KU LEUVEN

Trusted IoT - User group meeting
June 9th, 2023

Nele Mentens, Masoom Rabbani, Jo Vliegen

ES&S, imec-COSIC, ESAT, KU Leuven, Leuven, Belgium

WP2: Exploration, Specification & Requirements study

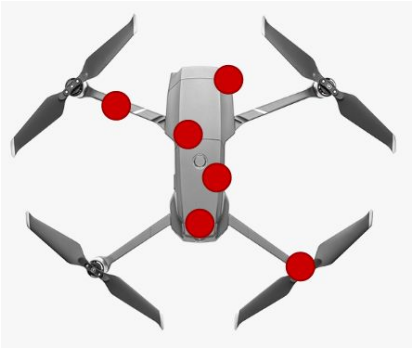
KU Leuven will have a deeper look into the requirements that come with a **unmanned aerial vehicle** (drone). Typically there are **multiple devices** that each serve a specific function. Some components are standalone hardware units and do not have any processor system. Other components are to be vetted so they can be mapped on multiple RISC-V processors. Subsequently an exploration must be done of **existing, open-source implementations of RISC-V**. As new implementations are regularly made available, choosing the most appropriate implementation for the targeted use case will also be a part in this WP.

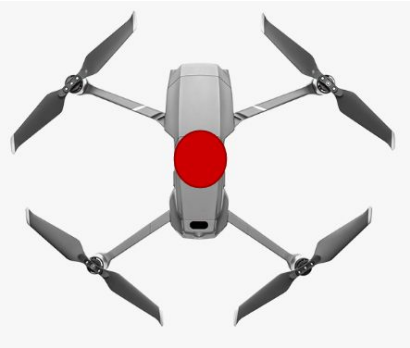
Trusted IoT - WP2

Within Trusted IoT - KU Leuven is working on Multi-Core RISC-V platforms

The industrial Use Case will focus on drones, operated by multiple RISC-V cores.

Multiple, isolated microprocessors will be **centralised** on a single **FPGA**



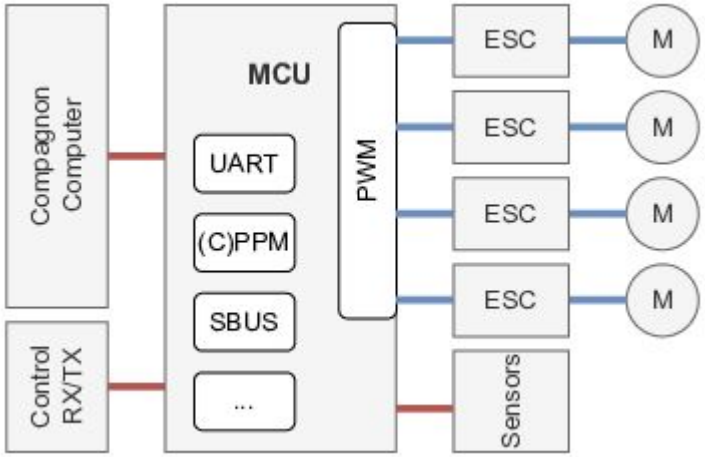


Trusted IoT - WP2

Within Trusted IoT - KU Leuven is working on Multi-Core RISC-V platforms

The industrial Use Case will focus on drones, operated by multiple RISC-V cores.

Three different RISC-V implementations will be used

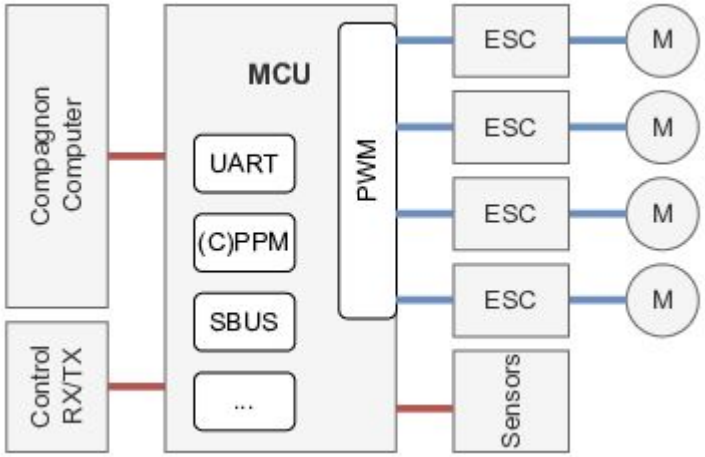


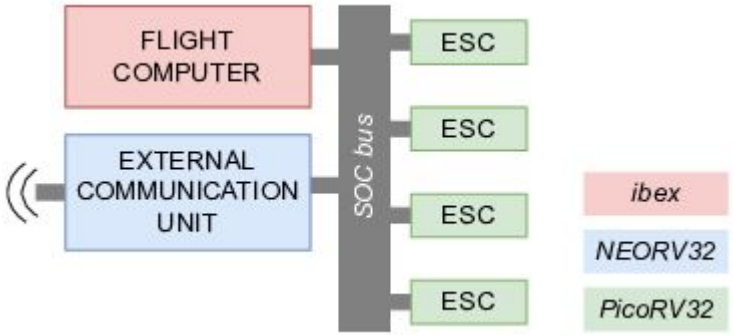
Trusted IoT - WP2

Within Trusted IoT - KU Leuven is working on Multi-Core RISC-V platforms

The industrial Use Case will focus on drones, operated by multiple RISC-V cores.

