Informasjon om kandidat/gruppe

Kurs: IKT 102

Øvingsoppgave: Minne Øving 3 for Opsys

Dato: 01.11/2019

Kandidat/gruppe: Pål Karlsen i samarbeid med Ola Grytting

Oppgave 1

Hva er forskjellen på en virtuell adresse og en fysisk adresse i minnehåndtering?

En virtuell adresse er generert av CPUen når et program kjører. Virtuelle adresse eksisterer ikke fysisk og heter derfor virtuell adresse. Denne adressen er brukt som en referanse for å få tilgang til da fysiska minne som er lokalisert i CPUen.

Fysisk adresse identifiseres som en fysisk lokasjon av nødvendige dataer som er i minne. brukeren har aldri noe med fysiske adresser å gjere, men har tilgang til det gjennom logical adresser. Bruk programmet genererer logiske adresser og trur at programmet kjører i desse logiske adressene, men programmet trenger fysiske minner for å kunne kjøre, derfor så må logiske adresser bli oversatt til fysiske adresser av MMUen før dei er brukt.

logiske adresser/virtuelle adresser blir generert av CPUen når et program kjører, medan fysiske adresser refererer til lokasjoner i minne enheten. [1]

Oppgave 2

Forklar en av metodene/policyene som kan benyttes for å fjerne en side fra minnet(sideskiftalgoritmer)?

Optimale sideskiftealgoritme

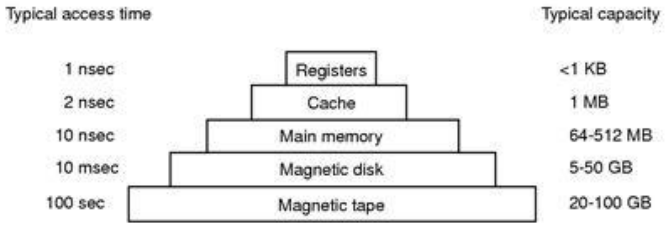
Denne sideskiftealgoritmen virker ved at den ser på hvilke sider som kommer til å bli brukt i fremtiden og skifter ut den den ser kommer til å bli minst brukt.

Dette er den mest optimale formen for en sideskiftealgoritme, men utrolig urealistisk ettersom at en kan ikke få et operativsystem til å vite hvilke sider som vil bli bedt om i fremtiden. Derfor blir denne typen av sideskiftealgoritme brukt som en standard vi måler andre sideskiftealgoritmer oppimot.

[2] [3]

Oppgave 3

Forklar hvordan minnehierarkiet virker(kapittel 1.3.2 i boka)?



minnehierarkiet er et separert hierarki som er basert fra topp til bunn fra raskes til seinest, minst til størst. Når en skal kjøre et program så blir det dytte opp i hierarkiet fram til det er like rask som CPUen, medan når det blir lagret blir det dytte ned helt til det når harddisken.

Top laget er rask, liten kapasitet og koster mer per bit en de lavere laga ofte med en faktor på en billion eller mer. top laget består av register som er internt til CPUen. dei er laga av det same som CPUen og er derfor like rask som CPUen.

neste laget er ‘’Cache’’ minnet, som er for det meste kontrollert av hardware. hoved minne er delt opp i cache lines, og er vanligvis delt opp i 64 bytes. Cache lines er som ofte holdt i en høyhastighets cache som er i eller nær CPUen. Cache minne er ofte begrensa i størrelse på grunn av kostnader. Noen maskiner har 2 eller 3 level av cache, hver er seinare og større enn den før.

etter cachen så er hoved minne eller RAM som det ofte er kalt i dag. detter er et relativ stort minne alt fra 4 og opp mot 128 GB. alle program må innom RAMen før datamaskinen bruke dem eller manipulere dem. I/O må også innom RAM for å kunne bli brukt.

så kommer magnetisk disk eller SSD som vi og har i dag. her blir all informasjon som ikke er i bruk lagret som spill, filer eller filmer. disse diskene kan vær små kjappe SSD som er et par hunder gigabyte eller store HDD kan være et par terabyte store [4]

Oppgave 4

Hva er hovedoppgavene for minne administratoren i operativsystemer?

Minne administratorens er ansvarlig for å administrera dataens hoved minne. da skal og effektivt administrere minne, holde øye av hva del av minne som er i bruk, allokere minne til prosessen når da er behov for, og å så deallokere når da er

ferdig.Håndterer også swapping mellom disk og hoveddminne. den håndterer også swapping mellom disk og hoved minne. [5]

Oppgave 5

Det er to problemer med multiprogrammering. Hvilke?

