Virtualios ir realios mašinos projektas

Atilko:
Irmantas Varačisnkas,
Karolis Šimaitis,
Miglė Augustinaitė

Turinys

1 Projekto sąlygos **2**

2 Realios mašinos modelis **3**
2.1 Realios mašinos brėžinys . . . 3
2.2 Realios mašinos registrai . . . 3
2.3 Taimerio mechanizmas . . . 4
2.4 Supervizoriniai pertraukimai . . . 4
2.5 Programiniai pertraukimai . . . 5
2.6 Realios mašinos rėžimai . . . 5
2.7 Puslapių transliacija . . . 5
2.8 Įvedimas ir išvedimas . . . 5
2.9 Atminties įrenginiai . . . 5
2.10 Išorinė atmintis . . . 6

3 Virtualios mašinos modelis **7**
3.1 Virtualios mašinos schema . . . 7
3.2 Virtualios mašinos samprata . . . 7
3.3 Virtualios mašinos loginiai komponentai . . . 7
3.3.1 Virtualios mašinos atmintis . . . 7
3.3.2 Virtualios mašinos procesorius . . . 7
3.3.3 Virtualios mašinos komandų sistema . . . 8

4 Programos pavyzdys **11**

5 Procesai ir jų primityvai **12**
5.1 Procesų būsenos . . . 12
5.2 Būsenų sąryšiai . . . 12
5.3 Proceso deskriptorius . . . 12
5.4 Procesų sąrašai . . . 13
5.5 Prioritetų sudarymo tvarka . . . 13
5.6 Sisteminiai procesai . . . 13
5.7 Procesų planuotojas (Job_Governor) . . . 13
5.8 Procesų planuotojo primityvai . . . 14

6 Resursai **15**
6.1 Resursų deskriptorius . . . 15
6.2 Resursų tipai ir magic number . . . 15
6.3 Resursų paskirstytojas . . . 16
6.4 Resursų primityvai . . . 16

1 Projekto sąlygos

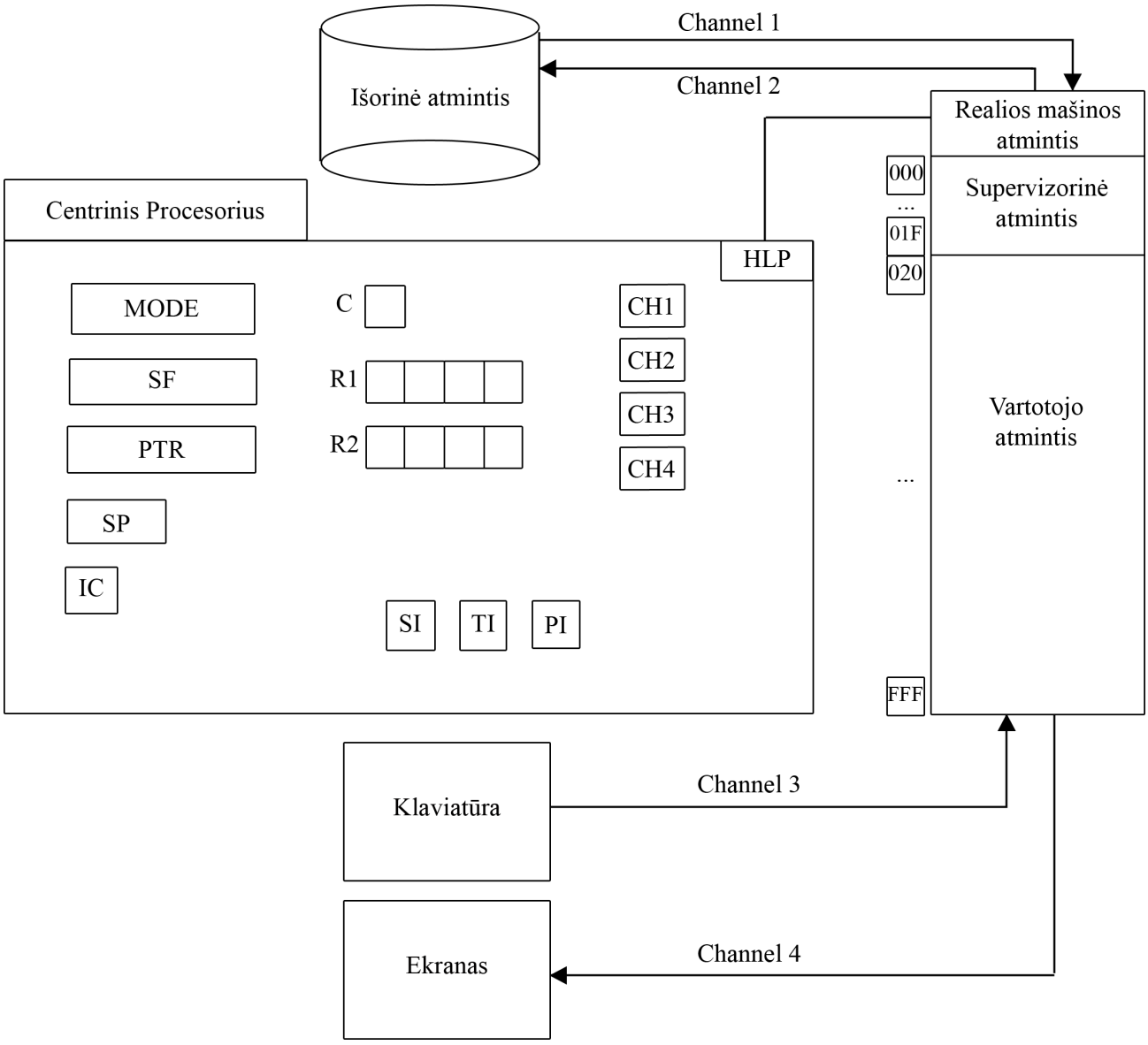
Projektuojama interaktyvi OS.

