TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik Institut für Technische Informatik, Professur für VLSI-Entwurfssysteme, Diagnostik und Architektur

Erhöhung der Ausfallsicherheit einer Mikropumpensteuerung mit Hilfe einer hierarchisch organisierten, heterogenen Controllerplattform

Zwischenpräsentation zum großen Beleg

Najdet Charaf – najdet.charaf@tu-dresden.de

Fraunhofer IWS

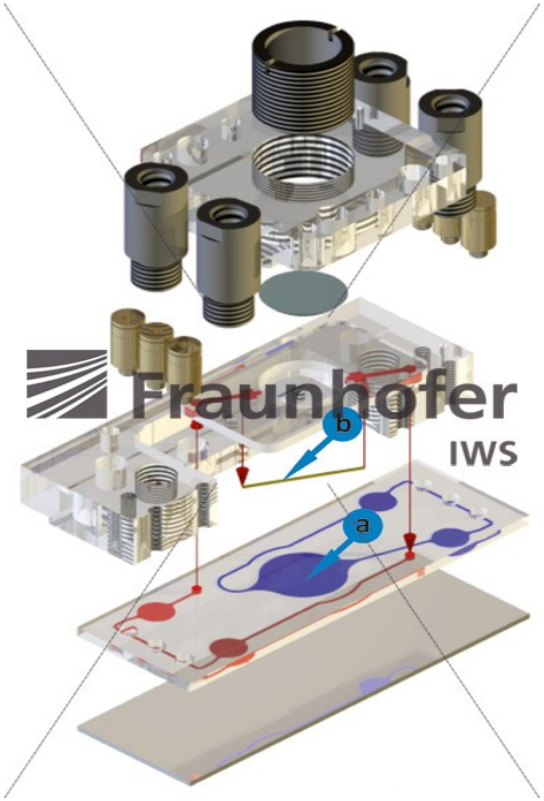
DRESDEN concept
Exzellenz aus Wissenschaft und Kultur

