Sistem Operasi

Catatan Kuliah #7

Alauddin Maulana Hirzan, M. Kom
0607069401

Manajemen CPU dan *Scheduling*

Manajemen CPU dan *Scheduling*

Apa itu **Manajemen CPU**?

Sistem Operasi melakukan manajemen CPU untuk memastikan proses yang berjalan mendapatkan kesempatan yang sama. Berikut ini beberapa cara OS mengelola CPU:

1. **Task Scheduling** - Penjadwalan tugas
2. **Process Management** - Manajemen proses
3. **Interruption Management** - Penanganan interupsi
4. **Memory Management** - Manajemen memori
5. **Power Management** - Manajemen daya

Manajemen CPU dan *Scheduling*

Manajemen CPU - Task Scheduling #1

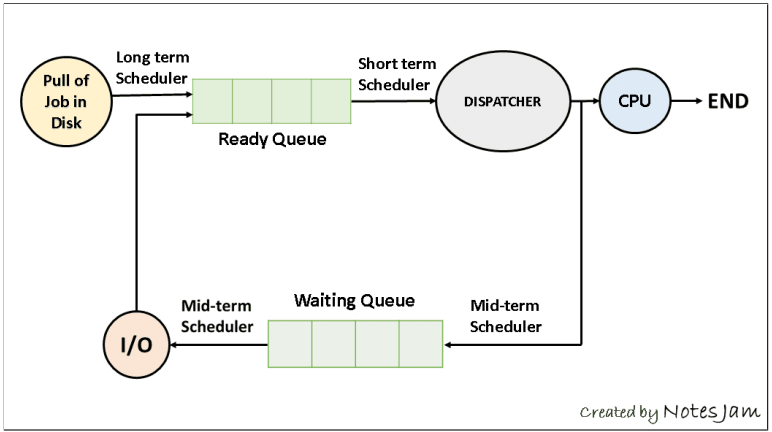
Penjadwalan tugas adalah proses menentukan tugas mana yang harus dijalankan oleh CPU (Central Processing Unit) sistem komputer pada waktu tertentu. Sistem operasi (OS) menggunakan algoritme penjadwalan untuk memprioritaskan tugas berdasarkan berbagai kriteria, seperti tingkat urgensi, jumlah sumber daya yang dibutuhkan, dan tingkat prioritas tugas.

Ada beberapa jenis algoritma penjadwalan yang digunakan oleh OS:

- First-come, first-served (FCFS)
- Shortest job first (SJF)
- Priority scheduling
- Round-robin scheduling
- Multi-level feedback queue scheduling

Manajemen CPU dan *Scheduling*

Ilustrasi Scheduling



Manajemen CPU dan *Scheduling*

Manajemen CPU - Task Scheduling #2

Burst Time mengacu pada jumlah waktu yang diperlukan tugas untuk menyelesaikan eksekusinya setelah mulai berjalan di CPU. Waktu ini juga dikenal sebagai waktu eksekusi atau waktu CPU.

Arrival Time mengacu pada waktu di mana sebuah tugas memasuki antrian tugas dan siap untuk dieksekusi. Waktu kedatangan tugas juga dapat bervariasi tergantung pada faktor-faktor seperti sifat tugas, tingkat prioritas tugas, dan sumber daya yang tersedia.

Manajemen CPU dan *Scheduling*

Manajemen CPU - Task Scheduling - **First-come, first-served (FCFS)**

First-come, first-served (FCFS) : Algoritma ini menjadwalkan tugas sesuai urutan penerimaannya, dengan tugas pertama yang diterima dieksekusi terlebih dahulu. Metode ini sederhana, tetapi dapat menyebabkan waktu tunggu yang lama untuk tugas dengan prioritas tinggi jika ada banyak tugas dengan prioritas lebih rendah yang mengantri.

