Journal of Theoretical and Applied Information Technology
10th June 2014. Vol. 64 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A SURVEY OF RESEARCH AND PRACTICES IN MULTIPROCESSOR SYSTEM ON CHIP DESIGN SPACE EXPLORATION

[1]V.PRASANNA SRINIVASAN, [2]A.P.SHANTHI

[1]Asstt Prof., Department of Information Technology, R.M.D Engineering College, India, Chennai

[2]Assoc. Prof., Department of Computer Science & Engineering, CEG, Anna University, Chennai

E-mail: [1]vas.sri81@gmail.com, [2]apshanthi@annauniv.edu

ABSTRACT

This survey presents a perspective on the existing research and practices initiated for the Design Space Exploration (DSE) in Multiprocessor System on Chip (MPSoC) technology. Reduction in size as well as adding more functionality within a single chip by incorporating multiple processors remains the key in the development of the modern MPSoC. This rapid development has been made possible because of the techniques used for scaling down the size of the chip in the field of integrated circuits. MPSoC has been considered as the best candidate for applications such as networking, telecommunication, multimedia, etc. which require high computational demand, high performance, flexibility, high energy efficiency, and low cost design. The designers have the onerous task of building MPSoCs for such applications because they have huge design options in terms of Processing Elements (PEs), micro-architectural features, interconnects, etc. to be considered with specific constraints. Exhaustive search is prohibitive because of the sheer design space size as well as the time to market considerations. Coverage of design space during the exploration process, and evaluating a single design point for finding the optimal design are the two issues that should be considered in any DSE process. This paper provides a comprehensive survey on the existing DSE techniques at the system and micro-architectural levels, the evaluation methodologies and the tools/frameworks with their comparison with respect to the design parameters considered.

Keywords: *Multiprocessor Systems On Chip (Mpsoc), Design Space Exploration (DSE), Embedded System Design, Design Space Pruning, Multi-Objective Optimization*

1. INTRODUCTION

Advances in semi-conductor technology have led to further and further scaling down of integrated circuit design. Starting from the single chip microprocessors, the technology has grown rapidly to incorporate a complete functional system into a single chip referred as System on Chip (SoC). The technology revolution and the different demands required by the applications under various domains have led to the emergence of MPSoCs as an important class of Very Large Scale Integration (VLSI) systems. An MPSoC is a VLSI system that incorporates most or all the components necessary for an application that uses multiple programmable processors as system components [1]. There has been an unprecedented increase in the use of MPSoC architectures in the design of embedded systems. This can be easily envisioned with the development of modern sophisticated communication devices that are available for the end user. For example, Apple's 4S iphone [2] includes dual core processors in its sub systems for supporting various communication standards, video/audio formats, etc.

MPSoCs are custom designed to meet a particular application's requirements. The applications can be categorized into several domains such as, Networking, Automotive, Telecommunication, Multimedia, etc. Furthermore, the requirements and constraints for designing an embedded system vary across the application domain. For instance, multimedia and wireless communication applications require high computational demand and high energy efficiency. Research direction in the field of parallel processing has given a hope to the designers to apply parallelism, in terms of processing the application on multiple processing elements in parallel. The development of parallel programming models for MPSoC domain is still at its infancy, because the underlying architecture modelled is purely application dependent. Specialized hardware architectures are essential to fulfil these stringent

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and contradictory requirements of high computational power and energy efficiency imposed by such applications. Hence, a heterogeneous MPSoC platform is considered a suitable candidate for such applications by the embedded system designers.

The common issues considered for heterogeneous MPSoC design are low power consumption, memory bandwidth and latency, time to market, less weight, etc. Apart from these basic design issues, there are many emerging steering factors behind the modern design trend such as, scalable and reusable architectures, embedded memory, and IP integration, Network on Chip (NoC), reliability, and support for reconfigurable logic [3]. To incorporate these design issues and factors, the following requirements are highly essential for MPSoC design solutions [4].

a) Integrated Development Environment (IDE) – necessary for establishing user interface, configuring the system components, system structural editing, simulation control, and debugging capabilities,
b) System Structure and Model generation – for specifying configuration parameters and generation of simulator models, and
c) Multiprocessor Programming Model – the most important challenge in developing multiprocessor programming model is the partitioning of a single application into multiple communicating tasks, enabling execution in parallel among different processing elements within the SoC, and providing API libraries for parallel execution of applications and Inter Process Communication (IPC).

Further, it has to be noted that evaluating each component of an MPSoC system separately, for measuring the performance of the complete system will not be appropriate because the interacting behaviour of the components is not taken into consideration during the evaluation process. Thus, the interacting behaviour of all system components requires a system-wide performance evaluation, which imposes additional challenges for the designer.

The system designer has the difficult task of selecting the appropriate components for building an MPSoC for a given application, along with certain constraints that need to be satisfied. There exist a lot of possible design options, from which the major objective for system architects is to identify an optimal design by considering the main objectives such as efficiency, performance, flexibility, energy, power and cost. This identification process requires a well planned and comprehensive design methodology defining the fundamentals of Design Space Exploration (DSE) [5]. The two most important phases in the DSE process are,

1. Exploration: It is the process of searching within the huge design space to find the optimal or near optimal design solution. Design space is composed of various combinations of software and hardware design options. A complete search with the various design options is prohibitive because of the sheer size of the design space. Hence, a knowledge-guided search strategy is highly essentially for reducing the exploration time and to meet time-to-market considerations.
2. Evaluation of single design point: It is necessary to determine its quality with respect to certain objectives and constraints. The time required for the individual design point evaluation is related to the number of investigated design points. This process constitutes the time spent in developing and describing the intended design point, as well as the time for evaluating the anticipated design finally.

Increasing design complexity has significantly influenced the embedded system design process to support early design decisions, and exploration process is mostly performed at the system level [6]. The most commonly used approach for systematic exploration process is the Y – chart approach [7] in which application descriptions and architecture specifications are specified separately. The exploration process usually has certain objectives to be met for either maximization or minimization of the corresponding metric taken for consideration. The mapping step binds application tasks to architectural building blocks. The evaluation step validates the mapping process based on the constraints and objectives that are to be met by performing test runs. The search process becomes even more cumbersome if multiple objectives are considered for the design.

Given a specification of the application and system requirements, the designer needs to shrink the range of feasible designs to a smaller number of possible designs that are best suited for the given application. The design space can increase exponentially with respect to the number of processing elements and other micro-architectural parameters considered, and thus

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

increase the complexity of the search process. Consequently, a disciplined approach to the DSE is needed in order to evaluate large design spaces with high number of potential designs. This includes algorithms to prune and cover the design space in a systematic way at different levels of abstraction and refinement. The goal of this paper is to provide a comprehensive survey of recent work in the area of MPSoC DSE.

This paper is organised as follows; the next section defines the task of design space exploration as an optimization problem, describes strategies for optimization and metrics considered for optimization during the exploration process. Section 3 provides the classification of strategies for covering the design space. Section 4 elaborates on the different techniques used to evaluate a single design point. In section 5, a brief description and comparison of the available DSE tools/frameworks is presented. Section 6 presents open research issues in MPSoC design space exploration. Finally, section 7 summarises and concludes the paper.

2. DSE AS AN OPTIMIZATION PROBLEM

The problem of design space exploration can be described as the process of searching and evaluating an optimal design option among several design parameters available within the target architecture, with certain objectives required to be satisfied for the given application. The design space becomes enormous when there are several design options available. The exponential growth of the design space has been best explained by Kempf et.al [5], by assuming an application consisting of T tasks which can execute on n_{PE} processing elements, the authors state that there are $(n_{PE})^T$ possible design points.

Several researchers have looked at an exhaustive search of the design space. Blythe and Walker [8] have performed high-level synthesis to find all optimal Pareto points in order to completely characterise the entire design space for the module selection, clock length determination, and scheduling problems. A theoretical framework that can capture the general performance properties for a class of multi core processors of interest over a large design space and workload space has been proposed by Jung et al. [9]. The methodology incorporated models the multi core processors at the thread level using queuing network model and an iterative procedure has been employed to cover the large design space.

A novel methodology has been employed to describe the inefficiency of general purpose processors for a specific application by Hameed et al. [10]. The technique explores the customisation that is possible at the functional unit level to improve the performance of the general purpose Chip Multiprocessor (CMP) with that of an Application Specific Integrated Circuit (ASIC) like solution. The work proposed by Shee and Sri Parameswaran [11] performs an exhaustive search for exploring the design space for ASIP's. Multiple cores for a single application have been considered instead of multiple applications executing in parallel. A single application has been parallelized using two different methods, a master-slave model and a sequential pipeline model. The models have been implemented using Tensilica's Xtensa customisable processor core [12] and manual DSE has been performed with different configurations of ASIP to find an optimal configuration for the given application.

