Project:	ISOLDE: customizable Instruction Sets and Open Leveraged Designs of Embedded RISC-V processors
Reference number:	101112274
Project duration:	01.05.2023 - 30.04.2026
Work Package:	WP1: Requirements and Specifications
Deliverable	**D1.3**
Title	**Consolidated Demonstrators Requirements and Specifications**
Type of deliverable:	Report
Deadline:	30.04.2024
Creation date:	30.04.2024
Authors:	Patrick Pype, Bogdan Kiszka, Antonio Sciarappa, Cosmin Moisa, Danut Rotar, Mikai Hurdugaciu, Aurel Gontean, Holger Blasum, Darshak Sheladiya, Samuel Ardaya-Lieb, Thomas Hartmann, Dominik Riemer, Michael Gautschi, Yvan Tortorella, Francesco Conti, Alessio Burrello, Daniele Jahier Pagliari, Calin Bira, Dario Pascucci, Antonio Leboffe, Alessandro Marini, Paolo Serri, Catriel De Biase, Davide Di Ienno, Federico Reghenzani, Giovanni Agosta, Martin Hobelsberger, Alexandru Pușcașu, Cătălin Ciobanu, Esther Soriano, George Suciu, Mari-Anais Sachian, Mattia Paladino, Josep Jorba, Wolfgang Ecker
Involved partners:	CAR-RO, CONS, E4, IFX, LDO, NXP-AT, NXP-CZ, POLIMI, POLITO, SAL, SYSGO, TASI, UNIBO, HM, BEIA, UNSTPB, FENTISS, IMT, RAP-ITA

Table of Contents

1 Introduction

1.1 Executive Summary

The purpose of this document is to collect the requirements for the project demonstrators in the chosen application areas (Automotive, Space, Smart home, IoT), as worked on in T1.1 “Collection of requirements of the demonstrators & high-level SoCs specifications” (M1-M12). The base for this document is the information contained in D1.1 “Demonstrators requirements and specifications”, and D1.2 “Requirements and specifications on architecture, hardware and software modules and IPs”, which has been updated following the progress of the work done in WP2, WP3, WP4 and WP5 adding also more detail on IPs implementation as well as the definitions of the architectures of the planned demonstrators. D1.3 is the final outcome of T1.1.

The document contains both functional and non-functional requirements for the following subjects: architecture (components, interfaces, accelerators), measurable metrics and goals (performances, power consumption, radiation tolerance, security metrics to be used), SW (stacks, tools, system SW features, SW for demonstrators use cases). There are also references, where significant, to standard to be used/considered.

D1.3 includes an overview of each demonstrator, whose objectives and proposed architecture are described in more detail in D5.1 “Description of demonstrators architecture” (due M12, same as this document), to contextualize the requirements and the verification strategy attached to them.

This document supports the technical report associated to the deliverable D1.1 “Demonstrators Requirements and Specifications” reporting to the WP1 “Requirements and Specifications” leader (E4 Computer Engineering S.p.A.) for the Project ISOLDE.

1.2 Definitions and Acronyms

Abbreviation	Description
AI	Artificial Intelligence
AMD	Advanced Micro Devices (Company)
ARM	Processor company, formerly Advanced RISC Machines
ASIC	Application Specific Integrated Circuit
BSD	Berkeley Standard Distribution
CA	Consortium Agreement
CCAM	Connected, Cooperative and Automated Mobility
CFI	Control Flow Integrity
CFR	Control Flow Reconstruction
CLIC	Core Level Interrupt Controller
CLS	Controllable Local System
CPU	Central Processing Unit
CSI	Camera Serial Interface
CU	(Timing) Continuous updates
CUDA	Compute Unified Device Architecture
DM	(Timing) Direct Meetings
DRAM	Dynamic Random Access Memory
DSE	Destructive Single Event
EC	European Commission

Abbreviation	Description
ECC	Error Check and Correct
EDA	Electronic Design Automation
EEE	Electrical, Electronic and Electro-mechanical
EMS	Energy management system
EoL	End-of-Life
EPI	European Processor Initiative
EY	(Timing) Each Year
FC	(Timing) Final Conference
FIFO	First In First Out (Buffer)
FLOP	Floating Point Operations per Second; unit to measure computing performance
FMC	FPGA Mezzanine Card
FPGA	Field-Programmable Gate Array
FPU	Floating Point Unit
GAFA	Google, Amazon, Facebook, Apple
GCC	GNU Compiler Collection
GFLOPS	Giga Floating Point Operations per Second
GPU	Graphic Processing Unit
GTH	Gigabit Transceiver type H (Xilinx/AMD)
GPP	General Purpose Processor
HBM	High Bandwidth Memory
HEMS	Home Energy Management System
HPC	High Performance Computing
HSI	Hardware-supported instrumentation
HW	Hardware

Abbreviation	Description
I3C MIPI	I3C successor to I2C
ICT	Information and Communication Technology
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IEEE	Institution of Electrical and Electronic Engineers
IOMMU	I/O Memory Management Unit
IoT	Internet-of-Things
IP	Intellectual Property
ISA	Instruction Set Architecture
ISO	International Standards Organisation
ISS	Instruction Set Simulator
JTAG	Joint Test Access Group
KDT	Key Digital Technologies
KDT-JU	KDT Joint Undertaking
KPI	Key Performance Indicator
LET	Linear Energy Transfer
LLVM	Low Level Virtual Machine
LTE	Long-Term Evolution (4G LTE)
MIPI / CSI	Mobile Industry Processor Interface / Camera Serial Interface
ML	Machine Learning
MTC	Machine-type communication
MTH	(Timing) Mid Term
NFC	Near Field Communication
NR	New Radio (standard)
OBI	On Board Interconnect

Abbreviation	Description
OS	Operating System
OSI	Open Standards Inter-connect
PCI	Peripheral Component Interconnect
PCIE	PCI Express
PDK	Physical Design Kit
PHY	PHYsical layer (imple-mentation)
PKI	Public Key Infrastructure
PMOD	Peripheral Module (inter-face)
PQC	Post Quantum Cryptog-raphy
PV	Photovoltaic
QSPI	Quad Serial Peripheral Interface
R&I	Research and Innovation
RISC	Reduced instruction set computer
RISC-V	Specific Open-Sourced RISC ISA
RTL	Register Transfer Lan-guage
RTO	Research & Technology Organisation
SDC	Synopsys Design Con-straints File
SDIO	Serial Digital Input/Out-put
SGA	Specific Grant Agree-ment
SHAKTI	RISC-V CPU from RISE, India
SIL	Safety Integrity Level
SIMD	Single Instruction Multi-ple Data

Abbreviation	Description
SME	Small/Medium Enterprise
SMGW	Smart Meter Gateway
SNS	Smart Networks and Ser-vices
SoC	System on Chip
ST	(Timing) Start of TRIS-TAN project
SW	Software
TC	(Timing) Targeted Con-ference
TF	(Timing) Targeted Fair
TID	Total Ionizing Dose
TNID	Total Non-Ionizing Dose
TRL	Technology Readiness Level
TSN	Time Sensitive Network-ing
UART	Universal Asynchronous Receiver/Transmitter
VHDL	VHSIC Hardware De-scription Language
VHSIC	Very High-Speed Inte-grated Circuits
VP	Virtual Prototype
VRP	Variable Precision
VXP	Variable Extended Preci-sion
WCET	Worst-Case Execution Time
WI	Work Item
WP	Work Package
X86	(Intel originated) X86 in-struction set
XNG	XtratuM Next Generation
XPU	X Processor Unit (CPU, GPU …)

2 Space Demonstrator

2.1 Overview of the Demonstrator

The space use case demonstrator focuses on testing the feasibility of performing inference of different deep learning models on a RISC-V processor aboard a satellite. The demonstrator will use different data sources, such as hyperspectral/multispectral imaging data for image processing and on-board telemetry data for satellite health monitoring.

