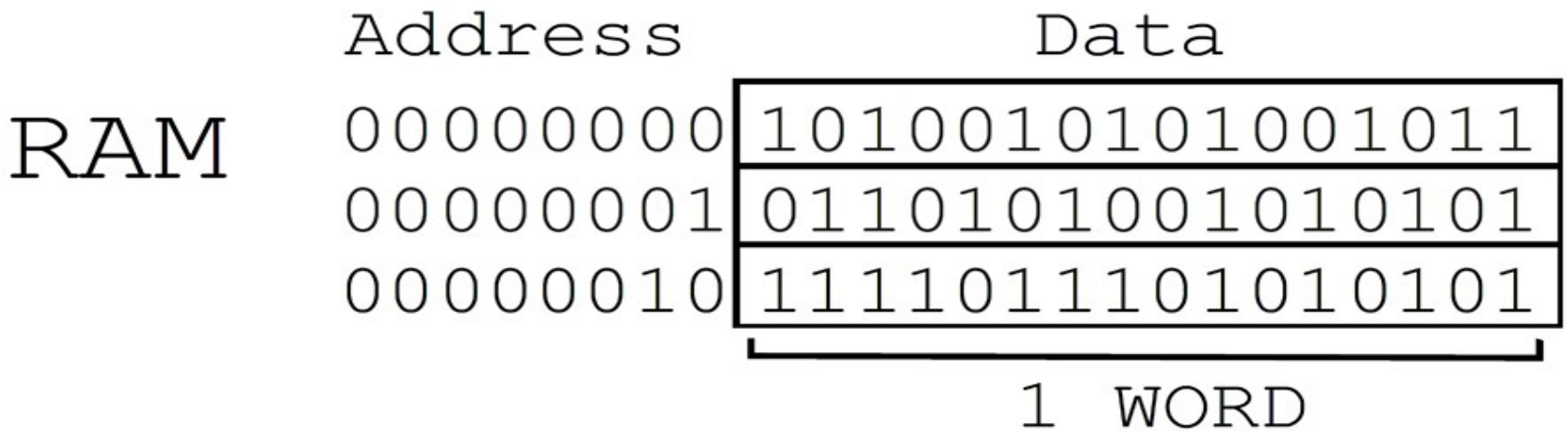
Cache Memory & Mapping

Organisasi Komputer

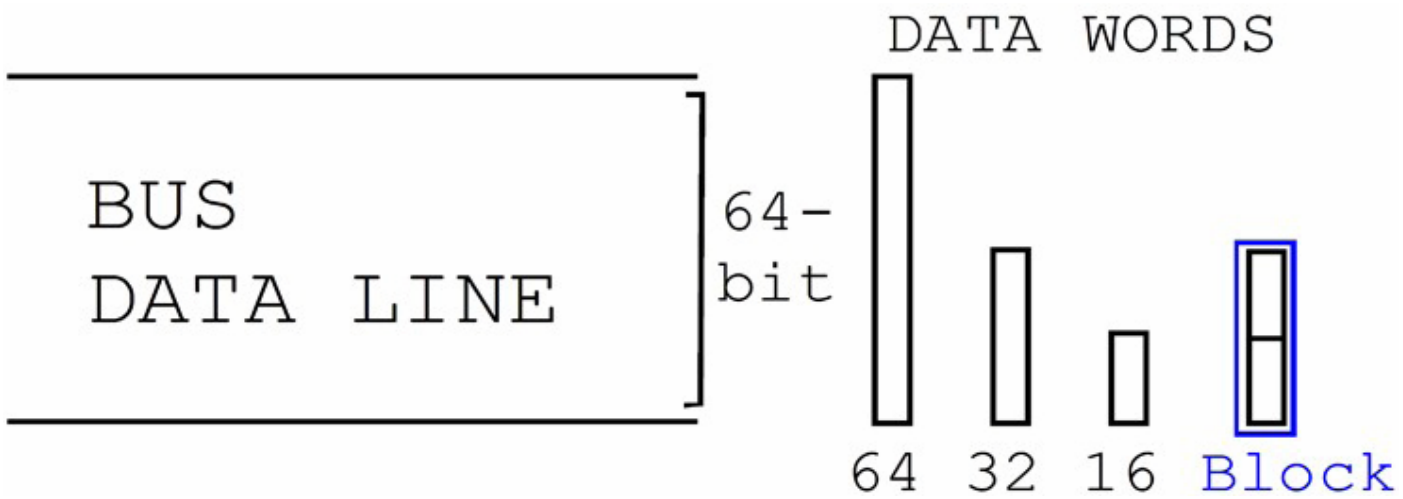
Satuan Penyimpanan Data

- Data disimpan dalam bentuk bit-bit bilangan biner (1 bit = 1 digit biner)
- Satu bagian penampung data di memory disebut word
- Satu word berapa bit? Tergantung masing-masing memory
-



Satuan transfer???