Most of the exhaustive design space search approaches do not consider the option of executing a single application by partitioning it into multiple parallel units, and assume fixed number of processing elements for exploring the design space. The design point can be even more enormous if a single application can be partitioned for its execution with multiple processing elements and if the number of processing elements is not fixed within the target architecture. Thus, exhaustive search is not the best way of exploring the large design options available for the embedded system design. It is very important to have a guidance scheme and fix some metrics as objectives for the search problem, so as to evaluate each design option against the metrics and guiding parameters that are considered. The guiding scheme will aid the search process, to walk only through the desired feasible regions of the design space based on the metrics considered, for evaluation to attain specified objectives, avoiding in-feasible design points. Thus, the DSE problem can be stated as a single/multi objective optimization problem. The following sub sections defines the Pareto optimality criterion, briefly describe the different strategies for optimization and objectives/metrics considered during the exploration process.

2.1 Definition Of Pareto Optimality Criterion

In order to classify the methods adopted for optimization, the term Pareto Optimality [13] needs to be defined. This criterion is applicable when a multi-objective search of the design space is performed. The objectives considered in the domain of micro – architectural design could be the minimization of cost, power consumption, or the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

maximization of speed. Mostly these objectives are inter-related with each other. Thus, optimizing toward a single objective may highly influence other objectives as well.

Definition 1 Pareto criterion for Dominance:

Given k objectives to be minimised without loss of generality and two solutions (designs) A and B with values (a_0, a_1, …, a_{k-1}) and (b_0, b_1, …, b_{k-1}) for all objectives, respectively, solution A dominates solution B if and only if,

$$\forall_{0 \le i < k}\, i : a_i \le b_i \; and \; \exists_j \; a_j < b_j$$

This means that a superior solution is at least better in one objective while being at least the same in all other objectives.

Definition 2 Pareto – Optimal solution:

A solution is called Pareto - optimal if it is not dominated by any other solution.

2.2 Strategies For Optimization

Optimization techniques can be generally classified according to the following three criteria [6].

- Decision making before search

Different objectives have been combined or aggregated into a single objective by the designer using some optimization methods such as considering the weighted sum of the objectives or converting certain objectives to constraints so that the number of objectives is reduced before the actual search process begins.

- Search before decision making

The optimal solutions have been identified by considering each objective separately during the search process. The result is the set of Pareto-optimal solutions. After performing the initial search with certain objectives, additional criteria can be imposed to find the optimal solution for a given problem.

- Decision making during search

The decision to include a design point has been made during the search process in which, initial search results may be used iteratively to further constrain the design space and guide the search to feasible regions of the design space.

2.3 Objectives/Metrics

This section describes some of the objectives that have been considered during the design space exploration. Weighted sum, ratio, or products of several objectives have been considered by single objective optimisers in order to compensate for conflicting criteria. To prevent biasing of search result towards a certain region of the design space, multi-objective search algorithms keep the objectives separately during the search process.

The objectives such as cost, power dissipation, speed, and flexibility have been considered as primary objectives because they reflect the overall properties of the design directly and are used as fundamental optimization goals. There are other objectives such as utilization of computation or communication resources, affinity metrics, memory specific metrics, physical size, reliability, etc. that are considered as secondary objectives. Sometimes, combined metrics such as energy-delay, computation-power ratio, speed-cost ratio etc. have also been considered for the design optimization.

Different strategies for covering the design space have been proposed in the literature with specific objectives to be satisfied, considering applications from various domains for the target MPSoC architecture. In the following section, different strategies for covering the design space are discussed in detail.

3. STRATEGIES FOR COVERING THE DESIGN SPACE

Other than the exhaustive search method for performing DSE, the strategies proposed in literature for covering the design space can be classified into three categories as given below [14].

- Techniques that try to reduce the design space size
- Techniques that provide exploration heuristics
- Techniques based on statistical analysis aimed at guiding the exploration to specific regions of the design space

3.1 Techniques For Reducing The Design Space Size

The techniques employed for reducing the design space size aim to limit the exponential increase of design space size by eliminating those configurations that are certainly non optimal. Generally, this methodology is known as design space pruning. There are different pruning methods proposed in the literature such as hierarchical

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

pruning, sub sampling or sub dividing the design space, and performing sensitivity analysis on the parameters considered for optimization during the exploration process.

Mohanty et al. [15] have introduced the hierarchical design space exploration, in which infeasible configurations have been eliminated initially by employing constraint satisfaction analysis, and then the remaining space has been evaluated through system – wide performance estimation models to identify the non – optimal configurations, thus avoiding their simulations. Affinity – Driven DSE has been proposed by Brandolese et al. [16] which utilize cost, communication and load as affinity metrics for initial clustering of architectural elements for reducing the design space and then use standard genetic algorithm for the exploration process towards allocation and optimization. Design of Experiments (DoE) has been used by Palermo et al. [17] for creating a coarse view of the target design space by generating an initial set of experiments. Different types of Response Surface Modelling (RSM) techniques have been utilized to refine the exploration, and this process has been iterated to cover the design space.

The idea proposed by Beltrame et al. [14] moves the design complexity from simulation to probabilistic analysis of parameter transformations by employing domain knowledge of the target MPSoC. Exploration has been modelled as a Markov Decision Process (MDP) and the solution to such MDP relates with the sequence of parameter transformations to be applied, in order to maximise or minimise the desired value function. The simulation is performed only in particular cases of uncertainty, thus massively reducing the simulation time needed to perform the exploration of a system, while maintaining near optimality of the results.

The methodology proposed by Yazdanbakhsh et al. [18] describes energy – aware custom instruction identification for critical code segments, which enables to constrain the number of input and output operands for custom instructions to reach acceptable performance, considering the energy dissipation of the register file. The technique proposed by Kokhazadeh and Fatemi [19] uses regular sub – sampling of the parameters considered for design space pruning for covering a wide range of performance metric values. The experimental evaluation reveals that the proposed method can prune the design space exponentially with the number of design parameters, while maintaining a reasonable set of Pareto optimal points of the original design space. An iterative design space pruning methodology based on static throughput analysis of different application mappings has been proposed by Piscitelli and Pimental [20] to reduce the simulation overhead. The methodology interleaves the analytical throughput estimation with simulation, thus forming a hybrid approach which significantly reduces the number of simulations that are needed during the process of DSE.

The overall power and performance of an on-chip multiprocessor system varies due to the manufacturing process variation. Palermo et al. [21] have proposed an improved design space pruning strategy taking into account the possibilities of process variation during manufacturing. At first, variability aware delay and energy models have been derived to exploit the process variation and the impact of application specific constraints and then RSM is utilized to further speed up the DSE process. Hung-Yi Liu et al. [22] have proposed a strategy for performing system level DSE with planning of High Level Synthesis (HLS). The main objective of the proposed strategy is to maximize the reusability of the component library, which has been designed using High Level Language (HLL) and characterized through HLS.

3.2 Techniques Provide Exploration Heuristics

Random sampling is employed usually for exploring the design space. Simulated annealing, Genetic Algorithms and Tabu search are the common techniques that are used in the literature for providing exploration heuristics.

A multi – objective DSE using genetic algorithm has been proposed by Palesi and Givargis [23] for efficiently exploring a parameterised SoC architecture to find all Pareto optimal configurations in a multi-objective design space. The approach uses a parameter dependency model of the target architecture to extensively prune non-optimal design spaces. Locally, the approach applies genetic algorithm to discover Pareto optimal configurations within the remaining design points. Srinivasan et al. [24] compare explorations driven by simulated annealing with results using an evolutionary approach. The work described by Shee et al. [25] consider streaming application which is manually partitioned into a series of algorithmic stages and formulates a heuristic to efficiently search the design space for a pipeline based multi-ASIP system. Anderson and Khalid [26] have proposed a framework for design space exploration, which performs initial configuration sweep for

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

collecting architectural information, and use genetic algorithm for the design space exploration.