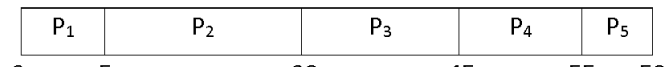
Informasi

Algoritma **First Come First Serve** dapat menyebabkan terjadinya **Starvation** jika ada tugas prioritas tinggi yang memiliki durasi proses yang panjang

Manajemen CPU dan *Scheduling*

Ilustrasi **First Come First Server**

Process	Burst Time(ms)
P_1	5
P_2	24
P_3	16
P_4	10
P_5	3



Manajemen CPU dan *Scheduling*

Manajemen CPU - Task Scheduling - **Shortest Job First (SJF)** #1

Algoritma ini memprioritaskan tugas berdasarkan waktu eksekusi yang diharapkan, dengan tugas yang lebih pendek dieksekusi terlebih dahulu. Metode ini dapat menghasilkan waktu tunggu yang lebih singkat untuk tugas-tugas dengan prioritas tinggi, tetapi memerlukan pengetahuan tentang waktu eksekusi yang diharapkan untuk setiap tugas, yang mungkin tidak selalu tersedia.

Informasi

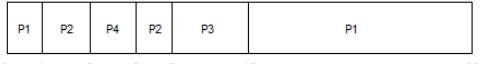
Sistem Operasi harus bisa memprediksi waktu yang dibutuhkan oleh proses untuk berjalan yang dalam kenyataannya hal itu tidak dapat diketahui karena bersifat **estimate**.

Manajemen CPU dan *Scheduling*

Ilustrasi **Shortest Job FIrst**

PROCESS	BURST TIME	ARRIVAL TIME
P1	21	0
P2	3	1
P3	6	2
P4	2	3

The GANTT chart for Preemptive Shortest Job First Scheduling will be,



The average waiting time will be, ((5-3) + (6-2) + (12-1))/4 = 4.25 ms

Manajemen CPU dan *Scheduling*

Manajemen CPU - Task Scheduling - **Shortest Job First (SJF)** #2

Dalam algoritma ini tugas dijadwalkan sebagai berikut:

1. Proses **P1** masuk pertama kali karena **Arrival Time** 0.
2. Pada saat **Arrival Time** 1, Proses **P2** masuk dan menghentikan Proses **P1** karena **Burst Time P2** < **Burst Time P1**
3. Ketika **Arrival Time** 2, Proses **P2** tetap berjalan dan mengabaikan Proses **P3** karena **Burst Time P2** < **Burst Time P3**
4. **Arrival Time** 3, Proses **P4** masuk dan menghentikan Proses **P2** karena **Burst Time P4** < **Burst Time P2**

Manajemen CPU dan *Scheduling*

Manajemen CPU - Task Scheduling - **Shortest Job First (SJF)** #3

5. Proses **P4** berjalan hingga selesai di **Arrival Time** 5. Sistem Operasi melakukan pengecekan Proses yang memiliki **Burst Time** terkecil.
6. Proses **P2** melanjutkan proses karena masih memiliki **Burst Time 1** $(3 - 2 = 1)$
7. Proses **P2** selesai menjalankan tugas dan berhenti di **Arrival Time** 6. Proses **P1** dan **P3** masih tersisa. Proses **P3** berjalankan karena **Burst Time P3** < **Burst Time P1**
8. Proses **P3** berjalan hingga selesai dan **Proses P1** berjalan hingga selesai

Waktu rata-rata pemrosesan : $((5 - 3) + (6 - 2) + (12 - 1))/4 = 4.25ms$

Manajemen CPU dan *Scheduling*

Manajemen CPU - Task Scheduling - **Priority Scheduling**

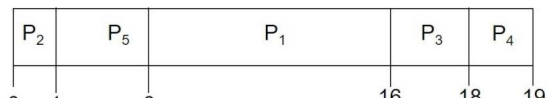
Algoritma ini menetapkan tingkat prioritas untuk setiap tugas, dengan tugas dengan prioritas lebih tinggi dieksekusi terlebih dahulu. Metode ini dapat memastikan bahwa tugas-tugas penting dieksekusi dengan cepat, tetapi dapat menyebabkan tugas-tugas dengan prioritas lebih rendah kekurangan sumber daya jika tugas-tugas dengan prioritas tinggi terus ditambahkan ke dalam antrean.

