Realios ir virtualios mašinos aprašas

Goda Gutparakytė ir Tomas Baublys

2025 m. rugsėjis

Turinys

1 Realios mašinos aprašas

Realią mašiną sudaro:

- procesorius
- vartotojo atmintis
- išorinė atmintis
- supervizorinė atmintis
- bendra atmintis
- duomenų perdavimo kanalai (kanalų įrenginys)
- įvedimo/išvedimo įrenginiai

1.1 Procesorius

Procesoriaus paskirtis - skaityti komandą iš atminties ir ją vykdyti (interpretuoti). Procesorius gali dirbti dviem režimais – supervizoriaus arba vartotojo. Supervizoriaus režime komandos iš supervizorinės atminties yra apdorojamos betarpiškai aukšto lygio kalbos procesoriaus HLP. HLP – bet kuris aukšto lygio kalbos procesorius (programavimo kalbos). Vartotojo režime HLP vykdo užduoties programą. Šiuo atveju HLP imituoja virtualios mašinos procesorių. Dabar apžvelgsime procesoriaus registrus:

- PC - 2 baitų virtualios mašinos programos skaitiklis
- PI - 2 baitų programinių pertraukimų registras
- SI - 2 baitų supervizorinių pertraukimų registras
- CI - 1 baito taimerio registras
- SF - 2 baitų požymių registras, 1 bitas - pernešimo bitas, 2 bitas - nulio bitas, 3 bitas - perpildymo bitas, 4 bitas - žingsniavimo bitas, 5 bitas - sekimo bitas
- TR - 2 baitų puslapių lentelės registras
- RA - 4 baitų bendrosios paskirties registras
- RB - 4 baitų bendrosios paskirties registras
- RC - 4 baitų bendrosios paskirties registras, taip pat naudojamas ciklams (LOOP komandai)
- MR - 1 baito registras, kurio reikšmė nusako procesoriaus darbo režimą (vartotojo arba supervizorinis)
- SS - 1 baito semaforo registras skirtas blokuoti bendrą atmintį

1.2 Taimerio mechanizmas

Šis mechanizmas atsakingas už geresnį užduočių išlygiagretinimą. Tai reiškia bus vykdoma ne daugiau N = 10 einamosios užduoties taktų. Laikysim kad įvedimo/išvedimo instrukcijos atliekamos per 3 taktus, visos kitos per 1 taktą. Dabar apie veikimo principą: Pradedant virtualios mašinos užduoties vykdymą TI registro reikšmė nustatoma reikšmei N = 10. Įvykdžius eilinę instrukciją TI reikšmė mažinama priklausomai nuo to per kiek taktų ši instrukcija yra atliekama. Kuomet TI reikšmė yra lygi nuliui, mikrokomanda Check () aptinka taimerio pertraukimą.

1.3 Pertraukimų mechanizmas

Pertraukimas – tai signalas apie įvykusį įvykį. Kiekvienas pertraukimas turi savo identifikaciją (sistema turi turėti galimybę atskirti pertraukimų tipus). Pertraukimas savaime nepertraukia sistemos darbo, kaip kad gali pasirodyti iš pirmo žvilgsnio. Pertraukimą sistema turi aptikti ir atitinkamai sureaguoti. Pertraukimas - tai tik pakeista kompiuterio būsena. Kompiuterio būseną keičia tą būseną sukėlusi priežastis. Pertraukimus aptinka procesoriaus komanda Check (), kuri apklausia reikalingus registrus. Ir tik sistemai aptikus pertraukimą, yra nutraukiamas vartotojo programos vykdymas. Modelyje bus realizuoti trijų tipų pertraukimai – programiniai, supervizoriniai ir taimerio. Programinių pertraukimų registras yra PI, supervizorinių pertraukimų registras – SI, taimerio - TI. Programiniai pertraukimai kyla vykdant virtualią mašiną, bandant įvykdyti kokį nors neleistiną veiksmą arba nuskaičius neleistiną reikšmę. Supervizoriniai pertraukimai kyla virtualiai mašinai norint įvykdyti veiksmą, kuris gali vykti tik supervizoriaus režime. Pertraukimai gali būti aptikti tik vartotojo režime. Supervizoriniame režime centrinio procesoriaus darbo pertraukti negalima.

