

Tema 11 - FPGAs para automatización

Sistemas Electrónicos para Automatización
Grado en Ingeniería Electrónica, Robótica y
Mecatrónica

Hipólito Guzmán Miranda

Contenido

- FPGAs como System-on-Chip
- El 'Design Gap'
- Soft processors y diseño con IP cores
- Buses para microprocesadores empotrados

Pero antes de todo esto...

¿Sabéis cómo es una FPGA por dentro?

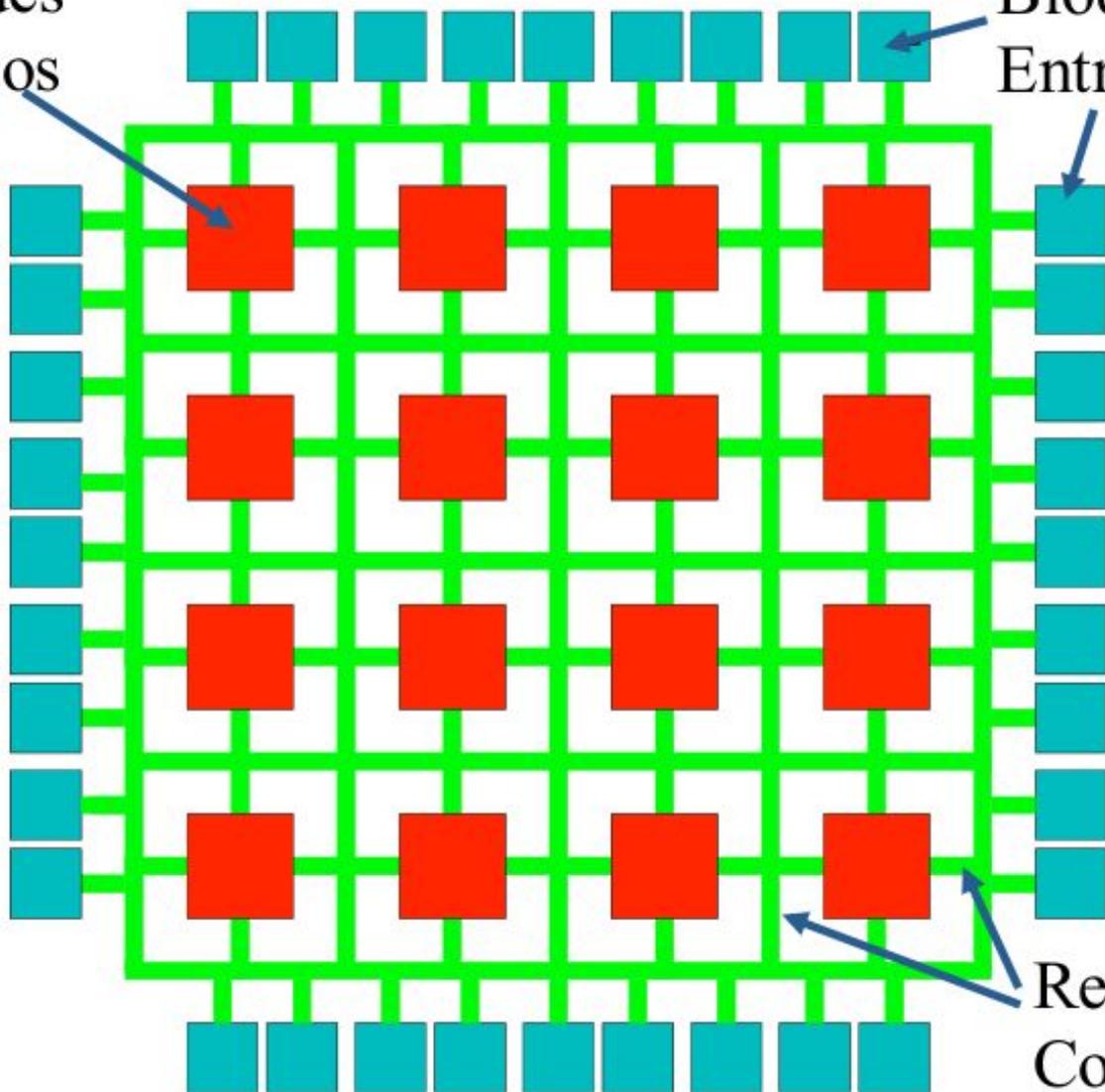
Arquitectura interna de una FPGA

- IOBs: In/Out Blocks (Bloques de Entrada/Salida)
- CLBs: Configurable Logic Blocks (Bloques Lógicos Configurables)
- Routing Resources (Recursos de Conexión)
- (Re)Programabilidad

