KU LEUVEN

Trusted IoT - Closing meeting
August 29th, 2024

Nele Mentens, Masoom Rabbani, *Jo Vliegen*, and Sem Kirkels

Industrial use case

Attestation of a multi-core RISC-V platform

Trusted IoT

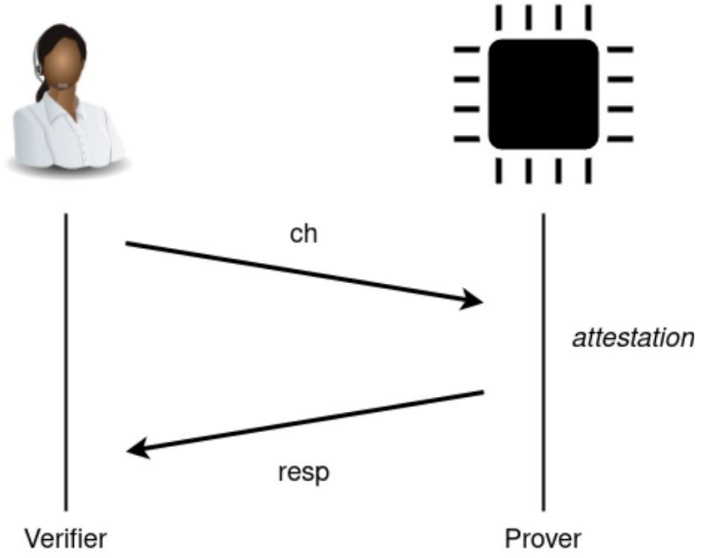
Many consumer products, today, consist of multiple IoT devices.

To verify that the product is still behaving as it is intended to, an ***attestation*** can be performed.

Trusted IoT

Many consumer products, today, consist of multiple IoT devices.

To verify that the product is still behaving as it is intended to, an ***attestation*** can be performed.



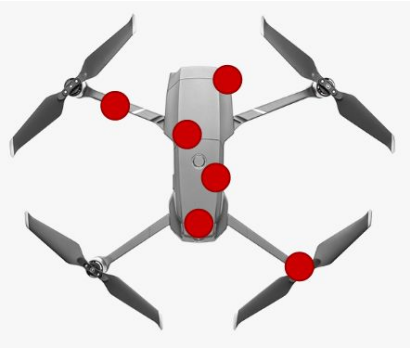
During an attestation protocol, the **Verifier** checks the health of the **Prover**.

The firmware that is running on the IoT device is verified this way.

Trusted IoT

Many consumer products, today, consist of multiple IoT devices.

To verify that the product is still behaving as it is intended to, an ***attestation*** can be performed.



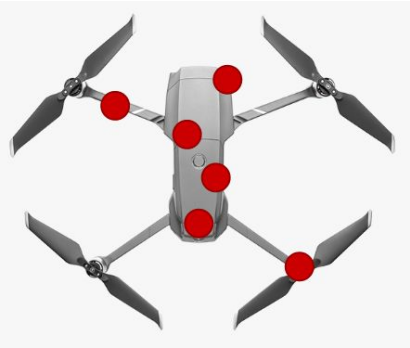
To have more complete attestation, a protocol has to be run with every single IoT device.

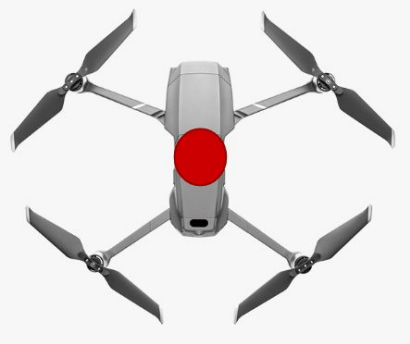
This doesn't scale !!

Trusted IoT

Many consumer products, today, consist of multiple IoT devices.

To verify that the product is still behaving as it is intended to, an ***attestation*** can be performed.





To have more complete attestation, a protocol has to be run with every single IoT device.

This doesn't scale !!

In this industrial use case a solution will be proposed by:

replacing all processors with a soft-core implementation on FPGA

Industrial use case

FPGA refresher

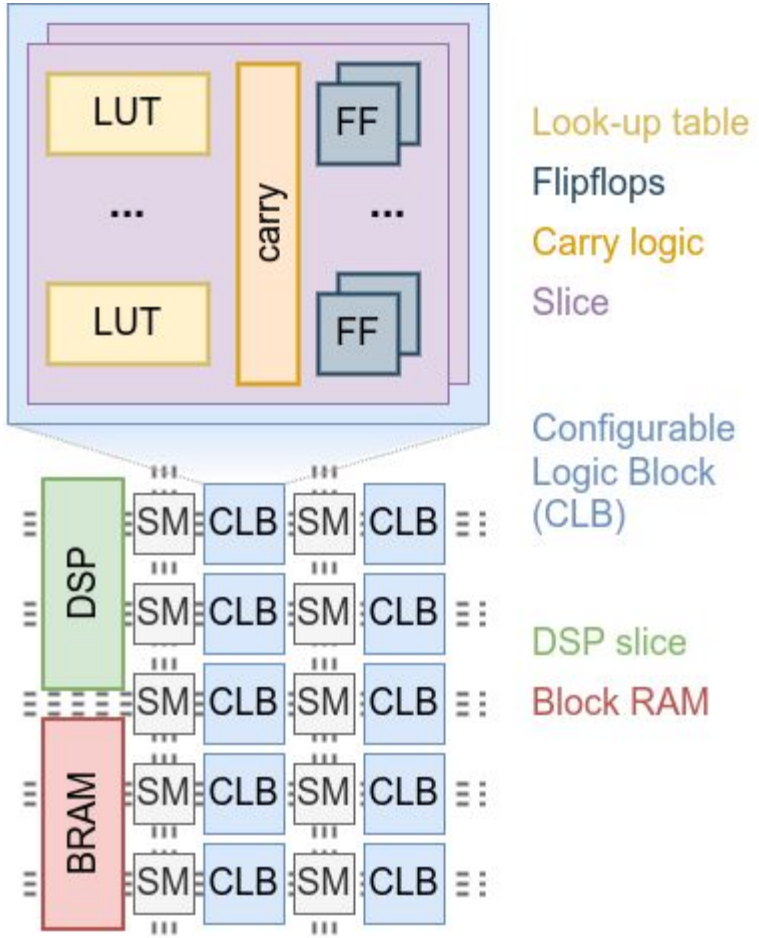
Trusted IoT

A small refresher on the FPGA and its configuration*.

An FPGA is a reconfigurable device that consists of predefined components, like:

- Look-up tables
- Flip-flops
- Carry logic
- Switch matrices
- Block RAMs
- DSP-slices
- …

These parameterisable components are *'always'* present on the FPGA



** the used terminology is that of AMD / Xilinx*

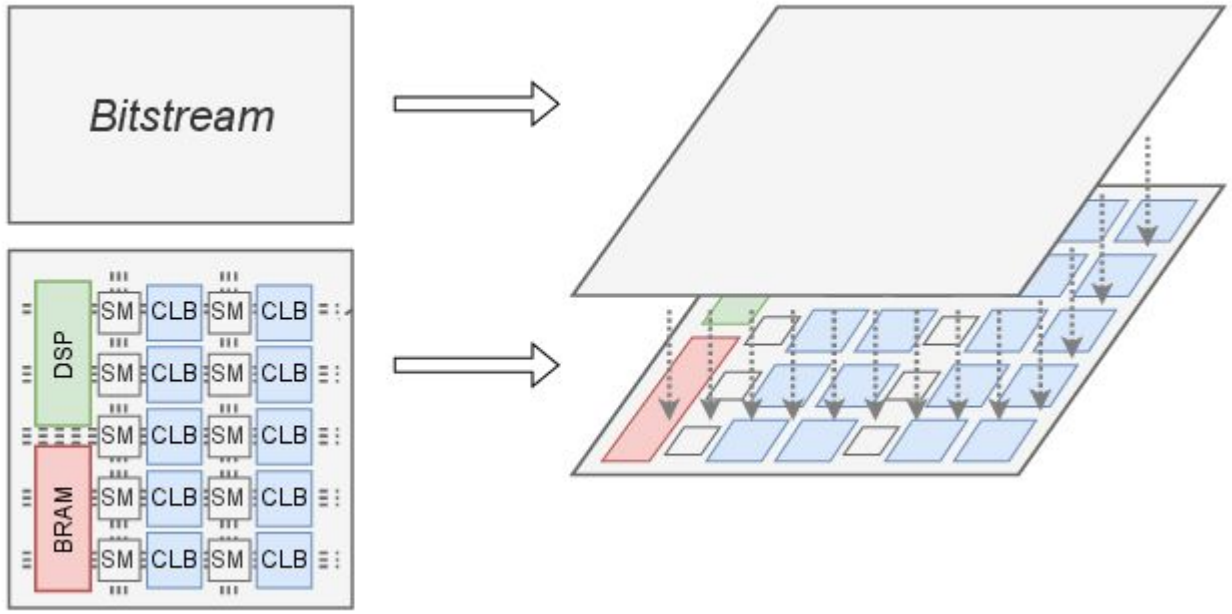
Trusted IoT

A small refresher on the FPGA and its configuration*.

These parameterisable components are configured in a (typically volatile) memory.

The configuration memory sets the behaviour of the reconfigurable fabric when the FPGA is power-cycled.

In a typical setting, the information flows in one direction: from configuration memory to reconfigurable fabric.



