Design of Real Time scheduler simulator and Development of Modified Round Robin architecture

Yaashuwanth .C IEEE Member, Dr.R. Ramesh
Department of Electrical and Electronics Engineering,
Anna University Chennai, Chennai 600 025

Abstract:
The main objective of this paper is to develop a new scheduling algorithm for scheduling of tasks in real time operating systems. The proposed architecture is a modified version of round robin architecture which is used for scheduling of tasks in real time operating systems. It is observed that the proposed architecture solves the drawbacks of simple round robin architecture in real time operating systems by decreasing the number of context switches waiting time and response time thereby improving the system performance This paper also explains the development of a new Web-enabled simulation framework: to study and evaluate the performance of various uniprocessor real-time scheduling algorithms for real-time scheduling. Task ID, deadline, priority, period, computation time, and phase are the input task attributes to the scheduler simulator and chronograph imitating the real-time execution of the input task set and computational statistics of the schedule are the output. The proposed framework for the scheduler simulator is mainly developed to be used as a teaching tool. The Web-based deployment of the simulator enables the user a platform-, machine- and software-independent utilization of the technical resource.

Keywords:
Scheduling algorithm, round robin, real-time, uniprocessor.

1. INTRODUCTION

A real-time computing system can be defined as a real-time application which is expected to respond to stimuli within some small upper bound on response time and any late result is as bad as a wrong one. Thus correctness of a real-time system could be stated true with logical perfection in the computational result and its timeliness. For real-time embedded systems, the primary objective is to ensure that all tasks meet their deadlines.. A real-time operating system (RTOS) is a multitasking operating system intended for real-time embedded applications. RTOS facilitates the creation of a real-time system, but does not guarantee the final result will be real-time; this requires correct development of the software. An RTOS does not necessarily have high throughput; rather, an RTOS provides facilities which, if used properly, guarantee deadlines can be met generally or deterministically. An RTOS will typically use specialized scheduling algorithms in order to provide the real-time developer with the tools necessary to produce deterministic behavior in the final system. An RTOS is valued more for how quickly and/or predictably it can respond to a particular event than for the amount of work it can perform over a given period of time. Key factors in an RTOS are therefore a minimal interrupt latency and a minimal thread switching latency.

Real-time scheduling theory has shown a transition from cyclical executive based infrastructure to more flexible scheduling models such as fixed-priority scheduling, dynamic-priority scheduling, feedback scheduling or extended scheduling[4]. In fact, recent studies shows that almost every existing real-time operating system provides only POSIX-compliant fixed-priority scheduling since it can be easily implemented in commercial kernels. The author[12] has discussed the basic architecture of fixed priority scheduling which implies that there will be single server and the jobs will be allocated based on the priority given by the user. The aurthor[5] proved that standard analysis of fixed priority assumes that all the computations of each task must be completed within task deadlines but in practice this is not the case, deadlines is most currently associated with last observed event of the task. The internal events and kernel overheads can occur after the deadlines. Reindir and Pieter revealed the worst case response time analysis of real time tasks using hierchial fixed priority scheduling[15] .Using an example which consist of single server the author portrays that the existing worst case response time analysis can be improved, from these analysis it is conclude that there is a need to develop a new scheduling algorithm which is a modified version of round robin architecture that incorporates priority component for tasks which solves the drawbacks like large waiting time and response time in real time operating systems.