Three different RISC-V implementations will be used





Trusted IoT - WP2

Three different RISC-V implementations will be used

ibex

- **source**: https://github.com/lowRISC/ibex
- **intended use**: MCU / Flight computer
- **licence**: Apache License Version 2.0
- **HDL**: SystemVerilog

NEORV32

- **source**: https://github.com/stnolting/neorv32
- **intended use**: Comm
- **licence**: 3-clause BSD
- **HDL**: Verilog

PicoRV32

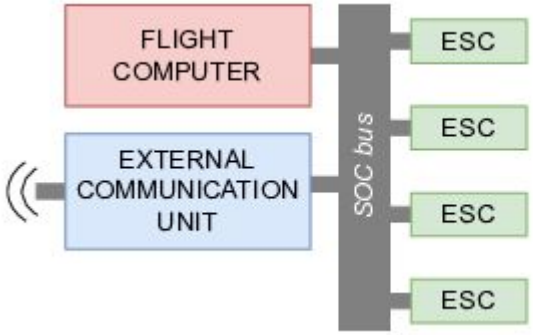
- **source**: https://github.com/YosysHQ/picorv32
- **intended use**: ESC
- **licence**: ISC
- **HDL**: VHDL

PicoRV32 - A Size-Optimized RISC-V CPU

Trusted IoT - WP2

Three different RISC-V implementations will be used

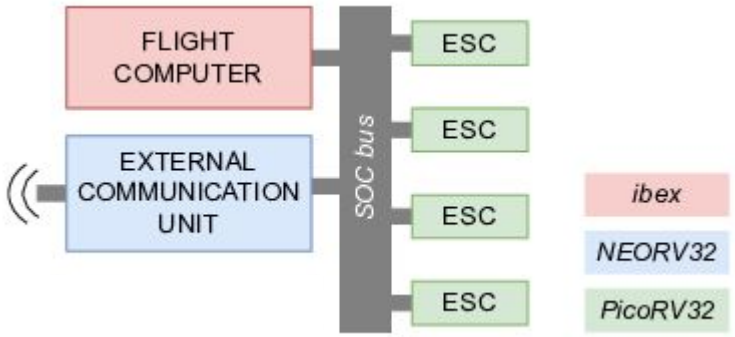
		ibex Apache License 2.0	NEORV32 BSD 3-clause	PicoRV32 ISC
cannot	Hold Liable	x	x	x
	Use Trademark	x	x	
must	Include copyright	x	x	x
	Include license	x	x	x
	State changes	x		
	Include notice	x		
can	Commercial use	x	x	x
	Modify	x	x	x
	Distribute	x	x	x
	Place warranty	x	x	
	Private use	x		
	Use patent claims	x		
	Sublicense	x		

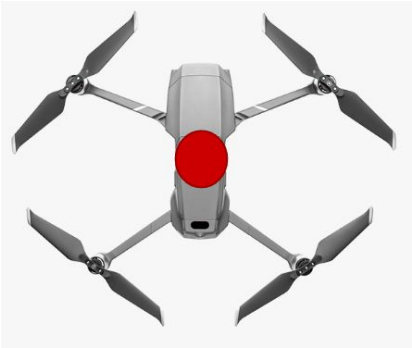


Trusted IoT - WP2

Benefits:

- improved inter-component communication
  - choice of protocols
  - throughput
  - security (authentication & confidentiality)
- compatibility with COTS
- (remote) attestation of ALL components