* the used terminology is that of AMD / Xilinx

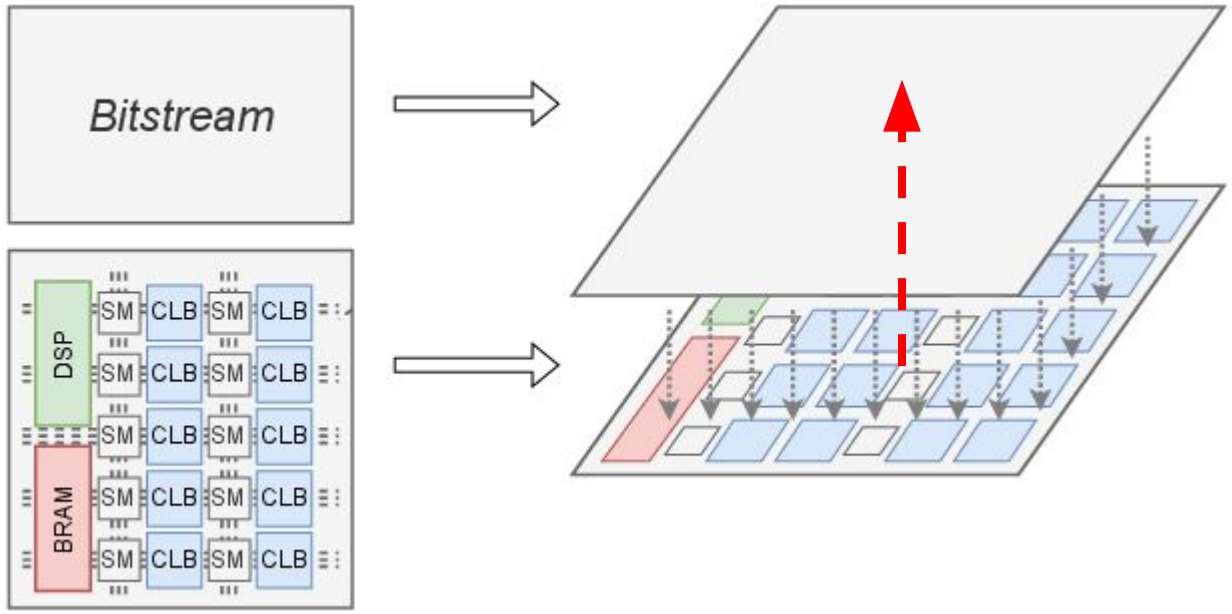
Trusted IoT

A small refresher on the FPGA and its configuration*.

These parameterisable components are configured in a (typically volatile) memory.

The configuration memory sets the behaviour of the reconfigurable fabric when the FPGA is power-cycled.

In a typical setting, the information flows in one direction: from configuration memory to reconfigurable fabric.



However, there is a component that allows information flow in the other direction:

the Internal Configuration Access Port (ICAP)

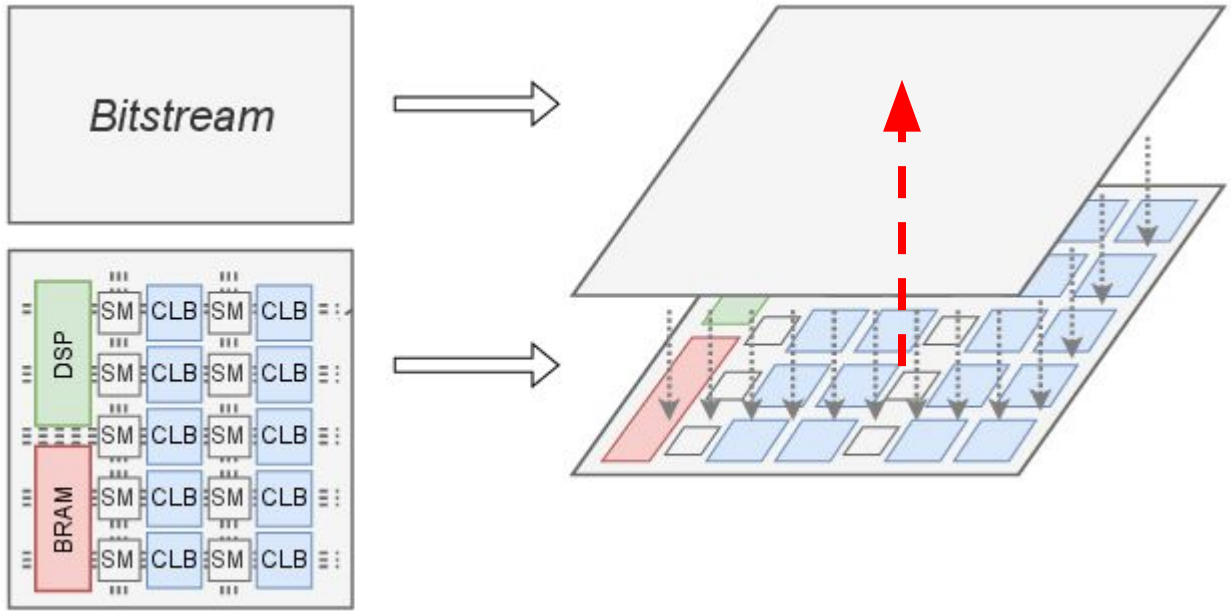
** the used terminology is that of AMD / Xilinx*

Trusted IoT

A small refresher on the FPGA and its configuration*.

Through the ICAP the entire configuration memory can be read back.

This means that the hardware configuration and the memory content of the reconfigurable fabric can be attested.



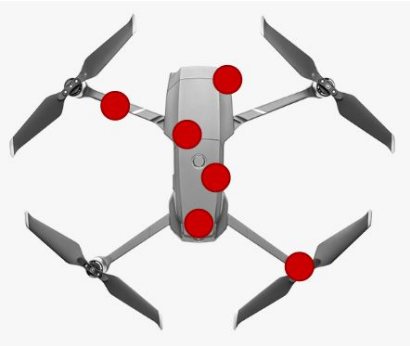
** the used terminology is that of AMD / Xilinx*

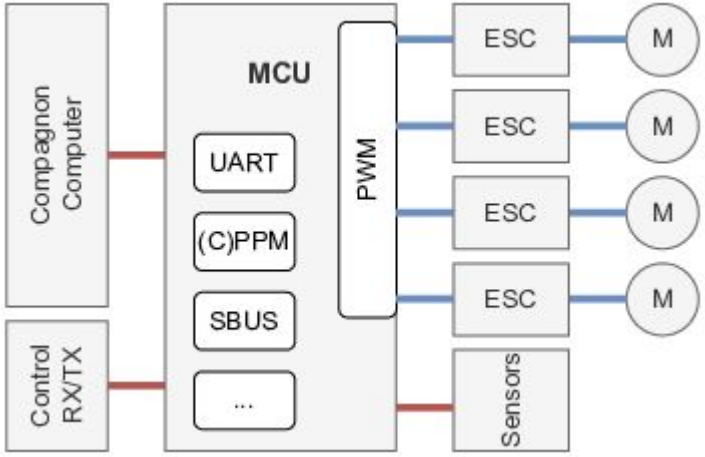
Industrial use case

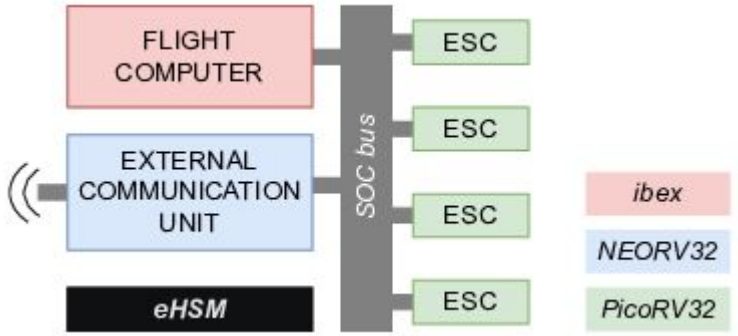
On the drone

Trusted IoT

Within Trusted IoT, KU Leuven has worked on Multi-Core RISC-V platforms. The industrial use case focused on drones, operated by multiple RISC-V cores.





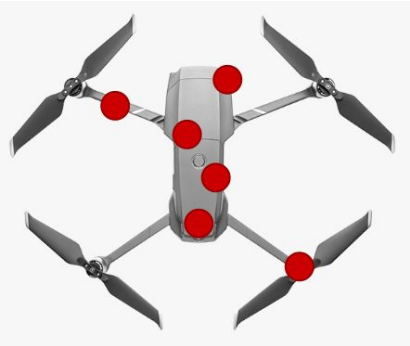


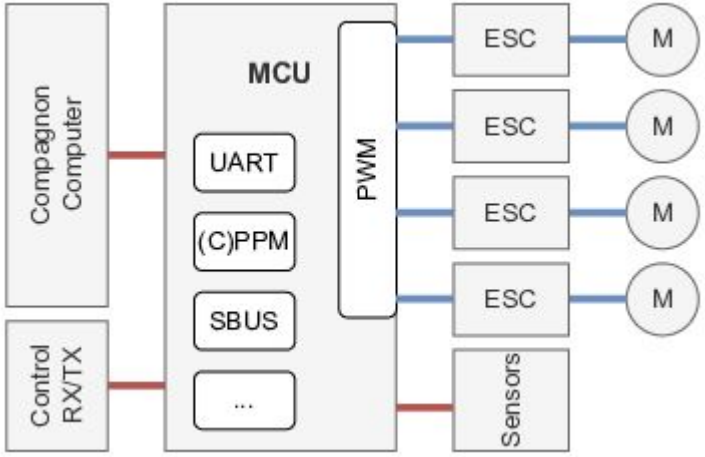
To replace the processors, three existing RISC-V implementations were selected.

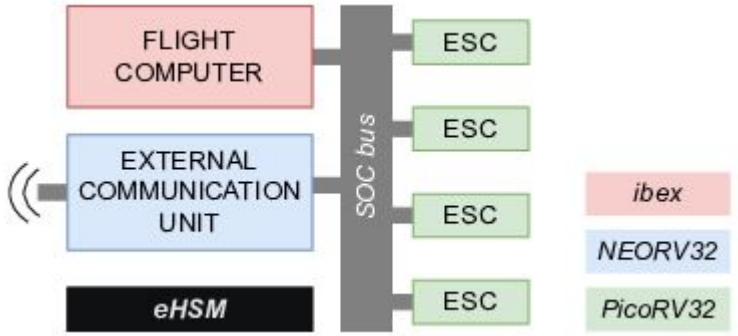
SPOILER: only 2 out of three made it

Trusted IoT

Within Trusted IoT, KU Leuven has worked on Multi-Core RISC-V platforms. The industrial use case focused on drones, operated by multiple RISC-V cores.







