Review on Multicore Signalling Processing

Himani Verma
M. E. Scholar
Department of Electronics and Communication Engg.
Ujjain Engineering College , Ujjain (UEC , Ujjain)

Prof. Vineeta Choudhary
Professor
Department of Electronics and Communication Engg.
Ujjain Engineering College , Ujjain (UEC , Ujjain)

Abstract: In this paper researcher have talked about different blame tolerant assignment booking calculation for multi-centre framework dependent on equipment and programming. Equipment based calculation which is mix of Triple Modulo Redundancy and Double Modulo Redundancy, in which Agricultural Vulnerability Factor is considered while choosing the booking other than EDF and LLF planning calculations. In a large portion of the ongoing framework the predominant part is shared memory. Low overhead programming based adaptation to non-critical failure approach can be executed at client space level with the goal that it doesn't require any progressions at application level. Here repetitive multi-strung procedures are utilized. Utilizing those procedures author can distinguish delicate mistakes and recuperate from them. This technique gives low overhead, quick mistake location and recuperation component. The overhead brought about by this strategy ranges from 0% to 18% for chose benchmarks. Cross breed Scheduling Method is another booking approach for continuous frameworks. Dynamic blame tolerant planning gives high plausibility rate though undertaking criticality is utilized to choose the kind of blame recuperation technique so as to endure the greatest number of issues. Developing CPU booking calculations and understanding their effect practically speaking can be troublesome and tedious because of the need to alter and test working framework bit code and measure the subsequent execution on a predictable remaining burden of genuine applications. As processor is the critical asset, CPU planning turns out to be imperative in achieving the working framework structure objectives. The aim ought to be permitted however many as could be expected under the circumstances running procedures at untouched so as to make best utilization of CPU.

Keyword: Multicore Processor, Blame Tolerant, Powerful Booking, Registration, Earliest Deadline First, Task Graph, Scheduler, State Diagrams, CPU-Scheduling, Performance.

I. INTRIDUCTION

1.1 A Brief Introduction to Digital Signal Processing: Signal processing is just the control of the properties of a particular signal to get a signal with progressively attractive properties. Properties, for example, plentifulness, stage, or recurrence range might be adjusted to meet a particular necessity. In the good 'ol days electronic specialists accomplished signal handling utilizing discrete equipment parts, for example, resistors, capacitors, inductors, transistors, diodes, and other semiconductor gadgets. In such a case a signal variable that was ceaseless with time was utilized as a contribution to an equipment gadget that delivered another form of the signal variable where a portion of the properties have been adjusted. In advanced signal preparing the procedures that were accomplished utilizing equipment are finished utilizing programming.

So as to process signal factors that are persistent with time utilizing programming the factors must be changed over to the right arrangement; regularly a grouping of numbers. This is finished utilizing simple to advanced converters. A processor would then control the signal in some ideal design. In the wake of experiencing the processors the subsequent succession of numbers must be changed over back to simple utilizing computerized to simple converters. In the beginning of computerized signal handling the processors were moderate and the uses of advanced signal preparing were constrained. Today, we have extremely quick and power proficient processors that the utilizations of computerized signal handling have expanded drastically.

So as to comprehend computerized signal preparing it is essential to initially take a gander at signal characterization in the time-area.

1.2 Signal Classification: Signals s can be ordered regarding the congruity of the autonomous and ward factors as pursues:

(i) An analogy signal: The autonomous and the reliant factors characterizing the signal are persistent in time and adequacy. This implies for each predefined time case, the signal has predetermined adequacy esteem.

(ii) Continuous-time signal: The time variable is ceaseless in the range in which the signal is characterized. On the off chance that the signal variable is spoken to by x, time variable is t such a signal is meant as x (t).

