***Transaction-Based Models (TBM) and Evaluation of their throughput***

*Kushal K S*

Aerospace Electronics & Systems Division,

CSIR-National Aerospace Laboratories,

Bengaluru, India

ksk261188@gmail.com

*Dr. Manju Nanda*

Aerospace Electronics & Systems

Division,

CSIR - National Aerospace Laboratories,

Bengaluru, India,

[manjun@nal.res.in](mailto:manjun@nal.res.in)

*J Jayanthi*

Aerospace Electronics & Systems

Division,

CSIR - National Aerospace Laboratories,

Bengaluru, India,

jayanthi@nal.res.in

*Abstract*- The perception of the system-level design, based on a modeling approach involving series of asynchronous events, for the preferred system specifications, articulates as a befitting solution. These models highlight approaches involved in developing heterogeneous complex embedded systems, with formal foundations. These models composed with distinct components, representing the behavioral integrant and calculative modes for the system. Such a modeling approach proves to be efficient in contending the possible clogging bottlenecks in the design and development cycle. This approach and analytical methodology uniquely strategizes the references for various abstraction levels in the development-cycle as in; early software development associated with the system, architecture analysis, functional & performance verification.

In this paper we propose a novel transaction-based high-level modeling approach in verifying and validating the architecture, as well as evaluating the performance of a system suitable for avionics application. The novel approach is proven by means of a case study of the avionics software scheduler. VisualSim is used for transaction-based evaluation and provides an emulative platform providing a combinatorial link between the hardware and its underlying software, deriving a more realistic solution for the feature of development.

Keywords - Transaction-Based Modeling, Software Development Life-Cycle, Transition-Level Models, Scheduling Algorithms.

# Introduction

Advances in the heterogeneity and the complexity of the advanced applicative embedded systems in the avionics and the aerospace domain, such systems are more prone to be software intensive and hardware dependent. Their dependencies determine the performance and efficiency in their functional and operational capabilities. This methodology of integrating such commercially computing systems poses as a major challenge, as well in the development of such modern applicative systems. A predictive practice that feature as an explication, a well known high-level or a system-level modeling, i.e. Transaction-based modeling. The transaction-based modeling along with the formal foundations have been proven to provide with certain systematic analyses of modeled systems, wherein the details of the event-based communications are abstracted from the features of the implementation properties associated with such units. This particular methodology is engrossed with the low-level details of the information exchange, with the occurrence of events happening between the controller unit and the modules of interest. The integrated modular avionics system architecture consists of federated resources, being integrated on a single platform for certain applicative and computational purposes. Such modules also seem to have been distributed over for a variety of applications on a single integrated hardware platform. This requires an extensive and an exclusive formulation in their performance at each level of abstraction and the evaluation of their capabilities, increasing their efficiencies in their throughput.

The formal representations of such modern applicative complex embedded systems, augmented by full-scale physical simulations are to be modeled suitably. These formal foundations aids in addressing the design interests at suitable abstracts of Software Development Life-Cycle (SDLC). The context of the transaction-based models are underlined with their software, hardware and the physical system components, their interactions and the properties of these elements. Formally verifying and analysing the critical system factors associated with these components within a suitable software packaged engineering environment is estimated. This imbibes an elated confidence in the design and development of such complex systems, with abbreviated maintenance, early testimony in the clogging factors that might be involved in the development phase, more concise time-to-market and outlay monitoring, and subsidized certification procedure.

In this paper we propose a unique and an interpretive approach in performing the evaluation of multi-threaded embedded system architecture suitable for avionics application, constraining the processor size to support for the parallel tasks to run efficiently and concurrently on a single processor and its execution. The paper focuses on the evaluation of the effect of the scheduling multiple concurrent tasks by identifying the right scheduling algorithm. Evaluation of the priority on tasks determining the impact on the response time for high priority and low priority tasks; evaluation of the effect of pre-emption on the executing tasks and their impact, with the generation of statistics and data to plot to provincially substantiate the impact of priority on the order of execution and the way in which the processing time affects the scheduler buffering, is also exclaimed in this paper.

The paper is organized as follows: Section II discussing about the related works for transaction-based modeling, while Section III briefly introduces the augmented methodology involved in designing the Scheduler Model. Section IV discusses the substantiating data and plots generated. Section V gives a conclusive note for the methodology and the purview of the methodology it can be expressed into, for further analytical procedures.

# Related Works

In the organization of modern processors and their depending digital systems, the atomic guard/action transitions in modeling such hardware architectures, conceptualizing the clarity in modelling such systems is popular and is drastically playing a major role in development of suitable notations. This ideology was proposed by Xiaofang Chen et.al [1]. They also proposed a verification methodology in verifying the high-level behavior of the hardware. Their paper makes a contribution in certain specific areas such as creating sitable notations allowing the designer to specify the designs embed with aggressively optimized implementations, theories to formally relate these implementations against the given set of specifications, and the methods of composition, verified at the model-level for global properties. In this paper they have successfully employed an extended version of the HMurphi (Hardware Murphi) in model implementations and have verified the specifications at the high-level interleaving transitions for global properties. The organization of the approaches involved in the vibrant area of transition-level modeling, being widely referred to system-level design methodologies, with the main instinct of extracting the characteristics of the systems from a low-level transition-level modeling approaches and annotating them with the high-level approaches, in making the accurate design decisions at an early stage was proposed by Lukai Cai et.al [2]. In their paper they had also proposed that using the defined Transition-Level Models (TLM), as standard models, implementation of the cross-approach designs and the reuase of the implemented and formally verified TLMs can be made extensively.