A Generalized Linear Model (GLM) and genetic algorithm based approach for efficiently exploring bus-based communication architectures of MPSoCs has been proposed by Esmeraldo and Barros [27]. The GLM is a statistical method based on the technique of Maximum Likelihood, in which parameters of a generalized linear model can be estimated when the errors follow an exponential family distribution. In order to train the model, sufficient number of simulation test results is needed. To reduce the number of training sets needed, genetic programming has been employed. Javaid et al. [28] have proposed a novel methodology of exploring the Application Specific Instruction Processors (ASIP) design space. The methodology first builds a pipeline of processors targeted for executing streaming applications. Initially, a heuristic has been used to rapidly explore a large number of processor configurations to find near Pareto Front of the design space, and then an Exact Integer Linear Programming Formulation (EIF) has been used to find an optimal solution. A Reduced ILP Formulation (RIF) has been used if the EIF does not find an optimal solution in the given time window.

An efficient two step DSE framework has been proposed for high performance MPSoC based on synchronous data flow specification by Lee et al. [29]. In the first step, the mapping avoidance and pinning-first co-synthesis algorithms allow the execution of multiple tasks. In the next step, the Pareto-optimal set of on chip bus architectures has been obtained by using the static performance analysis and trace-driven simulation. This is a recent work that considers the exploration of both the architecture and communication design spaces. Three different approaches such as multi-objective genetic algorithm, multi-objective Simulated Annealing, and multi-objective Tabu search have been used by Ceriani et al. [30] for simultaneously exploring the architecture, mapping and scheduling of the system considering multi-rate real-time applications with certain objectives to be optimised such as area, hard and soft deadlines, and dimensions of memory buffers.

A new online design space exploration algorithm - CAPS has been proposed by Liu et al. [31] which iteratively invokes a commercial tool (the oracle) to synthesise various instances of the components and implicitly builds approximations for their Pareto sets. Ascia et al. [32] compare the performance of various Multi Objective Evolutionary Algorithms (MOEA) considering real-time applications for demonstrating the scalability and accuracy of the approaches. Pomata et al. [33] have introduced a methodology enabling the use of FPGA based prototyping for micro-architectural DSE of ASIPs. To increase the emulation speed-up, the proposed technique exploits the translation of application binary code compiled for a custom ASIP architecture, into code executable on a different configuration. A reliability aware DSE has been proposed by Jia Huang et al. [34], considering both the temporal and spatial redundancy. The work also considers both the transient and intermittent faults with imperfect fault detectors during the reliability analysis. Multi-objective evolutionary algorithm has been implemented for incorporating the results of the reliability analysis for performing automated DSE.

3.3 Techniques Based On Statistical Analysis For Guiding The Dse Process

A predictive reaction – based model for DSE based on Artificial Neural Network (ANN) has been proposed by Khan et al. [35]. Given only a small fraction of the sample training set of the design space, the model is able to accurately predict the behaviour of the remaining designs, orders of magnitude faster than simulating them. The proposed model can predict performance metrics not only for unseen configurations for a given application, but also for unseen configurations of a new application that was not in the set of applications used to build the model, given only a very small number of results for this new application. A hybrid approach for DSE has been proposed by Ascia et al. [36]. The approach tackles the problem of DSE by reducing the number of evaluations and the time required to evaluate a system configuration. The proposed methodology uses Evolutionary Algorithm (EA) as an optimization technique and Fuzzy System (FS) for the estimation of the objective function. The EA explores the design space normally while the FS learns from the simulation until it becomes an expert and it can be used for estimating system performance.

Statistical simulation based guidance for DSE has been introduced based on memory data flow models by Genbrugge and Eeckhout [37]. Cache miss correlation, cache lines reuse distribution and through-memory read-after-write distribution modelling has been incorporated. The basic idea of this statistical modelling is to collect a number of important program characteristics and to

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

generate a synthetic trace from it. A guided DSE approach based on packing has been proposed by Ristau et al. [38], in which the performance of a single task on a specific processor has been estimated and used in the guidance step for mapping the task on to a processing element. The methodology described by Genbrugge et al.[39] studies statistical simulation as a fast simulation technique for Chip Multiprocessors (CMP) design space exploration which enhances the typical statistical simulation by modelling the memory address stream behaviour in a more micro-architectural independent way and modelling programs time-varying execution behaviour. Scenario – based DSE has been proposed by Van Stralen and Pimental [40] for MPSoCs by introducing the concept of workload scenarios, for capturing dynamic behaviour both within and between applications for guiding the DSE process. A co-evolutionary genetic algorithm has been implemented for performing the DSE. Feature selection algorithm [41] has been integrated as an extension to the scenario based DSE, to identify the different multi application workload scenario subset. The representative subsets obtained from the feature selection scheme is utilized in [42] for predicting the fitness of the scenario subsets, in order to improve the efficiency of DSE process and the quality of mapping. Stochastic, deterministic and hybrid prediction techniques have been implemented and their performance with respect to multi workload scenarios has been compared.

A correlation – based DSE approach has been introduced by Mariani et al. [43] for exploiting the correlation properties of the multiprocessor system configurations to accurately choose the most promising configurations to be analysed with a low level simulation. A model has been proposed based on the critical analysis of certain parameters that have been used to guide the DSE process by Navada et al. [44]. The analysis result of the model has been utilised by Simulated Annealing and Random Walk algorithms to perform the DSE. A semi-supervised learning algorithm (COMT) has been introduced by Guo et al. [45] for predicting the application's execution behaviour within the unlabelled design configuration. COMT significantly improves prediction accuracies and reduces excessive simulation costs.

A hybrid Genetic Algorithm – Machine Learning (GA-ML) based predictive model design space exploration has been proposed by Schafer et al. [46]. The technique uses the initial training set for reducing the number of simulations during high level synthesis. Even though the GA-ML technique scales well when compared to standard Genetic Algorithm (GA) and Simulated Annealing (SA) approaches, to further improve and accelerate the DSE process, a Divide and Conquer Design Space Exploration Algorithm (DC-ExpA) [47] has been proposed. The DC-ExpA uses clustering mechanism to group the operations and relates the attributes with clusters to achieve the optimal design points. The work proposed by Mark Thompson and Andy.D.Pimentel [48] extends the standard GA to guide and optimize the search behaviour through the exploitation of the domain knowledge of the system parameters. Reduction of redundancy in the chromosome representation and the implementation of cross over operator based on the distance metric have been the key in imparting knowledge about the system parameters.

After having described the various strategies for covering the design space, the following section presents a review of methods used for evaluating a single design point which is the most important phase of the DSE process.

4. METHODS FOR EVALUATING SINGLE DESIGN POINT

Evaluating a single design point has been considered a key element in the DSE process. The evaluation typically measures objectives like area, energy and timing during exploration. Generally, simulation-based evaluation and analytical model based evaluation are the two evaluation methods used for validating a single design point. Few literatures have considered the mixture of both methods for the evaluation of a single design point. In order to verify the accuracy of the exploration process, a definite set of experimental setup is required by every evaluation technique, which is formally known as benchmarking. The next sub section introduces a few established benchmarks used for the evaluation purpose.

4.1 Established Benchmarks

A benchmark includes a description of the application, a description of the architecture under test, constraints on the workload, a feasible mapping of the application onto the architecture, and defined metrics and cost functions. Available benchmarks can be classified according to their application domain. Some examples are,

- Network Processing: Benchmark suite for network processors is

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

available from NetBench [49] and Networking 2.0 from Embedded Microprocessor Benchmark Consortium (EEMBC) [50].

- General Purpose Computing: Standard Performance Evaluation Corporation (SPEC) provides benchmarks for general – purpose and scientific computing [51].
- Embedded Systems: Benchmark Collections from EEMBC [50], MiBench [52] and ALPBench [53], provides benchmarks for embedded systems applications such as, automotive, telecommunication, consumer, and control systems.
- Multimedia – Centric Computing: Benchmarks focusing on multimedia processing can be found in MediaBench [54].
- Parallel Computing: Example of benchmarks for parallel computing and multiprocessing is Stanford Parallel Applications for Shared Memory SPLASH2 [55], Rodinia [56] for heterogeneous parallel computing, and PARSEC 2.1 [57].

The two most widely used design point evaluation methodologies found in the literature are simulation based evaluation and analytical based evaluation. The following sub sections briefly describe both the evaluation methods.

4.2 Simulation Based Evaluation Approaches

Simulation based evaluation begins with initially modelling the behaviour of the system under consideration with a set of inputs. The inputs have been used to trigger the system for validating the execution characteristics. Simulations have been particularly suited to investigate dynamic effects in the system. The system architect uses different levels of abstraction for the representation of both the application and architecture while modelling the system. Ptolemy [58], a system level simulation framework uses different Models of Computation (MoC) through which several behaviours of the desired application and target architecture has been annotated at various abstraction levels for the evaluation.

