Vilniaus universitetas
Matematikos ir informatikos fakultetas
Programų sistemų katedra

„FluffyOS“

Virtualios mašinos projektas

Darbą atliko 2 kurso 1 grupės studentai (*„FluffySoft”*):
Karolis Jocevičius
Ugnė Laima Čižiūtė

Vilnius
2013

“FlufffyOS” operacinės sistemos virtualios ir realios mašinos projektas skirtas Programų Sistemų Operacinių Sistemų kursui.

Darbo autoriai:

Karolis Jocevičius
karolis.jocevicius@gmail.com
Tel. Nr. 869029294

Ugnė Laima Čižiūtė
ugne.ciziute@gmail.com
Tel. Nr. 868901518

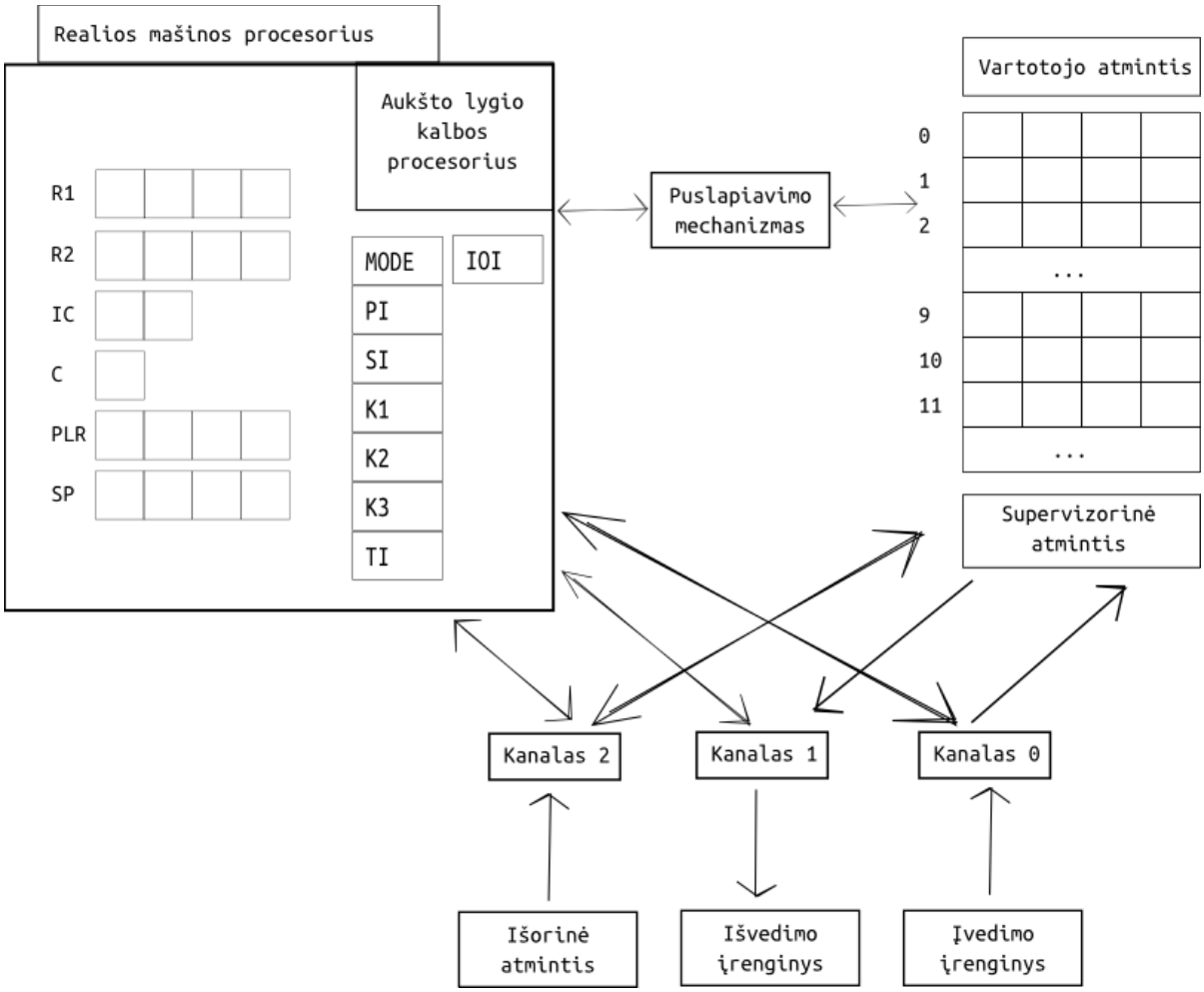
Turinys

Realios mašinos specifikacija

Aprašymas

Realios mašinos procesorius gali dirbti dviem režimais: supervizoriaus ir vartotojo. Ji naudoja bendrojo naudojimo, steko viršūnės, komandų skaitliuko, loginio trigerio, puslapių lentelės, procesoriaus darbo režimo registrus, bei programinių ir supervizorinių pertraukimų registrus. Taip pat jis turi registrus, nusakančius kanalo užimtumą, bei taimerio registrą. Operatyviąją atmintį sudaro 600 žodžių. Duomenų įvedimui yra naudojama klaviatūra, o išvedimui ekranas.

Schema



Realios mašinos komponentai ir jų aprašymas

Realią mašiną sudaro

- Procesorius
- Operatyvioji atmintis
- Išorinė atmintis
- Duomenų perdavimo kanalai
- Įvedimo/išvedimo įrenginiai
- Išorinės atminties įrenginys

Procesorius

Procesorius tikslas yra nuskaityti komandas iš atminties ir jas interpretuoti (įvykdyti).
