HiCoSim

- Simulator für einen konfigurierbaren Vektorprozessor -

Übersicht:

(1) Aufgabenstellung
(2) Einführung HiCoVec
(3) Anforderungen
(4) Realisierung HiCoSim
(5) Demonstration
(6) Zusammenfassung und Ausblick

Aufgabe & Ziel

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

Aufgabe:

→ Erstellung eines erweiterbaren Simulators für einen konfigurierbaren Vektorprozessor (HiCoVec)

Ziel:

→ Ausführung und Leistungsanalyse von beliebigen HiCoVec-Programmen
→ Softwareentwurf auch ohne Hardware
→ Simulation von beliebigen HiCoVec-Konfigurationen

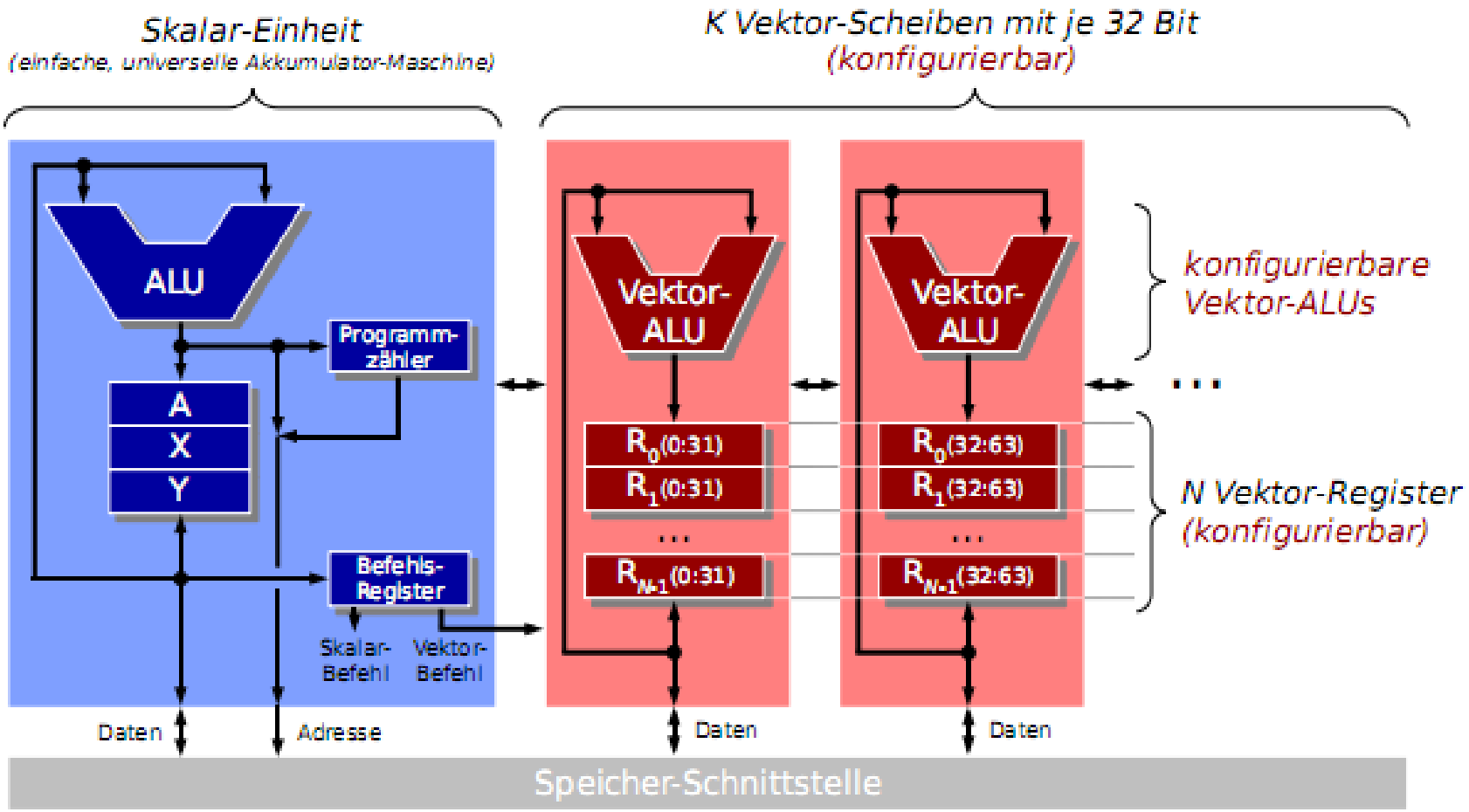
Der HiCoVec

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

→ ***Hi**ghly **Co**nfigurable **Vec**tor Processor*
→ Open Source
→ An HS Augsburg entwickelt (Prof. Dr. Kiefer)
→ Beschreibung in Diplomarbeit (H. Manske)
→ Realisierung auf FPGA-Board
→ Entwicklungsumgebung inkl. Assembler
→ Ausblick:
- C-Compiler (in Entwicklung)
- Erweiterungen, z.B. Gleitkommaeinheit

