**Pirmasis laboratorinis darbas „Virtualios ir realios mašinos projektas“**

*Darbą atliko VU MIF Informatikos 3 kurso 1 grupės (2 pogrupio) studentai Artiom Tretjakovas, Algis Perepečo ir Benas Nikitinas*

1. **Realios mašinos komponentai**
   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 Rust procesoriaus. Vartotojo režime su Rust parašytas procesorius vykdo užduoties programą. Procesoriuje laikas sekamas pasitelkiant dažnio osciliatorių, kas tam tikru laiko momentu generuoja elektros pulsus.

Procesoriaus registrai:

* PTR – 4 baitų puslapių lentelės registras;
* R – 4 baitų bendro naudojimo registras;
* PC – 2 baitų virtualios mašinos programos skaitiklis;
* C – 1 baito loginis trigeris;
* MODE – 1 baito registras, kurio reikšmė nusako procesoriaus darbo režimą (supervizoriaus – „S“, vartotojo – „V“);
* PI – programinių pertraukimų registras;
* SI – supervizorinių pertraukimų registras;
* TI – taimerio registras;
* SP – 1 baito steko rodyklės registras.
  1. **Vartotojo atmintis**

Vartotojo atmintis skirta virtualių mašinų atmintims bei puslapių lentelėms laikyti. Mes apibrėšime vartotojo atmintį taip: lentelės dydis – 512 žodžių po 4 baitus. 16 žodžių laikysime vienu bloku. Taigi, vartotojo atmintis lygi 32 blokų, sunumeruotų nuo 00 iki 31

* 1. **Supervizorinė atmintis**

Supervizorinėje atmintyje turėjusias būti komandas ir resursus valdys aukšto lygio kalbos procesorius. Supervizorinę atmintį naudos kanalų įrenginys, programų užkrovimui.

* 1. **Taimeris**

Šis mechanizmas atsakingas už geresnį užduočių išlygiagretinimą. Yra laikoma kad ta pati užduotis negali būti vykdoma ilgiau nei 10 laiko momentų. Mūsų OS atveju bus vykdoma ne daugiau einamosios 10 užduoties taktų. Laikysim kad įvedimo/išvedimo instrukcijos atliekamos per 3 taktus, visos kitos per 1 taktą. Pradedant virtualios mašinos užduoties vykdymą TI registro reikšmė nustatoma tam tikrai reikšmei. Įvykdžius eilinę instrukciją TI reikšmė mažinama priklausomai nuo to per kiek taktų ši instrukcija yra atliekama. Kuomet TI reikšmė yra lygi nuliui, mikrokomanda Test () aptinka taimerio pertraukimą.

* 1. **Klaviatūra**

Klaviatūra bus mūsų įvesties įrenginys. Vartotojas, dirbantis su sistema, programas paleidžia interaktyviai, surinkdamas atitinkamą komandą. Klaviatūra įvedimui generuos pertraukimus.

* 1. **Ekranas**

Ekranas bus mūsų išvesties įrenginys, jį atvaizduos konsolė.

* 1. **Kanalų įrenginys**

Kanalų įrenginys leidžia dirbti su atmintimis. Priklausomai nuo nustatytų registrų kanalų įrenginys gali vykdyti apsikeitimą duomenimis visomis galimomis kryptimis. Kai kuriama virtuali mašina, kanalų įrenginys užkrauna išorinėje atmintyje esančias programas (hdd.txt failas) į supervizorinę atmintį.

* 1. **Kietasis diskas (išorinė atmintis)**

Laikoma, kad vartotojo programos (tiek paleidžiamos paties vartotojo, tiek jo paleistų programų) yra realios mašinos kietajame diske, į kurį jos patalpinamos „išorinėmis“, modelio, o ne projektuojamos OS, priemonėmis. Diskas (išorinė atmintis) nebus realizuojamas – tai bus tik sąvoka t.y. mūsų modelyje išorinė atmintis integruojama į supervizorinę atmintį. Išorinės atminties dydis – 256 blokų.

* 1. **Lemputė**

Panašiai kaip ekranas funkcionuos kaip išvesties įrenginys, kuriam perdavus signalą (nesvarbu kokį) ji „išsijungs“ arba „įsijungs“.

