School of Computation, Information and Technology
Technische Universität München

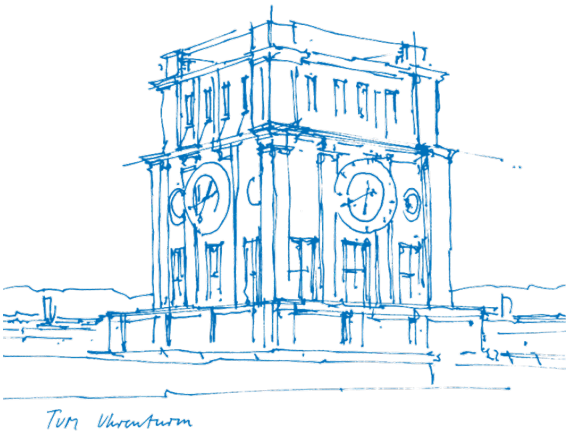
Übung 04: Calling Convention und Rekursion

Einführung in die Rechnerarchitektur

Niklas Ladurner

School of Computation, Information and Technology
Technische Universität München

10. November 2023



Durchzählen!

Keine Garantie für die Richtigkeit der Tutorfolien: Bei Unklarheiten/Unstimmigkeiten haben VL/ZÜ-Folien Recht!

Caller vs. Callee

```asm
caller:
    # hier befinden wir uns in der aufrufenden
    # Funktion (Caller)

    # Wir speichern die Rücksprungadresse auf
    # den Stack -> ra ist Caller-saved!
    addi sp, sp, -16
    sw ra, 0(sp)

    # ... irgendwas, das t0 verwendet, bspw.
    addi t0, zero, 2

    # da t0 caller-saved ist, müssen wir uns
    # t0 absichern, wenn wir den Inhalt später
    # noch brauchen
    sw t0, 4(sp)
    # ...
    jal ra, callee  # Sprung zur Unterfunktion
    # ...
    lw t0, 4(sp)
    # ... wieder irgendwas mit t0
    lw ra, 0(sp)
    addi sp, sp, 16
    jalr zero, 0(ra)
```

```asm
callee:
    # hier befinden wir uns in der aufgerufenen
    # Funktion (Callee)

    # hier dürfen wir t0-t6 bspw. verändern
    # falls wir s0-s6 verändern wollen würden,
    # würden wir das so machen:
    addi sp, sp, -16
    sw s2, 0(sp)
    sw s3, 4(sp)

    # s2, s3 können jetzt verwendet werden!
    # ...

    lw s2, 0(sp)
    lw s3, 4(sp)
    addi sp, sp, 16
    jalr zero, 0(ra)
```

Caller- und Callee-saved Register

Register	ABI Name	Description	Saver
x0	zero	Hard-wired zero	—
x1	ra	Return address	Caller
x2	sp	Stack pointer	Callee
x3	gp	Global pointer	—
x4	tp	Thread pointer	—
x5–7	t0–2	Temporaries	Caller
x8	s0/fp	Saved register/frame pointer	Callee
x9	s1	Saved register	Callee
x10–11	a0–1	Function arguments/return values	Caller
x12–17	a2–7	Function arguments	Caller
x18–27	s2–11	Saved registers	Callee
x28–31	t3–6	Temporaries	Caller
f0–7	ft0–7	FP temporaries	Caller
f8–9	fs0–1	FP saved registers	Callee
f10–11	fa0–1	FP arguments/return values	Caller
f12–17	fa2–7	FP arguments	Caller
f18–27	fs2–11	FP saved registers	Callee
f28–31	ft8–11	FP temporaries	Caller

Abbildung 1 Übersicht über die RISC-V-Register

Calling Convention

- „Aufrufkonvention“ $\rightarrow$ lediglich eine Vereinbarung
- definiert Parameterüberabe, Rückgabe, Registersicherung, Stack etc.
- Datentypen $\leq$ 4 Byte in a-Registern, signextended
- Datentypen = 8 Byte in 2 a-Registern, niedrigwertige Hälfte zuerst
- Datentypen > 8 Byte als Pointer (Zeiger auf Speicher)
- Falls zu wenige Register: Übergabe über Stack
- Stackpointer 16 Byte aligned!

Rekursion

- Funktion die sich selbst aufruft
- Stacklayout sehr wichtig, ra sichern
- $\exists$ äquivalente iterative Funktion für jede rekursive Funktion
- Aufbau: Abbruchbedingung(en), Sicherung, rekursiver Aufruf, Wiederherstellung, Berechnung, Rücksprung

Fragen?

(Die ZÜ-Folien sind sehr gut, schaut euch die an)

Artemis-Hausaufgaben

- H04 — Tribonacci bis 19.11.2023 23:59 Uhr
- Implementierung einer rekursiven Funktion
- Aufbau von Folie Rekursionbeachten
- Einhaltung der CC verpflichtend $\rightarrow$ siehe Link

Links

- Zulip: „ERA Tutorium - Mi-1600-MI4“ bzw. „ERA Tutorium - Fr-1100-MW2“
- RISC-V Spezifikation
- RISC-V ABI (Calling Convention)
- übersichtlichere Instruktionsliste

School of Computation, Information and Technology
Technische Universität München

Übung 04: Calling Convention und Rekursion

Einführung in die Rechnerarchitektur

Niklas Ladurner

School of Computation, Information and Technology
Technische Universität München

10. November 2023