

Intro to device testing

Where to start and what not to do



Imagine a device, which:

- Networks with other devices
- Does physical interactions

And you have to test its functionality.



Chapter I

Networking? My laptop has ethernet!

Your laptop speaks USB, Ethernet, WiFi, and Bluetooth.



Your device may speak any of...

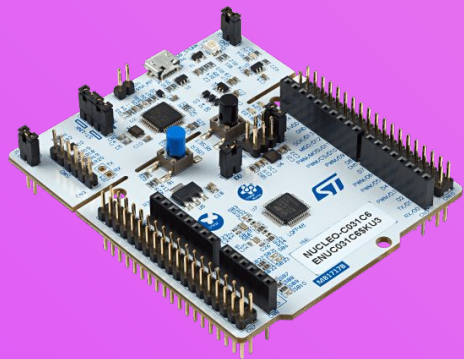
- What your laptop speaks
- UART
- JTAG
- CAN Bus
- HART
- ...many others





If your device only speaks weird protocols, you'll need some way to translate between it and your laptop.

LLMs now make building custom devices - including translators - much easier. Instead of relying on COM ports, all you need now is a dev board and Claude.



These can likely speak the protocols you need, plus USB. Ask your LLM for a firmware and a USB driver that ultimately interfaces with your test sequencer.

Once you have a way to physically talk to your device, you'll still have to figure out the protocols that actually get sent through those wires.

This will again be determined by the developers, and while Python libraries can cover standardized protocols, you'll likely need to get your hands dirty with protocol serialization/deserialization code.

[Imagine logo for CoAP here]



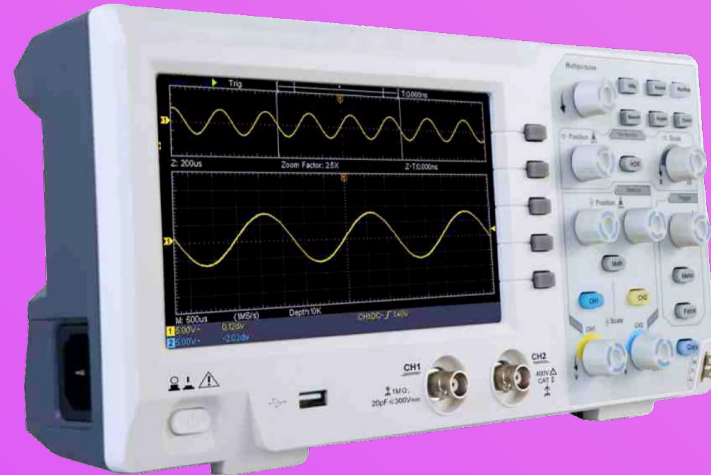
Chapter II

My boss said I have to press the buttons :(

Your device interacts with the real world in some way. Otherwise, it could just run on a server somewhere. Sensor measurements, buttons, LEDs, screens - you have to test everything.



However, your device only really cares about electronic signals, so you can interact via those rather than physical effects. Industry has been doing that for decades.



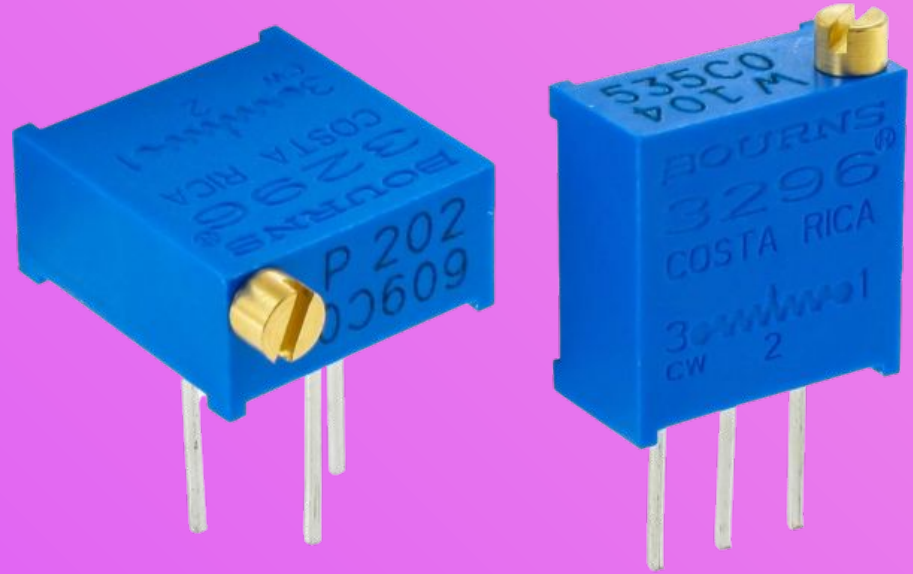
Just like with most communication protocols, your PC doesn't speak digital or analog I/O. You'll again need a translator.



The general industry name for these is DAQ - data acquisition devices. But multimeters, vital signs simulators and many other specialized devices can be a better fit than a general purpose DAQ.

You'll again need to decide if you want to make your own or use off-the-shelf. Either way, your solution will need validation and calibration.

The test harness validation should be short and rare, so manual is OK.

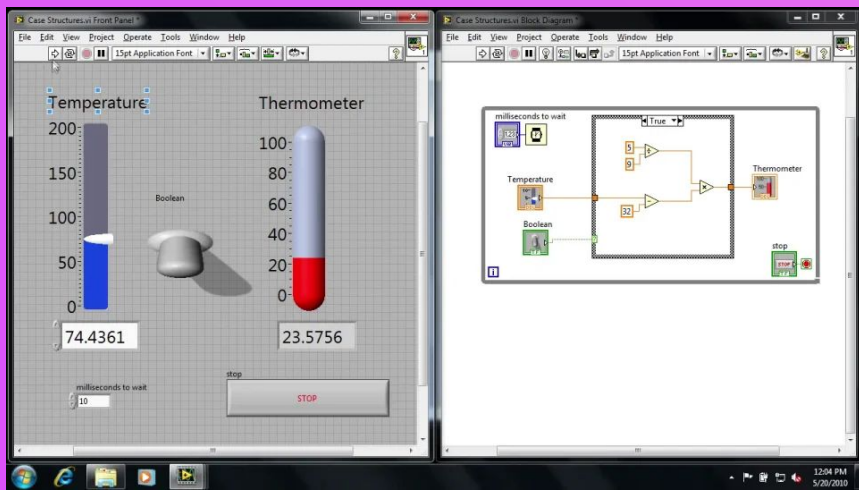


Chapter III

But how do I run the tests?

The language of your tests isn't tied to your device, it's tied to your networking and interaction tools. Some choices:

- LabVIEW, the SAP of testing solutions
- Python - has pytest, but is Python
- Rolling your own



```
/
8 @pytest.fixture
9 def connection(config):
10     return Connection(config)
11
12 def test_device_connects(connection):
13     reply = connection.ping()
14     assert reply is not None
```


Epilogue

If all of that sounds fun, get in touch!



stanislovas@pikta.lt