TestBed -- Automated Hardware-in-the-Loop Test Framework

COSYLAB

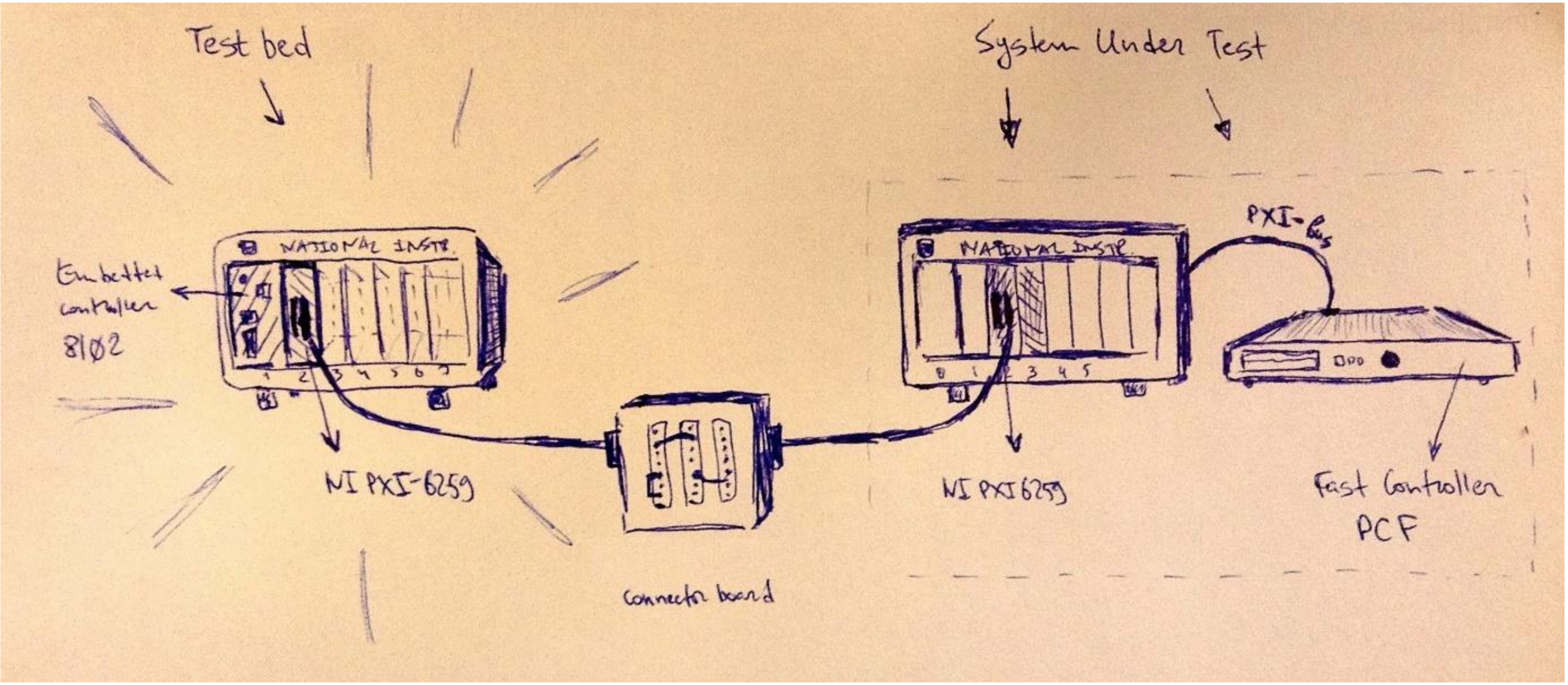
pavel.maslov@cosylab.com

Intro

- Control system updates (>3 times/year)
- DAQ hardware tests:
  - manual (+ precise; - slow, infrequent)
  - automatic (+ fast, repetitive, liberating human resources)

