VILNIAUS UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

INFORMATIKOS KATEDRA

3 KURSAS

Realios ir virtualios mašinos projektas

Darbą atliko:

Justinas Mačiulis (KM grupė: 2)

Mantas Pipinė (KM grupė: 2)

Paul ius Mitkus (KM grupė: 2)

Darius Turčinskas (MI grupė: 1)

Turinys

[Realios mašinos modelis 3](#_Toc445212584)

[Realios mašinos centrinis procesorius 3](#_Toc445212585)

[Realios mašinos atmintys 4](#_Toc445212586)

[Realios mašinos procesoriaus komandos 4](#_Toc445212587)

[Virtualios mašinos modelis 6](#_Toc445212588)

[Virtualios mašinos centrinis procesorius 6](#_Toc445212589)

[Virtualios mašinos atmintis 6](#_Toc445212590)

[Virtualios mašinos procesoriaus komandos 7](#_Toc445212591)

[Puslapiavimo mechanizmas 8](#_Toc445212592)

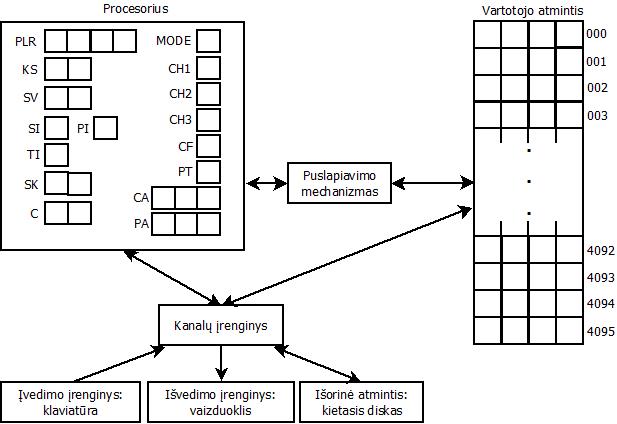
[Taimerio mechanizmas 9](#_Toc445212593)

[Procesoriaus algoritmas (Decode -> Execute) 9](#_Toc445212594)

[Pertraukimai 9](#_Toc445212595)

[Kanalų įrenginys 10](#_Toc445212596)

# Realios mašinos modelis



# Realios mašinos centrinis procesorius

Procesoriaus paskirtis - skaityti komandą iš atminties ir ją vykdyti (interpretuoti). Procesorius gali dirbti dviem režimais – **supervizoriaus** arba **vartotojo**. Šio procesoriaus registrai yra:

* **PLR** – 4 baitų puslapių lentelės registras, rodo kur saugoma puslapių lentelė (apie tai plačiau skyriuje „Puslapiavimo mechanizmas“).
* **KS** – 2 baitų komandų skaitliukas.
* **SV** – 2 baitų registras saugantis steko viršūnės žodžio indeksą.
* **SI** – 1 baito supervizorinių pertraukimų registras.
* **PI** – 1 baito programinių pertraukimų registras.
* **TI** – 1 baito taimerio pertraukimo registras.
* **MODE** – 1 baito registras, nusakantis esamą procesoriaus rėžimą. Jei registre saugomas 1, tai procesorius yra supervizoriaus režime, kitu atveju – vartotojo režime.
* **CH1** – 1 baito įvedimo įrenginio kanalo būsenos registras, rodo ar šis kanalas dabar užimtas ar ne. Jei CH1 = 1, tai užimtas, kitu atveju ne.
* **CH2** – 1 baito išvedimo įrenginio kanalo būsenos registras, rodo ar šis kanalas dabar užimtas ar ne. Jei CH2 = 1, tai užimtas, kitu atveju ne.
* **CH3** – 1 baito išorinės atminties įrenginio kanalo būsenos registras, rodo ar šis kanalas dabar užimtas ar ne. Jei CH3 = 1, tai užimtas, kitu atveju ne.
* **C** – 2 baitų registras, naudojamas kanalų įrenginyje, nusakantis kiek žodžių reikės kopijuoti.
* **CF** – 1 baito registras nurodantis iš kurios atminties (arba įrenginio) kopijuosime.
* **PT** – 1 baito registras nurodantis į kurią atmintį (arba įrenginį) kopijuosime.
* **CA** – 3 baitų registras nurodantis bloko, iš kurio kopijuosime, adresą.
* **PA** – 3 baitų registras nurodantis bloko, į kurį kopijuosime, adresą.
* **SK** – 2 baitų ciklo registras.

