Project:	ISOLDE: customizable Instruction Sets and Open Leveraged Designs of Embedded riscv processors
Reference number:	101112274
Project duration:	01.05.2023 - 30.04.2026
Work Package:	WP1: Requirements and Specifications
Deliverable	**D1.2**
Title	**Requirements and specifications on architecture, hardware and software modules and IPs**
Type of deliverable:	Report
Deadline:	31.10.2023
Creation date:	12.09.2023
Dissemination Level:	PU - Public

Authors: Antonio Sciarappa, LDO

Rafael Tornero, UPV
Daniel Gracia Pérez, TRT
Stefan Wallentowitz, HM
Georg Pacher, NXP-AT
Erik Kraft, NXP-AT
Samu Ardaya-Lieb, CONS
Holger Blasum, SYSGO
André Sintzoff, TDIS
Nils Wessman, GSL
Andrea Galimberti, POLIMI
William Fornaciari, POLIMI
Davide Zoni, POLIMI
Federico Reghenzani, POLIMI
Giovanni Agosta, POLIMI
Alessio Burrello, POLITO
Daniele Jahier Pagliari, POLITO
Jan Kaštil, CODA

Václav Šimek, BUT
Petr Fajmon, NXP-CZ
Behnam Razi Perjikolaei, OFFIS
Dominik Riemer, BYK
Esther Soriano, FEN
Christopher Blochwitz, UZL
Jaume Abella, BSC
Juan Antonio Rodríguez Gracia, BSC
Xavier Carril Gil, BSC
A. Wilson, D. Gigena-Ivanovich, W. Krenn, M. Freiberger, SAL
Sebastian Reiter, FZI
Mihai Munteanu, Alexandru Drimbarean, Honorius Galmeanu, XPERI
Paolo Serri, TASI
Matteo Perotti, ETHZ
Simona Costinescu NXP-RO
Calin Bira UNSTPB

Involved partners:	Leonardo S.p.A. (LDO)
	Universitat Politècnica de València (UPV)
	Thales Research & Technology (TRT)
	Hochschule München University of Applied Sciences (HM)
	NXP Semiconductors Austria GmbH & Co KG (NXP-AT)
	NXP Semiconductors Czech Republic, s.r.o. (NXP-CZ)
	SYSGO GmbH (SYSGO)
	Consolinno Energy GmbH (CONS)
	Thales DIS France SAS (TDIS)
	Frontgrde Gaisler (GSL)
	Politecnico di Milano (POLIMI)
	Politecnico of Turin (POLITO)
	Codasip SRO (CODA)
	Bytefabrik.AI GmbH (BYK)
	OFFIS e.V. (OFFIS)
	Fent Innovative Software Solutions S.L. (FEN)
	University of Lübeck (UZL)
	Barcelona Supercomputing Center (BSC)
	Silicon Austria Labs GmbH (SAL)
	Xperi FotoNation SRL (XPERI)
	FZI Forschungszentrum Informatik (FZI)
	Brno University of Technology (BUT)
	Thales Alenia Space Italia S.p.A. (TASI)
	Eidgenössische Technische Hochschule Zürich (ETHZ)
	NXP Semiconductors Romania (NXP-RO)

POLITEHNICA București National University for Science and Technology (UNSTPB)

Responsible partner: Antonio Sciarappa, LDO, antonio.sciarappa@leonardo.com

Change Records

REV	DATE	§ CHANGE RECORDS	AUTHOR
0.1	12/09/2023	Initial draft template	Antonio Sciarappa
0.2	19/10/2023	First version	All partners
0.3	27/10/2023	Second version	All partners
0.4	03/11/2023	Review	Aurel Gontean, Danut Rotar, Stefan Wallentowitz
0.5	10/11/2023	Comments solved	Antonio Sciarappa, all partners
0.6	24/11/2023	Deliverable cleaned	Antonio Sciarappa

Executive Summary

This Deliverable reports an initial list of requirements and specifications for the various components developed in the context of the ISOLDE project relative to core architecture, hardware and software modules. In addition, this Deliverable provides a first indication about the IP corresponding to each component. A consolidated version of this document will be produced as a follow-up Deliverable D1.4. All partners involved in Tasks 1.2, 1.3 and 1.4 contributed to this Deliverable.

1Introduction

Working in the spirit of a hardware / software co-design approach, Deliverable D1.2 “Requirements and specifications on architecture, hardware and software modules and IPs” provides an initial list of requirements on the different components developed in the context of the ISOLDE project, building on the previous D1.1 “Demonstrator requirements and specifications”.

The document is organized as follows:

- Section 2 focuses on the architectures, I.e. the components developed in WP2, whose requirements have been collected as part of the work done in Task 1.2;
- Section 3 focuses on hardware / accelerators, I.e. the components developed in WP3, whose requirements have been collected as part of the work done in Task 1.3;
- Section 4 focuses on software, I.e. the components developed in WP4, whose requirements have been collected as part of the work done in Task 1.4.

Each Section is divided according to the Tasks of the associated WP. For each Task, all partners involved contributed by inserting specific sub-subsections for each individual component developed. Each component is uniquely named and is described according to the following structure:

- An initial short description of the component;
- A brief motivation detailing why the component may be needed;
- An indication of the tentative demonstrator (Aerospace, Automotive, Smart Home, IoT) the component should be used for;
- A list of partners involved in the development of the component;
- The license associated to the component;
- List of dependencies to previous / other components;
- List of requirements, identified by a unique id and supplemented by means of verification and type of requirement (**PR:** Performance, **F:** Functional, **PH:** physical, **STD:** compliance to standards);
- Optional additional comments.

This Deliverable is intended to be an initial version of the requirements; a consolidated list of requirements will be provided in a later Deliverable D1.4.

1.1 Definitions and Acronyms

Abbreviation	**Description**
AI	Artificial Intelligence
API	Application Programming Interface
AXI	Advanced eXtensible Interface
BCFI	Backward-edge Control Flow Integrity
BSP	Board Support Package
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CTM	Cryptographically Tagged Memory
DSE	Design Space Exploration
EDA	Electronic Design Automation
EMI	Enclave Memory Isolation
FCFI	Forward-edge Control Flow Integrity
FPGA	Field-Programmable Gate Array
FPU	Floating Point Unit
HAL	Hardware Abstraction Layer
IDE	Integrated Development Environment
IP	Intellectual Property
ISA	Instruction Set Architecture
JIT	Just-in-Time
ML	Machine Learning
NN	Neural Network
NoC	Network-on-Chip
PMC	Performance Monitoring Counter
PQC	Post Quantum Cryptography
RISC-V	Specific Open-Sourced RISC ISA
RTL	Register Transfer Level
SAL	Software Abstraction Layer
SIMD	Single Instruction Multiple Data
SoC	System on Chip
TPU	Tensor Process Unit
VM	Virtual Machine
VPU	Vector Process Unit
WCET	Worst-Case Execution Time
XNG	XtratuM Next Generation

2 Cores, interfaces, peripherals & NoCs

This section will focus on requirements and specifications definition related to the extension of the processor cores, interfaces, peripherals and Network-on-Chip (NoC) IP cores developed in the scope of the ISOLDE project. A set of requirements will be generated for the extension of the processor cores (CVA6 and NOEL-V) and SoC infrastructure (interconnect and peripheral) that are needed for the demonstrator designs or otherwise proposed or planned to be developed within ISOLDE.

2.1 Processors

2.1.1 NOEL-V processor extensions

DESCRIPTION
The NOEL-V processor will be extended to support the following features: • Support for shadow stack (Zicfiss) and Landing pad (Zicfilp) RISC-V extensions. • Extend NOEL-V processor system to support trusted execution environment. • Generate IP-XACT description for the NOEL-V subsystem.

MOTIVATION
To support the increased demand for security in the space domain Control-flow Integrity (CFI) and trusted execution environment (TEE) are key features. Control-flow Integrity (CFI) capabilities help defend against Return-Oriented Programming (ROP) and Call/Jump-Oriented Programming (COP/JOP) attacks.

TARGET DEMONSTRATORS
Space, Automotive

IMPLEMENTERS
GSL

LICENSE
GPL

LINK TO OTHER MODULES AND IPs
TBD

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
NV-CFI-01	STD	The ratified version of the Zicfiss and Zicfilp shall be implemented.	Simulation (RISC-V architecture test).
NV-TEE-01	F	Support isolated execution.	Simulations and FPGA prototyping.

NV-CRY-01	F	Support RISC-V Cryptography Extensions.	Simulations and FPGA prototyping.
NV-IPX-01	STD	The IP description shall be done using the IP-XACT standard.	Integration in EDA tools supporting IP-XACT components.

COMMENTS
The CFI RISC-V standard is not ratified at the time these requirements are specified. The process to define a RISC-V standard for TEE (similar to WorldGuard) is at this time still ongoing.

2.1.2 CVA6 processor extensions

DESCRIPTION
The CVA6 processor will be extended to support the following features: • Support for post quantum cryptography extensions.

MOTIVATION
To support the increased demand for security due to quantum computing threat, support of post quantum cryptography is a must for the future.

TARGET DEMONSTRATORS
Smart home

IMPLEMENTERS
TDIS

LICENSE
Permissive open source (Solderpad)

LINK TO OTHER MODULES AND IPs
TBD

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
CVA6-PQC-01	STD	Instructions to support PQC algorithms standardised by NIST shall be implemented.	Simulation

COMMENTS
Depending on the instructions that will be defined in future RISC-V PQC extension, support of additional custom instructions can be provided.

2.1.3 CV32E40X processor extensions

DESCRIPTION
The goal is to accelerate bytecode virtual machines, in particular WebAssembly. For that, the CV32E40X processor will be extended to support the following features: • Support for 32-bit Pointer Masking (Xjpm32) RISC-V extensions. • Memory isolation for bytecode virtual machines • Computed gotos for bytecode virtual machines • Hardware-assisted stacks for bytecode virtualization • Microcode-generator for bytecode VMs

MOTIVATION
Bytecode Virtual Machines provide excellent isolation and portability properties. WebAssembly is an interesting bytecode VM for embedded use cases too. JIT techniques are not applicable on resource-constrained devices, and the acceleration of Interpreters is desirable.

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
HM

LICENSE
MIT

LINK TO OTHER MODULES AND IPs
– None

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
BYTEC-JPM-01	STD	Draft proposal of 32-bit Pointer Masking.	Simulation (RISC-V architecture test).
BYTEC-GOTO-01	F	Method for computed goto support for WebAssembly	Simulations and FPGA prototyping.
BYTEC-HWS-01	F	Method for hardware-assisted VM stack	Simulations and FPGA prototyping.
BYTEC-BYTE-01	F	Method for microcode generation from re-configurable microcode memory	Simulations and FPGA prototyping.

COMMENTS

2.1.4 Testing Design Parameters for CVA6

DESCRIPTION
Creation of CVA6 of different multicore architectures by systems-on-chip generator tools. For this purpose, various design parameters are systematically created according to the specifications of T1.3/T1.4 and compared with each other using various design metrics (computing power, power consumption, etc.) and the suitable design parameters are identified.

MOTIVATION
Gain Information about proper design parameters for RISC-V SoCs

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
UZL

LICENSE
Open Source (TBD)

LINK TO OTHER MODULES AND IPs
-

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
DSE-SOC-01	P	SoC Design Parameters	Simulation and FPGA-Prototype

COMMENTS

2.1.5 Analysis Framework (IFX WI2.1.1-2.1.3)

DESCRIPTION
Analysis Framework for RISC-V Cores

MOTIVATION
Even if there are broad benchmarks as DMIPS widely used, cores normally have their strength and weakness. A more detailed analysis shall give better insights. Further, processor performance depends on SoC or CPU Subsystem Architectures an Compiler usage, also these influences shall be measured.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
IFX

LICENSE
Proprietary

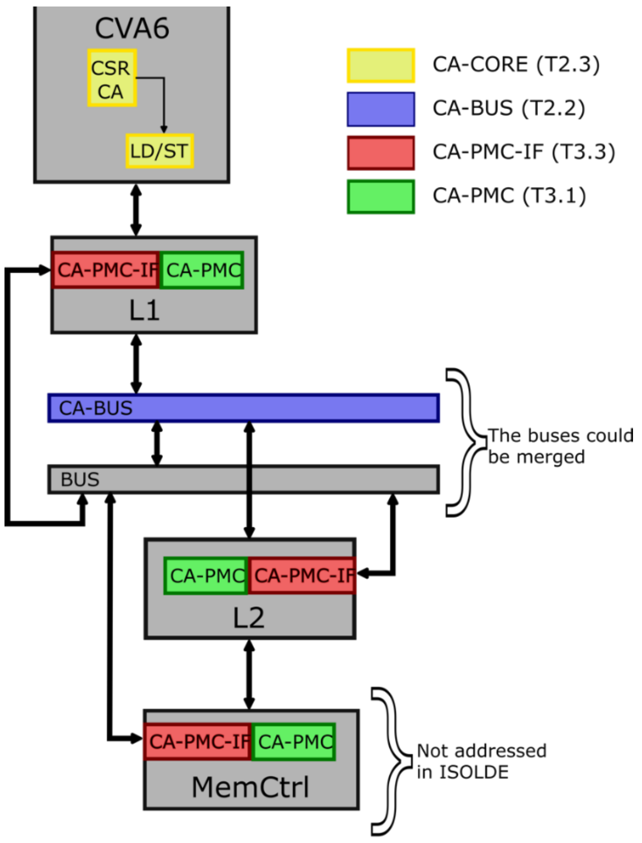
LINK TO OTHER MODULES AND IPs

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
AHBcb-01	F/STD	Have a list of features and variants to be analyzed	Review
AHBcd-02	F	Have a list of testcases that form the analysis framework	Review
AHBcb-03	F	Have a concept and use mode for the analysis framework	Work
AHBcd-04	F	Have a fully automated analysis	Simulation of interrupt scenarios

COMMENTS
Follow-up RISC-V Standardization

2.2 Peripheral and interconnect

2.2.1 Context-Aware Bus (CA-BUS)

DESCRIPTION
Extend the system bus (or interconnect) with execution context information associated to the request to enable the integration of context-aware PMCs (CA-PMCs). Figure below shows the complete context-aware monitoring infrastructure and the position of the CA-BUS in its design. 

