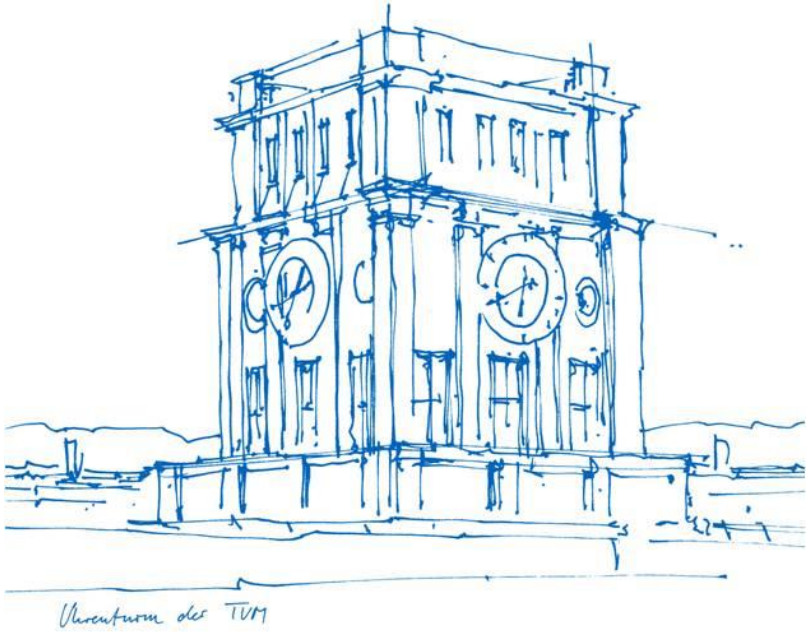
Abschlusspräsentation ERA-GP

Implementierung von Blockchain-Operationen auf einem FPGA

Eke Timur, Schapeler Nicolas,

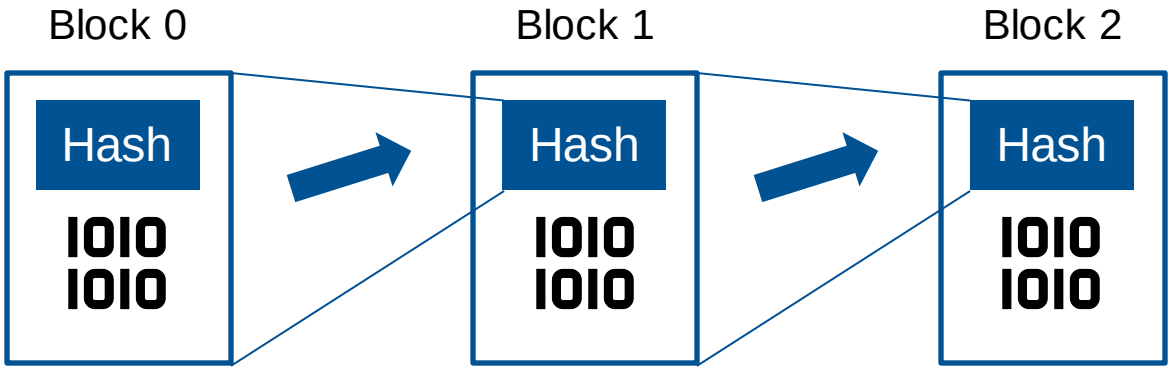
Starnecker Christoph, Weiser Florian

Garching, 01. Februar 2021



Blockchain

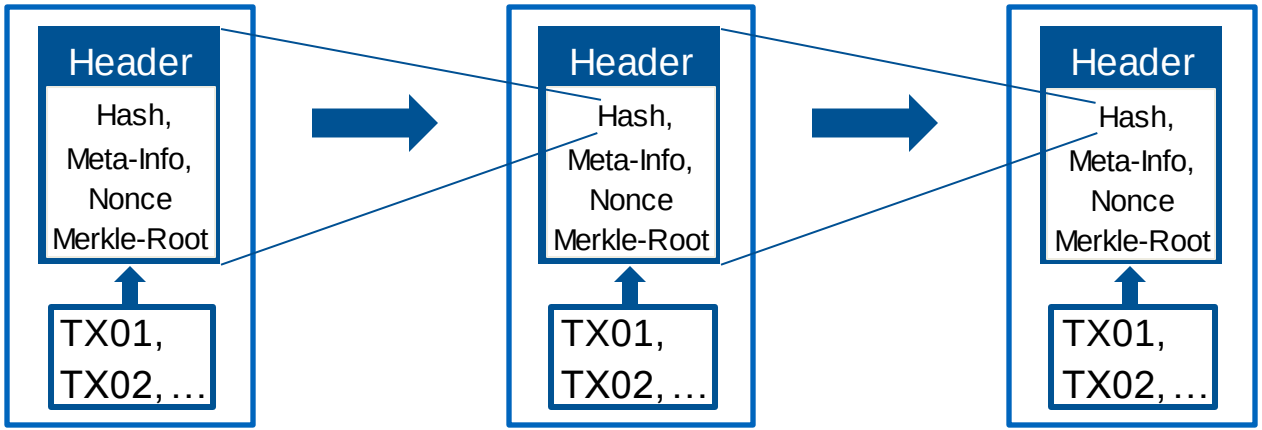
- Kette von kryptographisch verlinkten Informationsblöcken
- Verifizierung der Blöcke



Bitcoin

- Entwickelt 2008 von „Satoshi Nakamoto“

- **Blockchain**: *Verteiltes Kassenbuch* (Distributed Ledger)
- **Blockdaten**: *Transaktionen*
- **Verifizierung**: *Proof-of-Work*



Proof-of-Work

- Verifizierung der Blöcke durch Arbeitsaufwand
- Durchschnittlich hoher Aufwand für jede Blockerzeugung
  → Absicherung gegen böswillige Veränderungen
- Umsetzung durch besondere Anforderung an Block Hash
- Ausnutzung der *preimage resistance* des SHA-256

