MULTISHAPE TASK SCHEDULING ALGORITHM FOR REAL TIME MICRO-CONTROLLER BASED APPLICATION

Ankur Jain

Department of Computer Engineering, ABV-IIITM, Gwalior, M.P.

ABSTRACT

Embedded Systems are usually microcontroller-based systems that represent a class of reliable and dependable dedicated computer systems designed for specific purposes. Micro-controllers are used in most electronic devices in an endless variety of ways. Some micro-controller-based embedded systems are required to respond to external events in the shortest possible time and such systems are known as real-time embedded systems. So in multitasking system there is a need of task Scheduling, there are various scheduling algorithms like Fixed priority Scheduling (FPS), Earliest deadline first (EDF), Rate Monotonic(RM), Deadline Monotonic(DM), etc have been researched. In this Report various conventional algorithms have been reviewed and analyzed, these algorithms consists of single shape task, A new Multishape task scheduling algorithms has been proposed , implemented and analyzed.

KEYWORDS

RTOS,Microcontroller, Scheduling Algorithms,Embedded systems

1. INTRODUCTION

Today, Embedded electronics have stronghold in every space of the society. To full fill the demand of people lots of research and development are going on in this domain.In Particular, frequently changing needs of people gives rise to Reconfigurable real time system. Those systems are typically embedded devices implemented on Microcontroller based platforms which support reconfiguration in the resources even during run-time.These resource reconfiguration is managed by the Real time perating system (RTOS). RTOS is an operating systems designed for real time embedded systems. RTOS is a collection of different real time algorithms for managing resources, schedule the tasks and provide lots of other functionality for real time applications. In this paper we propose a novel Multishape task scheduling algorithm for scheduling real-time task and compare with the conventional algorithms [1].

Real time systems are those systems which supports real-time constraints. Real time systems can be categorized into three categories i.e. Hard real time systems , soft real time systems, and firm real time systems. Hard real time systems are those systems in which time delay in response can results potential failure to the system or the loss of human life. In general there is a cost function associated with the system. Many of these systems are considered to be safety critical. Soft real time system are those system where deadline overruns are tolerable, but not desired. There are no hazardous consequences of missing one or more deadlines. Firm real-

time systems are mix of hard real time and soft real time system. Here infrequent deadline missing are tolerable but it degrades the performance of the systems. To avoid the deadline missing, we need to schedule the task effectively [2].

To solve a scheduling problem, we try to find a schedule for the tasks to execute in such a way so that tasks can be completed before the deadline. Scheduling can be used in any domain where is the limited number of resources to be scheduled efficiently. This efficiency means optimizing desired criteria. Such criteria could be to minimize the schedule length, to maximize the resources utilization ratio to maximize the number of accepted tasks. In this paper, we analyzed algorithms for the periodic tasks. A periodic task is an infinite sequence of jobs with periodic ready times, where the deadline of a job could be equal to, greater than or less than ready time of the succeeding jobs [3]. Scheduling algorithms can be classified in, Uni- processor vs Multiprocessor Scheduling, Soft real time vs hard real time Scheduling, Static vs Dynamic Scheduling, Fixed vs Dynamic Priority Scheduling, Pre-emptive vs Non pre-emptive Scheduling. So for real time systems different Task scheduling algorithms have been studied to avoid the deadline missing. Some of the popular real time task Scheduling algorithms are Earliest Deadline first(EDF), Fixed-priority Task scheduling, rate monotonic(RM),Deadline monotonic (DM) etc[4] [5].

Rate monotonic Scheduling algorithm is mostly researched, reviewed, and analyzed algorithm. It is a priority driven algorithm in which priorities are assigned according to the cycle period of the job i.e. the task which has less cycle duration, get the higher priority. So for periodic Tasks, priority of the all instances of the task is known before the arrival and it will be same for all instance. It supports pre-emption and work well on uni-processor system [6].

In DM is a scheduling algorithm, priorities are assigned according to relative deadline Di i.e. higher priority is assigned to the task which has less deadline. It will be fixed for all the future instances. In general, DM is optimal for systems that consist of pre-emptive synchronous independent tasks whose relative deadline is less to their period (Di $\leq$ Period).DM could be used with periodic, aperiodic and sporadic tasks systems[7].

EDF algorithm is a dynamic priority algorithm, the priorities are assigned based on the value of relative deadline of the tasks in real time. The task with nearest deadline is given highest priority and it is selected for execution. EDF could be applied to various tasks models (preemptive, non-preemptive, periodic, non-periodic, etc.).EDF has also been shown to be optimal in the case of non-periodic tasks. EDF scheduling outperforms RM and produces less pre-emption compared to RM and it is very fast. [8].

