



# Recomendaciones en el uso de código RTL

- 1. Reglas Generales
- 2. Portabilidad del Código
- 3. Señales de reloj y reset
- 4. Recomendaciones para Síntesis

Técnicas de Modelado para Síntesis

- Genéricas
  - Regla: Elegir un nombre adecuado para el diseño que se use consistentemente.
  - Recomendación: Usar minúsculas para todos los nombres de señales, variables y puertos.
  - Recomendación: Usar mayúsculas para nombres de constantes y tipos definidos por el usuario.
  - Recomendación: Emplear nombres significativos para señales, puertos, funciones y parámetros.
  - Recomendación: Utilizar nombres descriptivos cortos a los parámetros del diseño, para evitar que las herramientas de síntesis generen nombres de unidades de diseño excesivamente largos.

Técnicas de Modelado para Síntesis

\_

### 1. Reglas generales

- Genéricas
  - Recomendación:
    - La señal de reloj se denominará clk.
    - Si se utiliza más de una señal de reloj en el diseño es necesario que todas comiencen con el prefijo clk.
    - Usar el mismo nombre para todas las señales de reloj que son controladas desde la misma fuente.
  - Recomendación:
    - Utilizar la misma nomenclatura para todas las señales activas a nivel bajo del diseño: un '\_'seguido por una letra en minúscula.
    - Para estandarización, se recomienda emplear \_n para indicar todas las señales activas a nivel bajo.
    - La señal de reset se denominará rst. Si esta señal es activa a nivel bajo: rst\_n.

Técnicas de Modelado para Síntesis

- Genéricas
  - Regla:
    - Emplear un orden consistente de bits para describir buses. En VHDL: (y downto x) o (x to y).
    - Se recomienda usar en VHDL para señales multibits: (y downto x) para establecer un estándar entre múltiples diseños y equipos de diseño.
  - Recomendación:
    - Siempre que sea posible, usar el mismo nombre o nombres similares para puertos y señales que estén conectados.

Técnicas de Modelado para Síntesis

-

### 1. Reglas Generales

- · Convenios en la denominación para soporte VITAL
  - VITAL (VHDL Initative Towars ASIC Libraries):
    - Estándar para modelado a nivel de puertas de librerías VHDL (IEEE 1076.4). Especifica las restricciones en los convenios de elección de nombres (y otras características) de las declaraciones de los puertos del nivel superior de un elemento de la librería.
  - Regla:
    - Un puerto no estará en modo linkage.
  - Regla:
    - El puerto, en un declaración de un puerto de una entidad no puede ser un puerto protegido. Además, la declaración no puede imponer una restricción en el rango del puerto, ni puede alterar la resolución del puerto desde la definición dada en el paquete lógico estándar.

Técnicas de Modelado para Síntesis

- · Cabeceras código fuente
  - Regla:
    - · Incluir una cabecera en la parte superior de cada fichero fuente, incluyendo los scripts de síntesis.
    - · La cabecera debe contener: nombre del fichero, autor, descripción de la función y lista de las características claves del módulo, fecha de creación, histórico de modificaciones incluyendo día, nombre de la persona que realiza la modificación y descripción del cambio.
    - Ejemplo:
- -- Author :Nombre <email>
  -- Date : DD/MM/YYYY
- -- Abstract : Este fichero contiene la entidad y arquitectura del módulo
- X. A partir de las entradas Y Z -- y E\_D se obtendrán las .
- -- Modification History: -- Date By Version Change Description

Técnicas de Modelado para Síntesis

### 1. Reglas Generales

- Uso de Comentarios
  - Regla:
    - · Usar comentarios adecuados para explicar todos los procesos, funciones y declaraciones de tipos y subtipos.
  - - Usar comentarios para describir puertos, señales y variables, o grupos de señales o variables, evitando comentarios obvios de funcionalidad.
- Mantener sentencias en líneas separadas
  - - · Usar líneas separadas para cada declaración de HDL.
- Longitud de las líneas
  - Recomendación:
    - Mantener la longitud de las líneas en 72 caracteres o menos, utilizando la indentación en las próximas líneas para indicar que es una continuación de la línea anterior.

Técnicas de Modelado para Síntesis

- Indentación
  - Regla:
    - Usar la indentación para mejorar la lectura de líneas de código continuas y bucles anidados.
    - Se recomienda emplear dos espacios para la indentación puesto que valores mayores limitan la longitud de la línea cuando existen varios niveles de bucles, evitando el empleo de tabuladores.
- No usar palabras reservadas del HDL
  - Regla:
    - No usar palabras reservadas de VHDL o Verilog para nombres de cualquier elemento de los ficheros fuente porque los diseño de macros deben ser traducibles de VHDL a Verilog y viceversa.

Técnicas de Modelado para Síntesis

11

### 1. Reglas Generales

- Organización de los puertos
  - Regla:
    - Declarar los puertos siguiendo un orden lógico y manteniéndolo consistente en todo el diseño, con un puerto por línea, seguido de un comentario.
  - Recomendación:
    - Declarar los puertos en el siguiente orden:
      - Entradas: reloj/es, reset/s, enable/s, otras señales de control, datos y líneas de direcciones.
      - Salidas: Reloj/es, reset/s, enable/s, otras señales de control y datos.
  - Recomendación:
    - Emplear comentarios para describir grupos de puertos.
  - Recomendación:
    - Dejar una línea en blanco entre los puertos de entrada y salida para mejorar la lectura.

Técnicas de Modelado para Síntesis

- Secciones de entidad, arquitectura y configuración en VHDL
  - Recomendación:
    - Localizar las secciones de entidad, arquitectura y configuración del diseño VHDL en el mismo fichero, para que sea más fácil entender y mantener un diseño
    - Si se incluye en un fichero fuente configuraciones de subdiseños con declaraciones de la entidad y arquitectura, se debe comentar para síntesis, empleando en el código VHDL los pseudo-comentarios pragma translate\_off y pragma translate\_on o aquellos que sean necesarios por la herramienta soportada.
- Uso de funciones
  - Recomendación:
    - Siempre que sea posible emplea funciones con sus correspondientes comentarios.
    - · Generalizar funciones las hace reutilizables.

Técnicas de Modelado para Síntesis

13

### 1. Reglas Generales

- Uso de bucles y arrays
  - Recomendación:
    - Usar bucles y arrays para mejorar la lectura del código fuente.
  - Recomendación:
    - Los arrays son significativamente más rápidos para simular que los bucles for, por lo que siempre que se pueda es conveniente usar operaciones con vectores en arrays en vez de bucles for.
- Uso de etiquetas significativas
  - Regla:
    - Etiquetar cada proceso con un nombre significativo, resulta muy útil en el momento de la depuración.
  - Reala.
    - No duplicar nombres de señales con variables o con entidades.

Técnicas de Modelado para Síntesis

### 2. Portabilidad del Código

- Usar sólo tipos estándar IEEE
  - Regla:
    - Usar sólo tipos estándar IEEE. Se pueden crear tipos y subtipos adicionales pero todos basados en tipos estándar IEEE.
  - Recomendación:
    - Usar std\_logic en vez de std\_ulogic.
    - Igualmente std\_logic\_vector en vez de std\_ulogic\_vector.
    - Los tipos std\_logic y std\_logic\_vector proporcionan las funciones de resolución requeridas para buses triestado, lo que NO ocurre con std\_ulogic y std\_ulogic\_vector
  - Recomendación:
    - Hay que ser conservador en el número de subtipos que se crean.
       Usar demasiados subtipos hace que sea difícil de entender el código.

Técnicas de Modelado para Síntesis

14

### 2. Portabilidad del Código

- Usar sólo tipos estándar IEEE (cont)
  - Recomendación:
    - No usar los tipos *bit* o *bit\_vector*, puesto que muchos simuladores no aportan funciones aritméticas construidas para estos tipos.
  - Recomendación:
    - No usar valores numéricos codificados de forma fija sino emplear constantes que definan dichos valores en el diseño.
    - Como una excepción, se puede usar los valores 0 y 1 (pero no en combinación, como 1001).

Técnicas de Modelado para Síntesis

### 2. Portabilidad del Código

- Paquetes
  - Recomendación:
    - Recoger todas las definiciones de funciones y valores de los parámetros para un diseño en un único fichero separado ("package") con el nombre DesignName\_package.vhd.
- Scripts empotrados dc\_shell
  - No es recomendable el uso de comandos de síntesis empotrados directamente en el código fuente, puesto que si el diseño va a ser reutilizado en una nueva aplicación, los objetivos de síntesis pueden de ser diferentes.
  - Si el código fuente es reutilizado con nuevas versiones de la herramienta de síntesis (*Design Compiler por ejemplo*), el comando puede que sea obsoleto.
  - Hay diversas excepciones a esta regla. En particular, las directivas de síntesis synthesis on y synthesis off deben estar empotradas en un lugar adecuado del código.

