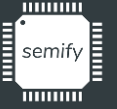


solutions to simplify.

# Smart Sensor Emulation for FPGA based Systems

## An efficient way of Developing and Testing Embedded Systems using SmartWave



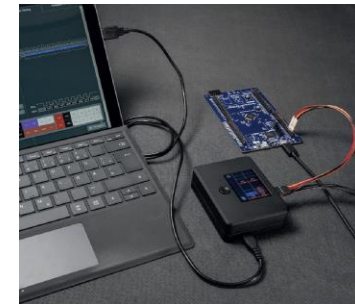
- ✓ SmartWave Introduction
- ✓ Sensor (Target Device) Emulation
- ✓ Demo for Temperature Sensor Emulation
- ✓ Demo for IMU (inertial measurement unit) Sensor Emulation

## Design & Development

- ✓ Digital Design for FPGAs and ASICs
- ✓ FPGA based Prototyping
- ✓ Embedded System Development

## Product Development

- ✓ SmartWave - enabling instant communication to any electronic device within embedded systems



## Instant connection for testing and emulation

Instantly communicate with multitude of devices like ADCs, DACs, Sensors...

Handle standard interfaces such as SPI, I2C or UART and flexible GPIOs.

Use the adaptable and powerful configuration possibilities via WebGUI.

Emulate sensors and actuators for validation and testing using SmartWave Python API.

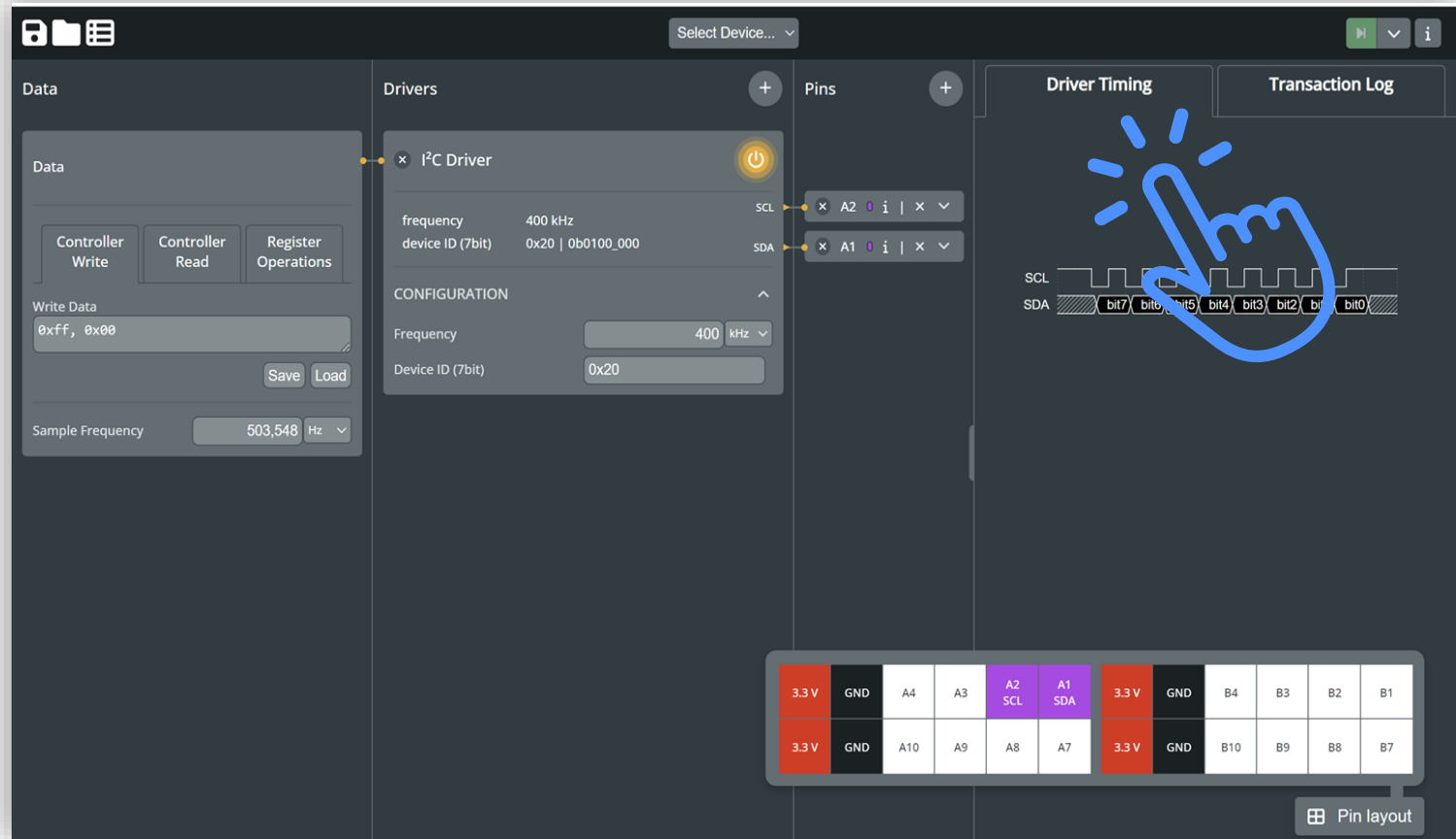


Just one click away ...

