Diseño Automático de sistemas

Diseño de Sistemas Digitales

Prof. Pablo Sarabia Ortiz

UNIVERSIDAD NEBRIJA

Objetivos Clase

- Entender los diferentes niveles y dominios de un sistema digital.
- Entender el rol del diseñador en cada uno de ellos y las metodologías/herramientas disponibles.
- Entender el flujo de diseño de un sistema digital
- Comprender algunos de los problemas de un diseño real.

Bibliografia

- **Capitulo 1**
  Pong P. Chu (2006), RTL Hardware Design Using VHDL: Coding for Efficiency, Portability, and Scalability, Willey, 1st Edition

Contenidos

1. Objetivos del diseño
2. Dominios y niveles de abstracción
3. Metodologías de diseño hardware
4. Flujo de diseño
5. Herramientas de diseño hardware
6. Consideraciones prácticas

Objetivos diseño digital

Los objetivos, diferente de las especificaciones, que debemos tener en cuenta son:

- Eficiencia: Va a depender de la descripción inicial que hagamos del sistema. Debe ir acorde con el hardware disponible.
- Escalabilidad: Cada módulo debe facilitar el diseño del conjunto del sistema. Facilitar la modularidad y reusabilidad.
- Portabilidad: Ser independientes del dispositivo y de la herramienta de diseño utilizada

Dominios y niveles de abstracción

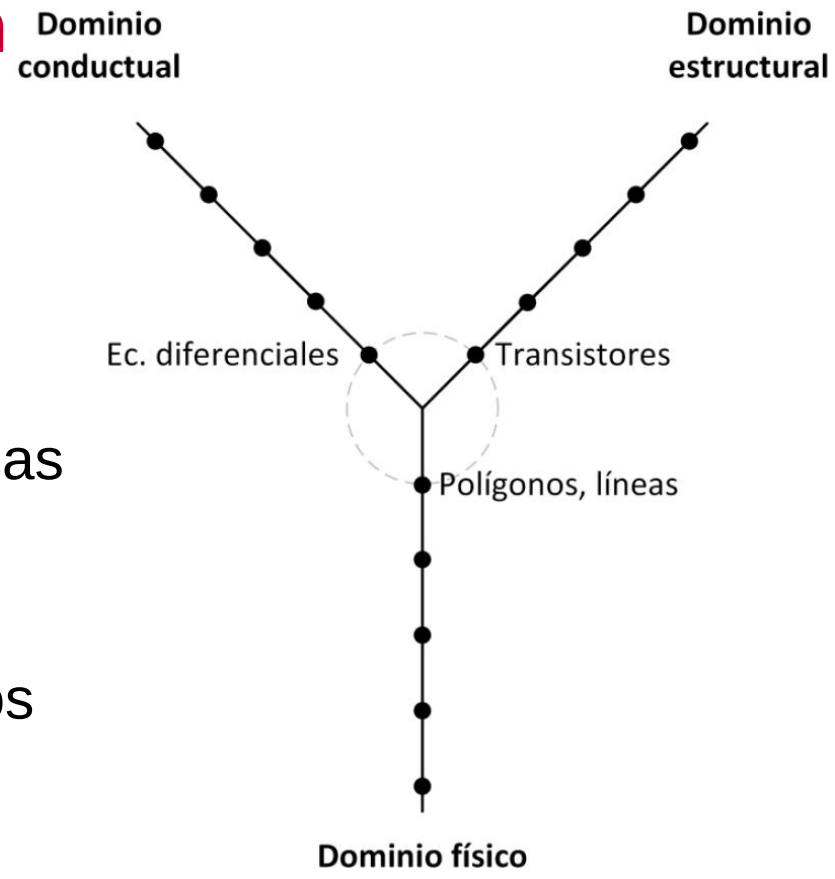
Todos los sistemas digitales se pueden describir según tres dominios/áreas:

- Conductual: según la función
- Estructural: según los elementos que lo conforman
- Físico: según la ubicación y los elementos que lo componen

