OS Projektas

I dalis

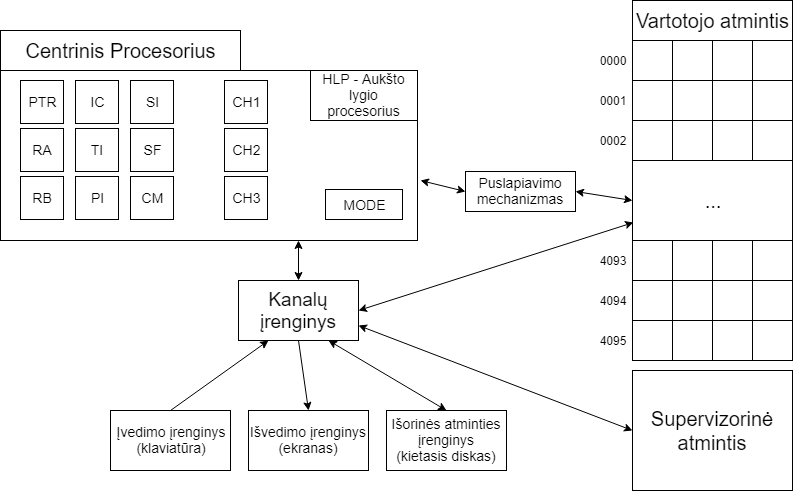
Rytis Petrauskas

Simas Karaliūnas

Reali mašina

Mūsų reali mašina yra kompiuteris kuris sudarytas iš šių esminių komponentų:

* Centrinis procesorius
* Puslapiavimo mechanizmas
* Atmintis:
  + Supervizoriaus
  + Vartotojo
  + Bendra
* Kanalų įrenginys
* Įvedimo įrenginys (klaviatūra)
* Išvedimo įrenginys (ekranas)



Realios mašinos schema

Plačiau apie komponentus:

**Centrinis procesiorius** – skaito komandą iš atminties ir ją vykdo. Turi du režimus – supervizoriaus ir vartotojo:

* Supervizoriaus režime – iš supervizorinės atminties komandos yra betarpiškai apdorojamos HLP (bet kuris aukšo lygio kalbos procesorius).
* Vartotojo režime – HLP vykdo užduoties programą, imituodama virtualios mašinos procesorių. Taip, naudodama puslapiavimo mechanizmą, prieina prie vartotojo atmintyje esančių programų.

Procesorius valdo šiuos registrus:

* MODE – registras, kuris nusako procesoriaus darbo režimą (vartotojas, supervizorius)
* PTR – 4 baitų puslapių lentelės registras
* TI – taimerio registras
* PI – programinių pertraukimų registras
* SI – supervizorinių pertraukimų registras
* RA – bendros paskirties registras
* RB – bendros paskirties registras
* IC – 2 baitų virtualios mašinos programos skaitiklis
* SF – 1 baito dydžio požymių registras savyje laikantis tris požymius – Overflow flag (OF), Carry flag (CF) ir Zero flag (ZF). Kiekvienas požymis užima po vieną bitą registre.
* CM – registras rodantis į bendrą atmintį.
* CH1, CH2, CH3 – kanalų registrai skirti įrenginiams valdyti. 1 – Jei kanalas užimtas, 0 – Jei kanalas laisvas.

**Atmintis** (4096 žodžiai po 4 baitus, 256 blokai):

* Supervizorinė – atmintis reikalinga pačios operacinės sistemos poreikiams (joje laikomi sisteminiai procesai, sisteminiai kintamieji, resursai ir pan.). Mūsų kuriamoje operacinėje sistemoje šią funkciją valdys HLP.
* Vartotojo – skirta virtualių mašinų atmintims bei puslapių lentelėms laikyti.
* Bendra – bendrai naudojamos atminties blokas, prieinamas visoms virtualioms mašinoms. Komandos leidžia į jį rašyti ir skaityti; sritis apsaugoma semaforais.
* Išorinė –realizuota failu kietame diske.

**Kanalų įrenginys**

Tai dalis realios mašinos skirtos darbui su atmintimi ir įvedimo bei išvedimo įrenginiams valdyti. Trys registrai suteikia kanalus, kurie įgaliną šį valdymą:

* CH1 – jungia klaviatūrą su vartotojo atmintimi.
* CH2 – jungia ekraną su vartotojo atmintimi.
* CH3 – jungia kietąjį diską su supervizorine bei vartotojo atintimi.

**Puslapiavimo mechanizmas**

Kiekvienai virtualiai mašinai yra išskiriama 16 blokų (puslapių). Virtuali mašina šiuos blokus sunumeruoja nuo 0 iki 15, tačiau realūs blokų adresai neatitinka VM sukurtos numeracijos. Siekiant sužinoti realius blokų adresus yra naudojamas puslapiavimo mechanizmas.

