Inhalt

- 2.4 Spezialprozessoren
  - 2.4.1 Mikrocontroller (µC)
  - 2.4.2 Digitale Signalprozessoren (DSPs)
  - 2.4.3 Grafik Prozessoren (GPU)
- **2.5 Bus, Network-on-Chip (NoC) & Multicore**
- 2.6 Anwendungs-spezifische Instruktionssatz Prozessoren (ASIP)
- 2.7 Field Programmable Gate Arrays (FPGA)
- 2.8 System-on-Chip (SoC)

2.5.1 Bussystem (Bus)

- Den Daten und Informationsaustausch zwischen verschiedenen Systemkomponenten ermöglichende, als Linien- oder Ringnetz aufgebaute mehradrige Sammelleitung, an die alle Komponenten angeschlossen sind.
- Sie verbindet den Ausgang jeder Komponente mit den Eingängen aller übrigen.
- Der Datenaustausch zwischen den Komponenten erfolgt im Multiplex-Betrieb.
- Transportsystem für Informationen mehrerer Teilnehmer (Broadcast).

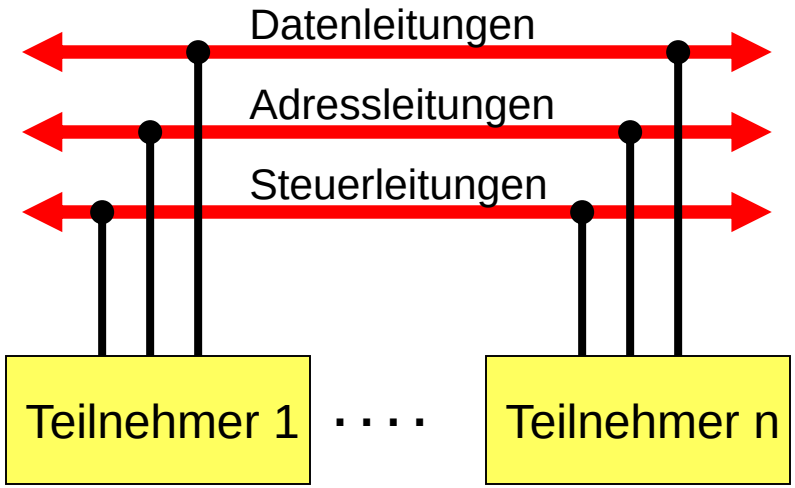
2.5.1 Bussystem (Bus) - Anwendungsbereich

- Verbindung von Komponenten auf allen Ebenen:
  - funktionale Einheiten auf Chips (on-Chip Bus)
  - Chips zum System/Rechner (Systembus)
  - Rechner zu Ein-/Ausgabegeräten (Peripheriebus)
  - Steuergeräte/Computer in Fahrzeugen, Flugzeugen (z.B. Feldbus)

- Teilnehmer sind über (Bus-)Schnittstelle an den Bus angeschlossen
  - Schnittstellendefinition ist Teil der Busdefinition
    (zusätzlich z.B. Festlegung der Buslänge, Zahl von Einschubplätzen, Terminierung)

2.5.1 Parallele Bussysteme

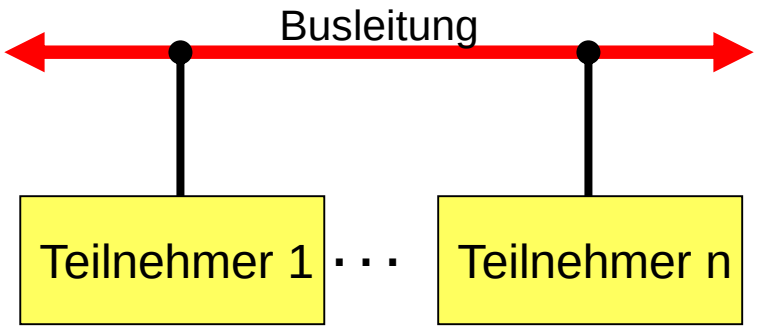
- Hardware:



- Datenbus:
  - Eigentliche Datenübertragung, typ. Wortbreite: 8, 16, 32, 64, 128 Bit
- Adressbus:
  - Auswahl einzelner Geräte und Adressen innerhalb der Geräte
    - z.Teil: Daten und Adressen zeitversetzt auf den selben Leitungen (z.B. IEC-Bus)
- Steuerbus:
  - Busanforderung, Arbitrierung, Interrupts, Handshaking
  - Versorgungsbus: Stromversorgung und Taktleitung

2.5.1 Serielle Bussysteme

- Hardware:



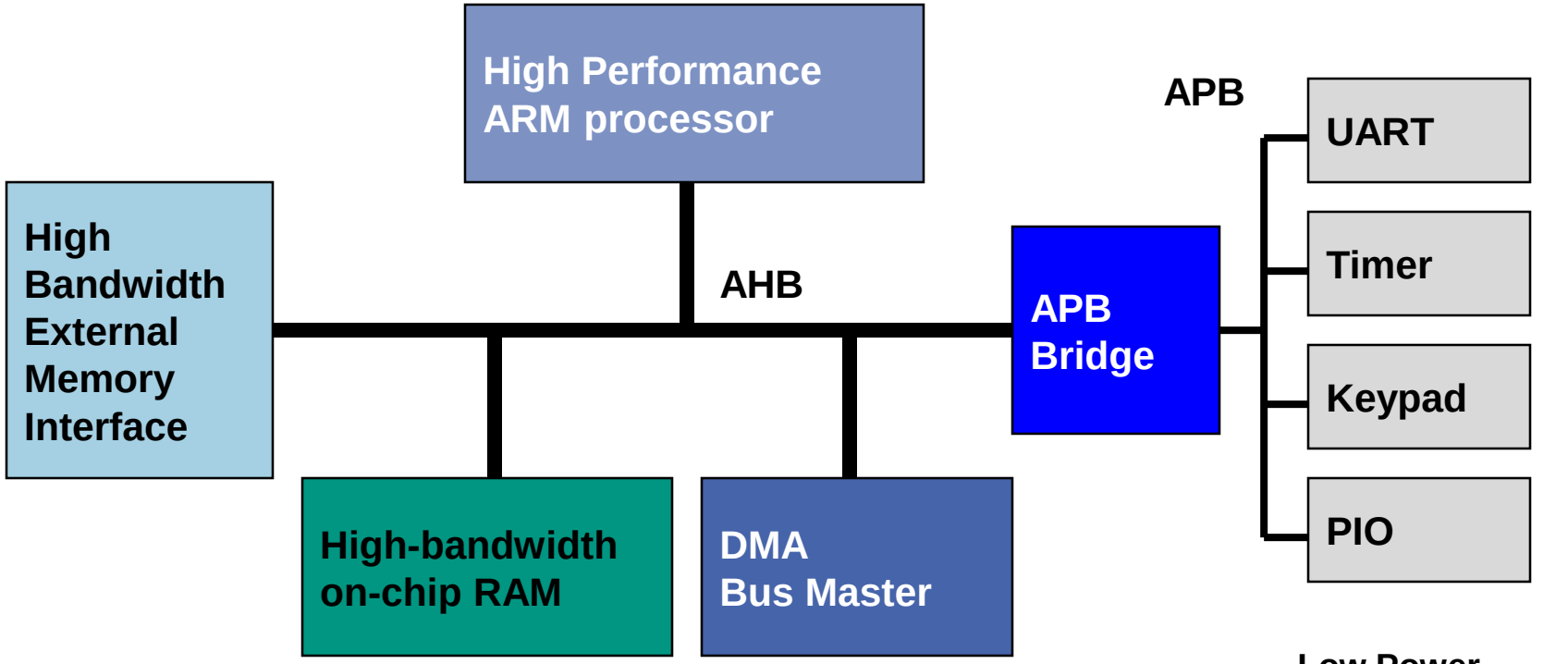
- Nur eine Leitung, die als Busstruktur ausgebildet ist.

- Funktionaler Aufbau (Busprotokoll):
  - Funktionen, die bei parallelen Bussystemen durch spezielle Leitungen übernommen werden, sind hier durch (Software-) Protokolle realisiert.
  - serielle Busse sind in Bezug auf Änderungen im allgemeinen flexibler als parallele Busse.

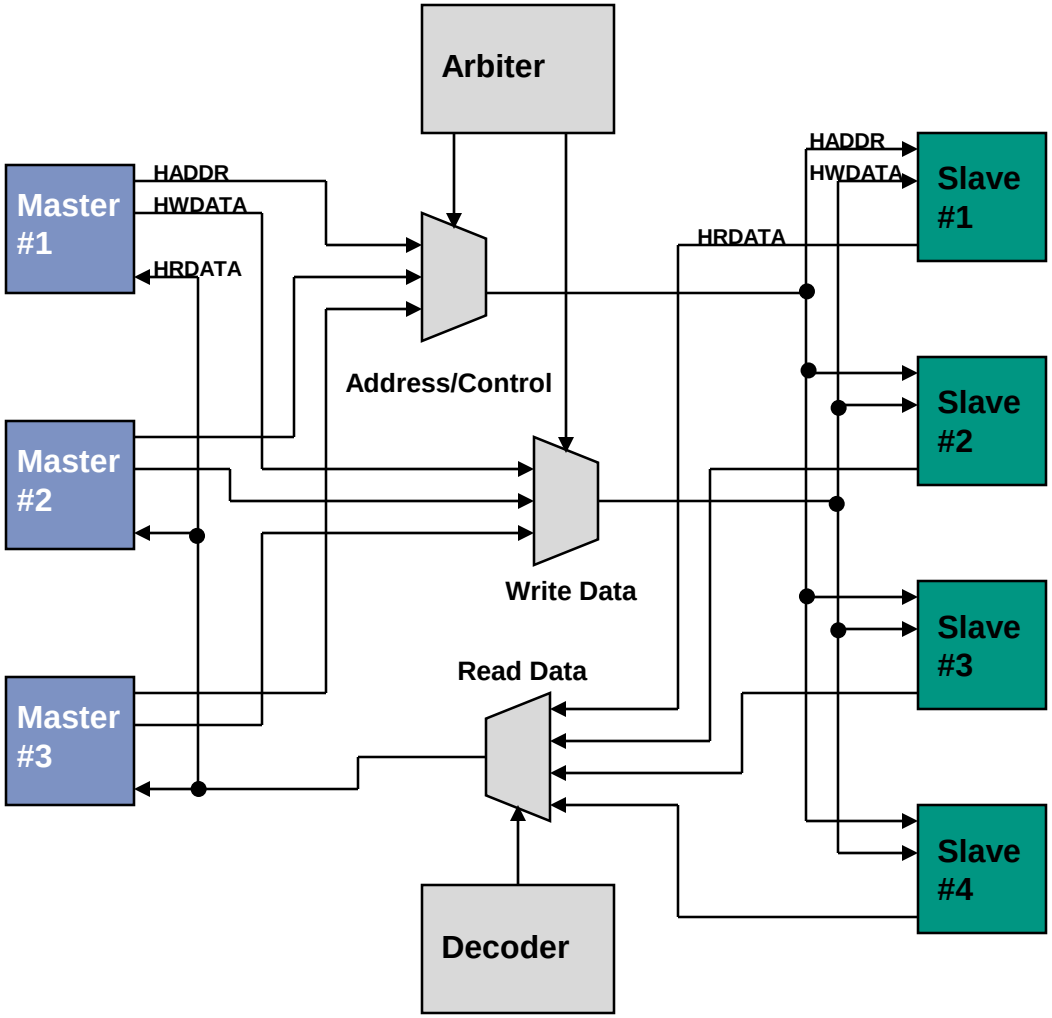
2.5.1 Bussystem Beispiel: Advanced Microcontroller Bus Architecture (AMBA)

- Offener Standard einer on-chip Bus Spezifikation für Plattform-basierte Designs (SoC).
- IP Module: eingebettete Prozessoren, Speicher, periphere Einheiten
- Wiederverwendbarkeit der Module durch gemeinsamen “backbone”.
- 2 on-chip Busse mit einer Bridge verbunden:
  - high-speed bus:
    verbindet Prozessoren, DMA controller, on-chip memory, high-performance units
  - low-speed, lower-power peripheral bus:
    einfacheres Protokoll, verbindet Timer, GPIO, SIO, ….

2.5.1 Ein AMBA Beispiel System

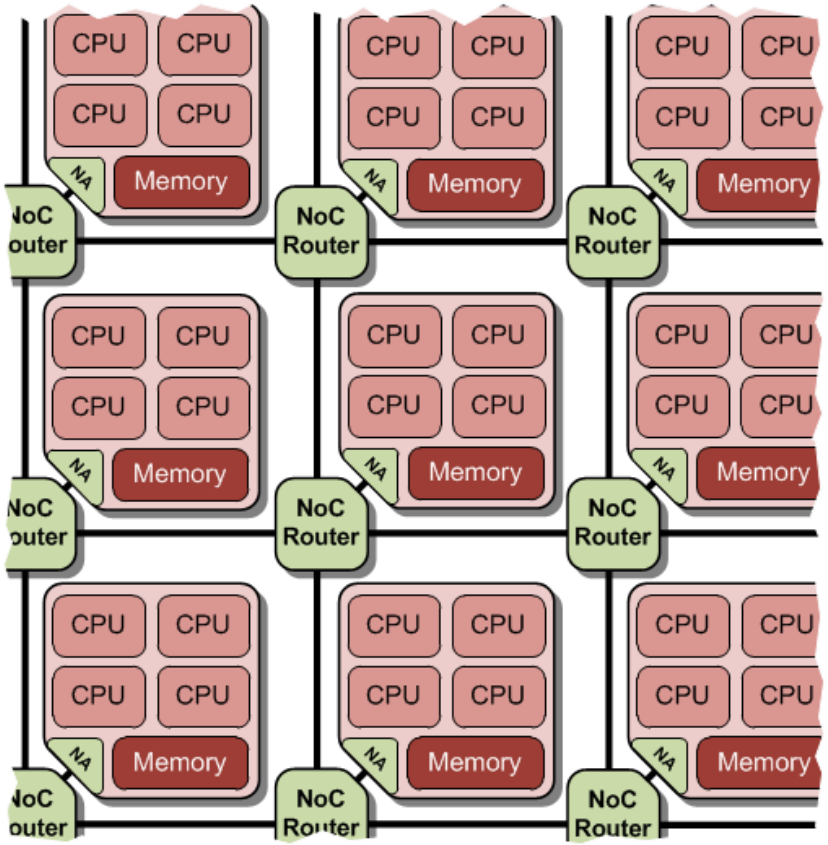


2.5.1 AHB Struktur



2.5.2 Network-on-Chip (NoC)

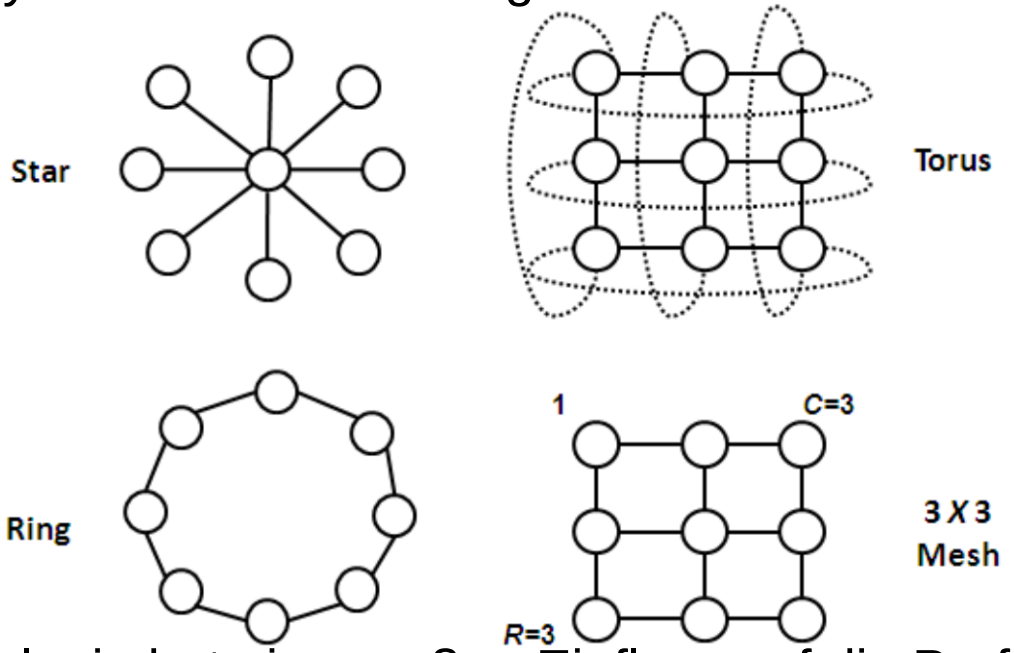
- Networks-on-Chip bieten eine skalierbare Kommunikationsinfrastruktur
- Jede Komponente bringt Kommunikationshardware mit
- Ein Kommunikationsvorgang blockiert immer nur einen Teil der Kommunikationsinfrastruktur
- Im Vergleich zu Bussystemen größere Komplexität
- Komponenten eines NoC:
  - Router
  - Routingentscheidung
  - Packete zwischenspeichern
  - Weiterleitung der Pakete
  - Network-Adapter
  - Protokollübersetzung
  - Zwischenspeichern/umsortieren ankommender Pakete



Quelle: *ITIV, Heißwolf*

2.5.2 NoC – Topologie

- Topologie: Physikalische Anordnung der Netzwerkknoten

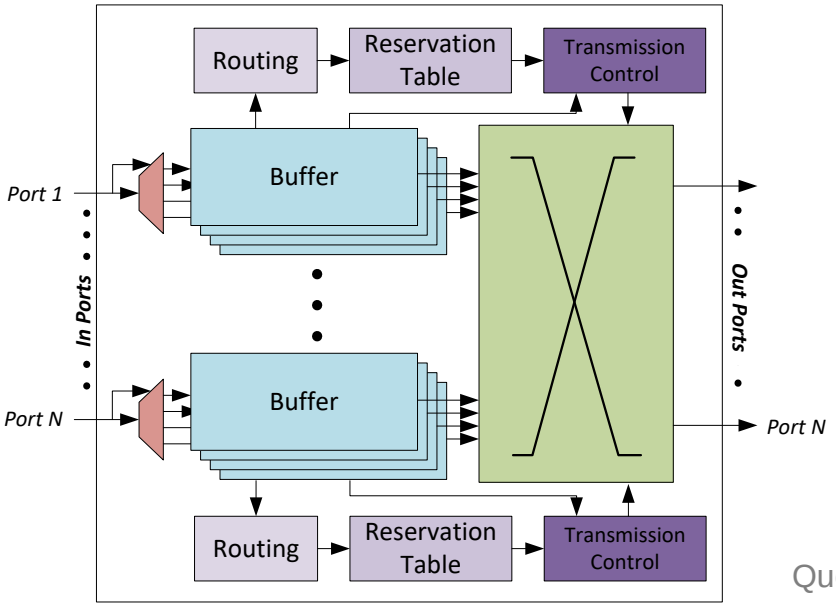


- Wahl der Topologie hat eine großen Einfluss auf die Performanz und den Flächenbedarf:
  - Komplexität des Routers ist abhängig von der Anzahl der Router-Ports
  - Performanz des Netzwerks ist abhängig von der Anzahl an direkten Nachbarn

Quelle: *Rickard Holsmark, Maurizio Palesi, and Shashi Kumar. 2006 Deadlock free routing algorithms for mesh topology NoC systems with regions*

2.5.2 Network-on-Chip – Router

- Aufbau eines Packet-Switching Routers mit virtuellen Kanälen:

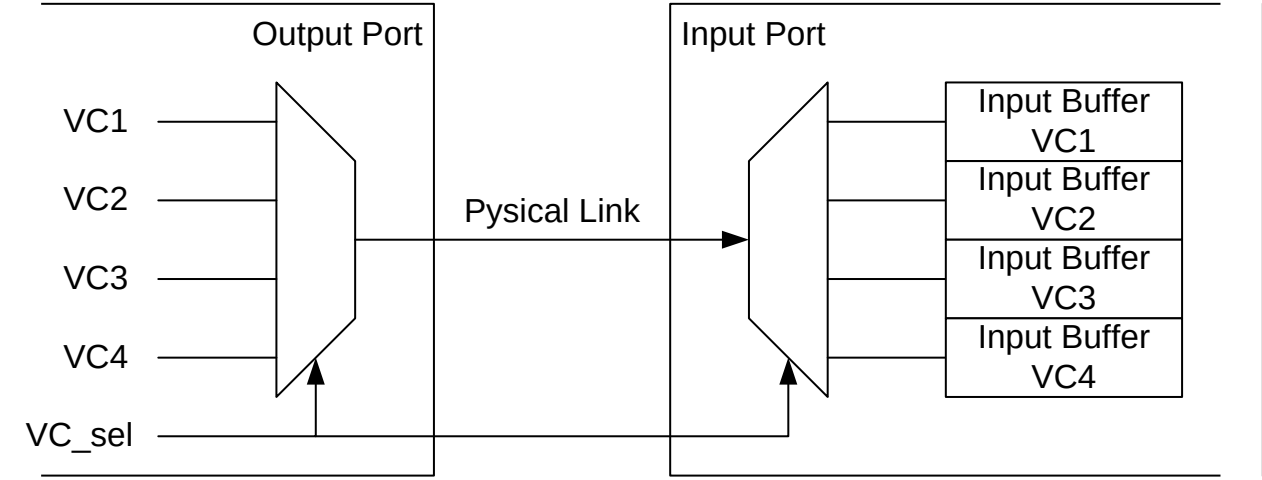


Quelle: *ITIV, Heißwolf*

- Funktionsweise:
  - Eingehende Pakete werden entsprechend der virtuellen Kanäle auf Buffer verteilt.
  - Zieladressen werden in der Routing-Einheit verarbeitet und Reservierungen des Ausgangsports vorgenommen
  - Die „Transmission Control“ regelt, welche Reservierung zum Zuge kommt und leitet die entsprechenden Daten weiter

