An Academic RISC-V Silicon Implementation Based on Open-Source Components

Jaume Abella*, Calvin Bulla*, Guillem Cabo*, Francisco J. Cazorla*, Adrián Cristal*, Max Doblas*, Roger Figueras*, Alberto González*, Carles Hernández*, César Hernández†, Víctor Jiménez*, Leonidas Kosmidis*, Vatistas Kostalabros*, Rubén Langarita*, Neiel Leyva†, Guillem López-Paradís*, Joan Marimon*, Ricardo Martínez‡, Jonnatan Mendoza*, Francesc Moll§, Miquel Moretó*§, Julián Pavón*, Cristóbal Ramírez*, Marco A. Ramírez†, Carlos Rojas*, Antonio Rubio§, Abraham Ruiz*, Nehir Sonmez*, Víctor Soria*, Lluís Terés‡, Osman Unsal*, Mateo Valero*§, Iván Vargas*, Luís Villa†

*Barcelona Supercomputing Center (BSC), Barcelona, Spain. Email: name.surname@bsc.es
†Centro de Investigación en Computación, Instituto Politécnico Nacional (CIC-IPN), Mexico City, Mexico.
‡Instituto de Microelectrónica de Barcelona, IMB-CNM (CSIC), Spain. Email: name.surname@imb-cnm.csic.es
§Universitat Politècnica de Catalunya (UPC), Barcelona, Spain. Email: name.surname@upc.edu

***Abstract*—The design presented in this paper, called preDRAC, is a RISC-V general purpose processor capable of booting Linux jointly developed by BSC, CIC-IPN, IMB-CNM (CSIC), and UPC. The preDRAC processor is the first RISC-V processor designed and fabricated by a Spanish or Mexican academic institution, and will be the basis of future RISC-V designs jointly developed by these institutions. This paper summarizes the design tasks, for FPGA first and for SoC later, from high architectural level descriptions down to RTL and then going through logic synthesis and physical design to get the layout ready for its final tapeout in CMOS 65nm technology.**

I. Introduction

Open-source software has represented a revolution in computing especially since the introduction of the Linux operating system. More recently, open Instruction Set Architectures (ISA) have been proposed, with the promise of a similar revolution in the hardware side, more specifically in the design of processors. Open ISAs offer the possibility to implement specific microarchitectures suited for particular applications.

RISC-V is an open ISA originated in 2010 at the University of California at Berkeley [1], and is now supported by the RISC-V International [2] with hundreds of members worldwide. RISC-V is composed of a base instruction set and extended in a modular fashion by a number of dedicated instruction extensions targeting higher performance or specialized application domains.

Since the first public release of the RISC-V ISA, many implementations of processors have risen both from industry and academia. PreDRAC is the first fabricated RISC-V processor from a Spanish or Mexican academic institution. This design will be the first in a series of high performance computing processors designed in the context of the DRAC project (**D**esigning **R**ISC-V-based **A**ccelerators for next generation **C**omputers), which started in June 2019.

This is a Post-Print version of the paper published in DCIS 2020 by IEEE, DOI: 10.1109/DCIS51330.2020.9268664.

This paper describes the design, verification, implementation and post-silicon validation of a RISC-V general purpose processor capable of booting Linux. The design and verification process is described in the paper, starting with the core and the rest of peripherals needed to implement a Linux-capable System-on-Chip (SoC), using open source and internally designed Intellectual Property (IP) blocks.

The team for preDRAC (around 30 people from four different institutions) was put together in January 2019 and the design development took place at three main design abstraction levels: architectural, register transfer level (RTL) and physical; with most of the team dedicated to the architectural and RTL levels. Verification efforts to ensure the integrity of the design between abstraction levels was also an essential part of the work.

The structure of the paper is as follows: Section II describes the chip architecture and IP components. Section III explains the different pre- and post-fabrication verification techniques. Next, Section IV details the synthesis and physical design results. Section V shows the measurements performed on the manufactured chip. Finally, Section VI presents some considerations about the open source processors on silicon and concludes this work.

