TRISTAN

Together for RISc-V Technology and ApplicatioNs

DELIVERABLE REPORT

Initial requirements and feedback for processor and hardware IPs

Document Number	D1.1
Primary Author(s)	Sara Bocchio
Document Date	**25.10.2023**
Document Version / Status	Final; 1.3
Distribution Level	Public
Reference DoA	December 2022
---	---
Project Coordinator	Patrick Pype, NXP Semiconductors, patrick.pype@nxp.com
Project Website	www.tristan-project.eu
JU Grant Agreement Number	101095947

TRISTAN has received funding from the Key Digital Technologies Joint Undertaking (KDT JU) under grant agreement nr. 101095947. The KDT JU receives support from the European Union's Horizon Europe's research and innovation programme and Austria, Belgium, Bulgaria, Croatia, Cyprus, Czechia, Germany, Denmark, Estonia, Greece, Spain, Finland, France, Hungary, Ireland, Israel, Iceland, Italy, Lithuania, Luxembourg, Latvia, Malta, Netherlands, Norway, Poland, Portugal, Romania, Sweden, Slovenia, Slovakia, Turkey

CONTRIBUTORS

Name	Organization	Name	Organization
Sara Bocchio	ST-I		

FORMAL REVIEWERS

Name	Organization	Date
Rob Wullems	NXP	6.6.2023
TRISTAN PSB	-	9.6.2023
WP1 contributors		3.6.2023

DOCUMENT HISTORY

Revision	Date	Author / Organization	Description
0.1	2/11/2023	Sara Bocchio	Initial document Content
1.2	10/10/2023	Sara Bocchio	Removed confidential information from this public report
1.3	25/10/2-23	Sara Bocchio	Removed confidential information from this public report

1 Executive summary

This deliverable reports the results of the requirements elicitation and of the Working Items definition, describing the activities carried on Task 1.2 during the first three months. The deliverable defines the concept of requirement, illustrates the process adopted for the requirements elicitation, reports the set of requirements collected following this process and provides some statistics on them.

The requirements elicitation will be further extended, refined and updated, and the new results of these activities will be presented in Deliverable D1.5 at month 18.

2 Table of Contents

3 Introduction

This deliverable reports the results of the requirements for processors and hardware IPs, describing the activities carried on in Task 1.2 during the first three months. The deliverable defines the concept of requirement, illustrates the process adopted for the requirements elicitation, reports the set of requirements collected following this process and provides some statistics on them.
The requirements will describe the relationships between the actual concrete results in the work packages and the objectives that they satisfy.
The following figure illustrates the timeline of the main deadlines related to the WP1 activity.

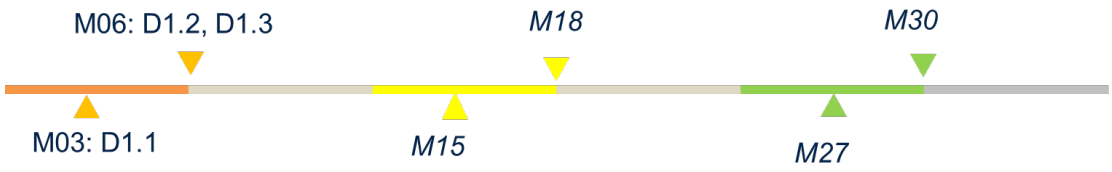

Figure 4 WP1 timeline.
The first set of deliverables (M3) describing the requirements contains the preliminary results of the study of the objectives of the different Ips block, a description that can be considered an initial reference point to that will further be extended, detailed, refined and eventually consolidated in the second iteration (M15), described in the next deliverables

In the second iteration (from M12 to M18) the processors and the Ips blocks were further clarified. WP2 and Wp3 will better define the specification of the different Ips and the correlations between the requirements and the specifications are defined in the WP1 second iterations. Moreover, an analysis of the different requirements coming from the other workpackages and the demonstrators will carried on in order to analyze the different correlations.
The final Iteration deals will target the validation of the requirements, by the definition of the means of validation and the validation results.

The requirements themselves can be found in the accompanying Excel files, where every working item has two dedicated sheets. The partners provide a brief description of the Working Item (hereafter referred as WI) and a first description of the relative requirements, in order to keep this deliverable a lean document and to let the consortium as a whole to manage in a easier way all the requirements.
Moreover, the excel files will serve as a living document during the project execution for internal reference across the different Workpackages to exchange information. WP1 will encourage the participants of TRISTAN project to read and exchange those elements that relate directly to project activities where they are involved. The details of the requirements and their assessments can influence the strategies and efforts in different parts of the project.

This deliverable is therefore composed by the three documents:

1. This document
2. RequirementD1.1_public.xlsx where the requirement of the open source working items of WP2 and WP3 are listed.
3. RequirementD1.1_confidential.xlsx where the requirement of the proprietary working items of WP2 and WP3 are listed. This file should be considered as a confidential annex of this public deliverable.

In the following table the Working Items of WP2 and Wp3 as listed in table 5 of the DoA are linked to the different excel files, as a reference

ID	WI and name	Stakeholders	File
	Cores (WP2)		
C:CVE2	**WI2.2.5:** CV32E20 https://github.com/RequirementD1.1_public.xlsx hwgroup/cv32e2	NXP-DE, ETH	RequirementD1.1_public.xlsx
C:CVE4	**WI2.3.2:** CV32E4 https://github.com/RequirementD1.1_public.xlsx hwgroup/cv32e40p	UNIBO, ECL, CEA	RequirementD1.1_public.xlsx
C:CVA6	**WI2.4.1:** CVA6: https://github.com/RequirementD1.1_public.xlsx hwgroup/cva6	TRT, TDIS, ETH, SYSGO	RequirementD1.1_public.xlsx
C:Ibex	**WI2.2.8, WI2.5.7:** "Ibex" core with MCB and V extensions	POLITO, SYNT	RequirementD1.1_confidential.xlsx
C:STIT	**WI2.2.4:** RISC-V core with vector extension for industrial applications (32imcv ISA)	ST-IT	RequirementD1.1_confidential.xlsx
C:TGC	**WI2.2.1:** TGC kernels https://github.com/Minres/TGC-VP	MNRS	RequirementD1.1_public.xlsx
C:FreNox	**WI2.2.7:** FreNox processor	UTW, TECHNL	RequirementD1.1_confidential.xlsx
C:VLSI	**WI2.3.5:** 32-bit RISC-V based on Bi-RISCV (https://github.com/kuopinghsu/biriscv)	VLSI	RequirementD1.1_public.xlsx
C:EMSA5-FS	**WI2.5.4:** ASIL-D certified embedded RISC-V core	FHG	RequirementD1.1_confidential.xlsx
	IPs and instruction set extensions (Task 2.5 and WP3): internal interfaces and pipeline		
I:CV-X-IF	**WI2.5.3:** Implementation of CV-X-IF interface	TDIS	RequirementD1.1_public.xlsx
I:SCAIE-V	**WI2.4.2, WI2.4.3:** Portable and scalable SCAIE-V extension interface	TUDA	RequirementD1.1_public.xlsx
I:NFC-DSP	WI2.5.1: Extension of the RISC-V ISA to improve performance of NFC specific DSP algorithms	NXP-AT, TUG, POLITO	RequirementD1.1_confidential.xlsx
T:RequirementD1.1_public.xlsx Asip	WI2.5.6: RequirementD1.1_public.xlsx ASIP support (RequirementD1.1_public.xlsx asip.org, also known as TTA-Based Co-design Environment).	TAU	RequirementD1.1_public.xlsx
	IPs and instruction set extensions (Task 2.5 and WP3): interrupt controllers/memory/interconnects		
I:SafetyIC	**WI3.3.1:** BOSCH-FR: Safety Interrupt Controller	BOSCH-FR	RequirementD1.1_public.xlsx

I:TinyFPU	**WI2.5.2:** Tiny FPU for CV32E20	NXP-DE, ETH, ECL	RequirementD1.1_public.xlsx
I:CLIC	**WI3.1.7:** Core Level Interrupt Controller (CLIC) as productization of the PULP baseline.	NXP-DE, ETH, ECL, NXP-FR	RequirementD1.1_public.xlsx
I:Smart-DMA	**WI3.1.4:** Low Power DMA with multi-bank memory access and performance enhancements.	UNIBO, ST-IT	RequirementD1.1_public.xlsx
I:Data-cache	**WI3.1.5:** Optimised Caches feature flexible strategies, multi-requestors, writeback control, support for error correction and performance stats etc. Applied to CVA6.	ETH, BOSCH-FR, CEA, ECL	RequirementD1.1_public.xlsx
I:LPDDR4X-DRAM-Intf	**WI3.2.3, WI3.2.5:** Ultra-Low Power DDR4X PHY Memory Interface and controller for ultra-low power RISC-V based SoCs (few tens of mW power budget)	GWT, ANTM, TUDA	RequirementD1.1_public.xlsx , but subject to foundry constraints
I:High-speed-serial-links	**WI3.2.4:** High-speed serial links (high-speed wireline transceiver IPs, PHY)	TAU, NOKIA	RequirementD1.1_public.xlsx , but subject to foundry constraints
I:AXI	**WI3.2.1:** High Performance AXI Interconnect building blocks supporting topology independence, modularity and heterogeneous networks.	ETH, BOSCH-DE, SYSGO	RequirementD1.1_public.xlsx RequirementD1.1_confidential.xls (BOSCH_DE) x
I:Heterogeneous-cluster-interconnect	**WI3.2.2:** Heterogeneous Cluster Interconnect between cores	UNIBO, BOSCH-DE, BOSCH-FR, SYSGO, CEA	RequirementD1.1_public.xlsx RequirementD1.1_confidential.xls (BOSCH-DE) x
I:OBI-interconnect	**WI3.2.6:** Low latency OBI interconnect as CV32E4 and CV32E2/IBEX processors cores use OBI native interfaces.	CEA	RequirementD1.1_public.xlsx
I:SoC	**W3.1.3:** SOC Core Building Blocks, including GPIO, UART, SPI, Interrupt controller, etc. CEA will work on quality improvement of the base building blocks to reach TRL7	CEA	RequirementD1.1_public.xlsx
I:Hypervisor	**WI2.5.10:** Hypervisor support for the application core (CVA6)	ETH, SYSGO	RequirementD1.1_public.xlsx
IPs and instruction set extensions (Task 2.5 and WP3): Application support/FPU/mathematics/NN			
I:Compression	**WI2.5.11:** Custom instructions for the CV32E40X core to improve the performance of decompression and compression of digital waveforms on the fly.	SEM	RequirementD1.1_confidential.xls x
I:ML-acceleration	**WI2.5.7**: ISA Extensions for low-bitwidth RequirementD1.1_confidential.xlsx - precision integer arithmetic.	UNIBO	RequirementD1.1_public.xlsx
I:RVV	**WI2.4.4:** RVV co-processor with support for low precision integer arithmetic and multi-precision floating point operations	ETH, UNIBO, ST-IT	RequirementD1.1_public.xlsx

I:Value-boundary-checking	**WI2.5.12:** Value boundary checking acceleration for automotive	TNR	RequirementD1.1_public.xlsx
I:Variable-prec-accelerator	**WI2.3.2:** Variable Precision Accelerator.	CEA	Dual/RequirementD1.1_confidential.xlsx
I:Low-power-FP-MatMul-accelerator	**WI2.2.4:** Low-power FP32/FP16 Matrix-Multiplication Accelerator for edge-oriented applications.	UNIBO, ST-IT	RequirementD1.1_public.xlsx
I:NN-accel	**WI2.5.9:** WI2.5.9: Extensions for AI native engines and custom arithmetic and control	BOSCH-FR, BOSCH-DE, TUDA	RequirementD1.1_confidential.xlsx (BOSCH), RequirementD1.1_public.xlsx (TUDA)
I:Embedded-fpga	**WI3.4.1:** Embedded FPGA for hardware acceleration with AXI interface.	YNGA	RequirementD1.1_public.xlsx
IPs and instruction set extensions (Task 2.5 and WP3): Observability, safety and security			
I:Side-channel	**WI3.3.2:** Side channel attack protection via hardware and software mechanisms.	ST-FR, ETH,	RequirementD1.1_public.xlsx
	WI2.1.1: support for timing channel protection: ETH	SYSGO, ST-IT, UTW	
I:Trace-unit	**WI2.1.4, WI2.2.1:** Portable trace unit for DBT RISE, TGC, EMSA5, CVA6	ACCT, FHG	RequirementD1.1_public.xlsx
I:Control-flow	**WI2.3.4:** Trace unit extension for live off-chip RISC-V control flow reconstruction.	ACCT, FHG	RequirementD1.1_confidential.xlsx
I:JTAG	**WI2.2.4, WI3.1.2:** ST-IT: JTAG IEEE 1149.1 Debug support unit development for area efficiency based on pulp_riscv_dbg IP	ST-IT	RequirementD1.1_public.xlsx
I:Postquantum	**WI3.3.4:** Accelerator for post-quantum cryptography	TUM, POLITO	RequirementD1.1_public.xlsx
I:Radhard	**WI2.1.2, WI3.3.3:** Components for support of Radiation Hardened RISC-V, including error detection and correction (Lock Step, External Memory Security, ECC memories etc)	TECHNL, UTW	RequirementD1.1_confidential.xlsx
I:Probabilistic	**WI.3.3.5:** Probabilistic data structures for safety and security for use in core checkers to detect hardware trojans and vulnerabilities	UTW, TECHNL	RequirementD1.1_confidential.xlsx
I:PMP-module	**WI3.1.6:** Low Granularity Physical Memory Protection (PMP) Module (PMP with core integration and a verification bench)	NXP-DE, ETH, ECL, NXP-FR	To be confirmed
“I: AUT_APP-module	**WI3.4.3:** BOSCH-DE will derive accelerator building blocks for automotive applications based on performance analysis. The output will be a specification as well as VP functional model	BOSCH-DE	RequirementD1.1_confidential.xlsx

4 Requirements definition and elicitation

This chapter describes the process for requirements elicitation held to collect requirements from the partners

4.1 The Role of the Requirements

First, we need to declare what we intend the requirements to mean and what role they should play in the project. A detailed requirements template has been established to get a comprehensive and uniform record of the operational goals at the start of the project.

The requirements will describe in detail the relationships between the actual concrete results in the work packages and the project objectives they satisfy. The excel table will in a compact way give an overview of the expected concrete outcomes to record and boost collaboration. Finally, the excel table will serve as a concrete checklist for achieving the project objects as results are ticked off.

4.2 Providing and Supplying Requirements

Obviously, a requirement cannot always be fulfilled by those that want the results of that requirement. Furthermore, requirements are not a wish list. Posing a requirement means that there is a need for what the requirement defines, and those that provide the requirement should be ready to help assess and explore the potential results that intend to supply the fulfilment of the requirement.

We have defined that project Working Items should be the providers of the requirements. Each Working Items provides an IP block in the project, as outlined in table 5 of the Tristan DoA and a working item comprises a unit of collaboration and discussion. The essence here is that the requirement constitutes a kind of binding between those in the project with explicit needs, and those in the project that can satisfy those needs. Typically, the demonstrators will be requirements providers and technical work packages will be the suppliers, but also the technical tasks can be providers.

4.3 The Granularity of Requirements

Requirements may come on many different abstraction levels and may relate to small details or larger modules. It is very difficult to know what is most useful, and what would suit TRISTAN project best. We also want to have a

manageable and fruitful set of requirements that most people in the project can relate to. What we have done to limit the number of requirements, and thereby control the abstraction level, is to ask each WI to provide around a "handful" of requirements. Some WIs have provided less than 5 requirements, while others have provided much more. We did not want to make the limit absolute in any way, it was just a soft way to indicate the nature of the requirements. In our first iteration we have ended up with around 140 requirements. There are areas that are better covered than others, and in our next iteration of the requirements in a year, we may see some modifications to the set, but we also want the requirements to be stable in order to be able to compare the project results with the first baseline requirements.

4.4 The Elicitation Process

The following list describes how the requirements have been created and consolidated:

- A template with a couple of example requirements was created. The template also contained instructions for how to fill them out. All of this was embedded in an Excel file. The template was discussed during the TRISTAN kick off.
- The first template was sent out for review to the members of Task 1.2 and others who might be interested. The template was modified slightly.
- The template was sent to all WPs leaders of the project and discussed during the PSB meeting in Munich of the project; further changes has been made to the template
- We initiate the consolidation phase, and the first sub-phase was direct by WP2 and WP3 leaders that set up a WP meeting to describe those requirements to all WPs partecipants (This is a very crucial step in the process as it is the first assessment of whether the requirement is understandable and realistic)
- Finally, the collection of requirements has been performed and reviewed by the coordinator.

4.5 The Requirements elements for D1.1

A Requirements Excel file is provided for every Working Item. The file is composed by two sheets: the first one contains a brief description of the Working Item, while the second one represents the requirements of the working items.

With the first sheet we intend to create a homogenous description of every working item, forcing the partners to provide the same set of information. In particular every Working Item description sheet provide the following information:

1. Description: A brief description of the Working Items
2. State of the art: a reference of what is already present as state of the art
3. Proposed solution: how the Working Item differentiates from the state of the art
4. Prerequisites: the baseline for the Working Item
5. Target TRL
6. Proposal Respondance: how the WI will contribute to reach Tristan Specific Objectives and KPI
7. List of generated objects: the concrete outcomes of the WI (RTL IP, FPGA prototype…)
8. Short description of the development flow: a brief description of development plan of the WI
9. Milestones: Internal WI milestones (typically the specification, RTL development and end of verification), possibly with a scheduling
10. Overview of verification/validation setup: a brief description of the verification and validation process. Since one of the key aspects of the Tristan Ips will be the TRL achieved we want to highlight how the TRL will be achieved through the verification and validation carried on.
11. Link at requirements file(s): at the second iterations, where the requirements of WP4, WP5 and WP6 are defined the different HW Ips can be linked to different demonstrators/tools/software Ips and relative requirements.

As an example, we show the brief description of WI3.2.1 that consists of a Debug Support Unit for RiSCV cores

Document Info	
Work Item	*WI 3.1.2 JTAG debugger*
Author(s)	*Sara Bocchio*
Version	*1.0*
Purpose (Use Case)	
Description	*The JTAG debugger will serve as debugger unit for 32 and 64 bits family of RiSCV core.*
State of the art	*The Ip will start from the riscv debugger developed in the pulp project https://github.com/pulp-platform/riscv-dbg*
Proposed solution	*The solutions will be fully verified in order to reach TRL6 and it will be used inside the industrial demonstrator. Moreover the final Ip will extend the existing functionality by providing a trigger module and the support of the abstract commands*

Prerequisites	*Initial evaluation of existing tests, testbenches of the IP*
Target TRL	*TRL6 minimum*
Proposal Respondance (SOs and KPIs)	*SO2 - Ecosystem of Industrial Quality SoC Building Blocks*
Objects List	
List of generated objects	*Revision of the riscv debugger specification, RTL and verification metrics.*
Development Flow	
Short description of development flow	*See the milestones*
Milestones	*M01 Revision of the riscv debugger specification and evaluation of existing tests in terms of coverage (M9)* *M02 RTL improved in functionality (M18)* *M03 RTL fully verified and prototyped in the industrial demonstrator (M27)*
Verification & Validation	
Overview of verification/validation setup	*Coverage metrics (code and functional) fault injection based qualification of the verification* *Prove in silicon (technology will be defined)*
Appendix	
Link at requirements file(s)	*<…>*

The requirement sheet has a number of columns while the rows represent the individual requirements. The columns are organized in groups, that are divided by years. The requirements excel files are meant to be carried on during all project duration, in order to track requirements during the project executions We describe the items that are defined for this first deliverable. Other fields are sketched out and described briefly in section xx. These fields are not fixed in the stone since they can be revised after a first analysis of the specifications.

4.5.1 Requirement id

a unique identifier to the requirement, that includes the Working Items in the id.

4.5.2 Requirement Description

Obviously the description of the requirement itself is of utmost importance. We have decided to let the provider himself/herself use natural language (English) to describe the requirement, the need that has been discovered. We also want the requirement to be motivated by a rationale, also to be given in natural language.

Then we ask the provider to suggest how the requirement should be assessed when it is supposedly satisfied. Closely associated with how it should be assessed is also who should supply the necessary technology to satisfy the requirement.

4.5.3 Priority

.The requirement can be mandatory, optional or at best effort (i.e. the partners that will satisfy the requirement will use their best efforts to take all actions and to do all things necessary, proper, or advisable to consumate, make effettive, and comply with all of the terms of requirements)

4.5.4 Requirement status

Requirements are in this initial phase all proposed but during the second phase of WP1, when the requirements will be consolidated, the requirement can be accepted as it is or rejected. In case that a requirement need to be changed, it will be rejected and a new update requirement will be submitted and accepted. This procedure is agreed among all TRISTAN partners: in such way any changes in the requirement will be evident.

4.5.5 Proposed by

The submitter's name are given.

4.6 Requirement template for Year 2 and Year 3

The consortium aims to sketch out from the beginning the possible requirement template for the future iterations: these can be still subject to changes and improvements: the purpose to having them since the beginning is also to facilitate the collection of feedback from all the consortium.

1. WI requirements interdependencies (cores, HW Ips, SW Ips, Demonstrators): Aims is to highlight interdependencies of the different WI/Demonstrator requirements once all requirements will be defined.
2. Eda Tool (commercial) Simple yes/no answer in order to highlight which IP will have a full open-source development flow
3. Tristan Tools: the HW Ips that will use tools developed in WP5 should explicitly mention it
4. Which need requirement? We refer to WP5 tools, HW Ips should mention if they need some specific requirements that will not have been required in D1.3 “Initial requirements and feedback for SW and EDA Tools Ips”
5. Related specification A list of specifications of the Ips that will define how the requirement will be implemented.
6. Means of validation A description on how the partners responsibles od the WI reuirement will check if the WI has met the requirement.
7. Validation results it will be the outcome of the validation process (in case of a performance requirement the WI will be validated by checking its reaction and speed under the specific workload: reaction time will be the result of the validation)
8. Validation Status: final will be a PASS/FAIL status, but we have inserted also an Ongoing status t inform that the validation process in taking action during the project duration, on on hold process if the validation for some reasons need to be further rediscussed, and an invalid if the requirement change.

