On the Impact of Address Space Assignment on Performance in Systems-on-Chip

G. Hazari, M. P. Desai, H. Kasture
Department of Electrical Engineering
Indian Institute of Technology Bombay
{ghaza, madhav, harshad}@ee.iitb.ac.in

Abstract

Today, VLSI systems for computationally demanding applications are being built as Systems-on-Chip (SoCs) with a distributed memory sub-system which is shared by a large number of processing elements. The memory sub-system is a potential performance bottle-neck in the system. In this paper, we consider such a distributed memory sub-system and study the impact of address space distribution on system performance. For a given application on such a system, we introduce the notion of address assignment quality. We show that this assignment quality metric is strongly correlated with memory sub-system throughput over large regions of the design space. We show this using open loop performance modeling of the memory sub-system, and justify this using a queueing and a Markov chain analysis. Further, we develop a detailed memory sub-system model for a multi-processor simulation system built on the Augmint [14] framework. Using two (highly parallel) applications (matrix multiplication and bubble sort) we show that application throughput and assignment quality are strongly correlated over large regions of the design space. We infer that maximization of the assignment quality metric can be a fundamental goal in designing memory sub-systems and in developing applications in such Systems-on-Chip.

1 Introduction

Many VLSI systems for demanding applications of today such as network processing, wireless and storage networks are being built as Systems-on-Chip (SoCs) [1] [2] which integrate a number of different circuit modules (e.g. processors, memories, buses, hardware accelerators) on to a single chip. For high performance, parallelism is incorporated by replicating components on the chip.

We visualize an SoC as consisting of four interacting sub-systems:

- Processing sub-system: responsible for computations and operations on data.
- Memory sub-system: responsible for storage and retrieval of data.
- Interconnect sub-system: responsible for the communication of data between different parts of the system.

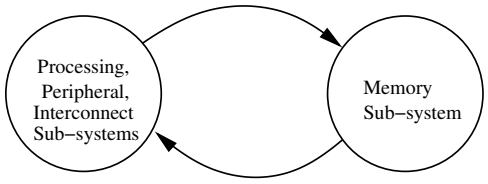

Figure 1. The SoC Memory Sub-system

- Peripheral sub-system: responsible for providing an interface to the external world.

The memory sub-system is organized as a distributed shared memory (DSM) constructed using available memory technology (SRAMs, DRAMs, CAMs). Of the four sub-systems listed above, the memory sub-system is likely to be a performance bottle-neck since memory speeds are not scaling as rapidly as processor speeds [4] [3] and the shared memory property restricts the amount of parallelism that can be incorporated.

Among the reported VLSI system design approaches, there is a trade-off between the use of detailed models [5] [7] [6] (with simulations being used to evaluate designs) and the use of simple models or metrics [8] (where analytical optimization techniques can be applied). It is not feasible to use simulations to explore a large design space and simple models may not accurately predict actual system performance, hence hybrid approaches (where metrics are used to obtain an initial design and simulations used to refine the design) are attractive options [9]. Our research work is directed towards identifying simple models and metrics for the SoC memory sub-system and establishing their validity.

The memory sub-system receives data accesses from the rest of the SoC. Each access is directed towards a logical address which resides in one of the physical memory devices (or modules). The memory sub-system interacts with the rest of the system as shown in Figure 1. From a memory performance perspective, we expect the *address space assignment*, which is the map between logical addresses and physical locations, to play an important role. In this paper we establish this relationship using empirical and analytical arguments by studying a metric for the address space assignment problem: our objective is to maximize memory sub-system throughput, and hence SoC throughput.

We start (in Section 2) by introducing three models for the SoC memory sub-system: an elementary model, an open

loop performance model and a detailed simulation model (which also captures the interaction with the processing sub-system). Using the elementary model, we define (in Section 3) an *assignment quality* metric for any given address space assignment. Using the open loop performance model (Section 4) and the simulation model (Section 7) we study the correlation between assignment quality and throughput. We find a strong correlation over a large region of the design space (but not over the entire design space). The simulation data also establishes the validity of the open-loop performance model for the memory sub-system.

In order to understand the observed correlation, we have used a queueing model (under Poisson approximations) and a discrete parameter Markov Chain analysis (for small sizes) of the memory sub-system (the details can be found in [16]). Using the results of these analysis we show that assignment quality is directly related to memory sub-system throughput (in Section 5 and Section 6 respectively), thus confirming the empirical observations.

