En el siguiente manual se detallan las acciones necesarias que consiguen controlar el rutado automático de un conjunto de líneas. La utilización de esta técnica de control permite realizar implementaciones de un diseño que mantienen un conjunto de líneas en unos recursos de rutado fijados de antemano, facilitando la reconfiguración dinámica de la FPGA para incorporar periféricos.

Se parte de un sistema desarrollado en algún tipo de lenguaje de alto nivel, como VHDL, Verilog, etc... Durante todo el desarrollo se utiliza siempre las herramientas normales de software proporcionadas por Xilinx, utilizando al completo su funcionalidad.

El control del rutado de la interfaz se fija hasta el nivel de asignar a cada *net* un recurso físico (cable) que enlaza las zonas fija y dinámica, y sólo éste puede realizar el enlace. Ninguna línea que resulte tras el rutado automático del resto de la lógica puede ocupar recursos físicos de la zona dinámica, y cada *net* de la interfaz sólo puede unir ambas zonas mediante la línea física especificada

Para obtener una interfaz en una zona asignada de la FPGA se descompone el proceso en varias fases, que constituyen la metodología a seguir. La fase más importante la constituye el *Routing* del diseño y la interfaz.

- Diseño de la interfaz en alto nivel
- *Floorplan*
- *Routing*
- Verificación y corrección
- Generación de bits de configuración (proceso *bitgen*)

Diseño de la interfaz en alto nivel:

1) Elegir el conjunto de *nets* cuyo rutado se quiere controlar y que constituirán la interfaz

2) Para poder rutar la interfaz es necesario realizar un estudio acerca de ella, y asegurarse de que aparezca en la *netlist* tras el proceso de síntesis.

3) Hay que configurar el diseño en VHDL en caso de que sea necesario para habilitar la creación de las *nets* que se desean constituyan la interfaz.

4) Si la interfaz no aparece en la *netlist* del diseño, es necesario generar lógica adicional que permita controlar su inclusión. Esta lógica suele ser un registro que captura todas las entradas de la interfaz (desde la zona dinámica) y las salidas se generan a partir de éstas capturas.

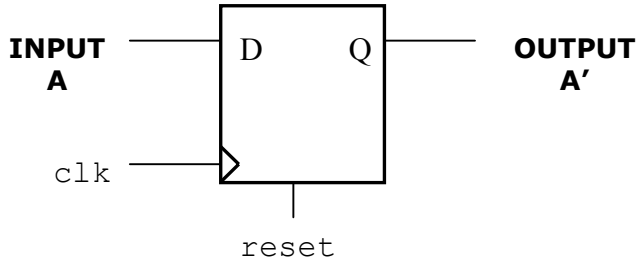

Figura 1:Lógica necesaria para la inclusión de A y A' en la *netlist*

5) Si el número de salidas es superior al número de entradas capturadas con el registro, es necesario volver a capturar con otro registro las salidas que sean necesarias del anterior hasta el total de salidas.

6) En la herramienta de síntesis fijar la restricción de preservar las fronteras de cada módulo diseñado (*preserve hierarchy*) y el esfuerzo de cálculo (*effort*) en el máximo.

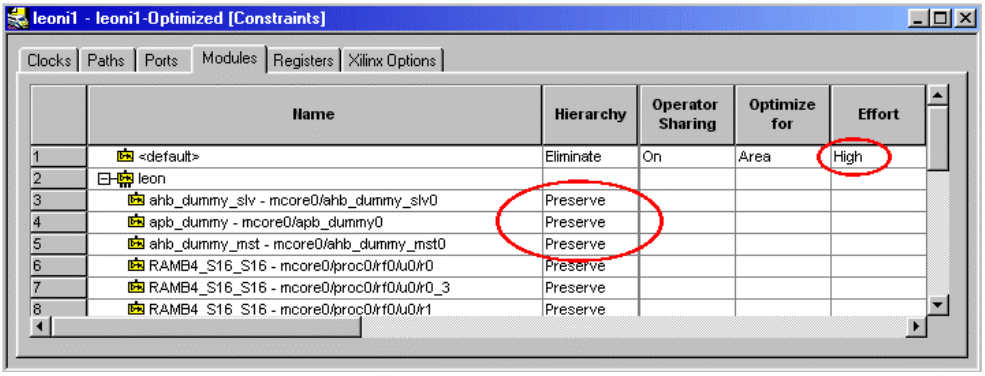

Figura 2: Opciones de síntesis

7) Realizar la síntesis y comprobar que en el resultado aparecen sin fusionar los módulos diseñados.