Arquitectura de una FPGA

Bloques Lógicos

Bloques de Entrada/Salida



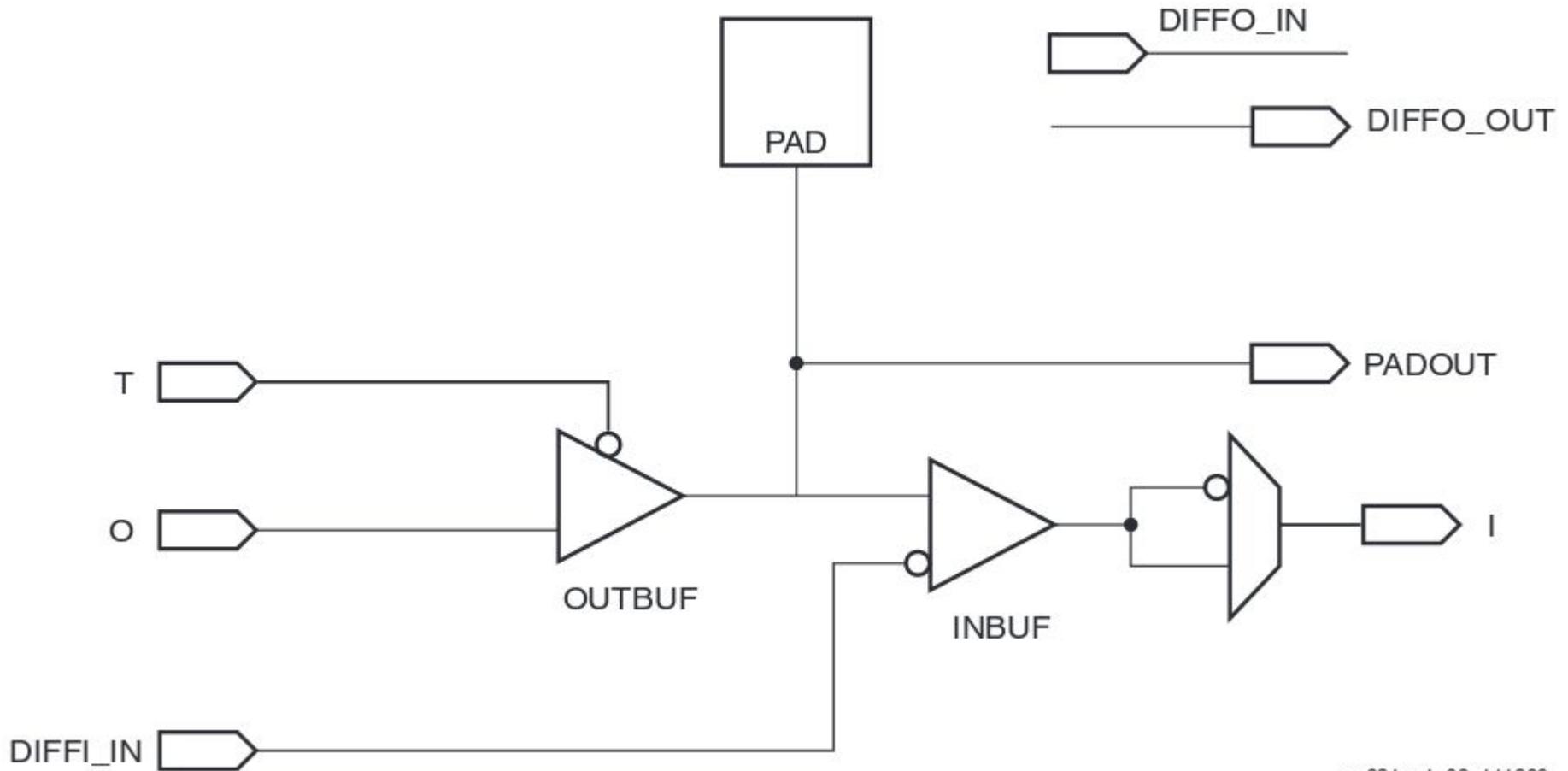
Programabilidad

Recursos de Conexión

In-Out Blocks (IOBs)

- PAD (conexión al exterior)
- Buffer de entrada
- Buffer de salida (triestado)
- Soporte para entradas/salidas diferenciales (dependiendo de la familia)

IOB Spartan-6



ug381_c1_02_111309

Configurable Logic Blocks (CLBs)

Compuestas por:

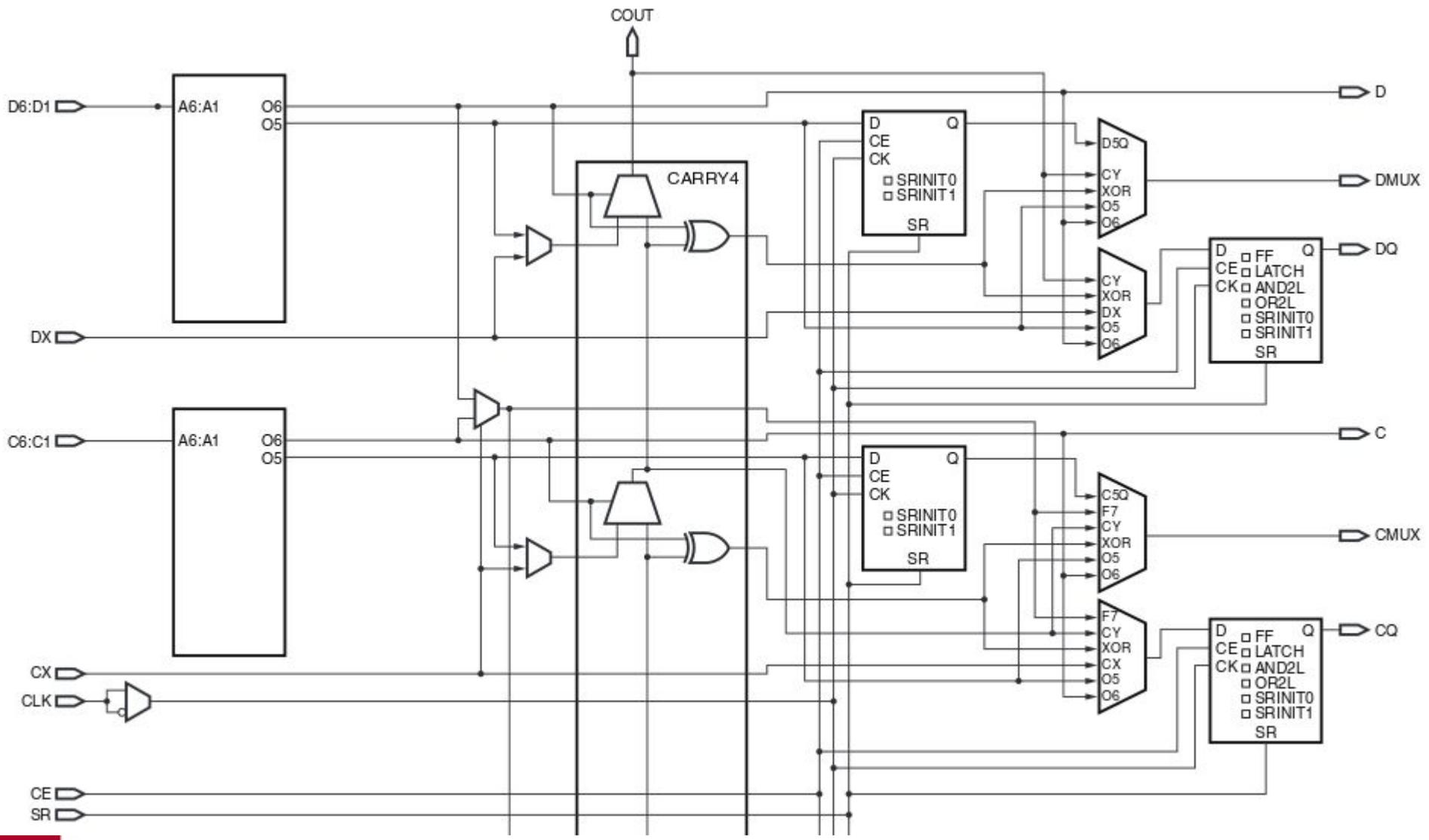
- $K * N$ -input LUT (Look-Up Tables de N entradas)
- $K * \text{Flip-flops configurables}$

$K = 2$ en tecnologías antiguas, $4+$ en tecnologías modernas.

N crece también en tecnologías modernas (6 en Spartan-6)

Los CLB se pueden dividir en 'Slices'

1/2 Slice Spartan-6 (= 1/4 CLB)

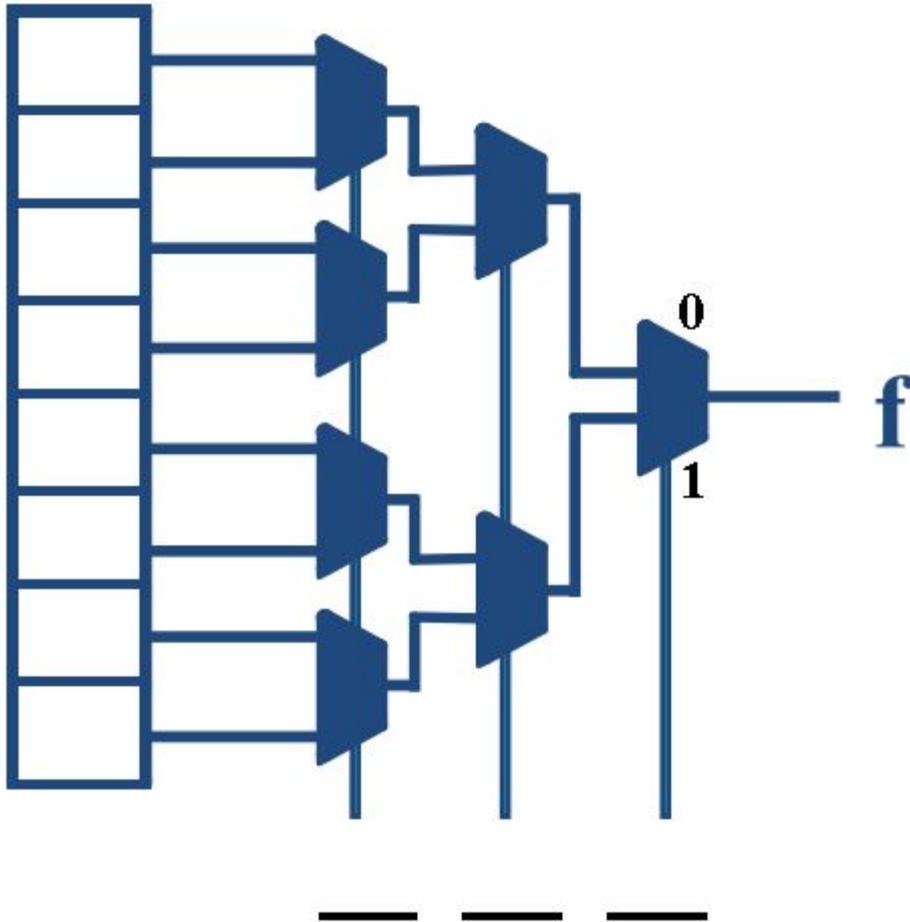


Look-Up Tables (LUTs)