Here we have observed that, Conventional algorithms work on single shape hardware task. That why effective utilization of Micro-controller is less. To increase the effective utilization, we need different task shape(version) of different execution time and area. To solve this problem we need an algorithm, which can schedule right shape of task, according to Microcontrollers free resources. At different load condition, Conventional algorithms gives different performance, So we need a switching mechanism so that we can get better performance. So here we propose a novel Multishape scheduling algorithm for microcontroller applications where tasks are rejected due to busy resources.

2. SYSTEM MODEL

2.1 Basic Definitions

Task is a process that needs to be done in sequential manner. Resources are the devices that can be used to perform the task it can be active (processor) or passive (mutex, file, database lock).Schedule is the assignment of resources to the particular task. Scheduling algorithm is the process by which RTOS creates the schedule. Feasible Schedule is the Sequence of task made by the RTOS Scheduler which satisfy all the constraints.

2.2 Model

- A unit of work which is to be scheduled and executed by the system is called task. All the tasks are assumed to be periodic. The system knows about arrival time, period of the task, required processing time and task deadline in priori.
- No, Precedence constraints are taken in account. It is assumed that any task can execute in any order relative to each other and a task is released as they arrive.
- we assumed that there is a resource constraints. Task can be pre-empt with the higher priority task. It is assumed that scheduling algorithm and pre-emption incurs no overhead.
- In real time system, each and every task keep some positive value. The main aim of algorithm is to gain maximum value. so if task succeed, then the system gets its value. If task fails then system losses its value[9].

3. METHODOLOGY

Here we have considered an instance on which four task occurring at the same time. Algorithm choose the compatible task from Task sets(primary task list **(PTL)** or secondary task list **(STL)**) suitable to resource available. These Task sets have been defined below,

T1 = {t1(PTL),t1(STL)}
T2 = {t2(PTL),t2(STL)}
T3 = {t3(PTL),t3(STL)} and so on..

3.1 Task attributes

In figure 1, Task attributes are shown graphically which are described below,

• **Name** it is the label of the task.

• **Processing Time {Pi}** it is the time consumed by the processor to execute the task. it it also know as completion time or execution time.

• **Period {per}**it is the inter arrival time for the periodic task.

• **Release time (or ready time) {Ri}** : It is time at which scheduler can process the task i.e. task is ready for execution.

• **Deadline {Di}**: It specifies a time limit by which the task has to be completed, otherwise the scheduling is assumed to fail.

• **Due Date {dj}** specifies a time limit by which the task should be completed, otherwise the criterion function is charged by penalty.

• **Minimum delay**:It is a average time duration that must be passed between task release time and the task start time.

• **Maximum delay** :It is maximum permitted amount of time that can be passed between task release time and the task start time.

• **Worst case execution Time** : It is maximum amount of time (after the task release) which is taken to execute the task completely .It is also known as the worst case response time.

• **Run time**:It is time taken(without interruption) to execute completely the task.

• **Weight (or priority){Wi}** :It is a positive value which give precedence over other task .

• **Start Time Si** it is the time at which task is executed by CPU.

• **waiting time {Wj }** it is the difference between Release Time and start time.
$W j = Si – Ri$

• **completion time {Ci}** it is the Time at which CPU has completely executed the task.
$Ci = W j + P i$

• **Flow Time** It is the time duration between Releasetime and completion Time.
$F i = Ci – Ri$

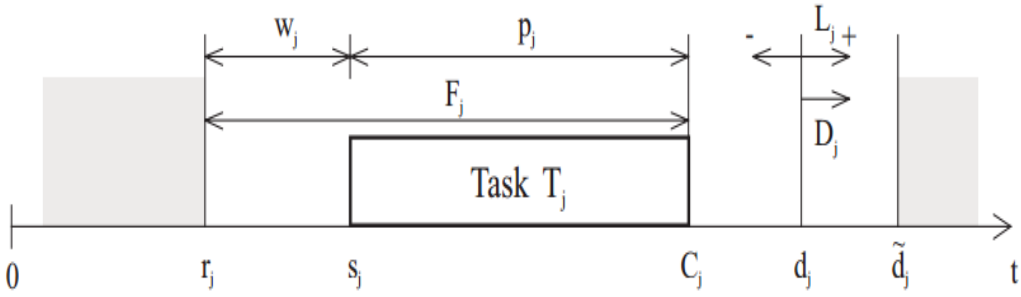

Figure 1. Graphical representation Task attributes

4. TASK STATES

Scheduler maintains three states of any task which is shown in figure 2.

