UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Tiit Vaino

Sun sensor software development for ESTCube-2

Bachelor's Thesis (9 ECTS)

Supervisor(s): Hans Teras, MSc
Kristo Allaje, MSc

Tartu 2022

Sun sensor software development for ESTCube-2

Abstract:

The main objective of this thesis is to continue the development and tests of the Sun sensors. They will be used in space to determine the position of a 3-unit CubeSat called the ESTCube-2, which is developed in the University of Tartu, Tartu Observatory.

During this thesis, software for Sun sensor operation and control was created. The end product has a sampling speed of 64 Hz and a timestamp accuracy of at least 1 µs. These Sun sensors will increase the pointing accuracy and stability of the ESTCube-2 Attitude and Orbit Control System (AOCS), allowing the satellite's two Earth Observation cameras to take high-quality pictures of Earth's vegetation as well as work towards meeting the primary mission goal of testing the Electrostatic Plasma Brake (EPB).

First, the ESTCube-2 AOCS architecture will be introduced along with an explanation of why the Sun sensors are an essential part of it. Second, prior work on this topic will be outlined to indicate the basis and starting position of this thesis. Then the explanation of the Sun sensor requirements along with the goals set for the thesis will be introduced. Also, an overview is given of the new software design that was developed in the course of this thesis is presented. Finally, the thesis focuses on hardware and software testing that was completed throughout different development phases, as well as the analysis of results.

Keywords:
Sun sensor software, CubeSat, AOCS, ESTCube-2

CERCS: P170: Computer science, numerical analysis, systems, control

ESTCube-2 päikesesensori tarkvara arendus

Lühikokkuvõte:

Bakalaureusetöö eesmärk on edasi arendada ja testida päikesesensoreid, mida hakkab kosmoses asendi määramiseks kasutama Tartu Ülikooli Tartu Observatooriumis loodud 3-ühikuline kuupsatelliit ESTCube-2.

Päikesesensori toimimise ja kontrollimise jaoks loodi tarkvara. Valminud toote mõõtmistihedus on 64 Hz ja ajatempli täpsus on vähemalt 1 µs. Päikesesensorid aitavad parandada ESTCube-2 satelliidi asendi- ja orbiidikontrolli alamsüsteemi (AOCS) suunamise täpsust ja stabiilsust. See aitab satelliidi kahel Maa jälgimis kaameral teha kõrge kvaliteediga pilte Maa vegetatsioonist ja võimaldab testida elektrilist plasmapidurit (EPB).

Kõigepealt tutvustatakse ESTCube-2 AOCS arhitektuuri koos seletusega, miks päikesesensor on oluline osa sellest. Teiseks räägitakse, mis oli tehtud ja kus oli selle bakalaureusetöö alguspunkt. Tuuakse välja, mis nõudeid ja eesmärke selles töös tahetakse täita. Lisaks antakse ülevaade tarkvara disainist, mida arendati selle töö jooksul. Viimaks keskendutakse riistvara ja tarkvara testimisele, mida sooritati kogu töö vältel, koos selle analüüsi ja tulemustega.

Võtmesõnad:
Päikese sensori tarkvara, CubeSat, AOCS, ESTCube-2

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

Introduction

Spacecraft play an essential part in today's world, fulfilling a myriad of different assignments. However, building and launching a conventional satellite is expensive, with the costs reaching hundreds of millions of dollars [1]. In 1999, a California Polytechnic State University project set out to mitigate this problem by standardising picosatellite[1] design to reduce their development and launch costs. They created the CubeSat Design Specification [2] that defined a class of picosatellites, which are composed of multiple 10 cm x 10 cm x 11.35 cm cubes, called CubeSat units with each unit's mass up to 2 kg.

Most satellite missions can only achieve their scientific and other objectives if the satellite can accurately determine and control its orientation (or 'attitude') in space or change their orbit. These problems fall into the jurisdiction of the Attitude and Orbit Control System (AOCS). A properly functioning AOCS can help the spacecraft gather energy more efficiently from the Sun, ensure a better radio communication link and allows for better pointing accuracy, making it possible to take high-quality pictures of specific places on Earth or in deep space. However, the CubeSat standard imposes strict restrictions when developing the AOCS - all the satellite hardware as well as its sensors and actuators have to fit into a tiny confined space. Additionally, there are often heavy constraints on available power and computational resources, meaning each sensor has to be small, low power, lightweight and still be accurate enough to meet the mission requirements.

At present, many universities and schools have their personal space program thanks to the CubeSat standard, which has lowered the barrier of entry into space technology [3]. One such university is the University of Tartu, which in partnership with the Estonian Student Satellite Foundation is developing ESTCube-2, a 3-unit CubeSat intended for a scientific mission in Low Earth Orbit (LEO) [4].The project is mainly composed of volunteer university and high school students with the goal of providing them with hands-on education, helping them become the next generation of space scientists, engineers, and entrepreneurs [5].

ESTCube-2's main mission, similar to ESTCube-1, is to test the plasma brake. P. Janhunen has described the plasma brake in his article [6] as a promising new technology that could help to reduce the space debris problem as well as open up possibilities for propellantless exploration of our Solar System and beyond in E-sail form. In ESTCube-2, the AOCS is especially important during the mission because it will enable the plasma

[1] Picosatellite is a satellite which mass is from 0.1-1 kilogram (https://www.defenseindustrydaily.com/Small-Is-Beautiful-US-Military-Explores-Use-of-Microsatellites-06720/)

brake to unwind its microtether by keeping it under tension using the centripetal force that will be created by spinning the satellite. For this task to succeed, the flawless cooperation of many different systems, sensors and actuators is required.

While an AOCS can incorporate many different sensors such as gyroscopes, star trackers and magnetometers, this thesis will focus on the Sun sensors. ESTCube-2 uses six custom-built Sun sensors, one for each side. Each sensor consists of two linear CMOS image sensors and a MCU dedicated for it. The main goal of this thesis is to get the ESTCube-2 Sun sensors as close to a flight-ready form as possible.

The thesis is divided into three main chapters. Chapter 1 provides a short overview of how the ESTCube-2's AOCS works, how different missions have used Sun sensors, and how the Sun sensor works more in-depth. Chapter 2 introduces the work that has been done so far with the Sun sensors, the ESTCube-2 Sun sensor's requirements with justifications for each of them, detailing the high-level software architecture, describing the final hardware configuration, and software implementation. Chapter 3 describes how the sensor design was validated along with conclusions where future improvements are discussed.

1 Background

The Attitude and Orbit Control System (AOCS) is an essential subsystem for spacecrafts as it's responsible for determining and controlling the satellite's orientation and orbit. A properly functioning AOCS allows the spacecraft operators to manoeuvre their craft for detumbling, orbital adjustments and pointing. All of this is accomplished by combining control system software, onboard sensors and actuators. The actuators move the spacecraft, the sensors give readings, indicating a change in direction and velocity, and the attitude determination control systems logic ties it all together.

It is important to note that some spacecrafts have an Attitude Determination and Control System (ADCS) similar to an AOCS. The main difference between these systems is that the ADCS cannot determine nor control the satellite's linear momentum, which affects the orbit of the satellite.

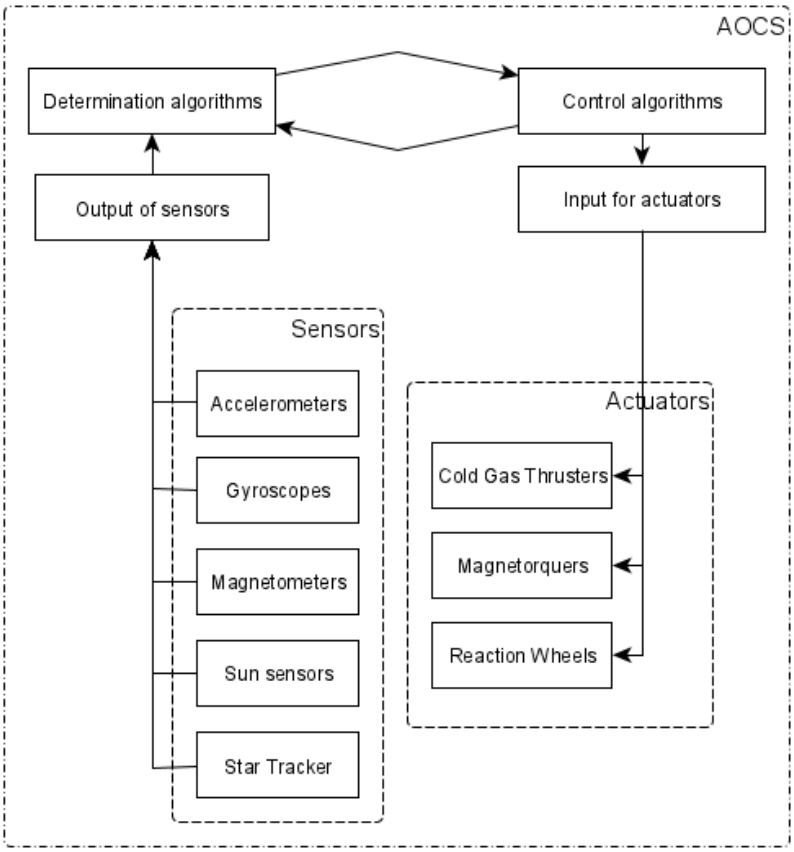

Figure 1. Simplified overview of the ESTCube-2 AOCS.

The ESTCube-2 AOCS, seen in Figure 1, broadly consists of sensors, control algorithms and actuators. A new satellite state is calculated using a Kalman filter by applying a Two-Line Element, sensor inputs and previously saved spacecraft positions. If mission control has set the desired orientation, the difference between the current orientation and the goal position can be calculated. The difference is then given to one of the control algorithms onboard, so it could use the satellite's actuators to achieve the desired position.

The list below describes each sensor in more detail that was used in the ESTCube-2 AOCS. This list of AOCS inputs is definitely not exhaustive as there are many other possible inputs like Earth sensors and GPS.

- **Gyroscopes** - sense rotation around a specific axis. They are most commonly used for pointing or spinning a spacecraft.
- **Magnetometers** - measure magnetic field strength and direction. They are used to detect a celestial body's, e.g. Earth's or the Sun's magnetic field. A spacecraft's approximate attitude and orientation can be calculated using magnetometers' measurements. Useful only around Earth and other plants with a magnetic field.
- **Accelerometers** - measure the change in speed, i.e. acceleration and its direction. They can be used to detect how a spacecraft's speed changes over time. For example, after the usage of means of propulsion, deployment of antennas, deployable panels or other sudden movement.
- **Star tracker** - takes pictures of space to identify star constellations using known star maps. The spacecraft can calculate its relative orientaton by referencing the taken photos and star maps located in onboard memory. The major value of this sensor is that it can be used for attitude and orbit determination in deep space, where magnetometers cannot be used, e.g. The NASA James Webb Space Telescope, which uses multiple star trackers to determine its position [7].
- **Sun sensors** - are one of the most common sensors on satellite missions, used for determining a spacecraft's orientation relative to the Sun. They do so by just measuring the intensity of the Sun from all the spacecraft sides or, in more advanced use cases, calculating the exact vector between the Sun and the spacecraft.

To save space, only a star tracker could be used on a satellite because AOCS could predict the satellite position from its output alone. Meaning, in theory, it is possible to control a satellite when the star tracker is the only sensor onboard. Yet it would not be wise to use only a single sensor to save room and weight because stars are not always visible in space. When a satellite is facing toward the Sun, its irradiance saturates the

star tracker photosensor, making it ineffective. That's why redundancy is needed; the other sensors can take over when one sensor stops functioning. As an additional benefit, multiple inputs from different sensors increase precision when they provide different readings to an AOCS determination algorithm.

1.1 Sun sensor

The Sun's direction is a reasonable external point to reference for a typical satellite in the Solar System, moreover as it's ever-present in Low Earth Orbit. The Sun's direction can be determined with a Sun sensor. There are different types of Sun sensors with varying working principles. The most common types have been thoroughly described in a Satsearch blogpost by Hywel Curtis [8].

The article divides Sun sensors into three broad groups:

- Coarse analogue Sun sensor
- Fine analogue Sun sensor
- Digital Sun sensor

Coarse analogue Sun sensors use the change of the Sun's intensity to calculate the Sun's angle towards the spacecraft. They use photodiodes or solar panels that supply different amounts of output current, which can be correlated with the Sun's angle. To find the Sun's angle in respect to the spacecraft, multiple diodes or solar panels are needed.

The good thing about coarse analogue Sun sensors is that they have a wide FoV, and when the solar panels are used for determining the Sun's direction, extra sensors are not required. The downside of coarse sensors is that they are not very accurate, and they are greatly affected by the Earth's albedo, potentially producing confusing results.

Fine analogue sun sensors can also use photodiodes or position-sensitive devices and, in addition, a mask to create a sunspot. The Sun vector is calculated based on the sunspot location in the light-sensitive area. Compared to the coarse sensor, fine sensors are more precise but with smaller FoV.

These sensors have accuracy in the range of 1°-3° or even below 0.5° if fine calibration tables are used. But albedo can add to inaccuracy up to 10° to 15°.

Digital Sun sensors are used less than analogue sensors. They can be just analogue sensors with a digital interface. Another possibility is that they use linear arrays and a

mask with a double-slit or a two-dimensional photosensor and a mask with a hole. The major upside for these sensors is that they can reduce the impact of indirect light, further refining the measurements of the Sun sensor.

1.2 Previous missions

There have been multiple CubeSats missions that have used Sun sensors. The following subsections give a short overview of a few different University CubeSat missions and their use of Sun sensors to determine the spacecraft's orientation towards the Sun.

