International Journal of Engineering Research and Technology.
ISSN 0974-3154 Volume 11, Number 7 (2018), pp. 1119-1128
© International Research Publication House
http://www.irphouse.com

Optimization of Simulation based System Level Modeling to Enhance Embedded Systems Performance

Murigendrayya M. Hiremath [1], Dr. A.R. Aswatha[2], Dr. Thangadurai.N[3]

[1] Research Scholar, Department of Electrical and Electronics Engineering, JAIN (Deemed-to-be-University), Bangalore, Karnataka, India.

[2] Professor and Head, Telecommunication Engineering, DSCE, Bangalore, Karnataka, India.

[3]Professor and Head, Department of Electronics and Communication Engineering, School of Engineering and Technology, JAIN (Deemed-to-be- University), Bangalore, Karnataka, India.

**Corresponding author*

Abstract

Simulation based system level modeling is better approach than analytical and concrete level modeling because it is more accurate than analytical modeling and faster than concrete level modeling. Long learning curve is the challenge of simulation-based system-level modeling. System engineer either has to study all available pre-built blocks or has to develop custom models. Irrelevant modeling components and templates selection can lead to more modeling effort. The proposed approach aims to optimize system-level modeling effort by exploring the systematic modeling approach and demonstrating the technique to generate guidelines report from system specifications. In this approach System Level Performance Modeling of Magnetic Resonance Imaging (MRI) Reconstruction algorithm processing on Graphical Processing Unit (CPU-GPU) architecture is considered as case study.

Keywords: Prebuilt, Modeling, Embedded design, Latency, Utilization.

1. INTRODUCTION

Performance metrics such as latency and utilization are important embedded system design metrics to decide the right hardware and software components of embedded systems. Unsuitable selection of design components leads to a number of implementations, more engineering cost and more time to market [1]. Latency prediction and analysis shows system speed for different input conditions and helps to identify the system performance. Utilization indicates whether hardware resources such as processor, memory and bus etc. are over-utilized or under-utilized [4]. Over utilization leads to system failure and under-utilization leads to more cost. System latency of simple system can be represented using equation 1. System latency is a function of task processing time and memory access time.

$$t_1 = (T_j + TM_j)\forall_j \in P \qquad \textit{Eq (1)}$$

Where

$t_1 \rightarrow$ *System latency of simple system*

$T_j \rightarrow$ *Task processing time mapped with resource*

$TM_j \rightarrow$ *Memory access time*

$P \rightarrow$ *Set of components*

Latency of complex systems with multi task and multiple resources also depends on task waiting time because among multiple tasks requests for the resource one task will able access resource and other tasks are waiting in the queue till release of current processing task. Hence equation 1 holds good for simple system. Equation 2 shows that system latency is also a function of task waiting time. Task waiting time computation using analytical approach is not accurate because modeling of unpredictable arrival of tasks and input conditions cannot be possible using a series of mathematical equations.

$$t_2 = (T_j + TM_j + TW_j), \forall j \in P \qquad \textit{Eq (2)}$$

Where

$t_2 \rightarrow$ *System latency with considering* TW_j

$TW_j \rightarrow$ *Total waiting time to access resource*

Simulation is the best approach for accurate performance analysis in which unpredictable arrival of tasks and input conditions can be randomized [2][8]. Discrete event simulator is used for performance modeling and analysis of embedded systems. During this simulation, model is triggered by every event of time and the result, of each triggering is recorded. Performance modeling involves modeling of traffic inputs, system behavior and resources used [5]. System behavior represents the set of tasks processed by resources. Prebuilt modeling component used for performance modeling is queue with a delay which indicates that multiple tasks are waiting in the resource queue to access the resource.

In simulation based system level modeling environment certain challenges remain to be addressed. Due to lack of information long learning is required to understand the modeling components and frameworks. Irrelevant modeling components and frameworks selection leads to inaccurate simulation results. Equation 3 shows the factor influences the modeling effort.

$$\mathrm{m} = \mathrm{f}(l, o, d, s, c) \qquad \textit{Eq (3)}$$

Where

m → modeling effort

l → learning curve

o → objective defining effort

d → effort of data collection

s → effort of modeling component selection

b → effort of building system model

Modeling effort can be optimized by increasing number of frame works prebuilt modeling components and system level modeling approach as specified in equation 3 and 4 [10][11].

$$\mathrm{l}, \mathrm{s}, \mathrm{b} \propto (f, p, a) \qquad \textit{Eq (4)}$$

Where

f → number frameworks

p → number of prebuilt components

a → approach of system level modeling

Proposed work explore the enhanced system level modeling approach. This approach includes generation of guidelines to build the system model for given modeling objectives and system details. Performance Modeling of MRI reconstruction algorithm on GPU platform is considered as case study to demonstrate the proposed approach.

