University of Applied Sciences and Arts of Southern Switzerland
Department of Innovative Technologies
Institute of Systems and Applied Electronics

SUPSI

From Simulation to Deployment:

The Evolution of Real-Time Validation for Embedded Systems

Dr. Mikael Bianchi
Senior Researcher
SUPSI

26.08.2026

Contents

1. University of Applied Sciences and Arts of Southern Switzerland (SUPSI).

- Brief description
- Control applications for embedded systems at the institute of systems and applied electronics at SUPSI

2. Real-Time Validation for Embedded System Platforms for the real time embedded control.

- Motivation
- Validation and Verification typologies: SIL - RCP - PIL - HIL
- Platforms and solutions
  → **How far can we go with low-cost solutions?**
- Examples

University of Applied Sciences of Southern Switzerland (SUPSI)

University of Applied Sciences of Southern Switzerland (SUPSI)

- One of the nine **professional Universities** recognized in Switzerland
- It offers more than **30 courses of Bachelor, Master and Diploma**, characterized by cutting edge education, which combines the traditional theoretical-scientific instruction with a professional approach
- Great care is given to **research**, carried out in key sectors through competitive projects acquired by mandate from enterprises or institutions or with National and European programs
- It is divided into:
  - **4 departments**:
    - Department for Environment, Constructions and Design,
    - Department of Business Economics, Health and Social Care,
    - Department of Education and Learning,
    - **Department of Innovative Technologies**
  - **3 associated schools**:
    - Accademia Teatro Dimitri,
    - Conservatorio della Svizzera italiana,
    - Fernfachhochschule Schweiz

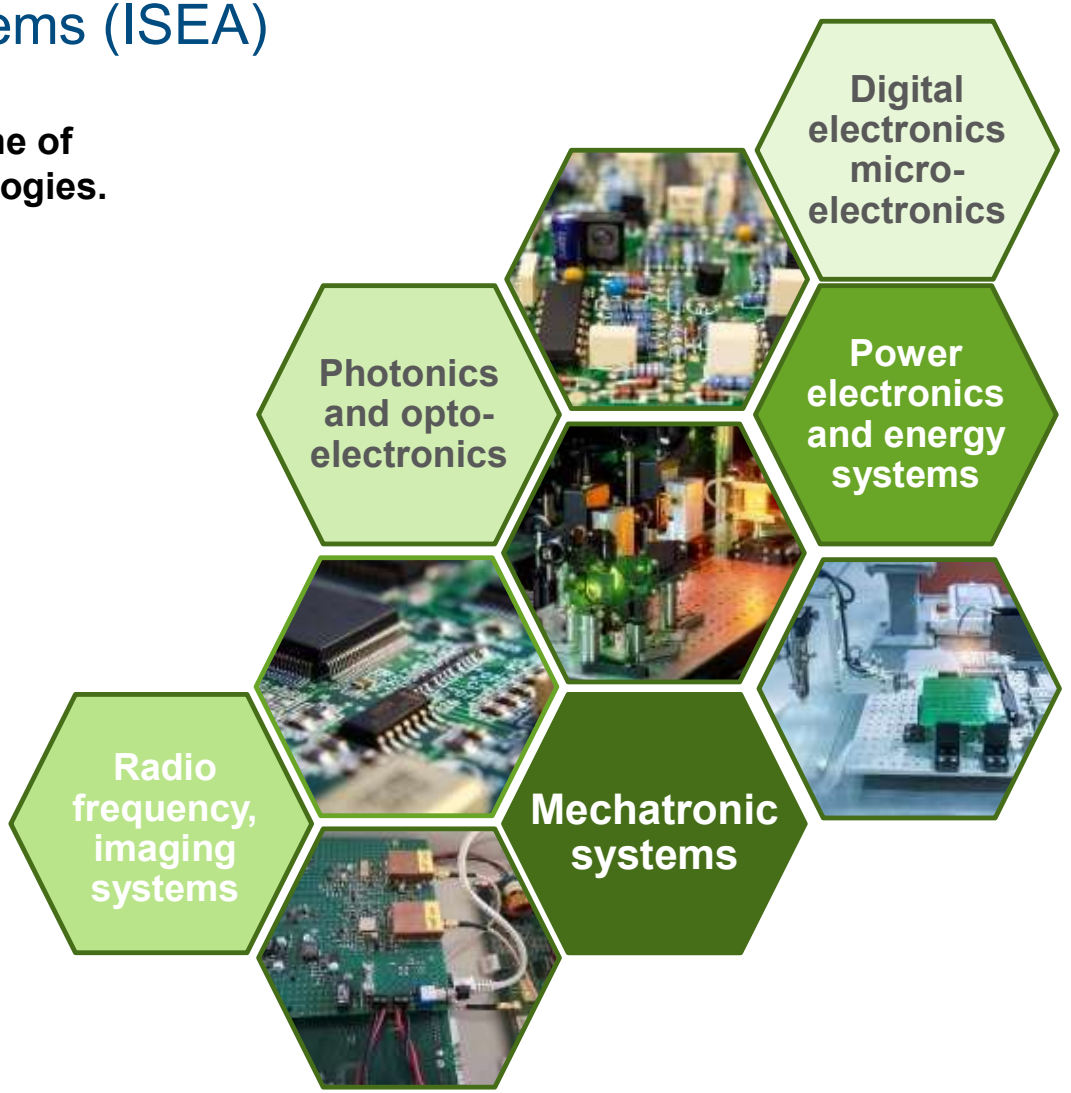
Institute of Applied Electronics and Systems (ISEA)

Institute of Applied Electronics and Systems (ISEA) is one of the six institutes at the department of innovative technologies.

