

Rechnerstrukturen im WS 2010/2011 Übungsblatt 9

Hinweis: Kommentieren Sie immer Ihre selbstgeschriebenen Assemblerprogramme !!!

Aufgabe 1 (Assemblerprogrammierung) (4 Punkte)

In einer Assemblerprozedur soll die Register-Transfer-Anweisung durchgeführt werden:

```
Speicher[0x10010000] := (Speicher[0x10010004] - Speicher[0x10010008])  
                    * (Speicher[0x10010010] + 8) - Speicher[0x10010020]
```

Vervollständigen Sie die nachfolgende Sequenz bis einschließlich Programmende. Die Verwendung von lw- und sw-Befehlen mit *Offset*-Werten, die nicht mit 16 Bit dargestellt werden können, ist nicht zulässig. Verwenden Sie möglichst wenige Register und möglichst wenige Befehle!

```
li    $2, 0x10010000  
lw    $3, 0x4($2)  
lw    $4, 0x8($2)
```

Aufgabe 2 (Assemblerprogrammierung) (4 Punkte)

Schreiben Sie ein MIPS-Programm, welches den Ausdruck $(x * y) / z$ berechnet.

Die Zuordnung der drei Variablen bezüglich der CPU-Register sei wie folgt:

\$2 x, **\$3** y, **\$4** z.

Die Variablen sollen wie folgt belegt werden:

x = 1000000 (0xF4240), y = 8000 (0x1F40), z = 4000 (0xFA0).

Ergänzen Sie das gegebene Programm so, dass am Ende des Programmablaufs das Berechnungsergebnis im Register **\$5** steht. Das berechnete Ergebnis muss korrekt sein. (2 Millionen)

Hinweis: Befehl sll (Anhang Script).

```
.data  
x:      .word 0           # Speicher reservieren  
y:      .word 0x1f40      # Konstante y  
z:      .word 0xfa0      # Konstante z  
.text  
.globl main  
main:   ori  $2, 0xF424    # der Wert für x ist noch zu klein  
        ...              # und wird hier korrigiert  
.  
.  
.  
        li  $2, 10        # Programm ordnungsgemaess beenden  
        syscall          #
```

Aufgabe 3 (Einfache Fallunterscheidung) (4 Punkte)

Ein MIPS-Programm soll bei der Auswertung der Klausurkorrekturen helfen. Es soll feststellen, ob jemand bestanden hat (Punkte ≥ 40) oder nicht (Punkte < 40). Zunächst ist die Variable „Bestanden“ mit 99 belegt, was soviel bedeutet wie „die Note steht noch nicht fest“.

Ergänzen Sie das Programmfragment so, dass in „Bestanden“ eine 1 für bestanden und eine 0 für „nicht bestanden“ steht.

```
.data
Punkte:      .word 42           # Klausurpunkte
Bestanden:   .word 99           # 0 Nein, 1 Ja, 99 weiß nicht
Grenze:      .word 40           # Bestehensgrenze 40Punkte

.text
.globl main
```

Aufgabe 4 (For-Schleife) (4 Punkte)

Implementieren Sie die Berechnung der Parität einer Variablen „wert“ (Typ .word). Der ermittelte Wert der Parität soll in der Variablen „pari“ (Typ .word) abgelegt werden.

Wenn die Binärdarstellung der Zahl eine gerade Anzahl von 1-Bits enthält, soll der Wert 0 zurückgeliefert werden, bei ungerader Anzahl der Wert 1. (Beispielsweise ermittelt das Programm für die Zahl $23=(0010111)_2$ den Rückgabewert 0. Für die Zahl $38=(0100110)_2$ ergibt sich der Rückgabewert 1.)

Hinweis: Sie benötigen keinen Multiplikationsbefehl. Der Befehl „srl“ könnte von Nutzen sein. (Anhang Script)

Die Abgaben sollen bis Mittwoch, den 15. Dezember 2010 um 20.00 Uhr in die Briefkästen im Pavillon 6 eingeworfen werden. Bitte Name (bei einem 3er-Team alle), Matrikel- und Gruppennummer oben auf der ersten Seite der Lösungen angeben.