con un retardo adicional de tiempo t_{SLEW}) mediante la programación. Ver la Figura 2.11.

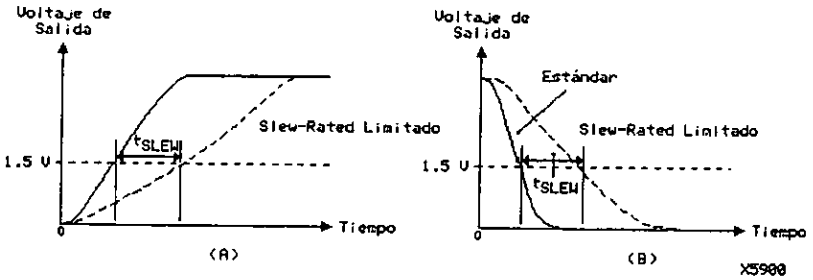


Figura 2.11: Control de Salida "Slew – Rate" para Salidas (a) Ascendentes y (b) Descendentes.

Cada IOB proporciona al usuario la capacidad de programar el pin de tierra. Esto permite al dispositivo que los pines I / O puedan ser configurados como pines de tierra adicionales. Mediante conexiones estratégicamente localizadas los pines de tierra programables poseen conexión de tierra externa, por lo que el ruido generado por el sistema debido a los altos índices de cambios simultáneos en las salidas puede reducirse.

Un resistor de control "pull – up" (típicamente 10 KΩ) se agrega a cada pin I / O del dispositivo para prevenir que los pines del dispositivo floten cuando el dispositivo no está en funcionamiento normal de usuario. Este resistor es activo durante el tiempo en que el dispositivo se encuentra en modo de programación y el sistema está energizado. Se activa también en el borrado del dispositivo. El resistor se desactiva durante la operación normal.

El manejador de salida es capaz de proporcionar 24 mA. en la salida. Todos los manejadores de salida en el dispositivo pueden configurarse para niveles de 5 V. TTL o para niveles de 3.3 V. conectando la salida del dispositivo a un voltaje de suministro (V_{CCIO}) de 5 V. o 3.3 V. La Figura

2.12 muestra cómo el dispositivo XC9500 sólo puede usarse con 5 V. y en sistemas mixtos 3.3 V. / 5 V.

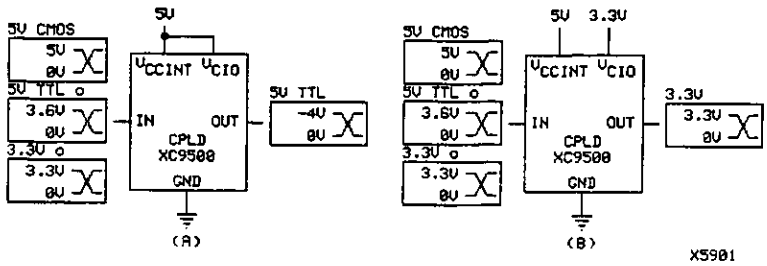


Figura 2.12: Dispositivos XC9500 en (a) Sistemas de 5 V. y (b) Sistemas Mixtos de 3.3 V. / 5 V.

2.3 CAPACIDAD DEL PIN DE BLOQUEO.

La capacidad de bloqueo es definida por el usuario mediante la asignación de pines durante los cambios de diseño, lo cual depende de la habilidad de la arquitectura para adaptarse a cambios inesperados. Los dispositivos XC9500 tienen características arquitectónicas que mejoran la habilidad de aceptar cambios de diseño mientras se mantiene el mismo pin de salida.

La arquitectura XC9500 proporciona un máximo enrutamiento dentro de la matriz de interconexión "FastCONNECT", e incorpora un Bloque de Función flexible que permite la amplia asignación de bloques disponibles de términos producto. Esto proporciona un alto nivel de confiabilidad para mantener los pines asignados de entradas y salidas frente a cambios inesperados de diseño.

Para los cambios extensivos de diseño que requieren una capacidad lógica más alta que la que está disponible en el dispositivo inicialmente elegido, el nuevo diseño debe ser capaz de acoplarse dentro del sistema y el número de pines del dispositivo debe ser compatible, usando las mismas asignaciones en éstos. Por lo que la misma tarjeta podrá usarse con un dispositivo de más alta densidad sin el gasto de refabricación.

2.4 PROGRAMACION EN EL SISTEMA.

Los dispositivos XC9500 se programan en el sistema por medio de un protocolo JTAG estándar de 4 – Pines, como se muestra en la Figura 2.13. La programación en el sistema ofrece un rápido y eficiente diseño de iteraciones y elimina el manejo de paquetería. El sistema de desarrollo de Xilinx proporciona la secuencia de datos de programación que emplea la transmisión por cable, una tercera parte es el sistema de desarrollo JTAG, probador JTAG compatible de tarjeta, o una simple interfaz de microprocesador que emule la secuencia de instrucciones JTAG.

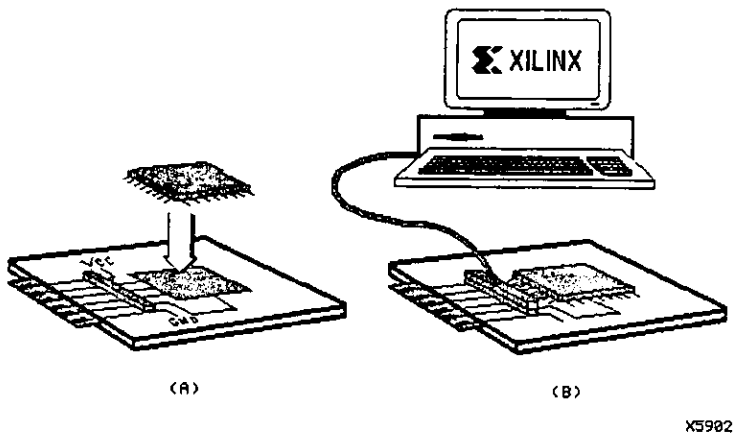


Figura 2.13: Operación de Programación en Sistema. (a) Dispositivo de Soldadura con PCB y (b) Programación Usando Transmisión por Cable.

Todos los I / O's están en tercer estado y en "pulled" alto para las resistencias del IOB durante la programación en sistema. Si una señal en particular debe permanecer en estado bajo durante este tiempo, entonces un resistor "pull – down" puede agregarse al pin.

2.4.1 Programación Externa.

Los dispositivos XC9500 pueden también programarse por el dispositivo programador Xilinx HW130 como tercera opción para los programadores. Esto proporciona la flexibilidad adicional de usar dispositivos preprogramados durante la fabricación, con la opción de programación en el sistema para futuros perfeccionamientos.

2.4.2 Resistencia.

Todos los CPLD's XC9500 proporcionan un nivel mínimo de resistencia de 10,000 ciclos de Programación / Borrado en el sistema. Cada dispositivo conjunta todo lo funcional, buen desempeño, y especificaciones de retención de datos dentro de su límite de resistencia.

2.5 "BOUNDARY SCAN" (JTAG) IEEE 1149.1.

Los dispositivos XC9500 soportan completamente las instrucciones del "Boundary – Scan" (JTAG) IEEE 1149.1. EXTEST, SAMPLE / PRELOAD, BYPASS, USER - CODE, INTEST, IDCODE, y HIGHZ en cada dispositivo. Para las operaciones ISP, se agregan cinco instrucciones adicionales: ISPEN, FERASE, FPGM, FVFY, e instrucciones ISPEX que son extensiones totalmente compatibles del conjunto de instrucciones 1149.1.

Los pines TMS y TCK se han destinado a las resistencias "pull – up" como es especificado por la norma IEEE 1149.1.

2.6 DISEÑO DE SEGURIDAD.

Los dispositivos XC9500 incorporan los aspectos más avanzados de seguridad de datos y ofrecen una total protección de los datos de programación contra la lectura no autorizada, la reprogramación o la borradura inadvertida del dispositivo. La Tabla 2.3 muestra las cuatro diferentes posibilidades de seguridad disponibles.

Escritura de Seguridad.	*Lectura de Seguridad.*	
	Valor por Omisión.	**Valor Fijado.**
Valor por Omisión.	Lectura Permitida. Programación / Borrado Permitido.	Lectura No Permitida. Programación / Borrado Permitido.
Valor Fijado.	Lectura Permitida. Programación / Borrado No Permitido.	Lectura No Permitida. Programación / Borrado No Permitido.

X5905.

Tabla 2.3: Opciones de Seguridad de Datos.

La lectura de los bits de seguridad puede ser colocada por el usuario en el patrón de programación interno para prevenir que sean leídos o puedan copiarse. Borrar el dispositivo entero es la única manera de restablecer el bit de seguridad leído.

La escritura de los bits de seguridad proporciona una protección adicional contra la borradura accidental del dispositivo o reprogramación cuando los pines JTAG están sujetos al ruido, tal como sucede durante la aplicación de potencia al sistema. Una vez establecida, la protección de escritura puede desactivarse cuando el dispositivo necesita ser reprogramado con un patrón de diseño.

2.7 MODO DE BAJO CONSUMO DE POTENCIA.

Todos los dispositivos XC9500 ofrecen un modo de bajo consumo de potencia para macroceldas individuales o para todas las macroceldas. Este aspecto permite reducir significativamente el consumo de potencia del dispositivo.

Cada macrocelda individual puede programarse en el modo de bajo consumo de potencia por el usuario. Las partes críticas del funcionamiento de la aplicación pueden permanecer en el modo estándar de consumo de potencia, mientras otras partes de la aplicación pueden programarse para su funcionamiento en el modo de bajo consumo de potencia para reducir la disipación total de potencia. Las macroceldas programadas para el modo de bajo consumo de potencia incurren en un retardo adicional de tiempo (t_{LP}) entre pines combinacionales así como en el tiempo de establecimiento del registro. Los retardos del reloj del término producto de salida y el habilitador de término producto de salida no son afectados por el modo de consumo de potencia de la macrocelda.

2.8 MODELO DE TEMPORIZACION.

La uniformidad de la arquitectura XC9500 permite un modelo de temporización simplificado para el dispositivo completo. El modelo de temporización básico, mostrado en la Figura 2.14, es válido para funciones de la macrocelda que sólo usan los términos producto directos, con el modo de consumo de potencia estándar, y el estándar del "Slew – Rate".

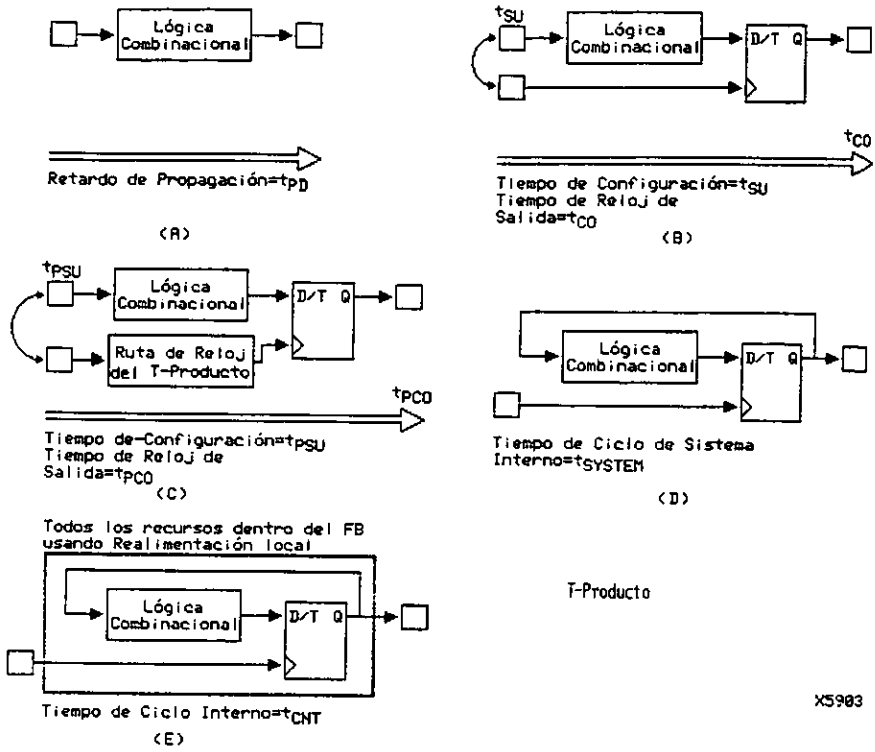


Figura 2.14: Modelo Básico de Temporización.

La Tabla 2.4 muestra cómo cada uno de los parámetros importantes, es afectado por el asignador de términos producto (sí es necesario), la implantación del modo de bajo consumo de potencia, y el establecimiento del límite de "Slew".

Descripción.	Parámetro.	Asignador de Términos Producto[1].	Macrocelda en Modo de Bajo Consumo de Potencia.	Salida en Modo "Slew " Limitado.
Retardo de Propagación.	t_{PD}	+ t_{PTA} * S	+ t_{LP}	+ t_{SLEW}
Reloj Global de Tiempo de Establecimiento.	t_{SU}	+ t_{PTA} * S	+ t_{LP}	-
Reloj Global de Salida.	t_{CO}	-	-	+ t_{SLEW}
Reloj de Tiempo de Establecimiento del Término Producto.	t_{PSU}	+ t_{PTA} * S	+ t_{LP}	-
Reloj de Salida del Término Producto.	t_{PCO}	-	-	+ t_{SLEW}
Período de Ciclo de Sistema Interno.	t_{SYSTEM}	+ t_{PTA} * S	+ t_{LP}	-

Nota: 1. S = intervalo lógico de la función, como se definió en el texto.

Tabla 2.4: Parámetros del Modelo de Temporización.

El tiempo del asignador de términos producto depende del período de la función de la lógica de la macrocelda, que se define como uno menos el número máximo del asignador en la trayectoria del término producto. Si sólo se usan los términos producto directos, entonces el período de la lógica es 0. En el ejemplo de la Figura 2.6 se muestra que hasta 15 términos producto están disponibles con un periodo de 1. En el caso de la Figura 2.7, las 18 funciones de término producto tienen un periodo de 2.

La información detallada de temporización puede derivarse del modelo de temporización detallado mostrado en la Figura 2.15.

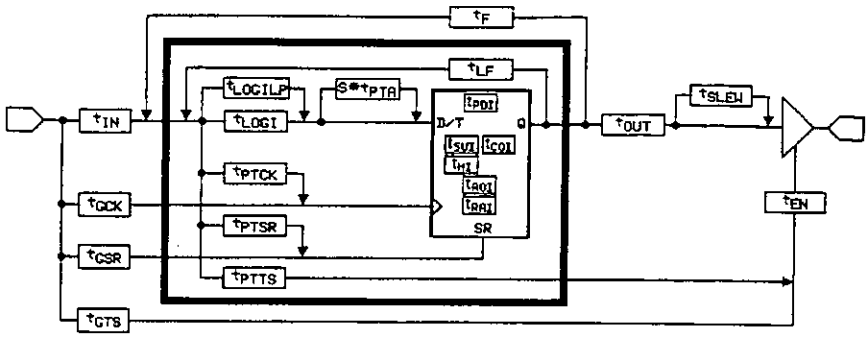


Figura 2.15: Modelo de Temporización Detallado.

2.9 CARACTERISTICAS DE ALIMENTACION DE POTENCIA.

Los dispositivos XC9500 se comportan bien bajo todas las condiciones de operación. Durante la etapa de alimentación de potencia cada dispositivo XC9500 emplea la circuitería interna que mantiene al dispositivo en el estado inmóvil hasta que el voltaje de abastecimiento V_{CCINT} esté en un nivel seguro (aproximadamente 3.8 V.). Durante este tiempo, todos los pines del dispositivo y los pines JTAG están deshabilitados, y todas las salidas del dispositivo están deshabilitadas con los resistores "pull – up" del IOB (~ 10 KΩ) habilitados, como se muestra en la Tabla 2.5.

Características del Dispositivo.	Estado Inactivo.	Operación de Borrado del Dispositivo.	Operación de Usuario Válida.
Resistores "Pull – up" del IOB.	Habilitado.	Habilitado.	Deshabilitado.
Salidas del Dispositivo.	Deshabilitado.	Deshabilitado.	Se Configura.
Entradas del Dispositivo y Relojes.	Deshabilitado.	Deshabilitado.	Se Configura.
Bloque de Función.	Deshabilitado.	Deshabilitado.	Se Configura.
Controlador JTAG.	Deshabilitado.	Habilitado.	Habilitado.

Tabla 2.5: Características delos Dispositivos XC95000.

Cuando el voltaje de alimentación alcanza un nivel seguro, todos los registros de usuario se inicializan (típicamente dentro de 100 µs. para 9536 - 95144, 200 µs. para 95216 y 300 µs. para 95288), y el dispositivo está inmediatamente disponible para la operación, como se muestra en la Figura 2.16.

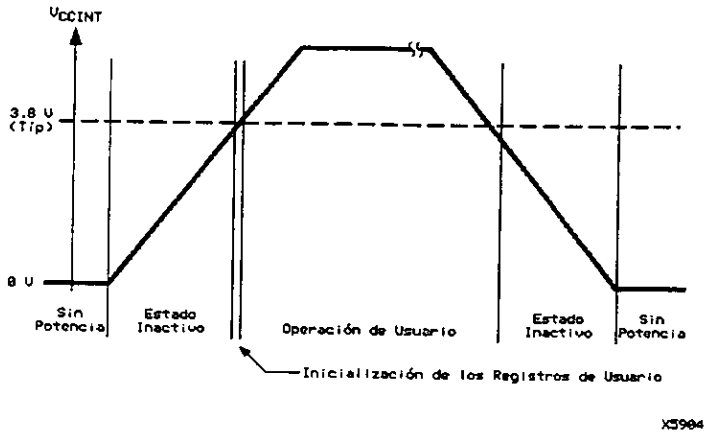


Figura 2.16: Comportamiento del Dispositivo Durante el Período de Alimentación de Potencia.

Si el dispositivo está en el estado de borrado (antes de que cualquier patrón del usuario se programe), las salidas del dispositivo permanecen inválidas con los resistores "pull – up" del IOB habilitados. Los pines JTAG están habilitados para permitir que el dispositivo sea programado en cualquier momento.

Si el dispositivo se programa, las entradas y salidas del dispositivo asumen sus estados configurados para el funcionamiento normal. Los pines JTAG se habilitan y permiten el borrado del dispositivo o las pruebas de "Boundary – Scan" en cualquier momento

2.10 SOPORTE DE SISTEMA DE DESARROLLO.

La familia XC9500 de CPLD's es soportada totalmente por los sistemas de desarrollo disponibles de Xilinx y los vendedores de Xilinx Alliance Program.

El diseñador puede crear el diseño usando ABEL, esquemáticos, ecuaciones, VHDL, o Verilog en una variedad de software con herramientas de punta. El sistema de desarrollo puede usarse para implementar el diseño y generar el archivo de mapa de bits JEDEC el cual puede usarse para programar al dispositivo XC9500. Cada sistema de desarrollo incluye JTAG para la transmisión de software que puede usarse para programar los dispositivos por medio de la interfaz del estándar JTAG y un cable de transmisión.

2.11 TECNOLOGIA DE FastFLASH.

El avanzado proceso CMOS Flash se usa para fabricar todos los dispositivos XC9500. Específicamente desarrollado por Xilinx en sistemas programables de CPLD's, el proceso de FastFLASH proporciona la capacidad de una lógica de alto rendimiento, tiempos de programación rápidos, y la resistencia de 10,000 ciclos de Programación / Borrado.

CAPITULO 3

Arquitectura de los FPGA's de la Serie XC4000.

3.1 INTRODUCCION.

La Serie XC4000 de los FPGA's de Xilinx se encuentra dentro de la clasificación de densidad de integración VLSI ofreciendo entre otras las siguientes ventajas: evita el costo inicial de diseño e implementación, el largo ciclo de desarrollo, el riesgo inherente de un arreglo convencional de compuerta enmascarada y una buena solución eficiencia – costo para índices de producción de alto volumen.

Los dispositivos XC4000 son desarrollados con una arquitectura programable que consta de Bloques Lógicos Configurables (CLB's) interconectados por poderosos recursos de enrutamiento, y circundados por Bloques de Entrada / Salida (IOB's). Ellos poseen abundantes recursos de enrutamiento para acomodar el más complejo modelo de interconexión. La arquitectura de los dispositivos XC4000 se muestra en la Figura 3.1.

Los FPGA's de la Serie XC4000 poseen celdas de memoria internas en las cuales son cargados los datos de usuario para la personalización de los mismos, y pueden activarse para leer o escribir sus datos de configuración en diversas formas. La Serie XC4000 está respaldada por un software sofisticado, que cubre todos los aspectos de diseño desde una entrada esquemática o de comportamiento, hasta la creación, transmisión y lectura posterior de la cadena de bits de configuración del dispositivo.

Los FPGA's de la Serie XC4000 pueden ser reconfigurados para cambiar la función lógica mientras esté residente en el sistema, cualidad no disponible en otro tipo de lógica programable. El hardware y el software, así como las actualizaciones o modificaciones al diseño, pueden cambiarse fácilmente aún ya en la práctica. Todas estas ventajas aunadas a los 19 miembros de que consta la Serie XC4000 ofrecen una gran versatilidad para elegir entre uno u otro dispositivo para una óptima implementación.

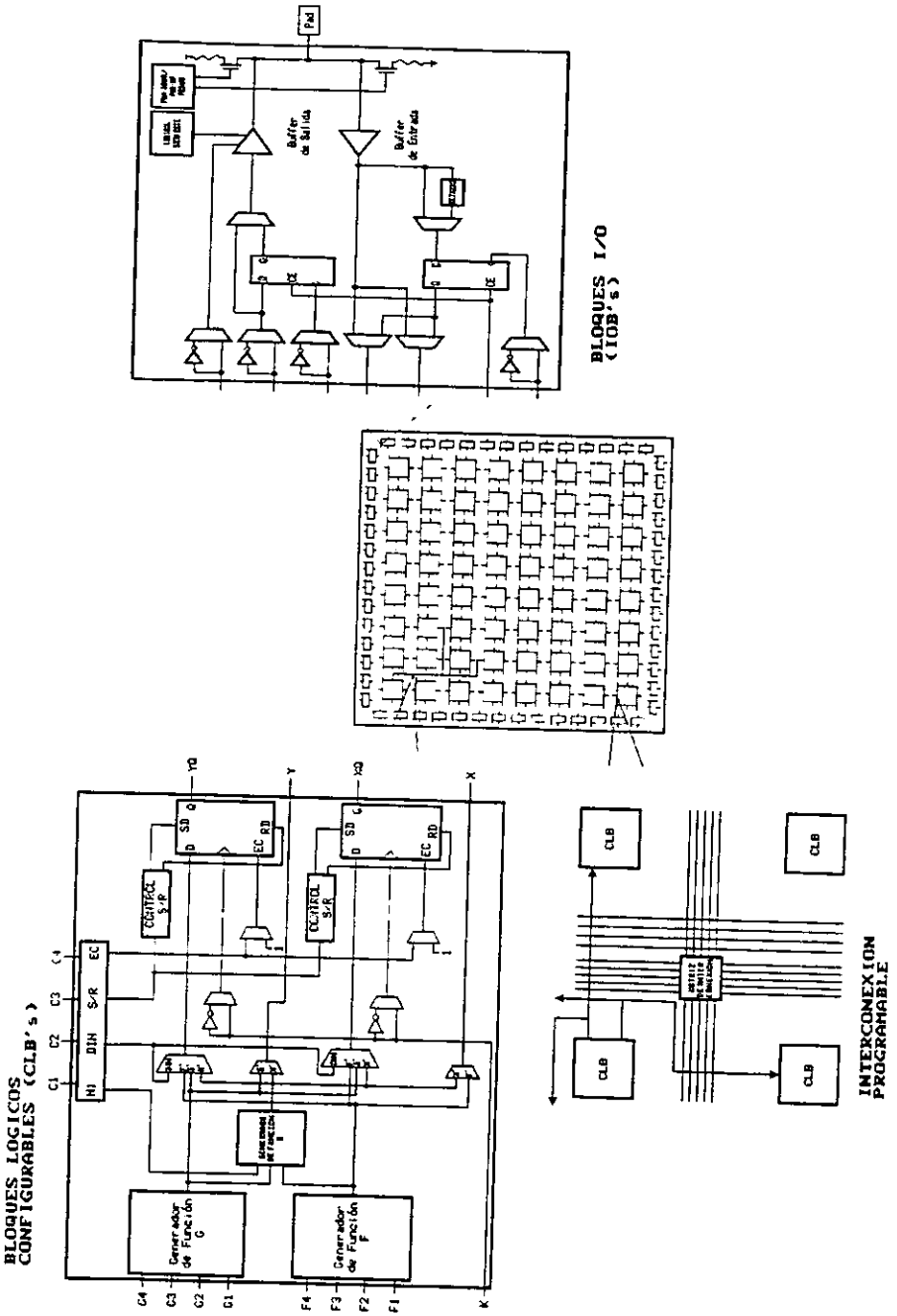


Figura 3.1: Arquitectura XC4000.

Los dispositivos de la Serie XC4000 de uso más común se muestran en la Tabla 3.1.

Dispositivo.	Max. Compuertas Lógicas. (No RAM)	Max. Bits RAM. (No Lógicos)	Rango Típico de Compuertas. (Lógicas y RAM)*	Matrices CLB	Total Bloques Lógicos.	Número de Flip - flops.	Max. Entradas de decodificador por lado.	Max. I / O de usuario.
XC4003E	3000	3200	2000-5000	10x10	100	360	30	80
XC4005E/L	5000	6272	3000-9000	14x14	196	616	42	112
XC4006E	6000	8192	4000-12000	16x16	256	768	48	128
XC4008E	8000	10368	6000-15000	18x18	324	936	54	144
XC4010E/L	10000	12800	7000-20000	20x20	400	1120	60	160
XC4020E	20000	25088	13000-40000	28x28	784	2016	84	224
XC4025E	25000	32768	15000-45000	32x32	1024	2560	96	256
XC4028EX/XL	28000	32768	18000-50000	32x32	1024	2560	96	256
XC4036EX/XL	36000	41472	22000-65000	36x36	1296	3168	108	288
XC4044EX/XL	44000	51200	27000-80000	40x40	1600	3840	120	320
XC4052XL5	52000	61952	33000-100000	44x44	1936	4576	132	352
XC4062XL	62000	73728	40000-130000	48x48	2304	5376	144	384

Los dispositivos más grandes estarán disponibles en la primera mitad de 1997.

* Valores Máximos del Rango Típico de Compuertas, incluye del 20 al 30% de CLB's usados como RAM.

Tabla 3.1: Serie XC4000 de Arreglos de Compuertas Programables de Campo.

La lógica reconfigurable de estos sistemas puede usarse en una gran variedad de aplicaciones tales como: los sistemas de auto – diagnostico, o para implementar hardware de propósito general como se muestra en la Tabla 3.2.

Clase de Diseño.	Función.	CLB's Usados.	XC4000E-3.	XC4000E-2.	Unidades.
Memoria.	Puerto sencillo (leer/modificar/escribir) 256x8.	72	63	80	MHz.
	FIFO 32x16 bits				
	lectura/escritura simultanea	48	63	80	MHz.
	lectura/escritura Multiplexada.	32	63	80	MHz.
Lógico.	Registro de corrimiento 9 bits (con habilitación).	5	170	200	MHz.
	Contador Pre – Escalonado 16 bits.	8	142	170	MHz.
	Contador Cargable 16 bits.	8	65	76	MHz.
	Acumulador 16 bits.	9	65	76	MHz.
	Filtro FIR de 16 ciclos de				
	evaluación 8 bits, razón de muestreo				
	paralelo	400	55	65	MHz.
	serial.	68	8.1	10	MHz.
	Multiplicador Paralelo 8x8				
	etapa única, registro a registro.	73	37	30	ns.
	Decodificador de Dirección 16 bits	3	4.7	3.9	ns.
	(decodificación interna).				
	Comprobador de Paridad 9 bits.	1	4.3	2.7	ns.

Nota: 1. La mayoría de las funciones son más rápidas en el XC4000EX debido a su más rápida Lógica de Acarreo, conexiones directas, y otras interconexiones adicionales.

Tabla 3.2: Densidad y Desempeño para Varias Funciones Comunes de Circuito en XC4000E[1].

3.2 DESCRIPCIÓN FUNCIONAL.

La arquitectura de los dispositivos de la Serie XC4000 se compone de dos bloques principales: los CLB's y los IOB's como anteriormente se menciono, los cuales a su vez se constituyen de diversos elementos tales como generadores de función, flip – flops, etc., los cuales serán descritos posteriormente.

3.2.1 Bloques de Construcción Básicos.

Los Arreglos de compuertas programables de usuario Xilinx incluyen dos grandes elementos configurables, estos son: los bloques lógicos configurables (CLB's) y los bloques de entrada / salida (IOB's).

- Los CLB's proporcionan los elementos funcionales para construir la lógica del usuario.
- Los IOB's proporcionan la interfaz entre los pines del encapsulado y las líneas de señal internas.

Los otros tres tipos de circuitos incluidos en la arquitectura son:

- Buffers de Tercer Estado (TBUF's) que manejan las largas líneas horizontales asociadas con cada CLB.
- Decodificadores amplios disponibles alrededor de la periferia de cada dispositivo.
- Oscilador interno.

Los recursos de interconexión programable proporcionan las trayectorias de enrutamiento para conectar las entradas y salidas de estos elementos configurables a las redes apropiadas.

La funcionalidad de cada bloque del circuito se personaliza durante la configuración programando las celdas de memoria estática internas. Los valores almacenados en estas celdas de memoria determinan las funciones lógicas y las interconexiones implementadas en el FPGA.

Cada uno de estos circuitos se describe en esta sección.

3.2.2 Bloques Lógicos Configurables (CLB's).

Los Bloques Lógicos Configurables implementan la mayoría de la lógica en un FPGA. Los principales elementos del CLB son:

- Generadores de Función.
- Flip – Flops y
- Multiplexores, tal como se muestra en la Figura 3.2.

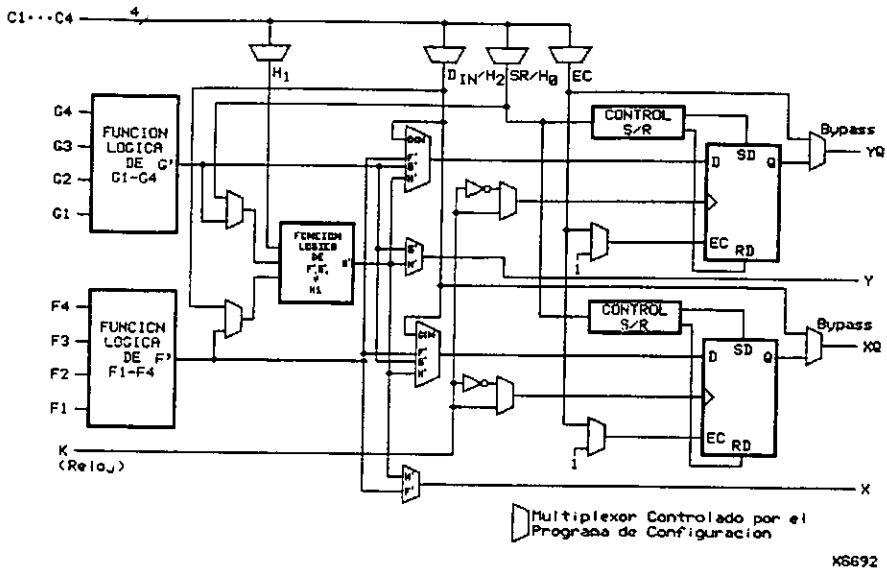


Figura 3.2: Diagrama de Bloques Simplificado del CLB de la Serie XC4000 (RAM y funciones lógicas de acarreo no se muestran).

Los generadores de función F Y G de 4 entradas ofrecen versatilidad sin restricción, ya que la mayoría de las funciones lógico combinacionales necesitan cuatro o menos entradas. Sin embargo, se proporciona un tercer generador de función (H). El generador de función H tiene tres entradas. Ya sea, la entrada cero, la uno, o ambas pueden ser las salidas de F y G; la (s) otra (s) entrada (s) se dirigen fuera del CLB. Por consiguiente, el CLB puede implementar funciones de hasta nueve variables.

Cada CLB contiene dos elementos del almacenamiento (D1 y D2) que pueden usarse para almacenar las salidas del generador de función. Sin embargo, los elementos de almacenamiento y los generadores de función pueden usarse independientemente. Estos elementos de almacenamiento pueden configurarse como flip – flops en los dispositivos XC4000E y XC4000EX; en los XC4000EX estos pueden configurarse opcionalmente como latches. La salida del multiplexor DIN puede usarse como una entrada directa a cualquiera de los dos elementos de almacenamiento, mientras que la salida del multiplexor H1 puede manejar al otro elemento de almacenamiento a través del generador de función H. Las salidas del generador de función pueden

también manejar dos salidas independientes de los elementos de almacenamiento de salida. Esta versatilidad aumenta la capacidad lógica y simplifica el enrutamiento.

Trece entradas del CLB ($C_1 - C_4$, $G_1 - G_4$ y $F_1 - F_4$, K) y cuatro salidas del CLB (YQ, Y, XQ, X) proporcionan acceso a los generadores de función y a los elementos de almacenamiento. Estas entradas y salidas están conectadas a los recursos de interconexión programable afuera del bloque.

a) Generadores de Función.

Se proporcionan cuatro entradas independientes en cada uno de los dos generadores de función (F1 – F4 y G1 – G4). Estos generadores de función, con salidas etiquetadas F' y G', son capaces cada una, de implementar cualquier función Booleana arbitraria definida para cuatro entradas. Los generadores de función se implementan como tablas de "Look – Up" (de búsqueda). El retardo de propagación es por lo tanto independiente de la función implementada.

Un tercer generador de función, etiquetado con H', puede implementar cualquier función Booleana en sus tres entradas. Dos de estas entradas pueden ser opcionalmente las salidas del generador de función F' y G'. Alternativamente, una o ambas de estas entradas pueden venir desde fuera del CLB mediante los multiplexores H2 y H0. La tercera entrada debe venir desde fuera del bloque a través de H1.

Las señales desde los generadores de función pueden salir del CLB en dos salidas. F' o H' pueden conectarse a la salida X. G' o H' pueden conectarse a la salida Y.

Un CLB puede usarse para implementar cualquiera de las siguientes funciones:

- Cualquier función de hasta cuatro variables, más cualquier segunda función de hasta cuatro variables no relacionadas, más cualquier tercera función de hasta tres variables no relacionadas[1].
- Cualquier función única de cinco variables.

1. Cuando se generan tres funciones separadas, debe capturarse una de las salidas de la función en un flip - flop interno del CLB. Sólo dos salidas del generador de función no registradas están disponibles desde el CLB.

- Cualquier función de cuatro variables junto con algunas funciones de seis variables.
- Algunas funciones de hasta nueve variables.

La implementación de funciones amplias en un solo bloque, reduce a la vez el número de bloques requeridos y el retardo en la trayectoria de la señal, alcanzando ambos, incrementos tanto de velocidad como de capacidad.

La versatilidad de los generadores de función del CLB mejoran significativamente la velocidad del sistema. Además, las herramientas de software de diseño pueden tratar de manera independiente cada generador de función. Esta flexibilidad mejora el uso de la celda.

b) Flip – Flops.

El CLB puede pasar su salida o salidas combinacionales a la red de interconexión, pero también puede almacenar los resultados combinacionales u otros datos entrantes en uno o dos flip – flops, y conectar sus salidas a la red de interconexión también.

Los dos flip – flops tipo D disparados por flanco tienen en común las entradas de reloj (K) y la habilitación de reloj (EC). También pueden habilitarse permanentemente cualquiera o ambas entradas de reloj. La funcionalidad del elemento de almacenamiento se describe en la Tabla 3.3.

Modo.	K.	EC.	SR.	D.	Q.
Aplicación de potencia o GSR.	X	X	X	X	SR
Flip – flop.	X	X	1	X	SR
	_/‾	1*	0*	D	D
	0	X	0*	X	Q
Latch.	1	1*	0*	X	Q
	0	1*	0*	D	D
Ambos.	X	0	0*	X	Q

Leyenda:

X	No importa.
_/‾	Flanco ascendente.
SR	Valor Set o Reset. Reset es por omisión.
0 *	La entrada está en estado bajo o desconectada (valor por omisión).
1 *	La entrada está en estado alto o desconectada (valor por omisión).

Tabla 3.3: Funcionalidad del Elemento de Almacenamiento del CLB (se muestra el flanco ascendente activo).

c) Latches (XC4000EX únicamente).

Los elementos de almacenamiento del CLB también pueden configurarse como latches. Los dos latches tienen en común las entradas de reloj (K) y la habilitación de reloj (EC). La funcionalidad del elemento de almacenamiento se describe en la Tabla 3.3.

d) Entrada del Reloj.

Cada flip – flop puede dispararse en el flanco de reloj ascendente o descendente. El pin del reloj es compartido por ambos elementos de almacenamiento. Sin embargo, el reloj es invertido individualmente para cada elemento de almacenamiento. Cualquier inversor colocado en la entrada del reloj es automáticamente detectado en el CLB.

e) Habilitación de Reloj.

La señal de habilitación de reloj (EC) se activa en alto. El pin EC es compartido por ambos elementos de almacenamiento. En caso de que cualquiera quedara sin conectar, la habilitación del reloj para ese elemento de almacenamiento tiene como valor predefinido el estado activo. EC no se invierte dentro del CLB.

f) Set / Reset.

Una entrada (SR) al elemento de almacenamiento asíncrono puede configurarse como set o reset. Esta opción de configuración determina el estado en que cada flip – flop comienza a operar después de la configuración. Y también determina, el efecto de un pulso Set / Reset Global durante el funcionamiento normal, y el efecto de un pulso en el pin SR del CLB. Las tres funciones set / reset para cualquier flip – flop son controladas por el mismo bit de configuración de datos.

El estado set / reset puede especificarse independientemente para cada flip – flop. Esta entrada también puede desactivarse independientemente para cualquier flip – flop.
El estado set / reset es especificado usando el atributo INIT, o colocando el símbolo de biblioteca apropiado para el set o reset de un flip – flop.

SR es activo en alto. Este no se invierte dentro del CLB.

g) Set / Reset Global.

Una línea Set / Reset Global separada (no mostrada en la Figura 3.2) establece (set) o limpia (clear) cada elemento de almacenamiento durante la aplicación de potencia, reconfiguración, o cuando una red "Reset" dedicada está activa. Esta red global (GSR) no compite con otros recursos de enrutamiento; esta usa una red de distribución dedicada.

Cada flip – flop se configura globalmente como "set" o "reset" de la misma manera que el set / reset (SR) local es especificado. Por consiguiente, si un flip – flop esta en "set" por SR, también esta en "set" por GSR. Similarmente, un flip – flop con "reset" es "reset" por SR y GSR a la vez.

GSR puede manejarse desde cualquier pin programable por el usuario como una entrada reset global. Para usar esta red global, se debe colocar un "pad" de entrada y un buffer de entrada en el esquemático o en el código HDL, manejando al pin GSR del símbolo STARTUP (vea la Figura 3.3). Una ubicación específica del pin puede asignarse a esta entrada usando un atributo o propiedad de LOC, como con cualquier otro "pad" programable por el usuario. Un inversor puede insertarse opcionalmente después del buffer de entrada para invertir el sentido de la señal Set / Reset Global.

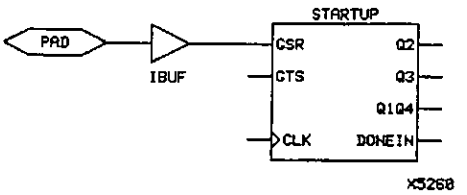


Figura 3.3: Símbolos Esquemáticos para el Set / Reset Global.

Alternativamente, GSR puede manejarse desde cualquier nodo interno.

h) Entradas y Salidas de Datos.

El suministro de un dato de entrada al elemento de almacenamiento es programable. Este es manejado por cualquiera de las funciones F', G', y H', o por el bloque de entrada directo (DIN). Los flip – flops o los latches manejan las salidas XQ y YQ del CLB.

Dos rutas rápidas de alimentación están disponibles, como se muestra en la Figura 3.2. Un multiplexor dos a uno en cada una de las salidas XQ y YQ selecciona de entre una salida del elemento de almacenamiento y cualquiera de las entradas de control. Esta desviación es usada a veces por el enrutador automatizado al reenergizar las señales internas.

i) Señales de Control.

Los multiplexores en el CLB dirigen las cuatro entradas de control (C1 - C4 en la Figura 3.2) hacia las cuatro señales de control internas (H1, DIN / H2, SR / H0, y EC). Cualquiera de estas entradas puede manejar a cualquiera de las cuatro señales de control internas.

Cuando la función lógica se habilita, las cuatro entradas son:

- EC – Habilitación del Reloj.
- SR / H0 – Set / Reset Asíncrono o Entrada 0 del generador de función H.
- DIN / H2 – Entrada directa o Entrada 2 del generador de función H.
- H1 – Entrada 1 del generador de función H.

Cuando la función de memoria se habilita, las cuatro entradas son:

- EC – Habilitación de Reloj.
- WE – Habilitación de Escritura.
- D0 – Dato de entrada al generador de función F y / o G.
- D1 – Dato de entrada al generador de función G (modos 16x1 y 16x2) o el quinto Bit de dirección (modo 32x1).

j) Uso de Flip – Flops y Latches en el FPGA.

La abundancia de flip – flops en la Serie XC4000 propicia diseños "pipelined". Esta es una forma poderosa de incrementar el desempeño, dividiendo la función en pequeñas subfunciones y ejecutándolas en paralelo, pasando los resultados a través de los flip – flops "pipeline".

Para incluir un flip – flop de CLB, se coloca el símbolo de biblioteca apropiado. Por ejemplo, FDCE es un flip – flop tipo D con habilitación de reloj y "clear" asíncrono. El símbolo correspondiente para el latch (para el XC4000EX únicamente) se llama LDCE.

En los dispositivos de la Serie XC4000, los flip – flops pueden usarse como registros o registros de corrimiento sin impedir que los generadores de función tengan un desempeño diferente, debido a una tarea no relacionada. Esta habilidad aumenta la capacidad funcional de los dispositivos.

El tiempo de establecimiento del CLB se especifica entre las entradas del generador de función y la entrada de reloj K. Por lo tanto, el tiempo de establecimiento del flip – flop del CLB incluye el retardo a través del generador de función.

k) Uso de los Generadores de Función como RAM.

Los modos opcionales para cada CLB forman tablas de búsqueda en los generadores de función F' y G' usándolos como un arreglo de celdas de memoria de Lectura / Escritura. Los modos disponibles son: de nivel sensitivo (similar a las familias XC4000 / A / H), disparo por flanco, y disparo por flanco puerto doble. Dependiendo del modo seleccionado, un solo CLB puede configurarse como un arreglo de bits 16x2, 32x1, o 16x1.

En la Tabla 3.4 se muestran los modos de configuración y temporización de memoria que soportan los CLB's para los modos de puerto sencillo y doble.

	16 x 1	**16 x 2**	**32 x 1**	**Temporización de Disparo por Flanco.**	**Temporización de Nivel Sensitivo.**
Puerto Sencillo.	✔	✔	✔	✔	✔
Puerto Doble.	✔			✔	

Tabla 3.4: Modos RAM Soportados.

Los dispositivos de la Serie XC4000 son los primeros dispositivos de la lógica programable con flanco de disparo (síncrono) y puerto doble RAM accesible al usuario. El flanco de disparo RAM simplifica la temporización del sistema. El puerto doble RAM duplica la eficiencia en las aplicaciones FIFO. Estas características pueden programarse individualmente en cualquier CLB de la Serie XC4000.

Ventajas de la RAM en el Chip Disparada por Flanco.

La RAM en el chip es sumamente rápida. El tiempo de acceso de lectura es igual al retardo lógico. El tiempo de acceso de escritura es ligeramente más lento. Los dos tiempos de acceso son mucho más rápidos que cualquier solución fuera del chip, ya que estos evitan los retardos de I / O.

La RAM disparada por flanco, también llamada RAM síncrona, es una característica nunca antes disponible en un Arreglo de Compuertas Programables de Campo. La simplicidad de diseñar RAM disparada por flanco, y el desempeño alcanzado, notablemente más alto, agregan una mejora significativa por encima de los dispositivos existentes con RAM en el chip.

Opciones de Configuración RAM.

Los generadores de función en cualquier CLB pueden configurarse como arreglos RAM en los siguientes tamaños:

- Dos RAM's 16x1: dos entradas de datos y dos salidas de datos con idéntico o, si se prefiere, diferente direccionamiento para cada RAM.
- Una RAM 32x1: una entrada de datos y una salida de datos.

Un generador de función F o G puede configurarse como una RAM 16x1 mientras los otros generadores de función se usan para implementar cualquier función de hasta 5 entradas.

Adicionalmente, la RAM de la Serie XC4000 puede tener cualquiera de los dos modos de temporización siguientes:

- Flanco de disparo (Síncrono): los datos son escritos por el flanco de reloj designado de CLB. WE actúa como una verdadera habilitación de reloj.
- Nivel Sensitivo (Asíncrono): una señal WE externa actúa como un "Write strobe".

El modo de temporización seleccionado se aplica a ambos generadores de función dentro de un CLB cuando ambos se configuran como RAM.

El número de puertos de lectura también es programable:

- Puerto Sencillo: cada generador de función tiene un puerto común de lectura y escritura.
- Puerto Doble: ambos generadores de función se configuran juntos como una sola RAM 16x1 de puerto doble con un puerto de escritura y dos puertos de lectura. Simultáneamente las operaciones de lectura y escritura soportan las mismas o diferentes direcciones.

Las opciones de configuración RAM son seleccionadas colocando el símbolo de biblioteca apropiado.

Selección de un Modo de Configuración RAM.

La elección apropiada del modo RAM para un determinado diseño deberá basarse en los recursos y temporizaciones requeridas, funcionalidad deseada, y la simplicidad del proceso de diseño. Los usos recomendados se muestran en la Tabla 3.5.

	Nivel Sensitivo.	Flanco de Disparo.	Puerto doble con Flanco de Disparo.
Para Nuevos Diseños se debe Usar.	No.	Si.	Si.
Tamaño (16x1, Registrado).	1 / 2 CLB.	1 / 2 CLB.	1 CLB
Lectura/Escritura Simultánea.	No.	No.	Si.
Desempeño Relativo.	X.	2X.	2X (efectivo 4x).

Tabla 3.5: Selección de Modo RAM.

La diferencia entre el nivel sensitivo, el flanco de disparo, y el puerto doble RAM es sólo en la operación de escritura. La operación de lectura y la temporización son idénticas para todos los modos de operación.

Entradas y Salidas RAM.

Las entradas a los generadores de función F1 – F4 y G1 – G4 actúan como líneas de dirección, seleccionando una celda particular de memoria en cada tabla de búsqueda.

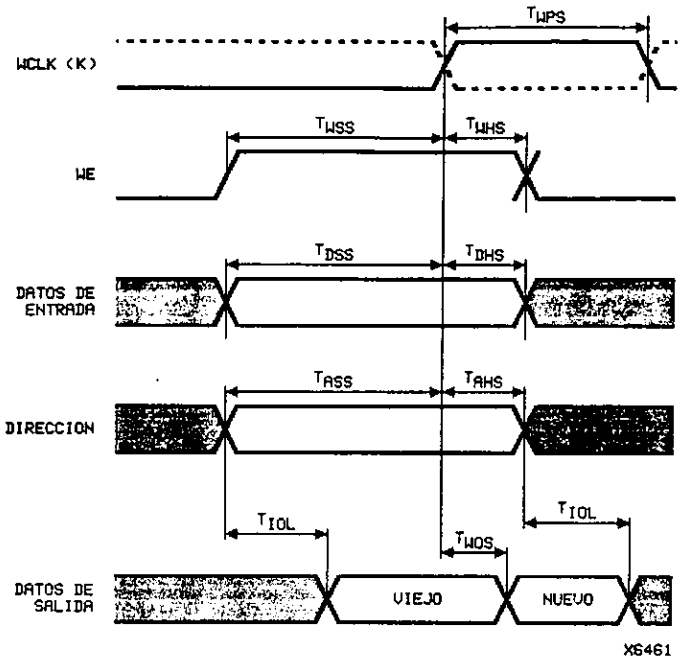
La funcionalidad de las señales de control del CLB cambia cuando los generadores de función se configuran como RAM. Las líneas DIN / H2, H1, y SR / H0 se convierten en las dos entradas de los datos (D0, D1) y la entrada de Habilitación de Escritura (WE) para la memoria 16x2, respectivamente. Cuando la configuración 32x1 se selecciona, D1 actúa como el quinto bit de dirección y D0 es la entrada de datos.

Los contenidos de la(s) celda(s) de memoria(s) siendo direccionados están disponibles en las salidas del generador de función F' y G'. Ellos pueden salir del CLB mediante sus salidas "X" y "Y", o pueden capturarse en el flip – flop o flip – flops del CLB.

Configurando los generadores de función del CLB como memoria de Lectura / Escritura no se afecta la funcionalidad de las otras porciones del CLB, con excepción de la redefinición de las señales de control. En los modos 16x2 y 16x1, el generador de función H' puede usarse para implementar funciones Booleanas de F', G', y D1, y los flip – flops D como latch de las señales F', G', H', o D0.

Modo de Puerto Sencillo Disparado por Flanco.

El flanco de disparo (síncrono) RAM simplifica los requerimientos de temporización. La temporización del flanco de disparo RAM de la Serie XC4000 opera como escritor de un registro de datos. Estando los datos y direcciones presentes. El registro se habilita para escribir mediante un estado alto en la entrada de habilitación para escritura, WE. Entonces, un flanco de reloj descendente o ascendente carga los datos en el registro, como se muestra en la Figura 3.4.



Nota: El pulso que sigue al flanco de disparo de WCLK (T_{WPS} en la Figura) debe ser de menor amplitud que un milisegundo. Para la mayoría de las aplicaciones, este requisito no es demasiado restrictivo; sin embargo, no se debe olvidar. En este punto, detener WCLK en el ciclo de escritura podría producir una corriente excesiva e incluso podría dañar a los dispositivos más grandes si muchos CLB's se configuran como flanco de disparo RAM.

Figura 3.4: Temporización de Escritura de Flanco de Disparo RAM.

Las complejas relaciones de temporización entre las señales de dirección, datos, y habilitación de escritura no son requeridas, y el pulso de habilitación de escritura externo se vuelve una simple habilitación de reloj. El flanco de disparo de WCLK mantiene la dirección, los datos de entrada y la señal WE. Entonces un pulso de escritura interno se genera para realizar la escritura. Vea la Figura 3.5 y la Figura 3.6 para los diagramas de bloques de un CLB configurado con flanco de disparo 16x2 y 32x1, puerto sencillo RAM.

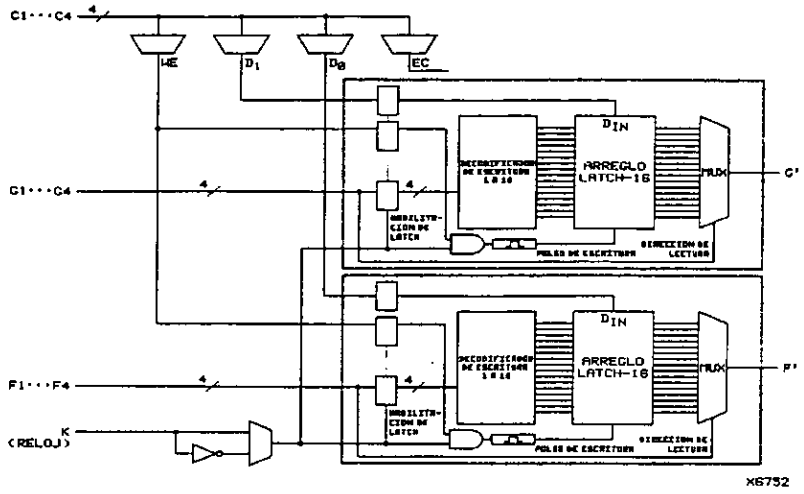


Figura 3.5: 16x2 (o 16x1) Puerto Sencillo RAM con Flanco de Disparo.

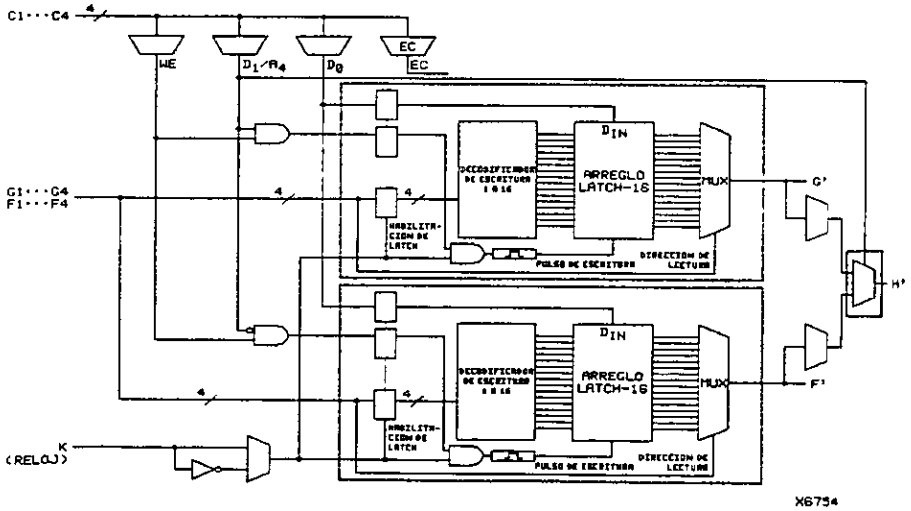


Figura 2.6: 32x1 Puerto Sencillo RAM con Flanco de Disparo (las direcciones F y G son idénticas).

Las relaciones entre los pines del CLB y las entradas y salidas de la RAM para puerto sencillo en modo de flanco de disparo se muestran en la Tabla 3.6.

Señal RAM.	Pin del CLB.	Función.
D.	D0 o D1 (16x2, 16x1). D0 (32x1).	Entrada de Datos.
A[3:0].	F1 – F4 o G1 – G4.	Dirección.
A[4].	D1 (32x1).	Dirección.
WE.	WE.	Habilitación de Escritura.
WCLK.	K.	Reloj.
SPO. (Salida de Datos).	F' o G'.	Puerto Sencillo de Salida. (Salida de Datos).

Tabla 3.6: Señales de Flanco de Disparo RAM Puerto Sencillo.

La entrada de Reloj de Escritura (WCLK) puede configurarse como "activa" en el flanco ascendente (valor por omisión) o en el flanco descendente. Esta usa el mismo pin del CLB (K) empleado para el reloj de los flip – flops del CLB, pero puede invertirse de forma independiente. Por consiguiente, la salida RAM puede opcionalmente almacenarse dentro del mismo CLB por el mismo flanco de reloj como la RAM, o por el flanco opuesto de este reloj. El sentido de WCLK se aplica a ambos generadores de función en el CLB cuando ambos se configuran como RAM.

El pin WE es activo en alto y no se invierte dentro del CLB.

Modo de Puerto Doble Disparado por Flanco.

En modo de puerto doble, ambos generadores de función F y G se usan para crear un arreglo sencillo RAM 16x1 con un puerto de escritura y dos puertos de lectura. El arreglo RAM resultante puede leerse y escribirse simultáneamente en dos direcciones independientes. Simultáneamente las operaciones de lectura y escritura también soportan la misma dirección.

El modo de puerto doble siempre tiene temporización de escritura de flanco de disparo, como se muestra en la Figura 3.4.

La Figura 3.7 muestra un modelo simple de un CLB de la Serie XC4000 configurado como puerto doble RAM. Un puerto de dirección, etiquetado como A[3:0], proporciona ambas direcciones de lectura y escritura para el generador de función F. Este generador de función se comporta igual que un arreglo RAM de puerto sencillo 16x1 disparado por flanco. La salida RAM, Salida de Puerto Sencillo (SPO), aparece a la salida del generador de función F. SPO, por lo tanto, refleja los datos de la dirección A[3:0].

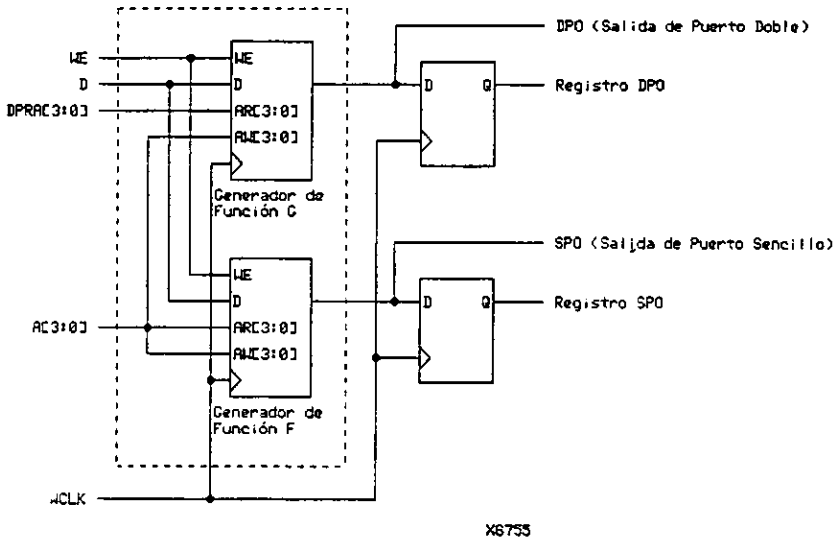


Figura 3.7: Puerto Doble RAM de la Serie XC4000, Modelo Simple.

El otro puerto de dirección, etiquetado como DPRA[3:0] para Dirección de Lectura de Puerto Doble, suministra la dirección de lectura para el generador de función G. La dirección de lectura para el generador de función G, sin embargo, viene desde la dirección A[3:0]. La salida desde este arreglo RAM 16x1, Salida de Puerto Doble (DPO), aparece a la salida del generador de función G. Por consiguiente, DPO refleja los datos a la dirección DPRA[3:0].

Por lo tanto, usando A[3:0] para la dirección de escritura y DPRA[3:0] para la dirección de lectura, y leyendo solo la salida DPO, una FIFO que pueda leerse y escribirse simultáneamente se genera fácilmente. El acceso simultáneo duplica la eficiencia de la FIFO.

Las relaciones entre los pines del CLB y las entradas y salidas RAM para puerto doble, en modo de flanco de disparo se muestran en la Tabla 3.7.

Señal RAM.	Pin del CLB.	Función.
D.	D0.	Entrada de Datos.
A[3:0].	F1 – F4.	Dirección de Lectura para F, Dirección de Escritura para F y G.
DPRA[3:0].	G1 – G4.	Dirección de Lectura para G.
WE.	WE.	Habilitación de Escritura.
WCLK.	K.	Reloj.
SPO.	F'.	Salida de Puerto Sencillo (direccionado en A[3:0]).
DPO.	G'	Salida de Puerto Doble (direccionado en DPRA[3:0]).

Tabla 3.7: Señales de Flanco de Disparo RAM Puerto Doble.

Vea la Figura 3.8 para un diagrama de bloques de un CLB configurado en este modo.

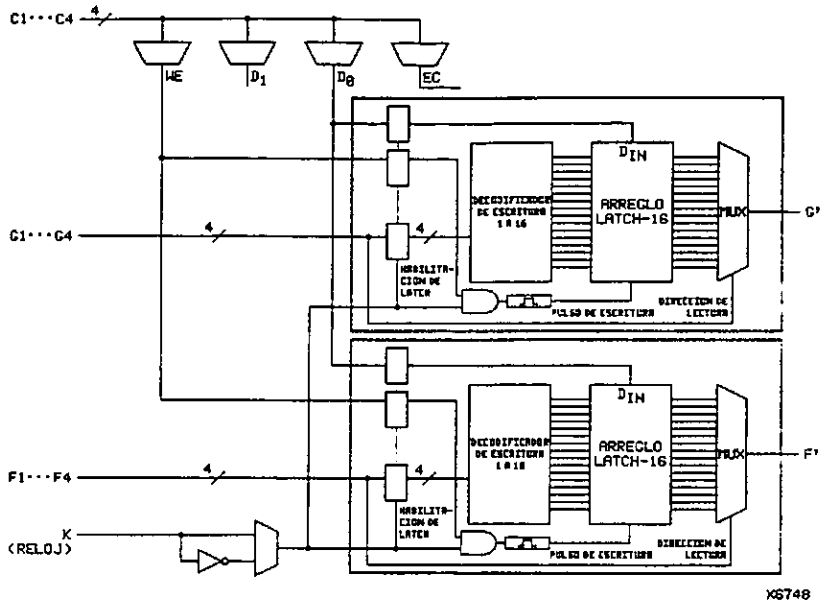


Figura 3.8: Puerto Doble RAM 16x1 Disparado por Flanco.

