Praktikum "Technische Grundlagen Medieninformatik"

Hochschule Osnabrück
Fakultät Ingenieurwissenschaften und Informatik
Labor für Digital- und Mikroprozessortechnik
Prof. Dr.-Ing. W. Gehrke

SS19

Versuch 5: Cache

In diesem Versuch wird die Funktionsweise von Caches behandelt. Durch das Studium des Verhaltens von Caches mit unterschiedlichen Parametern wird das Verständnis für das Cache-Verhalten vertieft.

Vorbereitung (zuhause!)

Eine CPU greift nach dem Start des Programms auf die in der nachfolgenden Tabelle angegebenen Adressen zu. Die Cache-Parameter (Blockgröße, Blockanzahl und Assoziativität) sollen anhand dieses Beispiels untersucht werden.

Nr.	Zugriff auf Adresse
1	0x1001 2000
2	0x1001 2020
3	0x1001 3138
4	0x1001 3150
5	0x1001 2004
6	0x1001 1028
7	0x1001 1420
8	0x1001 202C
9	0x1001 1020
10	0x1001 3108
11	0x1001 2024
12	0x1001 1030
13	0x1001 1410
14	0x1001 222C
15	0x1001 3120

Analysieren und dokumentieren Sie vor Versuchsbeginn zuhause das Verhalten (= welche Zugriffe führen auf Cache-Misses) von Caches mit den ersten drei der nachfolgenden Cache-Szenarios. Die Analyse soll "per Hand", also ohne Unterstützung durch PC-Programme o.ä. erfolgen.

Szenario	Blockgröße[1]	Cachegröße	Assoziativität	Ersetzungstrategie[2]
1	4 Worte	64 Bytes	Direct-Mapped	-
2	4 Worte	64 Bytes	Fully Associative	LRU
3	4 Worte	64 Bytes	2-Way-Set-Associative	LRU
4	64 Worte	2048 Bytes	2-Way-Set-Associative	LRU
5	64 Worte	2048 Bytes	4-Way-Set-Associative	LRU
6	64 Worte	2048 Bytes	Direct-Mapped	-

[1] 1 Wort = 4 Bytes
[2] Bei einem „direct-mapped“ Cache gibt es keine Ersetzungsstrategie. Wählen Sie im MARS-Simulator dennoch „LRU“

Aufgabe 1: Analyse des Cache-Verhaltens mit dem MIPS-Simulator MARS

Erstellen Sie in MARS ein MIPS-Programm, das wie zuvor beschrieben auf den Speicher zugreift.

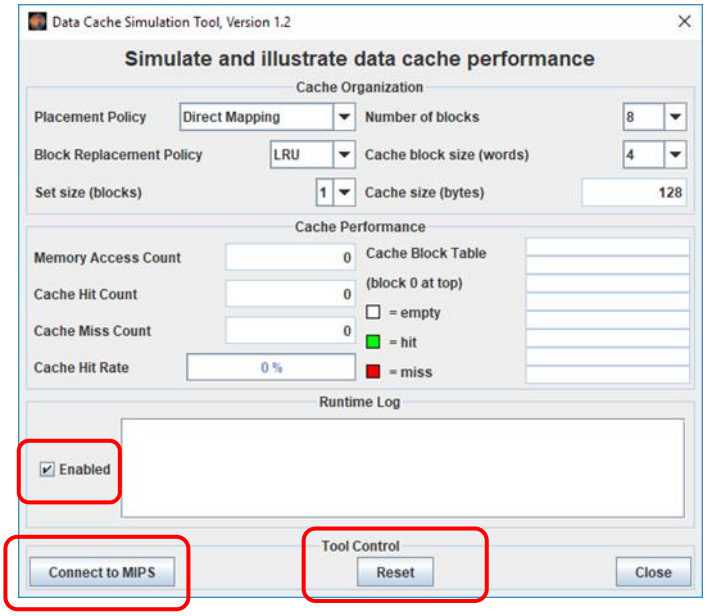
Hinweis: Die ersten Zeilen Ihres Programms könnten lauten

```
lui $t0,0x1001
lw  $t1,0x2000($t0)
```