As first use case, the demonstrator showcases the capabilities of the RISC-V-based space processor in supporting image processing tasks. The data consists of hyperspectral/multispectral images, where each pixel represents the reflectance of light within a fixed spectral range. To ensure computational efficiency for space applications, the model processes the image on a pixel-by-pixel basis, determining whether each pixel corresponds to the target material. The algorithm performs data normalization and a lightweight coarse co-registration and leverages neural network-based models for target classification. Furthermore, the demonstrator explores potential opportunities for parallelizing the processing, whenever applicable.

As second use case, the demonstrator showcases the capability of the RISC-V-based space processor to support Machine Learning (ML) applications, specifically for telemetry analysis. The ability to detect anomalies and to identify trends in satellite data is crucial for ensuring mission success while preventing costly downtimes. By leveraging advancements in ML algorithms, we can train models to recognize patterns in telemetry data and to alert operators on potential issues before they become critical.

One possible scenario in focus involves using sensor data from various subsystems within the satellite to monitor performance and to identify deviations from the expected behaviour. For instance, the demonstrator could utilize accelerometer readings to track changes in the orientation of the satellite or gyroscope measurements to assess spin rate fluctuations. Additionally, it could analyse temperature and power consumption data to ensure that the systems of the satellite are operating within safe margins.

To implement this use case, the demonstrator employs a combination of feature engineering and model training techniques. Feature engineering involves extracting relevant information from raw telemetry data, such as time-series data, and transforming it into features suitable for model training. This may include techniques like normalization, smoothing, and extraction of relevant frequency components.

Once the features are prepared, the demonstrator utilizes the already (on-ground) trained ML algorithms to identify patterns and anomalies in real-time data, which can then be used to generate alerts and notifications for operators on-ground or for on-board autonomous failure isolation and recovery system. The latter mechanism is called Fault Detection, Isolation and Recovery (FDIR) system and is currently based on traditional Out-Of-Limit (OOL) techniques, where an alert is sent only if the telemetry exceeds the pre-fixed upper and lower limit values. Injecting the ML models in this system can enhance the FDIR capabilities in anomaly detection.

It is important to notice that the current state of space technology poses challenges for performing the entire processing on-board a satellite (training and inference). As a result, the primary focus of our activities lies in optimizing the inference phase, which is well-suited for on-board execution, leaving the training phase to the beefy servers and workstations on-ground.

2.1.1 Description of the space processor

The space processor employed in the demonstrator is based on the CVA-6[1] RISC-V architecture. RISC-V processors are known for their energy efficiency, making them well-suited for satellite applications where power consumption, among other requirements, is critical. By utilizing RISC-V, the processor ensures optimal performance while adhering to the stringent resource constraints of space missions, with the possibility to benefit from the improvements coming from the community of RISC-V developers[2].

2.1.2 Functionalities and capabilities

The demonstrator showcases the ability to perform inference of a deep learning model directly on the RISC-V-based hardware present onto the satellite. This capability enables processing data onboard the satellite itself, thus reducing the need for extensive data transmission back to earth for analysis.

2.1.3 Key features and benefits

The primary benefit of demonstrating deep learning model inference on board of a satellite lies in the ability to move a significant portion of the computation to the edge. By processing the data locally on the satellite using the RISC-V processor, only the relevant results and features extracted need to be transmitted to earth. This approach significantly reduces transmission time, data bandwidth, power consumption and associated costs. Moreover, it enhances the overall efficiency of satellite missions, as critical decisions can be made in real-time, thus opening new possibilities for space-based research, remote sensing applications and new related services.

2.2 Functional and Non-Functional Requirements

In this Section we collect the functional and non-functional requirements for the space use case relative to the components developed in the various WP (Cores, Accelerators, Software).

2.2.1 Cores

Regarding the cores developed in WP2, the space demonstrator requires:

- The 64-bit CVA-6 processor core
- CVA-6 improvements and extensions developed in ISOLDE

2.2.2 Accelerators

Relatively to the accelerators developed in WP3, based on their maturity level and their potential performance improvements to the two proposed AI / ML applications described in the previous Section, the space demonstrator includes one or more of the following items:

- Tensor Processing Unit (TPU) for AI (T3.4) for accelerating AI inference on board. The TPU is expected to reside in the Accelerating cluster, which is based on the PULP cluster[3].

[1] GitHub - openhwgroup/cva6: The CORE-V CVA6 is an Application class 6-stage RISC-V CPU capable of booting Linux

[2] Technical Working Groups - Home - RISC-V International (riscv.org)

[3] Ref.: https://github.com/pulp-platform/pulp_cluster

- Parallel Computing Engine (T3.4) for accelerating AI and ML applications on board
- Accelerators for post-quantum cryptographic accelerators (T3.5)
- Root-of-Trust Unit (T3.1)

Other potentially interesting items to be considered are the following:

- Custom arithmetic units for AI and their power monitors (T3.2, T3.3) for accelerating AI inference and evaluating its performance.
- SIMD/Vector unit and relative scratchpad memories (T3.2, T3.4) for accelerating the use case application.

2.2.3 Software

Concerning the software stack developed in WP4, the space demonstrator may require all or a subset of the following elements, also depending on their proper functioning by the end of the project (as assessed by the IP developers and the application developers) and degree of utilization of the accelerators:

- Accelerated AI libraries and related software toolchains (T4.1, T4.2) to deploy the AI application using the relative WP3 accelerator and the CVA-6 processor. Based on the requirements and the chosen hardware, different libraries and optimization techniques can be employed to enhance the performance of the space demonstrator.
- System software supporting the space demonstrator (T4.1) to effectively run the use case application. The foreseen demonstrator involves neural network inference and/or execution of heavy algorithms; to optimize the execution, a system software dedicated to the target will be developed supporting the space demonstrator needs. The main requirements of the system software are to provide:
  - a way to execute cyclic activities (with or without OS).
  - a Command and Control front end to the operator exploiting one of the target input/output board communication links, where the Command and Control SW shall be able at minimum to start/stop an algorithm execution as well as monitor the execution status.
  - a SW abstraction layer to leverage the HW accelerator.
- Analyse hypervisor support SYSGO (T4.1).
- Compiler and compiler extensions (T4.2) to compile the application enabling utilization of the WP3 accelerators; more precisely:
- A compiler extension that can automatically manage the performance trade-off of floating point/fixed point arithmetic;
  - An automatic way to estimate the WCET of the real-time tasks during the compilation process and, possibly, perform optimizations targeted at improving the WCET.
- Software support for deploying the AI apps using the Tensor Processing unit and the Root-of-Trust unit (T4.2).
- System-level RISC-V simulators (T4.3) of extra-functional properties. Jointly with the target hardware functional simulators, the extra-functional simulator can be used to estimate the power, energy, and latency of the space demonstrator while prototyping the first version. The data can be used to finetune the application further and to fulfil the use-case constraints.

