Bab 7

MANAJEMEN PERANGKAT MASUKAN/KELUARAN

7.1. Pendahuluan

Manajemen perangkat masukan/keluaran merupakan aspek perancangan sistem operasi terluas dan kompleks karena sangat beragamnya perangkat dan aplikasinya.

Beberapa fungsi manajemen input/ouput (I/O) :

a. Mengirim perintah ke perangkat I/O agar menyediakan layanan.

b. Menangani interupsi perangkat I/O.

c. Menangani kesalahan perangkat I/O.

d. Menyediakan interface ke pemakai.

7.2. Klasifikasi perangkat I/O

Perangkat I/O dapat dikelompokkan berdasarkan :

a. Sifat aliran datanya, yang terbagi atas :

a.1 Perangkat berorientasi blok.

Yaitu menyimpan, menerima, dan mengirim informasi sebagai blok-blok berukuran tetap yang berukuran 128 sampai 1024 byte dan memiliki alamat tersendiri, sehingga memungkinkan membaca atau menulis blok-blok secara independen, yaitu dapat membaca atau menulis sembarang blok tanpa harus melewati blok-blok lain. Contoh : disk,tape,CD ROM, optical disk.

a.2 Perangkat berorientasi aliran karakter.

Yaitu perangkat yang menerima, dan mengirimkan aliran karakter tanpa membentuk suatu struktur blok. Contoh : terminal, line printer, pita kertas, kartu-kartu berlubang, interface jaringan, mouse.

b. Sasaran komunikasi, yang terbagi atas :.

b.1 Perangkat yang terbaca oleh manusia.

Perangkat yang digunakan untuk berkomunikasi dengan manusia.

Contoh : VDT (video display terminal) : monitor, keyboard, mouse.

b.2 Perangkat yang terbaca oleh mesin.

Perangkat yang digunakan untuk berkomunikasi dengan perangkat elektronik.

Contoh : Disk dan tape, sensor, controller.

b.3 Perangkat komunikasi.

Perangkat yang digunakan untuk komunikasi dengan perangkat jarak jauh.

Contoh : Modem.

Faktor-faktor yang membedakan antar perangkat :

- Kecepatan transmisi data (data rate).
- Jenis aplikasi yang digunakan.
- Tingkat kerumitan dalam pengendalian.
- Besarnya unit yang ditransfer.
- Representasi atau perwujudan data.
- Kondisi-kondisi kesalahan.

7.3. Teknik pemograman perangkat I/O

Terdapat 3 teknik, yaitu :

a. I/O terprogram atau polling system.

Ketika perangkat I/O menangani permintaan, perangkat men-set bit status di register status perangkat. Perangkat tidak memberitahu ke pemroses saat tugas telah selesai dilakukan sehingga pemroses harus selalu memeriksa register tersebut secara periodik dan melakukan tindakan berdasar status yang dibaca. Software pengendali perangkat (driver) dipemroses harus mentransfer data ke/dari pengendali. Driver mengekseksui perintah yang berkomunikasi dengan pengendali (adapter) di perangkat dan menunggui sampai operasi yang dilakukan perangkat selesai.

Driver berisi kumpulan instruksi :

a.1 Pengendalian.

Berfungsi mengaktifkan perangkat eksternal dan memberitahu yang perlu dilakukan. Contoh : unit tape magnetik diinstruksikan untuk kembali ke posisi awal, bergerak ke record berikut, dan sebagainya.

a.2 Pengujian.

Berfungsi memeriksa status perangkat keras berkaitan dengan perangkat I/O.

a.3 Pembacaan/penulisan

Berfungsi membaca/menulis untuk transfer data antara register pemroses dan perangkat eksternal.

Masalah utama I/O terprogram adalah pemroses diboroskan untuk menunggu dan menjagai operasi I/O. Diperlukan teknik lain untuk meningkatkan efisiensi pemroses.

b. I/O dikendalikan interupsi.

