

Praktikum 13

Manajemen Memori

POKOK BAHASAN:

- ✓ Ruang Alamat Logika dan Ruang Alamat Fisik
- ✓ Alokasi berurutan
- ✓ Paging
- ✓ Segmentasi

TUJUAN BELAJAR:

Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu:

- ✓ Memahami Ruang Alamat Logika dan Ruang Alamat Fisik
- ✓ Memahami Paging
- ✓ Memahami Segmentasi
- ✓ Membuat program simulasi penataan memori Alokasi berurutan
- ✓ Membuat program untuk menghitung perlambatan kecepatan memori akibat
 - Struktur paging
 - Segmentasi
 - Virtual Memori

DASAR TEORI:

1 ALOKASI MEMORI

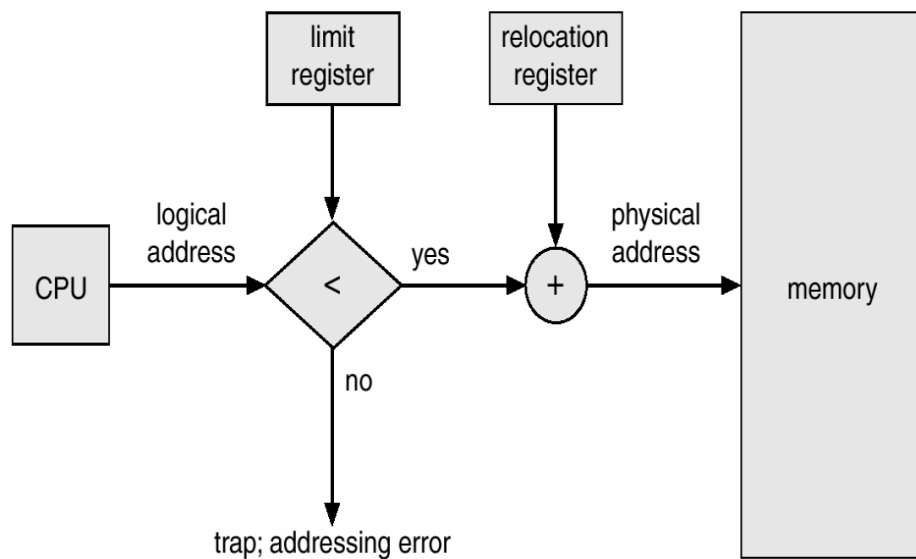
Memori utama biasanya dibagi ke dalam dua partisi yaitu untuk :

- Sistem operasi biasanya diletakkan pada alamat memori rendah dengan vector interupsi
- Proses user yang diletakkan pada alamat memori tinggi.

Alokasi proses user pada memori berupa single partition allocation atau multiple partition allocation.

1.1 Single Partition Allocation

Pada single partition allocation diasumsikan sistem operasi ditempatkan di memori rendah dan proses user dieksekusi di memori tinggi. Kode dan data system operasi harus diproteksi dari perubahan tak terduga oleh user proses. Proteksi dapat dilakukan dengan menggunakan register relokasi (relocation register) dan register limit (limit register). Register relokasi berisi nilai dari alamat fisik terkecil sedangkan register limit berisi jangkauan alamat logika dan alamat logika harus lebih kecil dari register limit. MMU memetakan alamat logika secara dinamis dengan menambah nilai pada register relokasi. Gambar 13.1 adalah perangkat keras yang terdiri dari register relokasi dan register limit.



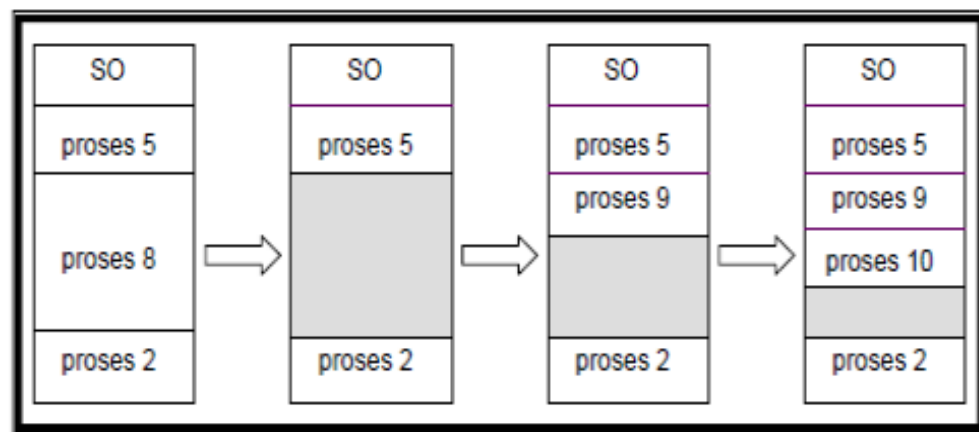
Gambar 13.1 : Perangkat keras untuk register relokasi dan limit

