Wrocław University of Science and Technology

PYTHON
LABORATORY REPORT

Faculty of Electronics, Photonics and Microsystems

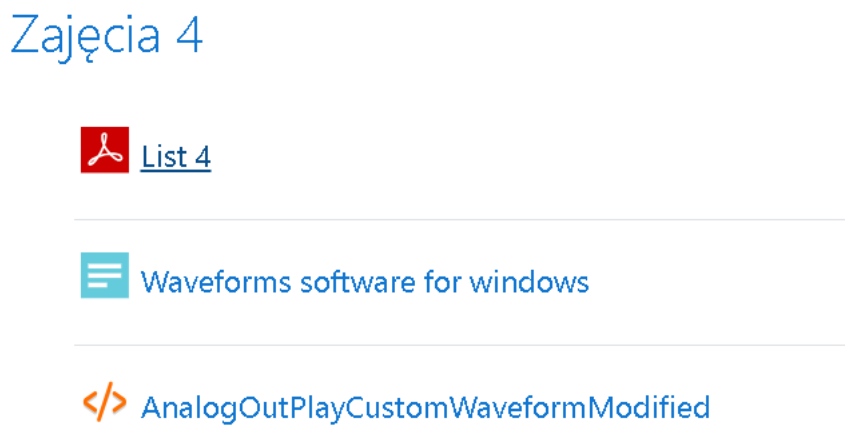
Theme of class: Analog Discovery

Student: Hayrettin Aycetin (276807)
Date of class: 04.12.2023 15:15-16:55
Group No:3
Submition Date:18.12.2023

Lab assistant: Aleksander Kubeczek, Alicja Kwaśny GRADE:

Task 1

Make sure that Digilent Waveforms and Digilent Adept packages are installed (available on eportal).



Task 2

Install pydwf package [pydwf documentation] (it is recommended to use pip – see website for documentation).

This allows installation using the standard *pip* (or *pip3*) tool:

```
pip install pydwf
```

After installing *pydwf*, the following command will show the version of *pydwf* and the underlying DWF library:

```
python -m pydwf version
```

The following command will list all Digilent Waveforms devices connected to the system and, for each of them, list the supported configurations:

```
python -m pydwf list -c
```

I just wrote these commands to my terminal in Visual Studio Code.

Output:

```
Requirement already satisfied: pydwf in c:\users\harettİn\appdata\local\packages\pythonsoftwarefoundation.python
.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (1.1.19)
Requirement already satisfied: numpy in c:\users\harettİn\appdata\local\packages\pythonsoftwarefoundation.python
.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (from pydwf) (1.26.1)
```

```
pydwf version ............. : 1.1.19
DWF library version ...... : 3.21.3
```

```
No Digilent Waveforms devices found.
```

Comment:
I already installed pydwf so it said already satisfied.
Second code let me check the version.
Third code gives me an output of no digilent waveforms found because I tried on my computer without digilent discovery at home but in University Lab it worked without any problems.

Task 3

Extract example scripts provided by the pydwf authors.

If desired, the documentation can also be installed locally after installing the package by executing the following command:

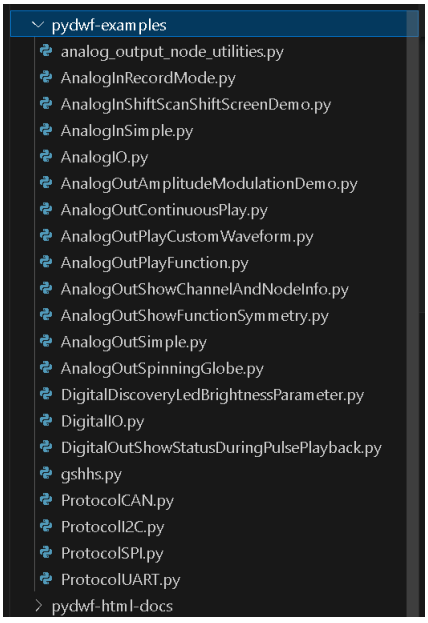
```
python -m pydwf extract-html-docs
```

This will create a local directory called *pydwf-docs-html* containing the project documentation in HTML format.