Jis gali dirbti dviem režimais: vartotojo ir supervizoriaus. Į supervizoriaus režimą yra patenkama prieš pradedant vykdyti programą arba įvykus pertraukimui. Supervizoriaus režime aukšto lygio kalbos procesorius apdoroja supervizoriaus atmintyje esančias komandas, o vartotojo režime vykdo užduoties programą.

Procesoriaus registrai

- R0, R1 – 4 baitų bendro naudojimo registrai.
- IC – 2 baitų virtualios mašinos komandų skaitliukas.
- C – 1 baito loginis trigeris (T – true, F – false).
- PLR – 4 baitų puslapių lentelės registras.
- SP – 4 baitų steko viršūnės registras. Saugo virtualų steko viršūnės adresą.
- MODE – registras, kurio reikšmė nusako procesoriaus darbo režimą (V - vartotojas, S - supervizorius).
- PI – programinių pertraukimų registras
    - 0 - nėra pertraukimo
    - 1 – kviečiama neegzistuojanti komanda
    - 2 – neigiamas rezultatas
    - 3 - dalyba iš 0
    - 4 - perpildymas
- SI – supervizorinių pertraukimų registras
    - 0 - nėra pertraukimo
    - 1 - jei kviečiama įvedimo komanda
    - 2 - jei kviečiama išvedimo komanda
    - 3 - HALT

- IOI - įvedimo/išvedimo pertraukimų registras
    - 1 - pertraukimas pirmame kanale
    - 2 - pertraukimas antrame kanale
    - 3 - pertraukimas trečiame kanale
- K[i] – parodo 1/2/3 kanalo užimtumą
    - 0 – laisvas
    - 1 – užimtas
- TI – taimerio registras
    - 0 - įvyko pertraukimas

Taimerio mechanizmas - pradedant virtualios mašinos užduoties vykdymą TI registro reikšmė bus nustatoma tam tikrai reikšmei. Įvykdžius eilinę instrukciją TI reikšmė bus mažinama priklausomai nuo to per kiek ši instrukcija yra atliekama.

