TÉCNICO LISBOA

Linux capable RISC-V CPU for IOb-SoC

Pedro Nuno de Melo Antunes

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor(s): Prof. José João Henriques Teixeira de Sousa

Examination Committee

Supervisor: Prof. José João Henriques Teixeira de Sousa
Member of the Committee: Prof. Horácio Cláudio De Campos Neto

September 19, 2022

Abstract

Blablabla

Honor Declaration

“Declaro por minha honra de que ”

Local, Date Signature

Contents

List of Figures

List of Tables

Code Listing

Acronyms

CLINT Core-local Interrupt Controller.

CPU Central Processing Unit.

CSR Control and status register.

ISA Instruction set architecture.

M Machine.

OoO Out-of-Order.

OS Operating system.

PLIC Platform-Level Interrupt Controller.

S Supervisor.

SoC System on a chip.

U User.

1 Introduction

1.1 Motivation

1.2 Objectives

2 Must Have Concepts

During this chapter, we're going to discuss topics that help understand the technology developments made along the thesis project.

2.1 The IOb-SoC Template

2.1.1 Internal Buses

2.1.2 How peripherals work

2.2 Open Source Verification tools

For testing purposes, it is important to have a good hardware simulation environment. For that, we take advantage of already existing and well-developed tools. There exist a number of simulation tools, most of them are proprietary, as for example *'xcelium'* from *'Candence'*. Its utilization can increase the cost of a project significantly. In this Thesis we will make an effort of using open-source, free to use, verification tools. In specific, we will take advantage of *'Icarus Verilog'* and *'Verilator'*. Although both tools are for verification, they serve different purposes, due to their characteristics.

- ***'Icarus Verilog'*** is a Logic Simulator that uses verilog or system-verilog testbenchs to test the UUT (Unit Under Test). Unfortunately, its support for system-verilog is limited and some designs might not run in this simulator. *'Icarus Verilog'* is also known as *'IVerilog'*.

  After compiling the hardware design, *'IVerilog'* outputs a file which can be run line by line to simulate designed logic.

- ***'Verilator'***

The biggest differences are: *'Verilator'* only represents logic signal as 1's or 0's, contrary to *'IVerilog'* which also represents unknown values as X's; Since *'Verilator'* ends up being a C++ program it is much faster to run the simulation than with *'IVerilog'*; On another perspective *'Verilator'* is slower than *'IVerilog'* to interpret the hardware logic design. As such, it is easier to use *'IVerilog'* to detect errors on the design, but it is better to use *'Verilator'* for more complexed simulations.

2.3 RISC-V architecture

The RISC-V **CLINT** is described The RISC-V **PLIC** was first described in the privilege instructions documentation, but since version 1.10 it was moved to its own document.

2.4 What is the UART16550?

2.5 The Linux Boot Flow

2.5.1 Bootloader firmware

2.5.2 What is a device tree?

2.6 Simulation

https://www.sifive.com/blog/risc-v-qemu-part-1-privileged-isa-hifive1-virtio

3 Existing Technologies

There already exists embedded microcontrollers capable of running Linux. Big companies as for example ARM, Qualcomm, MediaTek, Intel and AMD have created microcontroller capable of running Linux. But the processor architecture of those microcontrollers is not open-source, much less the microcontroller itself.

As an example, the *Raspberry Pi 4* is a very capable and cheap board where a developer can test and implement new software running in Linux. The Raspberry CPU is an *Cortex-A72* [1] witch is a System on Chip (SoC) developed by ARM on their ARMv8 64-bit CPU architecture. But if someone wanted to use the Raspberry as a base for his costume hardware design, that would be impossible. And thus appears the need for open-source hardware that allows creating something new without having to start from scratch every time. This led to the appearance of RISC-V the open-source CPU architecture.

3.1 Closed source RISC-V Embedded Systems

Since then, a few companies using RISC-V have appeared. RISC-V CPUs are already present in the automotive and IoT markets, besides AI chips in data centers. Due to the RISC-V ISA royalty free license new StartUps tend to look at RISC-V CPUs as a solution for their cores. Even if the CPU Core isn't free to use it ends up being a cheaper solution.