1.2.1 ESTCube-1

ESTCube-1 was launched into Low Earth Orbit (LEO) in 2013 with one of its goals to test a novel ADCS [9]. For this, the CubeSat used, among other sensors, a 2-axis custom-built Sun sensors that consist of two analogue one-dimensional photosensitive detectors, and one Sun sensor on each side of the satellite, according to an article [10] about ESTCube-1 ADCS. It also mentioned that the Sun sensors were built upon their own in-house electronics board, as seen in Figure 2.

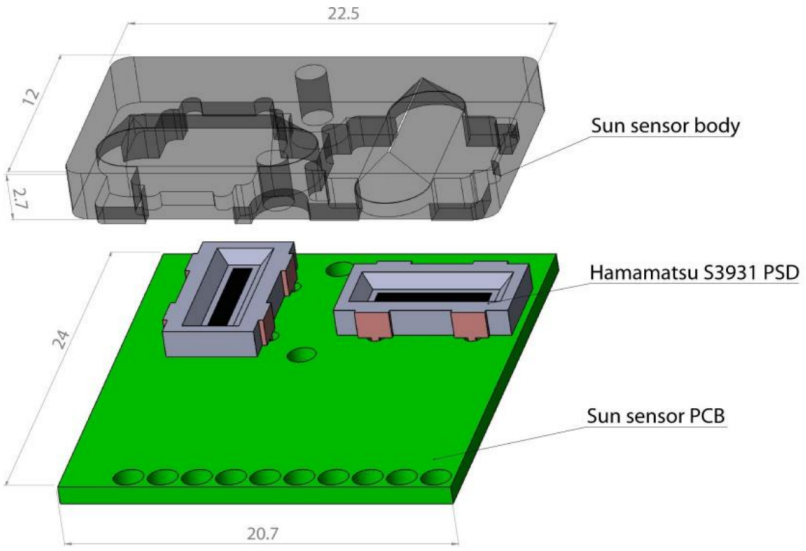

Figure 2. The custom-built Sun sensor hardware layout for ESTCube-1 [11].

In the ADCS article was also said, the ESTCube-1 team managed to get the uncertainty of their Sun sensor prototype down to 2.5°. However, there were some problems with the design, and a sensor mask had to be developed to minimise internal reflections.

Sünter talked in his thesis [12] the in-orbit demonstration showed that the effective Field of View (FoV) was limited from ±45° to ±36.7° as the photosensors provided weaker signals than expected. Furthermore, he pointed out that the final sensor measurement uncertainty was unknown.

In conclusion, ESTCube-1 Sun sensors were accurate enough and did their job. The sensors had a small footprint, so they managed to fit it all into a 1 unit satellite. Unfortunately, because of the limited FoV of 36.7°, the Sun sensors couldn't cover all sides of the satellite, creating blind spots. However, there is no info indicating that this caused any problems for the ADCS.

1.2.2 CubeSail

The article "CubeSail: A low-cost CubeSat based solar sail demonstration mission" [13] describes a 3U CubeSat called the CubeSail, developed by the University of Surrey in 2011. The primary goal of their mission was to demonstrate that a 5m x 5m membrane solar sail could be used for solar propulsion and deorbiting satellites. Their secondary objectives were to demonstrate a 3-axis active ADCS and self-deorbiting.

The ADCS used a Sun and nadir sensor module, a magnetometer, and a gyroscope. The Sun and nadir sensor, seen in Figure 3, consisted of two CMOS image sensors. The Sun sensor camera was pointed along the y-axis and the nadir sensor along the z-axis. In addition, there was a neutral density filter on the Sun sensor optics to reduce Sun intensity evenly across a portion of the specific light spectrum. The image sensors' data was stored in SRAM, and the microprocessor searched for the centre of the illuminated discs. The article says that the location of a light vector was calculated as an azimuth/elevation pair.

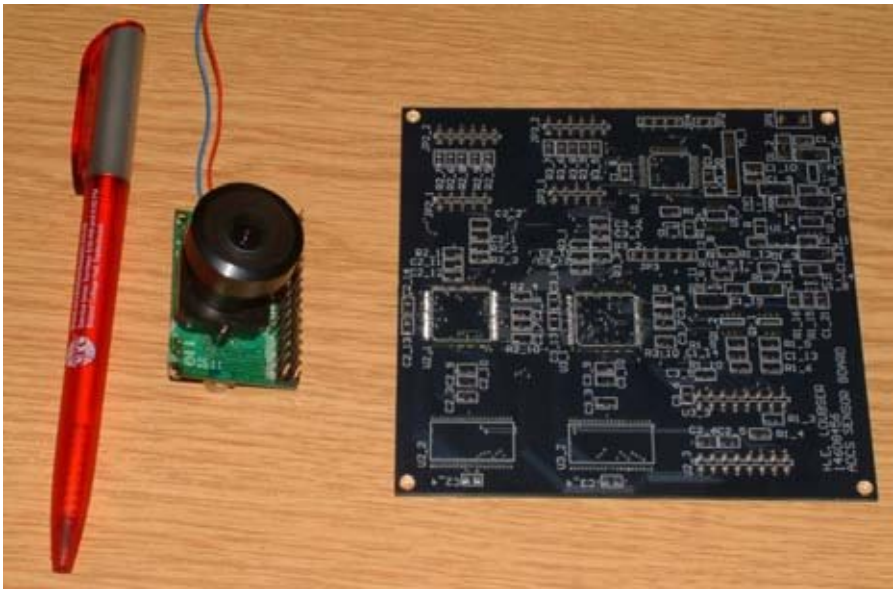

Figure 3. One camera module and Sun and nadir sensor PCB [14].

As CubeSail used only one Sun sensor, it couldn't use the sensor while the satellite was tumbling or spinning, but for their mission, it was suitable. Secondly, the entire sensor module took considerable room in a space-constrained environment. Finally, because the design used a CMOS camera, it needed sufficient computing power for image processing, which smaller and cheaper MCUs are not capable of.

1.2.3 Aalto-1

Aalto-1 was a 3-unit CubeSat with an Electrostatic Plasma Brake (EPB) as its primary payload [15]. Because of that, the ADCS was the most critical subsystem for mission success as it had to ensure the required pointing and spin up. The CubeSat used gyroscopes, magnetometers, and a star tracker for attitude determination. It is also pointed out that they used custom-built Sun sensors, as seen in Figure 4.

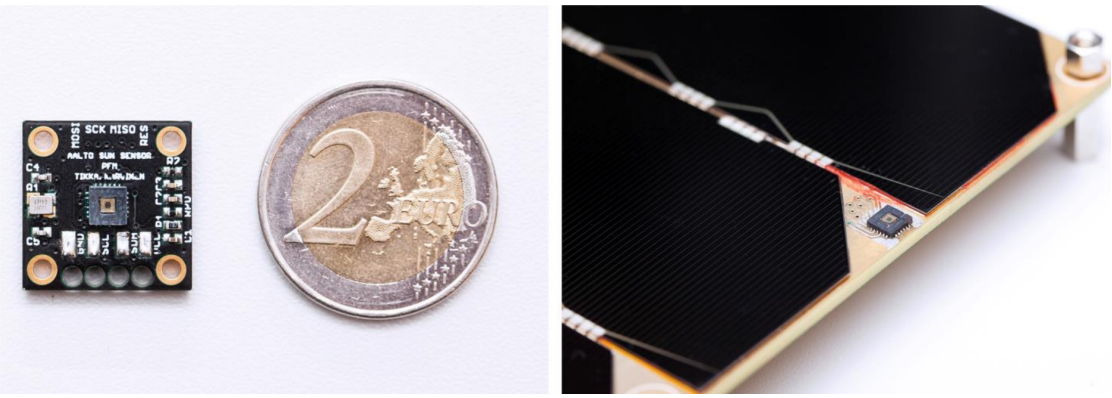

Figure 4. Sun sensor sub-module and integration with solar panels [16].

Aalto-1 Sun sensors were quite small and were placed on every side of the satellite to cover the entire area around the spacecraft, which is important because the satellite needed to spin up to deploy the plasma brake tether. Finally, as these digital sensors already gave out Sun vector with a 3° accuracy, they did not require a lot of computing power, making the Sun Sensor usable for less powerful MCUs.

The lessons learned article about the mission [17] mentions that Aalto-1 had some inoperative Sun sensors after the launch. However, they did not pose a detrimental problem for the ADCS because some algorithms could still work. While there were issues with the ADCS during the mission, like the EPB tether could not be deployed, they were not caused by the Sun sensors.

1.3 Lessons learned

As the ESTCube-2 mission is similar to those mentioned above, some of these experiences can be transferred.

First, it is important to have a small sensor footprint as the ESTCube-2 team is striving for a small, highly integrated module [18]. That means that the CubeSail's CMOS Sun sensor would be too big to fit onto every side to cover all the spacecraft sides. Furthermore, CubeSail's sensors had a dedicated MCU for calculating the Sun vector while the other missions did not. For others, calculating the Sun vector was an additional task for some other MCU on board. As ESTCube-2 uses two linear image sensors that require more information to be processed than coarse or even fine analogue Sun sensors, so it would be good to have a dedicated MCU. Then the subsystem main MCUs can

have more spare computing power for more complex algorithms and general subsystem management.

Second, during the ESTCube-1 mission, there were some problems with the Sun sensor's internal reflections[11]. As ESTCube-2 is going to use a similarly designed mask, it would be wise to take possible internal reflections into account when designing the mask and writing the Sun vector calculation algorithm.

Third, it is also vital that the Sun sensors have a wide enough field of view to cover all the sides of the spacecraft without any blind spots, aspect the ESTCube-1 mission failed in. Additionally, the system should be able to recover if some Sun sensors fail to work. Having several Sun sensors provides some form of redundancy, although with reduced accuracy and blind spots.

Finally, the common denominator for all missions was that testing is essential for reducing the risk of a sensor failure. The outcomes of each of the mentioned missions would have been better if the teams had had more time to test their Sun sensors.

1.4 ESTCube-2 Sun sensors

ESTCube-2 uses digital Sun sensors, and their simplified hardware overview can be seen in Figure 5.

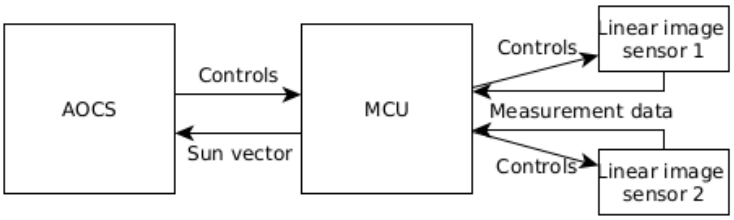

Figure 5. High-level hardware overview of ESTCube-2 digital Sun sensors.

The Sun sensors consist of two linear Hamatsu s9226 CMOS image sensors 1 by 1024 pixels, [19] and a MCU tasked with collecting measurement data from both image sensors and then calculating the Sun's unit vector from the measured data. After calculating the Sun vector, the MCU adds a timestamp to indicate measurement time. The measurement is then delivered to the AOCS at request.

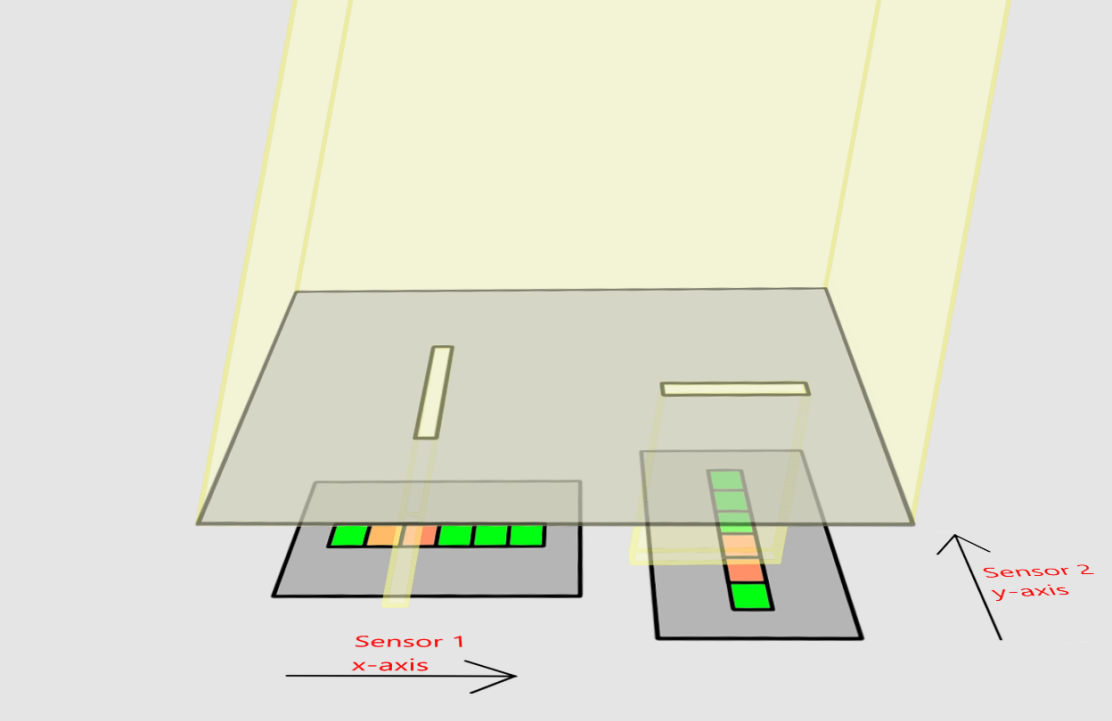

Figure 6. Two perpendicularly positioned linear image sensors make up the core of the Sun sensor.

