A Flexible CubeSat Education Platform Combining Software Development and Hardware Engineering

Daniel Schloms[1], *David Freismuth*[1], *Jakob Riepler*[1], *Raphael Böckle*[1]

Abstract

While many secondary schools offer courses or extracurricular activities that focus on satellite engineering, e.g. CanSats or the assembly of ground stations, these projects usually stay close to ground. With SpaceTeamSat1, the TU Wien Space Team wants to enhance this approach and tackle the challenge to perform various experiments in space, enabling students to participate in a space mission that actually orbits our planet. Therefore, our goal is to develop a 1U CubeSat platform, which allows students at secondary schools to access a set of different sensors connected to a Raspberry Pi. Consequently, students can write their own software experiments in Python and exploit the possibilities of sensors in space.

In this context, participation happens at different stages: For one, students are getting in contact with Python, which also allows an easy step into software engineering paradigms. Moreover, our team will pose some challenges, such as re-doing an earlier satellite mission and giving impressions about how CubeSats can be used, e.g. to combat climate change. To complete these challenges, the CubeSat is equipped with various sensors such as temperature sensors, gyrometers, magnetometers, as well as two cameras. Moreover, the participating students also have the possibility to design their own experiments independently to leave room for creativity.

Further enhancing this educational mission, participating students are also invited to work on hardware topics. This is mainly aimed at engineering schools, which are encouraged to assemble Raspberry Pi HATs which contain the actual mission sensors, as well as a SatNOGS ground station, which also enables students to get an insight on satellite communication. It needs to be considered that the educational mission follows a modular setup since the combination of all individual tasks is not realizable within a single school year. Thus, schools are also able to individually select appropriate tasks.

In the past we were already collaborating with the European Space Education Resource Office as we are acting as launch provider of CanSats for ESERO's Austrian CanSat competition. In this sense, STS1 shall be an extension to the space educational program in Austria. Based on that, we believe that the STS1 mission has a high potential to bring something that is currently out of reach for most people, outer space, closer to a demographic with a lot of talent and enthusiasm for engineering and potential future engineers.

Keywords

CubeSat, Python, Raspberry Pi, SatNOGS, software engineering

[1] TU Wien Space Team, Austria
Corresponding author: daniel.schloms@spaceteam.at

Acronyms/Abbreviations

COBC	*Communication module and on-board computer*
EDU	*Education module*
EPS	*Electrical power system*
ESA	*European Space Agency*
ESERO	*European Space Education Resource Office*
FW	*Firmware*
GS	*Ground station*
HAT	*Hardware attached on top*
LNA	*Low noise amplifier*
PCB	*Printed circuit board*
RBF	*Remove-Before-Flight (pin)*
RF	*Radio frequency*
RTOS	*Real time operating system*
RODOS	*Realtime Onboard Dependable Operating System*
ROS	*Robotics Operating System*
SatNOGS	*Satellite Networked Open Ground Station*
SBC	*Single board computer*
SDR	*Software defined receiver*
SPI	*Serial peripheral interface*
STS1	*SpaceTeamSat1*
TUST	*TU Wien Space Team*
UART	*Universal asynchronous receiver / transmitter*
UCI	*Umbilical cord interface*

1. Introduction

The TU Wien Space Team (TUST) is constantly expanding their breadth of topics, and since we are already working with students to some extent via ESA's CanSat competition in Austria, we are planning on an extension of the CanSat mission by launching an educational CubeSat mission dedicated for Austrian high schools. While there are many educational missions, e.g. Astro Pi,[1] deep insights from an educational point of view on space missions are still elusive. Therefore, the primary goal is to build and operate a self-developed CubeSat, and to enable students at secondary schools to run own-developed software on our satellite. Moreover, the space mission design shall also incorporate students to a certain extent and thus, students shall gain insights on the operation of such a mission. This combination shall help students to get a deeper knowledge on space technology as well as software engineering. The idea is also inspired by the increasing popularity of platforms such as Arduino[2] or Raspberry Pi[3], which a lot of students are already familiar with. Consequently, we started developing a satellite around the idea of a Raspberry Pi that can run programs using various instruments on board. For example, cameras, temperature sensors or magnetometers will be implemented to gain various insights. To further involve students, a pilot school supports us in selecting appropriate sensors. For ground communication, we are implementing a ground station based on the SatNOGS[4] stack, including a transmitter. This shall even deepen the understanding of space technology by actually getting in contact with space RF communication. Furthermore, students of technical high schools are invited to build and operate their own SatNOGS ground station.