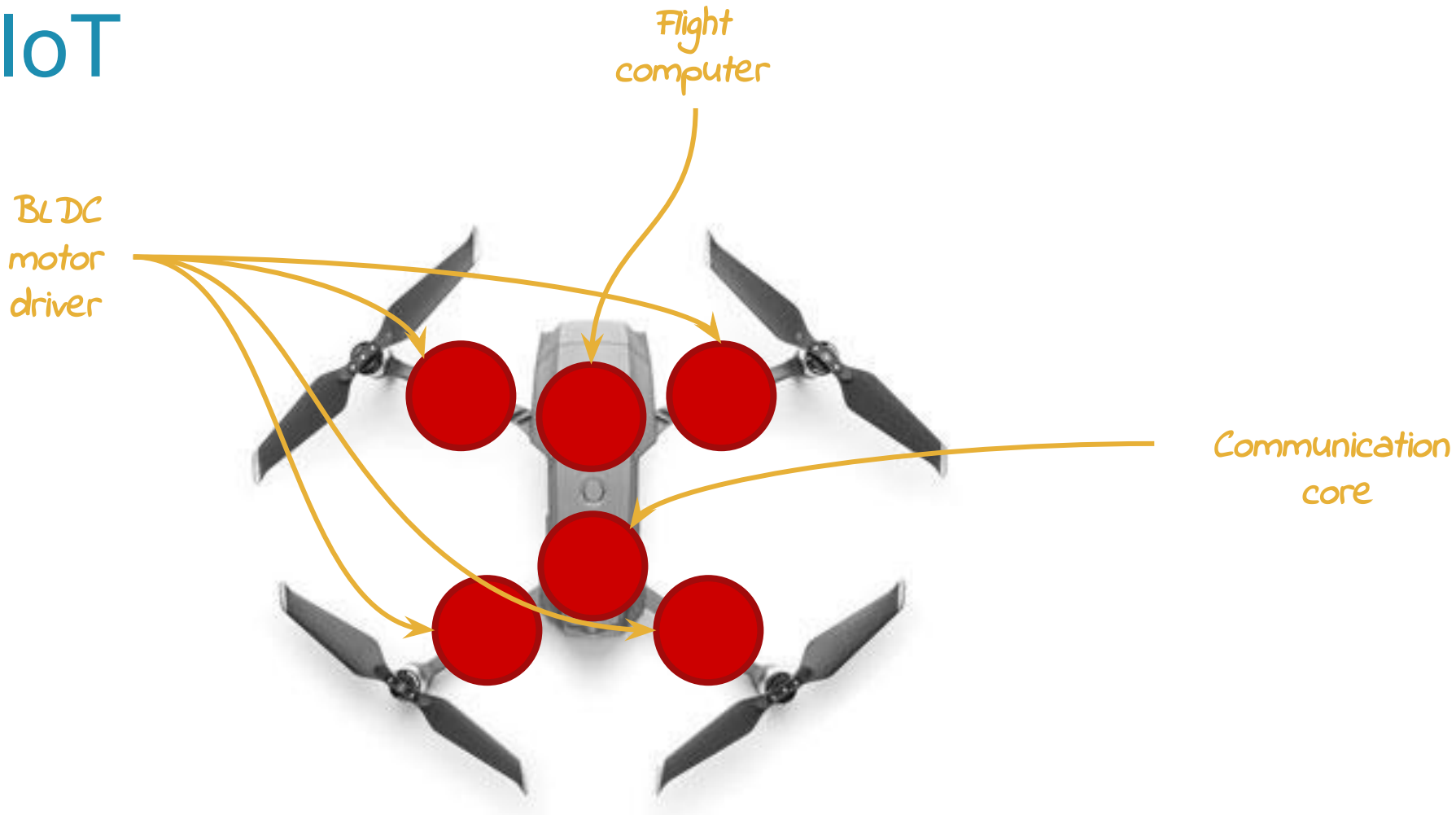
To replace the processors, three existing RISC-V implementations were selected.

How to approach the substitution?