- Analysis and characterization of timing interference between the CVA6 core and the accelerators in the SoC, and relevant configuration settings, including a port of Rapita's RVS RapiTime WCET measurement tool[4] based on available hardware event monitors in the platform (T4.3)
- Software power and timing analysis tools (T4.3) for collecting performance measurements of the use case, using novel ways to estimate the WCET (including probabilistic analysis) to overcome the traditional issues of static analyses.
- XNG hypervisor support for CVA-6 core (T4.1) to enhance the safety of the space demonstrator. The hypervisor provides a partitioned system in which different mixed-critical application can be executed together in different partitions.
- XNG hypervisor virtualization support for analysis, monitoring, and testing (T4.3). Port of XNG hypervisor providing support for SW timing analysis tools developed by RAPITA. Provide hypervisor services allowing access to necessary on-chip resources used by timing analysis tools, including the different HW Event Monitors in the platform and critical configuration settings relevant to contention. Also, provide a building environment in order to ease the development and building of the tools which support XNG hypervisor.
- ML based power/performance trade-offs modelling of CVA6 core used for the space demonstrator to find best trade-off in order to increase autonomy when performing on-board processing (T4.3).

2.2.4 Other requirements

Model Specification: The model employed for image processing will be based mainly on convolutional layers including both 3D and 2D filters. On top of all the convolutional layers, a fully connected layer is provided to classify the target pixels. Therefore, a mix of convolutional and fully connected layers define the entire architecture. The models employed for telemetry analysis, as described in D5.1, range from classical ML to DL methods. In particular, the NNs include Convolutional (1D) layers as well as LSTM architecture.

Data Requirements: The data elaborated by the model will be stored in .npy and/or tensor .pt format. They represent tensors of WxHxC dimension where WxH represents the spatial size and C is the number of channels containing all the band information. The values included in the matrix are represented as IEEE 754 32-bit single-precision floating-point format. Regarding the telemetry analysis use case, the input is the one expected for Conv1D: BxCxL, where B is the batch size, C is the number of channels and L is the length of the sequence. The input tensor is represented as IEEE 754 32-bit single-precision floating-point format.

2.3 Measurable metrics

In this Section we collect the measurable metrics for the space use case, indicating the requirements that the final product should satisfy in the case of its actual utilization on satellites. While it is not possible to validate some of these requirements during the execution of the project, since this can only be done with a final chip available and under particular experimental conditions, we can still identify an initial list, with the purpose to verify if some items need to be further investigated or if some appropriate estimates can be conceived.

[4] https://www.rapitasystems.com/products/rapitime

The verification methods are defined, based on *ECSS-E-ST-10-02C Rev.1*[5], as follows:

- Test [T] for those requirements that need to be tested on hardware to verify their functions and/or performance.
- Review of Design [R] for those requirements that can be verified by checking the project documentation (e.g., design documents, reports, technical descriptions).
- Analysis [A], for those requirements that can be verified performing theoretical or empirical evaluation (e.g., computational simulation, systematic or statistical design analysis). Verification by analysis include the possibility to demonstrate qualification of a product by similarity with an already qualified product, more details in ECSS-E-ST-10-02C Rev.1.
- Inspection [I], for those requirements that can be verified by visual determination of physical characteristics (e.g., constructional features, hardware conformance to document drawing)

2.3.1 Performance Requirements

SPACE-001	**Chip Core**	Verification: [R]
The chip shall contain, in its HOST part, a single core		

SPACE-002	**Chip Cluster SIMD**	Verification: [R]
The chip shall, in its CLUSTER part, work in a configuration scalable up to at least 8 (To be confirmed) cluster x 16 (To be confirmed) cores, which may be configured to work either with internal redundancy or purely in parallel. *Note: The configuration to be implemented in the final ASIC solution is determined based on the outcome of the FPGA prototyping activities.*		

SPACE-003	**Cluster FLOP Performances**	Verification: [A]
The CLUSTER part of the chip shall guarantee at least 500 GFLOPS *Note: It is expected to reach this performance level with an ASIC implementing a CLUSTER with at least 8 cluster x 16 cores.*		

SPACE-004	**HOST DMIPS Performances**	Verification: [T]
The processor core used in the HOST part of the chip shall guarantee at least 1.7 DMIPS/MHz Dhrystone MIPS (Million Instructions per Second). *Note: It is expected to reach at least 1700 DMIPS @1GHz*		

SPACE-005	**HOST FLOP Performances**	Verification: [T]

[5] ECSS-E-ST-10-02C Rev.1 – Verification (1 February 2018) | European Cooperation for Space Standardization

The processor core used in the HOST part of the chip shall guarantee at least 0.3 FLOP/cycle. *Note: It is expected to reach at least 270 MFLOPS @1GHz*

2.3.2 Design Requirements

SPACE-006	**Chip Architecture**	Verification: [R]
The chip architecture shall be the RISC-V CVA6 core.		

SPACE-007	**Processor Features**	Verification: [T]
The chip architecture shall support at least the following features: • cache coherency management; • advanced interrupt support (e.g., platform interrupt controller); • virtual memory management; • virtualization management (i.e., hypervisor support).		

3.3.3 Power Requirements

SPACE-008	**Power Consumption**	Verification: [A]
The chip shall not exceed a power consumption of 4 W (TBC) in an operational environment of +125 C° @ EOL.		

SPACE-009	**Current Consumption**	Verification: [A]
The chip shall not exceed a power consumption of 750 mA (TBC) in an operational environment of +125 C° @ EOL.		

SPACE-010	**Input Voltage**	Verification: [A]
The chip shall work with an input voltage of 3V (TBC) in an operational environment of +125 C° @ EOL.		

SPACE-011	**Exceeding Maximum Rating**	Verification: [R]
All the Electrical, Electronic and Electro-mechanical (EEE) components in which exceeding absolute maximum rating conditions might cause instantaneous or very short-term unrecoverable hard failure (destructive breakdown) shall be reported in the design document with a table like the following one:		

PARAMETER	MIN	MAX	UNIT
VDIG-3Vx (Part name #1)	0.3	4.5	V
VDIG-3Vy (Part name #2)	20	150	mA
…	…	…	…

Note: The standard ECSS-Q-ST-60C defines the requirements for selection, control, procurement and usage of EEE components for space projects.

SPACE-012	Working near Maximum Rating	Verification: [R]
All the EEE components that are stressed from operating near absolute maximum levels that affect long-term reliability of the device shall be reported in the design document with a table like the following one:		

PARAMETER	MIN	MAX	UNIT
VDIG-3Vx (Part name #1)	0.3	4.5	V
VDIG-3Vy (Part name #2)	20	150	mA
…	…	…	…

3.3.4 Radiation Requirements

Space radiation environments (e.g., Galactic Cosmic Rays, Solar Flares, Van Allen Belts) may induce unwanted effects on electronic devices and materials. The particles of concern for such radiations are mainly ions, protons, electrons and gamma rays (as secondary products of electrons, generated inside the satellite structure), which can cause effects such as:

- Single Event Effects (SEE).
- Displacement Damage (Total Non-Ionizing Dose).
- Total Ionizing Dose (TID).

