Rechnerarchitektur SS 2016

Exercises: Hardware Multithreading

Jian-Jia Chen

TU Dortmund

to be discussed on 20. April 2016

Aufgabe (Diskutieren am Mittwoch)

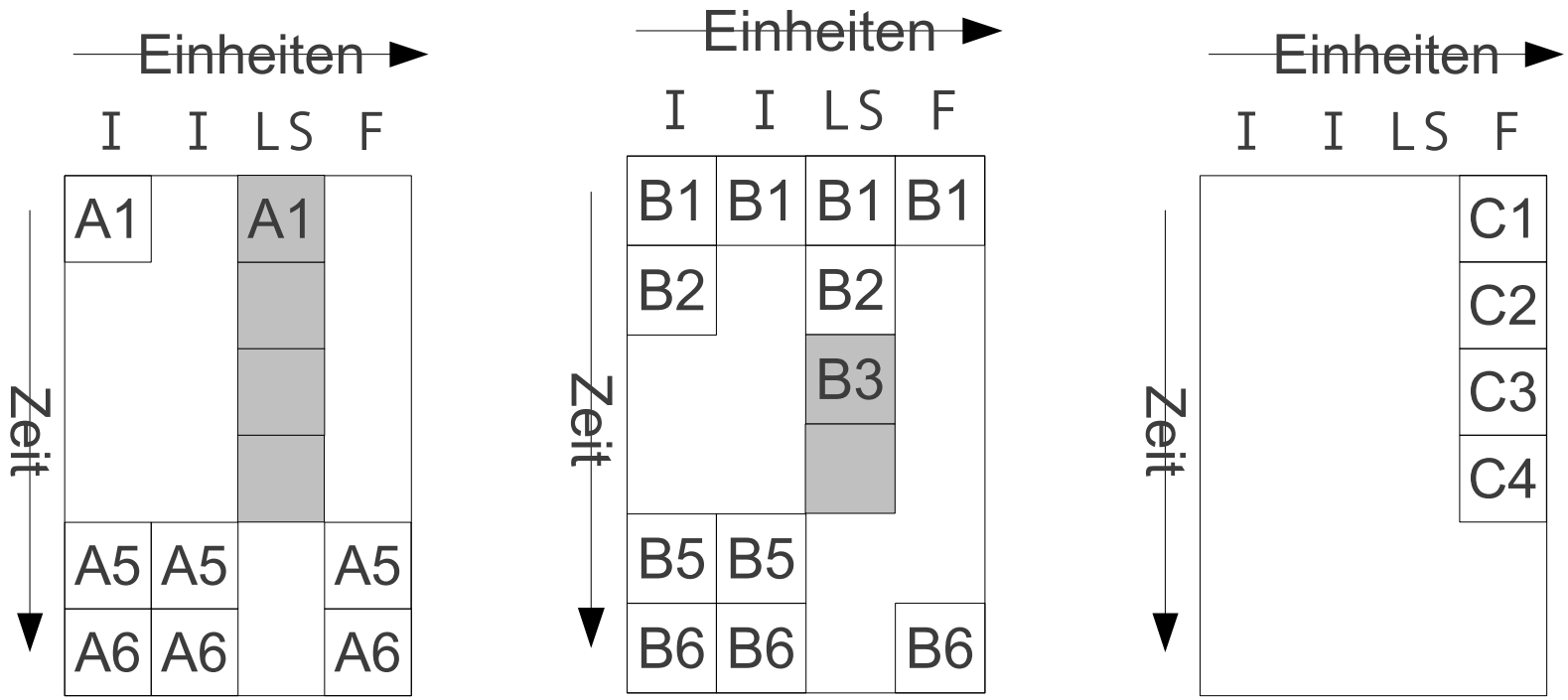
Gegeben sei dazu ein System mit

- einem Prozessor mit dynamischem Scheduling und
  - 2 Integer-Ausführungseinheiten (Latenz 1 Takt)
  - 1 Load/Store-Einheit (Latenz 1-4 Takte, je nach Cacheverhalten)
  - 1 Floating-Point-Einheit (Latenz 1 Takt)
- einem L1-Cache und einem L2-Cache. Die Latenzen der Zugriffe sind wie folgt:

L1-Zugriff	L2-Zugriff	Latenz (Takte)
Hit	-	1
Miss	Hit	2
Miss	Miss	4

Beide Caches unterstützen *Hit-Under-1-Miss*, d.h. während ein Miss bearbeitet wird, können weitere Hit-Zugriffe abgewickelt werden. Ein Zugriff, der einen weiteren Miss produzieren würde, wird abgebrochen und muss vom Prozessor wiederholt werden.

Aufgabe (fort.)



(a) Thread A (Priorität 0) (b) Thread B (Priorität 1) (c) Thread C (Priorität 2)

Abbildung: Ausführungsverlauf dreier Threads auf der gegebenen Plattform

Aufgabe (fort.)

Für dieses System sind die Ausführungsverläufe dreier Programme *A*-*C* in Abbildung 1 skizziert, wie sie sich bei isolierter Ausführung auf dem System ergeben. Operationen, die mehr als einen Takt benötigen, sind in hellgrau markiert. Außerdem ist die Priorität jedes Threads angegeben (nur bei CGMT und SMT nötig), wobei niedrigere Zahlen höherer Priorität entsprechen.
Zeichnen Sie für eine Ausführung mit CGMT, FGMT, SMT jeweils den zeitlichen Verlauf der Abarbeitung auf dem gegebenen System. Richten Sie sich bei der Notation nach Abbildung 1, d.h. tragen Sie die Verarbeitungseinheiten horizontal und die Zeit vertikal auf. Der *Context-Switch-Overhead* sei 1 Takt für CGMT und 0 Takte für die anderen Modelle.
Sie können davon ausgehen, dass die Speicherzugriffe der Threads auf disjunkte Cache-Bereiche abgebildet werden, d.h. dass diese Zugriffe interferenzfrei sind. Außerdem sei der verwendete Scheduler ein ASAP-Scheduler (*As Soon As Possible*). Falls nicht alle Instruktionen aus einem Zeitschritt aus den Abbildungen 1(a) bis 1(c) gleichzeitig gestartet werden können, führt der Scheduler die verbleidenden so bald wie möglich danach aus. Nehmen Sie in diesem Fall außerdem an, dass erst *alle* Instruktionen eines Zeitschritts i aus Abbildung 1 ausgeführt worden sein müssen, bevor Instruktionen aus Schritt $i + 1$ desselben Threads ausgeführt werden können.