II. Chip Architecture

The chip adapts an open-source SoC platform developed by lowRISC [3] for the Berkeley Rocket RISC-V core. In the preDRAC implementation, the core has been replaced by Lagarto, a RISC-V processor core developed at CIC-IPN in Mexico.

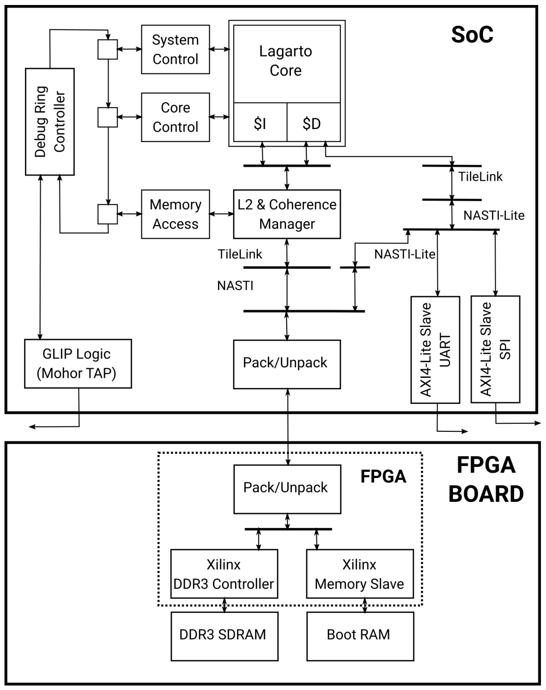


Fig. 1. PreDRAC processor block diagram.

A. Chip Overview

Figure 1 shows the block diagram of the preDRAC single core processor. The processor is a single core design that incorporates a 5-stage single-issue in-order pipeline. It implements the 64-bit RV64IMA scalar RISC-V ISA with a bimodal branch predictor, as well as the associated instruction and data L1 caches (each one 4-way 16KB, 2-cycle access latency) and a unified L2 cache (8-way 64KB, 3-cycle access latency). The main blocks in the cache hierarchy are adapted from the lowRISC project [4]. Moreover, the peripherals required to connect the processor with external devices (i.e. main memory, JTAG, UART and an SD card) combine IPs from different open source projects (i.e. TileLink, GLIP, OpenOCD and MohorTAP [5], [6]) as well as IPs designed in-house. Note that main memory access needs to be done via an FPGA board, where DRAM chips are located.

Table I summarizes the main IP blocks in the preDRAC processor design together with a short description of the block, the source code (some parts of the design are based on other projects), and hardware description language used for each block.

B. Core Microarchitecture

The current core design is strongly based on the Lagarto core, an earlier design at CIC-IPN using the MIPS32 ISA [7]. During the current project, the Lagarto design was ported to RISC-V resulting in a 64-bit in-order core which implements the RV64IMA instruction set architecture (ISA) and supporting the privileged ISA version 1.7. The design is composed by five stages: fetch , decode, read-registers, execution and write-back.

TABLE I
THE MAIN IP BLOCKS OF THE PREDRAC PROCESSOR.