Pertraukimai kils šiais būdais:

- Operacijos GD, PD ir STOP iššauks supervizorinius pertraukimus
  - SI = 1 - komanda GEDA,
  - SI = 2 - komanda PUTA,
  - SI = 3 - komanda PSTR,
  - SI = 4 – komanda LW,
  - SI = 5 - komanda SW,
  - SI = 6 - komanda BP,
  - SI = 7 - komanda BG,
  - SI = 8 - komanda STOP,
- Programiniai pertraukimai
  - PI = 1 – neteisingas adresas,
  - PI = 2 – neteisingas operacijos kodas,

  - PI = 3 – neteisingas priskyrimas,
  - PI = 4 – perpildymas (”overflow”),
  - PI = 5 - dalyba iš nulio.

- Taimerio pertraukimai
  - TI = 0.

Esant situacijai SI = 0 ir PI = 0 ir TI <> 0, pertraukimų sistema neaptiks.

1.4 Procesoriaus darbo režimai

Kaip ir aprašyta aukščiau, procesorius gali dirbti dviem režimais - vartotojo ir supervizoriaus. Jei MR registro reikšmė yra nulis, tai procesorius dirba vartojo režime, o jei MR registro reikšmė yra vienas, procesorius dirba supervizoriaus režime.

1.5 Vartotojo atmintis

Vartotojo atmintis skirta virtualių mašinų atmintims bei puslapių lentelėms laikyti. Mes apibrėšime vartotojo atmintį taip: lentelės dydis – 51 takelis po 16 žodžių.m Taigi vartotojo atmintis lygi 51-am blokui, sunumeruotų nuo 0 iki 32, arba 816 žodžių sunumeruotu nuo 0 iki 32F.

1.6 Išorinė atmintis

Išorinė atmintis bus realizuota failu kietame diske. Tarkime laikysime kad faile yra 512 blokų arba 8192 žodžių. Schematiškai išorinę atmintį galima pavaizduoti analogiškai atminčiai, tik ji nėra dalijama į kitas atmintis. Darbą su išorine atmintimi atliks HLP.

Išorinės atminties dalys:

- $[0x0000; 0x0003]$ - failų skaičius
- $[0x0004; 0x1000]$ - failų pavadinimai (8 baitai), poslinkis (4 baitai) ir dydis (4 baitai)
- $[0x1001; 0x7fff]$ - failai

1.7 Supervizorinė atmintis

Supervizorinė atmintis yra skirta pačios operacinės sistemos poreikiams. Supervizorinėje atmintyje laikomi sisteminiai procesai, sisteminiai kintamieji, resursai, mikroprogramos, interpretuojančios virtualaus procesoriaus komandas. Supervizorinei atminčiai skirti 16 takeliai po 16 žodžių.

1.8 Bendroji atmintis

Bendroji atmintis yra pasiekiama visų virtualių mašinų ir yra skirta bendravimui tarp skirtingų virtualių mašinų. Bendrai atminčiai yra skirti 2 takeliai po 16 žodžių.

1.8.1 Semaforai

Semaforų tikslas yra apsaugoti bendrąją atmintį. Virtualios mašinos gali pasiekti bendrąją atmintį priklausomai nuo SS registro reikšmės. Jei SS registro reikšmė lygi 0, tai bendroji atmintis nėra blokuojama ir į ją galima rašyti. Jei SS registro reikšmė lygi 1, tada bendroji atmintis yra blokuojama ir naudojama kitos mašinos ir ją pasiekti bus galima tik kai SS reikšmė taps 0.

