KPASSA: A Tool for Simulating Periodically Scheduled SoCs

Julien Boucaron, Anthony Coadou, Robert de Simone

Abstract

Correct-by-construction systems-on-chips design requires formal models for accurate simulation, analysis and optimization. We present KPASSA, a tool built to handle and transform such models.

1. Introduction

Systems-on-chips design requires the development of correct-by-construction methodologies, in order to handle growing complexity. Different models of computation and communication (MoCCs) have been studied over the past decades; they give mathematical tools and methodologies to model, analyze and transform these applications. Correctness of an application/implementation relies, for instance, on proofs that the system has finite buffers and is deadlock-free. Optimizations of schedules and memory sizes of such systems can no longer be performed “by hand” by designers. A wide range of multimedia and signal processing applications falls in such MoCCs, where the application reveals, after a bounded initialization phase, a steady regime that allows to statically compute periodic schedules for computations blocks and communications. Such models amount to partial orders of execution, meaning that there is no over-specification of the description of an application. We develop a software named KPASSA that handles several such MoCCs, and can perform both analysis and optimizations.

2. From pure dataflow…

Models like Marked Graphs (MG) [6] or Synchronous Data Flow graphs (SDF) [9] can be used in this way to model “pure” data-flow systems, where a computation is abstracted by a node, and a communication link (or buffer) by an arc. It is possible then to compute maximum reachable throughput, corresponding schedules and buffers size for software and/or hardware implementations.

A latency-insensitive system [4] can be modeled as a 2-bounded marked graph [2]. Then, it can be statically scheduled, and its maximum throughput computed. The equalization process introduces latencies on non-critical cycles, while preserving its maximum throughput. This gives an insight to the designer where there are margins in the design, for re-design and further optimizations.

2. …To static controlled dataflow

The main limitation of above models is their lack on control: each node always produces and consumes the same amount of data. It is therefore impossible to express if-then-else routing patterns. It is not possible to reuse resources of computation, communication or buffers. In order to overcome this limitation, we previously introduced K-periodic Routed marked Graphs (KRGs): they can be seen as a CSDF-like [7] extension of MG. Two new kinds of nodes are introduced, inspired from BDF [3]: Merges and Selects. They respectively behave like multiplexers and demultiplexers. Routing patterns are assigned to each of these nodes by ultimately k-periodic binary sequences, borrowed from the n-synchronous theory [5]: these sequences are of the form $u.v^\omega$, where $u$ corresponds to the prefix (or initial part), and $v$ to a binary word infinitely repeated (or periodic part). They statically define whether a given token will be routed from/to the left or right input/output. This means that, routing pattern are not data dependent, and allows deterministic concurrency.

As a consequence, most of the analysis techniques from MG and SDF still apply to KRGs: deadlock-freeness or checking for bounded buffers are both decidable. Another useful property is that static schedules and routing patterns allow to define local transformations which preserve the global behavior, such as resources sharing (computation nodes and interconnects). An example of KRG is given in figure 1:

![Figure 1: Basic example of KRG.](image-url)
3. Tool features

- Full featured GUI.
- Supported MoCCs: MG, LID, SDF and KRGs.
- Bounded buffers and deadlock-freeness checking.
- Throughput and static schedules for both computation and communication.
- Buffer sizing.
- Modular simulator to ease integration with third-party code with C++/SystemC/Java.
- Throughput aware optimization for LID, MG.
- Correct-by-construction transformations on KRG routing nodes: sharing and/or duplicating resources.

4. Further works

We plan to link KPASSA and TimeSquare, another software developed at INRIA Sophia-Antipolis, in order to combine both dataflow graphs and time analysis. TimeSquare is dedicated to the modeling and analysis of timed systems using logical clocks (e.g., the irregular execution cycle time of a processor can be a clock). TimeSquare supports an implementation of the Time Model defined as part of the official OMG UML MARTE profile. Clock relations and constraints between components and parts of the systems (IPs) are set using the Clock Constraint Specification Language. Modeling in this case relies on a multiform model of time including multiple clocks either chronometric (classical time modeling) or logical. We plan to extend our tool in several ways. We plan to introduce parameterized models, drawn from meta-modeling paradigms such as HPDF [8] in order to capture a larger class of applications. Another extension would be to handle the composition of different models into a single design.

5. References


Acknowledgement

This work was supported in part by the CIMPACA (Centre Intégré de Microélectronique Provence-Alpes-Côte d’Azur) French Regional Initiative, by STMicroelectronics at Grenoble under grant NANO2012, and a grant by Texas Instrument at Villeneuve-Loubet.