Tema 1:

# Diseño automático de sistemas digitales

## Diseño automático de sistemas

**José Manuel Mendías Cuadros**
*Dpto. Arquitectura de Computadores y Automática*
*Universidad Complutense de Madrid*

# Contenidos

- ✓ Introducción.
- ✓ Herramientas EDA.
- ✓ Niveles y dominios de descripción.
- ✓ Síntesis y validación.
- ✓ Herramientas de síntesis.
- ✓ Herramientas de validación.
- ✓ Estilos de diseño microelectrónico.
- ✓ Ciclo de producción VLSI.

# Introducción

- La microelectrónica estudia la integración de un gran número de dispositivos electrónicos sobre un único sustrato.
  - es la base del desarrollo de los sistemas informáticos.

- El aumento en la escala de integración ha sido debido a la continua reducción de tamaño de los dispositivos

| Año | Hito | λ | dispositivos por chip | contenido tipo de un chip |
|---|---|---|---|---|
| 1947 | invención del transistor | | | |
| 50's | diseño con componentes discretos | | 1 | transistor |
| 60's | SSI /MSI | 10 µm | $10/10^2$ | puertas/módulos estándar |
| 70's | LSI | 5 µm | $10^3$ | micros 8 bits |
| 80's | VLSI | 1 µm | $10^5$ | micros 16/32 bits |
| 90's | | 350 nm | $10^6$ | micros 64 bits |
| 00's | ULSI | 65 nm | $10^8$ | multicores |
| 10's | | 14 nm | $10^9$ | |
| 20's | límite de la ley de Moore | 5 nm | ??? | |

# Introducción


- Una mayor integración implica:
  - mayor número de dispositivos/chip.
  - menor número de chips/plataforma.
  - mayor velocidad de cálculo.
  - menor consumo/dispositivo.
- Pero también implica:
  - mayor coste de manufactura que si se compensa con un mayor volumen de ventas, se traduce en un menor coste/chip.
  - mayor esfuerzo en diseño.
  - mayor esfuerzo en corrección (menor posibilidad de reparación).

- Sin embargo, la capacidad de integración crece a un ritmo mayor que la productividad del diseñador.
  - Por ello, son necesarias herramientas EDA (de diseño automático)

# Herramientas EDA

## objetivos

- Gestionar la complejidad
  - reduciendo el esfuerzo humano.
- Incrementar la productividad
  - reduciendo del ciclo de diseño y por tanto el tiempo de lanzamiento de productos.
- Aumentar la calidad del diseño en términos de rendimiento (velocidad), coste (área), consumo y fiabilidad
  - evaluando un mayor número de diseños.
  - reduciendo tasa de errores mediante la automatización.

- Sin embargo los problemas de decisión / optimización que resuelven las herramientas EDA son intratables
  - Se usan algoritmos aproximados (heurísticos) que obtienen soluciones válidas, aunque no óptimas, en un tiempo razonable.
  - Muchos de ellos suelen ser algoritmos estocásticos, por lo que bajo unas mismas condiciones iniciales pueden obtener soluciones distintas.

# Herramientas EDA

## tipologías


- Herramientas de captura y reutilización:
  - Metodología ascendente (botton-up).
  - Las herramientas *asisten* al diseñador
    - herramientas de edición manual de diseños.
    - herramientas de chequeo de reglas.
    - herramientas de gestión de bibliotecas de componentes
- Herramientas de especificación y síntesis:
  - Metodología descendente (top-down).
  - Las herramientas *reemplazan* al diseñador
    - herramientas de especificación de requisitos (funcionales y no funcionales).
    - herramientas de síntesis y optimización (exploración de soluciones).
    - herramientas de estimación de características.

- Adicionalmente en todos los entornos de existen:
  - herramientas de simulación.
  - herramientas de verificación (de equivalencia, de propiedades).
  - herramientas de análisis de calidad (estático, dinámico).
  - herramientas de conversión de formatos.
  - herramientas de testeo.

# Herramientas EDA

## dominios de descripción

- Todo sistema digital puede ser descrito según tres ópticas distintas, denominadas dominios de descripción:
  - Dominio conductual: un sistema es descrito por la función que realiza.
  - Domino estructural: un sistema es descrito por la interconexión de los módulos que lo forman (netlist o esquemático).
  - Domino físico: un sistema es descrito por la ubicación espacial y propiedades de los elementos reales que lo constituyen.

# Herramientas EDA

## niveles de abstracción (i)


- El nivel de detalle con que se realice cualquiera de las anteriores descripciones, se denomina nivel de abstracción

- Nivel circuital:
  - el tiempo es continuo
  - las variables son magnitudes físicas continuas
  - no existen por separado las nociones alimentación y computación
  - la calidad se expresa en términos de tiempos de subida/bajada, pendientes de transición, área real, etc.
- Nivel lógico:
  - el tiempo es continuo
  - las variables son discretas y toman valores booleanos
  - la alimentación se abstrae permaneciendo la noción de computación
  - la calidad se expresa en términos de tiempo de propagación, tiempo de incertidumbre, área equivalente, etc

# Herramientas EDA

## niveles de abstracción (ii)

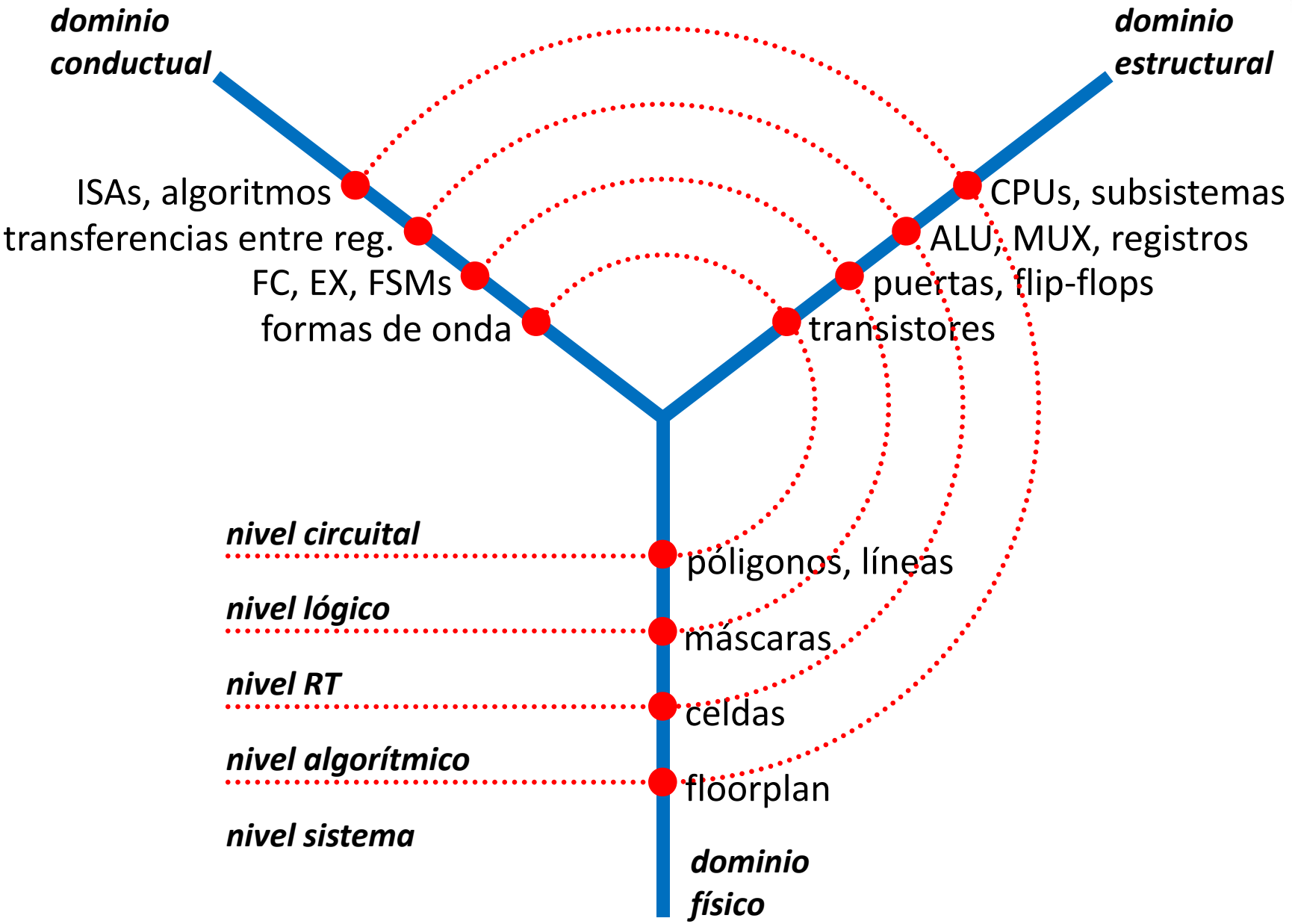

- Nivel transferencia entre registros:
  - el tiempo es discreto y se mide en ciclos de reloj
  - las variables se agrupan en palabras que representan datos
  - la noción de computación se divide en control y procesamiento de datos
  - la calidad se expresa términos de tiempo de ciclo, margen, num. puertas
- Nivel algorítmico:
  - el tiempo desaparece apareciendo la noción de dependencia
  - las variables se agrupan en estructuras abstractas
  - el control y los datos se estructuran
  - la calidad se expresa en términos de latencia, cadencia, núm de módulos
- Nivel sistema:
  - desparecen los detalles de los cálculos concretos e interesan las relaciones abstractas entre subsistemas,
  - aparecen nociones de sincronización y protocolo.
  - la calidad se expresa en términos de ancho de banda, MFLOPS, etc.