The work proposed by Lieverse et al. [59] annotates the application as a single event, which has been modelled based on its computational requirement using the Kahn Process Network (KPN) [60] model. The restriction with the KPN is that the time-dependent behaviour cannot be modelled. The Artemis [61] framework has re-structured the KPN model by incorporating the concept of virtual processors and bounded buffers in order to resolve the occurrence of deadlock. To increase the accuracy of the evaluation process, the work described in [8, 16, 29, 35, and 62] use instruction-set simulation on a cycle- accurate level. The application under evaluation has been described in High Level Language (HLL), or in Assembly language for performing simulation at cycle-accurate level. Micro-architectural features such as cache and branch prediction have been evaluated for the application under consideration, using cycle accurate simulator models such as, Simplescalar [63] and SESC [64].

In order to bridge the gap in existing design methodology between high level models and Hardware Description Language (HDL), the Electronic System Level (ESL) design community introduced SystemC specification language, based on the standard known as Open SystemC Initiative (OSCI) [65]. The significance of SystemC is that, it incorporates the domain knowledge of C and C++ programming languages for system level modelling and evaluation. The OSCI group continuously upgrades the standard by incorporating new Application Programmers Interface (API) for the ease of designers. The latest standard is the SystemC Transaction Level Modelling (TLM) 2.0 in which modelling and development of complex system can be performed more efficiently. Simulation based evaluation techniques require re-targetable compilers, to support modifications that affect instruction set or data path during compilation of every application, considering all micro-architectural features. To automate the modification and mapping process, Architecture Description Languages (ADL) such as MESCAL [66], EXPRESSION [67] and LisaTek [68] have been proposed.

An executable model is often necessary for performing simulation based evaluation which is the major drawback of this evaluation technique. Moreover, the simulated workload must be chosen by the designer in a way that it represents a variety of typical working scenarios to avoid the optimization of the design for a special case.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4.3 Analytical Based Evaluation Methods

Analytical methods have been used to evaluate a design for a class of workloads in a single pass. Analytical methods always follow a pessimistic approach, which considers deterministic or worst – case behaviour with a reasonable assumption for the system under evaluation. In addition, building an executable model of the system as well as simulations might be too costly or even impossible at the time of the evaluation [6]. The analytical method has been mostly used for performing complexity analysis of algorithms, dependency analysis of scheduling of tasks, and function call-graph analysis for determining the worst-case behaviour based on static profiling. Typical usage of analytical models for evaluating different design options can be found in [69] and [70]. The analytical approaches often provide less precise results than simulation based evaluation technique.

Table 1 presents the comparison of various techniques that have been used recently for covering the design space. The comparison is performed by considering parameters such as the representation scheme used for both the application and architecture considered, the simulators used for evaluating the single design point, the target architecture considered, the DSE methodology employed and the objectives that are taken into consideration during the exploration process.

5 FRAMEWORKS AVAILABLE FOR DESIGN SPACE EXPLORATION

This section provides a brief description of the various DSE frameworks/tools available. Followed by the description, Table 2 provides the comparison of various DSE frameworks with respect to the application/architecture models considered, the evaluation methodology and the exploration mode used.

The framework Ptolemy [58] studies the modelling, simulation and design of the concurrent, real-time embedded systems, for analyzing the interaction between the system components, using heterogeneous mixture of Models of Computation (MoC). SPADE [59], a system level framework, presents an architectural exploration method based on a single MoC, namely KPN. This framework follows the Y-chart principle for system level design based on various abstraction levels.

EXPRESSION [67] enables the modelling of a single programmable processor with its memory sub system. The simulator, compiler, and VHDL descriptions of the processor can be generated easily from the specifications of the processor. Lisa Tek [68] is another ADL for programmable processor, from which a cycle accurate simulator, assembler, and debugger can be generated. SpecC [78] is a modelling language that includes constructs like abstraction, orthogonalization of concerns and clearly defined interfaces, facilitating increased design reuse.

GRACE++ [79] is a SystemC based simulation environment for Network on Chip (NoC) centric MPSoC that targets abstraction levels higher than Register Transfer Logic (RTL) to achieve increased simulation speed. Co-centric System Studio Tool [73] is a SystemC based modelling environment for embedded system design. The tool supports different levels of abstraction and powerful debugging. ARTEMIS [61] yet another system level framework, has extended the work of SPADE by increasing the simulation speed and incorporating automatic heuristics for DSE. Mentor Seamless [80], an Electronic Design Automation (EDA) tool, enables the users to debug hardware/software integration issues early in the design cycle and supports combined simulation of HDL and ISS. StepNP framework [81] represents a development environment including the design aspects of architecture, application and tools. The main focus of this framework is to explore different network processor architectures.

EXPO [70] is a system level analytical DSE tool that targets the applications and SoC architecture in the domain of packet processing. PICO [82] framework constitutes one VLIW processing core, cache hierarchy, and one or several coprocessors. A coprocessor contains a number of functional units organized as a systolic array to accelerate compute-intensive loop nests of the original application description. MESCAL [69] aims at the design of heterogeneous, application - specific programmable multi processors. This framework is based on ADL, in which applications can be described using different combinations of MoCs. The framework MESH [83] raises the abstraction level from cycle-accurate modelling. Different techniques have been introduced to capture hardware effects, like complex processing elements and communication architectures by considering schedulers and scheduling operations as the key element.

MILAN [15] is a hierarchical DSE framework which includes various design space pruning techniques based on constraint satisfaction and symbolic search techniques. Metropolis [84]

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

includes modelling of Finite State Machine (FSM) at various abstraction levels, and incorporates simulation, modelling, synthesis, and validation tools. CASSE [85] is a fast, flexible, and modular SystemC based simulation environment for DSE and system level design at various abstraction levels. ARTS [86] is a SystemC based abstract system-level modelling and simulation framework, which allows the designers to model, analyze the components used at various levels of the design, and the interactions between them.

MPARM [62] a cycle accurate architectural simulator, focuses on the system level analysis and design tradeoffs in the usage of different processors, interconnections, memory hierarchies and other devices. SESAME [75] uses discrete event simulation language to implement its architectural models and extends the Artemis framework, thereby improving the simulation accuracy by incorporating fine-grained architectural models and implementing an automatic DSE technique. Hardware platforms based on Component based Design (CbD) has been addressed in the ROSES [87] framework. The major advantage of this framework is its reduced design effort and quick design composition and investigation.

Daedalus [88] another system level framework, incorporates the activities of the Sesame project and facilitates the designer to bridge the gap between system-level designs based on Sesame methodology and the final RTL implementation. MAMPS [89] is a design flow for mapping throughput constrained applications on MPSoC. It integrates several state-of-the-art mapping and synthesis tools into an automated tool flow. MULTICUBE [90] an automatic DSE framework, integrates the available standard evolutionary algorithms to be used at design time to find the best power and performance trade-offs. RSM is used for reducing the number of simulations. OPEN SCALE [91] is a tool for performing DSE for heterogeneous embedded multiprocessor systems with Network on Chip (NoC). FADSE [92] framework performs automatic DSE by integrating various simulators, thus enabling the exploration process to be completed in parallel across the computers. ArcheOpterix [93] framework is composed of several algorithms for performing DSE. The platform provides support for specifying, evaluating and optimizing the architecture of the component based software systems, taking into consideration the presence of uncertainty within the design parameters.

HeMPS [94] is an environment that comprises of different MPSoC models and Network On-Chip (NoC) models for aiding the DSE process with support for Instruction Set Simulation (ISS), Synthesizable VHDL and validation. DIPLODOCUS [95], a formal system level DSE framework, supports the analysis of the system at a lower abstraction, and formal verification. To perform formal analysis before and after mapping, a formal semantics has been provided within the framework to tasks, communication between tasks and channels onto hardware architectures. Heracles [96] is an open source tool based on fast Register Transfer Level (RTL) for performing DSE of multi core processors. The tool has been designed with high degree of modularity for supporting fast exploration of processors with different topologies, memory organization and routing schemes. MGSim [97] is an open source discrete event simulator for on-chip hardware components. The component library of MGSim includes support for core models with different instruction sets, multiple configurable caches with memory models, a configurable multi core interconnect, a dedicated I/O system and comprehensive monitoring and interacting facilities.