preDRAC processor IP blocks			
IP Block	Description	Source	Language
Lagarto pipeline	RV64IMA 5-stage in-order pipeline, Bimodal Branch Predictor with 1024 entries.	internal	Verilog and Chisel (CSRs)
Instr. cache	4-way 16KB, 2-cycle access latency, VIPT, 64B cache blocks, 8-entry TLB.	open, lowRISC 0.2	Chisel
Data cache	4-way 16KB, 3-cycle blocking access latency, VIPT, 64B cache blocks, 8-entry TLB.	open, lowRISC 0.2	Chisel
L2 cache	8-way 64KB, 3-cycle access latency, PIPT, 64B cache blocks, MESI protocol.	open, lowRISC 0.2	Chisel
TileLink	128-bit wide 0.3.3 version.	open, lowRISC 0.2	Chisel
UART	AXI4-Lite Slave interface, 11 bit per packet, configurable baud rate, parity bit and stop bits, up to 3MBauds.	based on Lagarto SoC v1.0, internal	Verilog
SD Card controller	AXI4-Lite Slave interface, Bidirectional 8-bit wide SPI miso/mosi packets, up to 25Mbps.	based on Lagarto SoC v1.0, internal.	Verilog
JTAG	Communication interface between (PC) C read/write functions and (Internal) in/out FIFOs, uses an FT2232H transciever, clock at 50MHz.	open, GLIP, OpenOCD and MohorTAP.	Verilog, C
Packetizer	AXI-4 front end interface, 64-bit wide, 50 MHz in 65nm TSMC standard I/O	internal.	Verilog
PMU	9 counters, user accessible.	internal	Verilog
Debug ring	start/stop execution, read/write register values, write program to L2 cache.	internal	Verilog

In order to reduce the branch instruction penalty, the design make use of a bimodal branch predictor with 1024 entries.

C. Cache Hierarchy

In this section, the principal characteristics of the Cache Hierarchy used in this project are explained. The preDRAC project adapts the cache design developed in the Untethered lowRISC SoC version 0.2 [3] implementation. The majority of the Cache Hierarchy used here is adopted from that project. One characteristic of this SoC platform is its high configurability: It is possible to modify the number of sets, ways, bits on a set, etc.

The lowRISC Cache Hierarchy has two levels of caches. In the first level, the private caches of each core are located and each core has a First level instruction cache (IL1) and a First level data cache (DL1). In the second level, there is a shared cache.

In the first level, the caches are Virtually Indexed Physically Tagged (VIPT). This means that the index used to access the metadata or data arrays are the virtual ones, but inside the metadata array, the saved tags are the physical ones. This configuration allows to access the Translation Lookaside Buffer (TLB) in parallel with the metadata and data arrays, making the critical path shorter. The only requirement to implement this type of cache is that the page size must be a multiple of the cache way size. For instance, given that DL1 and IL1 cache way size is 4 kB, VIPT is feasible for page sizes of $4 \times N$ kB.

In this hierarchy, the caches are inclusive. In this case, this means that in the second level cache (L2) there is an active copy of all the data that are in the first-level caches. That implies the need to invalidate all the copies in the first level cache of each line evicted from the L2. LowRISC implements a variety of coherence protocols. In the preDRAC tapeout, the protocol chosen is MESI. In this case, a directory is used to maintain the coherence. This directory is located in the last level of cache, the L2.

The replacement mechanism used in the lowRISC caches can only be random replacement. Therefore, on the preDRAC SoC, all the caches are using random replacement protocol. This random replacement uses the lower bits of a 16 bits LFSR in order to select the next set to be replaced. The feedback polynomial is:

$$x^{16} + x^{14} + x^{13} + x^{11} + 1 \tag{1}$$

D. *Access to Main Memory and Peripherals*

Ideally, the objective of the tapeout would be to include the whole preDRAC SoC, together with all the aforementioned peripherals on a single die. However, we faced many challenges regarding the availability of certain technologies and analog and digital IPs that are needed for fabrication.

The lack of a physical interface for a DDR3 memory controller motivated us to design a custom interface to communicate with memory using the physical DDR3 memory from an external FPGA board, in our case a Xilinx Kintex KC705.

This implied to split the original design into two parts, one that contains the memory controllers to access the main memory, and a second one containing the core and rest of the uncore system, including L1 and L2 cache memories. Both parts are connected with an FPGA Mezzanine Card (FMC) cable. This split is depicted in Figure 1. This external connection only supports up to 50 MHz transfer speed, and also the 128-bit bus must be split into 4 transactions of 32 bits each.

E. *Clocking*

Another challenge of this design is the lack of publicly available analog IPs for clock generation. For this reason, a 200 MHz off-chip oscillator is used as the main clock signal.