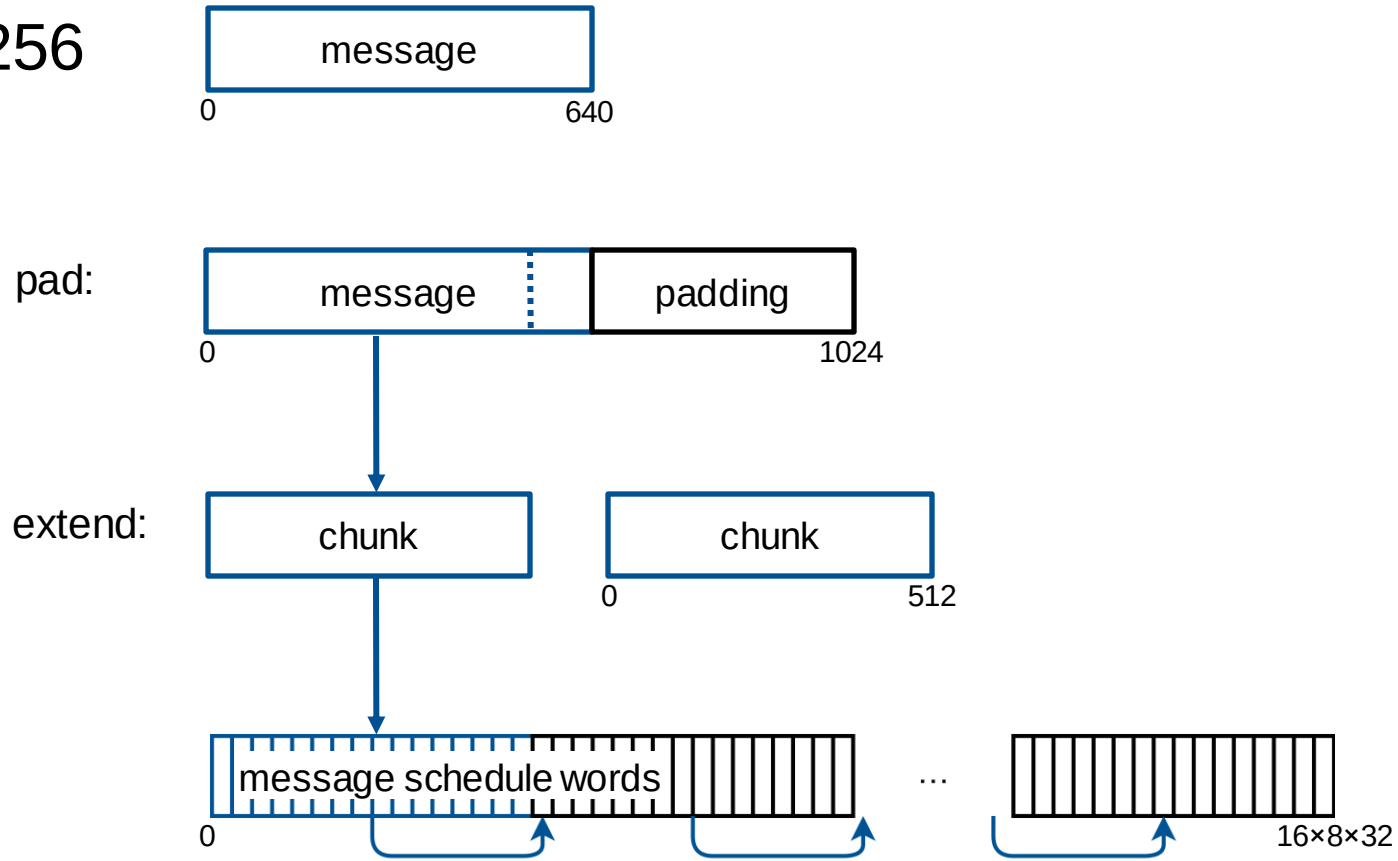
  →Target Value

  `00000000000000000000196feeea6621e209a58293f4d5648c755ce04bfa0010`

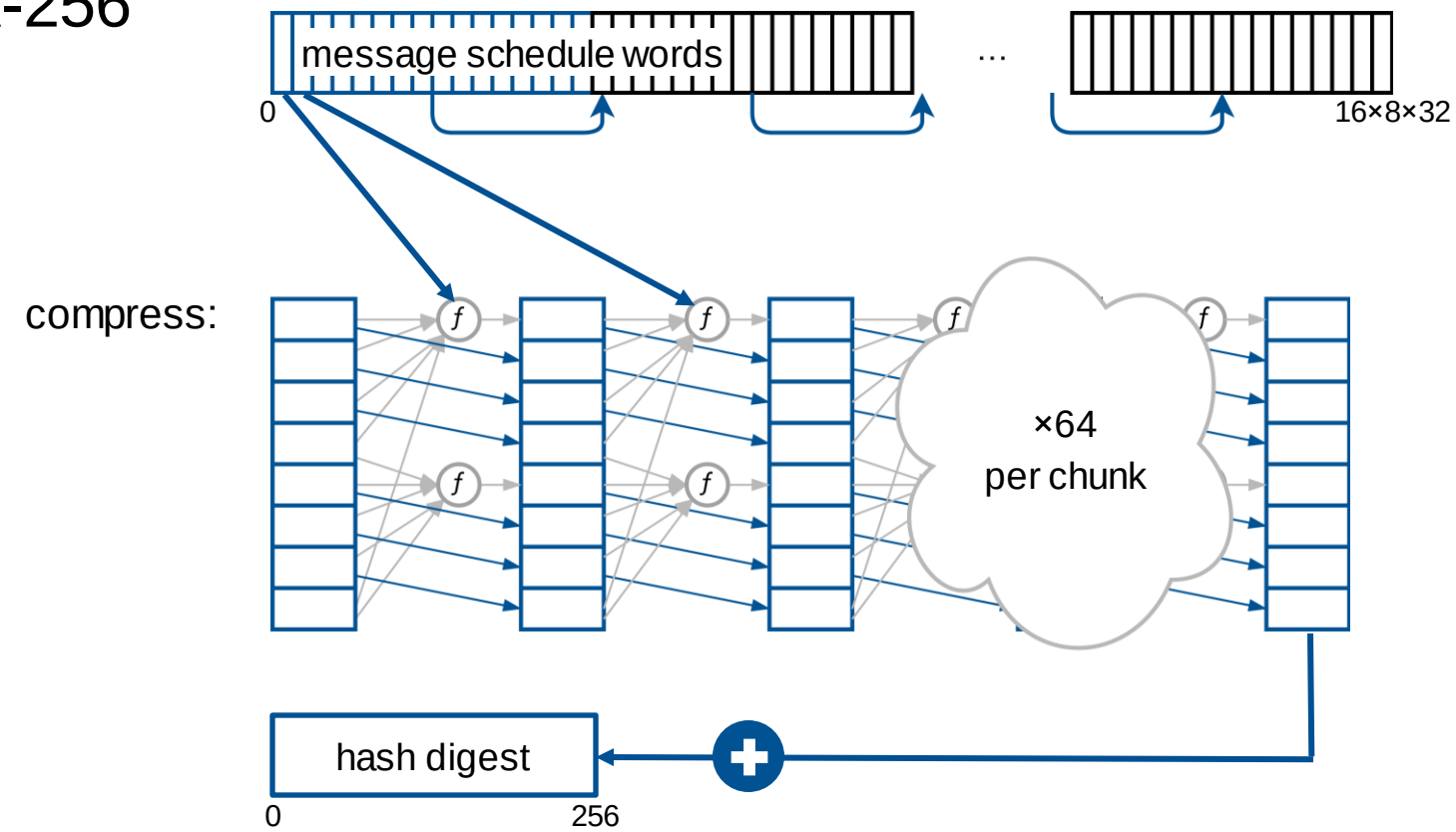
Unser Ziel

- **Effiziente Suche nach einem unter dem Target liegenden Hash mit einem FPGA**
- Implementierung des *SHA-256(SHA-256(Block-Headers)) (SHA-256d)*
