

Article

RisCO2: SoC Implementation and Performance Evaluation of RISC-V Processors for Low-Power CO₂ Concentration Sensing

Ricardo Núñez-Prieto ^{2,3,*} , David Castells-Rufas ^{2,3}  and Lluís Terés-Terés ^{1,2} 

¹ Institut de Microelectrònica de Barcelona, IMB-CNM (CSIC), Spain

² Microelectronics & Electronic Systems Department, Universitat Autònoma de Barcelona (UAB), Spain

³ Barcelona Supercomputing Center (BSC), Spain

* Correspondence: ricardo.nunez@bsc.es; Tel.: +34-93-4137716

Abstract: In the field of embedded systems, energy efficiency is a critical requirement, particularly for battery-powered devices. RISC-V processors have gained popularity due to their flexibility and open-source nature, making them an attractive choice for embedded applications. However, not all RISC-V processors are equally energy-efficient, and it is important to evaluate their performance in specific use cases. This paper evaluates the energy consumption and resource utilization of a new RISC-V processor, RisCO2, and four existing processors - Zero-riscy, Micro-riscy, Ri5cy, and CV32E40P - in a signal demodulation application for NDIR CO₂ sensors. The processors were implemented in the PULPino SoC and synthesized using Vivado IDE. The processor named RisCO2 is based on the RV32E_Zfinx instruction set and was designed from scratch by the authors specifically for low-power signal processing applications such as signal demodulation in CO₂ NDIR sensors. The other processors are Ri5cy, Micro-riscy, and Zero-riscy, developed by the PULP Platform team, and CV32E40P (derived from Ri5cy) from the OpenHW Group, all of them widely used in the RISC-V community. Our experiments showed that RisCO2 had the lowest energy consumption among the five processors, with a 53.5% reduction in energy consumption compared to CV32E40P and a 94.8% reduction compared to Micro-riscy. Additionally, RisCO2 had the lowest FPGA resource utilization compared to the best-performing processors, CV32E40P and Ri5cy, with a 46.1% and a 59% reduction in LUTs, respectively. Our findings suggest that RisCO2 is a highly energy-efficient RISC-V processor for NDIR CO₂ sensors that require signal demodulation. The results also highlight the importance of evaluating processors in specific use cases to identify the most energy-efficient option. This paper provides valuable insights for designers of energy-efficient embedded systems using RISC-V processors.

Keywords: RISC-V; PULPino; NDIR CO₂ sensors; FPGA; energy efficiency; signal demodulation; power consumption

1. Introduction

The increasing demand for low-power and high-performance processors for embedded systems has led to the development of many architectures and implementations in recent years. In particular, the RISC-V architecture has emerged as a promising candidate for such systems due to its modular and customizable architecture. The RISC-V ISA provides an open-source and royalty-free platform that offers great flexibility in the design of embedded systems.

One of the most critical factors in the design of embedded systems is power consumption, which is becoming increasingly important due to the rapid growth of battery-powered electronic devices and the emergence of the Internet of Things (IoT) applications. Therefore, reducing the energy consumption of processors is a key requirement in the design of these systems.

This paper focuses on the design and implementation of energy-efficient RISC-V processors for use in signal processing applications, specifically in non-dispersive infrared

(NDIR) CO₂ sensors [1], [2]. NDIR CO₂ sensors are widely used in various industrial and environmental monitoring applications, and their energy-efficient operation is a critical factor in their adoption.

We present a comparative study of five RISC-V processor designs in terms of energy consumption and FPGA resource utilization: RisCO2, Ri5cy, Zero-riscy, Micro-riscy, and CV32E40P. RisCO2 is an in-order, RV32E_Zfinx implementation optimized for energy consumption, designed from scratch specifically for use in NDIR CO₂ sensors that require signal demodulation. The Ri5cy, Zero-riscy, and Micro-riscy processors are reference implementations of RISC-V processors: Ri5cy [3] is a 4-stage, single-issue, in-order processor that features an RV32IMC[F] implementation that optionally can provide full support for RV32F single-precision floating-point instructions. Zero-riscy and Micro-riscy [4] are 2-stage, single-issue, in-order processors with an RV32IMC and RV32EC implementation, respectively, designed to be a simplified version of Ri5cy to demonstrate how small a RISC-V CPU core could actually be. And CV32E40P [5] is a 4-stage, in-order, 32-bit RISC-V core derived from Ri5cy that supports the RV32IMC[F][Zfinx] instruction set.

We used the PULPino SoC platform [6] for implementing the five processors and simulating their energy consumption. We used Vivado 2020.2 IDE from Xilinx to synthesize, implement, and generate switching activity files to improve the accuracy of the power simulations.

The results show that our design, RisCO2, is the most energy-efficient processor, consuming only 0.29 mJ of energy, which is 53.5% and 63.2% less energy than CV32E40P and Ri5cy, respectively. RisCO2 also has the lowest resource utilization compared with CV32E40P, using only 4,889 LUTs, 2,354 FFs, and 2 DSPs, which is 46.1%, 7.8% and 71.4% less than CV32E40P, respectively. These findings demonstrate the effectiveness of RisCO2 in reducing energy consumption and FPGA resource utilization, making it a suitable option for battery-powered electronic devices and IoT applications.