# Realios mašinos atmintys

**Vartotojo atmintis** skirta virtualių mašinų atmintims bei puslapių lentelėms laikyti. Mes apibrėšime vartotojo atmintį taip: lentelės dydis – 4096 žodžių po 4 baitus. 16 žodžių laikysime bloku. Taigi vartotojo atmintis yra 256 blokai, sunumeruotų nuo 0 iki 255, arba 4096 žodžių, sunumeruotų nuo 0 iki 4095.

**Išorinė atmintis** bus realizuota failu kietame diske. Tarkime laikysime kad faile yra 128 blokų arba 2048 žodžių. Procesorius darbą su atmintimis valdo naudodamas kanalų įrenginį.

# Realios mašinos procesoriaus komandos

Vartotojo ir supervizoriaus komandos:

* **MOD1** – MODE registrui priskiria 1 (supervizoriaus režimas)
* **MOD0** – MODE registrui priskiria 0 (vartotojo režimas)

Kanalų įrenginio komandos:

* **CH11** – CH1 įvedimo įrenginio registrui priskiria 1 (užimtas).
* **CH10** – CH1 įvedimo įrenginio registrui priskiria 0 (laisvas).
* **CH21** – CH2 išvadimo įrenginio registrui priskiria 1 (užimtas).
* **CH20** – CH2 išvedimo įrenginio registrui priskiria 0 (laisvas).
* **CH31** – CH3 išorinės atminties įrenginio registrui priskiria 1 (užimtas).
* **CH30** – CH3 išorinės atminties įrenginio registrui priskiria 1 (laisvas).
* **CF0** – CF registrui priskiria 0 (kopijavimas iš vartotojo atminties).
* **CF1** – CF registrui priskiria 1 (kopijavimas iš išorinės atminties).
* **CF2** – CF registrui priskiria 2 (kopijavimas iš įvedimo įrenginio).
* **PT0** – PT registrui priskiria 0 (kopijavimas į vartotojo atmintį).
* **PT1** – PT registrui priskiria 1 (kopijavimas į išorinę atmintį).
* **PT2** – PT registrui priskiria 2 (kopijavimas į išvedimo įrenginį).
* **Cxy** – C registrui priskimiama x reikšmė.
* **Fxyz** – CA registrui priskiria adresą (CA = 256\*x + 16\*y + z, čia x,y,z: 0 ≤ x,y,z ≤ 15).
* **Txyz** – PA registrui priskiria adresą (PA = 256\*x + 16\*y + z čia x,y,z: 0 ≤ x,y,z ≤ 15).

Pertraukimų komandos:

* **PI0** – Programinių pertraukimų registrui priskiria 0 (nėra pertraukimo).
* **PI1** – Programinių pertraukimų registrui priskiria 1 (atminties apsauga).
* **PI2** – Programinių pertraukimų registrui priskiria 2 (neleistinas operacijos kodas).
* **SI0** – Supervizorinių pertraukimų registrui priskiria 0 (nėra pertraukimo).
* **SI1** – Supervizorinių pertraukimų registrui priskiria 1 (vykdoma komanda READ).
* **SI2** – Supervizorinių pertraukimų registrui priskiria 2 (vykdoma komanda PRTN).
* **SI3** – Supervizorinių pertraukimų registrui priskiria 3 (komanda STOP).
* **TI1** – Taimerio registro reikšmę sumažina vienetu (TI = TI – 1).
* **TI3** – Taimerio registro reikšmę sumažina trejetu (TI = TI – 3).
* **TIxy** – Taimerio registrui priskiriama reikšmė xy (TI = xy).