2.5.2 Network-on-Chip – Virtuelle Kanäle

- Virtuelle Kanäle verbessern die Performanz und ermöglichen mehrere parallele Verbindungen über einen Link:

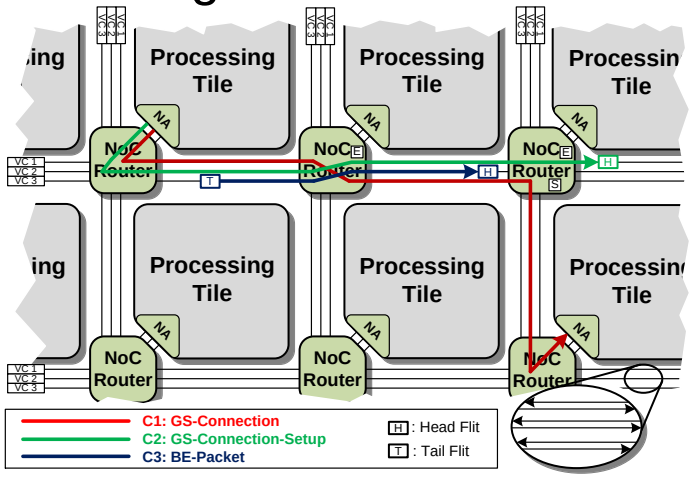


- Prinzip der virtuellen Kanäle:
  - Der physikalische Link wird im Zeitmultiplexing zwischen unterschiedlichen virtuellen Kanälen aufgeteilt
  - Performanz des Netzwerkes wird durch virtuelle Kanäle erhöht, da die Wahrscheinlichkeit von Blockaden reduziert wird
  - Die Anzahl der mögliche Verbindungen bei Verbindungsorientierter Kommunikation kann stark erhöht werden

Quelle: *ITIV, Heißwolf*

2.5.2 Network-on-Chip - Datenübertragung

- Paket- und Verbindungsorientierte Kommunikation in einem NoC:



Quelle: *ITIV, Heißwolf*

- Paketorientierte Kommunikation (Best Effort (BE)):
  - Nutzdaten werden in einem Packet zusammenhängend übertragen
- Verbindungsorientierte Kommunikation (Guaranteed Service (GS))
  - Vor der eigentlichen Übertragung der Nutzdaten wird eine Verbindung von der Quelle zum Ziel etabliert
  - Große Datenmengen können effizienter übertragen werden
  - Garantien für die Latenz und den Durchsatz sind möglich

2.5.3 Skalierung bei mehr als vier Kernen

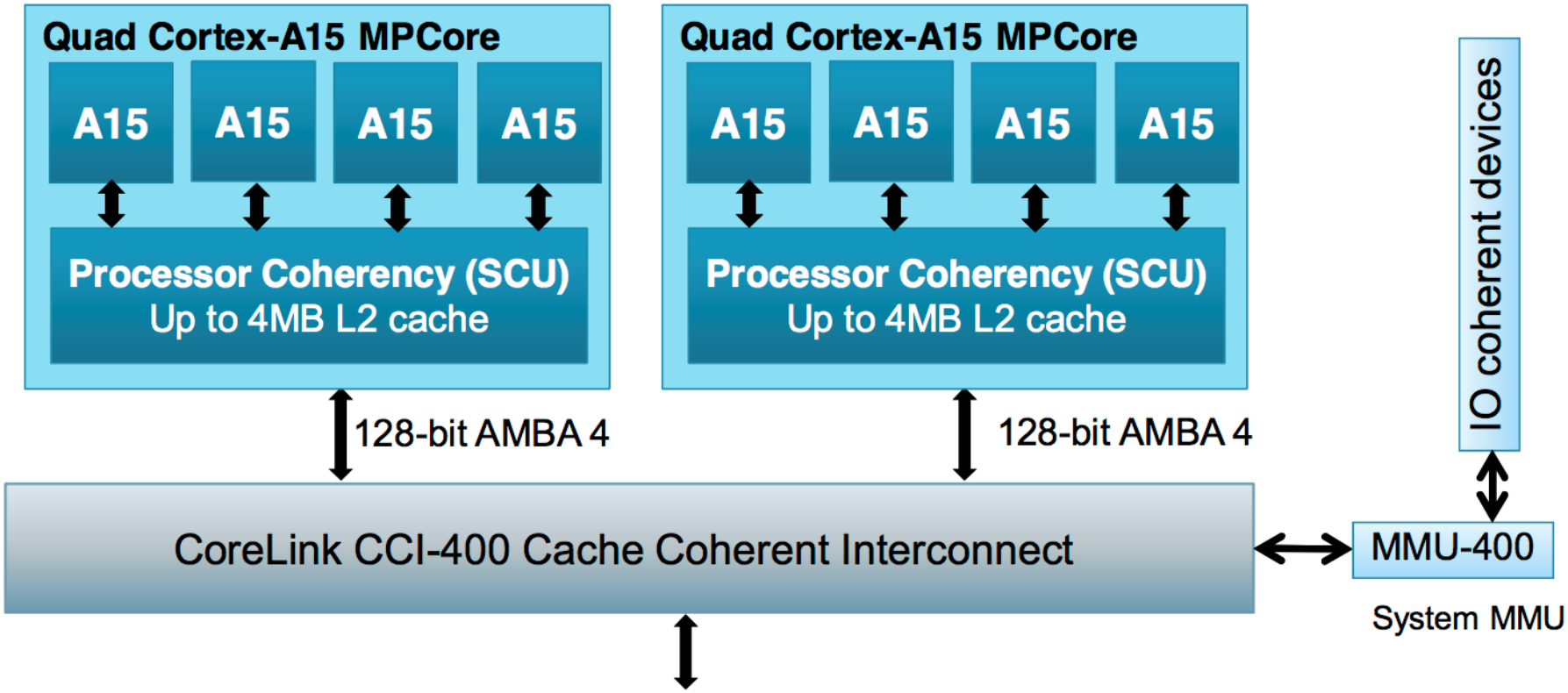
- Einführung von AMBA 4 Kohärenz Erweiterungen
  - Kohärenz, Barrier und Memory Management
- Software Implikationen
  - Hardware verwaltete Kohärenz vereinfacht Software
  - Prozessor verbraucht weniger Zeit zur Cache Verwaltung
- Kohärenz Typen
  - I/O Kohärenz
    - Geräte snoopen in Prozessor Cache
      (aber Prozessoren snoopen nicht ins Gerät)
  - Volle Cache Kohärenz
    - Cache snooping in beide Richtungen

2.5.3 Beispiel: ARM Cortex-A15

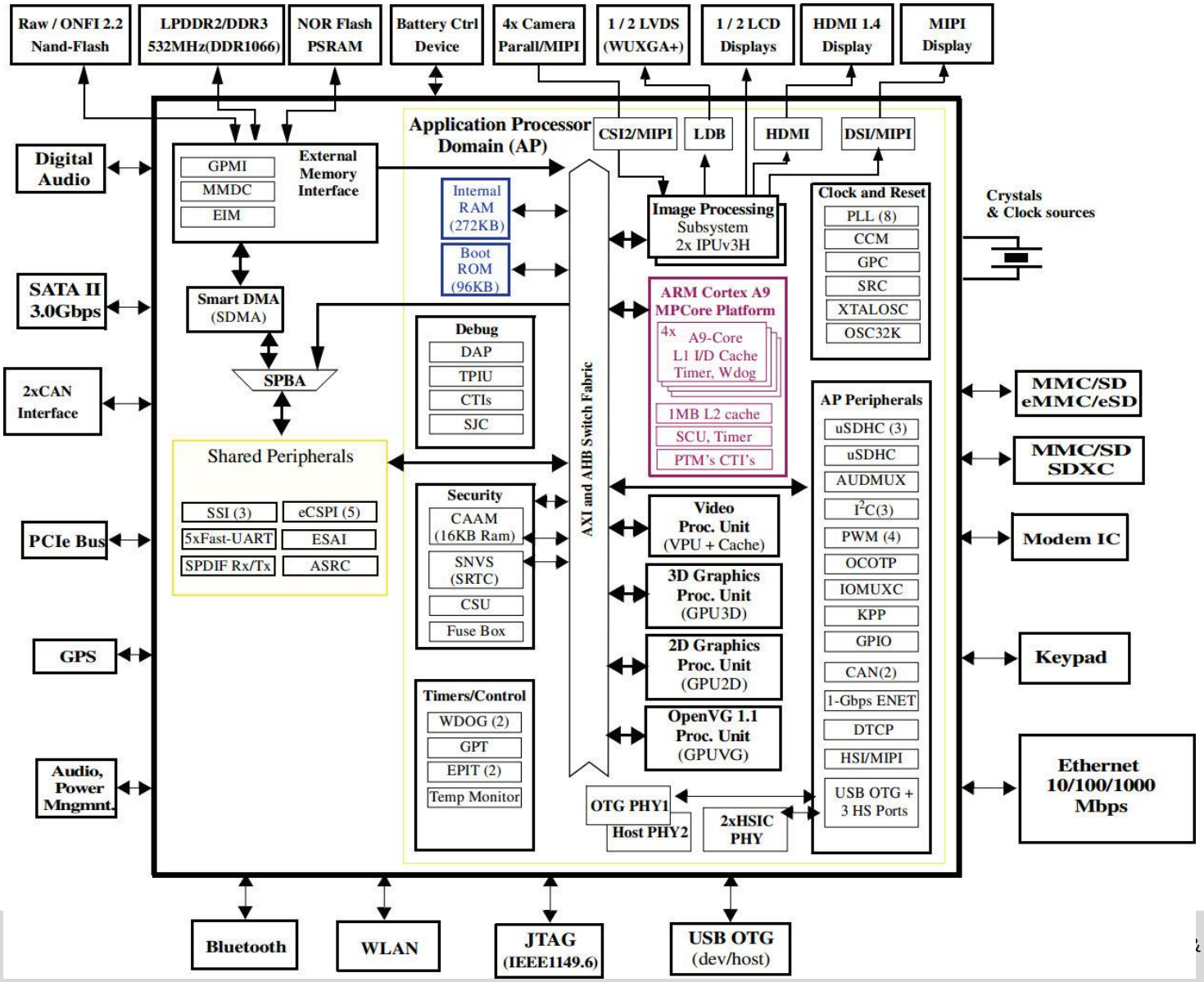
- Cortex-A class Multi-Prozessor
  - 40bit physikalische Addressierung (1TB)
  - Volle Hardware Virtualisierung
  - AMBA 4 System Kohärenz
  - ECC und Paritäts-Sicherung für alle SRAMs
- Fortschrittliches Power Management
  - Fein-granulares Abschalten der Pipeline
  - Aggressive L2 Power Reduzierungsfähigkeiten
  - Schnelles state save und restore
- Signifikante Performanz Verbesserung
  - Verbesserte single-thread und MP Performanz
- Integrierter L2 Cache mit SCU Funktionalität
- 128-bit AMBA 4 Interface mit Kohärenz Erweiterungen

2.5.3 ARM Cortex-A15 System Skalierbarkeit

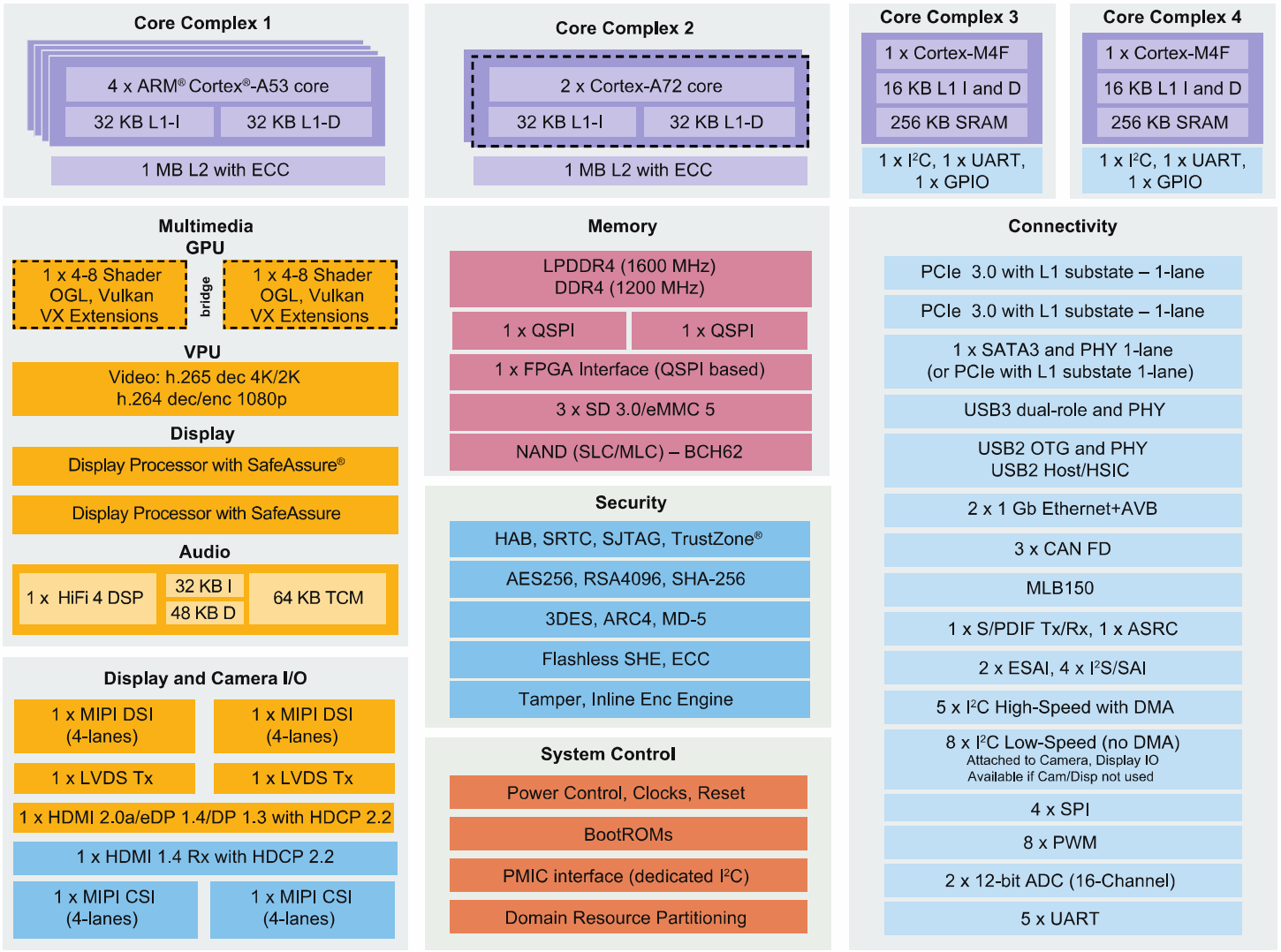
- Einführung von CCI-400 Cache Coherent Interconnect
  - Prozessor zu Prozessor Kohärenz und I/O Kohärenz
  - Speicher und Synchronizations Barrieren
  - TLB und Cache Versorgung



2.5.3 Beispiel NXP i.MX6

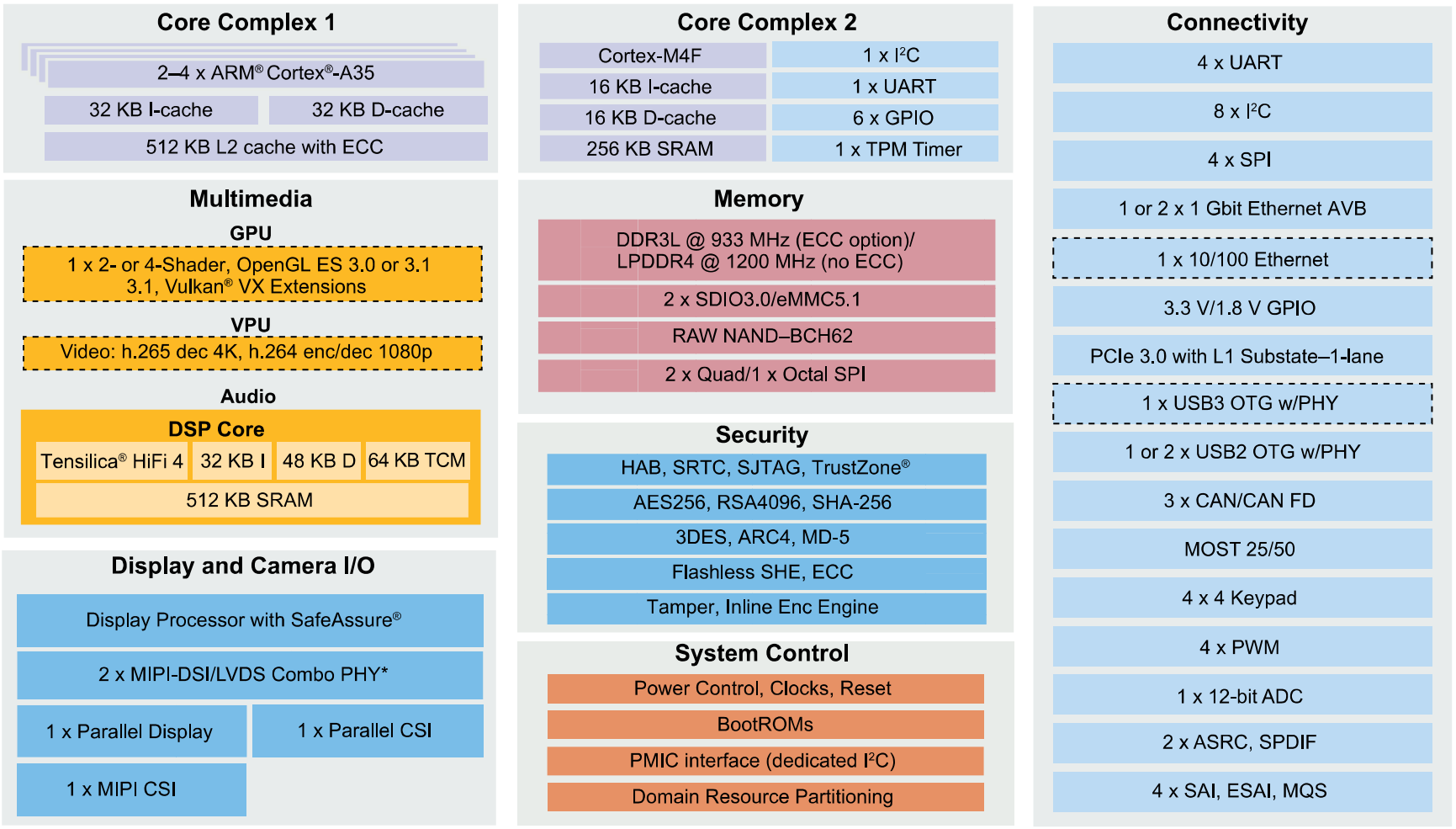


2.5.3 Beispiel NXP i.MX8



Source: nxp.com

2.5.3 Beispiel NXP i.MX8



Source: nxp.com

2.5.3 Beispiel NXP i.MX8

i.MX 8X FAMILY— DIFFERENTIATED FEATURES

Feature	i.MX 8DualXPlus/ i.MX 8QuadXPlus	i.MX 8DualX
ARM® Core	2 x Cortex-A35 (i.MX 8DualXPlus) 4 x Cortex-A35 (i.MX 8QuadXPlus)	2 x Cortex-A35
ARM Core	1 x Cortex-M4F	1 x Cortex-M4F
DSP Core	Tensilica® HiFi 4 DSP	Tensilica HiFi 4 DSP
DRAM	32-bit DDR3L (ECC option)/ LPDDR4 (no ECC)	16-bit DDR3L (ECC option)/ LPDDR4 (no ECC)
GPU	1 x GC7000Lite	1 x GC7000UltraLite
VPU	4K h.265 dec, 1080p h.264 enc/dec	1080p h.264 enc/dec
Ethernet	2 x Gigabit with AVB	1 x Gigabit with AVB 1 x 10/100
USB with PHY	1 x USB 3.0 (can be used as USB 2.0) 1 x USB 2.0	2 x USB 2.0

i.MX 8 FAMILY—DIFFERENTIATED FEATURES

Feature	i.MX 8QuadMax	i.MX 8QuadPlus	i.MX 8Quad
ARM® Core	2 x ARM Cortex®-A72	1 x Cortex-A72	–
ARM Core	4 x Cortex-A53	4 x Cortex-A53	4 x Cortex-A53
ARM Core	2 x Cortex-M4F	2 x Cortex-M4F	2 x Cortex-M4F
DSP Core	HiFi 4 DSP	HiFi 4 DSP	HiFi 4 DSP
GPU	2 x GC7000XSVX	2 x GC7000Lite/XSVX	2 x GC7000Lite/XSVX
PCIe 3.0	1 x PCIe (2-lane)*	1 x PCIe (1-lane)	1 x PCIe (1-lane)

