Aksesstid (20)

Vi har en datamaskinarkitektur der prosessoren har 2 nivå med cache og et fysisk minne.

Oppslag i det fysiske minnet tar 100 klokkesykler. Oppslag i nivå 2 cache tar 20 klokkesykler. Oppslag i cachen nærmest prosessoren (nivå 1) tar en klokkesykel.

Vi har en algoritme med mange minneaksesser. Oppslagene i cachen nærmest prosessoren har en treffrate på 80%. Oppslagene i nivå 2 cache har en treffrate på 90%. Oppslag i hovedminne gir alltid treff.

Hva blir gjennomsnittlig aksesstid for minneoppslagene? 6,4 eller 7 klokkesykler

Vi får 2 ulike verdier her pga ulik tolkning av oppslagstiden. Regner vi med at oppslag i høyere nivå cache er inkludert i lavere nivåer får vi følgende utregning:

1*0,8+ 0,2(0,9*20 + 0,1*100)= 0,8+0,2(18 + 10) = 0,8 + 0,2*28 = 0,8 + 5,6 = 6,4

Dersom vi regner med at oppslag i høyere nivå ikke er inkludert i den oppgitte cachen får vi følgende utregning:

1+0,2∗20+0,2∗0,1∗100=7, alternativt 1∗0,8+0,2∗0,9∗21+0,2∗0,1∗121=7

Vi kan kjøre algoritmen på en nyere prosessor med kun ett nivå cache. Ettnivås cachen har en høyere treffrate, og en aksesstid på en klokkesykel. Hovedminnet her bruker også 100 klokkesykler. Klokkefrekvensen er den samme.

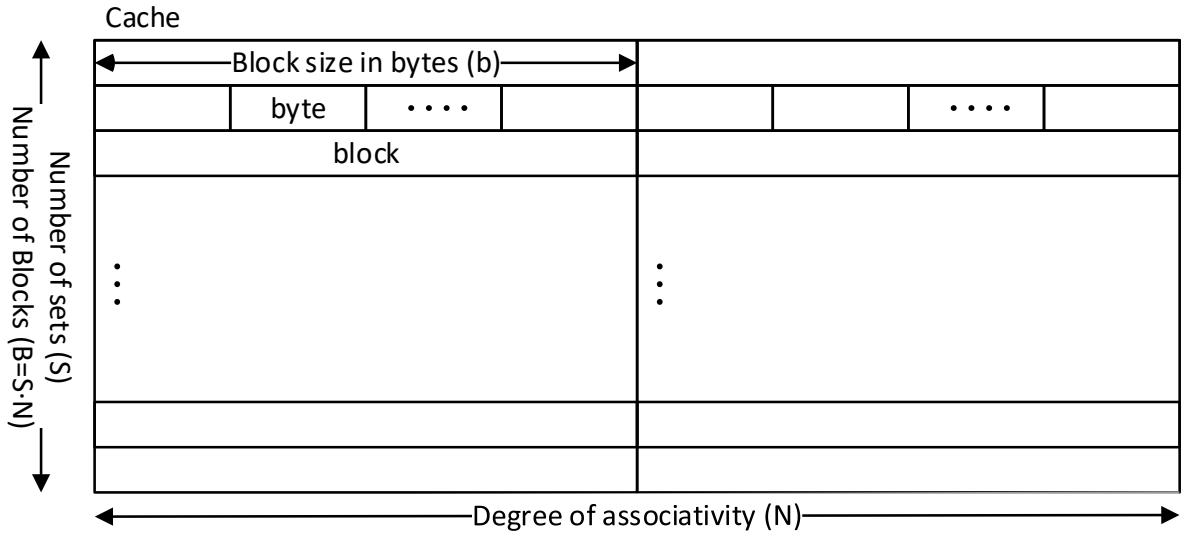
Hvor stor må treffraten for cachen i det nye prosessorsystemet være for at det skal lønne seg å benytte dette fremfor det forrige?