- Yang jelas, karena data di memory ditransfer melalui bus, maka ukuran word memorynya tidak akan lebih besar dari lebar busnya (data line).
- Namun ukuran word boleh lebih kecil dari lebar bus.
- Sekelompok word disebut dengan block

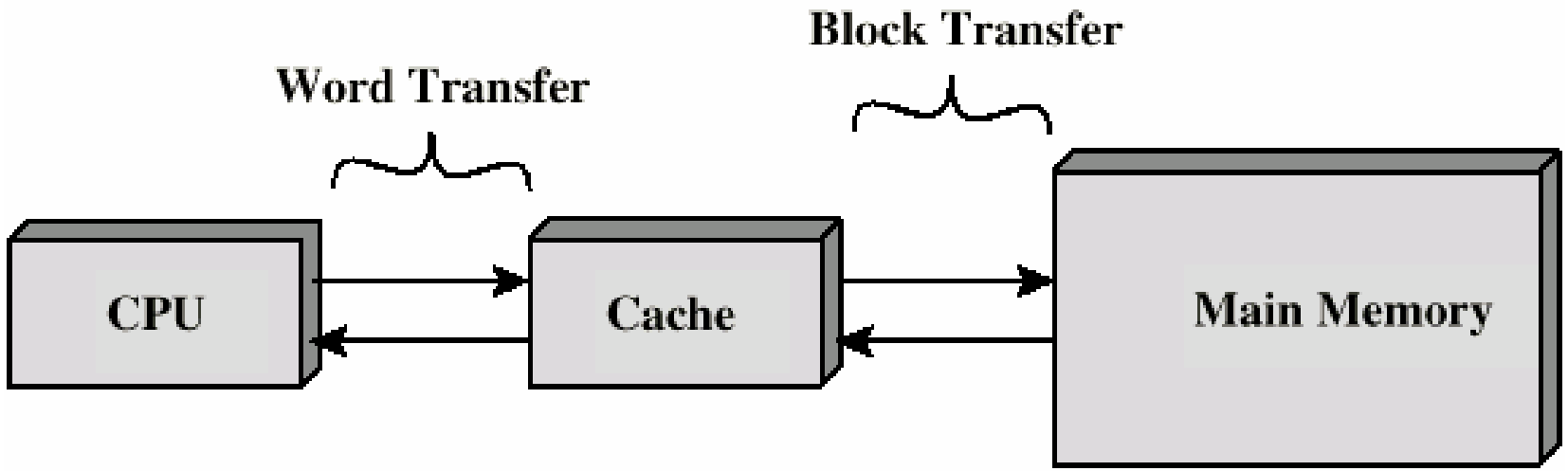


Cache

- Sebagian program mempunyai instruksi yang cenderung mengambil data dari alamat yang sama berkali-kali.
- Misal: menyimpan variabel, kemudian dilakukan loop.
- Kalau untuk data yang sama harus mencari lokasi alamat ke memory lagi, berarti buang-buang waktu
- Bagaimana seandainya data yang sering diakses, disimpan di dalam memory, tetapi harus static (punya jalur langsung ke CPU), agar lebih cepat → Cache

Cache

- Memori cepat dg kapasitas yg sedikit
- Terdiri dari slot-slot berukuran masing-masing satu word memory
- Terletak antara main memory dengan CPU
- Bisa saja diletakkan dalam chip CPU atau module tersendiri