The custom memory interface with the FMC cable only supported up to 50 MHz clock frequency. This required us to have multiple clock frequencies in the design (see Figure 2). The FMC clock (50 MHz) is simply the ASIC clock divided by 4, and is thus aligned with it, avoiding the need of FIFOs for clock domain crossing.

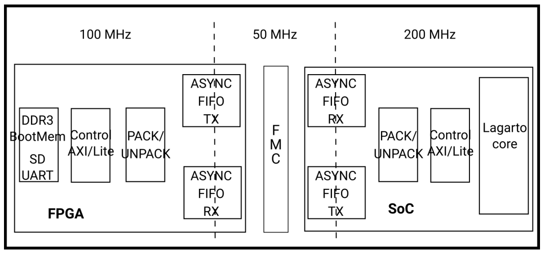


Fig. 2. Clock domains in the preDRAC processor design.

III. Design Verification

A significant effort in the preDRAC processor was devoted to verification at different stages of the design: RTL, post-synthesis and post-place and route.

A. *RTL and Gate Level Simulations*

As part of the verification process, logic simulations pre-synthesis, post-synthesis and post-place and route were performed, using 395 ISA tests. Each test consists of a small program that executes a particular instruction and performs some check to determine if it is executed properly. The applied tests exclude those using floating point and vector units, not present in the design. The result provided by the simulation is a list of the applied tests and whether they were successful or not.

Gate level simulations have also been used to get activity information and perform a more realistic power estimation of the design. Power estimation is performed with Cadence Joules.

B. *Tests on FPGA*

To verify the design before moving to the ASIC design flow, multiple tests are performed with an implementation of the system in FPGA, first in one FPGA to test the standalone SoC, and then in two different FPGAs to mimic the final design.

The testing and verification strategy for the preDRAC design consisted of 4 incremental steps: (i) simple RISC-V ISA tests, which test each instruction, (ii) additional RISC-V lowRISC tests, (iii) random generation of torture tests from lowRISC, as well as (iv) booting the Linux kernel.

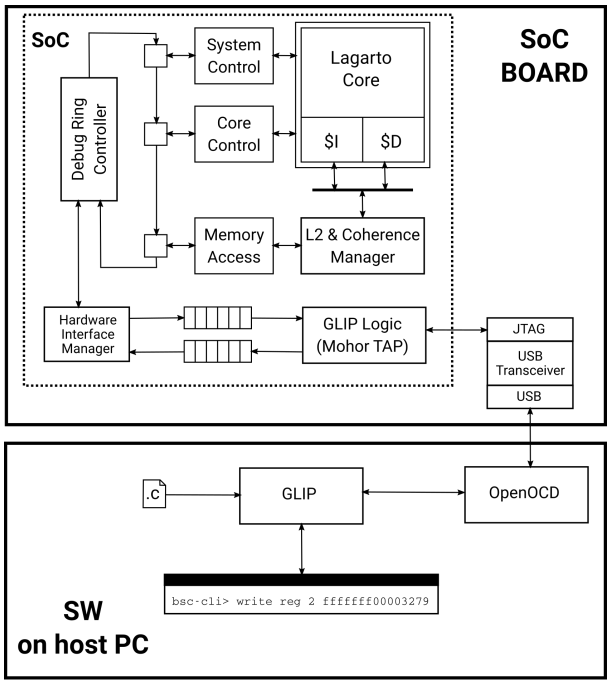
This verification phase allowed to start the ASIC design with confidence in the correct functionality of the system.

C. *Debug Infrastructure for Pre-Silicon Verification and Post-Silicon Validation*

We developed a debug infrastructure to verify at run-time the state of the SoC, and to be able to inject internal tests without using the packetizer interconnection. This system is referred in RISC-V literature as a Debug Ring [4]. Such a system was already developed by lowRISC. We decided to develop our own infrastructure for two reasons. Firstly, the lowRISC debug ring design was only available via a pre-built bitstream for FPGA and thus not synthesizable in an ASIC. Secondly, we needed to include this debug ring because it controls the SoC initialization for Linux booting operations, which means that the SoC would not be capable to boot the Linux kernel without the Debug Ring and user interaction.