In conclusion, the results of this comparative study demonstrate the effectiveness of custom-designed RISC-V processors in achieving energy-efficient signal processing in NDIR CO₂ sensors. The findings show that customizing RISC-V processors can achieve significant reductions in energy consumption and resource utilization compared to reference implementations and provides insights that can guide the selection and optimization of processors for use in energy-constrained embedded systems.

2. Previous work

The research on low-power RISC-V processors for embedded systems in the field of the IoT is essential because of the increasing demand for devices with extended battery life and reduced energy consumption. With the growth of IoT, there is a need for embedded processors that can perform tasks efficiently with minimal power consumption. A low-power RISC-V processor can help achieve this by reducing the device's power consumption while still providing sufficient processing power for the intended application. This can lead to longer battery life, lower costs, and more sustainable devices, which are essential in the IoT ecosystem. Additionally, the open-source nature of RISC-V makes it an attractive platform for researchers and developers who want to build low-power and energy-efficient embedded systems.

In that regard, there are numerous publicly available RISC-V processors with open-source RTL that cater to low-power requirements. Some of these processors are designed with a general-purpose approach, making them suitable for broad applications. Examples include VexRiscv and SweRV. On the other hand, specific applications, such as IoT, have dedicated processors like Ri5cy, Zero-riscy, and Rocket, which are optimized for the unique demands of IoT devices.

VexRiscv [7] is a 32-bit configurable RISC-V soft processor written in SpinalHDL and developed by C. Papon in 2019. It is designed for FPGA implementation and intended for use in embedded systems and supports various configurations for custom instruction sets and peripheral interfaces. Due to its low power consumption and high performance, it is

particularly well suited for FPGA-based embedded applications, such as real-time control and data processing. It has a pipeline with a configurable number of stages, from 2 to 5 stages, and provides support for the RV32I[M][F][C] instruction set. SweRV [8] is a 32-bit, 9-stage, dual-issue, superscalar, mostly in-order pipeline with some out-of-order execution capability that supports the RV32IMC_Zicsr_Zifence ISA. The SweRV processor is intended for a wide range of applications, including storage devices, embedded systems, and data centers.

As mentioned in the introduction, the Ri5cy and Zero-riscy processors are two open-source low-power RISC-V processors designed specifically for embedded systems and IoT applications, both written in SystemVerilog by the PULP Platform team from ETH Zurich. Ri5cy is a 32-bit, 4-stage, in-order processor with a small and efficient microarchitecture. Ri5cy aims to provide a balance between performance and power efficiency, making it suitable for resource-constrained embedded systems. It supports the RV32IMC[F] instruction set, which includes the integer, multiplication, and compressed instruction set extensions. The Zero-riscy processor is designed for ultra-low-power applications where power consumption is critical, such as IoT devices and wearable electronics. Zero-riscy features a 2-stage, in-order pipeline with a small footprint and optimized power efficiency. It supports the RV32IMC instruction set and includes various power-saving techniques, such as clock gating and dynamic voltage and frequency scaling (DVFS).

Rocket [9] is an open-source RISC-V processor written in Chisel and developed at the University of California, Berkeley. A 5-stage, in-order scalar core that implements the RV32G and RV64G ISA, it has a highly configurable and extensible design that serves as a platform for research and development in education and industry projects. Its open-source nature and flexible design make it a popular choice for exploring new ideas in computer architecture, prototyping novel processor features and building custom processor designs tailored to specific applications or domains.

In line with the IoT application-specific processors mentioned above, we presented in a previous conference paper [10] the design of a single-issue, in-order, 32-bit microprocessor utilizing the RISC-V ISA architecture intended for energy-efficient signal processing in wireless sensor nodes with a specific focus on non-dispersive infrared (NDIR) CO₂ sensors. The RISC-V microprocessor built from scratch is employed to demodulate the CO₂ sensor signal and compute the concentration levels.

By combining various design parameters, we evaluated the performance of three processor variants to support different instruction sets, namely RV32I, RV32IM, and RV32IMF. Our results indicate that including a floating-point unit (FPU) in the processor enhances energy efficiency in this context at the cost of an increase in hardware utilization. Specifically, adding a floating-point unit to an already optimized RV32IM design variant led to a decrease in the system's energy consumption by more than a factor of two, although with an equivalent increase in the FPGA resource utilization of the design.

To mitigate the area increase without affecting performance, we proposed a list of improvements to be made in future work that led to the final version of the processor design presented in this work, which we have named *RisCO2* to reflect its intended use in low-power embedded systems for CO₂ concentration measurements. The list of implemented improvements and their impact on the processor performance is provided in the sections below.

3. Proposed architecture

We made several improvements to our previous RISC-V RV32IMF processor, resulting in the development of *RisCO2*. The following is a list of the actions that were implemented in successive design steps:

1. support for the "E" extension, which halves the number of integer registers from 32 to 16 registers.
2. support for the "Zfinx" extension, which eliminates the need for a separate floating-point register file and enables sharing of the integer register file for both floating-point

and integer data. Overall, as a result of these optimizations, the register file is reduced by a factor of 1/4 compared to an RV32IMF implementation.

