System Requirements:-

1) Software Requirements:-

|  |  |
| --- | --- |
| Programming Language | C++ |
| Operating System | Windows 10 |
| Integrated Development Environment | Dev C++ |
| Compiler | MinGW |

System Analysis:-

1) Motivation

CPU scheduling plays an essential role in Real Time Operating Systems which always have a time constraint on computations.A real time system is the one whose applications are mission-critical and real-time tasks should be scheduled to be completed before their time duration elapses.

The Simple Round Robin architecture as well as the Priority Scheduling concept are not suitable to implement in real time operating systems due to more number of context switches and longer waiting time,both of which lead to a lower throughput in the system. If a real-time process arrives with a relatively larger CPU burst,it will lead to the problem of starvation,i.e,the lower priority process will have to wait for an indefinite amount of time before it can access the system resources.

Real time operating systems require real-time tasks to be completed before their deadlines and therefore require a dynamic task scheduling that can handle unforeseen configurations.

2) Proposed System

The aim of this project is to develop a new approach for task scheduling which improves the CPU performance in real time operating systems. The proposed algorithm is based on the integration of simple round robin scheduling algorithm, which helps reduce starvation, and priority scheduling,which helps to assign new priorities to the processes,thus implementing the concept of aging. The proposed architecture eliminates the drawbacks of implementing a simple round robin scheduling algorithm, which is inefficient for processes with smaller CPU bursts, and the feature that distinguishes it from other CPU scheduling algorithms is that, unlike others, it can be implemented in real-time operating systems.

The system working is as follows:-

* User inputs a tab-separated text file containing the process table
* The proposed algorithm runs on the provided input, and the processes are scheduled in an efficient manner such that the number of context switches, waiting time, and turnaround time are considerably reduced, thus resulting in improved CPU performance..
* For the given input, the algorithm generates performance parameters and a gantt chart, allowing the user to schedule processes in real time.

The proposed algorithm is implemented using C++ programming language.