CLEVER: Hardware Compiler für neuronale Netze

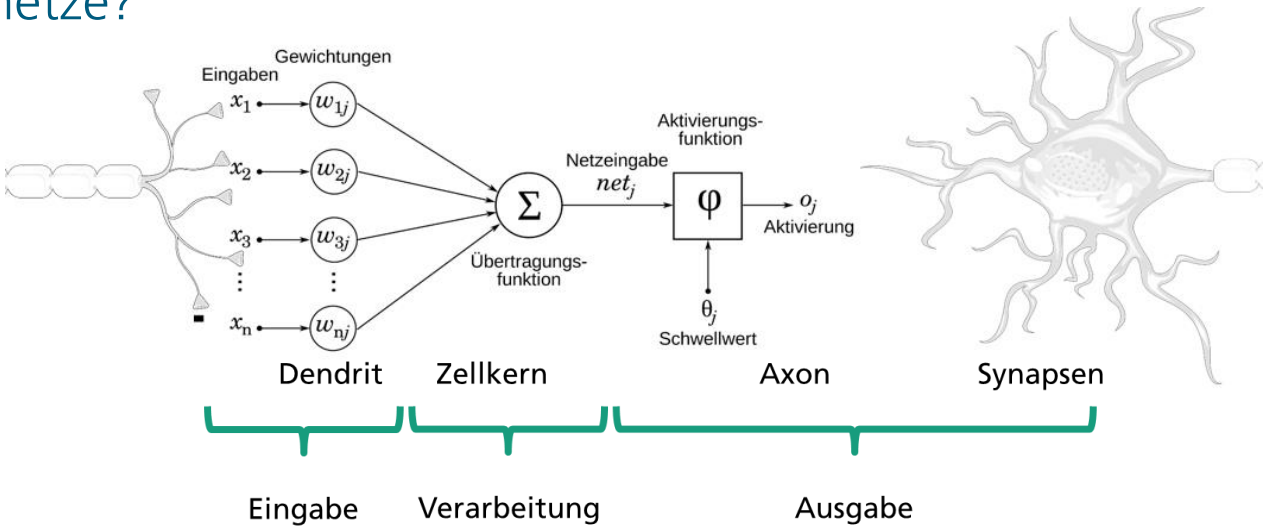
Agenda

1. Einführung und Motivation
2. Das CLEVER-Projekt
3. Hardware Compiler für neuronale Netze
4. Vorläufige Ergebnisse
5. Schlussfolgerungen
6. Referenzen

Einführung und Motivation

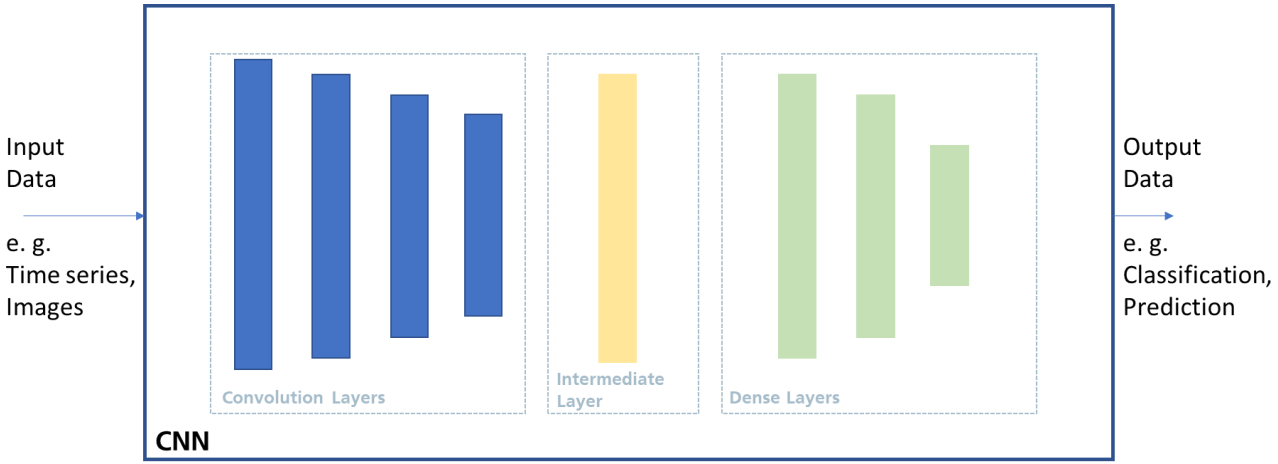
Was sind neuronale Netze, insbesondere Faltungsnetze?