While creating new products companies proved how advantageous the RISC-V architecture was. Furthermore, they have contributed to open-source software, hardware and documentation. Some companies with a big recognitions involved with RISC-V technology are:

- *Western Digital* who now uses RISC-V in its external storage disks.
- *Microchip* as launched the first RISC-V-Based System-on-Chip (SoC) FPGA, *PolarFire*.
- *Antmicro/Microsemi* [1] have built a software called Renode that is used to develop, debug and test multi-node RISC-V device systems.
- *BeagleBoard.org*, *Seeed Studio* and *StarFive* worked together to build the first affordable RISC-V computer designed to run Linux, *BeagleV* [3]. The board is priced around 150€.

These companies have all helped pave the way for a full-feature Operating System based on the Linux kernel to be compatible with the RISC-V architecture. However, there are two companies that have a bigger impact on RISC-V CPU design, those are Andes Technology and SiFive.

[1] Microchip has acquired Microsemi Corporation in May 2018.

3.1.1 Andes Technology

Andes Technology is one of the founder members of the RISC-V International. Since it is highly involved with RISC-V it ended up being one of the major contributor (and maintainer) of the RISC-V tool-chain. This is important because the RISC-V ISA is merely an instruction set architecture, there needs to exist complementing software, such as compiler and development tools.

Nowadays Andes CPU's are applied nearly everywhere, from telecommunications, storage controllers, touch screen sensors to data centers, etc. Andes Technologies has had incredible success using RISC-V technology, as prove they have shipped billions of embedded SoC with RISC-V processors based on their RISC-V ISA variant, AndeStar™ V5.

Andes CPUs witch are capable of running Linux are the *A25* [16] and *AX25* [17]. Both support single and double precision floating point, the RISC-V P-extension (draft) DSP/SIMD ISA and an MMU (Memory Management Unit) for Linux applications. Besides that both enable the use of Machine (M), User (U) and Supervisor (S) Privilege levels that allow running Linux and other advanced operating systems with protection between kernel and user programs. Furthermore, both have L1 instruction and data cache. The difference between them is that *A25* is based on 32-bit architecture and the *AX25* is 64-bit. This leads to the *AX25* being ideal for embedded applications that need to access address space over 4GB, and the *A25* being smaller in gate count. Both CPUs can be implemented on the AE350 [15] SoC allowing to use these CPUs on developer boards, for example in the *ADP-XC7K160/410* [14].

3.1.2 SiFive

SiFive is a company that was born from the RISC-V ISA. SiFive was founded by three researchers from the University of California Berkeley, Krste Asanović, Yunsup Lee, and Andrew Waterman. Those researchers were deeply involved with the development of the RISC-V ISA, from working on the base ISA to working on the floating point numbers and compressed instructions ISA extensions. It is no surprise that the first company to release a chip and development board that implemented the RISC-V ISA was SiFive. This happened in 2016 one year after the company was founded.

In 2017 SiFive launched *U54* [8] witch was the first RISC-V CPU capable of running a full fledge Operating System like Linux. With it they launched the *U54-MC* [9] SoC that had four *U54* 64-bit cores. Furthermore, the *U54-MC* implemented the initial CLINT and PLIC unit. The development of the CLINT and the PLIC made by SiFive would eventually lead to the documentation and specification of the respective hardware components with witch RISC-V systems must be compliant if they proclaim to use either one. One year after, in 2018, they launched *HiFive Unleashed* [6] witch was the first board that implemented the *U54* CPU and

run a Linux OS with a desktop environment (DE). The *HiFive Unleashed* has been discontinued and better hardware has been made available.

SiFive has since then extended their *U Cores* product lineup. All *U* cores are 64-bit application processors capable of running Linux. The highest performance core is the *U74* [10]. The core architecture is RV64GBC witch means it supports the RISC-V I, M, A, F, D, B and C ISA extensions (explained in ***ref section 2.3.x***). This CPU has already been applied to multiple boards, for example, the *BeagleV* has a SoC with dual core SiFive *U74* CPU. SiFive as also launched its own development board, *HiFive Unmatched* [7], with four *U74* cores on the *U74-MC* [11] SoC. Furthermore, in 2021, Canonical the developers behind Ubuntu have announced the OS support for both the HiFive Unmatched and HiFive Unleashed.