J.M. Mendías
2015



# Niveles y dominios

## representación gráfica



J.M. Mendías
2015



# Niveles y dominios

## transiciones

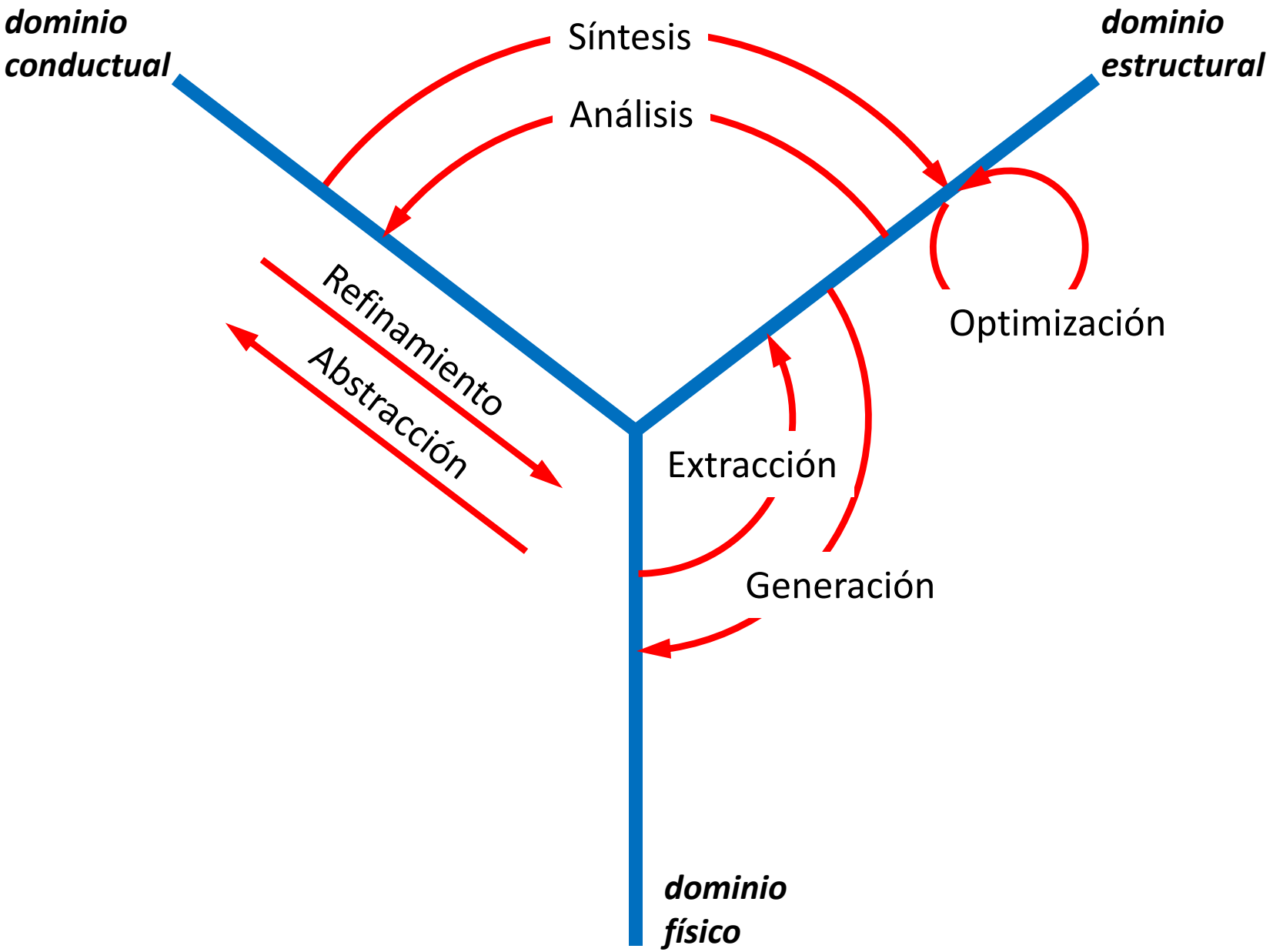
- Síntesis: obtención de una netlist a partir una descripción de su conducta (transición del dominio conductual al estructural).
- Análisis: obtención de la conducta que manifiesta una netlist (transición del dominio estructural al conductual).
- Optimización: modificación de una descripción, sin cambiar de dominio ni de nivel, de manera que tenga una mejor calidad.
- Generación: obtención de una estructura física a partir de una netlist (transición del dominio estructural al físico).
- Extracción: obtención de una netlist a partir de una estructura física (transición del dominio físico al estructural).
- Refinamiento: disminución del nivel de abstracción de una descripción sin cambiar de dominio.
- Abstracción: aumento del nivel de abstracción de una descripción sin cambiar de dominio.

J.M. Mendías
2015



# Niveles y dominios

## transiciones



# Síntesis y validación


- El objeto de un proceso de diseño es obtener una descripción física a nivel circuital a partir de una descripción conductual lo más abstracta posible.
  - La descripción de partida se denomina especificación y la obtenida tras el diseño, implementación.
  - En el caso del diseño automático se trata que este proceso se realice con la mínima intervención humana.
  - Conseguir este objetivo en un sólo paso es imposible: se realiza encauzando los resultados a través de una serie de procesos especializados.

- Consustancial a síntesis está la validación, que chequea si un sistema se ajusta a unos ciertos requisitos, que pueden ser:
  - Funcionales: el sistema se comporta adecuadamente.
    - Simulación, testeo, chequeo de reglas, verificación formal.
  - No funcionales: el sistema tiene cierto nivel de calidad
    - Simulación, testeo, análisis.

# Herramientas de síntesis

## síntesis de alto nivel


- Entrada: una descripción conductual de nivel algorítmico de una computación.
  - Ligaduras típicas: latencia, tiempo de ciclo, número y tipo de recursos, cadencia de admisión de datos, etc.
- Salida: una ruta de datos (netlist de nivel RT) y una descripción conductual de nivel RT del controlador (secuencia de transferencias entre registros).
- Tareas:
  - Planificación de operaciones (scheduling): fija el ciclo en que se ejecuta cada una de las operaciones que forman la computación.
  - Selección de recursos (resource binding): determina el número y tipo de recursos funcionales (FUs, regs, mux/buses) requeridos para realizar la computación
  - Asignación de recursos (resource allocation): fija el recurso en el que se ejecuta, almacena o transmite cada una de las operaciones y objetos de datos que la forman.
  - Otras tareas: transformación de operaciones, transformación de bucles, segmentación

J.M. Mendías 2015

# Herramientas de síntesis

## síntesis lógica / RT

- Entrada: una descripción formada por funciones combinacionales, secuenciales simples y máquinas de estados finitos.
  - Ligaduras típicas: retardo/longitud del camino crítico, frecuencia de reloj, área equivalente, consumo de potencia, etc.
- Salida: una netlist de puertas y biestables.
- Tareas:
  - Minimización combinacional: construye una netlist reducida (a dos niveles o multinivel) capaz de implementar una cierta función lógica.
  - Reestructuración combinacional: modifica la ubicación relativa y número de las puertas de una netlist combinacional para optimizar su rendimiento global.
  - Minimización de estados: reduce el número de estados de una FSM
  - Codificación de estados: elige una codificación binaria de estados que minimice las funciones de estado y salida de una FSM.
  - Reestructuración secuencial: modifica la ubicación relativa y número de los biestables de una netlist secuencial para optimizar su rendimiento global
  - Proyección tecnológica: transforma una netlist general de puertas y biestables en otra equivalente cuyos componentes están limitados en cantidad o tipo.
  - Generación de test: crea una colección de estímulos capaces de detectar un cierto porcentaje de errores posibles provocados por fallos del proceso de fabricación.

# Herramientas de síntesis

## síntesis física

- Entrada: una netlist de celdas.
  - Ligaduras típicas: área, retardos, caracteristicas eléctricas, etc.
- Salida: desde una colección de máscaras para la fabricación del circuito, hasta la configuración a ser descargada sobre una FPGA.
- Tareas:
  - Proyección tecnológica: transforma una netlist de puertas y biestables en otra equivalente del celdas pertenecientes a una biblioteca tecnológica.
  - Particionamiento: división de una netlist de celdas en varias particiones que se procesarán por separado.
  - Floorplaning: selección de la ubicación aproximada de cada una de las particiones o macroceldas que componen un circuito y de las regiones de interconexionado
  - Ubicación de celdas (placement): selección de la ubicación dentro de una cierta región de cada una de las celdas que componen una partición
  - Rutado (routing) global: asignación de interconexiones a regiones de interconexionado
  - Rutado detallado: trazado físico de las interconexiones.
  - Compactado: eliminación de los espacios libres de un layout

# Herramientas de validación

## validación por simulación (i)


- Simulador: es una herramienta que permite reproducir el comportamiento (típicamente temporal) de un circuito a un cierto nivel de abstracción, bajo varios escenarios de operación
  - Usa un modelo ejecutable del circuito, una colección de estímulos de entrada y evalúa las formas de onda que genera el circuito tras la simulación.
- Problemas:
  - Incompletitud: cualquier circuito requiere para su validación de una colección de estímulos que crece exponencialmente con su complejidad.
  - Complejidad: cuanto más precisa sea una simulación más tiempo demanda.
- Soluciones:
  - Métricas de cobertura: miden la calidad de un subconjunto de estímulos.
    - Cobertura de código (para modelos textuales)
    - Cobertura de datos.
  - Aceleración hardware: uso de computadores específicos para ejecutar modelo.
  - Emulación hardware: proyección de un modelo sobre hardware reconfigurable
  - Emulación de vectores: se usa un generador de patrones y un analizador lógico
  - Emulación sobre el circuito: el prototipo se integra en el sistema destino

# Herramientas de validación

## validación por simulación (ii)

- Existen distintos tipos de simuladores según la precisión con que se trate el tiempo y los dominios de datos.

- Simuladores eléctricos (nivel circuital)
  - Hacen un tratamiento continuo del tiempo y de todas las magnitudes físicas del circuito.
  - Típicamente toma como entrada una netlist de elementos microelectrónicos (transistores, resistencias, capacidades)
  - Está basada en la resolución de matrices de ecuaciones que relacionan los voltajes e intensidades que circulan por cada uno de los elementos del circuito.
  - Pueden usar modelos modelos de diferente nivel de complejidad que llevan aparejados diferentes márgenes de error (nunca es cero).
  - Las simulaciones son lentas y por tanto sólo son aplicables a pequeñas partes del circuito y a pequeñas ventanas temporales.

# Herramientas de validación

## validación por simulación (iii)

- Simuladores lógicos:
  - Hacen un tratamiento continuo del tiempo, pero abstraen el comportamiento eléctrico del circuito discretizando las magnitudes físicas.
  - Toma como entrada una netlist de puertas y biestables.
  - Utilizan elaborados modelos de puertas y retardos RC para dar una visión detallada del timing de un circuito, mostrando tanto los estados estables como transitorios de los elementos del sistema.
  - Pueden retroanotarse con información extraída de un layout

- Simuladores RT (o cycle-based):
  - Hacen un tratamiento discreto del tiempo y de las magnitudes físicas.
  - Toma como entrada una netlist de puertas y biestables.
  - Utilizan modelos funcionales de puertas, abstrayendo estados transitorios del sistema
  - Sólo pueden usarse para validación funcional

J.M. Mendías
2015



# Herramientas de validación

## testeo


- Tras fabricar un circuito, y antes de lanzarlo, debe validarse que funciona adecuadamente (asumiendo que el diseño era correcto).
- Testeo: proceso de detección de errores en un circuito resultado de los defectos físicos y de las variaciones del proceso de fabricación.
  - Es similar a la validación por simulación, excepto en que se utiliza un circuito real en lugar de un modelo ejecutable del mismo.
  - Se aplican una colección de estímulos de entrada y evalúa las formas de onda que genera el circuito en funcionamiento.
- La calidad del conjunto de test usado (completitud vs. complejidad) se evalúan usando modelos y métricas de cobertura de fallos.
  - Simulador de fallos: simulador que intencionadamente inserta fallos en el modelo de un circuito para evaluar la cantidad de ellos detectables por un conjunto de tests.
- Existen dos tipos de testeo:
  - Funcional: chequea que la conducta del circuito se ajusta a su diseño.
  - Paramétrico: chequea aspectos no funcionales tales como: márgenes de ruido, frecuencias de reloj, retardos de propagación, etc. bajo diferentes condiciones de funcionamiento

# Herramientas de validación

## chequedores de reglas

- Siempre que exista un modelo de un circuito procesable por máquina, es posible chequear que satisface ciertas reglas constructivas
  - La conformidad de dichas reglas no asegura la corrección, pero una violación de las mismas es síntoma de error.
  - Son especialmente útiles para chequear diseños editados a mano, las herramientas automáticas los llevan implícitamente incorporados (corrección por construcción).

- Ejemplos:
  - DRC (Design Rule Chequers): chequeadores de geometría y topología.
  - Chequeadores de conectividad en esquemáticos.
  - Chequeadores de estilo (para modelos textuales).

# Herramientas de validación

## analizadores

- Evalúan la calidad de un circuito mediante la extracción de medidas analíticas de rendimiento.
  - Estáticos: mediante la observación de su estructura.
  - Dinámicos: mediante el estudio de la respuesta a una colección de estímulos

- Analizadores de timing: chequean que un circuito satisface las restricciones derivadas de una cierta estrategia de temporización (por flanco, por nivel, etc.)

