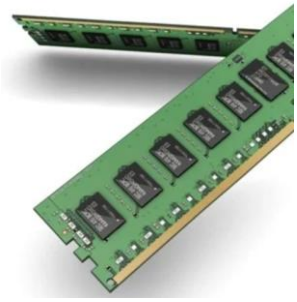


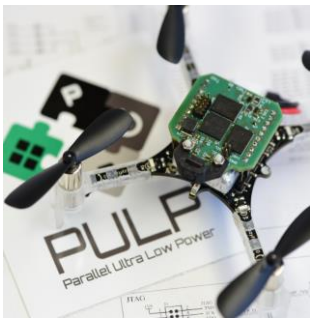

Recommendations and Roadmap for European Sovereignty in Open Source Hardware, Software, and RISC-V Technologies

Report from the

Open Source Hardware & Software Working Group

August 2022



Contents

EXECUTIVE SUMMARY

The use of open source hardware and software drastically lowers the barrier to design innovative System-On-Chips (SoCs), which is an area that Europe excels in today. Open source is a disruptive approach and strong competences in this field are being developed worldwide, for example in China, the US and India. In order for Europe to become a global player in the field and ensure sovereignty, it is important to put in place a roadmap and initiatives that would permit to consolidate ongoing European activities and to develop new technology solutions for key markets such as automotive, industrial automation, communications, health and aeronautics/defence. Notably open source can be used as a sovereignty tool providing Europe with an alternative to licensing IPs from non-EU third parties. A key success criterion for this is for Europe to develop a fully blown open source **ecosystem** so that a European fork is possible (i.e., create a fully equivalent variant of a given technology), if necessary.

The realization of such an ecosystem requires a radical change in working across the board with leadership and contribution from major European industrial and research players and other value chain actors. An approach similar to the European Processor Initiative which brings together key technology providers and users in the supply chain is needed, but with the goal of producing open source IP. This report suggests the following way forward to foster the development of an open source ecosystem in Europe:

- **Build a critical mass of European open-source hardware/software** that will permit to drive competitiveness, enable greater and more agile innovation, and give greater economic efficiency. At the same time, it will remove reliance on non-EU developed technologies where there are increasing concerns over security and safety. In building this critical mass, it is important to do so strategically by encouraging the design of scalable and re-usable technology.
- **Develop both open source hardware and software as they are interdependent.** An open-source approach to software such as EDA and CAD[1] tools can serve as a catalyst for innovation in open-source hardware. To ensure a thriving ecosystem, it is necessary to have accessible software.
- **Address cross-cutting issues.** In order to enable verticalisation, it is important to address a number of cross-cutting issues such as scalability, certification for safety in different application domains, and security. This requires consideration at both the component level and system level.
- **Cultivate innovation** facilitated through funding from the public sector that is conditional on an open-source approach. The public sector can also have a role in aiding the dissemination of open-source hardware through the deployment of design platforms that would share available IP especially those that were supported through public funds.
- **Engage with the open-source community.** Links with initiatives such as OpenHW Group, CHIPS Alliance, RISC-V International etc. should be strengthened to get and maintain industrial-grade solutions whilst encouraging standardisation where appropriate and keeping links with global open source communities.

To this end, the Working Group that is at the origin of this report has defined a strategic roadmap considering short (2-5 years), medium (5-10 years) and long term (>10 years) goals. The success of the roadmap depends on European actors working closely together to create a critical mass of activities that enhance and expand the European open source community and its influence on the world stage. The Working Group advocates that this roadmap of activities is supported via coordinated European level actions to avoid fragmentation and ensure that Europe retains technological sovereignty in key sectors.

[1] In this document CAD tools encompass both classical CAD tools, as well as MCAD, ECAD and system-level tools.

1 INTRODUCTION

Europe has a core competence in designing innovative System-On-Chips (SoC) for key application sectors such as automotive, industrial automation, communications, health and defense. The increasing adoption of open source, however, is potentially disruptive as it drastically lowers the barrier to design and to innovation. Already nations such as China, the US and India are investing heavily in open source HW and SW to remain competitive and maintain sovereignty in key sectors where there are increasing concerns over security and safety.

Europe needs to respond by creating a critical mass in open source. The development of a strong European open source ecosystem will also drive competitiveness as it enables greater and more agile innovation at much lower cost. However, to achieve this there is a need to align and coordinate activities to bring key European actors together.

This document proposes a high-level roadmap for Open Source HW/SW and RISC-V based Intellectual Property (IP) blocks which are of common interest to many European companies and organizations, considering design IP and supporting EDA tools. The aim of the roadmap is to consolidate ongoing European activities through the presentation of a number of concrete ideas, topics and priority areas for IP-blocks and future research in order to build sovereignty and competitive advantage in Europe towards future energy-efficient processor design and usage.

Chapter 2 presents an initial approach to open source, shows how sovereignty can be addressed via open source, and what Europe needs to do to create a strong and competitive critical mass in this area. This is followed by a roadmap for open-source hardware in Europe in Chapter 3, which also highlights the necessary toolbox of open-source IP for non-proprietary hardware to thrive. Chapter 4 presents the supporting software required for the ecosystem to flourish. Chapter 5 discusses some cross-cutting issues – such as scalability, safety and security – for a potential RISC-V processor in common use. Chapter 6 identifies gaps in the current European ecosystem for both hardware and software. In Chapter 7, some important elements of the proposed roadmap are discussed. In turn, Chapter 8 lists the short-, mid- and long-term needs for the different elements of the proposed roadmap. Chapter 9 presents a series of recommendations of both a global and specific nature. Chapter 10 lists a series of horizontal and vertical activities to address the needs of open-source development within Europe. Some concluding remarks and two annexes complete the report.

2 VIEWS OF AN OPEN SOURCE STRATEGY

2.1 Introduction

The interest in open source is rapidly rising as shown in Figure 1 with the number of published academic papers exploding in the area of RISC-V. This presents a major discontinuity for SoC design. The use of open source drastically lowers the barrier to design innovative SoCs which is an area that Europe excels in and strongly depends on in key application sectors. Use of open source also allows a research center or company to focus its R&D effort on innovation, leveraging an ecosystem of pre-validated IP that can be freely assembled and modified. This is disruptive in a market where traditionally a set of IPs is designed in house at a cost that is only accessible to a few companies, or where alternatively 3rd party IPs are licensed by companies with constraints on innovation due to architecture. A key advantage of open source is that it provides a framework that allows academia and companies to cooperate seamlessly, leading to much faster industrialization of research work.

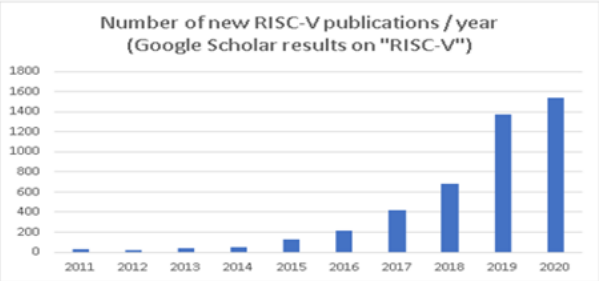


Figure *1* – Google Scholar results on RISC-V (search performed 1 Sept 2021) showing growth of annual publications persisting even despite COVID-19 conference cancellations

There are many parallels between open source hardware and open source software. There have been key exemplars in the software domain such as GNU and Linux. The latter started as an educational project in 1991 and has now become dominant in several verticals (High Performance Computing (HPC), embedded computing, servers, etc.) in synergy with open source infrastructure such as TCP/IP network stacks.

While there are a lot of commonalities in term of benefits, pitfalls and the transformative potential of the industry, there are also significant differences between open source hardware and open source software coming from the physical embodiment of any Integrated Circuit (IC) design via a costly and timely process.

The ecosystem is wide ranging and diverse including the semiconductor industry, verticals and system integrators, SMEs, service providers, CAD tools providers, open source communities, academics and research. It is an area with many opportunities to create innovative start-ups and service offers. The benefits and attraction of adoption of open source depends on the type of actor and their role within the value chain, and can include creating innovative products with lower costs and access barriers, and providing a faster path to innovation.

Annex A gives more details on open source, including potential benefits, the open source ecosystem, key players, business strategies, licensing models, and licensing approaches.

2.2 Current and growing market for RISC-V

Annex B presents an overview of the market trends in the global chips market. The value of this market in 2021 was around USD $550 billion and is expected to grow to USD $1 trillion in 2030.

RISC-based solutions are being used in a growing range of applications and the uptake of RISC-V is expected to grow rapidly.

Within these expanding markets RISC-based architectures have become important. RISC platforms have long enabled mobile phone and small-device consumer electronics vendors to deliver attractive price, performance and energy use compared to complex instruction set computing (CISC)-based solutions such as those of Intel and AMD. Also increasingly, RISC-based solutions are being used to power servers, moving beyond their small-device origins[2] as shown in Figure 2 with a rapid growth in the uptake of RISC-V[3].

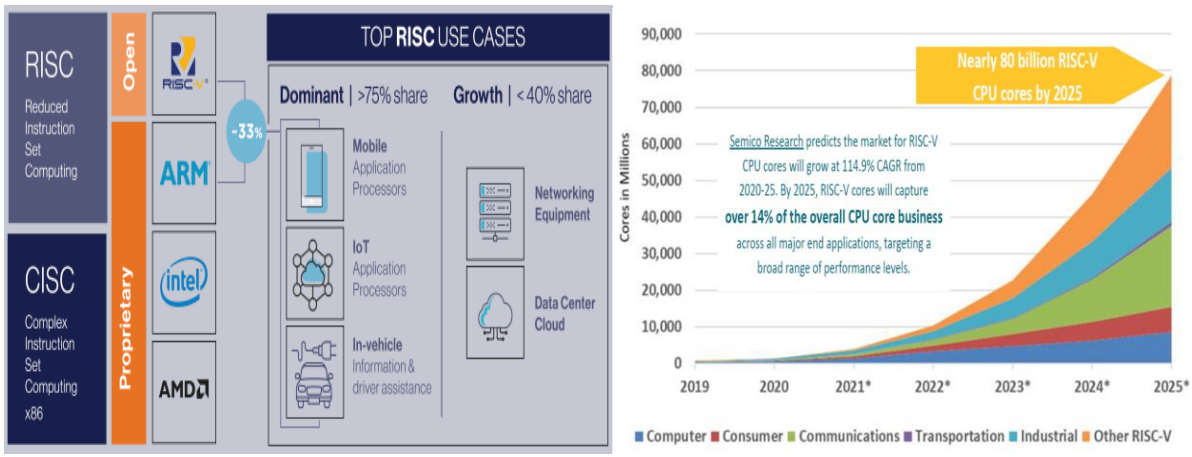


Figure 2 - Growing RISC-based Architectures Usage (Source: Altman Solon and Semico Research Corp.)

	Computer	Consumer	Communications	Transportation	Industrial	Other RISC-V	Total
Adv. Perf Multicore SoC	60%	73%	224%	153%	106%	171%	144%
Value Multicore SoC	60%	92%	222%	159%	122%	201%	177%
Basic SoC	63%	115%	217%	166%	127%	215%	190%
FPGA	62%	72%	190%	163%	102%	176%	149%
Total	61%	81%	209%	160%	110%	185%	158%

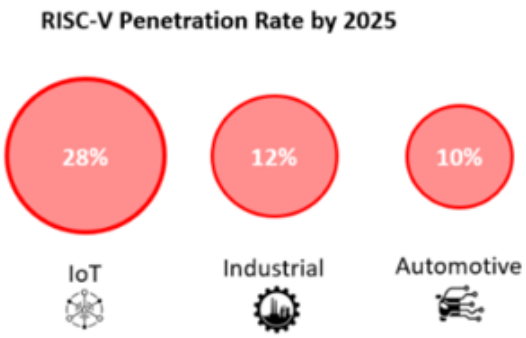


Figure 3 - CAGR for RISC-V and Predicted Penetration Rates in key European sectors

The compound annual growth rate for RISC-V from 2018-2025 is given in Figure 3[4]. In key European markets, the RISC-V penetration rate is expected to grow rapidly[5]. According to "Wilson Research Group

[2] https://www.eetimes.com/taking-a-risc-expanding-chip-options/#
[3] Semico Research Corp., March 2021
[4] Semico Research Corp., March 2021
[5] https://www.counterpointresearch.com/riscv-semiconductor-ip-market-2025/

Functional Verification" **in 2020,[6] 23% of projects in both the ASIC and FPGA spaces incorporated at least one RISC-V processor**.

The current RISC-V instruction set is very popular with RTOs and academics and is also attracting vertical companies that want to develop their own in-house semiconductors (Western Digital[7], Alibaba[8], …). This has led to a number of start-up companies adopting RISC-V to develop specific processors or processor families to sell to the market. Some European examples of success stories, are:

- Greenwaves Technologies (founded in 2014): ultra-low power and AI-enabled GAP processor family for energy constrained products such as hearable, wearables, IoT and medical monitoring products.
- Codasip (founded in 2014): their processor core portfolio combines two complementary processor families, Codasip RISC-V Processors and SweRV Cores, to cover a wide range of design needs. Their business consists in selling of the standard cores that they have developed, as well as to customize the cores to the needs of the customer.
- Cobham Gaisler (founded in 2001): NOEL-V processor family based on 32-bit and 64-bit RISC-V including fault-tolerant implementation and Hypervisor extension, focusing on application of processors in harsh environments.

Here it is useful to look at key European industrial sectors where RISC-V will have an impact.

2.2.1 Automotive Applications

Vehicles need advanced computing power for safety technology such as autonomous driving, but this needs to be provided at reasonable design costs. The cost share of electronic components in relation to the total value of the car is expected to grow from around 16% currently to around 35% by 2025[9]. **RISC-V can help European industry to provide flexible and reusable IP blocks in order to compete on the world market** by providing suitable processing differentiations for safe, secure, and ecofriendly cars built on decades of engineering by world leading European car makers.

There is growing interest in exploiting RISC-V in the automotive, communications and networking, industrial manufacturing, aerospace, health and well-being sectors.

2.2.2 Communication and Networking Applications

Communication and networking chips accounted for USD $140 Bn in 2020, which is around 32% of the total semiconductor market[10]. The communication and networking market is projected to grow at significant CAGR with the increasing demand for smartphones and smart devices around the world. The necessity of working from home is notably rising across developed and developing economies, thus enhancing the demand across this application segment. Upcoming 6G and Wifi7 developments are also important growth opportunities for the semiconductor market in this domain. **RISC-V is expected to be a key driver for new IC's developed for personal devices and communication infrastructure.**

---

[6] https://www.techdesignforums.com/blog/2020/11/27/risc-v-in-nearly-a-quarter-of-designs-wilson-functional-verification-2020-part-one/?mc_cid=4598aad76e&mc_eid=579aaa7d6f

[7] https://www.westerndigital.com/solutions/risc-v

[8] https://www.nextplatform.com/2020/08/21/alibaba-on-the-bleeding-edge-of-risc-v-with-xt910/

[9] https://www.rolandberger.com/en/Insights/Publications/The-car-will-become-a-computer-on-wheels.html

[10] Semiconductor Market Size & Share | Industry Growth [2028] (fortunebusinessinsights.com)

2.2.3 Industrial Applications and Manufacturing

Key technology drivers are the development of digital twins, implementation of AI and Machine Learning, as well as condition monitoring and supporting digital platforms. The expected growth in industrial applications is huge. Predictive maintenance alone is expected to grow by 38% per annum between 2017 and 2022 to reach USD $11 Bn. The additive manufacturing (3D printing) market is predicted to grow annually by 15% between 2015 and 2025 to reach over USD $20 Bn, according to Frost & Sullivan. Another growing area in industrial applications is augmented and virtual reality. In this sector BIS Research forecasts a market of around USD $40 Bn by 2025, i.e., a compound annual growth between 2018 and 2025 of 65%. **RISC-V is one of the strongest growing areas for embedded microprocessors in this field and will become a differentiator**.

2.2.4 Avionics Applications

The size of avionics market is around €50 billion[11]. Within this industry there is a strong safety, security and certification culture. A common challenge is to deploy multicore processing in a safe and secure manner, so that strict real-time deadlines are met, such as prescribed by US and European guidelines, such as CAST-32A.[12] **RISC-V, has the advantage of being inspectable and auditable offering the potential to meet safety requirements for multicore devices in avionics applications**.

2.2.5 Health and Well-Being Applications

Healthcare is already key to the European economy with around 10% of the EU's gross domestic product being spent on the sector. 10% of this spend is on medical technologies which is a market that has historically grown at 4.6% per annum. The size of the global semiconductor market in healthcare was USD $6 Bn in 2020[13], and is forecasted to growing at a CAGR of 11.4% from 2021 to 2026[14]. Healthcare institutions are further integrating technology to make medical treatment more sophisticated, dependable, and accurate. Semiconductor usage is growing in several different applications, e.g., clinical diagnostics and therapy, medical imaging, magnetic resonance imaging, ultrasound equipment, blood pressure monitors and pacemakers. The strongest growth areas are in remote patient monitoring and the use of wearable devices for health monitoring and well-being. In this sector **RISC-V is an enabler for secure, safe and reliable processing in medical devices.**

2.2.6 Hypervisors and Separation Kernel Applications

The global market for hypervisors and safe & secure operating systems (OSs) is expected to grow from USD $391M in 2017 to USD $528M in 2022[15]. Separation kernel vendors such as Green Hills Integrity and SYSGO PikeOS have already announced software support for RISC-V. **The specific value that RISC-V adds here is the complete inspectability of the software and hardware stack, which is important to guarantee non-interference.**

---

[11] MarketsandMarkets. "Avionics Market Size, Growth, Trend and Forecast to 2024." Accessed February 28, 2022. https://www.marketsandmarkets.com/Market-Reports/commercial-avionic-system-market-138098845.html.

[12] Agirre, Irune, Jaume Abella, Mikel Azkarate-Askasua, and Francisco J. Cazorla. "On the Tailoring of CAST-32A Certification Guidance to Real COTS Multicore Architectures." In 2017 12th IEEE International Symposium on Industrial Embedded Systems (SIES), 1–8. Toulouse: IEEE, 2017. https://doi.org/10.1109/SIES.2017.7993376.

[13] Medical Chip market & Semiconductor Market 2020 report (electronicsmedia.info)

[14] Semiconductor in Healthcare Market Size, Trends, Growth Analysis | 2021 to 2026 (marketdataforecast.com)

[15] VDC Research, Hypervisors, Safe & Secure Operating Systems, 2019. https://www.vdcresearch.com/Coverage/IoT-Tech/reports/18-Hypervisors-Operating-Systems.html

2.3 European Strategy

Open source shows lots of promise to boost the economy and welfare in our societies, but this is a non-trivial task. As illustrated with the RISC-V example, a generally agreed mechanism is to use open source to create or enlarge markets and to use proximity, prime-mover and other effects to benefit preferentially from those new markets, combining mostly publicly funded open source and privately funded proprietary offerings. This is well known already for software, and the aim is to replicate this for hardware, allowing the research and industry community to focus on new innovative points, rather than on re-developing a complex framework such as a processor and its ecosystem (compiler, interfaces, ...). There is no way to "protect IP" while still building real open source other than by providing adequate financing for Europe-based entities who work with open source as well as support for regional initiatives and leadership in identifying practical problems that can be solved. There is also a need to help connect open source vendors with customers in need of the innovation. **Speed from concept to product is also key to stay ahead of non-European competitors. This requires foundries and EU financing to build an ecosystem that can get from concept to products quickly in order to maintain European sovereignty.**

Europe is starting initiatives in terms of public adoption of Linux or Matrix in governmental institutions and other areas, and hardware presents a unique opportunity where the ecosystem is immature enough that it could really make a difference. Education is a key factor which could help. Talented and educated Europeans tend to stay in Europe, so significantly boosting open source hardware education could be a way to seed a strong ecosystem. RISC-V is already part of the curriculum in many universities and **Europe could further fund open source hardware education at an even earlier level and generally strongly encourage universities to produce open source.**

The success of an open source project requires efforts going beyond putting some source code in a public repository. The project needs to be attractive and give confidence to its potential users. It is said that the first 15 minutes are the most important. If a potential user cannot easily install the software (in case of open source software), or does not understand the structure, the code and the parameters, it is unlikely that the project will be used. Having good documentation and well written code is important. This requires time and effort to clean the code before it is made accessible. It is also necessary to support the code during its lifetime which requires a team to quickly answer comments or bug reports from users and provide updates and new releases. All modifications should be thoroughly checked and verified, preferably via automated tools. Without this level of support potential users will have little trust in the project. There is thus a need for support personnel and computing resources. This is a challenge for European Open Source projects as the code needs to be maintained after the official end of the project. Only if a project is attractive, provides innovation, is useful and is trusted, will it attract more and more users and backers who can contribute, correct bugs, etc. At this point it can be self-supported by the community with less effort from its initial contributors. The efforts and resources required to support an open source project should not be under-estimated. The project also needs to be easily accessible and hosted in a reliable industrial environment such as the Eclipse foundation, where the Open Hardware Group can provide support for ensuring the success of an open source project.

Success of open source depends not only on the IP blocks but also on the documentation, support and maintenance provided. This requires significant resources and personnel.

It is important to help steer Europe towards open source hardware and make sure that knowledge is shared, accessible, maintained and supported on a long-term basis. Barriers to collaboration need to be eliminated. Priorities should be discussed considering what is happening elsewhere in the world and **to**

maximize the value generated within Europe there is a need to build value chains, develop a good implementation plan and provide long-term maintenance and support.

A more detailed economic view on the impact of Open Source HW and SW on technological independence, competitiveness and innovation in Europe has been documented in a report edited by Fraunhofer ISI and OpenForum Europe[16].

2.4 Maintaining Europe's Sovereignty and Competitiveness

A key market challenge is that most of the current open source hardware technology resources are located outside of Europe, especially in the United States, in China (which has strongly endorsed RISC-V based open source cores) and India which has a national project. This raises the question: how can Europe maintain sovereignty and stay competitive in a rapidly developing open source market? The traditional value of the European landscape lies in its diversity and collaborative nature which reflects the nature of open source very well. EU funding looks favorably on open source solutions and there are well-established educational institutions which are of a less litigious / patent-heavy nature as compared to US private universities. The diversified industry within Europe is also a strength that can be leveraged to push the EU's competitiveness in open hardware.

To get sufficient return on investment for Europe, a focus should be given to application domains where there is stronger impact: automotive, industrial automation, communications, health, defense, critical systems, IoT, cybersecurity, etc. These application domains convey specific requirements towards open source technologies: safety, security, reliability, power and communication efficiency.

A key to sovereignty in the context of open source is the involvement of sufficient European actors in the governance of the various projects (CHIPS alliance, OpenHW group, etc.) and the achievement of a critical mass of EU contributors to these projects so that a fork could be pursued if this is forced onto EU contributing members.

Competitiveness in the context of open source should be looked at in a reversed manner, that is, how much competitiveness European players would lose by not adopting open source. Open source is becoming a major contributor to innovation and economic efficiency of the players who broadly adopt the approach. The massive on-going adoption of open source in China, from leading companies to start-ups and public research centers, with strong support from both the central and regional authorities, is a very interesting trend in China's strategy to catch up in semiconductors.

While this document focuses on open source IC design IPs, the major control point from a sovereignty standpoint is EDA (Electronic Design Automation) tools, which are strictly controlled by

Recommendation – Calls by the Key Digital Technologies Joint Undertaking (KDT JU) should bring benefits to open hardware. Use of this hardware by proprietary demonstrations (including software, other hardware, mechanical systems, in key sectors automotive, industrial automation, etc.) would be beneficial for pushing acceptance of open hardware. Calls could formulate a maximum rate (e.g. 30% of volume) for such demonstrations. It should be required that proposals do not merely implement open interfaces (e.g. RISC-V) but also release implementations (e.g. at least 50% of volume). It is reasonable to demand that the open source licenses of such released implementations allow combination with proprietary blocks.