Use SmartWave without any software installation

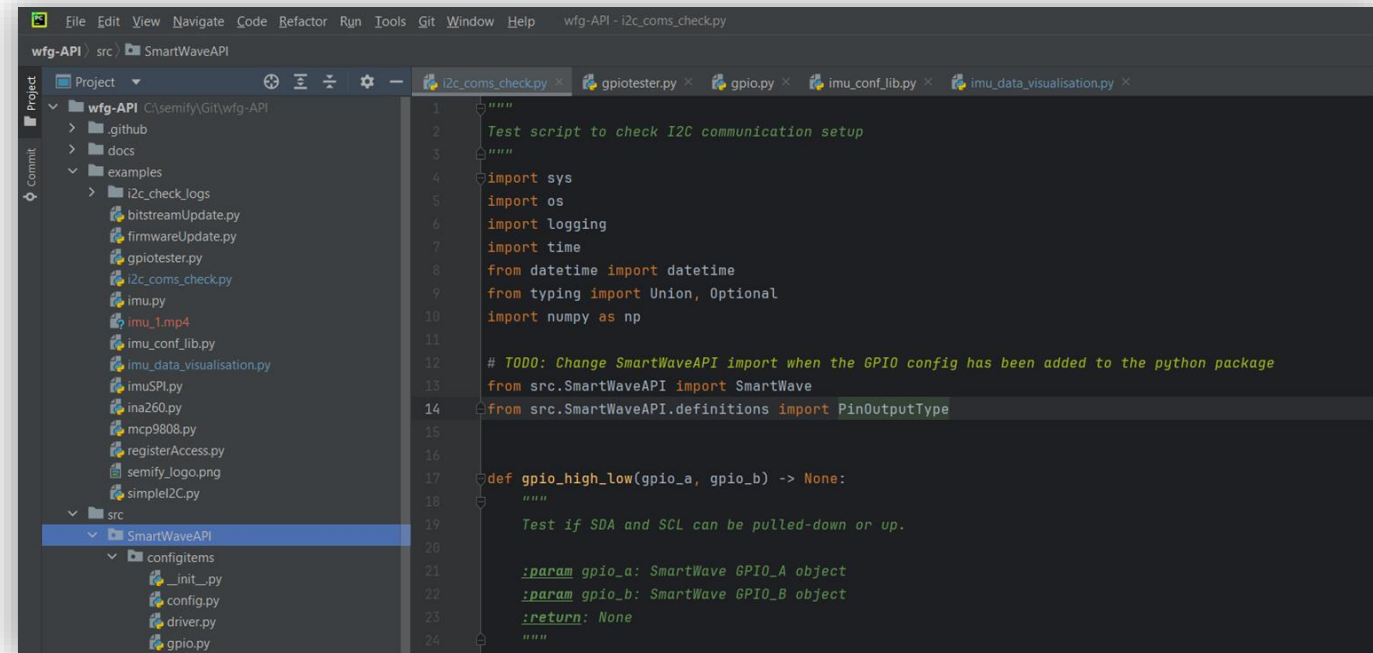
Intuitive and powerful User Interface

Supports easy collaboration within teams



## Unlock your test & debugging possibilities with SmartWave API

- ✓ Easy access to SmartWave using Python
- ✓ Automate repetitive tasks
- ✓ pip install SmartWaveAPI
- ✓ Can be used without USB Driver installation



```
File Edit View Navigate Code Refactor Run Tools Git Window Help wfg-API - i2c_comms_check.py
wfg-API src SmartWaveAPI
Project
  wfg-API C:\semify\Git\wfg-API
  .github
  docs
  examples
  i2c_check_logs
  bitstreamUpdate.py
  firmwareUpdate.py
  gpiotester.py
  i2c_comms_check.py
  imu.py
  imu_1.mp4
  imu_conf_lib.py
  imu_data_visualisation.py
  imuSPI.py
  ina260.py
  mcp9808.py
  registerAccess.py
  semify_logo.png
  simpleI2C.py
  src
    SmartWaveAPI
      configitems
        _init_.py
        config.py
        driver.py
        gpio.py
  i2c_comms_check.py x gpiotester.py x gpio.py x imu_conf_lib.py x imu_data_visualisation.py x
1 """
2 Test script to check I2C communication setup
3 """
4 import sys
5 import os
6 import logging
7 import time
8 from datetime import datetime
9 from typing import Union, Optional
10 import numpy as np
11
12 # TODO: Change SmartWaveAPI import when the GPIO config has been added to the python package
13 from src.SmartWaveAPI import SmartWave
14 from src.SmartWaveAPI.definitions import PinOutputType
15
16
17 def gpio_high_low(gpio_a, gpio_b) -> None:
18     """
19     Test if SDA and SCL can be pulled-down or up.
20
21     :param gpio_a: SmartWave GPIO_A object
22     :param gpio_b: SmartWave GPIO_B object
23     :return: None
24     """
```

# SmartWave - Sensor (Target) Device Emulation



## Using SmartWave API and configurable sensor emulation module

### Emulate all types of target devices

Generic Register File enables flexible target device modeling

### Protocol decoder

Configurable protocol decoder allows to emulate target device interfaces

### Full Flexibility

Offering a RISC-V (CV32E40X) exclusively available for target device emulation



Powered by **CV32E40X**  
Small and efficient, 32-bit, in-order RISC-V core with a 4-stage pipeline supporting general purpose extension interface



**OPENHW** GROUP  
— PROVEN PROCESSOR IP —



## Reasons for doing sensor emulation

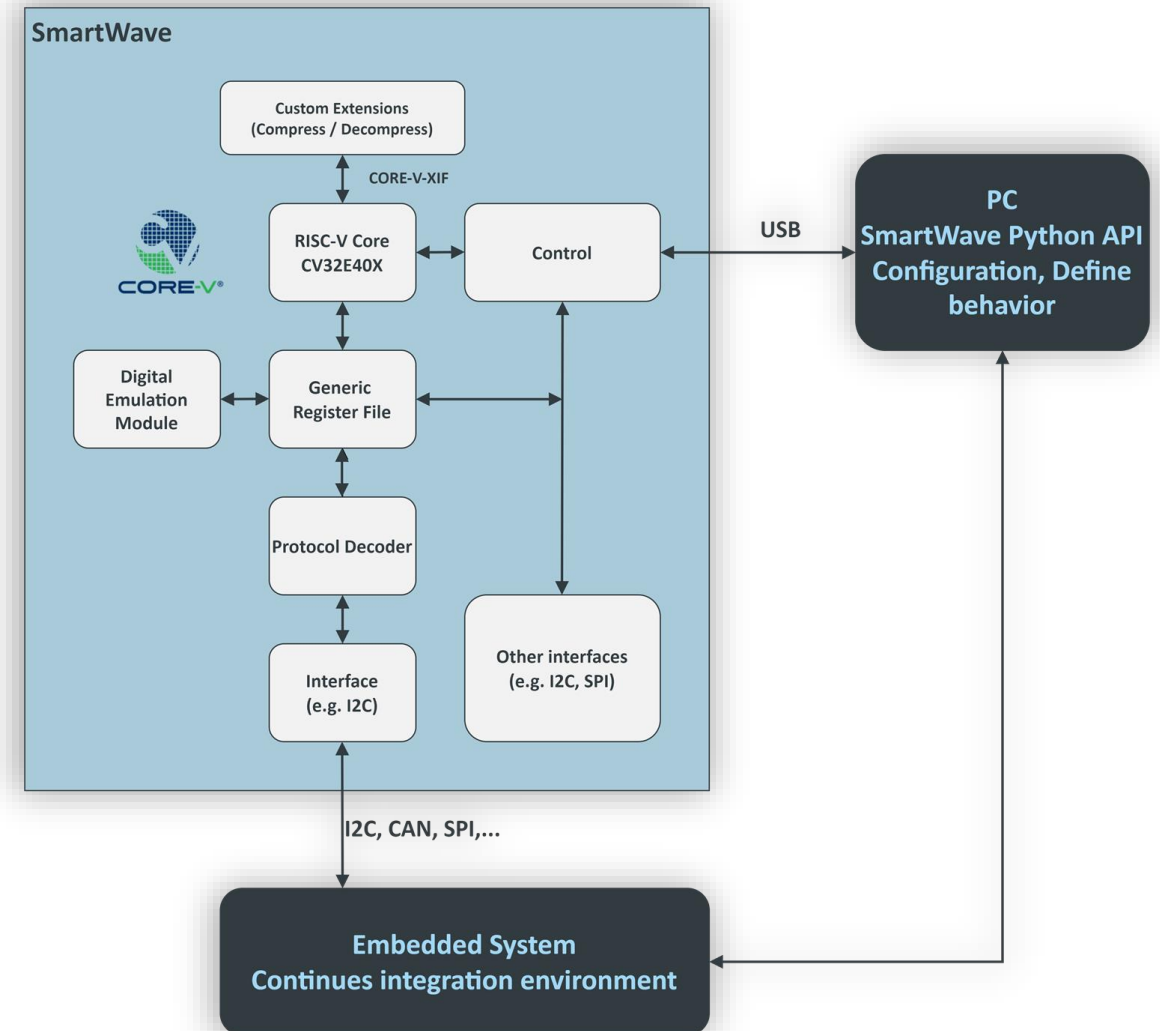
- ✓ Makes it easy to do corner case testing
- ✓ Allows to emulate destructive conditions (e.g. high temperature)
- ✓ Enables continuous integration setups
- ✓ Enables to generate misbehavior (e.g. protocol errors, invalid values...)

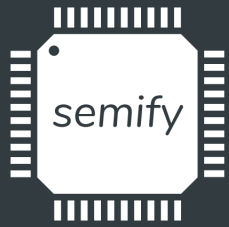


## Providing three modeling levels

Trade off modelling accuracy versus modelling update rate

Modelling Methodology	Model Update Rate	Effort
Python API	Up to 50ms	Low
SW on RISC-V core (CV32E40X)	Up to 50us	Medium
Digital FPGA Logic	Up to 50ns	High





solutions to simplify.

