SSC24-P2-21

An Open-Source Python-Based Coding Strategy for Efficiently Developing Test Software for CubeSats

Paula do Vale Pereira[1], Felipe Depine[2], Roberto Bucher[3]

[1] *Assistant Professor of Aerospace Engineering, Florida Institute of Technology, pdovalepereira@fit.edu*

[2] *Owner and Engineer, Robots5 LLC*

[3] *Retired Professor, University of Applied Sciences of Southern Switzerland (SUPSI)*

Key takeaway:

pysimCoder could be a time-efficient, user-friendly, open-source way of writing ground test software for CubeSats.

Introduction

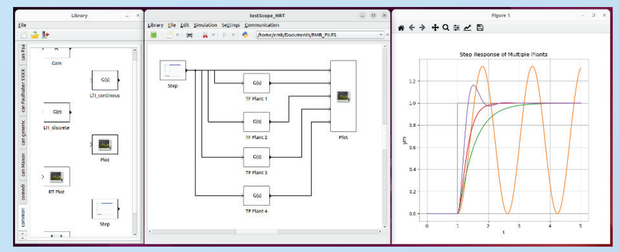
- Command and Data Handling (CNDH) has been a common cause of delays and failure among CubeSats.
- Ground testing of components and subsystems should happen throughout the development process and should not need to wait for the final flight software (FSW) to be ready.
- pysimCoder could be a solution for quickly preparing for ground tests, given its user-friendly coding platform and open-source framework.

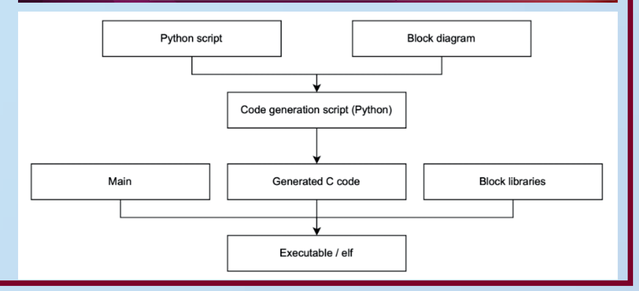
Conclusions and Future Work

- Based on Python and a visual drag-and-drop framework, pysimCoder simplifies the coding process and could facilitate and accelerate software implementation in CubeSat university teams.
- With pysimCoder, a working software for ground tests could be written in a day.
- We are working on comparing coding time with F' and C, and procuring more realistic CubeSat hardware for testing.

About pysimCoder

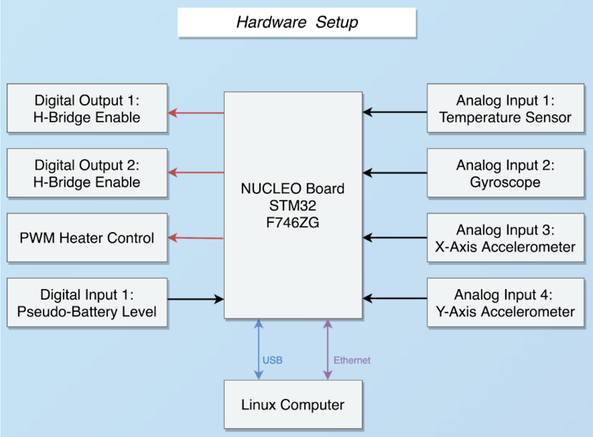
- pysimCoder is typically used to perform dynamic systems modeling, simulation, control, and real-time code generation as an alternative to expensive licensed programs such as Simulink.
- Software components are building blocks that can be assembled as block diagrams to design schematics for code generation.
- The code is written on a host Linux computer and, once generated, deployed to a target where it runs in real-time.
- Python scripts can be attached to the block diagram, allowing the user to import and use numerous Python libraries and APIs.
- In the background, pysimCoder creates an executable C code without the involvement of the user.
- Silicon Heaven can be used to change parameters on the fly.