[16] Knut Blind, Mirko Böhm, Paula Grzegorzewska, Andrew Katz, Sachiko Muto, Sivan Pätsch, Torben Schubert, ''The impact of Open Source Software and Hardware on technological independence, competitiveness and innovation in the EU economy'', Final Study Report, Contract Number LC-01346583, SMART 2019/0011.

the US, even for process nodes for which Europe has sovereign foundries. Europe has significant capabilities in this area in research, which it regularly translates into start-ups for whom the only exit path currently is to be acquired by US controlled firms. Open source EDA suites exist but are limited to process nodes for which PDK is considered as a commodity (90nm and above). **A possible compromise approach, complementary to the development of open source EDA, is to emphasize compatibility of open source IP design with proprietary, closed source design tools and flows**. This objective can be achieved with efforts on two fronts: (I) promote licensing agreement templates for commercial EDA tools that explicitly allow the development of open hardware: this is especially important when open hardware is developed by academic partners in the context of EU-funded R&D actions, as current academic licenses (e.g., as negotiated by EUROPRACTICE) can often not be used for commercial industrial activities. And (II) even more importantly, emphasis should be put in promoting open standards for data exchange of input and outputs of commercial EDA tools (e.g., gate-level netlists, technology libraries). **Proprietary specifications and file exchange formats can hinder the use of industry-strength tools for the development of open source hardware**. This matter requires a specific in-depth analysis to propose an actionable solution.

2.5 Develop European Capability to Fork if Needed

Open source also brings more protection against export control restrictions and trade wars, as was illustrated in 2019 when the US Administration banned Huawei from integrating Google proprietary apps in their Android devices. However, Huawei was not banned from integrating Android as it is open source released. Regarding US export rules, a thorough analysis is provided by the Linux Foundation[17].

Even for technologies developed in other parts of the world that have been open sourced and adopted in Europe at a later stage, **the ability to potentially "fork" i.e., create a fully equivalent variant of a given technology is a critical capability from a digital sovereignty perspective**. Should another geopolitical block decide to disrupt open source sharing by preventing the use of future open source IPs by EU players, the EU would need to carry on those developments with EU contributors only, or at least without the contribution of the adversary block. **For this there needs to be a critical mass of European contributors available to take on the job if necessary.**

The realization of such a critical mass will require an across-the-board change of working, with leadership and contribution from major European players (industrial and research) as well as a myriad other contributors. To achieve this there is a need to build or take part in sustainable open source communities (OpenHW Group, CHIPS Alliance, etc.) to get and maintain industrial-grade solutions. Care needs to be taken to avoid fragmentation (creating too many communities) or purely European communities with a disconnection from global innovation. A challenge is that current communities are young and essentially deliver processor cores and related SW toolchains. They need to extend their offer to richer catalogues including high-end processors, interconnects, peripherals, accelerators, Operating Systems and SW stacks, IDE and SDK, extensive documentation, etc.

Open source hardware targeting ASIC implementations requires software tools for implementation where licensing costs for a single implementation quickly amount to several hundred thousand Euro. **There is thus a need for high quality open source EDA tools supporting industrial-grade open source IP cores**. Europe also has a low footprint in the world of CAD tools, which are critical assets to design and

[17] See https://www.linuxfoundation.org/resources/publications/understanding-us-export-controls-with-open-source-projects/

deliver electronics solutions. Recent open source CAD tool initiatives are opportunities to bridge this gap.

Considering processors **involvement in RISC-V International should be encouraged** to influence and comply with the "root" specifications. RISC-V standardizes the instruction set, but additional fields will be needed in future for interoperability: SoC interconnect, chiplet interfaces, turnkey trust and cybersecurity, safety primitives, etc. **Europe should promote open source initiatives that would help maximize collaborations and reach a significant European critical mass. This is a key issue if Europe is to compete with China and the USA**. **A significant portion of the intellectual property produced in these European initiatives should be delivered as open source**, so that European actors can exploit these results.

There is also a need for an **educational programme to encourage the broader adoption of open source** by the European ecosystem. This needs to tackle barriers such as the perception that building defendable intellectual property is difficult, if not impossible, with open source so that financing becomes more available for open source-based start-ups. This also redefines the business model of the large number of European IP providers, design service providers and public research centers. EU advocacy for open source and financial support, specifically start-ups, SMEs and public research centers adopting open source, would help in securing this transition. There is also a **need for public endorsement of open source and material contribution to open source projects from leading European semiconductor vendors to provide credibility and momentum** to open source at the European level. These actors have a wealth of non-differentiating IPs that they could contribute which would greatly benefit the European ecosystem. In underpinning this, the European design service vendors can play a key role in supporting the open sourcing of major vendors' IPs, with financial support from the EU.

3 TOWARDS AN OPEN SOURCE HARDWARE AND SOFTWARE ROADMAP FOR EUROPE

The roadmap for common priority IPs covers the following areas:

- Processors (RISC-V, beyond RISC-V, ultra-low power and high-end)
- Accelerators -Domain Specific Architectures
- Peripherals and SoC
- Software (Compilers, Debuggers, Operating Systems, etc.)
- EDA Software

Cross cutting systems issues important for applications sectors are also considered such as scalability, safety and security.

3.1 Processors (RISC-V, beyond RISC-V, ultra-low power and high-end)

The Working Group has identified the strategic key needs for the development and support for:

1) a range of different domain focused processors,
2) the IP required to build complete SoCs and,
3) the corresponding software ecosystem(s) for both digital design tools and software development.

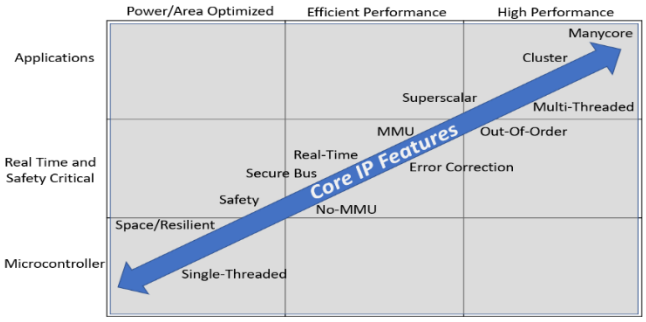


Figure 4 – RISC-V Core IP Roadmap V0.2

To align the software roadmap to the hardware IP core development roadmap, **efforts should be focused on supporting RISC-V implementations that correspond to the RV64GC and RV32GC set of extensions**. At the same time the creation/adoption of RISC-V profiles (microprocessor configurations) should be encouraged in the community. To scope this there is a need for an overall roadmap for RISC-V microprocessor core IP for specific market focus areas. A challenge is that microprocessor core IP comes in many shapes and sizes with differing requirements to meet the needs of applications ranging from 'toasters to supercomputers' as shown in Figure 4. In order to pursue this the Working Group has concentrated on a subset of this broader landscape whilst leveraging the underlying logic and thought process.

To be successful the IP roadmap must encompass attributes such as performance, scalability, software compatibility and deliver high levels of architectural reuse over many years of robust product creation. The focus has been on microprocessor core IP that drives system solutions ranging from edge IOT devices to manycore compute platforms which are key to many European applications. This needs to meet the challenges of scalability and must be sustainable in years to come. At the same time software and digital design tooling needs to be developed to support the foundational RISC-V IP. A key aim has been to leverage global open source hardware and software projects to maximize European reuse, avoid duplication and strategically direct investment.

3.2 Accelerators – Domain Specific Architectures

Accelerators are a key need and for many application domains, as meeting non-functional requirements (e.g., performance, energy efficiency, etc.) is often difficult or impossible using general-purpose

processor instruction sets such as RV32IM or RV64GC. This is true, for example, in machine learning and cryptography, but it also applies to high-performance storage and communications applications. The use of extended instruction sets to enable more parallel processing, such as RISC-V "P" for Single Input Multiple Data (SIMD) processing and RISC-V "V" for full-scale vector processing, can provide significant performance and/or efficiency gains, however, even higher gains are achievable in many cases by adding application domain-specific features to a hardware architecture.

This has led to the concept of Domain-Specific Architectures (DSA) which was highlighted by computer architecture pioneers John Hennessy and David Patterson in their 2018 Turing Award Lecture[18] (the "Nobel Prize" of computer science and computer engineering). Here Domain-Specific Architectures were noted as one of the major opportunities for the further advances in computer architecture to meet future non-functional targets.

Accelerators are one approach to implement a DSA. They do not provide all of a system's functionality, but instead assist more general-purpose processors by the improved execution of selected critical functions. In many cases, this "improvement" means "faster" execution, leading to the name "accelerator", however, it can also mean "more energy efficient" or "more secure", depending on the requirements. In order for the accelerator to exceed the capabilities of a general-purpose system on a similar semiconductor process node, the micro-architecture of the processor is highly specialized and very specific to the actual algorithm it implements. For example, machine-learning accelerators dealing with the efficient inference for Artificial Neural Networks will have very different micro-architectures depending on whether they aim to operate on dense or sparse networks. They, in turn, will have very different architectures from accelerators dealing, e.g., with 3D-Stereovision computations in computer vision for autonomous driving, or Quantum Computing-resistant cryptography for secure communications.

In most cases, accelerators rely on conventional digital logic and are designed/implemented/verified as is usual for the target technology (e.g., FPGA, ASIC). They would thus profit immediately from all open source advances in these areas, e.g., open source EDA tools. Open sourcing the accelerator designs themselves will benefit the ecosystem by encouraging reuse, standardization of testing and verification procedures and in inducing more academic interest and research. Some open sourced accelerators have seen widespread use. Successful cases include NVIDIA's NVDLA[19] Machine Learning inference accelerator, or the 100G hardware-accelerated network stack from ETHZ[20].

The opportunity for much higher impact when using open source for accelerators lies not so much in the accelerators themselves, but in the hardware and software interfaces and infrastructures enabling their use.

Accelerators are integrated into the surrounding computing system typically in one of three approaches (although others exist, e.g., network-attached accelerator appliance).

Custom Instructions: At the lowest level, the accelerator functions can be tightly integrated with an existing processor pipeline and are accessed by issuing special instructions, sometimes called custom instructions or ISA eXtensions (ISAX). This model is suitable for accelerator functions that require/produce only a limited amount of data when operating and thus easily source their operands and sink their result from/to one of the processor's existing register files (e.g., general-purpose, floating-

[18] https://cacm.acm.org/magazines/2019/2/234352-a-new-golden-age-for-computer-architecture/fulltext
[19] http://nvdla.org/
[20] https://github.com/fpgasystems/Vitis_with_100Gbps_TCP-IP

point or SIMD). The key benefit of this approach is the generally very low overhead when interacting with the accelerator from software. This approach has been exploited for decades in processors such as the proprietary Tensilica and ARC cores and is also used, for example, in Intel's AES-NI instructions for accelerating Advanced Encryption Standard (AES) cryptography. In the RISC-V domain, companies such as Andes Technologies (Taiwan) emphasize the easy extensibility of their cores with custom functionality. In the open source area, ETHZ has implemented their Xpulp[21] instructions, showing significant performance gains and code-size reduction over the base RISC-V ISA.

Custom instructions generally require deep integration into a processor's base microarchitecture that is often difficult with many proprietary offerings, e.g., from Arm. In the open source RISC-V hardware ecosystem, though, custom instructions have become a very effective means towards Hennessy & Patterson's Domain-Specific Architectures. **Examples of custom functionality added in this manner include Finite-Field Arithmetic for Post-Quantum Cryptography[22] , digital signal processing[23] , or machine-learning[24] .**

However, due to their need for deep integration into the base processor, such custom instructions are generally not portable between cores. What would be desirable is a lightweight, flexible interface for the interactions between base core and the accelerator logic realizing the custom functionality. This interface would need to be bidirectional as for example instruction decoding would still be performed by the base core, and only selected fields of the instruction would be communicated to the accelerator. The accelerator, in turn, would pass a result to the base core for write-back into the general-purpose register file. Additional functionality required includes access to the program counter computations, to realize custom control flow instructions (this is missing from UCB's RoCC interface), and the load/store-unit(s) for custom memory instructions. A key aspect of an efficient custom instruction interface will be that the unavoidable overhead (hardware area, delay, energy) is only paid for those features that are actually used. For example, a simple Arithmetic Logic Unit (ALU) compute operation should not require interaction with the program counter or memory access logic**. With a standard (de-facto or formal) interface in place, R&D work on tightly integrated accelerators could port across different base processors.**

Loosely Coupled Accelerators: More complex accelerators that require or produce more data than can easily be provided/accepted by a base processor core are generally not integrated into the core pipeline itself but are coupled more loosely using industry-standard protocols such as Arm AMBA. The communication requirements of the accelerator determine the specific flavour of interface used, which can range from Advanced Peripheral Bus (APB) for low-bandwidth interaction, or Advanced High-Performance Bus (AHB)/Advanced eXtensible Interface (AXI) for higher-performance accelerators. These loosely coupled accelerators accept commands from one or more base cores using memory-mapped interactions and are then capable of accessing their input and output data independently using Direct Memory Access (DMA) operations. Since these accelerators are integrated into a system using standard

[21] https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/461404/CARRV2020_paper_12_Perotti.pdf?sequence=1&isAllowed=y

[22] https://www.athene-center.de/en/research/publications/?tx_vierwdcrisp_publications%5Bpublication%5D=2338&tx_vierwdcrisp_publications%5Baction%5D=show&cHash=aba405ca3a9a90d328778a67ce99e7d2

[23] https://riscv.org/wp-content/uploads/2017/12/Tue1212-Customization_of_a_RISC_V_Processor_Zachariasova.pdf

[24] https://edge.seas.harvard.edu/files/edge/files/carrv_workshop_submission_2019_camera_ready.pdf

protocols, they are very portable. **For example, the open source NVIDIA machine-learning accelerator has been successfully integrated into a number of open source and proprietary Systems-on-Chips.**

However, for more complex accelerators, the open source ecosystem is much sparser when advanced capabilities are required. Two examples of this are shared-virtual memory and/or cache-coherent interactions between the accelerators and the software-programmable base cores. Both mechanisms can considerably simplify using the interaction between software on the base cores and accelerators, especially when more complex data structures (e.g., as used in databases) need to be operated on. To achieve shared-virtual memory, the accelerator and the base core(s) need to share address translations, invalidations, and fault-handling capabilities. The UCB RoCC interface achieves this by leveraging the capabilities of the underlying base core (Rocket, in most cases). However, for better scalability across multiple accelerators, it would be beneficial to have a dedicated Input-Output Memory Management Unit (IOMMU) serving just the accelerators, possibly supported by multiple accelerator-specific Translation-Lookaside Buffers (TLBs) to reduce contention for shared resources even further. However, no such infrastructure exists in an open source fashion. Currently, designers following Hennessy & Patterson's ideas towards accelerator-intensive systems do not just have to design the (potentially complex) accelerators themselves, but they also often have to start "from scratch" when implementing the supporting system-on-chip architecture allowing these accelerators to actually operate. **Portable, scalable and easily accessible open source solutions are sorely needed to lower this barrier of entry.**

Recommendation - **Low-level generic and higher-level domain-specific frameworks should be made available in an easily accessible open source manner. For the lower-level frameworks, the goal should be portable and scalable solutions that support the selected model(s) of computation in a *lightweight modular* manner.**

<u>High-Speed DRAM Interfaces</u>: Accelerators are often employed for highly data-intensive problems (e.g., graphics, vision, machine-learning, cryptography) that need to store significant amounts of data themselves, and/or require high-performance access to data via a network. Thus, the availability of high-speed interfaces to Dynamic Random Access Memory (DRAM), ideally even in the form of forward-looking 2.5D/3D memory technologies such as High Bandwidth Memory (HBM), and to a fast network or peripheral busses such as PCI Express, are absolutely crucial for the use of and research into accelerators. In most cases, it does not make sense to design an accelerator if it cannot interact with sufficient amounts of data. Moving up further in the layers of systems architecture, accelerators are not just used today as IP blocks in a system-on-chip, but also as discrete expansion boards added to conventional servers, e.g., for datacenter settings. **This not only applies to Graphics Processing Units (GPUs), as the most common discrete accelerator today, but also to many machine learning accelerators such as Google's TPU series of devices**. This usage mode will become even more common, now that there is progress on the required peripheral interfaces for attaching such boards to a server, specifically: Peripheral Component Interconnect Express (PCIe) has finally picked up again with PCIe Gen4 and Gen5. Not only do these newer versions have higher transfer speeds, they also support the shared-virtual

Recommendation - **To enable easier design and use of novel accelerator designs open source system-on-chip templates should be developed providing all the required external (e.g., PCIe, network, memory) and internal interfaces (e.g., Arm AMBA AXI, CHI, DRAM and interrupt controllers etc.), and into which innovative new accelerator architectures can be easily integrated. These template SoCs could then be fitted to template PCBs providing a suitable mix of IO and memory (as inspired by FPGA development boards) to the custom SoC. Ideally, all of this would be offered in a "one-stop-shop" like approach, similar to the academic/SME ASIC tape-outs to EUROPRACTICE for fabrication but in this case functional PCIe expansion boards.**

memory and cache-coherent operations between host and the accelerator board, using protocol variants such as Cache Coherent Interconnect for Accelerators (CCIX) or Compute Express Link™ (CXL). **Designing and implementing a base board for a PCIe-based accelerator, though, is an extremely complex endeavor.** Not just from a systems architecture perspective, but there are also many practical issues, such as dealing with high-frequency signals (and their associated noise and transmission artifacts), providing cooling in the tight space of a server enclosure, and ensuring reliable multi-rail on-board power-supplies.

These issues are much simplified when using one of the many FPGA based prototyping/development boards which have (mostly) solved these problems for the user. High-speed on-chip interfaces are provided by the FPGA vendor as IP blocks, and all of the board-level hardware comes pre-implemented. While these boards are generally not a perfect match to the needs of a specific accelerator (e.g., in terms of the best mix of network ports and memory banks), a reasonable compromise can generally be made choosing from the wide selection of boards provided by the FPGA manufacturers and third-party vendors. This is not available, however, to academic researchers or SMEs that would like to demonstrate their own ASICs as PCIe-attached accelerators. **To lower the barrier from idea to usable system in an open source ecosystem, it would be highly desirable if template Printed Circuit Board (PCB) designs like FPGA development boards were easily available, into which accelerator ASICs could easily be inserted, with all of the electrical and integration issues already being taken care of.** Providing these templates with well-documented and verified designs will incentivize designers to release their work in the public domain.

Hardware is one aspect of an accelerator, but it also needs to be supported with good software. This can range from generic frameworks, e.g., wrapping task-based accelerator operations using the TaPaSCo[25] system, to domain-specific software stacks like Tensor Virtual Machine (TVM)[26] for targeting arbitrary machine-learning (ML) accelerators. Combinations of software frameworks are also possible, e.g., using TaPaSCo to launch and orchestrate inference jobs across multiple TVM-programmed ML accelerators[27].

To enable broader and easier adoption of accelerator-based computing, a two-pronged approach would be most beneficial. Lower-level frameworks, like TaPaSCo, should provide broad support for accelerators operating in different models of computation, e.g., tasks, streams/dataflow, hybrid, Partitioned Global Address Space (PGAS), etc., drawing from the extensive prior work in both theoretical and practical computing fields. Many ideas originating in the scientific and high-performance computing fields originally intended for supercomputer-scale architectures have become increasingly applicable to the parallel system-on-chip domain. Lower-level frameworks can be used to provide abstractions to hide the actual hardware-level interactions with accelerators, such as programming control registers, setting up

Recommendation - For higher-level frameworks, future open source efforts should be applied to leveraging existing domain-specific solutions. Here, a focus should be on making the existing systems more accessible for developers of new hardware accelerators. Such an effort should not just include the creation of "cookbook"-like documentation, but also scalable code examples that can be easily customized by accelerator developers, without having to invest many person-years of effort to build up the in-house expertise to work with these frameworks.

[25] https://github.com/esa-tu-darmstadt/tapasco
[26] https://tvm.apache.org/
[27] https://sampl.cs.washington.edu/tvmconf/slides/2019/E12-Florian-Stock.pdf

memory maps or copies, synchronizing accelerator and software execution, etc. Application code can then access the accelerator using a high-level, but still domain-independent model of computation, e.g., launching tasks, or settings up streams of data to be processed by the hardware.

Even higher levels of abstraction, with their associated productivity gains, can be achieved by making the newly designed accelerators available from domain-specific frameworks. Examples include the TVM for machine learning, or Halide[28] for high-performance image and array processing applications. These systems use stacks of specialized Intermediate Representations (IR) to translate from domain-specific abstractions down to the actual accelerator hardware operations (e.g., TVM employs Relay[29], while Halide can use FROST[30]). Ideally, to make a new accelerator usable in one of the supported domains would just require development of a framework back-end for mapping from the abstract IR operations to the accelerator operations and the provision of an appropriate cost-model, to allow the automatic optimization passes included in many of these domain-specific frameworks to perform their work.

It should be noted that the development of OpenCL 2.x is a cautionary tale of what to *avoid* when designing/implementing such a lower-level framework. Due to massive overengineering and design-by-committee, it carried so many mandatory features as baggage that most actual OpenCL implementations remained at the far more limited OpenCL 1.x level of capabilities. This unfortunate design direction, which held back the adoption of OpenCL as a portable programming abstraction for accelerators for many years, was only corrected with OpenCL 3.0. This version contains a tightly focused core of mandatory functionality (based on OpenCL 1.2) supported by optional features that allow tailoring to the specific application area and target accelerators.

From a research perspective, it would also be promising to study how automatic tools could help to bridge the gap between the domain-specific frameworks, e.g., at the IR level, and the concepts used at the accelerator hardware architecture levels. Here, technologies such as the Multi-Level IR (MLIR)[31] proposed for integration into the open source Low Level Virtual Machine (LLVM) compiler framework may be a suitable starting point for automation.

3.3 Peripherals and SoC Infrastructure

3.3.1 SoC Infrastructure

In addition to processor cores, it is also very important to have a complete infrastructure to make SoCs and be sure that all IPs are interoperable and well documented (industry grade IPs). This requires the necessary views of IPs (IDcard with maturity level, golden model, Register-Transfer Level (RTL), verification suite, integration manual, Design For Test (DFT) guidelines, drivers) which are necessary to convince people to use the IPs. As highlighted system-on-chip (SoC) templates are needed that provide all the required external

Recommendation - Blocks do not solely consist of (relatively portable) digital logic, but they also have analog components in their high-speed physical interfaces (PHY layer) that must be tailored to the specific ASIC fabrication process used. Such blocks should be provided for the most relevant of the currently EUROPRACTICE-supported processes for academic/SME prototype ASIC runs. The easy exchange of IP blocks between innovators is essential leveraged by both the EDA companies and IC foundries. An exemplar is the EUROPRACTICE enabled R&D structure of CERN where different academic institutions collaborate under the same framework to exchange IP.

[28] https://halide-lang.org/
[29] https://arxiv.org/pdf/1810.00952
[30] http://groups.csail.mit.edu/commit/papers/2018/frost_workshop.pdf
[31] https://mlir.llvm.org/

(e.g., PCIe, network, memory) and internal interfaces and infrastructures (e.g., Arm AMBA AXI, Coherent Hub Interface (CHI), DRAM and interrupt controllers, etc.), and into which innovative new IPs could be easily integrated. High-speed lower-level physical interfaces (PHYs) to memories and network ports, are designed at the analog level, and are thus tailored to a specific chip fabrication process. The technical details required to design hardware at this level are generally only available under very strict NDAs from the silicon foundries or their Physical Design Kit (PDK) partners. Providing access to this information would involve inducing a major shift in the industry. As a compromise solution, though, innovation in open source hardware could profit immensely if these lower-level interface blocks could be made available in a low-cost manner, at least for selected chip fabrication processes supported by facilitators such as EUROPRACTICE for low-barrier prototyping (e.g., the miniASIC and MPW programs).

For open source hardware to succeed, standard interfaces such as DRAM controllers, Ethernet Media Access Controllers (MACs) and PCIe controllers need to be either available as open source itself, or at a very low cost at least for academic research and SME use. If they are not, then the innovation potential for Europe in both of these areas will often go to waste, because new ideas simply cannot be realized and evaluated beyond simulation in real demonstrators/prototypes. For a complete SoC infrastructure, it is necessary to have an agreed common interconnect at the:

- processor/cache/accelerator level (similar to the Arm Corelink Interconnect, or the already existing interconnect specifications for RISC-V such as TileLink.[32]): in the open source area, support for cache-coherency exists in the form of the TileLink protocol, for which an open source implementation is available from UC Berkeley. However, industry standard protocols such as AMBA ACE/CHI do not have open source implementations (even though their licenses permit it). This makes integration of existing IP blocks that use these standard interfaces with open source systems-on-chips difficult.
- memory hierarchy level, for example between cores for many-core architectures (NUMA effects) and between cores and accelerators,
- peripheral level (e.g., Arm's AMBA set of protocols).

The communication requirements of the accelerators determine the specific flavor of interface used, which can range from APB for low-bandwidth interaction, or AHB/AXI for higher-performance accelerators.

3.3.2 Networks on a Chip

Networks on a Chip (NoCs) and their associated routers are also important elements for interconnecting IPs in a scalable way. Depending on requirements, they can be synchronous, asynchronous (for better energy management) or even support 3D routing. Continuous innovations are still possible (and required) in these fields.

