Ludwig-Maximilians-Universität München
Institut für Informatik
Lehrstuhl für Mobile und Verteilte Systeme
Prof. Dr. Claudia Linnhoff-Popien

mobile and distributed systems group

Rechnerarchitektur im Sommersemester 2017
Übungsblatt 6

Abgabetermin: 12.06.2017, 12:00 Uhr

Besprechung: Besprechung der T-Aufgaben in den Tutorien vom 07. – 09. Juni 2017
Besprechung der H-Aufgaben in den Tutorien vom 12. – 16. Juni 2017

Aufgabe 29: (T) Umsetzung Boolescher Ausdrücke

(– Pkt.)

Übersetzen Sie folgendes Pseudocodefragment in MIPS-Code. Gehen Sie davon aus, dass der Wert der Variablen `a` bereits in das Register `$t0` geladen wurde.

```
1 IF (a < 0) OR (a > 99) THEN
2     a := a - 10;
3 ELSE
4     a := a - 1;
5 END;
```

Bedenken Sie dabei insbesondere: Der Ausdruck `a > 99` wird nur dann ausgewertet, wenn `a < 0` fehlgeschlagen ist.

Aufgabe 30: (T) SPIM Programmieraufgabe

(– Pkt.)

Erstellen Sie ein **vollständiges** SPIM–Programm, das folgendes durchführt:

- Es werden zwei positive Integer–Zahlen von der Konsole eingelesen.
- Es wird der Durchschnitt dieser beiden Zahlen auf eine Nachkommastelle genau berechnet.
- Das Ergebnis der Berechnung wird ausgegeben.

Tipp: Programmieren Sie diejenigen Schritte, die Sie auch beim handschriftlichen Dividieren durchführen!

Beachten Sie hierbei folgendes:

- Verwenden Sie nur die **unten aufgeführten Befehle**.
- Verwenden Sie für die Vorkommazahl das Register `$s0` und für die Nachkommazahl das Register `$s1`, ansonsten nur die temporären Register.
- **Kommentieren** Sie ihr Programm sinnvoll!
- Sowohl die Eingabe als auch die Ausgabe soll mit einem Anweisungstext versehen werden, wie z.B. *"Geben Sie die 1. Zahl ein: "*, etc.

Aufgabe 31: (T) Anwendungen der Digitalisierung

(– Pkt.)

In der Vorlesung „Rechnerarchitektur“ werden Grundlagen der Digitalisierung behandelt. Während diese in Zeiten der Entstehung erster Rechnerarchitekturen zunächst die Hardware betrafen, vollzog sich sukzessive eine Erweiterung hin zur Software und dann zur Digitalisierung von Prozessen und ganzen Branchen. Die Auswirkungen sind heute in unserem Leben und Arbeiten, in den Medien und unserer Gesellschaft sichtbar. Diesen Aspekt wollen wir mit Hilfe des Buchs „Marktplätze im Umbruch“ betrachten.

Beantworten Sie für jedes der folgenden Geleitworte die Frage:
Welche Beobachtungen macht der jeweilige Autor im Zeitalter der Digitalisierung in seinem Arbeitsumfeld?

– Andreas Kottmann: Mobile Services - Car Sharing, Parken und Intermodalität

– Alexander Sixt: Flexible Mobilität

Lesen Sie des Weiteren den folgenden Artikel und beantworten Sie die Frage:
Welche Mobilitätsprodukte nennen die Autoren und wovon hängt deren Erfolg und Akzeptanz ab?

– Robert Lasowski, Oliver Höft, Alexander Boone und Eric-Alexander Schäfer: Mobilität der Zukunft – Eine Vision, die beherrscht werden will

Hinweis: Als Student können Sie sich ein freies Exemplar des Buches „Marktplätze im Umbruch“ aus dem Internet herunterladen. Dabei haben Sie die folgenden 2 Möglichkeiten:

a. Rufen Sie aus dem LRZ-Netz den folgenden Link zum Buch auf: `http://link.springer.com/book/10.1007/978-3-662-43782-7`. Verwenden Sie dabei den PAC-Proxy (`https://www.lrz.de/services/netzdienste/proxy/zeitschriftenzugang/`)

b. Rufen Sie den folgenden Link zum Buch auf: `http://link.springer.com/book/10.1007/978-3-662-43782-7`. Gehen Sie auf „Sign up/ Login“ und dort auf „Log in via Shibboleth or Athens“. Unter „find your institution“ geben Sie „LMU“ ein und klicken Sie auf „Log in via Shibboleth“. Es folgt die Weiterleitung zur LMU und der Login mit Ihrer Campus-Adresse. Nach Bestätigung können Sie auf der Springer-Seite das Buch herunterladen.

Aufgabe 32: (H) Test des MIPS Simulators

(8 Pkt.)

Für diese Aufgabe sollten Sie sich mit dem MIPS-Simulator SPIM vertraut machen. Sie können einen MIPS-Simulator von der Vorlesungshomepage herunterladen, oder XSPIM aus dem CIP-Pool benutzen.

– Laden Sie sich das Assemblerprogramm `simple.s` von der Rechnerarchitektur–Homepage herunter und speichern Sie es in Ihrem Home–Verzeichnis ab.

– Starten Sie Ihren Simulator. Im CIP-Pool: `xspim` auf der Konsole eingeben.

– Laden Sie das Programm `simple.s` in den Simulator und führen Sie es aus. Dabei sollte eine Konsole erscheinen, über die die Ein- und Ausgabe erfolgt.

