

USN KONGSBERG

BACHELOR REPORT

ANDØYA SPACE EDUCATION

CanSat Redesign in CubeSat Frame

External Examiner: Harald Hovland

Internal Examiner: Karoline Moholth Mcclenaghan

Students:

Supervisors:

Daniel Østmoen (DØ)

Alberto Jimenez (Internal)

Glenn Andre Lyster Karlsen (GALK)

Kristian Enoksen (External)

Mohamed Reda El Meallem (RE)

Nathanael Nega Getaneh (NNG)

Thomas Lunde (TL)

Sebastian Tveito Benjaminsen (STB)

May 21, 2024

Acknowledgment

The bachelor group C.R.A.F.T extends our deepest gratitude to all the people who contributed and helped us during this bachelor's project. Without their support, guidance, and encouragement this project would not have been possible. We would like to specifically express our sincere appreciation to the following individuals and companies:

Alberto Jimenez: We are immensely grateful for his invaluable guidance and expertise throughout the entirety of this project. He provided insightful feedback, pushed us to think critically, and steered us in the right direction.

Kongsberg Automotive: We would like to thank them for their time and lending of their industrial freezer. This helped us immensely during our testing and it would not have been possible without their help.

Jegerservice, Ivar Fæhn Nilsen: We would like to thank Ivar for sharing his knowledge and enthusiasm regarding radio use. This made the laws and regulations regarding radio less intimidating for us to overcome.

Faculty Members: We would extend our thanks to all the faculty members who have shared their knowledge, expertise, and equipment with us during this journey, you have helped us with problems and given us feedback when we needed it and we are grateful.

Vestsiden Ungdomskole We would also like to extend our sincere thanks to Vestsiden Ungdomskole for allowing us to conduct product testing with their students. Their collaboration has been invaluable to our project.

Technip FMC Metering We would like to thank Technip FMC Metering for providing us with free furniture for storage in our project room during the bachelor project.

Jørgen Reisersøl We would like to thank Jørgen Reisersøl for giving us a speaker due design a PoC for our DIY shaker, free of charge

Halvor Haugen We would like to thank Halvor Haugen for renting us a car battery to use for our DIY vibration shaker.

Andøya Space Education: Lastly we would like to thank Andøya Space Education and our External supervisors Kristian Enoksen and Marcos Fraga for their feedback and guidance during this project. We appreciate the trust they have given us by making this project possible

Contents

1	Introduction	19
1.1	Group introduction: C.R.A.F.T	21
1.1.1	Scope	21
1.2	C.R.A.F.T-Sat	22
1.2.1	System Context	23
1.3	Group's members	24
1.3.1	Role distributions	24
1.4	Andøya Space Education	26
1.5	Rationale Behind Assignment	27
2	Work Method	29
2.1	Azure DevOps	29
2.1.1	Azure Boards	29
2.1.2	Epics	30
2.1.3	Features	30
2.1.4	User stories	31
2.1.5	Tasks	33
2.2	Secure Documentation and Development	34
2.2.1	Azure Repos (Version control)	34
2.2.2	Regular Backups	35
2.3	Visio Method	35
2.4	Length for each sprint	36
2.5	Weekly schedule	36
2.5.1	Ensuring good communication	36
2.6	Timetable	36
2.6.1	Terminology Explained	37
2.6.2	Roles explained	38
2.6.3	Sprint Master	38
2.7	Decision making	38
3	Requirements	40
3.1	Stakeholder Requirements	40
3.2	Functional requirements	42
3.3	Design Requirements	43
3.4	Testing Requirements	43
3.4.1	26 km range	44
3.4.2	Ease of Use and Documentation	44
3.4.3	Law and Regulations	45
3.4.4	Accessibility and Longevity	45
3.4.5	Functional Requirement Testing	46
3.4.6	Design Requirement Testing	48
3.4.7	Stakeholder Requirement Testing	48
4	Risk Analysis	50
4.1	Risk Assessment	50

