Lehrstuhl für Informatik III — Prof. Dr. M. Gössel	
Rechnerarchitektur (Sommersemester 2000)	
Übungen: M. Seuring, A. Dmitriev, P. Vogel	
Übungsblatt Nr. 11	20.06.2000
Abgabetermin: 27.06.2000	

Wichtig! Die Vorlesung vom 04. Juli 2000 wird verschoben auf den 07. Juli 2000 15.15. - 16.45 im kleinen Physikhörsaal. SPIM-Übungen finden am Mittwoch (HPI) und Donnerstag wie üblich statt.

SPIM-Hausarbeit! Die Ausgabe der Aufgabenstellung erfolgt am Donnerstag, den 29. Juni 2000, vor der Übung. Die Aufgabenstellung kann auch ab diesem Termin von der Webseite zur Übung geladen werden.

Hinweis! Größere Programme als Lösung der Aufgaben können ohne abgegebene Diskette nicht korrigiert werden. Wer Lösungen ohne Diskette abgegeben hat, umgehend melden und Diskette nachreichen!!!

Arbeit mit der Gleitkommaeinheit

Für die Verarbeitung von Gleitkommazahlen hat der MIPS-Prozessor eine Gleitkommaeinheit. Dieser mathematische Koprozessor verfügt ebenfalls 32 Register von 32 Bit Breite. Die Register werden mit $f0 bis $f31 angesprochen. Es sind general purpose Register, die beschreibbar und für jeden Zweck einsetzbar sind. Es gelten folgende Einschränkungen: für das Rechnen mit doppelter Genauigkeit werden zwei Register zusammengefaßt (dann 64 Bit). Zusammengefaßt werden ein Register mit gerader Nummer und das folgende Register (z.B. $f0 und $f1, $f2 und $f3). Die Register $f0 und $f1 werden vom Betriebssystem für die Benutzereingaben und die Register $f12 und $f13 für die Ausgaben benutzt. Unter Windows gibt es Probleme mit dem Anzeigen der Werte, aber nur mit dem Anzeigen.

Aufgabe 41

Das Ablegen der Gleitkommazahlen im Datensegment erfolgt über die Direktiven `.float` und `.double`.

a) Legen Sie im Datensegment je zwei Gleitkommazahlen einfacher und zwei doppelter Genauigkeit ab. Laden Sie dann die Daten in die geeigneten Floating-point-Register (Befehle `l.d` und `l.s`)

b) Verändern Sie die Werte in den FP-Registern und schreiben Sie die Werte wieder in das Datensegment.

Aufgabe 42

Daten können zwischen dem Haupt- und dem Koprozessor hin- und herbewegt werden. Dazu gibt es folgende Befehle:

`mfcz Rdest, CPsrc`	*Hole vom Koprozessor z Register `CPsrc` nach `Rdest`*
`mfc1.d Rdest, FRsrc1`	*Hole Double vom Koprozessor 1 aus `FRsrc1/FRsrc+1` nach `Rdest/Rdest+1`*
`mtcz Rsrc, CPdest`	*Bringe zum Koprozessor z*

Beachte: Der letzte Befehl funktioniert unter Windows nicht! Zum Laden eines unmittelbaren Wertes muß dieser zuvor im Speicher abgelegt und dann von dort geladen werden.
Befehle zum Verschieben der Registerinhalte im Koprozessor sind:

`mov.d FRdest, FRsrc` *Verschiebe Floating Point Double*

`mov.s FRdest, FRsrc` *Verschiebe Floating Point Single*

Erweitern Sie das Programmfragment aus **Aufgabe 41** wie folgt:

a) Laden Sie die Werte aus den FP-Registern in Register des Hauptprozessors (single und double beachten).

b) Verschieben Sie die Werte innerhalb der Register der FP-Einheit. Wie reagiert der Simulator, wenn eine Double-Zahl auf ein ungerades Register (z.B. `$f5`) geschrieben werden soll?

Aufgabe 43

Die Additions-Befehle unterscheiden sich von den Versionen für ganze Zahlen nur unwesentlich (keine Version ohne Überlaufbehandlung, dafür aber Versionen für double und single). Für die anderen arithmetischen Operationen gibt es ebenfalls Varianten für einfache und doppelte Genauigkeit (siehe Beiblatt bzw. Spimdokumentation). Die sechs Vergleichsbefehle setzen das `Bedingungs-Bit (condition-flag)` des Koprozessors auf 1 (wahr) bzw. 0 (falsch). Dieses `condition-flag` kann als Sprungkriterium verwendet werden. Die Vergleichsbefehle und die Verzweigungsbefehle finden Sie im Beiblatt/Spimdokumentation).

a) Schreiben Sie ein Programmfragment, daß einen bedingten Sprung (If ... then ... else ...) für den FP-Koprozessor nachbildet.

b) Schreiben Sie je ein Unterprogramm `print_float / print_double`, daß FP-Werte enfacher bzw. doppelter Genauigkeit ausgibt.

Aufgabe 44

Schreiben und testen Sie ein Assemblerprogramm, das einen Integerwert n einliest, die Wurzel aus n berechnet und in dem Format "Das Programm hat die Wurzel von n in i Iterationen berechnet. Die Wurzel von n ist x" ausgibt. Verwenden Sie folgendes Berechnungsschema:

```
begin
     wurzel_alt := n
     repeat
        wurzel_neu := 0.5 *(wurzel_alt+n/wurzel_alt)
        differenz := wurzel_alt - wurzel_neu;
        wurzel_alt := wurzel_neu
     until differenz <= 0;
end
```

Unter der Anzahl der Iterationen verstehen wir die Anzahl der Schleifendurchläufe. Beachten Sie, daß Sie in Ihrem Programm Gleitkommazahlen verwenden und somit den Coprozessor 1 ansprechen müssen.