Teknik I/O dituntun interupsi mempunyai mekanisme kerja sebagai berikut :

- Pemroses memberi instruksi ke perangkat I/O kemudian melanjutkan melakukan pekerjaan lainnya.
- Perangkat I/O akan menginterupsi meminta layanan saat perangkat telah siap bertukar data dengan pemroses.
- Saat menerima interupsi perangkat keras (yang memberitahukan bahwa perangkat siap melakukan transfer), pemroses segera mengeksekusi transfer data.

Keunggulan :

- Pemroses tidak disibukkan menunggui dan menjaga perangkat I/O untuk memeriksa status perangkat.

Kelemahan :

- Rate transfer I/O dibatasi kecepatan menguji dan melayani operasi perangkat.
- Pemroses terikat ketat dalam mengelola transfer I/O. Sejumlah intruksi harus dieksekusi untuk tiap transfer I/O.

c. Dengan DMA (direct memory access).

DMA berfungsi membebaskan pemroses menunggui transfer data yang dilakukan perangkat I/O. Saat pemroses ingin membaca atau menulis data, pemroses memerintahkan DMA controller dengan mengirim informasi berikut :

- Perintah penulisan/pembacaan.
- Alamat perangkat I/O.
- Awal lokasi memori yang ditulis/dibaca.
- Jumlah word (byte) yang ditulis/dibaca.

Setelah mengirim informasi-informasi itu ke DMA controller, pemroses dapat melanjutkan kerja lain. Pemroses mendelegasikan operasi I/O ke DMA. DMA mentransfer seluruh data yang diminta ke/dari memori secara langsung tanpa melewati pemroses. Ketika transfer data selesai, DMA mengirim sinyal interupsi ke pemroses. Sehingga pemroses hanya dilibatkan pada awal dan akhir transfer data. Operasi transfer antara perangkat dan memori utama dilakukan sepenuhnya oleh DMA lepas dari pemroses dan hanya melakukan interupsi bila operasi telah selesai.

Keunggulan :

- Penghematan waktu pemroses.
- Peningkatan kinerja I/O.

7.4. Evolusi fungsi perangkat I/O

Sistem komputer mengalami peningkatan kompleksitas dan kecanggihan komponen-komponennya, yang sangat tampak pada fungsi-fungsi I/O sebagai berikut :

a. Pemroses mengendalikan perangkat I/O secara langsung.

Masih digunakan sampai saat ini untuk perangkat sederhana yang dikendalikan mikroprosessor sehingga menjadi perangkat berintelijen (inteligent device).

b. Pemroses dilengkapi pengendali I/O (I/O controller).

Pemroses menggunakan I/O terpogram tanpa interupsi, sehingga tak perlu memperhatikan rincian-rincian spesifik antarmuka perangkat.

c. Perangkat dilengkapi fasilitas interupsi.

Pemroses tidak perlu menghabiskan waktu menunggu selesainya operasi I/O, sehingga meningkatkan efisiensi pemroses.

d. I/O controller mengendalikan memori secara langsung lewat DMA.

Pengendali dapat memindahkan blok data ke/dari memori tanpa melibatkan pemroses kecuali diawal dan akhir transfer.

e. Pengendali I/O menjadi pemroses terpisah.

Pemroses pusat mengendalikan.memerintahkan pemroses khusus I/O untuk mengeksekusi program I/O di memori utama. Pemroses I/O mengambil dan mengeksekusi intruksi-intruksi ini tanpa intervensi pemroses pusat.

Dimungkinkan pemroses pusat menspesifikasikan barisan aktivitas I/O dan hanya diinterupsi ketika seluruh barisan intruksi diselesaikan.

f. Pengendali I/O mempunyai memori lokal sendiri.

Perangkat I/O dapat dikendalikan dengan keterlibatan pemroses pusat yang minimum.