In a similar way as the lowRISC debug ring implementation, our implementation is based on the Open SoC Debug Library (OSD) [8], which provides a plug and play communication interface with a base architecture, and the Generic Logic Interfacing Project (GLIP) [6], which gives a generic data exchange protocol based on FIFO queues.

Figure 3 shows the block diagram from the debug ring implementation. It is divided into hardware inside the SoC and a USB transceiver on the daughter board, and software running on a host computer connected through USB. In the software side the important aspects to underline are the usage of the OSD and GLIP C++ libraries which control the communication with the debug interface and allow us to easily connect with different technologies such as JTAG, UART-RS232 and TCP, without changing the C kernel code.



```
bsc-cli> write reg 2 fffffff00003279
```

Fig. 3. Debug Ring Block diagram and interconnection.

IV. Synthesis and Physical Design

The preDRAC SoC is designed using a fully digital design flow based on standard cells in TSMC 65nm technology. Standard cell libraries and SRAM macros were obtained via Europractice. SRAM memories are used for the register file, caches and associated tables. Care was taken to adapt the RTL from the FPGA implementation to an ASIC implementation.

A. Synthesis

The synthesis tool is Cadence Genus. A multi-mode multi-corner flow was followed using 10-track, regular-Vt cells, and three corners: typical, best and worst cases with Non-Linear Delay Model (NLDM) characterization. Table II shows the corner conditions being considered.

TABLE II
List of standard cell corners.

Corner	Conditions (V and T)
Typical	1.2 V and 25 C
Fast	1.32 V and 0 C
Slow	1.08 V and 125 C

Table III lists the memory block sizes used in the design. The same corners as with the Standard Cells were used for synthesis (Typical, Fast and Slow.)

TABLE III
SRAM sizes.

DxW	Size (kb)	Description
32x64	2	Register file banks 0 & 1
1024x28	28	MEM_BIPC 0 & 1
256x128	32	I and D L1 Cache (4 instances)
4096x128	512	L2 Cache
64x80	5	ICache tag
64x88	5.5	MetadataArray_1
1024x2	2	MEM_PHT (2 instances)
1024x40	40	MEM_BTB (2 instances)
128x128	16	Part of 128x176 MetadataArray_0
128x48	6	Part of 128x176 MetadataArray_0

The synthesis results show that there are no violating paths in terms of timing for the defined timing constraints (200 MHz clock). It is worth to mention that to achieve this satisfactory result a few iterations in the design were necessary. In particular, making sure that reset signals acted on all relevant flip-flops was the most time consuming problem, one that became only evident by gate level simulations and not by RTL simulations. On the other hand, we avoided the use of clock domain crossing, which is another source of problems in ASIC designs.

Table IV shows a summary of the gates report, where it can be seen that of a total of 88,960 instances, only 21 correspond to memory cells, that nevertheless represent 79.7% of the cell area and 72.1% of leakage current.

TABLE IV
Number of instances by type and their contribution to area and leakage.

Type	Instances	Area (μm^2)		Leakage power (μW)	
macros	21	1,841,871.1	(79.7%)	75,100.8	(72.1%)
sequential	25,664	263,667.6	(11.4%)	14,051.6	(13.5%)
inverter	4,572	6,449.2	(0.3%)	681.0	(0.7%)
buffer	6,344	11,842.4	(0.5%)	1,385.3	(1.3%)
logic	52,359	185,915.2	(8.0%)	12,917.4	(12.4%)
Total	88,960	2,309,745.5	(100%)	104,136.1	(100%)

