

Intern Exit | Presentation

Chengming Li



Agenda

1. About Me & Future Journey
2. Accomplished Projects
3. Key takeaways
4. Acknowledgment
5. Q&A

About me

Chengming(Steven) Li

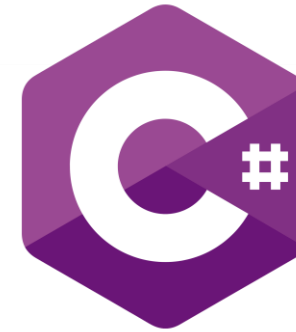
- BS Electrical & Computer Engineering Degree at University of Colorado, Boulder
- Will be MS Electrical & Computer Engineering Student at University of California, San Diego
- Here, at Eridan, RF Test Engineering Intern

Accomplished Projects

1. pyEridanLab

The Problem? Why?

- Cost of MATLAB License
- Transition from C# to Python
- How to use DLLs in Python



Accomplished Projects

1. pyEridanLab

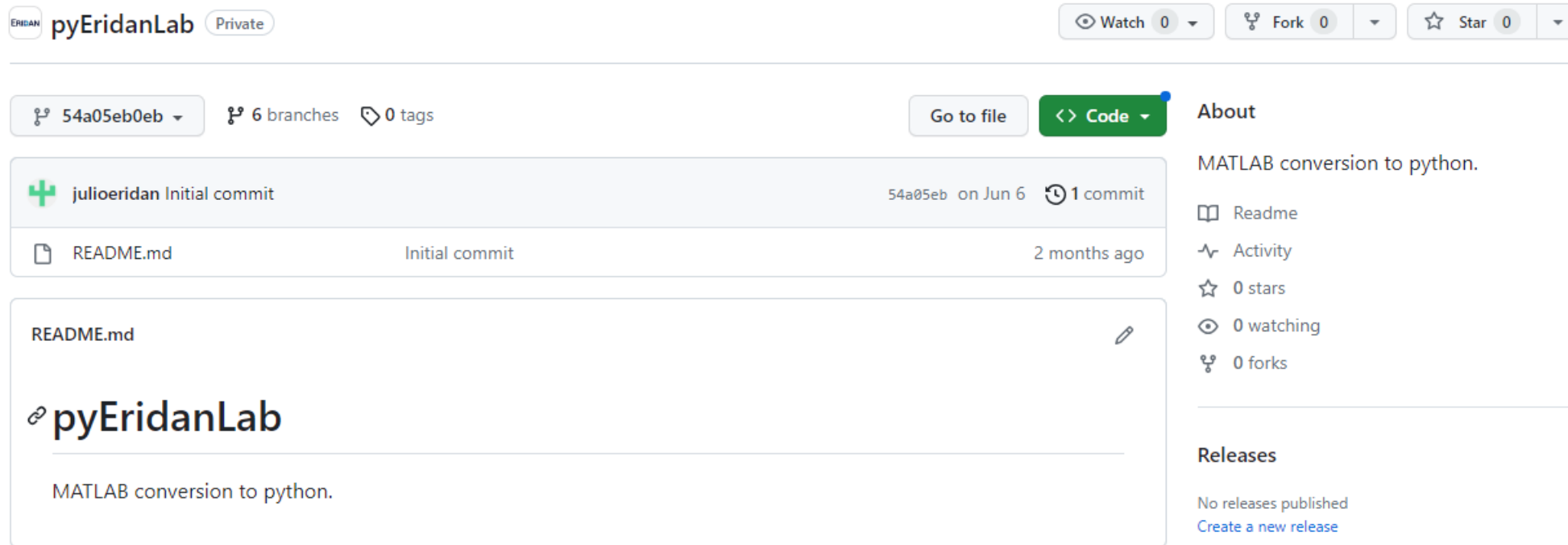
The Solution -> How !?:

- Cost of MATLAB License
 - Use C# and Python.
 - Open source,
 - Free to use.
- Transition From C# to Python
 - Use Dynamic Link Library (DLL)
 - .Net
 - pythonnet
- How to use DLLs in python
 - Type hint files, .pyi
 - Import clr

