

# Nuevos paradigmas de diseño de Sistemas electrónicos sobre silicio (SoC)

Prof. Dr. Sebastian Eslava G.

Departamento de Ingeniería Eléctrica y Electrónica  
Universidad Nacional de Colombia

# Contenido

- **Introducción**
- **Diseño de *SoC***
  - Diseño a nivel de sistema.
  - Exploración del espacio de diseño.
  - Verificación funcional y análisis de desempeño.
  - Modelamiento.
- **Conclusiones**
- **Preguntas**

# Mundo Digital



## ► Facilitadores

- Procesos de fabricación
- Herramientas de diseño
- Usuarios
- Creatividad



# Diseño de Sistemas digitales

- Metodologías tradicionales (*ASICs*)

- *VHDL y Verilog*
- *Top-down (idea)*

- Síntesis doble

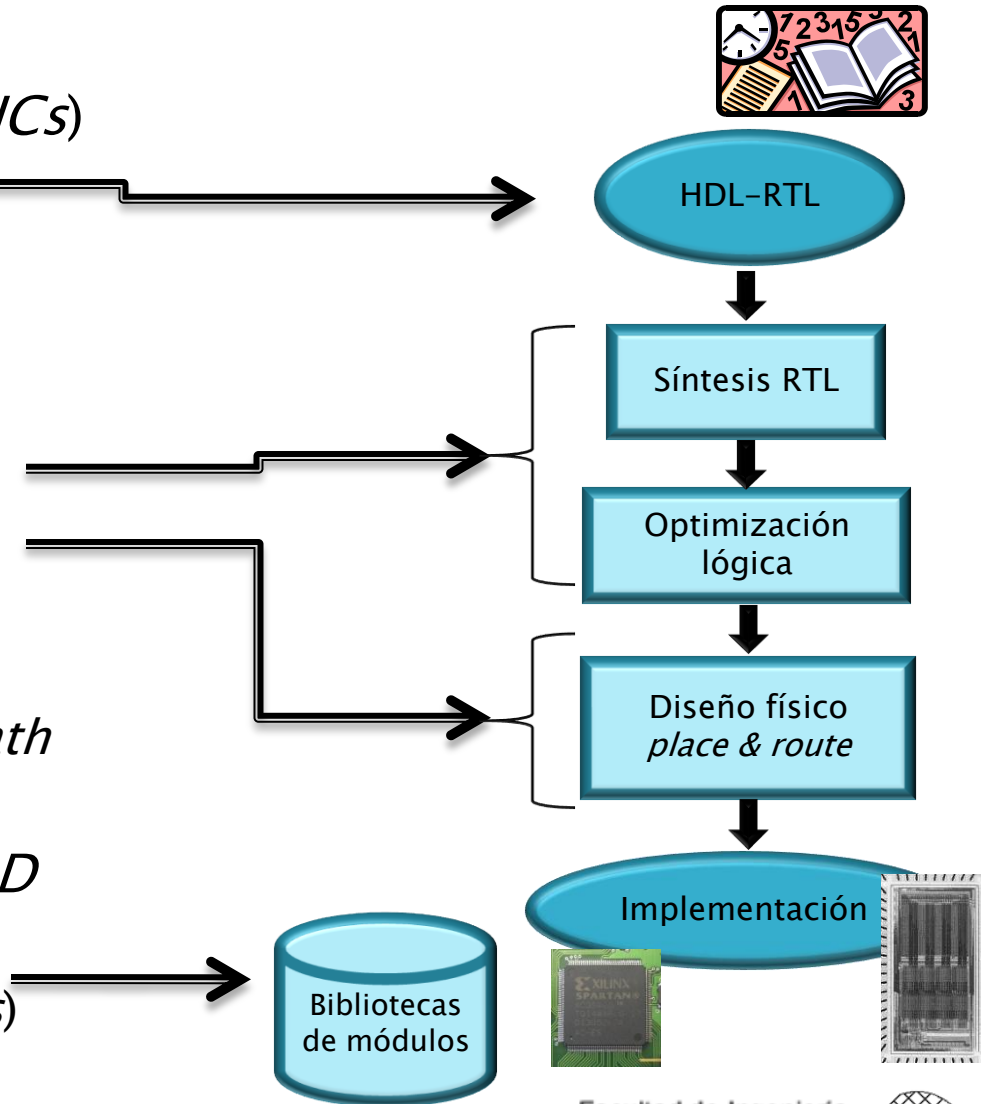
- Lógica (*netlist*, macro celdas)
- *Layout* (máscaras, *bitstreams*)

- Arquitectura *FSMD*

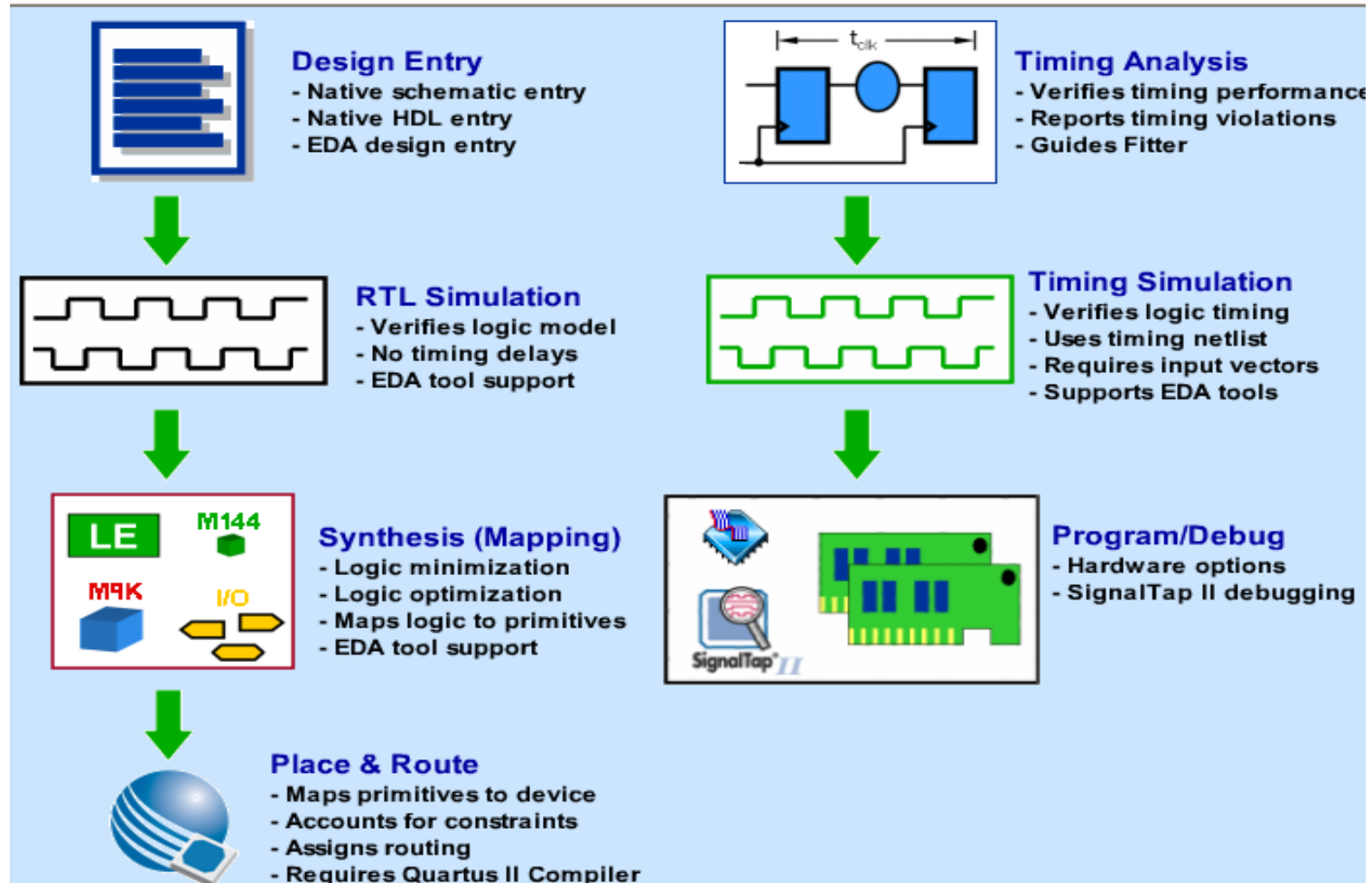
- *Finite state machine with datapath*

- Diversidad de herramientas *CAD*

- Bibliotecas de componentes
- Kits de diseño (*Silicon Foundries*)

