Компютърни архитектури

Кеш памети

Цел

- Запознаване с начина на работа на кеш паметта
- Запознаване с различните политики за разполагане в кеша и изследване на влиянието им върху производителността
- Изследване на влиянието на размера на кеш паметта върху производителността

План

- Теоретична част
- Практическа част
  - Симулация в MipsIT
  - Задачи и въпроси

Програми срещу процесори

- От гледна точка на програмирането
  - Паметта е непрекъснато голямо линейно пространство от запомнящи клетки
- От гледна точка на изпълнението в процесора
  - Регистри
  - Кеш памет
  - Основна памет
  - Твърд диск

Локалност на обръщенията

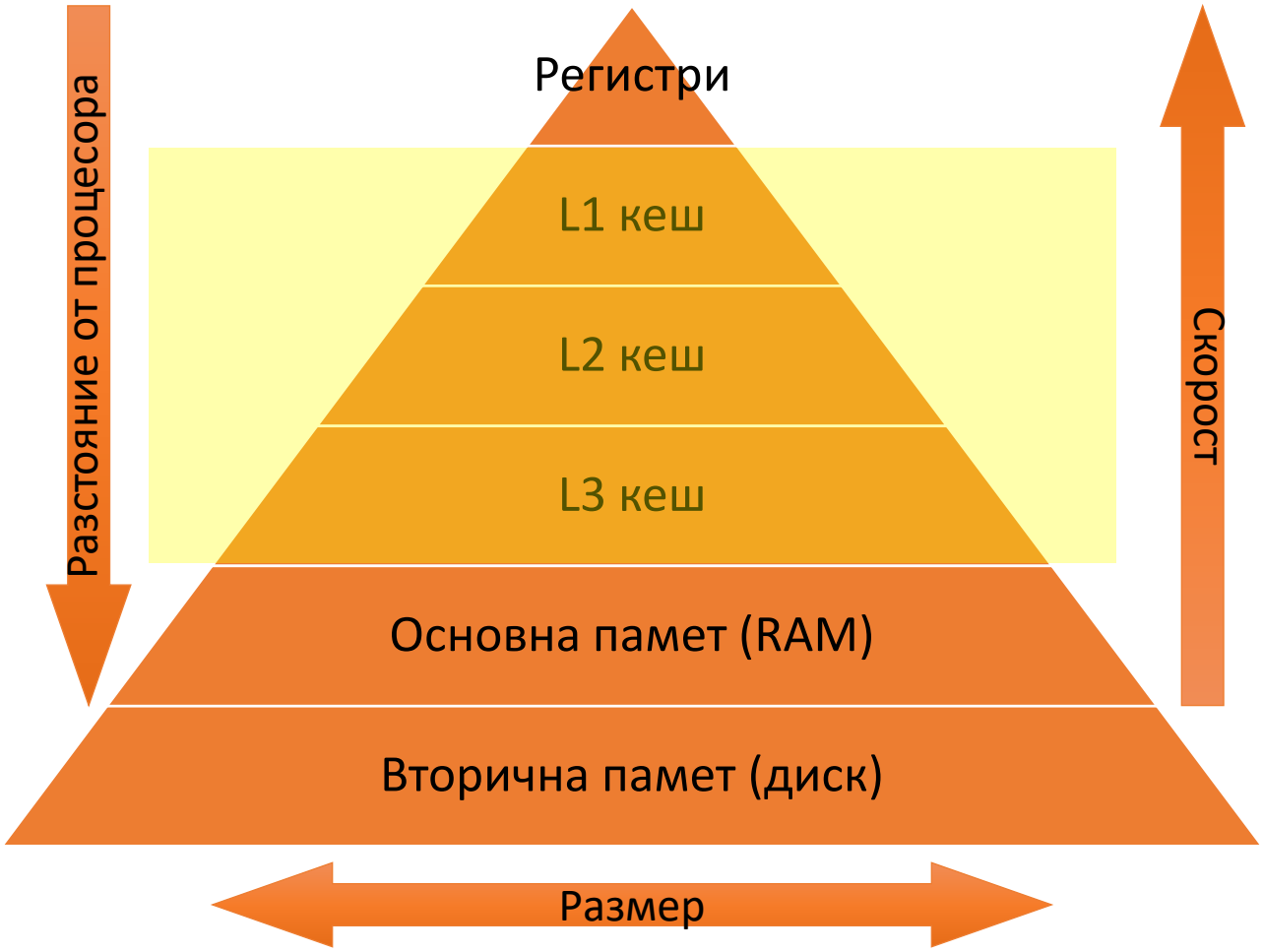
- По време на изпълнението си, програмите осъществяват достъп до малка част от своето адресно пространство
- Времева локалност
  - Вероятността отново да има обръщения към скоро достъпвани елементи в паметта е голяма
  - Например: инструкциите в цикъл; променливи, чиято стойност нараства или намалява в цикъл
- Пространствена локалност
  - Елементи в близост до тези, до които е имало достъп, наскоро, вероятно ще бъдат достъпвани в най-близко време
  - Например: достъпът до последователно разположените инструкции; достъпът до елементите в масив

Локалност на обръщенията

- Йерархия на паметта
- Всичко се съхранява на диска
- Скоро достъпваните (близко разположени) елементи от диска се копират в по-малката DRAM памет
  - Основна памет
- Наскоро достъпваните (близко разположени) елементи от DRAM се копират в по-малката SRAM памет
  - Кеш памет в CPU