The Radiation Source - Particle type – Effects cross table is shown in Figure 1:

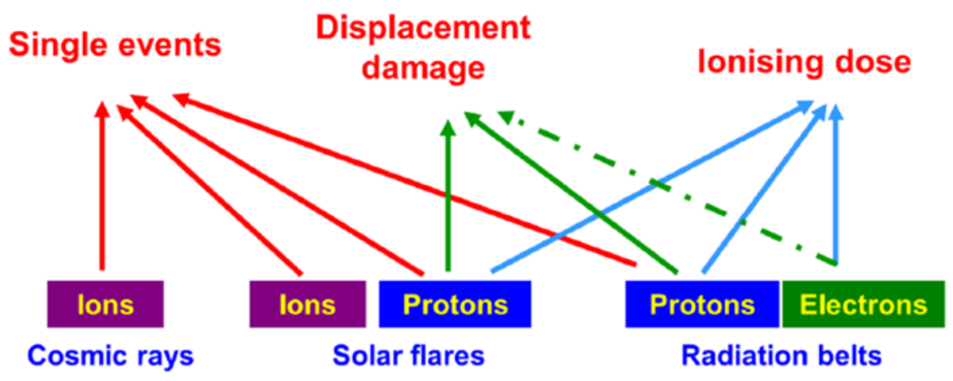

Figure 1: Radiation Source – Particle Type – Effects cross table

SPACE-013	Mission Parameter	Verification: [A]
The chip shall withstand a 12.5-year mission in a LEO orbit environment.		

SPACE-014	Total Ionizing Dose	Verification: [A]
The chip shall withstand a minimum Total Ionizing Dose of 5 krad(si).		

SPACE-015	Destructive Single Event	Verification: [A]
The chip shall withstand a LET $_{\text{threshold}}$ > **15 MeV.cm²/mg**. Destructive Single Event (DSE) free to proton.		

SPACE-016	**Non-Destructive Single Event**	Verification: [A]
The chip shall withstand a LET $_{\text{threshold}}$ > **0.4 MeV.cm²/mg**. Not sensitive to direct ionization by protons.		

SPACE-017	**Radiation Design Margin**	Verification: [A]
The chip shall take into account a Radiation Design Margin, RDM = 1.20 to be applied on number given in requirements for TID and TNID at End-of-Life levels		

SPACE-018	**Radiation Hardness Assurance**	Verification: [A]
The chip design shall take into account the ECSS-Q-ST-60-15C “Radiation Hardness Assurance Requirements for EEE Parts”		

2.4 FPGA Prototype Specifications

In this Section we outline the FPGA specifications chosen for the integration of the space demonstrator in WP5. Building upon the requirements compiled in D1.1 and internal discussions among project partners to further analyse the needs of the demonstrator and solidify its requirements, the VCU118 evaluation board by Xilinx emerged as the selected choice to fulfil the objectives outlined in Task 5.1.

The VCU118 evaluation board is a platform designed to facilitate developers' exploration of Virtex UltraScale+ FPGAs. This board integrates a Virtex UltraScale+ FPGA renowned for its efficient data processing, management of complex algorithms, and smooth communication capabilities. Its versatility renders it well-suited for a wide range of applications, encompassing networks, data centres, and radar systems. The key component of the board is the XCVU9P-L2FLGA2104E FPGA, which hosts critical elements like memory interfaces, high-speed connectors, and user controls. It supports DDR4 and RLDRAM3 interfaces for data storage, and offers various high-speed connectors including QSFP28, PCIe Gen3 x16, FMC+, FMC, and more, ensuring versatile connectivity options.

For comprehensive board specifications, please refer to the documents [6] and [7]. In this section we only provide a few general specifics of the Virtex UltraScale+ VU9P FPGA housed on the board, consistent with the guidelines established in the previous deliverable.

Xilinx Virtex UltraScale+ XCVU9P-L2FLGA2104E

- 2586K system logic cells
- 2364K CLB Flip-Flops
- 1182K CLB LUTs
- 345.9Mb of total on-chip integrated memory
- 6840 DSP Slices
- 21.3 TOP/s Peak INT8 DSP

[6] https://www.xilinx.com/support/documents/boards_and_kits/vcu118/ug1224-vcu118-eval-bd.pdf

[7] https://docs.amd.com/v/u/en-US/ultrascale-plus-fpga-product-selection-guide

- 9x 150G Interlaken and 9x 100G Ethernet with KR4 RS-FEC
- 120 GTY 32.75 Gb/s Transceivers (no 58Gb/s PAM4 Transceivers)
- 832 Max. Single-Ended HD I/Os
- PCI Express endpoint Gen3 x 16

The FPGA has already been purchased and is expected to arrive in about 14 weeks. Once received, E4 will make the board available by providing remote access via ssh to the partners involved in the Space Demonstrator.

2.5 Verification and validation requirements

The verification method of the individual requirements has been described in Section 2.3, based on what defined in the document *ECSS-E-ST-10-02C Rev.1.*

3 Automotive Demonstrator

3.1 Overview of the Demonstrator

Eye detection (see example in Figure 2) is an essential feature for driver monitoring systems acting as a base functionality for other algorithms like attention or drowsiness detection.

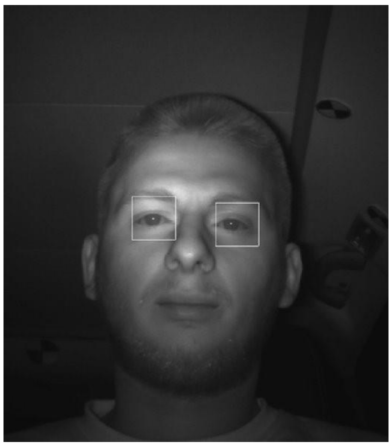
Figure 2: An example of eye detection

Multiple methods for eye detection exist. The ML based methods involve a manual labelling process in order to generate training and testing datasets. This use case presents an eye detection algorithm based on CNNs trained using automatically generated ground truth data and proves that we can train very good ML models using automatically generated labels. Such an approach reduces the effort needed for manual labelling and data preprocessing and it is applicable in image processing.

The algorithm and the relative processing engines are implemented on an FPGA based setup and validated using laboratory and real in-car environment setup.

Algorithms are fed with images from cameras with IR image sensors and illumination and the results will are tracked /analysed on a PC, Intel Core i7-10700, 16 GB, Intel UHD Graphics 630, 512 GB, M.2 PCIe, as depicted in Figure 3 and Figure 4.