4.1.1	Technical Risk Analysis: Electronics Field	51
4.1.2	Risk Assessment: Early Phase	51
4.1.3	Risk Assessment: Middle Phase	53
4.1.4	Risk Assessment: Late Phase	53
4.1.5	Mitigation Strategies for Risks in Electronics Field	54
4.1.6	Effectiveness of Mitigation Strategies	56
4.1.7	Technical Risk Analysis: Data Field	57
4.1.8	Mitigation Strategies for Risks in Data Field	59
4.1.9	Effectiveness of Mitigation Strategies	59
5	Preparatory Work	61
5.1	UV Radiation and Heat	62
5.1.1	Ozone Layer	62
5.2	Prototype material	65
5.2.1	Polyactic Acid (PLA)	65
5.2.2	Polyethylene Terephthalate Glycol (PETG)	65
5.2.3	Polyether Ether Ketone (PEEK)	66
5.3	Pugh-matrix and selection of prototype material	66
5.4	Polymer tension staff-test plan	67
5.4.1	The production phase	67
5.4.2	The test preparation phase	68
5.4.3	The tensile testing phase	69
5.5	Results from tension test and final polymer conclusion	70
5.6	Laws and regulations regarding radio and sensors	75
5.7	Pins used in software	76
5.8	Deciding radio module library	78
5.9	Deciding GY-91 library	78
5.10	Impact of Quality of Data Transmission: Doppler Shifting	79
5.10.1	Dedicated Software: SDRSharp	80
5.10.2	Hard-coding Doppler Shift Correction	81
5.11	Understanding our Target Audience - User Testing	82
5.12	Deciding Medium for Storage	83
5.12.1	Text File Types	83
5.12.2	Databases	84
5.13	Setting up Coding Environment	85
5.14	Choosing GUI Library	86
5.14.1	Tkinter	86
5.14.2	Comparison with Other GUI Libraries and Frameworks	86
5.14.3	Conclusion	87
6	Design Phase	88
6.1	Design process	88
6.2	Creation of the C.R.A.F.T-Sat frame	89
6.2.1	Brainstorming - C.R.A.F.T-Sat frame	89
6.3	Creation of the antenna release mechanism	90
6.3.1	Brainstorming - antenna release mechanism	90

6.4	Evaluation & selection of the antenna release mechanism & C.R.A.F.T-Sat frame	92
6.5	Timeline - iterations	96
6.6	Battery Heat Flux	96
6.7	Design Consideration: Electromagnetic Interference (EMI) and Thermal Management	99
6.7.1	Electromagnetic Interference (EMI)	99
6.7.2	Thermal Management	99
6.7.3	Mitigation Techniques	100
6.8	Prework for Component Placement: EMI Theory and Simulation	100
6.9	Voltage Regulator: Thermal Distribution	112
6.10	PCBs Component Placement	113
6.10.1	Component Placement	113
6.11	Shield circuit board	115
6.11.1	Overview	115
6.11.2	Dimensions of the circuit board	115
6.12	The size limit	116
6.12.1	Break out pins	116
6.12.2	Components	117
6.12.3	Ground plane and trace angles - Noise prevention	121
6.12.4	Ordering gerber files from manufacturer	122
6.12.5	Ground plane testing	122
6.12.6	Hand soldering the components to the PCB	123
6.13	Aerosols in the atmosphere	125
6.13.1	Water conductivity	125
6.13.2	Protecting electronics against moisture	125
6.14	Positioning system	125
6.14.1	The requirement for a tracking module	125
6.14.2	GNSS in general	126
6.14.3	The GNSS module	126
6.14.4	Components	126
6.14.5	NEO-6	126
6.14.6	NEO-M9N	126
6.15	Voltage Regulator	126
6.16	Antenna release electronics	128
6.16.1	Antenna activation	128
6.16.2	Current calculations	128
6.16.3	PWM to DC converter	131
6.16.4	DC Signal amplification	131
6.16.5	Components	131
6.16.6	Burn-wire	132
6.16.7	Simulation - PSpice	132
6.16.8	Connection and power	134
6.16.9	Circuit Schematic - Burn-Wire	134
6.17	Test Results	135
6.18	Connection of antenna to SMA connector	136

6.18.1	Quick overview	136
6.18.2	Connection	136
6.18.3	Material	137
6.18.4	Testing	137
6.18.5	Balun	138
6.18.6	Antenna matching	139
6.19	Components Weight	141
6.19.1	Weight Components: Estimation from Datasheet	141
6.20	Sensors Selection for Expansion Card	142
6.20.1	Advantages for Adding Expansion Card in C.R.A.F.T-Sat	142
6.20.2	Advantages for Implementing Expansion Card	142
6.20.3	Sensors Selection for Expansion Card	143
6.20.4	Significance of Measuring UV, CO ₂ , and Humidity, and Capturing Images	146
6.21	Design of Expansion Card	147
6.21.1	Components Implemented in Expansion Card	148
6.21.2	Ground Plane and Trace Angle	151
6.21.3	Increased Pad Size	151
6.21.4	Ground Plane Testing	151
6.21.5	Ordering The Expansion Card	151
6.22	Battery Selection for CanSat and C.R.A.F.T-Sat Mission	151
6.22.1	Li-ion Batteries	151
6.22.2	LiPo Batteries	152
6.22.3	Battery Selection: Risk Analysis	153
6.22.4	Battery Selection Process for C.R.A.F.T-Sat	154
6.23	Power Consumption for CanSat and C.R.A.F.T-Sat	156
6.24	Battery Test	156
6.24.1	Prework: Battery Isolation Before Test Execution	156
6.24.2	Battery Testing: Process and Results	158
6.24.3	Vacuum Test for Battery	159
6.25	Solar Cell Panels for Battery Charging in C.R.A.F.T-Sat	161
6.25.1	Purpose of Integration of Solar Panels in C.R.A.F.T-Sat	161
6.25.2	Seeed Studio 0.5W Kit solar panel: Technical and Physical Specification	161
6.25.3	Design Considerations For Solar Panels in C.R.A.F.T-Sat	162
6.25.4	Solar Panels in Seria: Voltage Output	170
6.25.5	Solar Charge Controller	173
6.25.6	DFRobot Solar Power Manager 5V: Main Features Explained Simply	175
6.25.7	Using Solar Charge Controller to Charge 3.7V Battery With Solar Panel	180
6.25.8	Simple Design Illustration of C.R.A.F.T-Sat with Solar Panels	181
6.25.9	Solar Panel Exclusion and Future Opportunities	183
6.26	Solar Cell Panels for Battery Charging in CanSat	185
6.26.1	Solar Panel Integration In CanSat	185
6.26.2	Challenges and Possible Solutions	185