The consortium is aware that this last field could be not defined at the end of WP1 (WP2 and WP3 will end in M34), but we are committed to use the excel files also inWP2 and WP3 for the final validations of the different IPS,

5 Requirements Statistics

To get an overview of what the requirements say, we have made some statistics about how the different columns have been filled. The statistics give some information about how our requirements cover our project and its purpose.

The following tables shows the number of requirements submitted by the different Working Items.

WP2 WIs	*Requirement number*
WI2.1.1	5
WI2.1.2	6
WI2.1.3	9
WI2.1.4	6
WI2.2.1	22
WI2.2.2	10
WI2.2.3	9
WI2.2.4	8
WI2.2.5	5
WI2.2.6	1
WI2.2.7	4
WI2.2.8	3
WI2.3.1	12
WI2.3.2	2
WI2.3.3	2
WI2.3.4	2
WI2.3.5	11
WI2.3.6	1
WI2.4.1	2
WI2.4.3	7
WI2.4.4	4
WI2.4.5	17
WI2.4.6	11
WI2.4.7	2
WI2.4.8	1
WI2.5.1	43
WI2.5.10	4
WI2.5.11	2
WI2.5.12	5
WI2.5.13	8
WI2.5.2	5
WI2.5.3	1
WI2.5.4	4
WI2.5.5	1
WI2.5.6	2
WI2.5.7	2
WI2.5.8	3
WI2.5.9-1	5
WI2.5.9-2	5
WI2.5.9-3	3

WP3 WIs	*Requirement number*
WI3.1.1	7
WI3.1.2	5
WI3.1.3	18
WI3.1.4	10
WI3.1.5 REQ-L1-DCACHE	29
WI3.1.5-REQ-LLC	14
WI3.1.6	5
WI3.1.7	5
WI3.2.1	14
WI3.2.2	12
WI3.2.3	9
WI3.2.4	10
WI3.2.5	7
WI3.2.6	35
WI3.3.2	1
WI3.3.3	21
WI3.3.4	4
WI3.4.1	1
WI3.4.2	4
WI3.4.3	5
WI3.4.4	1
WI3.4.5	57
WI3.4.6	17

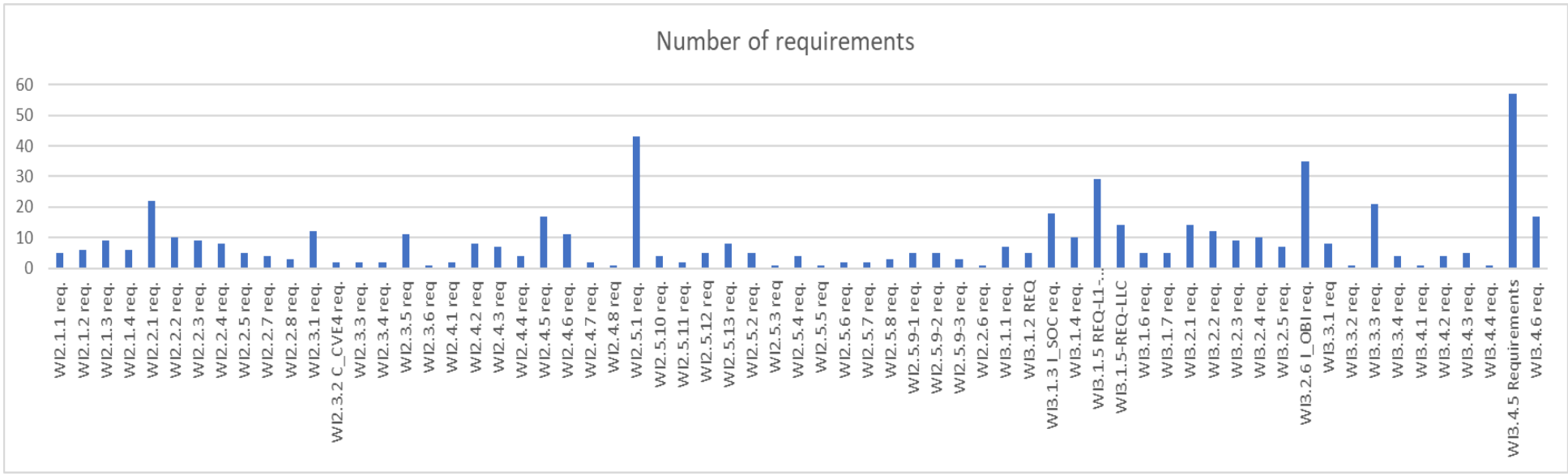

Figure 1 Requirements for Wis

We notice that the WIs offered very different number of requirements. The instructions we provided said ”a handful” which was intended to be interpreted in a liberal way so that the WIs themselves should decide how many requirements they would submit. The instruction of ”a handful” also indicated in a very informal way an expectation of granularity and detail. We did not want hundreds of very low level technical implementation details (that will be part of specifications deliverables from WP2 and WP3 respectively), but rather requirements on level with the project objectives. The total of 546 requirements is close to where we expected to be, and we think that this will constitute a useful first baseline and targets for the project.

5.1 Requirements relative to project objectives

The project as a whole has its focus on the project objectives, and the fulfilment of the WIs should contribute to achieving these project objectives. Therefore, it was recorded what objective would be affected by the different Working Items. We can see how the WIs relate to what objectives they are intended to satisfy in

Project Objectives	*Numbers of WIs*
SO1 - Processor Development	19
SO2 - Ecosystem of Industrial Quality SoC Building Blocks	28
SO3 - SoC Development Infra-Structure	1
SO4 - Vendor Independence	19
SO5 - Active European Open-source HW community	13
SO6 - Demonstration of building block interoperability	20
SO7 - Pre-Certification and Validation	8

6 Conclusion

This deliverable presented the results of the requirements and the WI definition. The methodologies adopted to carry on these two tasks have been described and some statistics and preliminary evaluations at global level have been presented.

The requirement elicitation allowed to identify 546 requirements and they cover the project objectives reasonably well. The process of elicitation has been intense, but we have conducted a fair amount of quality improvements with reviews and an extra iteration of providers' consolidation. Still, we expect to improve and to clarify these requirements as well as to evolve supplementary requirements in the time to come towards D1.5.

www.tristan-project.eu
info@tristan-project.eu

TRISTAN has received funding from the Key Digital Technologies Joint Undertaking (KDT JU) under grant agreement nr. 101095947. The KDT JU receives support from the European Union's Horizon Europe's research and innovation programme and Austria, Belgium, Bulgaria, Croatia, Cyprus, Czechia, Germany, Denmark, Estonia, Greece, Spain, Finland, France, Hungary, Ireland, Israel, Iceland, Italy, Lithuania, Luxembourg, Latvia, Malta, Netherlands, Norway, Poland, Portugal, Romania, Sweden, Slovenia, Slovakia, Turkey

Document Info	
Work Item	*WI2.1.1*
Author(s)	*ETH* *FHG, SYSGO*
Version	*0,1*
Purpose (Use Case)	
Description	*ETH will extend CVA6 to support timing channel protection.*
State of the art	*There are no protection mechanisms against timing channels in CVA6.*
Proposed solution	*Definition, implementation and evalutation of a custom RISC-V ISA extension* *Evaluation also of other means to achieve the same effect (absence/reduction/mitigation of timing channels).* *Observability see trace unit (see Task 2.1.4).*
Prerequisites	*Based on CVA6*
Target TRL	*TRL-5*
Proposal Respondance (SOs and KPIs)	*SO1 - Processor Development* *SO4 - Vendor Independence* *KPI: IP building block in virtual repository*
Objects List	
List of generated objects	*1. Definition of RISC-V ISA extension for timing channel protection* *2. RTL code to integrate the custom ISA extension into CVA6* *3. Short analysis of time protection needs / usability from user (system software) point of view, possibly in the form of README section to (1) and/or (2).*
Development Flow	
Short description of development flow	*RTL development, deployment on FPGA, tests using channel bench* *https://github.com/SEL4PROJ/channel-bench*
Milestones	*1. Definition of RISC-V ISA extension* *2. RTL support for extension and integration into CVA6* *3. FPGA tests*
Verification & Validation	
Overview of verification/validation setup	*Regression tests using RISC-V tests* *FPGA tests using channel bench*
Appendix	

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.1.1_1	Definition of ISA extension for timing channel protection	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI2.1.1_2	RTL support for ISA extension for timing channel protection	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI2.1.1_3	FPGA setup	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI2.1.1_4	Provide Trace-Unit (RTL), for details see WI2.1.4	Mandatory									**Proposed**	FHG		
WI2.1.1_5	There shall be countermeasures and/or guidance against side channels (timing and/or space)	Best Effort									**Proposed**	Holger Blasum (SYSGO)		

Document Info	
Work Item	*WI2.1.4*
Author(s)	*ACCT, FHG, SYSGO*
Version	*0,1*
Purpose (Use Case)	
Description	*The architecture for the RISC-V trace unit will be created that allow CPU tracing.*
State of the art	*see WI2.2.1*
Proposed solution	*see WI2.2.1*
Prerequisites	*n.a.*
Target TRL	*n.a.*
Proposal Respondance (SOs and KPIs)	*SO2 – Ecosystem of Industrial Quality SoC Building Blocks*
Objects List	
List of generated objects	*1. Specification of Trace Unit Architecture*
Development Flow	
Short description of development flow	*t.b.d in an updated version*
Milestones	*t.b.d in an updated version*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*n.a.*

Document Info	
Work Item	*WI2.2.1*
Author(s)	*FHG* *Alexander Weiss (ACCT)* *MNRES* *SYSGO*
Version	*1.0*
Purpose (Use Case)	
Description	*A RISC-V trace unit will be created that allow CPU tracing. Its compatiblity is shown due to integration with MNRS CPU and FHG CPU (EMSA5)*
State of the art	*No open source Trace-Unit for RISC-V cores are available. The commercially available trace units have some limitations, which are to be solved with the TRISTAN Trace IP (e.g. missing hardware-supported instrumentation, observability of interrupt requests).*
Proposed solution	*ACCT, FHG, SYSGO, MNRS will integrate a trace-unit into the MNRS core. EMSA5 core integration will be carried out by FHG. will provide industrial automotive requirements.*
Prerequisites	*None*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO2 – Ecosystem of Industrial Quality SoC Building Blocks*
Objects List	
List of generated objects	*1. Specification of Trace Unit* *2. Designed OS Trace Unit*
Development Flow	
Short description of development flow	*t.b.d in an updated version*
Milestones	*t.b.d in an updated version*
Verification & Validation	
Overview of verification/validation setup	*RTL simulation via unit tests and simulation tests on system level with MNRS/FHG CPU.*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.2.1_1	definition of eTIP interface	Mandatory									**Proposed**	**FHG, ACCT**		
WI2.2.1_2	FhG: provide interface of EMSA5 CPU to eTIP interface	Mandatory									**Proposed**	**FHG, ACCT**		
WI2.2.1_3	MNRS: provide interface of MNRS CPU to eTIP interface	Mandatory									**Proposed**	**ACCT, MNRS**		
WI2.2.1_4	FHG, ACCT: provide Trace IP with eTIP interface	Mandatory									**Proposed**	**FHG, ACCT**		
WI2.2.1_7	The Trace IP shall be suitable to support software test impact analysis.	Best Effort									**Proposed**	**ACCT**		
WI2.2.1_8	The Trace IP shall be suitable to support structural coverage measurement of the control flow.	Best Effort									**Proposed**	**ACCT**		
WI2.2.1_9	The Trace IP shall be suitable observe the RISC-V CPU for performance optimization purposes.	Best Effort									**Proposed**	**ACCT, Absint, Tensor**		
WI2.2.1_10	The Trace IP shall be suitable to support hybrid WCET analysis.	Mandatory									**Proposed**	**ACCT, Absint**		
WI2.2.1_11	The Trace IP shall be suitable to support the analysis of event chains and measure worst-case response times (WCRT).	Best Effort									**Proposed**	**ACCT, Tensor**		
WI2.2.1_12	The Trace IP shall be suitable to analyse data and control flow coupling, and the structural coverage of the data flow.	Best Effort									**Proposed**	**ACCT, Absint**		
WI2.2.1_13	The Trace IP shall be suitable to execute runtime verification tasks.	Best Effort									**Proposed**	**ACCT, SYSGO, Tensor**		

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.1.4_1	Define extended Hart to encoder interface (eTIP)	Mandatory									**Proposed**	**ACCT / FHG**		
WI2.1.4_2	Provide HIS extension	Mandatory									**Proposed**	**ACCT / FHG**		
WI2.1.4_3	Trace bus protocol suitable for CFI analysis	Best Effort									**Proposed**	**ACCT / FHG / SYSGO**		
WI2.1.4_4	Trace Unit extension for Multi-core support	Best Effort									**Proposed**	**ACCT / FHG**		
WI2.1.4_5	Trace Unit extension for Timstamp support	Mandatory									**Proposed**	**ACCT / FHG**		
WI2.1.4_6	CVA-6 can be used for mixed critical systems with strict resource isolation (for safety and security), that is, all shared resources (caches, memory, busses) are identified and there is a way to assign them in a dedicated way to a single application or multiple applications.	Best Effort						SYSGO analyses resources and interfaces.		OnGoing		SYSGO?		

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.2.5_1	The component shall support RV32(I, E)MC Instruction Set	Mandatory		Cadence, Imperas	None		The RISC-V Instruction Set Manual Volume I: Unprivileged ISA	verification, simulation		**ToBeDone**	**Proposed**	**CH**		
WI2.2.5_2	The component shall support machine mode and user mode	Mandatory		Cadence, Imperas	None		The RISC-V Instruction Set Manual Volume 2: Privileged ISA	verification, simulation		**ToBeDone**	**Proposed**	**CH**		
WI2.2.5_3	The component shall be synthesizable to less than 15 kGE	Best Effort		Cadence and/or Synopsys	None			synthesis		**ToBeDone**	**Proposed**	**CH**		
WI2.2.5_4	The component shall achieve a Coremark score of at last 0.9 Coremark/MHz for the smallest configuration	Best Effort		Cadence	None		https://github.com/eembc/coremark	simulation		**ToBeDone**	**Proposed**	**CH**		
WI2.2.5_5	The component shall have an OBI interface	Mandatory		Cadence, Imperas	None		https://github.com/openhwgroup/ob	verification, simulation		**ToBeDone**	**Proposed**	**CH**		

Document Info	
Work Item	*WI2.2.8*
Author(s)	*SYNTHARA*
Version	*1.0*
Purpose (Use Case)	
Description	*SYNT will work on optimizing the PPA (performance, power, area) of the low-end CV32E2 core, with a particular focus on the area and power aspect of it. This work-item will involve both low-level RTL and logic optimization and higher-level configurability of different functions (e.g. removing allowing to remove non-necessary HW submodules from a specific configuration). Integrate CV-X-IF interface for co-processor support.*
State of the art	*Current version (micro) of CV32E2 has an area of about 15kGE and a working frequency of 300MHz. Moreover, CV32E2 does not support CV-X-IF.*
Proposed solution	*The proposed solution is to optimize PPA metrics of the CV32E2 to raise the working frequency close to 500MHz, mantaining the area limited to 15kGE. Moreover, the CV-X-IF will be integrated in the CV32E2 core.*
Prerequisites	*None*
Target TRL	*The Technology Readiness Level (TRL) target for this project is 5. This means that we aim to have a fully functional and validated custom a RISC-V core.*
Proposal Respondance (SOs and KPIs)	*S02: Ecosystem of industrial quality Soc Building blocks* *S04: Vendor independence* *S05: Active European Open-source HW community*
Objects List	
List of generated objects	*1. Specifications of the core and the ISE extension supported* *2. RTL code of designed core* *3. Testbench for functional verification* *4. RTL code for integration of the CV'X'IF interface*
Development Flow	

Short description of development flow	*1. Run PPA analysis* *2. Collect requirements* *3. RTL code for reduced/optimized CV32E2 core version* *4. Testbench for verifying the design* *5. RTL code for integration of the CV-X-IF in the uArchitecture of the CV32E2 core* *6. Testbench for verifying the design with the interface* *7. Simulation and synthesys metrics*
Milestones	*M1 (Month 6): Initial Requirements collected* *M3 (Month 11): Detailed specifications for the CV32E2 RISC-V core and the CV-X-IF integration* *M8 (Month 14): Initial PPA characterization results of the CV32E2 core embedding the CV-X-IF interface* *M14 (Month 34): Functional Test and design proving of RISC-V Cores and CV-X-IF integration completed*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.2.8_1	The CV32E2 core shall be synthesizable to less than 15 kGE	Best Effort									**proposed**	SYNT		
WI2.2.8_2	optimize the CV32E2 RTL to work up to 500MHz	Best Effort									**proposed**	SYNT		
WI2.2.8_3	integrate the CV-X-IF interface in the CV32E2	Mandatory	WI2.5.2								**proposed**	SYNT		

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
C:CVE4_B_EXT	Support of bitmanip instruction set extension (aka RVB)	Mandatory												
C:CVE4_P_EXT	Support of packed SIMD instruction set extension (aka RVP)	Mandatory												

Document Info	
Work Item	*WI2.3.3*
Author(s)	*UNIBO*
Version	*1.0*
Purpose (Use Case)	
Description	*UNIBO will extend the micro-architecture of CV32E40P to support the ISA extensions for mixed-precision integer arithmetic, proposed by UNIBO in T2.5, through status-based execution modes*
State of the art	*Current version of CV32E4 core does not support SIMD operations on sub-byte and mixed-precision operand formats, required for Quantized Neural Network (QNN) workloads deployed on mid-end edge processors.*
Proposed solution	*The proposed solution is to extend the computing capabilities of the CV32E4 core with support for low-bit-width integer arithmetic, down to 2-bit. These operations will be performed by a custom integer unit that can be integrated into the CV32E4 pipeline, providing a solution faster than software implementation using the existing ISA instructions available on the CV32E40P core. This approach will enable efficient execution of Quantized Neural Networks, enhancing the overall performance of the system.*
Prerequisites	*None*
Target TRL	*Our target Technology Readiness Level (TRL) for this project is 6. This means that we aim to have a fully functional and validated custom instruction set implemented on a RISC-V core, with demonstrated performance improvements compared to pure software implementations.*
Proposal Respondance (SOs and KPIs)	*SO1: Fabrication of ASICs circuit based on CV32E4* *S02: Ecosystem of industrial quality Soc Building blocks* *S04: Vendor independence* *S05: Active European Open-source HW community* *S07: Pre-Certification and Validation*
Objects List	
List of generated objects	*1. Specifications of the designed integer processing unit* *2. RTL code of designed integer processig unit* *3. Testbench for functional verification* *4. RTL code for integration of the unit into the uArchitecture of the CV32E4 core (upstreamed to OpenHW repository)*
Development Flow	
Short description of development flow	*1. Collect requirements* *2. RTL code for design of integer processing unit* *3. Testbench for verifying the design* *4. RTL code for integration of the unit into the uArchitecture of the CV32E4 core* *5. FPGA implementation (RTL) of a simple uC system* *6. Development of C kernels to test the performance and efficiency of proposed solution* *7. PPA evaluation through physical implementation in an advanced technology node (e.g. GF22FDX)*
Milestones	*M1 (Month 6): Initial Requirements collected* *M3 (Month 11): Detailed specifications for integer processing unit, RISC-V core and Extensions complete* *- Object 1* *M8 (Month 14): Initial PPA characterisation results of the integer processing unit and analysis of the integration within the CV32E4 core in terms of PPA* *- Object 2* *M14 (Month 34): Functional Test and design proving of RISC-V Cores and Extensions completed* *- Objects 3, 4*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.3.3_1	Develop a SIMD integer processing unit to execute the low-precision integer instructions described in requirements WI2.5.7.	Mandatory	WI2.5.7											
WI2.3.3_2	Extend the computing capabilities of the CV32E4 core by integrating the integer processing unit into the CV32E4 pipeline	Mandatory												

Document Info	
Work Item	*2.3.5*
Author(s)	*Teppo Karema, Tuukka Vaaraniemi*
Version	*1.0*
Purpose (Use Case)	
Description	*Prototype IC (VLSI_IOT)*
State of the art	https://github.com/ultraembedded/biriscv
Proposed solution	*Improve cache, add low latency IO, modify cache and MMU to use standard SRAM, build simple bridge for co-processor, opimize MMU and branch prediction, convert to VHDL, clean and document the RTL.*
Prerequisites	*All other building blocks excluding developed RISC-V core must be available for proto IC due to the schedule*
Target TRL	*TRL 4: technology validated in Lab*
Proposal Respondance (SOs and KPIs)	*Boots Linux in 32-bit architecture, Performance vs silicon area*
Objects List	
List of generated objects	*Source code of the developed RISC-V core, prototype ICs (2 proto rounds), evaluation board that boots Linux*
Development Flow	
Short description of development flow	RTL-Simulation->FPGA->IC->Evaluation Board
Milestones	*Two tapeouts, 1st by end of 2023, second by end of 2024*
Verification & Validation	
Overview of verification/validation setup	*Simulation bench will be developed for initial verification, FPGA + PCB will be used to verify SW+HW co-operation, IC prototype will be fabricated.*
Appendix	
Link at requirements file(s)	*none*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.3.5_1	32-bit RISC-V Architecture	Mandatory	WI3.1.1_2_3		wi5.1.2, w5.2.7	wi5.1.2				**OnGoing**	**Accepted**	Teppo Karema/VLSI		
WI2.3.5_2	Support the latest linux kernel	Mandatory										Teppo Karema/VLSI		
WI2.3.5_3	Support a single port RAM for the cache	Mandatory										Teppo Karema/VLSI		
WI2.3.5_4	Low latency memory mapped IO	Mandatory										Teppo Karema/VLSI		
WI2.3.5_5	Compact bridge for a co-processor	Mandatory										Teppo Karema/VLSI		
WI2.3.5_6	MMU performance optimization	Optional										Teppo Karema/VLSI		
WI2.3.5_7	Branch prediction optimization	Optional										Teppo Karema/VLSI		
WI2.3.5_8	Convert the HDL from Verilog to VHDL	Mandatory										Teppo Karema/VLSI		
WI2.3.5_9	Cleanup and document the RTL code	Mandatory										Teppo Karema/VLSI		
WI2.3.5_10	Use existing proprietary peripherals (such as uart, sd-card, SPI) of VLSI Solution	Optional										Teppo Karema/VLSI		
WI2.3.5_10	Design for testability	Mandatory										Teppo Karema/VLSI		