En lugar de implementar las funciones lógicas con puertas lógicas, en FPGA se implementan con tablas de verdad

- Ej: Una LUT de 4 entradas ('4-LUT') puede implementar cualquier función lógica de 4 entradas

SRAM



3-LUT

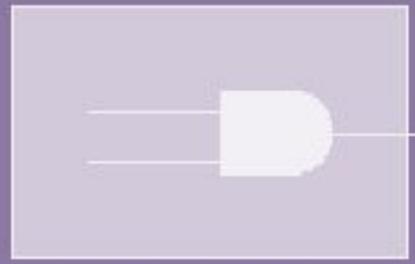
Ejercicio

Configura la LUT de forma que implemente la función

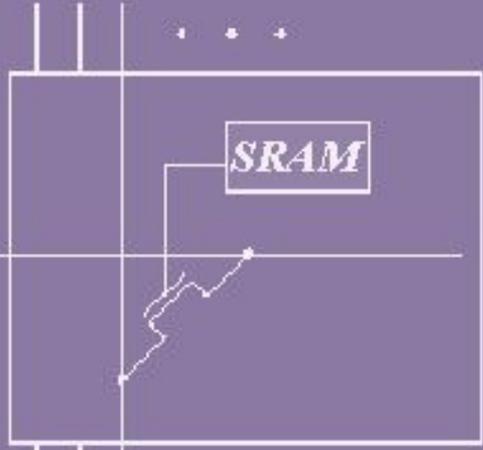
$$F = ABC + A\bar{B}\bar{C}$$

Recursos de Rutado

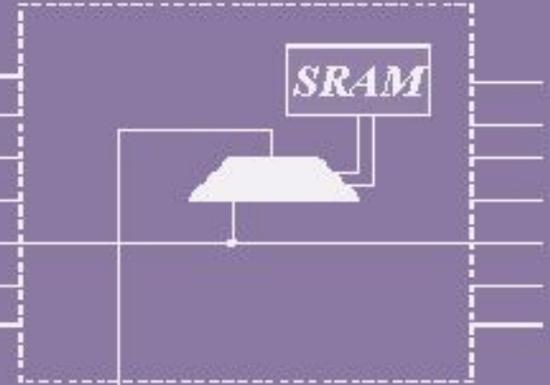
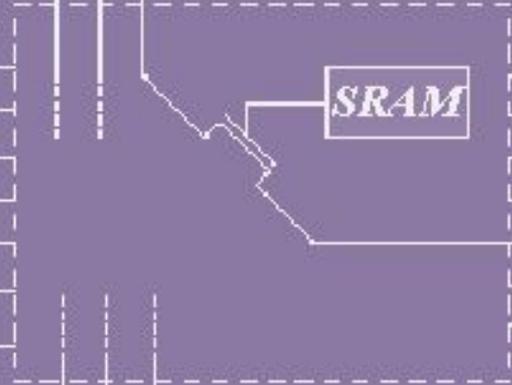
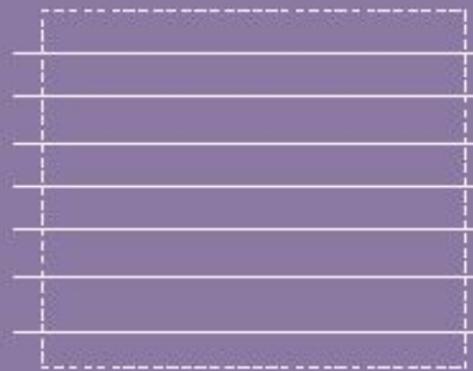
- PIP: Programmable Interconnection Points (Puntos de Interconexión Programable)
- Líneas cortas y largas
- Recursos dedicados para relojes (ej: BUFG)



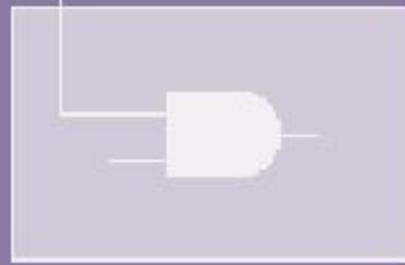
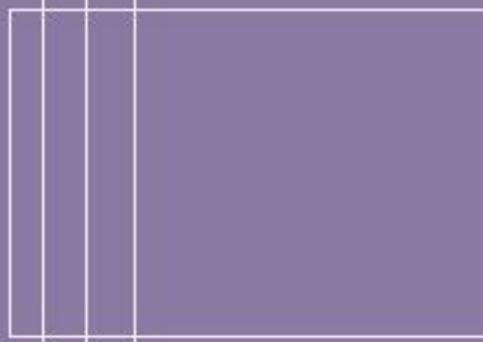
Celdas Lógicas