Dresden, 10.03.16

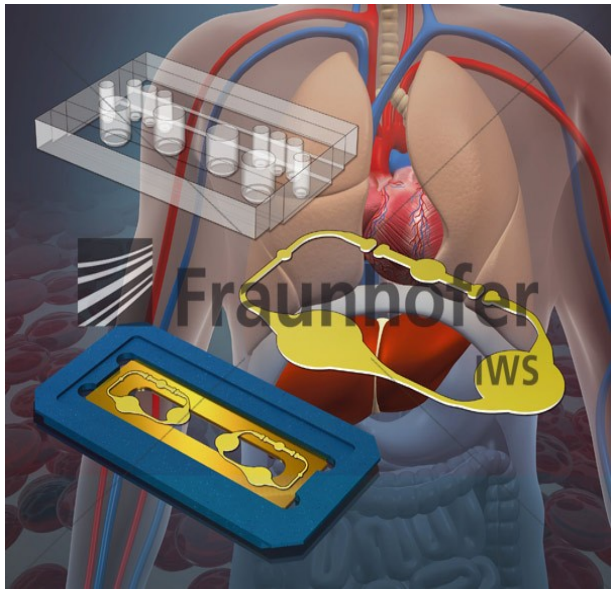
Gliederung

01. Motivation

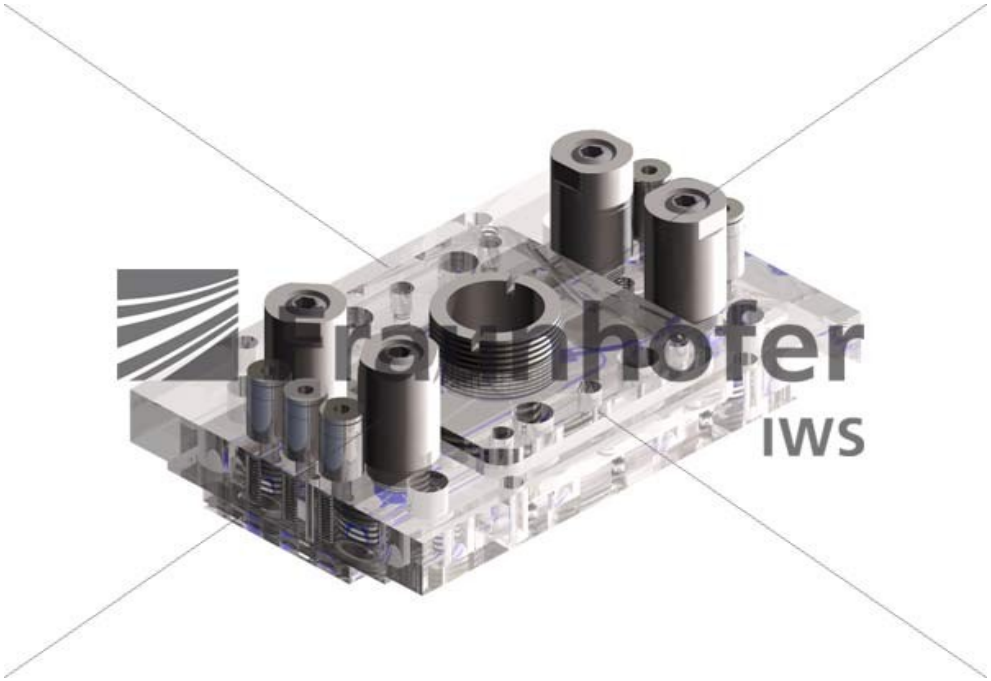
- **Lab-on-a-Chip**
  - Nachbildung biologischer Abläufe
  - Gesamte Funktionalität eines makroskopischen Labors auf einer nur kartengroßen Plattform
- **Multi-Organ-Chip (MOC)**
  - Nachbildung der im lebendigen Organismus ablaufenden Prozesse
  - mehrere komplexe Gewebekulturen, wie Niere, Leber und Haut, in einem Chip
  - Die dafür notwendige Mikrofluidik besteht u.a. aus Kanalstrukturen, Ventilen, Pumpen und Reservoiren
  - Anwendungsbeispiel:
    - Testen von neuen Medikamenten und kosmetischen Produkten



Explosionszeichnung des Dual-Perfusions-Systems mit (a) Position der CPOx-beads und (b) Hohlfaser



A dynamic multi-organ-chip for long-term cultivation and substance testing proven by 3D human liver and skin tissue co-culture



CAD-Modell eines mehrlagigen PDMS-Chips für die Kultivierung von Leber- und Hautgewebe in einem Chip

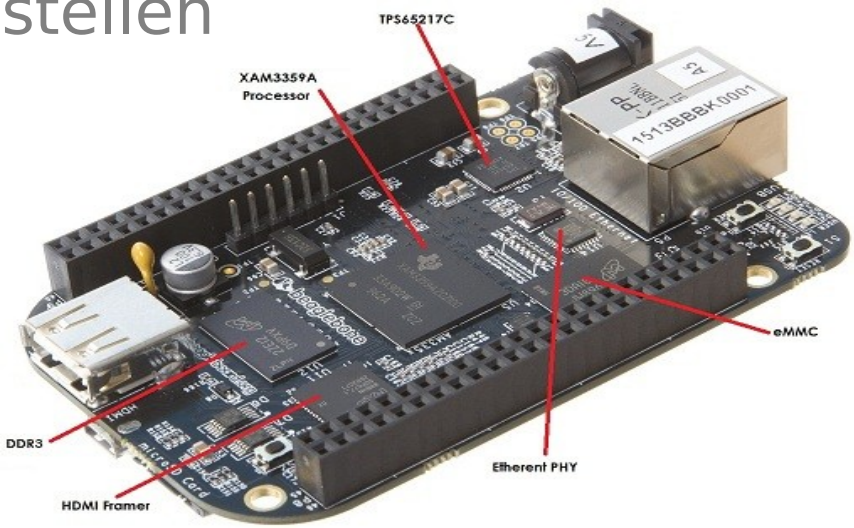
Hauptziel:

- Die MOC-Plattformen sollen in Zukunft u.a. auch die Tierversuche ersetzen
- Insgesamt 28 Tage lang ausfallfrei funktionieren
- Ausfallsicherheit der Mikropumpensteuerung erhöhen
- Umsetzung der Implementierung in Echtzeit