# Temperature Sensor Emulation using SmartWave

Graz, May 2024

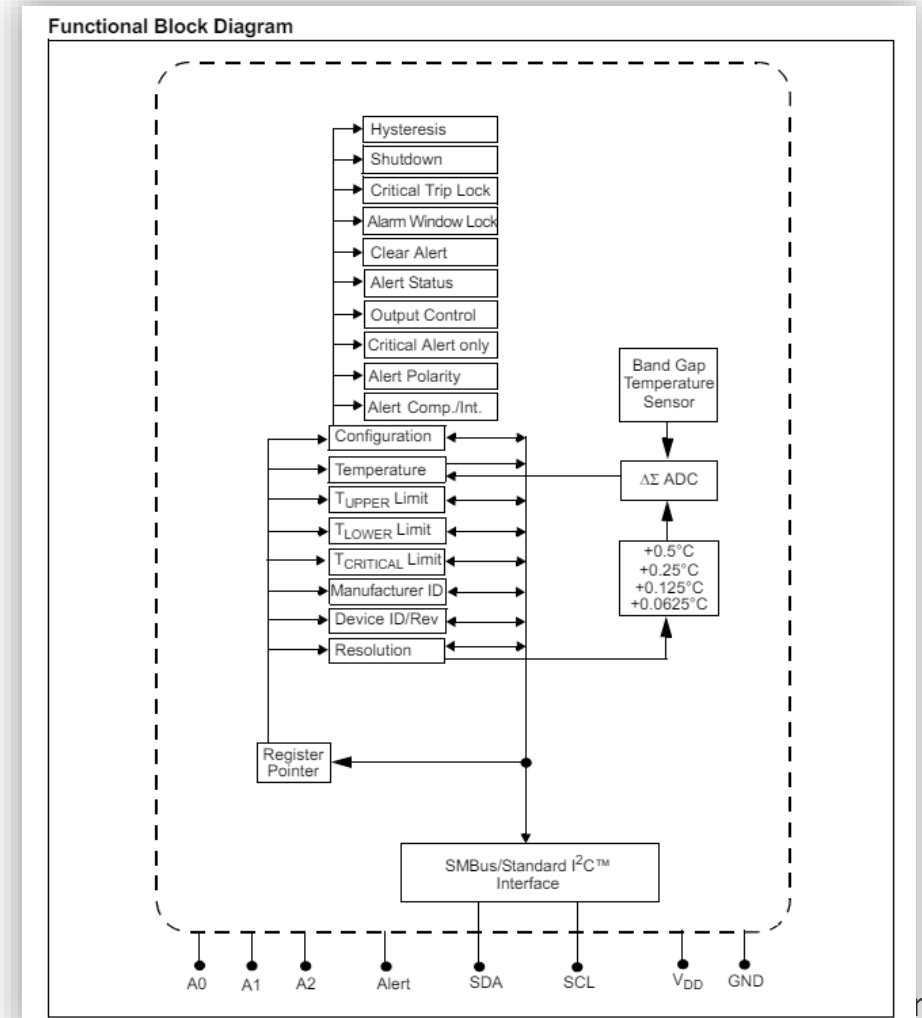
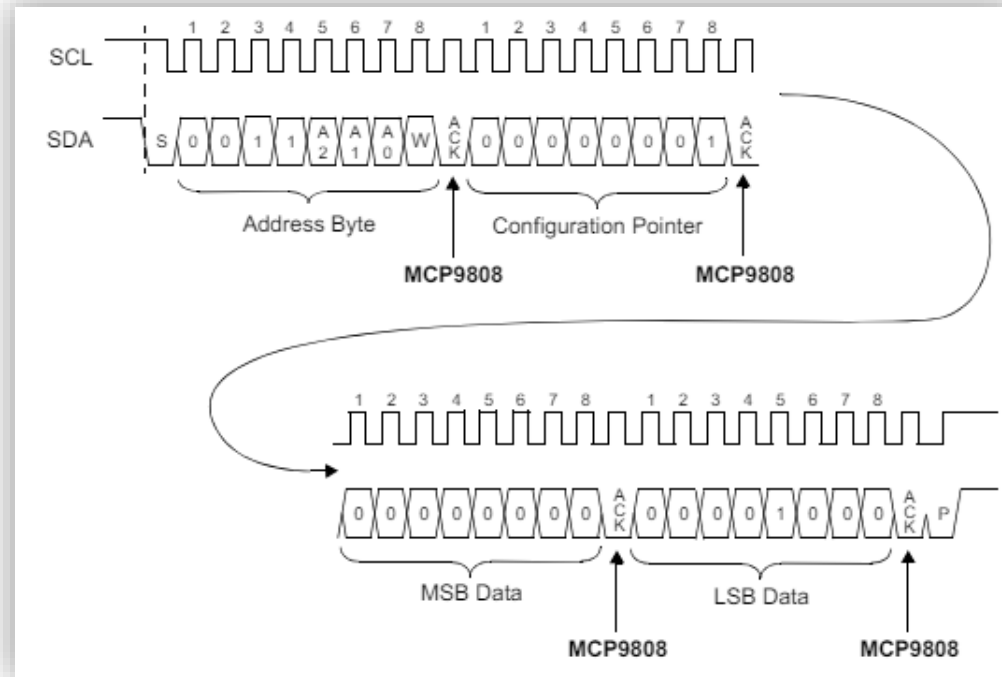
## Getting known an I2C based temperature sensor

- ✓ Readout the temperature sensor SmartWave and WebGUI
- ✓ Readout the temperature sensor using T\*Square Education Board
- ✓ Emulate temperature values using SmartWave

## MCP9808 - Maximum Accuracy Digital Temperature Sensor

✓  $\pm 0.5^{\circ}\text{C}$  (maximum) from  $-20^{\circ}\text{C}$  to  $100^{\circ}\text{C}$

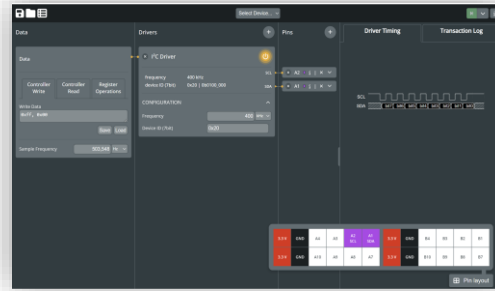
✓ 2-wire Interface: I2C™/SMBus Compatible



# 3 Steps to sensor emulation



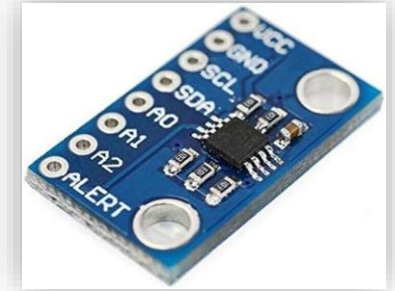
Readout  
Temperature sensor  
via WebGUI



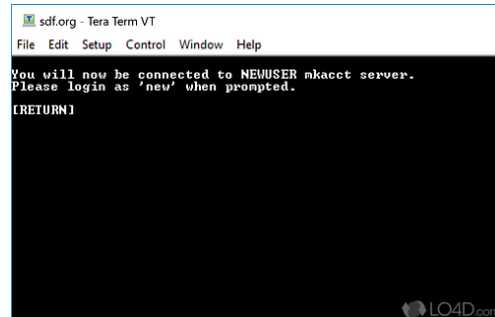
USB



I2C



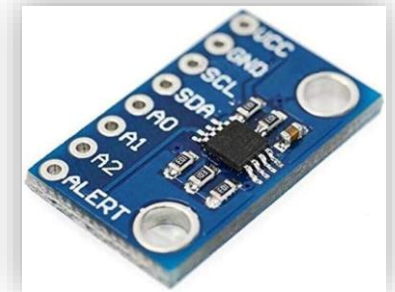
Readout  
Temperature sensor  
via FPGA (Efinix)



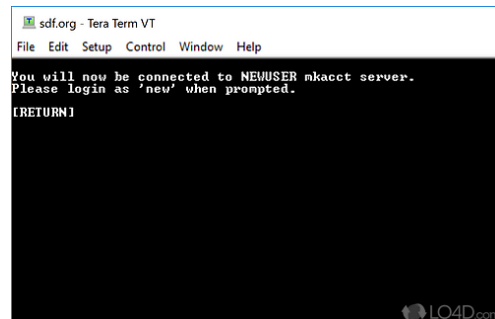
USB



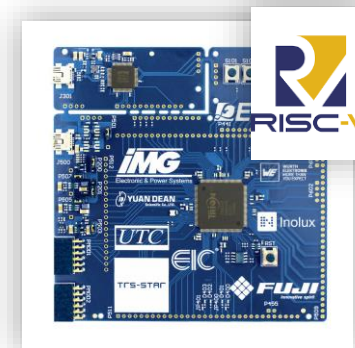
I2C



Readout emulated  
Temperature sensor  
via FPGA (Efinix)



