Special Issue on Compiling for Accelerators

Guido Araujo and Lucas Wanner, *University of Campinas (UNICAMP), Campinas, 13083-852, Brazil*

In the November/December 2021 issue, *IEEE Micro* celebrated the 50th anniversary of the microprocessor. Since Intel 4004 was launched in 1971, a half century of great technological advancements in semiconductors, microarchitecture, and software have consolidated a device that is the bedrock of all computing systems of our era. The tremendous impact of microprocessors in all sectors of human society cannot be overstated.

Moore's law and Dennard scaling were central to microprocessor development, as each new semiconductor manufacturing process node provided designers with smaller, more abundant, faster, and more energy-efficient transistors. The number of transistors and clock frequencies in microprocessors therefore saw continuous growth for many decades. But this history is not free of its own challenges and hurdles. By the early 2000s, manufacturers reached the limits of power density for traditional semiconductor technology and frequencies stagnated.

From that point on, it became evident that designing multicores was the way to move forward while still retaining the low-cost gains of a consolidated semiconductor industrial base. Multicores became a key element to continue improving computing performance, but again a new issue showed up on the horizon. With time, it became clear that programming and debugging multithreaded applications are challenging tasks that can also result in poor core utilization and power inefficient systems. As Moore's law continued to slow down, the inefficiencies inherent to general-purpose processors became more and more evident for many applications.

Hence, by early 2010, industry and academia realized that designing accelerators for the most relevant workloads was the way to push performance forward while still controlling power consumption. Since then, hardware accelerators have rapidly been considered a relevant architectural feature. Central processing unit (CPU) instruction set architecture (ISA) extensions, custom-designed engines, and field-programmable gate array (FPGA)-based systems have been proposed as acceleration architectures to improve program execution in scientific, machine learning, database, and other application domains.

ALTHOUGH MUCH EFFORT HAS BEEN DEVOTED TO THE DESIGN OF ACCELERATORS, THERE IS STILL A LARGE KNOWLEDGE GAP ON HOW TO EFFECTIVELY USE, GENERATE, AND COMPILE CODE FOR SUCH ARCHITECTURES.

Although much effort has been devoted to the design of accelerators, there is still a large knowledge gap on how to effectively use, generate, and compile code for such architectures. This special issue of *IEEE Micro* brings a set of articles that help to bridge this gap.

In this issue, articles are divided into two groups according to how the accelerator is coupled with the microprocessor. The first set of articles describes novel compilation techniques targeting accelerators that are CPU ISA extensions, such as vectorization and matrix multiplication. In the second half, advanced compilation techniques that use dedicated and FPGA-based accelerators are discussed.

Hardware acceleration is gradually becoming a key design technique for modern computing systems. This special issue offers the readers of *IEEE Micro* an overview of the compiling techniques for this new era of computer architecture.

COMPILING FOR CPU ISA-EXTENDED ACCELERATORS

› "TinyIREE: An ML Execution Environment for Embedded Systems From Compilation to Deployment." To enable the efficient usage of specialized instructions, compilers need to

0272-1732 © 2022 IEEE
Digital Object Identifier 10.1109/MM.2022.3193765
Date of current version 14 September 2022.

have the flexibility to tailor code generation to the underlying accelerator's hardware. Multi-level intermediate representation (MLIR) is a novel compiler intermediate representation (IR) that seeks to achieve that. In the first article of this section, the authors present a new execution environment for machine learning models on embedded systems, based on MLIR.

- "Retargetable Optimizing Compilers for Quantum Accelerators via a Multilevel Intermediate Representation." Quantum architectures have been gaining traction as a powerful acceleration engine to solve large-scale scientific problems. This article focuses on integrating the compilation of quantum applications into a code generation flow based on MLIR.

THIS SPECIAL ISSUE OFFERS THE READERS OF IEEE MICRO *AN OVERVIEW OF THE COMPILING TECHNIQUES FOR THIS NEW ERA OF COMPUTER ARCHITECTURE.*

- "Compiling for the IBM Matrix Engine for Enterprise Workloads." The recent increase in the demand for machine learning and scientific applications has put matrix multiplication at the center stage of computing. This article gives an in-depth discussion of the challenges of compiling for IBM's matrix multiplication outer-product acceleration extension.
- "Performance Left on the Table: An Evaluation of Compiler Autovectorization for RISC-V." Although vectorization has been intensively studied as an acceleration engine, there are still performance improvements that can be achieved using it. This article presents a characterization of the performance gap between hand-vectorized code and autovectorized code, identifying the key limitation of modern vectorization approaches, and showing how to compensate for that when using a RISC-V microprocessor.
- "Compiling for Vector Extensions With Stream-Based Specialization." In this article, the authors give a contribution toward automating vectorization for novel streaming-based architectures. These architectures provide different limitations and cost tradeoffs that are not traditionally captured by the single instruction, multiple data-focused vectorization machinery in existing compilers.

COMPILING FOR DEDICATED AND FPGA-BASED ACCELERATORS

- "Unifying Spatial Accelerator Compilation With Idiomatic and Modular Transformations." This work presents an approach to specialize reconfigurable spatial accelerator architectures to certain application domains. This article details novel compilation techniques that improve the performance of code running on spatial accelerators.
- "Synthesizing Legacy String Code for FPGAs Using Bounded Automata Learning." This article gives a new perspective on how to automatically optimize the performance of high-level synthesis (HLS) for code that is not written with hardware acceleration in mind. It uses a query-based learning algorithm to extract deterministic finite automata from existing source code, so the function can be executed in the generated kernel without requiring changes or annotations in the legacy code.
- "Bridging Python to Silicon: The SODA Toolchain." Tools for designing accelerators are important components of any design ecosystem. This article describes how MLIR transformations can be used to extract key dataflow patterns from Python source code to synthesize acceleration hardware.
- "Yin-Yang: Programming Abstractions for Cross-Domain Multi-Acceleration." Combining a collection of accelerators specialized in distinct workloads to solve a large-scale problem has become an emerging area of research. This article describes a compilation toolchain that seeks to address this problem.
- "SpecHLS: Speculative Accelerator Design Using High-Level Synthesis." SpecHLS is a promising technique to improve the throughput of HLS implementations. This article proposes an approach to speed up HLS accelerators, based on making the common execution case fast while injecting pipeline bubbles for the slow cases.
- "HPVM: Hardware-Agnostic Programming for Heterogeneous Parallel Systems." Portability of source and object code is a challenging issue for

heterogeneous platforms. HPVM introduces a hardware-agnostic IR that allows it to generate code for multiple targets including GPUs, FPGAs, function accelerators, and deep learning accelerators.

GUIDO ARAUJO is a full professor of computer science and engineering at the University of Campinas, Campinas, 13083-852, Brazil. His research interests include compiling technology, parallel programming, and heterogeneous computing, explored in close cooperation with Petrobras, IBM, Intel, Xilinx, LGE, and Samsung. Araujo received a Ph.D. degree in electrical engineering from Princeton University, Princeton, NJ, USA. Contact him at guido@ic.unicamp.br.

LUCAS WANNER is a professor with the Computer Systems Department, University of Campinas, Campinas, 13083-852, Brazil. His research focuses on energy efficiency in hardware and software. Wanner received a Ph.D. degree from the University of California Los Angeles, Los Angeles, CA, USA. Contact him at wanner@unicamp.br.