Йерархия на паметта

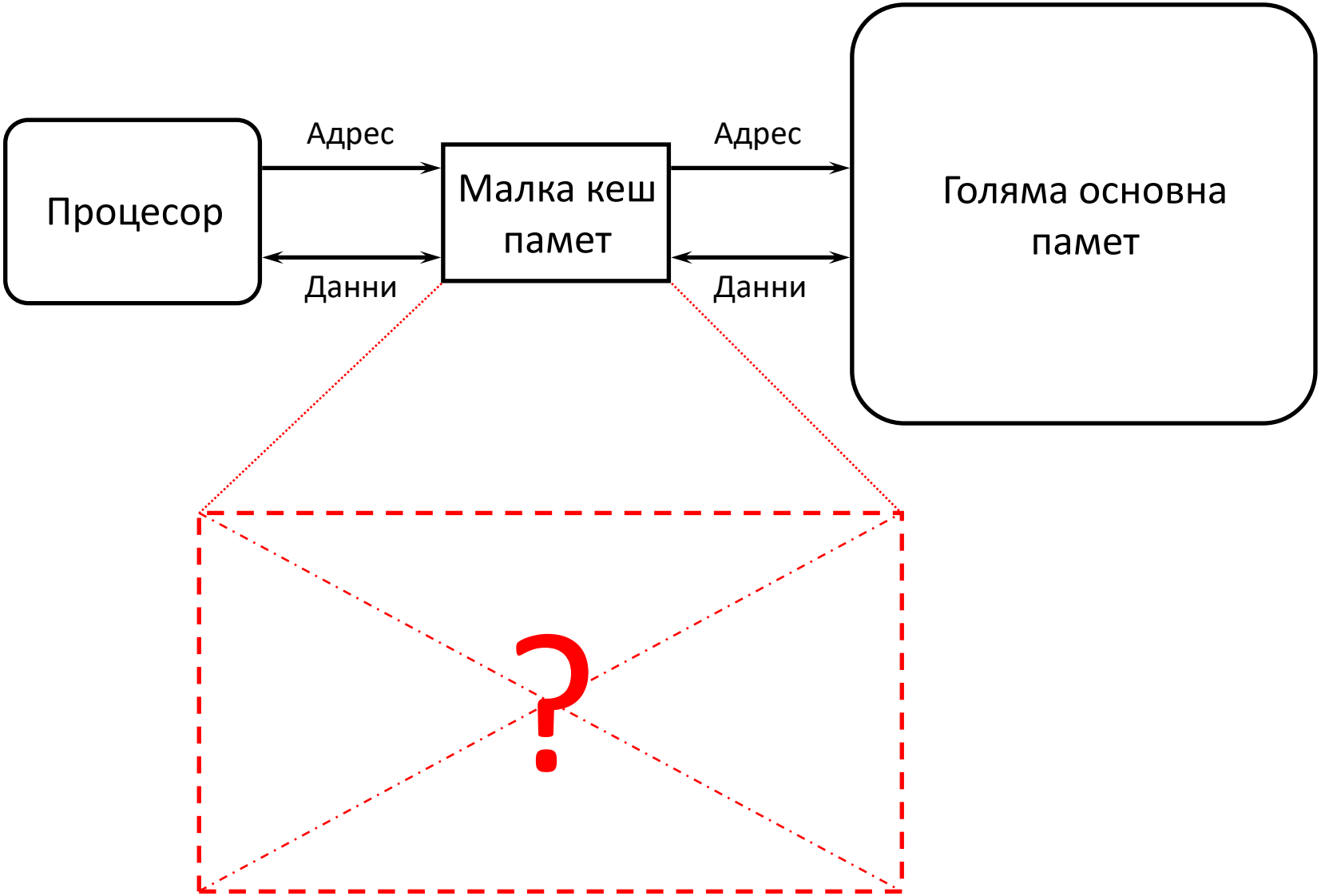
Йерархия на паметта



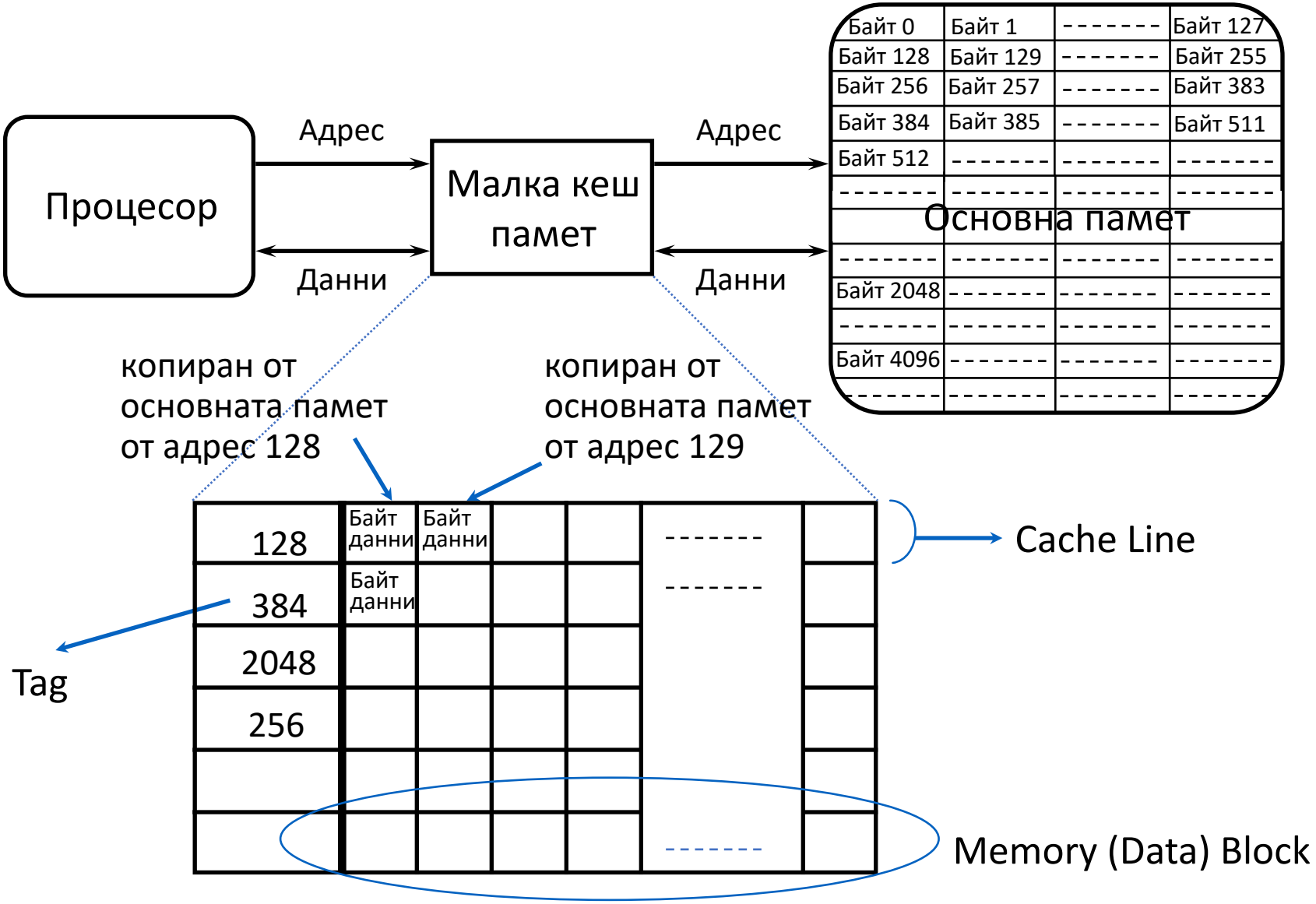
Достъп до паметта

- Процесорът извлича данни и инструкции от паметта
- Посредством адрес процесорът посочва клетка от паметта, за да извлече информацията
- Информацията (данни или инструкции) може да се намира в:
  - Кеш паметта (с малък размер)
  - Основната памет (със среден размер)
  - Вторичната памет (диска) (с много голям размер)
- Къде да се търси?!
- Как да се търси информацията според предоставения от процесора адрес?!

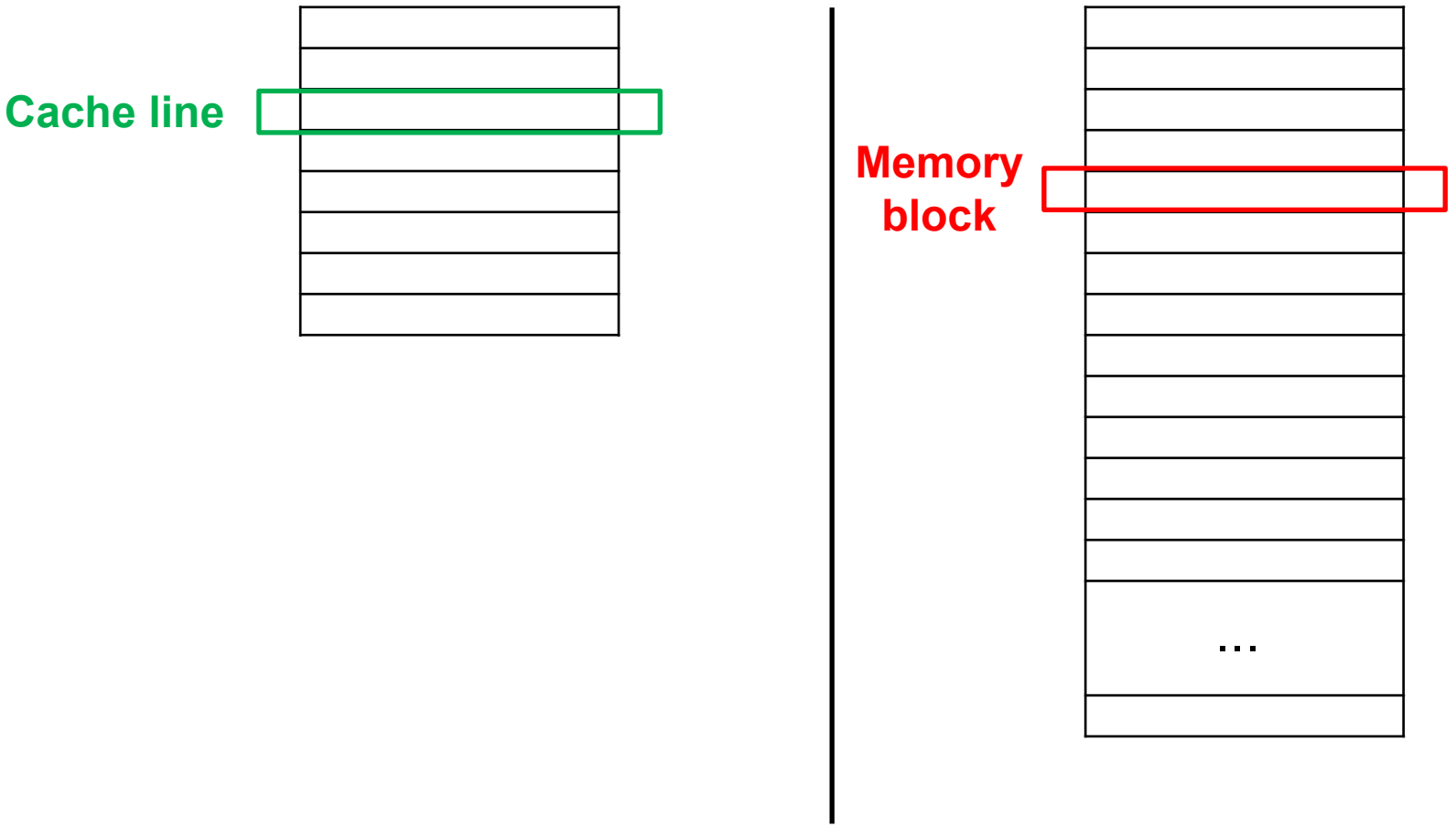
Организация на кеш паметта?