Document Info	
Work Item	*WI2.3.6*
Author(s)	*ECL*
Version	*0.1*
Purpose (Use Case)	
Description	*ECL will support via the OpenHW Europe Working Group industrial-grade*
State of the art	*CV32E4 is in TRL3, many efforts are needed to bring it to TRL5. The CVA6 is currently hosted in the OpenHW github which makes it publicly available.*
Proposed solution	*ECL will support the development and verification efforts of the WP2.3*
Prerequisites	*CV32E40P on TRL3*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO1 - Processor Development* *SO5 - Active European Open-source HW community* *SO7- Pre-certification and Validation*
Objects List	
List of generated objects	*RTL and testbench code*
Development Flow	
Short description of development flow	*The project will be split into 3 steps:* *- step1 to support the verification at TRL5 level* *- step2 to support the verification at TRL5 level* *- step3 to support the verification at TRL5 level*
Milestones	*T0 + 10 months: step1 completion* *T0 + 22 months: step2 completion* *T0 + 34 months: step3 completion*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*<...>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.3.6_1	Support the verifications efforts of the CV32E40P	Mandatory												

Document Info	
Work Item	*WI2.4.1_1*
Author(s)	*TDIS, BOSCH-FR, SYSGO*
Version	*0,1*
Purpose (Use Case)	
Description	*TDIS will increase the maturity of the CVA6 core, specifically by more thorough verification to reach TRL-5 and above. The verified CVA6 core will be delivered as open-source. BOSCH-FR will contribute to this verification.*
State of the art	*Designed from an academic organization, ETH Zurich, CVA6 is of TRL3 maturity. So far, CVA6 cannot be in industrialized.*
Proposed solution	*TDIS will lead a verification project within OpenHW Group to verify the CVA6.* *TDIS, OpenHW and BOSCH-FR will contribute to this verification.* *Verification is based on opensource tooling. SYSGO will give feedback from the perspective of usability for mixed-critical systems.*
Prerequisites	*None*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO1: Fabrication of ASICs circuit containing CVA6* *S02: Ecosystem of industrial quality Soc Building blocks* *S04: Vendor independance* *S05: Active European Open-source HW community* *S07: Pre-Certification and Validation*
Objects List	
List of generated objects	*CVA6 specifications, Design verification plans, UVM testbench*
Development Flow	
Short description of development flow	*The project will be split into 3 steps:* *- step1 to verify cv32a6 (32bits version of CVA6) at programmer view level* *- step2 to verify cv32a6 at TRL5 level* *- step3 to verify cv64a6 et TRL5 level*
Milestones	*T0 + 10 months: step1 completion* *T0 + 22 months: step2 completion* *T0 + 34 months: step3 completion*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.4.1_1	increase the maturity of the CVA6 core, specifically by more thorough verification to reach TRL-5 and above.	Mandatory										TDIS		
WI2.4.1_2	CVA-6 can be used for mixed critical systems with strict resource isolation (for safety and security), that is, all shared resources (caches, memory, busses) are identified and there is a way to assign them in a dedicated way to a single application or multiple applications. Marked as "optional" but very desirable.	Optional						Validation by analysis of resources (by SYSGO).				SYSGO		

Document Info	
Work Item	*WI2.4.2_1*
Author(s)	*TDIS*
Version	*0,1*
Purpose (Use Case)	
Description	*Coprocessor interface based on CV-X-IF supporting memory access through cva6 internal load and store unit*
State of the art	*CVA6 supports 3 over 6 CVXIF interfaces. So far, coprocessor cannot access memory.*
Proposed solution	*Support of memory access by coprocessor through CVXIF*
Prerequisites	*none*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*S02: Ecosystem of industrial quality Soc Building blocks* *S05: Active European Open-source HW community* *SO6: Demonstration of building block interoperability*
Objects List	
List of generated objects	*Specification, design verification plan, RT, UVM testbench*
Development Flow	
Short description of development flow	*Modification of the load and store unit of CVA6 to support the coprocessor request to access to the memory.*
Milestones	*T0 + 18 months: RTL merge on OpenHW Group repository*
Verification & Validation	
Overview of verification/validation setup	*Verified by WI2.4.1_1*
Appendix	
Link at requirements file(s)	*n.a.*

WI2.4.2_2
TDIS
0,1

Coprocessor interface based on CV-X-IF supporting coprocessor compressed instructions
CVA6 implements 3 over 6 CVXIF insterfaces. So far, compressed instructions cannot be executed by coprocessor.
Support of compressed instructions by coprocessor through CVXIF

none
TRL5
S02: Ecosystem of industrial quality Soc Building blocks
S05: Active European Open-source HW community
SO6: Demonstration of building block interoperability

Specification, design verification plan, RT, UVM testbench

Modification of the decode stage of CVA6 to support the compressed instructions to be offload to the coprocessor

T0 + 18 months: RTL merge on OpenHW Group repository

Verified by WI2.4.1_1

n.a.

WI2.4.2_SCAIE-V_1
TUDA
0,1

Tightly-coupled ISA extension interface for CVA6 based on SCAIE-V.

CVA6 currently just has a more loosely coupled CV-X-IF interface.

Allow insertion of custom single-cycle R-type instructions into CVA6 pipeline. An EDA tool will automatically insert the required interfaces into the base CVA6 core.
none
TRL4-5
S02: Ecosystem of industrial quality Soc Building blocks
S05: Active European Open-source HW community
SO6: Demonstration of building block interoperability

Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA comparison of CV-X-IF and SCAIE-V solutions.

Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions,

M11 (D2.1): Architecture specified for CVA6 SCAIE-V Interface
M23 (D2.2): CVA6 SCAIE-V Interface / Insertion Tool designed and implemented
M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed

Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.

n.a.

WI2.4.2_SCAIE-V_2
TUDA
0,1

Tightly-coupled ISA extension interface for CVA6 based on SCAIE-V.

CVA6 currently just has a more loosely coupled CV-X-IF interface.

Allow insertion of custom multi-cycle R-type instructions into CVA6 pipeline. An EDA tool will automatically insert the required interfaces into the base CVA6 core.
none
TRL4-5
S02: Ecosystem of industrial quality Soc Building blocks
S05: Active European Open-source HW community
SO6: Demonstration of building block interoperability

Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA comparison of CV-X-IF and SCAIE-V solutions.

Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions, interface with instruction flow control for multi-cycle operation

M11 (D2.1): Architecture specified for CVA6 SCAIE-V Interface
M23 (D2.2): CVA6 SCAIE-V Interface / Insertion Tool designed and implemented
M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed

Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.

n.a.

WI2.4.2_SCAIE-V_3
TUDA
0,1

Tightly-coupled ISA extension interface for CVA6 based on SCAIE-V.

CVA6 currently just has a more loosely coupled CV-X-IF interface.

Allow insertion of custom memory instructions into CVA6 pipeline. An EDA tool will automatically insert the required interfaces into the base CVA6 core.

none
TRL4-5
S02: Ecosystem of industrial quality Soc Building blocks
S05: Active European Open-source HW community
SO6: Demonstration of building block interoperability

Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA comparison of CV-X-IF and SCAIE-V solutions.

Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions, specifically the CVA6 load/store unit. Arbitration/scheduling of base-pipeline and ISA extension memory accesses needs to be considered.

M11 (D2.1): Architecture specified for CVA6 SCAIE-V Interface
M23 (D2.2): CVA6 SCAIE-V Interface / Insertion Tool designed and implemented
M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed

Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.

n.a.

WI2.4.2_SCAIE-V_4
TUDA
0,1

Tightly-coupled ISA extension interface for CVA6 based on SCAIE-V.

CVA6 currently just has a more loosely coupled CV-X-IF interface.

Allow insertion of custom control flow instructions, e.g., capable of branching on ISA-external state, into CVA6 pipeline. An EDA tool will automatically insert the required interfaces into the base CVA6 core.
none
TRL4-5
S02: Ecosystem of industrial quality Soc Building blocks
S05: Active European Open-source HW community
SO6: Demonstration of building block interoperability

Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA analysis of SCAIE-V solutions. Note: a comparison with CV-X-IF is not possible, as that lacks the required functionality

Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions, specifically the CVA6 next PC / branch prediction logic. Also consider selective flushes of the base pipeline in case of a mispredicted custom control flow instruction.

M11 (D2.1): Architecture specified for CVA6 SCAIE-V Interface
M23 (D2.2): CVA6 SCAIE-V Interface / Insertion Tool designed and implemented
M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed

Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.

n.a.

WI2.4.2_SCAIE-V_5
TUDA
0,1

Tightly-coupled ISA extension interface for CVA6 based on SCAIE-V.

CVA6 currently just has a more loosely coupled CV-X-IF interface.

Allow execution of custom instructions decoupled (in parallel) to the pipeline of the base core. EDA tool will automatically insert the required interfaces into the base CVA6 core.
none
TRL4-5
S02: Ecosystem of industrial quality Soc Building blocks
S05: Active European Open-source HW community
SO6: Demonstration of building block interoperability

Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA analysis of SCAIE-V solutions. Note: a comparison with CV-X-IF is not possible, as that lacks the required functionality

Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions. Decoupled execution interacts at multiple points with the base core: register file access, scheduling/dispatch, arbitration in writeback/commit-stages. Also requires automatic creation of management instructions for stopping/awaiting decoupled execution at end-of-scope (e.g., function call boundaries).

M11 (D2.1): Architecture specified for CVA6 SCAIE-V Interface
M23 (D2.2): CVA6 SCAIE-V Interface / Insertion Tool designed and implemented
M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed

Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.

n.a.

WI2.4.2_SCAIE-V_6
TUDA
0,1

Tightly-coupled ISA extension interface for CVA6 based on SCAIE-V.

CVA6 currently just has a more loosely coupled CV-X-IF interface.

Allow correct operation of ISA-extended cores in the presence of context switches, e.g. between Linux processes. EDA tool will automatically insert the required interfaces into the base CVA6 core.
none
TRL4-5
S02: Ecosystem of industrial quality Soc Building blocks
S05: Active European Open-source HW community
SO6: Demonstration of building block interoperability

Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA analysis of SCAIE-V solutions. Note: a comparison with CV-X-IF is difficult, as that lacks the required functionality. It might be possible to retrofit it into CV-X-IF, but this will be considered optional.

Custom instruction state will be moved into the SCAIE-V interface layer. Custom mechanisms for its save/restore will be generated (e.g., exposing custom instruction internal state as CSRs, or auto-generating save/restore instructions for flushing/loading internal state through the CVA6 load/store units. For decoupled instructions, writebacks must only be permitted in their originating context and suppressed in all others.

M11 (D2.1): Architecture specified for CVA6 SCAIE-V Interface
M23 (D2.2): CVA6 SCAIE-V Interface / Insertion Tool designed and implemented
M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed

Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.

n.a.

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.4.2_1	Coprocessor interface based on CX-X-IF supporting memory access through cva6 internal load and store unit	Mandatory										TDIS		
WI2.4.2_2	Coprocessor interface based on CX-X-IF supporting coprocessor compressed instructions	Optional										TDIS		
WI2.4.2_SCAIE-V_1	Extend SCAIE-V to insert custom **single-cycle "R-type"** instructions in CVA6 pipeline	Mandatory	WI2.4.1, WI2.4.3			I:SCAIE-V_1	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.2_SCAIE-V_2	Extend SCAIE-V to insert custom ***multi-cycle*** **"R-type"** instructions in CVA6 pipeline	Mandatory	WI2.4.1, WI2.4.3			I:SCAIE-V_2	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.2_SCAIE-V_3	Extend SCAIE-V to insert custom **memory** instructions in CVA6 pipeline	Best Effort	WI2.4.1, WI2.4.3			I:SCAIE-V_3	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.2_SCAIE-V_4	Extend SCAIE-V to insert custom **control** instructions in CVA6 pipeline	Best Effort	WI2.4.1, WI2.4.3			I:SCAIE-V_4	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.2_SCAIE-V_5	Extend SCAIE-V to insert custom **decoupled** instructions in CVA6 pipeline	Optional	WI2.4.1, WI2.4.3			I:SCAIE-V_5	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.2_SCAIE-V_6	Integrate custom instruction **internal state** handling with Linux context switches	Optional	WI2.4.1, WI2.4.3			I:SCAIE-V_6	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications, Linux process switching mechanisms for context save / restore	Execute code using SCAIE-V generated context save/restore code in RTL simulation and FPGA implementation, synchronize with other TRISTAN efforts to boot Linux on FPGA prototype	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF, measure software overheads for Linux context switches covering custom instruction state			TUDA-ESA		

Document info									
Work item	WI2.4.3_1	WI2.4.3_2	WI2.4.3_SCAIE-V_1	WI2.4.3_SCAIE-V_2	WI2.4.3_SCAIE-V_3	WI2.4.3_SCAIE-V_4	WI2.4.3_SCAIE-V_5	WI2.4.3_SCAIE-V_6	WI2.4.3_3
Author(s)	TDS	TDS	TUDA	TUDA	TUDA	TUDA	TUDA	TUDA	TRT
Version	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Purpose (Use Case)									
Description	Increase the performance of the CVA6 core, by architectural optimization with a focus on the architectural exploration of a multi-issue and superscalar evolution of CVA6.	Deliver a FPGA implementing a Hardware Secure Module (HSM) with Post Quantum Cryptography (PQC).	Tightly ISA extension interface for a wider (superscalar and/or out-of-order) CVA6 based on SCAIE-V.	Tightly-coupled ISA extension interface for wider (superscalar and/or out-of-order) CVA6 based on SCAIE-V.	Tightly-coupled ISA extension interface for wider (superscalar and/or out-of-order) CVA6 based on SCAIE-V.	Tightly-coupled ISA extension interface for wider (superscalar and/or out-of-order) CVA6 based on SCAIE-V.	Tightly-coupled ISA extension interface for wider (superscalar and/or out-of-order) CVA6 based on SCAIE-V.	Tightly-coupled ISA extension interface for wider (superscalar and/or out-of-order) CVA6 based on SCAIE-V.	TRT will work on the core architecture, with a focus on compact and efficient implementations and support of FPGA-based demonstrations.
State of the art	Current CVA6 implementation is the starting point.	Today, HSM are implemented with proprietary processors without dedicated PQC coprocessor. Resulting performance is limited.	CVA6 currently just has a more loosely coupled CV-X-IF interface.	CVA6 currently just has a more loosely coupled CV-X-IF interface.	CVA6 currently just has a more loosely coupled CV-X-IF interface.	CVA6 currently just has a more loosely coupled CV-X-IF interface.	CVA6 currently just has a more loosely coupled CV-X-IF interface.	CVA6 currently just has a more loosely coupled CV-X-IF interface.	Current CVA6 implementation is the starting point.
Proposed solution	Dual issue and superscalar optimizations are planned keeping the in-order characteristic	Connect ARA coprocessor to CVA6 and port the system on Genesys2 FPGA	Allow insertion of custom single-cycle R-type instructions into CVA6 pipeline. An EDA tool will automatically insert the required interfaces into the base CVA6 core.	Allow insertion of custom multi-cycle R-type instructions into wider CVA6 pipeline. An EDA tool will automatically insert the required interfaces into the base CVA6 core.	Allow insertion of custom memory instructions into wider CVA6 pipeline. An EDA tool will automatically insert the required interfaces into the base CVA6 core.	Allow insertion of custom control flow instructions, e.g., capable of branching an ISA-external state, into wider CVA6 pipeline. An EDA tool will automatically insert the required interfaces into the base CVA6 core.	Allow execution of custom instructions decoupled (in parallel) to the wider pipeline of the base core. EDA tool will automatically insert the required interfaces into the base CVA6 core, ideally reusing scheduling/issue/data-hazard handling facilities of base-core for decoupled custom instructions.	Allow correct operation of ISA-extended wider cores in the presence of context switches, e.g. between Linux processes. EDA tool will automatically insert the required interfaces into the base CVA6 core.	Focus on frequency increase: brooking critical paths.
Prerequisites	none	none	none	none	none	none	none	none	none
Target TRL	Close to TRL5: Verification developed in WI2.4.1_1 will be replayed on the new implementation	TRL5 for the CVA6 and best effort for the ARA coprocessor	TRL4-5	TRL4-5	TRL4-5	TRL4-5	TRL4-5	TRL4-5	TRL4-5
Proposal Respondance (SOs and KPIs)	SO4: Vendor independance SO5: Active European Open-source HW community	SO1: Fabrication of ASICs circuit containing CVA6 SO2: Ecosystem of industrial quality Soc Building blocks SO4: Vendor independance SO5: Active European Open-source HW community	SO2: Ecosystem of industrial quality Soc Building blocks SO5: Active European Open-source HW community SO6: Demonstration of building block interoperability	SO2: Ecosystem of industrial quality Soc Building blocks SO5: Active European Open-source HW community SO6: Demonstration of building block interoperability	SO2: Ecosystem of industrial quality Soc Building blocks SO5: Active European Open-source HW community SO6: Demonstration of building block interoperability	SO2: Ecosystem of industrial quality Soc Building blocks SO5: Active European Open-source HW community SO6: Demonstration of building block interoperability	SO2: Ecosystem of industrial quality Soc Building blocks SO5: Active European Open-source HW community SO6: Demonstration of building block interoperability	SO2: Ecosystem of industrial quality Soc Building blocks SO5: Active European Open-source HW community SO6: Demonstration of building block interoperability	SO4: Vendor independance SO5: Active European Open-source HW community
Objects List									
List of generated objects	Specification, design verification plan, RTL, UVM testbench	Specification, FPGA	Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA comparison of CV-X-IF and SCAIE-V solutions.	Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA comparison of CV-X-IF and SCAIE-V solutions.	Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA comparison of CV-X-IF and SCAIE-V solutions.	Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA analysis of SCAIE-V solutions. Note: a comparison with CV-X-IF is not possible, as that lacks the required functionality	Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA analysis of SCAIE-V solutions. Note: a comparison with CV-X-IF is not possible, as that lacks the required functionality	Specification, RTL, FPGA prototype validated, base functionality of extended core verified by WI2.4.1 framework, PPA analysis of SCAIE-V solutions. Note: a comparison with CV-X-IF is difficult, as that lacks the required functionality. It might be possible to retrofit it into CV-X-IF, but this will be considered optional.	RTL, FPGA demonstration
Development Flow									
Short description of development flow	A model of CVA6 pipeline will evaluate the different possible optimizations. Then implementation with the same IO interface will allow the interoperability of CVA6 with the new implementation.	The system CVA6 + ARA will be simulated in digital simulator, then ported on Genesys2 FPGA board.	Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions,	Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions, interface with instruction flow control for multi-cycle operation	Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions, specifically the CVA6 load/store unit. Arbitration/scheduling of base-pipeline and ISA extension memory accesses needs to be considered.	Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions, specifically the CVA6 next PC / branch prediction logic. Also consider selective flushes of the base pipeline in case of a mispredicted custom control flow instruction.	Modification of the CVA6 required base pipeline stages by automatically inserting the tailored logic for the newly added custom instructions. Decoupled execution interacts at multiple points with the base core: register file access, scheduling/dispatch, arbitration in writeback/commit-stages. Also requires automatic creation of management instructions for stopping/kewtting decoupled execution at end-of-scope (e.g., function call boundaries).	Custom instruction state will be moved into the SCAIE-V interface layer. Custom mechanisms for its save/restore will be generated (e.g., exposing custom instruction internal state as CSRs, or auto-generating save/restore instructions for flushing/loading internal state through the CVA6 load/store units. For decoupled instructions, writebacks must only be permitted in their originating context and suppressed in all others.	Assess critical paths, find the suitable solution: local or global optimization, tradeoffs.
Milestones	T0 + 18 months: first RTL version T0 + 36 months: Final RTL version	T0 + 18 months: first FPGA version T0 + 30 months: final FPGA version	M11 (D2.1): Architecture specified for CVA6 SCAIE-V interface M23 (D2.2): CVA6 SCAIE-V interface / insertion Tool designed and implemented M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed	M11 (D2.1): Architecture specified for CVA6 SCAIE-V interface M23 (D2.2): CVA6 SCAIE-V interface / insertion Tool designed and implemented M34 (D2.3): functional test of CVA6 core with mix of ISA extensions completed	M11 (D2.1): Architecture specified for CVA6 SCAIE-V interface M23 (D2.2): CVA6 SCAIE-V interface / insertion Tool designed and implemented M34 (D2.3): functional test of CVA6 core with mix of ISA extensions completed	M11 (D2.1): Architecture specified for CVA6 SCAIE-V interface M23 (D2.2): CVA6 SCAIE-V interface / insertion Tool designed and implemented M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed	M11 (D2.1): Architecture specified for CVA6 SCAIE-V interface M23 (D2.2): CVA6 SCAIE-V interface / insertion Tool designed and implemented M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed	M11 (D2.1): Architecture specified for CVA6 SCAIE-V interface M23 (D2.2): CVA6 SCAIE-V interface / insertion Tool designed and implemented M34 (D2.3): Functional test of CVA6 core with mix of ISA extensions completed	Incremental deliveries
Verification & Validation									
Overview of verification/validation setup.	WI2.4.1_1 verification environement will be reuxed to verify the optimized implementation	The PQC verification tests developed for current proprietary product will be reused to verify FPGA implementation and performance	Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.	Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.	Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.	Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.	Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.	Base pipeline verified by WI2.4.1_1, ISA extensions verified by targeted test cases.	Use OpenHW continuous integration + run Linux on CVA6.
Appendix									
Link at requirements file(s)	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.4.3_1	Increase the performance of the CVA6 core, by architectural optimization with a focus on the architectural exploration of a multi-issue and superscalar evolution of CVA6.	Mandatory												
WI2.4.3_SCAIE-V_1	Extend SCAIE-V to insert custom ***single-cycle*** **"R-type"** instructions in widened CVA6 pipeline (multi-issue and/or out-of-order)	Mandatory	WI2.4.1, WI2.4.2			I:SCAIE-V_1	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.3_SCAIE-V_2	Extend SCAIE-V to insert custom ***multi-cycle*** **"R-type"** instructions in widened CVA6 pipeline (multi-issue and/or out-of-order)	Mandatory	WI2.4.1, WI2.4.2			I:SCAIE-V_2	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.3_SCAIE-V_3	Extend SCAIE-V to insert custom **memory** instructions in widened CVA6 pipeline (multi-issue and/or out-of-order)	Best Effort	WI2.4.1, WI2.4.2			I:SCAIE-V_3	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.3_SCAIE-V_4	Extend SCAIE-V to insert custom **control** instructions in widened CVA6 pipeline (multi-issue and/or out-of-order)	Best Effort	WI2.4.1, WI2.4.2			I:SCAIE-V_4	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.3_SCAIE-V_5	Extend SCAIE-V to insert custom **decoupled** instructions in widened CVA6 pipeline (multi-issue and/or out-of-order)	Optional	WI2.4.1, WI2.4.2			I:SCAIE-V_5	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications	Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF			TUDA-ESA		
WI2.4.3_SCAIE-V_6	Integrate custom instruction **internal state** handling with Linux context switches in widened CVA6 pipeline (multi-issue and/or out-of-order)	Optional	WI2.4.1, WI2.4.2			I:SCAIE-V_6	CVA6 microarchitecture and CV-X-IF and SCAIE-V specifications, Linux process switching mechanisms for context save / restore	Execute code using SCAIE-V generated context save/restore code in RTL simulation and FPGA implementation, synchronize with other TRISTAN efforts to boot Linux on FPGA prototype	Functional correctness and analysis of PPA overheads of implementation, comparison with CV-X-IF, measure software overheads for Linux context switches covering custom instruction state			TUDA-ESA		

