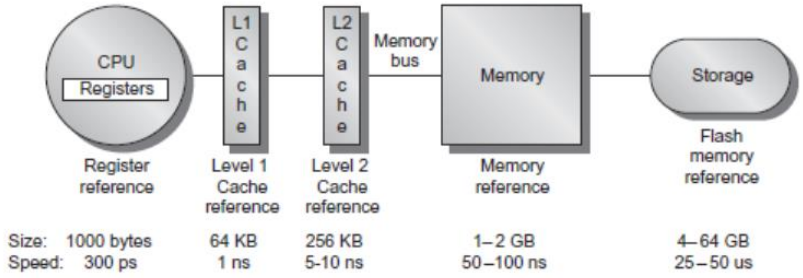
Bellek Performansı

5. HAFTA

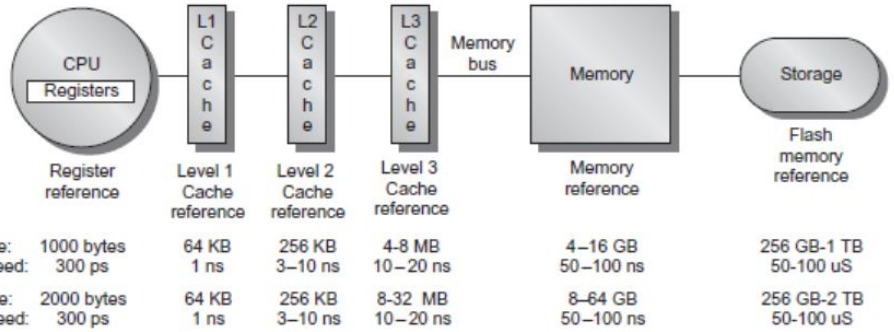
Tanıtım

- Programcılar, düşük gecikmeye sahip sınırsız miktarda bellek ister
- Hızlı bellek teknolojisi, yavaş bellekten bit başına çok daha pahalıdır
- Çözüm: bellek sisteminin hiyerarşik organizasyonu
  - Tüm adreslenebilir bellek alanı en büyük ve en yavaş bellektedir
  - İşlemciye doğru adım adım ilerlerken her biri altındaki belleğin bir alt kümesini içeren giderek daha küçük ve daha hızlı bellekler
- Zamansal ve uzaysal yerellik, neredeyse tüm referansların daha küçük belleklerde bulunabilmesini mümkün kılar.
  - İşlemciye sunulan büyük, hızlı bir bellek gibi davranır

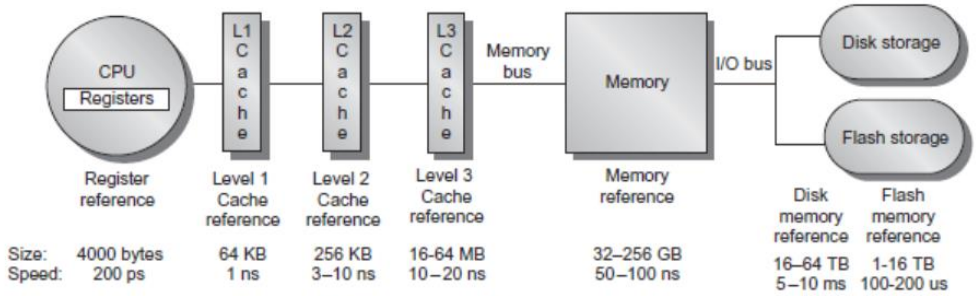
Bellek Hiyerarşisi



(A) Memory hierarchy for a personal mobile device

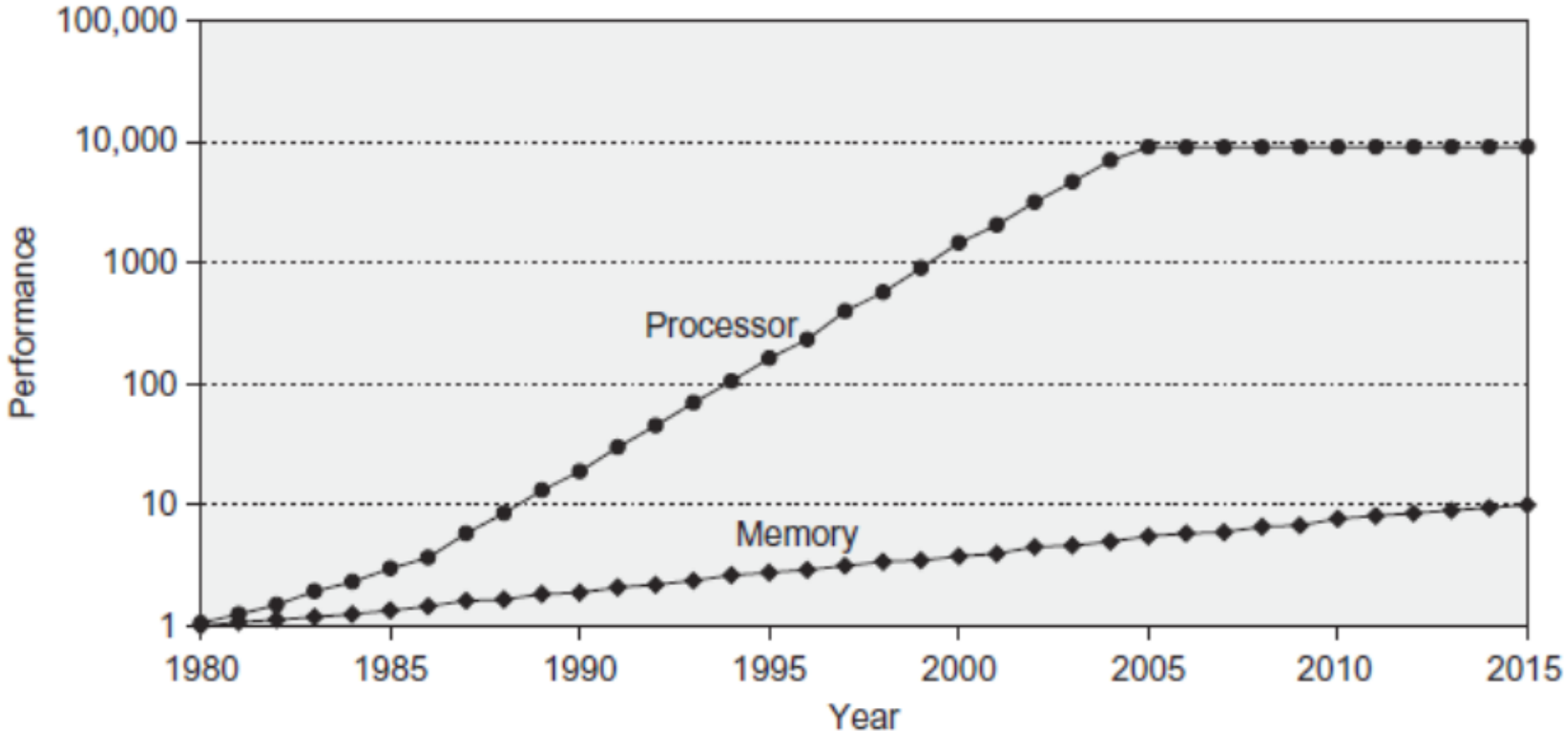


(B) Memory hierarchy for a laptop or a desktop



(C) Memory hierarchy for server

Bellek Performans Boşluğu



Bellek Hiyerarşisi Tasarımı

- Bellek hiyerarşisi tasarımı, son zamanlardaki çok çekirdekli işlemcilerle daha önemli hale geliyor:
  - Bant genişliği çekirdek sayısıyla büyür:
    - Intel Core i7, çevirm başına çekirdek başına iki referans oluşturabilir
    - Dört çekirdek ve 3.2 GHz saat hızı
      - 25,6 milyar 64 bit veri referansı/saniye +
      - 12,8 milyar 128 bit talimat referansı/saniye
      - = 409,6 GB/sn!
    - DRAM bant genişliği bunun yalnızca % 8'idir (34,1 GB/sn)
    - Gereklilikler:
      - Çok bağlantı noktalı, ardışık düzen önbellekleri
      - Çekirdek başına iki seviye önbellek
      - Çip üzerinde üçüncü düzey önbellek