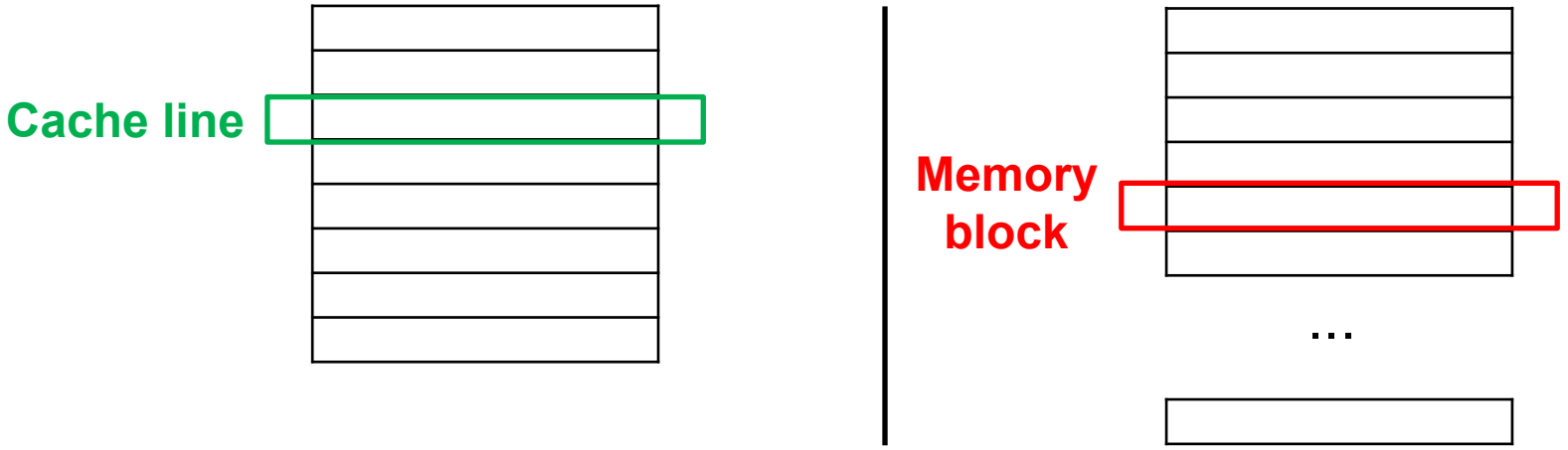
Организация на кеш паметта



Кеш памет ⟷ Основна памет



Кеш памет ↔ Основна памет



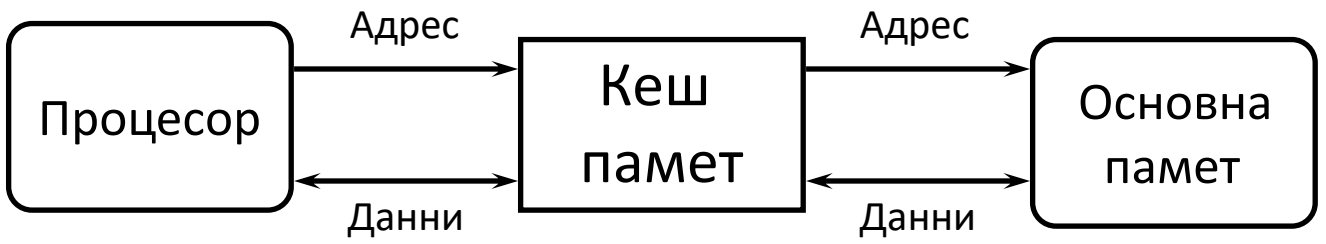
Дължината на един Cache line в байтове = Дължината на един Memory block в байтове

