Manajemen Memori (2)

Sistem Paging

- Salah satu cara untuk mengatasi fragmentasi eksternal (proses lebih besar daripada partisi yang tersedia) adalah dengan teknik pengalokasian memori dengan paging
- Paging : memori fisik dibagi menjadi blok-blok dengan ukuran tertentu yang disebut dengan frame/page frame, sedangkan memori logika/maya dibagi menjadi blok-blok yang disebut page.
- Page fault / fault = terjadi jika program dieksekusi tapi blm di-load ke memori utama.

Alamat Logika dan Fisik

- Alamat logika (logical address/virtual address)

= alamat yang dihasilkan oleh CPU disebut alamat logika/alamat maya

- Alamat fisik (physical address)

= alamat program yang sesungguhnya pada memori.

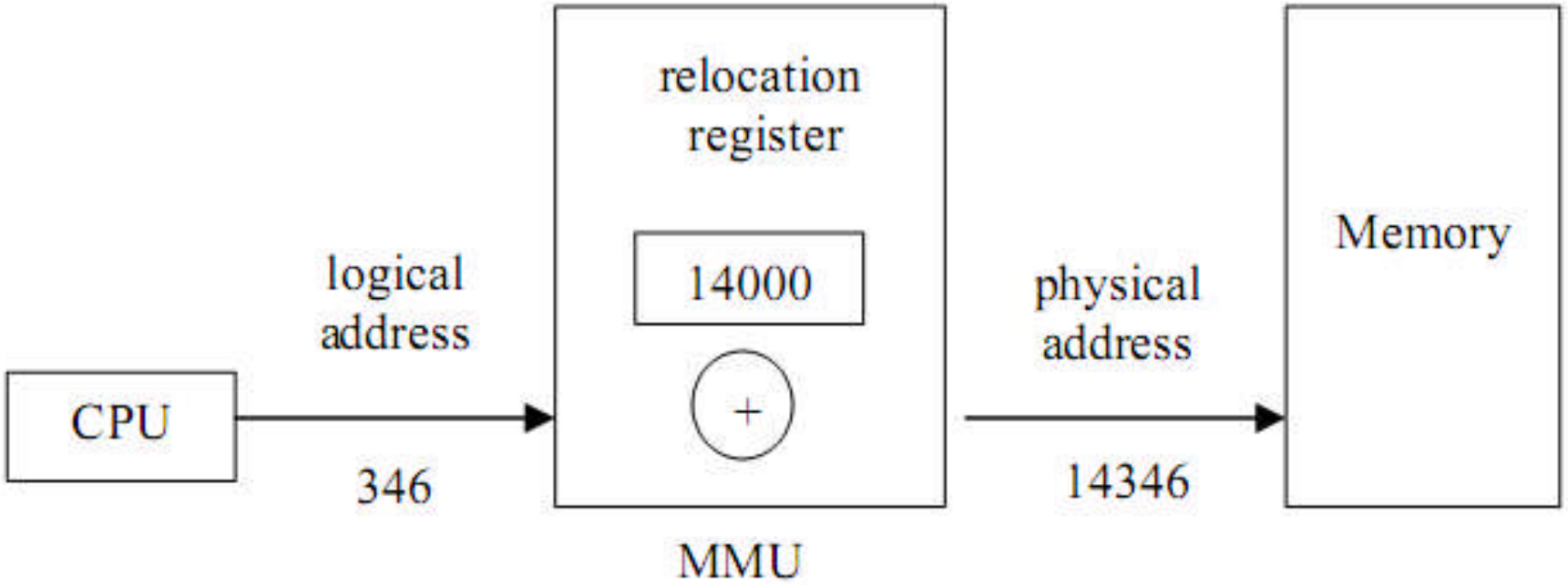
Alamat Logika dan Fisik

- Pada saat eksekusi setiap alamat logik harus dipetakan ke alamat fisik sehingga alamat logik berbeda dengan alamat fisik.
- Pemetaan dari alamat logik ke alamat fisik dilakukan dengan menggunakan perangkat keras yang disebut Memory Management Unit(MMU).