Niveles de abstracción

Nivel Circuital

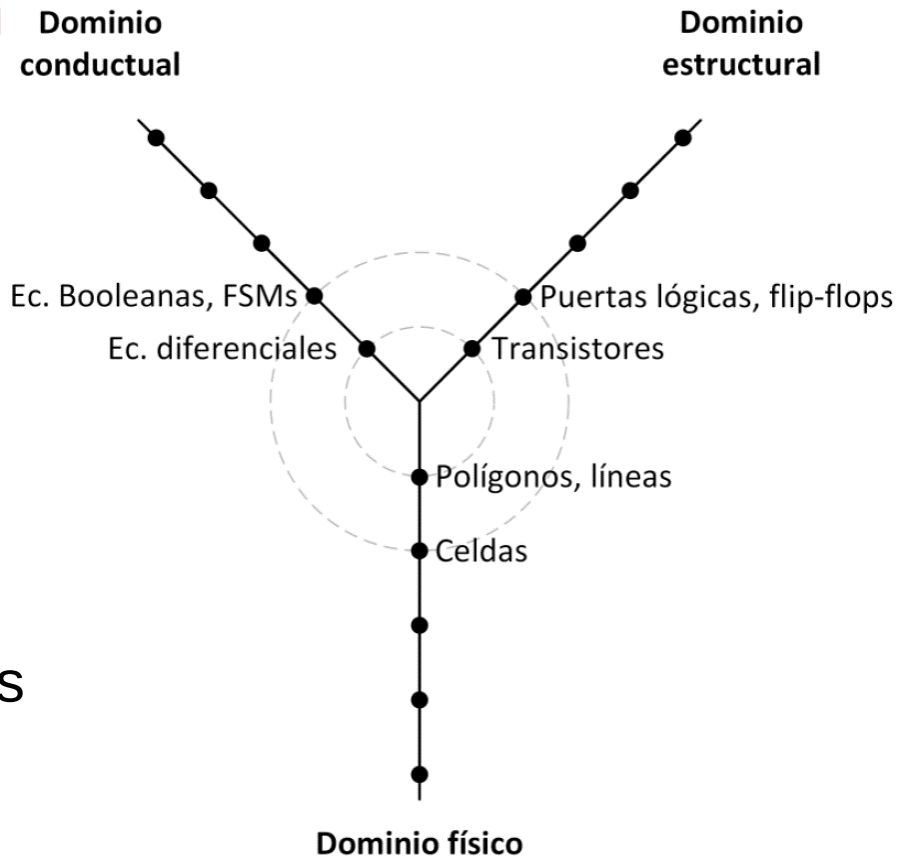
- **Tiempo**: continuo
- **Variables**: magnitudes físicas continuas
- **Campo**: Electrónica
- **Métricas**: Área real, tiempos subida y bajada, etc …



Niveles de abstracción

Nivel Lógico

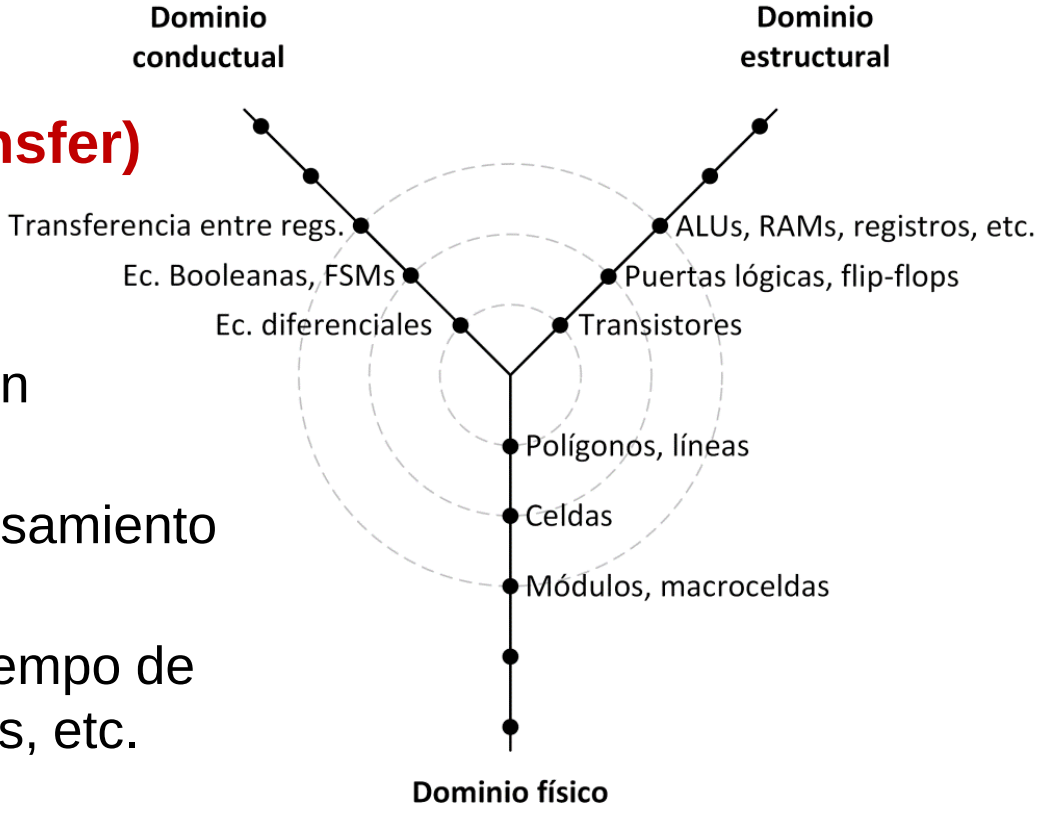
- **Tiempo**: continuo
- **Variables**: magnitudes discretas booleanas
- **Campo**: computación
- **Métricas**: área real, tiempos de propagación, etc.