Det finnes to forskjellige problemer på multiprogrammering, disse to heter Relokasjon og beskyttelse.

Beskyttelse handler om at en vil bare at de forskjellige programmene skal kunne bare lese fra minneområdet som er deres eget, så en trenger en vei å beskytte de forskjellige minneområdene fra å møtes sammen.

Relokasjon handler om at hver av prosessene som blir kjør har sitt egne minneområde, vi må derfor vite hvilket minne som hører sammen.

Oppgave 6

Forklar hvordan MMU virker sammen med minnet og disken. Bruk paging som et eksempel?

For å utnytte ram og virtuelt minne så godt som mulig så ble en teknikk kalt paging/sideveksling funnet opp. Paging/Sideveksling har to hoveddoppgaver, disse er å håndtere page faults som er gyldige og swappe ut pages som blir sjeldent eller lite brukt.

En page er kort fortalt en bit av en “address space” til ett program og finnes da i det virtuelle minnet. Pagene blir mappet til det fysiske minnet, men en trenger ikke alle i det fysiske minnet samtidig for å kunne kjøre programmet. Page frame derimot er ekvivalenten av en page bare i det fysiske minnet. En page og en page frame har alltid den samme størrelsen.

Ved paging så er det virtuelt minne som blir tatt i bruk, CPU’en sender ut virtuelle adresse. Disse virtuelle adressene går til MMU’en. MMU’en sin jobb å mappe og sende disse virtuelle adressene til fysisk minne adresse i page table, og dette blir sendt til RAM. I MMUen så er TLB. TLB oversetter virtuelle adresser til fysiske adresser uten å måtte bruke page table, men av og til er ikke den virtuelle pagen i TLB og da må MMUen manuelt sjekka page tablet. I noen system så er TLB entrien i OSet, så når en TLB miss skjer så genererer den en TLB fault og sender den til OS i steden for at MMUen skal henta page tablet sjølv. enn har 2 forskjellige missa. soft miss og hard miss. soft miss er når det ikke finne i TLB, men i minnet. da trenger TLBen kun og bli oppdater og dette går ganske raskt. I noen tilfeller er det ikke i TLB og ikke i minne, da må det ned i harddisken for og hente det opp. dette tar som regel mer tid enn å hente det i minne. [3] [6]

Oppgave 7

Forklar hvordan bitmaps blir brukt for å holde orden på minnebruken når man har swapping?

Med bitmaps så blir minne delt inn i tildelingsenheter som kan vær fra et par ord til flere kilobytes. Til hver tildelingsenhet er en bit i bitmapet som er 0 hvis enheten er ledig og 1 hvis den er opptatt.

Bitmaps gir en enkel måte å holde øye med minneorda i et fikst beløp av minnet fordi størrelsen av bitmape avhenger kun av størrelsen av minnet og størrelsen av tildelingsenheten.

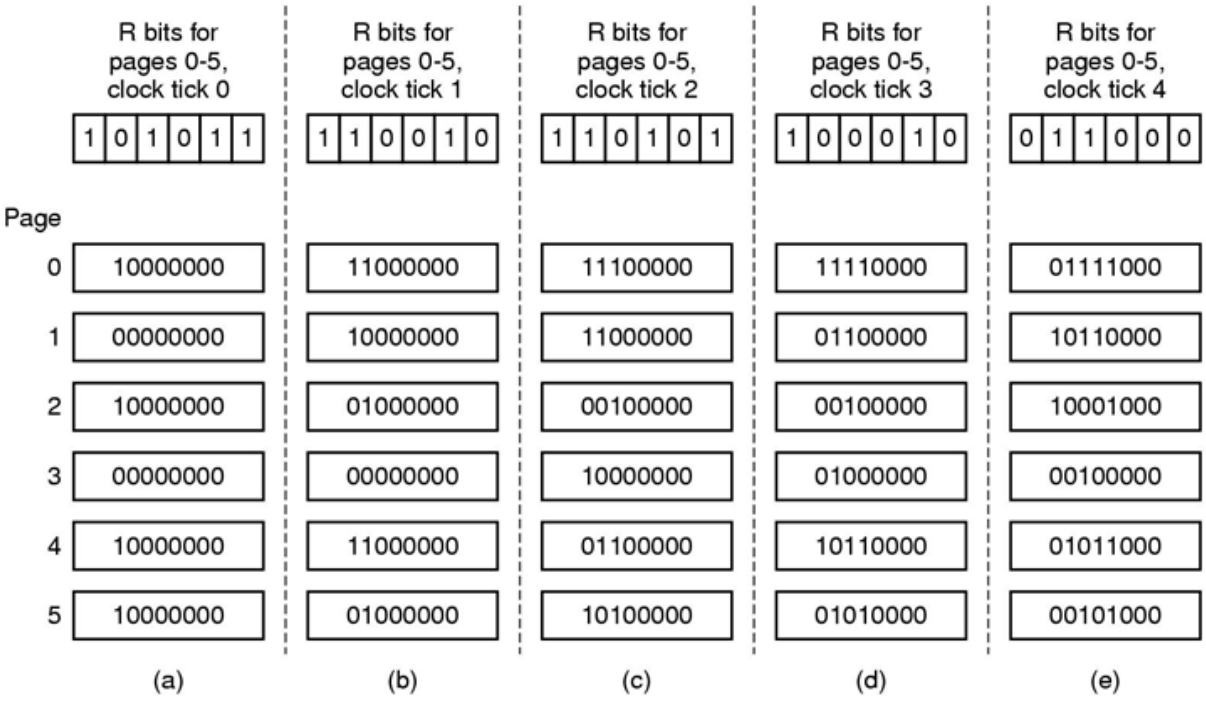
Bitmaps er et kart som holder øye med hva som er ledig og hvor stor plass det er. dette kan så bli brukt av swapping til å vite hva som kan gå hvor og om det er stor nok plass til det. [7]