1.2 Multiple Partition Allocation

Pada multiple partition allocation, memungkinkan memori user dialokasikan untuk

proses yang berbeda yang berada di antrian input (input queue) yang menunggu dibawa ke memori.

Terdapat dua skema yaitu partisi tetap (fixed partition) dimana memori dibagi dalam sejumlah partisi tetap dan setiap partisi berisi tepat satu proses. Jumlah partisi terbatas pada tingkat multiprogramming. Digunakan oleh IBM OS/360 yang disebut Multiprogramming with a Fixed number of Task (MFT). Skema yang kedua adalah partisi dinamis (variable partition) merupakan MFT yang digeneralisasi yang disebut Multiprogramming with a Variable number of Tasks (MVT). Skema ini digunakan terutama pada lingkungan batch.

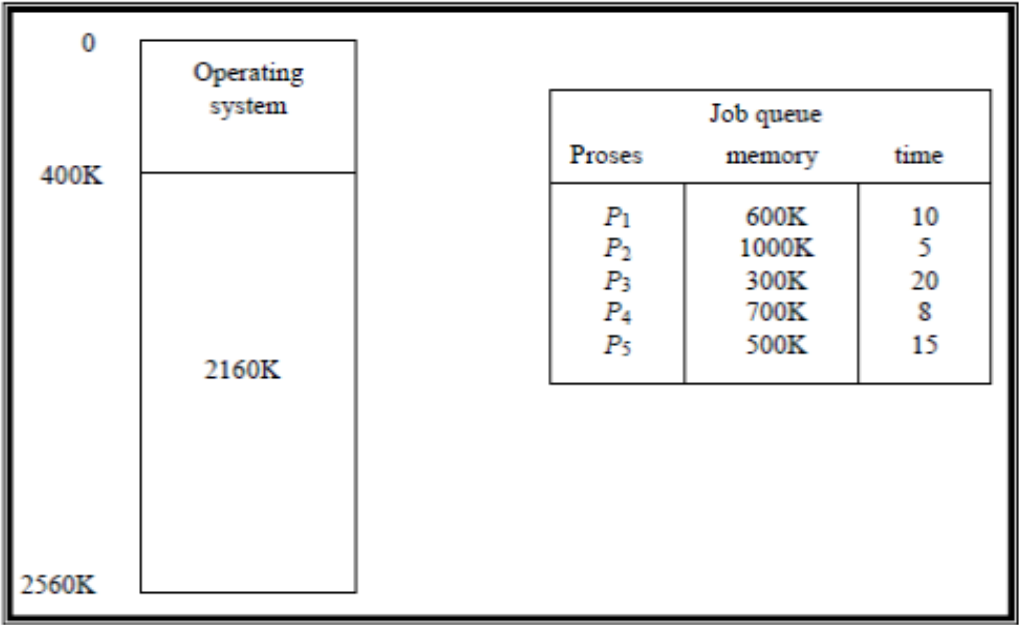


Gambar 13-2 : Hole untuk proses user

Pada MVT, sistem operasi menyimpan tabel yang berisi bagian memori yang tersedia dan yang digunakan. Mula-mula, semua memori tersedia untuk proses user sebagai satu blok besar (large hole). Lubang (hole) adalah blok yang tersedia di memori yang mempunyai ukuran berbeda. Bila proses datang dan memerlukan memori, dicari lubang (hole) yang cukup untuk proses tersebut seperti Gambar 13.2. Bila ditemukan memory manager mengalokasikan sejumlah memori yang dibutuhkan dan menyimpan sisanya untuk permintaan berikutnya. Sistem operasi menyimpan informasi tentang partisi yang dialokasikan dan partisi yang bebas (hole).