Niveles de abstracción

Nivel RT (Register Transfer)

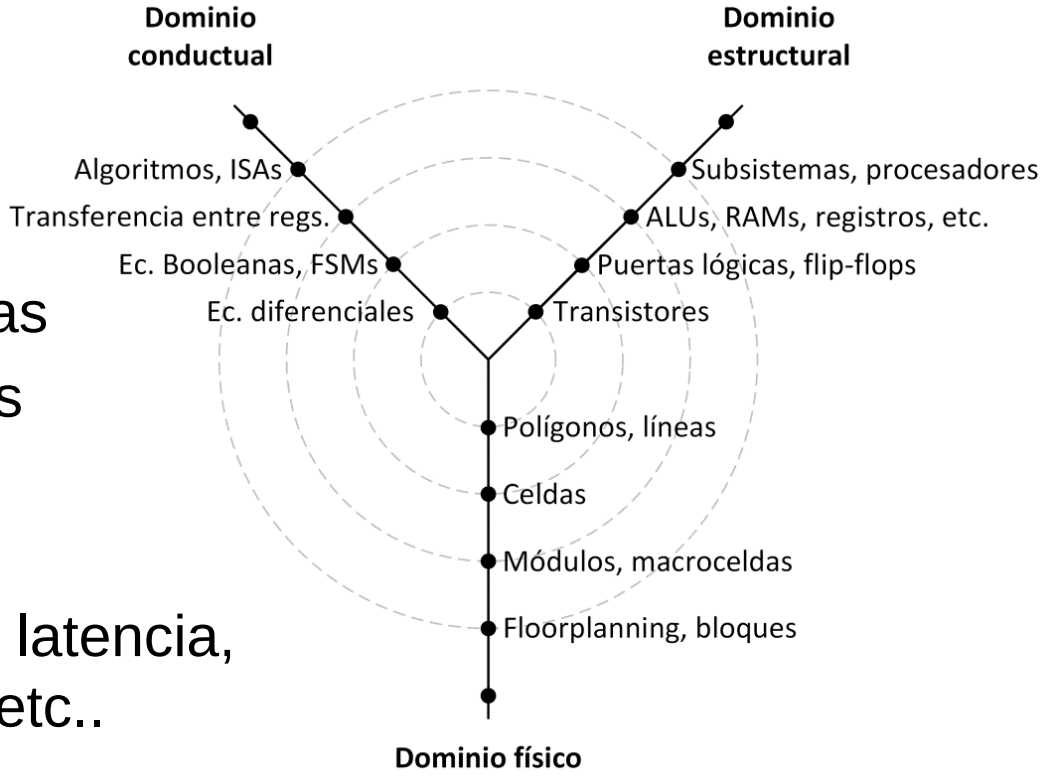
- **Tiempo**: discreto
- **Variables**: agrupados en palabras (datos)
- **Campo**: control y procesamiento de datos
- **Métricas**: se mide en tiempo de ciclo, número de puertas, etc.



Niveles de abstracción

Nivel Algorítmico

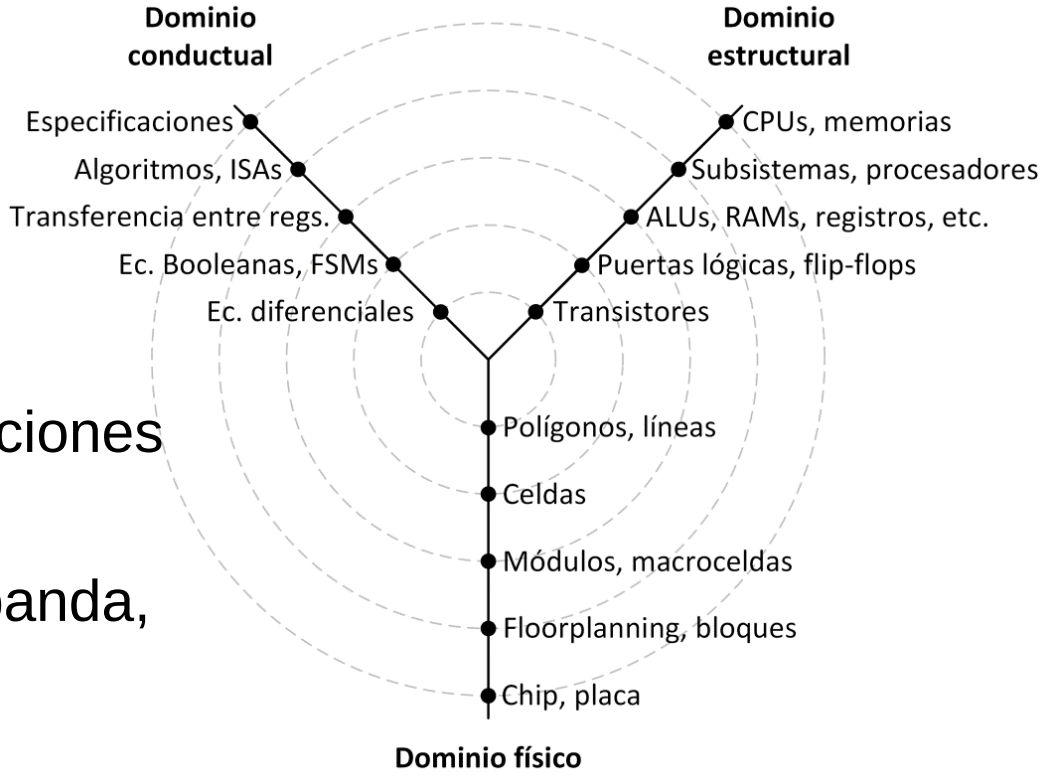
- **Tiempo**: dependencias
- **Variables**: estructuras abstractas
- **Campo**: algoritmia
- **Métricas**: se mide en latencia, número de módulos, etc..