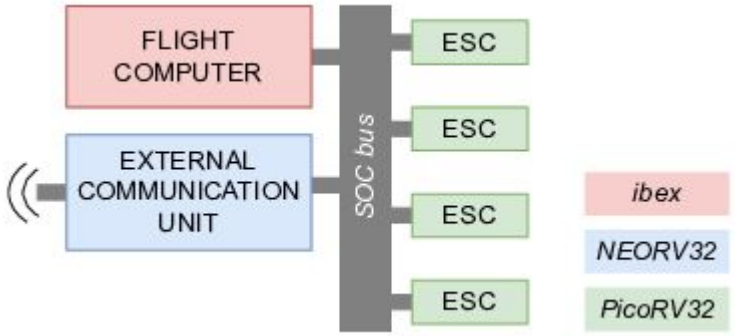


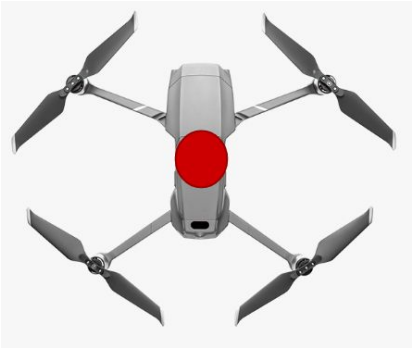


Trusted IoT - WP2

Finished

- final report is available on the website



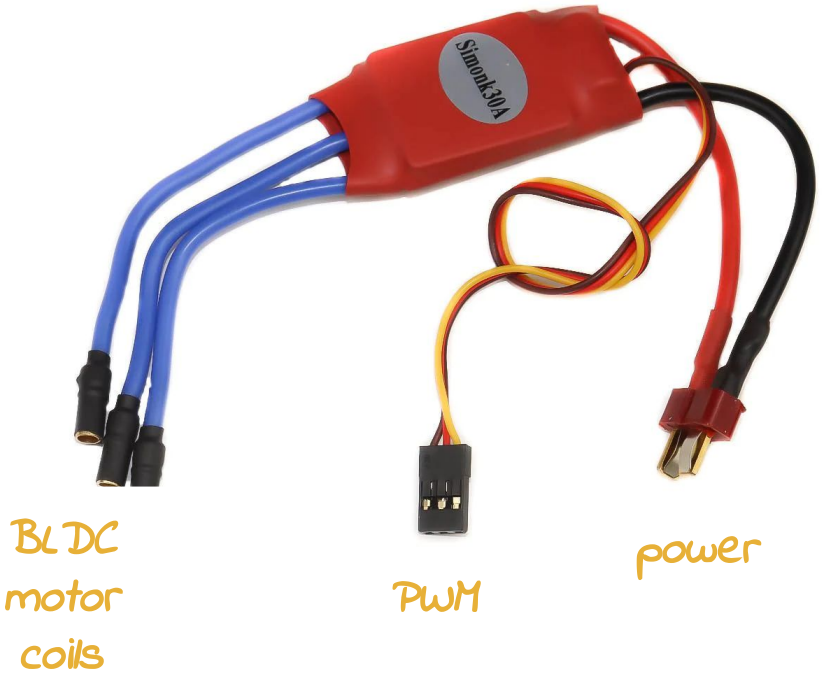


WP3: Platform-specific security solutions

T 3.3 (KU Leuven) **Multi-Core RISC-V** platforms. RISC-V is an open standard that implements the principle of a Reduced Instruction Set Computer (RISC), which comes down to the actual processor on which a system is running. A RISC-V core can be adjusted and/or extended to better fit the application it is hosting. With the fact that more and more different 'processors' are available comes the need to have some form of interaction. This, however, could pose a threat. As the weakest component could succumb to attackers, it might infect other components as well. **Having multiple cores should have a mechanism that they keep an eye on each other.** A typical approach is to provide some **trusted hardware** to each entity (processor) so they are equipped for overcoming this challenge. Having a reconfigurable processor makes this feasible. The results of the examining and comparing the state-of-the-art techniques and implementations and holding them against the established requirements (as done in WP2) will **set out the lines for a proof-of-concept implementation**. This will then be translated to suit the needs for the target use-case in WP4.

Trusted IoT - WP3

Started with ESC



There is a processor in the ESC that:

- receives PWM
- transmits coil-steering pattern to the BLDC
- a small **microcontroller** manages

There is "*quite some*" power electronics

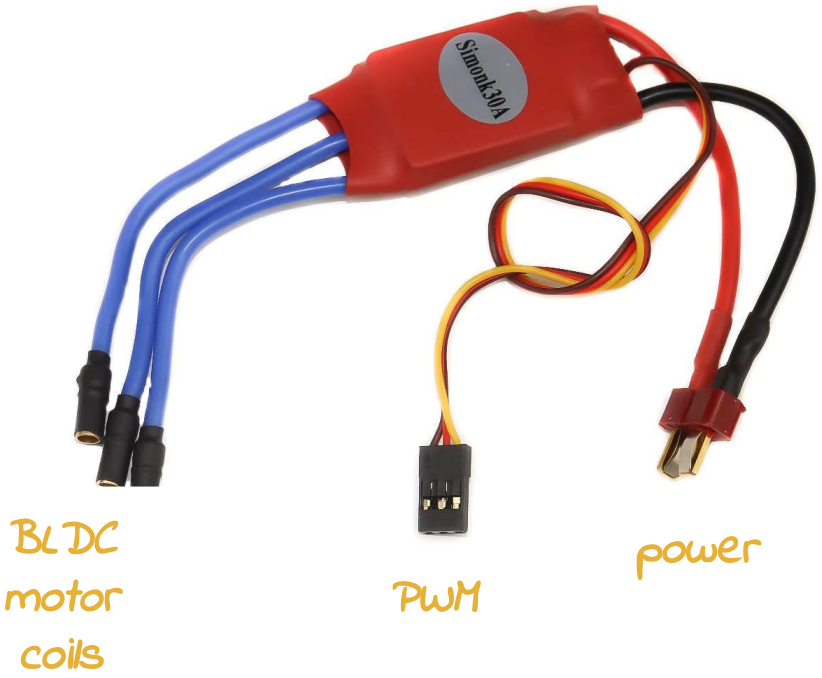
We want to get around using the microcontroller

(and we're stuck with the power electronics)

(curse you, back EMF)

Trusted IoT - WP3

Started with ESC



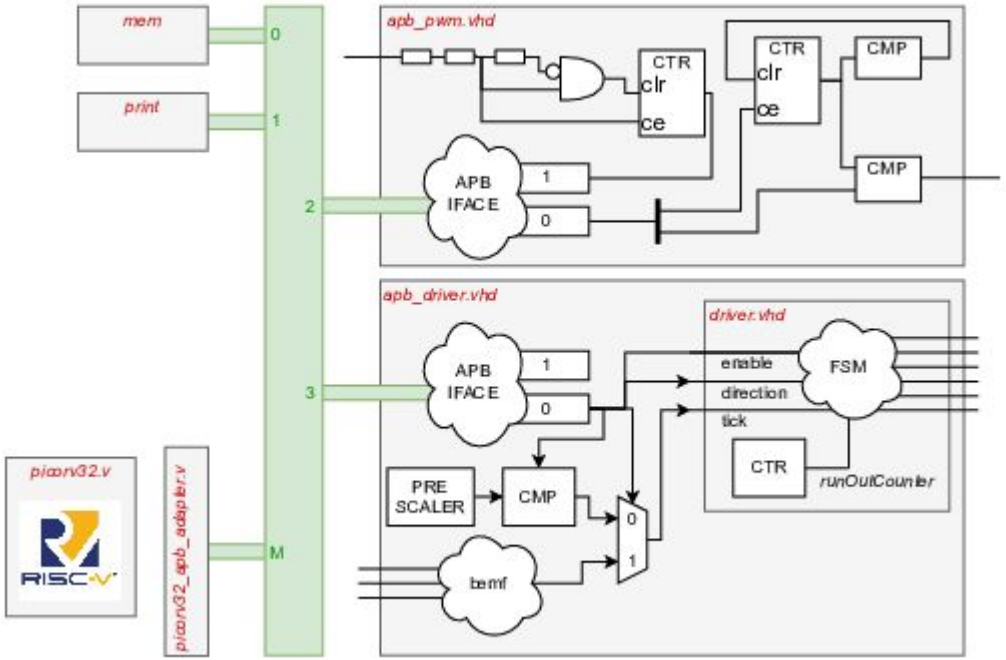
Drone BLDCs are bought

Power electronics has been made

A(n extremely) simple SOC replaces the microcontroller

Trusted IoT - WP3

Started with ESC



Drone BLDCs are bought

Power electronics has been made

A(n extremely simple) SOC replaces the microcontroller

AXI4-Stream(-like)