Figure 6 shows the main logic behind the Sun sensor's. Sensor 1 measures the light in the local x-axis and sensor 2 in the local y-axis. Both image sensors are covered with black anodised masks that have slits perpendicular to the pixel array, so the light shines only on a very narrow range of pixels. The anodised masks try to minimise the potential internal reflections inside the sensor. Furthermore, the image sensors' protective classes are covered with a thin layer of titanium to reduce the Sun's intensity, therefore reducing pixel saturation in the direct sunlight of outer space.

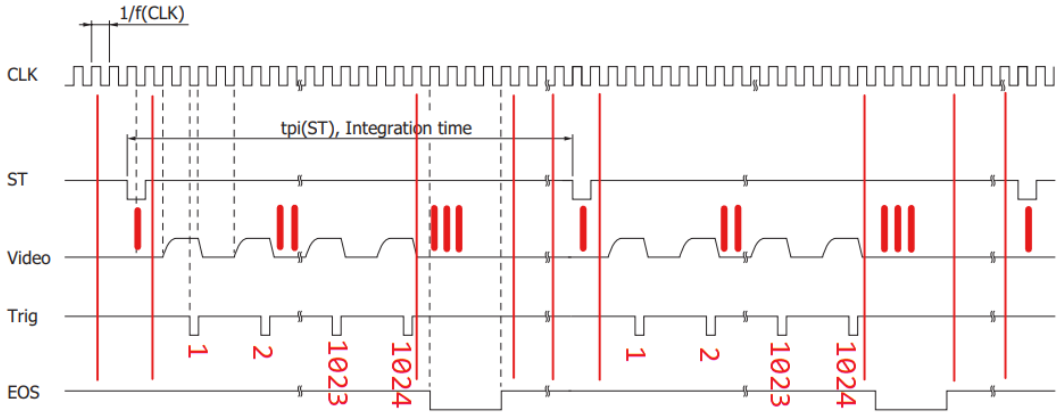

Figure 7. Image sensor input and output timings and phases marked with red Roman numbers [19].

Image sensor input and output timings can be seen in Figure 7. To initiate a measurement, phase I, the image sensor needs a reference clock (CLK) and a start pulse (ST). In phase II, after the start pulse, the sensors output an analogue (Video) and a trig pulse for each pixel for a total of 1024 pulses. The pixels are read in the same directions as the axes shown in Figure 6. Each video pulse represents light intensity on a given pixel, and the trig pulse indicates that a new video pulse is ready for measurement. After all 1024 pulses have been output, phase III, the image sensor outputs an End of Scan (EOS) signal to indicate all the pixels have been output on the Video line.

Each video pulse is quantised by an ADC and stored in the MCU's internal memory. By analysing each measured pixel's light intensity, the direction of the Sun from the sensor can be calculated.

2 Design and Implementation

The following section gives an overview of the prior work on the Sun sensors on board ESTCube-2 since 2016. Secondly, the requirements necessary for the Sun sensor operation are listed After that, the created software is described in detail along with the used technologies as well as the the results of the current development.

2.1 Work done prior to the thesis

Over the years, several people have worked on the development of ESTCube-2 Sun sensors. The basis of this thesis had been set by Aleksander Parelo during his bachelor's thesis [20] with some initial hardware development, along with prototyping the initial software and preliminary testing. Building on top of his work, Hans Teras headed hardware development and in depth selective testing throughout 2018 and 2020 for his masters' thesis, defended in 2020 [21]. This phase of the development mainly selected for the filter materials used in the Sun sensors to avoid saturation during mission conditions.

In Parelo's thesis [20], the Sun Sensors were tested to a FoV of over 68° with a standard deviation of 0.65°. Parelo's thesis helped pinpoint the strength of the additional filter needed for the Sun sensor to limit regions of full saturation which educe pointing accuracy. Hans Teras described new changes in the Sun sensor hardware design in his thesis [21]: a new Sun sensor mask was designed and mass-manufactured to increase reading accuracy and a thin-film reflective Titanium coating of the sensor was developed to reduce reading saturation during periods of direct line of sight with the Sun. During Hans' thesis, preliminary intensity mapping was also performed in two axes to characterise the Sun sensors, finding the FoV to be roughly ±55°. Although during testing, some problems were discovered, which were noted down as potential improvement points for the future, such as characerisation and mapping of the intensity readings.

The hardware mostly remained the same through all those tests, with the Sun sensor software running on the corresponding subsystem master MCU, the MSP430F5969 [22].

Before the beginning of this thesis, the ESTCube technical team had major concerns that the Sun sensor task schedule was too computationally expensive for the MSP430 micro-controller to handle, however, this claim was never thoroughly directly verfied. Instead, since the rest of the satellite was already using STM32 family processors, the Sun sensor was given its own dedicated MCU that would communicate with the rest of the satellite through a the master MCU of each side panel. The driving factor behind the design change was that it would enable more intensive measurement cycles and gives

the master MCU more time to coordinate with other subsystems. The new hardware architecture is seen in Figure 8.

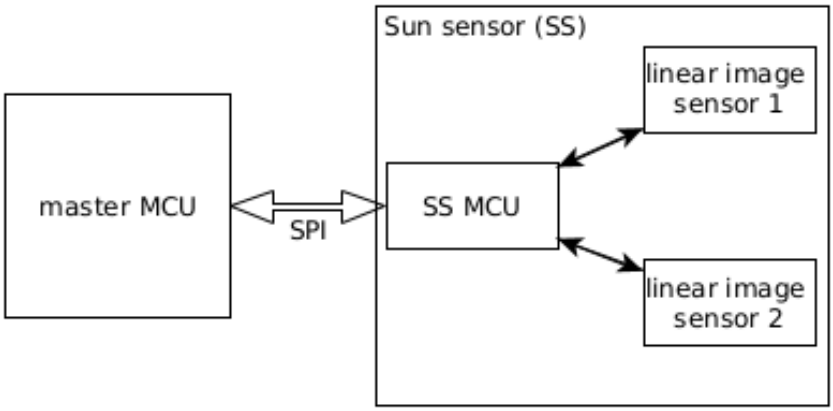

Figure 8. Updated Sun sensor hardware high-level logic.

The dedicated Sun sensor MCU is a STM32L422KBT [23] with two ADCs, each dedicated to sampling one of the image sensors. The image sensor signals are connected so that they would trigger a DMA transfer from the corresponding ADC registers into the MCU's memory after each pixel has been output from the image sensor. The team delegated as many tasks as possible to the underlying hardware peripherals. This way, the CPU inside the MCU can stay asleep until it is needed for the Sun vector calculation, thus saving power.

The biggest downside to the hardware redesign was that most of the software had to be rewritten because of the new MCU architecture. The MSP-based Sun sensor software analysis made by this thesis author can be seen in Appendix III. All the old prototypes used the MSP430 series micro-controllers; the newest design is STM32 based.

2.2 Requirements

The AOCS must determine the satellite's position and attitude, and for this, correct sensor input is required. This thesis aims to get the ESTCube-2's Sun sensors as close to a flight-ready state as possible to assist with attitude and orbit determination. To achieve this, the author had discussion with the ESTCube technical team and it was decided that the Sun sensors must meet the following requirements:

- Sun sensors must have an accuracy of at least 1°.
- Sun sensors must have FoV over 90°, so there would be no blind spots around the spacecraft.
- Sun sensors must have a sampling frequency over 10 Hz to maintain the relative desired accuracy while the satellite is spinning around 1 RPS at Plasma Brake deployment.
- Sun sensor output shall contain a unit vector with at least 4 ms accuracy timestamp so it would be usable in the AOCS algorithms.
- Sun sensors must provide raw measurement data, when desired, to help with calibrating the sensors once in orbit.
- Sun sensors must be able to communicate with the master MCU through SPI.
- Sun sensor MCU should sleep whenever possible to save power when the sensor is not needed as the image sensors of each Sun sensor take up a measureable amount of power.
- It shall be possible to update Sun sensor firmware, especially calibration matrices and other low lever filtering, while in orbit.

2.3 Software design

The final software consists of multiple layers, as shown in Figure 9. The Hardware Abstraction Layer (HAL) is responsible for managing the MCU so that developers would not need to burden themselves with the detailed knowledge of the MCU internals. Some peripherals' functionality was missing from the HAL to control the Sun sensors properly. The needed functionality was implemented by the author and the rest of the ESTCube team in parallel.

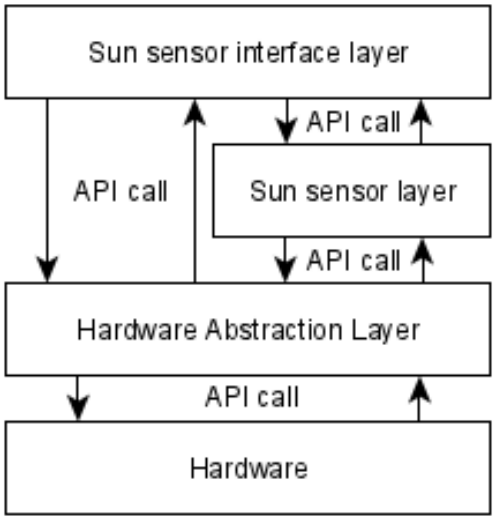

Figure 9. Sun sensor software layers.

The Sun sensor layer is responsible for managing the two image sensors, the measurement frequency, setting the Sun sensor operational mode and calculating and forwarding the measurement data. The Sun sensor interface layer is communicating with the master MCU and can control the work of the Sun sensor layer. There is no Real-Time Operating System (RTOS) in this project as it was decided that the Sun sensor is not complex enough, and a RTOS would take up a lot of memory in an already memory constrained device.

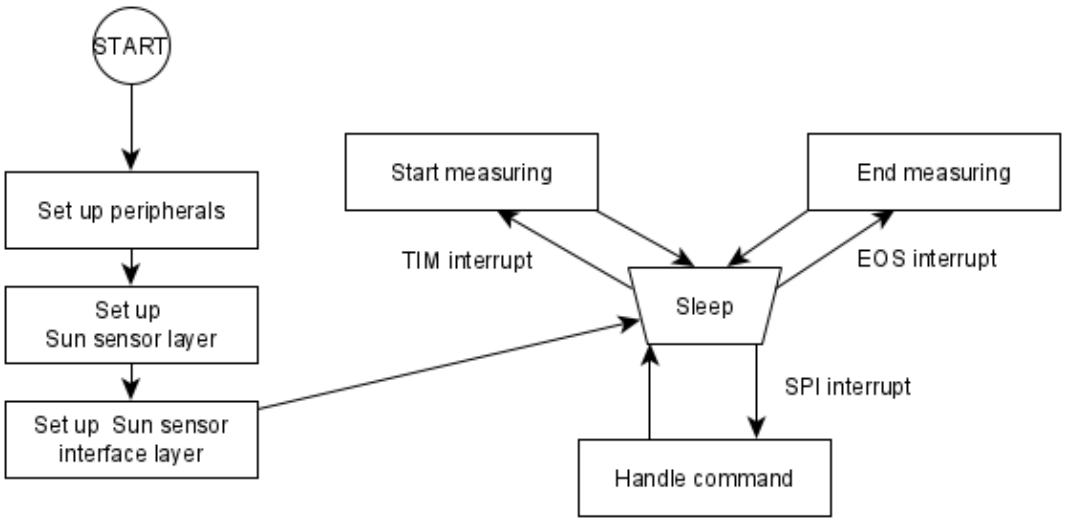

Figure 10. Sun sensor high-level software logic.

The final Sun sensor software logic can be seen in Figure 10. It consists of multiple functions that are executed by different interrupts. If no functions are being executed, then the CPU is sleeping, although some peripherals may be working in the background.

The necessary peripherals, the Sun sensor layer, and the Sun sensor interface layer are initialised at startup. Afterwards, the CPU goes to sleep and waits to respond to interrupts. Those interrupts will be handled by functions in the Sun sensor layer or in the Sun sensor interface layer.

2.3.1 Sun sensor layer

The Sun sensor layer can be seen in Figure 11. The algorithm starts setting up peripherals and the image sensors control layer during initialisation. For example, it is necessary to zero the image sensor internal counters if the sensors are stopped in the middle of measurement because then sensors fail to work as expected during the following measurement cycle. After setup, the CPU goes to sleep in order to save power.

It sleeps until a hardware timer, configured to create interrupts with the measurement frequency, wakes it up to signal the MCU to start a new measurement cycle. The MCU saves measurement start time and enables the image sensors' clock signal, the DMA, and

the ADCs. Then the MCU issues the start of a measurement signal. After this, the CPU goes back to sleep.

While the CPU sleeps, the ADCs measure the Video signal based on the EXTIs coming from the trig line and the DMA moves the raw measurement data from ADC registers into into MCU memory until the image sensors output an EOS signal, which wakes up the CPU again.

After wakeup, the MCU stops the ADCs, the DMA and the image sensor clock. It starts calculating the unit vector and saves it in the memory.

Finally, the CPU goes back to sleep and waits for the hardware timer to give it a new start signal, to begin with the measurement cycle again.