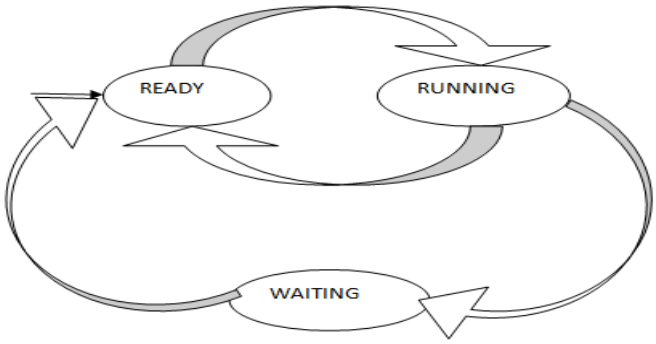

Figure 2. Task states

• Ready : It is list in which scheduler maintains those task which are ready to execute.

• Running : The task which is currently running in processor, stored in this list.

• Waiting : It is the list in which, scheduler maintains those tasks for which resources are not available.

5. ALGORITHM DESCRIPTION

Input :

Task Sets { T1,T2,T3,.....}
Constraints : Timing , Precedence Constraints

Output : Real time Schedule

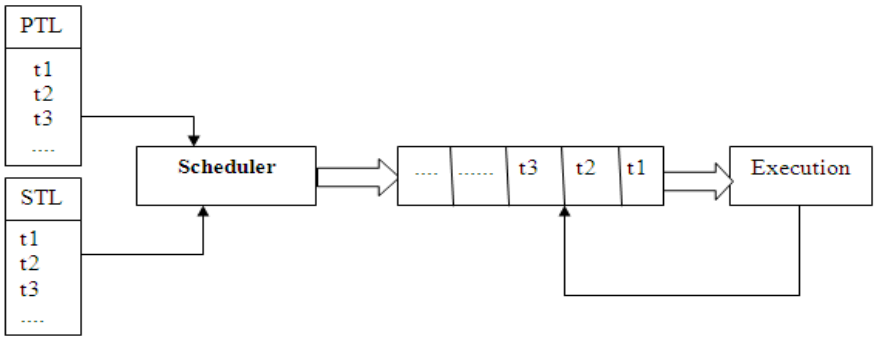

Figure 3.Scheduling process

1) **Task selection** : At first, tasks are selected from primary task list.
2) **Resource check** : Resources are checked from the status register, if resources are unavailable then go to step-1 select the task from secondary task list.
3) **Resource Allocation** : Resources are allocated till the scheduling interval.

4) **Assign Priority**: Priority will be assigned according to the deadline of the tasks i.e. the task has short deadline will get higher priority.
5) **Queuing**: Selected tasks are sent to the scheduler where they are queued according to priority.
6) **Execution** : Then tasks are applied to the processor to execute the task.
7) Terminate the process.

6. RESULT

6.1 Task instance

Table 1. Primary task list

Task name	Period	ProcTime	Deadline	Resources used
Task1(t1)	3	1	3	a,b,j
Task2(t2)	5	1	5	c,d
Task3(t3)	6	1	4	f,g
Task4(t4)	10	2	8	j,d

Table 2. Secondary task list

Task name	Period	ProcTime	Deadline	Resources used
Task1(t1)	3	1	3	a,c,j
Task2(t2)	5	1	5	e,f,g
Task3(t3)	6	1	4	b,g
Task4(t4)	10	1	8	e,h

In fixed-priority Scheduling, task priority has been set before arrival to the scheduler, scheduler takes the decision according to the priority i.e. higher priority task would be schedule first, if lower priority task will arrive at same time as higher priority task it will be rejected, in figure. 4 we can see that, task t1 has lowest priority thats why it is rejected by task t2 at time instant 2, and rejected by t3 at time instant 3.because of this rejection Task success ration an Effective CPU utilization is very low.

In Rate monotonic Scheduling, Tasks are prioritized according to the cycle period of the task i.e. The task has lower cycle time which will be schedule first or take higher priority. In figure 5 ,we can see that t1 has lower cycle duration, thats why it is being scheduled first and t4 has lower priority that why it being schedule last, it is being pre-empted by high priority tasks and has higher response time.

In Deadline monotonic Scheduling, Tasks are prioritized according to the Deadline of the task i.e. The task has lower Deadlines which will be schedule first or take higher priority. In figure 6,we can see that t1 has lower Deadlines, thats why it is being scheduled first and t4 has lower priority thats why it being schedule last, it is being pre-empted by high priority tasks and has higher response time.