1. **Realios mašinos modelis**

**Procesorius**

MODE

PI

SI

TI

Aukšto lygio kalbos Rust procesorius

PTR

R

PC

C

Puslapiavimo mechanizmas

Vartotojo atmintis

Kanalų įrenginys

Kietasis diskas

Ekranas

Klaviatūra

SP

1. **Virtuali mašina**
   1. **Aprašas**

Virtuali mašina (VM) tai realios mašinos modelis (kopija), kuris veikia kaip tam tikras tarpininkas. Ji smarkiai supaprastina tiek ir programų rašymą tiek ir pačią realizaciją. VM pagrindinė paskirtis vykdyti vartotojo programą.

* 1. **Virtualios mašinos komponentai**
     1. **Atmintis**

Kiekvienai virtualiai mašinai yra skiriama 16 vartotojo atminties blokų. Tuose šešiolikoje blokų (256 žodžių 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. Atmintis bus suskaidyta į 3 dalis:

**Parametrai** – 1 blokas, 0 (16 žodžių)

**Duomenys** – 7 blokai, nuo 1 iki 7 (112 žodžiai 7x16)

**Programa** – 6 blokai, nuo 8 iki 13 (96 žodžiai 6x16)

**Stekas** – 2 blokai, nuo 14 iki 15 (32 žodžiai 2x16)

* + 1. **Procesorius**

Virtualus procesorius yra gerokai paprastesnis realios mašinos atvejis. Virtualios mašinos procesoriaus paskirtis – vykdyti programą, kuri yra virtualioje atmintyje. Kiekvienas procesas turi savo virtualų centrinį procesorių, tačiau modelyje sisteminių procesų programas vykdys aukšto lygio kalbos procesorius. Taigi realiai mūsų projekte virtualius procesorius turės tik procesai – virtualios mašinos. Virtualus procesorius turi tris pagrindinius registrus:

**PC** (Program Counter) - 2 baitų komandų skaitiklis.

**R** (geneRal use) - 4 baitų bendro naudojimo registras.

**C** (Conditional) - 1 baito požymio registras, kurio bitais sekami šitie flag‘ai

1 bitas – CF perkėlimo flag‘as (Carry Flag)

2 bitas – OF perkrovimo flag‘as (Overflow Flag)

3 bitas – SF ženklo flag‘as (Sign Flag)

4 bitas – ZF nulinimo flag‘as (Zero Flag)

5 bitas – rezervuotas

6 bitas – rezervuotas

7 bitas **–** rezervuotas

8 bitas – LF lemputės flag‘as (Lamp Flag)

**3.2.3. Komandų sistema**

**Duomenų persiuntimas iš registrų ir atvirkščiai:**  
**LRxy (0x0001xxyy)** - Atminties ląstelės, kurios adresas x\*16 + y turinio kopijavimas į registrą R.

Trumpiau: R:= [x\*16 + y]

**SRxy (0x0002xxyy)** - Registro R reikšmė įrašoma į atminties ląstelę, kurios adresas x\*16 + y.

Trumpiau [x\*16 + y]:= R

**Aritmetinės:**

**ADxy (0x0003xxyy)** - Prie registro R reikšmės pridedama atminties ląstelės, kurios adresas x\*16 + y reikšmė. Nustato CF OF SF ZF flag‘us priklausomai nuo rezultato

Trumpiau: R:= R + [x\*16 + y]

**SUxy (0x0004xxyy)** - Iš registro R reikšmės atimama atminties ląstelės, kurios adresas x\*16 + y reikšmė. Nustato CF OF SF ZF flag‘us priklausomai nuo rezultato

Trumpiau: R:= R - [x\*16 + y]

**CRxy (0x0005xxyy)** - registrui C priskiriama reikšmė pagal SUxy rezultatą. Nustato CF OF SF ZF flag‘us priklausomai nuo rezultato

**Sąlyginio ir besąlyginio valdymo perdavimo:**

**JPxy (0x0006xxyy) –** nesąlyginio valdymo perdavimo komanda. Valdymas perduodamas kodo sričiai nurodytam adresui.

Trumpiau: PC = 16\*x+y

**JBxy (0x0007xxyy) –** sąlyginio valdymo perdavimo komanda. Jeigu CF == 1, valdymas perduodamas kodo sričiai nurodytam adresui.

Trumpiau: if (CF == 1) PC = 16\*x+y

**Stack‘o:**

**PUSH (0x00080000) –** iš R registro į stack‘ą įkraunama reikšmė. SP padidinamas vienetu.

**POP (0x00090000) –** iš stack‘o įkraunama į R registrą reikšmė, SP sumažinamas vienetu.

**Įvedimo ir išvedimo:**

**RKxy (0x000axxyy)** – Iš įvedimo srauto laukia klavišo paspaudimo, įrašoma į [x\*16 + y] paspausto klavišo kodas.

**RDxy (0x000bxx00)** - Iš įvedimo srauto perskaito 16 žodžių ir įrašo juos į ląsteles [x \*16 + i ], kur i = 0.. 15. Operandas “y” reikšmės neturi.