3.3.3 Verification and Metrics

IPs need to be delivered with a verification suite and maintained constantly to keep up with errata from the field. **For an end-user of IP the availability of standardized metrics is crucial as the application scenario may demand certain boundaries on power, performance, or area of the IP.** This will require searches across different repositories with standardized metrics. One industry standard benchmark is

[32] https://sifive.cdn.prismic.io/sifive%2Fcab05224-2df1-4af8-adee-8d9cba3378cd_tilelink-spec-1.8.0.pdf or https://bar.eecs.berkeley.edu/projects/tilelink.html

from the Embedded Microprocessor Benchmark Consortium (EEMBC)[33], however, the topic of metrics does not stop at the typical performance indicators. It is also crucial to assess the quality of the verification of the IP with some metrics. In a safety or security context, it is important for the end-user to assess in a standardized way the verification status in order to conclude what is still needed to meet required certifications. **Using standardized metrics allows end users to pick the most suited IP for their application and get an idea on needed additional efforts in terms of certifications.**

Another aspect is in providing trustworthy electronics. This is a continuous effort in R&D, deployment and operations, and along the supply chains. This starts with trustworthy design IPs developed according to auditable and certifiable development processes, which give high verification and certification assurance (safety and/or security) for these IPs. These design IPs including all artefacts (e.g., source code, documentation, verification suites) are made available ensuring integrity, authenticity and traceability using certificate-based technologies. Traceability along the supply chain of R&D processes is a foundation for later traceability of supply chains for components in manufacturing and deployment/operation.

3.3.4 Chiplet and Interposer Approach

Another domain which is emerging and where Europe can differentiate itself is in using the 2.5D approach, or "chiplets + interposers". This is already enabled by EUROPRACTICE for European academics and SMEs. **The idea is to assemble functional circuit blocs (called chiplets, see https://en.wikichip.org/wiki/chiplet) with different functions (processor, accelerator, memories, interfaces, etc.) on an "interposer" to form a more complex System-on-Chip**. The interposer technology physically hosts the chiplets and ensures their interconnection in a modular and scalable way, like discrete components on a Printed Circuit Board. This approach can range from low-cost systems (with organic interposers), to high-end silicon-based passive and active interposers, up to photonic interposers. In active interposers, the interposer also includes some active components that can help with interconnection (routers of a NoC), interfacing or powering (e.g., Voltage converters).

The industry has started shifting to chiplet-based design whereby a single chip is broken down into multiple smaller chiplets and then "re-assembled" thanks to advanced packaging solutions. **TSMC indicates that the use of chiplets will be one of the most important developments in the next 10 to 20 years. Chiplets are now used by Intel, AMD and Nvidia and the economics of this has already been proved by the success of the AMD chiplet-based Zen architecture.** As shown by the International Roadmap for Devices and Semiconductors (IRDS) roadmap, chiplet-based design is considered as a complementary route for More Moore scaling.

Chiplet-based design is an opportunity for the semiconductor industry, but it creates new technical challenges along the design value chain: architecture partitioning, chiplet-to-chiplet interconnect, testability, CAD tools and flows and advanced packaging technologies. None of the technical challenges are insurmountable, and most of them have already been overcome through the development and characterization of advanced demonstrators. They pave the way towards the "domain specific chiplet on active interposer" route for 2030 as predicted by the International Roadmap for Devices and Semiconductors.

With chiplet-based design, the business model moves from a soft IP business to a physical IP business, one in which physical IP with new interfaces is delivered to a new actor who integrates it with other

[33] http://eembc.org

outsourced chiplets, tests the integration and sells the resulting system. **According to Gartner, the chiplet market size (including the edge) will grow to $45B in 2025** and supporting chiplet-based design tools are available. A challenge is that the chiplet eco-system has not yet arrived due to a lack of interoperability between chiplets making chiplet reuse difficult. For instance, it is not possible to mix an AMD chiplet with a XILINX one to build a reconfigurable multi-core SoC. **Die-to-Die (D2D) communication is the "missing link" to leverage the chiplet-based design ecosystem, and its development in open source would enable a wide usage and could become a "de-facto" standard.**

The Die-to-Die interface targets a high-bandwidth, low-latency, low-energy ultra-short-reach link between two dies. Different types of interfaces exist and the final choice for a system lies in the desire to optimize six often competing, but interrelated factors:

1. Cost of packaging solutions
2. Die area per unit bandwidth (square mm per Gigabits per second)
3. Power per bit
4. Scalability of bandwidth
5. Complexity of integration and use at the system level
6. Realizability in any semiconductor process node

The ideal solution is an interconnect technology that is infinitely scalable (at fine-grained resolution), low power, area-efficient, totally transparent to the programming model, and buildable in a low-cost silicon and packaging technology. There are two classes of technologies that service this space:

- Parallel interfaces: High-Bandwidth Interface (HBI), Advanced Interface Bus (AIB) and "Bunch of Wires" (BoW) interfaces. Parallel interfaces offer low power, low latency and high bandwidth, but at the cost of requiring many wires between Die. The wiring requirement can only be met using Silicon interposer or bridging technology.
- Serial Interfaces: Ultra Short and eXtra Short Reach SerDes. Serial interfaces significantly reduce the total number of IOs required to communicate between semiconductor chips. They allow the organic substrate to provide the interconnection between dies and enable the use of mature System-in-Package technology.

One difficulty of the Die-to-Die approach is that no communication standard currently exists to ensure interoperability. In early 2020, the American Open Compute Project (OCP) initiative addressed the Die-to-Die standardization by launching the Open Domain Specific Architecture (ODSA) project that aims to bring more than 150 companies to collaborate on the definition of different communication standards suitable for inter-die communication.

Recommendation - The chiplet-based approach is a unique opportunity to leverage European technologies and foundries creating European HW accelerators and an interposer that could leverage European More-than-More technology developments. To achieve this, inter-operability brought by open source HW is key for the success, together with supporting tools for integration, test and validation.

The SoC infrastructure for "chiplet-based" components will require PHY and MAC layers of chiplet-to-chiplet interfaces based on standard and open source approaches. These interfaces could be adapted depending on their use: data for computing or control for security, power management, and configuration. This interposer + chiplet approach will leverage European technologies and even foundries, as the interposer does not require to be in the most advanced technology and could embed parts such as power converters or analog interfaces. The chiplets can use the most appropriate technology for each function

(memory, advance processing, support chiplets and interfaces). **Interoperability, that could be brought by open source HW, is key for the success, together with supporting tools for integration, test and validation.**

Regarding the connection of SoCs to external devices, some serial interfaces are quite mature in the microcontroller world and there is little differentiation between vendors in the market. It would make sense to align on standard implementations and a defined set of features that can be used by different parties. Open source standard implementations could contribute to the distribution of standards. However, there are also domain specific adaptions that require special features which would make it hard to manage different implementations.

4 SUPPORTING SOFTWARE

The software infrastructure necessary for a successful hardware ecosystem contains Virtual Prototypes (Instruction Accurate and Clock Accurate simulators), compilers and linkers, debuggers, programmers, integrated development environments, operating systems, software development kits and board support packages, artificial intelligence frameworks and more. Indeed, **the idea of open source originates from the software world and there are already established and futureproof software projects targeting embedded systems and hardware development such as LLVM, GDB, OpenOCD, Eclipse and Zephyr**. The interoperability and exchangeability between the different parts of the SW infrastructure are important fundamentals of the ecosystem.

4.1 Virtual Prototypes

Virtual Prototypes (VP) play a major role in different phases of the IP development and require different types of abstraction. They can range from cycle accurate models which are useful for timing and performance estimations, to instruction accurate models, applicable for software development, design-space exploration, and multi-node simulation. **Independent of the VP abstraction level, these platforms should strive towards modelling the IP to be functionally as close to real hardware as possible**, allowing users to test the same code they would put on the final product.

As modeling IP is usually a task less complex than taping out new hardware revisions, VPs can be effectively used for pre-silicon development. Modeling can be either done in an abstract way, or using RTL, or by mixing these two approaches in a co-simulated environment. VPs can bring benefits not only to hardware manufacturers that want to provide software support for their customers but also to customers in that they can reuse the same solutions to develop end products. Having models for corresponding open source IP could be beneficial for establishing such IP. With this in mind it would be reasonable to provide a permissively licensed solution, allowing vendors to close their non-public models.

4.2 Compilers and Dynamic Analysis Tools

Compilers significantly influence the performance of applications. Important open source compiler projects are LLVM (Low Level Virtual Machine) and GCC (GNU Compiler Collection). The **LLVM framework is evolving to become the 'de facto' standard for compilers**. It provides a modular architecture and is therefore a future-proof solution compared to the more monolithic GCC.

MLIR (Multi-Level Intermediate Representation) is a novel approach from the LLVM framework to building a reusable and extensible compiler infrastructure. MLIR aims to address software fragmentation, improve compilation for heterogeneous hardware, significantly reduce the cost of building domain specific compilers, and aid in connecting existing compilers together. Such flexibility on the compiler side is key to providing proper software support for the novel heterogeneous architectures made possible by the flexibility and openness provided by RISC-V. Note that both LLVM and GCC include extensions such as AddressSanitizer or ThreadSanitizer that help developers to improve code quality.

In addition, there are various separate tools such as Valgrind[34] and DynamoRIO[35] that strongly depend on the processor's instruction set architecture. Valgrind is an instrumentation framework for building

[34] https://valgrind.org/
[35] https://dynamorio.org/

dynamic analysis tools to detect things like memory management and threading bugs. Similarly, DynamoRIO is a runtime code manipulation system that supports code transformations on any part of a program while it executes. Typical use cases include program analysis and understanding, profiling, instrumentation, optimization, translation, etc. **For wide-spread acceptance of RISC-V in embedded systems, it is essential that such tools include support for RISC-V.**

Similar to hardware components, for safety-critical applications compilers must be qualified regarding functional safety standards. Here, the same challenges and requirements exist as for hardware components. For this reason, today mainly commercial compilers are used for safety-critical applications. These compilers are mostly closed source.

4.3 Debuggers

In order to efficiently analyze and debug code, debuggers are needed that are interoperable with chosen processor architectures as well as with custom extensions. Furthermore, debuggers should use standard open source interface protocols such as GDB (GNU Project Debugger) so that different targets such as silicon, FPGAs or Virtual Prototypes can be seamlessly connected.

4.4 Operating Systems

Chip design consists of many tradeoffs, and these trade-offs are best validated early by exercising the design by system software. For instance, when providing separated execution environments, it is a good idea to validate early that all relevant shared resources are separated efficiently. While this in principle should be easy it can be surprisingly difficult as highlighted by the Spectre/Meltdown vulnerabilities.

4.5 Real Time Operating System (RTOS)

Most of the common RTOS have already been ported to RISC-V (https://github.com/riscv/riscv-software-list#real-time-operating-systems), including the most popular open source options such as FreeRTOS and Zephyr RTOS. Even Arm's mbed has been ported to one RISC-V platform (https://github.com/GreenWaves-Technologies/mbed-gapuino-sensorboard), though mainline/wide support will most likely not happen. One of the key aspects of the OS is application portability. This can be achieved by implementing standard interfaces like POSIX which does not lock the software into a certain OS/Vendor ecosystem. There are many RTOS, but two key examples of relevance are:

Zephyr RTOS - Zephyr has been aligning with RISC-V as a default open source RTOS option. Currently RISC-V International itself is a member of the Zephyr project, along with NXP and open hardware providers Antmicro and SiFive. Zephyr is a modular, scalable RTOS, embracing not only open tooling and libraries, but also an open and transparent style of governance. One of the project's ambitions is to have parts of the system certified as secure (details of the certification process and scheme are not yet established). It is also easy to use, providing its own SDK based on GCC and POSIX support. The RISC-V port of Zephyr covers a range of CPUs and boards, both from the ASIC and FPGA worlds, along with 32 and 64-bit implementations. The port is supported by many entities including Antmicro in Europe.

Tock - **Tock provides support for RISC-V.** Implemented in Rust, it is especially interesting as it is designed with security in mind, providing language-based isolation of processes and modularity. One of the notable build targets of Tock is OpenTitan. Tock relies on an LLVM-based Rust toolchain.

4.6 Hypervisor

A hypervisor provides virtual platform(s) to one or more guest environments. These can be bare-bone applications up to full guest operating systems. **Hypervisors can be used to ensure strong separation between different guest environments for mixed criticality**, such platforms also have been called Multiple Independent Level of Safety and Security (MILS) or a separation kernel[36]. When a guest is a full operating system, then that guest already uses different privilege modes (such as user mode and supervisor mode). The hypervisor either modifies the guest operating system (paravirtualization) or provides full virtualization in "hypervisor" mode. **RISC-V is working on extensions for hypervisor mode, although these are not yet ratified.** Some hypervisors running on RISC-V are listed at https://github.com/riscv/riscv-software-list#hypervisors-and-related-tools.

At the moment, a number of hypervisors for critical embedded systems exist, provided by non-European companies such as Data61/General Dynamics, Green Hills, QNX and Wind River, and in Europe by fentiss, Hensoldt, Kernkonzept, Prove & Run, Siemens and SYSGO. Many of these are being ported/or could be ported to RISC-V. A weakness is that these hypervisors usually have to assume hardware correctness. An open RISC-V platform would offer the opportunity to build assurance arguments that cover the entire hardware/software stack. **Any new European RISC-V platform should be accompanied by a strong ecosystem of such hypervisors.**

Recommendation - If publicly funded research in the open hardware domain makes available some core results such as the used software/hardware primitives (such as HDL designs and assembly sequences using them) under permissive or weakly-reciprocal licenses, then these results can be used by both kinds of systems.

In the field of system software such as RTOS/hypervisors, currently there are several products which are closed source and have undergone certifications for safety (e.g., IEC 61508, ISO 26262, DO-178) and security (e.g., Common Criteria), and others which have not undergone certification and are open source. The existence of value chains as closed products on top of an open source ecosystem can be beneficial for the acceptance of the open source ecosystem and is common to many ecosystems. An example is the Linux ecosystem which is used for all kinds of closed source software as well.

4.7 NextGen OS for Large Scale Heterogeneous Machines

Recommendation - There is a need for research on RISC-V flexibility to revisit how to design together hardware and operating systems in order to better hide the heterogeneity of large machines to users, taking into account potential disruptive evolutions (non-volatile memory changing the memory hierarchy, direct object addressing instead of file systems, Data Processing Units to offload data management from processors).

To address the slowdown of Moore's law, current large-scale machines (e.g., cloud or HPC) aggregate together thousands of clusters of dozen of cores each (scale-out). **The openness of the RISC-V architecture provides multiple grades of heterogeneity, from specialized accelerators to dedicated ISA extensions, and several opportunities for scalability (e.g., large-scale cache coherency, chiplets, interconnects, etc.) to continue this trend (scale-in)**. However, manually managing the hardware-induced heterogeneity of the application software is complex and not maintainable.

[36] Tverdyshev, et al., MILS Architecture, 2013, EURO-MILS, http://dx.doi.org/10.5281/zenodo.45164.

4.8 Electronic Design Automation (EDA) Tools

Implementing a modern design flow requires a significant amount of EDA tools as shown in Figure 5 – see[37] which states **"With billions of transistors in a single chip, state-of-the-art EDA tools are indispensable to design competitive modern semiconductors"**.

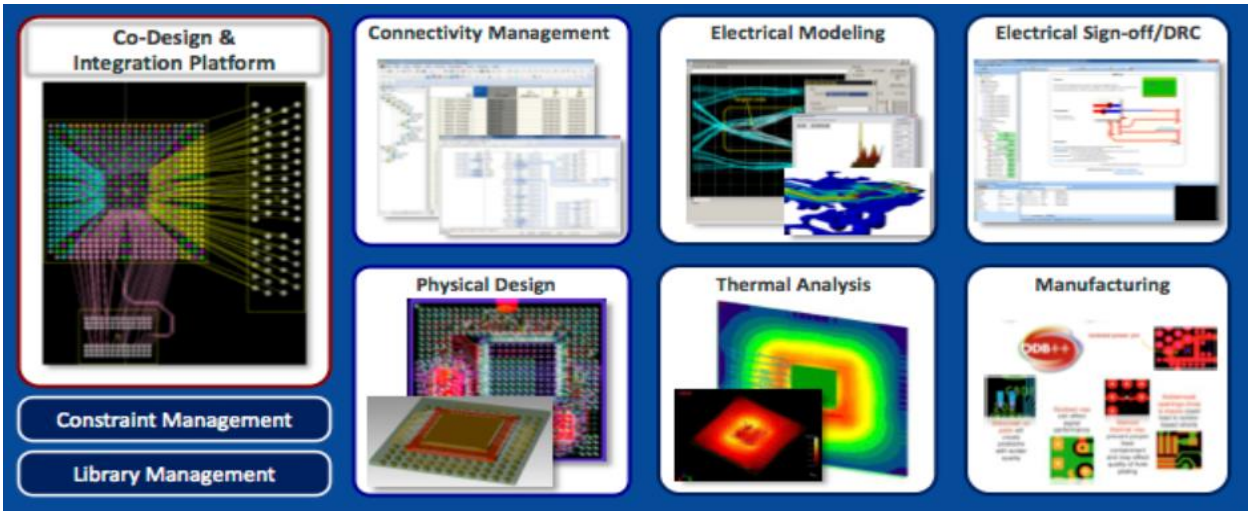


Figure 5 – Xpedition Package Integrator Suite (Source: Siemens)

Traditionally, EDA has been dominated by mostly US-based closed source commercial vendors. With projects like Verilator open source activities for specific parts of the design flow have also started to gain traction. However, open source support is still far from allowing a competitive fully open source design flow, especially when targeting digital design in advanced technologies, necessitating the co-existence of the existing commercial tools and upcoming open source ones for many years to come.

Open source tools are essential for introducing new companies and more developers into the field; especially developers with a software background who can bring in innovation in hardware-software co-design. Developers typically do not need to license their daily tools anymore and can freely work together across teams and organizations using massive collaboration hubs such as GitHub or GitLab. These benefits and capabilities need to be enabled via open source tooling for the sector to keep up with the demand for talent and innovation. A vital EDA community already exists in Europe with companies focusing on point solutions within the broader semiconductor flow. **Significant investment into open source tooling as well as cross-region collaboration is needed to energize the sector.** Contrary to common belief, the current EDA giants stand to benefit significantly from open source tooling investment, as there will be continued need for large, experienced players while open source solutions will enable new use cases and provide improvements in existing flows. New business opportunities will be created for the existing EDA players by incorporating new open source development. The recent acquisition of Mentor (one of the three leading EDA companies) by Siemens, means that it joins the

Recommendation - Current proprietary EDA tools can help open source design IP development get off the ground if they are available at sufficiently low cost and provided the tool licensing does not restrict the permissive open source usage of the developed design IPs, but a sustained and long term investment into open source tooling is needed to build a sustainable ecosystem.

[37] https://www.semiconductors.org/strengthening-the-global-semiconductor-supply-chain-in-an-uncertain-era/

European EDA community and can collaborate with the local ecosystem to support European sovereignty. The top EDA companies spend USD $1 Bn+ annually on R&D costs to continue innovating, so to provide meaningful progress in the open source space, continued investment from the public sector and cross-border collaboration are needed to bridge the gap. **Open source EDA should be a long-term goal in order to further the European sovereignty objectives, but existing European proprietary EDA will need to be utilized when necessary in the short to mid-term due to the significant investment that would be required to create a competitive, full flow open source EDA solution.**

Large semiconductor and system companies use their pool of proprietary EDA licenses to design, verify and get their open source-based designs ready for manufacturing with the expected productivity and yield. However, they may also be interested to introduce software-driven innovations into parts of the flows and can benefit greatly from the economies of scale of open source, enabling large teams to use the same tools free of charge. Smaller companies and research organizations need access to a comparable level of professional EDA tools which the large EDA vendors will most likely provide through a variety of business models (Cloud offering, specific terms for start-up, research licenses, Software as a Service (SaaS), etc.). The EC is continuing to invest to build a European open source EDA tooling ecosystem, encouraging open interchange formats and making sure current tools do not introduce restrictions on utilization on open source hardware designs independently from the open source hardware license used.

Considering the safety requirements of some of the IPs under discussion here, open source offers a unique possibility to create transparent, auditable processes for ensuring safety. Much like in the case of open source software, common procedures and pooling efforts through oversight bodies (such as the Zephyr RTOS or RISC-V Safety Committees) can be used to provide safety certificates or packages to reduce the burden of safety compliance off the shoulders of developers.

The key to enabling open source tooling in the EDA space (which will most likely also benefit existing, proprietary vendors) is in enabling specific components of the proprietary flows to be replaced by open source alternatives which can introduce point innovations and savings for the users. **This should be encouraged by focusing part of the EU investment on interoperability standards which could allow the mixing of open and closed ASIC tools**, much like the FPGA Interchange Format driven by CHIPS Alliance is doing for FPGAs. RISC-V is another example in the hardware space of how closed and open source building blocks can coexist in the same space and reinforce each other as long as there are common standards to adhere to. EDA tools are exploited in various parts of the development chain including lifecycle management, architecture exploration, design and implementation as well as verification and validation.

4.9 Lifecycle Management

A state-of-the-art development process comprising continuous integration and continuous delivery/deployment is at the core of typical projects. A key aspect is requirements traceability, both on the core and up to the system level to ensure that the verification of the requirements can be demonstrated. This is crucial for certification of safety (e.g., ISO 26262) where an IP can be used as a safety element by rigorously providing the requirements and then by ensuring that they are satisfied during integration. Another useful data point for users to decide if they should trust a particular configuration of an IP is the "proven in use" argument. This necessitates collecting data on which configurations of the IP has been taped out in given projects and the collection of related errata from the field. Here open source hardware provides the opportunity to propose better traceability metrics and methods than proprietary counterparts. Complete designs can be shared and even manufacturing

information on proper processes and good practices, as well as lifecycle management for open source hardware. This does not preclude the “out of spec” use of open source hardware in other domains such as low critical applications and education.

EDA tooling needs to support tracing the standardized verification metrics from IP level to system level. To successfully span the hierarchies from core to system level, a contract-based design is crucial, allowing to share interface contracts along the supply chain. This needs language and tooling to make it accessible to architects and designers. The open source toolchains, IPs and verification suites allow scaling the continuous integration/continuous deployment systems in server infrastructure reducing the build and test time. Proprietary solutions often require dedicated licensing infrastructure effectively preventing it from being used in scalable, distributed testing infrastructure. Different licensing and/or pricing of proprietary EDA tools for open source development may help to leverage the existing technology at the early stage in the open source development, before the open source alternatives are readily available.

4.10 Architecture Exploration

It is crucial to make the right architecture choices for the concrete application requirements before starting down a particular implementation choice. This requires tool support to profile at a high abstraction level of the application before software is developed. Usually, hardware is first modeled as a set of parameterized self-contained abstractions of CPU cores. Additional parameterized hardware components can then be added and software can be modeled in a very abstract level, e.g. a task graph and memory that get mapped to the hardware resources such as processing units. **Simulating executions on the resulting model allows analysis of the required parameters for the hardware components. This can include some first impressions on power, performance, area for given parameter sets, as well as first assessments of safety and security.** The derived parameter sets of the abstract hardware models can then be used to query the existing IP databases with the standardized metrics to identify matching candidates to reach the target systems goals.

The profiling can provide Power Performance Area requirements for instructions and suggest ways of potentially partitioning between accelerators and the relevant ISA instructions can be explored to meet these requirements. Ideally, a differentiated power analysis can already be started at this level, giving a rough split between different components like compute, storage, and communication. This can then be refined the more detail.

A widely used methodology is Model-Based System Engineering (MBSE) which focuses on using modeling from requirements through analysis, design, and verification. For the application of processor cores, this would mean the use of Domain Specific Languages (DSLs) down to the ISA level to capture the hardware/software breakdown. The breakdown would then be profiled before starting implementation of any design or custom instructions on top of a core. The modeling should lend itself for use in High-Level Synthesis (HLS) flows.

Recently, thanks to the popularization of open source hardware (which pushes more software engineers into the hardware industry) as well as a fast-moving landscape of modern AI software which in turn requires new hardware approaches, more software-driven architecture exploration methods are being pursued. For example, the Custom Function Unit (CFU) group within RISC-V International is exploring various acceleration methods and tradeoffs between hardware and software.

Google and Antmicro are developing a project called CFU Playground[38] which allows users to explore various co-simulation strategies tightly integrated with the TensorFlow Lite ML framework to prototype new Machine Learning accelerators using Renode and Verilator.