Arsitektur ini untuk pengendalian komunikasi dengan terminal-terminal interaktif. Pemroses I/O mengambil alih kebanyakan tugas yang melibatkan pengendalian terminal.

Evolusi bertujuan meminimalkan keterlibatan pemroses pusat, sehingga pemroses tidak disibukkan dengan tugas I/O dan dapat meningkatkan kinerja sistem.

7.5. Prinsip manajemen perangkat I/O

Terdapat dua sasaran perancangan I/O, yaitu :

a. Efisiensi.

Aspek penting karena operasi I/O sering menimbulkan bottleneck.

b. Generalitas (device independence).

Manajemen perangkat I/O selain berkaitan dengan simplisitas dan bebas kesalahan, juga menangani perangkat secara seragam baik dari cara proses memandang maupun cara sistem operasi mengelola perangkat dan operasi I/O.

Software diorganisasikan berlapis. Lapisan bawah berurusan menyembunyikan kerumitan perangkat keras untuk lapisan-lapisan lebih atas. Lapisan lebih atas berurusan memberi antar muka yang bagus, bersih, nyaman dan seragam ke pemakai.

Masalah-masalah manajemen I/O adalah :

a. Penamaan yang seragam (uniform naming).
   Nama berkas atau perangkat adalah string atau integer, tidak bergantung pada perangkat sama sekali.
b. Penanganan kesalahan (error handling).
   Umumnya penanganan kesalahan ditangani sedekat mungkin dengan perangkat keras.
c. Transfer sinkron vs asinkron.
   Kebanyakan I/O adalah asinkron. Pemroses mulai transfer dan mengabaikan untuk melakukan kerja lain sampai interupsi tiba. Program pemakai sangat lebih mudah ditulis jika operasi I/O berorientasi blok. Setelah perintah read, program kemudian ditunda secara otomatis sampai data tersedia di buffer.
d. Sharable vs dedicated.
   Beberapa perangk dapat dipakai bersama seperti disk, tapi ada juga perangkat yang hanya satu pemakai yang dibolehkan memakai pada satu saat.
   Contoh : printer.

7.6. Hirarki manajemen perangkat I/O

Hirarki manajemen perangkat I/O :

a. Interrupt handler.
   Interupsi harus disembunyikan agar tidak terlihat rutin berikutnya.
   Device driver di blocked saat perintah I/O diberikan dan menunggu interupsi. Ketika interupsi terjadi, prosedur penanganan interupsi bekerja agar device driver keluar dari state blocked.

b. Device drivers.
   Semua kode bergantung perangkat ditempatkan di device driver. Tiap device driver menangani satu tipe (kelas) perangkat dan bertugas menerima permintaan

abstrak perangkat lunak device independent diatasnya dan melakukan layanan permintaan.

Mekanisme kerja device driver :

- Menerjemahkan perintah abstrak menjadi perintah konkret.
- Setelah ditentukan perintah yang harus diberikan ke pengendali, device driver mulai menulis ke register-register pengendali perangkat.
- Setelah operasi selesai dilakukan perangkat, device driver memeriksa status kesalahan yang terjadi.
- Jika berjalan baik, device driver melewatkan data ke perangkat lunak device independent.
- Kemudian device driver melaporkan status operasinya ke pemanggil.

c. Perangkat lunak device independent.

Bertujuan membentuk fungsi-fungsi I/O yang berlaku untuk semua perangkat dan memberi antarmuka seragam ke perangkat lunak tingkat pemakai.

Fungsi-fungsi lain yang dilakukan :

- Sebagai interface seragam untuk seluruh device driver.
- Penamaan perangkat.
- Proteksi perangkat.
- Memberi ukuran blok perangkat agar bersifat device independent.
- Melakukan buffering.
- Alokasi penyimpanan pada block devices.
- Alokasi dan pelepasan dedicated devices.
- Pelaporan kesalahan.

d. Perangkat lunak level pemakai.