Performans ve Güç

- Üst düzey mikroişlemciler> 10 MB çip üzerinde önbelleğe sahiptir
  - Büyük miktarda alan ve güç - bütçe tüketir

Bellek Hiyerarşisinin Temelleri

- Önbellekte bir kelime bulunmadığında, bir *ıskalama-miss* meydana gelir:
  - Daha yüksek gecikme gerektiren, hiyerarşide daha düşük seviyeden kelime getirilir
  - Alt seviye başka bir önbellek veya ana bellek olabilir
  - *Blok* içinde yer alan diğer kelimeler de getirilir
    - Yerellik prensibinden yararlanır
  - Adrese göre belirlenen, kendi *kümesi* içindeki herhangi bir konuma önbelleğe blok yerleştirilir
    - blok adresi MOD önbellekteki set sayısı

Bellek Hiyerarşisinin Temelleri

- *n* küme => *n-yollu ilişkisel küme*
  - *Doğrudan eşlenmiş önbellek* => küme başına bir blok
  - *Tam ilişkisel* => bir küme

- Önbelleğe yazma: iki strateji
  - *Baştan sona yazma*
    - Alt hiyerarşi seviyelerini hemen güncelle
  - *Geriye doğru yazma*
    - Yalnızca güncellenmiş bir blok değiştirildiğinde alt hiyerarşi seviyelerini güncelle
  - Her iki strateji de yazma işlemlerini asenkron yapmak için *yazma tamponunu kullanır*

Bellek Hiyerarşisinin Temelleri

- Iskalama oranı
  - Bir ıskalama ile sonuçlanan önbellek erişiminin bir oranı
- Iskalama nedenleri
  - Zorunlu
    - Bir bloğa ilk referans
  - Kapasite
    - Bloklar atılır ve daha sonra geri alınır
  - Çakışma
    - Program, önbellekte aynı konuma eşlenen farklı bloklardan birden fazla adrese tekrar tekrar başvuru yapar

Ön Bellek Performansı

- $\frac{Iskalar}{talimat} = \frac{ıska\ oranı\ X\ bellek\ erişimleri}{Iska\ sayısı}$
- $= Iska\ oranı\ x\ \frac{Bellek\ erişimleri}{talimat}$

- Ortalama bellek erişim süresi = isabet süresi + ıskalama oranı ıska cezası

- Spekülatif ve çok iş parçacıklı işlemciler, bir ıskalama sırasında diğer talimatları yürütebilir.
  - Iskalamaların performansa olumsuz etkisini azaltır

Ön Bellek Performansı

- CPU süresi: programları yürütmek için harcanan saat çevrimleri

CPU süresi = (CPU yürütme saat çevrimleri + bellek bekleme saat çevrimleri) x saat çevrim süresi

Bellek bekleme saat çevrimleri = okuma bekleme çevrimleri + yazma bekleme çevrimleri

$$\text{Okuma bekleme çevrimleri} = \frac{\text{okumalar}}{\text{program}} \text{ X okuma ıska oranı x okuma ıska cezası}$$

$$\text{Yazma bekleme çevrimleri} = \left(\frac{\text{yazmalar}}{\text{program}} \text{ X yazma ıska oranı x okuma ıska cezası}\right) + \text{yazma ara bellek beklemeleri}$$

$$\text{Bellek bekleme çevrimi} = \frac{\text{bellek erişimleri}}{\text{program}} \text{ X ıska oranı X ıska cezası}$$

$$\text{Bellek bekleme saat çevrimleri} = \frac{\text{talimatlar}}{\text{program}} \text{ X } \frac{\text{ıskalar}}{\text{talimat}} \text{ X ıska cezası}$$

Örnek 1

- Talimat önbellek ıska oranının %2 olduğunu ve veri ön bellek ıska oranının % 4 olduğunu varsayalım. İşlemci hiçbir bellek bekletmesi olmaksızın 2 CPI değerine sahipse ve tüm ıskalar için ıska cezası 100 çevrim ise ıska yapmayan mükemmel bir ön bellek ile bir işlemcinin ne kadar hızlı çalışacağını belirleyiniz (tüm yükleme ve saklama frekansı %36 olarak varsayalım).

Örnek 2

- Talimat önbellek ıska oranının %2 olduğunu ve veri ön bellek ıska oranının % 4 olduğunu varsayalım. İşlemci hiçbir bellek bekletmesi olmaksızın 2 CPI değerine sahipse ve tüm ıskalar için ıska cezası 100 çevrim ise ıska yapmayan mükemmel bir ön bellek ile bir işlemcinin ne kadar hızlı çalışacağını belirleyiniz (tüm yükleme ve saklama frekansı %36 olarak varsayalım).

Talimat ıska çevrimi = I X 0,02 X 100 = 2 . I

Veri ıska çevrimi = I x 0,36 X 0,04 X 100 = 1,44 . I

Bellek bekleme çevrimleri = 2I + 1,44I = 3,44.I

Toplam CPI = 2 + 3,44 = 5,44

$$\frac{Beklemeli\ CPU\ süresi}{Mükemmel\ önbellekte\ CPU\ süresi} = \frac{I\ x\ CPI\ bekleme\ x\ saat\ çevrimi}{I\ x\ CPI\ mükemmel\ x\ saat\ çevrimi} = \frac{5{,}44}{2} = 2{,}72\ kat$$

Örnek 2

- 1ns saat çevrim süreli, 20 ıska cezalı, komut başına 0,5 ıska oranlı ve 1 çevrim önbellek erişim süreli bir işlemci için ortalama bellek erişim süresini bulunuz.

Ortalama bellek erişim süresi = isabet süresi + ıska oranı X ıska cezası

= 1 + 0,5 . 20

= 2 çevrim

=2 çevrim X 1ns = 2ns dir

Örnek 3

Example Assume we have a computer where the cycles per instruction (CPI) is 1.0 when all memory accesses hit in the cache. The only data accesses are loads and stores, and these total 50% of the instructions. If the miss penalty is 50 clock cycles and the miss rate is 1%, how much faster would the computer be if all instructions were cache hits?

Answer First compute the performance for the computer that always hits:

$$\begin{aligned}\text{CPU execution time} &= (\text{CPU clock cycles} + \text{Memory stall cycles}) \times \text{Clock cycle} \\ &= (\text{IC} \times \text{CPI} + 0) \times \text{Clock cycle} \\ &= \text{IC} \times 1.0 \times \text{Clock cycle}\end{aligned}$$

Now for the computer with the real cache, first we compute memory stall cycles:

$$\begin{aligned}\text{Memory stall cycles} &= \text{IC} \times \frac{\text{Memory accesses}}{\text{Instruction}} \times \text{Miss rate} \times \text{Miss penalty} \\ &= \text{IC} \times (1 + 0.5) \times 0.01 \times 50 \\ &= \text{IC} \times 0.75\end{aligned}$$

where the middle term (1 + 0.5) represents one instruction access and 0.5 data accesses per instruction. The total performance is thus

$$\begin{aligned}\text{CPU execution time}_{\text{cache}} &= (\text{IC} \times 1.0 + \text{IC} \times 0.75) \times \text{Clock cycle} \\ &= 1.75 \times \text{IC} \times \text{Clock cycle}\end{aligned}$$

The performance ratio is the inverse of the execution times:

$$\begin{aligned}\frac{\text{CPU execution time}_{\text{cache}}}{\text{CPU execution time}} &= \frac{1.75 \times \text{IC} \times \text{Clock cycle}}{1.0 \times \text{IC} \times \text{Clock cycle}} \\ &= 1.75\end{aligned}$$

The computer with no cache misses is 1.75 times faster.

