

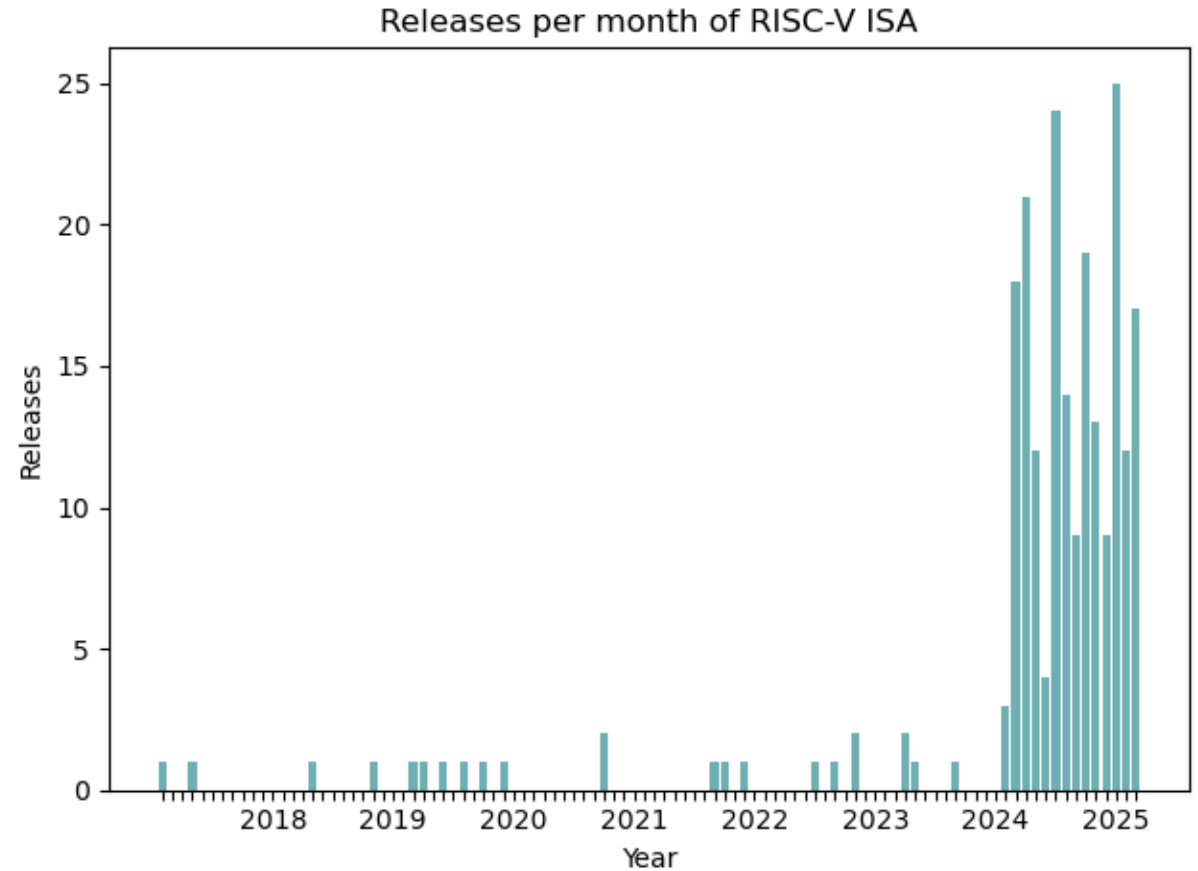
Test-in-the-loop for RISC-V development for space.