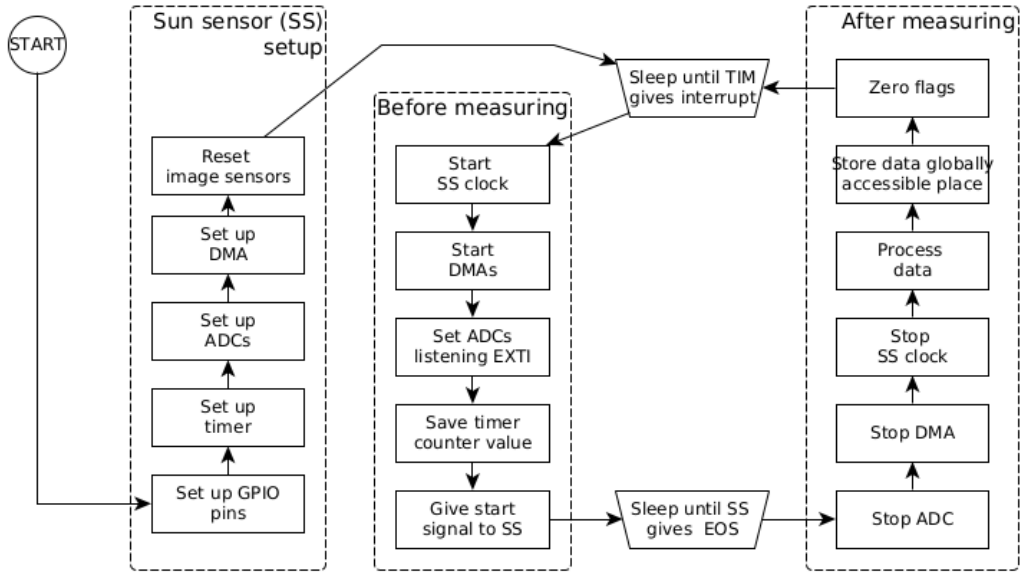

Figure 11. Sun sensor layer controlling two image sensors.

2.3.2 Sun sensor interface layer

The master MCU controls the Sun sensor interface and it controls the Sun sensor layer. Figure 12 shows the communication logic with the subsystem master. On the actual satellite over SPI communication will be used, but in the development phase, TTL serial[2]

[2]https://www.sparkfun.com/tutorials/215

was used due to its simplicity and wide support.

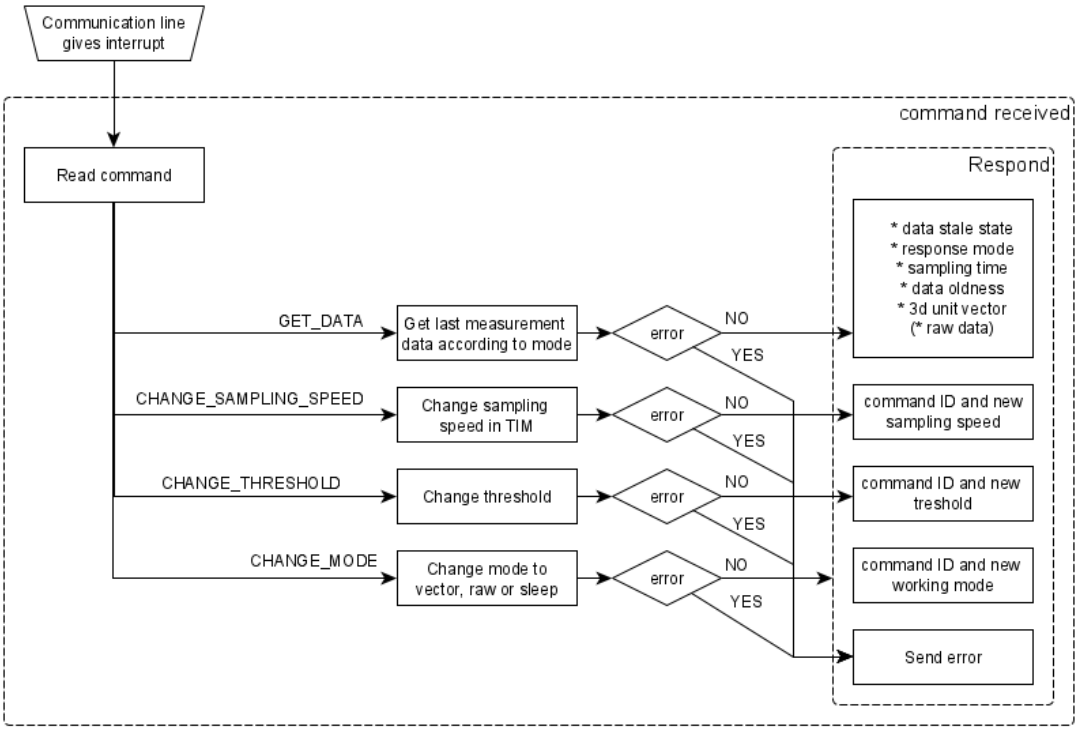

Figure 12. Sun sensor interface layer.

When the MCU gets a command, it finds out how to respond to it via the command's ID. It executes the necessary tasks and verifies that the tasks were successful. If no error occurs, the response is sent based on the command given. If an error takes place, a generic error packet is sent to the master as a notification.

The Sun sensor interface layer controls the Sun sensor layer through exposed functions which allows control over the entire measurement cycle from initialising to deinitialising. It is possible to change measurement modes, speed, configure the vector calculation algorithm, and to get the calculated vector and raw data.

If data is requested by the master MCU, then the latest measurement from the buffer will be read and the timestamp age calculated. For this are used the Sun sensor's MCU's timer combined with the internal oscillator, which can measure time with better accuracy than 1 µs based on the analysis done on the MCU datasheet [23]. Absolute timestamps are added to data by the master MCU because it can synchronise its time with the onboard

real-time clock. Therefore there is no need to use real-time clock inside the Sun sensor, as it would add complexity to the system and it can be done in the side panel master.

2.3.3 Sun vector calculation

Vector calculations are done under the Sun sensor layer, and the current vector calculation algorithm is simplistic. The raw data is saved as two images, each consisting of 1024 pixels, where each pixel is a 12-bit unsigned integer value.

The peak algorithm's code can be seen in Appendix IV. The first step is to find a peak from the saved image. For this, the pixel with the highest value is found. If the peak value is below a configurable threshold, then a zero vector is returned, as the Sun intensity is too low to correspond to the actual Sun. This helps to reduce false positives stemming from the Earth's albedo.

The Sun vector is then found from the average location from pixels above 90% peak intensity. This thresholding helps to reduce the influence of internal reflection on the final result as well as any major plateauing of the peak above the registerable range. The same method is applied for both saved images.

After peak distances from both sensor centres are known, a three-dimensional unit vector is calculated using mask and image sensors dimensions with trigonometry. The result is then saved in a globally accessible address.

2.3.4 Hardware changes

Based on the author's works, some changes were made in the electronics design. It was discovered that some of the image sensors' signals could be combined into one output pin, as seen in Figure 13. Those are the Start and the Gain pins. The clock pin was already tied together as a common signal for both sensors by previous Sun sensor developers.

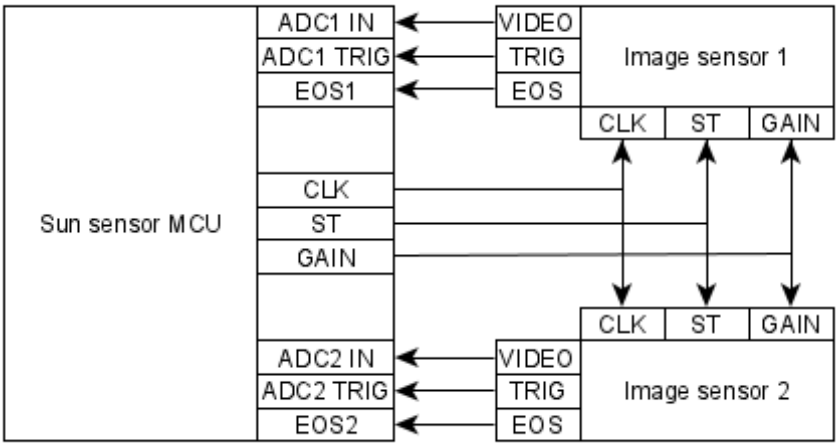

Figure 13. Image sensors connections with MCU.

2.4 Used technologies

Most of the development was done on the author's personal computer with the same toolchain as the rest of the ESTCube-2 team was using. This made it easier to integrate new pieces of code with the already existing codebase. More info about the ESTCube-2 codebase and software created during this thesis can be seen in Appendix II. Additional info about Hardware and Software can be seen in Appendix V.

Multiple steps needed to be done to get the code running on the desired MCU. First, as the bulk of the code was written in C, it needed to be compiled with GCC. Python scripts and Makefiles were used to make compiling multiple repositories easier. Second, after compilation, the binary was flashed on the MCU. For this, the development board was connected to the computer using a USB cable, and then the OpenOCD server was started. It acted as the software between the development hardware and the debugger that directly controlled the chip. With the OpenOCD server interfaced the GNU project debugger (GDB), what flashed the compiled binary on the MCU and enabled debugging it.

Python scripts were created to test the sensors' measurements. Scripts gathered, cleaned, processed and visualised data for the data analysis. The last parts were done with Jupyter Notebook with Anaconda, as it made it easy to manage the various Python modules.

A hardware platform was needed to run all this code. The Sun sensor had a prototype board where the two image sensors could easily be controlled. This board was initially attached to a MSP430 development board, and later was used the development board based on the STM32L4 MCU. The STM32L4 development board used an additional breakout board that the author of this thesis created.

A rotation bench and a strong light were used for the tests in the Optics Lab designated cleanroom in the Tartu observatory, as described in section 3. A Raspberry Pi 3 B+ was used to control the Sun sensor breakout board and gather data for outdoor Sun tracking tests.

2.5 Results

The current software architecture has some notable improvements compared to the older MSP430 version.

- As the analogue to digital converting and raw measurement storing is done by the hardware peripherals, the MCU's core now sleeps.
- Customisable measurement frequency helps to save power when there is no need for intensive measurements. Intensive measurements are only needed when the satellite is spinning to unwind the plasma brake tether.
- The Sun sensor has the ability to change its working modes. When there is no Sun, then the entire Sun sensor could be put into sleep. When there is a need for raw measurement data, then this could be achieved by the raw mode.
- The measurement speed can be raised almost to 200 Hz but it was decided that it is not necessary and the measurement speed upper hardcoded limit was set to 64 Hz.

3 Testing

There were multiple tests conducted during this thesis. Sun sensor firmware was tested, and the Sun sensor's physical characteristics were found. During the data analysis the vector calculation algorithm was updated, so it would find the highest and widest peak, as there were a lot of noise in the Sun tracking tests. All the cleanroom and the Sun tracking tests' results are based on the new algorithm simulated in the test scripts, but it have not been tested on the Sun sensor's hardware yet. Information about how to get access to the test scripts and stored measurement data is found in Appendix II.

3.1 Software and hardware testing

Every function in the Sun sensor layer code's has a simple test written for it. It was tested how setting modes, sampling speeds and thresholds affect the Sun sensor in a combination of different configurations. However, for verification, it was necessary to check the Sun Sensor behaviour with a logic analyser. All this helped to verify that the Sun sensor layer is functional.

In addition, a breakout board was created to test Sun sensors that mimicked the final hardware as much as possible. The created board can be seen in Appendix V. To test the Sun sensor and master MCU communication a Nucleo L4 development board was chosen to play the the role of the master MCU. The Nucleo board itself exchanged packets with the developers computer through USB and commanded the Sun Sensors through SPI. This setup helped to verify that the Sun sensor interface layer is functional and that the Sun sensor software as a whole should run on the final hardware.

3.2 Darkroom testing

The first test was conducted by the author of this thesis and Hans Teras in August 2021. It was conducted in the University of Tartu Tartu observatory's optics lab cleanrooms. The optical room was purposefully darkened with back floor, walls and ceiling, so there would not be incidental light or reflections. There was a high-powered source of light and a setup containing multiple rotating benches. More about the test setup can be seen in Appendix VI.

The Sun sensor collected raw data and could temporarily communicate with the computer, which controlled the sensor and testing bench. The test's main objective

was to find the current physical characteristics of the Sun sensor, e.g. it's measuring intensities and limits. Later it would be possible to fine-tune the Sun sensor algorithm based on the tests' data.

For the first two tests, one of the image sensors was covered, so only one image sensors' characteristics would be mapped at a time. Then the whole FoV with 10° steps was scanned from -60° to 60°. Later, it was discovered that there was a fault with the script controlling the bench's stepper motors. Some of the data points were shifted by 10 ° compared to the actual degrees defined on the bench , however this was later fixed in the data analysis script.

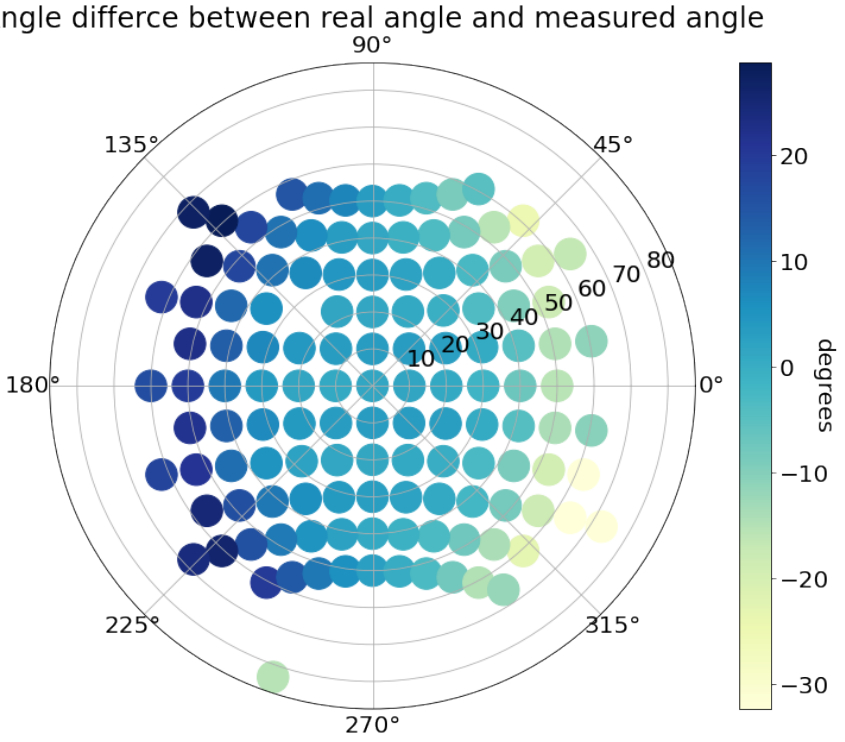

Figure 14. Sensor 1 measured vector error from the expected vector in different directions over FoV.