(iii) Discrete-time signal: The time variable is discrete in the range in which the signal is characterized. In the event that the signal variable is x and the time variable has been inspected at time examples n, where n = n'T then the signal is signified as x(n). A discrete time signal is additionally alluded to as a tested signal since it is gotten by specifically examining a focused on signal. It ought to be noticed that the abundance of the inspected signal can take any an incentive inside a predetermined plentifulness range and researcher thusly state that the sufficiency of discrete-time signal is constant.

(iv) A digital signal: This is a signal that is discrete in time and discrete in adequacy. It is spoken to similarly as a discrete-time signal.

Signals can likewise be grouped as far as the consistency of the reliant factors as for the autonomous variable as pursues:

(i) A signal is said to be deterministic if the needy variable is unsurprising at any occasion of the autonomous variable time. A deterministic signal can be communicated by an unequivocal scientific articulation.

(ii) A random signal, on the hand, has an eccentric ward variable at any occasion of the free factor time. Such a signal must be characterized as far as its measurable properties.

All the above arrangements of advanced signal can additionally be ordered regarding their dimensionality. Here, researcher will just expand this arrangement utilizing discrete-time groupings and researcher will leave the rest to the understudy.

(i) A one-dimensional signal has just a single free factor and one-subordinate variable. A discrete-time signal x(n) is a one-dimensional signal as it has just a single free factor, discrete-time (n), and one-subordinate variable, the abundance of x(n).

(ii) A two-dimensional signal has two-autonomous factors and one-subordinate variable. The examples n and m are taken in the spatial area. The two-dimensional signal is discrete in the spatial area in two-measurements. The autonomous factors are n, m which characterize the reliant variable x (n, m). A genuine model is a photographic picture where n, m characterize the spatial area and x (n, m) characterizes the dark dimension at the area.

(iii) A three-dimensional signal has three-free factors and one-subordinate variable. A discrete-time signal x (n, m, τ) is a three-dimensional signal as it has two-free factor in the spatial space (n, m) and one-autonomous variable τ in the time area. The three-free factors characterize the one-subordinate variable, the power of x(n, m, τ). A case of a three-dimensional signal is video signal where a signal at spatial area (n, m) is changing regarding time τ.

A framework can be named simple, discrete-time or advanced frameworks relying upon the sort of signal they handle. A simple framework would create a simple signal from a simple information signal. Then again, a discrete-time or advanced framework can create a discrete time or computerized signal from a discrete-time or computerized signal. In any case, a discrete-time or computerized framework can deliver a simple signal from a simple contribution with the guide of ADC and DAC. Starting now and into the foreseeable future we won't recognize discrete-time signals and advanced signals as these are dealt with similarly. Researcher will likewise not recognize computerized frameworks and discrete-time frameworks as there is in fact no contrast between them.

Numerous characteristic wonders produce simple signals and signal processors are inalienably computerized frameworks. In this manner so as to process simple signal with advanced processors the simple signal must be changed over to computerized. The procedure whereby simple signal are changed over to advanced signal includes inspecting and quantization. In the following area, researcher will talk about the examining procedure.

As the Semiconductor Technology is developing step by step it's currently conceivable to have a billion of transistors on a little chip. These little transistors are increasingly helpless to both transient and perpetual flaws. Thusly, by expanding the financial plan of the chip, Multicore configuration is prominent for upgrading the system throughput. By extending the amount of transistors and getting their sizes, the rate of Software bungles on processors can't be ignored; thus, the constancy of such structures has been a champion among the most fundamental challenges as more diminutive transistors are progressively vulnerable to faults. Time imperatives, vitality effectiveness throughput are the vital measure in the plan procedure of continuous frameworks [1]. A multi-center processor has at least two autonomous centers into a solitary bundle. A double center processor has two free centers and quad center has four centers. The benefits of multicore processor unit over the many single center processors unit are (1) higher throughput, (2) direct power utilization, (3) effective usage of processor centers, (4) superior per unit cost [2].