Modo de Temporización de Nivel Sensitivo Puerto Sencillo.

Nota: El modo de Flanco de disparo se recomienda para todos los diseños nuevos. El modo de nivel sensitivo, también llamado modo asíncrono, aún es soportado para la Serie XC4000 anterior en compatibilidad con la familia XC4000.

La temporización RAM nivel sensitivo es simple en concepto pero puede complicarse en la ejecución. Estando las señales de datos y direcciones presentes, entonces un pulso positivo en el pin de habilitación de escritura (WE) realiza una escritura en la RAM en la dirección designada. Como se indica por la etiqueta de "nivel sensitivo", esta RAM actúa como un latch. Durante el pulso en estado alto WE, cambiar las líneas de datos resulta en un nuevo dato escrito en la dirección anterior. Cambiar las líneas de dirección mientras WE esta en alto resulta en datos inexactos escritos en la nueva dirección y posiblemente en otras direcciones también, debido a que las líneas de dirección inevitablemente no cambian simultáneamente.

El usuario debe generar una cuidadosa temporización de la señal WE. El retardo en la señal WE y en las líneas de dirección debe verificarse cuidadosamente para asegurar que WE no se active hasta después de que las líneas de dirección se hayan establecido, y que WE continúe inactivo antes de que las líneas de dirección cambien de nuevo. Los datos deben ser estables antes y después del flanco descendente de WE.

En términos prácticos, WE es generado normalmente por un reloj 2X. Si un reloj 2X no está disponible, el flanco descendente del reloj del sistema puede usarse. Sin embargo, existen riesgos inherentes en este enfoque, ya que el pulso WE debe garantizarse que esté inactivo antes del siguiente flanco ascendente del reloj del sistema.

Sin embargo, la RAM disparada por flanco disponible en la Serie XC4000 es superior a la RAM de nivel sensitivo para casi cualquier aplicación.

La Figura 3.9 muestra la temporización de escritura para el nivel sensitivo, RAM puerto sencillo.

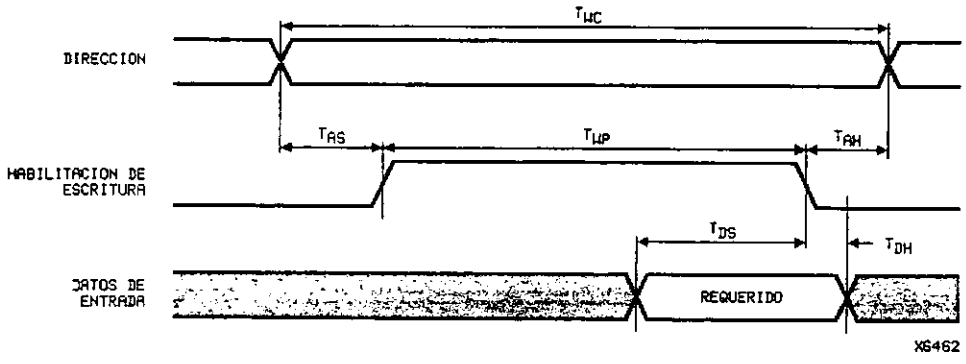


Figura 3.9: Temporización de Escritura RAM Nivel Sensitivo.

Las relaciones entre los pines del CLB y las entradas y salidas de la RAM para el modo de nivel sensitivo puerto sencillo se muestran en la Tabla 3.8.

Señal RAM.	Pin del CLB.	Función.
D.	D0 o D1.	Entrada de Datos.
A[3:0].	F1 – F4 o G1 – G4.	Dirección.
WE.	WE.	Habilitación de Escritura.
O.	F' o G'.	Salida de Datos.

Tabla 3.8: Señales de Nivel Sensitivo RAM Puerto Sencillo.

Las Figuras 3.10 y 3.11 muestran los diagramas de bloques de un CLB configurado como RAM puerto sencillo, con nivel sensitivo 16x2 y 32x1.

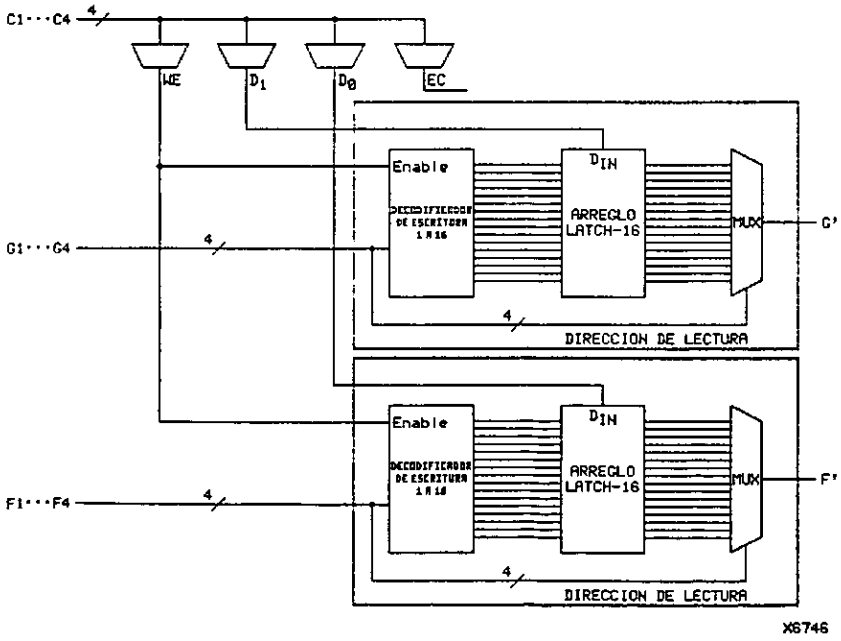


Figura 3.10: Nivel Sensitivo RAM Puerto Sencillo 16x2 (o 16x1).

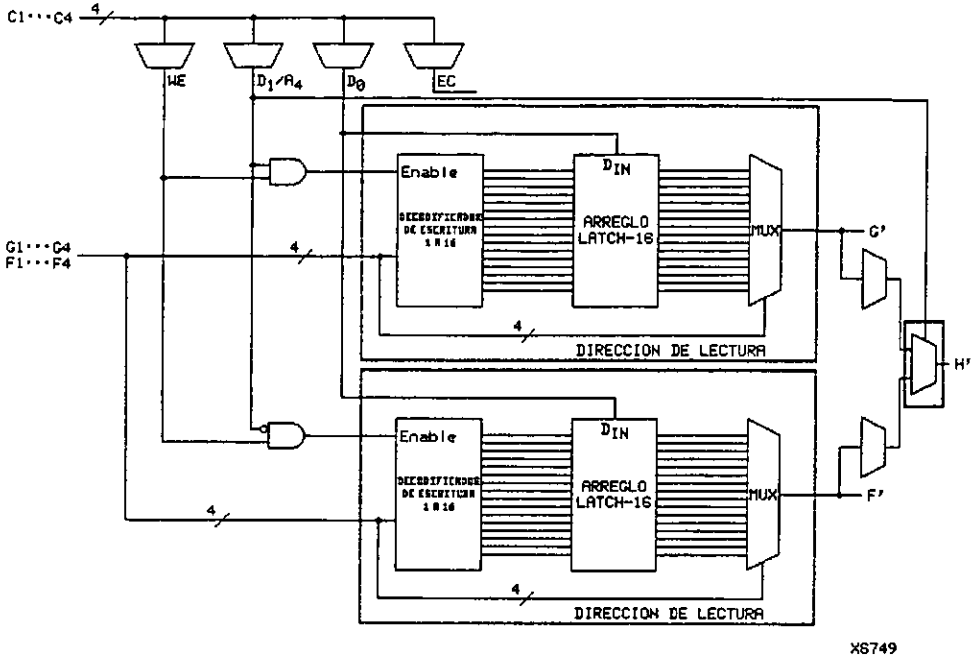


Figura 3.11: Nivel Sensitivo RAM Puerto Sencillo 32x1 (las direcciones F y G son idénticas).

Inicialización de la RAM en la Configuración.

Ambas implementaciones ROM y RAM en los dispositivos de la Serie XC4000 son inicializadas durante la configuración. Los contenidos iniciales se definen por medio de un atributo INIT o propiedad adjunta al símbolo ROM o RAM, como se describe en la guía de biblioteca esquemática.

Aunque no se haya definido, todos los contenidos RAM son inicializados a cero, como valor por omisión.

La inicialización de la RAM ocurre únicamente durante la configuración. El contenido de la RAM no es afectado por el Set / Reset Global.

l) Lógica de Acarreo Rápido.

Cada generador de función F y G del CLB contiene una lógica aritmética dedicada para la rápida generación de señales de "acarreo" y "préstamo". Esta salida extra se pasa al siguiente generador de función en el CLB adyacente. La cadena de acarreo es independiente de los recursos normales de enrutamiento.

La lógica de acarreo rápida dedicada aumenta la eficiencia y desempeño de los sumadores, restadores, acumuladores, comparadores y contadores. Esto también abre la puerta a muchas aplicaciones nuevas que involucran operaciones aritméticas, donde las generaciones previas de FPGA's no tenían la rapidez suficiente o eran demasiado ineficientes. Los cálculos de desplazamiento de dirección a alta velocidad en sistemas gráficos o de microprocesador, y la rama a alta velocidad en el procesamiento digital de señales, son dos aplicaciones típicas.

Los dos generadores de función de 4 entradas pueden configurarse como un sumador de 2 bits con construcción de acarreo interno que puede extenderse a cualquier longitud. Esta circuitería de acarreo dedicado es tan rápida y eficaz que los métodos de aceleración convencionales como la generación / propagación de acarreo es incluso insignificante para el nivel de 16 bits, y de beneficio marginal al nivel de 32 bits.

Esta lógica de acarreo rápida es uno de los aspectos más importantes de la Serie XC4000, con aritmética acelerada y un conteo interno de un rango de los 70 MHz.

La cadena de acarreo en los dispositivos XC4000E puede correr hacia arriba o hacia abajo. En la parte superior e inferior de las columnas donde ya no hay CLB's ni arriba ni abajo, el acarreo se propaga a la derecha (Ver la Figura 3.12). A fin de mejorar la velocidad en los dispositivos XC4000EX de alta capacidad, los cuales pueden tener potencialmente cadenas de acarreo muy largas, la cadena de acarreo únicamente viaja en forma ascendente, como se muestra en la Figura 3.13. Esta restricción deberá tener poco impacto, porque el dispositivo XC4000EX más pequeño, el XC4028EX, puede acomodar una cadena de acarreo de 64 bits en una sola columna. Adicionalmente, la interconexión estándar puede usarse para enrutar una señal de acarreo en una dirección descendente.

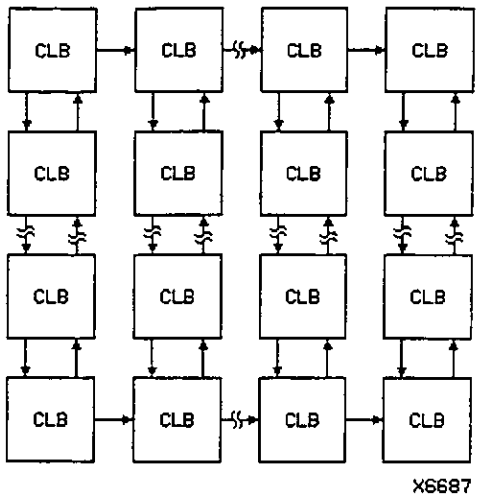


Figura 3.12: Rutas Disponibles de Propagación de Acarreo en los XC4000E.

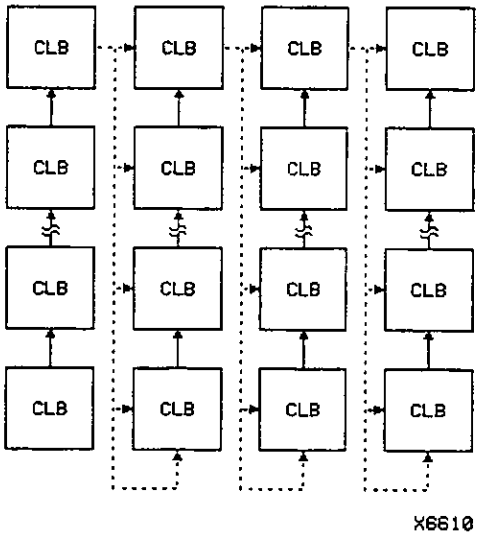


Figura 3.13: Rutas Disponibles de Propagación de Acarreo en los XC4000EX (las líneas punteadas se usan generalmente para interconexiones).

La Figura 3.14 muestra un CLB XC40000E con lógica de acarreo rápida dedicada. La lógica de acarreo en los XC4000EX es similar, excepto que COUT sólo sale en la parte superior, y la señal CINDOWN no existe. Como se muestra en la Figura 3.14, la lógica de acarreo comparte operandos y entradas de control con los generadores de función. Las salidas de acarreo se conectan a los generadores de función, donde se combinan con los operandos para formar las sumas.

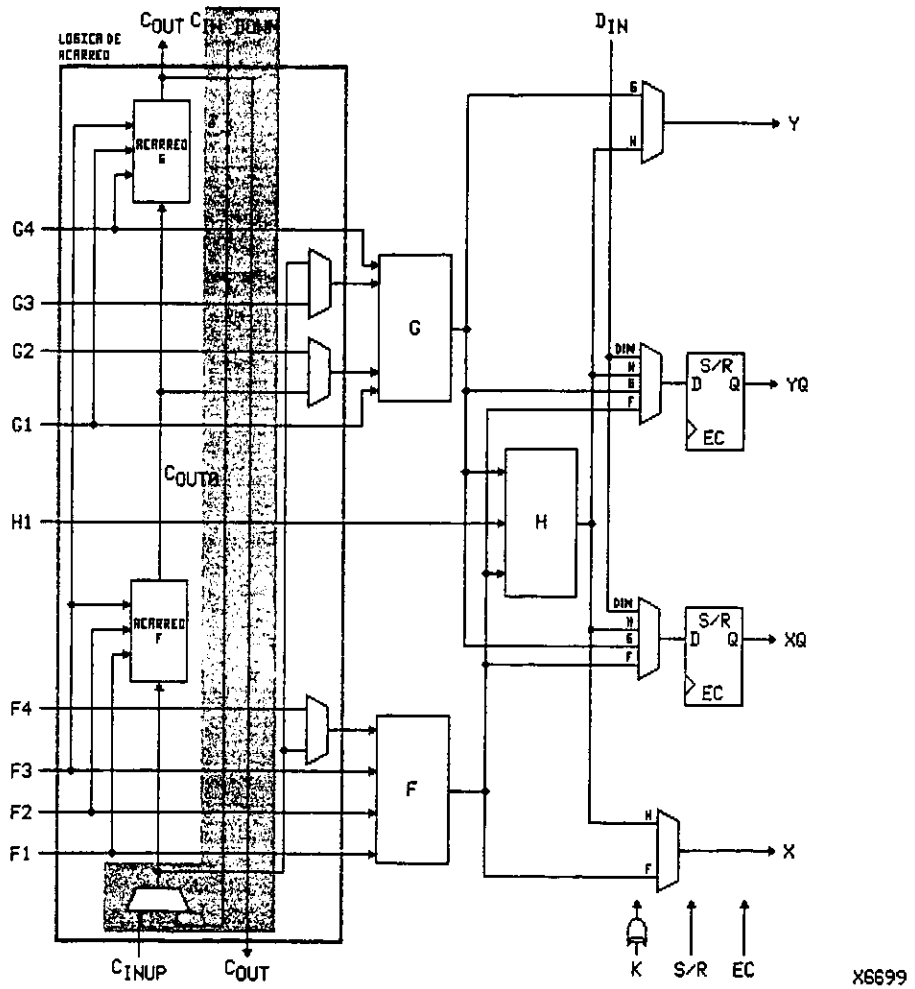


Figura 3.14: Lógica de Acarreo Rápido en el CLB XC4000E (el área sombreada no está presente en el XC4000EX).

La Figura 3.15 y la Figura 3.16 muestran los detalles de la lógica de acarreo para los XC4000E y los XC4000EX respectivamente. Estos diagramas muestran los contenidos de la caja etiquetada como "LOGICA DE ACARREO" en la Figura 3.14. Como se muestra, la lógica de acarreo XC4000EX elimina un multiplexor para reducir el retardo en el paso a través de la cadena de acarreo. Adicionalmente, el multiplexor en la ruta de G4 tiene ahora una entrada de memoria programable, que permite a G4 conectarse directamente a COUT. Así G4 se convierte en una ruta adicional de inicialización de alta velocidad para el acarreo de entrada.

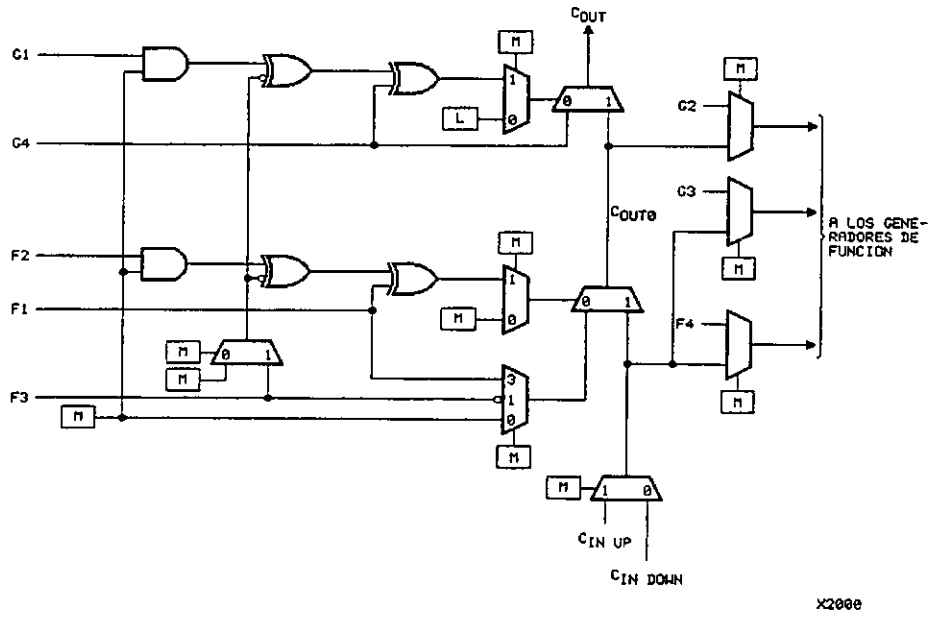


Figura 3.15: Detalle de la Lógica de Acarreo Dedicado XC4000E.

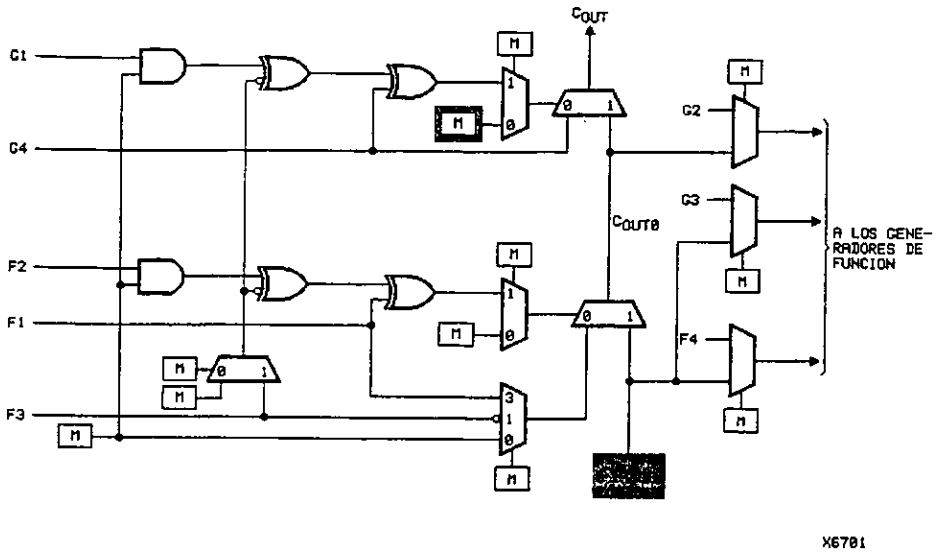


Figura 3.16: Detalle de la Lógica de Acarreo Dedicado XC4000EX (las áreas sombreadas muestran diferencias para la lógica de acarreo XC4000E).

La lógica de acarreo rápida puede ser accesada colocando símbolos de biblioteca especiales, o usando "Xilinx Relationally Placed Macros" (RPM's – Colocación de Macros Relacionadas) que ya incluyen estos símbolos.

3.2.3 Bloques de Entrada / Salida (IOB's).

Los Bloques de Entrada / Salida configurables por el usuario (IOB's) proporcionan la interfaz entre el paquete externo de pines y la lógica interna. Cada IOB controla un pin del encapsulado y puede configurarse para señales de entrada, salida, o bidireccionales.

La Figura 3.17 muestra un diagrama de bloques simplificado de los IOB's XC4000E. Un diagrama más completo de los IOB's XC4000E puede encontrarse en la Figura 3.43, en la sección "Boundary Scan". La Figura 3.43 incluye la lógica "Boundary Scan" en el IOB.

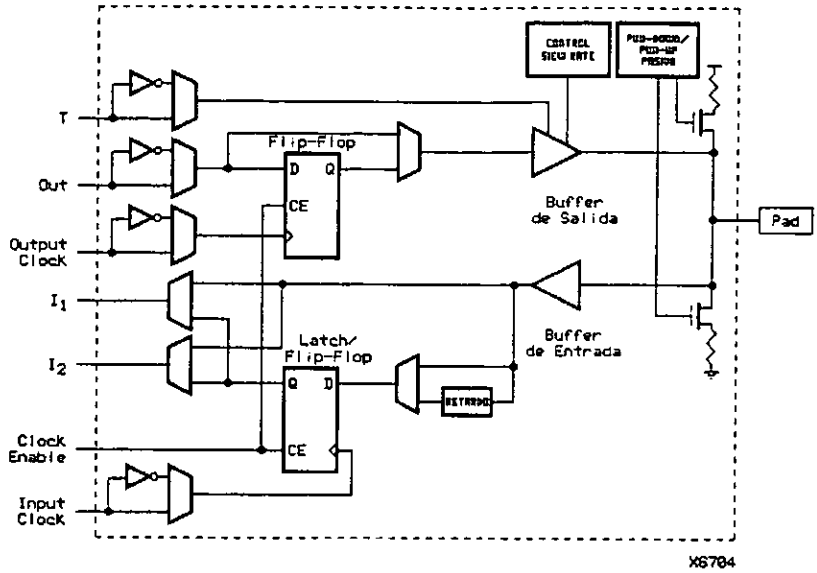


Figura 3.17: Diagrama de Bloques Simplificado del IOB XC4000E.

La Figura 3.18 muestra un diagrama de bloques simplificado de los IOB's XC4000EX. El IOB XC4000EX contiene algunas características especiales no incluidas en los IOB's XC4000E. Estas características se resaltan en la Figura 3.18, y se discuten a lo largo de este capítulo. Cuando las características especiales XC4000EX se discuten, ellas se identifican claramente en el texto. Cualquier característica no identificada se encuentra presente en ambos dispositivos XC4000E y XC4000EX.

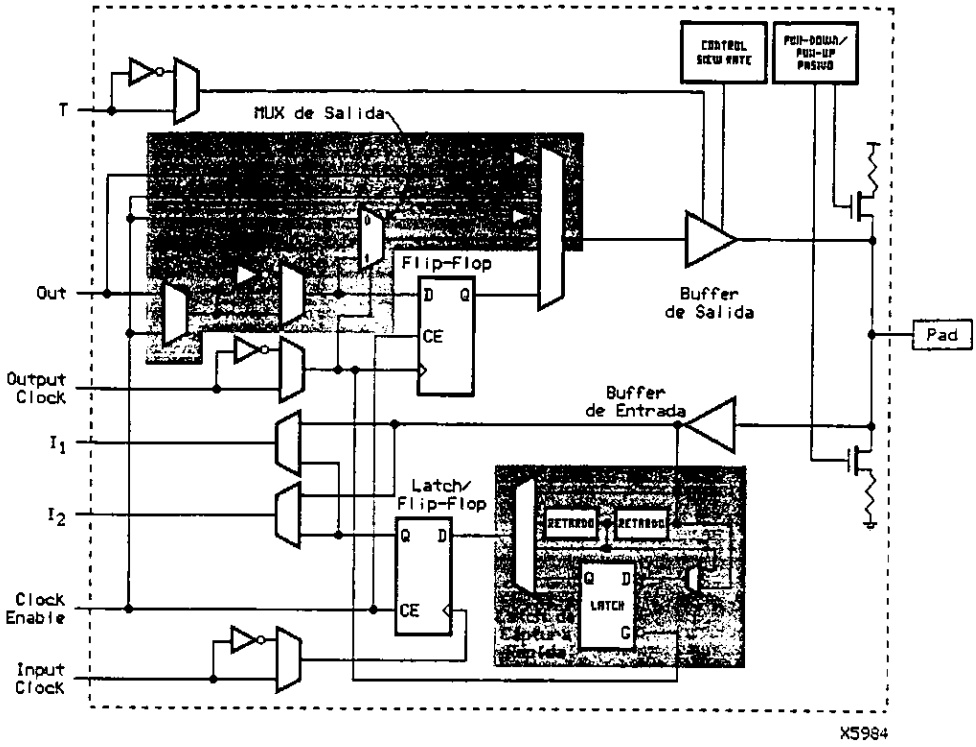


Figura 3.18: Diagrama de Bloques Simplificado del IOB XC4000EX (las áreas sombreadas indican diferencias con respecto al XC4000E).

a) Señales de Entrada al IOB.

Las dos rutas, etiquetadas como I1 e I2 en la Figura 3.17 y en la Figura 3.18, llevan las señales de entrada al interior del arreglo. Las entradas también se conectan a un registro de entrada que puede programarse como un flip – flop disparado por flanco o un latch de nivel sensitivo.

La elección se hace colocando el símbolo de biblioteca apropiado. Por ejemplo, IFD es el flip – flop de entrada básico (flanco de disparo ascendente), e ILD es el latch de entrada básico (alto). Las

variaciones con relojes inversores están disponibles, y algunas combinaciones de latches y flip - flops pueden implementarse en un solo IOB, como se describe en la Guía de Bibliotecas XACT.

Las entradas pueden configurarse globalmente para umbrales TTL (1.2 V., valor por omisión) o para umbrales CMOS, usando una opción en el programa de "MakeBits". Existe una ligera histéresis de aproximadamente 300mV. Los niveles de salida también son configurables; los dos ajustes globales del umbral de la entrada y el nivel de salida son independientes.

Las entradas de los dispositivos de bajo voltaje deben configurarse como CMOS en todo momento. Ellos pueden ser manejados por las salidas de 5 Volts de todos los dispositivos de la Serie XC4000, con tal de que los 5 Volts de las salidas estén en modo TTL. Ellos pueden también ser manejados por cualquier salida TTL que no exceda los 3.7 V. Las salidas de los dispositivos de 5 V. de la familia XC3000, por ejemplo, son compatibles con TTL, ya que el voltaje de salida puede exceder los 3.7 V., pero ellos no pueden usarse para manejar una entrada XC4000L o XC4000XL. Las entradas de los dispositivos de 5 V. de la Serie XC4000 pueden ser manejadas por las salidas de cualquier dispositivo de 3.3 Volts, si las entradas de 5 Volts están en modo TTL.

Los suministros soportados para las entradas de los dispositivos de la Serie XC4000 se muestran en la Tabla 3.9.

	Entradas de la Serie XC4000.		
Suministro.	**3.3 V., CMOS.**	**5 V., TTL.**	**5 V., CMOS.**
Cualquier dispositivo, Vcc = 3.3 V., salidas CMOS.	✔	✔	Datos
Serie XC4000, Vcc = 5 V., salidas TTL.	✔	✔	
Cualquier dispositivo, Vcc = 5 V., salidas TTL (Voh <= 3.7 V.)	✔	✔	Poco fiables.
Cualquier dispositivo, Vcc = 5 V., salidas CMOS.	Peligro[1]	✔	✔

1. El suministro aceptable para los XC4000L sería 5 Volts, pero en caso de llegar al límite (V_{TT}) será restringido a 5 V.

Tabla 3.9: Suministros Soportados para las Entradas de los Dispositivos de la Serie XC4000.

Entradas al Registro.

Las señales I1 e I2 que salen del bloque pueden cada una llevar cualquier señal de entrada de registro o directa.

La entrada y salida de los elementos de almacenamiento tienen en cada IOB una entrada común de habilitación de reloj, que a través de la configuración, puede activarse individualmente para la entrada, la salida o ambas del flip – flop. Esta habilitación de reloj opera exactamente igual que el pin EC en el CLB de la Serie XC4000. No puede invertirse dentro del IOB.

El comportamiento del elemento de almacenamiento se muestra en la Tabla 3.10.

Modo.	**Reloj.**	**Habilitación de Reloj.**	**D.**	**Q.**
Aplicación de potencia o GSR.	X	X	X	SR
Flip – flop.	_/‾	1*	D	D
	0	X	X	Q
Latch.	1	1*	X	Q
	0	1*	D	D
Ambos.	X	0	X	Q

Leyenda:

X	No importa.
_/‾	Flanco ascendente.
SR	Valor Set o Reset. Reset es por omisión.
0 *	La entrada está en estado bajo o desconectada (valor por omisión).
1 *	La entrada está en estado alto o desconectada (valor por omisión).

Tabla 3.10: Funcionalidad de la Entrada al Registro (se muestra el flanco de disparo ascendente).

Entrada Latch adicional para Captura Rápida (XC4000EX únicamente).

El IOB XC4000EX tiene un latch optativo adicional en la entrada. Este latch, como se muestra en la Figura 3.18, es temporizado por el reloj de salida, el reloj usado para la salida del flip – flop, en

lugar del reloj de entrada. Por lo tanto, dos relojes diferentes pueden usarse para temporizar las entradas de los dos elementos de almacenamiento. Este latch adicional permite la muy rápida captura de los datos de entrada, el cual es entonces sincronizado al reloj interno por el flip – flop o latch del IOB.

Para usar esta técnica de Captura Rápida, maneje el pin de reloj de salida (la Señal "Latching" de Captura Rápida) desde la salida de un buffer "Global Early" (Anticipado Global) o un buffer "FastCLK" proporcionados en los XC4000EX. El segundo elemento de almacenamiento deberá ser temporizado por un buffer "Low – Skew" (Bajo Devío) Global, para sincronizar los datos entrantes a la lógica interna (vea la Figura 3.19). Estos buffers especiales se describen en "Redes Globales y Buffers (XC4000EX únicamente)".

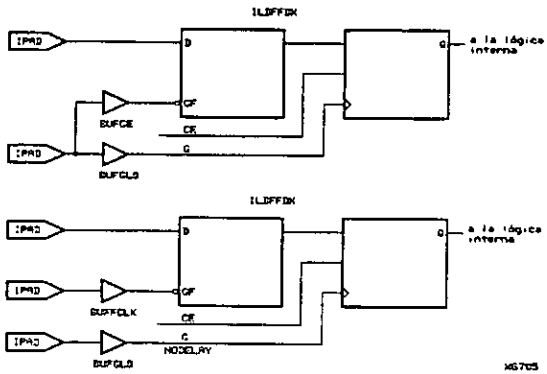


Figura 3.19: Ejemplos que Usan Latch de Captura Rápida en los XC4000EX.

El latch de Captura Rápida es diseñado principalmente para usarse con un buffer "Global Early". Para la Captura Rápida, una sola señal de reloj se enruta a través de un buffer "Global Early" y un buffer "Low – Skew" Global (los dos buffers comparten un mismo "pad" de entrada). El latch de Captura Rápida es temporizado por el buffer "Global Early", y el flip – flop o latch del IOB estándar es temporizado por el buffer "Low – Skew" Global. Este modo es la manera más segura de usar el latch de Captura Rápida, ya que los buffers del reloj en ambos elementos de almacenamiento son manejados por el mismo "pad". No existe desviación externa entre los "pad's" del reloj como para crear problemas potenciales.

Alternativamente, un buffer "FastCLK" puede usarse para minimizar el tiempo de establecimiento de las entradas del dispositivo, si el tiempo de retención positivo es aceptable. Se debe usar el

buffer "FastCLK" como reloj para el latch de Captura Rápida, y un buffer de reloj más lento como reloj para el flip – flop o latch del IOB estándar. Cualquier buffer "Global Early" o buffer "Low – Skew" Global puede usarse para el segundo elemento de almacenamiento, pero cualquiera que se use debe ser el mismo reloj como en la lógica interna relacionada. Puesto que los "pad's FastCLK" son diferentes a los "pad's Global Early" y a los "pad's Low – Skew Globales", se debe tener cuidado en asegurar que esa desviación externa al dispositivo no cree dificultades de temporización internas.

Para colocar el latch de Captura Rápida en un diseño, se debe usar uno de los símbolos de biblioteca especiales, ILDFFDX o ILDFLDX. ILDFFDX es un latch de Captura Rápida en estado bajo seguido por un flip – flop de entrada activa en alto. ILDFLDX es un latch de Captura Rápida de estado bajo seguido por un latch de entrada de estado alto. Cualquiera de las entradas de reloj puede invertirse antes de manejar al elemento de biblioteca, y el inversor es absorbido en el IOB. En caso de que una sola salida BUFG se use para manejar ambas entradas de reloj, el software corre automáticamente el reloj a través de un buffer "Low – Skew" Global y un buffer "Global Early", y relojes de latch de Captura Rápida apropiadamente.

También la Figura 3.18 muestra dos opciones de derivación del retardo en la entrada. Por omisión, si el latch de Captura Rápida se usa, el software de Xilinx asume un buffer "Global Early" manejando al reloj, y selecciona MEDDELAY para asegurar un tiempo de retención cero. Este valor por omisión puede anularse quitando el retardo, si se usa "FastClk", uniendo un atributo o propiedad NODELAY al ILDFFD o al latch ILDFLD.

b) Señales de Salida del IOB.

Las señales de salida pueden invertirse opcionalmente dentro del IOB, y pueden pasar directamente al "pad" o ser almacenadas en un flip – flip disparado por flanco. La funcionalidad de este flip – flop se muestra en la Tabla 3.11.

Modo.	Reloj.	Habilitación de Reloj.	T.	D.	Q.
Aplicación de potencia o GSR.	X	X	0*	X	SR
Flip – flop.	X	0	0*	X	Q
	_/‾	1*	0*	D	D
	X	X	1	X	Z
	0	X	0*	X	Q

Leyenda:

X	No importa.
_/‾	Flanco ascendente.
SR	Valor Set o Reset. Reset es por omisión.
0 *	La entrada está en estado bajo o desconectada (valor por omisión).
1 *	La entrada está en estado alto o desconectada (valor por omisión).
Z	Tercer Estado.

Tabla 3.11: Funcionalidad del Flip – Flop de Salida (se muestra el flanco ascendente activo).

Una señal de tercer estado activa en alto puede usarse colocando un buffer de salida en un estado de alta impedancia, implementando salidas de tercer estado o I / O bidireccionales. Bajo el control de configuración, en la salida (OUT) y en la salida de tercer estado (T) las señales pueden invertirse. La polaridad de estas señales se configura independientemente para cada IOB.

Los 4 mA. máximos de corriente de salida especificados de muchos FPGA's frecuentemente forzan al usuario a agregar buffers externos, los cuales son especialmente engorrosos en líneas I / O bidireccionales. Los dispositivos XC4000E y XC4000EX resuelven muchos de estos problemas proporcionando una corriente de salida garantizada de 12 mA. Pueden interconectarse externamente dos salidas adyacentes para proporcionar hasta 24 mA. (las salidas XC4000L y XC4000XL pueden proporcionar hasta 4 mA., y dos salidas adyacentes XC4000L y XC4000XL pueden proporcionar hasta 8 mA.). Los FPGA's XC4000E y XC4000EX pueden así manejar directamente buses sobre una placa de circuito impreso.

Por omisión, la estructura "pull – up" de salida se configura como un "totem – pole" TTL. El manejo del estado alto lo realiza un transistor "pull – up" de canal n, conectado a un voltaje de umbral del transistor por debajo de Vcc. Alternativamente, las salidas pueden configurarse globalmente como manejadores CMOS, con transistores "pull – up" canal p conectados a Vcc. Esta opción de "MakeBits" se aplica a todas las salidas del dispositivo. No es programable individualmente.

Las salidas de los dispositivos de bajo voltaje deben configurarse como CMOS en todo momento. Ellos pueden manejar las entradas de cualquier dispositivo de 5 Volts con los umbrales compatibles TTL.

Cualquier dispositivo de 5 Volts de la Serie XC4000 con sus salidas configuradas en modo TTL puede manejar las entradas de cualquier dispositivo de 3.3 Volts típico.

Los destinos soportados para las salidas de los dispositivos de la Serie XC4000 se muestran en la Tabla 3.12.

	Salidas de la Serie XC4000.		
Destino.	3.3 V., CMOS.	5 V., TTL.	5 V., CMOS.
Cualquier dispositivo típico Vcc = 3.3 V., entradas de umbral CMOS.	✔	✔	Alguno[1]
Cualquier dispositivo, Vcc = 5 V., entradas de umbral TTL.	✔	✔	✔
Cualquier dispositivo, Vcc = 5 V., entradas de umbral CMOS.	Datos Poco fiables.		✔

1. Sólo si el dispositivo de destino tiene entradas que toleren 5 V.

Tabla 3.12: Destinos Soportados para las Salidas de la Serie XC4000.

Una salida puede configurarse como "Open – Drain" (Colector Abierto) colocando un símbolo OBUFT en un esquemático o código HDL, uniendo entonces el pin de tercer estado (T) a la señal de salida, y el pin de entrada (I) conectado a tierra (vea la Figura 3.20).

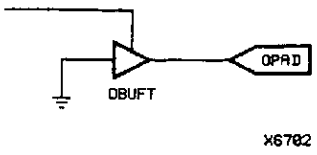


Figura 3.20: Salida "Open – Drain" (Colector Abierto).

Salida "Slew – Rate".

El "slew – rate" de cada buffer de salida, por omisión, pequeño, minimiza la potencia de los transitorios en el bus cuando cambian señales no críticas. Para señales críticas, se liga una propiedad o atributo "FAST" al buffer de salida o flip – flop.

Los dispositivos de la Serie XC4000 tienen una propiedad llamada "Soft Start – up" (Puesta en Marcha Suave), diseñada para reducir las variaciones de tierra cuando todos las salidas están cambiando simultáneamente al final de la configuración. Cuando el proceso de configuración finaliza y el dispositivo se pone en marcha, la primera activación de las salidas es automáticamente "slew – rate" limitado. Inmediatamente después de la activación inicial del I / O, el "slew – rate" de las salidas individuales es determinado por la opción de configuración individual para cada IOB.

Tercer Estado Global.

Una línea de Tercer Estado Global separada (no mostrada en la Figura 3.17 o en la Figura 3.18) forza a todas las salidas del FPGA a un estado de alta impedancia, a menos que el "Boundary – Scan" se habilite y se ejecute una instrucción EXTEST. Esta red global (GTS) no compite con otros recursos de enrutamiento; esta usa una red de distribución dedicada.

GTS puede manejarse desde cualquier pin programable por el usuario como una entrada de Tercer Estado Global. Para usar esta red global, ponga un "pad" de entrada y un buffer de entrada en el esquemático o en el código HDL, manejando el pin GTS del símbolo STARTUP. Una posición específica del pin se puede asignar a esta entrada usando un atributo o propiedad LOC, igual que como con cualquier otro "pad" programable por el usuario. Un inversor puede insertarse opcionalmente después del buffer de entrada para invertir el sentido de la señal de Tercer Estado Global. Usar GTS es similar a GSR. Vea la Figura 3.3 para más detalles.

Alternativamente, GTS puede manejarse desde cualquier nodo interno.

Multiplexor de Salida / Generador de Función de 2 Entradas (XC4000EX únicamente).

Como se muestra en la Figura 3.18, la ruta de salida en los IOB XC4000EX contiene un multiplexor adicional (MUX de salida) no disponible en los IOB XC4000E. El multiplexor puede también configurarse como un generador de función de 2 entradas, implementando una compuerta de paso, una compuerta AND, una compuerta OR, o una compuerta XOR, con 0, 1, o 2 entradas invertidas. La lógica usada para implementar estas funciones se muestra en el área superior gris de la Figura 3.18.

Cuando se configura como un multiplexor, esta característica permite que dos señales de salida compartan al mismo tiempo el "pad" de salida; duplicando efectivamente el número de salidas del dispositivo sin requerir un encapsulado más grande, y más caro.

Cuando el MUX se configura como un generador de función de 2 entradas, la lógica puede implementarse dentro del propio IOB. Combinado ya sea con un "FastCLK" o con un buffer "Global Early", este arreglo permite cambios de muy alta velocidad de una sola señal. Por ejemplo, un decodificador amplio puede implementarse en los CLB's, y sus cambios a la salida con un "Read Strobe" o "Write Strobe" manejado por un buffer "FastCLK", como se muestra en la Figura 3.21. El retardo crítico de la ruta pin – a – pin de este circuito es menor a 6 nanosegundos.

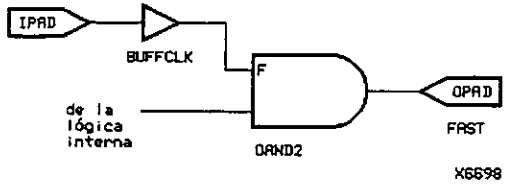


Figura 3.21: Ruta Pin – a – Pin Rápida en los XC4000E.

Como se muestra en la Figura 3.18, los pines de entrada del IOB OUT, Output Clock, y Clock Enable tienen diferentes retardos y flexibilidades con respecto a la polaridad. Adicionalmente, los suministros Output Clock están más limitados que en las otras entradas. Por consiguiente, el software de Xilinx no mueve la lógica a los generadores de función del IOB a menos que explícitamente se direccionen para hacerlo.

El usuario puede especificar que el generador de función del IOB sea usado, colocando símbolos de biblioteca especiales que empiezan con la letra "O". Por ejemplo, una compuerta AND de 2 entradas en el generador de función del IOB es llamada OAND2. Se debe usar el símbolo del pin de entrada etiquetada como "F" para la señal en una ruta crítica. Esta señal se coloca en el pin OK, la entrada del IOB con retardo más corto al generador de función. Se muestran dos ejemplos en la Figura 3.22.

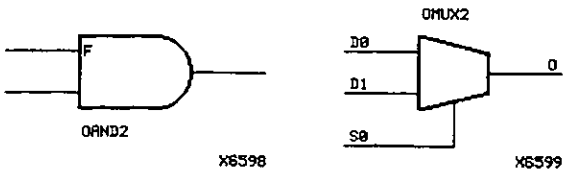


Figura 3.22: Salida AND y Símbolos MUX en el IOB XC4000EX.

c) Otras Opciones del IOB.

Existe un gran número de diferentes opciones programables en el IOB de la Serie XC4000.

Resistores "Pull – Down" y "Pull – Up".

Los resistores "pull – down" y "pull – up" programables son útiles para ligar los pines no utilizados, a Vcc o Ground para minimizar el consumo de potencia y reducir la sensibilidad del ruido. El resistor "pull – up" configurable es un transistor de canal p conectado a Vcc. El resistor "pull – down" configurable es un transistor de canal n conectado a Ground.

El valor de estos resistores es de 50 KΩ a 100 KΩ. Este valor alto, hace imposible una AND alambrada con los resistores "pull – up".

Los resistores "pull – up" para la mayoría de los IOB's programables por el usuario son activos durante el proceso de configuración. Vea la Tabla 3.22 para una lista de pines con "pull – up's" activos antes y durante la configuración.

Después de la configuración, los niveles de voltaje de los "pad's" no utilizados, garantizados o no garantizados, deben ser niveles lógicos válidos, para reducir la sensibilidad al ruido y evitar corrientes excesivas. Por lo tanto, por omisión, los "pad's" no utilizados se configuran con el resistor "pull – up" interno activo. Alternativamente, ellos pueden configurarse individualmente con el resistor "pull – down", o como una salida manejada, o para manejarse por una fuente externa. Para activar el "pull – up" interno, ligar el componente de biblioteca PULLUP a la red cercana al "pad". Para activar el "pull – down" interno, ligar el componente de biblioteca PULLDOWN a la red cercana al "pad".

Relojes Independientes.

Las señales separadas de reloj son proporcionadas para los flip – flops de entrada y salida. El reloj puede invertirse independientemente para cada flip – flop dentro del IOB, generando ya sea flip – flops disparados por flancos ascendentes o flancos descendentes. Las entradas de reloj para cada IOB son independientes, excepto en los XC4000EX, donde el latch de Captura Rápida comparte una entrada del IOB con el pin de reloj de salida.

Reloj "Early" (Anticipado) para los IOB's (XC4000EX únicamente).

Los relojes "Early" (Anticipados) especiales están disponibles para los IOB's. Estos relojes son suministrados por las mismas fuentes como los buffers "Low – Skew" Globales, pero pasan por buffers separados. Ellos tienen menos cargas y por lo tanto menos retardos. El reloj anticipado puede manejar ya sea el reloj de salida del IOB, el reloj de entrada del IOB, o ambos. El reloj anticipado permite la captura rápida de los datos de entrada, y el reloj de salida rápido se emplea para datos de salida. Los buffers "Global Early" que manejan estos relojes se describen en " Redes Globales y Buffers (XC4000EX únicamente)".

Reloj Rápido para los IOB's (XC4000EX únicamente).

Los relojes muy rápidos manejados por buffers "FastCLK" también están disponibles para los IOB. Estos relojes son suministrados por "pad's" semidedicados.- los "pad's" pueden usarse como I / O generales si no son usados para manejar a los buffers "FastCLK". Hay dos buffers "FastCLK" en el extremo izquierdo, y dos en el extremo derecho del dispositivo. Ellos proporcionan el método más rápido de llegar a los pines de reloj del IOB. El buffer "FastCLK" puede manejar ya sea el reloj de salida del IOB, o el reloj de entrada del IOB, o ambos. Estos buffers permiten los tiempos de establecimiento más rápidos y de reloj a salida, posibles. Los buffers "FastCLK" se describen en "Redes Globales y Buffers (XC4000EX únicamente)".

Set / Reset Global.

Como con los registros del CLB, la señal Set / Reset (GSR) puede usarse para establecer (set) o para limpiar (clear) los registros de entrada y salida, dependiendo del valor del atributo o propiedad INIT. Los dos flip – flops pueden configurarse individualmente para establecerse o limpiarse en el reinicio y después de la configuración. Otra cuestión es que en la red GSR global, ningún usuario controla la señal de establecer / limpiar que está disponible en los flip – flops I / O. La opción de establecer o limpiar se aplica a ambos estados iniciales del flip – flop y a la respuesta al pulso Set / Reset Global. Vea "Set / Reset Global" para una descripción de cómo usar GSR.

Soporte JTAG.

La lógica adjunta ligada a los IOB's contiene estructuras de prueba compatible con la norma IEEE 1149.1 para el rastreo bondadoso de prueba, permitiendo pruebas sencillas a nivel de placa y de chip. Se proporciona más información en "Boundary - Scan".

3.2.4 Buffers de Tercer Estado.

Un par de buffers de tercer estado están asociados con cada CLB en el arreglo (vea la Figura 3.28). Estos buffers de tercer estado pueden usarse para manejar señales sobre las líneas largas horizontales más cercanas arriba y abajo del CLB. Ellos pueden por lo tanto usarse para implementar buses bidireccionales o multiplexados sobre las líneas largas horizontales, ahorrando recursos lógicos. Los resistores "pull – up" programables, unidos a estas líneas largas ayudan a implementar amplias funciones AND alambradas.

La habilitación del buffer es en tercer estado activo en alto (es decir, una habilitación activa en bajo), como se muestra en la Tabla 3.13.

Entrada.	T.	Salida.
X.	1	Z
Entrada.	0	Entrada

Tabla 3.13: Funcionalidad del Buffer de Tercer Estado.

Otro buffer de tercer estado con acceso similar se ubica cerca de cada bloque I / O a lo largo de los extremos derecho e izquierdo del arreglo (vea la Figura 3.34).

Las líneas largas horizontales manejadas por los buffers de tercer estado tienen un pequeño protector en cada extremo. Este circuito previene niveles flotantes indefinidos. Sin embargo, este es sustituido por cualquier "driver" (manejador), incluso por un resistor "pull – up"

Pueden usarse las líneas largas especiales que corren a lo largo del perímetro del arreglo para usar señales alambradas AND que provienen de los IOB's cercanos o de las líneas largas internas. Estas líneas largas forman al decodificador amplio colocado en los extremos del dispositivo discutido en "Decodificadores Amplios".

a) Modos del Buffer de Tercer Estado.

Los buffers de tercer estado pueden configurarse en tres modos:

- Buffer de tercer estado estándar.
- AND – Alambrada con entrada en el pin I.
- OR – AND Alambrada.

Buffer de Tercer Estado Estándar.

Los tres pines se usan. Se debe colocar el elemento de biblioteca BUFT. Conectar la entrada al pin I y la salida al pin O. El pin T está activo en alto en tercer estado (es decir, una habilitación activa en bajo). Uniendo el pin T a Ground para implementar un buffer estándar.

AND Alambrada con Entrada en el Pin I.

El buffer puede usarse como una AND Alambrada. Se debe usar el símbolo de biblioteca WAND1, que es esencialmente un buffer de colector abierto. WAND4, WAND8, y WAND16 también están disponibles. Vea la *Guía de Biblioteca* XACT para información adicional.

El pin T se une internamente al pin I. Se debe conectar la entrada al pin I y la salida al pin O. Conectar las salidas de todos los WAND1's juntas y ligar un símbolo PULLUP.

OR – AND Alambrada.

El buffer puede configurarse como una OR – AND Alambrada. Un nivel alto en cualquier entrada apaga la salida. Se debe usar el símbolo de biblioteca WOR2AND, el cual es esencialmente una compuerta OR de 2 entradas de colector abierto. Los dos pines de la entrada son funcionalmente equivalentes. Se deben ligar las dos entradas a los pines I0 e I1 y una la salida al pin O. Unir las salidas de todos los WOR2AND's juntos y ligar un símbolo PULLUP.

b) Ejemplos de Buffer de Tercer Estado.

La Figura 3.23 muestra como usar los buffers de tercer estado para implementar una función AND alambrada. Cuando todas las entradas de los bufferes están en alto, el resistor o resistores "pull – up" proporcionan una salida alta.

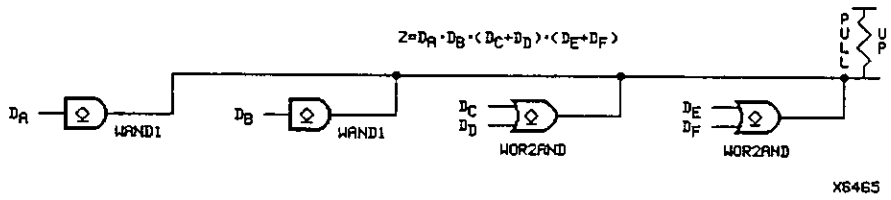


Figura 3.23: Buffers de Colector Abierto Implementados con Función AND Alambrada.

La Figura 3.24 muestra como usar los buffers de tercer estado para implementar un multiplexor. La selección se realiza mediante la señal de tercer estado del buffer.

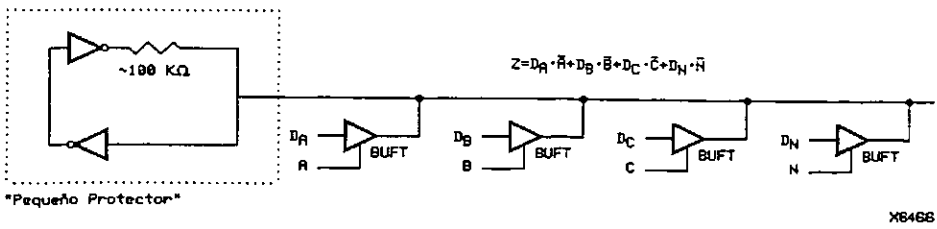


Figura 3.24: Multiplexor Implementado con Buffers de Tercer Estado.

Ponga atención especial a la polaridad del pin T cuando use estos buffers en un diseño. El Tercer estado activo en alto (T) es idéntico a una habilitación de salida activa en bajo, como se muestra en la Tabla 3.13.

3.2.5 Decodificadores Amplios.

La circuitería del decodificador especializada desarrolla funciones de decodificación amplias. Cuando la dirección o el campo de los datos es más amplio que las entradas del generador de función, los FPGA's necesitan una decodificación multinivel y son por consiguiente más lentos que los PAL's. Los CLB's de la Serie XC4000 tienen nueve entradas. Cualquier decodificador de hasta nueve entradas es, por consiguiente, rápido y compacto. Sin embargo, existe también la necesidad de decodificadores mucho más amplios, especialmente para la decodificación de dirección en sistemas de microprocesador grandes.

Un FPGA de la Serie XC4000 tiene cuatro decodificadores programables localizados en cada extremo del dispositivo. Las entradas a cada decodificador son cualquiera de las señales I1 del IOB en ese extremo, más una interconexión local por renglón o columna del CLB. Cada renglón o columna de los CLB's proporciona hasta tres variables o sus complementos, como se muestra en la Figura 3.25. Cada decodificador genera una salida alta (resistor "pull – up") cuando la condición AND de las entradas seleccionadas, o de sus complementos, es verdadera. Esto es análogo a un término de producto en los dispositivos PAL típicos.

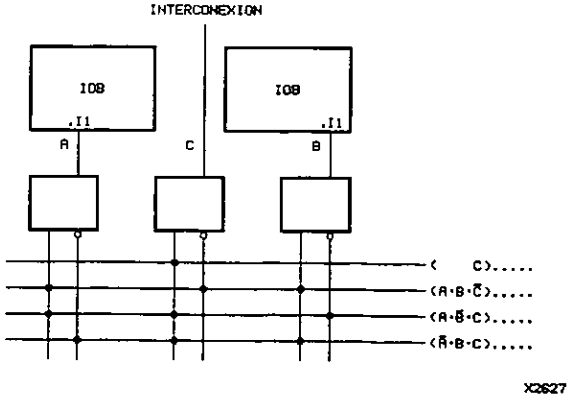


Figura 3.25: Ejemplo del Borde de Decodificación en la Serie XC4000.

Cada una de éstas compuertas AND alambradas es capaz de aceptar hasta 42 entradas en los XC4005E y 72 en los XC4013E. Existen hasta 96 entradas para cada decodificador en los XC4028EX y 132 en los XC4052EX. Los decodificadores también pueden dividirse en dos cuando un número más grande de decodificadores más estrechos se requieran, para un máximo de 32 decodificadores por dispositivo.

Las salidas del decodificador pueden manejar entradas de CLB, para que estos puedan combinarse con otra lógica para formar un PAL – como una estructura AND / OR. Las salidas del decodificador también pueden enrutarse directamente a las salidas del chip. Para tener una velocidad más rápida, la salida debe estar en el mismo extremo del chip que el decodificador. Los PAL's muy grandes pueden emularse mediante ORing en las salidas del decodificador en un CLB. Estas características de decodificación cubren lo que ha sido considerado desde hace tiempo una debilidad de los FPGA's más viejos. Los usuarios frecuentemente acudían a PAL's externos para funciones simples pero de decodificación rápida. Ahora, los decodificadores dedicados en los dispositivos de la Serie XC4000 pueden implementar estas funciones rápida y eficientemente.

Para usar los decodificadores amplios, se debe colocar uno o más de los símbolos de biblioteca WAND (WAND1, WAND4, WAND8, WAND16). Se debe ligar un atributo o propiedad DECODE a cada símbolo WAND. Se deben unir las salidas y ligar un símbolo PULLUP. Las propiedades o atributos locales tal como L (extremo izquierdo) o TR (mitad derecha del extremo superior) deberían usarse también para asegurar la colocación correcta de las entradas del decodificador.

3.2.6 Oscilador en el Chip.

Los dispositivos de la Serie XC4000 incluyen un oscilador interno. Este oscilador se usa para temporizar el tiempo de desactivación de la aplicación de potencia, para la limpieza de la memoria de configuración, y como fuente de CCLK en los modos de configuración Maestra. El oscilador corre a una frecuencia nominal de 8 MHz. que varía con el proceso, Vcc, y la temperatura. La frecuencia de salida se encuentra entre 4 y 10 MHz. (El oscilador opera más lentamente a voltajes bajos. La frecuencia de salida puede reducirse a lo mucho como por un 10% para los dispositivos de bajo voltaje).

La salida del oscilador está opcionalmente disponible después de la configuración. Dos de las cuatro derivaciones resincronizadas de un divisor incorporado también están disponibles. Estas derivaciones son el cuarto, noveno, decimocuarto y el decimonoveno bits del divisor. Por lo tanto,

si la salida de oscilador primario corre al valor nominal de 8 MHz., el usuario tiene acceso a un reloj de 8 MHz., más dos cualesquiera de 500 kHz., 16kHz, 490Hz. y 15Hz. (hasta un 10% menos para dispositivos de bajo voltaje). Estas frecuencias pueden variar por mucho como por - 50% o +25%.

Estas señales pueden ser accesadas al poner el elemento de biblioteca OSC4 en un esquemático o en código HDL (vea la Figura 3.26).

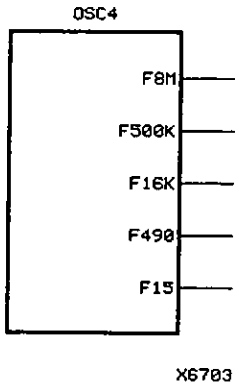


Figura 3.26: Símbolo del Oscilador de la Serie XC4000.

El oscilador es automáticamente deshabilitado después de la configuración si el símbolo OSC4 no se utiliza en el diseño.

3.3 INTERCONEXIONES PROGRAMABLES.

Todas las conexiones internas se componen de segmentos metálicos con puntos conmutables programables y matrices conmutables para implementar el enrutamiento deseado. Una estructurada matriz jerárquica de recursos de enrutamiento es proporcionada para lograr un enrutamiento eficientemente automatizado.

Los XC4000E y XC4000EX comparten una estructura básica de interconexión. Los dispositivos XC4000EX, sin embargo, tienen enrutamiento adicional no disponible en los XC4000E. Los recursos de enrutamiento extra permiten la utilización en los dispositivos de alta capacidad. Todos

los recursos de enrutamiento específicos XC4000EX se identifican claramente a lo largo de esta sección. Cualquier recurso no identificado como específico de XC4000EX está presente en todos los dispositivos de la Serie XC4000.

Esta sección describe los variados recursos de enrutamiento disponibles en los dispositivos de la Serie XC4000. El software de implementación asigna automáticamente los recursos apropiados con base en la densidad y las necesidades de temporización en el diseño.

3.3.1 Visión General de las Interconexiones.

Existen los siguientes tipos de interconexiones.

- El enrutamiento del CLB está asociado con cada renglón y columna del arreglo del CLB.
- El enrutamiento del IOB forma un anillo (llamado "VersaRing") alrededor del exterior del arreglo del CLB. Este conecta al I / O con los bloques lógicos internos.
- El enrutamiento global consiste de redes primarias dedicadas diseñadas para distribuir relojes a lo largo del dispositivo con desvíos y retardos mínimos. El enrutamiento global también puede usarse para otras señales de salida de carga alta (Fan – out alto).

Los cinco tipos de interconexión son distinguidos por la longitud relativa de sus segmentos: líneas de longitud simples, líneas de longitud doble, líneas cuádruples y octales (XC4000EX únicamente), y líneas largas. En los XC4000EX, las conexiones directas permiten el flujo rápido de datos entre CLB's adyacentes, y entre los IOB's y los CLB's.