Operatyvioji atmintis

Operatyvioji atmintis susideda iš supervizorinės ir vartotojo atminties. Operatyviąją atmintį sudaro 600 žodžių (nuo 0 iki 599), iš kurių kiekvienas turi po 4 baitus. Ji yra suskaidyta į 60 blokų (nuo 0 iki 59) po 10 žodžių. Pirmi 50 blokų yra skiriami vartotojo atminčiai, o paskutiniai 10 blokų yra skiriami supervizorinei atminčiai. Bloko dydis 40 baitų (10 žodžių).
Supervizorinė atmintis yra skirta pačios operacinės sistemos poreikiams: joje laikomi sisteminiai procesai, sisteminiai kintamieji, resursai ir t.t.
Vartotojo atmintis yra skirta virtualių mašinų atmintims bei puslapių lentelėms laikyti.

Išorinė atmintis

Kietasis diskas.

Duomenų perdavimo kanalai

Kanalai leidžia dirbti su išoriniais įrenginiais. Priklausomai nuo nustatytų registrų, kanalai gali vykdyti apsikeitimus duomenimis visomis nurodytomis kryptimis. Veiksmai su kanalais vykdomi tik supervizoriaus režime.
Įvedimo/išvedimo operacija nurodoma komanda StartIO(K, In, Out, n) (K - kanalo numeris, In - nuoroda į blokų, iš kurių imama informacija, masyvą, Out - nuoroda į blokų, gaunančių informaciją, masyvą, n - perduodamų blokų skaičius). Kai komanda StartIO duodama i-tajam kanalui, registras K[i] įgyja reikšmę 1, pasibaigus įvedimo/išvedimo veiksmui K[i] įgyja reikšmę 0. Jei komanda StartIO duodama užimtam kanalui, centrinis procesorius laukia kanalo atlaisvinimo.

Įvedimo/išvedimo įrenginiai

Įvedimo įrenginys – klaviatūra, perduoda informaciją per pirmąjį kanalą.
Išvedimo įrenginys – vaizduoklis, informaciją gauna per antrąjį kanalą.

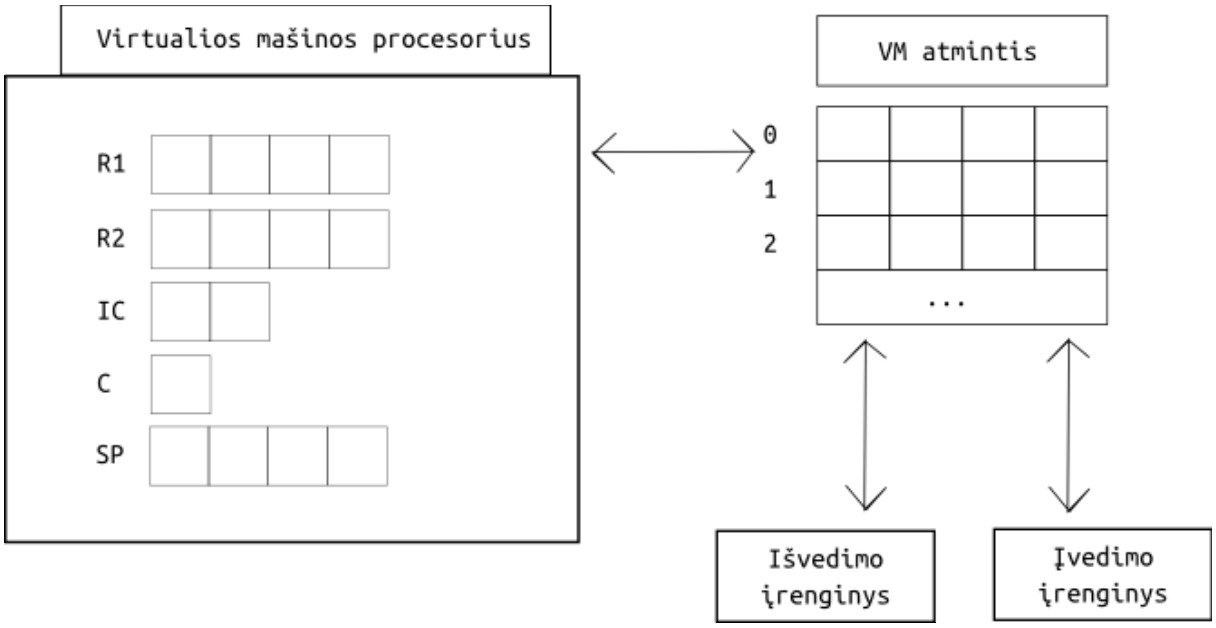
Atlieka savo veiksmus per 3 sąlyginius vienetus.