1* (TR) + (1-TR)*100 < 6,4
TR + 100 – 100TR < 6,4
100-99TR < 6,4
-99TR < 6,4-100 | :-99
TR > 93,6/99
TR > 94,55

Alternativt:
1*TR + (1-TR)*101 < 7, Hvilket gir
TR + 101 – 101TR < 7 =>
-100TR < 94
TR > 94%

Merk: Det er grensen her man er ute etter. *For et hvert reelt tall som er større enn grensen, så kan man finne et imellom som er mindre, men som fremdeles er lønnsomt, og ergo et bedre svar. Derfor bør man ikke legge til et lite tall som f.eks. 0,1% for oppgavens del.*

Cache (21)



Figur 1 Generisk cache

1: Cache oppbygning

Figuren over viser en generisk cache. Til et fysisk minne på 512MB bruker vi en direkte-mappet cache (N=1) med blokkstørrelse på fire ord, fire byte per ord, og kapasitet på 32kB. Hver byte i minnet har en egen adresse.

a) Hvor mange bit trenger adresser i det fysiske minnet? [29]
b) Hvor mange set vil cachen ha? [2000-2050]
c) Hvor mange bit trenger adresse-tagene til cachen? [14]

Tips: Vi bruker[1] **1K = 1024 = 2^{10}**, 1M = 1K[2].

Løsning:

a) Adresse: 512MB = $2^{10}*2^{10}*2^{9}$ byte = 2^{29} byte (29 bit adresse)

b) Kapasitet til Cache: 32kB = $2^{5}*2^{10}$ = 2^{15} byte
Blokk størrelse: 16 byte = 2^{4}
Antall set: Kapasitet / Blokkstørrelse = 2^{15} / 2^{4} = 2^{11} =2048

c) Tag holder det vi må ta vare på for å vite om oppslaget er gyldig:
bit i minneadresse – bit i cacheadresse =
29bit-15bit = 14 bit

[1] SI enheten Kilo betyr 1000, men siden adresserom er multiple av 2, bruker vi 1024 mht. minne. Dette er normal praksis, selv om korrekt bruk av SI enhetene også kan forekomme.

Eks: Når vi skriver 64KB mener vi normalt 2^{16} Byte som er 65536 Byte.
Å skrive 66KB for 2^{16} er strengt tatt korrekt avrunding for 2 gjeldende siffer og bruke SI-enheter, men det er upraktisk å gjøre dette når vi i praksis alltid jobber med multipler av 2.

Minneoppslag (22)

I denne oppgaven skal du oppgi tallsvar. Tallsvarene kontrolleres med en presisjon på 2 desimaler. Det er greit å benytte flere desimaler, men ikke nødvendig ved korrekt avrunding.

Vi har en direkte-mappet cache med kapasitet på 16 ord og en blokkstørrelse på 4 ord og en ordstørrelse på 4 byte.

Et program gjør minneaksess fra følgende adresser i sekvens.

```
0x40, 0x44, 0x48, 0x84, 0x60, 0x64, 0x40, 0x44, 0x48, 0x84, 0x60, 0x64
  tM,    H,    H, t/kM,   tM,    H,   kM,    H,    H,   kM,    H,    H
```

Hvor mange av disse minneaksessene vil resultere i bom (miss) ved oppslag i cachen? [5]

Hva blir treffraten for disse minneaksessene? 7/12 = 0,58 [0,58-0,59]

Vi sammenligner med en to-veis set-assosiative cache med samme kapasitet og en blokkstørrelse på ett ord. Den nye cachen skifter ut blokken det er lengst siden ble brukt.

Hva blir missraten om vi kjører den samme sekvensen med minneoppslag i den nye cachen?
9/12 = 0,75 [0,745-0,755]

```
0x40, 0x44, 0x48, 0x84, 0x60, 0x64, 0x40, 0x44, 0x48, 0x84, 0x60, 0x64
  tm,   tm,   tm, t/km,   tm, t/km,    H,   km,    H,   km,    H,   km
```

Samme område:
x40, x60
x44, x84, x64

Med to-veis cache så vil bare x44, x84, x64 føre til konfliktsbom. Vi får flere tvungne bom ved start pga redusert blokkstørrelse.

Hva konvergerer **miss-raten** mot dersom sekvensen gjentas uendelig mange ganger i når programmet kjøres?

3/6 = 0,5 [0,495-0,505]

Tips: Tegn opp cachene og få oversikt over hvordan adressene fordeler seg.

Merk: Desimaler er siffer etter komma, mens gjeldende siffer er siffer etter ledende null.
Eks 0,067 har tre desimaler, og to gjeldende siffer. Skriver vi $67*10^{-3}$, så har vi fremdeles to gjeldende siffer, men ingen desimaler. Skriver vi $67,00*10^{-3}$ har vi fire gjeldende siffer og vi benytter to desimaler.
Per 2022 har eksamensverktøyet inspera ikke muligheter til å skrive og tolke eksponenter eller SI-enheter, derfor angir man gjerne eksponent, SI-prefiks, prosent eller promille i oppgaveteksten/ ved svarboksen. Både komma og punktum blir lest som desimalskilletegn.