Manajemen Proses

Outline

- Proses
- Konkurensi
- Siklus Hidup Proses
- Struktur Kendali Sistem Operasi
- Modus Eksekusi Proses
- Konsep Thread

Proses: Pengertian

- **Definisi Umum:**

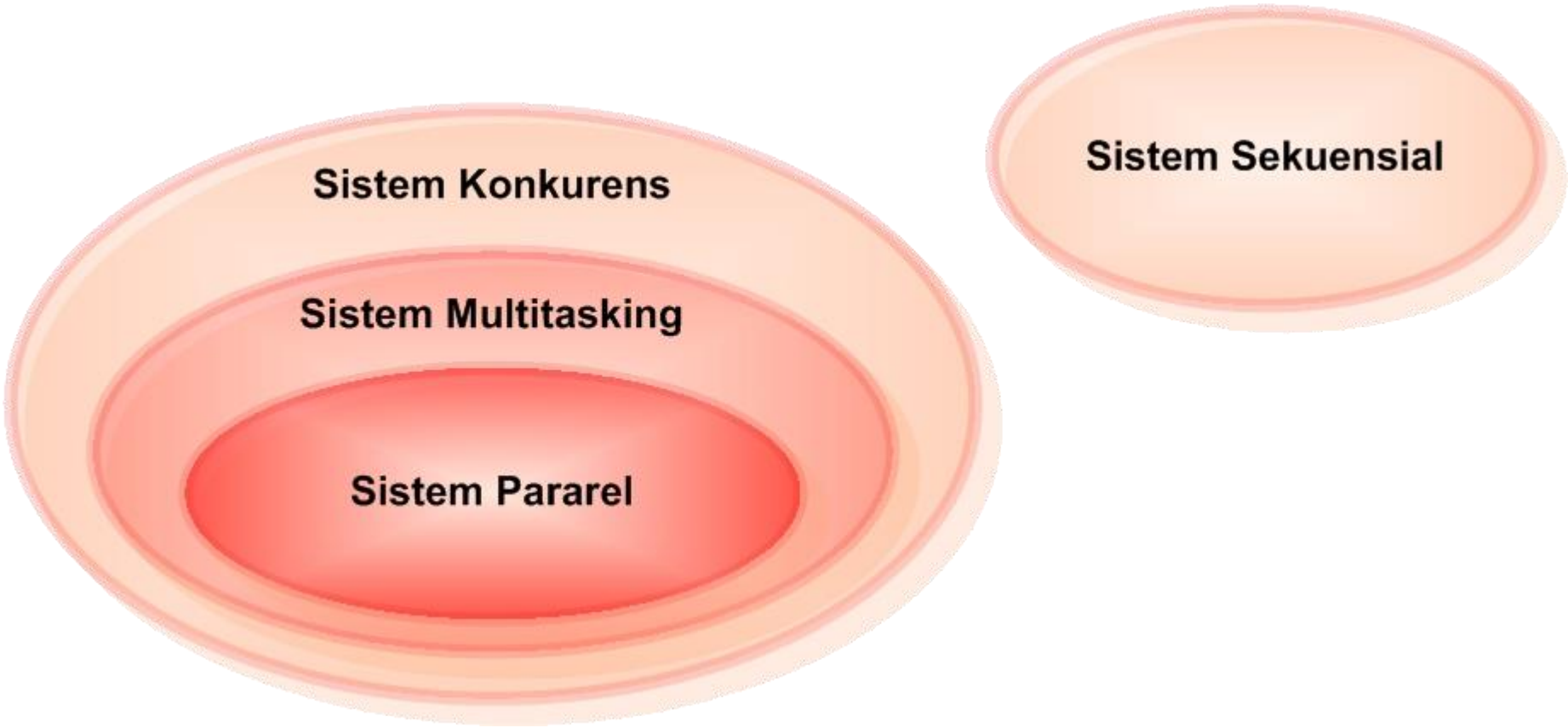
  Proses/*job/task* adalah bagian program yang sedang dijalanan/dieksekusi.

- **Proses *vs* program**

  Suatu proses lebih dari sekadar kode program itu sendiri. Dua proses yang berbeda bisa menjalankan kode program yang sama di memori, tapi memiliki status and atribut proses yang berbeda. Jadi, program adalah entitas pasif, sedang proses adalah entitas aktif.

- **Jadi suatu proses** meliputi kode program, data, *stack* dan atribut proses (informasi id proses, status proses, penggunaan sumber daya seperti memori, I/O, file,dll).

Berbagai Sistem Pemrosesan



Sistem Sekuensial

- Suatu proses dieksekusi sampai selesai, baru kemudian proses berikutnya dieksekusi
- Ketika proses yang sedang dieksekusi melakukan operasi I/O, prosesor harus menunggu sampai operasi tersebut selesai (prosesor tidak bisa dialihkan ke proses lain)
- Kelemahannya adalah tingkat utilitas prosesor rendah (idle time CPU besar)

