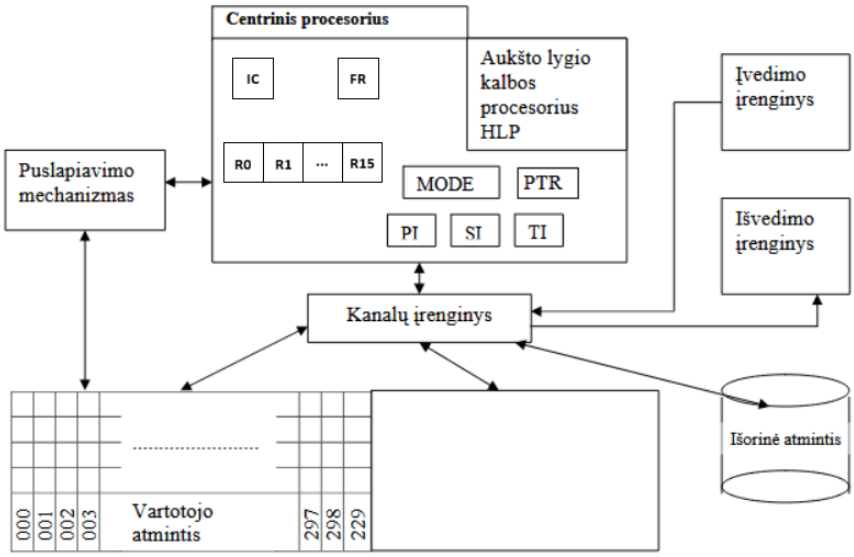
Realios Ir Virtualios Mašinos Aprašas

Mantvydas Skruodys,
Deivydas Tamulis,
Edgaras Gurasukas

2025 m. spalio 31 d.

1 Reali mašina



1 pav.: Realios mašinos bendroji schema

1.1 Procesorius

Procesoriaus funkcija – nuskaityti iš atminties komandą ir ją įvykdyti (interpretuoti). Jis gali veikti dviem režimais: supervizoriaus ir vartotojo.

Supervizoriaus režime komandos, esančios specialioje atminties srityje, yra tiesiogiai apdorojamos aukšto lygio kalbos procesoriaus (HLP). HLP gali būti bet kurios programavimo kalbos vykdymo mechanizmas. Vartotojo režime HLP vykdo konkrečios užduoties programą.

1.2 Registrai

- *R0-R15* – bendro naudojimo registrai, skirti laikinai saugoti duomenis ir tarpinius rezultatus.
- *IC* – registras, nurodantis sekančią komandą, programos skaitiklis.
- *FR* – baito dydžio registras, kurio bitai yra vėliavos:
  - *SF*: bitas lygus 1, kai paskutinės operacijos rezultatas yra neigiamas;
  - *ZF*: bitas lygus 1, kai paskutinės operacijos rezultatas yra nulis;
  - *CF*: bitas lygus 1, kai paskutinės operacijos rezultatas netelpa į registro rėžius.

- *MODE* – registras, kurio reikšmė nusako kuris darbo režimas veikia (vartotojas ar supervizorius).
- *PTR* – puslapių lentelės registras, palaiko adresų transformaciją tarp virtualios ir realios atminties.
- *PI* – programinių pertraukimų registras, naudojamas fiksuoti ir apdoroti programinių klaidų ar nenumatytų situacijų signalus, atsirandančius vykdant vartotojo programą.
- *SI* – supervizorinių pertraukimų registras, naudojamas fiksuoti operacinės sistemos iškvietimus, kurie yra tyčiniai – juos sukelia vartotojo programa, kai vykdomos duomenų įvedimo/išvedimo operacijos, failų atidarymas, uždarymas, įrašymas ar skaitymas ir t.t.
- *TI* – taimerio registras riboja užduoties vykdymo laiką: kiekvienai užduočiai suteikiamas tam tikras taktų skaičius (pvz., N = 10), o vykdant instrukcijas taimerio registras mažinamas. Kai jo reikšmė tampa nulinė, sukuriamas pertraukimas, sustabdantis einamąją užduotį.

1.3 Papildomi registro duomenys

Realiosios mašinos atmintyje taip pat yra saugomi papildomi duomenys, kaip:

VMs - masyvas, saugantis virtualias mašinas, pagal jų eilės tvarką.

BUFFER - buferis, skirtas įvedimo/išvedimo duomenims saugoti iki jų panaudojimo.

OFFSET - Skirtas optimizuoti visiškai visą atmintį.

1.4 Atmintis

Atmintis – tai įrenginys, skirtas informacijai saugoti. Mūsų realioje mašinoje yra keturi atminties tipai: supervizorinė, vartotojo, išorinė ir bendra.