3.2 Open-Source Solutions

Built upon the RISC-V open-source Instruction set architecture, various CPU designs have emerged. Some of them are fully open-source and might be implemented in other projects. Those CPUs were mostly developed by Universities research groups or by individuals with a grant.

RISC-V CPUs are most popular in embedded systems and IoT devices. Consequently there exists a widely variety of open-source CPUs witch are implemented on multiple embedded microcontrollers. A few examples of those CPUs would be the *PicoRV32* [18], *NEORV32* [13], *DarkRISCV* [4] and *Ibex* [5] from lowRISC. But those will not be discussed in detail on this paper since they do not meet the requirements to run the Linux Kernel. This CPUs either only support Machine (M) level privilege mode or support Machine (M)+Supervisor (S) mode. Moreover, none of the given examples support the Atomic RISC-V ISA extension. This extension is essential to run Linux. Since the kernel explicitly executes instructions from the Atomic extension.

To run a Linux based Operating System an application processor is needed. This means that the processor should have the required Control and status register (CSR), support M+S+U privilege modes and support atomic instructions. An open-source solution would be either the *CVA6* [19] (previously known as Ariane), *BOOM* [20] or *VexRiscv* [12] from SpinalHDL.

3.2.1 *CVA6*

The CVA6 is a 6-stage, single issue, in-order CPU which can execute either the 32-bit or 64-bit RISC-V instruction set. CVA6 has support for the I, M, A and C RISC-V ISA extensions. The original design was initiated in a research group by a Phd student at ETH Zurich (where they called the core Ariane). Since then the development and maintenance of CVA6 was incorporated in the *OpenHW Group* as part of their CORE-V processor lineup. The support

for RV32IMAC was developed more recently by Thales, and was also made open-source. The CPU design is illustrated in figure 3.1 that was obtained from: `https://github.com/openhwgroup/cva6/`.

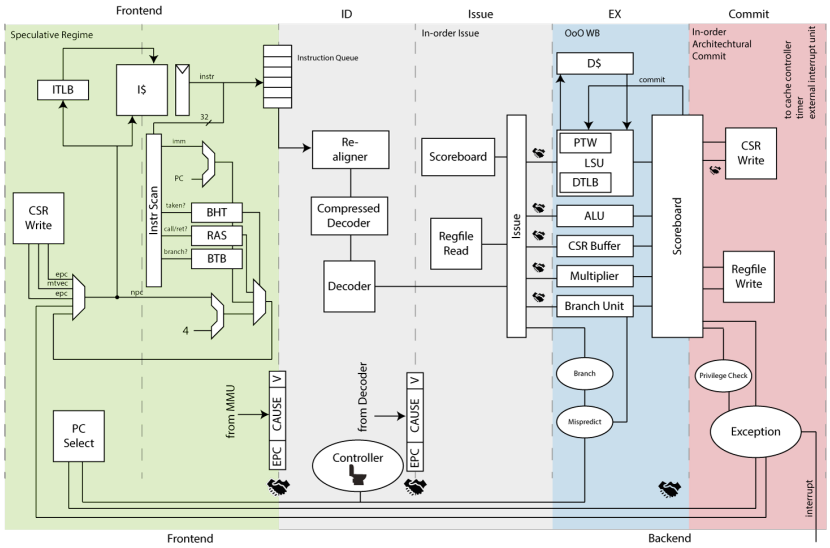

Figure 3.1: CVA6 core design architecture

The CVA6 supports any operating system based on Unix since it implements the three needed privilege levels M, S and U. The core is written in SystemVerilog, and its micro-architecture is designed to reduce the critical path length. Furthermore, since it is written in SystemVerilog, it is easier for someone knowledgeable in the classic Verilog and VHDL languages to understand and create a customized CPU based on the CVA6 if needed. However, although the CVA6 is an open-source project it is hard to take advantage of isolated hardware components. This is due to how it was developed. Every SystemVerilog module of the CVA6 depends on other files from the same project and the CPU is very little customizable. To illustrate the problem if we wanted to remove the L1 cache present in the CVA6 in order to use the L1 cache used on IOb-SoC, it would be very difficult and time consuming to create a CPU core without that component.