4.11 Design & Implementation

Architecture exploration leads to a generic parameterized model. This requires a modeling language that offers sufficient expressiveness and ease of use for wide acceptance. A challenge is that typically engineers engaged in the architecture exploration process do not come from a hardware background and are unfamiliar with SystemC or SystemVerilog, the established hardware languages. Modern programming languages like Python or Scala are thus gaining traction as they are more widely understood. An architecture exploration process requires the availability of models and an easy way to build a virtual platform and exchange models by implementations as soon as they become available. This procedure enables early HW/SW Co-Design in a seamless and consistent manner.

A traditional Register-Transfer Level development process manually derives RTL from a specification document. To address the large scope of parameterized IPs targeted in this initiative, more automation is required. **The parameterized Instruction Set Architecture (ISA) models for processors lend themselves for a high-level synthesis (HLS) flow where detailed pipeline expertise is not required by the users unless they require the highest performance. This is crucial for enabling a wider audience to design and verify processors.** The design process thus lends itself to a high degree of automation, namely offering some form of HLS starting from the parameterized models. For processors, for example, synthesizing the instruction behavior from an ISA model and thus generating the RTL is feasible for a certain class of processors. The design process should also generate system documentation in terms of:

- System memory map
- Register files of each component
- Top level block diagram
- Interconnection description

The documentation needs to be created in both user and machine-readable forms allowing manual and automated verification.

Recommendation - There is a need for a high degree of automation for adding various monitors in the design flow without having to manually pick a suitable monitor IP or even to design, configure and wire it up from scratch, which can be very error prone.

RTL development flows should rely on continuous integration processes, such as automated checking via linting and Design Rule Checks (DRC). There is a possibility for RISC-V specific linting or DRC. Similar checks can be applied down the flow after synthesis to catch simple mistakes which often result in huge consequences.

It is important to add some monitoring components in the design phase, among other things to address threats that cannot be fully verified during development or that arise because of hardware/software interaction. In addition to an early security threat warning, monitoring is also needed for the safety. Design integrity and performance should also be monitored. While RISC-V offers a generic performance counter mechanism, not much has been standardized, creating an overhead to get specific counters like cache statistics integrated into the toolchain. Common C++ libraries for accessing the counters should be

[38] https://github.com/google/CFU-Playground

available. Debug logic with a more active role is needed to replace today's mostly passive monitors. Heterogeneous cores, each with their individual debug features, creates additional software challenges. The generated monitoring and debug data can be evaluated with in-system software or off-loaded from the chip. **In order to evaluate the comprehensive monitor and debug data from such a SoC new analytics and visualization tools will need to be developed. There is also a need for system level and context specific debug tailored to applications like automotive, 5G, HPC.**

4.12 Verification & Validation

Verification management needs to be tightly integrated with lifecycle management for traceability from requirements to verification. **Verification should benefit as much as possible from the models produced by architecture exploration creating golden models for verification with coverage goals.** At the system level there are hard challenges like cache coherency and there is a need for portability of tests across different levels. State-of-the art verification approaches use simulation based and formal approaches. The introduction of automated formal verification has made the approach available to typical design and verification engineers allowing them to set up the targeted checks of the application much faster than they would in a simulation-based flow. **The approaches are complementary, lowering the overall verification effort with formal verification applications, while at the same time increasing verification quality in the crucial areas with formal proofs compared to an incomplete simulation.** To further enable a broad user base for custom processors, the verification side of the flow also needs a high degree of automation, for example deriving large parts of the verification from the parameterized models also used on the HLS side to create the implementations. For safety related projects, state-of-the-art tools and methodologies must be used driving the use of formal verification.

Cores and IPs should come with a reference flow similar to what is provided by Arm, allowing to re-run the provided verification in various tools, be it proprietary or open source. The CHIPS Alliance is developing an open source RISC-V core code generator/verification framework called RISC-V DV[39] which the OpenHW Group[40] is also using as the base of a simulation-based environment for verification of their cores. The CHIPS Alliance is working towards a fully open source Universal Verification Methodology verification flow based on Verilator[41]. However, the OpenHW environment still needs a proprietary simulator to run the full verification. Only a small subset of the verification suite can be run on the open source EDA tool. **For open source design IPs with industrial strength verification, there is no short or mid-term availability of an open source EDA tool suite that provides simulation-based and formal verification. Achieving an industrial strength verification with open source EDA verification tools is a long term goal.**

Another example of open source verification work can be found in the OpenTitan project[42] which provides a valuable example of an open source, continuous delivery system whose coverage and status can be traced for every commit. Software-based frameworks like cocotb are also gaining traction, especially with engineers and teams with

Recommendation - For open source EDA tools, establishing collaboration models realizing a sophisticated design and verification process is an important topic. Maintaining an EDA tool for safety-critical applications after an initial release also requires a significant amount of both manpower and computing resources to ensure persistent tool quality and up to date safety collaterals.

[39] https://github.com/google/riscv-dv
[40] https://github.com/openhwgroup/core-v-verif
[41] https://antmicro.com/blog/2021/07/open-source-systemverilog-tools-in-asic-design/#uvm-is-in-the-picture
[42] https://docs.opentitan.org/hw/

a software background, and while they are incompatible with traditional Universal Verification Methodology (UVM) style verification, many new open source IP implementations adopt them. Hardware description languages based on modern programming languages like migen/nMigen (Python based), CHISEL, SpinalHDL (Scala based) provide their own simulation and verification flows. Since the above languages are derived from modern programming languages, they can easily reuse testing methodologies known from the software world.

As an example Antmicro's Renode while used mainly for software development and testing, provides a means to create complex simulation environments based on the Hardware Description Language (HDL) code of both cores and peripheral IPs. This creates an easy way to test IP in complex software scenarios, instead of synthetic, hand-crafted tests. In addition, it gives the possibility to work on software development in the pre-silicon phase of an ASIC project reducing the overall time-to-market.

Recommendation - For the standard protocols used to connect the design IPs, the use of Verification IP (VIP) is strongly encouraged. VIP allows all protocol rules on the controller or peripheral sides to be checked, thus ensuring IP blocks from different vendors can properly communicate. Formal VIP is another typical application where formal verification is easy to setup and gives great verification results. A focus should be on the protocols used in the design IPs – be it the AMBA protocols or open source ones like OBI, OCP, or TileLink.

4.13 Tool Qualification for Safety-Critical Applications

In the automotive industry, a classification according to ISO 26262 is necessary for all design and verification tools that should be used in a safety-critical environment. If the classification shows that the tool could introduce errors into the design and these errors would not be detected by another step in the design flow, a tool qualification is needed.

This leads to a significant challenge for the quality of the tools. By having a sophisticated development process handling requirements traceability, change management and comprehensive documentation, the work on the user side to qualify a tool can be significantly reduced. Furthermore, proof of comprehensive test suites and therefore the complete verification of the tool is necessary for a tool verification.

5 CROSS CUTTING REQUIREMENTS

5.1 Scalability and Enhanced Instruction Sets

Instruction sets have traditionally been managed by a single "owner" (Intel, Arm) and their evolutions over time have been slow, lagging application requirements. This is because the ISA-owners are not strongly motivated to modify or overhaul their ISA and they hesitate to invest the required R&D and engineering effort, typically engaging only under extremely heavy customer requests. This situation has completely changed with the advent of the RISC-V open ISA, mostly for two reasons:

(i) from a technical viewpoint the RISC-V ISA is designed to be modular and extensible, with adequate provisions for ensuring backward compatibility issues

(ii) R&D and commercial efforts to extend the RISC-V ISA for specific application domains can be initiated as community efforts, with cost and risk sharing and can be also used to provide differentiated value-added solutions.

As a consequence of this paradigm shift toward open ISAs enabled by RISC-V, major innovation opportunities are enabled on open cores with enhanced instruction sets. The faster innovation cycle which is now possible by coupling ISA and core enhancements in an open source setting, has been demonstrated in several domains. Notable examples include: open extensions for supporting quantized computations in machine learning, with particular emphasis on deep neural network inference, general digital signal processing extensions (e.g. Single Instruction Multiple Data (SIMD), fixed-precision arithmetic), and security extensions (e.g. Galois field operations, bitwise operators). It is also important to note that extensions that have been successfully prototyped as non-standard ISA enhancements and have gathered wide adoptions, can then be moved toward new standardized parts of the ISA.

Scalability is also another important opportunity created by the RISC-V instruction set architecture. RISC-V does not prescribe a limit to instruction encoding size. In addition to the already standardized 16-bit, 32-bit and 64-bit instructions, it is also possible to specify intermediate (e.g., 48-bit) and even larger instructions sizes: 128-bit instructions are already being worked on in the RISC-V standardization committees.

5.2 Safety Certification – Open Standards; Safety-Ready Toolchains

RISC-V is promising for applications in the high-assurance market due to potential cost reductions from easier access to innovation, flexible and rapid design processes, stability and modularity, and availability as white box[43]. When considering safety there is a need to consider both the hardware component level and also the system level taking into account the interplay between hardware and software. At the hardware level the safety-critical hardware components should ideally be developed to sectoral safety standards. For example, in the automotive sector the key standard is the ISO 26262 standard which contains development process requirements in order to avoid systematic and random faults. These address management processes during the development lifecycle, role definitions, hazard

[43] H. Legenvre, P. Kauttu, M. Bos, and R. Khawand, "Is Open Hardware Worthwhile? Learning from Thales' Experience with RISC-V," Research-Technology Management, vol. 63, no. 4, pp. 44–53, Jul. 2020, doi:10.1080/08956308.2020.1762445; Blind, Knut, Mirko Böhm, Paula Grzegorzewska, Andrew Katz, Sachiko Muto, Sivan Pätsch, and Torben Schubert. "The Impact of Open Source Software and Hardware on Technological Independence, Competitiveness and Innovation in the EU Economy," 2021, https://ec.europa.eu/newsroom/dae/redirection/document/79021

analysis, risk assessment and development processes. These include conditions for requirements tracing, confidence levels of tools used for development and verification and verification requirements. Certifying a system that contains hardware components that are not developed according to the ISO 26262 is still possible but elaborate.

Recommendation - In order to make the use of open source hardware components possible in automotive applications, artifacts and methods have to be defined that help with certifying these components. A start could be the implementation of quality-management systems in open source projects.

In addition to the open source availability of source code, the open source availability of verification artefacts (verification plans, test benches, reference simulators, assertions, test sequences...) can be enablers of white-box analyses and certification processes. Notably the OpenHW Group already publishes these verification artefacts[44], however, available documentation of the verification process itself still might not be sufficient especially for the highest assurance levels (ISO 26262 ASIL D, DO254 DAL A...) where additional, and often very costly, verification practices might be needed.

EDA tools can be used for implementing and assessing the safety of hardware components. Automation plays a key role in order to avoid the need for RTL coding and development of verification strategies in the otherwise largely automated process. Similarly, tool support for adding and validating security mechanisms is needed. It is important to be able to assess the impact of safety or security mechanisms as much as possible based on models used as input to the design phase, i.e., before the design is completely done. In addition to assessing single safety/security mechanisms, the assessment at the system level is also crucial, including the fulfilling of critical timing deadlines in real-time safe systems. A key challenge is the huge parameter space of typical IPs as validation is needed for each and every concrete parametrization that is used. **For safety in particular, support for specific certification flows like Failure Modes, Effect and Diagnostic Analysis (FMEDA), is of huge value for integrators.**

At the system level a key issue is to guarantee Worse Case Execution Time (WCET) deadlines for the execution of critical tasks. For any time-critical system, requirements specify maximum response times. An issue is that most hardware manuals now do no not publish steps/cycles/execution times making *a-priori* determination of execution time difficult. In the past this information was available, e.g., Intel 486/Pentium manuals.[45] However, newer processor (Intel, PowerPC, Arm) manuals do not provide such numbers for multiprocessors like the P4080.[46] A workaround is to replace clock cycle analyses by own empirical time measurements which suffices for average case behavior but is inefficient and/or unsafe for assessing worst-case execution time.

[44] https://core-v-verif-verification-strategy.readthedocs.io/en/master/

[45] Wilhelm, Reinhard. "Determining Reliable and Precise Execution Time Bounds of Real-Time Software." IT Professional 22, no. 3 (May 1, 2020): 64–69. https://doi.org/10.1109/MITP.2020.2972138.

[46] Kästner, Daniel, Markus Pister, Simon Wegener, and Christian Ferdinand. "Obtaining Worst-Case Execution Time Bounds on Modern Microprocessors." Nuremberg, 2018. https://www.absint.com/papers/2018_ew_tw.pdf; Agirre, Irune, Jaume Abella, Mikel Azkarate-Askasua, and Francisco J. Cazorla. "On the Tailoring of CAST-32A Certification Guidance to Real COTS Multicore Architectures." In 2017 12th IEEE International Symposium on Industrial Embedded Systems (SIES), 1–8. Toulouse: IEEE, 2017. https://doi.org/10.1109/SIES.2017.7993376.

5.3 Security

Common Criteria security certifications for simple hardware IP such as smart cards[47] exist, however, for more complex processors, security is still in its infancy and it is not possible to buy a Common Criteria certified general purpose multicore processor. To get around this currently companies have to make liability limiting statements such as "The underlying hardware […] is working correctly and has no undocumented or unintended security critical side effect on the functions of ..." However, relying on the hardware's documented interface alone can be insufficient, as there may be parts of the hardware-software system architecture that cannot be inspected by the software developer. For instance, **the Spectre/Meltdown vulnerabilities which appeared in 2018 were unexpected for almost all OS vendors.**

In an exhaustive search of hardware-software systems (3800 publicly available security targets and evaluation reports available from **commoncriteriaportal.org**) it is noted that the focus is on describing hardware at a high level without going into the micro-architectural side effects. Often unjustified assumptions on hardware are made and no references were found to key issues such as "branch predict*", "translation lookaside buffer", or "branch history buffer" with very few mentions of "cache flush". Evaluation reports do, however, exist on the analysis of the security properties for public-key cryptography including both VHDL and software analysis. A recent effort on the security side is Accellera's SA-EDI (Security Annotation for Electronic Design Integration) that establishes a link to CWE (Common Weakness Enumeration). **A continuing challenge is long-term security as there is a need to continually innovate to guard against future new attacks.**

5.4 The Way Forward - Hardware-Software Contracts for Safety and Security

Safety and security properties need to be known and can be summarized in the concept of hardware-software codesign with contracts as shown in Figure 6. **In this open hardware allows development of hardware-aware system software and system software-aware hardware.**

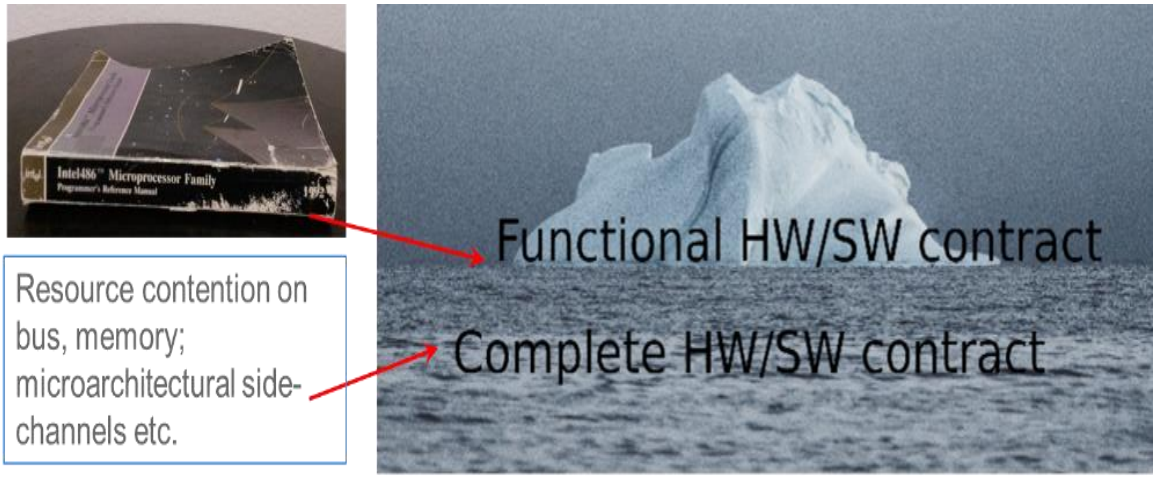


Figure *6* – Hardware/Software Contracts

[47] https://www.sogis.eu/documents/cc/domains/sc/JIL-Composite-product-evaluation-for-Smart-Cards-and-similar-devices-v1.5.1.pdf

In this approach rely-guarantee relations are made between hardware and software called "hardware-software contracts" [48]. Here there is a key need to bring together hardware providers and software providers to develop and produce a general-purpose open hardware platform or at least core components with access to the full hardware-software contract, allowing safety and security certification. This should be based on an instruction set architecture used in an existing or upcoming ecosystem, such as RISC-V. At the same time a verification methodology that provides reasonable assurance for the platform/its core components is needed. These verification artefacts need to be published to allow them to be used in use cases.

Recommendation - For supporting safety and security there is a need to demonstrate and share verification environments and artifacts of open source hardware at medium assurance levels and also at higher Safety Integrity Levels (SIL), e.g. safety SIL 4/ASIL D, security Common Criteria EAL 6 and higher, firstly addressing simple systems and then moving to complex systems.

[48] Ge, Qian, Yuval Yarom, and Gernot Heiser. "No Security Without Time Protection: We Need a New Hardware-Software Contract." Proc. 9th Asia-Pacific Workshop on Systems - APSys, 2018. https://doi.org/10.1145/3265723.3265724.

6 IDENTIFIED GAPS AND FUTURE NEEDS

The European economy consists of multiple industrial application domains which will be impacted by RISC-V and Open Source in different ways. These application domains include automotive, industrial manufacturing, consumer electronics, telecommunications, health, home applications, aerospace and defense. Within these domains there are some common requirements, and also specific challenges, such as security, safety, privacy and support for artificial intelligence. **Although these application domains can make use of generic common processor families, in many cases domain-specific processor families or specific features are needed to meet performance, reliability, cost, and energy consumption requirements.** In this section a number of common, as well as application/domain-specific gaps and needs are identified through the process chain as shown in Figure 7.

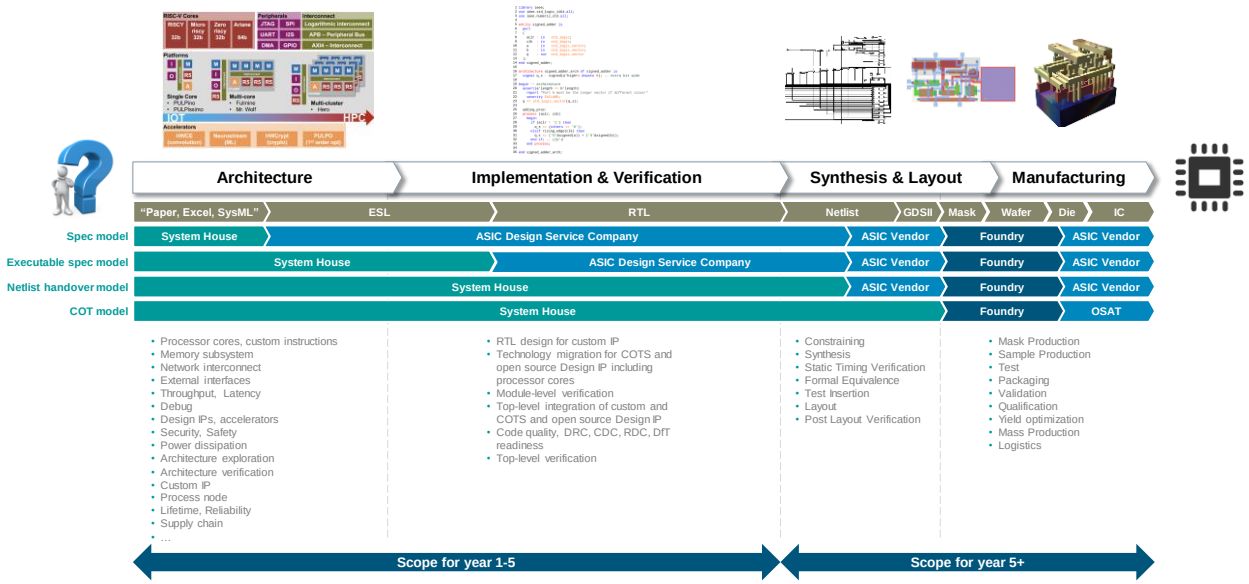


Figure 7 - From Architecture to Manufacturing

6.1 Processor Design

On the processor side, **European technology already is well established for low-end processors**, which could be considered micro-controller-class cores. Far fewer efforts deal with mid-end cores, which are sometimes called application cores. **The lack of European sovereignty is most pronounced, though, at the high-end, which includes server-grade and HPC cores**. In this area, there is a complete absence of European open source offerings.

6.1.1 Core Microarchitecture

To close these gaps, there is a need to strengthen efforts for both the mid-end and the high-end. In the first case it will be possible to build/improve upon existing open source offerings. For high-end processors, work will have to start mostly from scratch, as only very limited prior work is available even internationally. For instance, the BOOM processor initially developed at UC Berkeley, is still undergoing heavy revisions (e.g., from v2 to v3) and does not appear to be a stable base for future development yet. For the open source Alibaba Xuantie 910 core key documentation is only available in Chinese. A European effort should consider the results of these prior works but will most likely be starting (almost) from scratch for a European implementation. This is especially true when the high-end

core should be evolvable towards advanced features such as multi-threading, smart prefetching, multi/many cores. Furthermore, the verification of the complex out-of-order superscalar microarchitectures required for high-end processors is considerably more difficult than the simpler in-order ones in the low-end and mid-end cores. Since Europe has a strong research background in formal verification techniques, the design of a high-end core could be a key opportunity for applying that expertise to practice. For instance, a result of formal verification techniques could be proofs of the absence of side channels.

6.1.2 Hardware Peripherals and Interfaces

In most cases, the processor cores described in the previous section will not operate completely stand-alone but will be integrated in a more complex system having additional components both on-chip and off-chip. Interaction between some of the components requires complex and challenging interfaces that are so-far mainly available as proprietary solutions from commercial vendors (such as physical interfaces to high-speed links or memories). More complex use-cases include the communication between core(s), between cores and accelerator(s), and also to memories (unified memory approach), especially when coherency between distributed local caches has to be covered as well. This problem becomes even more complicated when also considering safety-critical or mixed-criticality aspects in the solution. **Depending on the communication scenario, e.g., number of cores and accelerators, throughput and latency requirements, it may become necessary to upgrade from mostly passive interconnects (matrices of multiplexer-based switches) to active structures such as networks-on-chip (NoCs), which can scale better with more communication partners.** Open source baselines already exist for some of these requirements, e.g., cache coherency between a relatively small number of nodes. However, these do not scale to the more complex accelerator-intensive systems becoming more common today, and do not address the safety-critical or mixed-criticality aspects at all. There is a need for a flexible open source infrastructure for these on-chip communications, which should be easily customizable for the specific functional and non-functional requirements of a concrete SoC. In addition to the physical IPs there is also a need for tools to help in the design and the dimensioning of the infrastructure, the simulation, validation and verification views.

The complexity of off-chip interfaces spans an extremely wide spectrum, mostly depending upon the desired bandwidth. For low-speed interfaces, such as UART, SPI, I2C, I2S, etc., a wide variety of open source blocks exist. However, for higher speed (and more complex) interfaces such as USB3, 1G Ethernet, (LP)DDR2+ dynamic memories, and PCI Express, the design challenge increasingly moves from the digital into the analog domain. These interfaces require not only increasingly complex digital logic controllers (memory, PCI Express), but also high-speed off-chip communications using the appropriate I/O logic and so-called fast PHY/SerDes interfaces to achieve the required signaling rates. In general, the design of the PHY/SerDes blocks requires knowledge of low-level electrical characteristics of the underlying chip fabrication technology that is often only available under strict Non-Disclosure Agreements (NDA).

Many relevant system-on-chips that go beyond simple microcontrollers require these high-speed interfaces, e.g., to work with AI/ML datasets exceeding the (at most) single-digit megabytes of static memory storage that can economically be supported on-chip, or to communicate with other peripherals attached via PCI Express (e.g., to mass storage), or with other systems over the network. The lack of open source blocks for the tasks makes it extremely challenging for innovators to prototype solutions, e.g., to attract further investment, as **a commercial license for a DRAM interface (controller and PHYs,**

commonly provided by an extremely small number of non-European companies) may well cost over half a million USD.