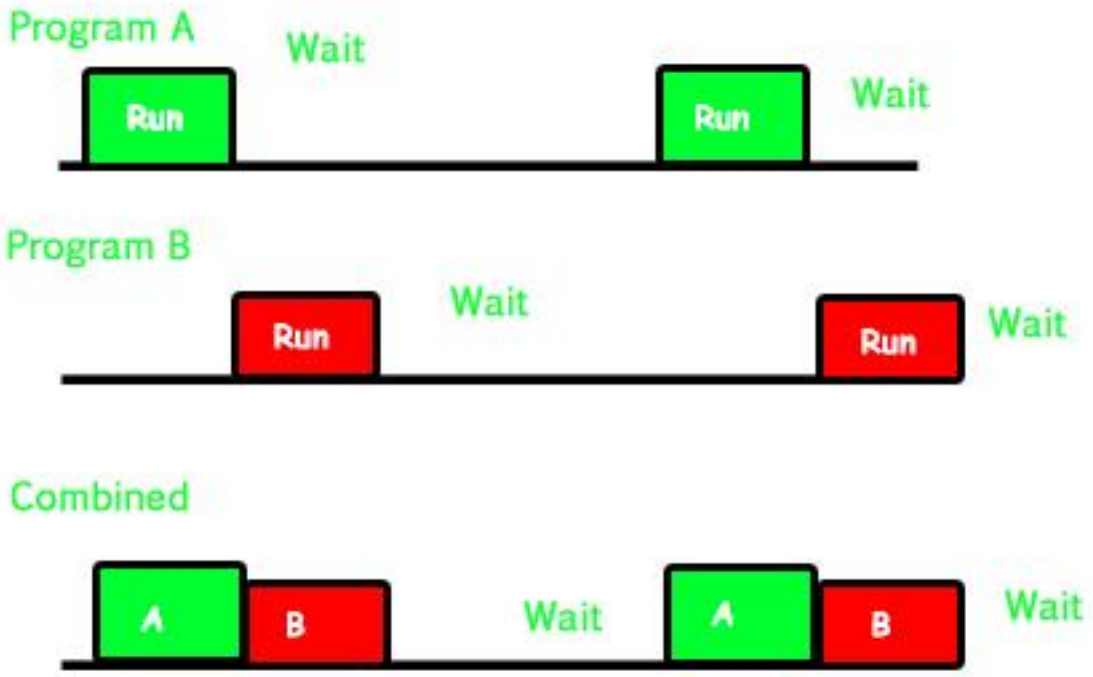
Konkurensi

- Utilitas prosesor yang rendah pada sistem sekuensial melahirkan konsep konkurensi (multitasking)
- Konkurensi merupakan keadaan dimana 2 atau lebih proses berjalan pada saat yang “bersamaan”.
- Bersamaan disini berarti suatu proses bisa aktif berjalan tanpa harus menunggu proses lainnya selesai seluruhnya.

Konkurensi dan Multiprogramming

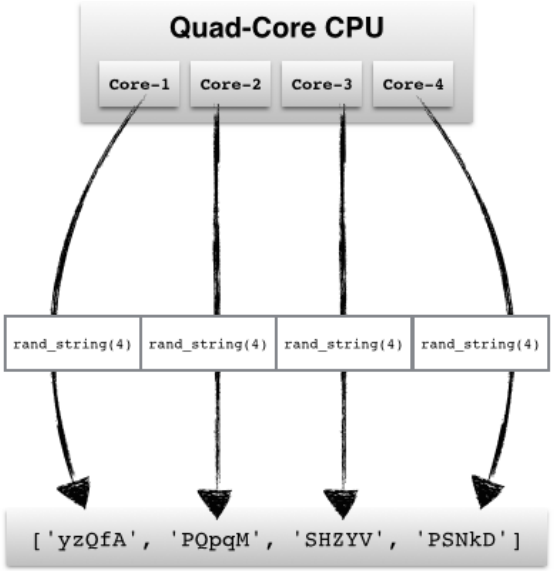
- Konkurensi adalah kondisi kejadian dimana banyak proses berjalan di waktu yang bersamaan.
- Konkuren dasar multiprogramming
- Pada sistem komputer dengan jumlah prosesor lebih dari satu atau satu prosesor multicore, proses-proses dieksekusi secara bersamaan pada saat yang bersamaan pula. Sistem ini dikenal dengan sebutan sistem paralel atau multiprocessing.
- Pada sistem single prosesor, konkurensi diimplementasikan dengan meng-*interleave* proses-proses, sedangkan pada sistem *multiprosesing*, proses-proses yg aktif di *interleave dan di overlap.*

Multi Programming

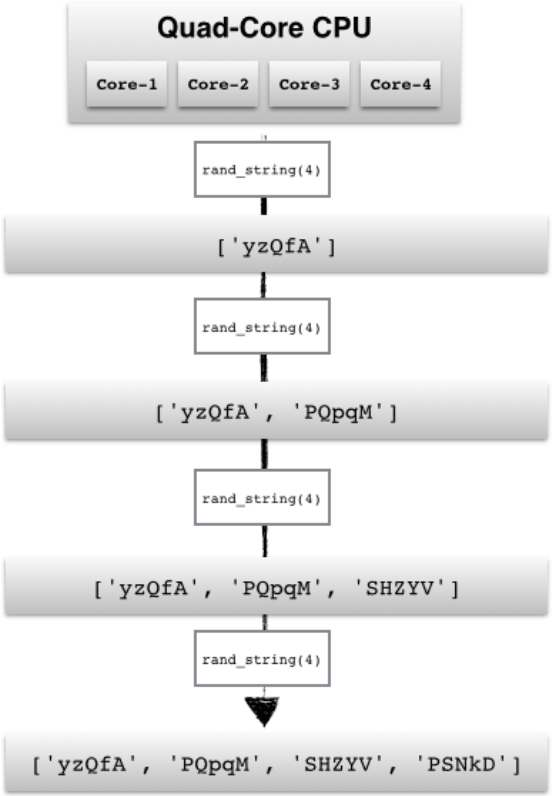


MultiProcessing

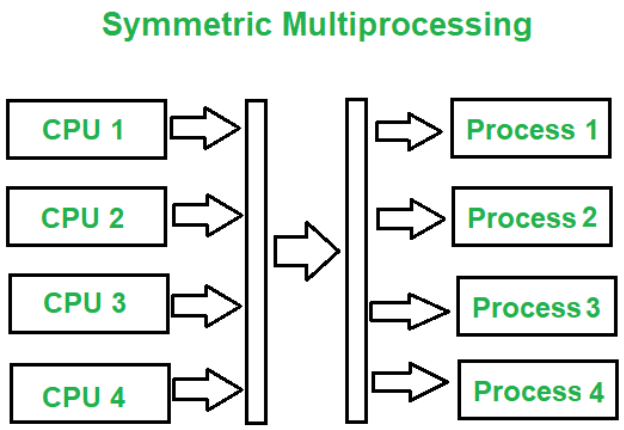
[parallel processing]