The CVA6 can be found implemented on *OpenPiton* [2]. *OpenPiton* is an open-source project developed by the Princeton Parallel Group. With it one can easily create a SoC that has multiple CV6 cores and run a full-feature Operating system (OS) on a development board with an FPGA.

3.2.2 The Berkeley Out-of-Order RISC-V Processor

The Berkeley Out-of-Order RISC-V Processor (*BOOM* [20]) is a superscalar Out-of-Order (OoO) processor executing the RV64GC variant of the RISC-V ISA. It is written on an unconventional Chisel Hardware Description Language (HDL). BOOM is optimized to run on ASICs, although it is also usable on FPGAs. Its priority is to be a high performance, synthesizable, and

parametrizable core for architecture research. The current release, named "SonicBOOM", is the fastest core in Instructions per Cycle (IPC) from the publicly available open-source RISC-V cores.

BOOM is a 10-stage CPU with the following stages: Fetch, Decode, Register Rename, Dispatch, Issue, Register Read, Execute, Memory, Writeback, and Commit. However, in most practical implementations, many of those stages are merged, generating seven stages altogether: Fetch, Decode/Rename, Rename/Dispatch, Issue/Register Read, Execute, Memory and Writeback. Since committing happens asynchronously, it is not counted as part of the “pipeline”. Although this CPU has good performance, it has a limited customization options. The load-store unit is optimized for the superscalar out-of-order architecture, and the data cache is organized into two dual-ported banks. At the front end, it is possible to customize the banked L1I cache, the TLB, and the decode stage. It is difficult or impossible to remove the cache from the core design and use the IOb-Cache instead.

rocket-chip from CHIPS Alliance

3.2.3 SpinalHDL

The *VexRiscv* [12] CPU is a 32-bit Linux Capable RISC-V CPU written in the Spinal HDL. The hardware description of this CPU is accomplished by utilizing a software-oriented approach. VexRiscv is an in-order CPU with five pipeline stages. Many CPU plugins are optional, which add many functionalities to build a custom RISC-V CPU. The architecture design approach in this processor is entirely unconventional, but it has its benefits: there are remarkably few fixed hardware components; Parts of the CPU can be swapped, turned on and turned off via the plugin system; without modifying any of the CPU sources, it is possible to add new functionalities/instructions easily; It permits the CPU arrangement to cover a significantly large spectrum of implementations, allowing the construction of an entirely parametrized CPU design. When the CPU is configured without plugins, it only includes the description of the five pipeline stages and their basic functionalities and nothing else. Everything else needs to be added to the CPU via plugins, including the program counter.

VexRiscv and NaxRiscv Talk about litex-vexriscv!

3.3 Overall CPU comparison

	ARM	Andes Technology		SiFive		PULP platform	UC Berkeley	SpinalHDL	
	Cortex-A72	A25	AX25	U54	U74	CVA6	BOOM	VexRiscv	NaxRiscv
Architecture bit widths	64-bit	32-bit	64-bit	64-bit	64-bit	32/64-bit	64-bit	32-bit	64-bit
MMU	Y	Y	Y	Y	Y	Y	Y	Y	Y
FPU	Y	Y	Y	Y	Y	X	Y	X	Y
16-bit instructions	X	Y	Y	Y	Y	Y	Y	Y	X
Cache L1(I+D)	Y	Y	Y	Y	Y	Y	Y	Y	Y
Interrupt Controller	X	Y	Y	Y	Y	X	X	X	X
S+U+M Mode	N/A	Y	Y	Y	Y	Y	Y	Y	Y
GNU/Linux	Y	X	Y	Y	Y	Y	Y	X	Y
Open-Source	X	X	X	X	X	Y	Y	Y	Y