Accomplished Projects

1. pyEridanLab

- What it looks like at the beginning



The screenshot shows the GitHub interface for the `pyEridanLab` repository. At the top, the repository name `pyEridanLab` is displayed with a `Private` badge. To the right are buttons for `Watch` (0), `Fork` (0), and `Star` (0). Below this, a commit history bar shows a single commit by `julioeridan` with the message `Initial commit`, commit hash `54a05eb`, dated `on Jun 6`, and `1 commit` made `2 months ago`. A file list below the commit shows `README.md` as the `Initial commit`. The main content area displays the `README.md` file, which contains the repository name `pyEridanLab` and the description `MATLAB conversion to python.`. On the right sidebar, the `About` section repeats the description `MATLAB conversion to python.` and lists repository statistics: `Readme`, `Activity`, `0 stars`, `0 watching`, and `0 forks`. The `Releases` section indicates `No releases published` and provides a link to `Create a new release`.

Accomplished Projects

1. pyEridanLab

- After a month effort...
- Ignore 65621 additions, which are dlls
- 8265 lines of code, wrap everything from C#

Showing 101 changed files with 73,886 additions and 1 deletion.

73886 - 65621 =
8,265

The screenshot shows the GitHub repository for pyEridanLab. The repository is private and has 11271073e3 as the latest commit, 6 branches, and 0 tags. The file list includes .vscode, Help, MainAndPyi, Pyenv_Config, TestExecutiveDLL, .gitignore, README.md, and pyEridanLab.code-workspace. The README.md file is open, showing the project's purpose: a Python Development Repo for importing and using all DLLs from C# in Python. It provides instructions on using Typehint file .pyi to wrap functions from C# DLLs and links to help for DLL or general Python help. The README also includes a section for Python Environment Configuration, stating that Python 3.11.4 and Virtual Studio Code are used for Python Development. It lists steps to setup the Python Environment, including downloading Pyenv_Config and installing it. The right sidebar shows repository statistics: 0 stars, 0 watching, 0 forks, 0 releases, 0 packages, 2 contributors (stevenieridan and julioeridan), and a language distribution chart showing Python at 88.1%, Jupyter Notebook at 10.6%, and Batchfile at 1.3%. Suggested workflows for Actions Importer, Python application, and Pylint are also visible.