6.26.3	Strategic Placement of Solar Cells on CanSat	185
6.26.4	Flexible Solar Panel from DFRobot	190
6.27	Antenna Research For C.R.A.F.T-Sat	190
6.27.1	Antenna Comparison	191
6.27.2	Reasons for Selecting the Dipole Antenna	194
6.27.3	The Length of the Dipole Antenna	195
6.28	Dipole Antenna: Design, Simulation, and Test	196
6.28.1	Dipole Antenna: Design in MatLab	196
6.28.2	Analyzing The Results Obtained in MatLab Simulation	200
6.29	Real World Antenna Test and Analysis	201
6.30	C.R.A.F.T-Sat Library Design	204
6.31	Real-Time Data Transfer	205
6.31.1	MVP Solution (Offline mode)	206
6.31.2	Uplink Solution	206
6.32	Data architecture	207
6.32.1	Easy to use	208
6.32.2	Longevity: Last for 10 years	208
6.32.3	Free and open source	208
6.32.4	System context	209
6.32.5	Using an External Server	209
6.32.6	Using Databases	210
6.32.7	Final Design: Flask and databases on Remote server	211
6.32.8	Deploying C.R.A.F.T-GUI - Pyinstaller	216
6.33	Safety Implementations for C.R.A.F.T-GUI	217
6.33.1	User Authentication - TOTP	218
6.33.2	User Authentication - Password Hashing	220
6.34	Setting up SQL database - Database Modelling	220
6.35	Setting up SQL database - Database Normalization	225
6.36	GUI Frontend Design	227
6.36.1	C.R.A.F.T-GUI - Code Structure	233
6.37	DIY Vibration Test	236
6.37.1	Solution - Homemade Shaker using a Speaker	236
7	Production	240
7.1	Sensor Array	241
7.2	Honoring duty cycles	242
7.3	Setting up SQL Database	245
7.4	Connecting SQL database to Microsoft Azure	245
7.5	Protecting SQL Database	247
7.5.1	Storing Database Credentials	247
7.5.2	Protecting SQL Database - Input Sanitation	247
7.5.3	Programming SQL Database	248
7.6	Using Tkinter and CustomTkinter - Frontend	248
7.6.1	Loading up C.R.A.F.T-GUI - SampleApp()	248
7.6.2	Implementing Layout - Frames	249
7.6.3	Creating Buttons	252

7.6.4	Adding Background Image	253
7.6.5	Instruction Label	253
7.6.6	Configuration Page	256
7.6.7	Administrators - GUI	257
7.7	Cleaning up code	257
7.8	Logging Page	258
7.8.1	General	258
7.8.2	User Capabilities	259
7.8.3	Admin Capabilities	259
7.8.4	SaveHistory Class Overview	260
7.9	The Plot Interface	261
7.9.1	File Management System in the Plot Interface	263
7.9.2	Serial Port Integration	265
7.9.3	Uplink to Satellite	266
7.9.4	Graph-, Table-, and GPS-Plot in Real-Time	268
7.10	Deploying a Flask Application to Google Cloud	269
7.10.1	Dockerizing the Flask Application	270
7.10.2	Configuring cloudbuild.yaml	270
7.10.3	Granting Necessary Permissions	270
7.10.4	Modularizing the Flask Application	271
7.10.5	app_backend.py	271
7.10.6	backend.py	272
7.11	Running Flask and Tkinter Concurrently	272
7.11.1	gui_poc.py	272
7.11.2	Conclusion	273
7.12	The final C.R.A.F.T-Sat frame and release mechanism	274
7.13	Finite Element Analysis (FEA) on the final design	276
7.13.1	FEA: Thermal analysis	277
7.13.2	FEA: Static analysis	278
7.13.3	FEA: Natural frequency analysis	280
7.13.4	FEA: Drop test	281
7.14	Conclusion about the design related to FEA	282
8	Testing Phase	283
8.1	PCB Vibration Test	283
8.2	Freeze Test Electronics	284
8.3	Noise testing of electronics	284
8.4	Re-usability of antenna release mechanism	285
8.4.1	Burn-wire placement	285
8.4.2	Spring releasing	285
8.4.3	Fishing rope mounting	286
8.4.4	Sealing & burn-wire releasing	287
8.4.5	Test conclusion	287
8.5	3D printed prototype	287
8.5.1	1U test pod - frame size	288
8.5.2	Design requirements approved	289