- Analizadores de potencia: chequean que un circuito satisface ciertas restricciones de consumo.

# Herramientas de validación

## verificación formal

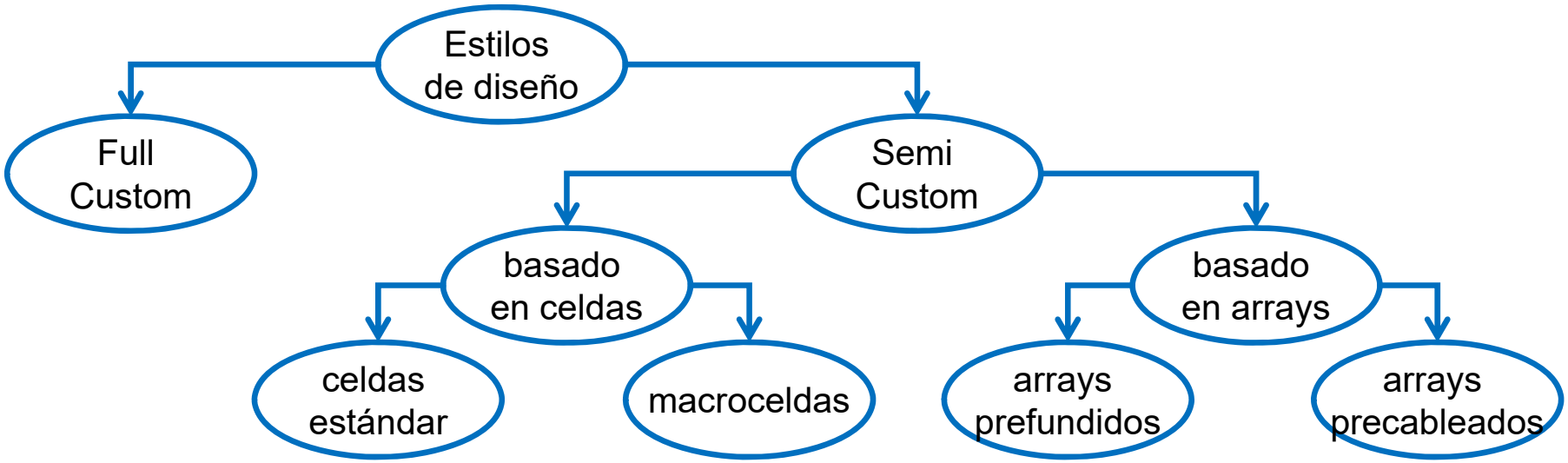
- Son métodos matemáticos de validación funcional que establecen sus resultados de una manera independiente a estímulos.
  - Resuelven el problema de la incompletitud de los métodos basados en simulación.

- Chequeadores de equivalencia: demuestran que dos modelos (de diferentes dominios o nivel de abstracción) son funcionalmente equivalentes.
  - Equivalencia estricta: tienen igual pinout y responderían igual a la misma secuencia de estímulos
  - Consistencia: son equivalentes para un subconjunto del pinout, o son equivalentes para un subconjunto de los posibles estímulos y/o de las respuestas.

- Chequeadores de propiedades: demuestran que un modelo posee una cierta característica funcional.

# Estilos de diseño

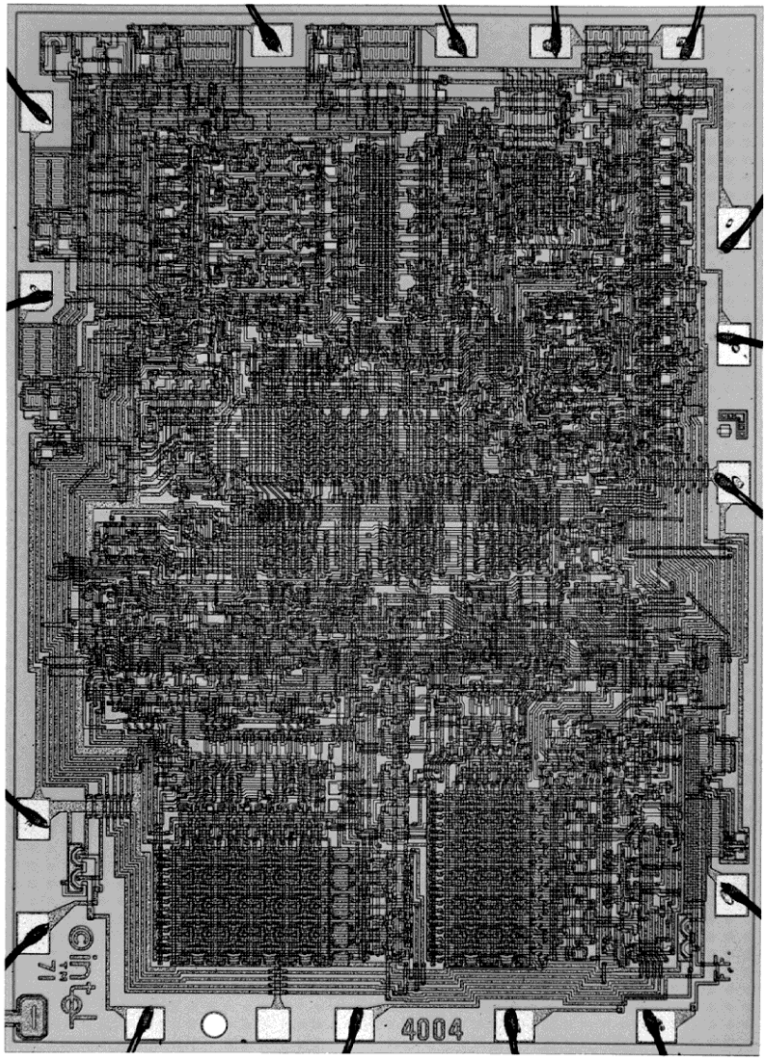
- La viabilidad económica de un diseño microelectrónico depende de:
  - rendimiento del circuito
  - volumen de fabricación
  - núm. de circuitos correctos / núm. de circuitos fabricados (yield)
  - precio y tiempo requerido de salida a mercado
- Existen distintos estilos de diseño físico para hacer frente a diferentes intereses y se clasifican según:
  - El grado de libertad que tiene el diseñador a la hora de decidir la topología física del circuito (punto de vista del diseñado).
  - El número de máscaras que pueden reutilizarse (punto de vista del fabricante)



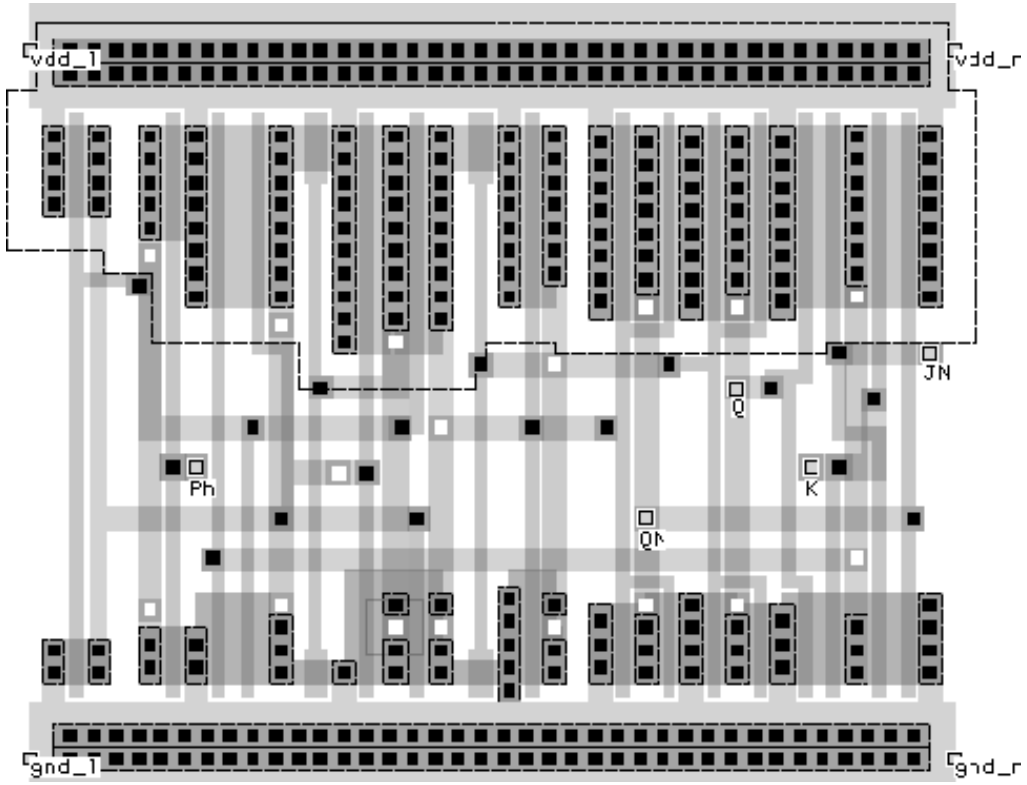
# Diseño full custom


- El diseñador tiene completa libertad durante el diseño físico:
  - no se utilizan elementos prediseñados ni prefabricados.
  - no existen restricciones en el diseño de un bloque funcional.
  - no existen restricciones en la ubicación de los bloques funcionales.
  - no existen restricciones en el trazado de las interconexiones.
  - puede optimizarse cualquier aspecto de un circuito.
  - se suele realizar jerárquicamente.
  - fue muy popular en los primeros años, su uso disminuye día a día.
- Ventajas:
  - flexibilidad.
  - se pueden obtener circuitos de alta calidad.
- Desventajas:
  - no es automatizable.
  - requiere un enorme esfuerzo y diseñadores con alta especialización.
  - tiempos largos de salida al mercado.
  - no hay seguridad de que el circuito funcione eléctricamente.
  - solamente es rentable cuando los costes se amortizan con un gran volumen de producción, con un tiempo de vida largo, o con un alto grado de reutilización.

# Diseño full custom




*4004 de Intel*




***layout de un flip-flop CMOS***

# Diseño semi custom


- El diseñador tiene ciertas restricciones durante diseño físico:
  - sólo pueden usar una colección de bloques funcionales primitivos prediseñados o prefabricados.
  - existen restricciones en la ubicación de los bloques funcionales.
  - existen restricciones en el trazado de las interconexiones.
  - sólo se pueden optimizar algunos aspectos del circuito.
- Ventajas:
  - es automatizable.
  - el funcionamiento eléctrico del circuito está asegurado.
  - reduce el esfuerzo de diseño y requiere diseñadores menos especializados.
  - reduce el tiempo de salida al mercado.
- Desventajas:
  - se obtienen circuitos de rendimiento medio.