1.9 Kanalų įrenginys

Kanalų įrenginys leidžia dirbti su atmintimis. Priklausomai nuo nustatytų registrų kanalų įrenginys gali vykdyti apsikeitimą duomenimis visomis galimomis kryptimis. Veiksmai su kanalų įrenginiu atliekami tik supervizoriaus režime. Dabar bus pateikta kanalų įrenginio vartotojo sąsaja:

Kanalų įrenginio registrai:

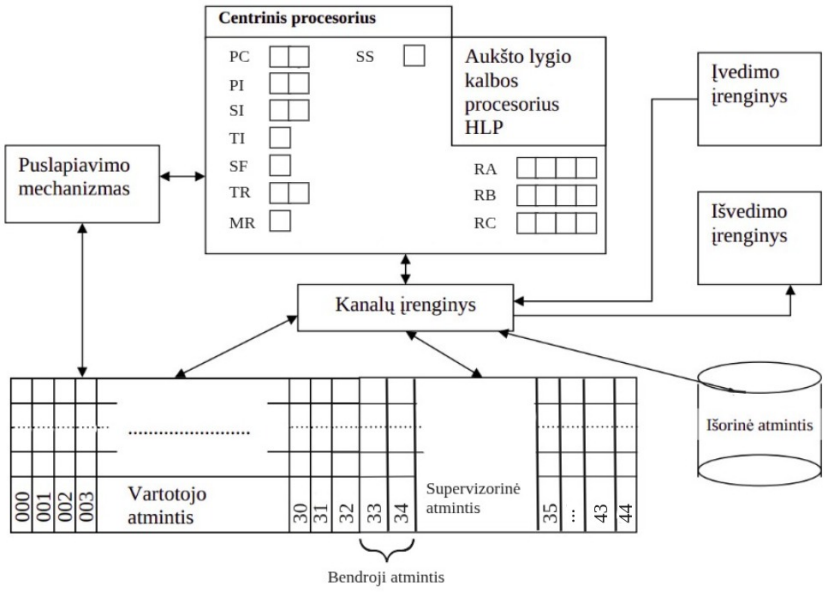
- SB - 2 baitų registras saugantis takelio, iš kurio kopijuosime numerį.
- DB - 2 baitų registras saugantis takelio, į kurį kopijuosime numerį.
- ST - 1 baito registras saugantis objekto, iš kurio kopijuosime, numerį.
- CB - 3 baitų registras saugantis skaičių kiek baitų kopijuosime.
- OF - 2 baitų registras saugantis poslinkį nuo takelio pradžios.
    1. Vartotojo atmintis;
    2. Išorinė atmintis;
    3. Įvedimo srautas;
    4. Bendroji atmintis;
    5. RA registras
- DT - 1 baito registras saugantis objekto, į kurį kopijuosime, numerį
    1. Vartotojo atmintis;
    2. Išorinė atmintis;
    3. Išvedimo srautas;
    4. Bendroji atmintis;
    5. RA registras;
- SA - 4 baitų registras, skirtas perduot argumentams tarp centrinio procesoriaus ir kanalų įrenginio.

Kartu kanalų įrenginys turi komandą **XCHG**, tačiau neturi procesoriaus, kuris galėtų ją įvykdyti. Šią komandą vykdo centrinis procesorius, taigi, šis kanalų įrenginys nėra lygiagrečiai su centriniu procesoriumi veikianti aparatūra. Procesas, norėdamas pasinaudoti kanalų įrenginiu, turi nustatyti kanalų įrenginio registrus ir tada įvykdyti komandą **XCHG**.

1.10 Išvedimo / Įvedimo įrenginiai

Įvedimui naudojama klaviatūra, išvedimui - ekranas.

1.11 Schema



1 pav.: Realios mašinos modelis.