TABLE V
POWER ESTIMATION OF THE NETLIST USING ACTIVITY INFORMATION OBTAINED FROM SIMULATION

	Leakage (mW)	Internal (mW)	Switching (mW)	Total (mW)
default	0.10918	190.44509	10.99893	201.55320
add	0.10917	138.12315	2.57574	140.80806
amoand	0.10922	130.48090	2.32073	132.91085
bne	0.10916	138.04635	2.45308	140.60859
mul	0.10917	138.01555	2.57164	140.69636
ld	0.10918	137.02828	2.71643	139.85389
jal	0.10914	138.61095	2.31653	141.03661
sd	0.10923	137.97742	2.73500	140.82164
idle	0.10917	107.07991	2.11543	109.30451

The total area (Top_asic) including a routing estimation is 2.487 mm^2, in line with the initial area budget of 2.5 mm^2 for the core.

The results of the power estimations performed after synthesis are summarized in Table V. The *default* row refers to an estimation where no switching activity information is provided, and the tool calculates the power using a random probability of 20% by default. The next rows provide the power estimation when using the switching information obtained from a set of ISA tests for the following operations:

- *add*: Addition.
- *amoand_d*: Atomic memory operation (AMO) which performs logical AND.
- *bne*: Conditional branch.
- *mul*: Multiplication.
- *ld*: Load a value from memory into a register.
- *jal*: Jump and link. Performs an unconditional jump and stores the address of the instruction following the jump into a register.
- *sd*: Store a value from a register to memory.

Finally, the *idle* row contains the power estimation when the processor is in idle state.

B. Place and Route

Place and Route of the synthesized netlist was executed using Cadence's Innovus and Mentor Calibre for DRC. The chip target was a single module of 2 mm×2 mm for a Multi Project Wafer (MPW) run through Europractice using the 65 nm technology node from TSMC, which uses 10 metal layers. The final chip has 108 pads with 60 μm pitch, including 19 pins dedicated to the power supply. Total chip area is 3.57 mm^2 with only 0.508 mm^2 used for the (156,326) stdcells (taking into account also the CTS and physical cells) of the design, and the rest mainly used for memory blocks (21 macros occupying around 2 mm^2) and pads. The chip is powered at 1.2 V for the core and 2.5 V for the IO pads. The estimated power consumption after place and route is 344 mW using the tool default random activity pattern of 20%. Fig. 4 presents the final layout showing details of the customized floorplan. A photograph of the fabricated chip is shown in Fig. 5.

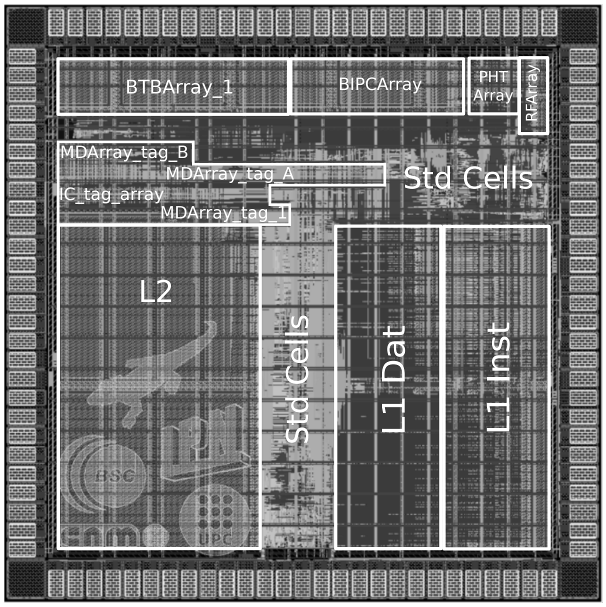


Fig. 4. preDRAC ASIC layout highlighting the different parts of the floorplan. Top metal layer was reserved for logos and pads. Metal filling was performed by TSMC. Side to side size is 1,850 μm.