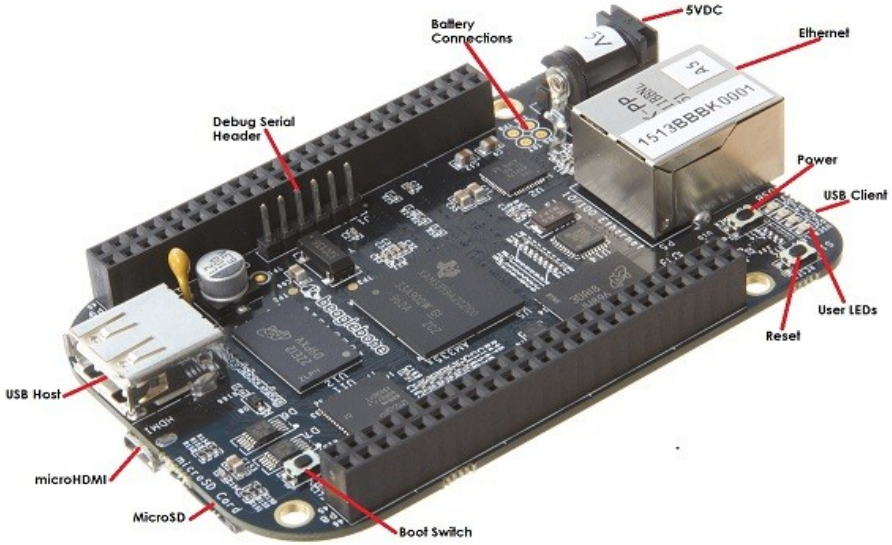
02 Controllerplattform Beaglebone Black

- Neuster Board der Beagleboard Familie
  - Beagleboards sind Einplatinen-Computer von Texas Instrument
- Kostengünstige Entwicklungsplattform
- Unterstützt primär Ubuntu, Debian und openSUSE
- **Eigenschaften:**
  - Prozessor: Sitara Cortex A8 ARM 1 GHz
  - SDRAM: 512 MB DDR3L 800 MHz
  - Energieversorgung: miniUSB, USB oder DC Stecker
  - MicroSD
  - 4 Serielle Schnittstellen
  - Ethernet
  - HDMI
  - SPI, I2C, GPIO(69 max), LCD, CAN usw.
  - 5 V ,3.3V Spannung

Komponenten und Schnittstellen



Quelle: [A5A]