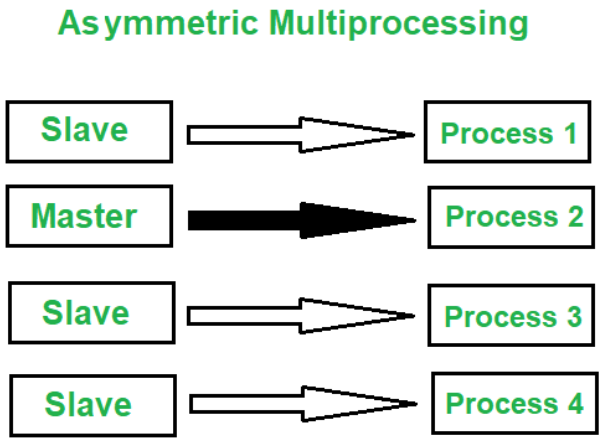


[serial processing]

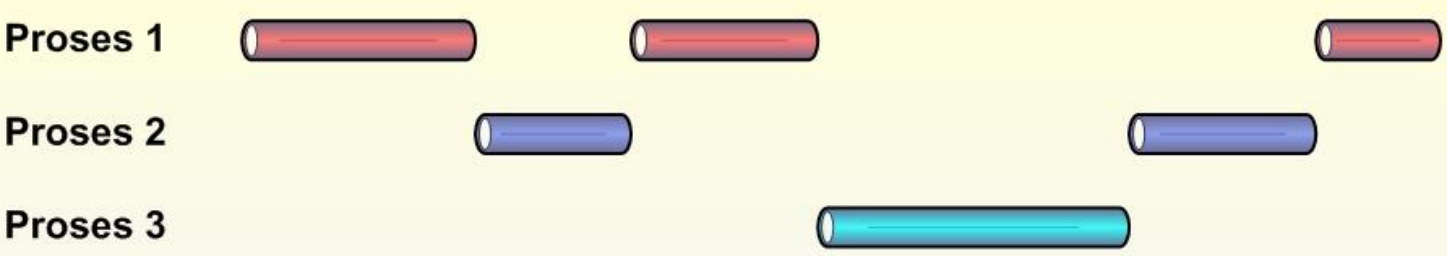


Multiprocessing

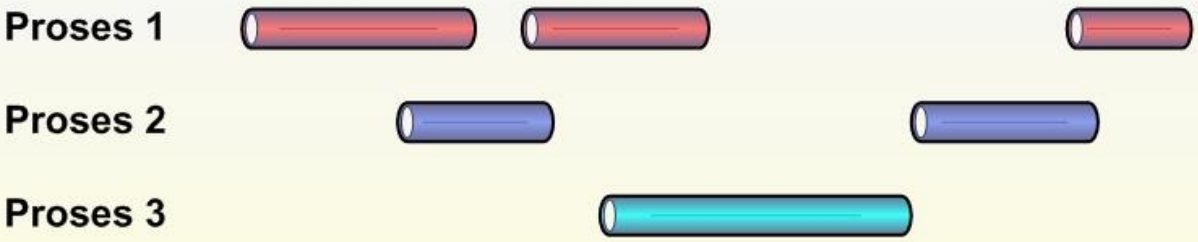




Implementasi Konkurensi



(a) *interleaving*



(a) *interleaving* dan *overlapping*

Permasalahan dalam Konkurensi

- **Permasalahan:** proses-proses yang konkurens memerlukan penanganan khusus seperti penjadwalan (karena kompetisi waktu prosesor), sinkronisasi (jika saling mempengaruhi), komunikasi (jika saling bekerja sama), dan penanganan deadlock (karena terjadi kompetisi pengggunaan resource).

Hubungan antar proses

- **Hubungan antar proses-proses konkuren**, dilihat dari **pengaruh** satu sama lain, bisa berbentuk:
  - **Saling bebas atau terpisah (*independence*),** yaitu proses-proses tidak saling menyadari dan tidak mempengaruhi satu sama lain.
  - **Saling mempengaruhi secara tidak langsung,** yaitu proses-proses tidak saling menyadari namun saling mempengaruhi karena saling berbagi pakai resource.
  - **Saling bekerja sama (*Cooperate*),** yaitu proses –proses secara sadar bekerja sama dan saling mempengaruhi satu sama lain.

    Contoh: Proses *parent-child* dan proses *consumer-producer*. Keuntungan dari kerjasama ini antara lain: Berbagi Informasi, Mempercepat komputasi,Modularitas, Kenyamanan.

Masalah Perebutan Sumber Daya

- Perebutan sumber daya antar prosesdalam sistem konkuren menimbulkan sejumlah masalah yang harus ditangani sistem operasi:
  - Race Condition
    - Suatu kondisi dimana ada dua atau lebih proses yang berusaha mengakses sumber daya yang sama, sehingga ada kemungkinan status sumber daya tersebut menjadi tidak valid
  - Deadlock
    - suatu kondisi dimana 2 proses atau lebih tidak bisa meneruskan eksekusinya karena saling menunggu aksi dari proses lainnya
  - Starvasion
    - kondisi dimana satu proses atau lebih selalu kalah dalam kompetisi untuk mendapatkan sumber daya yang diperlukannya, sehingga eksekusinya tertunda terus menerus.

