



## Архитектура и организација рачунара 2 – Колоквијум

1. (5) Посматра се кеш меморија реализована у техници сет асоцијативног пресликавања са 4 улаза по сету која користи модификовани псеудо LRU алгоритам замене. Нацртати и објаснити дијаграм по ком се на основу стања хардвера врши одабир улаза за замену, као и дијаграм по ком се обавља ажурирање хардвера једанпут када се открије сагласност и други пут када се уписује у кеш меморију.

2. (5) У процесору се налази кеш меморија са директним пресликавањем величине 8KiB, која користи *write-through* алгоритам за ажурирање садржаја оперативне меморије са *no write allocated* политиком довлачења. Величина блока кеш меморије је 16B. Дат је следећи код са елементима низова, а који су величине 32 бита. Претпоставити да низ **a** на почетку није учитан у кеш меморију, а **N** је константа која има вредност 127.

```
for (register int i = 0; i <= N; i = i + 1)
    b[i] = a[i] + a[N-i];
```

Потребно је оптимизовати дати код трансформацијом и убацивањем инструкције *prefetch* на одговарајућим местима тако да се максимално смањи број промашаја у кеш меморији података. Претпоставити да резултат инструкције *prefetch* је видљив након 8 итерација било рачунања суме било извршавања инструкције *prefetch*.

3. (5) Објаснити поступак виртуелизације техником *Trap and Emulate*. Посебно се осврнути на то у ком моду рада процесора ради гостујући оперативни систем (*Guest OS*), у ком програми у гостујућем оперативном систему, а у ком програм који обавља виртуелизацију. Објаснити и који су проблеми са овом техником код x86 архитектуре и како се они превазилазе.

4. (5) Дат је меморијски контролер који је повезан на 4 DDR меморијска чипа. Сваки чип је ширине 16 бита. У наставку је описан рад контролера.

1. Меморијски контролер издаје наредбу за **Precharge** да би банка била спремна за приступ новом реду. **Precharge** је завршен након времена  $t_{PC}$ .

2. Меморијски контролер затим издаје наредбу **Activate** за читање одговарајућег реда из банке. Команда **Activate** траје  $t_{RCD}$  времена и након тога сматра се да је цео ред прочитан у јединствен бафер (постоји само један бафер на нивоу свих банака).

3. Меморијски контролер тада може издати наредбу за приступ колони CAS и том приликом чита податке из бафера и ставља их на меморијски канал. После времена  $t_{CL}$  првих 64 бита података су постављени на меморијски канал. Контролер чита кружно осам сукцесивних локација (што представља и величину бафера) од адресе која је задата, а затим аутоматски враћа бафер у банку **Auto precharge**.

4. Ако контролер жели да приступи подацима у другом реду банке, он понавља кораке 1-3. Ако меморијски контролер жели да приступи другом блоку података у истом реду, који се налази унутар бафера, једноставно издаје другу CAS команду.

5. Меморијски контролер **не може** издавати команде различитим банкама у низу тако да не могу да изводе читања/писања паралелно.

Претпоставимо да је  $t_{PC}=t_{RCD}=t_{CL}=15\text{ ns}$  и да је фреквенција меморијског канала 1 GHz.

а) После ког времена најраније може се прочитати податак DATA из меморије, ако се тренутни ред који одговара податку DATA налази у баферу?

б) После ког времена најраније може се прочитати податак DATA из меморије, ако се тренутни ред који одговара податку DATA не налази у баферу, и да се у баферу не налази ни један ред?

в) После ког времена најраније може се прочитати читав ред из меморије, ако се тренутни ред не налази у баферу, и да се у баферу налази неки други ред?

г) Како би требало распоредити податке по банкама да би се најбрже добили сукцесивних 20 адреса. После ког времена се могу прочитати тих 20 адреса, ако се у баферу не налази ни један ред?

**Напомене:** На колоквијуму нису дозвољена никаква помоћна средства, ни калкулатори ни литература. Колоквијум траје **105 минута**.