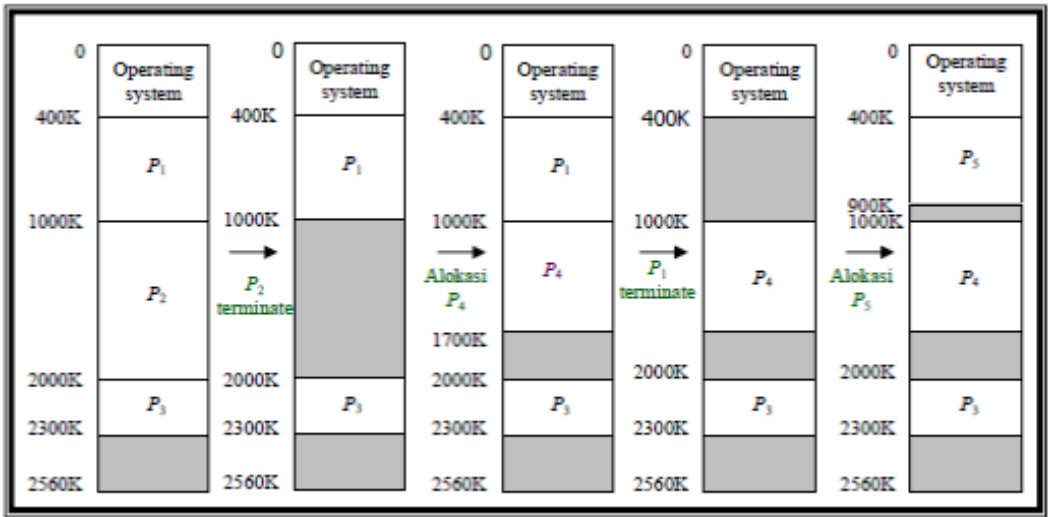
Sebagai ilustrasi, perhatikan contoh berikut pada Gambar 13-3. Diasumsikan tersedia memori 2560K dan untuk OS 400K. Sisa 2160K digunakan untuk user proses. Diasumsikan terdapat 5 job (P1 s/d P5) terdapat pada input queue. Diasumsikan penjadwalan FCFS digunakan untuk meletakkan job ke memori. Penjadwalan CPU

secara round-robin (quantum time = 1) untuk penjadwalan job yang sudah terdapat di memori.



Gambar 13-3 : Contoh proses yang akan dilakukan alokasi memori

Hasil alokasi berurutan pada lubang yang cukup untuk proses dapat dilihat pada Gambar 13-4



Gambar 13-4 : Alokasi memori pada contoh Gambar 13-3

Menggunakan MVT, terdapat beberapa lubang dengan ukuran berbeda. Bila proses datang dan memerlukan memori, dicari dari lubang yang cukup untuk proses.

Dynamic storage-allocation dapat dilibatkan untuk memenuhi permintaan ukuran n dari lubang yang bebas. Strategi yang digunakan meliputi :

- **First-fit** : alokasi lubang pertama yang cukup untuk proses.
- **Best-fit** : alokasi lubang terkecil yang cukup untuk proses. Strategi ini memerlukan pencarian keseluruhan lubang, kecuali bila ukuran sudah terurut.
- **Worst-fit** : alokasi lubang terbesar yang cukup untuk proses. Strategi ini memerlukan pencarian keseluruhan lubang, kecuali disimpan berdasarkan urutan ukuran.

Diantara algoritma diatas, first-fit dan best-fit lebih baik dibandingkan worst-fit dalam hal menurunkan waktu dan utilitas penyimpanan. Tetapi first-fit dan best-fit lebih baik dalam hal utilitas penyimpanan tetapi first-fit lebih cepat.

1.3 Implementasi Sistem Paging

Setiap sistem operasi mempunyai metode sendiri untuk menyimpan tabel page. Beberapa sistem operasi mengalokasikan sebuah tabel page untuk setiap proses. Pointer ke tabel page disimpan dengan nilai register lainnya dari PCB.