Once the E_Zfinx ISA was implemented, the application was compiled, and the executable `.elf` file was analyzed using a Python script [11]. This script searched for the number of occurrences of each ISA opcode within the compiled program. The analysis revealed that several instructions were unused, including multiplication, division, shifts, comparisons, and instructions for reading and writing in the control and status registers. Consequently, the decoder was modified in several design steps to eliminate the logic associated with those unused instructions.

3. remove the logic associated with integer multiplication and division `mul`, `mulh[u|su]`, `div[u]`, and `rem[u]`.
4. remove the logic associated with the management of control and status registers `csrrw[i]`, `csrrs[i]`, and `csrrc[i]`, as well as the unused instructions related to shifts and comparisons `sra`, `slti`, and `slt`.
5. remove the logic that supports misaligned memory accesses, resulting in a simplified load-store unit that assumes all memory addresses generated by the compiler are aligned to a 4-byte boundary.

The ultimate version of RisCO2 resulted in a 5-stage, single-issue, in-order processor based on the RV32E_Zfinx instruction set, with a specific focus on energy efficiency. This processor is intended for use in NDIR CO₂ sensors that necessitate signal demodulation to infer the gas concentration. The improvements made in the design of RisCO2 resulted in a reduction in energy consumption when compared to the original design, and the results are presented in Section 4. The simplified block diagram of the core is shown in Figure 1.

The RisCO2 pipeline consists of five stages through which instructions pass during execution. The stages are Fetch, Decode, Execute, Memory, and Writeback. In the Fetch stage (IF), the instruction is fetched from memory and placed in an instruction register.

In the Decode stage (ID), the instruction is decoded in the Control Unit, and the operands are identified and read from the General Purpose Register file (GPR). A Hazard Unit detects and resolves hazards that can occur due to dependencies between instructions. It inserts pipeline bubbles or forwards data from one stage to another to ensure that instructions are executed in the correct order and without errors.

In the Execute stage (EX), the ALU performs basic integer addition/subtraction arithmetic and logic operations, and the Floating-point Unit (FPU) performs addition, subtraction, multiplication, division, square-root, and fused multiply-add operations on floating-point numbers. The FPU is an open-source parameterized IP named *FPnew* [12] developed by the Digital Circuits and Systems Group at the ETHZ (PULP Platform). The FPU handles single-precision (32-bit) and adheres to the IEEE 754 standard for floating-point arithmetic. The unit exhibits varying latency based on the type of operation, often spanning multiple cycles. To ensure correct program execution, the unit incorporates an output to stall both the program counter (PC) and the pipeline.

In the Memory stage (MEM), data is read from or written to memory through the Load Store Unit (LSU). The Control and Status Register Unit (CSR) contains only two registers, namely `mcycle` and `minstret`, which are utilized for performance measurements.

In the Writeback stage (WB), the operation results are written back to the registers. The Commit Unit verifies when an instruction has reached the last stage of the pipeline, and its output is used to increment the `minstret` counter in the CSR.

3.1. Test methodology

The processor was implemented on a Nexys4 board from Xilinx (FPGA P/N XC7A100T-1CSG324C, 28 nm node) using the Vivado 2020.2 IDE, and the power simulations were performed using a switching activity file generated after post-implementation simulation to improve the accuracy. The processor was initially implemented as a stand-alone device

in the FPGA, with direct connection to a scratchpad memory built from BRAM blocks configured as true-dual port memory. This scratchpad memory stores both the program and data.

