GR ISPIT 2

***Ishod učenja 4***

1. **[I4\_M, 3 boda ] Objasnite što je prekidni sustav i kako radi.**

Prekidni sustav se koristi kada je za neki događaj potrebno instantno reagiranje procesora. Glavna zadaća prekidnog sustava je dati procesoru da privremeno zaustavi svoje aktivnosti i odradi posao s prekidom. Postoje programski i sklopovni.

Iako ovisi od procesora do procesora, proces se najčešće odvija ovako:

1. Događaj, najčešće nekakva greška, promjena očitanja ili sl.
2. Poziv prekida (IRQ), rezultat stavke 1.
3. Upravljač prekida (PIC) (najčešće se nalazi u south bridge-u u računalnim procesorima), prioritetizira prekide i potvrđuje zahtjev
4. Spremanje stanja, procesor sprema trenutno stanje
5. Izvođenje prekidne podrutine, procesor izvodi nekakav kod za izvršavanje prekidne podrutine
6. Vraćanje prethodnog stanja, procesor vraća prethodno stanje (prije prekida)
7. Procesor nastavlja sa uobičajenim radom

Prekidi omogućavaju da se u bilo kojem vremenu vanjski uređaj izvrši nešto bez obzira na što procesor radi. Najčešće se koristi u situacijama sa senzorima koje nema smisla pollati stalno (npr. senzore temperature koje očitavanje zagrijava itd.). Interrupt se poziva, procesor spremi što je radio na stack, pozove interrupt rutinu gdje piše gdje procesor treba skočiti (najčešće blizu zadnje adrese). Postoje hardwareski i softwareski.

1. **[I4\_M, 3 boda ] Koje vrste pipeline hazarda poznajete?**

Pipeline hazardi se dijele na strukturalne, podatkovne i kontrolne hazarde.

1. Strukturalni hazardi
   1. Opis problema

Dvije ili više instrukcija u isto vrijeme trebaju isti dio centralne procesorske jedinice.

* 1. Rješenje

Mora se čekati (najčešće još 1 ciklus) dok faza nije spremna za izvršavanje. Strukturalni hazardi su fundamentalna „uska grla“ po pitanju performansi i ne mogu se zaobići, jedina solucija je nastojati da se ne događaju.

1. Podatkovni hazardi
   1. Opis problema

Događa se kada se mora čekati na jedan operand da se pripremi od nekog prošlog koraka.

* 1. Rješenje

Mora se čekati da se potrebni podatak/podaci pripreme (latencija/kašnjenje kritičnog puta). Podatkovni hazardi se rješavaju rekonstrukcijom komputacija tako da se kritični put skrati.

1. Kontrolni hazardi
   1. Opis problema

Događaju se kada procesor ne može odrediti koju instrukciju da izvrši sljedeću.

* 1. Rješenje

Najčešće se mora odbaciti cijeli pipeline i početi iznova (što troši hrpu ciklusa). Kontrolni hazardi rješavaju se micanjem grana kompletno ili ih se čini pretpostavljivima kako bi procesor efektivno mogao spekulirati što se sljedeće treba izvršiti.

1. **[I4\_M, 4 boda ] Objasnite što se događa sa procesorom ako se dogodi iznimka (exception).**

Ovisi o procesoru, generalno procedura je:

Procesor stane s onim što radi, spremi stanje, dovrši to (ovisi kako odrađuje iznimke), pronađe gdje je adresa gdje se izvršava prekidna rutina, pozove ju i izvrši.

1. **[I4\_M, 5 boda ] Koristeći sedam segmentni display ispišite samo parne brojeve s kratkom petljom koja će omogućiti da između brojki postoji vremenski razmak.**

.equ SEGMENT\_ADR, 0xFF200020

.equ DELAY\_N, 2000000

.global \_start

\_start:

ldr r3, =SEGMENT\_ADR

mov r0, #0

loop:

ldr r1, =table

ldr r2, [r1, r0, LSL #2]

str r2, [r3, #0]

ldr r4, =DELAY\_N

delay:

subs r4, r4, #1

bne delay

add r0, r0, #2

cmp r0, #0x08

ble loop

exit:

b exit

.data

table:

.word 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6D

***Ishod učenja 5***

**1. [I5\_M, 2 boda ] Je li slijedeća izjava točna: „Procesori imaju ogromne količine brze cache memorije dok grafičke kartice imaju malo cache memorije. Procesori imaju vrlo brz pristup ogromnim količinama radne memorije (RAM), dok grafičke kartice imaju spor pristup grafičkoj memoriji.” Objasnite svoj odgovor.**

Procesori nemaju mnogo cache memorije jer je skupa.  
Grafičke imaju malo cache memorije, jer imaju hrpetinu jezgri (točnije ALU) i jednostavnije linearnije zadatke obavljaju.

Pristup RAM memoriji je spor od strane procesora (u usporedbi sa registrima, cacheom…)

Grafičke kartice imaju brz pristup grafičkoj memoriji.