- Mathematisches Modell, orientiert am biologischen Vorbild
  - Eingabewerte werden gewichtet und summiert
  - Ausgabewert hängt von einer Aktivierungsfunktion und ggf. einem Schwellwert ab
  - Netze werden uns vielen dieser Neuronen gebaut, die sich in Schichten (Layer) kombinieren



© Servier Medical Art by Servier is licensed under the terms of the Creative Commons Attribution 3.0

- Faltungsnetze (Convolutional Neural Network, CNN) verwenden spezielle Operationen um die Datenmenge zu reduzieren
- Nach den Faltungen folgen üblicherweise reine Multiplikationslayer (sog. Dense oder fullly connected Layer)



Einführung und Motivation

Was sind Anwendungsfelder für diese Architekturen? Was sind Herausforderungen?

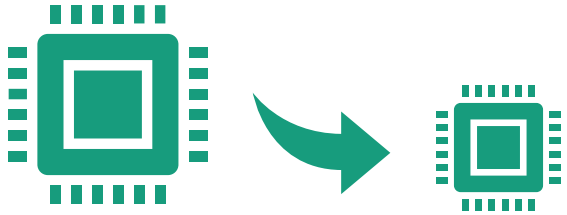
FMD.iDay23

Vielfältige Anwendungen

- Analyse von Zeitreihendaten, z. B.
  - Maschinenschwingungen, EKG-Signale
- Bildklassifikation, z. B.
  - Objekterkennung

Große Herausforderungen für die Hardware

- Leistungsfähigkeit
  - Algorithmen sind Rechenintensiv
- Energieeffizienz
  - Gerade in Edge Devices ist das Energiebudget begrenzt



Schnelles Design

Low Power

Performance

Hier besteht Bedarf an spezialisierter Hardware!

Das CLEVER-Projekt [1]

FMD.iDay23

Collaborative edge c**L**oud continuum and **E**mbedded AI for a **V**isionary industry of th**E** futu**R**e

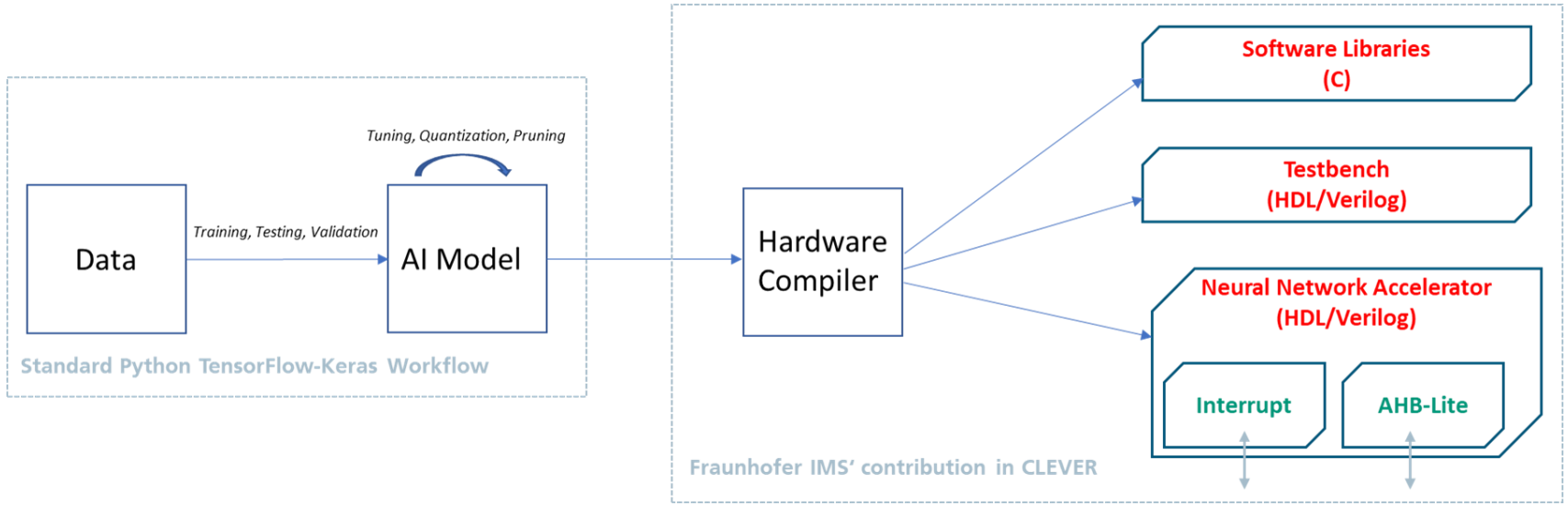
- Hardware Beschleuniger für Edge Computing und IoT-Kommunikation
- Middleware und Test-Setups um die Systeme zu integrieren und zu testen
- Einbringung der Technologien in drei Use-Cases
  - Smart Factory, Smart Shopping, Smart Farming

Das Projekt wird von der Europäischen Union unter 101097560 - CLEVER - HORIZON-KDT-JU-2021-2-RIA gefördert und vom Bundesministerium für Bildung und Forschung BMBF unter dem Förderkennzeichen 16MEE0263K kofinanziert.