2 Virtualios mašinos aprašas

2.1 Virtualios mašinos samprata

Virtuali mašina – tai tarsi realios mašinos atitikmuo programinėje erdvėje. Ji vadinama virtualia, nes nėra tikra – sudaroma dirbtinai, imituojant pagrindinius komponentus (procesorių, atmintį, įvedimo/išvedimo įrenginius) ir pateikiant paprastesnę sąsają. Ji laikoma kopija, nes atkuria realios mašinos veikimo principus, tačiau nėra identiška – sudėtingi elementai supaprastinami, kad būtų

patogiau programuoti. Virtuali mašina naudoja operacinės sistemos suteiktus virtualius resursus, kurie primena realius, bet yra lengviau valdomi, todėl palengvina programavimą.

2.2 Virtualios mašinos atmintis

Kiekvienai virtualiai mašinai yra skiriama 17 vartotojo atminties blokai. Tuose septyniolika bloku (272 žodžių) turi tilpti užduoties programa + puslapių lentelė. Kiekvienas virtualios atminties blokas turi virtualų ir realų adresą. Virtualiais adresais operuoja virtuali mašina, realiais – reali mašina. Ryšiai tarp virtualaus ir realaus adreso nusakomi puslapių lentelėmis.

2.2.1 Puslapiavimo mechanizmas

Realios mašinos vartotojo atmintis siekia 51 takelių (arba blokų). Kiekvienai naujai sukurtai virtualiai mašinai reikia skirti 17 takelius iš tų 51. Jie gali būti parinkti bet kokiu būdu. Puslapiavimo mechanizmas naudojamas tam, kad virtuali mašina sužinotų kokio nors jai priklausančio takelio realų adresą.

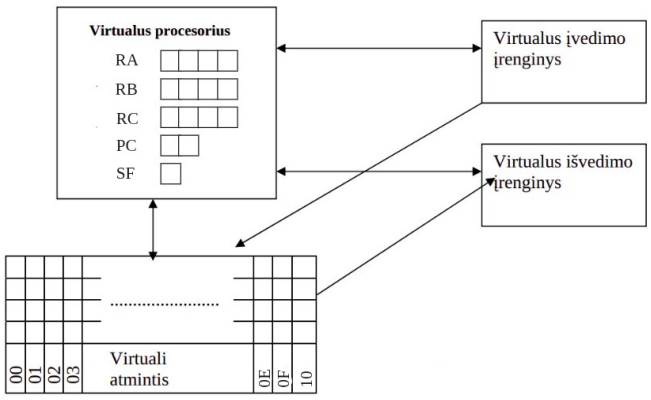
Norint išlaikyti sąryšius tarp realių ir virtualių takelių adresų, bus naudojama puslapių lentelė. Puslapių lentelę sudaro vienas takelis (16 žodžių) ir kiekvieno žodžio eilės numeris atitiks virtualios mašinos eilės numerį ir jame yra laikomas realus to takelio numeris.

2.3 Virtualios mašinos procesorius

Centrinis virtualus procesorius yra gerokai paprastesnis už realios mašinos centrinį procesorių. Virtualios mašinos procesoriaus paskirtis - vykdyti programą, kuri yra virtualioje atmintyje. Modelyje sisteminių procesų programas vykdys aukšto lygio kalbos procesorius. Šiame projekte virtualius procesorius turės tik procesai – virtualios mašinos. Virtualus procesorius turi tris pagrindinius registrus:

- RA - 4 baitų bendrosios paskirties registras
- RB - 4 baitų bendrosios paskirties registras
- RC - 4 baitų bendrosios paskirties registras, taip pat naudojamas ciklams (LOOP komandai)
- PC - 2 baitų virtualios mašinos programos skaitiklis
- SF - 2 baitų požymių registras

2.4 Virtualios mašinos modelis



2 pav.: Virtualios mašinos modelis

2.5 Virtualios mašinos procesoriaus komandų sistema

2.5.1 Atminties valdymo komandos

- **MOxy** - perkelia reikšmę x*16 + y į RA registrą.
- **LRa** - nukopijuoja RA registre esančią reikšmę į a registrą.
- **LLa** - nukopijuoja registre a esančią reikšmę į registrą RB.
- **LWxy** - žodzio atmintyje, kurio adresas yra x*16 + y turinio kopijavimas į registrą RA.
- **SWxy** - registro RA reikšmė įrašoma į zodį atmintyje, kurio adresas x*16 + y.
- **BPxy** - į bendrosios atminties ląstelę esančia adrese x * 16 + y įrašo reikšmę esančia RA registe (žodis)
- **BGxy** - į RA registrą irašo reikšmę esančia x * 16 + y bendros atminties lastėje (žodis)