El enrutamiento extra es incluido en un "pad" de anillo del IOB. Los XC4000EX también incluyen un anillo de líneas de interconexión octales cerca de los IOB's para mejorar un intercambio del pin y el enrutamiento a los pines de bloqueo.

Los dispositivos XC4000E incluyen dos tipos de buffers globales, mientras que los dispositivos XC4000EX tienen tres tipos diferentes. Estos buffers globales tienen propiedades diferentes, y

están destinados a propósitos diferentes. Ellos se describen en forma detallada más adelante en esta misma sección.

3.3.2 Enrutamiento de las Conexiones del CLB.

Un diagrama descriptivo de los recursos de enrutamiento asociados con un CLB se muestra en la Figura 3.27. Las flechas sombreadas representan el enrutamiento presente sólo en los dispositivos XC4000EX.

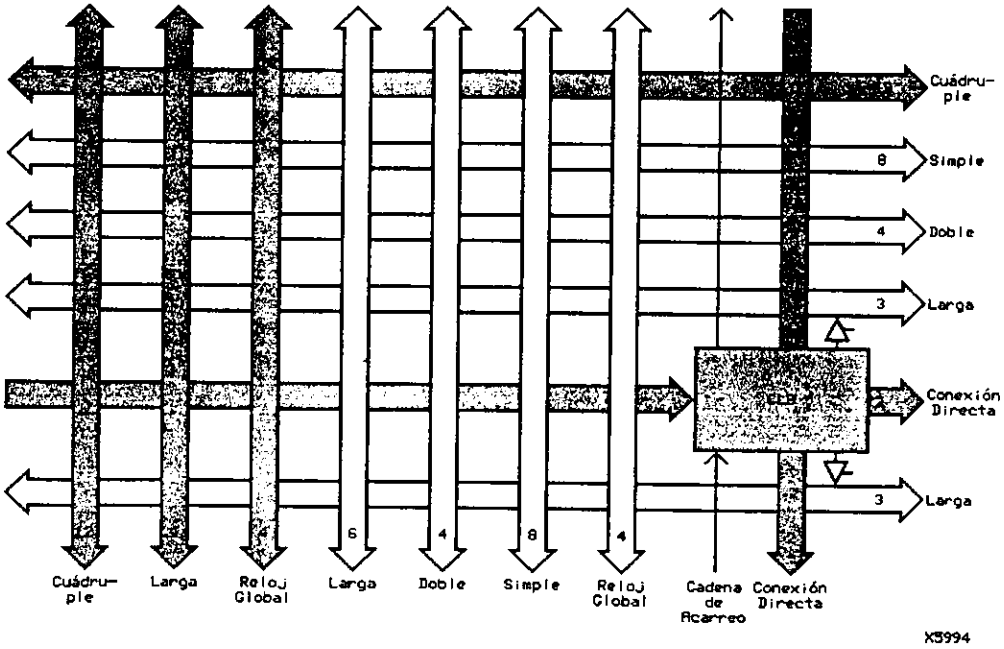


Figura 3.27: Diagrama de Enrutamiento Descriptivo del CLB de la Serie XC4000 (las flechas sombreadas indican a los XC4000EX únicamente).

La Tabla 3.14 muestra cuantos enrutamientos de cada tipo están disponibles en los arreglos de CLB XC4000E y XC4000EX. Claramente, los diseños muy grandes, o los diseños con un gran arreglo de interconexiones, se enrutan más fácilmente en los XC4000EX. Para diseños más pequeños los XC4000E, que típicamente requieren de un menor número de interconexiones, no necesitan de enrutamiento adicional.

	XC4000E.		XC4000EX.	
	Vertical.	Horizontal.	Vertical.	Horizontal.
Simples.	8	8	8	8
Dobles.	4	4	4	4
Cuádruples.	0	0	12	12
Líneas Largas.	6	6	10	6
Conexiones Directas.	0	0	2	2
Globales.	4	0	8	0
Lógica de Acarreo.	2	0	1	0
Total.	24	18	45	32

Tabla 3.14: Enrutamiento para el CLB en los Dispositivos de la Serie XC4000.

La Figura 3.28 es un diagrama detallado de ambos CLB's en el XC4000E y en el XC4000EX, con enrutamiento asociado. El cuadro sombreado es la matriz de interconexión programable, presente en ambos dispositivos XC4000E y XC4000EX. La L formada por el área sombreada está presente únicamente en los dispositivos XC4000EX. Como se muestra en la figura, el bloque XC4000EX es esencialmente un bloque XC4000E con enrutamiento adicional.

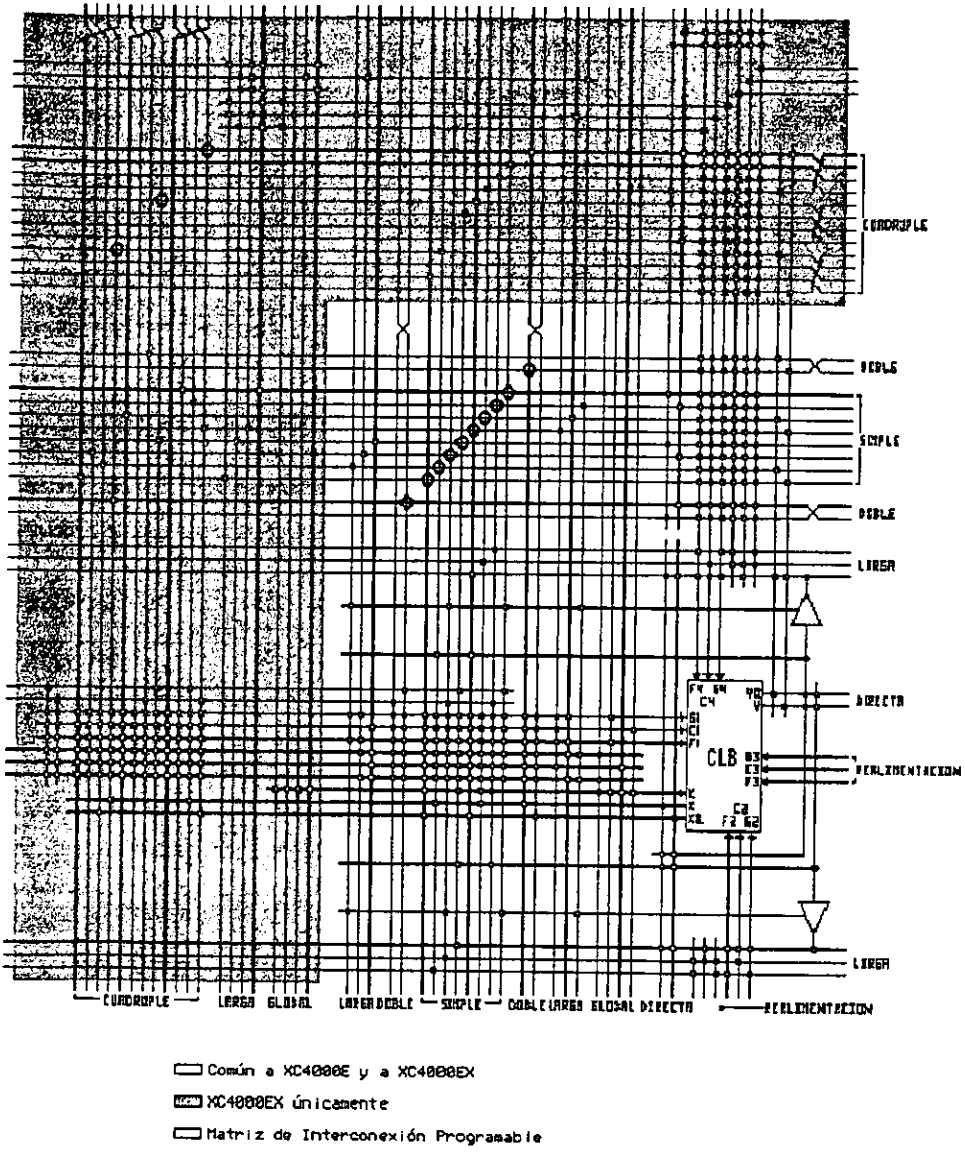


Figura 3.28: Detalle de las Interconexiones Programables Asociadas con el CLB de la Serie XC4000.

Las entradas y salidas del CLB están distribuidas a lo largo de los cuatro lados, proporcionando una máxima flexibilidad de enrutamiento. En general, la arquitectura entera es simétrica y regular. Satisface aceptablemente el establecimiento de los algoritmos de colocación y enrutamiento. Las entradas, salidas, y generadores de función pueden intercambiar posiciones libremente dentro de un CLB para evitar congestiones de enrutamiento durante la operación de colocación y enrutamiento.

a) Matrices de Interconexión Programable.

Las líneas de longitud simples y dobles, horizontales y verticales intersectan a una caja llamada una matriz de interconexión programable (PSM). Cada matriz de interconexión consiste en transistores de paso programables usados para establecer conexiones entre las líneas (vea la Figura 3.29).

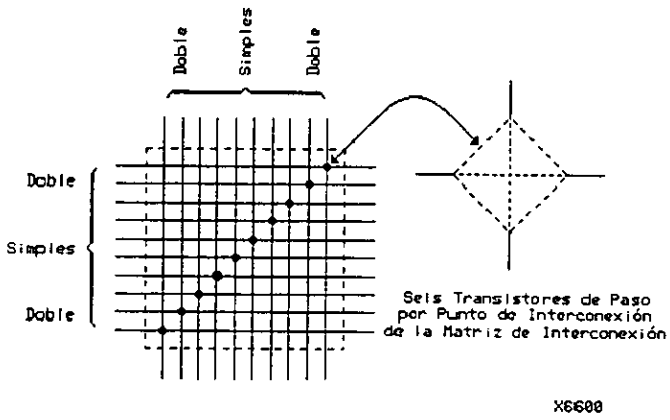


Figura 3.29: Matriz de Interconexión Programable (PSM).

Por ejemplo, una señal de longitud simple que entra en el lado derecho de la matriz de interconexión puede enrutarse a una línea de longitud simple en la parte superior, en la izquierda, o en los lados inferiores, o en cualquier combinación de estos, si se requieren múltiples ramas. De forma semejante, una señal de longitud doble puede enrutarse a una línea de longitud doble en cualquiera o todos de los otros tres extremos de la matriz de interconexión programable.

b) Líneas de Longitud Simple.

Las líneas de longitud simple proporcionan la más grande flexibilidad de interconexión y ofrecen un rápido enrutamiento entre los bloques adyacentes. Existen ocho líneas de longitud simple verticales y ocho horizontales asociadas con cada CLB. Estas líneas conectan a las matrices de interconexión que son localizadas en cada renglón y columna de los CLB's.

Las líneas de longitud simples están conectadas por medio de las matrices de interconexión programable, como se muestra en la Figura 3.30. La conectividad del enrutamiento se muestra en la Figura 3.28.

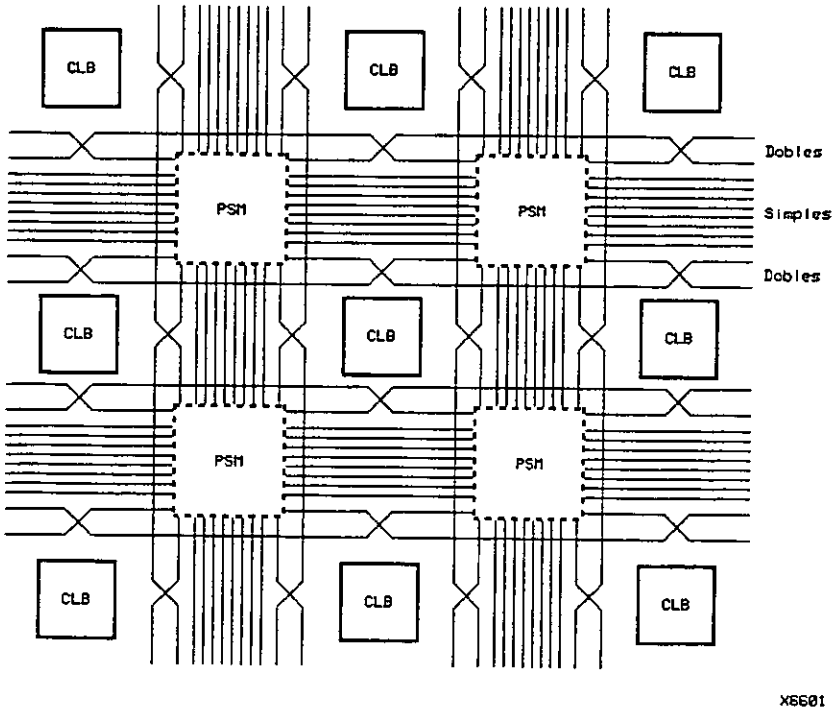


Figura 3.30: Líneas de Longitud Doble y Simple, con Matrices de Interconexión Programable (PSM's).

Las líneas de longitud simple incurren en un retardo siempre que ellas pasen por una matriz de interconexión. Por consiguiente, estas líneas no son convenientes para enrutamiento de señales

para largas distancias. Estas líneas normalmente se usan para conducir señales dentro de una área localizada y para proporcionar bifurcación a las redes con carga de salida mayor que uno.

c) Líneas de Longitud Doble.

Las líneas de longitud doble consisten en una rejilla de segmentos metálicos, con dos veces la longitud de las líneas de longitud simple: ellas pasan dos CLB's antes de entrar en una matriz de interconexión. Las líneas de longitud doble se agrupan en pares con las matrices de interconexión escalonadas, de manera que cada línea pase por una matriz de interconexión por uno u otro renglón o columna de los CLB's (vea la Figura 3.30).

Existen cuatro líneas de longitud doble verticales y cuatro horizontales asociadas con cada CLB. Estas líneas proporcionan enrutamiento de señal más rápido sobre distancias intermedias, mientras la flexibilidad del enrutamiento se mantiene. Las líneas de longitud doble se conectan por medio de las matrices de interconexión programable. La conectividad de enrutamiento se muestra en la Figura 3.28.

d) Líneas Cuádruples (XC4000EX únicamente).

Los dispositivos XC4000EX también incluyen doce líneas cuádruples verticales y doce horizontales por renglón y columna del CLB. Las líneas cuádruples son cuatro veces la longitud de las líneas de longitud simple. Ellas se interconectan por medio de matrices de interconexión tipo buffer (mostrado como diamantes en la Figura 3.28). Las líneas cuádruples pasan cuatro CLB's antes de entrar en una matriz de interconexión tipo buffer. Ellas se agrupan en cuatro, mediante las matrices de interconexión tipo buffer escalonadas, para que cada línea pase por una matriz de interconexión tipo buffer ubicada cada cuarto CLB en ese renglón o columna (vea la Figura 3.31).

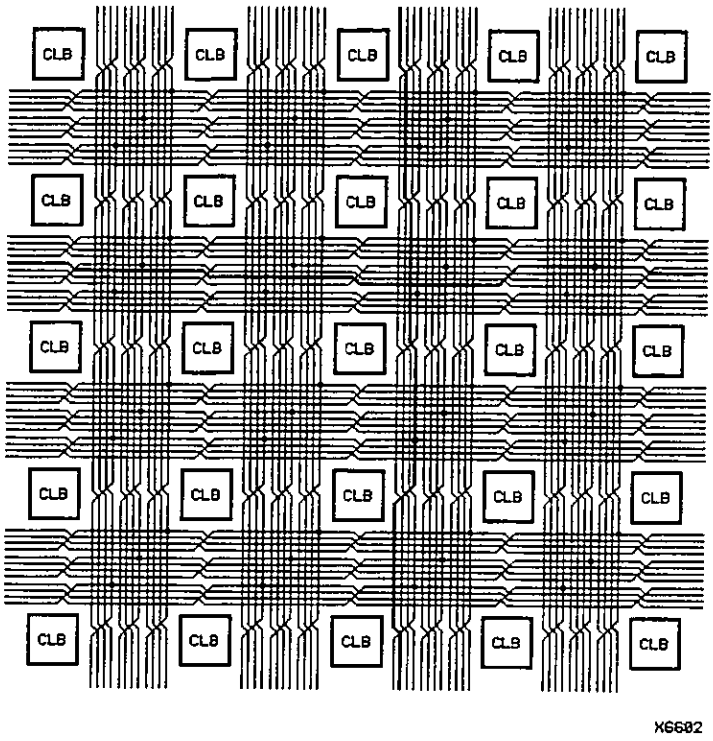


Figura 3.31: Líneas Cuádruples (XC4000EX únicamente).

Las matrices de interconexión tipo buffer tienen cuatro pines, uno en cada extremo. Todos los pines son bidireccionales. Cualquier pin puede manejar cualquiera o todos los otros pines.

Cada matriz de interconexión tipo buffer contiene un buffer y seis transistores de paso. Esta se asemeja a la matriz de interconexión programable mostrada en la Figura 3.29, con la adición de un buffer programable. Puede haber hasta dos entradas independientes y hasta dos salidas independientes. Unicamente una de las entradas independientes puede pasar por el buffer.

El software de colocación y enrutamiento automáticamente usa los requerimientos de temporización del diseño para determinar si una señal de línea cuádruple debe pasar o no por el buffer. Una señal fuertemente cargada típicamente se pasa por el buffer, mientras una ligeramente cargada no. Un panorama es alternar buffers y transistores de paso. Esto permite que ambas

líneas cuádruples verticales y horizontales puedan pasar por el buffer alternando en las matrices de interconexión tipo buffer.

Debido a las matrices de interconexión tipo buffer, las líneas cuádruples son muy rápidas. Ellas proporcionan el método más rápido disponible de enrutamiento de señales fuertemente cargadas para distancias largas a través del dispositivo.

e) Líneas Largas.

Las líneas largas forman una rejilla de segmentos metálicos interconectados que corren a lo largo o ancho de todo el arreglo. Las líneas largas están destinadas para factores de carga altos, redes de señal de tiempo crítico, o redes que se distribuyen sobre distancias largas. En los dispositivos XC4000EX, se prefieren las líneas cuádruples para las redes críticas, porque las matrices de interconexión tipo buffer las hacen más rápidas para redes de factor de carga alto.

Dos líneas largas horizontales por cada CLB pueden ser manejadas por "driver's" (manejadores) de colector abierto o tercer estado (TBUF's). Ellos pueden implementar por lo tanto, buses unidireccionales o bidireccionales, multiplexores amplios, o funciones AND alambradas (vea "Buffers de Tercer Estado" para más detalles).

Cada línea larga horizontal manejada por los TBUF's tiene ya sea dos (XC4000E) u ocho (XC4000EX) resistores "pull – up". Para activar estos resistores, se debe ligar un símbolo PULLUP a la red de línea larga. El software automáticamente activa el número apropiado de "pull – up's". Existe también una pequeña protección en cada extremo de estas dos líneas largas horizontales. Este circuito previene niveles flotantes indefinidos. Sin embargo, es anulado por cualquier manejador, incluso por un resistor "pull – up".

Cada línea larga XC4000E tiene una interconexión programable dividida en su centro, como se hace en cada línea larga XC4000EX manejada por TBUF's. Esta interconexión puede separar a la línea en dos canales de enrutamiento independientes, corriendo cada mitad a lo ancho o a lo largo del arreglo.

Cada línea larga XC4000EX no manejada por los TBUF's tiene una interconexión programable tipo buffer dividida en los puntos 1 / 4, 1 / 2, y 3 / 4 del arreglo. Debido a este buffer, el desempeño de las líneas largas XC4000EX no se deteriora con los arreglos de tamaños más grandes. Suponiendo que la línea larga fuera dividida, las resultantes líneas largas serían independientes.

La conectividad del enrutamiento de las líneas largas se muestra en la Figura 3.28.

f) Interconexión Directa (XC4000EX únicamente).

Los XC4000EX ofrecen dos conexiones directas, eficaces y rápidas entre los CLB's adyacentes. Estas redes facilitan el flujo de datos desde el lado izquierdo hasta el derecho del dispositivo, o desde la parte superior a la inferior, como se muestra en la Figura 3.32. El enrutamiento de señales en las interconexiones directas muestra mínimos retardos de propagación en la interconexión y no usa recursos de enrutamiento generales.

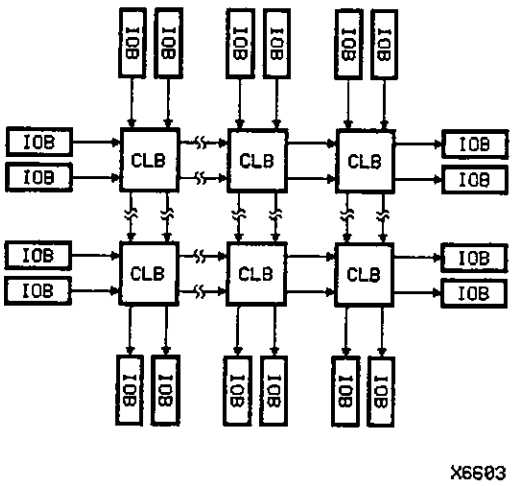


Figura 3.32: Interconexión Directa XC4000EX.

La interconexión directa también está presente entre los CLB's y los IOB's adyacentes. Cada IOB en los extremos izquierdos y superiores del dispositivo tiene una trayectoria directa al CLB más cercano. Cada CLB sobre los extremos derechos y los inferiores del arreglo tiene una trayectoria directa a los dos IOB's más cercanos, ya que existen dos IOB's por cada renglón o columna de los CLB's.

El software de asignación y enrutamiento usa interconexión directa cuando es posible, para maximizar los recursos de enrutamiento y minimizar los retardos de interconexión.

3.3.3 Enrutamiento I / O.

Los dispositivos de la Serie XC4000 tienen enrutamiento adicional alrededor del anillo del IOB. Este enrutamiento se llama "VersaRing". El "VersaRing" facilita el intercambio de pin y el rediseño sin afectar el diseño de la tarjeta. Incluye a ocho líneas de longitud doble extendidas sobre dos CLB's (cuatro IOB's), y cuatro líneas largas. Se proporcionan líneas globales y líneas de Decodificador Amplio. Los dispositivos XC4000EX también incluyen ocho líneas octales.

Un diagrama de alto nivel del "VersaRing" se muestra en la Figura 3.33. Las flechas sombreadas representan el enrutamiento presente únicamente en los dispositivos XC4000EX.

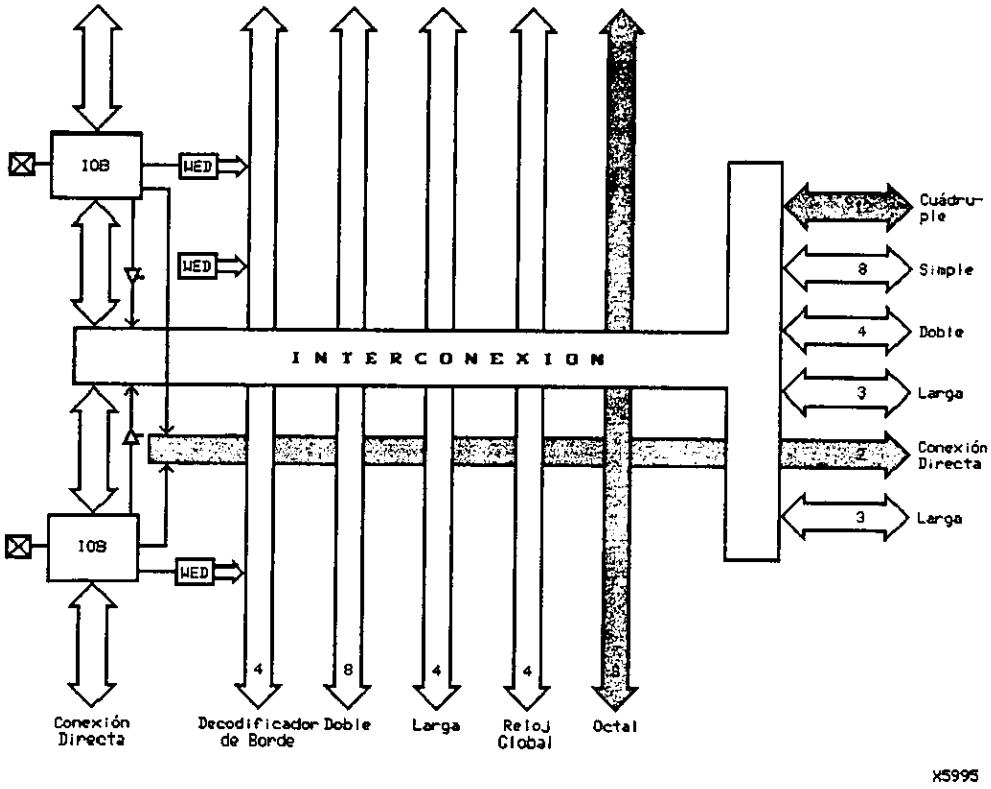


Figura 3.33: Diagrama de Enrutamiento Descriptivo del "VersaRing" de la Serie XC4000 (Borde Izquierdo).

WED = Decodificador Amplio, IOB = Bloque I / O (las flechas sombreadas se indican para los XC4000EX únicamente).

La Figura 3.34 es un diagrama detallado de los "VersaRing" XC4000E y XC4000EX. El área mostrada incluye dos IOB's. Existen dos IOB's por cada renglón o columna del CLB, por lo tanto este diagrama corresponde al diagrama de enrutamiento del CLB mostrado en la Figura 3.28. Las áreas sombreadas representan el enrutamiento y las conexiones de enrutamiento presentes únicamente en los dispositivos XC4000EX.

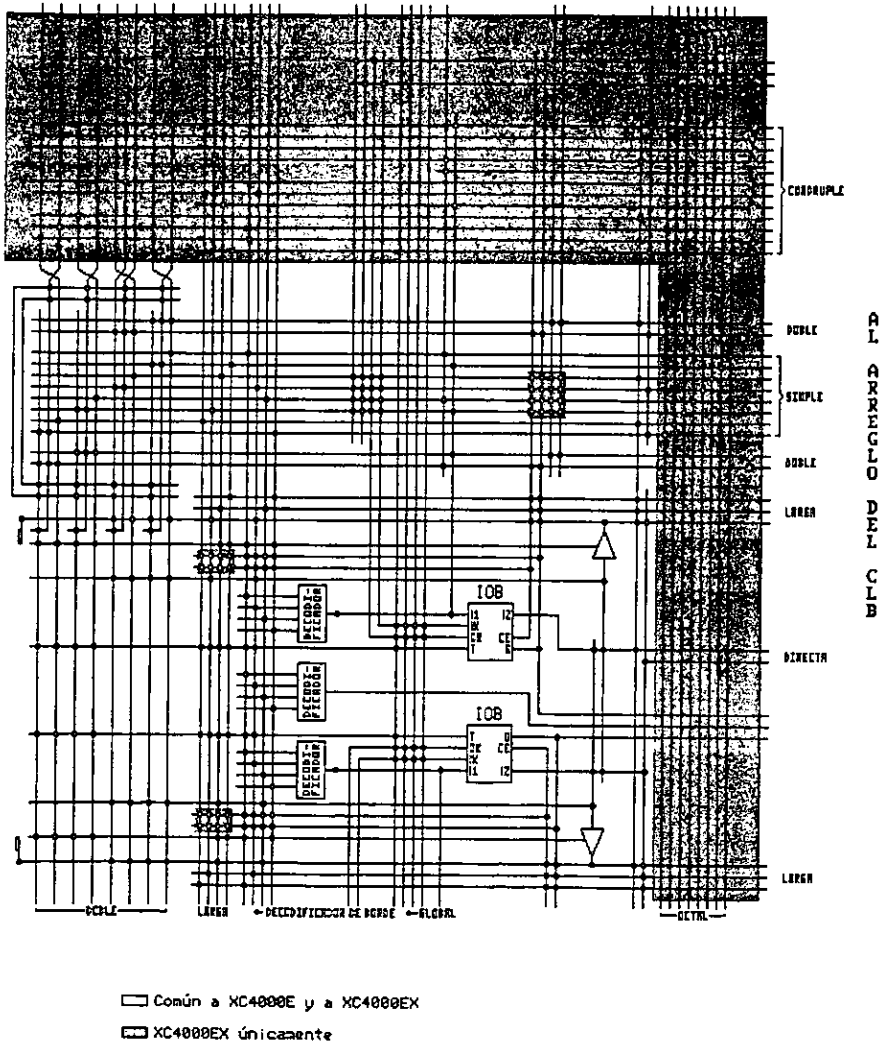


Figura 3.34: Detalle de las Interconexiones Programables Asociadas con el IOB de la Serie XC4000 (Extremo Izquierdo).

a) Enrutamiento I / O Octal (XC4000EX únicamente).

Entre el arreglo del CLB XC4000EX y el "pad" de anillo, se proporcionan ocho pistas de interconexión para versatilidad en la asignación de pin y la flexibilidad en la fijación del pin de salida (vea la Figura 3.35).

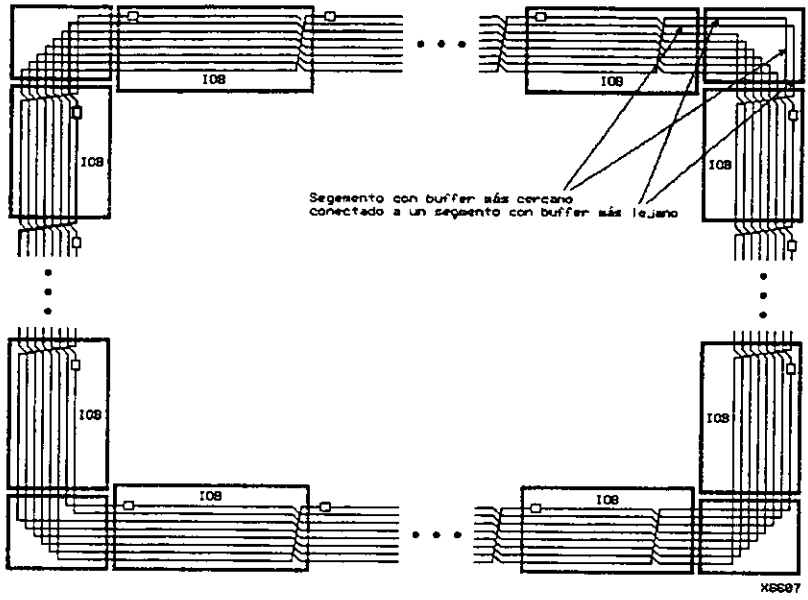


Figura 3.35: Enrutamiento I / O Octal de los XC4000EX.

Estas pistas de enrutamiento se llaman octales, porque ellas pueden partirse cada ocho CLB's (dieciséis IOB's) mediante un buffer programable que también funciona como una interconexión divisora. Los buffers están escalonados, para que cada línea pase por un buffer por cada octavo CLB localizado alrededor del extremo del dispositivo.

Las líneas octales doblan alrededor de las esquinas del dispositivo. Las líneas cruzan a las esquinas de tal manera que el segmento que pasó recientemente por el buffer antes de girar tiene la distancia más lejana para viajar antes del próximo buffer, como se muestra en la Figura 3.35.

La interfaz entre las entradas y salidas del IOB con las líneas octales se une por medio de las líneas de interconexión de longitud simple. También se usan líneas de longitud simple para la

comunicación entre las octales y las líneas de longitud dobles, cuádruples, y las líneas largas dentro del arreglo del CLB.

La segmentación en las líneas octales que pasan por el buffer fue establecida para hacer una óptima distribución de señales sobre distancias largas alrededor del dispositivo.

3.3.4 Redes Globales y Buffers.

Los XC4000E y los XC4000EX tienen redes globales dedicadas. Estas redes se diseñan para distribuir relojes y otras señales de control con factor de carga alto a lo largo de los dispositivos con un mínimo desvío. Los buffers globales se describen en detalle en las secciones siguientes. Las descripciones del texto y los diagramas se resumen en la Tabla 3.15. La Tabla muestra cuales pines de reloj del CLB e IOB pueden ser suministrados por cuales buffers globales.

	XC4000E.		XC4000EX.				Inter-
	BUFGP.	BUFGS.	BUFGLS.	L & R BUFGE.	T & B BUFGE.	BUFFC K.	conexión Local.
Todos los CLB's en el Cuadrante.	✔	✔	✔	✔	✔		✔
Todos los CLB's en el Dispositivo.	✔	✔	✔				✔
IOB's en la Mitad del Extremo Vertical Adyacente.	✔	✔	✔	✔	✔	✔	✔
IOB's en el Extremo Completo Vertical Adyacente.	✔	✔	✔	✔			✔
IOB's en la Mitad del Extremo Horizontal Adyacente (Directo).				✔			✔
IOB's en la Mitad del Extremo Horizontal Adyacente (a través de los CLB's globales).	✔	✔	✔	✔	✔		✔
IOB's en el Extremo Completo Horizontal Adyacente (a través de los CLB's globales).	✔	✔	✔				✔

L = Izquierdo, R = Derecho, T = Superior, B = Inferior.

Tabla 3.15: Acceso al Pin de Reloj.

En ambos dispositivos XC4000E y XC4000EX, la colocación de un símbolo de biblioteca llamado BUFG da por resultado en el software, la selección del buffer de reloj apropiado, basado en las necesidades de temporización del diseño. La información detallada en estas secciones se incluye únicamente para referencia.

a) Redes Globales y Buffers (XC4000E únicamente).

Cuatro líneas largas verticales en cada columna del CLB son manejadas exclusivamente por buffers globales especiales. Estas líneas largas se suman a las líneas largas verticales usadas para interconexión estándar. Las cuatro líneas globales pueden ser manejadas por cualquiera de los dos tipos de buffers globales. Los pines de reloj de cada CLB e IOB también pueden ser suministrados desde interconexiones locales.

Dos tipos diferentes de buffers de reloj están disponibles en los XC4000E:

- Buffers Globales Primarios (BUFGP).
- Buffers Globales Secundarios (BUFGS).

Los cuatro buffers Globales Primarios ofrecen un retardo más corto y un desvío despreciable. Los cuatro buffers Globales Secundarios tienen un retardo ligeramente más largo y ligeramente más desvío debido a la carga potencialmente más pesada, pero ofrecen mayor flexibilidad cuando se usan para manejar las entradas del CLB que no son de reloj.

Los buffers Globales Primarios deben ser manejados por "pad's" semidedicados. Los buffers Globales Secundarios pueden ser suministrados por cualquier "pad" semidedicado o por las redes internas.

Cada columna del CLB tiene cuatro líneas Globales verticales dedicadas. Cada una de estas líneas puede ser accesada a través de un buffer Global Primario particular, o por cualquiera de los buffers Globales Secundarios, como se muestra en la Figura 3.36. Cada esquina del dispositivo tiene un buffer Primario y un buffer Secundario.

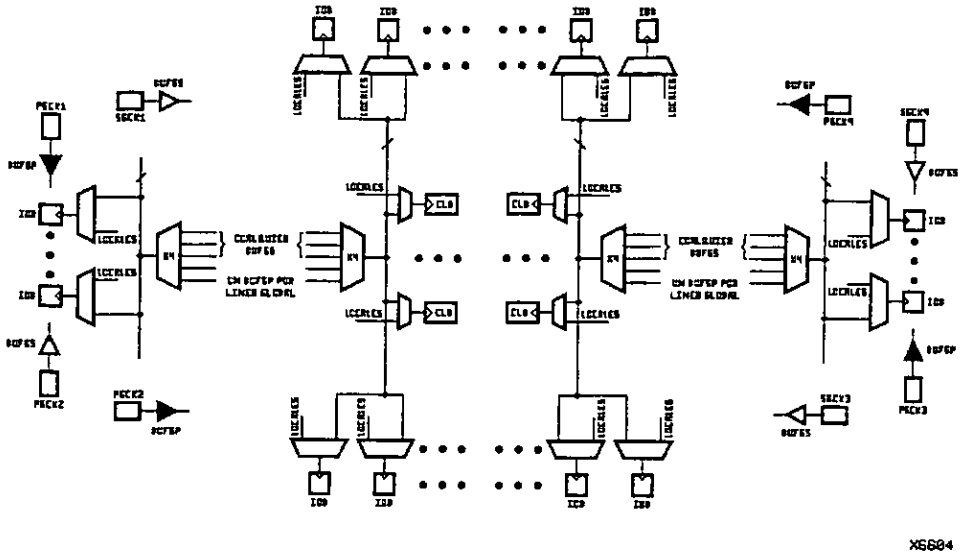


Figura 3.36: Distribución de la Red Global en los XC4000E.

Los IOB's a lo largo de los extremos izquierdo y derecho tienen cuatro líneas largas globales verticales. La parte superior e inferior de los IOB's es temporizada desde las líneas globales en la columna del CLB adyacente.

Un buffer global debe especificarse para toda distribución de señal global de temporización sensitiva. Para usar un buffer global, ponga un elemento BUFGP (buffer primario), BUFGS (buffer secundario), o BUFG (buffer primario o secundario) en un esquemático o en código HDL. Si se desea, se debe ligar un atributo o propiedad de LOC para direccionar la colocación en la posición designada. Por ejemplo, se debe ligar un atributo o propiedad LOC = L a un símbolo de BUFGS de manera que un buffer sea colocado en uno de los dos buffers Globales Secundarios en el extremo izquierdo del dispositivo, o un LOC = BL para indicar el buffer Global Secundario en el extremo inferior del dispositivo, en la parte izquierda.

b) Redes Globales y Buffers (XC4000EX únicamente).

Ocho líneas largas verticales en cada columna del CLB son manejadas por buffers globales especiales. Estas líneas largas se suman a las líneas largas verticales usadas para la interconexión estándar. Las líneas globales se parten en el centro del arreglo, para permitir una distribución más rápida y minimizar el desvío a través de todo el arreglo. En cada mitad de la columna la línea global tiene su propio multiplexor tipo buffer, como se muestra en la Figura 3.37. La parte superior e inferior de las líneas globales no pueden conectarse a través del centro del dispositivo, porque esta conexión podría introducir un desvío inaceptable. La parte superior y la inferior de las líneas globales se deben dividir en dos para manejarse separadamente, aunque ellas pueden ser manejadas por el mismo buffer global.

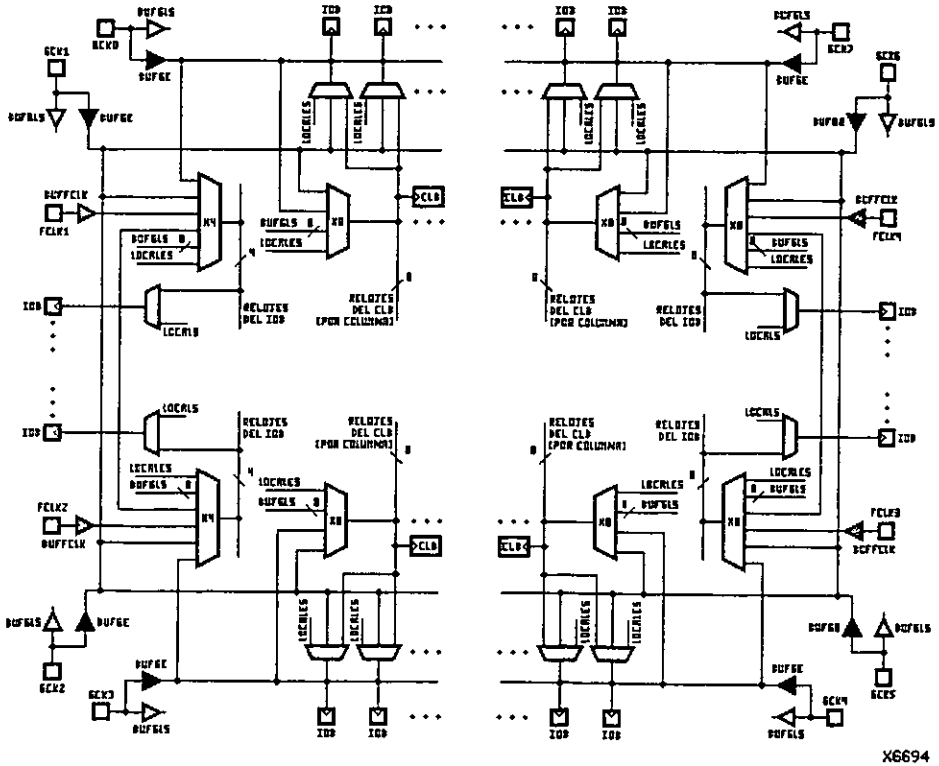


Figura 3.37: Distribución de la Red Global en los XC4000EX.

Las ocho líneas globales en cada columna del CLB pueden ser manejadas por cualquiera de los dos tipos de buffers globales. Ellas también pueden ser manejadas por la lógica interna, ya que pueden ser accesadas por líneas cuádruples, dobles, o simples en los puntos superior, inferior, medio, o un cuarto. Por consiguiente, el número de relojes diferentes que pueden usarse simultáneamente en un dispositivo XC4000EX es muy grande.

Hay cuatro líneas globales que alimentan a los IOB's en el extremo izquierdo del dispositivo. Los IOB's tienen a lo largo del extremo derecho ocho líneas globales. Hay una sola línea global a lo largo de los extremos superiores e inferiores con acceso a los IOB's. Todas las líneas globales del IOB se parten en el centro. Ellas no pueden conectarse a través del centro del dispositivo, porque esta conexión podría introducir un desvío inaceptable.

Las líneas globales del IOB pueden manejarse por cualquiera de los tres tipos de buffers globales, o desde las interconexiones locales. Alternativamente, los IOB's de la parte superior y la inferior pueden temporizarse desde las líneas globales en la columna del CLB adyacente.

Tres tipos diferentes de buffers de reloj están disponibles en los XC4000EX:

- Buffers "Low – Skew" Globales (BUFGLS).
- Buffers "Global Early" (BUFGE).
- Buffers "FastCLK" (BUFFCLK).

Los Buffers "Low – Skew" Globales son los buffers de reloj estándar. Ellos deben usarse para la mayoría de las temporizaciones internas, siempre que una porción grande del dispositivo deba ser manejada.

Los buffers "Global Early" se diseñan para proporcionar un acceso de reloj más rápido, pero el acceso al CLB se limita a un cuarto del dispositivo. Ellos también facilitan una interfaz I / O más rápida.

Los buffers "FastCLK" se diseñan específicamente para proporcionar el más rápido reloj I / O posible. Ellos tienen solo un acceso de entrada estándar a los CLB's, a través de interconexiones locales.

La Figura 3.37 es un diagrama conceptual de la estructura de la red global en los XC4000EX.

Los buffers "Global Early" y los buffers "Low – Skew" Globales comparten un solo "pad". Por consiguiente, el mismo símbolo IPAD puede manejar un buffer de cada tipo, en paralelo. Esta configuración es particularmente útil al usar los latches de Captura Rápida. El par formado por el buffer "Global Early" y el buffer "Low – Skew" Global comparten una entrada común; ellos no pueden manejarse mediante dos señales diferentes.

Selección de un Buffer de Reloj en los XC4000EX.

La estructura de temporización de los XC4000EX proporciona una gran variedad de características. Sin embargo, esto puede ser simple de usar, sin la necesidad de comprender todos los detalles. El software automáticamente maneja los relojes, conjuntamente con todo el enrutamiento, cuando el buffer de reloj apropiado se coloca en el diseño. De hecho, si un símbolo de buffer llamado BUFG se coloca, en lugar de un tipo específico de buffer, el software incluso escoge el buffer más apropiado para el diseño. En esta sección se proporciona información detallada para aquellos usuarios que quieran un nivel más excelente de control sobre sus diseños.

Si se desea un control fino, se debe usar el siguiente resumen y la Tabla 3.15 para escoger un buffer de reloj apropiado.

- Lo más simple es usar un buffer "Low – Skew" Global.
- En caso de que se necesite una ruta de reloj más rápida, pruebe un BUFG. El software intentará usar primero un buffer "Low – Skew" Global. Si las necesidades de temporización no se alcanzan, se usará un buffer más rápido automáticamente.
- En caso de que un solo cuadrante del chip sea suficiente para la lógica de temporización, y las necesidades de temporización requieran de un reloj más rápido que el buffer "Low – Skew", se debe usar un buffer "Global Early".
- En casos especiales, donde ambas temporizaciones externas e internas se han estudiado cuidadosamente, puede usarse un buffer "FastCLK", para la trayectoria de reloj I / O más rápida posible.

Buffers "Low – Skew" (Bajo Desvío) Globales.

Cada esquina del dispositivo XC4000EX tiene dos buffers "Low – Skew" Globales. Cualquiera de los ocho buffers "Low – Skew" Globales pueden manejar cualquiera de las ocho líneas Globales verticales en una columna de los CLB's. En resumen, cualquiera de los buffers puede manejar cualquiera de las cuatro líneas verticales que accesan a los IOB's en el extremo izquierdo del dispositivo, y a cualquiera de las ocho líneas verticales que accesan a los IOB's en el extremo derecho del dispositivo (vea la Figura 3.38).

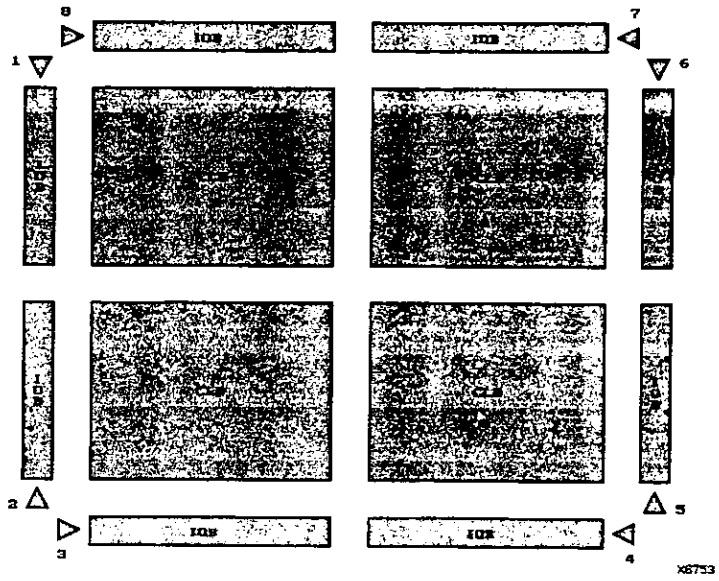


Figura 3.38: Cualquier BUFGLS (GCK1 – GCK8) Puede Manejar a Cualquiera o Todas las Entradas de Reloj en el Dispositivo.

Los IOB's en la parte posterior e inferior del dispositivo se accesan mediante las líneas Globales verticales en el arreglo del CLB, como en el XC4000E. Cualquier buffer "Low – Skew" Global puede, por lo tanto, accesar a cada IOB y CLB en el dispositivo.
Los buffers "Low - Skew" Globales pueden ser manejados por "pad's" semidedicados o por la lógica interna.

Para usar un buffer "Low – Skew" Global, se debe colocar un elemento BUFGLS en un esquemático o en código HDL. Si se desea, se debe ligar un atributo o propiedad LOC para

colocar directamente la posición designada. Por ejemplo, se debe ligar un atributo o propiedad LOC = T para que directamente un BUFGLS sea colocado en uno de los dos buffers "Low – Skew" Globales en el extremo superior del dispositivo, o un LOC = TR para indicar el buffer "Low – Skew" en el extremo superior del dispositivo, a la derecha.

Buffers "Global Early" (Anticipados Globales).

Cada esquina del dispositivo XC4000EX tiene dos buffers "Global Early". El propósito primordial de los buffers "Global Early" es proporcionar un acceso de reloj más anticipado que el de los buffers "Low – Skew" Globales potencialmente mucho más cargados. Un suministro de reloj aplicado a ambos buffers producirá un flanco de reloj anticipado global que ocurre varios nanosegundos antes que el flanco de reloj del buffer "Low – Skew" Global, debido a la carga más ligera.

Los buffers "Global Early" también facilitan la captura rápida de las entradas del dispositivo, usando los latches de Captura Rápida descritos en "Señales de Entrada al IOB". Para la Captura Rápida, tome una sola señal de reloj, y enrutelo a través de ambos buffers, el buffer "Global Early" y el buffer "Low – Skew" Global (Los dos buffers comparten un "pad" de entrada). Se debe usar el buffer "Global Early" como reloj al latch de Captura Rápida, y el buffer "Low – Skew" Global como reloj al flip – flop o latch de entrada normal, como se muestra en la Figura 3.19.

Los buffers "Global Early" también pueden usarse para proporcionar un Reloj de Salida rápido en los pines de salida del dispositivo. Sin embargo, un reloj anticipado en el flip – flop de salida del IOB debe considerarse al calcular la velocidad del reloj interno para el diseño.

Los buffers "Global Early" en los extremos izquierdo y derecho del chip tienen capacidades ligeramente diferentes que los que se encuentran en la parte superior e inferior. Refiérase a la Figura 3.39, a la Figura 3.40, y a la Figura 3.37 mientras lee la siguiente explicación.

Cada buffer "Global Early" puede accesar a las ocho líneas Globales verticales para todos los CLB's en el cuadrante. Por consiguiente, una cuarta parte de los pines de reloj del CLB pueden ser accesados. Esta restricción es en gran parte responsable de la velocidad más rápida de los buffers, relativa a los buffers "Low – Skew" Globales.

Los buffers "Global Early" del lado izquierdo pueden manejar dos de las cuatro líneas verticales que accesan a los IOB's en todo el extremo izquierdo del dispositivo. Los buffers "Global Early" del

lado derecho pueden manejar dos de las ocho líneas verticales que accesan a los IOB's en todo el extremo derecho del dispositivo (vea la Figura 3.39).

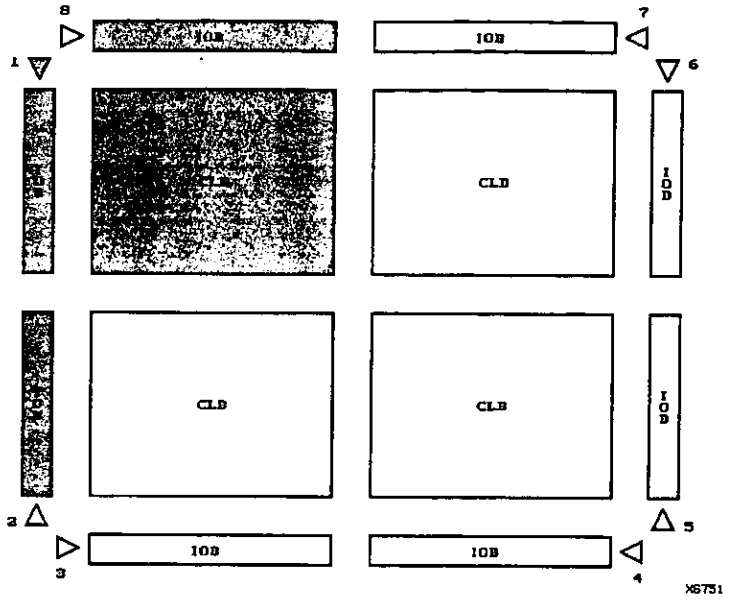


Figura 3.39: Los BUFGE's (Izquierdos y Derechos) Pueden Manejar Cualquiera o Todas las Entradas de Reloj en el Mismo Cuadrante o Extremo (se muestra GCK1. GCK2, GCK5 y GCK6 son similares).

Cada buffer "Global Early" izquierdo y derecho puede manejar también la mitad de los IOB's a lo largo de cualquier extremo superior o inferior del dispositivo, usando una línea dedicada que sólo puede accesarse a través de los buffers "Global Early".

Los buffers "Global Early" de la parte superior e inferior pueden manejar también la mitad de los IOB's a lo largo de cualquier extremo derecho o izquierdo del dispositivo, como se muestra en la Figura 3.40. Ellos pueden accesar únicamente a los IOB's de la parte superior e inferior por medio de las líneas globales del CLB.

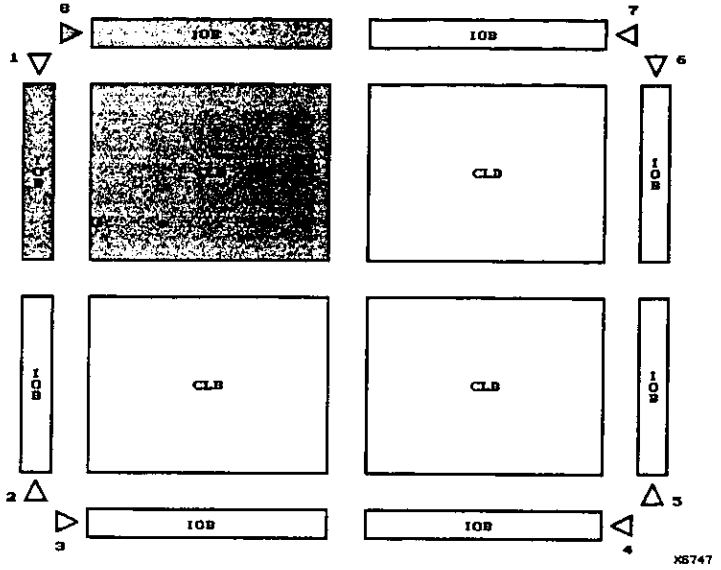


Figura 3.40: Los BUFGE's (Superiores e Inferiores) Pueden Manejar Cualquiera o Todas las Entradas de Reloj en el Mismo Cuadrante (se muestra GCK8. GCK3, GCK4 y GCK7 son similares).

Los buffers "Global Early" pueden ser manejados ya sea por "pad's" semidedicados o por la lógica interna. Ellos comparten "pad's" con los buffers "Low – Skew" Globales, para que una sola red pueda manejar ambos buffers globales, como se describió anteriormente.

Para usar un buffer "Global Early", se debe colocar un elemento BUFGE en un esquemático o en código HDL. Si se desea, se debe ligar un atributo o propiedad LOC para colocar directamente la posición designada. Por ejemplo, se debe ligar un atributo o propiedad LOC = T para que de manera directa un BUFGE sea colocado en uno de los dos buffers "Global Early" en el extremo de la parte superior del dispositivo, o un LOC = TR para indicar el buffer "Global Early" en el extremo superior del dispositivo, en la parte derecha.

Buffers "FastCLK".

La manera más rápida para ofrecer un reloj en los dispositivos XC4000EX es mediante un buffer "FastCLK". Dos buffers "FastCLK" están presentes en el extremo izquierdo, y dos en el extremo derecho, en los últimos XC4000EX. No hay buffers "FastCLK" en los extremos superiores o inferiores.

Un propósito de los buffers "FastCLK" es crear una ruta pin – a – pin más rápida mediante el uso del generador de función de 2 entradas del IOB en conjunción con el "FastCLK". Se debe manejar la entrada F del generador de función del IOB con la salida del buffer "FastCLK", como se describió en "Señales de Salida del IOB".

Alternativamente, un buffer "FastCLK" puede usarse para minimizar el tiempo de establecimiento de las entradas del dispositivo, si un tiempo de retención positivo es aceptable. Se debe usar el buffer "FastCLK" para temporizar al latch de Captura Rápida, y un buffer de reloj más lento para temporizar al latch o flip – flop del IOB estándar. Cualquier buffer "Global Early" o buffer "Low – Skew" Global pueden usarse para el segundo elemento del almacenamiento, pero cualquiera que se use, debe ser el mismo reloj que en la lógica interna relacionada. Ya que los "pad's FastCLK" son diferentes a los "pad's Global Early" y a los "pad's Low – Skew" Globales, se debe tener cuidado para asegurar que ese desvío externo del dispositivo no cree dificultades de temporización internas.

Los buffers "FastCLK" también pueden usarse para proporcionar un Reloj de Salida rápido en los pines de salida del dispositivo. Sin embargo, un reloj rápido en el flip – flop de salida del IOB debe considerarse al calcular la velocidad del reloj interno para el diseño.

Los buffers "FastCLK" se limitan a accesar a los IOB's en una mitad del último extremo únicamente, como se muestra en la Figura 3.41 y en la Figura 3.37. Ellos pueden manejar dos de las cuatro líneas verticales que accesan a los IOB's en el extremo izquierdo del dispositivo, o dos de las ocho líneas verticales que accesan a los IOB's en el extremo derecho del dispositivo. Ellos pueden únicamente accesar a los arreglos del CLB mediante líneas de longitud doble y simple.

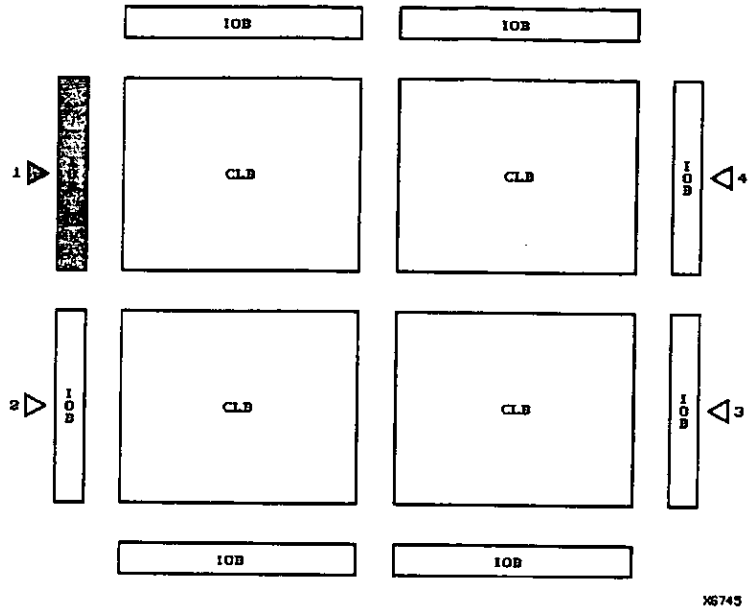


Figura 3.41: Cada BUFFCLK Puede Manejar Cualquiera o Todas las Entradas de Reloj en la Misma Mitad del Extremo (se muestra FCLK1. FCLK2, FCLK3 y FCLK4 son similares).

Los buffers "FastCLK" deben ser manejados por los IOB's semidedicados. Ellos no son accesibles desde las redes internas. Otra característica que los "FastCLK" ofrecen, es que estos IOB's son idénticos a todos los otros IOB's.

Para usar un buffer "FastCLK", se debe colocar un elemento BUFFCLK en un esquemático o en código HDL. Si se desea, se debe ligar un atributo o propiedad LOC para colocar directamente la posición designada. Por ejemplo, se debe ligar un atributo o propiedad LOC = LB para que directamente un BUFFCLK sea colocado en el extremo izquierdo del dispositivo en la parte inferior, o un LOC = L para indicar cualquiera de los buffers en el extremo izquierdo.

La entrada del símbolo BUFFCLK debe ser manejada por un símbolo "pad" de entrada, como IPAD, o por un flip – flop o latch de entrada, tal como INFF, ILD, ILDFFDX, o ILDFLDX.

3.4 DISTRIBUCION DE POTENCIA.

La potencia para el FPGA se distribuye a través de una rejilla para lograr una alta inmunidad al ruido y un aislamiento entre la lógica y las I / O. Dentro del FPGA, un anillo dedicado de Vcc y Ground que circunda al arreglo lógico proporcionan potencia a los manejadores I / O, como se muestra en la Figura 3.42. Una matriz independiente de líneas de Vcc y Ground abastecen a la lógica interna del dispositivo.

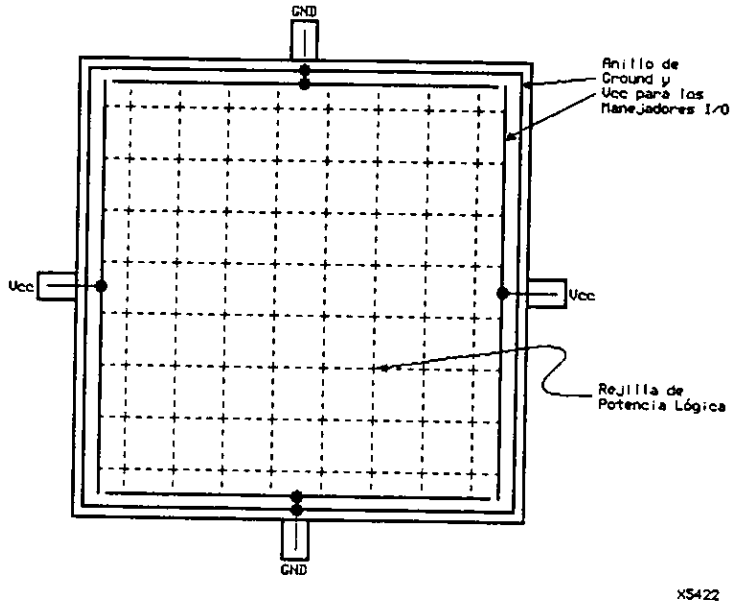


Figura 3.42: Distribución de Potencia en la Serie XC4000.

