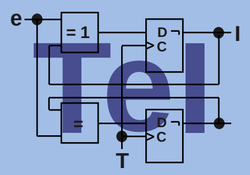
Dynamische Architekturadaption von Hardware-Agentensystemen

Marcel Naggatz
Informationssystemtechnik
Matrikel.-Nr: 2917875

marcelnaggatz@hotmail.de



Institut für Technische Informatik
http://www.inf.tu-dresden.de/TeI/

30.01.2008

Inhaltsverzeichnis

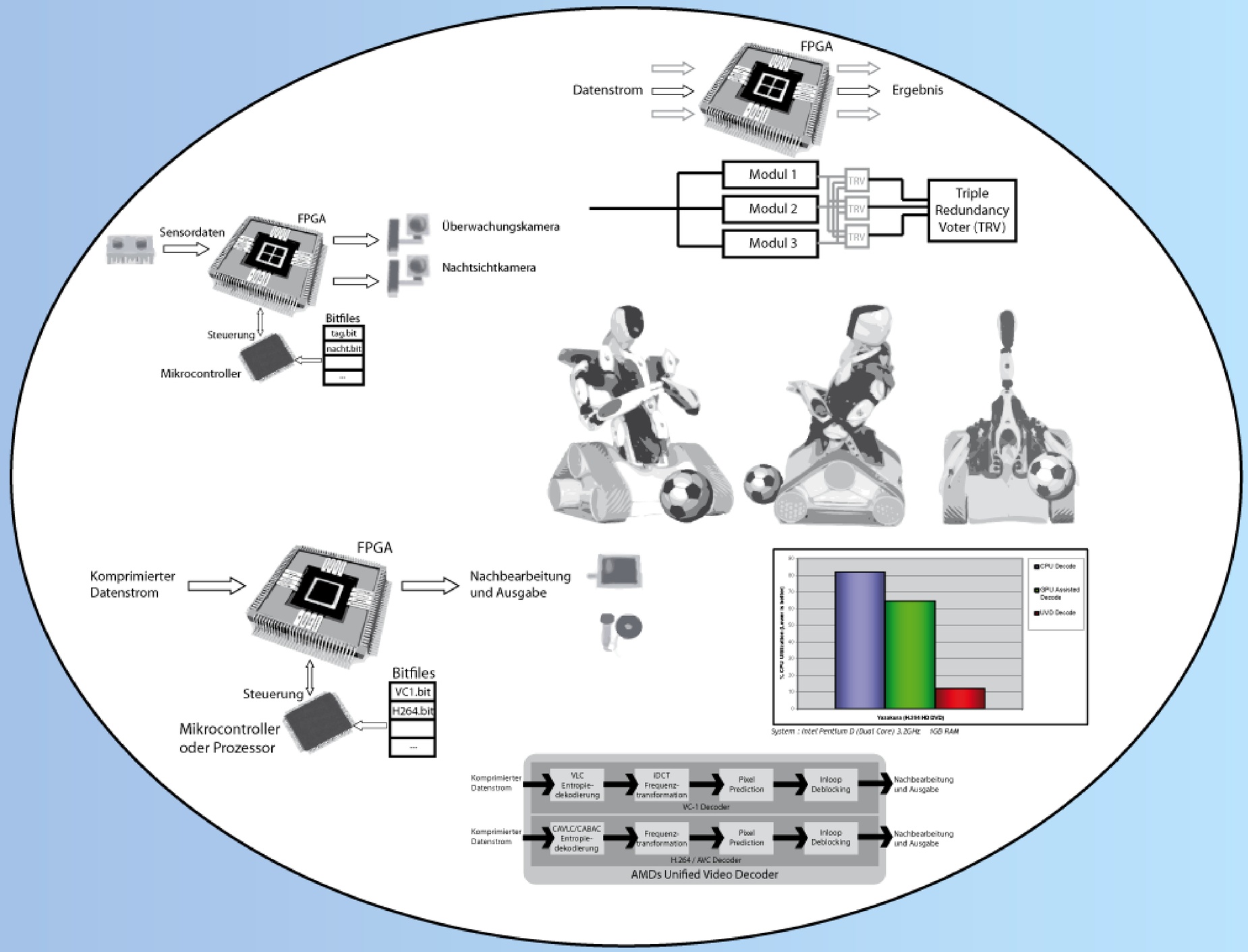
1. Aufgabe

Aufgabe der Diplomarbeit

Ausgehend von einer gegebenen statischen HW-Agentenarchitektur ist diese so zu ergänzen, dass eine Laufzeitanpassung von HW-Agenten auf FPGAs möglich ist. Zur Lösung der Aufgabe sind folgende Schwerpunkte zu bearbeiten:

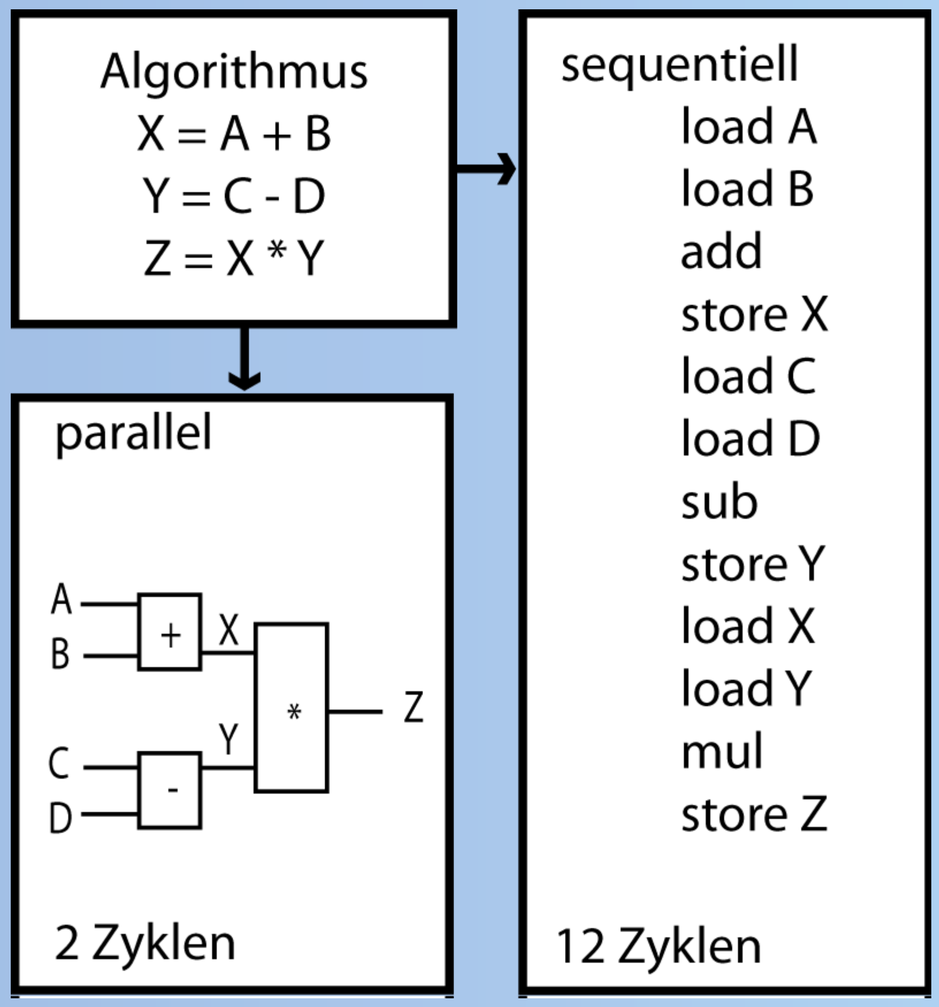
1. Einarbeitung in die Aufgabenstellung und Literaturstudium zu Möglichkeiten der adaptiven Anpassung von Agentensystemen auf FPGAs und deren Anwendungs- und Einsatzmöglichkeiten.
2. Konzeption einer adaptiven HW-Agentenarchitektur und Zusammenstellung notwendiger SW-Werkzeuge und Entwurfsabläufe.
3. Entwurf einer adaptiven HW-Agentenarchitektur und Entwicklung zugehöriger Systemschnittstellen .
4. Erstellung von Anwendungsszenarien zur Validation des entwickelten Architekturansatzes und deren Test.
5. Untersuchungen zu Anwendungsszenarien und Darstellung der Ergebnisse. Dokumentation der Implementierungen.