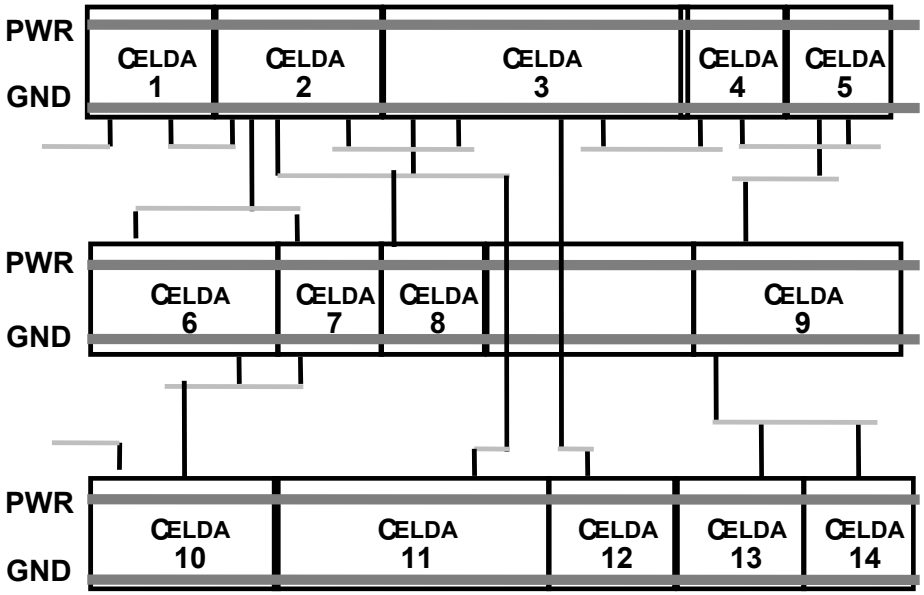
# Diseño semi custom

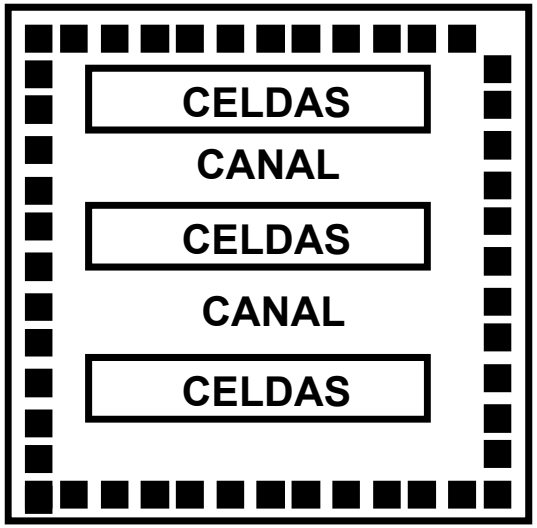
## celdas estándar (i)

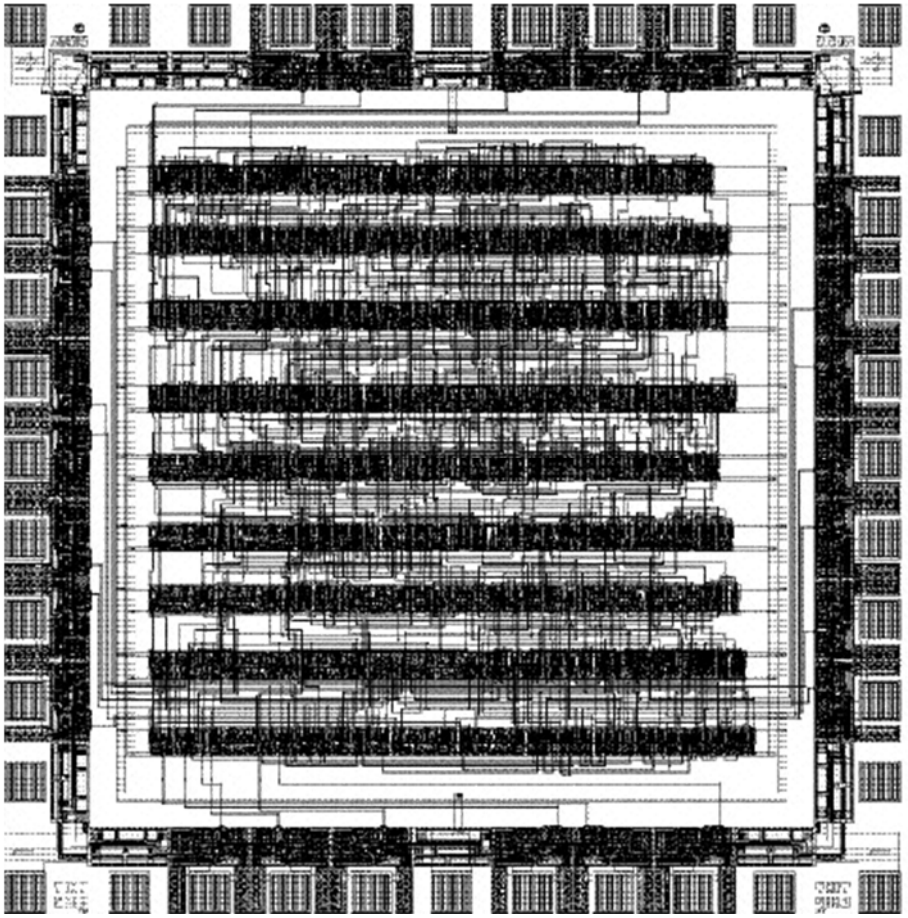

- Todo diseño se realiza mediante la interconexión de un conjunto de bloques funcionales prediseñados denominados celdas.
  - Las celdas (de entre 200 y 400 tipos diferentes) se agrupan en bibliotecas facilitadas por el fabricante y se actualizan cuando cambia la tecnología.
    - Funcionalmente las celdas son simples (puerta lógicas, flip-flops).
    - Han sido diseñadas full custom y caracterizadas por el fabricante.
    - Los diseños de nivel lógico deben proyectarse tecnógicamente sobre celdas de biblioteca
  - Geométricamente una celda es:
    - un rectángulo de altura fija y anchura variable que depende de la complejidad de funcionalidad implementada
    - sus entradas y salidas están ubicadas en los extremos superior e inferior del rectángulo
    - sus tomas de alimentación y tierra están ubicadas de manera que las líneas de distribución de alimentación y tierra se puedan trazar horizontalmente sobre ellas
  - Las celdas deben ubicarse en filas de igual altura, dejando entre las filas un espacio libre de anchura variable denominado canal.
  - Las interconexiones entre celdas de la misma fila o entre celdas de filas adyacentes (conexiones cercanas) se trazan en el canal.
  - Las restantes interconexiones (interconexiones lejanas) se realizan a través de celdas de paso o en otros niveles de metalización.

# Diseño semi custom

## celdas estándar (ii)



*diseño basado en Celdas estándar*

# Diseño semi custom

## mega-celdas y macro-celdas (i)

- Mega-celdas: celdas de mayor complejidad y tamaño con implementaciones físicas reutilizables.
  - Controladores de memoria, conversores AD/DC, temporizadores, etc.
  - Los diseños se realizan a nivel sistema enlazado componentes

- Macro-celdas: celdas de mayor complejidad y tamaño cuyas implementaciones físicas tienen una estructura regular, fácilmente escalable y con un buen rendimiento:
  - Multiplicadores, sumadores, desplazadores, RAM, ROM, PLA ...
  - Los generadores de macro-celdas automatizan su diseño físico
    - a partir de los parámetros característicos del módulo facilitados por el diseñador.

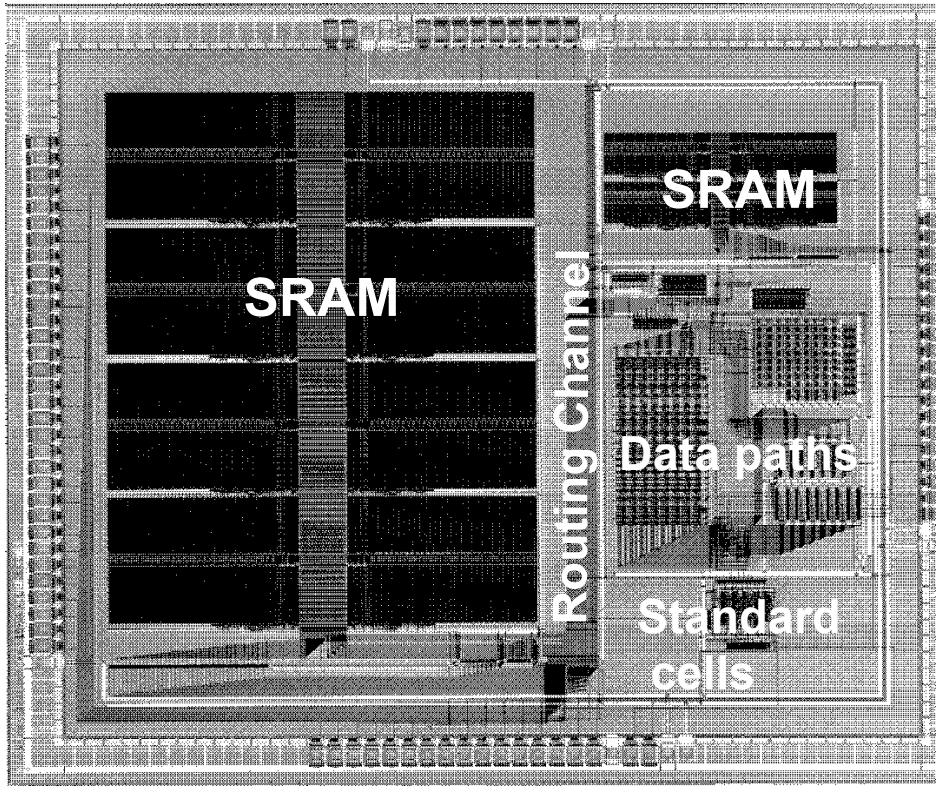
J.M. Mendías
2015



# Diseño semi custom

## mega-celdas y macro-celdas (ii)

- Típicamente, en un diseño semi custom encontraremos un diseño mixto formado por celdas de distinto tipo.



# Diseño semi custom

## arrays predifundidos (i)

- Todo diseño se realiza mediante la interconexión de una colección de celdas idénticas.
  - Funcionalmente las celdas son extremadamente simples (transistor, NAND o NOR).
  - Las celdas ya están prefabricadas y ubicadas regularmente sobre el dado de silicio.
  - Durante la fase de metalización se decide el interconexionado e incluso la función de las celdas.
  - Los diseños de nivel lógico deben proyectarse tecnógicamente sobre celdas de biblioteca.

- El diseño físico se reduce a trazar el interconexionado.
  - Solamente las últimas fases de fabricación (metalización) es dependiente del diseño.
  - Las primeras fases son comunes a cualquier diseño, son reusables y pueden tener un gran volumen de producción.
  - Siempre queda un porcentaje de celdas sin usar de la oblea.

