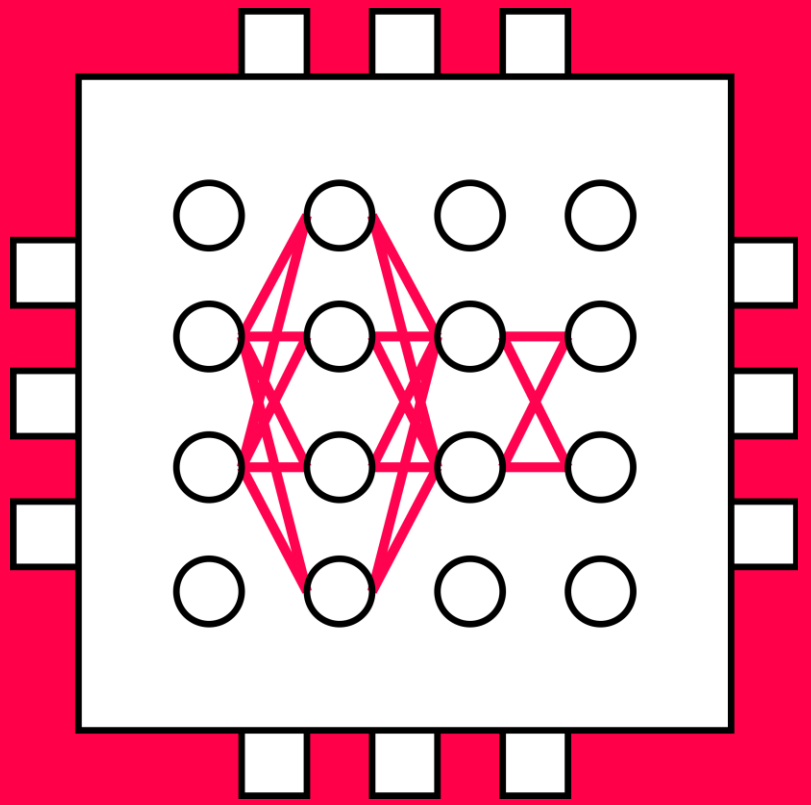
Technische Hochschule Augsburg

THANNA

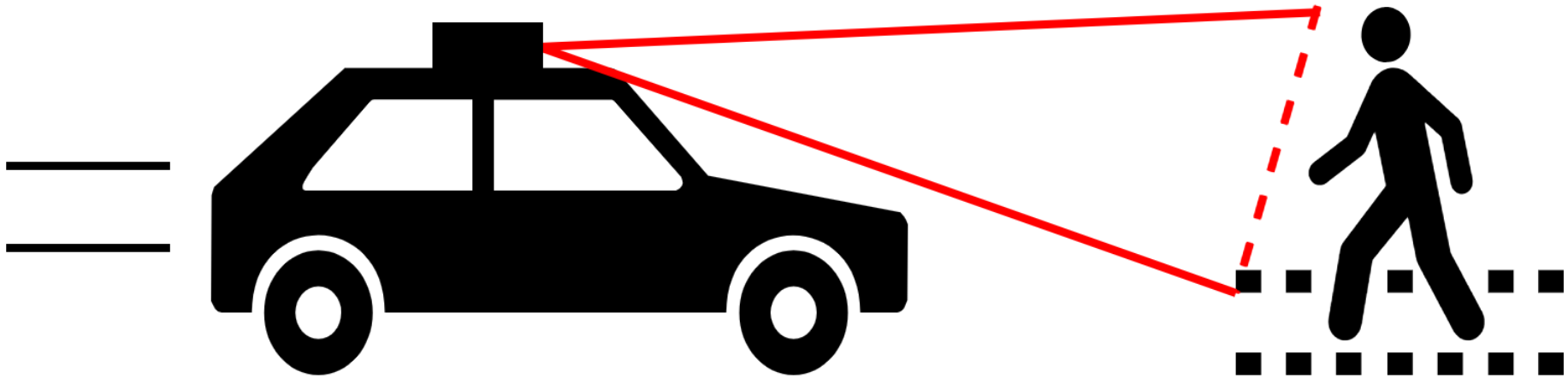


Technische Hochschule Augsburg

Neural Network Accelerator

THANNA - Maximilian Schunck, Korbinian Gast, Philipp Urlbauer, Michael Schäferling, Prof. Dr. Gundolf Kiefer

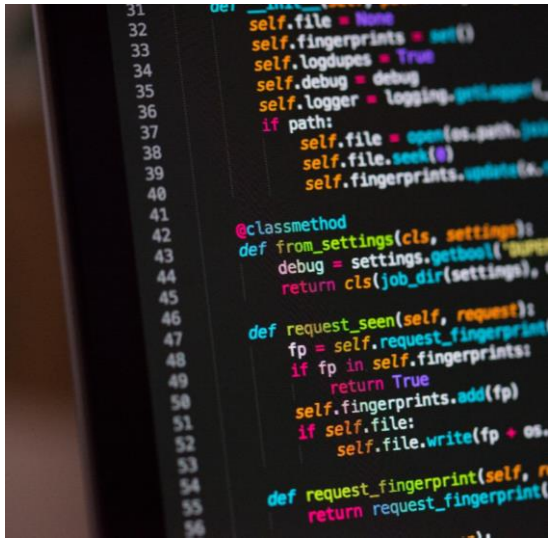
EINLEITUNG



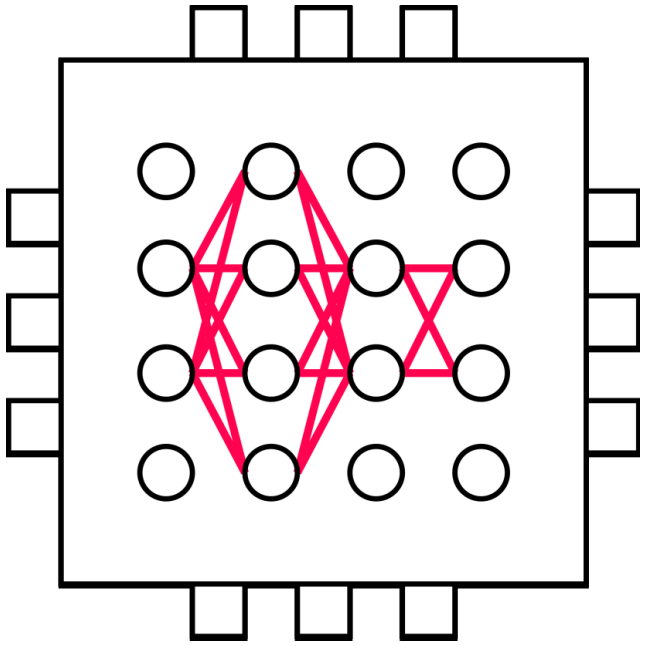
ZIELE

- Optimierte Neuronale Netze
- Effiziente Hardware für KI-Anwendungen
- Optimale Anpassung an gegebene Neuronale Netze
- Einfache Benutzung

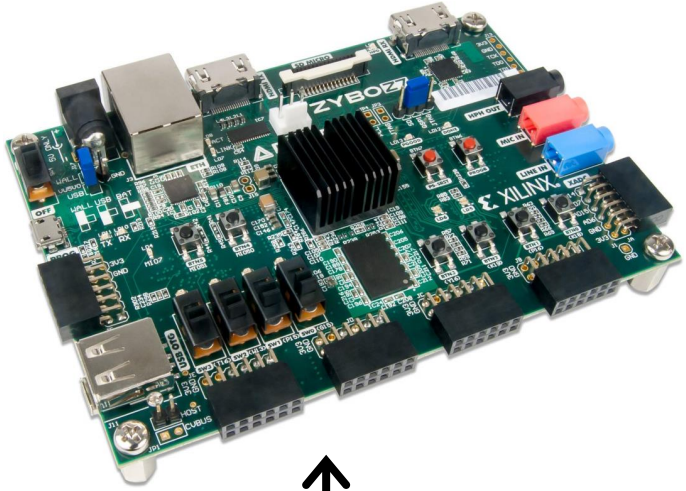
THANNA FRAMEWORK



THANNA Quantizer



THANNA Prozessor



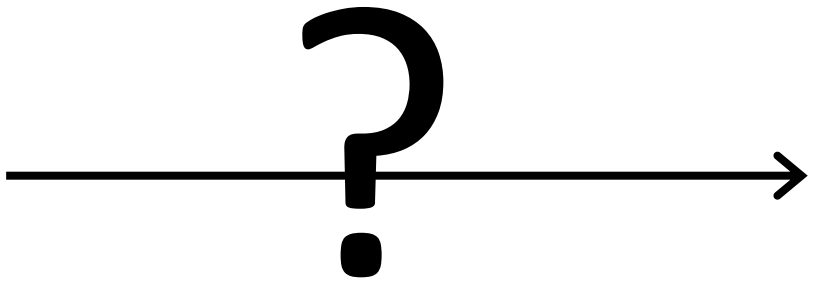
PetaLinux
THANNA Treiber
THANNA Library

Quelle: diligent.com

Technische Hochschule Augsburg

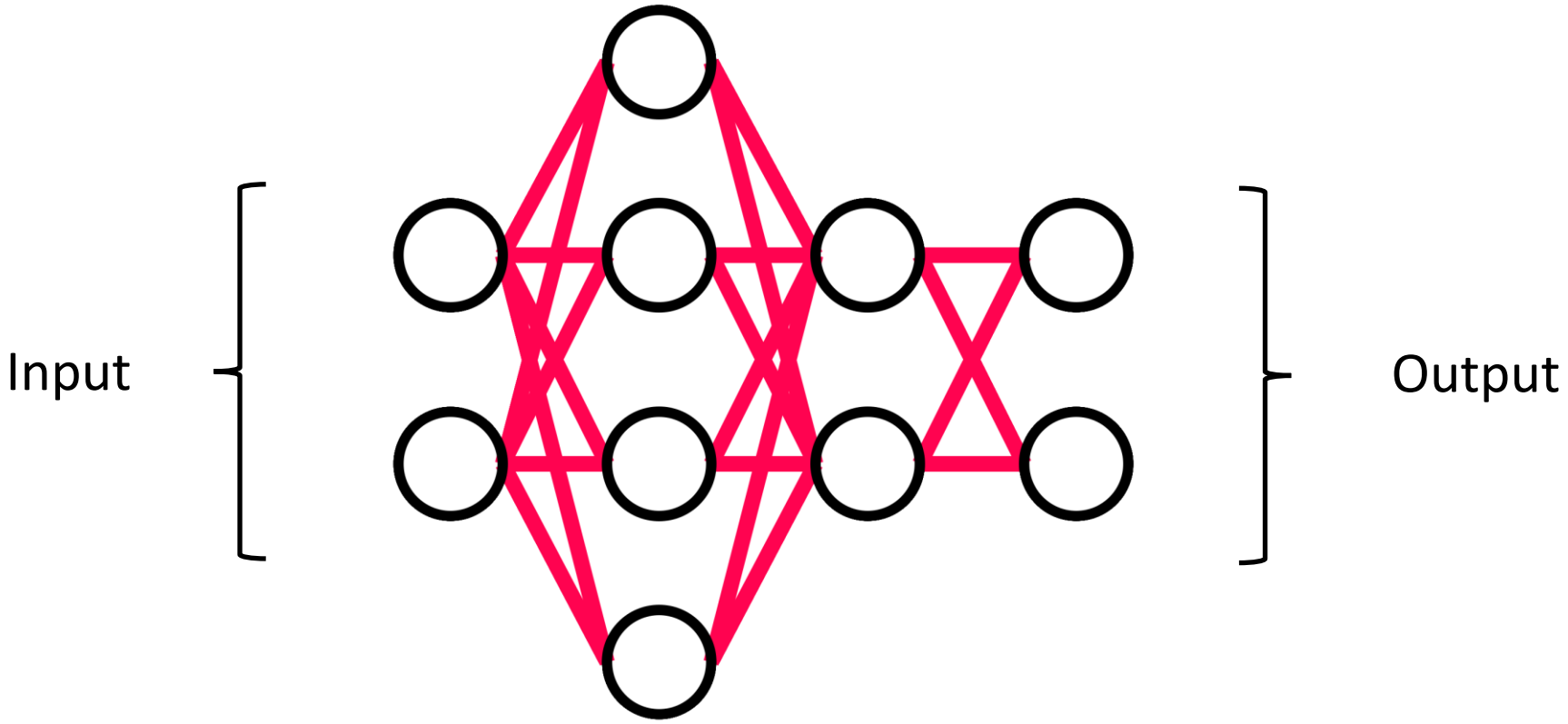
1. Neuronale Netze und Optimierungen
2. THANNA Quantizer
3. THANNA Prozessor
4. THANNA Treiber