“BLDC motor coils”

Trusted IoT - WP3

Started with COMM

There is a processor in the COMM that:

- sends and receives RF signals
- transmits and receives instructions over PWM
  - can be PWM
  - can be (C)PPM
  - can be SBUS
- a small **microcontroller** manages

We want to get around using the microcontroller

Trusted IoT - WP3

Started with COMM

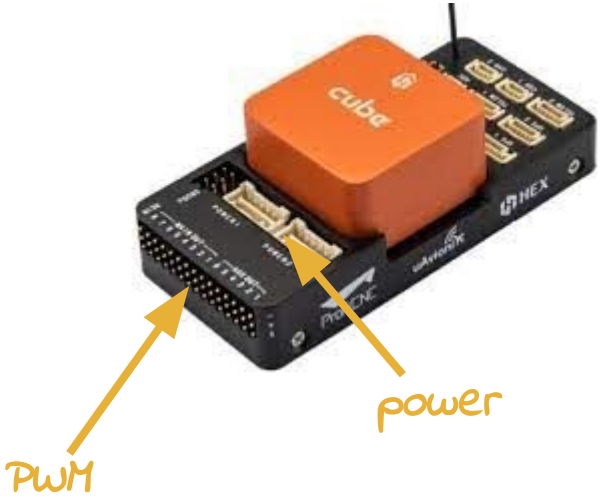
UART

AXI4Stream-like

Trusted IoT - WP3

NOT started with FC (yet)

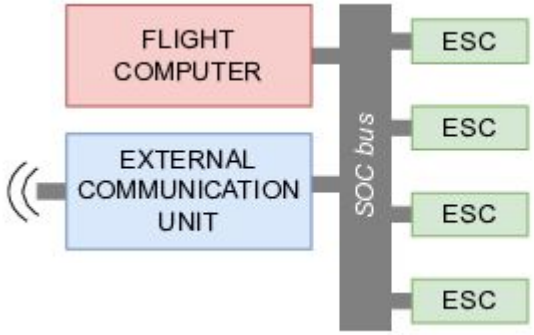
We want to get around using the microcontroller



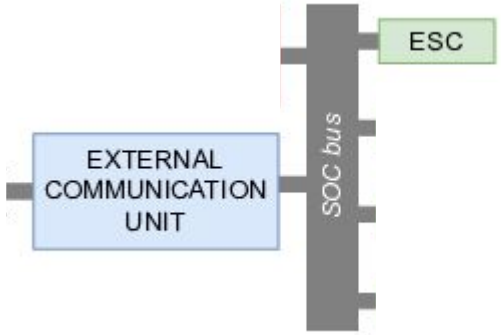
STILL TO come

Trusted IoT - WP3

Targeted implementation



Current state of implementation



This implementation replaces/represents the “main application”

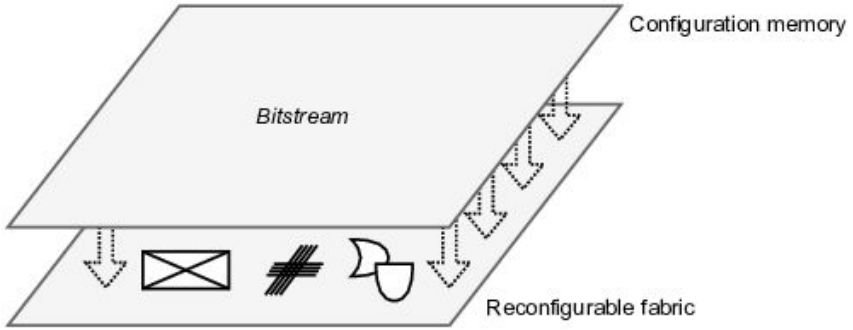
Trusted IoT - WP3

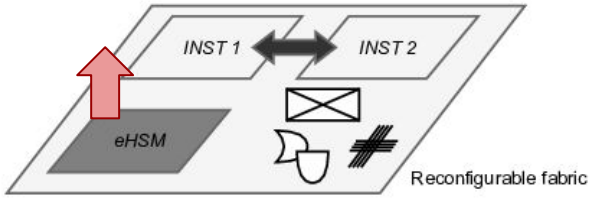
eHSM

The main application needs to be attested

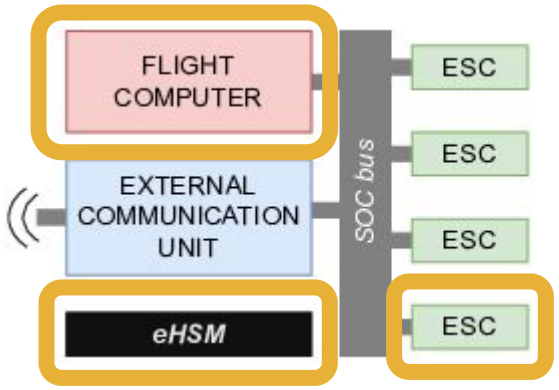
Because the main hackable components are on the FPGA, attestation of the entire FPGA is required.