Virtualios mašinos procesoriaus komandos operuoja su duomenimis, esančiais registruose ir ar atmintyje. Yra komandos duomenų persiuntimui iš atminties į registrus ir atvirkščiai, aritmetinės (sudėties, atimties, daugybos, dalybos, palyginimo), sąlyginio ir besąlyginio valdymo perdavimo, įvedimo, išvedimo, darbo su failais (atidarymo, skaitymo, rašymo, uždarymo, sunaikinimo) ir programos pabaigos komandos. Registrai yra tokie: komandų skaitiklis, bent du bendrosios paskirties registrai, požymių registras (požymius formuoja aritmetinės, o į juos reaguoja sąlyginio valdymo perdavimo komandos). Atminties dydis yra 16 blokų po 16 žodžių (žodžio ilgį pasirinkite patys).

Realios mašinos procesorius gali dirbti dviem režimais: vartotojo ir supervizoriaus. Virtualios mašinos atmintis atvaizduojama į vartotojo atmintį naudojant puslapių transliaciją. Yra taimeris, kas tam tikrą laiko intervalą generuojantis pertraukimus. Įvedimui naudojama klaviatūra, išvedimui - ekranas. Yra išorinės atminties įrenginys - kietasis diskas. Vartotojas, dirbantis su sistema, programas paleidžia interaktyviai, surinkdamas atitinkamą komandą. Laikoma, kad vartotojo programos yra realios mašinos kietajame diske, į kurį jos patalpinamos „išorinėmis", modelio, o ne projektuojamos OS, priemonėmis.

2 Realios mašinos modelis

2.1 Realios mašinos brėžinys



2.2 Realios mašinos registrai

1. HLP - bet kuris aukšto lygio kalbos procesorius. Vartotojo režime HLP vykdo užduoties programą.

2. MODE - realios mašinos rėžimo registras. Dydis - 1 baitas. Jei reikšmė 0, dirbama supervizoriaus rėžimu, jei reikšmė nėra 0, tada dirbama vartotojo rėžimu.

3. SF - požymių registras. Dydis - 1 baitas. Parodo procesoriaus būseną po aritmetinio veiksmo.
   Požymių registro struktūra: X X X X X CF ZF OF

   - X - nenaudojamas.

- CF - carry flag. Rezultatas netilpo į skaičiaus be ženklo rėžius.
- ZF - zero flag. Rezultatas yra nulis.
- OF - overflow flag. Rezultatas netilpo į skaičiaus su ženklu rėžius.