Técnicas de Modelado para Síntesis

17

### 2. Portabilidad del Código

- Usar librerías independientes de la tecnología
  - Recomendación:
    - Emplear librerías independientes de la tecnología (*DesignWare* o equivalentes).
    - Estas librerías aportan arquitecturas para los componentes aritméticos básicos (sumadores, multiplicadores, comparadores, incrementadores, decrementadores, etc), así como otros más complejos (senos, cosenos, divisores, raíz cuadrada, desplazadores aritméticos, etc.).
    - También incluye componentes secuenciales, completamente portables, que pueden ahorrar un considerable tiempo de diseño (FIFOs, controladores de FIFO, ECC, CRC, componentes JTAG, etc.).

Técnicas de Modelado para Síntesis

### 3. Señales de reloj y reset

- Estructura del reloj
  - Una estructura simple de reloj es más fácil de entender, analizar y mantener, además de producir mejores resultados de síntesis.
  - La mejor estructura de reloj en un sencillo reloj global y flip-flops activos por flanco de subida.
  - Evitar mezclar flancos de reloj (subida y bajada)
- Recomendación:
  - Evitar usar flip-flops activos por flanco de subida y por flanco de bajada en el mismo diseño.
    - Mezclar flancos de reloj puede ser necesario en algunos diseños, donde por ejemplo, sea útil capturar datos en ambos flancos de reloj.
    - Crea diversos problemas, ya que el ciclo de trabajo del reloj se convierte en un parámetro crítico en el análisis temporal,
    - La mayoría de las metodologías de test basadas en scan requieren un manejo separado de flip-flops activos por flanco de subida y de bajada.

Técnicas de Modelado para Síntesis

19

### 3. Señales de reloj y reset

- Evitar mezclar flancos de reloj (II)
  - Regla:
    - Si se debe emplear un flip-flop activo por flanco positivo y negativo en el diseño, hay que estar seguro de modelar el peor caso del ciclo de trabajo del reloj con precisión en los análisis de síntesis y temporización e indicar en la documentación de usuario el ciclo de trabajo asumido.
  - Recomendación:
    - Si se debe usar un gran número de flip-flops activos por flanco positivo y negativo en el diseño, puede ser útil separarlos en diferentes módulos.
      - Facilita la identificación de los flip-flops activos por flanco negativo y la partición en diferentes cadenas de scan

Técnicas de Modelado para Síntesis

### 3. Señales de reloj y reset

- Evitar buffers de reloj
  - Recomendación:
    - · Evitar el manejo de buffers de reloj en el código RTL.
      - Normalmente, los buffers de reloj se insertan después de la síntesis como parte del diseño físico.
      - En el código RTL sintetizable, las nets de reloj se consideran normalmente nets ideales, sin retardos.
      - Durante el diseño físico, la herramienta de inserción de árboles de reloj inserta la estructura adecuada para crearlo tan próximo a un ideal como sea posible, redes de distribución de reloj balanceado.

Técnicas de Modelado para Síntesis

0.1

### 3. Señales de reloj y reset

- · Evitar el empleo de relojes atacados por puertas
  - Recomendación:
    - Evitar el empleo de relojes atacados por puertas en el código RTL.
      - Estos circuitos tienden a ser específicos de la tecnología y dependiente del retardo.
      - Un esquema temporal inadecuado de un reloj atacado por puerta puede generar un falso reloj o *glitch*, causando que un flip-flop tome un dato incorrecto.
      - El skew de diferentes relojes locales pueden causar violaciones en el tiempo de retención (hold).
      - Se pueden crear problemas de testeabilidad debido a que esta lógica no puede formar parte de una cadena scan.

Técnicas de Modelado para Síntesis

### 3. Señales de reloj y reset

- · Evitar los relojes generados internamente
  - Recomendación:
    - Evitar usar los relojes que han sido generados internamente porque esta lógica no puede formar parte de una cadena scan y hacen más difícil (normalmente no es posible) restringir el diseño para la síntesis.
  - Recomendación:
    - Si se necesita un señal de reloj controlada por puertas o un reloj o reset generado internamente, es mejor mantener el circuito de generación del reloj y/o reset como un módulo separado en el más alto nivel de jerarquía del diseño.
    - Dividir el diseño de tal forma que toda la lógica de un único módulo usa un reloj y un reset.

Técnicas de Modelado para Síntesis

23

### 3. Señales de reloj y reset

- · Evitar resets generados internamente
  - Recomendación:
    - En la medida que sea posible evitar resets generados internamente.
    - Generalmente, todos los registros deberían ser reseteados al mismo tiempo. Esta restricción hace que el diseño sea mucho más sencillo y más fácil de controlar.
  - Recomendación:
    - Si se requiere un reset condicional, es necesario crear una señal separada para la señal de reset, y aislar la lógica de reset condicional en un módulo separado.
      - Esta recomendación permite un código más legible y mejora los resultados de síntesis.

Técnicas de Modelado para Síntesis

- Obtención de registros
  - Recomendación:
    - Los principales elementos que se utilizan en la lógica secuencial son los registros (flip-flops).
    - Para mantener la consistencia y asegurar una correcta síntesis, es conveniente utilizar procesos que infieren registros independientes de la tecnología.

```
-- proceso con reset sincrono
PROC: process(clk)
begin
if (clk'event and clk='1') then
if rst_n='0' then
...
else
...
end if;
end if;
end process PROC;
```

-- proceso con reset asíncrono
PROC: process(clk,rst\_n)
begin
if (rst\_n='0') then
....
elsif (clk'event and clk='1') then
....
end if;
end process PROC;

Técnicas de Modelado para Síntesis

25

### 4. Recomendaciones para Síntesis

- Evitar latches
  - Regla:
    - Evitar usar latches en el diseño.
  - Recomendación:
    - Se puede evitar la inferencia de latches usando cualquiera de las siguientes técnicas de código:
      - Asignar valores por defecto al comienzo de un proceso.
      - Asignar salidas para todas las condiciones de entrada.
      - Usar else (en vez de elsif) para la bifurcación final de prioridad.
- Evitar realimentaciones combinacionales
  - Recomendación:
    - Evitar la realimentación combinacional en los bucles de los procesos combinacionales.

Técnicas de Modelado para Síntesis

- Especificar las listas sensibles completas para los procesos
  - Regla
    - Incluir una lista sensible completa en cada uno de los procesos (VHDL).
      - Si no se usa una lista sensible completa, el comportamiento del diseño a nivel RTL y lógico puede diferir.
    - · Bloques combinacionales:
      - Para bloques combinacionales (bloques que no contienen registros ni latches), la lista de sensibilidad debe incluir cada señal que es leída por el proceso. En general, esto significa, cada señal que aparece en el lado derecho de una asignación (<=) o en una expresión condicional.</li>
    - Bloques secuenciales:
      - Para bloques secuenciales, la lista sensible debe incluir la señal de reloj que será leída por el proceso. Si el bloque del proceso secuencial utiliza también una señal de reset asíncrono, es necesario incluir ésta en la lista.
- Lista sensitiva y funcionamiento de la simulación.
  - Recomendación:
    - Hay que estar seguro de que las señales incluidas en la lista sensible son las necesarias.
      - Añadir más señales de las que se necesitan hace más lenta la simulación.

Técnicas de Modelado para Síntesis

27

### 4. Recomendaciones para Síntesis

- Asignaciones de señales vs. variables (VHDL)
  - En la simulación con VHDL, las asignaciones de señales son programadas para la ejecución en el próximo ciclo de simulación mientras que las asignaciones de variable tienen lugar inmediatamente donde aparece en el código.
  - Recomendación
    - Cuando se escribe código sintetizable, se sugiere emplear señales frente a variables para asegurar que el comportamiento de la simulación del diseño presíntesis sea igual que la netlist postsíntesis.
- Declaraciones case vs. declaraciones if-then-else
  - En VHDL, una declaración case supone un multiplexor de un único nivel, mientras que una declaración if-then-else supone un decodificador con prioridad.
  - Recomendación:
    - El multiplexor es un circuito más rápido. Si no se requiere una estructura de un decodificador con prioridad, se recomienda el empleo de declaraciones case frente a la if-then-else.

Técnicas de Modelado para Síntesis

- Declaraciones case vs. declaraciones if-then-else
  - En VHDL, una declaración case supone un multiplexor de un único nivel, mientras que una declaración if-then-else supone un decodificador con prioridad.
  - Recomendación:
    - El multiplexor es un circuito más rápido. Si no se requiere una estructura de decodificación con prioridad, se recomienda el empleo de declaraciones case frente a la if-then-else.
      - En un simulador basado en ciclo, la declaración case se simula más rápido que la declaración if-then-else.
      - Una declaración de señal condicional puede también ser usada para obtener un multiplexor.
      - Para grandes multiplexores, una declaración case se simulará más rápido que una declaración condicional en la mayoría de los simuladores, y especialmente en simuladores basados en ciclos.
      - Para pequeños multiplexores, la velocidad relativa de las dos construcciones varía con diferentes simuladores.