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.4.4_1	Support for RISC-V vector instructions	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI2.4.4_2	Support multiprecision floating-point operations	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI2.4.4_3	Support integer arithmetic from 64 to 8-bit formats	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI2.4.4_4	Scalable architectures (2 to 8 lanes)	Optional									**Proposed**	Luca Bertaccini (ETH)		

Document Info	
Work Item	*WI2.4.4*
Author(s)	*ETH, UNIBO*
Version	*0,1*
Purpose (Use Case)	
Description	*RVV co-processor with support for low-precision integer arithmetic and multi-precision floating-point operations*
State of the art	*No RVV co-processor available for CVA6*
Proposed solution	*Develop an RVV co-processor*
Prerequisites	*Coupled to CVA6*
Target TRL	*TRL-5*
Proposal Respondance (SOs and KPIs)	*SO1 - Processor Development* *SO4 - Vendor Independence* *KPI: IP building block in virtual repository*
Objects List	
List of generated objects	*1. Specifications of the RVV co-processor* *2. RTL code for RVV co-processor for CVA6*
Development Flow	
Short description of development flow	*RTL development of the RVV co-processor for CVA6*
Milestones	*1. Specifications of the RVV co-processor* *2. RTL code for RVV co-processor with support for low-precision integer arithmetic* *3. RTL code for RVV co-processor with support for low-precision integer arithmetic and mixed-precision floating-point operations* *Timeline: tbd*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*n.a.*

Document Info	
Work Item	*WI2.4.5*
Author(s)	*BOSCH-FR*
Version	*0,1*
Purpose (Use Case)	
Description	*BOSCH-FR intends to explore configuration for CVA6 usage as embedded processor*
State of the art	*CVA6 branch cv32a6_v5.0.0*
Proposed solution	*A configuration of CVA6 as embedded core will be proposed which will consider* *1) existing configuration capabilities of CVA6* *2) CVA6 evolution in the context of TRISTAN* *3) demonstrators needs*
Prerequisites	*D1.1 and D1.2*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO1 – Processor Development* *SO2 – Ecosystem of Industrial Quality SoC Building Blocks (WP3) Activity* *SO4 – Vendor independence* *SO5 – Active European Open-Source Hardware Community* *SO6 – Demonstration of Building Block Interoperability*
Objects List	
List of generated objects	*Excel configuration file with values of parameter and associated rationale* *then verified RTL of the configuration file integrated into demonstrator*
Development Flow	
Short description of development flow	*t.b.d in updated version*
Milestones	*t.b.d in updated version*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.4.5_1	A configuration file of CVA6 as embedded processor shall be provided	Mandatory	WI1.1, WI1.2, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_2	Existing configuration capabilities of CVA6 shall be considered within the configuration file	Mandatory	WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_3	Evolution of CVA6 shall be considered within the configuration file	Mandatory	WI1.2 WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_4	Demonstrator needs shall be considered within the configuration file	Mandatory	WI1.1, WI 6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_5	Evolution of CVA6 shall comprise dual issue capability (ALU-ALU & ALU-LS)	Mandatory	WI1.1, WI1.2, WI1.3, WI 6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_6	Evolution of CVA6 shall comprise bit manipulation capability	Mandatory	WI1.1, WI1.2, WI1.3, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_7	Evolution of CVA6 shall comprise integration of scratchpad memory	Mandatory	WI1.1, WI1.2, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_8	Evolution of CVA6 shall comprise integration of a hardware hypervisor	Best Effort	WI1.2, WI1.3							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_9	Evolution of CVA6 shall enable a dual core symmetrical processor capability	Best Effort	WI1.2, WI1.3							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_10	Evolution of CVA6 shall provide support for a second AXI slave interface to load scratchpad emory	Mandatory	WI1.1, WI1.2, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_11	Evolution of CVA6 shall integrate a hardware performance monitoring unit	Best Effort	WI1.1, WI1.2, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_12	Evolution of CVA6 shall match design constraints related to high yield production targets	Mandatory	WI1.1, WI1.2, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_13	There shall be an instruction accurate model	Mandatory	WI1.1, WI1.2, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_14	There shall be an instruction accurate model matching Synopsys PA	Optional	WI1.1, WI1.2, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_15	There shall be a cycle accurate model	Mandatory	WI1.1, WI1.2, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_16	There shall be a cycle accurate model matching Synopsys PA	Optional	WI1.1, WI1.2, WI6.1							**ToBeDone**	**Proposed**	BOSCH-FR		
WI2.4.5_17										**ToBeDone**	**Proposed**	BOSCH-FR		

Document Info	
Work Item	*WI 2.4.7*
Author(s)	*Alexander Weiss (ACCT)*
Version	*1.0*
Purpose (Use Case)	
Description	*Trace IP: see WI 2.2.1*
State of the art	*Trace IP: see WI 2.2.1*
Proposed solution	*Trace IP: see WI 2.2.1*
Prerequisites	*Trace IP: see WI 2.2.1*
Target TRL	*Trace IP: see WI 2.2.1*
Proposal Respondance (SOs and KPIs)	*Trace IP: see WI 2.2.1*
Objects List	
List of generated objects	*Trace IP: see WI 2.2.1*

Development Flow	
Short description of development flow	*Trace IP: see WI 2.2.1*
Milestones	*Trace IP: see WI 2.2.1*
Verification & Validation	
Overview of verification/validation setup	*Trace IP: see WI 2.2.1*
Appendix	
Link at requirements file(s)	*<…>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.4.7_1	Specify Trace-Unit extension for online analysis of application processors (e.g. observation of OS related events: tasks, virtualization), basic requirements are given in the WI 2.2.1 requirements.	Best Effort	WI2.1.4							**OnGoing**	**Proposed**	ACCT		
WI2.4.7_2	Implement Trace-Unit extension	Best Effort									**Proposed**	ACCT		

Document Info	
Work Item	*WI2.4.8*
Author(s)	*ECL*
Version	*0.1*
Purpose (Use Case)	
Description	*ECL will support via the OpenHW Europe Working Group industrial-grade development and verification of open-source IPs, as well as licencing, support, and maintenance. In addition, the OpenHW Europe Working Group will implement digital-sovereignty by physically hosting open-source databases in European servers with FAIR access.*
State of the art	*CVA6 is in TRL3, many efforts are needed to bring it to TRL5. The CVA6 is currently hosted in the OpenHW github which makes it publicly available.*
Proposed solution	*ECL will support the development and verification efforts of the WP2.4 in order to achieve TRL5.*
Prerequisites	*CVA6 is in TRL3*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO1 - Processor Development* *SO5 - Active European Open-source HW community* *SO7- Pre-certification and Validation*
Objects List	
List of generated objects	*RTL and testbench code*
Development Flow	
Short description of development flow	*The project will be split into 3 steps:* *- step1 to support the verification at TRL5 level* *- step2 to support the verification at TRL5 level* *- step3 to support the verification at TRL5 level*
Milestones	*T0 + 10 months: step1 completion* *T0 + 22 months: step2 completion* *T0 + 34 months: step3 completion*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.4.8_1	Support the verifications efforts of the CVA6.	Mandatory												

Document Info	
Work Item	*WI2.5.1*
Author(s)	*Tiberio Fanti; Luca Lingardo; Sammy Ipenza*
Version	*1.0*
Purpose (Use Case)	
Description	*The item implemented by WI2.5.1 is intended to partially replace the receiving chain of an existing NXP NFC digital modem for secure mobile payments and identifications.* *In the context of this document, as well as in all those that will be written about WI2.5.1, by NFC Digital Modem (also referred as Contactless Interface, CLIF) we will simply refer to the submodule that sits in the receiving path of an NFC modem, after the Analog Front-End (i.e., analog signal regulation, demodulation, sampling and numerical conversion) and handling ISO-OSI "Data-Link" Layer (Layer 2) of all the NFC standards supported by the existing NXP product.*
State of the art	*NFC devices are today deployed in billions of pieces world-wide to facilitate most of the common daily-life activities. NFC powered devices are found in mobile payment, ticketing, identification, and many other IoT applications. Today's NFC digital modems support multiple communication standards in different operative configurations and environmental conditions by means of an extensive use of custom digital signal processing (DSP) logic, which is tuned to meet the final product requirements at the expense of full functional flexibility. The main drawback of today's NFC architectures is that the modulation and demodulation of the signal is performed with custom analogue and digital circuitry, thus losing on functional flexibility of the end-product. The resulting pre-silicon engineering and verification effort can be unacceptable from the business perspective and limit the end product evolution. Additional limitations of this approach are that the final chip is not adaptable to different communication standards, as well as its performance might be negatively impacted by unexpected silicon non-linearities.*
Proposed solution	*One way to balance chip-resources with the growing need for flexibility is to consider the adoption of microprocessors that are able to perform DSP instructions. In contrast to a specialized signal processing realized with custom digital logic, a microprocessor provides the wanted flexibility at the cost of potentially higher resource usage (e.g. memory demand). Additionally, higher clock rates might be required to meet the end-application performance, which has an impact on the total power consumption.* *The ultimate ambition of WI2.5.1 is to implement a microprocessor sub-system based on an open source RISC-V core and extend its ISA to handle real-time DSP operations with high performance. The proposed architecture will be instantiated in an existing NFC system and take over features described in the layer 2 of the NFC standards which are currently implemented in NFC modems using custom digital logic, thus increasing the flexibility of the receiving chain of the final product at the cost of higher clock rate and silicon area requirements.*
Prerequisites	*We will make use of real-case stimuli taken from lab measurements sessions on the the so-called digital boundary between the CLIF AFE and the CLIF DSP module. In alternative stimuli generated by Matlab models, reproducing the antenna subsystem, matching network of the PCB and the AFE, can be used.* *Any SW code excuted by the RISC-V core will rely on the existence of a initial complete toolchain made of Compiler, ISS, Tracer and Debugger, when not a full fledged IDE.*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO1, SO4, (KPIs are not numbered! Used convention starting from 1 in Table2 of doc TRISTAN KDT ::) KPI2, KPI4, KPI6,*
Objects List	
List of generated objects	*This WI will mainly deliver two objects:* *- a RISC-V based microcontroller unit, equipped with application-specific internal IPs and interconnect subsystem, to be used as DSP co-processing unit in the receiver chain of a digital NFC modem; This deliverable will be nicknamed as NFC_DSP_BLUE;* *- a RISC-V based processing unit, equipped with a local co-processor unit able to execute a set of non-standardized DSP-specific instructions, to be used as main DSP processing engine in the receiver chain of a digital NFC modem. This deliverable will be nicknamed as NFC_DSP_RED.* *Both objects will be delivered as an RTL IP database, complemented by subsystem configurations sets, module-level testbench, test and (where applies) application FW source code, input stimuli sets and testcases, design and verification specs, as well as design environment set-up guidelines (including FW compilation and execution instructions).* *The first item makes use of NXP proprietary RTL cores & IPs, as well as NXP proprietary verification stimuli, algorithms and FW source code. It is not candidate for sharing on any Open Source platform.* *The second item makes use of available Open Source RTL cores and IPs. The possibility to share the NXP increments (namely the co-processor enabling the ISA extension) is under assessment.* *Additionally, the WI activity will engage on an exploration, together with academical partners (TUG and POLITO), in the development of a full-fledge SW Defined Radio subsystem. For such an activity there are no committed objects at the moment. The major commitment is on the development of a virtual prototyping platform that will be used to run the architectural and performance analysis of different viable solutions.* *This deliverable will be nicknamed as NFC_DSP_GREEN.*
Development Flow	
Short description of development flow	*The adopted development flow is the de-facto standard used in digital HW design. The first step will be to make an analysis of the state of the art and a feasibility study by comparing different architectures. After selecting the architecture, the architectural concept proposal is finalized and detailed. This proposal will define the requirements for the RTL implementation of the microprocessor sub-system. Once the RTL implementation is completed, the details will be documented in the system datasheet triggering the start of the verification activity. The verification campaign will make sure that the core implemented in RTL matches with the requirements (architecture proposal and RTL implementation are coherent => high level requirements verified), and that the low level technical details described in the datasheet are in accordance with the RTL implementation (datasheet and RTL implementation are coherent => low level requirements verified)*
Milestones	*Development of WI2.5.1 will follow the internal NXP Digital IPs development flow: milestones are defined accordingly. Shortly put, they can listed as: Concept Start, Design Env Start, Concept Ready, Design Specification Ready, Design Env Ready, RTL Ready, Verification Testbench and Plan Ready, RTL functionally verified, RTL integrated and verified at SoC top-level, IP Database Released.*
Verification & Validation	
Overview of verification/validation setup	*The item implemented by WI2.5.1 is intended to be verified using standard RTL verification methodologies, using main vendors tools (i.e. Cadence).* *The FW running on the implemented microprocessor platform will be compiled using LLVM.*
Appendix	
Link at requirements file(s)	*<...>*

V1 - 0.1.1			V2						V3					
Requirement id	Requirement Description	Priority	WI requirements interdependencies (cores, HW ips, SW ips, Demonstrators)	Eda Tool (commercial)	Tristan Tools	WNich need requirement?	Related specification	Means of validation	Validation results	Validation Status	Requirement Status	Proposed by		
WI2.5.1_NFC_DSP_BLUE_1	WI2.5.1 First Deliverable will be a RISC-V based microprocessor subsystem, tightly connected to an existing NFC Digital Modem, able to sense signals from the system and elaborate them in real-time. This module, delivered as a RTL IP, indicated with the acronym CCP (Contactless Co-Processor) will be able to operate as Floating Point Unit of the IC main CPU.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_2	The CCP shall be built around an existing NXP proprietary RISC-V core implementation. It is not meant for Open Source distribution.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_3	The CCP shall be instantiated in an existing NFC Digital Modem core and take over some of the inbuilt functionalities, otherwise implemented with custom logic (current State of the Art). To this purpose, NXP-AT will develop a set of algorithms that will demonstrate its functionality and performance on a FPGA-based demo functional unit developed in WI6.4.1.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_4	The CCP, once embedded in the system, shall be able to execute three 10-bits 16-taps FFT (Fast Fourier Transors) within 250us, in case a 54MHz system clock frequency is used.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_5	The adopted RISC-V core shall support IMCF ISA Extensions, by means of a HW embedded FPU.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_6	The adopted RISC-V core shall be accessible a standard RISC-V Debug Interface for on-chip validation tests.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_7	The CCP shall be reachable and configurable by the environment by means of a slave AMBA3 AHB port.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_8	The CCP shall reach the system memory provided by the environment by means of a master AMBA3 AHB port	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_9	The CCP will interoperate with an application host CPU by mean of the above mentioned AMBA ports, interrupt mechanism over a proprietary communication protocol.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_10	In order to pursue re-usability in different applications, the CCP top-level will be structured in two main submodules: - A fixed CPU submodule, comprising the RISC-V, an interrupt controller, peripheral interconnect, timers and debug interfaces; - An application specific System Adaptation Layer (AKA SAL) submodule.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_11	The CCP RTL will be implemented thus to make use of: - a single system clock, reset and power domain for the microprocessor unit (CPU); - an application specific clock, reset and power domain for the SAL.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_12	The RTL code shall be LINT checked against NXP coding rules (for future re-use), and synthesizable with commercial FPGA and ASIC (CMOS 28nm) technoloev.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_13	The CCP shall be fully verified within a specific System-Verilog UVM testbench, featuring AMBA compliant UVCs and proprietary models for the system Analog Front-End and connected Digital Modem.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_14	The CCP shall be initially validated within WI6.4.1 on a FPGA-Based prototyping platform, featuring a simplified environment of the end product, including main CPU and Memories Subsystem, as well as virtualized models of the system Analog Front-End (AFE).	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_BLUE_15	The CCP shall be eventually fully validated within WI6.4.1 on-silicon, against the end-product AFE, using Engineering Samples over NXP proprietary validation boards.	Best Effort	WI6.4.1							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_RED_1	WI2.5.1 Second Deliverable will be a RISC-V based microprocessor subsystem, able to elaborate in real-time numerical signals coming from the NFC application AFE. This module, delivered as a RTL IP, will be indicated with the acronym SDR-LD (SW Defined Radio - Light Decoder).	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_2	The SDR-LD shall perform ISO 14443-2 TypeA 106kBd PCD decoding, making use of a RISC-V specific FW algorithm, supported by ISA application specific extensions.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_3	The SDR-LD shall be able to decode a real-time ADCs stream on demodulated I/Q channel on 10bits dynamic at 13.56MHz.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_4	The SDR-LD RTL will be implemented thus to make use of a single system clock, reset and power domain. The AFE incoming signals are expected to be synchronous with the SDR-LD system clock.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_5	The max clock rate of the system should not exceed 108MHz. This puts a lower boundary to the performance of the implemented decoding algorithm over the RISC-V CPU + Memory subsystem.	Best Effort	WI6.4.1							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_RED_6	The SDR-LD shall be reachable and configurable by the environment by means of a slave AMBA3 AHB port.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_7	The SDR-LD shall reach the system memory provided by the environment by means of a master AMBA3 AHB port.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_8	The SDR-LD processing unit shall be a RISC-V Open Source core.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_9	The adopted RISC-V core shall support IMCF ISA Extensions, by means of a HW embedded FPU.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_10	The used RISC-V ISA shall be extended with DSP specific instructions, supported by an adapted compiling tool chain.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_11	The SDR-LD shall be integrated in an existing NFC system, comprising a host CPU for configuration and control, and a virtual AFE which provides the ADCs stream	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_12	The SDR-LD will interoperate with an application host CPU by means of the above mentioned AMBA ports, interrupt mechanism over a proprietary communication protocol.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_13	The RTL code of the SDR-LD will be portable: i.e. featuring no technology soecific cells.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_14	The RTL code shall be LINT checked against NXP coding rules (for future re-use), and synthesizable with commercial FPGA and ASIC (CMOS 28nm) technology	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_15	The SDR-LD shall be fully verified within a specific System-Verilog UVM testbench, featuring AMBA compliant UVCs and proprietary models for the system Analog Front-End and connected Digital Modem.	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_RED_16	The SDR-LD shall be validated within WI6.4.1 on a FPGA-Based prototyping platform, featuring a simplified environment of the end product, including main CPU and Memories Subsystem, as well as virtualized models of the system Analog Front-End (AFE).	Mandatory	WI6.4.1							ToBeDone	Accepted	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_1	WI2.5.1 Third Deliverable will be a RISC-V based microprocessor subsystem, able to elaborate in real-time numerical signals coming from the NFC application AFE. This module, delivered as a RTL IP, will be indicated with the acronym SDR-FD (SW Defined Radio - Full Decoder).	Best Effort	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_2	The SDR-FD shall perform decoding of signals from various NFC Reader/Card-Emulation Standard, making use of a RISC-V specific FW algorithm. supported by ISA application specific extensions.	Best Effort	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_3	The SDR-FD shall be able to decode a real-time ADCs stream on demodulated I/Q channel on 10bits dynamic at 13.56MHz.	Best Effort	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_4	The SDR-FD shall be reachable and configurable by the environment by means of a slave standard interconnect port.	Best Effort	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_5	The SDR-FD shall reach the system memory provided by the environment by means of a master standard interconnect port.	Best Effort	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_6	The SDR-FD will be built on one or more processing units based on RISC-V comoliant Open Source cores.	Best Effort	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_7	The used RISC-V ISA for one or all of the adopted cores shall be extended with DSP specific instructions, supported by an adapted compiling tool chain.	Best Effort	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_8	The SDR-FD shall be integrated in an existing NFC system, comprising a host CPU for configuration and control, and a virtual AFE which provides the ADCs stream	Best Effort	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_9	The SDR-FD will interoperate with an application host CPU by means of the above mentioned standard interconnect ports, interrupt mechanism over a proprietary communication protocol.	Best Effort	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_10	The RTL code shall be LINT checked against NXP coding rules (for future re-use), and synthesizable with commercial FPGA and ASIC (e.g., CMOS 28nm) technology	Optional	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_11	The SDR-FD shall be fully verified within a specific System-Verilog UVM testbench, featuring UVCs compiant to the selected interconnect standard and proprietary models for the system Analog Front-End and connected Digital Modem.	Optional	N.A.							ToBeDone	Proposed	NXP-AT		
WI2.5.1_NFC_DSP_GREEN_12	The SDR-FD shall be validated within WI6.4.1 on a FPGA-Based prototyping platform, featuring a simplified environment of the end product, including main CPU and Memories Subsystem, as well as virtualized models of the system Analog Front-End (AFE).	Optional	N.A.							ToBeDone	Proposed	NXP-AT		