4. PTR - puslapių lentelės registras. Dydis - 2 baitai. Vyresnysis baitas saugo puslapių lentelės bloko numerį, jaunesnysis baitas saugo puslapių lentelės dydį.
5. SP - steko rodyklė. Dydis - 1 baitas. Rodo į virtualios mašinos steko viršūnę.
6. IC - instrukcijų skaitliukas. Dydis - 1 baitas. Rodo virtualios mašinos einamąją instrukciją.
7. C - loginis trigeris. Dydis - 1 baitas. 0 yra false, visa kita yra true.
8. R1, R2 - bendros paskirties registrai. Dydis - po 4 baitus. Skirti atlikti komandoms.
9. Kanalų registrai. Dydis - po 1 baitą.
    - CH1 - registras rodantis ar yra atliekamas persiuntimas iš išorinės atminties į realią atmintį arba atvirkščiai.
    - CH2 - registras rodantis ar atliekamas įvedimas iš klaviatūros.
    - CH3 - registras rodantis ar atliekamas išvedimas į ekraną.
10. SI - supervizoriaus pertraukimų registras. Dydis - 1 baitas.
11. PI - programinių pertraukimų registras. Dydis - 1 baitas.
12. TI - taimerio registras. Dydis - 1 baitas.

2.3 Taimerio mechanizmas

Po kiekvienos įvykdytos komandos taimerio registras yra sumažinamas vienetu. Pradinę taimerio reikšmę gali nustatyti vartotojas, tačiau numatytoji reikšmė yra 10.

Kai šio registro reikšmė pasiekia nulį, iškviečiamas supervizorinis taimerio pertraukimas. Įvykdžius taimerio pertraukimo apdorojimo procedūrą, taimerio registro reikšmė vėl tampa 10.

2.4 Supervizoriniai pertraukimai

Supervizoriniai pertraukimai yra iškviečiami tuomet, kai SI registras nėra lygus nuliui arba TI registras yra lygus nuliui.

Pertraukimai pagal SI registro reikšmę:

- 0 - jokio pertraukimo.
- 1 - taimerio pertraukimas.
- 2 - HALT. Programa baigė darbą.

2.5 Programiniai pertraukimai

Programiniai pertraukimai yra iškviečiami tuomet, kai PI registras nėra lygus nuliui.
Pertraukimai pagal PI registro reikšmę:

- 0 - jokio pertraukimo.
- 1 - Undefined operation code. Neteisingas operacijos kodas.
- 2 - Undefined address. Neteisingas adresas.
- 3 - Division by zero. Dalyba iš nulio.

2.6 Realios mašinos rėžimai

Reali mašina gali vykdyti darbą dvejais rėžimais: supervizoriaus arba vartotojo.

Supervizoriaus rėžimu programa bus vykdoma nuo pradžios iki galo, ją nutraukti gali tik pertraukimai. Visi realios mašinos resusrsai yra prieinami.

Vortotojo rėžimu programa gali vykti kelios programos vienu metu, po kiekvieno veiksmo taimeris mažinamas ir kai jis pasiekia nulį, toliau procesoriaus darbą nustato prioritetų automatas. Prieinama atmintis yra tik ta, kurią išskyrė procesorius.

2.7 Puslapių transliacija

Puslapių transliavimo mechanizmas naudojamas, tam, kad galėtumę susieti virtualią ir realią atmintis. registre PTR bus laikomas puslapiavimo lentelės bloko adresas, kurio kiekviename žodyje bus laikomas virtualios mašinos puslapio takelio adresas. a_0 ir a_1 naudojami naudojamas gauti puslapių leneltės adresą, naudojama formulė yra tokia:
$10 * a_0 + a_1$

Dviženklis adresas x_0, x_1 realiojoje mašinoje apskaičiuojamas taip:
$10 * [10 * (10 * a_0 + a_1) + x_0] + x_1$

2.8 Įvedimas ir išvedimas

Kadangi projektuojama OS yra interaktyvi, vartotojas gali įvesti komandas ir/ar duomenis ir taip reguliuoti realios mašinos darbą ir matyti šių komandų rezultatus.

Įvedimo įrenginys - klaviatūra. Suvedama komanda arba jos duomenys ir paspaudžiamas mygtukas ENTER, kad būtų paleidžiamas arba pratęsiamas jos vykdymas.

Išvedimo įrenginys - ekranas. Įvykdytų komandų rezultatai gali būti išvedami į ekraną.

2.9 Atminties įrenginiai

Išorinė atmintis - kietasis diskas. Jame duomenys bus išsaugoti net ir išjungus realią mašiną.