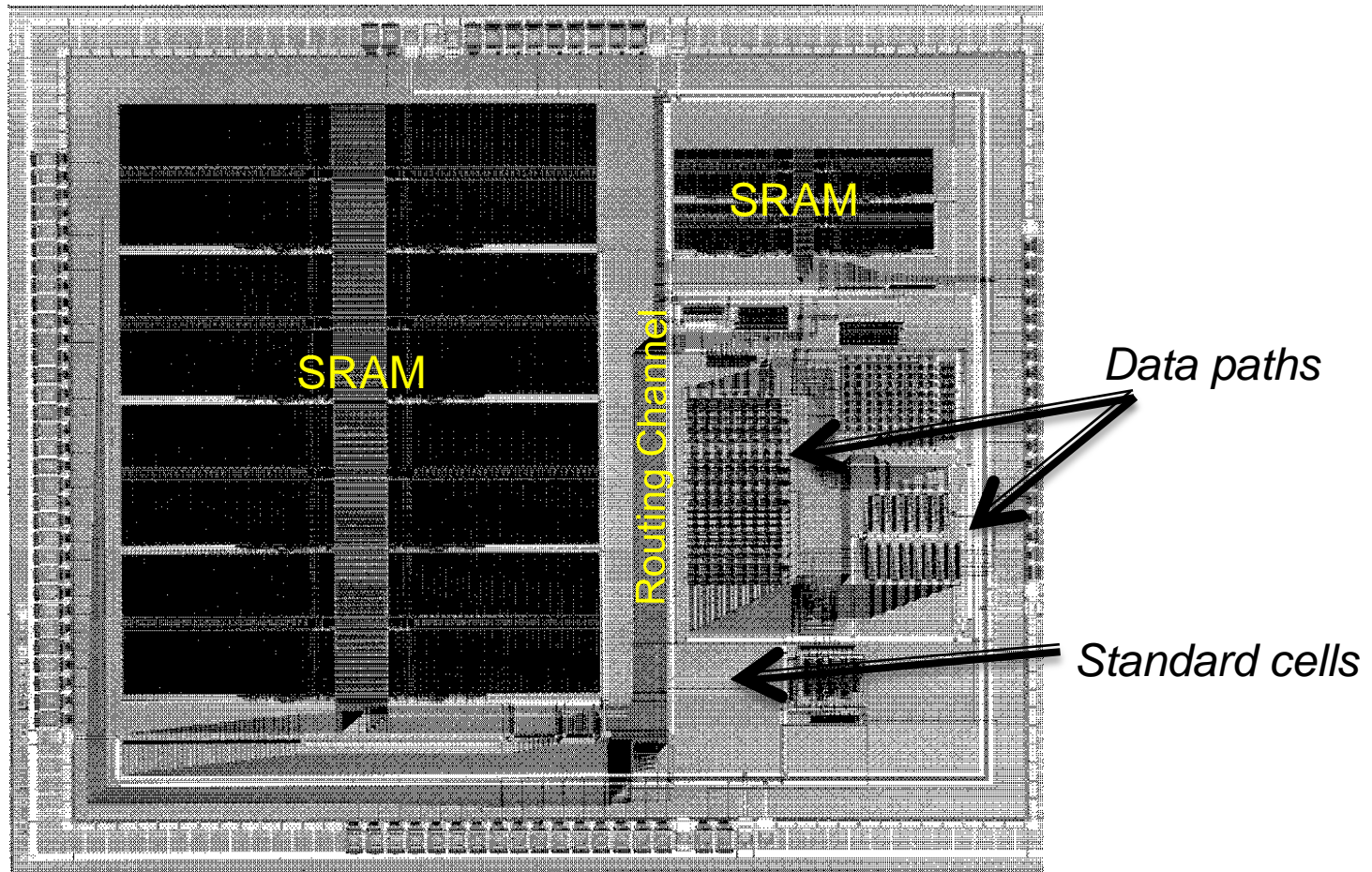


# Diseño de Sistemas digitales





# Ejemplo de sistemas digitales (ASIC)



Chip codificador de video [Brodersen92]

Referencia: *Digital Integrated Circuits: A Design Perspective* © 1996 Jan Rabaey

# Sistemas digitales modernos

- Sistemas completos
  - *Hardware–software*
- Elementos hechos a la medida
  - Escalables
- Implementación en un único circuito integrado
  - *System on Chip, SoC*

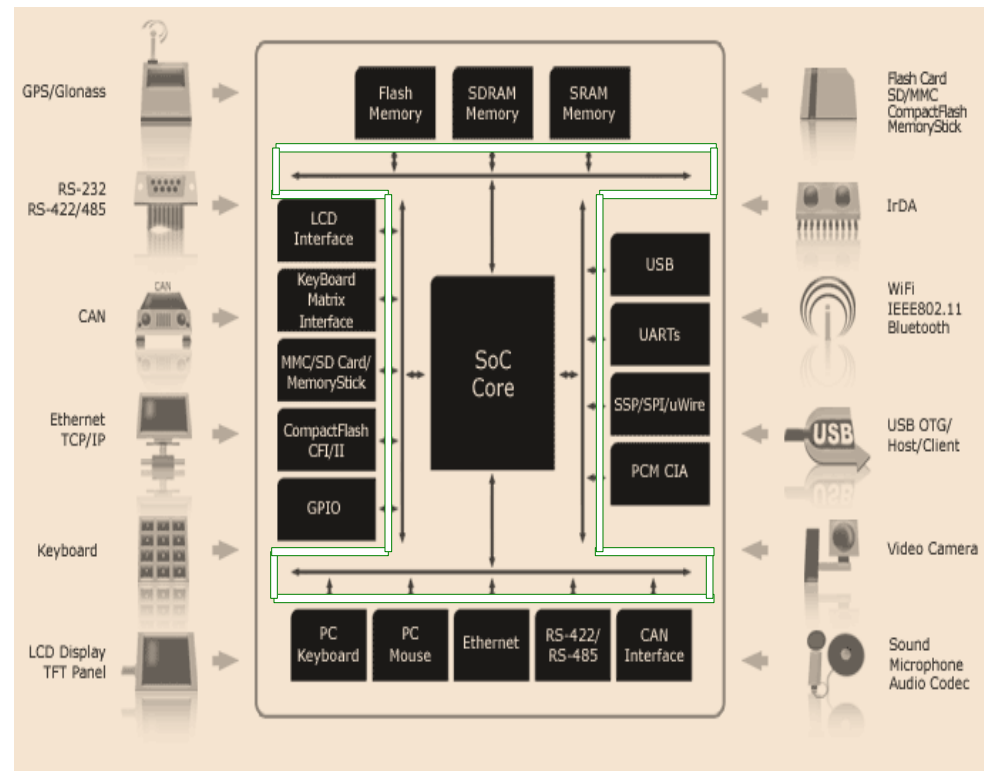
# Sistemas digitales modernos

## ► Sistemas complejos

- Componentes
- Funcionalidad

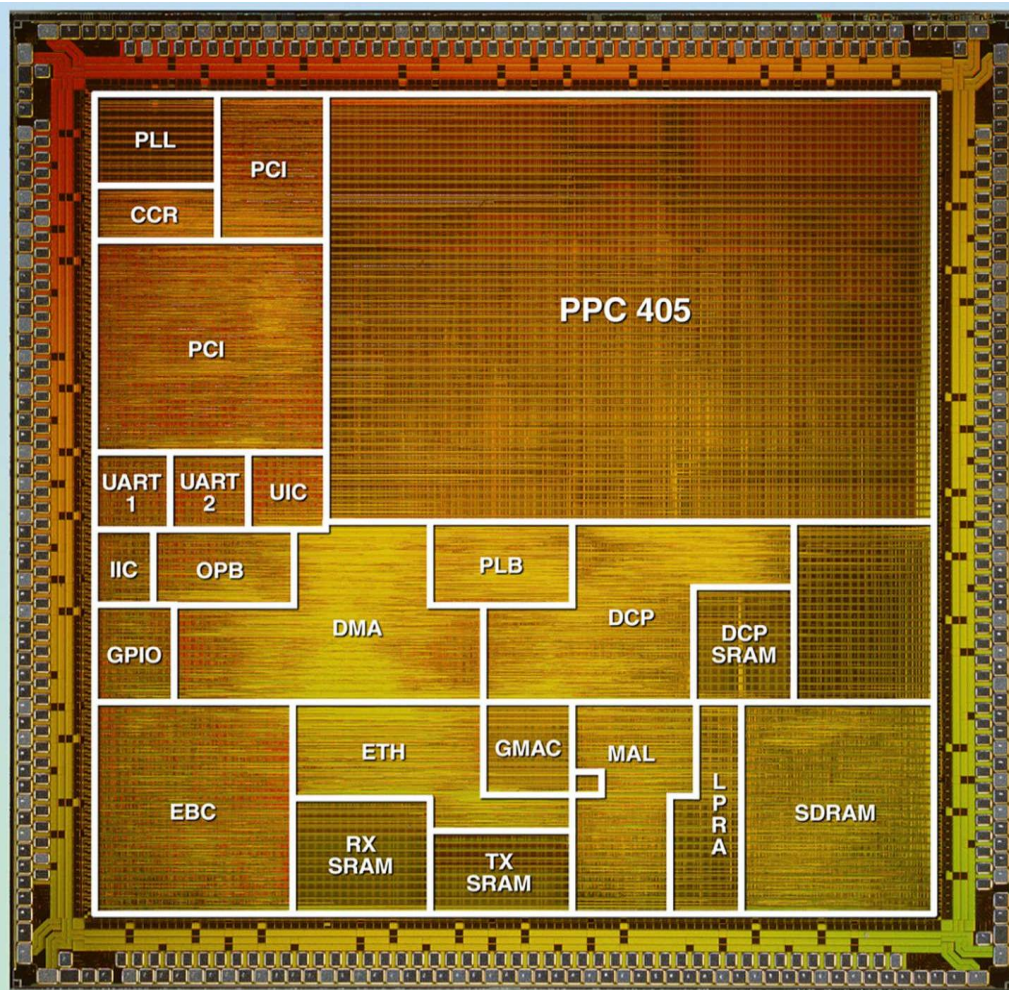
## ► Metodologías de diseño

- *Top-Down*
- Modularidad
- Jerarquía





# Ejemplo de un SoC



Un SoC contiene:

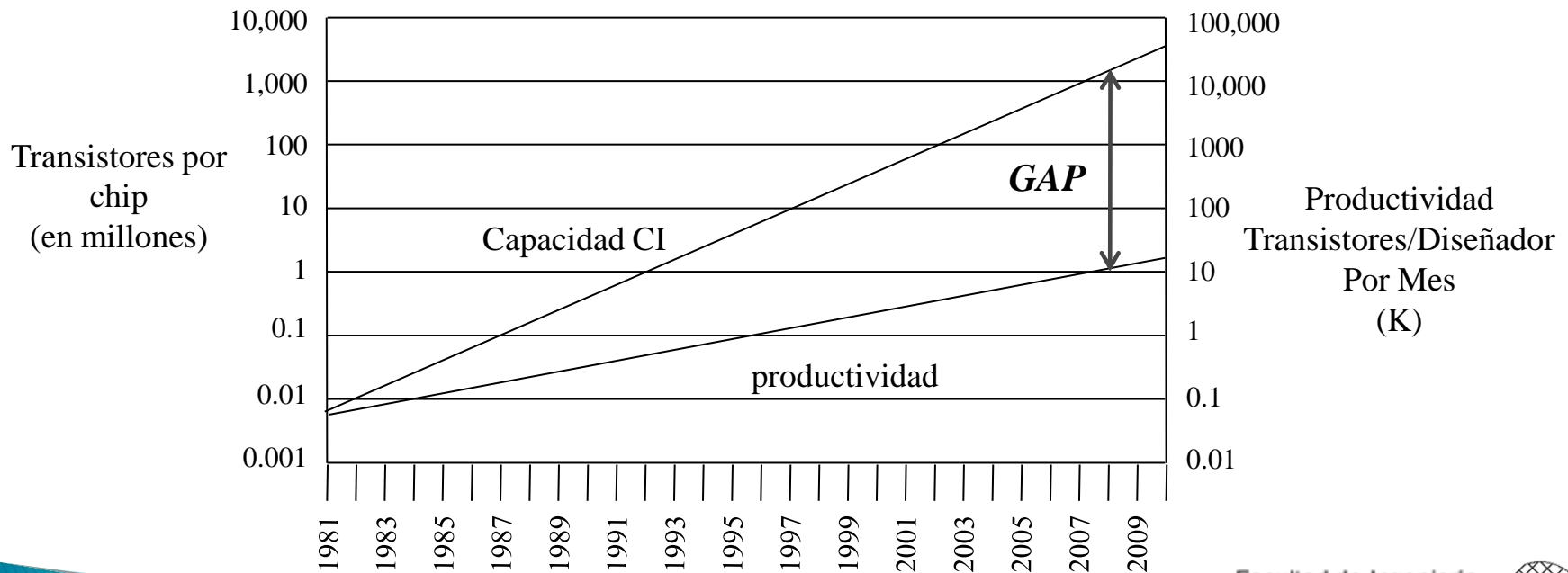
- Procesadores
- Módulos IP reutilizados
- Memorias (internas y externas)
- Interfaces (USB, PCI, Ethernet)
- Software (interno y externo)
- Bloques analógicos–digitales
- Hardware programable (FPGA)
- >> 500K puertas lógicas

# Diseño de SoC

- RTL

- Insuficiente como punto de partida
  - Sistemas con millones de puertas lógicas
  - Esfuerzo computacional → Tiempo de simulación
    - 53 horas para 1 seg, procesador de red.

- Gap de productividad



# Nuevos paradigmas de diseño

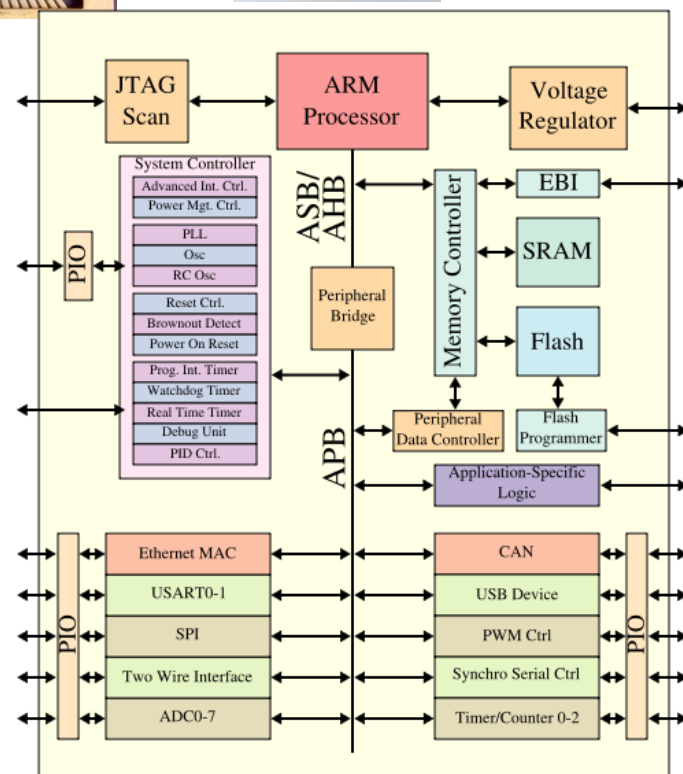
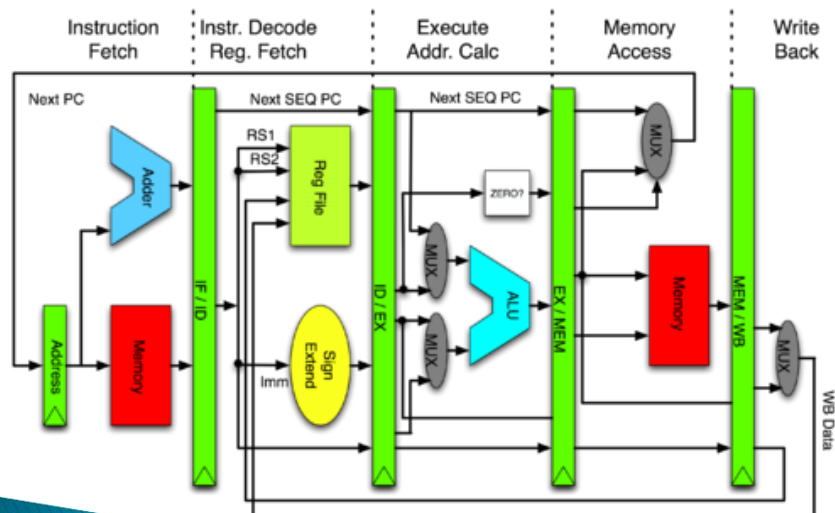
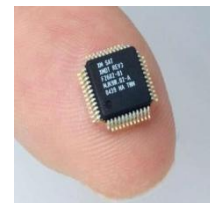
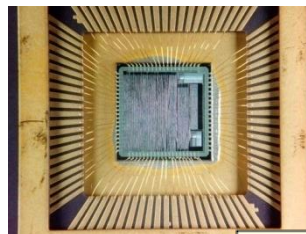
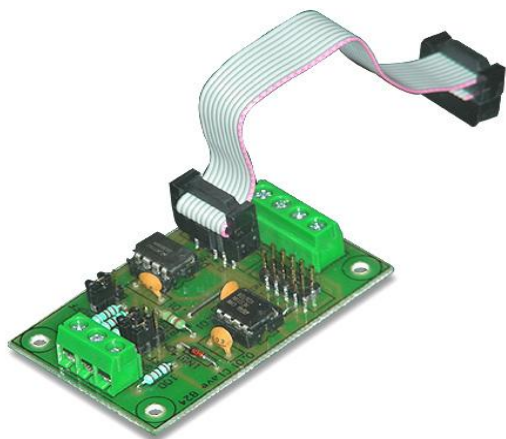
- Reducir el gap
  - Aumentar la productividad
- Manejar la creciente complejidad
  - De ASIC para *SoC*
- RTL no desaparece