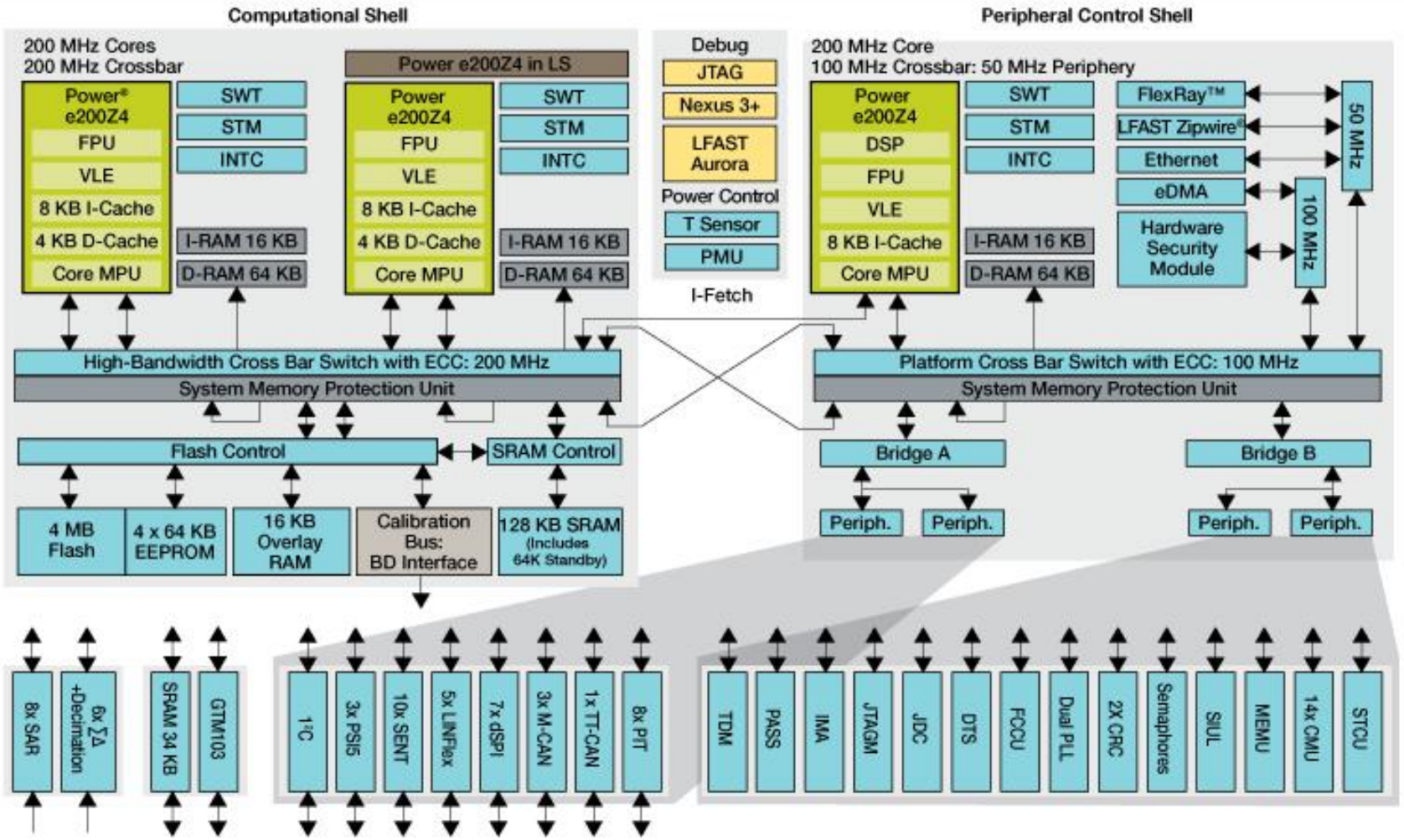
i.MX 8 FAMILY—COMMON FEATURES

Feature	Description	Feature	Description
DRAM	64-bit LPDDR4/ DDR4	QuadSPI	2 x QuadSPI (1 x OctoSPI)
VPU	4K h.265 dec, HD h.264 enc	USB with PHY	1 x USB 3.0, 2 x USB 2.0
Display controller	2 x DCs with WARP and failover	SPDIF Tx/Rx	1 x
MIPI DSI	2 x 4-lane MIPI DSI	SD and eMMC	3 x SD 3.0/eMMC 5.0
MIPI CSI	2 x 4-lane MIPI CSI	NAND	1 x – BCH62
LVDS	2 x LVDS	FPGA Interface	Yes - 4 x data lane, 1 x Clock
HDMI, eDP, DP Tx	1 x HDMI 2.0a/ eDP 1.4/DP 1.3 HDCP 2.2	I²C	5 x I²C (high speed) + 8 x I²C (low speed)
HDMI Rx	1 x HDMI 1.4 Rx HDCP 2.2	SPI	4 x SPI
SATA 3.0	1 x SATA 3.0 (1-lane) or PCIe (1-lane)*	Audio Interfaces	2 x ESAI, 5 x I²S/SAI
Security	HAB, DPA, enc/ dec, flashless SHE inline DDR encryption, 4 tamper pins	Keypad	1 x
CAN	3 x CAN FD	MPEG-2 T/S	2 x MPEG-2 T/S
MLB	1 x MLB 150/ MLB25	ADC	2 x 12-bit (16 channels each)
Ethernet	2 x Gigabit Ethernet with AVB	UART	5 x UART 1 x UART per ARM® Cortex®-M4F

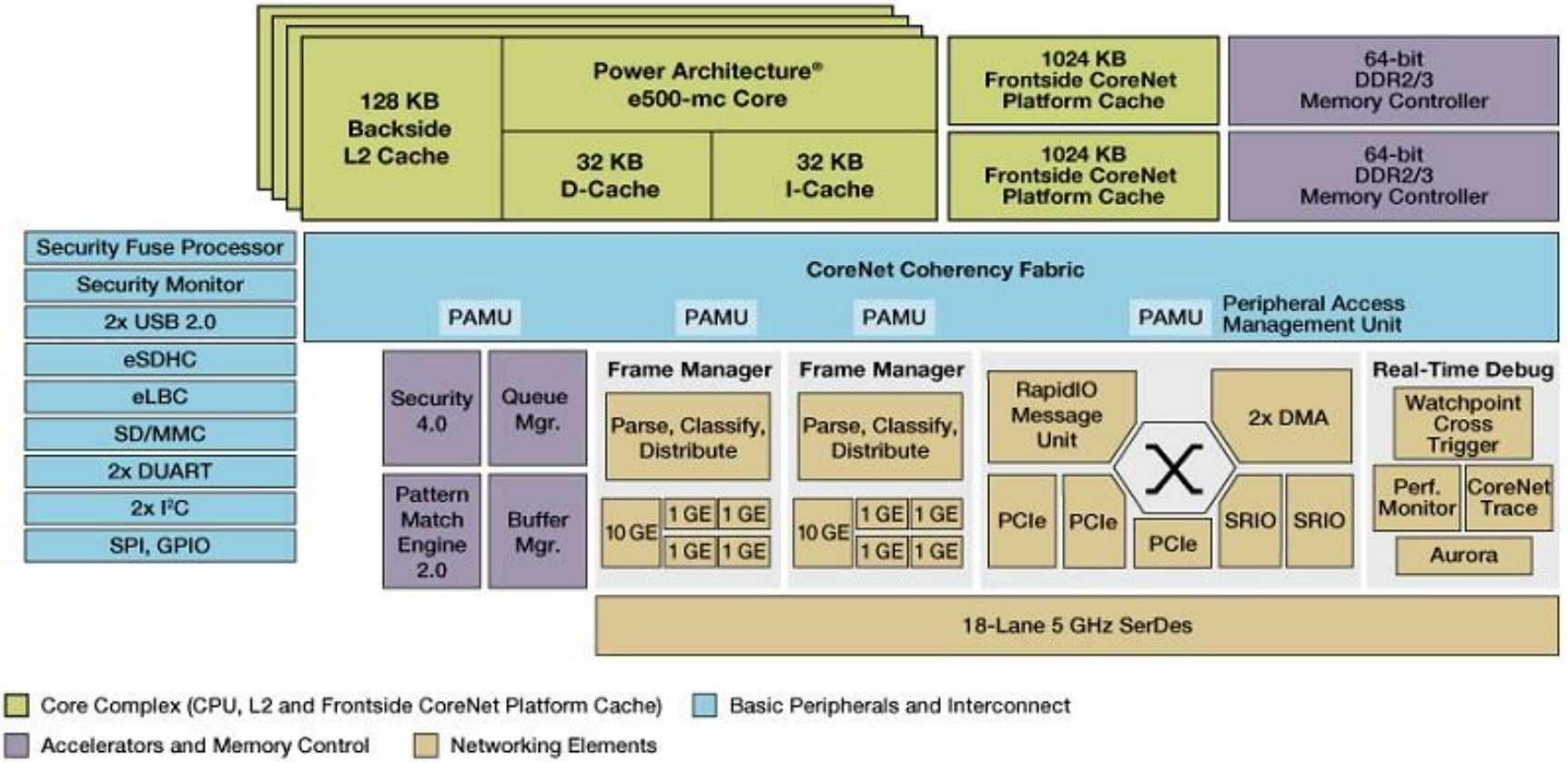
Source: nxp.com

2.5.3 Beispiel: Controller für Motorsteuergeräte

Qorivva MPC5746M Block Diagram



2.5.3 Freescale P4080 – Network Processor

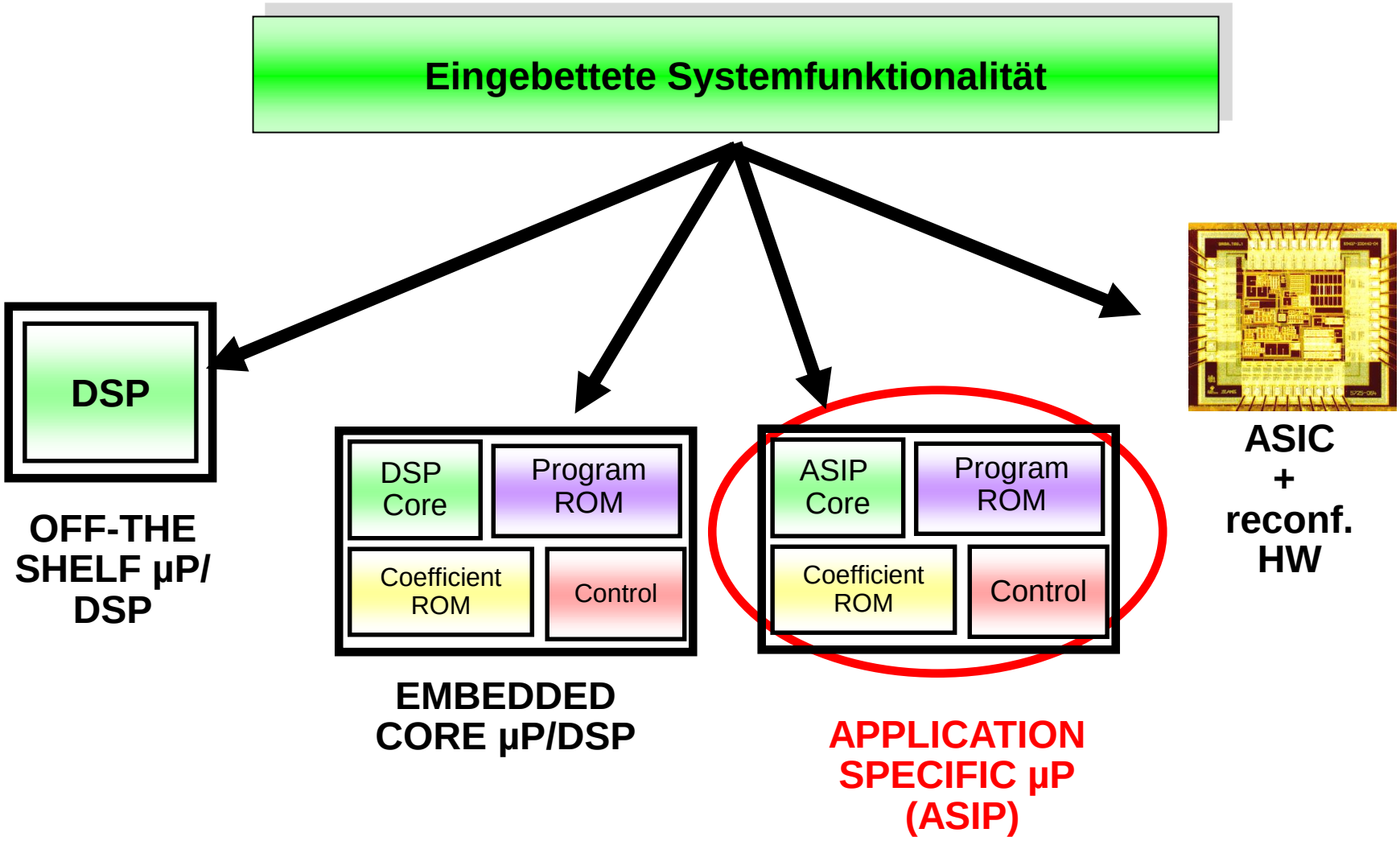


2.5.3 Herausforderung von Many-Core

- Kleinere Prozessoren können nicht das gleiche Single-Thread Programm in der gleichen Zeit wie ein High-Performance Prozessor ausführen
  - Vorhandene Applikationen können nicht effizient ausgeführt werden
- Viele Threads können nicht einfach auf einfachere, kleinere Tasks runtergebrochen werden, um vom Multiprocessing auf den kleineren Prozessoren profitieren zu können
- ➔ Software Entwicklungsherausforderung

- Welche Arten von Kommunikationsprotokollen gibt es?
- Aus welchen Komponenten besteht ein Bussystem? Wie funktioniert es?
- Was sind Limitierungen von Multicore Architekturen?
- Wie kann eine höhere Performanz erreicht werden?
- Was sind Herausforderungen von Manycore Systemen?
- Was ist ein Network-On-Chip? Wie ist es aufgebaut? Wie funktioniert es?

2.6 Implementierungsalternativen hochperformanter Algorithmen



2.6 Application-Specific Instruction Set Processors (ASIP)

- Anwendungs-orientierte Spezialisierung
  - des Instruktionssatzes für eingeschränkte Klasse von Anwendungen
    - Bsp.: Operatorverkettung
      - Multiply-Accummulate -> DSP
      - Filter-/Signalverarbeitungsoperationen -> Audio-/Video-Prozessoren
  - der Funktionseinheiten
    - Bsp.: Pixel-Operationen, 1/sqrt(x)
  - der Speicherarchitektur
    - Bsp.: mehrere Speicherbänke mit parallelem Zugriff

- Vorteile gegenüber anderen Prozessoren
  - höhere Performanz
  - niedrigere Kosten (kleinere Chipfläche, weniger Pins)
  - kleinere Codegröße -> mächtigere Instruktionen
  - geringere Leistungsaufnahme

2.6 ASIP: Klassifizierung nach Eigenschaften (I)

- Datentypen
  - Festkomma- oder Gleitkomma-Arithmetik, spezielle Bitbreiten
- Codetyp
  - Mikrocode, alle Instruktionen benötigen einen Maschinenzyklus
  - Makrocode, Instruktion benötigen mehrere Zyklen -> Instruktionspipelining
- Speicherorganisation
  - Load/Store-Architekturen oder Mem/Register-Architekturen
  - ASIPs besitzen häufig keinen Cache
    - Speicher (RAM, ROM, Register) werden meistens on-Chip realisiert
  - Registerstruktur entweder heterogen oder homogen
    - homogen: Register universell für alle Operationen einsetzbar
    - heterogen: verschiedene Registersätze, je nach Operation

2.6.1 ASIP: Klassifizierung nach Eigenschaften (II)

- Instruktionsformat
  - einzeln codiert
    - Felder in Abhängigkeit von Opcode interpretiert, oder
  - orthogonal codiert
    - Bsp. VLIW: Instruktionsteile voneinander unabhängig
      - erlaubt Ansteuerung von unabhängigen Funktionseinheiten

- Besonderheiten
  - spezielle arithmetische Einheiten,
  - spezielle Datentypen,
  - besondere Adressierungsarten,
  - HW-Unterstützung von Schleifenkonstrukten,
  - ...

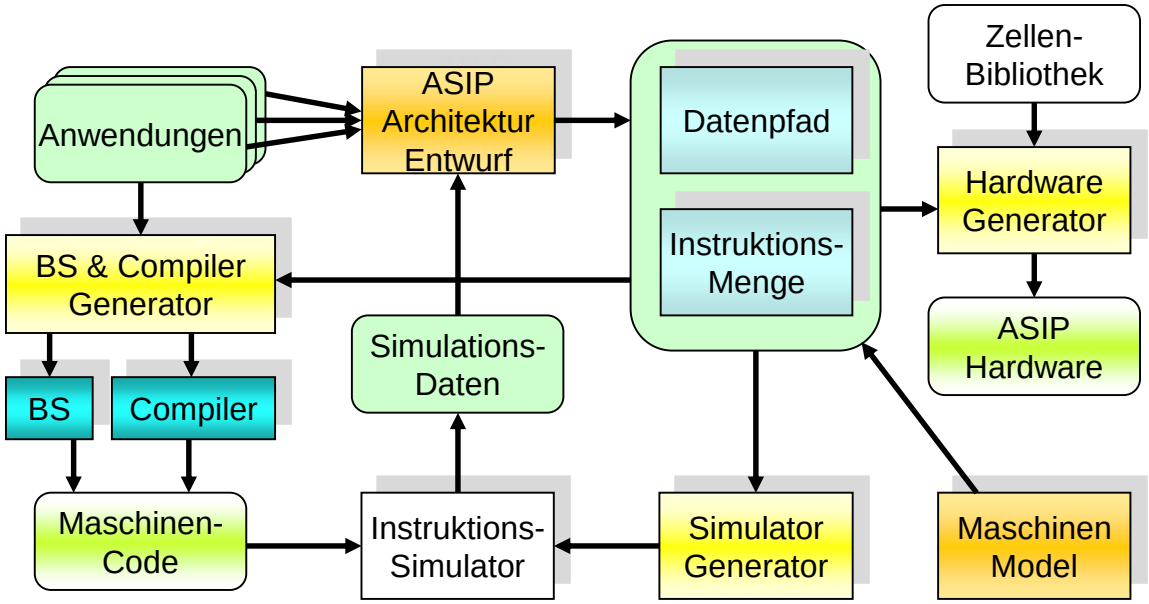
2.6 ASIP: Beispiele

- Da ASIPs sehr anwendungsspezifisch sind und oft speziell für einzelne Anwendungen entwickelt worden sind, sind diese i.a. nicht auf dem kommerziellen Markt erhältlich
- Beispiel: Network Processors -> oft firmeninterne Entwicklungen

Company	Product	RISC based	Task Specific Processor based	ASIC
Level One	*IXP1200*	✔		
IBM	*PNP*	✔		
MMC	*nP*	✔		
Maker	*MXT*	✔		
Sitera	*PRISM IQ1200*	✔		
EZChip	*NP-1*	✔		
C-Port	*C-5 DCP*	✔		
Agere	*PayloadPlus*		✔	
Fast-chip	*PolicyEdge*		✔	
Hi-fn	*7711, 7751*		✔	
Xaqti	*TeraPower-CL*		✔	
Broadcom	*StrataSwitch*		✔	
Solidum	*PAX.port 1100*		✔	
Netlogic	*Policy, CIDR*		✔	
Switchcore	*CXE*		✔	
Entridia	*Opera*			✔

2.6 ASIP: angestrebter Entwurfsablauf (I)

- Erzeugung der ASIP-Hardware
  - Wenn simulierte ASIP-Hardware allen funktionalen und zeitlichen Anforderungen genügt
    - Hardwarerealisierung wird erzeugt
  - Hardwaregeneratoren verwenden diverse Bibliotheken mit Prozessorkernen, Bus-Interfaces, Speichern, ALUs, Arithmetik in Pipelineausführung, Spezialoperatoren, Zählern, Registern usw.



