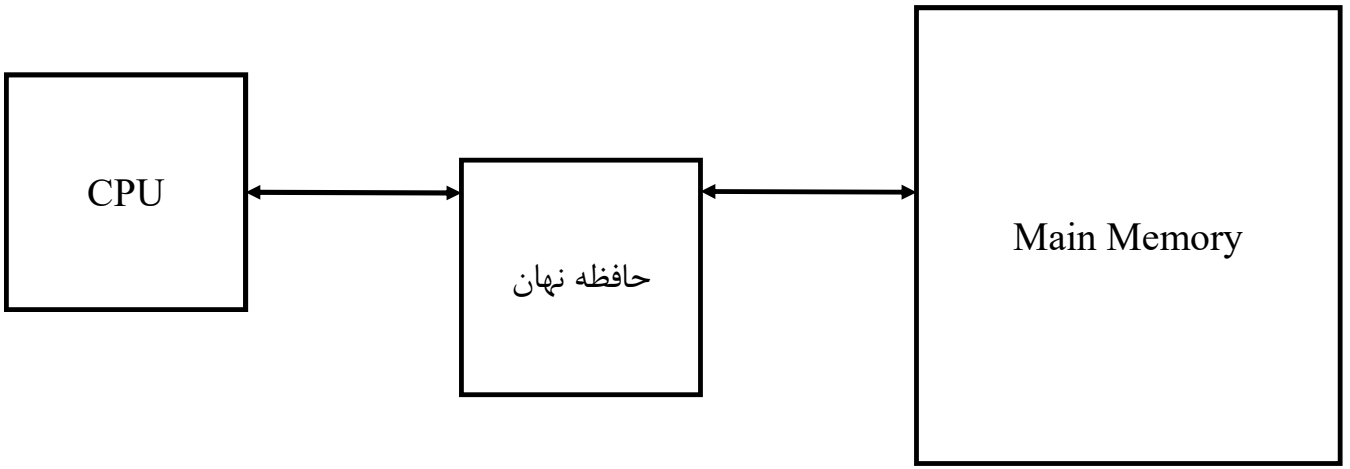
بنام خداوند جان و خرد

معماری کامپیوتر

جلسه هفتم: حافظه نهان-۱

حافظه نهان (Cache)

- درنظر گرفتن فضایی بین پردازنده و حافظه اصلی (مشابه جعبه کوچک کتابدار)
  - باهدف ذخیره‌سازی بخشی از حافظه اصلی جهت دسترسی سریع‌تر



حافظه نهان (Cache)

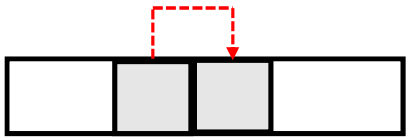
- **سوال اساسی:** در حافظه نهان چه اطلاعاتی ذخیره کنیم؟
  - **هدف اصلی:** افزایش سرعت دسترسی ⟵ نرخ hit بالا
  - **طراحی بد:** متوسط زمان دسترسی بیشتر نسبت به حالت دسترسی مستقیم به حافظه اصلی
- انتخاب داده‌های حافظه نهان، براساس پیشینه درخواست‌ها از حافظه اصلی
  - استخراج ویژگی‌های درخواست‌ها با نظارت بر انتقالات داده بین پردازنده و حافظه اصلی
  - خاصیت محلیت (locality) در داده‌ها

حافظه نهان (Cache)

- **خاصیت هم‌جواری (Locality):**
  - **هم‌جواری مکانی (Spatial Locality)**
  - **هم‌جواری زمانی (Temporal Locality)**
- **لحاظ کردن این خاصیت در طراحی حافظه نهان**

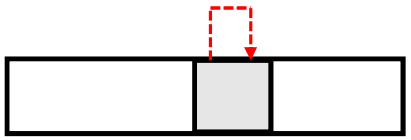
حافظه نهان (Cache)

- **هم‌جواری مکانی (Spatial Locality)**
  - درخواست‌های متوالی از حافظه در آدرس‌های نزدیک به‌هم قرار دارند و لذا وابستگی مکانی بین آن‌ها قابل استخراج می‌باشد.
    - در حلقه for تعدادی دستورالعمل متوالی داریم
    - زمانی که به یک کتاب در حیطه معماری کامپیوتر مراجعه کنیم به احتمال زیاد به کتاب دیگر در این حوزه هم نیاز داریم



