2011 IEEE Computer Society Annual Symposium on VLSI

A Design Space Exploration Methodology for Application Specific MPSoC Design

^{1}Amit Kumar Singh, ^{2}Akash Kumar and ^{1}Thambipillai Srikanthan
^{1} School of Computer Engineering, Nanyang Technological University, Singapore
^{2} Department of Electrical and Computer Engineering, National University of Singapore, Singapore
Email: ^{1}{amit0011, astsrikan}@ntu.edu.sg, ^{2}akash@nus.edu.sg

Abstract—**System designers need to perform design-space exploration (DSE) to find appropriate number and type of processing elements (PEs) to be present in a Multiprocessor Systems-on-Chip (MPSoC) to support a throughput-constrained application. This paper presents a DSE methodology that provides application to MPSoC architecture mappings, where different PEs get used. The methodology starts from a mapping using only one type of processors and evaluates different mappings by increasing heterogeneity to improve the performance.**

I. Introduction

Multiprocessor Systems-on-Chip (MPSoCs) contain multiple PEs that are connected through a communication network to fulfill the communication needs of the PEs. These MPSoCs may employ different type of PEs and their distinct features can be exploited to achieve high computation performance and energy efficiency.

Throughput (compute performance) of an application to be supported into an MPSoC depends upon how the MPSoC resources are used by the application components. There is a large number of possibilities for mapping the individual components to the MPSoC resources. Finding all of these possibilities manually is difficult and time consuming, so the system designers need to use automated DSE methodologies. This step to evaluate multiple mappings is important because it gives us an idea of the performance in advance before we actually go ahead to fabricate the system-on-chip (SoC).

Most of the existing DSE methodologies take an application and a fixed MPSoC platform as input, and perform DSE in view of some optimization criteria such as performance/power to find the best mapping [1], [2]. This mapping is then used to configure the MPSoC platform, provided it satisfies the user constraints. However, there are few works that provide multiple mappings for a fixed platform [3], [4]. Here, any mapping satisfying the user constraint can be used to design the MPSoC to support the application. The fixed platform may contain large number of PEs and some of them might not be used by the application, unnecessarily occupying the System-on-Chip resources. Further, exploration time gets increased with the number of PEs as the DSE methodologies have to look for more tasks to PEs mapping possibilities.

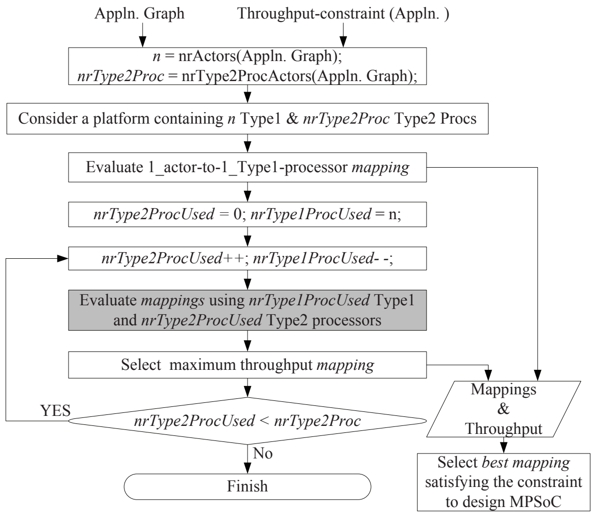
We propose an iterative DSE methodology that doesn't take a fixed MPSoC as input but starts from considering a *generic* MPSoC that contains PEs depending upon tasks and their implementation alternatives provided in the application. For example, the implementation alternatives could be a type1 and a type2 (more sophisticated) processor. The methodology first finds a mapping where each task is mapped on a separate one type of processor and then evaluates different mappings by moving tasks to more sophisticated processors to improve the performance. The exploration time in our methodology is controlled by limiting the number of PEs. The methodology selects the best mapping satisfying the throughput-constraint of the application to design a SoC on a Xilinx Field Programmable Gate Array (FPGA) by using the approach presented in [5]. We have targeted Vixtex-5 ML510 FPGA board [6] and have designed MPSoCs for multimedia applications H.263, JPEG and MP3 decoders after performing DSE for them.

II. Proposed Design Space Exploration Methodology for MPSoC Design

A. Iterative DSE Methodology

The iterative DSE flow to perform DSE for an application has been presented in Fig. 1, where the application graph (*Appl. Graph*) along with its throughput constraint is taken as input. The application is modeled as Synchronous Dataflow Graphs (SDFGs) [1]. The flow first finds the number of actors *n* and number of actors having their implementation alternative as type2 processors *nrType2Proc*, from the application graph. Next, a platform containing *n* type1 and *nrType2Proc* type2 IP-cores (processors, i.e., Procs) is considered and *1_actor-to-1_Type1-processor mapping* is evaluated and stored. Here, the number of type1 processors used (*nrType1ProcUsed*) and the number of type2 processors used (*nrType2ProcUsed*) are *n* and *0*, respectively. Next, we evaluate all the possible mappings for next type1 & type2 Proc combination that is obtained by decreasing and increasing *nrType1ProcUsed* and *nrType2ProcUsed*, respectively. This incorporates heterogeneity and we get improved throughput for the mappings. The maximum throughput mapping at this resource combination is selected and stored. The same process is repeated with the selected mapping to evaluate mappings at next type1 & type2 Proc combination. The process iterates by increasing the heterogeneity (decreasing *nrType1ProcUsed* and increasing *nrType2ProcUsed*) until *nrType2ProcUsed* reaches to its limit *nrType2Proc*.

