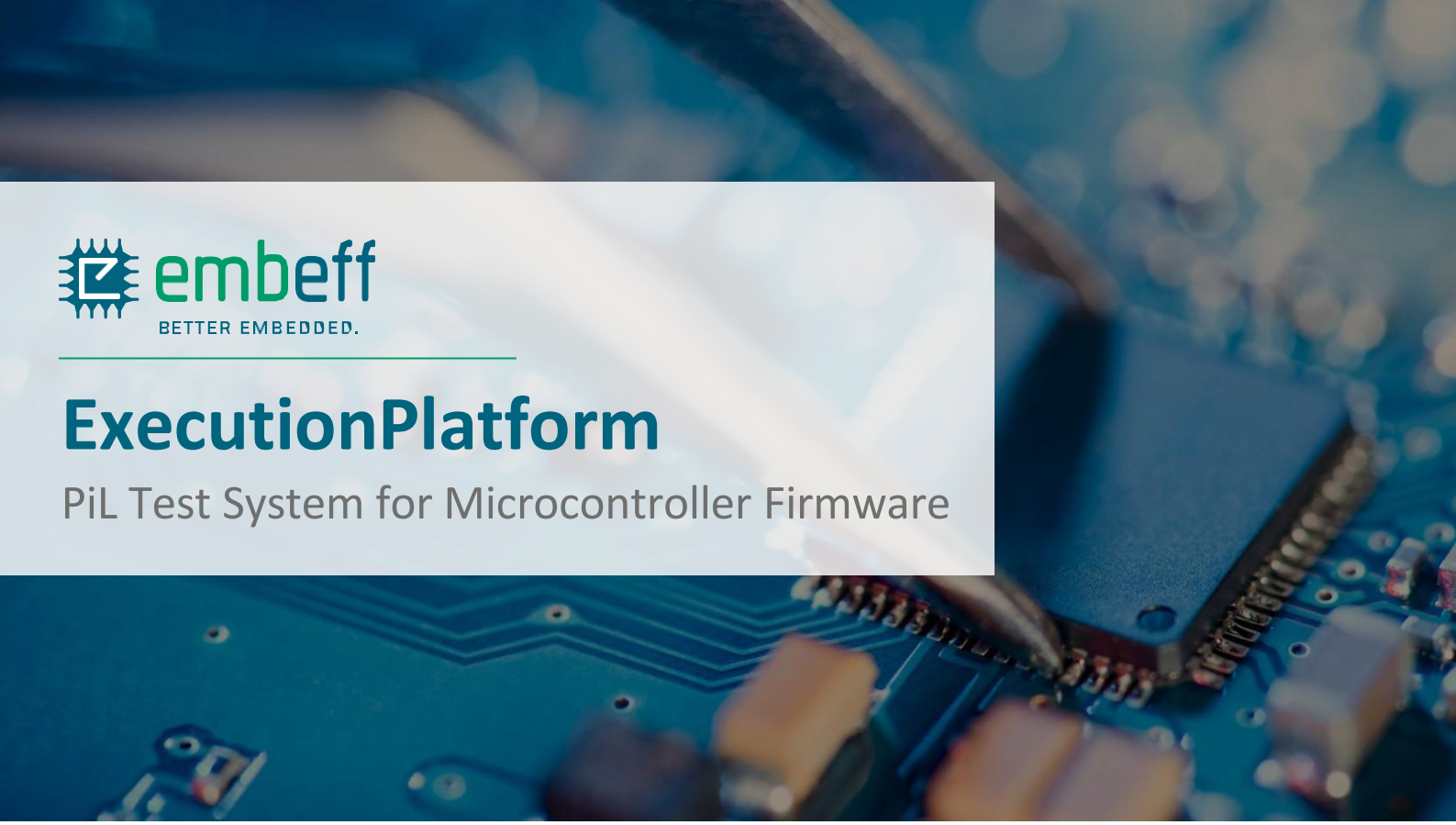


embeff
BETTER EMBEDDED.

ExecutionPlatform

PiL Test System for Microcontroller Firmware

Processor-in-the-Loop tests enable early firmware testing with HiL-level confidence at reduced cost and complexity.

Verify the physical pin behavior of your code at all times.

Implement and maintain tests within the development team.