2 Memory Sub-system Models

In this section, we introduce a series of three models for the SoC memory sub-system in increasing order of detail.

2.1 The Elementary Model

This model comprises the following simple visualization of the memory sub-system:

- The memory sub-system consists of a set of memory modules $M = \{M_1, M_2, \ldots M_p\}$. Each module ($M_i$) has a storage capacity Ω_i, access rate μ_i and an access latency τ_i. The data size for all accesses is fixed.
- The application views the memory sub-system as a logical address space $A = \{S_1, S_2, \ldots S_k\}$. Each logical address corresponds to the same amount of storage and the relative frequency of accesses to each address is known (let the access frequency to S_j be f_j).
- The logical addresses are assigned to the physical memory modules.

In this model, it is assumed that each module works in parallel on the accesses directed towards it, and the time taken to completely process the trace is determined by the module that takes the longest time to complete its sub-trace. The throughput of the memory sub-system in this model (defined as number of accesses served per second) can then be shown to be $\mu = \min_i \mu_i/F_i$, where F_i is the relative access frequency to module M_i.

2.2 The Open Loop Performance Model

The next model looks at the memory sub-system as an isolated system which responds to a sequence of accesses:

- The physical view is as shown in Figure 2. We assume that the ports, nets and arbiter have a very high bandwidth and very low latency, hence memory module access times and queue sizes are the only performance critical parameters.

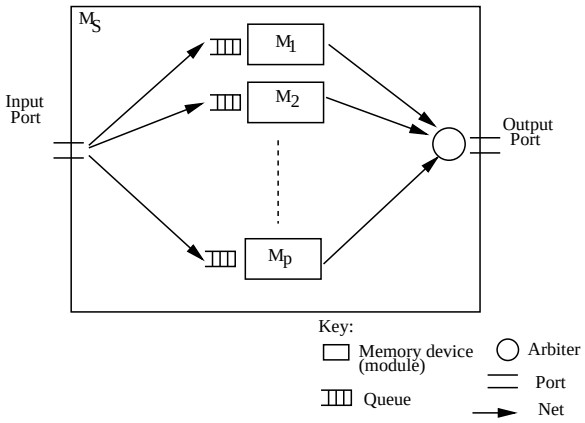

Figure 2. Open Loop Performance Model

- The parameters for a module include those in the elementary model and the size of the queue in front of the module (m_i) in terms of number of accesses.
- The application provides an access trace to the memory sub-system, which may be known or generated by a stochastic process.

In this model we assume that the memory sub-system is under stress: there are always accesses waiting at the input port. The access at the head is placed into the queue of the module it is directed towards as soon as there is space, blocking subsequent accesses in the mean time. Each module serves the access at the head of its queue at intervals of its cycle time (reciprocal of access rate). The head-of-line blocking at the input port causes an under utilization of the modules: some modules may be idle when they have accesses blocked at the input port.

2.3 The Memory Sub-system Simulation Model

This is a detailed model for the SoC memory sub-system, for a simulation system that is also intended to accurately capture the interaction with the processing sub-system. We assume that the SoC is synchronous and all measurements of time are in terms of the system clock.

The physical view is in the form of a template and shown in Figure 3. This template is general enough to model a wide range of distributed memory sub-systems and contains the following components:

- Memory modules: These are the main data storage components. Each module is a simple random access device and is characterized by an access cycle time.
- Ports: These are points of entry or exit for the memory sub-system, and provide connections to the processors.
- Queues: These serve as a first-in-first-out (FIFO) temporary storage space for accesses and are characterized by their size in terms of number of accesses.
- Arbiters: These devices arbitrate at points where multiple paths converge. The default policy is first-come-first-served (FCFS) and they are characterized by a cycle time (time required to make one decision).