Valdymo komandos:

* **Pxyz** – nesąlyginio valdymo perdavimo komanda. Valdymas perduodamas kodo sričiai nurodytam adresui. KS = 256\*x+16\*y+z (čia x,y,z: 0 ≤ x,y,z ≤ 15, komandos veikimas identiškas virtualios mašinos JPxy komandai).
* **Exyz** – jei steko viršūnėje yra 1 valdymas perduodamas adresu 256\*x+16\*y+z. IF( [SV] == 1) KS = 256\*x+16\*y+z; SV--; (čia x,y,z: 0 ≤ x,y,z ≤ 15, komandos veikimas identiškas virtualios mašinos JExy komandai).
* **Lxyz** – jei steko viršūnėje yra 0 valdymas perduodamas adresu 256\*x+16\*y+z. IF( [SV] == 0) KS = 256\*x+16\*y+z; SV--; (čia x,y,z: 0 ≤ x,y,z ≤ 15, komandos veikimas identiškas virtualios mašinos JLxy komandai).
* **Gxyz** – jei steko viršūnėje yra 2 valdymas perduodamas adresu 256\*x+16\*y+z. IF( [SV] == 2) KS = 256\*x+16\*y+z; SV--; (čia x,y,z: 0 ≤ x,y,z ≤ 15, komandos veikimas identiškas virtualios mašinos JGxy komandai).

Ciklo komandos:

* **Rxyz** – jei SK <> 0 then Pxyz; (komandos veikimas identiškas virtualios mašinos RExy komandai).
* **Kxyz** – SK = xyz; (komandos veikimas identiškas virtualios mašinos SKxy komandai).
* **SKDE** – SK=SK-1; (komanda identiška kaip ir virtualioje mašinoje).

Kanalų įrenginio komandos:

* **XCHG** – užpildo kanalo registrus atitinkamomis reikšmėmis ir atlieka duomenų apsikeitimą tarp atminčių (arba įrenginių).

Likusios realios mašinos komandos yra tokios pat, kaip ir virtualios mašinos komandos (išvardintos prie virtualios mašinos komandų).

# Virtualios mašinos modelis

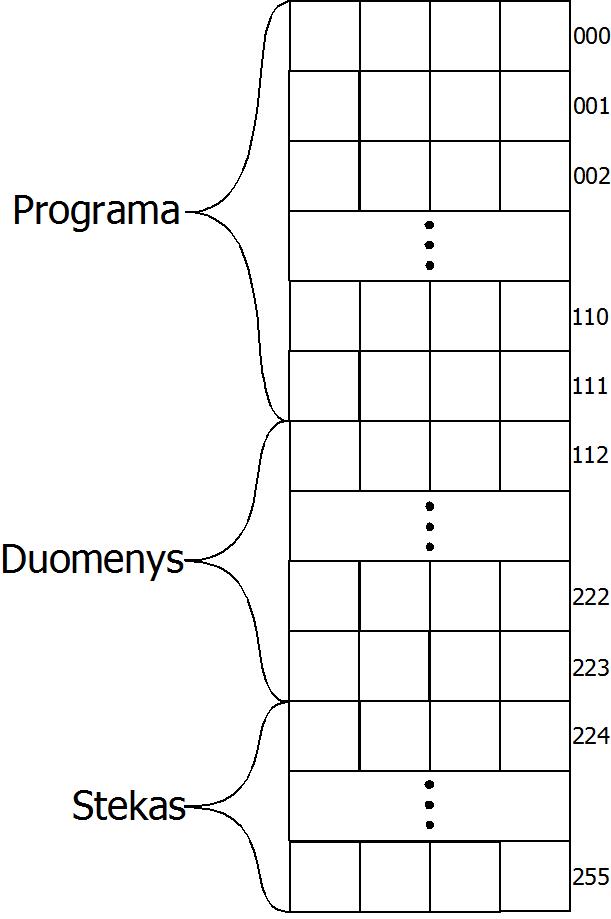
# Virtualios mašinos centrinis procesorius

Kaip buvo matyti iš schemos, virtualios mašinos centrinis procesorius yra daug paprastesnis nei realios. Pagrindinė virtualios mašinos centrinio procesoriaus paskirtis – vykdyti virtualioje atmintyje esančią programą. Virtualios mašinos procesorius turi du registrus:

* KS – 2 baitų komandų skaitliukas.
* SV – 2 baitų registras saugantis steko viršūnės žodžio indeksą.
* SK - 2 baitų ciklo registras.