Realios mašinos atmintis - atmintis esanti realioje mašinoje. Ji suskirstyta į 16 blokų po 16 žodžių, vieno žodžio dydis yra 32 bitai. Ją toliau skirstome į supervizorinę ir vartotojo atmintį:

- **Supervizorinė atmintis** - 2 blokai išskirti realios mašinos atminties pradžioje. Jie nėra prieinami vartotojui.

- **Vartotojo atmintis** - Visa likusi realios mašinos atmintis. Ją galima skirstyti virtualioms mašinoms.

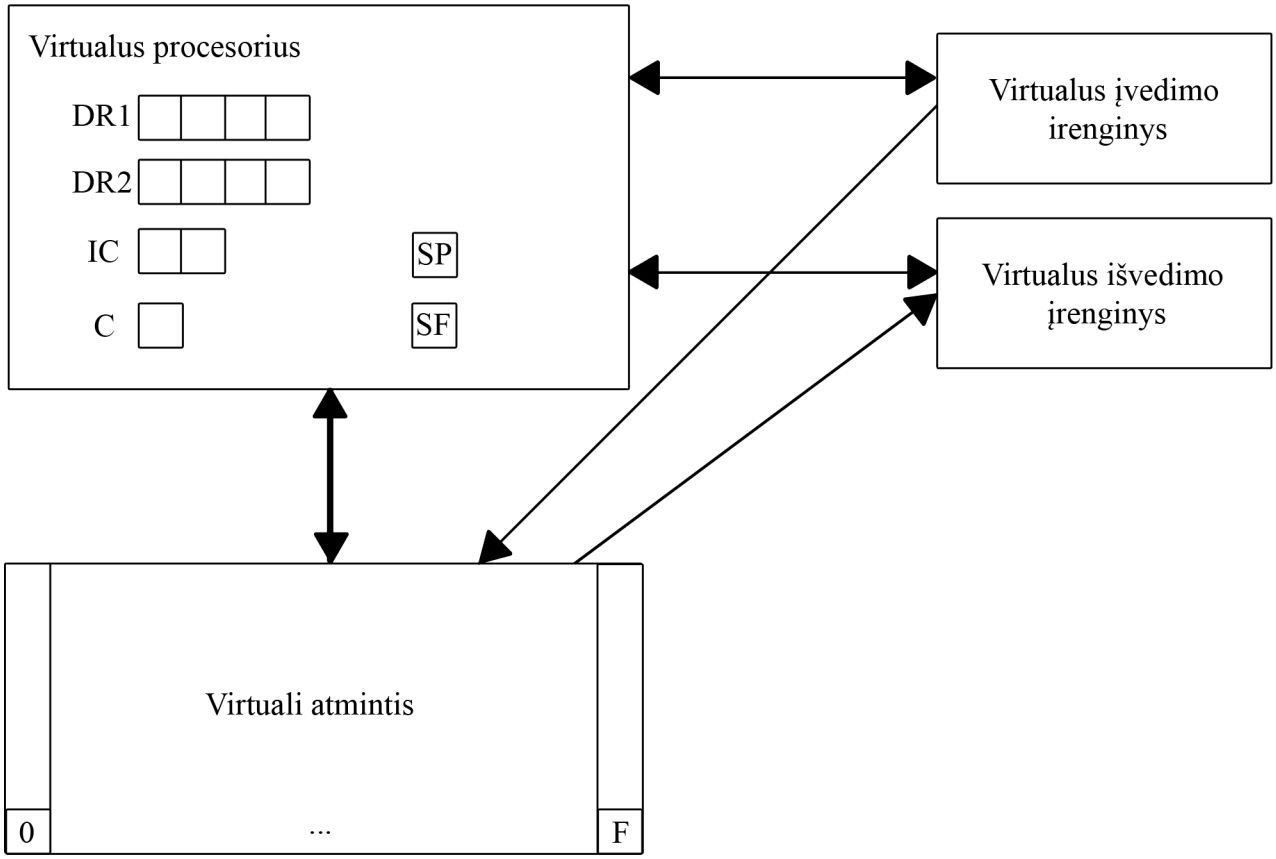
2.10 Išorinė atmintis

Išorinė atmintis bus realizuojama failu kietajame diske. Darbą su ja vykdys HLP.
Išorinė atmintis yra 256 blokai po 16 takelių po 16 žodžių po 4 baitus, t.y. 262144 baitų.
Pirmojo bloko struktūra:
Šis blokas yra skirtas aprašyti kokioms programoms yra duoti kiti blokai. Pirmasis žodis aprašo programą esančią pirmąjame bloke, antrasis - antrajame ir t.t. Vieno žodžio struktūra: $x_1x_2x_3x_4$
x_1x_2 yra skirti failo pavadinimui, x_3 yra skirtas nusakyti, ar failas atidarytas, ar ne(0 jei neatidarytas, 1, jei atidarytas), o x_4 yra skirti nusakyti kelinta išskaidyto failo dalis tai yra.
Kitų blokų struktūra:

- Pirmasis žodis yra specialus ženklas susidedantis iš keturių dolerių($$$$).
- Antrasis žodis yra toks pat, kaip ir šį bloką aprašantis žodis, esantis pimajame bloke.(Pvz: Jei esame trečiajame bloke, tada šio bloko antrasis žodis sutampa su pirmojo bloko trečiuoju).
- Visi kiti žodžiai skirti duomenims.

3 Virtualios mašinos modelis

3.1 Virtualios mašinos schema



3.2 Virtualios mašinos samprata

Virtuali mašina - tai tarsi realios mašinos kopija. Virtuali mašina yra sudaryta iš realios mašinos komponenčių, tokių kaip procesorius, atmintis, įvedimo/išvedimo įrenginiai. Jiems yra suteikiama paprastesnė nei reali vartotojo sąsaja.

3.3 Virtualios mašinos loginiai komponentai

3.3.1 Virtualios mašinos atmintis

Virtualiai mašinai skirta 16 blokų po 16 žodžių, vieno žodžio dydis yra 32 bitai.
Vienas blokas yra skirtas virtualios mašinos stekui.

3.3.2 Virtualios mašinos procesorius

Virtualios mašinos registrai:

1. Duomenų registrai:

    - DR1 - Data Register. Dydis - 4 baitai. Naudojamas duomenų pakrovimui į jį iš atminties ir iš jo į atmintį. Taip pat gali būti naudojamas operacijose.
    - DR2 - Data Register. Dydis - 4 baitai. Naudojamas duomenų pakrovimui į jį iš atminties ir iš jo į atmintį. Taip pat gali būti naudojamas operacijose.

2. Segmentų registrai:
    - CS - Code Segment. Rodyklė rodanti į kodo segmentą atmintyje.
    - DS - Data Segment. Rodyklė rodanti į duomenų segmentą atmintyje.
    - ST - Stack Segment. Rodyklė rodanti į steko segmentą atmintyje.
3. Nuorodų registrai:
    - IC - Instruction Counter. Saugoma einamosios komandos žodžio indeksas.
    - SP - Stack Pointer. Saugomas steko viršūnės žodžio indeksas.
4. Loginiai registrai:
    - SF - Status Flag. Rodo procesoriaus būseną po aritmetinio veiksmo.

3.3.3 Virtualios mašinos komandų sistema

Vieną komandą sudaro 4 baitai, tačiau nebūtinai visi baita privalo būti naudojami.

1. Aritmetinės komandos
    - ADRR - sudeda registrus DR1 ir DR2. Rezultatas laikomas registre R1. Gali pakeisti visus flag'us.
    - ADxy - sudeda registrą DR1 ir žodį esantį duomenų segmente adresu 10*x + y. Rezultatas laikomas registre DR1. Gali paskeisti visus flag'us.
    - SBRR - atema DR2 reikšmę iš DR1. Rezultatas laikomas registre R1. Gali pakeisti visus flag'us.
    - SBxy - atema žodį esantį duomenų segmente adresu 10*x + y iš registro DR1. Rezultatas laikomas registre DR1. Gali pakeisti visus flag'us.
    - MLRR - sudaugina registrus DR1 ir DR2. Rezultatas laikomas registre DR1. Gali pakeisti visus flag'us.
    - MLxy - sudaugina registrą DR1 iš žodžio esančio duomenų segmente adresu 10*x + y. Gali pakeisti visus flag'us.
    - DVRR - padalina DR1 iš DR2. Dalybos rezultatas laikomas registre DR1, modulio rezultatas laikomas registre DR2. Gali pakeisti visus flag'us.
    - DVxy - padalina DR1 iš žodžio esančio duomenų segmente adresu 10*x - y.
2. Loginės komandos
    - AND - įvykdo AND operaciją tarp DR1 ir DR2. Rezultatas laikomas registre R1. Gali pakeisti ZF.
    - OR - įvykdo OR operaciją tarp DR1 ir DR2. Rezultatas laikomas registre DR1. Gali pakeisti ZF.
    - XOR - įvykdo XOR operaciją tarp DR1 ir DR2. Rezultatas laikomas registre DR1. Gali pakeisti ZF.
    - NOT - įvykdo NOT operaciją DR1 registrui. Rezultatas laikomas registre DR1. Gali pakeisti ZF.