Document Info	
Work Item	*WI2.5.10*
Author(s)	*ETH, SYSGO*
Version	*0,1*
Purpose (Use Case)	
Description	*ETH will provide Hypervisor support for CVA6. SYSGO will give feedback on the usability of the approach and implementation for the hypervisor system software*
State of the art	*RISC-V Hypervisor extension available but not supported on CVA6*
Proposed solution	*Extending CVA6 to provide hypervisor support*
Prerequisites	*The WI builds upon CVA6*
Target TRL	*TRL-5*
Proposal Respondance (SOs and KPIs)	*SO1 - Processor Development* *SO4 - Vendor Independence* *KPI: IP building block in virtual repository*
Objects List	
List of generated objects	*RTL support for hypervisor extension on CVA6* *FPGA tests using https://github.com/bao-project/bao-hypervisor and/or custom-designed system software snippets*
Development Flow	
Short description of development flow	*RTL development, FPGA deployment, tests using https://github.com/bao-project/bao-hypervisor*
Milestones	*RTL support for hypervisor extension on CVA6* *FPGA tests using https://github.com/bao-project/bao-hypervisor*
Verification & Validation	
Overview of verification/validation setup	*FPGA tests using https://github.com/bao-project/bao-hypervisor* *Validation and/or custom-designed system software snippets*
Appendix	

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.10_1	Extending CVA6 to support hypervisor support	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI2.5.10_2	Support for FPGA	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI2.5.10_3	Hardware overhead < 20% CVA6	Best Effort									**Proposed**	Luca Bertaccini (ETH)		
WI2.5.10_4	Validation by software (Bao and/or custom snippets)	Best Effort									**Proposed**	Holger Blasum (SYSGO)		

Document Info	
Work Item	*WI2.5.11*
Author(s)	*Klaus Strohmayer (semify)* *Leo Moser (semify)* *Erwin Peterlin (semify)*
Version	*1.0*
Purpose (Use Case)	
Description	*SEM will develop custom instructions for the CV32E40X core to improve the performance of decompression and compression of digital waveforms on the fly. Compared to a pure software implementation, custom instructions will offer faster and more efficient compression and decompression capabilities for digital waveforms.*
State of the art	*he current state of the art for decompression and compression of digital waveforms is either done entirely in software or purely in hardware. Pure software implementations are slow and do not meet the real-time requirements for many applications. Pure hardware implementations, on the other hand, lack flexibility and cannot be easily modified or updated. The custom instructions developed by SEM for the CV32E40X core strike a balance between the two by providing the performance benefits of hardware acceleration while maintaining the flexibility of a software implementation.*
Proposed solution	*The proposed solution is to identify one or more specific custom instructions that can significantly reduce the calculation time required for compression and decompression of digital waveforms. These custom instructions will be integrated into the CV32E40X core, providing a hardware-based solution that is faster than a pure software implementation and more flexible than a pure hardware implementation. This approach will enable efficient and effective compression and decompression of digital waveforms on the fly, enhancing the overall performance of the system.*
Prerequisites	*The prerequisites for this project include an existing RISC-V core that provides an easy way to integrate custom instructions, as well as a reference software implementation of the compression and decompression functions.*
Target TRL	*Our target Technology Readiness Level (TRL) for this project is 8. This means that we aim to have a fully functional and validated custom instruction set implemented on a RISC-V core, with demonstrated performance improvements compared to pure software implementations. Our ultimate goal is to be able to fabricate an ASIC using this custom instruction set, and we are exploring the possibility of doing so through an eFabless shuttle program. If an ASIC implementation is not possible, an FPGA implementation will be provided instead.*
Proposal Respondance (SOs and KPIs)	*Related SOs:* *- SO1 – Processor Development (WP2, WP6)* *- SO4 – Vendor independence (WP2, WP3, WP4, WP5)* *Related KPI:* *- 2 ISS* *- Core with the instruction set extension will run in a demonstrator*
Objects List	
List of generated objects	*1. Collect requirements* *2. C-Code running the algorithim without instruction set extension* *3. C-Code running the algorithim with instruction set extension* *4. RTL code for instruction set extension* *5. Testbench for verifying the instruction set extension* *6. FPGA implementation (RTL) of a simple uC system* *7. FPGA demo board* *8. ASIC design for the system* *9. ASIC based on MPW shuttle*
Development Flow	
Short description of development flow	*Implement with C code* *Develop custom instructions* *Implement custom instructions in RTL* *Build FPGA prototype* *Implement ASIC if feasible*
Milestones	*M1 (Month 6): Initial Requirements collected* *- Object 1* *M3 (Month 11): Detailed specifications for RISC-V and Cores and Extensions complete* *- Objects 2, 3* *M8 (Month 23): Initial PPA characterisation results of RISC-V Cores and Extensions completed* *- Objects 4, 5, 6* *M11 (Month 24): Demonstrators pre-integration complete, Virtual Platform early availability* *- Object 7* *M14 (Month 34): Functional Test and design proving of RISC-V Cores and Extensions completed* *- Updated version objects of 4 and 6* *M18 (Month 36): Demonstrators complete with SW/Application demos running* *- Object 8, 9*
Verification & Validation	
Overview of verification/validation setup	*The verification of the custom instructions will be done through multiple stages. Initially, functional verification will be performed through simulation to ensure the correctness of the implementation. Then, an FPGA implementation will be used to verify the functionality on real hardware. Finally, an ASIC implementation will be used to verify the functionality and performance of the custom instructions.* *To evaluate the performance benefits of the custom instructions, we will compare the results obtained with and without the instructions for both the FPGA and ASIC implementations. This will allow us to quantify the improvement in performance achieved through the use of the custom instructions.*
Appendix	
Link at requirements file(s)	*<...>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.11_1	Custom instruction for CV32E40X core to improve the performance of compression of digital waveforms (WI2.5.11)	Mandatory		None				"RTL simulation FPGA implementation Silicon Validation (optional)"	RTL Simulation, FPGA implementation demonstrate the perfomance difference with and without instruction extension	**OnGoing**	**Accepted**	semify		
WI2.5.11_2	Custom instruction for CV32E40X core to improve the performance of decompression of digital waveforms (WI2.5.11)	Best Effort		None				"RTL simulation FPGA implementation Silicon Validation (optional)"	RTL Simulation, FPGA implementation demonstrate the perfomance difference with and without instruction extension	**OnGoing**	**Accepted**	semify		

Document Info	
Work Item	*WI2.5.12*
Author(s)	*Robert Hartung (Tensor embedded GmbH)*
Version	*1.0-draft*
Purpose (Use Case)	
Description	*TNS will develop custom instructions to improve the performance of boundary checking, a typical use-case in the automotive industry. Different implementations and features might be implemented, such a simplified check for lower <= var <= upper in different other formats, such as lower < var < upper, of even working on different types, such as float u8, u16, and so on.* *Another way would be a combined instruction that works on a vector of data and checks all of them using pre-defined and pre-filled boundary registers.*
State of the art	*The current state of boundary checks is based on the architecture and implementation. In the automotive context, software is often complex and therefore generated automatically. This often results in function checks being imported as a macro from other libraries and hence results in less optimized code.*
Proposed solution	*The proposed solution is to identify use cases of boundary checks that can benefit from an enhanced instruction set for boundary checks. For at least the two use cases mentioned above, we are writing a custom instruction that can be either used at compile-time to be used automatically by the compiler. Re-writing existing binaries with this improved instruction is not part of our proposal.*
Prerequisites	*The prerequisites for this project include an existing RISC-V core that provides an easy way to integrate custom instructions.*
Target TRL	*Our target Technology Readiness Level (TRL) for this work item is at least 5. We want to demonstrate the speed-up possible with this approach.*
Proposal Respondance (SOs and KPIs)	*None*
Objects List	
List of generated objects	*1. Collect requirements* *2. Refine use-cases* *2. C-Code for each use case (without any custom instruction)* *3. RTL code for instruction set extension* *4. Optimized C-Code using inline assembler or an enhanced compiler for the instruction* *5. Simulate using an existing simulator* *6. Optional FPGA prototype*
Development Flow	
Short description of development flow	*Implement C code for use-cases* *Develop custom instructions based on experience of writing the use-cases* *Implement custom instructions in RTL* *Try to simulate* *Ideally build FPGA prototype*
Milestones	-
Verification & Validation	
Overview of verification/validation setup	*Verification for this item is straight forward. Simple checks can be taken for typical boundary checks on the proposed custom set instructions.*
Appendix	
Link at requirements file(s)	*<...>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.12_1	Identify use cases	Mandatory	-	-	-	-				OnGoing	Accepted			
WI2.5.12_2	Example-C Code for each use case	Mandatory	WI2.5.12_1	Manually validated										
WI2.5.12_3	Design of possible custom instructions	Mandatory	WI2.5.12_1											
WI2.5.12_4	Implement instruction and integrate into a RISC-V code	Mandatory	WI2.5.12_3											
WI2.5.12_5	Extend C-code with custom instructions		WI2.5.12_4, WI2.5.12_1	Validated using existing WI2.5.12_1 results and show timing improvement.										

Document Info	
Work Item	*WI2.5.2*
Author(s)	*Christian Herber*
Version	*0.1*
Purpose (Use Case)	
Description	*FPU for low end cores. Slow but tiny FPU to bridge the gap between SW implementations and fixed cycles FPUs*
State of the art	*TinyFPU was developed as a research project, currently TRL2*
Proposed solution	*Based on the TinyFPU research, deliver a production grade TinyFPU*
Prerequisites	*Optionally CV-X-IF coprocessor interface*
Target TRL	*5*
Proposal Respondance (SOs and KPIs)	*SO2, SO5* *KPI: IP building block produced*
Objects List	
List of generated objects	*TinyFPU RTL*
Development Flow	
Short description of development flow	*tbd*
Milestones	*tbd*
Verification & Validation	
Overview of verification/validation setup	
Appendix	
Link at requirements file(s)	*<…>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.2_1	The component shall support single-precision floating point operations, excluding division and square root	Mandatory		Cadence, Imperas			RISC-V F	verification, simulation		**ToBeDone**	**Proposed**	CH		
WI2.5.2_2	The component shall be synthesiable to less than xx kGE	Best Effort						synthesis		**ToBeDone**	**Proposed**	CH		
WI2.5.2_3	The component should support integration via CV-X-IF coprocessor interface	Optional		Cadence, Imperas				verification, simulation		**ToBeDone**	**Proposed**	CH		
WI2.5.2_4	The component should support floating point operations on integer registers (Zfinx)	Optional		Cadence, Imperas			RISC-V Zfinx	verification, simulation		**ToBeDone**	**Proposed**	CH		
WI2.5.2_5	The component should support square root and division operations	Optional		Cadence, Imperas			RISC-V F	verification, simulation		**ToBeDone**	**Proposed**	CH		

Document Info	
Work Item	*WI2.5.3_1*
Author(s)	*TDIS*
Version	*0,1*
Purpose (Use Case)	
Description	*implement an interface (CV-X-IF) to connect CVA6 cores with coprocessors providing custom instructions to accelerate the execution of cryptography algorithm (AES, DES, RSA, …).*
State of the art	*CVA6 has implement a part of CV-X-IF, but no related CV-X-IF coprocessor exist Today.*
Proposed solution	*AES, DES and RSA TDIS internal coprocessor will be adapted to CV-X-IF. It allows to verify the CV-X-IF functionality and to speed-up cryptography algorythms on RISC-V processors.*
Prerequisites	*none*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*S04: Vendor independance* *S05: Active European Open-source HW community*
Objects List	
List of generated objects	*Performance improvement result*
Development Flow	
Short description of development flow	*The interfaces of the coprocessors will be adapted to CV-X-IF. Instructions will be defined to interact with coprocessors.Performance comparison will be done between coprocessor connected as usual (on AXI bus) and connected through CV-X-IF.*
Milestones	*T0 + 18 months: AES, DES* *T0 + 24 months: RSA*
Verification & Validation	
Overview of verification/validation setup	*AES, DES and RSA algorithms will be executed to verify the functionality.*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.3_1	*implement an interface (CV-X-IF) to connect CVA6 cores with coprocessors providing custom instructions to accelerate the execution of cryptography algorithm (AES, DES, RSA, ...).*	Mandatory												

Document Info	
Work Item	*WI2.5.4*
Author(s)	*ACCT* *FHG*
Version	*1.0*
Purpose (Use Case)	
Description	*The ISA extension will allow to send trace information while the opcode of the RISC-V remains functionally the same.*
State of the art	*NEXUS Standard, TCODE=7, Data Acquisiton Message*
Proposed solution	*Specify ISA extensions for automatic sending of trace messages (corresponding to Nexus TCODE 7 Data Acquisition Messages)*
Prerequisites	*none*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO2 - ISA-Extension required for Trace-Unit*
Objects List	
List of generated objects	** Architectural specification* ** Design and Implementation (prop)*

Development Flow	
Short description of development flow	*t.b.d in an updated version*
Milestones	*t.b.d in an updated version*
Verification & Validation	
Overview of verification/validation setup	*Unit Tests with RISC-V and system tests in combination with Trace Unit*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.4_1	Specify ISA extension	Mandatory									**Proposed**	**ACCT / FHG**		
WI2.5.4_1_1	Specify ISA extension based on HINT intructions if not implemented as CSR	Mandatory									**Proposed**	**ACCT / FHG**		
WI2.5.4_1_2	Specify ISA extension based on CSR intructions if not implemented as HINT	Mandatory									**Proposed**	**ACCT / FHG**		
WI2.5.4_2	FhG: implement ISA extension to EMSA5	Mandatory	WI2.5.4_1								**Proposed**	**FHG**		

Document Info	
Work Item	*WI2.5.5*
Author(s)	*ECL*
Version	*0.1*
Purpose (Use Case)	
Description	*ECL will support via the OpenHW Europe Working Group industrial-grade development and verification of open-source IPs, as well as licencing, support, and maintenance. In addition, the OpenHW Europe Working Group will implement digital-sovereignty by physically hosting open-source databases in European servers with FAIR access.*
State of the art	*CV-X-IF is a interface were coprocessors can be connected in order to offload instructions fo the processor*
Proposed solution	*ECL will support the development and verification efforts of the WP2.5.*
Prerequisites	*CVA6 on TRL3*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO1 - Processor Development* *SO5 - Active European Open-source HW community* *SO7- Pre-certification and Validation*
Objects List	
List of generated objects	*RTL and testbench code*
Development Flow	
Short description of development flow	*The project will be split into 3 steps:* *- step1 to support the verification at TRL5 level* *- step2 to support the verification at TRL5 level* *- step3 to support the verification at TRL5 level*
Milestones	*T0 + 10 months: step1 completion* *T0 + 22 months: step2 completion* *T0 + 34 months: step3 completion*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.5_1	Support the verifications efforts.	Mandatory												

Document Info	
Work Item	*WI2.5.6*
Author(s)	*Joonas Multanen (TAU)*
Version	*0.1*
Purpose (Use Case)	
Description	*This WI enables performance evaluation of RISC-V processor cores with OpenASIP (openasip.org, also known as TTA-Based Co-design Environment), an open-source toolset for co-design of customized processors. The WI will also enable automatically extending the RISC-V instruction set in order to optimize the execution of workloads.*
State of the art	*TBD*
Proposed solution	*TAU will work on RISC-V ISA support to OpenASIP. The toolset will be expanded to support performance evaluation and automated extension of the instruction set based on a set of applications / hot loops of interest.*
Prerequisites	*None*
Target TRL	*TRL 3: experimental proof of concept. The above mentioned solutions implemented and validated within the OpenASIP toolset.*
Proposal Respondance (SOs and KPIs)	*SO3 – SoC Development Infra-Structure*
Objects List	
List of generated objects	*Support for performance evaluation of RISC-V processors in OpenASIP, automated extension of RiSC-V ISA based on a set of applications or hot loops*
Development Flow	
Short description of development flow	*Implement support for automated RISC-V instruction set extension in OpenASIP* *Evaluate with appropriate benchmarks*
Milestones	*TBD*
Verification & Validation	
Overview of verification/validation setup	*The WI will be validated by using a golden reference to which simulations will be compared to.*
Appendix	
Link at requirements file(s)	*<…>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.6_1	Performance evaluation of RISC-V processors with the OpenASIP toolset	Mandatory						Instruction-set simulation or RTL simulation	Correctness of output compared to a golden reference	**ToBeDone**	**Proposed**	TAU		
WI2.5.6_2	Automated extension of the RISC-V instruction set with custom operations based on a set of applications or hot loops	Mandatory						Instruction-set simulation, RTL simulation	Correctness of output compared to a golden reference	**ToBeDone**	**Proposed**	TAU		

Document Info	Document Info	
Work Item	Work Item	*WI2.5.7*
Author(s)	Author(s)	*UNIBO*
Version	Version	*1.0*
Purpose (Use Case)	**Purpose (Use Case)**	
Description	Description	*UNIBO will develop a set of light-weight extensions (supporting byte, nibble, crumb and mixed-precision dot-product and ALU operations with SIMD instructions) to improve efficiency of RISC-V pipelines on low-bitwidth mixed-precision integer arithmetic.*
State of the art	State of the art	*Current version of CV32E4 core does not support SIMD operations on sub-byte and mixed-precision operand formats, required for Quantized Neural Network (QNN) workloads deployed on mid-end edge processors.*
Proposed solution	Proposed solution	*The proposed solution is to extend the instruction set of the CV32E40P core, supporting byte, nibble, crumb and mixed-precision dot-product and ALU operations with SIMD instructions. This approach will improve the performance and efficiency of CV32E40P core on low-bitwidth integer arithmetic kernels.*
Prerequisites	Prerequisites	*None*
Target TRL	Target TRL	*Our target Technology Readiness Level (TRL) for this project is 6.*
Proposal Respondance (SOs and KPIs)	Proposal Respondance (SOs and KPIs)	*S04: Vendor independence* *S05: Active European Open-source HW community*

Objects List	Objects List	
List of generated objects	List of generated objects	*1. Specifications of the ISA extension proposed* *2. New ISA instructions integrated into the ISS*

Development Flow	Development Flow	
Short description of development flow	Short description of development flow	*1. Collect requirements* *2. Analyze QNN kernels to identify a set of useful isntructions to speed-up the execution* *3. Integration of the designed instructions in an Instruction Set Simulator (ISS)* *4. Validate the improvements by inferring the new instructions into the baseline kernels*
Milestones	Milestones	*M1 (Month 6): Initial Requirements collected* *M3 (Month 11): Detailed specifications for the new ISA instructions - Object 1* *M14 (Month 34): Functional Test and design proving of RISC-V Cores and Extensions completed* *- Objects 3, 4*

Verification & Validation	Verification & Validation	
Overview of verification/validation setup	Overview of verification/validation setup	*n.a.*

Appendix	Appendix	
Link at requirements file(s)	Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.7_1	ISA extensions for low-bitwidth integer arithmetic to accelerate Quantized Neural Networks convolutional kernels on mid-end RISC-V processors	Mandatory												
WI2.5.7_2	Minimize ISA encoding space occupancy to be compliant with standard RISC-V extensions	Mandatory												

Document Info	
Work Item	*WI2.5.8*
Author(s)	*SYNTHARA*
Version	*1.0*
Purpose (Use Case)	
Description	*POLITO and SYNT will design and implement a fully synthesizable, bare-metal, edge-computing-oriented microprocessor using OpenHW CV32E2 as a starting point. The core, verified at the module level, will feature a scalable and configurable microarchitecture implementing the base 32-bit instruction set (RV32I) with some relevant standard extension (e.g., M, C, and B), and a "light" version ((i.e., tailored for edge-computing area/power budget) of the RISC-V Vector Extension (V). UNIBO will contribute to the efforts of SYNT and POLITO providing support for the initial instruction set extensions and the integration process. UNIBO will support POLITO and SYNT for the integration process of the extensions in WI2.5.7.*
State of the art	*Current version of CV32E2 core does not support any vector extension, as well as any SIMD operations on sub-byte and mixed-precision operand formats, required for Quantized Neural Network (QNN) workloads deployed on edge processors.*
Proposed solution	*The proposed solution is to extend the instruction set of the CV32E2 core, supporting light version of vector operation as well as byte, nibble, crumb and mixed-precision dot-product and ALU operations with SIMD instructions. This approach will improve the performance and efficiency of CV32E2 core on low-bitwidth integer arithmetic kernels.*
Prerequisites	*None*
Target TRL	*The Technology Readiness Level (TRL) target for this project is 5. This means that we aim to have a fully functional and validated custom a RISC-V core.*
Proposal Respondance (SOs and KPIs)	*S02: Ecosystem of industrial quality Soc Building blocks* *S04: Vendor independence* *S05: Active European Open-source HW community*
Objects List	
List of generated objects	*1. Specifications of the ISA extension proposed* *2. New ISA instructions integrated into the ISS* *2. RTL code of designed co-processor embedding the ISA extension* *3. Testbench for functional verification*
Development Flow	
Short description of development flow	*1. Collect requirements* *2. Analyze DNN kernels to identify a set of useful isntructions to speed-up the execution* *3. Integration of the designed instructions in an Instruction Set Simulator (ISS)* *3. RTL code for co-processor* *4. Testbench for verifying the design* *5.integration of the co-processor in the CV32E2 through the CV-X-IF interface* *6. Testbench for verifying the design* *7. Simulation and synthesys metrics*
Milestones	*M1 (Month 6): Initial Requirements collected* *M3 (Month 11): Detailed specifications for the new ISA instructions* *M14 (Month 34): Functional Test and design proving of RISC-V Cores and Extensions completed*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.8_1	ISA extensions for memory manipulation e.g. memory transpose, memory FIFO	Mandatory									**proposed**	SYNT		
WI2.5.8_2	ISA extensions for "light" version ((i.e., tailored for edge-computing area/power budget) of the RISC-V Vector Extension (V)	Best Effort									**proposed**	SYNT		
WI2.5.8_3	ISA extensions for low-bitwidth integer arithmetic to accelerate Quantized Neural Networks inference	Best Effort	WI2.5.7								**proposed**	SYNT		