Hardware Compiler für Neuronale Netze

Idee: Automatisiert Beschleuniger generieren

- KI-Algorithmen werden i. d. R. in Python entwickelt
- Einfache Integration des Compilers in die „gewohnte Python Umgebung“
- Automatische Erzeugung des Beschleunigers in HDL/Verilog
  - Inklusive Interfaces
- Automatische Erzeugung von Testbenches zur Verifizierung der Beschleuniger
- Automatische Erzeugung von Embedded Software (C) –Bibliotheken zur einfachen Einbindung



Hardware Compiler für Neuronale Netze

Anbindung an den AIRISC Microcontroller

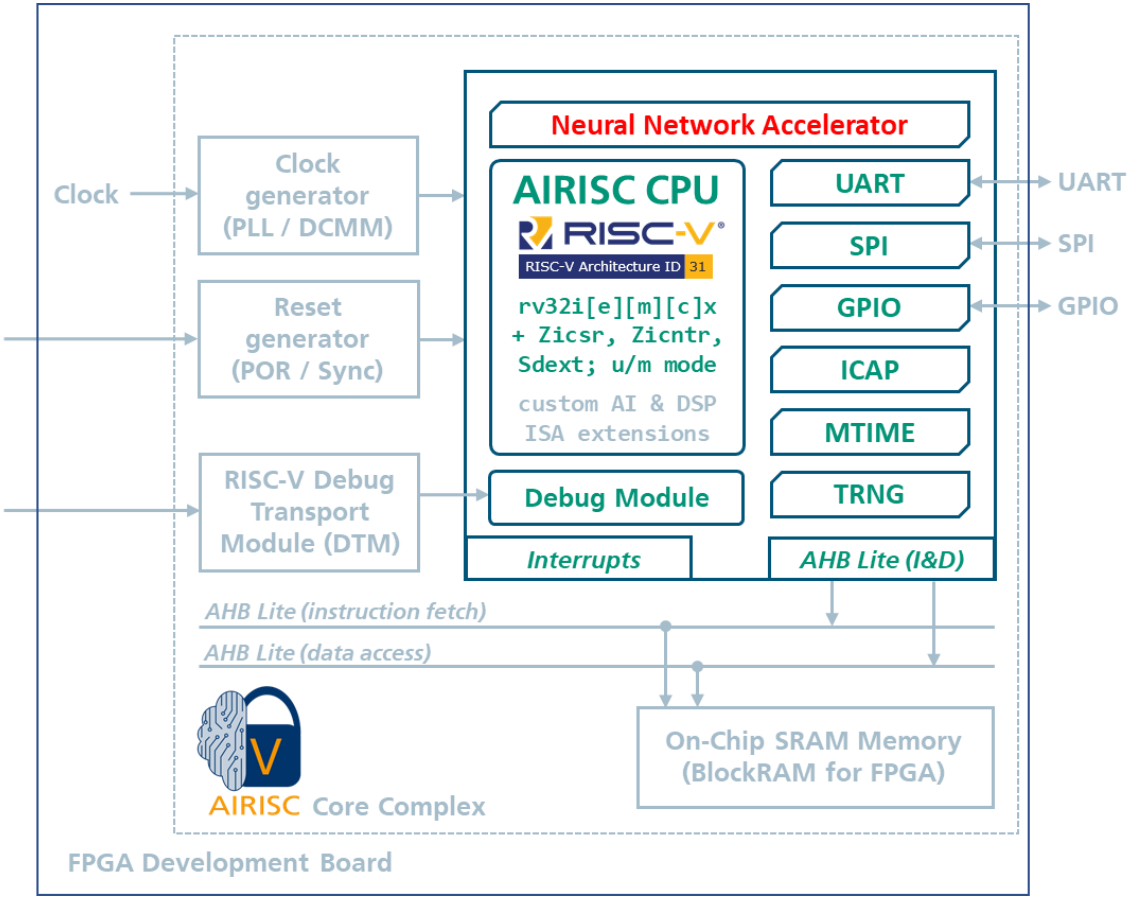
FMD.iDay23

- Einfache Anbindung des Beschleunigers über ein standardisiertes memory-mapped Interface (AHB-Lite) und Interrupts

