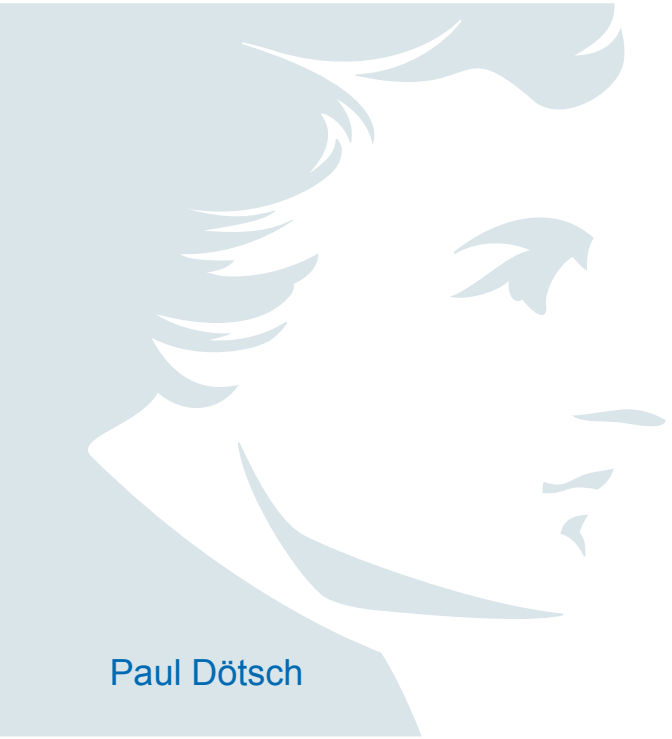
hhu

hhu
Heinrich Heine
Universität Düsseldorf



Themen Vorschlag: MCHPRS

Themen Vorstellung zur Bachelor Thesis über Gate-Level Logic-Optimization am Beispiel von Minecraft-High-Performance-Redstone-Server

Paul Dötsch

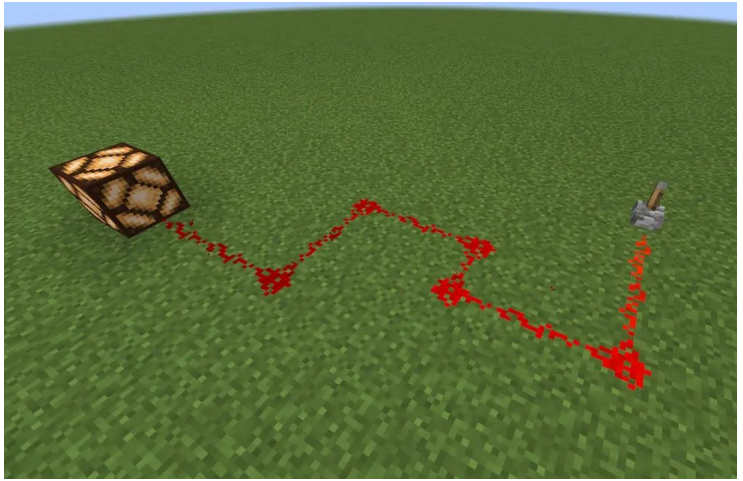
Einführung & Motivation

- Minecraft: Ein digitales Phänomen
  - Aktive Entwicklung für 15+ Jahre
  - Über 300 M aktive Spieler
- Redstone: Der "Elektronik-Baukasten"
  - Spielerischer Zugang zur digitalen Logik
  - Von einfachen Schaltungen bis zu kompletten Computern
- Praxis Relevanz
  - Optimierung komplexer logischer Systeme
  - Übertragung auf reale Computerarchitektur