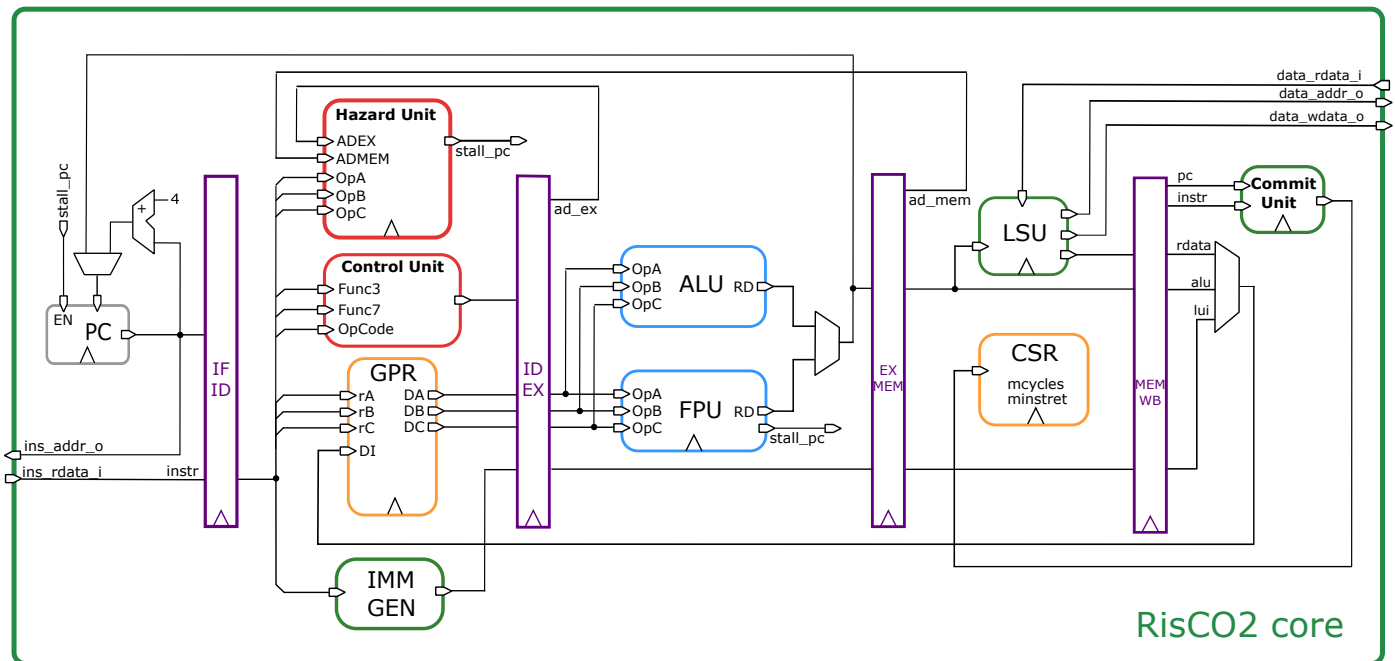


Figure 1. Simplified block diagram of the RisCO2 core architecture showing its five pipeline stages and all functional blocks.

Later, we integrated the processor into the PULPino SoC [6], a reference platform developed by the PULP Platform team, which is an open-source single-core RISC-V SoC built for the Ri5cy and Zero-riscy cores. The PULPino SoC has separate single-port data and instruction RAMs and includes a boot ROM with a boot loader capable of loading a program via SPI from an external flash device. The primary objective of integrating RisCO2 into the PULPino SoC was to compare its performance in terms of resource utilization and power consumption with other RISC-V processors. To achieve this goal, we integrated Zero-riscy, Micro-riscy, Ri5cy, and CV32E40P processors into separate projects on the PULPino SoC. Zero-riscy and Micro-riscy are small and efficient processors commonly used in low-power embedded systems, whereas the Ri5cy and CV32E40P processors are more complex cores commonly used in more powerful embedded systems.

There are several important considerations concerning the implementation of the PULPino SoC platform. Firstly, the two 32 kB single-port data and instruction RAMs were merged into a single 64 kB BRAM true dual-port memory utilizing a unified address space for both data and instruction program. Secondly, the boot ROM was eliminated from the design. Figure 2 depicts a block diagram of the PULPino SoC platform that has been customized for our testing purposes.

In the test program for our performance comparison, we utilized a for-loop-based algorithm that demodulates the digital data from the CO₂ sensor and computes its concentration in ppm. Notably, we did not use a physical gas sensor; instead, we opted to generate the data using a Python script. The script synthetically modulates the signal from the sensor and samples it at a rate of 16.38 KHz. Once the data was generated, we preloaded the samples in the data memory to be used in our study. Additionally, to enhance the compiled program's performance in Zero-riscy and Micro-riscy, we utilized a C++ template class written by Schregle [13] to emulate fixed-point support since these processors lack a hardware floating-point unit. As demonstrated in [10], this approach improved the program's execution performance.

As mentioned, we used Vivado 2020.2 IDE and a switching activity file obtained after post-implementation timing simulation to obtain power simulation results. Vivado was chosen for its ability to provide detailed power estimates for different regions of the FPGA fabric and for its comprehensive power analysis features. For synthesis, we utilized the default strategy provided by Vivado, and for implementation, we opted for the Performance_ExplorePostRoutePhysOpt strategy, which enables the physical optimization step and incorporates various algorithms for optimization, placement and routing to potentially enhance the outcomes. With a clock frequency setting of 25 MHz, the different designs always met the timing constraints.

The switching activity file was generated using the simulation data collected during post-implementation timing analysis. This file captures the dynamic behavior of the design, including the switching activity of all signals, and is used by Vivado to perform a gate-level power analysis.

Capturing the switching activity for the entire demodulation algorithm runtime is unnecessary. Instead, we set the simulation interval to 1 ms, which allows us to capture multiple iterations of the demodulation algorithm's main loop. This time setting provides a fair power average value that can be extrapolated to the entire program execution, as over 95% of the program runtime occurs within the demodulation loop.

Moreover, the power simulation tool could annotate more than 92% of the nets in all the different implementations of the SoC, performing probabilistic computations for the remaining nets. Using this methodology, we obtained accurate power estimates for each of the five RISC-V processors implemented in the PULPino SoC, providing insight into their respective energy efficiency and resource utilization.