Real time systems always have a time constraint on computation. Each task should be invoked after the ready time and must complete before its deadline[6],[7],[8]. An attempt has been made to satisfy the constraints in many scheduling algorithms both non preemptive and preemptive[12].. Richard roehi proposed a new way of scheduling which implements a new priority queue in the round robin architecture that gives priority to tasks with

Manuscript received March 5, 2010
Manuscript revised March 20, 2010

short central processing unit(CPU) burst thereby improving the performance of the tasks with less CPU burst[13]. Fair scheduling with tunable latency (Klaus H. Ecker 1996) is a round robin approach that proposes an alternative and lower complexity approach to packet scheduling, based on modifications of the classical round robin scheduler. The authors showed that appropriate modifications of the Weighted Round Robin service discipline can, in fact, provide tight fairness properties and efficient delay guarantees to multiple sessions. Round robin approach has its applications on networks by allowing the network devices to have a free share of network resources. Various types of scheduling algorithms such as Dificit Round Robin[9],[10],[11]. have been implemented. Real-time simulation tools speed up the decision making processes during the selection of suitable scheduling algorithm for a real-time embedded application. They also stand as teaching tool helping learners of real-time system grasp the core ideas related to system modeling quickly. There have been various simulator frameworks created for this purpose, too . The performance analyses of the above mentioned simulator frameworks were carried out and it is found that these simulators are not user friendly and hence the need for developing a new Web-based simulator framework was discovered.

2. NEED AND SIGNIFICANCE OF WEB-BASED DEPLOYMENT

There are vital reasons behind the implementation of the simulator as a Web-enabled framework and Web-based deployment as pointed out here. They are ease in deployment and enhancement of functionality. Anytime, anywhere access to users any computer with Web connectivity can be used for learning and teaching. Easy access to users over the Internet since no extra hardware or software is required to access the application.

3. DESIGN AND IMPLEMENTATION OF PROPOSED ALGORITHM ROUND ROBIN PRIORITY

The proposed algorithm eliminates the drawbacks of implementing a simple round robin architecture in real time system by introducing a concept called intelligent time slicing depends on two aspects they are Priority and context switch . The proposed architecture allows the user is allowed to assign Priority to the system . . A dedicated small Processor used to reduce the burden of the main Processor which calculates the time slice for the tasks based on their Priority and these tasks are arranged based on Priority execute in the main Processor with their individual time slices. The proposed architecture can be implemented in real time because of greater response time and throughput and the users can allocate Priority to every individual task. The proposed algorithm works efficiently when the range of CPU burst for tasks is large and the scheduler follows static scheduling. The function of the small Processor are shown in figure

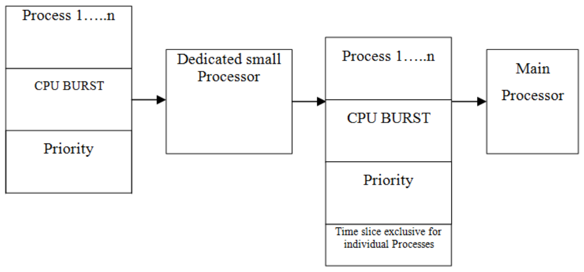


Figure 3.1 Block Diagram of proposed Algorithm

Table 3.1 Input component for the processor

Process ID	CPU burst time (milliseconds)	Priority
1	25	2
2	5	3
3	15	1
4	8	5
5	10	4

Time slice calculation for proposed architecture he time slice for the proposed architecture is shown

$$\text{Timeslice} = \frac{\text{Range}\left(\text{R}\right)\ \text{X Total no. of Process in the system}\ \left(\text{N}\right)}{\text{Priority}\left(\text{Pr}\right)\ \text{X total no. of Priority in the system}\ \left(\text{P}\right)}$$

$$\text{Range} = \frac{\text{maximum cpu burst} + \text{minimum cpu burst}}{2}$$

The time slice is calculated for all Process based on the Priority and scheduled in the ascending order based on Priority .figure shows Process 3 has the calculated time slice of 15 milliseconds and has the highest Priority so it is scheduled at the beginning. The Process 3 executes for 15 milliseconds and is pre empted and given to Process 1 which has the next highest Priority. The Process 1 executes in time slice of 8 8milliseconds by the Processor. In this way all the Processes are scheduled and executed in the ready queue

$$\text{Range} = \frac{25 + 5}{2}$$

Table 3.2 Calculation of timeslice for proposed algorithm

Process No	CPU Burst	Priority (Pr)	Range (R)	Total No. of Process in the system (N)	Total No. of Priority in the system (P)	Time slice $\frac{RxN}{\Pr xP}$
1	25	2	15	5	5	8
2	5	3	15	5	5	5
3	15	1	15	5	5	15
4	8	5	15	5	5	3
5	10	4	15	5	5	4