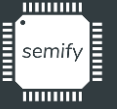
USB



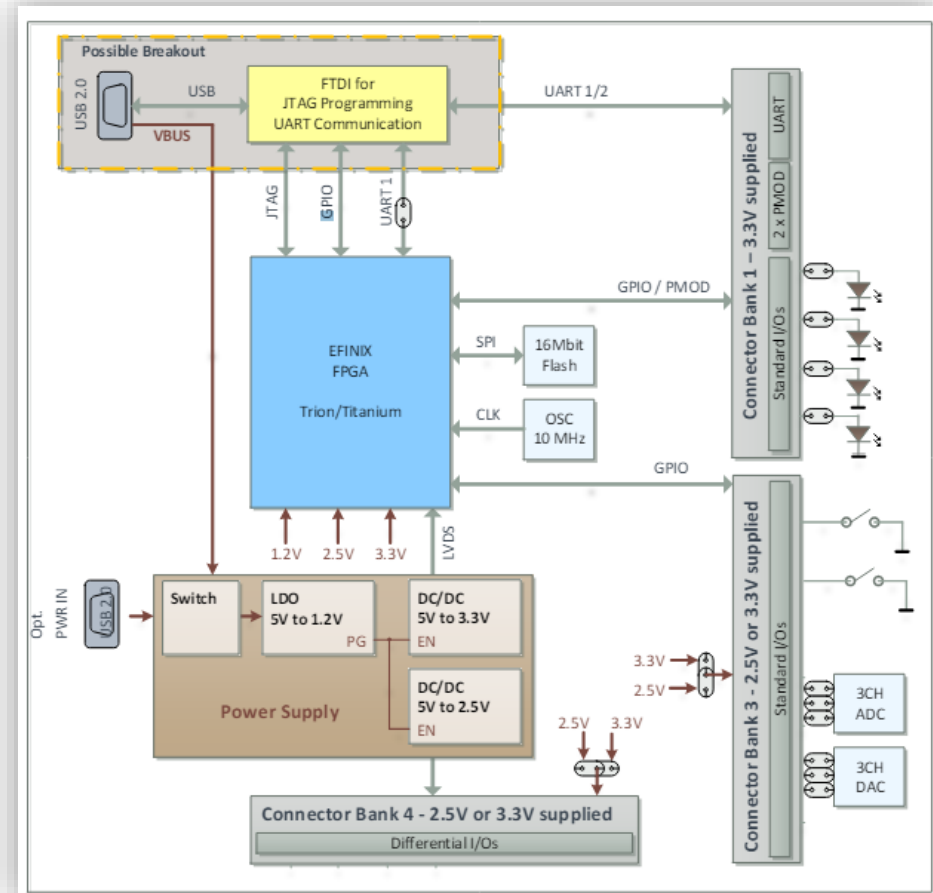
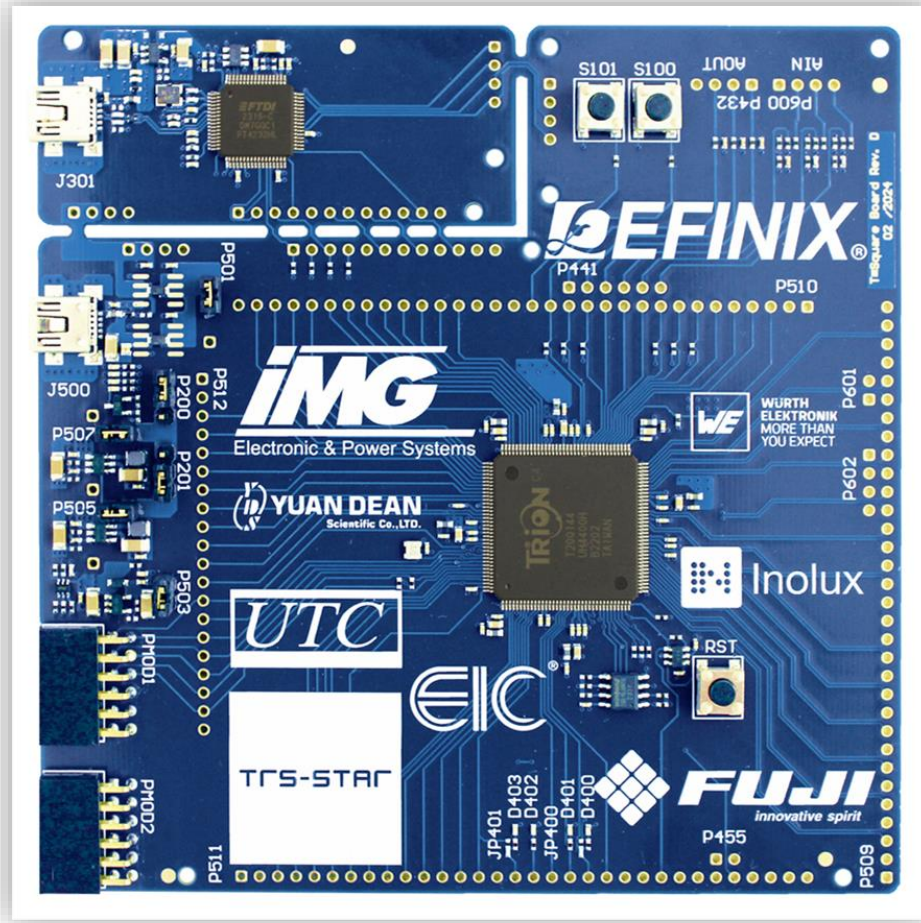
I2C



# Emulate the temperature sensor SmartWave and WebGUI

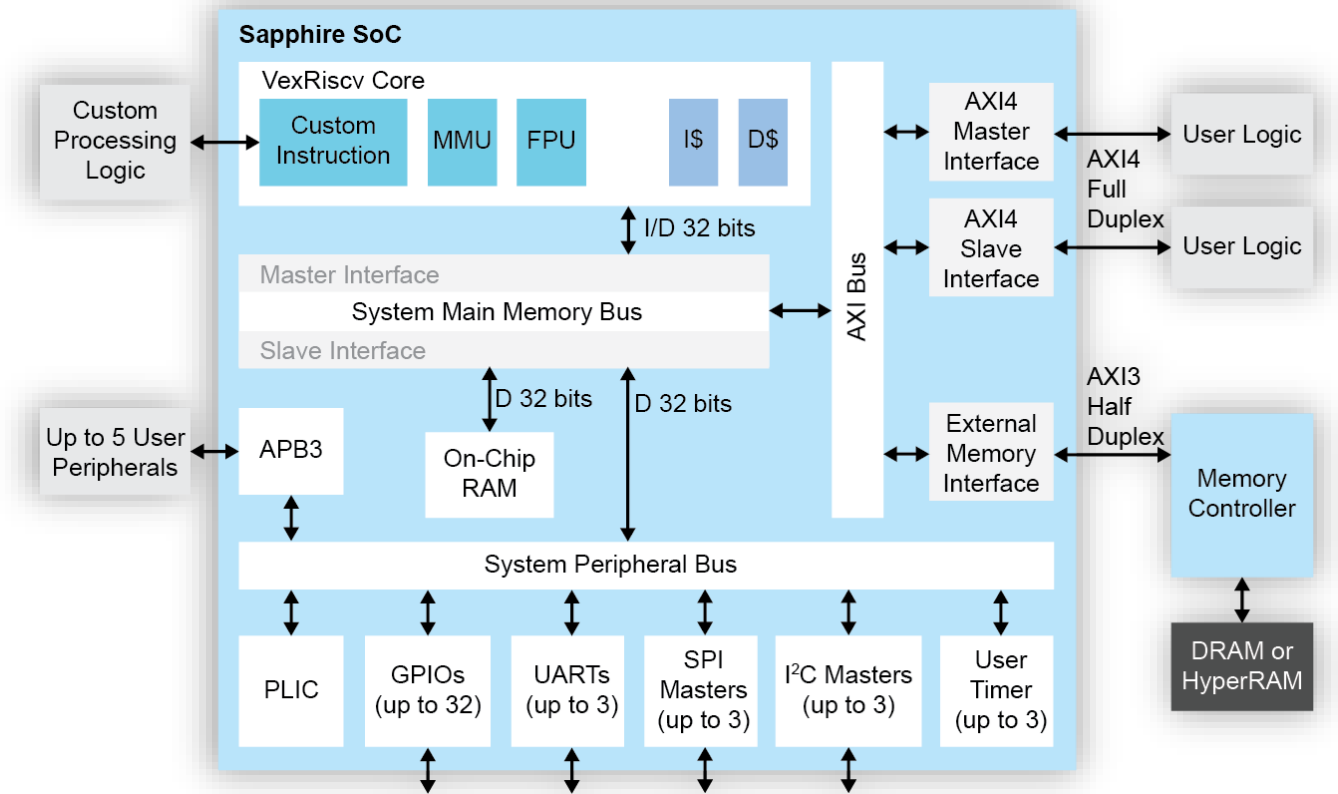


## Effinix Trion T20Q144 FPGA



## Sapphire RISC-V SoC (based on VexRiscv core)

- ✓ 32-bit CPU using ISA RISC-V32I
- ✓ Extensions M, A, F, D, and C
- ✓ Six pipeline stages
- ✓ Support AXI4 and APB3
- ✓ Configurable multi-way instructions and data caches
- ✓ Easy creation of entire systems