Técnicas de Modelado para Síntesis

20

### 4. Recomendaciones para Síntesis

- Las siguientes recomendaciones se utilizan para hacer un correcto particionado para síntesis,
  - Aporta ventajas como:
    - · mejoras en los resultados de síntesis,
    - tiempo de compilación más rápido, etc.
- · Registrar todas las salidas
  - Recomendación:
    - Para cada bloque de un diseño jerárquico, registrar todas las señales de salida desde el bloque.
      - Simplifica el proceso de síntesis porque hace que las salidas sean fuertemente controlables y los retardos de las entradas predecibles.
      - Todas las entradas de cada bloque llegan con el mismo retardo relativo.
- Localizar la lógica combinacional en un único módulo
  - Recomendación:
    - Mantener la lógica combinacional relacionada en el mismo módulo.
      - Las herramientas de síntesis producen mejores resultados cuando la lógica combinacional relativa está localizada en el mismo módulo ya que generalmente no es recomendable mover lógica a través de los límites jerárquicos del módulo.
      - Mantener la lógica combinacional relacionada en el mismo módulo permite una simulación más rápida.

Técnicas de Modelado para Síntesis

- Lógica asíncrona
  - Recomendación:
    - · Evitar la lógica asíncrona.
      - La lógica asíncrona es más difícil para diseñar correctamente y hacer verificaciones de un diseño.
      - Su temporización y funcionalidad correcta puede ser dependiente de la tecnología lo que limita la portabilidad del diseño.
  - Recomendación:
    - Si se requiere lógica asíncrona en el diseño, partir la lógica asíncrona en un módulo separado al de la lógica síncrona,
      - Aislar dicha lógica en un módulo separado hace mucho más fácil la inspección del código.

Técnicas de Modelado para Síntesis



# Parte II: Modelado RTL de componentes para Xilinx en VHDL Técnicas de Modelado para Síntesis

## Modelado de componentes para Xilinx

- Introducción
- Uso de tipos Signed y Unsigned
- Registros
- Triestados
- Contadores
- Acumuladores
- Registros de desplazamiento
- · Registros dinámicos

- Multiplexores
- Decodificadores
- Codificadores con prioridad
- · Desplazamiento lógico
- Operaciones aritméticas
- RAMS
- Máquinas de estado
- Soporte para "Cajas Negras"

Técnicas de Modelado para Síntesis

### Introducción

- Durante la fase de síntesis se trata de reconocer las construcciones del lenguaje que permitan reconocer estructuras hardware combinacionales y secuenciales tales como flip-flops, sumadores, contadores, maquinas de estado finito (FSM), RAMS, etc.)
- A partir de aquí dichas construcciones se optimizan dependiendo de la arquitectura de la FPGA de referencia.
- Es posible tener control sobre la lógica inferida mediante el uso de restricciones de diseño

Técnicas de Modelado para Síntesis

35

### Soporte de tipos Signed/Unsigned

- En VHDL, dependiendo del tipo de operación y de operandos, es necesario incluir diferentes librerías (signed, unsigned, numeric ...)
- Por ejemplo:
  - Para crear un sumador sin signo:

PACKAGE	TYPE
numeric_std	unsigned
std_logic_arith	unsigned
std_logic_unsigned	std_logic_vector

- Para crear un sumador con signo:

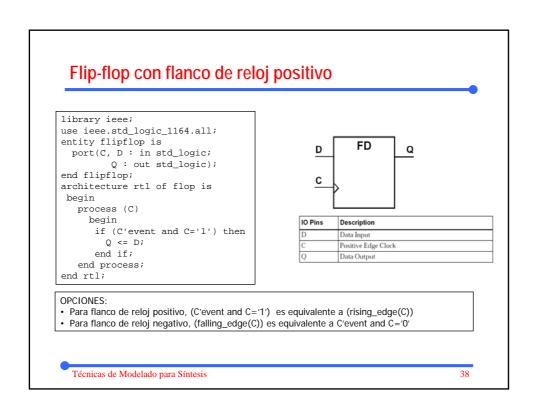
PACKAGE	TYPE
numeric_std	signed
std_logic_arith	signed
std_logic_signed	std_logic_vector

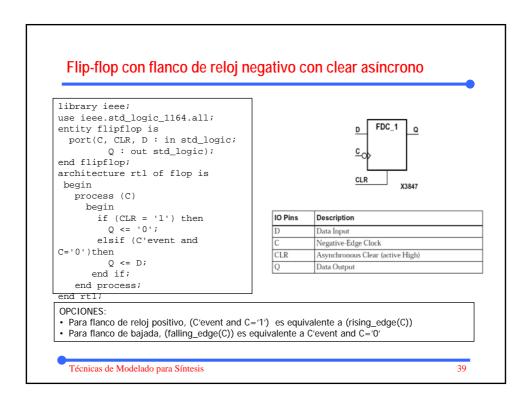
Técnicas de Modelado para Síntesis

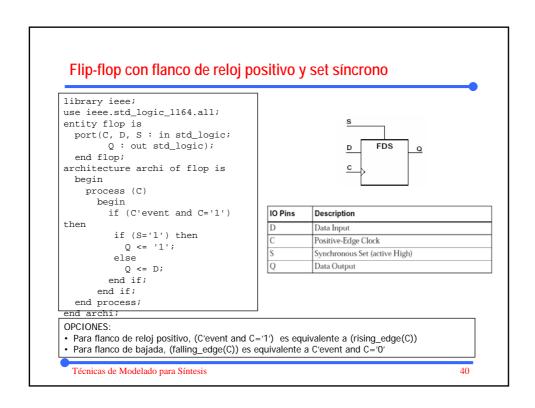
### Registros

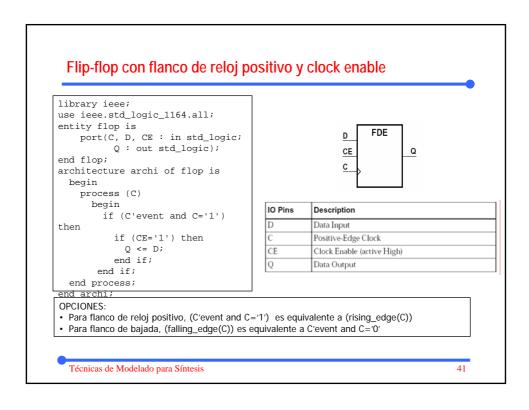
- Es posible inferir registros a partir de I descripción VHDL con las siguientes señales de control:
  - Set/Clear asíncrono
  - Set/Clear síncrono
  - Clock Enable
- · Ejemplo de fichero de información de resultados de síntesis