Anwendungsgebiete

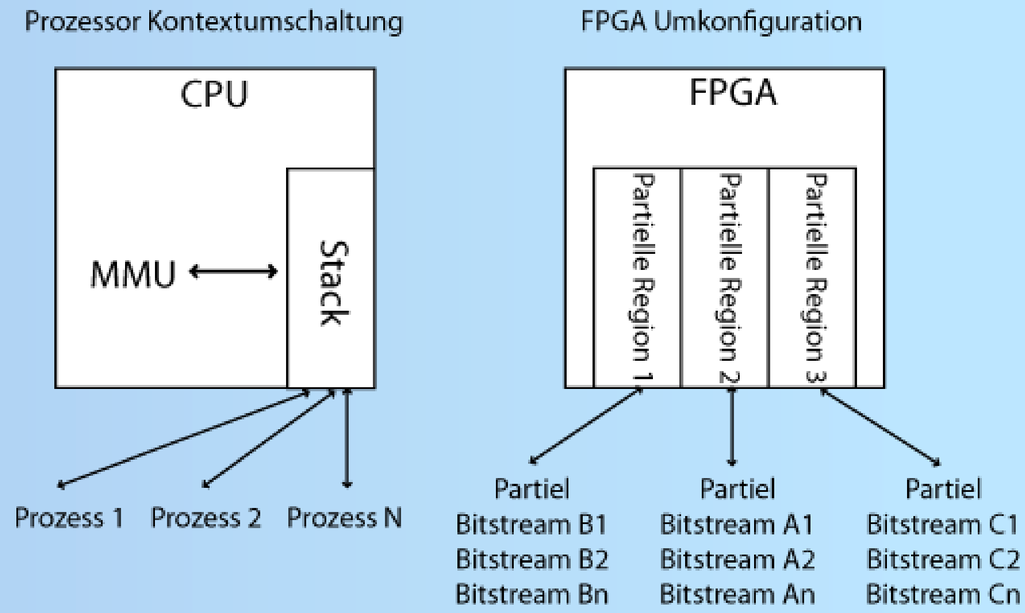


2. Konzepte und Grundideen

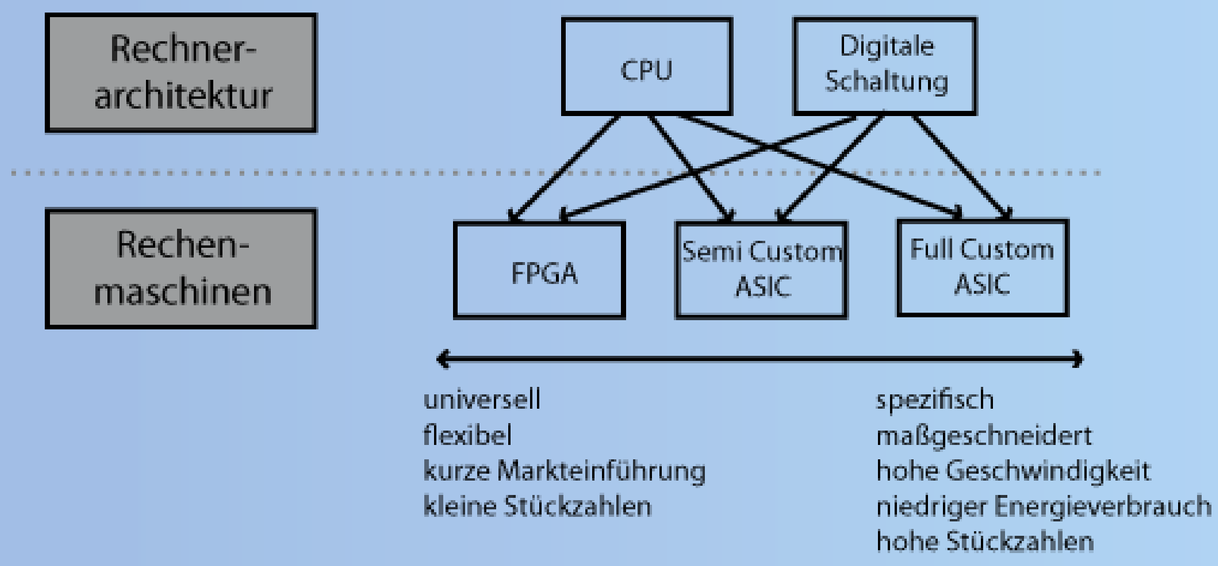
Algorithmen und Rechnerarchitektur

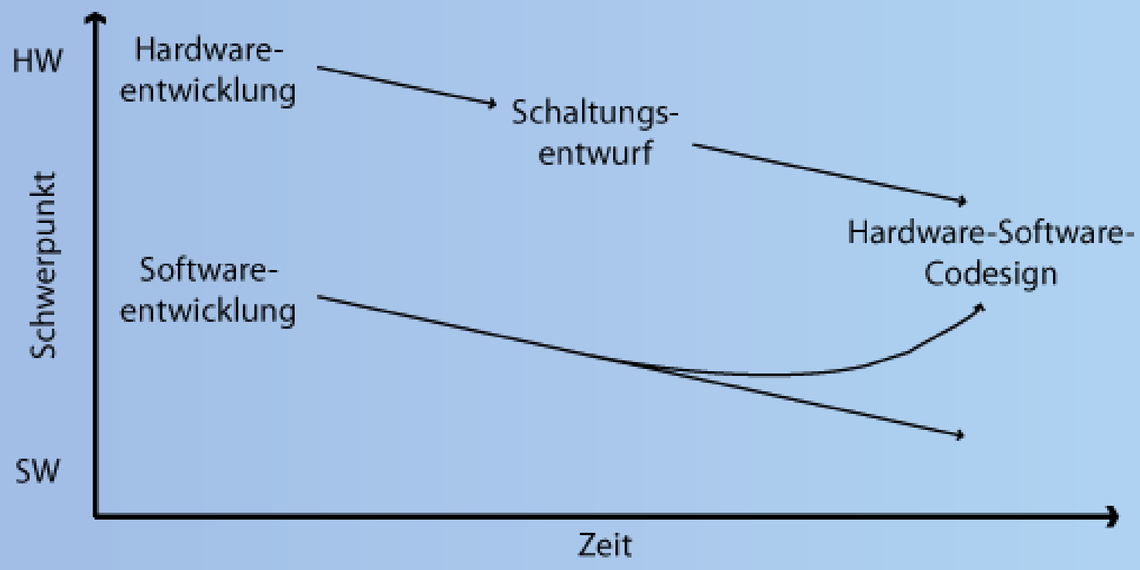


- Ein parallelisierbarer Algorithmus kann parallel oder sequenziell ausgeführt werden
- Die Rechnerarchitektur bestimmt die Ausführung des Algorithmus
- Mikroprozessor arbeitet sequenziell
- FPGA kann Aufgaben sehr gut parallel ausführen



