Karlsruhe Institute of Technology
Institut für Anthropomatik und Robotik (IAR)

Rechnerorganisation im WS 2017/18

Musterlösungen zum 09. Übungsblatt

Prof. Dr. Wolfgang Karl
Haid-und-Neu-Str. 7

Dr.-Ing. Ömer Terlemez
Adenauerring 2, Geb. 50.20

Email: ti@ira.uka.de
Web: http://ti.ira.uka.de

Lösung 1 (8 Punkte)

1. Die Größe des Caches kann berechnet werden. 2 P.

   Datenblock:

   Anzahl an Sätzen: $\frac{32\,Bit}{8\,Bit} = 4$

   $\Rightarrow$ Satzindex: 2 Bit

   $\Rightarrow$ Offset: $32\,Bit - 20\,Bit - 2\,Bit = 10\,Bit$

   Größe einer Cachezeile: $2^{10}\,Byte = 1024\,Byte$

   Bei 32 Cachezeilen belegen die im Cache gespeicherten Daten dann 32 KiloByte.

   Gesamter Cache:

   Um die Größe des gesamten Caches zu berechnen müssen pro Cache-Zeile noch die Breite des Tags und die Anzahl der Statusbits dazugezählt werden.

   Das sind: $32 * (20Bit + 2Bit) = 32 * 22Bit = 704Bit = 88Byte$

   Gesamtgröße: $32KByte + 88Byte = 32856Byte$

2. Die Größe eines Cacheblocks kann auf Basis der angegebenen Daten nicht berechnet werden. 2 P.

3. Die Assoziativität des Caches kann berechnet werden. 2 P.

   Offset bei einer Blockgröße von 64 Byte: 6 Bit

   Die Bits des Satzindex ergeben sich zu: $32\,Bit - 26\,Bit - 6\,Bit = 0\,Bit$

   Es gibt somit nur einen Satz und es handelt sich somit um einen vollassoziativen Cache.

   Die Assoziativität entspricht damit der Anzahl an Cachezeilen. Da es $\frac{512\,KByte}{64\,Byte} = 8K$ Cachezeilen gibt, ist die Assoziativität also 8192.

4. Bei einem Cache Hit kann bei einem Cache mit Rückschreibeverfahren niemals ein Schreibzugriff ausgelöst werden, da nur Cache Misses zu einem Schreibzugriff führen können (falls das Dirty-Bit der verdrängten Cachezeile gesetzt ist). 2 P.

   Daher liegt die obere Schranke für die Anzahl der tatsächlich durchgeführten schreibenden Speicherzugriffe bei 2.

   Eine untere Schranke größer Null kann nicht angegeben werden.

Lösung 2 (12 Punkte)

- Unterteilungen 3 P.
  - AV: 28 Bit Tag, 4 Bit Offset
  - A2: 27 Bit Tag, 1 Bit Satzindex (SI), 4 Bit Offset
  - DM: 26 Bit Tag, 2 Bit Zeilenindex (ZI), 4 Bit Offset
- Siehe Tabelle: 9 P.

Adresse	AV	A2	DM	SI	ZI
0x25	-	-	-	0	2
0x3A	-	-	-	1	3
0x12	-	-	-	1	1
0x74	-	-	-	1	3
0x36	X	-	-	1	3
0x08	-	-	-	0	0
0x09	X	X	X	0	0
0x16	X	-	X	1	1
0x28	-	X	X	0	2
0x52	-	-	-	1	1
0x22	X	X	X	0	2
0x11	X	X	-	1	1

Lösung 3 (5 Punkte)

Die Programmschleife sieht in C-Notation etwa wie folgt aus:

```
for (int sum = 0, int i = 0; i < 128; ++i)
    sum += a[i];
```

Die Anweisung lässt sich in die folgenden Einzeloperationen zerlegen:

		a)		b)	
		Miss	Hit	Miss	Hit
	sum = 0	1		1	
	i = 0	1		1	
loop:	read i		128		
	i < 128 ?				
	exit, if false				
	read a[i]	16	112	16	112
	read sum		128		
	compute sum+a[i]				
	write sum		128		
	read i		128		
	compute i+1				
	write i		128		
	jump to loop				
		18	752	18	112
		2,3%	**97,7%**	13,8%	**86,2%**
c)		10	760	10	120
		1,3%	**98,7%**	7,7%	**92,3%**

Lösung 4 (6 Punkte)

1. Es werden je 333 Lade- und Speicherbefehle ausgeführt, d.h. insgesamt 666 Befehle, die potentiell auf den Speicher zugreifen. 2 P.

   Bei der Breite einer Cachezeile von 16 Byte erzeugt das Programmstück $\lceil \frac{334}{4} \rceil = 84$ Compulsory Misses beim Zugriff auf das (im Speicher an einem Vielfachen von 16 ausgerichteten) Array. Es liegen somit $666 - 84 = 582$ Cache Hits vor.

2. Die Hit-Rate beträgt $\frac{582}{666} = 87,387\%$. 2 P.

   Die mittlere Zugriffszeit beträgt $0,87387 \cdot 2 + (1 - 0,87387) \cdot 10 = 3,009$ Taktzyklen.

3. Zwei Cachezeilen genügen. 2 P.

   Eine Zeile genügt nicht, da sonst alle vier Arrayelemente zusätzlich zu dem einen Compulsory Miss zwei Capacity Misses auftreten.

Lösung 5 (2 Punkte)

	wahr	falsch
Bei einem direktabgebildeten Cache ist es nicht notwendig, eine Block-Ersetzungsstrategie festzulegen.	×	
Die Verwendung des Rückschreibverfahrens (*write back policy*) verhindert das Auftreten von Konsistenzproblemen bei Mehrprozessorsystemen.		×
Bei einem satzassoziativen Cache können mehrere Cachezeilen gültig sein und denselben Tag, aber dennoch unterschiedliche Daten enthalten. Bei einem vollassoziativen Cache kann das nicht passieren.	×	
Je höher die Assoziativität eines Cache, desto weniger Komparatoren werden zu seiner Realisierung benötigt. Allerdings steigt damit auch die Auftrittswahrscheinlichkeit für Conflict Misses.		×