## *Evolución del diseño*

	De ASIC	Para SoC
Objetivo del diseño	Módulo de hardware	Múltiples módulos de <u>hardware</u> y <u>software</u>
Abstracción del diseño	Nivel de circuito RTL	Nivel de Sistema SLD



# Evolución del Diseño



# Contenido

- Introducción
- **Diseño de *SoC***
  - Diseño a nivel de sistema
  - Exploración del espacio de diseño
  - Verificación funcional y análisis de desempeño
  - Modelamiento
- Conclusiones
- Preguntas



# Diseño de *SoC*

## ► Especificación

- Requisitos → modelo ejecutable

Especificación



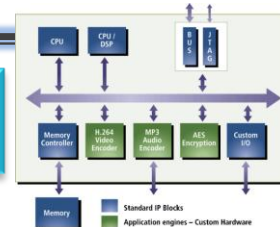
## ► Diseño al nivel de sistema

- Modelo ejecutable → modelo RTL/C

Particionamiento  
HW-SW

Refinamiento  
Sw

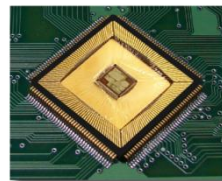
Refinamiento  
Hw



## ► Implementación física

- Modelo RTL/C → solución *HW-SW*
- Metodologías tradicionales

Implementación



# Contenido

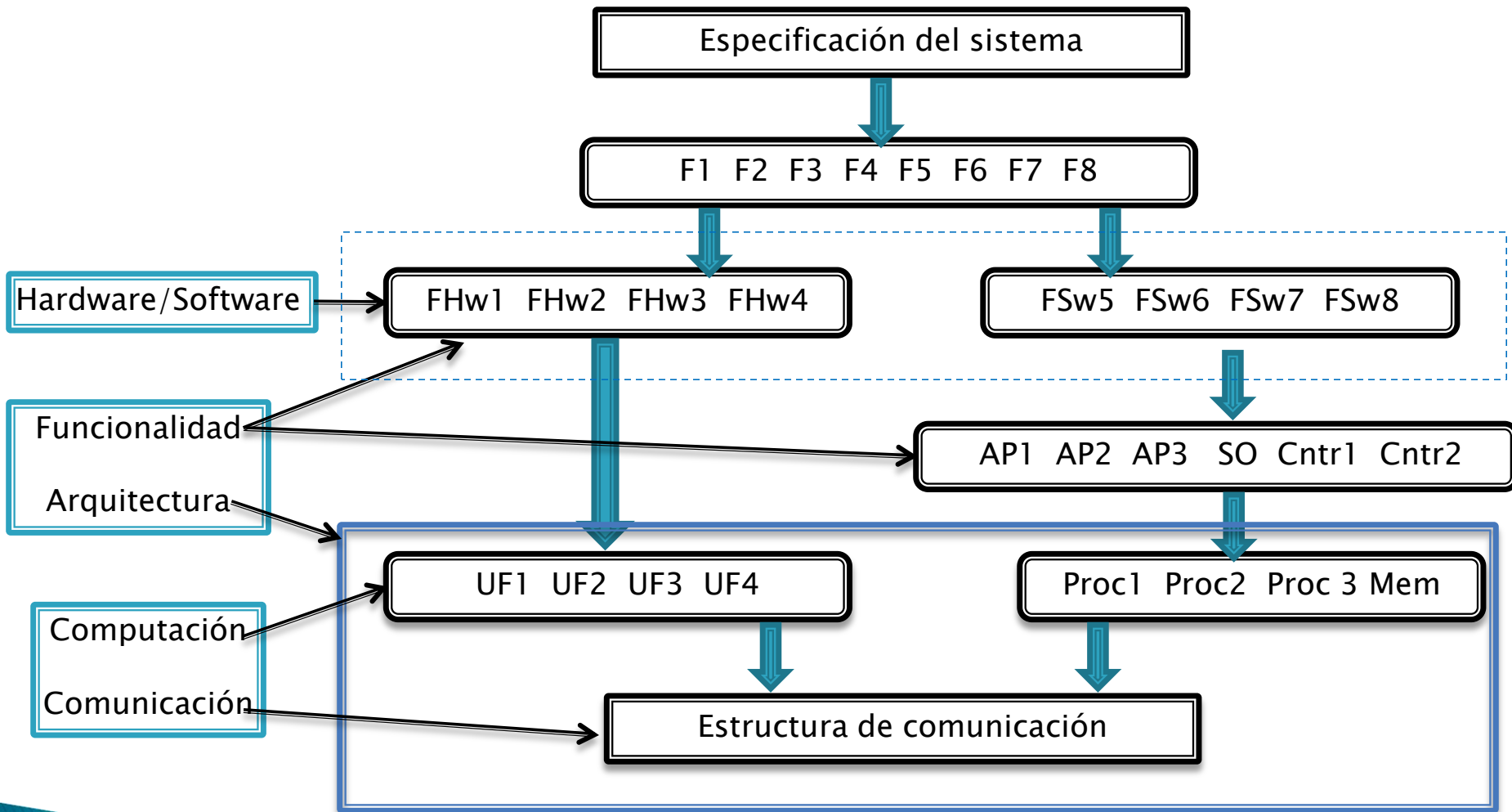
- Introducción
- Diseño de *SoC*
  - **Diseño a nivel de sistema**
    - Exploración del espacio de diseño
    - Verificación funcional y análisis de desempeño
    - Modelamiento
- Conclusiones
- Preguntas

# Diseño a nivel de sistema (*SLD*)

- Aumentar nivel de abstracción
  - Superior a RTL
- Uso de nuevos lenguajes
  - *SystemC*, *SpecC*, *Ptolemy*, *SystemVerilog*, otros
- Segmentar el diseño

Funcionalidad	Arquitectura
Hardware	Software
Computación	Comunicación

# Diseño a nivel de sistema (*SLD*)



# Tareas del nivel de sistema

- Exploración del espacio de diseño
  - Encontrar la solución mas adecuada
- Verificación funcional y análisis de desempeño
  - Modelos funcionalmente
    - Correctos? Completos?
    - Cumplen especificación?
- Modelamiento
  - Modelos usados en todas las tareas



# Contenido

- Introducción
- Diseño de *SoC*
  - Diseño a nivel de sistema
  - **Exploración del espacio de diseño**
  - Verificación funcional y análisis de desempeño
  - Modelamiento
- Conclusiones
- Preguntas

# Exploración del espacio de diseño

*Encontrar la solución mas adecuada que cumpla las especificaciones y restricciones.*

*“No existe una solución única”*

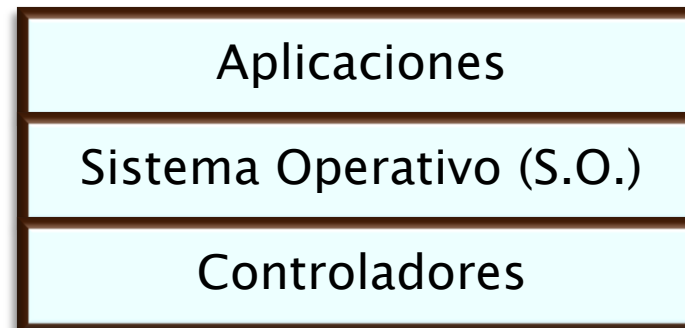
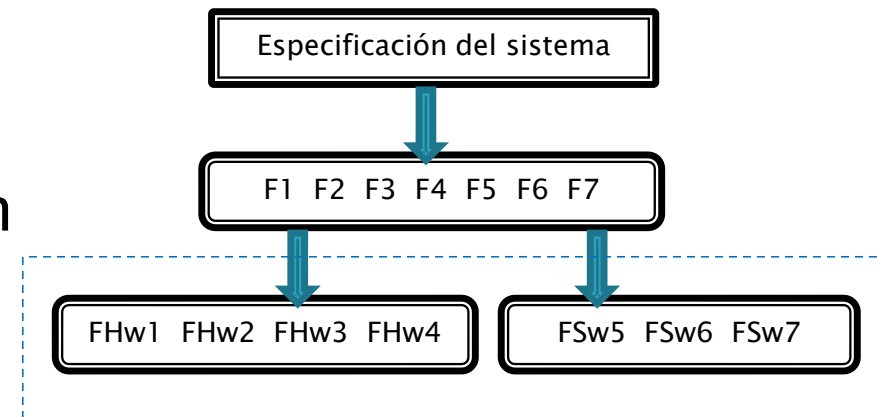
- Parte de la especificación funcional
- Termina en la Plataforma HW-SW (RTL/C)

# Tareas de exploración del espacio

- Diseño funcional
  - Qué?
- Diseño de la arquitectura
  - Cómo?
- Mapeo de funciones
  - SW en HW genérico, HW específico.
- Refinamiento elementos
  - Parámetros de configuración
- Selección de componentes