In Multishape task scheduling, we prioritize the task according to the resource availability and deadlines of the task. if primary tasks are not able to fulfil the Deadlines, secondary tasks are being selected at the cost of parallel high performance resources. Figure7 shows the Multishape

task schedule in which task 4 is being selected from the secondary task list, because task1 already captured resource j. So that we can save the task4 by selecting alternative task shape.

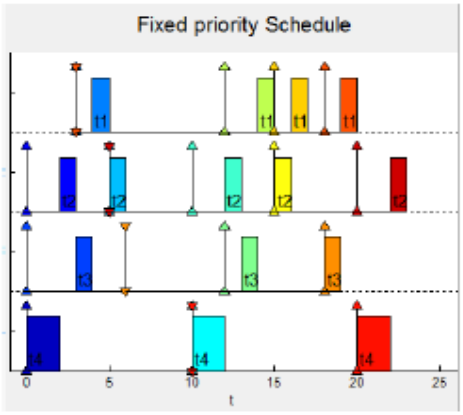

Fig. 4. fixed priority Schedule

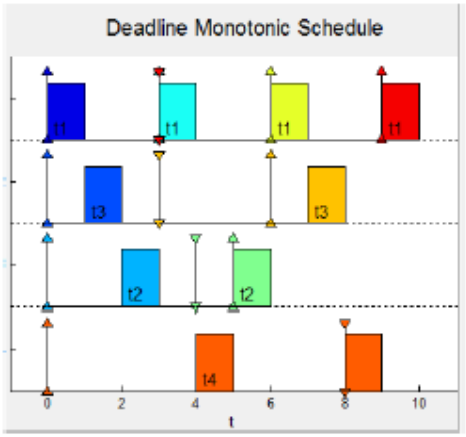

Fig. 6. Deadline monotonic Schedule

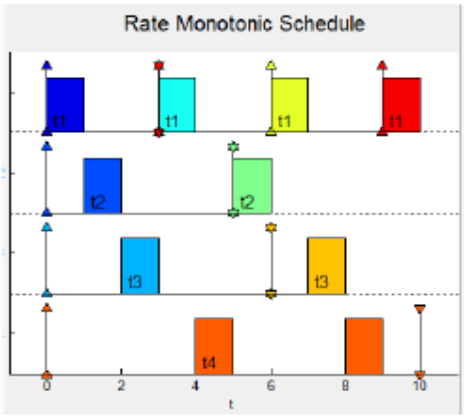

Fig. 5. Rate monotonic Schedule

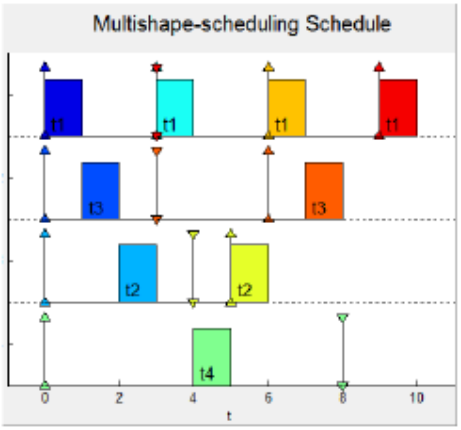

Fig. 7. multishape Task Schedule

6.2 Algorithm Analysis

6.2.1 Worst Case Response Time $(WCRT)_i$: It is a maximum time required to execute the task in a worst case.

$WCRT_i$ = Max(R(i)) in Scheduling interval

6.2.2 Level-I Active Period: It is a time interval (a,b) for which level-I pending workload (> = P i) is positive. All instances in level-I active period should be considered for the analysis of algorithm.

6.2.3 Schedulability Test: Task set is Schedulable if all task of Task set meet their deadlines.

Condition : $(WCRT)_i <= D_i$

Steps to determine Schedulability:

- Determine worst case response Time$(WCRT)_i$
- Identify the critical instant of Task(i)
- Find level-I active period of the Task(i)
- Check $WCRT_i <= D_i$

6.2.4 Critical Instant: It is a instance at which Task is pre-empted by higher priority task, so the task take largest time to execute. This situation comes when the task is released concurrently with higher priority tasks.[6] In fig 5 Critical instant of Task(4) occurs at a time when it arrives at the same time as Task(1),Task(2) or Task(3).

6.3 Performance parameter

we have considered four main performance measuring criteria on which the proposed algorithm will be analyzed.