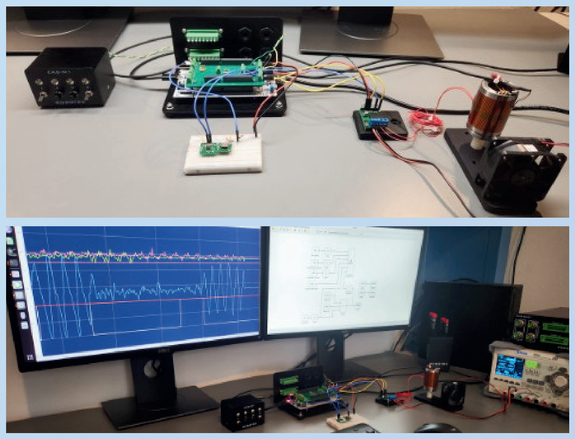


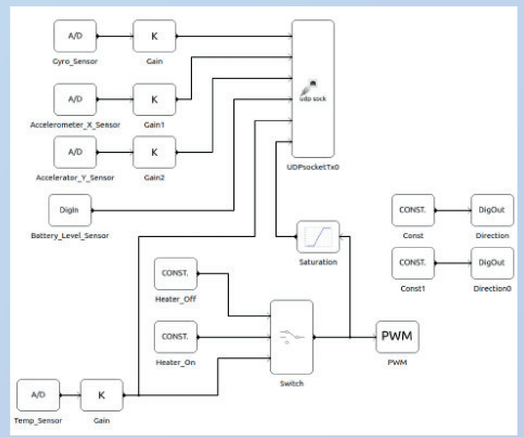


Preliminary Results

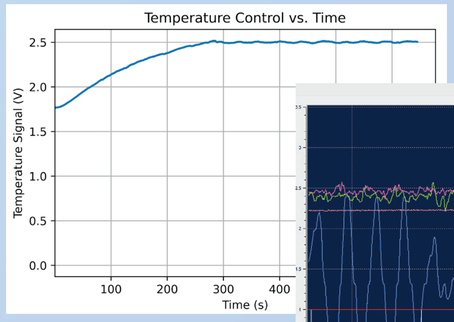
- Pseudo-flatsat assembled to test pysimCoder: STM32 (Nucleo) as on-board computer, two accelerometers and a gyroscope (analog inputs) as ADCS, a thermistor (analog input) and a PWM controlled heater (digital output) as thermal control, a pseudo-battery signal (digital input represented as a manual switch) as power.
- The simplified ADCS board was moved by hand, while the heater was controlled to turn on/off based on the data from the temperature sensor (closed-loop control). Manual switches on the interface module (EMB-IM1) were turned on/off to represent a low battery condition.
- The pysimCoder block diagram shown below receives and plots the signals.

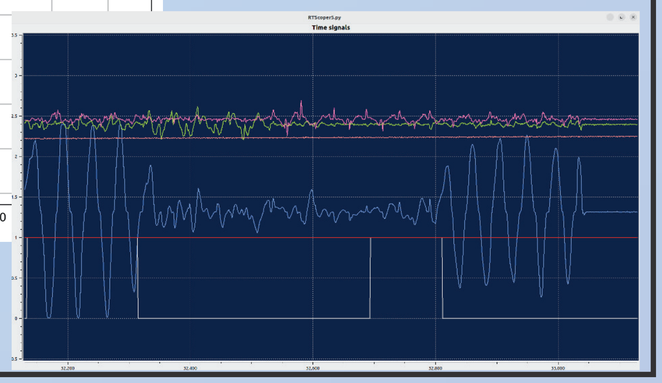






- The plots show the data from one of the experiments: the pink and green lines are accelerometers, the blue line is the gyroscope, the salmon line is the temperature, the white line is the manual switch data, and the red line is the PWM state.
- The heater control used the raw thermistor voltage data and was set to 2.5 V. The PWM control kept the signal at the desired level.





38th Annual Small Satellite Conference. Logan, Utah, 2024