A paper well before the definition of various approaches involved in the TLMs, was published with the proposal and an inferable note on transition-level modeling, referring to the system-level designs was coined by Lukai Cai et.al[3]. The not so well defined status of the TLMs and their extensive usage across various domains such as modeling, synthesis, verification & validation, and refinement, which were not so well co-ordinate, was put-forth. This paper also introduces to the various TLM taxonomies and the involving benefits adhered with the usage of TLMs.

With the proposal of several modeling abstraction levels, an improvement with respect to the simulation speed and the modeling time involved in all these proposed levels of modeling abstractions, across the much detailed cycle-accurate (CA) models, was detailed in the paper proposed by University of California-Irvine, from the Office of Technology Alliances[4]. In this paper a modeling abstraction which could successfully maintain the cycle-accuracy at the boundary of every cycle while communicating, in a system, with the capture of all the component interfaces at the pins termed as Pin Accurate Bus Cycle Accurate (PA-BCA) was introduced. Along with this, introduction of a new modeling abstraction which used the concept of transactions, in-order to speed-up the modeling process and the simulation time known as Transaction-based Bus Cycle Accurate (T-BCA) was made.

Over the substantially proven high-level modeling languages like SystemC, transaction-based modeling is adorned with the feature of emulating the system-of-interest thereby, proving its prominence in modern complex avionics embedded system developmental approaches The description of the proposed technology being the paltry in defining the high-level models of abstraction of the system, are fast to simulate and thus was considered as a substratum for the assimilated technology in the scenario considered in this paper.

# Methodology & Case Study

The methodology adopted in this paper is follows;

1. Model the system based on the functionality. The various mechanisms the system can be modeled are context switch, mulit-threaded, and multi-core mechanism.
2. Implement the model as per the selected mechanism
3. Execute the model and analyze the model for data statistics and simulation plots.
4. Compare this statistics and simulation plots against the system specifications.

In the case study we model and analyze the avionics real-time application. For modeling, the multi-threaded mechanism is adopted and the emulation results are analyzed.

A multi-threading environment and multi-core processors are very efficient methodologies in designing the real-time systems, which has a collection of many tasks operating independently, concurrently and communicate, if necessary, with each other. Such an efficient methodology is highly recommended for avionics and aerospace applications. The high-level transaction-based models breaks the entire system design down from a singular large monolithic block of code and design into a concurrent series of convenient tasks, greatly simplifying the design and as well speeding up the phenomenon of the product development.

Most of the avionics/aerospace systems are entrusted with multi-threaded design approach. In this paper the system under consideration is one precedent, with a collection of concurrent tasks. These systems are leveraged than a system with a unary program, as;

1. Multiple tasks can be designed individually. They can also be implemented, tested and verified individually for their integrity, even upon the specification of their dependency.
2. With the avionics application as the objective, concurrency of the tasks ensures an easy framework for the specification of the application affirmatively.

The system under consideration in this paper is expressed as with a single processor architecture system and a collection of three tasks. The single processor under consideration for the evaluation is PowerPC 7XX from Motorola clocked at 500MHz and 1GHz operating frequencies. In this, we also assume that the tasks/threads are scheduled at the instruction level, as it enables the concurrent tasks/threads to operate in real-time. These threads/tasks are implemented/scheduled on a uni-processor with the context switching mechanism.

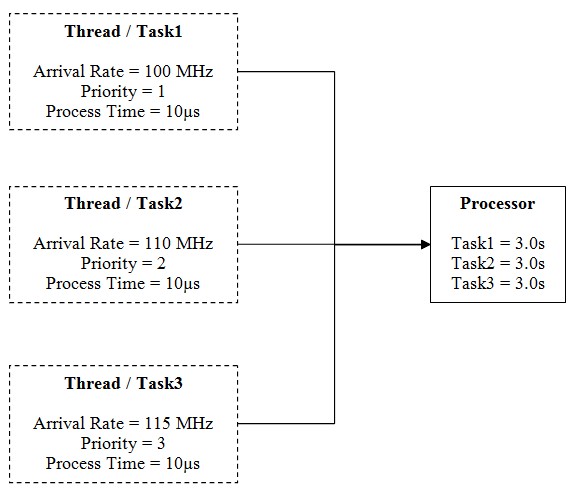


Figure Block Diagram of the Proposed System

The architecture in the above does not consider any logic in communicating the message passing mechanism, and we only consider the scheduling policy in providing the service to the thread/task. This particular case study is aimed at sizing the processor resource architecture so as to support the concurrent tasks/threads running on it. As in , we can visualize that the system-of-interest contains three concurrent tasks/threads that need to be executed on the single PowerPC 7XX processor. Each task/thread in the system-of-interest has got different arrival rate, and priority, which is as follows;