Bellek Teknolojisi ve Optimizasyonlar

- Performans metrikleri
  - Gecikme önbellek problemidir
  - Bant genişliği, çok işlemci ve G/Ç'yi ilgilendiren bir konudur
  - Erişim süresi
    - Okuma isteği ile istenen kelimenin gelmesi arasındaki süre
  - Çevrim süresi
    - Belleğe yapılan istekler arasındaki minimum süre
- SRAM belleğinin gecikme süresi düşüktür, önbellek için kullanılır
- Yüksek bant genişliği için DRAM çiplerini birden fazla blok halinde düzenleme, ana bellek için kullanılır

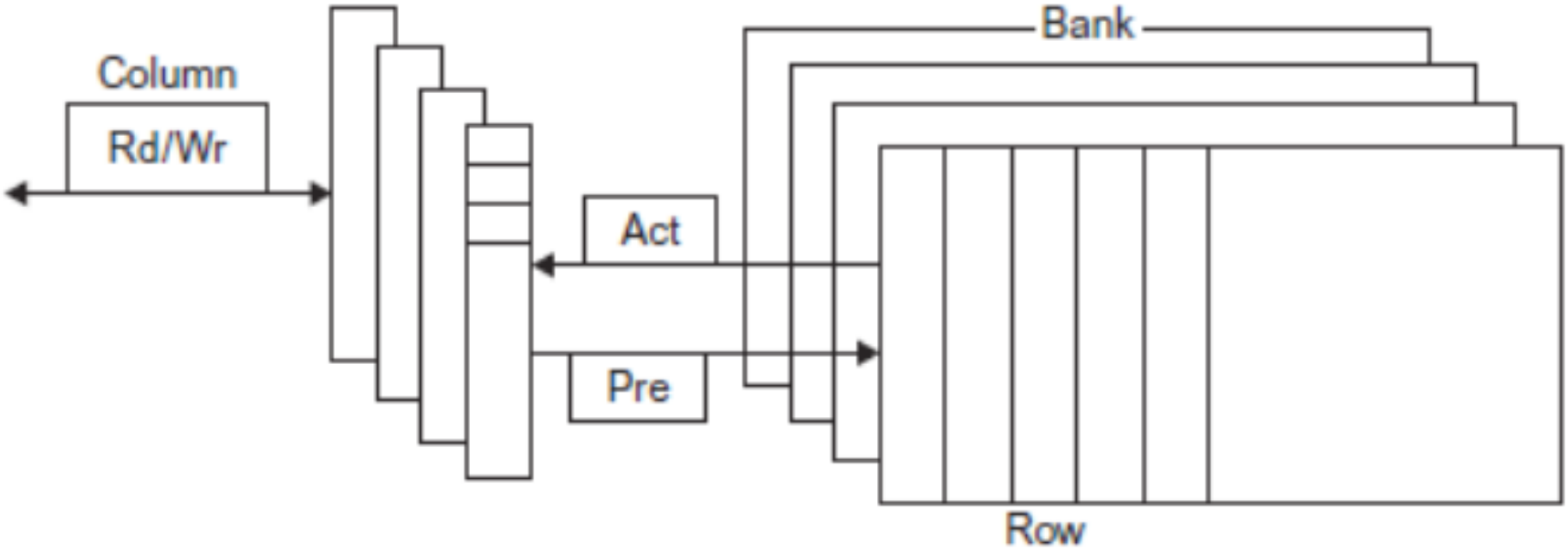
Bellek Teknolojisi ve Optimizasyonlar

- Performans metrikleri
  - Gecikme önbellek problemidir
  - Bant genişliği, çok işlemci ve G/Ç'yi ilgilendiren bir konudur
  - Erişim süresi
    - Okuma isteği ile istenen kelimenin gelmesi arasındaki süre
  - Çevrim süresi
    - Belleğe yapılan istekler arasındaki minimum süre
- SRAM belleğinin gecikme süresi düşüktür, önbellek için kullanılır
- Yüksek bant genişliği için DRAM çiplerini birden fazla blok halinde düzenleme, ana bellek için kullanılır

Bellek Teknolojisi

- SRAM
  - Biti tutmak için düşük güç gerektirir
  - 6 transistör/bit gerektirir
- DRAM
  - Okunduktan sonra yeniden yazılmalıdır.
  - Ayrıca periyodik olarak yenilenmelidir
    - Her ~ 8 ms'de bir (kabaca zamanın %5'i)
    - Her satır aynı anda yenilenebilir
  - Bir transistör/bit
  - Adres satırları çoğullanır:
    - Adresin üst yarısı: satır erişim strobu (RAS)
    - Adresin alt yarısı: sütun erişim strobu (CAS)

DRAM'ın İç Organizasyonu



Bellek Teknolojisi

- Amdahl:
  - Bellek kapasitesi, işlemci hızıyla doğrusal olarak büyümelidir
  - Ne yazık ki, bellek kapasitesi ve hızı işlemcilere ayak uyduramadı

- Bazı optimizasyonlar:
  - Aynı satıra birden çok erişim
  - Senkron DRAM
    - DRAM arayüzüne saat eklendi
    - Önce kritik kelime ile seri çekim modu
  - Daha geniş arayüzler
  - Çift veri hızı (DDR)
  - Her DRAM cihazında birden fazla bank

Bellek Optimizasyonları

			Best case access time (no precharge)			Precharge needed
Production year	Chip size	DRAM type	RAS time (ns)	CAS time (ns)	Total (ns)	Total (ns)
2000	256M bit	DDR1	21	21	42	63
2002	512M bit	DDR1	15	15	30	45
2004	1G bit	DDR2	15	15	30	45
2006	2G bit	DDR2	10	10	20	30
2010	4G bit	DDR3	13	13	26	39
2016	8G bit	DDR4	13	13	26	39

Bellek Optimizasyonları

Standard	I/O clock rate	M transfers/s	DRAM name	MiB/s/DIMM	DIMM name
DDR1	133	266	DDR266	2128	PC2100
DDR1	150	300	DDR300	2400	PC2400
DDR1	200	400	DDR400	3200	PC3200
DDR2	266	533	DDR2-533	4264	PC4300
DDR2	333	667	DDR2-667	5336	PC5300
DDR2	400	800	DDR2-800	6400	PC6400
DDR3	533	1066	DDR3-1066	8528	PC8500
DDR3	666	1333	DDR3-1333	10,664	PC10700
DDR3	800	1600	DDR3-1600	12,800	PC12800
DDR4	1333	2666	DDR4-2666	21,300	PC21300