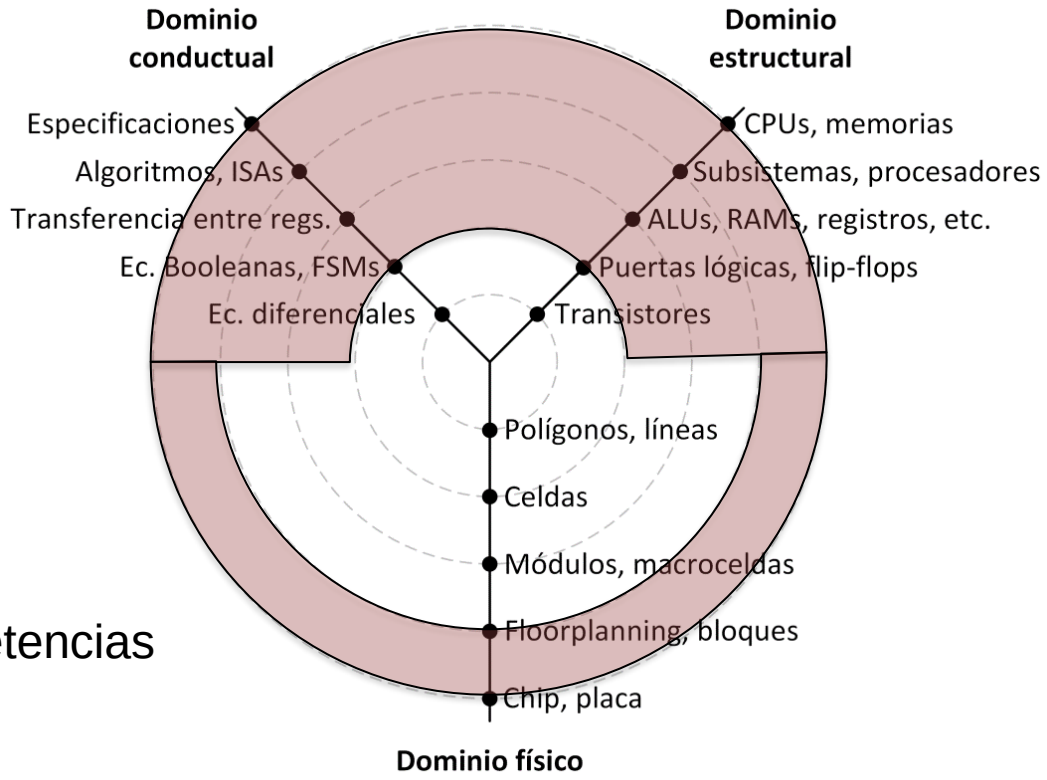
Niveles de abstracción

Nivel Sistema

- **Tiempo**: relaciones
- **Variables**: especificaciones
- **Campo**: descriptivo
- **Métricas**: ancho de banda, MIPS, etc …



Niveles de abstracción



Competencias

Metodologías de diseño

La metodología de diseño depende de la aproximación que realicemos al sistema digital tenemos:

- Arquitectura: según la libertad del diseñador
- Enfoque: según la técnica utilizada para realizar el diseño
- Descripción: según la descripción que hagamos del diseño