حافظه نهان (Cache)

- **هم‌جواری زمانی (Temporal Locality)**
  - یک داده در زمان‌های نزدیک به‌هم چندین بار مورد استفاده قرار می‌گیرد
    - متغیری که چندین بار در طی اجرای برنامه استفاده می‌شود (مثلا در حلقه‌ها)
    - زمانی که به یک کتاب در حیطه معماری کامپیوتر مراجعه کنیم به احتمال زیاد دوباره هم به آن کتاب مراجعه می‌کنیم و با یک‌بار مسئله حل نمی‌شود.



حافظه نهان (Cache)

هم‌جواری مکانی:

داده: ارجاع متوالی به خانه‌های آرایه

دستورالعمل: ارجاع به دستورات به‌صورت ترتیبی

هم‌جواری زمانی:

داده: دسترسی متوالی به متغیر sum در حلقه

دستورالعمل: چرخش متوالی در حلقه

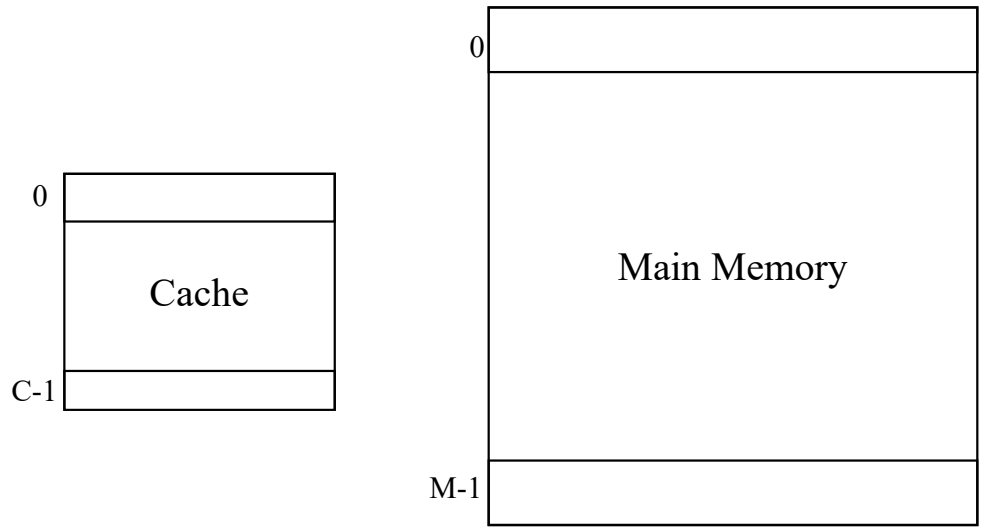
```
sum = 0;
for (i = 0; i < n; i++)
    sum += a[i];
return sum;
```

حافظه نهان (Cache)

- **سوال‌های مهم:**
  - داده‌ها را چگونه از حافظه اصلی به حافظه نهان ببریم؟
  - داده‌های جدید را چگونه در حافظه نهان جایگزین کنیم؟
- **برای پاسخ به این دو سوال، دو بحث اساسی مطرح می‌شود:**
  - سیاست جای‌دهی (Placement Policy)
  - سیاست جای‌گزینی (Replacement Policy)

حافظه نهان (Cache)

- فرض کنیم حافظه نهان C خط (Line) به‌طول یک کلمه داشته باشد و حافظه اصلی M خط
- C<M



سیاست جای‌دهی و انواع حافظه نهان

۱- حافظه نهان نگاشت مستقیم (Direct mapping):

- **اولین گام:** سیاست نگاشت خانه‌های حافظه اصلی به حافظه نهان (Address Mapping)
- مادامی که حافظه نهان خالی است نگاشت بدین‌صورت انجام می‌گیرد:

Cache Address = (Main Memory Address) mod (C)

- سیاست نگاشت آدرس باید پوشا و یکتا باشد
- روش بیان شده **پوشا** هست ولی **یکتا** نیست

حافظه نهان نگاشت مستقیم

- **علت یکتا نبودن نگاشت آدرس پیشنهادی:**
  - ذخیره داده‌ای با آدرس X از حافظه اصلی در حافظه نهان