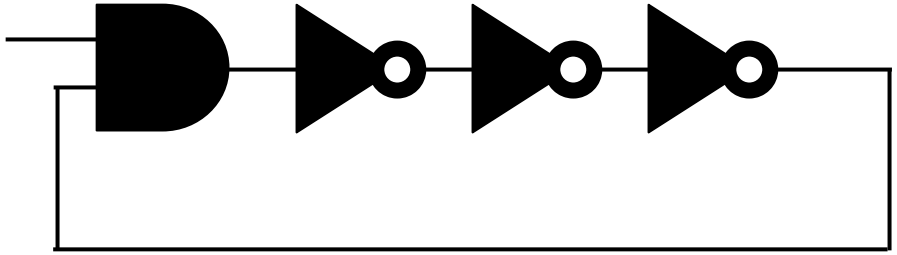
Secure communication through LWC winner: ASCON





Trusted IoT - WP3





We aim for minimal refactoring to protect industries already available solutions

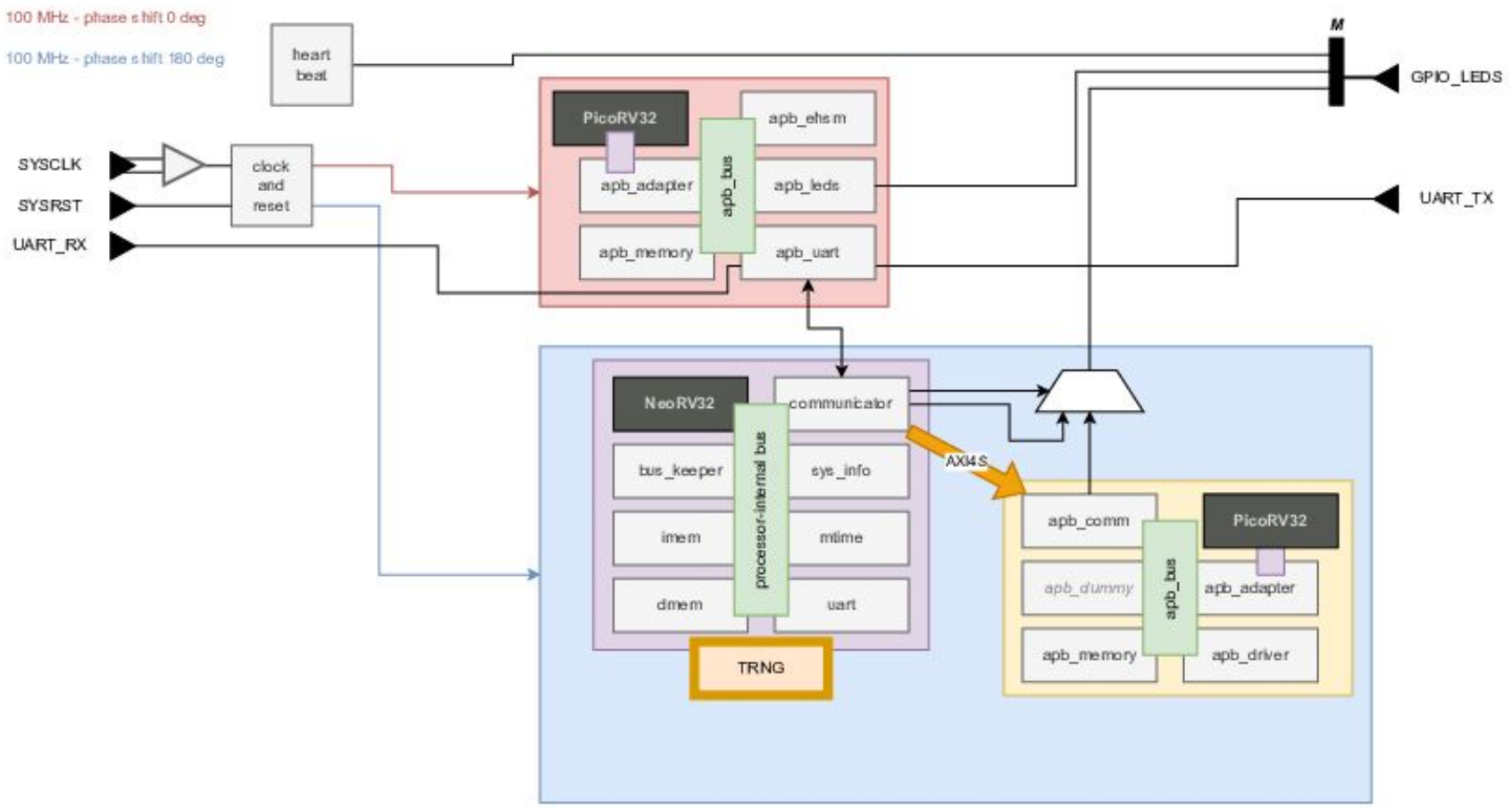
Different designs of different companies are present on the same chip

