TECHNISCHE UNIVERSITÄT DRESDEN

HZDR HELMHOLTZ ZENTRUM DRESDEN ROSSENDORF

DRESDEN concept

Jan Stephan
Fakultät Informatik // Institut für Technische Informatik
Seniorprofessor Dr.-Ing. habil. Rainer G. Spallek

Entwicklung eines SYCL-Backends für die Alpaka-Bibliothek und dessen Evaluation mit Schwerpunkt auf FPGAs

Verteidigung der Diplomarbeit

17. Dezember 2019

Gliederung

Motivation und Ziel

FPGAs als Beschleuniger
- Einsatzzwecke
- Programmierung

Die SYCL-Spezifikation

Die Alpaka-Bibliothek

Implementierung des SYCL-Backends
- Struktur
- Probleme

Ergebnisse
- Nutzbarkeit
- Performanz

Fazit

Motivation und Ziel

Motivation

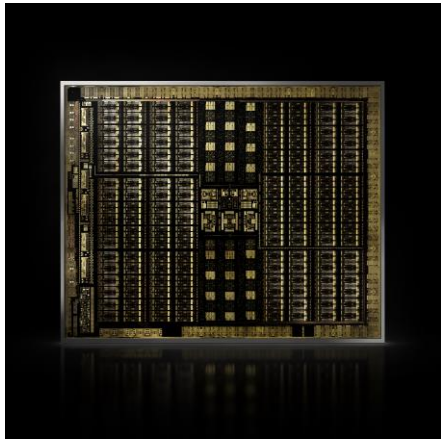
apple.com/de/shop/buy-iphone/iphone-11-pro/

https://www.hpe.com/de/de/servers/tower-servers.html

https://www.rewards.sony.com/ps4-game-system-1tb-pro/3003346.html

lenovo.com/de/de/laptops/thinkpad/t-series/T495/p/22TP2TTT495

newsroom.intel.de/news-releases/intel-announces-worlds-best-gaming-processor-new-9th-gen-intel-core-i9-9900k



nvidia.com/de-de/design-visualization/technologies/turing-architecture/

https://www.extremetech.com/extreme/171375-reverse-engineered-ps4-apu-reveals-the-consoles-real-cpu-and-gpu-specs

Motivation

GPUs sind Beschleuniger!

— Spezialisierte Hardware

— Datenparallele Algorithmen

Weitere Beschleunigertypen:

— CPUs mit besonders vielen Kernen (AMD EPYC)

— GPU-ähnlich (NVIDIA Tesla)

— Hybride (Intel Xeon Phi)

Problem: Allzweck-Hardware

— Nicht für jede Anwendung ideal

Motivation

Lösung: problemspezifische Hardware?

— *Application-specific integrated circuit* (ASIC)

— Hohe Produktionskosten → erst ab hohen Stückzahlen interessant

— Fertige Hardware schwer anpassbar

Dynamisch konfigurierbare Hardware

— *field programmable gate array* (FPGA)

— Frei verschaltbare Komponenten (Logikfunktionen, Speicher, …)

— Schaltungen beliebig austauschbar

— Langsamer als ASICs

— Synthese: Algorithmus → Schaltung

Motivation

Problem: Programme nicht portabel

— GPUs: CUDA (seit 2005, nur NVIDIA)

— CPUs: OpenMP (seit 1997) & Intrinsiken

— FPGAs: Hochsprachen-Dialekte (seit 2011)

developer.nvidia.com/language-solutions

openmp.org

xilinx.com/products/design-tools/vivado.html

Khronos: einheitliche API-Standards

— 2008: *Open Computing Language* (OpenCL)

— 2015: SYCL (FPGA-Unterstützung)

khronos.org/blog/opencl-2.2-maintenance-update-released

khronos.org/assets/utilities/retrieveFile.php?d=sycl&t=logopacks

Abstrahierende Bibliotheken

— Beispiel: Kokkos (seit 2014, keine FPGA-Unterstützung)

— Beispiel: Alpaka (seit 2015, keine FPGA-Unterstützung)

kokkos.org

github.com/ComputationalRadiationPhysics/alpaka

Ziel

Untersuchung des SYCL-Standards

— Existierende Implementierungen
— Nutzbarkeit
— FPGA-Schwerpunkt

Entwicklung eines Alpaka-SYCL-Backends

— Analyse der Schwierigkeiten und Unzulänglichkeiten
— Einschätzung der Leistungsfähigkeit
— Verifizierung durch reale Anwendung