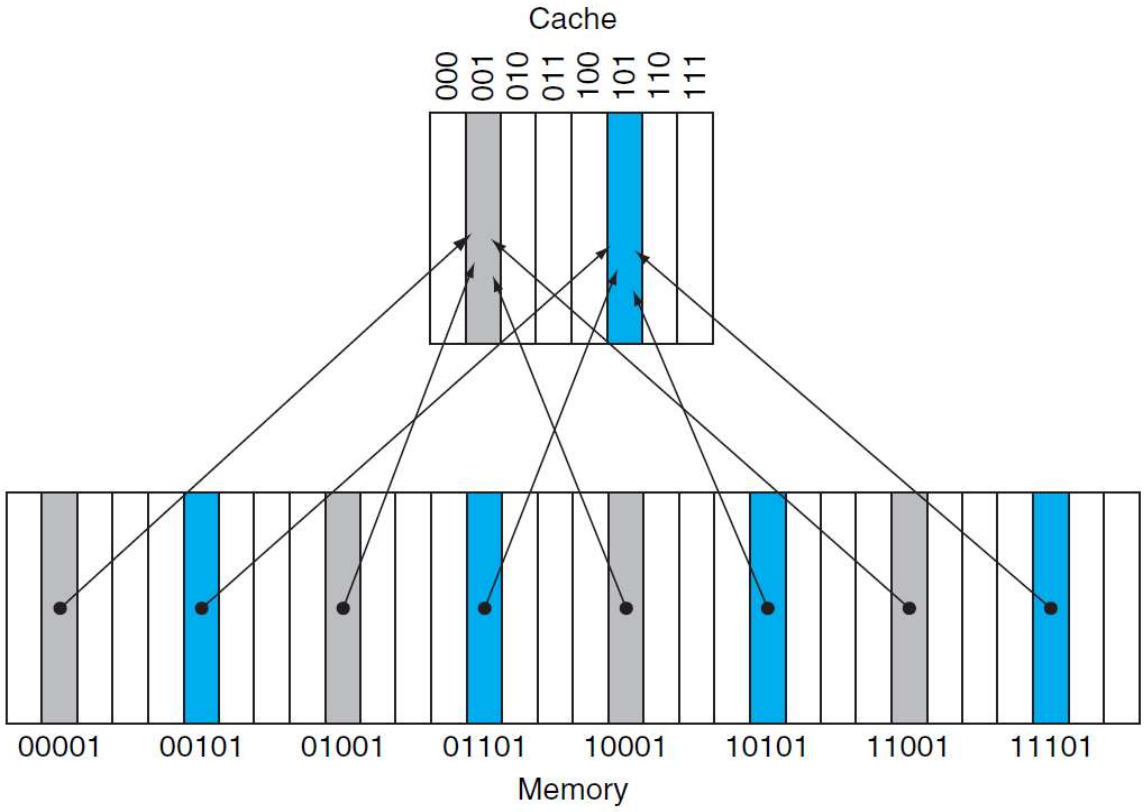
Cache address(X): X mod C

  - ذخیره داده‌ای با آدرس X+C از حافظه اصلی در حافظه نهان

Cache address(X+C): (X+C) mod C

  - هر دو به یک مکان از حافظه نهان اشاره دارند!