Recommendation – Effort should focus on areas where there is a current lack of technologies and tools to design and implement modern system-on-chips in order to strengthen European Digital Sovereignty.

To strengthen European Digital Sovereignty, it is essential that this scarcity of key technologies essential for the design and implementation of modern system-on-chips is alleviated. However, due to the confidentiality issue discussed above, realizing these high-speed mixed-signal peripheral interfaces blocks under an open source model is more complex than for the purely digital blocks (e.g., processor cores, many accelerators). **A potential approach could be to design the interface as much as possible in a portable, fabrication-process independent manner.** Only for those parts where it is unavoidable would a concrete ASIC process design be used, ideally one with fabrication facilities within Europe. The process-independent parts could then be customizable and freely open sourced under any suitable license. The process-specific parts, though, will require another approach, e.g., a multi-way NDA between the potential user, the foundry, and the designer of the interface IP. For any project involving such a mix of open and proprietary technologies, concrete work packages and tasks should be allocated to investigate and/or propose licensing strategies/legal frameworks for dealing with this scenario.

In addition to the benefits of making the functionality of high-speed interfaces available to innovators in an open source manner, there is also a secondary benefit that due to their open nature, it is possible to *analyze and verify* open source peripherals and interfaces in far greater depth than would be possible with closed-source offerings. This is required as the communication protocol is often quite complex making it difficult to ensure that the IP is fully compliant with specifications or standards. This can also enable advances in the areas of security and safety-critical systems, e.g., where interference effects such as bus contention, or side-channels between applications, need to be closely monitored. This especially applies to architectures *mixing* different criticality levels on the same hardware. For instance, using the design techniques described above (highly focused on customizability), it should be possible to parametrize an interconnect IP block so that selected communications between specific execution environments are *guaranteed* to be performed in an interference-free manner.

6.1.3 Non-Functional Requirements

In addition to the design/improvement of the actual cores there is also the need to fulfill two key non-functional requirements: **As many of the European users of these cores come from industry (e.g., automotive, Industry 4.0, etc.) where *safety/security/reliability* and the associated certifications are often more important than the performance or power efficiency that would be focus of purely data-center oriented cores. Second, due to the domain-specific needs of these industries, there is a requirement to easily *customize the operation* of the core for applications, e.g., by adding specialized instructions or compute accelerators for operations in AI/ML, cryptography, signal and image processing, custom arithmetic, communications and storage, etc.** It is this need for flexibility with regard to safety/security/reliability and extensibility with custom operations that will be a key differentiator between European core design efforts and the more focused ones (e.g., for data centers, mobile communications, etc.) from existing players. Ideally, by employing state-of-the-art design methodologies leveraging, e.g., new hardware construction languages such as Chisel, Spinal, Bluespec, it should be possible to design the new cores as highly parametrizable base designs to/from which new features can easily be added/omitted as required for an industrial use-case. Note that these hardware construction languages are all open source and based on underlying languages such as Scala and Haskell,

which historically have a strong European involvement. However, to fully unlock their productivity gains for industrial-strength use, verification support for these new languages needs to be significantly improved.

6.2 Software Tools

On the software side, the existing baselines are much more mature than on the hardware side. The huge open source software ecosystem, grown around efforts such as GCC, LLVM, and Linux, has advanced the state-of-the-art considerably over the last 30 years. However, to fully exploit the open RISC-V ISA and the capabilities of the customizable processor cores described above, there are two areas where there is a need for key improvements. The first one is to support the customizability of the new cores also in the associated software. For instance, automatically integrating custom instructions and accelerators into the compilers and operating systems/middleware/hardware abstraction layers. **Without this kind of support, the capabilities of the newly designed hardware will remain inaccessible to most software developers**. The second area plays again to the traditional European strength in safety/security-critical systems. Again, to exploit safety/security-hardened cores the related software must also be extended, e.g., to be suitable for certification under standards such as DO254/DO178 or ISO 26262. This effort can also profit from new technologies, e.g., the combination of processor trace capabilities with dynamic verification techniques that, together, can monitor/enforce safety and security constraints in a *running* system.

6.3 Electronic Design Automation Tools

Electronic Design Automation tools are required to create new hardware. **Central to any hardware design targeting an integrated circuit are the ASIC front-end and back-end implementation tools.** At the front-end, these tools take a description of the function or structure of the electronic circuit, for digital logic circuits such as the RISC-V cores (most commonly in hardware design languages such as Verilog and VHDL), or the more modern hardware construction languages and translate it into more basic building blocks (e.g., logic gates and state elements) by *logic synthesis*. In the next step, these basic building blocks are *mapped* to the available hardware blocks in a given chip fabrication process (e.g., standard cells from a library). The geometrical arrangement of these cells is then determined by *placing* them on a 2-D plane (or in the future: a 3-D volume) and computing the best way to establish the required connections using a *routing* algorithm. Today, most of this software pipeline relies on proprietary tools developed and sold by a very small number of vendors, mostly from the US. **To strengthen European digital sovereignty in this area, it is crucial that open source solutions are created as an alternative.** Fortunately, initial open source efforts such as the OpenROAD project have already sprung up, sometimes utilizing European-developed tools for core functionality, such as the *logic synthesis* and *mapping* steps. These existing efforts could be used to bootstrap advances that would increase robustness of the tools, improve the quality of results by the integration of newer algorithms (e.g., AI/ML-based), and allow fundamental innovation on better tool support for the topics of specific interest to the European microelectronics industry, such as safety/security/reliability. **Additionally, the *practical usability* of such an open source ASIC implementation tool flow should be strengthened, e.g., by ensuring its full interoperability with the process technologies supported at chip manufacturing facilities located in Europe.**

6.3.1 Supplemental System-Level Tools

With modern systems-on-chip becoming ever more complex, there is a growing need for additional steps in the EDA tools workflow *supplementing* the ASIC implementation flow. These include, e.g., the

ability to more abstractly describe the functions of a digital circuit at the purely behavioral level, and then employ a process called *high-level synthesis* to compile this behavioral description down to the traditional descriptions discussed above. Open source tools for high-level synthesis already exist, however, the approach has proven useful so far mainly for very specific use-cases, such as quickly creating specialized accelerators, but not so much for creating high-quality general-purpose processor cores. Thus, a greater focus should be applied to tools enabling more immediate practical benefits to a larger group of users. Two examples include *architecture exploration/optimization* as well as automated *system-on-chip composition*.

Architecture Exploration – **Architecture exploration takes place prior to chip design to determine the best hardware architecture for achieving the design goals**. It employs a number of techniques, such as virtual platforms and software frameworks to model application workloads, to quickly iterate on different hardware architecture choices before committing to a single one which is taken forward to the laborious and costly ASIC implementation and fabrication process. It is also required in the case of a parametrizable design to determine which parameters are best suited for the final design. Despite the importance of this step, as the architectural choices made for an actual chip implementation can have far-reaching consequences for the success of the entire endeavor, only very limited tool support, mainly from a small number of vendors of proprietary software, is currently available. Thus, it would be highly beneficial if a flexible tool framework for architecture exploration could be created in an open source fashion. The open source nature of the framework would then allow innovators to focus on key areas, e.g., the use of AI/ML techniques to optimize entire systems and help the designer to define the best system architecture, as well as alleviate the need to expend R&D effort on laborious engineering tasks, such as visualization, simulator interfaces, etc.

Automated System-on-Chip Composition – There is also a need for support for *automated system-on-chip composition*. Modern SoCs are, to a large degree, created by composing building *blocks* (also called IP blocks), where a block may be, e.g., a complete processor core, cache level, or memory controller. Although standards exist for the machine-readable description of *individual* blocks (e.g., IP-XACT), composing them into an entire system-on-chip encompassing the required hardware/software interfaces, is still mainly performed manually, at best aided by GUIs or low-level generic scripting languages such as Tcl. There is a need for more abstract descriptions of complete SoCs, above the view offered by formats such as IP-XACT. **A new class of EDA tool is needed that can interpret these more abstract descriptions to allow complete system-on-chips to be composed, including optimized design choices, e.g., for interconnect and bus protocol optimization.** Such an SoC composition tool could also be integrated with the architecture exploration flow discussed in the prior paragraph as one step of an automated design-space exploration mechanism. As before, having the SoC composition tool available as open source would enable both more innovation in the operation of the tool itself, but also allow its easier integration with other automation tools, such as architecture exploration tools. In this way, innovators in the hardware space could concentrate on their specific unique contributions (e.g., new AI/ML or cryptography accelerators), and expend less valuable engineering resources (especially for startups) on non-innovative, but complex engineering tasks such as manual SoC composition.

6.3.2 Verification

An overarching topic across all of the different hardware design and implementation steps already discussed is verification. This occurs at all levels, e.g., from architecture specifications, such as the formulation of custom instructions of a processor, down to the actual physical-level silicon design. **One of the crucial aspects for the success of design IPs is the completeness of specifications, validation of**

the architecture and verification quality of the resulting IP. Efficient design exploration tools and simulators allow to identify and specify the best architecture parameters for an IP (some tools even allow to generate VHDL or Verilog code for particular specific domains). The earliest that bugs or deviations from the specifications are identified, the less expensive they are to correct. State of the art industrial strength functional verification is required for ASIC tape-outs, even more so in safety critical applications mandated by standards like ISO 26262. Such industrial strength functional verification cannot be achieved by constrained random simulation alone but requires formal verification and emulation in the tool portfolio. A tool agnostic verification flow spanning these three elements is the basis for achieving industrial strength verification. This flow should be compliant with safety standards to allow using the results for safety certification. Such a complete verification includes a safety certification-ready verification plan, shareable/reusable requirements and artifacts, requirements tracing through to the verification plan and the tests verifying them.

Verification of the embedded software stacks using virtual platforms and hybrid platforms (mixing high-level models, e.g., SystemC and RTL) in the system context, if it comes after RTL design & verification is too late. **Tools enabling this task to be performed much earlier in the design cycle are thus essential to cut design & verification time and cost**.

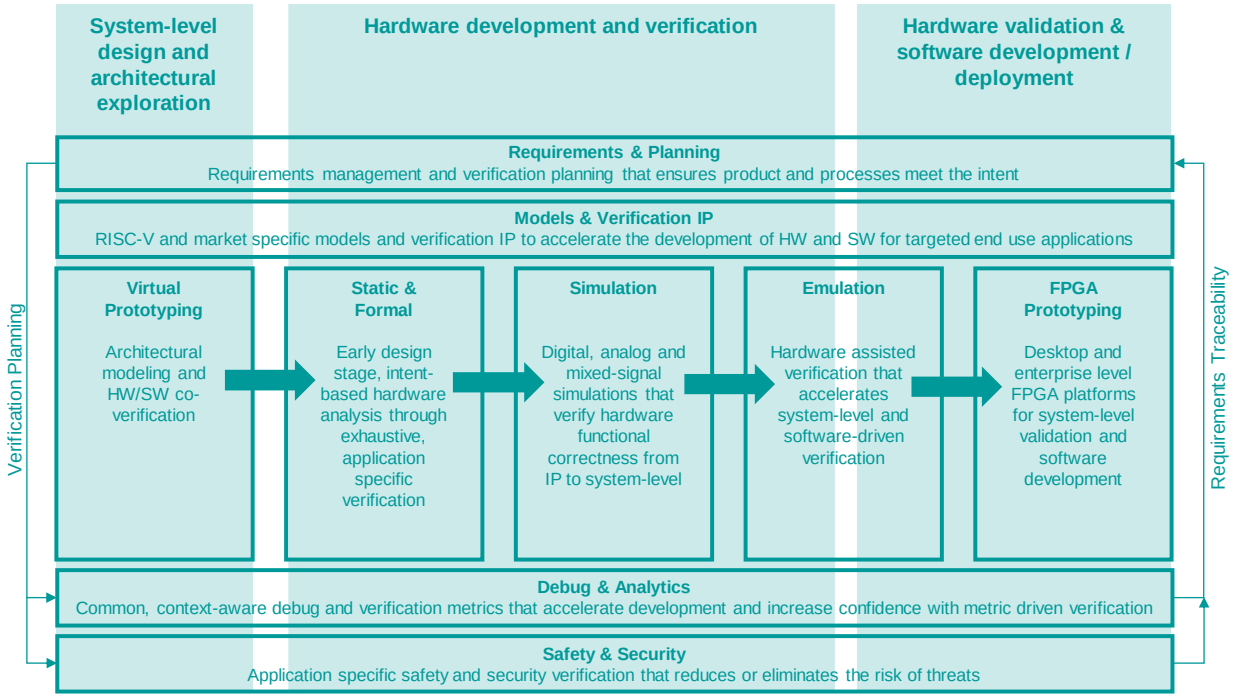


Figure *8* - V Model at System and Hardware Level

State of the art IP design also uses tools for linting or design rule check (DRC), clock domain crossing (CDC), reset domain crossing (RDC), X propagation and formal auto-checks. Additional tools may also be mandated for achieving compliance with downstream flows. **In order to achieve highest RTL verification quality in a manageable amount of time, automation is key.** The verification flow needs to be as automated as possible, from continuous integration (CI) to, for example, formal verification apps targeted at certain aspects of the design IP. **In the context of design IPs around RISC-V cores and other subsystem components, an automated formal verification is needed for those parts that are complex and hard to verify.** At the core, there is a need to verify a parameterized RISC-V core with custom instructions against its expected ISA. This requires checks that all instructions and registers behave

according to the ISA and no other instructions or registers exist that could offer backdoors for malicious attackers. Beyond the mere core, other areas of interest for an automated formal app are caches, interconnect, prefetch, or consistency for memory systems with several hardware threads or cores.

In addition to functional verification, security topics are starting to play a more and more crucial role. Unexpected discovery of attacks like Spectre or Meltdown on state-of-the-art CPUs highlighted a verification gap that calls for a formal approach that can actually find all kinds of such attacks instead of patching designs after the fact whenever a new attack is found[49]. Design IPs have additional security requirements that must be part of the verification flow, including traceability. Verification goals may also apply for security, resulting in a set of security checks applicable to an IP. This is again a domain where automated formal apps can offer great benefits. In terms of standards, the field of security is less mature than safety. **Nevertheless, the verification flow should also allow integration with standards like Accellera's Security Annotation for Electronic Design Integration (SA-EDI) and offer the corresponding automation.**

On top of the RTL, there is also system software that has to interact correctly with the design in order for the system to work properly with respect to safety and security. **Here, there is a need to focus on boot code and firmware.** Both are critical components with large security impact, and both are very close to the hardware. Again, formal based apps could be used to investigate these software parts in conjunction with the hardware. Formal apps can identify unexpected sources that can influence the boot process or spot parts of firmware code violating the programming rules.

6.4 Identified Gaps Summary

Conception Stage

- There is a need for tools allowing fast simulation and exploration of the design space configuration, and automated system-on-chip composition. Commercial tools exist already as well as open source ones, but they can be improved using new approaches such as ML/AI.
- The community would benefit from a repository of open source models and new technologies that can be used in both open source and commercial tools. New technologies, for example using Artificial Intelligence, can be specifically developed to increase the productivity of architects and designers.
- Simulation models for architecture exploration and optimization should be made available.
- Tools to support early verification of embedded software are needed.
- An open source repository of peripheral models, interconnect, etc., with different abstraction levels is needed.

Automated Composition/Optimization Tools

- Tools are needed to support complex heterogeneous SoCs composed of a mix of RISC-V cores, accelerators and arbitrary design IP to increase productivity in SoC design.
- Integration with existing frameworks and development of extended capabilities including the generation of HW hardware/software interfaces and low-level FW is required.

Functional Verification

- There is a need for an automated industry strength verification, including formal apps, of caches, interconnect, and memory consistency in multi core systems.

[49] M. Fadiheh, A. Wezel, J. Mueller, J. Bormann, S. Ray, J. Fung, S. Mitra, D. Stoffel, W. Kunz: An Exhaustive Approach to Detecting Transient Execution Side Channels in RTL Designs of Processors, https://arxiv.org/abs/2108.01979

- An automated tool-agnostic verification flow integrating simulation-based, formal and emulation-based verification compliant with safety standards is essential to lower the adoption barrier of open source design IP and productivity of SoC design.
- Point technologies need to be integrated into automated flows to hide complexity of verification away from users.
- An open source verification database including artifacts along lifecycle of design and verification of IPs for AI/ML-enabled EDA tool R&D is required.
- To ensure system software in sync with the RTL, additional automated formal verification apps for boot code and firmware together with the RTL are crucial.
- Design IP code quality is crucial on top of functional correctness. Integration of Lint/DRC, CDC, RDC, formal auto-checks, compliance with downstream flows, CI/CD flows into open source design flow management frameworks and point technologies for individual analysis steps is needed.

Functional Safety and Security

- Reference flows should be defined that describe and demonstrate the state-of-the-art methodologies in order to deliver products that maximize confidence and optimize development efficiency.
- Automated tools are needed to provide quantifiable verification and validation of safety and security allowing developers to focus on their core differentiation technologies.
- Tools, methodologies and associated work products should be developed to that they enable sharing of information throughout the supply chain to ensure the end system is free from safety and security vulnerabilities.

Customization Tools

- Automated application-driven identification and implementation of custom instructions and accelerators is needed in order to customize generic RISC-V design IPs to specific application segments.

Interoperability and Reuse

- There is a need for increased modularity and re-use of subsystems between building blocks. For the design of IP blocks, a gap is that the RISC-V foundation mainly focuses on the ISA interface specification and it has been argued there is relatively little / limited re-use between different components for RISC-V systems.[50]

Hardware/Software Co-certification

- Tool support is needed for co-certification of combined hardware/software systems.

[50] Taylor, Michael Bedford. "BaseJump STL: SystemVerilog Needs a Standard Template Library for Hardware Design." In 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC), 1–6. San Francisco, CA: IEEE, 2018. https://doi.org/10.1109/DAC.2018.8465909.

7 ROADMAP

In this section the important elements of the proposed Roadmap for European Sovereignty in Open Source HW & SW and RISC-V Technologies are given. This is based on the key messages derived from the previous discussions of hardware technology, supporting software and cross cutting requirement needs. These are summarized below:

- Domain-Specific Architectures/Accelerators are one of the major opportunities for advances in computer architecture and have great innovation potential for academia and industry.
- Accelerators need to be integrated with a general-purpose computing environment, in hardware at the chip- and system-levels, as well as into software stacks and programming tools to fully exploit their potential.
- Hardware integration requires significant engineering effort (interface IP blocks, high-performance printed circuit board design) that impedes innovation on accelerators by smaller players, such as the academia and SMEs, who often initiate open source efforts.
- The innovation potential could be unlocked by funding R&D into open standardized interfaces, and their corresponding hardware realizations, as well as scalable & reusable technology templates at both the system-on-chip and computing system-levels. These could then be used by innovators to turn their accelerator architecture ideas into practically usable artefacts (open source releases, products and services offered around the open source releases).
- The challenges of providing software support for innovative accelerators should be addressed. This requires R&D funding for new or improved general purpose accelerator integration software stacks, targeting one or [more] models of computation. Better developer support is also needed for adding new accelerators to established domain specific software frameworks, e.g., machine learning or image processing.
- The existing design sharing and commercialization mechanisms that exist within EUROPRACTICE should be leveraged to promote exchange of developed IP blocks between collaborators and commercialization of IP developed in whole or part within academia. (Note this is already done by the particle physics community to collaborate and contribute designs to CERN).
- Funding should be provided to commission the development/licensing of the required artefacts (e.g., for IP blocks, interfaces and templates) from academia or industry in an open source manner. These should then be made available to users in a “one stop shopping" approach similar to the existing EUROPRACTICE Foundry Access service. Support should also be provided for training & supporting users of the artefacts for a multi-year time period.
- For general-purpose and domain-specific accelerator software stacks a more decentralized approach is needed. Funding should be offered to academia and industry to create/provide the required artefacts in an open source manner, which could then be disseminated using established channels (e.g., github, local repositories). These artefacts may encompass documentation, training materials, and sample back-end implementations for established domain-specific accelerator frameworks. Training and support are also required for a multi-year time period.
- Creating an open source hardware/software ecosystem for innovation in accelerators and domain-specific architectures will require funding schemes which ensure that, after an initial broader experimentation phase, future funds are directed at those technologies that prove most beneficial in practice, e.g., measured in actual user uptake.

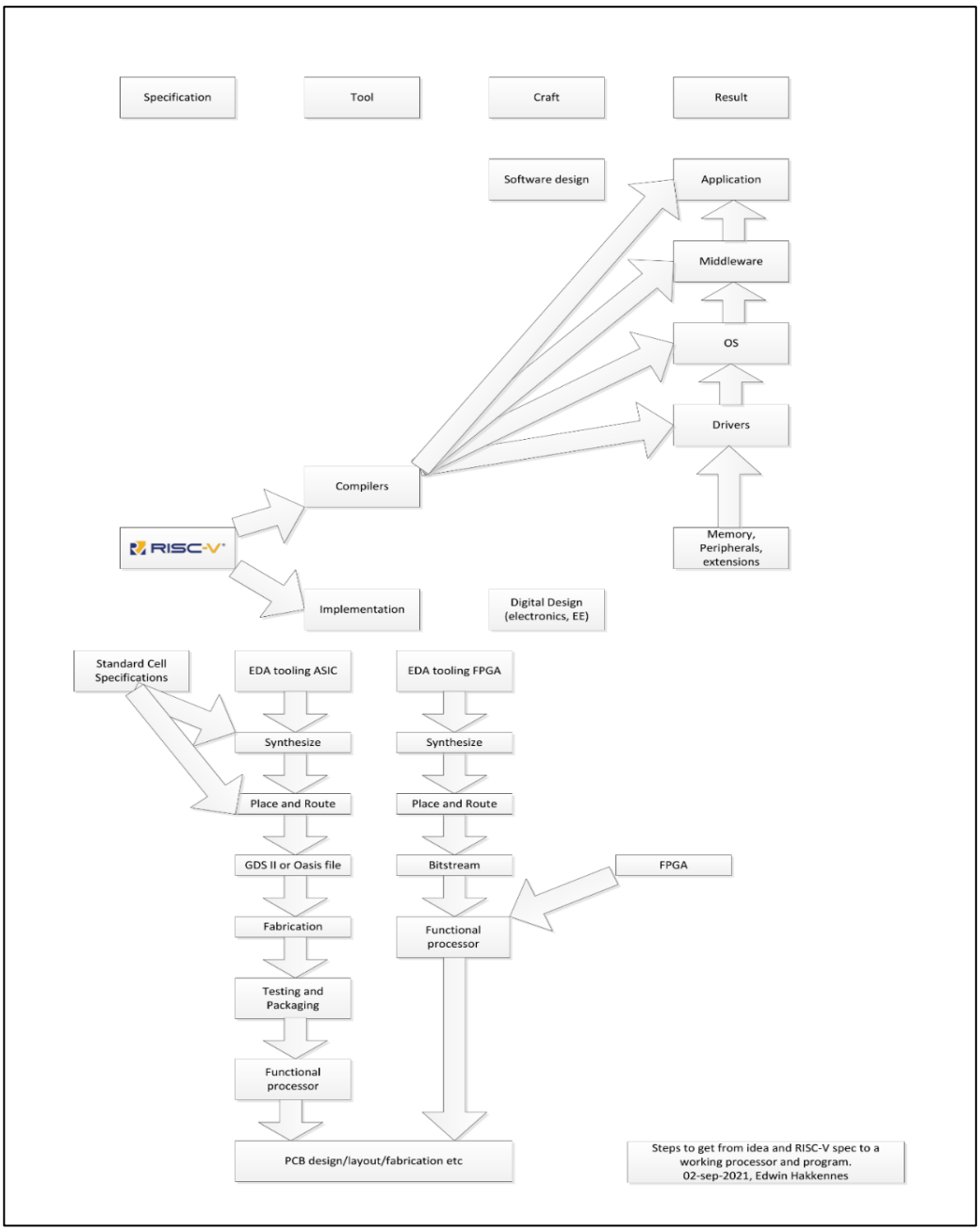


Figure *9* - Example Process for RISC-V

In order to meet the needs of the future it is important to sketch out a proposed process that is required to create a working processor. This is done in Figure 9 highlighting the necessary steps and activities in terms of design, fabrication and testing considering the hardware and software aspects. At the application level non-functional properties such as WCET for safety and security need to be addressed. The process shown has been derived for RISC-V, but it could be updated at a later stage for other IP's.

Going beyond this there is a need to develop other IPs. A non-exhaustive list of other IPs identified by the Working Group that will be required in order to give a good coverage of open source cores for making SoCs is given in Table 1. These could be used for monolithic SoC integration and for chiplet integration.