Software- und Hardwareentwicklung

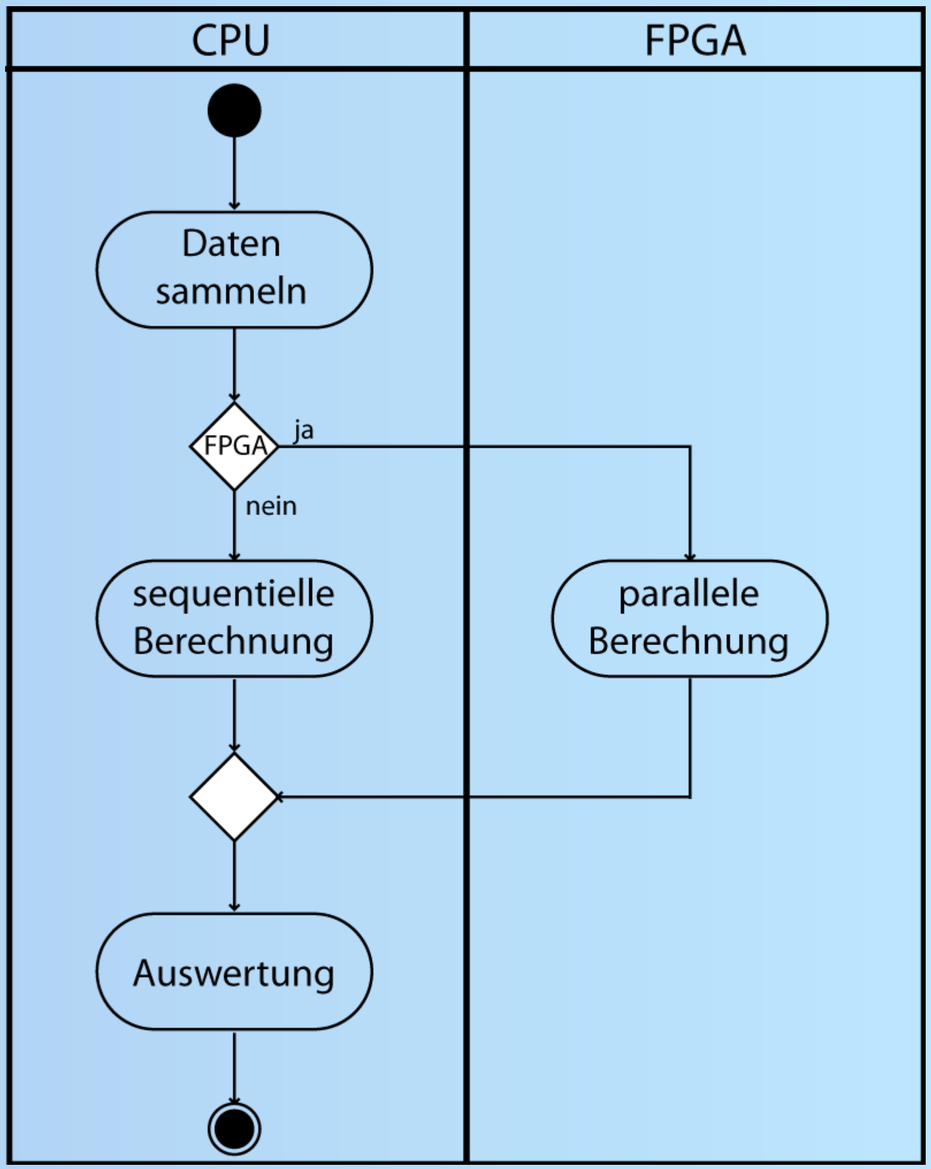




- Ein Rechenmaschine besteht aus einer Rechnerarchitektur, die auf einem Rechenbaustein abgebildet wird
- Rechnerarchitektur und Rechenmaschine sind in der Zuordnung unabhängig voneinander
- Der Entwicklung von Software und Hardware verlagert sich zur Software
- Beide Verfahren könnten sich im Hardware-Software-Codesign treffen

Hardware-Software-Codesign

- Der Begriff suggeriert einen gemeinsamen Entwurf von Hardware und Software
- Hardware wird jedoch nicht entworfen und anschließend hergestellt, sondern es wird eine Hardware programmiert (z.B. VHDL, Verilog)
- Gemeinsam entworfen wird eine digitale Schaltung (für FPGAs) und Software (z.B. für Mikroprozessoren) – in beiden Fällen also Software
- Software läuft auf FPGAs, Mikroprozessoren, oder gemischten Mikroprozessor-FPGA-Systemen

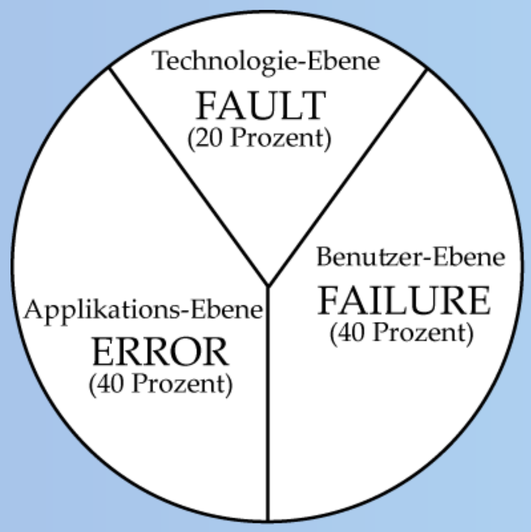


Fehler einer Digitalen Schaltung

Fehlertypen:

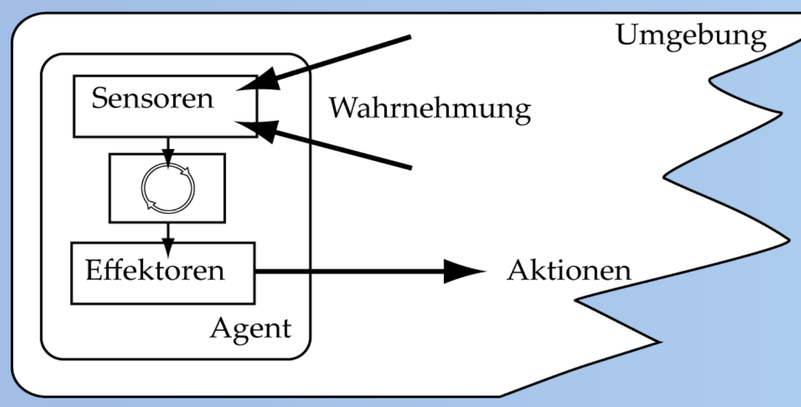
- Transient - Vorübergehender Fehler, z. B. Speicherbit kippt wegen Strahlung
- Permanent - z.B. ständig unterbrochene Leitung durch Herstellungsfehler, oder entstanden aus transientem Fehler durch wiederholte gezielte Fehlereinwirkung
- Intermittierend - ein in regelmäßigen Abständen wiederkehrender Fehler

Fehlerbegriffe und Fehlerhäufigkeit:



- FAULT : physikalischer Defekt
- ERROR : Auswirkungen des physikalischer Defekts, Bugs
- FAILURE : Fehlverhalten des Systems, Ausfall

Agenten



- Ein Agent muss vier Haupteigenschaften erfüllen
  - Autonomie
  - Reaktivität
  - Interaktivität
  - Proaktivität
- Zur genauen Spezifikation gibt es weitere Eigenschaft
  - Kontinuität
  - Adaptivität
  - Mobilität
  - Flexibilität
  - Charakter