Trusted IoT

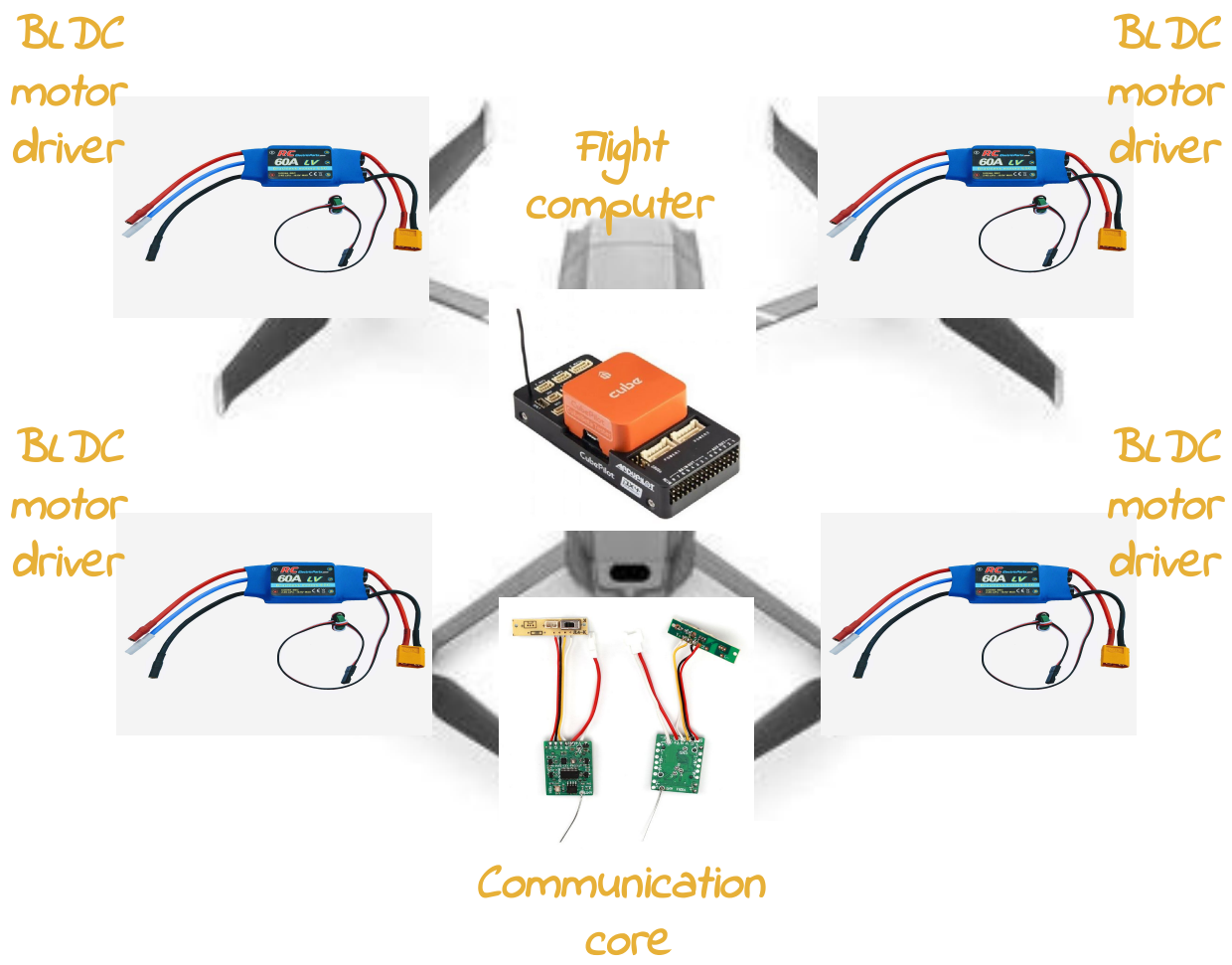
Take the device

Trusted IoT



Identify the processors

Trusted IoT



Identify the processors

Trusted IoT

Replace every processor with a suitable RISC-V implementation

Trusted IoT

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**: VHDL

PicoRV32

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

Trusted IoT

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**: VHDL

PicoRV32

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

		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

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

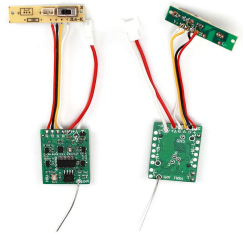
lowRISC

NEORV32

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

NEORV32

RISC-V

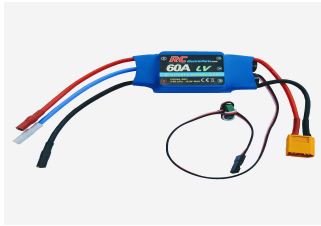


PicoRV32

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

YosysHQ/**picorv32**

PicoRV32 - A Size-Optimized RISC-V CPU

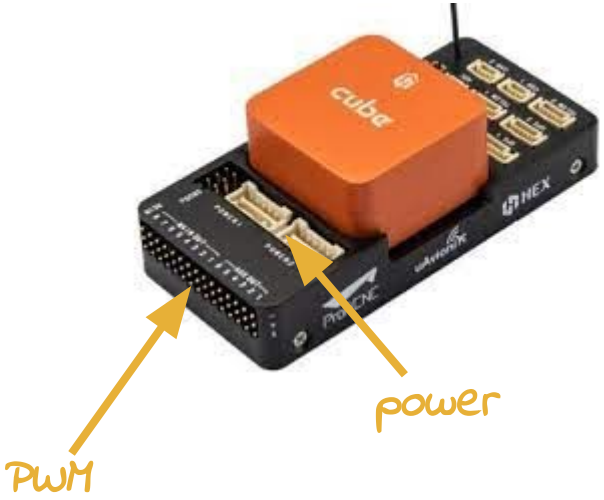


Trusted IoT

lowRISC

NOT started with FC

We want to get around using the microcontroller



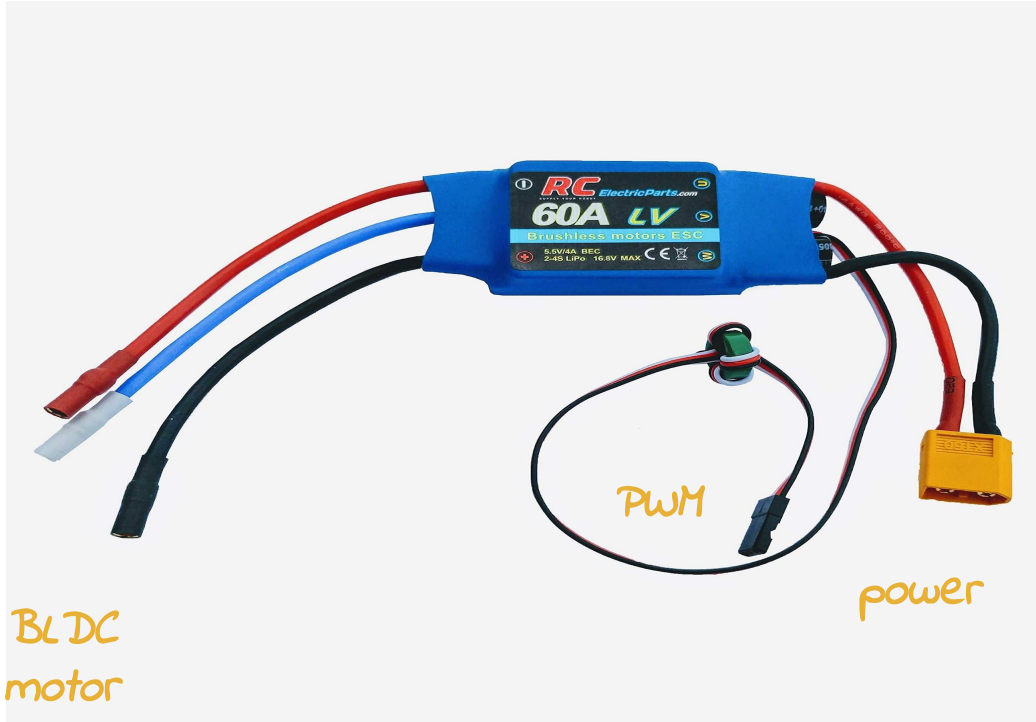
THIS DIDN'T make IT

Trusted IoT

YosysHQ/**picorv32**

PicoRV32 - A Size-Optimized RISC-V CPU

Work on the 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

Trusted IoT

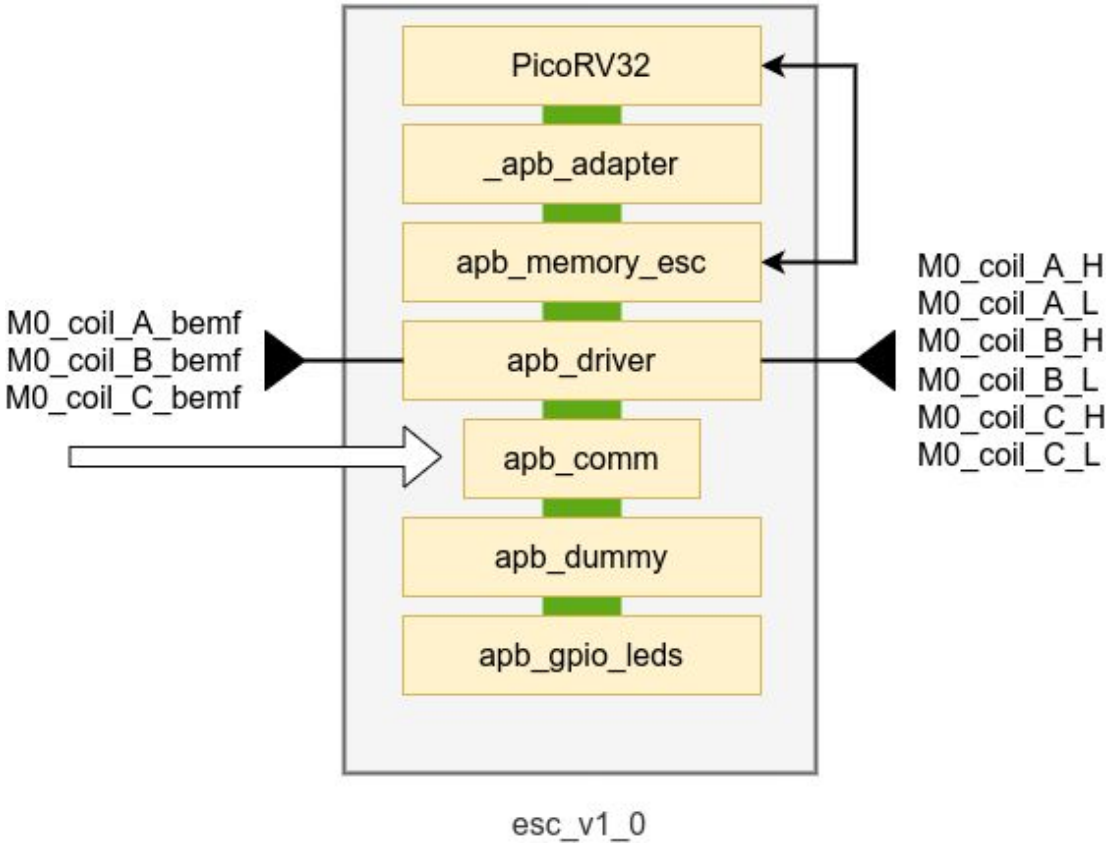
YosysHQ/**picorv32**

PicoRV32 - A Size-Optimized RISC-V CPU

YosysHQ/**picorv32**

PicoRV32 - A Size-Optimized RISC-V CPU