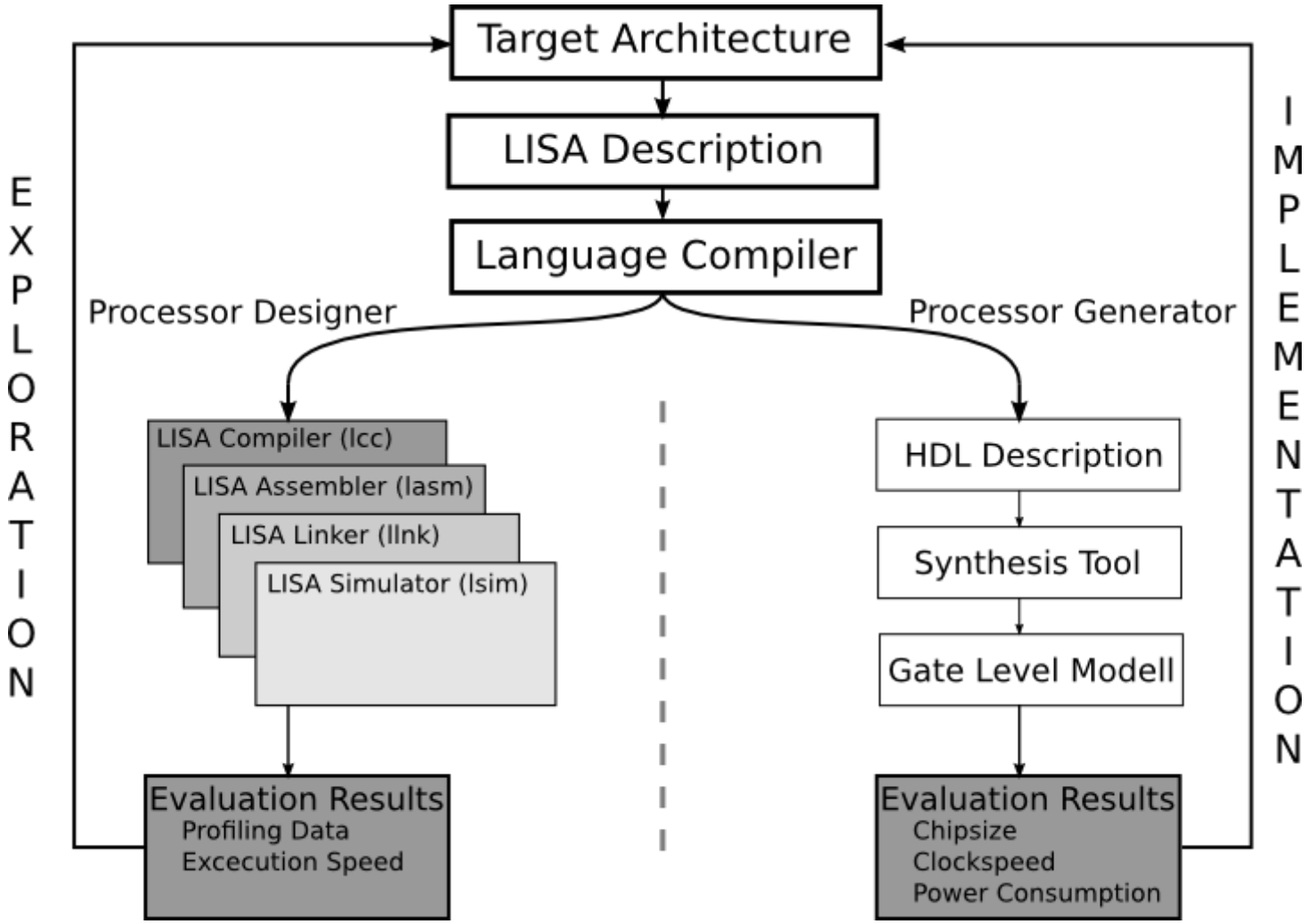
2.6 ASIP: angestrebter Entwurfsablauf (II)

- Architektur-Entwurf
  - Identifizierung neuer Instruktionssätze / Datenpfade aus Obermenge von Instruktionen für Standard-Arithmetik, Speicher- und Kontrollflußinstruktionen
  - Besondere Instruktionen verfügbar
    - z.B. digitale Filterung oder Viterbi-Dekodierung
  - Anpassung des Datenpfads + Instruktionen (+ Speicherstruktur)
    - erfolgt aus Informationen, die aus Anwendungs-Benchmarks und zu erwartenden Daten abgeleitet werden können (Nebenläufigkeit!)
    - Nebenbedingungen wie Ausführungsgeschwindigkeit, Fläche und Leistungsaufnahme sind zu berücksichtigen
  - Ein abstraktes Maschinenmodell (ggf. mit Pipelining) dient dabei als Entscheidungsgrundlage für die Hardwarearchitektur

2.6 ASIP: angestrebter Entwurfsablauf (III)

- Ziel: Erzeugung des Betriebssystems und Compilers für ASIPs
  - Compiler erzeugt den Maschinencode für abgeleitete ASIP-Architektur
    - generiert aus einer Hochsprachenbeschreibung z.B. C
  - Codeerzeugung ist auf andere Zielarchitekturen retargierbar
    - die in HDL gegeben sind (Instruktionsmuster, Ressourcen, Verbindungen)
  - Flexible Anpassung der Codeerzeugung auf neue / geänderte Zielarchitekturen
    - für den ASIP-Entwurf sehr vorteilhaft

2.6 ASIP Beispiel: Synopsys Processor Designer

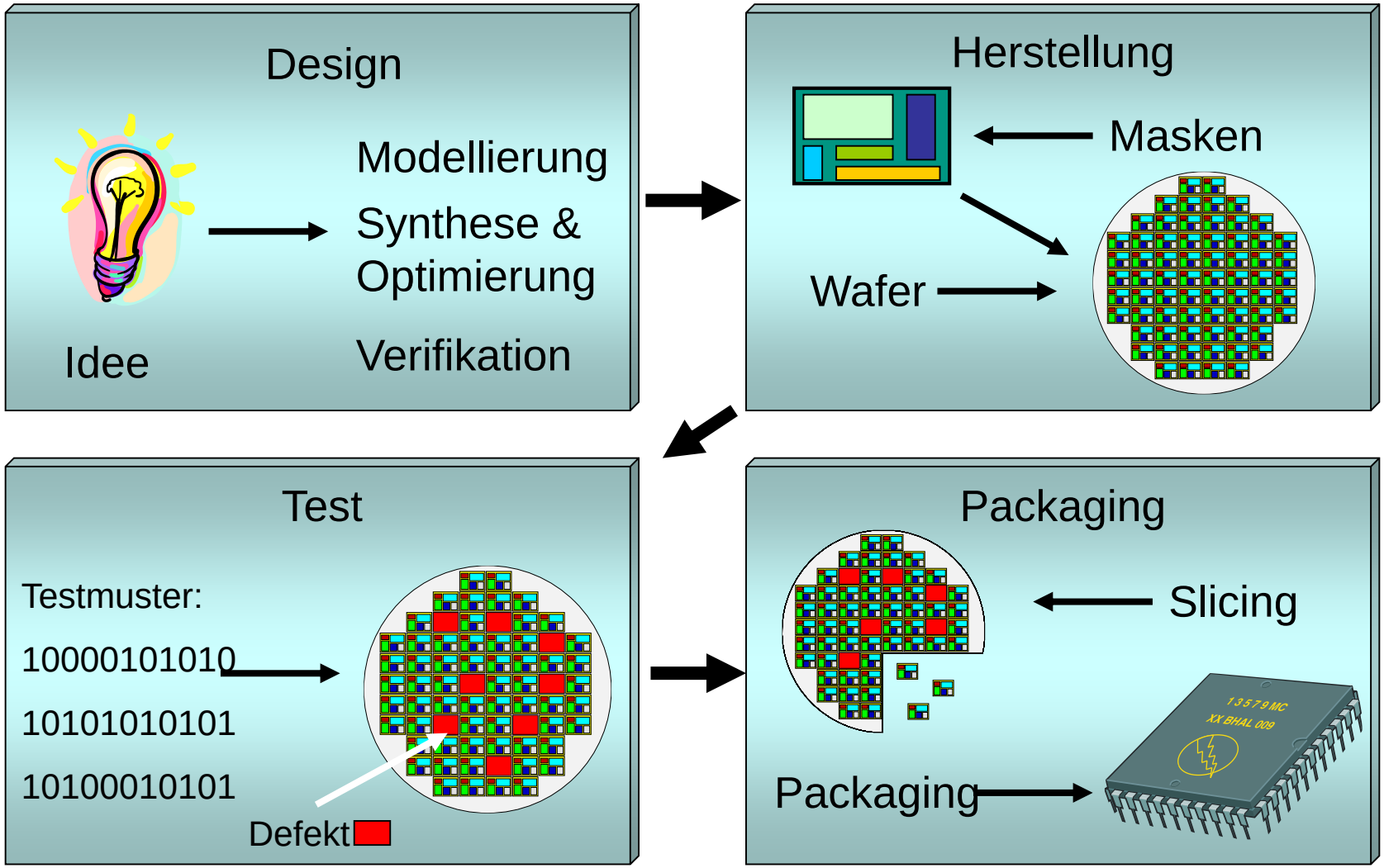


- Was ist ein ASIP?
- Was sind dessen besondere Eigenschaften?
- Welche Vor- oder Nachteile haben ASIPs gegenüber anderen Prozessoren?
- Wie wird ein ASIP entworfen?

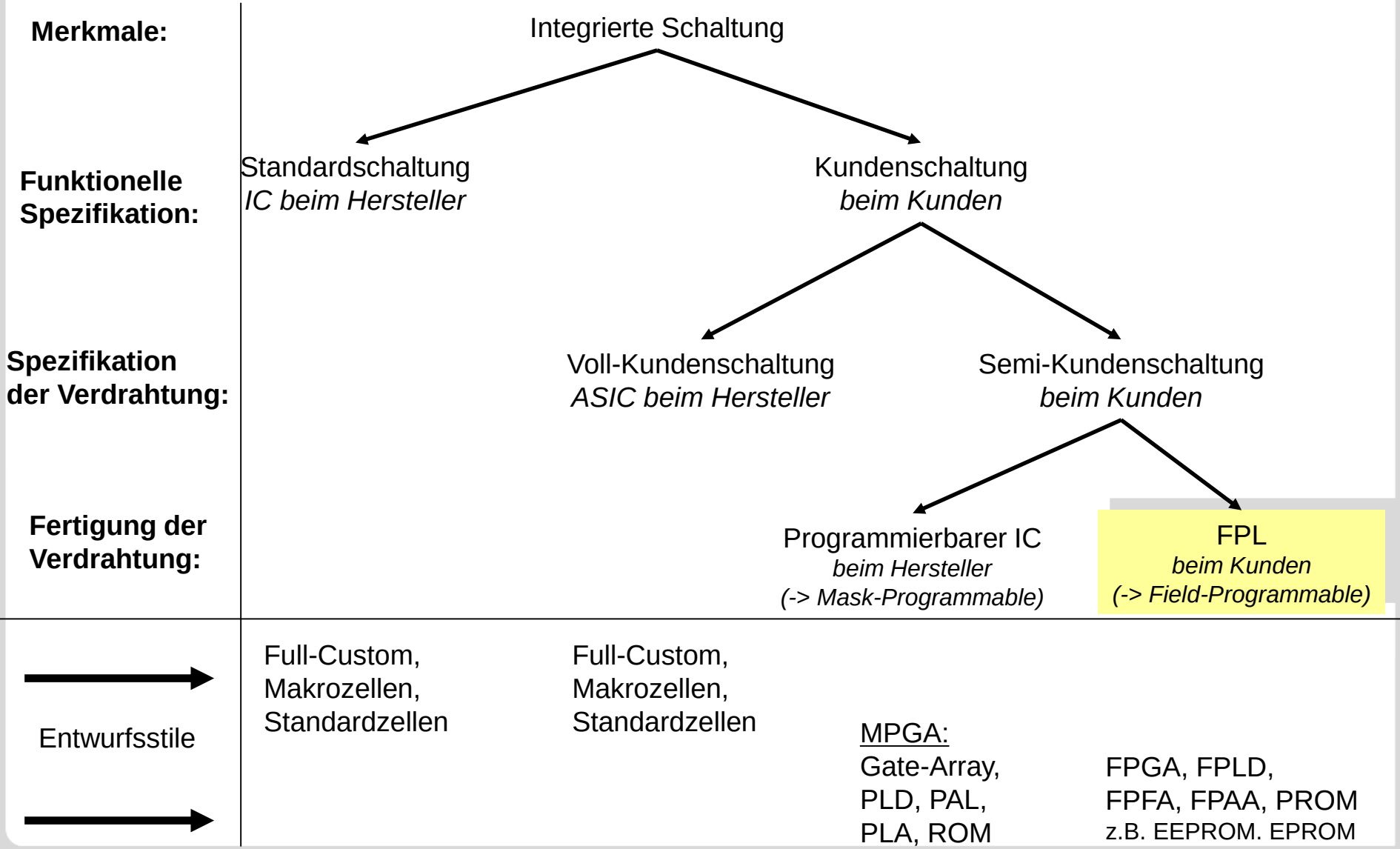
Inhalt

- 2.4 Spezialprozessoren
  - 2.4.1 Mikrocontroller (µC)
  - 2.4.2 Digitale Signalprozessoren (DSPs)
  - 2.4.3 Grafik Prozessoren (GPU)
- 2.5 Bus, Network-on-Chip (NoC) & Multicore
- 2.6 Anwendungs-spezifische Instruktionssatz Prozessoren (ASIP)
- **2.7 Field Programmable Gate Arrays (FPGA)**
- 2.8 System-on-Chip (SoC)

2.7.1 Entwurfsstile: Phasen bei Integrierten Schaltungen



2.7.1 Entwurfsstile für ICs



2.7.1 Vergleich der wichtigsten Entwurfsstile

	Custom	Cell-based	MPGA	FPGA
Dichte	Sehr hoch	Hoch	Hoch	Medium bis niedrig
Performance	Sehr hoch	Hoch	Hoch	Medium bis niedrig
Entwurfszeit	Sehr lange	Kurz	Kurz	Sehr kurz
Fabrikationszeit	Medium	Medium	Kurz	Sehr kurz
Kosten bei geringer Stückzahl	Sehr hoch	Hoch	Hoch	Niedrig
Kosten bei hoher Stückzahl	Niedrig	Niedrig	Niedrig	Hoch

MPGA: Mask Programmable Gate Array
FPGA: Field Programmable Gate Array

2.7.1 Evolution der Implementierungstechnologien

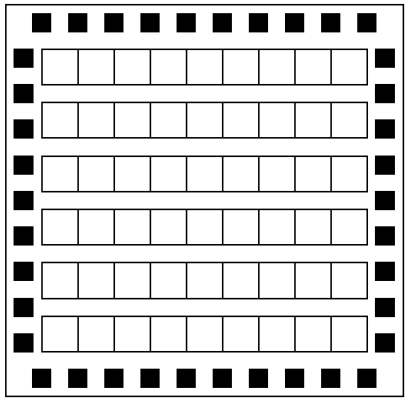
- Diskrete Bauteile: Relais, Transistoren (1940‘er -50‘er)
- Diskrete Gatter-Logik (1950‘er-60‘er)
- Integrierte Schaltungen (1960‘er-70‘er)
  - z.B. TTL (Transistor-Transistor-Logik): Datenbuch für Hunderte verschiedener Blöcke
  - Abbildung der Schaltung auf die TTL-Blöcke aus dem Katalog
- Gate-Arrays (IBM 1970‘er)
  - “Kundenspezifische” integrierte Schaltungs-Chips
  - Design unter Verwendung einer Bibliothek (TTL-Library)
  - Transistoren sind bereits auf dem Chip integriert
  - Place & Route Software personalisiert den Chip automatisch
    + Große Schaltungen auf einem Chip
    + Automatische Design-Tools (kein langwieriges Custom-Layout)
    - Nur gut bei sehr großen Stückzahlen

Trend in Richtung höherer Integration

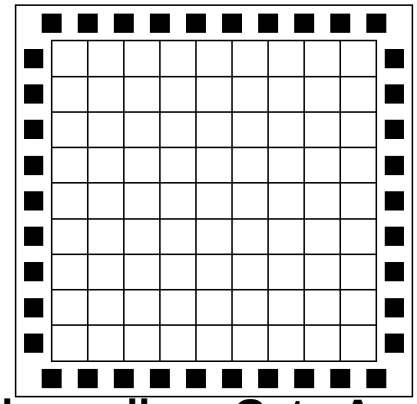
2.7.2 Gate-Array Technologie

2 Typen von Gate-Arrays:

- Einfache Logik Gatter
  - Gebrauch von Transistoren für kombinatorische und sequentielle Logik
    - Pre-Diffused Technologie, vor der Personalisierung
- Personalisierung
  - „Verdrahtung“ zur Verbindung der Ein-/Ausgänge der vorgefertigten Komponenten (Transistoren / Gatter)
- I/O-Blöcke
  - Spezielle Peripherieblöcke für externe Verbindungen
- Hinzufügen von Verdrahtung, um Verbindungen zu schaffen
  - Während der Chipherstellung
    - -> Kundenspezifische “Masken-Programmierung”
  - Erzeugung beliebiger Schaltungen durch anwendungsspezifische Metallisierung



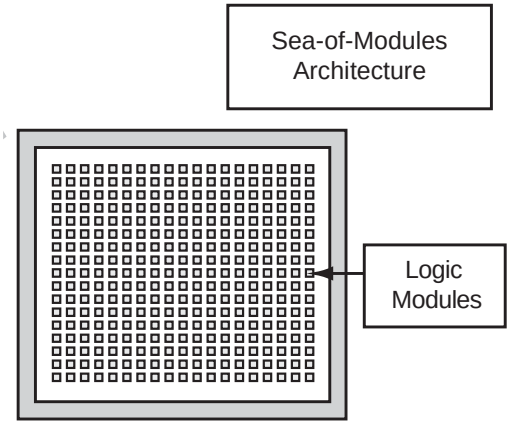
Channeled Gate-Array

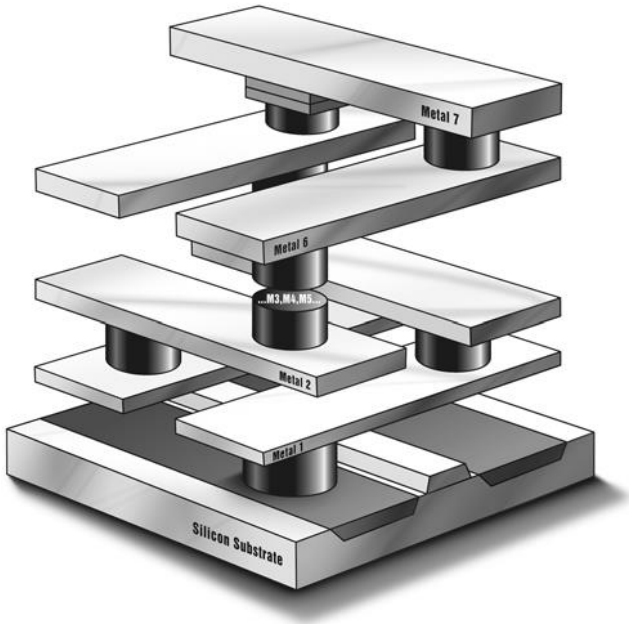


Channelless Gate-Array (*Sea-of-Gates*)

2.7.2 Einmal Programmierbare Logik – Fuse und Anti-Fuse

- „Schmelzsicherung“ unterbricht (-> Fuse) oder erzeugt (-> Anti-Fuse) Verbindung zwischen zwei Verdrahtungslagen
- **Einmalige** Programmierbarkeit (Tests vor der Programmierung?)
- Erlaubt sehr hohe Integrationsdichte
- Weniger empfindlich gegenüber ionisierender Strahlung (z.B. Weltraumeinsatz)



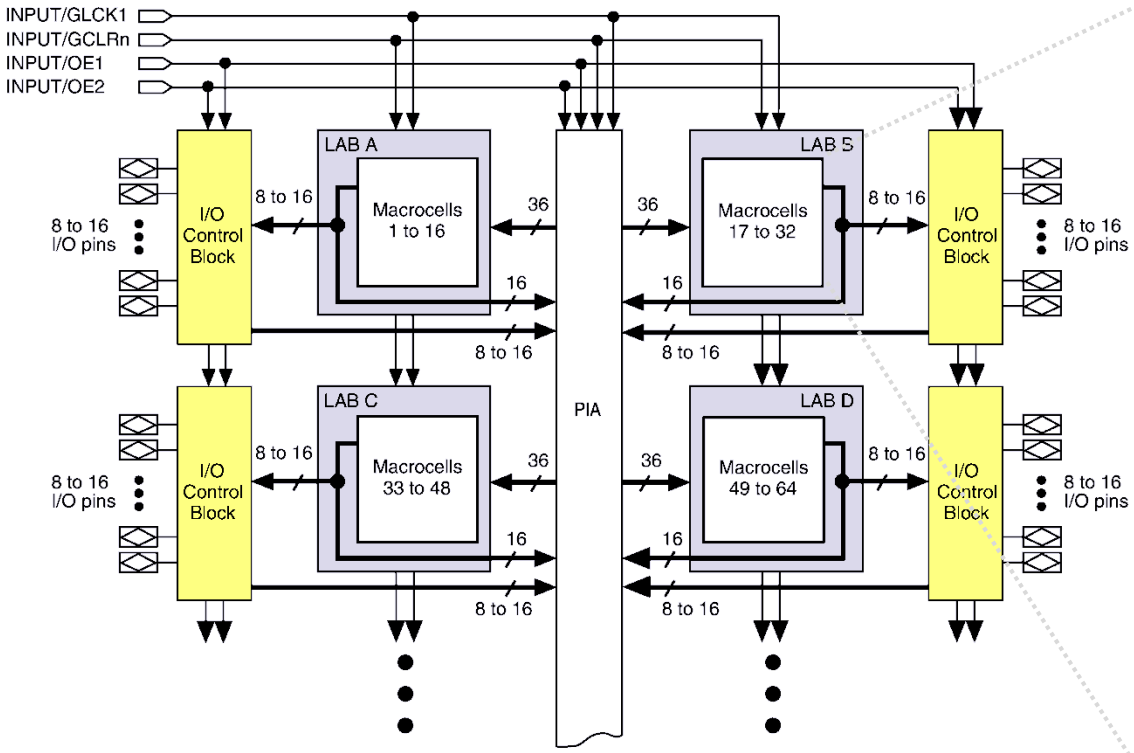


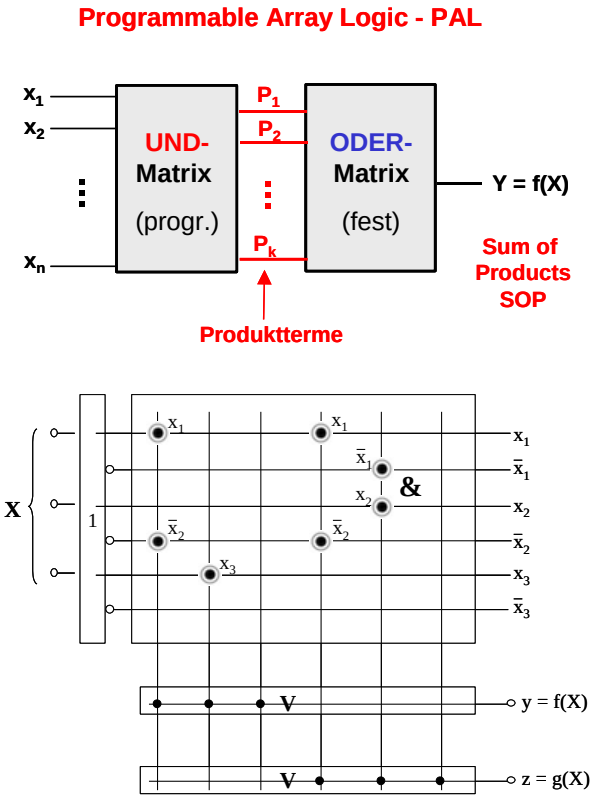
Source: microsemi.com

2.7.2 Mehrfach Programmierbare Logik – RAM-basiert

- Gesetzte Bits in Look-up-Tables (LUTs) realisieren Logikfunktionen
- Gesetzte Bits in einer FlipFlop-Zelle (FF) kontrollieren Schalter, die unterschiedliche Leitungen trennen / verbinden (-> Verdrahtung)
- Können mehrfach (re-) programmiert werden (auch teilweise dynamisch, während des Betriebs)
  - Dynamische (partielle) Rekonfiguration
- Geringe Integrationsdichte, strahlungsempfindlich (spezielle Typen verfügbar.)