Race Condition (1)

- Hasil proses bergantung pada urutan eksekusi setiap proses

  Contoh 1:

  - Dua buah proses P1 dan P2 sama-sama menggunakan variabel global a
  - P1 mengubah nilai a = 1, P2 mengubah nilai a = 2
  - Nilai a ditentukan oleh proses yang “kalah” →

    yang mengakses variabel a belakangan

Race Condition (2)

- Contoh 2:
  - Dua buah proses P3 dan P4 sama-sama menggunakan variabel global b dan c yang

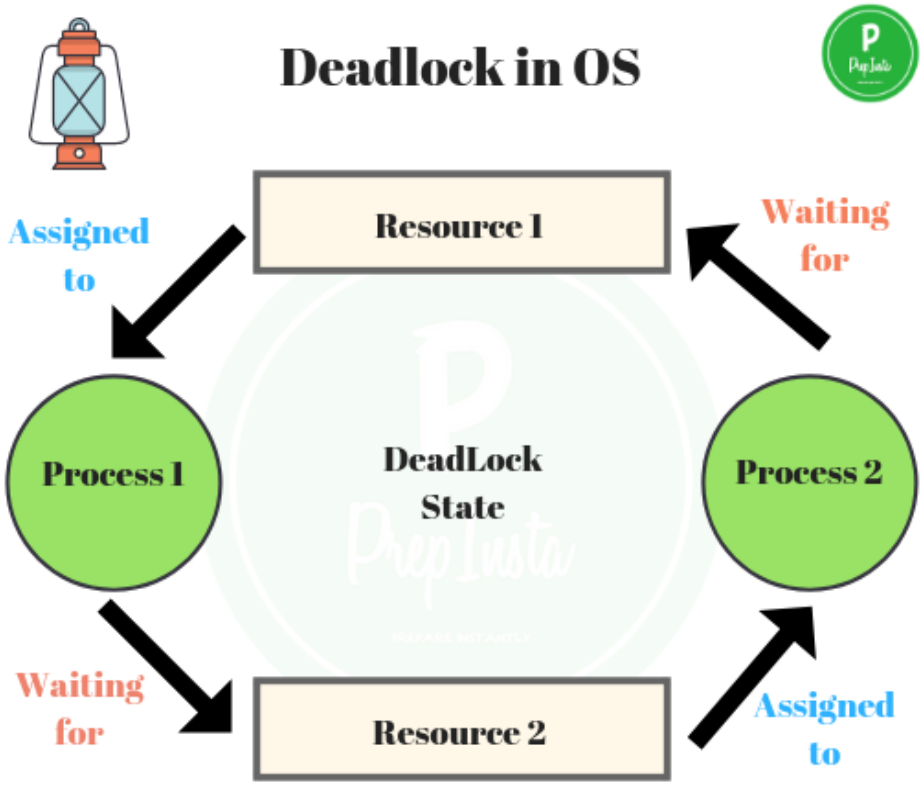
    masing-masing bernilai b=1 dan c=2
  - P3 menjalankan baris program b=b+c
  - P4 menjalankan baris pogram c=b+c
  - Jika P3 yang dieksekusi lebih dahulu, maka hasilnya: b=3 dan c=5
  - Jika P4 yang dieksekusi lebih dahulu, maka yg , hasilnya: b=4 dan c=3

→Hasil akhir sangat berbeda !!!