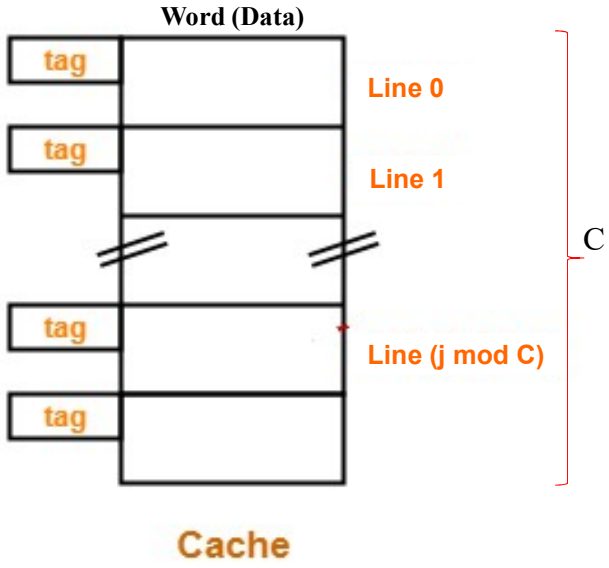
حافظه نهان نگاشت مستقیم



حافظه نهان نگاشت مستقیم

- **راهکار یکتا کردن نگاشت آدرس پیشنهادی:**
  - ذخیره اطلاعاتی از آدرس داده در حافظه اصلی در کنار داده در حافظه نهان
  - اندیس آدرس در حافظه نهان: باقی‌مانده تقسیم آدرس در حافظه اصلی بر C
  - استفاده از خارج قسمت این تقسیم به‌عنوان شناسه‌ای از آدرس داده
    - ذخیره این شناسه به‌عنوان برچسب (Tag) داده در حافظه نهان

حافظه نهان نگاشت مستقیم



حافظه نهان نگاشت مستقیم

Index	V	Tag	Data
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	N		
111	N		

a. The initial state of the cache after power-on

Index	V	Tag	Data
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	Y	10_{two}	Memory (10110_{two})
111	N		

b. After handling a miss of address (10110_{two})

Word address	Binary address	Hit/miss	Cache block
22	10110	Miss	110

حافظه نهان نگاشت مستقیم

- **گام دوم: شیوه ذخیره‌سازی داده‌ها در حافظه نهان**
  - در طراحی فعلی: هربار یک کلمه از حافظه اصلی به حافظه نهان منتقل می‌شود
  - کارایی cache وابسته به prediction مناسب درخواست‌ها
    - همجواری زمانی و همجواری مکانی
  - اصلاح طراحی برای لحاظ کردن همجواری مکانی
    - انتقال یک بلوک B کلمه‌ای از حافظه اصلی به حافظه نهان در هر بار دسترسی

