ABSTRACT: The real time computing systems respond to input immediately therefore there are strict timing constraints that have to be met to get the correct output. Real time applications are expected to generate output in response to stimuli within some upper bound. Real time systems change its state in real time even after the controlling processor has stopped its execution. The real time application respond to the stimuli within a particular deadline. Scheduling is deciding how use the processor’s time on the computer and to provide efficient service to all users it is an arrangement of performing functions at specified time. The intervals between each function have been defined by the algorithm to avoid any overlapping. The scheduling techniques has been used in order to achieve optimized results in real time. In the paper we analyze various scheduling techniques and observes the various issues on which there is still a need to work.

Key Words: Scheduling, resources, power, real time operations

1 INTRODUCTION:

SCHEDULING, in computing, means how the processes can be assigned on the available CPU(s). It is a key concept of multitasking, multiprocessing and real-time operating system design [1,3-4]. It is done by a means of scheduler and dispatcher. A scheduler is a person or machine that organizes or maintains schedules. In computing, scheduler is software program that arranges jobs or computer’s operations in an appropriate order. A dispatcher is a module which gives control of CPU to the process selected by the scheduler. It is a decision making process that deals with the allocation of common resources to various tasks at different time periods to achieve multiple objectives. The resources and tasks can be of different forms in homogeneous/heterogeneous organization. Priorities have been associated with the tasks; each task has its due date and earliest dead line. Similarly objectives can also be in the different forms, they can be minimization of completion time of last task or minimization of number of tasks completed after their respective due dates [2,8-9]. In an industry it may mean to assign appropriate workers to do some task each day. It enables us to perform tasks which are done in routine automatically and efficiently.

LITERATURE SURVEY:
The following sections show the work done by the various researchers in the field of scheduling for real time processors.

In 2003 Sanjoy K. Baruah et. al. [18] presented a schedulability analysis of Rate Monotonic (RM) scheduling algorithm to determine schedulability of periodic task set on a particular uniform multiprocessor. They considered periodic model of hard real time tasks with two parameters namely execution parameter (C_i) and period (T_i). They have proved that all the deadlines will be met if task set is scheduled using Rate Monotonic (RM) algorithm.

In 2008 Euiseong Seo et. al. [16] presented an energy efficient technique for scheduling real time tasks on multicore processors to lower the power consumption and increasing the throughput. They presented two techniques which modify existing techniques of unicore processors for multicore processors. The two techniques suggested by them are: (i) Dynamic Repartitioning algorithm, which dynamically balances the task loads multiple cores to minimize the power consumption during execution. (ii) Dynamic core scaling algorithm, which reduces leakage power consumption by adjusting the number of active cores. The simulation results show that 25% of energy consumed can be conserved by dynamic repartitioning and 40 % is conserved by dynamic core scaling.
3 MOTIVATION:

Real time systems are the computing systems that must react within the precise time constraints to events in the environment. The efficient scheduling of tasks on these systems becomes necessary in order to receive accurate and timely response. From the work done by various researchers it has been observed that there is still a need to optimize scheduling techniques so that the various constraints of real time systems can be met and performance can be enhanced.

4 Objective of this work:

1. To study various algorithms used for scheduling processes on a processor.
2. To study various techniques used for scheduling real time tasks.
3. Comparison of different Real Time scheduling algorithms.

4. **Objectives of Scheduling:**

There are some goals that must be achieved in order to perfectly schedule the task on the processor. Some of those objectives are mentioned below.

4.1 **Fairness:**

It is important goal to achieve under all circumstances. A scheduler guarantees that each process should get a fair amount of CPU time for its execution and there should be no condition of starvation. But giving equal or equivalent time to all the processes is also not fair as different processes are not equally critical.

4.1.2 **Throughput:**

It is the amount of work a computer can do in a period of time. The scheduler aims to maximize the number of jobs completed or processed per unit time.