<https://www.efinixinc.com/products-riscv.html>

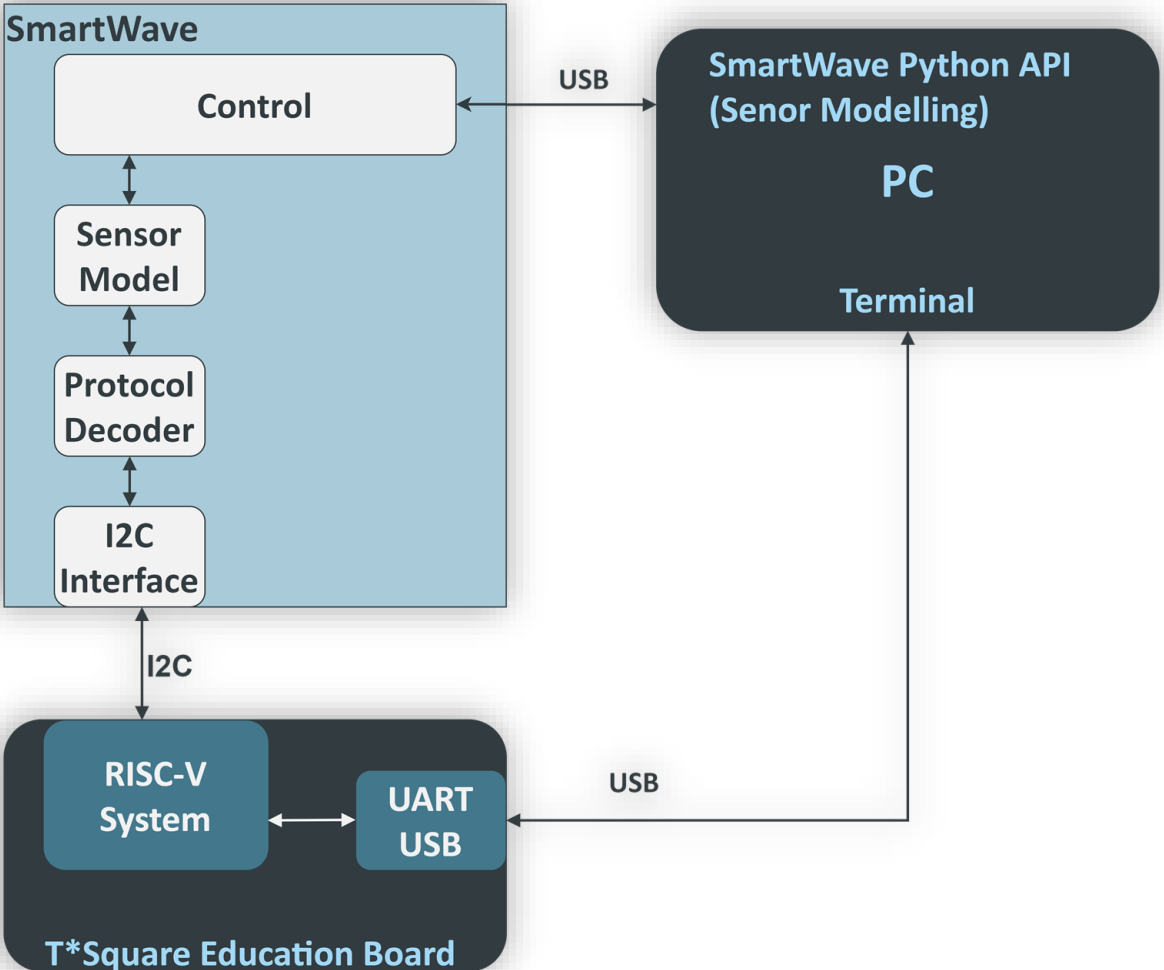


# Emulate temperature values using SmartWave

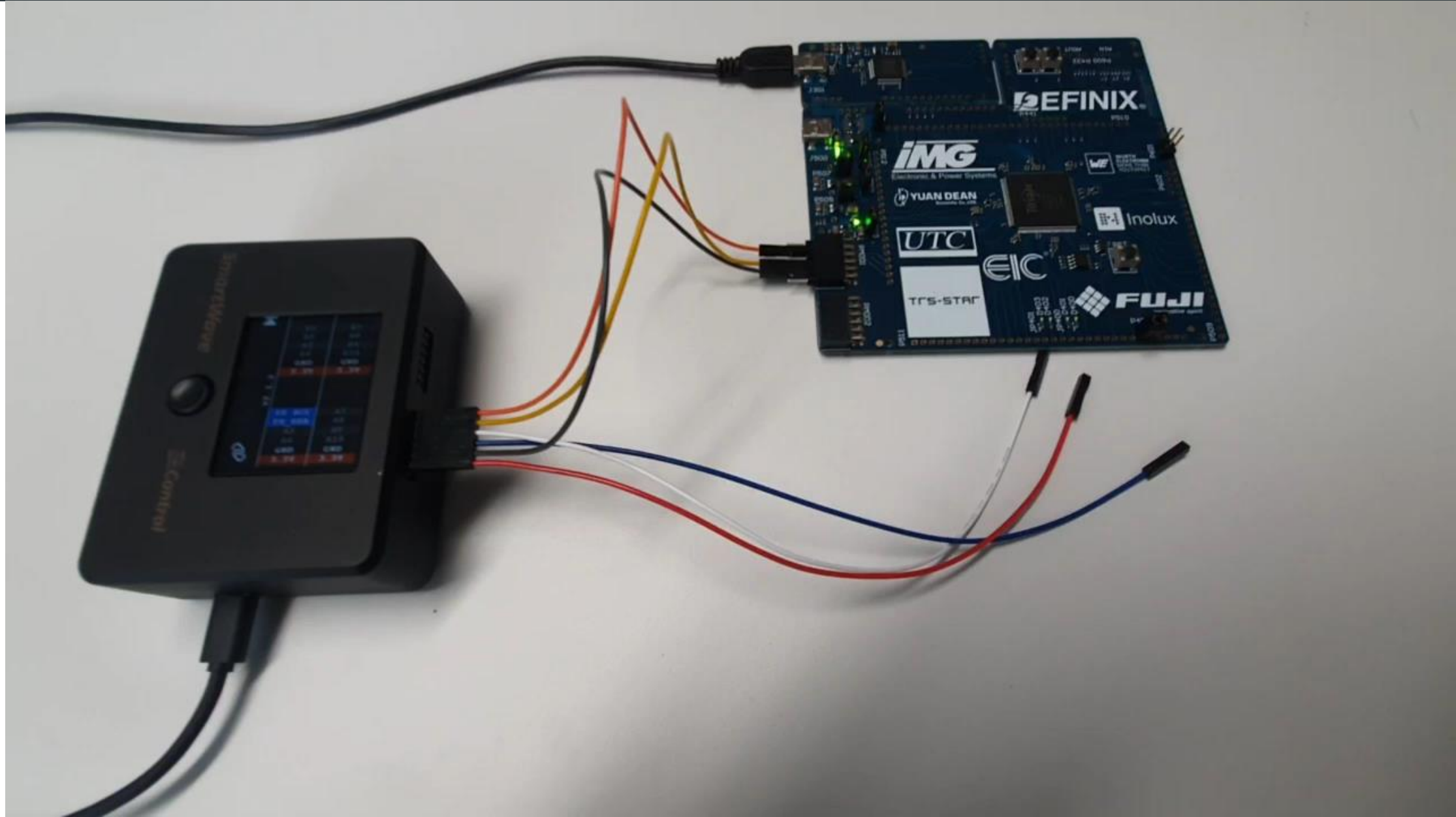


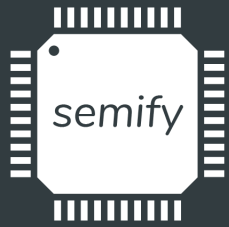


## Behavior is modelled via Python API



# Emulate temperature values using SmartWave





solutions to simplify.