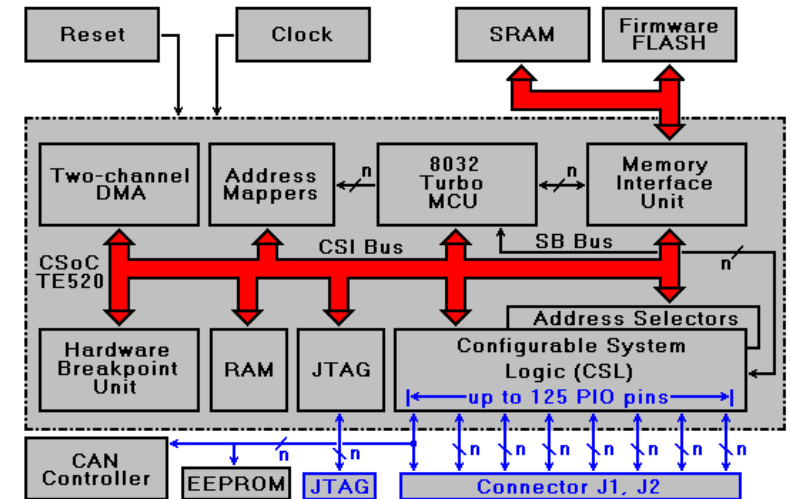
# Tareas de exploración del espacio

- Diseño funcional
  - Análisis de la especificación
  - Partición HW-SW
- Diseño de la arquitectura
  - Componentes Software



# Tareas de exploración del espacio

- Componentes Hardware
  - Procesadores
  - Unidades funcionales
  - Memorias
  - Componentes de E/S
  - Estructura de comunicación



- Mapeo funcional

Software



Hardware



Descripción a nivel RTL de un flip-flop.

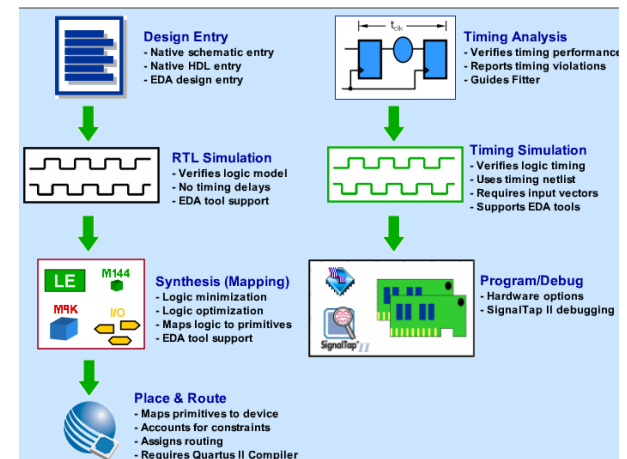


# Tareas de exploración del espacio

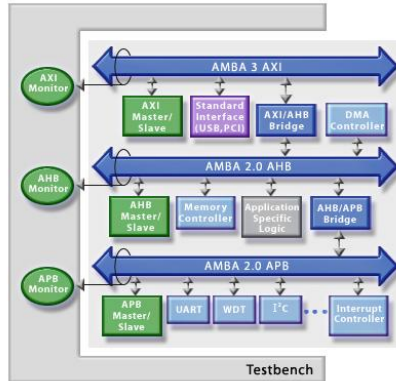
- Refinamiento de los elementos
  - Ajuste de parámetros
- Decisiones guiadas por:
  - Análisis de desempeño
  - Consumo de energía
  - Costo/Otras
  - Metodologías/Herramientas CAD

# Tareas de exploración del espacio

- Selección de componentes
  - Adquiridos a terceros
  - Desarrollos propios
- Implementación
  - Flujo RTL
  - Integración
  - Prototipos



# Ejemplo de exploración: Estructura de comunicación



Política del árbitro

Configuración del  
árbitro

# Puentes  
Clases

Interfaces de protocolo:  
Burst, Transferencias  
divididas

Ancho del Bus

Clase de Bus  
(Fases del  
Pipeline)

HBUS Tamaño  
■ # buses

Mapeo de IP  
IP/bus/clase

# Bus

Mapeo de IP

Ancho del link

Topología

Tamaño

Técnica de Conmutación

Clase:  
Homo/hetero

Estrategia de  
ruteo

Política del árbitro

Configuración  
del árbitro

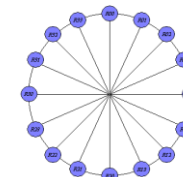
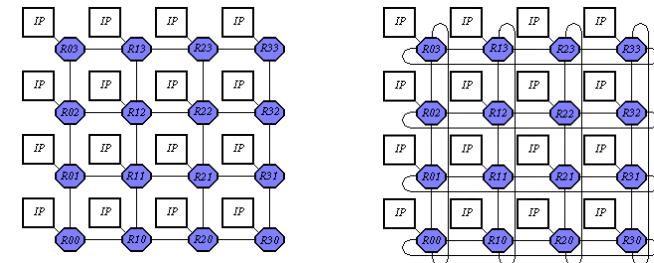
Puertos por router