2.7.3 Feinkörnig Rekonfigurierbare Hardware: Altera MAX 7000 CPLD

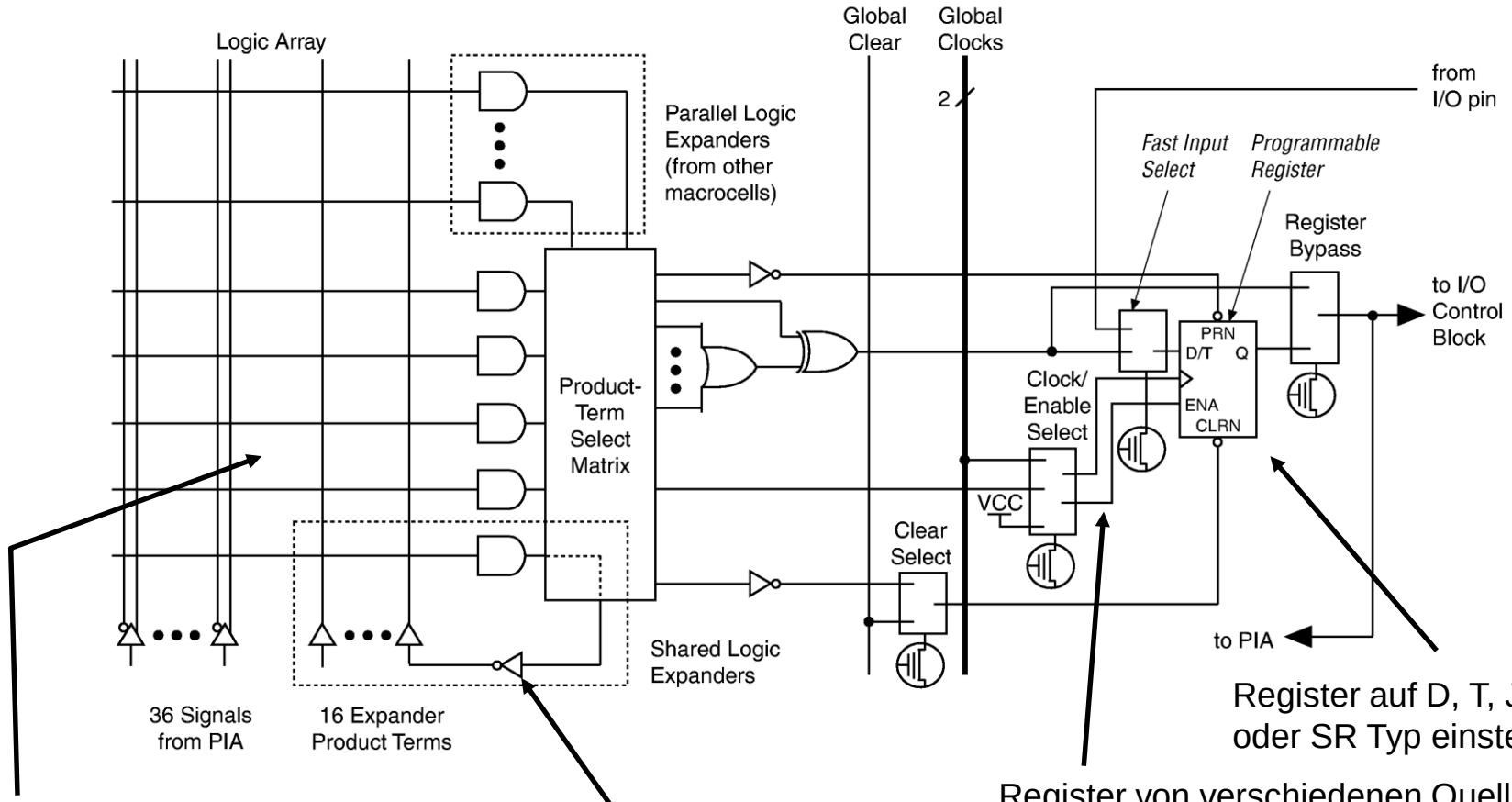




- Logic Array Blocks (LAB), I/O Blocks, und Programmable Interconnect Array (PIA)
- Programmierbarkeit basiert auf EEPROM Floating-Gate (Flash) Technology (nicht-flüchtig)
- 16 Makrozellen in jedem LAB
- Flip-Flops + kombinatorische Basis-Logik
- Programmierbares Interconnect-Array enthält Busse, welche die Eingänge/Ausgänge der Makrozellen verbinden.
- I/O Control-Blöcke verbinden alle benachbarten LABs sowie extern zu den Pins.

2.7.3 Altera programmierbare Makrozelle

- Ein LAB besteht bei der MAX 7000 aus 16 Makrozellen



Logic Array erlaubt 5 Produktterme je Makrozelle

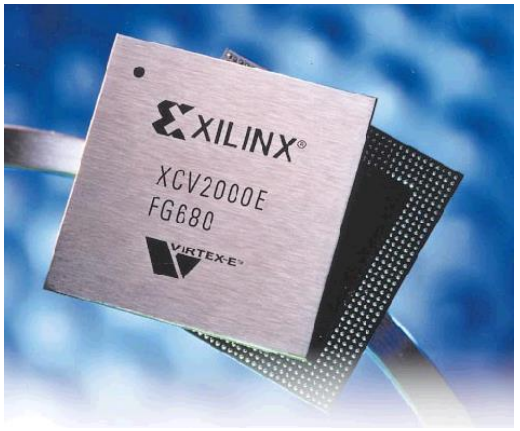
Rückkopplung auf den Eingang, auch von den anderen Makrozellen

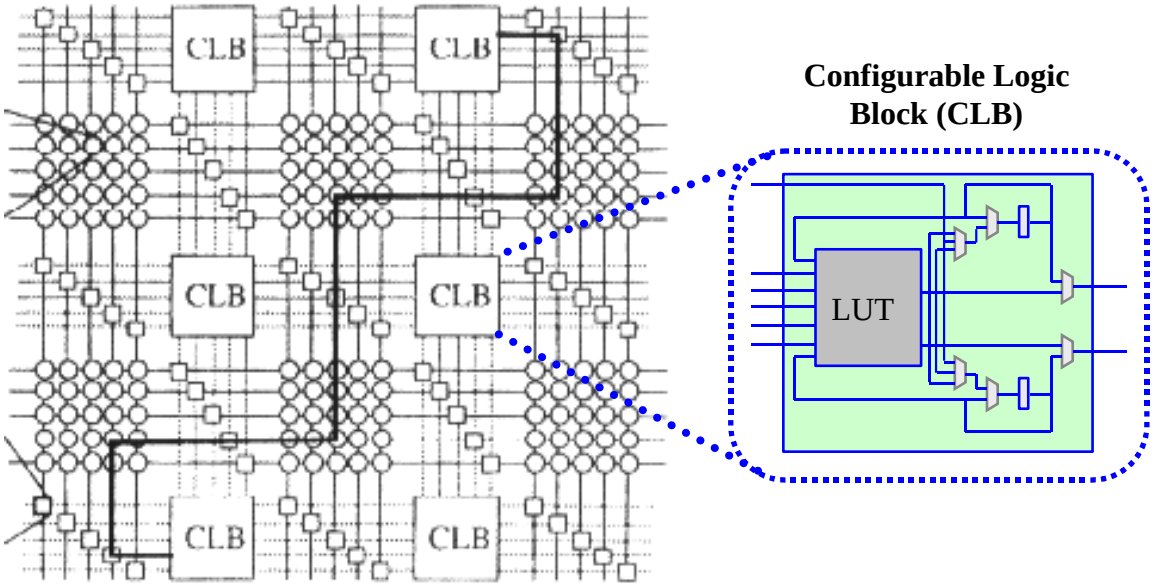
Register von verschiedenen Quellen taktbar (lokal, global ggf. mit Enable)

Register auf D, T, JK oder SR Typ einstellbar.

2.7.3 Field Programmable Gate Array: Struktur

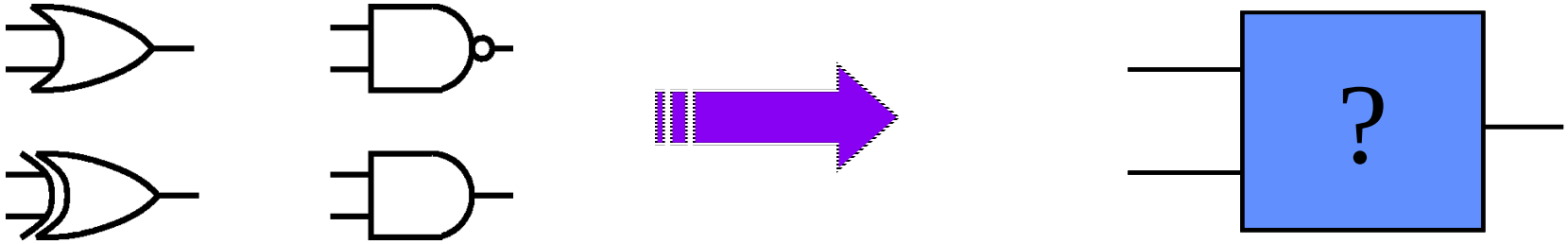
- FPGA-Architekturen
  - SRAM-basierende Look-up Tables (LUTs)
  - Probleme:
    - Verdrahtung: reduziert Performanz
    - Verhältnis: aktive / passive Elemente
- rekonfigurierbare Verbindungen (Switch Matrix)





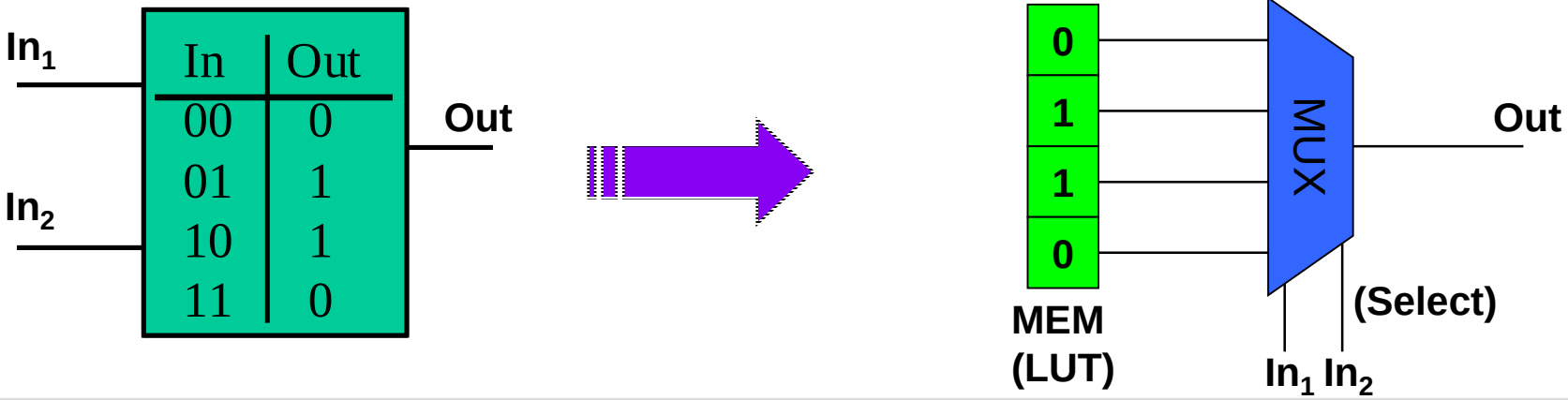
2.7.3 RAM-basierter FPGA : Logikrealisierung

- Menge programmierbarer “Logikgatter”, integriert in ein flexibles Verbindungsnetzwerk
  - eine “benutzerprogrammierbare” Alternative zu dedizierten Schaltungen

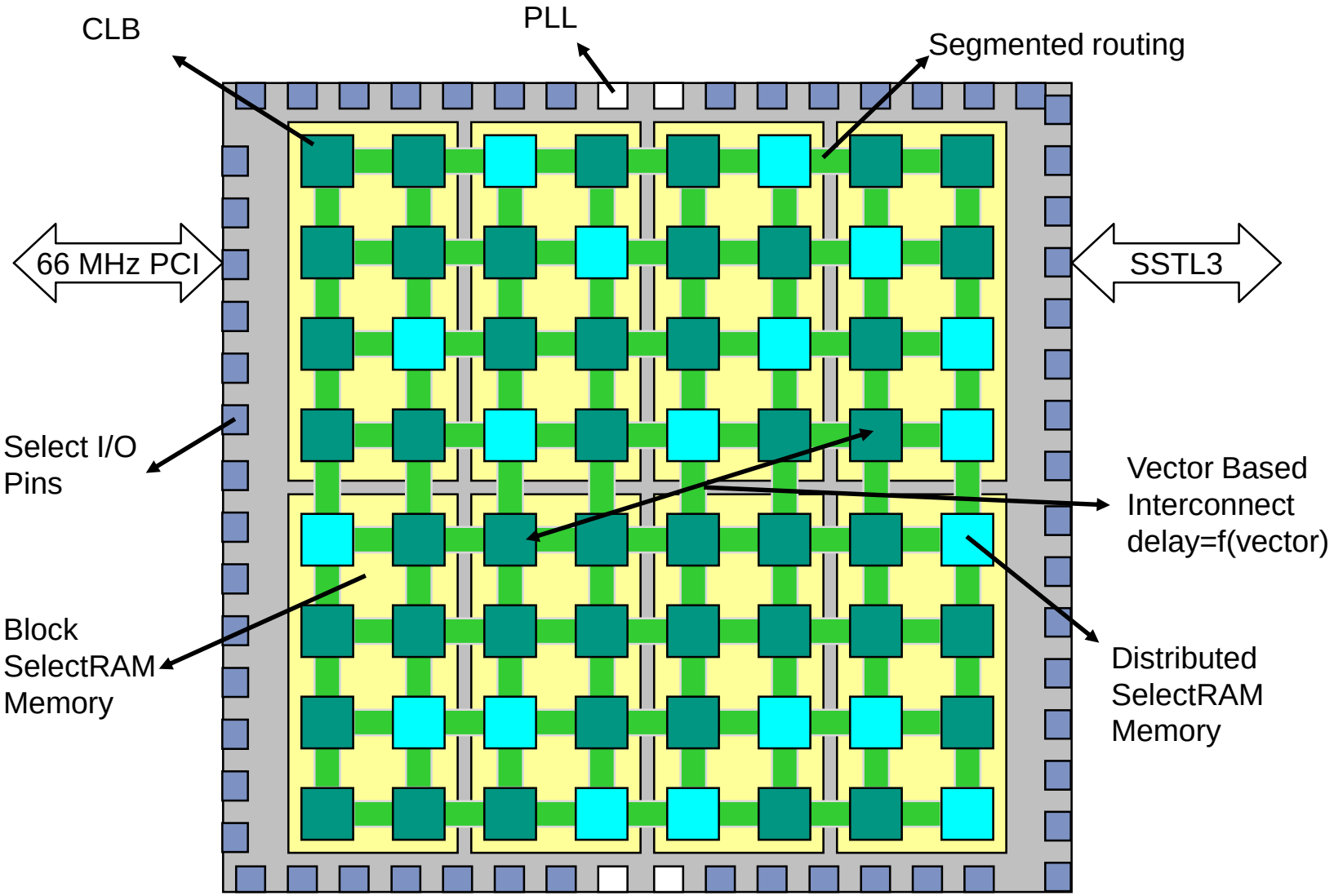


Programmierbares Gatter

- Lösung:
  - Look-up-Table (LUT) mit 2 Eingängen



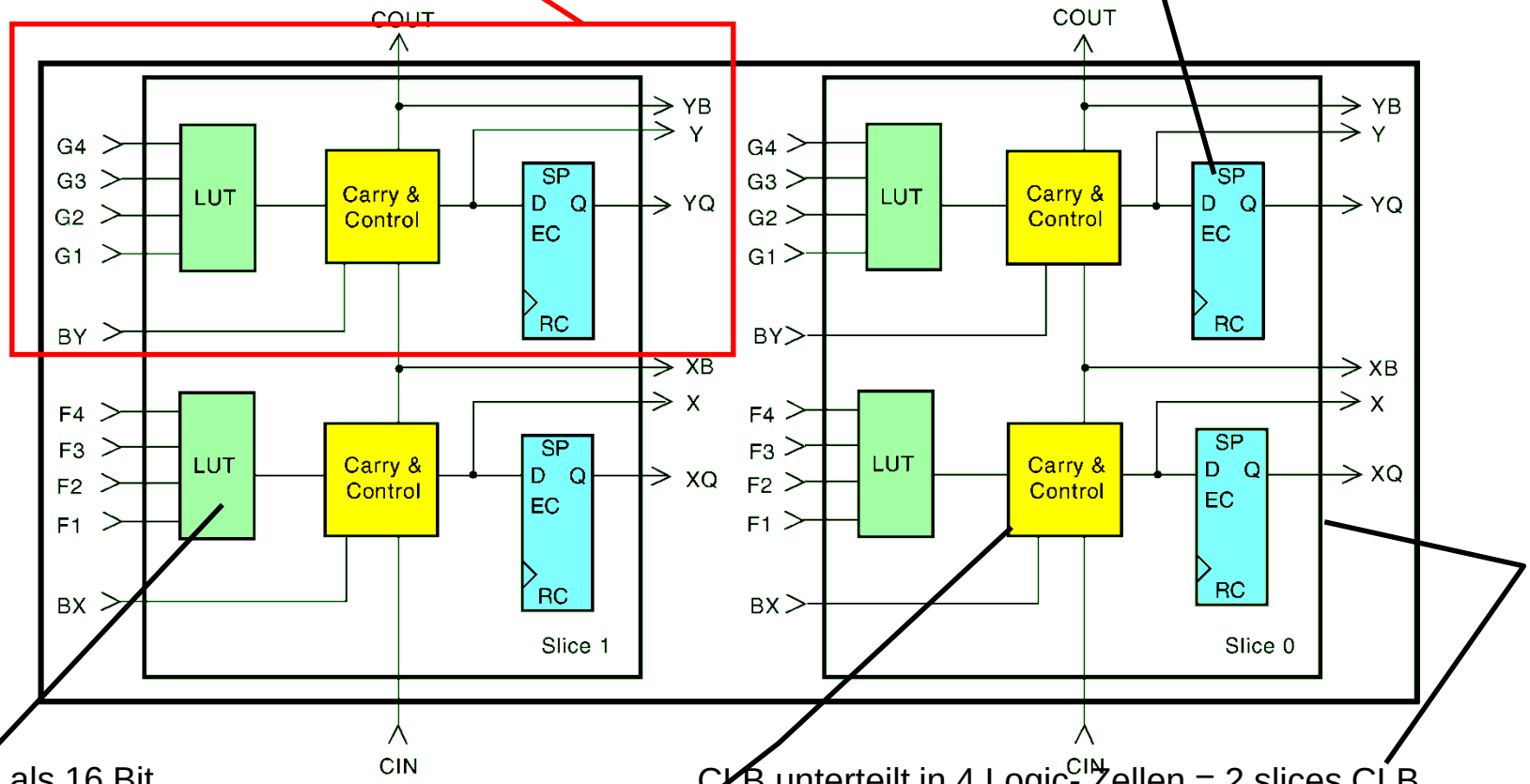
2.7.4 Xilinx Virtex: Funktionsblock-Architektur



2.7.4 Xilinx Virtex: CLB

Logic Cell
LUT mit 4 Eingängen
Carry-Logik und Speicher

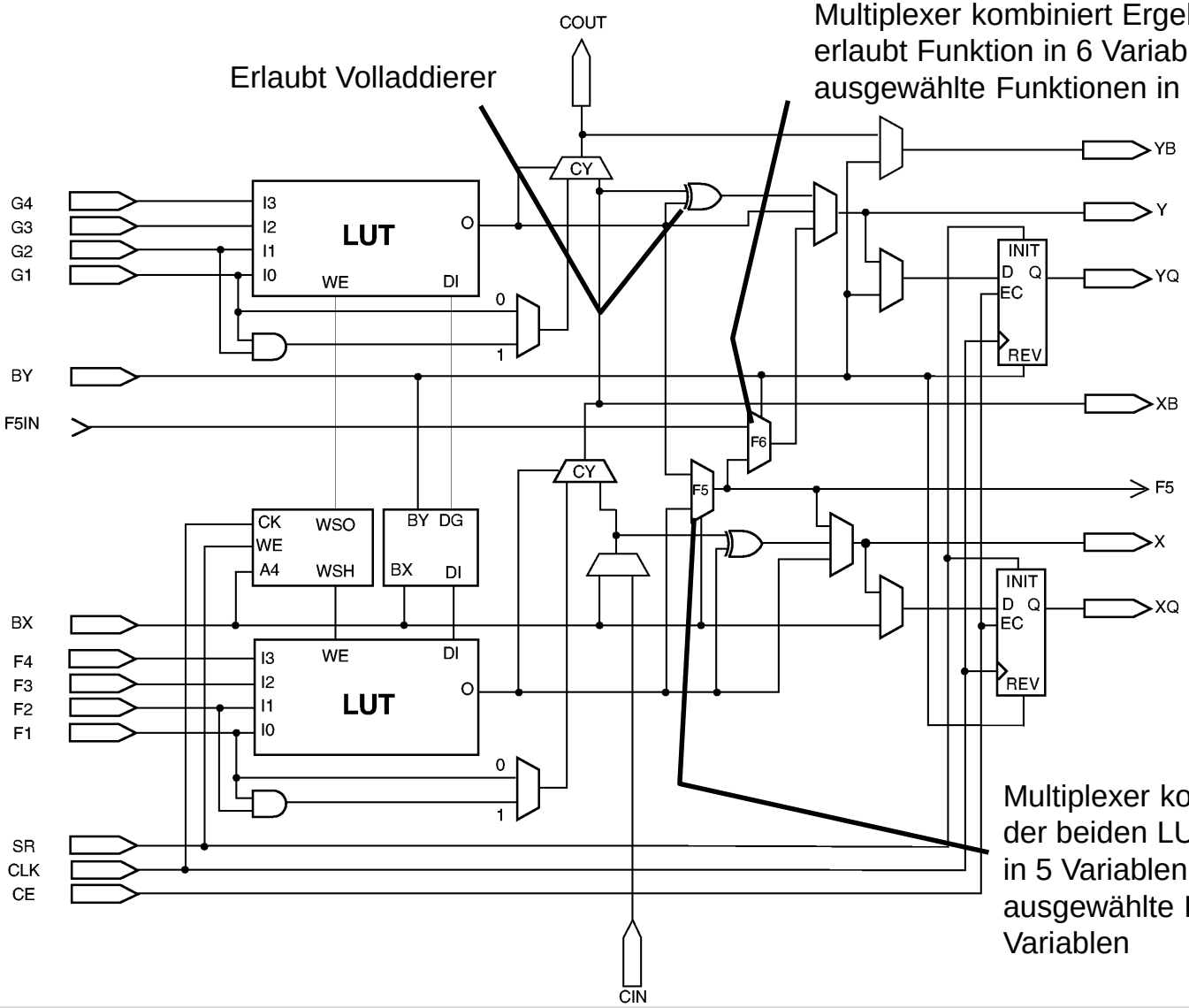
Speicher als D- FlipFlops
oder pegelgesteuerte Latches



LUT auch als 16 Bit
RAM verwendbar

CLB unterteilt in 4 Logic- Zellen = 2 slices CLB
Enthält zusätzliche Logik, um Funktionsgeneratoren
in 5 oder 6 Eingängen zu realisieren.

