



**Übungen zur Vorlesung
“Database Systems and Modern CPU Architecture”
Sommersemester 2009**

Jan Rittinger (jan.rittinger@uni-tuebingen.de)

4. Übungsblatt

Ausgabe: 20. Mai 2009 · Besprechung: 27. Mai 2009

Aufgabe 1: 5-Stufen-Pipeline

Gegeben sei das folgende MIPS-Programm:

```
1  loop: LD      R1,0(R2)      ; Load R1 from address 0+R2
2         DADDI   R1,R1,#1     ; Increment R1 by 1
3         SD      0(R2),R1     ; Store R1 at address 0+R2
4         DADDI   R2,R2,#4     ; R2 := R2 + 4
5         DSUB   R4,R3,R2     ; R4 := R3 - R2
6         BNEZ   R4,loop      ; Branch to loop if R4 != 0
```

Der anfängliche Wert von R3 ist $R2 + 396$.

1. Portieren Sie das Programm für die Verwendung unter SPIM.

Im folgenden nehmen wir eine fünf-stufige Prozessor-Pipeline an, wie sie in der Vorlesung vorgestellt wurde.

2. Nehmen wir an, der verwendete Prozessor implementiert *kein* „Forwarding“ (Folie 25 im Script) und reagiert auf „Branches“ mit einem erneuten „Instruction Fetch“. **Wieviele Taktzyklen benötigt der Prozessor zum Abarbeiten dieser Routine? Zeichnen Sie zur Illustration den Zustand der Pipeline in den einzelnen Schritten der Abarbeitung** (siehe z. B. Folie 10 im Script).

3. **Betrachten Sie das gleiche Problem, jedoch dieses Mal mit „Forwarding“.**

4. Tatsächlich implementieren MIPS-Prozessoren „Delayed Branches“. **Nutzen Sie dieses Verhalten und schreiben Sie den obigen Code so um, dass keine „Stalls“ mehr notwendig sind.**

Hinweis: Zum Testen eignet sich hier selbstverständlich der SPIM-Simulator. Um auch „Delayed Branches“ zu implementieren, muss dieser allerdings mit der Option `-delayed_branches` gestartet werden.