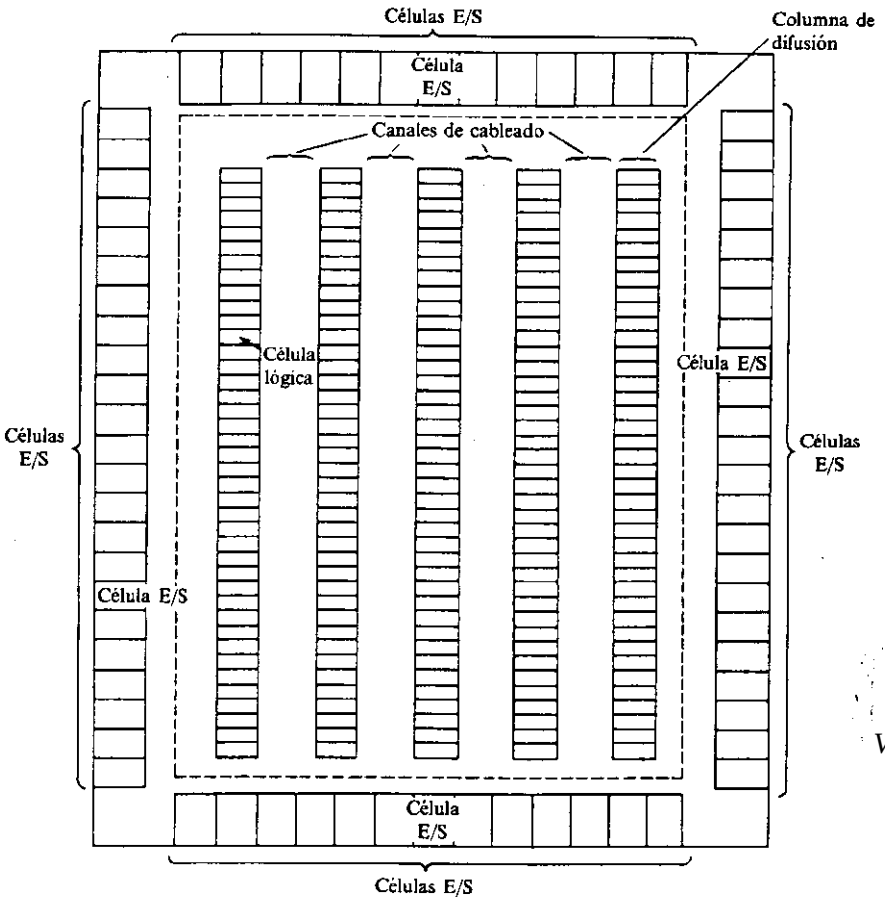
# Diseño semi custom

## arrays predifundidos (ii)

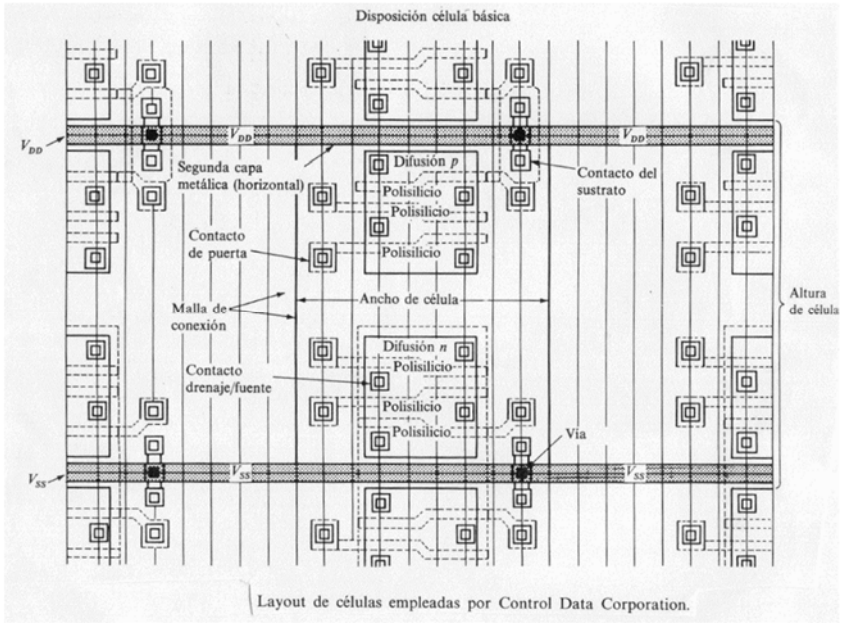
- Hay diferentes alternativas atendiendo a la complejidad de la celda
- Gate array:
  - Cada celda está formada por un pequeño número de transistores cuyo interconexionado local determina su función
  - Las celdas se disponen en filas dejando entre un espacio libre vertical y/o horizontal de tamaño fijo llamado canal.
  - Las interconexiones entre celdas se trazan por el canal.
- Sea of gates:
  - Cada celda es un único transistor.
  - Las celdas se disponen en filas sin dejar espacio libre entre ellas.
  - Las interconexiones se trazan sobre celdas sin utilidad, o usando las propias celdas como elemento de interconexión.
- Structured ASIC:
  - Cada celda es un subcircuito genérico pequeño (NAND, MUX, FA, FF...)
  - Las celdas se interconectan en niveles superiores de metalización.

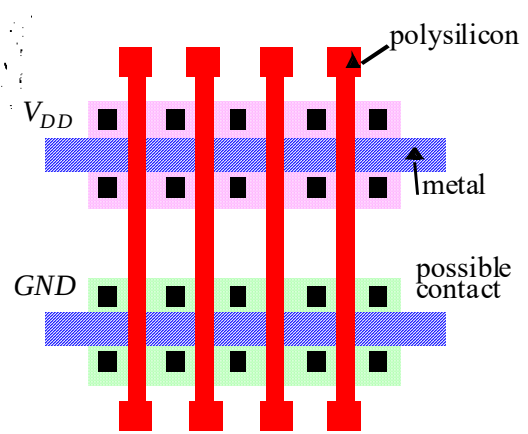
# Diseño semi custom

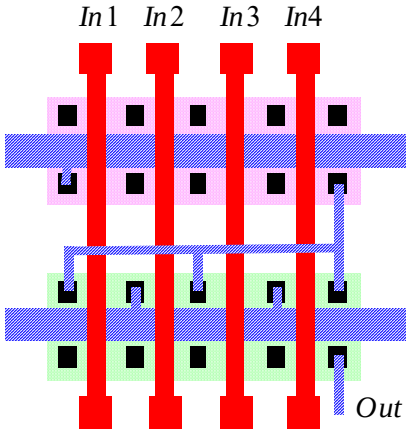
## arrays predifundidos: gate array




***esquema del layout de un Gate array***



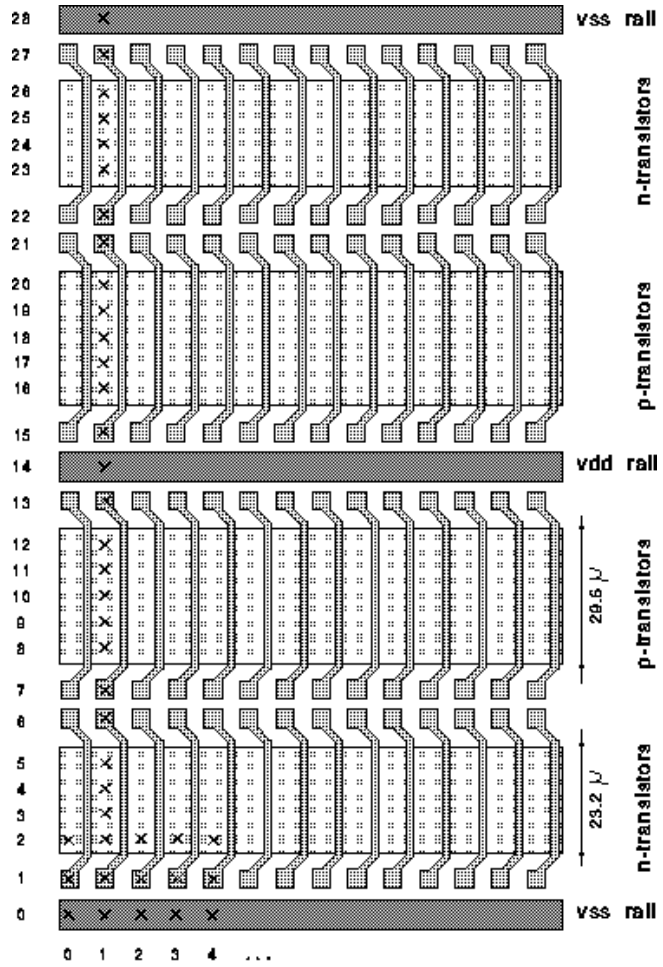



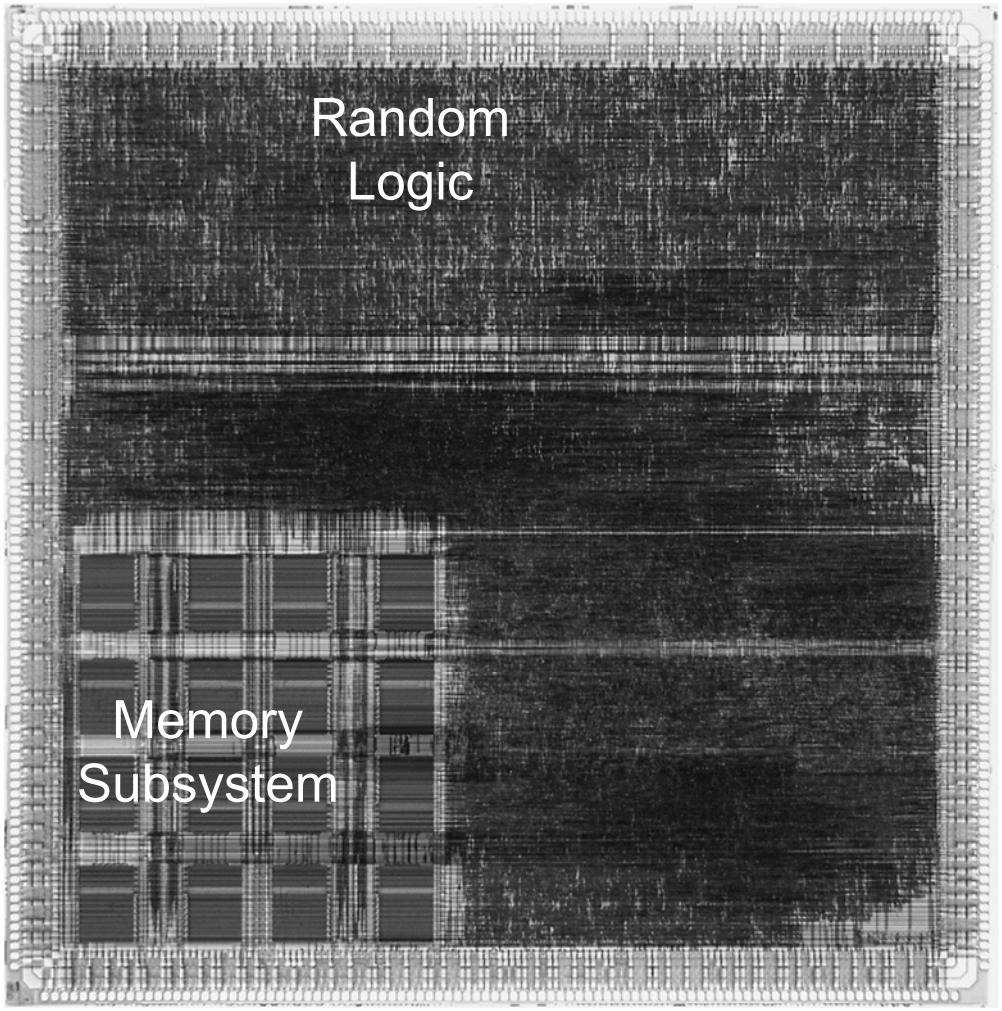

***4-input NOR***

# Diseño semi custom

## arrays predifundidos: sea of gates




*esquema del layout de un Sea of Gates*



# Diseño semi custom

## arrays precableados: PROM, PLA, PAL y CPLD


- Circuitos prefabricados con funcionalidad programable no-volátil
  - Celdas funcionales básicas de funcionalidad fija:
    - PROM, PLA, PAL: arrays de puertas NOT-AND-OR = sistemas combinacionales
    - CPLD: arrays de puertas NOT-AND-OR + arrays de FFD = sistemas secuenciales
  - Interconexiones configurables:
    - PROM: se programan las interconexiones del array de OR (array AND fijo)
    - PLA: se programan las interconexiones del array AND (OR fijo)
    - PAL: se programan las interconexiones de los arrays AND y OR
    - CPLD: se programan las interconexiones de los arrays AND, OR y FFD
- Tecnológicamente la configurabilidad se consigue:
  - Fundiendo fusibles/antifusibles que unen entre sí las interconexiones
- Los diseños no se fabrican, sino que se realizan configurando las interconexiones
  - Toda funcionalidad combinacional debe expresarse finalmente en forma de suma de productos (implementable en 2 niveles de puertas AND+OR)
  - Toda funcionalidad secuencial debe expresarse finalmente en su forma canónica (un único registro de estado).