# Virtualios mašinos atmintis

Kiekvienai virtualiai mašinai skiriama iki 16 blokų vartotojo atminties. Tuose šešiolikoje blokų (256 žodžiai po 4 baitus) turi tilpti programos kodas, duomenys ir stekas. 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.

Kaip ir minėta, virtualios mašinos atmintis bus skirstoma į tris dalis, į kurias bus įkeliamos atitinkamos programos dalys:

1. Programa – 7 blokai, nuo 0 iki 6 (112 žodžių).

* Registro KS adresas 00;
* Registro SV adresas 01;
* Registro SK adresas 02.

1. Duomenys – 7 blokai, nuo 7 iki 13 (112 žodžiai).
2. Stekas – 2 blokai, nuo 14 iki 15 (32 žodžiai).

# Virtualios mašinos procesoriaus komandos

Aritmetinės komandos:

* **ADD** – sudeda du viršutinius steko elementus. Rezultatą padeda į steko viršūnę ir steko viršūnės registro reikšmę sumažina vienetu.

[SV – 1] = [SV – 1] + [SV]; SV--;

* **SUB** – atima steko viršūnėje esantį elementą iš antro nuo viršaus elemento. Rezultatą padeda į steko viršūnę, ir steko viršūnės registro reikšmę sumažina vienetu. [SV – 1] = [SV – 1] – [SV]; SV--;
* **MUL** – sudaugina du viršutinius steko elementus. Rezultatą padeda į steko viršūnę ir steko viršūnės registro reikšmę sumažina vienetu. [SV – 1] = [SV – 1] \* [SV]; SV--;
* **DIV** – padalina (gauna sveikąją dalį) antrą nuo viršaus steke esantį elementą iš viršūnėje esančio elemento. Rezultatą padeda į steko viršūnę ir steko viršūnės registro reikšmę sumažina vienetu. [SV-1] = [SV – 1] / [SV]; SV--;

Palyginimo komandos:

* **CMP** –lyginasteko viršūnėje esančius du žodžius ir rezultatą padeda į steko viršūnę. 1- jei lygūs, 0 – jei viršutinis elementas mažesnis, 2 – jei viršutinis elementas didesnis. [SV + 1] = 0, jei [SV – 1] > [SV]; SV--; [SV + 1] = 1, jei [SV – 1] < [SV]; SV--;

[SV + 1] = 2, jei [SV – 1] == [SV]; SV--;

Darbo su duomenimis / steko komandos:

* **LDxy** – į steko viršūnę užkrauna reikšmę iš duomenų srities, adresu 16 \* x + y; SV--; (čia x: 7 ≤ x ≤ 14).
* **PTxy** – steko viršūnėje esantį žodį deda į duomenų sritį nurodytu adresu SV++; 16 \* x + y; (čia x: 7 ≤ x ≤ 14).
* **PUxy** – x kaip skaičių patalpina į steko viršūnę SV++; [SV] = xy;

Valdymo komandos:

* **JPxy** – nesąlyginio valdymo perdavimo komanda. Valdymas perduodamas kodo sričiai nurodytam adresui. KS = 16\*x+y
* **JExy** – jei steko viršūnėje yra 1 valdymas perduodamas adresu 16\*x + y. IF( [SV] == 1) KS = 16 \* x + y; SV--;
* **JLxy** – jei steko viršūnėje yra 0 valdymas perduodamas adresu 16\*x + y. IF( [SV] == 0) KS = 16 \* x + y; SV--;
* **JGxy** – jei steko viršūnėje yra 2 valdymas perduodamas adresu 16\*x + y. IF( [SV] == 2) KS = 16 \* x + y; SV--;
* **STOP** – programos sustojimo komanda.