Alternatively, a PDF version of the manual can be extracted as well:

```
python -m pydwf extract-pdf-manual
```

I just wrote these commands to my terminal in Visual Studio Code.



It automatically downloaded some files from the web to my directory.

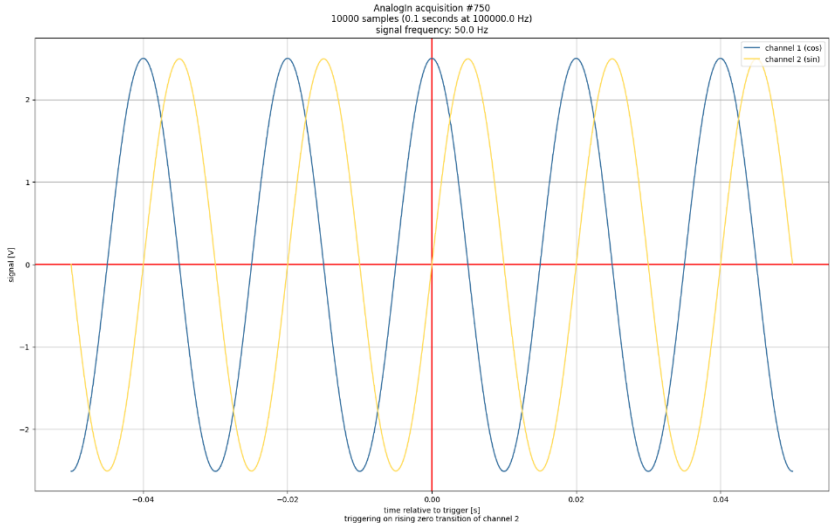
Comments:
Everything was made only using two commands so I think not much to comment.

Task 4

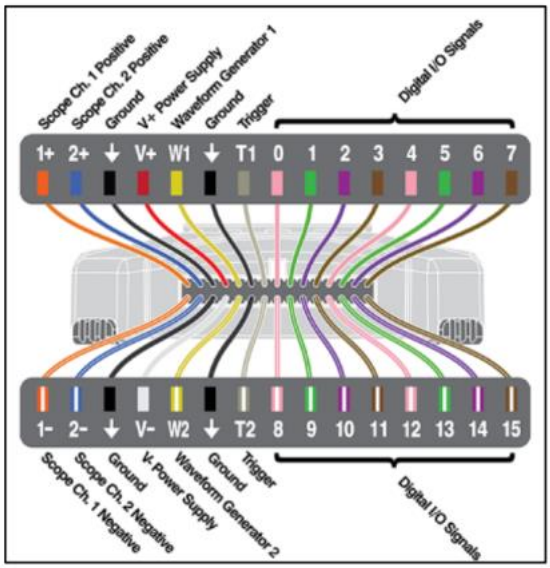
Run AnalogInRecordMode.py example script on your Analog Discovery board (remember to connect the wires properly).

```python
from pydwf.utilities import openDwfDevice


Codeium: Refactor | Explain | ×
def configure_analog_output(analogOut, analog_out_frequency, analog_out_amplitude, analog_out_offset):
    """Configure a cosine signal on channel 1, and a sine signal on channel 2."""

    CH1 = 0  # This channel will carry a 'cosine' (i.e., precede channel 2 by 90 degrees).
    CH2 = 1  # This channel will carry a 'sine'.

    node = DwfAnalogOutNode.Carrier

    analogOut.reset(-1)  # Reset both channels.

    analogOut.nodeEnableSet   (CH1, node, True)
    analogOut.nodeFunctionSet (CH1, node, DwfAnalogOutFunction.Sine)
    analogOut.nodeFrequencySet(CH1, node, analog_out_frequency)
    analogOut.nodeAmplitudeSet(CH1, node, analog_out_amplitude)
    analogOut.nodeOffsetSet   (CH1, node, analog_out_offset)
    analogOut.nodePhaseSet    (CH1, node, 90.0)

    analogOut.nodeEnableSet   (CH2, node, True)
    analogOut.nodeFunctionSet (CH2, node, DwfAnalogOutFunction.Sine)
    analogOut.nodeFrequencySet(CH2, node, analog_out_frequency)
    analogOut.nodeAmplitudeSet(CH2, node, analog_out_amplitude)
    analogOut.nodeOffsetSet   (CH2, node, analog_out_offset)
    analogOut.nodePhaseSet    (CH2, node, 0.0)

    # Synchronize second channel to first channel. This ensures that they will start simultaneously.
    analogOut.masterSet(CH2, CH1)
```

