Recent Achievements of the Open-Source CVA6 Core: FPGA Optimizations, Coprocessor Acceleration, Yocto Linux Support

Sébastien Jacq[1], Jean-Roch Coulon[2], Kevin Eyssartier[1], Jérôme Quévremont[1]

[1]Thales Research & Technology, Palaiseau, France
[2]Thales Secure Silicon, Meyreuil, France

Summary

In this extended abstract, recent contributions to the CORE-V CVA6 open-source RISC-V application core are presented: (1) the FPGA optimizations that reduce by 55% the resources while increasing performance by 40%; (2) the CV-X-IF coprocessor interface to extend and speed up the supported instruction set and (3) the availability of the Yocto embedded Linux distribution for 32- and 64-bit versions of the core. These contributions are available from open-source repositories and are brought to the first cooperative projects implementing the roadmap for European sovereignty on open-source hardware, software and RISC-V.

Introduction

CORE-V CVA6 is an open-source RISC-V application core and the related project running at the OpenHW Group. The core exists in two versions that share the same SystemVerilog source code [1]: the CV32A6 (32-bit) and the CV64A6 (64-bit). The CV64A6 originates from ARIANE, developed by the PULP team [2]. The CV32A6 is a later enhancement, designed by Thales.

Besides the choice between a 32- and a 64-bit core, the CVA6 is highly configurable: e.g. the MMU, the PMP, the FPU, the performance counters and the compressed instructions are optional. The L1 cache can be customized according to multiple parameters such as the storage method (write-back/write-through), the size and the number of ways.

One year ago, the CVA6 project was introduced at the RISC-V Spring Week 2022 [3][4]. This year, the focus is on some results recently achieved on the CVA6: resource and frequency optimizations for FPGA targets, integration of a coprocessor interface and support of the popular Yocto embedded Linux distribution generator.

Optimizations for FPGA Targets

Today, each FPGA manufacturer offers a proprietary soft processor (Xilinx's Microblaze, Intel's Nios-II…). When changing FPGA supplier, it is necessary to overhaul the complete design: new proprietary soft core, new SoC architecture, complete re-compilation of the software applications… Changing the FPGA supplier creates the risk of reduced performance. Moreover, this generates additional development costs related to the use of different tool chains.

With the CV32A6, our challenge is to offer a competitive and FPGA technology-agnostic soft core. For this, it was necessary to optimize the microarchitecture of the CV32A6 for FPGA targets, as it was originally developed for ASICs.

The first set of optimizations have reduced the amount of logic resources used (look-up tables and flip-flops) through several CV32A6 microarchitecture modules, such as the RV32C decoder, the branch prediction unit, the instruction and data caches, the register file, the performance counters, the scoreboard, or the memory management unit (MMU).

For instance, the MMU has gained a second level of translation lookaside buffers (TLBs) implemented in SRAM to reduce the size of the first level of TLBs implemented with flip-flops. This allowed to increase the number of TLBs while reducing the resource usage.

The second set of optimizations has increased the processor frequency by breaking critical timing paths.

Table 1 presents the key performance indicators obtained on a Xilinx FPGA.

Table 1: Key performance indicators of CVA6, including MMU and L1 caches on Xilinx Kintex 7 FPGA (XC7K325T-2)

	Original CV32A6	Optimized version	*Evolution*
Look-up tables	18,103	8,077	*−55%*
Flip-flops	11,484	4,403	*−61%*
DSP blocks	4	4	-
Block RAM	36	12	*−67%*
Max. freq.	100 MHz	140 MHz	*+40%*
CoreMark/MHz	2.8	2.8	-
CoreMark	280	392	*+40%*

Thanks to these optimizations, CV32A6 is a credible alternative to technology-specific cores for ASIC and FPGA developments, especially for teams relying on several technology providers.

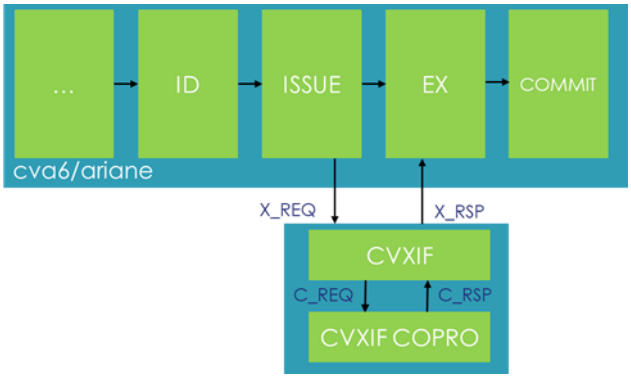
Domain-Specific Acceleration

There is today no standard bus interface to connect coprocessors to a RISC-V processor. This is why some OpenHW members have specified the CV-X-IF extension interface [6]. CV-X-IF enables the extension of the instruction set supported by a RISC-V processor without any change in the RTL source code of the processor (cf. figure 1). When the processor decodes an instruction that it