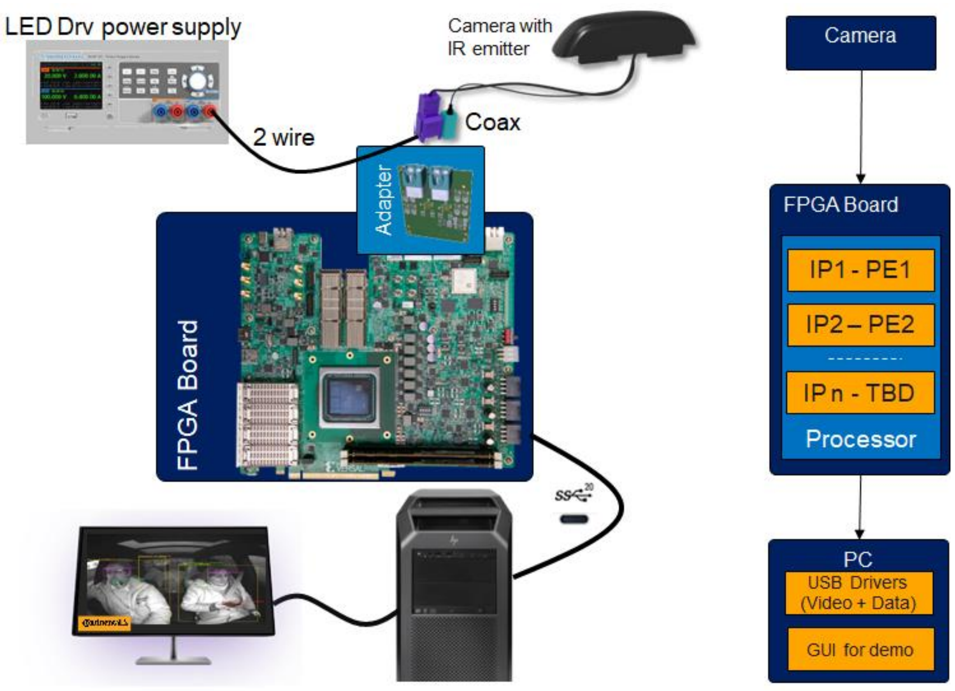

Figure 3: Block diagram of the demonstrator setup

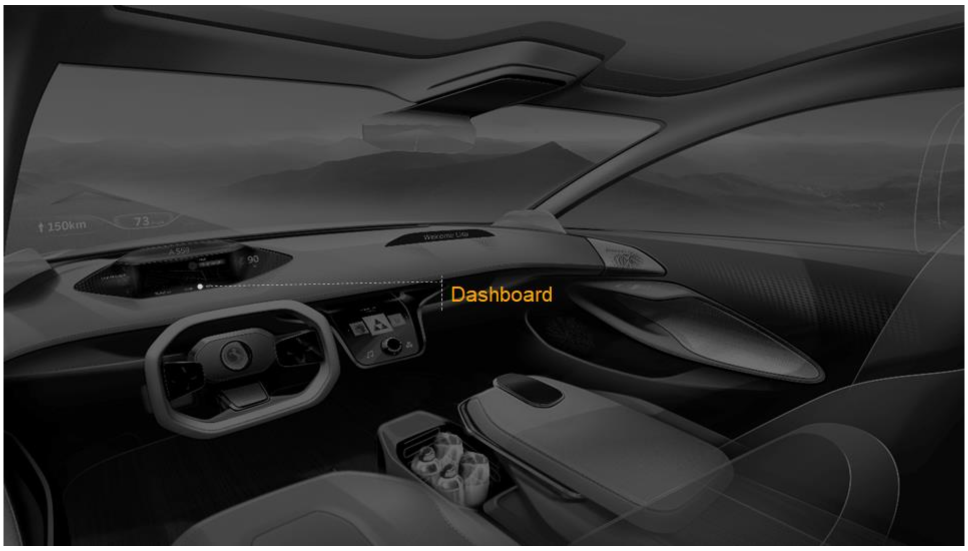

Figure 4: Possible position of the camera in a car

3.1.1 Description of the automotive processor

The automotive application (in our case Driver Monitoring App) can be decomposed in:

- Compute-intensive modules – e.g., machine learning inference, digital signal processing, cryptography.
- Regular modules, i.e., any module which doesn't belong to the above mentioned category (e.g. simple MCUs, ARM Mx (M4) based)

Given such a taxonomy, the compute-intensive modules shall be executed on dedicated HW accelerators, while the regular modules will get executed by general purpose processor (GPP).

This approach leads to a heterogenous HW setup (GPP, HW accelerators, network to interconnect them) which can be a challenge to be programmed.

In our proposal, we hide the HW complexity behind unified RISC-V ISA modules (both standard and custom) while providing a holistic view of the SW part.

This approach also needs a SW toolchain (compiler, assembler, etc.) to translate the application into a mix of custom RISC-V instructions intended for the HW accelerators and RISC-V vanilla instructions. A system level view is depicted in Figure 5.

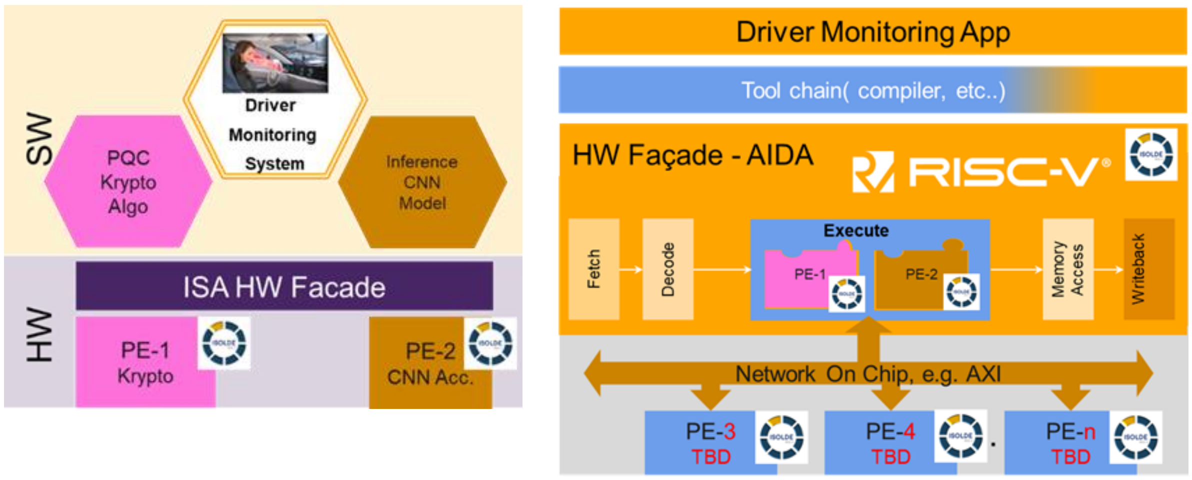

Figure 5: SW toolchain overview

Figure content:

AIDA - Application-specific RISC-V HW Accelerator defines a higher-level ISA which makes the HW accelerators easier to use from the SW perspective

PE = Processing Engines, a.k.a. HW accelerators

PQC = Post Quantum Cryptography

SW toolchains translate ONNX models/C++ applications into binary code to be executed by AIDA

Development effort will be concentrated in:

- ONNX Front-end
- RISC-V backend

while leveraging all the optimizations already available in LLVM expressed graphical in Figure 6.

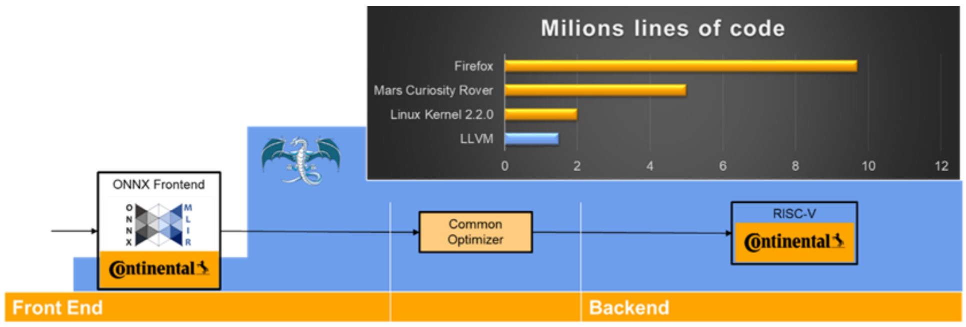

Figure 6: Effort comparison (Code Length) for different implementations

3.1.2 Functionalities and capabilities

To perform eye detection in real-time, a system of multiple CNNs is used. The models are trained using ground truth data generated using a fully automated solution which does not require any type of manual labelling or intervention.

The system contains between 2 and 6 CNN models, each one of them responsible for simplification of the task to output the final location of the eyes.

The first CNN of the system is a regression model responsible for computing an initial location of both eyes (2 bounding boxes).

The last CNN is a classification model that is used combined with a small searching mechanism in order to fix the initial bounding box in the correct position.