1. Thread/Task1

Arrival Rate = 100MHz

Priority = 1

Process Time = 10µs

1. Thread/Task2

Arrival Rate = 110MHz

Priority = 2

Process Time = 10µs

1. Thread/Task3

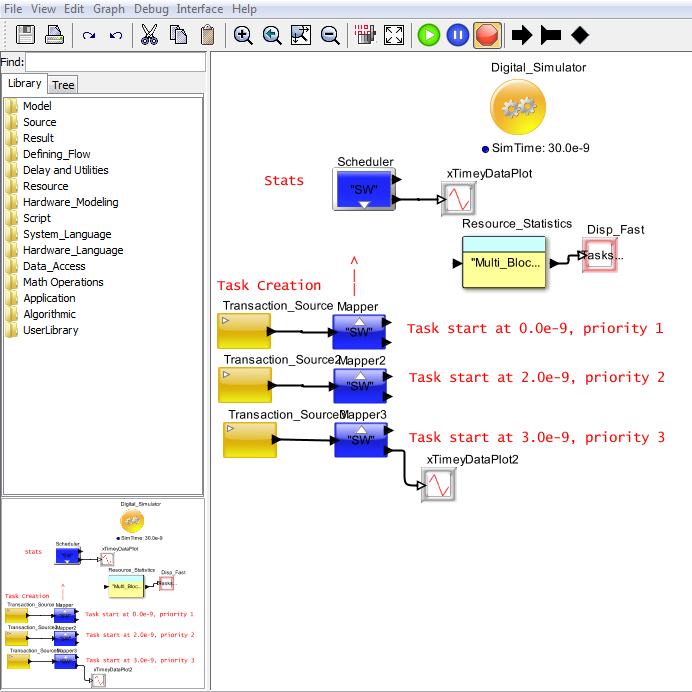
Arrival Rate = 115MHz

Priority = 3

Process Time = 10µs

The arrival time symbolizes the time unit at which the particular thread/task arrives to the scheduler, while it is supposedly marked up with the priority. The process time signifies the time duration for which the respective thread/task remains in the scheduler being serviced by the processor. This entire system architecture, definition of the model comprising multiple concurrent tasks or flows, definition of the shared resource unit such as scheduler in this scenario and mapping of the behavior of the target architecture is carried out suitably using the formal foundations available with a software package such as VisualSim.

VisualSim [14] is a modeling and simulation software package which is used to optimize, validate and evaluate the specific system-of-interest and also in verifying the implementation, generating suitable artifacts for early marketing. This software package uses a highly optimized simulator in evaluating the performance, power and functional trade-offs. The system with multi-threaded design is modeled in the VisualSim environment as block diagram in the (Block-Diagram Editor (BDE). The block diagram for the system-of-interest is as modeled in the VisualSim environment as shown in .



**7.**

**6.**

**5.**

**4.**

**3.**

**2.**

**1.**

Figure Block Diagram modeled in VisualSim BDE

The block diagram modeled, as in , consists of certain modeling blocks/components used to build the model, which are as;

1. Digital Simulator: a block used to model the protocols, hardware and mapping behavior to the desired architecture. This can be used to model either triggered or time based models.
2. Transaction Source: a block which outputs the Data Structure (DS) at a suitable time interval specified in the time distribution constraint.
3. Mapper: a block which is timed resource unit that combines a single input queue and a processing resource such as a server. This timed resource utilizes the time units in emulating the processing delay across an entity.
4. Time Plotter: a block which plots the incoming data on the Y-axis with the current simulation time on X-axis. Each connection to this block is considered as a separate dataset and is used to plot separately.
5. Resource Statistics Block: a block which is pre-built in the VisualSim environment and placed in a model to output the simulated statistics for all the Schedulers, Timed Resources in the model.
6. Text Display: a block which displays the arriving values on their input port within a textual display dialog. It also buffers the display data and updates with the new updated data when the buffer is full.

This paper focuses on building a scheduling model in a suitable software package environment such as VisualSim in order to evaluate the performance of targeting multiple tasks to a single scheduler as in for suitable safety-critical systems for avionics applications.

# Statistics Data and Resulting Plots

The analytical features include the evaluation of the effect of scheduling the three threads/tasks by identifying the right scheduling algorithms, such as;

1. FCFS
2. FCFS+Preempt
3. Scheduling\_RR

The scheduling algorithm First Come, First Served (FCFS), also called First In First Out (FIFO), in which the processor processes in the order of the task/thread arrival into the processing unit. In this particular scheduling policy the processes run until the processing unit is completed exhausted with the service to all the threads/tasks. It can be associated with the preemption. If this policy is non-preemptive, then none of the tasks/threads are ever blocked until the processor unit completely exhausts. This policy of non-preemptive policy resist to any changes being made in the policy of the Scheduler.

The FCFS policy is easy to implement, analyze and evaluate. This policy also proves to be more efficient as it minimizes the context switching. The clock rate of the processing unit is not considered as these policies are executed under the consideration of the Relative Time of the scheduler. The simulated results and the parameter settings for this policy are plotted as shown in and .