ARM MP Core and IBM Cell are the instances of multicore processors utilized in the constant installed frameworks. Multicore processors are grouped into two sections, (1) homogenous or heterogeneous [3]. [4] states that the greater part of the current multicore processors are homogenous. The multicore processor is fundamentally worry for dealing with the assignments in such an approach to use the centers viably. The scheduler in the working framework is in charge of keeping every one of the centers in the processor caught up with amid the execution of the constant errands to enhance the all-out execution time. Deficiencies can be classified into these principle classifications: changeless, transient and discontinuous issues [5].

Lasting Fault, for example, wear off of any parts which require substitution from the extra part to re-establish the framework usefulness. Transient blame are the momentary blames and can be recognized from others with their term of event and causes. These may happen because of outside clamor and different sources intermittent blame happens occurred at breaks by virtue of

some internal dull glitch of the fragments like temperature instabilities, control supply and commotion. To have adaptation to internal failure in multicore frameworks errand planning is finished. These adaptations to non-critical failure booking calculations increment the framework unwavering quality. In various blame tolerant framework, programming which is running on a solitary center utilize repetitive execution at various dimension of reflection, at guidance and virtual machine level. Techniques which work at guidance level have low blunder discovery latencies contrast with equipment level. Be that as it may, techniques which work at procedure level permit mistake engendering. In multi-strung projects which are running on multicore processors, Shared memory get to is visit than hinders or flags. For that accomplish productive execution of reproductions is particularly troublesome. For that diverse deterministic dialects are utilized. Author can perform adaptation to non-critical failure utilizing repetitive execution of programming in which reproduced duplicates give same yield for given information. This technique can be executed utilizing a client level library so it doesn't require alteration in part. The blunder identification system is enhanced to perform memory correlations of the reproductions proficiently in client space [6].

Relies upon the utilization of different strings execution can be further increment. For planning of delicate ongoing errands with non-continuous undertakings two dimension various levelled booking is utilized. This strategy diminishes normal due date miss proportion and furthermore bolsters the ongoing necessities for different assignments. The rule target of equipment based calculation is to find a fitting assignments undertaking on the centers and endure handling segment disappointments which could either be erogenous or heterogeneous. The heterogeneity of the processors suggests that they have assorted speeds or changing capacities. The transient imperfection probability that may occur in transistors, doors and even a bit, is called Architectural defencelessness variable (AVF) [5]. By averaging over some time, this segment can describe the rate of delicate mistakes that can appear on a center; while it runs an undertaking. This calculation when contrasted and different strategies we found that the proposed blame tolerant technique defeats both TMR and DMR strategies about 35% in normal. In processing most regular topologies in N-Modulo Redundancy are TMR and DMR. TMR technique is utilized when dependability of errand is critical as it can just veil the shortcomings. The issue is that voter can likewise be defective. The other technique DMR incorporates two centers running parallel and checks for the likeness in the yield consequently just show the confound doesn't endure the blame so need to utilize separate component for blame recuperation [5]. In programming based adaptation to non-critical failure based methodology reproduction of procedure are utilized so it is important that imitations utilize same memory addresses. Author likewise need to guarantee that pioneer and adherent utilize same copy duplicates.
Copy can be made utilizing fork framework bring in which process creates same procedure as supporter procedure and it works same as pioneer process. On the off chance that the consequence of both pioneer and adherent procedure is same, at that point author can say that there is no mistake. Be that as it may, on the off chance that there is distinction among pioneer and adherent procedure, at that point mistake is there. Author can utilize check focuses for productivity. In the event that after execution of pioneer and adherent procedure result is same, at that point past checkpoint is killed and if result is unique in relation to process is again begun from past checkpoint. Author can likewise utilize normal interim techniques for mistake location. In that pioneer and adherent procedures are analysed after settled interim of time say 100ms. On the off chance that author discover distinction it implies blame identified. In multicore engineering, no comparator equipment is utilized; along these lines assignment replication assumes a key job in their design. Picking a check pointing with rollback recuperation expands the likelihood of finishing the errand on time [7]. In the memory assignment, malloc can be utilized for memory portion. Malloc can be non-deterministic in light of the fact that it utilizes mmap for allotment of memory blocks of expansive sizes and mmap itself is non-deterministic. So author need to ensure that pioneer and adherent utilize same memory utilizing mutex, which is bolted and opened deterministically. Many examine are continuing planning of intermittent and a periodic undertaking. One of the traditional booking components is based on need of the errand [8]. A creator in [9] presents planning strategy which utilizes the rate moronic planning for which the need of the assignment is characterized based on the time required. The errand having shorter time span have the most astounding need and allocated the center first. The Earliest Deadline First calculation in which the undertaking with prior due date has the most noteworthy need. Another booking approach is Primary Backup (PB) that is a champion among alternate systems, in which two variant of the errand is plan on two distinct centers. An acknowledgment test is required for approval of results.