Esta rejilla de distribución de potencia proporciona un suministro estable y tierra para toda la lógica interna, dando como resultado que los pines de potencia del encapsulado externo estén todos conectados y apropiadamente desacoplados. Típicamente, un capacitor de 0.1 μF. conectado cerca de los pines de Vcc y Ground del encapsulado proporcionará un desacoplo adecuado.

Los buffers de salida son capaces de suministrar / consumir las cargas especificadas de 12 mA. (XC4000E) o de 24 mA. (XC4000EX) bajo las especificaciones de las peores condiciones de funcionamiento pueden ser capaces de suministrar / consumir hasta 10 veces la corriente, bajo las condiciones de funcionamiento más optimas.

El ruido puede ser reducido minimizando la carga externa capacitiva y reduciendo simultáneamente las transiciones de salida en la misma dirección. También puede ser benéfico colocar buffers de salida muy cargados cerca de los "pad's" de Tierra. Los buffers de salida del bloque I / O tienen un modo "slew – rate" limitado (valor por omisión) que debe usarse donde los tiempos de salida a alto y bajo no son críticos para la velocidad.

3.5 DESCRIPCIONES DE LOS PINES.

Existen tres tipos de pines en los dispositivos de la Serie XC4000:

- Pines dedicados permanentemente.
- Pines I / O de usuario que pueden tener funciones especiales.
- Pines I / O programables por el usuario sin ninguna restricción.

Antes y durante la configuración, todas las salidas no usadas por el proceso de configuración son de tercer estado con un resistor "pull – up" de 50 KΩ – 100 kΩ.

Después de la configuración, si un IOB está sin usar se configura como una entrada con un resistor "pull – up" de 50 KΩ – 100 kΩ.

Los dispositivos de la Serie XC4000 no tienen ninguna entrada dedicada "Reset". Cualquier I / O de usuario puede configurarse para manejar a la red Set / Reset Global, GSR.

Los dispositivos de la Serie XC4000 no tienen ninguna entrada de control "Powerdown" (Potencia abajo), en cambio, se usa la red de Tercer Estado Global, GTS. Esta red pone en tercer estado a todas las salidas, pero no pone al dispositivo en modo de bajo consumo de potencia.

Los pines del dispositivo para los dispositivos de la Serie XC4000 se describen en la Tabla 3.16. Las funciones del pin durante la configuración para cada uno de los siete modos de configuración se resumen en la Tabla 3.22, en la sección "Temporización de la Configuración".

Nombre del Pin.	I / O Durante la Configuración	I / O Después de la Configuración.	Descripción del Pin.
Pines Permanentemente Dedicados.			
VCC.	I.	I.	Proporciona ocho o más (dependiendo del encapsulado) conexiones al voltaje nominal + 5V. (+ 3.3 V. para los dispositivos de bajo voltaje). Todos deben conectarse, y cada uno debe ser desacoplado con un capacitor de 0.01 – 0.1 □F conectado a Tierra.
GND.	I.	I.	Ocho o más (dependiendo del tipo de encapsulado) conexiones a Tierra. Todos deben conectarse.
CCLK.	I o O.	I.	Durante la configuración, el Reloj de Configuración (CCLK) es una salida en los modos Maestro o en el modo Periférico Asincrono, pero es una entrada en el modo Esclavo, en el modo Periférico Síncrono, y en el modo Express. Después de la configuración, CCLK tiene un resistor "pull – up" pequeño y puede seleccionarse como el Reloj de "ReadBack" (Lectura Posterior). No hay ninguna restricción de tiempo en estado alto en CCLK, en los dispositivos de la Serie XC4000, excepto durante Lectura Posterior. Vea "Violando la Especificación Máxima de Tiempo en estado Alto y Bajo para el Reloj de Lectura Posterior", para una explicación de esta excepción.
DONE.	I / O.	O.	DONE es una señal bidireccional con un resistor "pull – up" interno opcional. Como una salida, indica la terminación del proceso de configuración. Como una entrada, un nivel bajo en DONE puede configurarse para retardar la inicialización lógica global y la habilitación de las salidas. El resistor "pull – up" opcional se selecciona como una opción en "MakeBits", el programa de "XACTstep" que crea "BitStream" (Cadena de Bits) de la configuración. La resistencia es incluida por omisión.
$\overline{\text{PROGRAM}}$.	I.	I.	$\overline{\text{PROGRAM}}$ es una entrada activa en bajo que forza al FPGA a que limpie su memoria de configuración. Se usa para iniciar un ciclo de configuración. Cuando $\overline{\text{PROGRAM}}$ va a alto, el FPGA termina el ciclo de limpieza actual y ejecuta otro ciclo

Tabla 3.16: Descripción de los Pines.

			de limpieza completo, antes de que entre en un estado de "WAIT" (ESPERA) y libere a $\overline{\text{INIT}}$. El pin $\overline{\text{PROGRAM}}$ tiene un "pull – up" pequeño permanentemente, de tal manera que no necesita conectarse externamente a Vcc.
Pines I / O de Usuario Que Pueden Tener Funciones Especiales.			
RDY / $\overline{\text{BUSY}}$.	O.	I / O.	Durante la configuración en modo Periférico, este pin indica cuando es apropiado escribir otro byte de datos en el FPGA. El mismo estado también está disponible en D7 en el modo Periférico Asíncrono, si una operación de lectura se ha realizado cuando el dispositivo esta seleccionado. Después de la configuración, RDY / $\overline{\text{BUSY}}$ es un pin I / O programable por el usuario. RDY / $\overline{\text{BUSY}}$ es conectado a un estado alto con un "pull – up" de alta impedancia previo al estado alto en $\overline{\text{INIT}}$.
$\overline{\text{RCLK}}$.	O.	I / O.	Durante la configuración Maestra Paralela, cada cambio en las salidas A0 – A7 (A0 – A21 para los XC4000EX) es precedido por un flanco ascendente en $\overline{\text{RCLK}}$, una señal de salida redundante. $\overline{\text{RCLK}}$ es útil para la temporización de las PROM's. Raramente se usa durante la configuración. Después de la configuración, $\overline{\text{RCLK}}$ es un pin I / O programable por el usuario.
M0, M1, M2.	I.	I (M0), O (M1), I (M2).	Como entradas del Modo, estos pines se cambian después de que $\overline{\text{INIT}}$ va a alto para determinar el modo de configuración que va a ser usado. Después de la configuración, M0 y M2 pueden usarse como entradas, y M1 puede usarse como una salida de tercer estado. Estos tres pines no tienen ninguna entrada o salida asociada a registros. Durante la configuración, estos pines tienen resistores "pull – up" pequeños. Para el modo de configuración más popular, Serial Esclavo, los pines de este modo pueden quedar así sin conectar. Las tres entradas del modo pueden configurarse individualmente con o sin el resitor "pull – down" o "pull – up" pequeño. Se recomienda un resistor "pull – down" de un valor de 4.7 KΩ. Estos pines sólo pueden usarse como entradas o salidas cuando son llamados por las definiciones esquemáticas especiales. Para usar estos pines, se deben poner los componentes de biblioteca MD0, MD1, y MD2 en lugar de los símbolos "pad" usuales. Los buffers de entrada o salida deben usarse aún.
TDO.	O.	O.	En caso de que el "Boundary - Scan" sea usado, este pin es la Salida del Dato de Prueba. Si el "Boundary - Scan" no se usa, este pin es una salida de tercer estado sin

Tabla 3.16: Descripción de los Pines (Continuación).

			un registro, después de que la configuración se completa. Este pin puede ser únicamente de salida cuando es llamado por las definiciones esquemáticas especiales. Para usar este pin, se debe poner el componente de biblioteca TDO en lugar del símbolo de "pad" usual. Un buffer de salida debe usarse todavía.
TDI, TCK, TMS.	I.	I / O o I (JTAG).	En caso de que el "Boundary – Scan" sea usado, estos pines son Entrada del Dato de Prueba, Reloj de Prueba, y entradas de Selección de Modo de Prueba, respectivamente. Ellas vienen directamente de los "pad's" y desvían a los IOB's. Estos pines también pueden usarse como entradas a la lógica del CLB después de quo la configuración se completa. Si el símbolo BSCAN no se pone en el diseño, todas las funciones del "Boundary – Scan" son canceladas una vez que la configuración se completa, y estos pines se convierten en I / O programables por el usuario. En este caso, ellos deben ser convocados por las definiciones esquemáticas especiales. Para usar estos pines, se deben poner los componentes de biblioteca TDI, TCK, y TMS en lugar de los símbolos "pad" usuales. Los buffers de entrada o salida deben usarse todavía.
HDC.	O.	I / O.	Alto Durante la Configuración (HDC) se maneja alto hasta que las I / O están activas. Está disponible como una salida de control indicando que esa configuración no se ha completado todavía. Después de la configuración, HDC es un pin I / O programable por el usuario.
$\overline{\text{LDC}}$.	O.	I / O.	Bajo Durante la Configuración ($\overline{\text{LDC}}$) se maneja bajo hasta que las I / O están activas. Está disponible como una salida de control indicando que esa configuración no se ha completado todavía. Después de la configuración, $\overline{\text{LDC}}$ es un pin I / O programable por el usuario.
$\overline{\text{INIT}}$.	I / O.	I / O.	Antes y durante la configuración, $\overline{\text{INIT}}$ es una señal bidireccional. Se recomienda un resistor "pull - up" de 1 KΩ - 10 KΩ externo. Como una salida de colector abierto activa en bajo, $\overline{\text{INIT}}$ se mantiene en bajo durante la estabilización de la potencia y en la limpieza interna de la memoria de configuración. Como una entrada activa en bajo, puede usarse para retener al FPGA en el estado de WAIT (ESPERA) interno antes del comienzo de la configuración. Los dispositivos en modo Maestro se quedan en un estado de WAIT (ESPERA) unos 30 a 300 μs. adicionales después de que $\overline{\text{INIT}}$ ha ido alto. Durante la configuración, un estado bajo en esta salida indica que un error de datos de configuración ha ocurrido.

Tabla 3.16: Descripción de los Pines (Continuación).

			Después de que las I / O son activas, $\overline{INIT}$ es un pin I / O programable por el usuario.
PGCK1 - PGCK4 (XC4000E únicamente).	Pull - up Pequeño.	I o I / O	Cuatro entradas Globales Primarias que manejan cada una, una red global interna dedicada con un retardo corto y un desvío mínimo. En caso de que no sea usado para manejar un buffer global, cualquiera de estos pines es una I / O programable por el usuario. Los pines PGCK1 – PGCK4 manejan a los cuatro Buffers Globales Primarios. Cualquier símbolo "pad" de entrada conectado directamente a la entrada de un símbolo BUFGP es colocado automáticamente en uno de estos pines.
SGCK1 - SGCK4 (XC4000E únicamente).	Pull - up Pequeño.	I o I / O	Cuatro entradas Globales Secundarias que manejan cada una, una red global interna dedicada con un retardo corto y un desvío mínimo. Estas redes globales internas pueden también manejarse desde la lógica interna. Si no se usan para manejar una red global, cualquiera de estos pines es una I / O programable por el usuario. Los pines SGCK1 - SGCK4 proporcionan la ruta más corta a los cuatro Buffers Globales Secundarios. Cualquier símbolo "pad" de entrada conectado directamente a la entrada de un símbolo BUFGS es colocado automáticamente en uno de estos pines.
GCK1 - GCK8 (XC4000EX únicamente).	Pull - up Pequeño.	I o I / O	Ocho entradas que pueden manejar cada una un buffer "Low – Skew" Global. Además, cada uno puede manejar un buffer "Global Early". Cada par de buffers globales también puede manejarse desde la lógica interna, pero debe compartir una señal de entrada. Si no es usado para manejar un buffer global, cualquiera de estos pines es una I / O programable por el usuario. Cualquier símbolo "pad" de entrada conectado directamente a la entrada de un símbolo BUFGLS o BUFGE es colocado automáticamente en uno de estos pines.
FCLK1 - FCLK4 (XC4000EX únicamente).	Pull - up Pequeño.	I o I / O	Cuatro entradas FCLK que pueden manejar cada una un buffer "FastCLK". Los buffers "FastCLK" no pueden manejarse desde la lógica interna. Si no es usado para manejar un buffer global, cualquiera de estos pines es una I / O programable por el usuario. Cualquier símbolo "pad" de entrada conectado directamente a la entrada de un símbolo BUFFCLK es colocado automáticamente en uno de estos pines.
$\overline{CS0}$, CS1, $\overline{WS}$, $\overline{RS}$.	I.	I / O.	Estas cuatro entradas se usan en modo Periférico Asíncrono. El chip es seleccionado cuando $\overline{CS0}$ es bajo y CS1 es alto. Mientras el chip este seleccionado, un estado bajo en "Write Strobe" ($\overline{WS}$) carga los datos presentes en las entradas D0 - D7 en el buffer de datos interno. Un estado bajo en "Read Strobe" ($\overline{RS}$) cambia a D7 dentro del estado de salida, alto si esta Listo, bajo si esta Ocupado, y manejadores altos D0

Tabla 3.16: Descripción de los Pines (Continuación).

			– D6. En modo Express, CS1 es usado como una señal serial de habilitación para la "Daisy Chain" (Cadena de Margarita). $\overline{WS}$ y $\overline{RS}$ deberían ser mutuamente exclusivos, pero si los dos son bajos simultáneamente, el "Write Strobe " es anulado. Después de la configuración, estos son pines I / O programables por el usuario.
A0 – A17.	O.	I / O.	Durante la configuración Paralela Maestra, estos 18 pines de salida direccionan la configuración EPROM. Después de la configuración, ellos son pines I / O programables por el usuario.
A18 – A21 (XC4000EX únicamente).	O.	I / O.	Durante la configuración Paralela Maestra con un XC4000EX Maestro, estos 4 pines de salida añaden 4 bits más para direccionar la configuración EPROM. Después de la configuración, ellos son pines I / O programables por el usuario.
D0 – D7.	I.	I / O.	Durante la configuración Paralela Maestra y Periférica, estos ocho pines de entrada reciben datos de configuración. Después de la configuración, ellos son pines I / O programables por el usuario.
DIN.	I.	I / O.	Durante la configuración Serial Esclava o Serial Maestra, DIN es la entrada de datos de configuración serial receptora de datos, en el flanco ascendente de CCLK. Durante la configuración Paralela, DIN es la entrada D0. Después de la configuración, DIN es un pin I / O programable por el usuario.
DOUT.	O.	I / O.	Durante la configuración en cualquier modo, pero en el modo Express, DOUT es una salida de datos de configuración serial que puede manejar el DIN de los FPGA's esclavos en la cadena de margarita. El dato DOUT cambia en el flanco descendente de CCLK, uno y medio periodos de CCLK después de que se recibió la entrada DIN. En modo Express, DOUT es la salida de estado que puede manejar el CS1 de los FPGA's en la cadena de margarita, para habilitar y deshabilitar dispositivos "Downstream" (Baja Cadena). Después de la configuración, DOUT es un pin I / O programable por el usuario.
Pines I / O Programables por el Usuario sin Ninguna Restricción.			
I / O.	Pull - Up Pequeño.	I / O.	Estos pines pueden configurarse para ser de entrada y / o salida después de que la configuración se completa. Antes de que la configuración se complete, estos pines tienen un resistor interno de alto valor "pull - up" (50 KΩ – 100 KΩ) que definen el nivel lógico como alto.

Tabla 3.16: Descripción de los Pines (Continuación).

3.6 "BOUNDARY SCAN".

La norma "Boundary – Scan" IEEE 1149.1 fue desarrollada para facilitar la prueba a nivel de placa de los ensamblajes electrónicos. Los ingenieros de prueba y diseño pueden infiltrar una estructura lógica de prueba estándar en su dispositivo para alcanzar una alta cobertura de falla para I / O y la lógica interna. Esta estructura se implementa fácilmente con una interfaz de cuatro pines en cualquier CI compatible con "Boundary – Scan".

Las implementaciones de la Serie XC4000 IEEE 1149.1, son compatibles con las instrucciones "Boundary – Scan" BYPASS, PRELOAD / SAMPLE y EXTEST.

Cuando la opción de configuración "Boundary – Scan" se selecciona, tres pines I / O programables por el usuario se convierten en entradas dedicadas para estas funciones. Otro pin de salida de usuario se convierte en una salida "Boundary – Scan" dedicada.

Para ejecutar éstas señales de entrada, el usuario puede cargar comandos y datos serialmente en estos dispositivos para controlar el manejo de sus salidas y examinar sus entradas.

La Figura 3.43 muestra un diagrama de bloques simplificado del Bloque "Input / Output" (Entrada / Salida) de los XC4000E con "Boundary – Scan" implementado. La lógica "Boundary – Scan" XC4000EX es idéntica.

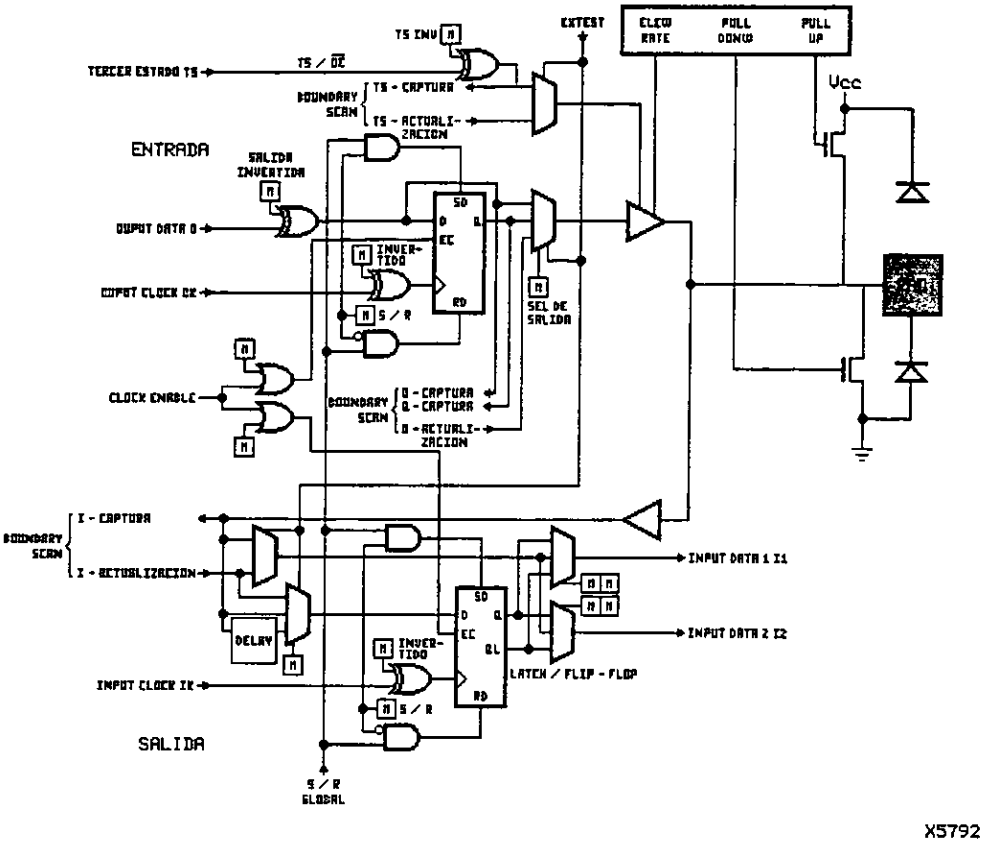


Figura 3.43: Diagrama de Bloques del IOB de los XC4000E con "Boundary Scan" (algunos detalles no son mostrados). La Lógica "Boundary Scan" de los XC4000EX es idéntica.

La Figura 3.44 es un diagrama de la lógica "Boundary – Scan" en la Serie XC4000. Este incluye tres bits de Registro de Datos por IOB, el controlador de Puerto de Acceso de Prueba IEEE 1149.1, y el Registro de Instrucciones con decodificaciones.

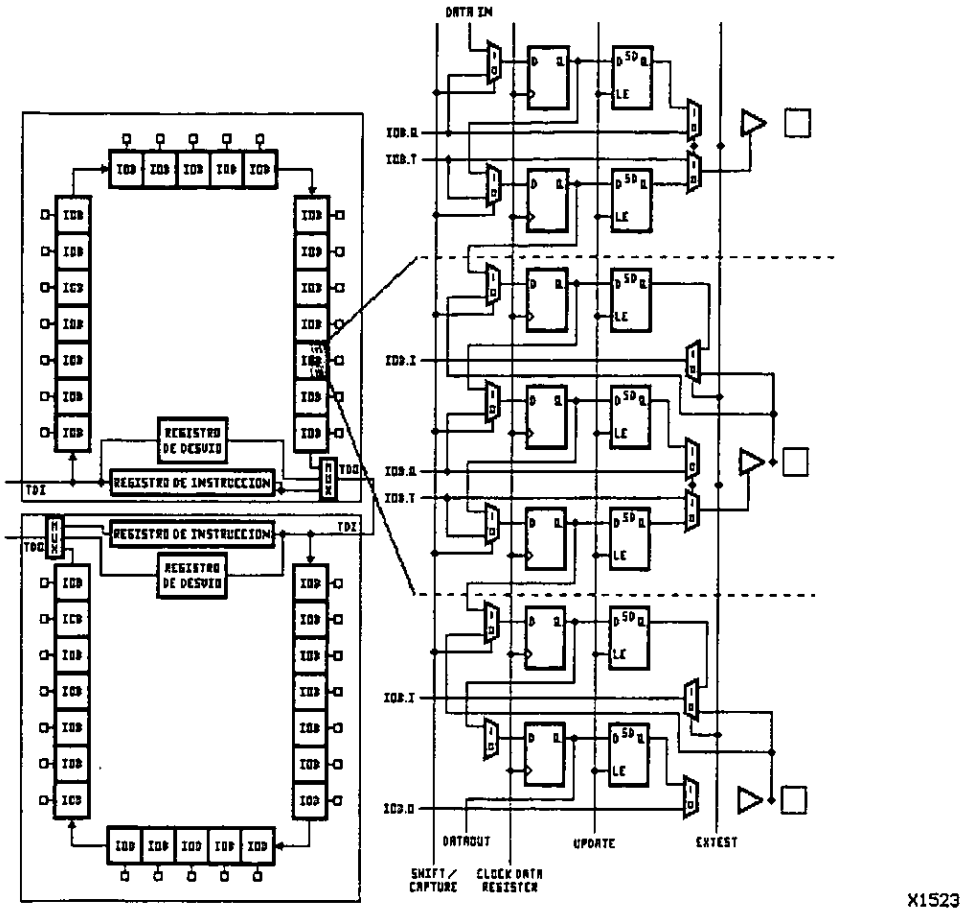


Figura 3.44: Lógica "Boundary Scan" de la Serie XC4000.

También pueden configurarse los dispositivos de la Serie XC4000 a través de la lógica "Boundary - Scan". Vea "Configuración a través de los Pines Boundary Scan".

3.6.1 Registros de Datos.

El registro de datos primario es el registro "Boundary – Scan". Por cada pin del IOB en el FPGA, garantizado o no, este incluye tres bits para Entrada (In), Salida (Out) y Control de Tercer Estado. Los pines que no son del IOB tienen un número de bits parciales apropiados para Entrada (In) o Salida (Out), únicamente. $\overline{\text{PROGRAM}}$, CCLK y DONE no están incluidos en el registro "Boundary – Scan". Cada estado EXTEST CAPTURE-DR captura todos los pines de Entrada (In), Salida (Out), y tercer estado.

El registro de datos también incluye a los siguientes bits (no pines): TDO.T, y TDO.O, los cuales siempre son bits 0 y 1 del registro de datos, respectivamente; y BSCANT.UPD, el cual siempre es el último bit del registro de datos. Estos tres bits "Boundary – Scan" son señales de prueba de Xilinx de propósito especial.

El otro registro de datos estándar es un flip – flop único de registro "BYPASS" (de Desvío). Este sincroniza los datos que se pasan a través del FPGA a la siguiente cadena de bits "Boundary – Scan" del dispositivo.

El FPGA proporciona dos registros de datos adicionales que pueden especificarse usando el macro BSCAN. El FPGA proporciona dos pines de usuario (BSCAN.SEL1 y BSCAN.SEL2) que decodifican las dos instrucciones de usuario. Para estas instrucciones, dos pines correspondientes (BSCAN.TDO1 y BSCAN.TDO2) permiten a los datos de prueba de usuario ser transferidos al exterior en TDO. El reloj del registro de datos (BSCAN.DRCK) está disponible para el control de la lógica de prueba que el usuario puede implementar si desea con CLB's. La NAND de TCK y el RUN-TEST- IDLE también se proporcionan (BSCAN.IDLE).

3.6.2 Conjunto de Instrucciones.

El conjunto de instrucciones "Boundary – Scan" de la Serie XC4000 también incluye instrucciones para configurar al dispositivo y para la configuración de lectura posterior de los datos. El juego de instrucciones es codificado como se muestra en la Tabla 3.17.

Instrucción.			Selección	Suministro	Suministro de
I2	I1	I0	de Prueba.	TDO.	Datos I / O.
0	0	0	EXTEST.	DR.	DR.
0	0	1	SAMPLE / PRELOAD.	DR.	Pin / Lógica.
0	1	0	USER 1.	BSCAN. TDO1.	Lógica de Usuario.
0	1	1	USER2.	BSCAN. TDO2.	Lógica de Usuario.
1	0	0	READBACK.	Lectura Posterior de Datos.	Pin / Lógica.
1	0	1	CONFIGURE.	DOUT.	Deshabilitado.
1	1	0	Reservado.	-	-
1	1	1	BYPASS.	Registro de Desvío.	-

Tabla 3.17: Instrucciones "Boundary Scan".

3.6.3 Secuencia de Bits.

La secuencia de bits dentro de cada IOB es: Entrada (In), Salida (Out), Tercer Estado. La única entrada de los pines de modo M0 y M2 contribuye únicamente con el bit "In" para el registro de datos I / O "Boundary – Scan", mientras que la única salida pin M1 contribuye con los tres bits.

Los primeros dos bits en el registro de datos I / O son TDO.T y TDO.O, los cuales pueden usarse para la captura de señales internas. El bit final es BSCANT.UPD, el cual puede ser usado para manejar una red interna. Estas localidades son usadas principalmente por Xilinx para la prueba interna.

Desde una vista de la cavidad superior del chip, comenzando en la esquina superior derecha del chip, los bits del registro de datos "Boundary – Scan" se ordenan como se muestra en la Figura

3.45. Las tablas de "Pin – Out" (Pin de Salida) para un dispositivo específico de la Serie XC4000 incluyen las localidades "Boundary – Scan" para cada pin IOB.

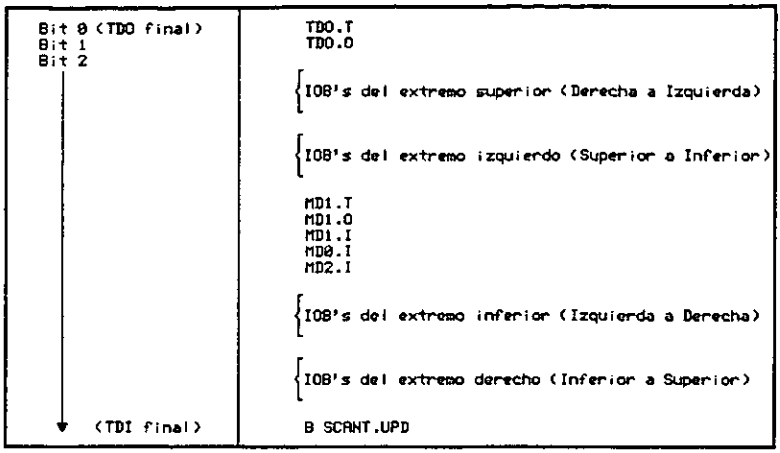


Figura 3.45: Secuencia de Bits "Boundary – Scan".

3.6.4 Para Incluir "Boundary Scan" en un Esquemático.

En caso de que "Boundary – Scan" sólo sea usado durante la configuración, ningún elemento esquemático especial necesita incluirse en el esquemático o código HDL. En este caso, los pines especiales "Boundary – Scan" TDI, TMS, TCK y TDO pueden usarse para funciones de usuario después de la configuración.

Para indicar que ese "Boundary – Scan" permanezca habilitado después de la configuración, se debe poner el símbolo de biblioteca BSCAN y conectar los símbolos "pad" TDI, TMS, TCK y TDO a los pines apropiados, como se muestra en la Figura 3.46.

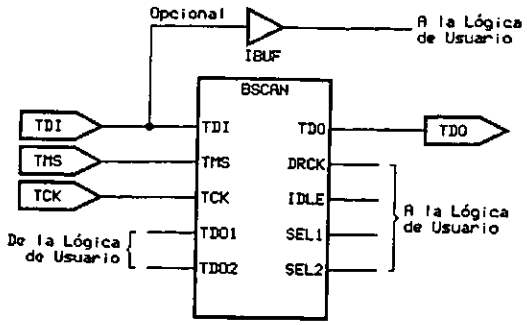


Figura 3.46: Ejemplo Esquemático "Boundary - Scan".

Aún cuando el símbolo "Boundary – Scan" se use en un esquemático, los pines de entrada TMS, TCK, y TDI todavía pueden usarse como entradas para ser enrutadas a la lógica interna. Debe de tenerse cuidado para no forzar al chip a un estado "Boundary – Scan" indeseado aplicando inadvertidamente patrones de entrada "Boundary – Scan" a estos pines. La manera más simple de prevenir esto, es mantener TMS en estado alto, y entonces aplicar cualquier señal que se desee a TDI y TCK.

3.6.5 Prevención de Activaciones Inadvertidas "Boundary Scan".

En caso de que TMS o TCK se usen como I / O de usuario, debe de tenerse cuidado en asegurar que por lo menos uno de estos pines se mantenga constante durante la configuración. En algunas aplicaciones, una situación indeseada puede ocurrir cuando TMS o TCK se usen durante la configuración. Esto puede ocasionar que el dispositivo entre en un modo "Boundary – Scan" y desbarate el proceso de configuración.

Para prevenir activaciones de "Boundary – Scan" durante la configuración, realice cualquiera de las recomendaciones siguientes:

- TMS: se debe ligar un estado alto para poner al controlador de Puerto de Acceso de Prueba en un estado "Reset" benéfico.
- TCK: se debe ligar un estado alto o bajo, no "Toggle" a esta entrada de reloj.

3.7 CONFIGURACION.

La configuración es el proceso de cargar los datos de programación de diseño específicos en uno o más FPGA's para definir la operación funcional de los bloques internos y sus interconexiones. Esto es algo así como cargar los registros de comando de un chip periférico programable. Los dispositivos de la Serie XC4000 usan varios cientos de bits de datos de configuración por CLB y sus interconexiones asociadas. Cada bit de configuración define el estado de una celda de memoria estática que controla ya sea una función de la tabla de búsqueda binaria, un multiplexor de entrada, o un transistor de interconexión de paso. El sistema de desarrollo XACT*Step* traduce el diseño en un archivo "Netlist". Estas particiones automáticas, asignan y enrutan la lógica y generan los datos de configuración en formato PROM.

3.7.1 Pines de Propósito Especial.

Tres pines de configuración de modo (M2, M1, M0) se emplean para determinar el modo de configuración. Después de la configuración, estos pines pueden usarse como conexiones auxiliares. M2 y M0 pueden usarse como entradas, y M1 puede usarse como una salida. El sistema de desarrollo XACT*step* no usa estos recursos a menos que estos se especifiquen explícitamente en la entrada del diseño. Esto se hace poniendo un símbolo "pad" especial llamado MD2, MD1, o MD0 en vez del símbolo "pad" de entrada o salida.

En los dispositivos de la Serie XC4000, los pines de modo tienen resistores "pull – up" pequeños durante la configuración. Con los tres pines de modo en estado alto, el modo Serial Esclavo se selecciona, el cual es el modo de configuración más popular. Por consiguiente, para el modo de configuración más común, los pines de modo pueden quedar sin conectar (note, sin embargo, que el valor del resistor "pull – up" interno puede ser tan alto como 100 kΩ). Después de la configuración, estos pines pueden tener individualmente resistores "pull – down" o "pull – up" pequeños, dependiendo de como se especifique en el diseño. Se recomienda un resistor "pull – down" de un valor de 4.7 KΩ.

Estos pines se localizan en la esquina inferior izquierda del chip y están cerca de las redes de lectura posterior. Esta ubicación permite un enrutamiento conveniente si se desea compatibilidad de M0 / RT y M1 / RD, con las familias XC2000 y XC3000 convencionales.

3.7.2 Modos de Configuración.

Los dispositivos XC4000E tienen seis modos de configuración. Los dispositivos XC4000EX tienen los mismos seis modos, más un modo de configuración adicional. Estos modos son seleccionados por un código de entrada de 3 bits aplicado a las entradas M2, M1, y M0. Existen tres modos Maestros con una misma forma de carga, dos modos Periféricos, y un modo Serial Esclavo, el cual se usa principalmente para los dispositivos en cadena de margarita. El séptimo modo, llamado modo Express, es un modo esclavo adicional que permite la configuración paralela de alta velocidad de los dispositivos XC4000EX de alta capacidad. La codificación para la selección del modo se muestra en la Tabla 3.18.

MODO.	M2.	MI.	M0.	CCLK.	Datos.
Serial Maestro.	0	0	0	salida.	Bit – Serial.
Serial Esclavo.	1	1	1	entrada.	Bit – Serial.
Paralelo Maestro (Alto).	1	0	0	salida.	Un Byte Ancho, incremento desde 00000.
Paralelo Maestro (Bajo).	1	1	0	salida.	Un Byte Ancho, decremento desde 3FFFF.
Periférico Síncrono.*	0	1	1	entrada.	Un Byte Ancho.
Periférico Asíncrono.	1	0	1	salida.	Un Byte Ancho.
Express (XC4000EX únicamente).	0	1	0	entrada.	Un Byte Ancho.
Reservado.	0	0	1	–	–

Nota: * El modo Periférico Síncrono puede ser considerado Paralelo Esclavo de byte ancho.

Tabla 3.18: Modos de Configuración.

Una descripción detallada de cada modo de configuración, con información de temporización, se incluye más adelante. Durante la configuración, algunos pines I / O se usan temporalmente para el proceso de configuración. Todos los pines usados durante la configuración se muestran en la Tabla 3.22.

a) Modos Maestros.

Los tres modos Maestros usan un oscilador interno para generar el Reloj de Configuración (CCLK) para manejar a los dispositivos esclavos potenciales. Ellos también generan direccionamiento y temporización para PROM o PROM's externas conteniendo los datos de configuración.

Los modos Paralelos Maestros (Alto y Bajo) generan la señal CCLK y las direcciones PROM y reciben un byte de datos en forma paralela. El dato es internamente serializado en el FPGA con formato de estructura de datos. Las selecciones Alto y Bajo generan direcciones empezando ya sea en Cero o 3FFFF, para compatibilidad con diferentes microprocesadores convencionalmente direccionados. El modo Serial Maestro genera un CCLK y recibe los datos de configuración en forma serial desde una PROM de configuración serial de Xilinx.

La velocidad de CCLK es seleccionable entre 1 MHz. (valor por omisión) u 8 MHz. (hasta un 10% menos para los dispositivos de bajo voltaje). La configuración siempre empieza en la frecuencia baja por predefinición, entonces puede cambiarse a la frecuencia más alta durante la primera estructura. La tolerancia de la frecuencia es -50% a +25%.

b) Modos Periféricos.

Los dos modos Periféricos admiten los datos en un byte ancho desde un bus. El estado RDY / $\overline{\text{BUSY}}$ está disponible como una señal de confirmación. En el modo Periférico Asíncrono, el oscilador interno genera una señal CCLK que serializa los datos en un byte ancho. CCLK también puede manejar dispositivos "Esclavo". En el modo síncrono, una entrada de reloj proporcionada externamente a CCLK serializa los datos.

c) Modo Serial Esclavo.

En el modo Serial Esclavo, el FPGA recibe los datos de la configuración serialmente en un flanco ascendente de CCLK y, después de cargar su configuración, pasa datos de salida adicionales, resincronizandose en el próximo flanco descendente de CCLK.

Los dispositivos Esclavo múltiples con configuraciones idénticas pueden ser alambrados con entradas DIN paralelas. De esta manera, múltiples dispositivos pueden configurarse simultáneamente.

Cadena de Margarita (Daisy Chain) Serial.

Múltiples dispositivos con configuraciones diferentes pueden conectarse juntos en una cadena de margarita, y sólo una cadena de bits es usada para configurar a la cadena de dispositivos "esclavo".

Para configurar una cadena de margarita de dispositivos, alambre los pines CCLK de todos los dispositivos en paralelo, como se muestra en la Figura 3.56. Se debe conectar el DOUT de cada dispositivo en el DIN del siguiente. El FPGA maestro o principal y cada uno de los esclavos siguientes pasan los datos de configuración resincronizados que vienen de un solo suministro. Los datos del encabezado, incluyendo el conteo de longitud, se pasan a través de ellos y son capturados por cada FPGA cuando reconoce el preámbulo 0010. Siguiendo los datos de conteo de longitud, cada salida DOUT de los FPGA's permanece en alto hasta que haya recibido su número necesario de estructuras de datos.

Después de que un FPGA ha recibido sus datos de configuración, pasa en cualquier estructura adicional los bits de inicio y los datos de configuración en DOUT. Cuando el número total de relojes de configuración aplicados después de la inicialización de la memoria iguala el valor de conteo de longitud de 24 bits, los FPGA's empiezan la secuencia de puesta en marcha y se ponen en operación simultáneamente. Las I / O del FPGA son liberadas normalmente dos ciclos CCLK después de que el último bit de configuración se recibe. La Figura 3.50 muestra la temporización de la puesta en marcha para un dispositivo de la Serie XC4000.

La cadena de bits de la cadena de margarita no es simplemente una concatenación de cadenas de bits individuales. El programa "MakePROM" debe usarse para combinar las cadenas de bits para una configuración cadena de margarita.

Cadena de Margarita (Daisy Chain) Multi Familia.

Todos los FPGA's de Xilinx de la Serie XC2000, XC3000, y XC4000 usan un formato de cadena de bits compatible y pueden, por consiguiente, ser conectados en una cadena de margarita en una secuencia arbitraria. Existe, sin embargo, una limitación. El dispositivo principal debe pertenecer a la familia más alta en la cadena. En caso de que la cadena contenga dispositivos de la Serie XC4000, el maestro normalmente no puede ser un dispositivo XC2000 o XC3000.

La razón para esta regla se muestra en la Figura 3.50. Ya que todos los dispositivos en la cadena almacenan el mismo valor de conteo de longitud y generan o reciben una secuencia común de pulsos de CCLK, todos ellos reconocen el conjunto de conteos de longitud en el mismo flanco de CCLK, como se indica en el extremo izquierdo de la Figura 3.50. El dispositivo maestro entonces genera pulsos CCLK adicionales hasta que alcanza su punto de finalización F. Las familias diferentes generan o necesitan diferentes números de pulsos CCLK adicionales hasta que ellos alcancen F. No alcanzar F significa que el dispositivo realmente no ha terminado su configuración, aunque DONE pueda estar en alto, las salidas estén activas, y el "Reset" interno se haya liberado. Para los dispositivos de la Serie XC4000, no alcanzar F significa que una lectura posterior no puede iniciarse y la mayoría de las instrucciones "Boundary – Scan" no pueden usarse.

El usuario tiene algún control sobre la temporización relativa de estos eventos y puede, por consiguiente, asegurar que ellos ocurrirán en el momento apropiado y que el punto final F sea alcanzado. La temporización se controla usando opciones de "MakeBits".

XC3000 Maestro con un Esclavo de la Serie XC4000.

Algunos diseñadores desean usar un dispositivo principal barato en el modo periférico y tener los pines I / O más importantes de los dispositivos de la Serie XC4000 completamente disponibles para I / O de usuario. La Figura 3.47 proporciona una solución para ese caso.

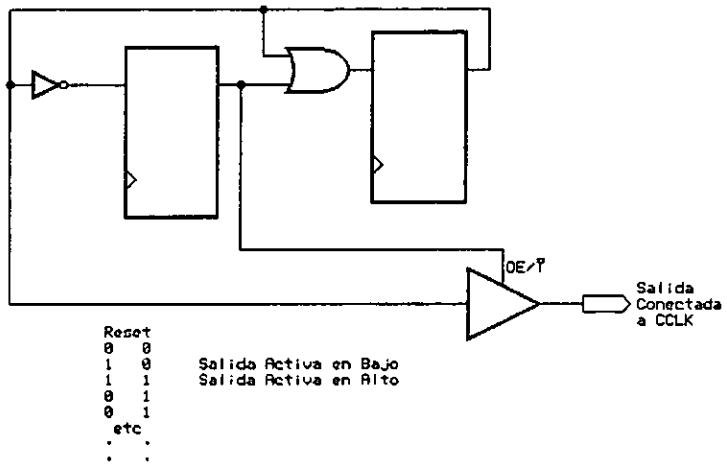


Figura 3.47: Generación de CCLK para un XC3000 Maestro que Maneja un Esclavo de la Serie XC4000.

Esta solución necesita un CLB, un IOB y un pin, y un oscilador interno con una frecuencia de hasta 5 MHz. como suministro de reloj. El dispositivo maestro XC3000 debe configurarse con "Reset" Interno retardado, el cual es la opción predefinida.

Un CLB y un IOB en el dispositivo principal de la familia XC3000 se usan para generar el pulso CCLK adicional requerido por los dispositivos de la Serie XC4000. Cuando el dispositivo principal quita la señal "RESET" interna, el registro de corrimiento de 2 bits responde a su entrada de reloj y genera una señal de salida activa en bajo mientras dure el periodo de reloj subsecuente. Una conexión externa entre esta salida y CCLK crea así el pulso CCLK extra.

d) Modo Express (XC4000EX únicamente).

El modo Express es similar al modo Serial Esclavo, excepto que los datos están presentes en formato paralelo, y que se temporiza en el dispositivo destinado un byte a la vez en lugar de un bit

a la vez. Los datos se cargan en paralelo en ocho columnas diferentes: no son serializados internamente. Ocho bits de datos de configuración se cargan en cada ciclo CCLK, por lo tanto en este modo de configuración el índice de datos corre ocho veces más rápido que en los otros seis modos. No se emplea un conteo de longitud en el modo Express.

El modo Express debe especificarse como una opción en el programa "MakeBits", que genera la cadena de bits. La cadena de bits del modo Express no es compatible con el de los otros seis modos de configuración.

Los múltiples dispositivos "esclavo" con configuraciones idénticas pueden alambrarse con las entradas D0 - D7 en paralelo. De esta manera, múltiples dispositivos pueden configurarse simultáneamente.

Pseudo Cadena de Margarita (Daisy Chain).

Múltiples dispositivos con configuraciones diferentes pueden conectarse juntos en una pseudo cadena de margarita, con tal de que todos los dispositivos estén en modo Express. Una sola cadena de bits combinada es usada para configurar la cadena de dispositivos en modo Express, pero el bus de datos de entrada debe manejar D0 - D7 para cada dispositivo. Se debe ligar un estado alto al pin CS1 del primer dispositivo a ser configurado. Se debe conectar el pin DOUT de cada FPGA al pin CS1 del siguiente dispositivo en la cadena. Las entradas D0 – D7 son alambradas en paralelo en cada dispositivo. Los pines DONE se alambran juntos, con uno o más "pull – up's" DONE internos activados. Alternativamente, un resistor externo de 4.7 kΩ puede usarse, si se desea (vea la Figura 3.64).

El requerimiento de que todos los pines DONE en una cadena de margarita sean alambrados juntos sólo se aplica para el modo Express, y sólo si todos los dispositivos en la cadena llegan a activarse simultáneamente. Todos los dispositivos XC4000EX en modo Express se sincronizan con el pin DONE. Las I / O de usuario para cada dispositivo llegan a ser activas después de que el pin DONE por cada dispositivo pasa a alto (la temporización exacta es determinada por las opciones de "MakeBits"). Ya que el pin DONE es de colector abierto y no maneja un valor alto, ligando los pines DONE de todos los dispositivos juntos se previene que todos los dispositivos en la cadena lleguen a un estado alto hasta que el último dispositivo en la cadena haya completado su ciclo de configuración.

Debido a que únicamente los dispositivos XC4000EX y XC5200 soportan el modo Express, únicamente estos dispositivos pueden usarse para formar una cadena de margarita en modo Express. Los dispositivos XC5200 usados en una cadena de margarita combinada con dispositivos XC4000EX deberán configurarse como sincronizados con DONE (la opción de "MakeBits" CCLK_SYNC o UCLK_SYNC), y sus pines DONE deberán ser alambrados juntamente con los de los dispositivos XC4000EX.

3.7.3 Fijando la Frecuencia de CCLK .

Para los modos Maestros, CCLK puede generarse en cualquiera de dos frecuencias. En el modo lento predefinido, el rango de frecuencia va desde 0.5 MHz. hasta 1.25 MHz. (hasta un 10% menos para los dispositivos de bajo voltaje). En modo CCLK rápido, los rangos de frecuencia van desde 4 MHz. hasta 10 MHz. (hasta un 10% menos para los dispositivos de bajo voltaje). La frecuencia es seleccionada mediante una opción cuando se corre "MakeBits", la herramienta de software de generación de cadena de bits. En caso de que un Maestro de la Serie XC4000 esté manejando un esclavo de la familia XC3000, o XC2000, debe usarse el modo CCLK lento. El modo lento es el predefinido.

3.7.4 Formato de Cadena de Datos.

El formato de cadena de datos ("Bitstream") es idéntico para todos los modos de configuración, con excepción del modo Express. En modo Express, el dispositivo se pone activo cuando DONE llega a un estado alto, por consiguiente, ningún conteo de longitud es necesario. Adicionalmente, el chequeo de error CRC no es soportado en modo Express.

Los formatos de cadena de datos se muestran en la Tabla 3.19. Los datos del modo Express se muestran con D0 a la izquierda y D7 a la derecha. Para todos los otros modos, los datos de bit serial se leen de izquierda a derecha, y los datos de byte paralelo son efectivamente ensamblados desde esta cadena de bits serial, con el primer bit en cada byte asignado a D0.

Tipo de Datos.	Modo Express. (D0 – D7).	Todos los Otros Modos (D0...).
Byte de Relleno.	11111111b.	11111111b.
Código de Preámbulo.	11110010b.	0010b.
Conteo de Longitud.	FFFFFFh.	COUNT(23:0).
Bits de Relleno.	–	1111b.
Campo de Inicio.	11010010b.	0b.
Estructura de Datos.	DATOS(n-1:0).	DATOS(n-1:0).
CRC o Chequeo de Campo Constante.	11010010b.	xxxx (CRC) o 0110b.
Ciclo de Escritura Extendido.	FFFFFFFFFFh	–
Postámbulo.	–	01111111b.
Bytes de Puesta en Marcha.	xxxxxxxxh.	xxh.

LEYENDA:

Sin Sombrear.	Una vez por cadena de bits.
Sombreado Claro.	Una vez por estructura de datos.
Sombreado Oscuro.	Una vez por dispositivo.

Tabla 3.19: Formatos de Cadena de Datos de la Serie XC4000.

La configuración de cadena de datos comienza con una cadena de ocho unos, un código de preámbulo, seguido por un conteo de longitud de 24 bits y un campo separador de unos (o 24 bits de relleno, en el modo Express). Este encabezado es seguido por los datos de la configuración actual en las estructuras. La longitud y el número de estructuras dependen del tipo de dispositivo (vea la Tabla 3.20 y la Tabla 3.21). Cada estructura comienza con un campo de inicio y finaliza con un chequeo de error. En todos los modos excepto en el modo Express, un código de "post – ámbulo " se requiere para señalizar el fin de los datos para un solo dispositivo. En todos los casos, los bytes de datos de puesta en marcha adicionales se necesitan para proporcionar cuatro relojes para la secuencia de puesta en marcha al final de la configuración. Las cadenas de margarita largas necesitan bytes de puesta en marcha adicionales para cambiar los últimos datos

a través de la cadena. Todos los bytes de puesta en marcha no tienen importancia; estos bytes no son incluidos en las cadenas de bits creadas por el software de Xilinx.

Dispositivo.	XC4003E.	XC4005E/L.	XC4006E.	XC4008E.	XC4010E/L.	XC4013E/L.	XC4020E.	XC4025E
Max. Compuertas Lógicas.	3,000	5,000	6,000	8,000	10,000	13,000	20,000	25,000
CLB's (Renglón x Columna).	100 (10x10)	196 (14x14)	256 (16x16)	324 (18x18)	400 (20x20)	576 (24x24)	784 (28x28)	1,024 (32x32)
IOB's.	80	112	128	144	160	192	224	256
Flip – Flops.	360	616	768	936	1,120	1,536	2,016	2,560
Líneas Largas Horizontales.	20	28	32	36	40	48	56	64
TBUF's por Línea Larga.	12	16	18	20	22	26	30	34
Bits por Estructura.	126	166	186	206	226	266	306	346
Estructuras.	428	572	644	716	788	932	1,076	1,220
Datos de Programa.	53,936	94,960	119,792	147,504	178,096	247,920	329,264	422,128
Tamaño PROM (bits).	53,984	95,008	119,840	147,552	178,144	247,968	329,312	422,176

Notas: 1. Bits por Estructura = (10 x número de renglones) + 7 por la parte superior + 13 por la parte inferior + 1 + 1 bit de arranque + 4 bits de chequeo de error.

Número de Estructuras = (36 x número de columnas) + 26 por el extremo izquierdo + 41 por el extremo derecho + 1.

Datos de Programa = (Bits por Estructura x Número de Estructuras) + 8 bits "post- ámbulo".

Tamaño PROM = Datos de Programa + 40.

2. El usuario puede agregar más bits "uno" como bits falsos principales en el encabezado, o, en caso de que CRC = 'off' (fuera de servicio), como una ruta de bits falsos al final de cualquier estructura, siguiendo con los cuatro bits de chequeo de error. Sin embargo, el valor del Conteo de Longitud debe ajustarse para todos esos bits "uno" extras, incluso para los unos principales extras al principio del encabezado.

Tabla 3.20: Datos de Programa de los XC4000E.

Dispositivo.	XC4028EX/XL	XC4036EX/XL	XC4044EX/XL	XC4052XL	XC4062XL
Max. Compuertas Lógicas.	28,000	36,000	44,000	52,000	62,000
CLB's (Renglón x Columna).	1,024 (32x32)	1,296 (36x36)	1,600 (40x40)	1,936 (44x44)	2,304 (48x48)
IOB's.	256	288	320	352	384
Flip – Flops.	2,560	3,168	3,840	4,576	5,376
Líneas Largas Horizontales.	192	216	240	264	288
TBUF's por Línea Larga.	34	38	42	46	50
Bits por Estructura.	421	469	517	565	613
Estructuras.	1587	1775	1963	2151	2,339
Datos de Programa.	668,127	832,483	1,014,879	1,215,323	1,433,807
Tamaño PROM (bits).	668,167	832,523	1,014,919	1,215,363	1,433,847

Notas: 1. Bits por Estructura = (12 x número de renglones) + 8 por la parte superior + 16 por la parte inferior + 8 + 1 bit de arranque + 4 bits de chequeo de error.

Número de Estructuras = (47 x número de columnas) + 27 por el extremo izquierdo + 52 por el extremo derecho + 4.

Datos de Programa = (Bits por Estructura x Número de Estructuras) + 8 bits "post- ámbulo".

Tamaño PROM = Datos de Programa + 40.

2. El usuario puede agregar más bits "uno" como bits falsos principales en el encabezado, o, en caso de que CRC = "off" (fuera de servicio), como una ruta de bits falsos al final de cualquier estructura, siguiendo con los cuatro bits de chequeo de error. Sin embargo, el valor del Conteo de Longitud debe ajustarse para todos esos bits "uno" extras, incluso para los unos principales extras al principio del encabezado.

3. En modo Express los "Bitfilos" (Archivos de Bits) son ligeramente más grandes (vea la Tabla 2.19).

Tabla 3.21: Datos de Programa de los XC4000EX.

El software de "MakeBits" crea la cadena de bits de configuración. En modo Express, sólo el chequeo de error que no es CRC es soportado. En todos los otros modos, "MakeBits" permite una selección de CRC o chequeo de error que no es CRC. Las pruebas de chequeo de error que no es CRC están diseñadas para un campo de fin de estructura para cada estructura. Para un chequeo de error CRC, "MakeBits" calcula una corrida CRC e inserta un chequeo parcial de 4 bits único en el final de cada estructura. El chequeo CRC en el bit 11 de la última estructura de un FPGA incluye los últimos siete bits de datos.

La detección de un error resulta en la suspensión del cargado de datos y el "pulling – down" del pin $\overline{INIT}$. En los modos Maestros, CCLK y las señales de dirección continúan operando externamente. El usuario debe detectar $\overline{INIT}$ e inicializar una nueva configuración pulsando el pin $\overline{PROGRAM}$ a bajo o ciclando a Vcc.

3.7.5 Chequeo Redundante Cíclico (CRC) para la Configuración y Lectura Posterior.

El Chequeo Redundante Cíclico es un método de detección de error en aplicaciones de transmisión de datos. Generalmente, el sistema transmisor realiza un cálculo en la cadena de bits serial. El resultado de este cálculo se etiqueta en la cadena de datos como bits de chequeo adicionales. El sistema receptor realiza un cálculo idéntico en la cadena de bits y compara el resultado con el "Checksum" (Resumen del Chequeo) recibido.

Cada estructura de datos de la cadena de bits de configuración tiene cuatro bits de error al final, como se muestra en la Tabla 3.19. Si un error de estructura de datos se detecta durante la carga del FPGA, el proceso de configuración con una cadena de bits potencialmente corrupta se termina. El FPGA va a bajo en el pin $\overline{INIT}$ y entra en un estado de "Wait" (Espera).

Durante la Lectura Posterior, 11 bits de los 16 bits del Resumen de Chequeo se adicionan al final de la cadena de datos de la Lectura Posterior. El Resumen de Chequeo es calculado usando el polinomio CCITT CRC – 16, como se muestra en la Figura 3.48. El Resumen de Chequeo consiste en los 11 bits más significativos del código de 16 bits. Un cambio en el Resumen de Chequeo indica un cambio en la cadena de bits de Lectura Posterior. Una comparación a un Resumen de Chequeo anterior sólo es significativa si los datos de lectura posterior son independientes del estado actual del dispositivo. Las salidas del CLB no deberán incluirse (la opción Capturar Lectura de MakeBits no se usa), y si la RAM está presente, el contenido de la RAM no debe estar alterado.

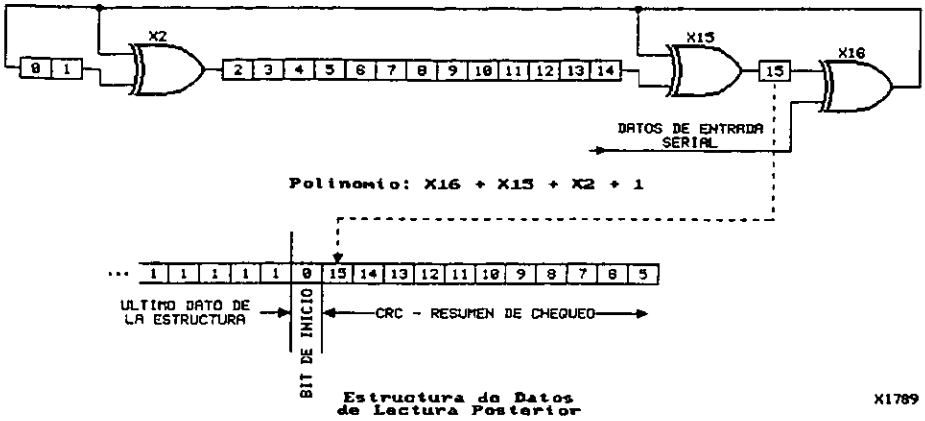


Figura 3.48: Circuito para Generación CRC – 16.

Estadísticamente, un error fuera de 2048 podría no ser detectado.

3.7.6 Secuencia de Configuración.

Hay cuatro pasos importantes en la secuencia de configuración de la Serie XC4000.

- Configuración de Inicialización de Memoria.
- Inicialización.
- Configuración.
- Puesta en Marcha.

El proceso completo se ilustra en la Figura 3.49.

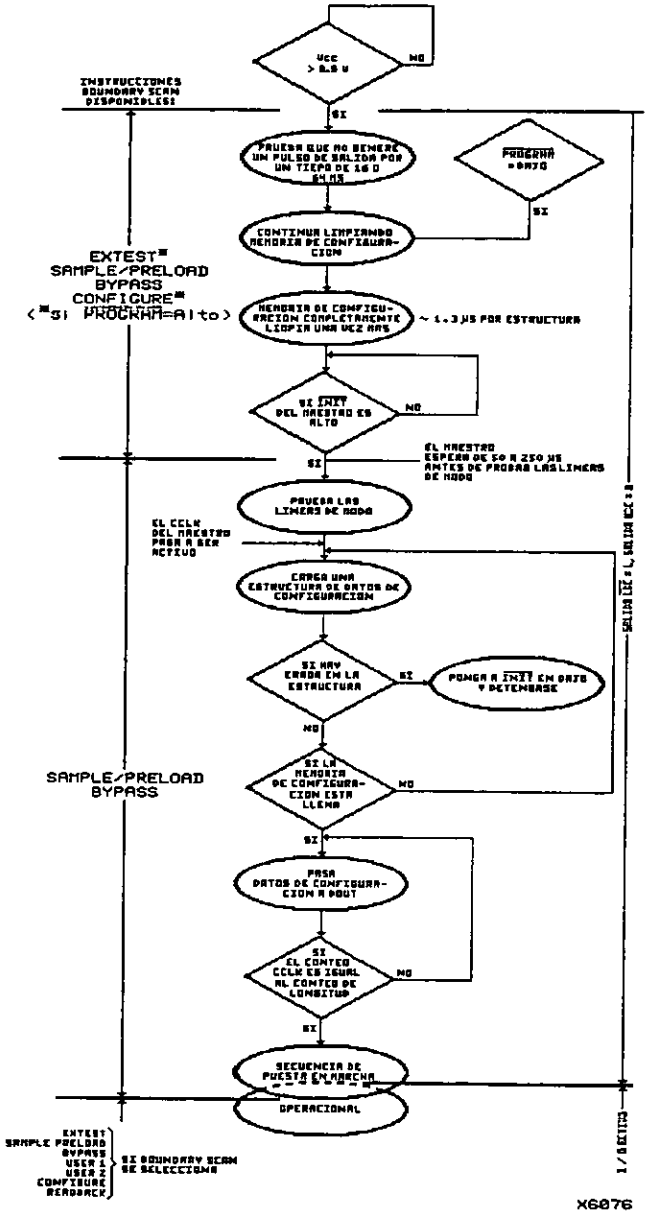


Figura 3.49: Secuencia de Configuración para la Aplicación de Potencia.

a) Configuración de Inicialización de Memoria.

Cuando se aplica potencia por primera vez o es reaplicada a un FPGA, un circuito interno forza la inicialización de la lógica de configuración. Cuando el Vcc alcanza un nivel operacional, y el circuito pasa la prueba de lectura y escritura de un par de muestras de bits de configuración, se inicializa un retardo de tiempo. Este retardo de tiempo es nominalmente 16 ms., y hasta un 10% más largo en los dispositivos de bajo voltaje. El retardo es cuatro veces tan largo como en los modos Maestros (M0 bajo), para permitir el tiempo suficiente para que todos los esclavos alcancen un estado de Vcc estable. Cuando todos los pines $\overline{INIT}$ se ligan juntos, como se comentó, toma prioridad el retardo más largo. Por consiguiente, los dispositivos con retardos de tiempo diferentes pueden fácilmente mezclarse y conjuntarse en una cadena de margarita.

Este retardo está presente únicamente durante la aplicación de potencia. No se emplea cuando se reconfigura un FPGA pulsando el pin $\overline{PROGRAM}$ a un estado bajo. Durante este tiempo de retardo, o tan largo como la entrada $\overline{PROGRAM}$ sea válida, la lógica de configuración se mantiene en un estado de Configuración de Inicialización de Memoria. Las estructuras de configuración – memoria se inicializan consecutivamente y usan el oscilador interno.