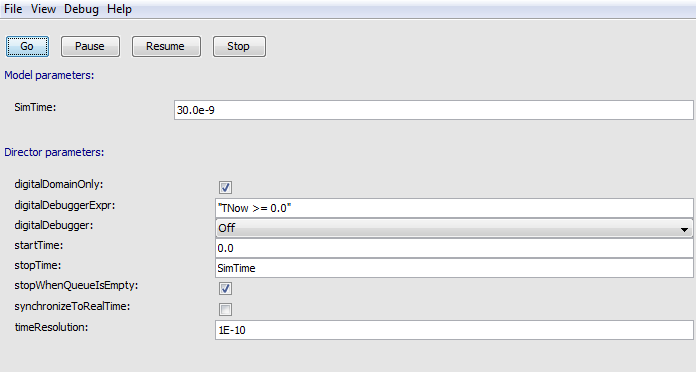


Figure Model Parameters and Director Parameters for FCFS Scheduling Policy

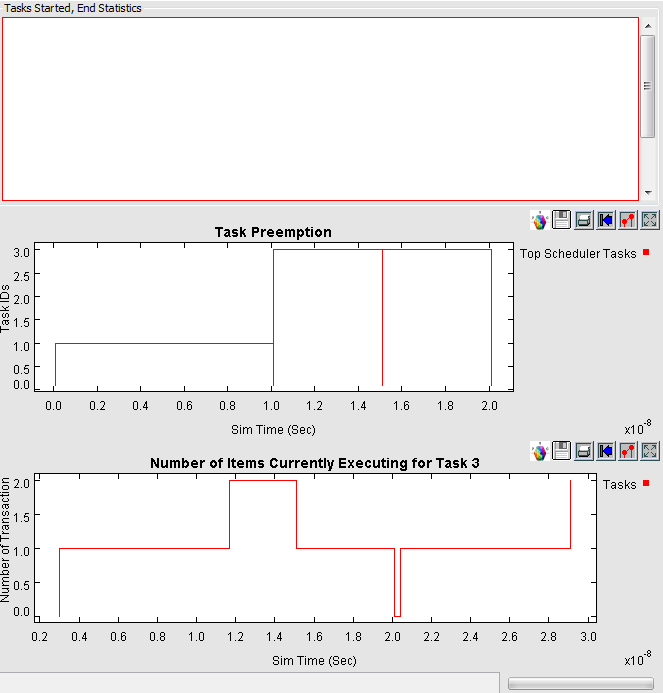


Figure FCFS Scheduling Policy Simulation Plots

For the above provided analyses, the resultant data generated is as given below.

DISPLAY AT TIME ------ 30.00 ns ------

{BLOCK = "Gen\_Lab3.SW",

DELTA = 0.0,

DS\_NAME = "Queue\_Common\_Stats",

ID = 1,

INDEX = 0,

Number\_Entered = 11,

Number\_Exited = 3,

Number\_Rejected = 0,

Occupancy\_Max = 8.0,

Occupancy\_Mean = 3.8823529411765,

Occupancy\_Min = 1.0,

Occupancy\_StDev = 1.8111534492165,

Queue\_Number = 1,

TIME = 3.0E-8,

Total\_Delay\_Max = 1.21E-8,

Total\_Delay\_Mean = 1.0166666666667E-8,

Total\_Delay\_Min = 8.4E-9,

Total\_Delay\_StDev = 1.5151090903151E-9,

Utilization\_Mean = 66.6666666666667}

With certain short-falls surrounding FCFS such as the penalty ratio for short jobs and an impossible way to break out of an infinite loop FCFS with preemption is incorporated and the resultant results with generated data and plots is as given below and as shown in .