HW architecture

- ❑ TestBed chassis is attached to System Under Test



- ❑ TestBed is running SL 6.3 and CODAC 4.1

SW architecture

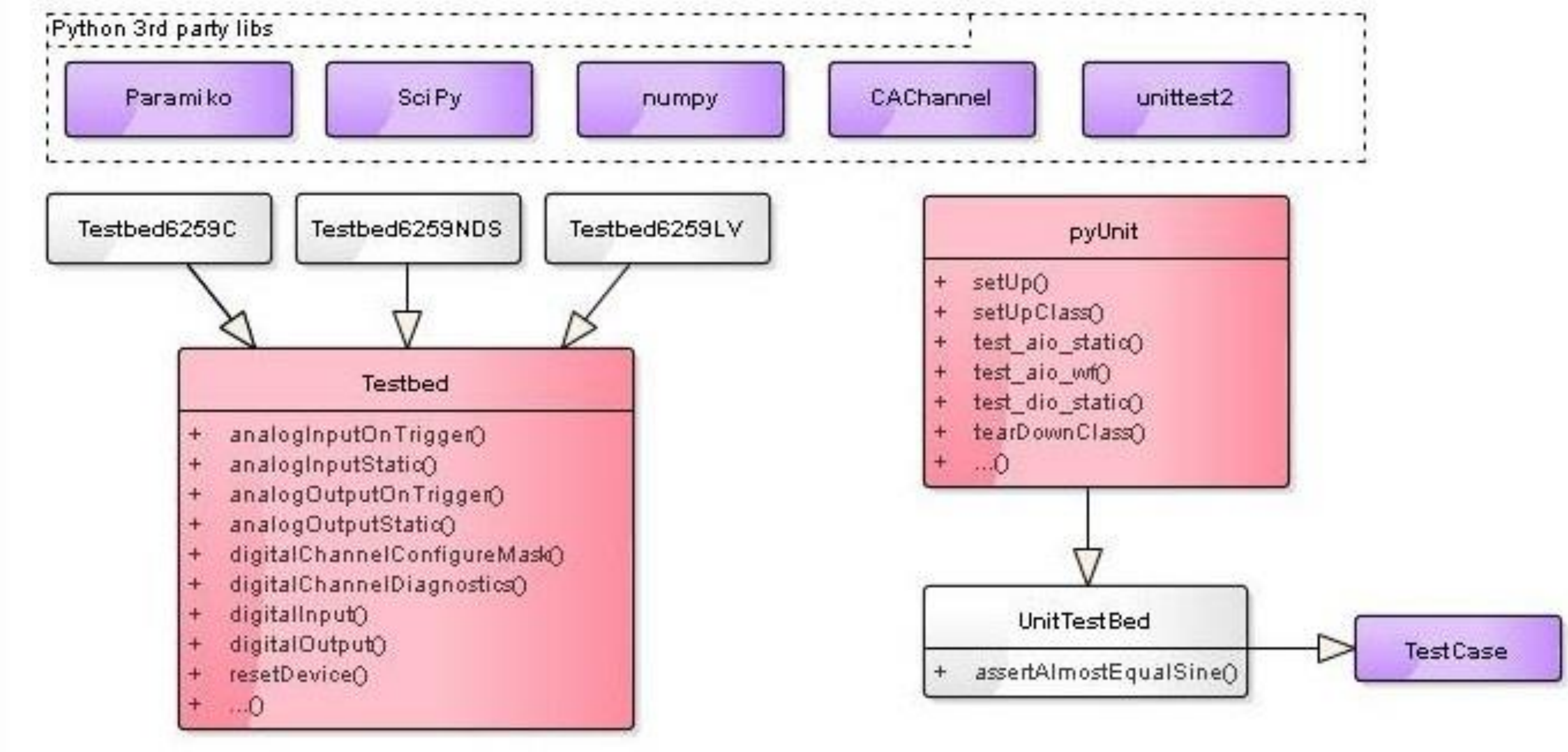
- The software part consists of 3 tiers:
  - Software that provides the desired functionality of a DAQ board:
    - C executables
    - EPICS device support (NDS driver + IOC)
    - *LabVIEW interface*
  - Python bindings in the form of a class
  - Automatic test cases written by the test-plan engineer

