Optimized Compilation of Executable UML/SysML Diagrams for the Design of Data-Flow Applications

Andrea Enrici, Julien Lallet
Nokia Bell Labs, Centre de Villarceaux, 91620 Nozay, France
Email: \{firstname.lastname\}@nokia.com

Renaud Pacalet, Ludovic Apvrille
LTCI, CNRS, Télécom ParisTech
Université Paris-Saclay, 75013 Paris, France
Email: \{firstname.lastname\}@telecom-paristech.fr

Future 5G radio access networks (RANs) are expected to provide higher (10x) data-rates and to support use cases such as the Internet of Things and cloud computing. To meet the computational requirements of these use cases, RAN servers will accelerate the execution of data-intensive computations (e.g., signal-processing applications) onto reconfigurable components (e.g., FPGAs). This change in the servers’ architecture raises the need for novel tools capable to rapidly automate the programming of these mixed hardware/software systems in a world that has been traditionally dominated by IT software.

At DATE 2018 University Booth, we demonstrate our latest improvements in the compilation of system-level models for a 5G data-link layer receiver, as defined in release 15 by the 3GPP consortium [1], for the single antenna case, in the uplink (SC-FDMA), eNodeB side. In our design flow, Fig. 1, models that specify different views (functionality, architecture and communication protocols) are created in TTool/DIPLODOCUS [2], a UML/SysML framework for the rapid prototyping of data-flow systems.

Multiple design solutions are then explored (Design Space Exploration, DSE, with simulation and formal verification in Fig. 1) and those satisfying the desired requirements (throughput and latency) are compiled into control C code. Our Optimizing Model Compiler (OMC, Fig. 1) automatically generates control code that includes a Synchronous-Data Flow (SDF) scheduler and a memory manager for the data processing (e.g., FFT) and transfer (e.g., DMAs) operations for the system under design and a target platform. The platform-specific code that provides the implementation of processing and transfer operations is, instead, linked by OMC against external libraries. OMC is integrated to the software architecture of TTool/DIPLODOCUS as a plugin.

The novelty of this year’s demonstration resides in the memory footprint analysis and optimizations that are performed by OMC [5]. The latter transforms the input UML/SysML specifications into a multi-rate SDF graph. This graph is transformed into a single-rate Directed Acyclic Graph where the production and consumption rates of each SDF actor are equal and FIFOs with initial tokens are ignored. At this point, a Memory Exclusion Graph (MEG) is created, where each node represents a FIFO of the initial SDF graph and edges link FIFOs that cannot be allocated to the same physical memory space. Based on these intermediate representations, the compiler’s back-end generates an optimized memory allocation, where the FIFO buffers between processing and transfer operations share physical memory. This is opposed to pure (un-optimized) translation-based code generation approaches, where physical FIFO buffers are separately allocated for each data dependency of an application. For this demonstration’s case study, OMC allows to reduce the overall memory footprint of the output C code up to 80%, for sequential schedulings.

In this demonstration, our target platform is composed of a general-purpose control processor that offloads some of the 5G receiver’s computations operations (e.g., LDPC decoder) to a IP-block that serves as an accelerator. The setup of the demonstration is composed of a laptop connected to a Xilinx FPGA prototyping board where the target platform is instantiated. On the laptop we installed the complete model-based design chain (TTool/DIPLODOCUS, OMC). The transmitter chain and the communication channel that are paired with the receiver under design are emulated in MATLAB.

**References**


