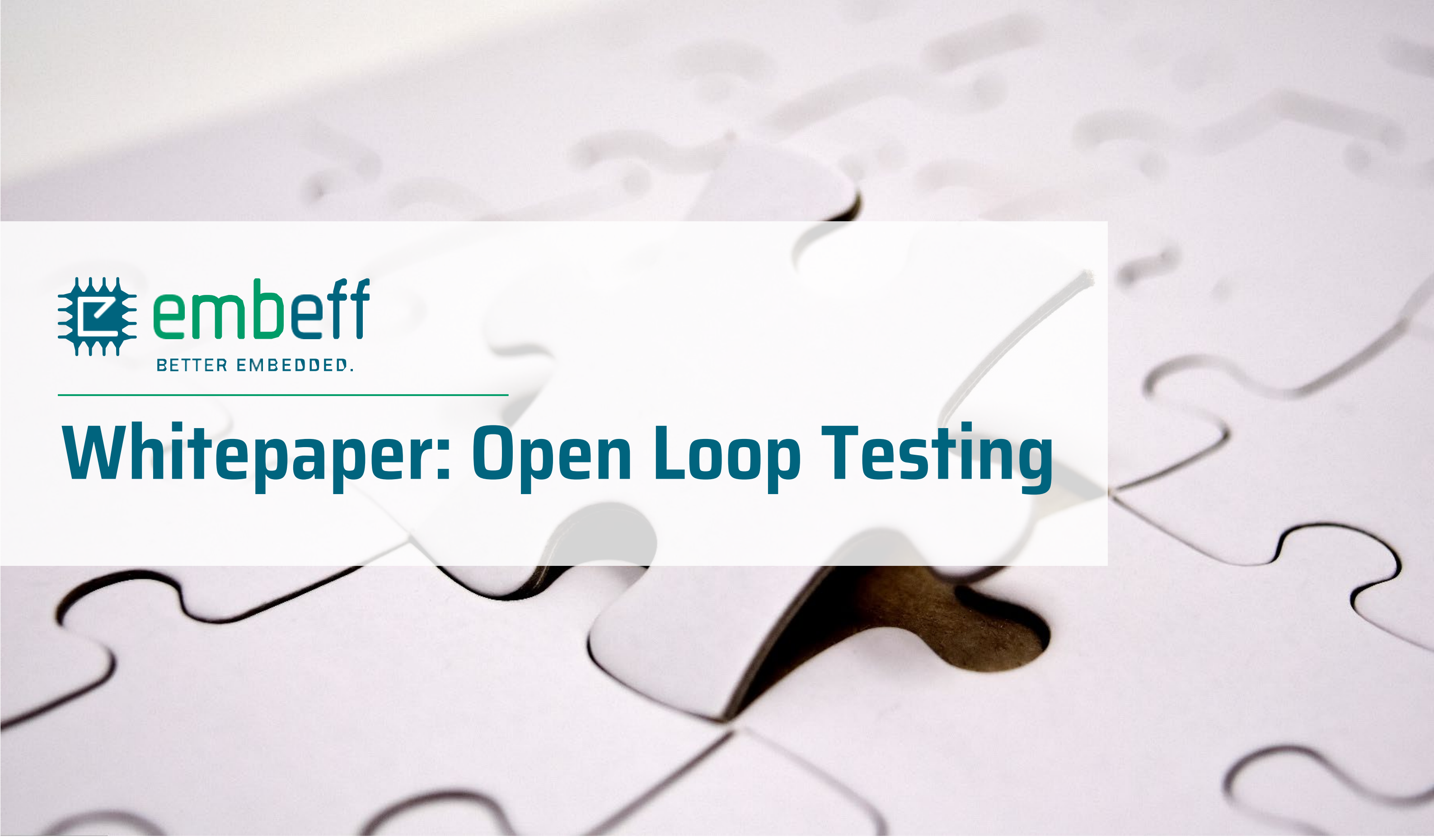


embeff
BETTER EMBEDDED.

# Whitepaper: Open Loop Testing

## Introduction

Testing embedded firmware is complex. Typically, most resources are spent on system testing. This whitepaper introduces a novel way to redirect some of these resources to early hardware-software integration tests with the aim to get more stable firmware.

