Rekursives Programmieren in Assembler

TechGI 2 - SoSe 2014

Sichern und Wiederherstellen relevanter Daten

Beim Aufruf von Unterprogrammen oder rekursiven Funktionen können Daten verloren gehen (Register werden Überschrieben). Deshalb müssen wichtige Registerinhalte gesichert und nach dem Funktionsaufruf wiederhergestellt werden.

Wann muss gesichert werden:

- Bevor **`jal <Label>`** aufgerufen wird

Was muss gesichert werden:

- Rücksprungadresse (**`$ra`**)
- Zwischenergebnisse (**`$v0`**)
- Parameter für weitere Funktionsaufrufe (**`$a0 – $a3`**)

Wie muss gesichert werden:

1. Platz auf dem Stack schaffen (**`addi $sp, $sp, -4`**)
   → Für jedes zu sichernde Register müssen 4 Byte freigehalten werden
   → "Unser" Stack wächst von hochwertigen zu niederwertigen Adressen (deshalb **`-4`**)
2. Relevante Daten auf dem Stack ablegen (**`sw $ra, 0($sp)`**)
   - Die Reihenfolge der zu sichernden Register ist frei wählbar
3. Es dürfen nur $s-Register und $ra gesichert werden (d.h. Daten in $s-Register kopieren)

Wann muss wiederhergestellt werden:

- Nach **`jal <Label>`** aufgerufen wird

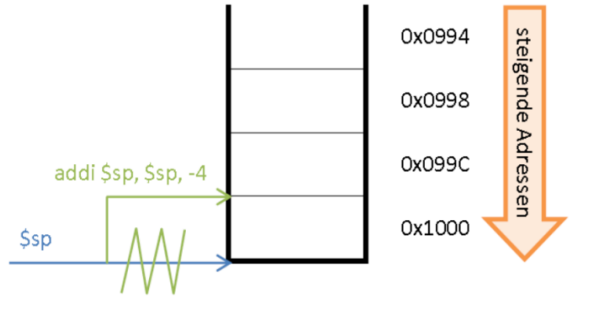
Was muss wiederhergestellt werden:

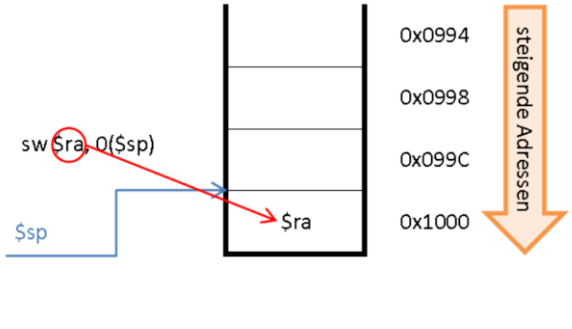
- Alles was gesichert wurde

  In manchen Fällen müssen nicht alle Register wiederhergestellt werden, alle Register wiederherzustellen ist jedoch einfacher und auch richtig

Wie muss wiederhergestellt werden:

1. Relevante Daten vom Stack laden (**`lw $ra, 0($sp)`**)
   - Die Reihenfolge der wiederherzustellenden Register MUSS der Reihenfolge des sicherns entsprechen
2. Platz auf dem Stack freigeben (**`addi $sp, $sp, 4`**)
   → Für jedes zu sichernde Register müssen 4 Byte freigehalten werden





Beispiele

f(n) = 1+f(n-1)	**$ra** muss gesichert werden Nach dem rekursiven Aufruf werden keine Parameter benötigt → Parameter müssen nicht gesichert werden Nach dem rekursiven Aufruf werden keine neuen Funktionen aufgerufen → Zwischenergebnisse müssen nicht gesichert werden Es muss also nur ein Register gesichert werden bzw. der Stackpointer muss um 4 Byte verringert werden (**$ra**).
f(x) = f(x-1)*x	**$ra** muss gesichert werden Nach dem rekursiven Aufruf wird der Parameter zur Multiplikation benötigt → der Parameter muss gesichert werden Nach dem rekursiven Aufruf werden keine neuen Funktionen aufgerufen → Zwischenergebnisse müssen nicht gesichert werden Es müssen also zwei Register gesichert werden bzw. der Stackpointer muss um 8 Byte verringert werden (**$ra**, **$a0**).
f(x,y) = f(x,0)+f(0,0)	**$ra** muss gesichert werden Nach dem rekursiven Aufruf werden keine Parameter benötigt → Parameter müssen nicht gesichert werden Nach dem rekursiven Aufruf wird eine weitere Funktionen aufgerufen → das erste Zwischenergebnis muss gesichert werden Es müssen also zwei Register gesichert werden bzw. der Stackpointer muss um 8 Byte verringert werden (**$ra**, **$v0**).
f(x) = f(x-1)*f(x-2)	**$ra** muss gesichert werden Nach dem rekursiven Aufruf wird der Parameter als Parameter benötigt → der Parameter muss gesichert werden Nach dem rekursiven Aufruf wird eine weitere Funktionen aufgerufen → das erste Zwischenergebnis muss gesichert werden Es müssen also drei Register gesichert werden bzw. der Stackpointer muss um 12 Byte verringert werden (**$ra**, **$v0**, **$a0**).