Al final de cada paso completo a través de la estructura de direccionamiento, la circuitería de aplicación de potencia de retardo de tiempo y el nivel del pin $\overline{PROGRAM}$ son probados. Si ninguno es válido, la lógica comienza una limpieza adicional de las estructuras de configuración y entonces se realizan pruebas a la entrada $\overline{INIT}$.

b) Inicialización.

Durante la inicialización y configuración, los pines de usuario HDC, $\overline{LDC}$, $\overline{INIT}$ y DONE proporcionan salidas de estados para la interfaz del sistema. Las salidas $\overline{LDC}$, $\overline{INIT}$ y DONE se mantienen en estado bajo y HDC se mantiene en estado alto cuando comienza la aplicación inicial de potencia.

El pin $\overline{INIT}$ de colector abierto se libera después del último paso de inicialización a través de las estructuras de direcciones. Hay un retardo deliberado de 50 a 250 μs. (hasta un 10% más para los dispositivos de bajo voltaje) antes de que un dispositivo en modo Maestro reconozca un $\overline{INIT}$

inactivo. Dos relojes internos después de que el pin $\overline{INIT}$ es reconocido en estado alto, el FPGA prueba las tres líneas de modo para determinar el modo de configuración. Las líneas de interfaz apropiadas son puestas en modo activo y el preámbulo de configuración y los datos pueden cargarse.

c) Configuración.

El código de preámbulo 0010, incluido para todos los modos excepto para el modo Express, indica que los siguientes 24 bits representan el conteo de longitud. El conteo de longitud es el número total de relojes de configuración necesarios para cargar completamente los datos de configuración (Cuatro relojes de configuración adicionales son requeridos para completar el proceso de configuración, como se discute más adelante). Después de que el preámbulo y el conteo de longitud se han pasado a través de todos los dispositivos en la cadena de margarita, DOUT se mantiene en un estado alto para prevenir que la estructura de los bits de arranque alcancen cualquier dispositivo de la cadena de margarita. En modo Express, los bits de conteo de longitud son ignorados, y DOUT se mantiene en un estado bajo, para deshabilitar al siguiente dispositivo en la pseudo cadena de margarita.

Un bit de configuración específico, anticipado en la primer estructura de un dispositivo maestro, controla la velocidad del reloj de configuración y puede aumentarla por un factor de ocho. Por consiguiente, si un reloj de configuración rápido es seleccionado por la cadena de bits, la velocidad del reloj más lenta se usa hasta que este bit de configuración se detecta.

Cada estructura tiene un campo de inicio seguido por la estructura de los bits de datos configuración y una estructura de campo de error. Si una estructura de error de datos se detecta, el FPGA detiene el proceso de carga, y señaliza el error llevando al pin $\overline{INIT}$ de colector abierto a un estado bajo. Después de que todas las estructuras de configuración se han cargado en un FPGA, DOUT nuevamente sigue los datos de entrada para que los datos restantes se pasen al siguiente dispositivo. En modo Express, cuando el primer dispositivo se programa totalmente, DOUT va a un estado alto para habilitar al siguiente dispositivo en la cadena.

d) Configuración de Retardo Después de la Aplicación de Potencia.

Existen dos métodos de retardar la configuración después de la aplicación de potencia: poner un estado lógico bajo en la entrada $\overline{\text{PROGRAM}}$, o conectar el pin $\overline{\text{INIT}}$ bidireccional a un estado bajo, usando un manejador de colector abierto (drenaje abierto). Vea la Figura 3.49.

Un estado bajo en la entrada $\overline{\text{PROGRAM}}$ es el acercamiento más radical, y se recomienda cuando el tiempo de ascenso del suministro de potencia es excesivo o pobremente definido. Mientras $\overline{\text{PROGRAM}}$ permanezca en un estado bajo, el FPGA sigue limpiando su memoria de configuración. Cuando $\overline{\text{PROGRAM}}$ llega a un estado alto, la memoria de configuración se limpia una vez más, seguido por el inicio de la configuración, mientras la entrada $\overline{\text{INIT}}$ no sea externamente mantenida en un estado bajo. Se debe notar que un estado bajo en la entrada $\overline{\text{PROGRAM}}$ automáticamente forza un estado bajo en la salida $\overline{\text{INIT}}$. El pin $\overline{\text{PROGRAM}}$ de la Serie XC4000 tiene un "Pull – up" pequeño permanentemente.

Usando un manejador de colector abierto o de drenaje abierto para mantener $\overline{\text{INIT}}$ en un estado bajo antes del inicio de la configuración ocasiona que el FPGA tenga que esperar después de completar la operación de limpieza de la memoria de configuración. Cuando $\overline{\text{INIT}}$ ya no se mantiene más tiempo en un estado bajo externamente, el dispositivo determina su modo de configuración capturando sus pines de modo, y está listo para iniciar el proceso de configuración. Un dispositivo maestro espera hasta unos 250 µs. adicionales para asegurarse de que cualquier esclavo en la cadena de margarita opcional ha visto que $\overline{\text{INIT}}$ esté en alto.

e) Puesta en Marcha.

La puesta en marcha es la transición desde el proceso de configuración a la operación de usuario propuesta. Esta transición involucra un cambio de un suministro de reloj a otro, y un cambio desde la interfaz paralela o serial de los datos de configuración donde la mayoría de las salidas están en tercer estado, a la operación normal con pines I / O activos en el sistema del usuario. La puesta en marcha debe asegurar que la lógica de usuario "levante" favorablemente, de manera que las salidas lleguen a ser activas sin que causen contratiempos con las señales de configuración, y que los flip – flops internos sean liberados del "Set o Reset Global" en el momento correcto.

La Figura 3.50 describe en detalle la temporización de puesta en marcha para las tres familias de Xilinx. La configuración de modo Express siempre usa la temporización CCLK_SYNC o UCLK_SYNC, los otros modos de configuración pueden usar cualquiera de las cuatro secuencias de temporización.

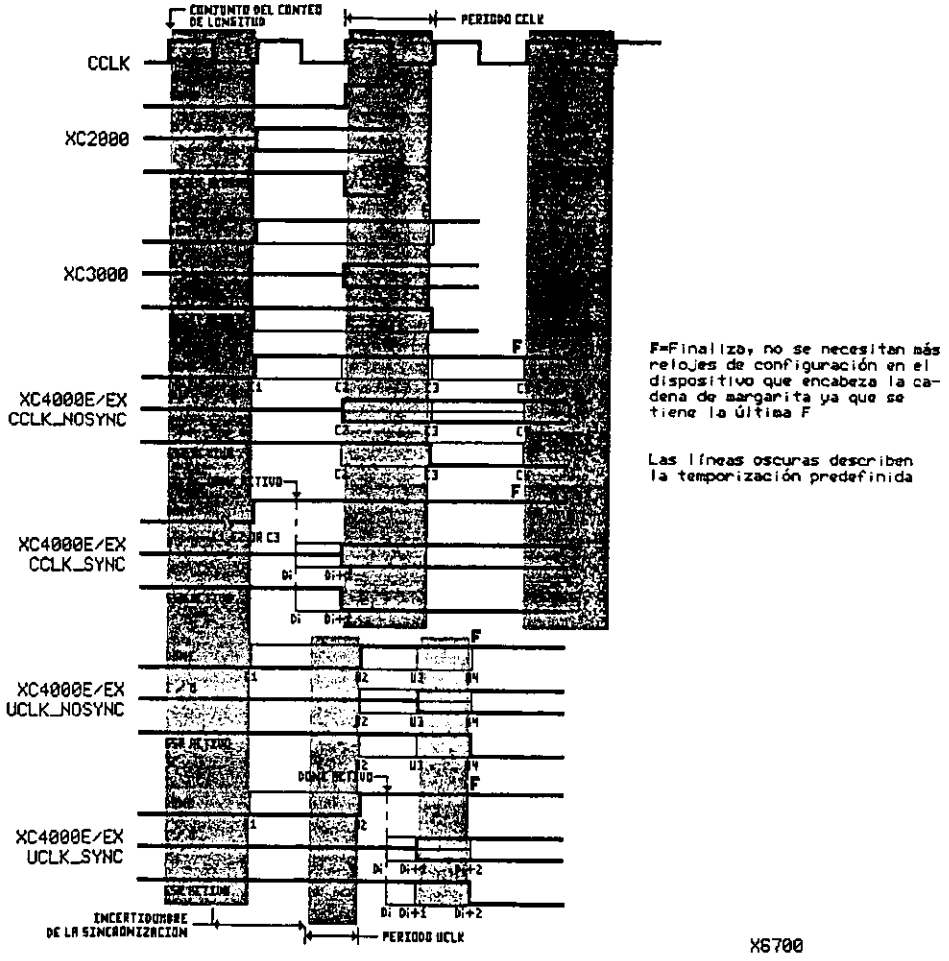


Figura 3.50: Temporización de Puesta en Marcha.

Para accesar a las señales de puesta en marcha internas, ponga el símbolo de biblioteca STARTUP.

Temporización de Puesta en Marcha.

Las diferentes familias de FPGA's tienen diferentes secuencias de puesta en marcha.

La familia XC2000 pasa por una secuencia fija. DONE pasa a un estado alto y el "Reset" global interno es desactivado en un período CCLK después de que los I / O lleguen a ser activos.

La familia XC3000A ofrece un poco de flexibilidad. DONE puede programarse para pasar a un estado alto en un período de CCLK antes o después de que los I / O lleguen a ser activos. Independiente de DONE, el "Reset" global interno es desactivado un período CCLK antes o después de que los I / O lleguen a ser activos.

La Serie XC4000 ofrece flexibilidad adicional. Los tres eventos, que DONE pase a alto, que el "Set / Reset" interno sea desactivado, y que los I / O de usuario lleguen a ser activos, pueden todos ocurrir en cualquier secuencia arbitraria. Cada uno de ellos puede ocurrir un período CCLK antes o después, o simultáneo con, cualquiera de los otros. Esta temporización relativa se selecciona por medio de las opciones de software en "MakeBits", y por el software de generación de la cadena de bits.

La opción predefinida, y una de las más prácticas, es que DONE pase a un estado alto primero, desconectando el suministro de datos de configuración y evitando cualquier contratiempo cuando los I / O's lleguen a ser activos un reloj después. Entonces, "Reset / Set" es liberado en otro período de reloj más tarde, para asegurar que la operación del usuario comience a partir de las condiciones internas estables. Esta es la secuencia más común, mostrada con líneas gruesas en la Figura 3.50, pero el diseñador puede modificarla para satisfacer necesidades particulares.

Normalmente, la secuencia de puesta en marcha es controlada por la salida de oscilador del dispositivo interno (CCLK), el cual es asíncrono al reloj del sistema.

La Serie XC4000 ofrece otra opción de temporización de puesta en marcha, UCLK_NOSYNC. Los tres eventos descritos anteriormente no necesitan activarse por CCLK. Ellos pueden, como una opción de configuración, ser activados por un reloj de usuario. Esto significa que el dispositivo puede establecerse en sincronismo con el sistema del usuario.

Cuando la opción de UCLK_SYNC se habilita, el usuario puede externamente mantener la salida DONE de colector abierto en bajo, y así mantener todo el progreso adicional en la secuencia de puesta en marcha hasta que DONE sea liberado y llegue a un estado alto.

Esta opción puede usarse para forzar la sincronización de varios FPGA's a un reloj común de usuario, o para garantizar que todos los dispositivos se configuren exitosamente antes de que cualesquiera de sus I / O's lleguen a ser activos.

Si cualquiera de estas dos opciones se selecciona, y ningún reloj de usuario se especifica en el diseño o se liga al dispositivo, el chip podría alcanzar un punto donde la configuración del dispositivo esté completa y el pin DONE sea válido, pero las salidas no lleguen a ser activas. La solución es recrear la cadena de bits especificando el reloj de puesta en marcha como CCLK, o proporcionando el reloj de usuario apropiado.

Secuencia de Puesta en Marcha.

La secuencia de puesta en marcha empieza cuando la memoria de configuración está llena, y el número total de relojes de configuración recibidos en $\overline{INIT}$ cuando está en alto, fueron iguales para el valor cargado del conteo de longitud.

Los siguientes flancos de reloj ascendente son colocados en un flip – flop Q0, como se muestra en la Figura 3.51. Q0 es el bit principal de un registro de corrimiento de 5 bits.

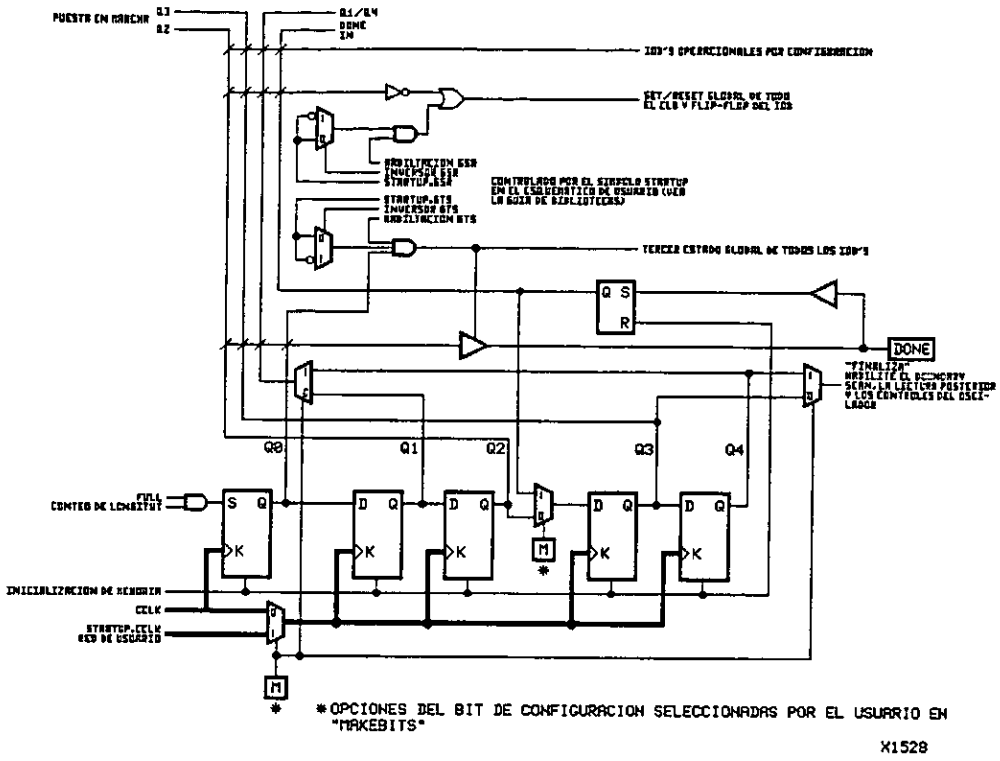


Figura 3.51: Lógica de Puesta en Marcha.

Las salidas de este registro pueden programarse para controlar tres eventos.

- La liberación de la salida DONE de colector abierto.
- El cambio de los pines relacionados con la configuración a la función de usuario, activando todos los IOB's.
- La terminación de la inicialización "Set / Reset" global de todos los CLB's y elementos de almacenamiento del IOB.

El pin DONE puede también ser AND – alambrado con los pines DONE de otros FPGA's o con otras señales externas, y puede entonces ser usado como entrada para el bit Q3 del registro de puesta en marcha. Esto se llama "Temporización de Puesta en Marcha Síncrona para DONE In" y es seleccionado por las opciones de "MakeBits" CCLK_SYNC y UCLK_SYNC.

Cuando DONE no se usa como una entrada, la operación es llamada "Temporización de Puesta en Marcha No Sincrona para DONE In", y es seleccionado por las opciones de "MakeBits" CCLK_NOSYNC y UCLK_NOSYNC.

Como una opción de configuración, el registro de control de puesta en marcha a través de Q0 puede ser temporizado ya sea por pulsos CCLK subsecuentes o desde una red de usuario sobre el chip llamada STARTUP.CLK. Estas señales pueden ser accesadas poniendo el símbolo de biblioteca STARTUP.

Puesta en Marcha desde CCLK.

Si CCLK es usado para manejar la puesta en marcha, Q0 hasta Q3 proporcionan la temporización. Las líneas gruesas en la Figura 3.50 muestran la temporización predefinida, que es compatible con los dipositivos XC2000 y XC3000 que usan "DONE" anticipado y "Reset" retardado. Las líneas delgadas indican todas las otras posibles opciones de temporización.

Puesta en Marcha desde un Reloj de Usuario (STARTUP.CLK).

Cuando, en lugar de CCLK, se selecciona un reloj de puesta en marcha proporcionado por el usuario, Q1 se usa como puente de la relación de fase desconocida entre CCLK y el reloj de usuario. Este arbitraje provoca una inevitable incertidumbre del único ciclo en la temporización del resto de la secuencia de puesta en marcha.

f) DONE Va a Alto para Señalizar la Finalización de Configuración.

En todos los modos de configuración excepto para el modo Express, los dispositivos de la Serie XC4000 leen el conteo de longitud esperado desde la cadena de bits y lo almacenan en un registro interno. El conteo de longitud varía según el número de dispositivos y la composición de la cadena de margarita. Cada dispositivo también cuenta el número de CCLK's durante la configuración.

Dos condiciones tienen que reunirse en el siguiente orden para que el pin DONE pase a un estado alto:

- la memoria interna de los chip's debe estar llena, y
- el conteo de longitud de configuración debe ser exacto.

Esto es importante porque el contador que determina cuando el conteo de longitud se ha completado, comienza con el primer CCLK, no con el primero después del preámbulo.

Por consiguiente, si un bit aislado se inserta antes del preámbulo, o el suministro de datos no está listo en el momento del primer CCLK, el contador interno que retiene el número de CCLK's irá uno adelante del número real de los bits de datos leídos. Al final de la configuración, la memoria de configuración estará llena, pero el número de bits en el contador interno no igualará el conteo de longitud esperado.

Como consecuencia, un dispositivo en modo Maestro continuará mandando CCLK's hasta que el contador interno vuelva a ponerse en cero, y entonces alcance el conteo de longitud correcto en una segunda ocasión. Esto tomará varios segundos [2^{24} * periodo CCLK] que a veces se interpreta como que el dispositivo no se configuro del todo.

Si no es posible tener los datos preparados en el momento del primer CCLK, el problema puede evitarse aumentando el número en el conteo de longitud por el valor apropiado.

En modo Express, no hay ningún conteo de longitud. El pin DONE para cada dispositivo llega a un estado alto cuando el dispositivo ha recibido su cuota de datos de configuración. Alambrando los pines DONE de varios dispositivos a la vez se tienen retardos de puesta en marcha en todos los dispositivos hasta que todos se configuran totalmente.

Es importante notar que DONE es una salida de colector abierto y no pasa a un estado alto a menos que un "pull – up" interno sea activado o un "pull – up" externo sea ligado. El "pull – up" interno es activado como valor por omisión por "MakeBits", que es el software de generación de cadena de bits.

g) Liberación de los I / O de Usuario Después de que DONE Pasa a un Estado Alto.

Por predefinición, los I / O de usuario se liberan un ciclo CCLK después de que el pin DONE pasa a un estado alto. Si CCLK no es temporizado después de que DONE pasa a un estado alto, las

salidas permanecen en su estado inicial, tercer estado, con un "pull – up" de 50 KΩ – 100 KΩ. El retardo desde que DONE es alto hasta que los I / O de usuario son activos es controlado por una opción de "MakeBits".

h) Liberación de "Set / Reset Global" Después de que DONE Pasa a un Estado Alto.

Por predefinición, el "Set / Reset Global" (GSR) se libera dos ciclos CCLK después de que el pin DONE pasa a un estado alto. Si CCLK no es temporizado dos veces después de que DONE pasa a un estado alto, todos los flip – flops se mantienen en su estado inicial "Set" o "Reset". El retardo desde que DONE es alto hasta que GSR sea inactivo es controlado por una opción de "MakeBits".

i) Configuración Completa Después de que DONE Pasa a un Estado Alto.

Tres ciclos CCLK completos son requeridos después de que el pin DONE pasa a un estado alto, como se muestra en la Figura 3.50. Si CCLK no es temporizado tres veces después de que DONE pasa a un estado alto, no puede iniciarse la lectura posterior y la mayoría de las instrucciones "Boundary – Scan" no pueden usarse.

3.7.7 Configuración a Través de los Pines "Boundary Scan".

Los dispositivos de la Serie XC4000 pueden configurarse a través de los pines "Boundary – Scan". El procedimiento básico es como se indica a continuación:

- Aplicar potencia al FPGA con $\overline{\text{INIT}}$ permanente en bajo (o manejar el pin $\overline{\text{PROGRAM}}$ en bajo por más de 300 ns. seguido por un estado alto mientras se mantiene a $\overline{\text{INIT}}$ en bajo). Con $\overline{\text{INIT}}$ en bajo se proporciona el tiempo suficiente para emitir el comando CONFIG al FPGA. El pin puede usarse como I / O después de la configuración si una resistencia se usa para mantener a $\overline{\text{INIT}}$ en un estado bajo.
- Emitir el comando CONFIG a la entrada TMS.
- Esperar a que $\overline{\text{INIT}}$ pase a un estado alto.

- Dar la secuencia de Puerto de Acceso de Prueba "Boundary – Scan" para el estado SHIFT – DR.
- Cambiar TCK para los datos de reloj en el pin TDI.

El usuario debe considerar todos los ciclos de reloj TCK después de que $\overline{INIT}$ pasa a un estado alto, porque todos estos ciclos afectan la comparación del conteo de longitud.

3.8 LECTURA POSTERIOR (READ BACK).

El usuario puede aplicar lectura posterior al contenido de la memoria de configuración y al nivel de ciertos nodos internos sin interferir con la operación normal del dispositivo.

La lectura posterior no sólo informa los bits de configuración bajados, sino también puede incluir el estado presente del dispositivo, representado por el contenido de todos los flip – flops y latches en los CLB's e IOB's, así como también el contenido de los generadores de función usados como RAM's.

Hay que observar que en los dispositivos de la Serie XC4000, los datos de configuración no son invertidos con respecto a la configuración como lo es en las familias XC2000 y XC3000.

Las cadenas de bits de lectura posterior en modo Express resultan en datos que no se parecen a la cadena de bits original, porque el formato de cadena de bits difiere de los otros modos.

La Lectura Posterior de la Serie XC4000 no usa ningún pin dedicado, pero usa cuatro redes internas (RDBK.TRIG, RDBK.DATA, RDBK.RIP y RDBK.CLK) que pueden enrutarse a cualquier IOB. Para accesar a las señales de Lectura Posterior internas, se coloca el símbolo de biblioteca READBACK y se ligan los símbolos "pad" apropiados, como se muestra en la Figura 3.52.

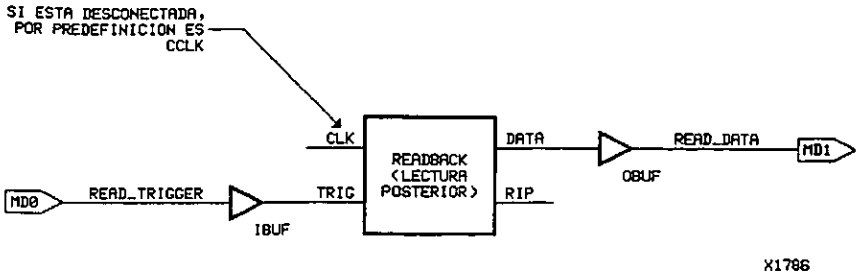


Figura 3.52: Ejemplo Esquemático de Lectura Posterior.

Después de que la Lectura Posterior ha sido iniciada por una transición de estado bajo a alto en RDBK.TRIG, la salida RDBK.RIP (Lectura en Progreso) pasa a un estado alto en el siguiente flanco ascendente de RDBK.CLK. Los flancos ascendentes subsecuentes de este reloj desplazan externamente los datos de Lectura Posterior en la red RDBK.DATA.

Los datos de Lectura Posterior no incluyen al preámbulo, pero inicia con cinco bits falsos (todos en alto) seguidos por el bit de arranque (en bajo) de la primera estructura. Los primeros dos bits de datos de la primera estructura siempre están en alto.

Cada estructura acaba con cuatro bits de chequeo de error. Ellos se leen posteriormente en un estado alto. Los últimos siete bits de la última estructura también se leen posteriormente en un estado alto. Un bit de arranque adicional (en bajo) y un Chequeo de Redundancia Cíclica de 11 bits (CRC) siguen esta característica, antes de que RDBK.RIP retorne a un estado bajo.

3.8.1 Opciones de Lectura Posterior.

Las opciones de Lectura Posterior son: Capturar Lectura, Abortar Lectura, y Seleccionar Reloj. Ellas son establecidas con "MakeBits", que es el software de generación de cadena de bits.

a) Capturar Lectura.

Cuando la opción de Capturar Lectura se selecciona, la cadena de datos de lectura posterior incluye valores probados de las señales del CLB e IOB. El flanco ascendente de los latches RDBK.TRIG invierte los valores de las cuatro salidas del CLB, las salidas de los flip – flops del IOB y las señales de entrada I1 e I2. Es importante resaltar que mientras los bits que describen a la configuración (interconexión, generadores de función, y contenido RAM) no son invertidos, las señales de salida del CLB e IOB si se invierten.

Cuando la opción Capturar Lectura no es seleccionada, los valores de los bits de captura reflejan los datos de configuración originalmente escritos en esas localidades de memoria.

En caso de que la capacidad RAM de los CLB's sea usada, los datos RAM están disponibles en lectura posterior, ya que ellos sobreescriben directamente la configuración de la tabla de función F y G del CLB.

RDBK.TRIG se localiza en la esquina inferior izquierda del dispositivo, como se muestra en la Figura 3.53.

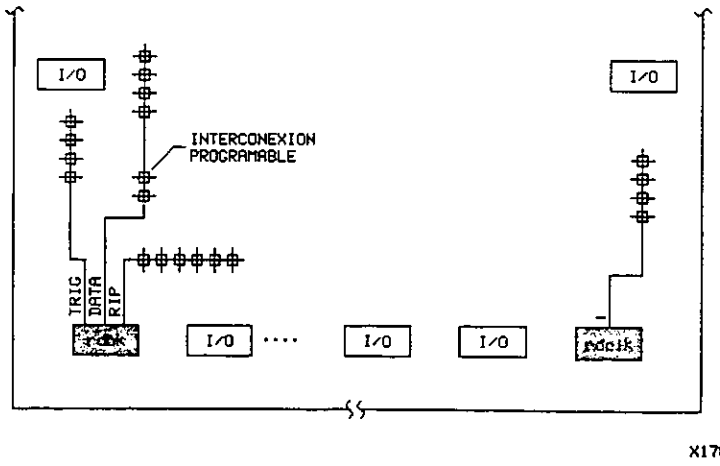


Figura 3.53: Símbolo "READBACK" (LECTURA POSTERIOR) en Editor Gráfico.

b) Abortar Lectura.

Cuando la opción Abortar Lectura es seleccionada, una transición de estado alto a bajo en RDBK.TRIG termina la operación de lectura posterior y prepara a la lógica para aceptar otra activación.

Después de que una lectura posterior es abortada, los relojes adicionales (hasta un reloj de lectura posterior por estructura de configuración) pueden requerirse para reinicializar a la lógica de control. El estado de lectura posterior es indicado por la red de control de salida RDBK.RIP. RDBK.RIP está en estado alto siempre que una lectura posterior esté en progreso.

c) Seleccionar Reloj.

CCLK es el reloj predefinido. Sin embargo, el usuario puede insertar otro reloj en RDBK.CLK. El control de lectura posterior y los datos son temporizados en los flancos ascendentes de RDBK.CLK. En caso de que la lectura posterior deba cancelarse por razones de seguridad, las redes de control de lectura posterior simplemente no se conectan.

RDBK.CLK se localiza en la esquina inferior derecha del chip, como se muestra en la Figura 3.53.

3.8.2 Violando la Especificación Máxima de Tiempo en Estado Alto y Bajo para el Reloj de Lectura Posterior.

El reloj de lectura posterior tiene una máxima especificación de tiempo en estado alto y bajo. En algunos casos, esta especificación no puede alcanzarse. Por ejemplo, si un procesador está controlando el proceso de lectura posterior, una interrupción puede obligarle a que se detenga a la mitad de este proceso de lectura posterior. Esto hace necesario detener el reloj, y por lo tanto se viola la especificación.

La especificación sólo es obligatoria en los datos de temporización al final de una estructura anterior al próximo bit de arranque. El mecanismo de transferencia cargará los datos a un registro de corrimiento durante los últimos seis ciclos de reloj de la estructura, anterior al bit de arranque de la siguiente estructura. Este proceso de carga es dinámico, y es la fuente de las máximas necesidades tiempo en estado alto y bajo.

Por consiguiente, la especificación sólo se aplica a los seis ciclos de reloj anteriores e incluso a cualquier bit de arranque, incluyendo a los relojes antes del primer bit de arranque en las cadenas de datos de lectura posterior. En otras ocasiones, los datos de la estructura ya están en el registro y el registro no es dinámico. Por lo tanto, este puede desplazar la salida como un registro de corrimiento regular.

El usuario debe calcular precisamente la posición de los datos de lectura posterior con respecto a la estructura. El sistema debe guardar rastro de la posición dentro de una estructura de datos, y deshabilitar las interrupciones antes de la estructura del "Boundary – Scan". Las longitudes de la estructura y los formatos de datos se listan en la Tabla 3.19, 3.20 y 3.21.

3.8.3 Lectura Posterior con el Cable XChecker.

El cable "Download / Read – Back" (Transmisión / Lectura Posterior) Universal XChecker y la Sonda Lógica usan la característica de lectura posterior para la comprobación de la cadena de bits. Puede también mostrar señales internas seleccionadas en la PC o en la pantalla de una "Workstation", funcionando como un emulador de circuito de bajo costo.

3.9 CONFIGURACION DE LA TEMPORIZACION.

Los siete modos de configuración se discuten detalladamente en esta sección. Las especificaciones de temporización son incluidas.

3.9.1 Modo Serial Maestro.

En el modo Serial Maestro, la salida CCLK del FPGA principal direcciona a una PROM Serial de Xilinx que alimenta a la entrada DIN del FPGA. Cada flanco ascendente de la salida CCLK incrementa el contador de dirección interno de la PROM Serial. El siguiente bit de datos se coloca en la salida de datos de la SPROM, conectado al pin DIN del FPGA. El FPGA principal recibe estos datos en el subsecuente flanco CCLK ascendente.

El FPGA principal entonces presenta los datos de preámbulo, y todos los datos que desbordan del dispositivo principal, lo hacen en su pin DOUT. Existe un retardo "Pipeline" interno de 1.5 periodos

de CCLK, que significa que DOUT cambia en el flanco CCLK descendente, y que el siguiente FPGA en la cadena de margarita acepta datos en el siguiente flanco CCLK ascendente.

En "MakeBits", el usuario puede especificar "Fast ConfigRate", el cual, empezando con varios bits en la primera estructura, aumenta la frecuencia de CCLK por un factor de ocho. El valor aumenta desde entre 0.5 y 1.25 MHz., a un valor entre 4 y 10 MHz. (para dispositivos de bajo voltaje, la frecuencia puede ser hasta un 10% menor). Esto asegura que la PROM serial y los esclavos son lo suficientemente rápidos para soportar esta velocidad de datos. Los dispositivos XC2000, XC3000 / A, y los XC3100A no soportan la opción de "Fast ConfigRate".

La entrada CE de la SPROM puede manejarse ya sea desde $\overline{LDC}$ o DONE. Usando $\overline{LDC}$ se evitan contratiempos potenciales en el pin DIN, si este pin es configurado como I / O de usuario, pero $\overline{LDC}$ se restringe entonces a ser una salida de usuario permanentemente en estado alto después de la configuración. Usando DONE también se pueden evitar contratiempos en DIN, con tal de que la opción "DONE" anticipada sea invocada.

El modo Serial Maestro es seleccionado por un <000> en los pines de modo (M2, M1, M0).

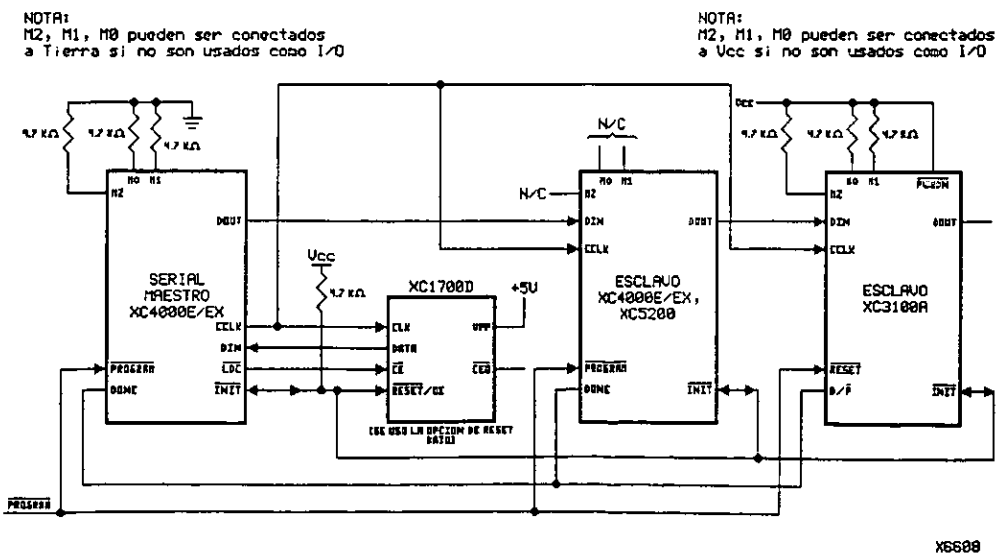
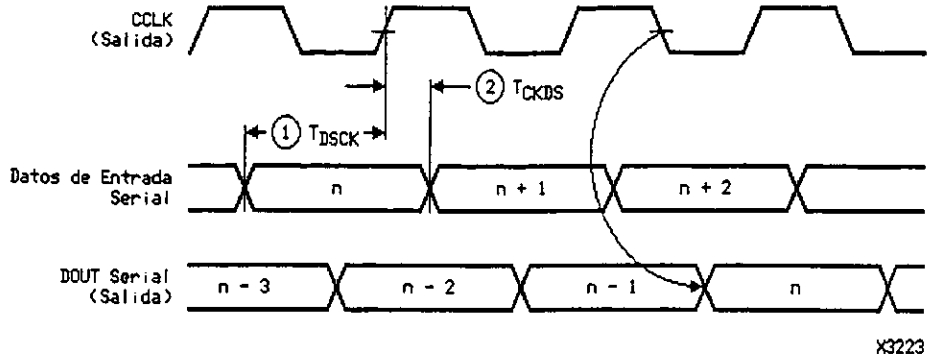


Figura 3.54: Diagrama de Circuito del Modo Serial Maestro.



	Descripción.	Símbolo.	Mínimo.	Máximo.	Unidades.
CCLK	"Setup" de DIN.	1 T_{DSCK}	20		ns.
	Retención de DIN.	2 T_{CKDS}	0		ns.

Notas: 1. Al momento de la aplicación de potencia, Vcc debe de subir desde 2 V. hasta el Vcc mínimo en un tiempo menor a 25 ms, por otra parte la configuración del retardo conectando $\overline{\text{PROGRAM}}$ desde un estado bajo hasta Vcc es válida.

2. La temporización del modo Serial Maestro se basa en pruebas realizadas en el modo esclavo.

Figura 3.55: Características de Conmutación Durante la Programación en Modo Serial Maestro.

3.9.2 Modo Serial Esclavo.

En el modo Serial Esclavo, una señal externa maneja la entrada CCLK del FPGA. La cadena de bits de la configuración serial debe estar disponible en la entrada DIN del FPGA principal en un tiempo de establecimiento corto antes de cada flanco CCLK ascendente.

El FPGA principal entonces presenta los datos de preámbulo, y todos los datos que desbordan del dispositivo principal, lo hacen en su pin DOUT. Existe un retardo interno de 0.5 períodos de CCLK, lo cual significa que DOUT cambia en el flanco CCLK descendente, y que el siguiente FPGA en la cadena de margarita acepta datos en el siguiente flanco CCLK ascendente.

El modo Serial Esclavo es seleccionado por un <111> en los pines de modo (M2, M1, M0). Serial Esclavo es el modo predefinido si los pines de modo quedan sin conectar, ya que ellos tienen resistores "pull – up" pequeños durante la configuración.

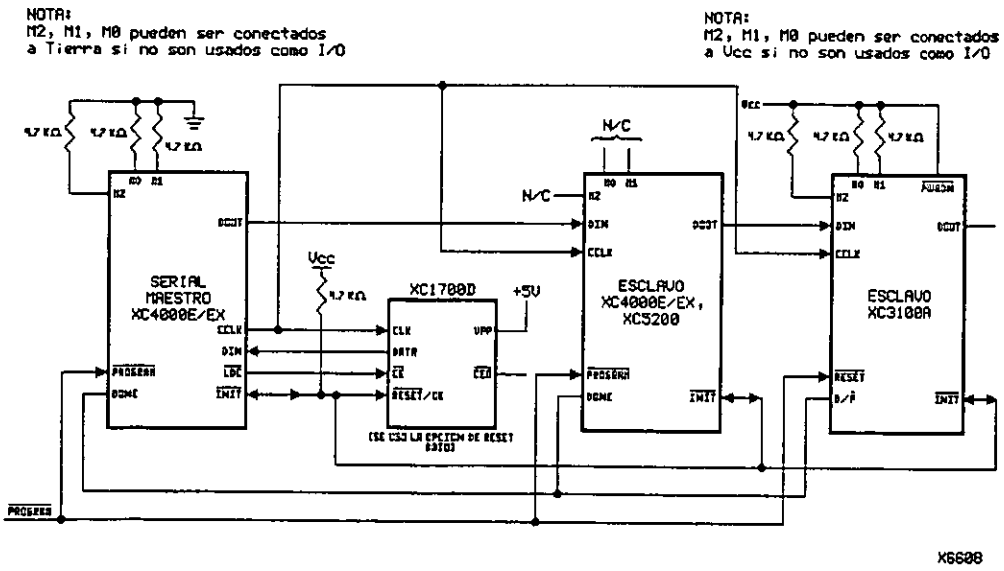
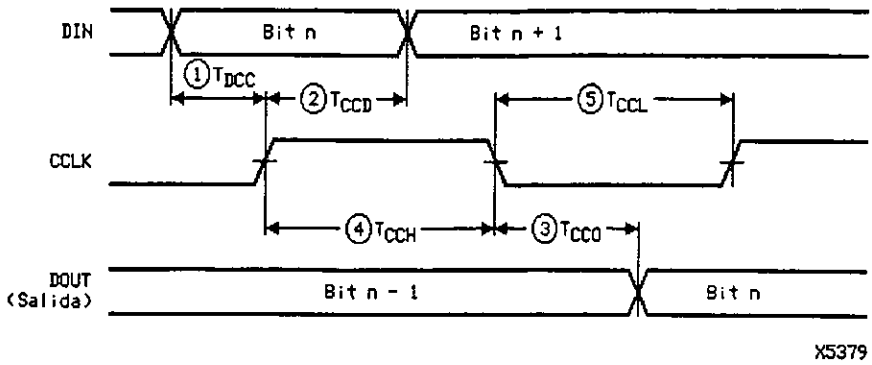


Figura 3.56: Diagrama de Circuito del Modo Serial Esclavo.



	Descripción.	Símbolo.	Mínimo.	Máximo.	Unidades.
CCLK	"Setup" de DIN.	1 T_{DCC}	20		ns.
	Retención de DIN.	2 T_{CCD}	0		ns.
	DIN para DOUT.	3 T_{CCO}		30	ns.
	Tiempo alto.	4 T_{CCH}	45		ns.
	Tiempo bajo.	5 T_{CCL}	45		ns.
	Frecuencia.	F_{CC}		10	MHz.

Nota: La configuración debe ser retardada hasta que los pines $\overline{INIT}$ de todos los FPGA's de la cadena de margarita estén en estado alto.

Figura 3.57: Características de Conmutación Durante la Programación en Modo Serial Esclavo.

3.9.3 Modos Paralelos Maestros.

En los dos modos Paralelos Maestros, el FPGA principal directamente direcciona un byte ancho de aplicación estándar a la EPROM, y acepta ocho bits de datos justo antes de incrementar o decrementar las salidas de dirección.

Los ocho bits de datos son serializados en el FPGA principal, el cual entonces presenta los datos de preámbulo, y todos los datos que salen del dispositivo principal, lo hacen en su pin DOUT. Existe un retardo interno de 1.5 periodos de CCLK, después de que el flanco CCLK ascendente que acepta un byte de datos (y también los cambios de dirección EPROM) hasta el flanco CCLK descendente que hace que el LSB (D0) de este byte aparezca en DOUT. Esto significa que DOUT cambia en el flanco CCLK descendente, y que el siguiente FPGA en la cadena de margarita acepta datos en el siguiente flanco CCLK ascendente.

Los pines de dirección PROM pueden incrementarse o decrementarse, dependiendo del establecimiento del pin de modo. Esta opción le permite al FPGA compartir la PROM con una amplia variedad de microprocesadores y microcontroladores. Algunos procesadores deben inicializarse desde la parte inferior de la memoria (todos ceros) mientras otros deben inicializarse desde la parte superior. El FPGA es flexible y puede cargar su cadena de bits de configuración desde cualquier extremo de la memoria.

El modo Paralelo Maestro Alto es seleccionado por un <100> en los pines de modo (M2, M1, M0). La EPROM es direccionada para arrancar en 00000 e incrementarse.

El modo Paralelo Maestro Bajo es seleccionado por un <110> en los pines de modo (M2, M1, M0). La EPROM es direccionada para arrancar en 3FFFF y decrementarse.

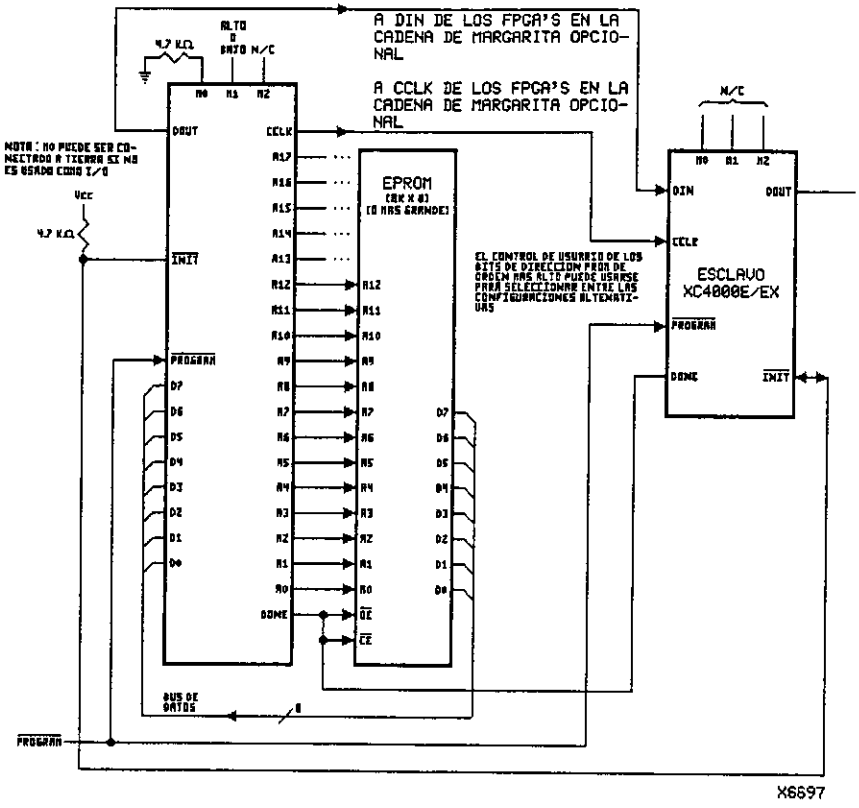
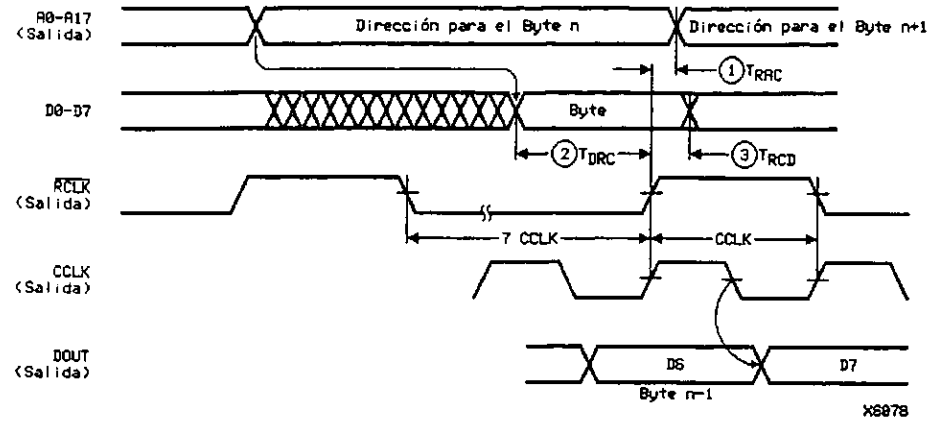


Figura 3.58: Diagrama de Circuito del Modo Paralelo Maestro.



	Descripción.	Símbolo.	Mínimo.	Máximo.	Unidades.
CCLK	Retardo para Dirección válida.	1 T_{RAC}	0	200	ns.
	Tiempo de establecimiento de los Datos.	2 T_{DRC}	60		ns.
	Tiempo de retención de los Datos.	3 T_{RCD}	0		ns.

Notas: 1. Al momento de la aplicación de potencia, Vcc debe de subir desde 2 V. hasta el Vcc mínimo en un tiempo menor a 25 ms, por otra parte la configuración del retardo conectando $\overline{\text{PROGRAM}}$ desde un estado bajo hasta Vcc es válida.

2. El primer byte de Datos es cargado y entonces arranca CCLK al final de los primeros ciclos activos $\overline{\text{RCLK}}$ (Flanco Ascendente).

Este diagrama de temporización muestra que las necesidades de la EPROM están sumamente reducidas. El tiempo de acceso a la EPROM puede ser mucho mayor a 500 ns. La salida de datos de la EPROM no tiene necesidades de tiempo de retención.

Figura 3.59: Características de Conmutación Durante la Programación en Modo Paralelo Maestro.

3.9.4 Modo Periférico Síncrono.

El modo Periférico Síncrono puede ser considerado también como modo Paralelo Esclavo. Una señal externa maneja la entrada (s) de CCLK del FPGA (s). El primer byte de datos de configuración paralela debe estar disponible en las entradas de Datos del FPGA principal en un tiempo de establecimiento corto antes del flanco CCLK ascendente. Los bytes de datos subsecuentes son temporizados consecutivamente en cada flanco octavo CCLK ascendente.

El mismo flanco CCLK que acepta datos, también ocasiona que la salida RDY / $\overline{\text{BUSY}}$ pase a un estado alto por cada periodo CCLK. El nombre del pin es un nombre erróneo. En el modo Periférico Síncrono éste es realmente una señal "ACKNOWLEDGE" (RECONOCIMIENTO). La operación síncrona no requiere esta respuesta, pero si es una señal significativa para propósitos de prueba. Hay que observar que RDY / $\overline{\text{BUSY}}$ está conectado a un estado alto con un "pull – up" de alta impedancia previo a que $\overline{\text{INIT}}$ pase a un estado alto.

El FPGA principal serializa los datos y presenta los datos de preámbulo (y todos los datos que salen del dispositivo principal) en su pin DOUT. Existe un retardo interno de 1.5 periodos de CCLK, que significa que DOUT cambia en el flanco CCLK descendente, y que el siguiente FPGA en la cadena de margarita acepta datos en el siguiente flanco CCLK ascendente.

A fin de completar la operación de desplazamiento serial se requieren 10 flancos CCLK ascendentes adicionales después de que el último byte de datos ha sido cargado, más un ciclo CCLK adicional por cada dispositivo de la cadena de margarita.

El modo Periférico Síncrono es seleccionado por un <011> en los pines de modo (M2, M1, M0).

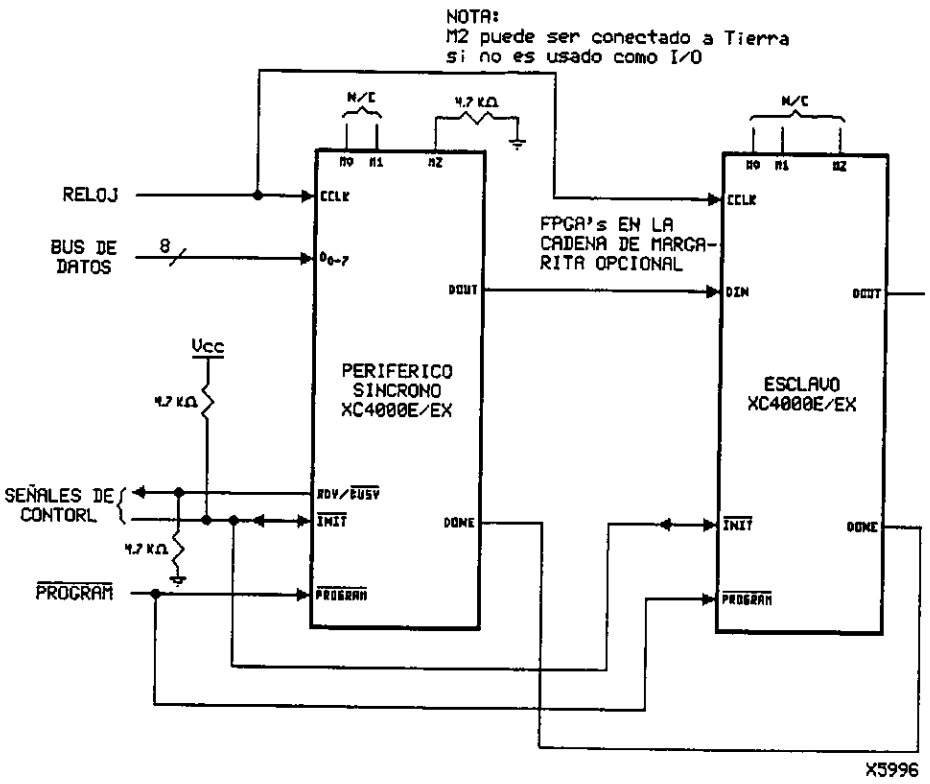
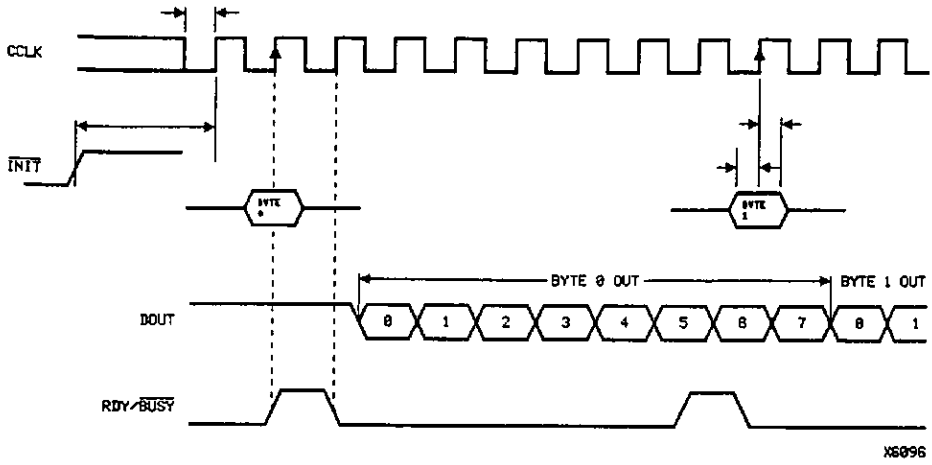


Figura 3.60: Diagrama de Circuito del Modo Periférico Síncrono.



	Descripción.	Símbolo.	Mínimo.	Máximo.	Unidades.
CCLK	Tiempo de establecimiento (Alto) $\overline{INIT}$.	T_{IC}	5		µs.
	Tiempo de establecimiento D0 – D7.	T_{DC}	60		ns.
	Tiempo de retención D0 – D7.	T_{CD}	0		ns.
	Tiempo alto CCLK.	T_{CCH}	50		ns.
	Tiempo bajo CCLK.	T_{CCL}	60		ns.
	Frecuencia CCLK.	F_{CC}		8	MHz.

Notas: 1. El modo Periférico Síncrono puede ser considerado modo Paralelo Esclavo. Un CCLK externo proporciona la temporización, temporizando en el primer byte de datos en el segundo flanco ascendente de CCLK después de que $\overline{INIT}$ pasa a un estado alto. Los bytes de datos subsecuentes son temporizados en cada flanco octavo ascendente de CCLK consecutivamente.

2. La línea RDY / $\overline{BUSY}$ pasa a un estado alto por cada período CCLK después de que los datos han sido temporizados internamente, aunque la operación síncrona no requiere tal respuesta.

3. El nombre del pin RDY / $\overline{BUSY}$ es un nombre erróneo. En el modo Periférico Síncrono esto es realmente una señal "ACKNOWLEDGE".

4. Observar que datos de arranque son serializados externamente en el pin DOUT 0.5 períodos de CCLK después de que fueron cargados en paralelo. Por consiguiente, pulsos de CCLK adicionales son requeridos después de que el último byte ha sido cargado.

Figura 3.61: Características de Conmutación Durante la Programación en Modo Periférico Síncrono.

2.9.5 Modo Periférico Asíncrono.

a) Escritura del FPGA.

El modo Periférico Asíncrono usa el flanco proveniente de la condición AND lógica de $\overline{WS}$ y $\overline{CS0}$ que son bajos y $\overline{RS}$ y CS1 que son altos para admitir datos de byte amplio desde un bus de microprocesador. En el FPGA principal, estos datos son cargados en una UART doble tipo buffer del mismo modo que en un convertidor paralelo a serie y se desplazan serialmente en la lógica interna.

El FPGA principal presenta los datos de preámbulo (y todos los datos que salen del dispositivo principal) en su pin DOUT. La salida RDY / $\overline{BUSY}$ proveniente del FPGA principal actúa como una señal de "acknowledge" (reconocimiento) para el microprocesador. RDY / $\overline{BUSY}$ pasa a un estado bajo cuando un byte se ha recibido, y pasa a un estado alto nuevamente cuando el buffer de entrada de byte ancho ha transferido su información al registro de corrimiento, y el buffer está listo para recibir nuevos datos. Una nueva escritura puede comenzarse inmediatamente, tan pronto como la salida RDY / $\overline{BUSY}$ pase a un estado bajo, reconociendo que ha recibido los datos previos. La escritura no puede terminarse hasta que RDY / $\overline{BUSY}$ esté nuevamente en estado alto por un período de CCLK. Hay que observar que RDY / $\overline{BUSY}$ es conectado a un estado alto con un "pull – up" de alta impedancia previo a que $\overline{INIT}$ pase a un estado alto.

La longitud de la señal $\overline{BUSY}$ depende de la actividad en la UART. En caso de que el registro de corrimiento hubiera estado vacío cuando el nuevo byte fue recibido, la última señal $\overline{BUSY}$ dura sólo dos períodos de CCLK. En caso de que el registro de corrimiento hubiera estado lleno cuando el nuevo byte fue recibido, la señal $\overline{BUSY}$ puede ser tan larga como nueve períodos de CCLK.

Después de que el último byte ha entrado, sólo siete de sus bits se desplazan externamente. CCLK permanece en estado alto con una longitud en DOUT igual a 6 bits (del siguiente bit al último) del último byte que ha entrado.

El reconocimiento READY / $\overline{BUSY}$ puede ignorarse en caso de que el retardo proveniente de cualquier Write al extremo del siguiente Write se garantice para ser tan largo como 10 períodos de CCLK.

b) Estado de Lectura.

La condición AND lógica de las entradas $\overline{CS0}$, CS1 y $\overline{RS}$ pone el estado del dispositivo en bus de Datos.

- D7 en estado alto indica Listo (Ready).
- D7 en estado bajo indica Ocupado (Busy).
- D0 a D6 pasa incondicionalmente a un estado alto.

Es obligatorio que la secuencia de puesta en marcha sea arrancada y finalizada por una entrada de byte ancho. Por otra parte, los pines usados como "Write Strobe" o "Chip Enable" (Habilitación del Chip) podrían volverse salidas activas y tal vez interfieran con el traslado del byte final. En caso de que este traslado no ocurra, la secuencia de puesta en marcha no es completada totalmente hasta su finalización (punto F en la Figura 3.50).

En este caso, el peor, el "reset" interno no es liberado. En el mejor, la Lectura Posterior y "Boundary – Scan" están inhibidas. El valor del conteo de longitud, generado por "MakeBits" y "MakePROM", asegura que estos problemas nunca ocurran.

Aunque RDY / $\overline{BUSY}$ sean llevados al exterior como una señal separada, los microprocesadores pueden leer con más facilidad esta información en una de las líneas de datos. Para este propósito, D7 representa el estado RDY / $\overline{BUSY}$ cuando $\overline{RS}$ está en estado bajo, $\overline{WS}$ está en estado alto, y las dos líneas de selección del chip están ambas activas.

El modo Periférico Asíncrono es seleccionado por un <101> en los pines de modo (M2, M1, M0).

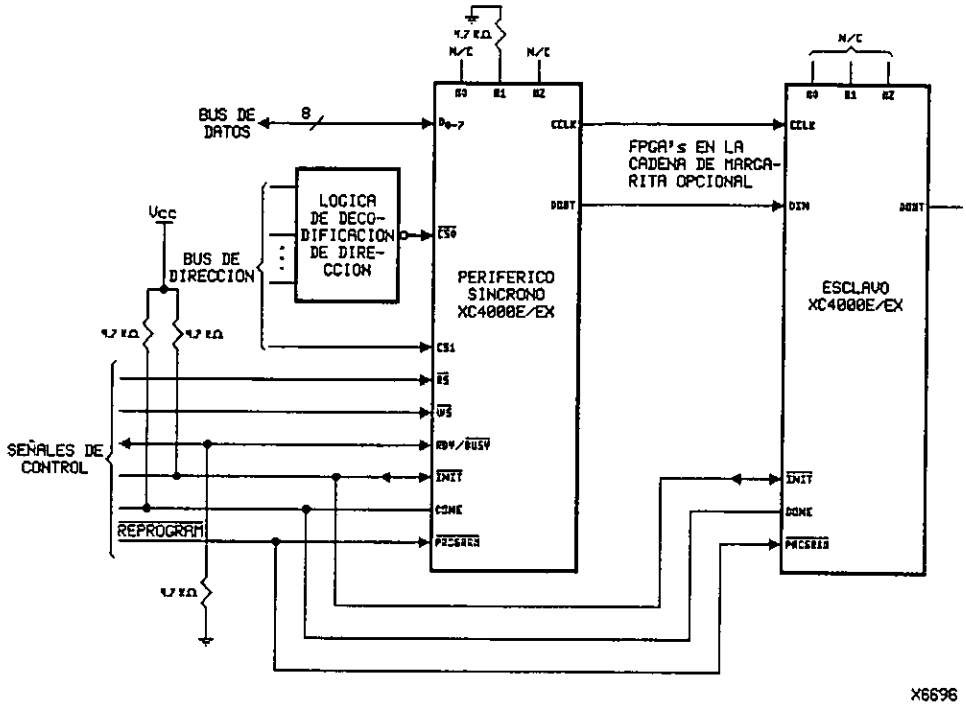
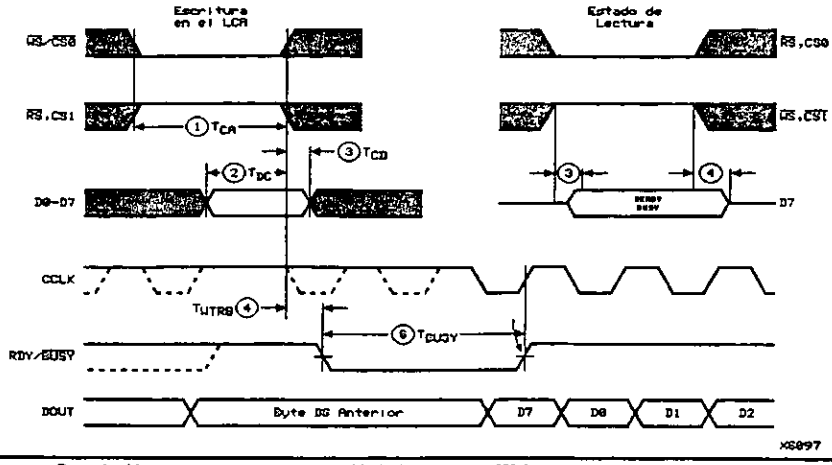


Figura 3.62: Diagrama de Circuito del Modo Periférico Asíncrono.