1.4.1 Supervizorinė atmintis

Supervizorinė atmintis skirta pačios operacinės sistemos poreikiams – čia laikomi sisteminiai procesai, kintamieji, resursai bei mikroprogramos, kurios interpretuoja virtualaus procesoriaus komandas. Šiame modelyje supervizorinės atminties komandas ir resursus valdo aukšto lygio kalbos procesorius (HLP). Siekdami supaprastinti modelį, supervizorinės atminties atsisakome, bet pačią sąvoką vis tiek naudosime.

1.4.2 Vartotojo atmintis

Vartotojo atmintis skirta virtualių mašinų atmintims ir puslapių lentelėms laikyti. Mūsų modelyje vartotojo atmintis apibrėžiama taip: jos dydis – 16 blokų po 16 žodžių, kiekvienas žodis 4 baitų ilgio. Iš viso turime 256 žodžius (arba 1024 baitus). Blokai numeruojami nuo 0 iki 15, o žodžiai – nuo 0 iki 255.

1.4.3 Išorinė atmintis

Išorinė atmintis – tai įrenginys, skirtas programoms ir duomenims saugoti ir laikyti, kurie netelpa arba nėra laikomi vartotojo atmintyje. Mūsų modelyje tai atitinka kietąjį diską, į kurį saugomos vartotojo programos.

1.4.4 Bendra atminties sritis

Bendra atminties sritis - galima pasiekti visoms vartotojo programoms. Ji yra naudojama duomenims persiūsti tarp programų. Ji yra apsaugota semaforais. Semaforas – loginis kintamasis: reikšmė 1 reiškia - sritis laisva, o reikšmė 0 – kad užimta.

1.5 Įvedimas / išvedimas

Duomenų įvedimui naudojama klaviatūra – vartotojo įrašyta informacija pirmiausia patenka į buferį, iš kurio, naudojant specialias komandas, ji perkeliama į vartotojo atmintį. Rezultatams pateikti skirtas ekranas, kuriame programos išvedami duomenys pasirodo po to, kai jie iš atminties perduodami į išvedimo buferį.

1.6 Kanalų įrenginys

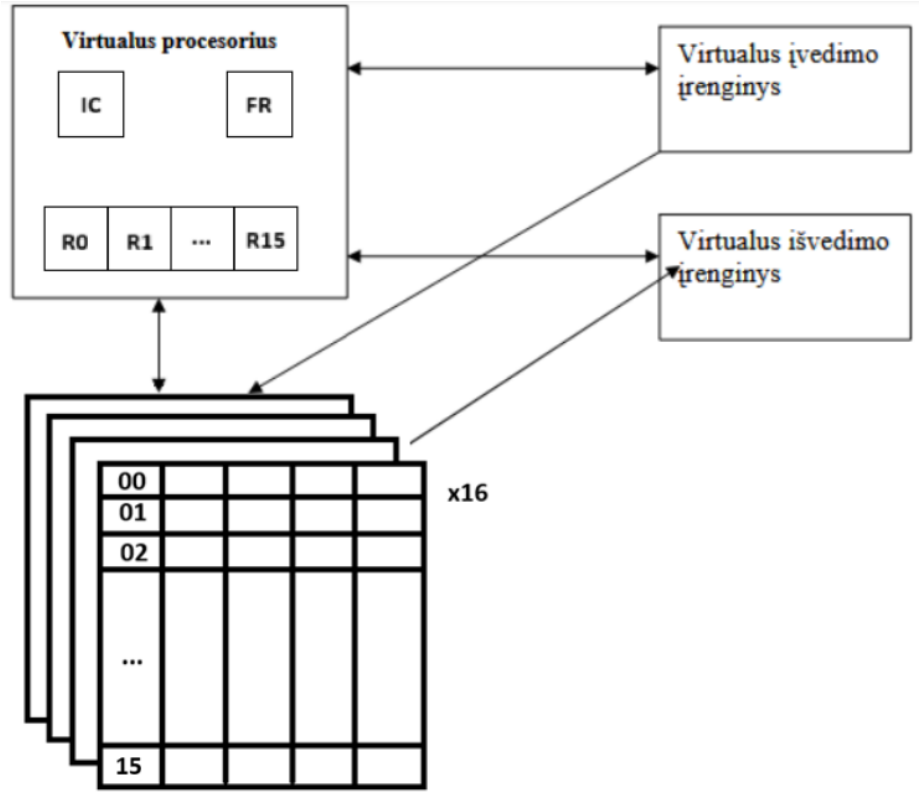
Kanalų įrenginys skirtas duomenų mainams tarp įvairių atminties sričių ir išorinių įvedimo ar išvedimo srautų. Jo paskirtis – sumažinti procesoriaus apkrovą, kai reikia perkelti didelius duomenų kiekius. Kanalų įrenginys turi kelis registrus, kurie nustato, iš kur bus kopijuojami duomenys ir kur jie turi būti įrašyti. Šie registrai yra SB (šaltinio takelio numeris), DB (paskirties takelio numeris), ST (šaltinio objekto tipas) irDT (paskirties objekto tipas). Objektais gali būti vartotojo atmintis, supervizorinė atmintis, išorinė atmintis arba įvedimo ar išvedimo srautai. Kai visi šie registrai yra nustatyti, inicijuojamas duomenų perdavimas. Su kanalų įrenginiu galima dirbti tik supervizoriaus režime, nes jis tiesiogiai sąveikauja su atminties struktūromis ir išoriniais įrenginiais.