Celdas Lógicas



Celdas Lógicas



Celdas Lógicas

Configurabilidad y Reconfigurabilidad

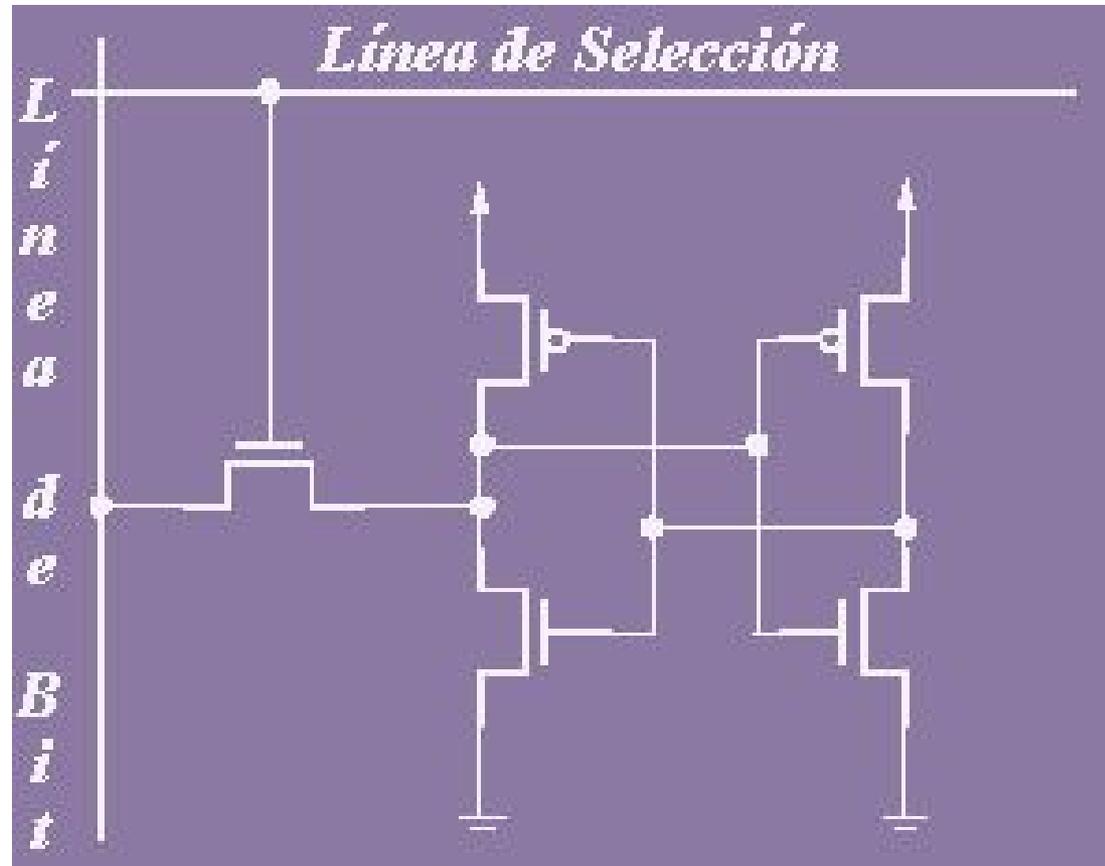
Existen varias tecnologías:

- SRAM: reconfigurable, volátil, muy extendida, aprovecha proceso CMOS estándar
- Flash: reconfigurable, no volátil, proceso no estándar
- Antifusible: no reconfigurable, proceso no estándar

Celda SRAM

Son dos
inversores
realimentados

4 transistores,
pero CMOS
estándar



Tecnologías Flash

Se basan en el uso de FGMOS (Floating-Gate MOS)

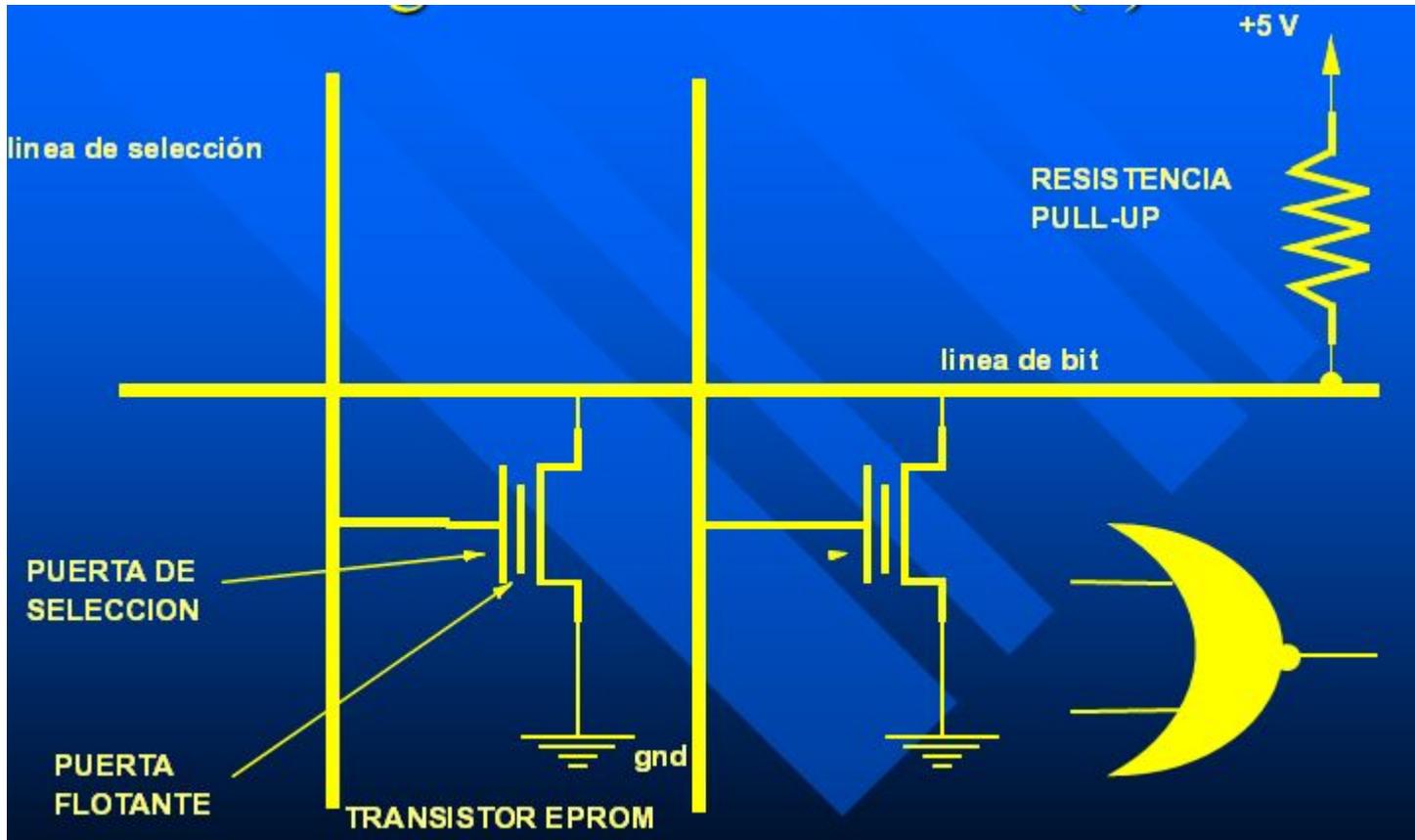
- Por lo que requieren de tecnologías con 2 niveles de polisilicio

Si la puerta flotante está cargada:

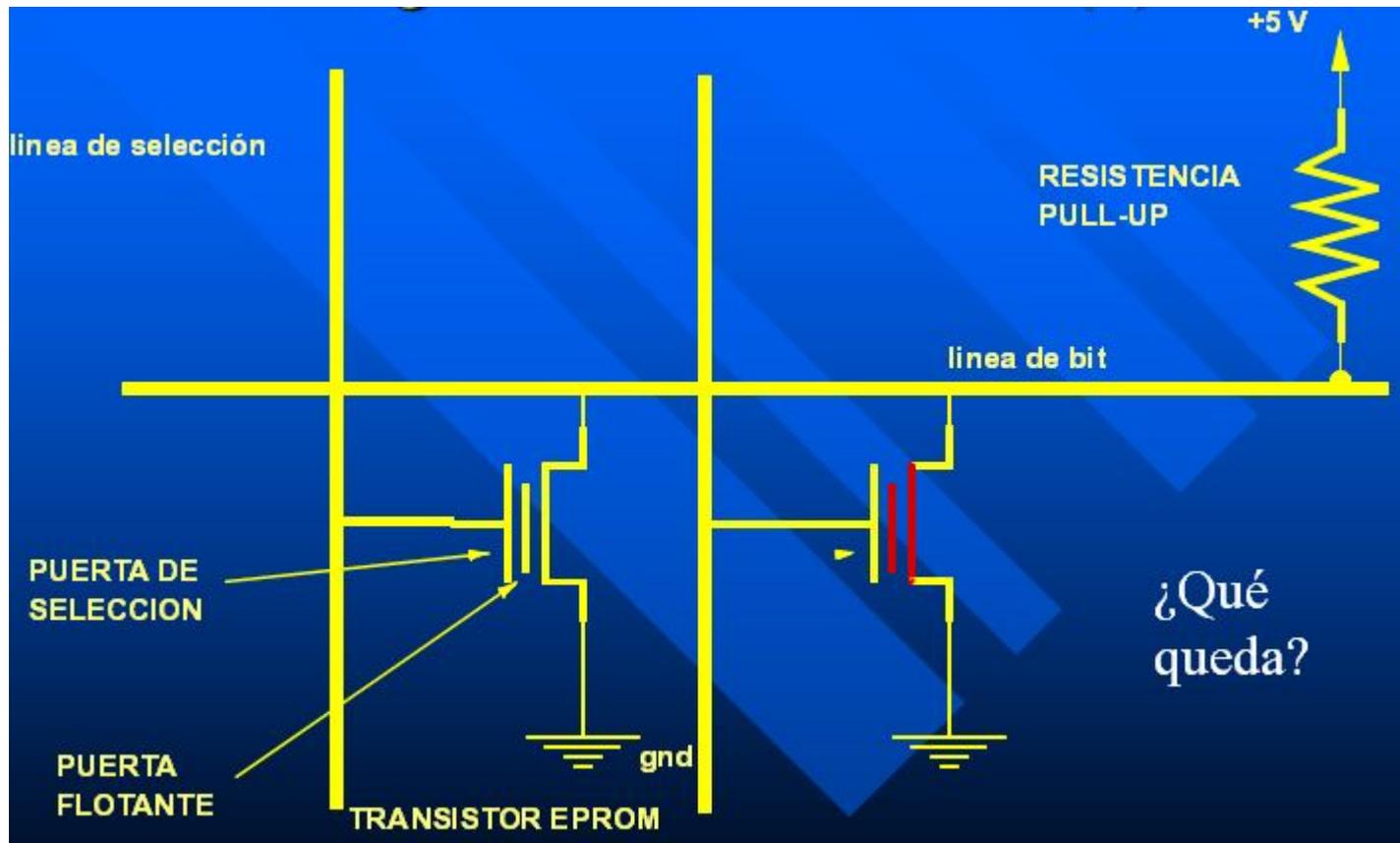
-> Incremento de la V_t

-> Transistor no puede encenderse ni con V_{dd} en la puerta

Ejemplo configurabilidad Flash

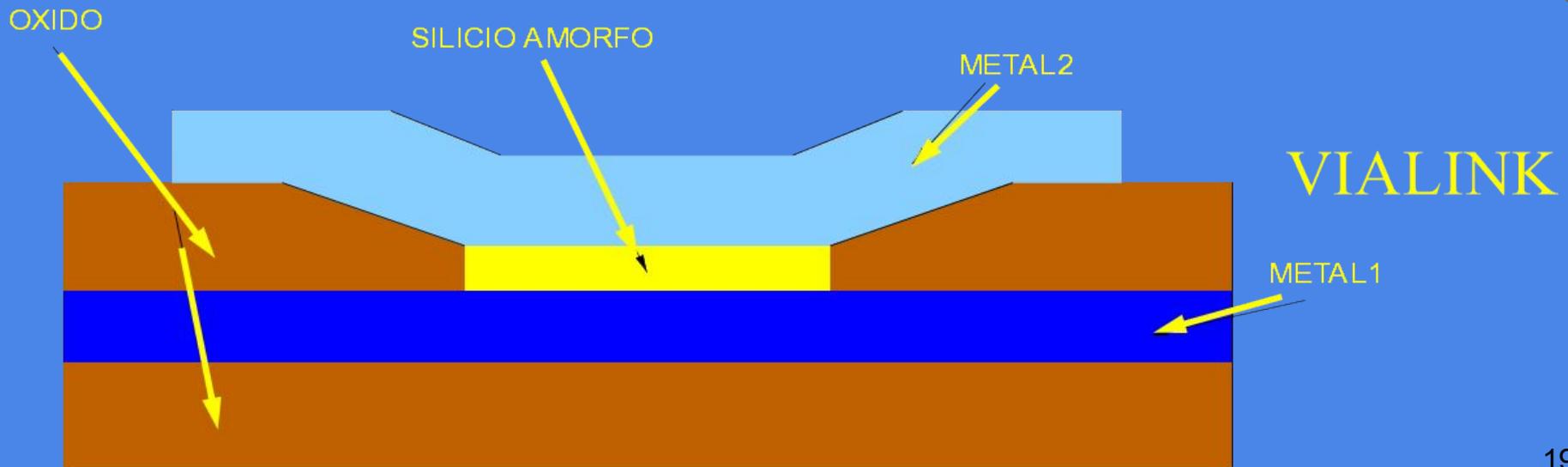


Ejemplo configurabilidad Flash



Tecnologías Antifusible

- OTP: One Time Programmable
- Requieren un proceso específico (no CMOS estándar)