Малко на брой Cache lines

Много на брой Memory blocks

Адресът е според вида на съпоставяне

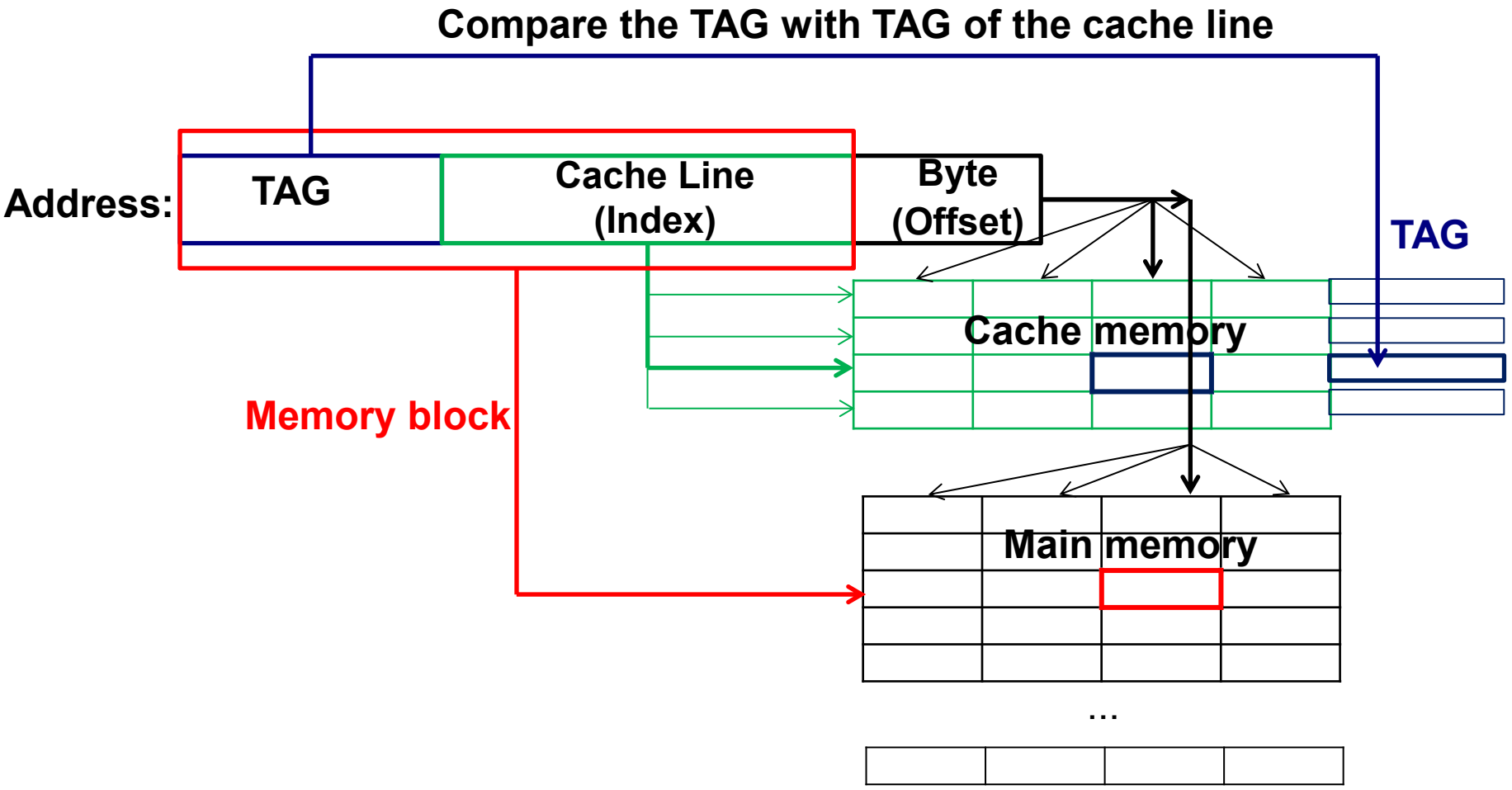
Адресът включва: Memory block + byte



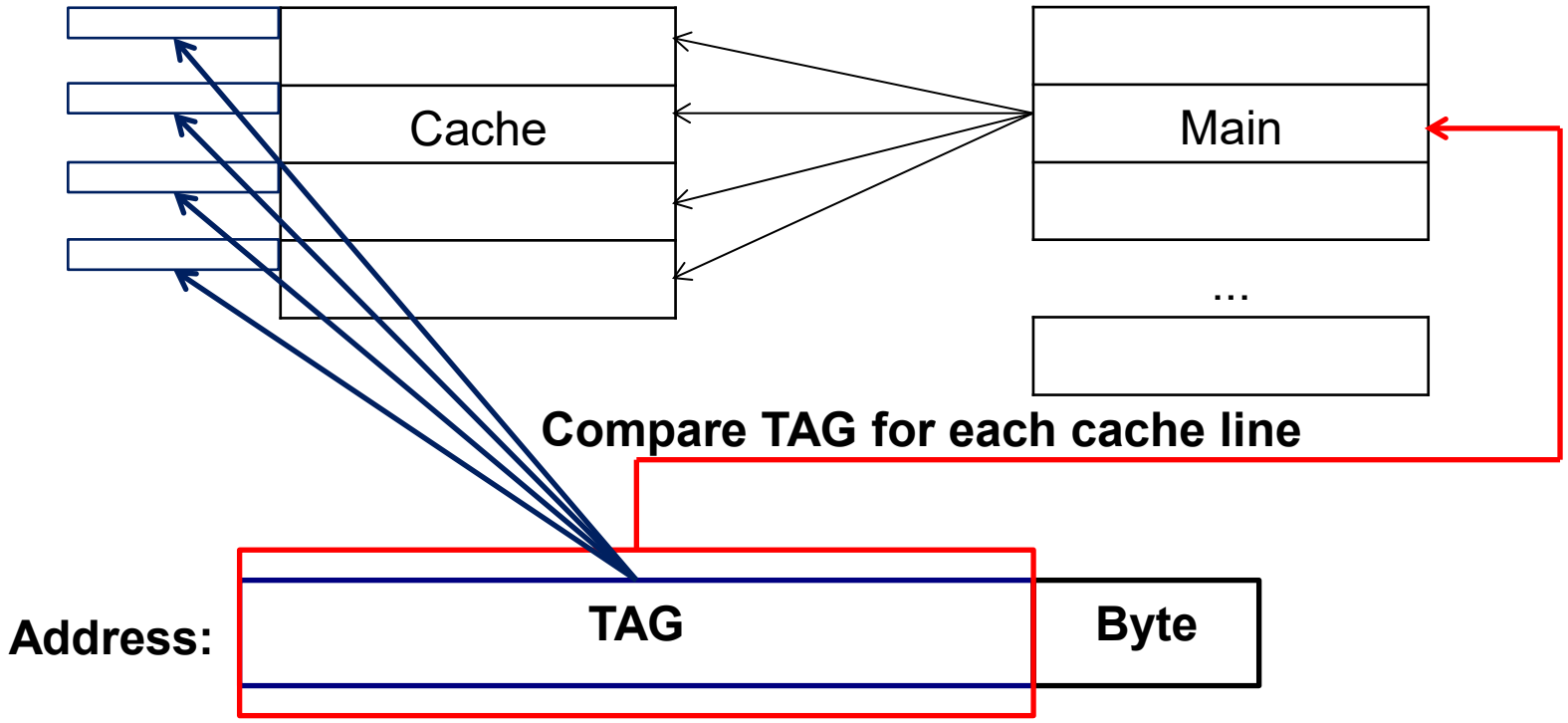
Съпоставка с кеша (Cache mapping)

- Къде в кеш паметта се разполага един блок от основната паметта?
- Пряка съпоставка (Direct mapping)
  - Един „memory block“ може да се постави само в един точно определен „cache line“
- Изцяло асоциативна съпоставка (Fully associative mapping)
  - Един „memory block“ може да се постави във всеки един „cache line“
- Множествено асоциативна съпоставка (Set-associative mapping или N-way associative)
  - A memory block maps into N cache lines

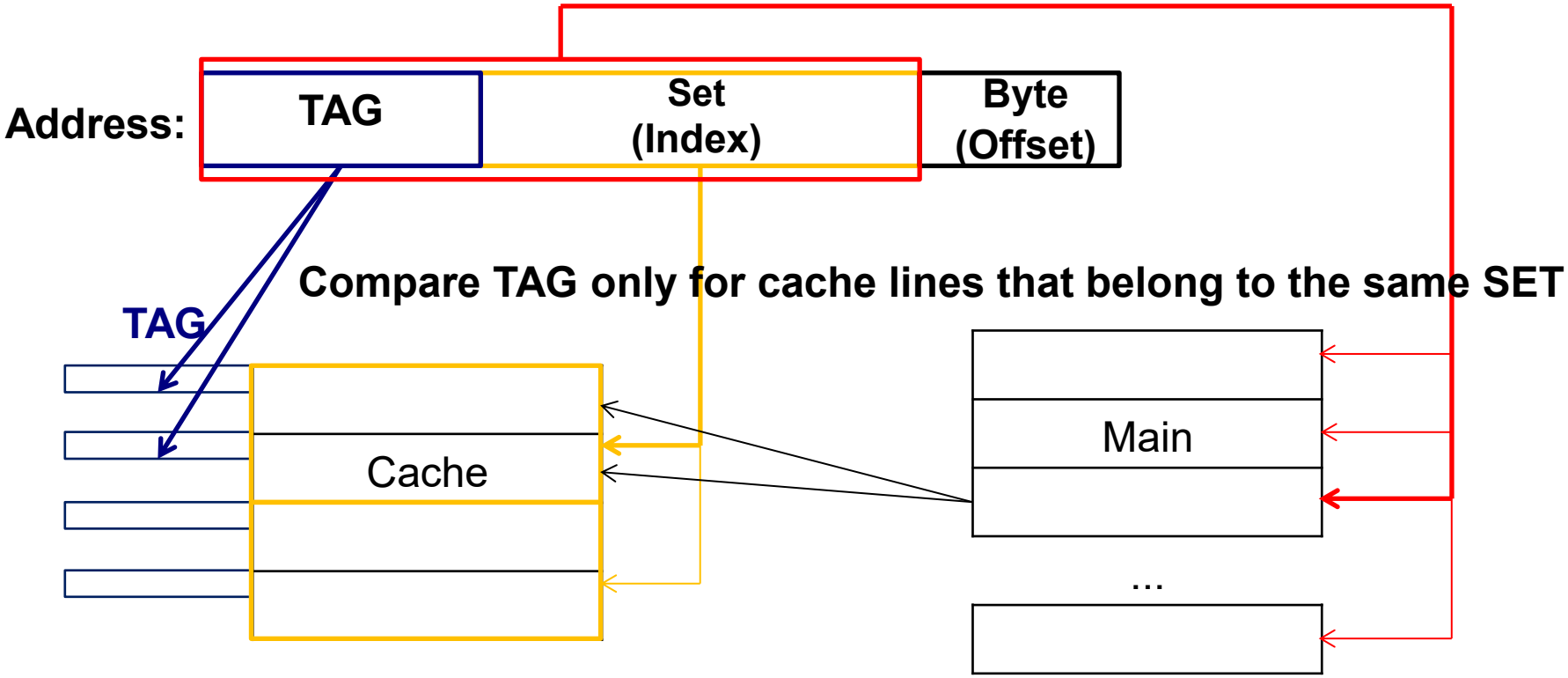
Кеш с пряка съпоставка (Direct mapping Cache)



Кеш с изцяло асоциативна съпоставка (Fully associative cache)