# IMU Sensor Emulation using SmartWave

Graz, May 2024

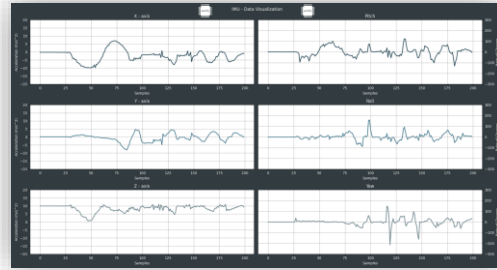
## Getting known an I2C based IMU sensor

- ✓ Readout the IMU sensor using SmartWave and Python API
- ✓ Readout the IMU sensor using T\*Square Education Board
- ✓ Fall detection with IMU and emulated IMU sensor

# 3 Steps to sensor emulation



Readout  
IMU sensor  
via Python API



USB



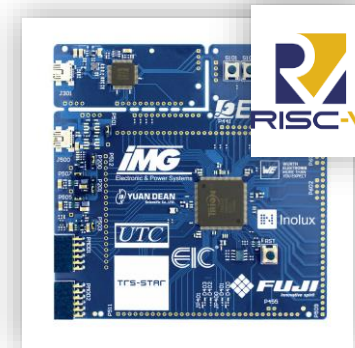
I2C



Readout  
IMU sensor  
via FPGA (Efinix)

```
sdf.org - Tera Term VT
File Edit Setup Control Window Help
You will now be connected to NEWUSER mkacct server.
Please login as 'new' when prompted.
[RETURN]
```

USB



I2C



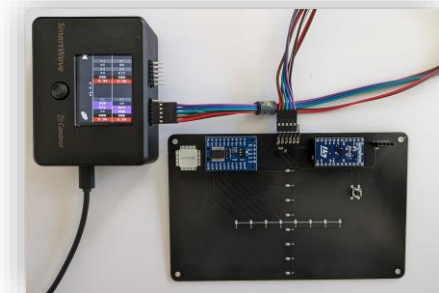
Fall detection  
non-emulated and  
emulated

```
sdf.org - Tera Term VT
File Edit Setup Control Window Help
You will now be connected to NEWUSER mkacct server.
Please login as 'new' when prompted.
[RETURN]
```

USB

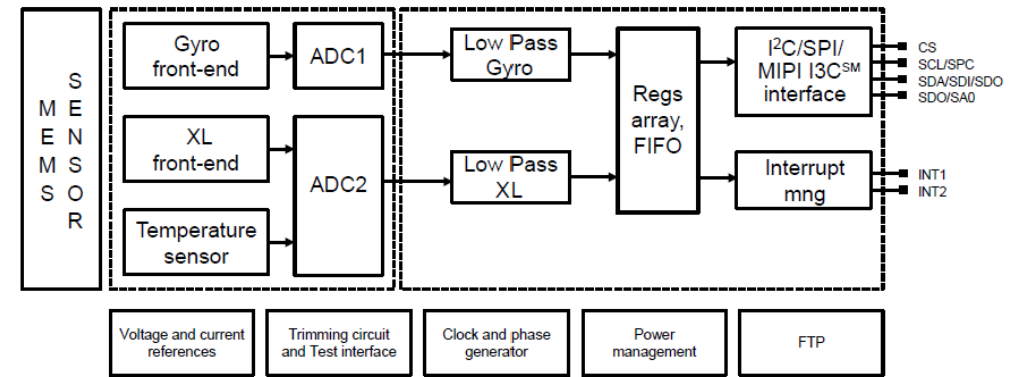


I2C



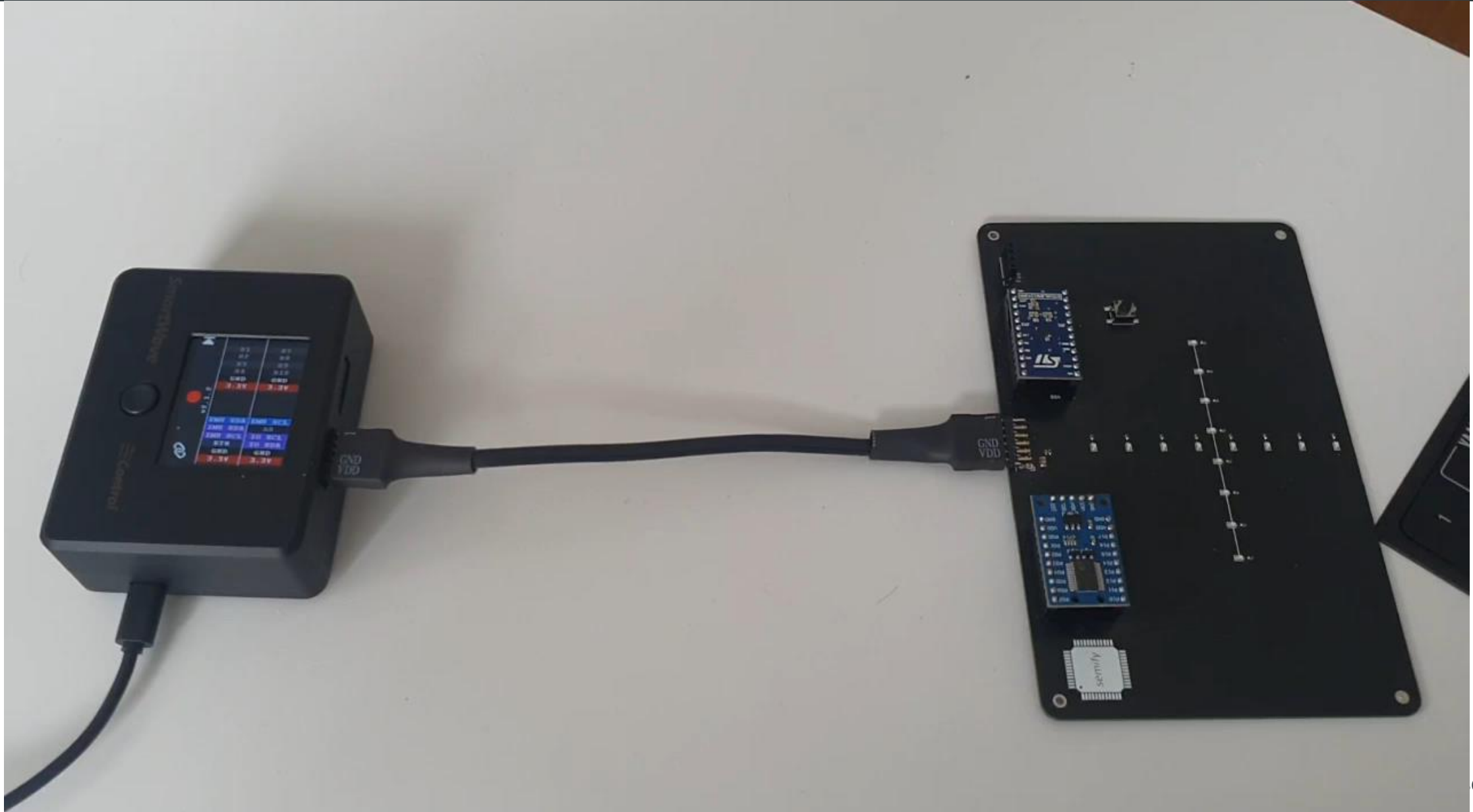
## ASM330LHHXG1 - High-accuracy 6-axis automotive inertial measurement unit

- ✓ Accelerometer user-selectable full scale up to  $\pm 16$  g
- ✓ Extended gyroscope range from  $\pm 125$  to  $\pm 4000$  dps
- ✓ I<sup>2</sup>C, MIPI I3C<sup>SM</sup>, and SPI serial interfaces