MOTIVATION
Performance Monitoring Counters (PMCs) are essential tools for the verification and monitoring of a system. Safety systems heavily rely on them to implement different safety features. However, they are typically limited, collecting information of every event happening, which puts constraints or limitation on the implementation of the safety mechanisms and/or the system itself. Context-Aware PMCs (CA-PMCs) and the associated infrastructure (CA-CORE, CA-BUS and CA-PMC-IF) will enhance current PMCs and enable more efficient safety verification and monitoring of the system. CA-PMCs will allow the software (typically the OS/Hypervisor) to define the events that need to be collected. CA-PMCs require the context information to operate, and the context depends on each request/event. The system bus represents a critical IP to transfer such information. The CA-BUS extension will enable the transfer of the context information added to each core request with the CA-CORE (see Section 2.3 – CA-CORE) to the CA-PMCs (see Section 3.3 – CA-PMC).

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
TRT

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
– CVA6 core developed in T2.1 (see Section 2.1) – CA-PMCs developed in T3.3 (see Section 3.3)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
CA-BUS-01	STD	The implementation should be compatible with the AXI standard.	Compatibility with SoC FPGA design using AXI system bus.
CA-BUS-02	PH	The size of the context information should be configurable at design time.	Synthesis of designs with different context information sizes.

COMMENTS

2.2.2 Wormhole NoC

DESCRIPTION
Develop a Wormhole Network-on-Chip with support for Virtual Networks and Virtual Channels and extensions for monitoring and debugging

MOTIVATION
On the one hand, NoCs are scalable interconnect solutions to SoC designs requiring many IP cores. On the other hand, a SoC design could require more than one NoC. In this case, there are basically two options to address that aspect in NoC design: I) the designer can replicate the NoC physically, keeping a simple router. II) the designer can opt to a more complex router with support for VNs and VCs, such that there is a single physical NoC. In general, the combination of VNs/VCs will help to diminish network contention and provide a good degree of Quality of Service, while maintaining area usage for the interconnect controllable.

TARGET DEMONSTRATORS
TBD

IMPLEMENTERS
UPV

LICENSE
MIT (or another permissive license)

LINK TO OTHER MODULES AND IPs
Not applicable. The IP does not rely on any other tasks

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
WormNoC-01	F/STD	Support for AXI interfaces	Simulation of external IPs exchanging data through the network using standard AXI interfaces
WormNoC-02	F	Bandwidth reservation to support different levels of QoS	Simulation testbench using synthetic traffic
WormNoC-03	F	Physical traffic isolation	Simulation testbench using synthetic traffic
WormNoC-04	F	Support for monitoring traffic-related metrics	Simulation testbench using synthetic traffic

COMMENTS

2.2.3 AXI Sniffer

DESCRIPTION
Hardware module to monitor AXI traffic

MOTIVATION
Safety-related applications require deriving execution time bound for the different software components. This is challenged by the existence of timing interferences in heterogeneous multicore SoCs. Timing monitoring and enforcement is a well-known approach to control execution time interference. The AXI sniffer will enable extracting request-level interference in interconnects usings AXI interfaces.

TARGET DEMONSTRATORS
TBD

IMPLEMENTERS
UPV

LICENSE
MIT (or another permissive license)

LINK TO OTHER MODULES AND IPs
Not applicable. The IP does not rely on any other tasks

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
AXISniffer-01	F/STD	Seamless Integration in AXI-based Interconnects	Integration and validation of the module in complete SoC demonstrators
AXISniffer-02	F	Communication with global monitoring units	Integration and validation of the module in complete SoC demonstrators

COMMENTS

2.2.4 AHB Crossbar (IFX WI2.2.1-2.2.3)

DESCRIPTION
Develop an AHB crossbar generally usable for different number of initiators and responders. *Please note "master" and "slave" has been widely used, fortunately more and more these terms are replaced. I used here "initiator" and "responder". As these terms are not fully aligned, we should agree in Isolde on consistent names.*

MOTIVATION
AHB is the standard bus used

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
IFX

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
IPs that want to connect to the crossbar must follow AHB specification, some adjustment/classification of AHB interfaces may be needed.

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
AHBcb-01	F/STD	Support for AHB interfaces	Simulation of crossbar stand alone and with connected IPs
AHBcd-02	F	Different number of Initiators and Responders	Simulation testbench using synthetic traffic
AHBcb-03	F	Prioritization	Simulation testbench using synthetic traffic
AHBcd-04	F	Support for privileged support	Simulation testbench using synthetic traffic

COMMENTS

2.2.5 AHB/APB Interrupt Controller (IFX WI2.2.4-2.2.5)

DESCRIPTION
Develop an AHB Interrupt Controller for RISC-V SoC

MOTIVATION
AHB is the standard bus used, interrupt is a standard feature being provided with RISC-V

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
IFX

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
Follow the CLINT discussions and decide if an RISC-V internal interrupt is the better way to go.

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
AHBcb-01	F/STD	Decide on external interrupts	Simulation of use cases
AHBcd-02	F	Different number of interrupts and priorities	Simulation of interrupt scenarios
AHBcb-03	F	Different ways of Interrupt Service Address handling	Simulation of interrupt scenarios
AHBcd-04	F	Make Interrupt controller robust	Simulation of interrupt scenarios

COMMENTS
Follow-up RISC-V Standardization

2.2.6 AHB/APB Timer (IFX WI2.2.6-2.2.7)

DESCRIPTION
Develop an AHB-Timer for RISC-V SoC

MOTIVATION
AHB is the standard bus used, timer is a standard feature being provided with RISC-V

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
IFX

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
Analyse the internal timer features and decide on internal/external solution.

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
AHBcb-01	F/STD	Decide on external timer	Simulation of use cases
AHBcd-02	F	Different number of timer channels	Simulation of interrupt scenarios
AHBcb-03	F	Different use cases: Watchdog, Clock, PWM, …	Simulation of interrupt scenarios
AHBcd-04	F	Make Timer robust	Simulation of interrupt scenarios

COMMENTS
Follow-up RISC-V Standardization

2.2.7 GRLIB IP library

DESCRIPTION
Provide access to the GRLIB IP library to be used as SoC building blocks for the demonstrators. Extend TBD IP to meet demonstrator needs.

MOTIVATION
GRLIB provides a variety of IP cores under GLP license.

TARGET DEMONSTRATORS
All

IMPLEMENTERS
GSL

LICENSE
GPL

LINK TO OTHER MODULES AND IPs
None

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
GRLIB-IP-01	F	Extend TBD IP.	

COMMENTS
Work can be done to extend GRLIB IP requested by demonstrators.

2.3 Common extension interfaces

2.3.1 Context-Aware CORE Extensions (CA-CORE)

DESCRIPTION
Extend core design with register (CSR) to define current context information and transfer this information to each request issued by the core. Refer to Figure [Section 2.2 – CA-BUS] for a description of the context-aware monitoring and the position of the CA-CORE in its design.

MOTIVATION
Performance Monitoring Counters (PMCs) are essential tools for the verification and monitoring of a system. Safety systems heavily rely on them to implement different safety features. However, they are typically limited, collecting information of every event happening, which puts constraints or limitations on the implementation of the safety mechanisms and/or the system itself. Context-Aware PMCs (CA-PMCs) and the associated infrastructure (CA-CORE, CA-BUS and CA-PMC-IF) will enhance current PMCs and enable more efficient safety verification and monitoring of the system. CA-PMCs will allow the software (typically the OS/Hypervisor) to define the events that need to be collected. In order to implement context-aware PMCs (CA-PMCs, see Section 3.3) the cores require providing the context information with each request: The CA-CORE extension will integrate the current context information (e.g. stored in a dedicated CSR) to each of the request issued by the core, effectively extending the request interface.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
TRT

LICENSE
TBD

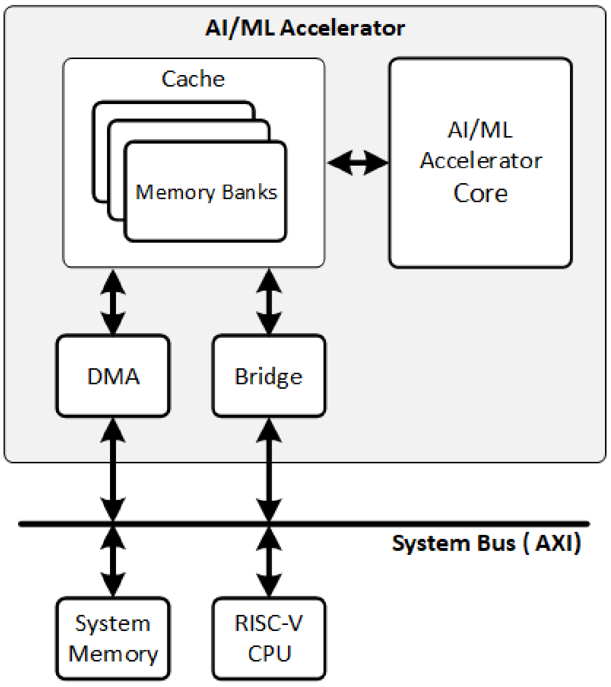
LINK TO OTHER MODULES AND IPs
– CVA6 core developed in T2.1 (see Section 2.1) – BUS-CA developed in T2.2 (see Section 2.2) – CA-PMCs developed in T3.3 (see Section 3.3) – CA-PMCs interface developed in T3.1 (see Section 3.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
CA-CORE-01	STD	The context-aware information should be implemented using the CSRs interface.	Unit tests in SoC FPGA design.

CA-CORE-02	F	The size of the context information should be configurable at design time.	Synthesis of designs with different context information sizes.
CA-CORE-03	PR	The integration of the context information into the ld/st request should not have performance penalties (i.e. same number of cycles with or without the context information).	Simulation measurements.

COMMENTS

2.3.2 Memory bank interface

DESCRIPTION
The cache of the AI/ML accelerator is organized into several memory banks. The memory bank's content can be accessed using a simple memory bank interface. Transactions consist of two phases: address phase and data phase, making this interface easy to adapt to another standard interface. Connecting a RISC-V core directly to the banks using this interface enables tighter integration and more efficient data transfers between the core and the accelerator.  AI/ML Accelerator Cache Memory Banks AI/ML Accelerator Core DMA Bridge System Bus (AXI) System Memory RISC-V CPU

MOTIVATION
Allowing the RISC-V core to directly access the AI/ML accelerator memory banks means that the accelerator does not need to save or load intermediary results to/from a system memory (e.g., DDR or L2 cache). This has the following advantages: • Quick access to data stored in the banks. No need to transfer intermediate data to system memory. • High bandwidth. Up to 256 bit per clock cycle to feed a SIMD module of the core. • Random access to any location in the banks. The memory banks address space is mapped into the RISC-V address space. • Much lower power as no transactions to intermediate memories are needed. • Tighter integration of the RISC-V core with the AI/ML accelerator. The RISC-V core can perform data processing that is not supported by the accelerator.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS

XPERI

LICENSE
Open Source (license TBD)

LINK TO OTHER MODULES AND IPs
– CVA6 or NOEL-V cores (T2.1) and AI/ML accelerator (T3.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
MB-IF-01	F	RISC-V core R/W access to memory bank	HW verification tests
MB-IF-02	STD	AXI Protocol Standard	HW verification tests
MB-IF-03	PR	Memory bank read transfer performance	HW verification tests
MB-IF-04	PR	Memory bank write transfer performance	HW verification tests

COMMENTS
-

2.3.3 Extension interface for NOEL-V

DESCRIPTION
Extend the NOEL-V processor with an extension/co-processor interface to support integration of third-party accelerators.

MOTIVATION
A standardized extension interface is essential for easy integration of tightly coupled accelerator IPs to the processor core. With this interface, the processor core can be extended with additional features (e.g. Vector processing unit VPU, other custom extensions) without modifying the actual processor core.

TARGET DEMONSTRATORS
Space, Automotive

IMPLEMENTERS
GSL

LICENSE
GPL

LINK TO OTHER MODULES AND IPs
– VPU developed in T3.4 (see Section TBD)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
NV-XIF-01	STD	The extension interface should implement an open standard	
NV-XIF-02	F	The interface should support integration with a VPU	Simulation and FPGA prototyping.

COMMENTS
The specification for the extension interface to be used is still TBD.

2.3.4 Extension interface for CVA6

DESCRIPTION
Revisit the currently implemented CV-X-IF extension interface on CVA6 to support integration of third-party accelerators.

MOTIVATION
A standardized extension interface is essential for easy integration of tightly coupled accelerator IPs to the processor core. With this interface, the processor core can be extended with additional features like post-quantum cryptography co-processor or vector co-processor without modifying the existing processor core.

TARGET DEMONSTRATORS
Smart home

IMPLEMENTERS
TDIS

LICENSE
Permissive open source (Solderpad)

LINK TO OTHER MODULES AND IPs
Accelerators (3.4.1, 3.4.2, 3.4.3, 3.4.4, 3.6.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
CVA6-XIF-01	STD	The extension interface should implement an open standard	
CVA6-XIF-02	F	The interface should support integration with vector co-processor.	Simulation and FPGA prototyping.

COMMENTS
The specification for the extension interface to be used is still TBD.

2.3.5 Extension interface between CVA6 and Vector accelerator

DESCRIPTION
Tune the CV-X-IF extension interface on CVA6 to support integration of third-party accelerators (e.g., vector accelerator).

MOTIVATION
A standardized extension interface like the CV-X-IF is essential for easy integration of tightly coupled accelerator IPs to the processor core. The current CV-X-IF specifications likely need to be tuned to take into account the needs of complex accelerators with specific requirements on their load-store unit and/or register file.

TARGET DEMONSTRATORS
IoT

IMPLEMENTERS
ETHZ

LICENSE
Permissive open source (Solderpad)

LINK TO OTHER MODULES AND IPs
2.3.4

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
CVA6-V-XIF-01	F	The interface should support integration with vector co-processor.	Cycle-accurate simulation

COMMENTS

2.4 SW interfaces to general purpose cores

2.4.1 XNG NOEL-V BSP update to new standards

DESCRIPTION
Development and testing of XNG (XtratuM Next Generation) NOEL-V architectural port to support new standards of RISC-V and new IPs developed in the scope of the project.