Aufbau des HiCoVec

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

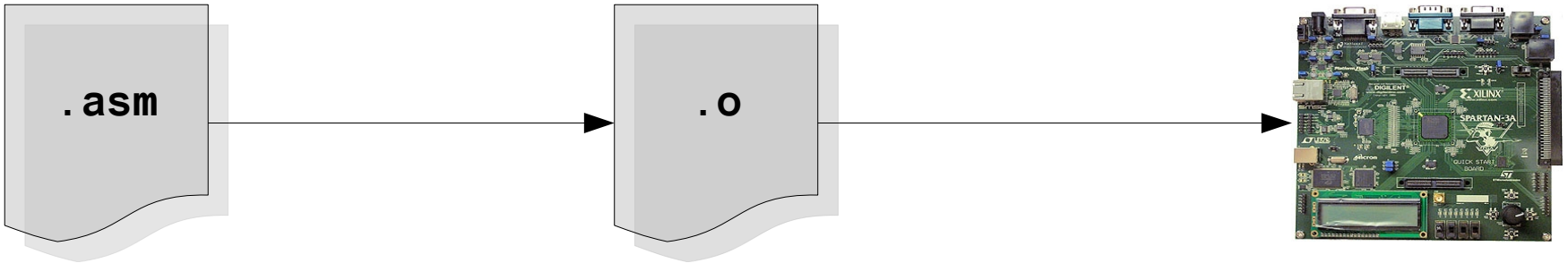


Quelle: HiCoVec-Flyer für Embedded World

Bisheriger Softwareentwurf

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

→ Assemblerprogramm schreiben
→ Assemblieren mit scotchAS
→ Objektdatei auf FPGA laden
→ On-Chip-Debugging möglich (aber eingeschränkt)
→ Kein Testen ohne Hardware möglich



Anforderungen

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

→ Simulation beliebiger HiCoVec-Programme
→ Unterstützung beliebiger Konfigurationen
→ Protokollierung des Programms (Statistiken)
→ Erweiterbarkeit des Befehlssatzes
→ Erweiterbarkeit der Peripherie
→ Grafische Oberfläche unabhängig von Simulator
→ Benutzerfreundlich und Portierbar

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

Komfortabler Software-Entwurf durch:

→ Breakpoints setzen
→ Watchpoints setzen
→ Disassembler
→ Simulation unterbrechen & Fortsetzen
→ Single Stepping
→ Speicher zur Laufzeit manipulierbar
→ Programmanalyse durch Statistiken
→ Anzeige des aktuellen Zustands (z.B. Register)
→ Änderbare Konfiguration des HiCoVec

Übersicht Realisierung

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

→ Aufteilung in Backend / Frontend
- → *Backend*:
  - Simulation des HiCoVec
  - Einfache Konsoleneingabe
- → *Frontend*:
  - Benutzereingaben und Anzeige

→ Verwendung von Java als Programmiersprache
- → Java 6
- → GUI: Swing
- → keine externe Bibliotheken notwendig

Aufgaben Frontend

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

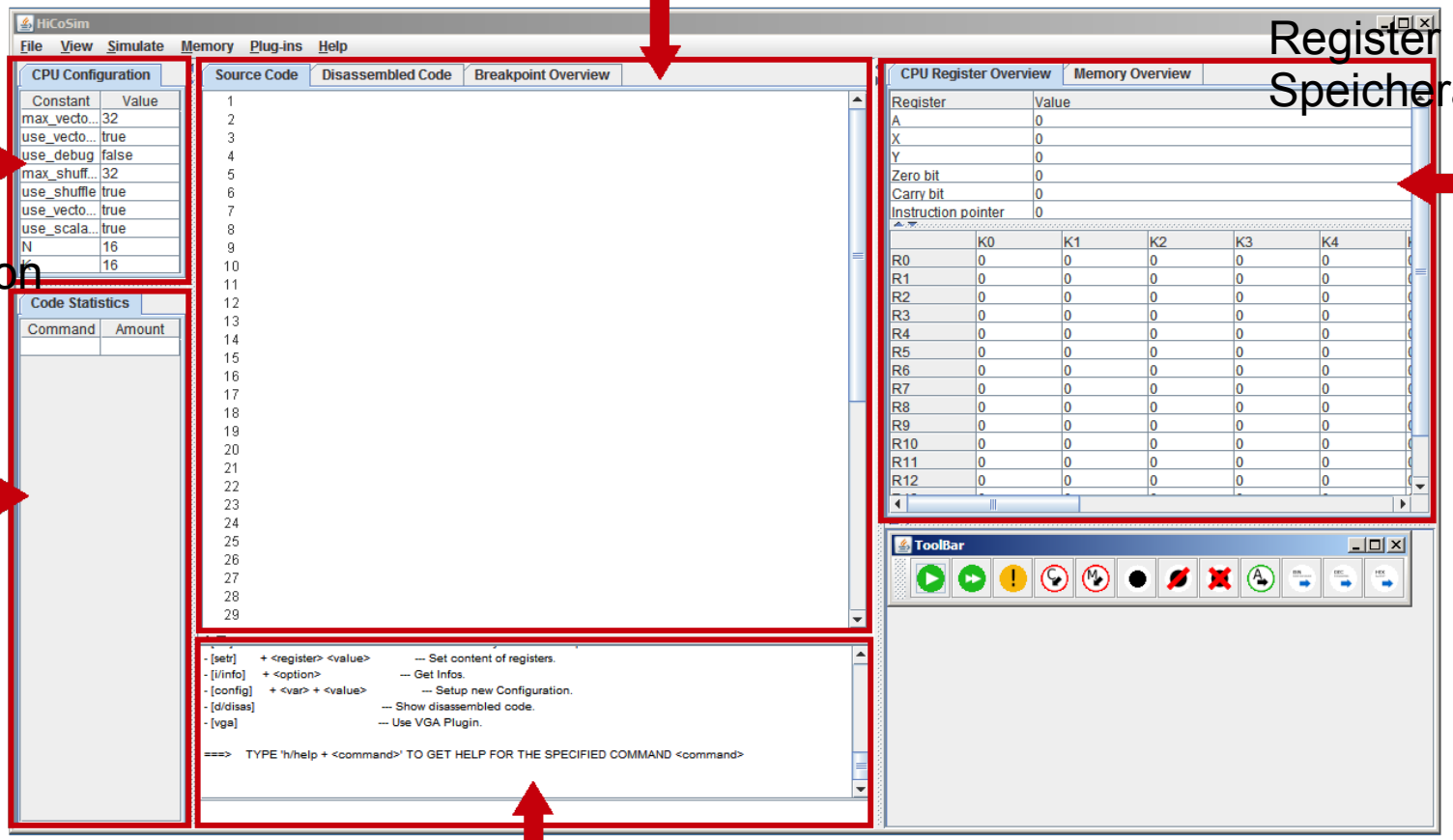
→ Benutzerfreundliche und konfigurierbare Oberfläche
→ Aufrufe der Backend-Methoden
→ Grafische Darstellung für:

- Code (Quellcode/Disassembliert)
- Register & Speicher
- Statistiken
- CPU-Konfiguration
- Konsole

Grafische Benutzeroberfläche

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

Quellcode



Register & Speicheranzeige

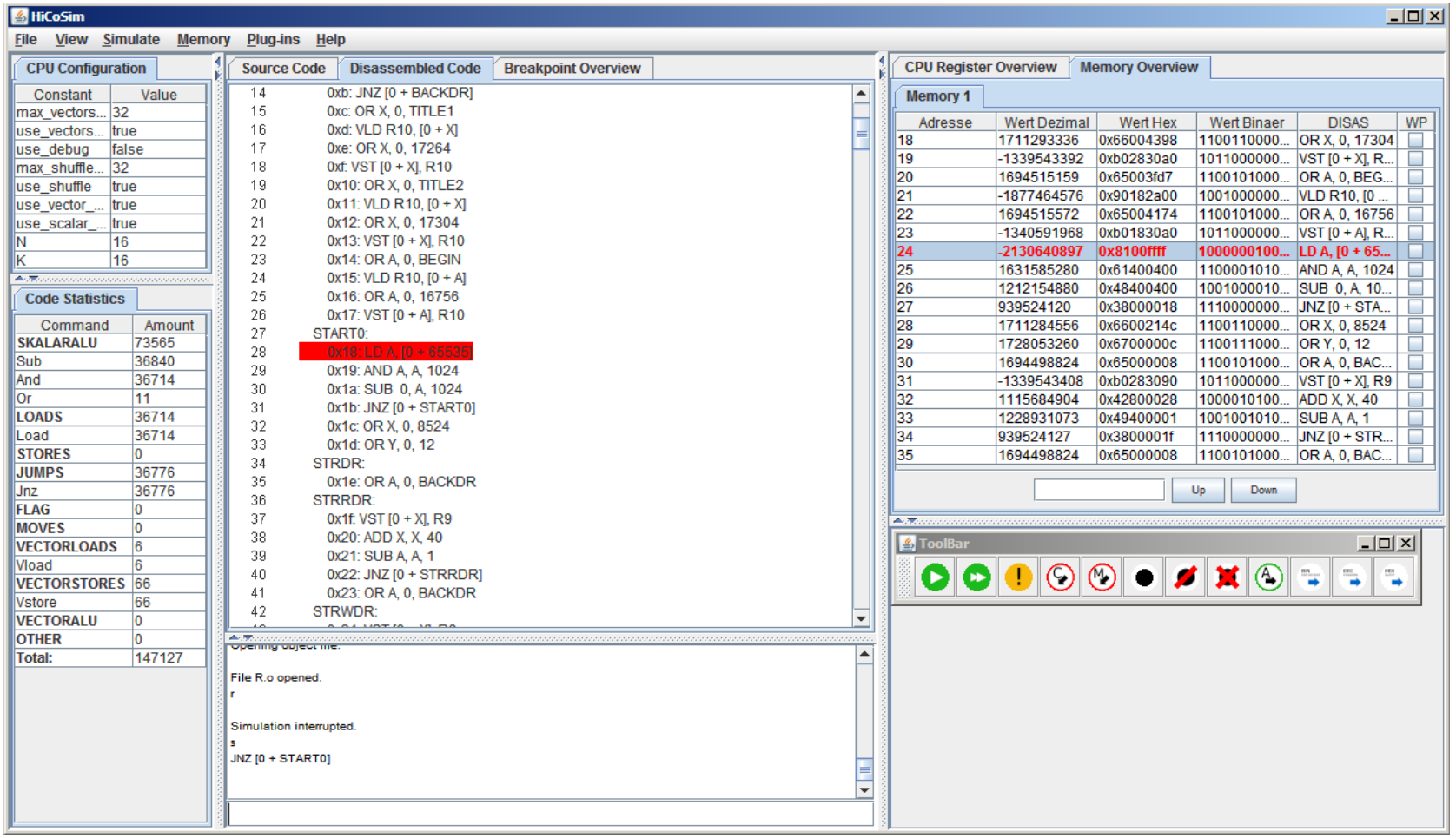
CPU-Konfiguration

Statistiken

Konsolen Ein-/ Ausgabe

Programmablauf GUI

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -



Aufgaben Backend

→ unabhängig von GUI ausführbar durch Callbacks
→ Simulation des HiCoVec
Verwaltung von CPU und Speicher
→ Ablaufsteuerung (starten, unterbrechen)
→ Einfache Konsolen Ein- und Ausgabe für scriptgesteuerte Aufrufe