3. Palyginimo komandos

    - CMP - palygina registrus DR1 ir DR2. Jei elementai lygūs, tada ZF = 1. Jei višutinis elementas didesnis, tada CF = 0 ir ZF = 0. Jei viršutinis elementas mažesnis, CF = 1.

4. Darbo su duomenimis komandos

    - LWxy - į DR1 įdedamas žodis esantis duomenų segmente adresu 10 * x + y.
    - SWxy - į duomenų segmentą, adresu 10 * x + y įdedama DR1 reikšmė.
    - MOV1 - į DR2 įdedama DR1 reikšmė.
    - MOV2 - į DR1 įdedama DR2 reikšmė.
    - PRNT - iškviečiamas pertraukimas, spausdinantis eilutę į ekraną. Eilutės adresas nurodytas registre DR1 viršūnėje, eilutės ilgis nurodytas registre DR2
    - PRNS - iškviečiamas pertraukimas, DR1 reikšmę kaip skaičių į ekraną.

5. Darbo su steku komandos

    - PUSH - steko viršūnės rodyklė padidinama vientu ir į steko viršūnę idedama DR1 reikšmė.
    - POP - į DR1 įdedamas žodis esantis steko viršūnėje ir steko virūnės rodyklė sumažinama vienetu.

6. Valdymo perdavimo komandos

    - JMxy - besąlyginio valdymo perdavimo komanda. Valdymas perduodamas kodo segmento žodžiui 10 * x + y, t.y. PC = 10 * x + y.
    - JExy - sąlyginio valdymo perdavimo komanda. Jei ZF = 1 (elementai yra lygūs), tada valdymas perduodamas kodo segmento žodžiui 10 * x + y.
    - JAxy - sąlyginio valdymo perdavimo komanda. Jei CF = 0 ir ZF = 0 (virūnės elementas yra didesnis), tada valdymas perduodamas kodo segmento žodžiui 10 * x + y.
    - JLxy - sąlyginio valdymo perdavimo komanda. Jei CF = 1 (viršūnės elementas yra mažesnis), tada valdymas perduodamas kodo segmento žodžiui 10 * x + y.

7. Failų rašymo, skaitymo komandos

    - FOxy - Atidaromas failas esantis kietajame diske, failo pavadinimas yra xy. Failo handleris yra grąžinamas į DR1.
    - FC - Uždaromas failas, DR1 yra šio failo handleris.
    - FD - Ištrinamas failas, DR1 yra šio failo handleris.
    - FRxy - DR1 yra failo handleris. 10*x + y nurodo vietą, į kurią rašysim duomenų segmente. DR2 nurodo adresą iš kurio skaitysim.
    - FWxy - DR1 yra failo handleris. 10*x + y nurodo vietą, iš kurios rašysim iš duomenų segmento, DR2 nurodo adresą, kur įrašysim į failą.

8. Darbo pabaigos komanda

- HALT - parodo programos pabaigą.

Kodo struktūra turi būti tokia:
DATASEG
...
CODESEG
...
HALT

DATASEG viduje galima įdėti duomenis į duomenų segmentą naudojant DW komandą.

4 Programos pavyzdys

Mūsų programos pavyzdys atliks aritmetinius veiksmus ir juos išspausdins į ekraną. Programa skaičiuos šį reiškinį: $45^2 + 13 * 4 - 3$

```
DATASEG
DW 45
DW 13
DW 4
DW 3
CODESEG
LW01
MOV1
MLRR
PUSH
LW02
MOV1
LW03
MLRR
MOV1
POP
ADRR
SB04
PRNS
HALT
```

5 Procesai ir jų primityvai

5.1 Procesų būsenos

- **READY** - procesas, laukiantis procesoriaus.
- **STOPPED** - procesas, sustabdytas kito proceso.
- **BLOCKED** - procesas, prašantis resurso.
- **RUNNING** - vykdomas procesas.