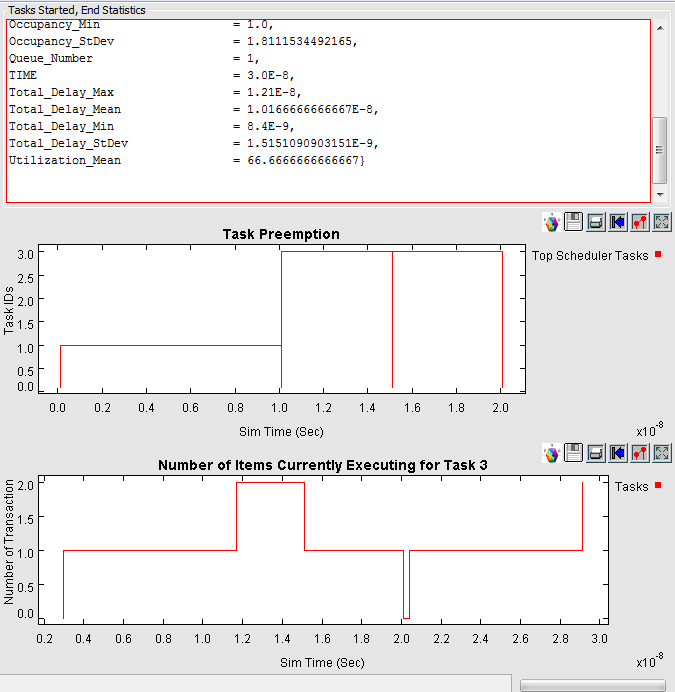


Figure FCFS+Preempt Scheduling Policy Simulation Plots

DISPLAY AT TIME ------ 30.00 ns ------

{BLOCK = "Gen\_Lab3.SW",

DELTA = 0.0,

DS\_NAME = "Queue\_Common\_Stats",

ID = 1,

INDEX = 0,

Number\_Entered = 11,

Number\_Exited = 3,

Number\_Rejected = 0,

Occupancy\_Max = 8.0,

Occupancy\_Mean = 4.5,

Occupancy\_Min = 1.0,

Occupancy\_StDev = 2.0169734302122,

Queue\_Number = 1,

TIME = 3.0E-8,

Total\_Delay\_Max = 5.0E-9,

Total\_Delay\_Mean = 5.0E-9,

Total\_Delay\_Min = 5.0E-9,

Total\_Delay\_StDev = 0.0,

Utilization\_Mean = 50.0}

We can evaluate from the retrieved data that the mean utilization time appears to be decremented with FCFS+Preempt. Further we analyze the scenario with Round Robin scheduling policy. Under this scheduling policy the processing unit services a single process for only a single quantum of time unit, q, known as Time Slice. The considerations are being made as 2\*10-9 and 4\*10-9, before moving onto the next task/thread. This scheduling policy is expected to achieve good penalty ratio by preempting processes that are monopolizing the processing unit. The tuning of the parameter is required as their short amounts results in an approximation resulting in the processor sharing, wherein every process is provided with constant service from the processing unit which is proportional to the number of processes. Here the time allocated is also 3.0s for 3 processes. The simulated results for the different quantum parameters and the resultant data is as shown in , & , and as given below, respectively.