**2. [I5\_M, 2 boda ] Je li slijedeća izjava točna: I kod SMP-a i kod NUMA-e postoji problem latencije memorije, koja je značajno sporija od cache memorije. Ako se pravilno dizajnira, sustav baziran na NUMA arhitekturi će u ogromnoj količini slučajeva ostvarivati bolje performanse i manje probleme sa kvalitetom usluge (latencije pristupa memoriji, brzine pristupa memoriji). Objasnite svoj odgovor.**

Točna je, ali NUMA sustav će u principu biti bolji.

**3. [i5\_M, 3 boda ] Kod heterogenih i asimetričnih arhitektura (primjer: CPU + GPU), koje vrste jezgri postoje i za kakve poslove bismo ih mogli idealno iskoristiti (1 bod)?**

a. male CPU jezgre (akceleracija serijskih dijelova koda) i velike CPU jezgre (akceleracija paralelnih dijelova koda), uz kvalitetnu predikciju koji kod spada u koju grupu (serijski ili paralelni). Također, GPU-ove za masovno paralelne izračune.

d. najbolje bi bilo napraviti optimalan heterogeni OpenCL kod korištenjem heterogene platforme (CPU+GPU) jer ćemo tako uvijek dobiti najbolje performanse.

Zašto (2 boda)?  
A je najbliže istini, ali D je optimalno, ali jako teško za izvesti.

**4. [I5\_M, 3 boda ] Koje su prednosti i mane privatnog vs dijeljenog pristupa priručnoj (cache) memoriji kod višejezgrenih procesora?**

Privatni cache je nešto što je brže i sigurno će biti jednostavnije za handleanje, dijeljeni je dobar jer možemo imati veću količinu cachea ali će imati probleme s latencijama.

5. [I5\_Ž, 2 boda ] Iz aspekta paralelizma, zašto nam je bitno da paralelno procesiranje ima kvalitetno dizajniran memorijski sustav?

Generalno brzina i efikasnost.

Da možemo iz memorije izvući veliki blok, smanjiti konkurenciju za resurse, smanjiti latencije, optimizirati raspodjelu podataka, povećati propusnost memorije itd.

6. [I5\_Ž, 3 boda ] Koristeći Amdahl-ov zakon izračunajte koliko je ubrzanje u postocima ako su vrijednosti za petlje u primjeru u vježbama redom 2048, 1024, 5120 i 4096, a maksimalno ubrzanje koje možete postići za petlju je 3 puta.

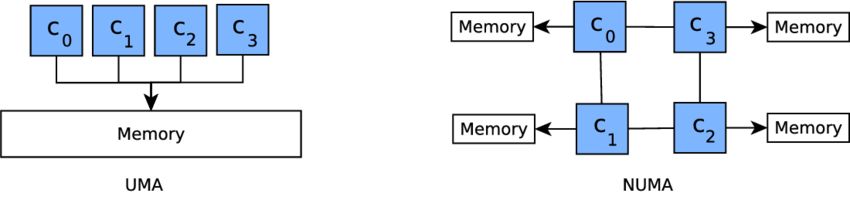
=36/26

***Ishod učenja 6***

1. [I6\_M, 3 boda ] Da li SIMD instrukcije procesora rade različitom brzinom na SMP ili NUMA arhitekturi (1 bod). Zašto? (2 boda)

NUMA će biti brža za workloadove koji su manji. Zbog latencije jer se vrti na različitim podacima NUMA će biti brža. Može biti problema ako su podaci veliki.

2. [I6\_M, 4 boda ] Koja je osnovna razlika između SMP i NUMA modela organizacije sustava sa jednim ili više procesorskih utora i jednom ili više procesorskih jezgri?



3. [I6\_M, 2 boda ] Koja je cijena koju kod NUMA arhitekture moramo „platiti“ ukoliko koristimo remote memoriju (memoriju drugog fizičkog procesora)?

Latencija. Bit će sporiji.

4. [I6\_Ž, 2 boda ] Kako bi u NUMA arhitekturi izgledala optimalna memorijska hijerarhija, ako na izboru imate sve komponente koje na tržištu možete pronaći?

#1 Puno cache memorije, #2 Puno lokalne memorije za svaki socket

5. [I6\_Ž, 3 boda ] Na raspolaganju su vam dvije serverske SMP arhitekture sa dva i četiri socketa (podnožja za procesor na matičnoj ploči). Aplikacija koju trebate instalirati na jedan od ta dva servera je jako osjetljiva na brzinu rada i latenciju memorije. Uz uvjet da obje arhitekture imaju dovoljno procesorske snage za izvršavanje aplikacije čak i na jednom socketu, koja arhitektura bi bila optimalniji izbor za izvršavanje aplikacije (1 bod)? Zašto (2 boda)?

U principu je svejedno, ali 2 generalno iz budžetnih razloga, ako oboje mogu podržati tu aplikaciju.