2.5.2 Įvedimo / Išvedimo komandos

- **GEDA** - iš įvedimo srauto perskaito 1 žodį ir įrašo juos į RA.
- **PUTA** - išsiunčia išvedimui 1 žodžį srautą iš registro RA.

- **PSTR** - išsiunčia išvedimui RA adrese esantį žodį, RC nusako kiek baitų siusti.

2.5.3 Aritmetinės komandos

- **APxy** - prideda reikšmę y*16 + z prie RA registro reikšmės.
- **ADa** - prideda a registro reikšmę prie RA registro.
- **SBa** - atima a registro reikšmę iš RA registro.
- **MUa** - padaugina RA registro reikšmę iš a registro reikšmės.
- **DVa** - padalina RA registro reikšmę iš a registro reikšmės. Sveikoji dalis saugoma RA, liekana RB registre.
- **CMxy** - palygina reikšmę x*16 + y su RA registro reikšme. Rezultatas pažymimas SF registre atitinkamai - jei x*16 + y = RA: ZF = 1, CF = 0, jei x*16 + y < RA: ZF = 0, CF = 1, jei x*16 + y > RA: ZF = 0, CF = 0.
- **CRRB** - palygina RB registro reikšmę su RA registro reikšme. Rezultatas pažymimas SF registre atitinkamai - jei RB = RA: ZF = 1, CF = 0, jei RB < RA: ZF = 0, CF = 1, jei RB > RA: ZF = 0, CF = 0.

2.5.4 Valdymo perdavimo komandos

- **JUxy** - besąlygiškai perduoda valdymą į nurodytą adresą PC = 16*x + y.
- **JZxy** - perduoda valdymą į adresą PC = x*16 + y (jei ZF = 1).
- **JAxy** - perduoda valdymą į adresą PC = x*16 + y (jei CF = 0 ir ZF = 0).
- **JBxy** - perduoda valdymą į adresą PC = x*16 + y (jei CF = 1).
- **JNxy** - perduoda valdymą į adresą PC = x*16 + y (jei ZF = 0).

2.5.5 Ciklo valdymo komandos

- **LOxy** - perduoda valdymą į adresą x*16 + y, jei RC registre esanti reikšmė yra daugiau už nulį bei sumažina RC reikšmę per vienetą.

2.5.6 Loginės komandos

- **XRa** - atlieka registrų a ir RA verčių sumą moduliu 2 pabičiui. Rezultatas įrašomas į RA registrą.
- **ANa** - atlieka registruose a ir RA esančių verčių konjunkciją pabičiui. Rezultatas įrašomas į RA registrą.

- **NOa** - atlieka registro a vertės inversiją pabičiui. Rezultatas įrašomas į tą patį registrą.

2.5.7 Programos terminavimo komanda

- **STOP** - terminuoja programos veikimą.

2.6 Darbo su bendra atmintimi komandos

2.7 Virtualios mašinos bendravimo su įvedimo/išvedimo įrenginiais mechanizmo aprašymas

Virtuali mašina bendrauja su įvedimo/išvedimo įrenginiais komandomis GDxy ir PDxy aprašytomis aukščiau.

2.8 Virtualios mašinos vykdomojo failo struktūra

Užduotys bus laikomos failuose, su kuriais dirbs aukšto lygio kalbos procesorius. Norint sukurti naują užduotį, užtenka sukurti ir teisingai užpildyti naują tekstinį failą modelio išorėje. Bendras užduoties pavidalas bus sudarytas iš pradžios žy,ės, programos ir pabaigos žymės.