Dependable Computing Systems, University of Twente

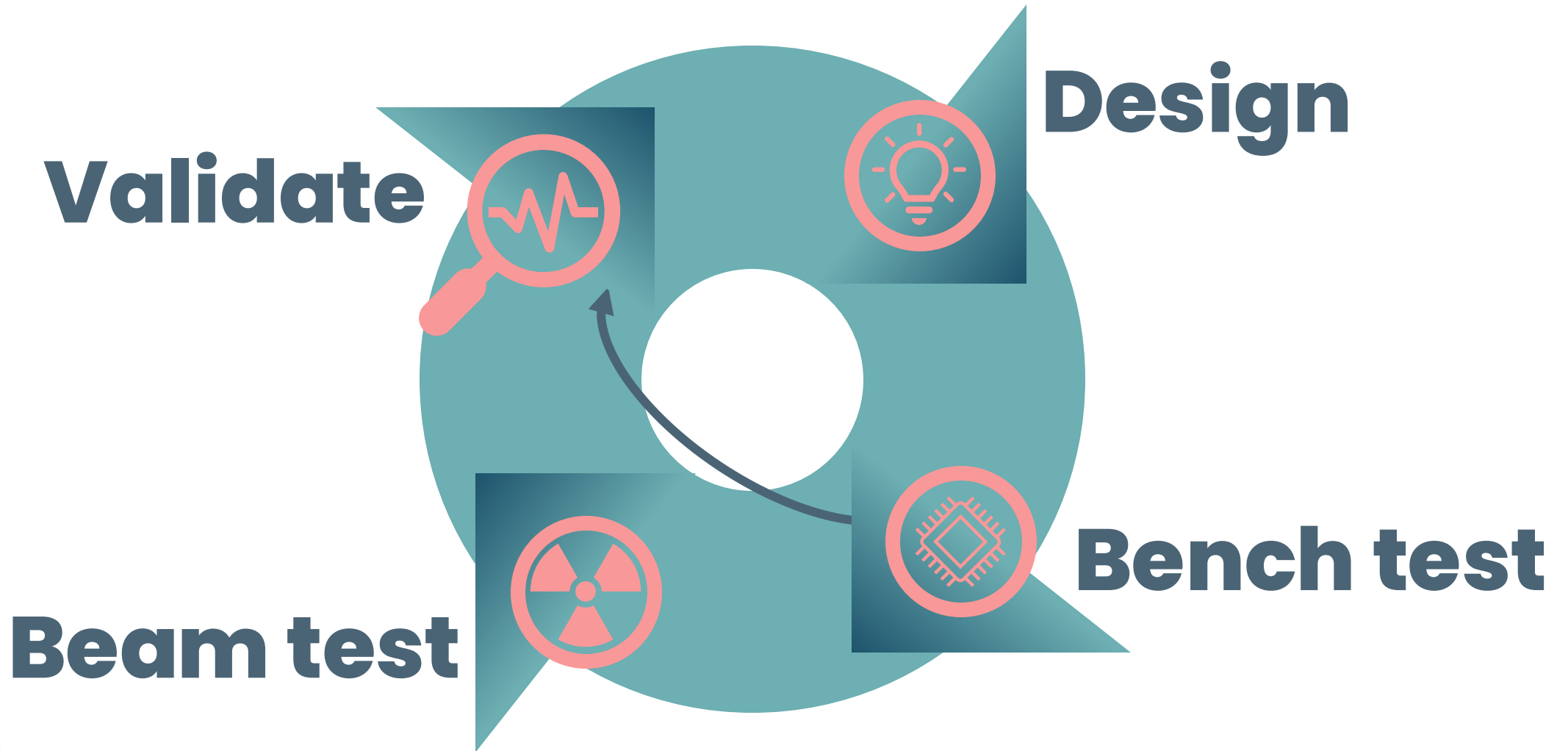
Tijmen Smit

RISC-V is advancing fast

- RISC-V is not static
- Fast design changes require quick test cycles



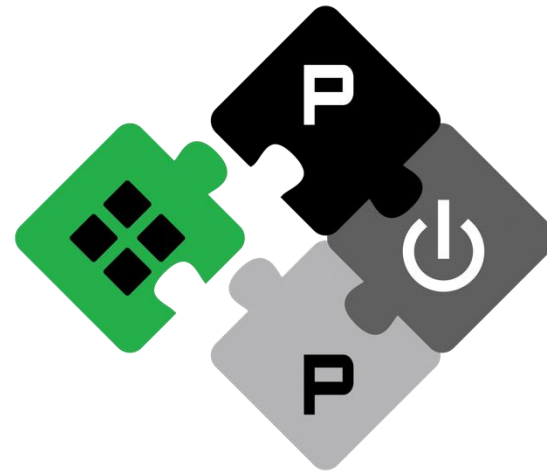
Test-in-the-loop



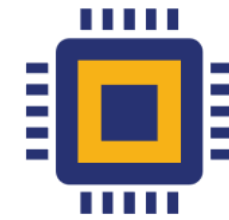
Design

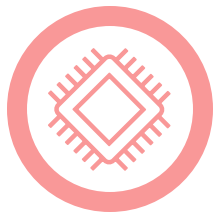
Design of lightweight checkers in terms of performance and area

- Dependable execution environment
- Probabilistic Data Structures instruction checker
- Performance counters monitoring for security and reliability
- Delay Detection methods



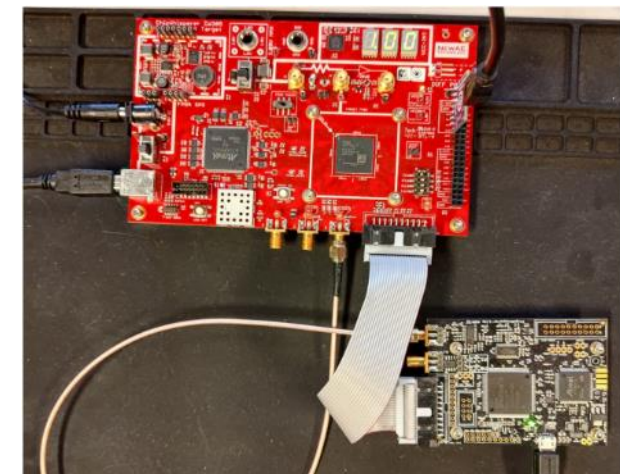
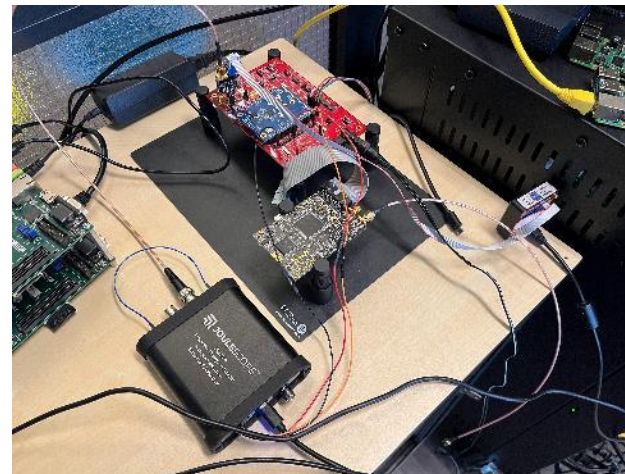
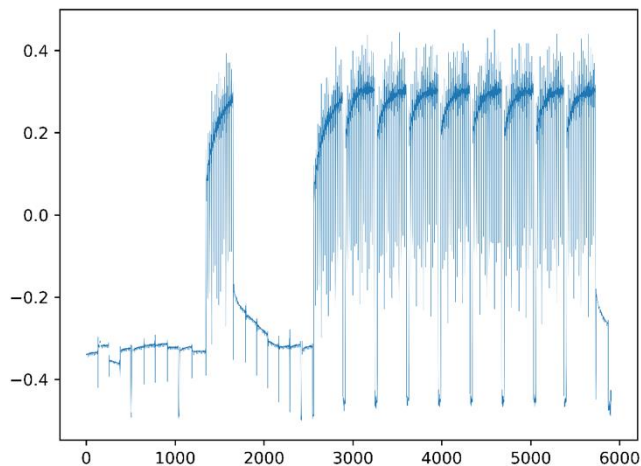
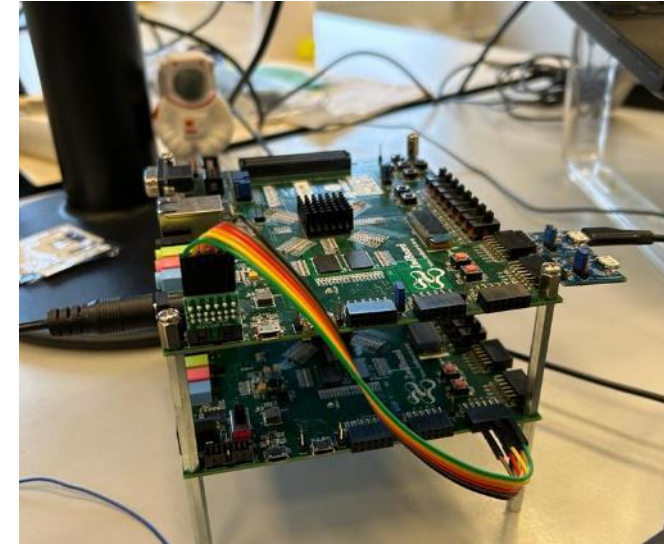
NEORV32
RISC-V®