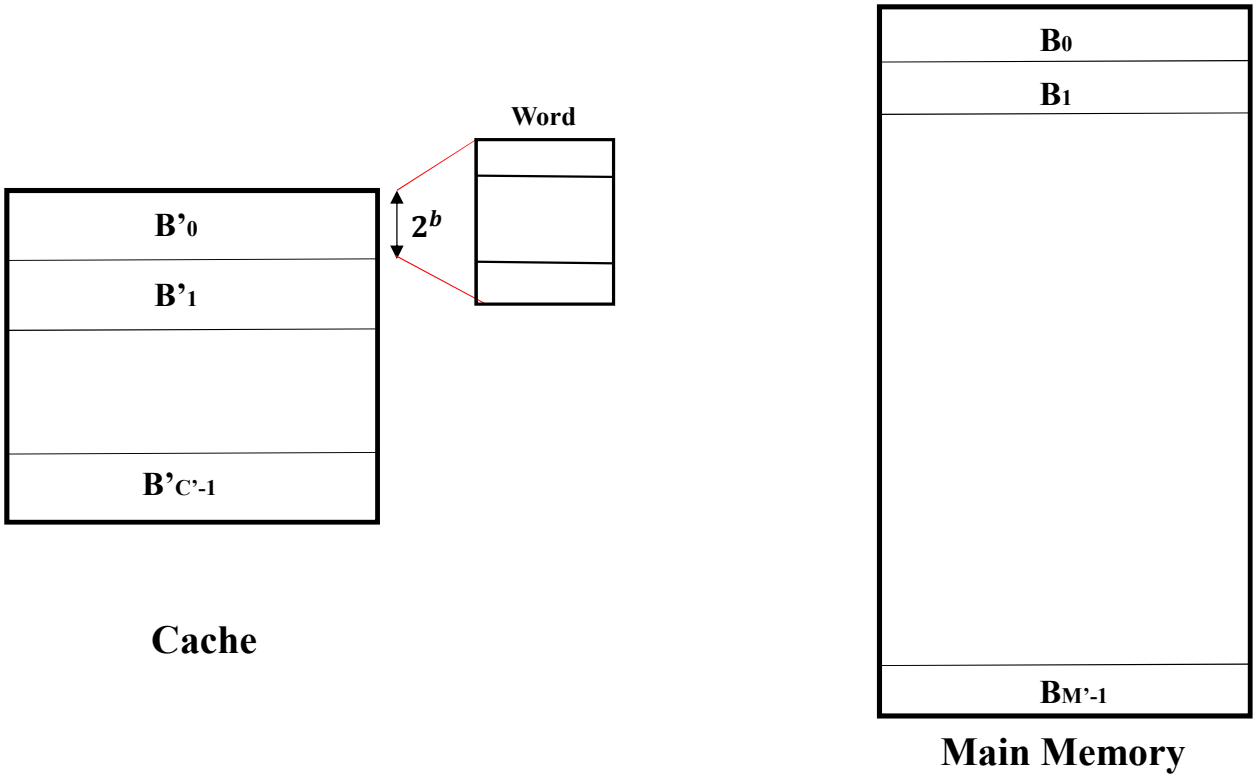
حافظه نهان نگاشت مستقیم

$$\begin{cases} M = 2^m \\ C = 2^c \\ B = 2^b \end{cases}$$

- نکته مهم: M، C و B همگی توان‌هایی از ۲ هستند
- تقسیم حافظه نهان و حافظه اصلی به بلوک‌های B کلمه‌ای
  - همگی توان‌هایی از ۲ هستند پس تقسیم ← شیفت
  - در هربار انتقال کلمه از حافظه اصلی به حافظه نهان، کل بلوکی که داده در آن است منتقل می‌شود
  - **فرض:** حافظه اصلی دارای M’ بلوک B کلمه‌ای و حافظه نهان دارای C’ بلوک B کلمه‌ای (M’=M/B)

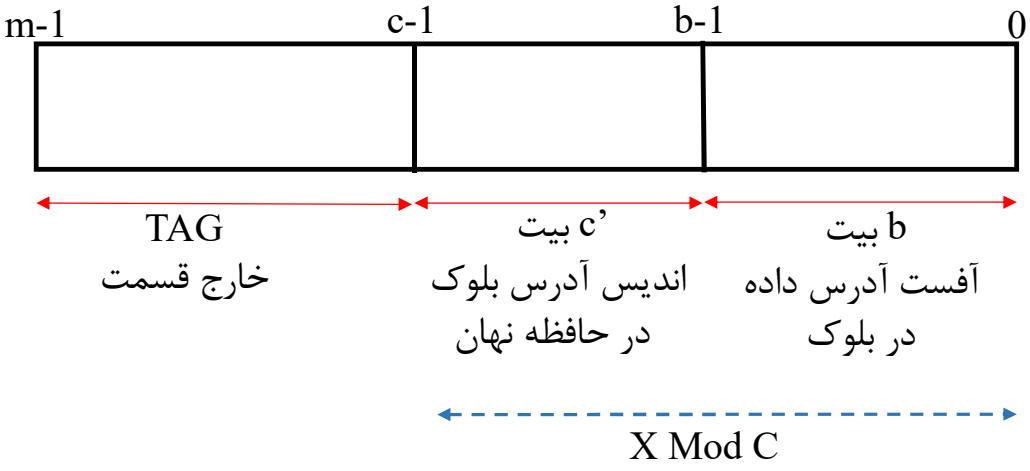
حافظه نهان نگاشت مستقیم

طول هر بلاک: 2^b کلمه



حافظه نهان نگاشت مستقیم

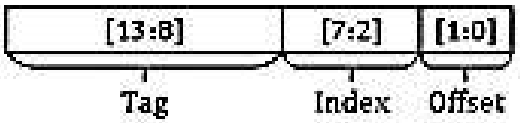
- قالب آدرس حافظه اصلی ورودی به حافظه نهان:

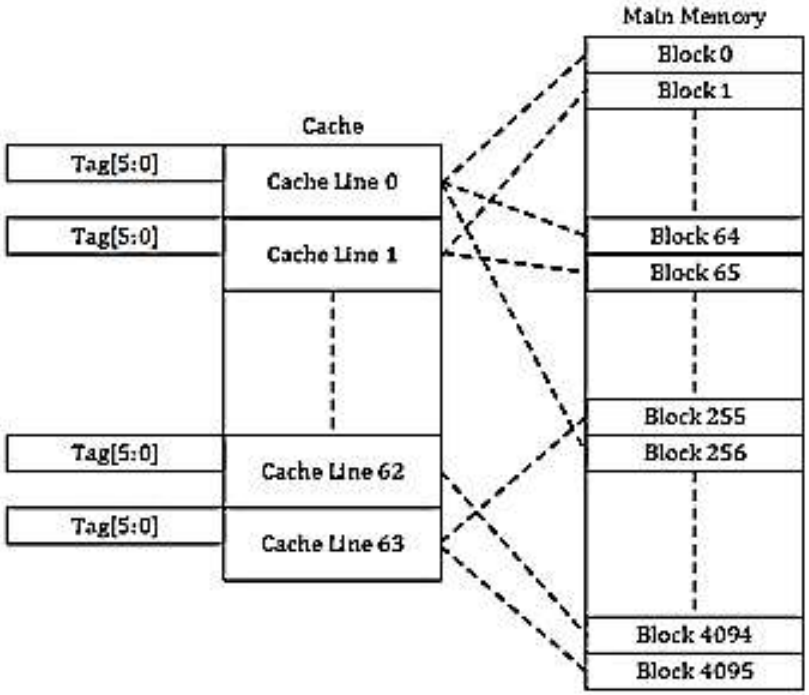


حافظه نهان نگاشت مستقیم

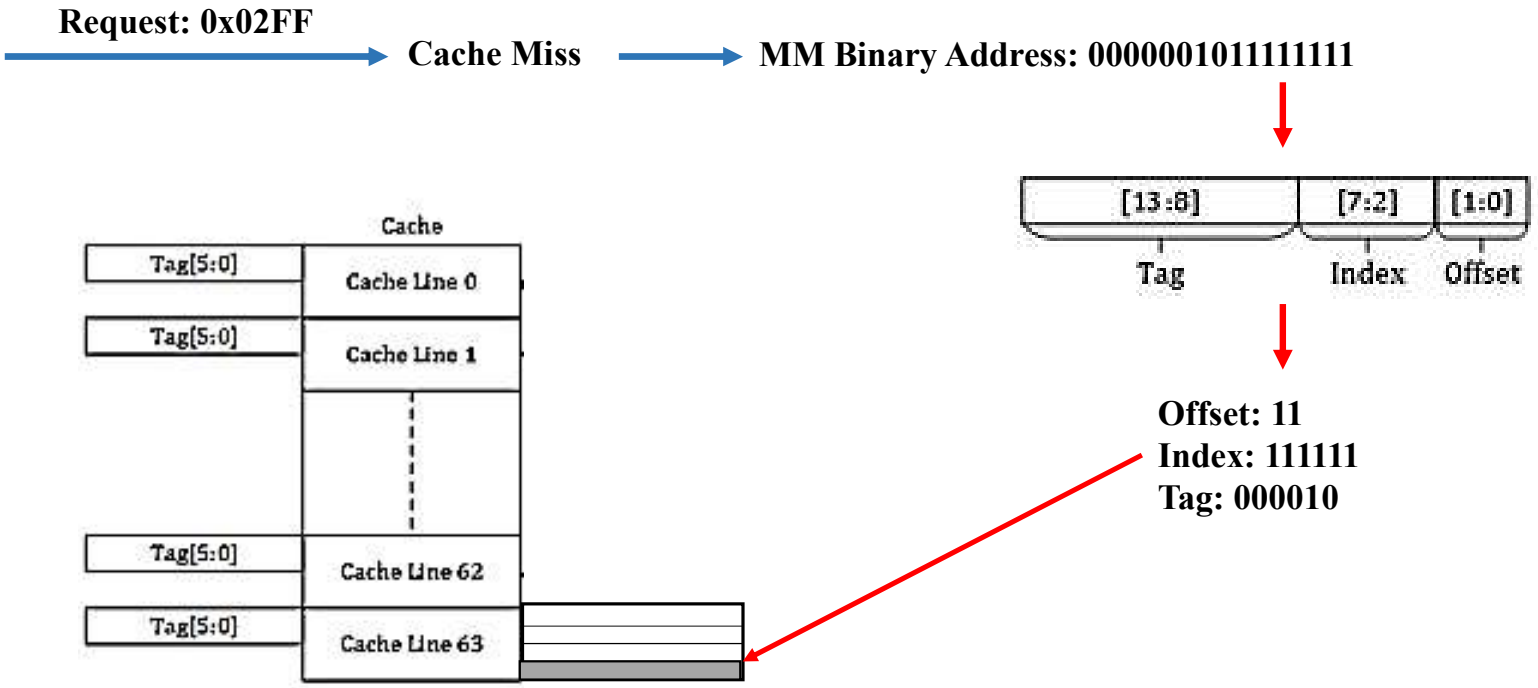
Memory Size (M) = 16 KB, m=14
Block Size (B) = 4 B, b = 2
Cache Size (C) = 256 B, c=8