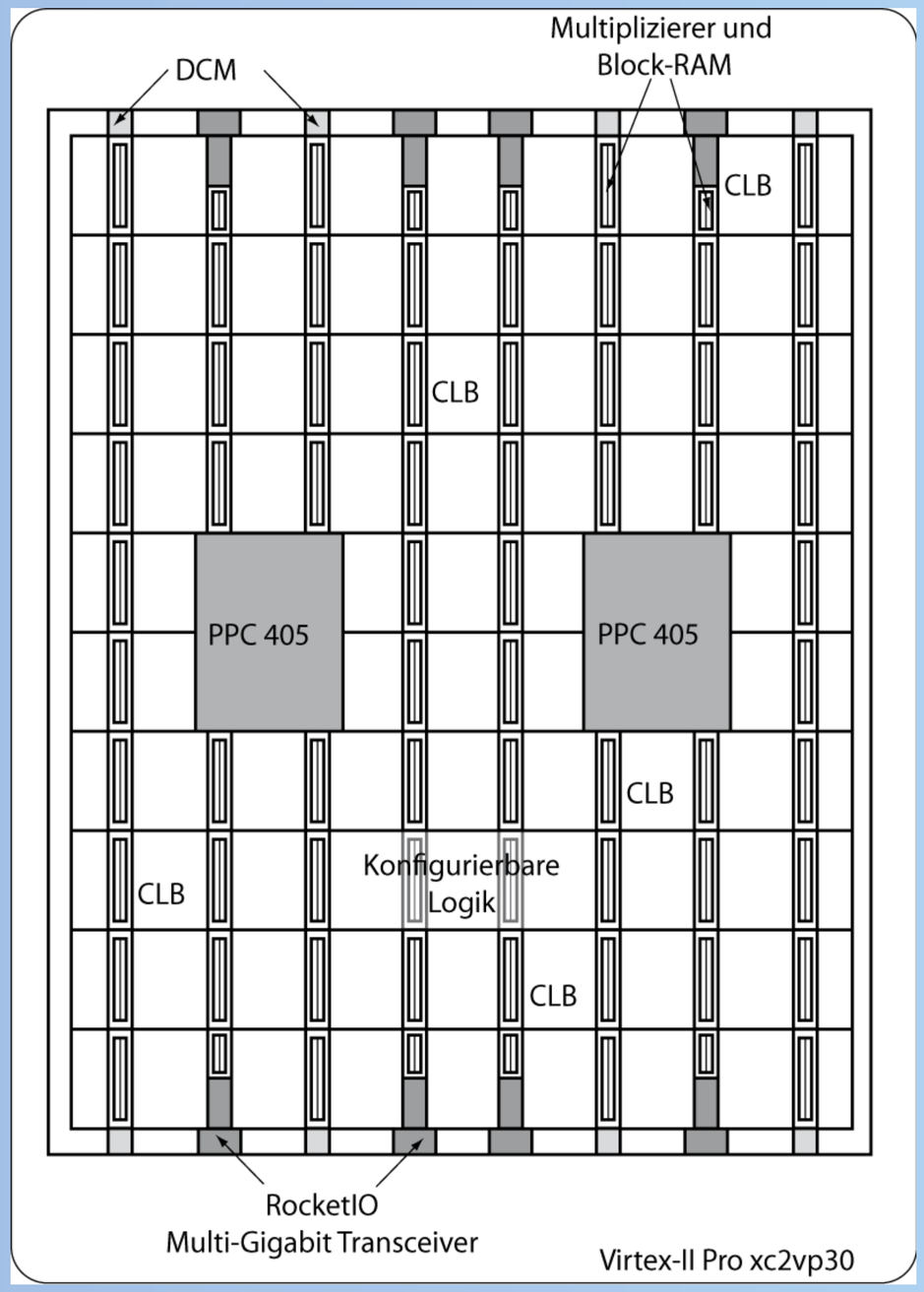
Voraussetzungen einer Hardware-Agentenarchitektur

- Es muss eine modulare, partiell und dynamisch rekonfigurierbare FPGA-Architektur vorhanden sein
- Zur Konfiguration und Verteilung der Aufgaben muss ein Mikrocontroller oder Mikroprozessor an den FPGA angeschlossen werden (Hardware-Task-Manager)
- Es muss eine schnelle Konfiguration (SelectMap) und eine optimale Verbindung (serielle Verbindung, Bußsystem) vorliegen

Aufbauend auf diesen Voraussetzungen kann eine Hardware-Agentenarchitektur entwickelt werden