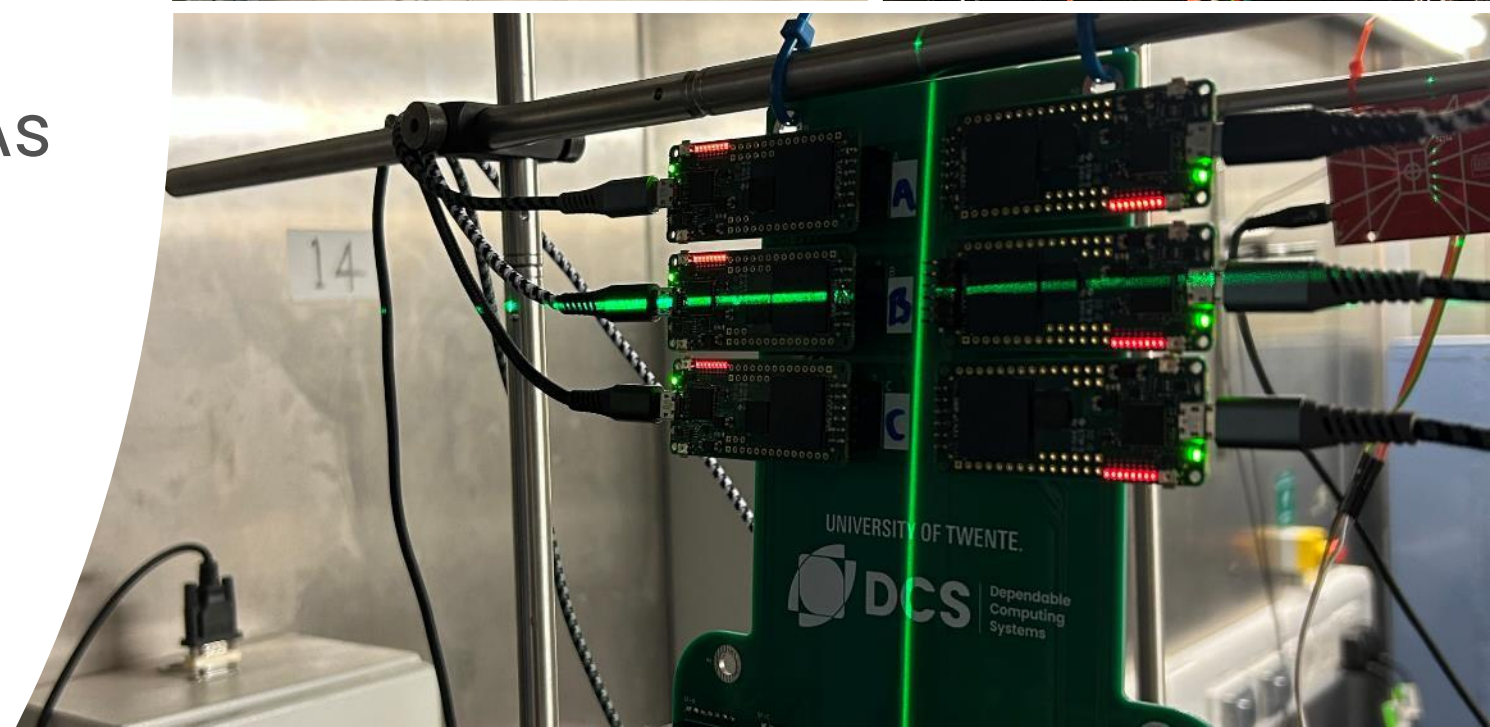
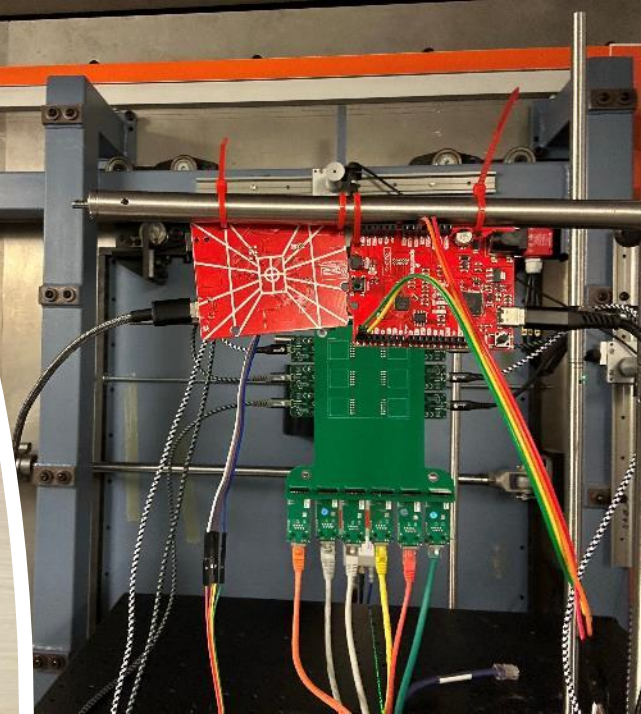
Bench