- Memory controllers addressing Double Data Rate (DDR), Low Power Double Data Rate (LPDDR), High Bandwidth Memory (HBM) and handling the different end use applications and processing types.
- RISC-V based power management controller with placeholder for customer-specific sensors and actuators.
- RISC-V based security controller with placeholder for customer-specific secure elements (Physical Unclonable Function (PUF), cryptographic IPs, etc.).
- Coherent cache infrastructure for many-cores with controller directory.
- Scratchpad memory infrastructure for many-cores with smart DMA.
- NoC with on-chip interfaces at router level to connect cores (coherent), memory (cache or not) and IOs (IO coherent or not).
- SerDes library in line with PCIe standards.
- PCIe controllers and switches.
- Chiplet and interposer interfacing units (similar to pads) (only for chiplet + interposer approach).
- Trace and debug solutions.
- Monitoring IP for safety, security, design integrity.
- Guidelines for selection of low/medium speed peripherals (I2C, UART, SPI, etc.).

Table *1* - List of IPs Required

All of these IPs have to be delivered with a verification suite and will need to be maintained constantly to keep up with errata from the field and to incorporate newer requirements.

The availability of standardized metrics is crucial. The application scenario may demand certain boundaries on power, performance, or area of the IP, so searches across different repositories with standardized metrics on these indicators are needed to successfully use these IPs. Building on standardized metrics for all these crucial features, end users will be able to pick the most suited IP for their application and get an idea on needed additional efforts in terms of certifications.

Recommendation - Innovation in open source hardware would profit immensely if lower-level interface blocks could be made available in a low-cost manner, at least for selected chip fabrication processes supported by facilitators such as EUROPRACTICE to allow low-barrier prototyping (e.g., the miniASIC and MPW programs). The availability of automated SoC composition is also desirable to quickly transform innovation into Proof of Concept and to bring productivity gains and shorter time-to-market for industrial projects.

IPs including all artefacts (e.g., source code, documentation, verification suites) should be made available ensuring integrity, authenticity and traceability using certificate-based technologies. Traceability along the supply chain of R&D processes is a foundation for later traceability of supply chains for components in manufacturing and deployment/operation.

Finally, an ecosystem of chiplet + interposers can be enabled via open source development. The key "missing link" here is Die-to-Die (D2D) communication. This is needed to create a chiplet-based design ecosystem. **Development of Open source D2D communication could become one of the most important developments in coming years enabling wide-scale use of chiplets with potential to become a "de-facto" standard.**

8 PRIORITISATION OF OPEN SOURCE ECOSYSTEM NEEDS

In this section the short-, mid- and long-term needs for different items on the roadmap are given. They are based on a detailed analysis of gaps and future needs identified. For some critical aspects, proof-of-concept down to physical implementation or layout is needed. For other aspects, a proof-of-concept at synthesis level (netlist) would be sufficient, e.g., for mapping onto a FPGA.

8.1 Repository of RISC-V based Processor Platforms

Short-term needs (2-5 years):

- High-end: Highly customizable Multi-core Out of Order 64-bit open source infrastructure with the associated memory hierarchies (caches&coherency, off-chip) and communication (fast cores to cores, cores to accelerators, cores/accelerators to system). This should be suitable for various instances of processor IP.
- Highly customizable high-end domain-specific cores for high-performance embedded system and/or general-purpose application (link with EuroHPC-call on HPC processors)
- Mid-end: Highly customizable mid-end Open Cores (32-bit) with support for advanced 32-bit ISA and system extensions such as security, scalable vector (Zve), tiny FP (Zfinx) or support for specific arithmetics, DSP, bit-manipulation, scalable and customizable interrupt management such as AIA/ACLINT, e.g., for low latency in critical workloads (TinyML, near-sensor processing, secure IoT stacks).

Mid-term needs (5-10 years):

- High-end: Moving coprocessor functions inside the core by adding new instructions (variable precision, customizable vector, tensor, etc.) with all the necessary software environment support (compiler, libraries, automated HW/SW co-design flows) and validation/verification environment.
- Mid-end: Parametrizable open source soft RISC-V cores, and a range of associated interoperable IPs (e.g., interconnect), in relationship with open source EDA flow (this can lead to a tool generating RTL code according to high-level specifications).

8.2 Domain-Specific Processor Features

A verified open source hardware for high-assurance and reliable systems via a whitebox approach.

Short-term needs (2-5 years):

- Provide public artifacts for safety and security by architecture at an initial assurance level.
- Provide public artifacts for safety and security by architecture at a high assurance level, e.g., including formal models.

Mid-term needs (5-10 years):

- Demonstrate how the approach can be carried over to more complex compositional architectures (e.g., multipipeline CPU, a full advanced embedded board or even general-purpose desktop computer mainboard).

8.3 Repository of Open Source HW Peripheral Blocks

Create in open source all the generic re-usable elements required to create a complete SoC. All the IPs should be interoperable and composable to build a complete SoC.

Short-term needs (2-5 years):

- High-speed open source memory infrastructure (DDR3+, SDRAM, HBM memories).
- Open source high-performance interfacing: PCIe Gen3+, Ethernet 1+G, and USB 2.0/USB 3++ interfaces and controllers, including process-specific analog PHY (SerDes) components.
- Open source test, trace, debug IP blocks (e.g., JTAG, etc.).
- All the ancillary functions (power management, SPI, I2C, etc.) as open source blocks.
- Support for heterogeneous many-core open source SoCs (e.g., scalable coherent caches, scratchpad memories, w/smart DMA, and a coherency-capable network-on-chip).
- Inter-chip communication links with 2+ Gb/s per pin.
- Template for ASIC SoC and PCB designs, the latter for PCIe-attached accelerators, similar to existing FPGA-based evaluation/prototyping boards, but for plugging in user-provided ASICs.
- Easy and affordable “one stop” access to and support for the IP and technologies developed above.
- Open source repository of peripheral models with different levels of accuracy with their verification artefacts.

Mid-term needs (5-10 years):

- Updates of base building blocks to track emerging standards, possibly also adding support for improvement of 2.5D/3D technologies such as High Bandwidth Memory.
- Support open source infrastructure for easy building of SoC from various sources of IPs, in an as automated way as possible.

Long-term needs (beyond 10 years):

- Updates of base building blocks to track emerging standards and technologies.

8.4 Interconnect for Real-Time and Mixed Criticality

Verified open source real-time interconnects between logical units such as cores, memory and other peripherals, regardless of whether they are on the same or different chiplet.

Short-term needs (2-5 years):

- Verified open source real-time interconnect proof-of-concept.
- Verified ready-to-use Design IP blocks for safety-critical interconnects.
- Demonstrate use of interconnects by integrating with existing RISC-V open hardware CPUs, memory controllers, and other relevant resources.
- Verification environment for targeting real-time interconnect, including formal proof of worst-case execution time.

Mid-term needs (5-10 years):

- Verified open source real-time interconnects, including mixed criticality, reliability and security, including industrial-grade verification artefacts.
- Verified ready-to-use Design IP blocks for real-time interconnect for mixed criticality. This interconnect shall give system integrators full control over allocation of available bandwidth to different subjects.
- Demonstrate use of interconnects by integrating with existing RISC-V open hardware CPUs, memory controllers, and other relevant resources.
- Verification environment for target for real-time interconnect, including formal proof of worst-case execution time, including mixed criticality cases.

8.5 Interconnect for System Integration

IP can be integrated in a monolithic SoC or can be part of chiplets that are physically interconnected through an interposer, which can also provide services (e.g., routing, power supply). A European ecosystem should emerge from this approach, as it is seen as a major evolution in the chip industry by TSMC for example, and now used by AMD, Intel, Nvidia, etc. Industrials are already working on setting up specifications for "*Building an open ecosystem of chiplets for on-package innovations*", like ODSA with its "bunch of wires" and the Universal Chiplet Interconnect Express consortium (The promoter members of UCIe are AMD, Arm, Advanced Semiconductor Engineering, Inc. (ASE), Google Cloud, Intel, Meta, Microsoft, Qualcomm, Samsung, and Taiwan Semiconductor Manufacturing Company)[51].

Short-term needs (2-5 years):

- Verified open source interconnect between chiplets and Interposer.
- Verified open source interconnect for active interposers with parametrization bandwidth, latency, energy and performances.
- All the tools and views to easily use various chiplets and design affordable interposers.
- Tools to help the partitioning of complex designs (Design Space Exploration including chiplet/interposers and 3D technologies).

8.6 Domain-Specific Accelerators

Development of flexible hardware/software frameworks for accelerator-intensive System-on-Chips.

Short-term needs (2-5 years):

- Create open source template system-on-chip designs including processors, on-chip/off-chip memory and peripheral interfaces/controllers enabling the easy insertion of domain-specific accelerators.
- Create open source software-stacks (e.g., middleware, drivers, advanced OS integration, programming tools) for interacting with the accelerators from software.
- Create reusable and configurable state-of-the-art open source domain-specific accelerators for common operations, e.g., in the areas of security (crypto), ML/AI, communications, information/signal processing.

Mid-term needs (5-10 years):

- Update SoC templates, software stacks and reusable accelerators to track the state-of-the-art, possibly moving to full support for 2.5D/3D technologies such as multi-process heterogeneous stacks.

Long-term needs (beyond 10 years):

- Update SoC templates, software stacks and reusable accelerators to track the state-of-the-art.
- Interoperability with unconventional acceleration hardware solutions, e.g., neuromorphic computing and quantum computing.

[51] https://www.uciexpress.org/

8.7 Software

A successful processor requires industry-grade software and tools e.g., compilers/debuggers.

Short-term needs (2-5 years):

- Euro Centralized SW Open Source Consortium.
- Development of and or extension of existing compilers and core libraries with emphasis on:
    - Code density
    - Performance
    - Memory Hierarchy

- Safety-certified open source RTOS and/or hypervisor/separation kernel (e.g., Zephyr, Tock or other hypervisors/separation kernels).
- Open source AI codesign tools, HW/SW optimized, in order to tweak the application on the RISC-V platform.

Mid-term needs (5-10 years):

- Open source ISA Extension design tooling/management (Simulate/Iterate/Create).
- Open source Core Validation Test SW.
- Open source Safety/Security Cert Tooling.
- ISA migration tools.

Long-term needs (beyond 10 years):

- Unconventional acceleration software solutions, e.g., neuromorphic computing.

8.8 Methodology and EDA Tools

Development of open source SoC development tools and/or RISC-V specific enhancements of leading-edge closed source tools, including interoperability of open and closed source tools.

Short-term needs (2-5 years):

- Tools for design space and architecture exploration and fast simulation of RISC-V based SoC architectures to optimize processor core configuration, memory hierarchy and HW-SW-partitioning for multiple criteria like performance, latency, power and area (e.g., for integration with model-based systems engineering frameworks), tools for automated system-on-chip composition.
- Tools for automated exploration and implementation of RISC-V application specific customization like custom instructions and tightly coupled accelerators.
- Investment in developing open source interoperability standards to interface between different parts of the ASIC development toolchain (both open and closed), encouraging proprietary vendors to adopt the standard.
- Point investment in open source replacements for specific parts of the flow (synthesis, linting, formatting, simulation, DRC, etc.). Focus on integration and functionality, e.g., in the form of stable and easy-to-use reference flows, but less on state-of-the-art algorithms.
- Open source tools for new, more software-driven verification methodologies, possibly based on next-generation hardware description languages, tested on open source IP.
- Investment into open source Continuous Integration infrastructure for open source IPs/cores, extracting metrics, providing feedback to change requests.

- Open source infrastructure with a well-defined semantic layer enabling to use open source IPs and their associated design flows to train AI/ML models enabling faster, higher quality design and verification.
- Investment studies on using selected open source tools in the flow to realize and evaluate practically relevant sample designs, where innovation can be produced using older, lower-complexity processes.
- Automation support for the composition of complex SoCs (heterogeneous mix of cores and accelerators), including generation of hardware/software interfaces and integration/abstraction with ASIC development toolchain (both open/closed).
- Extending tools and tool flows for full compliance with domain specific standards and regulations like functional safety (e.g., ISO26262, IEC61508, EN50129).

Mid-term needs (5-10 years):

- Enhance open source verification tools towards the state of the art, adding more automation, point tools, interactive debug capabilities, and integration with requirements tracing.
- Extend open source tool flows to also cover non-functional aspects, allowing certifiably secure and/or safe designs.
- Update initial open source tools with more advanced algorithms for better Quality of Result and support for more advanced processes, focusing on those with fabrication capabilities located in Europe.
- Enhancing tools for the earlier phases of the EDA pipeline for high-end complex designs
- Enhancing sections of the complete flow for full automation without human expert intervention in the loop as a “push-button-flow”.
- Extend open source tool flow to cover a larger part of the complete EDA pipeline. Focus on achieving a *front-to-back* open source tool flow capable of handling less complex designs on commodity processes.

Long-term needs (beyond 10 years):

- Complete front-to-back open source EDA tool flow capable of realizing even more complex industrial and academic designs with acceptable results, targeting current fabrication processes.

8.9 Domain-Specific Demonstrators

It is of key importance that the RISC-V processor families and related Open Source hardware, software and EDA tools are applied in challenging demonstrators.

Short-term needs (2-5 years):

- Domain specific adaptation of RISC-V based processor solutions for safe, secure and reliable computationally intensive applications, e.g., for automotive, industrial automation, medical applications, etc. Such solutions are expected to address appropriate functional and non-functional, high-performance requirements aiming at realizations on advanced technology, e.g., 16 nm Finfet or below.

Mid-term needs (5-10 years):

- Domain specific adaptation of RISC-V based processor solutions for safe, secure and reliable computationally intensive applications, e.g., for automotive, industrial automation, medical applications, etc. Such solutions are expected to address appropriate functional and non-

functional, high-performance requirements aiming at realizations on advanced technology, e.g., 7 nm or below.

Long-term needs (beyond 10 years):

- Domain specific adaptation of RISC-V based processor solutions for safe, secure and reliable computationally intensive applications, e.g., for automotive, industrial automation, medical applications, etc. Such solutions are expected to address appropriate functional and non-functional, high-performance requirements aiming at realizations on advanced technology, e.g., 5 nm or below.

9 RECOMMENDATIONS

In the following sections the specific recommendations and needs are broken down.

9.1 Specific Recommendations

9.1.1 Development of Open Source Hardware

- Fund the creation of open source or easily accessible low-cost/no-cost, fundamental building blocks (e.g., chip IPs (including processors – RISC V, accelerators, peripherals, data management IPs, debug, etc.), templates for SoCs/chiplets/interposers/PCBs, software frameworks for heterogeneous SoC operation) that reduce the high engineering effort required to practically realize a new hardware design and allow creators from academia and SMEs to focus on their actual innovation.
- The building blocks should have all the views, software supports (drivers), test and documentation so that they can be easily combined (interoperable) and used (support).
- Make these building-blocks available for free, or with only minimum financial and administrative overheads, from “one stop shops”, e.g., by integrating them into service portfolios of organizations such as OpenHW Group or EUROPRACTICE.
- Ensure that these building blocks are distributed under an open source license that is adapted to hardware artefacts allowing exploitation by all stakeholders (semiconductor industry, OEMs, SMEs, academy/research, etc.).

9.1.2 Community Support

- Provide a one stop shop model with long-term activities and overall support (e.g., advice for licensing, productization, etc.) for SMEs and start-ups.
- Encourage the use of standard specifications and standardization efforts when gaps are identified.

9.1.3 Development of a Chiplets + Interposer Ecosystem in Europe

- Encourage an ecosystem of chiplet + interposers via open source development. Die-to-Die (D2D) communication is the “missing link” to leverage the chiplet-based design ecosystem, and its development in open source would enable a wide adoption and could result in a “de-facto” standard.
- A SoC infrastructure template should be developed that includes communication between IPs, interfaces with memories and the external world, supported by tools that allow the easy integration of new chiplets. A validation suite should be developed allowing the rapid design of a complete SoC from various Open source (or not) IPs, together with a set of “best practices” allowing SMEs, start-ups and industries to “make the step” towards the chiplet+interposer approach.

9.1.4 Tools, Validation, Methods and Demonstration

- The development of industrial strength open source design IPs would benefit greatly from proprietary EDA tools. The licensing of the EDA tools should be made available under appropriate conditions, not at the prohibitive prices paid by commercial end-users, and with permissive licensing of the designed IPs.
- Funding should be provided to create an open source EDA ecosystem. The effort should be guided by the steps (e.g., logic synthesis, placement, routing) of a typical ASIC EDA flow, and proceed by

incrementally replacing one or more closed-source steps with open source tools. To enable the required interoperability between open and closed source tools (similar to the FPGA/ASIC space) open interfaces (e.g., APIs, data formats) between the steps are required to allow open source in and open source out.

- The development of re-usable verification infrastructure (e.g., IP blocks accompanied by test frameworks) should be supported.
- Industrial demonstrators using Open Source IPs (RISC-V hardware, accelerators and SoC IPs) should be supported to validate the complete chain. If these industrial demonstrators include Printed Circuit Boards, the designs of the PCBs should also be licensed under an appropriate open hardware license. Similarly, software developed for these demonstrators and any documentation should be made available under appropriate software and documentation licenses.

9.1.5 Special focus on European Need in Terms of Safety/Security Solutions

- Research projects creating methods to develop safety- and security-critical open source hardware should be supported. Key aspects are collaboration, documentation, verification and certification in open source communities.
- A re-usable verification infrastructure (e.g., IP blocks accompanied by test frameworks) for safety/security should be established.
- Industrial demonstrators using RISC-V hardware, especially in the safety/security area, should be supported to promote hardware/software co-certification for safety and security.

9.2 Global Longer-Term Recommendations

In addition to short term goals there are also longer-term goals that need to be supported to create a European critical mass and ecosystem in open source. At the same time there is a need to educate the public and industry on the economic and sovereignty benefits of open source approaches.

9.2.1 Non-profit Organisation for Coordinating European Open Source HW IP providers

- A neutral non-profit organisation could be set up for coordinating European Open Source HW IP providers. The aim of this organisation would be to develop a compliance standard that certifies interoperability and industrial readiness of Open Source HW solutions. The organisation would also orchestrate market specific requirements (e.g., safety and security features for the automotive or industrial automation domains). Although the organisation could be funded by the EU, it may have to be open also for companies outside the EU to avoid the emergence of closed Open Source HW IP clusters around the world.
- Set-up an IP exchange system between academia and industry (integrated in a one stop shop model). This will encourage new business models for EDA vendors, design & IP houses and IC foundries.
- Ensure that services that allow the realization of test chips or small production runs will continue to support prototyping and small series manufacturing in Europe. This includes providing all the libraries (PDKs) and support for transforming open sources IPs into silicon chips.

9.2.2 Educational measures

- It is very important to communicate the advantages and reasons for open source hardware to the public and industry in an effective way. This requires development of appropriate pedagogical material and communication campaigns. In particular, it is necessary to explain how open source hardware can have a very positive impact on the economy of the EU, and why it is strategically important in guaranteeing digital sovereignty.
- To change mind sets the EC should provide incentives to public institutions in the EU, including e.g., universities and research laboratories, so that workers in those institutions contribute more to open hardware developments, with that work appropriately recognized in their career development. Open Hardware should become the default paradigm for all hardware development in publicly financed institutions.

9.3 Summary Table: Key Topics and Timescales

A roadmap needs concrete actions and timescales in order to become operational. The Working Group has thus developed a detailed list of key topics which need addressing in the short term (2-5 years), medium term (5-10 years) and long term (>10 years). These can be used as input into strategic actions to address the core aspects and needs highlighted in this report. Notably there will also be a need for political or bilateral actions between specific stakeholders to take the roadmap forward.

Topics	**Overall Topics**	**Short Term (2-5 years)**	**Mid Term (5-10 years)**	**Long Term (> 10 years)**
Open HW Base Building Blocks	Repository for open source HW base building blocks.	Create in open source all the elements required to create a complete SoC: (1) Support the creation of: • High-speed open source memory infrastructure (DDR3+, SDRAM, HBM memories); • Open source High performance interfacing: PCIe Gen3+, Ethernet 1+G, and USB 2.0+ interfaces and controllers, including process-specific analog PHY (SerDes) components; • Open source Test, trace, debug IP blocks (e.g., JTAG, etc.); • All the ancillary functions (power management, SPI, I2C, etc.) as open source	Support updates of base building blocks to track emerging standards, possibly also adding support for 2.5D/3D technologies such as HBM. Supporting open source infrastructure for easy building of SoC from various sources of IPs, as automated as possible.	Support updates of base building blocks to track emerging standards and technologies.

Topics	Overall Topics	Short Term (2-5 years)	Mid Term (5-10 years)	Long Term (> 10 years)
		blocks. Remark: All the IPs should be interoperable and composable to build a complete SoC. (2) Create support for heterogeneous many-core open source SoCs (e.g., scalable coherent caches, scratchpad memories, w/ smart DMA, and a coherency-capable network-on-chip). (3) Create open source Inter-chip communication links with 2+ Gb/s per pin. (4) Create open source template ASIC SoC and PCB designs, the latter for PCIe-attached accelerators, similar to existing FPGA-based evaluation/prototyping boards, but for plugging-in user-provided ASICs. (5) Establish easy and affordable "one stop" access to and support for the technologies developed in (1-5).		
Processors	Verified open source hardware for high-assurance systems by whitebox approach.	Provide public artifacts for safety and security by architecture at low assurance level. Provide public artifacts for safety and security by architecture at high assurance level, e.g., including formal models.	Demonstrate how the approach can be carried over to more complex compositional architectures (e.g., multipipeline CPU, a full advanced embedded board or even general purpose desktop computer mainboard).	Based on feedback of previous work.
	High-end application cores for High-performance embedded	Develop Multi-core Out of Order 64-bit open source infrastructure with all the near memory communication support	Move coprocessor functions inside the core by adding new instructions (variable precision, vector, tensor,	

Topics	Overall Topics	Short Term (2-5 years)	Mid Term (5-10 years)	Long Term (> 10 years)
	system and/or general-purpose application (link with EuroHPC-call on HPC processors).	(caches) and communication (fast cores to cores and core to accelerators). This should be suitable for various instances of processor IP.	etc.) with full software environment support (compiler, libraries) and validation/verification environment.	
	Open Cores (32-bit) with support for advanced 32-bit ISA and system extensions such as security, vector (Zve), tiny FP (Zfinx), DSP, bit-manipulation, fast interrupts (CLIC) for critical workloads (TinyML, near-sensor processing, secure IoT stacks).	-	Create parametrizable open source soft RISC-V cores and a range of associated interoperable IPs (e.g. interconnect), supported by open source EDA flow.	
Domain-Specific Accelerators	Flexible Hardware/Software Frameworks for Accelerator-intensive System-on-Chips.	(1) Create open source template system-on-chip designs including processors, on-chip/off-chip memory and peripheral interfaces/controllers enabling the easy insertion of domain-specific accelerators. (2) Create open source software-stacks (e.g., middleware, drivers, advanced OS integration, programming tools) for interacting with the accelerators from software. (3) Create reusable and configurable state-of-the-art open source domain-specific accelerators for	Update SoC templates, software-stacks and reusable accelerators to track the state-of-the-art, possibly moving to full support for 2.5D/3D technologies such as multi-process heterogeneous stacks).	Update SoC templates, software-stacks and reusable accelerators to track the state-of-the-art.

Topics	Overall Topics	Short Term (2-5 years)	Mid Term (5-10 years)	Long Term (> 10 years)
		common operations, e.g., in the areas of security (crypto), Machine Learning/AI, communications, information processing, etc.		
Interconnect for real-time and mixed criticality	Verified open source interconnects.	Create a verified open source real-time interconnect PoC Develop verified ready-to-use Design IP blocks for safety-critical interconnects. Demonstrate use of interconnects by integrating with existing RISC-V open hardware CPUs, memory controllers, and other relevant resources. Provide verification environment for real-time interconnect, including formal proof of worst-case execution time.	Create verified open source real-time interconnects, including mixed-criticality and security. Develop verified ready-to-use Design IP blocks for real-time interconnect for mixed criticality. This interconnect shall give system integrators full control over allocation of available bandwidth to different subjects. Demonstrate use of interconnects by integrating with existing RISC-V open hardware CPUs, memory controllers, and other relevant resources. Provide verification environment for target for real-time interconnect, including formal proof of worst-case execution time, including mixed criticality cases.	Based on feedback of previous work.
Software	Organize: Open Source SW. Develop: Software Tools.	Establish Euro Centralized SW Open Source Consortium. Support compiler projects -Code density	Create Open source ISA Extension design tooling/management (Simulate/Iterate/ Create). Create Open source Core validation Test SW.	Create Open source AI Codesign tools, HW/SW optimized.