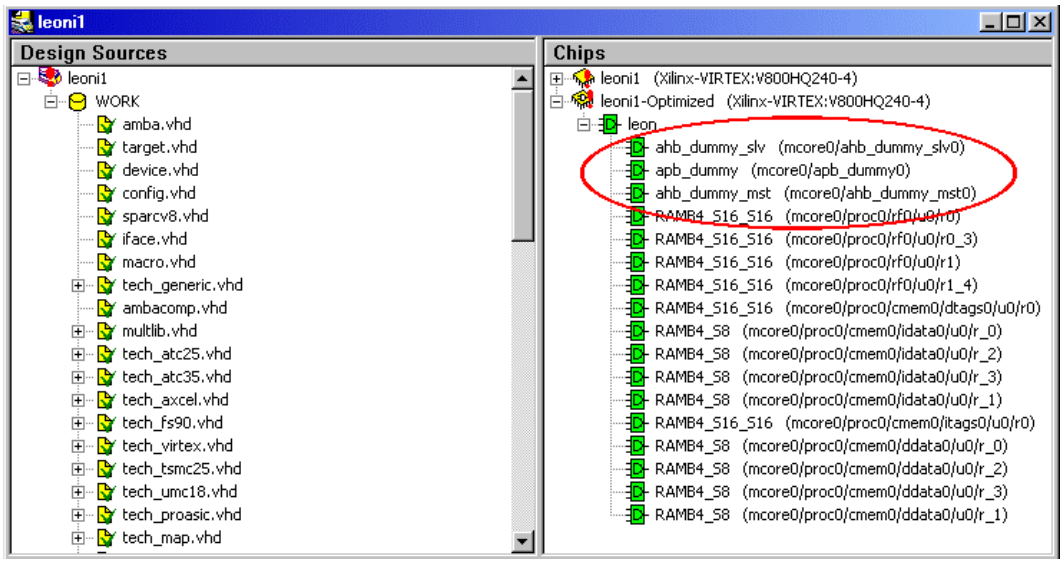

Figura 3: Resultado a obtener en la síntesis

8) Generar la *netlist* en formato *.edf*

9) Ejecutar las herramientas *ngdbuild* y *map* para obtener un fichero Xilinx preliminar, *map.ncd,* del diseño

Floorplan:

1) Abrir el diseño en el *Floorplanner* y centrarse en la ventana *Desing Hierarchy*

2) Realizar una agrupación (*Group* F3) de la lógica adicional diseñada para la interfaz. Al usar las opciones de síntesis indicadas debe de aparecer agrupada por defecto en la estructura jerárquica

3) Hay que realizar también una agrupación de la lógica perteneciente a la zona fija que tiene como entradas o salidas señales que forman parte de la interfaz (*other side logic*). Hay que usar la herramienta de búsqueda para seleccionarla y posteriormente realizar la agrupación.

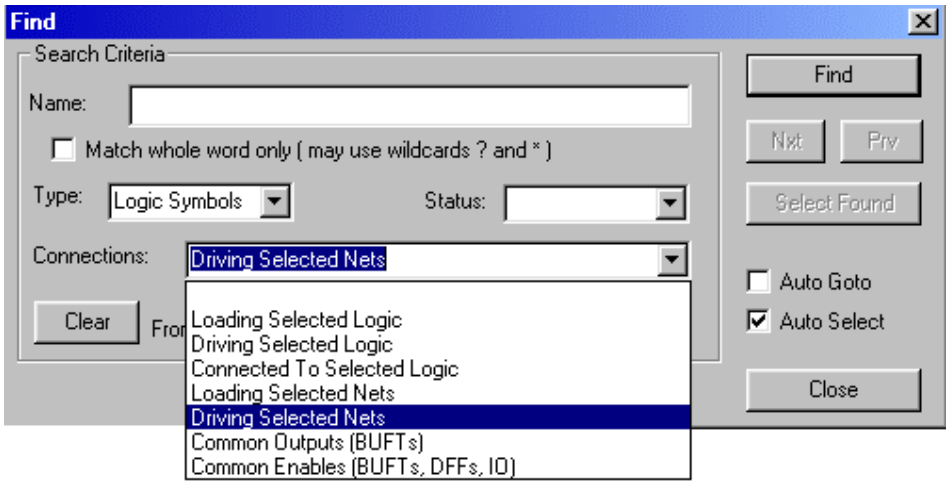

Figura 4: Herramienta de búsqueda

4) Seleccionar un grupo de *nets* en la ventana *Nets* y fijar la propiedad *Connections* a *Loading Selected Nets* o *Driving Selected Nets* según sea salida o entrada de la interfaz respectivamente. Realizar la búsqueda y seleccionar la lógica encontrada.