2.7.4 Detaillierte Ansicht eines Virtex-Slice



2.7.4 FPGA Beispiel: Xilinx 7-Series

- Artix-7 Family:
  - Optimized for lowest cost and power with small form-factor packaging for the highest volume applications.
- Kintex-7 Family:
  - Optimized for best price-performance with a 2X improvement compared to previous generation, enabling a new class of FPGAs.
- Virtex-7 Family:
  - Optimized for highest system performance and capacity with a 2X improvement in system performance. Highest capability devices enabled by stacked silicon interconnect (SSI) technology.

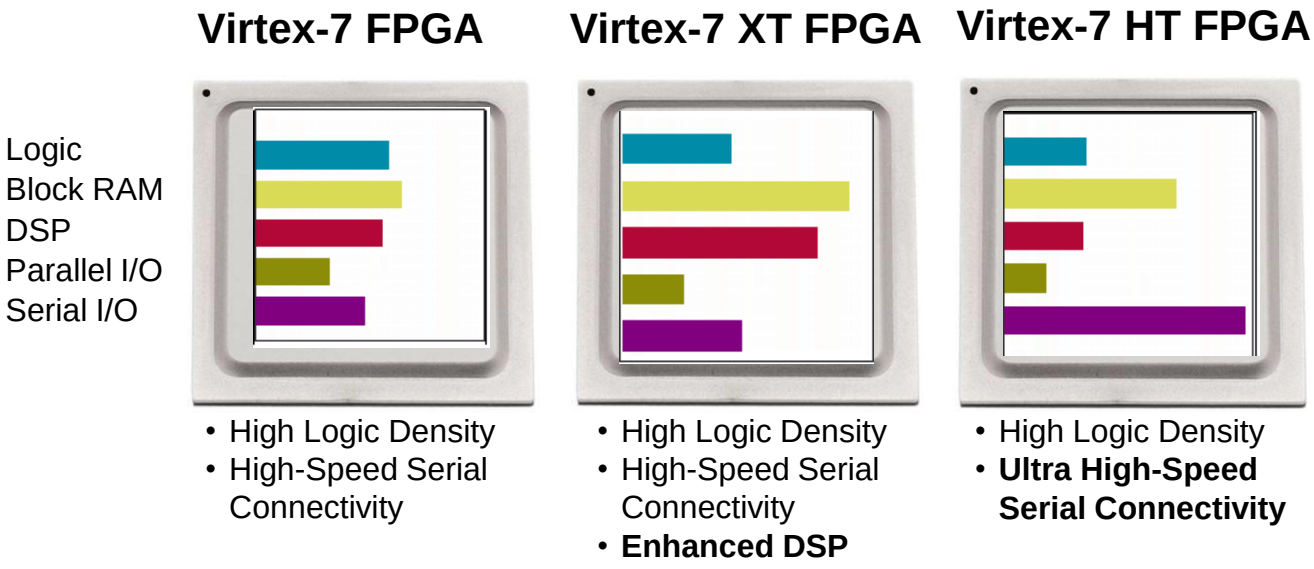
2.7.4 FPGA Beispiel: Xilinx 7-Serie

Maximum Capability	Artix-7 Family	Kintex-7 Family	Virtex-7 Family
Logic Cells	215K	478K	1,955K
Block RAM	13 Mb	34 Mb	68 Mb
DSP Slices	740	1,920	3,600
Peak DSP Performance	929 GMAC/s	2,845 GMAC/s	5,335 GMAC/s
Transceivers	16	32	96
Peak Transceiver Speed	6.6 Gb/s	12.5 Gb/s	28.05 Gb/s
Peak Serial Bandwidth (Full Duplex)	211 Gb/s	800 Gb/s	2,784 Gb/s
PCIe Interface	X4 Gen2	X8 Gen2	X8 Gen2
Memory Interface	1,066 Mb/s	1,866 Mb/s	1,866 Mb/s
I/O Pins	500	500	1,200

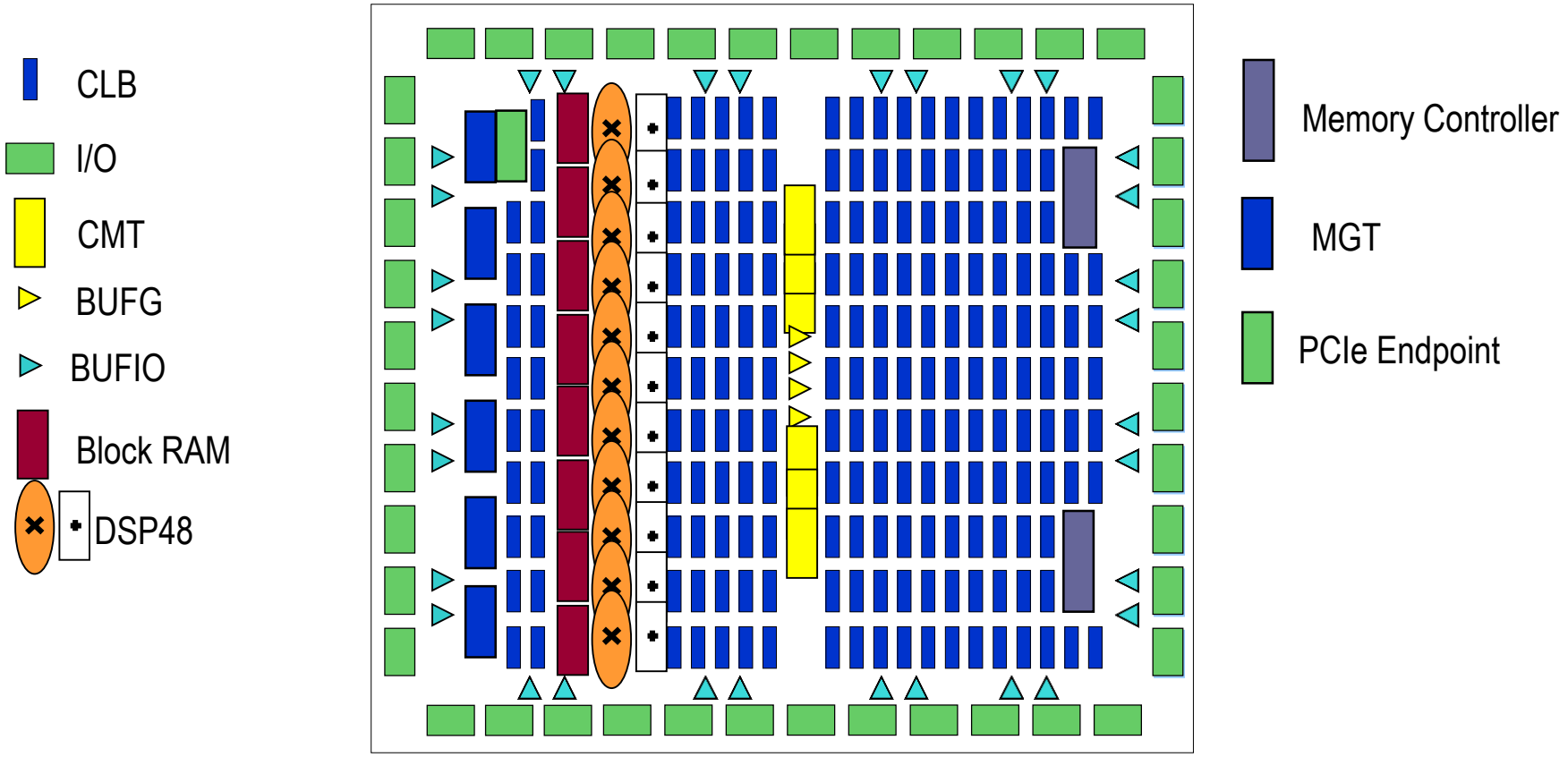
http://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf

2.7.4 FPGA Beispiel: Xilinx 7-Serie Sub-Familien

- The Virtex-7 family has several sub-families
  - Virtex-7:
    - General logic
  - Virtex-7XT:
    - Rich DSP and block RAM
  - Virtex-7HT:
    - Highest serial bandwidth

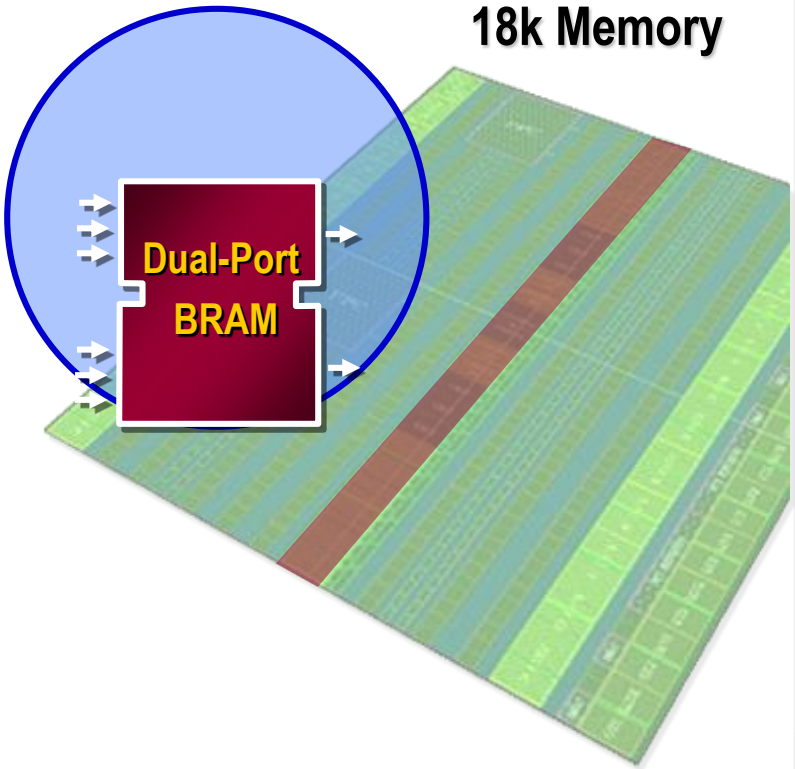


2.7.4 FPGA Beispiel: Spartan-6

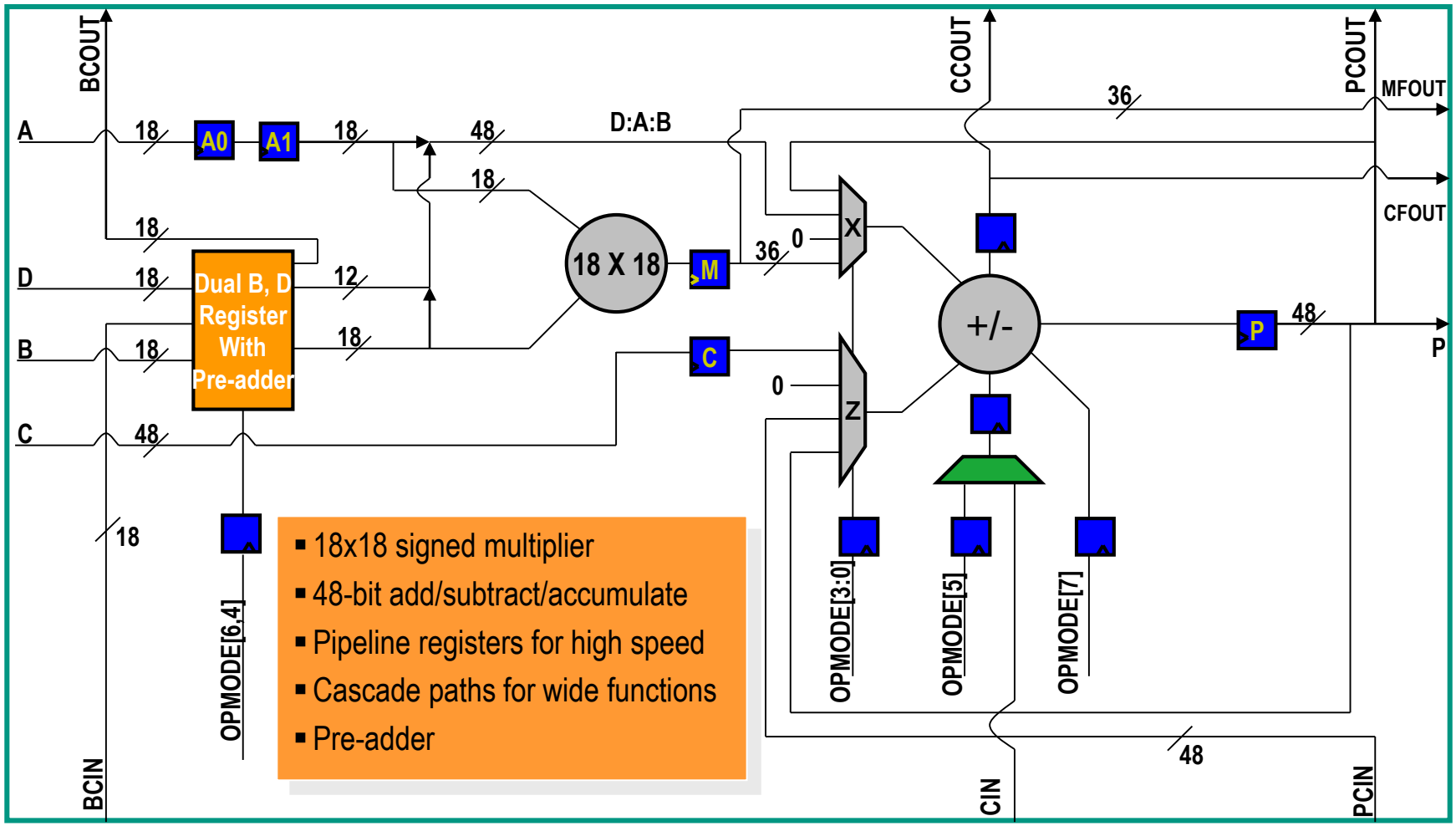


2.7.4 FPGA Beispiel: Spartan-6 Block RAM Features

- 18 kb size
  - Can be split into two independent 9-kb memories
- Performance up to 300 MHz
- Multiple configuration options
  - True dual-port, simple dual-port, single-port
- Two independent ports access common data
  - Individual address, clock, write enable, clock enable
  - Independent widths for each port
- Byte-write enable

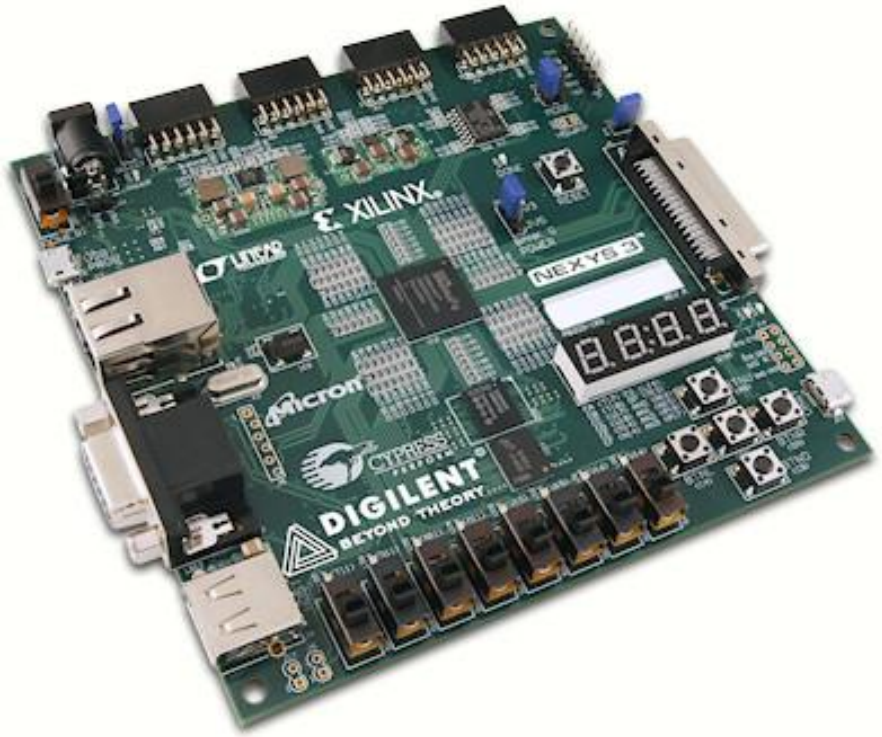


2.7.4 FPGA Beispiel: Spartan-6 DSP48A1 Slice



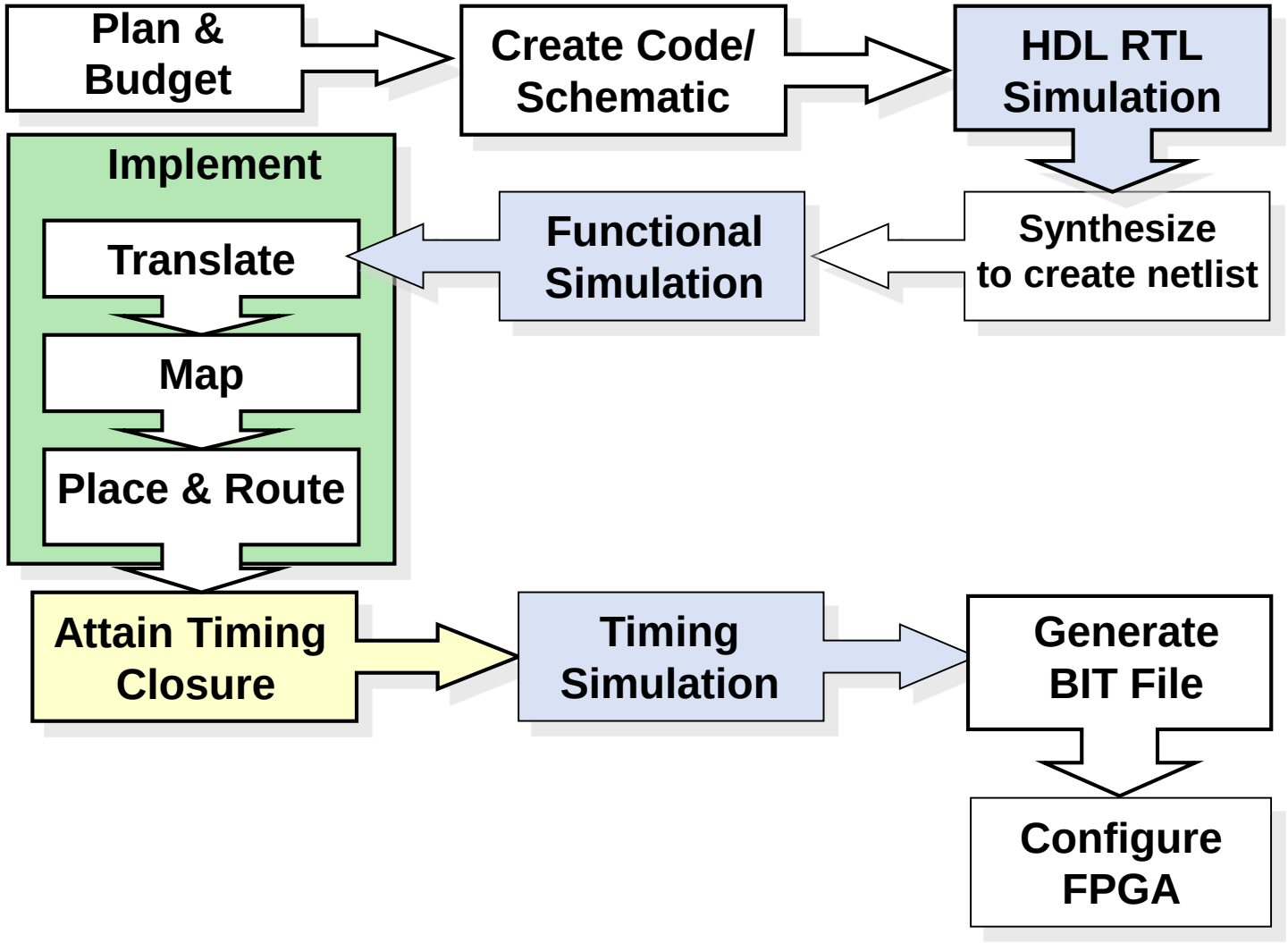
2.7.4 FPGA Beispiel: Nexys3 Spartan-6 FPGA Board