Alamat Logika dan Fisik

- MMU memiliki register relokasi yang berisi alamat awal proses.
- Nilai alamat awal ini akan ditambahkan ke setiap alamat logik pada proses untuk menciptakan alamat fisik.
- Sebagai contoh, bila alamat awal adalah 14000,maka pengguna yang ingin mengakses lokasi 0, secara otomatis akan dipetakan ke alamat 14000. Akses ke lokasi 346 akan dipetakan ke lokasi 14346.

Alamat Logika dan Fisik

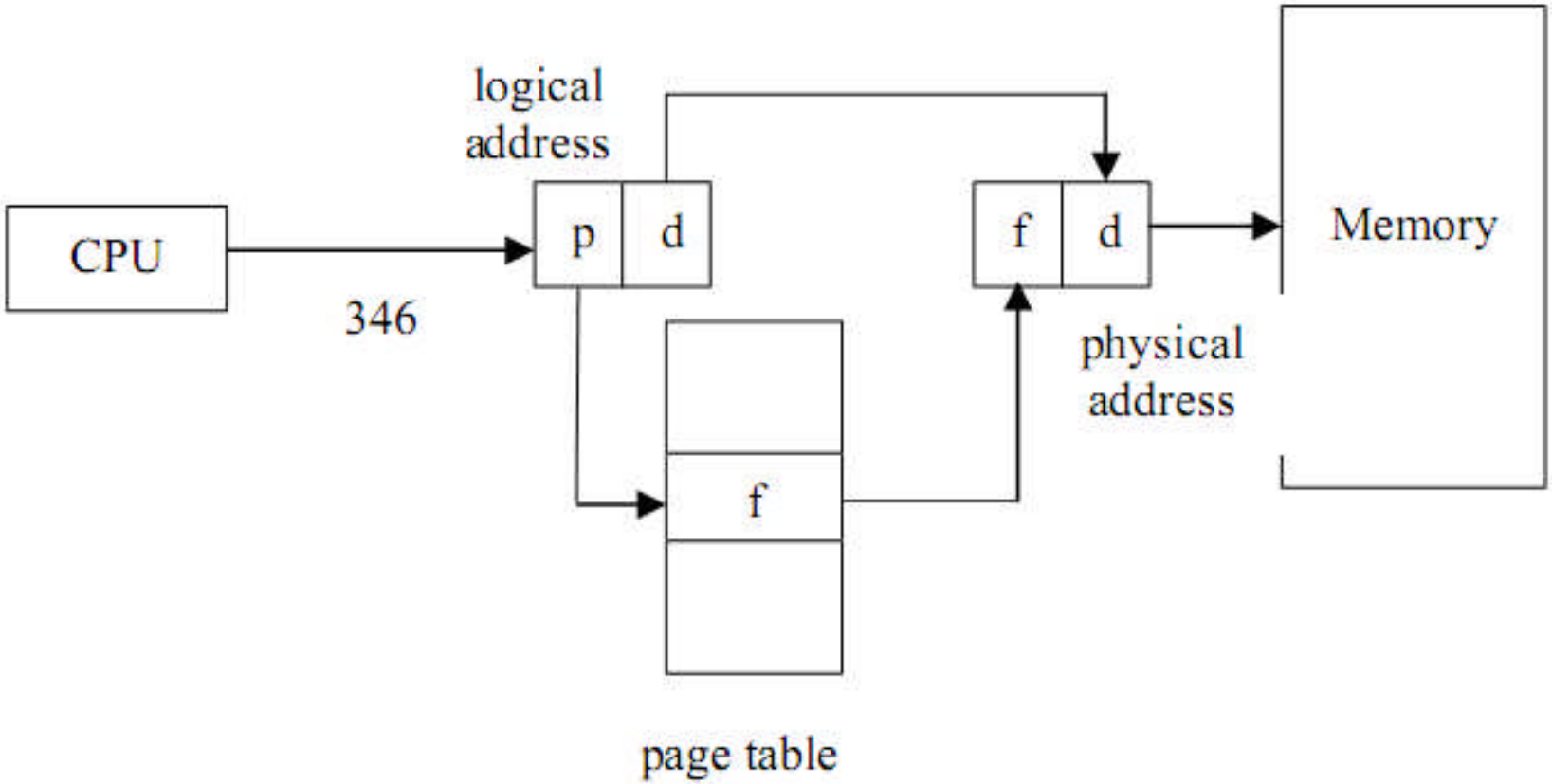


Alamat Logika dan Fisik

- Program milik pengguna tidak akan pernah mengetahui alamat fisik ini.
- Program dapat menciptakan penunjuk (pointer) ke lokasi dengan alamat logik 346, menyimpannya, memanipulasinya, dan membandingkannya dengan lokasi memori lain, tetapi semuanya menggunakan alamat 346.
- Program hanya berhubungan dengan alamat logik. Ketika instruksi tersebut akan dieksekusi (dimasukkan ke memori) barulah terjadi pemetaan dari alamat logik ke alamat fisik.