WIE FUNKTIONIEREN NEURONALE NETZE?



Hund

FUNKTIONSWEISE NEURONALER NETZE

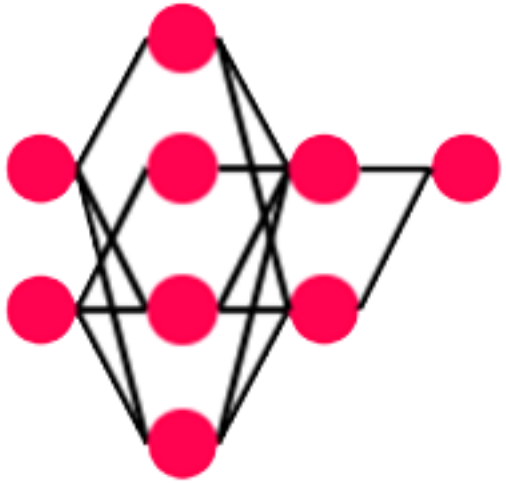


FUNKTIONSWEISE NEURONALER NETZE

Hund

HAUPTKOMPONENTE

- Fully Connected Layer

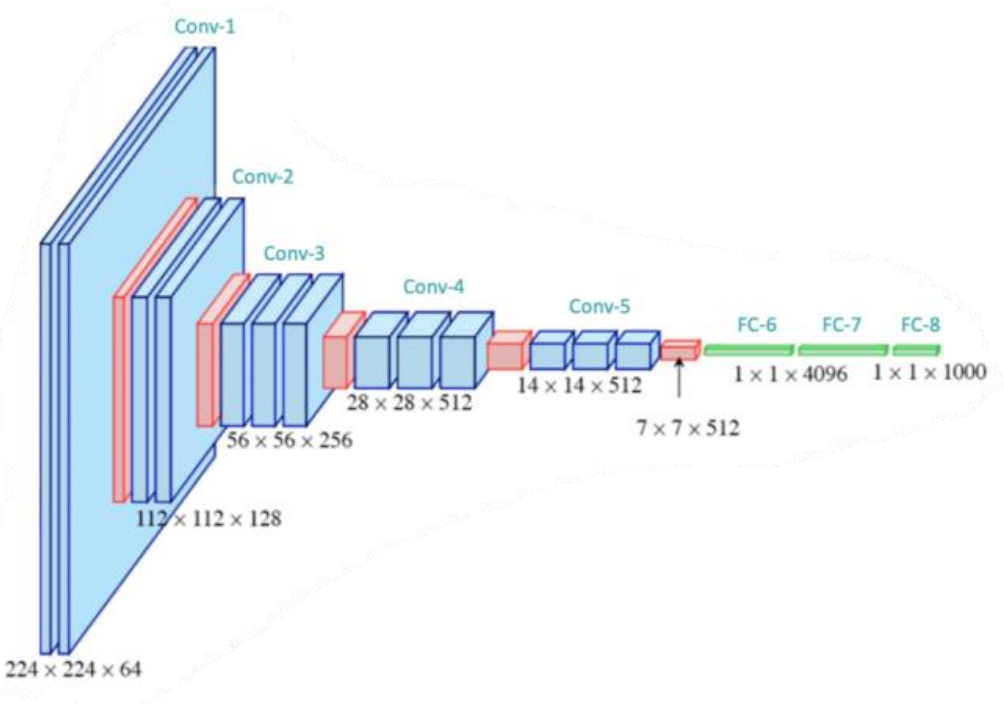


- Convolutional Layer

HAUPTKOMPONENTE

- Fully Connected Layer: $$y_i = \sum_{j=0}^{n} x_j * w_{i,j}$$
- Convolutional Layer: $$y_{i,j} = \sum_{a=0}^{m-1} \sum_{b=0}^{n-1} x_{i+a,j+b} * w_{a,b}$$

FUNKTIONSWEISE NEURONALER NETZE



Hund

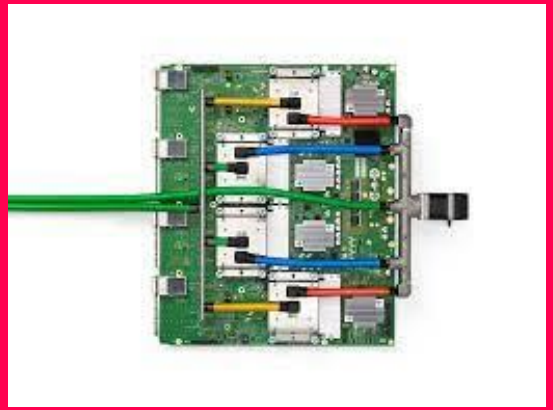
Quelle: Max Ferguson et al. “Automatic localization of casting defects with convolutional neural networks”

KLASSISCHE BESCHLEUNIGER

- Feste Anzahl und Struktur der Recheneinheiten
- Fixe Bit-Breite

➔ **Geringes Optimierungspotential**

GPU (RTX 2080)



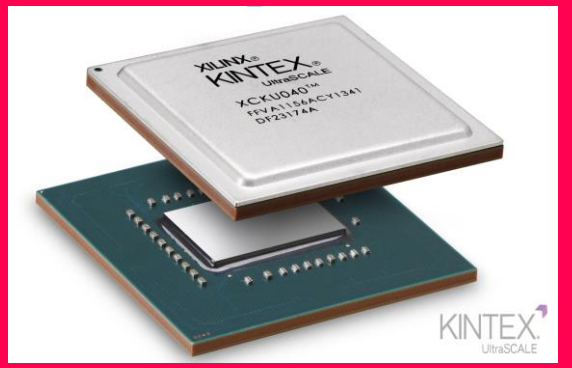
SoC (Edge TPU)

Quelle:wikipedia.com

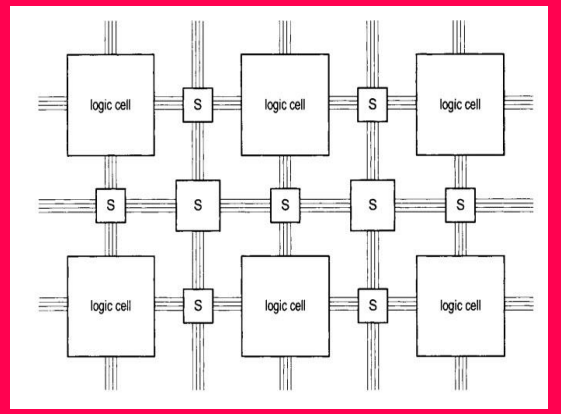
FPGA
(FIELD PROGRAMMABLE GATE ARRAYS)