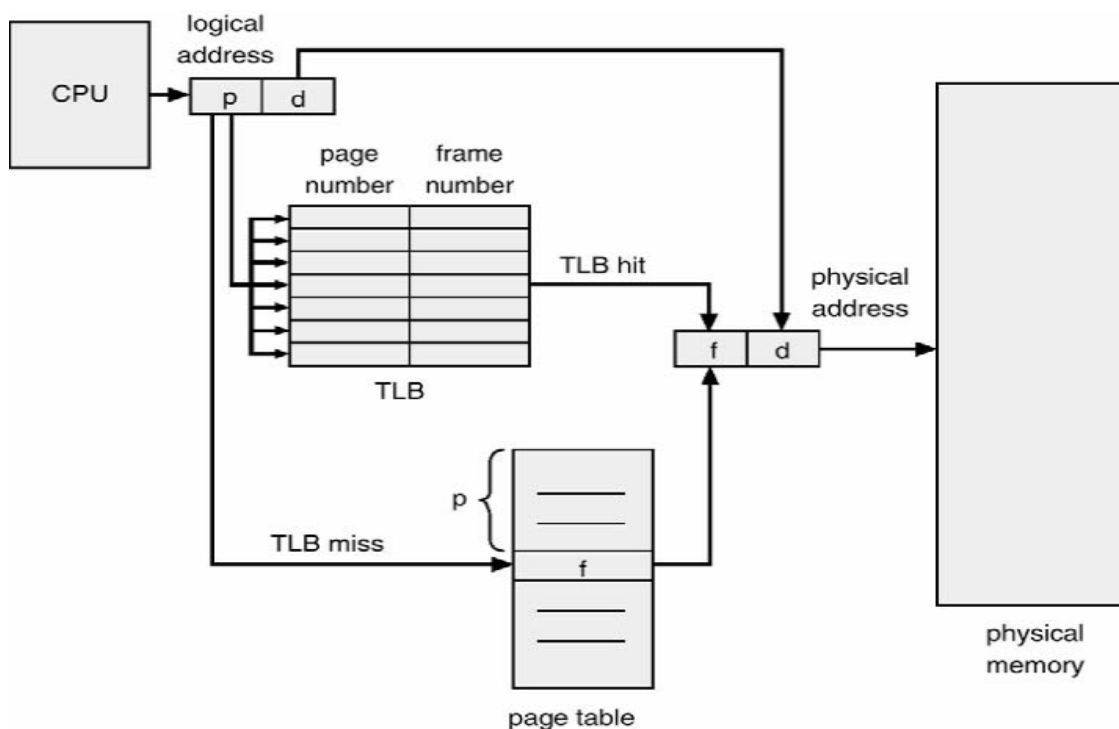
Pada dasarnya terdapat 3 metode yang berbeda untuk implementasi tabel page :

1. Tabel page diimplementasikan sebagai kumpulan dari “dedicated” register. Register berupa rangkaian logika berkecepatan sangat tinggi untuk efisiensi translasi alamat paging. Contoh : DEC PDP-11. Alamat terdiri dari 16 bit dan ukuran page 8K. Sehingga tabel page berisi 8 entri yang disimpan pada register. Penggunaan register memenuhi jika tabel page kecil (tidak lebih dari 256 entry).
2. Tabel page disimpan pada main memori dan menggunakan page table base register” (PTBR) untuk menunjuk ke tabel page yang disimpan di main memori. Penggunaan memori untuk mengimplementasikan tabel page akan memungkinkan tabel page sangat besar (sekitar 1 juta entry). Perubahan tabel page hanya mengubah PTBR dan menurunkan waktu context-switch. Akan tetapi penggunaan metode ini memperlambat akses memori dengan faktor 2. Hal ini dikarenakan untuk mengakses memori perlu dua langkah : pertama untuk lokasi tabel page dan kedua untuk lokasi alamat fisik yang diperlukan.

3. Menggunakan perangkat keras cache yang khusus, kecil dan cepat yang disebut associative register atau translation look-aside buffers (TLBs). Merupakan solusi standar untuk permasalahan penggunaan memori untuk implementasi tabel page. Sekumpulan associative register berupa memori kecepatan tinggi. Setiap register terdiri dari 2 bagian yaitu key dan value. Jika associative register memberikan item, akan dibandingkan dengan semua key secara simultan. Jika item ditemukan nilai yang berhubungan diberikan. Model ini menawarkan pencarian cepat tetapi perangkat keras masih mahal. Jumlah entry pada TLB bervariasi antara 8 s/d 2048.