Sistem Paging

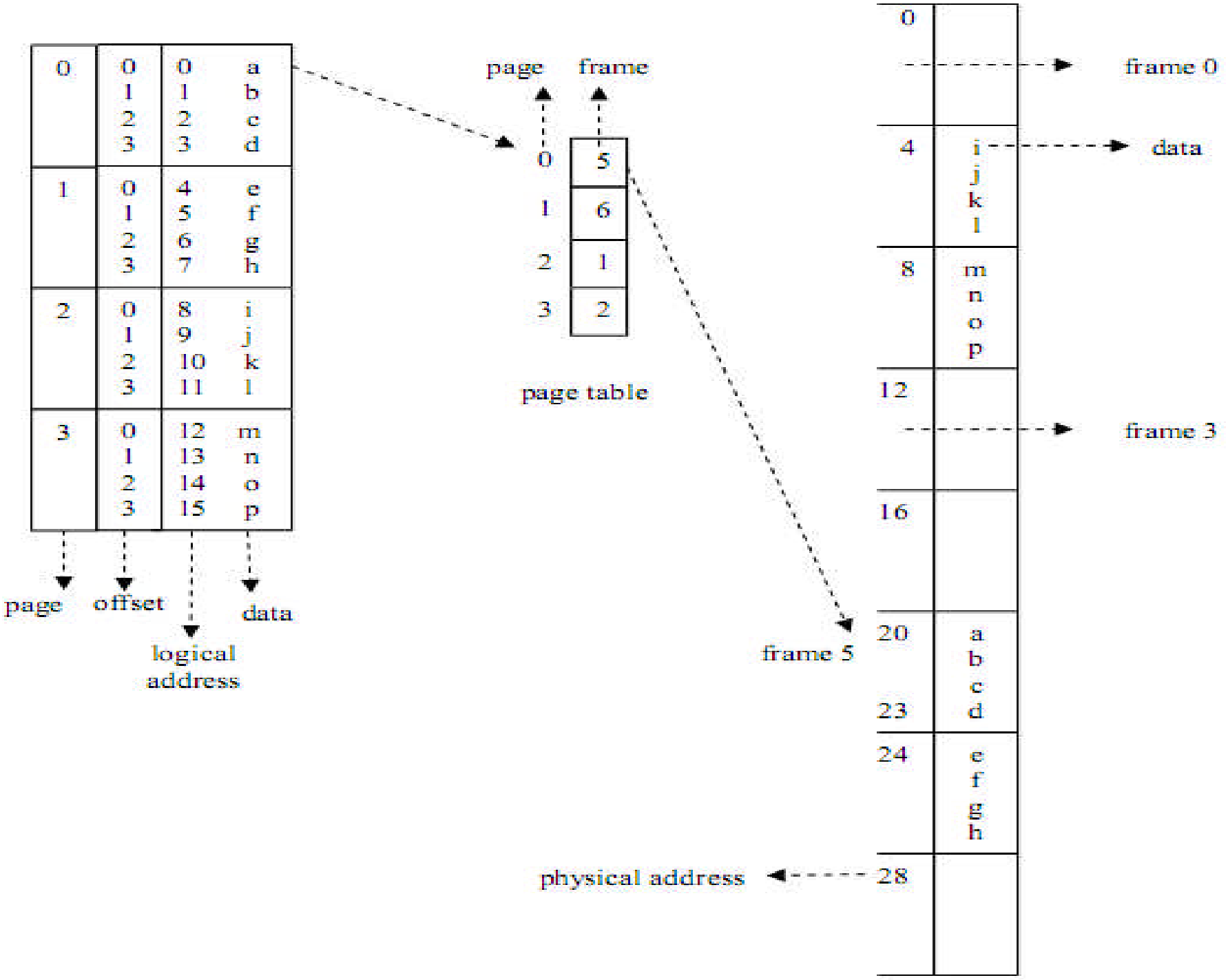
- Pada sistem paging alamat logika terdiri dari 2 bagian yaitu :
  - nomor page p : digunakan sebagai indeks untuk page table yang berisi alamat awal f untuk setiap page pada memori
  - offset page d : offset page ditambahkan pada alamat awal untuk menghasilkan alamat memori sebenarnya.