Manajemen CPU dan *Scheduling*

Ilustrasi **Priority Scheduling**

Process	Burst Time	Priority
P1	10	3
P2	1	1
P3	2	4
P4	1	5
P5	5	2

Gantt Chart



Manajemen CPU dan *Scheduling*

Manajemen CPU - Task Scheduling - **Round-Robin Scheduling**

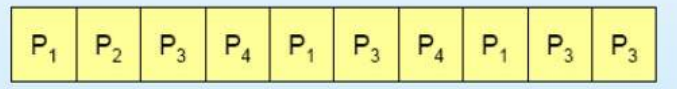
Algoritma ini mengalokasikan jumlah waktu CPU yang tetap untuk setiap tugas secara bergantian, dengan setiap tugas mendapatkan kesempatan untuk dieksekusi sebelum beralih ke tugas berikutnya. Metode ini dapat memastikan bahwa setiap tugas mendapatkan bagian waktu CPU yang adil, tetapi mungkin tidak cocok untuk tugas dengan tenggat waktu yang ketat.

Manajemen CPU dan *Scheduling*

Ilustrasi **Round-Robin Scheduling**

Process	Burst Time
P_1	53
P_2	17
P_3	68
P_4	24

The Gantt chart is:



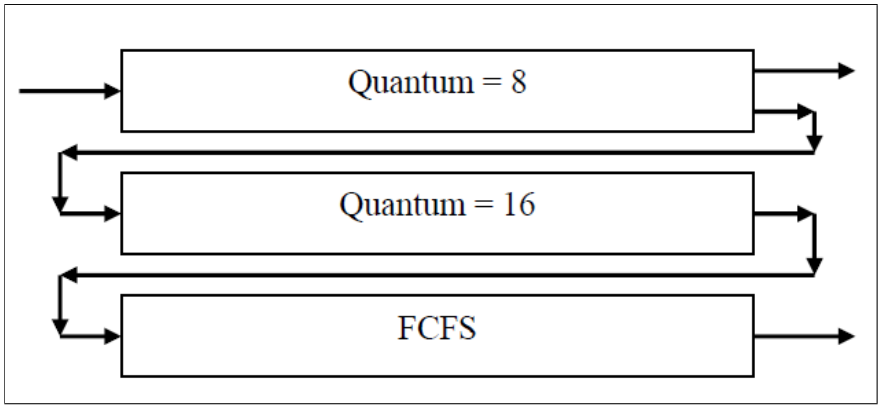
Manajemen CPU dan *Scheduling*

Manajemen CPU - Task Scheduling - **Multi-level feedback queue scheduling**

Algoritma ini menggabungkan beberapa metode penjadwalan, seperti penjadwalan prioritas dan penjadwalan round-robin, ke dalam satu sistem penjadwalan. Metode ini sangat fleksibel dan mudah beradaptasi dengan berbagai jenis tugas dan persyaratan sistem.

Manajemen CPU dan *Scheduling*

Ilustrasi **Multi-level feedback queue scheduling**



Manajemen CPU dan *Scheduling*

Manajemen CPU - Interruption Management #1

Manajemen interupsi adalah proses penanganan interupsi, yang merupakan sinyal dari perangkat keras yang memerlukan perhatian dari CPU (Central Processing Unit).

Ketika interupsi terjadi, CPU akan menghentikan tugas yang sedang dikerjakan dan untuk sementara beralih ke rutinitas penangan interupsi, yang merupakan bagian kecil dari kode yang dirancang untuk menangani interupsi. Penangan interupsi biasanya melakukan beberapa tugas tertentu, seperti membaca data dari perangkat, memperbarui status sistem, atau memproses paket jaringan.

Manajemen CPU dan *Scheduling*

Manajemen CPU - Interruption Management #2

Manajemen interupsi adalah bagian penting dari desain sistem operasi, karena memungkinkan penggunaan sumber daya perangkat keras secara efisien dan memastikan penanganan peristiwa kritis secara tepat waktu. Berikut ini adalah beberapa aspek utama dari manajemen interupsi:

- Interrupt handling routines
- Interrupt priority
- Interrupt masking
- Interrupt queuing