4.1.3 **Turnaround Time:**

It is the amount of time taken for the process to get executed. It is the interval from the time of submission of a process to the time of completion of the process. Turnaround time, $T_r = T_s + T_w$

$T_s =$ Execution Time, $T_w =$ Waiting time

The scheduler should minimize the turnaround time, but it is not upto the scheduler to affect the execution time but it can minimize the waiting time so as to reduce the turnaround time [3,8].

4.1.4 **Waiting Time:**

Similar processes are allocated equal CPU time or times are divided according to processes’ priority. The time every process waits for its execution in the ready queue is known as waiting time and it is the duty of scheduler to minimize this time [3].

4.1.5 **Efficiency**

It signifies the amount of time the system is performing tasks. Scheduler should keep the system busy for most of the time when possible. If both CPU and I/O devices are working all the time then more work gets done per unit time [5, 9]. Usually the above goals conflict with each other, as amount of the CPU time is finite, so there is a need of trade-off between the achievable goals. Every scheduling technique aim to achieve either of above mentioned goals. For scheduling tasks on a processing unit, there can be an optimization for power along with clock period. There are various techniques devised to optimize power and processing speed.

5 **TYPES OF OPERATING SYSTEM SCHEDULING**

Basic operating systems schedulers can be of three distinct types. Scheduling types are divided according to the schedulers and the names of schedulers suggest the relative frequency with which these functions are performed [3,6]. The schedulers are:

5.1 **Long-term or high-level scheduling:**

It decides which processes are to be added to the set currently executing processes and which are to be exited. It controls the degree of multiprogramming in multitasking systems with a need of trade off between degree of multiprogramming and throughput. The higher the number of processes, the smaller the time each of the processes may control CPU for. Long-term scheduler is also known as Admission Scheduler [7]. It is required to ensure that real time processes get enough CPU time to complete their tasks. The long term queue exists in the hard disk or virtual memory.

5.2 **Mid-term scheduling:**

The scheduling of processes is mainly done based on the requirement of the resources. It is essentially concerned with memory management and often designed as a memory management subsystem of an operating system. It temporarily removes a process from the main memory which is of low priority or has been inactive for a long time. This is known as “swamping out” of a process. The scheduler may decide to swap out the process which is page-faulting frequently or a process which is taking large amount of memory. Its efficient interaction with the short term
scheduler is very essential for the performance of the systems with virtual memory.

5.3 Short term scheduling/CPU Scheduler:
It is concerned with the allocation of CPU time to meet the processes in order to meet some pre-defined objectives. It decides which of the in-memory process is executed following an interrupt or operating system call. This scheduler makes more frequent scheduling decisions than long-term and mid-term schedulers [5, 10] It can be preemptive or non-preemptive depending on the requirement. Mostly, it is written in assembler as it is a critical part of operating system

6 SCHEDULING PROBLEMS:
Consider we have m machines Mj(j=1,2,...,m) and we have to process n jobs Ji(i=1,2,...,n). Allocation of time intervals on machine is known as schedule and is represented by Gantt charts, which can be machine oriented or job oriented. [9]

Basic notations used in scheduling:
n: it represents the number of operations which is consisted in Job Ji, represented by Oi1, Oi2,.....,Oin
p: processing time requirement associated with operation Oi
r: represents when the first operation of job Ji is available for processing, known as release date.
µi: represents the set of machines associated with each operation Oi. µi ∈ [M1,M2,....,Mm]. any operation can be processed on any machine. d: due date or deadline (in case of real time operations), time by which Ji should be ideally completed.
w: weight, representing the relative importance of Ji.
f(t): non decreasing cost function, that measures the cost of completing Ji in time t. There are many classes of scheduling problems which are classified according to their complexity and specified in terms of classification which consists of three fields [10].

7 SCHEDULING ALGORITHMS:
It is a method by which resources are allocated to the processes such as bandwidth, processor time etc. It is usually done to distribute workload across multiple computers or processing units so as to achieve optimal resource utilization, maximize throughput, minimize response time and avoid overload [4]. The goal of scheduling algorithm is to fulfill the following criterion.
i. No Starvation:
This means that a particular process should not be held up indefinitely. There should be proper allocation of resources to every process so as to ensure that all processes get CPU time.

ii. Preemption in case of priority based algorithms
Scheduling algorithms have to ensure that there should be fairness in the preemption policy. It should make sure that high priority tasks should not hold low priority tasks indefinitely.

