

## Aufgabe II-1: Fragen zur Rechnerarchitektur (12 P)

a) Geben Sie an, ob die folgenden Aussagen wahr (W) oder falsch (F) sind:

W F

- 1) Pipelining ist eine Mikroarchitekturtechnik.
- 2) Superskalarität ist eine Architekturtechnik.
- 3) Die Adresse des Speicherwortes zeigt bei einem wortadressierbaren Speicher mit Big-Endian-Anordnung immer auf das höchstwertige Byte.
- 4) Bei dem Befehl *Shift Logical Right* (SLR) wird als oberstes Bit immer eine Null eingefügt.
- 5) Die effektive Adresse ergibt sich bei der indizierten Adressierung als Summe von Indexregister und einem im Befehlswort stehenden Offset.
- 6) Der Sprungzieladdress-Cache ist ein kleiner Cache-Speicher, auf den in der ID-Stufe zugegriffen wird und der mehrere Tupel bestehend aus Sprungadresse, Sprungzieladresse und Vorhersagebits enthält.
- 7) Eine typische Eigenschaft eines CISC-Prozessors ist die hohe Anzahl an universell nutzbaren Registern.
- 8) In der Operandenbereitstellungsphase werden der *Arithmetic Logic Unit* (ALU) die Operanden aus dem Daten-Cache bereitgestellt.
- 9) Durch Forwarding können Datenkonflikte aufgelöst werden, indem spätere (und jeweils unabhängige) Befehle im Programmverlauf vorgezogen werden.
- 10) Bei der Block-Interleaving-Technik werden die Befehle *eines* Kontrollpfades so lange wie möglich direkt hintereinander ausgeführt.

b) Geben Sie an, wie die **Latenz** und der **Durchsatz** einer Pipeline definiert sind.

- c) Im Kurstext haben Sie die drei Befehlsformate (R-Typ, I-Typ und J-Typ) des DLX- bzw. MIPS-Prozessors kennengelernt. Geben Sie für jeden der drei Typen zwei Befehle an, welche diesem Typ zugeordnet werden können. Stellen Sie außerdem für jeden Typen dar, in welche Felder ein 32-Bit-Befehl unterteilt wird und geben Sie deren Bezeichner an. (Diese Aufgabe muss je nach belegter Kursversion **entweder** für den DLX- **oder** den MIPS-Prozessor gelöst werden!)
- d) Geben Sie an, was man unter den Prinzipien der **örtlichen Lokalität** und **zeitlichen Lokalität** versteht.

## Aufgabe II-2: Gleitkommadarstellung (6 P)

Gegeben sei die Dezimalzahl  $Z_{10} = -40,375$ .

- a) Stellen Sie die Zahl  $Z_{10}$  als gebrochene, normalisierte Zahl  $Z_{32}$  im 32-bit-Format des IEEE-754-Standards dar und tragen Sie dazu die entsprechenden Werte für Vorzeichen, verschobenen Exponenten und Mantisse in das folgende Schema ein:

$$Z_{32} = (-1)^{\dots\dots\dots} \cdot 2^{(\dots\dots\dots)_{10}} \cdot (\dots\dots\dots \bullet \dots\dots\dots)_2$$

- b) Tragen Sie die Zahl  $Z_{32}$  in binärer Darstellung in den folgenden Bitrahmen ein und kennzeichnen sowie bezeichnen Sie die unterscheidbaren Bitfelder.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		