Arquitectura: Full Custom

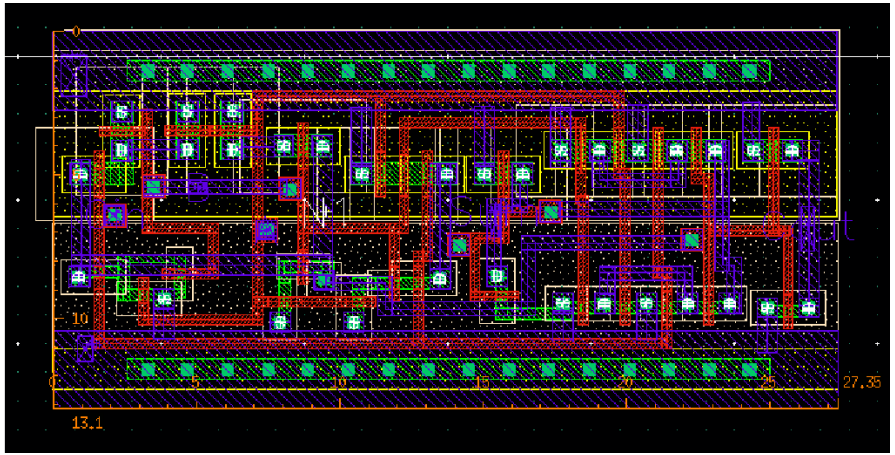
Consiste en diseñar a nivel del silicio, dibujando el propio diseñador los componentes.

Ventajas:

- Flexibilidad
- Alto rendimiento

Desventajas:

- No es automatizable
- Complejo
- Coste Elevado
- Tiempo de salida a mercado elevado



Arquitectura: Semi Custom

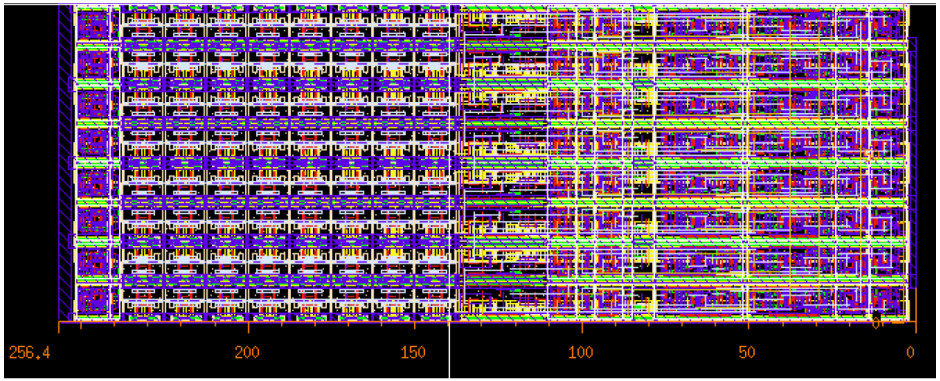
Se utilizan células standard proporcionadas por el fab.

Ventajas:

- Automatizable
- Funcionamiento eléctrico asegurado
- Tiempo de diseño menor

Desventajas:

- El rendimiento es peor que el full custom



Enfoque

- Diseño bottom-up: Describimos primero los componentes y luego construimos el sistema agrupándolos.

  Útil diseños pequeños.

- Diseño top-down: Partimos de la idea de alto nivel y vamos subdividiendo en tantos niveles y módulos como sea necesario.

  Necesario en diseños grandes permite decidir la implementación al final.

Descripción

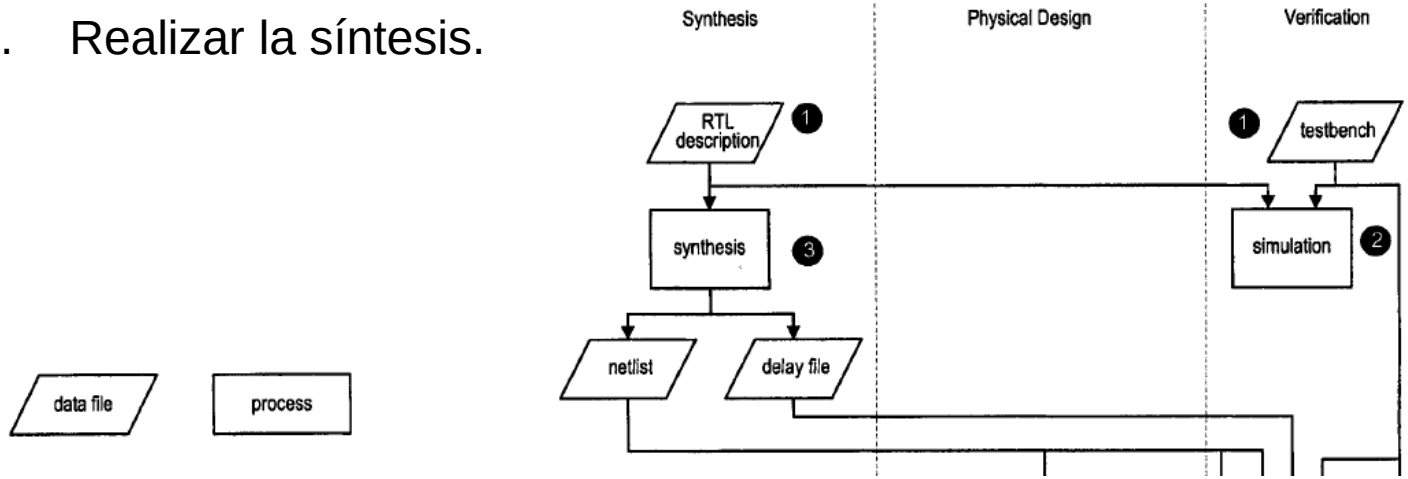
- Diseño data-flow: utilizamos ecuaciones lógicas y las especificaciones del diseño. Es necesario conocer el funcionamiento del circuito a bajo nivel.