8.6	C.R.A.F.T-Sat library testing	289
8.6.1	Sending and receiving	289
8.6.2	Sensor testing	290
8.6.3	User Testing	290
8.6.4	GUI Designs	290
9	Verification Phase	295
9.1	Objective 2 & 3	295
9.2	Objective 5	295
9.3	Object 6	297
10	Validation Phase	298
11	Conclusion	299
References		300
Bibliography		310
A Appendices		311
A.1	CanSat Competition Requirements	312
A.2	CanSat Input/Output blackbox diagram and GY-91 block diagram documentation	314
A.3	What is LoRa?	316
A.4	CanSat	317
A.5	CubeSat	317
A.6	CanSat Package Diagram 1. iteration Documentation	318
A.7	Class diagram 1. iteration documentation	319
A.8	Harsh top lines in 3D print Model	321
A.9	Activity diagram documentation	322
A.10	Implementing SD Card	323
A.11	GPS implementation	324
A.12	Communication Protocols	325
A.12.1	Universal asynchronous receiver-transmitter	325
A.12.2	Serial peripheral interface	326
A.12.3	Inter-Integrated Circuit	327
A.13	LoRa Packet	329
A.14	Using the C.R.A.F.T-Sat Library	330
A.14.1	terminology	330
A.15	C.R.A.F.T-Sat library User guide	330
A.15.1	Example programs	333
A.16	Library Installation Guide	337
A.17	C.R.A.F.T-Sat Library API	340
A.18	ESP32 Camera Experiment	344
A.19	CanSat frame documentation	347
A.20	Relevant CubeSat Design Specification (1U – 12U) REV 14.1 CP-CDS-R14.1.348	

B Tools for GUI	350
B.1 Use of Tkinter	350
B.1.1 Ttkbootstrap error	350
B.2 PEP 8 – Style Guide for Python Code	352
B.3 JetBrains AI Service	354
B.3.1 Map Plot with coordinates	354
B.3.2 RSSI Plot	357
C Making the plot interface	362
C.1 Real-time RSSI Plotting	362
C.2 Plot Pop-Up Buttons	363
C.3 ‘Add Plot’ buttons with ‘plot options’ popup	365
C.4 ‘Plot option’ replaces ‘Add Plot’ button	365
C.5 New ‘Add Plot’ button appears after plot	367
C.6 Solution with PyQt	369
C.7 Tkinter with inspiration from pyqt program	370
C.8 Plot on a grid	371
C.9 Plot on button-grid	371
C.10 Save to file	372
C.11 Serial port	374
C.12 Thread v1	375
C.13 Queue (dead end)	377
C.14 Thread v2	378
C.15 Read and store all sensor data	379
C.16 Dropdown, datapoint and merged file format	382
C.17 ID check	386
C.18 Plot and button design	389
C.19 Converting from ttkbootstrap to CustomTkinter	392
C.20 Nested frames for plot buttons	393
C.21 Plug and play with auto-detection for serial port	394
C.22 Integrated MongoDB and added Logging page	395
C.23 Table- and GPS-Plot implementation	396
C.24 Exit button for plot frames and Uplink to satellite	397
D Lab report voltage regulator testing	399
D.1 Quick overview	399
D.2 Equipment	399
D.3 Procedure	399
D.4 Results	399
E Lab report antenna testing	401
E.1 Quick overview	401
E.2 Procedure	401
E.3 Results	404
F Desoldering of components	405

G Burn-Wire Prototype	409
G.1 KiCad	412
G.1.1 Software Structure	412
G.2 Teensy 4.1	413
G.2.1 Description of Teensy 4.1	413
G.2.2 Key Features of Teensy 4.1	413
G.3 Buzzer	415
G.3.1 Results from low pressure testing	415
G.3.2 Gyroscope GY-91	416
G.3.3 BMP-280 Barometric Pressure Sensor	416
G.3.4 MPU-9255 Multi-Sensor Package:	416
G.3.5 NTC Temperature Sensor	418
G.3.6 RFM96 Radio	419
G.4 Backup antenna	421
G.5 Gantt diagram	422
G.6 Project Progress Report	423
G.6.1 System overview	423
G.6.2 Project Management and Planning	423
G.6.3 The Progress in Electronic System	424
G.6.4 The Progress in Data System	424
G.6.5 The Progress in Mechanical System	425
H Burn-wire description "Final design"	427
H.1 Why is a burn-wire mechanism defined as reusable	427
H.2 How does it work with the sealings	427
I The optimal material for the C.R.A.F.T-Sat using Additive Manufacturing method	429
J Fused Deposition Modeling (FDM)	430
K Cura; Slicer settings	431
K.1 Cura; slicer settings decided to be tested - variable	431
K.2 Cura; slicer settings decided to be constant	433
L Selective Laser Melting (SLM)	434
M User manual - Assembly & disassembly the C.R.A.F.T-Sat	435
M.1 Description about this manual	435
M.2 Items needed to assembly C.R.A.F.T-Sat	435
M.3 3D printing: Manual and explanation - Final design	436
M.4 Circuit boards - sheet	472
M.5 Assemblies	473
N Rough Mechanical Test Plan	479
O Documentation about polymer	480