Page table = tabel page untuk semua proses yang ada di memori

Contoh

- Misal setiap page berukuran 4 byte, memory berukuran 32 byte (yang dapat menampung 8 page).
- Alamat logik 0 memiliki nomor page 0 dan offset 0. Ketika dihubungkan dengan page table, maka diketahui bahwa page 0 terletak pada frame 5.
- Maka alamat logik 0 akan dipetakan ke alamat fisik 20 (yang diperoleh dari (5 x 4) + 0).
- Untuk alamat logik 3 (dengan page 0 offset 3) maka akan dipetakan ke alamat fisik 23 (=(5 x 4) +3).
- Untuk alamat logik 4 (dengan page 1 offset 0) maka akan dipetakan ke alamat fisik 24 (=(6 x 4) +0).


0
1
2
3
0 1 2 3
0 1 2 3
a b c d
4 5 6 7
e f g h
8 9 10 11
i j k l
12 13 14 15
m n o p
page
offset
logical address
data
page
frame
0
1
2
3
5
6
1
2
page table
frame 5
0
4
8
12
16
20
23
24
28
frame 0
data
frame 3
physical address

Contoh

- Diasumsikan suatu sistem komputer memiliki memori utama dengan kapasitas 16 MB. Diketahui ukuran page sebesar 64 byte, maka :
  - Berapa jumlah frame yang tersedia
  - Jika suatu program COBA berukuran 914 byte, berapa page yang dibutuhkan.
  - Apabila diketahui page table sebagai berikut :

Contoh

Nomor page	Frame
0	8
1	2
2	10
3	22
4	12
5	1
:	:
:	:

- Dengan asumsi bahwa program membutuhkan page secara berurutan dari 0 sampai n, dimanakah letak alamat fisik dari alamat logika 50, 121, dan 380

Contoh

- Jawaban :
- Jumlah frame yang tersedia :
  - jumlah memori utama / page = 16 Mbyte / 64byte per page = 16.777.216 byte / 64 = 262.144 page
  - keterangan : 16 Mbyte = 16 * 1024 * 1024 = 16.777.216 byte
- COBA berukuran 914 byte sedangkan 1 page berukuran 64 byte, maka page yang dibutuhkan adalah 914 / 64 = 14,3 → 15 page
- 1 page = 64 byte.
  - Menurut page table diatas page 0 akan dipetakan ke frame 8, maka alamat logika 0 akan dipetakan ke alamat fisik (8 * 64) + 0 = 512.

Contoh

- Keadaan memori logika dapat digambarkan sebagai berikut :

Page	Memori Logika
0	0 : 63
1	64 : 127
2	128 : 191
3	192 : 255
4	256 : 319
5	320 : 383
6	384 :
:	:

Contoh

- Dari gambar tersebut dapat dilihat bahwa :
- alamat logika 50 berada di page 0, offset 50 sehingga alamat fisiknya (8 * 64) + 50 = 562
- alamat logika 121 berada di page 1,offset 57 sehingga alamat fisiknya (2 * 64) + 57 = 185
- alamat logika 380 berada di page 5,offset 60 sehingga alamat fisiknya (1 * 64) + 60 = 124

Contoh

- Keterangan :
- alamat offset diperoleh dari nilai absolut alamat logika yang ditentukan dikurangi dengan alamat logika awal dari page yang diketahui.
- Contoh : jika alamat logika 380 berarti alamat offsetnya adalah absolut(380 – 320) = 60

Algoritma Penggantian Page

- Jika suatu proses menginginkan page, tapi page tersebut belum ada pada memori utama, maka akan terjadi page fault.
- Untuk memilih page yang dikeluarkan dari memori maka diperlukan algoritma penggantian yang akan memberikan tingkat page fault yang terendah yang mungkin.