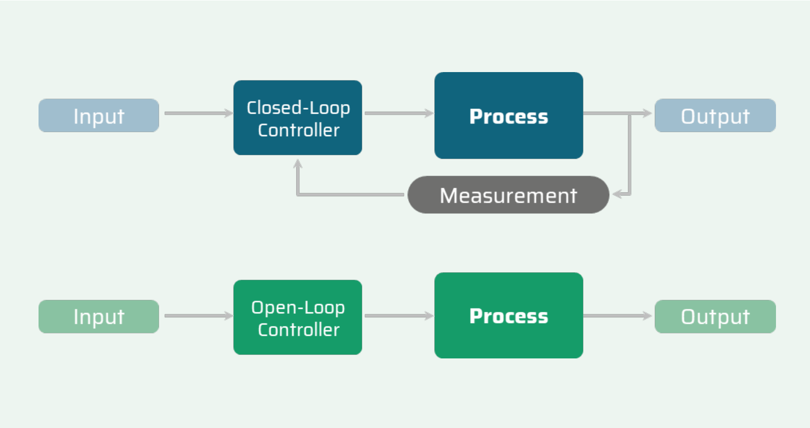
With embedded firmware, correct handling of the microcontroller peripherals is crucial to interact correctly with the environment. For example, the CAN periphery sends and receives frames. Driver code controls the peripherals. It is critical for robust firmware that this driver is correct and handles error situations properly.

The complexity of modern microcontrollers periphery is startling. Their documentation typically ranges in the thousands of pages. This complexity allows for powerful applications – but also gives much concern regarding the driver's code correctness.

**System testing covers drivers only indirectly. Open Loop Testing focuses solely on this important low-level code.**

## Background

The term **open loop** originates from the control engineering sphere. The purpose of control engineering is to create the desired state at the output, by influencing the actual process. **Closed loop** controllers measure the output constantly. The controller uses the discrepancy between the setpoint and the actual state to calculate a control action.




**Conventional system tests are structurally identical to closed loop controllers. This whitepaper explores, if we can apply the known advantages of open loop controllers to embedded testing.**

**Open loop controllers control a process blindly. They do not have to measure and consider the whole state of the system. This makes handling them much easier.**

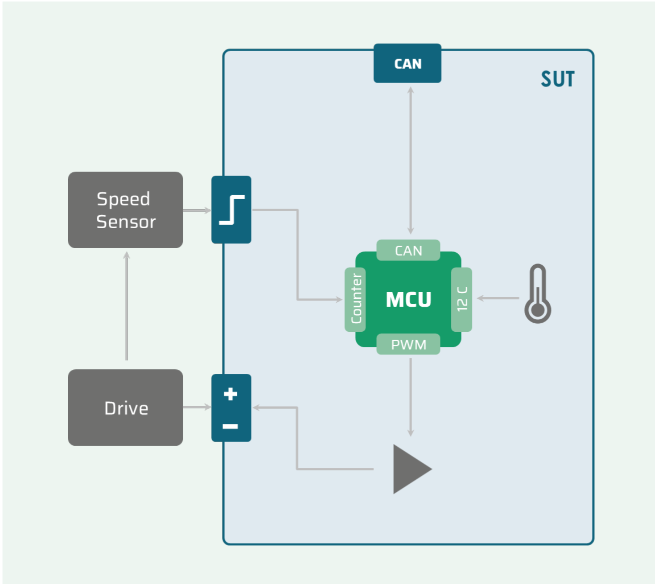
# Conventional System Testing

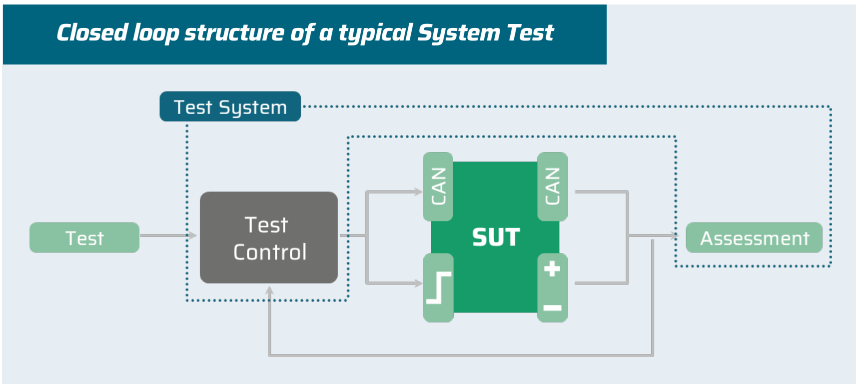
**Consider an embedded system for the speed control of a motor. This system:**

- is based on a microcontroller,
- receives via CAN messages with a setpoint speed,
- controls power electronics via PWM to control the external motor,
- receives impulses via a rotary encoder and calculates the current speed,
- contains an internal temperature sensor that switches the system off if it overheats.