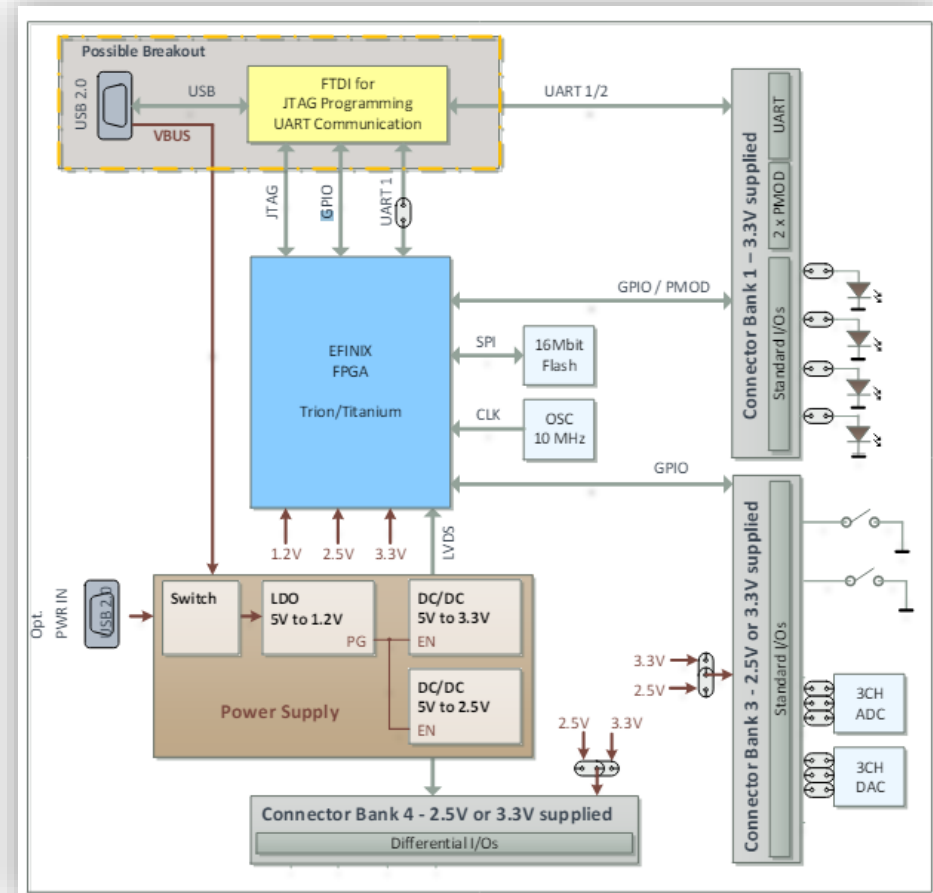
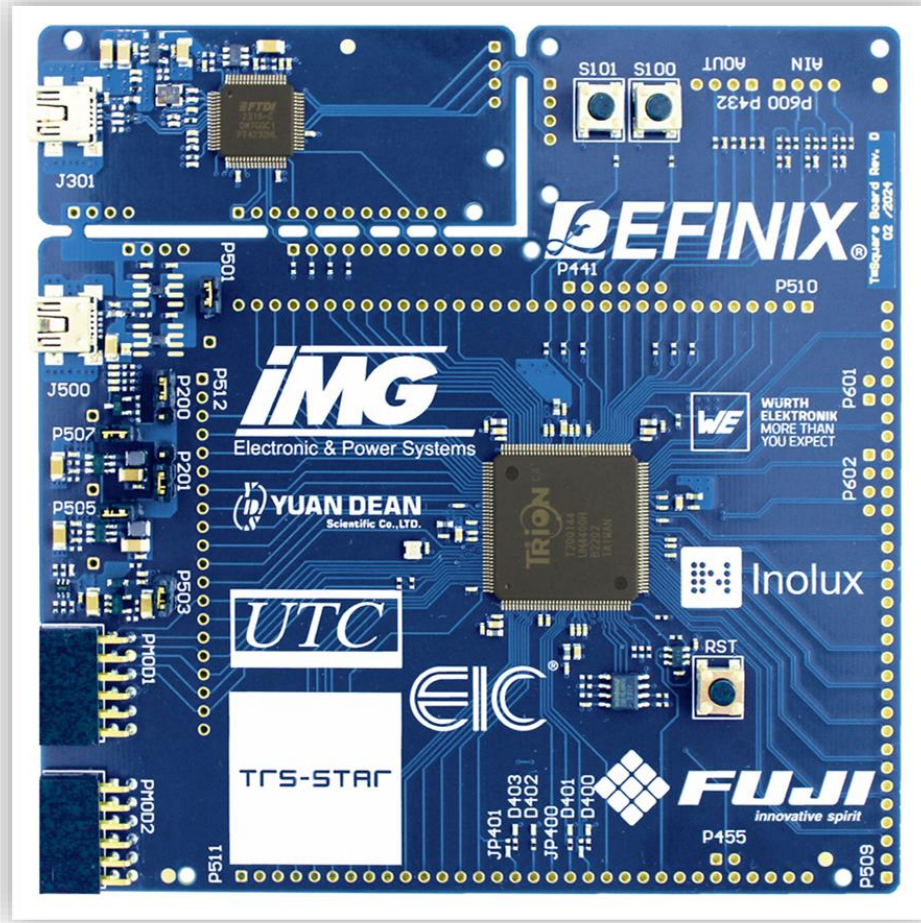
Master	ST	SAD + W		SUB		DATA		SP
Slave			SAK		SAK		SAK	