FPGAs als Beschleuniger

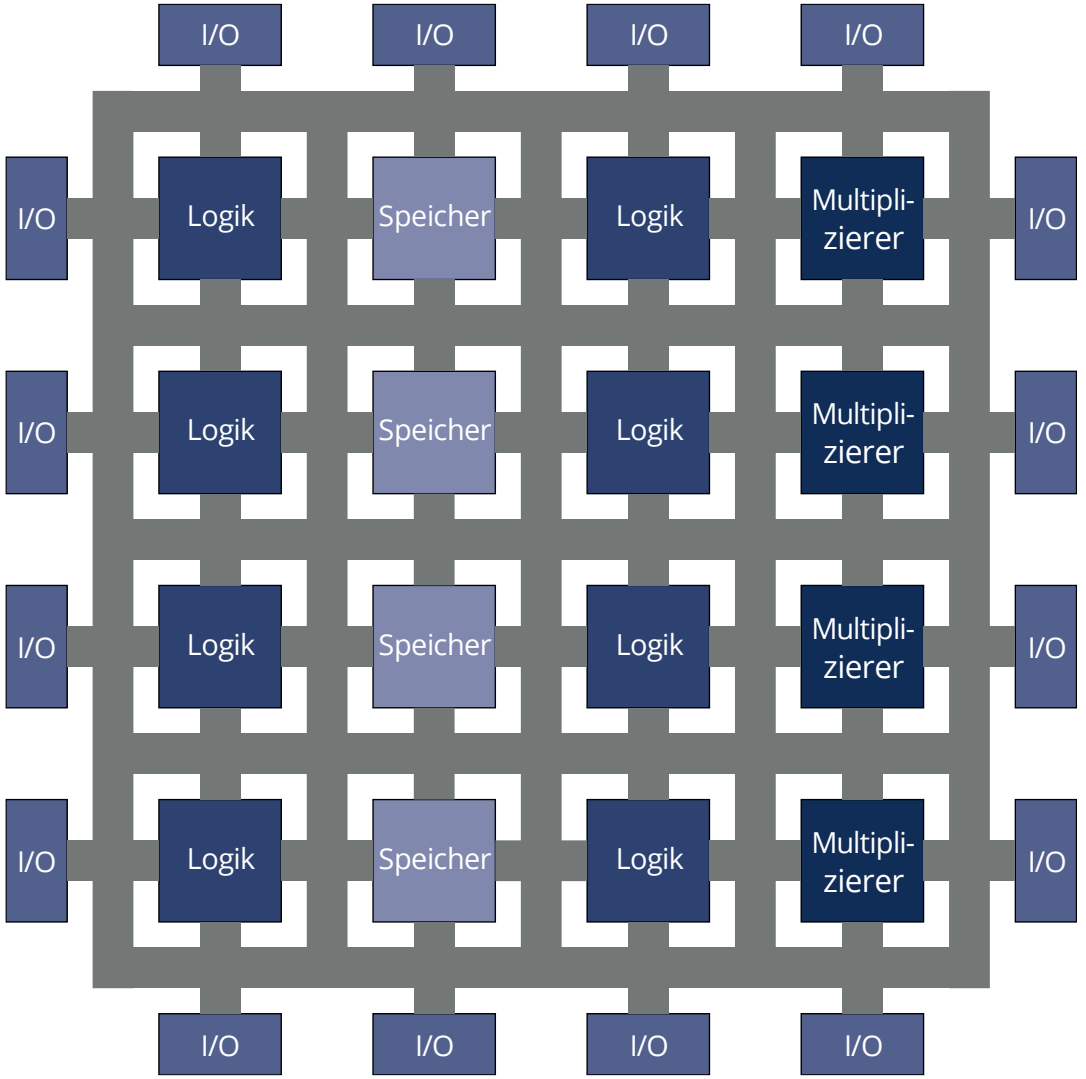
FPGAs als Beschleuniger

Aufbau

- Logikzellen mit geringer Komplexität
- Regelmäßige spaltenweise Feldstruktur
- Programmierbare Verdrahtungen
- Puffer für Ein- und Ausgabe (I/O)
- Weitere Elemente (Speicher, Multiplizierer, …)

Moderne FPGAs (Xilinx Virtex UltraScale+)