Mūsų virtualios mašinos modelyje kanalų įrenginys nėra realizuojamas kaip savarankiškas komponentas. Jo funkcijos įgyvendinamos per aukšto lygio kalbos procesoriaus vykdomas specialias komandas, kurios imituoja duomenų judėjimą tarp skirtingų atminties sričių bei tarp įvedimo ir išvedimo srautų. Tokiu būdu pasiekiamas tikslas pavaizduoti duomenų mainus, nors pats kanalų įrenginys egzistuoja tik kaip loginė modelio dalis.

1.7 Puslapiavimo įrenginys

Realios mašinos atmintis padalinta į dvi dalis: supervizorinę atmintį, skirtą operacinės sistemos procesams, resursams ir puslapių lentelėms saugoti ir vartotojo atmintį – 16 blokų po 16 žodžių, skirtą virtualių mašinų duomenims ir programoms. Kiekvienai naujai virtualiai mašinai skiriama iki 16 vartotojo atminties blokų. Virtuali mašina savo blokus mato sunumeruotus nuo 0 iki 15, tačiau realiame atminties išdėstyme šie blokai gali būti bet kurioje vartotojo atminties vietoje.

2 Virtuali mašina



2 pav.: Virtualios mašinos schema

Virtuali mašina tai realios mašinos modelis (kopija) tokiu pavidalu, kokį gali priimti mašinos vartotojas. Ji yra kaip tarpininkė tarp programinės įrangos ir konkrečios mašinos. Virtuali mašina yra modeliuojama realios mašinos pagrindu. Ji simuliuoja realios mašinos funkcionalumą pasiekdama visus esamus lygmenis.

2.1 Registrai

- *R0-R15* – bendro naudojimo registrai, skirti laikinai saugoti duomenis ir tarpinius rezultatus.
- *IC* – registras nurodantis sekančią komandą, programos skaitiklis.
- *FR* – baito dydžio registras, kurio bitai yra vėliavos:
  - *SF*: bitas lygus 1, kai paskutinės operacijos rezultatas yra neigiamas;
  - *ZF*: bitas lygus 1, kai paskutinės operacijos rezultatas yra nulis;
  - *CF*: bitas lygus 1, kai paskutinės operacijos rezultatas netelpa į registro rėžius.
  - *Papildomi duomenys*: Offset - pointeris į poslinkį nurodantį kintamąjį realios mašinos CPU.

2.2 Atmintis

Virtuali mašina turi 16 blokų po 16 žodžių(1 žodis užima 4 baitus) atminties.

Brėžinyje vienas baitas yra vienas stačiakampis.

Iš viso turime 256 žodžius atminčiai:

- 64 – duomenys,
- 128 – programos kodas,
- 48 – laisva atmintis,
- 16 – bendra atmintis.

2.3 Komandos

2.3.1 Aritmetinės

- *ADDxyz* - (Addition):
  prie registro *Rx* reikšmės prideda registro *Ry* reikšmę ir atsakymą įrašo į registrą *Rz*;
- *SUBxyz* - (Subtraction):
  iš registro *Rx* reikšmės atima registro *Ry* reikšmę ir atsakymą įrašo į registrą *Rz*;
- *MULxyzw* - (Multiplication):
  Sudaugina registrų *Rx* ir *Ry* reikšmes, rezultatai patalpinami į *Rz* ir *Rw* registrus.

- *DIVxyzw* - (Division):
  Registro Rx reikšmė padalinama iš registro Ry reikšmės, dalybos rezultato sveikoji dalis patalpinama į registrą Rz, o rezultato liekana patalpinama į Rw registrą.

2.3.2 Palyginimo

- *CMPxy* - (Compare):
  Palygina registrų Rx ir Ry reikšmes, jei reikšmės vienodos į ZF įrašoma reikšmė 1, jei Ry reikšmė didesnė už Rx tai į SF įrašoma reikšmė 1.