The paper is structured as follows:

In Section 2 the structure as well as the utilized subsystems of the satellite are outlined and described. Section 3 discusses the education aspect and describes its high variety. In Section 4, a self-developed game, the dataflow game, is presented. This topic is also of high importance from an educational point of view as it shows how to find errors in a CubeSat design and outlines a way on how to add additional functionality to subsystems. This tool shall also be presented to students, as it depicts a way to overcome complex issues and paves an understandable way to define software architectures. This tool shall also be presented to pupils, as it depicts a way to overcome complex issues and paves an understandable way to define software architectures. In Section 5, the current results given by our prototype are presented and the impact this mission has is discussed. Finally, Section 6 concludes the discussed topics and gives an outlook.

2. Subsystems

2.1. System Architecture

Our 1U CubeSat is divided into multiple subsystems, each performing different tasks that can be combined under a common theme. Here, we use a combined communication module and on-board computer (COBC), an electrical power system (EPS), and an

education module (EDU). The following diagram shows the subsystems and its communication interfaces.

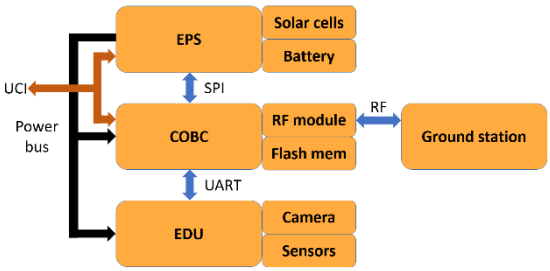

Figure 1. System architecture of STS1.

The EPS distributes power to the COBC and EDU via the power bus while also being capable of charging its batteries with solar cells. The COBC gets housekeeping data from the EPS via a serial peripheral interface (SPI), and it uses the universal asynchronous receiver / transmitter (UART) protocol to communicate with the EDU. Importantly, the COBC acts as the master and merely operates the EDU on command as a slave. However, educational software as well as results are managed and stored on the flash memory integrated on the COBC. Lastly, radio frequency (RF) communication is used to send data from and to the ground station. Additionally, STS1 features an umbilical cord interface (UCI), that is mainly used to charge the batteries (EPS) and program the microcontroller used on the COBC.

2.2. Electrical Power System (EPS)

The EPS distributes power to the subsystems and contains the system level safety features. We use a Remove-Before-Flight (RBF) pin, deployment switches, and a deployment timer, which shall prevent a premature activation of the CubeSat. In general, the satellite is powered by solar cells, which are also capable of charging a battery which is used to power the subsystems in parts of the orbit without light incidence. Importantly, no microcontroller is used, ensuring highly reliable operation. To further increase reliability only a single voltage is supplied by the EPS.

2.3. Communication Module and On-board Computer (COBC)

While other missions might separate communication modules and the on-board computer into distinct subsystems, we decided to combine both areas into a single subsystem. Here, the combination is possible since experiments are performed respectively executed merely on the education module (EDU).

On the on-board computer we use an STM32F411RE microcontroller in conjunction with a real time operating system (RTOS) called RODOS, which was developed especially for space applications where a high dependability is needed.[5] This operating system is also used by the German Aerospace Center[6], as well as in other satellite missions such as TechnoSat[7] launched by TU Berlin. The COBC will be responsible for receiving and transmitting data via a RF module, managing memory access, deploying the antenna, and managing the Python files as well as the result files to respectively from the EDU. Furthermore, it collects housekeeping data, which is then combined into a beacon and sent to earth periodically.

The commercial off-the-shelf RF module transmits data at 434 MHz band, which is an amateur radio band. Again, from an educational point of view this shall envision students the possibilities of amateur radio technology.

2.4. Education Module (EDU)

The EDU handles the educational task apart from developing and operating a satellite from scratch. It contains a Raspberry Pi Compute Module 3+ which enables students to perform their own experiments in space. Students are able to submit Python programs which can use an array of sensors and a camera to collect data and take pictures. The resulting data is stored and transmitted back to earth, where further analysis and evaluation can happen. Ambitious students may also use the Raspberry Pi on-board the CubeSat for pre-evaluations and further decision making in their Python software experiment.

For the execution of the student software, we are using the Robotics Operating System (ROS).[8] This allows us to easily handle sensor data via a publisher-subscriber pattern. Moreover, it can be used seamlessly with Python.

2.5. Ground Station

The primary ground station handles all uplink commands as well as is capable of receiving downlinked data. To further increase downlink capabilities, we also include SatNOGS, which is an open-source satellite ground station network, striving to build a free to use receive-only network of ground stations around the globe.[4] Importantly, these ground stations are easy to build and are an extension to our educational mission, as schools shall be encouraged to build and operate their own SatNOGS stations.

