

# Weiterentwicklung Prozessor-Simulator

## Studenten



Tobias Petter



Michael Schneider

**Einleitung:** Informatikstudenten lernen in den ersten Semestern ihres Studiums im Doppelmodul "Betriebssysteme" unter anderem, wie moderne Prozessoren funktionieren. Dies erfordert ein hohes Mass an abstraktem Denkvermögen und ist deshalb für Studiums-Neulinge oft anspruchsvoll. Um Studenten dabei zu helfen, wurde in der Vergangenheit die Entwicklung eines grafischen Prozessor-Simulators in Auftrag gegeben. Unsere Arbeit hatte zum Ziel, diesen weiterzuentwickeln und um zusätzliche Funktionen zu ergänzen. Wir fokussierten uns dabei hauptsächlich auf mehrere Verbesserungen in der Bedienbarkeit, sodass der Simulator zum Verständnis der Vorlesung und zur Prüfungsvorbereitung eingesetzt werden kann.

**Vorgehen:** Unser erster Schritt war ein Erheben der Bedürfnisse der Studenten. Dazu besuchten wir eine Übungsstunde in "Betriebssysteme 1" und befragten die dort anwesenden Studenten nach ihrer Meinung über den Simulator. Die dort geäußerten Vorschläge setzten wir dann gemeinsam mit unseren eigenen Ideen um.

Darunter fielen unter anderem:

- 1) die Möglichkeit, im Simulator "zurückzuspulen" und beliebig viele Schritte in der Simulation rückwärts zu gehen
- 2) der Einbau eines Sliders zum Regeln der Animationsgeschwindigkeit
- 3) zwei neue Farbschemen (Light & Dark Mode) und eine Möglichkeit, zwischen ihnen zu wechseln
- 4) eine neue, verständlichere Animation für Instruktionen, welche den Instruction Pointer modifizieren (z.B. Jumps, Call, Return)

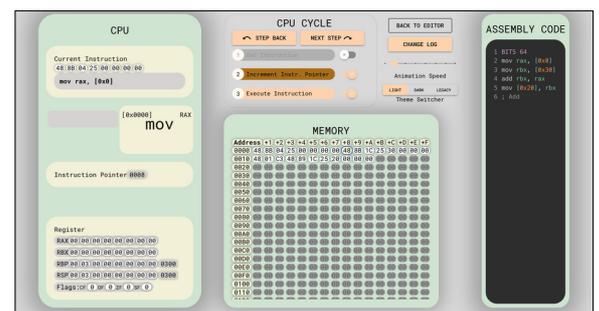
**Ergebnis:** Der Hauptteil der Arbeit bestand aus der Einbindung des sogenannten „Reverse Debugging“, das es Nutzenden ermöglicht, vergangene Simulationsschritte erneut durchzuführen. Dies kommt dem Lerneffekt zugute, da man sich die Ausführung nicht gänzlich klarer Schritte leicht nochmals ansehen kann, ohne alles von vorne starten zu müssen. Wir haben uns dabei für eine gemischte Lösung entschieden: die Felder des GUI werden mittels einer Transaktionsanalyse in jedem Schritt auf Änderungen überprüft (Javascript Proxies) und diese aufgezeichnet. Beim Schritt zurück werden die alten Werte aus dem Speicher wieder eingefügt. Der dahinterliegende Assembly-Emulator hingegen wird bei Bedarf neu aufgebaut und läuft von Anfang des Programms bis zur gewünschten Stelle durch. Es ist uns gelungen, den Reverse Debugger ohne signifikante Performance- und Speicher-Overheads zu implementieren.

Der Simulator ist unter einer OpenSource-Lizenz veröffentlicht und kann unter <http://cpusimulator.pages.gitlab.ost.ch/cpusim-frontend/> verwendet werden.

## Simulator - Code-Editor Eigene Darstellung



## Simulator - Hauptbildschirm Eigene Darstellung



Examinator  
Prof. Stefan Richter

Themengebiet  
Software