- Änderung der **nonce & timestamp** innerhalb des Blockheaders
  - Bereitstellung der Header durch einen Host

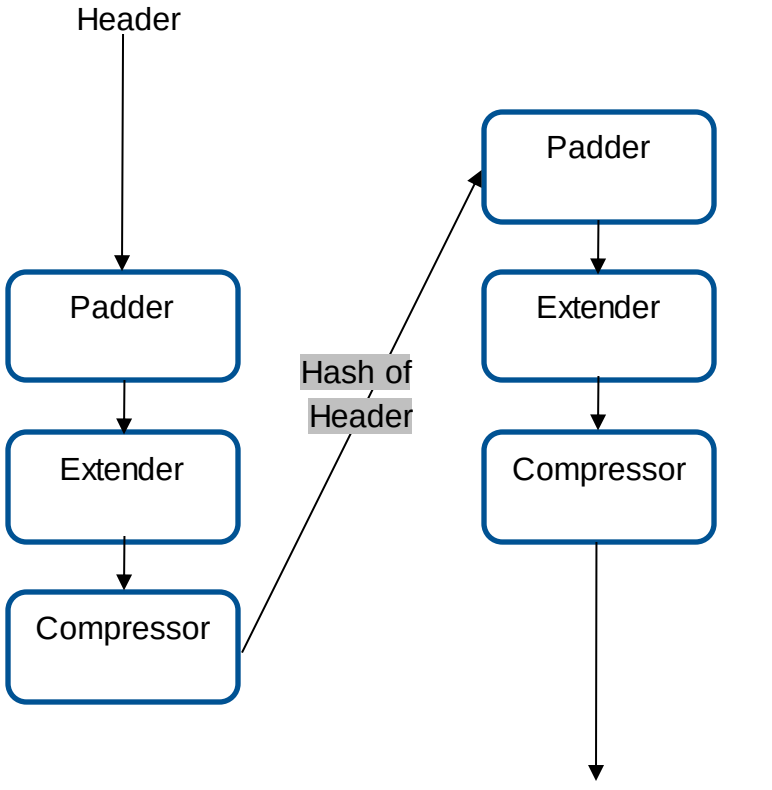
SHA-256


message
0
640
pad:
message
padding
0
1024
extend:
chunk
chunk
0
512
message schedule words
…
0
16×8×32