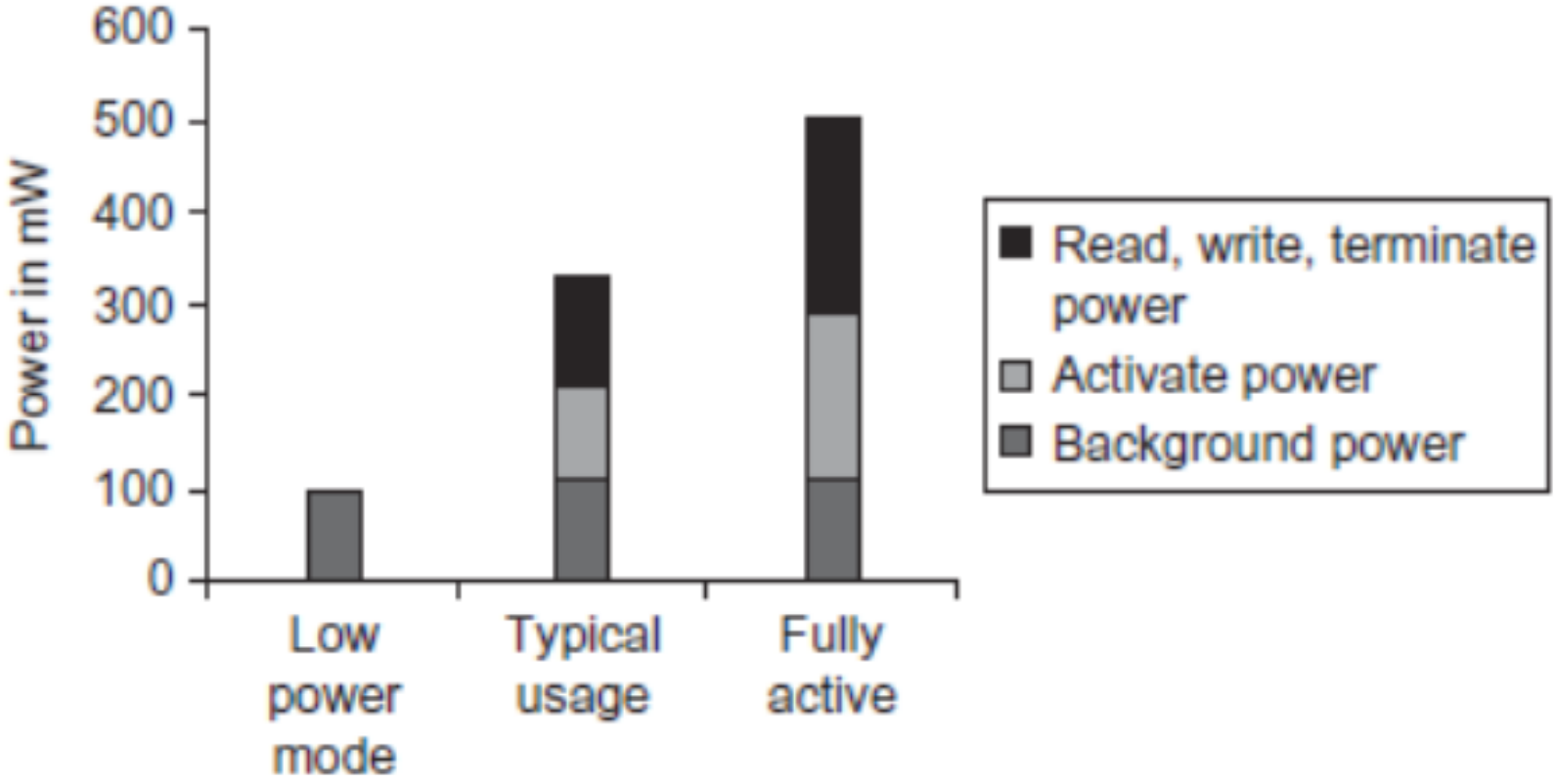
Bellek Optimizasyonları

- DDR:
  - DDR2
    - Daha düşük güç (2,5 V -> 1,8 V)
    - Daha yüksek saat hızları (266 MHz, 333 MHz, 400 MHz)
  - DDR3
    - 1,5 V
    - 800 MHz
  - DDR4
    - 1-1.2V
    - 1333 MHz
- GDDR5, DDR3 tabanlı grafik belleğidir

Bellek Optimizasyonları

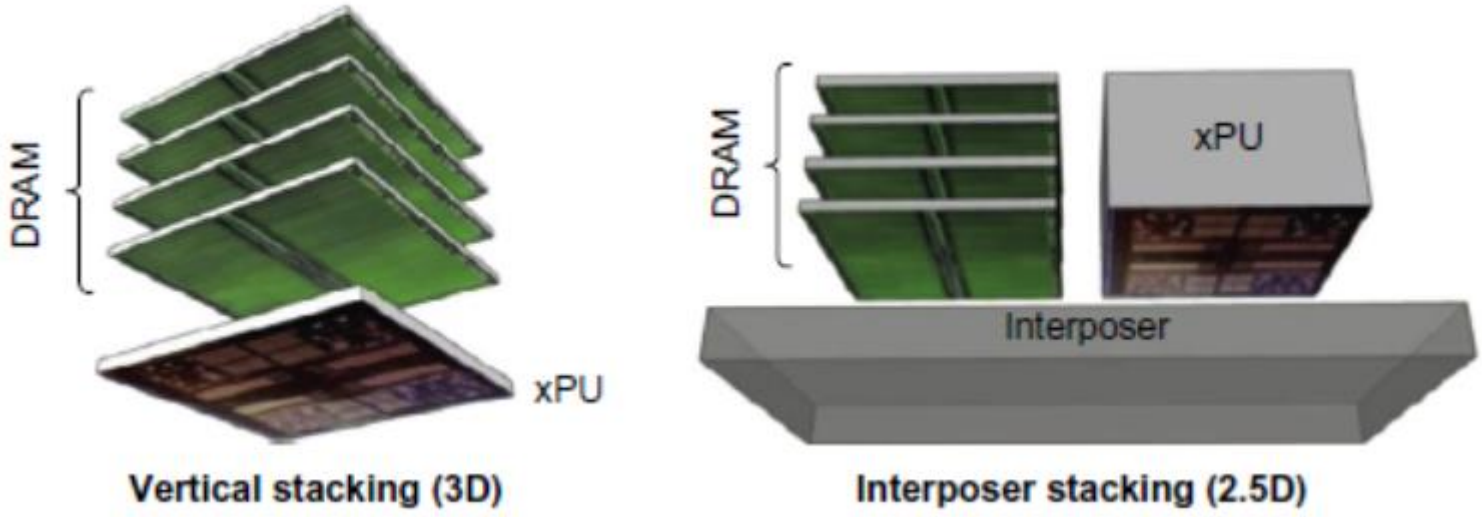
- SDRAM'lerde gücün azaltılması:
  - Düşük voltaj
  - Düşük güç modu (saati yok sayar, yenilemeye devam eder)
- Grafik belleği:
  - DDR3'e karşı DRAM başına 2-5 X bant genişliği
    - Daha geniş arayüzler (16'ya karşı 32 bit)
    - Daha yüksek saat hızı
      - Soket yerine lehimleme ile bağlanan DIMM modülleri

Bellek Güç Tüketimi



Yığılmış/Gömülü DRAM'ler

- İşlemci ile aynı pakette yığılmış DRAM'ler
  - Yüksek Bant Genişliği Belleği (HBM)



Flash Bellek

- EEPROM Türü
- Türler: NAND (daha yoğun) ve NOR (daha hızlı)
- NAND Flaş:
  - Okumalar sıralıdır, tüm sayfayı okur (.5 ila 4 KiB)
  - İlk bayt için 25 us, sonraki baytlar için 40 MiB/s
  - SDRAM: İlk bayt için 40 ns, sonraki baytlar için 4,8 GB/sn
  - 2 KiB aktarımı: SDRAM için 75 uS vs 500 ns, 150X daha yavaş
  - Manyetik diskten 300 ila 500X daha hızlı

NAND Flash Bellek

- Üzerine yazılmadan önce (bloklar halinde) silinmelidir
- Uçucu değildir, sıfır güç
- Sınırlı sayıda yazma döngüsü (~100.000)
- 2$/GiB, SDRAM için 20-40$/GiB ve manyetik disk için 0.09 GiB ile karşılaştırıldığında

- Faz Değişimi/Memrister Belleği
  - Muhtemelen yazma performansında 10 kat ve okuma performansında 2 kat iyileştirme