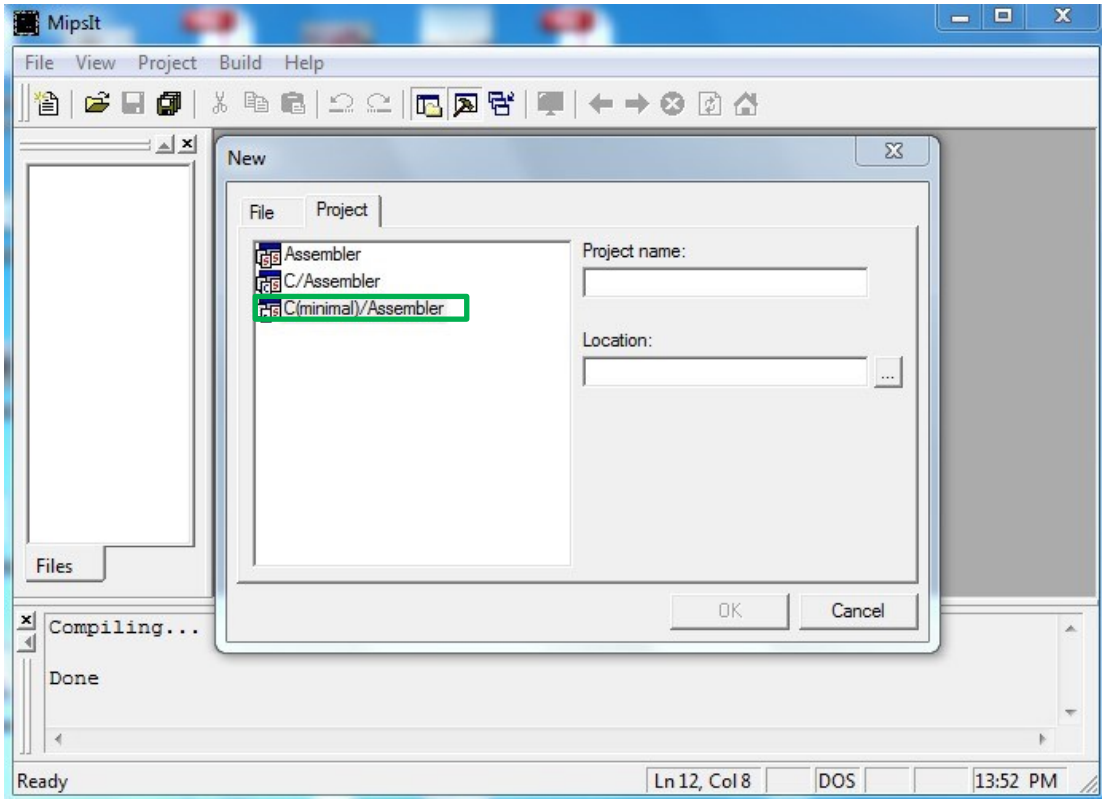
Кеш с множествено асоциативна съпоставка (Set Associative cache)