→ Thanks to its in-depth specialist expertise, it offers **competences in 5 specific scientific areas**.

With its work, ISEA carries out two main mandates:

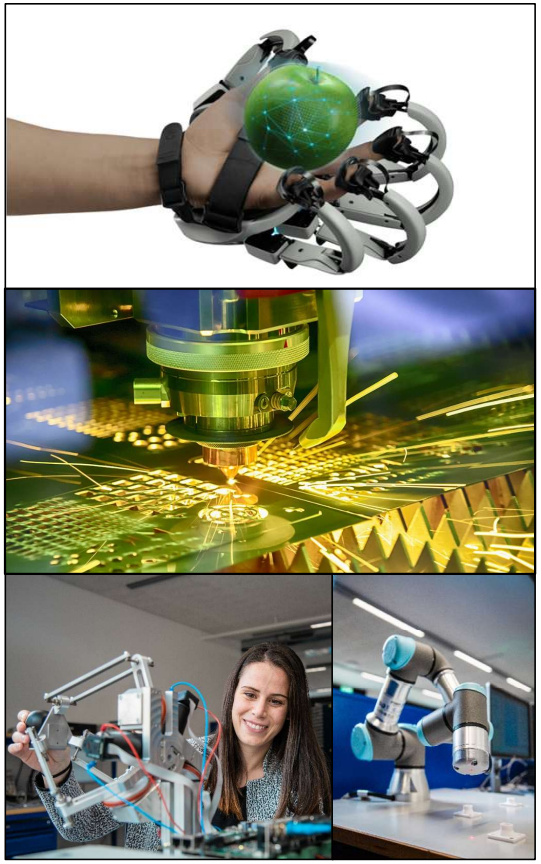
- Bachelor course in Electronics
  - → Education of electronic engineers of the future
- Applied research with industry, public bodies and other academic institutes
  - → **Innovation** in companies through development of projects.
  - → **Knowledge** transfer toward industries.



Mechatronics for high precision and dynamics

The group is involved in research & education in mechatronics & control

- **Main focus area**
  - Simulation of physical processes
  - Modeling, control and identification of mechatronic systems
  - Development of electronic boards
  - Implementation of algorithms in microprocessors
- **Specific research fields / competences**
  - Process control for high precision applications
  - Low to high level implementation and development of industrial comunication protocols
  - Rapid Control prototyping and real time systems
  - Sensors (Encoders,..) and actuators (electromagnetic devices)
  - Design, model and control of electrical machines
  - Computer vision for industrial processes



Real-Time Validation for Embedded Systems

Motivation

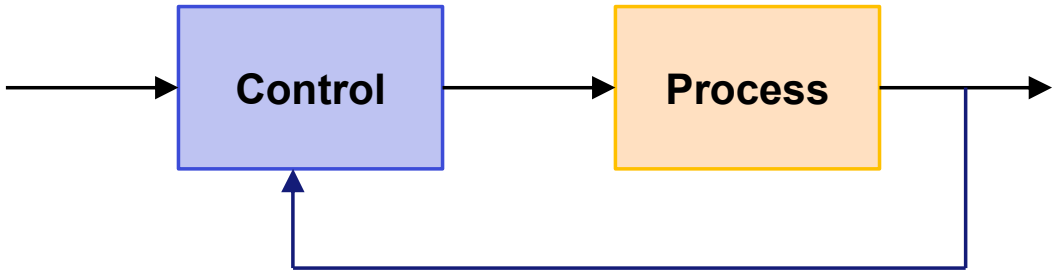
Embedded systems **often operate as closed-loop systems**, requiring control actions to be computed at high frequencies while minimizing dead times.

- Processor-based control loops can operate at frequencies of up to **hundreds of kHz**, with **jitter and dead times in the sub-microsecond range**.

A **model-based approach** supports the design and development of control systems by providing a **physical representation** of the process.

- The physical model can be used to validate the expected system behavior and to **verify the control algorithm through simulation**.

BUT: simulation ≠ real hardware ≠ real-time constraints:
How can we reliably validate and deploy a control system?

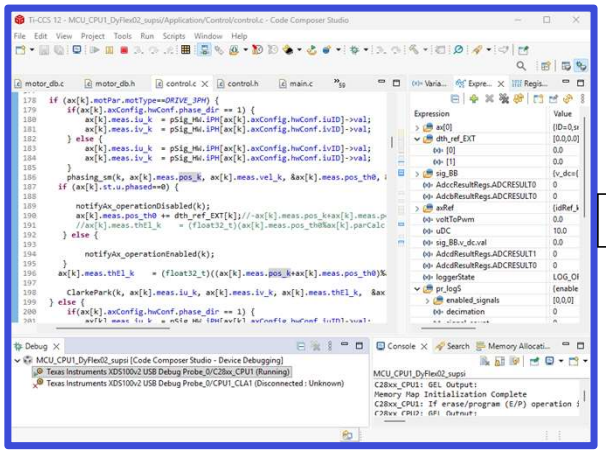


Concrete example: control of a self-balancing electric vehicle

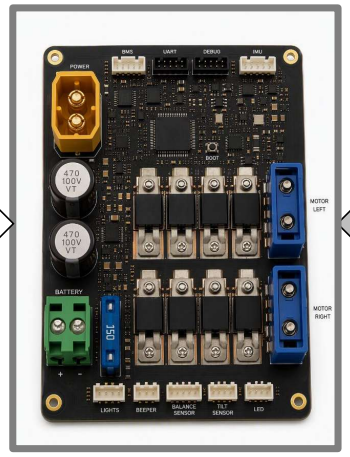
Objective: evaluate, design and develop a control system for a self-balancing vehicle.