Output:



This is a guide to how to use Analog Discovery.



Comments:

This script operates by producing signals on two AnalogOut channels and recording them on corresponding AnalogIn channels. Ensure that you connect analog-out channel #1 to analog-in channel #1 and analog-out channel #2 to analog-in channel #2 before running the script. After setup and execution, the displayed graph will show the resulting waveforms.

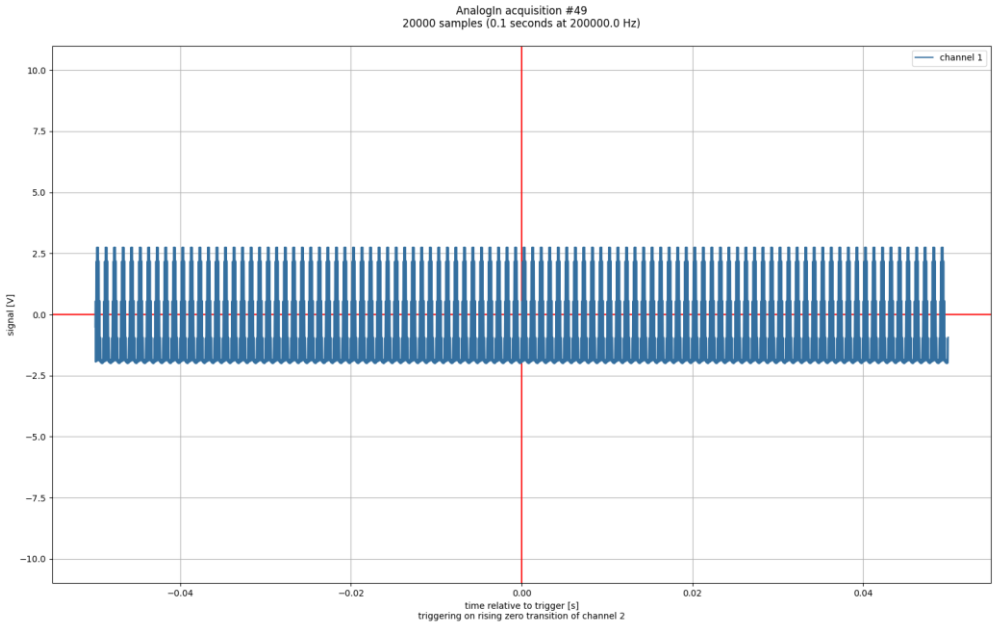
Task 5

Run AnalogOutPlayCustomWaveformModified.py. (available on eportal) Generate some custom waveform and feed it to the instrument.

```
import argparse
import sys

import numpy as np
import matplotlib.pyplot as plt

import time

from pydwf import (DwfLibrary, DwfEnumConfigInfo, DwfAnalogOutIdle, DwfTriggerSource, PyDwfError, DwfTriggerSlope, DwfState,
                   DwfAnalogOutNode, DwfAnalogOutFunction, DwfAnalogInFilter, DwfAcquisitionMode, DwfAnalogInTriggerType)
from pydwf.utilities import openDwfDevice


def demo_custom_analog_out_waveform(analogIn, analogOut, waveform, waveform_duration, wait_duration):
    """Put the given waveform on the first analog output channel."""

    CH1 = 0

    channel = CH1
    node = DwfAnalogOutNode.Carrier

    analogOut.reset(channel)

    # Show the run of the AnalogOut device on trigger pin #0.
    analogOut.device.triggerSet(0, DwfTriggerSource.AnalogOut1)

    analogOut.nodeEnableSet(channel, node, True)
    analogOut.nodeFunctionSet(channel, node, DwfAnalogOutFunction.Custom)

    # Determine offset and amplitude values to use for the requested waveform.

    (amplitude_min, amplitude_max) = analogOut.nodeAmplitudeInfo(
        CH1, node)  # pylint: disable=unused-variable           insert CH1 and node for proper working
```

Above is the code which is given on eportal.

Output:



Comments:

This codes shows the analog output of a custom waveform.

Task 6

Run DigitalIO.py. Try to modify the script so that the digital input states are stored in some data structure (e.g. numpy.ndarray). Experiment with the interval that the digital input is read.

```
"""Demonstrate the use of the DigitalIO functionality."""

import time
import argparse
import random

from pydwf import DwfLibrary, PyDwfError
from pydwf.utilities import openDwfDevice


Codeium: Refactor | Explain | ×
def demo_digital_io_api(digitalIO) -> None:
    """Demonstrate the Digital I/O functionality."""

    digitalIO.reset()

    print("Pins that support output-enable (i.e., tristate) functionality:")
    print()
    print("  outputEnableInfo (32 bit) ...... : {}0b{:032b}".format(32 * " ", digitalIO.outputEnableInfo()))
    print("  outputEnableInfo (64 bit) ...... : {}0b{:064b}".format( 0 * " ", digitalIO.outputEnableInfo64()))
    print()

    print("Pins for which output-enable is active (i.e. are not tri-stated):")
    print()
    print("  outputEnableGet (32 bit) ....... : {}0b{:032b}".format(32 * " ", digitalIO.outputEnableGet()))
    print("  outputEnableGet (64 bit) ....... : {}0b{:064b}".format( 0 * " ", digitalIO.outputEnableGet64()))
    print()
```

Comments: I run this code successfully and saw some binary numbers in terminal but I couldn't store them in a data structure.My code gave errors which I couldn't solve with DigitalIO.py but I uploaded normally how would I store input states in a data structure below.

```
import numpy as np
num_inputs = 5
input_states = np.zeros(num_inputs, dtype=np.int8)
input_states[0] = 1
input_states[1] = 1
input_states[2] = 0
input_states[3] = 1
input_states[4] = 0
print("Current input states:", input_states)
```

In this code I assumed I had five inputs and I assigned them one by one manually using np.zeros function.

