

# Eingebettete Prozessorarchitekturen

## 10. Einführung in Adaptive Computer

Andreas Koch

FG Eingebettete Systeme und ihre Anwendungen  
Informatik, TU Darmstadt

Wintersemester 2009/2010

# Organisatorisches

- 4 Programmieraufgaben mit je 8 Punkten
  - VLIW, Tensilica, Stretch, Verilog
  - 5. Aufgabe (PICO) optional, nicht bewertet
- Bewertung
  - Ab 10 Punkten: +0,3
  - Ab 18 Punkten: +0,7
  - Ab 26 Punkten: +1,0

# Einführung

- VLIW (Vex/Lx)
  - Architektur **eingeschränkt** vor Fertigung änderbar
  - Beispiele: Anzahl ALUs, Multiplizierer, Registerfelder
  - Schnittstellen und Kommunikation **fest**
- Konfigurierbare Prozessoren (Xtensa)
  - Architektur **feiner** vor Fertigung änderbar
  - Weglassen/Hinzunehmen von Funktionseinheiten
  - Kommunikationsstruktur **fest**
  - Ausschnittstellen **variabel**
- Rekonfigurierbare Prozessoren (S5000)
  - Architektur innerhalb gegebener Grenzen **erweiterbar**
  - Neue Funktionseinheiten hinzunehmbar
  - Ausschnittstellen und Kommunikation **fest**

- Statt 2 ALUs/1 MUL auf gleicher Fläche 4 MUL?
  - Bei rekonfigurierbarem Prozessor möglich
  - Bei anderen Technologien **nicht** möglich
- Statt 1x 128b breiten Speicher-Port 2x 64b breite Ports?
  - Auch bei rekonfigurierbarem Prozessor **nicht** möglich!
  - Würde bei **konfigurierbarem** gehen, da aber nur einmalig festlegbar

➡ Mehr rekonfigurierbare Flexibilität wünschenswert.

Praktisch begrenzt durch Flexibilität der Leiterplatte

- In der Regel nach Fertigung nur stark eingeschränkt variierbar

- Wie damit umgehen?
  - Wenn alles möglich ist: “Leeres Blatt”-Effekt
- Hilfreich: Zuerst Basiskonzepte entwickeln!

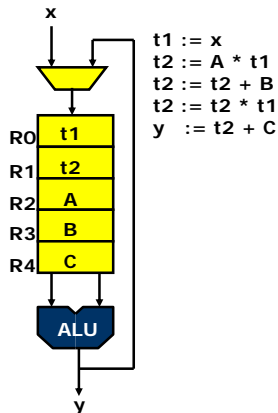
# Berechnungsmodelle



# Zeitliche Verteilung der Schritte

In allen bisher gezeigten Ansätze

$$y = A x^2 + B x + C$$



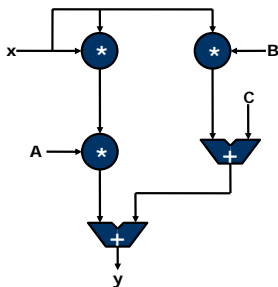
- Instruktionsgesteuerte Berechnung
- Im Extremfall: Eine Operation pro Zeitschritt
  - Aber auch mehrere ALUs möglich
  - Superskalar, TIE, EI
- **Gesteuert** durch **variable** Software
- Berechnungsuniversell

A. Koch

# Räumliche Verteilung der Schritte

Innerhalb von TIEs/EIs

$$y = A x^2 + B x + C$$



- Getrennte, dedizierte Recheneinheiten
- Mehrere Operationen pro Zeitschritt
- Innerhalb der Flächengrenzen beliebig parallel
- Gesteuert durch festes Steuerwerk
  - In TIEs/EIs i.d.R. nur **sehr einfache** Kontrollstrukturen
  - Kaum endliche Zustandsautomaten (FSMs)
- Berechnungsuniversell durch **Rekonfigurierbarkeit**

A. Koch

➡ So auch aufwendigere Berechnungen realisierbar?