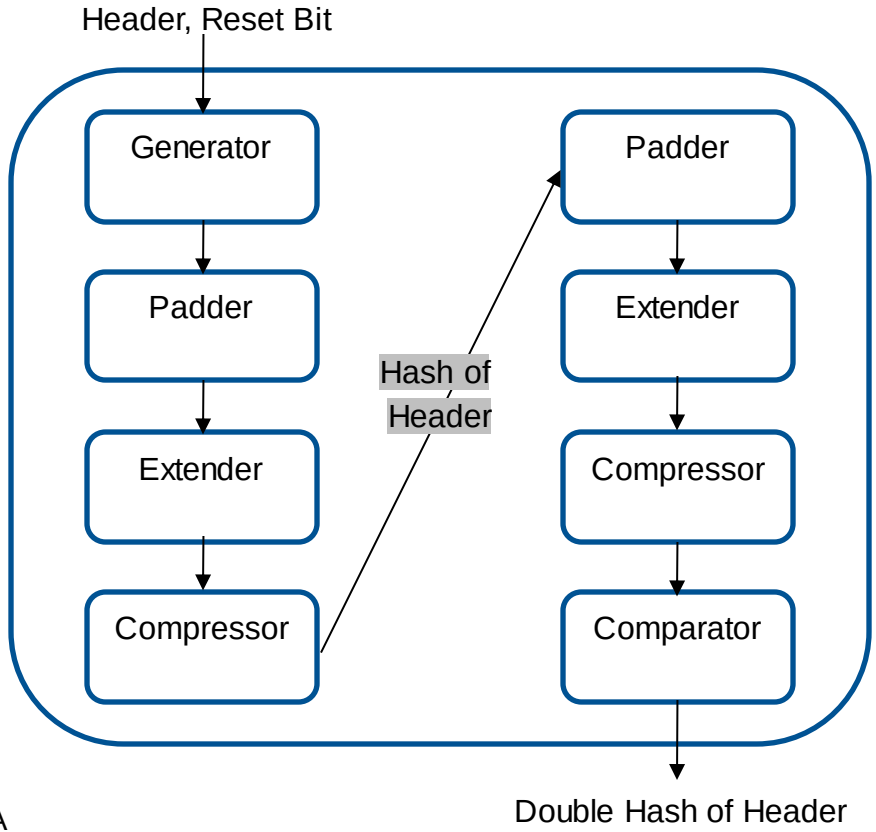
SHA-256


message schedule words
0
...
16×8×32
compress:
f
×64
per chunk
hash digest
0
256

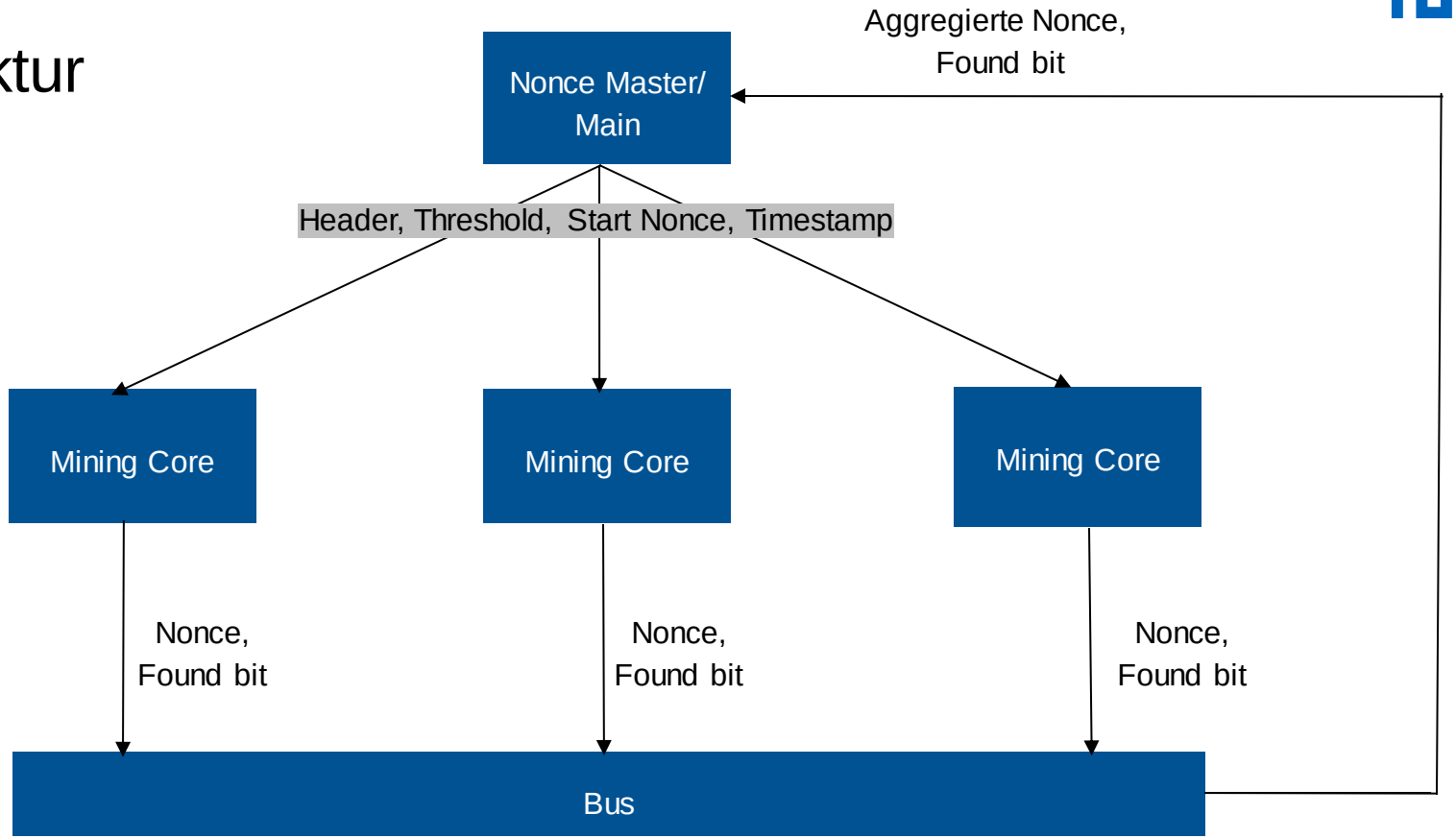
SHA-256


Header
Padder
Extender
Compressor
Hash of
Header
Padder
Extender
Compressor
Double Hash of Header

Erweiterung von SHA-256 zur Mining Core



Architektur



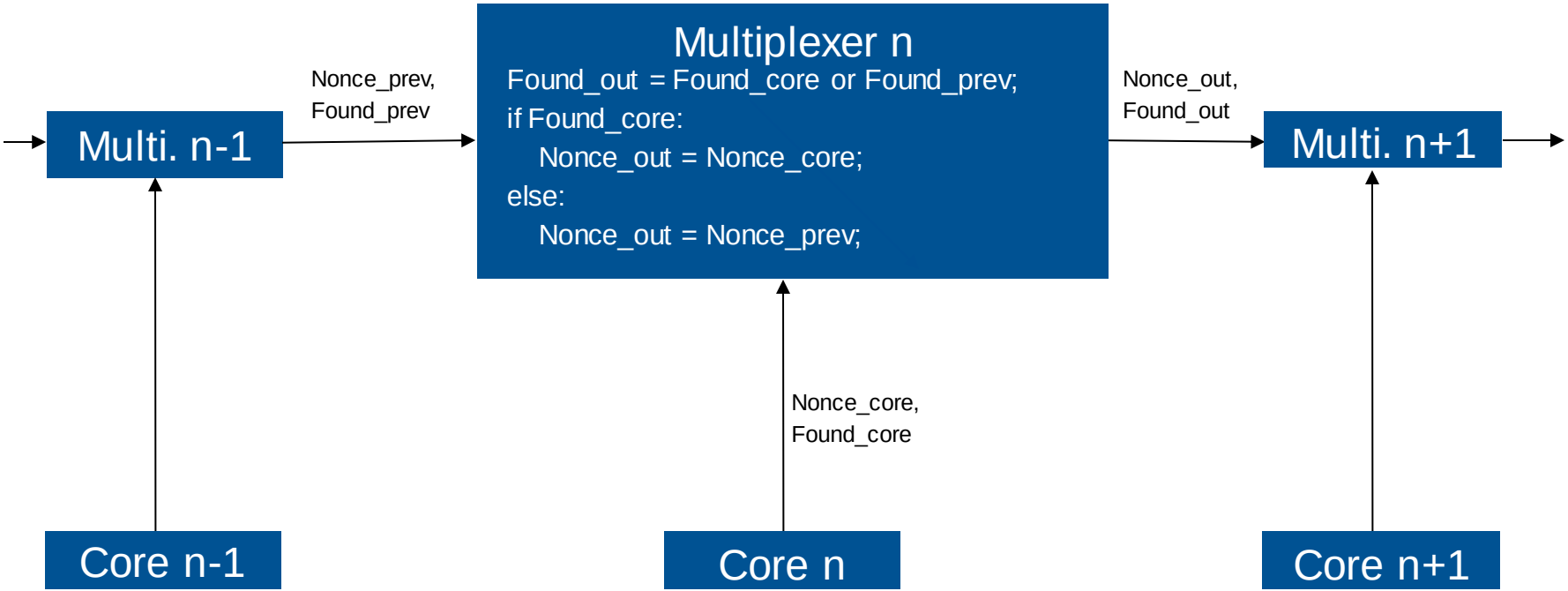
Wie werden Nonces verteilt?