2. METHODOLOGY

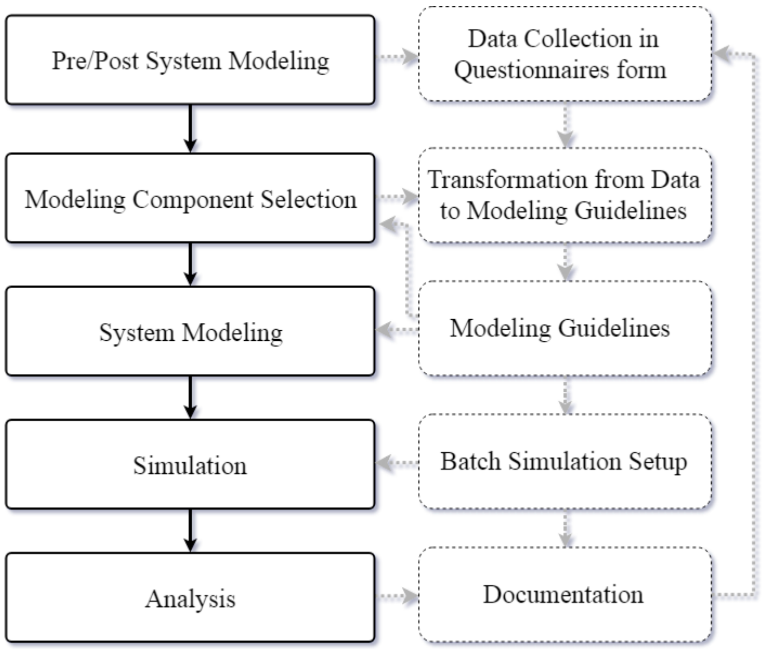

Pre/Post System Modeling
Modeling Component Selection
System Modeling
Simulation
Analysis
Data Collection in Questionnaires form
Transformation from Data to Modeling Guidelines
Modeling Guidelines
Batch Simulation Setup
Documentation

Figure 1: Enhanced system level modeling

Figure 1 shows the enhanced System level modeling approach. Modeling objectives and modeling related data collection is done in pre/post system modeling stage. Based on modeling objectives and available modeling information prebuilt modeling components are [3]. Complete system model is built by using VisualSim modeling library. System modeling includes work load modeling, behavior modeling and resource modeling. Workload model is the stimulus to the system model which represents the arrival of data as traffic with defined rate.

Behavior describes the order and dependency of tasks processed on the data and contains timing details for performance modeling. Resource modeling involves modeling resources such as processor, memory, and bus etc. These resources are modeled specific to performance analysis. Discrete event simulator of Ptolemy is used for simulation. Experimental setup of configurations and conditions are made for simulation runs to generate simulation results. Performance analysis is done by understanding the variations in performance with respect to the input conditions and configurations. Based on analysis results, design specification with viable configurations and conditions are decided for further implementation. Flow with the dotted line specifies the enhanced approach through which the modeling guidelines generated by taking collected modeling details. This reduces the effort of learning, selecting and modeling effort. In enhanced approach initial modeling data collected is in questioner form. On the other side these questions are included in enhanced approach based on available frameworks and prebuilt modeling components.

3. ENHANCED APPROACH

Performance modeling of GPU based MRI reconstruction algorithm is considered to demonstrate enhanced simulation based system level modeling approach. The questionnaire for data collection in enhanced approach is categorized as shown in Figure 2.

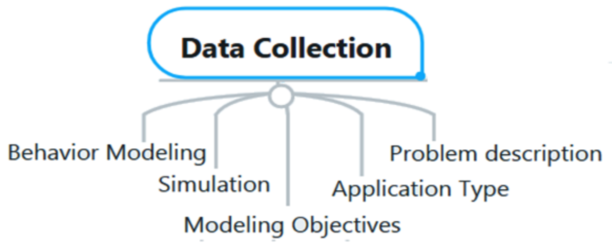


Figure 2: Data collection in enhanced approach

Data collection modeling objectives are as below.