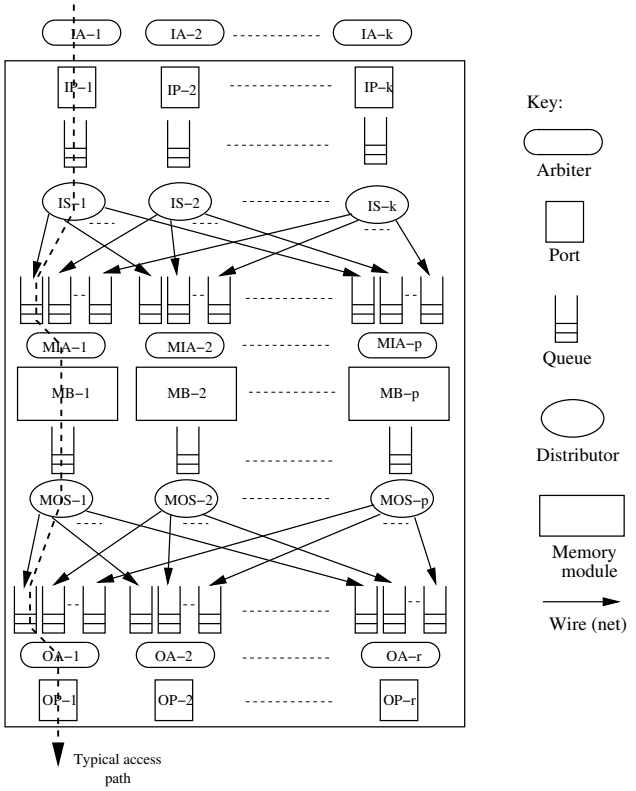

Figure 3. Simulation Model

- Distributors: These devices direct the accesses at points where multiple paths diverge and are characterized by a cycle time.
- Wires: These are long distance interconnects, they may be pipelined and are characterized by number of stages.

The path taken by a typical access is also indicated in Figure 3. The write accesses are presently assumed to return an acknowledge before the processor thread can continue and also follow the same path as the read accesses.

The application programs are written as parallel algorithms in a high level language. The simulator is intended to simulate the application running in parallel on multiple processors in tandem with the memory sub-system model.

Our present models treat all accesses as having equal data size, however it is simple to extend these models to account for different access data sizes. Our present models also neglect techniques such as caching and row buffering (these techniques can be applied independently of the address space assignment). It is easy to incorporate such techniques into advanced versions of each of the models.

3 Address Space Assignment Quality

Recall the elementary model for the memory sub-system and note that $\mu_0 = \sum_{j=1}^{p} \mu_j$ is the maximum access rate (throughput) the memory sub-system can support.

An *Address Space Assignment* (Θ) allocates each address to a module subject to the storage capacity constraints:

$$\Theta : AS \rightarrow M \qquad such\ \ that$$
$$\sum_{i:\Theta(S_i)=M_j} 1 \quad \leq \Omega_j \qquad j = 1, 2, \ldots p$$

Given an assignment Θ, the relative access frequency to module M_j is $F_j = \sum_{i:\Theta(S_i)=M_j} f_i$. We define the effective load on the module as $l_j = F_j/\mu_j$ and the total load on the memory system as $L(\Theta) = \max_j l_j$.

Expected access rate is given by: $R(\Theta) = 1/L(\Theta)$. The assignment that gives maximum expected access rate (let $R_b = \mu_0$) will balance the effective load across all modules:

$$l_j = \frac{F_j}{\mu_j} = c \ \ \forall j \quad \Rightarrow c = \frac{1}{\sum_{j=1}^{p} \mu_j}$$

The assignment that gives minimum expected access rate assigns the entire load to the slowest module: $R_w = \min_j \mu_j$. We define *assignment quality* '$q(\Theta)$' as:

$$q(\Theta) = \frac{R(\Theta)}{R_b - R_w} - \frac{R_w}{R_b - R_w}$$

The assignment quality metric ranges from 0 to 1, and is linearly related to expected access rate in this model.

In the subsequent sections we will establish the relation between assignment quality and performance on the more realistic open loop performance model and also on the detailed simulation model. We expect assignment quality to give a reasonably good estimate of system performance when the access stream is i.i.d. across the address space. Formulating appropriate metrics for an access stream with correlations and time-varying distributions form separate directions of research. Certain aspects of time-varying access patterns have been addressed in [13].

4 An Empirical Study on the Open Loop Performance Model

In this section we take up the open loop performance model and generate i.i.d. access traces. We have used *PARTSim*, an SoC performance modeling and simulation tool suite for trace generation and simulation.