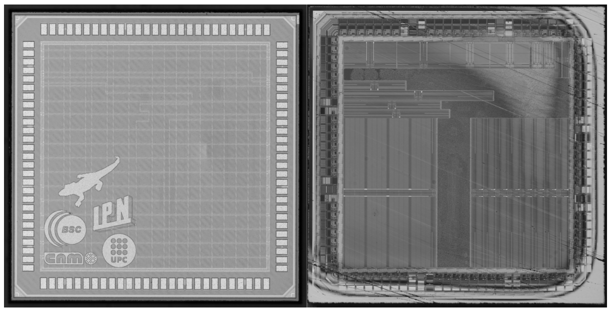


Fig. 5. Microscope photograph of the preDRAC chip (left) and at the polySi level (right).

V. Post-Silicon Results

The preDRAC design was sent to Europractice for fabrication in May 2019 and the chip samples were received in September. To test it, we designed a PCB with all the required circuits (power supply, external 200 MHz clock, reset) and connectors (FMC for the FPGA, JTAG, UART, micro SD).

Before running any code on the fabricated SoC, we performed several tests to the chip to check its operation: adequate standby current, and checking that the implemented clock divider functions properly: from 200 MHz to 50 MHz. This clock divider is used to transmit data at a lower frequency through the custom interface that we implemented.

The following functional tests have been applied:

- JTAG debug-ring interface test by connecting it to a computer running Linux and ensuring successful operation.
- Access to L2 cache by writing/reading through JTAG.
- Exercising fetch and execute cycles on some small kernels written into L2, while checking the register file to

verify the correct code execution.
- Running a set of ISA tests to verify the correctness of individual instructions.
- Running a subset of Mälardalen WCET Benchmarks [9] to get some average performance numbers: IPC of 0.33 with a clock frequency of 200 MHz. These are bare-metal applications stored on the FPGA BootRAM and accessed via the Packetizer.
- Finally, we tested different SoC peripherals (UART, SPI, PMU and DDR3) from kernels loaded into main memory using the custom interface.

Results of above tests concluded that all of them worked as expected, except for the SPI controller, which shows a buggy behaviour and prevented the use of the on-board SD card. As a workaround to this issue, we implemented an alternative path to access the peripherals using the custom interface. We moved the UART and the SPI controllers to the FPGA side, and the SoC uses the custom interface to access them. In this way, we were able to use the external SD card memory and successfully boot Linux.

VI. Conclusions and Future Work

RISC-V open ISA is a great opportunity for educational, research and industry organizations to work on open processor based developments while opening many potential collaborations among these communities. It is expected that this new drive will also impact other hardware development domains other than just processors. Different organizations like OpenHW Group [10], CHIPS Alliance [11], FOSSi Foundation [12] or local networks as RISC-V France [13] and Red-RISCV [14], in addition to the parent organization RISC-V International [2], are working on a collaborative scheme towards this new open hardware paradigm and opportunities. Europe has also decided to work on this direction by applying open-source HW/SW to avoid excessive technology dependencies from third parties, firms or countries, and has chosen the RISC-V ISA as the base-line architecture for next European processors [15].

In this design experience, it was possible to assess the existing resources and limitations of the open-source proposal regarding processor ASIC design.

Many resources do exist, and some of them were indeed leveraged for this project. Processor design may use open-source RTL cores and resources for SoC (e.g. lowRISC, PULP [16]).

It is also true that porting a processor to silicon implies a large effort and currently there are some technical and economic hurdles to surmount that prevent this activity to be a mainstream one like: in depth knowledge of specialized design flows, expensive and proprietary EDA tools, limited or costly access to libraries, memory macros and analog IPs.

Some of these limitations are alleviated by the Europractice program in the case of academic and research institutes, which have access to low cost CAD tools, libraries and manufacturing. In the case of analog IPs for silicon there is not a straightforward solution, although some efforts, such as the Berkeley Analog Generator approach [17], [18] go in the direction to offer open-source analog blocks as well.

In 2020 we plan to fabricate a new iteration of the preDRAC design incorporating more robust verification methodologies and multiple IPs (i.e. a PLL or a new interface with memory), opening the door to future high performance designs.