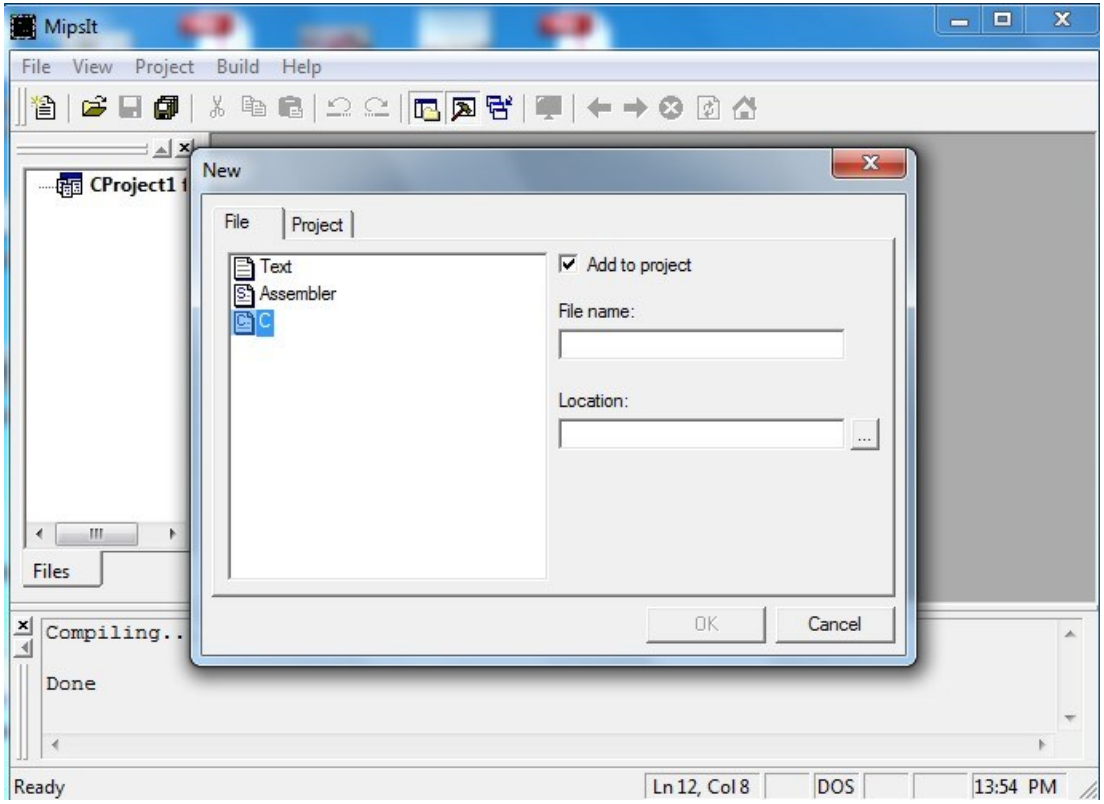
План

- Теоретична част
- Практическа част
  - Симулация в MipsIT
  - Задачи и въпроси

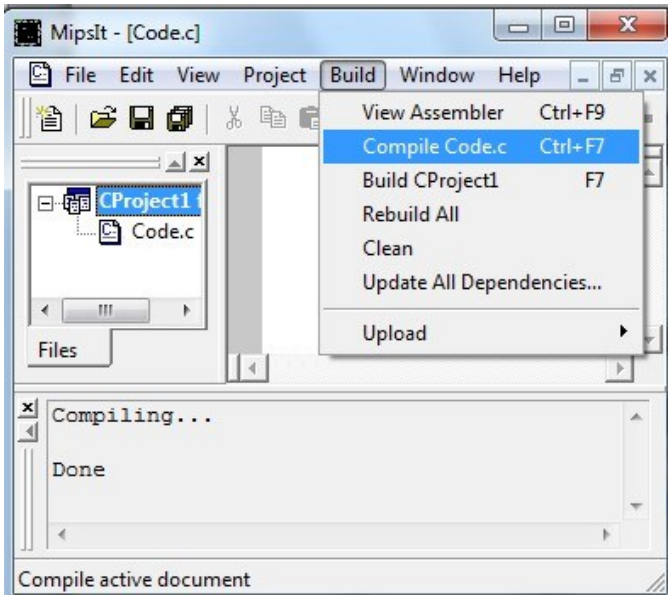
Създаване на нов проект

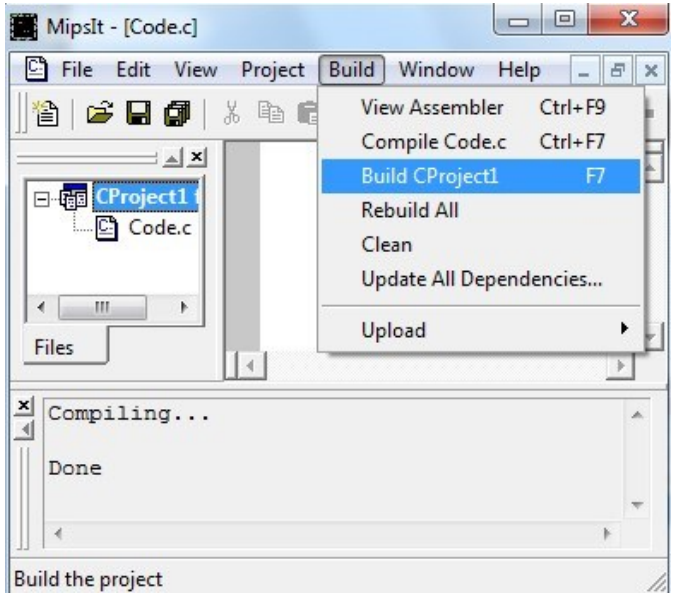


Добавяне на нов сорс файл

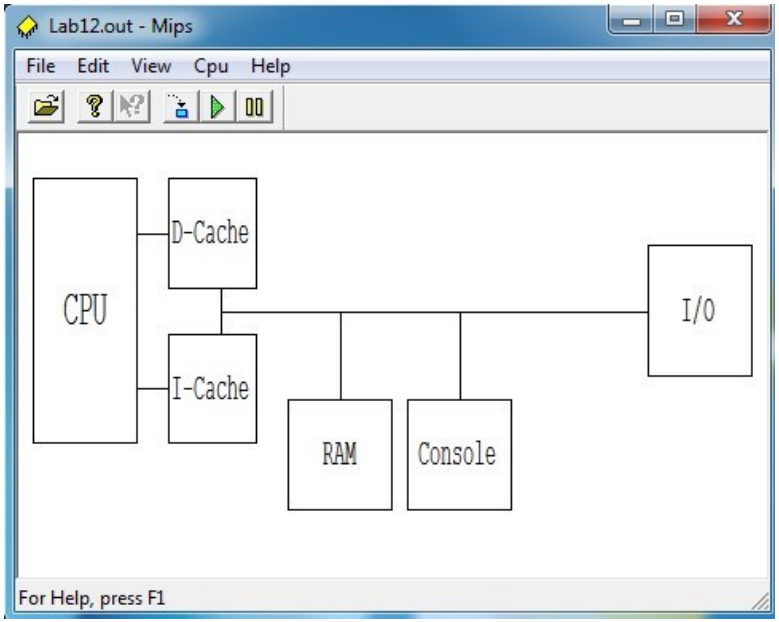


Компилиране

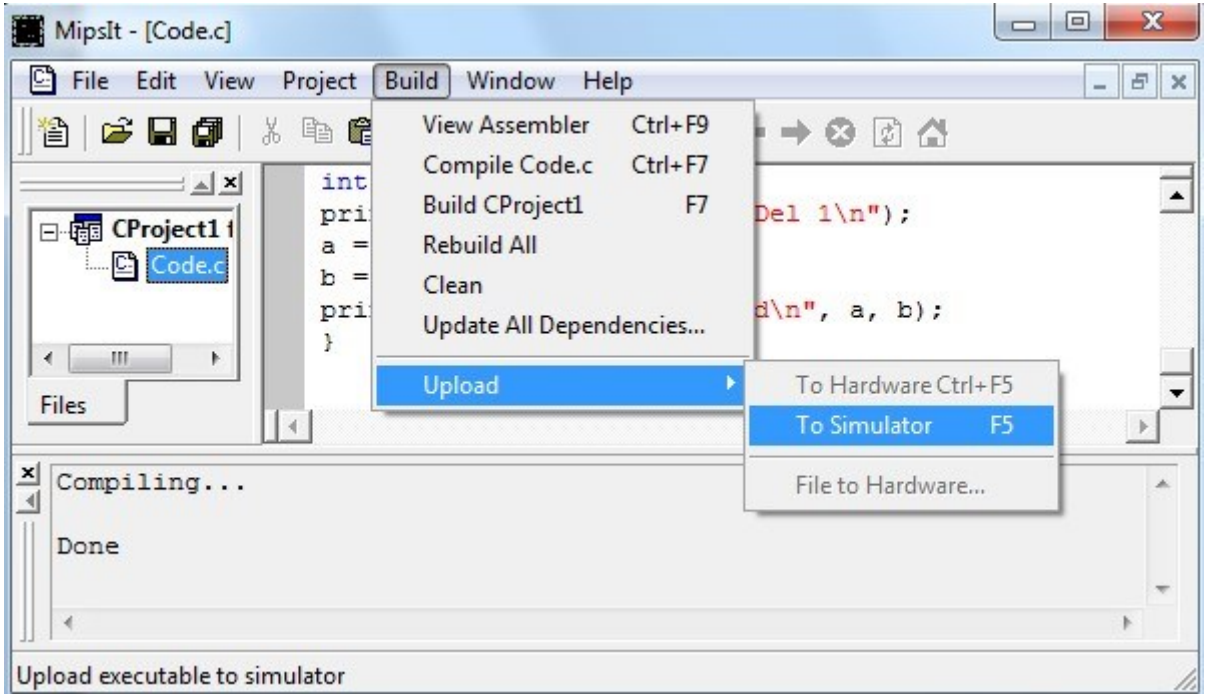




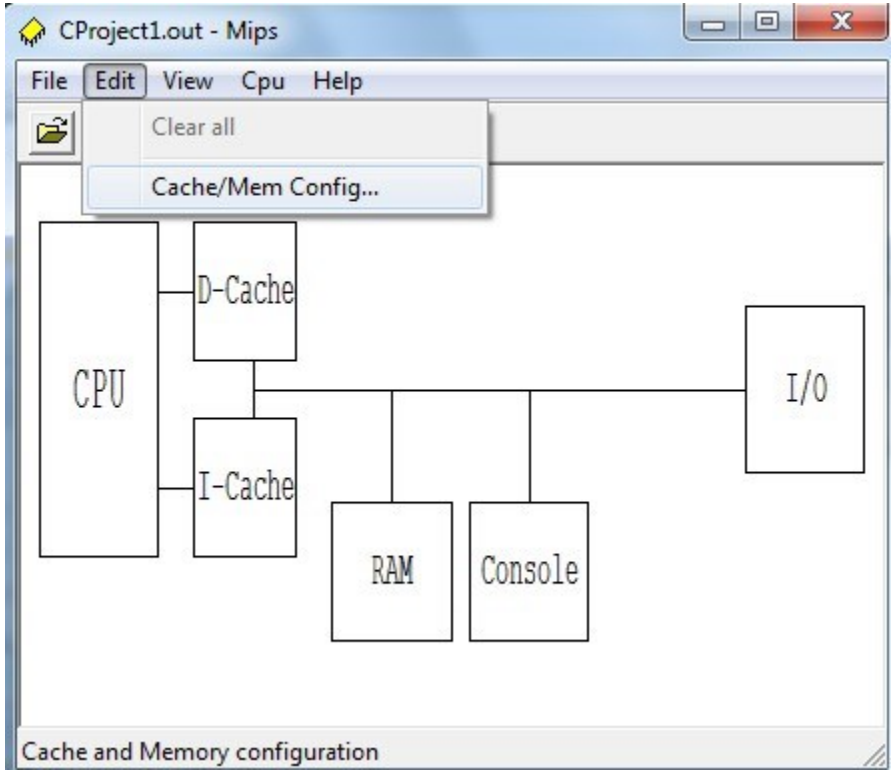
Отваряне на програмата за симулация MIPS



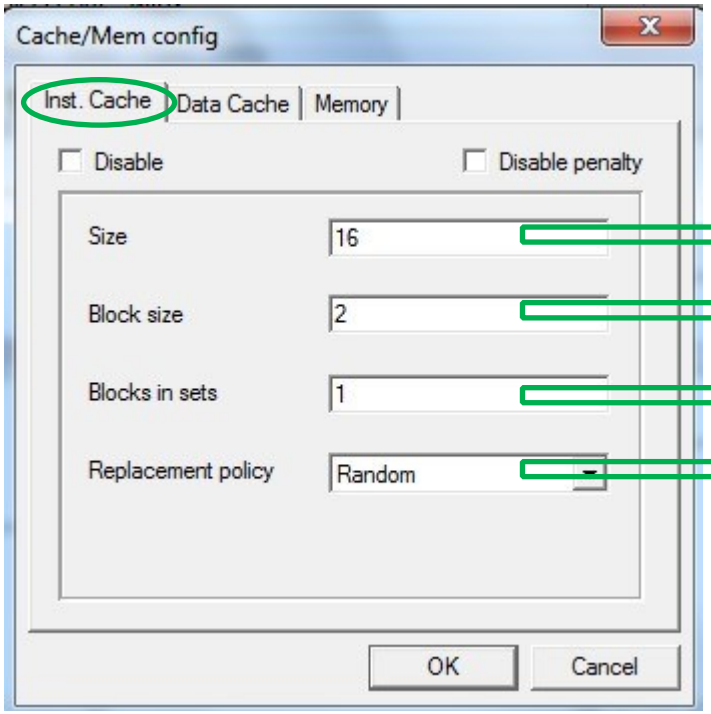
Зареждане на компилираната програма в симулатора