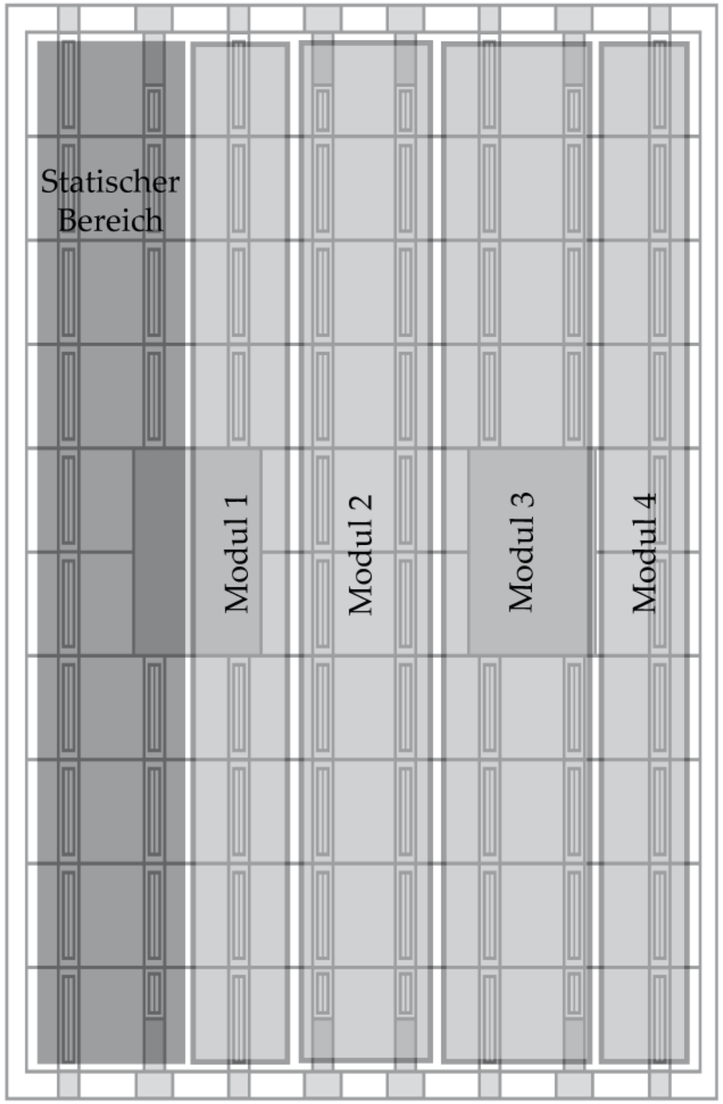
3. Entwurf einer modularen Architekur

Partielle Rekonfiguration

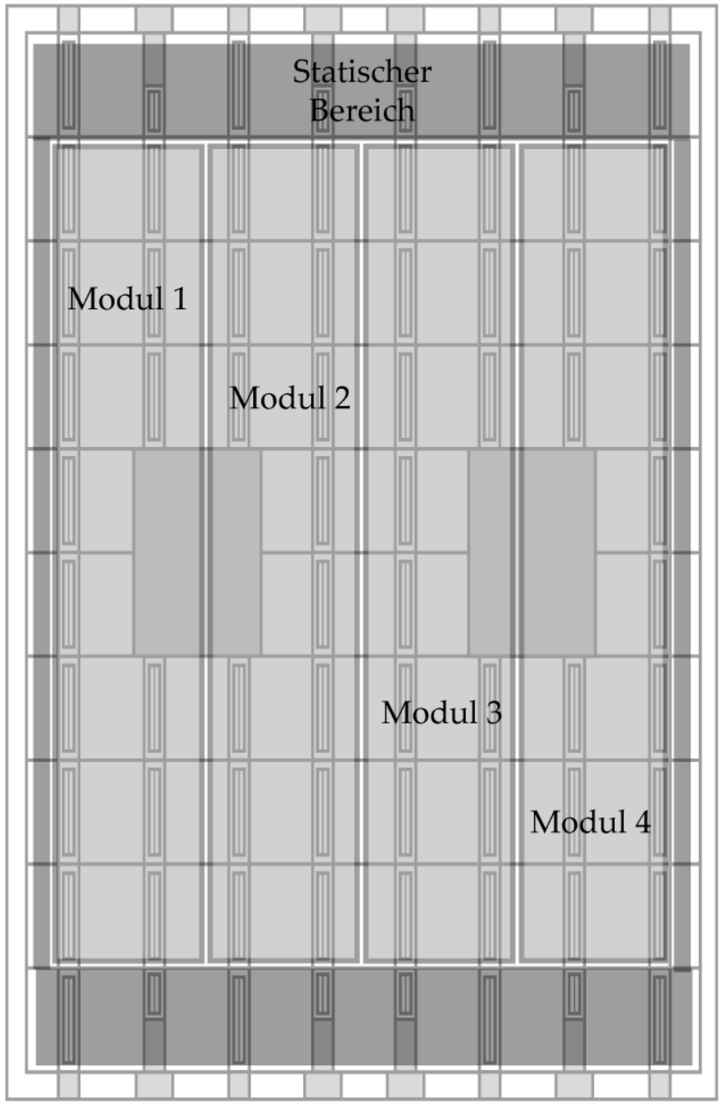


- Virtex-II Pro FPGA
  - 30 000 Gatterelemente
  - 180 HW-Multiplizierer
  - 180 BlockRAM
  - 8 Digital Clock Manager
  - Konfiguration mittels JTAG(USB), SelectMap(PROM, CompactFLash), Parallel-JTAG
  - 2 PowerPC Einheiten
  - Partiell Konfigurierbare Bereiche müssen sich über 16 CLBs erstrecken

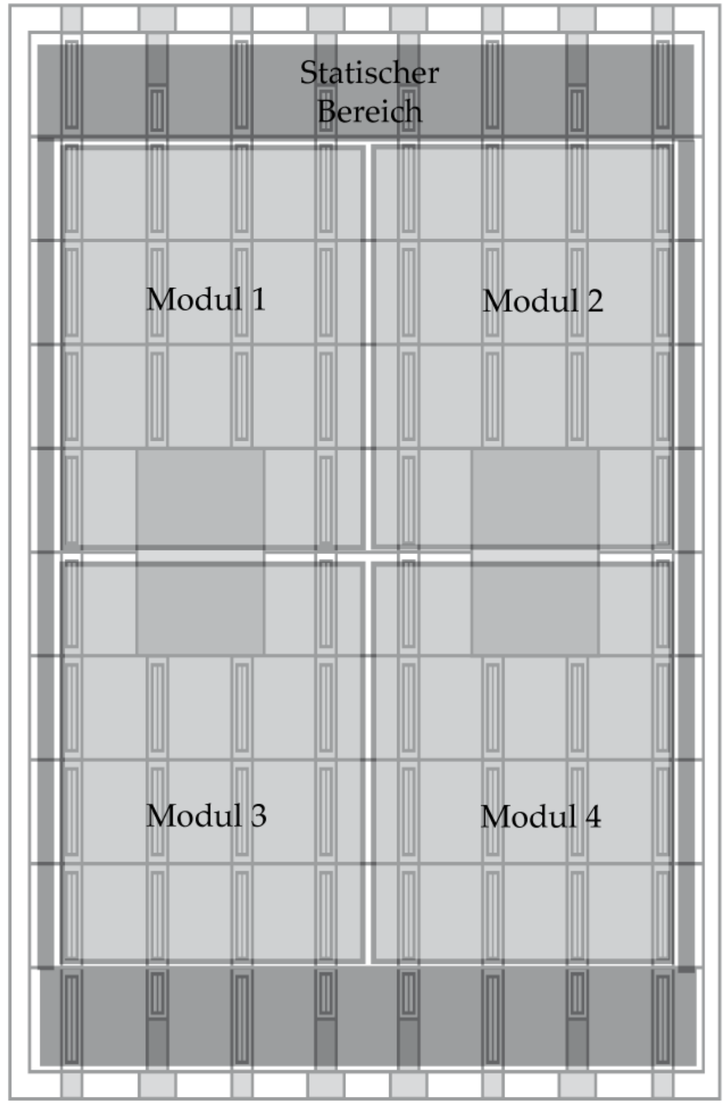
Partielle Rekonfiguration



a. Spaltenweise Rekonfiguration - bisherige eingeschränkte Platzierung der statischen und der rekonfigurierbaren Bereiche



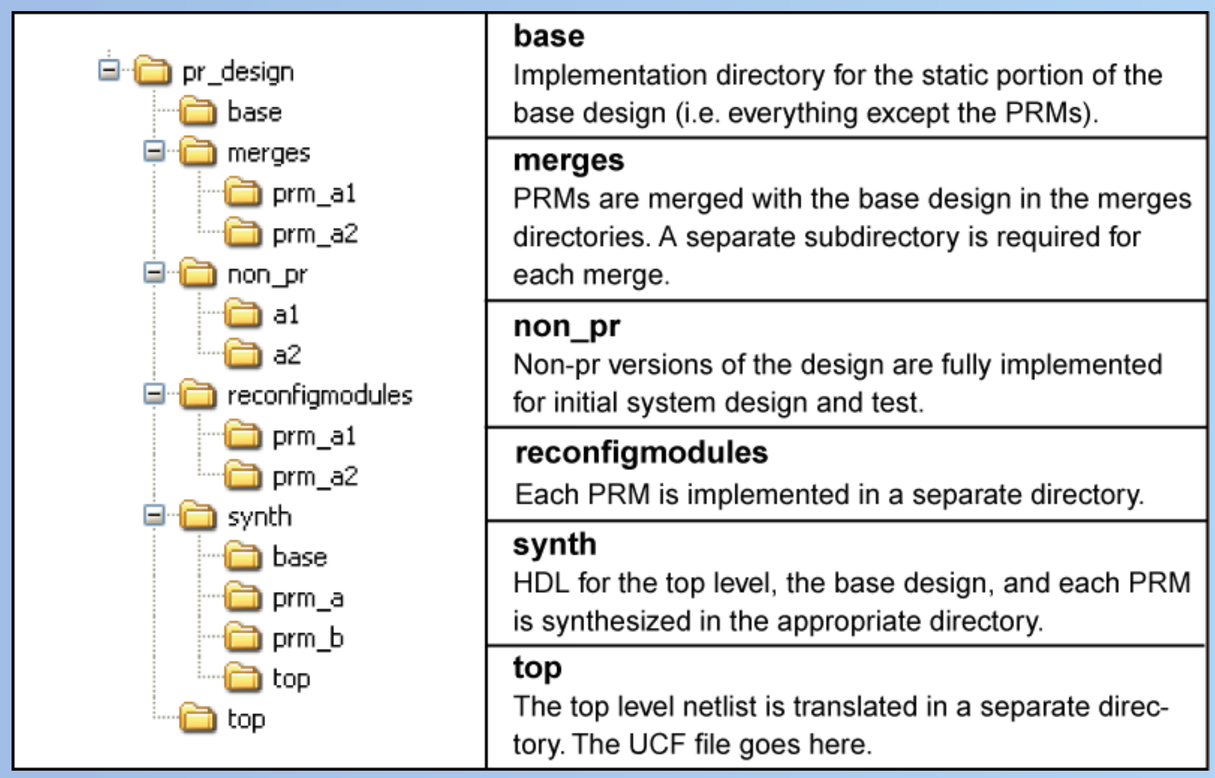
b. Neue Rekonfiguration - ermöglicht für Virtex 2 - FPGAs eine uneingeschränktere Platzierung der einzelnen Bereiche - nur in Verbindung mit neuem Design-Flow