Once this updated design is being fabricated and tested, we plan to prepare the Lagarto RISC-V core to be offered as free and open-source hardware.

Acknowledgments

The DRAC project is co-financed by the European Union Regional Development Fund within the framework of the ERDF Operational Program of Catalonia 2014-2020 with a grant of 50% of total eligible cost. The authors are part of Red-RISCV which promotes activities around open hardware. The Lagarto Project is supported by the Research and Graduate Secretary (SIP) of the Instituto Politécnico Nacional (IPN) from Mexico, and by the CONACyT scholarship for Center for Research in Computing (CIC-IPN).

References

[1] Andrew Waterman, Yunsup Lee, David A. Patterson, and Krste Asanovic. "The RISC-V Instruction Set Manual, Volume I: Base User-Level ISA". In *Tech. Report UCB/EECS-2011-62, EECS Dept., UC Berkeley*, pages 97–116, 2011.
[2] RISC-V International. https://riscv.org/. [Online; accessed 23-April-2020].
[3] lowRISC. "Untethered lowRISC v0.2". https://www.lowrisc.org/docs/untether-v0.2/. [Online; accessed 27-August-2019].
[4] lowRISC. "Overview of the debug infrastructure". https://www.lowrisc.org/docs/debug-v0.3/overview/, 2018.
[5] U. C. Berkeley. "The TileLink Specification, Version 0.3.3". https://docs.google.com/document/d/1Iczcjigc-LUi8QmDPwnAu1kH4Rrt6Kqi1_EUaCrfrk8. [Online; accessed 23-July-2019].
[6] Institute for Integrated Systems (LIS). "The Generic Logic Interfacing Project". https://www.glip.io/index.html, 2018.
[7] Cristóbal Ramírez, César Hernández, Carlos Rojas Morales, Gustavo Mondragón García, Luís A. Villa, and Marco A. Ramírez. "Lagarto I–Una plataforma hardware/software de arquitectura de computadoras para la academia e investigación". In *Research in Computing Science*, volume 137, pages 19–28, 2017.
[8] The Open SoC Debug Contributors. "The Open SoC Debug Documentation Library". https://opensocdebug.readthedocs.io/en/latest/index.html#, 2018.
[9] "Mälardalen WCET benchmarks homepage". www.mrtc.mdh.se/projects/wcet/benchmarks.html, 2010. [Online; accessed 4-September-2019].
[10] Open Hardware Group. https://www.openhwgroup.org/. [Online; accessed 23-April-2020].
[11] CHIPS Alliance. https://chipsalliance.org/. [Online; accessed 23-April-2020].
[12] FOSSi Foundation. https://fossi-foundation.org/. [Online; accessed 23-April-2020].
[13] RISC-V France. https://www.riscv.fr/. [Online; accessed 23-April-2020].
[14] Red RISC-V. https://www.red-riscv.org/. [Online; accessed 23-April-2020].
[15] European Processor Initiative. https://www.european-processor-initiative.eu/. [Online; accessed 23-April-2020].
[16] PULP Platform. https://pulp-platform.org/. [Online; accessed 23-April-2020].
[17] J. Crossley, A. Puggelli, H. P. Le, B. Yang, R. Nancollas, K. Jung, L. Kong, N. Narevsky, Y. Lu, N. Sutardja, E. J. An, A. L. Sangiovanni-Vincentelli, and E. Alon. "BAG: A designer-oriented integrated framework for the development of AMS circuit generators". In *IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, ICCAD*, pages 74–81, 2013.

[18] Eric Chang, Jaeduk Han, Woorham Bae, Zhongkai Wang, Nathan Narevsky, Borivoje NikoliC, and Elad Alon. “BAG2: A process-portable framework for generator-based AMS circuit design”. In *2018 IEEE Custom Integrated Circuits Conference (CICC)*, pages 1–8. IEEE, apr 2018.