Virtualios mašinos specifikacija

VM samprata

Virtuali mašina (toliau VM) - tai supaprastintas realios mašinos (RM) modelis, veikiantis kaip tarpininkas tarp realios mašinos ir programinės įrangos. Virtualios mašinos procesorius naudoja bendro naudojimo, steko viršūnės, komandų skaitliuko bei loginio trigerio registrus. Virtualios mašinos atmintis atvaizduojama į vartotojo atmintį naudojant puslapių transliaciją. Atmintis yra 100 žodžių dydžio. Yra komandos duomenų persiuntimui iš atminties į registrus ir atvirkščiai, aritmetinės (sudėties, atimties, daugybos, dalybos), palyginimo, valdymo perdavimo, įvedimo/išvedimo ir programos pabaigos komandos.

VM schema



VM sandara

VM atmintis

- Žodžio dydis: 4 B
- 100 žodžių (nuo 0 iki 99)
- 10 blokų po 10 žodžių

VM procesorius

VM turi virtualų procesorių, kurio paskirtis - vykdyti virtualioje atmintyje esančią programą.

VM procesoriaus registrai

- R1, R2 - 4 baitų bendro naudojimo registrai
- SP - 4 baitų steko viršūnės registras
- IC - 2 baitų VM komandų skaitliukas
- C - 1 baito loginis trigeris (True/False)
    - Pirmasis bitas: zero flag (ZF):
        - ZF = 1, jei registro ir w, esančios adresu xy skirtumas lygus 0.
        - ZF = 0 – priešingu atveju.
    - Antrasis bitas: sign flag (SF)
        - SF = 1, jei atlikus aritmetinę operaciją su registro ir atminties xy reikšmėmis rezultato pirmasis bitas yra 1.
        - SF = 0 – priešingu atveju.
    - Trečiasis bitas: overflow flag (OF)
        - OF = 0, jei registro ir reikšmės, esančios adresu xy ženklo bitai yra vienodi, o jų skirtumo ženklo bitas skiriasi.
        - OF = 1 – priešingu atveju.

Įvedimo/išvedimo įrenginiai

Virtualus įvedimo ir išvedimo įrenginys valdomas klaviatūra. Išveda informaciją į ekraną.

VM procesoriaus komandos

Atmintyje esantis 4 baitų žodis gali būti traktuojamas kaip duomenys arba komanda. Pirmieji du baitai (vyresnieji) laikomi operacijų kodu, likę naudojami kaip operandai, paprastai tai bus atminties adresas. Kiekviena virtualios mašinos komanda vykdoma per 1 sąlyginį laiko vienetą.

Atminties komandos

L1xy	Load R1	Užkrauna reikšmė į R1 registrą iš atminties ląstelės xy R1:=[x*10+y]
L2xy	Load R2	R2:=[x*10+y]
LxRy	Load Rx from [Ry]	Užkrauna reikšmę į Rx iš [Ry] (x, y = 1 arba 2)
S1xy	Save R1	Išsaugo R1 reikšmę į atminties ląstelę xy [x*10+y]:= R1
S2xy	Save R2	[x*10+y]:= R2
CPY1	Copy to R1	Nukopijuoja R2 reikšmę į R1
CPY2	Copy to R2	Nukopijuoja R1 reikšmę į R2
AWx	Allocate word	Išskiria žodį, patalpina į jį reikšmę x. Išskirto žodžio adresą išaugo R2 registre.