The system consists of three main components:

- **Electromechanical system**, including drive motors, inertial and current sensors, and the mechanical structure.
- **Actuation board,** including the sensor interfaces and the power stage (inverter), required to control the motors.
- **Firmware** deployed in the microcontroller of the actuation board, implementing the control algorithms and managing all system peripherals.



Control Algorithm & FW



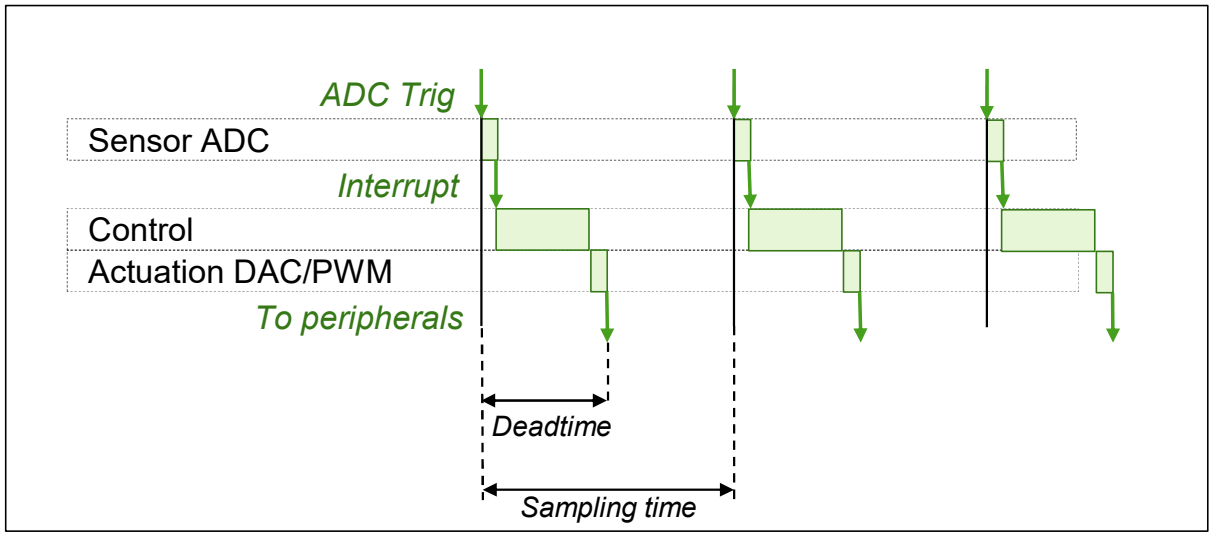
Actuation / sensor board

Process

Concrete example: control of a self-balancing electric vehicle

Embedded Firmware implements the **control algorithm** and is subject to stringent real-time requirements. Typical requirements for motor control include:

- **Sampling time:** 100 μs, with deterministic, interrupt-based execution
- **Control algorithm execution time:** < 50 μs
- **Jitter and latency**: < 100 ns
- **Measurement-to-actuation deadtime**: < 50 μs



Can we meet these requirements with the final hardware?

AND:

Can we meet these requirement in a validation environment?

AND:

Can we do it using low-cost, off-the shelf targets?

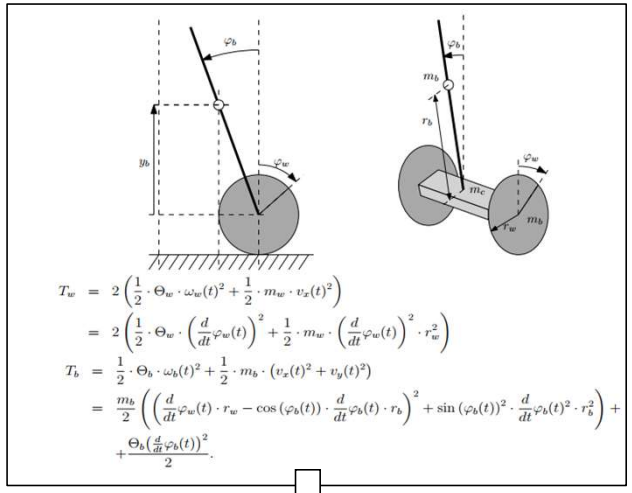
Development steps:

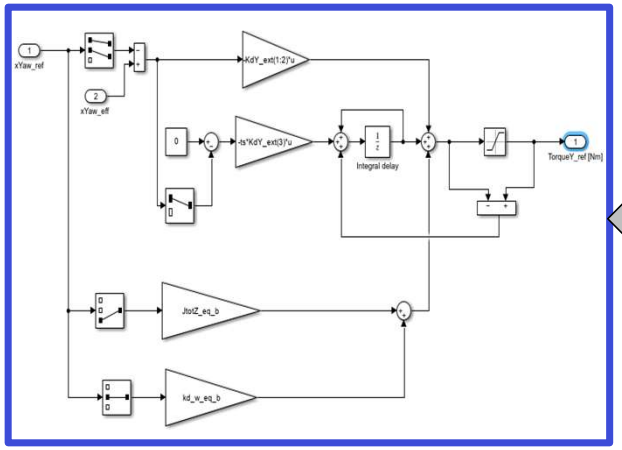
1. Modeling and simulation: Simulation in the loop (SIL)

No Hardware required, only a PC with simulation environment.