DAQ functionality

- The NI-PXI6259 functionality supported in TestBed:
  - Device reset
  - Digital input/output (static) on a desired line
  - DIO diagnostics: port mask and lines state
  - Configuration of the DIO port mask
  - Analog input/output (static) on a desired channel
  - Analog input (waveform) on a trigger
  - Analog output (waveform – sine/saw/square/file) on a trigger

SW architecture

- ❑ Python class diagram:

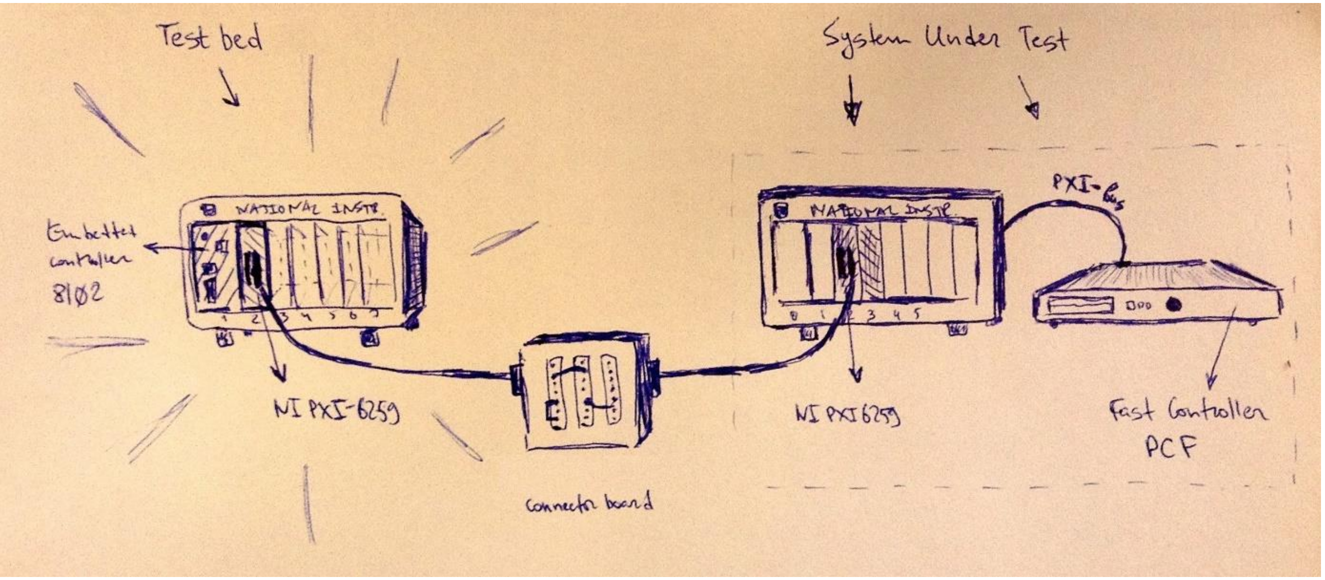


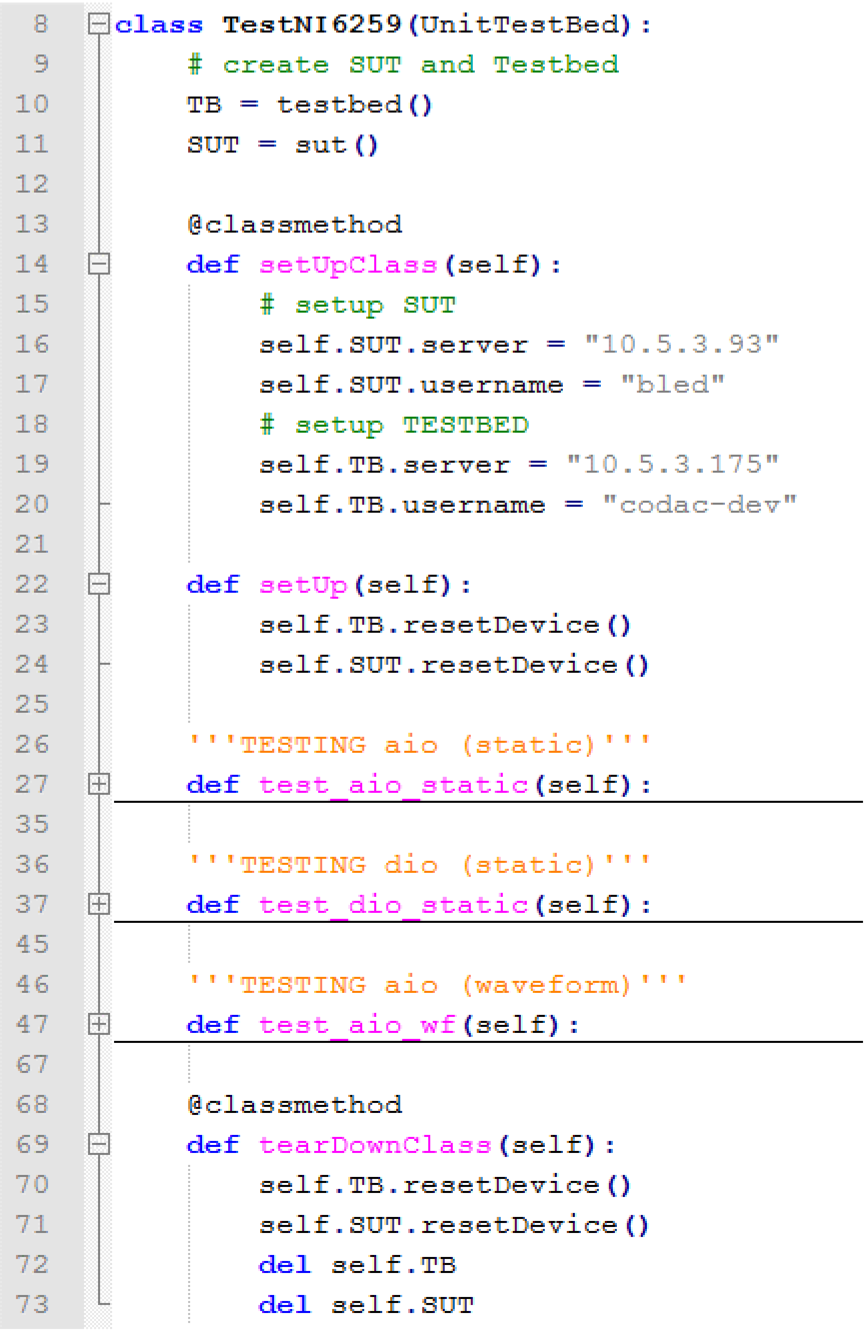
Implementations

- C executables
  - Uses NI PXI-6259 Linux Device Driver (Cosylab)
- EPICS device support
  - Asyn based
  - NDS based
- *LabVIEW*
  - *NI-DAQmx driver supports the full functionality*
  - *start an IOC in LabVIEW (using CA Lab by BESSY)*