7.1 Categorization of scheduling algorithms
Scheduling algorithms can be classified on the basis of the type of environment in which they are being used. This categorization is based on user's point of view. According to this, scheduling algorithms can be classified into three categories.

7.2 First Come First Serve scheduling (FCFS):
It is the simplest scheduling algorithm in which processes are dispatched according to their arrival time on the ready queue. It is a non-pre-emptive technique, so when processes get the CPU time, they are executed to completion. It is fair in the human sense of fairness but unfair in the sense that long jobs make short jobs wait or important jobs might get held up because of unimportant jobs. As the process is executed until it gets completed, there is no condition of starvation as long as process completes its execution.

No prioritization of processes is there. It cannot guarantee good response time, so this technique is inappropriate for interactive systems. It is rarely used in modern operating systems but it is usually embedded within other schemes.[5]

7.3 Shortest Process Next:
In this scheduling policy, scheduler schedules the processes with the least estimated processing time to be next in queue. This technique is also known as Shortest-Job-First or shortest next-CPU-burst first. CPU burst is the time processor is being used by process before it is no longer ready or how long a process requires CPU between I/O waits. This policy considers that we know the next CPU burst of all the ready processes. The estimation of the length of next CPU burst is based on the recent CPU bursts. This technique is non-preemptive but pre-emption can be included with this technique and the resulting technique is known as Shortest Remaining Time next. The problem with shortest job first algorithm is that it cannot handle infinite loops and delivers poor performance if the task with short CPU burst comes after the process with longer burst which has started execution. Another problem associated with this technique is starvation i.e. processes with long burst times are indefinitely postponed from getting on the processor. In most of the cases, this technique is designed to achieve maximum throughput as short processes get most of the CPU time.

7.4 Scheduling Issues
While performing scheduling, there are multiple constraints which are to be met for accurate scheduling. There are issues relating to different tasks the system performs. The requirement of the system decides the scheduling algorithm design. We have to consider the application profile of the system; the level of
scheduling required, the time at which a process is scheduled, the parameter to be optimized, the need of pre-emption etc. Some of the issues are classified below:

1. **Application Profile**
   It specifies the type of applications to be run on the system. A program alternates between CPU utilization and I/O. While performing scheduling we need to consider the bound of the programs executed on the system. It can be either compute-bound or I/O bound. For the I/O bound operations scheduler decides which processes should be scheduled while the currently running task is performing some I/O operation.

2. **Scheduling level**
   There can be only one process at a time which gets CPU time. So, the scheduler has to decide the process which is to be executed next. The decision is dependent on the choice of scheduling algorithm. Swapper is provided to decide about the processes which should reside in the memory. This type of scheduling is known as midterm scheduling.

3. **Time of schedule**
   It is the duty of scheduler to decide about the time at which schedule is started. This is done by the help of scheduling algorithm. The response of the system on receiving an interrupt, on creation/termination of a process, etc is determined by the scheduler. When there is a system call, it is the scheduler which decides which processes are to be put in ready queue.

4. **Pre-emptive or Non pre-emptive**
   The system can be interrupted during the execution of the process in case of arrival of a higher priority process. This type of system is known as pre-emptive system. If no interruption is there then the system is non pre-emptive system.

8. **Conclusion and Future Scope:**
   The various objectives of my work have been met. Study of scheduling algorithms have been done and it has been observed that preemptive scheduling with dynamic priorities works very well in case of scheduling tasks on real time systems. From the comparison of real time scheduling algorithms, it is clear that earliest deadline first is the efficient scheduling algorithm if the CPU utilization is not more than 100%. For hard real time systems, calculations of probabilistic worst case execution time (WCET) analysis can be done. Implementation of scheduling algorithm on FPGA can be done for scheduling tasks with dynamic priorities and schedulability of the task can be checked.

**REFERENCES**