Buffers por  
router

Tamaño de  
los buffer

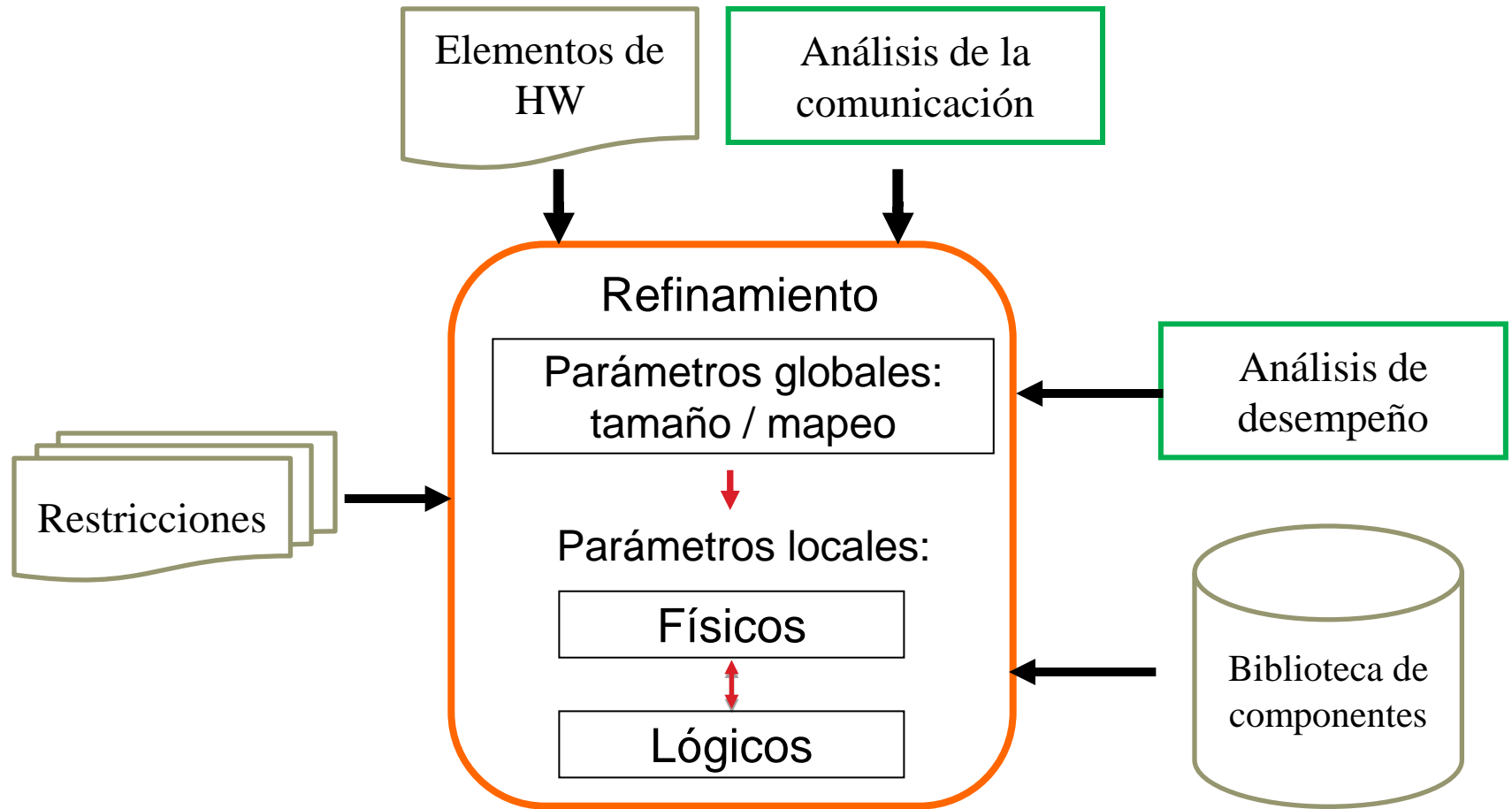
Control de  
flujo

# NoC



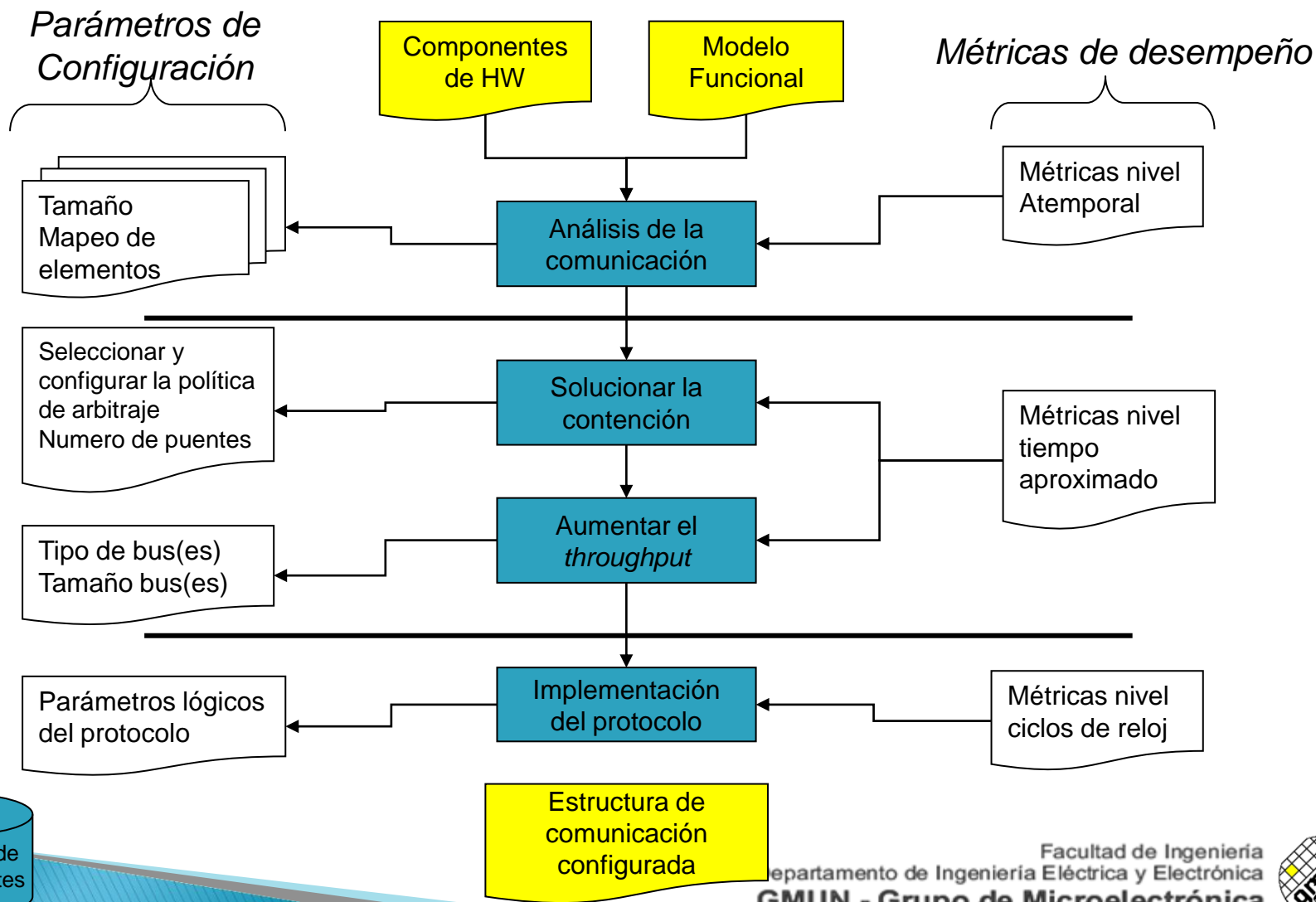


# Ejemplo de exploración: Estructura de comunicación





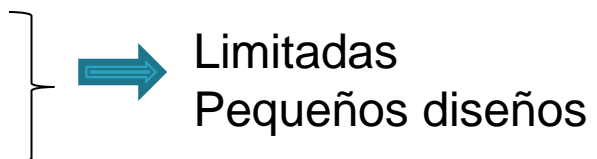
# Ejemplo de exploración: Metodología MaLOC



# Contenido

- Introducción
- Diseño de *SoC*
  - Diseño a nivel de sistema
  - Exploración del espacio de diseño
  - **Verificación funcional y análisis de desempeño**
  - Modelaje
- Conclusiones
- Preguntas

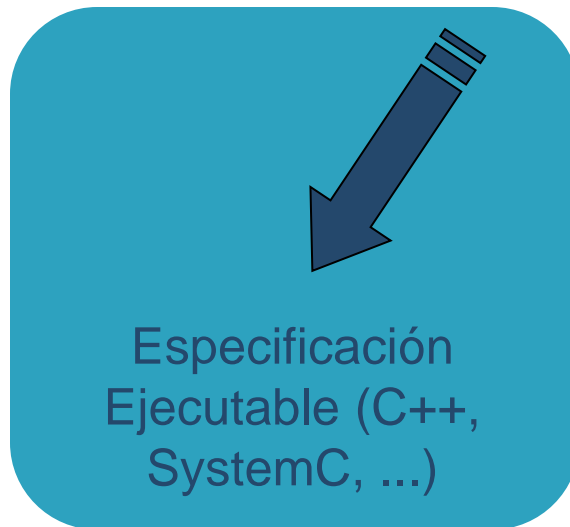
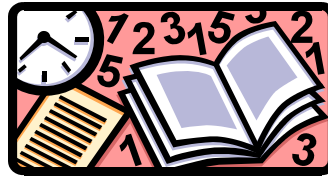
# Verificación funcional

- Los modelos usados están:
  - Correctos
    - Describen la funcionalidad?
  - Completos
    - Describen todas las características?
- Técnicas de verificación
  - Formal
  - Híbrida
  - Dinámica

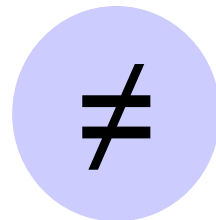
Limitadas  
Pequeños diseños

# Verificación funcional dinámica

- Gran importancia en la industria
- Utilizado en grandes diseños



Equipo de verificación



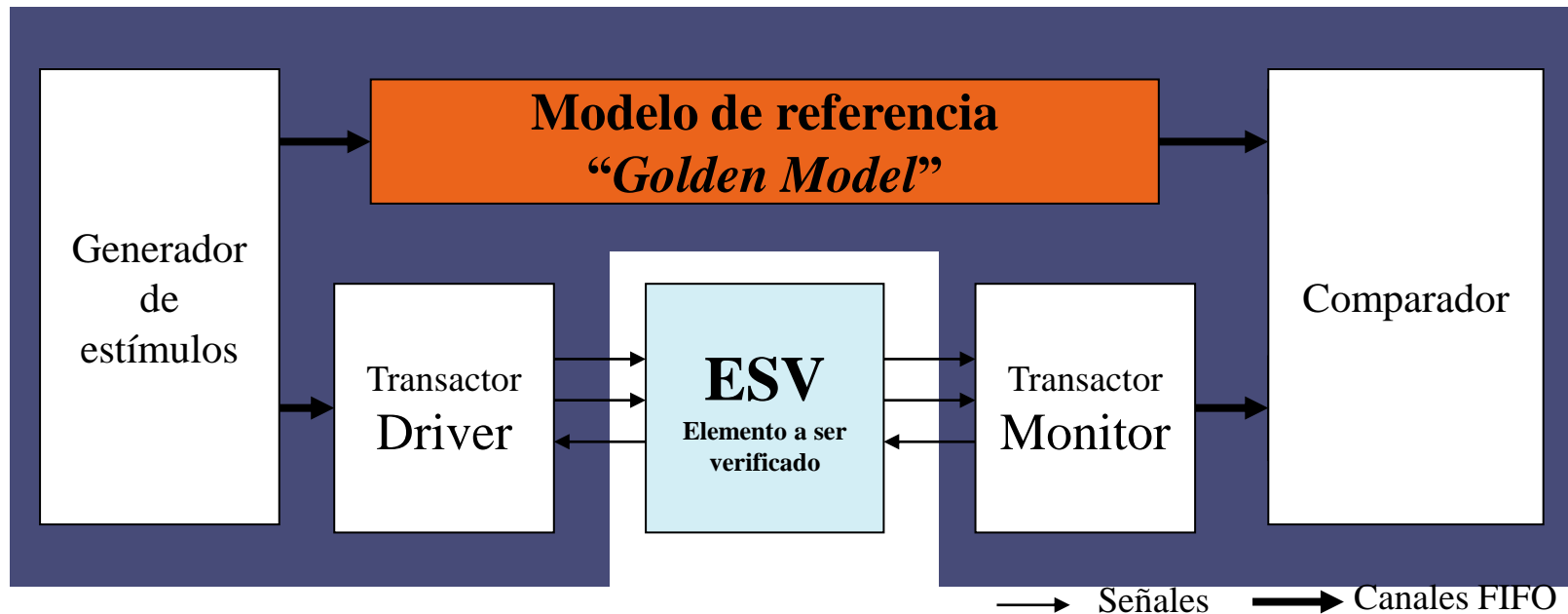
Errores



Equipo de implementación

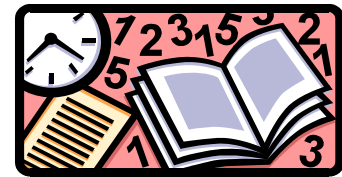
# Verificación funcional dinámica

- Estímulos generados
  - Dirigidos, aleatorios, reales, casos extremos
- Cobertura
  - Objetivos de la verificación a ser observados



