Laimonas Beniušis, Kompiuterių Mokslas 1, 1gr. stud.nr 1410102

OS app

Turinys

Stekinė virtuali mašina (Vykdo „Bytecode“)

Registrai:

IP – sekanti instrukcija

SP – steko „pointer“

PLR – pusliapių lentelės adresas

Reali Mašina (Kernel)

Papildomi registrai

T - „timer“ (intervalas – 8);

IT – pertraukimų įjungimas/išjungimas

Visi registrai yra 4 baitų

Komandos

Aprašymuose naudojama: b – stekas[sp], a – stekas[sp-1]

nop – nėra veiksmų

iadd – stekas[sp – 1] = a + b

isub – stekas[sp – 1] = a - b

imul – stekas[sp – 1] = a * b

ilt – (less then) stekas[sp - 1] = (a < b) ? 1 : 0

ieq – (equals) = stekas[sp -1] = (a == b) ? 1: 0

br – (branch) ip = code[ip]

brt – (branch true) daryti br, jeigu stekas[sp] = 1

brf - (branch false) daryti br, jeigu stekas[sp] = 0

iconst – stekas[sp +1] = code[ip]

gload – stekas[sp +1] = stekas[code[ip]]

gstore – stekas[code[ip]] = stekas[sp]

pop – sumažina sp vienetu

int – sisteminis pertraukimas

Komandos su 1 privalomu argumentu: iconst, gload, gstore, int

Komandos su žymėmis (tekstinio tipo): br, brt, brf

Visos kitos neturi argumentų. Padarytas MiniCompiler kuris leidžia rašyti komentarus ir tekstinias žymes (label), naudojamas su br, brf, brt instrukcijomis. Visas komandas verčia į „Bytecode“.

Puslapiavimas

Puslapiavimas yra „lazy allocation“ mechanizmo, t. y. Priskiria lentelėi puslapį tik kada jo trūksta.
Visada VM turi turėti bent 1 „overhead“ puslapį.

PAGE_SIZE = 32
Dėl patogumo, atmintis (RAM) suskirtyta į 16 PAGE_SIZE dydžio „Frame“
Žodžio ilgis 32bit (int ilgio). Taigi, atmintes yra 32 * 4 * 16 = 2048 baitai, arba 2 KiB
Kernel’io atminties neskaičiuojame, nes jos neadresuojame.

Funkcija va2pa (virtualus adresas į realų adresą)

```
MemFrame pagingTable = Kernel.ram[ptr];
int frameIndex = va / PAGE_SIZE;
int offset = va % PAGE_SIZE;
int tableIndex = pagingTable.mem[frameIndex];
int pa = offset + tableIndex * PAGE_SIZE;
return pa;
```

Kaip iš pa nuskaityti arba įrašyti į atmintį:

```
int frameAddress = pa / PAGE_SIZE;
MemFrame frame = Kernel.ram[frameAddress];
int offset = pa % PAGE_SIZE;
frame.mem[offset] = val;
```

Multiprograminė OS

Procesas

Būsenos:

UNUSED – laisva proceso vieta

READY – procesas laukia CPU

ACTIVE – procesas turi CPU

BLOCKED – procesas laukia signalo, kad jo užklausa (job) yra atlikta

ZOMBIE – procesas baigė darbą, tačiau neatlaisvino atminties ir vietos

SysProc (Sisteminiai procesai):

Neturi adresuojamos atminties, naudoja „Kernel“ atmintį.

Viso SysProc tipo procesai vykdo kitų procesų užklausas (job).

Yra atskirti 3 procesai, kurie atlieka tam tikro tipo užklausas

Kada turi 0 užklausų, procesas keičia būseną į BLOCKED, kada gauna naują užklausą, keičia būseną į READY. Dar gali būti ACTIVE būsenos.

FileSystemHandler – su išvedimu į konsolę (standard output), kodo vertimu į „Bytecode“, ir tada tą kompiliuotą kodą siunčia kaip užklausą ProcessHandler

ProcessHandler – naujų procesų sukūrimas

MemoryHandler – atminties tvarkymas (procesų atlaisvinimas ir atminties pakrovimas po puslapį)

1 Papildomas procesas

IdleProcess – daro nieko (užima CPU kai jis laisvas)

Gali būti tik ACTIVE ir READY būsenos, nors ir turi 0 užklausų, nes jis užklausų nevykdo.

IdleProcess kievieną kartą vykdo Context Switch (aprašyta toliau).

UserProc (Vartotojo procesai):

Visi procesai, kurie yra sukurti iš „Bytecode“. Jie vyksta VM viduje.

Turi adresuojamą atmintį.

Gali būti 6 procesai vienu metu.

Pertraukimai (PIC – Programmable Interrupt Controller)

Pertraukimo mechanizmas

Naudojamas PIC

Šis mechanizmas apjungia Vartotojo Procesus ir Operacinę sistemą, nes tada vartotojas gali naudoti priveligijuotomis operacijomis (kurios išreiškiamos per sisteminius pertraukimus).

PIC apjungia Sisteminius (System call), Išimtinius (Exception) ir Asynchroniškus;

Pertraukimų sąrašas:

Išimtiniai 0-15

USER_ERROR	= 15	
INVALID_ADRESS	= USER_ERROR – 1	už proceso adreso rėžio ribų
INVALID_OPCODE	= USER_ERROR – 2	neatpažinta instrukcija
UNKNOWN_ERROR	= USER_ERROR – 3	neatpažinta klaida

Asynchroniški:

TIMER	= 16	reikia keisti aktyvų procesą (Vykdyk Context Switch)
PAGE_FAULT	= 17	procesas prašo daugiau atminties

Sisteminiai:

SYSCALL	= 32	
HALT	= SYSCALL + 32	kode rasta programos pabaiga
WRITE	= SYSCALL + 1	proceso steko paskutinę reikšmę išvesti į konsolę

Visais išimtinio pertraukimo atvejais procesas yra naikinamas.

Context Switch

1. Išsaugojama esamo proceso būsena(PLR, IP ir SP registrai)
2. Surandamas sekantis iš eilės pasiruošęs (READY būsenos) procesas (UserProc arba SysProc)
3. Pakraunama rasto proceso būsena
4. Procesas gauna CPU.

Resursas

MicroKernel resursų (pranešimų) mechanizmą įgivendina Job;

Atliktas Job, keičia jį iškvietusio proceso būseną BLOCKED → READY

SysProc darydami užklausas perduoda ne savo PID (process identification number), o PID_IDLE (proceso kuris nevykdo užklausų)

Išimtys

ACTION_REPEAT_LIMIT = 6

Kaikurios užduotys gali būti įvykdomos ne iš karto (pvz. Sukurti naują procesą, arba Priskirti procesui dar vieną puslapį atminties). Tada yra ACTION_REPEAT_LIMIT kartų bandoma pakartoti tą užklausą. Jeigu per ACTION_REPEAT_LIMIT bandymų pavyksta – tada viskas gerai, jeigu ne, tada ta užklausą pasiuntęs resursas yra atlaisvinamas. Pvz. yra daug procesų ir vienam iš jų pritrūko atminties. Jeigu atminties artimuoju metu (< 6) neatsilaisvins, tada procesas, kuriam pritrūko atmities bus nutrauktas.

JobCleanup užklausos pradžioje procesas keičia būseną į ZOMBIE ir savo pabaigoje proceso, kuris ją sukūrė, pakeičia būseną ne į READY, tačiau į UNUSED.