We study memory sub-systems built up of 2, 4, 8, 16 and 32 (non-pipelined) modules. In this paper we present data for the 16 module case. We have observed identical trends over the entire range from 2 to 32 modules [17]. For each system size we consider various sets of module access rates (labeled *ds 1 - 5*) ranging from a system with identical modules to one where the maximum imbalance in module access rates is 1:20. We consider all modules to have identical queue size m, which we vary between 1 and 50. We vary assignment quality from 0.0 to 1.0 in steps of 0.05 and corresponding to each assignment quality we generate a random set of module access frequencies.

In Figure 4 we present the observed dependence of memory sub-system throughput ($T(\Theta)$) on assignment quality ($q(\Theta)$) for select queue sizes (for *ds 1*). In Figure 5 we present the observed correlation between throughput and assignment quality as the queue size is varied.

We observe a strong correlation in all cases and this correlation improves as queue sizes are increased. The minimum correlation observed for the system with 2 modules is

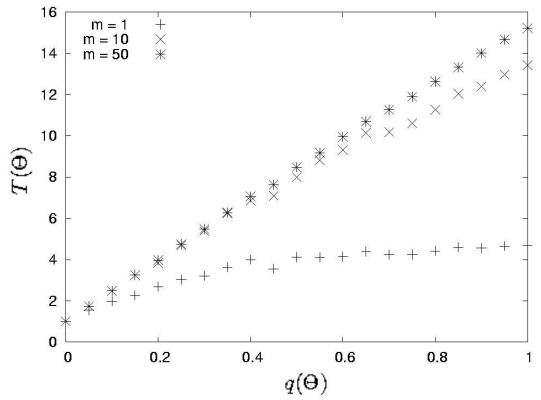

Figure 4. Throughput (16 Modules, ds 1)

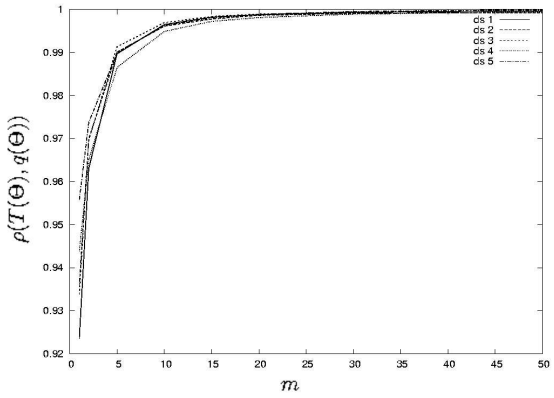

Figure 5. Throughput Corr. (16 Modules)

0.95 and reduces to 0.92 as the number of modules is increased to 32 [16].

We next take a closer look within the following ranges of assignment quality: $0.9-1.0, 0.8-0.9, 0.4-0.5$ and $0.0-0.1$. We vary assignment quality in steps of 0.02 within each range. In Figure 6 we show the mean correlation (across *ds 1 - 5*) of memory throughput with assignment quality.

We observe that with small queue sizes the correlation is weak in the higher ranges of assignment quality. The extremely low correlation leads to the conjecture that the assignment with maximum quality may not always be the assignment that gives maximum throughput. This conjecture is validated in the Markov Chain analysis (Section 6).

Based on our observations we make the following conclusions: assignment quality gives a good prediction of memory sub-system throughput at a coarse level, however assignment quality is not a good prediction of memory sub-system throughput at a fine level within higher ranges of assignment quality when queue sizes are small.

We define the region where memory sub-system throughput is strongly correlated with assignment quality as the *linear region*. Within the linear region, throughput can be approximated as a linear function of assignment quality where the coefficients of the approximating polynomial depend on the queue sizes [16].

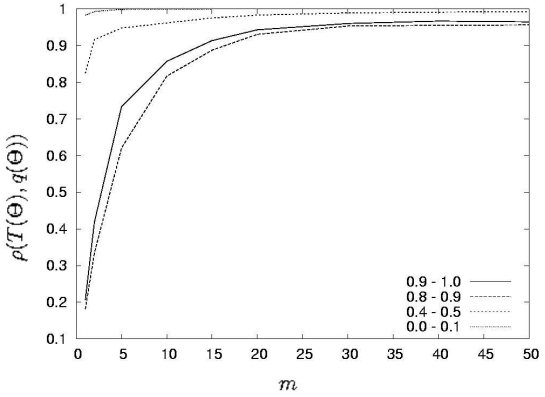

Figure 6. Throughput Correlation in Specific Ranges of Assignment Quality (16 Modules)

This study establishes the address space assignment problem as a fundamental design problem towards enhancing memory sub-system throughput and that assignment quality is a strong candidate as a performance metric when queue sizes are not small. We will justify this conclusion using analytical techniques in the next two sections.