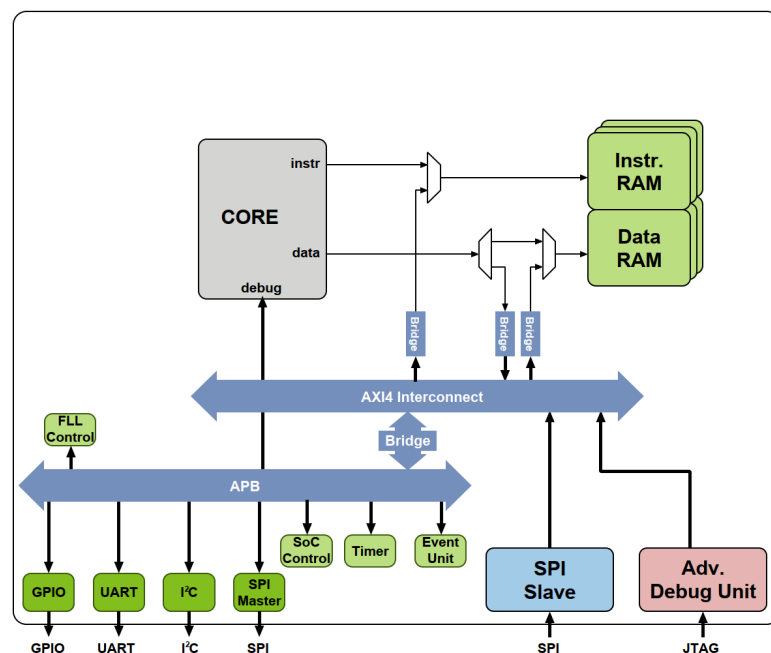


Figure 2. Block diagram of the modified PULPino SoC used to test the different cores.

4. Results

As mentioned in Section 2, RisCO2 is the result of a list of improvements applied to a RISC-V processor previously developed by the authors and presented in [10]. The list of such modifications is detailed in Section 3, and they led to a further reduction in resource utilization and consumed energy compared to the initially proposed processor. The result of applying those actions is summarized in Table 1 step by step, together with the overall reduction in resources and consumed energy.

Table 1. This table shows the result of the improvements made in RisCO2 and enumerated in Section 3. For each applied action, the table shows the FPGA resource utilization, the time it takes for the demodulation algorithm to complete, and the total energy consumed by the processor.

Work	Action item	ISA	LUT	FF	DSP	Clock [MHz]	Time [ms]	Energy [mJ]
[10]	-	rv32imf	7,085	4,188	12	50	12.2	0.45
-	1, 2	rv32em_zfinx	6,518	2,693	12	50	10.3	0.42
-	3	rv32e_zfinx	5,307	2,545	2	50	10.3	0.32
-	5	rv32e_zfinx	5,126	2,478	2	50	10.3	0.31
RisCO2	4	rv32e_zfinx	4,692	2,293	2	50	10.3	0.28
reduction%	-	-	33.8%	45.2%	83.3%	-	15.6%	37.8%

The graphical representation in Figure 3 depicts the incremental performance improvements achieved through the individual actions outlined in Section 3. The plot demonstrates a nearly proportional relationship between energy consumption and the utilization of LUTs in the design.

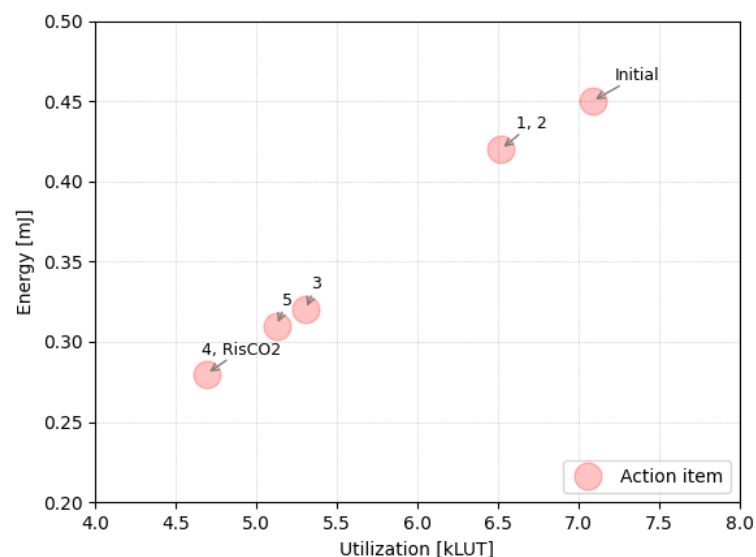


Figure 3. The outcome of implementing the action items enumerated in Section 3 can be observed in the correlation between the FPGA's LUT utilization and the energy consumption of the core.

Table 2 compares the performance of five different RISC-V processors, including RisCO2, Zero-riscy, Micro-Riscy, Ri5cy, and CV32E40P. The comparison is based on FPGA resource utilization (LUT, FF, DSP), the number representation used by the application algorithm (fixed-point or single-precision floating-point), the number of instructions (#instr. $\times 10^6$) that take for the program to demodulate the signal and calculate the CO₂ concentration, instructions per clock cycle (IPC), the time it takes for the demodulation algorithm to complete, and the total power and energy consumed by the processor. The clock frequency used for the comparison is 25 MHz.

RisCO2 outperforms the other processors in terms of energy consumption, with a 53.5% and 63.2% reduction in energy consumption compared to the best-performing ones, CV32E40P and Ri5cy, respectively. Additionally, RisCO2 has a lower FPGA resource utilization compared to these two. In contrast, Micro-riscy has the lowest resource utilization among the five processors and power consumption as low as RisCO2. Still, because of the much longer execution time for the same algorithm, it has a significantly higher energy consumption than RisCO2, more than 19 times higher. On the other hand, Ri5cy has the

most increased resource utilization and consumes 2.7 times more energy than RisCO2, although it has a 50% better performance in terms of instruction throughput (IPC).

Table 2. Comparison of RisCO2 with other RISC-V reference processors in terms of FPGA resource utilization and total energy consumed by the processor for a specific application. Clock frequency = 25 MHz.