Steko komandos

PUx	Push	įdeda registrą (nurodytą “x”) į steką. Pvz. “PUR1” Jei registro vardas pertrumpas pridedamas “0” (pvz. “PUC0”). Jei per ilgas - nukertamas (pvz. IOI -> IO)
POx	Pop	išima registrą (nurodytą “x”) iš steko

Aritmetinės komandos

A1xy	Add R1	Prie R1 registro prideda ląstelėje xy esančią reikšmę. R1:=R1 + [x*10+y]
A2xy	Add R2	R2:=R2 +[x*10+y]
B1xy	Sub R1	Iš R1 registro reikšmės atimama xy esanti reikšmė. R1:=R1 - [x*10+y]
B2xy	Sub R2	R2:=R2 - [x*10+y]
M1xy	Multiply R1	Sudaugina R1 registro reikšmę su xy reikšme. R1:=R1 * [x*10+y]

M2xy	Multiply R2	R2:=R2 * [x*10+y]
DIxy	Divide R1	Padalina R1 esančią reikšmę iš xy esančios reikšmės. Atsakymą išsaugo R1, liekaną R2. R1:=R1 div [x*10+y], R2:=R2 mod [x*10+y]
DEC1	Decrement R1	Sumažina R1 registro reikšmę R1:=R1 - 1
DEC2	Decrement R2	Sumažina R2 registro reikšmę. R2:=R2 - 1
INC1	Increment R1	Padidina R1 registro reikšmę. R1:=R1 +1
INC2	Increment R2	Padidina R2 registro reikšmę. R2:=R2 +1

Loginės komandos

C1xy	Cmp R1	Palygina R1 reikšmę su xy ląstelės reikšme. Atsakymą įrašo į požymių registrą.
C2xy	Cmp R2	Palygina R2 reikšmę su xy ląstelės reikšme. Atsakymą įrašo į požymių registrą.

Valdymo perdavimo komandos

JPxy	Jump	Perduoda valdymą į adresą xy
JExy	Jump if equal	Perduoda valdymą į adresą xy, ZF = 1.
JLxy	Jump if less	Perduoda valdymą į adresą xy, ZF = 0, SF = OF.
JGxy	Jump if greater	Perduoda valdymą į adresą xy, ZF = 0, SF<>OF.

Įvedimo/išvedimo komandos

INxy	Input	Registrui SI nustatoma reikšmė SI = 1 ir valdymas perduodamas OS, duomenų kopijavimui iš įvedimo įrenginio į ląstelę xy.
OPxy	Output	Registrui SI nustatoma reikšmė SI = 2 ir valdymas perduodamas OS, duomenų kopijavimui į išvedimo įrenginį iš ląstelės xy.

Programos pabaigos komanda

HALT	Halt	Programos pabaiga. Registrui SI nustatoma reikšmė SI = 3.

Registrų tikrinimo/keitimo komandos

GEIC	Get IC	Įrašo IC registro reikšmę į R2

GECx	Get C[x]	Įrašo C[x] registro reikšmę į R2 x - C registro baitas
GESP	Get SP	Įrašo SP registro reikšmę į R2
SEIC	Set IC	Nustato IC registro reikšmę, į nurodytą R2
SESP	Set SP	Nustato SP registro reikšmę į nurodytą R2
SECx	Set C[x]	Nustato C[x] reikšmę į nurodytą R2

Puslapiavimo mechanizmas

Puslapių lentelė leidžia virtualiai mašinai sužinoti kurio nors jos bloko realų adresą. Sukuriant VM jai yra išskiriama atmintis (10 blokų) bei vienas papildomas blokas su puslapių lentele. PLR registras saugo to bloko adresą.
PLR registrą sudaro 4 baitai: a1, a2, a3, a4.