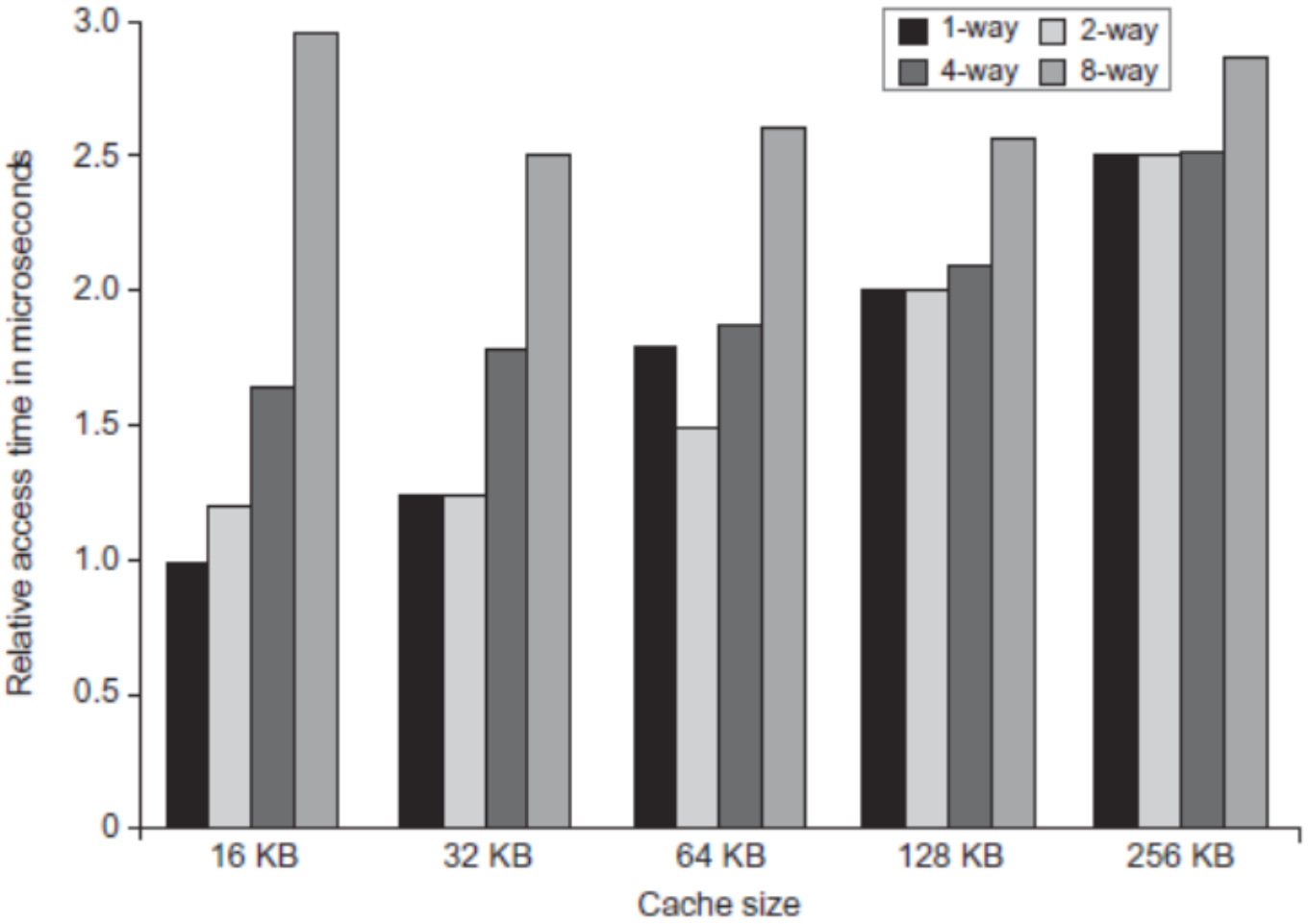
Bellek Güvenilirliği

- Hafıza kozmik ışınlara karşı hassastır
- *Yazılım hataları* : dinamik hatalar
  - Hata düzeltme kodları (ECC) tarafından tespit edilir ve düzeltilir
- *Zor hatalar* : kalıcı hatalar
  - Arızalı satırları değiştirmek için yedek satırları kullanır
- Chipkill : RAID benzeri bir hata kurtarma tekniği

Gelişmiş Optimizasyonlar

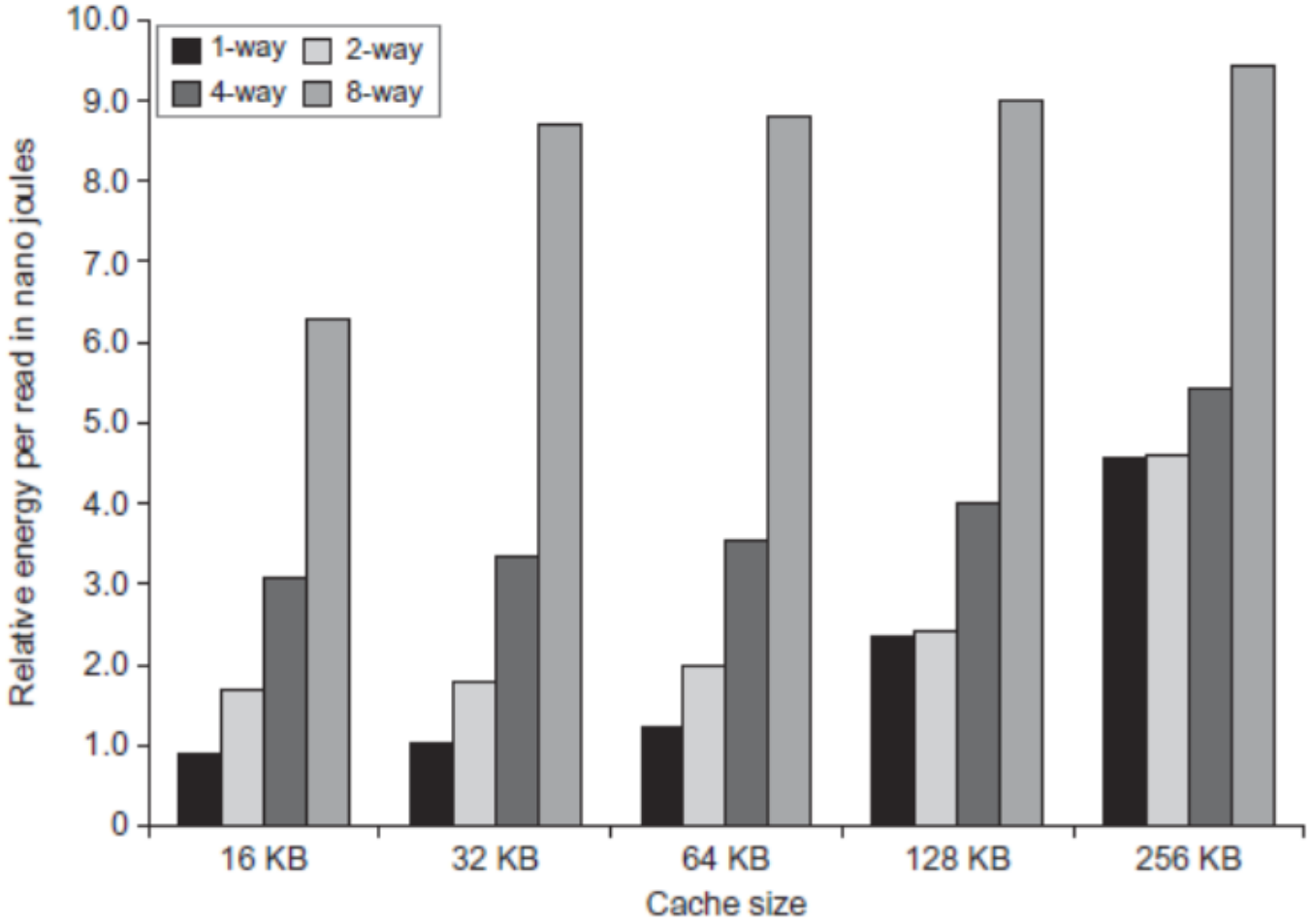
- İsabet Sayısını düşürme
  - Küçük ve basit birinci seviye önbellekler
  - Yol tahmini
- Bant genişliğini artırma
  - Sıralı önbellekler, çok banklı önbellekler, bloke olmayan önbellekler
- Iskalama cezasını azaltma
  - Önce kritik kelime, yazma arabelleklerini birleştirme
- Iskalama oranını azaltma
  - Derleyici optimizasyonları
- Paralelleştirme yoluyla ıskalama cezasını veya ıskalama oranını azaltma
  - Donanım veya derleyici prefetch

L1 Boyutu ve İlişkilendirme



Erişim süresi, boyut ve ilişkilendirilebilirlik karşılaştırması

L1 Boyutu ve İlişkilendirme



Okuma başına enerji ve boyut ve ilişkilendirme

Yol Tahmini

- İsabet süresini iyileştirmek için pre-set mux yolu tahmini
  - Iskalama tahmini daha uzun isabet sayısı sağlar
  - Tahmin doğruluğu
    - > iki yollu için %90
    - > dört yollu için %80
    - I-cache, D-cache'den daha iyi doğruluğa sahiptir
  - İlk olarak 90'ların ortalarında MIPS R10000'de kullanıldı
  - ARM Cortex-A8'de kullanılır
- Bloğu da tahmin etmek için genişletilir
  - “Yol seçimi”
  - Iskalama tahmin cezasını artırır

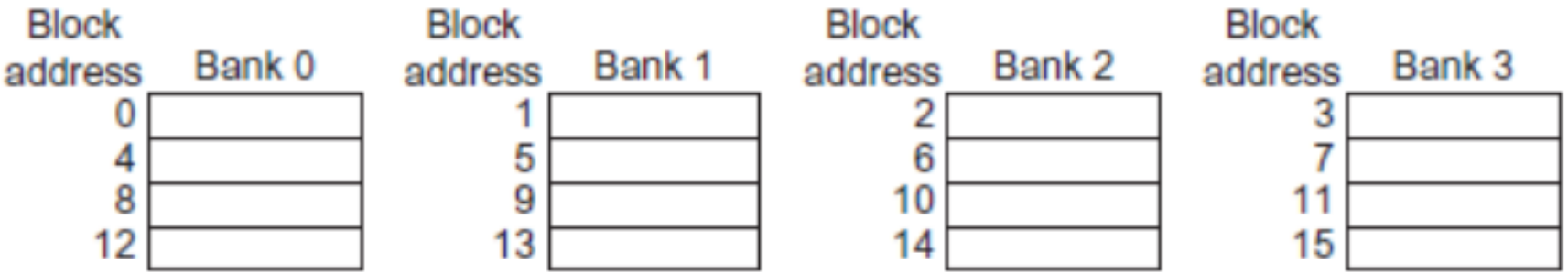
Ardışık(pipeline) Önbellekler

- Bant genişliğini iyileştirmek için ardışık düzen önbelleği erişimi
  - Örnekler:
    - Pentium: 1 çevrim
    - Pentium Pro – Pentium III: 2 çevrim
    - Pentium 4 – Core i7: 4 çevrim
- Dallanma ıskalama tahmin cezasını artırır
- İlişkiselliği artırmayı daha kolay yapar