- *Configurable logic block* (CLB)
- *Input/output block* (IOB)
- Block RAM (36 kbit, paralleler Zugriff)
- UltraRAM (288 kbit, kein paralleler Zugriff)
- *Digital signal processor* (DSP)
- *Clock management tile* (CMT)



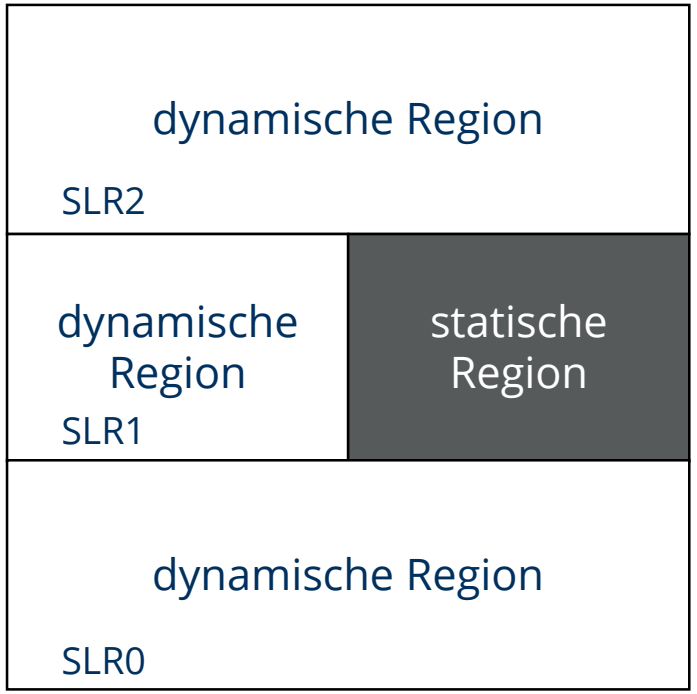
Nach [HS10], S. 10-14

FPGAs als Beschleuniger

Xilinx XCU200

- Aufteilung in drei *super logic regions* (SLR)
- Zugriff auf Off-Chip-Speicher (DRAM)
- Dynamische Region für Programmierer
- Statische Region für Laufzeitumgebung

- 111.500 CLBs
- 1706 Block RAMs
- 800 UltraRAMs
- 5867 DSPs



nach Xil19a, S. 5

FPGAs als Beschleuniger
Einsatzzwecke

- Schaltungsentwurf
  - *Rapid prototyping* für integrierte Schaltkreise
  - Einfache Fehlerbehebung
- Schaltkreise in kleiner Stückzahl
  - Einstiegskosten geringer als bei ASICs
  - Höhere Kosten pro Stück bei größeren Produktionsvolumen
- Microsoft: Inferenz tiefer neuraler Netzwerke [Fow+18, Chu+18]
  - Trainierte Netzwerke als Schaltung
- Microsoft: FPGAs als Netzwerkkarten [Fir+18]
- Amazon: FPGA-Cloud-Instanzen [Ama]
  - FPGAs als Beschleuniger [Di+17]

FPGAs als Beschleuniger
Programmierung

Die SYCL-Spezifikation

Die Alpaka-Bibliothek

Implementierung des SYCL-Backends

Ergebnisse

Ergebnisse
Nutzbarkeit // Übersicht

Implementierung	Nutzbarkeit mit Alpaka
ComputeCpp	Nicht nutzbar (fehlerhaft)
Intel	Nutzbar
Xilinx	Nicht nutzbar (fehlerhaft)
hipSYCL	Nicht nutzbar (unvollständig)
sycl-gtx	Nicht nutzbar (unvollständig)

Ergebnisse
Nutzbarkeit // ComputeCpp

Verwendete Version: ComputeCpp 1.1.5 Community Edition

Problem: Zeiger

```
template <typename T>
void f(T* ptr);

auto x = 42;
f(&x);  // void f(int* ptr);

auto ptr = sycl_accessor.get_pointer();
f(ptr); // void f(__global int* ptr);

std::is_same_v<int*, decltype(ptr)>;  // false
__global int* ptr2 = nullptr;         // Syntaxfehler
```

Ergebnisse
Nutzbarkeit // Xilinx

Verwendete Softwareversionen:

- github.com/triSYCL/sycl, `sycl/unified/next`-Zweig, Commit `#dfb95af`
- SDAccel 2019.1
- XRT 2.2
- `xilinx-u200-xdma` & `xilinx-u200-xdma-dev` für Ubuntu 18.04 (Version `201830.2-2580015`)

Problem #1: Mathematikfunktionen

- Compiler generiert fehlerhafte Instruktionen
- Manche Funktionen führen zu Compiler-Absturz
  - `rsqrt(double)`

Ergebnisse
Nutzbarkeit // Xilinx

Problem #2: benutzerdefinierte Strukturen

```
struct coord
{
    std::size_t x;
    std::size_t y;
};

struct kernel
{
    operator()()
    {
        auto c = coord{42, 42}; // Compiler-Absturz
    }
};
```

Ergebnisse
Performanz // Verifizierung

Verifizierung des Alpaka-Backends

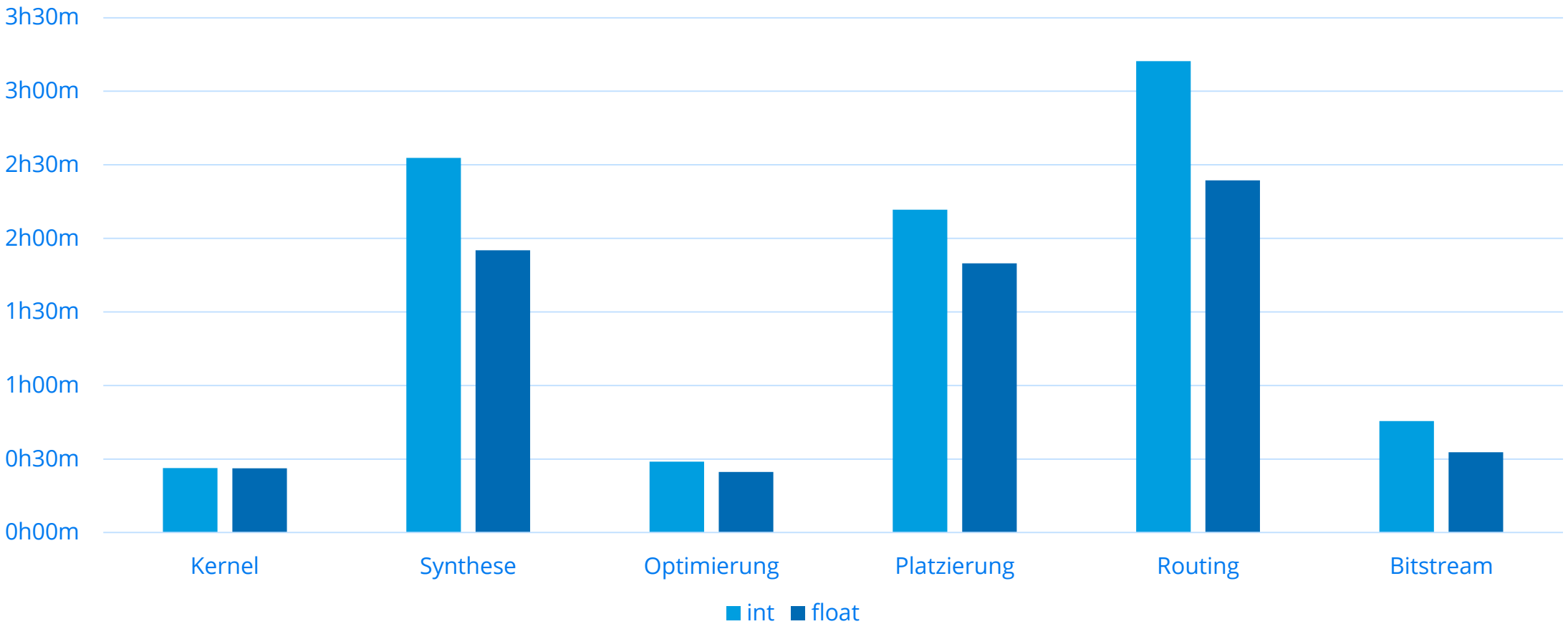
- Alpaka-Programm: *jungfrau-photoncounter*
- Photonenzähler für JUNGFRAU-Detektor (Paul Scherrer Institut, PSI)
  - 16 Megapixel
  - 2 Byte pro Pixel
  - Derzeitige Frequenz: 100 Hz (3,2 GB/s)
  - Zielfrequenz: 2,2 kHz (74 GB/s)

$$N_\gamma = \frac{\text{ADC} - \text{Sockel}}{\text{Verstärkung} \cdot E_\gamma}$$