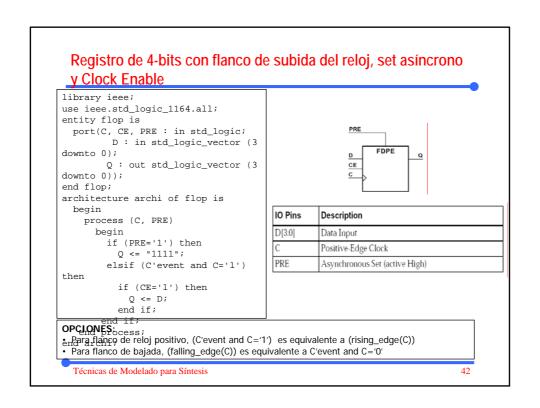
Técnicas de Modelado para Síntesis

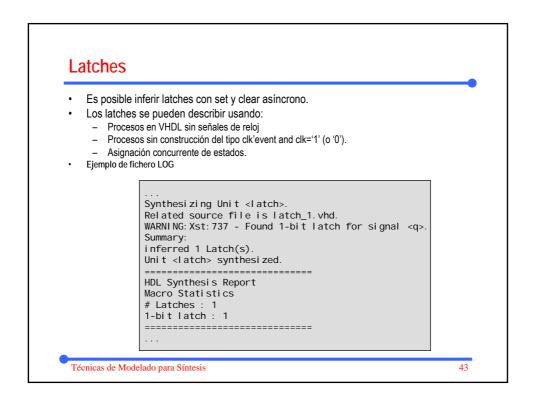


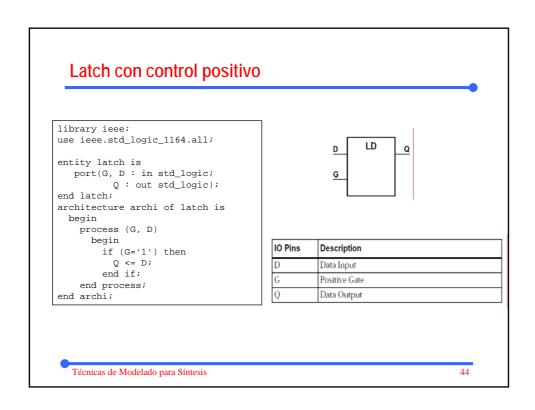


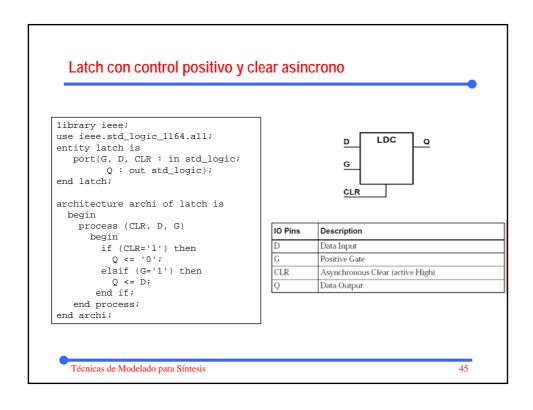


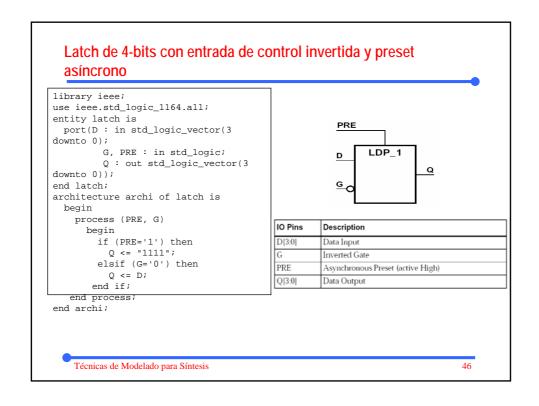




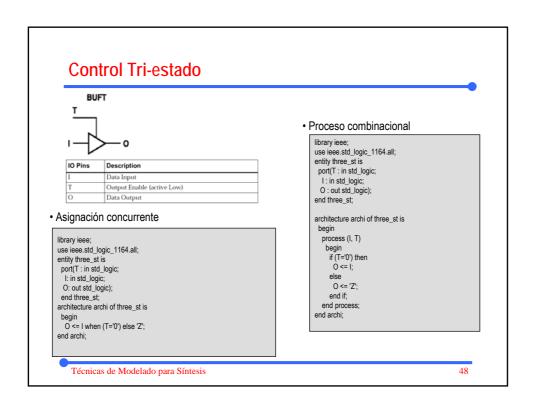




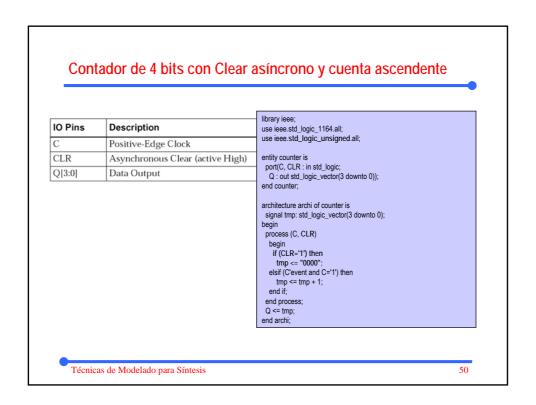


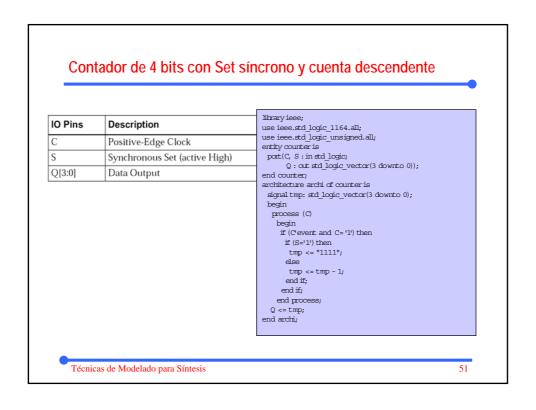


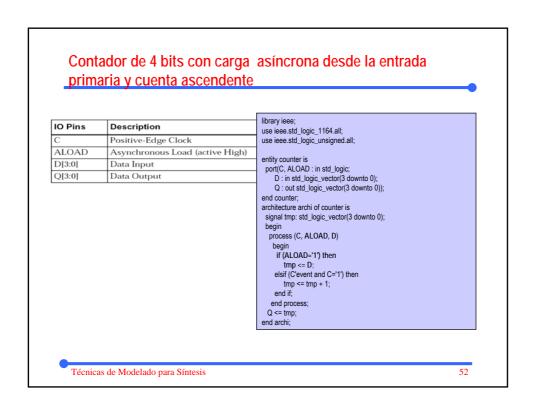
### **Control Tri-estado** El control triestado facilita el control de buses de datos con operaciones de lectura y escritura compartidos entre diferentes dispositivos. En VHDL es posible describirlos mediante: - Un proceso combinacional Una asignación concurrente Ejemplo de fichero LOG: Synthesizing Unit <three\_st>. Related source file is tristates\_1.vhd. Found 1-bit tristate buffer for signal <o>. Summary: inferred 1 Tristate(s). Unit <three\_st> synthesized. HDL Synthesis Report Macro Statistics # Tristates : 1 1-bit tristate buffer : 1 Técnicas de Modelado para Síntesis

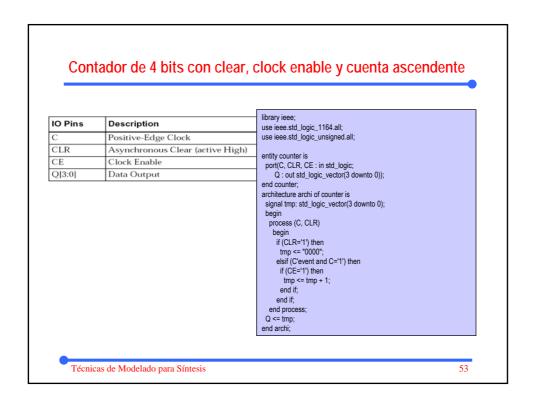


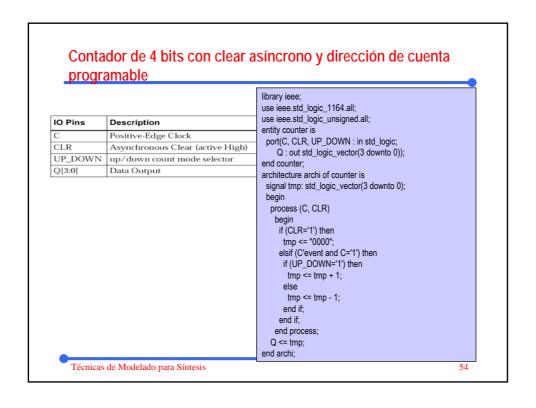
### **Contadores** Fichero LOG Es posible obtener contadores con las siguientes señales de control: - Set/Clear Asíncrono Synthesizing Unit <counter>. Related source file is counters 1.vhd. - Set/Clear Síncrono Found 4-bit up counter for signal <tmp>. - Carga de datos Summary: inferred 1 Counter(s). Unit <counter> synthesized. Asíncrona/Síncrona (señal o contante) - Clock Enable HDL Synthesis Report Modos (Up, Down, Up/Down) Macro Statistics # Counters : 1 Mezcla de todas las posiblidades 4-bit up counter: 1 anteriores Las técnicas de codificación VHDL son similares a las descritas para los registros. Si no existe una señal de LOAD síncrona la herramienta de Está soportada la posiblidad de síntesis puede descomponer el contador en registros y inferir contadores con signo y sin él. sumadores. En este caso en el report aparecerán inferidos únicamente sumadores y registros. Técnicas de Modelado para Síntesis 49