Конфигуриране на кеш паметта

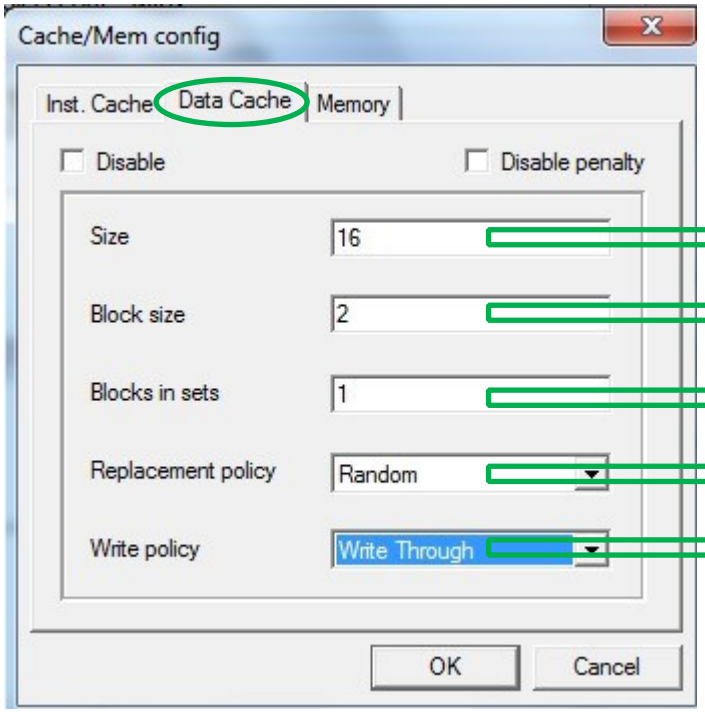


Конфигуриране на кеш паметта за инструкции



- Размер на кеша в думи
- Размер на кеш линията в думи
- Брой блокове в множество
- Политика за заместване