Accomplished Projects

1. pyEridanLab

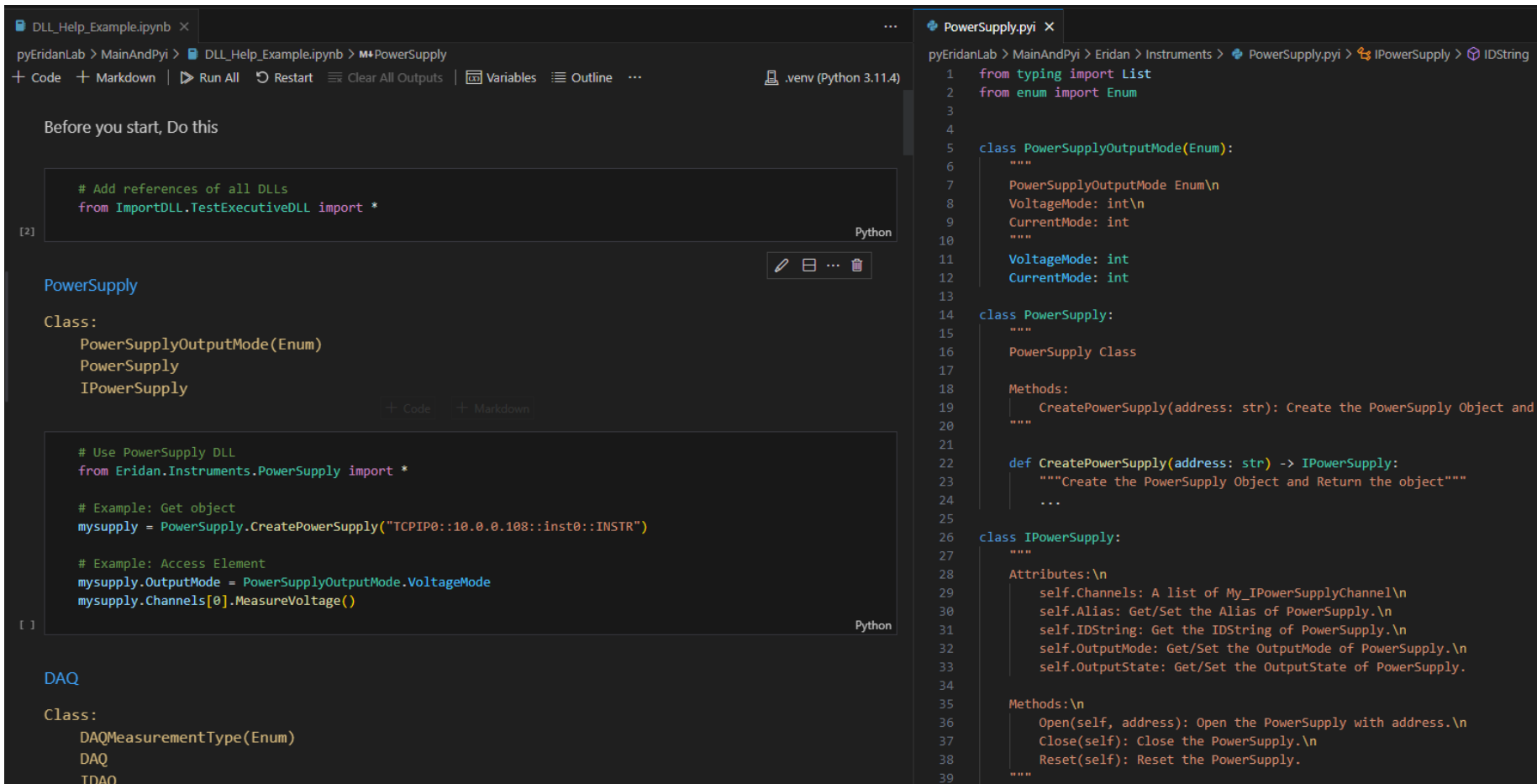
- Double click to install everything as you need

Name	Last commit message	Last commit date
..		
activate_env.bat	Newwrapper (#2)	2 weeks ago
rclone_install.ps1	update the name	3 days ago
requirements.txt	Newwrapper (#2)	2 weeks ago
step1_python3.11_install.bat	Newwrapper (#2)	2 weeks ago
step2_vscode_install.bat	Newwrapper (#2)	2 weeks ago

Accomplished Projects

1. pyEridanLab

- Need help for DLLs usage? --> **DLL Help Script!**



```
pyEridanLab > MainAndPy > DLL_Help_Example.ipynb > M*PowerSupply
+ Code + Markdown | ▶ Run All ⌂ Restart ⌂ Clear All Outputs | Variables Outline ... .venv (Python 3.11.4)

Before you start, Do this

[2] # Add references of all DLLs
    from ImportDLL.TestExecutiveDLL import *

Python

PowerSupply

Class:
    PowerSupplyOutputMode(Enum)
    PowerSupply
    IPowerSupply

+ Code + Markdown

# Use PowerSupply DLL
from Eridan.Instruments.PowerSupply import *

# Example: Get object
mysupply = PowerSupply.CreatePowerSupply("TCPIP0::10.0.0.108::inst0::INST")

# Example: Access Element
mysupply.OutputMode = PowerSupplyOutputMode.VoltageMode
mysupply.Channels[0].MeasureVoltage()

Python

DAQ

Class:
    DAQMeasurementType(Enum)
    DAQ
    IDAQ
```

```
PowerSupply.py
pyEridanLab > MainAndPy > Eridan > Instruments > PowerSupply.py > IPowerSupply > IDString

1 from typing import List
2 from enum import Enum
3
4
5 class PowerSupplyOutputMode(Enum):
6     """
7     PowerSupplyOutputMode Enum\n
8     VoltageMode: int\n
9     CurrentMode: int
10    """
11    VoltageMode: int
12    CurrentMode: int
13
14 class PowerSupply:
15     """
16     PowerSupply Class
17
18     Methods:
19     | CreatePowerSupply(address: str): Create the PowerSupply Object and
20     """
21
22     def CreatePowerSupply(address: str) -> IPowerSupply:
23         """Create the PowerSupply Object and Return the object"""
24         ...
25
26 class IPowerSupply:
27     """
28     Attributes:\n
29     | self.Channels: A list of My_IPowerSupplyChannel\n
30     | self.Alias: Get/Set the Alias of PowerSupply.\n
31     | self.IDString: Get the IDString of PowerSupply.\n
32     | self.OutputMode: Get/Set the OutputMode of PowerSupply.\n
33     | self.OutputState: Get/Set the OutputState of PowerSupply.
34
35     Methods:\n
36     | Open(self, address): Open the PowerSupply with address.\n
37     | Close(self): Close the PowerSupply.\n
38     | Reset(self): Reset the PowerSupply.
39     """
```