Both the control algorithm and the process are simulated: simulations can be performed faster than real time.

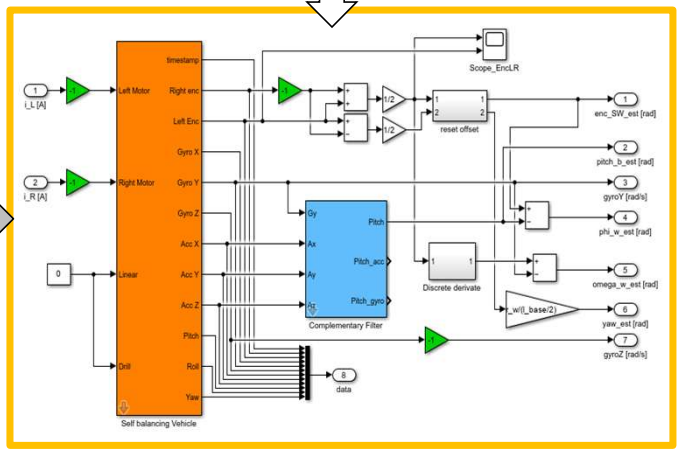
Objective: verify the control concept in simulation and the process behavior in simulation.





Control Algorithm

No Hardware



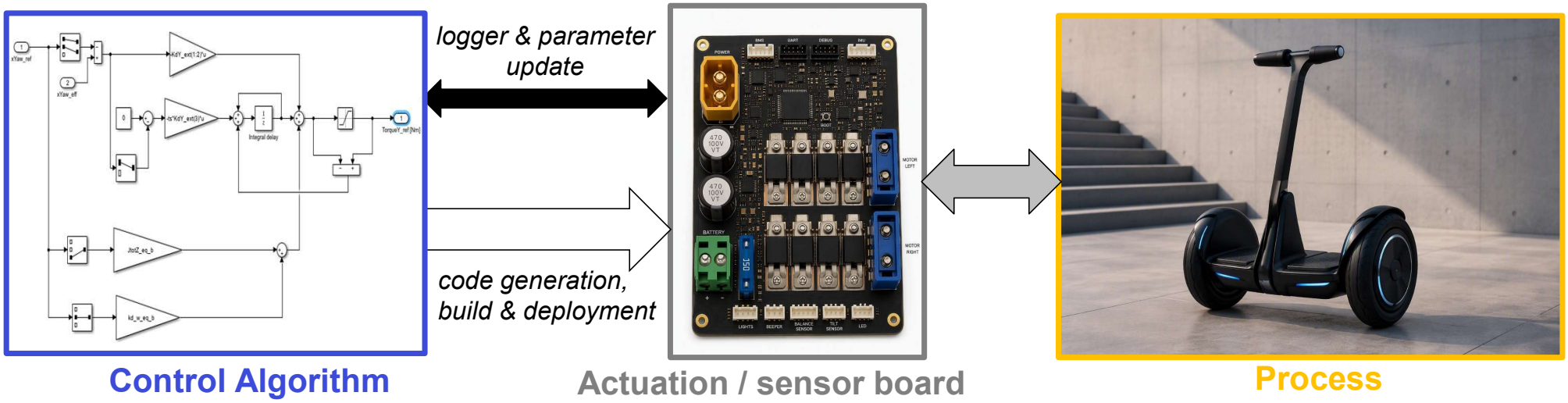
Process

Development steps:

2. Control verification in the real process: Rapid Control Prototyping (RCP)

Before starting product development and industrialization, Rapid Control Prototyping can be used to verify the feasibility of a concept before its implementation on a real-time embedded system.

- Early verification can accelerate feasibility assessment and help identify potential challenges at an early stage.

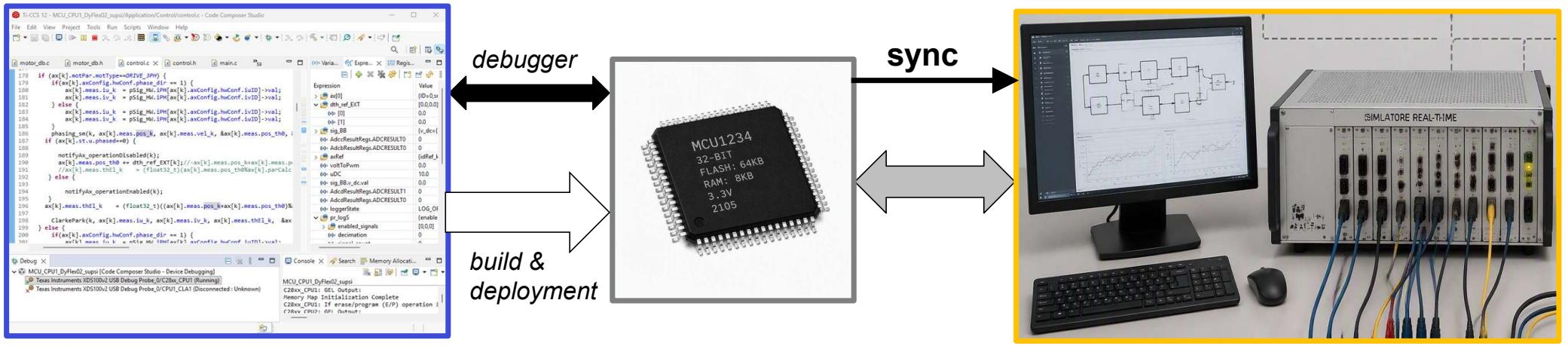


Development steps:

3. Code verification in the processor: Processor In the Loop (PIL)

Before testing the developed Firmware in the final board and real process, the code can be verified in simulated environments, logging the signals at frequencies suitable for the intended process.

