Leheküljetabelite realiseerimine

- Leheküljetabeleid hoitakse mälus
- Leheküljetabeli algust näitab vastav register (PTBR)
- Leheküljetabeli pikkust näitab kah vahel register (PTLR)
- Iga soovitav mälupöördus nõuaks kahte tegelikku mälupöördust (leheküljetabeli kirje + tegelik pöördus tõlgitud aadressil)
- Seda probleem lahendab viimati- (sagedamini?) kasutatud kirjete puhverdamine protsessori registrites
- TLB (*Translation Lookaside Buffer*) – assotsiatiivmälu abil realiseeritud leheküljetabeli kirjete puhver
- TLB *miss* – alati ei leita vastavat kirjet TLB-st, sel juhul tuleb ta mälust TLB-sse lugeda
- ASID (*Address Space ID*), TLB *flush*

Efektiivne mälupöördusaeg

- Reaalne mälupöördus = 1 ajaühik
- Assotsiatiivmälust otsimine = ϵ ajaühikut
- Assotsiatiivmälu edukate otsimiste osakaal (*hit ratio*) – mitmel protsendil juhtudest leiab tulemuse assotsiatiivmälust. Tähistame α
- Efektiivne mälupöörduse aeg:

$$EAT = (1+\epsilon)\alpha + (2+\epsilon)(1-\alpha) = 2+\epsilon-\alpha$$

Mälukaitse

- Mälukaitset saab (ja on mõtet) siduda iga lehekülgega
- Iga lehekülje juurde paneme lipu *valid*
  - *valid* – lehekülg on protsessi mälupiirkonnas
  - *invalid* – lehekülge ei ole protsessi mälupiirkonnas
- Näiteks mälupiirkonna lõpu tähistamiseks

Hierarhilised leheküljetabelid

- Kuna üks leheküljetabel läheks liiga suureks, teeme leheküljetabeli mitmetasemelise (hierarhilise)
- Näiteks kahetasemeline – aadress jagatakse leheküljesiseseks aadressiks ning lehekülje number omakorda kaheks bitigrupiks (kummagi taseme jaoks üks grupp, määrab indeksi vastavas tabelis)
- Näiteks 32-bitine aadress, 4K lehekülg (12 bitti), ülejäänud 20 bitti jagatakse kaheks 10-bitiseks leheküljenumbriks (välimise leheküljetabeli indeks ja sisemise leheküljetabeli indeks)
- Füüsilised leheküljed võivad endiselt olla läbisegi (sõltumata kummagi tabeli järjekorrast)
- Kolmetasemelised, neljatasemelised, . . .

Paisksalvestusega leheküljetabelid

- 64-bitisel süsteemil tuleks palju tasemeid leheküljetabeleid, pole efektiivne
- Virtuaalse lehekülje number salvestakse paiskfunktsiooni abil leheküljetabelisse
- Leheküljetabel on välisahelatega paisktabel (mitu erinevat lehekülge võivad samale positsioonile sattuda)
- Kasutamine kiire
- Kontekstivahetus (väga) aeglane

Pööratud leheküljetabel

- Idee: leheküljetabelis on kirje iga füüsilise mälu lehekülje kohta
- Kokku 1 tabel, mitte igal protsessil oma tabel, mida siis vahetada tuleks
- Füüsilise mälu leheküljele vastav tabelikirje koosneb protsessi identifikaatorist ja protsessisisesest loogilisest aadressist
- Mälutarve väiksem
- Otsimisaeg suurem
- Otsimise kiirendamiseks paisktabel leheküljetabeli ees (muidugi ka TLB)

Jagatud mäluleheküljed

- Koodi sisaldavaid lehekülgi on mõtet jagada
- Võimalik siis, kui kood on taassisenetav (*reentrant*) – ei modifitseeri iseennast
- Jagatud lehekülg on nähtav mitme protsessi aadressruumist
- Sissekirjutatud aadressidega kood peab igas protsessis samal virtuaalaadressil asuma
- Positsioonist sõltumatu kood (PIC – *Position Independent Code*) võib igas protsessis olla erineva virtuaalaadressi peal
- Uuemal ajal on jagatud teegid enamasti positsioonist sõltumatud
- Igal protsessil oma andmed ja enamasti ka mingi osa mittejagatud koodi
- Pööratud leheküljetabelite puhul on lehekülgede jagamine raskem

Segmenteerimine

- Segmenteerimine on mäluhalduse skeem, kus ei kasutata mitte ühte suurt pidevat loogilist aadressruumi, vaid paljusid segmente
- Läheb paremini kokku paljude kasutajate mõttemaailmaga
- Läheb halvemini kokku paljude süsteemprogrammeerijate mõttemaailmaga :)
- Programm on komplekt segmente. Segment on loogiline ühik, näiteks
  - Põhiprogramm
  - Protseduur, funktsioon
  - Objekt
  - Meetod
  - Lokaalsed muutujad, globaalsed muutujad
  - Pinu
  - Massiiv
  - Sümbolitabel

Segmenteerimisega arhitektuur

- Loogiline aadress on paar <segmendi number, nihe>
- Segmenditabel – seab igale segmendi numbrile vastavusse mälutüki:
  - Baas – segmendi alguse füüsiline aadress
  - Limiit – segmendi pikkus
- Segmenditabeli algus näidatakse vastava registriga (STBR)
- Segmenditabeli pikkus kah enamasti registris (STLR) – sisuliselt on tegemist maksimaalse kasutatava segmendi numbriga
- Mälu jagatakse tervete segmentidega kaupa (first-fit, best-fit näiteks), seega tekib väline fragmenteerumine
- Segmente saab protsesside vahel jagada

Segmenteerimine ja mälukaitse

- Mälukaitse läheb üsna loomulikku rada – enamasti on tervel segmendil sama kaitsetase, seega seome loabiti(d) segmenditabeli kirjega:
  - *valid* – kas segment on kasutusel
  - lugemise/kirjutamise/koodi täitmise lubamise bitid (suvalises kombinatsioonis)

Segmenteerimine koos lehekülgedega

- MULTICS lahendas välise fragmenteerumise probleemi segmenditabelite lehekülgedeks jagamisega – iga segmenditabeli kirje oli viide vastava segmendi lehekülgetabelile
- Sarnane kombineeritud lähenemine on tänapäevalgi kasutusel
- Inteli 386 (tegelt kogu edasine x86 haru) kasutab kah segmenteerimist ja lehekülgi mõlemaid, kuid üksteise järel: segmeditabelist saadakse tulemusena mingi aadress ning sellele rakendatakse kahetasemelist lehekülgetabelit
- Mõlema head küljed ära kasutatavad, natukese mälu ja keerukuse (OS ja protsessor) hinnaga