- To build the performance simulation model of GPU architecture
- Modeling Reconstruction algorithm and mapping
- Plotting performance matrices that are latency and utilization with respect to attributes such as resource speed, traffic rate and simulation time

Input workload type described as user defined with image frame traffic and defined traffic rate. System Behavior is described by task flow diagram as shown in Figure 3.

Captured image is in the form of raw data is fragmented and processed to get discrete signal stored in CPU memory is known as K-Space data. Acquired K-space data is transferred from CPU to GPU to perform the computation of inner loop and outer loop iteratively based on the image size [12]. Each task having fixed priority defined task time and allocates tasks to resources based on their availability.

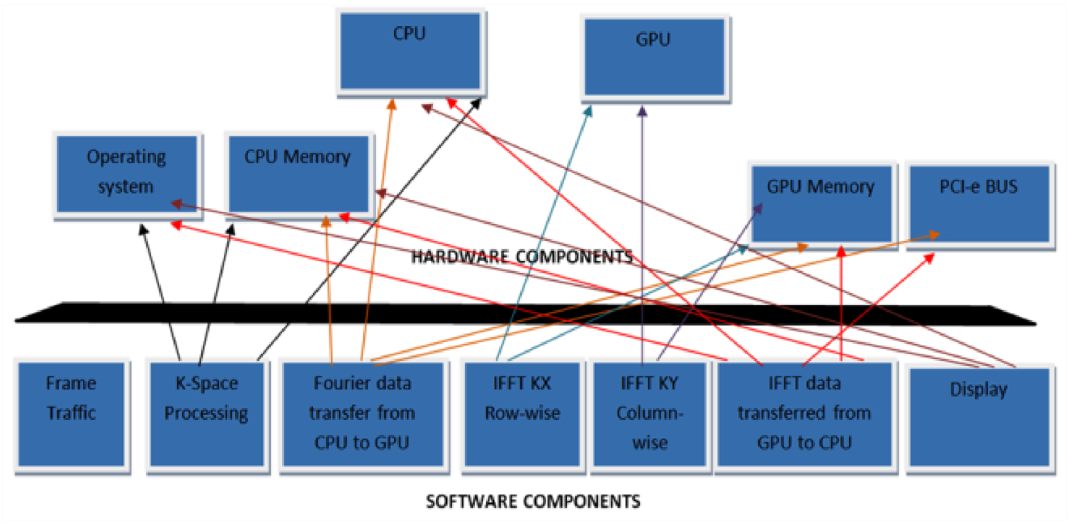


Figure 3: Task flow of MRI Reconstruction

Particularly K-space, transferring data from CPU to GPU memory and from GPU to CPU memory and display of reconstructed tasks are mapped to CPU. Process of IFFT for column and row is allocated for GPU. 1-D inverse Fourier transform is done in KY direction [8]. Split Bergman reconstruction algorithm considered requires the parallel computations for IFFT; hence GPU can support huge computation with parallel processing [6][7]. This processed data stored in GPU memory is transferred to CPU memory. Computed information is transferred to the CPU to display the computed image.