cannot execute, the instruction is offloaded to the coprocessor. Such instruction can be a custom one or can belong to any RISC-V extension, including compressed instructions. The coprocessor can also submit memory requests to the processor through CV-X-IF. In CVA6, CV-X-IF is the only implementation that handles speculation to increase performance. CV-X-IF is an opportunity to accelerate access to coprocessors, to implement domain-specific instructions without CPU impact, and to promote the interoperability of CPU cores and coprocessors.

As of today, the instruction offloading is implemented in CVA6. The memory access and compressed instructions through CV-X-IF will be implemented in future projects. An OpenHW Group project is currently validating the specification by implementing it on several OpenHW CORE-V processor cores including CVA6.

Figure 1: CV-X-IF coprocessor interface with CVA6 pipeline



Embedded Linux Support

Yocto is a popular generator of Linux distribution for embedded systems. It allows access to a wide catalog of ports of popular applications and frameworks, and handles the whole embedded complexity with a packaged SDK and easy deployment.

Thanks to our recent contribution, both CV32A6 and CV64A6 now support a Yocto-based state of the art RISC-V software tooling.

The OpenSBI firmware implements RISC-V SBI (Supervisor Binary Interface) which allows the execution of privileged operations by the bootloader and the supervisor mode. This firmware will ease future development e.g. on multi-core or performance counters support for Linux. U-Boot bootloader now supports CVA6 and allows to boot from an SDCard or through TFTP booting. Linux 5.10.7 kernel support has been added and is packaged in a Yocto layer [5]. On-board porting has been achieved through different kernel and embedded debugging techniques such as source offsetting and log_buf dumping.

A debug demonstration project is also available. It performs a hands-on debug of both bare-metal and Linux software through the Eclipse integrated development environment. Command line interface is also available.

This work leverages the RISC-V open-source ecosystem and allows contributors and users to quickly run a Linux distribution on CVA6.

Conclusion

In this abstract, recent CVA6 advances were presented that are readily available as open-source from the OpenHW Group's repositories. CVA6 is an ongoing project and more is to come in a near future. Among the team's plans are further performance optimizations, improvements in the core's documentation and the industrial-grade verification of some 32- and 64-bit configurations of the core, targeting the quality needed for production of integrated circuits.

More open-source contributions from SMEs and the industry are desired, to deliver more, quicker and better for the community and have a positive impact on systems-on-chips and FPGAs embedding CVA6.

The growth of the RISC-V open-source ecosystem is key to European technology sovereignty as exposed in [7]. As a building block of the two first projects that implement this roadmap (TRISTAN and upcoming ISOLDE), CVA6 should bring a competitive advantage to the European industry.

References

[1] https://github.com/openhwgroup/cva6/

[2] Zaruba, F., Benini, L. "The cost of application-class processing: Energy and performance analysis of a Linux-ready 1.7-GHz 64-bit RISC-V core in 22-nm FDSOI technology." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 27.11 (2019): 2629-2640.

[3] Quévremont, J. "An Open-Source Application Core: CVA6 from the OpenHW Group" Spring 2022 RISC-V Week. 2022.

[4] Coulon, J.-R. "Verification of the CVA6 Open Source Core", Spring 2022 RISC-V Week. 2022.

[5] https://github.com/openhwgroup/meta-cva6-yocto

[6] https://github.com/openhwgroup/core-v-xif

[7] "Recommendations and Roadmap for European Sovereignty in Open Source Hardware, Software and RISC-V Technologies", August 2022, https://digital-strategy.ec.europa.eu/en/library/recommendations-and-roadmap-european-sovereignty-open-source-hardware-software-and-risc-v

Acknowledgements

These activities are supported by the FRACTAL and TRISTAN projects, which have received funding from the Key Digital Technologies Joint Undertaking (KDT JU) under grant agreements 877056 and 101095947. The JU receives support from the European Union's Horizon Europe research and innovation program. The present action reflects only the authors' view; the European Commission and the JU are not responsible for any use that may be made of the information it contains.