Befehle auf Konsolenebene

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

> open
Öffnet Objektdatei und initialisiert Speicher

> list
Zeigt Sourcecode an

> run
Startet Simulation

> i
Unterbricht Programmausführung

> step
Führt einen Step durch

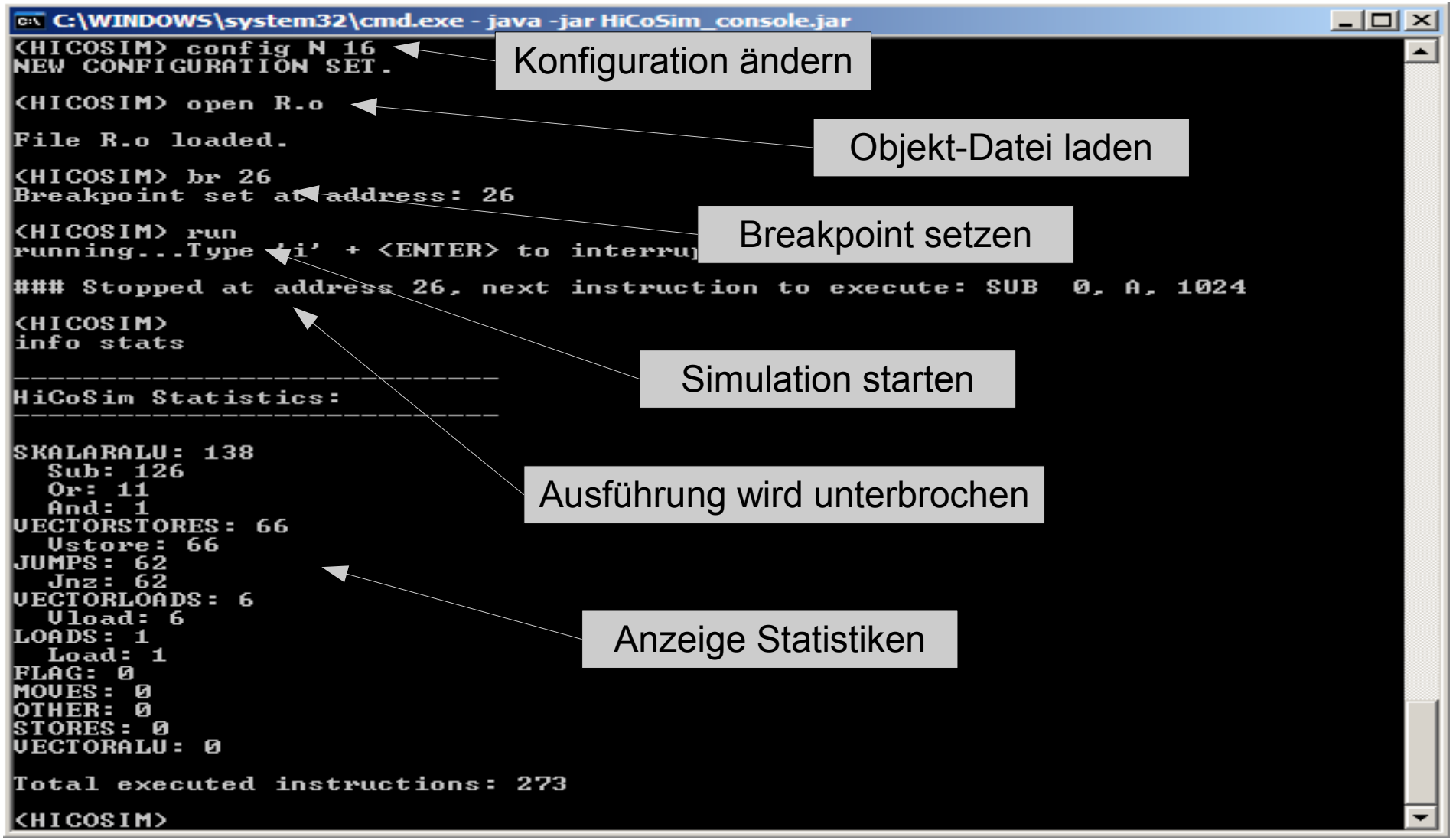
> br
Setzt breakpoint. `Beispiel: br 0x15`