- Post simulation
- Multiplatform Emulation Fault Injection
- Side-Channel Analysis
 - ChipWhisperer
 - In-house developed platform



Beam

- Real beam, real results
- Protons (HollandPTC)
- Neutrons (ChipIR)
- Evaluation in Flash FPGAs



Single Event Effect

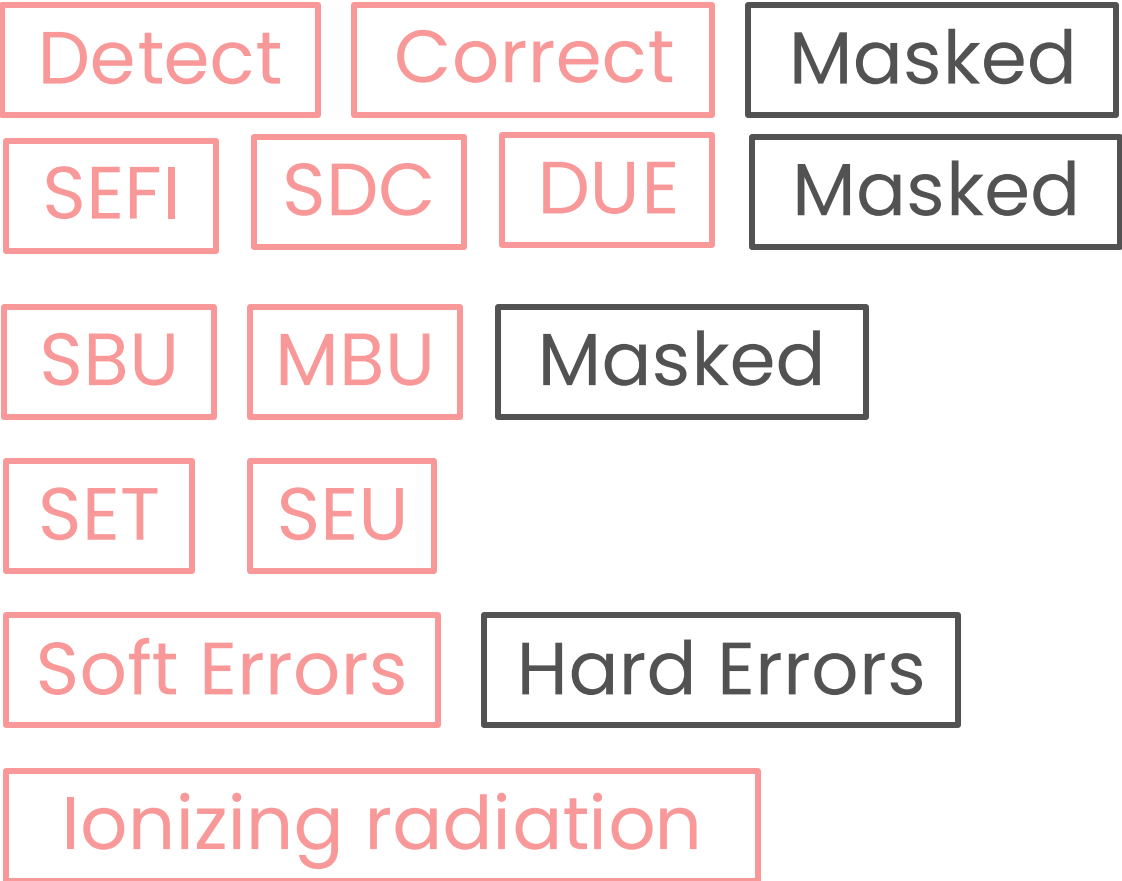
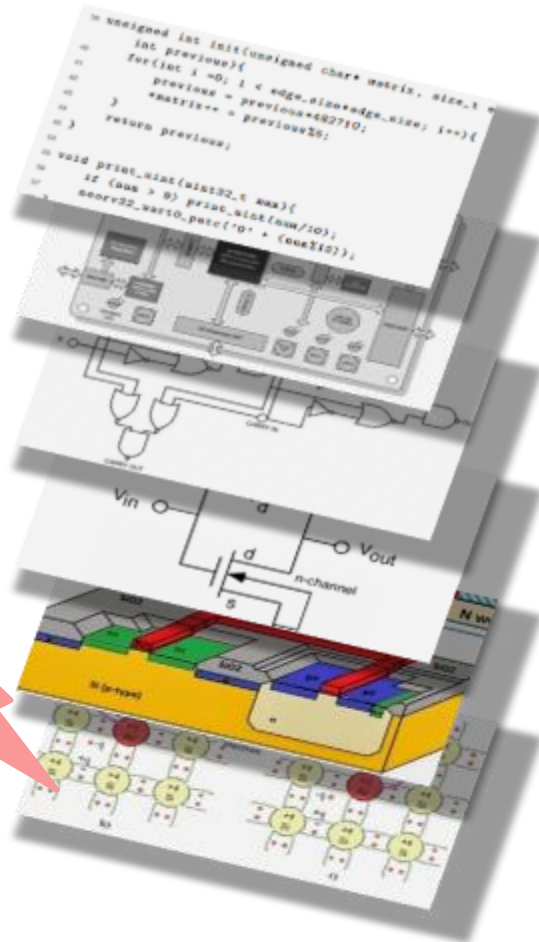
System level: Program,
Microarchitecture