# Readout via Python API

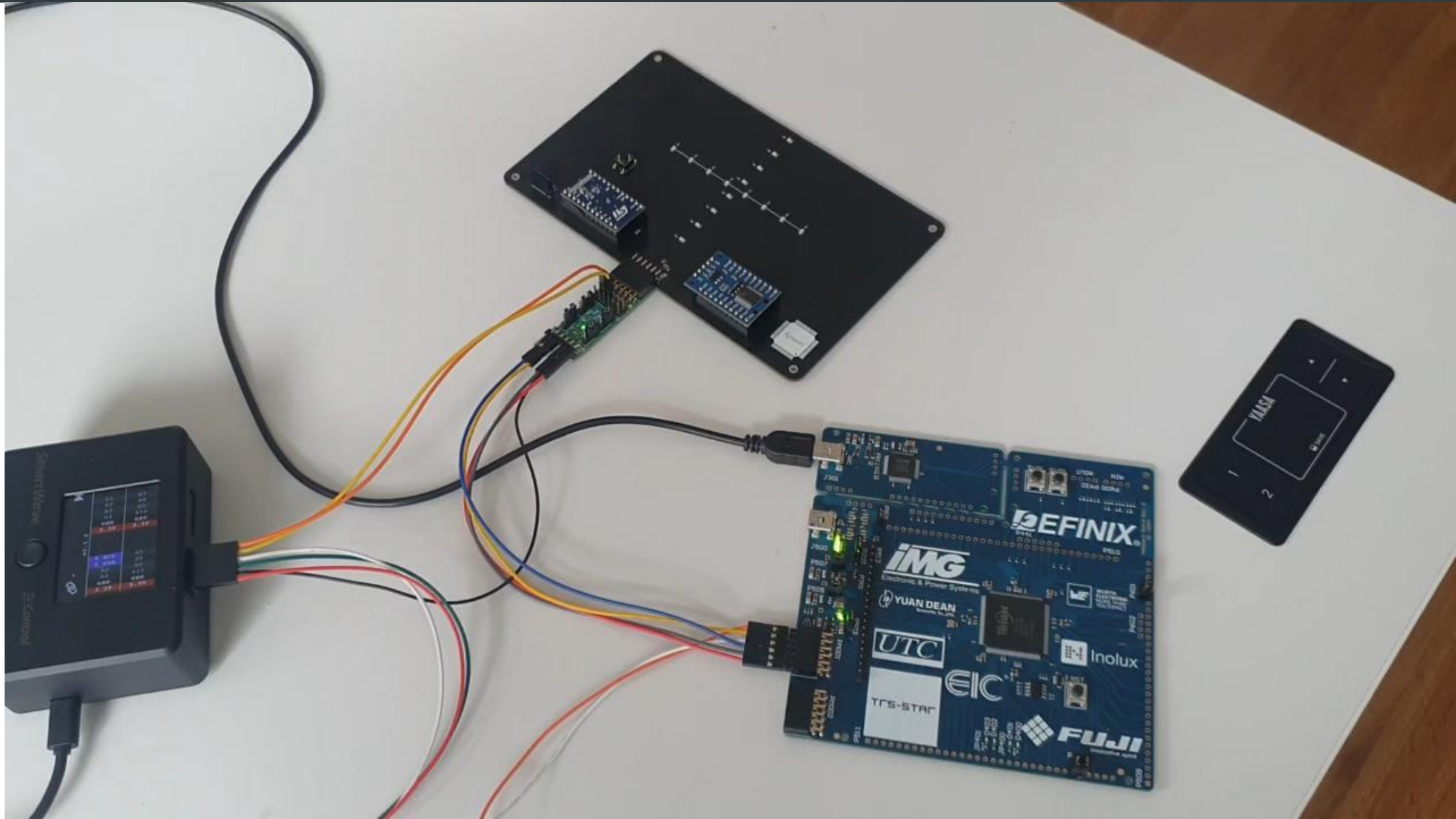




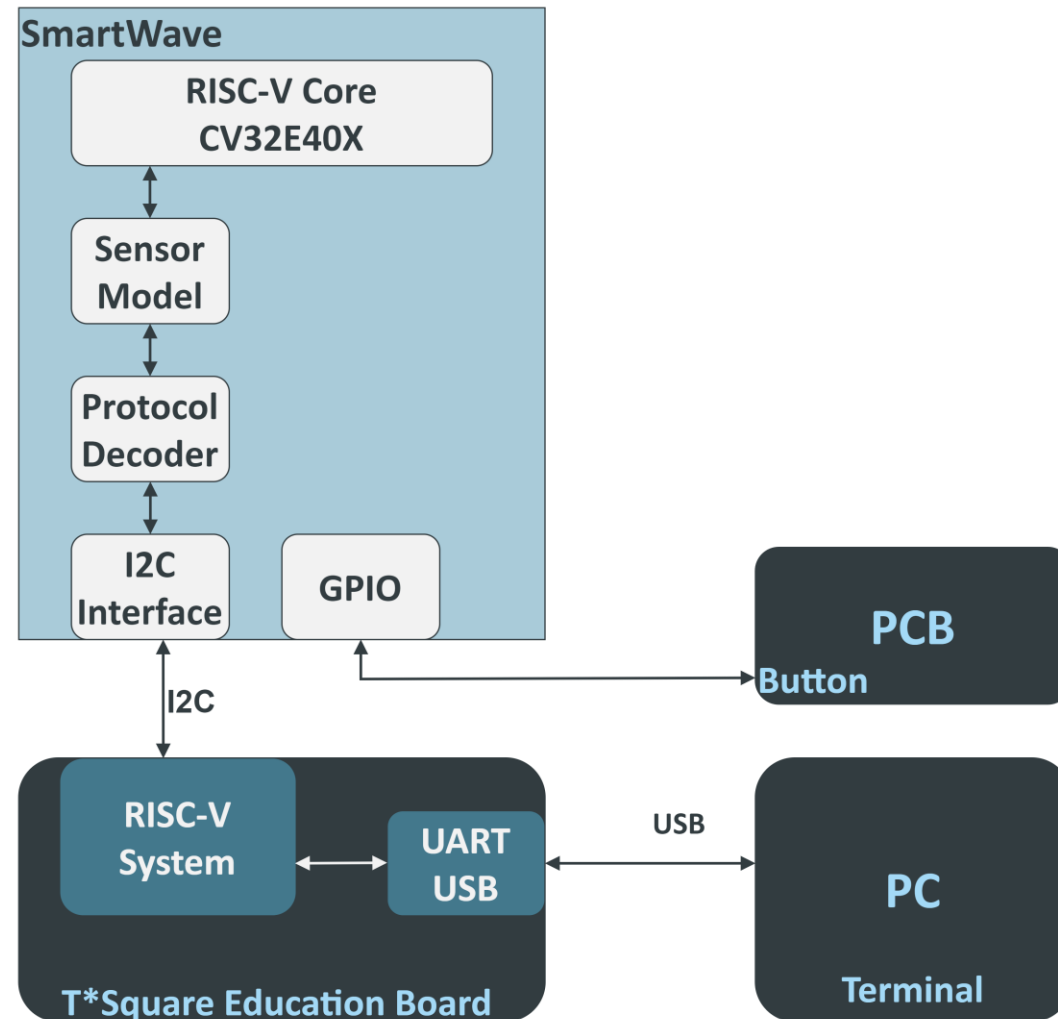
## Effinix Trion T20Q144 FPGA



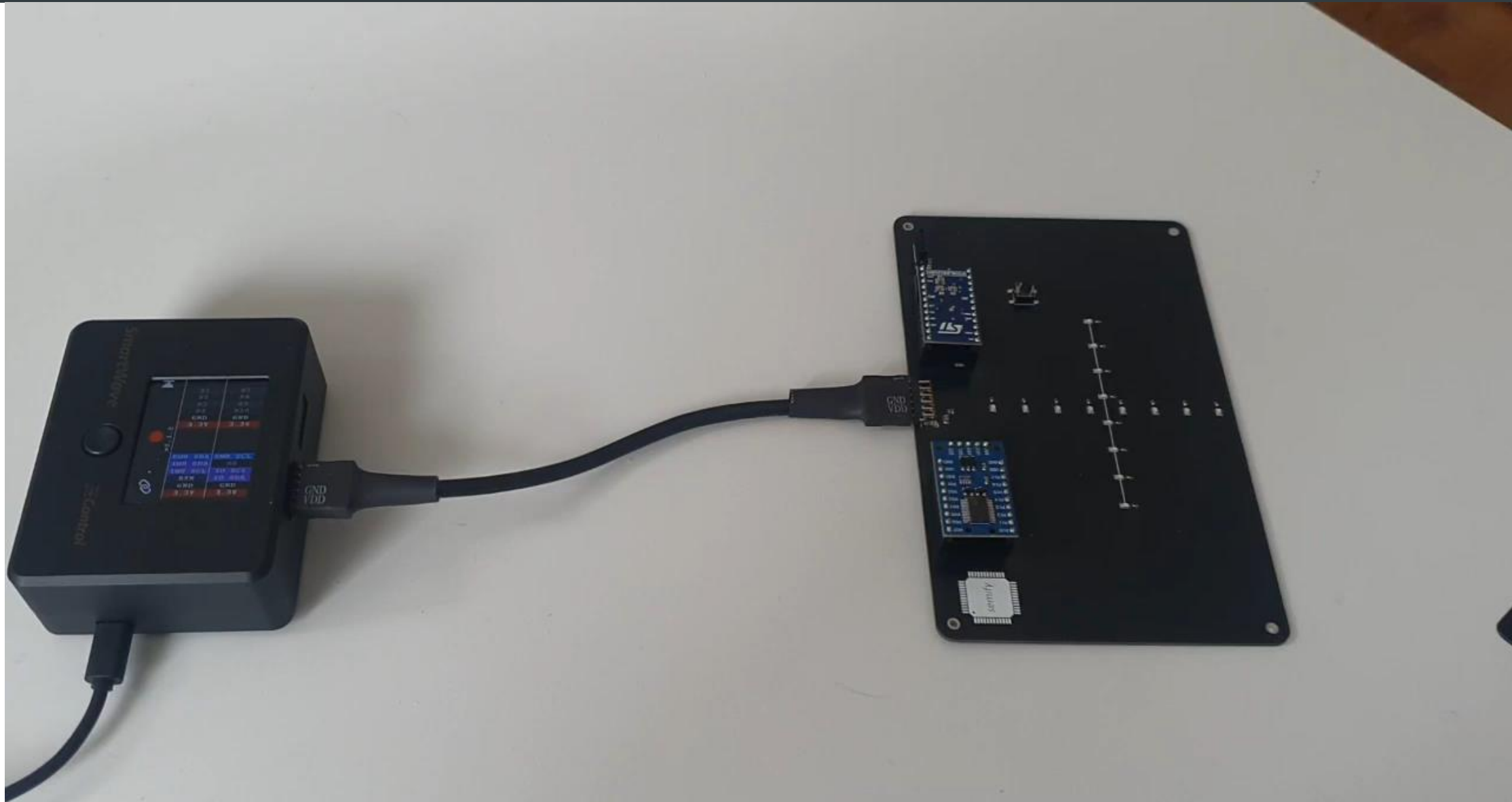
# Readout via T\*Square Education Board



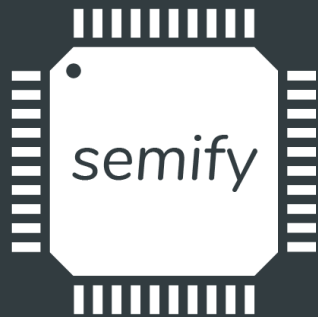
## Behavior is modelled via RISC-V core



# IMU Emulation – Fall Detection



- ✓ Introduction into SmartWave
- ✓ Introduction into SmartWave based Sensor Emulation
- ✓ Demo on how to start using a Temperature Sensor
  - ✓ WebGUI, T\*Square Education Board (real / emulated sensor, API based)
- ✓ Demo on how to start using a IMU Sensor
  - ✓ Python API, T\*Square Education Board (real / emulated sensor, core based)



solutions to simplify.

# Contact Information

Email: [office@semify-eda.com](mailto:office@semify-eda.com)

Website: [www.semify-eda.com](http://www.semify-eda.com)

LinkedIn: <https://www.linkedin.com/company/semify-eda>

Phone: +43 699 1555 7769

**Address:**

semify GmbH

Neubaugasse 24, 8020 Graz - Austria