- Jede Nonce gleichwahrscheinlich

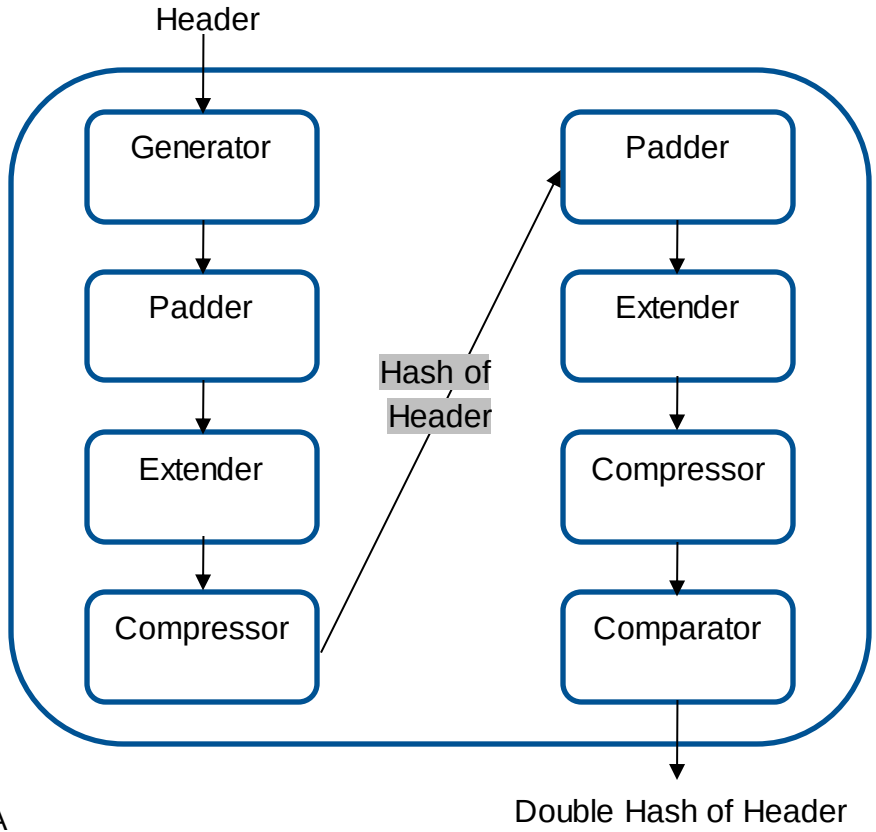
- Wenig Sinn bei einer bestimmten Nonce anzufangen / besondere Schrittgröße zu machen
  → Alle Nonces durchprobieren

- Start: Modulo Reste von der Anzahl Mining Cores
- Step: Anzahl von Mining Cores