Oppgave 8

Forklar sideskift algoritmen ”aging”. Hvordan virker den?

Sideskifte algoritmen “aging” er LRU (Least Recently Used) i software. Dette er en type modifisert NFU (Not frequently used).

Den fungerer slik at hvis en side/page blir referert så blir det telt til (1) ved hjelp av software. Dette kan kalles for R (Referanse) bit. I aging så skifter bitene til høyre før vi legger til dette R bit’et, og dette bit’et blir plassert til venstre for telleren.



Figur 1 Simulating LRU in Software

Så når det etterhvert oppstår en page fault så vil denne algoritmen velge å skifte ut den siden som har den laveste telleren. Så hvis det skulle oppstå en page fault i eksempelet over så ville det vært page 3 som gikk ut. [3]

Oppgave 9

Forklar forskjellen mellom Translation lookaside buffer og Software TLB managament.

Translation Lookaside Buffer er en minne catch som holder styr på nylige oversettelse av virtuelle minne til fysisk minne. Dette reduserer tiden det tar for å få tilgang til en bruker minne lokasjon. da blir brukt til å kartlegger virtuelle adresser til fysiske adresser uten at da må gå igjennom page tabelt. da er som oftast inni MMUen og består av få antall med oppføringer og som oftes overgår det ikke 256 oppføringer. [8]

Ved Hardware TLB miss så vil CPUen automatisk gå over til page table for å finna virtuell adresse, om den finner den så blir den lasta opp inn i TLB og derifra vil den få en TLB hit. Hardware TLB vet ikke software formaten til TLB entrien, entrien kan vær forskjellig fra CPU til CPU uten at det påvirker kompatibiliteten

ved software TLB miss så blir en TLB miss exception generert så blir OS ansvarlig for å gå til page table og utføre oversettelsen fra page tabel entrien til virtuell adresse. sotfware TLB er OSen som bestemmer hvilke pages som skal bli oppført i TLBen. [9]

References

[1] Geeksforgeeks, "geeksforgeeks.org," [Online]. Available: https://www.geeksforgeeks.org/logical-and-physical-address-in-operating-system/. [Accessed 31 10 2019].

[2] Geeksforgeeks, "geeksforgeeks.org," [Online]. Available: https://www.geeksforgeeks.org/page-replacement-algorithms-in-operating-systems/ - . [Accessed 1 11 2019].

[3] Harlvard, *Powerpoint Minne administrering.*

[4] A. S. T. o. H. Bos, "Modern Operating System," in *Modern Operating System fourth edition*, Pearson Eductaion Limited, 2015, pp. 24-27.

[5] A. S. T. o. H. Bos, "Modern Operating System," in *Modern Operating System fourth edition*, Pearson Education Limited, 2015, p. 180.

[6] A. S. T. o. H. Bos, "Modern Operating System fourth edition," Pearson Education Limited, pp. 194-196 203-204.

[7] A. S. T. o. H. Bos, "Modern Operating System fourth edition," Pearson Education Limited, p. 191.

[8] A. S. T. o. H. Bos, Modern Operating System fourth edition, Pearson Education Limited.

[9] S. F. Leder.