Kebanyakan perangkat lunak I/O terdapat di sistem operasi. Satu bagian kecil berisi pustaka-pustaka yang dikaitkan pada program pemakai dan berjalan diluar kernel. System calls I/O umumnya dibuat sebagai prosedur-prosedur pustaka. Kumpulan prosedur pustaka I/O merupakan bagian sistem I/O. Tidak semua perangkat lunak I/O level pemakai berupa prosedur-prosedur pustaka. Kategori penting adalah sistem spooling. Spooling adalah cara khusus berurusan dengan perangkat I/O yang harus didedikasikan pada sistem multiprogramming.

7.7. Buffering I/O

Buffering adalah melembutkan lonjakan-lonjakan kebutuhan pengaksesan I/O, sehingga meningkatkan efisiensi dan kinerja sistem operasi.

Terdapat beragam cara buffering, antar lain :

a. Single buffering.

Merupakan teknik paling sederhana. Ketika proses memberi perintah untuk perangkat I/O, sistem operasi menyediakan buffer memori utama sistem untuk operasi.

Untuk perangkat berorientasi blok.

Transfer masukan dibuat ke buffer sistem. Ketika transfer selesai, proses memindahkan blok ke ruang pemakai dan segera meminta blok lain.

Teknik ini disebut reading ahead atau anticipated input. Teknik ini dilakukan dengan harapan blok akan segera diperlukan. Untuk banyak tipe komputasi, asumsi ini berlaku. Hanya di akhir pemrosesan maka blok yang dibaca tidak diperlukan.

Keunggulan :

Pendekatan in umumnya meningkatkan kecepatan dibanding tanpa buffering.

Proses pemakai dapat memproses blok data sementara blok berikutnya sedang dibaca. Sistem operasi dapat menswap keluar proses karena operasi masukan berada di memori sistem bukan memori proses pemakai.

Kelemahan :

- Merumitkan sistem operasi karena harus mencatat pemberian buffer-buffer sistem ke proses pemakai.
- Logika swapping juga dipengaruhi. Jika operasi I/O melibatkan disk untuk swapping, maka membuat antrian penulisan ke disk yang sama yang digunakan untuk swap out proses. Untuk menswap proses dan melepas memori utama tidak dapat dimulai sampai operasi I/O selesai, dimana waktu swapping ke disk tidak bagus untuk dilaksanakan.

Buffering keluaran serupa buffering masukan. Ketika data transmisi, data lebih dulu dikopi dari ruang pemakai ke buffer sistem. Proses pengirim menjadi bebas untuk melanjutkan eksekusi berikutnya atau di swap ke disk jika perlu.

Untuk perangkat berorientasi aliran karakter.

Single buffering dapat diterapkan dengan dua mode, yaitu :

- o Mode line at a time.

  Cocok untuk terminal mode gulung (scroll terminal atau dumb terminal).

  Masukan pemakai adalah satu baris per waktu dengan enter menandai akhir baris. Keluaran terminal juga serupa, yaitu satu baris per waktu.

  Contoh mode ini adalah printer.

  Buffer digunakan untuk menyimpan satu baris tunggal. Proses pemakai ditunda selama masukan, menunggu kedatangan satu baris seluruhnya.

  Untuk keluaran, proses pemakai menempatkan satu baris keluaran pada buffer dan melanjutkan pemrosesan. Proses tidak perlu suspend kecuali bila baris kedua dikirim sebelum buffer dikosongkan.

- o Mode byte at a time.

  Operasi ini cocok untuk terminal mode form, dimana tiap ketikan adalah penting dan untuk peripheral lain seperti sensor dan pengendali.

b. Double buffering.

Peningkatan dapat dibuat dengan dua buffer sistem.Proses dapat ditransfer ke/dari satu buffer sementara sistem operasi mengosongkan (atau mengisi) buffer lain. Teknik ini disebut double buffering atau buffer swapping.