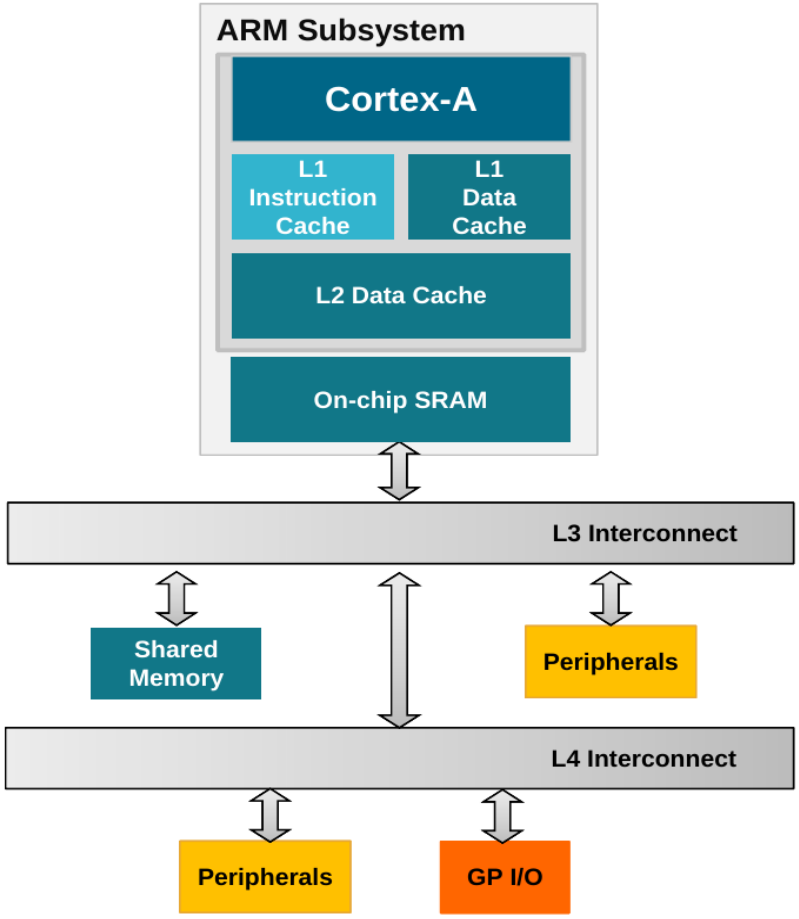


Quelle: [RA5A]

- **Besonderheiten:**
  - 2x Programmable real-time unit (PRU)
  - 2x 46 pin headers
  - 4 GB 8-bit eMMC on-board flash

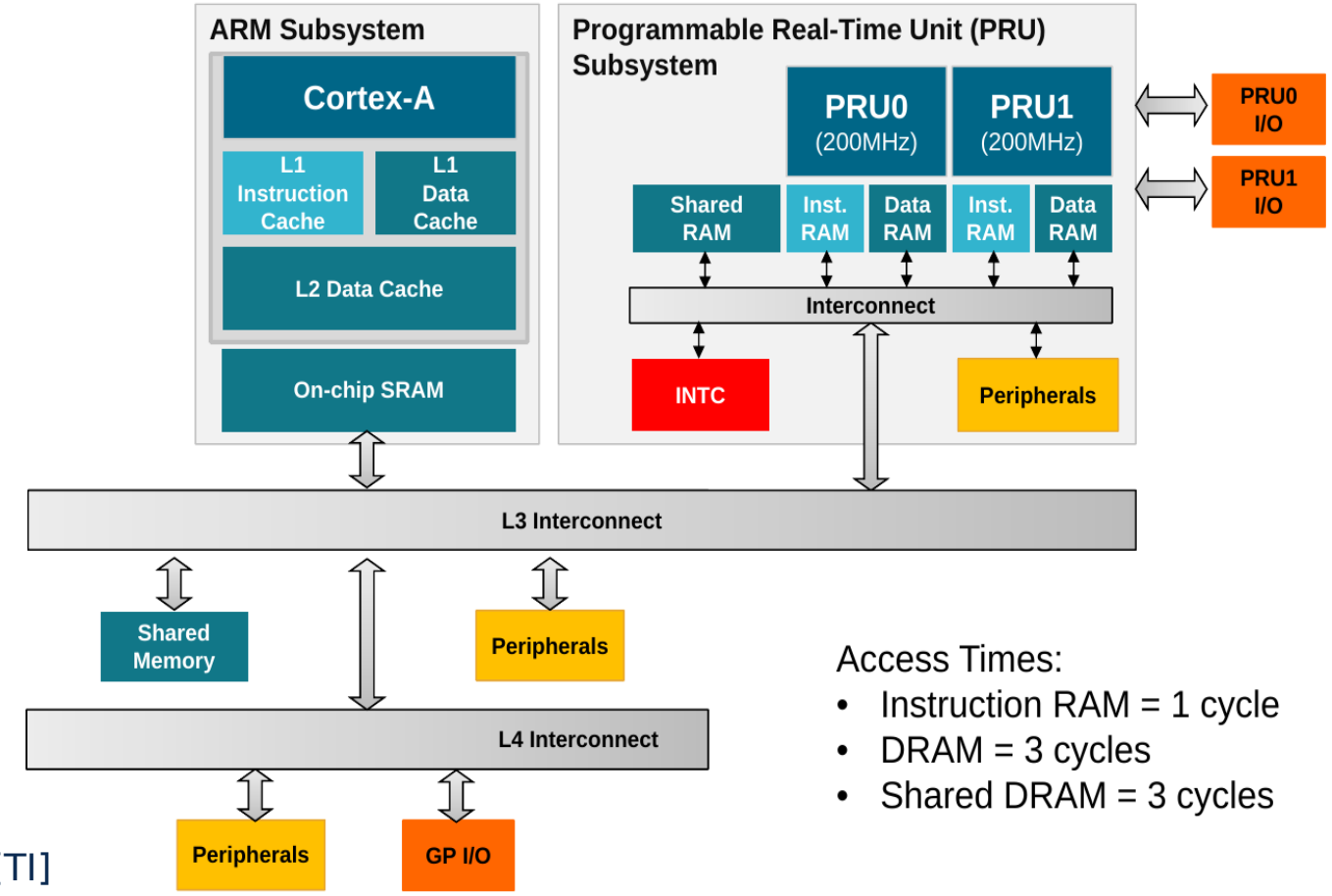
- **PRU:**
  - Subsystem mit geringer Latenz
  - 32-bit RISC Architektur
  - 200 MHz
  - Keine Pipeline – 1 Zyklus pro Befehl
  - Eigene 8 KB Daten- und Befehlsspeicher
  - Gemeinsame 12 KB Speicher
  - Direkt verbunden mit einigen Ein- / Ausgangspins
  - Zugriff auf den System-Hauptspeicher
  - Interrupt-Kontroller