Çok Banklı Önbellekler

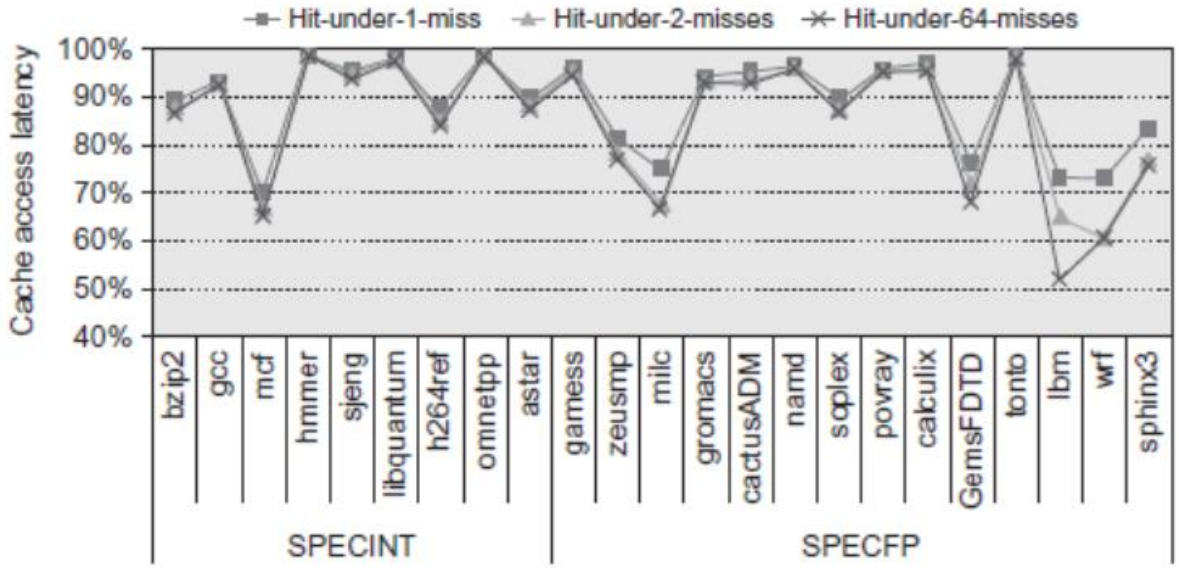
- Eşzamanlı erişimi desteklemek için önbellek bağımsız banklar olarak düzenlenir
  - ARM Cortex-A8, L2 için 1-4 bankı destekler
  - Intel i7, L1 için 4 bank ve L2 için 8 bankı destekler

- Banklar blok adresine göre dağıtılır



Bloke olmayan Önbellekler

- Önceki ıskalar tamamlanmadan isabetlere izin verir
  - "Iskanın altında vur"
  - "Birden çok ıskalamanın altında vur"
- L2 desteklemeli
- Genel olarak, işlemciler L1 ıskalama cezasını gizleyebilir ancak L2 ıskalama cezasını gizleyemez



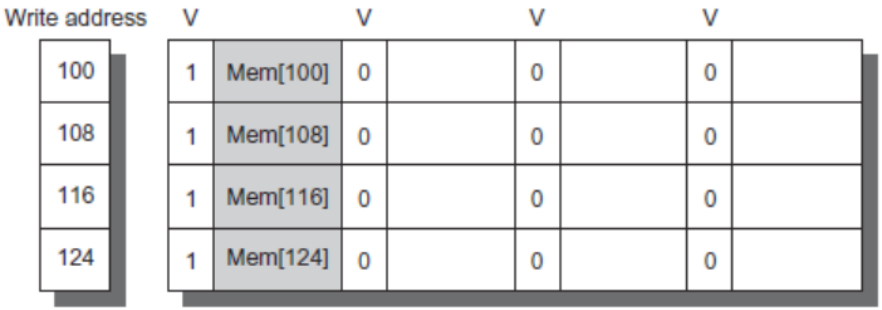
Önce Kritik Kelime, Erken Yeniden Başlatma

- Önce kritik kelime
  - Önce bellekten ıskalanmış kelime istenir
  - Gelir gelmez işlemciye gönderilir
- Erken yeniden başlatma
  - Kelimeler normal sırada istenir
  - Gelir gelmez ıskalanan iş işlemciye gönderilir

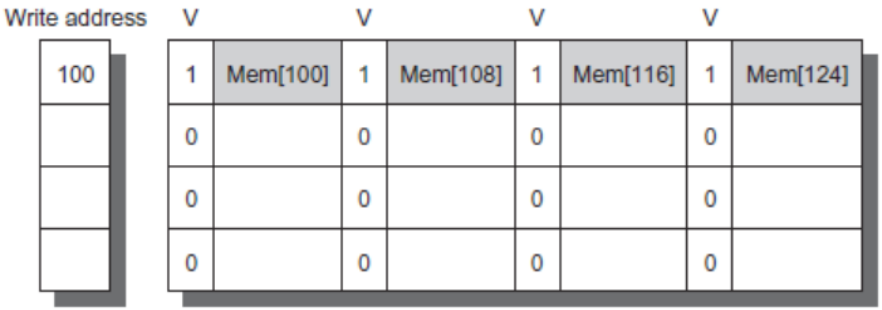
- Bu stratejilerin etkinliği, blok boyutuna ve bloğun henüz getirilmemiş kısmına başka bir erişimin olma olasılığına bağlıdır.

Yazma Tamponunu Birleştirme

- Yazma tamponunda zaten bekleyen bir bloğa depolarken, yazma tamponu güncellenir
- Yazma tamponunun dolması nedeniyle duraklamaları azaltır
- G/Ç adreslerine uygulanmaz



Yazma tamponu yok



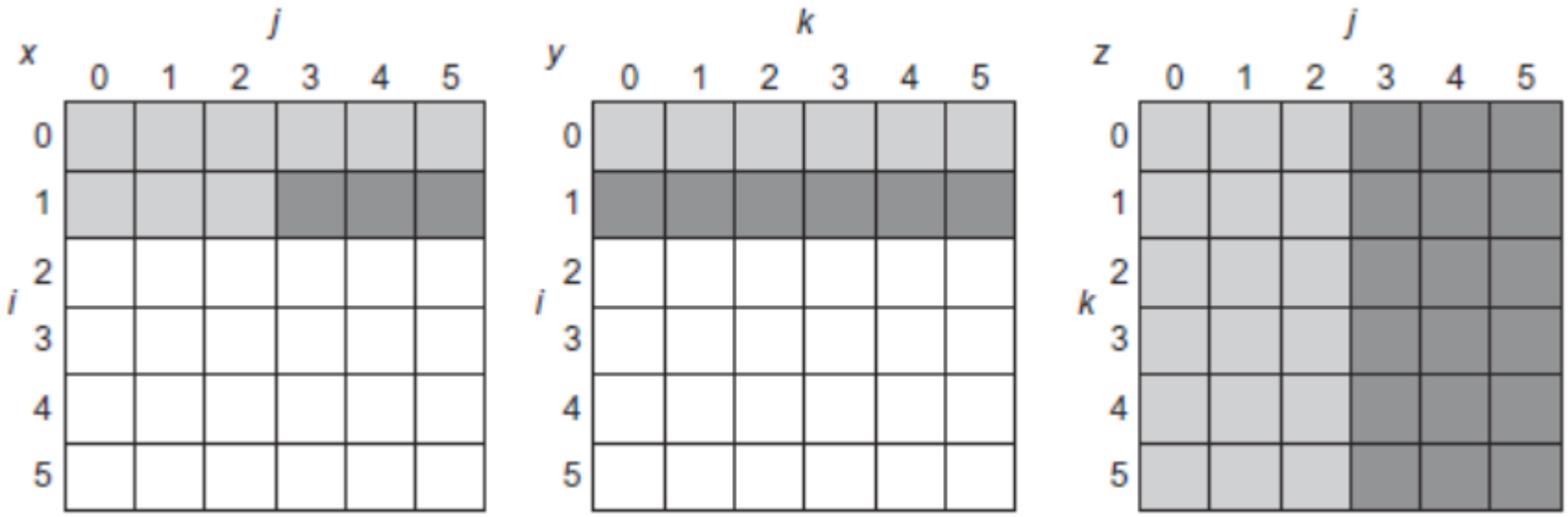
tampona yazma

Derleyici Optimizasyonları

- Döngü Değişimi
  - Belleğe sırayla erişmek için iç içe döngüleri değiştirir
- Bloklama
  - Tüm satırlara veya sütunlara erişmek yerine matrisleri bloklara ayırır
  - Daha fazla bellek erişimi gerektirir, ancak erişimlerin yerelliğini iyileştirir

