heiko.falk [☺] tu-dortmund.de
timon.kelter [☺] tu-dortmund.de

Übung zur Vorlesung
Rechnerarchitektur
Sommersemester 2011

Übungsblatt 11

(10 Punkte)

Abgabe bis spätestens Mittwoch, 22. Juni 2011, 12:00 Uhr

11.1 Beispiel zur Sequentiellen Speicherkonsistenz (4 Punkte)

Es wird angenommen, dass drei Prozessoren P_1, P_2, P_3 eines Mehrprozessorsystems auf einen gemeinsamen Speicher zugreifen und dass das System sequentiell konsistent ist. Vor Ausführung des Programmcodes gilt $A = B = 0$. Die drei Prozessoren führen nun eine Reihe von Operationen gemäss der unten stehenden Tabelle aus. Dabei ist o_{i1} die erste Operation, die Prozessor P_i ausführt, und o_{i2} die zweite.

P_1	P_2	P_3
$o_{11} : A = 1$	$o_{21} : u = A$	$o_{31} : v = B$
	$o_{22} : B = 1$	$o_{32} : w = A$

Das Ergebnis der Ausführung kann durch das Zahlentupel (u, v, w) beschrieben werden. Dabei kann das Ergebnis variieren und zwar abhängig von der Reihenfolge, in der die Operationen der einzelnen Prozessoren global sichtbar gemacht werden. Einige der in der Tabelle unten gezeigten Tupel sind auf einem sequenziell konsistenten System nicht möglich. Füllen Sie die Tabelle folgendermassen aus (siehe auch das Beispiel in Zeile 2): Geben Sie an, ob das Tupel in der jeweiligen Zeile auf einem sequentiell konsistenten System als Ergebnis auftauchen kann und falls ja, nennen Sie eine mögliche Reihenfolge, in der die Operationen der einzelnen Prozessoren global sichtbar gemacht wurden.

u	v	w	**konsistent?**	**Reihenfolge**
0	0	0		
0	0	1	Ja	$o_{21}, o_{31}, o_{11}, o_{22}, o_{32}$
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

11.2 Locks (1 Punkt)

Ist die im Assemblercode des TriCore-Prozessors angegebenen Funktion eine sichere Implementierungen für das Setzen eines Locks? Begründen Sie ihre Antwort. (Eine Dokumentation zu dem TriCore-Befehlssatz finden Sie auf der Übungs-Webseite.)
`LO:var` bezeichnet die unteren 16Bit der Speicheradresse von Variable 'var'. `HI:var` steht für die oberen 16Bit der Speicheradresse von Variable 'var'.

```
lock:
      movh.a  %a12, HI:var
_loop:
      ld.w    %d8, [%a12] LO:var
      jne     %d8, 0, _loop
```

```
_success:
      mov     %d8, 1
      st.w    [%a12] LO:var, %d8
      ret
```

11.3 Locks (2 Punkte)

Wie kann mit Hilfe eines atomaren SWAP-Befehls, der den Inhalt eines Prozessorregisters mit dem einer Speicherstelle vertauscht, ein sicherer Lock implementiert werden?

11.4 Semaphore (3 Punkte)

In einem Programm führen zwei Threads `T1` und `T2` jeweils die folgenden Operationen aus.

T1	T2
$X = Y - 1$	$Y = 2$
$X = X + Y$	$Y = X - Y$

Zu Beginn sind beide Variablen mit dem Wert 0 initialisiert.

Ergänzen Sie die obigen Operationen mit den **notwendigen** Synchronisationsaktionen auf Basis von Semaphoren, so dass nach der Ausführung aller Operationen die Variablenbelegungen $X = 3$ und $Y = 1$ sind. Verwenden Sie die aus der Vorlesung bekannte Notation für Semaphore.

Allgemeine Hinweise: Die Übungstermine und weitere Informationen finden Sie unter:
http://ls12-www.cs.tu-dortmund.de/de/teaching/courses/ss11/ra/uebungen
Die Übungszettel werden donnerstags um 10:00 Uhr online gestellt und müssen bis zum darauf folgenden Mittwoch um 12:00 Uhr bearbeitet werden. Die Abgaben können in den beschrifteten Briefkasten vor dem Sekretariat des LS12 (Raum E22, OH16) eingeworfen werden. Auf Wunsch kann für diese Veranstaltung ein Übungsschein ausgestellt werden. Hierzu müssen 45% der Gesamtpunkte bei den Übungszetteln erreicht und eigene Lösungen in der Übungsgruppe präsentiert werden.