> set / setr
`Setzt Speicher/Register. Beispiel: setr A 0x3`

> get
Gibt Wert an Speicherstelle zurück. `Beispiel: get 0x74`

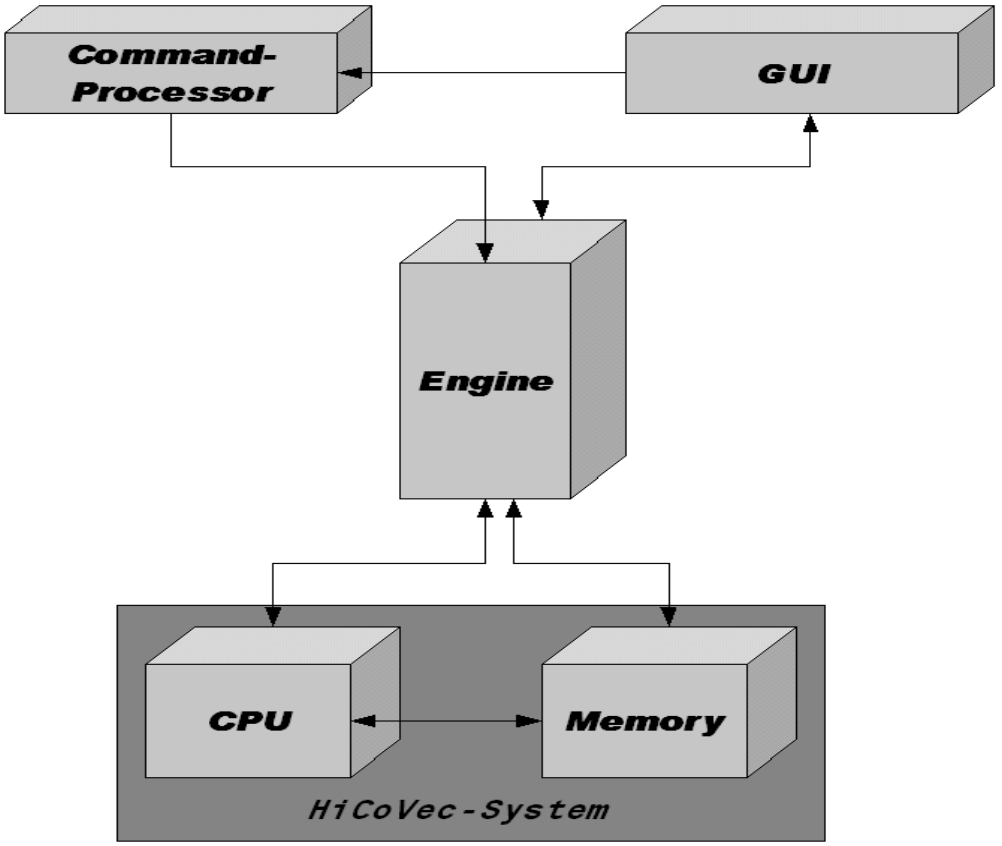
Programmablauf Konsole

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -



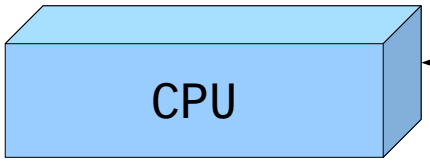
Struktur des HiCoSim

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

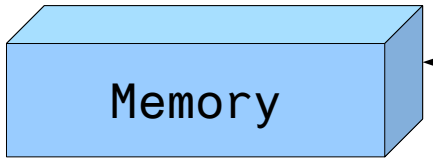


Die wichtigsten Module

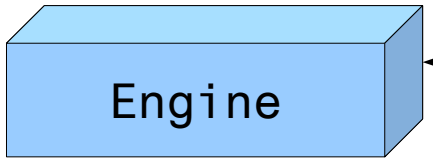
HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -



→ Abbildung des HiCoVec-Prozessors inkl. Register
→ Decodierung und Ausführung der Maschinenbefehle
→ Führt Lese-/Schreiboperationen auf Memory durch
→ Lädt Prozessor-Konfiguration



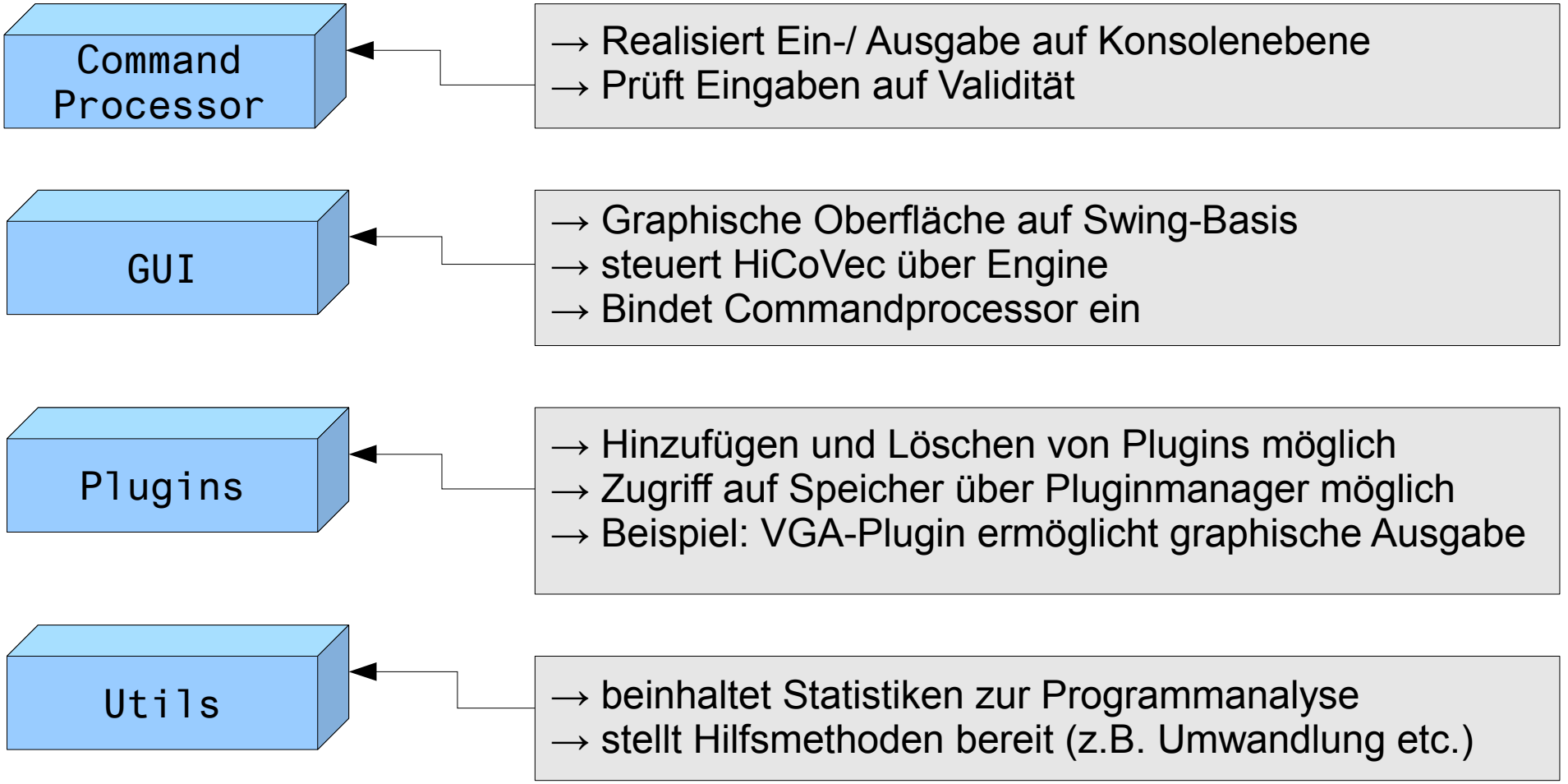
→ Organisiert Speicher in Form einer SortedMap
→ stellt Methoden zum Laden der Objektdatei bereit