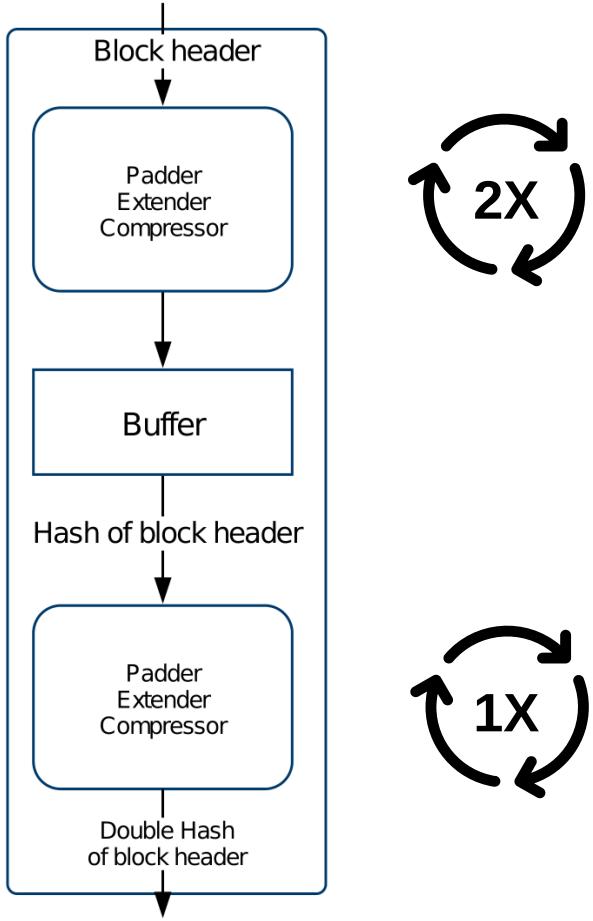
Synchronisation



Einfaches Pipelining

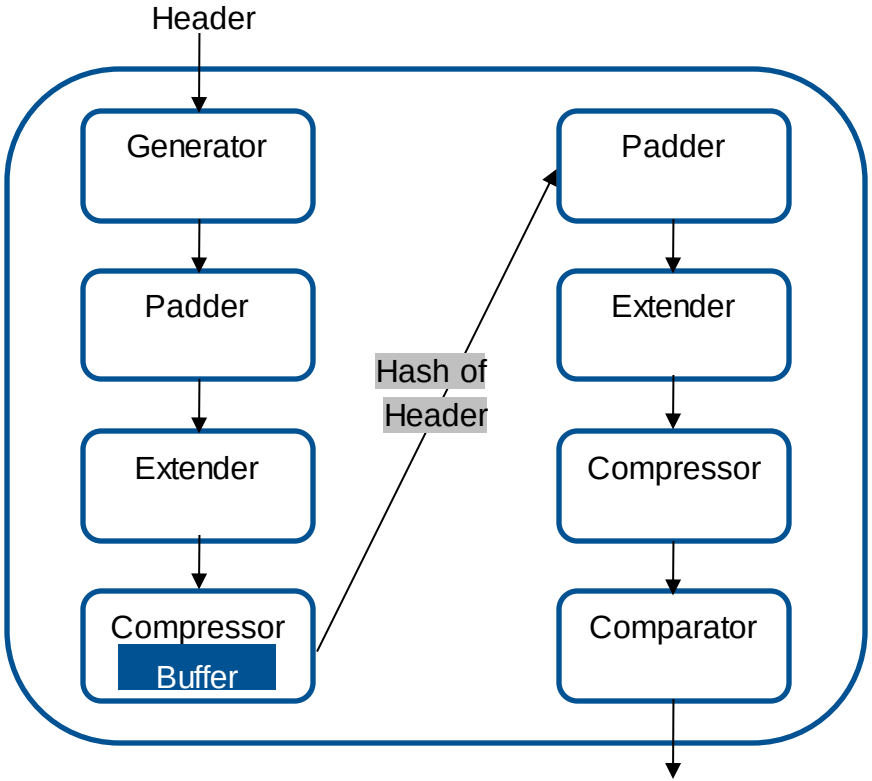


Pipelining

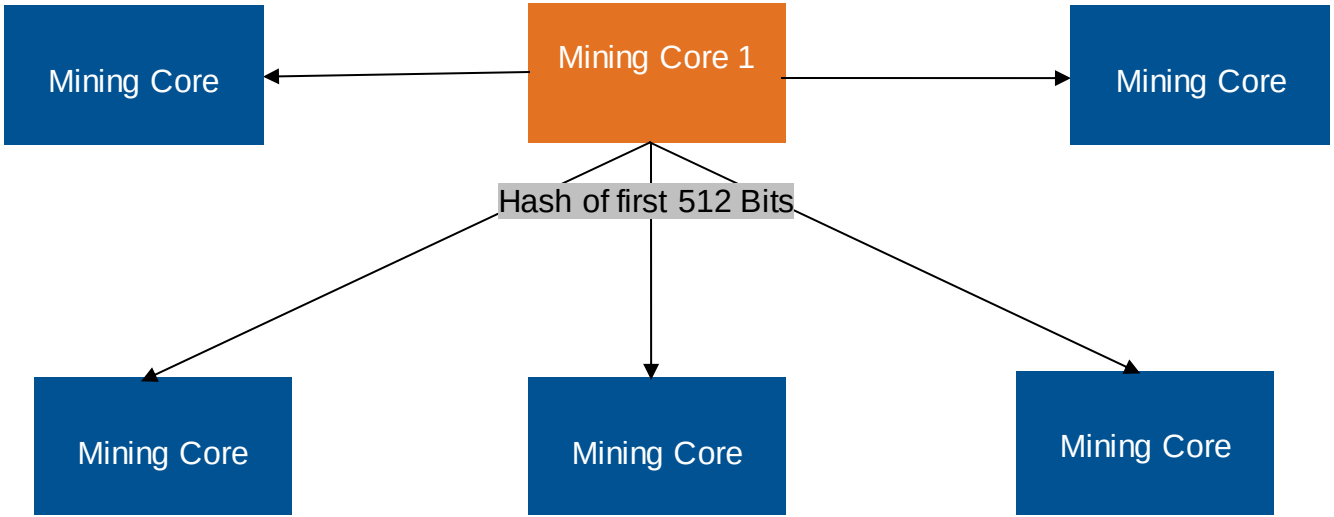

Block header
Padder
Extender
Compressor
Buffer
Hash of block header
Padder
Extender
Compressor
Double Hash
of block header
2X
1X

Length Extension Attack

Length Extension Attack



Length Extension Attack



Flächenoptimierung

- Ziel: Platzverbrauch eines Mining Cores verringern, um insgesamt mehr Cores deployen zu können

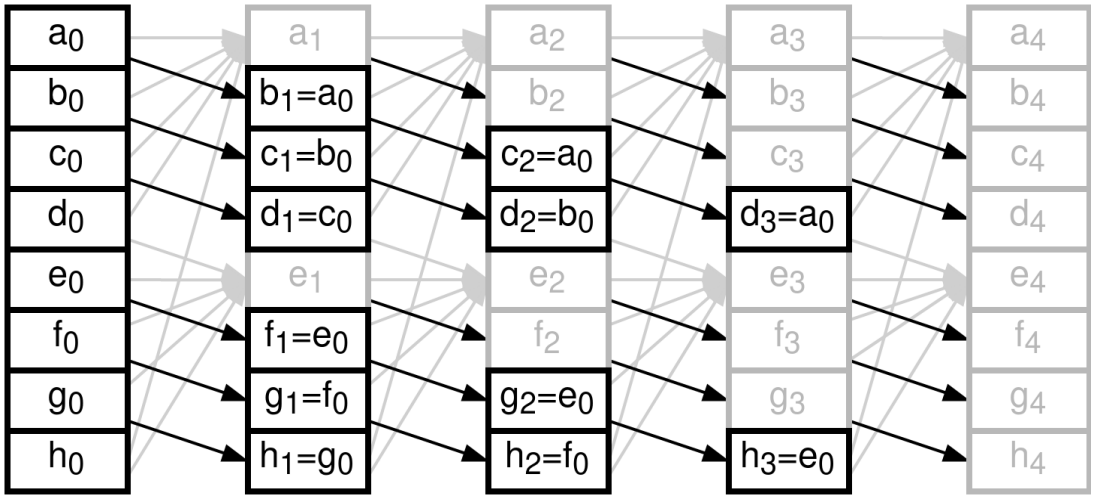
- Beispiel Extender:
  - Nur 16 statt 64 Einträge des Message Schedules speichern
  - Verwendung des Distributed RAMs

$$W_t = \begin{cases} M_t^{(i)} & 0 \leq t \leq 15 \\ \sigma_1\left(W_{t-2}\right) + W_{t-7} + \sigma_0\left(W_{t-15}\right) + W_{t-16} & 16 \leq t \leq 63 \end{cases}$$