- Aufbau aus Logikblöcken
- Darstellung beliebiger Hardwarestrukturen

➔ **Hohes Optimierungspotential**



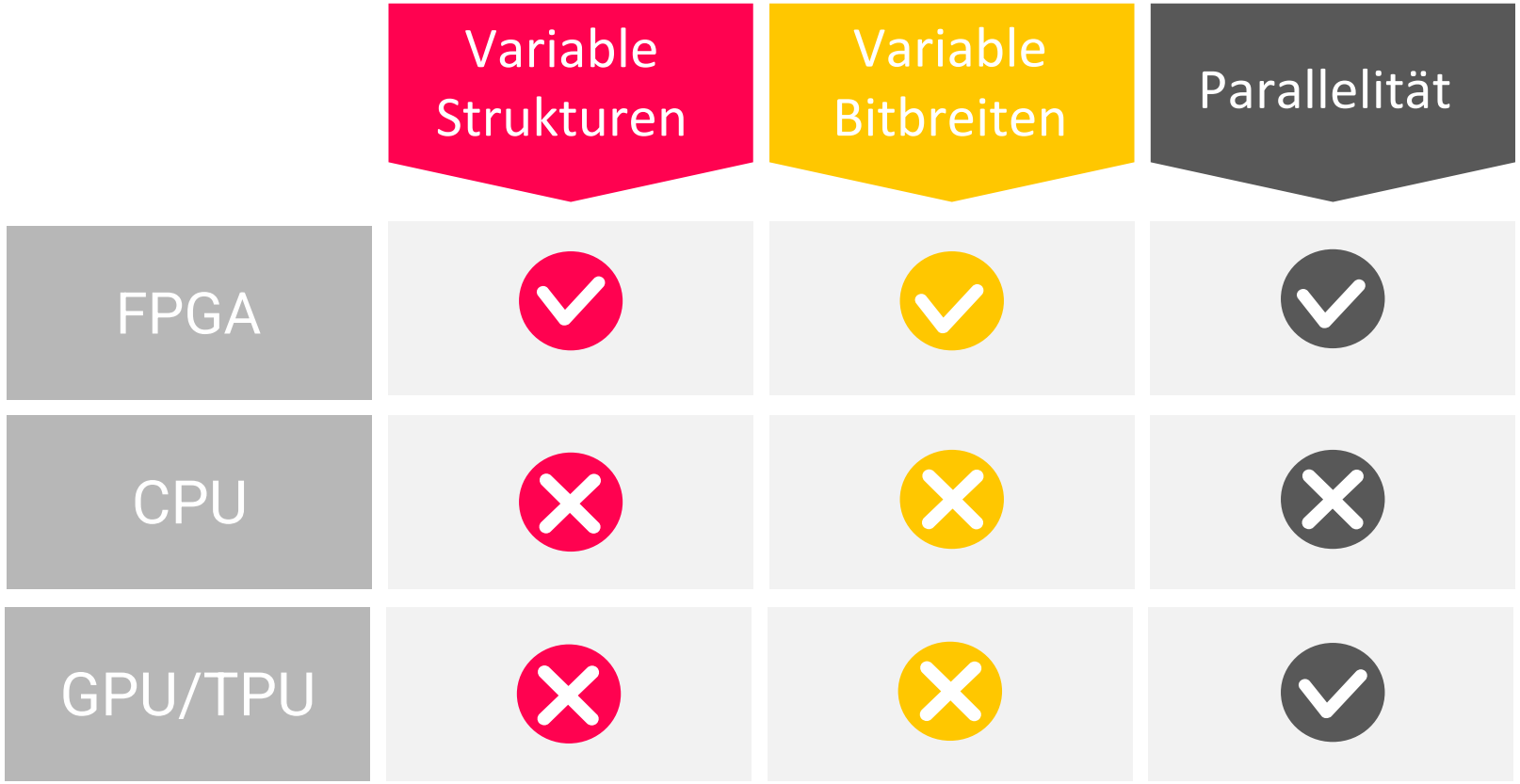
Xilinx FPGA



Interner Aufbau

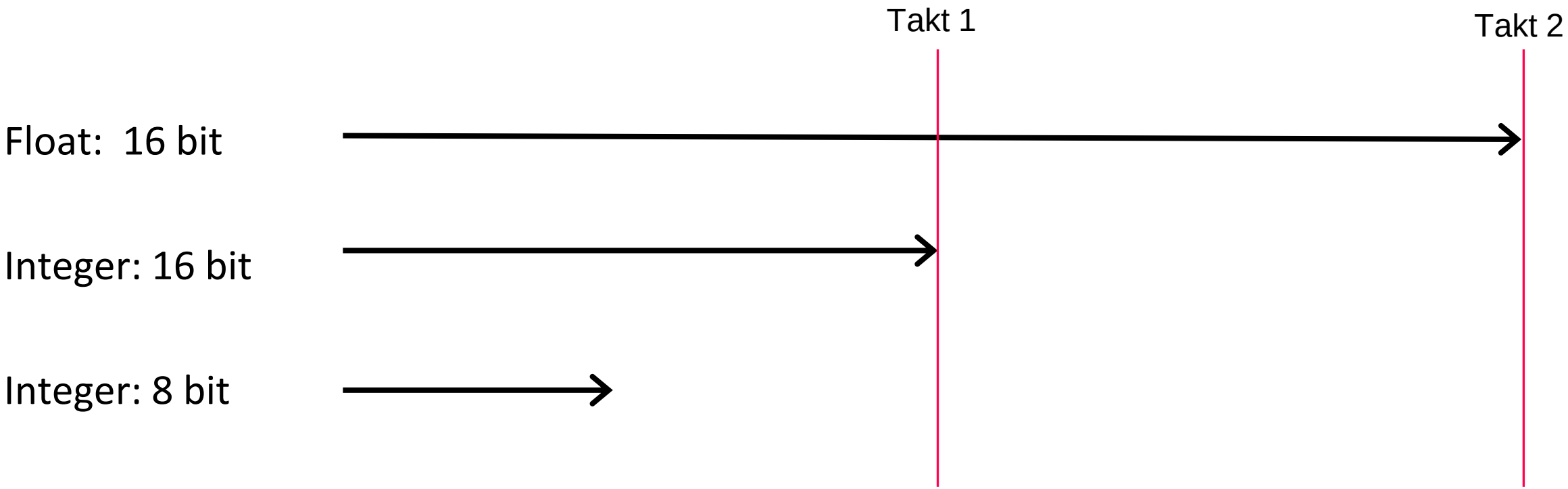
Quelle:xilinx.com

FPGA VS FESTE HARDWARE



VORTEILE VON VARIABLEN FIXEDPOINT BITBREITEN

GESCHWINDIGKEIT



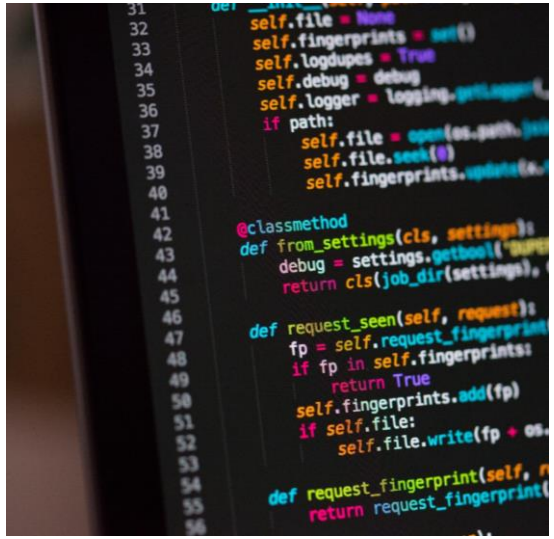
VORTEILE VON VARIABLEN FIXEDPOINT BITBREITEN

STROMVERBRAUCH / HARDWARE-AUFWAND

Technische Hochschule Augsburg

1. Neuronale Netze und Optiemierungen
2. THANNA Quantizer
3. THANNA Prozessor
4. THANNA Treiber

THANNA FRAMEWORK



THANNA Quantizer

Host-PC

QKERAS

Features:

- Open Source
- Quantisierungstool
- Post Training Quantization
- Quantization aware Training

Probleme:

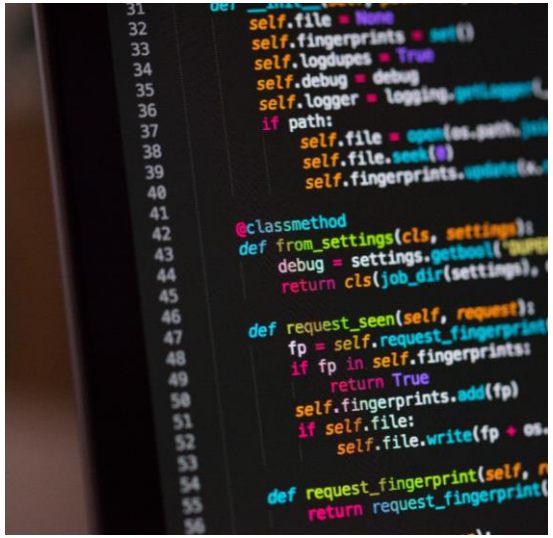
- Fehlerhafte Beispiele
- Mehrere Bugs
- Hoher Genauigkeitsverlust bei vortrainierten Neuronalen Netzen
- 3-4x langsameres Training

THANNA QUANTIZER

Features:

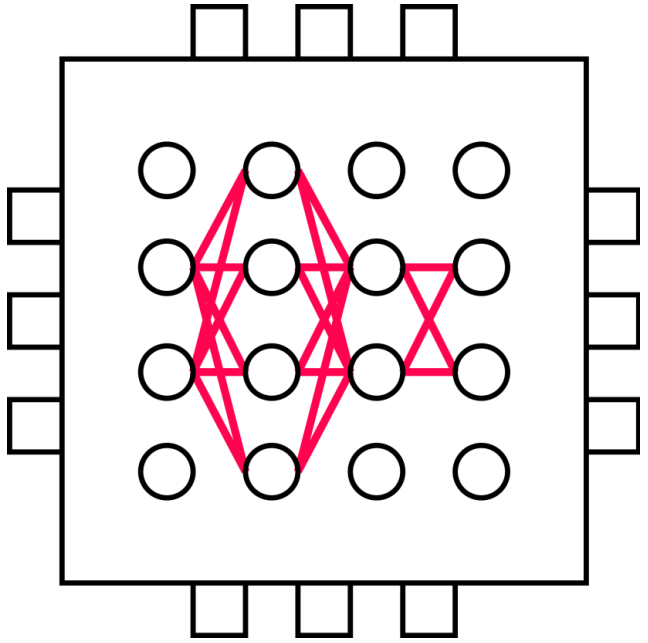
- Open Source
- Minimale Funktionalität von QKeras
- Quantisierung kleiner Netze
- Optimierung der Quantisierungsergebisse