5.2 Būsenų sąryšiai

1. READY -> RUNNING - paskisrtytojas paskiria procesorių procesui.
2. READY -> READY-STOPPED - pasiruošęs procesas buvo sustabdytas einamojo proceso.
3. READY-STOPPED -> READY - einamasis procesas ”nuima” būseną sustabdytas, tačiau jis laukia procesoriaus.
4. RUNNING -> BLOCKED - procesas užblokuojamas, kai paprašo resurso.
5. RUNNING -> READY - iš proceso ”atimamas” procesorius ir jam netrūksta resursų.
6. BLOCKED -> BLOCKED-STOPPED - einamasis procesas sustabdo blokuotą procesą.
7. BLOCKED -> READY - procesas gauna reikiamą resursą.
8. BLOCKED-STOPPED -> READY-STOPPED - gauna resursą, tačiau procesas vis dar sustabdytas.
9. BLOCKED-STOPPED -> BLOCKED - einamasis procesas ”nuima” būseną sustabdytas, tačiau jam vis dar reikia resurso.

5.3 Proceso deskriptorius

- PID - proceso vidinis vardas.
- PPID - proceso tėvo vidinis vardas.
- STATE - proceso būsena.
- PTR - puslapiavimo lentelės adresas.
- RES - laukiamų resursų sąrašas.
- REG - registrai (R1,R2,SF,IC) .
- CRES - proceso sukurtų resursų sąrašas.

5.4 Procesų sąrašai

1. **Procesų prioritetų sąrašas** - nurodomas proceso vidinis vardas ir jo prioritetas.
2. **Blokuotų procesų sąrašas** - sąrašas užblokuotų procesų.
3. **Pasiruošusių procesų sąrašas** - sąrašas pasiruošusių procesų.
4. **Resursų sąrašas** - nurodomas resurso vidinis vardas.

5.5 Prioritetų sudarymo tvarka

Prioritetų reikšmės bus skiriamos nuo 0 iki 100. Sisteminiai procesai turės didesnį prioritetą, nei vartotojo. Kiekvieną kartą procesui gavus procesorių, jo prioritetas sumažinamas vienetu. /*ar yra apatinės ribos sisteminiams procesams?*/ Vartotojas turi galimybę nustatyti prioriteto reikšmę rankiniu būdu (load funkcijos metu).

5.6 Sisteminiai procesai

Sisteminiai procesai sukuriami paleidžiant sistemą, o sunaikinami tik sistemai baigus darbą. Jie yra pasiruošę dirbti visą sistemos gyvavimo laiką.

- **Start_Stop** - Visų procesų tėvinis procesas; kuria ir naikina sisteminius procesus ir resursus. Visalaiką užsiblokavęs, kol dirba OS. PID - 1.
- **Job_Governor** - procesų paskirstytojas, tvarkantis virtualių mašinų darbą. PID - 2.
- **Loader** - užkrauna procesą iš išorinės atminties į vidinę atmintį. PID - 3.
- **Chan_1_Device** - perrašo duomenis tarp išorinės atminties ir vidinės atminties. PID - 4.
- **Interrupt** - procesas, kuris, įvykus pertraukimui, jį identifikuoja ir siunčia pranešimą Job_Governor. PID - 5.
- **Get_Put_Data** - procesas, dirbantis su duomenimis vidinėje atmintyje. PID - 6 .
- **Chan_2_Device** - darbui su įvesties srautais. PID - 7
- **Chan_3_Device** - darbui su išvesties srautais. PID - 8.
- **Process_Killer** - ištrina programą ir jos sukurtus resursus iš vidinės atminties. PID - 9.
- **Resource_Manager** - resursų paskirstytojas. PID - 10.

5.7 Procesų planuotojas (Job_Governor)

Procesų planuotojas iškviečiamas kiekvieną kartą, kai iššaukiamas timer'io pertraukimas. Taip pat jis gali būti kviečiamas procesui baigus darbą arba jam užsiblokavus. Iškvietus planuotoją, imamas kitas READY būsenos procesas pagal didžiausią prioritetą.

5.8 Procesų planuotojo primityvai

1. **Kurti procesą** - šiam primityvui perduodama nuoroda į jo tėvą, pradinė proceso būsena, jo prioritetas, vidinis vardas. Primityvo viduje kuriamas procesas, jis įdedamas į atitinkamus sąrašus (pasiruošusių/blokuotų), sukuriamas jo vaikų sąrašas.

