# Universität Karlsruhe (TH)

**Lehrstuhl für Programmierparadigmen**

Sprachtechnologie und Compiler WS 2009/2010 http://pp.info.uni-karlsruhe.de/
Dozent: Prof. Dr.-Ing. G. Snelting snelting@ipd.info.uni-karlsruhe.de
Übungsleiter: Sebastian Buchwald sebastian.buchwald@kit.edu


Übungsblatt 13 | Ausgabe: 28.1.2010 | Besprechung: 3.2.2010

**Aufgabe 1:** *Codeerzeugung*

Für folgende Ausdrücke soll Code erzeugt werden (z.B. für die RISC Maschine aus Übungsblatt 10/11):

- $(a1 * (a2 + (a3 << (a4 + a5))))$
- $((((a1 + a2) * a3) >> a4) + a5)$
- $((((a1 + a2) + a3) * a4) + (a5 * (a6 << (a7 + a8))))$

**1.1** *Ausdrucksbäume*

Verwenden Sie zuerst das Verfahren: *zuerst linken, dann rechten Ausdruck auswerten*, dann das Verfahren *zuerst rechten, dann linken Ausdruck auswerten.*

Wieviele Register benötigen Sie?

**1.2** *Optimale Registerzuteilung für Ausdrucksbäume*

Gibt es ein besseres Verfahren? Sind die Voraussetzungen für dieses Verfahren erfüllt? Wieviele Register benötigen Sie für die Ausdrücke aus der letzten Aufgabe?

**1.3** *Auslagern*

- Was versteht man unter dem Auslagern eines Wertes?
- Nehmen Sie an die Anzahl der Register ist auf 3 beschränkt, spielt die Auswertungsreihenfolge bei den Ausdrücken jetzt noch eine Rolle?

**Aufgabe 2:** *Befehlsauswahl*

Die Befehlsauswahl setzt eine Reihe von Zwischen- oder Quellsprachbefehlen in eine Sequenz von Befehlen der Zielmaschine um. Was sollte man bei einer guten Auswahl beachten? Was könnte in eine Kostenfunktion für Befehlsauswahlen einfließen.

**Aufgabe 3:** *Befehlsauswahl als Termersetzung*

**3.1** *Bäume*

Stellen Sie folgendes Programm als Folge von Ausdrucksbäumen dar, Anweisungen sind dabei Baumwurzeln.

```
int f(int x, int y, int z)
{
    int k  = x*2 / (y + z);
    int k2 = (k * 4) + (y+(z*x - (y+z)));
    int k3 = x*8 - (z*x);

    if (k3 * (k2 - k) < 100) {
        return 42 - z;
    } else {
```

```
        return 10 * k;
    }
}
```

**3.2** *Ausdrucksbäume für Maschinenbefehle*

Wiederholung: Der x86 Speicheradressierungsmodus erlaubt Adressierungen nach folgendem Schema:

$$\text{Offset} + R_b + R_i * s$$

Offset ist eine beliebige 32bit-Konstante, $R_b$ und $R_i$ sind Register und $x \in \{1, 2, 4, 8\}$. $R_b$ und $R_i * s$ sind optional.

- Stellen Sie Ersetzungsregeln für Ausdrucksbäume zum Erzeugen des IA-32 LEA Befehls auf. Der LEA Befehl berechnet eine Speicheradresse und schreibt das Ergebnis in ein Register. Wie viele mögliche Bäume können Sie finden?
- Wie könnte man die Anzahl der benötigten Regeln reduzieren?
- Stellen Sie weitere Ersetzungsregeln für IA32-Befehle auf, bis Sie in der Lage sind die Ausdrucksbäume aus der vorherigen Teilaufgabe zu überdecken. Geben Sie IA32-Befehle für die Ausdrucksbäume an.

**3.3** *Registerzuteilung mit Linear Scan*

- Erweitern Sie die IA32-Befehle der Ausdrucksbäume zu einer vollständigen Implementierung der Funktion f.
- Nehmen Sie an es stehen ihnen nur 3 Register zur Verfügung. Teilen sie die Register mithilfe des Linear Scan Verfahrens zu und geben Sie den resultierden IA32-Assembler an.