- Räumliche Verteilung oft effizienter
    - Keine Instruktionen mehr, beliebig parallele Ausführung
  - Aber **nicht** für alle Arten von Operationen sinnvoll
    - Ausnahmebehandlung
      - Z.B. bei Fehler in Eingabedaten
    - Komplexe Bibliotheksfunktionen wie **printf**
      - Sehr komplizierter Ablauf
      - In der Regel nicht zeitkritisch
    - Bei räumlicher Verteilung müssen für **alle** Fälle dedizierte Rechenressourcen vorgehalten werden
      - Auch für nur **sehr selten** auftretende Fälle
- ➔ Dafür besser instruktionsgesteuerten Standardprozessor
- Verwendet Recheneinheiten **wieder**
    - Für unterschiedliche Aufgaben
    - Kein Verschwenden von "rechnender" Fläche

- Diverse Alternativen zwischen
  - Reiner zeitlichen Verteilung
  - Reiner räumlichen Verteilung
- Beispiele
  - VLIW
    - Mehrere Recheneinheiten pro Zeitschritt
    - Parallelität gesteuert durch Software
  - Flächenbegrenzte rekonfigurierbare Recheneinheiten
    - Wiederverwendung von Fläche durch Rekonfiguration
    - Wiederverwendung von Operatoren
    - Nicht-gepipelinte sequentielle Operatoren
    - Reduzierung der Parallelität aus Platzgründen

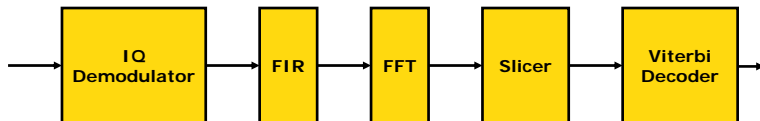
- Kombinieren
  - **Programmierbarkeit**
    - Prozessor mit fester Struktur
  - **Rekonfigurierbarkeit**
    - Recheneinheiten mit variabler Struktur
    - Reconfigurable Compute Unit (RCU)
- Ziel: Hohe Rechenleistung bei hoher Effizienz
- Realisiert in rekonfigurierbarem Prozessor S5000
- Unterschied ACS zu rekonfigurierbaren Prozessoren
  - **Mehr** Freiheitsgrade im ACS
  - Variabilität auch auf System-Ebene
    - Beispiel: Speicher-Ports
  - Auch **komplexe** Berechnungen rein auf RCU
  - Ohne Intervention des software-programmierten Prozessors (SPP)

# ACS Erfolge

- Früher Erfolg: Vergleich von Gensequenzen
  - 1993: SPLASH-2 schlägt MasPar MP-1 um 1300x
- Kryptographie
  - 1999: IDEA auf ACS 12x schneller als CPU, 1.4x als ASIC
  - 2001: Weltrekord in RSA Entschlüsselung (600 Kb/s)
  - 2001: DES-Verschlüsselung 2x ASIC (13.3 Gb/s)
- Digitale Signalverarbeitung (DSP)
  - Praktisch 10x-1000x schneller als DSPs
  - Beispiel
    - FPGA @ 250 MHz: 56 GMACs/s
    - DSP @ 600 MHz: 4.8 GMACs/s

# Kostenbetrachtung

## Empfänger mit Orthogonal Frequency Division Multiplexing



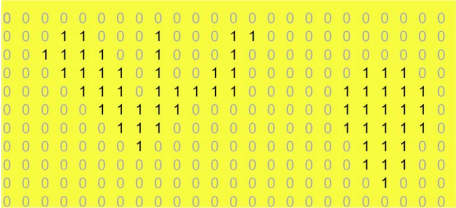
- DSP @ 300 MHz
  - 4 DSPs **pro Kanal** erforderlich: ca. USD 500/Kanal
- FPGA als RCU
  - Schafft mehr als 12 Kanäle **pro Chip**: ca. USD 10/Kanal