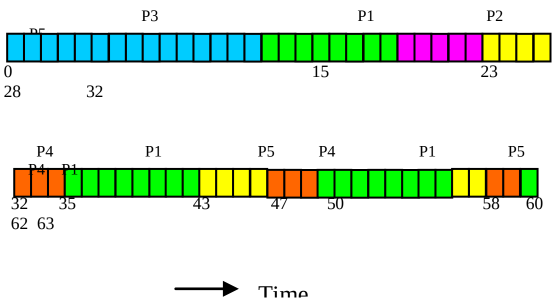


Figure 3.2 Execution of tasks in the proposed algorithm

Comparison with round robin round robin architecture cannot incorporate priority in the system. It schedules the tasks in first come first serve manner based on the time slice which is equal for all the tasks in the system. In the proposed architecture priority is incorporated in the round robin architecture. The tasks are scheduled based on priority and the time slice will be calculated for each and every individual tasks and the tasks executes on the processor based on the calculated time slice. As shown in the table P3 has the highest priority so it has less waiting time and less turn around time when executed in normal round robin architecture. This proposed architecture explains the proposed architecture is superior to normal round robin architecture. Tasks with highest priority less waiting time and less turn around time thereby increasing the overall system performance of hgher priority tasks

4. DEVELOPMENT OF WEB BASED SIMULATOR

4.1. Design of Proposed Web-Enabled Simulator

This section describes the development of the proposed simulator framework in java environment. A framework for evaluation of a scheduling algorithm must satisfy characteristics such as simplicity, compatibility with the PC platform usage of the standard operating system functions, accuracy of results, ease of use etc. Majority of these requests are aimed for use in the visual user interface that looks as shown in the Figure 4.1.The proposed Web-enabled scheduler simulator could be operated through a Web browser through a set of click-on and data input windows.

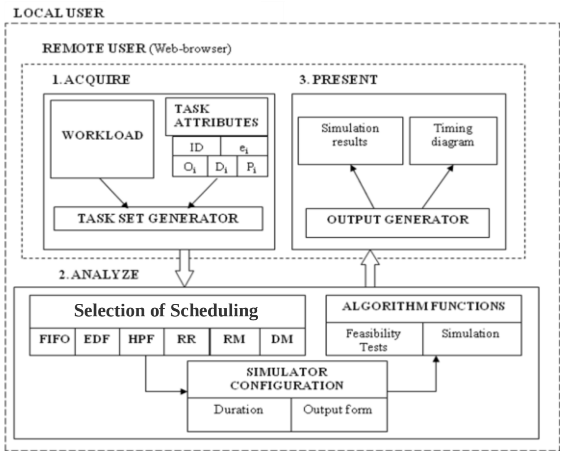
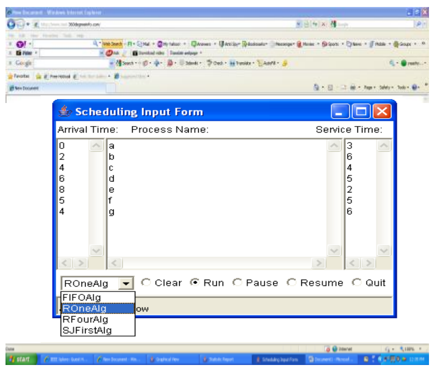


Figure 3 Block Diagram of proposed simulator

Scheduling algorithm evaluation and analysis tool performs the task definition, task sets generation, execution of selected algorithms, execution analysis of the execution and results are displayed. The performance evaluation of the real-time scheduling algorithms is carried out based on the results obtained through computational analysis. The proposed simulator has one input panel (scheduling input form) and two output panels (statistics form, graphical view panel) .

