SECDA Design Suite: Efficient HW-SW Co-Design of DNN Accelerators for Edge Inference

Jude Haris, José Cano
School of Computing Science, University of Glasgow, Scotland, UK

***Abstract*—The progress in the current Artificial Intelligence (AI) landscape has led to state-of-the-art performance in many emerging tasks, such as object detection, audio-to-text translation, and image generation. To enable AI on edge devices (Edge AI), specialized hardware accelerators are developed to achieve the best performance per watt while dealing with the limited hardware resources of a given edge device. Unfortunately, designing, deploying and evaluating new specialized accelerators requires a lot of time and hardware-software expertise.**

We propose the SECDA Design Suite, a set of specialized tools that enable an efficient design process for developing AI accelerators for edge inference, utilizing the SECDA design methodology. It provides a fully integrated environment that allows developers to design, deploy and evaluate new hardware architectures using FPGAs. The SECDA Design Suite is *open-source* and available at: https://github.com/gicLAB/SECDA-Design-Suite.

I. Introduction

As the need for Artificial Intelligence (AI) accelerators grows, it has become evident that processes and tools to develop them, especially for edge devices, are limited. Hardware-Software (HW-SW) co-design methodologies such as SECDA [1] (*SystemC Enabled Co-design of DNN Accelerators*) provide a guideline for developers and researchers to design new hardware architectures utilizing FPGAs to prototype, deploy, and evaluate new inference accelerators.

In this short paper we introduce the SECDA Design Suite, an *open-source* set of specialized tools that employs the SECDA methodology to: i) enable hardware-software co-design of new specialized accelerators; and ii) explore new hardware architectural ideas with ease. The SECDA Design Suite allows researchers and hardware accelerator developers to design, deploy, and evaluate new accelerator architectures for Deep Neural Networks (DNNs), including large language models (LLMs) and Convolution Neural Networks (CNNs).

The following sections provide an overview of the SECDA methodology and the five key components of the SECDA Design Suite: SECDA-Core, SECDA-TFLite, SECDA-LLM, SECDA-Sandbox, and SECDA-Benchmarking.

II. SECDA Methodology

The SECDA methodology [1] is the fundamental cornerstone of the SECDA Design Suite. SECDA is a hardware-software co-design methodology to efficiently produce optimized DNN inference accelerators for edge devices using FPGAs. Figure 1 provides an overview of the SECDA methodology. SECDA uses SystemC [2] as an accelerator simulation framework, which allows candidate designs to be iterated upon efficiently. Using SystemC High Level Synthesis (HLS), we can produce a synthesizable design from the same accelerator definition ('SystemC Accelerator'). The co-design of a software 'Accelerator Driver' and a hardware accelerator is achieved by integrating SystemC's simulation features within the target 'Application Framework', i.e. Machine Learning (ML) frameworks such as TensorFlow Lite or *llama.cpp* [3], allowing designers to quickly test potential optimizations such as varying data transfer and tiling strategies. Embedding the simulation environment and the hardware accelerator into the same software environment reduces the costs of exploring hardware-software co-design trade-offs via simulation, relative to synthesizing the design on an FPGA with every change.

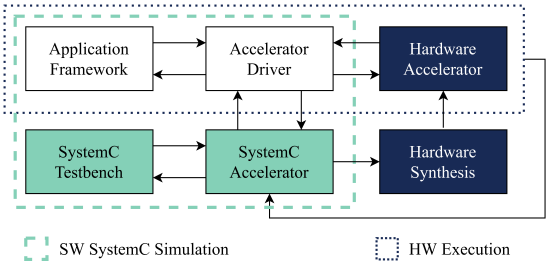

Fig. 1: Overview of SECDA Methodology [1].

The SECDA methodology relies on two different iterative design loops to explore the accelerator design space (Figure 1). The most frequently used design loop is iterations in fast SystemC Simulation (SW SystemC Simulation), with the second loop being benchmarking on FPGAs (HW Execution). Hardware benchmarking requires logic synthesis which can be very time consuming, thus SECDA minimizes the number of times this occurs through simulation iterations.

III. SECDA Design Suite

While the SECDA methodology provides a guideline on how to efficiently and iteratively design new specialized hardware accelerators, there is a need for additional tooling to enable quick and easy instantiation of SECDA such that the target application frameworks and workloads can be used without requiring additional developer overhead. The SECDA Design Suite provides an overall hardware-software co-design environment that enables developers to use the SECDA methodology supporting multiple ML frameworks and custom workloads while utilizing same SECDA-Core across the different SECDA instantiations. Figure 2 provides a high-level overview of the SECDA Design Suite and how its components are connected.

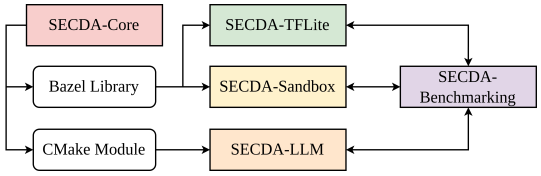

Fig. 2: Overview of the SECDA Design Suite.