Multi-tenancy

Protection against SCA: active fence

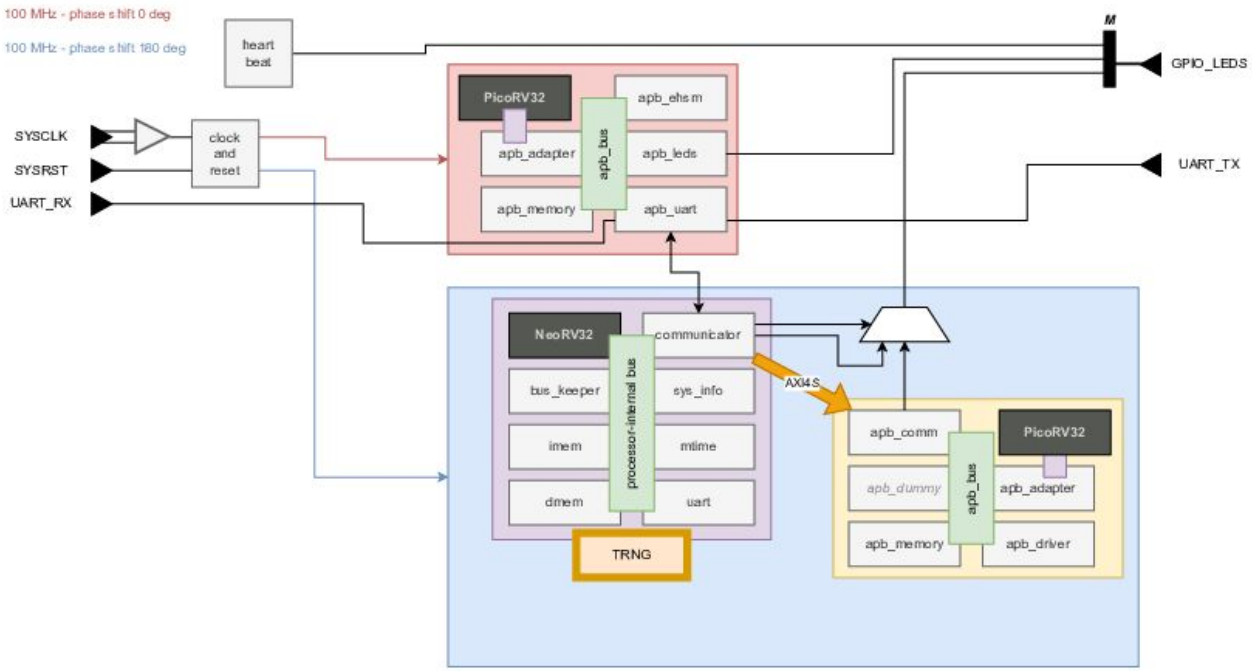
Trusted IoT - WP3

Current state of implementation



Trusted IoT - WP3

Current state of implementation



We want to get around using the microcontroller

Different clock domains for eHSM and application

Shared UART (for now)

TRNG is aimed to be dual use:

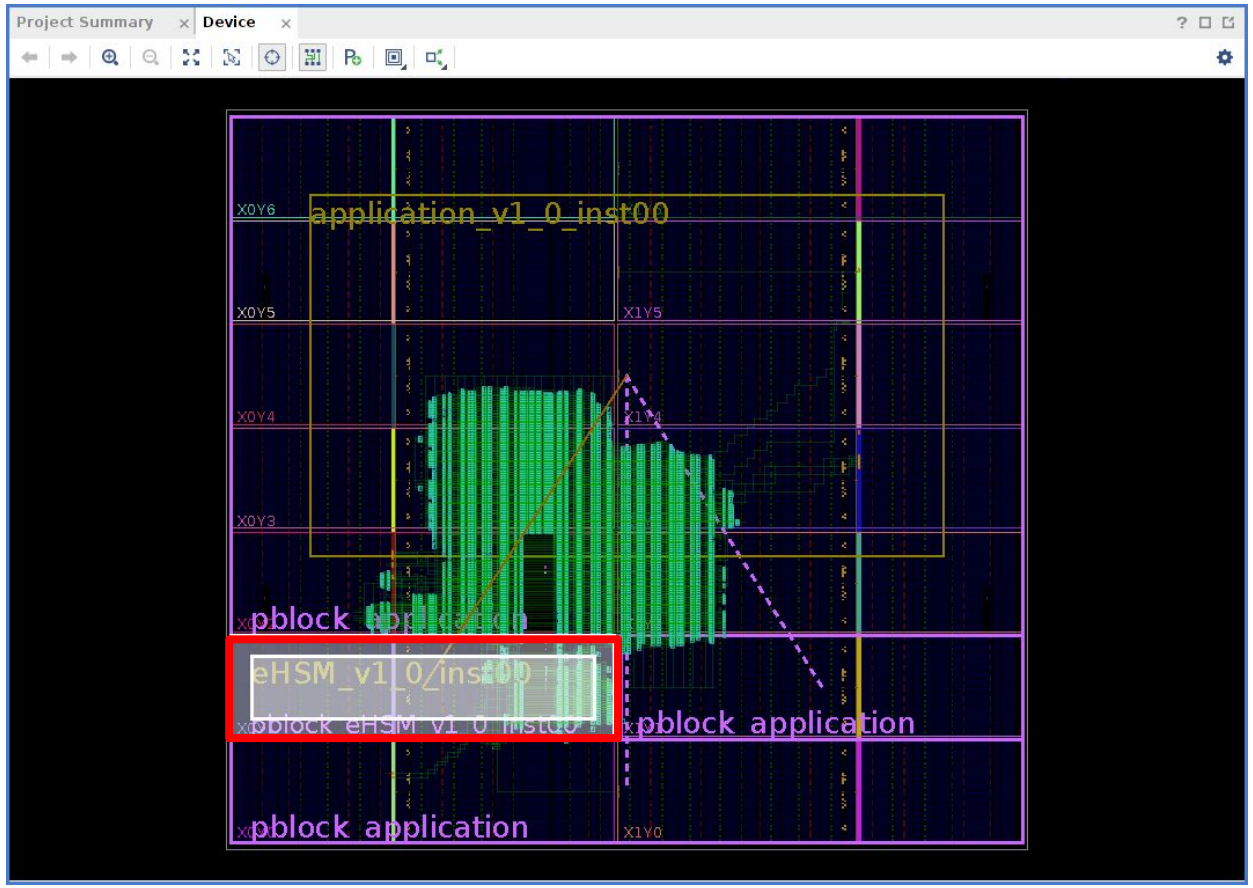
- as TRNG
- as fence

Trusted IoT - WP3

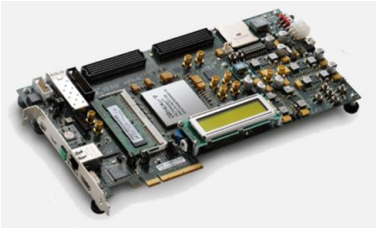
- 3 out of 4 RISC-V implementations have been tested standalone:
  - eHSM
  - ESC
  - communication

- a **merge** was made
  - partitioned implementation
  - ▭: active fence

- due to practicality reasons:
  - only 1 shared UART for all
  - (ideally: 1 for eHSM, 1 for application)