THANNA FRAMEWORK



THANNA Quantizer

quantisiertes
neuronales Netz



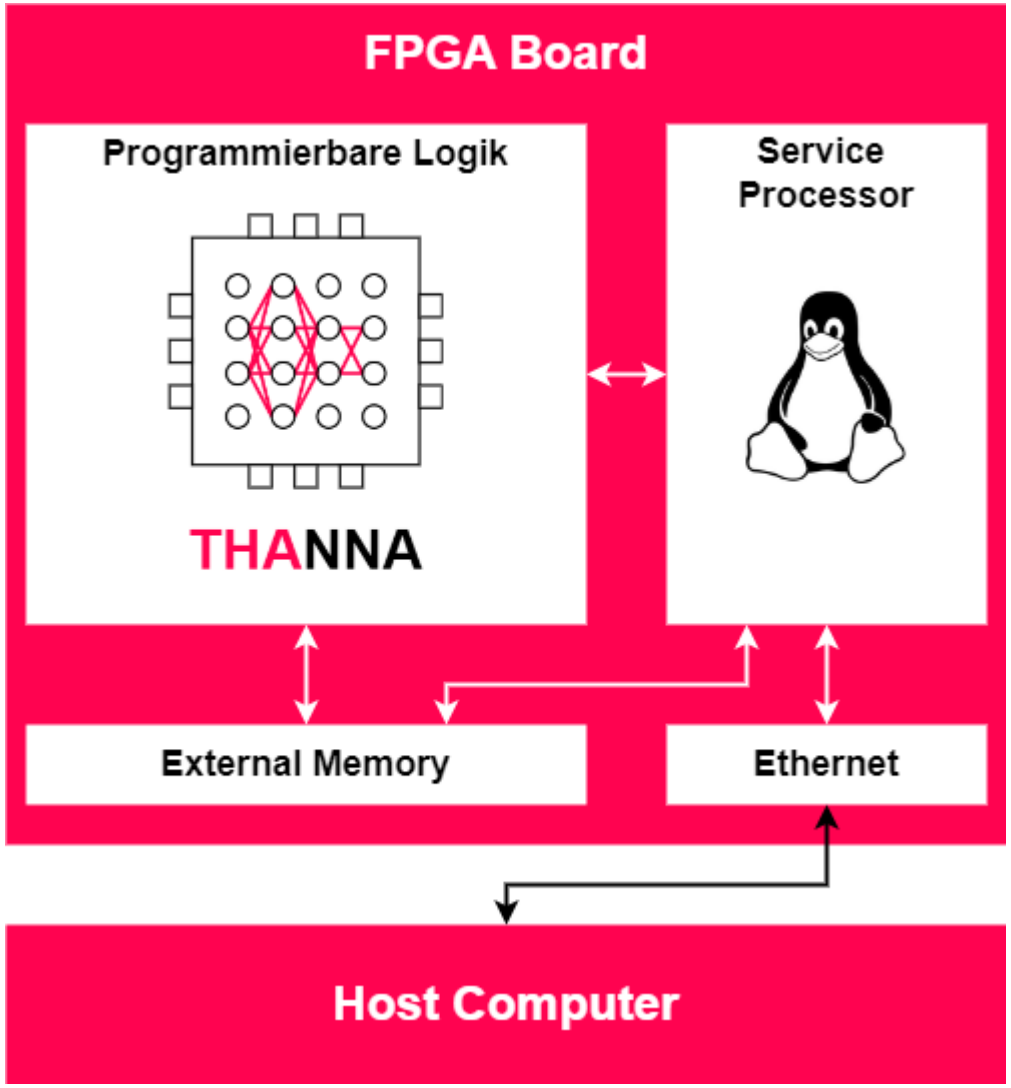
THANNA Prozessor

Technische Hochschule Augsburg

1. Neuronale Netze und Optiemierungen
2. THANNA Quantizer
3. THANNA Prozessor
4. THANNA Treiber

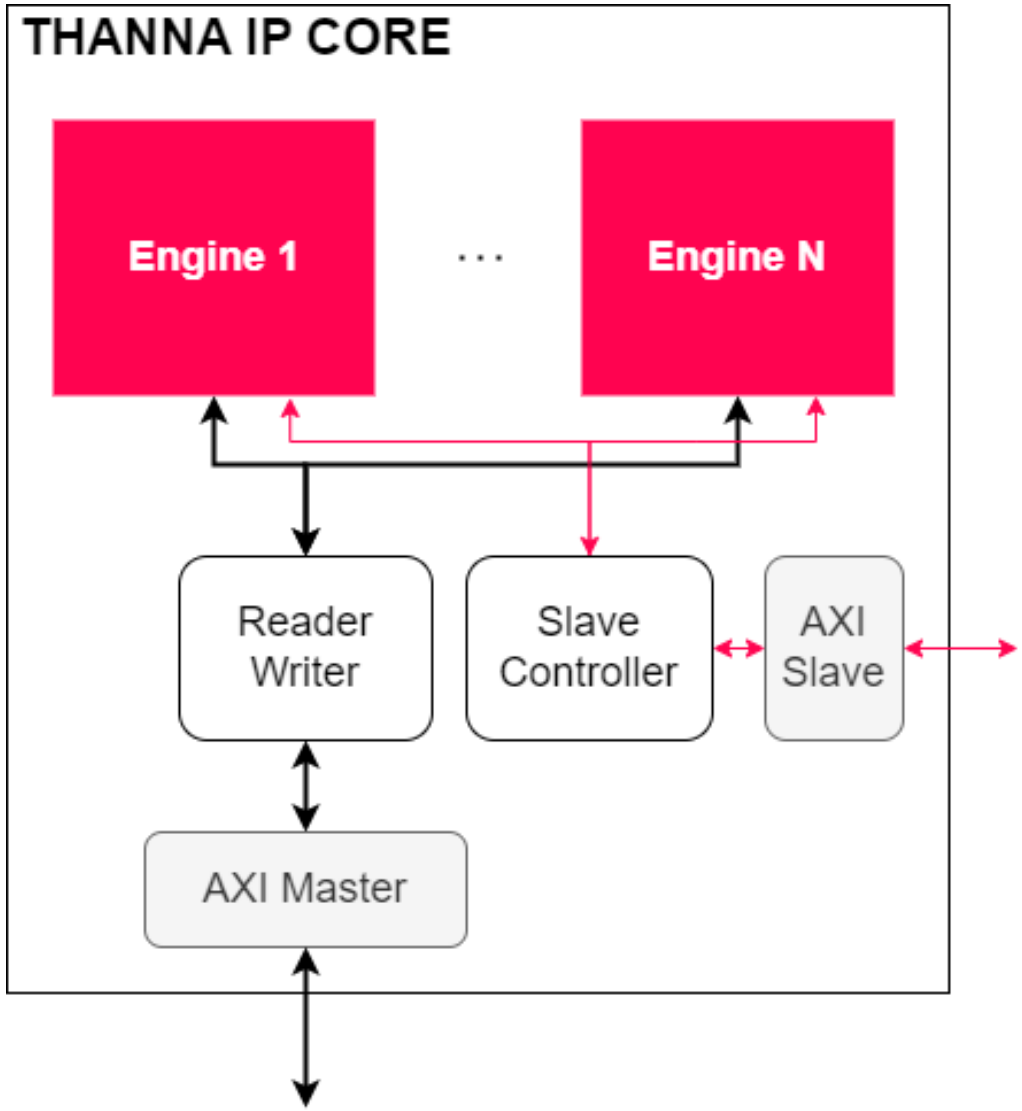
SYSTEM ÜBERSICHT

- Datenübertragung vom **Computer** bis zum **IP Core**



DER THANNA IP CORE

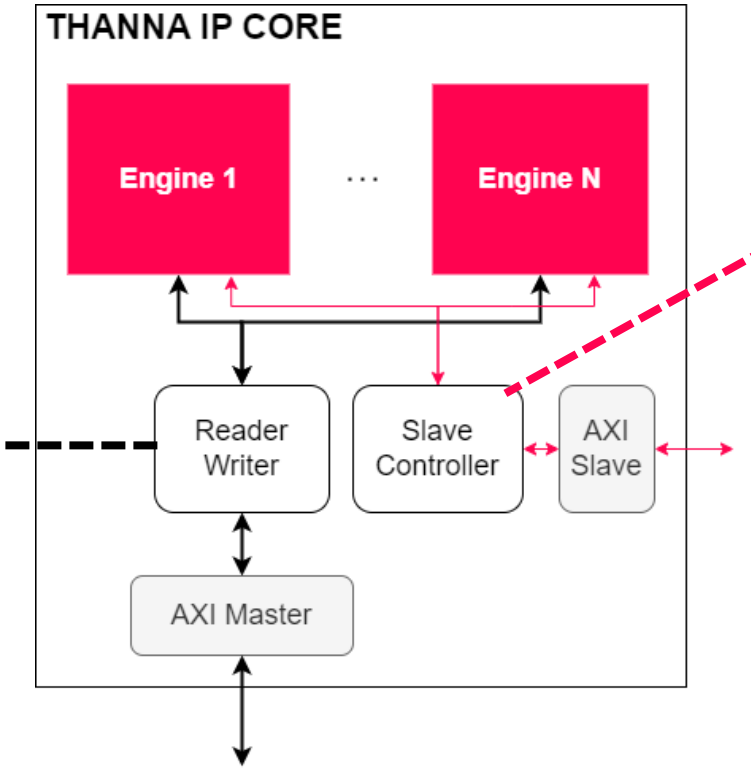
- **Flexible** Architektur
- **Erweiterbar** durch Anknüpfung beliebig vieler Engines