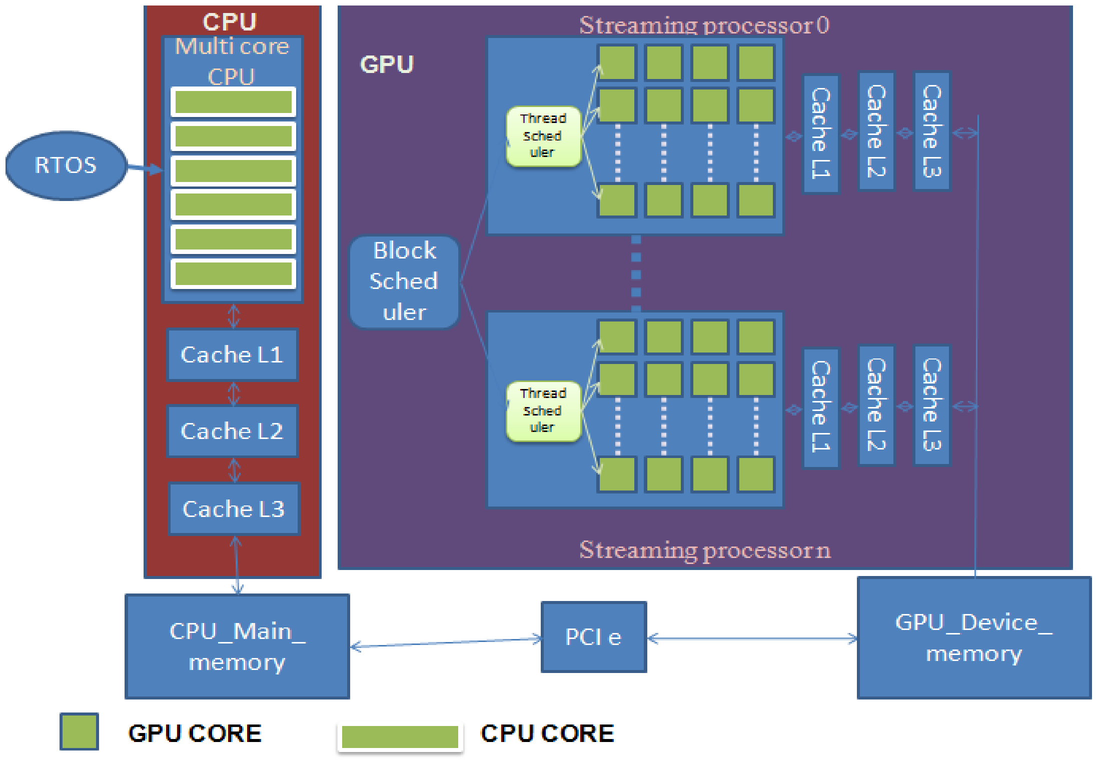
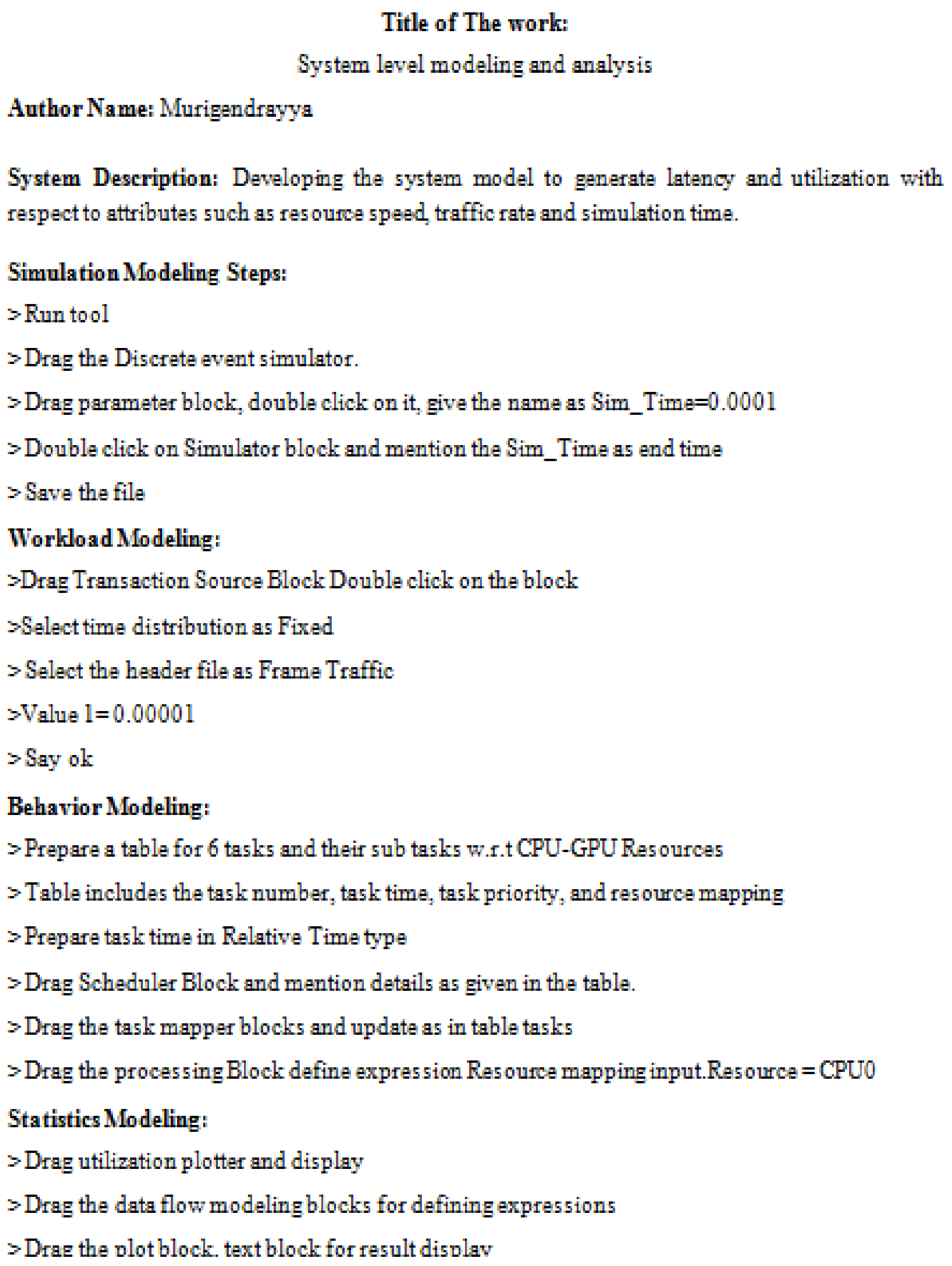