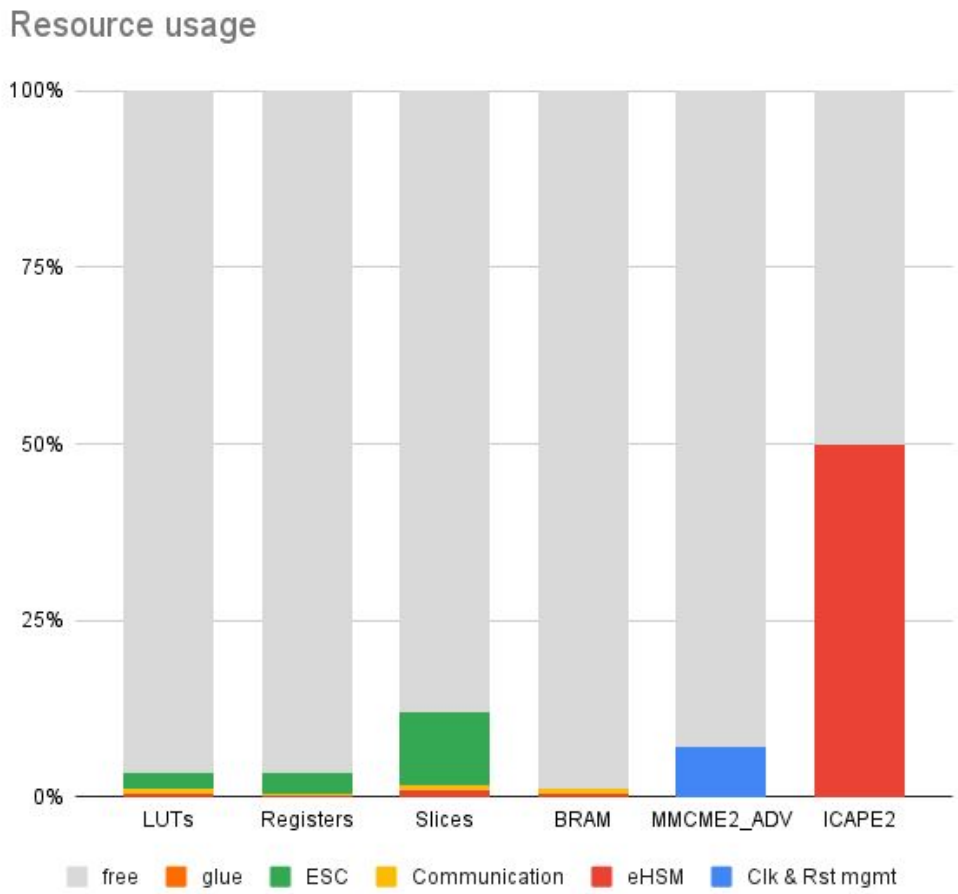
Trusted IoT - WP3



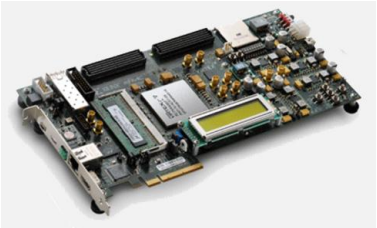
Resource usage

- **NO** FC
- **ONE** single ESC
- **NO** encryption
- communication only over (shared) **UART**

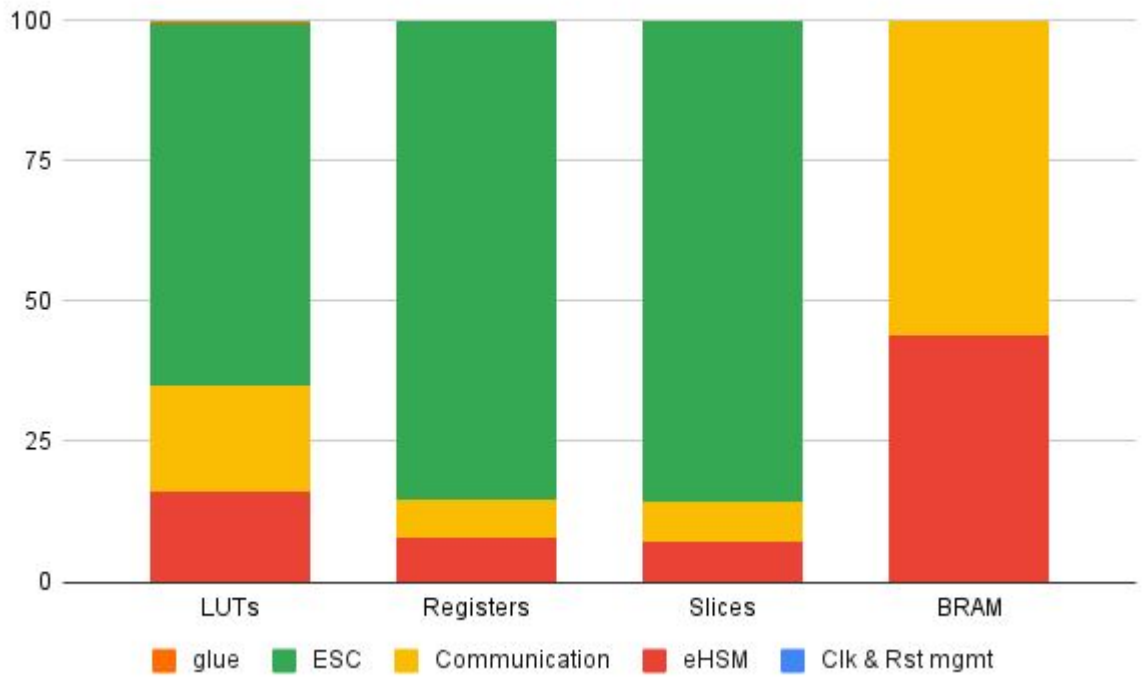
		Slices	Regs	BRAM
Application		1’198	2’788	9
	comm	635	1’867	7
	ESC	566	1’329	2
eHSM		606	1’524	5
Total (avail)		75’900	607’200	1’020