MOTIVATION
New standards of RISC-V architecture implemented by the NOEL-V core make it interesting to update the XNG RISC-V BSP (Board Support Package) to support all the newest features. Also, it is interesting to integrate inside the hypervisor improvements and new IPs developed inside the scope of this project.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
FEN

LICENSE
Proprietary

LINK TO OTHER MODULES AND IPs
NOEL-V core (T2.1) Other links TBD

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
XNG-01	F	XNG should support Kintex UltraScale FPGA integrating a multicore NOEL-V processor as hardware platform	XNG runs on the target platform and functional tests succeed
XNG-02	F	XNG should integrate the custom interrupt controller developed in the project	Functional testing
XNG-03	F	XNG should support the IOMMU developed in the scope of the project	Functional testing by executing specific tests
XNG-04	F	XNG should provide services (or direct access) to allow the integration of XNG in the RAPITA benchmarks of stress generation on shared resources. Note: this feature could depend on the underlaying hardware	Functional testing
XNG-05	F	XNG should integrate fault injection tests	Functional testing

XNG-06	F	Fault injection tests periodicity should be configurable in the XNG configuration files (XCF)	Functional testing
XNG-07	F	XNG should implement a health monitor to perform an action previously defined in the XCF when a fault is detected	Functional testing

COMMENTS
- Interrupt controller integration will be addressed in Task 3.1 - IOMMU integration will be addressed in Task 2.2. - The integration of the RAPITA benchmarks will be addressed in Task 4.3. - Fault injection tests integration will be addressed in Task 4.3.

2.4.2 CVA-6 multicore experiments and analysis

DESCRIPTION
Investigate CVA-6 multicore setups and safety/security properties.

MOTIVATION
Multicore mixed critical systems have a need for resource partitioning (e.g. cache partitioning). We investigate what is available and whether it can/could be used meaningfully in a CVA-6 multicore setup.

TARGET DEMONSTRATORS
Space, Automotive, Smart Home

IMPLEMENTERS
SYSGO

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
Link to CVA-6 activities TBD

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
CVAM-01	F	It shall be analyzed whether/how to use multicore for mixed critical systems, also in context of demonstrator.	Report in demonstrator WP.

COMMENTS
None

2.4.3 SW Abstraction Layer for RISC-V Processors (IFX WI 2.4.1 2.4.3)

DESCRIPTION
Define a software abstraction layer (SAL) that strives for treating different RISC-V cores the same

MOTIVATION
New RISC-V variants offer specific solutions in domain specific areas but are that different that parts of the SW stack must be re-written. A common abstraction layer may at least reduce the effort.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
IFX

LICENSE
Proprietary

LINK TO OTHER MODULES AND IPs
NOEL-V core, CVA6 Core, and other cores

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
SAL-01	F	List of differences between processors, either ISA support, ISA extension, CSRs or architectural	Review
SAL-02	F	Provide proposal for an abstraction layer for discussion	Review
SAL-03	F	Provide concept for abstraction layer	Reviews
SAL-04	F	Provide implementation of abstraction layer.	Functional testing
SAL-05	F	Measure impact of abstraction layer	Functional testing

COMMENTS
- Processor details may be subject of confidentiality.

2.4.4 NOEL-V software tools

DESCRIPTION
Provide bare-metal and Linux RISC-V tool chains for NOEL-V. Extend support for NOEL-V in TBD software.

MOTIVATION
To support building software for NOEL-V targets

TARGET DEMONSTRATORS
All

IMPLEMENTERS
GSL

LICENSE
GPL

LINK TO OTHER MODULES AND IPs
None

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
NV-SW-01	F	Extend support for NOEL-V in TBD software.	

COMMENTS
Work can be done to extend software support requested by demonstrators.

3 Accelerators and extensions

This section will be devoted to defining functional / non-functional requirements and specifications for the different components related to accelerators and extensions developed in the context of ISOLDE's WP3. These components are divided according to the tasks inside WP3 and include modules for safety and security, accelerator interfaces and infrastructure, modules for infrastructure monitoring, VEC and AI accelerators, cryptographic accelerators and ASIPs and neuromorphic accelerators.

3.1 Safety and Security modules

3.1.1 Safety-related Traffic Injector (SafeTI)

DESCRIPTION
Integration, enhancement, and documentation of a safety-related traffic injector

MOTIVATION
Validating complex SoCs with software-only means or externally (e.g., through JTAG or similar interfaces) poses many challenges to produce relevant traffic scenarios in the internal interconnects. This is mainly due to the inability to generate those traffic patterns by software means, or the fact that they can only be generated with asynchronous events (w.r.t. the computing cores activities), such as Ethernet traffic. This challenges the ability to assess scenarios where specific traffic events occur simultaneously. Having a programmable traffic injector on-chip allows tackling this limitation and achieving appropriate validation for safety-relevant SoCs.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
BSC

LICENSE
MIT (other open-source permissive licenses also possible)

LINK TO OTHER MODULES AND IPs
Not applicable. The work does not rely on previous tasks

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
SafeTI-01	F	Support for AMBA AXI	Traffic patterns can be programmed, and injected traffic observed in the AXI interface (e.g., by means of simulation)

SafeTI-02	F	Flexible traffic patterns supported (read/write, variable burst length, variable inter-request delay)	Traffic injected in isolation matches the patterns programmed (e.g., observed in the AXI interface by means of simulation)
SafeTI-03	PR	Traffic injected by a given core decreases linearly with the size of the SafeTI injected bursts	A core sending sustained requests through the AXI interface with N cycles between consecutive requests observes an inter-request delay of (B-N) cycles, where B is the minimum number of cycles needed to send a data burst of the size programmed in the SafeTI

COMMENTS

3.1.2 Context-Aware PMC Interface (CA-PMC-IF)

DESCRIPTION
Development of addressable interface for the CA-PMCs for their integration into the Safety Island. Refer to Figure [Section 2.2 – CA-BUS] for a description of the context-aware monitoring and the position of the CA-PMC-IF in its design.

MOTIVATION
Performance Monitoring Counters (PMCs) are essential tools for the verification and monitoring of a system. Safety systems heavily rely on them to implement different safety features. However, they are typically limited, collecting information of every event happening, which puts constraints or limitation on the implementation of the safety mechanisms and/or the system itself. Context-Aware PMCs (CA-PMCs) and the associated infrastructure (CA-CORE, CA-BUS and CA-PMC-IF) will enhance current PMCs and enable more efficient safety verification and monitoring of the system. CA-PMCs will allow the software (typically the OS/Hypervisor) to define the events that need to be collected. CA-PMCs deployed in the system should be accessible for (1) the Safety Island for its internal use and/or (2) the host core. An addressable interface (e.g. through a bus) provides a scalable method to provide this access.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
TRT

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
– CA-PMCs developed in T3.3 (see Section 3.3)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
CA-PMC-IF-01	F	The interface should provide capabilities to configure the context filtering capabilities of the CA-PMCs.	Unit tests in SoC FPGA design.
CA-PMC-IF-02	F	The interface should provide means to retrieve and reset CA-PMCs counters values.	Unit tests in SoC FPGA design.
CA-PMC-IF-03	F	The interface should provide means to set up thresholds on CA-PMCs.	Unit tests in SoC FPGA design.

CA-PMC-IF-04	F	The interface should provide the means to access multiple CA-PMCs.	Unit tests in SoC FPGA design.

COMMENTS

3.1.3 Forward-Edge Control Flow Integrity Unit (FCFI)

DESCRIPTION
Development, testing and RISC-V integration of a forward-edge control flow integrity unit (FCFI) which protects the instruction stream.

MOTIVATION
A common physical attack class are fault injection attacks which aim to modify or skip instructions bypassing security measures (e.g., skipping boot image authentication). FCFI probabilistically thwarts such attacks by verifying the integrity of the instruction stream.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
NXP-AT

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
– NOEL-V 32-bit core (T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
FCFI-01	F	FCFI shall offer different security levels.	-
FCFI-02	F	FCFI can be enabled and disabled dynamically to ensure compatibility with unprotected code parts.	-
FCFI-03	F	FCFI shall use a small number of custom instruction encodings.	-
FCFI-04	PR	The code size overhead shall be below 15%.	Benchmarks

COMMENTS

3.1.4 Backward-Edge Control Flow Integrity Unit (BCFI)

DESCRIPTION
Development, testing and RISC-V integration of a backward-edge control flow integrity unit (BCFI) based on a shadow stack isolated by memory encryption / scrambling.

MOTIVATION
A common logical attack class are the exploitation of memory safety violations (e.g., buffer overflows). These attacks allow an attacker to divert the genuine control flow and take over control by overwriting return addresses stored on the stack. BCFI thwarts such attacks by using tweaked memory encryption / scrambling ensuring that controlled modifications of return values by an attacker are infeasible.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
NXP-AT

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
– NOEL-V 32-bit core (T2.1) – Inline Memory Encryption / Scrambling Engine (T3.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
BCFI-01	F	BCFI can be enabled and disabled dynamically per privilege level.	-
BCFI-02	STD	BCFI should be optionally compliant to the *Zicfilp* RISC-V ISA extension.	-
BCFI-03	PR	BCFI shall have less memory overhead than a regular shadow stack.	Benchmarks
BCFI-04	PR	BCFI shall have a comparable performance and code size overhead to a regular shadow stack.	Benchmarks

COMMENTS
Impact on area, power and maximum frequency mainly depend on used inline memory encryption / scrambling engine.

3.1.5 Cryptographically Tagged Memory Unit (CTM)

DESCRIPTION
Development, testing and RISC-V integration of a cryptographically tagged memory unit (CTM) based on memory encryption / scrambling.

MOTIVATION
A common logical attack class are the exploitation of memory safety violations (e.g., buffer overflows). These attacks allow an attacker to leak or corrupt other sensitive objects in memory. CTM thwarts such attacks by tagging pointers and implicitly link the corresponding memory region with the genuine tag using it as tweak for memory encryption / scrambling.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
NXP-AT

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
– NOEL-V 32-bit core (T2.1) – Inline Memory Encryption / Scrambling Engine (T3.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
CTM-01	F	CTM shall support sufficiently large tag sizes.	-
CTM-02	PR	CTM shall have negligible impact on performance and code size.	Benchmarks

COMMENTS
Impact on area, power and maximum frequency mainly depend on used inline memory encryption / scrambling engine.

3.1.6 Enclave Memory Isolation Unit (EMI)

DESCRIPTION
Development, testing and RISC-V integration of an enclave memory isolation unit (EMI) based on memory encryption / scrambling.

MOTIVATION
Secure enclaves offer a trusted execution environment for sensitive assets and services isolated from the rest of the system. For tightly coupled memories, the needed isolation is often provided by logical isolation using memory protection units. However, logical isolation may be ineffective against advanced fault or bus probing attacks. Further, most existing schemes target virtualization, which is not available on small embedded cores. EMI thwarts such attacks by additionally encrypting the enclave memory regions using enclave-specific encryption / scrambling tweaks.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
NXP-AT

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
– NOEL-V 32-bit core (T2.1) – Inline Memory Encryption / Scrambling Engine (T3.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
EMI-01	F	EMI supports cores without MMU.	-
EMI-02	F	EMI provides cryptographic isolation.	-
EMI-03	F	Employing EMI shall need no compiler adaptions.	-

COMMENTS
Impact on area, power and maximum frequency mainly depend on used inline memory encryption / scrambling engine.

3.1.7 Lightweight Tweakable Inline Memory Encryption Engine (IEE)

DESCRIPTION
Development, testing and RISC-V integration of a tweakable inline memory encryption / scrambling engine (IEE).

MOTIVATION
Most of the security modules contributed by NXP-AT (BCFI, CTM, EMI) require a lightweight tweakable memory encryption / scrambling engine which is provided by this module.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
NXP-AT

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
– NOEL-V 32-bit core (T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
IEE-01	F	IEE shall support a 128-bit tweak.	-
IEE-02	F	IEE shall support a configurable security level.	-
IEE-03	PR	IEE shall not considerably impact the maximum frequency.	RTL Synthesis Reports

COMMENTS

3.1.8 Memory subsystem support for Bytecode VMs

DESCRIPTION
The goal is to develop a concept for memory subsystem support to support bytecode virtual machines, in particular WebAssembly. For this the VexRiscv processor core is extended with extensions to the memory subsystem to accelerate interpreter execution.

MOTIVATION
Memory interaction is a significant part of an interpreter’s execution overhead.

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
HM

LICENSE
MIT

LINK TO OTHER MODULES AND IPs
– VexRiscv 32-bit core (T2.2)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
VEX-TAG-01	F	Concept for memory tagging based on Xjpm32	Simulation and FPGA-Prototype

COMMENTS

3.1.9 Safety Island – Interface Definition

DESCRIPTION
The goal is to develop an interface between the Safety Island and the main processing unit. To continuously ensure security and proper function, test scenarios are generated, and the implementation is tested against them. Furthermore, the interface performance is evaluated, and results from this evaluation are fed back into the development of the interface implementation.

MOTIVATION
The Safety Island is relevant for the overall system's safety.

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
UZL

LICENSE
Open Source (TBD)

LINK TO OTHER MODULES AND IPs
– Requirements (T1.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
SI-ID-01	F	Concept for Interface between Safety Island and CPU	Simulation and FPGA-Prototype

COMMENTS
SYSGO will contribute to safety island interface by security and safety analysis/review from a system software developer point of view.

3.1.10 Root-of-Trust unit design and interface with RISC-V Host processor (TitanCFI)

DESCRIPTION
The goal is to develop a HW monitor integrated in the RISC-V processor that will interact with an on-chip RoT module implemented using OpenTitan. The monitor will extract real time execution traces (e.g. instructions, execution context and performance counters) that are processed by OT to enforce Control-Flow-Integrity in the OT firmware. The HW interface will enable real-time or quasi-real-time detection of control flow diversion interface. The HW monitor will be tested using benchmarks related to specific use cases of the project, namely aerospace and avionics applications.

MOTIVATION
Fast and reactive Control-flow-Integrity enforcement.

TARGET DEMONSTRATORS
Aerospace

IMPLEMENTERS
UNIBO

LICENSE
SolderPad Hardware License v0.51

LINK TO OTHER MODULES AND IPs
CVA6 core (T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
TitanCFI-01	F	AXI port and Interrupt line for mailbox interface with OT	
TitanCFI-02	PR	Operating frequency ratio CVA6/OT=1/1	
TitanCFI-03	PR	CFI overhead	Simulation and FPGA-Prototype

COMMENTS
SYSGO gives feedback on design.

3.1.11 Safety Control (IFX WI3.1.1-3.1.4)

DESCRIPTION
The goal is to develop a HW safety control unit closely or loosely coupled to a RISC-V core. The safety control unit observes detected faults and raises an exception if a threshold is reached.