O.1	Glass-transition temperature	480
O.2	Thermoplastics	482
O.2.1	Effects of temperature	482
O.2.2	Water Absorption	482
O.2.3	Thermal and Electrical Properties	482
O.2.4	Material data from a digital database	482
P	Tension testing results	484
P.1	Material: PETG x Infill pattern: Cubic Infill density: 50% Preparation temp.: -18 to -20 Celsius	484
P.2	Material: PETG x Infill pattern: Cubic Infill density: 50% Preparation temp.: -38 to -40 Celsius	489
P.3	Material: PETG x Infill pattern: Cubic Infill density: 50% Preparation temp.: 20 to 24 Celsius	496
P.4	Material: PETG x Infill pattern: Cubic Infill density: 75% Preparation temp.: -18 to -20 Celsius	503
P.5	Material: PETG x Infill pattern: Cubic Infill density: 75% Preparation temp.: -38 to -40 Celsius	510
P.6	Material: PETG x Infill pattern: Cubic Infill density: 75% Preparation temp.: 20 to 24 Celsius	517
P.7	Material: PETG x Infill pattern: Gyroid Infill density: 50% Preparation temp.: -18 to -20 Celsius	525
P.8	Material: PETG x Infill pattern: Gyroid Infill density: 50% Preparation temp.: -38 to -40 Celsius	533
P.9	Material: PETG x Infill pattern: Gyroid Infill density: 50% Preparation temp.: 20 to 24 Celsius	541
P.10	Material: PETG x Infill pattern: Gyroid Infill density: 75% Preparation temp.: -18 to -20 Celsius	549
P.11	Material: PETG x Infill pattern: Gyroid Infill density: 75% Preparation temp.: -38 to -40 Celsius	557
P.12	Material: PETG x Infill pattern: Gyroid Infill density: 75% Preparation temp.: 20 to 24 Celsius	565

P.13	Material: PLA x Infill pattern: Cubic Infill density: 50% Preparation temp.: -18 to -20 Celsius	572
P.14	Material: PLA x Infill pattern: Cubic Infill density: 50% Preparation temp.: -38 to -40 Celsius	580
P.15	Material: PLA x Infill pattern: Cubic Infill density: 50% Preparation temp.: 20 to 24 Celsius	587
P.16	Material: PLA x Infill pattern: Cubic Infill density: 75% Preparation temp.: -18 to -20 Celsius	595
P.17	Material: PLA x Infill pattern: Cubic Infill density: 75% Preparation temp.: -38 to -40 Celsius	603
P.18	Material: PLA x Infill pattern: Cubic Infill density: 75% Preparation temp.: 20 to 24 Celsius	611
P.19	Material: PLA x Infill pattern: Gyroid Infill density: 50% Preparation temp.: -18 to -20 Celsius	619
P.20	Material: PLA x Infill pattern: Gyroid Infill density: 50% Preparation temp.: -38 to -40 Celsius	626
P.21	Material: PLA x Infill pattern: Gyroid Infill density: 50% Preparation temp.: 20 to 24 Celsius	634
P.22	Material: PLA x Infill pattern: Gyroid Infill density: 75% Preparation temp.: -18 to -20 Celsius	642
P.23	Material: PLA x Infill pattern: Gyroid Infill density: 75% Preparation temp.: -38 to -40 Celsius	650
P.24	Material: PLA x Infill pattern: Gyroid Infill density: 75% Preparation temp.: 20 to 24 Celsius	658
Q	ISO Standards: Documentation, modeling- & test specimen	666
Q.1	NS-EN ISO 527-1:2019 Determination of tensile properties	666
Q.2	NS-EN ISO 291:2008 Plastics - Standard atmospheres for conditioning and testing.	668
Q.3	NS-EN ISO 527-2:2012 Determination of tensile properties - Part 2: Test conditions for moulding and extrusion of plastics.	669
R	Simple CAD	671