Double buffering menjamin proses tidak menunggu operasi I/O. Peningkatan ini harus dibayar dengan peningkatan kompleksitas. Untuk berorientasi aliran karakter, double buffering mempunyai 2 mode alternatif, yaitu :

- o Mode line at a time.

  Proses pemakai tidak perlu ditunda untuk I/O kecuali proses secepatnya mengosongkan buffer ganda.

- o Mode byte at a time.

  Buffer ganda tidak memberi keunggulan berarti atas buffer tunggal.

  Double buffering mengikuti model producer-consumer.

c. Circular buffering.

Seharusnya melembutkan aliran data antara perangkat I/O dan proses. Jika kinerja proses tertentu menjadi fokus kita, maka kita ingin agar operasi I/O mengikuti proses. Double buffering tidak mencukupi jika proses melakukan

operasi I/O yang berturutan dengan cepat. Masalah sering dapat dihindari denga menggunakan lebih dari dua buffer.

Ketika lebih dari dua buffer yang digunakan, kumpulan buffer itu sendiri diacu sebagai circulat buffer. Tiap buffer individu adalah satu unit di circular buffer.

Perangkat keras dan parameter kinerja disk

Disk diorganisasikan menjadi silinder-silinder dengan tiap permukaan terdapat head yang ditumpuk secara vertical. Track terbagi menjadi sektor-sektor.

Waktu yang dibutuhkan untuk membaca dan menulis disk dipengaruhi oleh :

o Waktu pencarian (seek time).

Merupakan faktor yang dominan. Waktu yang diperlukan untuk sampai ke posisi track yang dituju, yaitu : $S = S_c + di$, dimana :

S_c : adalah waktu penyalaan awal (initial startup time).

d : adalah waktu yang bergerak antar-antar track.

i : adalah jarak yang ditempuh (dalam ukuran ruang antar track).

Untuk track terdekat, $S_1 = S_c + d$ lebih kecil dibanding waktu yang diperlukan untuk satu putaran.Untuk memudahkan perhitungan maka dipakai s rata-rata,yaitu :

$$S = \sum_{i=1}^{j-1} S_i \, pdi,$$

S_i : adalah waktu tempuh untuk jarak ke-1.

Pdi : adalah probabilitas menempuh jarak ke-1.

Seek time rata-rata biasanya diinformasikan oleh pabrik pembuat.

o Waktu rotasi (rotational latency).

Waktu yang diperlukan mekanisme akses mencapai blok yang diinginkan.

Rumus untuk mendapatkan r adalah :

R=1/2*((60*1000)/rpm).

Rpm atau jumlah putaran permenit, biasa diinformasikan oleh pabrik pembuat.

o Waktu transfer (t).

Tergantung pada kecepatan rotasi dan kepadatan rekaman. Transfer rate (t) adalah kecepatan transfer data sesaat, data ini diberikan oleh pembuat. Maka dapat dihitung :

> Waktu transfer per rekord (TR, record transfer time).

TR (waktu untuk transfer rekord dengan panjang rekord, R), yaitu :

TR=R/t.

> Waktu transfer per blok (btt).

Bit (block transfer time,waktu yang diperlukan untuk transfer 1 blok),

yaitu : btt=B/t.

> Bulk transfer time (t').

Didalam kasus pembacaan/penulisan secara sekuens besar maka harus melewati gap dan daerah-daerah bukan data. Pada akhir tiap silinder, seek akan terjadi dan selama seek time, tidak ada data yang ditransfer.

Untuk keperluan didefinisikan bulk transfer time (t'), yaitu :

t'=(R)/(((R+W)/t)+s')

dimana :

R : adalah ukuran rekord.

W : adalah ruang yang disiakan.

s' : adalah seek time untuk sekuen.

t : adalah transfer mode.