J.M. Mendías
2015

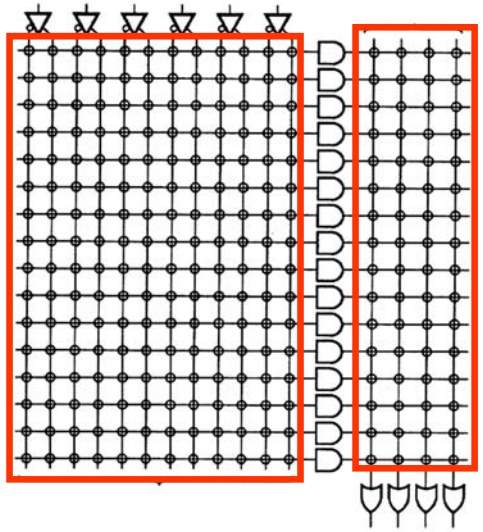




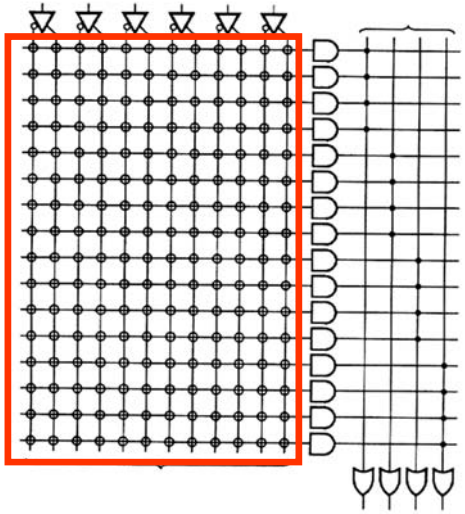
# Diseño semi custom

## arrays precableados

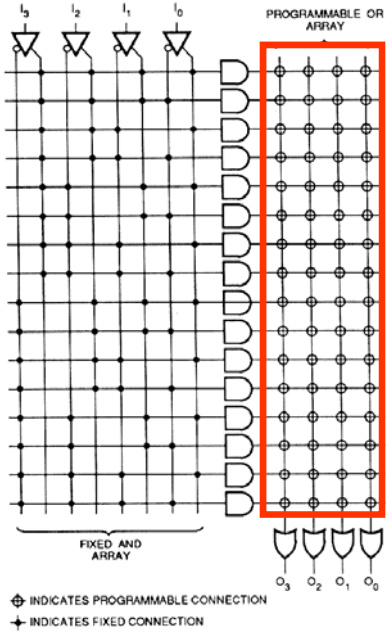
***interconexiones (re)programables***



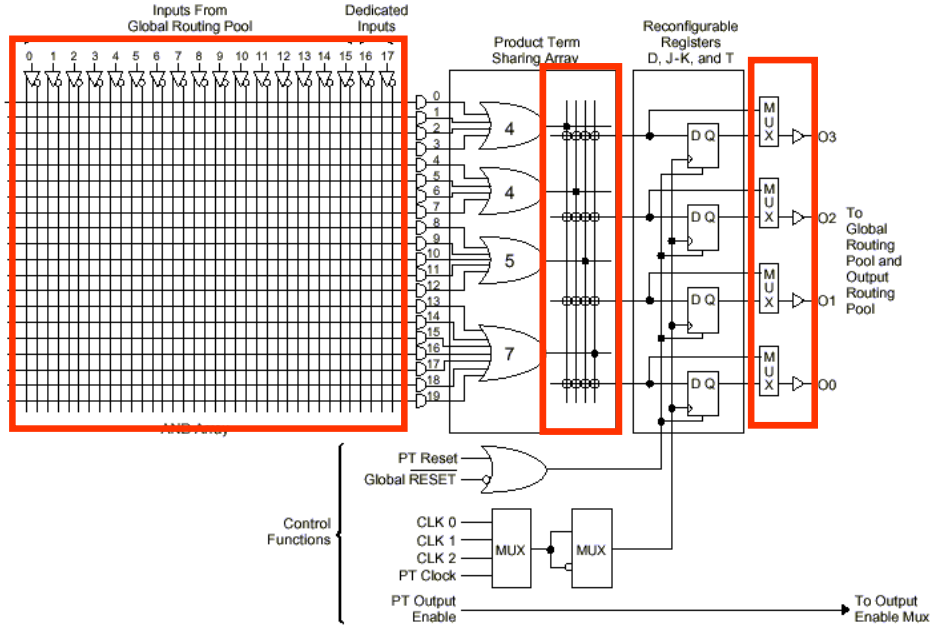
***PLA***



***PAL***




***PROM***




***CPLD***

J.M. Mendías 2015



# Diseño semi custom

## arrays predifundidos: FPGA


- Circuitos prefabricados con funcionalidad programable volátil
  - Celdas de funcionalidad reconfigurable (CLB, slices, etc. ) dispuestas regularmente sobre el silicio siguiendo un patrón matricial
  - Bloques de interconexión reconfigurables dispuestos regularmente alrededor de CLB.
  - Celdas de entrada/salida (IOB) de características reconfigurables dispuestas perimetralmente (estándares IO: SSTL, CTL, LVTTL, LVCMOS, HSTL, PCI, AGP, GTL, ...)
  - Memorias de configuración regularmente distribuidas y conectadas en scan-path
  - Circuitería de control de la configuración (carga, startup y readback).
- Tecnológicamente la reconfigurabilidad se consigue:
  - Funcionalidad reconfigurable: pequeñas memorias de configuración que almacenan tablas de verdad cuyo contenido es seleccionado por un multiplexor
  - Interconexión reconfigurable: segmentos metálicos unidos a través de transistores de paso controlados por una memoria de configuración
- Los diseños no se fabrican, sino que se realizan configurando las memorias de los CLB, IOB y bloques de interconexión
  - Previamente toda funcionalidad debe proyectarse sobre la arquitectura objetivo.

