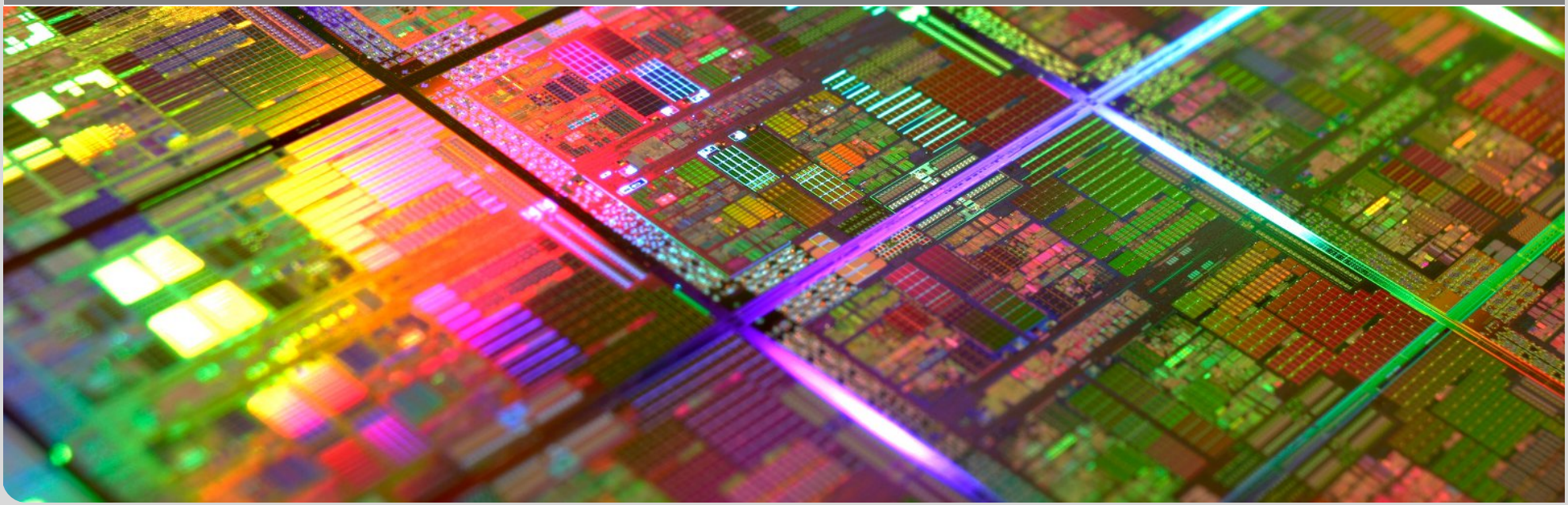
Karlsruher Institut für Technologie

RO-Tutorien 3 / 6 / 12

Tutorien zur Vorlesung “Rechnerorganisation”

Christian A. Mandery

WOCHE 7 AM 10./11.06.2013



Heute

- Übungsaufgaben

Übungsaufgabe 1.1

```
            .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
```

```
# Fortsetzung von links
            .globl main
main:       la $a0, Y
            lw $a1, x
            jal subroutine

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

            li $v0, 10
            syscall
            jr $ra
```

Welche Funktion das Unterprogramm subroutine?

Welche Ausgabe hat das gesamte Programm?

Übungsaufgaben 1.2 - 1.4

2. Geben Sie die MIPS-Instruktionen zu den folgenden Pseudoinstruktionen an:
   - `b marke`
   - `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?

Übungsaufgaben 1.2 - 1.4

2. Geben Sie die MIPS-Instruktionen zu den folgenden Pseudoinstruktionen an:
   - `b marke`
   - `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?

Übungsaufgaben 1.2 - 1.4

2. Geben Sie die MIPS-Instruktionen zu den folgenden Pseudoinstruktionen an:
   - `b marke`
   - `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?

Übungsaufgaben 1.2 - 1.4

2. Geben Sie die MIPS-Instruktionen zu den folgenden Pseudoinstruktionen an:
   - `b marke`
   - `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?

Übungsaufgaben 1.5 - 1.6

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?

Übungsaufgaben 1.5 - 1.6

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?

Übungsaufgabe 2.1

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` abgelegt.

1. ```
   if ( a <= b ) { ... }
   marke1:
   ```

2. ```
   if ( a >= b ) { ... }
   marke2:
   ```

3. ```
   do {
     marke3: ...
             ...
   } while ( a != b );
   ```
   ```

Übungsaufgabe 2.1

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` abgelegt.