SCHNITTSTELLEN

Datenschnittstelle

- Burstdaten
- Lese- / Schreib-adressen
- Statusinformationen



Kommunikations - schnittstelle

- Kommando
- Kommando Kontext
- Status

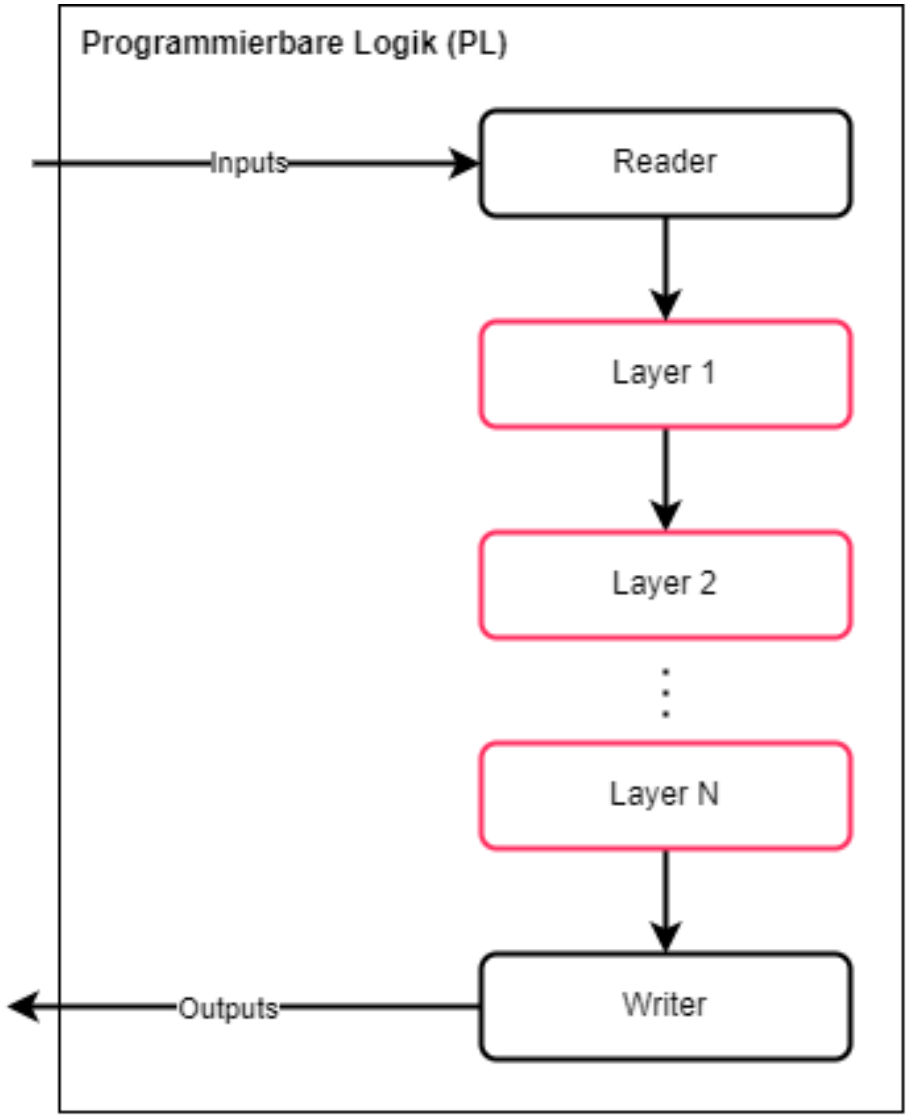
STREAMING ARCHITEKTUREN

- Berechnung von mehreren Layern durch mehrere Recheneinheiten

➔ **Schnell**

➔ **Hoher Ressourcenverbrauch**

➔ **Unflexibel**



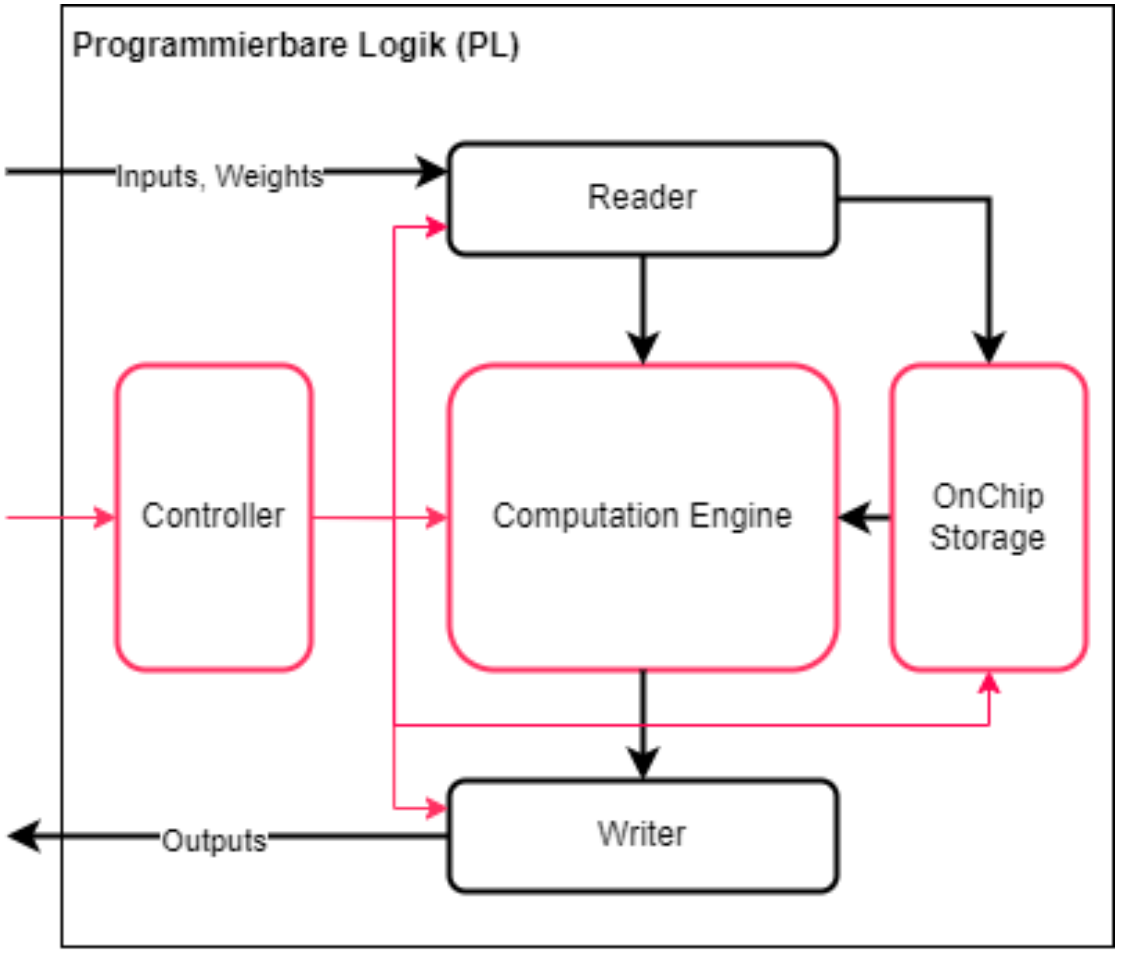
SINGLE ENGINE ARCHITEKTUREN

- Berechnung von mehreren Layern durch eine Recheneinheit

➔ **Langsamer**

➔ **Geringer Ressourcenverbrauch**

➔ **Flexibel**

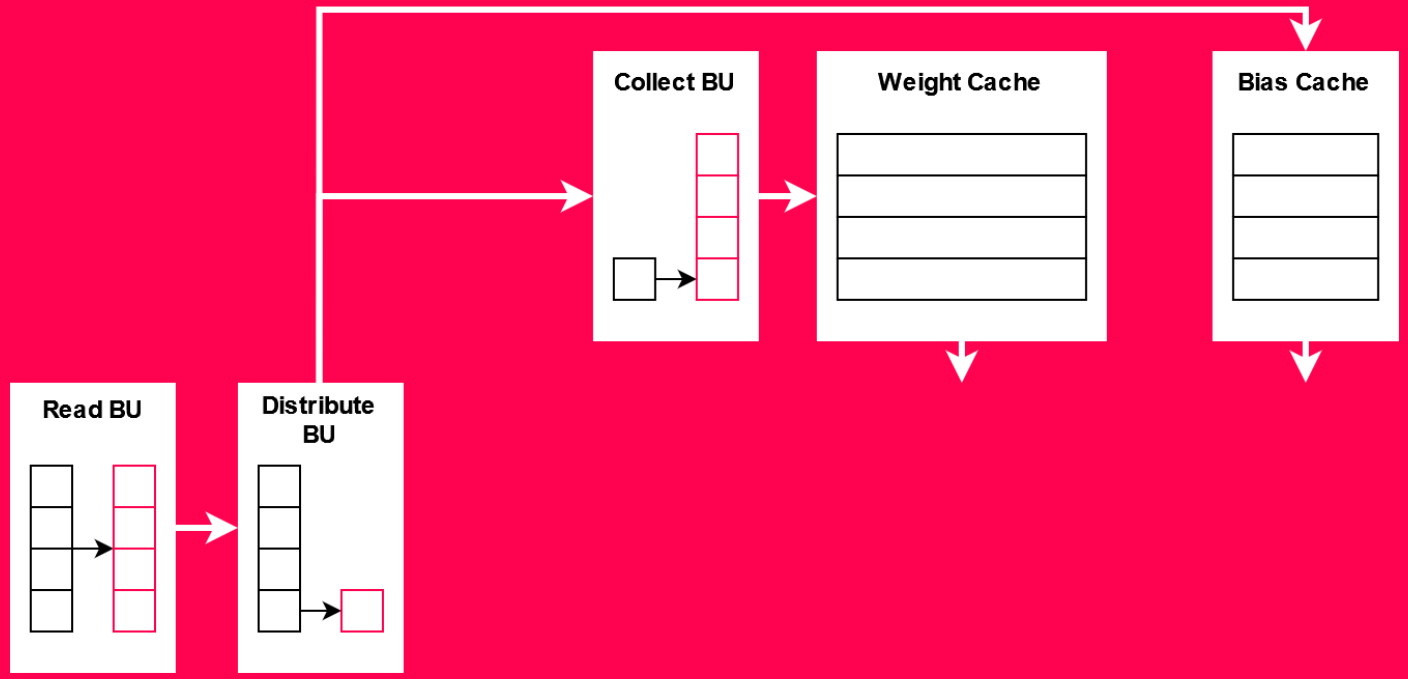


SINGLE ENGINE CORE
ZIELE

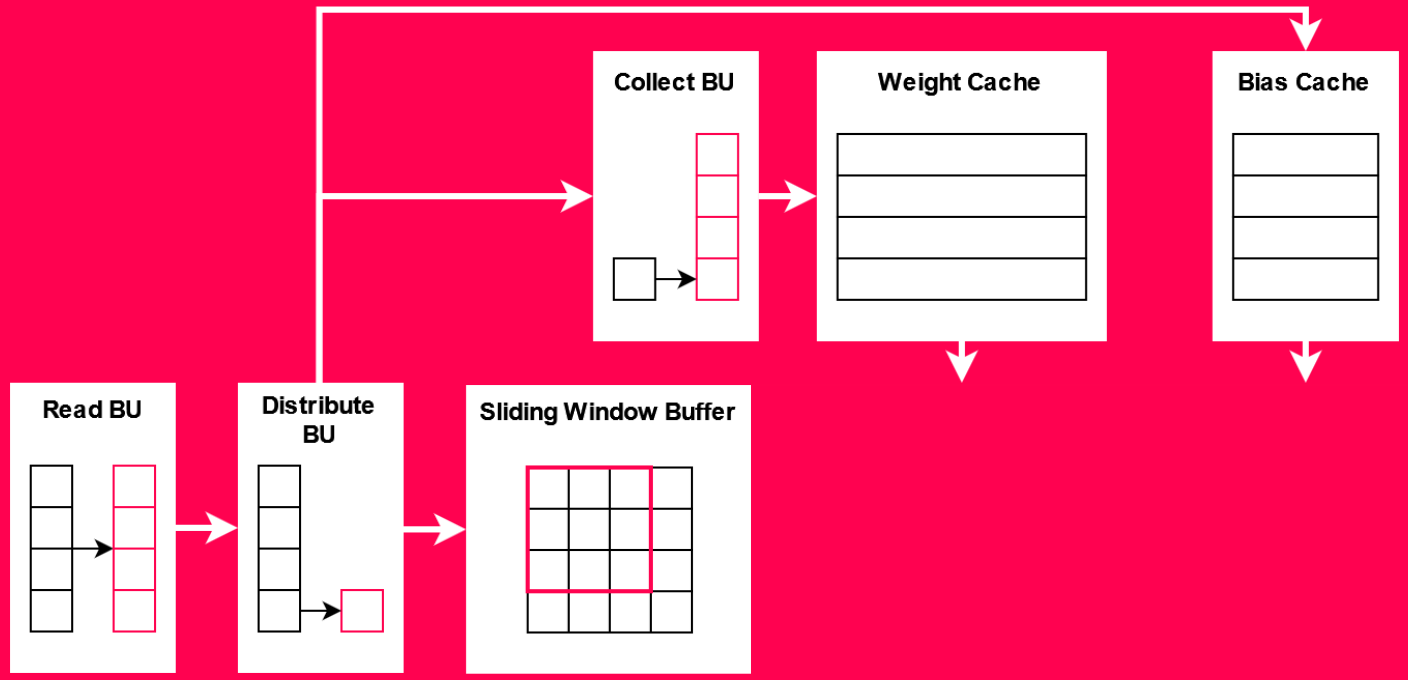
- Flexibel für unterschiedliche Schichttypen
- Gut kombinierbar

  ➔ Geringer Verbraucht an Logik Gattern
- Anpassbare Funktionalität und Ressourcenverbrauch