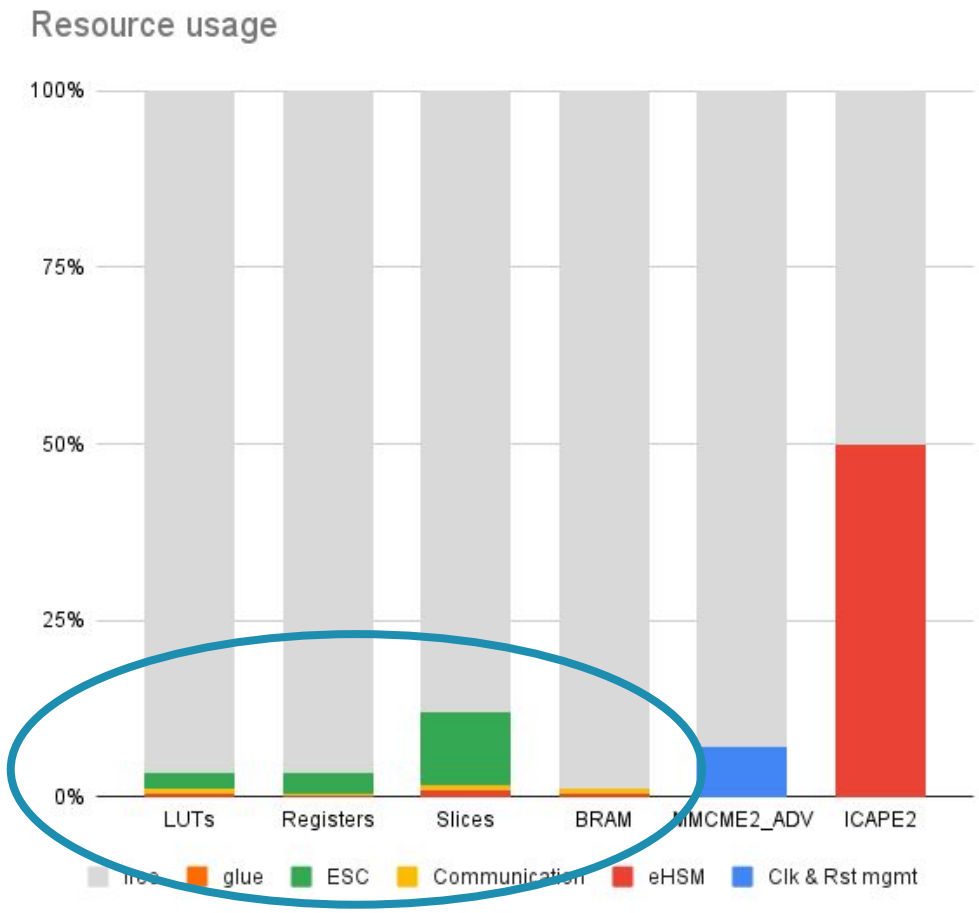


Trusted IoT - WP3

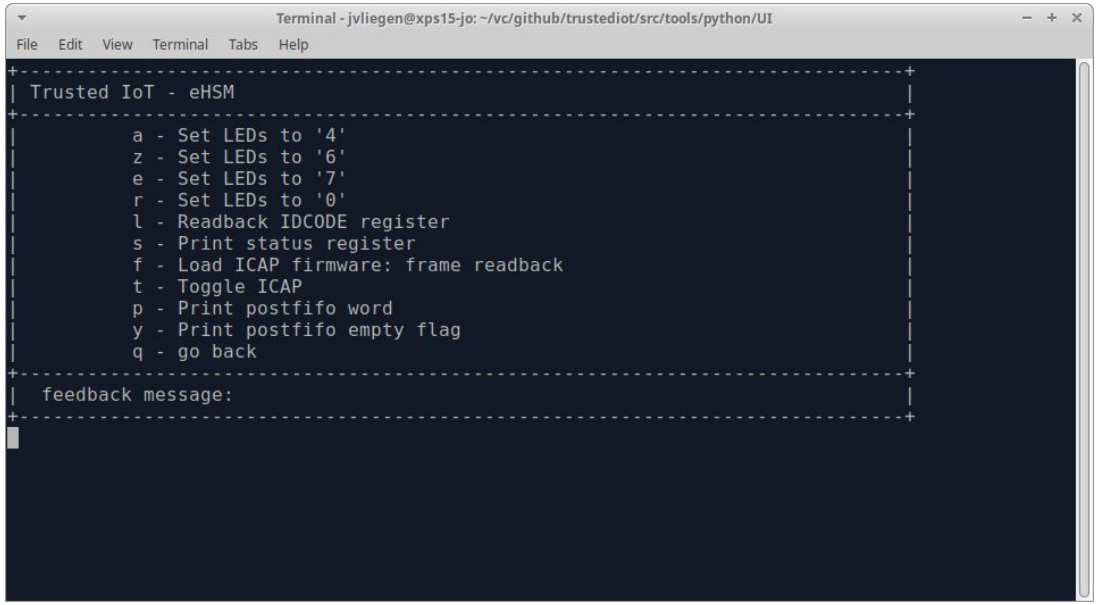


Resource usage





Trusted IoT - WP3



first UI with eHSM up-and-running

readback capabilities are currently being tested

Trusted IoT - demonstrator

Working (to certain extend)

- Application
  - single BLDC spins
  - communication is alive
- eHSM
  - implemented crypto is finished but needs to be merged
  - ICAP readback provides IDCODE

Still needs attention

- Application
  - adding the FC
  - making a small PCB for power electronics
  - repeat motor 3 more times
- eHSM
  -

?

thank you !!