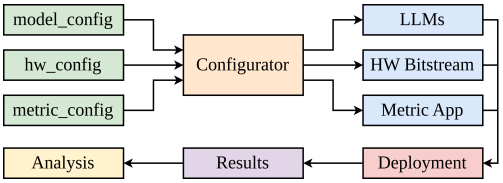

Fig. 4: Overview of SECDA-Benchmarking tool.

A. SECDA-Core

Figure 3 provides an overview of SECDA-Core, the main hardware-software library that enables all the key features of the SECDA Design Suite across the various SECDA-enabled tools/platforms. It consists of five main components: i) AXI-API: used to define communication channels and data transfers between the accelerator and the host CPU; ii) SystemC HW Modules: a set of hardware modules used for simulation and ease of development of new accelerators (e.g., DMA Engine module); iii) ML Utilities: a set of helper functions to enable pre/post-processing of ML operations (e.g., calculating data size of tensors); iv) Multi-threading Support: a C++ class that enables the creation of CPU threads and tasks that can interact with custom accelerators independently. v) Hardware-Software Profiler: a set of utilities interlinked with the previous four components to track and profile key software and hardware metrics; additionally it provides users the ability to create custom profiling.

SECDA-Core is written in C++ and is managed as both a Bazel Library and a CMake module, enabling easy integration with ML frameworks that utilize these two build tools.

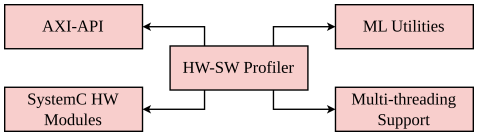

Fig. 3: Overview of the SECDA-Core library.

B. SECDA Instantiations

We have three instantiations of the SECDA methodology: SECDA-TFLite, SECDA-LLM and SECDA-Sandbox.

1) SECDA-TFLite: is a TensorFlow Lite (TFLite) [4] specific toolkit [5] that provides the initial development environment to instantiate the SECDA methodology within TFLite. The integration is through the TFLite delegate system, which supports custom backends for DNN inference.

2) SECDA-LLM: is a *llama.cpp* [3] specific platform [6] that enables the research, design, and deployment of LLM accelerators for edge devices using FPGAs. Within SECDA-LLM, we create a custom GGML [3] backend to enable the integration of the SECDA-Core tools within *llama.cpp*. This custom backend allows you to design and deploy hardware accelerators for any of the LLMs that *llama.cpp* supports.

3) SECDA-Sandbox: is a testing and development environment that enables quick and easy testing of full hardware accelerators or specific hardware components without requiring a full-scale ML framework. It allows the user to create simple C/C++ programs that quickly test and verify the correctness of various workload characteristics of DNNs with new hardware designs and their respective software drivers. Users can export these hardware-software designs to SECDA-LLM or SECDA-TFLite to perform full-scale end-to-end evaluation on pre-trained DNN models.

C. SECDA-Benchmarking

The SECDA-Benchmarking tool automates the process of running benchmarks on the target FPGA-based device and collecting the results. All SECDA instantiations are connected to the benchmarking tool. Figure 4 shows the high-level structure of the benchmarking tool. It is designed to be extensible, allowing developers to add DNNs and accelerators with minimal effort. Additionally, the tool provides a versioning system to track the performance of the accelerators across different versions of the accelerator design. The configurations (green boxes) for benchmarks consist of the DNNs to execute, the metrics to measure (e.g., power, latency, throughput), and the accelerators to use. Once the experiment is configured, the developer can invoke the benchmarking tool to run the experiment on the target FPGA board, collect the target metric data and visualize the results.

IV. Conclusion

We summarized and provided insight about the SECDA Design Suite which is based on the SECDA design methodology. We provided an overview of the different components of the design suite and their purpose and capabilities. Finally, we add that the SECDA Design Suite is an on-going open-source project that we plan to extend for additional purposes such as processing-in-memory (PIM) based accelerators.

Acknowledgments

This work was partially supported by the UK Engineering and Physical Sciences Research Council (grant EP/R513222/1), the EU Project dAIEDGE (GA Nr 101120726) and the Innovate UK Horizon Europe Guarantee (GA Nr 10090788).

References

[1] J. Haris *et al.*, "SECDA: Efficient Hardware/Software Co-Design of FPGA-based DNN Accelerators for Edge Inference," in *SBAC-PAD*, 2021.
[2] IEEE, "IEEE Standard for Standard SystemC Language Reference Manual," *IEEE Std 1666-2011*, 2012.
[3] "llama.cpp," https://github.com/ggerganov/llama.cpp, 2024.
[4] Google, "TensorFlow Lite," https://www.tensorflow.org/lite/.
[5] J. Haris *et al.*, "SECDA-TFLite: A toolkit for efficient development of FPGA-based DNN accelerators for edge inference," *JDPC*, 2023.
[6] J. Haris *et al.*, "Designing Efficient LLM Accelerators for Edge Devices," 2024. [Online]. Available: https://arxiv.org/abs/2408.00462