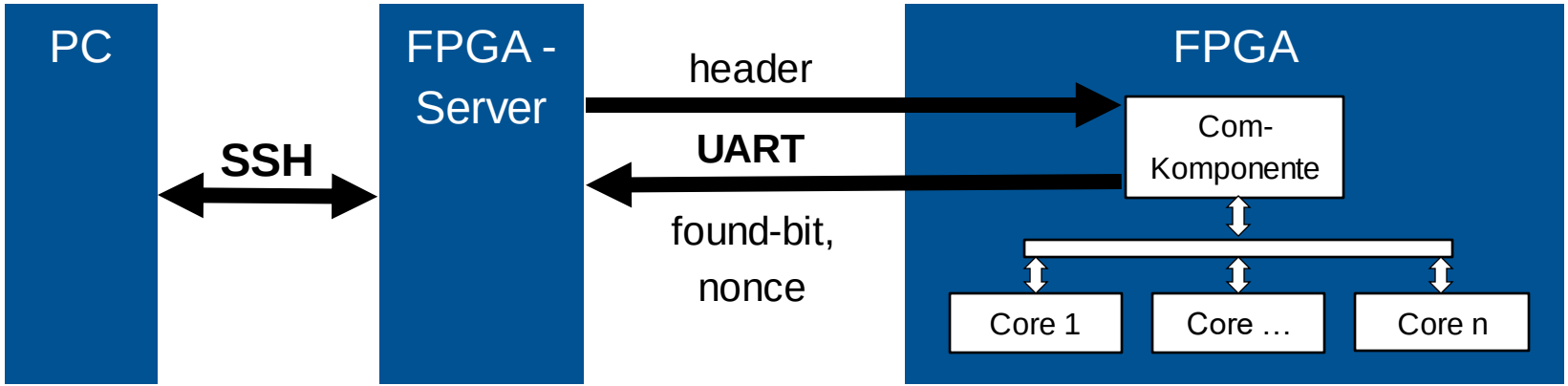
Padded Chunk: $M_0, M_1, \ldots, M_{15}$
Message Schedule: $W_0, W_1, \ldots, W_{63}$

Timing

- Ziel: Mehr Cycles pro Sekunde und/oder mehr Arbeit pro Cycle
- Clockfrequenz erhöhen, aber beschränkt durch Länge des kritischen Pfades
- 4 bzw. 2 Runden per Cycle
- Komplett ausgerollter Compressor (Runde = Pipeline-stage)



Kommunikation mit dem FPGA



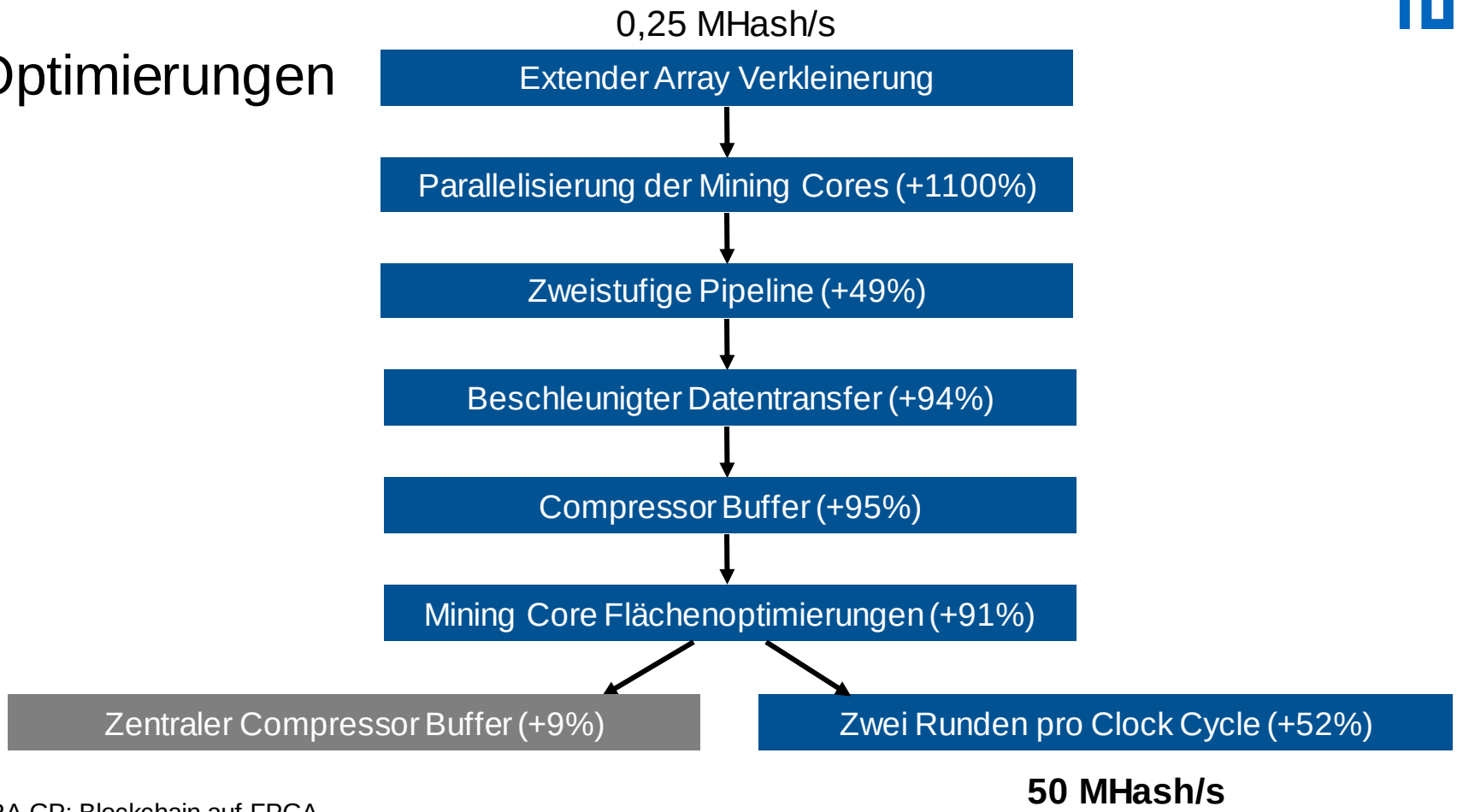
Tests

- **GHDL-Tests**
  - Syntaxfehler
  - Theoretische Funktionalität
    - Testbenches
    - Simulation