6. OPEN RESEARCH ISSUES IN MPSoC DSE

This section provides various research issues involved during MPSoC DSE,

a) Selection of processor

The first and foremost challenge for the system designer is to select the processor for the given application specification. Since the applications impose conflicting constraints such as power/performance trade off, the best choice for the designers is the heterogeneous processor which satisfies the specified constraints with better performance.

b) Mapping of Application Tasks onto Processors

The given application has to be divided into several tasks which are capable of executing in different processors. The designer has to select a suitable interconnection network for enabling processor to processor communication and synchronization.

c) Domain Knowledge based DSE

The DSE approaches which do not have any knowledge about the target platform are not suitable for exploring the corner cases of the Pareto points. In order not to compromise for the optimal

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

design solutions, domain knowledge about the target architecture should be incorporated in the DSE strategy. The inheritance of domain knowledge will definitely improve the search process toward the optimal and best design options.

d) Automatic DSE

Several researchers are still working toward providing necessary tools for MPSoC design. The current design flow does not inherently include the complete DSE process. The challenge for the designer is to incorporate a complete DSE strategy integrated with MPSoC design flow, which will automate the entire process. The integration of DSE strategy with the MPSoC design flow will provide a concrete platform for system architect to explore different options and choose best design for the given application.

7. SUMMARY AND CONCLUSION

This paper has provided a comprehensive survey of DSE techniques employed for MPSoC architectures. Various techniques implemented for covering the design space and evaluating a single design point have been briefly described. There are a number of framework/tools available for the design and synthesis of MPSoCs. A brief comparison of the available framework/tools has been described based on the various key parameters.

The following observations outline the prospects for future research:

a) Most methodologies found in the literature use GA and heuristic approaches for covering the design space. The disadvantage of using GA or heuristic techniques is that the optimal solution will not be always possible.

b) Very few implementations consider the analytic modelling of both the application characteristics and target architecture. Analytic modelling will aid the exploration process for better convergence towards getting an optimal solution.

c) A generic DSE framework without any assumptions of the target architecture and applications has not been proposed.

d) Automated DSE approaches are still at an early stage of research. There is a demand for a complete EDA tool in the design of MPSoC.

e) NoC based MPSoCs are very popular nowadays, but there are very few articles that are reported in the literature which provide DSE techniques for these types of architectures.

f) The MPSoC design community needs a well established and standardised DSE framework, which can be regarded as a benchmark and can be easily integrated with the commercially available EDA tools.

The other challenges include identification of application characteristics, program feature-driven DSE and clustering of applications which share the same promising architectural configurations

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1: Comparison Of Techniques Used For Covering Design Space

S.no	Author Name	Representation of Application	Representation of Architecture	Simulator/ Framework Used	Target Architecture Considered	DSE Technique Employed	Objectives Considered
1	Shee, et al. (2008)	High level language (HLL), C	Tensilica Xtensa Processor Core	Tensilica Instruction Set Simulator (ISS)	Heterogeneous ASIP	Manual/ Exhaustive	Area
2	Hameed, et al. (2011)	High level language (HLL), C	Customisable, Extensible Functional Units	Tensilica ISS	Homogeneous CMP	Manual	Energy and Power
3	Jung, et al. (2011)	Thread Level	Queuing Model For CPU and Memory	Simulation Tool in [71]	Multi-core	Exhaustive	Performance
4	Brandolese, et al. (2006)	Procedural Interaction Graph (PIG)	SystemC	Timing Simulation	Heterogeneous MPSoC	Genetic Algorithms (GA)	Performance with Affinity metrics
5	Palermo, et al. (2009)	High level language (HLL), C	Abstract Simulation Model	SESC	Homogeneous MPSoC	DoE, RSM	Performance
6	Beltrame et al. (2010)	High level language (HLL), C	CMP Simulation Platform	ReSP [72] Simulation Tool	Homogeneous CMP	Decision Theory with MDP	Performance
7	Yazdanbakhsh , et al. (2010)	High level language (HLL), C/C++	Register File Modelling Using Synopsis Compiler [73]	VEX Tool Chain [74]	Homogeneous	Mathematical Model for Energy Aware DSE	Energy, Area-Delay
8	Kokhazadeh, and Fatemi (2011)	Kahn Process Network (KPN)	Abstract Graph Based	SESAME Framework [75]	Heterogeneous MPSoC	Weighted Sub sampling Algorithm	Performance
9	Piscitelli, and Pimentel (2012)	KPN	Abstract Graph Based	SESAME Framework	Heterogeneous MPSoC	Analytical Throughput Analysis with GA	Performance
10	Palermo, et al. (2012)	High level language (HLL), C	Shared memory Chip Multiprocessor (CMP)	SESC	Homogeneous MPSoC	DoE, RSM with Process Variation	Energy, Power
11	Javaid, et al. (2010)	High level language (HLL), C	Tensilica Xtensa Processor Core	Tensilica Instruction Set Simulator (ISS)	Heterogeneous ASIP	ILP and Heuristics	Area, Energy
12	Lee, et al. (2010)	Synchronous Data Flow Graph (SDF)	Abstract Architectural Templates	Modelsim HDL Simulator	Heterogeneous MPSoC	Mapping Algorithms and Comparison	Performance
13	Ceriani, et al. (2010)	Abstract Task Graph	Abstract	Xilinx Synthesis Tools	Heterogeneous MPSoC	Multi Objective GA's	Area, Hard and Soft Deadlines
14	Ascia, et al. (2011)	High level language (HLL), C	Very Large Instruction Word Architecture (VLIW) Template	EPIC Explorer [76]	Heterogeneous	Performance Comparison of Various GA's	Area, Energy, Performance
15	Pomata, et al. (2012)	High level language (HLL), C	Verilog High Definition Language	FPGA Synthesis Tool	Heterogeneous ASIP	Binary Translation Algorithm	Execution Time, Area, Power.
16	Jia Huang, et al. (2013)	Directed Acyclic Graph (DAG)	Heterogeneous Multiprocessor Simulation Platform	GENESYS Architecture Platform [77]	Heterogeneous Multiprocessor	Multi Objective Evolutionary Algorithm (MOEA)	Performance

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

17	Khan, et al. (2007)	Thread Level Speculation (TLS)	CMP Architectural Template	SESC Simulator	Homogeneous	Predictive Modelling based on ANN	Energy – Delay
18	Ascia, et al. (2007)	High level language (HLL), C	Very Large Instruction Word Architecture (VLIW) Template	EPIC Explorer	Heterogeneous	GA and Fuzzy Approach	Area, Energy, Performance
19	Ristau, et al. (2009)	Conditional Data Flow Graphs (CDFG)	SAMIRA Vector Digital Signal Processor (DSP)	Synchronous Transfer Architecture	Heterogeneous	Guided DSE based on Packing	Performance
20	Genbrugge, and Eeckhout (2009)	Statistical Flow Graph (SFG)	M5 Architectural Tool	M5 Simulator	Heterogeneous	Statistical Analysis	Performance
21	Van Stralen, and Pimentel (2010)	KPN	Abstract Graph Based	SESAME Framework	Heterogeneous	Co evolutionary GA & Scenario Based Guidance	Performance
22	Mariani, et al. (2010)	Parallel Application from SPLASH	Abstract Architectural Model	SESC Simulator	Homogeneous	Correlation based	Performance
23	Navada, et al. (2010)	High level language (HLL), C	Superscalar Architectural Template	Simple Scalar Simulator	Homogeneous	Criticality based Guidance for DSE	Performance
24	Guo, et al. (2011)	High level language (HLL), C	Abstract Architectural Model	Simple Scalar Simulator	Homogeneous	Prediction Based Learning Algorithm	Performance
25	Van Stralen, and Pimentel (2012)	KPN	Abstract Graph Based	SESAME Framework	Heterogeneous	Co evolutionary GA & Scenario Based Guidance, with Feature Selection Algorithm	Performance
26	Van Stralen, and Pimentel (2013)	KPN	Abstract Graph Based	SESAME Framework	Heterogeneous	Co evolutionary GA & Scenario Based Guidance, with fitness prediction	Performance
27	Mark Thompson and Pimentel (2013)	KPN	Abstract Graph Based	SESAME Framework	Heterogeneous	GA, with domain knowledge	Performance

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2: Comparison Of Frameworks Used For Design Space Exploration