MOTIVATION
Safety Analysis should be made independent. Safety control should underly a special safety concept.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
IFX

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
To IPs and Cores

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
SaCU-01	F	Provide a Safety Concept	Review
SaCU-02	PR	Provide a hardened safety architecture (HW/SW)	Review
SaCU-03	PR	Have an open interface between IPs and Cores as well as Safety Control UNit	Simulation and FPGA-Prototype

COMMENTS

3.2 Accelerator infrastructure, memories, arithmetic units, interfaces and virtualization

3.2.1 Floating-Point Unit (FPU) for RISC-V

DESCRIPTION
FPU development is a multi-stage process. The goal is to start the implementation with a simple FPU, which has a reduced function and instruction set scope, implemented. After successful testing and integration, the unit is planned to be successively extended with further functions as defined by the requirements analysis.

MOTIVATION
Support of floating-point operation, which is required for a wide range of applications.

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
UZL

LICENSE
Open Source (TBD)

LINK TO OTHER MODULES AND IPs
Requirements (T1.3)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
FPU-01	F	Support of Floating-Point Operations	Simulation
FPU-02	F	Support of Floating-Point Standard (SP, DP, QP)	Simulation
FPU-03	F	FPU Integration into the RISC-V SoC	Simulation and FPGA-Prototype

COMMENTS

3.2.2 FPU for mixed-precision computing (FPMIX)

DESCRIPTION
Customizable floating-point arithmetic unit for mixed-precision computing

MOTIVATION
Supporting multiple precisions for computation-heavy kernels is crucial to the energy efficiency of a computing platform, enabling the search for optimal accuracy-energy-performance tradeoffs that better suit the requirements and constraints of the system. The customizable FPU for mixed-precision computing features a precision (i.e., number of mantissa bits) that can be configured at design time independently for each type of operations, while maintaining a common dynamic range (i.e., number of exponent bits) fixed to the one of the IEEE-754 float32 standard format to avoid the need for any compiler-side changes.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
POLIMI

LICENSE
Open Source (bfloat16 version), proprietary licensing (other versions)

LINK TO OTHER MODULES AND IPs

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
FPMIX-01	F	Correctness of single-precision floating-point operations	Correctness checked through simulation
FPMIX-02	F	Design-time configurability of precision of FP format for the implemented FP operations	Synthesis and implementation targeting FPGA
FPMIX-03	F	Support for RISC-V F instructions	Simulation
FPMIX-04	PR	FPU instances with reduced precision have better or equal performance and area compared to instances with more mantissa bits in the FP operations	Synthesis and implementation on FPGA target for area metrics, simulation for performance evaluation

COMMENTS
None.

3.3 Monitoring infrastructure

3.3.1 Safety-related Statistics Unit (SafeSU)

DESCRIPTION
Integration, enhancement, and documentation of a safety-related statistics unit

MOTIVATION
The safety concept often requires (i) the implementation of safety measures related to monitoring that some tasks using some cores; and (ii) accelerators or peripherals do not abuse some specific shared hardware resources, which could lead to starvation or deadline overruns for some safety-critical tasks. The lack of observability of those resources, or high latency to detect such utilization abuse could jeopardize the safety of the system. Therefore, deploying appropriate statistical units capable of observing such behavior, limiting the abuse by means of quotas, and/or reporting immediately such an abuse (e.g., with interrupts) becomes mandatory.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
BSC

LICENSE
MIT (other open-source permissive licenses also possible)

LINK TO OTHER MODULES AND IPs
Not applicable. The work does not rely on previous tasks

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
SafeSU-01	F	Support for AMBA AHB and AXI	Statistics for different types of traffic traversing the corres-ponding AHB or AXI interface are accurately captured (e.g., by means of simulation). Such traffic can be injected with the SafeTI in T3.1
SafeSU-02	F	Traffic quotas can be set for specific devices	Quotas can be programmed in the SafeSU, properly monitored, and upon exhaustion, the corresponding interrupt raised (e.g., observing the SafeSU and interconnect behavior during simulation)

SafeSU-03	PR	Quota violation is reported within 1,000 clock cycles to the system software	No more than 1,000 cycles elapse since the time the quota is exhausted until the software interrupt routine starts its execution

COMMENTS
None.

3.3.2 Time Contract Monitoring Co-Processor (TCPP)

DESCRIPTION
Development, testing and integration of a modular/composable time contract monitoring co-processor in Safety Island.

MOTIVATION
Time contract monitoring co-processor can monitor the execution of timing properties received from safety island infrastructure and validate them based on the given specifications in a contract-based language. This co-processor can be used for safety and security.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
OFFIS

LICENSE
Apache License V2.0, TBC

LINK TO OTHER MODULES AND IPs
T1.2, T2.2, T3.1

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
TCCP-01	F	The TCCP should be able to monitor different properties simultaneously	Verified in RTL testbench
TCCP-02	F	The TCCP should be able to store and update timing specification	Verified in system integration and RTL testbench
TCCP-03	F	The TCCP should be able to process data captured by safety island infrastructure	Verified in system integration
TCCP-04	F	The results of TCCP should be available for action in HW and SW (as interrupt and MMIO)	Verified in system integration and RTL testbench
TCCP-05	PR	Monitoring per sec. > (No. of events per message * input traffic rate)	Verified in system integration and simulation
TCCP-06	PR	Support monitoring at least 100 contract-expressions at full monitoring rate	Verified in system integration and simulation

COMMENTS

3.3.3 Context-Aware PMC (CA-PMC)

DESCRIPTION
Implementation of PMCs with context filtering capabilities (CA-PMCs, for context-aware PMCs). Refer to Figure [Section 2.2 – CA-BUS] for a description of the context-aware monitoring and the position of the CA-PMC in its design.

MOTIVATION
Performance Monitoring Counters (PMCs) are essential tools for the verification and monitoring of a system. Safety systems heavily rely on them to implement different safety features. However, they are typically limited, collecting information of every event happening, which puts constraints or limitation on the implementation of the safety mechanisms and/or the system itself. Context-Aware PMCs (CA-PMCs) and the associated infrastructure (CA-CORE, CA-BUS and CA-PMC-IF) will enhance current PMCs and enable more efficient safety verification and monitoring of the system. CA-PMCs will allow the software (typically the OS/Hypervisor) to define the events that need to be collected. Basic PMCs are not able to filter the events, i.e. all the events that occur in the target IP are counted. More advanced PMCs are able to filter events based on address range, when applicable. The PMCs developed in the ISOLDE monitoring infrastructure (CA-PMCs) will be able to filter the requests based on a context (e.g. VM or process executing in the host core) on which the event was generated.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
TRT

LICENSE
TBD

LINK TO OTHER MODULES AND IPs
– CA-PMC interface developed in T3.1 (see Section 3.1) – bus interface with context information developed in T2.2 (see Section 2.2) – core with context registers developed in T2.3 (see Section 2.3)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
CA-PMC-01	F	CA-PMC implementation for cache PMCs.	Unit tests in SoC FPGA design.
CA-PMC-02	F	CA-PMC implementation for bus interfaces PMCs.	Unit tests in SoC FPGA design.
CA-PMC-03	P	Bounded event to counter processing time. Target: within 10 clock cycles.	Simulation measurement.

COMMENTS

3.3.4 Run-time power monitoring instrumentation (RTPM)

DESCRIPTION
Automatic generation of the run-time power monitoring infrastructure

MOTIVATION
Run-time optimization techniques enable improving the energy efficiency of modern computing platforms, and their effectiveness is strongly related to the quality of the measurements or estimates of power consumption provided by a run-time power monitoring system, which can leverage either direct measurement or indirect estimation. The latter methods leverage the correlation between the power consumption and a set of the platform's run-time statistics, ranging from the architectural performance monitoring counters down to the switching activity of the microarchitecture, to deliver periodic power estimates. The tool for the automatic generation of on-line power monitoring infrastructure employs a linear regression model taking the switching activity of selected signals in the monitored component as its input and instantiates hardware counters to monitor such signals while considering accuracy, area overhead, and side-channel information leakage metrics as constraints in the model identification phase.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
POLIMI

LICENSE
Proprietary licensing

LINK TO OTHER MODULES AND IPs
Applicable to cores and accelerators in the target system (e.g., see 3.5.3)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
RTPM-01	F	Instrumented monitoring infrastructure satisfies defined area overhead constraint	Synthesis and implementation on FPGA target
RTPM-02	F	Instrumented monitoring infrastructure satisfies accuracy constraint.	Simulation and measurement on board prototype
RTPM-03	F	Instrumented monitoring infrastructure does not enable side-channel attacks.	Measurement on board prototype

RTPM-04	PR	Instantiated power monitor delivers power estimation at a temporal resolution not higher than 1ms	System integration and simulation

COMMENTS
None.

3.4 SIMD/Vector, AI accelerator and tensor processor unit design

3.4.1 Tensor Processing Unit

DESCRIPTION
Development, testing and RISC-V integration of a Tensor Processing Unit

MOTIVATION
Tensor Processing Units are special-purpose units that can be integrated with general-purpose processors in a computing fabric to accelerate matrix and tensor operations, common in AI and ML scenarios and all applications exploiting linear algebra (e.g., matrix multiplications, additions, N-dimensional convolutions)

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
UNIBO

LICENSE
SolderPad Hardware License v0.51

LINK TO OTHER MODULES AND IPs
- CVA6 core (T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
TPU-01	F	Support for IEEE754 FP16, Bfloat16, MiniFloat (8-bit 1-4-3, 8-bit 1-5-2)	-
TPU-02	F	Support for General Matrix Multiplication (GEMM) operations and General Matrix-Matrix Operations (GEMM-Ops)	-
TPU-03	PR	Computational efficiency per unit of area	>= 50 GFLOPS/mm^2 (FP16, 22nm, 100% utilization)
TPU-03	PR	Computational efficiency per unit of power	>= 500 GFLOPS/W (FP16, 22nm, 100% utilization)

COMMENTS

3.4.2 Parallel Compute Engine

DESCRIPTION
Development, testing and integration of a Parallel compute engine into a RISC-V system based on the CVA6 core.

MOTIVATION
The recent shift of the programming paradigm towards data-intensive algorithms has exposed the limitations of traditional Von Neumann architectures in terms of both performance and energy efficiency. This is especially true for edge applications, where most of the computation is carried out on board, possibly in near-real time. This calls for a new generation of heterogeneous computing architectures that offload computationally-intensive parts of the application to parallel and/or distributed processing engines, that have been proved to increase energy efficiency by reducing the pressure on the memory hierarchy and interconnections.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
PoliTo

LICENSE
Solderpad (open source)

LINK TO OTHER MODULES AND IPs
– CVA6 core (T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
COMPUTE-ENGINE-00	PR	The compute engine SHOULD outperform a CPU-only system in highly-parallelizable workloads.	The compute engine must show a maximum theoretical throughput, defined as arithmetic-logic operations per cycle, that is at least 10 times the one obtained by the system's CPU in the same application (T.B.D.).
COMPUTE-ENGINE-01	F	The compute engine SHOULD efficiently support machine-learning algorithms (e.g., convolutional neural networks).	The compute engine must support multiply-and-accumulate operations in its functional units.

COMPUTE-ENGINE-02	PH	The compute engine SHOULD not bottleneck the operating performance of the host system.	The maximum propagation delay of the compute engine must not be higher than the critical path of the host system without it (i.e., the operating frequency of the system must not be reduced by connecting the compute engine).
COMPUTE-ENGINE-03	STD	The compute engine MUST be connected to the host system through a standard interface to enforce portability to different platforms and future implementations.	The compute engine must be connected to the host system through the T.B.D. standard interface.

COMMENTS

3.4.3 AI/ML Accelerator

DESCRIPTION
Development, testing, verification and RISC-V integration of a power efficient Neural Network Accelerator

MOTIVATION
AI accelerators are special-purpose units that can be integrated in a system with general-purpose processors. They are used to accelerate specific operations used in deep Convolutional Neural Networks, characteristic for typical AI and ML scenarios. AI Accelerators provide much higher computing power, typically at least one level of magnitude higher than a general-purpose processor with SIMD extension. Typical performance for embedded solutions can range from 512 to 4096 or more operations per clock cycle, at clock frequencies of 500MHz to 1GHz.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
XPERI

LICENSE
The extension design of the existing IP will be licensed under XPERI Proprietary License (closed).

LINK TO OTHER MODULES AND IPs
CVA6 core (T2.1) and/or NOEL-V core (also T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
AIACC-01	F	Support for IEEE754 FP16	HW verification tests
AIACC-02	F	Support for NN operations: Convolution, Pooling, Activation Functions	HW verification tests
AIACC-03	F	Standard interfaces (AXI) for System Integration	HW verification tests
AIACC-04	F	Integration with CPU (RISC-V)	HW verification tests
AIACC-05	PR	Computational efficiency	Comparison will be made against a plain RISC-V implementation and speedup determined

COMMENTS

3.4.4 Vector Processing Unit (with multi-precision capabilities)

DESCRIPTION
Development and implementation of a multi-precision-ready RISC-V vector processor.

MOTIVATION
RISC-V V is the largest RISC-V ISA extension and can accelerate a plethora of different data-parallel applications, making it suited for multiple domains and needs, including IoT. The vector processor architecture has an intrinsic energy efficiency advantage, i.e., every instruction is amortized on multiple elements. Lower-precision processing, on the other hand, is key to both speed up the target applications and reduce their energy cost.

TARGET DEMONSTRATORS
IoT

IMPLEMENTERS
ETHZ

LICENSE
SolderPad Hardware License v0.51

LINK TO OTHER MODULES AND IPs
- CVA6 core (T2.1) - Scalar-Accelerator Interface (T2.3)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
VPU-01	STD/F	Support for integer operations on 64, 32, 16, and 8 bits	Cycle-accurate simulation
VPU-02	F	Support for IEEE754 FP64, FP32, FP16	Cycle-accurate simulation
VPU-03	F	Integration with the scalar core	Cycle-accurate simulation
VPU-04	PR	Peak efficiency of 33 pJ/DP-FLOP, 19 pJ/SP-FLOP, 12 pJ/HP-FLOP in 22-nm technology, at 0.8V, TT, 25C (matrix multiplication)	Post-layout simulations (switching activity) + power analysis in 22-nm technology at 0.8V, TT, 25C

COMMENTS

3.4.5 CNN Accelerator for an Event-based Sparse Neural Networks

DESCRIPTION
Development, testing, verification and RISC-V integration of an efficient CNN accelerator for event-based sparse neural networks computation.