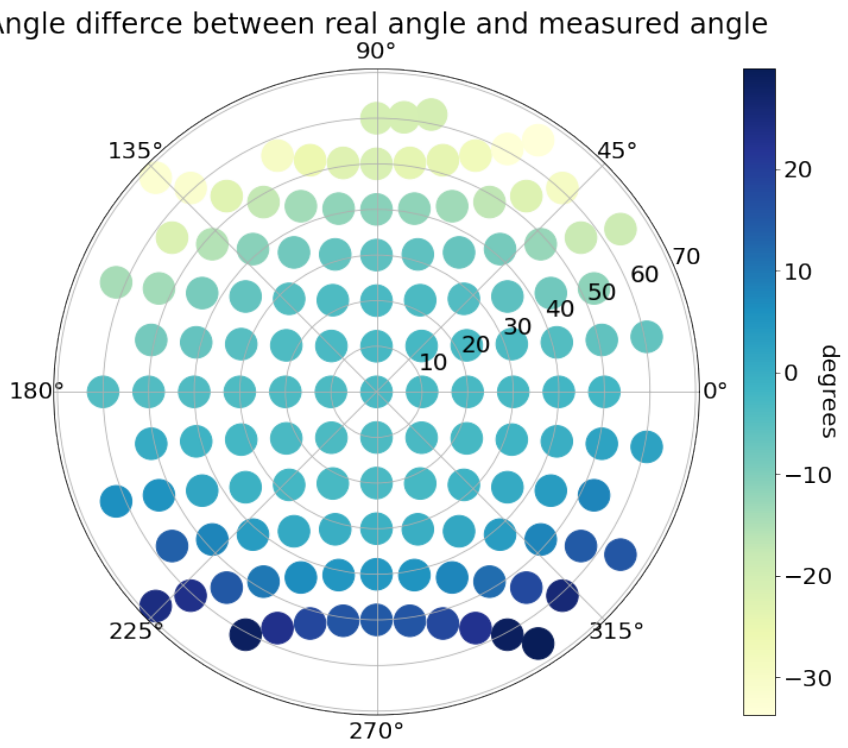

Figure 15. Sensor 2 measured vector error from the expected vector in different directions over FoV.

The results of the tests can be seen in Figures 14 and 15, where relatively high inaccuracies of incident light are shown. These characterizations can be therefore used to calibrate each of the 12 image sensors of the satellite on ground. One other possible reason for a large error margin in the Sun sensor measurements is some obscure error in the testing script and randomly lost images during the measuring. It is possible that the accuracy could be improved by turning some degrees measured vectors into certain direction, therefore eliminating testing setup error.

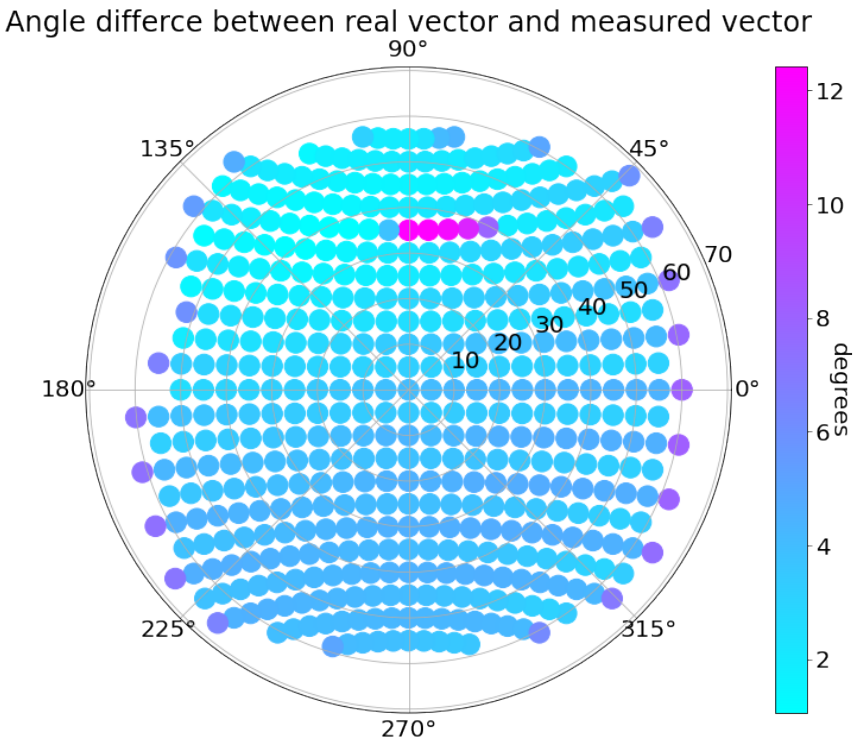

Figure 16. Whole area scan results.

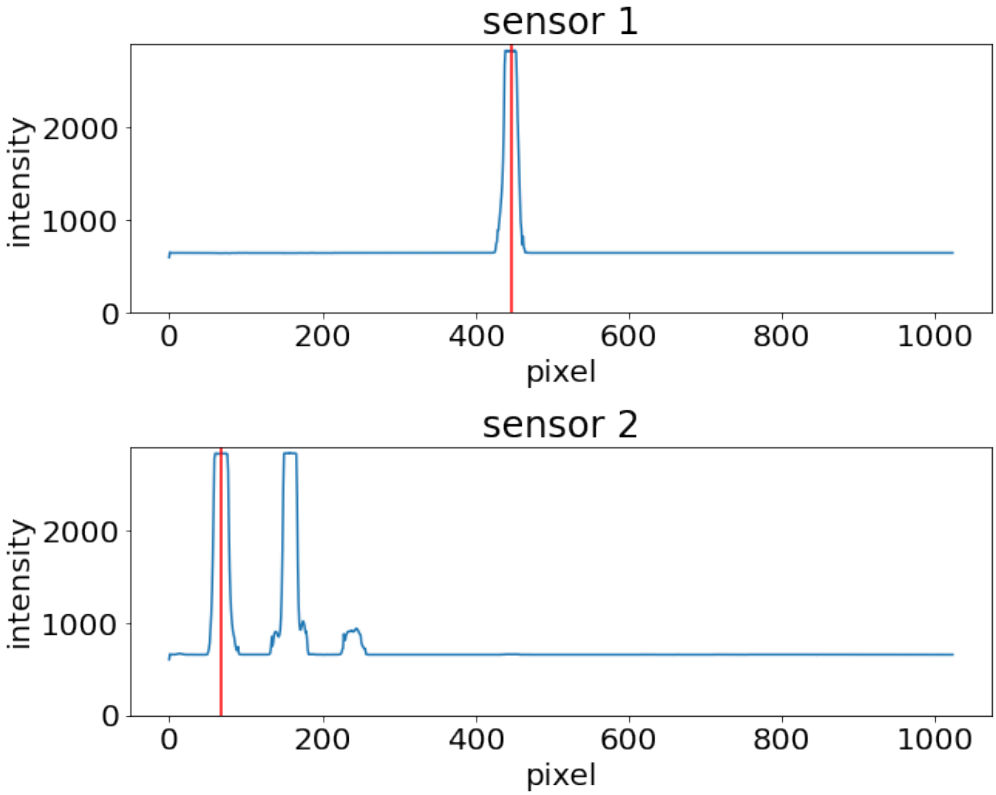

Figure 17. Pixel values of the image pair from the anomaly area.

The second test scanned the whole FoV with a 5° step. These results can be seen in Figure 16. This test had similar problems as the previous one. As can be seen from the test results - there is an anomaly. This is because one image sensor gets light from the other sensor through its mask slit, which caused the extra peak seen in Figure 17. This could be prevented by making more changes in the hardware to better separate the two image sensors.

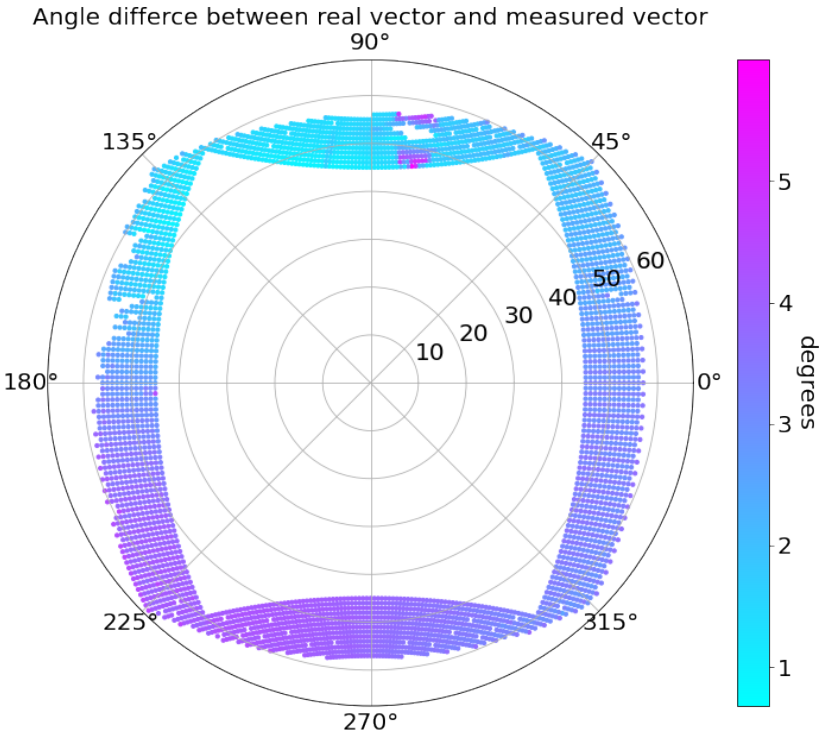

Figure 18. Edge scan results.

The third test scanned only the edge of the FoV, with a step of 1°, seen in Figure 18. This test also shows an anomaly. It was calculated that the FoV with the current algorithm is at least ±50°. But with further analysis, it is possible to make corrections and to improve the results.

In all the tests, there were problems with real vectors as some images were lost during the measurement. This is probably because some bits were lost during communication and therefore the image's pixel values were discarded, causing a shift in the real vector. This in return, causes a bigger error in predicted angle. However, the tests did show that the new hardware design worked more or less as expected.

3.3 Sun tracking testing

In April 2022 two Sun tracking tests were conducted . The test setup consisted of a Sun sensor breakout board and a Raspberry Pi 3B+, which played the role of the master. The main objective of the test was to determine if the improved Sun sensor firmware works and to test the calculation algorithm. A more detailed test description is provided in Appendix VII.

The ground truth for the Sun azimut and elevation was gotten from Torsten Hoffman SunCalc [3] website based on the test time and location. Then the Sun angles are compared with the angles measured by the Sun Sensor. The results of the two test can be seen in Figures 19 and 20.

Those results show that the Sun sensor can track the Sun with a accuracy of roughly 1.4° with a standard deviation below 1. In the future, it is possible to raise the accuracy with further corrections.

[3] https://www.suncalc.org/

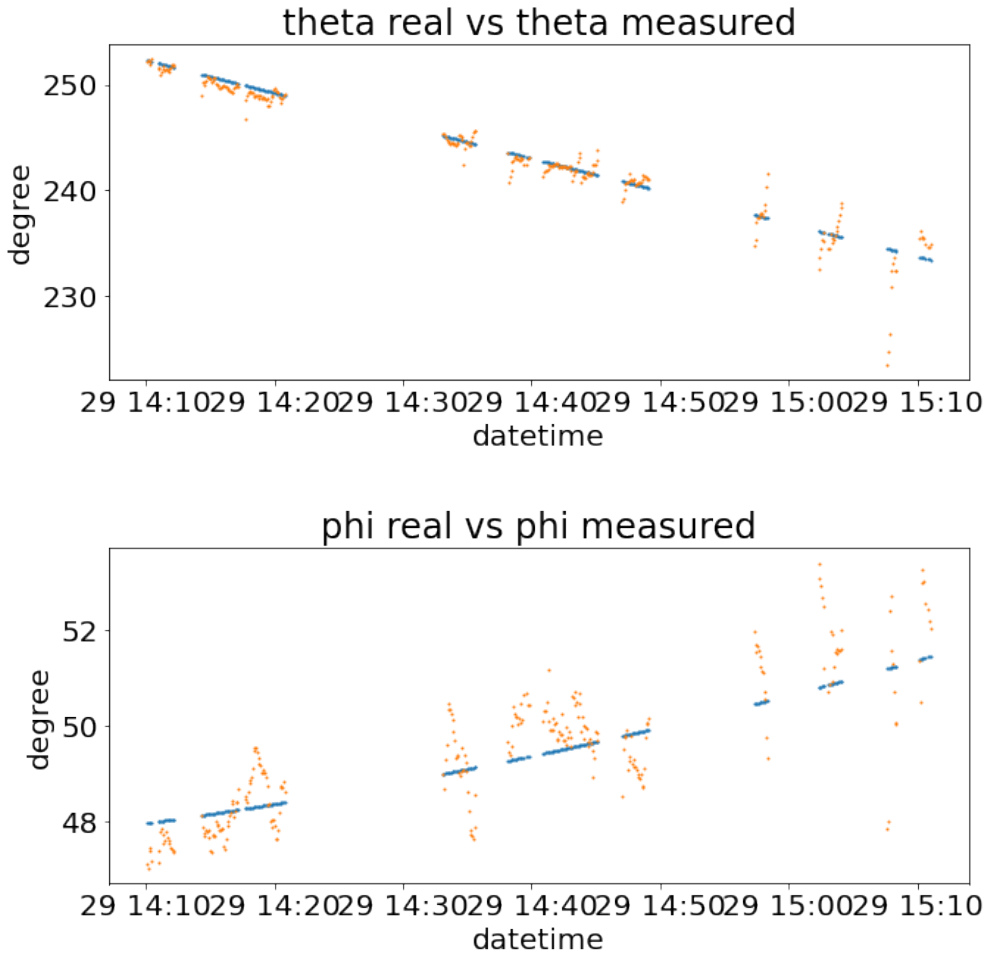

Figure 19. Sun tracking results on a tripod in spherical coordinates, where $\rho = 1$.