5) Agrupar la lógica encontrada por la herramienta de búsqueda

6) Realizar una agrupación del resto de lógica que quede en el diseño

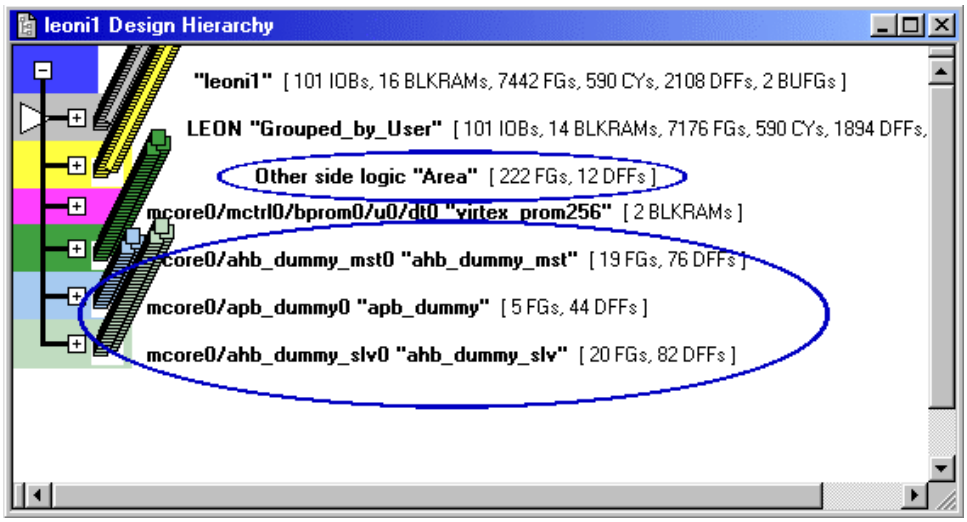

Figura 5: Ejemplo de estructura jerárquica a obtener

7) Asignar restricciones de área a los grupos anteriores. A los grupos de lógica adicional dentro de la zona dinámica de la FPGA. A la lógica de la interfaz del diseño principal, en la zona fija de la FPGA. Al grupo que engloba al resto de lógica, un área que ocupe toda la zona fija y situada en el lado de la FPGA que se usen menos bloques IOBs

8) Incorporar las restricciones que existan en el diseño para pins de E/S, intentando usar IOBs de la zona fija

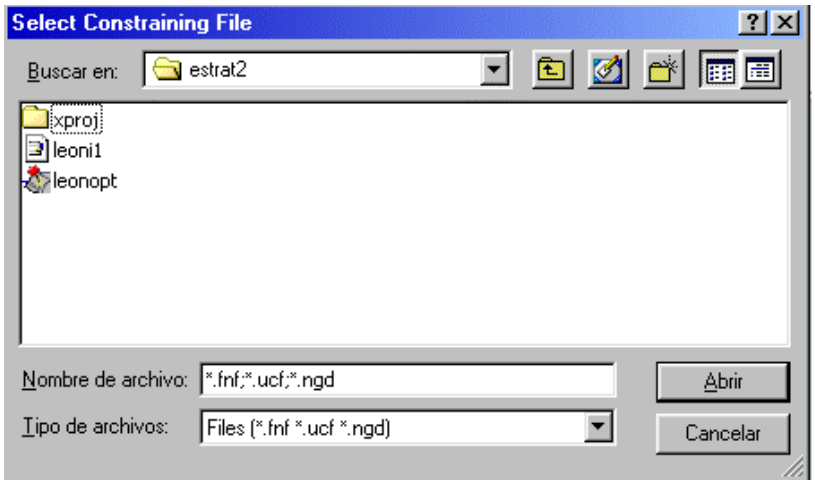

Figura 6: Cuadro de diálogo para insertar restricciones *.ucf*

9) Usar la restricción de prohibición *Not Allowed* para el resto de recursos que quedan libres, entre ellos, CLBs de la zona dinámica no utilizados e IOBs que la rodean