- Diseño conductual: utilizamos el comportamiento del circuito para describirlo mediante processy condiciones if-else, case-when, etc.

- Diseño estructural: utilizamos componentes ya definidos con alguno de los modos anteriores y los conectamos entre sí para conseguir el funcionamiento deseado. Es una forma de diseño a alto nivel ya que no necesitamos conocer el funcionamiento interno de los componentes, sólo sus conexiones

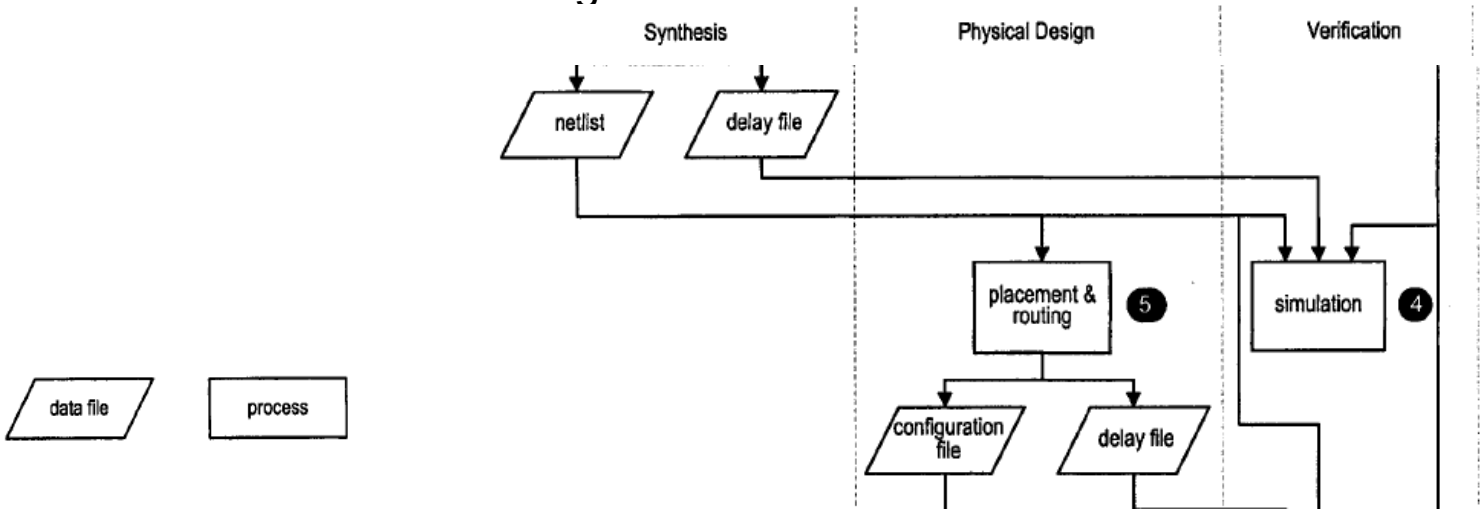
Flujo de diseño: Comportamiento

1. Desarrollar archivo de diseño y síntesis.
2. Usar el archive de diseño como descripción del circuito para verificar el diseño funcional.
3. Realizar la síntesis.