Tragen Sie hier die Bezeichnungen der Bitfelder ein

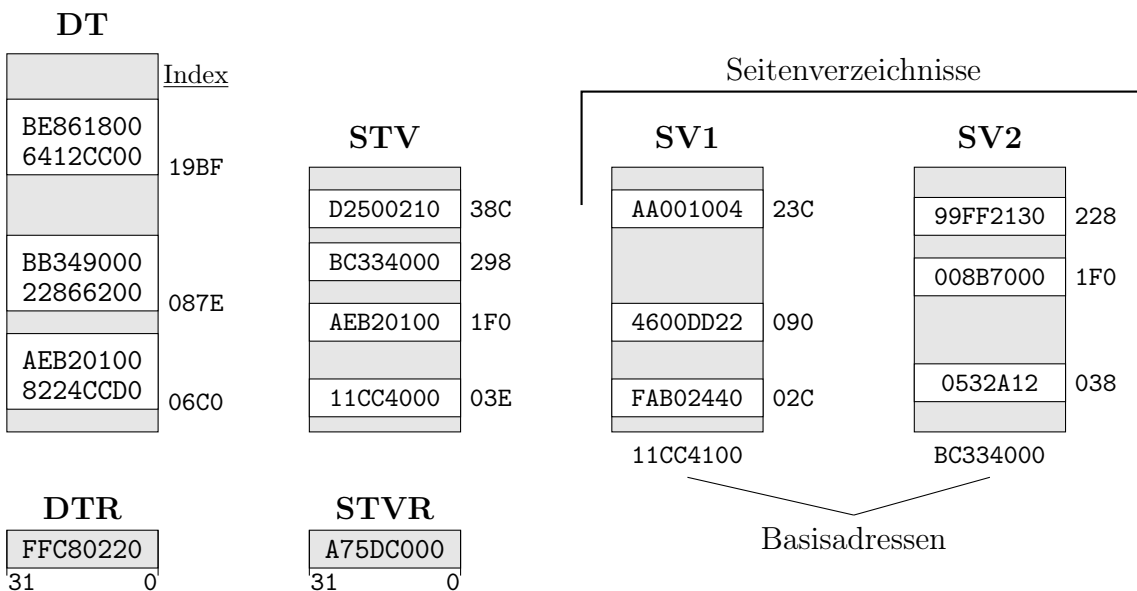
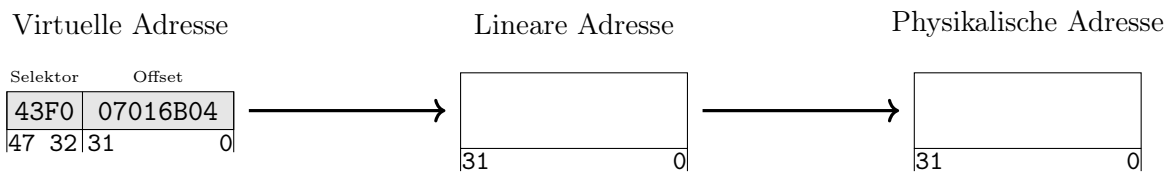
- c) Geben Sie die Zahl  $Z_{32}$  als Hexadezimalzahl  $Z_{16}$  an.

$$Z_{16} = \dots\dots\dots$$

**Hinweis:**  
 Die Indizes  $\dots_2$ ,  $\dots_{10}$ ,  $\dots_{16}$  und  $\dots_{32}$  kennzeichnen jeweils Zahlen im Binär-, Dezimal- sowie Hexadezimal-System bzw. im 32-Bit-IEEE-Format.

## Aufgabe II-3: Virtuelle Speicherverwaltung (20 P)

In dieser Aufgabe wird die virtuelle Speicherverwaltung der x86-Prozessoren betrachtet. Es werden dabei sowohl eine Segmentierung als auch ein Seitenwechselverfahren angewendet. Die Adressbreite beträgt 32 Bit und die Seitengröße 4 kByte. Die unten aufgeführte Abbildung zeigt für einen aktiven Prozess den relevanten Ausschnitt der Speicherbelegung mit Deskriptortabelle (DT), Seitentabellenverzeichnis (STV) und zwei Seitentabellen (SV1 und SV2). Außerdem sind die Belegungen vom Deskriptortabellen-Basisregister (DTR) sowie vom Seitentabellenverzeichnis-Basisregister (STVR) angegeben. Alle Werte der Abbildung sind bis auf die Bitgrenzen im Hexadezimalformat angegeben.



- a) Die oberen 13 Bit des Selektors der virtuellen Adresse werden als Index verwendet, um einen der 8 Byte großen Einträge in der Deskriptortabelle zu selektieren. Geben Sie an, wie viele Einträge die Deskriptortabelle insgesamt enthalten kann und wie groß die Tabelle (in Byte oder kByte) maximal wird.

- b) Die unteren 4 Byte des selektierten Deskriptors ergeben addiert mit dem Offset der virtuellen Adresse die lineare Adresse. Berechnen Sie diese und tragen Sie sie in das oben stehende freie Feld ein.
- c) In der Abbildung sind jeweils die unteren 12 Bit der Adressen sowohl für die STV als auch die SV1/SV2 angegeben. Für den Zugriff auf das STV werden die oberen 20 Bit des STVR mit den oberen 10 Bit der linearen Adresse (31..22) sowie zwei Nullen ( $00_b$ ) konkateniert. Der selektierte Eintrag in dem STV enthält die Basisadresse des auszuwählenden Seitenverzeichnisses. Die nächsten 10 Bit der linearen Adresse (21..12) werden ebenfalls mit zwei Nullen ( $00_b$ ) konkateniert und zur Selektion eines Eintrags im soeben ausgewählten Seitenverzeichnis verwendet. Die physikalische Adresse setzt sich dann aus den oberen 20 Bit des Eintrags im Seitenverzeichnis konkateniert mit den unteren 12 Bit der linearen Adresse zusammen. Bestimmen Sie auf diese Weise die physikalische Adresse und tragen Sie diese in das entsprechende Feld ein.
- d) Erläutern Sie, wie das Seitenwechselverfahren durch ein im Kurs genanntes Vorgehen beschleunigt werden kann. Welche zusätzliche Funktionseinheit wird hierfür benötigt und was speichert diese ab?

- e) Bestimmen Sie die Anfangsadressen für folgende Tabelleneinträge (Nummerierung hier im Dezimalsystem!):

Eintrag Nr.  $12_{10}$  in der DT .....