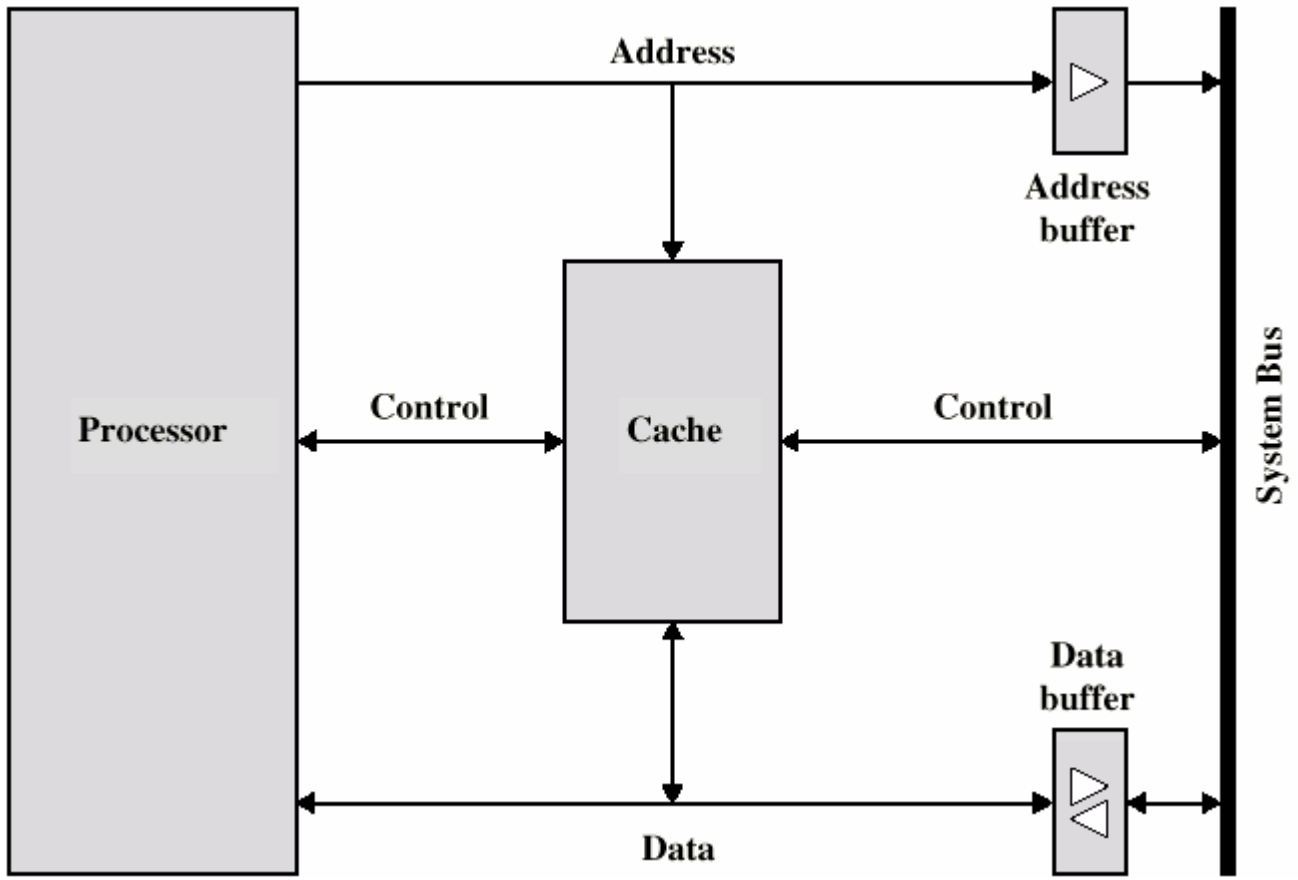
Operasi pada Cache

- Saat CPU meminta alamat sebuah lokasi memory, cache dicek untuk data tersebut
- Kalau ada, ambil dari cache (cepat)
- Kalau tidak ada, ambil data dari memory, simpan ke dalam slot cache, kemudian teruskan data dari cache ke CPU
- Pada saat menyimpan ke slot, cache memberikan tag dari blok memory mana data tersebut berasal, untuk referensi berikutnya.

Desain Cache – Size Does Matter

- Semakin besar cache, semakin bagus, karena akses data akan semakin cepat
- Tapi untuk membuat cache yang besar, biayanya akan sangat mahal
- Cache akan selalu berukuran kecil, dan hanya menyimpan data-data berfrekuensi akses tinggi saja.
- Akan selalu butuh metode MAPPING
- Akan selalu butuh WRITE POLICY

Organisasi Cache



Fungsi Pemetaan
MAPPING FUNCTION

- Metode yang digunakan untuk menemukan/menempatkan sebuah alamat memori dalam sebuah cache
- Menyalin sebuah blok dari memori utama ke dalam cache dan digunakan juga untuk menerima data dari cache
- Terdapat tiga macam MF:
  - Direct
  - Associative
  - Set Associative

Fungsi Mapping

- Ukuran Cache 64kByte
- Ukuran block 4 bytes
  - diperlukan 16k (2^{14}) alamat per alamat 4 bytes
  - Jumlah jalur alamat cache 14
- Main memory 16MBytes
- Jalur alamat perlu 24 bit
  - (2^{24}=16M)

Direct Mapping

- Satu word data dari main memory akan dimapping ke satu slot/baris cache
- Alamat main memory akan dibagi menjadi:
- w bit LSB mewakili nomor word dalam 1 block
- s bit MSB mewakili block
- MSB kemudian diambil r bit sebagai penanda baris cache, sisanya (s-r) sebagai tag (penanda)

Direct Mapping

- Setiap block dari memori utama hanya memetakan ke dalam satu baris cache. Jika suatu block ditemukan di cache, maka block tersebut selalu ditemukan pada tempat yang sama.
  - Nomor baris dihitung menggunakan rumus berikut:

i = j modulo m

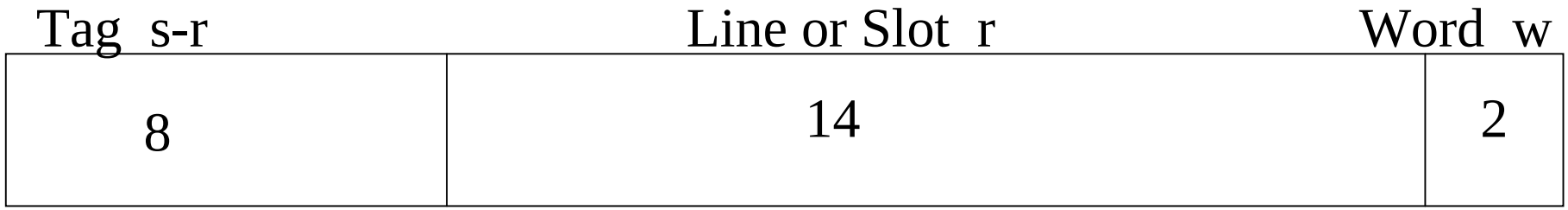
di mana

i = Nomor baris cache

j = Nomor block pada memori utama

m = Jumlah baris di cache

Struktur Alamat Direct Mapping

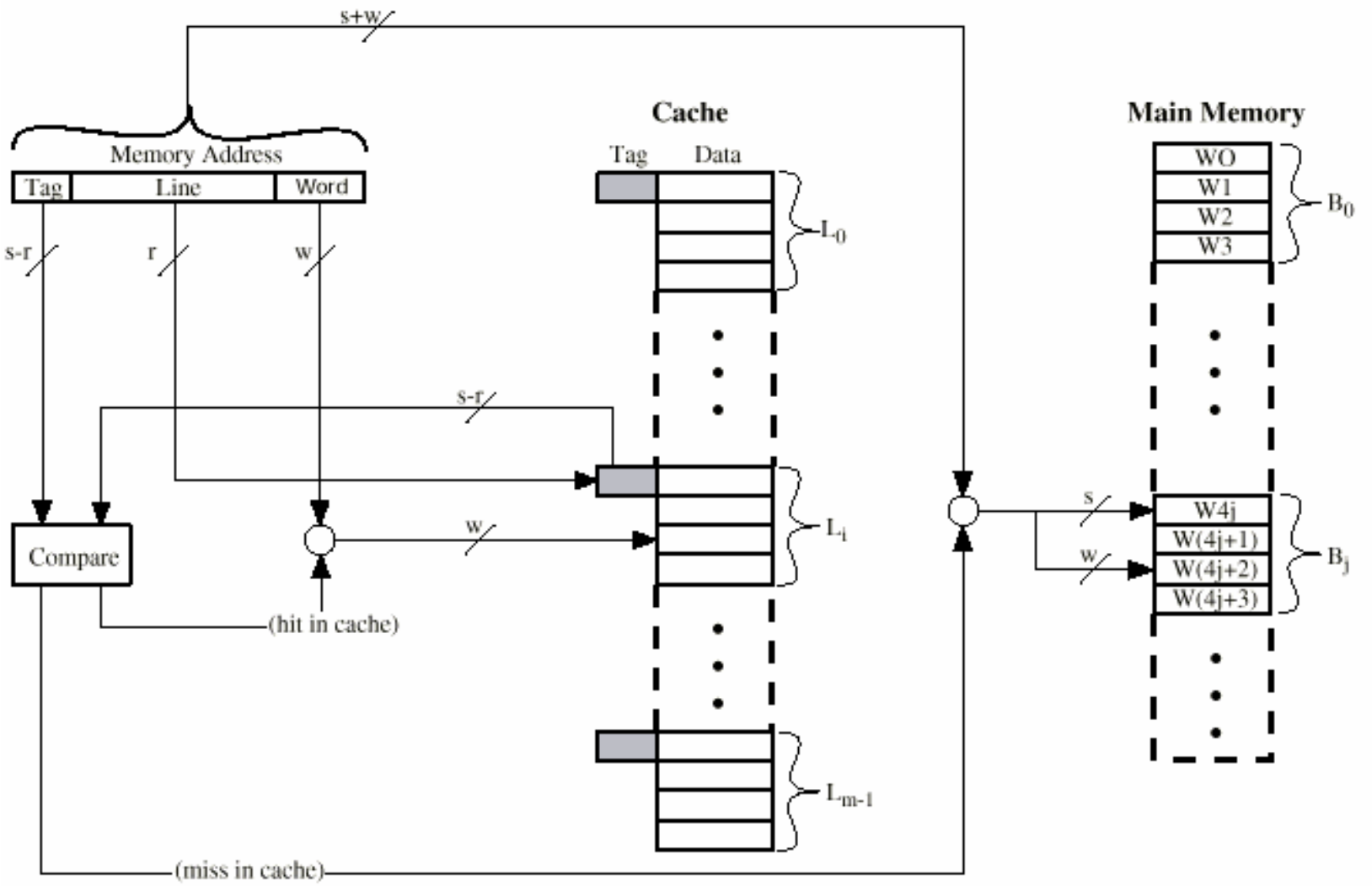


- 24 bit address
- 2 bit : word identifier (4 byte block)
- 22 bit: block identifier
  - 8 bit tag (=22-14)
  - 14 bit slot atau line
- 2 blocks pada line yg sama tidak boleh memiliki tag yg sama
- Cek isi cache dengan mencari line dan Tag