We can formulate a typical requirement as follows:

***If a speed of 100/min is transmitted via a CAN message, the system must have reached a speed of between 98 and 102 rpm within 2 seconds.***




The test system must simulate an encoder signal for the embedded system (System Under Test, SUT). For this purpose, the test system must simulate the electronic behavior for the motor, based on the +/- signals. The SUT inputs are thus dependent on the SUT outputs. This **closed loop characterizes every system test**. Based on the measured outputs the system evaluates tests as a pass or fail.

# Closed Loop Limitations

1. If an error occurs, the cause cannot be determined. Potential causes are the embedded firmware, the hardware (board) or even a faulty simulation in the test system.
2. The test only checks the system for a single condition. For example, it cannot indicate that the PWM generation on the microcontroller is working correctly for all duty cycles.
3. The internal temperature sensor is not part of the test interface. To assess the temperature cut-off function, the tester must supplement the test stand with a heating system. But even then, it is not possible to check how the system reacts to something like a faulty sensor.
4. Every test stand is specific to its system under test. Reuse is limited.

We can trace back all these problems to one **fundamental limitation of closed loop testing**: the SUT has many internal state variables that are not visible externally. All the tests can do is attempt to produce as many of these internal states as possible through suitable selection of input variables. This is time-consuming and not comprehensive.

# Open Loop Testing

**Open Loop Tests focus on the interaction between software and microcontroller pins. They are characterised by three aspects:**

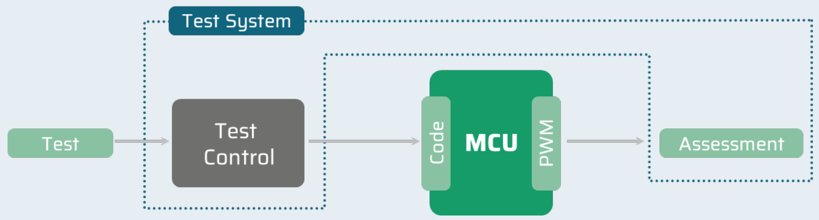
1. The Device Under Test (DUT) is the microcontroller.
2. Tests access microcontroller pins and code directly.
3. No simulation of the DUT environment.

The test objective shifts from making sure that the entire system works to ensure that single components work reliably in all circumstances. The example requirement used in System Testing is no longer testable. Instead, we deconstruct it to independent parts, such as:

***If the pulse driver is called with lengths between 50µs and 50ms, then the generated pulses must have a maximum tolerance of 1%.***

The microcontroller runs firmware that contains all the software functions to be tested in an unchanged form. The test system invokes and reads these functions. We can therefore consider Open Loop Testing as a special case of hardware-software integration testing.

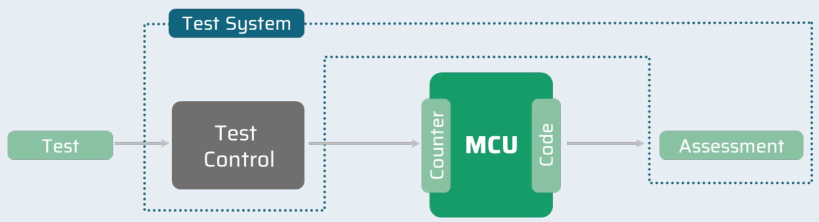
***Example 1: Open Loop Test for a PWM driver***




The PWM pin is a microcontroller output. The purpose of this test is to check that the PWM driver is functioning correctly. The test system calls the PWM driver code with a duty cycle. The microcontroller outputs a PWM signal at a pin. The test then checks whether this signal has the desired duty cycle and evaluates it as a pass or a fail.