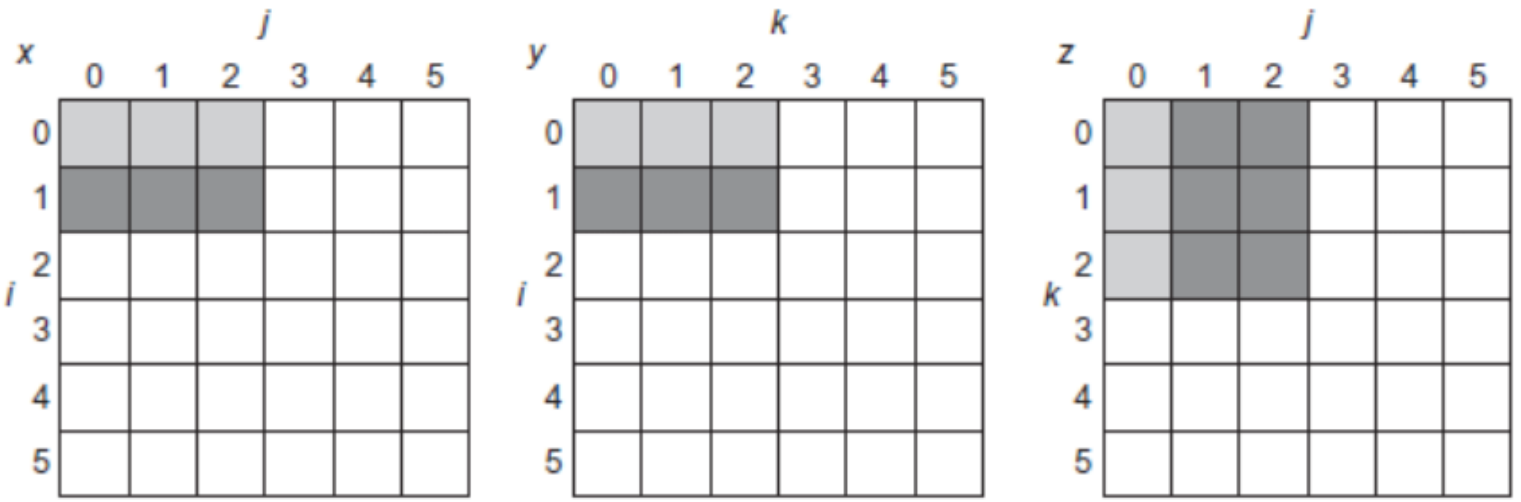
Bloklama

```
(i = 0; ben < N; ben = ben + 1)
için
(j = 0; j < N; j = j + 1) için
{
r = 0;
(k = 0; k < N; k = k + 1) için
r = r + y[i][k]*z[k][j];
x[i][j] = r;
}
```



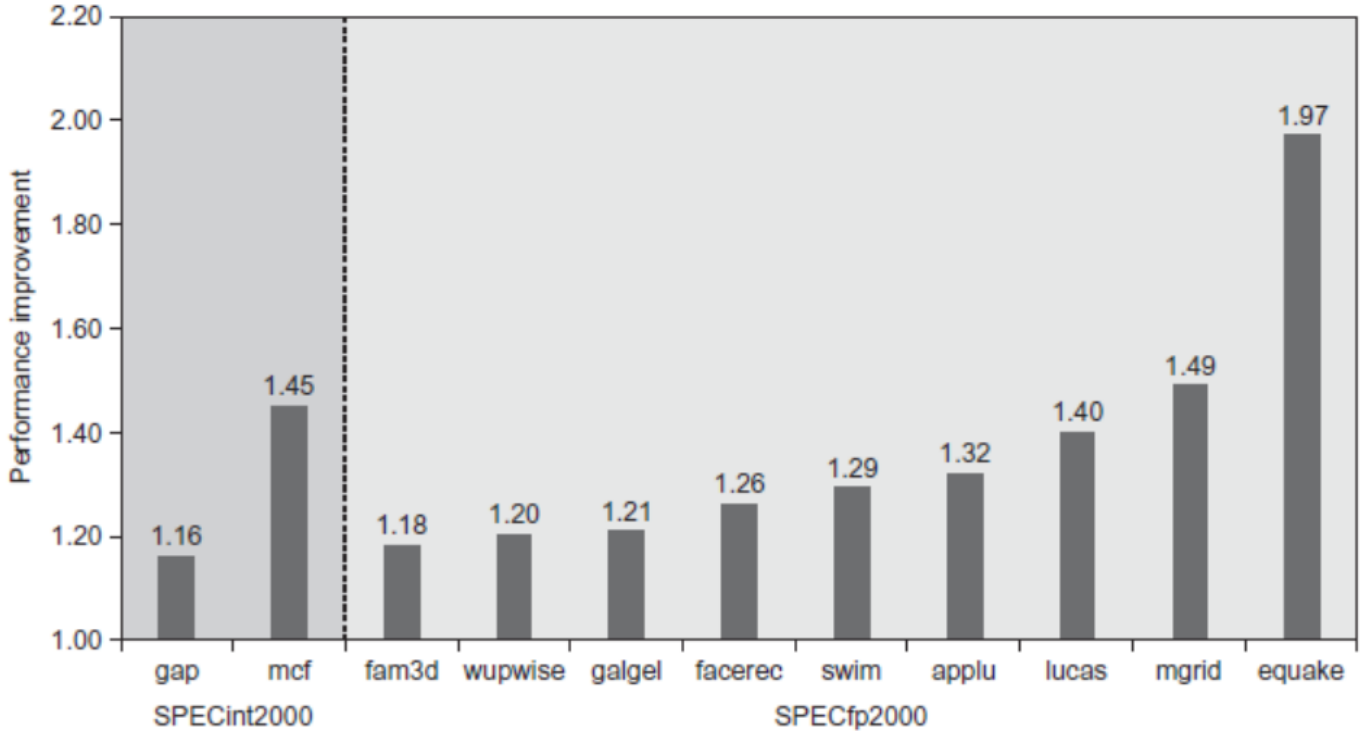
Bloklama

```
için (jj = 0; jj < N; jj = jj + B)
için (kk = 0; kk < N; kk = kk + B)
(i = 0; ben < N; ben = ben + 1) için
(j = jj; j < min(jj + B,N); j = j + 1) için
{
r = 0;
(k = kk; k < min(kk + B,N); k = k + 1) için
r = r + y[i][k]*z[k][j];
x[i][j] = x[i][j] + r;
};
```



Donanım Önyükleme (prefetching)

- Iskalama üzerine iki blok getirilir (sonraki sıralı blok)



Pentium 4 Ön yükleme

Derleyici Önyükleme

- Veri ihtiyacından önce önyükleme talimatları girilir
- Hatalı değil: önyükleme, istisnalara neden olmaz

- Kaydedici önyükleme
  - Verileri kaydedicilere yükler
- Önbellek önyükleme
  - Verileri önbelleğe yükler