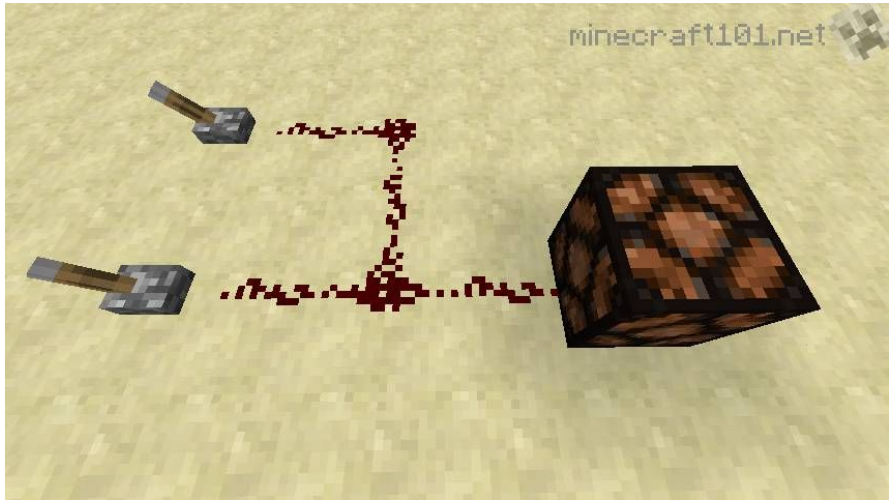
Minecraft Redstone - Grundlagen

Heinrich Heine
Universität Düsseldorf

Redstone Komponente	**Digitale Logik**
Redstone-Dust	Stromleiter / Kabel (OR-Gate)
Redstone-Repeater	Diode
Redstone-Torch	NOT-Gate (NOR-Gate)
Diverse I/O: Levers, Redstone-Lamps, etc.	Schalter, LEDs, etc.