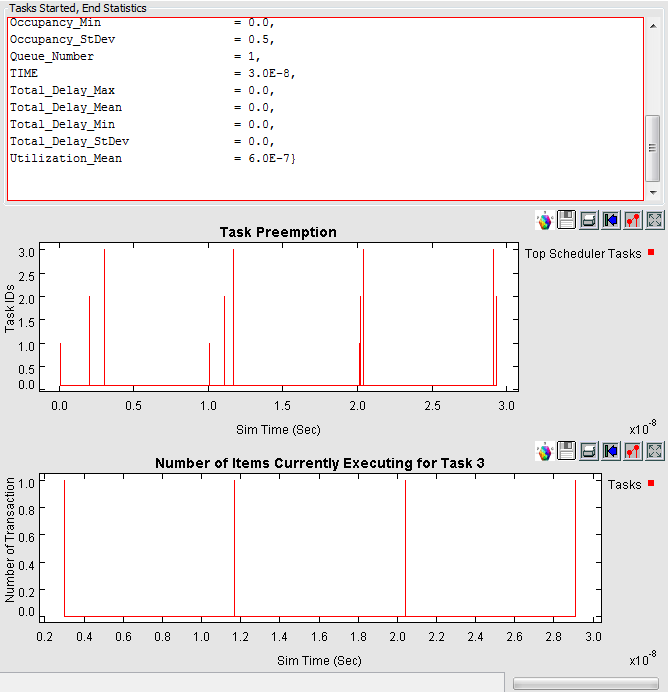


Figure Round Robin Scheduling Policy Simulation Plots

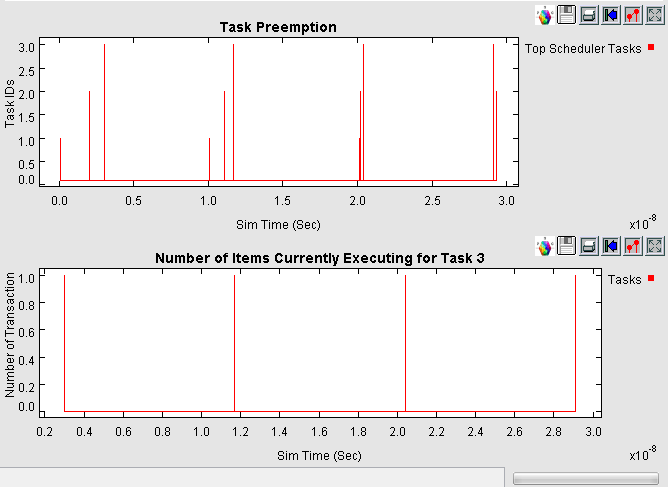


Figure Round Robin Scheduling Policy with Time Slice of 2\*10-9 Simulation Plots

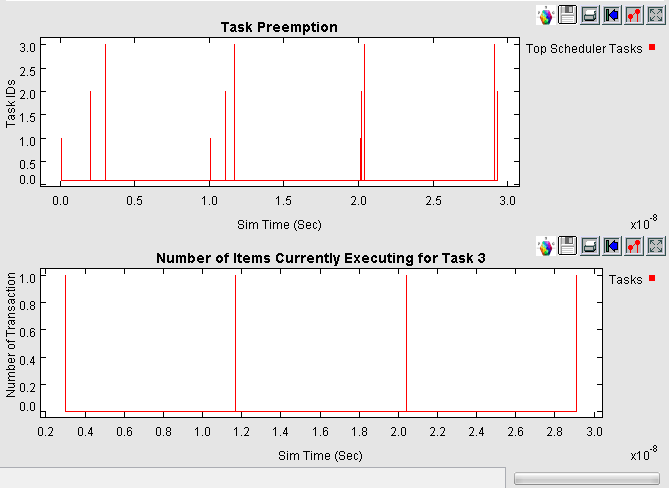


Figure Round Robin Scheduling Policy with Time Slice of 4\*10-9 Simulation Plots

DISPLAY AT TIME ------ 30.00 ns ------

{BLOCK = "Gen\_Lab3.SW",

DELTA = 0.0,

DS\_NAME = "Queue\_Common\_Stats",

ID = 1,

INDEX = 0,

Number\_Entered = 11,

Number\_Exited = 3,

Number\_Rejected = 0,

Occupancy\_Max = 8.0,

Occupancy\_Mean = 4.4117647058824,

Occupancy\_Min = 1.0,

Occupancy\_StDev = 2.0017293561381,

Queue\_Number = 1,

TIME = 3.0E-8,

Total\_Delay\_Max = 2.21E-8,

Total\_Delay\_Mean = 1.6733333333333E-8,

Total\_Delay\_Min = 1.0E-8,

Total\_Delay\_StDev = 5.0334437073991E-9,

Utilization\_Mean = 83.3333333333333}

**Data with Time\_Slice of 4\*10-9:**

DISPLAY AT TIME ------ 30.00 ns ------

{BLOCK = "Gen\_Lab3.SW",

DELTA = 0.0,

DS\_NAME = "Queue\_Common\_Stats",

ID = 1,

INDEX = 0,

Number\_Entered = 11,

Number\_Exited = 11,

Number\_Rejected = 0,

Occupancy\_Max = 1.0,

Occupancy\_Mean = 0.5,

Occupancy\_Min = 0.0,

Occupancy\_StDev = 0.5,

Queue\_Number = 1,

TIME = 3.0E-8,

Total\_Delay\_Max = 0.0,

Total\_Delay\_Mean = 0.0,

Total\_Delay\_Min = 0.0,

Total\_Delay\_StDev = 0.0,

Utilization\_Mean = 6.0E-7}

The Relative Time scenario is superceeded with the Number of Clocks consideration in evaluating the performace of the system transaction, in which the Clock Rate is to be considered suitably and is verified for two conditions as of with 500MHz and 1GHz for the PowerPC 7XX processor architecture. The resultant plots and the suitable data obtained are as shown in & , and as given below.

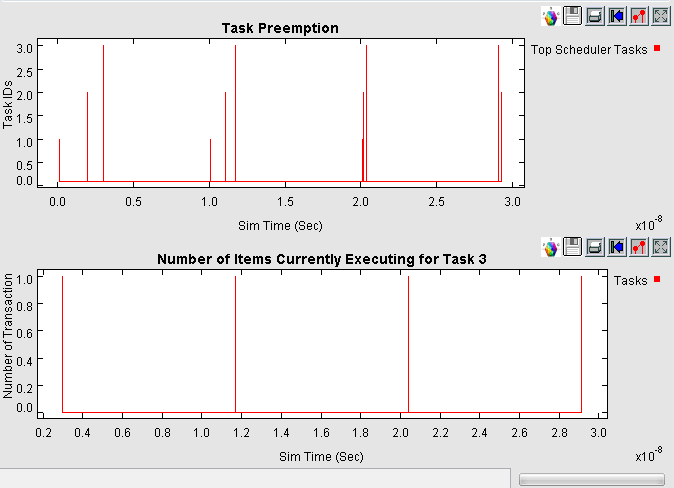


Figure RR Scheduling Policy based on the Number of Clocks parameter with 500MHz Clock Rate

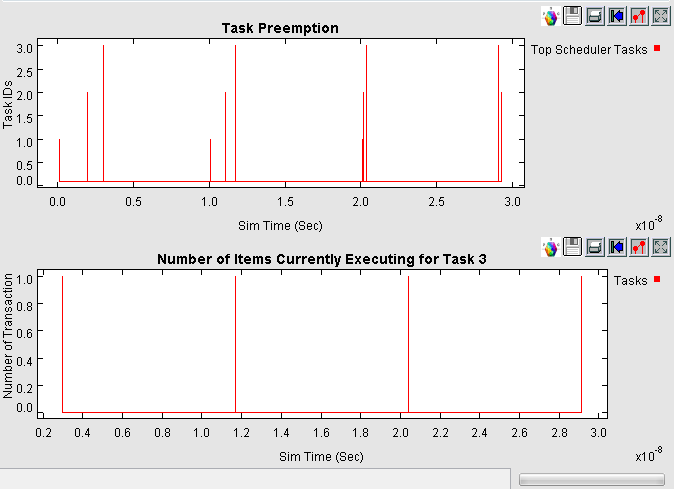


Figure RR Scheduling Policy based on the Number of Clocks parameter with 1GHz Clock Rate

DISPLAY AT TIME ------ 30.00 ns ------

{DELTA = 0.0,

DS\_NAME = "Queue\_Common\_Stats",

ID = 1,

INDEX = 0,

Number\_Entered = 11,

Number\_Exited = 11,

Number\_Rejected = 0,

Occupancy\_Max = 1.0,

Occupancy\_Mean = 0.5,

Occupancy\_Min = 0.0,

Occupancy\_StDev = 0.5,

Queue\_Number = 1,

TIME = 3.0E-8,

Total\_Delay\_Max = 0.0,

Total\_Delay\_Mean = 0.0,

Total\_Delay\_Min = 0.0,

Total\_Delay\_StDev = 0.0,

Utilization\_Mean = 6.0E-7}

The relative time plots for FCFS, FCFS+Preempt and Round Robin, as well with the number of clocks parameter for 1GHz clock rate is given below, as shown in , , , and .