2.3.3 Darbo su duomenimis

- *MRxyz* - (Memory Read):
  atminties ląstelės, kurios adresas $x \times 16 + y$ turinio kopijavimas į registrą Rz. Trumpiau: $Rz := [x \times 16 + y]$;

- *MWxyz* - (Memory Write):
  registro Rz reikšmė įrašoma į atminties ląstelę, kurios adresas $x \times 16 + y$. Trumpiau $[x \times 16 + y] := Rz$;

- *SMRxyz* – (Shared Memory Read):
  Iš bendros atminties ląstelės, kurios adresas $x \times 16 + y$, perskaito reikšmę ir įrašo ją į registrą Rz.
  Trumpiau: $Rz :=$ Shared$[x \times 16 + y]$.

- *SMWxyz* – (Shared Memory Write):
  Į bendros atminties ląstelę, kurios adresas $x \times 10 + y$, įrašo registro Rz reikšmę.
  Trumpiau: Shared$[x \times 16 + y] := Rz$.

- *WAIT* – (Semaforo užėmimas):
  Jeigu semaforo reikšmė yra 1 (laisva), ji pakeičiama į 0 ir programa tęsia darbą.
  Jeigu semaforo reikšmė yra 0 (užimta), procesas laukia, kol semaforas bus 1.

- *SIGNAL* – (Semaforo atlaisvinimas):
  Semaforo reikšmė nustatoma į 1.

2.3.4 Valdymo

- *JMPxy* (jump) – besąlygiškas šuolis; registras IC perstumiamas į vietą kode, pažymėtą „X:“;

- *JExy* (jump if equal) – jei ZF (Zero Flag) yra 1, registras IC perstumiamas į vietą kode, pažymėtą „X:“; priešingu atveju vykdymas tęsiamas toliau;

- *JGxy* (jump if greater than) – jei *SF* (Sign Flag) yra 0, registras *IC* perstumiamas į vietą kode, pažymėtą „X:“; jei *SF* yra 1, šuolis nevykdomas;
- *JLxy* (jump if less than) – jei *SF* (Sign Flag) yra 1, registras *IC* perstumiamas į vietą kode, pažymėtą „X:“; jei *SF* yra 0, šuolis nevykdomas;
- *JLExy* (jump if less or equal) – jei *SF* yra 1 arba *ZF* yra 1, registras *IC* perstumiamas į vietą kode, pažymėtą „X:“;
- *JCxy* (jump if carry) – jei *CF* (Carry Flag) yra 1, registras *IC* perstumiamas į vietą kode, pažymėtą „X:“;
- *JNCxy* (jump if not carry) – jei *CF* reikšmė yra 0, registras *IC* perstumiamas į vietą kode, pažymėtą „X:“;
- *HALT* – programos pabaigos komanda, stabdanti VM vykdymą.

2.3.5 Įvedimo/Išvedimo

- *DMARx* - iš įvedimo srauto perskaito 16 žodžių ir įrašo juos į ląsteles $[x \times 16 + i\ ]$, kur $i = 0, 1, ..., 15$.
- *DMAWx* - išsiunčia išvedimui 16 žodžių srautą iš atminties ląstelių $[x \times 16 + i\ ]$, kur $i = 0, 1, ..., 15$.

2.4 Interpretuojamojo ar vykdomojo failo formatas

Virtualios mašinos vykdomojo failo struktūra susideda iš trijų pagrindinių dalių:

1. parametrų,
2. programos pradžios žymės,
3. programos pabaigos žymės.

Pirmieji keturi baitai visada turi turėti simbolių rinkinį $START, nurodantį, kad tai multiprogramavimo užduotis. Kiti keturi baitai skirti maksimaliam išvedimo eilučių skaičiui – tai apsauga nuo programos užstrigimo amžinuose cikluose. Likę baitai iki programos pradžios skirti užduoties pavadinimui, kuris gali būti sudarytas iš tekstinių simbolių.

Po parametrų seka programos dalis, kurios ilgis – 512 baitai, arba 128 žodžiai po 4 baitus.

Paskutiniai 4 baitai failo pabaigoje visada turi turėti žymę $END, kuri signalizuoja vykdomojo failo pabaigą. Toks struktūruotas formatas leidžia virtualiai mašinai tiksliai perskaityti užduotį, jos parametrus ir vykdyti programą nuo pradžios iki pabaigos, užtikrinant nuoseklų ir patikimą darbą su įvedimo/išvedimo

srautais bei duomenų blokais.

Sąryšis su realios mašinos technine įranga:

Registrai (R0–R15, IC, C) – atitinka realios CPU registrus;

Atmintis – atitinka RAM bloką;

Komandų sistema – emuliuoja procesoriaus instrukcijų rinkinį;

Įvedimo/išvedimo mechanizmas – susijęs su realios sistemos įvedimo/išvedimo įrenginiais.

Virtuali mašina operacinės sistemos kontekste veikia kaip tarpininkas, leidžiantis vykdyti programas nepriklausomai nuo konkrečios aparatūros, užtikrinant vienodą funkcionalumą visiems vartotojams.