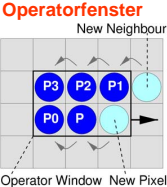
# Regionserkennung

Problem und Vorgehen

A. Koch



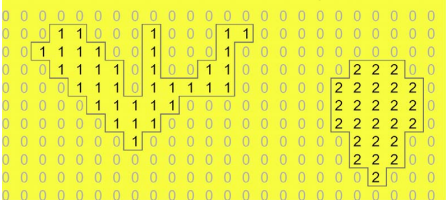
Schwarz/Weiß-Bild



Matrix aus Region IDs

Zusammenhangstabelle

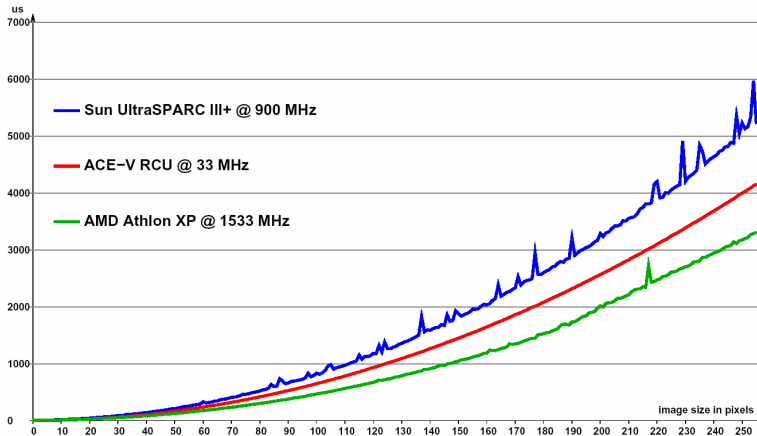
Object ID	Adjacent to
1	1
2	1
3	2
4	4



# Regionserkennung

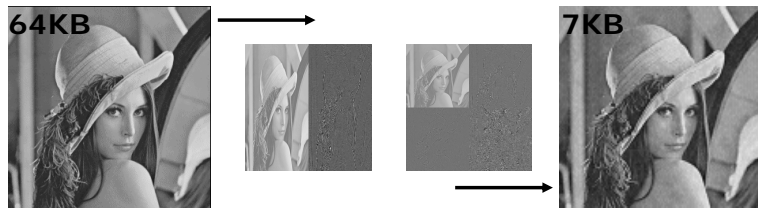
## Rechenleistung

A. Koch

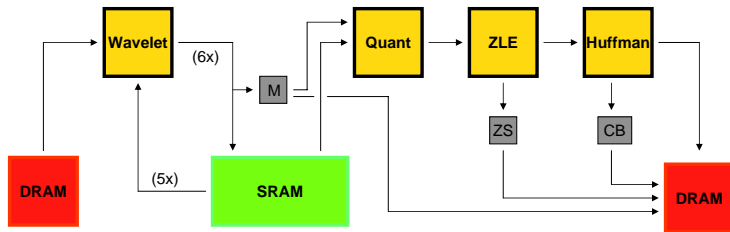


# Wavelet-Bildkompression

## Problem und Vorgehen

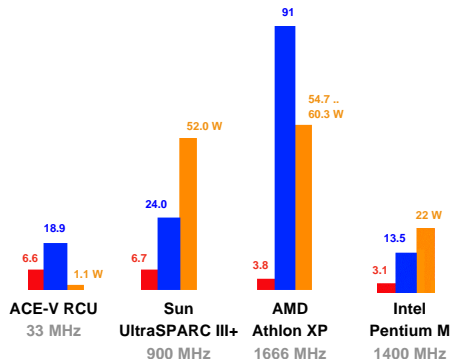


A. Koch



# Wavelet-Bildkompression

## Rechenleistung



Compression Execution  
Times incl. Data Transfer in  
ms:

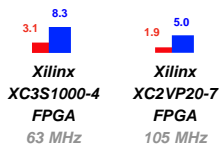
■ L = 256

■ L = 512

Power Consumption in  
Watts:

■ L = 256 and 512

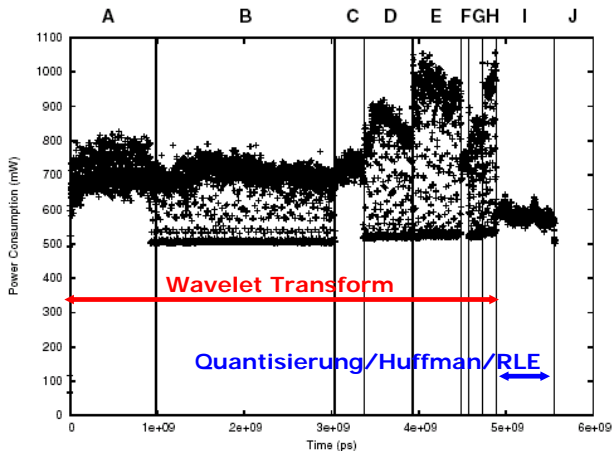
- *Simulated* -



A. Koch

# Wavelet-Bildkompression

## Leistungsaufnahme



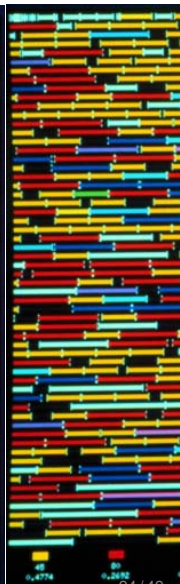
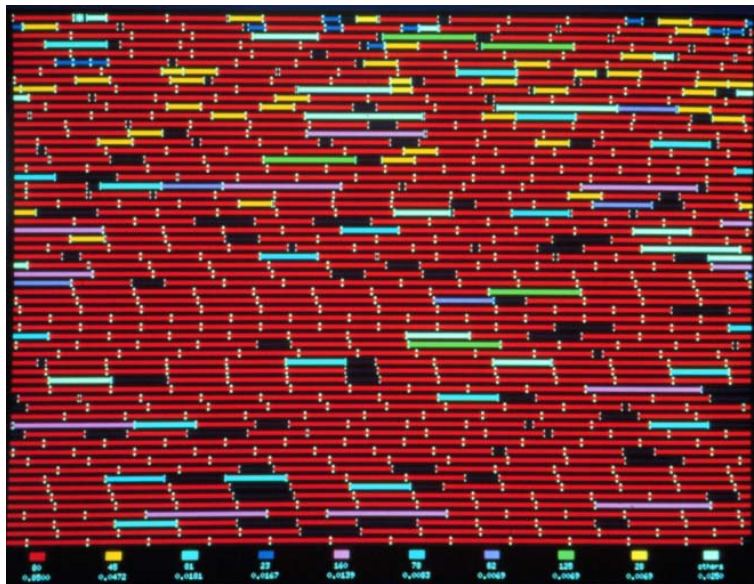
A. Koch

- ACS-Ansatz lohnend
- Insbesondere im Bereich eingebetteter Systeme

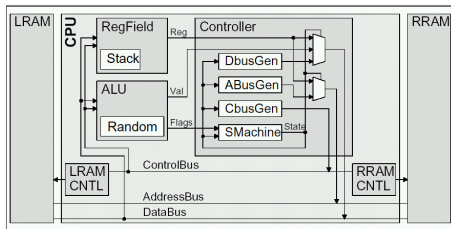
- Untersuchung von Populationen
  - Konkurrenz und Koexistenz
  - Populationsökologie
    - Räuber/Beuteverhältnis
  - Artenvielfalt
    - Interaktion zwischen Arten
  - Entwicklung von
    - Wirten
    - Parasiten
    - Hyper-Parasiten
- Idee: Nachbildung im Rechner
  - Beschleunigte Simulation
  - “Artificial Life”

- Softwaresimulation mittels MIMD-Prozessor
  - Organismen sind parallele Programme
- Spezielle Prozessorfähigkeiten erlauben
  - Mutation
    - Änderung der Instruktionsemantik
  - Rekombination
    - Austausch von Code zwischen Programmen
  - Tolerieren fehlerhafter Instruktionen
    - “Erkrankung”
  - Abbruch bei zu vielen Fehlern
    - “Tod”

# Beispiel für Evolution





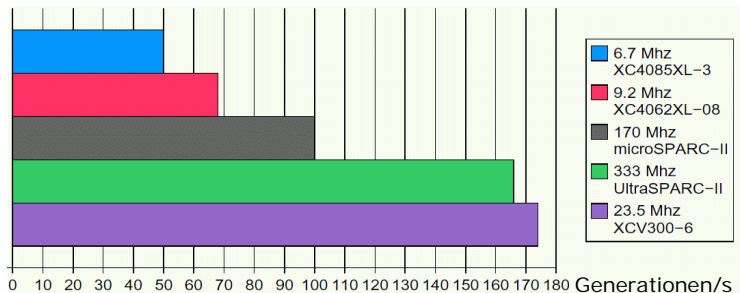


A. Koch

- x86-ähnlich
- Spezialfunktionen, z.B.
  - Mutation
  - Marker als Sprungziel
- OS-Dienste in Hardware
  - Prozessverwaltung
  - Speicherverwaltung
- RISC-Ansatz zu groß
  - Komplexer Microcode: Mikro und Nanoinstruktionen