- Xilinx Spartan6 XC6LX16-CS324
- 16Mbyte Micron Cellular RAM
- 16Mbyte Micron Parallel PCM
- 16Mbyte Micron Quad-mode SPI PCM
- 10/100 SMSC LAN8710 PHY
- Digilent Adept USB port for power, programming & data transfers
- USB-UART
- Type-A USB host for mouse, keyboard or memory stick
- 8-bit VGA
- 100MHz fixed-frequency oscillator
- 8 slide switches, 5 push buttons, 4-digit 7seg display, 8 LEDs
- Four double-wide Pmod™ connectors, one VHDC connector
- Rugged plastic case, USB cable included



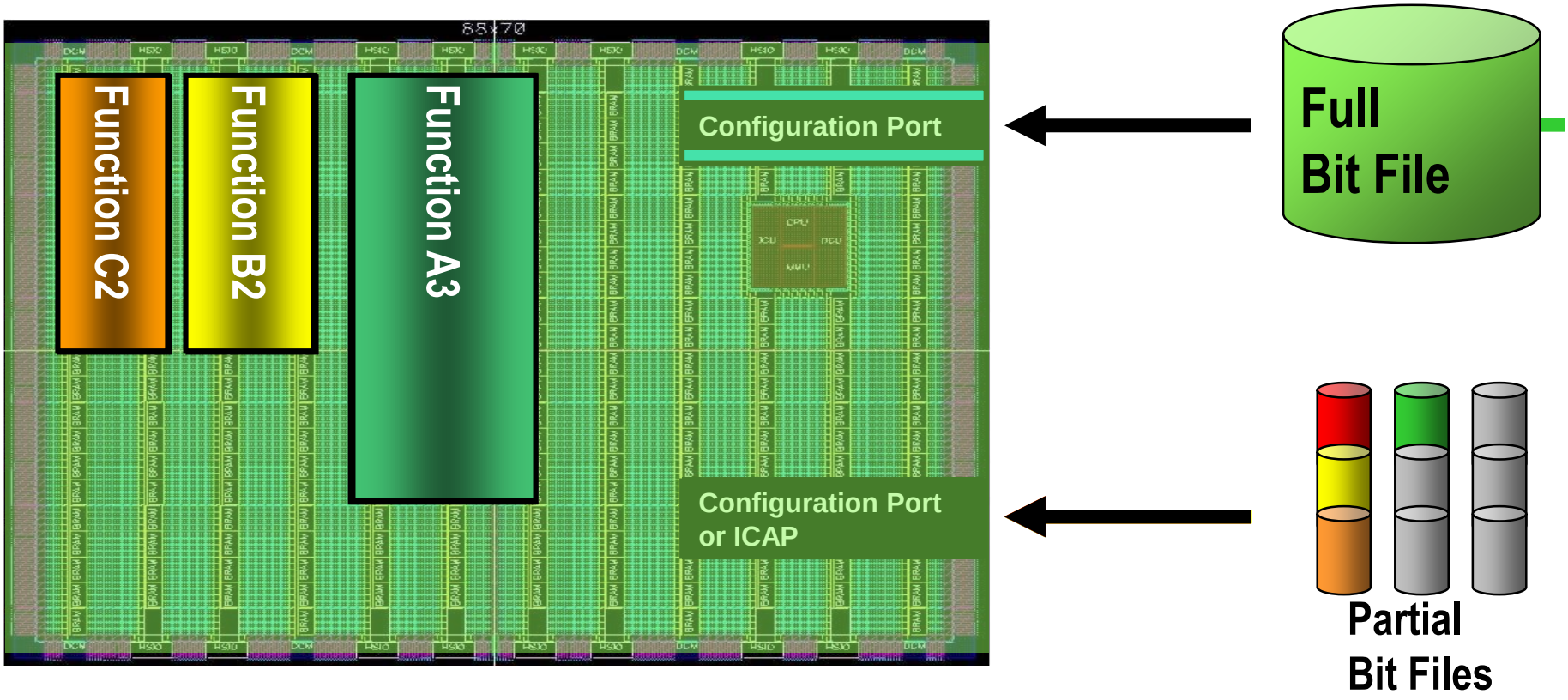
http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,400,897&Prod=NEXYS3

2.7.5 FPGA Design Flow



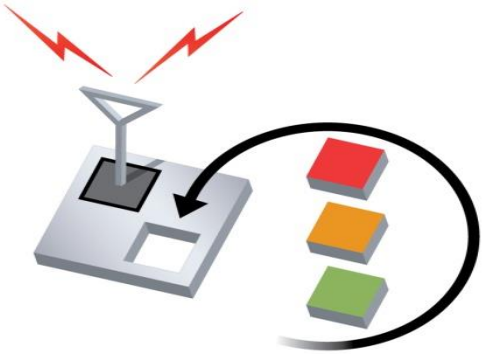
2.7.6 Was ist partielle Rekonfiguration?

- Partielle Rekonfiguration ist die Fähigkeit teile der Hardwarestrukturen dynamisch während der Laufzeit durch das Laden eines partiellen Bitfiles anzupassen, ohne dass die restliche Funktion beeinflusst wird.

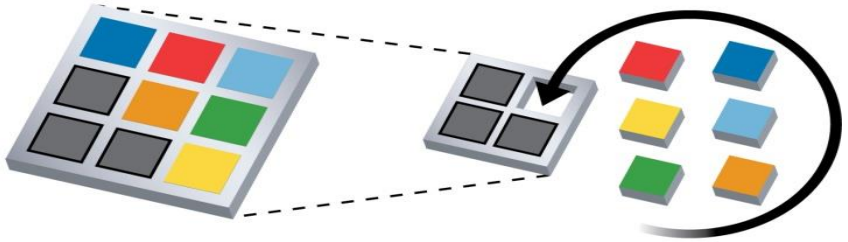


2.7.6 Partielle Rekonfiguration – Technologie & Nutzen

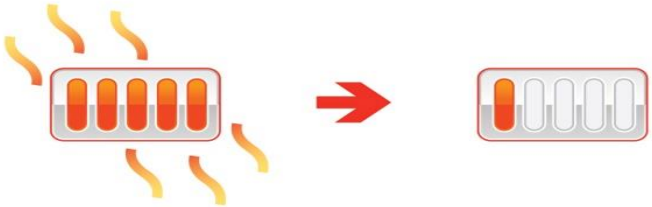
- Partial Reconfiguration ermöglicht:
  - Flexibilität
    - Funktionen können angepasst werden, während die Kommunikation bestehen bleibt



  - Flächen- und Kostenreduktion
    - Betrieb der Hardware im Zeitmultiplexbetrieb um einen kleineren FPGA nutzen zu können.



  - Reduktion des Energieverbrauchs
    - Abschalten unbenötigter Tasks
    - Verwendung eines kleineren FPGAs



2.7.7 Einsatzgebiete von rekonfigurierbarer Hardware

- Glue Logic (Interfaces)

- Rapid Prototyping, Emulation
  - Ensemble von Gate-Arrays, die zur Emulation einer Schaltung vor der eigentlichen Herstellung erzeugt werden
  - Erlaubt schnelleres und besseres Debugging als mit reiner Simulation

- Eingebettete Systeme
  - schneller als Prozessor & flexibler als ASIC
  - ASIC Replacement
    - Stückzahl zu gering für ein ASIC
    - verkürzte Time-to-market
  - Prozessor-/Datenpfad-Replacement
    - Controller/Datenpfad als Teil eines Configurable Systems-on-Chip (CSoC)

- Custom Computing
  - Ziel: Verbinden der Flexibilität von Prozessoren mit der Performanz von ASICs
  - Hardwarebeschleunigung von Algorithmen

- Welche Entwurfsstilte gibts es? Was sind Vor- und Nachteile?
- Welche Gate-Array Technologien gibt es?
- Welche Technologien gibt es für programmierbare Logik? Welche Vor- bzw Nachteile haben diese?
- Wie sieht eine FPGA-Architektur aus?
- Wie wird die Logik bei SRAM-basierten FPGAs abgebildet?
- Was ist (partielle) dynamische Rekonfiguration?

Inhalt

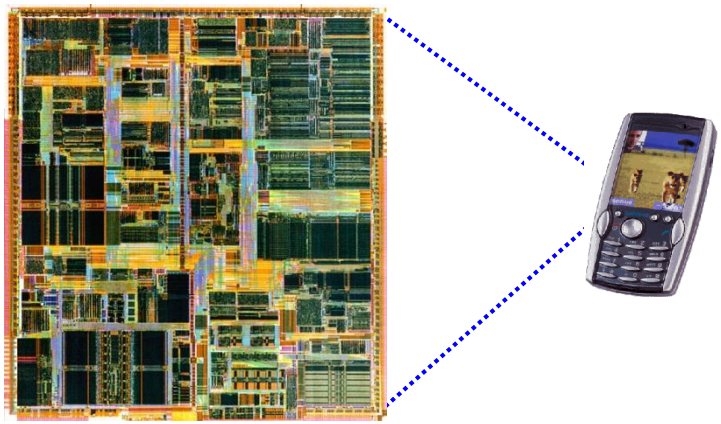
- 2.4 Spezialprozessoren
  - 2.4.1 Mikrocontroller (µC)
  - 2.4.2 Digitale Signalprozessoren (DSPs)
  - 2.4.3 Grafik Prozessoren (GPU)
- 2.5 Bus, Network-on-Chip (NoC) & Multicore
- 2.6 Anwendungs-spezifische Instruktionssatz Prozessoren (ASIP)
- 2.7 Field Programmable Gate Arrays (FPGA)
- **2.8 System-on-Chip (SoC)**

2.8.1 System-on-Chip (SoC): Kostenmodelle + Realisierungsbewertung

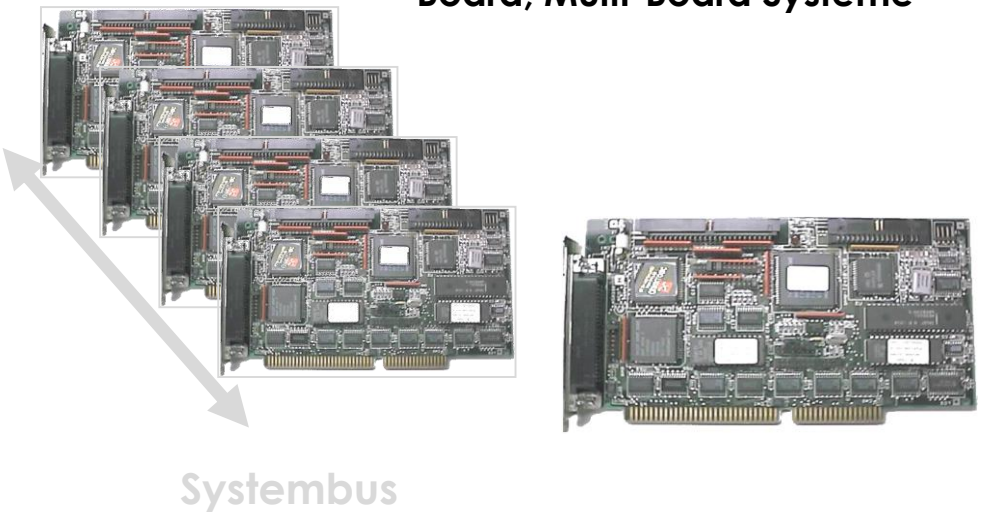
- Kriterien:
  - Kosten
    - Fläche + Entwicklung
  - Verlustleistung
  - Performanz
  - Flexibilität
    - Risikominimierung, Time-to-Market

	SoC	Board	Multi-Board
Gewicht, Größe, Leistungsverbrauch	niedrig	hoch	Sehr hoch
Zuverlässigkeit	Sehr hoch	Niedrig/hoch	niedrig
Kosten bei hoher Stückzahl	niedrig	hoch	hoch

System-on-Chip (SoC)



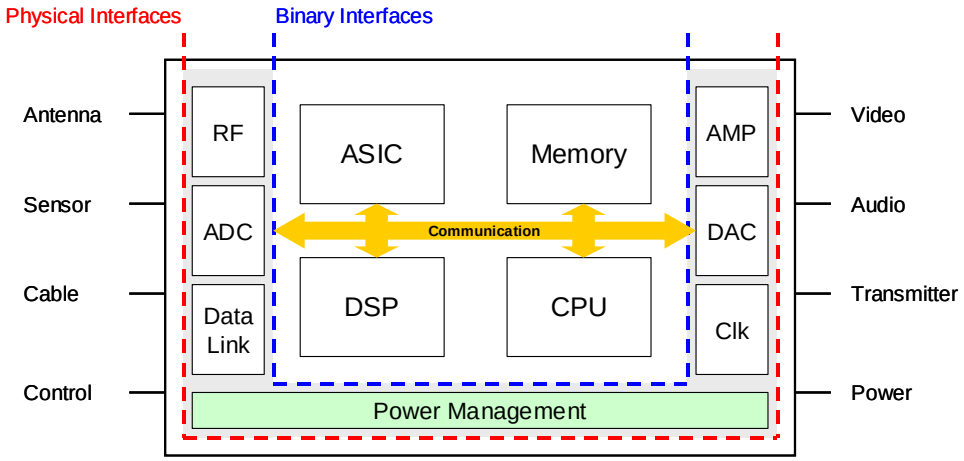
Board, Multi-Board Systeme



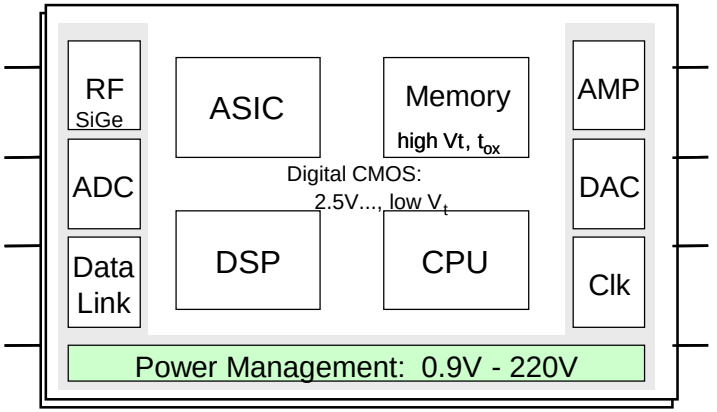
2.8.1 SoC: Funktionale und Technologische Perspektiven

- Aspekte für “Single Die” SoC:
  - unterschiedliche Prozess-Anforderungen
  - Unterschiedliche Spannungen erforderlich
  - Schaltungen: re-designed für Haupt-Technologien

<u>Funktional:</u> Komponenten + Interfaces



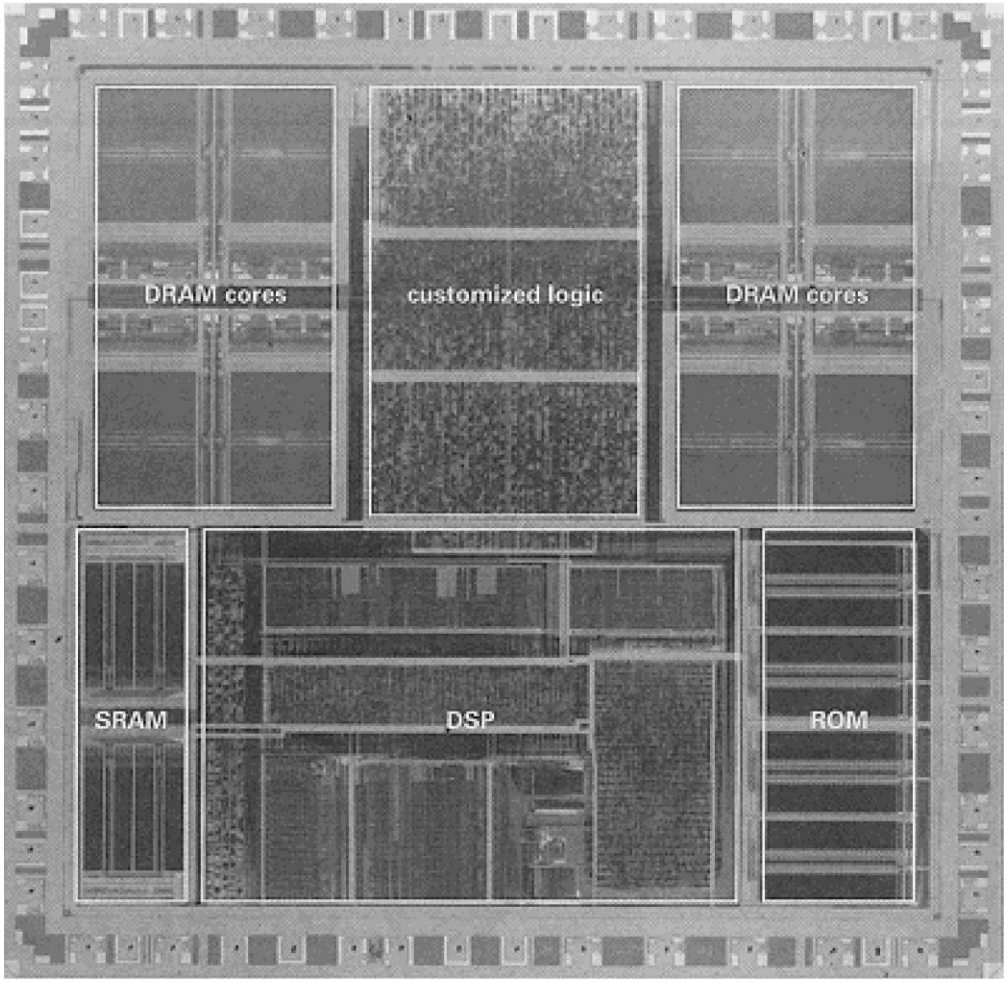
<u>Technologisch:</u> Komponenten + Realisierung



2.8.1 SoC Beispiel: Audio Processing SoC von Siemens

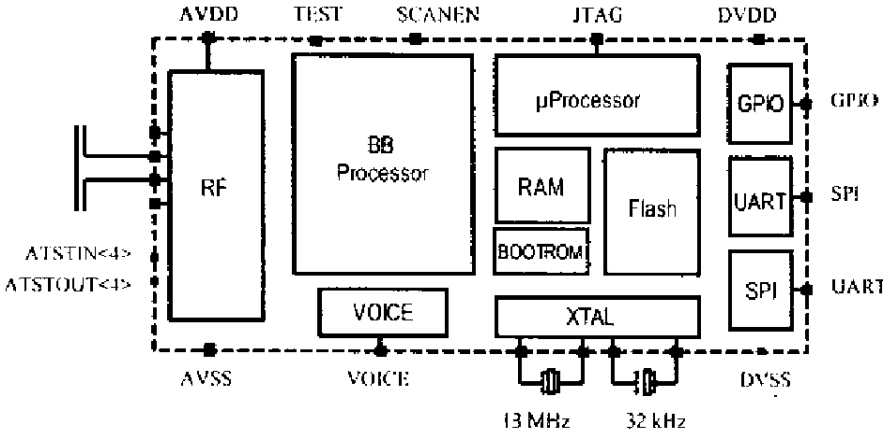
- 16 bit DSP
- 15 k gates customized logic
- SRAM, ROM
- 1 M Bit DRAM

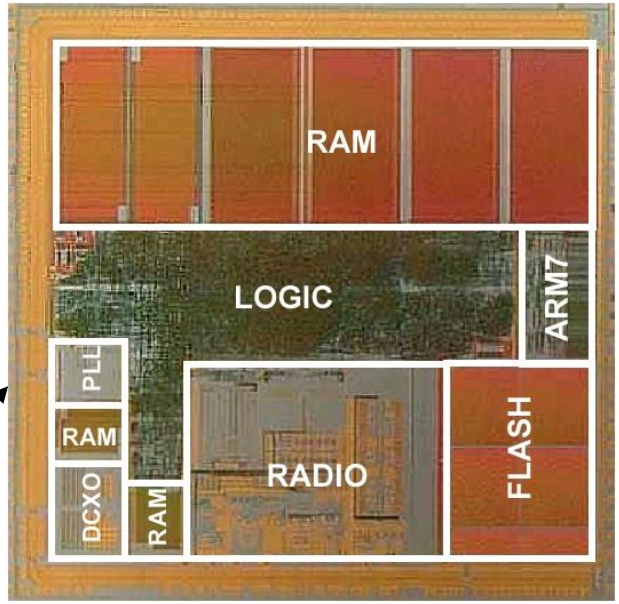
- Anwendungen:
  - Hörgeräte
  - Biometrie
  - Spracherkennung



2.8.1 SoC Beispiel: Bluetooth SoC von Alcatel

- Entwurf wegen Inkompatibilitäten zur Bluetooth Spezifikation eingemottet.
  - CMOS 0,25 □m, 2,5 V Core
  - 40,1 mm2 „die“, wovon 50 % vom RAM belegt werden
  - Empfindlichkeit –80dBm bei 0,1% BER
  - Erfüllt nur BT Spezifikation 1.0.b
  - 5 GHz VCO on Chip, mit automatischer Rekalibrierung
  - 48 KB RAM und 256 KB Flash ROM für Applikationssoftware
  - Entwicklungszeit :18 Monate
  - Rekonfigurierbare UART/Voice Interfaces
  - Frei programmierbares Interface (GPIO), erlaubt Applikationen für eingebettete Systeme, z.B. Bluetooth Headset
  - Demodulation auf „low IF“ mit 1MHz IF
  - Leistungsklasse 2 und 3, (125 mW bei Empfang
  - Antenne sowie andere externe Komponenten auch „on Chip“, d.h. im Chip- Gehäuse (in Ball Grid Array Package)
  - 0,18 □m Prozeß war geplant.