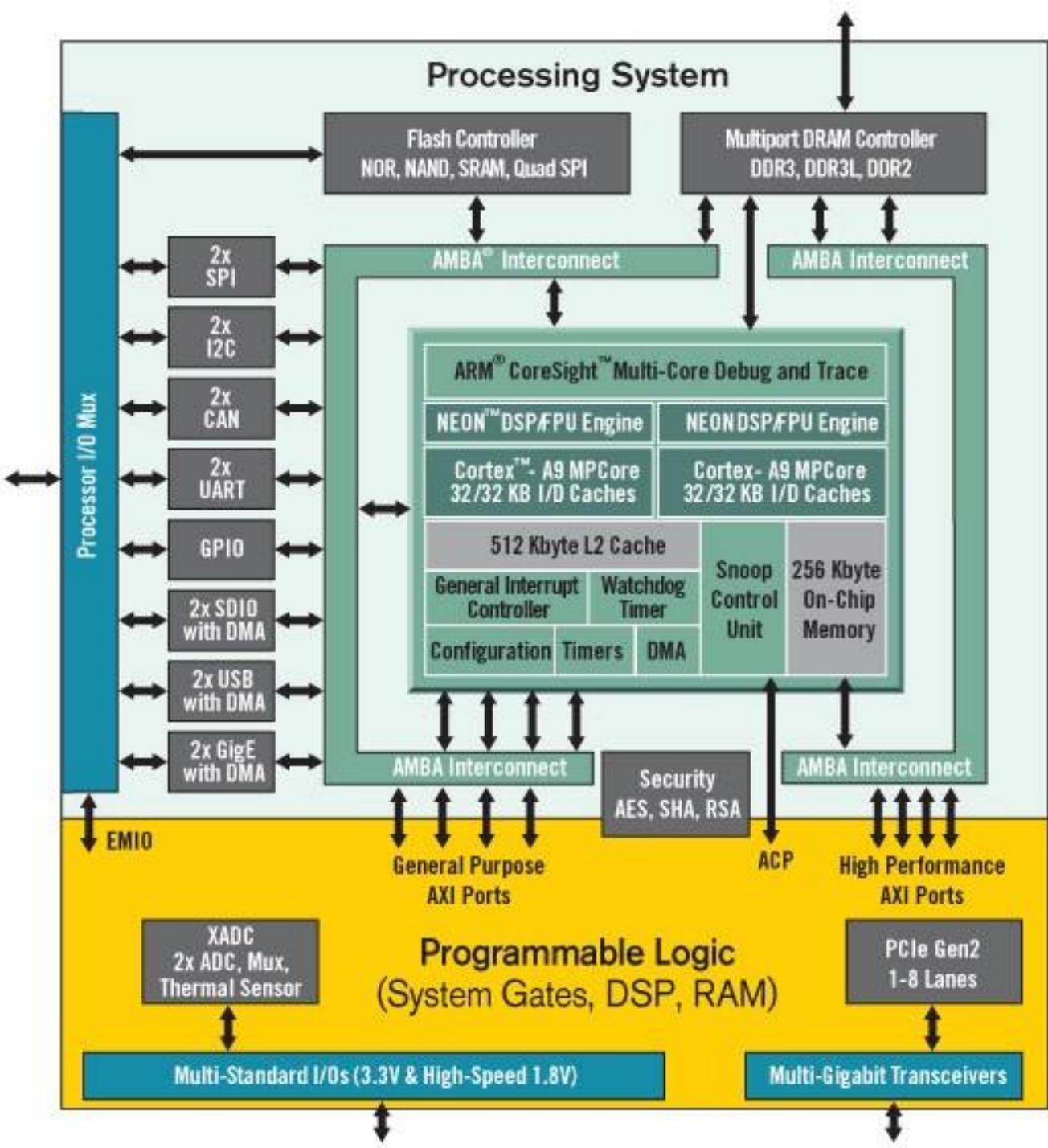


Contenido

- FPGAs como System-on-Chip
- El 'Design Gap'
- Soft processors y diseño con IP cores
- Buses para microprocesadores empotrados

FPGAs como System-on-Chip

- Evolución de las tecnologías microelectrónicas (Ley de Moore)
- Nos lleva a la siguiente arquitectura de una FPGA moderna:

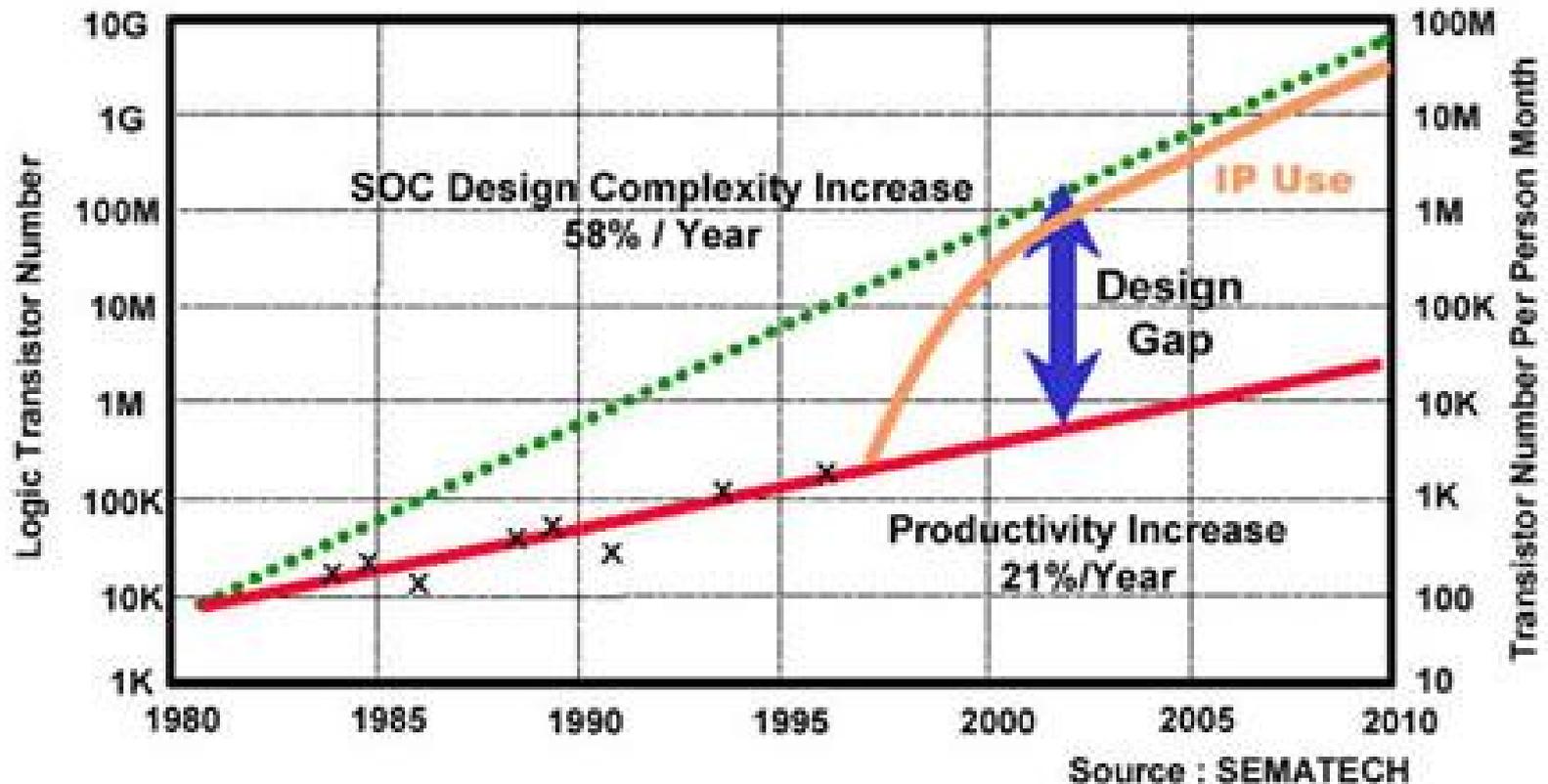


Familia Zynq

Arquitectura de una FPGA moderna

- Además de IOBs, CLB's y recursos de rutado:
- Memorias empotradas (Block RAMs)
- Conexiones de alta velocidad (Gigabit transceivers, PCIe, ...)
- Microprocesadores!

El 'Design Gap'



El 'Design Gap'

- La capacidad de diseño crece más lento que la capacidad de fabricación
- Si un diseñador diseña a 100 puertas por día, y en un chip caben 10M puertas... tardamos 100K días en diseñar el sistema completo : 500 ingenieros * 1 año