Topics	Overall Topics	Short Term (2-5 years)	Mid Term (5-10 years)	Long Term (> 10 years)
	Execute: Real Time Operating Systems.	-Performance -Memory Hierarchy Create Safety Certified Open Source RTOS (e.g., Zephyr).	Create Open source Safety/Security Certification Tooling ISA migration tools.	
Methodology & EDA Tools	Open source ASIC development tools, including interoperability of open and closed source tools.	(1) Develop open source interoperability standards to interface between different parts of the ASIC development toolchain (both open and closed), encouraging proprietary vendors to adopt the standard. (2) Identify and target investment in open source replacements for specific parts of the flow (synthesis, linting, formatting, simulation, place and route, Static Timing Analysis, Design Rule Checking etc.). Focus on integration and functionality, e.g., in the form of stable and easy-to-use reference flows, rather than state-of-the-art algorithms. (3) Create Open source tools for new, more software-driven verification methodologies, possibly based on next-generation hardware description languages, tested on open source IPs. (4) Investment in open source CI infrastructure for open source IPs/cores, extracting metrics, providing feedback to	(1) Enhance open source verification tools towards state of the art, adding more automation, point tools, interactive debug capabilities, and integration with requirements tracing. (2) Extend open source tool flows to also cover non-functional aspects, allowing certifiably secure and/or safe designs. (3) Update initial open source tools with more advanced algorithms for better QoR and support for more advanced processes, focusing on those with fabrication capabilities located in Europe. (4) Extend open source toolflow to cover a larger part of the complete EDA pipeline. Focus on achieving a *front-to-back* open source tool flow capable of handling less complex designs on commodity processes.	Complete front-to-back open source EDA tool flow capable of realizing even more complex industrial and academic designs with acceptable results, targeting current fabrication processes.

Topics	Overall Topics	Short Term (2-5 years)	Mid Term (5-10 years)	Long Term (> 10 years)
		change requests. (5) Perform investment studies on using selected open source tools in the flow to realize and evaluate practically relevant sample designs, where innovation can be produced using older, lower-complexity processes.		
	Chiplets & interposers for modular architectures.	Develop an Open source Die-to-Die (D2D) communication interface to leverage the chiplet-based design ecosystem to enable wide usage with target to become a "de-facto" standard.	Develop and promote a SoC infrastructure for "chiplet-based" components, with all the views and tools allowing easy partitioning and use of various chaplets (from various origins). Provide support for active interposers.	Develop support for photonic interposers and evolution of automated toolset environment to increase productivity of developers.
	Analog IPs.	For SoC, develop analog IPs which are an inherent part of the design solutions (PLL, FLL, DC/DC, ADC/DAC...). This will also help address the resistance of EDA companies to allow open sourcing of Analog IPs.		
	Economics and social aspects of open hardware.	Perform research on economics and social aspects of open hardware.		

Table *2* - Key Topics and Timescales

10 NEED FOR HORIZONTAL AND VERTICAL ACTIVITIES

It is clear that in order to address the needs of Open Source development within Europe both horizontal and vertical activities are needed. These can be summarized as:

Horizontal Activities

- Fundamental Research
- Capacity Building
- Platform Components

Vertical Activities

- Use Case Scenarios
- Application Platforms
- Platform Usage

Recommendation – Horizontal foundational activities and vertical application driven activities are needed to develop new tools and IP libraries which can be built upon, re-used and exploited in future horizontal and vertical calls, systematically growing a repository of benefit to European stakeholders. In parallel with this there is a need for a strategic "Governance Initiative'' to coordinate activities, maintain and promote the repository.

In Figure 10 a proposed vision is given on how these horizontal and vertical activities could be combined to support a European RISC-V and Open Source ecosystem. This requires work on "Foundational Building Blocks" to provide both HW and SW building blocks, along with the development of tools to integrate these building blocks into complex SoC designs. Evolution is needed in both the vertical and horizontal activities. Vertically, the basic foundation infrastructure needs to be expanded with new IP's and different technological dimensions. Horizontally, there is a need for development of three types of processor families, ranging from low- via mid- to high-end generic processors. In addition, there is a need to develop application-specific elements that can be added to the generic processors in order to create dedicated solutions for specific domains, e.g., automotive, industry, telecommunications, health, aerospace, defense, etc. There is also a need for horizontal activities addressing security, verification and certification which can then be exploited in the vertical domains targeting specific implementations.

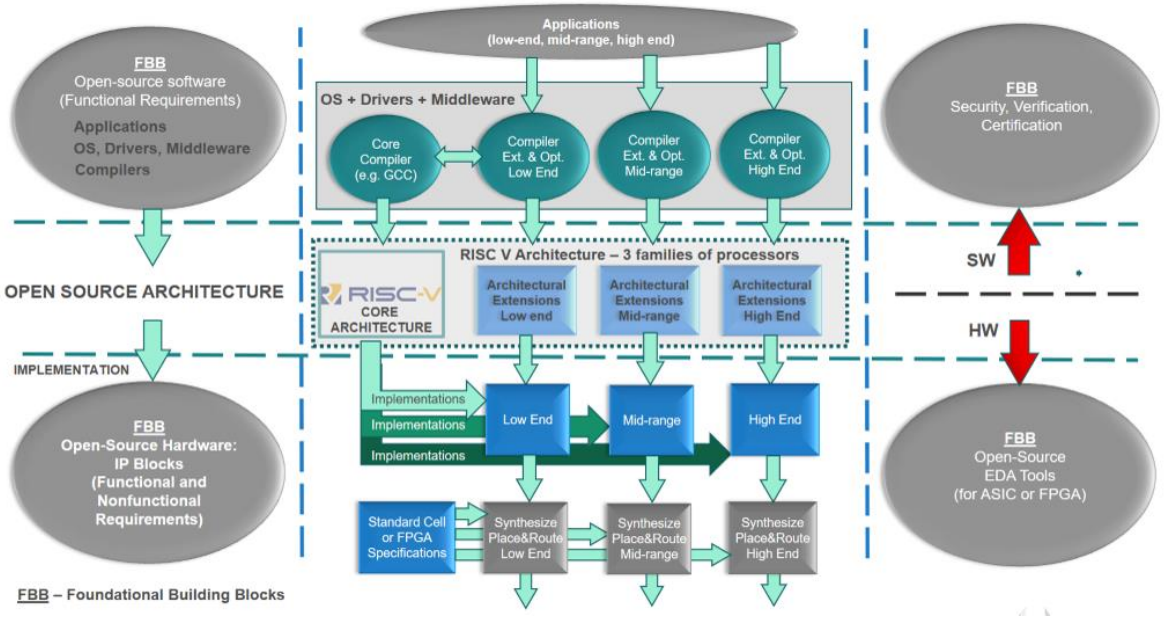


Figure *10* - Horizontal Foundational Activities and Vertical Industry Facing Activities

Developing the IP and tools to support the design of new processor architectures will not result in European benefits unless these designs can be physically realized. To support this there is a need for further horizontal and vertical activities as shown on Figure 11 to address the needs of fabrication considering the non-functional properties. Here there is a need for linkage to pilot lines to fabricate, test and package chips, firstly at a prototyping level, and then considering fabrication for specific application sectors.

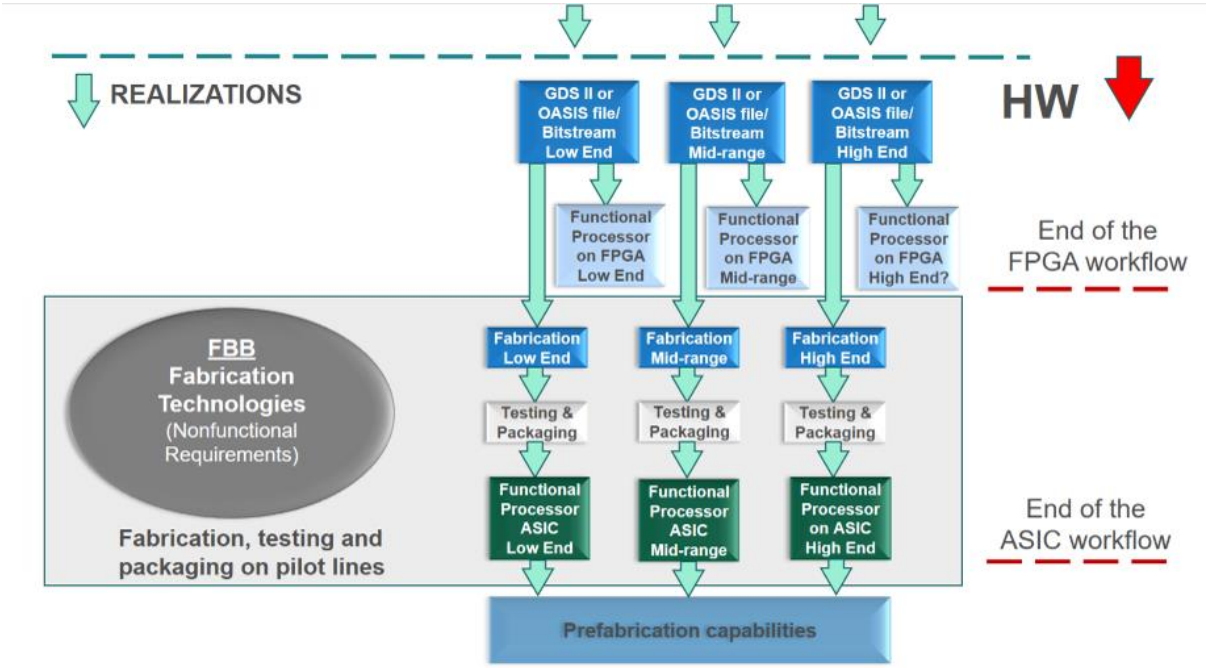


Figure *11* - Horizontal and Vertical Fabrication and Testing Activities

For the vertical application demonstrators key strategic areas are automotive, industry, high performance computing and communication. A first key vertical application demonstrator could be in the automotive sector in cooperation with OEMs, TIER 1's and TIER 2's. To support development of applications there is a need to intensify interaction with key strategic European stakeholders in application sectors and in other relevant PPPs (e.g., EuroHPC, SNS, etc.) to jointly define specifications, development plans and financing models.

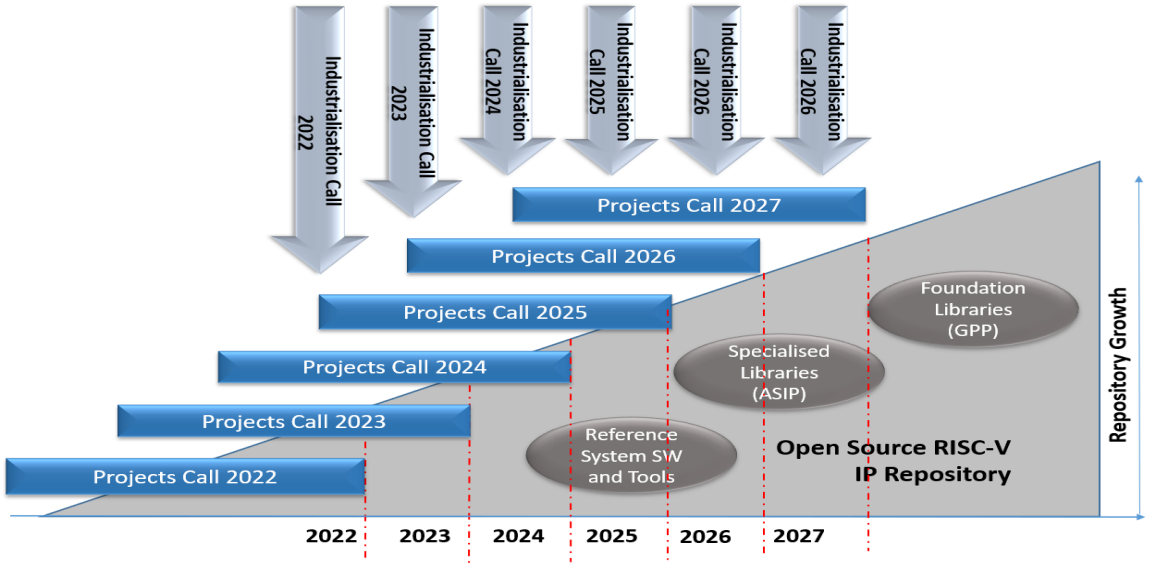


Figure *12* - Coordination of Horizonal and Vertical Calls to Create an Open Source RISC-V IP Repository

A possible approach to the coordination of the horizontal and vertical activities is shown in Figure 12 which shows the coordination of project calls addressing both the horizontal aspects as well as application driven vertical calls. In this approach new tools, foundation and specialized IP libraries will be created that are open to the ecosystem. These can be built upon, reused and exploited in future horizontal and vertical calls systematically growing a repository which will be of benefit to European stakeholders.

In parallel with this there is a need for a strategic "**Governance Initiative**" which should be launched as early as possible that orchestrates the EU activities in a top-down manner (to avoid fragmentation), maintains and curates the repository, and promotes the repository to both the open source community and industry. The governance initiative should also define the strategic roadmap of activities, how they interrelate and define expected achievements with milestones. An explicit aim is to ensure multi-year continuity, for instance by reviewing as well as potentially amending submitted research project proposals from a long-time perspective, and by supporting the refinement and definition of future calls.

The governance initiative should be sufficiently supported to be able to technically as well as administratively provide an optimal governance, that takes into account the special needs and characteristics of open source hardware, e.g., with regards to licenses, its economics across different design and production stages, etc. It also can optimize and monitor the submissions, and outcomes of calls, e.g., supervise a study on economic and social aspects of open source hardware innovation. It may additionally have responsibility to maintain the repository and create visibility. The governance initiative could be implemented by an existing organisation, by an agreement between existing organisations, or by a new organisation. The governance initiative should be backed by industry, academia, and users. Members of the working group that wrote this report (i.e., the Open Source Hardware & Software Working Group), are willing to support its initial setup, e.g., by widening the working group and eventually via exchanging with other organisations such as DARPA initiatives on open source hardware.

Already work has begun on developing the Open Source strategy with the launch of the *Call 2021-1-IA - Focus Topic 1: Development of open-source RISC-V building blocks* with funding of EUR 20 million from Horizon Europe. This addresses mainly horizontal foundational work on RISC-V IP cores, methodologies and tools as well as system software. A further call was launched in 2022 (2022-1-IA- Focus Topic 3)

addressing the *Design of Customizable and Domain Specific Open-source RISC-V Processors*. This call is supported by both national and EUR 20 million of Horizon Europe funding and addresses vertical aspects in terms of specialized processors, e.g., for automotive. The expectation is that the horizontal activities will contribute to the open-source RISC-V IP repository, which is shared and reusable, and the vertical activities will provide domain-specific cores, ready for industrialization which can be either shared or proprietary.

11 CONCLUDING REMARKS

This report highlights the need for a strong European open source ecosystem to drive competitiveness and enable greater and more agile innovation. There are increasing concerns over security and safety in application markets such as automotive, industrial automation, communications, health and defense where there is a reliance on non-EU developed technologies. Notably open source can be used as a means of retaining sovereignty when the only alternative is to license IPs from non-EU 3rd parties. This is only possible if there is a critical mass of European contributors to open source projects so that a European fork is possible (i.e., create a fully equivalent variant of a given technology) if necessary.

A key message is that there is a need to build or take part in sustainable open source communities such as OpenHW Group, CHIPS Alliance, etc. This is important to get and maintain industrial-grade solutions. Care must be taken to avoid fragmentation by creating too many communities. A challenge is that the current communities are young and deliver limited processor cores and toolchains. There is a need to extend the offer to the community with high-end processors, interconnects, peripherals, accelerators, Operating Systems and SW stacks, Integrated Development Environments (IDE)/Software Development Kit (SDK) with supporting documentation.

A major issue is the availability of Open Source EDA and CAD tools. In addition to open source hardware targeting ASIC implementations there is a need for development tools that do not incur significant licensing costs. Here high-quality open source EDA tools are needed for industrial-grade open source IP cores. A challenge is that Europe has a low footprint in the world of CAD tools, which are critical assets to design and deliver electronics solutions, but there are open source CAD tool initiatives which may bridge this gap.

Considering processors, the involvement in RISC-V International should be strongly encouraged to influence and comply with the “root” specifications. Currently, RISC-V standardizes the instruction set, but additional fields will be needed in the future for more interoperability considering SoC interconnect, chiplet interfaces, turnkey trust and cybersecurity, safety primitives, etc. At a European level it is important to support open source cooperative funded projects to maximize collaborations and create a significant European critical mass to compete with China and the USA. The goal should be to ensure that intellectual property produced is delivered as open source so that European actors can exploit these results.

For open source to be broadly adopted by the European ecosystem, a number of barriers need to be addressed. In particular, there is a perception that building defendable intellectual property is difficult, if not impossible, with open source. This is a key challenge for obtaining financing for open source based start-ups. This requires a redefinition of the business model for many European IP providers, design service providers and public research centers. EU advocacy and financial support is needed for start-ups, SMEs and public research centers to encourage the adoption of open source. At the same time public endorsement of open source and material contribution to open source projects from the leading European semiconductor vendors is critical to provide credibility and momentum to open source at European level. The European design service vendors can also play a key role in supporting the open sourcing of major vendors’ IPs, with financial support from the EU.

In order to meet the needs of applications there is a need to address a number of cross cutting issues such as scalability, certification for safety in different application domains, and security. This requires consideration at both the component level and system level.

Key Topics

A number of key topics which need addressing are highlighted in the report. These include “open hardware base building blocks”, “verified open source hardware & interconnect” “chiplets and modular interposers”, and EDA-tools. More specifically work on open hardware base building blocks should target highlighting benefits of open hardware through proprietary demonstrations (including software, other hardware, mechanical systems) to push the acceptance of open hardware. Considering processors, verified open source hardware should consider leveraging existing open hardware design IP suitable for high-assurance verification and the development of a re-usable verification infrastructure, which includes a complete verification plan considering module level verification, silicon technology specific verification and system integration verification including safety, security and other non-functional aspects (e.g., code quality, clock domain crossing, reset domain crossing, power domain crossing, resource efficiency, power efficiency). This should also consider IP blocks accompanied by verification frameworks for safety/security starting from ESL down to RTL. The aim here would be to release these frameworks and tests under permissive or weakly reciprocal licenses. The use of open source verification technology should be encouraged but leveraging non-open source verification technologies (e.g., simulation tools, formal verification tools, HW emulators) is also possible, as long as the output of the tools can be integrated and checked with open source verification environments. Going a step further to gain acceptance of open source it would be beneficial to demonstrate certification according to relevant safety and security standards with use cases/demonstrations. This should also include verified open source real-time interconnects, considering mixed-criticality and security.

Chiplets and interposers for modular architectures have been highlighted as an opportunity for the semiconductor industry and here there is a need to address the technical challenges such as architecture partitioning, chiplet-to-chiplet interconnect, testability, CAD tools and flows, advanced packaging technologies. Die-to-Die (D2D) communication, in particular, is the “missing link” to leverage the chiplet-based design ecosystem, and its development in open source would enable a wide use of the approach and could become a “de-facto” standard. This needs to target a high-bandwidth, low-latency, low-energy, ultra-short-reach link between dies.

Open source ASIC development tools are a necessary element of the hardware ecosystem, enabling unlimited access to experimentation and lowering barriers to implementing new ideas by all actors, including new players with a more software background. The availability of at least some open source components that could be used for ASIC design will allow for increased automation, new developer-focused workflows and open the door for cloud- and AI-assisted EDA design.

Finally, other important topics that need addressing are the economics and social aspects of open source hardware innovation as the approach is a significant disruptive change from current practice.

To achieve all of these aims a strategy based on addressing both horizontal foundational aspects of open source development and targeted activities addressing vertical sectors is proposed that will contribute to the creation of a European open source repository of IP blocks and tools.

ANNEX A – Open Source – what, why, how

Benefits of Open Source within the Value Chain

The ecosystem is wide ranging and diverse including the semiconductor industry, verticals and system integrators, SMEs, service providers, CAD tools providers, open source communities, academics and research. It is an area with many opportunities to create innovative start-ups and service offers. The benefits and attraction of adoption of open source depends on the type of actor and their role within the value chain. These can include:

- Creating innovative products with lower costs and access barriers.
- Providing a faster path to innovation and smoother cooperation between actors (academic, research, industry, SME, alliances) as no Non-Disclosure Agreement (NDA) and commercial license need to be negotiated.
- Influencing technical choices and specifications.
- Allowing customization of open source IP to user needs, delivering differentiating products.
- Sharing development costs.
- Reducing risks related to third-party IP (unbalanced commercial relationship, end of maintenance/ discontinued products, export control and trade wars).
- For RISC-V compliant processors, the advantage of the open source SW ecosystem (compilers, debuggers, Operating Systems, frameworks, etc.).
- Building support and design service businesses based on open source IP.
- Conducting research with easier access to digital technologies.
- Using open source material to educate talented students who will fuel future European successes.
- Better auditing of security and safety, ensuring that solutions can be fully audited and checked/verified (e.g., possibility to look for back doors in open source IPs). This is not possible for IPs licensed from 3rd parties.

The Open Source Ecosystem

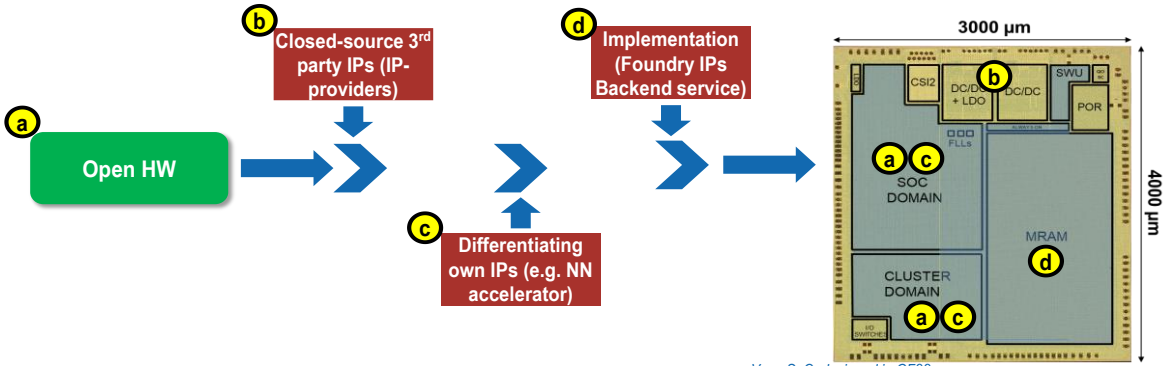


Figure *13* **- Example Flow of IP Creation of Open Hardware**

Open hardware IP is part of an ecosystem composed of other IPs (open source or not), tools to design, simulate, estimate, integrate, synthesize and validate the integration of all IPs (as shown conceptually in Figure 13) as well as PDK tools to translate the RTL into a low-level representation specific to the

foundry. The foundry utilizes the file generated (generally in GDSII/OASIS format) to produce the masks and the technology process. This is then used to produce a wafer from which the dies are extracted, tested, and packaged, before the final testing and the integration into the final electronic board. At each step significant financial investment is needed.

Although it is generally accepted that high-quality industry ready open source IP that has been through extensive testing, validation and documentation has a lot of value, **it is also believed that there is even greater value from using the RISC-V and Open Source HW/SW IP in the community rather than from the IP itself**. There are different levels of ''Open Source" IP, ranging from standard peripherals to complex application-specific multi-processor-based IP. Business can be made from all types of IP, dependent on the markets addressed, the customers involved and the application needs. Ongoing research, such as by Professor Nagle[52], illustrates the economic benefits of contributing to open source at the state level[53], which would become even more compelling if done at the EU level. The RISC-V ecosystem has the open RISC-V ISA as common starting point for flexibility and interoperability. **The openness of RISC-V also allows inspection which is important for hardware auditability.** Hardware needs to be trusted by consumers and experts alike. This was emphasized by the occurrence of security vulnerabilities such as Meltdown and Spectre which only became visible in 2019 after being deployed in billions of x86 and Arm devices for many years. Open hardware and RISC-V offer a very good chance to improve the situation and the wide acceptance of the RISC-V ISA also allows some degree of interoperability among devices.

Key players in Open Source

Business can be made on all types of IP, dependent on the markets addressed, the customers involved and the application needs.