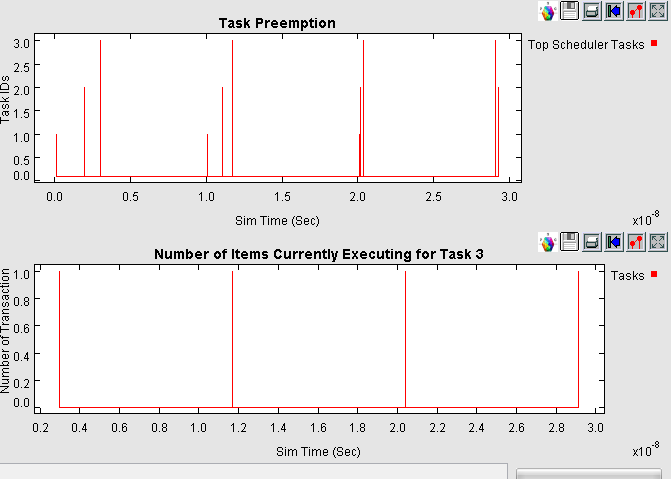


Figure FCFS Scheduling Policy with Number\_of\_Clock Simulation Plot

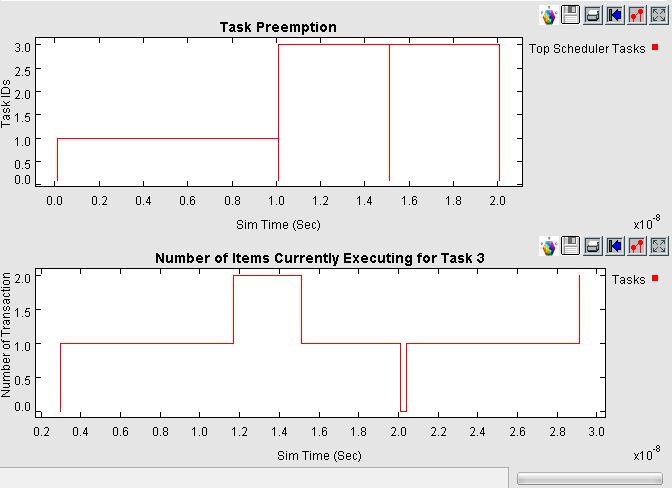


Figure FCFS Scheduling Policy with Relative Time Simulation Plot

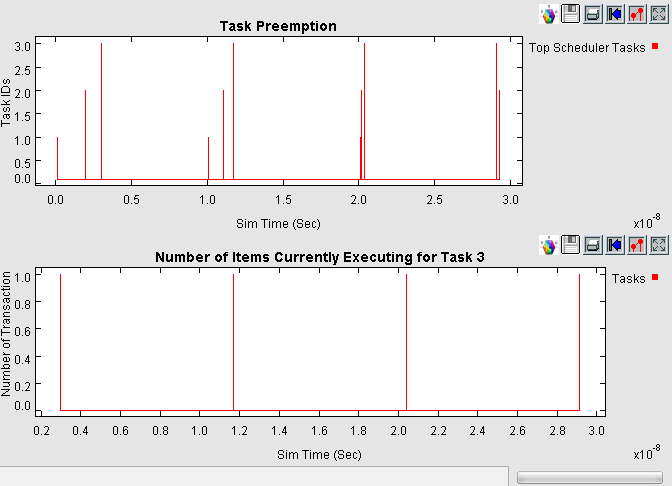


Figure FCFS+Preempt Scheduling Policy with Number\_of\_Cycles Simulation Plot

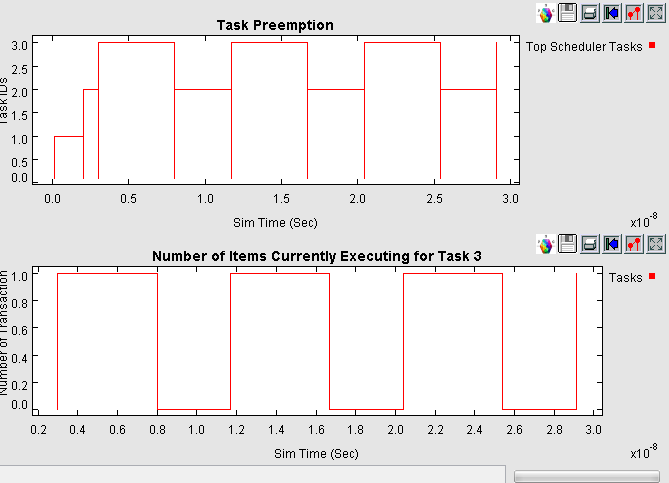


Figure FCFS+Preempt Scheduling Policy with Relative Time Simulation Plot

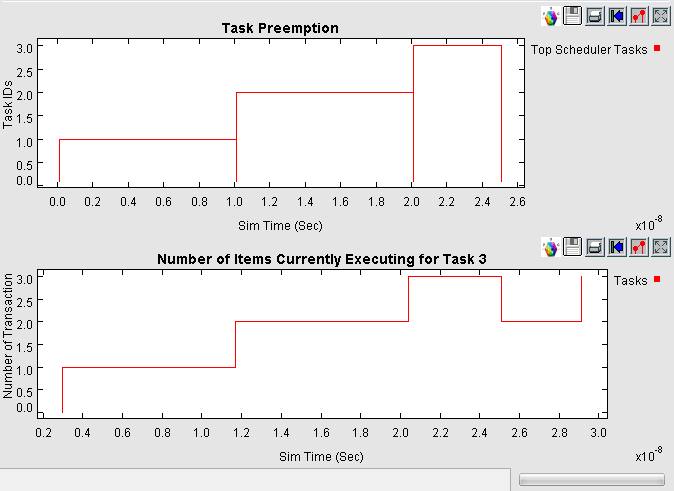


Figure RR Scheduling Policy with Relative Time Simulation Plot

From all these informative plots and data statistics we analyze and evaluate the transaction-based model such as Software Scheduler unit suitably and decide with accuracy and desired design by choosing an appropriate scheduling policy necessary for the system, for its divergent avionics application.

# Conclusion

This paper proposes a novel approach in evaluating the throughput of the transaction-based models with respect to concurrently triggered tasks/threads using a suitable software package such as VisualSim. Evaluation was carried out and the system was validated with a substantially efficient scenario subjective to avionics and aerospace application and categorized as a functional basis unit of safety-critical systems. Suitable formal foundations in conjunction with the modeling parameters were incorporated. The simulation plots and the statistical data generated, aids in the perceptive conceptualization of the system-of-interest, unraveling the clogging factors that might be involved in the development life-cycle, as well validate the designed model with the interleaving specifications providing a benchmark for the engineering process. This approach can suitably be applied across in modeling various such modern applicative complex embedded systems, as well in validating the models, against the specifications, formally.

##### ACKNOWLEDGMENT

The authors would like to thank the Director, CSIR – NAL, Bengaluru for supporting this work.

##### REFERENCES

1. Xiaofang Chen, Steven M German, Ganesh Gopalakrishnan, “Transaction Based Modeling and Verification of Hardware Protocols”, Seventh International Conference on Formal Methods in Computer-Aided Design, IEEE Proceedings, pp. 53-61, 2007, ISBN: 0-7695-3023-0/07, DOI: 10.1109/.20.