S.no	Framework	Application Model Used	Architecture Model Used	Evaluation Methodology	Exploration Mode
1	Ptolemy (1999)	KPN, Discrete Event, DFG, SDF	Abstract	Simulation Based	Manual
2	SPADE (1999)	KPN	Abstract Performance Model	Instruction-Accurate Simulation	Manual
3	EXPRESSION (1999)	C++	ADL	Simulation Based	Manual
4	LisaTek (1999)	Assembler	ADL	Cycle Accurate Simulation	Manual
5	SpecC (2000)	Spec C Language	Abstract	Analytical Model	Manual
6	GRACE++ (2001)	Abstract Graph Based	SystemC	Simulation Based	Manual
7	Co-Centric System Studio Tool (2000)	FSM, SDF, MATLAB, SystemC	Abstract to HDL	High – Level Synthesis	Manual, Scripts
8	ARTEMIS (2001)	KPN	Abstract	Instruction Accurate	Automatic Heuristics
9	Mentor Seamless (2001)	HDL	Abstract to HDL	Instruction Set Simulators	Manual
10	StepNP (2002)	Click MoC	Abstract	ISS	Manual
11	EXPO (2002)	DAG task level	Abstract	Analytical Model Based Evaluation	GA based Evolutionary Optimizers
12	PICO (2002)	C	Micro architectural Templates	Simulation Based	Automatic Heuristics
13	MESCAL (2002)	Mixed MoC	Architectural Description Language (ADL)	Analytical Model Based Evaluation	Partially Automatic
14	MESH (2002)	Abstract	Abstract	Cycle Accurate Simulation	Manual
15	MILAN (2002)	MATLAB, SystemC	SystemC	Cycle Accurate Simulator	Manual
16	Metropolis (2003)	Mixed MoC	Abstract	Simulation Based	Manual
17	CASSE (2004)	KPN	SystemC	Simulation Based	Manual

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

18	ARTS (2005)	Task Graph	SystemC	Simulation Based	Manual
19	MPARM (2005)	Abstract Description	SystemC	Cycle Accurate Simulator	Manual
20	SESAME (2006)	KPN, Integer Controlled Data Flow (IDF)	SystemC TLM	Simulation Based	Automatic GA based (SPEA2)
21	ROSES (2007)	Abstract	IP Component based Design	Simulation Based	Manual
22	Daedalus (2008)	KPN,IDF	SystemC, TLM	Simulation Based	Automatic
23	MAMPS (2011)	KPN	Abstract	Xilinx Synthesis Evaluation	Manual
24	MULTICUBE (2011)	C	Abstract XML based	Simulation Based	Automatic Heuristics
25	OPEN-SCALE (2011)	KPN	Secretblaze Architectural Template	Simulation Based, With Real Time Operating System (RTOS)	Manual
26	FADSE (2011)	C	No Specific Architectural Models	Simulation Based Evaluation	Automatic
27	Arche Opterix (2012)	ADl, XML	Abstract	Simulation Based Evaluation	Several Algorithms
28	HeMPS (2012)	Task Graphs	Low level Abstract Models	Simulation Based Evaluation	Manual
29	DIPLODOCUS (2013)	Unified Modelling Language (UML)	UML Specification	Simulation Based Evaluation	Manual
30	Heracles (2013)	HLL	Component based hardware	Simulation Based Evaluation	Manual
31	MGSim (2013)	HLL	Abstract Component Libraries	Simulation Based Evaluation	Manual

REFERENCES:

[1] W. Wolf, A.A. Jerraya, G. Martin, "Multiprocessor System on Chip (MPSoC) Technology", *IEEE Transaction on Computer Aided Design, Vol. 2*, 2008, pp. 1701-1713.

[2] Apple Inc, http://www.apple.com/, accessed July 2012.

[3] Y.K. Chen, S.Y Kung, "Trend and Challenge on System on Chip Designs", *Journal of Signal Processing System, Vol. 53*, 2008, pp. 217-229.

[4] G.Martin, "Multiprocessor SoC Based Design Methodologies Using Configurable and Extensible Processors", *Journal of Signal Processing System, Vol. 53,* 2008, pp. 113-127.

[5] T. Kempf, G. Ascheid, R. Leupers, "*Multiprocessor System on Chip – Design Space Exploration"* (1st Edition, Springer, 2011).

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[6] M. Gries, "Methods for Evaluating and Covering the Design Space during Early Design Development", *Technical Report UCB/ERL M03/32, Electronics Research Laboratory, University of California at Berkeley*, 2003.

[7] B. Kienhuis, E. Deprettere, K. Vissers, P. Van Der Wolf, "*An Approach for Quantitative Analysis of Application – Specific Data Flow Architectures*", Proceedings of the International Conference on Application – Specific Systems, Architectures, and Processors, 1997, pp. 338-349.

[8] S. Blythe, R. Walker, "Efficient Optimal Design Space Characterization Methodologies", *ACM Transactions on Design Automation for Electronic Systems, Vol. 5*, 2000, pp. 322-336.

[9] H. Jung, M. Ju, H. Che, "*A Theoretical Framework for Design Space Exploration of Many core Processors*", *Proceedings of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, 2011, pp. 117-125.

[10] R. Hameed, W. Qadeer, M. Wachs, O. Azizi, A. Solomatnikov, B.C. Lee, S. Richardson, C. Kozyrakis, M. Horowitz, "Understanding Sources of Inefficiency in General – Purpose Chips", *Communications ACM, Vol. 54,* 2011, pp. 85-93.

[11] S.L. Shee,Sri Parameswaran, "*Design Methodology for Pipelined Heterogeneous Multiprocessor System*", *Proceedings of the International Conference on Design Automation,* 2007, pp. 811-816.

[12] Xtensa Processor. Tensilica Inc. http://www.tensilica.com/, accessed July 2012

[13] V. Pareto, "Manuel deconomie Politique, *Bullitan of American Mathematical Society*", *Vol. 18*, 1912, pp. 462-474.

[14] G. Beltrame, L. Fossati, D. Sciuto, "Decision – Theoretic Design Space Exploration of Multiprocessor Platforms", *IEEE Transaction on Computer Aided Design, Vol. 29*, 2010, pp. 1083-1095.

[15] S. Mohanty, V.K.Prasanna, S. Neema, J. Davis, "*Rapid Design Space Exploration of Heterogeneous Embedded Systems Using Symbolic Search and Multi-granular Simulation*", Proceedings of the International Workshop on Languages, Compilers and Tools for Embedded Systems, 2002, pp. 18-27.

[16] C. Brandolese, W. Fornaciari, L. Pomante, F. Salice, D. Sciuto, "Affinity – Driven System Design Exploration for Heterogeneous Multiprocessor SoC", *IEEE Transaction on Computers, Vol. 5,* 2006, pp. 508-519.

[17] G. Palermo, C. Silvano, V. Zaccaria, ReSPIR – "A Response Surface – Based Pareto Iterative Refinement for Application – Specific Design Space Exploration", *IEEE Transaction on Computer Aided Design, Vol. 28,* 2009, pp. 1816-1829.

[18] A. Yazdanbakhsh, M. Kamal, M.E. Salehi, H. Noori, S.M. Fakhraie, "*Energy-Aware Design Space Exploration of Register File for Extensible Processors*", Proceedings of the International Conference on Embedded Computer Systems, Architecture, Modeling and Simulation, 2010, pp. 273-281.

[19] A. Kokhazadeh, O. Fatemi, "*Design Space Pruning of MPSoC's Using Weighted Sub – Sampling*", Proceedings of the IEEE International Conference on Electronics, Circuits and Systems, 2011, pp. 550-553.

[20] R. Piscitelli, A.D. Pimentel, "*Design Space Pruning through Hybrid Analysis in System Level Design Space Exploration*", Proceedings of the International Conference on Design Automation, Test in Europe, 2012, pp. 781-786.

[21] G. Palermo, C. Silvano, V. Zaccaria, "A Variability Aware Robust Design Space Exploration Methodology for On-Chip Multiprocessors subject to Application-Specific Constraints", *ACM Transactions*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

on Embedded Computing Systems. Vol. 11, no. 2, 2012, pp. 29:1 – 29:28.

[22] Hung-Yi Liu, Michele Petracca, Luca.P. Carloni, "*Compositional System-Level Design Space Exploration with Planning of High Level Synthesis*", Proceedings of the International Conference on Design Automation, Test in Europe, 2012, pp. 641-646.

[23] M. Palesi, T. Givargis, "*Multi – Objective Design Space Exploration Using Genetic Algorithms*", Proceedings of the IEEE International Symposium on Hardware/Software Co-Design, 2002, pp. 67-72.

[24] V. Srinivasan, S. Radhakrishnan, R. Yemuri, "*Hardware Software Partitioning with Integrated Hardware Design Space Exploration*", Proceedings of the International Conference on Design Automation, Test in Europe, 1998, pp. 28-35.

[25] S.L. Shee, A. Erdos, Sri Parameswaran, "Architectural Exploration of Heterogeneous Multiprocessor Systems for JPEG", *International Journal on Parallel Processing, Vol. 36,* 2008, pp. 140-162.