→ Schnittstelle zwischen HiCoVec und GUI/Konsole
→ Methoden zur Ablaufsteuerung des System (run, step)
→ Benachrichtigt GUI bei Änderungen
→ Kommunikation mit GUI über Listener

Die wichtigsten Module

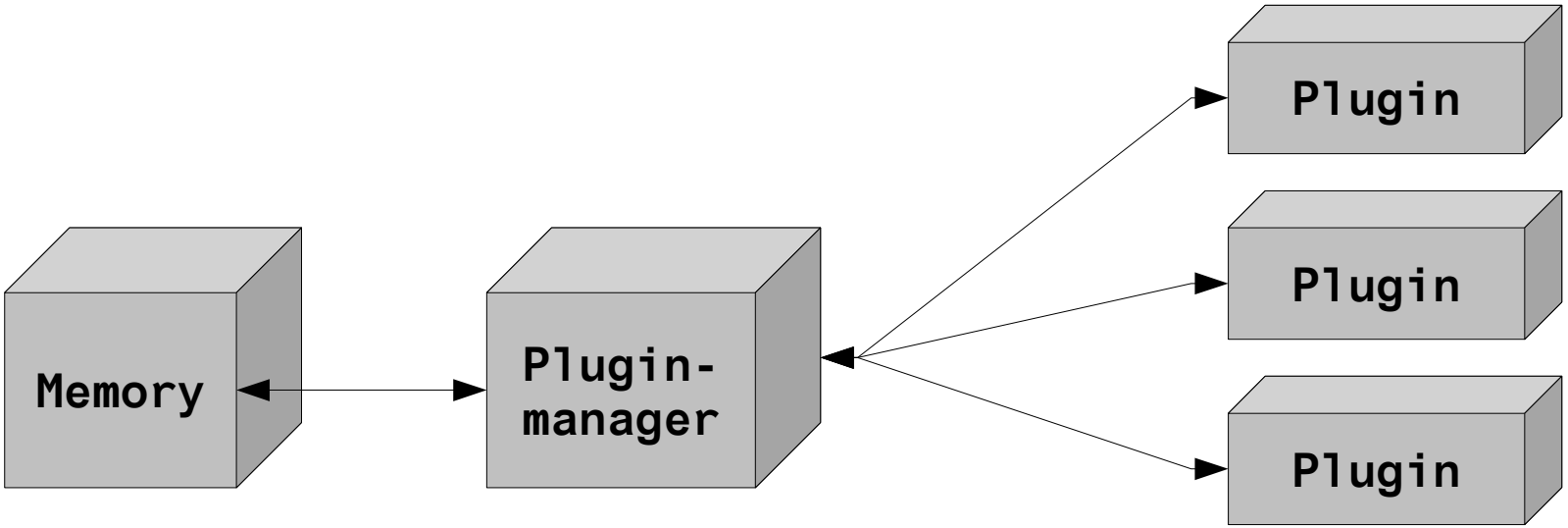
HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -



Erweiterbarkeit durch Plugins

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

→ PluginManager verwaltet Plugins zentral
→ Plugins müssen Interface implementieren
→ Observer-Modus möglich