Test scenarios

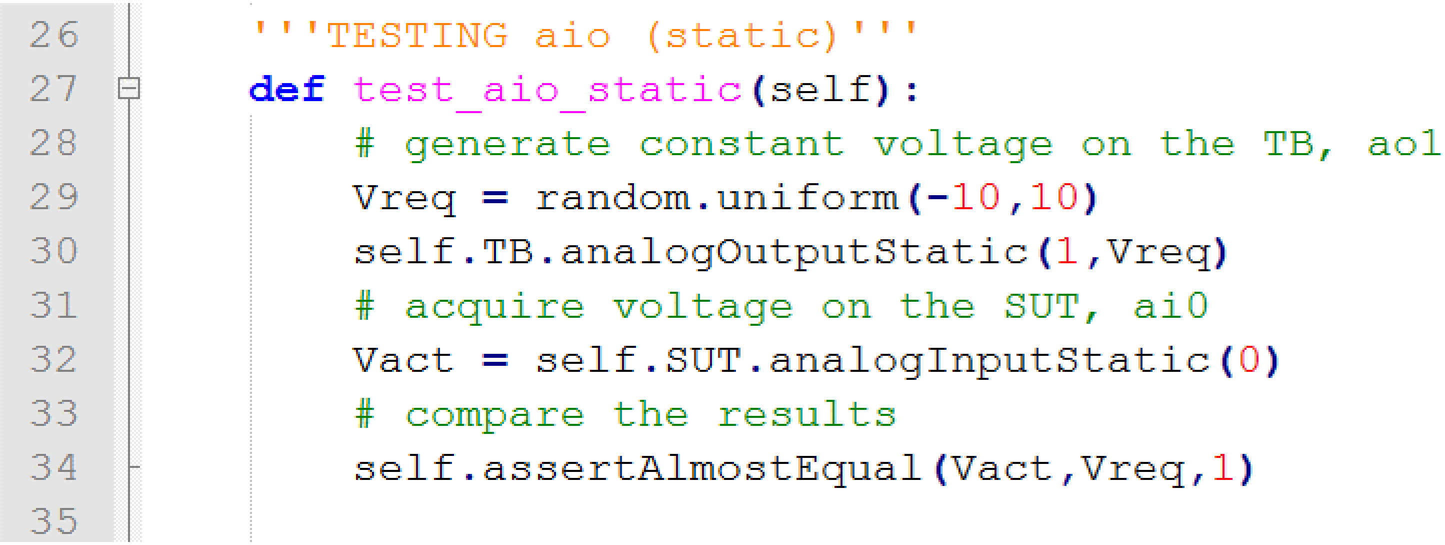
- ❑ Generate on TB, acquire on SUT, check
- ❑ Generate on SUT, acquire on TB, check
- ❑ Generate on SUT, acquire on SUT