Between the previously mentioned ones, other CNN models may be used to reduce and simplify the task of the final classification model. This makes the execution of the entire system faster from a SW point of view together with the HW accelerator.

3.1.3 Key features and benefits

Key benefits for DMS (Driver Monitoring System):

- Best performance/watt for an in-cabin driver monitoring system.
- Exploration for HW solutions for in-cabin sensing.

3.2 Functional and Non-Functional Requirements

3.2.1 Cores

In the HW Facade we aim to modify the core pipeline, therefore we most probably use NOEL-V.

Stretch goal: benchmark the several HW Facade implementations (listed in priority order):

- NOEL-V
- In-house HLS RV32I core (RISC-V core implemented in C++)
- CVA6

3.2.2 Accelerators

We analyse as many as possible the use of the below listed accelerators, provided by the following partners:

- UZL: RISC-V Floating-Point accelerator.
- FotoNation (former XPERI): AI/ML accelerator IP.
- POLIMI: Customizable floating point arithmetic unit for mixed-precision computing. Configurable HW accelerator for the BIKE post-quantum key encapsulation mechanism.
- UNIBO: Tensor Processing Unit.
- POLITO: HW accelerators for parallel processing.
- TUI: vector/SIMD units.
- SAL: Event-Based / Sparse convolution accelerator.
- SAL: Resource efficient PQC.
- BSC: PQC accelerator implementing Classic McEliece, Crystals Dilithium and Crystals Kyber.
- UNSTPB: XNPU.

HM: Bytecode-Interpreter Accelerator for portability and strong isolation

- IMT: SIMD/Vector Accelerator

3.2.3 Software

We use the following compilers:

Procedural compiler:	The LLVM Compiler Infrastructure
ML compiler:	onnx-mlir

3.2.4 Other requirements

N/A

3.3 Measurable metrics

The following metrics are investigated:

- Peak Performance (GOPS/s)
- Peak Power Consumption(W)

3.3.1 Performance Requirements

We aim at throughput: 30fps@1280x720 pixels.

3.3.2 Power Requirements

Power estimation due to running the demonstrator is below 5W.

3.3.3 Thermal Requirements

N/A

3.3.4 Radiation Requirements

N/A

3.3.5 Other Requirements

N/A

3.4 FPGA Prototype Requirements

The Continental team owns Xilinx boards as ZCU102/ZCU104. As a possible enhancement, we target the VMK180. As a backup solution, if resources are exhausted, the VCU118 will be used.

Verilog and System Verilog are the preferred HDL languages; however, for faster HW solution exploration, HLS (C++ and Vitis) represents the main used language and environment.

3.4.1 Memory

Our demonstrator does not require any special memory, as the on-board memory is good enough for our implementations.

3.4.2 I/O

Camera interface: MIPI/CSI.

10 GPIOs for debug (other than JTAG) and control.

High speed USB 3.x to interface the FPGA board to a PC for the evaluation of the algorithm results and of the image recording.

3.5 Verification and validation requirements

- 3.5.1 Requirements traceability

CAR internal development process requires no traceability for this proof of concept.

3.5.2 Verification plan

- CAR internal development process requires no verification plan for this proof of concept.

3.5.3 Validation plan

Validate the results against CUDA implementation running on GPUs.

4 Smart Home Demonstrator

Note: after careful review of requirements of D1.1, only few updates have been considered necessary.

4.1 Overview of the Demonstrator

The goal of the demonstrator is to show the usability of RISC-V application platforms for an SME in the energy sector. Figure 7 describes the setting for smart home energy management demonstrator.

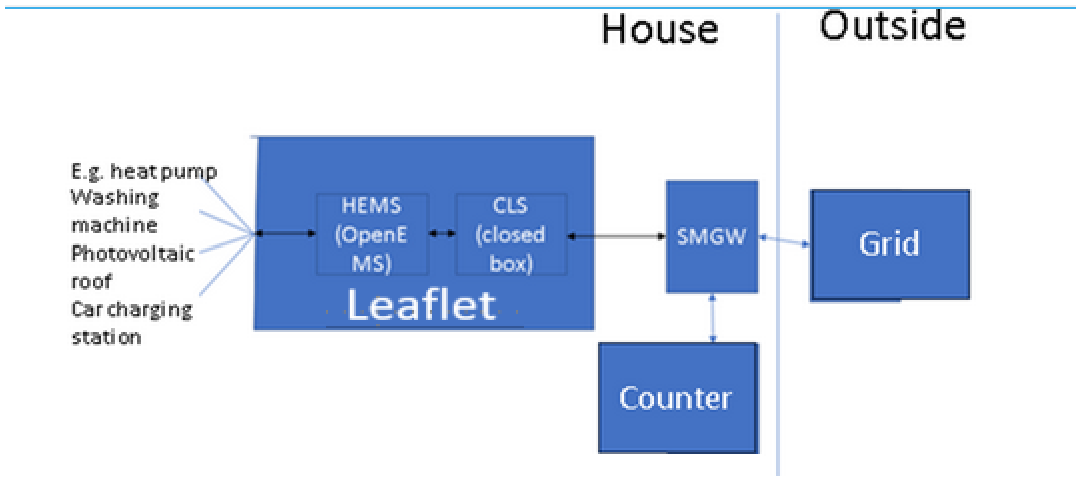

Figure 7: Setup of the demonstrator

The application is an end consumer house network in which various electrical producers (e.g., PV on the roof and/or balcony module, wind turbine, etc.) and consumers (e.g., heat pump, heating rod, charging column for electric car, washing machine, light, etc.) communicate with the Leaflet device (see Figure 7). The goal is to optimize the local use of the generated energy using prognosis, e.g., the charging column battery, operation of the heat pump, are operated when electricity is provided from your own PV system. The decisions / optimizations are made here by a HEMS (Home Energy Management System), in our case based on OpenEMS, which is located on the Leaflet device developed by Consolinno. Since it is not cost-effective to strive for complete self-sufficiency for home networks, as e.g. in winter the PV generation from a small home system e.g. 5-10 kWp in central Europe is usually below power consumption, there is still a connection to the public power grid via the supply system operator. In detail, the house network is separated from the supply system operator by a smart meter gateway (SMGW), which also accesses the counter. The SMGW communicates (e.g., via IEC 61850 protocol) with that of the Leaflet device, which also has a CLS unit (Controllable Local System).

The HEMS unit in Figure 7 operates on the knowledge of all connected devices in the system, while the CLS unit only must know the aggregated value of the generation and consumption in order to communicate with the distribution network operator (e.g., as depicted, by abstracting the whole home energy production and consumption as a single controllable local system interacting with a smart meter gateway via TCP/IP/Ethernet, or (not depicted) e.g. via controllable relays connected to ripple control receiver, e.g. via VDE FNN impulse pre-standard), whether fed or consumed, and in what amount this happens.

4.1.1 Description of the smart home processor

We target three implementation stacks, one on i.MX8 (ARM), two on RISC-V CVA-6, preferably multi-core. Concerning the two RISC-V CVA-6 stacks, one is a pure open source demonstrator that can be made fully available (e.g., on GitHub) as sample payload and the other CVA-6 stack is a demonstrator using a special kind of small code-size hypervisor (separation kernel, https://en.wikipedia.org/wiki/Separation_kernel), which is good at providing strong isolation of execution environments, and controlled execution between them. The separation kernel provided in this use case is called PikeOS (https://www.pikeos.com), and amongst others, has been evaluated against the security standard Common Criteria EAL5.