 # blocks in MM: M/B= 4096
 # blocks in cache: C / B = 64





حافظه نهان نگاشت مستقیم

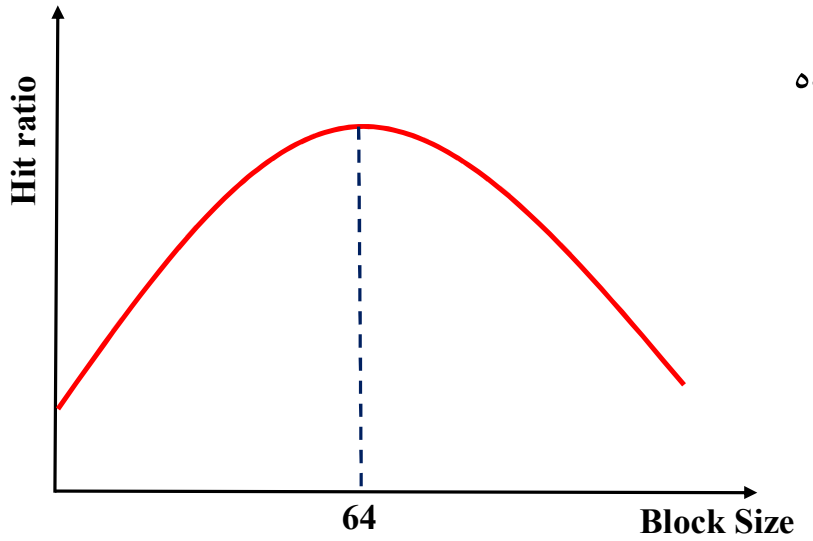


حافظه نهان نگاشت مستقیم

- تمامی اعضای یک بلوک حافظه نهان، Tag مشترک دارند
  - وبرای یک بلوک نگهداری یک برچسب کافی است
- روش نگاشت مستقیم در طراحی حافظه نهان، ساده‌ترین و کم‌هزینه‌ترین روش است
  - طبق آزمایشات و ارزیابی‌های، hit ratio حافظه نهان طراحی شده این روش حدود ۹۲٪

حافظه نهان نگاشت مستقیم

- در این روش، سایز بلوک را می‌توان افزایش داد
  - نمودار شدن بیشتر همجواری مکانی
  - کم شدن تعداد بلوک‌های ذخیره شده و تنوع داده



حافظه نهان نگاشت مستقیم

- **مدیریت درخواست نوشتن در حافظه:**
  - مشابه درخواست خواندن ابتدا درخواست به حافظه نهان می‌رود:
    - Miss: داده در حافظه اصلی پیدا شده و مقدار آن به‌روز می‌گردد و به حافظه نهان منتقل می‌شود.
    - Hit: داده در حافظه نهان موجود است و باید به‌روز شود. این به‌روزرسانی باید در حافظه اصلی هم انجام شود.

(چه‌زمانی؟)

حافظه نهان نگاشت مستقیم

- **مدیریت درخواست نوشتن در حافظه:**
  - چگونگی به‌روزرسانی داده در حافظه اصلی در صورت hit شدن درخواست در حافظه نهان:
    - **Write through**: هرگاه داده‌ای در حافظه نهان به‌روز شد، محتویات آدرس متناظر در حافظه اصلی هم به‌روز می‌شود
    - **Write back**: به‌روز رسانی داده در حافظه اصلی تا زمان پاک شدن بلوک از حافظه نهان به تاخیر می‌افتد.
      - یک بیت کنترلی U برای هر بلوک نگه می‌داریم که نشانگر به‌روز شدن آن بلوک است
      - کارایی بهتری دارد اما ریسک از دست دادن داده جدید وجود دارد