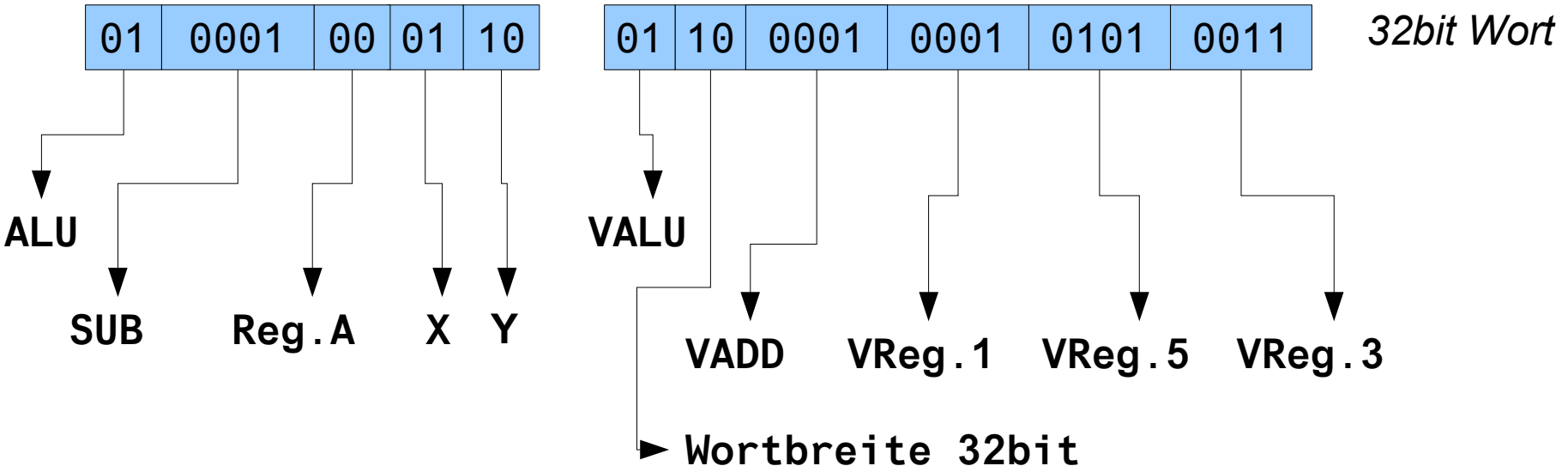


HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

Dekodieren eine Maschinen-befehls

```
....
.org 0x00
.start
ADD A, X, 0 + 5
SUB A, X, Y # VADD.DW R1, R5, R3
....
HALT
.end
```

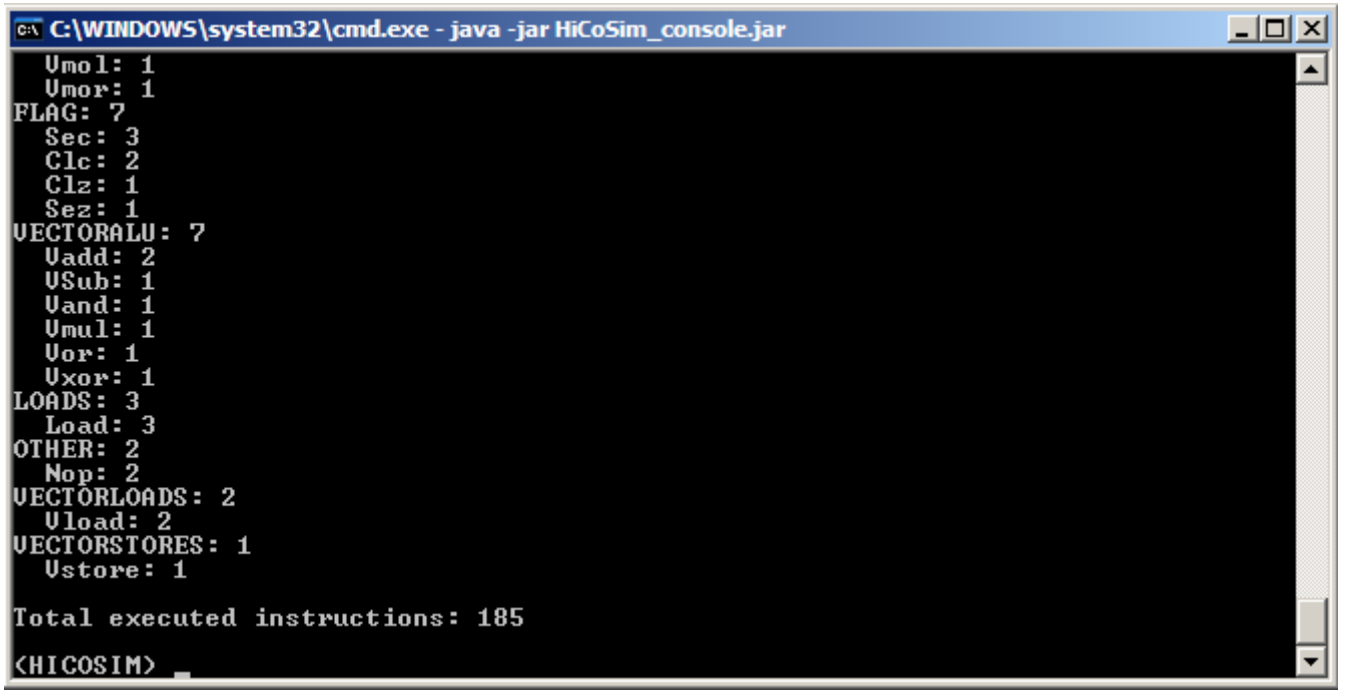
→ Wortlänge Prozessor: 32bit
→ Besonderheit: parallele Befehle möglich
→ Bei Vektorbefehlen: Wortbreite möglich
→ Aufteilung:
- 12bit für Skalarbefehle
- 20bit für Vektorbefehle



Demonstration Konsole

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

Ablauf Testdatei all.o – Ausführung aller im HiCoSim implementierter Befehle



Demonstration GUI

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

→ Aufgabe: Addition zweier Arrays, je 16 Wörter

vek1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
									+							
vek2	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
									=							
ERG	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17