Конфигуриране на кеш паметта за данни



Размер на кеша в думи

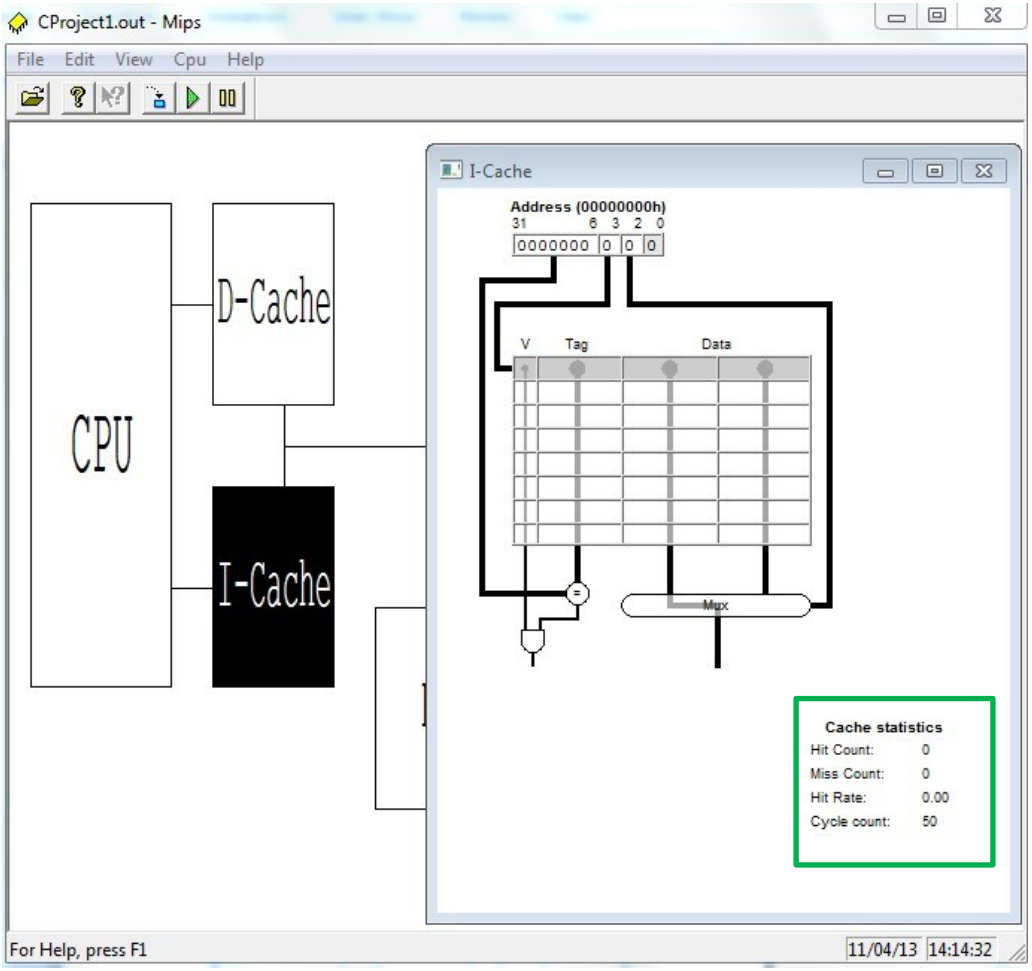
Размер на кеш линията в думи

Брой блокове в множество

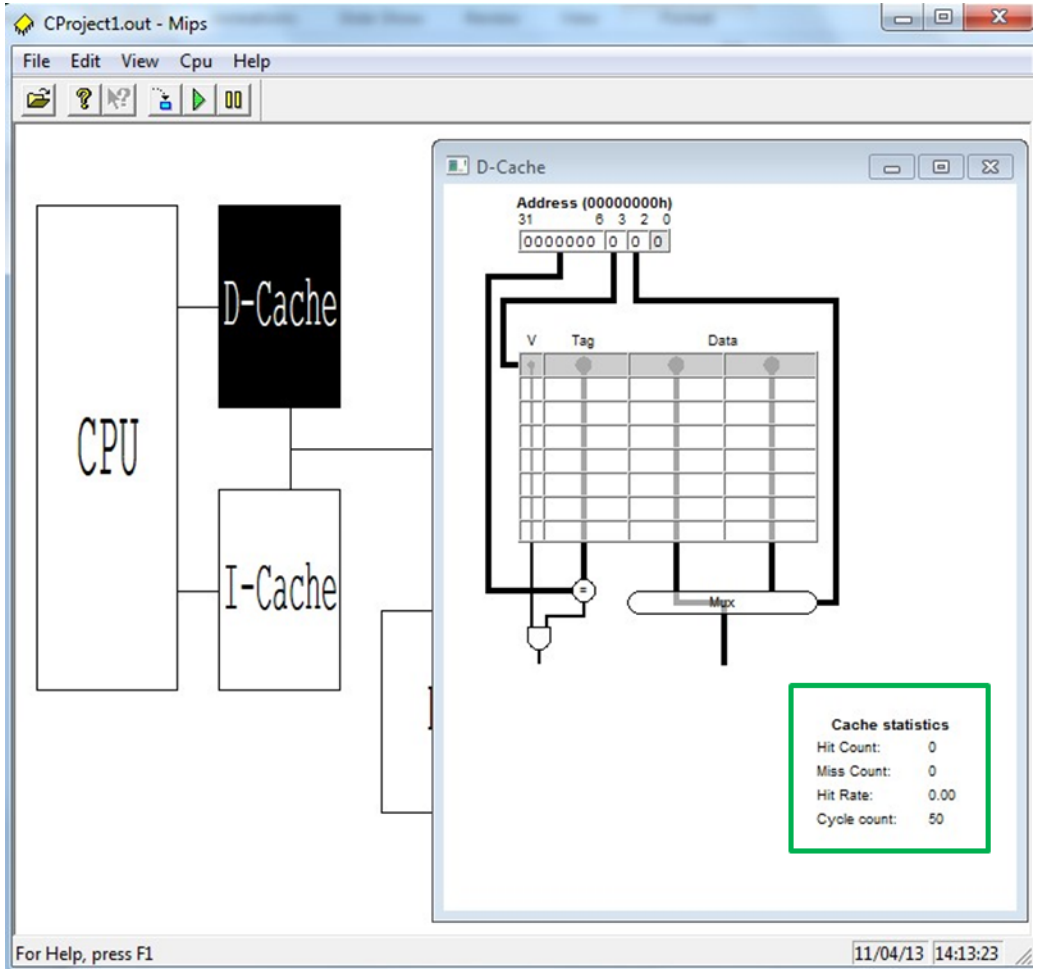
Политика за заместване

Политика при запис

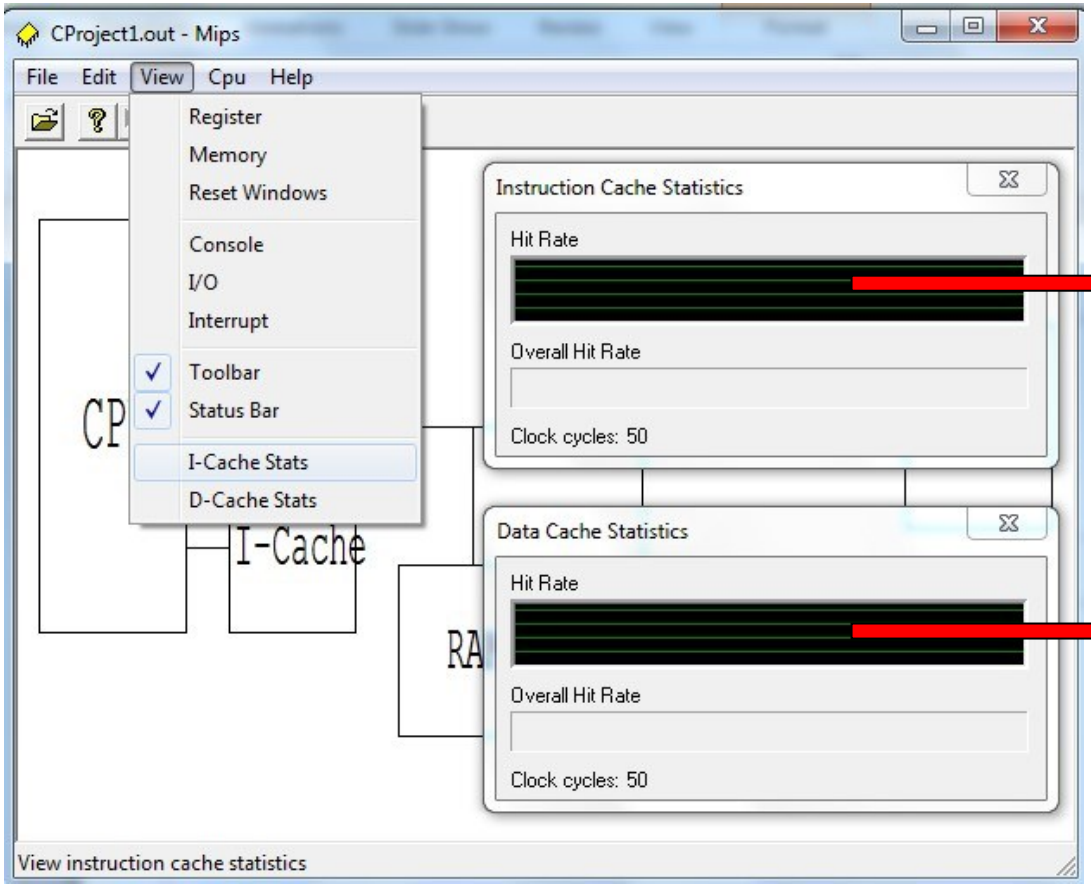
Кеш памет за инструкции (I-cache)



Кеш памет за данни (D-cache)



Резултати от симулацията



Промяна на параметъра „Hit rate“ във времето

План

- Теоретична част
- Практическа част
  - Симулация в MipsIT
  - Задачи и въпроси

Задача 1

- С помощта на инструмента „Address Bit Partitioner“ определете как трябва да се разпределят битове от адреса за достъп до следната система на паметта:
  - Размер на основната памет: 32 МБ
    - *Memory size = 32MB*
  - Размер на кеш паметта: 64 КБ
    - *Cache size = 64 KB*
  - Размер на блока: 16 байта
    - *Block size = 16 Bytes*
  - Множествено асоциативна с 8 блока в множество
    - *Set associative with 8 blocks per set.*

Задача 2

- Дайте пояснение за всяко от изброените:
  - Блок
  - Размер на кеш паметта
  - Размер на блока
  - Брой на множествата

Задача 3

- На следващия слайд е дадена програма на „C”, с две функции.
- И двете функции връщат като резултат сумата от всички елементи в двумерен масив.
- Разликата между функциите е, че обхождат елементите в масива в различен ред.
- Това на пръв поглед изглежда маловажно, но когато се използва кеш памет може да се отчете извесна разлика.
- Разгледайте внимателно програмата, за да разберете в какъв ред се използват елементите от масива.

Задача 3

```
#include <stdio.h>
#include <idt_entrypt.h>
#define N 10
int A[N][N];
// Функция 1
int SumByColRow (int Matrix[N][N]) {
    int i, j, Sum = 0, Time;
    flush_cache();
    timer_start();
    for (j = 0; j < N; j ++) {
        for (i = 0; i < N; i ++) {
            Sum += Matrix[i][j];    }    }
    Time = timer_stop();
     printf("SumByColRow time: %d\n", Time);
     return Sum; }  // Продължава …
```

```
// Функция 2
int SumByRowCol (int Matrix[N][N]) {
     int i, j, Sum = 0, Time;
    flush_cache(); timer_start();
    for (i = 0; i < N; i ++) { for (j = 0; j < N; j ++) {
                Sum += Matrix[i][j] } }
    Time = timer_stop();
    printf("SumByRowCol time: %d\n", Time);
    return Sum; }
main () {
     int a, b;
     printf ("Laboratory Exercise, Task 3\n");
     // Run one of the cases below;
    //comment out the other
     // Case 1
     a = SumByColRow (A);
     printf ("The sum is %d\n", a);
      // Case 2 b = SumByRowCol (A);
     //printf ("The sum is %d\n", b); }  //Край
```

Задача 4

- Създайте проект в средата за разработка MIPS.
- Въведете горната програма, компилирайте и заредете в симулатора (mips.exe).
- Стартирайте програмата в симулатора с настройките по подразбиране и изследвайте как работи кеш паметта за инструкции.
- Попълнете таблицата.

Настройки	Функция от програмата	I-cache Hit rate	D-cache Hit rate	Simulation time
По подразбиране	SumByColRow			
	SumByRowCol			

Въпроси

- Как 32-битовият адрес се използва за достъп до кеш паметта?
- Какво се случва когато няма съвпадение в кеша, т. е. когато се получи „cache miss“?
- Какво се случва когато има съвпадение, т. е. когато се получи „cache hit“?
- Какъв е размерът на блока?
- За какво служи т. нар. маркер (tag)?

Задача 5

- Попълнете следващата таблица, като стартирате програмата с показаните настройки.
- Изследвайте ефекта от промяната на параметрите.

Настройки	Функция от програмата	I-cache Hit rate	D-cache Hit rate	Simulation time
I-cache and D-cache: cache size=32; Others: default	SumByColRow			
	SumByRowCol			
I-cache and D-cache: blockSize=8; Others: default	SumByColRow			
	SumByRowCol			
I-cache and D-cache: Number of blocks in sets =2; Others: default	SumByColRow			
	SumByRowCol			
D-cache Write: policy=WriteBack; Others: default	SumByColRow			
	SumByRowCol			
D-cache: Replacement policy=FIFO; Others: default	SumByColRow			
	SumByRowCol			

Задача 6

- Компилирайте горната програмата, като преди това от менюто Project->Settings повишите нивото на оптимизация, чрез Optimization level = 3 (High).
- Симулирайте програмата в Mips, като зададете настройките от следващата таблица.
- Изследвайте как работи кеш паметта за данни и попълнете таблицата.

Настройки	Функция от програмата	I-cache Hit rate	D-cache Hit rate	Simulation time
I-cache: Disable Penalty(*) D-cache: cache size=64; block size= 16; Others: Default	SumByColRow			
	SumByRowCol			

Въпроси

- Изследвайте внимателно функциите SumByColRow и SumByRowCol.
- Обяснете в какъв ред се обхождат адресите в едната и в другата функция.
- Установете какви са стойностите за „cache hits” в двете функции. Има ли разлика? Защо?
- Ще ви бъде от помощ, ако може да откриете в какъв ред се разполагат елементите на двумерния масив в основната памет.

Заключение

- След приключване на упражнението бихте могли да дискутирате и да разбирате следните въпроси:
  - Каква е основната идея на кеш паметта?
  - Как размерът на блока влияе на ефективността на кеш паметта?
  - Какво е относителното бързодействие на кеш паметта спрямо основната DRAM памет?
  - Оптимални ли са параметрите на кеш паметта според програмния код?
  - Как могат да се изберат оптимални параметри за кеш паметта?