4.2 IMPLEMENTATION OF PROPOSED WEB BASED SIMULATOR

Scheduling input form The task parameters (eg arrival time , service time etc) can be defined by the user in the scheduling input form . similarly the users can also select the scheduling algorithm. The control functions are also defined in the scheduling input form. The user view of the input panel is shown in figure 4.2.


Scheduling Input Form

Arrival Time:	Process Name:	Service Time:
0	a	3
2	b	6
4	c	4
6	d	5
8	e	2
5	f	5
4	g	6

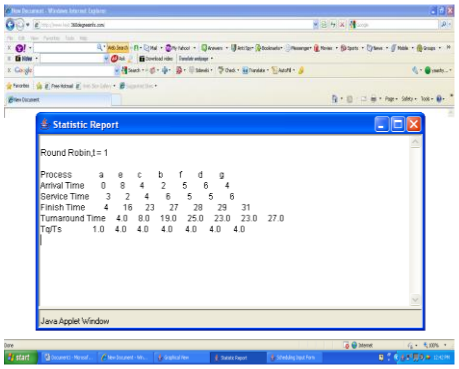
ROneAlg
FIFOAlg
ROneAlg
RFourAlg
SJFirstAlg

Clear [✓ Run] Pause Resume Quit

Figure 4.2 Task scheduling input form in the simulator

The proposed simulator has been developed and deployed in the web platform and the users can access the simulator in the weblink shown below
http//: www.test.360degreeinfo.com.

Statistics form The statistics form gives the statistics which is computed during the execution of tasks in a simulator. Criteria such as task waiting time, turn around time etc are computed and are visible to the user in this panel as shown in figure 4.3



Statistic Report

```
Round Robin,t= 1

Process          a     e     c     b     f     d     g
Arrival Time     0     8     4     2     5     6     4
Service Time     3     2     4     6     5     5     6
Finish Time      4    16    23    27    28    29    31
Turnaround Time    4.0   8.0   19.0   25.0   23.0   23.0   27.0
Tq/Ts        1.0   4.0   4.0   4.0   4.0   4.0   4.0
```

Java Applet Window

Figure 4.3 Statistics Form

The most successful scheduling algorithm for the periodic tasks scheduling is the one that has minimal response times, minimal number of tasks with missed deadlines and maximal resource utilization in the given workload and with other parameters.

The complete task model is too complex for implementation and some of the task parameters are hence ignored. In real-time systems two characteristics of tasks are considered to be of primary interest: criticality or importance; and timing. Task importance is frequently a subjective issue, whereas timing is objective. The essential timing attributes of tasks are deadline (T_D), worst-case computation time (Tc_w) and period (T_p),

Graphical view panel the output of task simulator graphical view panel is shown in figure 4.4

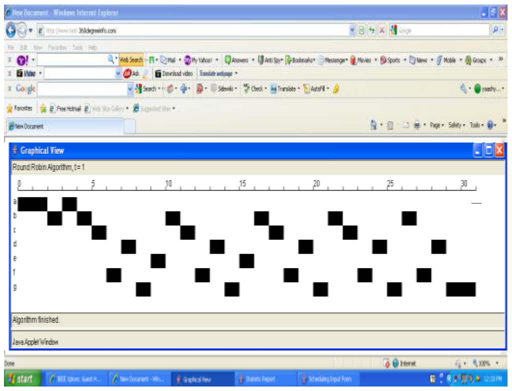


Figure 4.4 Output (timing diagram)

5. CONCLUSION

The proposed Web-enabled framework gives the developer the possibility to evaluate the schedulability of the real time application.. Numerous benefits were quoted in support of the Web-based deployment technique employed. The framework which is proposed can be used as an invaluable teaching tool. Further, the GUI of the framework will allow for easy comparison of the framework of existing scheduling policies and also simulate the behavior and verify the suitability of custom defined schedulers for real-time applications. In addition, the real-time scheduler co-processor hardware can help the learners have a closer view of scheduling tasks in real-time hardware. also the new proposed algorithm which is developed is implemented on the proposed java simulator and this algorithm has greater waiting time and response time compared to normal round robin architecture