It consists of a helical antenna for the 70 cm satellite RF band, rotatable using an antenna rotor. The signal is amplified using a low noise amplifier (LNA) and then fed into a HackRF-One[9] software defined receiver (SDR). The software for satellite tracking as well as data handling is controlled by a Raspberry Pi 4 single board computer (SBC).

As a RF uplink for our satellite is also required, we plan on extending the capabilities of the standard SatNOGS software stack with transmission capabilities. Therefore, a simple solution is to add a power amplifier to the HackRF-One's transmission output. By doing so, legal aspects also need to be considered. A diagram of the data flow between the ground station and the CubeSat STS1 is shown in Figure 2.

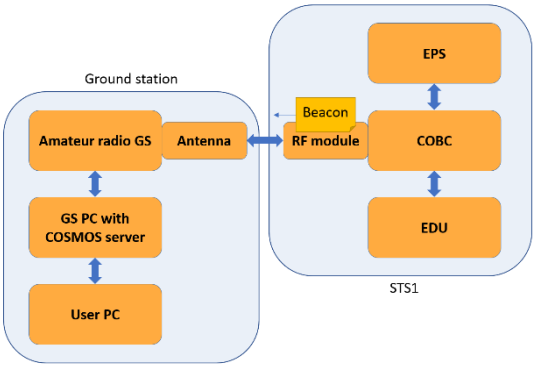

Figure 2. Data flow interfaces of the ground station and the CubeSat STS1.

While our SatNOGS ground station has up- and downlink capabilities, we also plan for students to build their own receive-only SatNOGS ground station. Furthermore, Figure 2 also symbolizes part of our design process that we want participating students to use, dataflow games. It works by people acting as subsystems and giving data notes to recipient subsystems, as demonstrated with the "Beacon" note. A person acting as the COBC, for example, will thereby assemble a data packet (beacon) from housekeeping data by creating a note and then drag it to the RF module, where the next person performs the steps necessary to send it to the ground station.

3. Education

3.1. Educational Mission

Figure 3 shows various aspects of the educational mission, where the CubeSat educational mission stands in the center of the mission. Our educational mission can be split into three types of objectives: scientific objectives, software objectives, and hardware objectives.

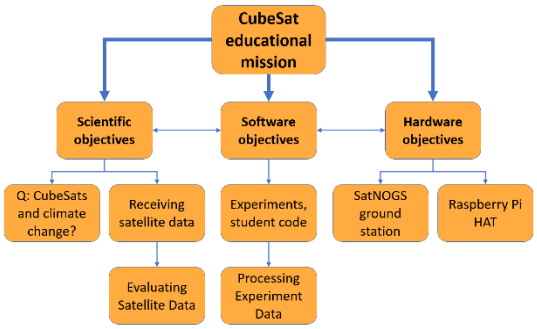

Figure 3. Modular approach of the educational mission.

We realize that working on topics of all objectives is not feasible for most students within one school year. Therefore, a modular approach is proposed. This will mostly affect the hardware topics, as they may be difficult to approach for students that do not attend engineering schools with the necessary teaching staff and equipment. Those groups will receive pre-assembled development hardware (Raspberry Pi HATs with the missions sensors) and the satellite communication will run entirely through the TUST. However, students are encouraged to focus on scientific topics and the design of their experiment instead, as this is a perfect opportunity to write their final thesis. Students from engineering schools on the other hand can also work on hardware challenges. This includes assembling the PCBs for their development hardware and most importantly, building a SatNOGS ground station. While transmissions to the satellite will still be operated by the TUST, this enables students to receive satellite data on their own, which also includes other satellites registered in the SatNOGS network. As with the final thesis before, students from engineering schools can easily use their work with STS1 as a final year project.

3.2. Predefined Challenges

Moreover, while we want students to express their creativity with this project, we will also pose some mandatory and optional challenges. Those challenges include topics such as climate change, which is something that we believe should be part of the project, or recreating an older satellite mission. This has the added benefit of creating a guided path, which offers an easier start for beginners. Additionally, students gain experience for new experiments. Lastly, it also allows the TUST to offer more appropriate support in the case problems arise.

4. Dataflow Games

Dataflow games are one of our most important tools to find flaws in the satellite design. As mentioned before, it is based on the idea of people acting out subsystems or even just parts of a subsystem. One starts out by choosing a scenario to run through, such as sending a Python file to the COBC for storage and further execution on the EDU. Within the scenario, every actor will act out the steps that their subsystem would take, e.g. the ground station is responsible for encoding and signing the file, then transmitting it at a certain time stamp, when the CubeSat is in sight. Utilizing this tool also helps to understand the challenges and requirements for satellite missions. Therefore, our dataflow game is also part of the educational objective for students. In a general introduction at the beginning of the actual mission, this game will be played in a simplified manner with students to introduce them to the delicate and fascinating topic of CubeSat mission design.