5 A Queueing Network Analysis of the Open Loop Model

To understand the observations in the previous section we model the access stream as a Poisson process and the module service times as exponentially distributed [10].

Let the access arrival process be Poisson(λ_0) and the service time distribution for module M_i be Exponential(μ_i). Let p_i be the probability that an access is directed towards module M_i. Recall m_i is the queue size for M_i and $\mu_0 = \sum_{j=1}^{p} \mu_j$ is the maximum access rate supported.

We analyze the system as an open queueing network using the approximations in [11]. (The complete details of our analysis are in [16].) The module queues are not blocked by any downstream nodes. We approximate the probability with which they will be full by the probability an M/M/1 queue has length greater than $m_i + 1$: $(\lambda_0 p_i/\mu_i)^{m_i+2}$ [11] (we add '1' to account for the access being served).

Next we model the input port queue as a Hyperexponential distribution [11], and the expected service time is given by: $\tau = \sum_{i=1}^{p} p_i(\lambda_0 p_i/\mu_i)^{m_i+2}/\mu_i$. For the maximum arrival rate supported, $\tau = 1/\lambda_0$ and λ_0 must be maximized.

We are able to solve this problem using Lagrange multipliers when all queue sizes are equal. The solution is: $p_i = \mu_i^{1+1/(m+2)} / \sum_{i=1}^{p} \mu_i^{1+1/(m+2)}$. As m increases, $p_i \rightarrow \mu_i/\mu_0$ (the assignment with maximum quality).

This confirms that at small queue sizes the assignment that gives maximum expected throughput differs from the assignment with maximum quality, however the respective assignments converge as queue sizes are increased.

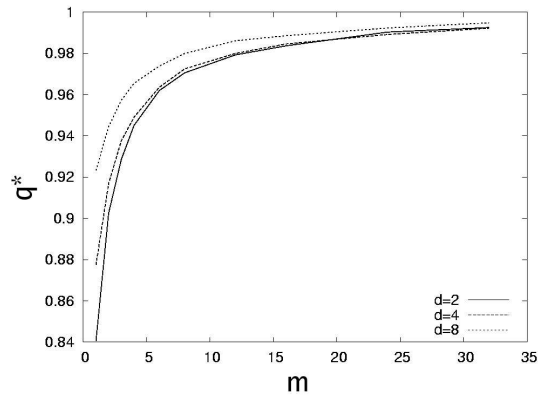

Figure 7. Assignment Quality Giving Maximum Expected Throughput

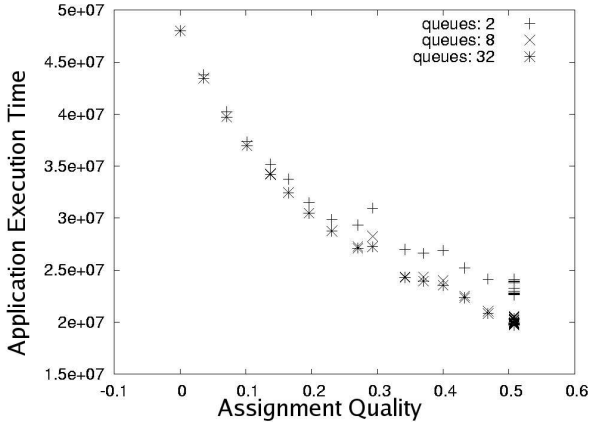

Figure 8. Execution Time Observations

6 A Markov Chain Analysis of the Open Loop Model

In this analysis we build and analyze a discrete parameter Markov Chain for the open loop memory sub-system model and an i.i.d. access trace. We define the state of the system as the length of the queue and number of pending clocks for the access being served on each module, along with the destination of the access at the head of the input port queue. We can hence list the state space. From the module access frequencies we get the transition probability matrix and then the stationary probability vector and expected throughput. A demonstration of this procedure on some simple cases is available in [16], here we shall only present the results.