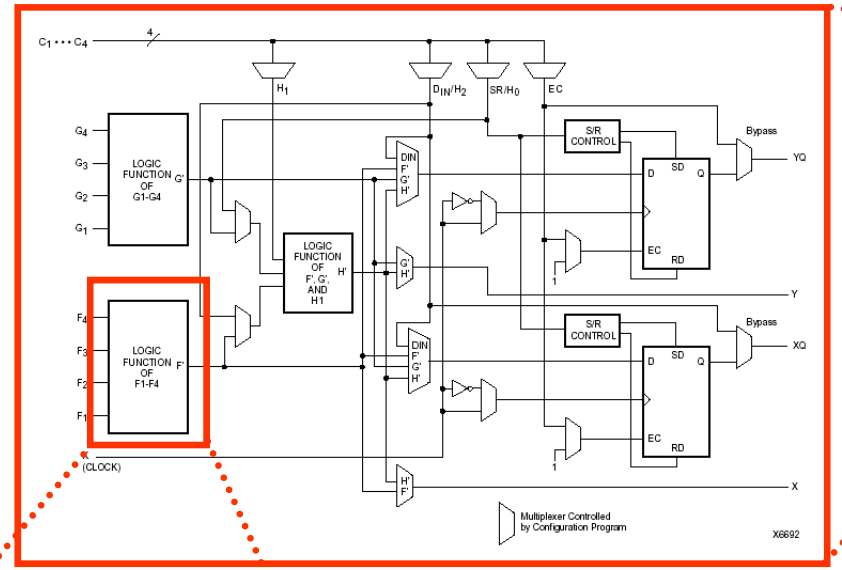
J.M. Mendías
2015

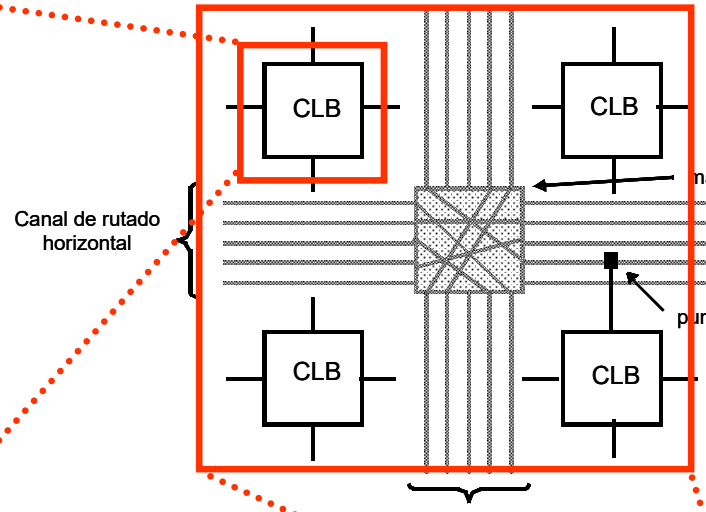


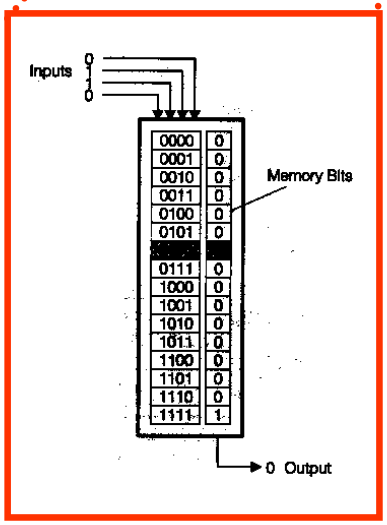


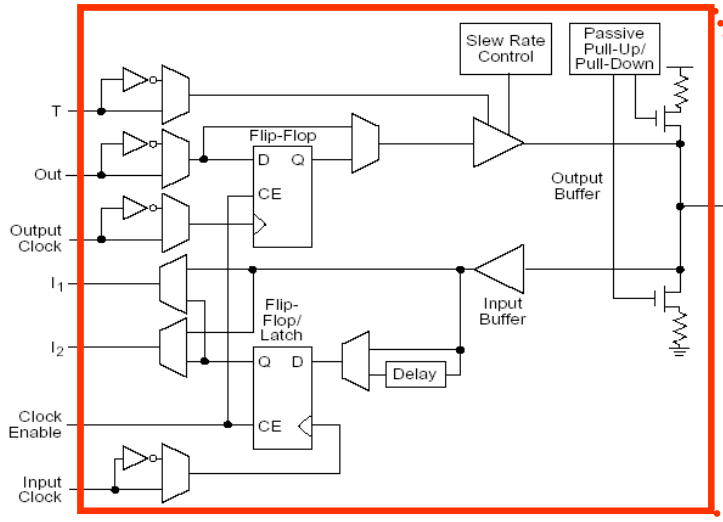
# Diseño semi custom

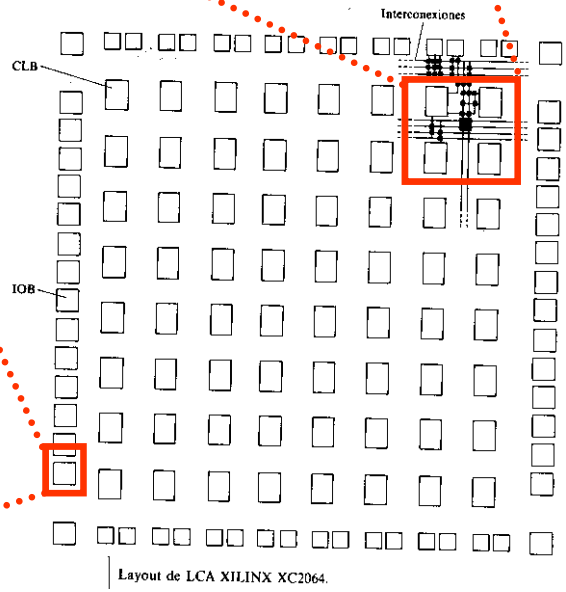
## arrays predifundidos: FPGA



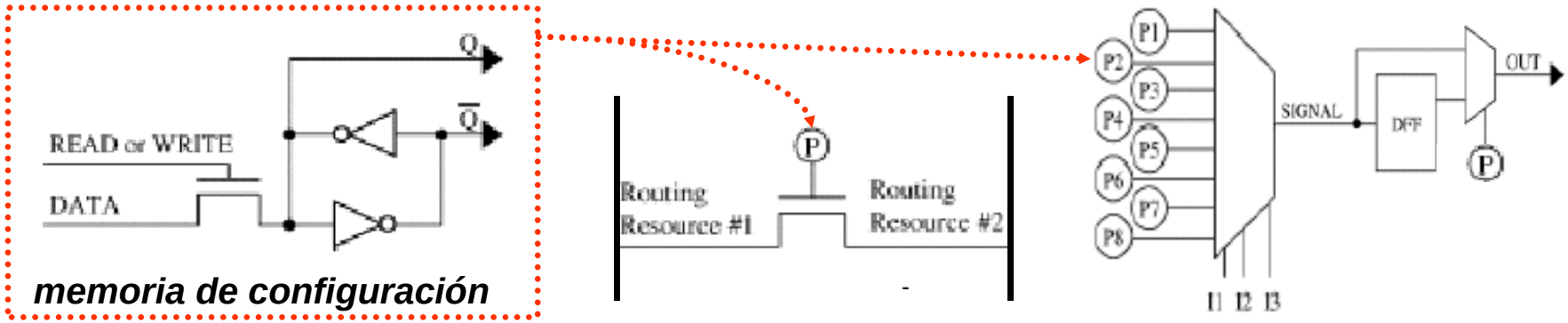



# Diseño semi custom

## arrays predifundidos: FPGA

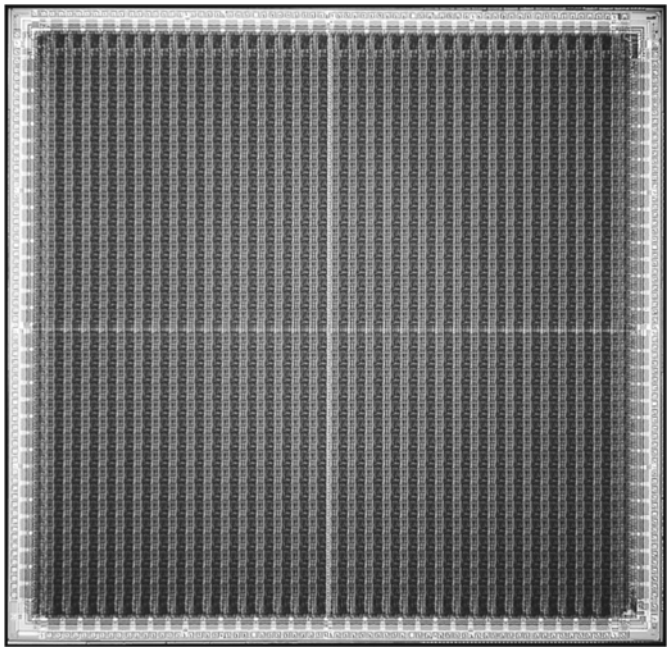
- Una FPGA se puede considerar formada por 2 planos de electrónica
  - Electrónica de usuario: CLB, IOBs, interconexionado (toda programable)
    - accesible por el diseñador que usa la FPGA como tecnología objetivo
  - Electrónica de soporte a la programación: puertas de paso, memorias de configuración, lógica de programación
    - accesible por el diseñador de la placa (o software) basada en FPGA
- Según el grado de variedad de elementos configurables
  - Arquitectura homogénea: CLBs + IOBs + nets
  - Arquitectura heterogéneas: CLBs + IOBs + nets + RAM + mult + micros + buses



# Diseño semi custom

## arrays predifundidos: FPGA

- Tipos de configuración
  - FPGA master: lee su propia configuración accediendo a una memoria externa no volátil
    - típicamente EPROM serie / paralela
  - FPGA slave: la configuración le es suministrada en serie o en paralelo (8 bits) por un dispositivo externo
    - típicamente CPLD, micros, FPGAs
    - estos dispositivos direccionan memorias (FLASH, SRAM, ...) o redirigen información entrante (puerto paralelo/serie, bus estándar, interfaz de red, ...)
    - también es posible la configuración directa desde un PC a través de JTAG
  - Varias FPGAs pueden ser configuradas en cascada
  - La configuración volcada puede ser leída junto con el valor de todos los FF (readback)



***Xilinx XC4025***

J.M. Mendías 2015

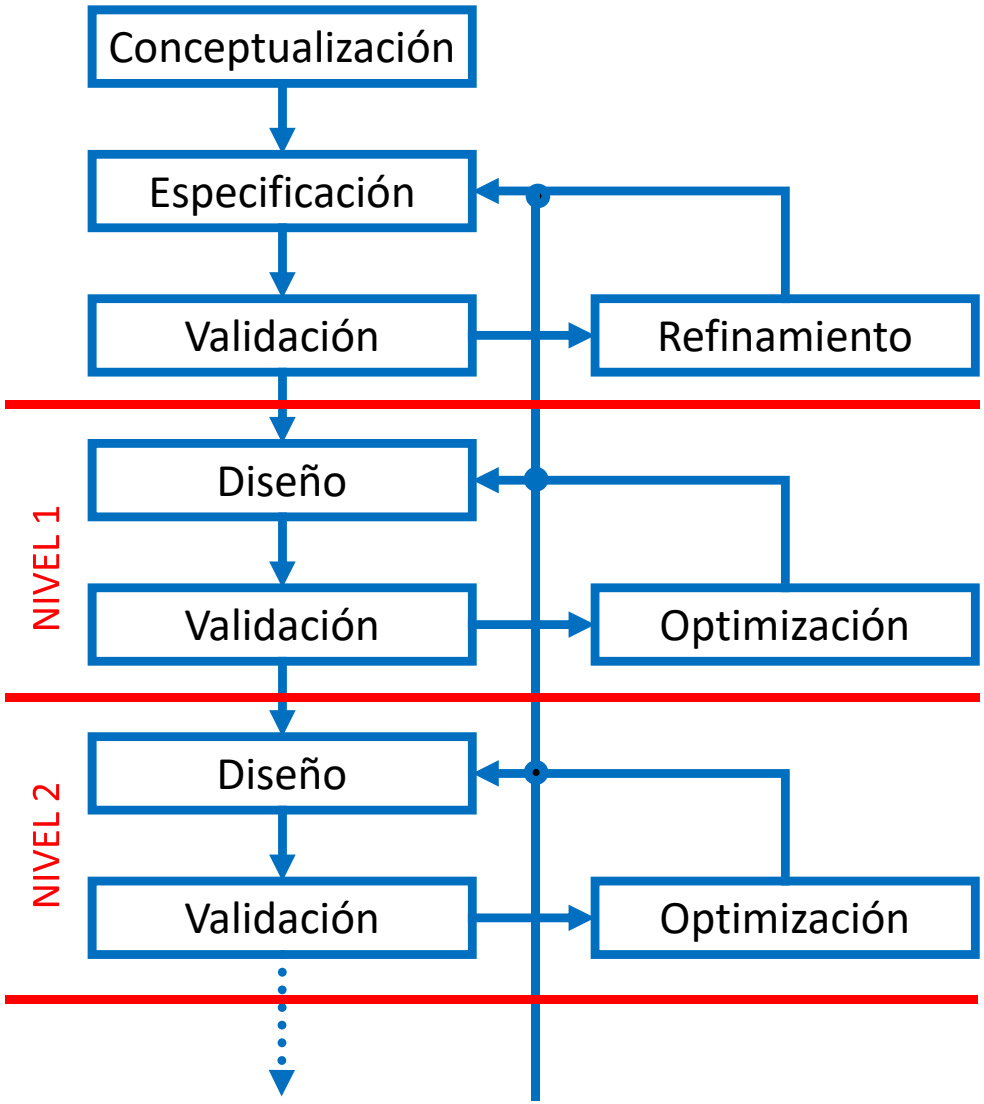


# Estilos de diseño

## comparativa

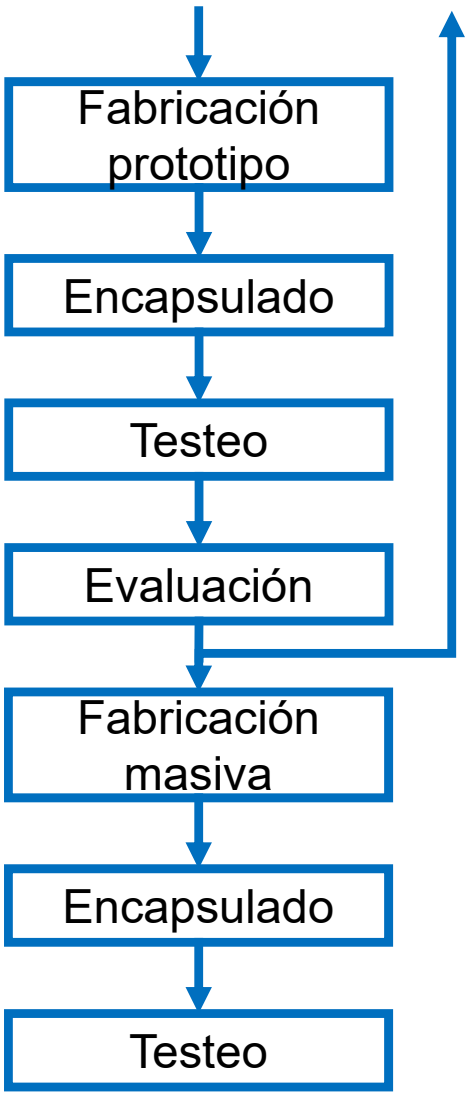

| | Full-custom | Basado en celdas | Arrays Predifundidos | Arrays Precableados |
|---|---|---|---|---|
| Tamaño de celda | Variable | Altura Fija | Fijo | Fijo |
| Tipo de celda | Variable | Variable | Fijo | Programable |
| Emplazamiento | Variable | En Filas | Fijo | Fijo |
| Interconexionado | Variable | Variable | Variable | Programable |
| Fiabilidad eléctrica | Media | Alta | Alta | Muy Alta |
| Densidad funcional | Muy Alta | Alta | Alta | Medio |
| Rendimiento funcional | Muy Alto | Alto | Alto | Medio |
| Flexibilidad en diseño físico | Muy Alta | Alta | Media | Ninguna |
| Tiempo de diseño físico | Muy Alto | Medio | Medio | Ninguno |
| Tiempo de fabricación | Medio | Medio | Bajo | Muy Bajo |
| Coste baja producción | Muy Alto | Alto | Alto | Bajo |
| Coste alta producción | Bajo | Bajo | Bajo | Muy Alto |