Algoritma penjadwalan disk

Pada sistem multiprogramming, banyak proses yang melakukan permintaan membaca dan menulis rekord-rekord disk. Proses-proses membuat permintaan-permintaan lebih cepat dibanding yang dapat dilayani disk, membentuk antrian permintaan layanan disk. Diperlukan penjadwalan disk agar memperoleh kinerja yang optimal.

Terdapat dua tipe penjadwalan disk, yaitu :

1. Optimasi seek.
2. Optimasi rotasi (rotational latency).

Karena waktu seek lebih tinggi satu orde dibanding waktu rotasi, maka kebanyakan algoritma penjadwalan berkonsentrasi meminimumkan seek kumpulan atau antrian permintaan layanan disk. Meminimumkan latency biasanya berdampak kecil pada kinerja seluruh sistem.

Penjadwalan disk melibatkan pemeriksaan terhadap permintaan-permintaan yang belum dilayani untuk menentukan cara paling efisien melayani permintaan-permintaan. Penjadwal disk memeriksa hubungan posisi diantara permintaan-permintaan. Antrian permintaan disusun kembali sehingga permintaan-permintaan akan dilayani dengan pergerakan mekanis minimum.

Beberapa kriteria penjadwalan disk, yaitu :

- Throughput, yaitu berusaha memaksimumkan.
- Waktu tanggap rata-rata, nilai ini diusahakan minimum.
- Variansi waktu tanggap, diusahakan minimum.

Beberapa algoritma penjadwalan disk, antara lain :

- First come first serve (FCFS).

  Disk driver melayani satu permintaan sesuai urutan kedatangannya, merupakan metode yang adil. Saat rate permintaan sangat berat, FCFS dapat menghasilkan waktu tunggu sangat panjang. Dengan FCFS, sangat sedikit usaha optimasi waktu seek. FCFS dapat menyebabkan banyak waktu untuk seek silinder yang paling dalam ke silinder paling luar.

  Ketika permintaan-permintaan terdistribusi seragam pada permukaan-permukaan disk, penjadwalan FCFS menghasilkan pola seek yang acak. FCFS mengabaikan keterhubungan posisi diantara permintaan-permintaan yang menunggu di antrian. FCFS tidak membuat upaya optimasi pola seek. FCFS dapat diterima ketika beban disk masih ringan, tetapi begitu beban tumbuh cenderung menjenuhi perangkat dan menyebabkan waktu tanggap membesar.

- Shortest seek first (SSF).

  Algoritma ini melayani permintaan seek track terdekat dari track dimana head berada.

  Kekurangan : lengan disk akan berkutat ditengah disk. Permintaan di daerah ekstrim (pinggir) akan menunggu sampai fluktuasi statistik menyebabkan tidak

ada permintaan track-track tengah. Terdapat konflik antara meminimalkan waktu tanggao dengan fairness (adil).

- Elevator (SCAN).

  Yaitu head bergerak searah sampai tidak ada permintaan ke arah itu, kemudian berbalik arah. Diperlukan bit tambahan untuk mencatat arah gerak head. Kebaikan : batas atas jumlah gerak adalah tetap yaitu dua kali jumlah silinder.

- Elevator dimodifikasi (C-SCAN).

  Lengan head hanya bergerak searah, setiap kali mencapai silinder tertinggi, maka head akan bergerak ke silinder terendah dan dilanjutkan terus head bergerak searah. Ada kontroller yang dapat mengetahui pada track mana ia berada, dengan ini dapat dibuat optimasi untuk mencari sektor yang ada pada track tersebut.

- N-step scan.

  Lengan disk bergerak maju mundur seperti algoritma SCAN, tapi dengan semua permintaan yang tiba selama menyapu dalam satu arah dikumpulkan dulu dan disusun kembali agar layanan optimal selama penyapuan balik.

- Exchenbach scheme.