Realius blokų numerius talpiname į puslapių lentelę, kuri pati užims vieną bloką vartotojo atmintyje. Kiekvienas žodis puslapių lentelėje atititks vieną VM bloką ir jame saugos to bloko realų numerį.

Tačiau puslapių lentelė taip pat turi būti saugoma atmintyje. Šiai lentelei rasti atmintyje naudosime PTR registrą.

PTR registrą sudarys 4 baitai, simboliškai juos galime pažymėti a0a1a2a3. Baitas a0 bei a1 bus nenaudojami. Laikykime x1x2 virtualiu adresu, tada siekiant rasti realų adresą naudojame formulę:

16\*[16\*(16\*a2+a3) + x1] + x2

**Pertraukimai ir jų tipai**

Taimeris – mechanizmas atsakingas už užduočių išlygiagretinimą. Viena užduotis negali būti vykdoma ilgiau nei N taktų. Laikysim, kad visos instrukcijos bus atliekamos per 1 taktą, išskyrus įvedimas/išvedimas, kas užtruks 2 taktus. Pradedant VM užduoties vykdymą, TI registras įgauną tam tikrą reikšmę, 30. Priklausomai nuo įvykdytos instrukcijos registro reikšmė mažėja, kol pasiekia nulį. Įvykus tokiam įvykiui mikrokomanda Test() aptinka taimerio pertraukimą.

Pertraukimas – tai signalas apie įvykusį įvykį, kuris turi unikalią identifikaciją (kiekvienas pertraukimas turi tam tikrą unikalią reikšmę). Pertraukimas gali būti aptiktas tik vartotojo rėžime. Įvykus pertraukimui, jis nenutraukia sistemos darbo, o išsaugo VM registrų reikšmės ir perjungia procesorių į supervisoriaus rėžimą. Supervisoriaus rėžime yra nustatomas pertaukimo pobūdis. Nustačius pertraukimo pobūdį, kviečiama pertraukimą apdorojanti programa. Kai pertraukimą apdorojanti programa baigia savo darbą, valdymas grįžta atgal į VM, vartotojo rėžimą ir atstatomi visi registrai.

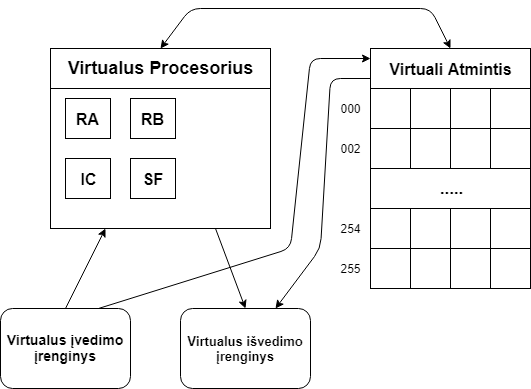
Pertaukimus aptinka komanda Test().

Pertraukimų tipai:

* Programiniai, kuriuos valdo PI registras:
  + PI = 1 – blogas operacijos kodas.
  + PI = 2 – neteisingas priskyrimas.
  + PI = 3 – neteisingas adresas
  + PI = 4 - perpildymas
* Supervizoriniai, kuriuos valdo SI registras:
  + SI = 1 – komanda PRTS
  + SI = 2 – komanda PRTN
  + SI = 3 – komanda READ
  + SI = 4 – komanda END
* Taimerio, kuriuos valdo TI registras:
  + TI = 0 – taimerio skaitliukas lygus 0

Virtuali mašina

**Virtuali mašina** – tai virtuali, paprastesnė realios mašinos kopija. Ji kuriama naudojant realios mašinos komponentes tokias kaip procesorius, atmintis ir įvedimo/išvedimo įrenginiai bei suteikiant jiems paprastesnę vartotojo sąsaja.



Virtualios mašinos schema

Virtualios mašinos komponentes:

**VM atmintis** bus sudaryta iš 16 blokų, po 16 žodžių, taigi išvis atmintyje bus 256 žodžių (kiekvieno žodžio ilgi laikysime 4 baitus).

Pati atmintis bus suskirstyta į dvi lygias dalis, viena skirta duomenims, kita kodui laikyti. Kiekviena dalis užims 8 blokus atminties, arba 128 žodžius. Programos kodas bus laikomas pirmuose 8 blokuose atminties (0 – 7), o likusieji (8 – 15) bus paskirti duomenų laikymui.

**VM procesorius** kur kas paprastesnis nei realios mašinos. Jo paskirtis – vykdyti programa kuri yra virtualioje atmintyje.