The three targeted implementation stacks are shown in Figure 8.

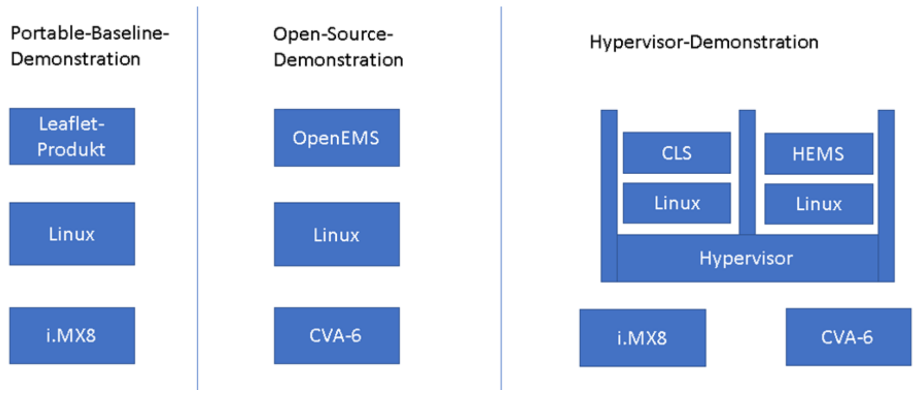

Figure 8: The three implementation stacks covered in this use case

4.1.2 Functionalities and capabilities

CVA-6 shall be used to run a demonstration of the OpenEMS energy management system. CVA-6 shall be able to run mixed-critical systems, such as the PikeOS hypervisor.

We want to run prediction and/or optimization algorithms for energy management, that could be either classical or AI-enhanced (e.g., spiking NNs, we also look at applicability of tools like Apache StreamPipes). We also want to evaluate the use of an additional Python-based SW component.

Extensions to be developed for Apache StreamPipes include lightweight adapters for smart home applications, e.g., KNX and BACNet, and the evaluation of lightweight adapters in RISC-V architectures. These will be implemented in form of an edge client, which will also provide an execution environment for the afore-mentioned ML algorithms.

4.1.3 Key features and benefits

Our demonstrator pursues the following goals:

- Demonstration that a RISC-V platform can be, in principle, used to host the HEMS. The aim here is to be ready as soon as a European RISC V-application processor has reached the critical mass for a tape-out (which does not necessarily happen within the project itself), or to contribute with our use case on an FPGA without tape-out to generate this critical mass (open source demonstration, see Section 5.1.1).
- Demonstration that it is possible for an SME to develop at the same time with target platforms i.MX 8 and RISC-V (portable baseline demonstration, see Section 5.1.1). The problem is to maintain two different sets of expertise (e.g. expertise for building on RISC-V and ARM) at the same time.

- Demonstration that through virtualization it is possible to develop a portable mixed-critical system with possible target platforms i.MX8 and CVA-6 (see Section 5.1.1 hypervisor demonstration). This goal is attractive because it enables device consolidation. This virtualization is motivated to comply with a possible need for a strong separation of zone boundaries, such as the one defined, e.g., in BSI-TR-03109.
- Demonstration that Apache StreamPipes can (pre-)process data in a lightweight RISC-V environment and can collect data using smart home protocols such as BACNet. Demonstration of a lightweight edge client that is able to collect data from resource-constraint hardware devices and forward data to a central analytics server. The demonstrated components are developed within WP4.

4.2 Functional and Non-Functional Requirements

4.2.1 Cores

The demonstrator shall run on a general-purpose CVA-6, ideally multicore.

4.2.2 Accelerators

We plan to evaluate the use of the CEA high-performance L1 data cache. 4.2.3 Software

We want to use a stack consisting of Linux, Java, OpenEMS as well as a version with virtualization consisting of PikeOS, Linux, Java, and OpenEMS. The Apache StreamPipes software stack (fully available as open source) consists of a Java/Linux stack as the primary stack but also includes client libraries for Python and Go.

4.2.4 Other requirements

N.A. In particular, there are no non-functional requirements.

4.3 Measurable metrics

Hereinafter are described the measurable metrics and goals of the smart home demonstrator. Something that we measure is the time-point when functionality (visible by console output of simulated electricity current measurements). This measurement thus measures the boot time up to that point, where the device is operational.

4.3.1 Performance Requirements

The smart home demonstrator shall run on the Genesys2 FPGA development board and shall show the online running of the OpenEMS energy management system.

4.3.2 Power Requirements

Currently not defined.

4.3.3 Thermal Requirements

The demonstrator shall run at room temperature (10°C - 40°C).

4.3.4 Radiation Requirements

No radiation hardness is needed.

4.3.5 Other Requirements

None

4.4 FPGA Prototype Requirements

Hereinafter are described the requirements that will be used to select the FPGA for the integration of the demonstrator in WP5.

4.4.1 FPGA

For single core we use Genesys2 for the implementation of CVA-6. For the multicore, the dual-core Genesys2 seems to be an option, however the analysis of finding an optimal FPGA is still in progress, and a decision will be taken once we have more experience with the multicore setup.

4.4.2 Memory

We envision 512 MB of RAM, 8 GB of non-volatile FLASH memory.

4.4.3 I/O

For the simulation, e.g., via serial interface, we need a console interaction with the system (e.g., Linux terminal).

4.5 Verification and validation requirements

It is planned to validate the demonstrator by producing a prototype on FPGA and to run the scenario of the OpenEMS tutorial scenario of a smart home energy management system with electricity producer(s), consumer(s) and battery.

4.5.1 Requirements traceability

There is no planned special provision for requirements traceability.

4.5.2 Verification plan

There is no verification plan beyond running demonstration scenarios.

4.5.3 Validation plan

It is planned to juxtapose results with regards to the chosen OpenEMS scenarios on the RISC-V platform to functionality of the i.MX8 platform.

5 Cellular IoT Demonstrator

The cellular IoT demonstrator (cIoT) aims to combine 5G cellular IoT standards for wireless communication with embedded HPC to build a modem that can connect to the internet through the cellular network, and at the same time offering enough compute capabilities for embedded applications. Target markets of the proposed system are wearables, industrial monitoring, environmental sensing and monitoring, smart-city, and connectivity in the automotive vertical market.

The HPC part will be tackled with a powerful vector engine attached to a 64-bit RISC-V host processor, while the real-time constraints of cellular communication protocols are addressed with tightly coupled, dedicated accelerators in a communication subsystem. For cIoT (AKA machine-type communications MTC) applications there are the additional requirements of low power (10 years of battery life), low cost (SoC integration) and extended coverage (high sensitivity).

Throughout the project, the partners involved will work towards a 22nm CMOS prototype and demonstrate functionalities on a FPGA platform.

5.1 Overview of the Demonstrator

The demonstrator consists of two main blocks, an embedded HPC subsystem that is handling general high-performance tasks such as image and video processing, and a communication subsystem that is specialized for digital baseband operations and allows for wireless connectivity through the cellular mobile network.

The embedded HPC subsystem consists of a 64-bit host processor, a powerful vector unit, and an IOMMU that will allow to offload communication tasks to the communication subsystem.

The communication subsystem consists of smaller cores that run real-time critical tasks, including SW protocol for modern IoT protocols such as LTE Cat1Bis, the control of a set of accelerators, and the RF-subsystem.

