1. Virtueller Speicher

LRU:			**LIFO:**		**FIFO:**	
referenziert	Cache (links am längsten im Cache)	Fehler	Cache (links am längsten im Cache)	Fehler		Fehler
-	9415		9415		9415	
3	4153	x	9413	x	4153	x
1	4531	x	9413		4153	
2	5312	x	9412	x	1532	x
4	3124	x	9412		5324	x
2	3142		9412		5324	
4	3124		9412		5324	
7	1247	x	9417	x	3247	x
6	2476	x	9416	x	2476	x
3	4763	x	9413	x	4763	x
4	7634		9413		4763	
1	6341	x	9413		7631	x
9	3419	x	9413		6319	x

Bei LRU und FIFO führt z.B: die Folge 3941539415 ... zu einem Maximum an Seitenfehlern.
Bei LIFO ist das z.B. die Folge 353535...

2. Caching

a) Durch die Aufteilung spart man an Komperatoren. Den durch den fest verdrahteten Index kann sofort auf den richtigen Satz zugegriffen werden und es muss nur noch mit dem Komperator der Tag verglichen werden.

b) Wie bei dir

c) Man strebt keine Konsistenz an, da dies unnötig die Datenbusse belastet und außerdem einen enormen Zeitaufwand mit sich bringt.

d) Der Vorteil von physikalisch adressierten Adressräumen ist, dass die Länge des Tages weniger Bits benötigt, da im allgemeinen dieser Adressraum viel kleiner ist.. Bei virtuellem Cache ist es hingegen günstig, dass die MMU bei einem Treffer nicht beansprucht wird. (siehe Folie 4.115)

e) Wenn ein Prozess gewechselt wird, dann ist quasi ein virtueller Cache aufgeschmissen, da der neue Prozess einen ganz anderen logischen Adressraum hat. Es müssen also alle Daten hier erst eingelesen werden bzw. die Adressen durch die MMU berechnet werden.
Bei einem physikalisch Cache macht ein Prozesswechel nichts aus. Die Adressen

ändern sich ja nicht. Die MMU liegt hier ja noch vor dem physikalischen Raum und hat bereits die logischen Adressen umgewandelt, wenn auf den Cache zugegriffen wird.

3. Größe der Tags

(Annahme, jedes Wort besteht aus 4Byte)

Direct mapped:

Hat 4k (=4096) Sätze, die entspricht der Anzahl der Blöcke. Die Aufteilung der Adressierung sähe so aus

16 TAG | 12 Satz | 2 Wort | 2 Datum

2-way:

Hat 4k (=4096) / 2 = 2048 Sätze. In jedem Satz sind zwei Blöcke. Die Aufteilung der Adressierung sähe so aus

17 TAG | 11 Satz | 2 Wort | 2 Datum

4-way:

Hat 4k (=4096) / 4 = 1024 Sätze. In jedem Satz sind vier Blöcke. Die Aufteilung der Adressierung sähe so aus

18 TAG | 10 Satz | 2 Wort | 2 Datum

vollassoziativ:

Hat genau einen Satz, in dem alle Blöcke enthalten sind. Die Aufteilung der Adressierung sähe so aus

28 TAG | 2 Wort | 2 Datum

4. Cache-Zugriff

	Miss	Hit
`write i = 0`	1	
`write j = 0`	1	
`loop: read j`		128
`if (j >= 128) exit else`		
`read g[j]`	4	4* 31 = 124
`read i`		128
`compute i + g[j]`	-	-
`write i`		128
`read j`		128

`compute j + 1`	-	-
`write j`		128
`jump to loop`		-

Am Anfang ist der Cache leer, darum finden wir i und j am Anfang dort noch nicht. Dann wird allerdings Platz gesichert und ab da an wird das Datum gefunden und nur im Wert geändert. Dies ist allerdings kein Miss.
Bei dem ersten Zugriff aufs Array wird gleich eine ganze Zeile mitgeladen. Da ein Wert im Array aus 16bit = 2Byte besteht und in eine Zeile 64 Byte passen, werden immer 32 Zahlen geladen. Daher haben wir bei der 1. Zahl ein Miss, und laden gleich die nächsten 31 mit. Dann ist bei der 33 wieder ein Miss usw. Da wir 128 Zahlen laden haben wir also 4 mal ein Miss.

Insgesamt handelt es sich also um 6 Miss und 764 Hits.

Die Miss-Quote ergibt sich als 6 / 770 = 0,00779 = 0,78 %

5. (Aufgabe ohne Namen ;-))

Jede Zeile fasst 8Byte = 64 Bit. Ein Datum ist 32-Bit groß (=4Byte), also haben wir pro Block 2 Wörter.

a) Es gibt 32 Cache-Zeilen.
   Und zwar gilt Gesamtgröße des Caches / Anzahl der Bytes pro Zeile.
   Also 256Byte / 8 Byte = 32

b) Die Hauptspeicheradresse ist 24Bit lang und teilt sich wie folgt auf:
   Direct-mapped:

16 TAG | 5 Block | 1 Wort | 2 Datum

   vollassoziativ:

21 TAG | 1 Wort | 2 Datum

   4-way:

c) | 18 TAG \| 3 Block \| 1 Wort \| 2 Datum |
   |---|

   Für direct-mapped wird genau ein Vergleicher benötigt. Die Tag-Länge ist hier 16bit. Bei vollassiziativem Cache haben wir für jede Zeile einen Vergleicher, also insgesamt 32. Der Tag ist hier auch Länger, nämlich 21bit groß. Bei dem 4-way-set benötigt man vier Vergleicher. Der Tag ist hier 18Bit groß.

6. Noch eine ohne Namen

a) Der direct-mapped Cache hat 265 (Gesamtkapazität) / 4 (Anzahl Wörter pro Zeile) = 64 Sätze. In jedem Satz ist auch nur eine Zeile enthalten.
   Bei dem 4-way-set sind je 4 Zeilen zu einem Satz zusammengefasst. Darum gibt es 16 Sätze à 4 Zeilen.