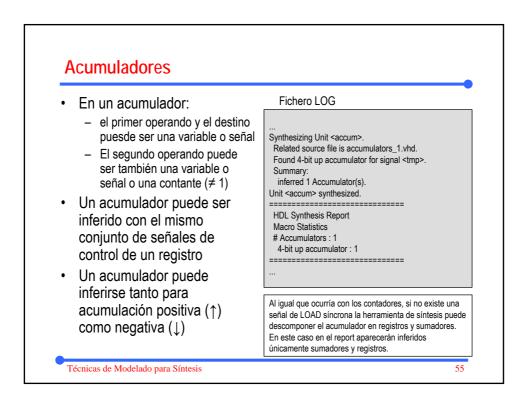


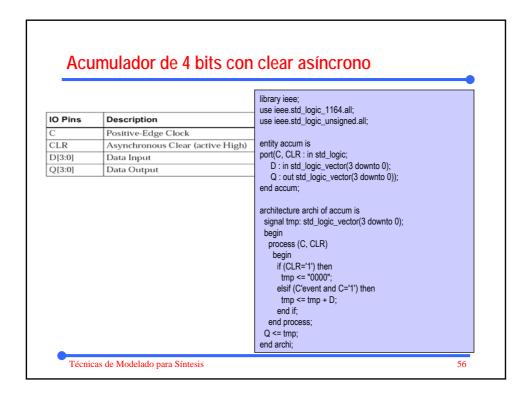










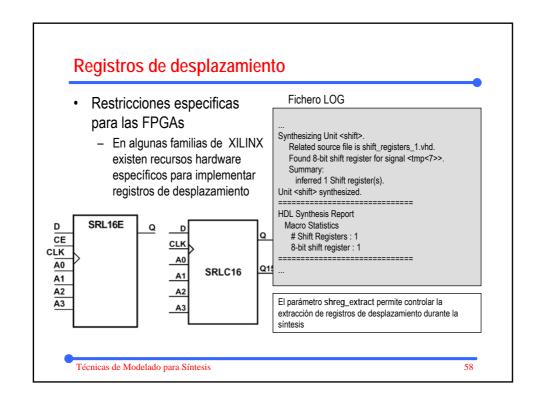


### Registros de desplazamiento

- Un registro de desplazamiento (RD) se caracteriza por las siguientes señales de control:
  - Clock
  - Entrada serial
  - Set/Reset Asíncrono
  - Set/Reset Síncrono
  - Carga paralela síncrona/asíncrona
  - Clock enable
  - Salida serial o paralela. Los modos de salida pueden ser:
    - Serie: únicamente se tiene acceso a la salida del último F/F
    - Paralela: Se accede a cualquiera de los otros F/Fs
  - Modos de desplazamiento: izquierda, derecha, etc.

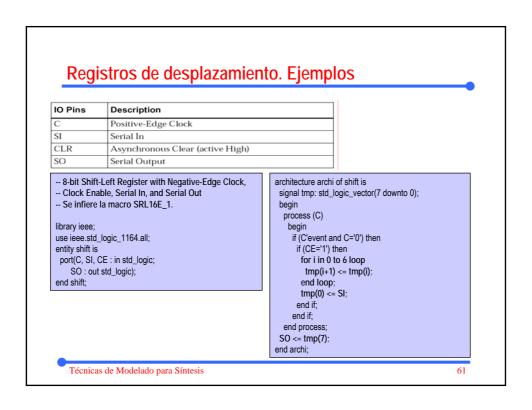
- Existen diferentes modos de describir un RD:
  - Mediante el operador concatenación:
    - shreg <= shreg (6 downto 0) & SI;
  - Mediante bucles FOR:
     for i in 0 to 6 loop
     shreg(i+1) <= shreg(i);</li>
     end loop;
     shreg(0) <= SI;</li>
  - Mediante los operadores predefinidos en el lenguaje (sll, srl, sla, ...)

Técnicas de Modelado para Síntesis





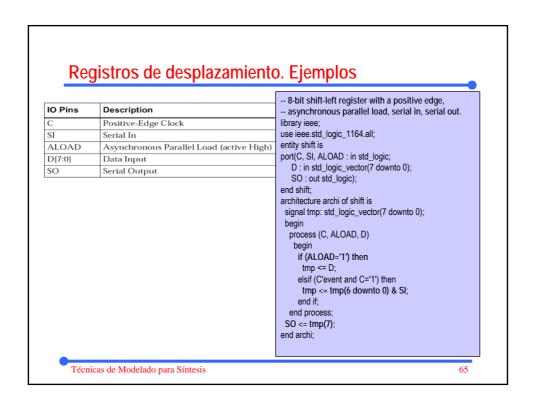




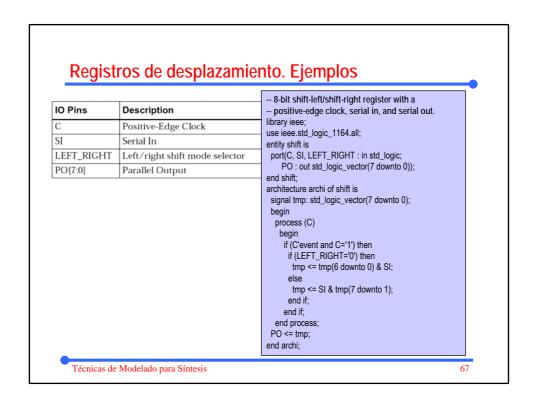






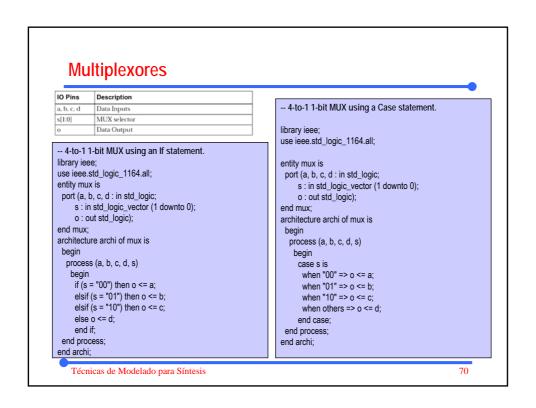


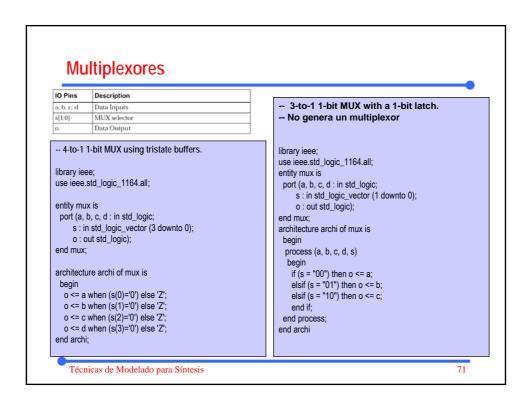


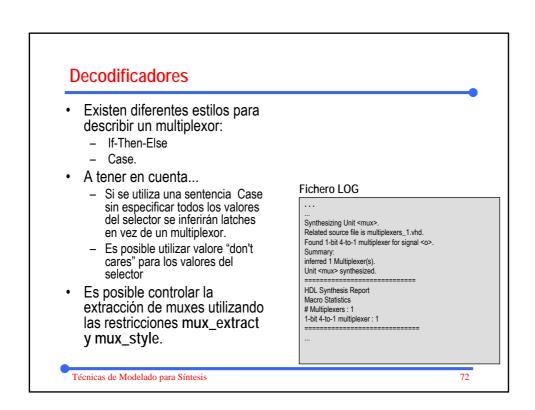


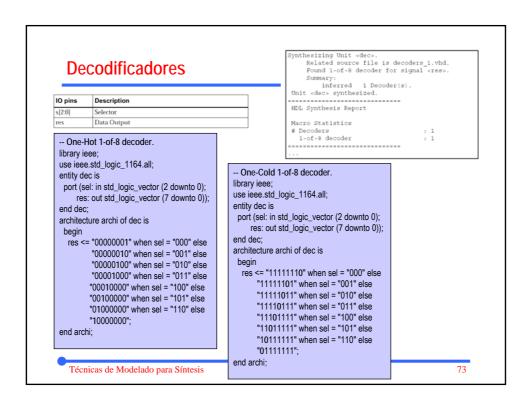


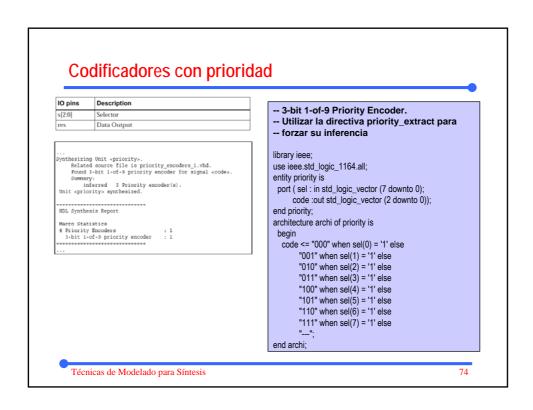
### **Multiplexores** Existen diferentes estilos para describir un multiplexor: - If-Then-Else Case. A tener en cuenta... Fichero LOG Si se utiliza una sentencia Case . . . sin especificar todos los valores del selector se inferirán latches Synthesizing Unit <mux> en vez de un multiplexor. Related source file is multiplexers\_1.vhd. Found 1-bit 4-to-1 multiplexer for signal <o>. Es posible utilizar valore "don't inferred 1 Multiplexer(s). cares" para los valores del Unit <mux> synthesized. selector HDL Synthesis Report Es posible controlar la extracción de muxes utilizando # Multiplexers : 1 1-bit 4-to-1 multiplexer : 1 las restricciones mux\_extract y mux\_style. Técnicas de Modelado para Síntesis 69