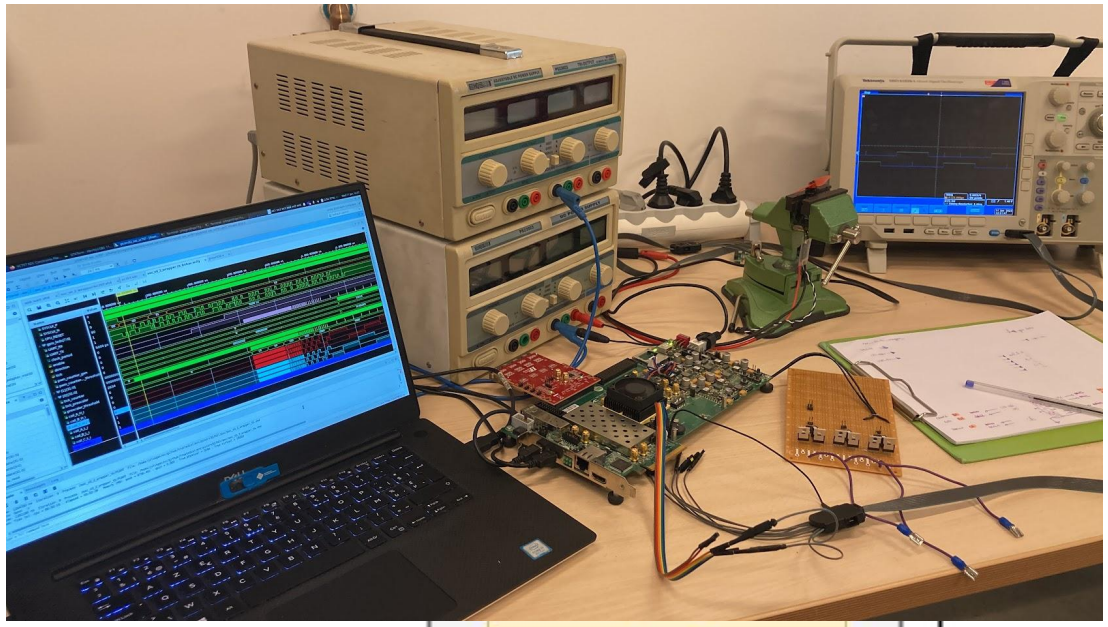
Work on the ESC



Drone BLDC motors are bought

Power electronics have been **re-made**

A simple SOC replaces the microcontroller



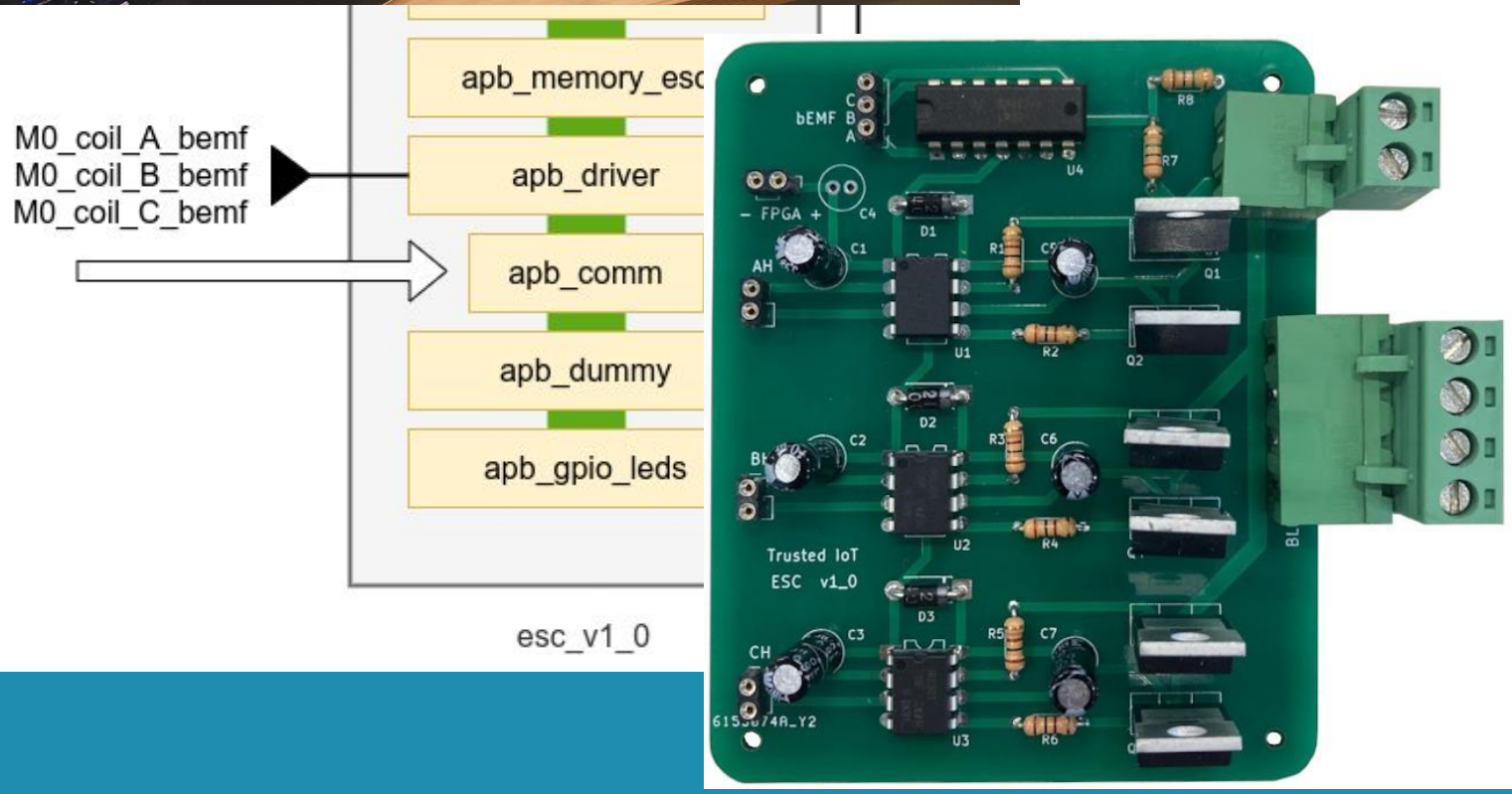
YosysHQ/pico

PicoRV32 - A Size-Optimized RISC-

Drone BLDC m

Power electron

SOC



KU LEUVEN

Trusted IoT

NEORV32 RISC-V

Work on the COMM

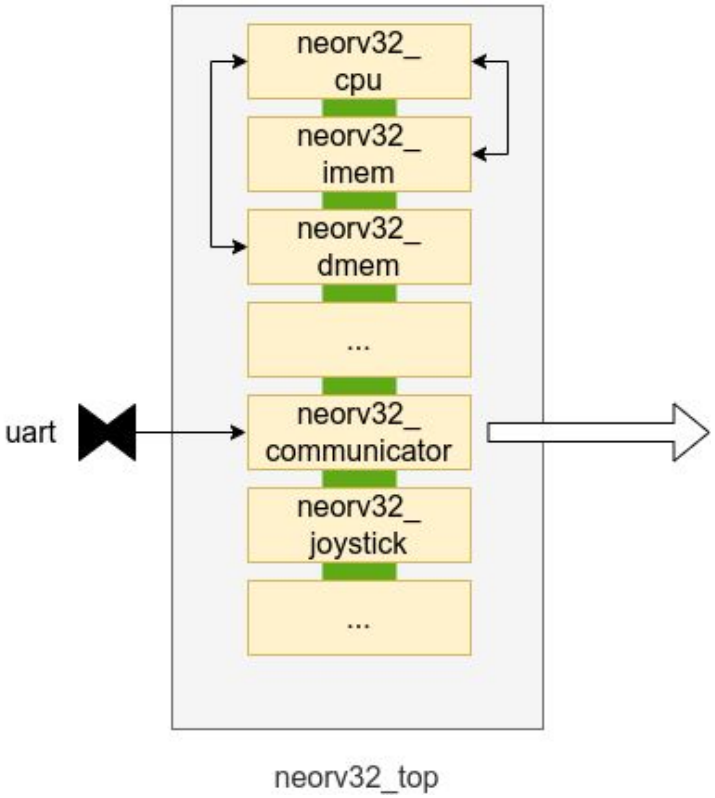
There is a processor in the COMM that:

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

We want to get around using the microcontroller

Trusted IoT

Work on the COMM

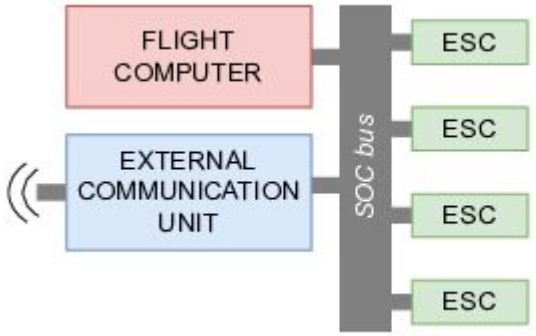


RF control channel is replaced by UART

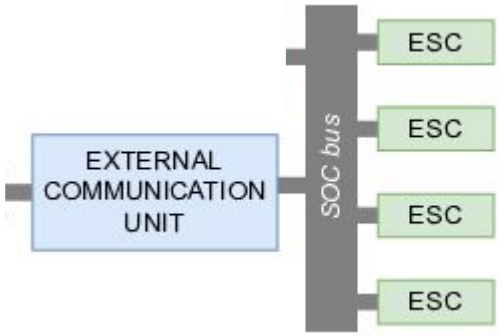
PWM signal is replaced by AXI4 Stream-like

Trusted IoT

Targeted implementation



Current state of implementation

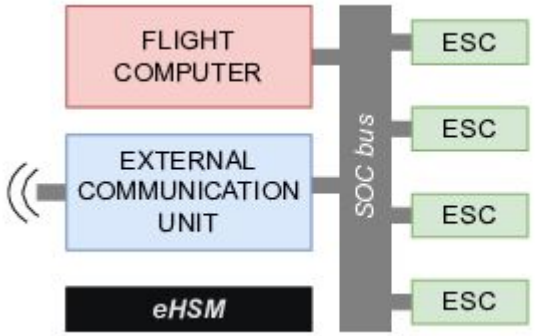


This implementation replaces/represents the “main application” in the assumed use case.

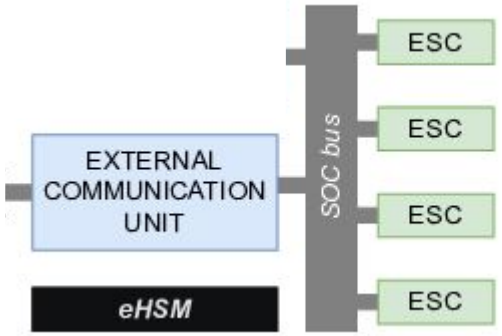
This is the part that needs to be attested.

Trusted IoT

Targeted implementation



Current state of implementation



To attest the entire FPGA, secure communication and access to the ICAP is required.

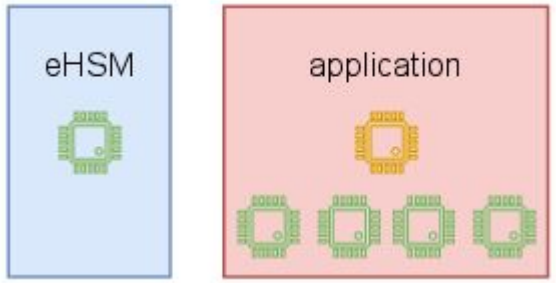
- eHSM
- Secure communication of attestation outcome through LWC winner: ASCON

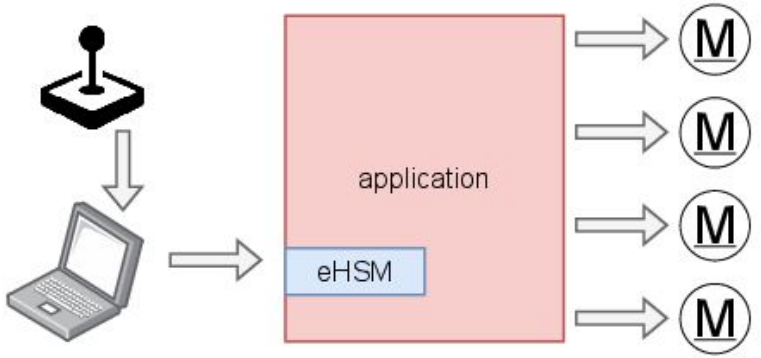
Trusted IoT

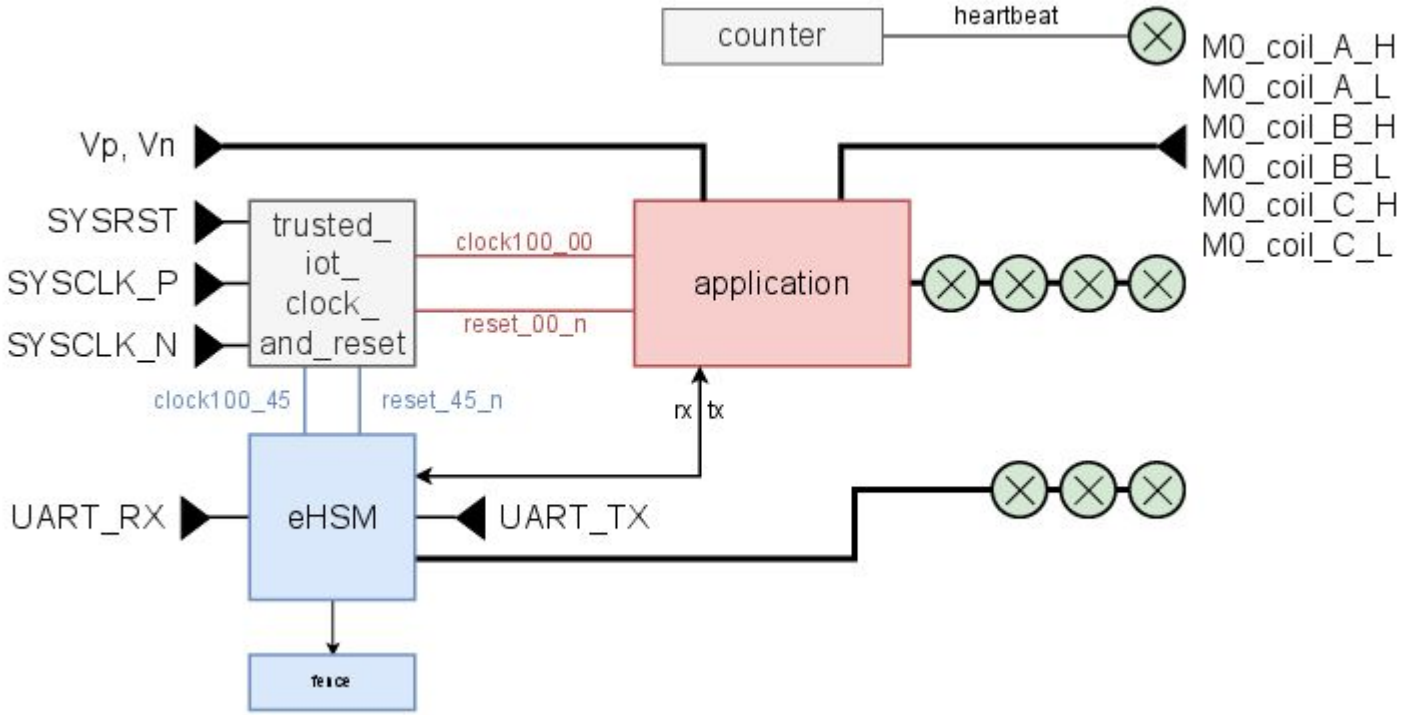
To recap:

The design is split in:

- Application
- eHSM





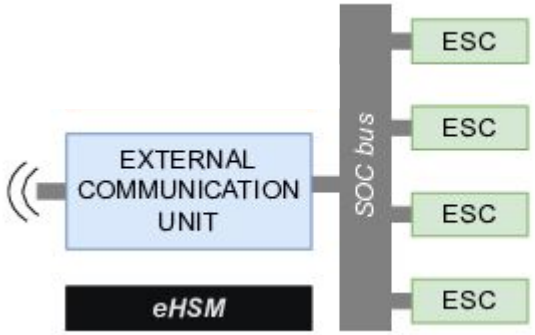


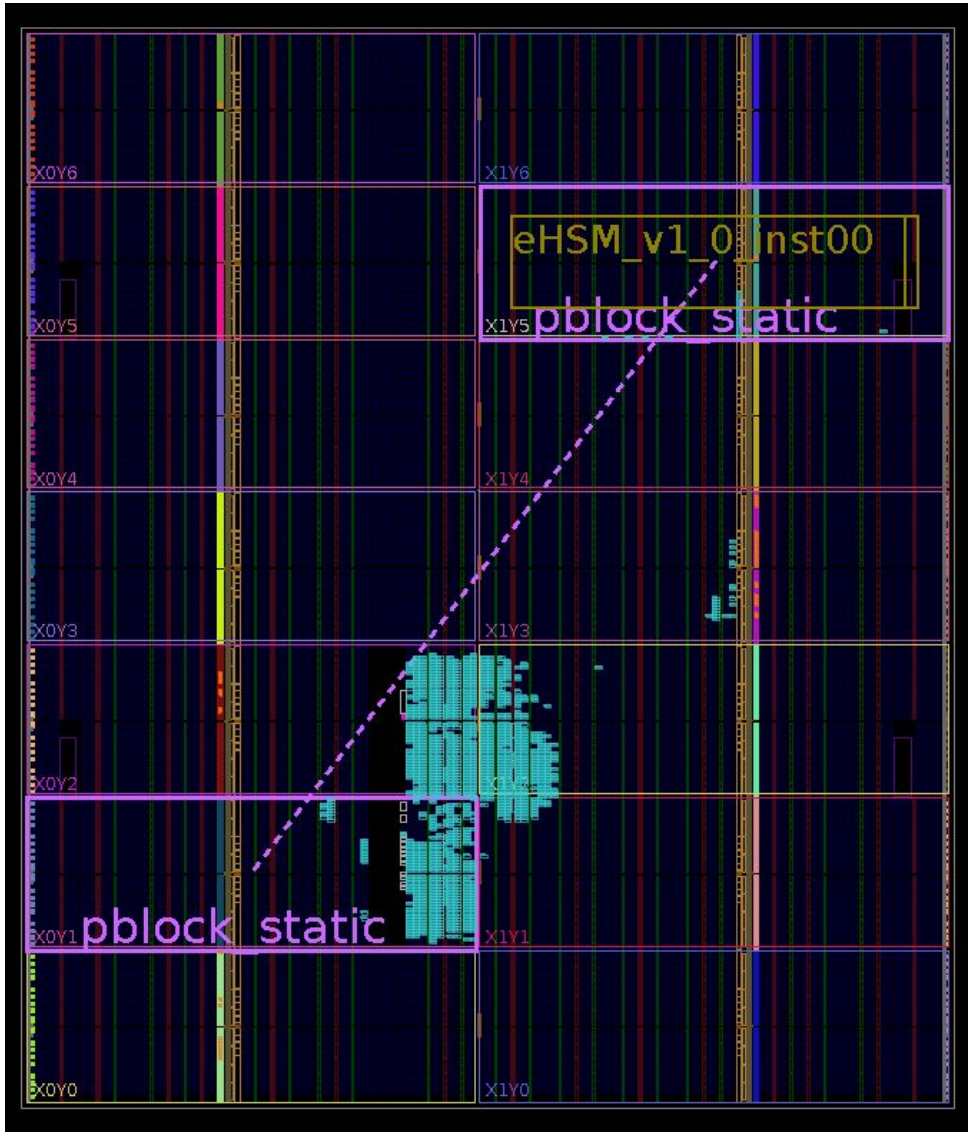
Industrial use case

Results

Trusted IoT

FPGA implementation done

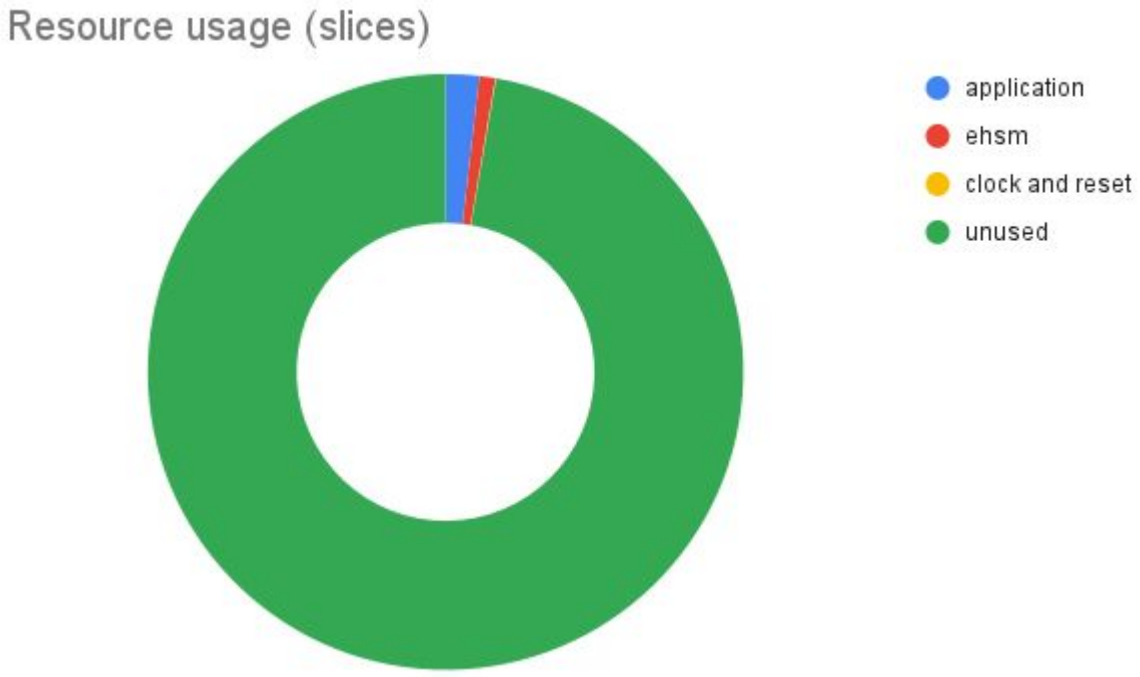


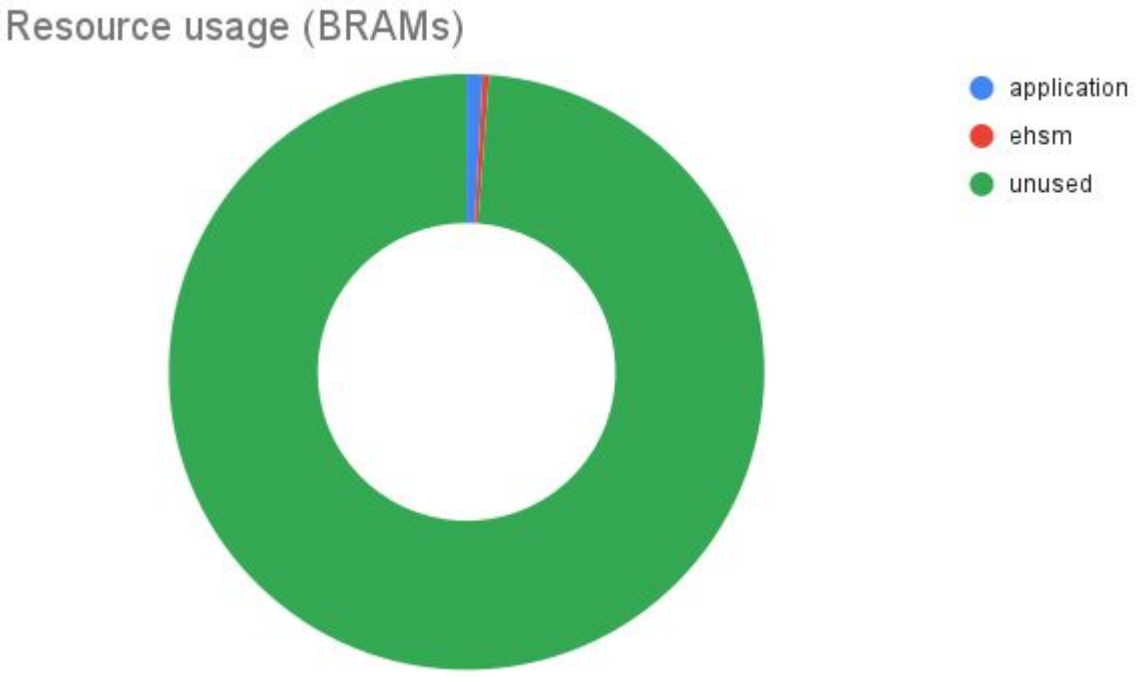


- **pblock_static** contains the eHSM
- the remainder of the reconfigurable fabric is for “the application”