Accomplished Projects

1. pyEridanLab

- Need help for Python usage? --> **Python Help Script!**

The screenshot displays two side-by-side windows. The left window is a JupyterLab workspace titled 'pyEridanLab (Workspace)'. It contains a code editor with a Python script for installing and using a library, and a terminal window showing the execution of the script. The right window is a web browser showing the Matplotlib gallery page, which displays various example plots and their corresponding source code.

JupyterLab Code Editor:

```
pyEridanLab > Help > Py_Helpipynb > Import python lib (Click me for more information)
+ Code + Markdown | Run All | Restart | Clear All Outputs | Variables | Outline ...
.venv (Python 3.11.4)

Import python lib (Click me for more information)

# If the library is not installed in this library
# Type this in terminal:
#
# pip install mymodule_name
#
# Replace "mymodule_name" with the actual module name

# Using:
# import mymodule_name
#
# Or:
# from mymodule_name import mymodule_function
#
# Replace "mymodule_name" with the actual module name
# Example:

import numpy
from numpy import *
```

Terminal Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
pyparsing-3.0.9 python-dateutil-2.8.2 pythonnet-3.0.1 pywin32-306 pyzmq-25.1.0 six-1.16.0 stack-data-0.6.2
tornado-6.3.2 traitlets-5.9.0 wcwidth-0.2.6
[notice] A new release of pip is available: 23.1.2 -> 23.2.1
[notice] To update, run: python.exe -m pip install --upgrade pip
Terminal will be reused by tasks, press any key to close it.
```

Web Browser (matplotlib.org):

Examples

This page contains example plots. Click on any image to see the full image and source code.

For longer tutorials, see our tutorials page. You can also find external resources and a FAQ in our user guide.

Lines, bars and markers

Bar color demo, Bar Label Demo, Stacked bar chart, Grouped bar chart with labels, Horizontal bar chart


Broken Barh, CapStyle, Plotting categorical variables, Plotting the coherence of two signals, CSD Demo


Curve with error band, Errorbar limit selection, Errorbar subsampling, EventCollection Demo, Eventplot demo

Accomplished Projects


1. pyEridanLab

- Hands off? --> Confluence Page



Project status
Aug 7, 2023


Accomplishments


1. All DLLs are imported in .pyi form
2. DLL & Python Help Scripts
3. Python, VScode and rc1one installation scripts finished
4. README about repo usage
5. Test team have cloned the pyEridanLab repo to their Lab PCs


Next steps

- ☐ Put more documentation in the Help scripts (Variable Type Conversion)
- ☐ Collect feedback about user's experience
- ☐ Interface with hardware
- ☐ Duplicate test sequence using Python
- ☐ Risks in the right side ----->


Periodic
Aug 7, 2023

- ☐ Periodic Maintenance
- ☐ requirements.txt
- ☐ .pyi
- ☐ DLL version



Risks, constraints & project issues


1. **Maintenance:**
 - a. update .pyi when corresponding C# file is updated
 - b. requirements.txt
 - c. DLL Version
2. **Naming issue of DLLs:**
 - a. two dot in the DLL file's name
 - b. where to store DLL-related scripts (they can only be in /MainAndPyi right now, no sub-folder is allowed)
3. **Variable Type Conversion:**
 - a. C# variable type to Python variable type
4. **Import Warning:**
 - a. regarding #3, this reports a warning, but runs freely. (Python can't understand these from C#)