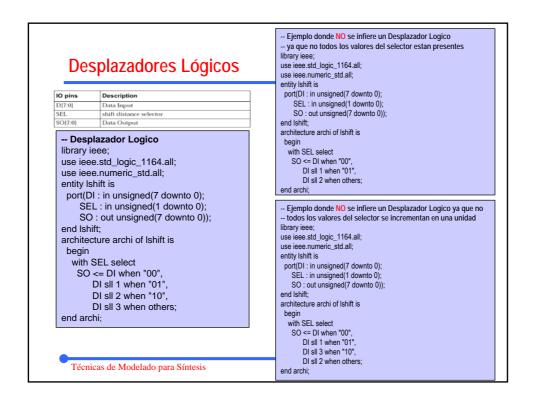








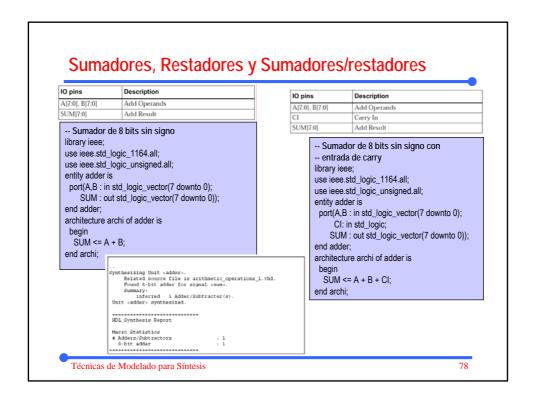


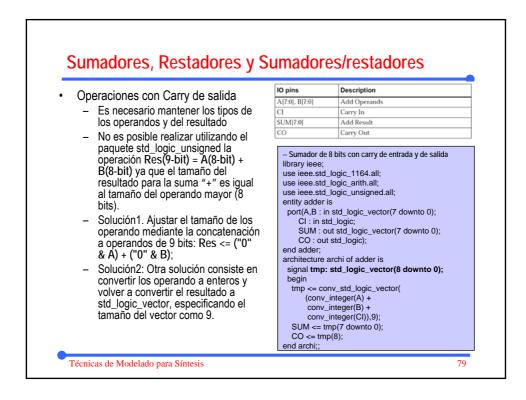


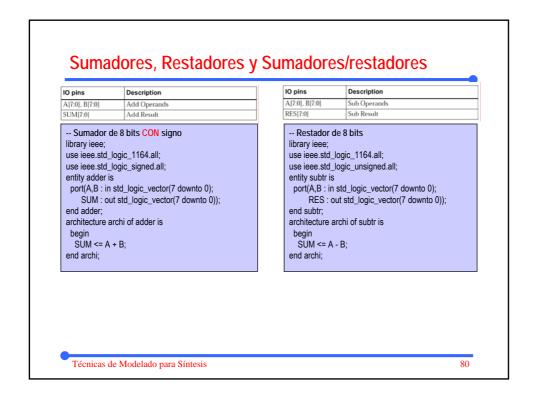
# Operaciones aritméticas

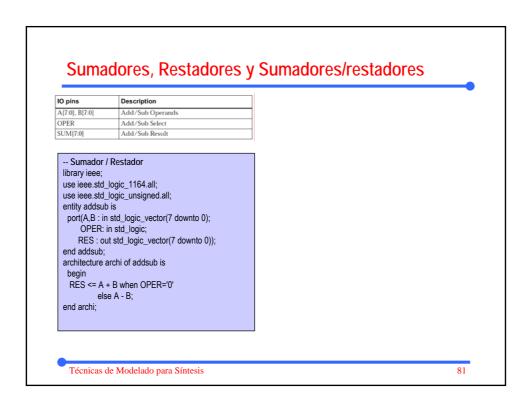
- Operaciones soportadas:
  - Suma (con carry de entrada, de salida y de entrada y salida),
  - Resta.
  - Sumadores/restadores,
  - Comparadores (=, /=, <, <=, >, >=), Multiplicadores y divisores,
  - Operadores con signo o sin signo.
    - · Paquetes:
      - IEEE.std\_logic\_unsigned
      - IEEE.std\_logic\_signed
- Optimización de recursos (sharing) para los operadores aritméticos

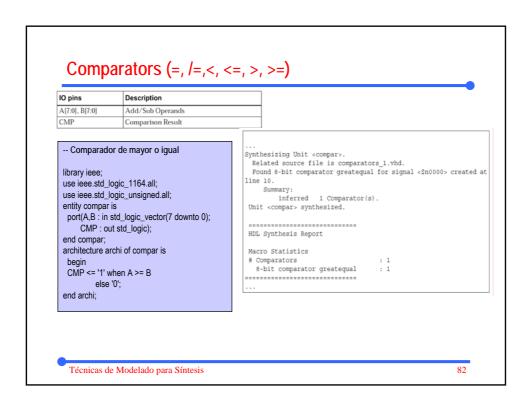
Técnicas de Modelado para Síntesis







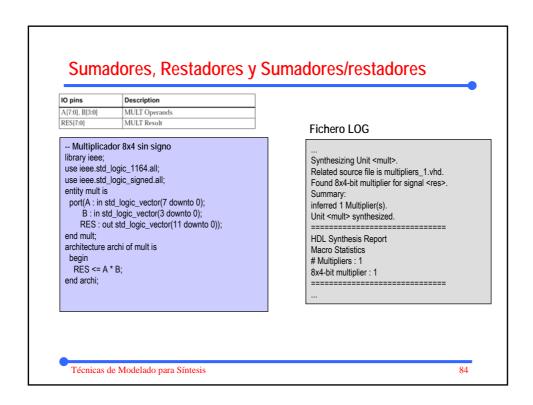




## **Multiplicadores**

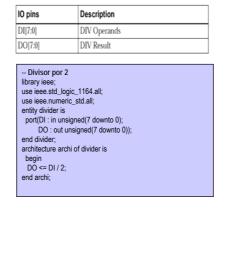
- Cuando se implementa un multiplicador, el tamaño del resultado es igual a la suma de los tamaños de los operandos:
  - A (8-bit) \* B (4-bit) → P (12-bit)
- En Virtex-II and Virtex-II Pro es posible implementar multiplicadores de 18x18 bits usando los bloques multiplicadores presentes en el dispositivo.
- También es posible generar multiplicadores registrados
  - En los siguientes casos se usará un multiplicador + registro:
    - · La salida del multiplicador va a otro componente diferente de un registro
    - · La selección del tipo de multiplicador es LUT
    - · El multiplicador es asíncrono
    - El multiplicador tiene señales de control diferentes de un reset síncrono y un clock enable
    - El multiplicador no se puede mapear a un único módulo de 18x18 bits
  - Las siguientes E/S son opcionales para un multiplicador registrado:
    - · Clock enable, Reset síncrono y asíncrono, reset y puerto de carga

Técnicas de Modelado para Síntesis





La síntesis de la división está soportada únicamente cuando el divisor es una constante potencia de 2: implementación como un desplazador.



Técnicas de Modelado para Síntesis

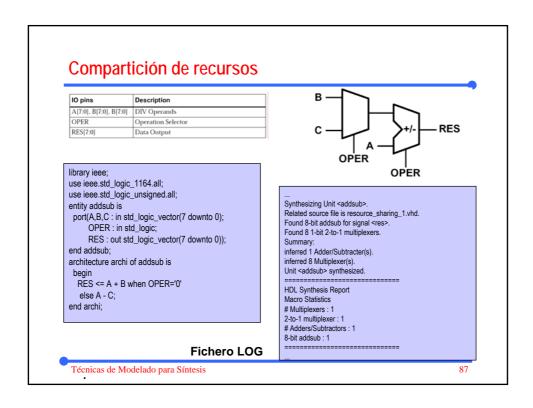
85

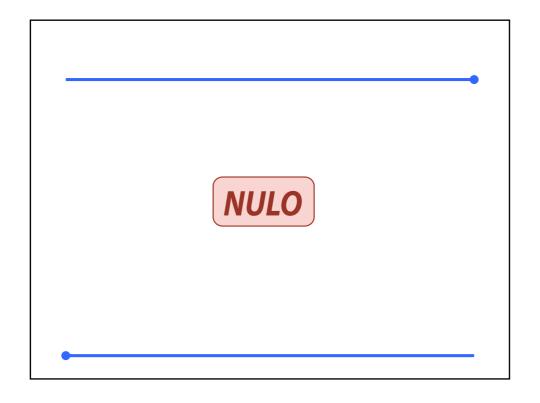
# Compartición de recursos

- El objetivo es minimizar el número de operadores y por tanto la lógica asociada en el diseño sintetizado
- Dos operaciones aritméticas similares se pueden implementarse en una única unidad funcional si no se están usando al mismo tiempo
- Para ello se utilizan multiplexores para seleccionar los datos a utilizar en cada momento.
- Esta capacidad está soportada para sumadores, restadores, sumadores/restadores y multiplicadores.