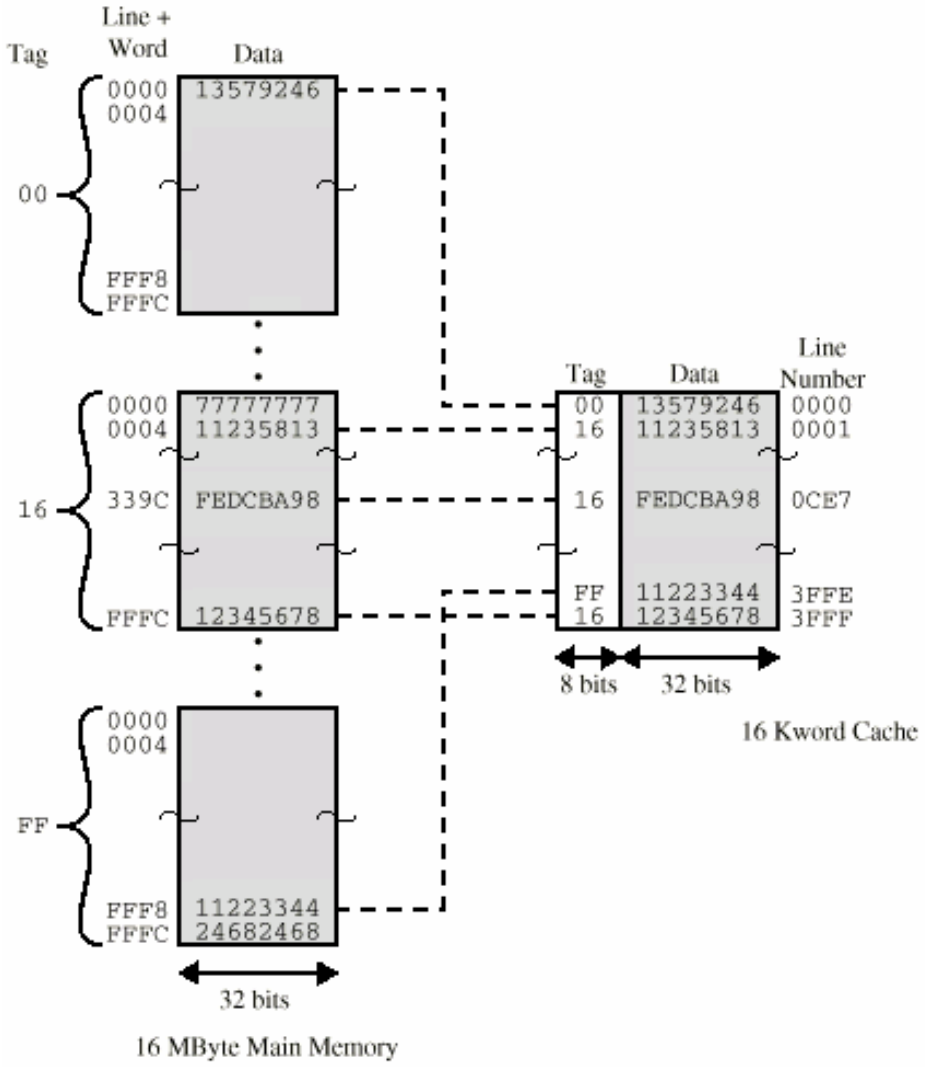
Table Cache Line pada Direct Mapping

Cache line	blocks main memori
0	0, m, 2m, 3m…2^s-m
1	1,m+1, 2m+1…2^s-m+1
m-1	m-1, 2m-1,3m-1…2^s-1

Organisasi Cache Direct Mapping



Contoh Direct Mapping



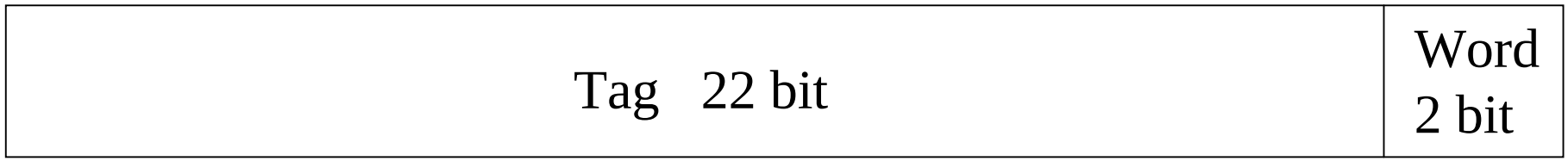
Keuntungan & Kerugian Direct Mapping

- Sederhana
- Tidak boros resource
- Alamat mapping tiap word ke cache sama
- Suatu blok memiliki lokasi yang tetap
  - Jika program mengakses 2 block yang di map ke line yang sama secara berulang-ulang, maka cache-miss sanagat tinggi

Associative Mapping

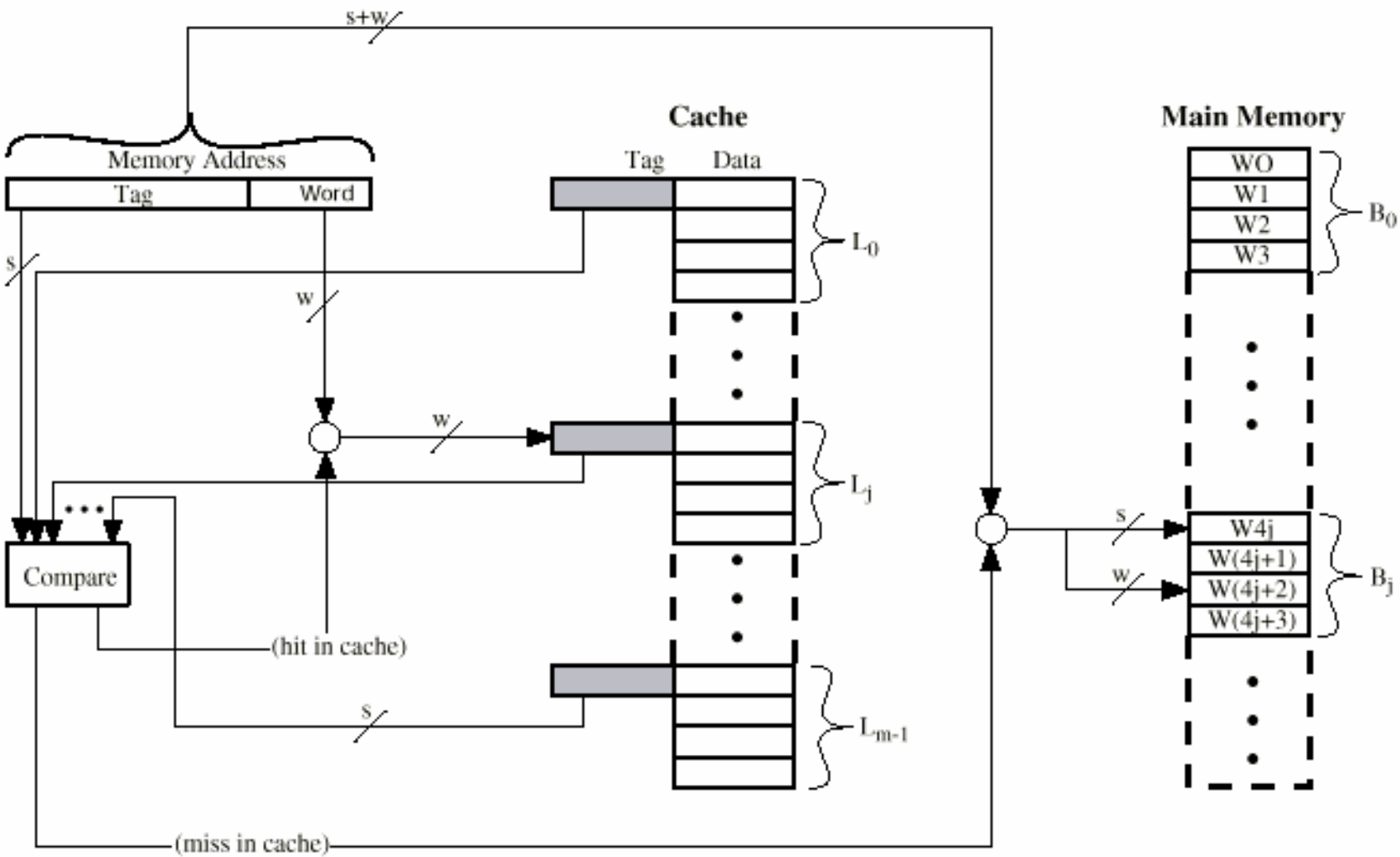
- Satu word data dari main memory akan dimapping ke satu slot/baris cache
- Alamat main memory akan dibagi menjadi:
- w bit LSB mewakili nomor word dalam 1 block
- s bit MSB sebagai Tag yang mewakili 1 block
- Tag disimpan sebagai penanda dari block mana di memory, data tsb berasal.