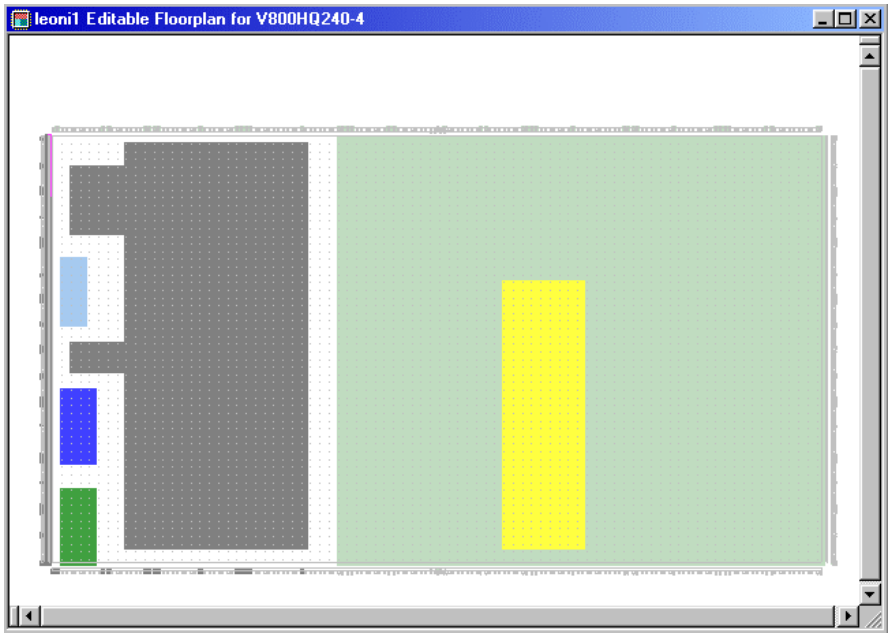

Figura 7: Ejemplo de *floorplan*

10) Anotar las restricciones de área asignadas a cada grupo de lógica, con la notación CLB_R?C?:CLB_R?C?, para usarlas en la siguiente fase

11) Realizar de nuevo el mapeado del diseño pero usando el *floorplan*.

```
map -p dev -o map.ncd -fp <floorplan.mfp> <file.ngd>
   <file.pcf>
```

12) Realizar únicamente el *placement* del diseño con la herramienta *par*

```
par  -w -ol 2 -r map.ncd <file.ncd> <file.pcf>
```

13) Abrir el fichero *.ncd* con *Floorplanner* para que se actualice la información de localización del *Floorplan*

Routing:

1) Elegir el tipo de recurso de rutado a utilizar, y la línea física que será utilizada para cada *net* de la interfaz. Los recursos de rutado a usar deben atravesar la FPGA de lado a lado. En esta elección hay que tener en cuenta la proximidad a la lógica que va a enlazar.

2) Usar una aplicación desarrollada en un entorno de programación para generar un fichero *script* (*.scr*). Debe de extraer del fichero *floorplan* (*.fnf*), para cada *net*, el pin *driver* y un *pin load* que se encuentre en la zona contraria de la FPGA. Para facilitar la búsqueda a la aplicación se puede utilizar las restricciones de área anotadas en la fase de *floorplan*. La estructura del *script* generado debe ser la siguiente:

```
# net        mcore0/apbi<0><PWDATA><9> (comentario)
unselect -all
select pin CLB_R24C52.S1.X       (pin de una zona de la FPGA)
select wire H_HLONG/x:8845/y:4196 (línea física a usar)
select pin CLB_R20C4.S1.BX       (pin de la otra zona)
route                            (orden de rutado manual)

## los siguientes comandos se ejecutan una vex rutadas todas las nets de
##la interfaz, por tanto al final del script
unselect -all
select net mcore0/apbi<0><PWDATA><9>
autoroute                  (rutado automático del resto
                            de pins conectados a la net)
```

3) Abrir el fichero que resultó del *placement* con FPGA *Editor*, con la opción *read/write*

4) Ejecutar el fichero *script* mediante el menú *Script Playback*

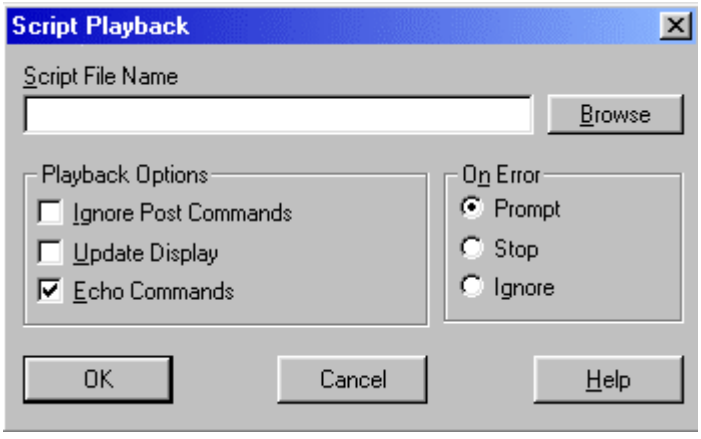

Figura 8: Ejecución de *script*

5) Guardar el diseño con la interfaz rutada. La información generada en el *script* será necesaria para el desarrollo de futuros módulos que sean conectados a la FPGA.