# ciclo de producción VLSI



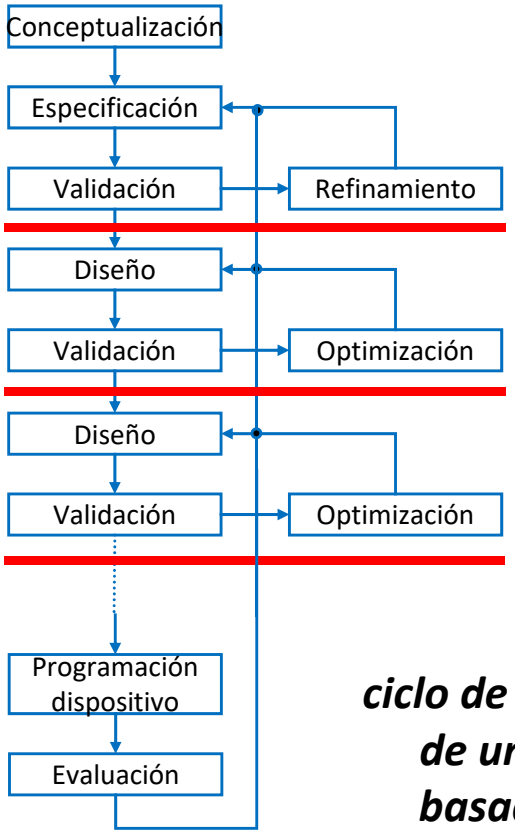

- Conceptualización: definición de los requisitos funcionales y no funcionales del circuito.
- Especificación o modelado: formalización de los anteriores requisitos en una representación uniforme y procesable por máquina.
- Validación: comprobación de que se ha especificado correctamente el concepto / se ha implementado correctamente lo especificado
- Refinamiento del modelo: mejora de la calidad de una especificación.
- Diseño o síntesis: transformación de la especificación en una implementación.
- Optimización: mejora de la calidad de una implementación.

# ciclo de producción VLSI

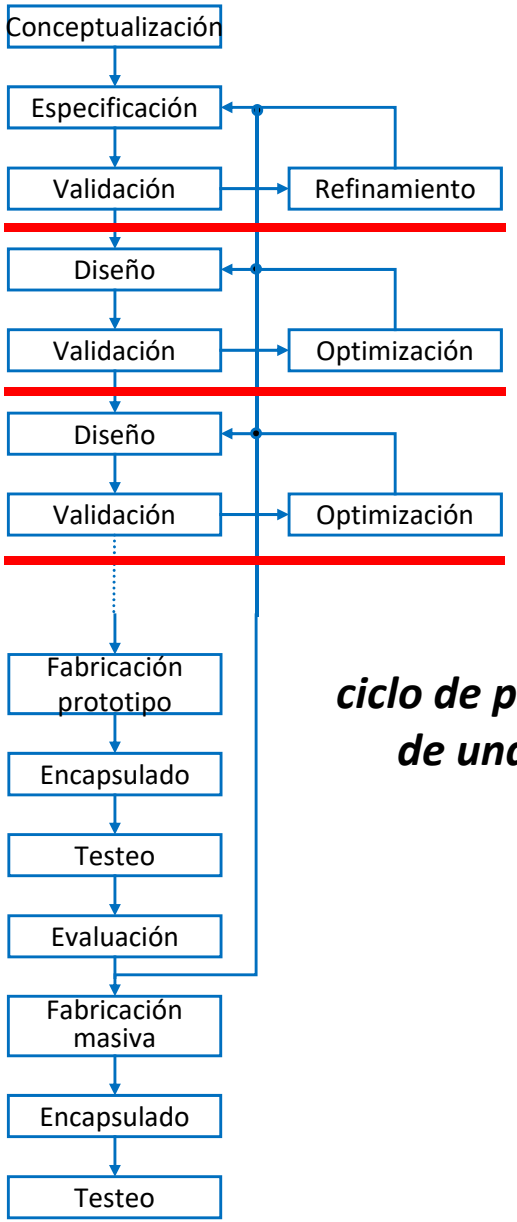



- Fabricación:
  - preparación de la oblea (10 cm de diámetro).
  - deposición, implantación y difusión de materiales en la oblea según las máscaras resultado del proceso de síntesis física.
- Encapsulado:
  - cortado de la oblea en dados (dies).
  - descartado de dados defectuosos por imperfecciones del substrato.
  - encapsulado del dado en un soporte plástico o cerámico.
  - soldado de los pads del dado con la patillas del chip.
  - sellado del chip.
- Validación de la producción o testeo: comprobación que se ha fabricado correctamente la implementación (de aspectos funcionales y no funcionales).

# ciclo de producción VLSI




***ciclo de producción de un diseño basado FPGA***




***ciclo de producción de una FPGA***

# FPGA de Xilinx


- Familia Spartan-3 (45 nm) 10-150€
  - Elementos configurables:
    - Celdas lógicas: hasta 75K (67K FF + 67K 4-input LUT ~ 1,1M puertas equivalentes)*
    - Pines: hasta 633 pines disponibles
    - SRAM interna: hasta 1,8 Mb
    - Multiplicadores/MAC: hasta 104 multiplicadores 18b / 126 MAC (18x18b mult + 48b add)
    - Digital clock manager (DCM)
- Familia Spartan-6 (45 nm) 20-550€
  - Elementos configurables:
    - Celdas lógicas : hasta 147K (184K FF + 92K 6-input LUT ~ 2,2 puertas equivalentes)
    - Pines: hasta 570 pines disponibles
    - SRAM interna: hasta 4,8 Mb
    - MAC: hasta 180 (18x18b mult + 48b add)
    - Clock management Tile (CMT): DCM + PLL
    - Transceivers de alta velocidad: hasta 8 a 3,2 Gb/s (max)
    - Interfaz PCI-express

(*) 1 celda lógica = 15 puertas equivalentes

# FPGA de Xilinx


- Familia Virtex-6 (28 nm) 200-2000€
  - Elementos configurables:
    - Celdas lógicas: hasta 567K (66K FF + 33K 6-input LUT ~ 8,5M puertas equivalentes)
    - Pines: hasta 1200 pines disponibles
    - SRAM interna: hasta 38 Mb
    - MAC: hasta 2016 (25x18b mult + 48b add)
    - Mixed Mode Clock Manager (MMCM)
    - Transceivers de alta velocidad: hasta 62 a 11 Gb/s (max)
    - Interfaz PCI-express: hasta 4
    - Ethernet MAC: hasta 4

# FPGA de Xilinx


- Familia Artix-7 / Kintex-7 / Virtex-7 (28 nm)
  - Elementos configurables:
    - Celdas lógicas: hasta 215K / 478K / 2M (2,4M FF + 1,2M 6-input LUT ~ 30M puertas equiv.)
    - Pines: hasta 500 / 500 / 1200 pines disponibles
    - SRAM interna: hasta 13 / 34 / 68 Mb
    - MAC: hasta 740 / 1920 / 3600 (25x18b mult + 48b add)
    - Clock management Tile (CMT): MMCM + PLL
    - Transceivers de alta velocidad: hasta 16 / 32 / 96 a 28 Gb/s (max)
    - Interfaz PCI-express: hasta 4 / 8 / 8
    - Ethernet MAC: hasta 4
    - Conversor analógico digital: ADC dual de 12b

# FPGA de Xilinx


- Familia Kintex UltraScale / Virtex UltraScale (20 nm)
  - Elementos configurables:
    - Celdas lógicas: hasta 1,2M / 4,4M (5M FF + 2,5 6-input LUT ~ 66M puertas equivalentes)
    - Pines: hasta 832 / 1456 pines disponibles
    - SRAM interna: hasta 76 / 133 Mb
    - MAC: hasta 5,5K / 2,9K (27x18b mult + 48b add)
    - Clock management Tile (CMT): MMCM + PLL
    - System monitor
    - Transceivers de alta velocidad: hasta 64 / 120 a 16,3 Gb/s / 30,5 Gb/s (max)
    - Interfaz PCI-express: hasta 6 / 6
    - 100G Ethernet: hasta 9 (solo virtex)
    - 150G Interlaken: hasta 9 (solo virtex)

# FPGA de Xilinx


- Familia Kintex UltraScale+ / Virtex UltraScale+ (16 nm)
  - Elementos configurables:
    - Celdas lógicas: hasta 915K / 2,9M (3,3M FF + 1,6M 6-input LUT ~ 44M puertas equiv.)
    - Pines: hasta 668 / 832 pines disponibles
    - SRAM interna: hasta 71 / 526 Mb
    - MAC: hasta 3,5K / 12K (27x18b mult + 48b add)
    - Clock management Tile (CMT): MMCM + PLL
    - System monitor
    - Transceivers de alta velocidad: hasta 76 / 128 a 32,8 Gb/s (max)
    - Interfaz PCI-express: hasta 5 / 6
    - 100G Ethernet: hasta 9 (solo virtex)
    - 150G Interlaken: hasta 12 (solo virtex)
    - Video codec (solo kintex)

# SoC de Xilinx


- Zynq-7000 SoC (16 nm) 60-3000€
  - Elementos programables:
    - Dual-core ARM Cortex A9 (1GHz) / coprocesador coma flotante
    - Cache: L1 32+32 KB / L2 512 KB
    - SRAM interna: 256 KB
    - Controlador de memoria: DDR2 / DDR3
    - Controladores de dispositivos: 2x USB, 2x UART, 2x CAN, 2x I2C, 2x SPI, 4x 12b-GPIO, 2x Gigabit Ethernet, 2x SD
  - Elementos configurables:
    - Celdas lógicas: hasta 444K (555K FF + 277 6-input LUT ~ 6,6M puertas equivalentes)
    - SRAM interna: hasta 3 Mb
    - MAC: hasta 2020 (25x18b mult + 48b add)
    - Interfaz PCI-express: hasta 8
    - Conversor analógico digital: ADC dual de 12b

# SoC de Xilinx

- Zynq UltraScale MPSoc (16 nm)
  - Elementos programables:
    - Quad-core ARM Cortex A53 (1,3GHz)
    - Memoria interna: Cache L1 32+32 KB (por core) / Cache L2 1 MB / SRAM 256 KB
    - Dual-core ARM Cortex R5 (600MHz) / GPU Mali 400MP (466MHz)
    - Memoria interna: Cache L1 32+32 KB (por core) / Cache L2 64 KB / SRAM 128 KB
    - Memoria interna:
    - Controlador de memoria: DDR3 / DDR4
    - Controladores de dispositivos: 2x USB, 2x UART, 2x CAN, 2x I2C, 2x SPI, 4x 12b-GPIO, 4x Gigabit Ethernet, 2x SD
  - Elementos configurables:
    - Celdas lógicas: hasta 915K (1M FF + 522K 5-input LUT ~ 14M puertas equivalentes)
    - SRAM interna: hasta 71 Mb
    - MAC: hasta 2528
    - System monitor
    - Interfaz PCI-express: hasta 5
    - 100G Ethernet: hasta 4
    - 150G Interlaken: hasta 4
    - Video codec

# Acerca de *Creative Commons*


- Licencia CC (Creative Commons)
  - Ofrece algunos derechos a terceras personas bajo ciertas condiciones. Este documento tiene establecidas las siguientes:

Reconocimiento (*Attribution*):
En cualquier explotación de la obra autorizada por la licencia hará falta reconocer la autoría.

No comercial (*Non commercial*):
La explotación de la obra queda limitada a usos no comerciales.

Compartir igual (*Share alike*):
La explotación autorizada incluye la creación de obras derivadas siempre que mantengan la misma licencia al ser divulgadas.

Más información: https://creativecommons.org/licenses/by-nc-sa/4.0/