II. SCHEDULING METHODOLOGIES

This paper discuss about various scheduling algorithm for multi core system.

A. Hybrid Scheduling : The calculation talked about in [5] is the blend of TMR and DMR adaptation to internal failure planning calculation in which by utilizing the Agricultural Vulnerability Factor (AVF) of each undertaking the scheduler can foresee the event of blame when the errand is running on the center. Each errand is characterized as a 4-tuple, Ti = (ai, di, ci, AVFi) where ai, di, ci and AVFi indicate landing time, due date, execution time and building powerlessness factor of each undertaking. For compelling booking every one of the errands must fulfill their time constraint and each center ought to have one undertaking doled out to it. In this way the effectiveness of calculation is assessed by all out execution time of the assignment and the use of the center. Center use can be determined as:

$$Utilisation = \frac{\sum_{i=1}^{n} P_i}{N \times T} \quad (1)$$

Where, Pi signifies the time each center is occupied with running distinctive assignments, N decides number of centers and T is all out execution time of the undertaking gathering. According to [10], in Hybrid Scheduling Method approach one can segment application in to most extreme number of parallel assignments with the goal that best outcome can be accomplished. Each parallel undertaking can run simultaneously with different assignments. Here It might happen that constant errands and non-ongoing undertaking can run simultaneously. In this half breed planning strategy, two dimension booking arrangement is utilized. At the best dimension sporadic server is utilized for planning approach and at base dimension, a rate-monotonic OS scheduler is utilized.

B. Implementation Method: To start with, arranging of the errands in holding up line is done based on their landing time. Than the quantity of centers required for undertaking to keep running in experimental mode is dictated by contrasting AVF of each assignment with the AVF threshold. In the event that the assignment AVF is more noteworthy than or equivalent to AVF threshold than it will executed in TMR mode or else in DMR mode. On the off chance that any fault occurs than undertaking is re executed in DMR mode [5].