Figure 4: CPU-GPU architecture

CPU-GPU architecture is used for resources modeling as shown in the Figure 4. CPU specification (Xeon X5650) includes 6 cores of 12 threads with speed 2.66 GHz, 3200 MHz Bus speed and 3 levels of memory cache. CPU controls the data transfer between CPU memory and GPU memory, computing physical address and performs the Fourier transform. GPU specification (Tesla C2050) includes 488 CUDA cores, 16 streaming processors with 32 cores and 578 MHz core speed. GPU performs concurrent execution of IFFT [9].

4. RESULTS AND DISCUSSION

Algorithm used in the enhanced approach, collect the mentioned modeling information in the questioner form and generated Modeling guidelines report as shown in figure 5.



Title of The work:

System level modeling and analysis

Author Name: Murigendrayya

System Description: Developing the system model to generate latency and utilization with respect to attributes such as resource speed, traffic rate and simulation time.

Simulation Modeling Steps:

> Run tool

> Drag the Discrete event simulator.

> Drag parameter block, double click on it, give the name as Sim_Time=0.0001

> Double click on Simulator block and mention the Sim_Time as end time

> Save the file

Workload Modeling:

>Drag Transaction Source Block Double click on the block

>Select time distribution as Fixed

> Select the header file as Frame Traffic

>Value 1= 0.00001

> Say ok

Behavior Modeling:

> Prepare a table for 6 tasks and their sub tasks w.r.t CPU-GPU Resources

> Table includes the task number, task time, task priority, and resource mapping

> Prepare task time in Relative Time type

> Drag Scheduler Block and mention details as given in the table.

> Drag the task mapper blocks and update as in table tasks

> Drag the processing Block define expression Resource mapping input.Resource = CPU0

Statistics Modeling:

> Drag utilization plotter and display

> Drag the data flow modeling blocks for defining expressions

> Drag the plot block, text block for result display

Figure 5: Modeling Guidelines

Figure 6 shows System model built by referring modeling guidelines. Modeling guidelines gives only basic information to build the system. Complete executable model built using VisualSim modeling library and Ptolemy simulator as shown in Figure 6.