ARM SoC Architektur



Quelle: [TI]

ARM + PRU SoC Architektur



Access Times:

- Instruction RAM = 1 cycle
- DRAM = 3 cycles
- Shared DRAM = 3 cycles

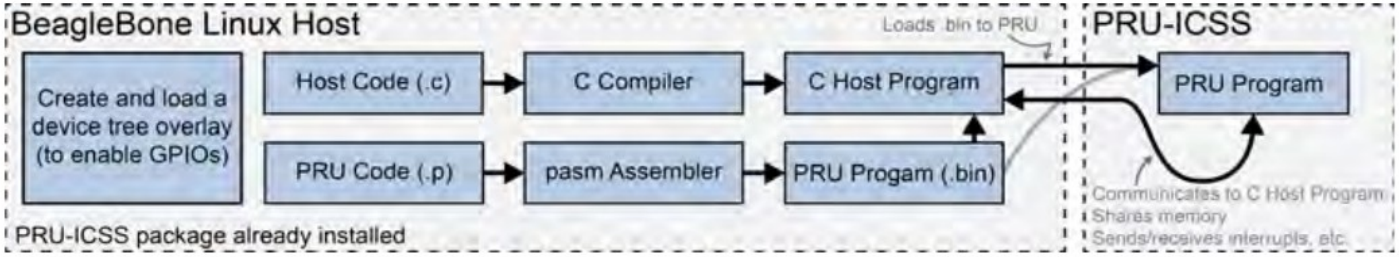
Quelle: [TI]

Warum Beaglebone Black ?