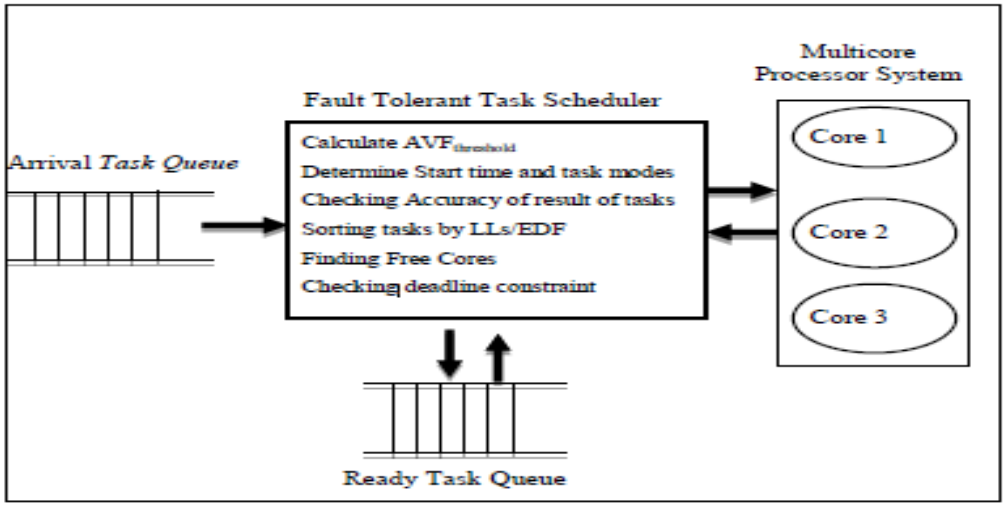

Figure 1. The Schematic of Fault tolerant task scheduling algorithm for multicore systems.

C. Non-pre-emptive EDF Scheduling: There are a few powerful need booking algorithms such as Earliest-Deadline-First (EDF), Least-Laxity (LL), Least Slack-Time-First (LST), and Minimum-Laxity-First (MLF), where in every one assignments are organized and planned by explicit parameter. According to [1] EDF is an optimal calculation for single processor framework with a lot of preemptiveindepedent tasks.[11] demonstrates that the adequate yet not the essential condition to have a practical planning for various pre-emptive assignment with all out usage U on a lot of M single center processor is

$$U \ll \frac{M}{2M-1} \qquad (2)$$

This can be considered as a limit for multicore processor with M centers. At whatever point an assignment is discharged, it will be included to Ready List, and its total due date (AD) will be determined. At the point when there is just a single inactive center, the framework checks whether the errand with the most punctual due date can be booked on this inert center or not. On the off chance that the errand can be booked, it will be allocated profoundly, and the center will be expected occupied with amid the execution. Something else, missing the assignment due date implies that planning of this application on the given design isn't doable. Consider a case of an undertaking set, comprising of six assignments is appeared in figure 2. It shows the EDF booking of this assignment set on a quad-center processor. From the figure, it tends to be seen that, T2 has least AD so T2 is booked first on P1, following that T1is planned on P2, T3 onP3. The challenge is seen somewhere in the range of T4 and T5 where both are discharged in the meantime, however the supreme due date of T5 is sooner than T4,soT5is planned first. Alternate undertakings will be planned on inactive centers in a similar way [1].

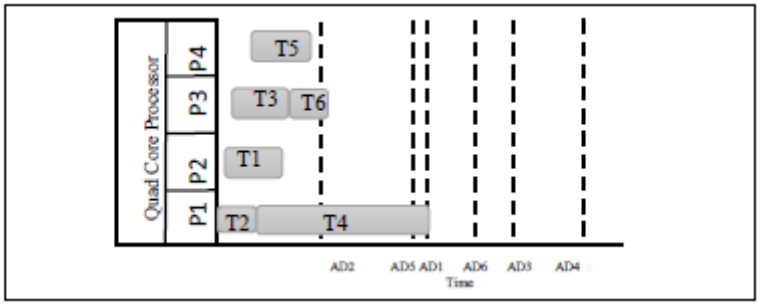

Figure 2: EDF task scheduling of a task set on a quad-core processor

Reexecution and recuperation with checkpoint are progressively proper to be used with delicate constant systems whereas time excess strategies are fitting for hard ongoing frameworks with tight deadlines. However, check pointing and re-execution strategy expands the complete execution time. In [12], real-time planning alludes to the issue in which there is a due date or time associated with the execution of an endeavor. Creator's present a planning plan, called most punctual due date with vitality ensure (EDeg) that combine represents qualities of the vitality source, limit of the vitality stockpiling just as vitality utilization of the assignments, and time.

