CACHE MEMORY

(Direview oleh: Arfika Nurhudatiana, Ph.D.)

Memori merupakan tempat menampung data dan kode instruksi program. Memori adalah pusat kegiatan pada sebuah komputer, karena setiap proses yang akan dijalankan harus melalui memori terlebih dahulu. Sistem operasi bertugas untuk mengatur peletakan banyak proses pada suatu memori. Manajemen memori berkaitan dengan aktivitas pengelolaan penggunaan memori pada saat komputer aktif dan menjalankan proses-proses.
Pengertian *cache memory* adalah memori yang berukuran kecil yang sifatnya *temporary* (sementara). Walaupun ukuran *file*-nya sangat kecil namun kecepatannya sangat tinggi. Dalam terminologi *hardware*, istilah ini biasanya merujuk pada memori berkecepatan tinggi yang menjembatani aliran data antara prosesor dengan memori utama (RAM) yang biasanya memiliki kecepatan yang lebih rendah.

Berfungsi mempercepat kerja memori sehingga mendekati kecepatan prosesor. *Cache memory* berisi salinan memori utama. Memori utama lebih besar kapasitasnya namun lambat operasinya, sedangkan *cache memory* berukuran kecil namun lebih cepat. *Buffer* berkecepatan tinggi yang digunakan untuk menyimpan data yang diakses pada saat itu dan data yang berdekatan dalam memori utama. Waktu akses memori *cache* lebih cepat 5–10 kali dibandingkan memori utama.

Fungsi dari *cache memory* adalah sebagai tempat menyimpan data sementara atau intruksi yang diperlukan oleh *processor*. Secara gampangnya, *cache* berfungsi untuk mempercepat akses data pada komputer karena *cache* menyimpan data atau informasi yang telah diakses oleh suatu *buffer*, sehingga meringankan kerja *processor*. Jadi Bisa disimpulkan fungsi *cache memory* yaitu:

1. Mempercepat akses data pada komputer
2. Meringankan kerja prosesor
3. Menjembatani perbedaan kecepatan antara CPU dan memori utama
4. Mempercepat kinerja memori

Cara Kerja dari *Cache Memory*

Jika prosesor membutuhkan suatu data, pertama-tama dia akan mencarinya pada *cache*. Jika data ditemukan, prosesor akan langsung membacanya dengan *delay* yang sangat kecil. Tetapi jika data yang dicari tidak ditemukan, prosesor akan mencarinya pada RAM yang kecepatannya lebih rendah. Pada umumnya, *cache* dapat menyediakan data yang dibutuhkan oleh prosesor sehingga pengaruh kerja RAM yang lambat dapat dikurangi.

Dengan cara ini maka *memory bandwidth* akan naik dan kerja prosesor menjadi lebih efisien. Selain itu kapasitas *cache memory* yang semakin besar juga akan meningkatkan kecepatan kerja komputer secara keseluruhan. Dua jenis *cache* yang sering digunakan dalam dunia komputer adalah *memory caching* dan *disk caching*. Implementasinya dapat berupa sebuah bagian khusus dari memori utama komputer atau sebuah media penyimpanan data khusus yang berkecepatan tinggi.

Cara Pembacaan pada *Cache*

- CPU meminta data 1 alamat
- Data akan dicari di lokasi *cache*
- Jika ada maka akan langsung dikirim ke CPU
- Jika tidak ditemukan, *cache* akan meminta atau mengambil 1 blok data yang mengandung alamat yang diminta dari *main memory*
- Akan dikirim satu blok data ke *cache*, *cache* akan mengirim 1 alamat yang diminta CPU
- Sisa data dari *main memory* akan disimpan di *cache*

Prinsip Kerja *Cache Memory*

- *Cache* berisi salinan sebagian isi memori utama.
- Pada saat CPU membaca sebuah *word memory*, dilakukan pemeriksaan untuk mengetahui apakah *word* berada di *cache.*
- Jika *word* berada di *cache*, maka akan dikirimkan ke CPU yang dikenal sebagai proses HITT.

- Jika tidak ada maka blok memori utama yang terdiri dari sejumlah *word* tetap akan diletakkan di *cache* yang dikenal sebagai proses MISS dan selanjutnya dikirim ke CPU.

Elemen Rancangan *Cache*

- ***Size*/Kapasitas**

  Ukuran memori *cache* sangatlah penting untuk mendongkrak kinerja komputer. Semakin besar kapasitas *cache* tidak berarti semakin cepat prosesnya, dengan ukuran besar akan terlalu banyak gate pengalamatannya sehingga akan memperlambat proses.

- ***Mapping Function*/Fungsi Pemetaan**

  Mapping function cache memory yang besar akan mengurangi kecepatan pada saat pencarian data.

  Terdapat tiga metode, yaitu:

  1. Pemetaan langsung
  2. Pemetaan asosiatif
  3. Pemetaan asosiatif set

- ***Replacement Algorithm*/Algoritme Pengganti**

  Digunakan untuk menentukan blok mana yang harus dikeluarkan dari *cache* untuk menyiapkan tempat bagi blok baru.

  Ada 2 metode, yaitu:

  1. *Write- through*
  2. *Write-Back*

- ***Write Policy***

  1. *Write-Through*

     - ✓ *Cache* dan memori utama di-*update* secara bersamaan waktunya.
     - ✓ Keunggulannya salinan data di memori utama dan *cache* tetap.
     - ✓ Kelemahannya pada proses *write* memerlukan jumlah waktu sama dengan proses MISS.

2. *Write-Back*

    - Melakukan *update* data di memori utama hanya pada saat *word* memori telah dimodifikasi dari *cache.*
    - Keunggulannya proses *update word* tidak terbatas.
    - Kelemahannnya salinan data di memori utama tidak tetap/konsisten selama data termodifikasi benar-benar ada di memori utama.

- ***Block Size*/Ukuran Blok**
- ***Number of Caches*/Jumlah *Cache***
    - Disesuaikan kebutuhannya dalam membantu kerja memori utama.
    - Semakin besar ukuran *cache*, maka semakin besar pula jumlah gerbang yang terdapat dalam pengalamatan *cache*, yang mengakibatkan *cache* berukuran besar akan lebih lambat dari *cache* yang berukuran kecil.
    - Ukuran *cache* antara 1K sampai 512K.

Penulis: *Heru Wijayanto, M.M., M.B.A., M.M.T.*

Sumber Referensi:

1. Abb-El-Barr, M., & El-Rewini, H. (2005). ***Fundamentals of computer organization and architecture*** (1st ed.). US: Wiley.
2. Patterson, D. A. (2014). ***Computer organization and design: The hardware/software interface*** (5th ed.). Netherlands: Elsevier.
3. Stallings, W. (2013). ***Computer organization and architecture*** (9th ed.). UK: Pearson.