El 'Design Gap': soluciones

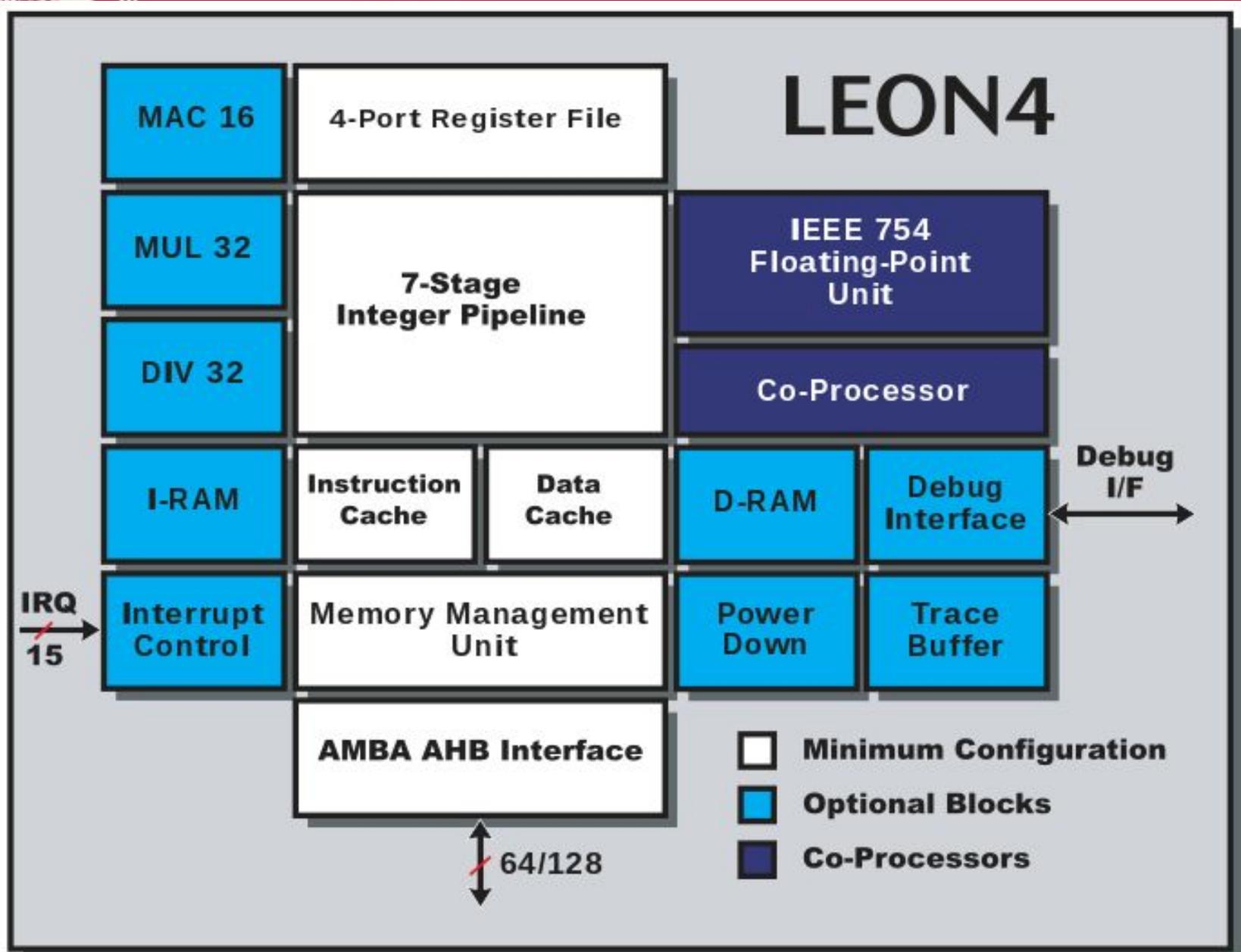
- Nuevas metodologías de diseño (síntesis de alto nivel)
- Uso de IP cores
- Uso de soft processors

Ojo! Diseños más complejos implica mayor esfuerzo en **verificación!** También existe un 'verification gap'

Soft processors

- No necesitamos una FPGA de última generación para tener un microprocesador
- Si tuviéramos una descripción en VHDL/Verilog de una ALU, registros, contador de programa, decodificador de instrucciones... = **un microprocesador**
- Aunque no esté implementado en silicio (como una BRAM), podríamos dedicar una parte de la FPGA a implementar un microprocesador

Soft Processors



No necesitamos crearlo desde cero

Algunos soft processors:

- Microblaze (Xilinx)
- Nios II (Altera)
- Leon 4 (Aeroflex Gaisler)
- Plasma (Opencores)
- OpenSparc
- OpenRisc
- ...

Algunos problemas:

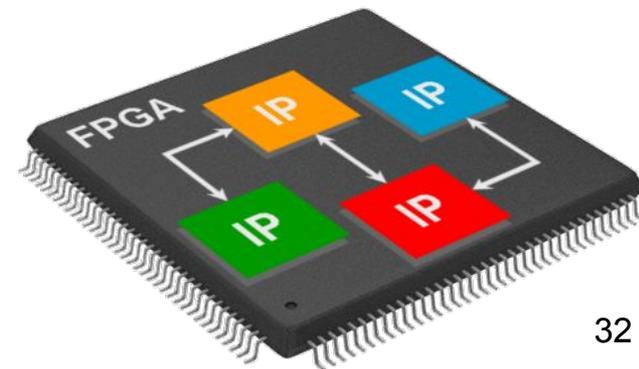
- Configurar los periféricos y mapa de memoria
- Disponer de un toolchain completo (compilador, linker, etc)
- Sistema operativo o programa 'standalone'
- Configurar las BRAM con el ejecutable
- Desarrollo de periféricos 'custom'
- Disponer de un modelo de simulación
- Conseguir que arranque el micro!

Algunas ventajas:

- Diseño óptimo: HW y SW se encargan cada uno de lo que les es más eficiente
- Simplicidad en las comunicaciones de tu diseño HDL con el exterior: USB, TCP/IP, ...
- Una actualización no es sólo cambiar el programa: podemos añadir periféricos nuevos (por ejemplo un timer)

Diseño con IP cores

- Esfuerzo evelado de desarrollar un sistema complejo
- Reutilizar módulos que ya estén probados
- Reducción del esfuerzo de diseño
- Principal problema es la **integración** de los módulos:
 - Interfaces
 - Calidad de la documentación
 - Configuración de los IP cores



IP (Intellectual Property) Cores deben ser:

- Reusables
- Configurables
- Simulables con los simuladores estándares de la industria
- Con interfaces basadas en estándares
- Verificados con un alto nivel de confianza
- Completamente documentados

Consejos

- Un buen integrador acelera el diseño tanto o más que un buen diseñador
- Hay que entender los 'quirks' de los fabricantes/proveedores
- Es extremadamente recomendable simular casos básicos para hacerse a los bloques
- Leer mucho y probar poco a poco!!

Buses para microprocesadores empotrados

IP cores específicos tienen interfaces específicos. Un softcore específico soportará uno o varios buses:

- PLB, AXI4 (MicroBlaze)
- Wishbone (estándar en Opencores)
- AMBA (Leon)