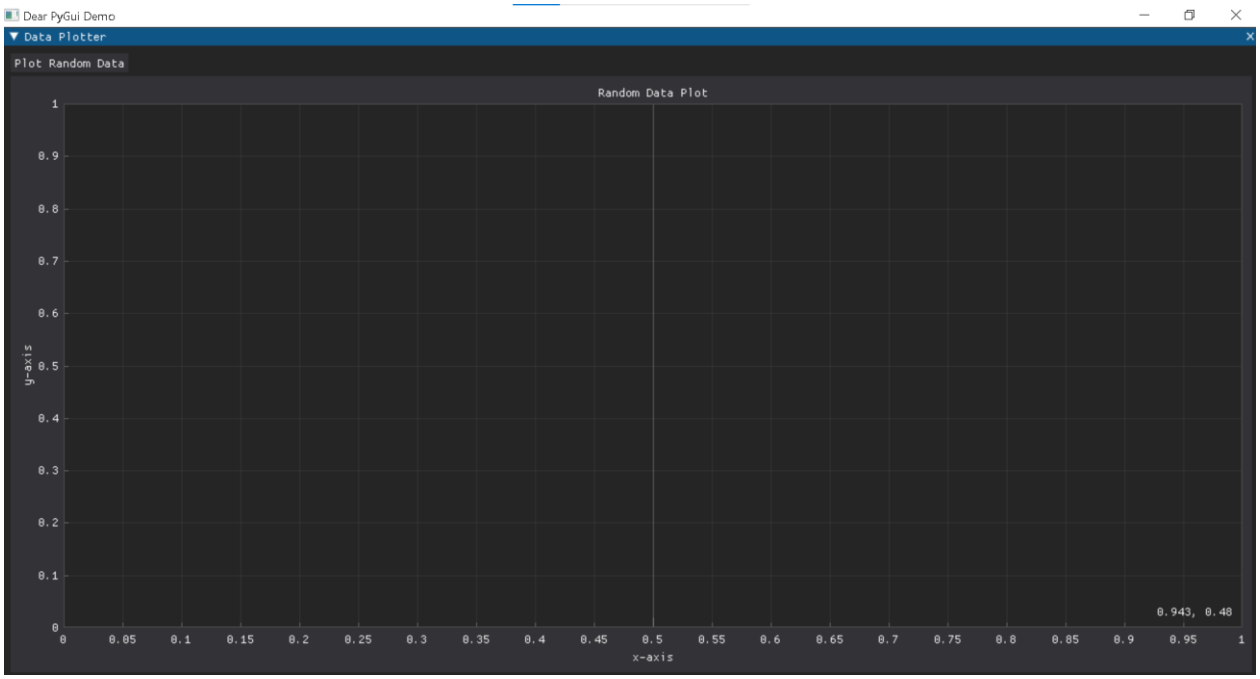
Task 7

Write a very simple GUI for plotting data from either task 4 or 6. It is recommended to use Dear PyGui [Dear PyGui documentation], but you can choose to use other GUI library/framework.

```
import dearpygui.dearpygui as dpg
import numpy as np

Codeium: Refactor | Explain | Generate Docstring | ×
def plot_random_data(sender, app_data, user_data):
    x = np.linspace(0, 10, 100)
    y = np.random.random(100)


    dpg.delete_series(plot_id, " Data")
    dpg.add_line_series(plot_id, " Data", x, y)


dpg.create_context()

with dpg.window(label="Data Plotter", width=600, height=400):

    dpg.add_button(label="Plot Random Data", callback=plot_random_data)


    with dpg.plot(label="Data Plot", height=-1, width=-1):
        dpg.add_plot_axis(dpg.mvXAxis, label="x-axis")
```

```
        plot_id = dpg.add_plot_axis(dpg.mvYAxis, label="y-axis")

dpg.create_viewport(title='Dear PyGui Demo', width=600, height=400)
dpg.setup_dearpygui()
dpg.show_viewport()
dpg.start_dearpygui()
dpg.destroy_context()
```

Output:



Comments:

This script uses the Dear PyGui library to create a simple graphical user interface (GUI) application for plotting random data. The GUI includes a window with a button labeled "Plot Random Data" and a plot area with x and y axes. When the button is clicked, a callback function generates a set of random x and y data points, updates the plot, and displays the result. The overall application operates within the Dear PyGui context, and the graphical interface is presented in a viewport. The script sets up the Dear PyGui event loop to handle user interactions, and upon closing the application, the Dear PyGui context is appropriately destroyed.

Conclusion

The Analog Discovery 2 stands out as an entry-level "lab-on-a-board," offering versatile functionalities such as a basic oscilloscope, logic analyzer, and function generator. Remarkably cost-effective compared to alternative tools, it consolidates multiple functions into one compact device. While it proves to be a valuable and feature-rich tool, one potential drawback lies in its software interface. Users may find the graphical user interface (GUI) somewhat lacking in certain aspects. Nevertheless, the hardware performs admirably, showcasing the convenience of having diverse tools integrated into a single device. Despite any GUI limitations, the Analog Discovery 2 effectively fulfills its intended tasks, making it a practical choice for various applications..