  Pergerakan lengan disk sirkular seperti C-SCAN, tapi dengan beberapa kekecualian penting setiap silinder dilayani tepat satu track informasi baik terdapat permintaan atau tidak untuk silinder itu. Permintaan-permintaan disusun untuk layanan dalam silinder itu untuk mendapatkan keunggulan posisi secara rotasi (agar dapat diterapkan optimasi rotasi), tapi jika terdapat dua permintaan dengan sektor-sektor yang overlap dalam satu silinder, hanya satu permintaan yang dilayani pada satu kesempatan.

Penanganan masalah operasi disk

Beberapa tipe kesalahan saat operasi disk dikategorikan sebagai berikut :

o Programming error.

Kesalahan disebabkan programming. Driver memerintahkan mencari track, membaca sektor, menggunakan head atau mentransfer ke atau dari memori

yang tak ada. Biasanya tiap controller memeriksa parameter sehingga tidak melakukan operasi yang tak valid. Kesalahan ini seharusnya tidak pernah ada.

o Transient checksum error.

Kesalahan disebabkan adanya debu diantara head dengan permukaan disk.

Untuk mengeliminasi kesalahan ini maka dilakukan pengulangan operasi pada disk.

o Permanent checksum error.

Kesalahan disebabkan kerusakan disk.

o Seek error.

Kesalahan ini ditanggulangi dengan mengkalibrasi disk supaya berfungsi kembali.

o Controller error.

Kesalahan ini ditanggulangi dengan menukar pengendali yang salah dengan pengendali yang baru.

o Track at time caching.

Kontroller mempunyai memori untuk menyimpan informasi track dimana ia berada, permintaan pembacaan blok track dilakukan tanpa pergerakan mekanik.

b. Clock

Perangkat keras clock.

Komputer dilengkapi dengan RTC (real time clock). Tipe perangkat clock, terdiri dari :

- Clock yang ditimbulkan impulse tegangan listrik.
  Clock in menginterupsi 50-60 interupt tiap detik sesuai dengan frekuensi listrik.
- Programmable interval timer (PIT).
  Clock ini terdiri dari crystal oscilator, counter, dan holding register.
  Dua keunggulan PIT, yaitu :
  - Mempunyai akurasi tinggi.
  - Frekuensi interupsi dapat diatur secara perangkat lunak.
    Dengan crystal oscilator 2 MHz, menggunakan 16 bit holding register, interupsi yang terjadi dapat diatur antara 1 ms sampai 65.536 ms.

PIT biasa digunakan sebagai :

- Waktu sistem.

- Pembangkit band rate.
- Penghitung kejadian.
- Pembangkit musik.
- Dan diberagam aplikasi yang memerlukan pewaktuan.

Ketika digunakan untuk pewaktuan PIT menghasilkan interupsi secara periodik. PIT bekerja dengan menghitung pulsa eksternal yang diberikan crystal oscilator. Keluaran PIT berupa pulsa yang diteruskan secara langsung ke IRQm(Interupt Request) sehingga menimbulkan interupsi ke pemroses. Periode waktu antara dua interupsi timer berturutan dapat diprogram dengan memasukkan nilai ke holding register.

Interval interupsi mempunyai rumus sebagai berikut :
Interval = (periode clock) x (nilai holding register).

Contoh :
Dikehendaki interval pewaktuan setiap 10 ms.
Frekuensi crystal oscilator adalah 2 MHz.

Berapa nilai yang harus dimasukkan ke holding register ?
Perhitungan :
Periode clock = 1/(2x106) = 0.5 x 10-6 = 0,5mms.
Nilai yang harus diberikan ke holding register = (10x10-3)/(0.5x10-6)=20x103.
Agar PIT menimbulkan interupsi dengan waktu interval 10 ms, maka holding register diset dengan nilai 20.000.