Document Info	
Work Item	*WI2.5.9*
Author(s)	*BOSCH-FR* *TUDA*
Version	*0,1*
Purpose (Use Case)	
Description	*WI2.5.9-1: BOSCH-FR intends to extend native ISA with AI and Automotive specific op-codes. WI2.5.9-2: develops custom extensions targeting the acceleration of automotive sensor and actuator applications. WI2.5.9-3: TUDA will select /FR proposed ISA extensions for implementation using the SCAIE-V extension interface and analyse their benefits vs. costs*
State of the art	*Reference ISA*
Proposed solution	*BOSCH-FR intends to develop ad-hoc needs for the sake of AI specific needs worth implementing as tightly attached coprocessor. will identify representative sensor and actuator applications and profile the applications to identify and develop custom instruction extensions. TUDA will create hardware for the selected ISA extensions, attach them to the base processor pipeline, and perform a PPA analysis down to the ASIC layout level.*
Prerequisites	*D1.2*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO1 – Processor Development* *SO2 – Ecosystem of Industrial Quality SoC Building Blocks (WP3) Activity* *SO4 – Vendor independence* *SO6 – Demonstration of Building Block Interoperability*
Objects List	
List of generated objects	*Accelerator meant for specific AI operations and Automotive AutoSar compliance* *Custom extensions for automotive sensor and actuator applications*
Development Flow	
Short description of development flow	*tbd*
Milestones	*tbd*
Verification & Validation	
Overview of verification/validation setup	*N.A on Req. Mgt so far*
Appendix	
Link at requirements file(s)	*N.A on Req. Mgt so far*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.5.9-3_SCAIEVISAX_1	Select industry-proposed ISAX Extensions for implementation using SCAIE-V interface	Best Effort	WI2.4.2,WI2.4.3			I:SCAIE-V		Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation			TUDA		
WI2.5.9-3_SCAIEVISAX_2	Define microarchitecture for industry-proposed ISAX Extensions for implementation using SCAIE-V interface	Best Effort	WI2.4.2,WI2.4.3			I:SCAIE-V		Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation			TUDA		
WI2.5.9-3_SCAIEVISAX_3	Implement RTL hardware for industry-proposed ISAX Extensions for implementation using SCAIE-V interface	Best Effort	WI2.4.2,WI2.4.3			I:SCAIE-V		Execute code using custom instruction in RTL simulation and FPGA implementation	Functional correctness and analysis of PPA overheads of implementation			TUDA		

Document Info	
Work Item	*WI2.2.6*
Author(s)	*ECL*
Version	*0.1*
Purpose (Use Case)	
Description	*ECL will support via the OpenHW Europe Working Group industrial-grade development and verification of open-source IPs, as well as licencing, support, and maintenance. In addition, the OpenHW Europe Working Group will implement digital-sovereignty by physically hosting*
State of the art	*CV32E4 is in TRL3, many efforts are needed to bring it to TRL5. The CVA6 is currently hosted in the OpenHW github which makes it publicly available.*
Proposed solution	*ECL will support the development and verification efforts of the WP2.2*
Prerequisites	*CV32E4 on TRL3*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO1 - Processor Development* *SO5 - Active European Open-source HW community* *SO7- Pre-certification and Validation*
Objects List	
List of generated objects	*RTL and testbench code*
Development Flow	
Short description of development flow	*The project will be split into 3 steps:* *- step1 to support the verification at TRL5 level* *- step2 to support the verification at TRL5 level* *- step3 to support the verification at TRL5 level*
Milestones	*T0 + 10 months: step1 completion* *T0 + 22 months: step2 completion* *T0 + 34 months: step3 completion*
Verification & Validation	
Overview of verification/validation setup	*n.a.*
Appendix	
Link at requirements file(s)	*<...>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.2.6_1	Support the verifications efforts of the CV32E4	Mandatory												

Document Info	
Work Item	*WI3.1.1*
Author(s)	*Michael Faulwaßer (FHG)* *Martin Zimmerling (FHG)* *Alexander Weiss (ACCT)*
Version	*1.0*
Purpose (Use Case)	
Description	*ACCT, FHG, and SYSGO will develop and test a secured transmission trace bus with message, TSN and control flow reconstruction elements (leveraging trace collection from WP2). The output shall be TSN shared trace integration for EMSA5.*
State of the art	*There are Trace-Interfaces using interfaces like Parallel Bus or Aurora before, but no trace interface with TSN is known.*
Proposed solution	*The data from the CPU trace unit will be streamed to the TSN trace sink, repacked and transferred via TSN Ethernet. On the off-chip side the trace data stream is extracted from the TSN Ethernet data.The trace sink uses the FHG TSN-EP core.*
Prerequisites	*This item requires the results of the architectural results, specified trace data output and interface of trace unit from WP2.*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO2 – Ecosystem of Industrial Quality SoC Building Blocks* *SO6 – Demonstration of Building Block Interoperability*
Objects List	
List of generated objects	*1. Specification of trace bus* *2. Trace Bus Netlist* *3. Simulation report*
Development Flow	
Short description of development flow	*t.b.d in an updated version*
Milestones	*t.b.d in an updated version*
Verification & Validation	
Overview of verification/validation setup	*HDL Simulation of Digital Design* *Hardware Test will be done with WP6 TRACE_FS*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.x.y_id_subids(optional)	<Describe here the requirement>		WI3.1.1_2_3		wi5.1.2, w5.2.7	wi5.1.2				OnGoing	Accepted			
WI3.1.1_0	The TSN tracebus contains the FHG TSN-EP IP	Mandatory									Proposed	FHG		
WI3.1.1_1	1GbE (1-Gigabit-Ethernet) will be provided by the interface	Mandatory									Proposed	FHG		
WI3.1.1_2	Trace Bus allows to transmit trace data and user data with secured bandwidth	Mandatory									Proposed	FHG		
WI3.1.1_3	Trace buffer can be accessed via application	Mandatory									Proposed	ACCT, FHG		
WI3.1.1_4	Trace Bus provides AXI4-Stream interface for User Data	Mandatory									Proposed	FHG		

Document Info	
Work Item	*wi 3.1.2 JTAG debugger*
Author(s)	*Sara Bocchio*
Version	*1*
Purpose (Use Case)	
Description	*The JTAG debugger will served as debugger unit for 32 and 64 bits family of core.*
State of the art	*The Ips will start from the riscv debugger developed in the pulp projecthttps://github.com/pulp-platform/riscv-dbg*
Proposed solution	*The solutions will be fully verified in order to reach TRL6 and it will be used inside the industrial demonstrator. Moreover the final Ips will extend the existing functionality by providing a trigger module and the support of the abstract commands*
Prerequisites	*Initial evaluation of existing tests, testbenches of the Ips-*
Target TRL	*TRL6 minimum*
Proposal Respondance (SOs and KPIs)	*SO2 - Ecosystem of Industrial Quality SoC Building Blocks*
Objects List	
List of generated objects	*Revision of the riscv debugger specification, RTL and verification qualification metrisc.*
Development Flow	
Short description of development flow	*<Provide here an overview on how the WI will be developed>*
Milestones	*M01 Revision of the riscv debugger specification and evaluation of existing tests in terms of coverage (M9)* *M02 RTL improved in functionality (M18)* *M03 RTL fully verified and prototyped in the industrial demonstrator*
Verification & Validation	
Overview of verification/validation setup	*Coverage metrics (code and functional) fault injection based qualification of the verification* *Prove in silicon (technology will be defined)*
Appendix	
Link at requirements file(s)	*<…>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.1.2_1	The IP will support JTAG (IEEE Std 1149.1-2013).	Mandatory								**OnGoing**	**Proposed**	Sara Bocchio		
WI3.1.2_2	The IP will implement a system Bus Access with standard 32 or 64 bit bus interface that will be the same implemented in WI 22.1	Mandatory									**Proposed**	Sara Bocchio		
WI3.1.2_3	The IP will abstract coimplementmmand according the specification described in RISC-V Debug Specification version 0.13.2	Mandatory									**Proposed**	Sara Bocchio		
WI3.1.2_4	The IP will implement trigger module according the specification described in RISC-V Debug Specification version 0.13.2	Mandatory									**Proposed**	Sara Bocchio		
WI3.1.2_5	The IP implementation shall be optimized in terms of area	Best Effort									**Proposed**	Sara Bocchio		

Document Info	
Work Item	*WI3.1.3*
Author(s)	*CEA*
Version	*0,1*
Purpose (Use Case)	
Description	*A minimal set of peripherals is needed to build a relevant SoC for embedded aplications*
State of the art	*fragmented open source solutions, no "high TRL" reference implementation (including RTL code, reference SW driver, documentation and testbench) available in a self-contained form*
Proposed solution	*consolidate a set of existing open source IPs (GPIO, UART, SPI...) to make them reach quality requirement for their usage in commercial-grade product*
Prerequisites	*none*
Target TRL	*TRL-5*
Proposal Respondance (SOs and KPIs)	*SO2 - Ecosystem of Industrial Quality SoC Building Blocks* *SO4 - Vendor Independence* *KPI: IP building block in virtual repository*
Objects List	
List of generated objects	*RTL code* *documentation* *sample bare-metal SW driver* *sample testbench*
Development Flow	
Short description of development flow	*RTL development, functional verification*
Milestones	*RTL Code verified with APB-compliant interface*
Verification & Validation	
Overview of verification/validation setup	*Functional testbenches*
Appendix	
Link at requirements file(s)	*none*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
I:SOC-GPIO_1	read I/O value	Mandatory												
I:SOC-GPIO_2	write I/O value	Mandatory												
I:SOC-GPIO_3	trigger interrupt on value 0/1	Mandatory												
I:SOC-GPIO_4	trigger interrupt on edge rise/fall	Mandatory												
I:SOC-GPIO_5	Compatibility with AMBA APB Protocol Specification, version C (aka "APB4", aka v2.0)	Mandatory												
I:SOC-GPIO_6	configure I/O pad direction (output enable)	Optional												
I:SOC-GPIO_7	configure I/O pad features (pad-specific config)	Optional												
I:SOC-UART_1	16550 pin/registers compatibility	Mandatory												
I:SOC-UART_3	Compatibility with AMBA APB Protocol Specification, version C (aka "APB4", aka v2.0)	Mandatory												
I:SOC-SPI-MASTER_1	serial mode support	Mandatory												
I:SOC-SPI-MASTER_2	quad mode support	Optional												
I:SOC-SPI-MASTER_3	slave select support	Mandatory												
I:SOC-SPI-MASTER_4	mode 0 support	Mandatory												
I:SOC-SPI-MASTER_5	mode 1/2/3 support	Optional												
I:SOC-SPI-MASTER_6	TX/RX FIFO	Mandatory												
I:SOC-SPI-MASTER_7	Command header support	Optional												
I:SOC-SPI-MASTER_8	Dummy cycles support	Optional												
I:SOC-SPI-MASTER_9	Compatibility with AMBA APB Protocol Specification, version C (aka "APB4", aka v2.0)	Mandatory												

Document Info	Document Info	
Work Item	Work Item	*WI3.1.4*
Author(s)	Author(s)	*UNIBO*
Version	Version	*1.0*
Purpose (Use Case)	**Purpose (Use Case)**	
Description	Description	*We propose an autonomous low-power I/O DMA which is tightly-coupled to a multi-banked shared memory to sustain the bandwidth of most advanced peripherals commonly used on IoT end-nodes. The DMA is meant to be integrated into a low-end microcontroller system, coupled with a multi-banked shared memory and a tiny RISC-V CPU (e.g. RV32E2 family). Through the multi-banked, multi-port system memory we aim to improve the limits for the I/O data transfer, surpassing the performance of traditional MCU architectures. The I/O peripherals will directly connect to dedicated DMA channels, avoiding using the system interconnect. The tiny processor is dedicated to the control of I/O transfers and micromanage the peripherals. The I/O sub-system would reside in a dedicated power domain, separated from the computational resources, to enable aggressive power management.*
State of the art	State of the art	*Most advanced MCUs feature autonomous I/O subsystems able to acquire data from multiple sensors when the CPU is in idle state. However, in these traditional I/O subsystems the interconnect is shared with the processing resources of the system, both converging in a single-port system memory. Moreover, both I/O and the data processing subsystems stand in some power domain.*
Proposed solution	Proposed solution	*Our solution overcomes the bandwidth and power-management limitations of current MCU's I/O architectures introducing an autonomous I/O subsystem coupling an I/O DMA tightly-coupled with a multi-banked system memory controlled by a tiny CPU, which stands on a dedicated power domain.*
Prerequisites	Prerequisites	*None*
Target TRL	Target TRL	*Our target Technology Readiness Level (TRL) for this project is 6*
Proposal Respondance (SOs and KPIs)	Proposal Respondance (SOs and KPIs)	*S02: Ecosystem of industrial quality Soc Building blocks* *S04: Vendor independence* *S05: Active European Open-source HW community*
Objects List	**Objects List**	
List of generated objects	List of generated objects	*1. Specifications of the designed HCI* *2. RTL code of designed HCI* *3. Testbench for functional verification*
Development Flow	**Development Flow**	
Short description of development flow	Short description of development flow	*Revision and improvement of the IP RTL. Standalone simulation for functional verification. System level integration, prototype and testing through RTL simulations and FPGA implementations. We expect to reach TLR 6. We will provide the IP with test-benches, to enable verification flows. We will provide the documentation of the IP as well.*
Milestones	Milestones	*M4 (Month 11) Functional blocks detailed specifications defined -Object 1* *M9 (Month 23) Functional blocks design and implementation complete -Object 2* *M15 (Month 34) Functional blocks test & design proving complete -Object 3*
Verification & Validation	**Verification & Validation**	
Overview of verification/validation setup	Overview of verification/validation setup	*n.a.*
Appendix	**Appendix**	
Link at requirements file(s)	Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.x.y_id_subids(optional)	<Describe here the requirement>		WI3.1.1_2_3		wi5.1.2, w5.2.7	wi5.1.2				**OnGoing**	**Accepted**			
WI2.x.y_id_subids(optional)	Data													
WI2.x.y_id_subids(optional)	....													
WI2.x.y_id_subids(optional)														
WI2.x.y_id_subids(optional)														
WI2.x.y_id_subids(optional)														
WI3.1.4_1	Design of an autonomous low-power I/O DMA which is tightly-coupled to a multi-banked shared memory	Mandatory												
WI3.1.4_2	I/O peripherals will directly connect to dedicated DMA channels, avoiding using the system interconnect.	Mandatory												
WI3.1.4_3	A tiny processor is dedicated to the programming of the IO DMA and micromanagement	Mandatory												
WI3.1.4_4	The I/O sub-system would reside in a dedicated power domain, separated from the computational resources.	Mandatory												

Document Info	
Work Item	*WI3.1.5*
Author(s)	*Adrian Evans (CEA)* *César FUGUET TORTOLERO (CEA)*
Version	*1*
Purpose (Use Case)	
Description	*For RISC-V processors to achieve high performance, they require high-performance L1 data-caches that support multiple requestors and that support multiple outstanding requests. Both write-through and write-back modes of operation are required. In some applications, it can be beneficial to switch a portion of the L1 cache memory to operate in a scratch-pad mode.*
State of the art	*The current state of the art D$ for CVA6 only supports a single requestor and a single outstanding request. There are separate versions which operate either as write-through or write-back.* *In its current state, the new L1D$ cache being developed for Tristan is at a TRL 4 and the objective is for it to reach TRL 7 through a rigorous verification process. This new L1 D$ supports multiple requestors, multiple outstanding requests and can configurably operate in write-back, write-through or scratch-pad modes.*
Proposed solution	*The solution proposed in TRISTAN is to integrate the two features that are not yet implemented (configurable write-back and scratch-pad modes). Then a full verificaiton suite will be developed using a UVM methodology including functional coverage and assertions*
Prerequisites	*The CVA6 is required, and is available.*
Target TRL	*TRL7*
Proposal Respondance (SOs and KPIs)	** Design compatible with competitive clock frequencies in relevant process technologies (GF22FDX, TSMC 65nm, ST 28FDSOI,...)* ** At least a 20% improvement in average memory access time compared to existing CVA6 cache, with equal L2 bandwidth, and AXI supporting multiple outstanding requests* ** A verification suite that ensures 100% code coverage (statement and branch) and 100% functional coverage as defined in a test-plan* ** Verification coverage of at least 50 different parameter configurations of the IP (Verilog compile time parameters)* ** Industrial grade specification documents* ** Industrial grade verification documents*
Objects List	
List of generated objects	** Design specificaiton document (latex + pdf)* ** RTL source code in System Verilog* ** Scripts for compiling and running simulations (compatible with multiple commercial simulators)* ** Verification test-plan (spreadsheet)* ** Verification architecture document (pdf)* ** Verification test-bench source code*
Development Flow	
Short description of development flow	** Design specification review* ** RTL Coding Complete (MS)* ** Initial verification test-plan review (MS)* ** Initial verification architecture document review (MS)* ** Verification execution* ** Verification coverage 100% (MS)* ** Final verification review* ** Verification complete (MS)*
Milestones	*See milestones listed as MS above*
Verification & Validation	
Overview of verification/validation setup	*See above*
Appendix	
Link at requirements file(s)	*<...>*

Document Info	
Work Item	*WI3.1.5*
Author(s)	*ETH*
Version	*0,1*
Purpose (Use Case)	
Description	*A last level cache (LLC) can be integrated in SoCs to boost performance and energy efficiency. The design will use AXI4 protocol for data transfers.*
State of the art	*No open-source LLC AXI module*
Proposed solution	*Development of an LLC AXI module*
Prerequisites	*None*
Target TRL	*TRL-5*
Proposal Respondance (SOs and KPIs)	*SO2 - Ecosystem of Industrial Quality SoC Building Blocks* *SO4 - Vendor Independence* *KPI: IP building block in virtual repository*
Objects List	
List of generated objects	*LLC AXI module (RTL code)*
Development Flow	
Short description of development flow	*RTL development, functional verification*
Milestones	*RTL Code for optimised caches with AXI interface*
Verification & Validation	
Overview of verification/validation setup	*Functional testbenches*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.1.5_1_LLC	RTL development of AXI LLC module	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI3.1.5_2_LLC	Documentation	Mandatory									**Proposed**	Luca Bertaccini (ETH)		