2. **Naikinti procesą** - pirmiausia sunaikinami proceso sukurti vaikiniai procesai bei resursai. Tada procesas pašalinamas iš sąrašų, kuriems jis priklauso. Pabaigoje sunaikinami procesui perduoti resursai ir jo pačio deskriptorius.

3. **Stabdyti procesą** - proceso būsena pakeičiama iš BLOCKED į BLOCKED-STOPPED arba iš READY į READY-STOPPED.

4. **Aktyvuoti procesą** - būsena keičiama iš BLOCKED-STOPPED į BLOCKED arba iš READY-STOPPED į READY.

6 Resursai

6.1 Resursų deskriptorius

1. **RID** - resurso vidinis vardas.
2. **TYPE** - resurso tipas.
3. **PROC** - sąrašas procesų, kurie prašė šio resurso.
4. **STATE** - resurso būsena.
5. **PID** - procesas, sukūręs resursą.
6. **CONTENT** - ???.

6.2 Resursų tipai ir magic number

(INT) - resursą norima sukurti dėl programinio arba supervizorinio pertraukimo, tačiau ne dėl timer'io pertraukimo. (USER) - vartotojiškas procesas.

Resursų tipai, jų aprašymai, susiję procesai ir šių resursų MagicNumber (nurodomas kaip numeris sąraše):

1. **OSExit** - OS darbo baigimas. Laukia Start_Stop procesas, kuria - ProcessKiller.

   Resursai, skirti darbui su vidine atmintimi: Kuriantys procesai gali būti bet kokie, o laukiantis - visada Get_Put_data.

2. **ReadBlock** - Blokas, esantis atmintyje, atmintyje, įrašomas į duomenų segmentą.
3. **WriteBlock** - Blokas, esantis duomenų segmente, įrašomas į vietą, nurodytą atmintyje.
4. **ReadWord** - Žodis, esantis atmintyje, įrašomas į duomenų segmentą.
5. **WriteWord** - Žodis, esantis duomenų segmente, įrašomas į vietą, nurodytą atmintyje.

   Resursai, skriti darbui su srautais:

6. **NeedInput** - Įvedimo srautas iš vartotojo. Laukia Chan_2_Device, kuria - (INT).
7. **NeedOutput** -Išvedimo srautas į ekraną. Laukia Chan_3_Device, kuria - (INT).

   Resursai, skirti darbui su failais: Kuriantysis procesas visada (INT), o laukiantysis - Chan_1_Device.

8. **ReadFile** - Failo nuskaitymas.
9. **OpenFile** - Failo atidarymas.
10. **WriteFile** - Rašymas į failą.
11. **CloseFile** - Failo uždarymas

12. **DeleteFile** - Failo ištrinimas pagal nurodytą deskriptorių.
13. **SeekFile** - Pozicijos faile pakeitimas.

Resursui, skirti darbui su programomis:

14. **ProgramStart** - Programos pakrovimas į atmintį. Laukia Loader procesas, kuria - (INT).
15. **ProgramHalt** - Programos sustabdymas. Laukia ProcessKiller procesas, kuria - (INT).

6.3 Resursų paskirstytojas

Resursų paskirstytojas suteikia paprašytus resursus procesams pagal prioritetus, jo skirstymo pabaigoje kviečiamas procesų paskirstytojas.

6.4 Resursų primityvai

1. **Kurti resursą** - procesas kuria resursą. Perduodami parametrai yra tokie: nuoroda į proceso tėvą, resurso vidinis vardas. Resursas pridedamas prie bendrojo resursų sąrašo, taip pat prie tėvo sukurtų resursų sąrašo, sukuriamas resurso elementų sąrašas ir kuriamas laukiančių procesų sąrašas.
2. **Prašyti resurso** - procesui paprašius resurso, jis užsiblokuoja ir yra įtraukiamas į laukiančiųjų resurso procesų sąrašą.
3. **Atlaisvinti resursą** - primityvą kviečia procesas, kuris nori nereikalingą resursą arba perduotį informaciją kitam procesui. Primityvo pabaigoje kviečiamas resursų paskirstytojas.
4. **Naikinti resursą** - resurso deskriptorius išmetamas iš tėvo sukurtų resursų sąrašo, bendrojo resursų sąrašo, atblokuojami procesai, kurie laukė šio resurso, sunaikinamas pats deskriptorius.