- Döngü ve ardışık yazılım oluşturma ile birleştirilir

Hiyerarşiyi Genişletmek için HBM

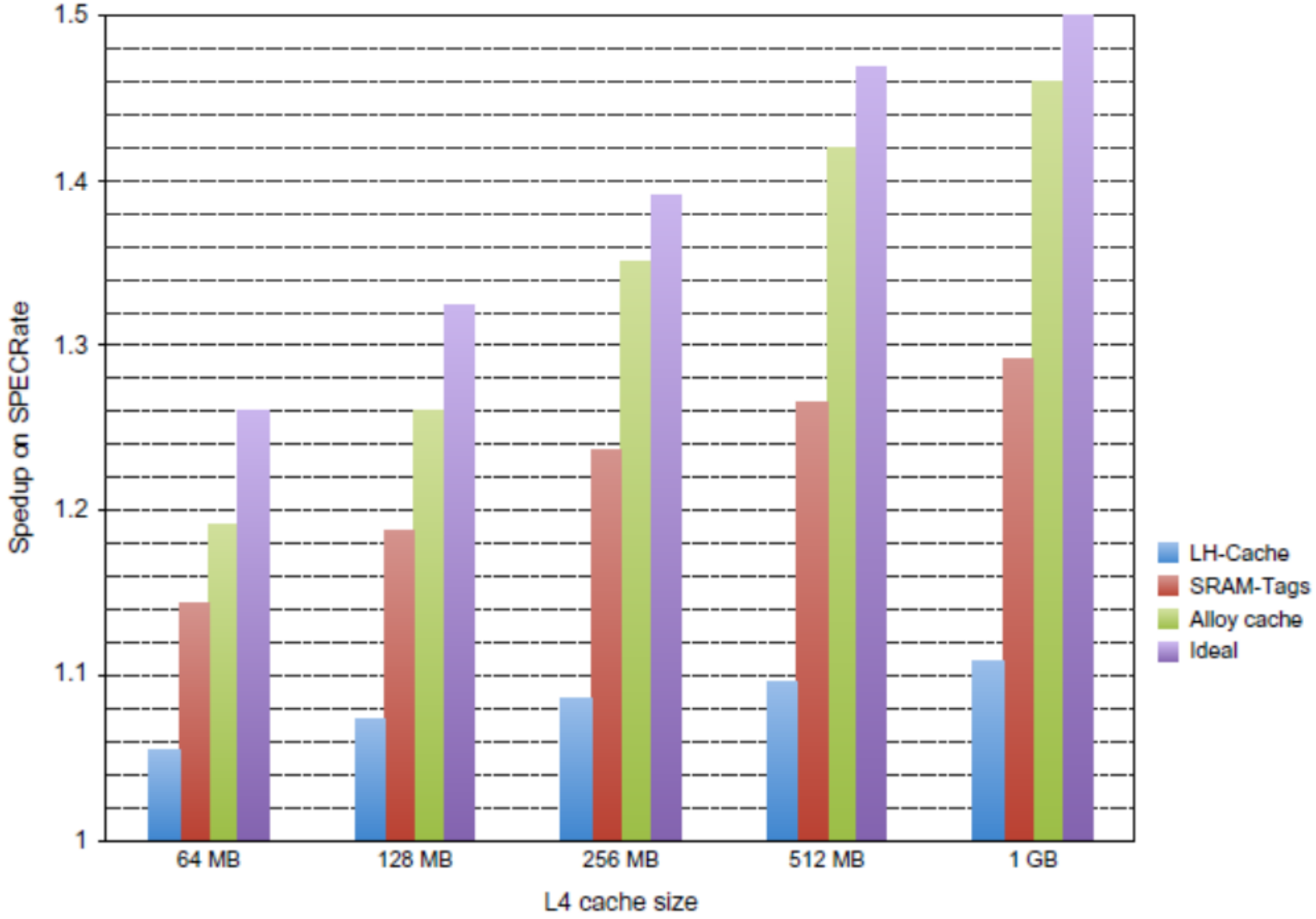
- 128 MiB - 1 GiB
- Daha küçük bloklar, önemli miktarda etiket depolaması gerektirir
- Daha büyük bloklar potansiyel olarak verimsizdir

- Bir yaklaşım (LH):
  - Her SDRAM satırı bir blok indeksidir
  - Her satırda bir dizi etiket ve 29 veri segmenti bulunur.
  - 29 küme ilişkisel
  - İsacet bir CAS gerektirir

Hiyerarşiyi Genişletmek için HBM

- Başka bir yaklaşım (Alaşım önbelleği):
  - Etiket ve veriler birlikte
  - Doğrudan eşleme kullanılır
- Her iki düzende, ıskalamalar için iki DRAM erişimi gerektirir
  - İki çözüm:
    - Blokları takip etmek için harita kullanılır
    - Olası ıskaları tahmin eder

Hiyerarşiyi Genişletmek için HBM



Özet

Technique	Hit time	Band-width	Miss penalty	Miss rate	Power consumption	Hardware cost/ complexity	Comment
Small and simple caches	+			–	+	0	Trivial; widely used
Way-predicting caches	+				+	1	Used in Pentium 4
Pipelined & banked caches	–	+				1	Widely used
Nonblocking caches		+	+			3	Widely used
Critical word first and early restart			+			2	Widely used
Merging write buffer			+			1	Widely used with write through
Compiler techniques to reduce cache misses				+		0	Software is a challenge, but many compilers handle common linear algebra calculations
Hardware prefetching of instructions and data			+	+	–	2 instr., 3 data	Most provide prefetch instructions; modern high-end processors also automatically prefetch in hardware
Compiler-controlled prefetching			+	+		3	Needs nonblocking cache; possible instruction overhead; in many CPUs
HBM as additional level of cache		+/–	–	+	+	3	Depends on new packaging technology. Effects depend heavily on hit rate improvements

Sanal Bellek ve Sanal Makineler

- Sanal bellek ile koruma
  - Prosesleri kendi bellek alanlarında tutar
- Mimari Rolü
  - Kullanıcı modu ve süpervizör modu sağlar
  - CPU durumunun belirli yönlerini korur
  - Kullanıcı modu ve süpervizör modu arasında geçiş yapmak için mekanizmalar sağlar
  - Bellek erişimlerini sınırlamak için mekanizmalar sağlar
  - Adresleri dönüştürmek için TLB sağlar

Sanal makineler

- İzolasyon ve güvenliği destekler
- Bir bilgisayarı birçok kullanıcı arasında paylaşma
- İşlemcileri ek yükü kabul edilebilir hale getirir

- Farklı ISA'ların ve işletim sistemlerinin kullanıcı programlarına sunulmasına izin verir
  - “Sistem Sanal Makineleri”
  - SVM yazılımına "sanal makine monitörü" veya "hipervizör" denir.
  - Monitörün altında çalışan bireysel sanal makinelere "misafir VM'ler" denir.

VMM gereksinimleri

- Konuk yazılımı şunları yapmalıdır:
  - Yerel donanım üzerinde çalışıyormuş gibi davranır
  - Gerçek sistem kaynaklarının tahsisini değiştirememek
- VMM, konukları "bağlam değiştirme" yapabilmelidir
- Donanım aşağıdakilere izin vermelidir:
  - Sistem ve kullanım işlemci modları
  - Sistem kaynaklarını tahsis etmek için ayrıcalıklı talimat alt kümesi

VM'lerin Sanal Belleğe Etkisi

- Her konuk işletim sistemi kendi sayfa tablolarını tutar
  - VMM, fiziksel ve sanal bellek arasına “gerçek bellek” adı verilen bir bellek düzeyi ekler.
  - VMM, konuk sanal adreslerini fiziksel adreslerle eşleyen gölge sayfa tablosunu korur
    - VMM’in konuğun kendi sayfa tablosunda yaptığı değişiklikleri algılamasını gerektirir
    - Sayfa tablosu işaretçisine erişim ayrıcalıklı bir işlemse doğal olarak gerçekleşir

Sanallaştırma için ISA'yı Genişletme

- Hedefler:
  - TLB'yi temizlemekten kaçınılır
  - Gölge sayfa tabloları yerine iç içe sayfa tablolarını kullanır
  - Cihazların verileri taşımak için DMA kullanmasına izin verilir
  - Konuk işletim sistemlerinin cihaz kesintilerini işlemesine izin verilir
  - Güvenlik için: programların şifrelenmiş kod ve veri bölümlerini yönetmesine izin verilir

Yanılgılar ve Tuzaklar

- Bir programın diğerinden önbellek performansını tahmin etme
- Bellek hiyerarşisinin doğru performans ölçümlerini almak için yeterli talimatı simüle etmek
- Önbellek tabanlı bir sistemde yüksek bellek bant genişliği sağlamamak