```


1  # from System import ValueTuple, UInt32, Int32
2  # from System.Collections.Generic import List

```
5. **Repo Growth:**
 - a. regarding #2,b, repo will get messy after lots of DLLs scripts are created



Repo Usage


How to use pyEridanLab

1. Follow the Python Environment Configuration in repo's README.md
2. For .ipynb file
 - a. Before you run, on top right side, Choose Select Kernel → python environment → .venv (recommended)
 - b. + code is used to create code section
 - c. + markdown is used to create header/comment
3. For .py file
 - a. on top right side, Click triangle button to run the script


What to Add?

1. For None-DLL scripts, it is recommended to put the scripts inside the **/MainAndPyi**, and can also be stored in a sub-folder
2. For DLL scripts, it must be stored inside the /MainAndPyi, **(NO SUB-FOLDER!)**
3. For help script, if you find any common problems with solutions, please do add them to the **Help/Py_Help.ipynb**


Note

1. You can skip step4 in README if you already have VScode installed on your computer
2. Please comment line12 - 14 in **.vscode/tasks.json** if you don't want the VSCode to reinstall the library and virtual environment every time. Note: please don't commit this change to the remote repo
3. Please **CLOSE ALL** current open files in vscode if you are reopening the workspace with newer **requirements.txt**

Accomplished Projects

2. Temperature Chamber

The Problem? Why?

- Oscillation during the heating/cooling
- How to know if there is an error in 30s
 - Whether the hardware switch is on
 - Whether it is going to the right direction
- How to handle the overshooting



Accomplished Projects

2. Temperature Chamber

The Solution -> How !?:

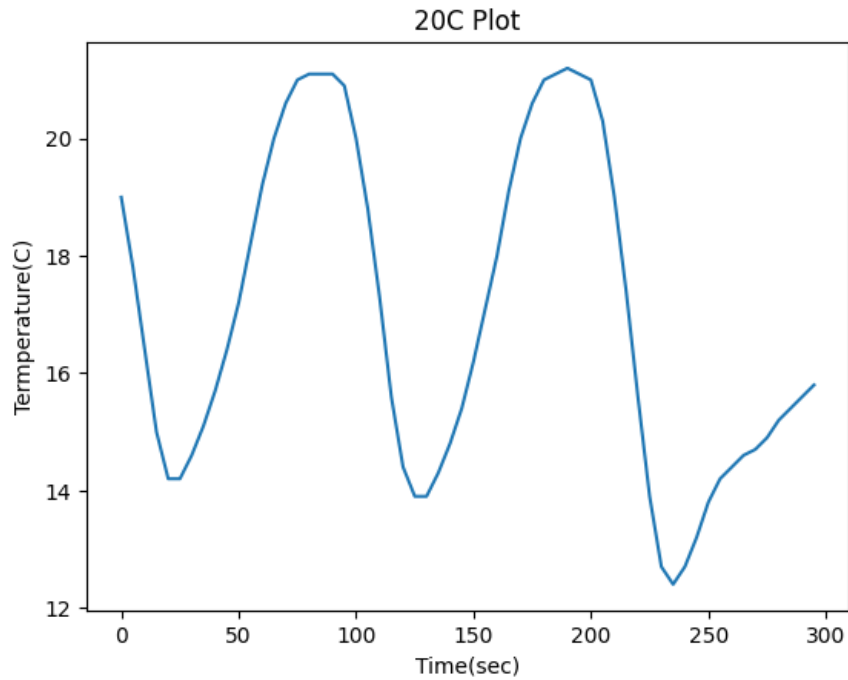
- Oscillation during the heating/cooling
 - AutoTune-off
- How to know if there is an error in the 30s
 - Whether the hardware switch is on
 - Check if the temperature change after starting the program
 - Whether it is going in the right direction
 - Compare the delta between the current temp and the target temp
- How to handle the overshooting
 - Make sure it converges for 30s



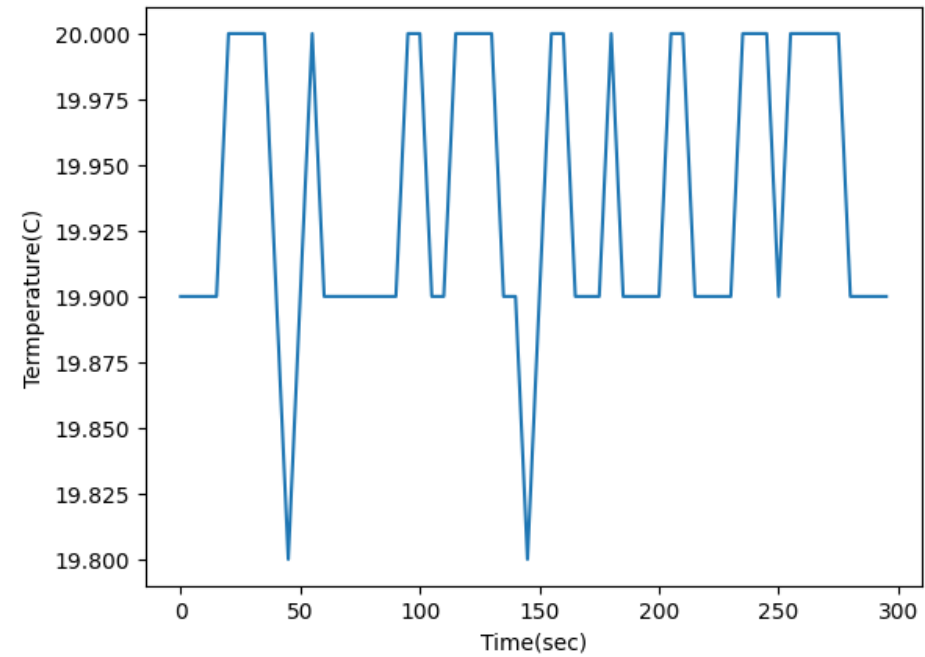
Accomplished Projects

2. Temperature Chamber

Before



After



Accomplished Projects

3. Digital Multimeter

The Problem? Why?

- Read Temperature from DMM
- Different Models support different Thermo Probes



Accomplished Projects

3. Digital Multimeter

The Solution -> How !?:

- Read Temperature from DMM
 - Create GetTemperature() Function in C#
- Different Models support different Thermo Probes
 - “Switch” Statement to choose ThermoProbe for different models