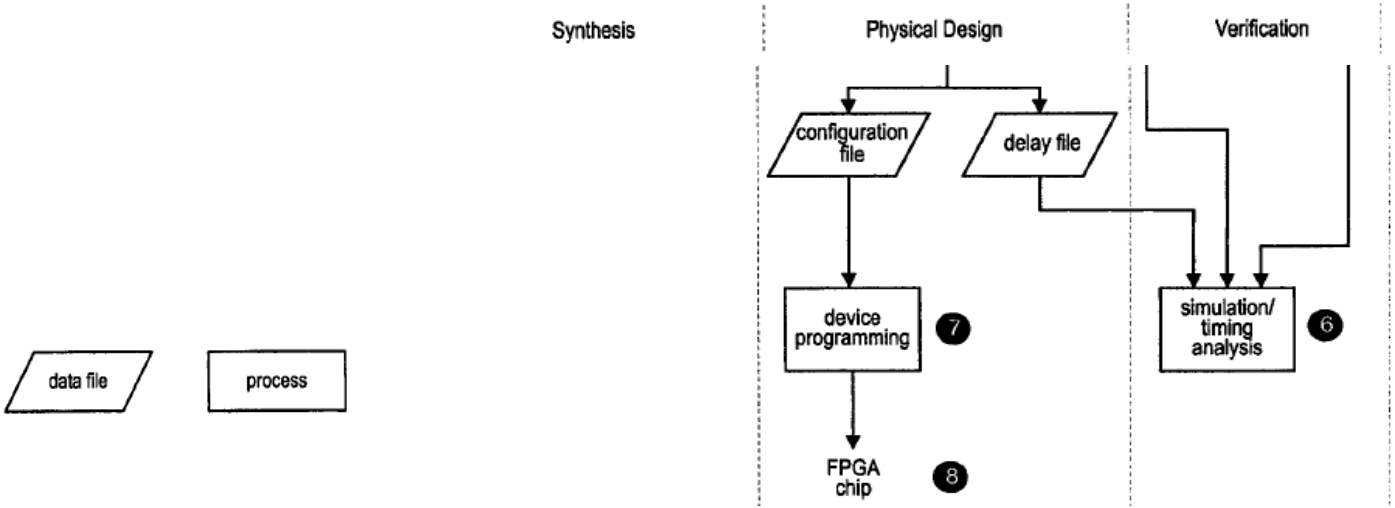
Flujo de diseño: Síntesis

4. Usar la salida de la síntesis (netlist) como descripción del circuito y usarla para verificar correcta síntesis y tiempos preeliminares.
5. Placement and routing.



Flujo de diseño: Placement and Routing

6. Usar la salida del placement and routing para verificar tiempos
7. Generar el bitstream
8. Verificar en el dispositivo físico



Herramientas de diseño

El programador tiene a su disposición diferentes herramientas para el diseño en las diferentes etapas:

- Descripción: Describir el funcionamiento del diseño.
- Simulación: Evaluar el funcionamiento del diseño.

Herramientas de diseño: Descripción HW

- Esquemáticos: diagramas de cajas y flechas que representan la **estructura** del sistema. Pueden incluir información sobre tiempos, señales, etc.
- Grafos, diagramas de flujo o redes de Petri: permiten describir el sistema desde el punto de vista funcional o de **comportamiento**.
- Lenguajes de descripción hardware: permiten **describir** un circuito digital desde diferentes niveles de abstracción. Ej: VHDL, Verilog.

Herramientas de diseño: Simulación HW

- Simulación funcional: Estudia el funcionamiento a nivel de **algoritmo/sistema**. Ej: Vivado, ModelSim
- Simulación post-síntesis: Estudia el funcionamiento a nivel de **lógico/RT**, es dependiente del hardware. Ej: Vivado
- Simulación eléctrica: Estudia el funcionamiento a nivel circuital. Es una simulación de los transistores, resistencias y demás elementos básicos. Ej: Cadence

Herramientas de diseño: Software

- Software de diseño de circuitos integrados: permite crear diseños físicos trazando líneas, componentes, conexiones, etc. Ej: Cadence, Eagle, Synopsys

- Software de implementación: permite obtener datos de utilización de recursos, tiempos y consumo del sistema. Ej: Vivado, Cadence, Synopsys

- Software de programación de dispositivos: permite implementar el sistema diseñado en un dispositivo físico para comprobar su funcionalidad. Ej: Vivado

Consideraciones prácticas

Las entradas externas generadas con un interruptor o un pulsador accionados por un ser humano tienen una serie de problemas:

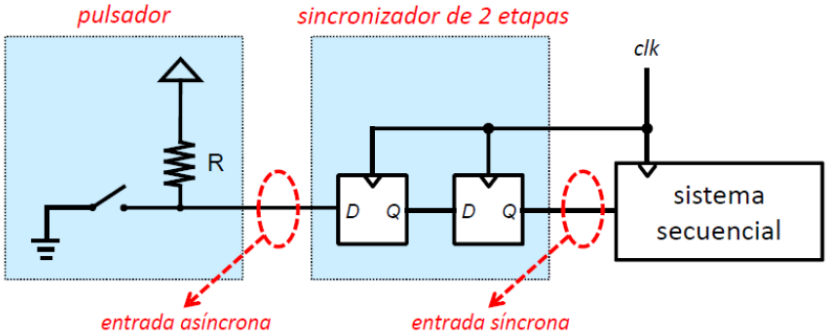
- Son señales asíncronas: pueden cambiar en cualquier momento independientemente del reloj del sistema
- Tienen rebotes: la naturaleza mecánica de los pulsadores hace que cada pulsación se pueda interpretar como una sucesión de pulsos.
- Son de baja frecuencia: una única pulsación se prolonga durante un gran número de ciclos de reloj consecutivos

Problema: Señales asíncronas

- Vivado garantiza que los circuitos internos tengan un comportamiento predecible.
- Sin embargo, una señal externa puede cambiar en cualquier momento.
- Solución: **Sincronizador**