# Rechenleistung des TIERRA-Prozessors

A. Koch

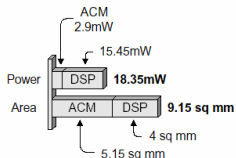
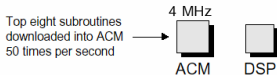
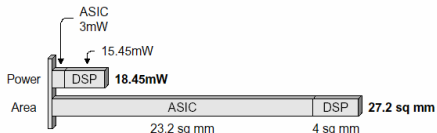
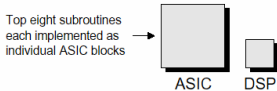
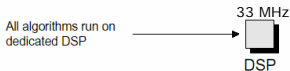


- Erfahrungswert:  
Leistungsaufnahme einer gut realisierten ACS-Anwendung liegt bei ca. 20% einer Software-Lösung gleicher Rechenleistung
- Geht aber häufig noch besser
  - Wavelet-Bildkomprimierung (1.1W statt 52W auf CPU)
  - Experimentelle Low-Power-FPGAs (BWRC LP\_PGAI):  
1/70 des Energieverbrauchs von kommerziellen Chips (vergleichbar: Xilinx XC4005XL)
  - VSELP Sprachkompression auf Maia rSoC:  
1/20 des Energieverbrauchs von Low-Power CPU (2.5V ARM8 @ 120 MHz)

# Leistungsaufnahme

## QCELP-Sprachkompression auf QuickSilver ACM

A. Koch



- Video-Bearbeitung
  - Entrauschen und Farbkorrektur in 2048x1536
- Multi-Standard-Radio (DAB, DRM, ...)
- Radar in verschiedenen Betriebsarten
- Echtzeit-Röntgenvisualisierung
- Adaptive OFDM-Filter
  - Verwende simpleren Filter bei gutem Empfang
- Multi-Standard verschlüsselter Funk
- Cognitive Radio
  - Keine feste Frequenzvergabe mehr, Partner suchen selber Lücken im Spektrum

➔ Nicht mehr nur akademische Luftschlösser

# ACS-Komponenten

- Einfacher instruktionsgesteuerter Prozessor als CPU
  - Liefert **Grundlast** der Rechenleistung
  - Systemsteuerung und langsames I/O
  - Führt Abläufe ungeeignet für räumliche Verteilung aus
- Flexible RCU
  - Liefert **Spitzenlast** der Rechenleistung
  - Führt Abläufe geeignet für räumliche Verteilung aus
  - Kann auch komplexe Operationen selbständig durchführen
- Speicher
  - Gemeinsamer Hauptspeicher für CPU und RCU
    - RCU kann **eigenständig** auf Speicher zugreifen
  - Lokaler Speicher für RCU (optional)
- Spezialschnittstellen an RCU (optional)
  - Hochgeschwindigkeits-I/O

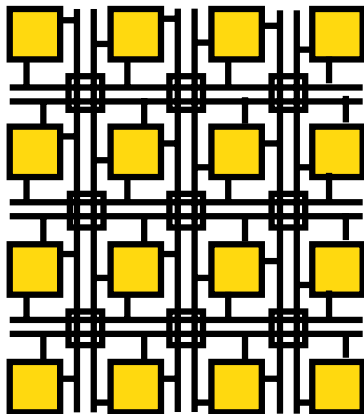
# Aufbau einer RCU



- Fundamente
  - Konfigurierbares Verbindungsnetzwerk
    - Programmierbare Schalter verbinden/trennen Leitungen
  - Konfigurierbare Funktionsblöcke
    - Berechnete Funktion kann je Block individuell programmiert werden
- Viele Variationsmöglichkeiten!