Y1 -0.1.1			Y2						Y3						
Requirement id	Requirement Description	Priority	WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)	Eda Tool (commercial)	Tristan Tools	Which need requirement?	Related specification	Means of validation	Validation results	Validation Status	Requirement Status	Proposed by			
	Cache must arbitrate correctly between multiple clients.::The cache must ensure requesters are served based on their priority (strict priority).									OnGoing	Accepted				
I:Data-cache_1.1		H													
	Requester interface must suport read + write operations.::The DCACHE_REQ_LOAD operation must trigger a read operation on the cache.														
I:Data-cache_2.1		H													
	Requesters must provide valid requests.::The LSBits of the address (LOAD/STORE) must be aligned to the SIZE (naturally aligned).														
I:Data-cache_3.1		H													
	Cache must relay requests correctly from requestors to memory interface.::The request address must be correctly transferred from requestors to the memory write interface.														
I:Data-cache_4.1		H													
	Must support multiple outstanding requests from each requestor.::Support for multiple outstanding read requests originating from one or multiple requestors.														
I:Data-cache_5.1		H													
	Cache must provide correct read responses to requesters.::Provide 1 to 32 bytes of data, based on request size														
I:Data-cache_6.1		H													
	The cache must handle cacheable and non-cacheable requests.::The cache determines whether a given requets is cacheable based on the DCACHE_REQ_UNCACHEABLE signal.														
I:Data-cache_7.1		H													
	Provide a memory model compliant with RISC-V Weak Memory Ordering (RVWMO).::If two transactions have overlapping address ranges, then they must be comitted in the order they were issued.														
I:Data-cache_8.1		H													
	Provide an 8-channel interface to memory.::Provide a memory miss read request interface with ready/valid handshake														
I:Data-cache_9.1		H													
	Respect the ready/valid protocol on all requester and memory interfaces::Once VALID is set to '1' it must stay at '1' until RDY goes to '1' and the transfer occurs.														
I:Data-cache_10.1		H													
	The cache must support 11 AMO operations.::DCACHE_REQ_AMO_LR (size fixed at 8 bytes)														
I:Data-cache_11.1		H													
	Requester interface must support CMO and fence operations.::Support for CMO operation must be compatible with definitions in RISC-V CMO standard.														
I:Data-cache_12.1		H													
I:Data-cache_13.1	Minimize number of internal RAMs activated in parallel::	M													
	Enforce a non-allocate, write-through policy.::All write requests from requestors must appear on the memory interface (possibly after requests are coallesced).														
I:Data-cache_14.1		H													
I:Data-cache_15.1	Minimize latency using a write-buffer to mask write acknowledgement delay.:::	H													
	Provide a configurable memory pre-fetcher::Support for 4 simultaneous pre-fetching streams.														
I:Data-cache_16.1		M													
	Provide a memory-mapped CSR configuration interface::Provide CSRs for configuration														
I:Data-cache_17.1		H													
	Provide a memory interface wrapper which is compatible with AXI5.::When used with this wrapper, the memory interface must be compliant with AXI5.														
I:Data-cache_18.1		0													
	Provide RTL whose functionality is configurable based on numerous System Verilog parameters::The design that is provided must work correctly when compiled with any combination of parameters that is within range.														
I:Data-cache_19.1		H													
	Provide RTL such that the partitioning of data into SRAM cuts is configurable::Regardless of the partitioning of the SRAM cuts, the cache functionality must be correct.														
I:Data-cache_20.1		H													
I:Data-cache_21.1	Entire data-cache operates in a single clock domain::Single clock provided by CKL I	H													
	Provide an asynchronous, active low reset::Single external resetpin RST_NI. After application of the reset, the cache must restore the default directory and CSR state														
I:Data-cache_22.1		H													
	Provide the ability to initiate a flush of the write buffer.::Initiate a flush of the write buffer via a one-cycle pulse on the input pin WBUF_FLUSH I														
I:Data-cache_23.1		H													
	Respect the ready/valid protocol on all requester and memory interfaces::Once VALID is set to '1' it must stay at '1' until RDY goes to '1' and the transfer occurs.														
I:Data-cache_24.1		0													
	The cache must be able to put certain requests on-hold in the RTAB and re-execute them later, while still assuring the Memory Consistency Rules (MCRs).::In case 1 (cacheable Load or pre-fetch) which hits on a pending miss - the 2nd read must wait for 1st to complete to get data.														
I:Data-cache_25.1		0													
	The cache must be dead-lock free.::No sequence of events entering the replay-buffer must trigger a dead-lock or live-lock condition.														
I:Data-cache_26.1		0													
	The cache must be live-lock free.::No sequence of events entering the replay-buffer must trigger a dead-lock or live-lock condition.														
I:Data-cache_27.1		0													
	Provide the following run-time configuration bits for the cache.::cfg_info.Sets = read only field with number of sets														
I:Data-cache_28.1		H													
	The data-cache must maintain a set of 64-bit performance counters.::perf_write_req counts processed write requests														
I:Data-cache_29.1		M													

Document Info	
Work Item	*WI3.1.6*
Author(s)	*Christian Herber*
Version	*0.1*
Purpose (Use Case)	
Description	*Memory protection is used to improve security in embedded systems through spatial isolation of memory towards different privileges and applications*
State of the art	*The RISC-V standard knows three different means of memory protection: PMP, the basic separation, ePMP - an improved version addressing security holes in PMP, and SPMP, which provides supervisor-user and user-user isolation in addition.* *The open source core IBEX supports ePMP, but lacks verification for industrialization*
Proposed solution	*Industrialization of memory protection module to provide memory isolation in microcontrollers*
Prerequisites	*- SPMP ratification*
Target TRL	*5*
Proposal Respondance (SOs and KPIs)	*SO2, SO5* *KPI: IP building block produced*
Objects List	
List of generated objects	*<List here the planned objects delivered>*
Development Flow	
Short description of development flow	*tbd*
Milestones	*tbd*
Verification & Validation	
Overview of verification/validation setup	*tbd*
Appendix	
Link at requirements file(s)	*<…>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.1.6_1	The component shall support isolating memory areas between machine mode and lower privilege modes	Mandatory								**ToBeDone**	**Proposed**	CH		
WI3.1.6_2	The component shall support the enhanced mode of memory protection known as the Sepmp extension	Mandatory								**ToBeDone**	**Proposed**	CH		
WI3.1.6_3	The component shall support supervisor-user and user-user isolation	Mandatory								**ToBeDone**	**Proposed**	CH		
WI3.1.6_4	The component shall support low granularity configurations for area sensitive applications	Mandatory								**ToBeDone**	**Proposed**	CH		
WI3.1.6_5	The component shall support hardware assisted context switching between configurations	Mandatory								**ToBeDone**	**Proposed**	CH		

Document Info	
Work Item	*WI3.1.7*
Author(s)	*Christian Herber*
Version	*0.1*
Purpose (Use Case)	
Description	*Interrupt controller for microcontroller applications*
State of the art	*Most open source implementations of RISC-V processors use a so called CLINT - the default interrupt controller defined in the privileged specification of RISC-V. This controller lacks a sufficient number of interrupt sources and low latency interrupt response necessary in microcontroller applicaitons*
Proposed solution	*Interrupt controller based on the draft standard of CLIC. The standard is expected to be ratified in the first year of the project. This WI will leverage an existing implementation of CLIC.*
Prerequisites	*- ratification of CLIC specification*
Target TRL	*5*
Proposal Respondance (SOs and KPIs)	*SO2, SO5* *KPI: IP building block in virtual repository*
Objects List	
List of generated objects	*CLIC RTL*
Development Flow	
Short description of development flow	*tbd*
Milestones	*tbd*
Verification & Validation	
Overview of verification/validation setup	*tbd*
Appendix	
Link at requirements file(s)	*<…>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.1.7_1	The component shall support CLIC according to the first ratified version	Mandatory								**ToBeDone**	**Proposed**	CH		
WI3.1.7_2	The component shall support a configurable amount of privilege levels	Mandatory								**ToBeDone**	**Proposed**	CH		
WI3.1.7_3	The component shall support hardware vectoring	Mandatory								**ToBeDone**	**Proposed**	CH		
WI3.1.7_4	The component shall support tail-chaining of vertical interupts	Mandatory								**ToBeDone**	**Proposed**	CH		
WI3.1.7_5	The component shall support fast preemption of interrupts based on interrupt level and privilege	Mandatory								**ToBeDone**	**Proposed**	CH		

Document Info	
Work Item	*WI3.2.1*
Author(s)	*ETH, , SYSGO*
Version	*0.1*
Purpose (Use Case) Development Flow	
Short description of development flow	*Specifications, RTL development, testbench development*
Milestones	*AXI building blocks* *Testbench*
Verification & Validation	
Overview of verification/validation setup	*Functional testbenches; review for suitability for mixed-critical systems*
Appendix	
Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.2.1_1	Specification of AXI blocks	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI3.2.1_2	Development of AXI components (RTL code)	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI3.2.1_3	Documentation	Mandatory									**Proposed**	Luca Bertaccini (ETH)		
WI3.2.1_14	The AXI interconnect shall be suitab	Best Effort									Proposed	SYSGO		

Description	Document Info	
State of the art	Work Item	*WI3.2.2*
Proposed solution	Author(s)	*UNIBO, , SYSGO*
Prerequisites	Version	*1.0*
Target TRL	Purpose (Use Case)	
Appendix	Development Flow	
Link at requirements file(s)	Short description of development flow	*Revision and improvement of the IP RTL. Standalone simulation for functional verification. System level integration, prototype and testing through RTL simulations and FPGA implementations.We expect to reach TLR 6. We will provide the IP with test-benches, to enable verification flows. We will provide the documentation of the IP as well.*
	Milestones	*M4 (Month 11) Functional blocks detailed specifications defined* *-Object 1* *M9 (Month 23) Functional blocks design and implementation complete* *-Object 2* *M15 (Month 34) Functional blocks test & design proving complete* *-Object 3*
	Verification & Validation	
	Overview of verification/validation setup	*Review on suitability for mixed-critical systems (SYSGO)*
	Appendix	

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI2.x.y_id_subids(optional)	<Describe here the requirement>		WI3.1.1_2_3		wi5.1.2, w5.2.7	wi5.1.2				**OnGoing**	**Proposed**			
WI3.2.2_HCI_1	Design a low-power scalable heterogeneous cluster interconnect that serves requests in one cycle of latency and offers high-bandwidth	Mandatory												
WI3.2.2_HCI_2	Extend the HCI described in I:Heterogeneous-cluster-interconnect_1 with Vector Lockstep Unit and Broadcast Units to enable optimized and reconfigurable MIMD/SIMD execution of the multi-core compute cluster	Optional												
WI3.2.2_HCI_11	The HCI shall be suitable for mixed-critical systems	Best Effort									**Proposed**	SYSGO		

Document Info	
Work Item	*03-02-2003*
Author(s)	*Klaus Hofmann, Jerome Martin, Lucien Dos Santos Farret*
Version	*0.2*
Purpose (Use Case)	
Description	*Low Power DDR Memory PHY to be used with the DDR controller developed in WI 3.2.5, in coordination with ANTM. This involves the design and implementation of an extreme low power LPDDR4X PHY exploiting digital design features, targeting GF22FDX for ASIC implementation. Using a novel, dominantly digital and synthesizable approach for precise timing conditioning, a test-friendly and power-efficient LPDDR4X PHY for min 800 Mb/p/s data transfer using 0,6V VDDQ will be designed. The output shall be a LPDDR4X PHY Model, RTL, Schematic, GDS2 and Testbench, whereby the IP blocks are preferrably open source.*
State of the art	*Proprietary LPDDR4X PHYs (and memory controllers), such as purchaseable from Synopsys and other companies, Open source LPDDR4X/LPDDR5 phy from waivious (targeting higher speeds and power usage)*
Proposed solution	*We intend to provide an as-open-source-as-possible, extreme power efficient PHY with a low pin count.*
Prerequisites	*Access to GF22FDX+ PDK*
Target TRL	*min. TRL7*
Proposal Respondance (SOs and KPIs)	*At least 20 distinct IP building blocks available via the virtual repository, with repository available and filled; all blocks with clear and unambigious licensing.* *Tristan will produce at least 23 IPs. In addition, at least 18 ISS Instruction set extensions will be delivered in the project. Furthermore, eight cores are to run in the demonstrators. It is envisioned that more than half of the IPs and extensions will be Open-Source.* *7 demonstrators in 6 different business domains delivered by the end of the project.*
Objects List	
List of generated objects	*IP block abstractions (e.g. schematic, layout, verification environment)*
Development Flow	
Short description of development flow	*Typically Cadence/Synopsys/SiemensEDA tools are used in conjunction with implementation PDK (currently targetting GF 22PDXplus)*
Milestones	*M1: Initial Requirements connected (M6)* *D3.1: Detailed specifications for RISC-V peripheral blocks (M11)* *D3.2: Design and implementation of RISC-V peripheral blocks (M23)* *D3.3: Functional Test and design proving of RISC-V peripheral blocks (M34)*
Verification & Validation	
Overview of verification/validation setup	*Classical ASIC development verification flow (schematic simulation, backannotated timing, layout verification (LVS, DRC, RCX), backannotated simulation, STA ...*
Appendix	
Link at requirements file(s)	*currently none*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.2.3_1	Low Power LPDDR4X PHY able to exchange data with a LPDDR4X-compliant DRAM devices	Mandatory								**OnGoing**	**Accepted**			
WI3.2.3_2	Low pin count number for PHY	Best Effort								**OnGoing**	**Proposed**			
WI3.2.3_3	PHY able to interface Single LPDDR4X DRAM Device (single chip) with 4Gbit capacity minimum	Mandatory	WI3.2.3_2							**OnGoing**	**Accepted**			
WI3.2.3_4	Required Data Transfer Rates for PHY: 800MT/s (= 400 MHz)	Mandatory								**OnGoing**	**Accepted**			
WI3.2.3_5	Required Data Transfer Rates for PHY: 1600MT/s (= 800 MHz)	Best Effort								**OnGoing**	**Proposed**			
WI3.2.3_6	Required Data Transfer Rates for PHY: higher than 1600MT/s (= 800 MHz)	Best Effort								**OnGoing**	**Proposed**			
WI3.2.3_7	PHY is able to interface with V_DDQ_nom = 0,6V DRAMs	Mandatory								**OnGoing**	**Accepted**			
WI3.2.3_8	PHY (resp. its abstractions (simulation, physical block, verification setup) is provided to 3rd party users as Open-Source IP	Best Effort				Tristan aims to provide RISC-V IP blocks as true open source. Since a PHY has always dependencies on the implementation (technology, PDK), the goal of WI3.2.3 is to provide the PHY "as open source as possible"				**OnGoing**	**Proposed**			
WI3.2.3_9	Emphasis on extreme high power efficiency / low power consumption	Best Effort								**OnGoing**	**Proposed**			

Document Info		
Work Item	*WI 3.2.4*	
Author(s)	*Juha Tulonen (Nokia), Tmo Hämäläine n(TAU)*	
Version	*0.1 (17.3.2023)*	
Purpose (Use Case)		
Description	*high-speed wireline transceiver IPs (PHY) for use with various types of channels ranging from very short die-to-die interconnects all the way up to backplane and cable interconnects*	
State of the art	*Previously (in Ballast) implemented serial interfaces*	meidän taiteen taso - ei maailman parasta
Proposed solution	*SERDES signal chain including tx buffering, coding, serialization, CDR, deserialization, decoding and rx buffering*	
Prerequisites	*None*	
Target TRL	*6*	
Proposal Respondance (SOs and KPIs)	*The WI provides key functionality to a modern integrated system-on-chip*	
Objects List		
List of generated objects	*Serial communication IPs for full link implementation*	
Development Flow		
Short description of development flow	*Literature review, high-level planning, HDL implementation + analog design, layout design, pin and package planning, tapeout, packaging, PCB design, software (driver) design, testing, reporting*	
Milestones		
Verification & Validation		
Overview of verification/validation setup	*During development, the structures will be constantly simulated and tested, as applicable. Finally the implemented system will be tested in real (laboratory) use scenario for verification and validation.*	
Appendix		

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.2.4_1	IF data rate 100Mbit/s (min)	Mandatory								**ToBeDone**	**Proposed**	Nokia		
WI3.2.4_2	Lane count 1 or more	Mandatory									**Proposed**	Nokia		
WI3.2.4_3	Enables cable interconnects (example same IP in two diferent IC able to handshake, distance > 10mm)	Mandatory									**Proposed**	Nokia		
WI3.2.4_4	Configurable multi-purpose high speed serial link (example different datarate)	Mandatory									**Proposed**	Nokia		
WI3.2.4_5	Enables Advanced modulation schemes (example NRZ - nonreturn to zero)	Best Effort									**Proposed**	Nokia		
WI3.2.4_6	Energy efficiency (example setting IP to idle state when not TX/RX)	Best Effort									**Proposed**	Nokia		
WI3.2.4_7	Enables die-to-die link (example same IP in two diferent IC able to handshake, distance 1mm...10mm)	Optional									**Proposed**	Nokia		
WI3.2.4_8	Enables high density constellations beyond PAM4	Optional									**Proposed**	Nokia		
WI3.2.4_9	Fully qualified IP (silicon and back-end testing)	Optional									**Proposed**	Nokia		
WI2.x.y_id_subids(optional)	<Describe here the requirement>		WI3.1.1_2_3		wi5.1.2, w5.2.7	wi5.1.2				**OnGoing**	**Accepted**			

Document Info	
Work Item	*WI 3.2.5*
Author(s)	*Klaus Hofmann, Jerome Martin, Karol Gugala*
Version	*0.1*
Purpose (Use Case)	
Description	*ANTM, GWT and TUDA will jointly focus on the development of an Open-Source Ultra-Low Power DDR Memory Digital Interface for ultra-low power RISC-V based SoCs targeting to reduce the power budget to a few tens of mW. The aim is to be able to handle workloads that mix predictive and dynamically streamed traffic. The configurable balancing interface with the PHY developed in WI 3.2.3 will be used. The outputs shall be a synthesizable RTL model.*
State of the art	*Commercial LPDDR4X Digital Interfaces (such as purchaseable from Synopsys and other companies)*
Proposed solution	*We intend to provide an as-open-source-as-possible, extreme power efficient LPDDR4X Memory Digital Interface for ultra-low power RISC-V based SoCs*
Prerequisites	*None*
Target TRL	*min. TRL7*
Proposal Respondance (SOs and KPIs)	*At least 20 distinct IP building blocks available via the virtual repository, with repository available and filled; all blocks with clear and unambigious licensing.* *Tristan will produce at least 23 IPs. In addition, at least 18 ISS Instruction set extensions will be delivered in the project. Furthermore, eight cores are to run in the demonstrators. It is envisioned that more than half of the IPs and extensions will be Open-Source.* *7 demonstrators in 6 different business domains delivered by the end of the project.*
Objects List	
List of generated objects	*Synthesizable description (e.g. Verilog) plus constraints and synthesis setup*
Development Flow	
Short description of development flow	*Functional coding based on requirements (JEDEC, DFI, ...)*
Milestones	*M1: Initial Requirements connected (M6)* *D3.1: Detailed specifications for RISC-V peripheral blocks (M11)* *D3.2: Design and implementation of RISC-V peripheral blocks (M23)* *D3.3: Functional Test and design proving of RISC-V peripheral blocks (M34)*
Verification & Validation	
Overview of verification/validation setup	*Functional simulation/verfiication. Testchip-TO and usage of IP in demonstrator (wearable)*
Appendix	
Link at requirements file(s)	*currently none*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.2.5_1	LPDDR4X Memory controller connectable to PHY (WI 3.2.3) via reduced DFI interface	Mandatory								**OnGoing**	**Accepted**			
WI3.2.5_2	LPDDR4X Memory controller connectable to testplatform via SPI, UART, JTAG interface	Mandatory								**OnGoing**	**Accepted**			
WI3.2.5_3	LPDDR4X Memory controller providing (preferrable programmable) state machine for PHY training	Mandatory								**OnGoing**	**Accepted**			
WI3.2.5_4	Required Data Transfer Rates from testplatform: 800MT/s (= 400 MHz)	Mandatory								**OnGoing**	**Accepted**			
WI3.2.5_5	Required Data Transfer Ratesfrom testplatform: 1600MT/s (= 800 MHz)	Best Effort								**OnGoing**	**Accepted**			
WI3.2.5_6	Memory controller (resp. its abstractions (simulation, physical block, verification setup) is provided to 3rd party users as Open-Source IP	Mandatory								**OnGoing**	**Accepted**			
WI3.2.5_7	Emphasis on extreme high power efficiency / low power consumption	Best Effort								**OnGoing**	**Accepted**			

Document Info	
Work Item	*WI3.2.6*
Author(s)	*CEA*
Version	*0,1*
Purpose (Use Case)	
Description	*need for an open source reference implementation of an OBI-compliant interconnect (for use with OHG CV32 cores family), suitable for embedded low-power/real-time applications*
State of the art	*no open source reference implementation available yet*
Proposed solution	*Provide an OBI-compliant library that allows to build a low-latency OBI interconnect, and a set of bridges to interface it with AMBA-compliant IPs (AHB, APB)* *this library will include documentation and sample verification test-suite*
Prerequisites	*AMBA and OBI specification vailable*
Target TRL	*TRL-5*
Proposal Respondance (SOs and KPIs)	*SO2 - Ecosystem of Industrial Quality SoC Building Blocks* *SO4 - Vendor Independence* *KPI: IP building block in virtual repository*
Objects List	
List of generated objects	*RTL code* *documentation* *sample testbench*
Development Flow	
Short description of development flow	*RTL development, functional verification*
Milestones	*MS1 : RTL Code verified with commercial VIPs using UVM testbed* *MS2 : RTL Code verified with open-source VIPs using UVM testbed*
Verification & Validation	
Overview of verification/validation setup	*UVM testbenches*
Appendix	
Link at requirements file(s)	*none*

Y1 -0.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
I:OBIXBAR-interconnect_1	Compatibility with OBI protocol version V1.1	Mandatory												
I:OBIXBAR-interconnect_2	Only default OBI configuration is supported	Mandatory												
I:OBIXBAR-interconnect_3	Up to 4 masters are supported	Mandatory												
I:OBIXBAR-interconnect_4	up to 24 slaves are supported	Mandatory												
I:OBIXBAR-interconnect_5	Full crossbar architecture	Mandatory												
I:OBIXBAR-interconnect_6	Low latency crossbar	Mandatory												
I:OBIXBAR-interconnect_7	Single clock design	Mandatory												
I:OBIXBAR-interconnect_8	Fixed priority arbitration	Mandatory												
		Mandatory												
I:AHB2OBI-interconnect_1	Compatibility with AMBA 3 AHB-Lite Protocol Specification v1.0	Mandatory												
I:AHB2OBI-interconnect_2	Compatibility with OBI protocol version V1.1	Mandatory												
I:AHB2OBI-interconnect_3	Only default OBI configuration is supported	Mandatory												
I:AHB2OBI-interconnect_4	AHB supported configuration limited to OBI default configuration comoatibilitv	Mandatory												
I:AHB2OBI-interconnect_5	AHB burst transfers support	Mandatory												
I:AHB2OBI-interconnect_6	AHB back to back transfers support	Mandatory												
I:AHB2OBI-interconnect_7	Low latency bridge	Mandatory												
I:AHB2OBI-interconnect_8	Single clock design	Mandatory												
		Mandatory												
I:OBI2AHB-interconnect_1	Compatibility with AMBA 3 AHB-Lite Protocol Specification v1.0	Mandatory												
I:OBI2AHB-interconnect_2	Compatibility with OBI protocol version V1.1	Mandatory												
I:OBI2AHB-interconnect_3	Only default OBI configuration is supported	Mandatory												
I:OBI2AHB-interconnect_4	AHB supported configuration limited to OBI default configuration compatibility	Mandatory												
I:OBI2AHB-interconnect_5	OBI back to back transfers support	Mandatory												
I:OBI2AHB-interconnect_6	Low latency bridge	Mandatory												
I:OBI2AHB-interconnect_7	Single clock design	Mandatory												
		Mandatory												
I:OBI2APB-interconnect_1	Compatibility with AMBA APB Protocol Specification, version C (aka "APB4", aka v2.0)	Mandatory												
I:OBI2APB-interconnect_2	Compatibility with OBI protocol version V1.1	Mandatory												
I:OBI2APB-interconnect_3	Only default OBI configuration is supported	Mandatory												
I:OBI2APB-interconnect_4	APB supported configuration limited to OBI default configuration compatibility	Mandatory												
I:OBI2APB-interconnect_5	OBI back to back transfers support	Mandatory												
I:OBI2APB-interconnect_6	Low latency bridge	Mandatory												
I:OBI2APB-interconnect_7	Single clock design	Mandatory												
I:OBI2APB-interconnect_8	Retiming stages on request and/or response paths	Optional												
		Mandatory												
I:OBI2OBI-interconnect_1	Compatibility with OBI protocol version V1.1	Mandatory												
I:OBI2OBI-interconnect_2	Only default OBI configuration is supported	Mandatory												
I:OBI2OBI-interconnect_3	Low latency bridge	Mandatory												
I:OBI2OBI-interconnect_4	Single clock design	Mandatory												