# Análisis de desempeño

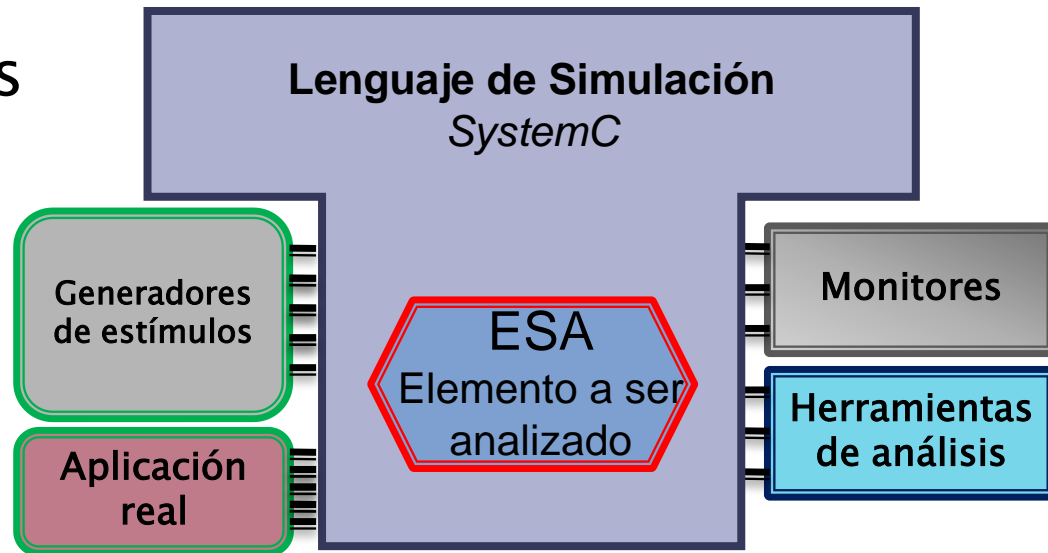
- Las especificaciones son cumplidas?
- Medir el comportamiento
  - Modelos, prototipos
- Cuantificar y evaluar
  - Métricas de desempeño
  - Estrategias de análisis





# Análisis de desempeño

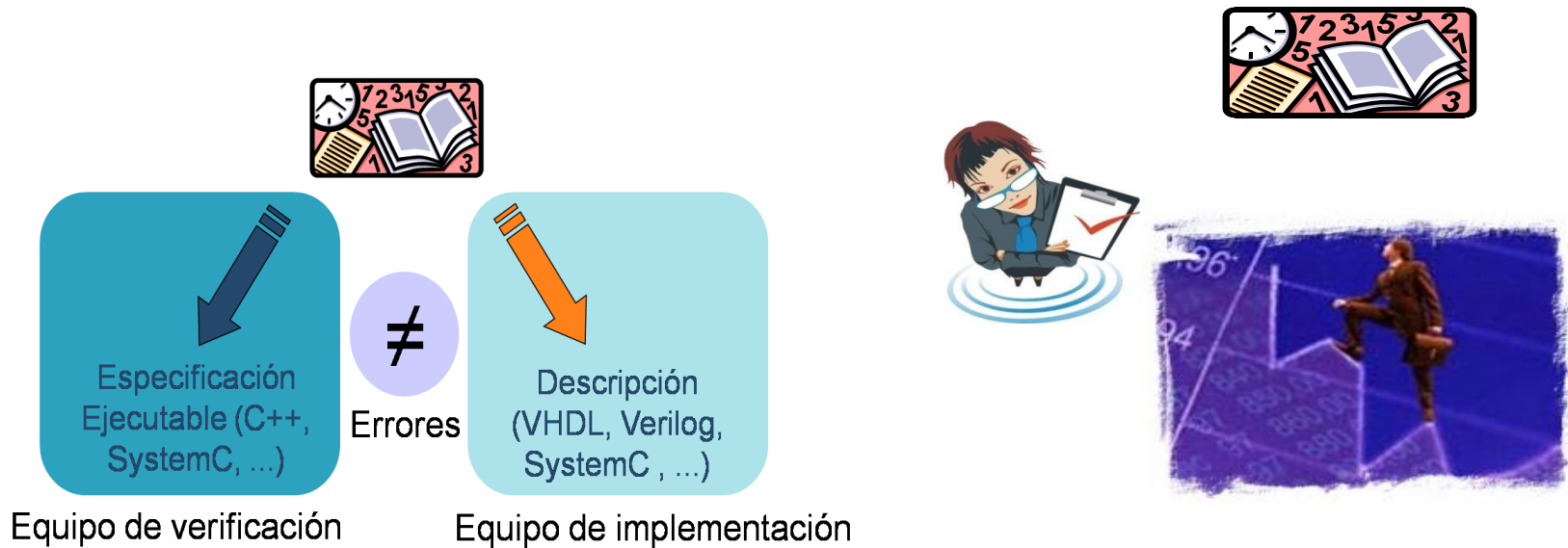
- Generadores de tráfico
- Monitores
- Herramientas de análisis
- Motor de simulación



# Contenido

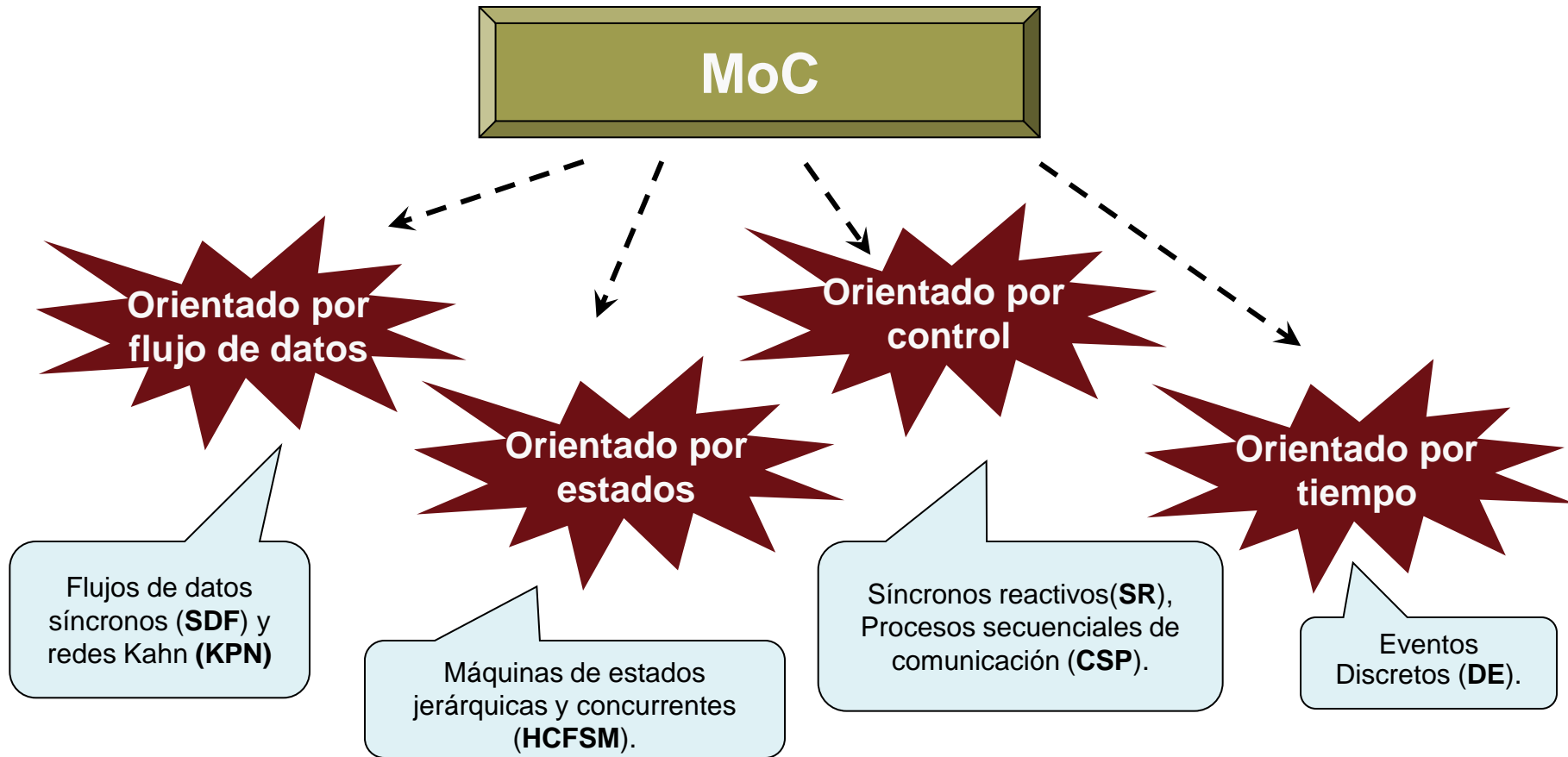
- Introducción
- Diseño de *SoC*
  - Diseño a nivel de sistema
  - Exploración del espacio de diseño
  - Verificación funcional y análisis de desempeño
- **Modelamiento**
- Conclusiones
- Preguntas