S Hydrostatic pressure distance in gasses	672
T System Interface diagram	674
U Simple FEA guide used with Vestsiden secondary school	676
U.1 What need's to be tested	677
U.2 How can we test it	679
U.3 Why was it tested	680
V Finite Element Analysis - Reports	692
V.1 Static analysis	692
V.2 Thermal Analysis	731
V.3 Natural frequency	761
V.4 FEA: Harmonic dynamic analysis	815
V.5 Drop test	854
W Vacuum Test	876
X C.R.A.F.T-Sat mounting description used during visit: Vestsiden secondary school	882
X.1 Items needed to assembly C.R.A.F.T-Sat	882
X.2 Mounting manual of C.R.A.F.T-Sat frame	886
X.3 Risks	891
Y Helical Spring - Antenna release mechanism calculation	892
Z Timeline-iterations	894
Z.1 First Timeline Iteration	894
Z.2 Second Timeline Iteration	898
Z.3 Third Timeline Iteration	907

Glossary

3D printing It's a method of producing a prototype using technology which use powdered metal or plastic and other adhesive materials to construct object layer-by-layer.. 14, 21, 41, 65, 67, 68, 70, 429, 430, 882

Additive Manufacturing It's the process of creating an object by building it one layer at a time. It typically refers to 3D printing.. , 65, 429, 434, 436, 671

CanSat Is a representaiton of a satelite where all components must fit within the volume of a soda can. This is used as a teaching platform to introduce students to space technology. , 21–23, 27, 40, 42–45, 47, 55, 56, 75, 76, 78–80, 93, 99, 109, 115, 116, 123, 125, 127, 134, 141, 151, 154, 184–187, 189, 190, 208, 222, 229, 236, 238, 299, 312–314, 317–319, 322, 324, 330, 347, 413–415, 479

CubeSat Is a nanosatellite where their size is based on a standard size called Unit(U), One U is approximate 10cm x 10cm x 11cm. , 21–23, 27, 28, 40–44, 47, 54, 55, 65, 93, 115, 191, 192, 195, 282, 288, 295, 299, 317, 321, 348, 415, 427, 429

FR-4 Is a representaiton of a satelite where all components must fit within the volume of a soda can. This is used as a teaching platform to introduce students to space technology.

heat flux It's a definition of how much heat energy passes through a certain surface.. , 277–279

infill It's an % amount of material inside the body of a 3D print.. , 906

Product Strength It represent the strength of a chosen material in a Rapid Prototyping, related to the material data and printing data..

Rapid Prototyping It's a group techniques used to quickly produce a physical part, model or assembly using Computer Aided Design (CAD).. 14, 65, 429, 434, 895, 897, 904

timeline-iteration Is a definition of mechanical design - timeline work between submission dates.. , 96, 894–899, 901–903, 905, 906

Acronyms

1:1 One-to-one. , 224

1:m One-to-many. , 222, 224

AC Alternating Current.

AFA Adaptive Frequency Agility. , 75

API Application Programming Interface. , 247, 248

ASE Andøya Space Education. , 24, 26–28, 30, 34, 40, 42–46, 51, 52, 54, 59, 63–65, 78–81, 83, 84, 86, 95, 141, 146, 151, 154, 184, 185, 203–205, 207–210, 212, 214, 217, 218, 222, 224, 225, 228, 234, 236, 241, 245, 248, 249, 256, 258, 277, 298, 314, 315, 317, 319, 323, 331, 344, 370, 418, 424, 900, 901

BOM Bill of material. , 676

C.R.A.F.T CanSat Research and Advanced Fabrication Team. , 21, 45, 83, 84, 100, 209, 248, 332, 480, 882

C.R.A.F.T-GUI GUI used to plot telemetry data. , 82–84, 207–212, 214–220, 222–234, 247, 249, 254, 256–258, 290, 298

C.R.A.F.T-Sat CanSat Research and Advanced Fabrication Team Satellite. , 21–23, 40–45, 47, 48, 56, 63, 65–68, 75, 76, 79, 80, 88–91, 96, 97, 99, 109, 114, 115, 123, 125, 128, 135–137, 141–143, 146–148, 151, 153, 154, 156, 161, 162, 164, 169, 173–175, 181, 183, 184, 190–194, 196, 197, 204–208, 222, 229, 236, 238, 241, 242, 261, 274–277, 279–283, 285–289, 295, 296, 299, 325–327, 330, 337, 339, 344, 347, 362, 401, 404, 428, 429, 431, 438, 441, 443, 446, 449, 452, 454, 455, 458, 459, 462, 464, 467, 468, 470, 471, 484, 489, 497, 504, 511, 518, 525, 533, 541, 549, 557, 565, 573, 580, 588, 595, 603, 611, 619, 627, 634, 642, 650, 658, 672, 676, 677, 679, 680, 692, 882, 892, 894–896, 898, 900, 906

CAD Computer Aided Design. 14, 67, 89, 90, 274, 276, 436, 451, 454, 892, 894, 895, 897, 898, 900, 905

CO2 carbon dioxide. , 143, 144, 146, 148

DBMS Database Management Systems. , 210, 224

DC Direct Current. , 128, 131, 409

DIY Do It Yourself. , 56

DMLS Direct Metal Laser Sintering.