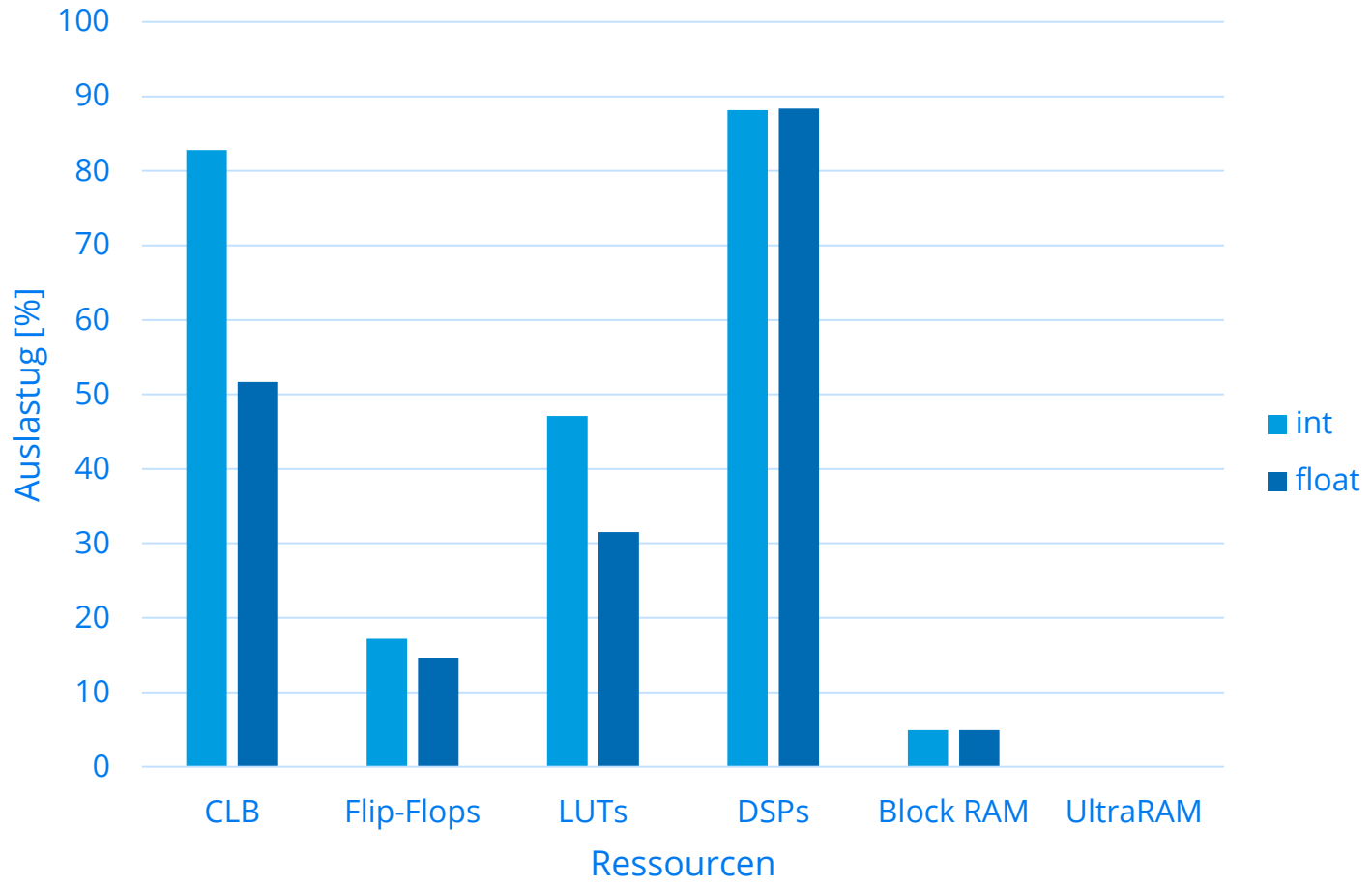
- N_γ: Anzahl der Photonen
- ADC: Messergebnis des Pixels
- Sockel: Grundrauschen des Pixels (engl. *pedestal*)
- Verstärkung: Signalverstärkung des Pixels (engl. *gain*)
- E_γ: Photonenenergie