Evaluating Different Type of Processors Combination Mappings. To evaluate all the possible type1 and type2 processors combination mappings, the best (maximum throughput) mapping using *nrType1ProcUsed* type1 and *nrType2ProcUsed* type2 processors (Procs) is taken as input and



Appln. Graph
Throughput-constraint (Appln.)
n = nrActors(Appln. Graph);
nrType2Proc = nrType2ProcActors(Appln. Graph);
Consider a platform containing n Type1 & nrType2Proc Type2 Procs
Evaluate 1_actor-to-1_Type1-processor mapping
nrType2ProcUsed = 0; nrType1ProcUsed = n;
nrType2ProcUsed++; nrType1ProcUsed- -;
Evaluate mappings using nrType1ProcUsed Type1 and nrType2ProcUsed Type2 processors
Select maximum throughput mapping
YES
nrType2ProcUsed < nrType2Proc
No
Finish
Mappings & Throughput
Select best mapping satisfying the constraint to design MPSoC

Fig. 1. Design Space Exploration Flow.

all the possible type1 and type2 processors combination mappings using *nrType1ProcUsed-* type1 and *nrType2ProcUsed+* type2 processors are evaluated by moving each actor with its implementation alternatives as type2 Procs from the type1 to a type2 Proc having no previous actor. The number of mappings is limited by the number of actors mapped on type1 Procs with their implementation alternatives as type2 Procs too in the input mapping.

The best mapping using maximum number of type1 processors (less area) satisfying the throughput constraint is used to design the MPSoC. The selected mapping uses minimum number of type2 processors (more area) towards designing a low area overhead MPSoC. The template of the MPSoC platform is based on a NoC-based architecture as used in [5].

III. Experiments

The proposed DSE methodology has been implemented by extending the publicly available tool set for SDF graphs used in [1]. We compared our iterative DSE methodology with an exhaustive methodology and the one presented in [3]. To evaluate the different methodologies for an application, the same generic platform graph is considered. Models of multimedia applications H.263 (4 actors), JPEG (6 actors) and MP3 (14 actors) decoders are considered for DSE and MPSoCs are designed for them on Xilinx Virtex-5 ML510 development board using EDK/ISE 12.1. Concerning to the type1 and type2 processors, we have used microblaze processors running at 100 MHz with disabled (low area, less sophisticated) and enabled (high area, more sophisticated) integer-multiplier/floating-point unit, respectively. The experiments have been performed on a Core 2 Duo at 3.16 GHz.

The exhaustive DSE has been performed to check if we miss any potential mapping by our iterative DSE. The exhaustive DSE considers all the possible one actor to one IP-core mappings such that each actor is mapped on a different IP-core and evaluates more number of mappings over the iterative DSE. A case study has been performed where the number of

TABLE I
H.263 decoder DSE at varying *nrType2Proc*

	Exhaustive DSE		Iterative DSE	
nrType2Proc	Number of Mappings	Evaluation Time (ms)	Number of Mappings	Evaluation Time (ms)
0	1	27.99	1	27.99
1	2	80.45	2	80.45
2	4	215.96	4	215.96
3	8	1652.21	7	1606.75
4	16	4832.56	11	2210.66

TABLE II
Applications evaluation on given homogeneous MPSoCs

	Evaluation Time (ms)	
	Approach of [3]	Iterative DSE
H.263 decoder (on 4 IP-cores)	9177.60	27.99
JPEG decoder (on 6 IP-cores)	14134.30	120.54
MP3 decoder (on 14 IP-cores)	355745.15	754.88

actors having their implementation alternative as type2 Procs, i.e., *nrType2Proc*, is varied. At each *nrType2Proc*, exhaustive and iterative DSE flow is executed by considering a platform containing *n* type1 and *nrType2Proc* type2 Procs, where, *n* is number of actors in the application. Table I shows DSE results for H.263 decoder at varying *nrType2Proc*. On an average, iterative DSE is faster by 39.17% over the exhaustive DSE. It has been observed that the best mappings at different resource combinations are the same when exhaustive and iterative DSE methodologies are employed.

Table II shows evaluation time to find the best mappings by our and the approach of [3]. It can be observed that evaluation time increases with the platform size (number of Procs), i.e., more mappings need to be evaluated when approach of [3] is employed. However, our methodology evaluates only one actor to one processor mapping and thus evaluation time is in control.

IV. Conclusion

Design space exploration (DSE) for MPSoCs is very important as it gives us an idea of the performance before we actually go ahead and design the hardware. This paper describes an iterative DSE methodology that provides multiple mappings and selects the best mapping satisfying throughput constraint of the application. The best mapping is used to design an MPSoC on a Xilinx FPGA.

References

[1] S. Stuijk et al., "Multiprocessor resource allocation for throughput-constrained synchronous dataflow graphs," in *Proceedings of the 44th annual Design Automation Conference*, 2007, pp. 777–782.

[2] A. Schranzhofer et al., "Dynamic power-aware mapping of applications onto heterogeneous mpsoc platforms," *Industrial Informatics, IEEE Transactions on*, vol. 6, no. 4, pp. 692 –707, 2010.

[3] S. Stuijk et al., "A predictable multiprocessor design flow for streaming applications with dynamic behaviour," in *Proceedings of the 13th Euromicro Conference on Digital System Design*, 2010, pp. 548–555.

[4] G. Mariani et al., "An industrial design space exploration framework for supporting run-time resource management on multi-core systems," in *Proceedings of DATE*, 2010, pp. 196–201.

[5] A. K. Singh et al., "Mapping real-life applications on run-time reconfigurable noc-based mpsoc on fpga," in *Proceedings of the International Conference on Field-programmable Technology*, 2010, pp. 365–368.

[6] "Xilinx," 2008, http://www.xilinx.com/.