DMPTM Digital Motion ProcessorTM. , 417

EMC Electromagnetic Compatibility. , 100, 196, 201

-
- EMI** Electromagnetic Interference. , 99, 100, 113
- ESA** European Space Agency. , 40, 43, 45, 312, 317
- ESRO** European Space Education Resource Office. , 317
- FDM** Fused Deposition Modeling. , 65–67, 70, 274, 288, 429–431, 433, 436, 671
- FEA** Finite Element Analysis. , 65, 70, 71, 73, 276, 277, 280, 281, 482, 676, 677, 680, 692, 892
- FEM** Finite Element Method. , 276, 676, 679
- FK** Foreign Key. , 222
- FR-4** Flame Retardant 4. , 283
- FSK** frequency shift keying. , 419
- GD & T** Geometric dimensioning and tolerancing.
- GDPR** General Data Protection Regulation. , 40, 217, 220
- GFSK** gaussian frequency shift keying. , 419
- GMSK** gaussian minimum shift keying. , 419
- GNSS** Global Navigation Satellite System. , 125, 126
- GPS** Global Positioning System. , 125, 126, 312, 315, 318, 324, 325
- Grms** root mean square of gravitational force. , 283
- GUI** Graphical User Interface. , 21–23, 25, 28, 40, 42, 43, 45, 47, 82–87, 205, 207, 208, 212, 214, 216, 227–229, 241, 247, 261, 289, 290, 293, 294, 296, 314, 322, 340, 341, 350, 354, 355, 362, 369, 424, 900
- HI** High Impact. , 50, 52, 53, 58
- HP** High Probability. , 50, 53, 58
- I²c** Inter-Integrated Circuit. , 21, 28, 43, 79, 143, 324, 325, 327, 328, 344, 345, 416, 417
- IC** Integrated Circuit. , 126, 127
- IoT** Internet of Things. , 419
- LBT** Listen Before Talk. , 75
- LI** Low Impact. , 50, 52, 53, 58, 59
- Li-ion** lithium-ion. , 151–154, 156
- LiPo** Lithium polymer. , 52, 62, 64, 97, 151–154, 156, 173–175, 190
- LoRa** Long Range. , 80, 117, 202, 203, 289, 296, 316, 322

LP Low Probability. , 50, 52, 53, 58

m:m many-to-many. , 224

MI Medium Impact. , 50, 52–54, 58

MP Medium Probability. , 50, 52–54, 58

MPPT Maximum Power Point Tracking. , 173, 174

MSK minimum shift keying. , 419

MVP Minimum Viable Product.

NASA National Aeronautics and Space Administration. , 161

NDT Nondestructive Testing.

NF Normalization Form. , 210, 225–227

NKOM Nasjonal kommunikasjonsmyndighet. , 40, 42

NTC Negative Temperature Coefficient. , 100, 112–114, 117, 141, 418

OOK on–off keying. , 419

OOP Object Oriented Programming.

PCB printed circuit board. , 40, 54–56, 99, 100, 109, 112–117, 121–123, 126, 127, 133, 141, 147, 148, 151, 154, 156, 158, 159, 184, 283, 401, 405, 407, 409, 412

PEEK Polyether Ether Ketone. , 65, 66

PETG Polyethylene Terephthalate Glycol. , 65–67, 69, 71–73, 276, 278, 279, 282, 285, 430, 433, 436, 482, 484, 489, 497, 504, 511, 518, 525, 533, 541, 549, 557, 565

PK Primary key. , 222, 225–227

PLA Polyactic Acid. , 65–67, 69, 71–73, 285, 430, 433, 436, 482, 573, 580, 588, 595, 603, 611, 619, 627, 634, 642, 650, 658

PWM Pulse Width Modulation. , 128, 131, 133–135, 143, 409

QR quick-response. , 218

RF Radio Frequency. , 99

SCC Solar Charge Controller. , 175, 190

SLM Selective Laser Melting. , 65, 429, 434

SMD Surface-mount devices. , 117, 123, 148

SPI serial peripheral interface. , 21, 28, 43, 77, 143, 323–328, 344, 345, 416, 417, 420

-
- SQL** Structured Query Language. , 218, 247
- STEM** Science, Technology, Engineering and Mathematics. , 27
- STL** STereoLithography. , 895, 897
- SW** Solid Works. , 67, 274, 276, 277, 436, 671, 894, 898
- SWS** Solid Works Simulation. , 276, 676
- TOTP** Time-based One-Time Passwords. , 218, 219
- UART** universal asynchronous receiver / transmitter. , 21, 28, 43, 143, 324, 325, 327, 328, 344–346
- UHF** Ultra-High Frequency. , 22, 424
- UML** Unified Modelling Language. , 228, 229
- UTS** ultimate tensile strength. , 72–74, 282
- UV** ultraviolet. , 143, 144, 146, 148
- VOC** volatile organic compounds. , 146

1 Introduction

TL|DØ: Thomas has contributed to and documented this task in the report. Daniel has reviewed it.