Struktur Address Associative Mapping

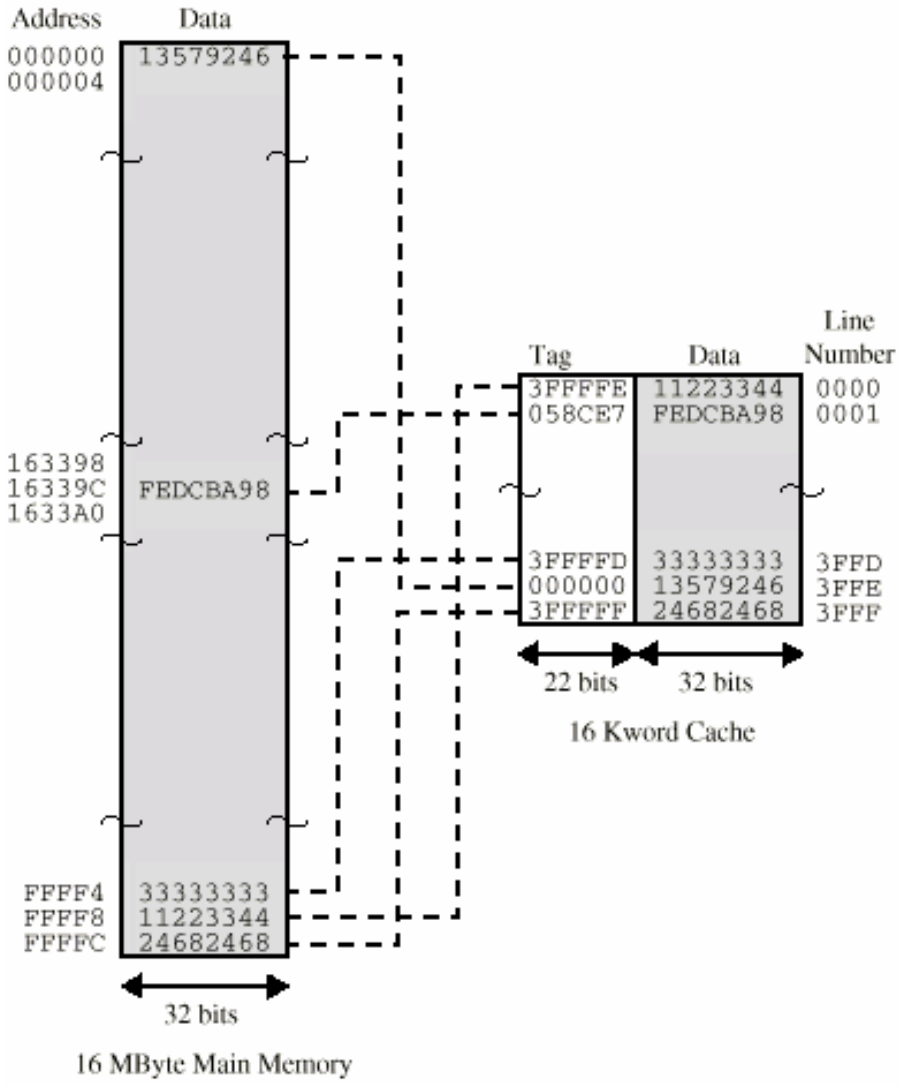


- 24 bit address
- 2 bit word identifier (4 word per block)
- 22 bit tag identifier
- Check contents of cache by finding TAG