Algoritma Penggantian Page Acak

- Setiap terjadi page fault, page yang diganti dipilih secara acak.
- Teknik ini tidak memakai informasi apapun dalam menentukan page yang diganti.
- Semua page di memori utama mempunyai bobot sama untuk dipilih.
- Teknik ini dapat memilih sembarang page, termasuk page yang sedang diacu (page yang seharusnya tidak diganti).

Algoritma Penggantian Page FIFO (First In First Out)

- Merupakan algoritma penggantian page yang paling sederhana.
- Ketika suatu page harus diganti, maka page terlama yang dipilih.
- Pada algoritma ini akan muncul Anomali Belady dimana kecepatan page fault akan bertambah jika framenya juga bertambah.

Algoritma Penggantian Page FIFO (First In First Out)

String referensi	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
	7	7	7	2	2	2	2	4	4	4	0	0	0	0	0	0	0	7	7	7
		0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1	1	0	0
			1	1	1	1	0	0	0	3	3	3	3	3	2	2	2	2	2	1
Fault	F	F	F	F		F	F	F	F	F	F			F	F			F	F	F

Algoritma Penggantian Page FIFO (First In First Out)

- Anomali belady → jika frame bertambah maka lebih banyak page fault

String referensi	**0**	**1**	**2**	**3**	**0**	**1**	**4**	**0**	**1**	**2**	**3**	**4**	
	0	0	0	3	3	3	4	4	4	4	4	4	
		1	1	1	0	0	0	0	0	2	2	2	
			2	2	2	1	1	1	1	1	3	3	
Fault	F	F	F	F	F	F				F	F		Fault = 8

String referensi	**0**	**1**	**2**	**3**	**0**	**1**	**4**	**0**	**1**	**2**	**3**	**4**	
	0	0	0	0	0	0	4	4	4	4	3	3	
		1	1	1	1	1	1	0	0	0	0	4	
			2	2	2	2	2	2	1	1	1	1	
				3	3	3	3	3	3	2	2	2	
Fault	F	F	F	F			F	F	F	F	F	F	Fault = 10

Algoritma Penggantian Page Optimal

- Algoritma optimal akan mereplace/mengganti page yang tidak digunakan dalam waktu dekat.
- Contoh : pada referensi/penunjukan ke-4 terlihat page 7 akan direplace dengan 2 karena 7 baru akan digunakan lagi pada penunjukkan ke-18.
- Sedangkan page 0 akan digunakan pada penunjukan ke-5 dan page 1 pada penunjukan ke-14.

Algoritma Penggantian Page Optimal

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
String referensi	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
	7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	7	7	7
		0	0	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0
			1	1	1	3	3	3	3	3	3	3	3	1	1	1	1	1	1	1
Fault	F	F	F	F		F		F			F			F				F		

Algoritma Penggantian Page LRU (Least Recently Used)

- Algoritma FIFO = memandang page dari sisi berapa lama page tersebut telah berada di memori
- Algoritma Optimal = memandang page pada kapan page tersebut akan digunakan lagi
- Algoritma LRU = perpaduan FIFO dan Optimal → LRU akan me-replace page yang tidak baru saja dipakai