[26] I.D.L Anderson, M.A.S. Khalid, "SC build: a Computer – Aided Design Tool for Design Space Exploration of Embedded Central Processing Unit Cores for Field Programmable Gate Arrays", *IET Computers and Digital Techniques, Vol. 3*, 2009, pp. 24-32.

[27] G. Esmeraldo, E. Barros, "*A Genetic Programming Based Approach for Efficiently Exploring Architectural Communication Design Space of MPSoC's*", Proceedings of the IEEE International Conference on Programmable Logic Conference, 2010, pp. 29-34.

[28] H. Javaid, A. Ignjatovic, Sri Parameswaran, "Rapid Design Space Exploration of Application Specific Heterogeneous Pipelined Multiprocessor Systems". *IEEE Transaction on Computer Aided Design, Vol. 29*, 2010, pp. 1777-1789.

[29] C. Lee, S. Kim, S. Ha, "A Systematic Design Space Exploration of MPSoC Based on Synchronous Data Flow Specification", *Journal of Signal Processing Systems, Vol. 58*, 2010, pp. 93-213.

[30] M. Ceriani, F. Ferrandi, P.L Lanzi, D. Sciuto, A. Tumeo, "*Multiprocessor System on Chip Synthesis Using Multi – Objective Evolutionary Computation*", Proceedings of the International Conference on Genetic and Evolutionary Computation, 2010, pp. 1267-1274.

[31] H.Y. Liu, I. Diakonikolas, M. Petracca, L. Carloni, "*Supervised Design Space Exploration by Compositional Approximation of Pareto Sets*", Proceedings of the International Conference on Design Automation, 2011, pp. 399-404.

[32] G. Ascia, V. Catania, A.G.D. Nuovo, M. Palesi, D. Patti, "Performance Evaluation of Efficient Multi – Objective Evolutionary Algorithms for Design Space Exploration of Embedded Computer Systems", *Applied Soft Computing, Vol. 11,* 2011, pp. 382-398.

[33] S. Pomata, P. Meloni, G. Tuveri, L. Raffo, M. Lindwer, "*Exploiting Binary Translation for Fast ASIP Design Space Exploration on FPGA's*", Proceedings of the International Conference on Design Automation, Test in Europe, 2012, pp. 566-569.

[34] Jia Huang, Andreas Raabe, Kai Huang, Christian Buckl, Alois Knoll, "A Framework for Reliability-Aware Design Exploration on MPSoC Based Systems", *Design Automation for Embedded Systems, Vol. 16, 2013, pp. 189-220.*

[35] S. Khan, P. Xekalakis, J. Cavazos, M. Cintra, "*Using Predictive Modeling for Cross – Program Design Space Exploration in Multi core Systems*", Proceedings of the IEEE International Conference on Parallel Architecture and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Compilation Techniques, 2007, pp. 327-328.

[36] G. Ascia, V. Catania, A.G.D. Nuovo, M. Palesi, D. Patti, "Efficient Design Space Exploration for Application Specific Systems on Chip", *Journal of System Architecture. Vol. 53*, 2007, pp. 733-750.

[37] D. Genbrugge, L. Eeckhout, "Memory Data Flow Modeling in Statistical Simulation for the Efficient Exploration of Microprocessor Design Spaces", *IEEE Transaction on Computers, Vol. 57,* 2008, pp. 41-54.

[38] B. Ristau, T. Limberg, G. Fettweis, "A Mapping Framework Based on Packing for Design Space Exploration of Heterogeneous MPSoC's", *Journal of Signal Processing Systems, Vol. 57*, 2009, pp. 45-56.

[39] D.Genbrugge, L.Eeckhout, "Chip Multiprocessor Design Space Exploration through Statistical Simulation", *IEEE Transaction on Computers, Vol. 58*, 2009, pp. 1668-1681.

[40] P.VanStralen, A.D.Pimentel, "*Scenario – Based Design Space Exploration of MPSoC's*", Proceedings of the International Conference on Computer Design, 2010, pp. 305-312.

[41] P.VanStralen, A.D.Pimentel, "*Fast Scenario-Based Design Space Exploration Using Feature Selection*", Proceedings of the International Workshop on Parallel Programming and Run-Time Management Techniques for Many Core Architectures, 2012, pp. 1-7.

[42] P.VanStralen, A.D.Pimentel, "Fitness Prediction Techniques for Scenario-Based Design Space Exploration", *IEEE Transaction on Computer Aided Design, Vol. 32*, n. 8, 2013, pp. 1240-1253.

[43] G.Mariani, G.Palermo, V.Zaccaria, "*A Correlation Based Design Space Exploration Methodology for Multiprocessor Systems on Chip*", Proceedings of the International Conference on Design Automation, 2010, pp. 120-125.

[44] S.Navada, N.K.Choudhary, E.Rotenberg, "*Criticality – Driven Superscalar Design Space Exploration*", Proceedings of the International Conference on Parallel Architecture and Compilation Techniques, 2010, pp. 261-272.

[45] Q.Guo, T.Chen, Y.Chen, Z.H. Zhou, W.Hu, Z.Xu, "*Effective and Efficient Microprocessor Design Space Exploration Using Unlabeled Design Configuration*", Proceedings of the IEEE International Conference on Artificial Intelligence, 2011, pp. 1671-1677.

[46] B.Carrion Schafer, K.Wakabayashi, "Machine Learning Predictive Modeling High Level Synthesis Design Space Exploration", *IET Computers and Digital Techniques, Vol. 6,* 2012, pp. 153-159.

[47] B.Carrion Schafer, K.Wakabayashi, "Divide and Conquer High Level Synthesis Design Space Exploration", *ACM Transaction on Design Automation for Electronic Systems, Vol. 17,* no. 3, 2012, pp. 29:1-29:19.

[48] Mark Thompson, Andy.D.Pimentel, "Exploiting Domain Knowledge in System-Level MPSoC Design Space Exploration", *Journal of System Architecture*, Vol. 59, no. 7, 2013, pp. 351-360.

[49] G.Memik, W.H. Mangione – Smith, W. Hu, "*NetBench: A Benchmarking Suite for Network Processors*", *Proceedings of the International Conference on Computer – Aided Design*, 2001, pp. 39-43.

[50] Embedded Microprocessor Benchmark Consortium (EEMBC). http://eembc.org/. (2012).

[51] SPEC Benchmark Suite. http://www.spec.org/, accessed March 2012

[52] M.Guthaus, J.Ringenberg, D.Ernst, T.Austin, T.Mudge, R.Brown, "MiBench: A free, Commercially Representative Embedded Benchmark Suite", *Proceedings of the IEEE International Annual Workshop on Workload Characterization*, 2001, pp. 3-14.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[53] M.Li, R.Sasanka, S.Adve, Y.K.Chen, E.Debes, "TheALPBench Benchmark Suite for Complex Multimedia Applications", *Proceedings of the IEEE International Annual Workshop on Workload Characterization Symposium*, 2005, pp. 34-45.

[54] C.Lee, M.Potkonjak, W.H.Mangione – Smith, "*MediaBench: A Tool for Evaluating and Synthesizing Multimedia and Communications Systems*", *Proceedings of the IEEE/ACM International Symposium on Micro-architecture*, 1997, pp. 330-335.

[55] S.C Woo, M. Ohara. E. Torrie, J.P. Singh, A. Gupta, "SPLASH-2 Programs: Characterization and Methodological Considerations", *Proceedings of the International Symposium on Computer Architecture*, 1995, pp. 24-36.

[56] S. Che, M. Boyer, J. Meng, D. TarjanSheaffer, S.H. Lee, S.H, K. Skadron, "*Rodinia: A Benchmark Suite for Heterogeneous Computing*", *Proceedings of the International Symposium on Workload Characterization*, 2009, pp. 44-54.

[57] Parsec 2.1: Parallel Benchmark Suite. http://parsec.cs.princeton.edu/ accessed July (2011).

[58] E.A. Lee, "Overview of the Ptolemy Project", *Technical Report UCB/ERL M03/25, University of California*, 2003.

[59] P. Lieverse, P. Van Der Wolf, K. Vissers, E. Deprettere, "A Methodology for Architectural Exploration of Heterogeneous Signal Processing Systems", *Journal of Kluwer VLSI Signal Processing, Vol.29,* 2991, pp. 197-207.

[60] G. Kahn, "The Semantics of a Simple Language for Parallel Programming", *Proceedings of the International Conference on IFIP Congress,* 1974, pp.471-475.

[61] A.D. Pimentel, L.O. Hertzberger, P. Lieverse, P.Van Der Wolf, E. Deprettere, "Exploring Embedded Systems Architectures with Artemis", *IEEE Computers, Vol. 34*, 2001, pp. 57-63.