Ciklo komandos:

* **RExy** – jei SK <> 0 then JPxy;
* **SKxy** – SK = xy;
* **SKDE** – SK=SK-1;

Procedūrų komandos:

* **PUKS** – SV++; [SV] := KS;
* **POKS** – KS = [SV]; SV--;
* **CAxy** – PUKS; JPxy;
* **RET** – POKS;

Įvedimo bei švedimo komandos:

* **PRTN** – steko viršūnėje esantį žodį traktuoja kaip skaitinę reikšmę ir išveda į išvedimo įrenginį.
* **READ** – nuskaito vartotojo įvedimą kaip skaičių ir įrašo į steko viršūnę.

# Puslapiavimo mechanizmas

Realios mašinos vartotojo atmintis siekia 256 blokų. Kiekvienai naujai sukurtai virtualiai mašinai reikia skirti 16 blokų iš tų 256. Jie gali būti parinkti bet kokiu būdu. Klausimas: kaip virtuali mašina gali sužinoti kokio nors jai priklausančio bloko realų adresą? Tam naudosime puslapiavimo mechanizmą.

Puslapiavimo mechanizmo esmė: sakykime, kuriama nauja virtuali mašina. Jai reikia šešiolikos blokų atminties. Mes parinkome blokus su numeriais: 3, 4, 10, 20, 35, 36, 40, 60, 69, 70, 71, 72, 73, 79, 100, 111. Šiais blokais naudosis virtuali mašina. Pati virtuali mašina šiuos blokus mato sunumeruotus nuo 0 iki 15, t.y. 3 blokas jai yra nulinis, 4 blokas jai yra antras, o 111 blokas – penkioliktas. Kaip išlaikyti sąryšius tarp realių ir virtualių blokų adresų? Naudosime puslapių lentelę. Puslapių lentelė – tai vienas blokas (t.y. 16 žodžių). Kiekvieno žodžio eilės numeris atitiks virtualios mašinos bloko numerį, ir jame (žodyje) bus laikomas realus to bloko numeris. Pavyzdžiui, prieš tai pateikto pavyzdžio puslapių lentelė bus tokia:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Virtualaus  bloko  numeris | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| Realaus  bloko  numeris | 3 | 4 | 10 | 20 | 35 | 36 | 40 | 60 | 69 | 70 | 71 | 72 | 73 | 79 | 100 | 111 |

PLR yra 4 baitų ir simboliškai žymėsime taip a1,a2,a3,a4. Virtualų adresą žymėsime x1x2.

* a1 – nenaudojamas.
* a2 – nenaudojamas.
* 16 \* a2 + a3 – puslapių lentelės bloko numeris vartotojo atmintyje.
* 16 \* (16 \* a2 + a3) – puslapių lentelės bloko adresas.
* 16 \* (16 \* a2 + a3) + x1 – bloko x1 adresas puslapių lentelėje. Jame saugomas bloko į kurį atvaizduotas yra x1 blokas virtualioje mašinoje.
* 16 \* [16 \* (16 \* a2 + a3) + x1] – virtualios mašinos bloko x1 realus bloko adresas.
* 16 \* [16 \* (16 \* a2 + a3) + x1] + x1 – realus adresas atitinkantis virtualų adresą x1x2.

# Taimerio mechanizmas

Laikysime, kad įvedimo/išvedimo instrukcijos atliekamos per 2 taktus, visos kitos per 1 taktą. Kiekviena komanda turi jai skirtą laiko tarpą, kurį ji vykdoma.Tuo yra užtikrinama, kad komanda nebus pertraukta ankščiau laiko. Tarkime registui TI yra nustatoma reikšmė = 9. Tada kiekvieną kartą, kai įvykdo makromanda registro TI reikšmė yra sumažinama atitnkamai nuo komandos duotojo vykdymo laiko. Kai TI registro reikšmė tampa lygi nuliui, mikrokomanda Test() aptinka taimerio pertraukimą. TI reikšmę galima nustatyti ar pakeisti supervizoriaus rėžime.