Beantworten Sie nun folgende Fragen.

a. Welches Ergebnis liefert das Programm für die Eingabefolge "1, 2, 3, 4, 0"? (D.h. nach Start des Programms erfolgt über die Konsole die Eingabe "1", gefolgt von *Enter*, dann die Eingabe "2", gefolgt von *Enter*, usw.)

b. Kommentieren Sie **jede** Zeile des Programms sinnvoll. (Geben Sie bei Ihrer Abgabe die Zeilennummer und den zugehörigen Kommentar an.)

c. Welche mathematische Funktion berechnet das Programm?

Aufgabe 33: (H) Einfachauswahlaufgabe: MIPS/SPIM

(5 Pkt.)

Für jede der folgenden Fragen ist eine korrekte Antwort auszuwählen („1 aus n“). Eine korrekte Antwort ergibt jeweils einen Punkt. Mehrfache Antworten oder eine falsche Antwort werden mit 0 Punkten bewertet.

a) Der MIPS Prozessor besitzt die folgende Architektur:			
(i) Stack	(ii) MISC	(iii) CISC	(iv) RISC
b) Jedes MIPS-Register hat eine feste Breite. Sie beträgt:			
(i) 64 Bit	(ii) 32 Bit	(iii) 16 Bit	(iv) 8 Bit
c) In der MIPS Architektur steht ein Wort für...			
(i) ...die maximale Datengröße, die in einem Rechenschritt verarbeitet werden kann.	(ii) ...die Größe einer Speicherzelle	(iii) ...die größte adressierbare Informationseinheit	(iv) ...die kleinste adressierbare Informationseinheit.
d) Wie muss der Assembler-Befehl lauten, wenn der Inhalt von Register `$t1` durch den Inhalt von Register `$t2` dividiert und das Ergebnis im Zielregister `$t0` gespeichert werden soll?			
(i) `div $t1,$t0,$t2`	(ii) `div $t2,$t1,$t0`	(iii) `div $t0,$t1,$t2`	(iv) `mul $t2,$t1,$t0`
e) Gegeben sei folgende Zeile in SPIM Code: `var: .word 10, 11, 12, 13` Welcher Befehl lädt den Wert 11 in das Register $t0?			
(i) `lw var, $t0+4`	(ii) `lw $t0, var+4`	(iii) `lw $t0, var`	(iv) `la $t0, var+4`

Überblick über die wichtigsten SPIM Assemblerbefehle		
Befehl	Argumente	Wirkung
add	Rd, Rs1, Rs2	Rd := Rs1 + Rs2 (mit Überlauf)
sub	Rd, Rs1, Rs2	Rd := Rs1 - Rs2 (mit Überlauf)
addu	Rd, Rs1, Rs2	Rd := Rs1 + Rs2 (ohne Überlauf)
subu	Rd, Rs1, Rs2	Rd := Rs1 - Rs2 (ohne Überlauf)
addi	Rd, Rs1, Imm	Rd := Rs1 + Imm
addiu	Rd, Rs1, Imm	Rd := Rs1 + Imm (ohne Überlauf)
div	Rd, Rs1, Rs2	Rd := Rs1 DIV Rs2
rem	Rd, Rs1, Rs2	Rd := Rs1 MOD Rs2
mul	Rd, Rs1, Rs2	Rd := Rs1 $\times$ Rs2
b	label	unbedingter Sprung nach label
j	label	unbedingter Sprung nach label
jal	label	unbed.Sprung nach label, Adresse des nächsten Befehls in $ra
jr	Rs	unbedingter Sprung an die Adresse in Rs
beq	Rs1, Rs2, label	Sprung, falls Rs1 = Rs2
beqz	Rs, label	Sprung, falls Rs = 0
bne	Rs1, Rs2, label	Sprung, falls Rs1 $\neq$ Rs2
bnez	Rs1, label	Sprung, falls Rs1 $\neq$ 0
bge	Rs1, Rs2, label	Sprung, falls Rs1 $\geq$ Rs2
bgeu	Rs1, Rs2, label	Sprung, falls Rs1 $\geq$ Rs2
bgez	Rs, label	Sprung, falls Rs $\geq$ 0
bgt	Rs1, Rs2, label	Sprung, falls Rs1 > Rs2
bgtu	Rs1, Rs2, label	Sprung, falls Rs1 > Rs2
bgtz	Rs, label	Sprung, falls Rs > 0
ble	Rs1, Rs2, label	Sprung, falls Rs1 $\leq$ Rs2
bleu	Rs1, Rs2, label	Sprung, falls Rs1 $\leq$ Rs2
blez	Rs, label	Sprung, falls Rs $\leq$ 0
blt	Rs1, Rs2, label	Sprung, falls Rs1 < Rs2
bltu	Rs1, Rs2, label	Sprung, falls Rs1 < Rs2
bltz	Rs, label	Sprung, falls Rs < 0
not	Rd, Rs1	Rd := $\neg$ Rs1 (bitweise Negation)
and	Rd, Rs1, Rs2	Rd := Rs1 & Rs2 (bitweises UND)
or	Rd, Rs1, Rs2	Rd := Rs1 | Rs2 (bitweises ODER)
syscall		führt Systemfunktion aus
move	Rd, Rs	Rd := Rs
la	Rd, label	Adresse des Labels wird in Rd geladen
lb	Rd, Adr	Rd := MEM[Adr]
lw	Rd, Adr	Rd := MEM[Adr]
li	Rd, Imm	Rd := Imm
sw	Rs, Adr	MEM[Adr] := Rs (Speichere ein Wort)
sh	Rs, Adr	MEM[Adr] MOD 2^{16} := Rs (Speichere ein Halbwort)
sb	Rs, Adr	MEM[Adr] MOD 256 := Rs (Speichere ein Byte)

Funktion	Code in $v0	Funktion	Code in $v0
print_int	1	read_float	6
print_float	2	read_double	7
print_double	3	read_string	8
print_string	4	sbrk	9
read_int	5	exit	10