6.3.1 Execution Time(E): It is the time required to complete the task. The execution time required by task is an important parameter especially when working with real time systems.

6.3.2 Load: The periodic tasks load (L) of the system is the sum of the ratio of the execution time and the deadline of individual task. it can be determined using the following equation:

$$L = \sum_{i=0}^{m} E_i / D_i$$

where,
m= No. of task, E= Execution time of Task, D= Deadline of the Task

- $L > 1$: it shows the overload condition, some task will fail to meet its deadline no matter what algorithm is being used.

- $L < 1$: it shows the under load condition, so Schedule will depend on scheduling algorithm.

6.3.3 Success Ratio(SR): It is defined as the ratio of No of tasks which is achieved there deadline to Total no of tasks arrived. In real-time systems, deadline meeting is the most important. It is defined as,

$$SR = \frac{N}{TA}$$

Where , N= no of task achieved deadline, TA = total no of Task arrived

6.3.4 Effective CPU Utilization(ECU): The CPU utilization is a measure on how busy the processor could be during the shortest repeating cycle, Effective CPU Utilization gives information about how efficiently the processor is used and it is defined as

$$\text{ECU} = \sum_{i=0}^{t} \frac{E_i}{T}$$

where, E = Execution time E_i(if deadline met) otherwise '0'
T = Total scheduling time of the tasks,
t = No of task which met the deadlines.

7. CONCLUSION

In this paper, we have studied various task scheduling algorithms and proposed a novel Multishape task scheduling algorithm. we found that in distributed shared resource environment Multishape algorithms works well because of the different versions available. In future, we are going to work on a practical real time application. It will be the Target application for performing the scheduling algorithm. we will implement different versions of task for satisfying the concept of Multishape task scheduling algorithm.

REFERENCES

[1] K. Jozwik, S. Honda, M. Edahiro, H. Tomiyama, and H. Takada,(2013) "Rainbow: An operating system for software-hardware multitasking on dynamically partially reconfigurable fpgas," Int. J. Reconfig.Comput., vol. 2013, pp. 5:5–5:5.

[2] C. Leng, Y. Qiao, H. Wang, J. Liu, and X. Zhang,(2013) "A new utilization based admission control algorithm for aperiodic tasks with constant time complexity under edf scheduling," in Embedded and Real-TimeComputing Systems and Applications (RTCSA), 2013 IEEE 19th International Conference on, Aug 2013, pp. 338–341.

[3] V. Prajapati, A. Shah, and P. Balani, (2013)"Design of new scheduling algorithm llf dm and its comparison with existing edf, llf, and dm algorithms for periodic tasks," in Intelligent Systems and Signal Processing (ISSP), 2013 International Conference on. IEEE, 2013, pp. 42–46.

[4] S. Tak, T. Kim, and E. Park,(2010) "Integrating real-time inter-task communication channels into hardware–software codesign," Microprocessors and Microsystems, vol. 34, no. 6, pp. 182–199.

[5] J. Carpenter, S. Funk, P. Holman, A. Srinivasan, J. Anderson, and S. Baruah,(2004) "A categorization of real-time multiprocessor scheduling problems and algorithms," Handbook on Scheduling Algorithms, Methods, and Models, pages, pp. 30–1.

[6] J. Lehoczky, L. Sha, and Y. Ding,(1989) "The rate monotonic scheduling algorithm: Exact characterization and average case behavior," in Real Time Systems Symposium, 1989., Proceedings. IEEE, 1989, pp. 166–171.

[7] N. C. Audsley, A. Burns, M. F. Richardson, and A. J. Wellings,(1991) "Realtime scheduling: the deadline-monotonic approach," in in Proc. IEEE Workshop on Real-Time Operating Systems and Software. Citeseer,1991.

[8] Q. Zhao, Z. Gu, and H. Zeng, (2013) "Integration of resource synchronization and preemption-thresholds into edf-based mixed-criticality scheduling algorithm," in Embedded and Real-Time Computing Systems and Applications (RTCSA), 2013 IEEE 19th International Conference , pp. 227–236.

[9] K. Kotecha and A. Shah, "Adaptive scheduling algorithm for realtime operating system," in Evolutionary Computation, 2008. CEC2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on. IEEE, 2008, pp. 2109–2112.

[10] G. Wassi, M. E. A. Benkhelifa, G. Lawday, F. Verdier, S. Garcia et al., (2014)"Multi-shape tasks scheduling for online multitasking on fpgas," in International Symposium on Reconfigurable Communication-centric Systems on-Chip (ReCoSoC'2014).