D. Check Pointing Optimization: There are exchange offs between applying successive and rare checkpoints for errands in a framework. Visit check pointing diminishes re-execution time within the sight of deficiencies, while assignment execution time is expanded. Then again, inconsistent check pointing has lower time overhead without flaws, while the measure of re-execution will be expanded if blame is recognized. In [13], it is demonstrated that the optimal number (m) of checkpoints considering k blames in the undertaking is given by:

$$m = \left\| \sqrt{\frac{K \times C}{C_s}} \right\| \qquad (3)$$

Where, Cs is time overheads of sparing a checkpoint, Cr is time overhead of recouping from a checkpoint. The most pessimistic scenario reaction time of an errand for Check pointing with rollback recuperation (Cc) can be given by

$$Cc = (C + m \times C_s) + K \times (C_s \times C_r) + \frac{K \times C}{m+1} \qquad (4)$$

Where (C+ m x Cs) gives the execution time of an undertaking utilizing check pointing with no shortcomings, and (Cs + Cr) +C/m+ 1 is the expense of fault recuperation for single blame, (see also [p18]). If the rate of event of blame on one center is high, at that point an assignment needs more opportunity to recoup from issues; this may prompt miss its due date. In such cases equipment replication techniques are utilized as they have the capacity of parallel execution of the duplicated duplicates of unique errands on the other preparing centers.

E. Harvesting Aware Real-Time Scheduling Algorithm: In [14] continuously implanted framework alongside planning of the assignments control the board is additionally an issue that ought to be embraced. In this paper, creators propose a gathering mindful continuous planning calculation which expects to diminish the vitality utilization while feasibly plan the course of action of irregular assignments inside their due date (deadline).This should be possible by Dynamic Voltage and recurrence choice, executing the undertaking inside the speed with the end goal that it can devour as much vitality as required for its culmination fulfilling its time constraint.

F. Dynamic Voltage and Frequency Selection Algorithm: In [15] this paper Author's propose a low-multifaceted nature and errand task mapping, planning, and power the board technique for multi-center continuous installed frameworks with vitality gathering. This strategy depends on the CPU use. This technique consolidates new powerful voltage and recurrence choice calculation alongside the vitality reaping mindfulness structure the proposed usage based calculation. On multi center framework this calculation gives successful use of gathered vitality.

G. Dynamic Fault Tolerant Scheduling: [1] portrays the dynamic blame tolerant planning (DFTS) calculation. This calculation utilizes assignment criticality which depends on "usage" and "time of asset portion". Assignment usage is utilized to progressively choose the sort of blame recuperation technique so as to endure the most extreme number of flaws. In light of the errand criticality parameters, undertaking is ordered into basic and non-basic. Basic undertakings will be recreated on isolated centers to expand the likelihood of on time errand fruition in defective condition. Non basic errands are planned on a solitary center and check pointing with rollback recuperation blame tolerant method is connected on them. The planning possibility rate of the DFTS calculation is higher than other blame tolerant booking techniques.

H. Fault-Tolerant Scheduling Based on Task Criticality: The calculation referenced in [1] chooses an appropriate adaptation to internal failure strategy task for each errand when the assets for the undertaking are accessible. Amid booking it is to be taken consideration that no errands will be planned until every single other undertaking with higher needs are planned and the recently happened blames in the framework are endured. The scheduler ascertains the criticality limit for the errand which is available at the primary position of the prepared rundown when perfect center is accessible in the framework. When a perfect center is dispensed to each undertaking is characterized as Resource Allocation Time (RAi).

$$Delay_i = RA_i - R_i \quad (5)$$

Postponement shows the time squandered between the undertakings went into Ready rundown and time when the scheduler appoints equipment assets to that assignment and applies blame tolerant arrangements. Expanding delay for an assignment may lead a noncritical undertaking ends up basic. So as to figure the criticality edge and endure the normal blames in each errand, the scheduler applies registration procedure to non-basic undertakings while equipment replication system to basic assignments.