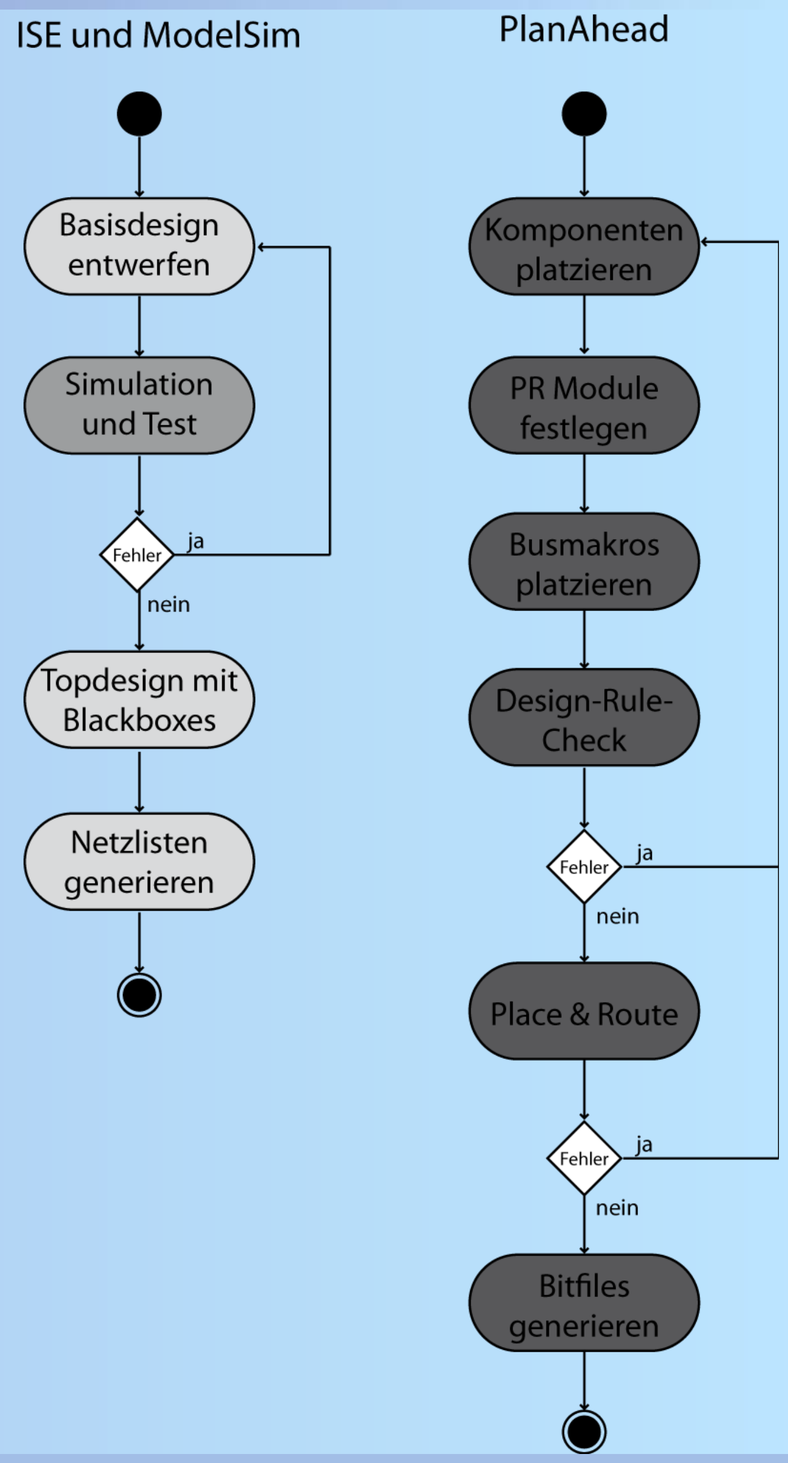


c. Wahlfreie Rekonfiguration - ermöglicht eine freie Platzierung der rekonfigurierbaren Module, nur für Virtex 4 und Virtex 5 - FPGAs möglich

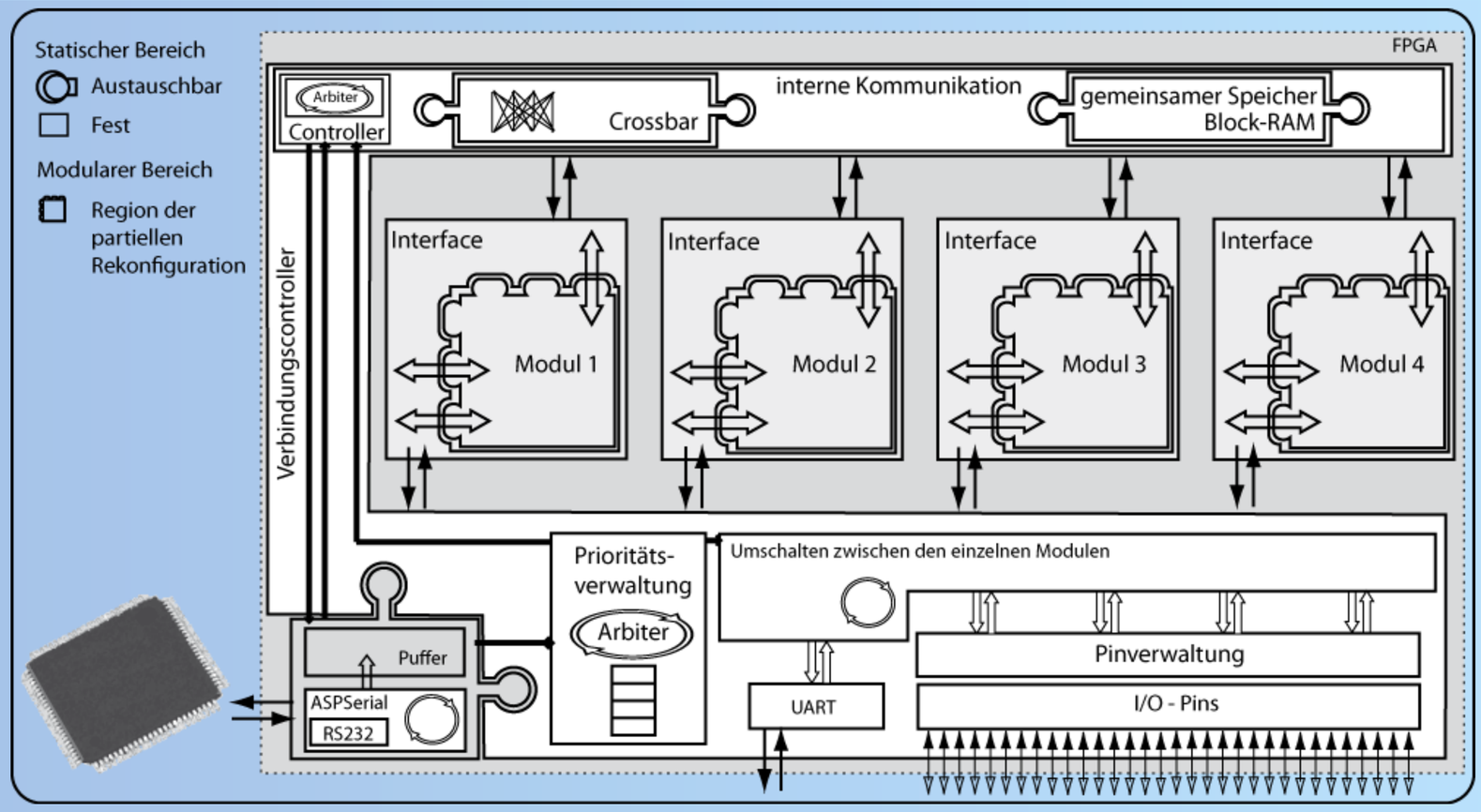
Partielle Rekonfiguration

- Verwendete Software
  - Xilinx ISE 9.1i und PlanAhead 9.2.5
  - ModelSim 6.2b
  - TCL-Script zur Verwaltung der Ordnerhierarchie
  - Impact zum konfigurieren der FPGA