Metode pemrograman PIT.
Terdapat dua mode pemograman PIT, yaitu :
1. One shot mode.
   Setiap kali PIT diinisialisasi maka dikopikan nilai holding register ke counter.Counter diturunkan setiap terjadi pulsa crystal oscilator.
   Ketika counter bernilai 0, PIT membuat interupsi ke pemroses dan berhenti. PIT menunggu diinisialisasi secara eksplisit oleh perangkat lunak. Mode ini hanya untuk menghasilkan satu kejadian tunggal, diperlukan ketika clock diaktifkan berdasarkan kejadian.

2. Square wave mode.

Sesudah counter mencapai 0 maka menyebabkan interupsi ke pemroses.

Holding register dikopikan secara otomatis ke counter dan seluruh proses diulangi lagi sampai tak berhingga. Periode ini disebut clock ticks. Mode ini untuk menghasilkan kejadian-kejadian interupsi timer secara periodik, dilakukan secara otomatis tanpa melibatkan pemroses (perangkat lunak untuk inisialisasi kembali). Biasanya chip berisi dua atau tiga PIT independen dan mempunyai banyak option pemrograman (seperti menghitung keatas, pematian interupsi, dan sebagainya).

Perangkat lunak clock

Beberapa fungsi clock disistem operasi, antara lain :

1. Mengelola waktu dan tanggal (waktu nyata).

Tekniknya adalah counter dinaikkan setiap terjadi clock tick.

Teknik ini bermasalah karena keterbatasan jumlah bit counter.

Counter berukuran 32 bit akan overflow setelah 2 tahun bila clock ratenya bernilai 60Hz, solusinya adalah :

- Menggunakan counter 64 bit.
- Waktu dihitung dalam detik bukan dalam clock tick.
- Waktu dihitung relatif dengan saat komputer dihidupkan.

2. Mencegah proses berjalan lebih dari waktu yang ditetapkan.

Setiap kali proses dimulai, penjadwal inisialisasi counter dalam hitungan clock ticks. Setiap kali terjadi clock ticks, counter diturunkan. Saat counter mencapai 0 maka penjadwal mengalihkan pemroses ke proses lain.

3. Menghitung penggunaan pemroses (CPU).

Bila dikehendaki penghitungan dengan akurasi tinggi maka dilakukan dengan menggunakan timer kedua. Timer kedua terpisah dari timer sistem utama. Begitu proses dimulai, timer diaktifkan, saat proses berhenti maka timer dibaca. Timer menunjukkan lama waktu yang telah digunakan proses. Akurasi rendah dapat diperoleh dengan mengelola pointer ke tabel proses dan counter global.

4. Menangani system call alarm yang dibuat proses pemakai.
   Mensimulasi banyak clock dengan membuat senarai semua permintaan clock, terurut berdasar waktu. Isinya adalah jumlah clock ticks setelah signal proses sebelumnya.

5. Mengerjakan profiling, monitoring dan pengumpulan statistik.
   Untuk membuat data statistik kegiatan komputer.

c. RAM Disk.

Adalah perangkat disk yang disimulasikan pada memori akses acak (RAM).

RAM disk sepenuhnya mengeliminasi waktu tunda yang disebabkan pergerakan mekanis dalam seek dan rotasi. Kegunaannya untuk aplikasi yang memerlukan kinerja disk yang tinggi. Perangkat blok mempunyai dua perintah, yaitu membaca dan menulis blok. Normalnya blok-blok disimpan di disk berputar yang memerlukan mekanisme fisik.

Gagasannya adalah meniru perangkat dengan mengalokasikan terlebih satu bagian memori utama untuk menyimpan blok-blok data.

Keunggulan :

Berkecepatan tinggi karena pengaksesan sesaat (tidak ada waktu tunda seek dan rotational latency), sangat baik untuk menyimpan program atau data yang sering diakses. Memori utama dibagi menjadi n blok berukuran sama, bergantung banyak memori yang dialokasikan. Ketika driver untuk RAM disk menerima perintah membaca atau menulis suatu blok, driver tinggal menghitung dimana lokasi memori tempat blok berada kemudian membaca atau menuliskannya.