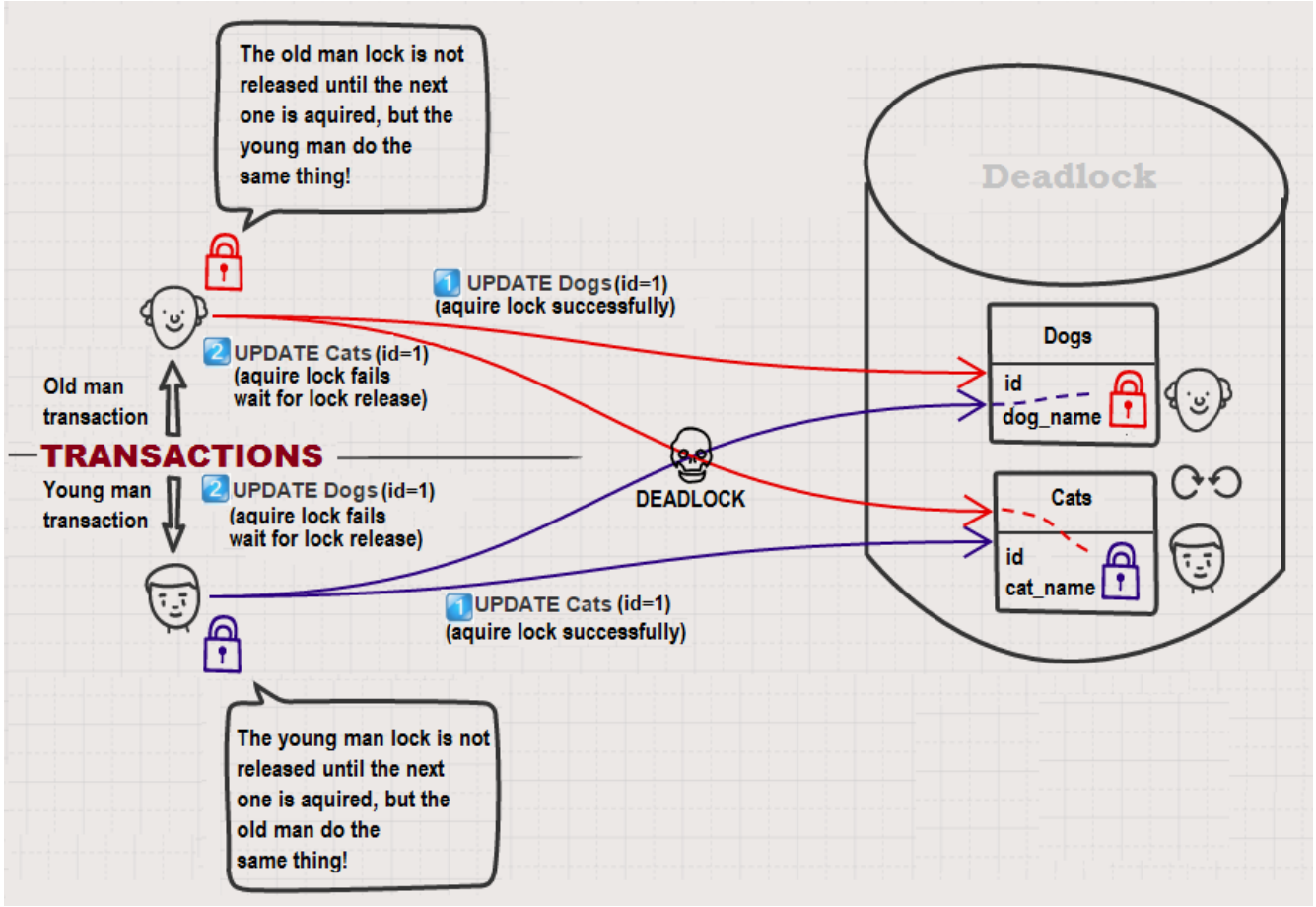
Peranan OS dalam Concurrency

- Apa yang harus dilakukan OS untuk memperoleh concurrency ?
  - OS harus dapat menjaga track (info state, (prioritas, resource, dll) setiap proses
  - OS harus dapat memberi dan mengambil resource (waktu prosesor, memory, file, I/O device) kepada setiap proses yang aktif.
  - OS harus dapat memproteksi data dan resource yang sedang digunakan suatu proses.
  - OS harus dapat menjamin hasil suatu proses tidak dipengaruhi oleh kecepatan pemrosesan.

Deadlock


Deadlock in OS
Assigned to
Resource 1
Waiting for
Process 1
DeadLock State
Process 2
Waiting for
Resource 2
Assigned to