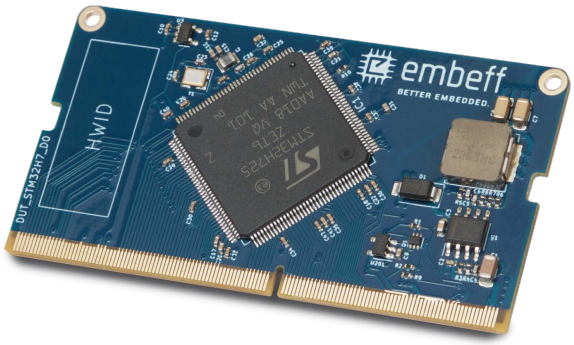
Unit test

HW/SW Integration test

System test

- Generic test system for all microcontrollers.
- Real-time simulation of the environment.
- 100% automated without wiring.

Your microcontroller as a test object



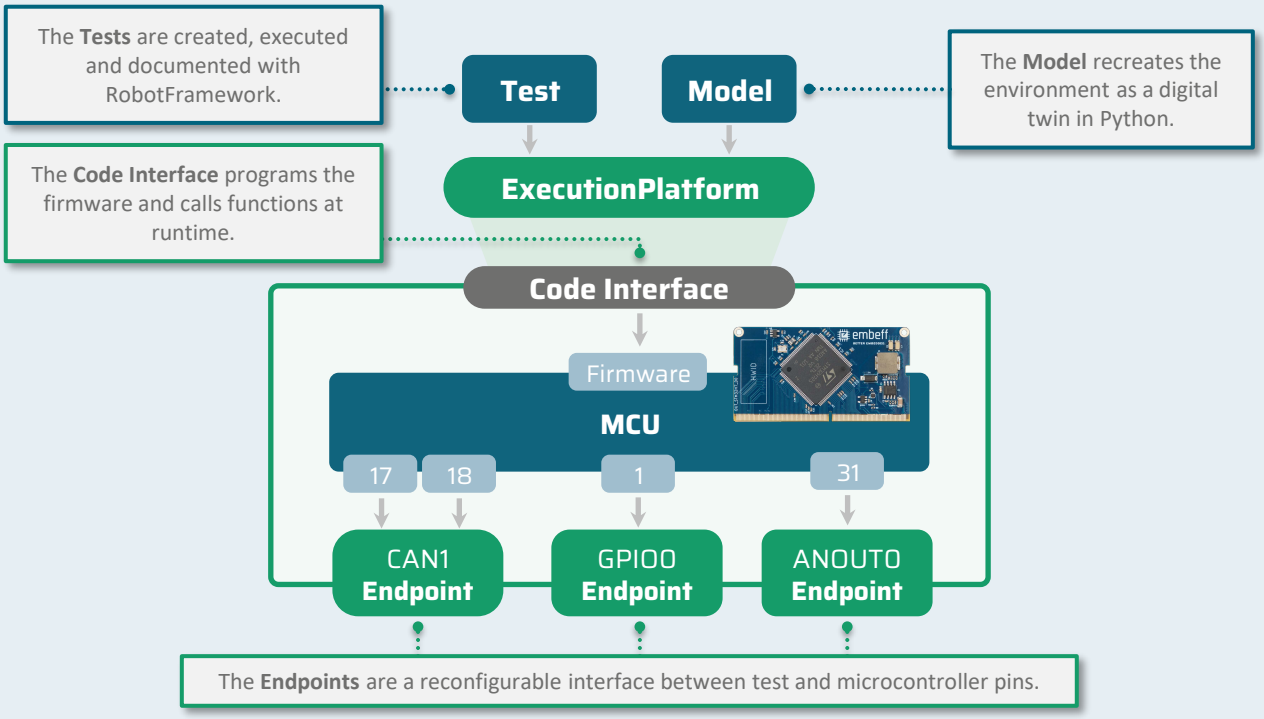
The ExecutionPlatform is a Processor-in-the-Loop (PiL) test system to which your microcontroller is connected as a plug-in board. This makes pins and programming interface accessible to the test system. You can replace the test board yourself at any time.

Based on your specifications, we create the schematic and take over design and production after your approval. Alternatively, you can use our template and manufacture the PCB by yourself.

The test system provides

- up to 144 digital I/O
- up to 24 analog outputs
- up to 8 analog inputs

A uniform structure for all tests



- **Test has access to all pins.**
- **Visualization of any pins with optional logic analyzer.**
- **Built-in SEGGER flasher for programming firmware.**

Unit test

Unit tests test individual functions, classes or modules.

With the **Code Interface**, you can easily flash and start your unit tests on the microcontroller. The results are conveniently prepared and evaluated on the PC.

Using a built-in benchmark function, you can ensure that the timing behavior meets your requirements.

gtest_suite1.bin

```
#include <gtest/gtest.h>
#include <SafeSensor.hpp>

TEST(SafeSensor, calculate_temp_0C) {
    ASSERT_EQ(SafeSensor::calc_from_r(1000),0);
}
```

- Ready-made support for GoogleTest, Unity, Catch2.
- No need to run on PC with another compiler.
- Permanently check timing behavior for critical paths.