Mekanisme penggunaan associative register (Gambar 7-15) adalah sebagai berikut :

- Associative register berisi hanya beberapa entry tabel page (sampai dengan ukuran maksimum).
- Jika memori logika dibangkitkan oleh CPU, nomor page berupa sekumpulan associative register yang berisi nomor page dan nomor frame yang berkorespondensi.
- Jika nomor page ditemukan pada associative register, nomor frame segera tersedia dan digunakan untuk mengakses memori.
- Sebaliknya, jika nomor page tidak ditemukan pada associative register, acuan memori ke tabel page harus dibuat. Jika nomor frame tersedia, maka dapat menggunakannya untuk mengakses ke memori yang tepat.
- Kemudian ditambahkan nomor page dan nomor frame ke associative register sehingga akan mudah ditemukan pada acuan berikutnya.
- Setiap kali tabel page baru dipilih, TLB harus dihapus untuk menjamin eksekusi proses berikutnya tidak menggunakan informasi translasi yang salah.



Gambar 13-5 :Perangkat keras paging dengan TLB

Persentase waktu sebuah page number ditemukan pada associative register disebut hit ratio. Hit ratio 80% berarti penemuan page number yang tepat pada associative register adalah 80% dari waktu. Misalnya, untuk mencari entry di associative register memerlukan waktu 20 ns dan untuk mengakses memori memerlukan waktu 100 ns sehingga untuk memetakan ke memori memerlukan waktu 120 ns. Apabila tidak menemukan page number pada associative register (20 ns), maka harus lebih dahulu mengakses tabel page di memori (100 ns) dan kemudian akses ke lokasi memori yang tepat (100 ns). Maka effective access time (EAT) menjadi

$$0.8 \times 120 + 0.2 \times 220 = 140 \text{ ns}$$

Artinya terjadi 40% penurunan kecepatan waktu akses memori. Hit ratio berhubungan dengan jumlah associative register. Apabila jumlah associative register antara 16 s/d 512, maka hit ratio yang dapat dicapai antara 80% sampai 98%. Processor Motorola 68030 yang digunakan pada sistem Apple Mac mempunyai TLB 22 entry. CPU Intel 80486 mempunyai 32 register dan hit ratio 98%.

TUGAS PENDAHULUAN:

1. Terdapat partisi memori 100K, 500K, 200K, 300K dan 600K, bagaimana algoritma First-fit, Best-fit dan Worst-fit menempatkan proses 212K, 417K, 112K dan 426K (berurutan) ? Algoritma mana yang menggunakan memori secara efisien?
2. Apa yang dimaksud dengan fragmentasi eksternal dan fragmentasi internal ?
3. Diketahui ruang alamat logika dengan 8 page masing-masing 1024 word dipetakan ke memori fisik 32 frame.
4. Berapa bit alamat logika ?
5. Berapa bit alamat fisik ?
6. Diketahui sistem paging dengan page table disimpan di memori
7. Jika acuan ke memori membutuhkan 200 nanosecond, berapa lama waktu melakukan paging ?
8. Jika ditambahkan associative register, dan 75 persen dari semua acuan ke page-table ditemukan dalam associative register, berapa effective access time (EAT) acuan ke memori ? (diasumsikan bahwa menemukan entri pada page table di associative register membutuhkan waktu 0, jika entri ada).

Percobaan 1 : Alokasi Memori

1. Buatlah program untuk menanimasikan tentang alokasi memori dengan menggunakan:
 - a. First Fit
 - b. Best Fit
 - c. Worst Fit

Nb: sesuai dengan Tugas Pendahuluan no 1

Percobaan 2 : Paging dan Segmentasi

1. Buatlah program untuk menghitung penurunan kecepatan memori seperti kasus yang diselesaikan pada tugas pendahuluan no 7

LAPORAN RESMI:

1. Analisa hasil percobaan yang Anda lakukan.
2. Buatlah Program di atas.
3. Berikan kesimpulan dari praktikum ini.