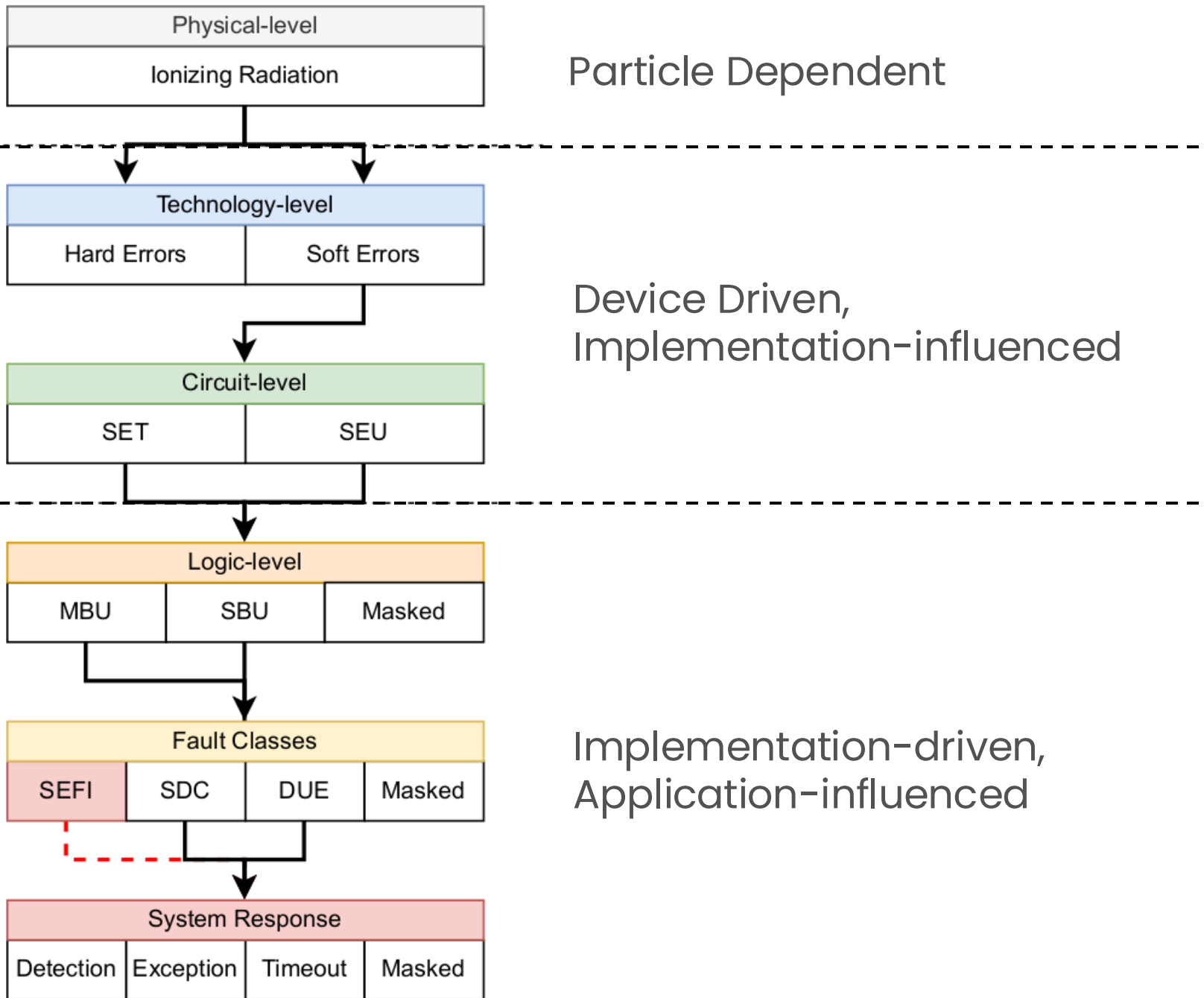
Logic-level

Circuit-level

Technology level

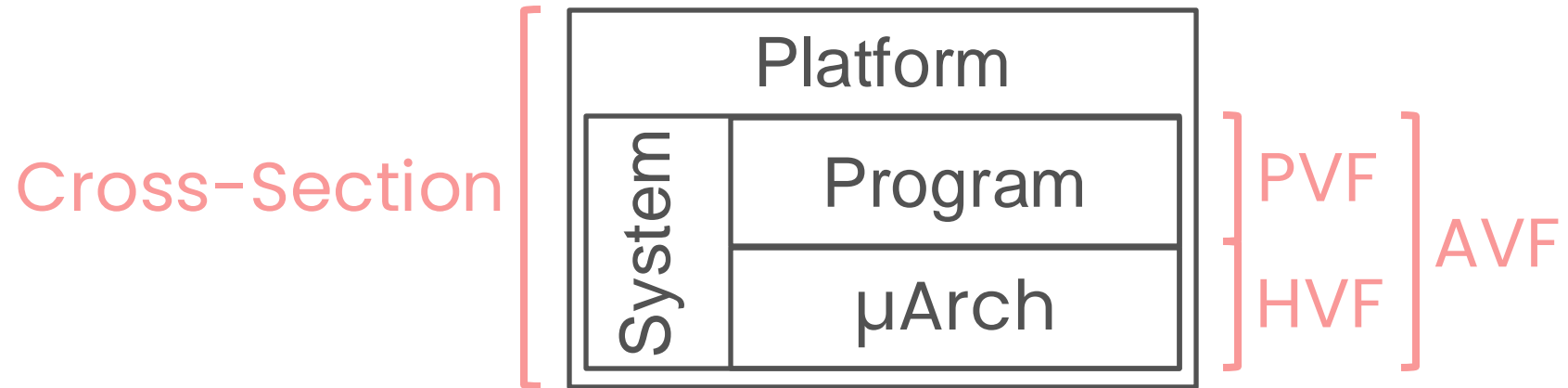
Physical level



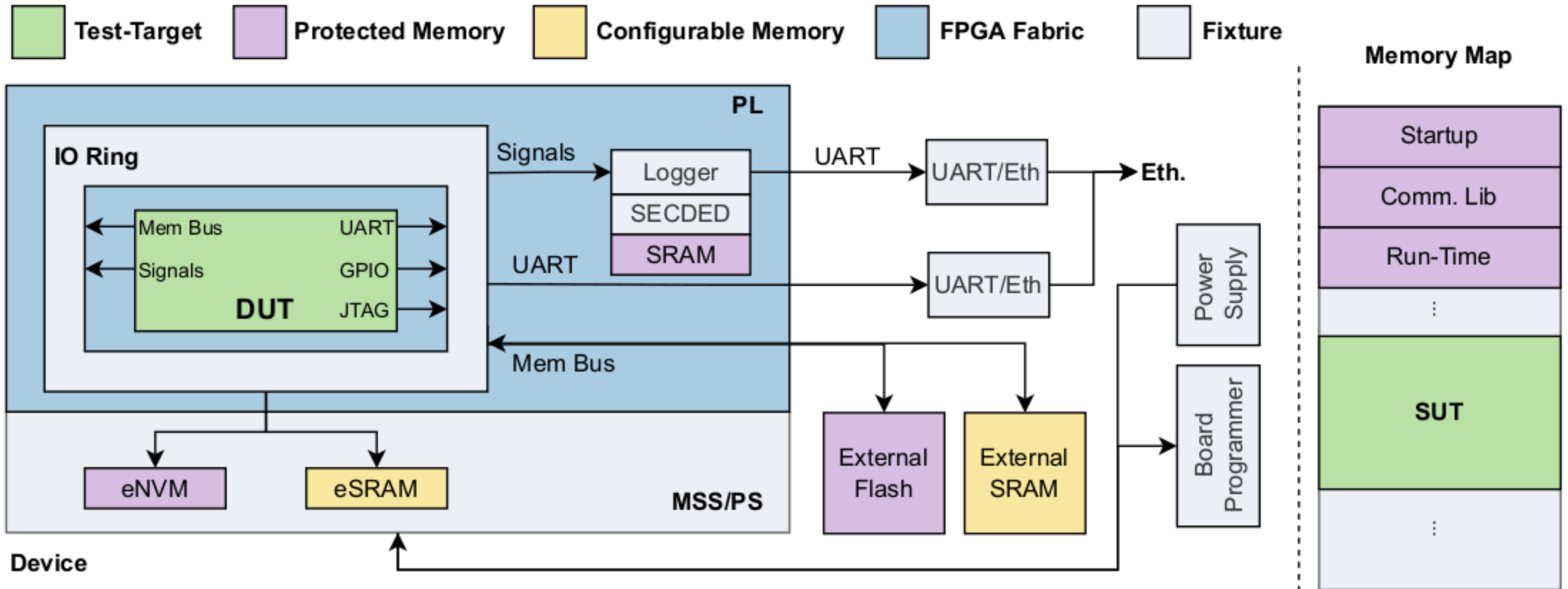




Validate

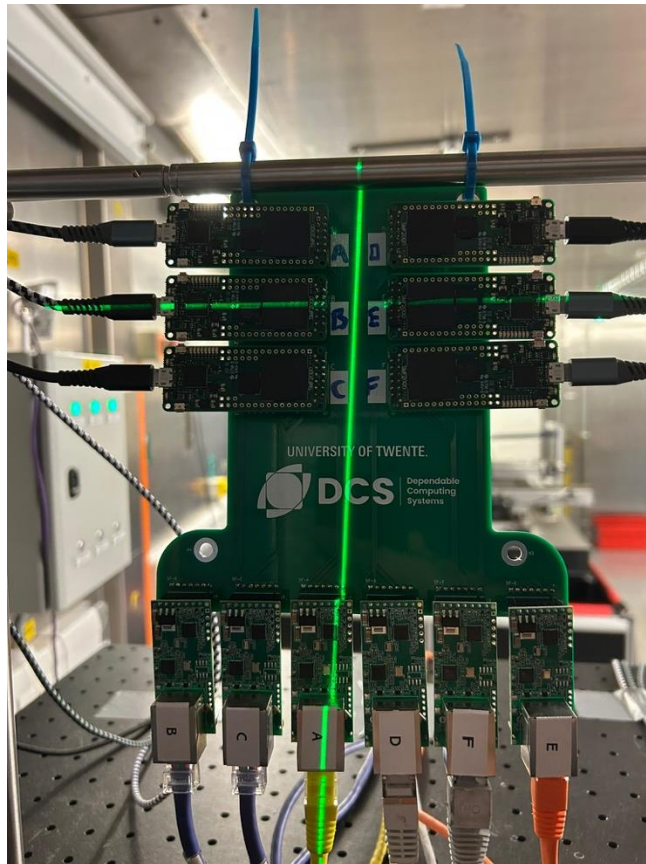


DUT Fixture and SW Harness

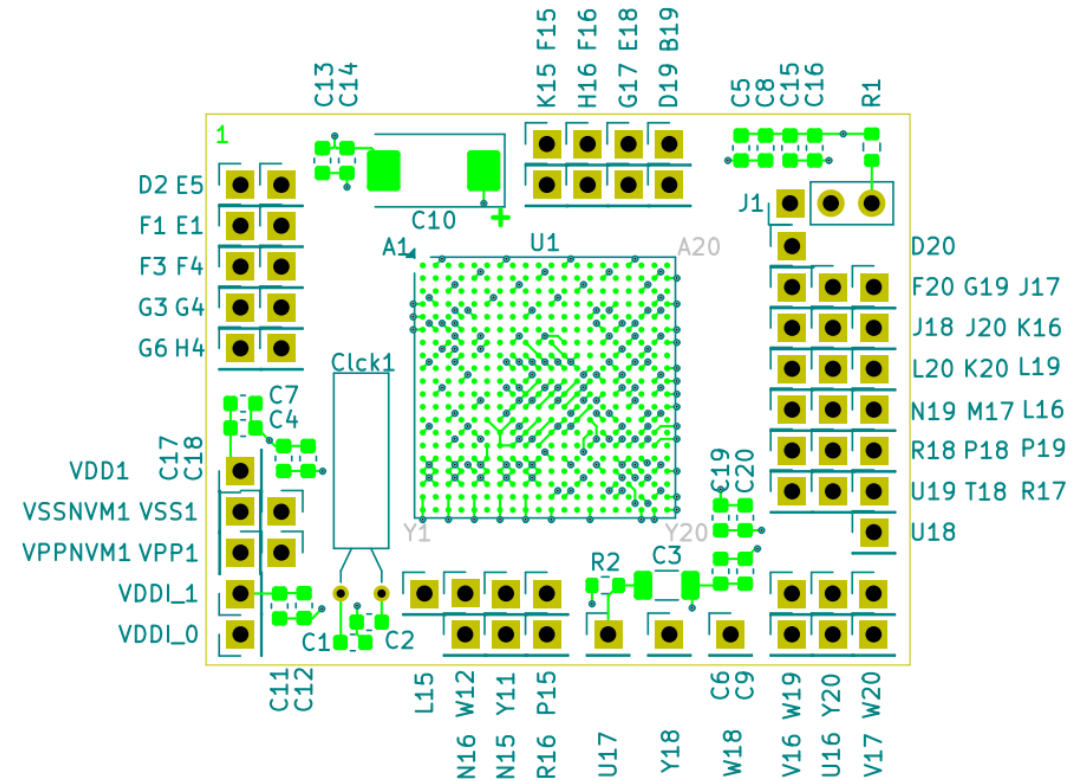