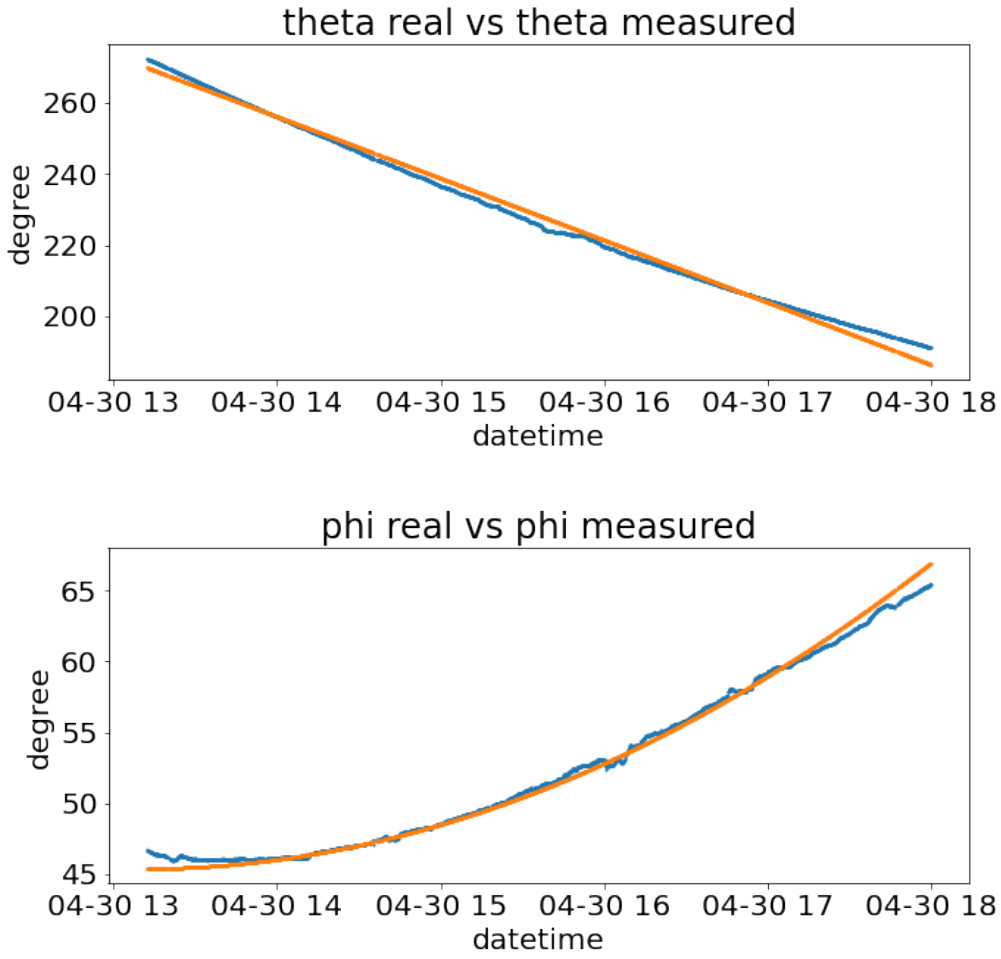

Figure 20. Sun tracking results on a window sill in spherical coordinates, where $\rho = 1$.

3.4 Results

The conducted tests confirmed that the sensors' software is functional and the sensors can measure faster than 10 Hz with a timestamp inaccuracy below 1 µs. This is critical for the functioning of the Sun sensors during the high-speed rotations of the Plasma brake deployment. The stationary Sun tracking tests concisely indicate, that the sensors need further embedded calibration related to accuracy and FoV improvements, which can be accomplished by enhancing the Sun vector algorithm with a signal correction table as planned for ESTCube-2, in addition to the temperature and other calibrations, which are part of Hans Teras' thesis[21]. Also, updating the Sun sensor's mask by separating the space between the two image sensors mechanically would potentially further increase measurement accuracy as false signals would be avoided.

Conclusion

During the thesis, the author tested an existing MSP430 based Sun sensor prototype in order to map its faults. A new STM32 based Sun sensor software prototype was developed with the accompanying firmware to measure and relay the Sun sensor data to the AOCS along with verifying that the new software is functional.

The final result has an accuracy of roughly 1.4°, which needs to be confirmed with final flight model testing of the ESTCube-2 satellite in the near future, prior to the delivery to the launch provider. Secondly, there is an area in the FoV, where accuracy is lower. This anomaly is caused by the light that shines on the one image sensor from the other sensor's mask's slit.

The FoV is approximately ±50°so there are no expected blind spots around the spacecraft. Furthermore, the sensor has a maximum sampling speed of 64 Hz which is over the desired initial speed of 10 Hz, with a timestamp inaccuracy below 1 µs, filling the requirements of the sensor by the AOCS.

The Sun sensor is controllable over a SPI line which makes it possible to send commands to get measurement data, change sampling speed, working mode and fine-tune the Sun vector algorithm's light intensity threshold. It is possible to get raw measurement data to calibrate sensors in-orbit with the debugging mode. The heavy use of sleep in the final firmware helps to save power when the sensor is not needed.

Most of the requirements were sufficiently met, however the Sun sensor's accuracy is not below 1°without extra emphasis on sensor output pre-calibration. The required possibility to update software in-orbit was not implemented as part of this thesis.

In the future, more work needs to be done on the Sun vector calculating algorithm to raise measurement accuracy in the expected ±1°range. A possibility to update Sun sensor firmware in-orbit would be very beneficial to have as designated by the ESTCube-2 team. On the hardware side Sun sensor masks need to be potentially corrected to to eliminate the discovered double peak anomaly, however, this effect is not as critical to the overall functionality of the sensor to warrant a high priority.

Acknowledgments

I have had a lot of help during my journey. I would like to thank the ESTCube team and especially my supervisors Kristo Allaje and Hans Teras, as most of the help with the thesis practical and writing side was offered by them. I would like also to thank Kaivi for her support.

References

[1] *The Cost of Building and Launching a Satellite*. GLOBALCOM. URL: https://globalcomsatphone.com/costs/ (visited on 12/09/2021).

[2] *CubeSat Design Specification*. Version REV 14, DRAFT. California Polytechnic State University, 2020. URL: https://static1.squarespace.com/static/5418c831e4b0fa4ecac1bacd/t/5f24997b6deea10cc52bb016/1596234122437/CDS+REV14+2020-07-31+DRAFT.pdf.

[3] Jamie Chin et al. *CubeSat 101: Basic Concepts and Processes for First-Time CubeSat Developers*. NASA CubeSat Launch Initiative, 2017. URL: https://www.nasa.gov/sites/default/files/atoms/files/nasa_csli_cubesat_101_508.pdf.

[4] ESTCube Team. *About ESTCube-2 mission*. Eesti Tudengisatelliidi Sihtasutus. 2022. URL: https://www.estcube.eu/blog/About (visited on 05/10/2022).

[5] ESTCube Team. *The ESTCube-2 mission*. 2022. URL: https://www.estcube.eu/blog/The-ESTCube--2-mission (visited on 05/10/2022).

[6] Pekka Janhunen. "Electrostatic Plasma Brake for Deorbiting a Satellite". In: *Journal of Propulsion and Power* 26.2 (2010), pp. 370–372. DOI: 10.2514/1.47537.

[7] *JWST User Documentation (JDox). JWST Attitude Control Subsystem*. Baltimore, MD: Space Telescope Science Institute. May 17, 2017. URL: https://jwst-docs.stsci.edu/jwst-observatory-hardware/jwst-spacecraft-bus/jwst-attitude-control-subsystem (visited on 05/10/2022).

[8] Hywel Curtis. *Sun sensors on the global marketplace for space*. satsearch. Mar. 16, 2022. URL: https://blog.satsearch.co/2020-02-12-sun-sensors-an-overview-of-systems-available-on-the-global-marketplace-for-space (visited on 03/20/2022).

[9] Andris Slavinskis et al. "ESTCube-1 in-orbit experience and lessons learned". In: *IEEE Aerospace and Electronic Systems Magazine* 30.8 (2015), pp. 12–22. DOI: 10.1109/MAES.2015.150034.

[10] Andris Slavinskis et al. "Attitude determination and control for centrifugal tether deployment on the ESTCube-1 nanosatellite". In: *Proceedings of the Estonian Academy of Sciences* 63.2S (2014), pp. 200–209. DOI: 10.3176/proc.2014.2S.05.

[11] Robert Valner. "Characterization of custom built Sun sensors for ESTCube-1". BA thesis. University of Tartu, 2013.

[12] Indrek Sünter. "Design and characterisation of subsystems and software for ESTCube-1 nanosatellite". PhD thesis. University of Tartu, 2019.

[13] Vaios Lappas et al. “CubeSail: A low cost CubeSat based solar sail demonstration mission”. In: *Advances in Space Research* 48.11 (2011). SOLAR SAILING: CONCEPTS, TECHNOLOGY, AND MISSIONS, pp. 1890–1901. ISSN: 0273-1177. DOI: https://doi.org/10.1016/j.asr.2011.05.033. URL: https://www.sciencedirect.com/science/article/pii/S0273117711003991.

[14] Vaios Lappas et al. “CubeSail: a low cost small cubesat mission for solar sailing and deorbiting LEO objects”. In: *Proceedings of the Second International Symposium on Solar Sailing (ISSS 2010), The New York City College of Technology of the City University of New York* (Jan. 2010).

[15] J. Praks et al. “Aalto-1, multi-payload CubeSat: Design, integration and launch”. In: *Acta Astronautica* 187 (2021), pp. 370–383. ISSN: 0094-5765. DOI: https://doi.org/10.1016/j.actaastro.2020.11.042. URL: https://www.sciencedirect.com/science/article/pii/S0094576520307189.

[16] Kimmo Karvinen, Tuomas Tikka, and Jaan Praks. “Using Hobby Prototyping Boards and Commercial-off-the-shelf (COTS) Components for Developing Low-cost, Fast-delivery Satellite Sub-systems”. English. In: *JOURNAL OF SMALL SATELLITES* 4.1 (2015), pp. 301–314. ISSN: 2327-4123.

[17] M. Rizwan Mughal et al. “Aalto-1, multi-payload CubeSat: In-orbit results and lessons learned”. In: *Acta Astronautica* 187 (2021), pp. 557–568. ISSN: 0094-5765. DOI: https://doi.org/10.1016/j.actaastro.2020.11.044. URL: https://www.sciencedirect.com/science/article/pii/S0094576520307190.

[18] Janis Dalbinš et al. “ESTCube-2: an integrated platform demonstation for future deep-space CubeSat missions”. In: Finnish Sattelite workshop & Remote Sensing Days 2021 (Aug. 23–24, 2021). Conference presentation abstract. Helsinki, Finland, 2021. URL: https://drive.google.com/file/d/19hTPqoKEHzWEV08TydieYqj406XT-yNw/view (visited on 05/08/2022).

[19] *CMOS linear image sensor. S9226-03*. Hamamatsu Photonics K.K. 2021. URL: https://www.hamamatsu.com/resources/pdf/ssd/s9226_series_kmpd1121e.pdf (visited on 10/06/2021).

[20] Aleksander Parelo. “Development of ESTCube-2 side panels”. BA thesis. University of Tartu, 2018.

[21] Hans Teras. “ESTCube-2 attitude and orbit control subsystem sensor and actuator calibration and evaluation”. MA thesis. University of Tartu, 2020.

[22] *MSP430FR596x, MSP430FR594x Mixed-Signal Microcontrollers*. Texas Instruments. 2018.

[23] *STM32L422xx. Ultra-low-power Arm® Cortex®-M4 32-bit MCU+FPU, 100DMIPS, 128KB Flash, 40KB SRAM, analog, AES*. STMicroelectronics. 2019. URL: https://www.st.com/en/microcontrollers-microprocessors/stm32l422kb.html (visited on 10/06/2021).