SINGLE ENGINE CORE

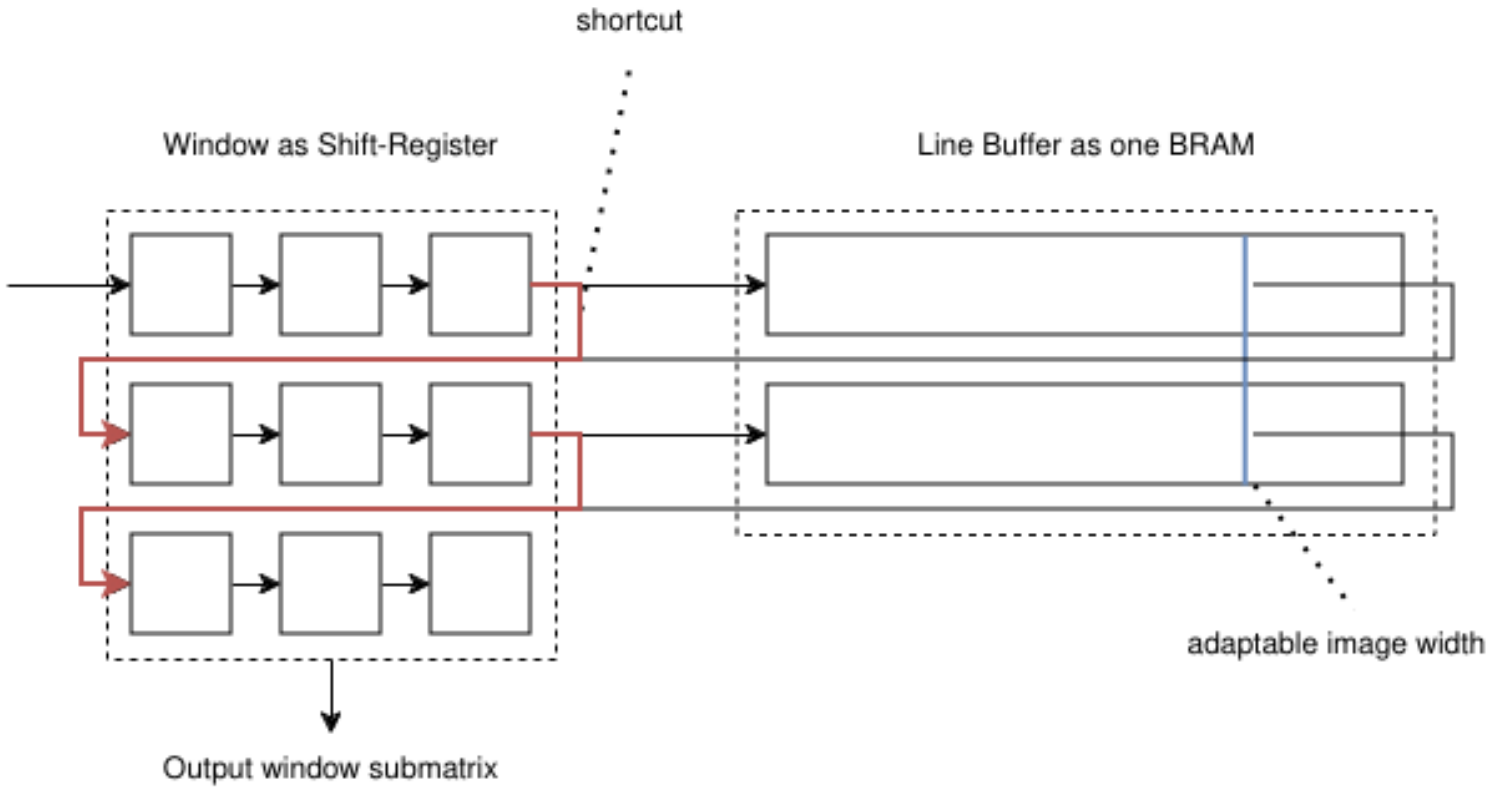


SINGLE ENGINE CORE

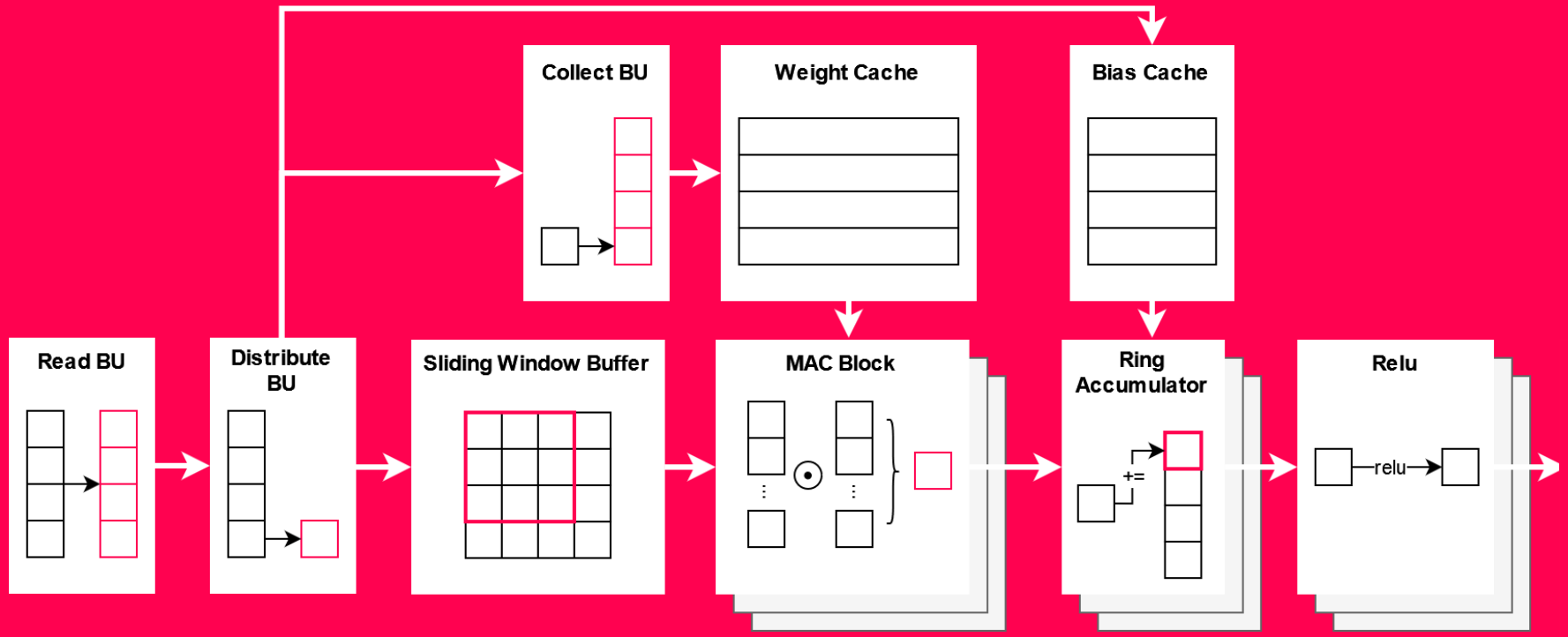


SINGLE ENGINE CORE
SLIDING WINDOW BUFFER

- Für Konvolutionen und Fully Connected Schichten
- Geringer Ressourcen-verbrauch