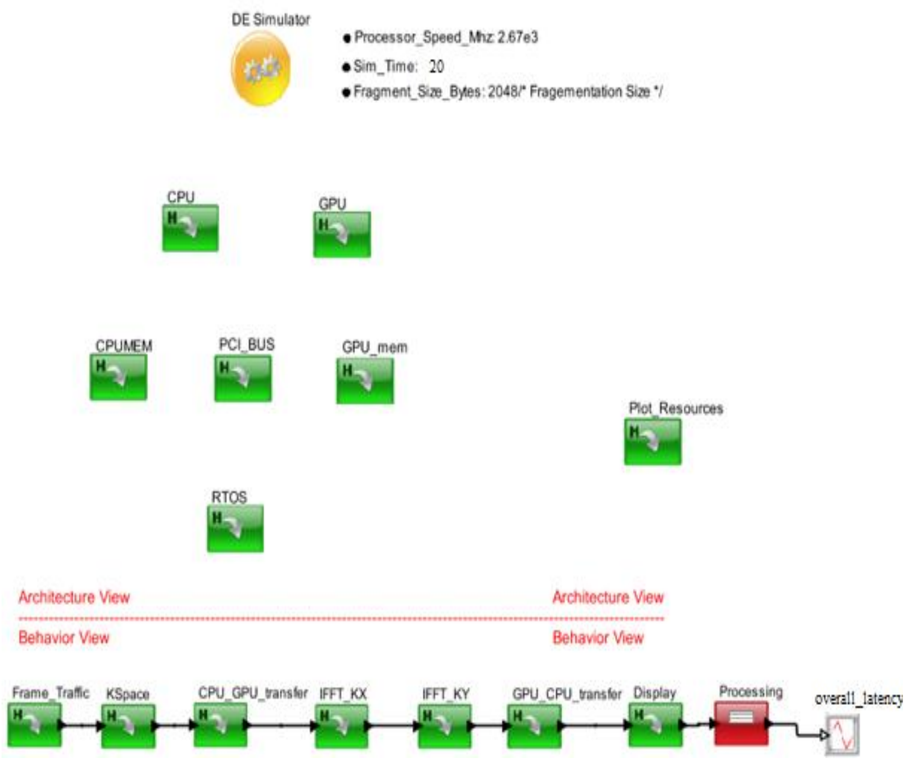
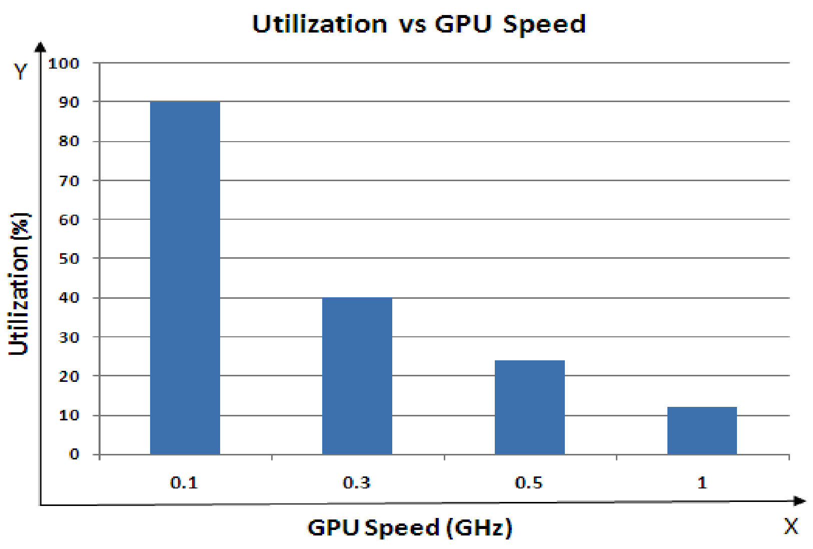


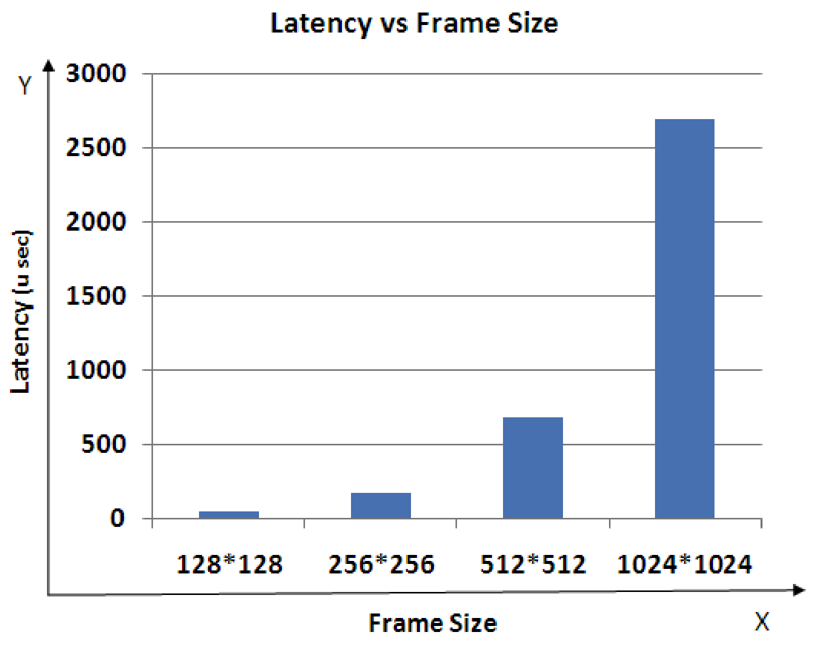
Figure 6: System Model View



X-axis: GPU Speed (GHz), Y-axis Utilization (%)