Appendices

I Acronyms

ADC Analog to Digital Converter. 17, 19, 23, 46, 47

ADCS Attitude Determination and Control System. 8, 11–14

AOCS Attitude and Orbit Control System. 2, 3, 6–10, 15, 20, 38, 47

CMOS Complementary Metal Oxide Semiconductor. 7, 12–15

CPU Central Processing Unit. 19, 22, 23

DMA Direct Memory Access. 19, 22, 23, 46, 47

EOS End of Scan. 17, 23, 47

EPB Electrostatic Plasma Brake. 2, 3, 13, 14

EXTI External Interrupt. 23

FoV Field of View. 10, 12, 18, 20, 29, 30, 32, 33, 37, 38, 58

GPS Global Positioning System. 9

HAL Hardware Abstraction Layer. 21

LEO Low Earth Orbit. 6, 10, 11

MCU Micro-controller Unit. 7, 13–15, 17–24, 26–28, 47

PCB Printed Circuit Board. 13

RPS Rotations Per Second. 20

RTOS Real-Time Operating System. 21

SPI Serial Peripheral Interface. 20, 23, 28, 38

SRAM Static Random Access Memory. 12

TLE Two-Line Element. 9

TTL Transistor-Transistor Logic. 23

UART Universal Asynchronous Receiver-Transmitter. 46

II Source code

Software for this thesis is stored in in the ESTCube Bitbucket server. Measurements, documentation and guides for the environment setup are stored in Confluence. To get access to the ESTCube Bitbucket and Confluence, contact the ESTCube team at estcube@estcube.eu.

III MSP code analysis

At the beginning of the thesis there was no practical documentation about the MSP-based Sun sensor as it was still in development. For that, the author conducted a code analysis on the MSP430 based prototype. The results can be seen in Figures 21, 22, and 23

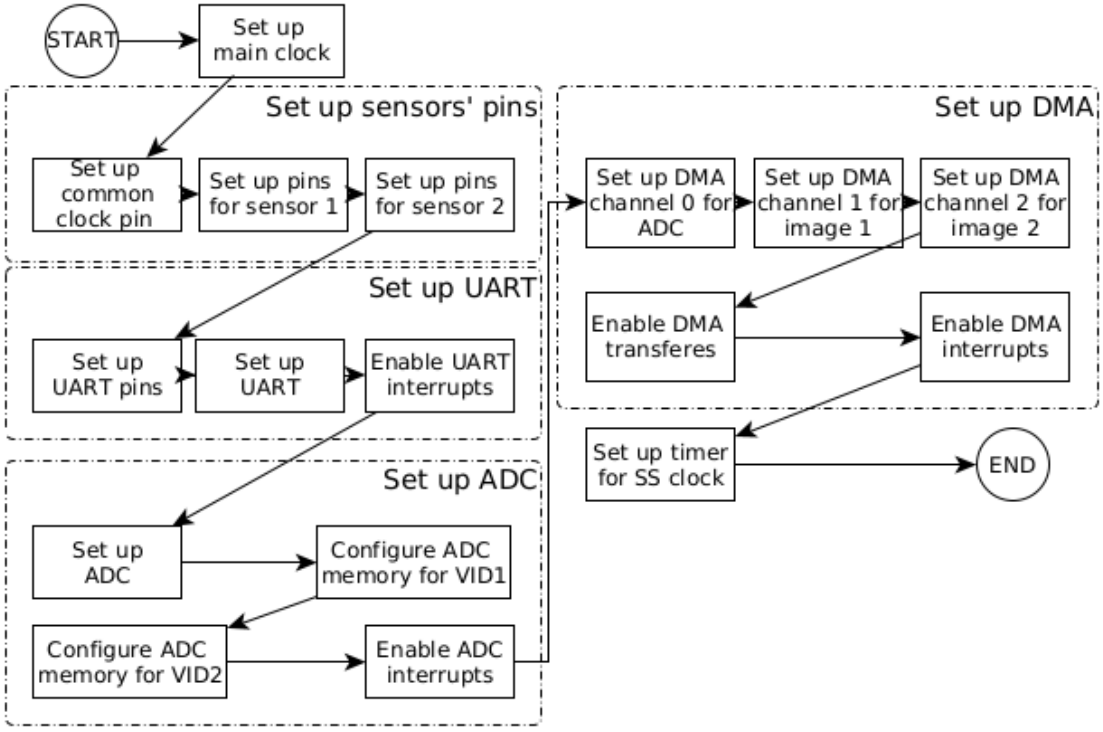

Figure 21. MSP-based Sun sensor setup.

In Figure 21, there can be seen that it is only setting up peripherals. It can be seen that UART is used for communication; one timer for the image sensors' input clock, one ADC and one DMA with three channels are used.

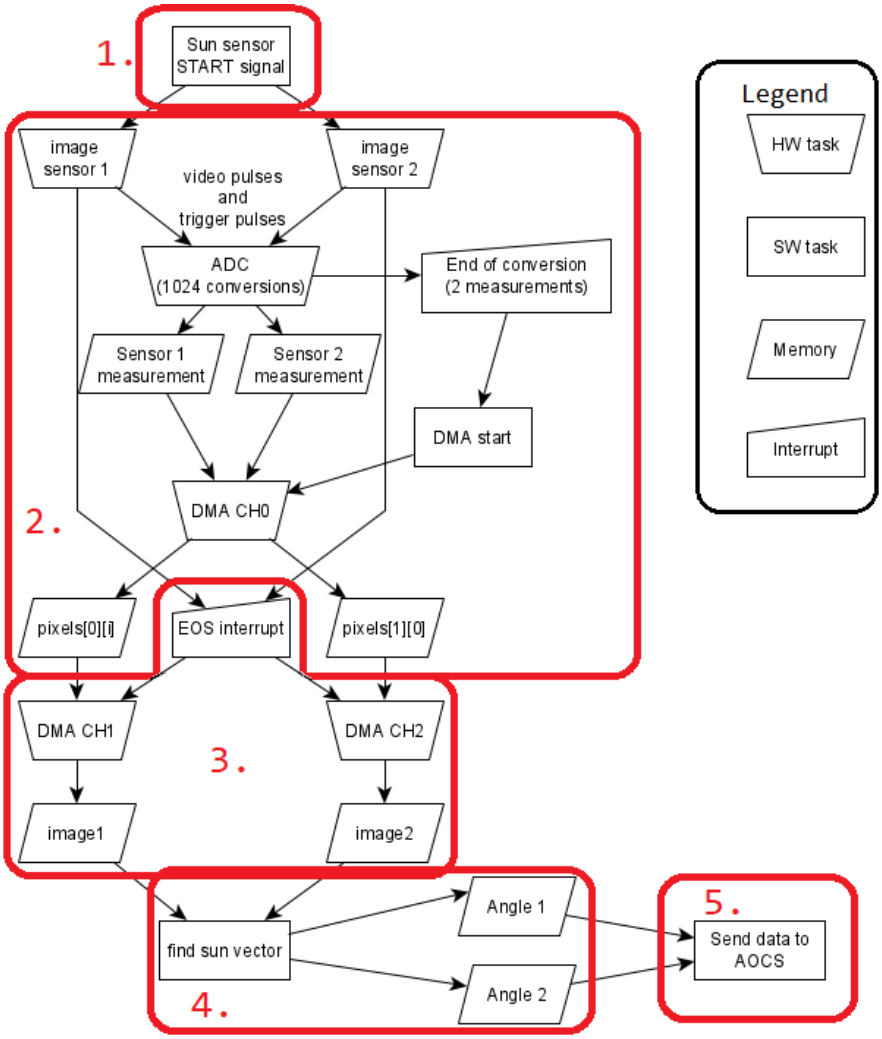

Figure 22. MSP-based Sun sensor data flow.

The Sun sensor's initial logic is the same as described in Section 1.4, but more deeply can be seen in Figure 22. In the 1. phase, the measurement is initiated, and then in the 2. phase, image sensors start sending video and trigger pulses that the one ADC handles. After each pulse pair of measurements, DMA is started to save measurements. The EOS pulse starts the 3. phase, after which measurements are copied to the new location in memory to avoid later memory corruption. In the 4. phase, the Sun vector is found and sent to AOCS in the 5. phase. From the logic, some tasks are handled by the hardware, like measuring pulses with ADC and moving data with DMA. This helped to reduce MCU's core usage.

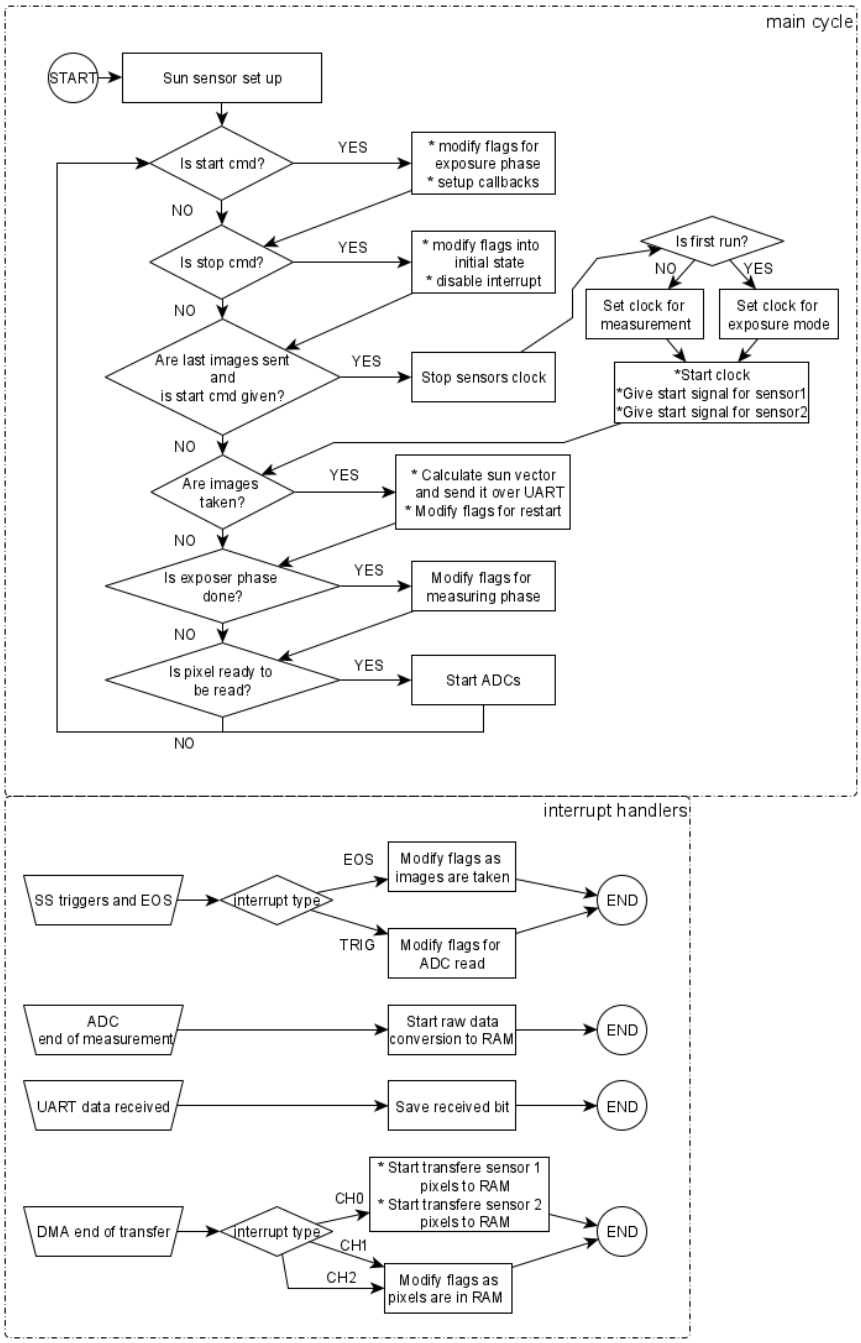

Figure 23. Sun sensor data flow.

Initial software logic can be seen in Figure 23. There we can see that the Sun

sensor tasks are handled in a closed while loop with different interrupt handlers. First, peripherals necessary for the Sun sensor are set up, and flags are initialised and valued for the first run. Then it will enter into the main loop and start checking in what phase the Sun sensor is. When some interrupt occurs, then in an interrupt handler, some task is done or flags are changed for a new phase.

IV Algorithm for finding a peak

```
#define SS_IMAGE_SIZE 1024
#define SS_SENSOR_LENGTH 7.9872
#define SS_PIXEL_LENGTH 0.0078
/**
 * Finds the distance from the centre of the sensor in millimeters.
 */
static bool find_peak(volatile uint16_t *image, float *peak) {
  uint16_t max_value = 0;
  uint16_t px_peak = 0;

  for (uint16_t px_idx = 0; px_idx < SS_IMAGE_SIZE; px_idx++) {
    uint16_t px_value = image[px_idx];
    if (px_value > max_value) {
      max_value = px_value;
      px_peak = px_idx;
    }
  }

  if (max_value < g_sun_threshold)
    return false;

  // Getting rid of the peak plateau randomness.
  uint16_t summed_max_peak = 0;
  uint16_t nr_of_pixels = 0;
  uint16_t max_value09 = max_value * 0.9;
  for (uint16_t px_idx = 0; px_idx < SS_IMAGE_SIZE; px_idx++) {
    uint16_t px_value = image[px_idx];
    if (px_value > max_value09) {
      summed_max_peak += px_idx;
      nr_of_pixels++;
    }
  }

  float average_peak = (float)summed_max_peak / (float)nr_of_pixels;
  float half_sensor = SS_SENSOR_LENGTH / 2;
  float dist_from_start = (average_peak * SS_PIXEL_LENGTH +
     SS_PIXEL_LENGTH / 2);
  *peak = dist_from_start - half_sensor;
  return true;
}
```

V Used technologies

Hardware:

- HP Pavilion Gaming Laptop 15-cx006nw (thesis author's personal computer)
- MSP430FR5969 LaunchPad™ Development Kit (MSP-EXP430FR5969)
- Nucleo496ZG-P development board
- Rotation bench (see more in Appendix VI)
- Raspberry Pi 3 Model B+
- ESTCube in-house developed image sensor test board in Figure 24 and Figure 25
- Thesis author developed adapter board for connecting Nucleo board and image sensors board in Figure 26 and Figure 27
- Thesis author developed Sun sensor breakout board in Figure 28 and Figure 29

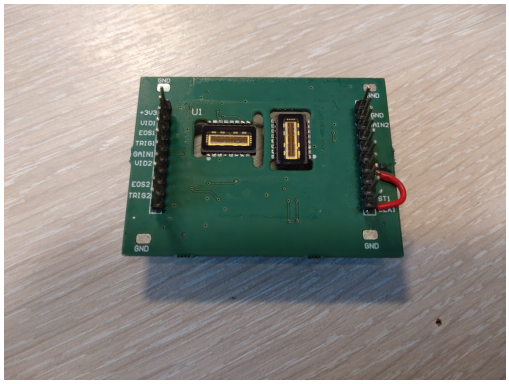

Figure 24. Image sensors without the mask.