I. Task Graph Scheduling Using NABBIT: As talked about in [16], this calculation relies upon some of the data from the client about the assignment diagram like Task key, Sink undertaking, Predecessors and Successors. In the wake of giving this data the undertaking chart booking calculation catches the structure of the errand diagram. This calculation is based on NABBIT errand chart scheduler who depends on the work taking. In the errand diagram the undertakings are alluded by keys and the runtime through a simultaneous hash guide will control the execution. A made undertaking is embedded into the hash map utilizing the INSERT TASK IF ABSENT daily practice and later with a call to GETTASK. For each assignment the runtime holds the join, inform exhibit and status to continue further. The assignment execution starts with creation and addition of the sink errand into the hash map, trailed by a summon of the initandcompute work. initandcompute begins the assignment and advances its quick antecedents through calls to tryi nitcompute. conjuring initandcompute and tryinitcompute in a recursive design, the execution extends the assignment diagram and achieves one of the source errands with no approaching conditions in the wake of executing the undertaking refreshes its status as Computed and begins informing the successors engaged with its advice exhibit. After the last successor in the advice exhibit, the errand changes its status to completed.

III. ANALYSIS OFSCHEDULING TECHNIQUES

Scheduling Algorithm	Characteristics
Hybrid Scheduling	This assignment booking calculation can guarantee the correct execution of running errand and decrease total time of undertaking execution by growing the centers while considering AVF of each undertaking.
Non-Pre-Emptive EDF Scheduling	EDF is an ideal calculation for single processor framework with a lot of pre-emptive autonomous errands
Check Pointing Optimization	check pointing diminishes re-execution time within the sight of deficiencies, while assignment execution time is expanded
Harvesting Aware Real-Time Scheduling Algorithm	This calculation diminishes the vitality utilization while feasibly plan the course of action of irregular errands
Dynamic Voltage And Frequency Selection Algorithm	This is the low-multifaceted nature and undertaking task mapping, booking.
Dynamic Fault Tolerant Scheduling	booking achievability rate of the DFTS calculation is higher than other blame tolerant planning techniques
Fault-Tolerant Scheduling Based on Task Criticality	Here, undertaking criticality is utilized to choose the kind of blame recuperation strategy so as to endure the greatest number of issues
Task Graph Scheduling Using NABBIT	This calculation can effectively recuperate from a discretionary flaws that are relative to the measure of work lost

IV. SCHEDULING PARAMETERS

a) CPU Utilization: It is the normal portion of time, amid which the processor is occupied.
b) Throughput: It alludes to the measure of work finished in a unit of time. The quantity of forms the framework can execute in a timeframe. The higher the number, the more work is finished by the framework.
c) Waiting Time: The normal timeframe a procedure spends pausing. Holding up time might be communicated as turnaround time less the real execution time.
d) Turnaround Time: The interim from the season of accommodation of a procedure to the season of culmination is the turnaround time.
e) Response Time: Response time is the time from accommodation of a demand until the main reaction is created.
f) Priority: give particular treatment to forms with higher needs.
g) Fairness: Avoid the procedure from starvation. Every one of the procedures must be given equivalent

V. CONCLUSION

In this paper, numerous fault tolerant scheduling algorithms applicable for multicore processor systems are discussed. According to the system requirements the scheduling technique should be chosen. But all algorithm of real time scheduling have some restriction so it is future's need to suggest some hard real-time scheduling algorithm which will decrease the energy and also timing overhead by utilizing speed in such a way that response time of task is less than or simply equivalent to the existing approach despite the fact on the cost of lesser energy consumption.