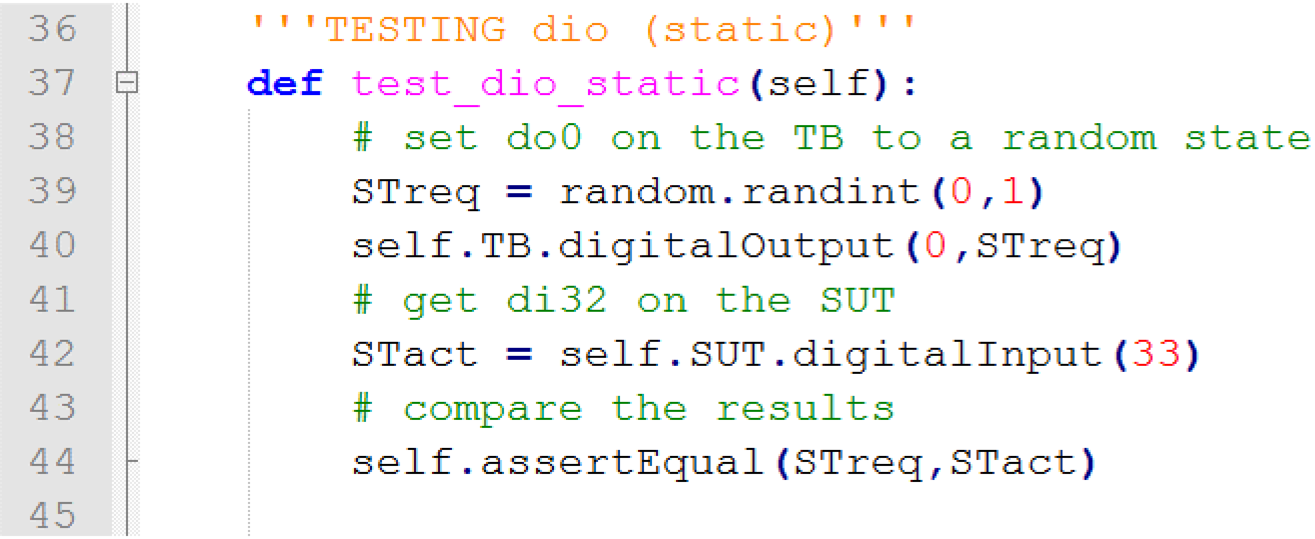
```python
class TestNI6259(UnitTestBed):
    # create SUT and Testbed
    TB = testbed()
    SUT = sut()

    @classmethod
    def setUpClass(self):
        # setup SUT
        self.SUT.server = "10.5.3.93"
        self.SUT.username = "bled"
        # setup TESTBED
        self.TB.server = "10.5.3.175"
        self.TB.username = "codac-dev"

    def setUp(self):
        self.TB.resetDevice()
        self.SUT.resetDevice()

    '''TESTING aio (static)'''
    def test_aio_static(self):

    '''TESTING dio (static)'''
    def test_dio_static(self):

    '''TESTING aio (waveform)'''
    def test_aio_wf(self):

    @classmethod
    def tearDownClass(self):
        self.TB.resetDevice()
        self.SUT.resetDevice()
        del self.TB
        del self.SUT
```