The attraction of open source is clear and there is a growing community of developers and contributors to open source repositories both within Europe and at the global scale. A danger is fragmentation of effort and there is a need to strengthen European activities to create a critical mass that can contribute both at the European level, but also on the world stage. This is to ensure that European needs are serviced and that there is more resilience to forks in development that may be deleterious to European interests. Key initiatives identified by the Working Group are described in Table 3.

RISC-V International is probably one of the most well-known players in open hardware, but it only works on the RISC-V Instruction Set Architecture (ISA) and related specifications. Many entities and individuals have designed RISC-V compatible cores, processors and SoC (https://riscv.org/exchange/cores-socs/). Among them, there are two well-known organizations that host open source co-operations: CHIPS Alliance and OpenHW Group. lowRISC is also notable being mostly recognized for the OpenTitan project.

OpenHW Group is a not-for-profit, global organization driven by its members and individual contributors where hardware and software designers collaborate in the development of open source cores, related IP, tools and software. OpenHW provides an infrastructure for hosting high-quality open source HW developments in line with industry best practices and builds upon the ETH-

[52] https://www.hbs.edu/faculty/Pages/profile.aspx?facId=566431
[53] https://www.hbs.edu/ris/Publication%20Files/19-103_70f212c8-c4fe-4989-ac99-e03cf8bbf02d.pdf

Zurich/University of Bologna PULP open source project. A good fraction of its 60+ members are from the EU (companies and research centers), some being involved in its governance. The Group hosts the development of several RISC-V cores, CV32E4 embedded family deriving from RI5CY and CVA6 application cores deriving from ARIANE and will extend with TAIGA (from Simon Fraser University) and CV32E20 (compact core deriving from Ibex/Zero-riscy). SoC-level projects are also in the pipeline, as well as several software projects relating to the cores. The list and status of projects can be found at

https://github.com/openhwgroup/core-v-docs/blob/master/program/dashboard/Dashboard.md

CHIPS Alliance is pursuing a vision of creating a completely open hardware ecosystem, with tools, IP, cores, interconnects and more. As a Linux Foundation project it is based in USA but includes parties from all over the world, including EU companies like Antmicro. CHIPS should be viewed as a key strategic partner in pushing the open source technologies to the market. CHIPS Alliance is developing a number of improvements in the open tools workflow, verification, standardization, chiplets and other fields, including:

- SystemVerilog parsers, formatters, linters and other tooling – Verible, Surelog, UHDM, sv-tools (led by Google & Antmicro)
- Open source Universal Verification Methodology (UVM) support in Verilator (led by Antmicro & Western Digital)
- The OpenROAD/OpenLANE and OpenFASoC Application Specific Integrated Circuit (ASIC) design flows (led by Precision Innovation UCSD/UMich, etc.)
- A number of open source Analog tools (with diverse university involvement)
- The RISC-V Domain Validation (DV) verification environment (led by Google)
- The Chisel Hardware Description Language (HDL) and related tools (led by SiFive / Berkeley)
- Advanced Interface Bus (AIB) open source chiplet standard and reference implementation (spearheaded by Intel)
- Field Programmable Gate Array (FPGA) Interchange format, an interoperability standard for open and closed source FPGA tooling which can be used as a reference for similar activities in the ASIC domain (Antmicro / Google)

SkyWater technologies are pursuing a new, open foundry model where their 130nm Process Design Kit (PDK) has been open sourced together with Google, E-Fabless, Antmicro and a number of university partners in a crowdsourcing design platform leveraging also open source IPs and EDA tools. Other PDKs are likely to follow suit. The 3-some partnership offers free access to MPW (Multi Project Wafers) sponsored by Google. SkyWater is considered to be quite US-Centric.

EUROPRACTICE is a highly effective integrated service for Europe, that has been supported by the European Commission for more than 25 years. EUROPRACTICE lowers the entry-barrier for academic institutions and less well-established companies (i.e., start-ups and other SMEs) by offering affordable access to a rich portfolio of industrial-grade design tools and prototyping technologies for customized ASICs, Micro-electromechanical Systems (MEMS) and photonics. The service includes initial advice, training and ongoing support, reduced entry costs and a clear route to chip manufacture and product supply.

FOSSi: The Free and Open Source Silicon (FOSSi) Foundation predates the OpenHW Group and the CHIPS Alliance, with a strong European presence on its Board of Directors. It is completely different in that no big corporate actors are involved in its governance. Its legitimacy in the open source silicon arena is derived from their long-standing commitment to facilitate the sharing of open source digital designs and their related ecosystems. In this regard, they are well known for organizing the ORConf and LatchUp conference series, as well as managing librecores.org, a place on the Web to share HDL code, seen as the evolution of the old opencores.org.

Table *3* - Key Open Source Initiatives

Going forward the Working Group advocates for building upon and consolidation of Open Source Hardware (OSH) communities where there is a significant participation of European actors, e.g., the OpenHW Group, which is a worldwide initiative, and the CHIPS Alliance, which has a strong US footprint. Notably OpenHW is registered in Canada (a more neutral country) and is in the process of creating OpenHW Europe, as a working group hosted by the Eclipse Foundation (registered in Belgium). The OpenHW Group pays strong attention to verification (processes, environment, coverage, etc.) and open source availability of most verification artefacts. There is also a focus on designs written in widespread Hardware Description Languages (HDLs) to ease adoption by many teams and a rule not to include technology subject to export control laws in any jurisdiction. The CHIPS Alliance provides governance, structure and legal assistance including patents and export controls offering the opportunity for collaboration on legal aspects of open hardware.

Key in Europe is EUROPRACTICE and this already offers significant support for open hardware ICs through a design sharing agreement, which is a legal agreement that requires EUROPRACTICE members to share the IP that they have developed with other members. The standard academic license agreements for IC design tools preclude the sharing of IP, so this design sharing agreement is absolutely essential for any serious open source hardware endeavour. **EUROPRACTICE could extend its current offering and organize an IP exchange system, where academic members could submit their IC IP and gain access to all of the member contributed IP by signing a single agreement to join this 'club' rather than concluding a design sharing agreement for each individual piece of IP. Routes to commercialization from this pool would also exist. Furthermore, EUROPRACTICE could negotiate with its European foundries (such as GlobalFoundries, STMicroelectronics and X-FAB) to organize low-cost themed design challenges, similar to the Google-Skywater sponsorship**. Ideally, this should fuel hardware exchange between innovators (in the form of chiplets), which would require a standard interposer offer on which chiplets can be tested and brought together into a system.

Business Strategies

There are a number of different business models for commercializing Open Source[54]:

- Service and support firms
- Single-vendor open source firms
- Open source distributor firms

These can be driven by different business models or motivations:

[54] Prof. Dr. Dirk Riehle, Univ. Erlangen, Bayave GmbH, „Commercial Open Source", OFA Symposium, 16 Nov. 2021.

- **Aggressive**: customizing Open Source HW to provide differentiation, e.g., with respect to current commercial cores (e.g., Arm) and targeting advanced processing nodes.
- **Cost-sensitive**: using Open Source HW ''as is'' to reduce cost with respect to licensing costs (e.g., from Arm), or with respect to in-house effort, targeting older processing nodes.
- **Efficiency increase**: design priority setting to re-use standard blocks and focus on the application-specific differentiating elements in the SoC-design, in order to get to the market faster.
- **Share development costs:** share costs for pools of open source IP blocks between actors and manage joint projects in communities, such as the OpenHW Group, the Eclipse Foundation, etc.

The creation of value and the path to real chip production differs according to the types of users of the Open Source IP, such as IDMs, IP vendors, fabless companies, foundries, service companies, etc. From the perspective of a private company, one example on how to address Open Source is the following:

- Commit with academic partners to open source some IPs (theirs or their partners') at the latest when they go into production. This provides them with a time advantage where they think it is critical, i.e., for differentiating features, in an industry where competitive advantage that is driven by timing. Those IPs then enrich the Open Source ecosystem.
- Most of the constituents of an SoC are not differentiating. They are basic constituents that can be easily reused from one design to another. Having a rich open source ecosystem allows companies to focus their efforts on the differentiating features.

Licensing models for Open Source

A key aspect of open source business models is the licensing model and Open Hardware projects typically also include extensive documentation and software for simulation, validation and testing.

Permissive licensing is a mechanism whereby the rights holder gives permissions and imposes very few and lightweight obligations in exchange. For example, one obligation might be to explicitly mention the original author in a piece of modified work. The piece of modified work, and any larger work in which it is included, do not need to be released under the same license.
Weakly-reciprocal licensing, whereby a licensee who modifies the work must release the modified version under the same license. This is where reciprocity comes in. The licensee gets a piece of work with a number of permissions, but must share back in exchange, therefore contributing to a virtuous circle of sharing. The "weak" in "weakly reciprocal" refers to the fact that the reciprocity obligations do not extend beyond the initial piece of work that was shared. The original piece of work can be embedded in a larger work and the sources for that larger work do not need to be released under an open source license.
Strongly-reciprocal licensing is a sharing regime whereby the obligations to share back extend to any larger piece of work embedding the originally licensed work.
Documentation Licensing - When considering documentation, the de-facto standard for sharing is the Creative Commons family of licenses. These include CC0 (permissive with no attribution requirement), CC-BY (permissive with an obligation to recognize the original authors) and CC-BY-SA (reciprocal). These three licenses are adequate to share documentation in the context of the open hardware developments.

Table *4* - Key Licensing Approaches

It is important to choose appropriate licenses for each of the released components. Copyright law is "all rights reserved" by default. This means that putting a file somewhere on the web is not enough because the recipients of that file do not have the basic permissions they need to allow copying, modification and publishing of modified versions of that file. Licenses provide the permissions that the rights holder grants to users so they can benefit from and contribute to the open source effort. There are several pros and cons to different licensing approaches. For software, the best choice of license depends on the project. Apache v2 is a modern permissive license which is used in many successful open source software projects. GNU Lesser General Public License (LGPL v3) and Mozilla Public License (MPL2) are also good choices in weakly-reciprocal contexts, and GNU General Public License (GPL v3) is the de-facto standard for strongly reciprocal work. When considering hardware and IC design there are some issues with using open source software licenses. This applies mainly to reciprocal regimes and permissive licensing is a much simpler in practice. As such Apache v2 can be used as a permissive license for hardware projects in the IC design realm with alternatives being Solderpad v2.1, which takes Apache v2 as a basis and modifies some terms to better adapt them to a hardware context. More specifically the CERN Open Hardware License comes in three variants: permissive, weakly-reciprocal and strongly-reciprocal, providing a one stop shop for hardware licensing. The reciprocal variants take into account the specificities of IC design and its legal and commercial environment.

Approaches to licensing in Europe

Past efforts show that, **in order to reap the benefits of open source, it is vitally important to communicate clearly from the beginning the line between open source and proprietary for a given project/initiative (and how they complement in certain cases), as well as to stick to standard commonly agreed practices and definitions within the open source realm**. In practice this is a very challenging endeavor, so it important to agree on some principles. Open source provides a more level playing field for all actors, and this is an integral part of its success. Many people have tried to get one thing without the other and failed as they both go together. There are ways to give industry a competitive edge inside an open source ecosystem, but it should be understood that preferential access to the source for a component automatically makes that component proprietary. Existing licensing terms include:

- **Truly open source** (i.e., a license approved by OSI). This should already ensure it is not discriminatory to the EU or any other state or group of states.
- **Meaningful for Open Hardware**. Some OSI-approved licenses which were drafted for software are difficult to interpret in the hardware case, which can be very confusing and bring legal uncertainty.
- **Permissive or weakly-reciprocal**

Another approach could be to use exclusive licenses for EU projects. However, as the whole world is embracing the use of RISC-V and both open as well as proprietary IP, there is no need for a special or separate license model for EU purposes only. Despite this there is still a need to select the right licenses using European laws. **Licenses need to be carefully selected to avoid potential contamination or being used for restricting usages for European companies**. Export control laws can restrict the use even of Open Source in some cases, for example, in case of not (yet) fully public disclosure. Some countries'

export control regulations, such as the United States, may require taking additional steps to ensure that an open source project is satisfying obligations under local regulations[55].

In many cases the value of Open Source IP is questioned by industry for critical designs. **Open source is considered a risk because of lack of available support and because of the fact that it is open towards the whole world, which could make it difficult to differentiate with competitors.** As Europe is lagging behind today in some domains, it can remain a net beneficiary of open source projects for years, provided that it builds the community and the infrastructure capable of leveraging and influencing those projects. EU funding can contribute to this.

The other big family of relevant actors is that of public institutions. In these institutions there is an ongoing debate as to how much of their work should be made open source. The fear that "giving everything away" to competitors will harm the economy is largely based on an incomplete understanding of best practices to combine open source and commercial activity. One reason why groups in public institutions do not share much publicly is the perception that the results of research should benefit exclusively the subset of people who have financed it. For national and international laboratories, one often hears the objection in terms of the problem of offering all our knowledge and technology for free to others in the world. One possible line of reasoning is that this knowledge can then be used by these external actors to harm our economy, by letting 'them' cheaply manufacture what companies in the financing states could have offered for sale, using profit margins to properly pay their employees and reinvest in innovation. This reasoning is incomplete and can lead to decisions which forego important opportunities to benefit our societies. The opening of the RISC-V Instruction Set Architecture (ISA) itself is a good example of a strategic move from a country (the US) in which the open sourcing was crucial in the generation of large benefits for that country. **RISC-V created a new market because it is open and the main beneficiaries of that new market were institutions and companies which were physically and strategically close to the originating university.**

There is a fear of giving everything away to competitors. This is based on an incomplete understanding of best practices to combine open source with commercial activity.

[55] https://www.linuxfoundation.org/tools/understanding-us-export-controls-with-open-source-projects/

ANNEX B – Market trends in the chip market

RISC-V and OpenSource HW/SW IP can be applied in strategic SoC designs targeting specific application domains, thanks to an extensible Instruction Set Architecture (ISA). **The value of the global chips market in 2021 was around USD $550 billion.**[56] The bulk of global demand comes today from end-use applications in computing, including PCs and data centre infrastructure (32%), communications, including mobile handsets and network infrastructure (31%), and consumer electronics (12%). **The growth rate is highest, however, in segments previously ruled by analogue and mechanical technology, such as automotive and industrial manufacturing (12% each)**, as shown in Figure 14[57].

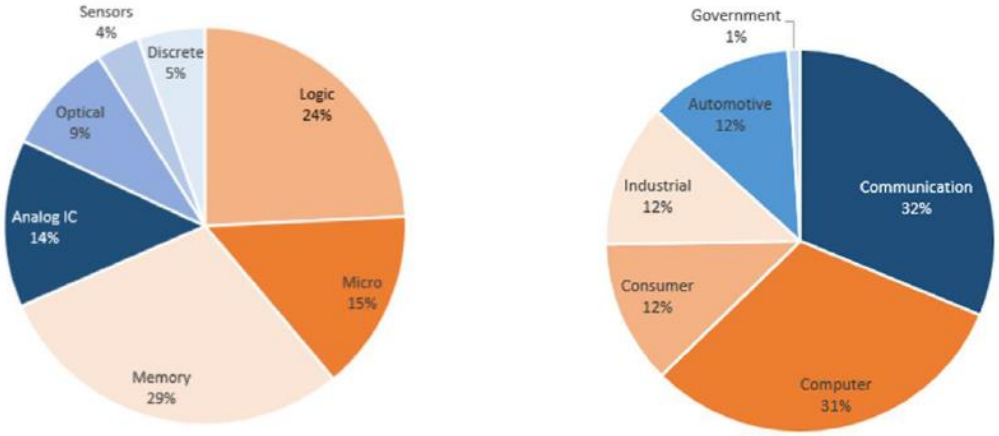


Figure *14* - Semiconductor Market Segments, by Device Type and by End-user Sector Demand

Due to strong industry sectors within Europe the split across different domains differs from the worldwide situation with the automotive semiconductors market being 37% of the total European market, the industrial market being 25%, and less of a presence in the computing (15%), communication (15%) and consumer markets (7%)[58]. Overall, the European Semiconductor Industry Association (ESIA) reported that yearly **semiconductor sales in the European market reached USD $ 47.757 Bn in 2021, a 27.3% increase from 2020 and a 27% increase considering the same month in 2020**. At the global scale semiconductor sales in 2021 amounted to USD $ 555.893 Bn, a 26.2% increase from 2020. All figures are based on World Semiconductor Trade Statistics (WSTS) and represent a three-month rolling average unless otherwise indicated. In the table and the graph shown in Figure 15 this growth in European semiconductor sales is clearly shown[59].

[56] Ravi, Sarah. "Global Semiconductor Sales, Units Shipped Reach All-Time Highs in 2021 as Industry Ramps Up Production Amid Shortage." Semiconductor Industry Association, February 14, 2022. https://www.semiconductors.org/global-semiconductor-sales-units-shipped-reach-all-time-highs-in-2021-as-industry-ramps-up-production-amid-shortage/

[57] Semiconductor Industry Association Factbook 2021 (figures for 2020)

[58] https://www.eusemiconductors.eu/esia

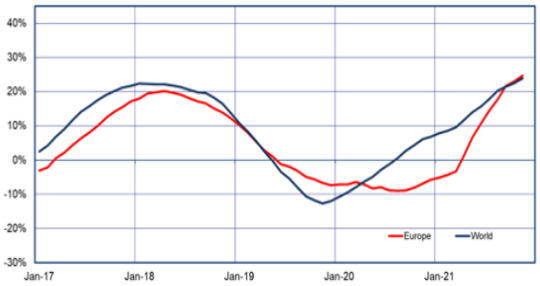
[59] ESIA Release Note, 18 February 2022

Monthly European semiconductor sales

(3-month-average data, except YTD growth which is calculated based on current month data)

Market data for the 3 month moving average ending:								
Region	sales (in billions)		Month on Month growth		Year on Year growth		YTD growth	
	Oct 21	Nov 21	Oct 21	Nov 21	Oct 21	Nov 21	Oct 21	Nov 21
in $:								
Europe	4.140	4.267	2.7%	3.1%	27.8%	26.3%	26.5%	26.4%
Americas	11.027	11.490	2.3%	4.2%	29.7%	28.7%	23.2%	24.4%
Japan	3.890	3.933	1.3%	1.1%	24.7%	19.5%	20.1%	19.3%
Asia Pacific	29.913	29.999	0.3%	0.3%	22.1%	21.7%	25.9%	26.3%
of which China	16.895	16.869	0.2%	-0.2%	22.0%	21.4%	26.1%	26.6%
World	48.971	49.690	1.0%	1.5%	24.4%	23.5%	24.9%	25.3%
In EURO:								
Europe	3.531	3.677	3.3%	4.1%	28.6%	28.4%	19.7%	20.8%
Rate ($/Euro)	1.163	1.142	-1.3%	-3.5%	< Euro against $ versus prev. Year			

Semiconductor sales worldwide and in Europe (in US Dollars)



Note: On a 12/12 basis, i.e. on the basis of the percent change of a 12-month period compared to a similar period twelve months before

Figure *15* - Monthly European Semiconductor Sales and Worldwide Growth in Semiconductor Sales

The global Microprocessor market was estimated to be almost USD $80Bn in 2020 with the main markets being in computing and communication as shown in **Figure *16***. **Europe has a strong position in the embedded market and this rapidly growing market is estimated to account for 21% of sales**[60] .

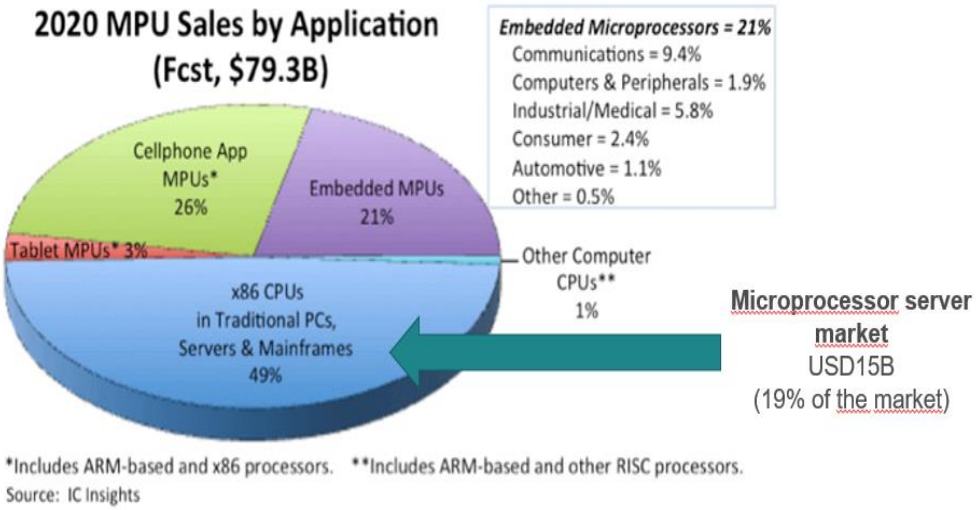


Figure *16* - Microprocessor Sales by Application

This is important for Europe as with a global output of more than €1,000 billion, Electronic Systems for embedded and professional applications have overtaken, in production value in 2018, the amount dedicated to the traditional stand-alone electronic goods encompassing smartphones, PCs, TVs, etc. (see Figure 17[61]).

[60] https://epsnews.com/2020/09/14/total-microprocessor-sales-to-edge-slightly-higher-in-2020/

[61] "Creating Intelligent Digital Systems Together", A study commissioned by the INSIDE Industry Association (INSIDE-IA), 24 November 2021

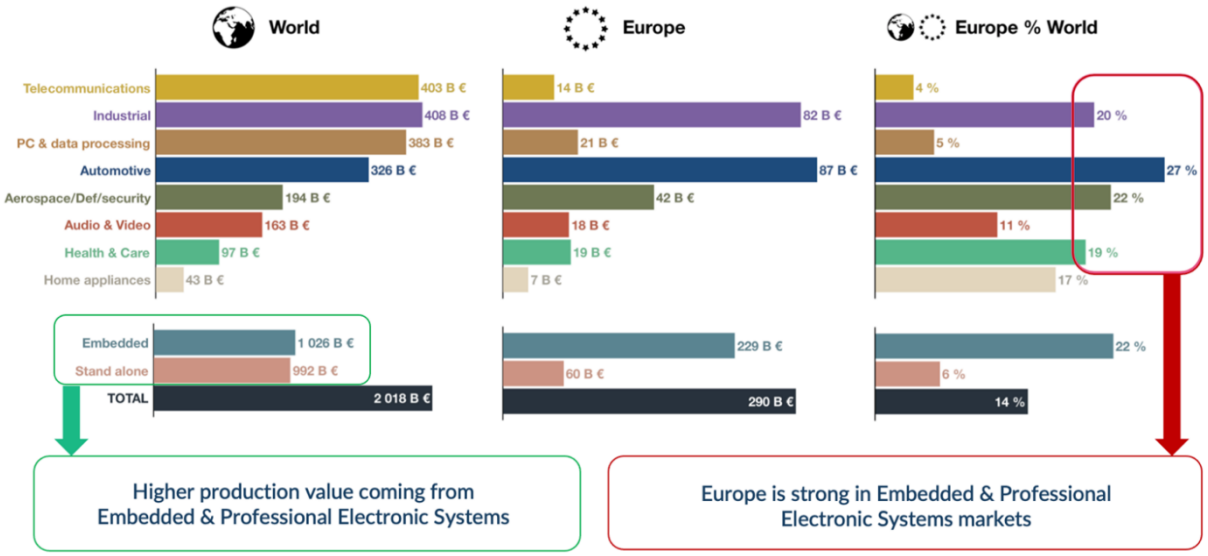


Figure *17* **- World & Europe Production of Electronics Systems by Application Domain**

Members of the Working Group

Chair :

Patrick Pype	NXP Semiconductors

Participants:

Jan Andersson	Cobham Gaisler
Luca Benini	ETH Zürich / Univ. Bologna
Sven Beyer	Siemens
Holger Blasum	SYSGO GmbH
Sylvain Depierre	Nano-explorer
Marc Duranton	CEA
Wolfgang Ecker	Infineon
Michael Gielda	Antmicro
Edwin Hakkennes	Technolution
Andreas Koch	Technische Universität Darmstadt
Loic Lietar	Green Waves
Andreas Mauderer	Bosch
Jan-Hendrik Oetjens	Bosch
Jérôme Quévremont	Thales
John Round	NXP Semiconductors
Javier Serrano	CERN
Herbert Taucher	Siemens
Jean-Marc Talbot	Siemens

EC-Representatives:

Georgi Kuzmanov	KDT JU
Panos Tsarchopoulos	DG CONNECT
Arian Zwegers	DG CONNECT

Editor:

Haydn Thompson	THHINK