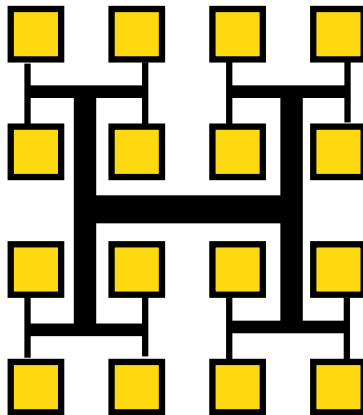
# Verbindungsnetz: Symmetrische Matrix

A. Koch



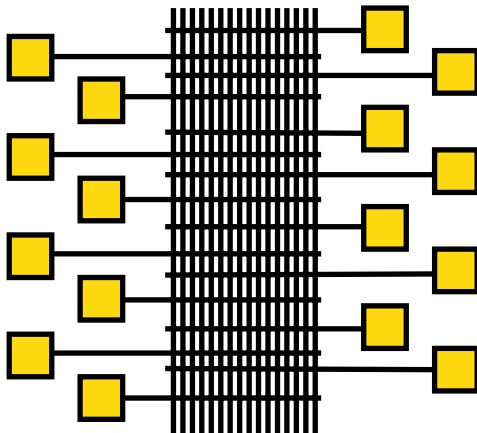
# Verbindungsnetz: Hierarchische Matrix

A. Koch



# Verbindungsnetz: Kreuzschienenverteiler

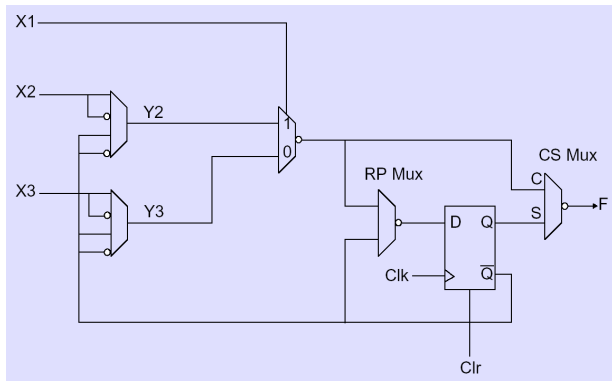
A. Koch



# Feingranularer Funktionsblock

Xilinx XC6200

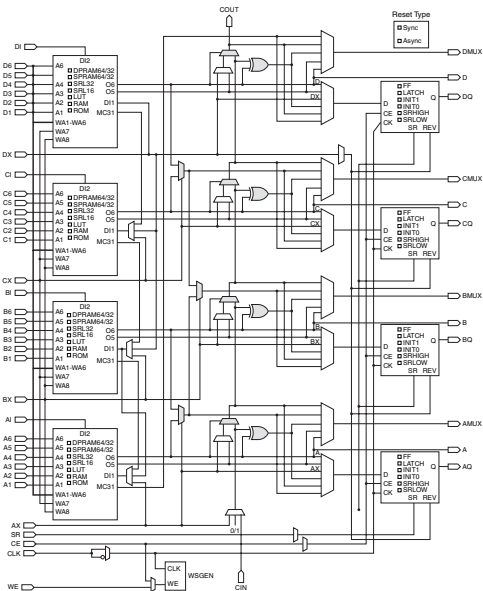
A. Koch



- Kann realisieren
  - **Alle** logischen Funktionen mit zwei Eingängen
  - **Einige** Funktionen mit drei Eingängen

# Mittelgranularer Funktionsblock

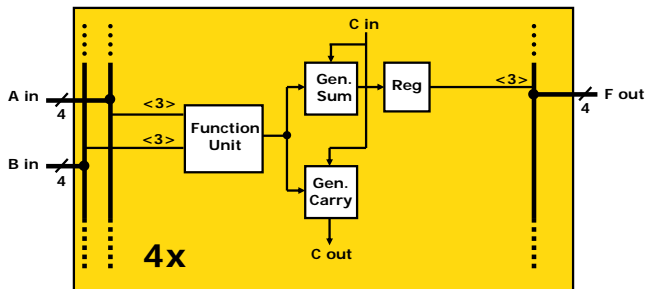
Xilinx Virtex 5: 4x Funktionen mit 6 Eingängen und Add/Sub/RAM/Shift



A. Koch

# Grobgranularer Block

HP Labs CHESS (dann Elixent D-Fabrix, nun bei Toshiba)