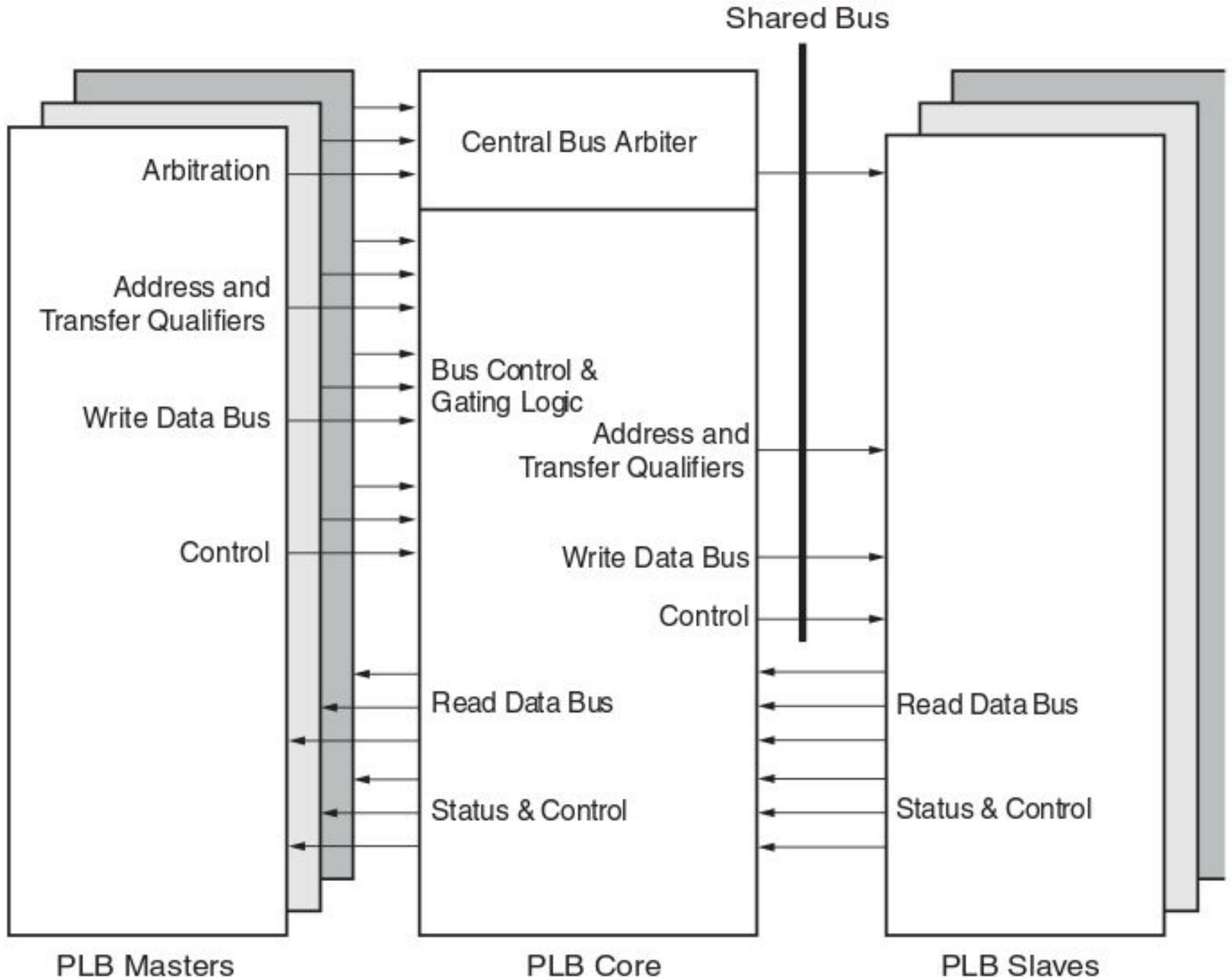
Buses para microprocesadores empotrados

Los IP cores de terceros que integremos y los que desarrollaremos nosotros deberían tener un interfaz con el bus elegido

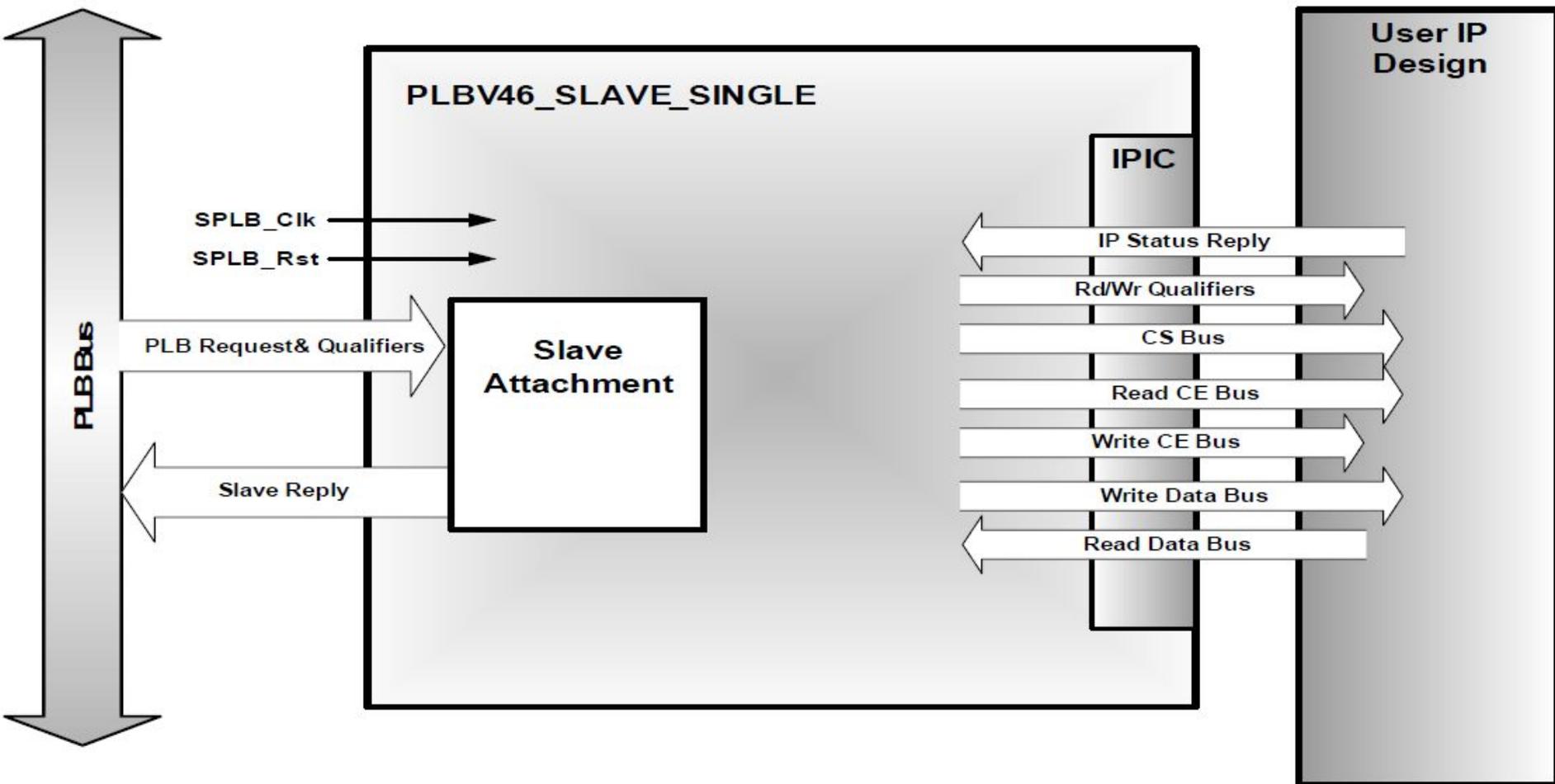
Aunque si no necesitamos velocidad podremos utilizar una conexión por GPIO

De esta forma: diseño de System-On-Chip complejos utilizando IP cores basados en estándares

PLB: maestro, árbitro y esclavos



PLB: IP de usuario



Buses para microprocesadores empotrados

Nosotros trabajaremos con:

- Spartan 6
- Spartan 6 LX9 MicroBoard
- Microblaze
- PLB bus (Processor Local Bus)
- Software Xilinx EDK (Embedded Development Kit - disponible en el CDC)

Referencias

- [Xilinx Large FPGA Methodology Guide \(UG872\)](#)
- [Spartan-6 FPGA Configurable Logic Block User Guide \(UG384\)](#)
- [Spartan-6 FPGA SelectIO Resources User Guide \(UG381\)](#)
- [Logicore IP Processor Local Bus \(PLB\) Product Specification](#)
- Michael Keating, Pierre Bricaud, [Reuse Methodology Manual for System-on-a-Chip Designs](#)