MOTIVATION
Unlike traditional frame-based approaches, event-based cameras only record changes in brightness, producing a stream of events when pixels detect variations in light intensity. This results in several key benefits. Firstly, they operate at extremely high temporal resolution, enabling them to capture rapid motion and dynamic scenes with minimal motion blur. Secondly, they are highly efficient in terms of data transmission and storage, as they only record relevant changes, reducing the amount of redundant information. Moreover, these cameras consume significantly less power, making them suitable for resource-constrained environments and applications like robotics, and automotive. In this scenario, implementing an accelerator capable of exploiting the sparseness of the data, leads to great savings in terms of power, and computation time.

TARGET DEMONSTRATORS
Automotive, Possibly Space

IMPLEMENTERS
SAL

LICENSE
The IP will be licensed under a Proprietary License (closed).

LINK TO OTHER MODULES AND IPs
CVA6 core (T2.1) and/or NOEL-V core (also T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
EB-CNN-01	F	Storage and upload/download of CNN bias' and weights	HW verification tests
EB-CNN-02	F	CNN image recognition on hardware given a previously defined network (no training in the RTL design)	HW verification tests
EB-CNN-03	F	Event-based processing of image data	HW verification tests
EB-CNN-04	F	Sparse matrices computations using coordinate format (COO)	HW verification tests

COMMENTS

3.5 Cryptographic and security accelerators

3.5.1 HLS-based Post-Quantum Cryptographic Accelerator (HLS-PQC)

DESCRIPTION
Integration, enhancement, and documentation of a HLS-based Post-Quantum Cryptographic Accelerator

MOTIVATION
Communication security is one of the important parts of a system. In recent years, researchers have discovered how to break current public-key cryptographic algorithms with quantum computing. As a result, organizations, such as NIST, are standardizing several quantum-resistant algorithms to circumvent this problem. Therefore, we have decided to integrate an HLS-based accelerator, implementing the Key Encapsulation Mechanism ML-KEM (FIPS-203), and the Digital Signature Scheme ML-DSA (FIPS-204). Both are based on CRYSTALS-Kyber and CRYSTALS-Dilithium schemes, respectively.

TARGET DEMONSTRATORS
IoT

IMPLEMENTERS
BSC

LICENSE
SolderPad Hardware License v0.51

LINK TO OTHER MODULES AND IPs
Not applicable. The work does not rely on previous tasks

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
HLS-PQC-01	F	Integration of ML-KEM Accelerator with AXI interface	Execution on bare-metal, comparing original ML-KEM software execution output with the accelerator output.
HLS-PQC-02	F	Integration of ML-DSA Accelerator with AXI interface	Execution on bare-metal, comparing original ML-DSA software execution output with the accelerator output.
HLS-PQC-03	PR	The accelerator shall provide at least 50x of improvement against the unaccelerated software version	FPGA execution on bare-metal, using core PMU to check the performance results.

COMMENTS
None.

3.5.2 A Multipurpose Accelerator for PQC (PQC-MA)

DESCRIPTION
Creation, and/or incorporation of a number of cryptographic hardware modules implementing PQC primitives into a larger, general purpose PQC accelerator. This will target common primitives for PQC algorithms as well as specific PQC standards/cryptosystems. A possible extension of the work would be the integration with other existing accelerators (e.g. a vector accelerator).

MOTIVATION
Each PQ cryptosystem uses a set of primitive functions to encrypt data, some of these functions are common across different algorithms. Instead of just targeting one or two algorithms, it makes more sense to try to provide acceleration which targets these common primitives with further design of particularly resource intensive portions of specific algorithms also being targeted. The stretch goal of other accelerator integration allows for optimal use of design space with more general-purpose use cases for the designs.

TARGET DEMONSTRATORS
Automotive, possibly Space

IMPLEMENTERS
SAL

LICENSE
Multi-Licensing - AGPL v3, Proprietary Licensing

LINK TO OTHER MODULES AND IPs
CVA6 core or NOEL-V core (T2.1), possibly T3.6 (3.6.3)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
PQC-MA-01	F	Capable of interfacing with CV-X-IF or similar Domain Specific Extension interface	Testing of bare-metal implementation (FPGA/ASIC) against the published standard
PQC-MA-02	F	Accelerator interface with AXI-4 Bus	Testing of bare-metal implementation (FPGA/ASIC) against the published standard
PQC-MA-03	PR	Performance of each implemented PQC algorithm must exceed software-based implementation by a factor of 25	Timing tests of RTL design on bare metal (FPGA or ASIC) with a comparison of an appropriate x86, ARM, or RISC-V software-based implementation (with considerations

			of the difference in clock speeds, processing capabilities, etc.)
PQC-MA-04	F	Implementation of an appropriately chosen code-based PQC algorithm	Testing of bare-metal implementation (FPGA/ASIC) against the published standard and Known Answer Tests of the chosen code-based PQC algorithm
PQC-MA-05	F	Implementation of an appropriately chosen lattice-based PQC algorithm	Testing of bare-metal implementation (FPGA/ASIC) against the published standard and Known Answer Tests of the chosen lattice-based PQC algorithm
PQC-MA-06	F	Implementation of appropriately chosen PQC algorithm primitives (e.g. Number Theoretic Transforms)	Testing of bare-metal implementation (FPGA/ASIC) against an appropriately chosen software implementation of the NTT
PQC-MA-07	STD	Compliance (unless impractical) with NIST and/or ETSI PQC algorithm standards	Comparison against published known answer tests

COMMENTS
None.

3.5.3 Accelerator for post-quantum key encapsulation mechanism BIKE (ACC-BIKE)

DESCRIPTION
Integration and documentation of a hardware accelerator for post-quantum key encapsulation mechanism BIKE

MOTIVATION
Post-quantum cryptography (PQC) aims to design cryptoschemes that can be executed on traditional computers and that are secure against both traditional and quantum attacks. BIKE is a post-quantum key encapsulation mechanism (KEM) that is a candidate for standardization in USA's NIST. This accelerator aims to provide hardware support for the key generation, encapsulation, and decapsulation primitives of the BIKE KEM and to be integrated in platforms making use of an AXI interface.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
POLIMI

LICENSE
Proprietary licensing

LINK TO OTHER MODULES AND IPs
CVA6 core or NOEL-V core (T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
ACC-BIKE-01	F	Integration of the BIKE accelerator with AXI interface	Comparison between the output of the software execution of BIKE’s reference sources with the output of the hardware accelerator.
ACC-BIKE-02	F	Compliance with BIKE specification	Comparison between the hardware accelerator’s outputs and the known hardware tests provided by BIKE specification.

COMMENTS
None.

3.6 Signal processing, neuromorphic and application-specific instruction set processors

3.6.1 Motor Control accelerator

DESCRIPTION
The profiling of the algorithms used in the automotive industry with the focus on the motor control. The tailored RISC-V ISA extensions are going to be presented to accelerate some of the most computationally intensive set of algorithms. The identified ISA extensions will be implemented in the Codasip RISC-V core and Codasip Studio, and the performance increase will be demonstrated.

MOTIVATION
The automotive industry relies on the standard set of algorithms and transformations. A custom compute core that is tailored for a given set of algorithms will provide higher performance for the specific application without significant increase of the cost or energy consumption.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
CODA, BUT, NXP-CZ

LICENSE
The designed IP will be licensed under Codasip proprietary commercial license.

LINK TO OTHER MODULES AND IPs
Not applicable. The work does not rely on previous tasks.

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
MC-ACCELERATOR-UNIT-01	STD	The designed IP shall be compliant with relevant parts of the RISC-V specification.	RISC-V compliant tests will run with the provided accelerator unit
MC-ACCELERATOR-UNIT-02	PR	The accelerated IP shall provide at least 10 percent improvement against the unaccelerated version of IP for the targeted automotive application/ algorithm.	Test suite will be implemented to measure performance of the selected algorithms.

MC-ACCELERATOR-UNIT-03	STD	The accelerated IP shall be compliant with the FPGA demonstrator platform.	The results will be run on the FPGA demonstrator platform.
MC-ACCELERATOR-UNIT-04	F	At least three algorithms commonly used in automotive will be profiled and accelerated.	The profiled algorithms are going to be listed in the documentation.

COMMENTS

3.6.2 SCA

DESCRIPTION
The SCA enables the mobile device to synchronize to the base station by correlating the received signal with the known synchronization sequences.

MOTIVATION
Initial synchronization is very compute heavy as up to 210 complex mega correlations have to be completed per second which translates to 430 GOp/s (16b arithmetic operations). The raw throughput and latency requirements for synchronization require a dedicated hardware accelerator that can achieve high performance and energy efficiency.

TARGET DEMONSTRATORS
IoT

IMPLEMENTERS
ACP

LICENSE
TBD

LINK TO OTHER MODULES AND IPs

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
SCA-UNIT-01	PR	Throughput requirement: > 210 Mcorr/s (mega correlations per second)	Measurement in RTL simulation
SCA-UNIT-02	F	The accelerator shall be controlled over an APB interface	RTL testbench that verifies the full functionality
SCA-UNIT-03	F	The accelerator shall have an option to generate an interrupt signal	Verified in system integration and unit testbench

COMMENTS

3.6.3 LDPC

DESCRIPTION
Hardware accelerator for forward error correction of the 5G NR physical downlink shared channel (PDSCH).

MOTIVATION
A processor or vector unit would require a performance in the range of 75 GOp/s of arithmetic operations to complete the decoding tasks of a LDPC decoder for NR PDSCH in the required time. By also considering load, stores, and address computations the required performance will likely be above 200GOp/s. The latency, throughput, and energy efficiency requirements in NR demand a dedicated unit in hardware to perform channel decoding tasks.

TARGET DEMONSTRATORS
IoT

IMPLEMENTERS
ACP

LICENSE
TBD

LINK TO OTHER MODULES AND IPs

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
LDPC-UNIT-01	PR	Latency requirements: <250us	Measurement on FPGA or in RTL simulation
LDPC-UNIT-02	PR	Throughput requirements >= 10mbps	Measurement on FPGA or in RTL simulation
LDPC-UNIT-03	F	The accelerator shall be controlled over a handshake interface	RTL testbench that verifies the full functionality

COMMENTS
In new radio (NR), the LDPC decoder is constrained by the minimal time available for processing a physical downlink shared channel (PDSCH). The budget for the LDPC decoder is 250us of 350us in case of a 5kbit block.

3.6.4 Neuromorphic HW Accelerator

DESCRIPTION
Development, testing and integration of a Neuromorphic HW accelerator into a RISC-V system.

MOTIVATION
Spiking Neural Networks (SNNs), also known as third-generation neural networks, are gaining popularity due to their low power consumption and efficient processing of sparse data. These characteristics make them highly suitable for power-constrained edge and IoT applications. While several open-source FPGA and ASIC neuromorphic processors have been developed in this field, they often require additional computing elements to handle data and communications. This limitation hinders the widespread adoption of such solutions in practical use cases. Consequently, there is a growing need for integrated solutions that combine RISC-V–based processors with neuromorphic hardware accelerators, which can seamlessly function as co-processor units. Integrating these technologies will facilitate the effortless adoption of emerging brain-inspired technologies in IoT and industrial applications relevant to this field.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
PoliTo

LICENSE
Solderpad (open source)

LINK TO OTHER MODULES AND IPs
– CVA6 core (T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
NEUROMORPH-ACC-00	F	The Neuromorphic HW accelerator SHOULD efficiently support the simulation of Spiking Neural Networks.	The Neuromorphic HW accelerator must support spiking neuron simulation and manage input data from external sources (compliant with the Address Event Representation protocol).

NEUROMORPH-ACC-01	PH	The configuration of the Neuromorphic HW accelerator SHOULD not bottleneck the operating performance of the host system.	The maximum propagation delay for configuring the Neuromorphic HW accelerator must not be higher than the critical path of the host system without it (i.e., the operating frequency of the system must not be reduced by connecting the HW accelerator).
NEUROMORPH-ACC-02	F	The accelerated IP MUST be compliant with the FPGA demonstrator platform.	The results will be run on the FPGA demonstrator platform.

COMMENTS

4 Operating systems and applications SW

This section will focus on the definition of the requirements and specifications related to the software components that will be developed in the scope of the ISOLDE project. More precisely, these components to be developed include hypervisors, operating systems, drivers, application libraries, compilers, etc. Also, validation tools such as benchmarks, simulators or other software needed for testing purposes will be part of the software components as well.

A set of functional and non-functional requirements for the software components will be defined to target the applications and demonstrators, covering also safety and security specification.

4.1 System SW design, implementation and testing

4.1.1 Risc-V-NN

DESCRIPTION
Development, testing and benchmarking of support libraries for the acceleration of AI workloads on RISC-V cores and accelerators.

MOTIVATION
The demand for efficient AI execution has given rise to a proliferation of custom instruction set architectures (ISA) and hardware accelerators tailored to address specific, intensive AI workloads. However, effectively harnessing the full potential of these accelerators demands a comprehensive support library. This library acts as a bridge, facilitating seamless communication and synchronization between accelerators and cores, and exploiting their full computation capability. In the current landscape, this library is a critical enabler for the optimization of AI workloads.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
PoliTo

LICENSE
Apache License 2.0

LINK TO OTHER MODULES AND IPs
– CVA6 core (T2.1) – Parallel processing Unit (T3.4, PoliTo) – Neuromorphic HW accelerator (T3.6, PoliTo)

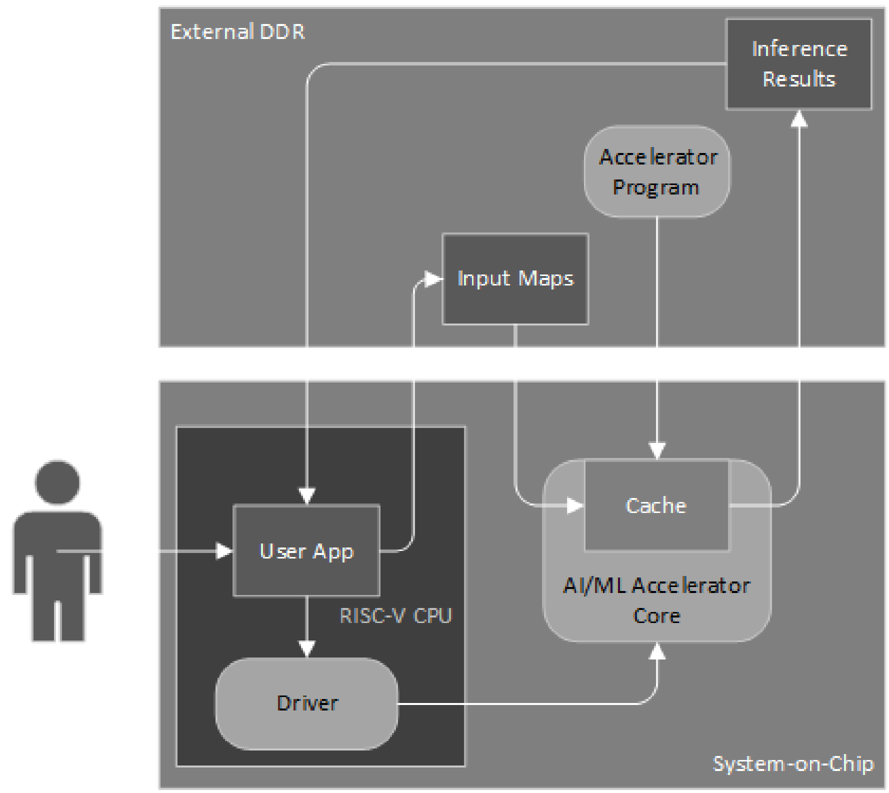
REQUIREMENTS

ID	TYPE	DESCRIPTION	VERIFICATION
RISC-V-NN-00	PR	The library SHOULD enable accelerators to exploit their full utilization in ideal cases.	RISC-V NN should reach maximum accelerator performance when benchmarked on ideal cases, i.e., AI workload that fully utilizes the target hardware.
RISC-V-NN-01	F	The compute engine MUST run on the CVA-6 core, supporting accelerators employed in the Space demonstrator.	The library should be benchmarked and verified on CVA-6 multi-core architectures and on the parallel processing unit from T3.4.