Organisasi Cache Fully Associative



Contoh Associative Mapping



Keuntungan & Kerugian Associative Mapping

- Lebih banyak bit yang disimpan sebagai
- penanda
- Lebih boros resource karena harus mencari dari line paling atas dengan mencocokkan tiap-tiap tag

Set Associative Mapping

- Satu word data dari main memory akan di-mapping ke slot mana saja di cache dalam baris set yang sama
- Alamat main memory akan dibagi menjadi:
- w bit LSB mewakili nomor word dalam 1 block
- s bit MSB mewakili block
- MSB kemudian diambil r bit sebagai penanda baris set cache, sisanya (s-r) sebagai tag (penanda)

Struktur Address: Set Associative Mapping

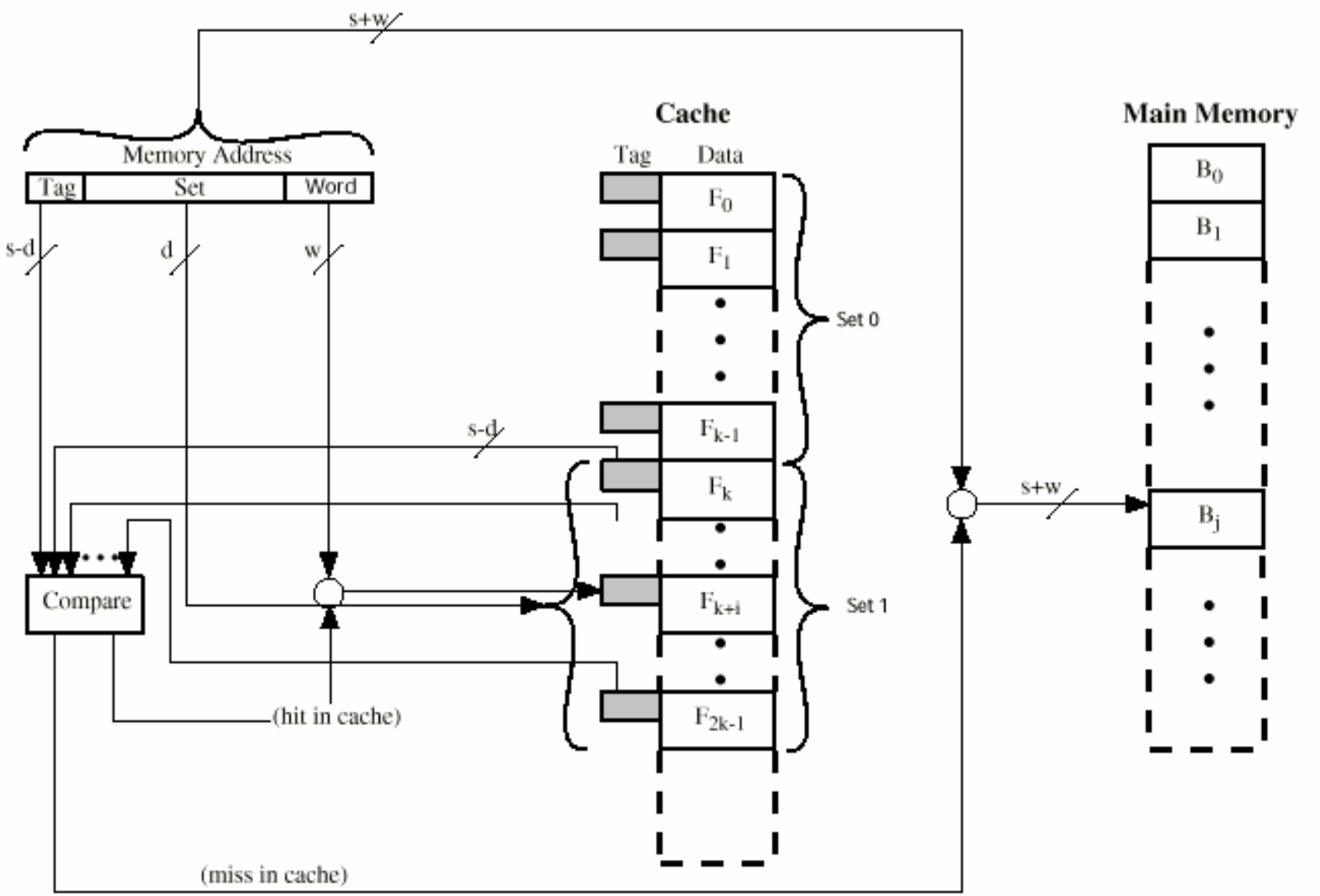
Tag 9 bit	Set 13 bit	Word 2 bit

- 24 bit address
- 2 bit word identifier (4 word per block)
- 22 bit block identifier
- 9 bit tag (=22-13)
- 13 set number, each having 2 slots
- Check contents of cache by finding SET and checking TAG

