Automatic Generation of Transaction Level Code for Fast SoC Design Space Exploration

Ganghee Lee, Yongjin Ahn, and Kiyoung Choi
Design Automation Lab., Seoul National Univ. (Email: berean97@poppy.snu.ac.kr)

Abstract

As billion transistors system-on-chip (SoC) design becomes a reality, the productivity gap between rapidly increasing design complexity and designer productivity lagging behind is becoming a more serious problem to be solved. To reduce the gap, we present a system that generates executable transaction level models automatically. It speed up the SoC design space exploration process at various abstraction levels.

1. Introduction

As billion transistors system-on-chip (SoC) design becomes a reality, the productivity gap between rapidly increasing design complexity and designer productivity lagging behind is becoming a more serious problem to be solved. To reduce the gap, the focus is being moved recently toward higher levels of abstraction [1]. So the designer starts with a very abstract mathematical model such as a process network or a finite state machine. Then the model is refined to lower levels of abstraction step by step before the final implementation is obtained [2]. Each refinement step typically requires generating an executable model for simulation to validate the refined design, which is a cumbersome and error-prone process if it is done manually.

In this paper, we present a system that generates such an executable model automatically from an intermediate mathematical model. It generates a SystemC model at the transaction level, since many efficient commercial/ non-commercial simulators are available for this language and level. The designers can use the proposed system for evaluating the refined design fast as well as exploring large design space efficiently.

The system generates transaction level models at two different levels of abstraction. One is timed functional model and the other is bus cycle accurate model. It is integrated into an MP-SoC design framework that we have developed.

2. Overview

Figure 1 shows our design flow for SoC design. It starts with a process network model written in SystemC. Then it automatically generates a hierarchical Synchronous Data Flow (SDF) model which is a combination of SDFs and FSMs. After that, we statically estimate the performance using Integer Linear Programming. The hierarchical SDF model, annotated with the static estimation results is used for application-to- architecture mapping. In this mapping stage, we give proper FIFO buffer size for communication. For fast system evaluation, we first generate a Timed Functional SystemC (TFSC) model. The simulation using this model is very fast but not accurate enough to evaluate dynamic behaviors such as bus conflict. After communication refinement, we generate C code for each ISS, which will be executed in Bus Cycle Accurate (BCA) simulation. In this paper, we focus on TFSC code and BCA C code generation, which are highlighted in Figure 1.

3. Transaction code generation

For TFSC, we generate two kinds of models. First one is called unscheduled TFSC model. The unscheduled model has no static schedule but works with data-driven scheduling supported by the SystemC kernel. Second one is called scheduled TFSC model. It contains a static schedule which is pre-determined in the mapping stage. The simulation of these two models can be used for comparing the static schedule strategy with the data-driven schedule strategy. Since the SystemC simulation terminates when it encounters with a deadlock, our model can also be used for checking deadlocks due to buffer empty or full. Finally, these evaluation results are fed-back to the mapping stage to find a better solution.

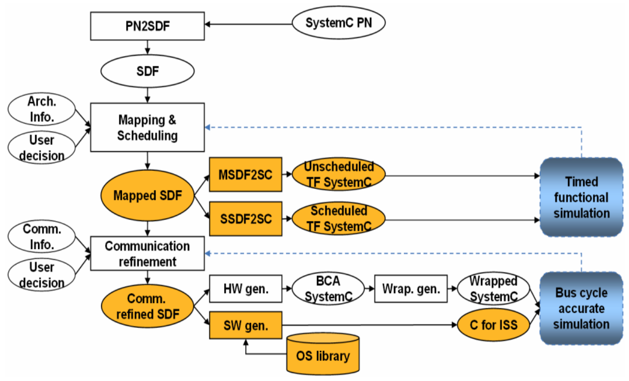

Figure 1. Proposed design flow.

The TFSC model is based on a simple model of point-to-point FIFO connection rather than actual bus system. In addition, the execution delay of each process is roughly estimated and annotated to itself before the simulation run. Therefore, the simulation is fast but has limited accuracy.

After refining communication, we generate a BCA model which is slower but more accurate than TFSC model. This model can be simulated with various commercial tools such as MaxSim. Our BCA model contains C code blocks that are compiled to be executed on one or more ISSs.

4. Experimental result

We tested the automatic code generation system with JPEG and H.264 examples. Table 1 shows the result.

Table 1. Generation and simulation runtime

(unit: second)

	JPEG	H.264 (2 frames)
TFSC generation	1	3
TFSC simulation	1	2
BCA (C) generation	1	1
BCA simulation	15	1881

We used a machine with P4 2GHz processor for the experiment. To measure the runtime, a simple platform consisting of two ARM9 processors and a shared memory connected with AHB was used. For BCA simulation, we used MaxSim, which is currently the fastest commercial tool for TLM simulation to the best of our knowledge. In spite of using MaxSim, it takes 15 and 1881 seconds for each example. But it's still tens to hundreds times faster than RTL simulation. The table shows that automatic code generation takes only a few seconds. Replacing the error-prone manual process that takes hundreds of hours, our automatic code generation approach is very useful for speeding up the SoC design space exploration process at various abstraction levels.

References

[1] K. Keutzer et al. "System-level design: Orthogonalization of concerns and platform-based design," In *IEEE Transactions on CAD of ICs and Systems*, Dec. 2000

[2] Abdi, S et al. "Automatic communication refinement for system level design" In *Proc of DAC*, 2003