COMMENTS

4.1.2 AI/ML accelerator driver

DESCRIPTION
Driver and system software to support the AI/ML efficient neural network accelerator.

MOTIVATION
AI accelerators are special-purpose units that can be integrated with general-purpose processors using the same bus for memory access. In the implemented scenario the processor will act as a host for the accelerator and will decide when to give the control to the accelerator. A specific piece of software known as a driver will serve as an API with communication with the AI accelerator. This will set up the program to be executed by the accelerator, give control to the accelerator to execute the program and relinquish control once the accelerator program has completed.  External DDR Inference Results Accelerator Program Input Maps Cache User App RISC-V CPU AI/ML Accelerator Core Driver System-on-Chip The figure above illustrates the system level integration at SoC (System on Chip) level of AI /ML Accelerator with a Host RISC-V processor together with external DDR memory as well the programing model. The User Application running on RISC-V CPU communicates with AI/ML Accelerator through a Driver with a dedicated API that includes services such as initialization, configuration, monitoring and so on. Once started, the accelerator reads the execution instructions (prepared by the Compiler) and processes the Input Data. This processing is done without any assistance from the Host CPU. When processing is complete, the Accelerator informs the Host CPU that Inference results are available for the User Application.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
XPERI

LICENSE
Open Source (License TBD) / Proprietary License for additional support

LINK TO OTHER MODULES AND IPs
AI/ML accelerator (T3.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
AIDRV-01	F	Support for setting up accelerator’s program	Capability verified in the demonstrator.
AIDRV-02	F	Support for starting the accelerator execution of the previously configured program	Capability verified in the demonstrator.
AIDRV-03	F	Support for intercepting the termination of accelerator program execution	Capability verified in the demonstrator.

COMMENTS

4.1.3 Optimization of WebAssembly Interpreter

DESCRIPTION
Extend an established WebAssembly interpreter to support novel features introduced in previous tasks. This includes the support for off-the-shelf standard features that are added (like pointer masking) and the configurable support for new methods.

MOTIVATION
To make use of the novel features the WebAssembly interpreter needs to be extended.

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
HM

LICENSE
Apache 2.0

LINK TO OTHER MODULES AND IPs
VexRiscv Extensions for Webassemly (T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
WASM-01	F	Support for new hardware features	Capability verified in the demonstrator.

COMMENTS

4.1.4 Driver for Floating-Point Unit and Saftey Island

DESCRIPTION
In addition to the actual hardware or instruction set definition, the RISC-V ecosystem includes a large software stack consisting of compiler, operating system, drivers, etc. For seamless integration into this ecosystem, a driver for the Floating-Point Unit is created. The security properties of the Security Island are to be made available to the software running on the RISC-V by implementing a software stack.

MOTIVATION
Support of floating-point operations and the Safety Island on software level.

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
UZL

LICENSE
Open Source (TBD)

LINK TO OTHER MODULES AND IPs
Requirements (T1.3, T1.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
D-FP-SI-01	F	FPU support on software level	Simulation
D-FP-SI-02	F	Safety Island support on software level	Simulation

COMMENTS

4.1.5 Use of device tree abstraction for Linux on RISC-V

DESCRIPTION
Use of Linux device tree for portable abstraction of SoC for smart home secure gateway for RISC-V and (for comparison) ARM.

MOTIVATION
The device tree abstraction serves to describe the system at high level, and can be used for comparison of architecture of RISC-V and ARM.

TARGET DEMONSTRATORS
Smart Home

IMPLEMENTERS
CONS, SYSGO

LICENSE
Open Source (TBD)

LINK TO OTHER MODULES AND IPs
Requirements

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
D-FP-DT-01	F	Linux device tree	Report on demonstrator

COMMENTS
This serves to as part of demonstrator description.

4.1.6 Execution Framework for Space edge computing

DESCRIPTION
Development and testing a support framework for the deployment, management and testing of high-performance algorithm using RISC-V cores and accelerators, supporting the space demonstrator.

MOTIVATION
The demand for efficient AI execution has increased the need for an adaptable framework for a clean and efficient deployment of AI algorithm and heterogeneous computational unit in the new space edge-computing space System. The accelerators for the exploitation of this algorithm are used by means of a proper library made up by the opportune drivers. However, a framework that supplies a homogeneous and complaint support to the space development ecosystem and speed up the development and testing processes for space use cases can be very useful.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
TASI

LICENSE
Proprietary

LINK TO OTHER MODULES AND IPs

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
RISC-V-FRK-00	F	The framework must allow to monitor the execution of a computation	Functional Test
RISC-V-FRK -01	F	The framework must provide a command interface for the management and control a computation	Functional Test
RISC-V-FRK -02	F	The framework must support the deployment for parallel execution	Functional Test

COMMENTS

4.1.7 Development and Execution Framework for Automotive

DESCRIPTION
An Integrated Development Environment (IDE) is required to allow app development; easy code-writing, debugging, simulation and deployment in multiple targets (simulator, emulator, hardware) are powerful features any developer would like and use in creating parallel high-efficiency / low energy-consumption programs

MOTIVATION
A parallel accelerator like the XPU where algorithms should be mapped efficiently, may only be matched by custom software capable of exposing its many features. Having an intimate look into its inner workings will allow a developer to fully harness the power of such a heterogenous many-core system. An IDE capable of interfacing various flavors of the XPU (the simulator, emulator or hardware) will improve the coding speed and will allow efficient debugging sessions to happen.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
UNSTPB

LICENSE
Proprietary

LINK TO OTHER MODULES AND IPs

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
XPU-IDE-00	F	The IDE must allow development of source code for the various layers of software that will run on the XPU	Functional Test
XPU-IDE-01	F	The IDE must allow calling backend tools like asm or compiler (to generate object files or hex files)	Functional Test
XPU-IDE-02	F	The IDE must allow loading of program sections of hex files into the program memory of the simulator or hardware	Functional Test
XPU-IDE-03	F	The IDE must allow loading of data sections of hex files into the data memory of the simulator or hardware	Functional Test
XPU-IDE-04	F	The IDE must allow monitoring of the execution of a program when using the software simulator (program memory, data memory, register inspection...)	Functional Test

COMMENTS

4.1.8 Evaluation of TRISTAN software ecosystem results for ISOLDE usage

DESCRIPTION
Make applicable TRISTAN results available to ISOLDE.

MOTIVATION
RISC-V needs a strong software ecosystem. We intend to carry over results from TRISTAN, an adapt where needed.

TARGET DEMONSTRATORS
All

IMPLEMENTERS
SYSGO

LICENSE
Open Source (TBD)

LINK TO OTHER MODULES AND IPs
Requirements

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
D-TECO-DT-01	F	Software ecosystem available.	Report on demonstrator

COMMENTS

4.2 Application development tools design, implementation and testing

4.2.1 Deployment Flow

DESCRIPTION
Development of software toolchains for the optimization of AI models (architecture search, precision optimization, etc.) supporting heterogeneous hardware targets based on RISC-V.

MOTIVATION
A deployment flow that integrates neural architecture search (NAS) with an AI compiler is an indispensable component for achieving maximum efficiency in AI workload execution. NAS generates networks specifically tailored for the underlying hardware, creating layers and topologies that maximize its utilization. The AI compiler manages the critical elements of memory management and layer orchestration, which are foundational for optimizing neural network execution on hardware. Having such a tool is a crucial component for reaching high efficiency for AI inference on custom accelerators, given their specific architectures that support only a limited set of operators with a limited set of parameters.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
PoliTo

LICENSE
Apache License 2.0

LINK TO OTHER MODULES AND IPs
– AI support library (T4.1, PoliTo)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
D-FLOW-00	PR	The space demonstrator SHOULD use the deployment flow to reach the required accuracy, respecting the real-time constraint.	End-to-end maximum latency (T.B.D.) of Space demonstrator respected with networks that reach a sufficient accuracy (T.B.D.).

D-FLOW-01	F	The deployment flow SHOULD produce networks that run on the CVA-6 core, supporting accelerators employed in the Space demonstrator.	The deployment flow should optimize the inference on CVA-6 multi-core architectures and on the parallel processing unit from T3.4. Further, it should produce architectures that are more efficient in terms of accuracy/latency for the space use case.
D-FLOW-02	F	The deployment flow MUST support the AI libraries from T4.1 to create end-to-end NN prototypes.	The library should be benchmarked on end-to-end networks to verify compatibility with AI libraries.

COMMENTS

4.2.2 RF-SP: Resource-efficient IoT data processing and analytics

DESCRIPTION
Development of a resource-efficient IoT data collection and preprocessing library based on Apache StreamPipes

MOTIVATION
Develop a resource-efficient version of Apache StreamPipes for flexible IoT management. Building on generated adapters, this task addresses the development and deployment of a resource-efficient version of Apache StreamPipes for target architectures. The scope of this task is to investigate the underlying microservices infrastructure and develop a portable and resource-efficient minimal version of StreamPipes that can be used for edge implementations with limited computing power. The edge implementation will be complemented by an orchestration component for centralized resource management. The goal is to provide a lightweight edge component suitable for RISC-V that supports the data acquisition protocols needed for the smart home demonstrator and other IoT applications, and interacts with a central instance for configuration and data exchange.

TARGET DEMONSTRATORS
Smart Home

IMPLEMENTERS
Bytefabrik

LICENSE
Open Source (Apache 2.0)

LINK TO OTHER MODULES AND IPs
-

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
RF-SP-00	PR	The smart home demonstrator should be able to run the developed extensions on resource-constrained hardware.	Availability of an edge client that communicates with a central Apache StreamPipes instance.
RF-SP-01	F/PR	Edge Client: An edge client that should be able to extend the adapter library for connecting industrial data sources with an edge client for data collection and preprocessing, deployable in RISC-V platforms.	Minimal system requirements (tbd) for the edge client in terms of memory and CPU usage, data collection from example (Industrial) IoT protocols

RF-SP-02	F	Edge-Cloud orchestration: Concept and implementation that should be able to add/register the edge client to a central IoT data analytics infrastructure	The library should be benchmarked on end-to-end networks to verify compatibility with AI libraries.
RF-SP-03	STD	The developed extensions should be available in the Apache StreamPipes codebase	Availability in an official release

COMMENTS

4.2.3 Model-based generation framework for hardware safety pattern

DESCRIPTION
Model-based automation flow to alter the hardware architecture in a generator-based SoC design flow to address safety concerns.

MOTIVATION
Modern generator-based design flows make it easy to explore design alternatives that affect both hardware and software. The adjustments are made in a high-level SoC specification, e.g., in CHISEL. Currently this is a manual task. To reduce the manual effort and provide means for design space exploration, an approach is developed to specify safety mechanisms in a generalized, configurable way and semi-automatically apply them to the generator-based design flow. This way an automatic design space exploration regarding safety architectures can be supported.

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
FZI

LICENSE
TBD

LINK TO OTHER MODULES AND IPS
– T4.2 SoC simulator for Rocket Chip toolchain

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
RC-SAFE-01	F	The framework MUST support the user with the integration of safety mechanisms in the hardware architecture used in the Rocket Chip design flow	Semi-automatic integration of at least one safety mechanism
RC-SAFE-02	F	The framework SHOULD provide means to specify safety mechanisms in a general, abstract way	Modeling of various safety mechanisms
RC-SAFE-03	F	The framework COULD make these safety patterns configurable, to increase reusability	Model one safety mechanism with different characteristics

RC-SAFE-04	F	The framework SHOULD utilize the specified safety mechanism to alter the hardware architecture in the Rocket Chip design flow in a reusable way	Integration of at least one safety mechanism
RC-SAFE-05	F	The framework COULD utilize as many assets as possible available in the Rocket Chip design flow	Evaluation of the amount of information that can be reused in comparison to the amount specified manually
RC-SAFE-06	F	The framework WON'T have to provide means of design space exploration or automatic selection of safety mechanisms	

COMMENTS
A similar approach used currently only for software (1) should be evaluated for hardware. (1) https://universalsafetyformat.github.io/

4.2.4 Automated Precision Tuning Compiler

DESCRIPTION
Development of a precision tuning set of plugins for the LLVM compiler framework to support effective and automatic tuning of data sizes and operations to minimize latency and energy consumption

MOTIVATION
To improve energy-to-solution it is possible to trade off some computation accuracy, within the margins of error tolerance of the application. However, manual application of this technique is error-prone and tedious, so an automated tool is needed. Furthermore, the tool can be coupled with architectural optimization techniques (e.g., DMA) to achieve greater speedups and energy savings.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
POLIMI

LICENSE
Permissive open source (MIT)

LINK TO OTHER MODULES AND IPS
-

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
AXCC-01	F	The tool should apply precision tuning while keeping the computation average error under 10%	Comparison of results with the reference implementation provided by the unmodified code
AXCC-02	PR	The tool should obtain a performance improvement over the baseline	Comparison of execution times with the reference implementation provided by the unmodified code, employing the same hardware platform
AXCC-03	F	The tool should support IEEE754 as well as fixed point numbers. On compatible architectures, it should also support arbitrary size floating point	Capability verified on the demonstrator

COMMENTS

4.2.5 AI/ML Accelerator Compiler Toolchain

DESCRIPTION
API specification for library and compiler extension to support the efficient neural network accelerator.