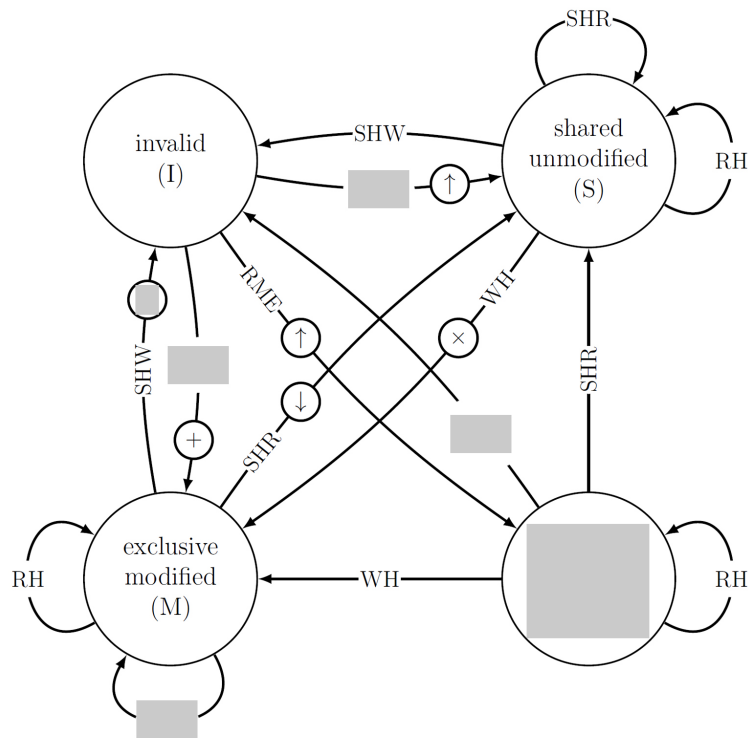
Eintrag Nr.  $64_{10}$  im STV .....

Eintrag Nr.  $16_{10}$  im SV1 .....

## Aufgabe II-4: Cache-Kohärenz-Protokoll (12 P)

a) Geben Sie an, was man unter den Begriffen **Konsistenz** und **Kohärenz** in Bezug auf die Speicherhierarchie versteht.

b) Ergänzen Sie im unten stehenden MESI-Zustands-Diagramm die sieben leeren und grau hinterlegten Felder.



RH	read hit	↓	veränderte Cache-Line zurückschreiben
RMS	read miss, shared	×	
RME	read miss, exclusive	+	lesen einer Cache-Line zum Ändern
WH	write hit	↑	Cache-Line laden
WM	write miss		
SHR	snoop hit on read		
SHW	snoop hit on write <i>oder</i> lesen einer Cache-Line zum Ändern		

- c) Es sei ein System mit zwei Prozessoren gegeben. Jeder der beiden Prozessoren verfügt dabei über einen eigenen, in Von-Neumann-Architektur realisierten L1-Cache. Weitere Cache-Ebenen existieren nicht. Beide Prozessoren teilen sich einen Hauptspeicher und sind an diesen über einen gemeinsam genutzten Speicher-Bus angeschlossen. Zur Gewährleistung der Cache-Kohärenz wird das MESI-Protokoll eingesetzt. Jeder der L1-Caches besitzt vier Cache-Blöcke und ist als vollassoziativer Cache-Speicher mit LRU-Ersetzungsstrategie realisiert. Bei mehreren freien oder ungültigen Cache-Blöcken werden diese von unten an gefüllt. Zur Vereinfachung wird angenommen, dass sowohl die Cache-Blöcke als auch die Cache-Lines durchnummeriert sind. Die genaue Adressaufteilung in Tag/Index/Byteauswahl wird dabei in dieser Aufgabe nicht betrachtet.

Es finden nacheinander folgende Speicherzugriffe statt:

Prozessor-Nr.	Cache-Line	Lesen/Schreiben
1	1	Schreiben
1	4	Lesen
2	5	Schreiben
2	2	Lesen
1	2	Lesen
1	5	Schreiben
2	4	Schreiben

Geben Sie in der folgenden Tabelle den zeitlichen Verlauf der Cache-Inhalte an, der sich durch die oben angegebenen Speicherzugriffe sowie die Verwendung des MESI-Protokolls ergibt. Hierfür sollen in den Einträgen sowohl die Nummer der aktuell enthaltenen Cache-Line als auch die Abkürzung des Zustandes entsprechend des MESI-Protokolls eingetragen werden. Die initialen Belegungen und die Belegung nach dem ersten Zugriff sind als Beispiel vorgegeben.



Prozessor 1				Prozessor 2			
Block 1	Block 2	Block 3	Block 4	Block 1	Block 2	Block 3	Block 4
E/4	E/6	S/5	I/-	E/1	S/5	I/-	I/-
E/4	E/6	S/5	M/1	I/-	S/5	I/-	I/-

**ENDE**