```
public enum ThermoProbeType
{
    /// <summary>
    /// This Enum is only used for Temperature measurement
    /// No need if other measurement are performed
    /// Command:
    /// <probe_type>: {FRTD|RTD|FThermistor|THERmistor|TCouple}. Default: FRTD.
    /// <type>: 85 (only possible value for RTD/FRTD), 5000 (only possible value for THERmistor/FThermistor), or E, J, K, N, R, T (TCouple).
    ///
    /// 34420A uses THERmistor|FRTD|TC
    /// 34450A uses THERmistor
    /// 34460A and 34461A only use FRTD|RTD|FThermistor|THERmistor
    /// 34465A and 34470A only use FRTD|RTD|FThermistor|THERmistor|TCouple
    /// </summary>

    fourWireRTD_85,
    RTD_85,
    fourWireThermistors_5000,
    thermistors_5000,
    thermoCouple_J,
    thermoCouple_K,
    thermoCouple_E,
    thermoCouple_T,
    thermoCouple_N,
    thermoCouple_R,
    // Additional probe for 34420A only
    thermoCouple_S,
    thermoCouple_B
}
```


Key takeaways

1. Be Proactive
2. Take the Responsibility
3. Try First
4. One Problem/Thing at a time
5. Be Precise/Easy to Understand

What skills do I learned

1. C#
2. GitHub
3. Python Programming Practice
4. Instrument Interface
5. Batch Script

Acknowledgment

- Shawn
- Julio
- Tanner
- Angel
- Jason
- Hala
- Eridan



Q&A