Table 3.1: CPU comparison table

4 Hardware Developed

4.1 Simulation Unit Under Test Wrapper

4.2 VexRiscv Wrapper

4.3 UART16550 Wrapper

4.4 CLINT Unit

4.5 PLIC Unit Wrapper

5 Software Developed

5.1 Python Console

5.2 Verilator Testbench

5.3 Barebones Interrupt Routine

5.4 IOb-SoC Linux Stage 0 Bootloader

5.5 IOb-SoC on OpenSBI

5.6 IOb-SoC Device Tree

5.7 IOb-SoC Linux *'rootfs'*

5.8 Makefiles

6 Products of the expedition, AKA Project Results

6.1 FPGA Resources Consumption

6.2 Run/Boot Linux Performance

7 Contributed Repositories

- **iob-soc**
- **iob-soc-vexriscv**
- **iob-lib**
- **iob-vexriscv**
- **iob-uart16550**
- **iob-clint**
- **iob-plic**

8 Conclusions

8.1 Achievements

8.2 Future Work

A Annex

A.1 Annex 1

Annex 1

A.2 Annex 2

Annex 2

Bibliography

[1] Arm Holdings (arm). *The Cortex-A72 processor specifications*. <https://developer.arm.com/Processors/Cortex-A72>.

[2] Jonathan Balkind et al. “OpenPiton: An Open Source Manycore Research Framework”. In: *Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems*. ASPLOS ’16. Atlanta, Georgia, USA: ACM, 2016, pp. 217–232. ISBN: 978-1-4503-4091-5. DOI: 10.1145/2872362.2872414. <http://doi.acm.org/10.1145/2872362.2872414>.

[3] Seeed Studio BeagleBoard.org and StarFive. *BeagleV, The First Affordable RISC-V Computer Designed to Run Linux*. <https://beagleboard.org/static/beagleV/beagleV.html>.

[4] darklife. *DarkRISCV*. <https://github.com/darklife/darkriscv>.

[5] lowRISC. *Ibex RISC-V Core*. <https://github.com/lowRISC/ibex>.

[6] SiFive. *HiFive Unleashed*. <https://www.sifive.com/boards/hifive-unleashed>.

[7] SiFive. *HiFive Unmatched*. <https://www.sifive.com/boards/hifive-unmatched>.

[8] SiFive. *U54*. <https://www.sifive.com/cores/u54>.

[9] SiFive. *U54-MC*. <https://www.sifive.com/cores/u54-mc>.

[10] SiFive. *U74*. <https://www.sifive.com/cores/u74>.

[11] SiFive. *U74-MC*. <https://www.sifive.com/cores/u74-mc>.

[12] SpinalHDL. *VexRiscv*. <https://github.com/SpinalHDL/VexRiscv>.

[13] stnolting. *The NEORV32 RISC-V Processor*. <https://github.com/stnolting/neorv32>.

[14] Andes Technology. *ADP-XC7K160/410, FPGA Based Development Platform*. <http://www.andestech.com/en/products-solutions/andeshape-platforms/adp-xc7k160-410/>.

[15] Andes Technology. *AE350 Platform*. <http://www.andestech.com/en/products-solutions/andeshape-platforms/ae350-axi-based-platform-pre-integrated-with-n25f-nx25f-a25-ax25/>.

[16] Andes Technology. *AndesCore™ A25, Compact High-Speed 32-bit CPU for Real-time and Linux Applications*. <http://www.andestech.com/tw/%E7%94%A2%E5%93%81%E8%88%87%E8%A7%A3%E6%B1%BA%E6%96%B9%E6%A1%88/andescore-processors/riscv-a25/>.

[17] Andes Technology. *AndesCore™ AX25, Compact High-Speed 64-bit CPU for Real-time and Linux Applications*. <http://www.andestech.com/tw/%E7%94%A2%E5%93%81%E8%88%87%E8%A7%A3%E6%B1%BA%E6%96%B9%E6%A1%88/andescore-processors/riscv-ax25/>.

[18] YosysHQ. *PicoRV32 - A Size-Optimized RISC-V CPU*. 2015. <https://github.com/YosysHQ/picorv32>.

[19] F. Zaruba and L. Benini. “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”. In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 27.11 (Oct. 2019), pp. 2629–2640. ISSN: 1557-9999. DOI: 10.1109/TVLSI.2019.2926114.

[20] Jerry Zhao et al. “SonicBOOM: The 3rd Generation Berkeley Out-of-Order Machine”. In: (May 2020).