Physical Fixture

Carrier Board with COTS boards

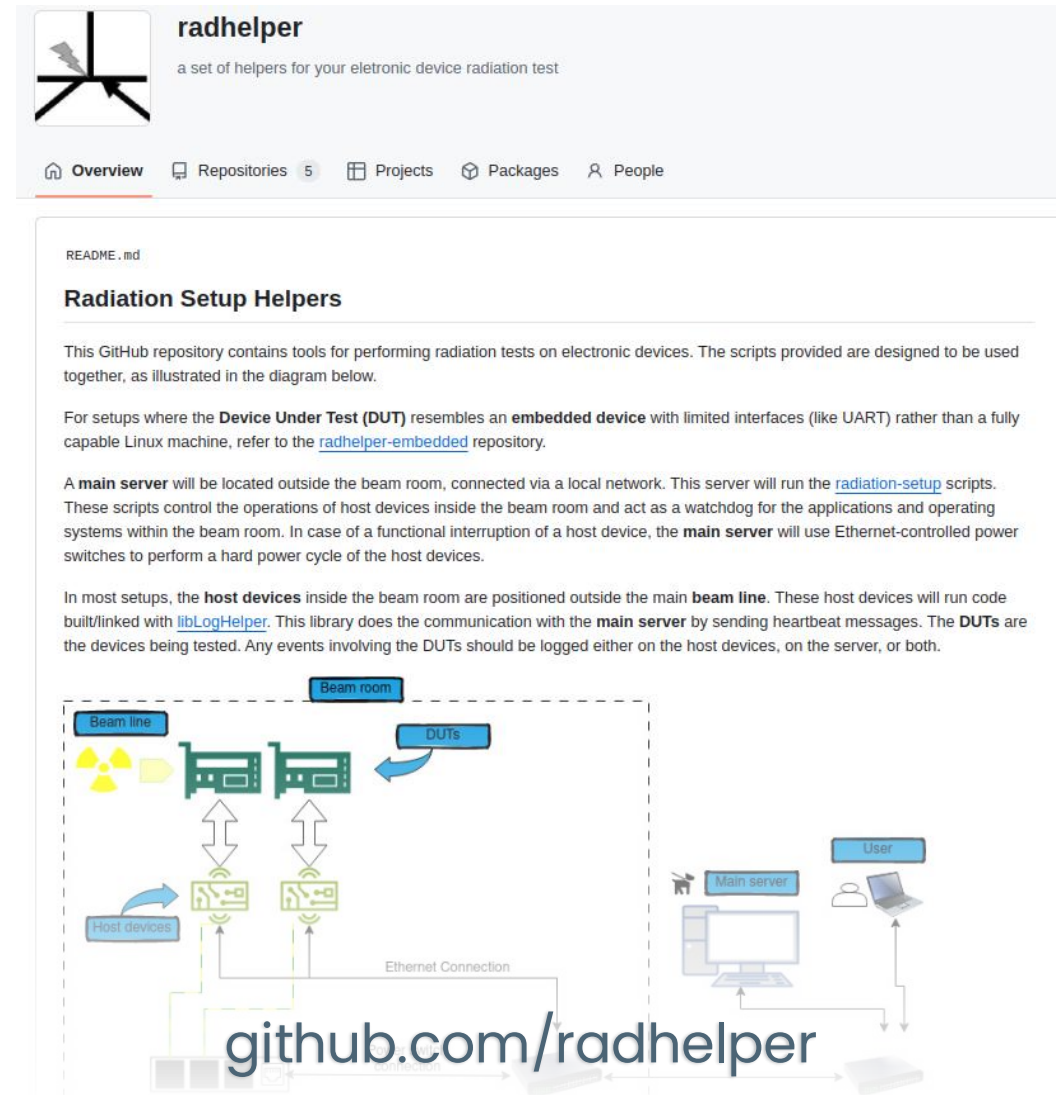


Carrier Board with Fully custom boards



radhelper-embedded

- Contributing to radhelper project
- Open SW fixture
- Focus on devices with few standard interfaces
- Custom communication protocol



The screenshot shows the GitHub repository page for **radhelper**, which is described as "a set of helpers for your electronic device radiation test". The repository has 5 repositories, 0 projects, 0 packages, and 0 people. The README file is open, showing the following content:

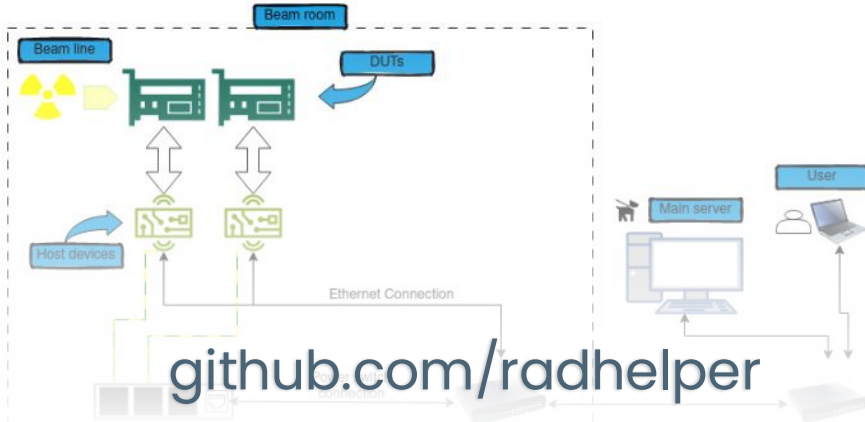
Radiation Setup Helpers

This GitHub repository contains tools for performing radiation tests on electronic devices. The scripts provided are designed to be used together, as illustrated in the diagram below.

For setups where the **Device Under Test (DUT)** resembles an **embedded device** with limited interfaces (like UART) rather than a fully capable Linux machine, refer to the [radhelper-embedded](#) repository.

A **main server** will be located outside the beam room, connected via a local network. This server will run the [radiation-setup](#) scripts. These scripts control the operations of host devices inside the beam room and act as a watchdog for the applications and operating systems within the beam room. In case of a functional interruption of a host device, the **main server** will use Ethernet-controlled power switches to perform a hard power cycle of the host devices.

In most setups, the **host devices** inside the beam room are positioned outside the main **beam line**. These host devices will run code built/linked with [libLogHelper](#). This library does the communication with the **main server** by sending heartbeat messages. The **DUTs** are the devices being tested. Any events involving the DUTs should be logged either on the host devices, on the server, or both.



The diagram illustrates the system architecture. A dashed box labeled "Beam room" contains a "Beam line" with a radiation source, "Host devices" connected to the beam line, and "DUTs" (Device Under Test) connected to the host devices. A "Main server" is located outside the beam room, connected to the host devices via an "Ethernet Connection". A "User" is also connected to the main server. The URL github.com/radhelper is displayed at the bottom of the diagram.

Validate

Correlate expected with observed results for contextual understanding

Beam test

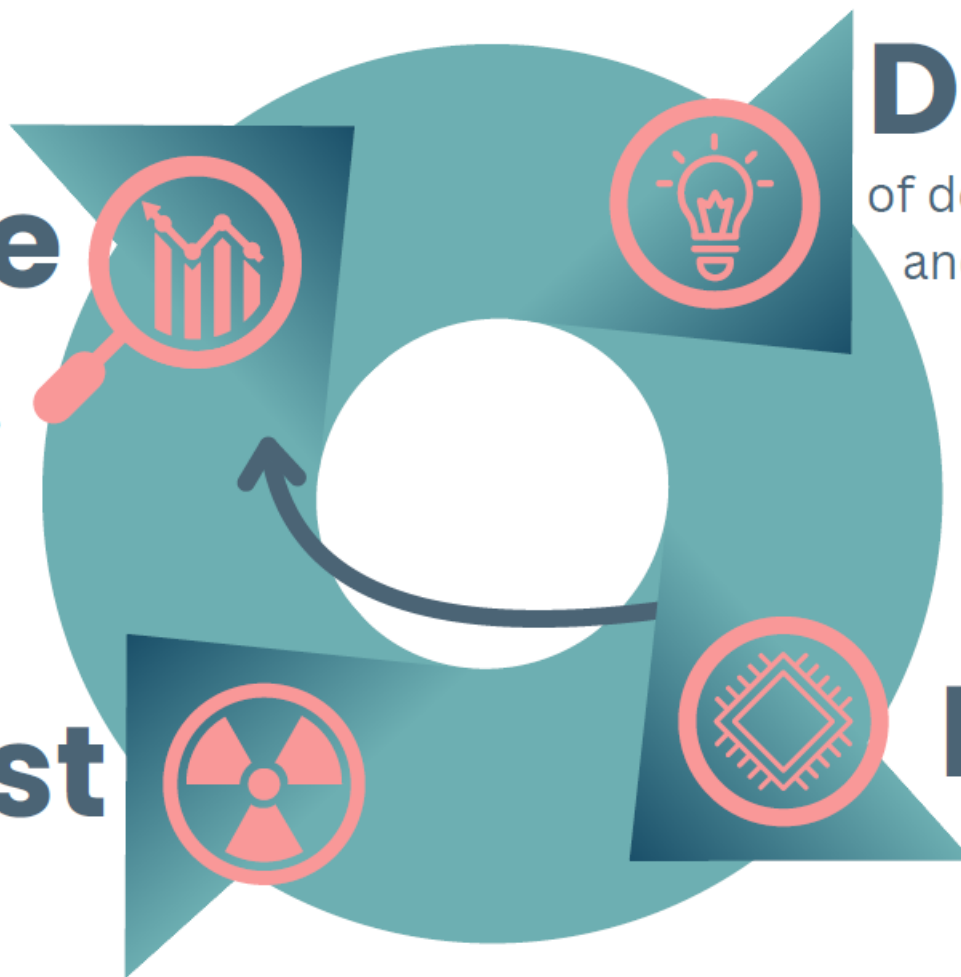
Real beam experiments yield real result

Design

of dependable hardware and software framework

Bench test

Side-Channels, Glitching Attacks, Fault Injection



Dependable Computing Systems Group

Contact us: ut.onl/dcs-group

P16. The RERI-Lite Error Logging Framework
by Michiel Koenderink

**UNIVERSITY
OF TWENTE.**





DCS

Dependable
Computing
Systems