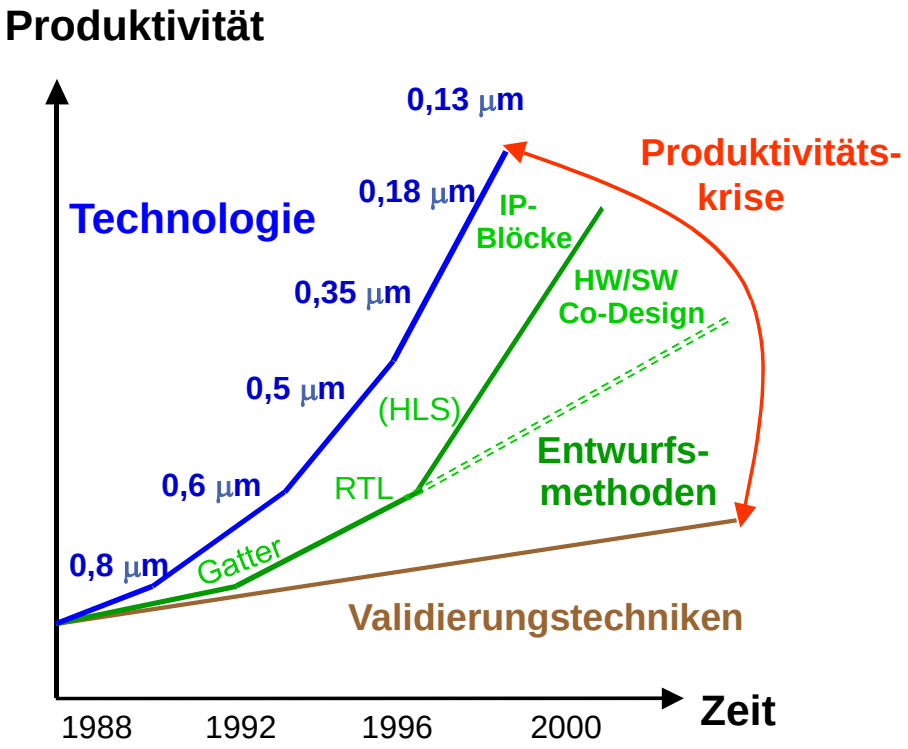
Single Chip Design

2.8.2 SoC - Produktivitätsgrenze

- Komplexe eingebettete Systeme benötigen neue Entwurfsmethoden
  - > 50% Entwicklungszeit für Simulation
  - > 25% Validierung/Verifikation

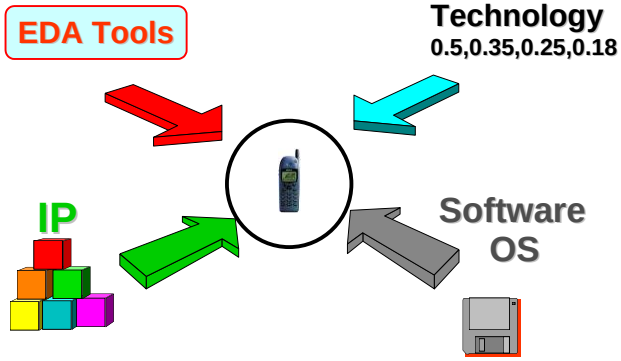
- Komplexität größer als Produktivitätssteigerung
  - Rapid Prototyping
  - HW/SW Co-Simulation
  - HW/SW Co-Verifikation
  - IP-basierter Entwurf

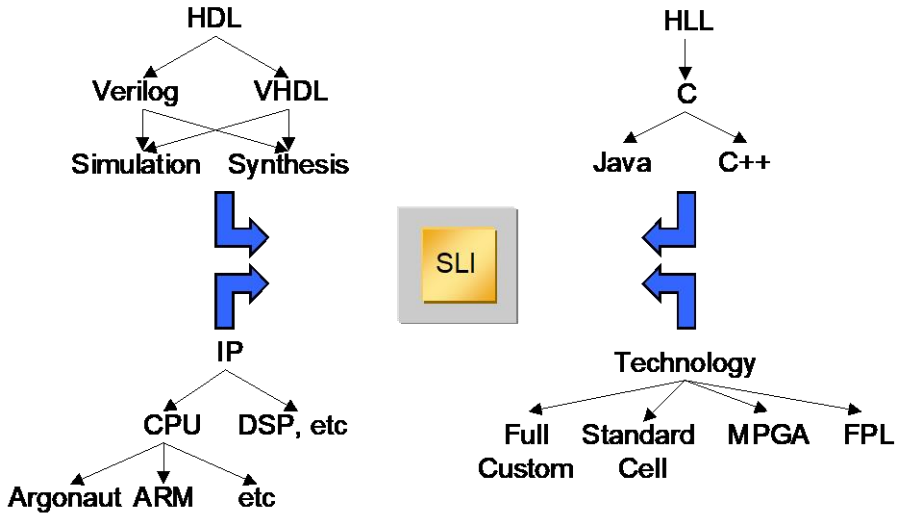
- Neue effiziente CAD-Methoden
  - Entwurfsraums Exploration



2.8.2 Komplexe SoCs: System-Level Integration

- SoC-Entwurf: Neue IP-basierte Methodik (Intellectual Property)
  - Erhöhung der Abstraktionsebene
  - Prozess-basierte nebenläufige Modelle
    - SystemC, UML-RT, CC++, SDL, JAVA
    - MATLAB, STATEMATE, COSSAP, SPW
- Gekoppelte Simulation und Integration heterogener Modelle
  - HW-SW, digital-analog, elektronisch-mechanisch (Hardware-in-the-Loop)
  - Standardisierung von Werkzeugen + Formaten Anwendungsspezifik





2.8.2 IP-basierte Entwurfsmethodik: IP-Blocktypen

- Soft blocks / IP
  - in einer HDL beschrieben
  - Synthetisierbar, ggf. technologieabhängig
- Firm blocks / IP
  - HDL und Netzlisten, evtl. auch Floorplanning
  - schneller synthetisierbar (technologie-spezifische Syntheseinformationen)
- Hard blocks / IP
  - komplettes Layout + Maskendaten für bestimmte Zieltechnologie
  - nicht mehr synthetisierbar (keine HDL Beschreibung)

Blocktyp	Flexibilität	Vorhersagbarkeit	Portabilität	IP-Schutz
Soft	Sehr flexibel	Unvorhersehbar	Unbegrenzt	Keiner
Firm	Flexibel	Unvorhersehbar	Abbildung auf Bibliothek	Keiner
Hard	Inflexibel	Sehr vorhersehbar	Prozess-Abbildung	gut

2.8.2 SoC IP Bibliothek
Beispiel: Aeroflex Gaisler

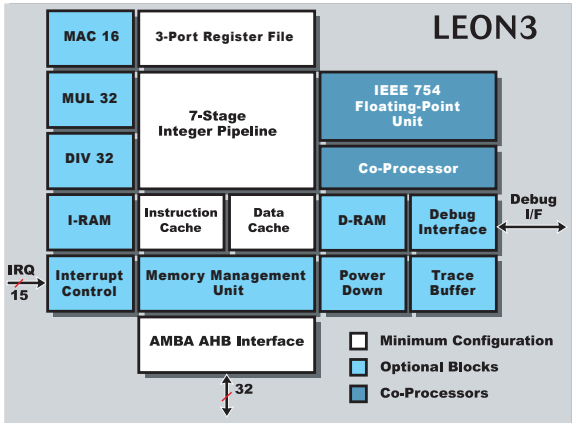
- Komplettes Framework für die Entwicklung von Prozessor-basierten SOC Designs:
  - LEON Prozessor Kern
  - Große IP Sammlung
    - Verhaltens-Simulatoren
    - Verwandte Software Entwicklungswerkzeuge
- Framework wurde für duale Nutzung entwickelt:
  - Entwicklung von kritischen Systemen z.B. Luft- und Raumfahrtsystems
  - Kosten-effizienten Verbraucher Produkte
- Weit verbreitet (insbesondere auch im akademischen Umfeld)

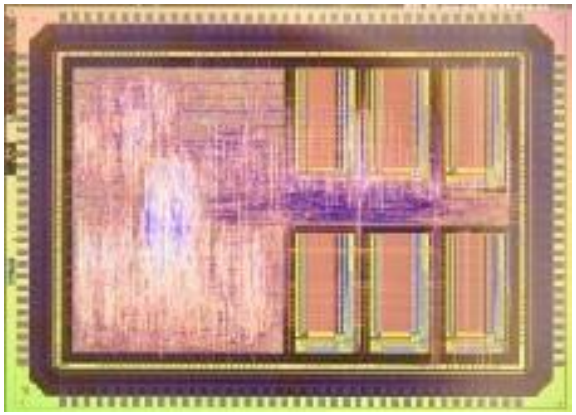
2.8.2 SoC IP Beispiel: Aeroflex Gaisler GrLib

- Integrierte SoC Entwicklungsplattform
- Basiert auf dem AMBA Bus Standard
  - LEON3 SPARC Prozessor
  - LEON4 SPARC Prozessor
  - Voll gepipelinete double-precision IEEE-754 floating point Einheit
  - 32-bit master/target PCI core mit DMA und FIFOs
  - Speicher Controller
  - SpaceWire codec mit RMAP support
  - 10/100/1000 Mbit Ethernet MAC
  - USB Host und Geräte Controller
  - CAN Controller
  - Timer, Interrupt Controller, UART, VGA Controller, PS/2 Interface, GPIO
  - AES Kryptographie

2.8.2 SoC IP Beispiel: Aeroflex Gaisler – Leon3

- Synthetisierbares VHDL Modell eines 32-bit Prozessors
- Hoch konfigurierbar, besonders passend für system-on-a-chip (SoC) Designs
  - SPARC V8 Instruktionssatz mit V8e Erweiterungen
  - 7-stufige Pipeline
  - Hardware Multiplizierer, Dividierer und MAC Einheit
  - High-performance, fully pipelined IEEE-754 FPU
  - Separater Instruktions- und Daten Cache (Harvard architecture)
  - AMBA-2.0 AHB bus interface
  - Symmetric Multi-processor support (SMP)
  - Bis zu 125 MHz in FPGA und 400 MHz auf 0.13 µm ASIC Technologien
  - High Performance: 1.4 DMIPS/MHz, 1.8 CoreMark/MHz (gcc -4.1.2)
- GNU GPL Lizenz: freie und uneingeschränkte Nutzung für Forschung and Lehre

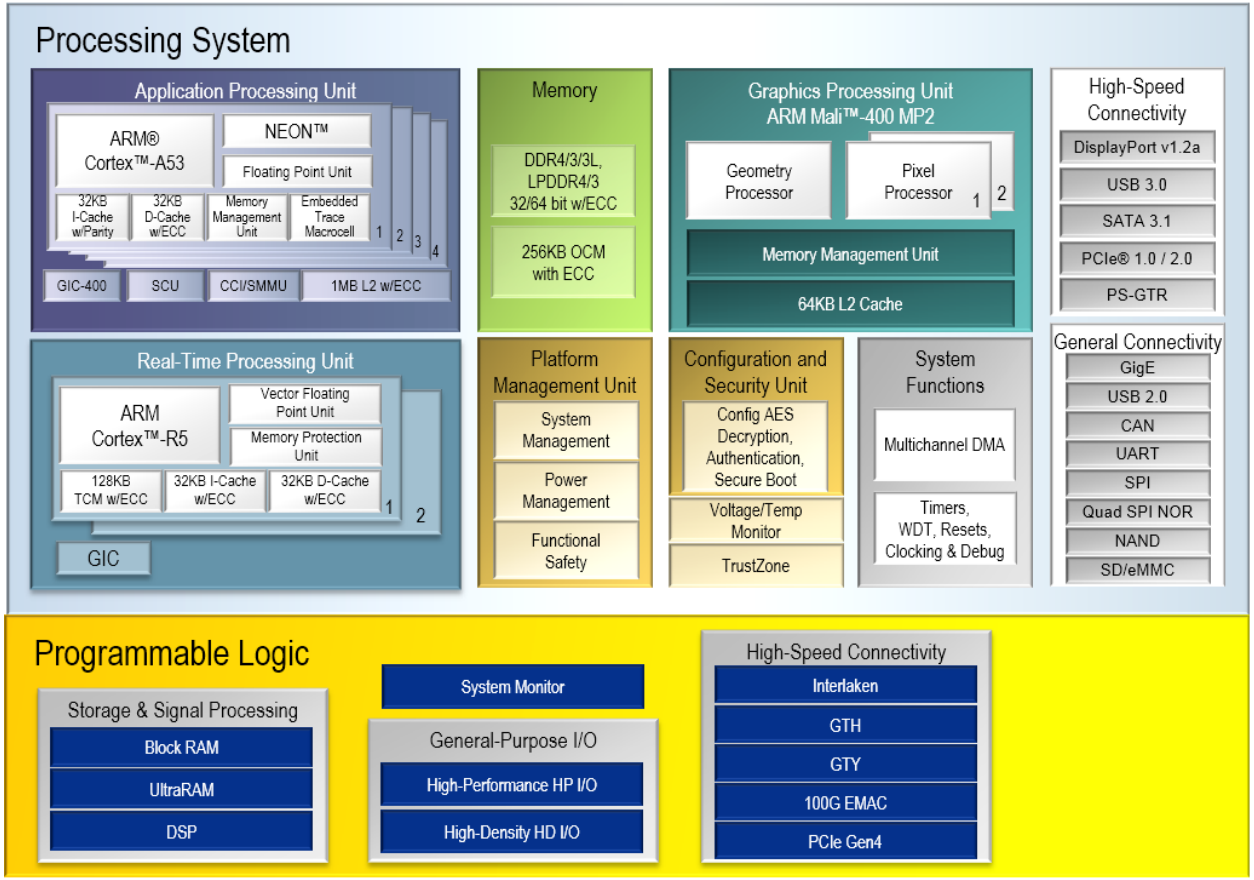




http://www.gaisler.com/index.php/products/processors

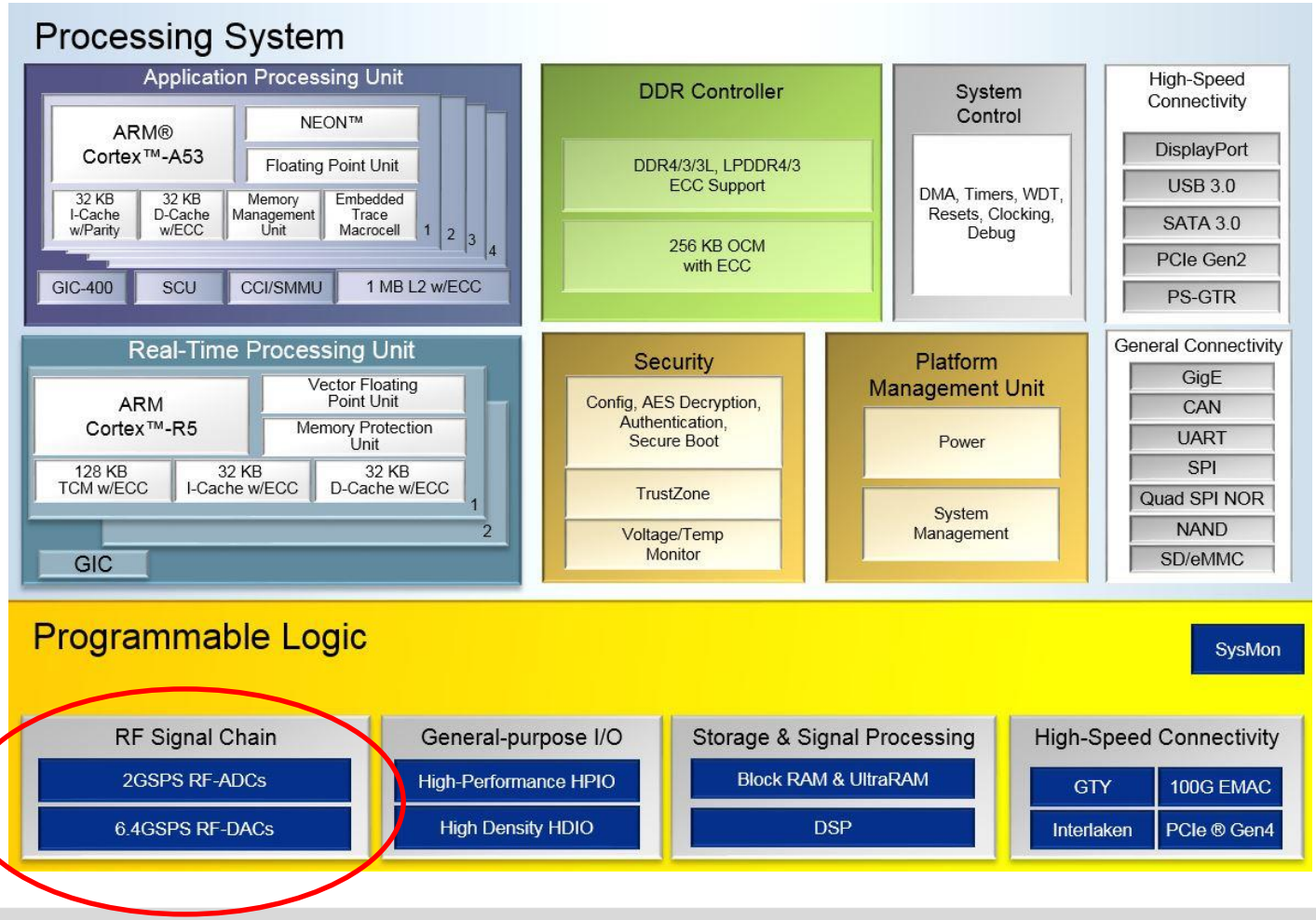
2.8.2 SoC Beispiel: Zynq Ultrascale+ EG

- 16nm Finfet+ Low Power programmable Logic
  - 103k to 1143k Logic Cells
  - 5,3 MB to 34,6 MB Block RAM
  - 240 to 1968 DSP Slices
- Processing System
  - Quad Core ARM Cortex A53
    - Up to 1500 MHz
    - 32kB I-&D-Cache
    - 1 MB L2 Cache
    - 256kB on-chip memory
  - Dual Core ARM Cortex R5
    - 32 kB I-&D-Cache
    - 128 kB TCM
  - Mali-400 MP2 GPU
- High Speed Transceivers
  - 16 bis 44 16.3 Gbps
  - 0 bis 28 32,75 Gbps



2.8.2 SoC Beispiel: Zynq UltraScale+ RFSoCs

- Trend: Integration of ADC / DAC



- Was ist ein System-on-Chip?
- Welche Kriterien zeigen die Relevanz von SoCs?
- Wo liegt das Problem im SoC-Entwurf? Welche Lösungen werden angetrebt?
- Was ist IP-basierter Entwurf? Wie funktioniert dieser?

Kapitel 2 Inhalt (I)

- 2.1 Allgemeiner Aufbau
- 2.2 Klassifikation
- 2.3 General-Purpose Prozessoren (GPP)
  - 2.3.1 Architekturen
    - 2.3.1.1 Akkumulatormaschine
    - 2.3.1.2 Stackmaschine
    - 2.3.1.3 Registersatzmaschine
  - 2.3.2 Performanzsteigerung
    - 2.3.2.1 Pipelining
    - 2.3.2.2 Superskalarität / Out-of-Order Execution
    - 2.3.2.3 Very Long Instruction Word (VLIW)
    - 2.3.2.4 Single Instruction Multiple Data (SIMD)
    - 2.3.2.5 Caches
    - 2.3.2.6 Multiple Instruction Multiple Data (MIMD)

Kapitel 2 Inhalt (II)

- 2.4 Spezialprozessoren
  - 2.4.1 Mikrocontroller (µC)
  - 2.4.2 Digitale Signalprozessoren (DSPs)
  - 2.4.3 Grafik Prozessoren (GPU)
- 2.5 Bus, Network-on-Chip (NoC) & Multicore
- 2.6 Anwendungs-spezifische Instruktionssatz Prozessoren (ASIP)
- 2.7 Field Programmable Gate Arrays (FPGA)
- 2.8 System-on-Chip (SoC)