- Verification in simulated or safe environments can avoid damage or injuries in critical processes and power boards.

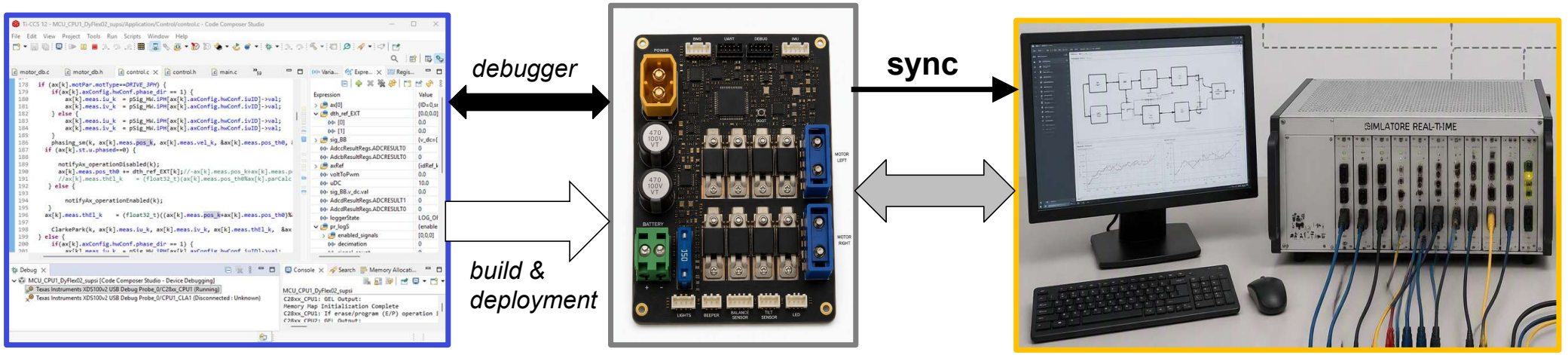


Development steps:

3. Verification of the embedded system: Processor In the Loop (HIL)

Finally, the embedded controller is connected to an emulated or simulated process running in real time.

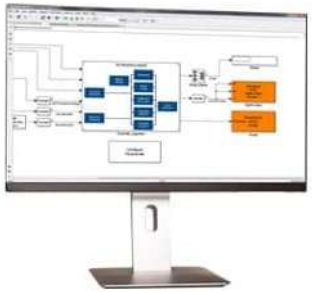
- The controller therefore behaves as if it were connected to the real system, while the physical process is replaced by a real-time model.This allows us to test the complete embedded system under controlled and safe conditions.



Solutions for embedded real-time verification using RCP - PIL - HIL

Proprietary platform-based solutions ($$$)

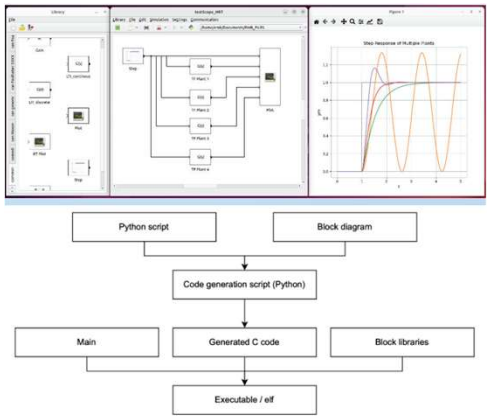
- Complete solutions for the real-time validation of embedded systems that provide customers with advanced functionalities and reduce engineering effort.
- Examples: SpeedGoat, Typhoon HIL, dSpace, NI, Opal-RT, Beckhoff, B&R.
- Generally, the most expensive but robust solution.

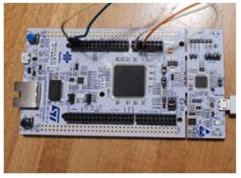


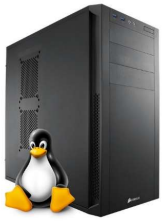
Generic target solutions ($ - $$)

- **Microcontroller boards** (Texas, Microchip, STM,..), Linux RT_PREEMPT based systems (Raspberry, PC,..) can be used
- Variety of developing and fast code generation SW can be used: Matlab ($$), Scilab ($$-$) and **PySimCoder** ($)

Very interesting solution in terms of quality/price!







Generic targets: low-cost does not necessarily mean low performance.

Microcontroller boards (Texas, Microchip, STM32,..)