The TUST used an online whiteboard tool for documentation. Therefore, we can keep track of messages and the system state by putting notes with relevant information into the relevant place, for example a beacon getting sent down will be modeled as a note that gets passed down from subsystem to subsystem until it has reached its destination, as seen in Figure 4. Memory contents, such as Firmware (FW), that stay the same for some time are also depicted with notes. An example is shown in Figure 4.

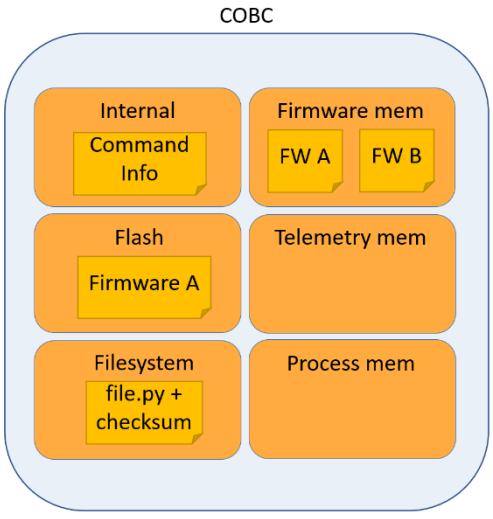

Figure 4. An example of COBC memory states at a certain time within the dataflow game.

For more complex processes within a subsystem, the use of flowcharts is more appropriate. This approach made it possible to flesh out the subsystems before submitting our design to external reviewers. We will also encourage students to use this approach, as it shows flaws in the design and the processes very quickly. Therefore, it can teach beginners how to work on complex projects and derive software and hardware architectures from specified behavior.

5. Results & Discussion

Our results so far are considerable. Before constructing the actual CubeSat, we are working on a prototype board which shall have the same functionality as STS1, but all the subsystems are mounted on an easily accessible PCB. At the time of writing, we could verify that the design is operational, and we are running a demonstration project. This includes sending simple commands to the COBC via UART and performing tasks such as reading the reset counter from a backup register or controlling the EDU from the COBC and receiving its data.

With other space education programs getting traction and STS1 already generating much interest from schools we can see that projects like ours are in demand by students. Even at this stage we are cooperating with a pilot school that has committed to participate in an extensive capacity, aiming to work on all topics, including the SatNOGS ground station. This is not surprising, as space is a topic that has always been popular among every age group, as demonstrated by the number of students participating in ESERO's CanSat competition every year, for which the TUST launches the rockets in Austria. We believe that giving students hands-on experience has the potential to make an impact on their future field of study or profession. Since the TUST can only work with a limited number of groups within Austria, we encourage other university engineering teams to think about developing their own educational projects.

6. Conclusions

We believe that our CubeSat mission is very promising, further enhancing the scope of space education. Due to its modular approach with different tasks, it allows us to bring space education to the next step. Consequently, we are not merely constructing a working prototype with a demonstration program. Moreover, we already received feedback from schools, which are interested in participating. We designed a

satellite system from scratch and employed dataflow games successfully to debug our design. The first pilot school has also already received a demonstration of the prototype and their students are given the opportunity to write their diploma theses on their involvement with the STS1 project.

References

[1] Astro Pi Website: https://astro-pi.org, last visited: 17th March 2022.

[2] Arduino Website: https://www.arduino.cc, last visited: 17th March 2022.

[3] Raspberry Pi Website: https://www.raspberrypi.com, last visited: 17th March 2022.

[4] SatNOGS Website: https://satnogs.org, last visited: 9th March 2022.

[5] S. Montenegro, F. Dannemann, RODOS – Real Time Kernel Design for Dependability, *DASIA 2009 – Data Systems in Aerospace*, Istanbul, 2009.

[6] DLR Core Avionics – RODOS Website: https://www.dlr.de/sc/desktopdefault.aspx/tabid-1262/1765_read-15871, last visited: 20th March 2022.

[7] R. Gerlich et al., Verification of the C++-Operating System RODOS in Context of a Small-Satellite, *ARCS Workshop 2018; 31st International Conference on Architecture of Computing Systems*, Braunschweig, 2018.

[8] ROS Website: https://www.ros.org, last visited: 9th March 2022.

[9] HackRF-One Website: https://greatscottgadgets.com/hackrf/one, last visited: 9th March 2022.