First, consider a memory sub-system having two identical modules (M_1 and M_2) with queues of size m_1 and m_2 respectively. Let each access be directed towards M_1 with probability x, then the expected throughput is given by:

$$R = 2 - \frac{1 + \gamma^{m_1+m_2+1}}{1 + \gamma + \gamma^2 + \ldots + \gamma^{m_1+m_2+1}}$$

where $\gamma = \frac{(1-x)}{x}$. The maximum of this function is attained at assignment quality $q = 1.0$ for all queue sizes.

Next, consider a system with two modules with service times 1 and d and having equal queue size m. For $d = 2$ and $m = 1$, we have observed that the throughput attains its maximum at $x = 0.746$ (accuracy up to three places of decimal), whereas the assignment with maximum quality is at $x = 0.667$ [16]. In Figure 7 we present the assignment quality giving maximum expected throughput (q^*) for $d = 2, 4, 8$ and m varying from 1 to 32.

We again find that at small queue sizes the assignment that gives maximum throughput differs from the assignment with maximum quality, and there is a convergence as queue sizes are increased. This explains the observed empirical results and confirms the important role that assignment quality plays in capturing memory sub-system throughput.

7 A Memory Sub-system Simulator for Multi-processor Systems: A Case Study

We have developed a memory sub-system simulator [15] which incorporates the simulation model introduced in Section 2 and works on the Augmint platform [14] (which is a simulation framework for multi-processor systems using the Intel x86 processor). The system clock is assumed to be the processor clock. The user can write application programs in C using a multi-threaded format.

We use the simulator to understand the dependence of application performance on assignment quality as well as judge the validity of the open loop memory sub-system model as a means of predicting SoC performance. We have selected two applications for this case study: matrix multiplication and bubble sort (the details are in [16]). In our experiments, we have split the address space into 32 distinct memory spaces using the 3rd to 7th LSBs of the virtual address (since the last two bits are always 00).

In our experiments we consider memory sub-systems of size 2, 4 and 8 modules. The memory sub-system has a single input port and a single output port, all arbiter and distributor cycle times are two clocks, and all wires have a single stage. We vary module latencies as follows: half of the modules have latency d_0 clocks and the remaining have latency $r \times d_0$ clocks. We consider $d_0 = 2, 5, 10, 20$ and $r = 1, 2, 10$. We keep all queue sizes equal for an experiment and vary the queue size as $2, 8, 32$. For each memory sub-system configuration we generate approximately 30 different address assignments, such that they span the assignment quality axis well.

In Figure 8, we show as a representative plot, application execution time versus assignment quality for the bubble sort application on the memory system with 4 modules for $d_0 = 10$, $r = 1$. In Table 1 we tabulate the correlation of application throughput (reciprocal of execution time) with assignment quality for the bubble sort application on the memory sub-system with 4 modules, for each of the memory configurations. The rest of the correlation tables are available in [17] and identical trends are seen.

We find a strong correlation when memory latencies are

d_0, r	queues: 2	queues: 8	queues: 32
2,1	0.96	0.97	0.96
2,2	0.93	0.92	0.92
2,10	0.93	0.94	0.93
5,1	0.98	0.99	0.99
5,2	0.96	0.98	0.98
5,10	0.95	0.98	0.98
10,1	0.98	0.99	0.99
10,2	0.97	0.99	0.99
10,10	0.94	0.98	0.99
20,1	0.99	0.99	0.99
20,2	0.97	0.99	0.99
20,10	0.93	0.98	0.99

Table 1. Appl. Throughput v/s Ass. Quality: Correlation

greater than five clocks, and when queue-sizes are equal to or greater than eight. We hence infer that the concept of a *linear region* in the design space is applicable to this model as well. (The coefficients of linear regression are in [16].)

We conclude this case by reiterating that application throughput is strongly correlated with assignment quality in large regions of the design space. We have also observed a strong correlation with throughput predicted on the open loop model in the same regions of the design space [16].

In this case study, we have explored only memory module latencies and queue sizes as co-ordinates of the SoC design space. Using our simulator it is possible to explore the following co-ordinates as well: number of input and output ports and their connectivity, arbiter and distributor cycle times, wire stages, the selection of address bits to define the memory spaces, number of processing threads available to the application programs and different data structures or algorithms for the same application.