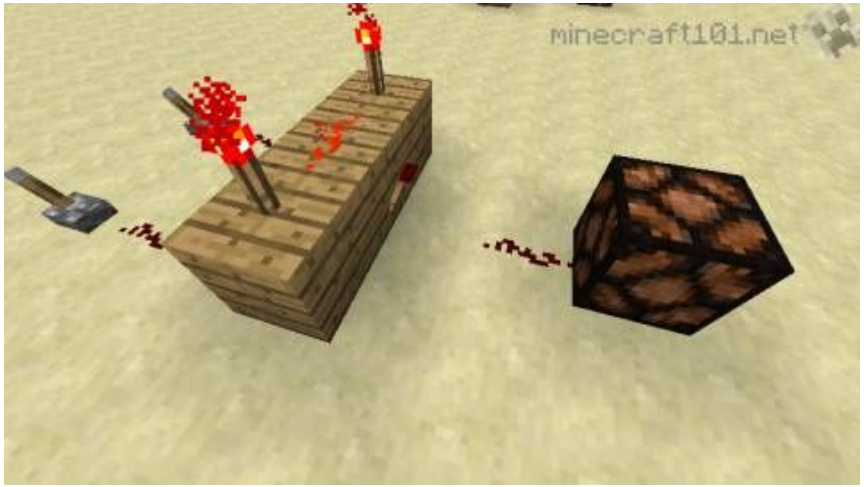


Minecraft Redstone - Beispiele

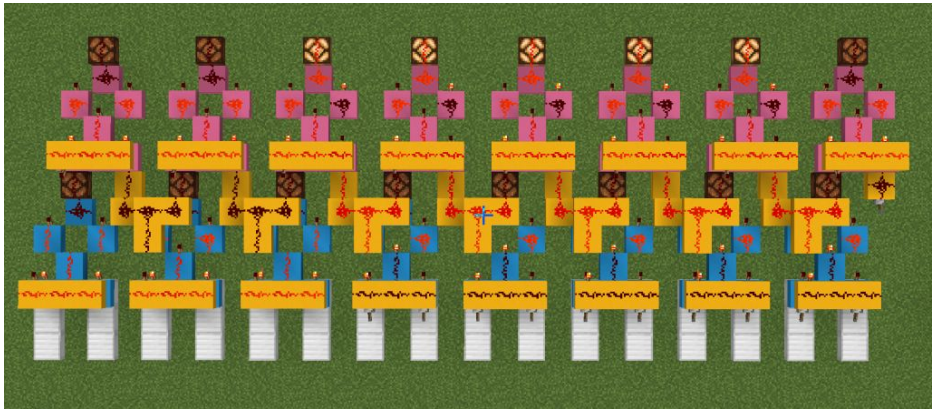


OR-Gate

AND-Gate



Minecraft Redstone - Beispiele



Binary Adder

8-bit CPU

Problemstellung und Lösungsansatz

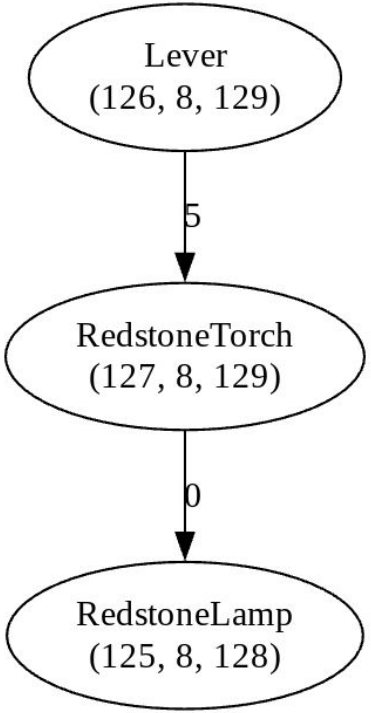
Heinrich Heine
Universität Düsseldorf

- Problem:
  - Minecraft nicht für Simulation von ganzen Computern ausgelegt
  - Ineffiziente Signalausbreitung
- Lösung:
  - Re-Implementierung eines Minecraft-Servers, ausschließlich für diesen Zweck
  - Keine direkte Simulation des Komponenten-Graphens
  - Zusätzlicher Kompilierungs- und Optimierungsschritt

Lösungsansatz - Graph-Transformation

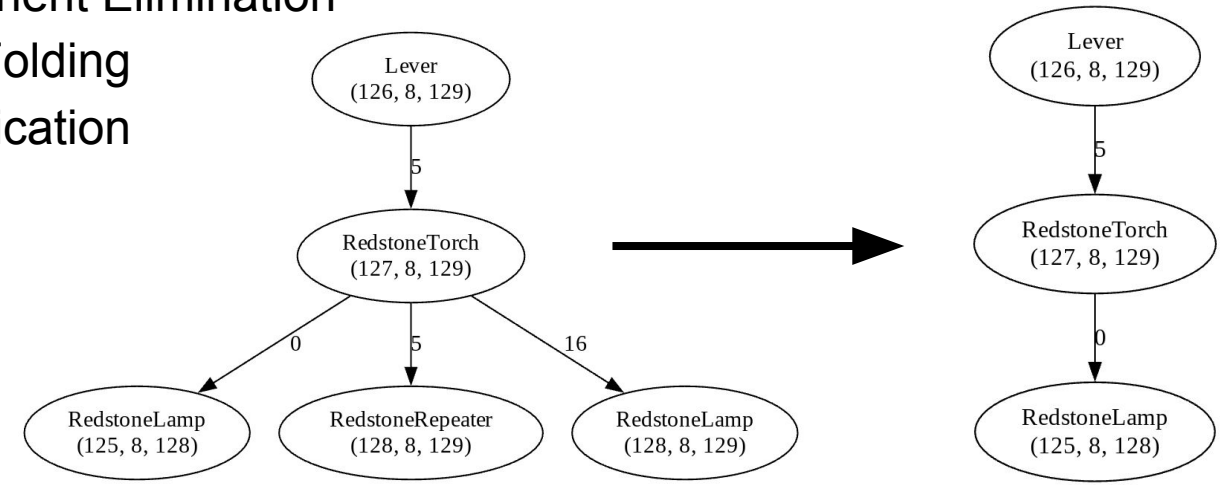
Heinrich Heine
Universität Düsseldorf

Redstone Komponente	**Graph-Struktur**
Redstone-Dust	Kanten (Param.: Länge) *Theoretisch auch Knoten**
Redstone-Repeater	Knoten (Typ: Diode)
Redstone-Torch	Knoten (Typ: Torch)
Diverse I/O: Levers, Redstone-Lamps, etc.	Diverse weitere Knoten