FPGA - Architektur

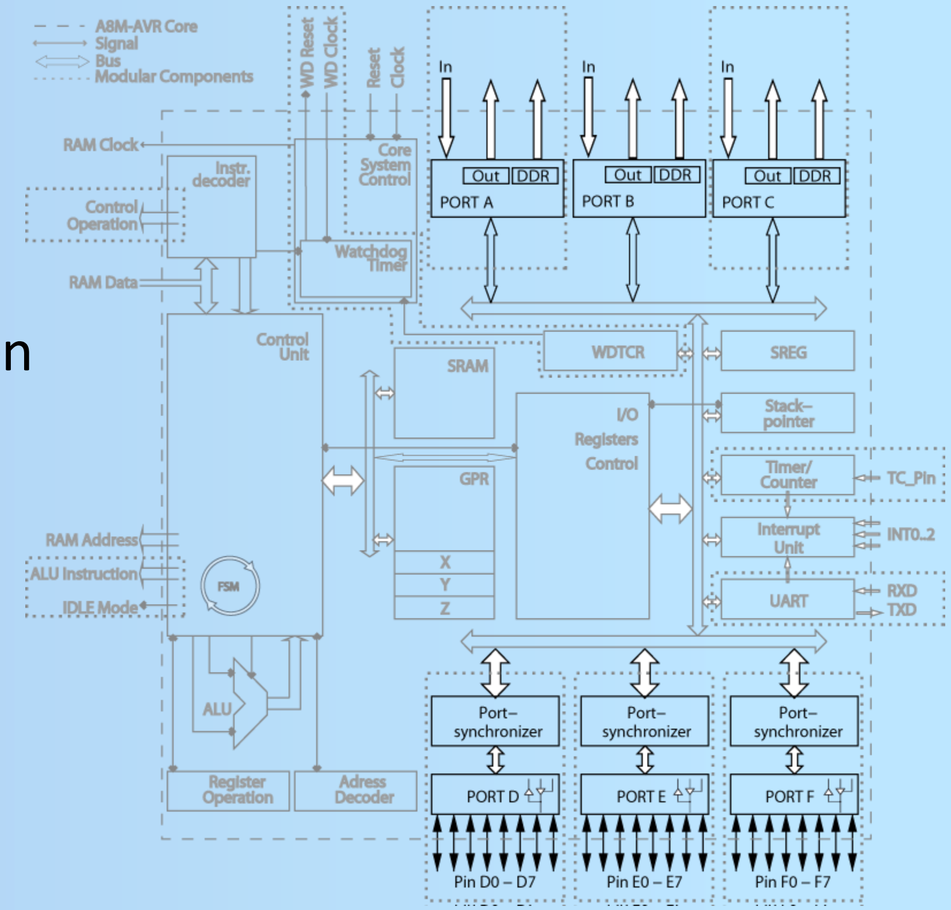


- Statische Bereich mir variabler Anbindung der I/O-Ports (einfache Pins, UART, etc.)
- Verbindung zum Mikrocontroller und PC zum Test und Debuggen (realisiert durch serielle Verbindung) – Austauschbar, z.B. schnelles Bußsystem
- Interne Kommunikationsstrukturen (Verbindungsnetzwerk und Gemeinsamer Speicher) – Austauschbar, z.B. Crossbar durch Omeganetz
- Bis zu 16 partiell rekonfigurierbare Bereiche
- Einheitliches Interface zwischen den Modulen und dem statischen Bereich
- Prioritätsverwaltung zur Bestimmung der Serviceprozessoren

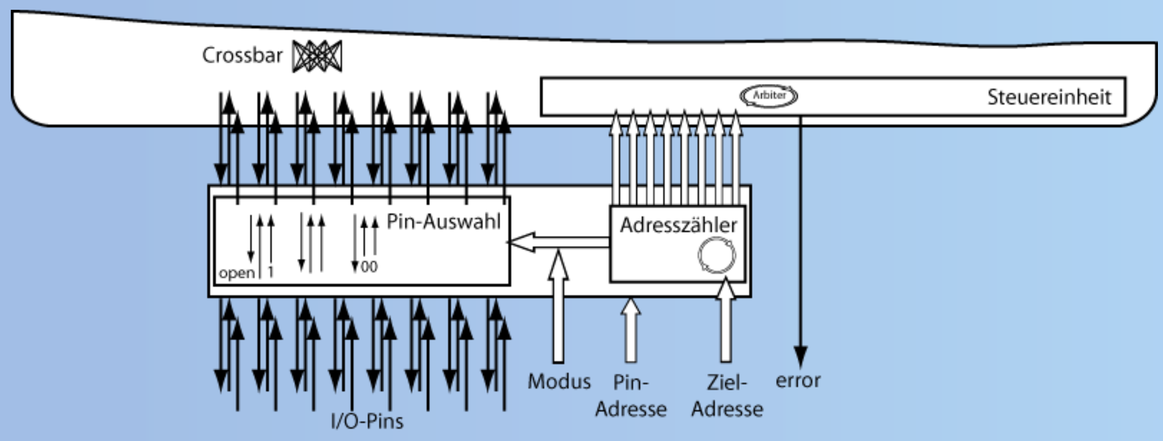
Modulsteuerung

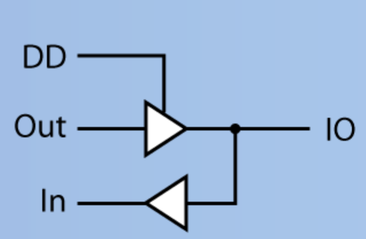
Zielstellung:

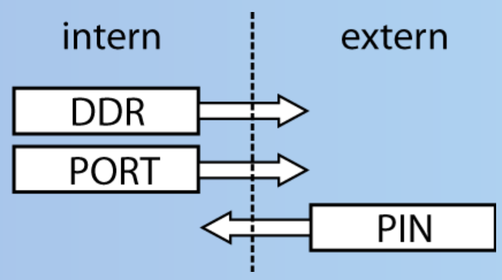
- Ein Port des A8M-AVR soll alles steuern können
  - I/O-Ports konfigurieren
  - Zugriff auf gemeinsamen Speicher (lesen und schreiben)
  - Zugriff auf Verbindungsnetzwerk (Ausgangs-FIFO beschreiben, Eingangs-FIFO lesen)
- Statusabfrage bezüglich vergebener Priorität
  - 4 Prioritätsstufen
    - 0 - normales Arbeitsmodul (beliebige Anzahl)
    - 1 - interne Stufe zum Wechseln der Prioritäten
    - 2 - stellvertretender Serviceprozessor
    - 3 - Serviceprozessor