AIRISC ist optimiert für Embedded AI [2]

- 32-bit RV32IMFCPX ISA
- Open-Source auf Github [3], siehe QR-Code
- Maßgeschneidert für Embedded Anwendungen: Modular konfigurierbar und Erweiterbar
- FPGA-Projekte zum einfachen Ausprobieren auf Github

AIRISC



Vorläufige Ergebnisse

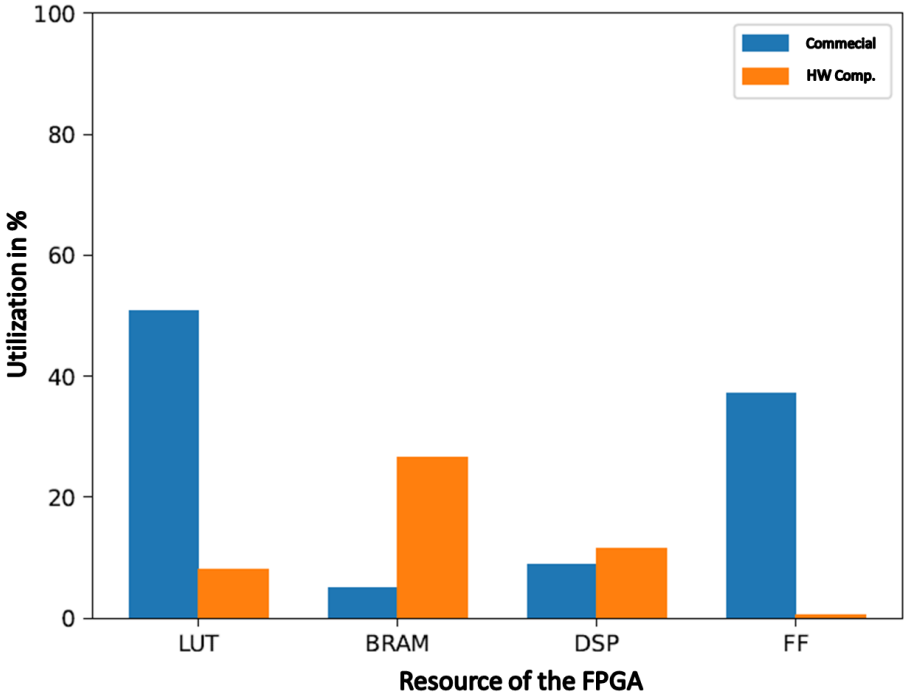
Proof of Concept

Proof of Concept

- Beschränkter Funktionsumfang mit wenigen Layern
- Nur für 1D-Daten
- Nur teilweise Validiert

Vergleich zu Beschleuniger erzeugt mit Vivado HLS (blau)

- CNN zur Analyse von Maschinenschwingungen
- Hardware Compiler hat vergleichbaren Ressourcen-Verbrauch
  - Auf Digilent NexysVideo Board mit Xilinx-FPGA
- Spart bis zu 99,4 % der Latenz ein



Timing Results			
Implementation	Clock Cycles	Critical Path in ms	Minimum Latency in ms
Software	77 044 504	25.62	1 973.88
Accelerator	330 494	35.36	11.69

Fazit und Ausblick

Ansatz ist vielversprechend und wird im Projekt bis Ende 2025 weiterentwickelt

- Ausgiebige Verifikation der erzeugten Layer
- Ergänzung weiterer Layer
- Ausbau für 2D-Daten (Bilder)
- User Constraints zur Optimierung auf Latenz oder Ressourcen-Verbrauch

Weitere Forschung zur Implementierung von konfigurierbaren Beschleunigern

- Spezielle konfigurierbare Arrays basierend auf FPGA-Fabric

Referenzen

[1] https://www.cleverproject.eu/

[2] https://www.ims.fraunhofer.de/de/Kernkompetenz/Smart-Sensor-Systems/Integrated-Sensor-Systems/airisc-family.html

[3] https://github.com/Fraunhofer-IMS/airisc_core_complex

Vielen Dank für Ihre Aufmerksamkeit!

Ingo Hoyer
Ingo.hoyer@ims.fraunhofer.de

Forschungsfabrik Mikroelektronik Deutschland
Eine Kooperation von Fraunhofer MIKROELEKTRONIK