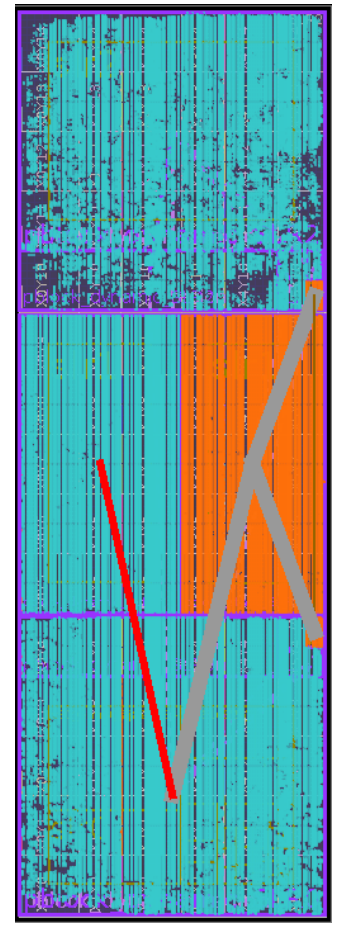
Ergebnisse
Performanz // Verifizierung

Ergebnisse
Performanz // Compile-Zeit

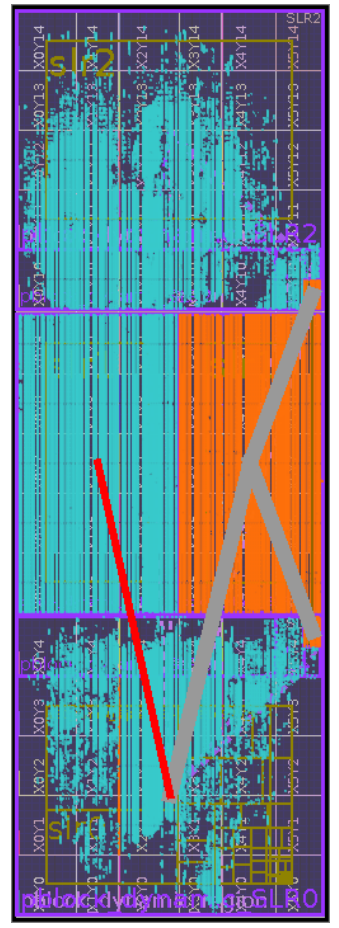


Ergebnisse
Performanz // Ressourcenverbrauch





int



float

Fazit

Literatur

Literatur

[Ama] Amazon Web Services, Inc. *Amazon EC2 F1-Instances*. URL: https://aws.amazon.com/de/ec2/instance-types/f1/ (besucht am 22.11.2019)

[Chu+18] Eric Chung u.a. „Serving DNNs in Real Time at Datacenter Scale with Project Brainwave“. In: *IEEE Micro* Jahrgang 38. Ausgabe 2 (März 2018), S. 8 – 20. DOI: 10.1109/MM.2018.022071131

[Di+17] Lorenzo Di Tucci u.a. „The Role of CAD Frameworks in Heterogeneous FPGA-Based Cloud Systems“. In: IEEE 35th International Conference on Computer Design. Nov. 2017, S. 423 – 426. DOI: 10.1109/ICCD.2017.75

[Fir+18] Daniel Firestone u.a. „Azure Accelerated Networking: SmartNICs in the Public Cloud“. In: *15th USENIX Symposium on Networked Systems Design and Implementation*. Apr. 2018, S. 51 – 64.

[Fow+18] Jeremy Fowers u.a. „A Configurable Cloud-Scale DNN Processor for Real-Time AI“. In: *Proceedings of the 45th Annual International Symposium on Computer Architecture*. Juni 2018, S. 1 – 14. DOI: 10.1109/ISCA.2018.00012

[HS10] Charles Hawkins und Jaume Segura. *Introduction to Modern Digital Electronics.* Preliminary Edition. SciTech Publishing, Inc., 2010. ISBN: 978-1-891-12107-4

Literatur

[Xil19a] Xilinx, Inc. *Alveo U200 and U250 Data Center Accelerator Cards Data Sheet*. DS962 (v1.1). Xilinx, Inc. 2100 Logic Drive, San Jose, CA 95124, Vereinigte Staaten von Amerika, Juni 2019.