- a1 - nenaudojamas
- a2 - lentelės dydis (dydis - 1)
- a3, a4 - nurodo lentelės adresą realioje atmintyje (adresas = 10 * a3 + a4)

Kiekvienas puslapių lentelės žodis saugo realų adresą į išskirtus blokus.
Formulė, kuri virtualiam adresui xy grąžina realų adresą:
realus adresas = 10*[10*(10 * a2 + a3) + x1] + x2

Programos failo formatas

Programos paleidžiamos interaktyviai, t.y. vartotojui įvedus programos failo vardą kaip komandą. Programos saugomos realios mašinos kietąjame diske, kur jos patalpinamos išorinėmis, o ne projektuojamos OS priemonėmis.

Išeities teksto formatas

Pageidaujama išeities failo galūnė: **.fluff*
Komentarai rašomi po simbolio: “/”

$FLJ	“Fluffy Job”, antraštė.	4B
.NAM x	Nurodo užduoties vardą.	10B
.DAT x	x - nurodo kiek blokų skiriama duomenų segmentui.	
$*x*	x - nurodo bloką, į kurį bus rašomi duomenys. * - nenaudojamas baitas. Pvz.: “$000”, “$020”, ...	4B
$FLC	“Fluffy Code”, Nurodo, kad duomenys bus rašomi į kodo segmento pradžią. $FLC = $000	4B
Kodas	Kiekviena komanda rašoma naujoje eilutėje (“n” eilučių).	n*4B
$DAT	Nurodo, kad toliau duomenys rašomi į duomenų segmentą. $DAT = $0x0, kai x - duomenų segmento pirmas blokas. x = (blokų kiekis) - (blokų paskirtų duomenims kiekis)	4B
Duomenys	Kiekvienas duomuo skaidomas į žodžius (4B). Įterpiant daugiau nei 1 duomenį - toliau reikia nurodyti adresą.	
$END	Nurodo kodo pabaigą.	4B

Programos pavyzdys

	$FLJ	**Antraštė**
	.NAM Demo	**Vardas**
	.DAT 2	**Duomenų segmento dydis**
	$DAT	**Duomenys, kitaip -** *$080*
80	81	**Įterpinėjamos skaitinės reikšmės į:** *80: 0000*
81	84	*81: 0084*
82	5000	*82: 5000*
83	10	*83: 0010*
84	20	*84: 0020*
85	0	85: 0000
	$090	**Keičiamas rašymo adresas į** *90*
90	Rezultatas#	**Eilutė įrašoma į:** *90: "Rezu"* *91: "ltat"* *92: "as#"* *"#"* simbolis nurodo eilutės pabaigą.
	$FLC	**Kodas, kitaip -** *$000*
01	L280	**Užkrauna duomenis iš ląstelės ties adresu** *80 į registrą R2* *R2 = 81* Čia saugome kito skaičiaus adresą.
02	INC2	*R2 = R2 + 1*
03	S298	**Išsaugome R2** adresu 99 (adresas)
04	L1R2	*R1 = [R2]*
05	INC2	*R2 = R2 + 1*
06	A1R2	*R1 = R1 + [R2]*
07	C285	**Lyginam R2 su ląstele** *81: 0084*
08	JE10	**Jei** *R2 = 84***, vykdom toliau** (šokam į adresą *10*)
09	JP05	**Jei ne - grįžtam**
10	OP90	Ląstelė ties adresu *90* **išvedama į ekraną**

11	S199	**Išsaugom R1** *98: [R1]*
12	OP99	Ląstelė ties adresu *98* **išvedama į ekraną**
13	HALT	Programa užbaigiama
	$END	