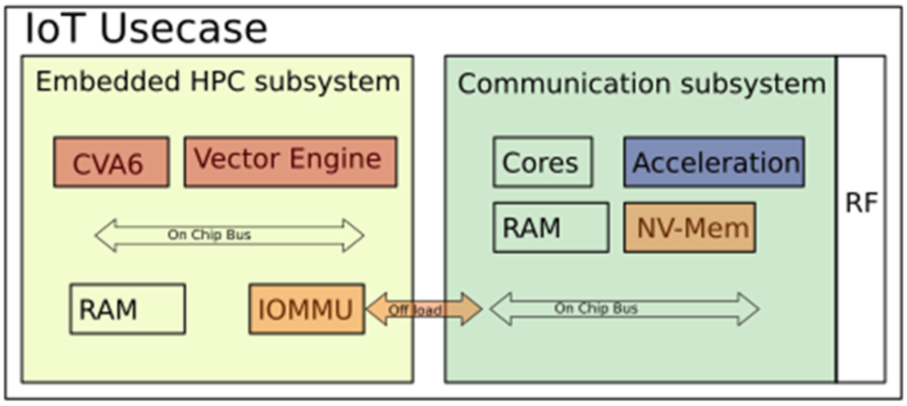

Figure 9: Cellular IoT use case diagram with contributions from WP2 (red), WP3 (blue), and WP5 (orange).

5.1.1 Description of the IoT processor

The processor of the embedded HPC subsystem shall be a 64b core, with a tightly coupled vector engine. Both components are developed as part of the work done in WP2.

On the communication subsystem a set of smaller cores are used to build a multi-core subsystem that allows for concurrent execution of many SW tasks, and control of multiple accelerators.

5.1.2 Functionalities and capabilities

The cIoT demonstrator system has two main functionalities:

- Wireless communication through LTE Cat1Bis, NR
  - The system must be capable of handling all real-time constraints of the SW protocol.
- Scalable computing capabilities to meet performance requirements for wearables, drones, etc.
  - The system must be capable of processing acquired data and transmitting it over the communication subsystem.

5.1.3 Key features and benefits

While the requirements on the communication subsystem are given by the cellular protocol stack, the requirements on the HPC subsystem differ from application to application. Splitting the system in two parts allows to scale the complexity of the HPC system to the applications requirements, and at the same time does not interfere with the real-time requirements of the communication subsystem. To achieve the required throughput, an efficient way of offloading data to the communication subsystem is required.

5.2 Functional and Non-Functional Requirements

5.2.1 Cores

The demonstrator plans to use a 64-bit RISC-V architecture, likely the CVA6 core.

5.2.2 Accelerators

To meet the real-time requirements given by the cellular communication protocol, a set of accelerators, that are developed in WP3, are used. The most performance critical tasks are synchronization, equalization, and channel decoding. The accelerators are connected to a tightly coupled data memory and controlled by a RISC-V multicore system that is running the protocol stack.

5.2.3 Software

The HPC system runs a Linux distribution, and the communication subsystem runs a real-time OS (lk: https://github.com/littlekernel/lk).

5.2.4 Other requirements

Since wearables are one target application of the proposed system, low power consumption is required. In particular, a very low deep sleep power consumption is important to allow for a long-lasting battery. Furthermore, to decrease the cost and size of the IC, while guaranteeing a certain level of security, it is desirable to have non-volatile on chip memory for code storage as well as a file system.

5.3 Measurable metrics

Hereinafter are described the measure metrics and goals of the cIoT demonstrator.

5.3.1 Performance requirements

The throughput requirements of Cat1Bis shall be fulfilled:

- 10mbps downlink
- 5mbps uplink

5.3.2 Power requirements

Deep sleep power consumption shall be in the order of a few uA.

5.3.3 Other requirements

When integrated in an advanced 22nm technology, the size of the die shall not exceed 1cm^2.

5.4 FPGA Prototype Requirements

Hereinafter are described the requirements that have been used to select the FPGA for the integration of the demonstrator in WP5. The preferred board for the cIoT use case is the Xilinx Versal VMK180 for the following reasons:

5.4.1 Vendor / Toolchain Preferences

ACP has been using Xilinx FPGA boards for 10+ years but has no experience with other vendors. Given the experience and success with previous Xilinx boards, it is preferred to stick with Xilinx boards and the Vivado toolchain.

5.4.2 Programming Language Requirements

The demonstrator is built on top of existing components that are written in SystemVerilog and VHDL. While SystemVerilog is superior for system design, VHDL has its advantages for arithmetic operations that are used in accelerators. To be able to reuse as many components as possible, multi-language support is a must.

5.4.3 FPGA Requirements

The communication subsystem requires a large amount of LUTs and Block RAMs to implement all the RISC-V processors cores, accelerators, memories and RF interface. A first estimate shows that roughly 500k-1M system logic cells are required for the communication subsystem while the rest of the system is likely to require another 200-400k.

In terms of BlockRams the communication subsystem will roughly need 4MB of memory to implement all RAMs, caches and buffers.

The cIoT demonstrator aims to integrate non-volatile on-chip memory for code and file storage. This is typically not available on FPGA boards but can be emulated with DDR memory that is not powered down.

5.4.4 Speed Requirements

To achieve the required throughput, and to fulfil all real time requirements, the accelerators of the communication subsystem must run at a minimum of 100MHz.

5.4.5 External Interface Requirements

A set of external interfaces are required. Most notably, 3 GTH transceivers and an FMC connector to interface the RF board. Further, an ethernet adapter, 2 UART interfaces, and a PMOD (or equivalent) are required.

The interface between the two domains must achieve the maximum throughput given by the cellular communication protocol which is 10mbps.

5.4.6 Functionality Requirements

Xilinx MPSoC Development boards come with a processing system (PS), consisting of a multicore ARM core that can run a Linux distribution, and a Programmable Logic (PL) part, that can be programmed according to the needs of the use case. While the PS is not strictly required for the cIoT use case, it offers useful functionalities for debugging, and tracing.

The Xilinx Versal VMK180 board offers all the desired functionalities and will be used for most of the project. However, sub functionalities can also be demonstrated on smaller, cheaper boards such as the Genesys2 board.

5.5 Verification and validation requirements

5.5.1 Requirements traceability

Currently no requirements for traceability are defined.

5.5.2 Verification plan

Individual components are first verified in RTL simulation with dedicated testbenches and then combined in a system. A set of integration tests are performed to make sure that all components are correctly integrated.

5.5.3 Validation plan

The cIoT demonstrator is validated in four main steps:

- The cellular communication is validated on the Versal VM180K board with the help of protocol tester HW to guarantee that the cellular protocol stack is correctly implemented.
- The performance requirements of the HPC cluster are validated.
- The throughput requirements between the communication and HPC cluster are validated using two FPGAs.
- The final validation is a successful connection to the cellular network, and a throughput measurement during uplink and downlink data transmissions to verify that the required throughput can be reached.

6 Conclusions

The Consolidated Demonstrators Requirements and Specifications deliverable represents a collaborative effort among partners in the ISOLDE project, working across different WPs. This document combines the collective expertise and perspectives of all involved partners, providing a comprehensive set of requirements and specifications for the demonstrators of the project. In particular, the final requirements for the demonstrators in the four different application domains have been collected in this deliverable, as a refined and consolidated version of the ones previously presented within deliverable D1.1.

This document collects key information from demonstrators' characteristics to their verification, working as a relevant input for the subsequent phases of the ISOLDE project, in particular within WP5 activities, where it is foreseen the realization of the demonstrators. D1.3 also marks the conclusion of the activities of Task 1.1.