Trusted IoT

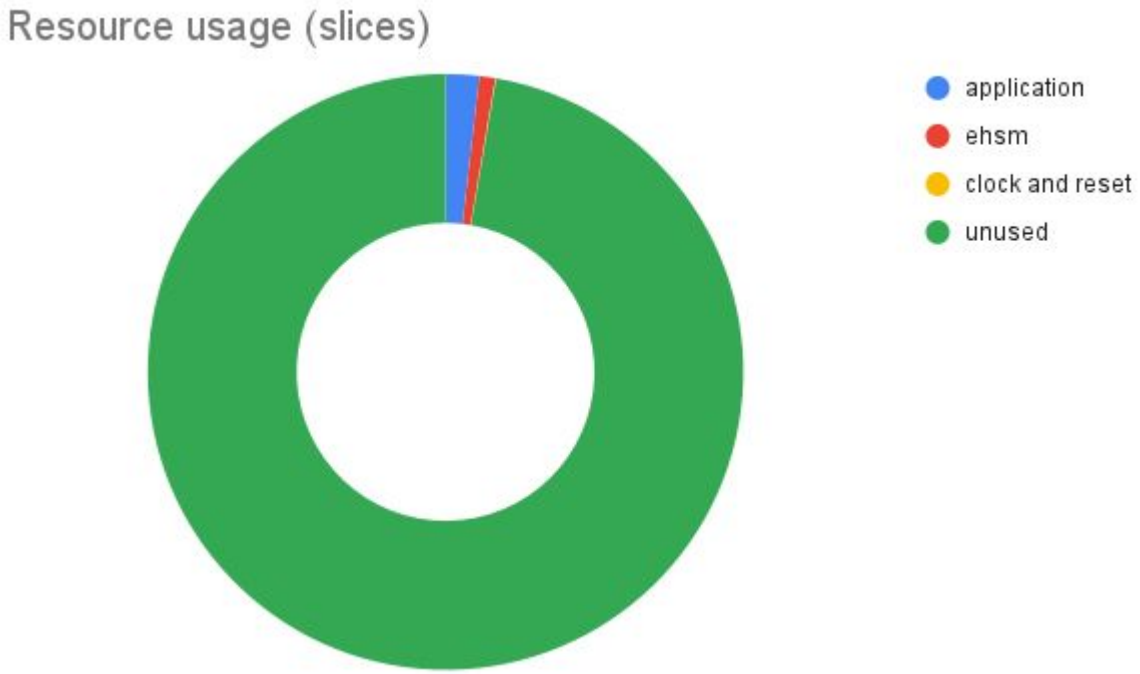
FPGA implementation done (XC7VX485)

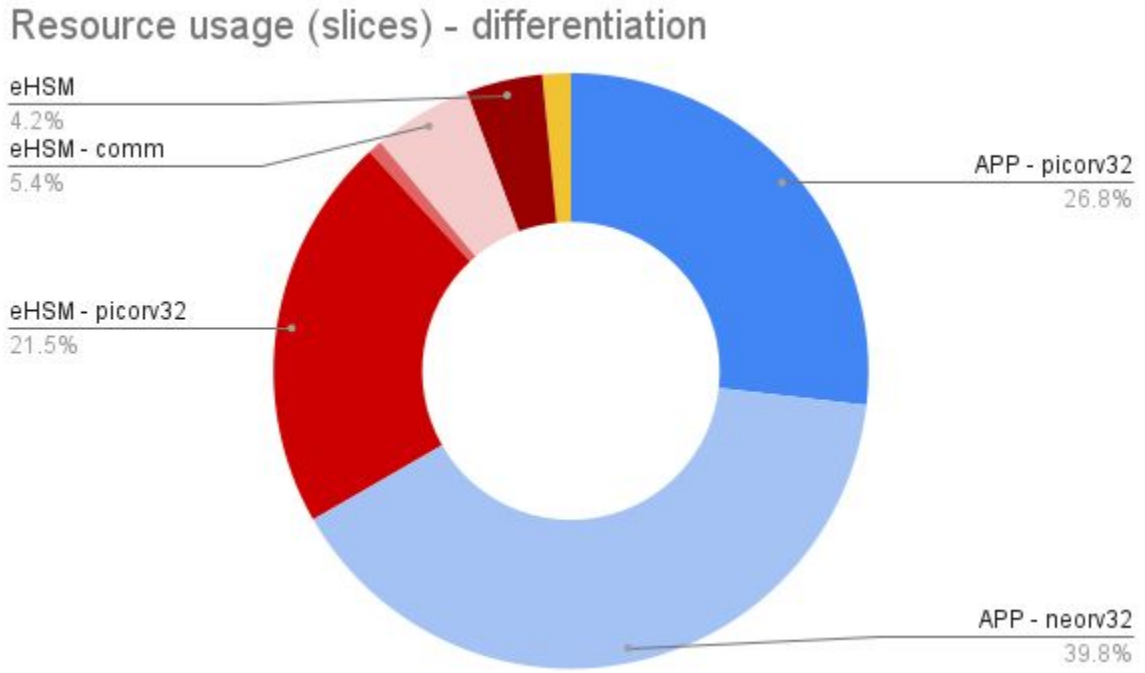




Trusted IoT

FPGA implementation done (XC7VX485)