Skalareinheit:

```
      OR Y, 0, 0
LOOP: LD A, [Y + vek1]
      LD X, [Y + vek2]
      ADD A, A, X
      ST [Y + erg], A
      INC Y, Y
      SUB 0, Y, 16
      JNZ [0 + LOOP]
```

C-Code:

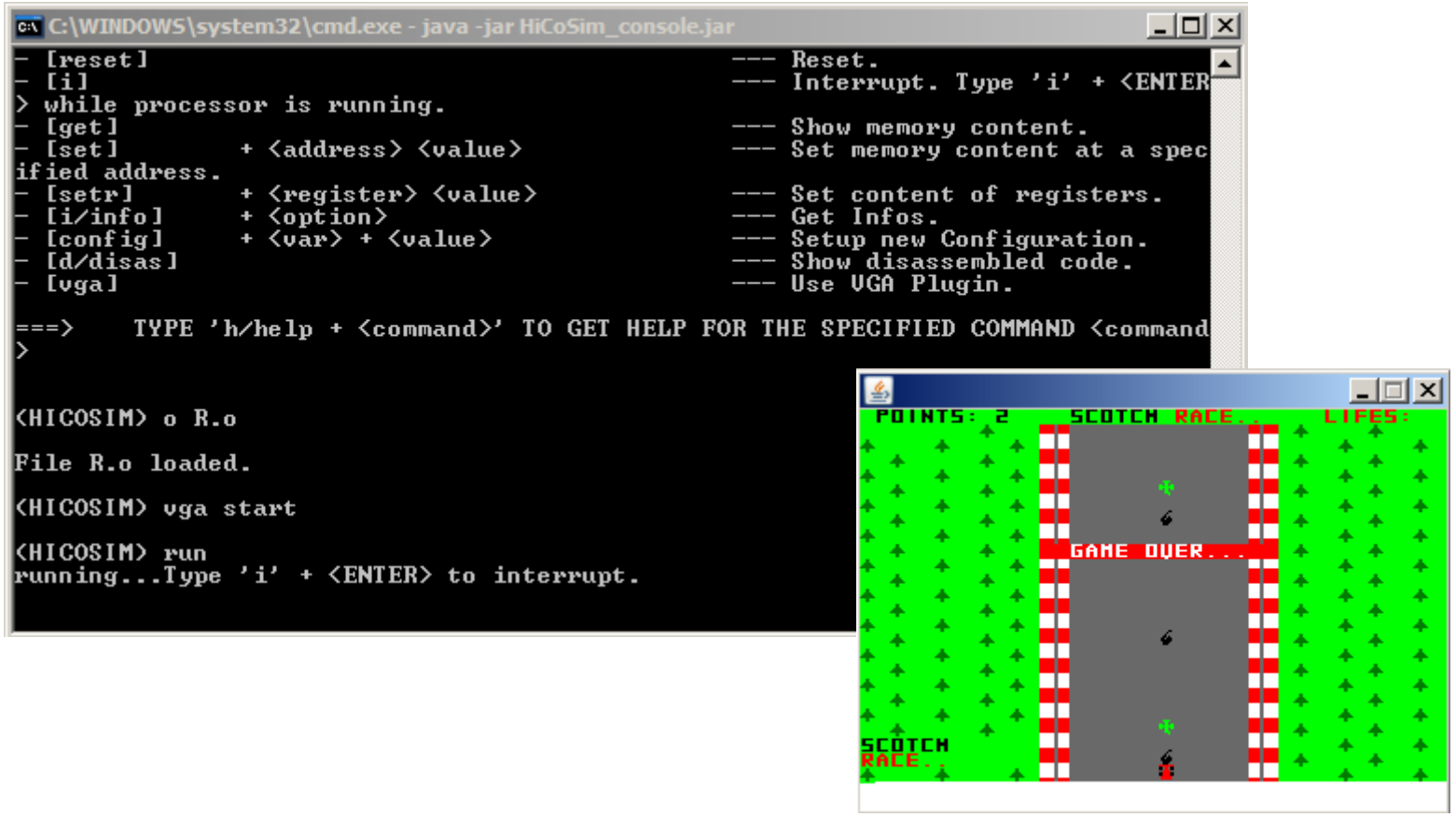
```
for(i=0;i<16;i++){
 erg[i] = vek1[i] + vek2[i];
}
```

Vektoreinheit:

```
OR A, 0, vek1
OR X, 0, vek2
OR Y, 0, erg
VLD R0, [0 + A]
VLD R1, [0 + X]
VADD.W R2,R0, R1
VST [0 + Y], R2
```

Demonstration Scotchrace

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -



Team

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

Backend

Müge Özugur

Michael Wager

Andreas Weber

Luca Calchera

Hibetoullah Ftima

Frontend

Sebastian Minning

Daniel Obermüller

Projektleiter

Sebastian Minning

Betreuer

Prof. Dr. Kiefer

Projektstand

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

Wir freuen uns auf ihren Besuch an unserem Stand im Raum J2.01

HiCoSim
- Simulator für einen konfigurierbaren Vektorprozessor -

Zusammenfassung

Umfang:

Codezeilen: 16.300
54 Klassen
468 Methoden

Fazit:

→ Software-Entwurf auch ohne Hardware möglich
→ Unterstützung für Compileroptimierungen
→ Einfacher Test verschiedener CPU-Konfigurationen