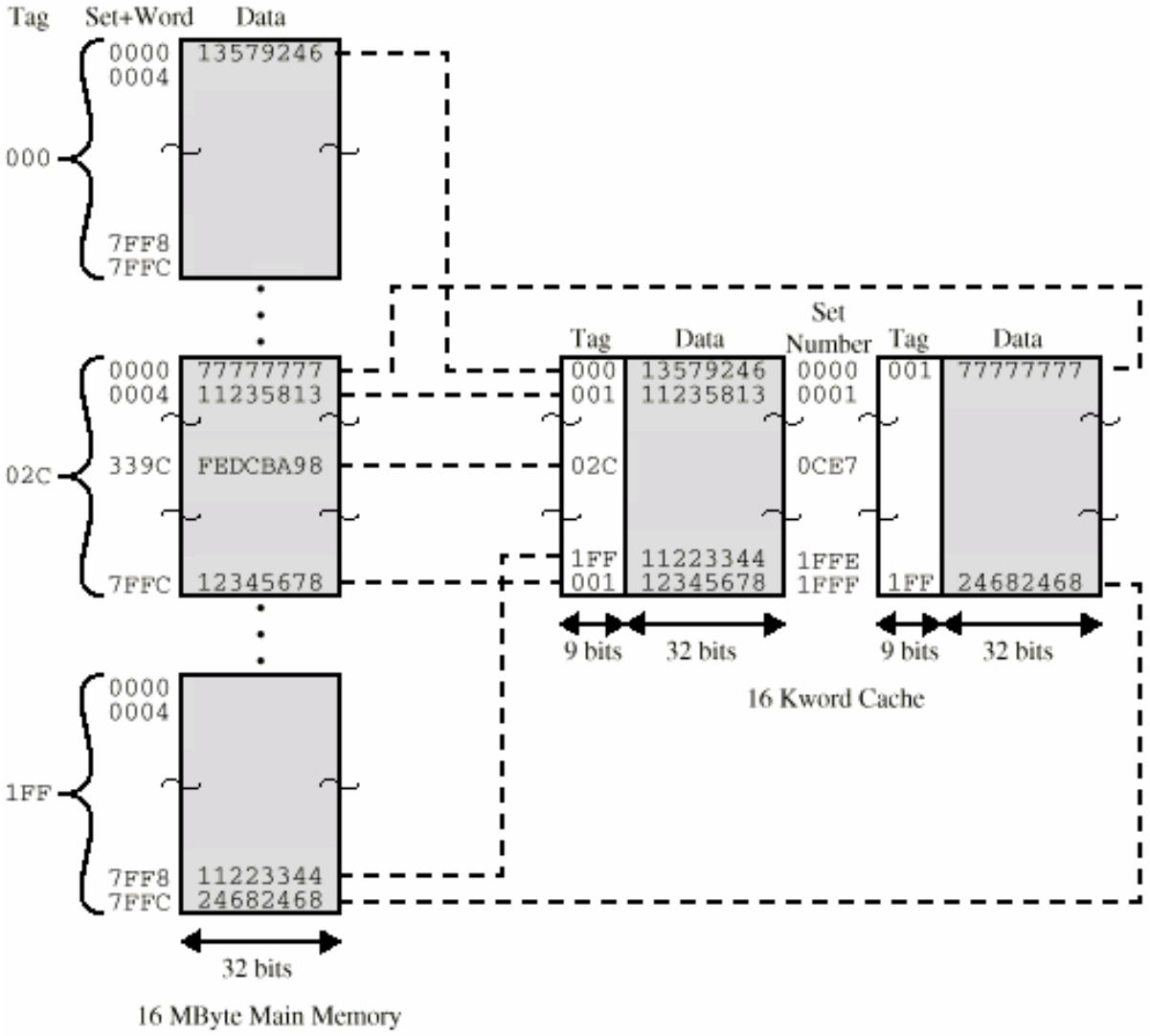
Contoh Set Associative Mapping

- Nomor set 13 bit
- Nomor Block dlm main memori adl modulo 2^{13}
- 000000, 00A000, 00B000, 00C000 … map ke set yang sama

Organisasi Cache: Two Way Set Associative



Contoh Two Way Set Associative Mapping



Replacement Algorithms (1)
Direct mapping

- Tidak ada pilihan
- Setiap block hanya di map ke 1 line
- Ganti line tersebut

Replacement Algorithms (2) Associative & Set Associative

- Implementasi algoritma bisa langsung pada hardware (cepat)
- Least Recently used (LRU)
  - Misal pada 2 way set associative
  - Mana dari kedua slot yang is LRU?
- First in first out (FIFO)
  - Replace slot yang paling lama berada di cache
- Least frequently used
  - Replace slot yang memiliki hit paling rendah
- Random

Direct Mapping Summary

- Panjang alamat = (s + w) bits
- Jumlah unit yang dapat dialamatkan = 2s+w words or bytes
- Besar blok = besar baris = 2w words or bytes
- Jumlah blok pada main memory = 2s+ w/2w = 2s
- Jumlah baris di cache = m = 2r
- Ukuran tag = (s – r) bits

Associative Mapping Summary

- Panjang alamat = (s + w) bits
- Jumlah unit yang dapat dialamatkan = 2s+w words or bytes
- Besar blok = Besar baris = 2w words or bytes
- Jumlah blok di main memory = 2s+ w/2w = 2s
- Jumlah baris di cache = bebas
- Ukuran tag = s bits

Set Associative Mapping Summary

- Panjang Alamat = (s + w) bits
- Jumlah unit yang bisa dialamatkan = 2s+w wordsor bytes
- Besar blok = besar baris = 2w words or bytes
- Jumlah blok di main memory = 2d
- Jumlah baris dalam sebuah set = k
- Jumlah set = v = 2d
- Jumlah baris dalam satu cache = kv = k * 2d
- Ukuran tag = (s – d) bits