Procesų registrai:

* + RA – bendros paskirties registras
  + RB – bendros paskirties registras
* IC – 2 baitų virtualios mašinos programos skaitiklis
* SF – požymių registras

**VM komandos:**

* + - Aritmetinės:
* ADD – sudeda reikšmes esančias RA ir RB registruose. Rezultatą pateikia RA registre.   
  *RA := RA + RB*
* SUB – iš reikšmės esančios RA register atima reikšmę esančią RB register. Rezultatas pateikiamas RA registre.   
  *RA := RA – RB*
* CMP – lyginamos reikšmės esančios RA ir RB registruose. Rezultatas pateikiamas SF registre.   
  *IF(RA == RB) ZF := 1*

*IF(RA < RB) CF := 1*

*IF(RA > RB) CF := 0*

* + - Darbas su atmintimi:
      * MAxy - Atminties ląstelės, kurios adresas x\*16 + y turinio kopijavimas į registrą RA.   
        *RA:= [x\*16 + y]*
      * MBxy - Atminties ląstelės, kurios adresas x\*16 + y turinio kopijavimas į registrą RB.   
        *RB:= [x\*16 + y]*
      * AMxy - Registro AR reikšmė įrašoma į atminties ląstelę, kurios adresas x\*16 + y.   
        *[x\*16 + y]:= AR*
      * BMxy - Registro BR reikšmė įrašoma į atminties ląstelę, kurios adresas x\*16 + y.   
        *[x\*16 + y]:= BR*
      * SAxy – Bendros atminties ląstelės, kurios adresas x\*16 + y turinio kopijavimas į registrą RA.   
        *RA:= [x\*16 + y]*
      * ASxy - Registro AR reikšmė įrašoma į bendrą atminties ląstelę, kurios adresas x\*16 + y.   
        *[x\*16 + y]:= AR*
* LCx *–* užblokuojama bendros atminties blokas x.
* UCx *–* atblokuojama bendra atmintis blokas x.
  + - Valdymo perdavimas:
      * JPxy – nesąlyginio valdymo perdavimo komanda. Valdymas perduodamas kodo sričiai nurodytam adresui.   
        *IC := 16\*x+y*
      * JExy – jei ZF = 1valdymas perduodamas adresu 16\*x + y.  
        *IF( ZF == 1) IC := 16 \* x + y;*
      * JBxy – jei CF = 1 valdymas perduodamas adresu 16\*x + y.  
        *IF( CF == 1) IC := 16 \* x + y;*
      * JAxy - jei CF = 0 ir ZF = 0 valdymas perduodamas adresu 16\*x + y.  
        *IF( CF == 0 && ZF == 0) IC := 16 \* x + y;*
      * HALT – vartotojo programos vykdymo pabaiga
    - Įvedimo bei išvedimo komandos
      * PRTS – RA registre esančią reikšmę traktuoja kaip simbolius ir išveda į išvedimo įrenginį.
      * PRTN – RA registre esančią reikšmę traktuoja kaip skaitinę reikšmę ir išveda į išvedimo įrenginį.
      * Pxyz – į išvedimo įrenginį išveda x numeriu nurodyto atminties srities bloko nuo y iki z žodžius (y <= z). (7 < x <= 15)  
        *Print([16\*x+i, i=y..z])*
      * READ – nuskaito vartotojo įvedimą kaip skaičių ir įrašo į RA registrą.
    - Duomenų segmento apraše galimos komandos:
      * DATA – duomenų segmento aprašo pradžia
      * DW X – išskiriams žodis į kurį talpinama skaitinė reikšmė X
      * DD dddd – išskiriamas žodis į kurį talpinami nurodyti simboliai
      * DN – talpinama simbolinė konstanta reiškianti \n

Kodo pavyzdys (apskaičiuoja ((300 + 200) – 55) ir palyginą reikšmę su 445) (kodas ir atminties numeracija nuo šiol bus pateikiami šešioliktainių skaičiavimo formatu):

|  |  |  |
| --- | --- | --- |
| 0000 | DATA | |
| 0001 | DW 12C | |
| 0003 | DW C8 | |
| 0005 | DW 37 | |
| 0007 | DD LYGU | |
| 0009 | DD NELY | |
| 000B | DD GU | |
| 000D | START | |
| 000E | MA80 | |
| 000F | MB81 | |
| 0010 | ADD | |
| 0011 | MB82 | |
| 0012 | SUB | |
| 0013 | MB83 | |
| 0014 | CMP | |
| 0015 | JE18 | |
| 0016 | P856 | |
| 0017 | HALT | |
| 0018 | P844 | |
| 0019 | HALT | |
| 0080 | \_12C |
| 0081 | \_\_C8 |
| 0082 | \_\_37 |
| 0083 | \_1BD |
| 0084 | LYGU |
| 0085 | NELY |
| 0086 | GU\_\_ |