- Can achieve **high performance** thanks to direct access to peripherals and powerful microprocessors. Interrupt-based scheduling enables **"hard real-time" behavior**.
- Require in-depth knowledge of the processor and its peripherals to fully exploit the available performance.
- Typical timing performance:
  - High sampling frequencies: 10 - 50 kHz
  - Low latency and jitter: down to ~10 ns
  - Reasonable data-logging capability:
    - With simple serial communication, theoretically one 2-byte variable can be transmitted every 5 µs.
    - In practice, a few variables can be logged at 10kHz.

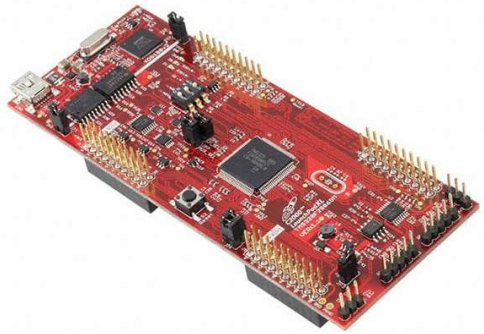
Linux PREEMPT_RT based systems (Raspberry, PC,..)

- Powerful processors enable Linux PREEMPT_RT based systems to run complex models, but higher jitter and latency make them suitable for “soft real-time” applications.
- PC's, Low-cost embedded platforms such as Raspberry Pi and BeagleBone, and industrial-grade platforms such as Compulab and Toradex can be used as real-time targets.
- Typical timing performance:
  - Good sampling frequencies: 1 - 5 kHz
  - Higher latency and jitter due to OS & peripherals access:
    - Depends on peripheral implementation.
    - Typical measured values: ~50 µs
  - Good logging possibility: 1 - 5kHz

Implementation example

Launchpad Texas:

- Dualcore 200MHz
- 32-bit floating point precision
- 10-50kHz real-time control



USB Cable
FW programming via RCP
Logging with RS232, 5Mbit/s, soft real-time

PC

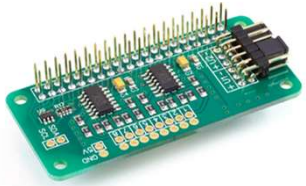
- SW: Matlab / SciLab / PySimCoder.
- Fast algorithm implementation using schematic diagrams.
- Dedicated peripheral implementation to achieve hard real-time operation (on microcontroller).
- Fast serial communication for logging and parameter update.

HW

Inverter 3ph

Encoders SinCos+ABZ

ADC / DAC / GPIO / SPI / ...



Implemented examples using low-cost off-the-shelf targets

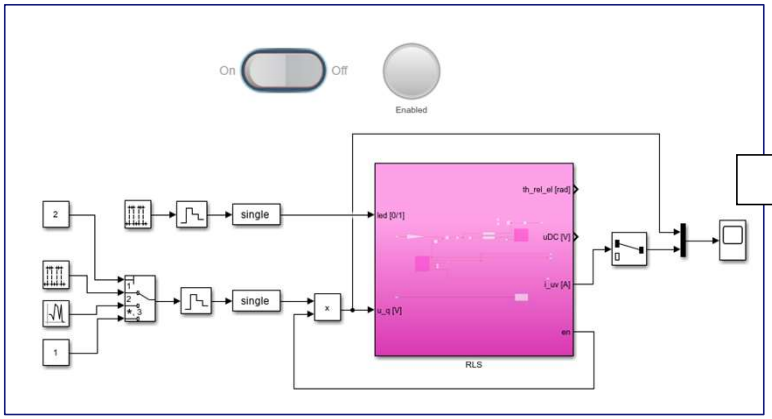
Examples: sensorless position detection of an electromagnet

Context: Student work for electromechanical characterization of electromagnets.

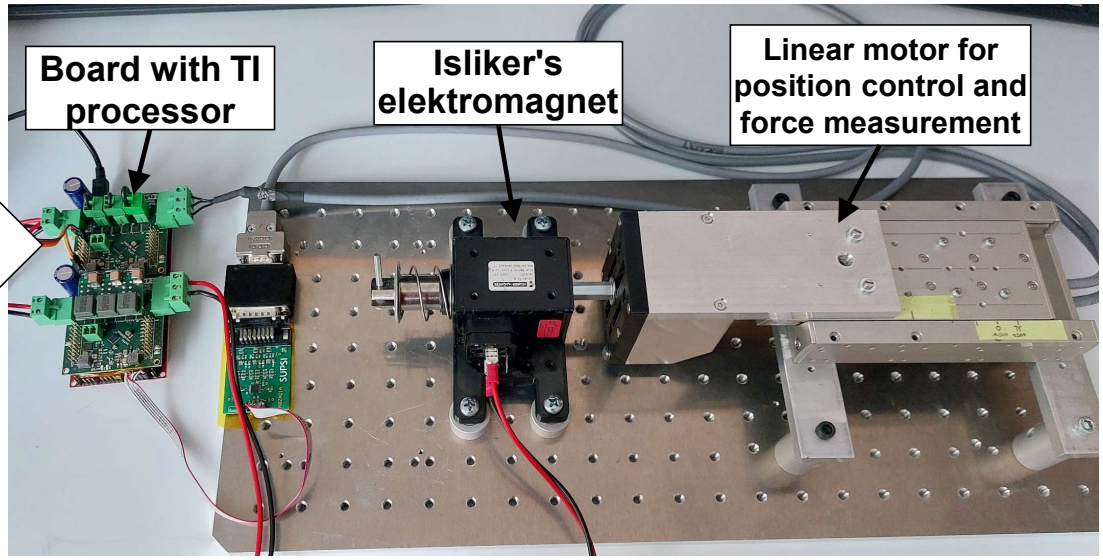
Objectives: Characterization and online sensorless identification of position for Isliker's electromagnets.