REFERENCES

[1] Arnoldo Diaz, Ruben Batista and Oskardie Castro, (Sep.2007), Realtss: a real time scheduling simulator, in 4th International Conference on Electrical and Electronics Engineering (ICEEE), pages 165-168, Mexico City, Mexico, http//:www.ieeexplore.ieee.org/iel5/4344970/4344971/04344998.pdf?arnumber

[2] Jan Blumenthal, Frank Golatowski, Jens Hildebrandt, Dirk Timmermann, (2002), Framework for Validation, Test and Analysis of Real-Time Scheduling Algorithms and Scheduler Implementations, in Proceedings of the 13th IEEE International workshop on Rapid System Prototyping (RSP'02).

[3] F. Singhoff, J. Legrand, L. Nana, L. Marce, (Nov. 2004), Cheddar: a Flexible Real Time Scheduling Framework, in ACM SIGADA'2004 International conference proceedings, Atlanta, Giorgia, USA.

[4] Sha, L., Abdelzaher, T, Arzen, K., Cervin, A., Baker, T.P., Burns, A., Buttazzo, G., Caccamo, M., Lehoczky, J., and Mok, (2004) "A. Real time scheduling theory: A historical perspective". Real-Time Systems, 28(2):46-61.www.springerlink.com/index/W2312110GL0243K8.pdf

[5] Burns.A (1994) "Fixed priority scheduling with deadline prior to completion" Real-Time systems Research Group Department of computer science university of york, UK

[6] Enrico Bini and Giorgio C. Buttazzo (2004) "Schedulability Analysis of Periodic Fixed Priority Systems" journal on IEEE TRANSACTIONS ON COMPUTERS VOL 53 NO.11, pg 1462-1473.

[7] Zhi Quan and Jong-Moon Chang (2003) " A Statistical Framework for EDF Scheduling" journal on IEEE COMMUNICATION LETTERS VOL 7 NO.10, pg 493-495.

[8] Houssine Chetto and Maryline Chetto (1989) " Some Results of Earliest Deadline Scheduling Algorithm" journal on IEEE TRANSACTION ON SOFTWARE ENGINEERING. VOL 15. NO.10, pg 1261-1269.

[9] Shreedhar. M , George Varghese (1996) "Efficient fair queuing using deficit round robin" IEEE/ACM

TRANSACTIONS ON NETWORKING VOL 4, NO3, pg375 -385

[10] Viet.L.Do and Kenneth Y.Yun (2003) “An Efficient Frame based scheduling Algorithm: Credit round robin “ San Diego, IEEE

[11] Marco.J, . Meziat.D, . Alarcos.B “Dificit round robin alternated : a new scheduling algorithm”_ E.P. 28871 Alcalá de Henares, Spain

[12] Barbara Korousic –Seljak (1994) “Task scheduling policies for real-time systems” Journal on MICROPROCESSOR AND MICROSYSTEMS, VOL 18, NO. 9, pg501-512.

[13] Richard roehi (1995) “Designing a fairer round robin scheduling algorithm” SIGCOMM ’95 Cambridge, MA USA 0 ACM. www.cmg.org/proceedings/2005/5056.pdf

[14] J.E. Cooling, P. Tweedale, (1997), Task scheduler co-processor for hard real-time systems, in Microprocessors and Microsystems, Vol. 20, pp. 553-566.

[15] Reindir J. bril and Pieter J.L. Cuijpers (2006) “Analysis of hierchial fixed priority preemptive scheduling “TU/e, CS-Report 06-36, www.win.tue.nl/~pcuijper/docs/CSR-06-36.pdf.