Übersetzen Sie Ihr Programm und aktivieren Sie den Cache-Simulator unter *Tools -> Data Cache Simulator*. Klicken Sie auf "*Connect to MIPS*" und führen Sie Ihr Programm mit den Parametern der einzelnen Szenarien aus. Vergleichen Sie Ihre Lösung, die Sie zuhause erstellt haben, mit den Ergebnissen des Simulators. Können Sie eventuelle Unstimmigkeiten erklären?

Simulieren Sie alle in der Tabelle angegebenen Szenarien, also auch die Szenarien, die Sie nicht zuhause vorbereitet haben. Tauschen Sie sich über Ihre Erkenntnisse in Ihrer Arbeitsgruppe aus.

Hinweise:
1. Um die Statistik zurückzusetzen klicken Sie vor jedem Durchlauf den Button "Reset".
2. Der Runtime-Log-Bereich ist ein wenig klein geraten. Sie können den Inhalt dieses Bereichs selektieren und in einen Editor kopieren.



Aufgabe 2: MIPS-Programm mit definiertem Cache-Verhalten

Erstellen Sie in MARS ein MIPS-Programm, das insgesamt 10 Speicherzugriffe ausführt und im Fall eines Direct-Mapped Caches genau 5 Cache-Misses erzeugt. Auf jede Adresse soll maximal einmal zugegriffen werden. Der Cache besitzt 4 Blöcke mit einer Größe von 4 Worten.

Überprüfen Sie Ihr Programm und führen Sie es Ihrem Versuchsbetreuer vor.

Aufgabe 3: Arrayzugriffe

Ein Cache ist für den Programmierer nicht sichtbar und deshalb muss man sich als Hochsprachen-Programmierer auch nicht um den Cache kümmern? Na, das wollen wir doch mal sehen!

Wir betrachten ein einfaches Code-Fragment, das die Werte eines zweidimensionalen Arrays setzt[3].

```
int size = n;
int data[size][size];
int value = 0;
for (int col = 0; col < size; col++) {
   for (int row = 0; row < size; row++) {
      data[row][col] = value;
      value++;
   }
}
```

Ein entsprechendes MIPS-Programm finden Sie in den Praktikumsdateien (*row-column.asm*).

1. Öffnen Sie das Programm im MARS-Simulator und aktivieren Sie die Tools "Memory Reference Visualization" und "Data Cache Simulator". Setzen im Cache Simulator die folgenden Parameter: Direct Mapping, Number of Blocks = 4, Cache block size = 4.

i) Um das Speicher-Zugriffsmuster zu verstehen, führen Sie das Programm in Einzelschritten aus. Wie wird auf den Speicher zugegriffen? Ist das im Hinblick auf die Cache-Performance eher gut oder eher schlecht?

ii) Führen Sie das Programm für die *size* = 4,8,16 aus und notieren Sie sich die Anzahl der Speicherzugriffe und die Anzahl der Cache-Hits.

2. Speichern Sie das Programm unter dem Namen "row-major.asm" und vertauschen Sie die Reihenfolge row- und der column-Schleife in dem MIPS-Code (vgl. Hochsprachencode unten). Wiederholen Sie die Schritte i) und ii) und diskutieren Sie Ihre Erkenntnisse aus dieser Aufgabe mit Ihrem Versuchsbetreuer.

```
…
for (int row = 0; row < size; row++) {
   for (int col = 0; col < size; col++) {
      data[row][col] = value;
      value++;
   }
}
```

[3] Dies ist ein sehr einfaches und vergleichsweise übersichtliches Beispiel. In der Praxis werden Ihnen komplexere Beispiele begegnen: Zum Beispiel bei Filterung von Bildern.