Document Info	
Work Item	*WI3.3.2*
Author(s)	*TECHNL*
Version	*0,1*
Purpose (Use Case)	
Description	*ST-FR, TECHNL will jointly work on side channel attack protection via hardware and software mechanisms, Exploration and Simulation. Test chip is possible for some mechanisms. The output will be several simulation Models and a FPGA/Test Chip to evaluate results.*
State of the art	*DOI:10.1145/3316781.3323485*
Proposed solution	*Explore side-channel attack mitigations by using hardware peripheral IP. These should make it harder for attackers to find attack surface by analysing currents in the chip or FPGA.*
Prerequisites	*none*
Target TRL	*TRL5*
Proposal Respondance (SOs and KPIs)	*SO2*
Objects List	
List of generated objects	*Simulation models, FPGA or test chip*

Development Flow	
Short description of development flow	*t.b.d. in a later revision*
Milestones	*t.b.d. in a later revision*
Verification & Validation	
Overview of verification/validation setup	*t.b.d. in a later revision*
Appendix	
Link at requirements file(s)	*Available after D3.1*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.3.2_1	The core checker RTL shall be written in VHDL.	Mandatory		cocotb						**ToBeDone**	**Proposed**	TECHNL		
WI3.3.2_2	The protection mechanism will be implemented in verilog possibly through HLS and part of the RISC-V CPU IP	Mandatory		Internal know-how, possibly using SNPS ASIP designer				Analysis to be done in a lab. IP not leaking information during the execution of some algorithms is successful validation.		**ToBeDone**	**Proposed**	ST-FR		

Document Info	Document Info	
Work Item	Work Item	*WI3.4.1*
Author(s)	Author(s)	*UNIBO*
Version	Version	*1.0*
Purpose (Use Case)	Purpose (Use Case)	
Description	Description	*UNIBO aims to develop a cluster-coupled accelerator to boost the performance of FP32 and FP16 matrix multiplication kernels. The IP will work within a power budget typical of low-end microcontroller based systems usually employed as IoT end-nodes. From our preliminary analysis we expect the IP to reach performance in the order of tens GFLOPS with an energy efficiency of hundreds GFLOPS/W, implemented with a leading edge GF22FDX technology. The resulting IP will be integrated into a computing cluster of RISC-V processors through a tighly-coupled low-latency interconnect (proposed as an IP of this WP3). The IP will come with UVM test-benches and a complete documentation. All the outcomes will be open-sourced.*
State of the art	State of the art	*Online finetuning and adaptation of general DL models on edge devices are still highly challenging. One of the key stumbling stones is the need for parallel floating-point operations, which are considered unaffordable on sub-100 mW extreme-edge SoCs.*
Proposed solution	Proposed solution	*Low power accelerator for low-end RISC-V based systems. Expected one order of magnitude better energy efficiency on FP MatMul kernels, compared to purely software execution.*
Prerequisites	Prerequisites	*None*
Target TRL	Target TRL	*Our target Technology Readiness Level (TRL) for this project is 6*
Proposal Respondance (SOs and KPIs)	Proposal Respondance (SOs and KPIs)	*S02: Ecosystem of industrial quality Soc Building blocks* *S04: Vendor independence* *S05: Active European Open-source HW community*
Objects List	Objects List	
List of generated objects	List of generated objects	*1. Specifications of the designed cluster-coupled accelerator* *2. RTL code of designed cluster-coupled accelerator* *3. Testbench for functional verification*
Development Flow	Development Flow	
Short description of development flow	Short description of development flow	*Revision and improvement of the IP RTL. Standalone simulation for functional verification. System level integration, prototype and testing through RTL simulations and FPGA implementations. We expect to reach TLR 6. We will provide the IP with test-benches, to enable verification flows. We will provide the documentation of the IP as well.*
Milestones	Milestones	*M4 (Month 11) Functional blocks detailed specifications defined* *-Object 1* *M9 (Month 23) Functional blocks design and implementation complete* *-Object 2* *M15 (Month 34) Functional blocks test & design proving complete* *-Object 3*
Verification & Validation	Verification & Validation	
Overview of verification/validation setup	Overview of verification/validation setup	*n.a.*
Appendix	Appendix	
Link at requirements file(s)	Link at requirements file(s)	*n.a.*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.4.1	Design low-power parametric accelator to boost the performance and efficiency of general-purpose Matrix-Matrix operations, targeting multi-precision Floating-point formats (32-bit, 16-bit, 8-bit)	Mandatory												

Document Info	
Work Item	*3.4.2*
Author(s)	*Jérôme FEREYRE (CEA)*
Version	*1,0*
Purpose (Use Case)	
Description	*Many applications in different scientific domains use linear algebra kernels, such as linear solvers or eigensolvers.* *The continuously growing size of the systems of linear equations has led researchers to use iterative methods (such as Krylov subspace projective methods) where the working memory space is o(N) where N is the size of the matrix. Direct solvers result in a working space that is the same as the matrix (O(n2)), and are no longer used for large systems. With these iterative/projective approaches, the matrix A is not modified.* *Unfortunately, these iterative/projective methods are very sensitive to numerical instability because of round-off errors, and, in some cases may even prevent the methods to converge. To increase stability, there are two solutions:* *- apply mathematical transformations, such as preconditioning or reorthogonalization, to the problem matrix; or* *- use extended precision (this is floating-point numbers with a precision greater than 64 bit double).* *In some cases, the first option may be unpractical because of the memory or computational cost. Moreover, applying these techniques needs significant effort and must tailored to the problem at hand.* *The latter solution, is easier and more scalable, but currently, software libraries that provide extended precision on computers with only hardware support for double (64-bit) introduce a very significant run-time overhead.* *The VXP (formerly called VRP) is a hardware accelerator, designed by the CEA(France), based on the RISCV ISA, which provides hardware support for extended precision floating point numbers. [1][2]. This support is provided through custom Instruction Set Architecture (ISA) extensions for loading, storing and manipulating extended precision numbers.* *To use the VXP, programmers need software tools to enable them to generate object code which uses the new instructions. The CEA has developed versions of the standard BLAS libraries that work with the variable precision hardware, and this would be the typical entry point for programmers, although it is also possible for programmers to directly manipulate variable precisions scalars using C macros that are also provided.*
State of the art	*The VRP tools currently exist at the CEA at a TRL3-4 level. During the course of Tristan they will be enhanced to reach TRL7. This includes, packaging for external use, improved documentation, user testing.*
Proposed solution	*Different use models are possible. First, if the user has access to the CEA's VXP hardware, then the tool-flow will generate RISC-V code using the new custom instructions. However, in the Tristan project, the CEA is also making available a software flow which enables users to compile a program using variable precision and run it on a standard Linux machine (x86 or ARM), so that users can evaluate the benefit that variable precision can bring to their numerical algorithms, without necessarily having access to the hardware. There are two ways to run VXP code on a Linux host. In one case, the user compiles RISC-V object code that is executed on a modified RISC-V Spike model. Alternatively, the user can compile the code directly to the target architecture and execute the variable precision operations using the standard MPFR library. All three software flows are supported.*
Prerequisites	** RISC-V Run-time (supporting malloc/printf). The package will contain such a run-time, but user can provide their own.* ** Standard BLAS and MPFR (v4.1..0) libraries. If BLAS libraries are not present, user can still issue VRP scalar commands.* ** gcc (version 9.2.0 or later)* ** RTL implementation of VRP (proprietary) - only for the case of running on the VRP proprietary hardware.*
Target TRL	*TRL7*
Proposal Respondance (SOs and KPIs)	** Code base publicly available on git-hub* ** Compatible with Ubuntu 20.04 and RHEL 8.X.*
Objects List	
List of generated objects	*<List here the planned objects delivered>*
Development Flow	
Short description of development flow	** The specification of the VRP SDK will be written (done)* ** The SDK will be completed tested internally at CEA* ** A request for a project with OpenHW Group will be created* ** The SDK will be made available externally*
Milestones	** MS1 = SDK Specification Complete* ** MS2 = SDK ready internally* ** MS3 = First Open Source Release* ** MS4 = Final Open Source Release*
Verification & Validation	
Overview of verification/validation setup	** The SDK will be provided with several example applications.* ** For each external release, it will be verified that these example applications run on each of the target platforms*
Appendix	
Link at requirements file(s)	*<...>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
I:VXP_ACCEL_REQ_1	The Spike model must be compatible with the software flow defined in WP4 (WI4.2.1 and WI4.4.1)	Mandatory								**OnGoing**	**Accepted**			
I:VXP_ACCEL_REQ_2	The Spike model must be sufficiently fast to allow executing long applications (1e9 instructions) on a desk-top Linux host.	Optional												
I:VXP_ACCEL_REQ_3	The spike model must provide a bit-accurate representation of the operation of the RTL implementation of the VRP. Note RTL code is not delivered in the scope of TRISTAN.	Mandatory												
I:VXP_ACCEL_REQ_4	The VXP accelerator will run on Ubuntu 20.04 and RHEL 8.X	Mandatory												

Document Info	
Work Item	*WI3.4.4*
Author(s)	*POLITO*
Version	*1,0*
Purpose (Use Case)	
Description	*PQC algorithms introduce new mathematical operations which are not easy to implement on common processor. POLITO aims to develop loosely coupled RISC-V accelerators able to boost the performance of these algorithms. The accelerators will be integrated into a low-end microcontroller system including a tiny RISC-V core (e.g. CV32E40P). The supported operations will be selected among the most time consuming functions present in the final post-quantum crytography algorithms identified by the NIST (National Institute of Standards and Technology), such as the Number Theoretic Transform and the Keccak family of cryptographic primitives.* *The developed accelerators will come with synthesizable and verified models, supported by complete documentation.*
State of the art	*It was proved that current public key cryptosystems (e.g. RSA and ECC) are made breakable by the advent of quantum computers. Therefore, post-quantum cryptosystems are urgently needed to reinforce security in the quantum computing era. Fully dedicated hardware accelerators for PQC computation are feasible, but they lack of the needed flexibility to support future algorithms. Short execution time and flexibility can be jointly achieved by means of properly designed accelerators coupled to a programmable core.* *Current state of the art includes several models of hardware accelerators for specific PQC algorithms, e.g. loosely coupled accelerators and tighly couple accelerators, exploring mapping of cryptographic primitives over ISA extensions, shared hardware resources and configurability.*
Proposed solution	*We propose loosely coupled accelerators supporting multiple critical functions of NIST round-3 candidates (e.g. CRYSTALS-KYBER). Advantages offered by the loosely coupled solution include the possibility to adapt the designed accelerator to different cores, the high speed performance, and the better robustness with respect to the security issues. On the other hand, a tightly coupled accelerator is intrisically more flexible than a loosely coupled co-processor and therefore supporting multiple standards over the same loosely couple accelerators is a challenging task.*
Prerequisites	*None*
Target TRL	*Our target Technology Readiness Level (TRL) for this project is 7.*
Proposal Respondance (SOs and KPIs)	*S02: Ecosystem of industrial quality Soc Building blocks* *S04: Vendor independence*
Objects List	
List of generated objects	*1. Specifications of the designed accelerators* *2. Synthesizable RTL code of designed accelerators* *3. Testbench for functional verification*
Development Flow	
Short description of development flow	*1. Specifications of the designed accelerators* *2. RTL code of designed accelerators* *3. Testbench for functional verification* *4. Synthesis and P&R* *5. Post synthesis verification*
Milestones	*N/A*
Verification & Validation	
Overview of verification/validation setup	*N/A*
Appendix	
Link at requirements file(s)	*N/A*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.4.4_1	The official software models specified by the NIST must run on the accelerated RISC-V platform, optimizing them for RISC-V applications.	Mandatory		Synopsys Design Compiler and FPGA synthesis tools.							**Proposed**	POLITO		

Y1-0.1.1			Y2						Y3			
Requirement ID	**Requirement Description**	**Priority**	**WI Requirements Interdependecies (Cores, HW IPs, SW IPs, Demonstrators)**	**EDA Tools Used (Commercial)**	**TRISTAN Tools Used**	**Which Need Requirement?**	**Related Specification**	**Means of Validation**	**Validation Results**	**Validation Status**	**Requirement Status**	**Proposed by**
WI3.4.5-1	eFPGA IP shall have single clock domain.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-1.1	All modes shall use the same clock.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-2	eFPGA IP shall have single reset domain.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-2.1	All synchronous logic elements shall have active-low reset.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-2.2	All synchronous logic elements shall have synchronous reset.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-3	eFPGA IP shall have single power domain.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-4	eFPGA IP shall have a tile-based architecture.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-5	Logic tiles shall have LUTs, registers, and connection blocks.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-5.1	eFPGA IP shall have 10000 logic tiles.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-5.2	Each logic tile shall include 8 logic blocks.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-5.3	Each logic block shall include a 4-input LUT.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-5.4	Each logic block shall include two FFs.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-5.5	Each logic block shall have multiple operating modes.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-6	I/O tiles shall support bidirectional operation.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-6-1	I/O tiles shall have multiple operating modes.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-6-2	I/O tiles shall be used to interface with system bus.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-6-3	I/O tiles shall be used to interface with external IOs.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-6-4	I/O tiles shall be used to interface with control IOs.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-6-5	Number of I/Os to interface with the system bus shall be 100.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-6-6	Number of external I/Os shall be 8.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-6-7	I/O tiles shall be placed at the boundary of IP.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-7	eFPGA IP shall have a system bus interface.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-7-1	Bus interface shall be AMBA AXI4-Lite compliant.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-7-2	Address width of read and write channels shall be 8.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-7-3	Data width of read and write channels shall be 32.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-8	Memory tiles shall have memory blocks.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-8-1	Memory blocks shall have different operating modes.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-8-2	Size of total memory blocks shall be at least 180 Kbits.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-8-3	Memory blocks shall have single-port SRAM macros.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-8-4	Memory blocks shall support write-first policy.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-8-5	Memory blocks shall support synchronous read.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-9	DSP tiles shall include MAC blocks.	Optional	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-9-1	MAC blocks shall have two 18-bit inputs and output 36-bit output.	Optional	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-9-2	Number of total DSP tiles shall be 8.	Optional	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-10	eFPGA IP shall have a configuration logic to program the FPGA fabric.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-10-1	Each tile shall have configuration circuitry to define the interconnection and wiring in tile components.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-10-2	Switch blocks shall provide the interconnect between the tiles.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-10-3	All configurable elements shall implement a configuration chain in the FPGA fabric.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-10-4	eFPGA IP shall have a JTAG configuration interface.	Best Effort	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-11	eFPGA IP shall have a test mode.	Best Effort	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-11-1	All FFs in located in the logic tiles shall be controllable and observable via a scan chain.	Best Effort	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-11-2	Each SRAM macro shall have BIST circuitry.	Best Effort	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-12	eFPGA IP shall have a synthesizable RTL design.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-12-1	RTL design shall be SystemVerilog-2005 compliant.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-12-2	RTL design shall not contain any primitives that belong to a 3rd party except the functional model of SRAM	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-13	eFPGA IP shall be verified by functional simulation.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-13-1	Code coverage shall be at least 90%.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-13-2	Functional coverage shall be 100%.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-14	Achievable maximum clock frequency shall be at least 50MHz in TSMC 65nm technology.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-15	eFPGA IP shall be supported by a software stack.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-15-1	Software driver shall provide low-level routines for the CPU to access to the eFPGA IP.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-15-2	User-level functions shall provide access to the eFPGA IP for the applications running on the CPU core.	Mandatory	TBD	TBD	TBD	TBD	TBD	TBD	TBD	OnGoing	Proposed	YNGA
WI3.4.5-16												
WI3.4.5-17												
WI3.4.5-18												
WI3.4.5-19												
WI3.4.5-20												

Document Info	
Work Item	*WI3.4.5*
Author(s)	*YNGA:* *1. Abdullah Yildiz* *2. Rifat Demircioglu* *3. Hanim Ay* *4. Deniz Kurt* *5. Ugur Nezir*
Version	*0,1*
Purpose (Use Case)	
Description	*<Describe here the major use-cases of the WI>* *An embedded FPGA IP that can be used to offload certain processor tasks and improve overall system performance. eFPGA IP will have a common interface such as AXI, which will provide the RISC-V processor to configure and access the IP. eFPGA IP will also be supported by a software stack running on the RISC-V core to manage complex operations. Integration of an eFPGA IP into a RISC-V based SoC product will improve its market access for domain-specific applications.*
State of the Art	*<Describe here the state of the art that will be used as reference>* *There have been limited work in the literature which focuses on eFPGA IPs targeting RISC-V based systems. Existing solutions do not provide a turnkey product.*
Proposed solution	*<Describe here the proposed solution and ambition>*
Prerequisites	*<List here any pre-requisite to start the activity>*
Target TRL	*TRL 5*
Proposal Respondance (SOs and KPIs)	*SO2 - Ecosystem of Industrial Quality SoC Building Blocks* *SO4 - Vendor Independence* *SO5 - Active European Open-Source Hardware Community* *SO6 - Demonstration of Building Block Interoperability*
Objects List	
List of Generated Objects	*<List here the planned objects delivered>*
Development Flow	
Short Description of Development Flow	*<Provide here an overview on how the WI will be developed>*
Milestones	*<Descibe milestones and connected list of objects>*
Verification & Validation	
Overview of Verification/Validation	*<Provide here an overview on how the WI will be verified and/or validated>*
Appendix	
Link at Requirements File(s)	*<...>*

Document Info	
Work Item	*WI3.4.6*
Author(s)	*Jean-Christian KIRCHER (BOSCH-FR)*
Version	*v1.0*
Purpose (Use Case)	
Description	*NN IP to process inferences of neural network*
State of the art	*edge AI commercial IPs*
Proposed solution	*A module composed of a processing module, a DMA and an interrupt controller, with some scratchpad memory*
Prerequisites	
Target TRL	*TRL 5*
Proposal Respondance (SOs and KPIs)	
Objects List	
List of generated objects	*A HW IP that can be added to a CVA6 core to build a demonstrator*
Development Flow	
Short description of development flow	
Milestones	
Verification & Validation	
Overview of verification/validation setup	
Appendix	
Link at requirements file(s)	*<…>*

Y1 -D.1.1			Y2						Y3					
Requirement id	**Requirement Description**	**Priority**	**WI requirements interdependecies (cores, HW Ips, SW Ips, Demonstartors)**	**Eda Tool (commercial)**	**Tristan Tools**	**Which need requirement?**	**Related specification**	**Means of validation**	**Validation results**	**Validation Status**	**Requirement Status**	**Proposed by**		
WI3.4.6_0001	NN IP shall provide the integrator with the possibility to connect via an AXI bus for data transfers. This shall be the data interface of the NN IP.		WI3.1.1_2_3		wi5.1.2, w5.2.7	wi5.1.2				**OnGoing**	**Accepted**			
WI3.4.6_0002	NN IP shall provide the integrator with the possibility to connect via an AXI bus for control/configuration signals. This shall be the control interface of the NN IP.													
WI3.4.6_0003	The SoC shall provide the possibility to connect interrupt lines from the NN IP to the SoC for interrupts, e.g. computation results ready.													
WI3.4.6_0004	The SoC shall provide the possibility to connect interrupt lines from the NN IP to the SoC for safety interrupts.													
WI3.4.6_0005	NN IP should provide the possibility to connect external interrupt lines from the SoC to the NN IP, e.g. signal the abortion of the current inference.													
WI3.4.6_0006	NN IP shall integrate an AXI bus with a data bit width of, at least, 128 bits.													
WI3.4.6_0007	NN IP shall integrate an AXI interface that can support up to 16 outstanding transactions.													
WI3.4.6_0008	NN IP shall provide the integrator with the possibility to configure the local shared SRAM memory.													
WI3.4.6_0009	The HKP shall have the possibility to access the local shared SRAM memory.													
WI3.4.6_0010	The DMA shall have the possibility to access the local shared SRAM memory.													
WI3.4.6_0011	NN IP shall provide the integrator with the possibility to monitor the current NPU execution.													
WI3.4.6_0012	The SoC shall provide the NN IP the possibility to read from and write into an external memory. The memory can be located either inside or outside the SoC.													
WI3.4.6_0013	NN IP shall provide the integrator with the ability to reset its logic with active low reset signals issued asynchronously with regard to clocks.													
WI3.4.6_0014	The SoC shall provide the NN IP a core clock for the HKP, independent/asynchronous from the other clock.													
WI3.4.6_0015	The SoC shall provide the NN IP a clock for the rest of the design, independent/asynchronous from the HKP clock.													
WI3.4.6_0016	NN IP should provide the possibility to divide clocks.													
WI3.4.6_0017	NN IP shall provide the integrator with the ability to integrate it in a unique power domain.													