REFERENCES

1 Mohammad H. Mottaghi, Hamid R. Zarandi, ―DFTS: A dynamic fault-tolerant scheduling for real-time tasks in multicore processors‖, Microprocessors and Microsystems 38 (2014) 88–97.
2 F. Kong, W. Yi, Q. Deng, ―Energy-efficient scheduling of real-time tasks on cluster-based multi-cores‖, in: Design Automation and Test in Europe, 2011, pp. 1–6.
3 Saifullah, K. Agrawal, C. Lu, C. Gill, ―Multi-core real-time scheduling for generalized parallel task models‖, in: 32nd IEEE Real-Time Systems Symposium (RTSS), 2011, pp. 217–226.
4 Chen, L.K. John, ―Efficient program scheduling for heterogeneous multi-core processors‖, in: Design Automation Conference (DAC), 2009, pp. 927–930.
5 ShamimShiravi* and Mostafa E. Salehi, ―Fault Tolerant Task Scheduling Algorithm for Multicore Systems‖,The 22nd Iranian Conference on Electrical Engineering (ICEE 2014), May 20-22, 2014,pp. 885–890.
6 Hamid Mushtaq, Zaid Al-Ars, Koen Bertels, ―Efficient Software-BasedFault Tolerance Approach on Multicore Platforms‖, EDAA, 2013
7 Ying Zhang and KrishnenduChakrabarty, ―Fault Recovery Based on Checkpointing for Hard Real-Time Embedded Systems‖, Proceedings of the 18th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'03) , 2013
8 S. Gotoda, M. Itoa and N. Shibata, ―Task scheduling algorithm for multicore processor system for minimizing recovery time in case of single node fault,‖ 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp.260,267, 13-16, 2012
9 Ching-Chih Han, K.G. Shin and J. Wu, ―A fault-tolerant scheduling algorithm for real-time periodic tasks with possible software faults,‖ IEEE Transactions on Computers, vol.52, no.3, pp.362,372, 2003
10 Pengliu Tan, Jian Shu and Zhenhua Wu, A Hybrid Real-Time Scheduling Approach on Multi-Core Architectures, JOURNAL OF SOFTWARE, VOL. 5, NO. 9, SEPTEMBER 2010, pp 958-965.
11 R.I. Davis, A. Burns,‖A survey of hard real-time scheduling for multiprocessor systems‖, ACM Comput. Surv. 43 (4) (2011) (Article 35).
12 Agrawal, S., Yadav, R. S. and Ranvijay. ―A Pre- emption Control Approach for Energy Aware Fault Tolerant Real Time System‖, International Journal of Recent Trends in Engineering, 381-386, 2009.
13 Y. Zhang, K. Chakrabarty,‖A unified approach for fault tolerance and dynamic power management in fixed-priority real-time embedded systems‖, IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst. 25 (1) (2006) 111–125.
14 Bertogna, M. and Baruah, S., ―Limited Preemption EDF Scheduling of Sporadic Task Systems‖, IEEE Transactions on Industrial Informatics, 579 – 591, 2010.
15 Dehghan and Maryam, 2010. ―Adaptive checkpoint placement in energy harvesting real-time systems‖, 18th Iranian Conference on Electrical Engineering (ICEE), 932 - 937.
16 Mehmet Can Kurt, SriramKrishnamoorthy, Kunal Agrawal and Gagan Agrawal, ―Fault-Tolerant Dynamic Task Graph Scheduling‖, SC14, November 16-21, 2014, New Orleans IEEE, 2014.
17 Md. Mamunur Rashid and Md. Nasim Adhtar, ― A New Multilevel CPU Scheduling Algorithm, Journals of Applied Sciences 6 (9): 2036-2039, 2009
18 Silberschatz, A. P.B. Galvin and G. Gagne (2012), Operating System concepts, 8th edition,Wiley India,
19 Sabrian, F., C.D. Nguyen, S. Jha, D. Platt and F.Safaei, (2005). Processing resource scheduling inprogrammable networks.Paper ID: J2013200 133 of 134 ISSN (Online): 2347-3878 Volume 2 Issue 3,

March 2014
20 Umar Saleem and Muhammad Younus Javed, Simulation of CPU Scheduling Alogrithm,0-7803-6355-8/00/IEEE
21 Sun Huajin', Gao Deyuan, Zhang Shengbing, Wang Danghui, “Design fast Round Robin Scheduler in FPGA‖, 0-7803-7547-5/021/2002 IEEE