test_aio_static



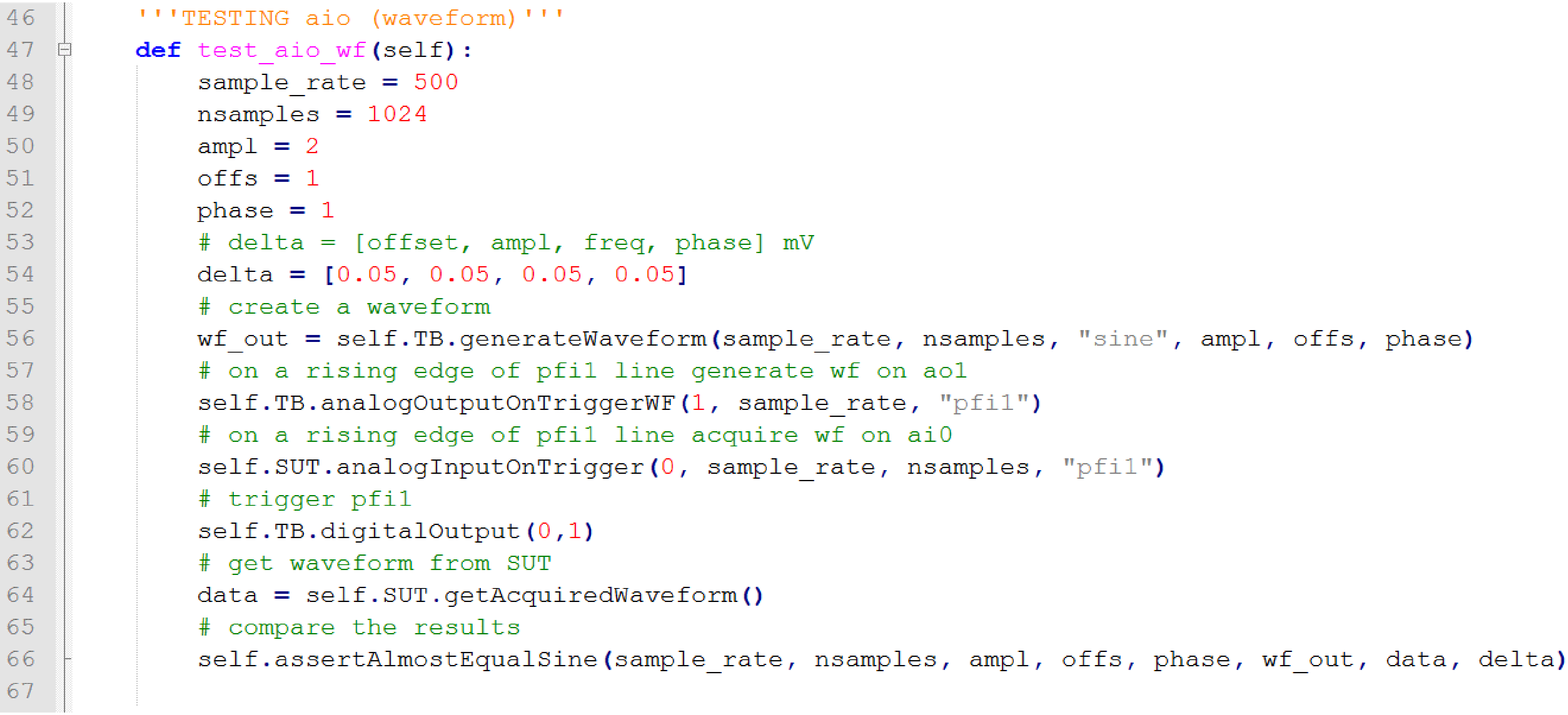
```
    '''TESTING aio (static)'''
    def test_aio_static(self):
        # generate constant voltage on the TB, ao1
        Vreq = random.uniform(-10,10)
        self.TB.analogOutputStatic(1,Vreq)
        # acquire voltage on the SUT, ai0
        Vact = self.SUT.analogInputStatic(0)
        # compare the results
        self.assertAlmostEqual(Vact,Vreq,1)

```

test_dio_static



```python
    '''TESTING dio (static)'''
    def test_dio_static(self):
        # set do0 on the TB to a random state
        STreq = random.randint(0,1)
        self.TB.digitalOutput(0,STreq)
        # get di32 on the SUT
        STact = self.SUT.digitalInput(33)
        # compare the results
        self.assertEqual(STreq,STact)

```

test_aio_wf



```python
    '''TESTING aio (waveform)'''
    def test_aio_wf(self):
        sample_rate = 500
        nsamples = 1024
        ampl = 2
        offs = 1
        phase = 1
        # delta = [offset, ampl, freq, phase] mV
        delta = [0.05, 0.05, 0.05, 0.05]
        # create a waveform
        wf_out = self.TB.generateWaveform(sample_rate, nsamples, "sine", ampl, offs, phase)
        # on a rising edge of pfi1 line generate wf on ao1
        self.TB.analogOutputOnTriggerWF(1, sample_rate, "pfi1")
        # on a rising edge of pfi1 line acquire wf on ai0
        self.SUT.analogInputOnTrigger(0, sample_rate, nsamples, "pfi1")
        # trigger pfi1
        self.TB.digitalOutput(0,1)
        # get waveform from SUT
        data = self.SUT.getAcquiredWaveform()
        # compare the results
        self.assertAlmostEqualSine(sample_rate, nsamples, ampl, offs, phase, wf_out, data, delta)
```

IT ALL COMES
NATIONAL INSTRUMENTS
NI PXI-1042Q
PXI
RIGOL DS1052D
UltraZoom
SCB-68

This project has received funding from the European Union’s Seventh Framework Programme for research, technological development and demonstration under grant agreement no 289485.

THANK YOU!

Pavel Maslov

COSYLAB

Tel.: +386 406 32 571

Web: www.cosylab.com