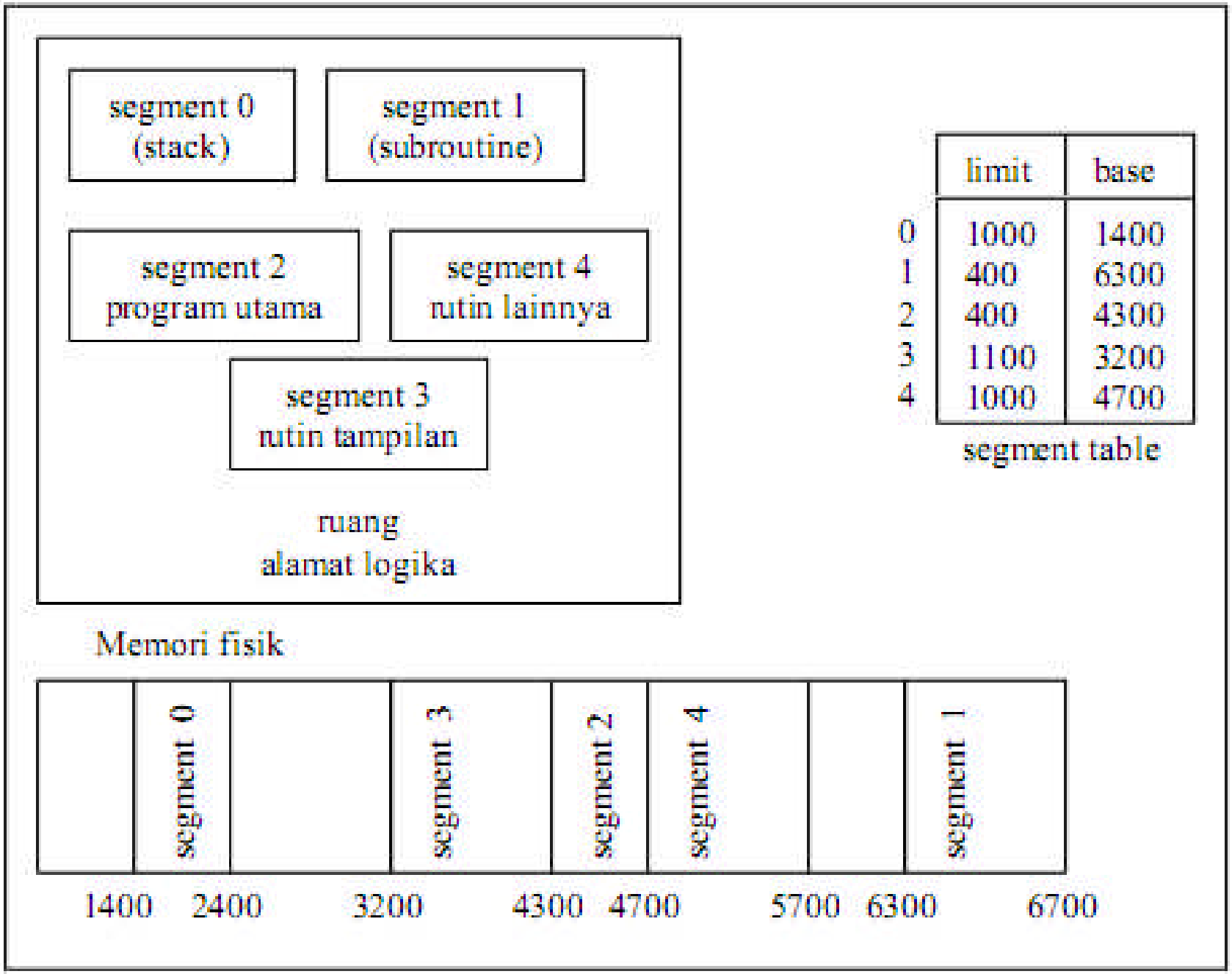
Algoritma Penggantian Page LRU (Least Recently Used)

String referensi	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
	7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
		0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
			1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7
Fault	F	F	F	F		F		F	F	F	F			F		F		F		

Segmentasi

- Paging merupakan suatu pembagian / pemecahan suatu ruang alamat logika yang tidak beraturan (tidak tetap) menjadi potongan-potongan kecil yang berukuran tetap.
- Ada suatu cara yang membagi suatu ruang alamat logika menjadi potongan-potongan kecil yang sesuai dengan ukuran logika suatu objek sehingga ukuran potongan tersebut tidak tetap.
- Potongan tersebut disebut segment.
- Jadi suatu ruang alamat logika merupakan suatu kumpulan segment yang tidak membutuhkan pengurutan di antara segment.

Contoh


segment 0
(stack)
segment 1
(subroutine)
segment 2
program utama
segment 4
rutin lainnya
segment 3
rutin tampilan
ruang
alamat logika
limit
base
0 1000 1400
1 400 6300
2 400 4300
3 1100 3200
4 1000 4700
segment table
Memori fisik
segment 0
segment 3
segment 2
segment 4
segment 1
1400
2400
3200
4300
4700
5700
6300
6700

Contoh

- Misal :
- Diberikan alamat awal segment dalam memori fisik (base) dan panjang segment (limit). Misal segment 2 mempunyai panjang 400 byte dan dimulai pada lokasi 4300, sehingga suatu referensi ke byte 53 dari segment 2 dipetakan ke lokasi : 4300 + 53 = 4353.
- Suatu referensi ke segment 3, byte 852 dipetakan ke 3200 + 852 = 4052