Trumpiau: for i:= 0 to 15 do [x \*16 + i ]:= buffer[i]. Buffer – šešiolikos žodžių masyvas, kur yra laiko įvedimo srauto rezultatai.

**SDxy (0x000cxx00)** - Išsiunčia išvedimui 16 žodžių srautą iš atminties ląstelių [x \*16 + i ], kur i = 0.. 15. Operandas y reikšmės neturi

Trumpiau: for i:= 0 to 15 do buffer[i]:= [x \*16 + i ]. Buffer – šešiolikos žodžių masyvas, kuris bus išvedamas į ekraną.

**Programos paleidimo ir pabaigos:**

**GOxy (0x000dxxyy)** – Paleidžia programą, kurios pavadinimo pradžia nurodyta x block‘e, o iš y block‘o kopijuojami programos parametrai.

**HALT (0x00000000)** – Vartotojo programos vykdymo pabaiga.

**Ciklų:**

**LOOPxy (0x000exxyy)** – atimama iš R registro 1 ir patikrinama ar ZF == 1, jeigu ne tada valdymas perduodamas kodo sričiai nurodytai x\*16 + y.

Trumpiau: R := R – 1; if(R != 0) PC = 16\*x+y

**Loginės:**

**XORxy (0x000fxxyy)** – R registro ir nurodytos atminties ląstelės bitams, kurios adresas x\*16 + y, įvykdoma XOR operacija. Nustato CF=0 OF=0 ir SF ZF flag‘us priklausomai nuo rezultato

Trumpiau: R:= R ^ [x\*16 + y]

**Lemputės:**

**ON (0x00100000) –** nustato LF = 1

**OFF (0x00110000)** – nustato LF = 0

**ISON (0x00120000)** – išveda į R registrą LF.

* 1. **Įvedimas/išvedimas**

Įvedimo ir išvedimo srautai jungia modelį su vartotojo sąsaja. Vartotojas turės galimybę naudojant klaviatūra suvesti atitinkamas komandas, kurios bus įvykdomos ir gražinamas kažkoks rezultatas, taip pat vartotojas galės paleisti programas, kurios taip pat bus įvykdomos su rezultatu. Grįžtamasis ryšys vyks per „ekraną“ (konsolę), kurioje bus išvedamas vartotojo komandų rezultatas (pvz. aritmetinių atveju bus išvedamas rezultatas, įvedant informaciją bus išvedama ar įvyko).

* 1. **Vykdomojo failo formatas**

Programos laikomos hdd.txt faile. Programos sudėtis:

$PROG

Programos vardas

$DATA

<...> duomenys

$CODE

<...> komandos

$ENDPROG

Duomenų formatas:

DB X - Išskiriamas vienas žodis ir į jį talpinama nurodyta skaitinė reikšmė.

DW ssss - Išskiriamas vienas žodis ir į jį talpinami keturi nurodyti simboliai.

* 1. **Sąryšis su realios mašinos komponentais**

Virtuali mašinos loginiai komponentai bendrauja su realios mašinos techninės įrangos komponentais pasitelkiant puslapiavimą. Virtuali mašina reikalauja 16 blokų atminties savo reikmėms, jai išskiriama 16 blokų. Tada puslapių lentelė (1 blokas) užpildomas išskirtų 16 blokų realiais adresais ir 4 baitų registro PTR reikšmei priskiriamas puslapių lentelės bloko realus adresas. Virtuali mašina baigiama kurti ir gauna procesorių. Su juo virtuali mašina paverčia virtualų adresą (x1x2) realiu pagal formulę: Realus\_adresas:=16\*[16\*(16\*a2+a3)+x1]+x2, kur a2 – registro PTR 3-iasis baitas, a3 – registro PTR 4-asis baitas (a0, a1 – nenaudojami), x1 – virtualaus adreso 1-asis baitas, x2 – virtualaus adreso 2-asis baitas.

1. **Virtualios mašinos modelis**

Virtuali mašina yra modeliuojama realios mašinos pagrindu.

R

PC

C

**Virtualus procesorius**

Ekranas

Virtuali atmintis

Klaviatūra

SP