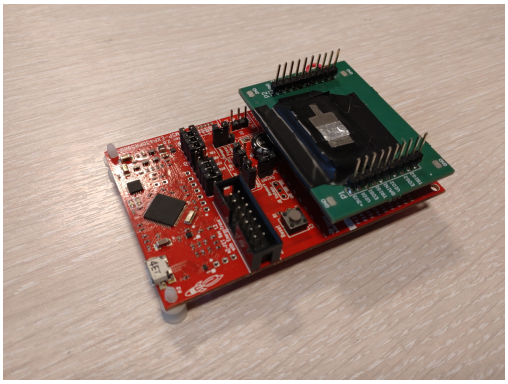

Figure 25. Image sensors board on the MSP board.

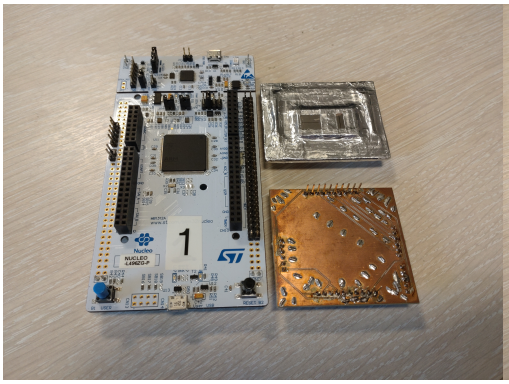

Figure 26. Separately Nucleo board, image sensors test board, adapter board.

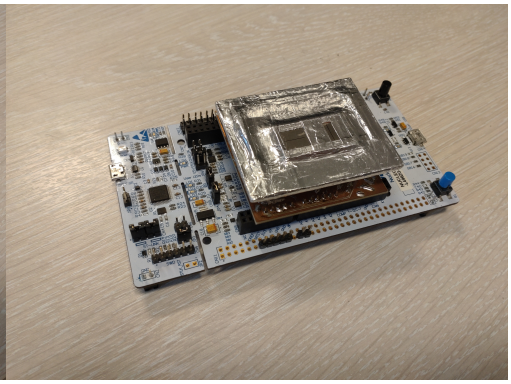

Figure 27. Stacked Nucleo board, adapter board, image sensors test board.

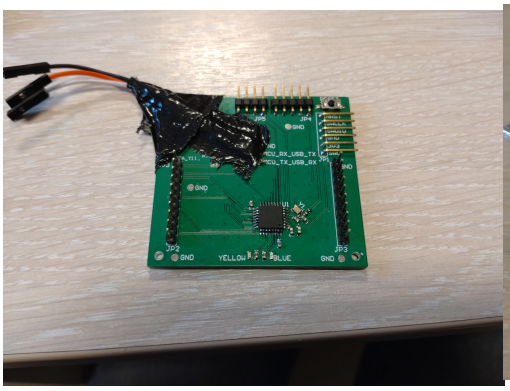

Figure 28. Sun sensor breakout board.

Figure 29. Sun sensor breakout board with image sensors board.

Software:

- Ubuntu 20.04.4 LTS
- Rasbian GNU/Linux 11 (bullseye)
- C standard C11
- GNU Arm Embedded Toolchain 10.3-2021.10 (includes GCC and GDB)

- GNU Make 4.2.1
- Open On-Chip Debugger 0.11.0 (OpenOCD)
- Python 3.9.7
- Jupyter Notebook 6.4.5
- conda 4.10.3 (Anaconda)

VI Cleanroom test setup

Testing was conducted in one of the University of Tartu Tartu Observatory clean rooms. This room was specially designed for optics testing. The tests were conducted by the author of this thesis and Hans Teras. The Sun sensor prototype (in Figure 30), a rotation bench (in Figure 32 and Figure 33) and a strong point of light (in Figure 31) was needed for testing.

Firstly, set up needed to be done. For this, the Sun sensor prototype was fixed onto a rotation bench, as shown in Figure 33. Then the light source and the prototype were aligned with a laser, as can be seen in Figure 34, Figure 35, and Figure 36. After that, measurement tools were removed from the rail, and the developer's computer was connected with the prototype and the rotation bench computer. Then strong light was turned on, and unnecessary light was reduced to a minimum resulting state as can be seen in Figure 37. Then the testing script was started, which controlled the rotation bench and gathered raw data from the prototype.

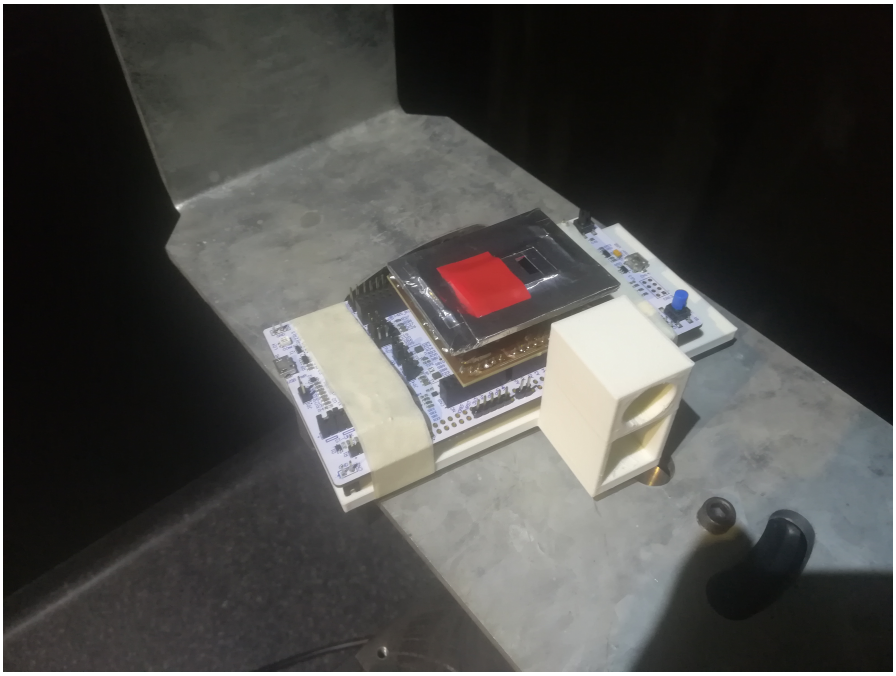

Figure 30. The Sun sensor prototype with one image sensor covered.

Figure 31. The strong point of light.

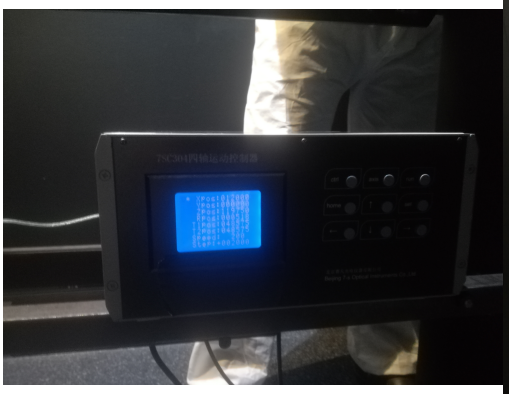

Figure 32. The rotation bench computer.

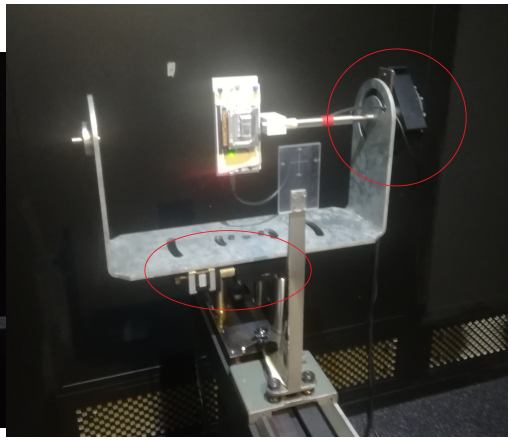

Figure 33. The rotation bench two motors.

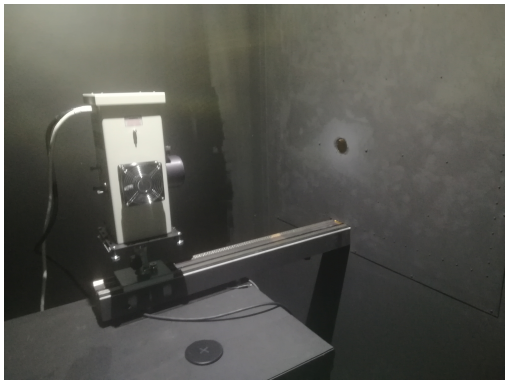

Figure 34. Light directed through the hole.

Figure 35. Using the laser to align the light source.

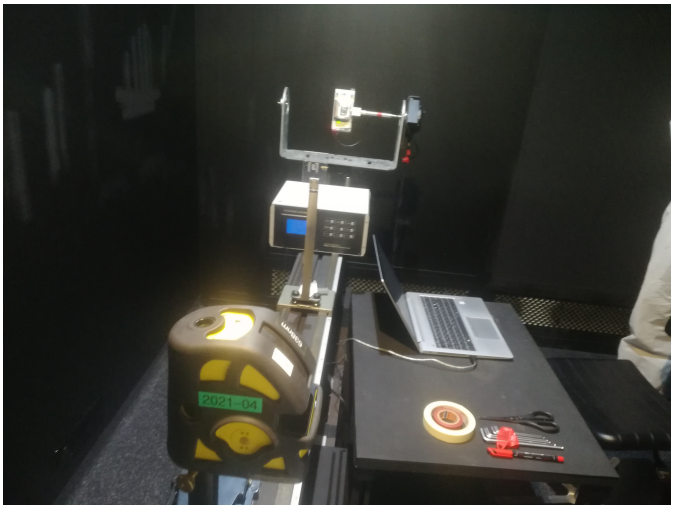

Figure 36. Using the laser to align the sensor prototype.

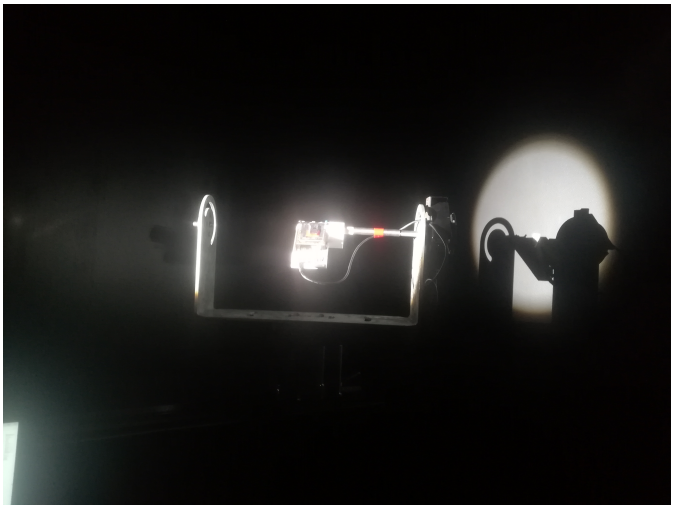

Figure 37. Conducting test measuring with Sun alike light.

VII Sun tracking test setup

Two tests were conducted in April 2022. For those tests were needed the Sun sensor breakout board with the image sensors test board, Raspberry Pi, power bank, and 3D printed holder. All this can be seen in Figure 38. Raspberry was used to control the prototype board, store the raw measurements and track time.

The first test measuring took place outside. The test setup was put on the tripod facing 45° upwards. It was necessary so that Sun would be in the FoV. Then it was directed in the same direction as the pavement, as shown in Figure 39. It was done this way because the approximate direction and location were later found from the Maaamet geoportaal map[4]. Pictures were taken after measurements when there was no direct Sunlight on testing equipment anymore. On this day, there was a considerable amount of wind and clouds.

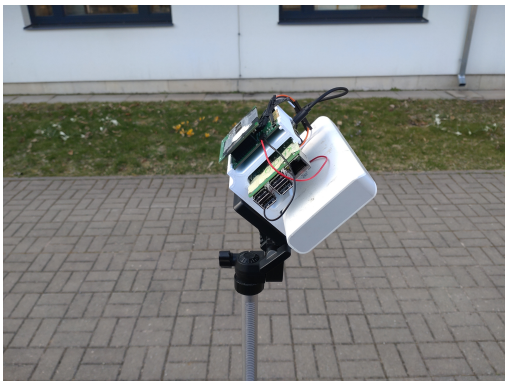

Figure 38. Testing device facing 45° upward.

Figure 39. Testing device aligned with the sidewalk.

The second test measuring took place at the author's home on a window sill. In this test, the testing setup was without a tripod, and the z-axis was facing horizontally out of the window. Test location and direction were found the same way as in the first test. This test was longer, and there were fewer clouds on that day.

[4]https://xgis.maaamet.ee/xgis2/page/app/maainfo

VIII Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Tiit Vaino**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Sun sensor software development for ESTCube-2,

   supervised by Hans Teras and Kristo Allaje.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Tiit Vaino
10/05/2022