#### **Fichero LOG**

Técnicas de Modelado para Síntesis





#### **RAMs**

- Se puede inferir memoria RAM distribuida o de bloque según su descripción...
  - Descripción con lectura asíncrona... RAM distribuida.
  - Descripción con lectura síncrona... RAM en bloque (block-RAM).
- · Características de las memorias descritas...
  - Escritura síncrona.
  - Habilitación de escritura y de dispositivo.
  - Lectura asíncrona o síncrona.
  - Reset de la salida de datos y de sus latches.
  - Lectura de puerto único, doble y multipuerto.
  - Escritura de puerto único.
- Directivas relacionadas...
  - > RAM extract ... habiltia o no el reconocimiento de RAM.
  - > RAM\_style ... selecciona el tipo de memoria a usar (distribuida o bloque).

Técnicas de Modelado para Síntesis

89

### **RAMs: Fichero LOG**

```
Synthesizing Unit <raminfr>.
Related source file is rams_1.vhd.
Found 128-bit single-port distributed RAM for signal <ram>.

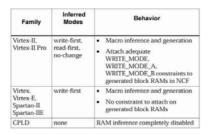
| aspect ratio | 32-word x 4-bit | clock | connected to signal <clk> | rise | connected to signal <clk> | rise | connected to signal <clk> | rise | connected to signal <a > | connected to signa
```

Cuando la herramienta de síntesis reconoce la presencia de una RAM, incluye en el informe la descripción de la misma como tipo y tamaño, así como la relativa a todos sus puertos de E/S.

Técnicas de Modelado para Síntesis

### RAMs: Modos de Lectura/Escritura

- La memoria tipo block-RAM dispone de diferentes modos de sincronización de lectura/escritura.
- · Los modos disponibles según la familia del dispositivo...



 En caso de RAM multipuerto, cada puerto puede usar un modo distinto.

Técnicas de Modelado para Síntesis

91

### RAMs: Modos de Lectura/Escritura - Read First

library ieee;
use ieee.std\_logic\_unsigned.all;
use ieee.std\_logic\_unsigned.all;
entity raminfr is
port (clk : in std\_logic;
 we : in std\_logic;
 en : in std\_logic;
 en : in std\_logic;
 edd : in std\_logic\_vector(4 downto 0);
 di : in std\_logic\_vector(3 downto 0);
 do : out std\_logic\_vector(3 downto 0));
end raminfr;
architecture syn of raminfr is
type ram\_type is array (31 downto 0)
 of std\_logic\_vector (3 downto 0);
 signal RAM: ram\_type;
begin
 process (clk)
 begin
 if clk'event and clk = '1' then
 if en = '1' then
 if we = 1'I then
 RAM(conv\_integer(addr)) <= di;
 end if;
 do <= RAM(conv\_integer(addr));
 end if;
 end if;
 end process;
end syn;

En este modo, cuando se realiza una escritura sobre una posición de la memoria RAM, a la salida de ésta se muestra el contenido que tenía la posición seleccionada <u>antes</u> de ser escrita con el nuevo valor.

P. Ej., supongamos que en la posición 20 está el valor 5 y se accede en escritura a esta posición para poner un 10. La RAM modifica el contenido de su posición 20 al valor 10 y muestra en su salida el valor de la posición 20 previo a la escritura, es decir, 5.

Técnicas de Modelado para Síntesis

### RAMs: Modos de Lectura/Escritura - Write First

En este modo, cuando se realiza una escritura sobre una posición de la memoria RAM, a la salida de ésta se muestra el contenido de la posición seleccionada <u>después</u> de ser escrita con el nuevo valor.

P. Ej., supongamos que en la posición 20 está el valor 5 y se accede en escritura a esta posición para poner un 10. La RAM modifica el contenido de su posición 20 al valor 10 y muestra en su salida el valor de la posición 20 tras la escritura, es decir, 10.

Técnicas de Modelado para Síntesis

93

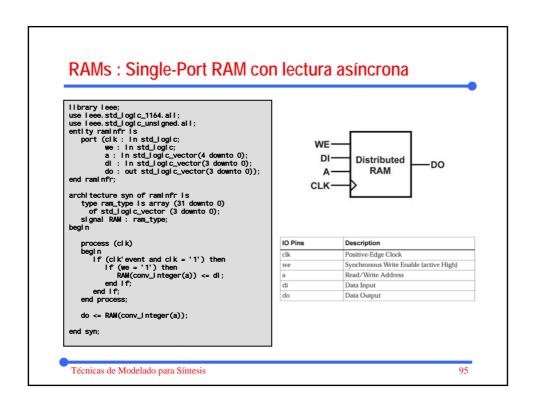
# RAMs: Modos de Lectura/Escritura - No Change

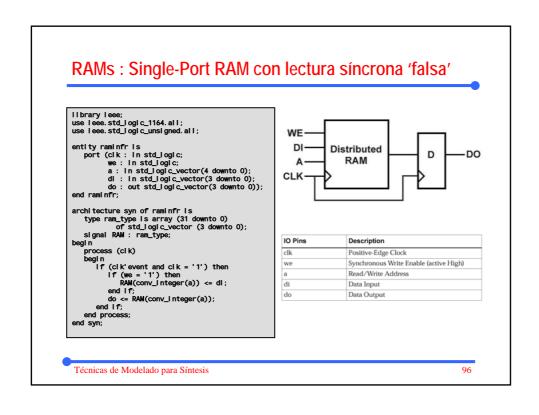
```
library ieee;
use ieee. std logic_unsigned.all;
use ieee. std logic_unsigned.all;
entity raminfr is
port (clk: in std_logic;
we :in std_logic;
en: in std_logic;
addr: in std_logic;
could std-logic_vector(4 downto 0);
do: out std_logic_vector(3 downto 0);
do: out std_logic_vector(3 downto 0);
end raminfr;
architecture syn of raminfr is
type ram_type is array (31 downto 0)
of std_logic_vector (3 downto 0);
signal RAM: ram_type;
begin
process (clk)
begin
if clk'event and clk = '1' then
if en = '1' then
if en = '1' then
RAM(conv_integer(addr)) <= di;
else
do <= RAM(conv_integer(addr));
end if',
end if',
end process;
end syn;
```

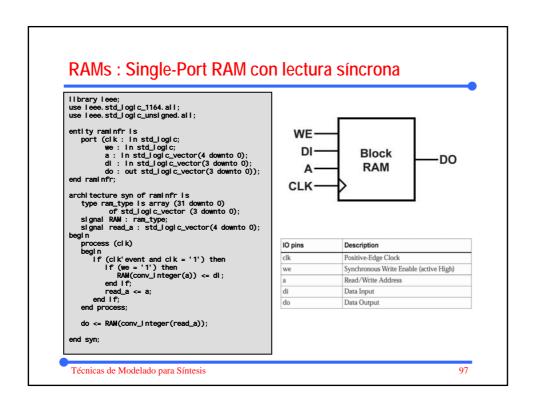
En este modo, cuando se realiza una escritura sobre una posición de la memoria RAM, el valor presente en la salida de la memoria no se modifica.

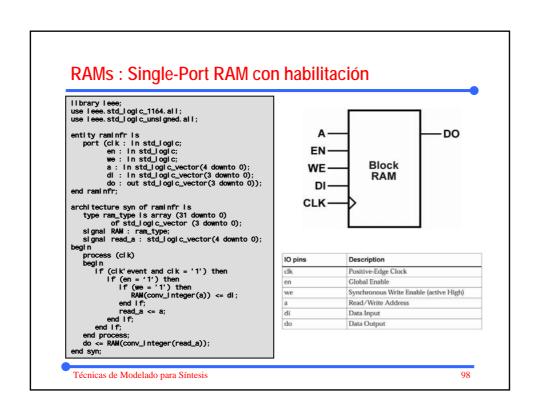
P. Ej., supongamos que en la posición 20 está el valor 5, en la 21 está el valor 8, se accede en lectura a la posición 21 y en escritura a la posición 20 para poner un 10. Tras el acceso de lectura, la RAM muestra en su salida el valor 8 y tras el acceso de escritura modifica el contenido de su posición 20 al valor 10, pero su salida sigue mostrando el valor 8.