Core	ISA	Pipeline stages	LUT	FF	DSP	Var. type	#instr x10 ⁶	IPC	Exec. [ms]	Power [mW]	Energy [mJ]
Micro-riscy	rv32e	2	2,225	1,276	0	Fixed-point	7.24	0.78	373.13	15	5.60
Zero-riscy	rv32im	2	3,171	1,928	1	Fixed-point	1.38	0.82	67.71	20	1.35
Ri5cy	rv32imf	4	11,912	4,249	8	SP Floating	0.28	0.74	15.15	52	0.79
CV32E40P	rv32imf_zfinx	4	9,072	2,553	7	SP Floating	0.25	0.79	12.73	49	0.62
RisCO2	rv32e_zfinx	5	4,889	2,354	2	SP Floating	0.25	0.49	20.71	14	0.29

Figure 4 presents a plot similar to Figure 3 but includes the reference processors examined in our study. Notably, RisCO2 occupies a highly advantageous position within the design space, combining the strengths of both worlds. On the one hand, it shares similarities with Micro-riscy and Zero-riscy processors, characterized by minimal resource utilization and a focus on low-power consumption. On the other hand, RisCO2 draws from the strengths of Ri5cy and CV32E40P, which prioritize achieving maximum performance within a limited power budget.

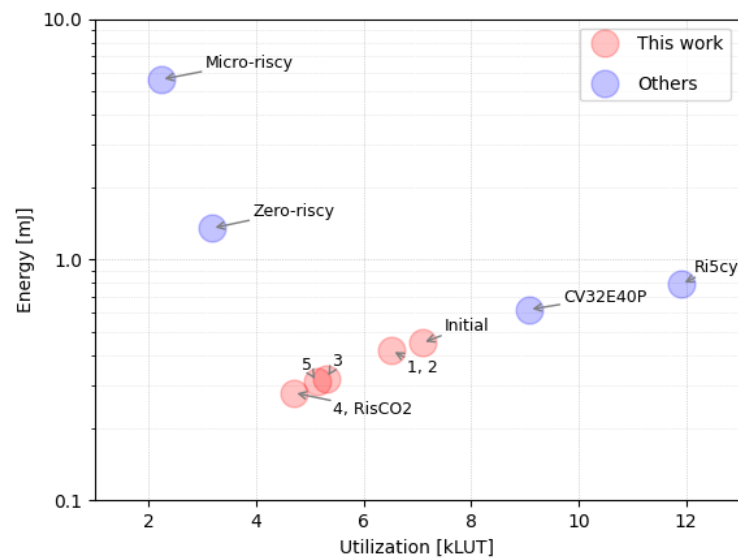


Figure 4. FPGA's LUT utilization and the energy consumption of the different cores tested in this study.

Figure 5 depicts a segmented pie chart illustrating the power distribution among various components of the PULPino platform when integrating the RisCO2 processor for executing the demodulation test program. The implementation was carried out on the Xilinx Nexys-4 board with a clock frequency of 25 MHz. The estimated overall power consumption was 31 mW, and its breakdown is presented in the chart.

The core region consumes 60% of the total power and corresponds to the system components depicted above the AXI interconnect in Figure 2, where the processor occupies the prominent role, but excludes the instruction and data memory, which are represented separately. The processor and the memory alone account for 76% of the total power consumption.

The peripherals in the PULPino platform consume 16% of the power, although the test program does not intensively use them. They serve the purpose of extending the system’s capabilities and facilitating connectivity. They are designed to interface with the processor core, enabling seamless communication with external devices like sensors, actuators, memory, and communication interfaces. Meanwhile, the AXI interconnects, which facilitate efficient communication and data transfer between the processor, peripheral modules, and other system components, consume as little as 4%. Additionally, another small 4% of power consumption corresponds to the leaf cells responsible for crucial functionalities related to interfacing the FPGA with external devices and ensuring optimal signal integrity.

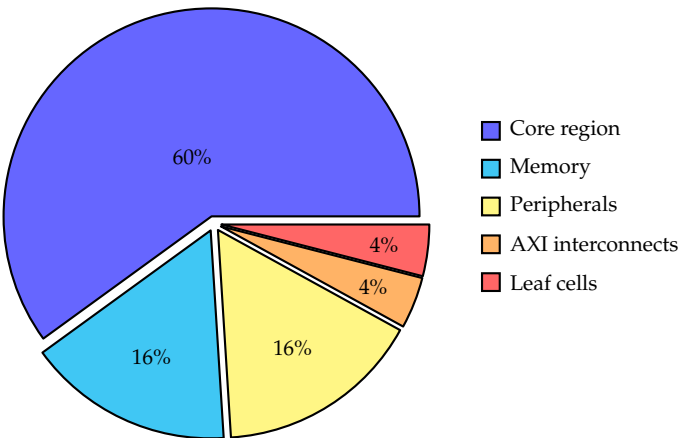


Figure 5. Power distribution among various components of the PULPino platform when integrating a RisCO2 processor.

The pie chart depicted in Figure 6 presents the distribution of power consumption among different modules of the RisCO2 processor during the execution of the demodulation test program, and the power breakdown analysis reveals interesting insights.