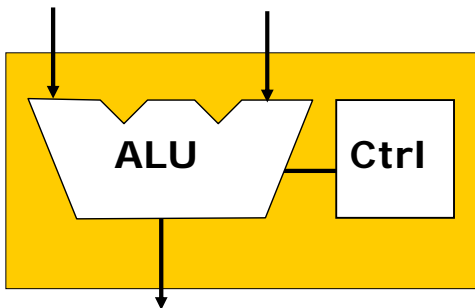
A. Koch

- 4b ALU
- Logikfunktionen und einfache Arithmetik (Add/Sub)
- Funktion kann durch anderen Block **zur Laufzeit** gesteuert werden
- Beispiel: JPEG-Kompressor braucht 512 Blöcke

# Sehr grobgranularer Funktionsblock

PACT XPP ALU Block

A. Koch

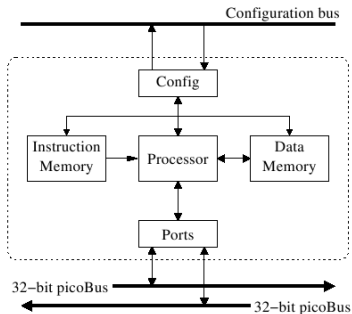


- Operanden: 24b oder (12b,12b) Worte
- Logische und arithmetische Funktionen
  - Einschliesslich Multiplikation
- Automatische Datensynchronisation
  - Datenfluß
  - Partielle Rekonfiguration zur Laufzeit



# Extremfall: Chip-level Multiprocessors (CMP)

PicoChip PC102: Spezialisiert auf Funknetze (WiMAX, WCDMA)

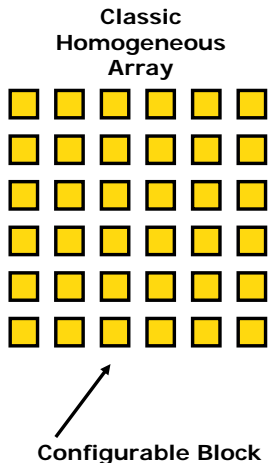


- **Komplette** Prozessoren als Funktionsblöcke

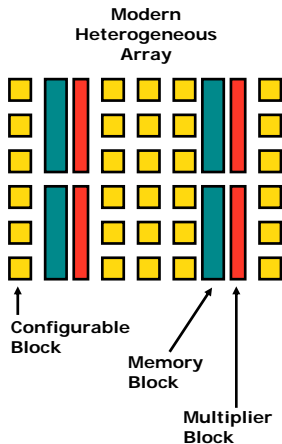
- 16b LIW mit 3 Operationen pro Instruktion
- Vier verschiedene Arten von Prozessoren
- 1KB... 32KB Speicher je Block

A. Koch

Processor Type	LIW Fields/Execution Units				
	LIW.0	LIW.1		LIW.2	
STANdard	ALU.0	Comms Unit	Memory Access Unit/ ALU.1	Branch Unit	Application Specific Unit
MAC	ALU.0	Comms Unit	Memory Access Unit/ ALU.1	Branch Unit	MAC Unit
MEMory/Control	ALU.0	Comms Unit	Memory Access Unit/ ALU.1	Branch Unit	Multiply Unit



- Traditionelle FPGAs sind **homogen**
- Haben nur eine Art von Funktionsblock
  - Möglicherweise aber verschiedene Betriebsarten
  - Z.B. Logik/RAM/Schieberegister
- Vorteile
  - Einfachere CAD-Werkzeuge
  - Einfacheres Chip-Layout
- Nachteile: Ineffizient, z.B. für
  - Multiplizierer
  - Größere Speicher



- Moderne Bausteine enthalten einen **Mix** von Blöcken

- Schnelle Multiplizierer
- Größere Speicherblöcke (insgesamt > 10Mb)
- Komplette Prozessoren
- Taktmanipulation (PLL/DLL)
- Spezielle I/O-Schnittstellen (3.2 Gb/s per pin)

- Vorteile

- Höhere Rechenleistung
- Bessere Flächeneffizienz
  - Wenn Spezialblöcke auch **benutzt** werden