MOTIVATION
AI accelerators are special-purpose units that offer optimized implementation of most common operators used in deep convolutional neural networks. However, they implement a subset of accelerated operators for specific parameters only. The compiler is the software that translates the input neural network into a specific program designed for this reduced set of instructions, considering the constrained parameters. Also, the compiled code is run on the accelerator with the help of the RISC-V accelerator working as host. To drive the accelerator, a specific RISC-V host program runs and can load the accelerator with both data and specific instructions, start the execution of the program on the accelerator and synchronize with it, awaiting termination and recovering the results.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
XPERI

LICENSE
Proprietary License

LINK TO OTHER MODULES AND IPS
AI/ML accelerator (T3.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
AICT-01	F	Support for IEEE754 FP16	Capability verified in the demonstrator.
AICT-02	F	Support for ONNX Convolution, MaxPool and Activation Function operations	Capability verified in the demonstrator.
AICT-03	PR	Performance of the accelerated program should surpass the one of a plain RISC-V implementation.	Performance comparison to be executed.

COMMENTS

4.2.6 TPU HAL

DESCRIPTION
Tensor Processing Unit Hardware Abstraction Layer

MOTIVATION
Efficient software support for TPU developed by UNIBO.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
UNIBO

LICENSE
Apache License 2.0

LINK TO OTHER MODULES AND IPS
- TPU (3.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
TPU-HAL-01	F	Support for IEEE754 FP16, Bfloat16, MiniFloat (8-bit 1-4-3, 8-bit 1-5-2)	-
TPU-HAL-02	F	Support for General Matrix Multiplication (GEMM) operations and General Matrix-Matrix Operations (GEMM-Ops)	-
TPU-HAL-03	PR	Expose >90% of hardware performance at software level	-

COMMENTS

4.2.7 HAL for RoT unit as a secure-coprocessor

DESCRIPTION
Hardware abstraction layer to enable the use of OpenTitan RoT (see 3.1.10) as a secure co-processor, by implementing an interface to standard security software libraries (OpenSSL) on Linux OS. Comparison with pure software implementation on aerospace benchmarks.

MOTIVATION
Software support for acceleration of cryptographic operations in OT.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
UNIBO

LICENSE
Apache License 2.0

LINK TO OTHER MODULES AND IPS
- RoT (3.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
TitanSSL-01	F	GP compliant	
TitanSSL-02	PR	Overhead of HAL layer	Simulation and FPGA prototype

COMMENTS

4.2.8 CFI software algorithm in OT firmware

DESCRIPTION
Firmware implementation on OT RoT to enforce CFI policy in OT (see 3.1.10). Implementation of backward and forward edge protection policies. Test of overhead on aerospace benchmarks.

MOTIVATION
Software support for acceleration of cryptographic operations in OT.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
UNIBO

LICENSE
Apache License 2.0

LINK TO OTHER MODULES AND IPS
- RoT (3.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
TitanSSL-01	F	OT RoT integrated	
TitanSSL-02	PR	Overhead of CFI policy	Simulation and FPGA prototype

COMMENTS

4.2.9 OT-based secure boot support (TitanBoot)

DESCRIPTION
Firmware implementation of secure boot procedure using OT.

MOTIVATION
Support for secure firmware verification.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
UNIBO

LICENSE
Apache License 2.0

LINK TO OTHER MODULES AND IPS
- RoT (3.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
TitanBoot-01	F	OT RoT integrated	
TitanBoot-02	PR	Overhead boot verification	Simulation and FPGA prototype

COMMENTS

4.2.10 OpenML Lite Offloading

DESCRIPTION
OpenMP lite offloading support for multicore RISCV systems

MOTIVATION
OpenMP lite is an implementation of OpenMP on a bareboard system without an operating system. The contribution is about adding offloading support to OpenMP lite solution for a system composed of an ARM target and a multicore RISCV architecture.

TARGET DEMONSTRATORS
TBD

IMPLEMENTERS
NXP-RO

LICENSE
Proprietary

LINK TO OTHER MODULES AND IPS
-

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
OMPLITE-1	F	OpenMPLite library shall be supported for a bareboard multicore RISCV architecture	The library should be benchmarked and verified on a multi-core RISCV architecture.
OMPLITE-2	F	Offloading shall be supported for an ARM host and a multicore RISCV architecture	The library should be benchmarked and verified on an ARM host and a multi-core RISCV architecture.
OMPLITE-3	F	Architecture information shall be collected from SHIM specification	The library should be benchmarked and verified on an ARM host and a multi-core RISCV architecture.
OMPLITE-4	F	Qualification tests for offloading support	The qualification tests should be run on an ARM host and a multicore RISCV architecture.

COMMENTS

4.2.11 LLVM and debugger support for Forward-edge CFI

DESCRIPTION
Extension of LLVM build tools and debugger to support a forward-edge control flow integrity unit (FCFI) which protects the instruction stream.

MOTIVATION
Adoption of control flow schemes strongly depends on a smooth integration in the build tools, to avoid additional effort at developers, as well as minimal code size impact.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
NXP-AT

LICENSE
TBD

LINK TO OTHER MODULES AND IPS
– NOEL-V 32-bit core (T2.1) – Safety & Security Modules (T3.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
BT-FCFI-01	F	The build tools shall provide attributes, to enable FCFI with a given security level on a per function basis.	-
BT-FCFI-02	F	The build tools shall provide an option, to enable FCFI with a given security level on a per file basis.	-
DBG-FCFI-03	F	The debugger shall support to debug code that is protected by FCFI.	-
BT- FCFI-04	PR	The build tools shall minimize the code size overhead, which shall be below 15%.	Benchmarks

COMMENTS

4.2.12 LLVM and debugger support for backward-edge CFI

DESCRIPTION
Extension of LLVM code generation tools and OpenOCD debugger, to support a backward-edge control flow integrity unit (BCFI) based on a shadow stack isolated by memory encryption / scrambling.

MOTIVATION
To support the BCFI scheme, the build tools need to support the instructions required for it. Furthermore, a debugger needs additional information to be able to support stack unwinding, as the content of the stack is isolated.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
NXP-AT

LICENSE
TBD

LINK TO OTHER MODULES AND IPS
– NOEL-V 32-bit core (T2.1) – Inline Memory Encryption / Scrambling Engine (T3.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
BT-BCFI-01	F	The build tools shall support to enable and disable BCFI dynamically per privilege level.	-
BT-BCFI-02	F	The build tools shall support the additional stack storage instructions, required for BCFI.	-
DBG-BCFI-03	F	The debugger shall support to read the (unencrypted/unscrambled) stack content, when BCFI is enabled.	-
BT-BCFI-04	PR	The build tools shall minimize the code size overhead required for BCFI.	-

COMMENTS

4.2.13 LLVM and debugger support for cryptographically tagged memory

DESCRIPTION
Build tools and debugger extensions to support a cryptographically tagged memory unit (CTM) based on memory encryption / scrambling.

MOTIVATION
To support the user (software developer) of a CTM scheme, the build tools need to take the memory tagging into account for code generation. When debugging such code, the debugger needs to support to read the tagged memory and show it decrypted to the user.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
NXP-AT

LICENSE
TBD

LINK TO OTHER MODULES AND IPS
– NOEL-V 32-bit core (T2.1) – Inline Memory Encryption / Scrambling Engine (T3.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
BT-CTM-01	F	The built tools shall support to use global and local data types in cryptographically tagged memory.	-
BT-CTM-02	F	The build tools shall support to allocate tagged memory.	-
DBG-CTM-03	F	The debugger shall support to read the unencrypted content of tagged memory.	-

COMMENTS

4.2.14 Debugger support for enclave memory isolation

DESCRIPTION
Debugger support of an enclave memory isolation unit (EMI) based on memory encryption / scrambling.

MOTIVATION
Debugging an encrypted code of a secure enclave, requires the debugger to be able to decrypt the code and data, so the user can see the code and data that is being debugged.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
NXP-AT

LICENSE
TBD

LINK TO OTHER MODULES AND IPS
– NOEL-V 32-bit core (T2.1) – Inline Memory Encryption / Scrambling Engine (T3.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
DBG-EMI-01	F	The debugger shall support to show the unencrypted code and data of the secure enclave that is currently debugged.	-

COMMENTS

4.2.15 LLVM/MLIR Extensions for PQC Accelerator

DESCRIPTION
Extension of LLVM/MLIR to implement instruction Set Extensions for PQC accelerators, this will likely follow the RISC-V special interest group's decisions on new instructions.

MOTIVATION
With the creation of a multipurpose PQC accelerator, the fastest performance would be achieved by an ISA extension, but in order to compile for such a device an extension of a chosen compiler is required.

TARGET DEMONSTRATORS
Automotive, possibly Space

IMPLEMENTERS
SAL

LICENSE
Multi-Licensing – Apache2.0, AGPL v3, Proprietary Licensing

LINK TO OTHER MODULES AND IPs
PQC Accelerators T3.5 (3.5.2)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
PQC-LLVM-01	F	Capable of compiling source code into firmware using new instructions relevant for the implemented PQC algorithms found in 3.5.2	Profiling and software testing of firmware running on completed, bare-metal implementation (FPGA/ASIC) provided in 3.5.2

COMMENTS
None.

4.3 SW for HW design and validation

4.3.1 Extra-functional simulation

DESCRIPTION
Integration of a RISC-V system with a framework for extra-functional simulation of properties like power, temperature, and reliability.

MOTIVATION
The design flow of modern systems requires taking into account extra-functional properties at all stages, to evaluate the mutual impact of functional and extra-functional aspects on the overall system behavior. This requires the construction of an environment where functionality and extra-functional models seamlessly interact and exchange information. The extra-functional properties should be modeled in an environment that natively supports dynamic and continuous-time aspects, but that can be at the same time easily integrated with the digital simulation of a RISC-V ISS/simulator.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
PoliTo

LICENSE
Apache License 2.0

LINK TO OTHER MODULES AND IPS
– CVA6 core (T2.1) – Models from WP3 Hardware blocks (optional)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
COMPUTE-ENGINE-00	STD	The extra-functional framework SHOULD allow an easy modeling of extra-functional aspects and support current standard languages.	The framework must comply with the current modeling and simulation standards.
COMPUTE-ENGINE-01	F	The extra-functional framework SHOULD integrate easily with the functional RISC-V environment.	The framework must be easy to integrate with the digital RSIC-V ISS/simulator in a simulation or co-simulation environment.

COMPUTE-ENGINE-02	F	The extra-functional simulator MUST reproduce the mutual impact of functionality and extra-functional aspects.	The extra-functional simulator must ease information exchange between the RISC-V ISA/simulator and the extra-functional models without impacting simulation performance.

COMMENTS

4.3.2 Time Contract Co-processor Compiler

DESCRIPTION
Methodology for modular/composable specification of timing constraints in Safety Island

MOTIVATION
The goal of this task is to design a compiler that corresponds to our machine language from the timing monitor coprocessor developed in T3.3. This compiler provides the ability to initialize the timing monitor coprocessor and to also reconfigure the specifications of the monitors during software updates.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
OFFIS

LICENSE
Apache License V2.0, TBC

LINK TO OTHER MODULES AND IPS
T1.2, T2.2, T3.1, T3.3

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
TCCP-CO-01	F	The compiler **MUST** accept a formal contract-based language as input language	Functional testing with the output TCCP
TCCP-CO-02	F	The compiler **SHOULD** support at least basic timing constraints (aging, event occurrence, reaction)	Verification by comparison of input timing constraints and extracted timing specifications from simulation
TCCP-CO-03	F	The result of the compiler **MUST** be in the correct format to be correctly loaded into TCCP-UNIT	Verified in system integration test

COMMENTS

4.3.3 Power/Performance Modeling for Design Space Exploration PPM-DSE

DESCRIPTION
ML-Based Methodology for modeling power/performance for CVA6 DSE

MOTIVATION
Our goal is to streamline power and performance design for CVA6, guiding users towards designs that either boost performance without increasing power or reduce power without compromising performance. To achieve this, we first simulate a CVA6 configuration to generate power and performance data, which we then use to build a machine learning (ML) model. The focus of our effort lies in optimizing this ML process, which involves simulation, data generation, and preprocessing to fit ML algorithms. We tackle feature selection for high-dimensional data and choose the most representative simulation samples. Leveraging ensemble and meta-learning, as well as Neural Architecture Search, we aim to fully utilize the data, ensuring the resulting model tackles issues like overfitting. With the optimized ML models, we can more efficiently explore the design space, sidestepping the need for exhaustive trials across numerous configurations. Once the models can reliably handle different benchmarks for a single configuration and adequately represent the entire configuration space, it is possible to integrate them into a target simulator. This integration enables precise power and performance predictions for new test configurations and supports the identification of optimal configurations within flexible design spaces.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
Silvaco

LICENSE
Proprietary

LINK TO OTHER MODULES AND IPS
CVA6 core (T2.1)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
PPM-DSE-01	F	This tool **WON'T** accept directly high-level inputs and will require preprocessing of power and performance traces	Partners will do preprocessing to generated power and performance traces from CVA6 RTL, GL
PPM-DSE-02	F	The tool **MUST** accept tabular input	Support CSV, TSV formats
PPM-DSE-03	F	The input **MUST** have at least one independent parameter column	The tool accepts one or many independent parameter columns

PPM-DSE-04	F	The input **MUST** have at least one dependent parameter column	The tool accepts one or many dependent parameter columns
PPM-DSE-05	F	The input **MUST** support different value types	Supported types include float, integer, boolean, enumeration
PPM-DSE-06	F	The input **SHOULD** support missing values	Tool runs even if value is missing in some raws
PPM-DSE-07	EOU	The tool **SHOULD** be able to do automatic parameters/features selection	Test suite will be implemented to measure the ease of the ML process after the Parameter selection phase
PPM-DSE-08	PR	Parameters/features selection **SHOULD** enhance accuracy and performance	This will be measured by RMSE, R2, relative error, accuracy, precision
PPM-DSE-09	PR	The Tool **SHOULD** be optimized to explore CVA6 configurations space	The performance enhanced for DSE will be compared to other classical solutions
PPM-DSE-10	PR	**SHOULD** implement ensembling and meta-learning techniques to enhance prediction accuracy.	Validate ensemble/meta models performance against individual models.
PPM-DSE-11	STD	Model output **MUST** be compatible with partners' flows	Output fromats to be specified by partners
PPM-DSE-12	F	The DSE exploration **MUST** have freedom to operate on reduced set of independent parameters	Users can assign values to a subset of independent CVA6 parameters
PPM-DSE-13	F	The DSE **SHOULD** assign values to unconstrained parameters	The design is specified by a set of selected parameters values representing one CVA6 configuration