Work: Implementation of the recursive identification algorithm (recursive prediction error method) using Matlab/Simulink + C2000 Blockset. Implementation of the position control for the three-phase motor.

Comments: Peripheral preparation is relatively straightforward (assuming good knowledge of the C2000 processor) and strict timings can be met. Logging is the main bottleneck (10kHz for a few variables).



RCP



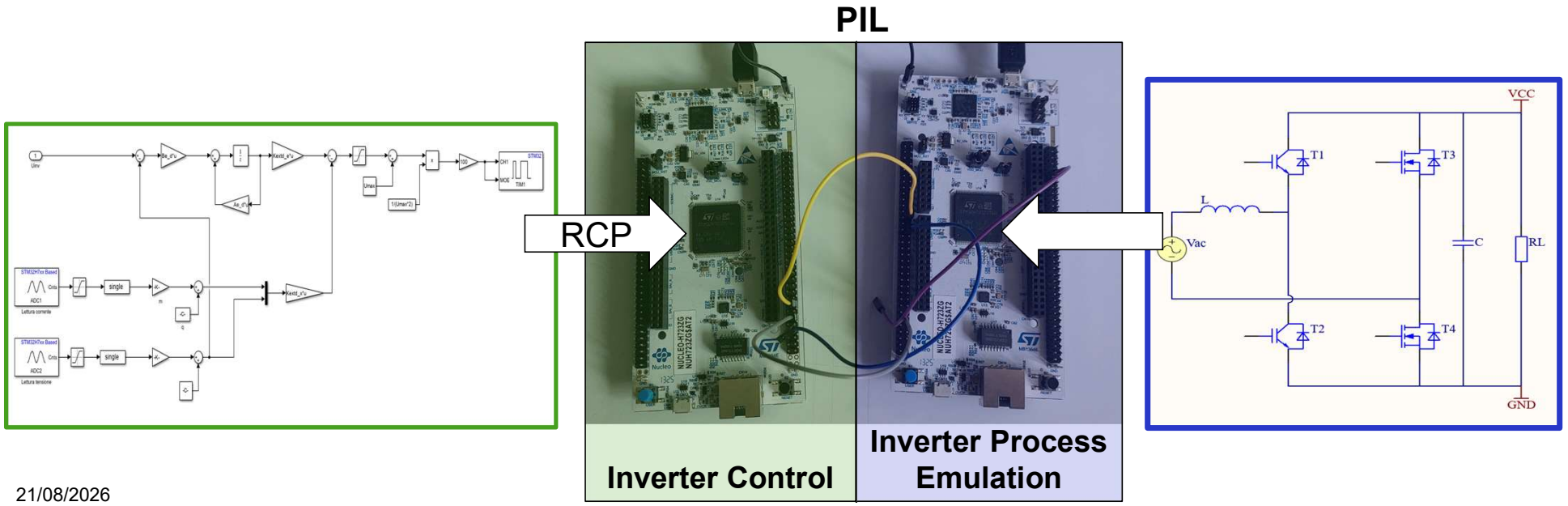
Examples: RCP and HIL of an UPS system

Context: Student work for feasibility of combined PIL - RCP in switching systems, using generic HW.

Objectives: Implement the process of an UPS (inverter-rectifier) in a Processor In the Loop system, as well as the control algorithm in a processor, using Rapid-Control-Prototyping methodology.

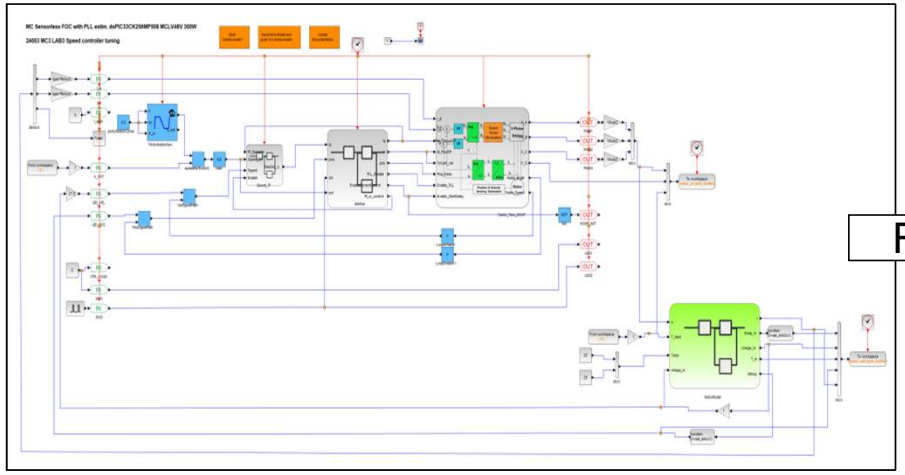
Work: Modeling, control and peripheral implementation & synchronization, using Matlab/Simulink + STM32 Blockset.