- Nachteile

- CAD-Werkzeuge deutlich komplizierter

# Trends

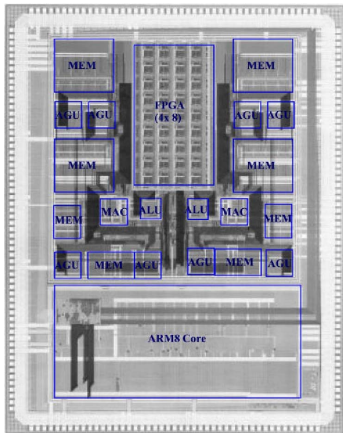
# System-FPGAs

Hauptziel: Höhere Integrationsdichte, weniger Bausteine im System

- **Größere** FPGAs
    - Xilinx XCV5LX330: > 200K Lookup-Tables mit je 6 Eingängen
  - **Mehr** integrierte Blöcke
    - Bis zu vier komplette 32b RISC-Prozessoren
    - Konfigurierbare Impedanz pro Pin ersetzt Terminierungswiderstände
  - Rekonfigurationszeit immer noch relativ **langsam**
    - 80 Mb Konfigurationsdaten, ladbar mit 32b @ 48 MHz:  
ca. **50 ms** Rekonfigurationszeit
- ➔ Nicht wirklich auf rekonfigurierbares Rechnen ausgelegt
- Seltene Umschaltung von Betriebsarten
  - “Soft-Hardware” Updates

# Rekonfigurierbare Systems-on-Chip (rSoC)

Spezialisiert auf bestimmtes Anwendungsgebiet



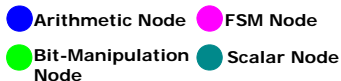
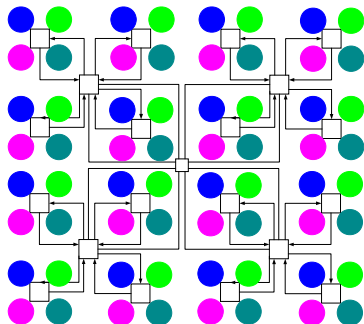
Berkeley Wireless  
Research Center **Maia**

- Stark **heterogener** Aufbau
- Kombiniert CPU mit RCU auf **selbem** Chip
- Aber **losere** Kopplung als TIE/EI
  - Keine Einbindung in Prozessor-Pipeline
- RCUs **schneller** rekonfigurierbar
  - 500us für 200K Gatter (M2000 FLEXEOS FPGA)
  - 33us für 128 ALUs (PACT XPP)

A. Koch

# Spezialisierte ACS-Bausteine

Quicksilver ACM



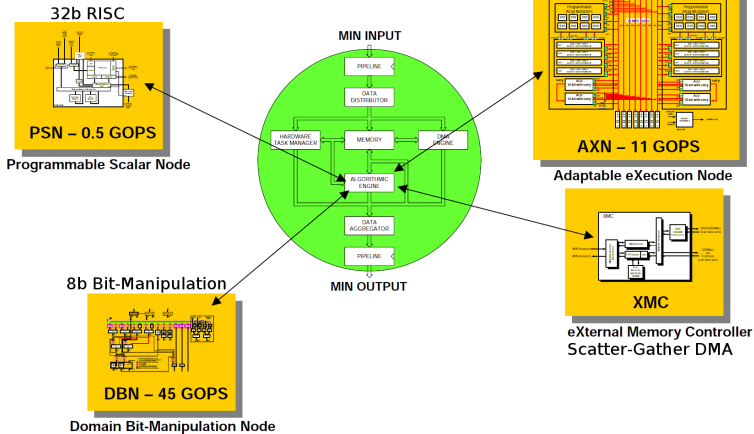
- **Allgemeine**  
Beschleunigung von Berechnungen
- Kombiniert verschiedene von Recheneinheiten
- Sehr niedrige Leistungsaufnahme
- In **einem Takt** rekonfigurierbar
- Beispiel:  
57K Rekonfigurationen/s für CDMA2000 Rake Finger

A. Koch

# Funktionsblöcke der Quicksilver ACM

Alle Arten von Blöcken haben gleiche Schnittstellen zum Verbindungsnetz

A. Koch



Konzept ist leider gescheitert (Firma bankrott)



- Idee adaptiver Computersysteme
- Berechnungsmodelle
- Erfolge
- ACS-Komponenten
- Aufbau einer RCU
  - Verbindungsnetz
  - Funktionsblöcke
- Trends

➔ Weiter mit

- Systemarchitektur
- Programmierung
- Programmierwerkzeugen