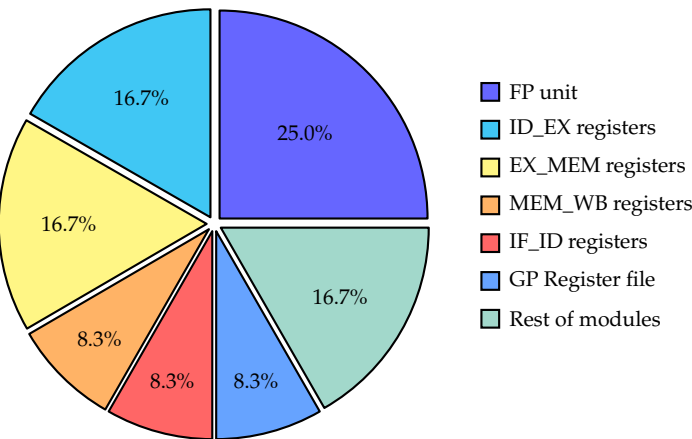


Figure 6. Power distribution among various modules of the RisCO2 processor.

The Floating-Point Unit (FPU) stands out as the most power-hungry component, accounting for 25% of the total power consumption. That is expected since the FPU performs complex floating-point operations that typically require more computational resources and power. The pipeline stage registers also contribute significantly to power consumption. The aggregated power consumption of all the pipeline stage registers in RisCO2 is 50% of the total. These registers play a crucial role in the processor’s instruction execution pipeline, facilitating the flow of data and control signals between different stages. Their relatively high power consumption can be attributed to the need for fast and efficient data transfer within the pipeline. The GP Register file, which contains the processor’s general-purpose

registers, accounts for 8.3% of the power consumption. That indicates that the register file, although essential for storing data during program execution, consumes less power than other critical components.

Lastly, the remaining modules collectively consume 16.7% of the power. This category includes various auxiliary circuits, control logic, and other supporting components necessary for the overall functionality of the processor.

Understanding the power breakdown helps identify the power-intensive areas of the processor design. It provides valuable insights for optimizing power consumption, such as implementing power-saving techniques in the FPU, optimizing data flow in the pipeline registers, or exploring alternative register file designs to reduce power consumption further.

5. Discussion

The study presented a comparative analysis of five RISC-V processor designs in terms of energy consumption and FPGA resource utilization. The processors compared were RisCO2, Ri5cy, Zero-riscy, Micro-riscy, and CV32E40P. RisCO2 is a 32-bit, in-order processor that supports the RV32E_Zfinx instruction set. It is optimized for energy consumption and designed specifically for use in NDIR CO₂ sensors that require signal demodulation. The study used the PULPino SoC platform to implement the five processors and simulate their energy consumption. The results showed that RisCO2 is the most energy-efficient processor, consuming only 0.29 mJ of energy, which is 53.5% and 63.2% less energy than CV32E40P and Ri5cy, respectively. RisCO2 also has the lowest resource utilization compared to the best-performing processors of the study, using only 4,889 LUTs, 2,354 FFs, and 2 DSPs, which is 46.1%, 7.8%, and 71.4% less than CV32E40P, respectively. Our experiments show that RisCO2 is a promising candidate for low-power embedded systems that require efficient processing in complex applications with limited hardware resources. The energy-efficient design of RisCO2 also makes it suitable for battery-powered devices, where minimizing energy consumption is critical.

5.1. Future work

There is potential for improving the performance of RisCO2. The results demonstrate that the instruction throughput (IPC) of RisCO2 is 50% worse than CV32E40P, despite RisCO2 having a deeper pipeline (one more stage) and using the same FPU as CV32E40P. However, increasing the number of pipeline stages can also lead to pipeline hazards and pipeline stalls, decreasing the processor's overall performance. This issue should be further studied and improved.

Another potential area for improvement is the addition of custom instructions to support hardware loops, a feature that is already present in CV32E40P and Ri5cy. Hardware loops have zero stall cycles for jumping to the first instruction of a loop, which could reduce the runtime of the demodulation algorithm since it involves an iterative process with a large number of iterations equal to the number of samples of the modulated signal. However, the addition of these instructions could increase the hardware complexity of the processor and potentially hinder energy consumption savings. Additionally, modifying the compiler is necessary to generate code that uses the new opcodes.

6. Conclusions

In conclusion, the experimental results presented in this study demonstrate the efficiency of the RISC-V architecture for low-power applications, particularly in the context of signal demodulation for NDIR CO₂ sensors. Our RisCO2 processor design, optimized for energy consumption, showed a significant reduction in energy consumption compared to Ri5cy and CV32E40P while still maintaining competitive performance levels. RisCO2 is a promising candidate for low-power embedded systems that require efficient processing in complex applications with limited hardware resources.

Our results also highlight the importance of considering FPGA resource utilization in designing low-power processors, as it can significantly impact the feasibility of the

implementations on resource-limited hardware platforms. Our findings have important implications for the development of energy-efficient processors for low-power applications, especially those requiring real-time signal processing. The results of this study provide a strong foundation for future research in this area, with potential applications in the development of low-power processors for a variety of fields, such as IoT, wearables, and mobile devices. Furthermore, the experimental methodology presented in this study, including the use of switching activity files for power simulation and FPGA implementation, can serve as a valuable reference for other researchers in this area.

Overall, this study contributes to the ongoing efforts to improve the energy efficiency of processors and promote sustainable computing. With the increasing demand for low-power devices in various fields, developing energy-efficient processors is becoming more critical than ever. Our study shows that the RISC-V architecture can offer promising solutions for these challenges, and we hope that our findings will inspire further research and development in this direction.