2. Lukai Cai, Daniel Gajski, “Transaction Level Modeling: An Overview”, CODES+ISSS’03, ACM 1-58113-742-7/03/0010, ACM Proceedings, October 2003.
3. Lukai Cai, Daniel Gajski, “Transaction Level Modeling in System Level Design”, CECS Technical Report 03-10, pp. 01-09, March 2003.
4. “Efficient Transaction Based Modeling with Cycle Count Accurate at Transaction Boundary (CCATB) Models”, Office of Technology Alliances, University of California-Irvine, Tech ID: 18822/ UC Case 2005-288-0.
5. Zhenghua Fu, Nalini Venkatasubramanian, “AN Evaluation of Composite Routing and Scheduling Policies for Dynamic Multimedia Environments”, ACM Proceedings, pp. 01-20, 2004.
6. “Transaction Based Emulation”, by Synopsys,

<https://www.synopsys.com/Tools/Verification/hardware-verification/emulation/Pages/transaction-level-modeling.aspx>

1. Wolfgang Klingauf, “Systematic Transaction Level Modeling of Embedded Systems with SystemC”, DATES’05, Proceedings of the Conference on Design, Automation and Test in Europe, Vol. 1, pp. 566-567, ISBN: 0-7695-2288-2.
2. Amine Anane, El Mostapha Aboulhamid, “A Transaction-Based Environment for System Modeling and Parallel Simulation”, International Journal on Parallel Programming, Vol. 43, Issue. 1, pp. 24-58, ISSN: 1573-7640.
3. Richard Buchmann, Mickael Cartron, Yannick Bonhomme, “Transaction-based modeling for large scale simulations of heterogeneous system”, SIMUTOOLS’09, Proceedings of the 2nd International Conference on Simulation Tools and Techniques, Article No. 33, Belgium, 2009, ISBN: 978-963-9799-45-5.
4. Anthony Barrteau, Sebastien Le Nours, Olivier Pasquier, “A State-Based Modeling Approach for Efficient Performance Evaluation of Embedded System Architectures at Transaction Level”, Journal Of Electrical and Computer Engineering, Vol. 2012, Article ID 537327, June 2011.
5. Andrew Pavlo, Evan P C Jones, Stanley Zdonik, “On Predictive Modeling for Otimizing Transaction Execution in Parallel OLTP Systems”, Proceedings of the VLDB Endowment, Vol. 5, No. 2, pp. 85-96, Istnabul, Turkey, August 2011, ISBN: 2150-8097/11/10.
6. Dominique Borrione, “Advances in Design Methods from Modeling Languages for Embedded Systems and SoC’s”, Springer.
7. Toczek T, Houzet D, Mancini S, “Another take on functional system-level Design and Modeling”, Forum on Specification & Design Languages, FDL’09, pp. 01-06, IEEE, ISBN: 978-2-9530504-1-7.
8. [www.mirabilisdesign.com/new/visualsim/](http://www.mirabilisdesign.com/new/visualsim/)