Database Deadlock



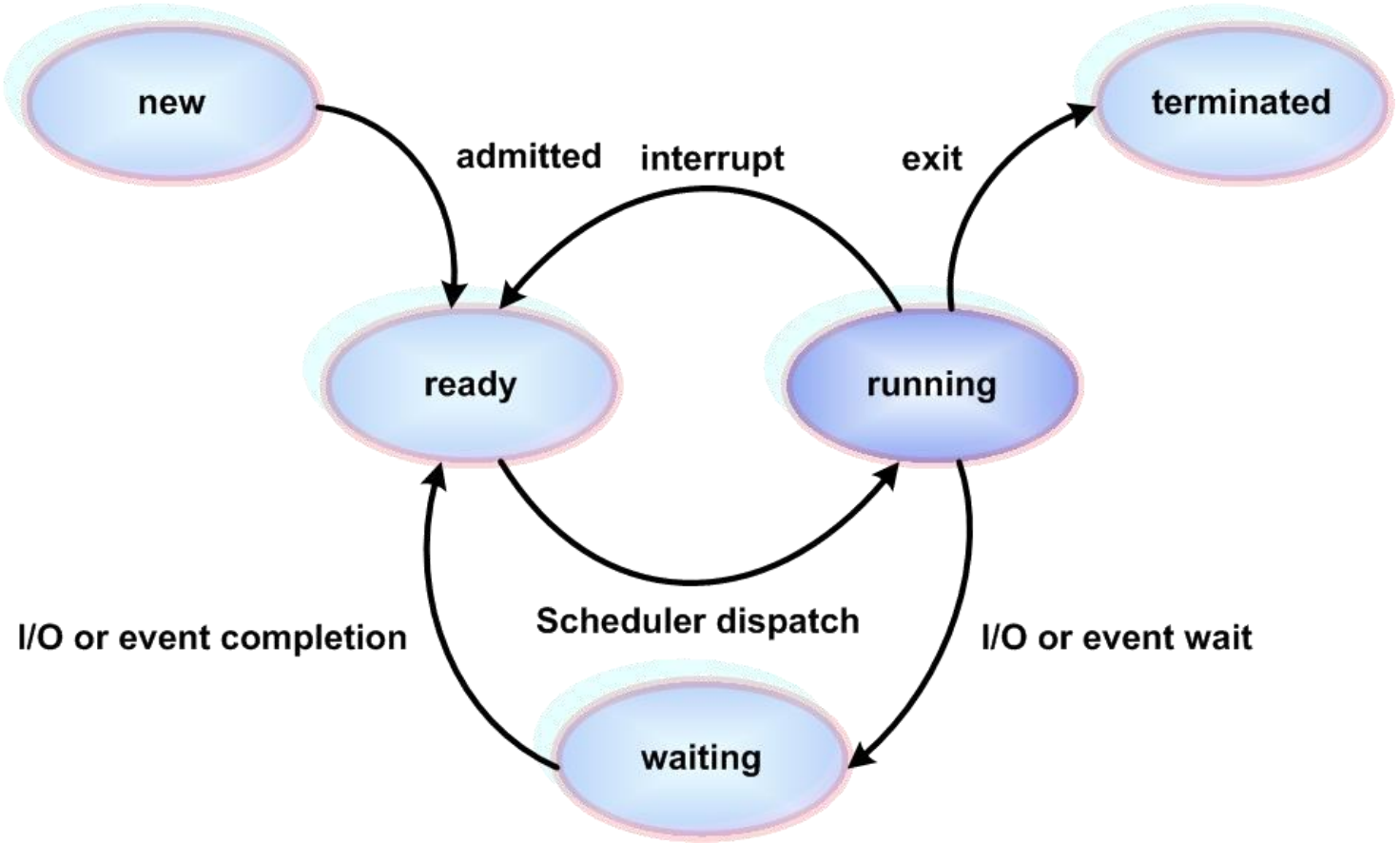
Kompleksitas Multitasking

- Selain permasalahan diatas, penerapan konkurensi juga menimbulkan kompleksitas baru:
  - Mengawasi berbagai proses aktif mengalokasikan resource melalui PCB (Process Control Block)
  - Melakukan penjadualan
  - Alokasi/dealokasi berbagai sumber daya
  - Melindungi sumber daya yang sedang digunakan
  - Menjaga aliran data ke sumber daya secara konstan
  - Komunikasi antar proses

Status Proses

- **Status Proses** bisa diartikan sebagai keadaan aktivitas yang sedang berlangsung dari suatu proses.
- **Dlm siklus hidup suatu proses, status proses tersebut berubah-ubah:**
  - ***new***: Proses sedang diciptakan atau diinisialisasi.
  - ***running***: Intruksi dari proses sedang dieksekusi.
  - ***waiting***: Proses lagi menunggu suatu *event*
  - ***ready***: Proses siap diaktifkan (ke status *running*).
  - ***terminated***: Proses sudah selesai dieksekusi.
- ***Multi-programming OS*** menggilir eksekusi proses-proses secara bergantian (***interleaving)***, sehingga suatu proses mengalami beragam status dalam siklus hidupnya.

Siklus Proses



Status: NEW

- Proses yang berstatus *new*, masih dalam tahap inisiasi oleh prosedur atau rutin sistem operasi.
- Tahap inisiasi ini meliputi alokasi memori utama untuk proses, pengisian tabel proses, pembuatan struktur data kendali untuk menyimpan informasi dan status proses.
- Penciptaan(*creation*) proses dpt disebabkan:
  - Eksekusi *job batch* baru
  - *User logon* pada lingkungan interaktif
  - Tanggapan OS atas permintaan layanan
  - Proses menciptakan proses lain (child process)

Status: READY

- Proses yang siap untuk dieksekusi (berstatus ready) dimasukkan ke dalam antrian penjadualan penggunaan prosesor
- Secara berkala, ataupun dalam kondisi tertentu, sistem operasi akan melakukan penjadwalan prosesor, yaitu menghentikan eksekusi proses yang lagi berjalan, memilih salah satu dari proses-proses yang berada dalam antrian *ready* dan mengeksekusinya.
- Rutin sistem operasi yang bertugas untuk melakukan pemilihan ini disebut dengan *scheduler.*

Status: RUNNING

- Jika sebuah proses dalam antrian *ready* terpilih oleh *scheduler* untuk dijalankan, maka proses yang terpilih akan mulai dieksekusi dan berubah statusnya menjadi *running*. Proses yang berstatus *running* menguasai prosesor sepenuhnya.
- Proses yang berstatus *running* memiliki 3 kemungkinan peralihan ke status lainnya.
  - Jika proses sudah menyelesaikan aktivitasnya maka proses akan berubah statusnya menjadi *terminated*.
  - Jika jatah waktu, *time-slice*, yang dialokasikan untuk eksekusi proses sudah habis, maka proses akan dialihkan statusnya menjadi *ready*.
  - Jika proses membutuhkan pembacaan data dari piranti I/O, maka akan beralih ke status *blocked (waiting)*

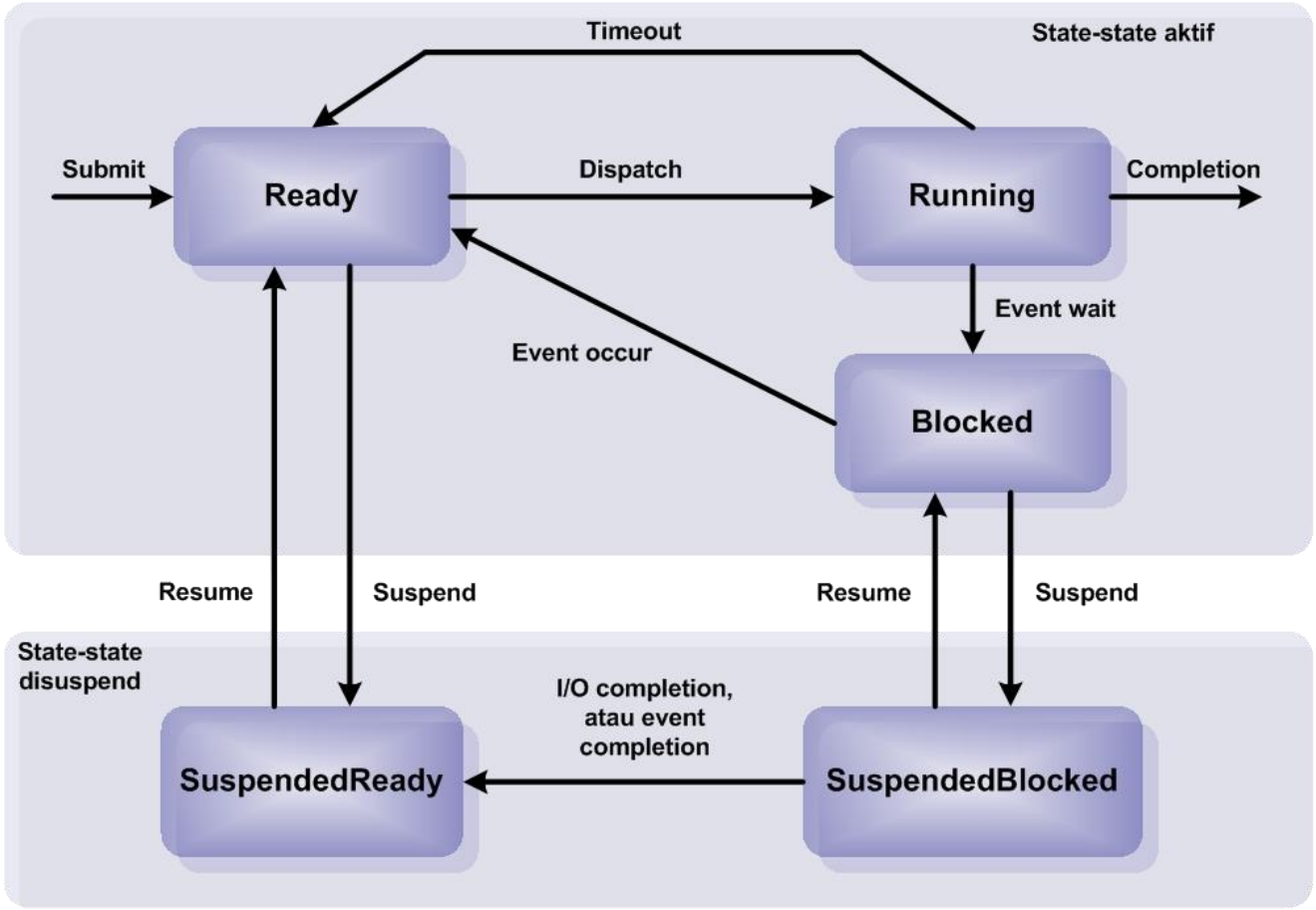
Status: BLOCKED (WAITING)