	Descripción.	Símbolo.	Mínimo.	Máximo.	Unidades.
ESCRITURA	Tiempo de Lectura Efectivo ($\overline{CS0}$, $\overline{WS}$ = bajo; $\overline{RS}$, CS1= alto).	1 T_{CA}	100		ns.
	Tiempo de establecimiento de DIN.	2 T_{DC}	60		ns.
	Tiempo de retención de DIN.	3 T_{CD}	0		ns.
RDY	Retardo RDY / $\overline{BUSY}$ después del final de la Lectura o Escritura.	4 T_{WTRB}		60	ns.
	Activación RDY / $\overline{BUSY}$ después de empezar la Lectura.	7		60	ns.
	Salida Baja RDY / $\overline{BUSY}$ (Nota 4).	6 T_{BUSY}	2	9	Períodos CCLK.

Notas: 1. La configuración debe ser retardada hasta que los pines $\overline{INIT}$ de todos los FPGA's de la cadena de margarita estén en estado alto.

2. El tiempo desde la finalización de $\overline{WS}$ hasta un ciclo de CCLK para el nuevo byte de datos depende de la terminación del procesamiento del byte anterior y de la fase del generador de temporización interno para CCLK.

3. Las temporizaciones CCLK y DOUT son probadas en modo esclavo.

4. T_{BUSY} indica que el convertidor tipo buffer doble paralelo a serial no está todavía listo para recibir nuevos datos. El T_{BUSY} más corto ocurre cuando un byte es cargado en un convertidor paralelo a serial vacío. El T_{BUSY} más largo ocurre cuando una nueva palabra es cargada en el registro de entrada antes de que el buffer de segundo nivel haya empezado ha cambiar los datos externamente.

Este diagrama de temporización muestra necesidades muy reducidas. Los datos no necesitan ser retenidos más allá del flanco ascendente de $\overline{WS}$. RDY / $\overline{BUSY}$ pasará a un estado activo dentro de 60 ns. después de que finaliza $\overline{WS}$. Una nueva escritura puede insertarse inmediatamente después de que RDY / $\overline{BUSY}$ pasa a un estado bajo, pero la escritura no puede terminarse hasta que RDY / $\overline{BUSY}$ haya pasado a un estado alto a causa de un período de CCLK.

Figura 3.63: Características de Conmutación Durante la Programación en Modo Periférico Asíncrono.

3.9.6 Modo Express (XC4000EX únicamente).

El modo Express es similar al modo Serial Esclavo, excepto que los datos se procesan un byte por cada ciclo de CCLK en lugar de un bit por cada ciclo de CCLK. Un suministro externo es usado para manejar CCLK, mientras los datos de byte ancho son cargados directamente en los registros de corrimiento de configuración de datos. Una frecuencia de CCLK de 1 MHz. es equivalente a una velocidad serial de 8 MHz., porque ocho bits de datos de configuración son cargados por cada ciclo de CCLK. El modo Express no soporta el chequeo de error CRC, pero soporta el chequeo de error de campo constante.

En el modo Express, una señal externa maneja a la entrada CCLK del dispositivo FPGA. El primer byte de datos de configuración paralela debe estar disponible en las entradas D del FPGA en un tiempo de establecimiento corto antes del segundo flanco CCLK ascendente. Los bytes de datos siguientes son temporizados en cada consecutivo flanco CCLK ascendente.

El modo Express sólo es soportado por las familias XC4000EX y XC5200. Este no puede usarse, por consiguiente, cuando un dispositivo XC4000EX o XC5200 está en una cadena de margarita con dispositivos de otras familias de Xilinx.

En caso de que el primer dispositivo sea configurado en modo Express, los dispositivos adicionales pueden estar en cadena de margarita sólo si cada dispositivo en la cadena es también configurado en modo Express. Los pines CCLK se unen todos a la vez y los pines D0 – D7 se unen de la misma manera para todos los dispositivos a lo largo de la cadena. Una señal de estado es pasada desde DOUT hasta CS1 de los siguientes dispositivos a lo largo de la cadena. El dispositivo principal en la cadena tiene su entrada CS1 ligada a un estado alto (o flotando, ya que hay un "pull – up" interno). Los datos de la estructura son aceptados únicamente cuando CS1 está en estado alto y la memoria de configuración de los dispositivos no está llena totalmente. El estado del pin DOUT es conectado a un estado bajo dos ciclos del oscilador interno después de que $\overline{INIT}$ es reconocido en estado alto, y permanece en estado bajo hasta que la memoria de configuración del dispositivo esté llena. DOUT es entonces llevado a un estado alto para indicarle al siguiente dispositivo en la cadena que acepte los datos de configuración en el bus D0 - D7.

Los pines DONE de todos los dispositivos en la cadena deberán ligarse juntos, con uno o más "pull – up's" internos activos. En caso de que un número grande de dispositivos sea incluido en la cadena, se deberán desactivar algunos de los "pull – up's" internos, ya que se maneja en estado bajo el pin DONE del último dispositivo en la cadena para que consuma la corriente de todos los

"pull – up's" en la cadena. El "pull – up" DONE es activado por predefinición. Puede desactivarse usando una opción de "MakeBits".

Los dispositivos XC4000EX en modo Express son siempre sincronizados para DONE. El dispositivo se pone activo después de que DONE pasa a un estado alto. DONE es una salida de colector abierto. Por lo tanto, con los pines DONE unidos, la señal externa DONE permanece en estado bajo hasta que todos los dispositivos son configurados, entonces todos los dispositivos en la cadena de margarita se activan simultáneamente. En caso de que el pin DONE de un dispositivo quede sin conectar, el dispositivo se activa en cuanto ese dispositivo se haya configurado. Los dispositivos XC5200 en la cadena deberán ser configurados como sincronizados con DONE (opción de "Makebits" CCLK_SYNC o UCLK_SYNC), y sus pines DONE alambrados junto con los de los dispositivos XC4000EX.

El modo Express debe especificarse como una opción al programa de "MakeBits", que genera la cadena de bits. La cadena de bits del modo Express no es compatible con el de los otros seis modos de configuración.

El modo Express es seleccionado por un <010> en los pines de modo (M2, M1, M0).

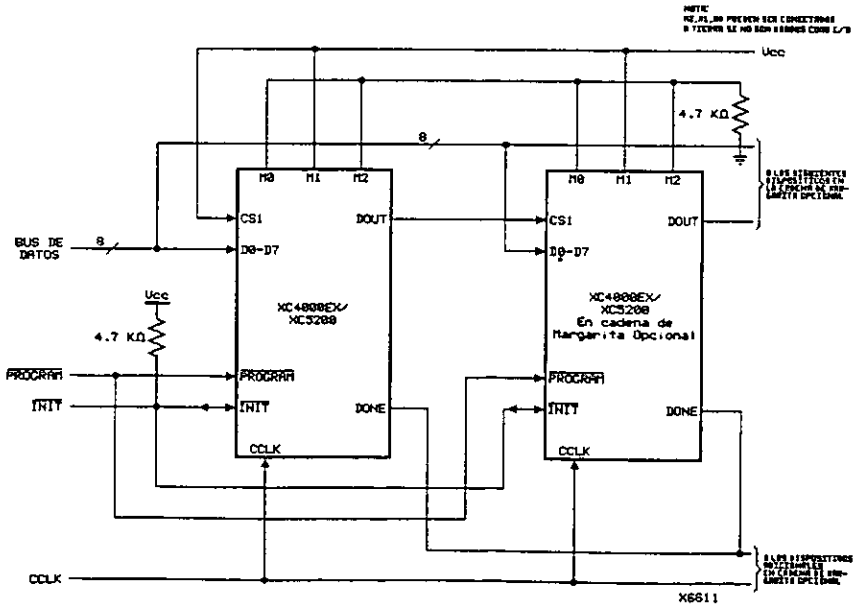
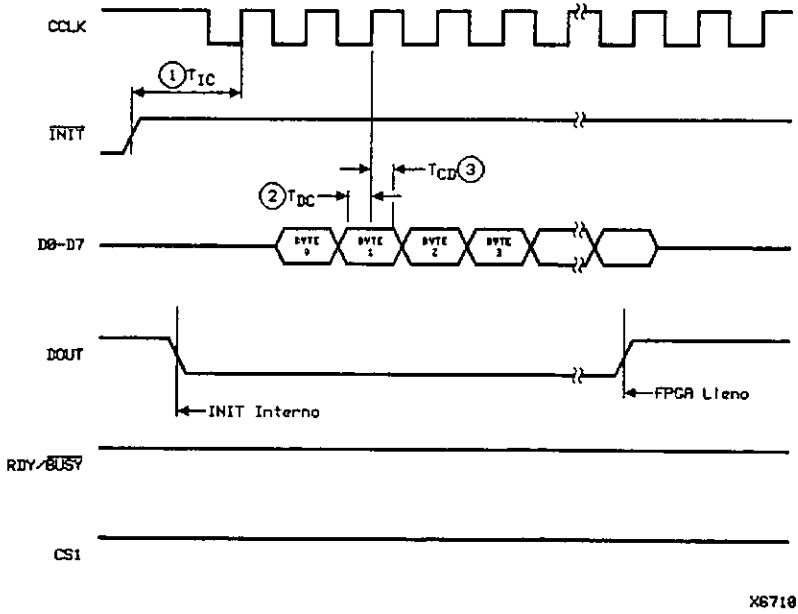


Figura 3.64: Diagrama de Circuito del Modo Express.

	Descripción.	Símbolo.	Mínimo.	Máximo.	Unidades.
CCLK	Tiempo de estableci-	T_{IC}	-		µs.
	miento (alto) $\overline{INIT}$.				
	Tiempo de estableci-	T_{DC}	-		ns.
	miento D0 – D7.				
	Tiempo de retención	T_{CD}	0		ns.
	D0 – D7.				
	Tiempo alto CCLK.	T_{CCH}	-		ns.
	Tiempo bajo CCLK.	T_{CCL}	-		ns.
	Frecuencia CCLK.	F_{CC}		-	MHz.
			Preliminar		



Nota: En caso de que no sea manejado por el procedimiento DOUT, CS1 *debe* permanecer en estado alto hasta que el dispositivo sea configurado totalmente.

Figura 3.65: Características de Conmutación Durante la Programación en Modo Express.

MODO DE CONFIGURACION <M2: M1: M0>.							
Serial Esclavo. <1: 1: 1>	Serial Maestro. <0: 0: 0>	Periférico Síncrono. <0: 1: 1>	Periférico Asíncrono. <1: 0: 1>	Paralelo Maestro Bajo. <1: 1: 0>	Paralelo Maestro Alto. <1: 0: 0>	Express. <0: 1: 0>	Operación de Usuario.
M2(ALTO) (I)	M2(BAJO) (I)	M2(BAJO) (I)	M2(ALTO) (I)	M2(ALTO) (I)	M2(ALTO) (I)	M2(BAJO) (I)	(I)
M1(ALTO) (I)	M1(BAJO) (I)	M1(ALTO) (I)	M1(BAJO) (I)	M1(ALTO) (I)	M1(BAJO) (I)	M1(ALTO) (I)	(O)
M0(ALTO) (I)	M0(BAJO) (I)	M0(ALTO) (I)	M0(ALTO) (I)	M0(BAJO) (I)	M0(BAJO) (I)	M0(ALTO) (I)	(I)
HDC (ALTO)	HDC (ALTO)	HDC (ALTO)	HDC (ALTO)	HDC (ALTO)	HDC (ALTO)	HDC (ALTO)	I/O
LDC (BAJO)	LDC (BAJO)	LDC (BAJO)	LDC (BAJO)	LDC (BAJO)	LDC (BAJO)	LDC (BAJO)	I/O
INIT	INIT	INIT	INIT	INIT	INIT	INIT	I/O
DONE	DONE	DONE	DONE	DONE	DONE	DONE	DONE
PROGRAM (I)	PROGRAM (I)	PROGRAM (I)	PROGRAM (I)	PROGRAM (I)	PROGRAM (I)	PROGRAM (I)	PROGRAM
CCLK (I)	CCLK (O)	CCLK (I)	CCLK(O)	CCLK (O)	CCLK (O)	CCLK (I)	CCLK (I)
		RDY/BUSY (O)	RDY/BUSY (O)	RCLK (O)	RCLK (O)		I/O
			RS (I)				I/O
			CS0 (I)				I/O
		DATO 7 (I)	DATO 7 (I)	DATO 7 (I)	DATO 7 (I)	DATO 7 (I)	I/O
		DATO 6 (I)	DATO 6 (I)	DATO 6 (I)	DATO 6 (I)	DATO 6 (I)	I/O
		DATO 5 (I)	DATO 5 (I)	DATO 5 (I)	DATO 5 (I)	DATO 5 (I)	I/O
		DATO 4 (I)	DATO 4 (I)	DATO 4 (I)	DATO 4 (I)	DATO 4 (I)	I/O
		DATO 3 (I)	DATO 3 (I)	DATO 3 (I)	DATO 3 (I)	DATO 3 (I)	I/O
		DATO 2 (I)	DATO 2 (I)	DATO 2 (I)	DATO 2 (I)	DATO 2 (I)	I/O
		DATO 1 (I)	DATO 1 (I)	DATO 1 (I)	DATO 1 (I)	DATO 1 (I)	I/O
DIN (I)	DIN (I)	DATO 0 (I)	DATO 0 (I)	DATO 0 (I)	DATO 0 (I)	DATO 0 (I)	I/O
DOUT	DOUT	DOUT	DOUT	DOUT	DOUT	DOUT	SGCK4-GCK5-I/O
TDI	TDI	TDI	TDI	TDI	TDI	TDI	TDI-I/O
TCK	TCK	TCK	TCK	TCK	TCK	TCK	TCK-I/O
TMS	TMS	TMS	TMS	TMS	TMS	TMS	TMS-I/O
TDO	TDO	TDO	TDO	TDO	TDO	TDO	TDO-(O)
			WS (I)	AO	AO		I/O
				A1	A1		PGCK4-GCK8-I/O
			CS1	A2	A2		I/O
				A3	A3		I/O
				A4	A4		I/O
				A5	A5		I/O
				A6	A6		I/O
				A7	A7		I/O
				A8	A8		I/O
				A9	A9		I/O
				A10	A10		I/O
				A11	A11		I/O
				A12	A12		I/O
				A13	A13		I/O
				A14	A14		I/O
				A15	A15		SGCK1-GCK7-I/O
				A16	A16		PGCK1-GCK8-I/O
				A17	A17		I/O
				A18*	A18*		I/O
				A19*	A19*		I/O
				A20*	A20*		I/O
				A21*	A21*		I/O
							TODOS LOS OTROS

*XC4000EX únicamente.

Notas: 1. Las celdas sombreadas de la tabla representan un "pull – up" de 50 KΩ - 100 KΩ antes de y durante la configuración.

2. (I) representa una entrada; (O) representa una salida.

3. $\overline{\text{INIT}}$ es una salida de colector abierto durante la configuración.

Tabla 3.22: Funciones del Pin Durante la Configuración.

CAPITULO 4

Técnicas de Diseño Lógico Programable.

4.1 INTRODUCCION.

El método convencional de diseño digital consta principalmente de ocho pasos que son:

- **Especificaciones:** Tareas a realizar
- **Entradas y Salidas:** Definición de los valores de entrada y establecimiento de los valores que deben generarse a la salida.
- **Tablas de Verdad:** Listado de los valores de entrada así como de los valores que tendrán las salidas para cada combinación posible de los valores.
- **Ecuaciones Booleanas:** Obtención de las ecuaciones booleanas, las cuales describen el funcionamiento de cada salida binaria a partir de las entradas, usando operaciones lógicas. Dichas ecuaciones pueden simplificarse con las técnicas de reducción de diseño digital más comunes.
- **Diseño a Nivel de Compuertas:** Creación del dibujo esquemático a nivel de compuertas para el circuito lógico. Cada operación lógica es reemplazada por el símbolo correspondiente y posteriormente se entrelazan las entradas y las salidas para representar la ruta entre las operaciones lógicas a realizar.
- **Simulación del Diseño a Nivel de Compuertas:** Antes de construir el circuito físico se debe verificar su funcionamiento mediante una simulación y así asegurar que no existen errores de diseño.

- **Construcción del Circuito:** Implementación física del diseño mediante circuitos integrados (CI's) que contienen diversos tipos de compuertas lógicas, sobre una placa prototipo y conexiones por medio de cables.

- **Depuración del Circuito:** Una vez que el circuito se construye físicamente, se aplican las entradas adecuadas al circuito y se checa que las salidas sean las correctas. Si las salidas son correctas, el circuito funciona adecuadamente, si no, se debe de verificar cada paso de la implementación lógica y física del sistema con el fin de detectar el error y poder corregirlo adecuadamente.

Con este método convencional se pueden construir circuitos digitales de hasta varios cientos de compuertas usando CI's de pequeña escala de integración (SSI) y de mediana escala de integración (MSI). Sin embargo, existen muchos inconvenientes derivados de este método de construcción:

- Cortar y pelar los cables para hacer las conexiones entre CI's implica un gran consumo de tiempo.

- Los cables se conectan con frecuencia en un lugar equivocado, por lo que debe hacerse un largo chequeo para encontrar el error a fin de hacer que el circuito funcione adecuadamente.

- Con un presupuesto limitado, no se puede tener disponible inmediatamente de cada tipo de CI TTL o CMOS.

- Una vez que el circuito funciona, se tienen que quitar partes del sistema para hacer espacio para un próximo diseño. Usar un circuito previamente construido como parte de otro diseño es algo inusual.

Por otro lado, existe una gran probabilidad de cometer errores cuando se hacen todos los pasos de diseño manualmente. Una simulación manual de la operación del circuito se realiza comúnmente de forma incompleta, por lo que se pueden encontrar errores adicionales al final, cuando el circuito se construye y no funciona correctamente.

Actualmente se usa un procedimiento completamente diferente para el diseño y construcción de circuitos. Dicho procedimiento comienza todavía con los pasos convencionales de diseño, pero posteriormente los detalles del circuito lógico necesarios para cumplir con la tabla de verdad son implementados por un programa de síntesis lógico. La operación de este circuito lógico se verifica usando un programa de simulación. Si la simulación del circuito es correcta, las compuertas y cableados se planean en un CI programable por el usuario; en este caso, un FPGA de la Serie XC4000 de Xilinx. Los programas de software determinan la forma en que las compuertas del dispositivo pueden conectarse para construir el circuito lógico. La salida de dichos programas es un archivo de configuración de cadena de bits que se transmite al FPGA para implementar el circuito lógico del diseño planteado.

Esto permite concentrar totalmente los recursos en las partes creativas del diseño lógico. Los programas lógicos se pueden escribir en un lenguaje de descripción de hardware (HDL) usando un editor de textos, o dibujando su circuito en un editor esquemático. El HDL o esquemático es compilado por programas de diseño asistido por computadora (CAD) para crear circuitos lógicos detallados que realizan las acciones especificadas en sus programas. Un programa de PC simula el circuito para asegurar una operación correcta. Entonces el diseño se transmite mediante el puerto paralelo de la PC a un FPGA, donde puede ser depurado.

4.2 PROCESO DE DISEÑO CON EL SOFTWARE PROGRAMABLE.

Para el proceso de diseño se empleara el Software Xilinx Foundation Series 1.5 el cual está formado por herramientas de soporte tales como Synopsys, Cadence, Mentor Graphics, Aldec, Viewlogic, Synplicity, Orcad, Model Technologies, Synario, Examplar, entre otras; y soporta formatos tales como VHDL, Verilog, EDIF, SDF y VITAL (actualmente existe una nueva versión del Software, la 2.1, que sólo posee algunos cambios poco significativos con respecto a la utilizada, razón por la cual sirve perfectamente de referencia). El Foundation posee un ambiente para crear programas que definen diseños lógicos. El proceso principal de diseño usando Foundation (mostrado en la Figura 4.1) se describe a continuación:

1. Los diseños digitales se ingresan usando editor HDL, esquemático, o de máquina de estados únicamente o en combinación.

2. Un simulador funcional verifica la operación del diseño compilado y permite inspeccionar los resultados para ver si realiza la función deseada. En caso de que se encuentre algún error, se puede retroceder y editar el archivo HDL, esquemático, y/o máquina de estados.

3. Las herramientas de implementación del Foundation compilan la lista de compuertas y conexiones, o "Netlist", en una cadena de bits que se usa para programar al FPGA. Este es el paso en el que un dispositivo en particular es especificado. Para los dispositivos de la Serie XC4000, la implementación requiere planear el circuito para la arquitectura FPGA, lo cual implica poner las compuertas en CLBs específicos, y entonces enrutar los cableados usando los PSM's.

4. Una simulación temporizada del diseño puede correrse después de que las herramientas de implementación del Foundation han determinado las compuertas y retardos de enrutamiento asociados con una planeación particular para una arquitectura FPGA.

5. El programa XSLOAD es usado para transmitir la cadena de bits a la Tarjeta XS40 (específicamente la XS40-010XL V 1.2).

6. La depuración se lleva a cabo forzando las entradas en la Tarjeta XS40 por medio del cable de puerto paralelo. Un diodo emisor de luz de siete segmentos (Display) sobre la tarjeta muestra la respuesta del FPGA.

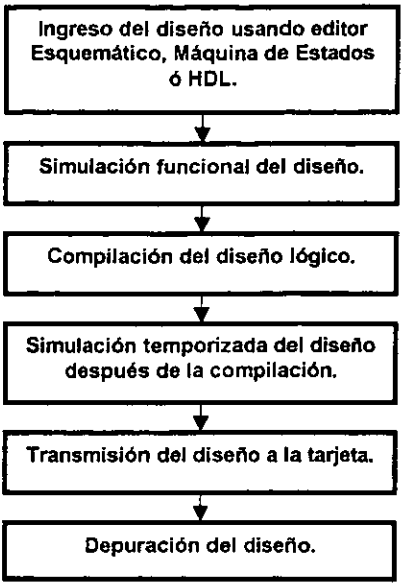


Figura 4.1: Proceso de Diseño Digital Usando Foundation.

Primero se describirá el proceso de un diseño combinacional para mejor entendimiento y posteriormente en la siguiente sección de un diseño secuencial.

Ejemplo de Diseño Combinacional.

A continuación se describe el proceso detallado a partir de la entrada o captura de un diseño sencillo.

1. **Especificaciones:** Decodificador de BCD a 7 segmentos (Figura 4.2).

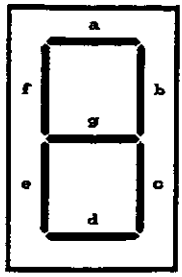


Figura 4.2: DISPLAY de Siete Segmentos de un Dígito.

2. **Entradas y Salidas:** A la entrada será aplicado el código BCD natural por lo que se tendrán cuatro entradas etiquetadas como ID, IC, IB y IA. Y a la salida, los siete segmentos del DISPLAY (sa, sb, sc, sd, se, sf y sg) los cuales se iluminan para cada combinación del código de entrada.

3. **Tabla de Verdad:** La Tabla 4.1 muestra la tabla de verdad para este diseño (un 1 lógico en una salida indica que el segmento correspondiente del DISPLAY se ilumina).

ID	IC	IB	IA	sa	sb	sc	sd	se	sf	sg
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

Tabla 4.1: Tabla de Verdad para el Decodificador de DISPLAY.

4. **Ecuaciones Booleanas:** A continuación se muestran las ecuaciones Booleanas que se derivaron de la tabla de verdad.

$$sa = (\overline{IA} \cdot \overline{IC} \cdot \overline{ID}) + (IA \cdot IB \cdot \overline{ID}) + (IA \cdot \overline{IB} \cdot IC \cdot \overline{ID}) + (\overline{IA} \cdot IB \cdot IC \cdot \overline{ID}) + (\overline{IB} \cdot \overline{IC} \cdot ID);$$

$$sb = (\overline{IB} \cdot \overline{IC} \cdot ID) + (\overline{IC} \cdot \overline{ID}) + (\overline{IA} \cdot \overline{IB} \cdot IC \cdot \overline{ID}) + (IA \cdot IB \cdot IC \cdot \overline{ID});$$

$$sc = (\overline{IB} \cdot \overline{IC} \cdot ID) + (IA \cdot IB \cdot \overline{ID}) + (\overline{IB} \cdot \overline{ID}) + (\overline{IA} \cdot IB \cdot IC \cdot \overline{ID});$$

$$sd = (\overline{IA} \cdot \overline{IC} \cdot ID) + (IA \cdot IB \cdot \overline{IC} \cdot \overline{ID}) + (IA \cdot \overline{IB} \cdot IC \cdot \overline{ID}) + (\overline{IA} \cdot IB \cdot IC \cdot \overline{ID}) + (\overline{IB} \cdot \overline{IC} \cdot ID);$$

$$se = (\overline{IA} \cdot \overline{IC} \cdot \overline{ID}) + (\overline{IA} \cdot IB \cdot IC \cdot \overline{ID}) + (\overline{IA} \cdot \overline{IB} \cdot \overline{IC} \cdot ID);$$

$$sf = (\overline{IA} \cdot \overline{IB} \cdot \overline{ID}) + (IA \cdot \overline{IB} \cdot IC \cdot \overline{ID}) + (\overline{IA} \cdot IB \cdot IC \cdot \overline{ID}) + (\overline{IB} \cdot \overline{IC} \cdot ID);$$

$$sg = (IB \cdot \overline{IC} \cdot \overline{ID}) + (\overline{IB} \cdot IC \cdot \overline{ID}) + (\overline{IA} \cdot IB \cdot IC \cdot \overline{ID}) + (\overline{IB} \cdot \overline{IC} \cdot ID);$$

4.2.1 Inicio de un Nuevo Proyecto.

El proceso de diseño comienza cuando se inicia el Foundation y se crea un nuevo proyecto.

Para comenzar, se debe crear un directorio principal para todos los proyectos (Ej: C:\XCPROJ) y posteriormente realizar los siguientes pasos:

1. Dar click sobre el icono **XILINX Foundation Project Manager**. Lo cual mostrará la ventana de la Figura 4.3.

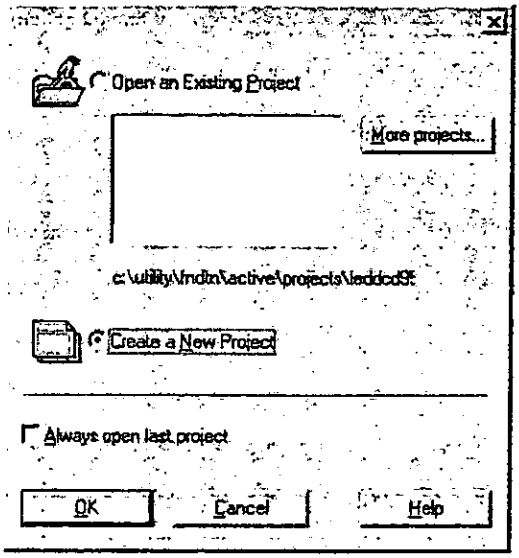


Figura 4.3: Ventana de Inicio Getting Started.

2. En la ventana **Getting Started** seleccionar **Create a New Project**. Y dar click en **OK** con lo que se muestra la ventana **New Project** (Figura 4.4).

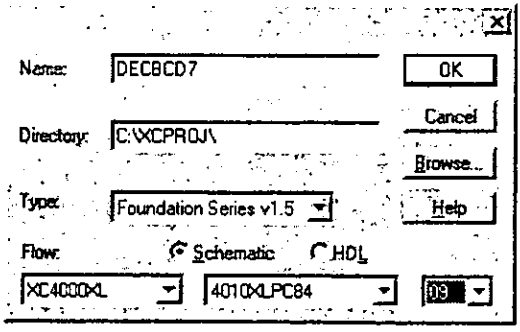


Figura 4.4: Ventana New Project.

3. En esta ventana, se debe capturar el nombre del proyecto, directorio del proyecto, tipo de sistema de desarrollo, modalidad del diseño (Esquemático o HDL), familia del chip, número de parte del chip, y velocidad el dispositivo (en un rango de 09, 1, 2, 3).

La Figura 4.4, muestra un ejemplo el cual crea un proyecto llamado DECBCD7 en el directorio C:\XCPROJ, que es especificado para un FPGA XC4010XL en encapsulado PLCC de 84 pines y de la velocidad más lenta.

4. Enseguida, dar un click en **OK** para volver a la ventana de **Project Manager**. Ahora la ventana deberá verse como la de la Figura 4.5.

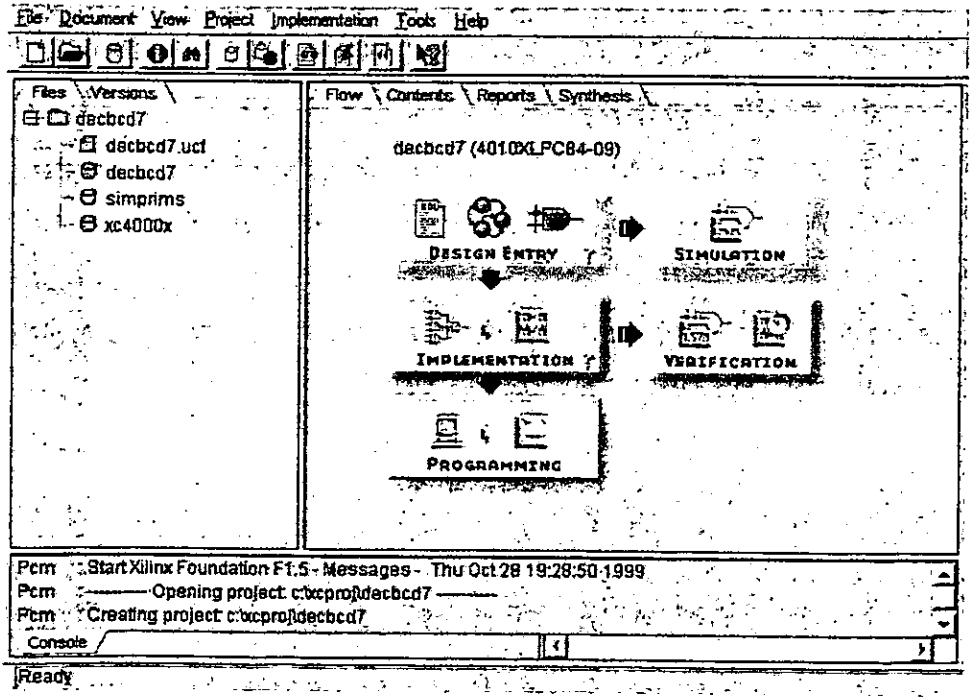


Figura 4.5: Ventana Principal Project Manager de Foundation Series 1.5.

Posteriormente, se elige la forma de ingreso del diseño de entre las tres opciones existentes; estas son: HDL, Editor Esquemático y Máquina de Estados.

4.2.2 Ingreso del Diseño Usando el Lenguaje de Descripción de Hardware ABEL.

En esta sección, se mostrara como ingresar el diseño. ABEL es uno de varios HDL's de uso común. Los otros son Verilog™ y VHDL (que son soportados por las herramientas Xilinx Foundation Series).

1. En la ventana **Project Manager**, seleccionar el menú **Tools** → **Design Entry** → **HDL Editor...** (o hacer click sobre el botón **HDL Editor** en la sección a extrema derecha de la ventana **Project Manager**).

2. En la ventana que aparece, dar click sobre el botón de opción etiquetado como **Use HDL Design Wizard** y entonces dar click sobre OK.

3. Enseguida dar click sobre **Siguiente** en la ventana de **Design Wizard** que se muestra para ponerse en contacto con la ventana **Design Wizard-Language**.

4. Aquí, seleccionar el botón de opción **ABEL** y hacer click sobre **Siguiente** para moverse a la ventana **Design Wizard-Name**.

5. Ingresar el nombre (Ej: DECODER) y dar click en **Siguiente**.

La ventana **Design Wizard-Ports**, especifica las entradas y salidas del diseño (Figura 4.6).

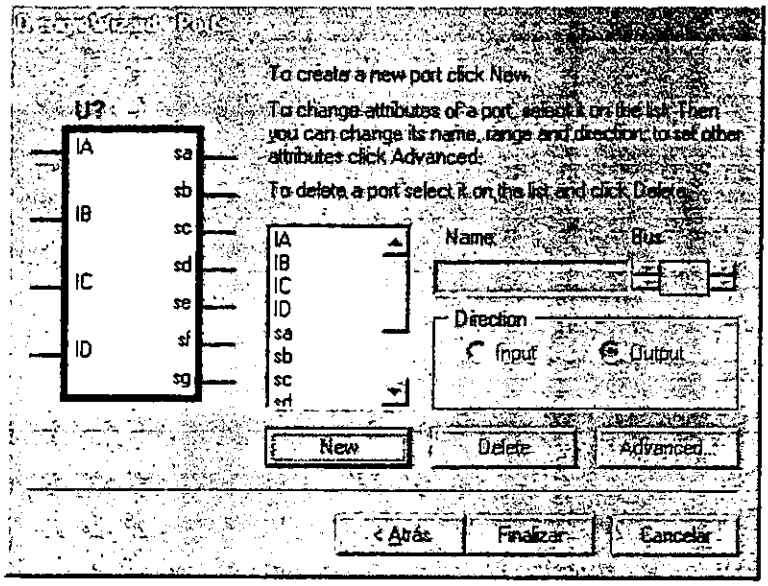


Figura 4.6: Adición de las Entradas y Salidas en el Diseño.

6. Para agregar una entrada, es necesario dar click sobre **New** y enseguida teclear el nombre de la entrada (ej: IA) en el campo de **Name**.

7. Dar click en el botón de opción **Input** para definir o declarar el puerto como una entrada.

Los pasos mencionados, se repiten para el resto de las entradas.

8. Para agregar una salida al circuito dar click sobre **New**, ingresar el nombre de la salida (Ej: sa) en el campo de **Name**, y dar click sobre el botón de opción **Output**.

9. Enseguida dar un click sobre el botón **Advanced** lo que mostrará la ventana **Advanced Port Settings**.

10. Dar click sobre el botón de opción **Combinatorial** y entonces dar click sobre el botón **OK**.

11. Una vez que todas las entradas y salidas han sido definidas, dar click en el botón **Finalizar**.

En este punto, la ventana del Editor HDL (etiquetada como **HDL Editor**) mostrará (Figura 4.7) la estructura lógica del diseño.

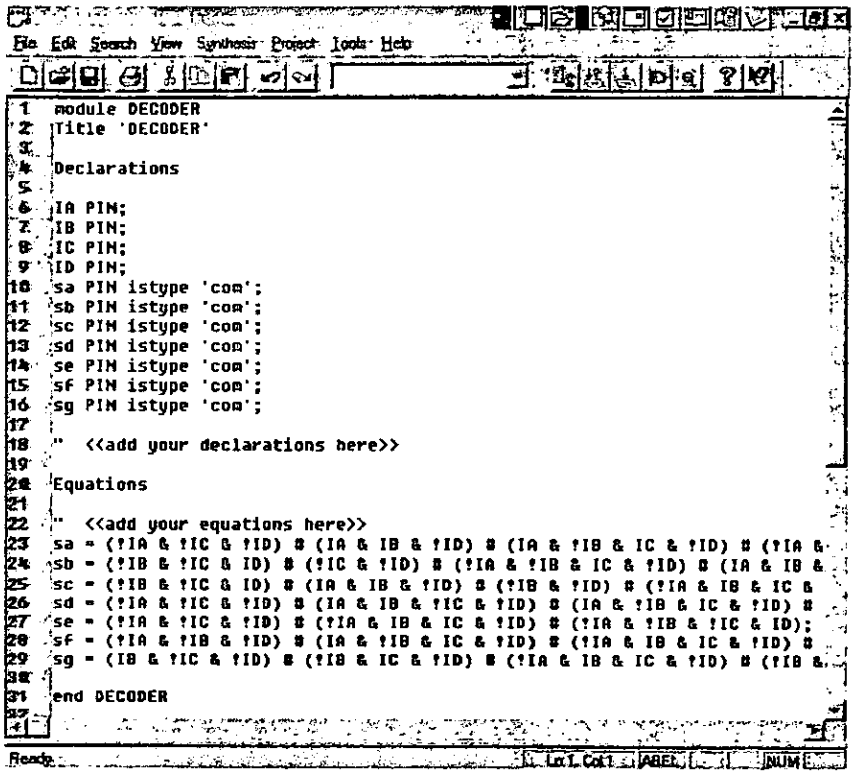


Figura 4.7: Ventana HDL Editor con el Esqueleto de Código en DECODER.

En esta ventana se muestran los nombres del módulo y título en la parte superior. Enseguida se observan las declaraciones de los pines de la(s) entrada(s) y salida(s) que se agregaron anteriormente. Además, este archivo, posteriormente tiene una sección destinada a las ecuaciones lógicas.

12. En dicha sección, se deben ingresar todas las ecuaciones booleanas que se derivaron de la (s) salida (s) en los primeros pasos de diseño. Estas se escriben a partir de la línea 23.

Nota: El ingreso de las ecuaciones a partir de la línea número 23 no es algo que deba seguirse necesariamente, ya que dicho número de línea depende de la cantidad de entradas y salidas que sean declaradas al inicio.

Aquí se usan operadores ABEL para expresar las siguientes operaciones Booleanas:

Operaciones Booleanas.	**Operador ABEL.**
AND.	&
OR.	#
NOT.	!

Ahora se tiene que verificar el diseño para asegurar que no se ha cometido ninguna equivocación.

13. Seleccionar el menú **Synthesis** → **Check Syntax**.

Una pequeña ventana se desplegará para mostrar información de que el código ABEL está siendo examinado para encontrar errores. En unos instantes, esta ventana es reemplazada por otra ventana que afirma que el chequeo ha sido exitoso (**Check Successful**).

14. Dar click en el botón **Aceptar** de esta ventana.

Dado que es común encontrar errores de sintaxis cuando se utiliza ABEL por primera vez. Cada error es resaltado en la ventana del Editor HDL y un mensaje de error aparece en la parte baja de la ventana.

Si se desea disponer de más ejemplos de errores de código ABEL realizar los siguientes pasos:

a) Seleccionar el menú **Tools** → **Language Assistant**.

En una ventana etiquetada como **Language Assistant-ABEL** que aparece se muestra una lista de temas sobre el lado izquierdo.

b) Dar click sobre los símbolos + para ingresar al tema en particular.

c) Dar click sobre el tema y un ejemplo de código ABEL para ese tema aparece sobre el lado derecho de la ventana. Se puede cortar y pegar el ejemplo en el código y entonces editarlo para el diseño en particular.

Una vez que se sabe que no hay errores de sintaxis, se debe generar una "Netlist" del código ABEL. La "Netlist" es un listado del código ABEL para el establecimiento de compuertas lógicas y las conexiones entre ellas.

15. Dar click en el menú **Synthesis** → **Options**.

16. En la ventana **ABEL6** que aparece, dar click sobre el botón de opción **Chip** y enseguida dar click en **Aceptar**.

Esto le indica al sintetizador que cree una "Netlist" para un diseño de una sola sección (el botón de opción **Macro** se emplea cuando el código ABEL describe un circuito lógico que se usa como parte de un diseño más grande).

17. Entonces dar click en el menú **Synthesis** → **Synthesize** para comenzar el proceso de compilación.

Una pequeña ventana se desplegará para mostrar información de que el código ABEL está siendo sintetizado. En unos instantes, esta ventana es reemplazada por otra ventana que afirma que la síntesis ha sido exitosa (**Synthesis Successful**).

18. Dar click en el botón **Aceptar** de esta ventana.

En este momento la "Netlist" del diseño ya existe; y la entrada del diseño se ha completado.

19. Entonces se debe seleccionar **File** → **Save** en la ventana del Editor HDL. Y enseguida se selecciona **File** → **Exit**.

Después de volver a la ventana **Project Manager**, se debe agregar el Archivo.ABL (Decoder.ABL) el cual es ahora parte del proyecto.

20. Seleccionar el menú **Document** → **Add** para que sean listadas extensiones de tipo **HDE (*.VHD, *.ABL)** en la ventana de diálogo.

21. Se debe resaltar el Archivo.ABL (Decoder.ABL) y dar click en **Abrir**.

El archivo Decoder.ABL será mostrado en la lista de archivos de la ventana **Project Manager** constituyendo el proyecto (Figura 4.8).

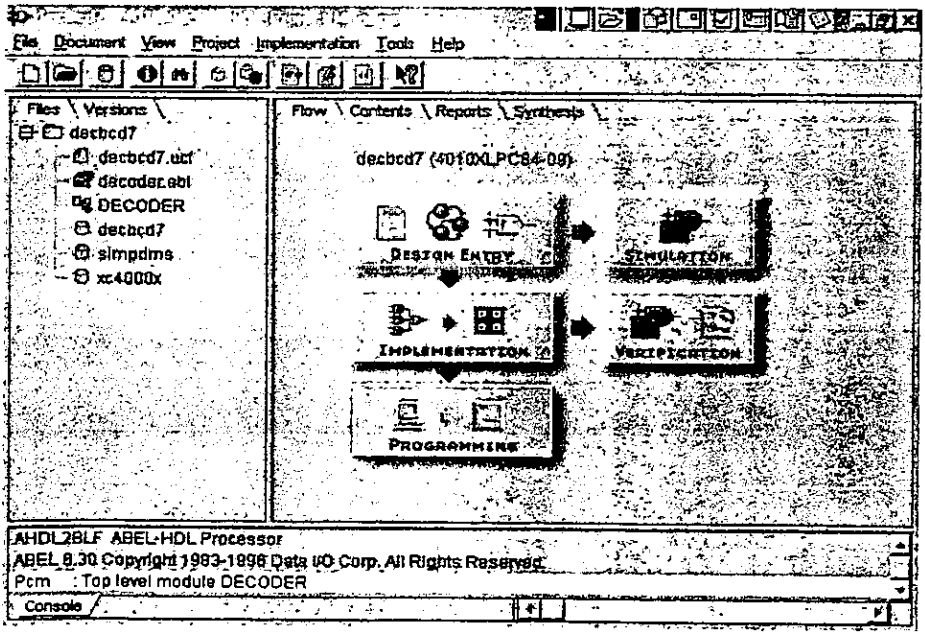


Figura 4.8: Ventana Project Manager Después de que el Archivo. ABL es Agregado al Proyecto.

a) Forma Análoga de Ingresar el Diseño Usando el Lenguaje de Descripción de Hardware ABEL.

Otra forma de efectuar el procedimiento anterior de una manera más rápida y fácil es comenzar como se explicó en la sección anterior el editor HDL y escoger ABEL como el HDL (Pasos 1 a 5). Y enseguida usar el **Design Wizard** para establecer las entradas y salidas. De la tabla de verdad del diseño de ejemplo, se sabe que se requieren cuatro entradas y siete salidas que se podrían escribir completamente, pero ahora se hará de una forma más práctica.

1. Dar click sobre el botón **New** y escribir una I (por ejemplo) en el campo de **Name**.

2. Enseguida se debe dar click en el botón superior del extremo derecho del campo de **Name** debajo de la etiqueta de **Bus**. Lo cual mostrará que I en el campo de nombre cambia a I[1:0].

3. Se deben dar dos clicks más y enseguida cambiará a I[3:0]. Esto especifica cuatro entradas (I0, I1, I2, e I3) como un vector de entrada de 4 bits.

4. Nuevamente se da click en **New** para ingresar la salida en el campo de **Name** (por ejemplo: S). Y enseguida dar click en el botón superior del extremo derecho del campo de **Name** debajo de la etiqueta de **Bus** para avanzar el nombre a S[6:0].

5. Al dar click sobre el botón de opción **Output** quedará establecido como un vector de salida de 7 bits.

6. Posteriormente, se da click en el botón **Advanced** y se establece el tipo de salida como Combinacional.

La Figura 4.9 muestra la asignación de puertos I/O para el decodificador de BCD a 7 segmentos.

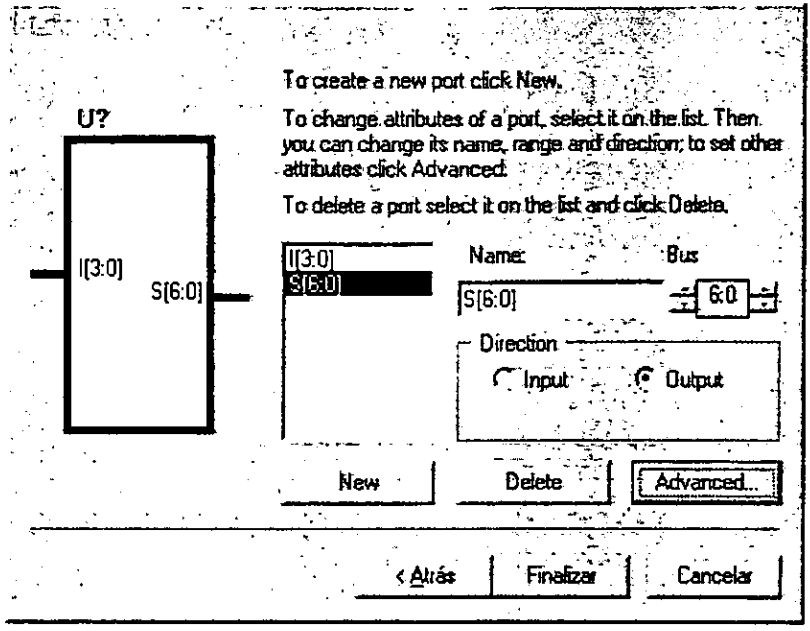


Figura 4.9: Puertos de Entrada y Salida Tipo Vector para el Decodificador de BCD a 7 Segmentos.

Una vez que se han definido las entradas y salidas, la ventana del Editor HDL aparece con la estructura del decodificador. Se podría tratar de ingresar las ecuaciones booleanas para el decodificador en esta ventana, pero en este caso se empleará la tabla de verdad.

7. Se ingresa la tabla de verdad con la sintaxis que se muestra en el Listado 4.1 y las herramientas del Foundation (herramientas F1) se encargarán de derivar las ecuaciones.

Nota: En el listado 4.1 se agregaron números de línea para propósitos de referencia, pero estos no son parte del código ABEL.

```
 1 module DECODER
 2 Title 'DECODER'
 3
 4 Declarations
 5
 6 I3..I0 PIN;
 7 I = [I3..I0];
 8 S6..S0 PIN istype 'com';
 9 S = [S6..S0];
10
11 Equations
12
13 TRUTH_TABLE
14 (I -> [S6, S5, S4, S3, S2, S1, S0])
15 0 -> [1, 1, 1, 1, 1, 1, 0 ];
16 1 -> [0, 1, 1, 0, 0, 0, 0 ];
17 2 -> [1, 1, 0, 1, 1, 0, 1 ];
18 3 -> [1, 1, 1, 1, 0, 0, 1 ];
19 4 -> [0, 1, 1, 0, 0, 1, 1 ];
20 5 -> [1, 0, 1, 1, 0, 1, 1 ];
21 6 -> [1, 0, 1, 1, 1, 1, 1 ];
22 7 -> [1, 1, 1, 0, 0, 0, 0 ];
23 8 -> [1, 1, 1, 1, 1, 1, 1 ];
24 9 -> [1, 1, 1, 1, 0, 1, 1 ];
25
26 end DECODER
```

Listado 4.1: Descripción del Archivo ABEL para el Decodificador de BCD a 7 segmentos.

El listado tiene varios elementos diferentes con respecto al método de la anterior sección 4.2.2, a continuación serán revisadas:

Línea 6: Las cuatro entradas I0, I1, I2, e I3 se declaran aquí como un vector. La notación ".." permite ingresar un vector de rango completo con sólo dar los puntos de principio y fin.

Línea 7: Esta es simplemente una expresión de igualdad que permite el uso de "I" como una forma abreviada para el vector de entrada [I3..I0].

Líneas 8-9: Estas declaran el vector de salida de siete elementos y su nombre de forma abreviada.

Línea 13: TRUTH_TABLE en ABEL es la palabra clave que señaliza el comienzo de una tabla de verdad.

Línea 14: Las entradas y salidas para la tabla de verdad son listadas en esta línea. El símbolo "->" separa la lista de entradas (en el lado izquierdo) de la lista de salidas (en el lado derecho). Se usa la forma abreviada del nombre para las entradas, pero el nombre completo de las salidas si es escrito.

Líneas 15-24: Cada combinación de entrada y las salidas resultantes se listan sobre estas líneas. Las combinaciones de entrada se escriben como dígitos decimales en vez de valores booleanos. Esto esta permitido porque se ha usado un vector de campo y de esta manera ABEL tiene la información suficiente para determinar cual es su significado (por ejemplo, en la línea 21 se estableció I = 6 que ABEL expande en las siguientes asignaciones I3 = 0, I2 = 1, I1 = 1, e I0 = 0). Se podría haber usado también un vector de campo para las salidas en la línea 14, pero esto hace que el listado de los segmentos individuales, tenga más sentido. Entonces, si se comete una equivocación sobre un segmento en particular, es más fácil retroceder y modificar la tabla de verdad sin afectar inadvertidamente otra salida.

Una vez que el código ABEL es ingresado, todos los pasos siguientes son los mismos que en la sección anterior, a partir del Paso 13 (chequeo de sintaxis, generar la "netlist", etc.).

4.2.3 Ingreso del Diseño Usando el Editor Esquemático.

En esta sección, se mostrará como ingresar el diseño usando el editor esquemático (esta sección es equivalente a la 4.2.2 de ABEL).

Nota: Si se prueba este proceso de ingreso se debe crear un Nuevo Proyecto.

1. En la ventana **Project Manager**, seleccionar el menú **Tools** → **Design Entry** → **Schematic Editor...** (o dar click sobre el botón **Schematic Editor** en la sección del lado derecho de la ventana **Project Manager**) y una ventana de Editor Esquemático será mostrada.

2. Se debe nombrar el diseño usando el menú **File** → **Save**.

El siguiente paso a realizar es agregar las compuertas para el circuito.

3. Seleccionar el menú **Mode** → **Symbols** de la ventana de Editor Esquemático y la ventana **SC Symbols** aparece con una lista de todos los tipos de componentes que se pueden usar.

A continuación, serán usadas las ecuaciones booleanas como base para el circuito:

$SA = (\overline{IA} \cdot \overline{IC} \cdot \overline{ID}) + (IA \cdot IB \cdot \overline{ID}) + (IA \cdot \overline{IB} \cdot IC \cdot \overline{ID}) + (\overline{IA} \cdot IB \cdot IC \cdot \overline{ID}) + (\overline{IB} \cdot \overline{IC} \cdot ID);$

$SB = (\overline{IB} \cdot \overline{IC} \cdot ID) + (\overline{IC} \cdot \overline{ID}) + (\overline{IA} \cdot \overline{IB} \cdot IC \cdot \overline{ID}) + (IA \cdot IB \cdot IC \cdot \overline{ID});$

$SC = (\overline{IB} \cdot \overline{IC} \cdot ID) + (IA \cdot IB \cdot \overline{ID}) + (\overline{IB} \cdot \overline{ID}) + (\overline{IA} \cdot IB \cdot IC \cdot \overline{ID});$

.

.

.

$SG = (IB \cdot \overline{IC} \cdot \overline{ID}) + (\overline{IB} \cdot IC \cdot \overline{ID}) + (\overline{IA} \cdot IB \cdot IC \cdot \overline{ID}) + (\overline{IB} \cdot \overline{IC} \cdot ID);$

Se debe desplazar a través de la lista de componentes en la ventana **SC Symbols** hasta encontrar las compuertas requeridas que son: compuertas AND de 2, 3 y 4 entradas, inversores, y compuertas OR de 3, 4 y 5 entradas.

Nota: La ventana **SC Symbols** incluye compuertas con diferente número de entradas, las cuales pueden ser normales o negadas, dependiendo de lo que se requiera (por ejemplo: **AND3B2** – compuerta AND de tres entradas con inversión en dos de estas).

4. Dar click sobre el nombre de la compuerta deseada (Ej: **AND3B3**) para seleccionarla y entonces mover el cursor para sacar el esquemático al área de dibujo.

   Enseguida deberá ver el símbolo para la compuerta deseada ligado al cursor.

5. Se debe dar click para colocar la compuerta en el área de dibujo (para ligar otra copia de la compuerta al cursor sólo se necesita dar click sobre la compuerta anteriormente colocada).

Para colocar una nueva compuerta al esquemático se debe seguir el procedimiento antes dicho. Los pasos 4 y 5, se repiten para el resto de las compuertas (Figura 4.10).

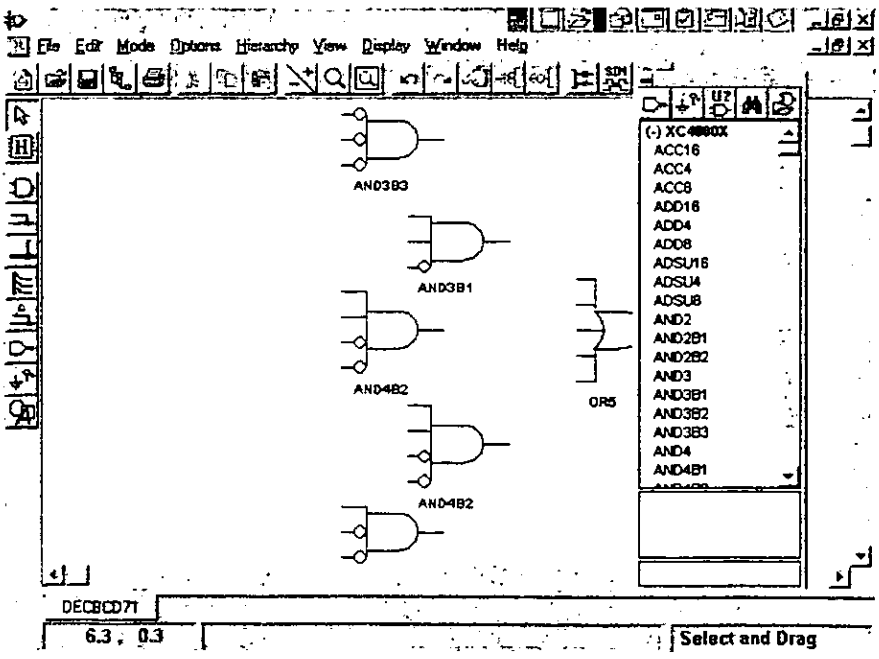


Figura 4.10: Colocación de Compuertas en la Ventana Schematic Editor.

También se necesita disponer las entradas y salidas del circuito.

6. Dar click sobre el botón final de la esquina superior izquierda de la barra de herramientas en la ventana **SC Symbols**. Una ventana de diálogo etiquetada como **Hierarchy Connector** será mostrada.

En esta ventana se verá el nombre de la terminal y tipo de esta (entrada, salida, etc.).

7. Se debe ingresar la información para la (s) entrada (s) (Ej: IA), como se muestra en la Figura 4.11.

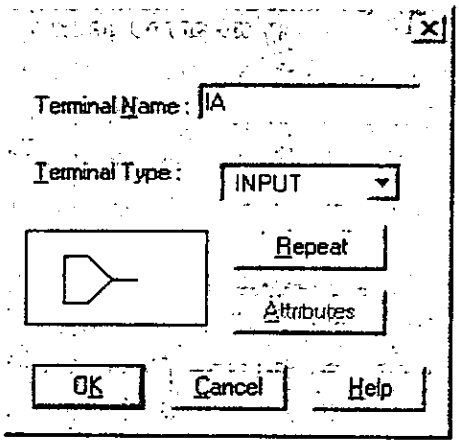


Figura 4.11: Adición de una Terminal al Esquemático.

8. Al dar click en **OK** un símbolo terminal de entrada será ligado al cursor. Solo se necesita dar click para colocar la terminal en el esquemático (ver la Figura 4.12).

Las demás entradas se agregan de la misma manera.

9. Las salidas se agregan del mismo modo, con la diferencia de que se debe seleccionar el botón de opción **Output** como tipo de terminal.

Hasta este momento, se han ingresado las terminales de entrada y salida, pero todavía se necesita agregar buffers entre las terminales y las compuertas lógicas. Estos buffers indican que las señales conectadas a ellos ingresarán y saldrán del FPGA a través de los pines I / O.

10. Para agregar buffers a la entrada, seleccionar el símbolo **IBUF** de la ventana **SC Symbols** y posteriormente situarlo cerca de cada terminal de entrada.

Dicho procedimiento debe repetirse para las demás entradas.

11. Para agregar buffers a la salida, seleccionar el símbolo **OBUF** para un buffer de salida y luego colocarlo cerca de la terminal de salida.

Repetir este procedimiento para las demás salidas.

12. Ya que se tienen todos los componentes necesarios dar doble click sobre la esquina superior izquierda de la ventana **SC Symbols** (y doble click nuevamente sobre la esquina superior izquierda de la ventana que aparece) para salir de esta sección.

El próximo paso es conectar las compuertas.

13. Se debe seleccionar el menú **Mode** → **Draw Wires** para comenzar el proceso.

14. Enseguida dar click sobre la salida de una compuerta seguido por otro click sobre la entrada de la siguiente compuerta en cuestión.

Una línea aparecerá conectando la salida de una compuerta con la entrada de otra, como se muestra en la Figura 4.12.

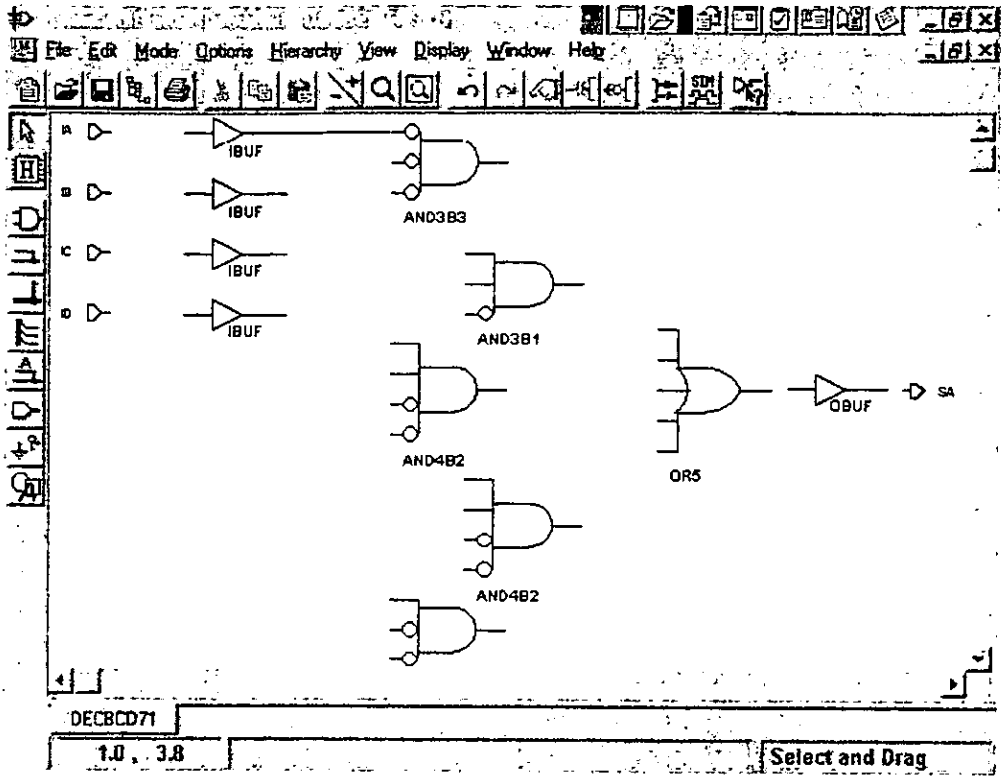


Figura 4.12: Conexión de Compuertas en la Ventana Schematic Editor.

Se debe continuar de esta manera hasta que todas las compuertas estén conectadas de acuerdo a las ecuaciones booleanas. Las terminales de entrada y salida se tienen que conectar únicamente con los símbolos **IBUF** y **OBUF**, respectivamente. Un ejemplo de un esquemático completo se muestra en la Figura 4.13.