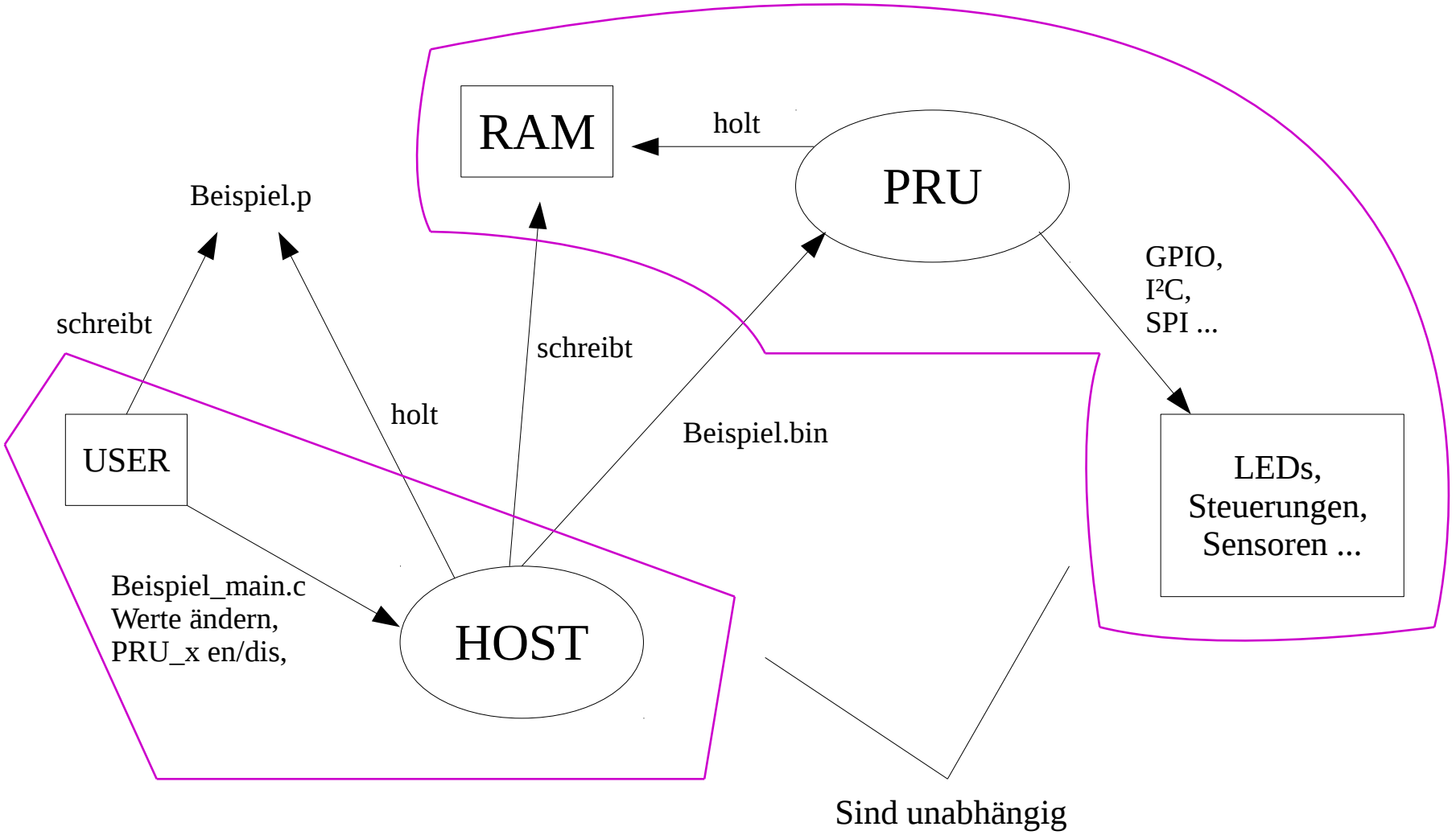
- Zwei unabhängige Programmierbare Echtzeit Einheiten
- ARM/PRU Kombination
- 4 GB On-board Flash
- Unterstützung zahlreiche Standardschnittstellen
- Sehr schnelle Verbindung mit den Ein- / Ausgangspins
- Unterstützt Webbasierte Entwicklungsumgebung
- Bootable von zwei verschiedenen Medien

03 ARM PRU Kommunikation

- Die PRU zuerst aktivieren
- Device tree overlay schreiben und/oder im Kernel laden
- Linux Host-Programm schreiben in der Programmiersprache C/C++
- PRU Programm schreiben in Assemblersprache
- C-Code compilieren und Assembler-Code assemblieren
- Übertragung der binären .bin-Datei in der PRU Befehlsspeicher



