# Oppgave 1:

En parallell beregning på et $n$-prosessorsystem kan karakteriseres ved de to størrelsene $P(n)$, som er totalt antall enhets-operasjoner som skal utføres, og $T(n)$, som er den totale utførelsestiden (i antall tidsenheter) på systemet. For en seriell beregning på en uniprosessor ($n = 1$), kan en skrive $T(1) = P(1)$, fordi hver enhetsoperasjon trenger én tidsenhet for å utføres.

Generelt gjelder at dersom beregningen inneholder noe parallellitet, er $T(n) < P(n)$ for $n \geq 2$.

Lee (1980) har foreslått fem mål på ytelsen til en parallell beregning i forhold til en tilsvarende seriell :

$$\text{Hastighetsøkningen} \qquad S(n) = \frac{T(1)}{T(n)}$$

$$\text{Effektiviteten} \qquad E(n) = \frac{T(1)}{n \cdot T(n)}$$

$$\text{Redundansen} \qquad R(n) = \frac{P(n)}{P(1)}$$

$$\text{Utnyttelsen} \qquad U(n) = \frac{P(n)}{n \cdot T(n)}$$

$$\text{Kvaliteten} \qquad Q(n) = \frac{T(1)^3}{n \cdot T(n)^2 \cdot P(n)} \quad .$$

a) Finn fysiske betydninger av de ovennevnte ytelsesmål.

b) Vis at de følgende relasjoner holder i alle mulige sammenlikninger mellom parallelle og serielle beregninger :

(a)
$$1 \leq S(n) \leq n$$

(b)
$$\frac{1}{n} \leq E(n) \leq 1$$

(c)
$$U(n) = R(n) \cdot E(n)$$

(d)
$$Q(n) = \frac{S(n) \cdot E(n)}{R(n)}$$

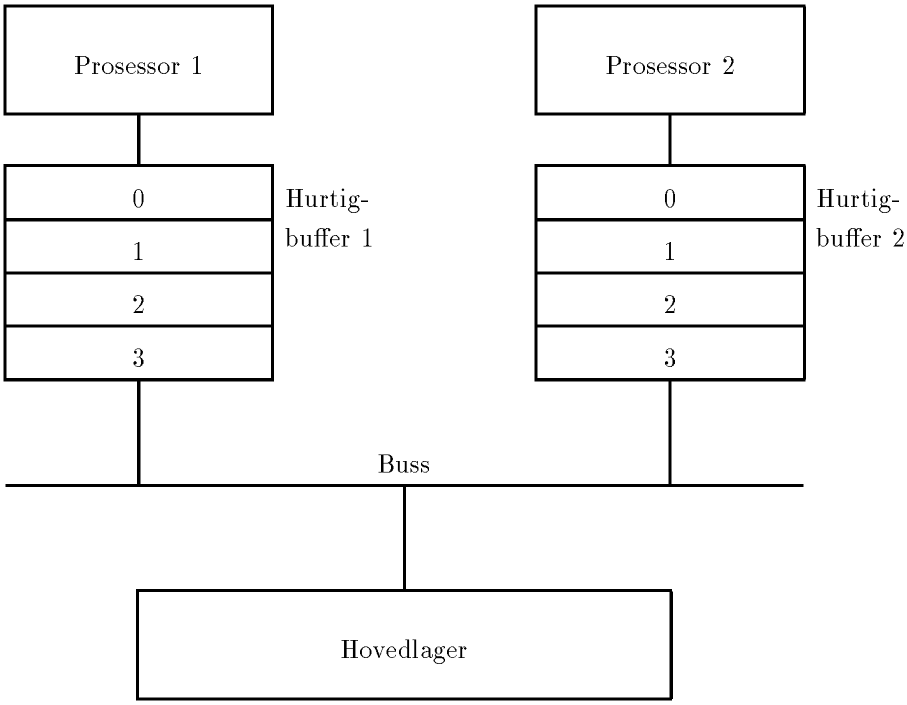
(e)
$$\frac{1}{n} \leq E(n) \leq U(n) \leq 1$$

(f)
$$1 \leq R(n) \leq \frac{1}{E(n)} \leq n$$

(g)
$$Q(n) \leq S(n) \leq n$$

## Oppgave 2:

Anta et to-prosessor system (P1 og P2) som benytter private hurtigbuffere med tilbakeskriving. Se figuren under. Systemet har delt minne (hovedlager) og er koblet sammen vha. en felles buss. Hver hurtigbuffer har fire blokk-rammer merket 0, 1, 2 og 3. Det delte minnet er delt opp i 8 hurtigbuffer




blokker 0, 1, ... 7. For å opprettholde hurtigbuffer-koherens, benytter systemet en 3-tilstands (RO, RW, og invalid) snuse protokoll basert på skriv-ugyldiggjør strategien som er beskrevet i Fig. 7.15b i læreboka [Hwang93].

Anta at den samme klokken driver prosessorene og minne-bussen. I løpet av en sykel kan begge prosessorene spørre om aksess for bussen (bus request). Hvis det oppstår en simultan forespørsel fra de to prosessorene, vil P1 bli gitt aksess, mens P2 må vente.

I alle tilfelle vil bussen tillate bare én aksess pr. transaksjonssykel. Når en buss-forespørsel først er gitt, må transaksjonen fullføres før neste forespørsel tillates. Så lenge det ikke er buss-konflikt (Eng: contention), vil minne-aksess fra hver prosessor trenge en eller to sykler for å fullføres.

- Lese-treff (read-hit) i hurtigbuffer trenger én sykel og ikke buss-aksess.
- Lese-bom (read-miss) i hurtigbuffer trenger to sykler (uten buss-konflikt): én for blokk-henting og én for CPU lesing fra hurtigbuffer.
- Skrive-treff (write-hit) trenger én sykel for CPU skriving og buss-invalidering samtidig.
- Skrive-bom (write-miss) trenger to sykler: én for blokk-henting og buss-invalidering, og én for CPU skriving.
- Utskifting av en modifisert (dirty) blokk trenger en sykel for å oppdatere minnet via bussen.
- Hvis det snakk om buss-konflikt, trengs en ekstra sykel for buss-arbitrering i alle tilfeller ovenfor, untatt for lese-treff.

a) Vis hvordan åtte hurtigbuffer blokker avbildes til fire hurtigbuffer blokker vha. en direkte-avbildet hurtigbuffer organisering. Vi også hvordan åtte hurtigbuffer blokker avbildes ved bruk av en to-veis sett-asossiativ hurtigbuffer organisering.

b) Anta følgende asynkrone sekvenser av minne-aksess hendelser, hvor *tall er skriving og resten er lesing.*

   *Prosessor #1: 0,* 0, 0, *1, 1, 4, 3, 3, 5,* 5, 5
   Prosessor #2: 2, *2, 0, 0,* 7, 5, 5, 5, *7, 7, 0*

   *Spor (Eng: trace) utførelsen av de to sekvensene på de to prosessorene. Begge hurtigbufferene er tomme fra starten av. Anta en direkte-avbildet organisering av begge hurtigbuffere. Vis tilstanden (RO eller RW) til hver valid blokk og merk hurtigbuffer-bom og buss-tilstand (busy/idle) for blokk-sporingen for hver sykel. Anta at prosessorene starter samtidig, med første aksess i sykel 1. Beregn treff-raten for hurtigbuffer 1 og 2.*

*c) Utfør beregningene i a) også for en to-veis sett-assosiativ hurtigbuffer-organisering med en LRU erstatnings-strategi.*