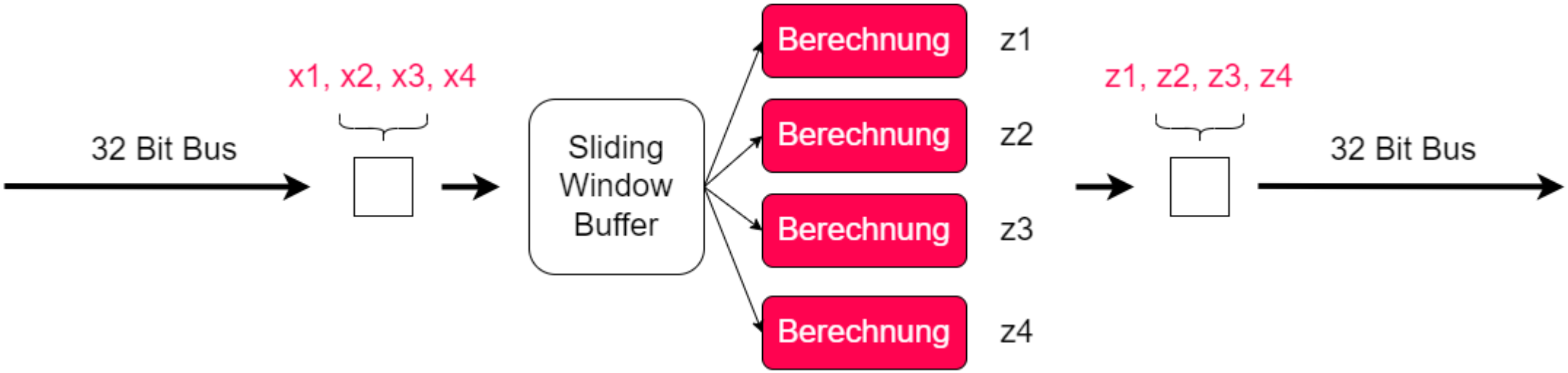


SINGLE ENGINE CORE

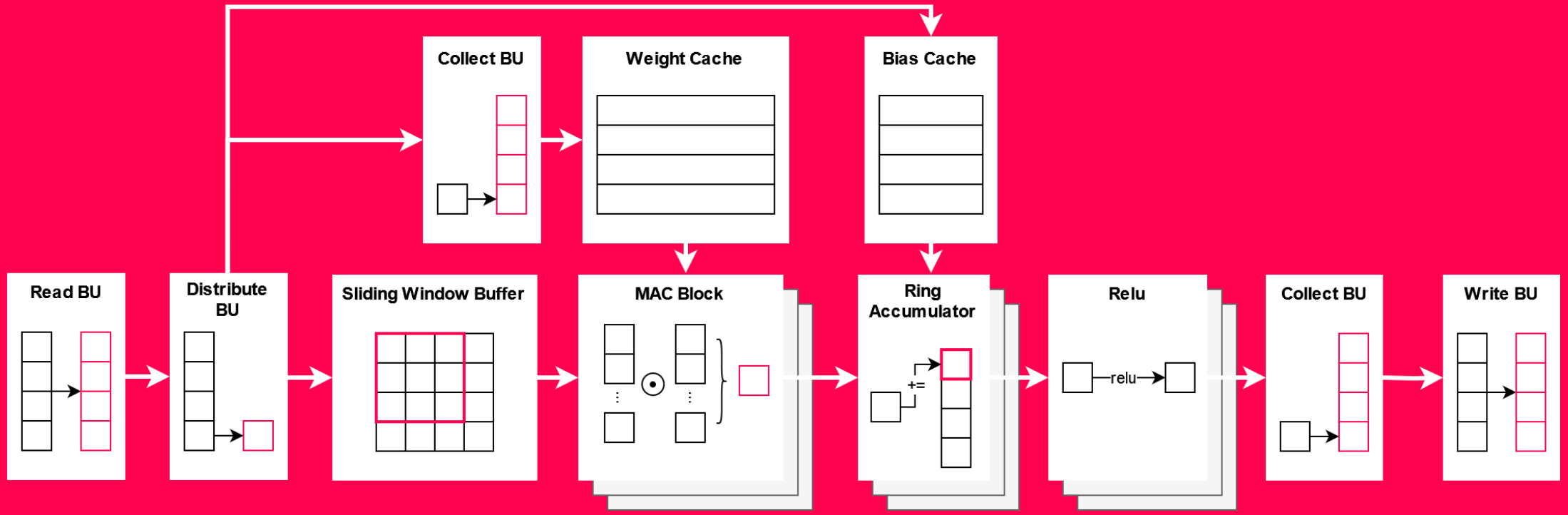


SINGLE ENGINE CORE
BUS BASIERTE SKALIERUNG

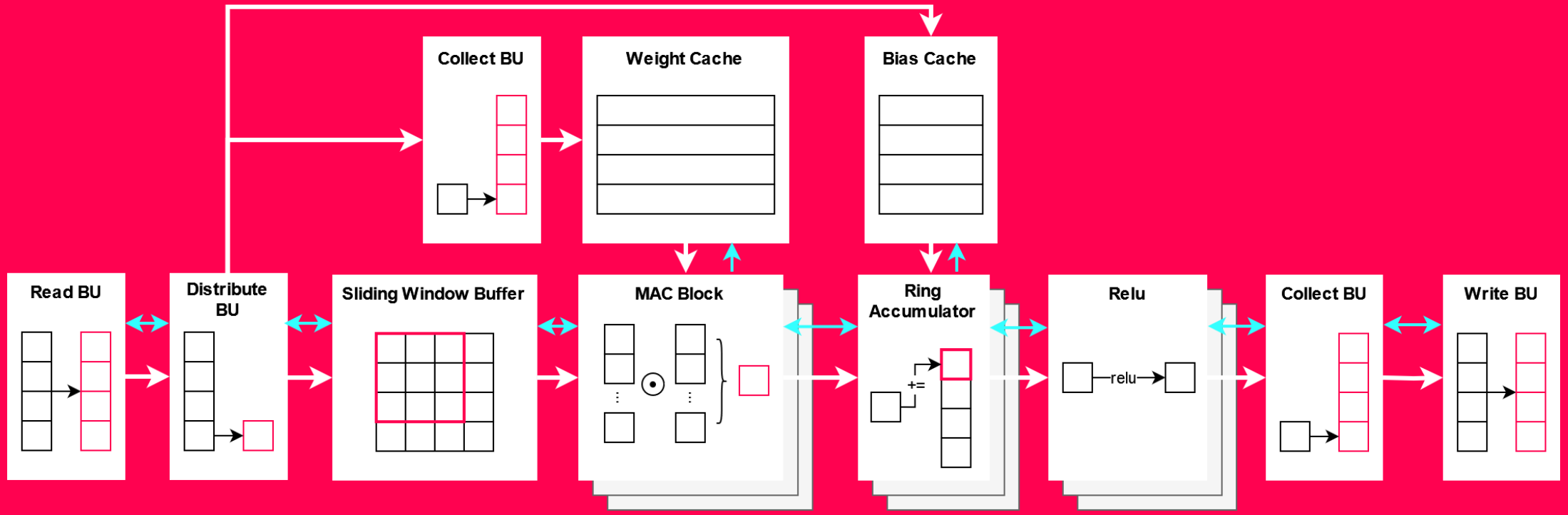
- Skalierung der Parallelisierung auf Basis der Busbreite des FPGAs



SINGLE ENGINE CORE

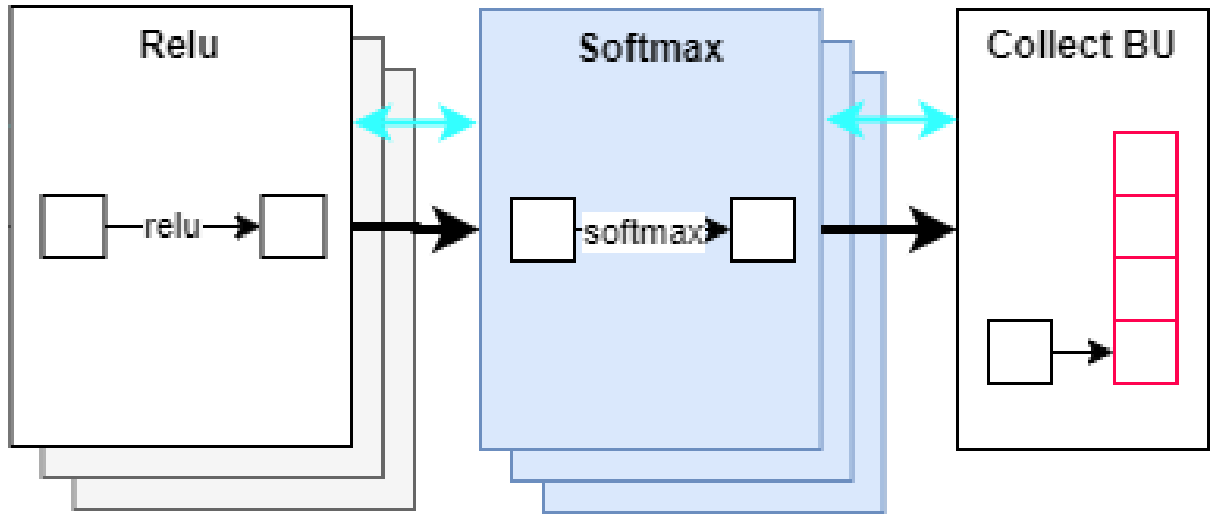


SINGLE ENGINE CORE

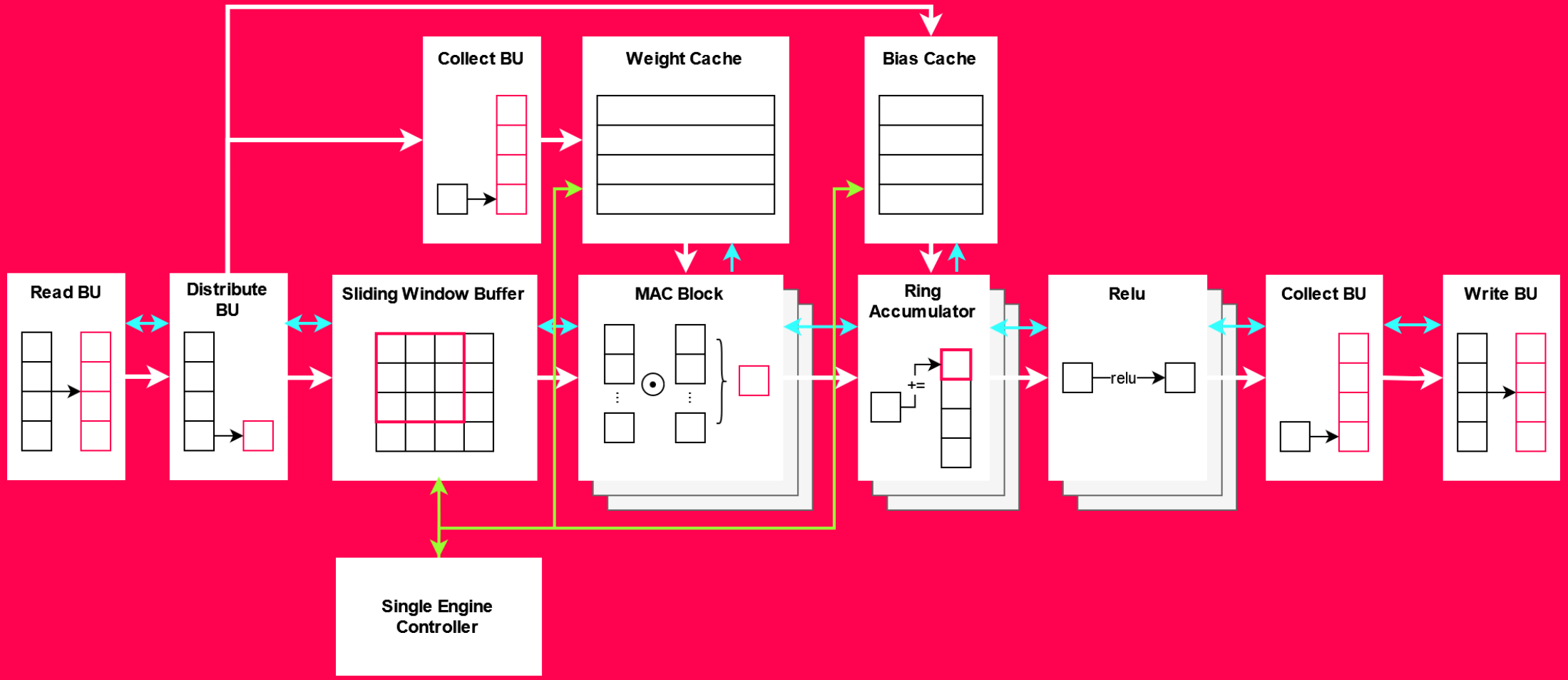


SINGLE ENGINE CORE
ERWEITERBARKEIT

- Konfigurierbarere Einreihung neuer Module



SINGLE ENGINE CORE



SINGLE ENGINE CORE
FEATURES

- Berechnung von Fully Connected Layern und Convolutions über dieselben **Hardwareeinheiten**
- Geringer Verbrauch von **Logik Resourcen**
- **Bus basierte Skalierung**
- Einfach **erweiterbar** dank einheitlicher Interfaces
- **Parametrisiertes** Design

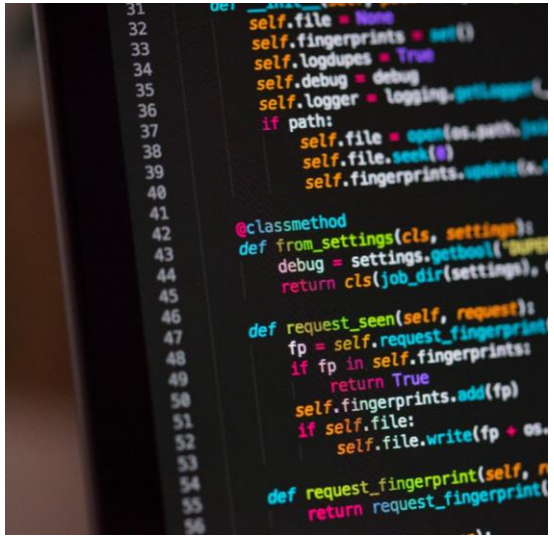
GENERIERUNG DER HARDWARE

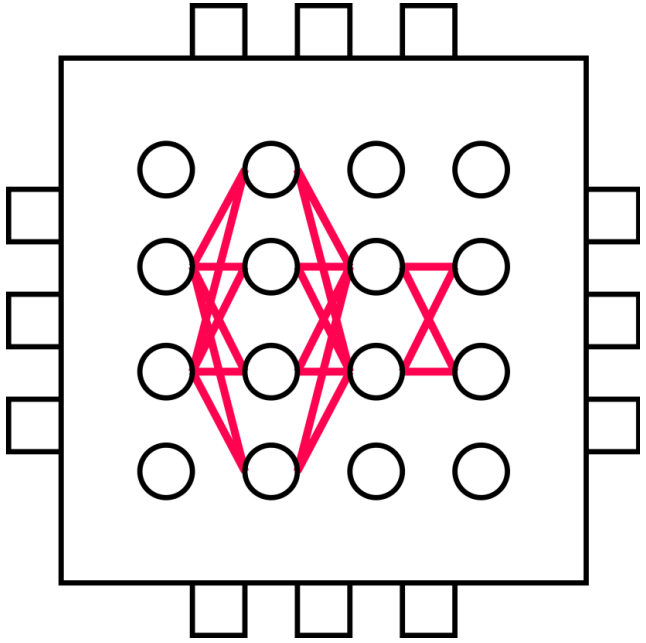
	LeNet	VGG-16
Input Bitbreite	5	7
Gewichte Bitbreite	5	7
Parallelisierungsfaktor	2	4
Bildgröße	28x28	224x224
Filtergröße	5x5	3x3
Speichereinheiten	20	713642
Pipelinetiefe	5	9