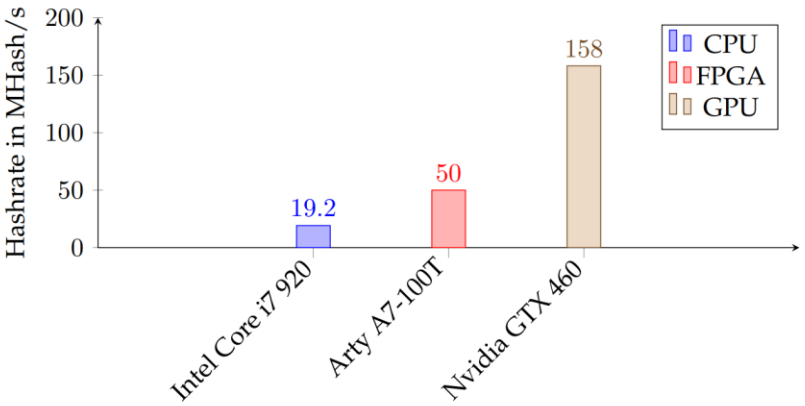
- **Build**
  - Ressourcenausnutzung (Belegte Logikzellen, RAM, Strom, …)
  - Timing

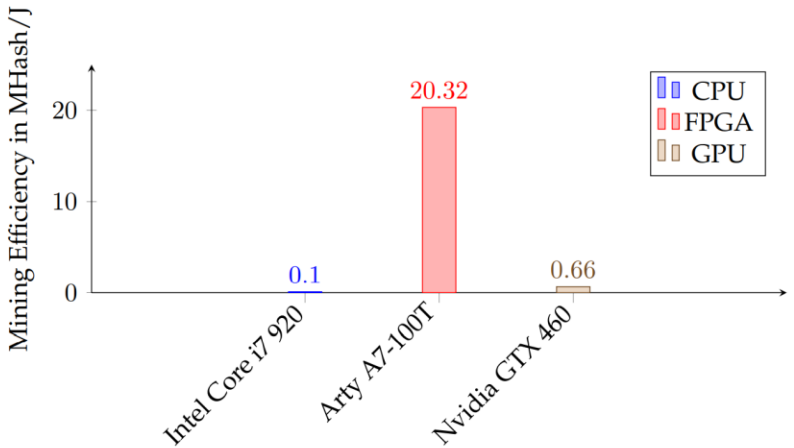
- **Deployment**
  - Prüfung aller Cores → Verbindung + Korrektheit Berechnung
  - Performance
  - (Ausreichende Stromversorgung)

Optimierungen

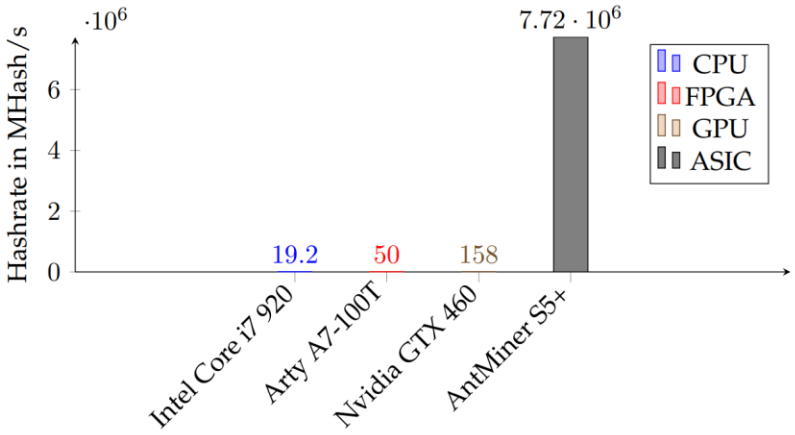


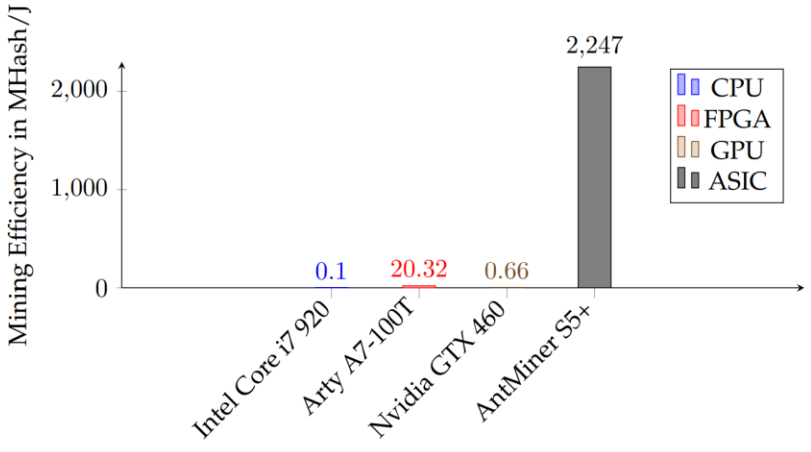
Vergleich mit anderen Implementierungen





Vergleich mit anderen Implementierungen





DEMO-TIME

```
python3 connector/cli.py mine
0000c0209178710b365783606147a9e53716e8a3cdb001356f2b1000000000000000004d2629573f7682
008d55c4c53250b35832658d6963fc24514224b01d6b117c100668ac5e33a31117 -f
```

Quellen

Nakamoto, Satoshi (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Retrieved July 17, 2020 from https://bitcoin.org/bitcoin.pdf

Haber, S., Stornetta, W.S. (1991). How to time-stamp a digital document. *J. Cryptology* 3**,** 99–111. doi:10.1007/BF00196791

NIST (2015). Secure Hash Standard (SHS). FIPS PUB 180-4. doi: 10.6028/NIST.FIPS.180-4