6) El resto del diseño se ruta con la opción *Incremental Routing* (-k) de la herramienta *par*. No es necesario bloquear las *nets* ya rutadas

```
par  -w -ol 4 -k -p  <file.ncd> <filer.ncd> <file.pcf>
```

Verificación y Corrección (V&C):

1) Comprobar con FPGA *Editor* que la interfaz mantiene la situación física asignada antes de realizar el rutado automático del resto de conexiones

2) Si no ha sido modificada ninguna *net* que forme parte de la interfaz, entonces ya se ha obtenido el diseño completamente rutado y cumpliendo todas las restricciones que se le han impuesto. El último paso consiste en generar los bits necesarios para la configuración de la FPGA (*bitgen)*

3) La modificación de una *net* rutada por parte de la herramienta *par* se produce sólo en ciertos casos con unas circunstancias de diseño muy especiales. Si la interfaz rutada contiene un gran número de líneas y el dispositivo FPGA utilizado contiene el diseño con un gran porcentaje de utilización, puede ocurrir que alguna línea cambie su rutado inicial

4) Deteminar la/s *net/s* que no se ha/n rutado correctamente

5) El problema se debe a la saturación de la matriz de rutado que existe en cada CLB, y en particular la del CLB que se conecta a la *net* modificada

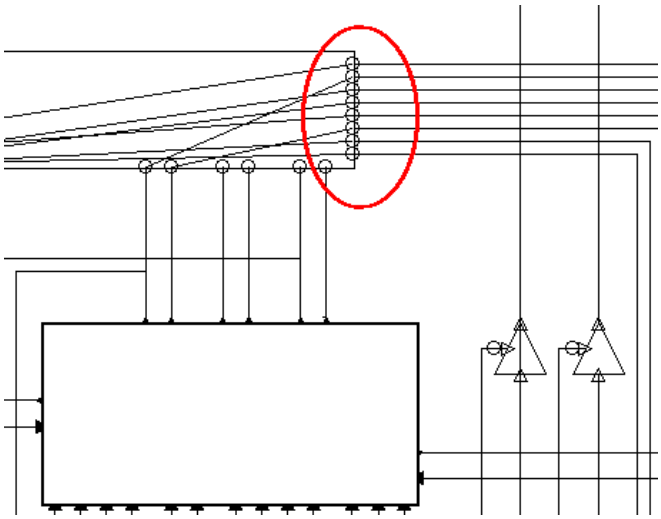

Figura 9: Ejemplo de saturación de la matriz de rutado

6) Corrección en el *placement* del diseño intercambia a una localización cuya matriz de rutado tenga una menor densidad de conexiones, siempre dentro de la zona asignada (restricción) al correspondiente grupo de lógica

7) Realizar el rutado de la *net* perteneciente a la interfaz modificada con el CLB en la nueva localización. Los comandos se obtienen del *script*, con la única modificación de la nueva denominación del pin del CLB afectado

8) El resto de conexiones modificadas con el cambio que no forman parte de la interfaz se rutan usando el comando *autoroute* aplicado al CLB cuya localización ha sido modificada

9) Comprobar el diseño haciendo un chequeo del DRC.

10) Tras las acciones de corrección, se ha obtenido el diseño completamente rutado, y con la *net*s pertenecientes a la interfaz en los recursos físicos que se han especificado

Bitgen:

1) Si no se han obtenido errores en el DRC se pueden generar los bits de configuración

2) La información acerca de la estructura de los bits de configuración es proporcionada por Xilinx. En el FPGA Editor se abre una ventana de comando y se da la siguiente orden:

```
bitgen –g <filerp.ncd>  <bitfile.bit>
```

3) O en el *prompt* del S.O. teclear:

```
bitgen <filerp.ncd>  -l -w -f bitgen.ut
```

4) Si se desea aplicar reconfiguración parcial al diseño hay que identificar las tramas de bits que corresponden a la zona fija y las tramas que corresponden a la zona dinámica. La unidad utilizada es una columna de la matriz de CLBs. Una vez identificadas se puede obtener los bits de configuración dinámica, o en su caso aplicarlos durante el diseño de los periféricos reconfigurables

Tras la realización de las anteriores acciones se obtiene una implementación operativa de un diseño con un conjunto de nets rutadas sobre los recursos de rutado que se han especificado durante el desarrollo, además de separar la lógica del diseño para obtener una zona donde se pueda aplicar reconfiguración parcial de la FPGA.