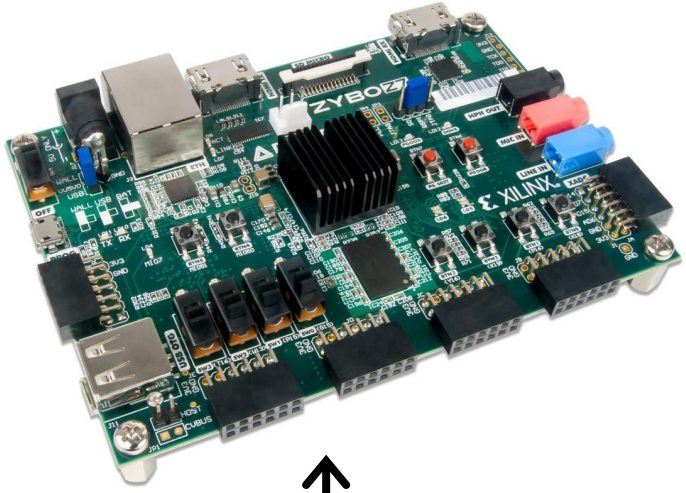
Technische Hochschule Augsburg

1. Neuronale Netze und Optiemierungen
2. THANNA Quantizer
3. THANNA Prozessor
4. THANNA Treiber

THANNA FRAMEWORK







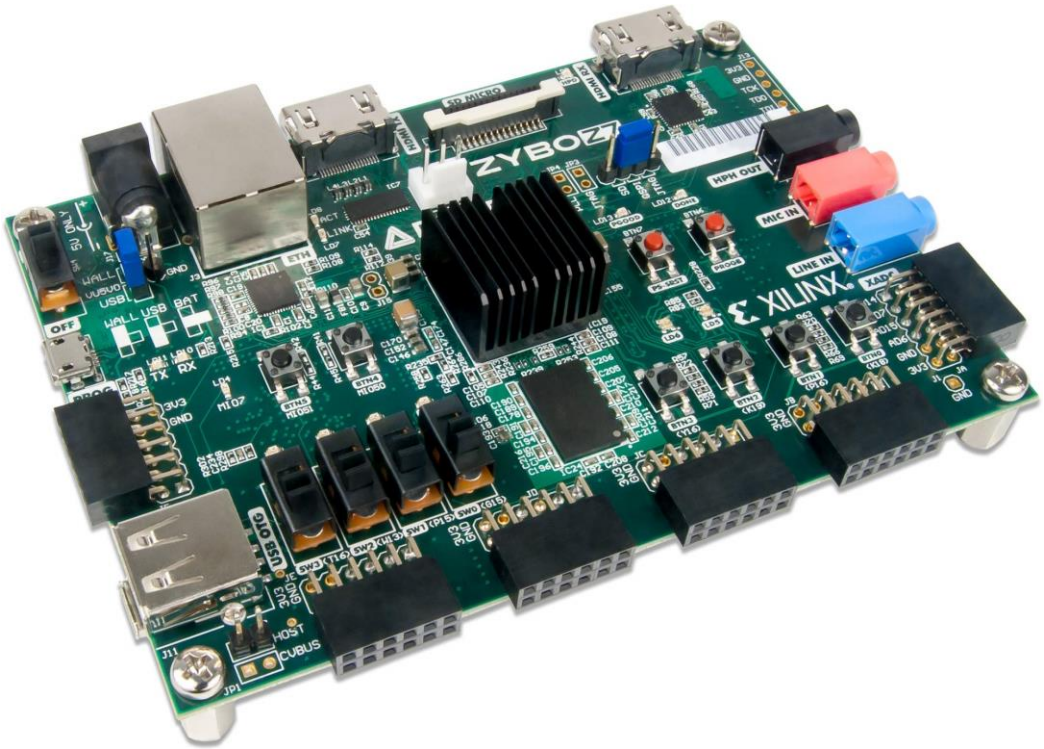
THANNA Quantizer

THANNA Prozessor

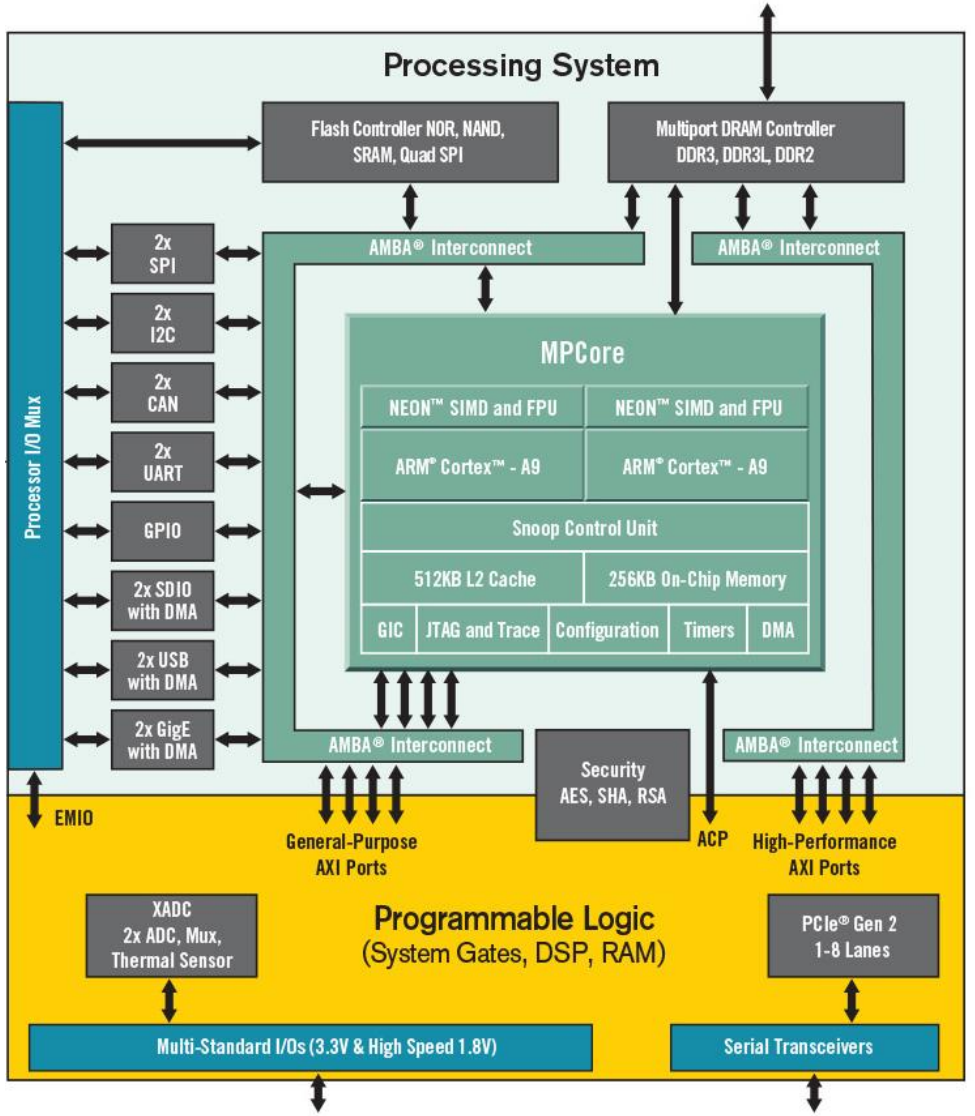
PetaLinux
THANNA Treiber
THANNA Library

Quelle: diligent.com

DAS ZYBO Z7 MIT ZYNQ-7020



Quelle: digilent.com



Quelle: xilinx.com

WARUM LINUX VERWENDEN?

- Full-Stack Python/Tensorflow Lite
- Treiber für Peripherie (Kamera, Tastatur)
- Einfaches Debuggen der Hardware

→ **Weniger Entwicklungsaufwand**

LINUX TREIBER

KERNEL TREIBER

THANNA LIBRARY

- Umsetzung über Character-Device
- Steuerung des THANNA-Cores über Register
- DMA für Gewichtsmatrizen

- C-Library
- Python Wrapper

DEMONSTRATOR

PROJEKTSTAND

✓THANNA Quantizer aus QKeras Code

✓THANNA Prozessor mit Convolution, Fully-Connected und Relu

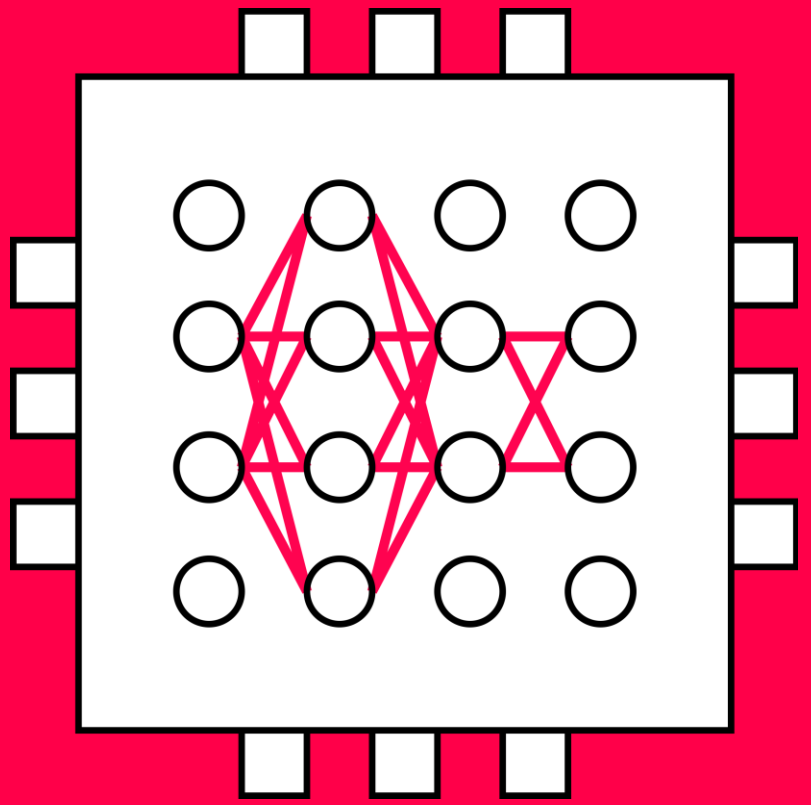
✓THANNA Treiber

PROJEKTSTAND

- x Post Training Quantisierung mit Trainingsdaten
- x Pooling & mehr Layer in THANNA Prozessor
- x Pruning Unterstützung in THANNA Prozessor

Technische Hochschule Augsburg

THANNA



Technische Hochschule Augsburg

Neural Network Accelerator

THANNA - Maximilian Schunck, Korbinian Gast, Philipp Urlbauer, Michael Schäferling, Prof. Dr. Gundolf Kiefer