DVCon Europe 2022 Tutorial Proposal

Title: Verification of Inferencing Algorithm Accelerators

Tutorial Type: Industrial

Organizer:

Mathilde Karsenti
Marketing Programs Manager
Siemens EDA, U.S.A
+1-503-685-7000
Mathilde.Karsenti@siemens.com

Speakers:

Russell Klein
Program Director
Siemens EDA, U.S.A.
+1-503-685-7000
Russell.Klein@siemens.com

Petri Solanti
Applications Engineer
Siemens EDA, Germany
+49-162-2166 914
Petri.Solanti@siemens.com

Introduction:

Inferencing is becoming ubiquitous, from consumer electronics to industrial automation, many systems will have inferencing as part of their core functionality. The computational demands for inferencing are often beyond the capabilities of embedded processors, especially when considering power and performance requirements. This means hardware acceleration of some kind will need to be included in the system. Some systems can use general purpose AI accelerators, available as discrete components or IP. But bespoke accelerators allow developers to achieve optimal performance and efficiency. Creating a custom accelerator for processing inferences introduces some significant design and verification challenges, this tutorial will focus on the latter. Specifically, it is not sufficient to verify just the low-level implementation, but the higher-level function, the inferencing, needs to be proven as well.

Summary:

This tutorial will show how to verify a machine learning algorithm from Python code in a machine learning framework, like TensorFlow or Caffe, to RTL. As the algorithm is successively refined, at each stage it is necessary to show that both the refinement goals have been achieved, and that the implementation has not deviated from the prior design stage. The algorithm will be migrated from Python, to C++, and finally to RTL using High-Level Synthesis (HLS).

A keyword spotting inferencing algorithm will be used to Illustrate these concepts. Starting with a Keras implementation in Python, the algorithm will be optimized for neural network topology, and pruning and initial quantization can be performed. Next an equivalent C++ implementation is created, suitable for HLS. This is verified against the original Python. At the C++ level, parallelism, pipelining, and data caching and buffering are added to the algorithm. As these refinements are introduced, the resulting implementations need to be verified against the Python, or the algorithmic C++ code. Finally, the C++ will be synthesized to RTL thought High-Level Synthesis. Using both formal and dynamic simulation techniques the equivalency of the C++ and the resulting RTL will be shown. The verification goals and techniques will be described for each stage of the design flow.

Power analysis will be performed on the design. The power, performance, and area (PPA) for several design architectural alternatives will be presented and compared against metrics for software implementations on a RISC-V core. Power can be estimated during HLS, and more accurately measured after RTL synthesis and post place and route, for greater accuracy.

Source code for the keyword spotting algorithm and related code will be made available to participants after the tutorial.

Intended Audience: Hardware and verification engineers and managers intending to design and verify inferencing accelerators.