COMMENTS
Addition of Ease-Of-Use (EOU) type

4.3.4 Fault-injection tools for robustness evaluation FFI-RE

DESCRIPTION
A bit-accurate FPGA-fault injection (FFI) tool with relevant fault-models for robustness evaluation of target RISC-V platform

MOTIVATION
This task will develop a bit-accurate FPGA-based fault injector, named BAFFI, that will significantly push forward the fault injection capabilities of existing tools for evaluating the robustness of modern hardware designs on Xilinx FPGAs. Unlike state-of-the-art tools, BAFFI will not impose area constraints on evaluated hardware components. The tool will make fault injection feasible despite the size of its target (even if it is an individual netlist cell) by only requiring the specification of its hierarchical path in the design tree under evaluation. This will be possible because BAFFI will define a bit-accurate mapping between the FPGA configuration memory and the main cells of the hierarchical netlist of the considered design. The result of this mapping will be the location of those essential configuration memory bits that can potentially be perturbed by faults during experimentation. The existence of this mapping will not only strongly reduce the fault injection space to consider, and the time required by fault injection campaigns, but it will also dramatically increase the complexity and dimension of component design that the tool will be able to evaluate. In addition, by eliminating the need for area constraints, BAFFI will also avoid altering the place and route results of the designs under test, which will reduce the level of intrusiveness required by fault injection with respect to existing state-of-the-art alternatives. The adaptation of BAFFI technology to the consideration of RISCV designs will allow a more precise dependability assessment of FPGA-based prototypes, which encompasses with the identification of its safety bottlenecks and the verification of their integrated safety/security mechanisms.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
UPV

LICENSE
MIT

LINK TO OTHER MODULES AND IPS
WP2

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
FFI-RE-01	PH	MUST support modern Xilinx FPGAs (7-series and Ultrascale+)	FFI experiments on relevant FPGA designs

FFI-RE-02	F	MUST support fault injection into design hierarchy from architectural components up to individual netlist cells	FFI experiments on relevant FPGA designs, comparison with simulation-based fault injection (where applicable)
FFI-RE-03	F	MUST support emulation of representative permanent and transient faults	FFI experiments on relevant FPGA designs, comparison with simulation-based fault injection (where applicable)
FFI-RE-04	F	MUST support statistical fault injection (constrained-random fault-loads), as well as directed fault injection (manually defined fault-loads)	FFI experiments on relevant FPGA designs, comparison with simulation-based fault injection (where applicable)
FFI-RE-05	F	SHOULD support automated robustness assessment of RISCV FPGA prototypes	FFI experiments on RISCV FPGA prototypes

COMMENTS

4.3.5 SoC simulator for Rocket Chip toolchain

DESCRIPTION
Simulation-based HW/SW co-verification approach for the Rocket Chip generation pipeline

MOTIVATION
Modern generator-based design flows make it easy to explore design alternatives that affect both hardware and software. Examples are the sizing of (AI) accelerators or the integration of safety mechanisms. For a ‘fast’ exploration of various design alternatives a simulator is required to evaluate the functional behavior, to debug the hardware/software system or to estimate non-functional properties. To cope with the generator flexibility the SoC simulation has to be (partially) derived from the hardware design flow to minimize manual effort. For processor simulation existing simulators such as QEMU can be used. To consider peripherals like the mentioned (AI) accelerators into account, the QEMU based emulator has to be extended with dedicated hardware models. Both parts should be generated/configured semi-automatic, based on the information already used in the hardware generator flow.

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
FZI

LICENSE
TBD

LINK TO OTHER MODULES AND IPS
– T4.2 Model-based generation framework for hardware safety pattern

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
RC-SIM-01	F	The framework MUST support the assets provided by the Rocket Chip generator	Reuse of available assets in the Rocket Chip design flow
RC-SIM-01	F	The framework MUST reuse existing information to derive HW simulation models	Integration of an AI accelerator in the SoC design
RC-SIM-02	F	The framework MUST reuse existing information to configure QEMU	Integration of a RISC-V based processor in the SoC design
RC-SIM-03	F	The framework SHOULD use information like device trees to configure an embedded OS	Integration of an embedded OS, such as Zephyr

RC-SIM-04	F	The resulting simulation environment COULD provide fault injection	TBD – Assessment of different safety architectures

COMMENTS

4.3.6 Software simulator for the AI/ML Accelerator

DESCRIPTION
Software simulator, efficiency benchmarking and automation of analysis and simulation tools for the efficient Neural Network Accelerator.

MOTIVATION
Compilers for AI accelerators are complex pieces of software. On implementing the software toolchain, the developer needs to ensure that the results of the implemented operation decomposition are still consistent with the original results of the operation as produced by a standard implementation. Moreover, these results should be verified as the model of the designed compiler functionality builds up. The systematic testing is implemented by creating a software model of accelerator's implemented operations that can be verified against both the standard and the hardware implementation. Systematic testing can be achieved by continuous integration, using regression tests. The efficiency of the neural network implementation can also be assessed by determining, from the compile phase, the number of cycles used by an operation. Thus, performance of the accelerated neural network model can be compared against a direct plain execution on the RISC-V processor.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
XPERI

LICENSE
Proprietary License

LINK TO OTHER MODULES AND IPS
AI/ML accelerator (T3.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
SSIM-01	F	Implementation of software simulator	Automated unit tests comparing with the reference model
SSIM-02	F	Systematic unit tests for the model	-
SSIM-03	F	Off-line comparison of accelerated vs. plain RISC-V implementation	Verified through FPGA use case implementation.

COMMENTS

4.3.7 Fault-injection RTL-simulation

DESCRIPTION
The goal is to extend the popular Verilator RTL simulator with a mechanism to instrument designs with extension points. Then utilize those capabilities in an open source framework for constrained random co-verification of hardware and software mitigations.

MOTIVATION
Fault injection techniques are needed in safety and security contexts to test countermeasures and increase robustness. A generalized approach for the instrumentation of RTL designs in Verilator is very useful beyond this project. Working on upstream Verilator increases the impact. To allow for the co-verification of hardware and software countermeasures, an extensible framework to build on this basic functionality is needed.

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
HM

LICENSE
LGPL 3.0, Apache 2.0

LINK TO OTHER MODULES AND IPS
None

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
VFINJ-01	F	Implementation and upstreaming of extension points in Verilator	Functional verification and regression tests
VFINJ-02	STD	Research on standard fault models	-

COMMENTS

4.3.8 Open Source Simulation Framework (IFX WI4.3.*)

DESCRIPTION
Providing and Analyzing and Open Source Tool Chain for Simulation.

MOTIVATION
It is beneficial to have open-source tool simulation support in addition to open-source cores. As these tools are said non-competitive in Simulation and Verification, a flow and a comparison with commercial tools would be beneficial.

TARGET DEMONSTRATORS
Automotive

IMPLEMENTERS
IFX / HM

LICENSE
Same as the open source tools

LINK TO OTHER MODULES AND IPS
Open Source Cores

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
OSV-01	F	Have a requirement specification / use cases for the open source verification approach	review
OSV-02	F	Have a concept for the open source verification approach. Define tool of choice (verilator)	Review
OSV-03	F	Implement the verification approach that allows for comparison of the open source verification tool and a commercial solution	Test cases
RC-SIM-03	F	Comparison Report	Review

COMMENTS

4.3.9 Tooling to ensure compliance with multicore timing certification objectives

DESCRIPTION
Analysis of the multicore hardware architecture developed in ISOLDE, with focus of the timing impact of the implementation of shared hardware resources in the processor, as well as any other features inherent to multicore impacting predictability in timing of software applications running in the platform. A set of artifacts aiding the collection of evidence of compliance of certification objectives with respect to multicore standards in the applicable domains will be developed. They include the integration with Rapita's RVS RapiTime tool of the software of the use-cases.

MOTIVATION
Inherent features of multicore processors introduce a high level of variability in actual timing of application tasks of running software, with respect to execution in isolation. Certification standards for application domains of the space and automotive use-cases proposed in ISOLDE require providing evidence on compliance of deterministic timing bounds for safety critical applications running on multicore platforms. Rapita has developed a measurement-based approach addressing the generation of the necessary set of documents for certification on the applicable standards, which will be ported to the actual hardware platform resulting of ISOLDE.

TARGET DEMONSTRATORS
Space, Automotive

IMPLEMENTERS
RAPITA

LICENSE
RAPITA Proprietary

LINK TO OTHER MODULES AND IPS
T2.1 Processor IP: Noel-V and CVA6 cores T2.2 Peripheral and interconnect IP cores T2.4 Software interfaces to general purpose cores T3.3 Integration of monitoring capabilities

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
RAPITA-01	STD	Enumerated identification of interference channels (ICID) for the ISOLDE hardware platform	Verification of actual interference via experiments using benchmark code
RAPITA-02	STD	Enumerated identification of critical configuration settings, as set of registers (CCSID) impacting multicore interference in the ISOLDE hardware platform	Benchmark-based verification of different selected scenarios

RAPITA-03	STD	Enumerated identification of hardware event monitor events on existing hardware resources in the platform (HEMID) aiding the measurement-based characterization of identified interference channels	Library accessing those HEMs will be developed. Correctness of counts of HEMs implemented by the library will be checked by executing snippets of code producing a deterministic number of events and checking those against HEM reading results.
RAPITA-04	STD	Developed set of RapiDaemons (benchmarks) capable of exercising the identified interference channels	On-target profiling of the RapiDaemons based on their footprint on HEMs. Resulting HEM values for each RapiDaemon are checked against their design spec.
RAPITA-05	STD	Developed observability library, to be executed in the ISOLDE hardware platform, able of configure/start/stop/reset/read the identified hardware event monitors characterizing interference	The correctness of implementation of the library will be checked by executing snippets of code producing a deterministic number of events and checking those against HEM reading results.
RAPITA-06	STD	Characterization documents for identified interference channels based on definition and execution of test scenarios using the developed RapiDaemons and identified hardware event monitors relevant to each interference channel	Cross-check with Interference Channel identification and HEM footprint of the benchmark code executed in each scenario of the characterization.
RAPITA-07	STD	Set of accompanying documentation (requirement and verification artifacts) for both RapiDaemons (benchmarks) and identified hardware event monitors aiding certification	Traceability check
RAPITA-08	STD	Integration of developed observability library and RapiDaemons with RVS RapiTime for the auto and space use-cases	Functional verification using on-platform test scenarios during RapiDaemon design and/or Interference Channel characterization

COMMENTS

- Access to TRM / technical documentation of the hardware design, describing the platform, for CVA6 (Space use case) and NOEL-V (Auto use case), including a list with a detailed description of countable events via every PMU units of the platform, and identification and description of all registers in the register maps of the cores and all IPs in the SoC will be needed to perform the analysis tasks (RAPITA-01 to 05).
- In RAPITA-08, it's assumed that hypervisors & OS for space and automotive use-cases (Sysgo, Continental and Fentiss) provide the necessary accesses to:
    - Registers related to Critical Configuration Settings regarding multicore interference.
    - Registers related to the hardware event monitoring resources.

4.3.10 Tool based on the LLVM compiler to provide WCET estimations

DESCRIPTION
Development of a set of plugins and passes for the LLVM compiler framework to estimate the WCET during the compilation process and brings this information to the Intermediate Representation (IR) of LLVM. Having the WCET information knowledge at IR-level allows novel mechanisms in the compilation stage in order to perform better optimizations and inject hooks to provide run-time information to the real-time scheduler. This hybrid analysis compiler-based tool will be integrated with the probabilistic-WCET measurement-based tool called *chronovise* and developed by POLIMI since 2018.

MOTIVATION
Estimating the WCET is a challenging problem for any non-trivial application and hardware architecture. A multi-core RISC-V processor targeting critical application domains need to integrate innovative technologies to estimate the WCET. This tool explores the possibility to use the compiler as a mechanism to, on the one hand, optimize the WCET of the applications during the compilation process, on the other hand, help in the (probabilistic-)WCET estimation.

TARGET DEMONSTRATORS
Space

IMPLEMENTERS
POLIMI

LICENSE
Open Source

LINK TO OTHER MODULES AND IPS

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
P-WCET-01	F	Implementation of the plugins and LLVM passes to obtain the WCET estimation at IR-level	Functional verification using existing benchmarks and LLVM suite for RISCV
P-WCET-2	PR	Accuracy of the WCET estimation obtained by the compiler	Comparison with existing static tools and/or simulations
P-WCET-3	PR	Code optimization and instrumentation targeting WCET	Differential comparison between binaries via static analyses or simulations

COMMENTS

4.3.11 Safety analysis of Safety Island by Fault injection

DESCRIPTION
The creation of various reference workloads is done with the aim of recreating as many system states from real applications as possible. Reference workloads are essential for the reproducible determination of any implementation and integration errors. The security analysis is carried out using the fault injection framework (HDFIT) provided by the project partner, based on reference workloads.

MOTIVATION
Gain information of the safety level of the Safety Island

TARGET DEMONSTRATORS
Smart Home, Automotive

IMPLEMENTERS
UZL

LICENSE
Open Source (TBD)

LINK TO OTHER MODULES AND IPs
Requirements (T1.4)

REQUIREMENTS			
ID	TYPE	DESCRIPTION	VERIFICATION
SA-FI	F	Reference Workloads of real-world Applications	Simulation

COMMENTS

5 Conclusions

This Deliverable collected an initial list of requirements and specifications for the various components to be developed in ISOLDE, building on the previous initial demonstrator requirements defined in Deliverable D1.1. For simplicity in matching against the project structure, these components have been divided according to the WP of reference (foundation cores, accelerators, software) and further subdivided according to the WP tasks. An initial tentative definition of IP for each component has also been provided.

Based on this document, the work in WP2, WP3 and WP4 can now start with a clearer picture of what would be needed and the interdependencies of components across tasks; in addition, work in WP5 can benefit from having provided an initial list of items to be tentatively used by the four different demonstrator (Space, Automotive, Smart Home, IoT).

A refined and consolidated version of this initial report will be provided in Deliverable D1.4 to be due in M15.