# Procesoriaus algoritmas (Decode -> Execute)

1. Jei registras MODE = 0, tada dirbame vartotojo režimu, MODE = 1 reikštų darbą supervisoriaus režime.
2. Galimos kelios situacijos vykdant instrukciją KS registre esančiu adresu:
   1. Komanda atpažystama ir įvykdoma sėkmingai. TI sumažėja atitinkamai pagal komandą.
   2. Komanda neatpažįstama, įvyksta pertraukimas, identifikuojantis neteisingą komandą (apie pertraukimus žemiau).
   3. TI registrui sumažėjus iki 0, įvyksta taimerio pertraukimas
3. KS registro reikšmė padidėja vienetu.
4. Vartotojo rėžime su funkcija test() tikrinama ar įvyko pertraukimų.
5. Tęsiame darbą nuo 2.

# Pertraukimai

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.

Ar įvyko pertraukimas tikrinama po kiekvienos komandos. Kad pertraukimas įvyko sužinoma iš pertraukimo registrų reikšmių. Jei SI = 0, PI = 0 ir TI <> 0, tai pertraukimas neįvyksta.

Yra trys pertraukimų rūšys:

**PI** – programiniai:

* PI = 1 (atminties apsauga);
* PI = 2 (neleistinas operacijos kodas).

**SI** – supervizoriaus:

* SI = 1 (vykdoma komanda READ);
* SI = 2 (vykdoma komanda PRTN);
* SI = 3 (komanda STOP).

**TI** – taimerio:

* TI = 0 bus fiksuojamas taimerio pertraukimas.

Kiekvieną kartą komandų interpretatoriui įvykdžius programą, kviečiama komanda test(), kuri apklausia registrus, ir, jei kilo pertraukimas, gražina informaciją apie tai. Komanda test() gali būti tokia:

IF(((PI + SI) > 0) OR (TI = 0)) THEN return 1 ELSE return 0;

Nustačius įvykusį pertraukimą (t.y. test() <> 0) virtualios mašinos procesoriaus registrų reikšmės yra išsaugomos, procesorius perjungiamas į supervizoriaus režimą, nustatomas pertraukimo pobūdis ir kviečiama pertraukimą apdorosianti paprogramė. Vėliau, valdymas grįžta į virtualią mašiną, registrai atstatomi, procesorius perjungiamas į vartotojo režimą, ir operacinė sistema sprendžia, ką daryti toliau (test() komandos adresas yra 0-tame vartotojo atminties bloke).

# Kanalų įrenginys

Kanalų įrenginys leidžia procesoriui dirbti su įvairiomis atmintimis ir išoriniais įrenginiais. Veiksmai su kanalų įrenginiu atliekami tik supervizoriaus rėžime. Kanalų įrenginys turi keletą nuosavų registrų:

C – 2 baitų, nurodo kiek žodžių kopijuosime. Jei C=00, tai kopijuosime visą bloką.

CA – 3 baitų registras. Bloko iš kurio kopijuosime adresas.

PA – 3 baitų registras. Bloko į kurį kopijuosime adresas.

CF – 1 baito registras. Įrenginys, iš kurio kopijuosime

Jei CF = 0, tai vartotojo atmintis

Jei CF = 1, tai išorinė atmintis atmintis

Jei CF = 2, tai įvedimo įrenginys.

PT – 1 baito registras. Įrenginys, į kurį kopijuosime

Jei PT = 0, tai vartotojo atmintis

Jei PT = 1, tai išorinė atmintis atmintis

Jei PT = 2, tai išvedimo įrenginys.

Kanalų įrenginys turi specialią komandą XCHG(C, CA, PA, CF, PT), kuri užpildo kanalo registrus atitinkamomis reikšmėmis ir atlieka duomenų apsikeitimą tarp atminčių. Kad vienu metu kelios virtualios mašinos nesinaudotų tuo pačiu kanalu, procesoriuje yra specialūs registrai CH1, CH2, CH3, kurie rodo, ar kanalai yra užimti. Kol kanalai užimti, jais naudotis pageidaujančios virtualios mašinos yra užsiblokavusios, centrinis procesorius laukia kanalo atlaisvinimo.