1. ```
   if ( a <= b ) { ... }
   marke1:
   ```

2. ```
   if ( a >= b ) { ... }
   marke2:
   ```

3. ```
   do {
     marke3: ...
             ...
   } while ( a != b );
   ```
   ```

Übungsaufgabe 2.1

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` abgelegt.

1. ```
   if ( a <= b ) { ... }
   marke1:
   ```

2. ```
   if ( a >= b ) { ... }
   marke2:
   ```

3. ```
   do {
     marke3: ...
             ...
   } while ( a != b );
   ```
   ```

Übungsaufgabe 2.2

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

Übungsaufgabe 3.1

Setzen Sie die folgende C-Kontrollstruktur mit MIPS-Assembler um. Die Variablen `a` und `b` vom Typ `int32_t` sollen hierbei in den Registern `$s0` und `$s1` abgelegt werden.

```
if (a * 2 < b) {
  a *= 2;
}
```

Übungsaufgabe 3.2

Setzen Sie die folgende C-Kontrollstruktur mit MIPS-Assembler um. Die Variablen `a` und `b` vom Typ `int32_t` sollen hierbei in den Registern `$s0` und `$s1` abgelegt werden.

```
for (a = 10; a >= 0; a -= 2) {
  b += a;
}
```

Übungsaufgabe 3.3

Setzen Sie die folgende C-Kontrollstruktur mit MIPS-Assembler um. Die Variablen `i` und `sum` vom Typ `int32_t` sollen hierbei in den Registern `$s2` und `$s3` abgelegt werden. Das Array `array` ist im Datensegment durch eine `array: .word ...` Direktive abgelegt.

```
int array[100];
...
sum = 0;
for (i = 0; i < 100; i++)
  sum = sum + array[i];
```

Übungsaufgabe 4

1. Was ändert sich am Assembler-Code der Teilaufgabe 3.3, wenn die Variablen im Datensegment abgelegt und mit Marken entsprechend der Namen der Variablen versehen sind?
2. Beschreiben Sie die Funktion der folgenden MIPS-Befehle:
   - addu $t3, $t2, $t1
   - andi $t3, $t2, 0x2000
   - slt $t3, $t2, $t1
   - lui $t3, 0x2000
3. In welchem Register wird die Rücksprungadresse beim Unterprogrammaufruf gesichert?

Übungsaufgabe 4

1. Was ändert sich am Assembler-Code der Teilaufgabe 3.3, wenn die Variablen im Datensegment abgelegt und mit Marken entsprechend der Namen der Variablen versehen sind?
2. Beschreiben Sie die Funktion der folgenden MIPS-Befehle:
   - addu $t3, $t2, $t1
   - andi $t3, $t2, 0x2000
   - slt $t3, $t2, $t1
   - lui $t3, 0x2000
3. In welchem Register wird die Rücksprungadresse beim Unterprogrammaufruf gesichert?

Übungsaufgabe 4

1. Was ändert sich am Assembler-Code der Teilaufgabe 3.3, wenn die Variablen im Datensegment abgelegt und mit Marken entsprechend der Namen der Variablen versehen sind?
2. Beschreiben Sie die Funktion der folgenden MIPS-Befehle:
   - addu $t3, $t2, $t1
   - andi $t3, $t2, 0x2000
   - slt $t3, $t2, $t1
   - lui $t3, 0x2000
3. In welchem Register wird die Rücksprungadresse beim Unterprogrammaufruf gesichert?

Übungsaufgabe 5

1. Geben Sie für das folgende MIPS-Programmstück den Inhalt des Zielregisters nach der Ausführung des jeweiligen Befehls in hexadezimaler Schreibweise an.

```
ori $s1, $zero, 20
sll $s2, $s1, 3
slti $s3, $s2, 100
sub $s4, $s3, $s2
lui $s5, -7
```

2. Wie ist die Trennung von Programmen und Daten bei der Ihnen bekannten MIPS-R2000-Architektur realisiert?

Übungsaufgabe 5

1. Geben Sie für das folgende MIPS-Programmstück den Inhalt des Zielregisters nach der Ausführung des jeweiligen Befehls in hexadezimaler Schreibweise an.

```
ori $s1, $zero, 20
sll $s2, $s1, 3
slti $s3, $s2, 100
sub $s4, $s3, $s2
lui $s5, -7
```

2. Wie ist die Trennung von Programmen und Daten bei der Ihnen bekannten MIPS-R2000-Architektur realisiert?

Fertig!

Quelle: http://xkcd.com/749/