Comments: Synchronization strategy is more complex, as controller requires the emulated data at very specific timings, high-frequency switching (typical 10-20kHz) makes the preparation in PIL/HIL system difficult

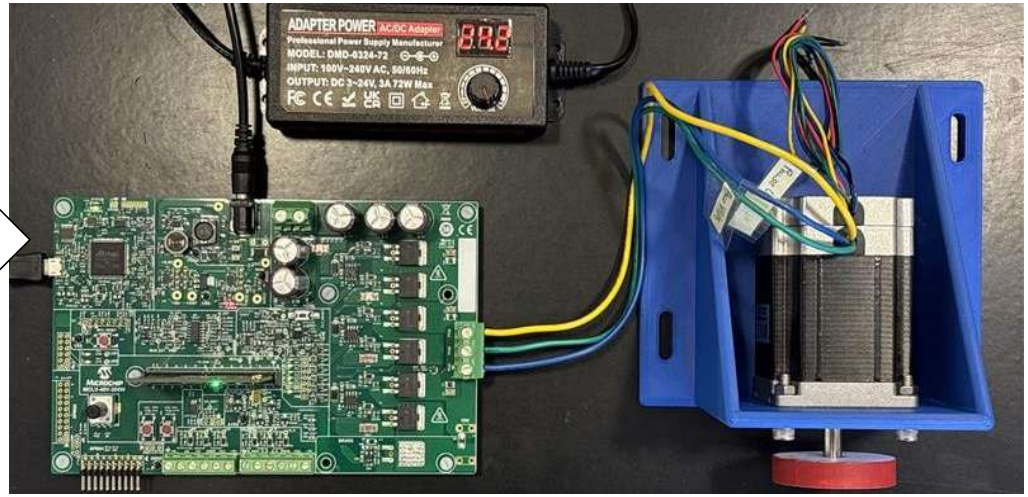


Examples: Active Noise Cancellation (ANC) in a PMSM

- **Context:** Student work to evaluate Microchip's rapid control prototyping solution and libraries using Scilab.
- **Objective:** Study the feasibility of the ANC and implement a new algorithm for electrical motors using Scilab/XCos and Microchip controller.
- **Work:** Modeling, and algorithm implementation using Scilab/XCos/X2C.



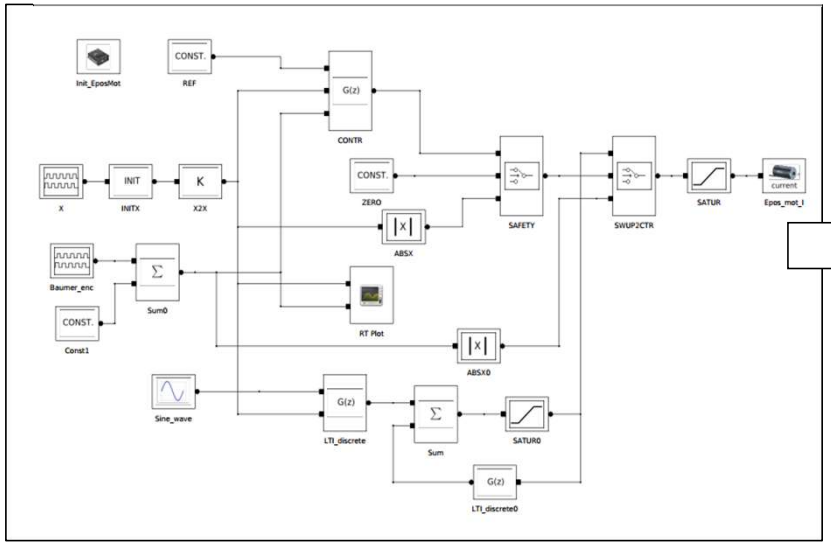
RCP



Examples: control of an inverted pendulum

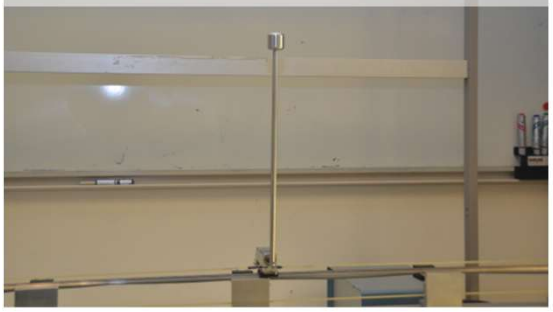
- **Context:** Laboratory experiences using pySimCoder, to be conducted during courses.
- **Objective**: Prepare easy-to-deploy experiences for students in control & automation.

Python / pyControl / pySimCoder



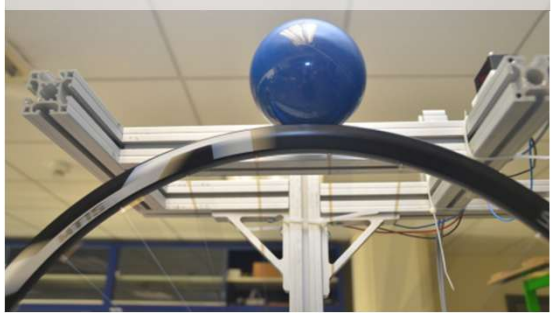
RCP

Inverted pendulum



Ball on plate

Ball on wheel



What did we learn?

What did we learn?

1. Low-cost RCP can achieve hard real-time control.
2. Logging is often the real bottleneck.
3. High-frequency switching makes synchronization challenging.
4. Processor/peripheral knowledge is essential.
5. The same approach can support education, applied research and product development.

Thank you!

Questions?

Contact details: Dr. Mikael Bianchi
T +41 (0)58 666 65 32
mikael.bianchi@supsi.ch

Polo universitario Lugano
Via Serafino Balestra 16. CH-6900 Lugano