Figure 7: Utilization versus GPU Speed

Figure 7 shows that the Utilization of the resource decreases as increase in resource speed. Number of streaming processors considered here is 8. Utilization of eight streaming processors is 90%, 40%, 24%, 12% at 0.1GHz, 0.3GHz, 0.5GHz, and 1GHz respectively. Observation made here is that the utilization of streaming processors decreases as increase in streaming processors speed. Figure 9 shows, latency increases as increases in image size. Image sizes we considered here are 128*128, 256*256, 512*512, and 1024*1024. Latency numbers 44us, 169us, 673us and 2688us at frame size 128*128, 256*256, 512*512 and 1024*1024 Maximum latency is at 1024*1024. Here latency increases as increase in frame size.



X-axis frame size, Y- axis latency (u sec)

Figure 8: Latency variations vs. Frame Size

5. CONCLUSIONS

Modeling effort is the challenge of simulation based system level modeling addressed by demonstrating the enhanced system level modeling approach. Using this approach system model can select the right modeling components and modeling template in a short period of time. Hence with the help of this approach system level modeling and analysis can be done with less modeling effort, modeling time and with minimum modeling knowledge. Future scope of this work is to increase the number of modeling templates and integrating them as a part synthesis process from idea to system model. This synthesis automation is required to be accessible online because faster and large up gradation of synthesis frameworks can be possible in huge and fast manner. System engineer can get synthesized analysis specific simulation model online by providing system details and modeling objectives. On the other side, system designers can upload their models in the form of questionnaires.

REFERENCES

[1] F.Vahid and T.Givargis, "Embedded System Design – A Unified Hardware/Software Introduction" , Join Wiley & Sons, New York, Ny, 2002.

[2] Zhenmin Li, Heejong Park, Avinash Malik, Kevin Wang, Zoran Salcic, Boris Kuzmin, Michael Glaß, Jürgen, "Using design space exploration for finding schedules with guaranteed reaction times of synchronous programs on multi-core architecture", Journal of Systems Architecture, vol.74, pp.1-16, 2016.

[3] ACJ Kienhnis, "Design space Exploration of stream based Dataflow Architectures: Methods and Tools", PhD Thesis, Delft University of Technology, 1999.

[4] Nouri A., Bozga M., Bensalem S., "Building Faithful Embedded Systems Models: Challenges and Opportunities", Model-Implementation Fidelity in Cyber Physical System Design. Springer, Cham, 2017.

[5] VisualSim : Mirabilis Design, Inc http://www.mirabilisdesign.com/

[6] David S. Smith, John C. Gore, Thomas E. Yankeelov, and E. BrianWelch,"Real-Time Compressive Sensing MRI Reconstruction Using GPU Computing and Split Bregman Methods" International Journal of Biomedical Imaging, vol. 2012, pp.1-6, 2012.

[7] Górski, Maciej J. Ogorzałek, "Assignment of unexpected tasks in embedded system design process", Microprocessors and Microsystems, vol. 44, pp.17-21, 2016.

[8] Steven Haveman, G. Bonnema, "Communication of Simulation and Modelling Activities in Early Systems Engineering," Procedia Computer Science, vol. 44, pp.305-314, 2015.

[9] PetruEles, "System Design and Methodology/ Embedded Systems Design (Modelling and Design of Embedded Systems)",TDTS30/TDDI08.

[10] Marwedel, Peter, "Specifications and Modeling, Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things", Embedded Systems Design, Ed. 3, Springer, pp. 27–123, 2018.

[11] Stewart Robinson, "A Tutorial on Conceptual Modeling For Simulation," Proceedings of the 2015 Winter Simulation Conference, Huntington Beach, CA, pp. 1820-1834, 2015.

[12] Abu Asaduzzaman, Vidya R. Suryanarayana, Fadi N. Sibai, "On Level-1 Cache Locking for High-Performance Low-Power Real-Time Multicore Systems," Computers & Electrical Engineering, vol. 39, no. 4, pp. 1333–1345, 2013.