- Proses berstatus *running* beralih ke status *blocked (waiting)* karena membutuhkan pembacaan data dari piranti I/O.
- Untuk meningkatkan kinerja komputer, karena piranti I/O relatif sangat lambat dibanding kecepatan prosesor.
- Jika *event* yang ditunggu sudah terjadi atau operasi I/O yang ditunggu sudah selesai, maka rutin sistem operasi akan memindahkan proses yang berstatus *blocked* tersebut kembali ke antrian *ready*.

Status: EXIT (TERMINATED)

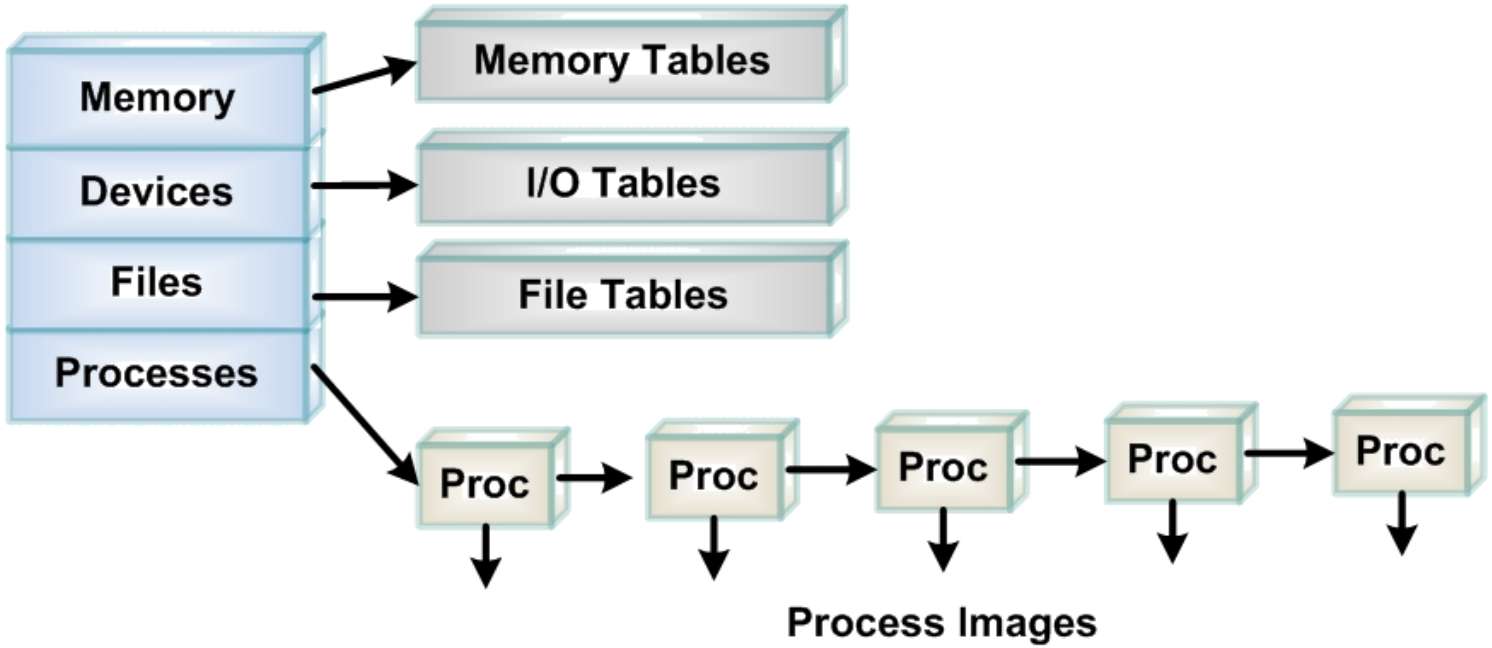
- Proses telah selesai dieksekusi, baik karena telah selesai secara normal maupun karena terjadi kesalahan.
- Penghentian(*termination*) proses dpt disebabkan:
  - Proses sudah selesai (dgn memanggil exit system call)
  - Batas waktu total telah terlewati
  - Kekurangan memori
  - Pelanggaran batas memori
  - Pelanggaran proteksi file atau sumber daya
  - Kesalahan aritmetika
  - Kegagalan I/O (tidak menemukan file atau gagal baca)
  - Intruksi yang tidak benar
  - Pengaksesan instruksi yang tidak diijinkan.
  - Kesalahan penggunaan data
  - Intervensi dari operasi atau sistem operasi
  - Proses induk (*parent*) berakhir
  - Atas permintaan proses *parent*