For completeness: **1/14** MMCME2_ADV used, and **1/2** ICAP

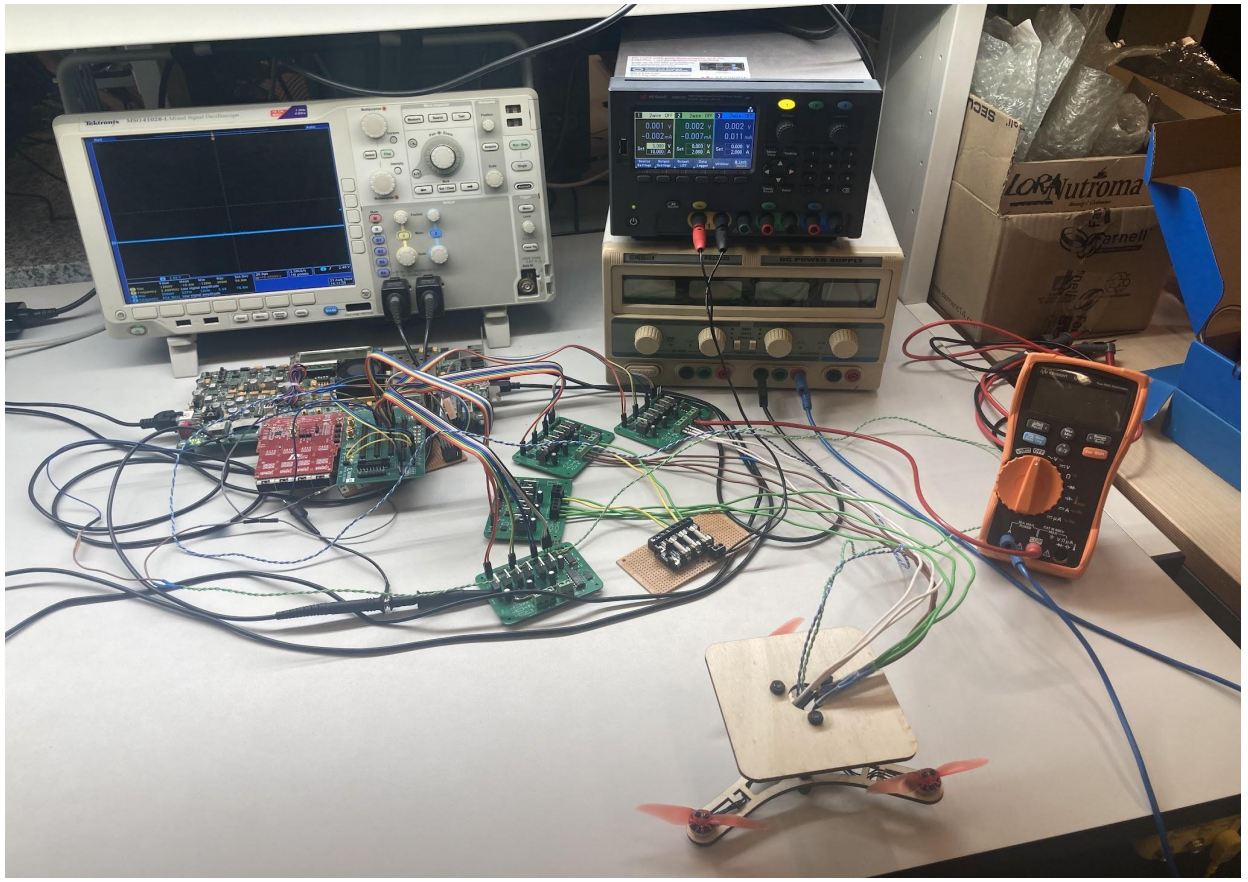
Industrial use case

On the drone

Trusted IoT

Demonstrator

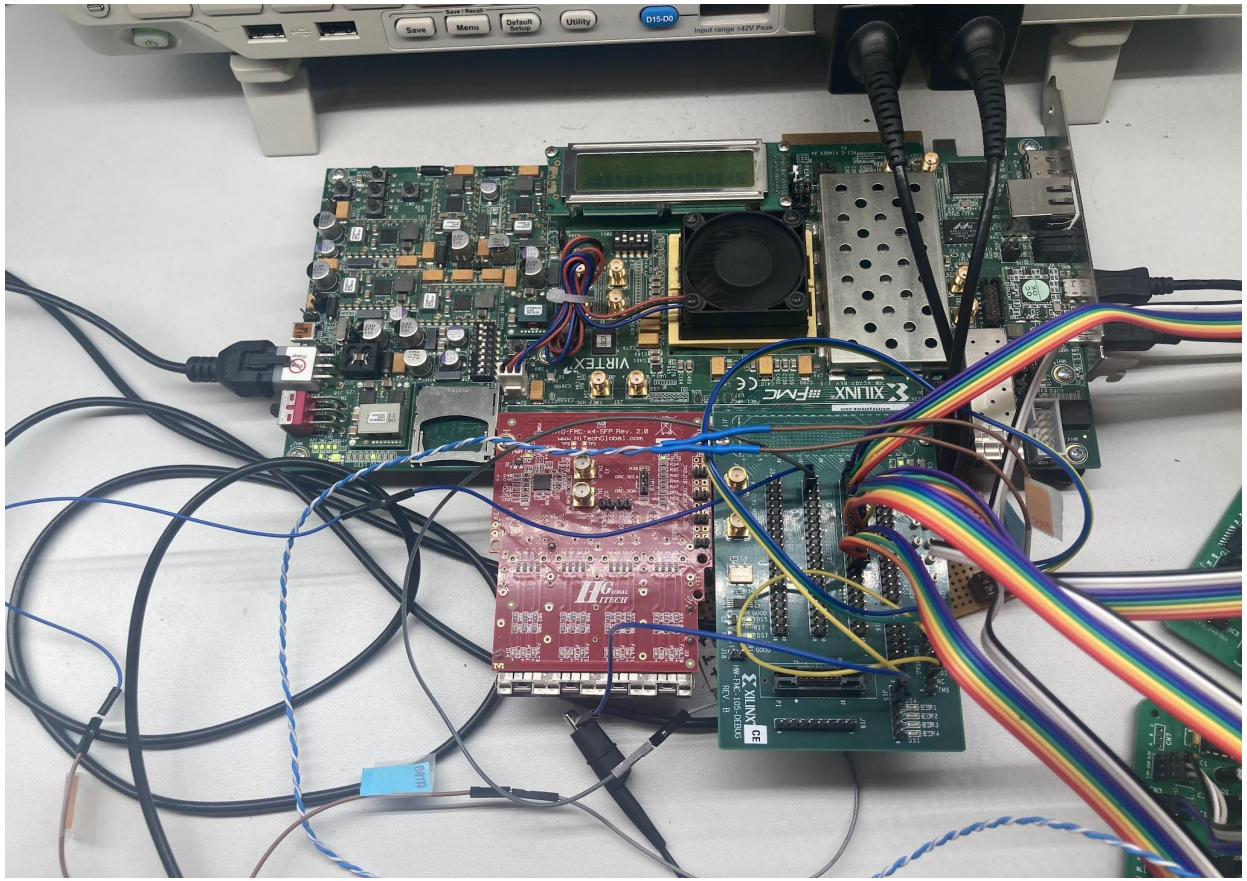
- FPGA with all RISC-V
- power supply
- power electronics PCB’s
- Fuses :-)
- drone with IMU



Trusted IoT

Demonstrator

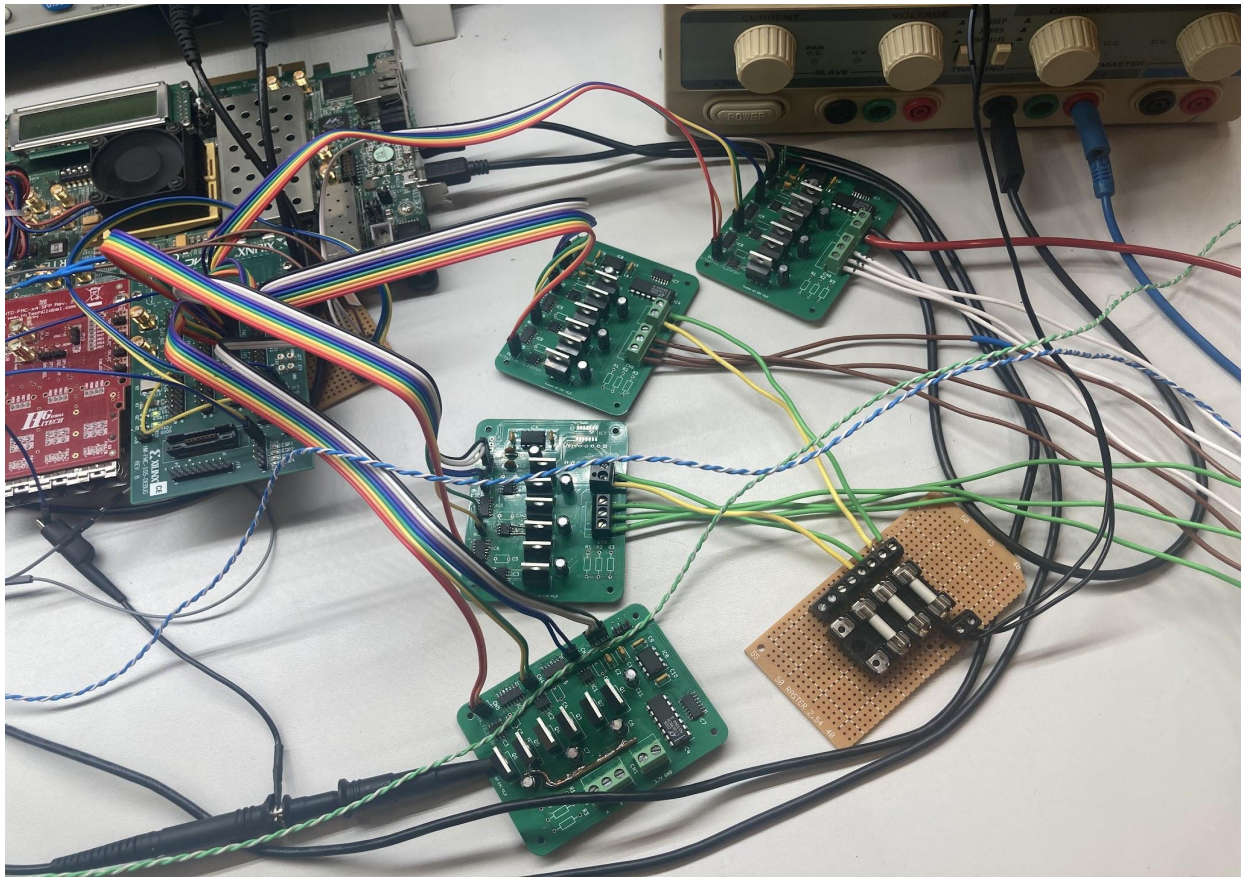
- FPGA with all RISC-V
- power supply
- power electronics PCB’s
- Fuses :-)
- drone with IMU



Trusted IoT

Demonstrator

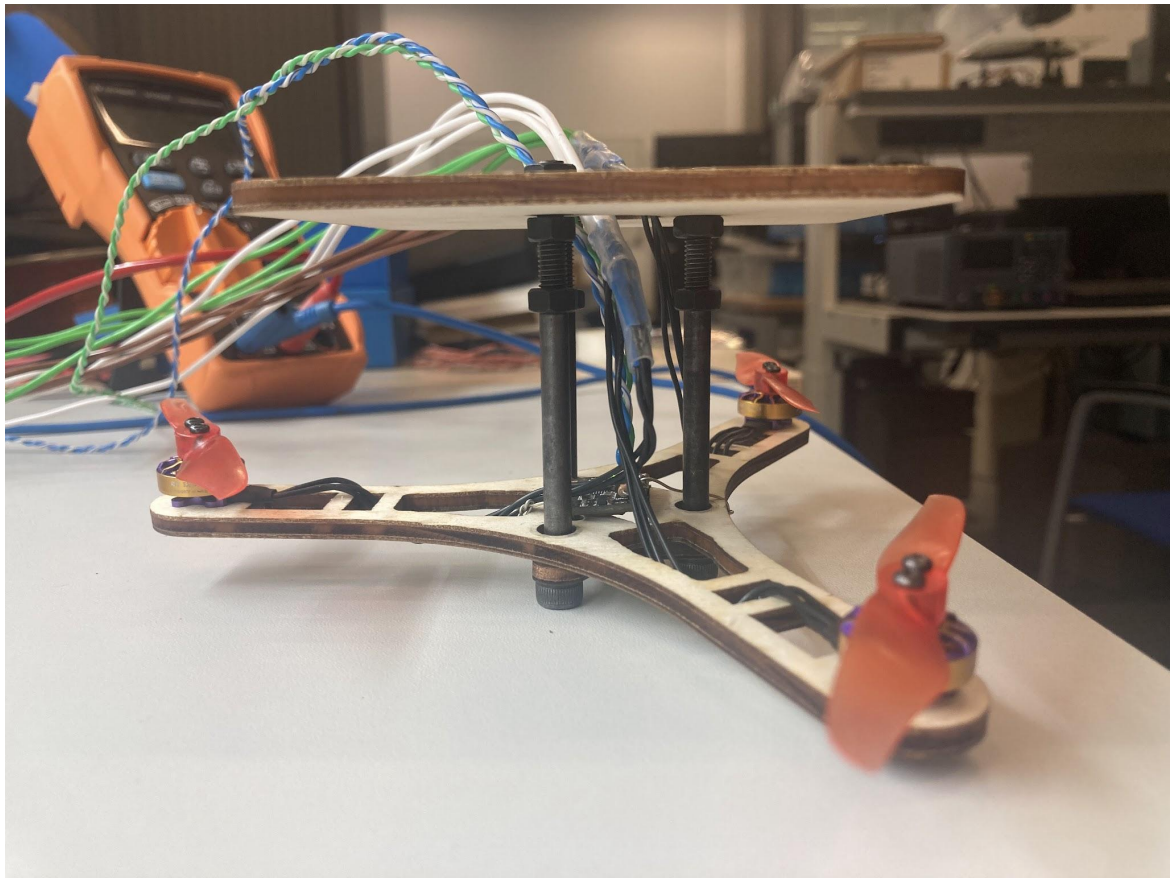
- FPGA with all RISC-V
- power supply
- power electronics PCB’s
- Fuses :-)
- drone with IMU



Trusted IoT

Demonstrator

- FPGA with all RISC-V
- power supply
- power electronics PCB’s
- Fuses :-)
- drone with IMU



Industrial use case

Conclusions & Future work

Trusted IoT

Conclusions:

- Scalable attestation **can be achieved**
- Swapping processors with softcores on FPGA **can be done**
  - better “fitting” processors
  - open source options
- Improved “on-node” communication

Trusted IoT

Conclusions:

- Scalable attestation **can be achieved**
- Swapping processors with softcores on FPGA **can be done**
  - better “fitting” processors
  - open source options
- Improved “on-node” communication

Future work:

- Further exploration of FAR mapping
- Introduction of PR for flexible missions

Follow-up project in preparation: **STELLA**

?

thank you !!