**The open loop structure is apparent: there is no loopback from the outputs to the inputs of the microcontroller.**

***Example 2: Open Loop Test for a Counter Driver***

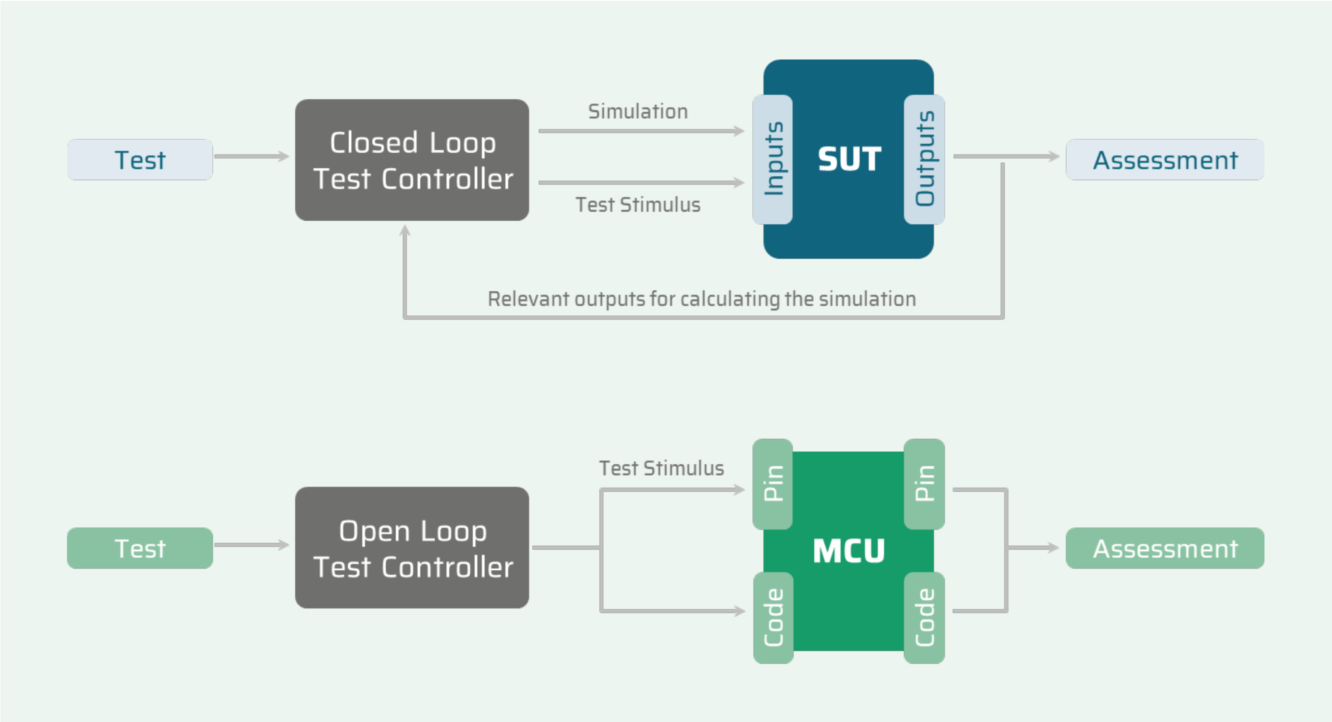



The Counter pin is a microcontroller input. The test system generates an electrical signal. This signal reproduces the impulses from the encoder. The software measures a speed. The test reads back the result from the microcontroller.

# Open Loop Advantages

1. By focusing on one component, the complexity of each test case is low. The tests are compact and easy to read. If a test fails, it reveals the cause immediately.
2. Each component can be systematically tested for every imaginable input value.
3. Internal error states (that are not attainable in a system test) can be tested. For example, the behavior of the drivers can be tested in case of an overload of the CAN bus or a temporary electrical disturbance of the I2C communication.
4. Tests are performed at the pin level of the microcontroller. This pin behavior is standardized for peripherals.
5. The same test system can be reused for different DUTs.

## Comparison: System vs. Open Loop Testing




| Feature | System Test | Open Loop Test |
|---|---|---|
| Runs on real microcontroller | ✓ | ✓ |
| Tests system as a whole | ✓ | - |
| Accesses internal states | - | ✓ |
| Independent from PCB | - | ✓ |
| Tests single functions | - | ✓ |
| Reusable | - | ✓ |
| Costs | **High** | **Low** |

## Summary

System tests must always provide both a simulation of the environment and the test stimulation. This requires a closed loop structure.

Open Loop Tests do not need a simulation. The test stimulus specifies the pin behavior or calls functions in the code.

The open loop principle presented here enables the formulation of HW-SW integration tests. These tests can check components of the system under **every conceivable condition.** Such testing is not economically viable with system tests.

The embeff ExecutionPlatform is a commercial Open Loop test system with support for all microcontroller types. Explore practical examples and learn more at **https://embeff.com**.

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.