# Modelamiento de sistemas

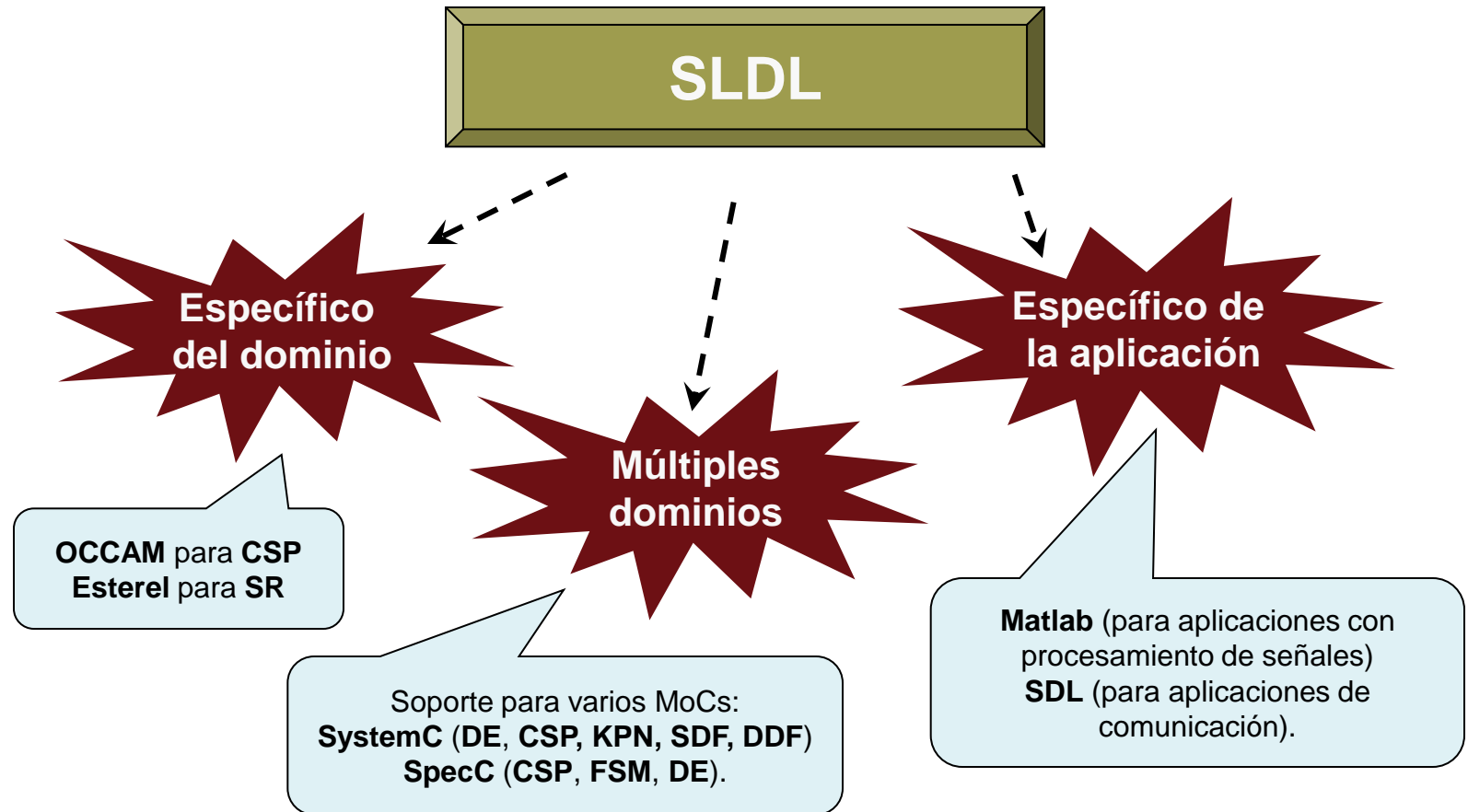


- Conceptos claves
  - Modelo de computación (*MoC*)
  - Lenguajes de diseño a nivel de sistema (*SLDL*)

# Dominios de modelos de computación (MoC)

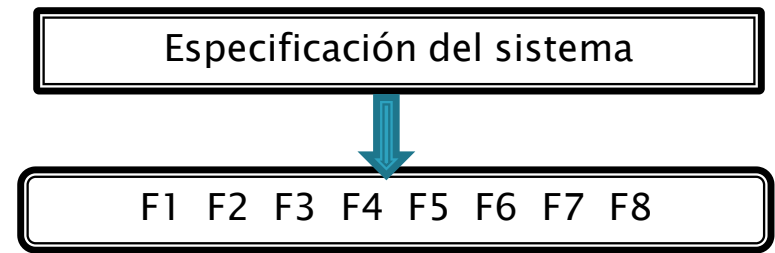


# Lenguajes de diseño a nivel de sistema (*SLDL*)



# Modelaje para análisis funcional

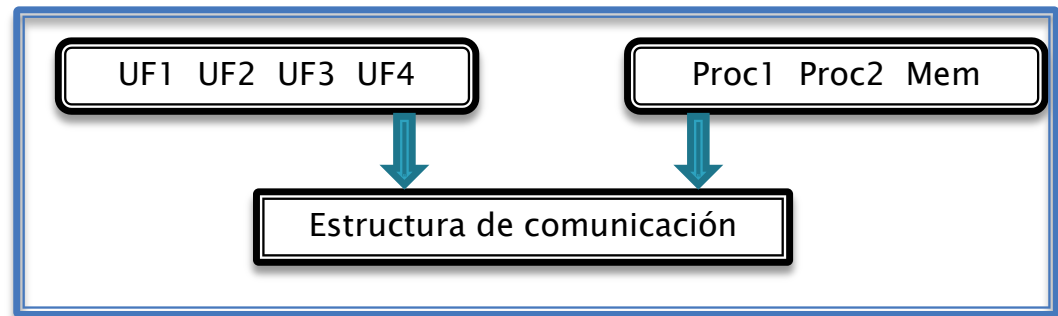
- Basado en MoC
  - Ambientes multi-MoC
  - Ptomely (java)
  - Metrópolis (MML)
- Basado en SLDL
  - Sintaxis y semántica de un lenguaje SLDL
  - Soporte a uno o varios MoC





# Modelaje para análisis arquitectural

- Soportado por SLDL
- Diferente granularidad funcional y temporal
- Modelos basados en transacciones
  - Diferentes niveles de abstracción
    - Atemporal
    - Tiempo aproximado
    - Ciclos de reloj



# Contenido

- Introducción
- Diseño de *SoC*
  - Diseño a nivel de sistema
  - Exploración del espacio de diseño
  - Verificación funcional y análisis de desempeño
  - Modelamiento
- **Conclusiones**
- Preguntas

# Conclusiones

- Sistemas completos HW–SW en un único CI (*SoC*)
- RTL no es eficiente para análisis del diseño
- RTL no desaparece
  - Desplazado
- Nuevos paradigmas de diseño
  - Anticipar decisiones
- Diseño al nivel de sistema
  - Nuevos niveles de abstracción
  - Nuevos lenguajes de diseño
  - Segmentación (*divide and conquer*)
- NO existen soluciones únicas

# Conclusiones

- Futuro del diseño a nivel de sistemas
  - *Menthor, Cadence, Synopsys, Coware, Forte.*
  - Estándar IEEE (SystemC, TLM)
  - Consorcio OSCI – SystemC
    - Patrocinado por diversas empresas
  - Herramientas académicas, código abierto
    - ArchC, GreenSoCs
- Oportunidad para investigaciones
  - Falta de herramientas/metodologías

# Contenido

- Introducción
- Diseño de SoC
  - Diseño a nivel de sistema
  - Exploración del espacio de diseño
  - Verificación funcional y análisis de desempeño
  - Modelamiento
- Conclusiones
- **Preguntas**

# PREGUNTAS?