Lösungsansatz - Optimierung

- Compiler-inspirierte Optimierungen
- Ähnlich zum Ansatz von LLVM
- Beispiele:
  - Dead Component Elimination
  - Component Folding
  - Logic Deduplication



Aktueller Stand & Ergebnisse

Heinrich Heine
Universität Düsseldorf

- Bessere Performance, gemessen an einer Beispielhaften Redstone CPU ➜
- Wissenschaftlicher Beitrag:
  - Logik-Optimierung / Simulation unter komplizierten Bedingungen
  - Übertragbarkeit auf RTL optimierungen
  - Sandbox zum testen von neuen Optimierungs-Strategien

Implementierung	**Simulations-Taktrate**
Vanilla / Basisspiel	≤ 20 Hz
Alternative Current Mod + Carpet Mod (State-of-the-Art)	~ 200 - 1000 Hz
MCHPRS (unser)	~ 100 kHz - 2 MHz

Vertiefender Forschungsfokus

Heinrich Heine
Universität Düsseldorf

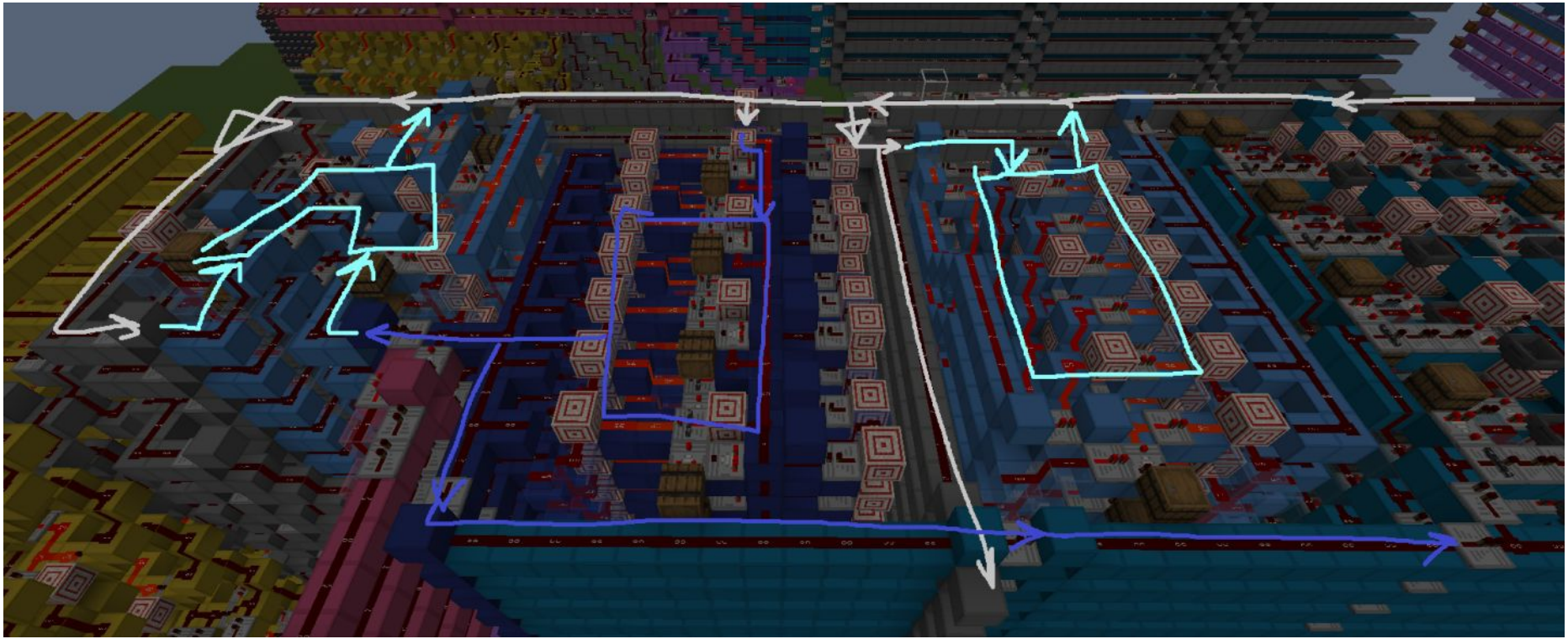
Alle vorherigen Punkte sind bereits erfolgreich implementiert worden.
Folgende Forschungsfragen & -probleme sind als Vertiefung vorstellbar:

1. Charakterisierung der verschiedenen, von uns Implementierten, Optimierungs-Schritte (Qualitative and Quantitative Analyse)
2. Weitergehende Kompilierung zu einer HDL zur Simulation auf einem FPGA (Implementierungsarbeit?)
3. Multithreading der Simulation des Optimierten Graphens (Efficient/Heuristic Graph-Partitioning)
4. Automatische Erkundung von neuen Optimierungs-Regeln mit Hilfe eines Verifizierer (Dynamic Programming?)

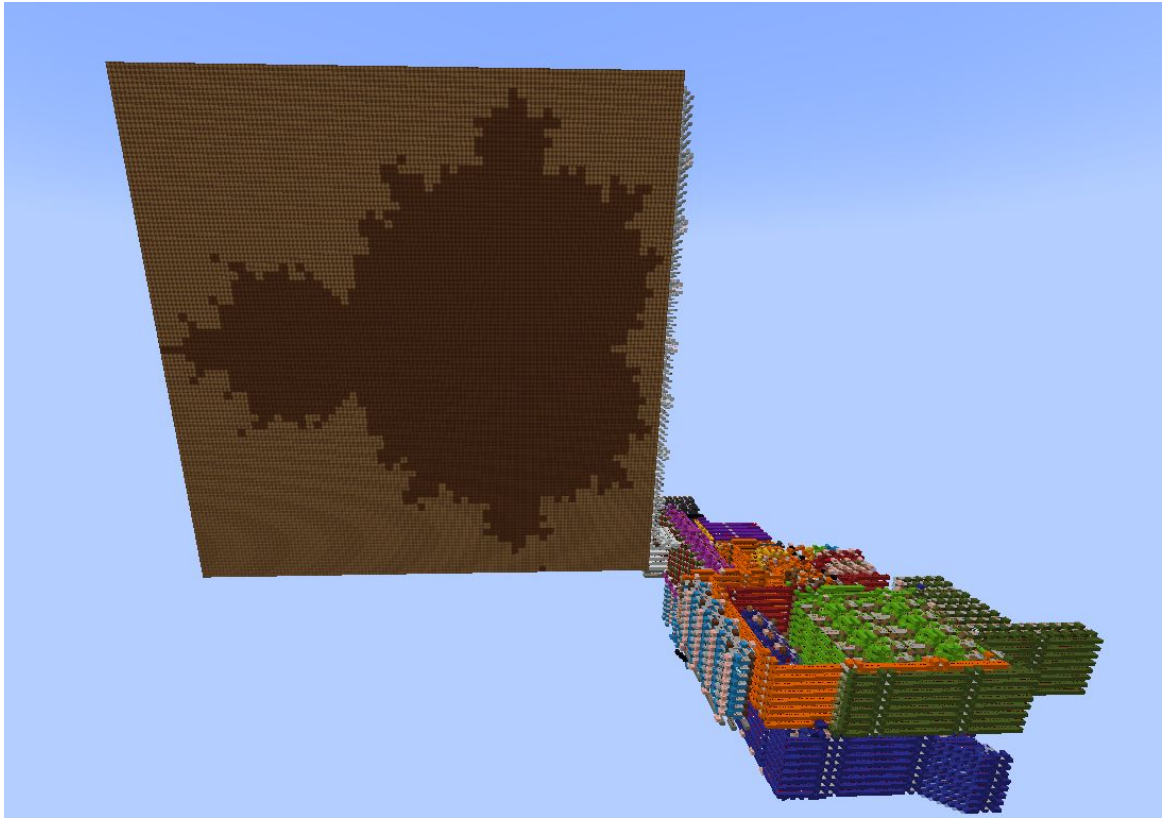
Anhang A

- Tetris in Minecraft:
  https://www.youtube.com/watch?v=USH-PME_rls
- Minecraft in Minecraft:
  https://www.youtube.com/watch?v=-BP7DhHTU-I
- Colored Minecraft in Minecraft:
  https://www.youtube.com/watch?v=qvm6N4zj1OM
- Personal MCHPRS Fork:
  https://github.com/Paul1365972/MCHPRS/
- Main MCHPRS Repository:
  https://github.com/MCHPR/MCHPRS/

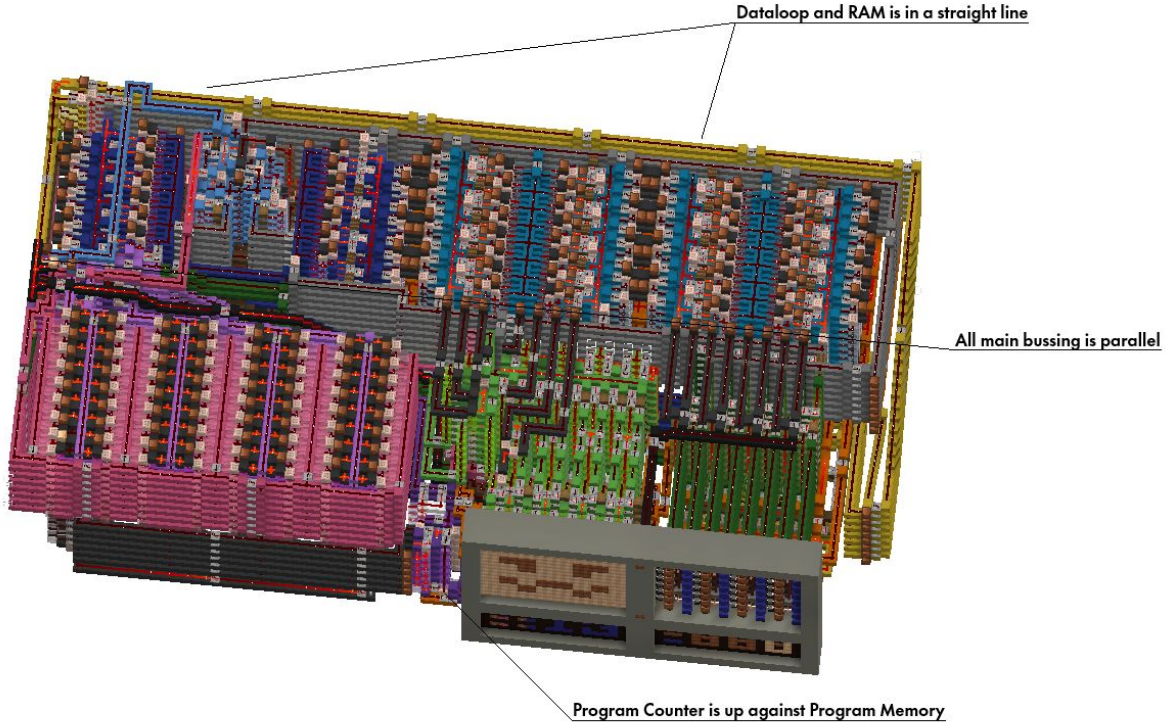
Anhang B



Anhang C



Anhang D


Dataloop and RAM is in a straight line
All main bussing is parallel
Program Counter is up against Program Memory