Model Proses 5 Status



Struktur Kendali Sistem Operasi

- Dalam melakukan pengelolaan alokasi sumber daya untuk proses-proses, sistem operasi membentuk dan memelihara struktur data yang menyimpan berbagai sumber daya yang ada.



Struktur Kendali

- Memory Table
  - Mencatat alokasi dan proteksi main memory dan virtual memory
- I/O Table
  - Mencatat status piranti I/O dan saluran (channel) sistem komputer
- File Table
  - Mencatat lokasi penyimpanan, status, hak akses dan atribut lainnya
- Process Table
  - Mencatat status proses

Image Process

- ***Image* proses** adalah keseluruhan ruang memori virtual yang dialokasi untuk suatu proses.
- ***Image* proses** terdiri atas elemen:
  - ***PCB (Process Control Block)***

    menyimpan atribut dari proses, dibutuhkan OS untuk mengontrol proses
  - ***User Program***

    menyimpan kode program yang dieksekusi
  - ***User Data***

    menyimpan data program (nilai variabel), data termodifikasi
  - ***System Stack***

    Menyimpan parameter dr *prosedur* atau *system call*

PCB

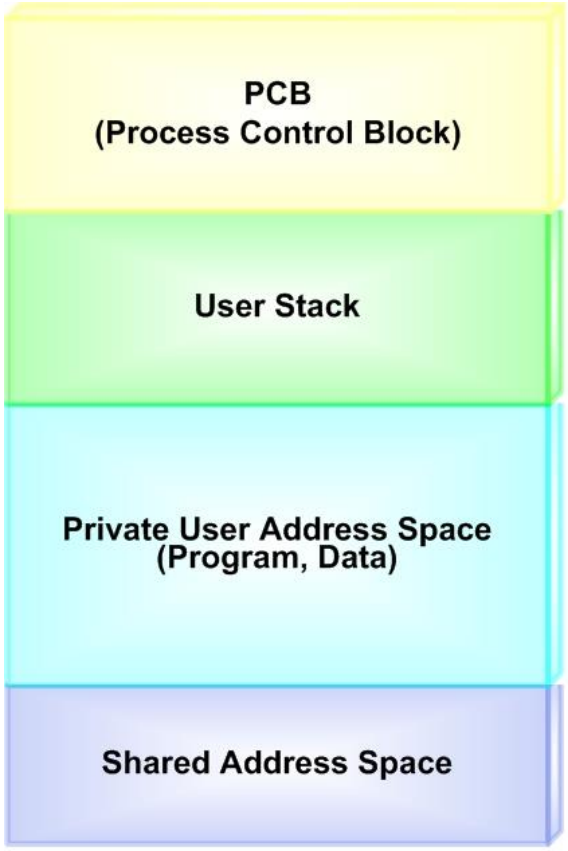
- ***PCB (Process Control Block)*** berisi deskripsi atribut dari proses.
- **Informasi dlm suatu *PCB* dpt dikategorikan:**
  - **Informasi Identitas proses**

    ID proses, ID parent, ID pemakai
  - **Informasi status CPU proses**

    Isi PC dan Register
  - **Informasi pengontrolan proses**

    Status proses, prioritas penjadwalan, *privileges* proses, penggunaan sumber daya (memori, I/O, cpu time, file),dll

Struktur Image Process dan PCB



Struktur Image proses

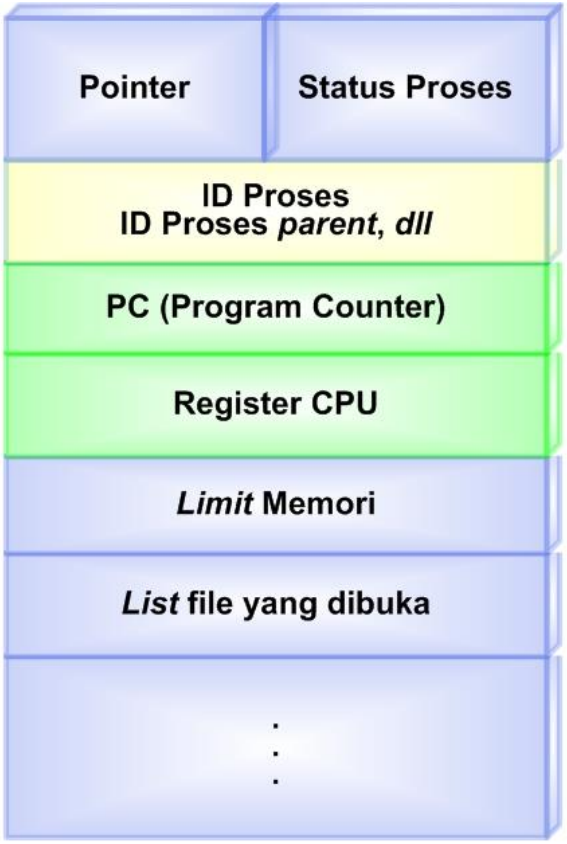


Diagram PCB