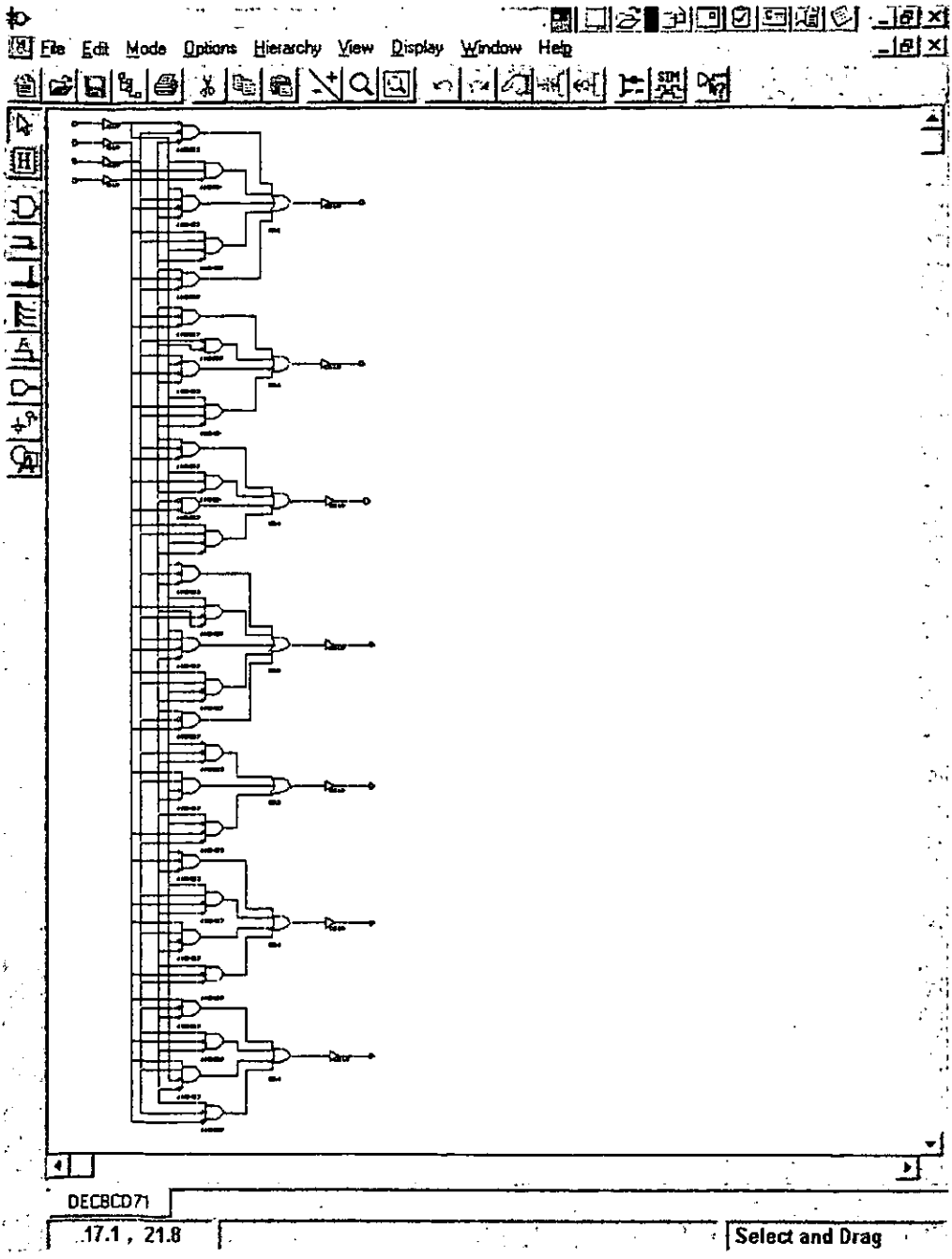


Figura 4.13: Esquemático Completo (Reducción).

Una vez que se ha terminado el esquemático, se tiene que verificar para encontrar posibles errores si es que los hubiera.

15. Se debe seleccionar **Options → Create Netlist**.

Esto activa un programa que examina el dibujo esquemático y genera un formato de máquina "Netlist" legible que describe los tipos de compuertas usadas y como son conectadas una con otra.

16. Posteriormente, se selecciona **Options → Integrity Test** para iniciar un chequeo de errores sobre la "Netlist".

El chequeo deberá indicar que no existen errores en el esquemático. Si la "Netlist" no tiene errores, se debe guardar el esquemático.

17. Para guardar el esquemático, se debe seleccionar el menú **File → Save As...**

Después, se debe exportar la "Netlist" al diseño. La "Netlist" es generada a partir del esquemático y contiene un listado de máquina legible de las compuertas lógicas y las conexiones entre ellas. La "Netlist" debe exportarse en un formato que las otras herramientas (como por ejemplo el simulador) comprendan.

18. Dar click en el menú **Options → Export Netlist...** Lo cual mostrará una ventana llamada **Export Netlist** (Figura 4.14).

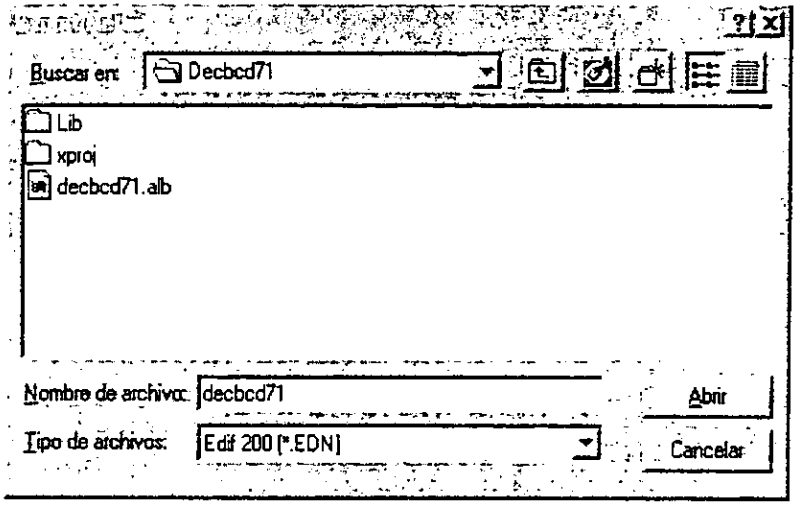


Figura 4.14: Ventana para Exportar una Netlist a Formato EDIF200.

19. Se debe seleccionar **Edit 200 [*.EDN]** en la casilla de selección **Tipo de Archivos** (EDIF 200 es el formato "Netlist" estándar para las herramientas F1).

Por algunos segundos, el proceso desplegara una ventana para mostrar como el formato "Netlist" interno del editor esquemático es exportado al formato EDIF 200.

20. Al terminar el proceso anterior, seleccionar **File** → **Exit** para cerrar el editor esquemático.

Después de volver a la ventana **Project Manager**, se debe agregar al Archivo.SCH (Decbcd71.SCH) el cual es ahora una parte del proyecto.

21. Seleccionar el menú **Document** → **Add** para que sean listadas extensiones de tipo Esquemático (***.SCH**) en la ventana de diálogo.

22. Se debe resaltar el Archivo.SCH (Ej. Decbcd71.SCH) y dar click en **Abrir**.

El archivo SCH será mostrado en la lista de archivos de la ventana **Project Manager** constituyendo el proyecto (Figura 4.15).

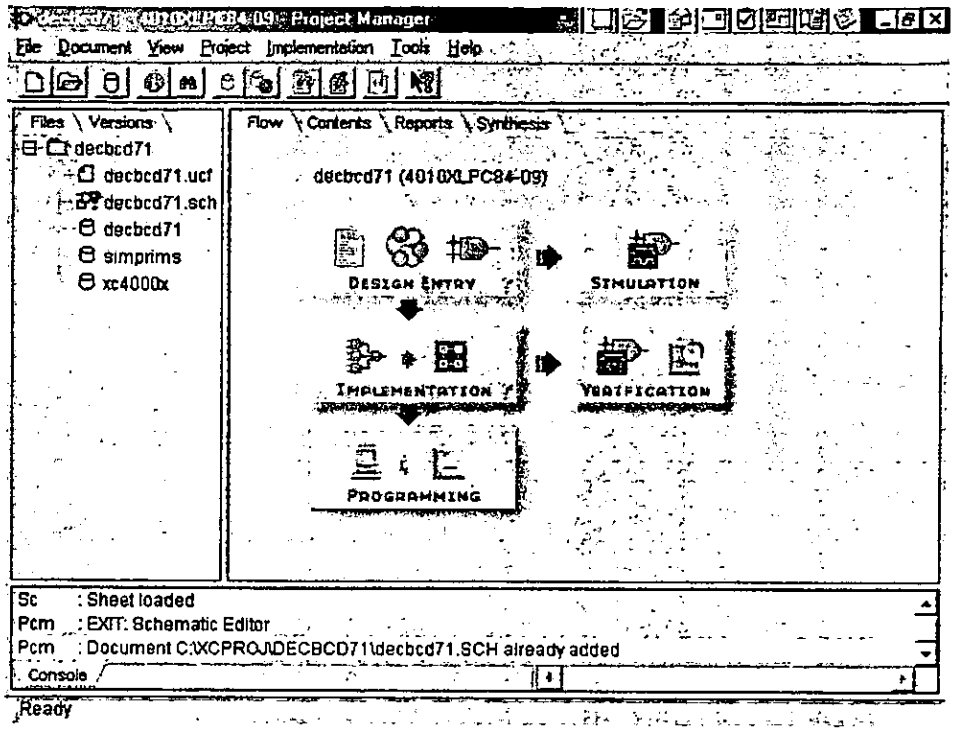


Figura 4.15: Ventana Project Manager Después de que el Archivo.SCH es Agregado al Proyecto.

4.2.4 Simulación Funcional del Diseño.

Hasta este momento, se ha ingresado la lógica para el diseño. Ahora es el momento de usar el simulador funcional para ver si lo que se ingreso funciona correctamente.

1. Para empezar, se debe seleccionar el menú **Tools** → **Simulation/Verification** → **Gate Simulator** (o dar click sobre el botón **Simulation**) en la ventana **Project Manager**. Esto mostrará la ventana **Logic Simulator**.

2. Enseguida se debe dar click sobre el botón de opción etiquetado como **Load Current Project Netlist** y dar click en **OK**, y a su vez, otra ventana individual vacía llamada **Waveform Viewer** aparecerá (Figura 4.16).

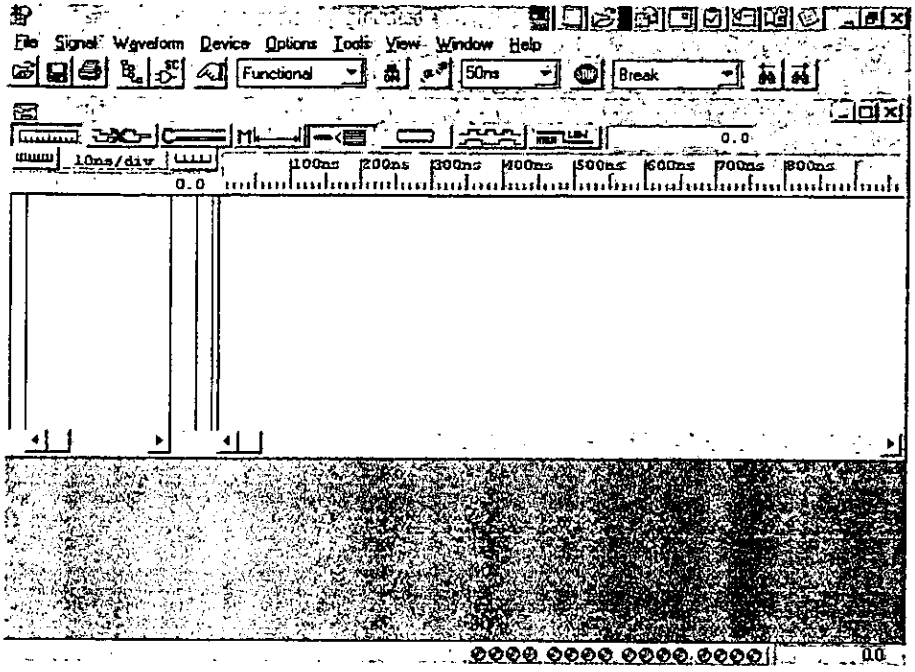


Figura 4.16: Ventanas Logic Simulator y Waveform Viewer.

A continuación se tienen que agregar las entradas y salidas del circuito lógico en la ventana **Waveform Viewer** donde se puede ver que sucede cuando el circuito se simula.

3. Se debe seleccionar el menú **Signal** → **Add Signals** de la ventana **Logic Simulator**. La ventana **Component Selection for Waveform Viewer** aparece.

4. En esta ventana, dar click sobre la entrada para resaltarla y enseguida dar otro click sobre el botón **Add**. Una forma de onda etiquetada con el nombre de la entrada será mostrada en **Waveform Viewer** y una marca de verificación roja aparecerá en la señal seleccionada (Figura 4.17).

Nota: Las señales adecuadas a seleccionar para la simulación son las que aparecen únicamente con el nombre que se les asigno al momento de definirlas (Ej: IA)

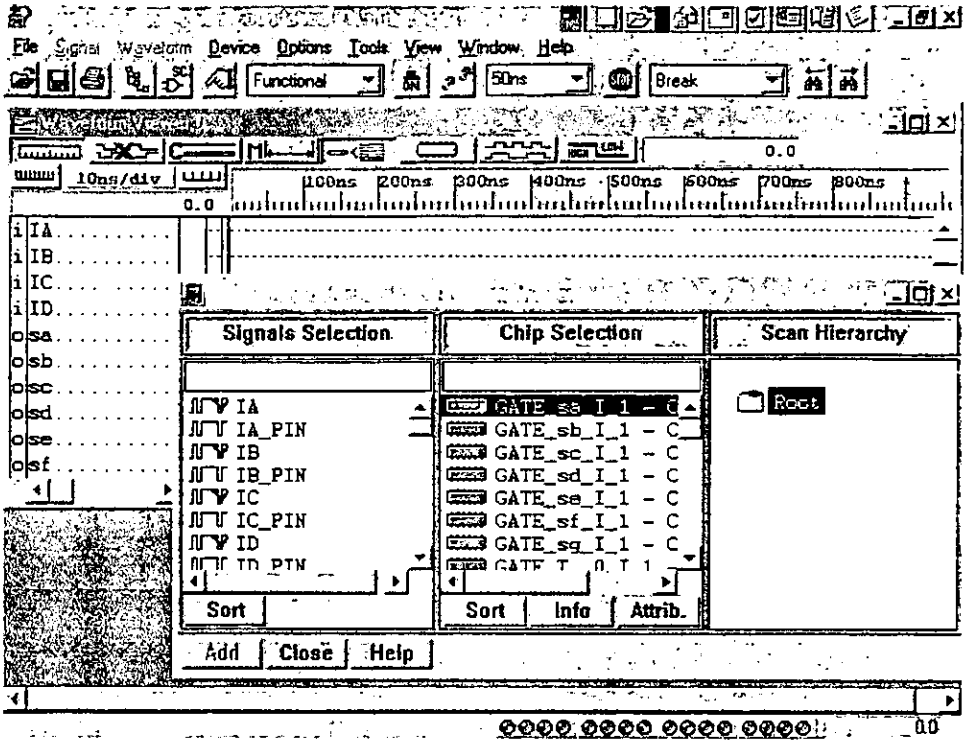


Figura 4.17: Adición de Señales al Waveform Viewer.

Se debe repetir este procedimiento para las demás entradas y salidas.

5. Una vez finalizado el procedimiento anterior hay que dar un click sobre el botón **Close**.

Ahora las entradas y salidas son desplegadas, pero nada interesante sucede porque todos las entradas están establecidas a 0 lógico. Por lo que se necesita aplicar un estímulo al circuito.

6. Para agregar un estímulo, se debe seleccionar el menú **Signal** → **Add Stimulators...** de la ventana **Logic Simulator**. Esto mostrará la ventana **Stimulator Selection** (Figura 4.18).

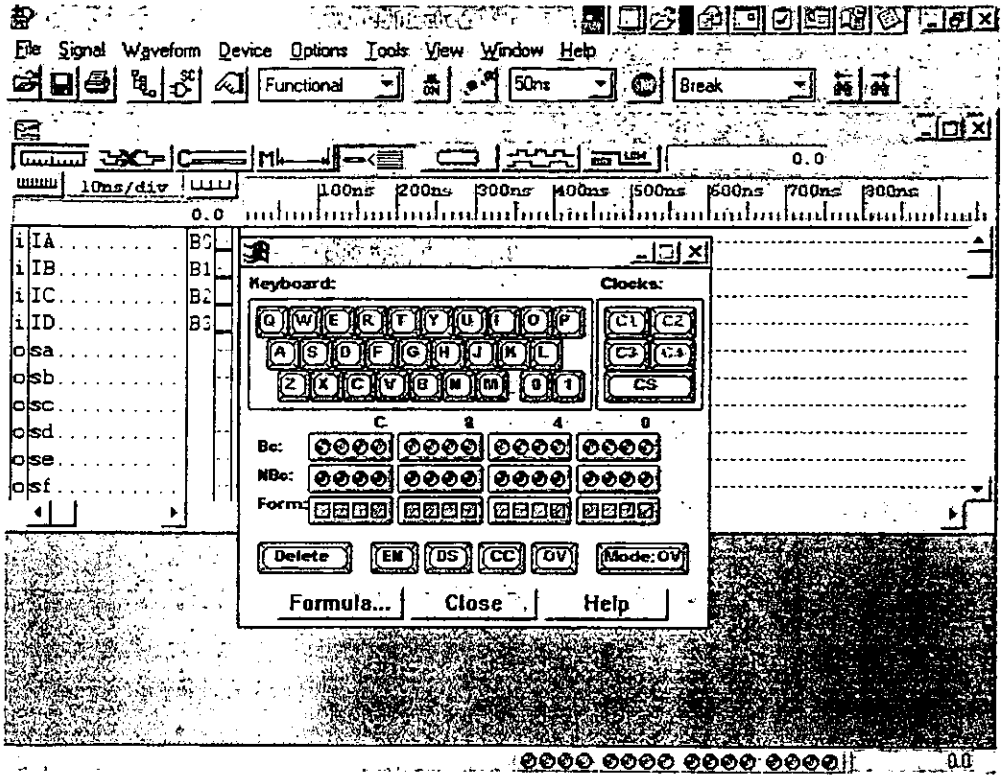


Figura 4.18: Ventana para la Selección de Estímulos.

Existe un gran número de botones en esta ventana, pero únicamente se ocuparan algunos de ellos según las necesidades. Durante una simulación, los bits de un contador irán a través de la siguiente secuencia (la secuencia se muestra para cuatro bits como ejemplo, ya que es la que se empleará para la simulación): 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000,1001, 1010, 1011, 1100, 1101, 1110, 1111. Se puede probar completamente la respuesta del circuito para cada combinación posible de entrada ligando las entradas del circuito lógico a los bits de un contador.

7. Con sólo dar click sobre uno de los bits cíclicos en la sección del contador (Bc) en la ventana **Stimulator Selection** y enseguida dar click sobre el nombre de la entrada en la ventana **Waveform Viewer** se logra ligar una entrada del circuito a un bit del contador (ver tercera columna de la Figura 4.18).

La etiqueta del bit del contador ligado a la entrada del circuito aparece a la derecha del nombre de la entrada en la ventana **Waveform Viewer**. Este procedimiento se debe repetir para todas las demás entradas.

8. Una vez que todas las entradas se han ligado a todos los bits del contador, se debe dar click en **Close** para abandonar la ventana **Stimulator Selection**.

Como ejemplo se muestra un conjunto de ligas entre los bits del contador y las entradas del circuito.

IA.	B0.
IB.	B1.
IC.	B2.
ID.	B3.

Ahora se necesita establecer un parámetro que controle la velocidad de la simulación.

9. Primero, se selecciona el menú **Options** → **Preferences...** de la ventana **Logic Simulator**.

10. Después, en la ventana etiquetada como **Preferences** (Figura 4.19), se debe establecer la frecuencia del bit B0 del contador binario (Ej: 50 MHz.). Y a continuación dar click sobre **OK**.

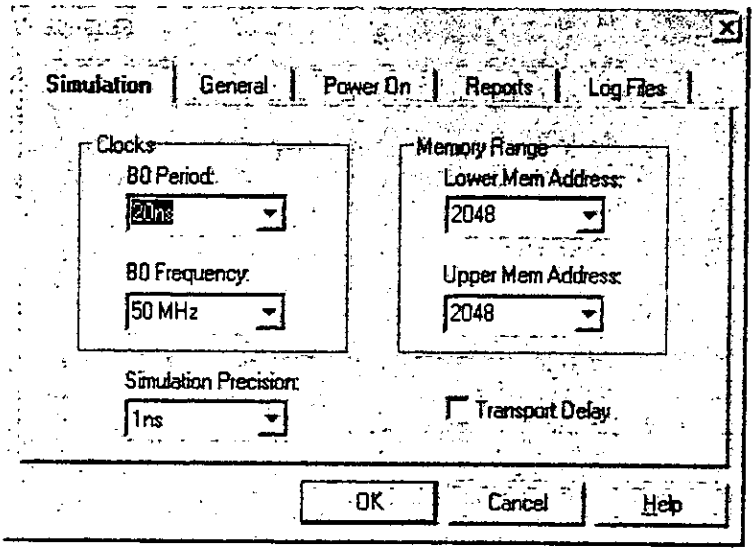


Figura 4.19: Ventana de Ajuste de los Parámetros de Control del Simulador.

11. Para correr la simulación, se debe seleccionar el menú **Options** → **Start Long Simulation...** de la ventana **Logic Simulator.**

12. A continuación, se da click sobre el botón **Start** de la ventana **Start Long Simulation** que aparece para iniciar la simulación funcional del diseño. Dentro de unos segundos, los resultados de la simulación aparecen mostrados en la ventana **Waveform Viewer** como se muestra en la Figura 4.20.

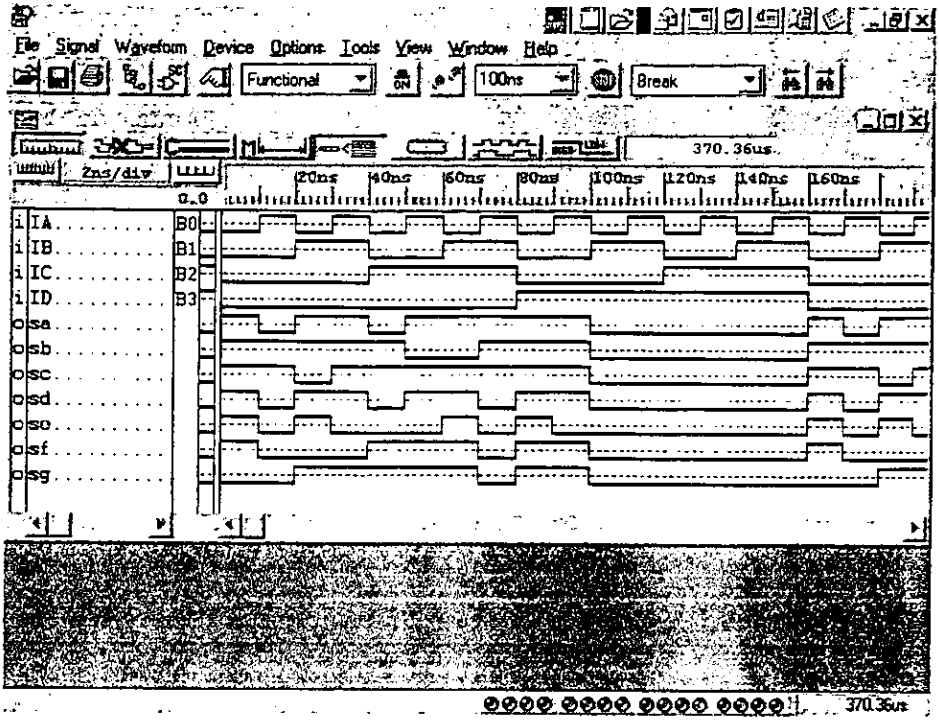


Figura 4.20: Resultados de la Simulación para las Salidas del Circuito.

Se deben notar los cambios a la (s) salida (s) con respecto a la (s) entrada (s), dichos cambios deben equiparar a lo establecido en la tabla de verdad. Esto indicará que el circuito funciona correctamente.

Después de realizar la simulación, se puede guardar, si es que se desea.

13. Para guardarla, se selecciona el menú **File** → **Save Simulation State...** y así quedará guardada en un archivo con extensión .DES.

14. Posteriormente, se debe seleccionar **File** → **Exit** para cerrar la ventana **Logic Simulator**.

4.2.5 Preparativos para Compilar el Diseño Lógico.

Después de que se ha ingresado el circuito lógico y simulado para verificar que funciona correctamente, se dará inicio con los preparativos para compilar el diseño en una cadena de bits para que pueda ser cargado en un FPGA y posteriormente probar físicamente el funcionamiento. Pero probar físicamente el diseño significa que se tendrán que aplicar señales a los pines del FPGA que corresponden a las entradas del circuito diseñado. También se necesitan conectar los pines al LED o DISPLAY que lleva las salidas del circuito para observar visualmente si el circuito está operando correctamente.

Pero como se mencionó al inicio, el FPGA utilizado es uno de la Serie XC4000 el cual se encuentra montado sobre la Tarjeta XS40 (específicamente la XS40-010XL V 1.2). En este dispositivo los pines pueden ser manejados por el puerto paralelo de una PC conectada a la tarjeta. Lo cual permite usar el puerto paralelo para aplicar estructuras de prueba al chip. Además, otro conjunto de pines está conectado a un DISPLAY de siete segmentos. Entonces, la tarjeta cuenta con todo lo que se necesita para trabajar.

Sin embargo, las herramientas de implementación del Foundation que compilan el archivo de HDL o esquemático no saben nada acerca de que los pines del FPGA sean ligados al puerto paralelo o a los segmentos del DISPLAY. Para esto, se necesita un método que permita pasar las asignaciones de pines requeridos a las herramientas Foundation.

Nota: Existen otras tarjetas en el mercado, además de la XS40-010XL V 1.2, por lo que las asignaciones aquí presentadas únicamente se aplican a esta; ya que para otras tarjetas puede haber diferencias. Se sugiere verificar la tarjeta a utilizar en futuras aplicaciones, así como los manuales de usuario correspondientes.

Hay tres métodos de hacerse:

1. Insertar los números de pin en las declaraciones de PIN del archivo de HDL.

2. Ingresar los números de pin como atributos de los buffers IBUF y OBUF en el esquemático.

3. Crear un Archivo de Coacción de Usuario (UCF) que liste el número de pin para cada entrada y salida.

Método 1.

Este método consiste en ingresar los números de pin directamente en el archivo de HDL lo cual resulta muy fácil. A continuación se muestra como debe ser la asignación de pines para el diseño implementado (ver Apéndice A):

```
IA      Pin=44;      # argumento B0 de XSPORT.

IB      Pin=45;      # argumento B1 de XSPORT.

IC      Pin=46;      # argumento B2 de XSPORT.

ID      Pin=47;      # argumento B1 de XSPORT.

sa      Pin=19;      # segmento a del DISPLAY.

sb      Pin=23;      # segmento b del DISPLAY.

sc      Pin=26;      # segmento c del DISPLAY.

sd      Pin=25;      # segmento d del DISPLAY.

se      Pin=24;      # segmento e del DISPLAY.

sf      Pin=18;      # segmento f del DISPLAY.

sg      Pin=20;      # segmento g del DISPLAY.
```

La Figura 4.21 muestra el Archivo.ABL (Ej: DECODER.ABL) con los números de pin agregados para permitir probar el diseño sobre la Tarjeta XS40.

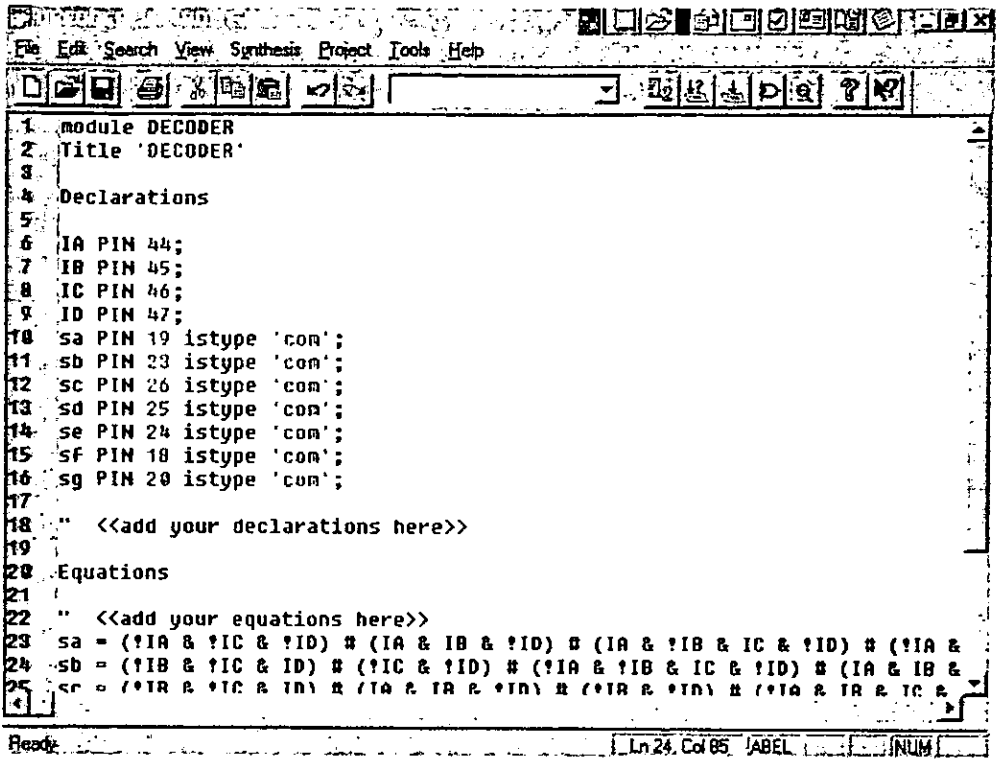


Figura 4.21: Archivo .ABL con las Asignaciones de Pin Insertadas.

Método 2.

Para este método se necesitan seguir los siguientes pasos:

1. Dar doble click sobre cualquier símbolo **IBUF** u **OBUF** del esquemático para establecer los atributos. Por ejemplo, dar doble click sobre cualquier IBUF ligado a una terminal de entrada con lo que aparece la ventana **Symbol Properties** mostrada en la Figura 4.22.

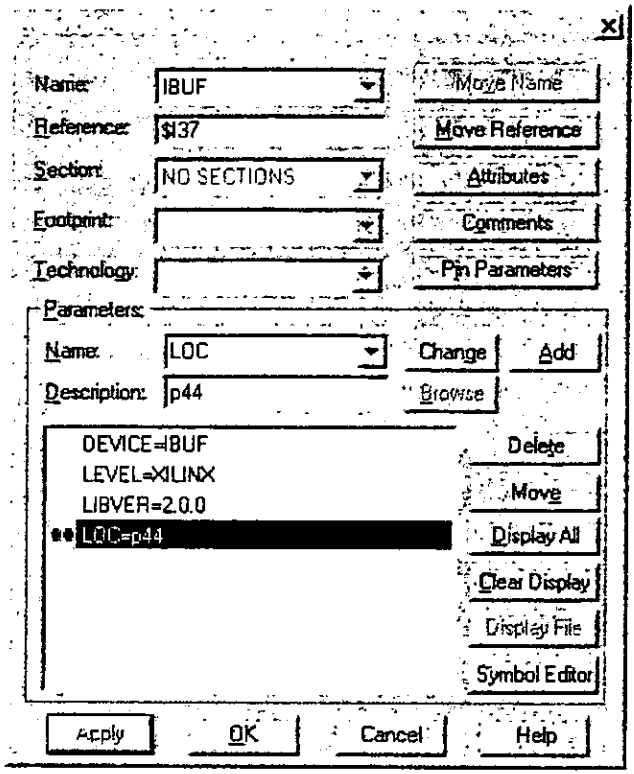


Figura 4.22: Ajuste del Atributo de Asignación de Pin para un Buffer en el Esquemático.

2. Posteriormente, dar click en el recuadro etiquetado como **Name** de esta misma ventana y escribir LOC.

3. Dar click en el recuadro etiquetado como **Description** y escribir el número de pin (Ej: p47).

4. Dar click en el botón **Add**.

5. Dar click en **OK**.

Este procedimiento para asignar las entradas y salidas a los pines deseados, debe repetirse de igual manera para todas las entradas y salidas, respectivamente.

Método 3.

Finalmente, en este método se puede usar el Editor HDL para crear un UCF que asigne las entradas y salidas del diseño a los pines I / O del FPGA.

1. En la ventana **Project Manager**, se debe seleccionar el menú **Tools** → **Design Entry** → **HDL Editor** y enseguida dar click sobre el botón **create empty** de la ventana etiquetada como **HDL Editor** que aparece (Figura 4.23), para mostrar una ventana de editor HDL vacía.

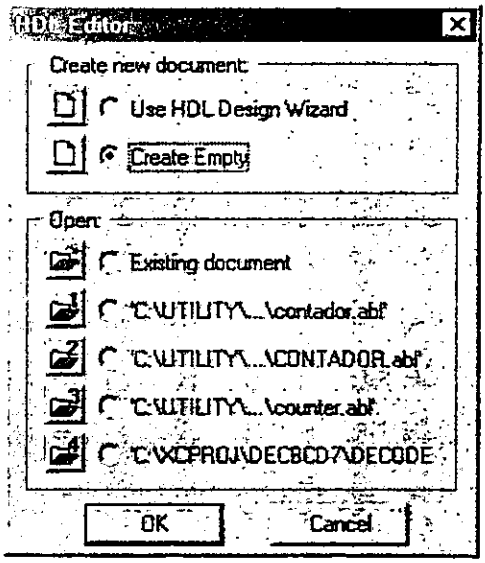


Figura 4.23: Ventana HDL Editor.

2. En esta ventana, se deben escribir las siguientes líneas (como ejemplo):

```
NET IA          LOC=P44;
NET IB          LOC=P45;
NET IC          LOC=P46;
NET ID          LOC=P47;
NET SA          LOC=P19;
NET SB          LOC=P23;
NET SC          LOC=P26;
NET SD          LOC=P25;
NET SE          LOC=P24;
NET SF          LOC=P18;
NET SG          LOC=P20;
```

Se debe notar que hay que usar letras mayúsculas para los nombres en el segundo UCF porque el esquemático (Archivo.SCH) sólo permite letras mayúsculas para los nombres de las terminales I / O.

3. A continuación se debe seleccionar en el menú **File → Save As...** de esta misma ventana para salvar el archivo como un Archivo.UCF (Ej: DECODER.UCF).

Todos los UCF informan a las herramientas de implementación del Foundation que pines se han asignado a la (s) entrada (s) y a la (s) salida (s).

4.2.6 Compilación del Diseño Lógico.

Una vez que se han ingresado las asignaciones de los pines y se está de vuelta en la ventana **Project Manager** se puede proceder con la compilación del diseño.

1. Se debe dar click sobre el botón **Implementation** ubicado en la sección derecha de la ventana para comenzar el proceso de compilación. La ventana **Implement Design** de la Figura 4.24 será mostrada.

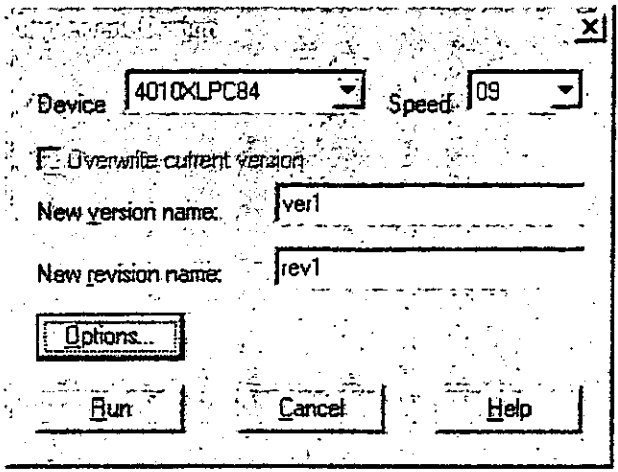


Figura 4.24: Ventana Implement Design.

El recuadro etiquetado como **Device** deberá reflejar lo que fue escrito cuando se comenzó el proyecto; y la versión y revisión deberán ambas estar a 1 (se puede cambiar cualquiera de estos parámetros si fuera necesario).

2. En la ventana **Implement Design**, dar click sobre el botón **Options...** mostrará la ventana usada para establecer los parámetros de control para las herramientas de implementación del Foundation (Figura 4.25).

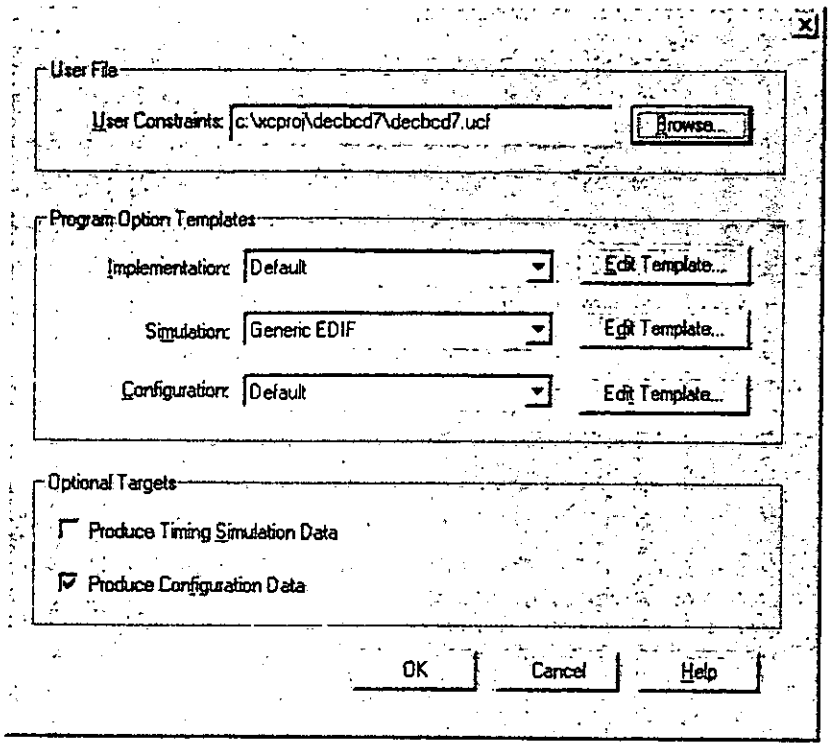


Figura 4.25: Ventana Options para Ajuste de los Parámetros de Control para las Herramientas de Implementación del Foundation.

3. Lo único que se necesita hacer es ingresar la ubicación del UCF en el recuadro **User Constraints** ubicado en la parte superior de la ventana. Se puede usar el botón **Browse...** para localizar el UCF; o escribir directamente su ubicación.

4. Enseguida se debe dar click en **OK** para regresar a la ventana etiquetada como **Implement**.

5. En la ventana **Implement Design**, se debe dar click sobre **Run**. La ventana **Flow Engine** aparece, la cuál muestra una interpretación gráfica de las etapas requeridas para crear una "Netlist" para un FPGA.

Para el diseño implementado la ventana **Flow Engine** es como la de la Figura 4.26.

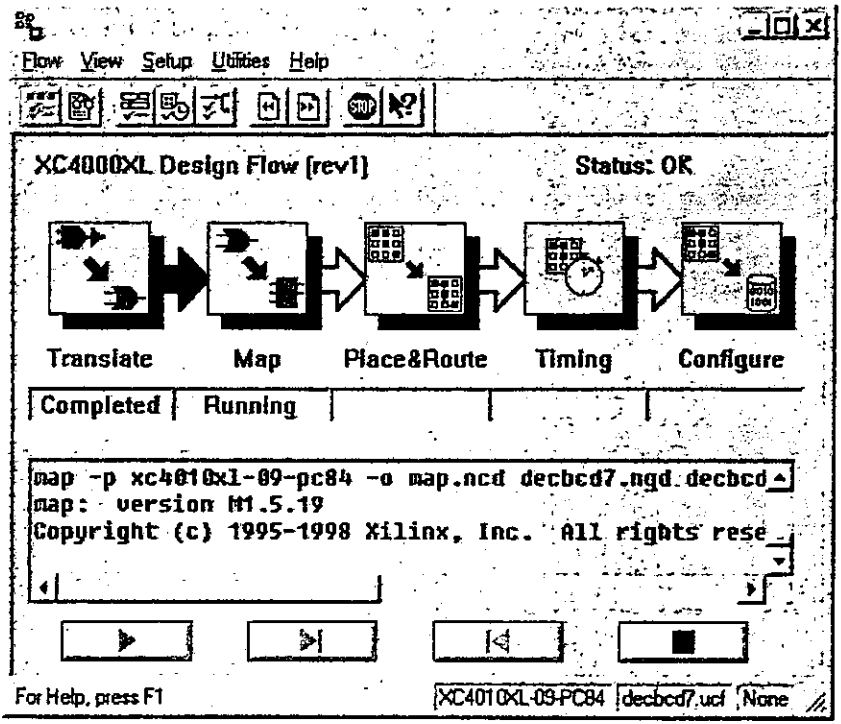


Figura 4.26: La Ventana Flow Engine Muestra los Pasos del Proceso para Crear una Netlist para un FPGA XC4000XL.

Las etapas principales en el proceso de implementación son las siguientes (existen más etapas dependiendo del FPGA):

Translate: la "Netlist" EDIF es convertida internamente en un formato "Netlist" y se realizan chequeos habituales de diseño.

Map: diversas optimizaciones del circuito lógico tratan de aumentar la velocidad del circuito y/o disminuir el número de compuertas usadas.

Place&Route: las compuertas en la "Netlist" se asignan a CLB's específicos y sus interconexiones se enrutan a través de PSM's y otros recursos de enrutamiento del FPGA.

Configure: la cadena de bits es generada para ser transmitida al FPGA y configurarlo para efectuar las funciones lógicas descritas en el archivo de HDL o esquemático.

Conforme avanza el proceso, la etiqueta para cada etapa cambia de **Running** a **Completed** como finalización (si ocurre un problema durante una etapa en particular del proceso, un cuadro de diálogo etiquetado como **Aborted** es mostrado de manera instantánea y un indicio del tipo de error será visualizado en un recuadro de diálogo en la parte inferior).

6. Después de que todas las etapas concluyen, se tiene que seleccionar el menú **Flow** → **Close** en la ventana **Flow Engine** para regresar nuevamente a la ventana **Project Manager**.

4.2.7 Como Llevar a Cabo una Simulación Temporizada.

En este ejemplo no será realizada una simulación temporizada detallada del circuito. Más adelante se verá con más detalle.

4.2.8 Transmisión del Diseño Lógico a la Tarjeta XS40.

Se debe seguir el siguiente procedimiento para la transmisión del diseño al FPGA.

Nota: Antes de iniciar el proceso de transmisión del diseño lógico a la tarjeta, se siguiere revisar el Apéndice A para la correcta instalación de esta.

1. Conectar la Tarjeta al puerto paralelo de la PC y escribir la ruta a donde se encuentra el diseño en una ventana DOS.

Para el diseño implementado se tiene la siguiente ruta de acceso:

CD C:\XCPROJ\DECBCD7

2. Posteriormente se debe emitir el comando:

XSLOAD DECBCD7.BIT

Una vez que finaliza el comando XSLOAD, la Tarjeta XS40 (específicamente la XS40-010XL V 1.2) ha sido programada con el diseño lógico.

4.2.9 Prueba del Diseño Lógico después de que es Transmitido.

Después de la transmisión del diseño, se necesita aplicar potencia a las entradas y ver si el circuito trabaja correctamente. El programa XSPORT acepta una cadena de bits "1" y "0" de salida en los ocho pines de datos del puerto paralelo que está conectado a los pines del FPGA en la tarjeta. Posteriormente, al dar el comando

XSPORT $b_7b_6b_5b_4b_3b_2b_1b_0$

donde b7 – b0 representan bits.

La correspondencia entre los bits de la cadena y los pines en el FPGA es como se indica a continuación:

Cadena de Bits.	Pin del FPGA.
b_0	44
b_1	45
b_2	46
b_3	47
b_4	48
b_5	49
b_6	32
b_7	34

XSPORT establece el bit más significativo a cero si se dan menos de ocho bits. Si a XSPORT se le dan más de ocho bits, guarda únicamente los octavos bits menos significativos.

La correspondencia entre las asignaciones de pin (Apéndice A), para las entradas, usadas en el diseño y la cadena de bits pasada a través de XSPORT es:

Cadena de bits.	Pin.	Entrada.
b_0	44	IA
b_1	45	IB
b_2	46	IC
b_3	47	ID
b_4	48	No Usado
b_5	49	No Usado
b_6	32	No Usado
b_7	34	No Usado

La asignación de pines realizada anteriormente también asignan las salidas sa, sb, sc, sd, se, sf y sg a los pines 19, 23, 26, 25, 24, 18 y 20 del FPGA.

Para probar completamente el circuito se tiene que usar XSPORT con todas las 10 combinaciones de entrada. A continuación se muestran los comandos que se pueden emitir y los resultados que se deberán obtener:

Comando.	ID.	IC.	IB.	IA.	DISPLAY 7-Seg.
XSPORT 0000	0	0	0	0	"0"
XSPORT 0001	0	0	0	1	"1"
XSPORT 0010	0	0	1	0	"2"
XSPORT 0011	0	0	1	1	"3"
XSPORT 0100	0	1	0	0	"4"
XSPORT 0101	0	1	0	1	"5"
XSPORT 0110	0	1	1	0	"6"
XSPORT 0111	0	1	1	1	"7"
XSPORT 1000	1	0	0	0	"8"
XSPORT 1001	1	0	0	1	"9"

Si todo el proceso se ha realizado correctamente, la prueba del diseño transmitido deberá equiparar los resultados de la simulación con los obtenidos en la tabla de verdad.

4.3 PROCESO DE DISEÑO PARA UN CIRCUITO SECUENCIAL.

Hasta ahora se ha explicado el proceso de diseño experimentado con circuitos combinacionales que generan sus salidas en respuesta única a la entrada actual. Cuando la entrada cambia, la salida cambia inmediatamente (después de algún retraso por compuertas, o de enrutamiento). Los circuitos combinacionales no tienen ninguna manera de recordar que ha sucedido antes.

Esta sección introducirá el diseño de circuitos secuenciales. Los circuitos secuenciales son capaces de recordar que ha sucedido anteriormente y pueden usarlo para alterar su comportamiento futuro. Es decir, el estado anterior determina el próximo estado. La razón de implementar un diseño secuencial es porque será descrito principalmente el proceso de simulación temporizada; debido a que una simulación de este tipo presenta diferencias con respecto a una simulación normal.

Ejemplo de Diseño Secuencial.

Como ejemplo de diseño, primero será implementado un flip – flop SR temporizado (construido con compuertas NAND), ya que es un ejemplo sencillo de un circuito secuencial. Posteriormente, será implementado un circuito secuencial un poco más complejo, un contador ascendente de 3 bits con flip – flops Toggle disparados por flanco y Clear Asíncrono, esto a consecuencia de que es uno de los circuitos secuenciales básicos a partir del cual podemos derivar cualquier otro.

4.3.1 Ingreso del Diseño.

A continuación, serán ingresados ambos diseños secuenciales.

Flip – Flop SR Temporizado.

Para empezar, será ingresado el diseño de un Flip – Flop SR Temporizado (Figura 4.27).

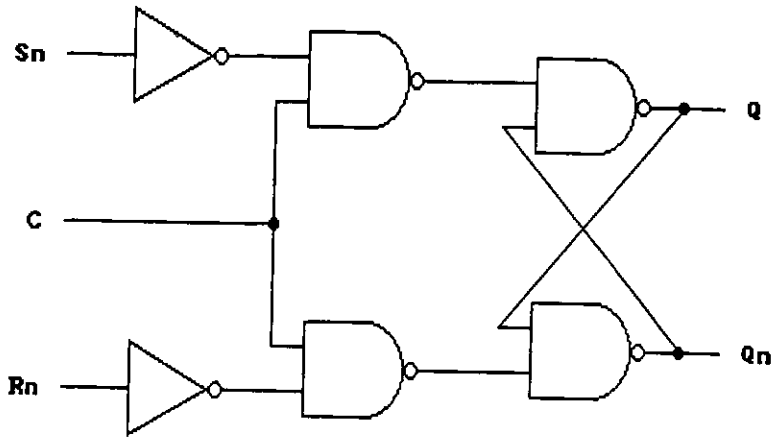


Figura 4.27: Flip – Flop SR temporizado.

1. Se debe crear un nuevo proyecto (Sección 4.2.1).

2. Posteriormente, se procede a ingresar el diseño usado el editor HDL (Sección 4.2.2).

El listado 4.2 muestra el código para este flip – flop.

```
1  module FFSRT
2  Title 'FFSRT'
3
4  Declarations
5
6  C PIN;
7  Sn PIN;
8  Rn PIN;
9  Q PIN istype 'com';
10 Qn PIN istype 'com';
11
12
13 Equations
14
15  Q = !( !(!Sn & C) & Qn);
16  Qn = !( !(!Rn & C) & Q);
17
18
19 end FFSRT
```

Listado 4.2: Código ABEL del Flip – Flop SR Temporizado.

Las líneas 15 y 16 describen las conexiones para un par de compuertas NAND acopladas por cruzamiento a las cuales se les ha agregado un conjunto de compuertas NAND entre el primer conjunto de compuertas acopladas por cruzamiento y las entradas Sn y Rn (como se muestra en la Figura 4.27). También se agregaron inversores a las entradas Sn y Rn para contrarrestar la inversión de las compuertas NAND agregadas y una entrada C para la temporización del flop – flop.

Se debe notar que se usó una "n" al final del nombre para denotar las señales que son activas en "0" lógico (Ejemplo: Sn); esta es simplemente una convención personalizada, que no tiene que seguirse necesariamente. También, las salidas Q y Qn se definieron como combinacionales (COM), pero esto debido a que el flip – flop se diseñó a partir de un circuito combinacional.

3. La simulación de formas de onda para este tipo de diseño se realiza igualmente ejecutando el simulador al seleccionar el menú **Tools** → **Simulation/Verification** → **Gate Simulator** de la ventana **Project Manager**.

4. Luego se debe dar click sobre el botón de opción etiquetado como **Load Current Project Netlist** y dar click en **OK**, y a su vez, otra ventana individual vacía llamada **Waveform Viewer** aparece.

5. Enseguida, se deben agregar las entradas y salidas tal como se describió anteriormente en la sección 4.2.4 (Pasos 3 a 5).

6. Después, se usa la ventana **Stimulator Selection** para ligar las entradas Sn, Rn y C a las teclas "q", "w" y "c".

Nota: El teclado de la PC debe permanecer en modo minúculas al momento de ligar las entradas y para el proceso de simulación.

Esto permite conmutar el valor de las entradas Sn, Rn o C precionando las teclas "q", "w", o "c", respectivamente.

7. Luego, se debe dar click sobre el botón **Simulation Step** de la barra de herramientas de la ventana **Logic Simulator** para llevar a cabo un paso de la simulación con las entradas actuales.

8. Entonces se usan las teclas para cambiar las entradas de datos y de temporización y hacer otro paso de la simulación al dar click nuevamente en el botón **Simulation Step**.

La simulación de formas de onda para el flip – flip SR temporizado se muestra en la Figura 4.28.

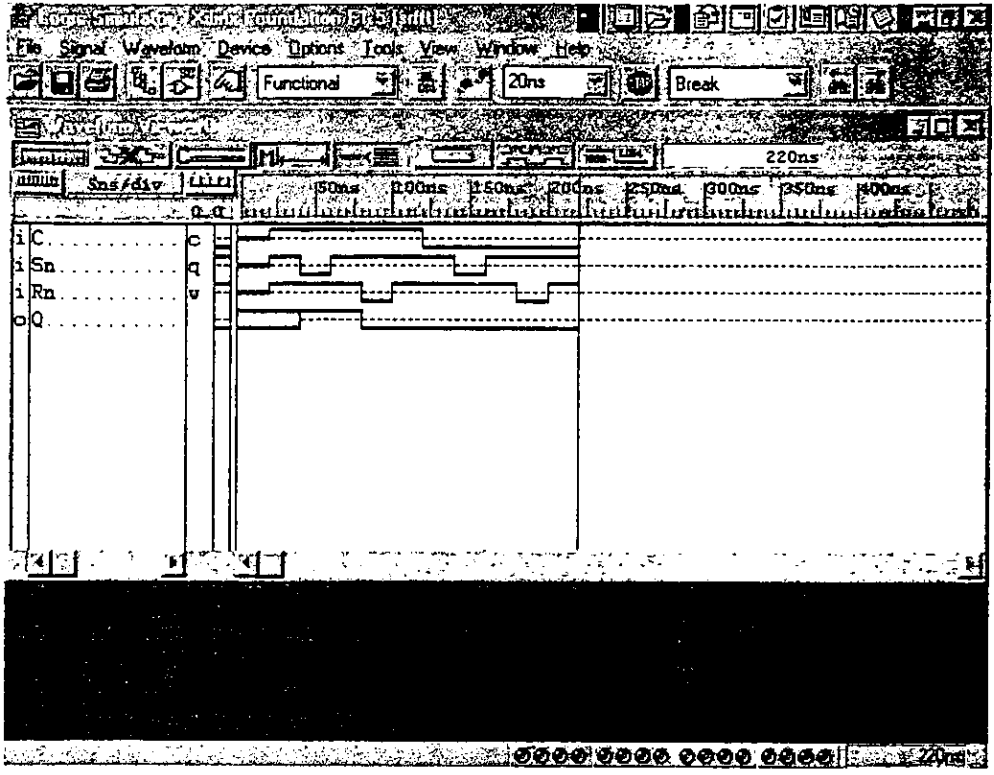


Figura 4.28: Simulación de Formas de Onda para el Flip – Flop SR Temporizado.

Cuando la entrada de reloj, C, está en estado alto las formas de onda son las esperadas para el flip – flop SR temporizado. Cuando C = 0, las entradas Sn y Rn no tienen ya ningún efecto sobre la salida Q. La operación simulada del flip – flop SR temporizado es correcta.

9. Para transmitir el flip – flop SR temporizado a la Tarjeta XS40 (específicamente la XS40-010XL V 1.2) para una prueba real, se crea un archivo de coacción de usuario tal como se describió en la sección 4.2.5 (Método 3) que asigne las entradas y salidas como se indica a continuación:

```
NET C     LOC=P44;   # argumento B0 de XSPORT.
NET Sn    LOC=P45;   # argumento B1 de XSPORT.
NET Rn    LOC=P46;   # argumento B2 de XSPORT.
NET Q     LOC=P25;   # segmento d del DISPLAY.
NET Qn    LOC=P26;   # segmento c del DISPLAY.
```

10. A continuación, se debe compilar y transmitir el diseño lógico a la Tarjeta XS40 (Sección 4.2.6 y 4.2.8).

11. Finalmente, se debe de probar el diseño lógico (Sección 4.2.9) después de que es transmitido.

Se deben ver los siguientes resultados al ejecutar el diseño en la Tarjeta XS40 con la secuencia específica de comandos:

Comando.	Rn.	Sn.	C.	Q (Seg. d).	Qn (Seg. c).
XSPORT 101	1	0	1	1	0
XSPORT 111	1	1	1	1	0
XSPORT 011	0	1	1	0	1
XSPORT 111	1	1	1	0	1
XSPORT 100	1	0	0	0	1
XSPORT 110	1	1	0	0	1
XSPORT 010	0	1	0	0	1
XSPORT 110	1	1	0	0	1

Estos resultados muestran que la entrada de reloj pasa y bloquea las entradas Sn y Rn como se desee.

Contador Ascendente de 3 Bits con Flip – Flops Toggle Disparados por Flanco y Clear Asíncrono.

Hasta este momento se ha ingresado el diseño completo para un flip – flop SR temporizado, por lo que a continuación se procede a ingresar el diseño del contador ascendente de 3 bits con flip – flops Toggle (Figura 4.29) construido a partir de flip – flops D.

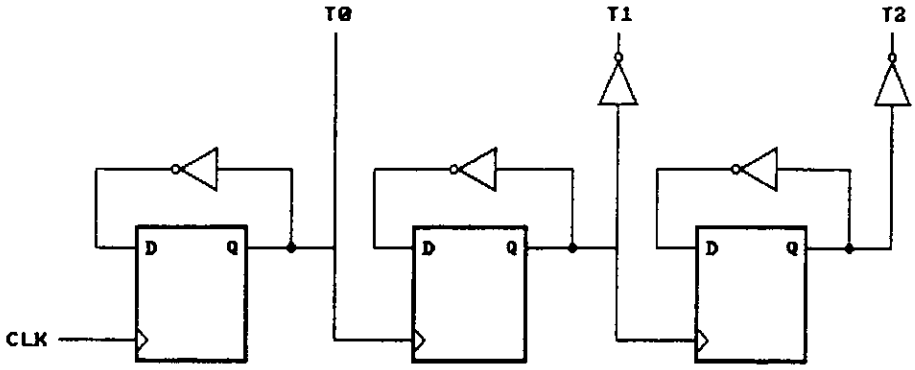


Figura 4.29: Contador ascendente de 3 Bits con Flip – Flops Toggle.

Como se puede notar, el diseño anterior fue construido con un flip – flop que usa compuertas lógicas básicas. Para mejorar la utilización de recursos en los FPGA's, XILINX ha incluido flip – flops incorporados.

El presente diseño hará uso de estos, y lo único que se tiene que hacer es aprender como usarlos.

1. Se debe crear un nuevo proyecto (Sección 4.2.1).

2. Posteriormente, se procede a ingresar el diseño usando el editor HDL (Sección 4.2.2).

El Listado 4.3 muestra el código para un contador ascendente de 3 bits como el de la Figura 4.27 construido a partir de flip – flops Toggle los cuales se obtienen de la conversión de flip – flops D.

```
1  module CONTADOR
2  Title 'contador'
3
4  Declarations
5
6  C PIN;
7  ACLR PIN;
8  T2 PIN istype 'reg';
9  T1 PIN istype 'reg';
10 T0 PIN istype 'reg';
11
12
13 Equations
14
15 [T2,T1,T0] := ![T2,T1,T0];
16 T0.CLK = C;
17 T1.CLK = !T0;
18 T2.CLK = !T1;
19 [T2,T1,T0].ACLR = ACLR;
20
21 end CONTADOR
```

Llistado 4.3: Código ABEL del Contador Ascendente de 3 Bits con Flip – Flops Toggle.

Comenzando con la línea 8, 9 y 10, tenemos que se declaran tres flip – flops D: T0, T1, y T2. El uso del tipo "REG" en vez de "COM" distingue a T0, T1 y T2 como salidas de registro en vez de salidas combinacionales como se ha usado anteriormente.

Estos flip – flops D se transforman en flip – flops Toggle a través de cargarlos con los inversos de sus contenidos (línea 15). Se debe notar el uso de ":=" en vez de "=". El ":=" implica que la salida únicamente toma el valor de la entrada cuando ocurre una transición en la entrada de reloj.

Entonces el bit menos significativo del contador es temporizado por los flancos ascendentes del reloj de entrada C, en la línea 16. En la línea 17, el siguiente bit del contador es temporizado por T0, la salida del bit menos significativo. De la misma manera, el bit más significativo del contador, T2, es temporizado por T1 en la línea 18. Se debe notar que las salidas T1 y T2 en las líneas 17 y 18 son cargadas con sus inversos para que el contador sea ascendente. Finalmente, en la línea 19 se provee un clear asíncrono activo en alto para cada flip – flop T de tal manera que se pueda dar clear al contador en cualquier momento si es necesario.

3. La simulación de formas de onda se realiza igual que en el ejemplo anterior ejecutando el simulador al seleccionar el menú **Tools** → **Simulation/Verification** → **Gate Simulator** de la ventana **Project Manager**.

4. Luego se debe dar click sobre el botón de opción etiquetado como **Load Current Project Netlist** y dar click en **OK**, y a su vez, otra ventana individual vacía llamada **Waveform Viewer** aparece.