Test sequence

```
*** Test Cases ***
Run Suite 1
    DUT Flash Firmware    gtest_suite1.bin
    Run Gtest Tests
```

Hardware/Software Integration test

test_can.bin

```
#include <ep/core.h>
#include "hal_can.hpp"

int32_t send_frame(uint8_t const* payload, int len) {
    can_msg msg;
    msg.id = 0x100;
    msg.dlc = len;
    for (int i=0;i<len;++i) { msg.data[i] = payload[i]; }
    return hal_can_send(msg) ? 0 : -1;
}
```

Test sequence

```
*** Test Cases ***
Send Frame Returns Error When Twice Not Acked
    CAN1 Start    Bitrate=500kbit/s
    CAN1 Cause Bus Error On Next Received Frame
    ...    Count=2    BusError=NoAck
    ${res} =    Dut Invoke    send_frame    abc
    Should Be Equal As Integers    ${res}    -1
```

- Reproducibility guaranteed through automation.
- Easily check error states.
- Prove coverage for drivers/HAL.

HW/SW integration tests check the interaction between hardware-dependent code (driver/HAL) and the physical pin level of the microcontroller.

Using **Endpoints**, you start processes on the pins or evaluate processes started by the microcontroller. This can be, for example, setting/reading a GPIO pin or sending/receiving a CAN frame.

Tests use the **Code Interface** to execute a function on the microcontroller and thus trigger the behavior to be tested on the pins. This behavior is then automatically checked using the appropriate endpoint.

Conversely, the code interface can also be used to check whether the microcontroller is responding correctly to pin behavior generated by an endpoint.

There are separate endpoints with extensive functions for each peripheral. The range of functions is designed so that all error scenarios can be triggered or tested with fault injection.

System test

Traditional system tests use complex HiL systems that test the product at the outer limits of the circuit board.

The ExecutionPlatforms PiL approach is an innovative and practical alternative. It does not require a finished circuit board so you can start testing in an early development phase.

A **Model** simulates the microcontroller's environment on the pins. **Endpoints** can trigger one-off or regular events on the pins. The model uses callbacks to respond to events initiated by the microcontroller.

pt1000_model.py

```
class SystemModel(config.Configuration):
    def __init__(self):
        self.outputs['FailureCount'] = 0
        self.simulated_temp = 20.0

    def input_changed(self, name, value):
        if name == "SimulatedTemp_C":
            self.simulated_temp = value
            self.update_pt1000_voltage()

    def update_pt1000_voltage(self):
        r_pt1000 = 1000 * (1 + 3.9083E-03 *
                   self.simulated_temp)
        i = 502e-6
        u = r_pt1000 * i
        self.ANOUT0.Set_Static(f"{u}V")
```

Test sequence

```
*** Test Cases ***
Simulate Change to 50° and Check Value In CAN Frame
    System Start Model    pt1000_model.py
    System Set Input    SimulatedTemp_C    ${50.0}
    Sleep    1.0s
    ${fames} = CAN1 Get Frames
    ${temp} =  Get Temp in Celsius From Frame  ${frames[1]}
    Should Be True    45 <= ${temp} <= 55
    Check That Failure LED on GPIO0 was never enabled
```

- System tests during development.
- Digital twin of the environment via model.
- Real-time capability guaranteed.

Your benefits

- Generic system replaces your time-consuming custom solutions.
- Flexible scaling with minimal investment costs.
- Excellent documentation and fast support.
- Gain the confidence of a HiL system without its cost and complexity.

Fast onboarding

Our **Training** courses provide practical knowledge about testing during development in microcontroller projects. This includes unit, integration and system tests and their specific significance in the embedded context.

All principles and techniques presented are generally valid and can be used as a starting point for your own test concepts. The exercises take place on real hardware, supported by the ExecutionPlatform as a test device.

After the training, participants have in-depth knowledge of test procedures for microcontroller projects and can independently integrate tests into their projects. You will learn how to use common test frameworks and develop an understanding of efficient testing procedures.

In our **Workshops**, we work together on concepts for testing your projects on the basis of these training courses.

Learn more

Videos

Training

Do you want to try out the ExecutionPlatform - directly and without installing anything?

Our browser-based demo at **embeff.com/ep-demo** offers many prepared examples.

embeff GmbH
Willy-Brandt-Allee 31 b · D-23554 Lübeck · Germany
Tel. +49 451 16088690 · Fax +49 451 16088699
info@embeff.com · www.embeff.com

embeff
BETTER EMBEDDED.