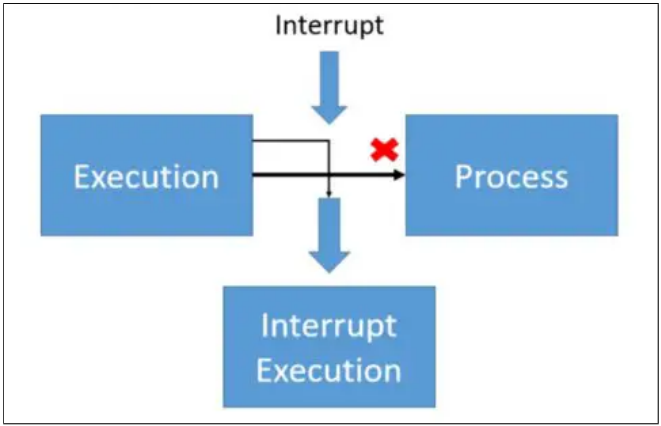
Manajemen CPU dan *Scheduling*

Manajemen CPU - Interruption Management - **Interrupt Handling Routines**

OS biasanya menyertakan serangkaian rutinitas penanganan interupsi yang dirancang untuk menangani berbagai jenis interupsi. Setiap rutin penanganan interupsi bertanggung jawab untuk melakukan tugas tertentu yang terkait dengan interupsi.

Manajemen CPU dan *Scheduling*

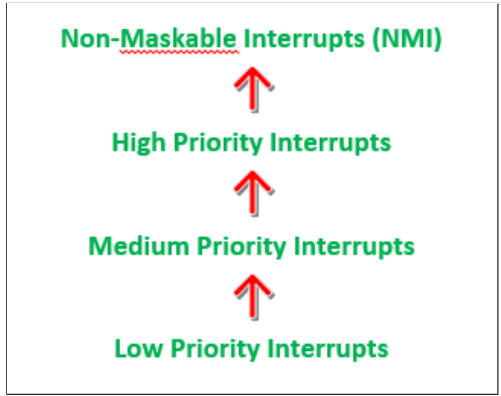
Ilustrasi **Interrupt Handling Routine**



Manajemen CPU dan *Scheduling*

Manajemen CPU - Interruption Management - **Interrupt Priority**

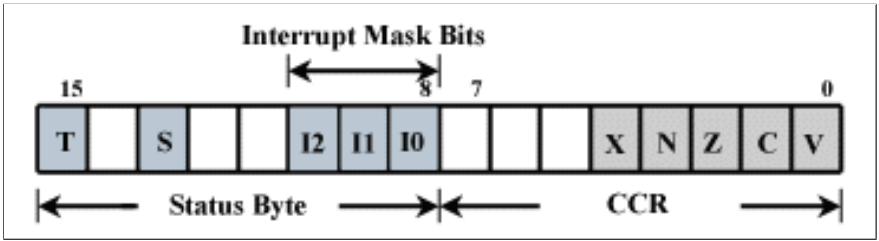
Interupsi dapat memiliki tingkat prioritas yang berbeda, tergantung pada tingkat kepentingan peristiwa interupsi.
OS menggunakan prioritas interupsi untuk menentukan interupsi mana yang harus ditangani terlebih dahulu.



Manajemen CPU dan *Scheduling*

Manajemen CPU - Interruption Management - **Interrupt Masking**

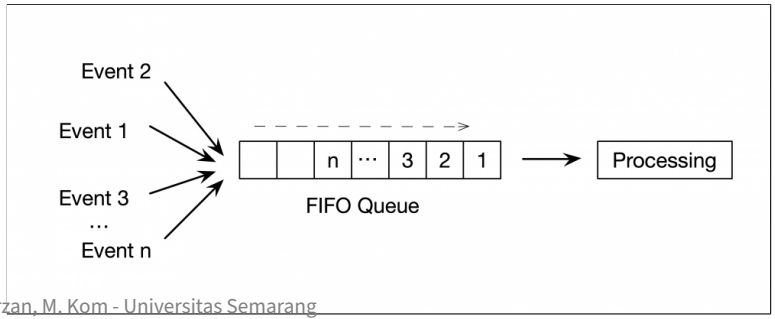
CPU dapat dikonfigurasikan untuk menutupi atau menonaktifkan interupsi tertentu, sehingga tidak mengganggu tugas yang sedang berjalan. Penyembunyian interupsi dapat berguna dalam situasi tertentu, seperti saat tugas penting harus diselesaikan tanpa gangguan.



Manajemen CPU dan *Scheduling*

Manajemen CPU - Interruption Management - **Interrupt Queuing**

Jika beberapa interupsi terjadi pada saat yang sama, biasanya interupsi tersebut akan di-antrekan dalam urutan tertentu untuk ditangani oleh CPU. OS menggunakan antrian in-terupsi untuk memastikan bahwa interupsi ditangani dalam urutan yang benar.




THANK YOU