5. Cuando se usan flip – flops incorporados, se necesita incluir un "reset", (SimGlobalReset), en el conjunto de entradas de control al momento de agregar las entradas como se describió en la sección 4.2.4 (Pasos 3 a 5).

Poner un "0" lógico en el "reset" al arranque de la simulación inicializa a todos los flip – flops en estados conocidos de tal manera que el simulador no consigue desorientarse. Después de que el reset se aplica, los flip – flops incorporados trabajan simplemente como los que se construyeron desde compuertas básicas.

6. Después de agregar las entradas, se usa la ventana **Stimulator Selection** para ligar las entradas a las teclas deseadas.

Nota: El teclado de la PC debe permanecer en modo minúculas al momento de ligar las entradas y para el proceso de simulación.

7. Luego, se debe dar click sobre el botón **Simulation Step** de la barra de herramientas de la ventana **Logic Simulator** para llevar a cabo un paso de la simulación con las entradas actuales.

8. Entonces se usan las teclas para cambiar las entradas de control y de temporización y hacer otro paso de la simulación al dar click nuevamente en el botón **Simulation Step**.

La simulación de formas de onda para el contador se muestra en la Figura 4.30.

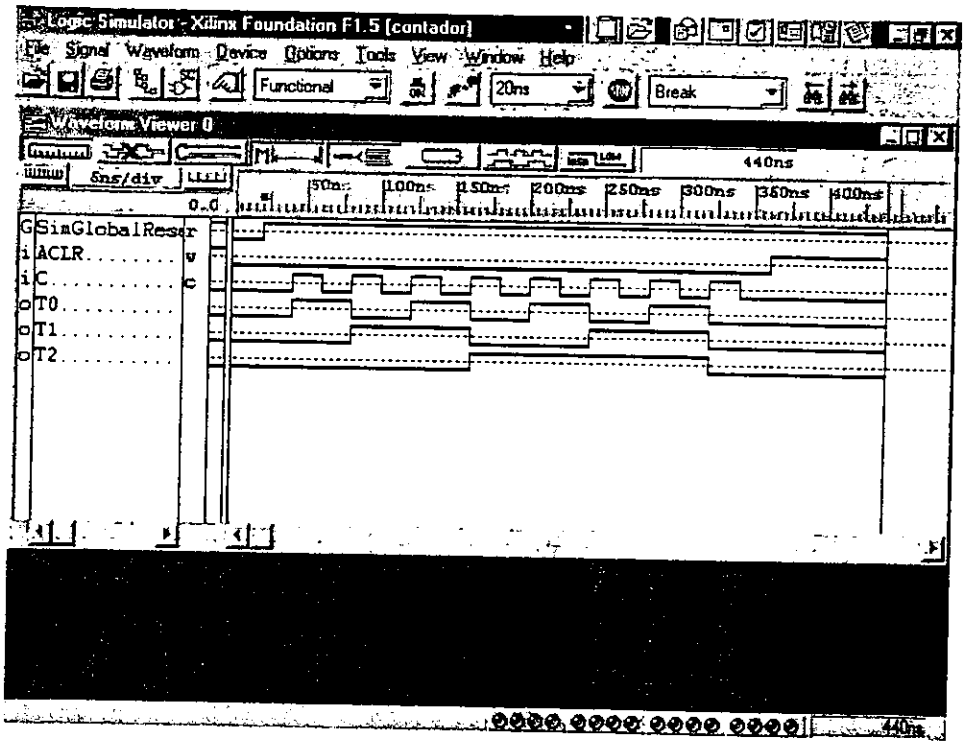


Figura 4.30: Simulación de Formas de Onda para el Contador Ascendente de 3 Bits con Flip – Flops Toggle.

Entonces una serie de flancos ascendentes en C desplaza los bits del contador a través de la secuencia 000 (0), 001 (1), 010 (2), 011 (3), 100 (4), 101 (5), 110 (6), y 111 (1). Entonces un 1 lógico en la entrada clear asíncrono lleva al contador a 000 (0). Todas las características del contador aparecen mientras esta funcionando.

9. Para transmitir el diseño, se crea un archivo de coacción de usuario tal como se describió en la sección 4.2.5 (Método 3) que asigne las entradas y salidas como se indica a continuación:

```
NET C       LOC=P44;   # argumento B0 de XSPORT.
NET ACLR    LOC=P45;   # argumento B1 de XSPORT.
NET T0      LOC=P24;   # segmento e del DISPLAY.
NET T1      LOC=P25;   # segmento d del DISPLAY.
NET T2      LOC=P26;   # segmento c del DISPLAY.
```

Para este experimento, la entrada de reloj, C, debe asignarse a B0 o B1. Estas entradas pasan a través de un buffer Schmitt – trigger en la Tarjeta XS40 para quitar cualquiera de los pequeños fallos ocasionados por las transiciones lentas de las señales del puerto paralelo de la PC. Usando cualquiera de las entradas B2-B7 se puede ocasionar que el contador funcione erróneamente.

10. A continuación, se debe compilar y transmitir el diseño lógico a la Tarjeta XS40 (Sección 4.2.6 y 4.2.8).

11. Finalmente, se debe de probar el diseño lógico (Sección 4.2.9) después de que es transmitido.

La siguiente secuencia de comandos después de la transmisión del diseño deberá ser el mismo conjunto de estados como en la simulación:

Comando.	ACLR.	C.	T2.	T1.	T0.
XSPORT 00	0	0	0	0	0
XSPORT 01	0	1	0	0	1
XSPORT 00	0	0	0	0	1
XSPORT 01	0	1	0	1	0
XSPORT 00	0	0	0	1	0
XSPORT 01	0	1	0	1	1
XSPORT 00	0	0	0	1	1
XSPORT 01	0	1	1	0	0
XSPORT 00	0	0	1	0	0
XSPORT 01	0	1	1	0	1
XSPORT 00	0	0	1	0	1
XSPORT 01	0	1	1	1	0
XSPORT 00	0	0	1	1	0
XSPORT 01	0	1	1	1	1
XSPORT 00	0	0	1	1	1
XSPORT 01	0	1	0	0	0
XSPORT 00	0	0	0	0	0
XSPORT 01	0	1	0	0	1
XSPORT 00	0	0	0	0	1
XSPORT 01	0	1	0	1	0
XSPORT 00	0	0	0	1	0
XSPORT 10	1	0	0	0	0

CONCLUSIONES

Al finalizar la presente tesis, se puede concluir que los FPGA's de la Serie XC4000, los cuales son parte de una amplia gama de dispositivos programables del mismo tipo, poseen una arquitectura programable compuesta de Bloques Lógicos Configurables (CLB's), interconectados por poderosos recursos de enrutamiento, y rodeados por Bloques de Entrada / Salida (IOB's). Además, de un alto grado de integración lo cual permite la implementación de diseños mucho más complejos, tales como el procesamiento digital de señales o las comunicaciones. A diferencia de sus antecesores los CPLD's, cuya arquitectura un tanto más simple y de menor grado de integración prácticamente sólo permite la implementación de diseños combinacionales o secuenciales.

También se establecieron las principales características arquitectónicas de ambos tipos de dispositivos lógicos, después de analizar por separado a cada uno de estos. Para los CPLD's XC9500 se tienen las siguientes características:

- Programación en Sistema.
- Grado de integración que va desde las 36 hasta las 536 macroceldas (800 a 1,200 compuertas lógicas utilizables).
- Soporte de herramientas CAD.
- Costo mínimo a menor volumen.
- Implementación de diseños combinacionales y secuenciales ideales para proyectos estudiantiles.

Y para los PFGA's XC4000:

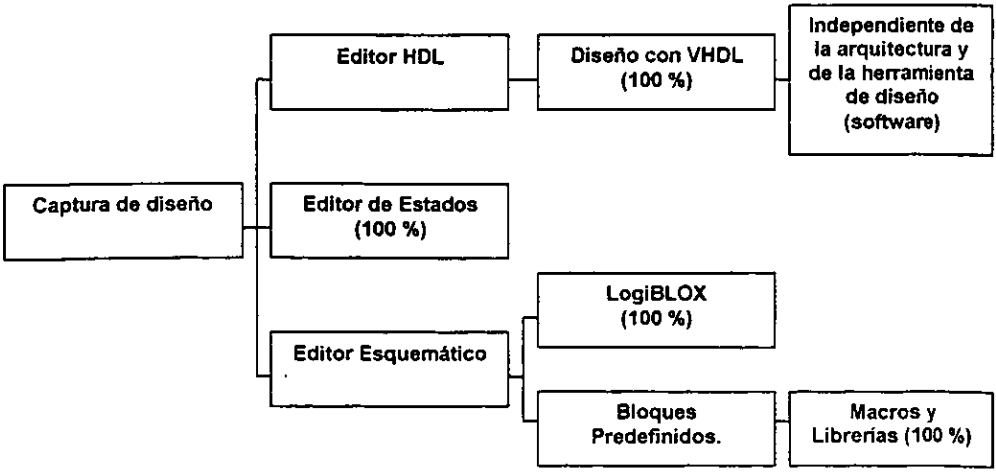
- Grado de integración que va desde las 2,000 hasta las 130,000 compuertas útiles.
- Reprogramabilidad ilimitada.
- Abundantes recursos de enrutamiento.

- Soporte de herramientas CAD mucho más versátiles.
- Costo menor a mayor volumen.
- Implementación de diseños de arquitectura computacional, procesamiento digital de señales y comunicaciones. A nivel escuela o industria.

Posteriormente fue descrito el proceso de diseño lógico programable (empleando el Software Xilinx Foundation Series 1.5.), el cual está compuesto por los siguientes puntos:

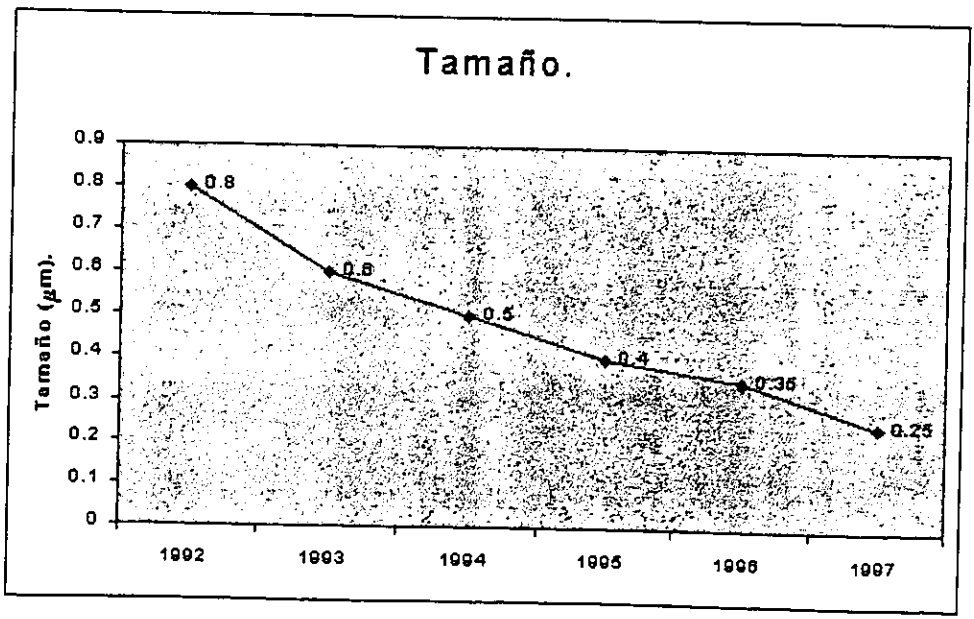
- Creación de un nuevo proyecto.
- Ingreso del diseño usando el Lenguaje de Descripción de Hardware ABEL o el Editor Esquemático.
- Simulación funcional del diseño (o una simulación temporizada para diseños secuenciales).
- Preparativos para compilar el diseño lógico.
- Compilación del diseño lógico.
- Transmisión del diseño lógico a la tarjeta.
- Prueba del diseño lógico después de que es transmitido.

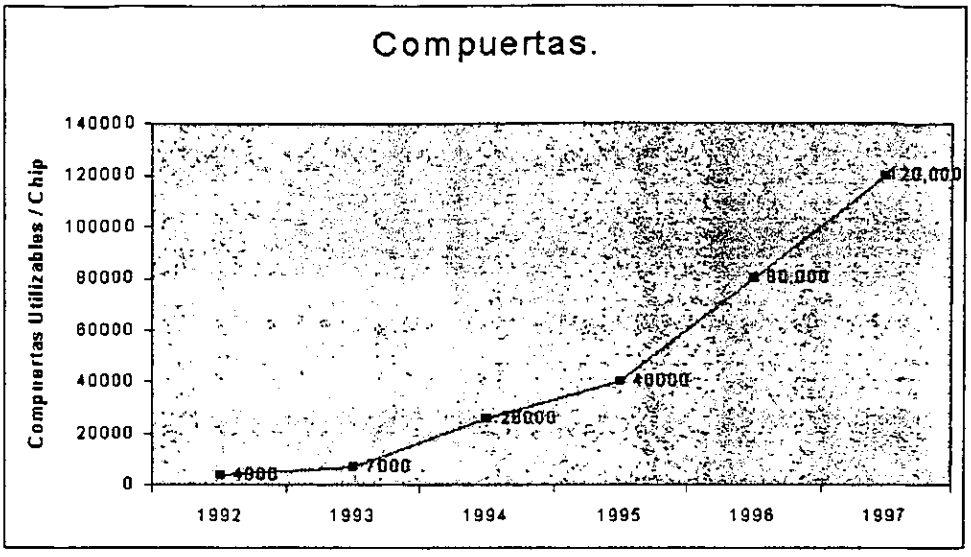
Se debe notar que en los pasos anteriores del proceso de diseño sólo se ocuparon las opciones fundamentales del software para poder llevar a cabo la implementación de un diseño; pero existe un gran número de opciones de las que se sugiere su investigación posterior para aplicaciones de diseño de sistemas más grandes y complicados, en las siguientes áreas:



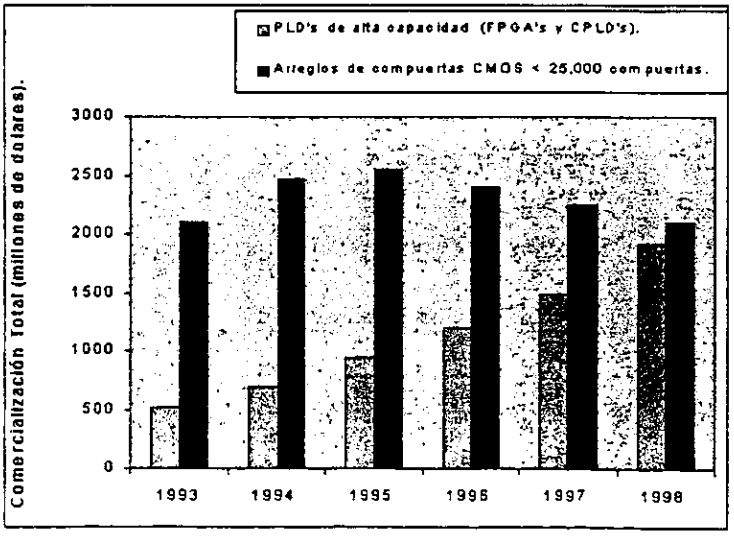
En este caso, se plantearon algunas aplicaciones sencillas con el fin de poder entender el proceso de una manera más clara, pero existen varias aplicaciones (transmisión de datos, procesamiento digital de señales, filtros digitales, etc.) para las cuales se puede emplear éste; ya que las bases aquí presentadas así lo permiten.

En fin, se ha procurado dar el máximo de información acerca de los FPGA's, así como las bases principales para el proceso de diseño, con el propósito de tener una idea bastante precisa de lo que es posible implementar con estos dispositivos. Sin embargo, la tecnología avanza rápidamente (actualmente ya existen FPGA's analógicos). Las siguientes gráficas muestran como cada vez se reducen todavía más los tamaños de encapsulado y a la vez integran un mayor número de compuertas utilizables.





Además, la comercialización de este tipo de dispositivos lógicos programables (FPGA's y CPLD's) cada vez es mayor con respecto a los demás arreglos de compuertas (Ver la siguiente gráfica), por lo que se recomienda ampliar la información para futuras implementaciones.



Configuración de la Tarjeta.

1. TABLA DE DESCRIPCION DE PINES DE LA TARJETA XS40.

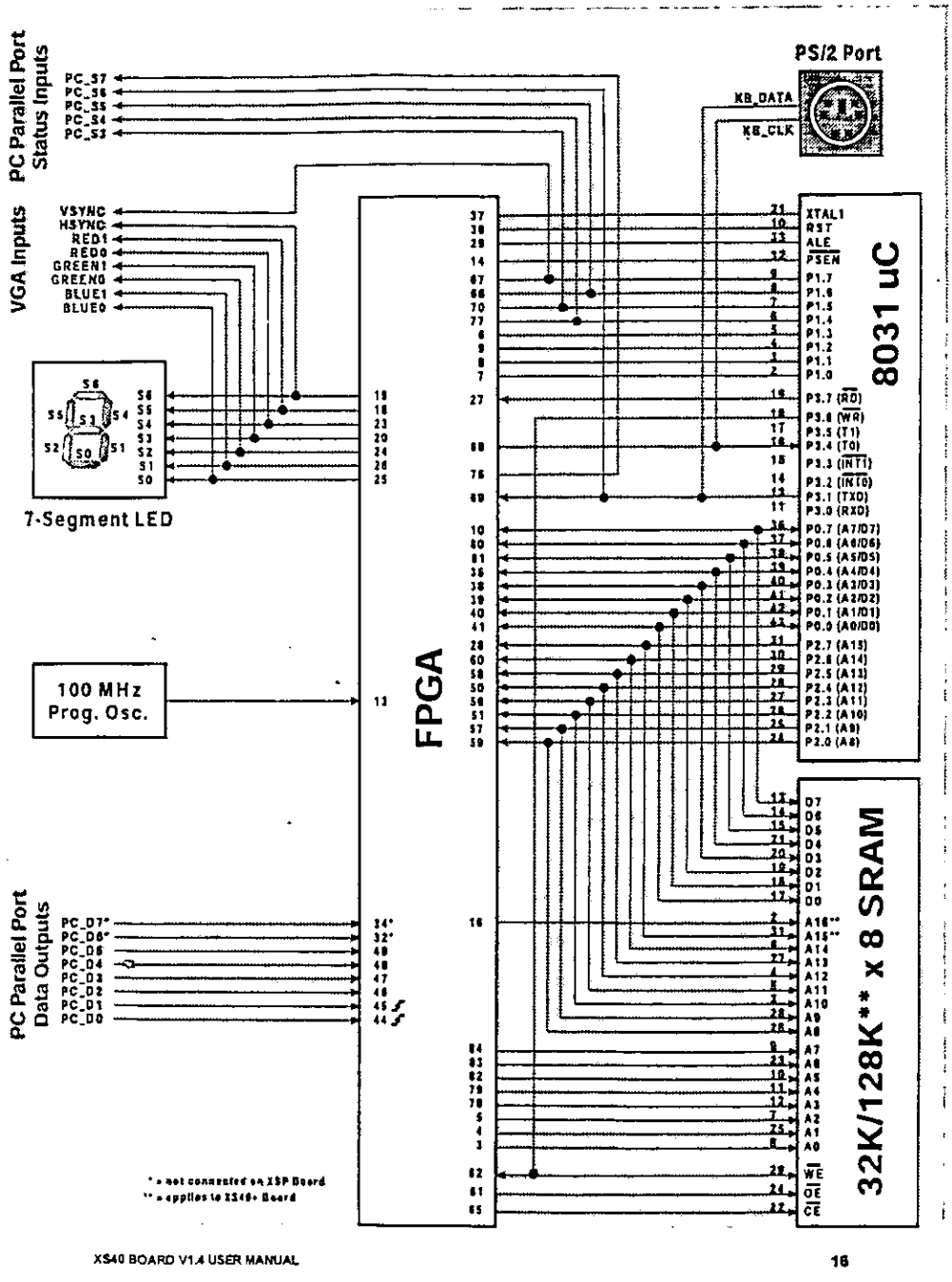
A continuación se muestra una tabla que contiene la descripción de los pines de la Tarjeta XS40 presentada en el manual de usuario V 1.4. Se incluye un diagrama esquemático de la Tarjeta.

Descripción de los pines de la Tarjeta XS40.

Pin XS40	Conexión a...	Descripción.
25	S0, AZUL0 (d)	Estos pines manejan los segmentos individuales del DISPLAY (S0-S6). También manejan el color y las señales de sincronización horizontal para un monitor VGA.
26	S1, AZUL1 (c)	
24	S2, VERDE0 (e)	
20	S3, VERDE1 (g)	
23	S4, ROJO0 (b)	
18	S5, ROJO1 (f)	
19	S6, HSYNCB (a)	
13	CLK	Este pin es una entrada manejada por el oscilador programable de 100 MHz.
44	PC D0	Estos pines son manejados por los pines de salida de datos del puerto paralelo de la PC. Las señales temporizadas son fiables únicamente cuando se aplican a través de los pines 44 y 45 ya que estos tienen un circuito de histéresis adicional. Los pines 32 y 34 son señales de modo para el FPGA que se deben ajustar al diseño con las herramientas del Fundación que manejan estos pines. Los pines 32 y 34 no son utilizados como I / O de propósito general en el FPGA Spartan sobre la Tarjeta XSP.
45	PC D1	
46	PC D2	
47	PC D3	
48	PC D4	
49	PC D5	
32	PC D6	
34	PC D7	

Pin XS40	Conexión a...	Descripción.
37	XTAL 1	Este pin maneja la entrada de reloj del μC.
36	RST	Este pin maneja la entrada de "reset" del μC.
29	ALEB	Este pin controla la habilitación tipo latch de dirección del μC.
14	PSENB	Este pin controla la habilitación de guardado de programa del μC.
7	P1.0	Estos pines están conectados a los pines del Puerto 1 del μC. Algunos de los pines también están conectados a los pines de entrada de estado del puerto paralelo de la PC. El pin 67 maneja la señal de sincronización vertical para un monitor VGA.
8	P1.1	
9	P1.2	
6	P1.3	
77	P1.4, PC S4	
70	P1.5, PC S3	
66	P1.6, PC S5	
67	P1.7, VSYNCB	
69	P3.1(TXD), PC_S6	Estos pines están conectados a algunos de los pines del Puerto 3 del μC. Las funciones especializadas del μC para cada pin del puerto están indicadas entre paréntesis. El pin 62 esta conectado al pin de escritura de datos del μC y al pin de habiltación de escritura de la SRAM. El pin 69 está conectado al pin de entrada de estado del puerto paralelo de la PC y a la línea de datos en las PS / 2. El pin 68 está conectado a la línea de reloj en las PS / 2.
68	P3.4(T0), PS/2 CLK	
62	P3.6(WRB), WEB	
27	P3.7(RDB)	
41	P0.0(AD0), D0	Estos pines están conectados al Puerto 0 del μC el cual también es un puerto de dirección / datos multiplexado. Estos pines también están conectados a los pines de datos de la SRAM.
40	P0.1(AD1), D1	
39	P0.2(AD2), D2	
38	P0.3(AD3), D3	
35	P0.4(AD4), D4	
81	P0.5(AD5), D5	
80	P0.6(AD6), D6	
10	P0.7(AD7), D7	

Pin XS40	Conexión a...	Descripción.
59	P2.0(A8), A8	Estos pines están conectados al Puerto 2 del μC y también a las salidas del byte de direcciones superior. Estos pines también están conectados a los bits de dirección superiores de la SRAM. Los pines 28 y 16 están conectados a los pines de dirección de la SRAM de 128 KB únicamente en la Tarjeta XS40+. Los pines 28 y 16 no están conectados a la SRAM de 32 KB en la Tarjeta XS40.
57	P2.0(A9), A9	
51	P2.0(A10), A10	
56	P2.0(A11), A11	
50	P2.0(A12), A12	
58	P2.0(A13), A13	
60	P2.0(A14), A14	
28	P2.0(A15), A15	
16	A16	
3	A0	Estos pines manejan los 8 bits de dirección inferiores de la SRAM.
4	A1	
5	A2	
78	A3	
79	A4	
82	A5	
83	A6	
84	A7	
61	OEB	Este pin maneja la habilitación de salida de la SRAM.
65	CEB	Este pin maneja la habilitación del chip de la SRAM.
75	PC_S7	Este pin maneja a un pin de entrada de estado del puerto paralelo de la PC.


PC Parallel Port
Status Inputs
PC_S7
PC_S6
PC_S5
PC_S4
PC_S3
PS/2 Port
KB_DATA
KB_CLK
VGA Inputs
VSYNC
HSYNC
RED1
RED0
GREEN1
GREEN0
BLUE1
BLUE0
7-Segment LED
100 MHz
Prog. Osc.
FPGA
8031 uC
32K/128K** x 8 SRAM
PC Parallel Port
Data Outputs
PC_D7*
PC_D6*
PC_D5
PC_D4
PC_D3
PC_D2
PC_D1
PC_D0
* = not connected on XSP Board
** = applies to XS40+ Board
XS40 BOARD V1.4 USER MANUAL
16

2. INSTALACION DE LA TARJETA XS40.

2.1 Advertencia.

La Tarjeta XS40 requiere de una fuente de alimentación para operar. Esta tarjeta no obtiene potencia a través del cable de transmisión del puerto paralelo de la PC.

2.2 Lista de Componentes.

- Una Tarjeta XS40.
- Un cable de 6' con un conector macho DB-25 en cada extremo.
- Un disco flexible con el software de utilidades para usar la Tarjeta XS40 y documentación.

2.3 Instalación de las Herramientas de Software para la Tarjeta XS40.

Xilinx actualmente provee las herramientas XACTstep F1 para programar sus FPGA's. Cualquier versión reciente del software de XILINX deberá generar los archivos de configuración de cadena de bits que son compatibles con la Tarjeta XS40. Siga las instrucciones que XILINX provee para instalar su software.

XESS Corp. provee un software de utilidades adicional para establecer la interfaz entre la PC y la Tarjeta XS. Simplemente se activa el programa SETUP.EXE del disco de 3.5" para instalar estas herramientas.

Una vez que las herramientas del software adicional se instalan, se verán los siguientes subdirectorios:

XSTOOLS \ BIN: Contiene los programas ejecutables para transmitir a la Tarjeta XS40 y para aplicar las señales correspondientes a la Tarjeta XS40 mediante el puerto de la impresora. Se incluye también un ensamblador para el microcontrolador.

XSTOOLS \ DOCS: Contiene la documentación y esquemáticos para la Tarjeta XS40.

2.4 Instalación de la Tarjeta XS40.

Operación.

Se pueden usar todas las Tarjetas XS40 para experimentar con los diseños basados en FPGA's XC4000. Simplemente se pone la Tarjeta XS40 sobre una superficie "no conductora". Entonces se aplica potencia al conector J9 de la tarjeta desde una fuente de poder de 9 V. CD. con un conector hembra de 2.1 mm.; conector central positivo. El circuito de regulación de voltaje en la tarjeta creará los voltajes requeridos para el resto de los circuitos.

Instalación de la Tarjeta de Prueba.

Las dos filas de pines de la Tarjeta XS40 pueden insertarse en una tarjeta de prototipo (protoboard). Una vez insertada, todos los pines del FPGA y los del microcontrolador 8031 son accesibles a otros circuitos sobre la tarjeta de prototipo (ver el esquemático al final del Apéndice para establecer la relación entre los pines del FPGA y del microcontrolador con la Tarjeta XS40). Los números impresos cerca de las filas de pines en la Tarjeta XS40 corresponden a los números de pin del FPGA.

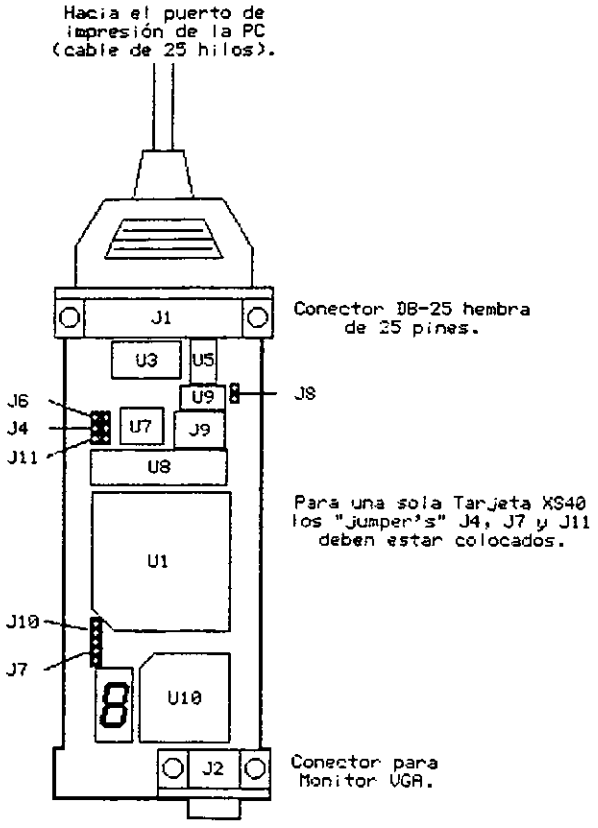
La potencia puede ser suministrada a la Tarjeta XS40 a través del conector J9, o puede aplicarse directamente a través de los pines de esta. Simplemente se deben conectar a +5 V. y +3.3 V. los siguientes pines de VCC de la Tarjeta XS40, y conectar a tierra el pin GND (se necesitan +3.3 V. si la tarjeta contiene un FPGA de tipo XC4000XL).

Conexión de Potencia y Tierra para Todos los Tipos y Versiones de Tarjetas XS40.

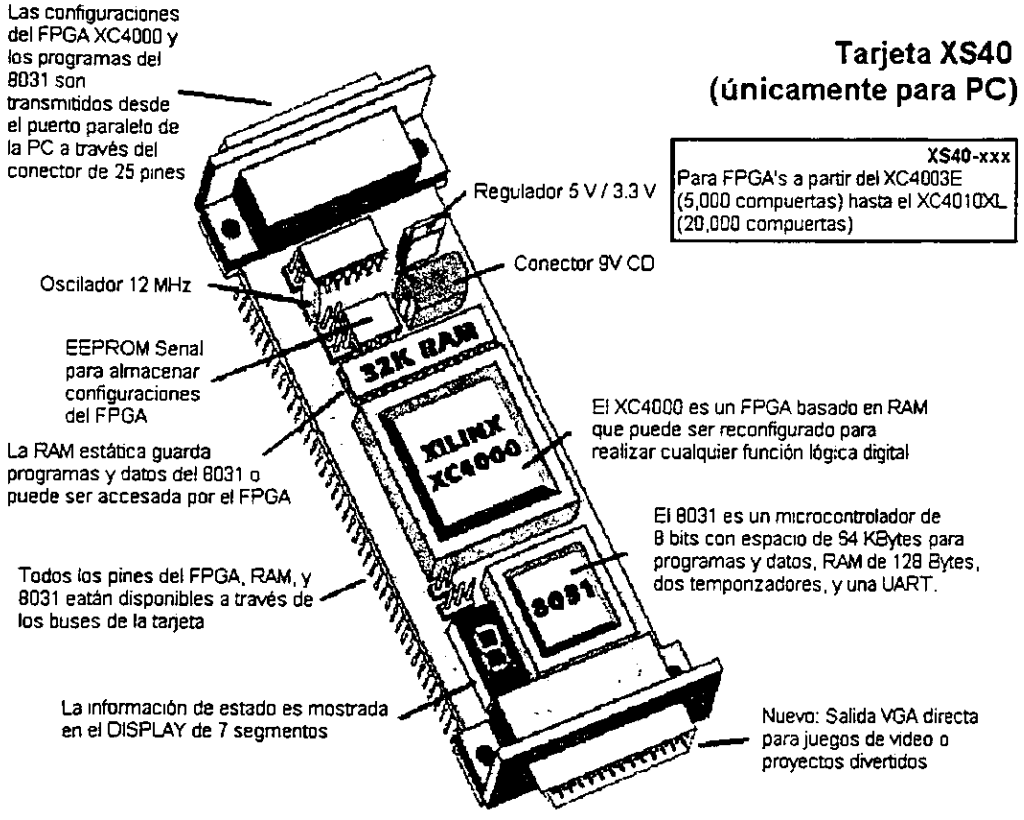
Tipo de Tarjeta XS40.	Pin GND.	Pin +5V.	Pin +3.3V.
XS40-005E V 1.0	52	2	Ninguno
XS40-005E V 1.1	52	2, 54	Ninguno
XS40-005E V 1.2	52	2, 54	Ninguno
XS40-005XL V 1.0	52	Ninguno	54
XS40-005XL V 1.1	52	2	54
XS40-005XL V 1.2	52	2	54
XS40-010E V 1.0	52	2	Ninguno
XS40-010E V 1.1	52	2, 54	Ninguno
XS40-010E V 1.2	52	2, 54	Ninguno
XS40-010XL V 1.0	52	Ninguno	54
XS40-010XL V 1.1	52	2	54
XS40-010XL V 1.2	52	2	54

2.5 Conexión PC – Tarjeta XS40.

El cable de 6' incluido con la Tarjeta XS40 se conecta al puerto paralelo de la PC. Un extremo del cable se conecta al puerto de la impresora y el otro al conector DB-25 hembra (J1) en la parte superior de la tarjeta como se muestra en la siguiente Figura (se incluye también un diagrama descriptivo de los componentes de la Tarjeta XS40).



Tarjeta XS40 V 1.2.


Las configuraciones del FPGA XC4000 y los programas del 8031 son transmitidos desde el puerto paralelo de la PC a través del conector de 25 pines
Tarjeta XS40 (únicamente para PC)
XS40-xxx
Para FPGA's a partir del XC4003E (5,000 compuertas) hasta el XC4010XL (20,000 compuertas)
Regulador 5 V / 3.3 V
Conector 9V CD
Oscilador 12 MHz
EEPROM Serial para almacenar configuraciones del FPGA
32K RAM
La RAM estática guarda programas y datos del 8031 o puede ser accesada por el FPGA
XILINX XC4000
El XC4000 es un FPGA basado en RAM que puede ser reconfigurado para realizar cualquier función lógica digital
El 8031 es un microcontrolador de 8 bits con espacio de 64 KBytes para programas y datos, RAM de 128 Bytes, dos temporizadores, y una UART.
8031
Todos los pines del FPGA, RAM, y 8031 están disponibles a través de los buses de la tarjeta
La información de estado es mostrada en el DISPLAY de 7 segmentos
Nuevo: Salida VGA directa para juegos de video o proyectos divertidos

2.6 Configuración de "Jumper's" de la Tarjeta XS40.

Jumper	Posición.	Propósito.
J4	**Encendido (Valor por omisión)**	Se debe poner si se transmite a la tarjeta XS40 a través del puerto paralelo.
	Apagado	Se debe quitar si la Tarjeta XS40 está siendo configurada desde la EEPROM serial en la tarjeta (U7).
J5 (absent on V1.2 of XS40)	**Encendido (Valor por omisión)**	Se debe poner si se usa una sola Tarjeta XS40 o si ésta es la última tarjeta en una cadena de Tarjetas XS40.
	Apagado	Se debe quitar en todas las tarjetas, excepto la última, en una cadena de Tarjetas XS40.
J6	Encendido	Se debe poner cuando la EEPROM serial de la tarjeta (U7) está siendo programada.
	Apagado (Valor por omisión)	Se debe quitar durante el uso normal de la tarjeta.
J7	**1-2 (ext) (Valor por omisión)**	Se debe poner en los pines 1 y 2 (ext) si el programa del microcontrolador 8031 es almacenado en la RAM de 32 Kbytes externa (U8) de la Tarjeta XS40.
	2-3 (int)	Se debe poner en los pines 2 y 3 (int) si el programa es almacenado internamente en el chip 8031.
J8	Encendido	Se debe poner en las Tarjetas XS40 que usan los FPGA's de tipo XC4000XL de 3.3 V.
	Apagado	Se debe quitar en las Tarjetas XS40 que usan los FPGA's de tipo XC4000E de 5 V.

Jumper	Posición.	Propósito.
J10	Encendido	Se debe poner si la Tarjeta XS40 está siendo configurada desde la EEPROM serial en la tarjeta.
	Apagado (Valor por omisión)	Se debe quitar si la Tarjeta XS40 está siendo configurada desde el puerto paralelo de la PC.
J11	**Encendido (Valor por omisión)**	Se debe poner si la Tarjeta XS40 está siendo configurada desde el puerto paralelo de la PC.
	Apagado	Se debe quitar si la Tarjeta XS40 está siendo configurada desde la EEPROM serial en la tarjeta.

Una vez que la Tarjeta XS40 ha sido instalada y los "jumper's" están en su configuración predefinida, se puede probar la tarjeta usando uno de los siguientes comandos (se debe estar en el directorio XSTOOLS \ BIN para correr el comando XSTEST):

Comandos para la Prueba de Varios Tipos de Tarjetas XS40.

Tipo de Tarjeta XS40.	Comando Test.
XS40-005E	XSTEST XS40-005E
XS40-005XL	XSTEST XS40-005XL
XS40-010E	XSTEST XS40-010E
XS40-010XL	XSTEST XS40-010XL

El programa de prueba del FPGA, carga a la RAM con un programa de prueba para el microcontrolador, y entonces el microcontrolador ejecuta este programa. El período total de prueba (incluyendo la programación de la tarjeta) es de 20 segundos para una Tarjeta XS40. Si la prueba se completa exitosamente, entonces se verá un **0** en el DISPLAY de la tarjeta.

Sin embargo, si el programa de prueba detecta un error, entonces el DISPLAY muestra una **E** o permanece en blanco. En este caso, se deben verificar los siguientes aspectos:

- Comprobar que la tarjeta realmente recibe potencia de la fuente de alimentación de 9 V. CD. a través del conector J9 o a través de los pines VCC y GND.

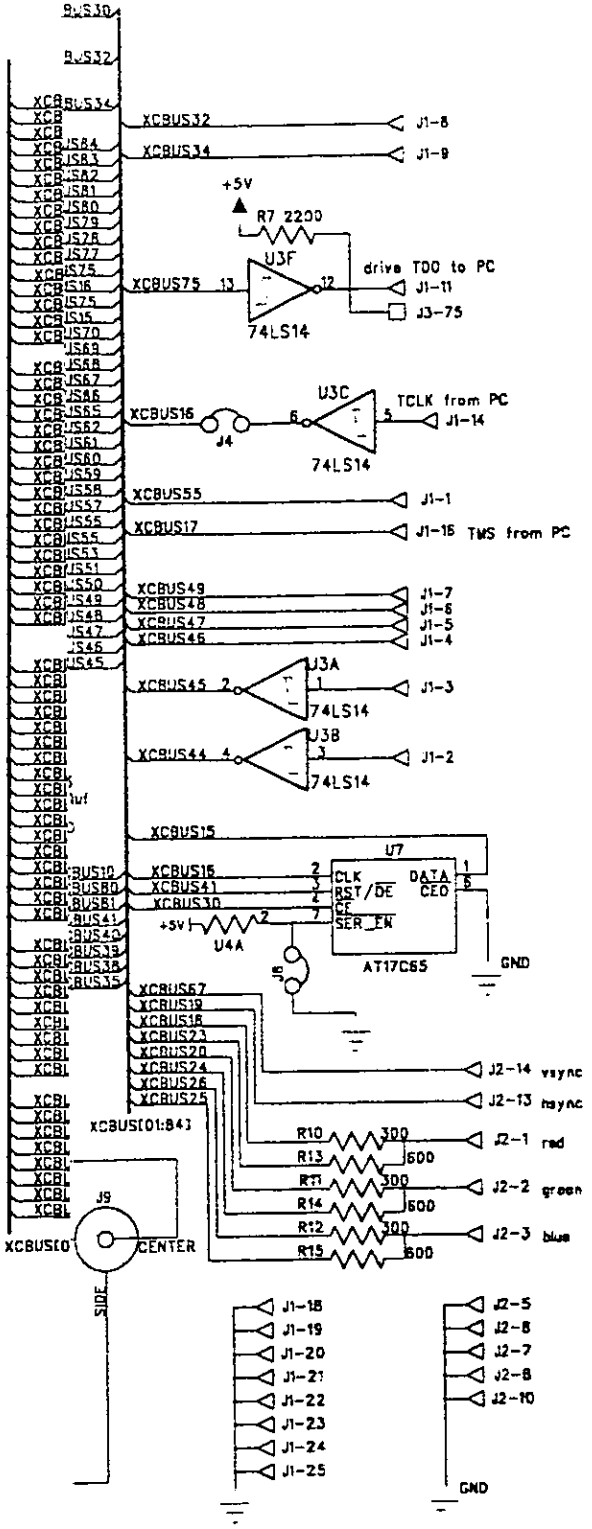
- Verificar que la tarjeta esté sobre una superficie "no conductora" y que no existan conexiones en ninguno de los pines (a excepción de los pines VCC y GND si esta es la forma de aplicación de potencia a la tarjeta).

- Verificar que los "jumper's" están en su modo de configuración predefinida.

- Comprobar que el cable de transmisión está firmemente conectado a la Tarjeta XS40 y al puerto paralelo de la PC.

- Verificar que el puerto paralelo está en modo SPP (el modo comúnmente se establece en el BIOS como SPP, EPP, o ECP. SPP es el modo menor y más seguro).

Si después de verificar los aspectos anteriores no existe ningún error, entonces se debe probar la tarjeta usando otra PC. Por experiencia, el 99.9% de todos los problemas se deben al puerto paralelo.


XCBUS32
XCBUS34
J1-8
J1-9
+5V
R7 2200
U3F
drive TDO to PC
XCBUS75
J1-11
J3-75
74LS14
U3C
TCLK from PC
XCBUS16
J4
J1-14
74LS14
XCBUS55
J1-1
XCBUS17
J1-16
TMS from PC
XCBUS49
XCBUS48
XCBUS47
XCBUS46
J1-7
J1-6
J1-5
J1-4
U3A
XCBUS45
J1-3
74LS14
U3B
XCBUS44
J1-2
74LS14
1uf
XCBUS15
U7
XCBUS10
XCBUS40
XCBUS41
XCBUS30
CLK
RST/DE
CE
SER_EN
DATA
CEO
+5V
U4A
J6
AT17C65
GND
XCBUS67
XCBUS19
XCBUS18
XCBUS23
XCBUS20
XCBUS24
XCBUS26
XCBUS25
J2-14 vsync
J2-13 hsync
XCBUS[01:84]
R10 300
R13 600
R11 300
R14 600
R12 300
R15 600
J2-1 red
J2-2 green
J2-3 blue
J9
CENTER
SIDE
XCBUS00
J1-18
J1-19
J1-20
J1-21
J1-22
J1-23
J1-24
J1-25
J2-5
J2-6
J2-7
J2-8
J2-10
GND

APENDICE B

Glosario.

ABEL: Lenguaje de Expresión Booleana Avanzado. Un lenguaje de diseño universal para PLD's apoyado en herramientas de software de I / O de Datos.

Antifusible: Lo contrario a un fusible; un circuito abierto que se programa para ser una resistencia baja permanentemente (Esta es una palabra generalmente con muchos significados, pero éste es bastante específico cuando se usa con PLD's).

Arquitectura: Dentro del campo de los PLD's, esto significa un sistema de organización lógica de elementos junto con un sistema de organización de interconexiones programables entre esos elementos lógicos. Los esquemas arquitectónicos de enrutamiento y la organización de las celdas lógicas normalmente intentan optimizar la velocidad y la capacidad funcional dentro de los requisitos de las tecnologías físicas del dispositivo usadas para fabricar PLD's en una modalidad de propósito general, para cubrir completamente el mayor número de posibles aplicaciones. Sin embargo, algunas arquitecturas proporcionan las funciones elementales, balancean la velocidad y la capacidad funcional para proporcionar los mejores resultados para tipos particulares de aplicaciones cómo máquinas de estado, decodificación, acumulación, registro "rich" u otros tipos de aplicaciones.

Arreglo – AND: Arreglo bidimensional de señales lógicas de entrada que se combinan en sus puntos de cruce mediante circuitos AND. Cada punto de cruce por lo tanto provee dos entradas a la función AND.

Arreglo – NOR: Dos arreglos dimensionales de entradas de señales lógicas a través de las que se combinan sus puntos de cruce mediante circuitos NOR. Cada punto de cruce por lo tanto provee entradas de función NOR.

Arreglo – OR: Un arreglo bidimensional de señales lógicas de entrada que se combinan en sus puntos de cruce a través de los circuitos OR. Cada punto de cruce por lo tanto provee dos entradas de función OR.

Basado en RAM: Un PLD reprogramable en el que la tecnología de programación es igual a la usada para los CI's de memoria volátil, comúnmente SRAM's.

"Bitstream": Cadena de bits usada para programar un FPGA basado en SRAM.

Bloque: Elemento funcional básico programable en una arquitectura de PLD, el bloque comprende una colección de los elementos lógicos con interconexiones programables entre ellos. El dispositivo total se compone de unos o varios bloques, nuevamente, rodeado por interconexiones programables.

Bus Global: Un conjunto de líneas paralelas interconectadas que corren a lo largo de un chip y está disponible para conectar a todos los elementos lógicos.

Campo programable: Esto indica un dispositivo que puede programarse en el laboratorio de Ingeniería de sistemas en lugar de en la fábrica del productor de circuitos integrados. Los PLD's son campos programables, considerando que los arreglos de compuertas son programados de fábrica.

Captura Esquemática: Método gráfico de captura de diseño.

Celda: Elemento funcional básico programable en una arquitectura PLD, igual que "bloque".

Celda I / O: Celdas Lógicas que reciben las señales de entrada desde los pines del dispositivo o proporcionan señales de salida a los pines del dispositivo o proporcionan ambas funciones.

CLB: Bloque Lógico Configurable. Agrupamiento de los elementos lógicos usados en arquitecturas de FPGA.

Colocación y Enrutamiento: La determinación de la colocación física de celdas y la configuración de las interconexiones en CI's.

Compiladores de PLD: Herramientas de diseño que soportan una variedad de técnicas de captura de diseño y a veces incluyen capacidades de síntesis lógica.

Compuerta Equivalente: Normalmente, una compuerta NAND de dos entradas. Este término se ha usado como una celda lógica elemental en arreglos de compuertas. La capacidad funcional de los arreglos de compuertas se expresa normalmente en términos de compuertas equivalentes. Sin embargo, las diferentes arquitecturas hacen esta medida muy inexacta comparando las capacidades de uno a otro dispositivo.

Conectividad Global: Un aspecto arquitectónico en un CPLD que permite que toda entrada, I / O y señales sean usadas en todos los términos producto de todas las macroceldas a la vez.

Control "Slew – Rate" de Salida: Una característica en muchos CPLD's que permiten que el "Slew – Rate" de salidas especificadas pueda ser controlado. Esto le permite al usuario perfeccionar la relación entre la velocidad y las distorsiones de señal asociadas con transiciones de salida.

CPLD: PLD Complejo. Término común para un PLD compuesto de muchos bloques donde la arquitectura de los bloques es muy semejante a la de los PAL's.

Descripción del comportamiento: Técnicas de Captura de Diseño que describen el diseño desde el punto de vista de su comportamiento. El nivel de comportamiento de las herramientas de captura de diseño comúnmente permite a los diseñadores describir el diseño de CI de lógica programable desde el punto de vista de tablas de verdad, formas de onda, diagramas de estado, ecuaciones booleanas, o ecuaciones de alto nivel. Este término es opuesto a la "Descripción Estructural". Las técnicas de Síntesis se usan entonces frecuentemente para perfeccionar esta descripción y lo preparan para una implementación física.

Diseño Estructural: Una de las dos clases amplias de métodos de captura de diseño, el otro siendo de comportamiento, en que el diseño se expresa desde el punto de vista de su estructura, más bien que su comportamiento. Un esquema es una descripción estructural mientras una forma de onda es una descripción del comportamiento, por ejemplo.

EDA: Automatización de Diseño Electrónica. Nombre genérico para el software que ayuda en el diseño de circuitos electrónicos.

EDIF: Formato Electrónico de Intercambio de Datos. Un formato estándar usado para transferir datos de diseño.

EEPLD: Un PLD que usa celdas de memoria EEPROM para retener la configuración lógica programable (pero es de una complejidad mucho más alta que los dispositivos PLD Simples).

Elemento Programable: Los interruptores o los fusibles que realizan la interconexión entre los elementos lógicos en un PLD.

Enrutamiento: El acto de interconectar los elementos lógicos que han sido elegidos por la colocación de la función de circuito deseada en un PLD.

Ensambladores de PLD: Un tipo de herramienta de diseño que puede convertir las ecuaciones de diseño, de suma de productos en datos que programan fusibles.

Entrada Dedicada: Un pin de un PLD que no puede usarse como una salida.

EPLD: Un PLD que usa celdas de memoria EPROM en lugar de fusibles, antifusibles o RAM controlada por transistores de paso para retener la configuración lógica programada. Los EPLD's pueden ser pequeños o grandes en su capacidad funcional, pero alguna confusión los ha igualado inadecuadamente con los CPLD's, porque EPLD fue el primer término que se popularizó para los CPLD's.

Flash: Una tecnología de memoria de silicio no volátil usada en muchos dispositivos PLD's y CPLD's. La tecnología permite una confiabilidad más alta y disminuye el costo de los dispositivos. Los dispositivos son eléctricamente reprogramables y borrables.

FLEX: Una arquitectura FPGA de Altera.

Formato "Berkeley" PLA: Formato desarrollado por la Universidad de California como medio de transferencia de datos de diseño entre diferentes herramientas de síntesis.

FPGA: Arreglo de Compuertas Programables de Campo. Un PLD que contiene elementos lógicos o celdas que se interconectan mediante canales de señal por interconexiones programables. Las celdas pueden variar en complejidad desde una sola compuerta en algunas arquitecturas a funciones programables que son del tamaño de los dispositivos PAL más grandes. Este tipo de

arquitectura produce diversos resultados estadísticamente en desempeño y capacidad funcional a causa de las muchas opciones para escoger celdas y rutas de interconexión, así como en los arreglos de compuertas. Los medios de programación en un FPGA significan que pueden ser reprogramables (como las SRAM's) o los OTP (como los antifusibles y fusibles).

FPLA: Arreglo Lógico Programable de Campo. Un PLD con ambos términos programables AND y OR, pero en la gama de complejidad de los PLD's Simples.

Fusible: Un elemento de circuito de baja resistencia que puede programarse para llegar a ser un circuito abierto. La programación se llama "Quemar" el fusible y normalmente se hace térmicamente, por oleajes de corrientes relativamente altas.

GAL: Lógica de Arreglo Genérico. Nombre de los productos de Lattice Semiconductor. Dispositivos basados en EEPROM que son compatibles pin a pin con los populares dispositivos de tipo PAL.

HDL: Lenguaje de Descripción de Hardware. Un lenguaje para describir generalmente circuitos lógicos.

Interconexiones: Los medios físicos que realizan conexiones entre las señales de entrada y las salidas de bloques lógicos y entre las señales de entrada y salidas de los elementos lógicos dentro de los bloques lógicos. Estas interconexiones son programables. El término no se refiere a las conexiones hechas permanentemente en la fábrica entre los bloques y los elementos lógicos.

IOB: Bloque de Entrada / Salida. Un bloque lógico en lógica programable que puede configurarse para ser una entrada o una salida o pin bidireccional I / O y puede programarse para proveer un gran número de funciones lógicas y opciones de registro.

IOLMC: Macrocelda Lógica de Entrada / Salida.

JEDEC: Consejo de Ingeniería de Dispositivos Electrón de Juntura. Una organización de estándares que es parte de la Asociación de la Industria de la Electrónica (EIA).

JEDEC Estándar 3: Un formato estándar usado para transferir datos del dispositivo de programación a los programadores de dispositivo.

JTAG: Grupo de Acción de Prueba de Juntura. Una interfaz serial estándar usada como prueba e interfaz de programación en ISP – PLD's.

Lógica Cache FPGA: Un FPGA que puede ser totalmente o parcialmente reconfigurado en sistema, es decir sobre la marcha, sin la pérdida de los datos almacenados.

Macrocelda: a) Es un grupo de celdas lógicas que desarrollan una función específica que se define en una biblioteca de funciones en el sistema de diseño. Una macrocelda puede ser "dura" porque su configuración se fija dondequiera que se ponga en el dispositivo. O, puede ser "suave" porque su topología varía dependiendo de donde se ponga en el dispositivo. b) Bloque lógico fundamental de SPLD o CPLD, puede estar en una salida o alambrado.

Macrocelda de Salida: Una función de la Macrocelda que provee una señal a un pin de salida.

Matriz de Interconexión: Matriz de Interconexión de Propósito General usada por Xilinx.

Netlist: Una descripción simbólica para las conexiones indicadoras esquemáticas entre los elementos lógicos, normalmente expresado en formato EDIF.

OLMC: Macrocelda Lógica de Salida. Término usado por Lattice Semiconductor para designar una característica importante de su familia de GAL's y de PLD's. Vea Macrocelda de Salida

OrCAD / PLD: Sistema de diseño PLD de OrCAD que se apoya en ecuaciones, tablas de verdad, descripciones de estado, y técnicas esquemáticas de entrada.

OTP: Programable "Sólo una vez". Un término que se refiere a dispositivos basados en fusible que no pueden borrarse, y por consiguiente, sólo pueden usarse una sola vez.

PAL: Lógica de Arreglo Programable. Una arquitectura PLD que simplificó las PLA's fijando el arreglo OR y manteniendo el arreglo AND programable. Una marca registrada por AMD pero originada por Monolithic Memories Inc.

PAL Configurable: Un tipo avanzado de PAL, tal como los 22V10, que tienen circuitos de salida configurable.

PALASM: Ensamblador PAL. Herramientas de software desarrolladas por MMI para sus dispositivos PAL, ahora propiedad de AMD.

PCI: Componente Periférico de Interconexión. Especificación de interconexión de Bus estándar de la industria.

PLA: Arreglos Lógicos Programables. Una arquitectura que usa un arreglo AND programable en serie con un arreglo OR programable.

PLD: Un dispositivo lógico que puede ser configurado por un diseñador de sistemas en el laboratorio con medios fácilmente disponibles. Este es un término general aunque se use frecuentemente para indicar únicamente pequeños PLD's tales como PAL's y FPLA's.

PLD Simple: Un PLD con la complejidad funcional de los dispositivos PAL populares, alcanzando hasta unos cientos de compuertas equivalentes. Estos incluirían dispositivos con nombres tales como PAL, GAL, FPLA o PLS y se alojarían en paquetes con hasta 28 pines. Muchas veces el término general "PLD" se usa para PLD's simples.

Posicionado: El acto de distribuir el circuito deseado mediante los recursos lógicos del PLD.

Producto de Sumas: Una expresión lógica que describe la salida de un arreglo de compuertas OR seguida por un arreglo de compuertas AND.

Programable: Tener la capacidad de ser configurado, o para ser programado a una configuración deseada.

PROM: Memoria de Sólo Lectura Programable

Prueba "Boundary – Scan": Tendencia que emerge en la prueba ha nivel de tarjeta. Proporciona pruebas y costos de fabricación más inferiores principalmente para dispositivos de alto número de pines debido al silicio por arriba del requerido para implementar la tecnología.

RAM de Usuario: RAM que puede ser accesada por el usuario mediante la implementación lógica (incluye los tipos síncronos, asíncronos y puerto simple / doble).

Referencia: Así como el término "Arquitectura" tiene muchos significados, "Referencia" también tiene muchos significados en general, pero llega a ser muy específico cuando se usa en diferentes campos de aplicación de tecnologías. En el ámbito de PLD, el término "referencia" se usa normalmente para indicar la configuración de circuito o una función la cual se usa para medir y comparar el desempeño funcional de diferentes dispositivos PLD. El ingeniero implementa el circuito de referencia con diferentes dispositivos y hace comparaciones de los resultados medidos. El software de las herramientas de diseño puede también ser "referenciado." Referenciando el software de las herramientas de diseño se informa el grado de desempeño logrado de los diseños, usando las herramientas y el tiempo de cómputo requeridos. Tales medidas de desempeño pueden usarse entonces para comparar la eficiencia del diseño de las diferentes herramientas cuando ellas se usan con un dispositivo de referencia.

Registro: Una función lógica que tiene uno o más flip – flops en su salida para retener información después de que la función se ha realizado.

Registro Alambrado: Un registro del PLD cuyas entradas y salidas no están disponibles en los pines de entrada o salida del circuito.

Registro de Entrada: Un flip – flop o latch disponible en algún CPLD para mantener la señal de entrada, útil en sistemas de bus multiplexado.

Registro PAL: Un dispositivo PAL que contiene un flip – flop en su salida que puede también ser alimentado al arreglo lógico.

Relojes Asíncronos: Una característica arquitectónica en determinados CPLD's por medio del cual los términos producto se usan como los relojes para el flip – flop.

Reprogramable: Tener la capacidad para poseer una configuración anteriormente programada, borrarla y adoptar una nueva configuración.

Retroalimentación: La trayectoria de la señal dentro de una celda que puede conectar una señal internamente generada a una entrada. La retroalimentación es comúnmente programable y puede tener elementos lógicos acumulativos en su trayectoria.

SDF: Formato de Retraso Estándar. Formato estándar de la industria para describir retardos en la colocación y enrutamiento.

Síntesis Lógica: Un proceso automatizado que convierte una descripción de diseño en una forma apropiada para la implementación. Comúnmente incluye diversas capacidades que pueden perfeccionar un circuito limitado por parámetros tales como tamaño, velocidad, costo, y / o consumo de potencia.

SRAM: Memoria de Acceso Aleatorio Estática.

Suma de Productos: Una expresión lógica que describe la salida de un arreglo de compuertas AND seguida por un arreglo de compuertas OR.

Término Producto: La salida de un arreglo AND.

Transistor de Paso: Un transistor que actúa como un interruptor en la función de programación que proporciona interconexión programable. El transistor actúa como un tercer dispositivo de interrupción terminal.

VERILOG: Lenguaje de diseño de entrada de descripción de Hardware de alto nivel de Cadence.

VHDL: Lenguaje de Descripción de Hardware VHSIC. Un desarrollo HDL bajo el Departamento de Defensa de programa VHSIC.

XACT: Tecnología CAD Avanzada de Xilinx. Un nombre para las herramientas de diseño de Xilinx para ser usadas con sus productos FPGA.

XC: Nombre de la familia de productos de Xilinx.

XNF: Formato "Netlist" Externo. Un formato para convertir esquemas en los sistemas de diseño de FPGA's de Xilinx.

22V10: Un dispositivo muy popular inventado por AMD que está en el extremo superior de la clase SPLD en cuanto a capacidad funcional. Tiene 12 entradas y 10 macroceldas de salida flexible.

BIBLIOGRAFIA

- Ashok K. Sharma.
  Programmable Logic Handbook
  Mc Graw – Hill.
  1998.

- Carter, John W.
  Digital Designing with Programmable Logic Devices.
  Prentice Hall.
  1997.

- Fabricius, Eugene D.
  Diseño Lógico Moderno y Teoría de la Conmutación.
  CECSA.
  1996.

- Floyd, Thomas L.
  Digital Fundamentals.
  Prentice – Hall.
  N. J. 1997.

- García Sánchez y Gil T.
  Circuitos y Sistemas Digitales.
  Tebar Flores.
  México 1992.

- Katz, Randy H.
  Contemporary Logic Design.
  The Benjamin / Cummings Publishing Company Inc.
  1994.

- Morris Mano, M.
  Lógica Digital y Diseño de Computadoras.
  Prentice – Hall.
  México 1991.

- Nelson, Victor P.
  Análisis y Diseño de Circuitos Lógicos Digitales.
  Prentice – Hall.
  México 1996.

- Taub Schilling, H.
  Digital Integrated Electronics.
  Mc. Graw – Hill.
  1991.

- Tavernier, Christian.
  Circuitos Lógicos Programables.
  Paraninfo.
  España, 1994.

- Tocci, Ronald J.
  Sistemas Digitales
  Prentice – Hall.
  México, 1993.

- Van den Bout, D.
  The Practical Xilinx Designers Lab Book.
  Prentice – Hall.
  N. J. 1998.

- Wakerly, John F.
  Diseño Digital, Principios y Prácticas.
  Prentice – Hall.
  México 1992.

- The Programmable Logic Data Book.
  Xilinx.
  1996.