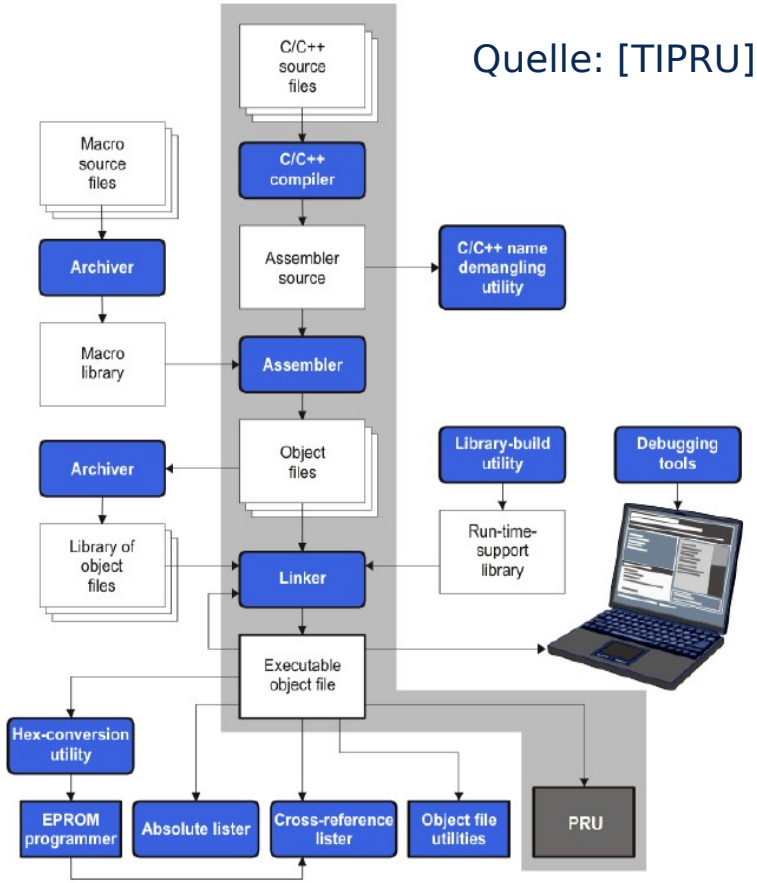
Quelle: [MODE]


RAM
holt
PRU
Beispiel.p
GPIO,
I²C,
SPI ...
schreibt
schreibt
holt
Beispiel.bin
USER
LEDs,
Steuerungen,
Sensoren ...
Beispiel_main.c
Werte ändern,
PRU_x en/dis,
HOST
Sind unabhängig

Alternative zum normalen Weg

- Texas Instrument C Compiler
  - Compiliert, assembliert, optimiert und linkt in einer Anweisung
  - Syntax:
    - clpru [options] [filenames] [--run_linker [link_options] object files]
  - Mit den Optionen das Compiler-Verhalten ändern
- Cloud9 IDE Integrierte online Entwicklungsumgebung, Node.js und BoneScript

Quelle: [TIPRU]



PRU Software Development Flow

04 Aktueller Stand

- PRU aktivieren/ deaktivieren
- Overlays nachladen
- ARM PRU Kommunikation
- PRU unabhängig ausführen
- Werte dynamisch in der PRU DRAM ändern
- ARM hängt, PRU führt weiter aus

05 Ausblick

- Der Integrierte I²C-Kontroller nutzen
- ARM PRU unabhängig wie möglich belassen
- Den C++ Code, der vorhanden Ansteuerlogik analysieren
- Die Steuerungsabläufe partitionieren in Haupt- (ARM) und Elementarsteuerung (PRU)
- Realisierung der partitionierten Steuerung mit der Einbindung in die vorhandene C++ Klassenbibliothek
- Im Labor testen

Quellen

[A5A] http://elinux.org/File:COMP_A5A.jpg

[BIO] https://www.biotechnologie.de/BIO/Navigation/DE/root,did=179124.html

[BBB] http://elinux.org/Beagleboard:BeagleBoneBlack

[IWS1] http://www.iws.fraunhofer.de/de/zentren/medizin-biosystemtechnik.html

[IWS2] http://www.iws.fraunhofer.de/de/geschaeftsfelder/mikrotechnik/mikro-biosystemtechnik/produkte_projekte/mikrofluidik.html

[MODE] Molloy, D., Exploring Beaglbone: Tools and Techniques for building with embedded Linux, 2015, S. 508

[RA5A] http://elinux.org/File:CONN_REVA5A.jpg

[TI] http://processors.wiki.ti.com/images/3/34/Sitara_boot_camp_pru-module1-hw-overview.pdf

[TIPRU] http://www.ti.com/lit/ug/spruhv6a/spruhv6a.pdf

»Wissen schafft Brücken.«