[62] L. Benini, D. Bertozzi, A. Bogliolo, F. Menichelli, M. Olivieri, "MPARM: Exploring the Multiprocessor SoC Design Space with SystemC", *Journal of VLSI Signal Processing Systems, Vol. 41,* 2005, pp. 169-182.

[63] D. Burger, T.M. Austin, "The SimpleScalar Tool Set", *Version 2.0, Technical Report 1342*, 1997.

[64] J. Renau, B. Fraguela, J. Tuck, W. Liu, M. Prvulovic, L Ceze, S. Sarangi, P.Sack, K. Strauss, P. Montesinos, SESC Simulator. http://sesc.sourceforge.net/.(2011).

[65] Open SystemC Initiative (OSCI). http://www.systemc.org/. (2012).

[66] A. Mihal, C. Kulkarni, K. Vissers, M. Moskewicz, M. Tsai, N. Shah, S.Weber, Y. Jin, K. Keutzer, C. Sauer, S. Malik, "Developing Architectural Platforms: A Disciplined Approach", *IEEE Design and Test of Computers, Vol. 19,* 2002, pp. 6-16.

[67] A. Halambi, P. Grun, V. Ganesh, A. Khare, N. Dutt, A. Nicolau, "EXPRESSION: A Language for Architecture Exploration through Compiler/Simulator Retargetability", *Proceedings of the International Conference on Design, Automation and Test,* 1999, pp. 485-490.

[68] S.Pees, A. Hoffmann, V. Zivojnovic, H. Meyr, "LISA- Machine Description Language for Cycle Accurate Models of Programmable DSP Architectures", *Proceedings of the International Conference on Design Automation*, 1999, pp. 933-938.

[69] M. Gries, K. Keutzer, "Building ASIP's: The MESCAL Methodology", 2005.

[70] L.Thiele, S. Chakraborty, M. Gries, S. Kunzli, "A Framework for Evaluating Design Tradeoffs in Packet Processing Architectures", *Proceedings of the International Conference on Design Automation*, 2002, pp. 880-885.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[71] H. Jung, M. Ju, H. Che, H, Z. Wang, "A Fast Performance Analysis Tool for Multi core, Multithreaded Communication Processors", *Proceedings of the IEEE International Conference on High Assurance Systems Engineering Symposium*, 2008, pp. 135-144.

[72] G. Beltrame, C. Bolchini, L. Fossati, A. Miele, D. Sciuto, "ReSP – A Non Intrusive Transaction Level Reflective MPSoC Simulation Platform for Design Space Exploration", *Proceedings of the International Conference on Asia South Pacific Design Automation*, 2008, pp. 673-678.

[73] Synopsis System Studio. http://synopsis.com/. (2012)

[74] VEX Tool Chain. www.hpl.hp.com/downloads/vex/. (2011).

[75] A.D Pimentel, C. Erbas, "A Systematic Approach to Exploring Embedded System Architectures at Multiple Abstraction Levels", *IEEE Transaction on Computers, Vol. 55*, 2006, pp. 1-14.

[76] D. Patti, M. Palesi, A.G.D. Nuovo, EPIC Explorer, http://epic-explorer.sourceforge.net/. (2011).

[77] GENESYS. http://www.genesys-platform.eu/ (2013).

[78] R. Domer, "System Level Modeling and Design with the SpecC language", *PhD Thesis*, 2000.

[79] W. Muller, J. Ruf, D. Hoffmann, J. Gerlach, T. Kropf, W. Rosenstiel, "The Simulation Semantics of SystemC", *Proceedings of the International Conference on Design Automation Test in Europe*, 2001, pp. 64-70.

[80] Mentor Graphics Seamless, http://www.mentor.com/. (2011).

[81] E. Bensoudane, P.G Paulin, C. Pilkington, "StepNP: A System-Level Exploration Platform for Network Processors", *IEEE Design and Test of Computers, Vol. 19*, 2002, pp. 17-26.

[82] V. Kathail, S. Aditya, R. Schreiber, B.R Rau, D. Cronquist, M. Sivaraman, "PICO: Automatically Designing Custom Computers", *IEEE Computers, Vol. 35*, 2002, pp. 39-47.

[83] J.M.Paul, D.E. Thomas, "A Layered Co-Design Virtual Machine Approach to Modeling Computer Systems", *Proceedings of the International Conference on Design, Automation and Test in Europe*, 2002, pp. 522-528.

[84] F. Balarin, Y. Watanabe, H. Hsieh, L. Lavagno, C. Passerone, A. SangiovanniVincentelli. "Metropolis: An Integrated Electronic System Design Environment", *IEEE Computers, Vol. 36,* 2003, pp. 45-52.

[85] V. Reyes, T. Bautista, G. Marrero, P.P. Carballo, W. Kruijtzer, "CASSE: A System Level Modeling and Design Space Exploration Tool for Multiprocessor System on Chip", *Proceedings of the International Conference on Euromicro Systems on Digital Systems Design*, 2004, pp. 476-483.

[86] S. Mahadevan, M. Storgaard, J. Madsen, K. Virk, "ARTS: A System Level Framework for Modeling MPSoC Components and Analysis of their Causality", *Proceedings of the IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 2005, pp. 480-483.

[87] F.R Wagner, W. Cesario, A.A. Jerraya, "Hardware/Software IP Integration Using the ROSES Design Environment", *ACM Transaction on Embedded Computing Systems, Vol. 6,* 2007.

[88] H. Nikolov, M. Thompson, T. Stefanov, A.D. Pimentel, S. Polstra, R. Bose, C. Zissulescu, E. Deprettere, "Daedalus: Toward Composable Multimedia MPSoC Design", *Proceedings of the International Conference on Design Automation*, 2008, pp. 574-579.

[89] R. Jordans, F. Siyoum, S. Stuijk, A. Kumar, H. Corporaal, "An Automated Flow to Map Throughput Constrained Applications to a MPSoC", Proceedings of the International Workshop on Bringing

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Theory to Practice, Predictability and Performance in Embedded Systems, 2011, pp. 47-58.

[90] V. Zaccaria, G. Palermo, G. Mariani, F. Castro, C. Silvano, "Multicube Explorer – A Design Space Exploration Framework for Chip Multiprocessors", *Proceedings of the International 2PARMA - Workshop on Parallel Programming and Run-time Management Techniques for Many-core Architectures*, 2010, pp. 325-331.

[91] R. Busseuil, L. Barthe, G.M Almeida, L. Ost, F. Bruguier, G. Sassatelli, P. Benoit, M. Robert, L. Torres, "OPEN – SCALE: A Scalable, Open Source NoC Based MPSoC for Design Space Exploration", *Proceedings of the IEEE International Conference on Reconfigurable Computing and FPGA's*, 2011, pp. 357-362.

[92] C. Horia, V. Lucian, "An Automatic Design Space Exploration Framework for Multi core Architecture Optimizations", *Proceedings of the IEEE International Conference on RoEduNet, Sibiu* , 2010, pp. 202-207.

[93] IndikaMeedeniya, AldeidaAleti, ImanAvazpour, Ayman Amin, "Robust ArcheOpterix: Architecture Optimization of Embedded Systems under Uncertainty", *Proceedings of the International Workshop on Software Engineering for Embedded Systems (SEES*), 2012, pp. 23-29.

[94] Carlos.A.Petry, Eduardo.W.Wachter, Guilherme.M.de Castilhos, Fernando.G.Moraes, Ney.L.V.Calazans, "A Spectrum of MPSoC Models for Design Space Exploration and its Use", *Proceedings of the International Symposium on Rapid System Prototyping* (RSP), 2012, pp. 30-35.

[95] Daniel Knorreck, LudovicApvrille, Renaud Pacalet, "Formal System-Level Design Space Exploration", *Concurrency and Computation: Practice and Experience. Vol. 25,* 2013, pp. 250-264.

[96] Michel.A.Kinsy, Srinivas Devadas, Michael Pellauer, "Heracles: A Tool for Fast RTL Based Design Space Exploration of Multi Core Processors", *Proceedings of the ACM International Symposium on Field Programmable Gate Arrays (FPGA),* 2013, pp. 125-134.

[97] Raphael Poss, Mike Lankamp, Qiang Yang, Jian Fu, Irfan Uddin, Chris.R. Jesshope, "MGSim: A Simulation Environment for Multi Core Research and Education", *Proceedings of the International Conference on Embedded Computer Systems: Architecture, Modeling and Simulation (SAMOS)*, 2013, pp. 80-87.