8 Conclusion

In an SoC, we expect that the memory sub-system is likely to be a performance bottle-neck. In this paper we have identified the address space assignment as having a fundamental impact on memory sub-system performance and SoC performance, and established the relationship. We have studied an assignment quality metric and an open loop performance model for the memory sub-system, and found them to give a good prediction of SoC throughput (on the cases studied) within the linear region of the design space (namely when memory latencies are not small and either queue sizes are not small or the quality of assignments under consideration is not very high). This metric and performance model can be used in SoC design procedures to set up analytical optimization techniques and fast simulation methods respectively.

In the process, we have developed a memory sub-system simulator which can be used to explore a large number of co-ordinates of the SoC design space and to conduct further studies on the relationship between the assignment quality metric and the open loop performance model with SoC performance. Further, our Markov Chain analysis accurately predicts the behaviour of the open loop model when queue sizes are small, a deeper understanding of the observations can extend the results to systems of larger size.

References

[1] Panos C. Lekkas, *Network Processors: Architectures, Protocols, and Platforms,* McGraw Hill 2004.

[2] Lo Jien-Chung, C. Metra, F. Lombardi, "Special Section on Design and Test of Systems-on-Chip (SoC)" *IEEE Trans. on Computers,* vol. 55, no. 2, Feb. 2006, pp. 97–98.

[3] S. McKee, "Reflections on the Memory Wall", *Proc. First ACM Conf. on Computing Frontiers,* Apr. 2004, pp. 162.

[4] Harvey G. Cragon, *Memory Systems and Pipelined Processors,* Jones and Bartlett Publishers Inc., 1996.

[5] R. Domer, A. Gerstlauer, D. Gajski, "SpecC Language Reference Manual", *url=www.specc.org.*

[6] V. Kathail, et. al., "PICO: Automatically Designing Custom Computers", *IEEE Computer,* vol. 35, issue 9, Sept. 2002, pp. 39–47.

[7] A. Mihal, et. al., "Developing Architectural Platforms: A Disciplined Approach", *IEEE Design and Test of Computers,* vol. 19, issue 6, Nov.-Dec. 2002, pp. 6–16.

[8] "Data Transfer and Storage Exploration (DTSE)", *url = http://www.imec.be/design/dtse/.*

[9] S. Kim, C. Im, S. Ha, "Efficient Exploration of On-Chip Bus Architectures and Memory Allocation", *IEEE Int. Conf. on Hardware/Software Codesign and System Synthesis,* Sept. 2004, pp. 248–253.

[10] L. Kleinrock, *Queuing Systems,* John Wiley, New York, 1975.

[11] H. G. Perros, *Queuing Networks with Blocking,* Oxford University Press, New York, 1994.

[12] A. Mittal, M. P. Desai, P. Pandekar, G. Hazari, "PARTSim: User's Manual", *Technical Report,* Processors Algorithms Research & Technology, Pune, May 2004.

[13] E. De Greef, F. Catthoor, H. De Man, "Memory Size Reduction through Storage Order Optimization for Embedded Parallel Multimedia Applications", *Proc. Parallel Processing and Multimedia Workshop,* Geneva, Apr. 1997, pp. 84-98.

[14] A. Nguyen, M. Michael, A. Sharma, J. Torrellas, "The Augmint Multiprocessor Simulation Toolkit for Intel x86 Architectures", *Proc. IEEE Int. Conf. on Computer Design (ICCD),* Oct. 1996, pp. 486–490.

[15] H. Kasture, "A Memory Subsystem Simulator for SoC Applications", *Dual Degree Project Report, Dept. of Elect. Engg., I.I.T. Bombay,* July 2006.

[16] G. Hazari, M. P. Desai, "Towards a Memory Sub-system Design Procedure for Enhanced Performance in SoC Architectures", *PhD. Progress Report, Dept. of Elect. Engg., I.I.T. Bombay,* Aug. 2006.

[17] G. Hazari, H. Kasture, M. P. Desai, "On the Impact of Address Space Assignment on Performance in Systems-on-Chip", *Technical Report, Dept. of Elect. Engg., I.I.T. Bombay,* July 2006.