- "#LOS" (Love Operating Systems). Pirmasis laukas visada turi turėti šią reikšmę. Ji užima pirmus keturis baitus
- Programos dalis. Šiai daliai skirta 16 * 16 * 4 = 1024 baitai.
- Programos dalį seka pabaigos žymė, kuriai skirti 4 baitai ir ten turėtų būti reikšmė "#BYE".

2.9 Demonstracinis vykdomojo failo pavyzdys

```
0x0B // failų skaičius
fibcfive 0x1080 0x0040 // poslinkis // failo dydis
divszero 0x1264 0x0014
hellwrld 0x1130 0x002C
parottti 0x1314 0x0014
inftloop 0x141C 0x0010
logijnzr 0x1474 0x002C
nesamone 0x1524 0x0011
raadsmem 0x157C 0x0014
biggfile 0x1658 0x0088
bigfile2 0x183C 0x0210
invalid1 0x1AA4 0x006F

#LOS
```

```
MOOO
LRRB
MOOf
LRRC
MOO1
SW44
ADRB
SW40
LW44
LRRB
LW40
LO14
BP40
STOP
#BYE

#LOS
MOOD
LRRC
MO14
PSTR
STOP
HELLO WORLD!
RRR
#BYE

#LOS
MOOO
LRRB
DVRB
#BYE

#LOS
GEDA
PUTA
STOP
#BYE

#LOS
JUOO
STOP
#BYE

#LOS
MOO5
LRRB
```

```
MOO1
XRRB
PUTA
CRRB
JN2O
PUTA
STOP
#BYE

#LOS
dhsaiuyduhfdnsfhdsufhdsifhsdiffdshiufhd

#LOS
BG4O
PUTA
STOP
#BYE

#LOS
MOO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
APO1
```

```
AP01
AP01
AP01
PUTA
STOP
#BYE

E

#LOS
MO01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
```

```
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
```

```
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
AP01
```

```
PUTA
STOP
#BYE

djiohdfioshfosdhfdshoahfdofdshofhdsoahf odihsaodasohfxoahodhohaohfodahofhdoahofxdhoahofhxoah
```

2.10 Virtualios mašinos loginių komponentų sąryšio su realios mašinos techninės įrangos komponentais aprašymas.

Virtualios mašinos loginiai komponentai yra modeliuojami taip, kad atitiktų realios mašinos aparatūros funkcijas. Kiekviena VM (virtuali mašina) dalis – procesorius, atmintis, registrai, įvedimo/išvedimo sąsajos – turi savo atitikmenį fizinėje mašinoje.

Šis sąryšis leidžia:

- vykdyti programas virtualioje aplinkoje, naudojant realios mašinos resursus;
- supaprastinti vartotojo programų kūrimą, nes VM suteikia vienodą, paprastą sąsają;
- efektyviai panaudoti realios mašinos įrenginius, tokius kaip procesorius, atmintis, I/O įrenginiai ir taimeriai.

Pavyzdžiui, virtualus procesorius RA, RB ir RC registrus atvaizduoja į realius bendrosios paskirties registrus, o virtualios atminties blokai yra susieti su realiais vartotojo atminties takeliais per puslapių lenteles. Įvedimo/išvedimo komandos GD ir PD virtualiai sąveikauja su klaviatūra, ekranu ir kietuoju disku, tačiau realūs duomenų perdavimai vyksta per kanalų įrenginį ir supervizorinį režimą.

3 Virtuali mašina operacinės sistemos kontekste

Virtuali mašina operacinės sistemos kontekste yra suprantama kaip izoliuota, abstrakti kompiuterio aplinka, kuri imituoja tikrą aparatūrą. Ji suteikia vartotojo programoms vienodą sąsają su procesoriumi, atmintimi ir įvedimo/išvedimo įrenginiais, tuo pačiu leidžiant operacinei sistemai kontroliuoti resursus, užtikrinti saugumą ir reaguoti į pertraukimus.