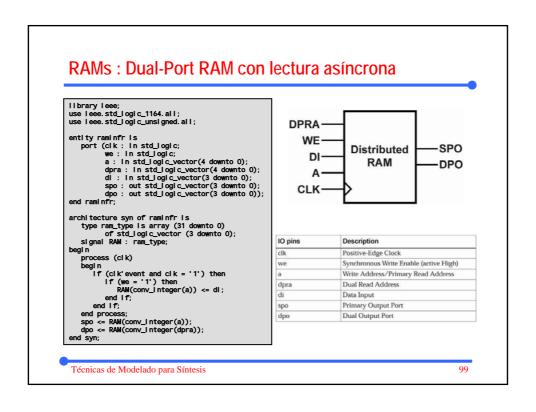
Técnicas de Modelado para Síntesis

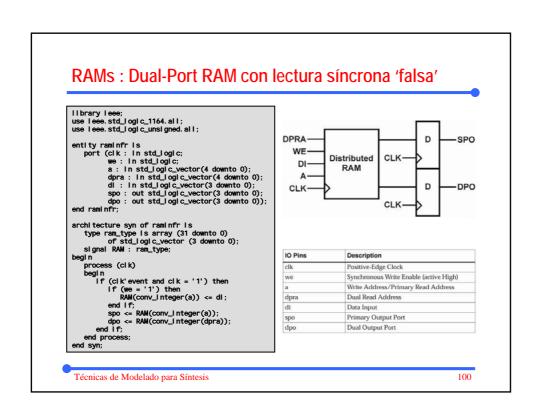


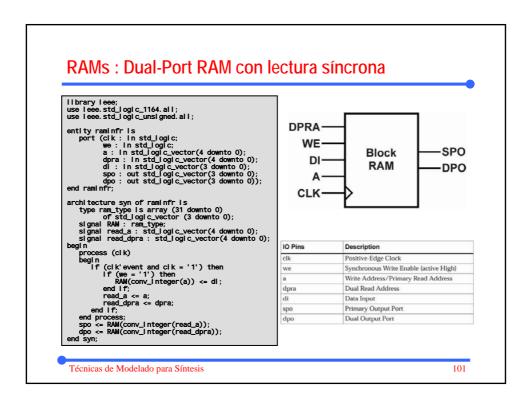


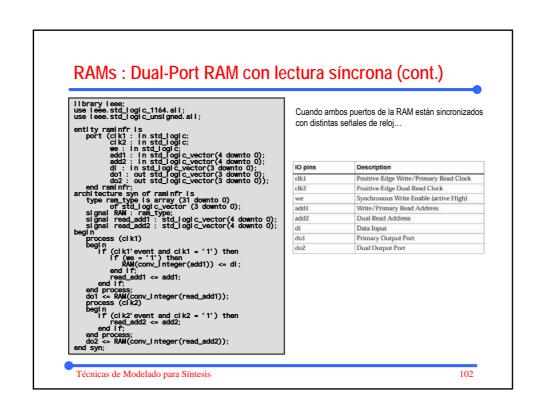


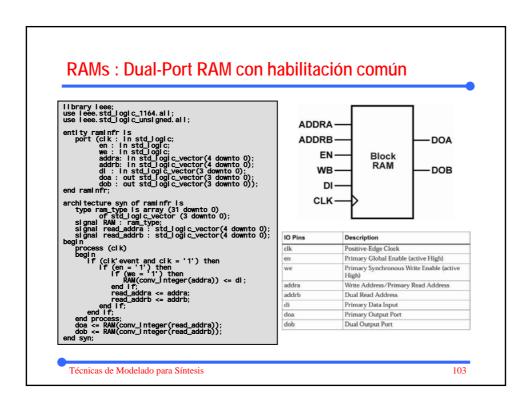


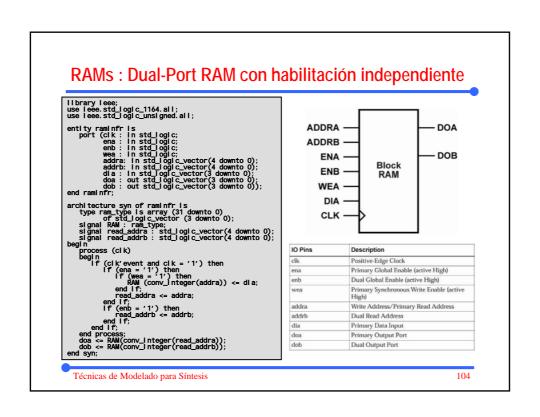


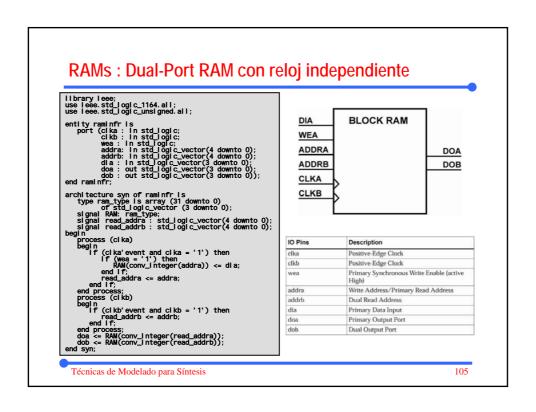


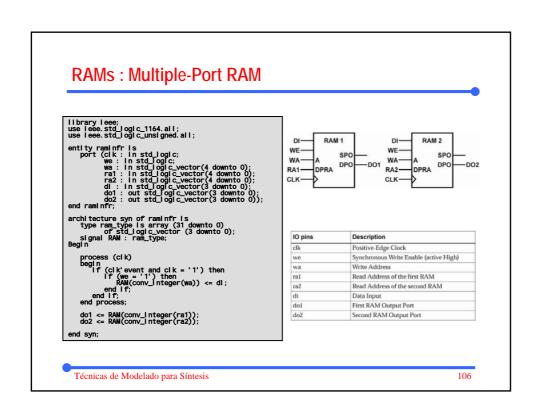






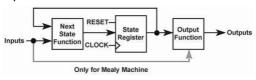






## Máquina de Estados Finitos (FSM)

- Sólo se reconocen las FSM síncronas.
- La codificación de los estados puede alterarse para obtener implementaciones optimizadas según diversos parámetros.
- Esquema típico...



- Típicamente se puede describir con 1, 2 ó 3 procesos.
- · Directivas relacionadas...
  - > FSM\_extract ... habilita o no el reconocimiento de FSMs.
  - > FSM\_encoding ... selecciona el tipo de codificación.
  - > FSM\_fftype ... selecciona el tipo de flip-flops para el registro de estado
  - > ENUM\_encoding... valores binarios para los estados enumerados.

Técnicas de Modelado para Síntesis

107

### **FSM: Secciones**

- Registros de estado...
  - Deben ser inicializados con un reset (asíncrono o no).
  - Su tipo suele definirse como una enumeración que incluye todos los posibles estados.
- Función de estado siguiente...
  - Puede definirse en el mismo procedimiento secuencial o en un procedimiento combinacional separado (y su lista sensible debe mostrar la señal de estado y todas las entradas).
- Salidas
  - Las registradas deben ir dentro del mismo proceso secuencial.
  - Las no registradas en proceso combinacional o sentencias concurrentes.
- Entradas...
  - Señales internas asignadas dentro del proceso secuencial.

Técnicas de Modelado para Síntesis

# FSM: Técnicas de Codificación de Estados

- Auto...
  - La herramienta selecciona automáticamente el método de codificación óptimo para la máquina descrita.
- One-Hot...
  - Asocia un bit de código (1 flip-flop) a cada estado (001, 010, 100).
  - Apropiado en FPGAs con elevado número de flip-flops.
  - Optimiza la velocidad y reduce la disipación de potencia.
- Gray...
  - Garantiza el cambio de una única variable de estado cuando se produce una transición entre estados consecutivos.
  - Adecuado para FSM con muchos estados en línea y sin saltos.
  - Reduce la presencia de glitches.
  - Resultados óptimos cuando el registro de estado usa flip-flops tipo T.

Técnicas de Modelado para Síntesis

100

# FSM: Técnicas de Codificación de Estados (cont.)

- Compacto...
  - Minimiza el número de variables de estado (y flip-flops).
  - Adecuado para la optimización de área.
- Johnson...
  - Adecuado para muchos estados en línea y sin saltos.
- · Secuencial...
  - Identifica trayectos lineales largos y asigna códigos binarios sucesivos a sus estados.
  - Minimiza las ecuaciones de estado siguiente.
- Usuario...
  - Usa la codificación indicada por el usuario en el código fuente.
  - Si se define un tipo enumerado se combina con la directiva enum\_encoding para asignar valores binarios concretos a cada estado.

Técnicas de Modelado para Síntesis