I/O-Pins





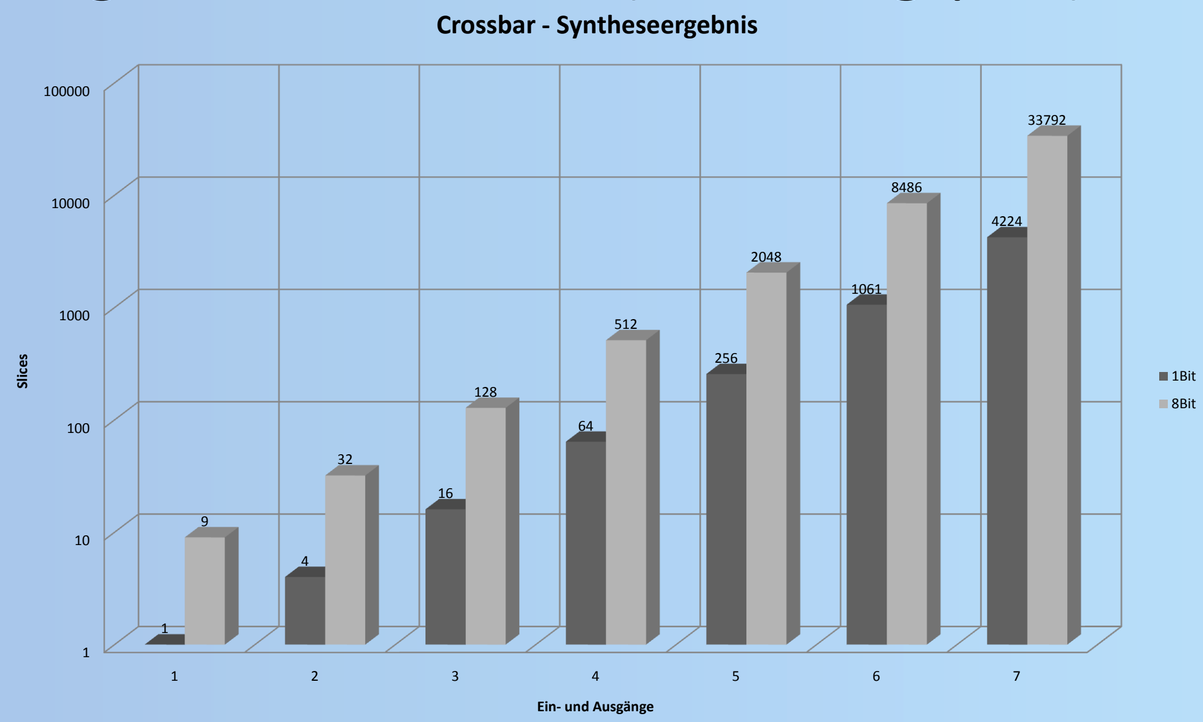


3 Modi zur Verbindung des Moduls

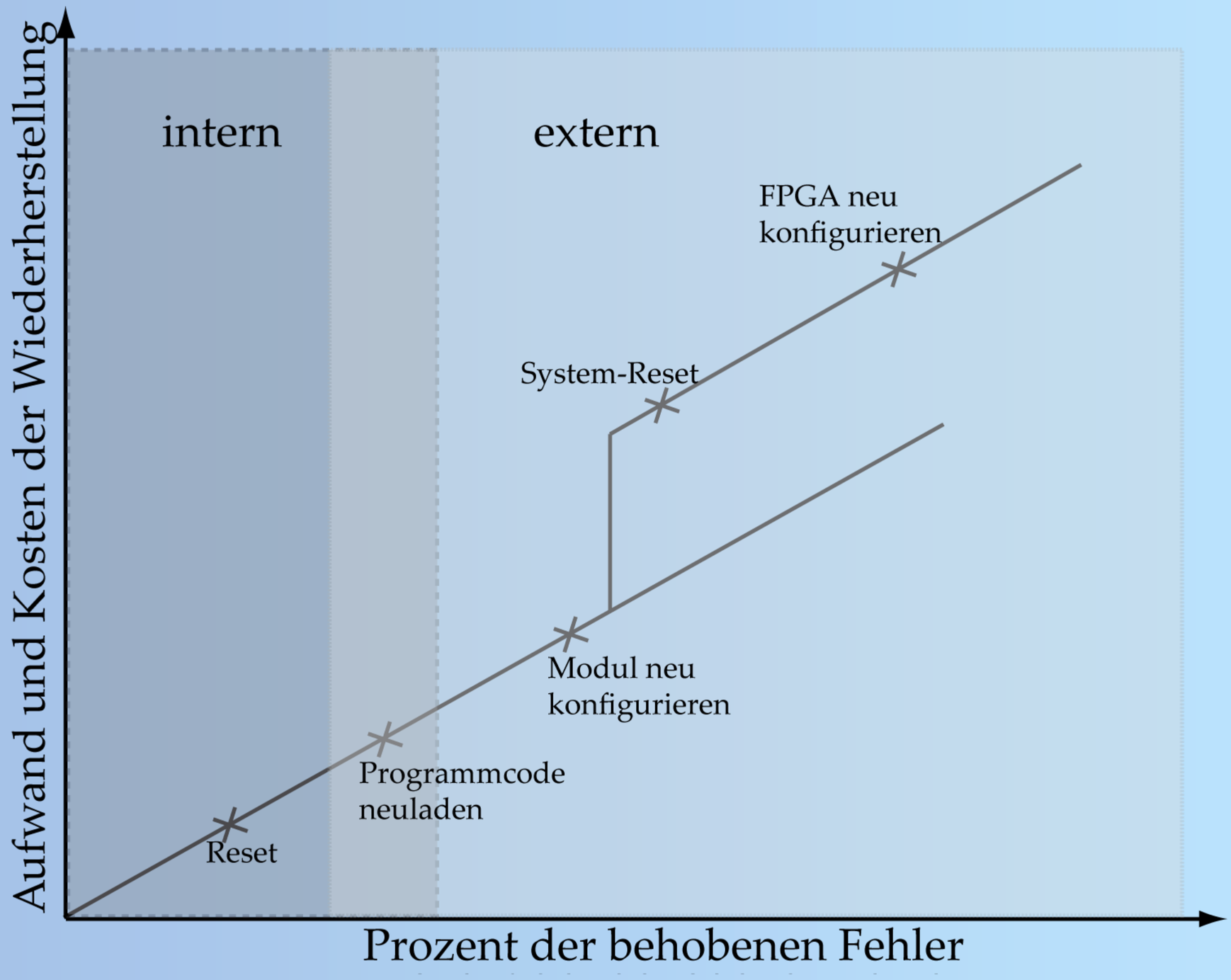
- IO ist immer mit entsprechendem I/O-Pin verbunden
- Ausgang
  - DD => ´1´, IN => ´open´
  - OUT wird von Modul gesteuert
- Eingang
  - DD => ´0´, OUT => ´0´
  - Modul empfängt Datum von IN
- Bidirektional
  - Alle Verbindungen werden an Modul angeschlossen
- Anwendung
  - Normale I/O-Ports
  - UART (RX- und TX-Pin)

Interne Kommunikation und Gemeinsamer Speicher

- Interne Kommunikation wird über eine Crossbar mit Ausgangssteuerung realisiert (je Modul ein unabhängiger Arbiter)
- Der Gemeinsame Speicher wird durch mehrere zusammengeschaltete Block-RAM-Komponenten gebildet (ein Arbiter zur Zugriffssteuerung)
- Jedes Modul kann durch andere Implementierungen ersetzt werden
  - Crossbar kann z.B. durch Omeganetz ersetzt werden
  - Implementation des gemeinsamen Speichers mit DDR-Speicher möglich
- Ansprechbar über gemeinsamen Port (Steuerungsport)



Mögliche Fehlerbehebung



4. Test und Auswertung

Vorläufige Auswertung und Test

- Das entworfene System muss noch getestet werden
  - Als modulare Einheiten dienen A8M-AVR
  - Modul zur Ansteuerung von LED und 7-Segment-Anzeige
  - Modul zum einlesen von Tastern

- Performancetest durch eine spezielle Zählerkomponente
  - Erwarteter Theoretischer Wert einer Kompletten Neukonfiguration mittels SelectMap: 22ms
    - Konfigurationsdatei: 1,4MB
    - Taktfrequenz SelectMap: 66MHz
    - Übertragene Daten pro Takt: 8Bit
  - Durchzuführender Vergleich von JTAG und SelectMap

5. Zusammenfassung aktueller Stand

Zusammenfassung

- Implementierung der einzelnen Einheiten ist abgeschlossen
- Zusammenschaltung der Komponenten
- Test steht für die nächste Woche an
- Implementation eines Prototypen

6. Ausblick

Ausblick

- Implementierung weiterer Module (Audio- und Videocodec, Sensorabfrage, Motorstererung)
- Fertigstellen der Agentenarchitektur
  - Mikrocontroller zur Konfiguration der FPGA
  - Implementierung eines Hardware-Taskmanager
  - Konfiguration und Readback mittels SelectMap
  - Fehlerermittlung und –korrektur mittels Scrabbing-Verfahren
- Vorstellbar währe eine Softwareumgebung, in der Code durch den Kompiler selbstständig in Software und Hardware getrennt wird und an das vorhanden System angepasst wird

Fragen?