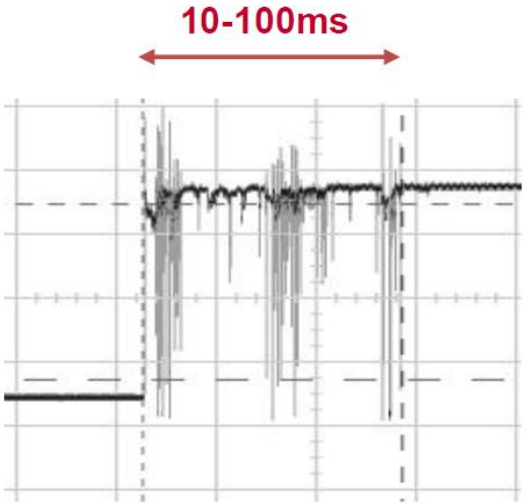
Problema: Sincronizador

- Un sincronizador retras los cambios de señal para evitar la metaestabilidad, hablaremos más adelante de ello.
- La señal de salida (sincronizada) será una señal gestionable por Vivado
- Se crea encadenando registros.



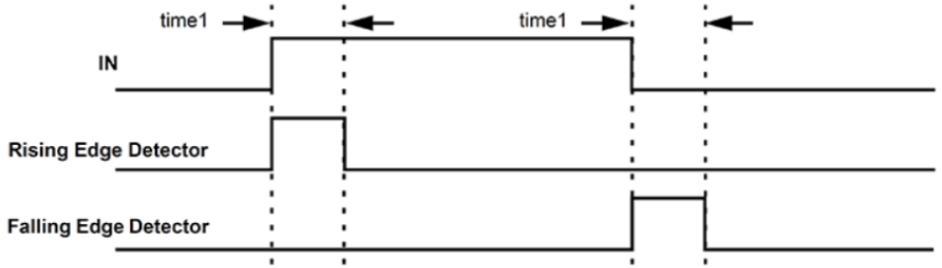
Problema: Efecto rebote

- Un actuador mecánico tarda un tiempo en alcanzar un estado estable.
- Se puede eliminar por SW o HW
- La idea es que si ha transcurrido un tiempo razonable el contenido del registro es igual al del pulsador.



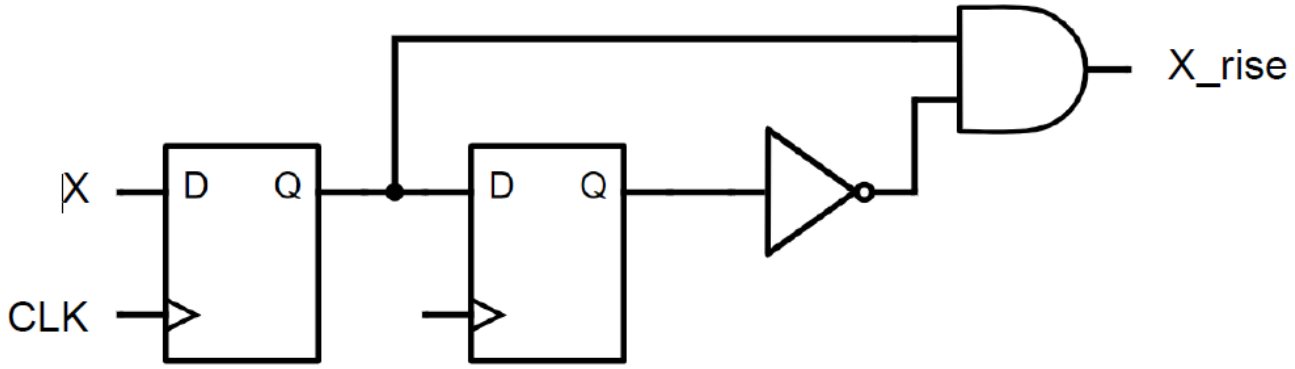
Problema: Señales de baja frecuencia

- Los sistemas digitales típicamente funcionan a velocidades más altas que las acciones humanas.
- La señal de entrada dura varios ciclos de reloj
- Mediante un detector de flancos convertimos esta señal en pulsos que duran un único ciclo de reloj.



Problema: Señales de baja frecuencia

- Detector de flanco de subida:



Resumen

- Seguir las buenas prácticas y tener en mente siempre: Eficiencia, Escalabilidad y Portabilidad
- Entender en que ámbito (conductual, estructural y físico) trabajamos y el nivel en el que estamos.
- El flujo de trabajo habitual en un diseño de sistemas digitales incluye diferentes simulaciones cada una de ellas baja en el nivel y se parece más al HW real.

Resumen

- Un diseño suele usar más de una metodología de diseño.
- Las señales asíncronas (no registradas), pueden ser problemáticas si no se las trata.

A continuación

- Repaso a lógica digital.
- Repaso VHDL
- Proyecto.

¿Preguntas?

UNIVERSIDAD
NEBRIJA