



Technische Informatik II im SS 2005

Aufgaben zu den Tutorien in der Woche
vom 23. bis 25. Mai 2005

Dr.-Ing. T. Asfour

Haid-und-Neu-Str.7, Geb. 07.21
D-76131 Karlsruhe
Telefon: +49-721-608-7379
Fax: +49-721-608-8270
Email: asfour@ira.uka.de
<http://i61www.ira.uka.de/users/asfour/TI>

Lernziele:

- MIPS-Assembler

Aufgabe 1

1. Gegeben sei das folgende Programmstück:

```
for (i=0; i < 100; i++)
{
    A[i] = B[i] + c;
}
```

Dabei sind A und B Felder von 32-Bit Integerzahlen. Die Anfangsadressen der Felder A und B sind 0x100 und 0x4000. Die Variable c ist ebenfalls eine 32-Bit Integerzahl an der Adresse 0x1000. Speichern Sie den letzten Wert von i an der Adresse 0x2000.

Schreiben Sie das Programmstück in MIPS-Assembler um.

2. Geben Sie die Register- und Speicherinhalte nach der Ausführung des folgenden MIPS-Programms an. Verwenden Sie die im Lösungsblatt angegebenen Tabellen.

```
addi $t3, $zero, 0x38
lw   $t1, 4($t3)
sw   $t4, 8($t3)
sw   $t4, -8($t3)
add  $t2, $t3, $t1
sll  $t0, $t0, 4
```

Registersatz	
Register	Inhalt
\$t0	0x10
\$t1	0x14
\$t2	0x16
\$t3	0x38
\$t4	0x2003

Hauptspeicher	
Adresse	Inhalt
0x30	0x200
0x34	0x300
0x38	0x400
0x3C	0x500
0x40	0x600

Aufgabe 2

Analysieren Sie das folgende MIPS-Programm.

```

        .data
x:      .word 3
Y:      .word 1, 3, 7

        .text
subroutine: li $v0, 0
           li $t3, 0
marke1:   bge $t3, $a1, marke2
           lw $t0, 0($a0)
           mul $t1, $t0, $t0
           add $v0, $v0, $t1
           addi $a0, $a0, 4
           addi, $t3, $t3, 1
           b marke1
marke2:   jr $ra

        .globl main
main:     la $a0, Y
           lw $a1, x

           jal subroutine

#        Aufruf eines Unterprogramms, welches die
#        Quadratwurzel einer Integerzahl in $v0 berechnet.
#        Das Ergebnis steht in $v0

           move $a0, $v0
           li $v0, 1
           syscall

           li $v0, 10
           syscall
           jr $ra

```

1. Welche Funktion hat das Unterprogramm `subroutine`?
Welche Ausgabe hat das gesamte Programm? (Hinweis: Bei dem Systemaufruf `print_int` muß das Argument im Register `$a0` stehen).
2. Geben Sie die MIPS-Instruktionen zu den folgenden Pseudoinstruktionen an.
 - (a) `b marke`
 - (b) `neg $s3, $s2`
3. Was bewirkt die Assemblerdirektive `.align 3` ?
4. Warum dürfen bei Arithmetikoperationen mit doppelter Genauigkeit nur die Register mit gerader Registernummer verwendet werden?
5. Welche Adressen haben A, B, C und D im folgenden MIPS-Programmabschnitt.

```
        .data 0x10000003
        .align 3
A:      .byte 6, 5
B:      .word 7, 4
C:      .double 3.1415
D:      .float 2.71828
```

6. Welche Gründe machen die Programmierung der MIPS-Architektur schwierig?

Aufgabe 3

1. Schreiben Sie die folgenden Kontrollstrukturen in MIPS-Assembler um. Sie dürfen nur die MIPS-Befehle `slt`, `beq` und `bne` verwenden. Zur Speicherung temporärer Variablen verwenden Sie das Register `$at`. Die Variable `a` ist im Register `$t4`, die Variable `b` im Register `$s0`.

```
(a)    if ( a <= b )
        {
            ...
        }
marke1:
```

```
(b)    if ( a >= b )
        {
            ...
        }
marke2:
```

```
(c)    do
        {
            marke3:
                ...
        } while ( a != b )
```

2. Was sind die Unterschiede zwischen einer statischen Speicherallokierung und einer dynamischen Speicherallokierung?