Author Contributions: Conceptualization, David Castells-Rufas; Funding acquisition, Lluís Terés-Terés; Investigation, Ricardo Núñez-Prieto; Supervision, Lluís Terés-Terés; Writing – original draft, Ricardo Núñez-Prieto; Writing – review & editing, David Castells-Rufas.

Funding: This work is partially funded by the Catalan Government industrial Ph.D. programme under grant 2019-DI-056 and by the Spanish Ministry of Science, Innovation, and Universities under grant RTI2018-095209-B-C22. This research is also part of the activity of the Catalan Government Research Group 2021SGR01623 (CICS, Costume Intelligent Computing Systems).

Data Availability Statement: Data available on request from the authors.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ALU	Arithmetic-logic unit
BRAM	Block RAM
CO ₂	Carbon dioxide
DSP	Digital signal processor
FF	Flip-flop
FPGA	Field-programmable gate array
FPU	Floating-point unit
IDE	Integrated development environment
IoT	Internet of Things
IP	Intellectual Property
ISA	Instruction set architecture
IPC	Instruction per cycle
LUT	Lookup table
NDIR	Non-dispersive infrared
ppm	Parts per million
ROM	Read-only memory
SoC	System on chip

References

1. Popa, D. & Udrea, F. Towards Integrated Mid-Infrared Gas Sensors. *Sensors*. **19** (2019), DOI: [10.3390/s19092076](https://doi.org/10.3390/s19092076)

2. Gibson, D. & MacGregor, C. A Novel Solid State Non-Dispersive Infrared CO₂ Gas Sensor Compatible with Wireless and Portable Deployment. *Sensors*. **13**, 7079-7103 (2013), DOI: [10.3390/s130607079](https://doi.org/10.3390/s130607079)

3. Gautschi, M., Schiavone, P., Traber, A., Loi, I., Pullini, A., Rossi, D., Flamand, E., Gürkaynak, F. & Benini, L. Near-Threshold RISC-V Core With DSP Extensions for Scalable IoT Endpoint Devices. *IEEE Transactions On Very Large Scale Integration (VLSI) Systems*. **25**, 2700-2713 (2017), [10.1109/TVLSI.2017.2654506](https://doi.org/10.1109/TVLSI.2017.2654506).

4. Davide Schiavone, P., Conti, F., Rossi, D., Gautschi, M., Pullini, A., Flamand, E. & Benini, L. Slow and steady wins the race? A comparison of ultra-low-power RISC-V cores for Internet-of-Things applications. *2017 27th International Symposium On Power And Timing Modeling, Optimization And Simulation (PATMOS)*. pp. 1-8 (2017), [10.1109/PATMOS.2017.8106976](https://doi.org/10.1109/PATMOS.2017.8106976).
5. D. Schiavone, "OpenHW Group CV32E40P User Manual", 2020. Available online: https://docs.openhwgroup.org/projects/cv32e40p-user-manual/en/cv32e40p_v1.0.0_doc/index.html (accessed on 05-May-2023).
6. Traber, A., Zaruba, F., Stucki, S., Pullini, A., Haugou, G., Flamand, E., Gurkaynak, F.K. and Benini, L., 2016, January. PULPino: A small single-core RISC-V SoC. *In 3rd RISC-V Workshop*.
7. C. Papon, "VexRiscv: A FPGA friendly 32-bit RISC-V CPU implementation", 2019. Available online: <https://github.com/SpinalHDL/VexRiscv> (accessed on 16-May-2023).
8. Western Digital, "EH1 SweRV RISC-V Core", 2019. Available online: http://github.com/westerndigitalcorporation/swerv_eh1 (accessed on 16-May-2023).
9. Asanovic, K., Avizienis, R., Bachrach, J., Beamer, S., Biancolin, D., Celio, C., Cook, H., Dabbelt, D., Hauser, J., Izraelevitz, A. and Karandikar, S., 2016. The rocket chip generator. EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17.
10. Núñez-Prieto, R., Castells-Rufas, D., Avellana, N., Martínez, R. & Terés, L. Processor Optimization of an Energy-Efficient NDIR CO₂ Wireless Sensor Node. *2022 37th Conference On Design Of Circuits And Integrated Circuits (DCIS)*. pp. 01-06 (2022), DOI: [10.1109/DCIS55711.2022.9970089](https://doi.org/10.1109/DCIS55711.2022.9970089).
11. D. Castells-Rufas, "RISC-V ISA Coverage", 2023. Available online: <https://github.com/davidcastells/riscvisacoverage> (accessed on 22-May-2023).
12. Mach, S., Schuiki, F., Zaruba, F. & Benini, L. FPnew: An open-source multiformat floating-point unit architecture for energy-proportional transprecision computing. *IEEE Transactions On Very Large Scale Integration (VLSI) Systems*. **29**, 774-787 (2020), DOI: [10.1109/TVLSI.2020.3044752](https://doi.org/10.1109/TVLSI.2020.3044752).
13. P. Schregle, "A C++ template class for fixed-point mathematics", 2009. Available online: <http://www.codeproject.com/Articles/37636/Fixed-Point-Class> (accessed on 04-May-2023).