TL,DØ|RE: Both Thomas and Daniel have contributed equally to this task. Thomas has documented it in the Report. Reda has reviewed it.

TL, DØ|RE: Thomas and Daniel have both contributed to this task, with Thomas having the larger share of contributions. Thomas has documented it in the Report. Reda has reviewed it.

TL, DØ|RE: Both Thomas and Daniel have contributed to this task, Daniel has made the majority of contributions. Thomas has documented it in the Report. Reda has reviewed it.

1.1 Group introduction: C.R.A.F.T

In the start of this project, we started with defining a name of our group is CanSat Research and Advanced Fabrication Team (C.R.A.F.T). This name was generated using Chat-GPT 3.5.

Group-members in this project are based from three different disciplines of engineering study. Those fields are Mechanical Engineering, Electronic Engineering, and Computer Engineering. During the start of the bachelor report period, we started with a focus of understanding the assignment, and define a method of how the task should be solved.

1.1.1 Scope

The given assignment is to develop and produce a combination of CanSat - electronical components inside of a CubeSat frame. This provides the project with a wide array of standards and laws to adhere to. There is also a big difference between the design of the CubeSat and CanSat which will give a big opportunity to define and a high risk how it should be combined. To give this project a conclusion of how it is going to be produced, the chosen scope for this assignment is therefore:

Design a 1U CubeSat, which fits inside a CubeSat dispenser. The CubeSat design must include an internal design to fit CanSat properties. Those properties is used to receive and send received data down to a Graphical User Interface (GUI) during a specific period. The solution of how this assignment is solved will be an combination of requirements from the CubeSat and CanSat.

Our assignment for this project was divided into 7 interim goals:

1. Update design of CanSat NAROM 2018 circuit board in accordance with new requirement specifications. That is, upgrade the previous Teensy 3.5 microcontroller with Teensy 4.1. The circuit board must also have breakout points for serial peripheral interface (SPI), universal asynchronous receiver / transmitter (UART), Inter-Integrated Circuit (I^2C), and 3v3 and 5v power supply, with the ability to connect expansion circuit boards within the CanSat Research and Advanced Fabrication Team Satellite (C.R.A.F.T-Sat).
2. Design and produce a 3D printing of CubeSat 1U frame module that fits inside a CubeSat 1U test pod. The design must show consideration for future implementation of balloon mechanism for upward trajectory, but it is not a requirement to design a balloon fastener.

-
3. Design and produce an antenna release mechanism with an Ultra-High Frequency (UHF) antenna that fits to the CubeSat frame. This mechanism must be easily reusable, meaning that the mechanism can be reused for multiple launches (i.e., at least 10 launches).
 4. Develop circuit boards as sub modules of selected sensors which can easily be integrated in the CubeSat frame. The entire CubeSat system and its sensors must tolerate external surroundings up to a height of approximately 26 km (+5km).
 5. Program CubeSat's sub modules to measure data and establish downlink communication via RF signals between CubeSat and Ground Station within a specified time interval (<30 seconds). In addition to a default code configuration for plug-and-play functionality, the system should also provide the (well documented) library that was used to program the CubeSat.
 6. Program a GUI with graphical ground station software to be able to plot telemetry data within a specified time interval (<30 seconds from data being transmitted by the CubeSat), save data to file (i.e., text file) and check status on the CubeSat and all its sub modules.
 7. If time permits it, develop and complete a simplified test procedure for vibration testing of fully integrated CubeSat.
 - That is, conduct a simplified vibration test of the CubeSat + synthesise a procedure for students to conduct similar vibration tests.

STB|TL

1.2 C.R.A.F.T-Sat

During the start of this project, there have been difficulties understanding the core concept of our project, which involves redesigning a CanSat with a CubeSat design, as well as understanding the necessary requirements addressing this. The trouble has been because of the differences between a CubeSat and a CanSat. To solve this problem, we made the title C.R.A.F.T-Sat. This title is made to lock the statement about the description of the CanSat with CubeSat design.

C.R.A.F.T-Sat is a title for the description for the total system with the combination of CanSat and CubeSat. This title is used to give a definition for the requirements that should be used in our project.

- As mentioned, all electrical work and all software work, due to CanSat, is following the requirements for an CanSat.
- As mentioned, all electrical work, due to CubeSat, is following the requirements for an CubeSat.
- All mechanical work, due to the frame and antenna release mechanism is following requirements for a CubeSat.

NNG| TL

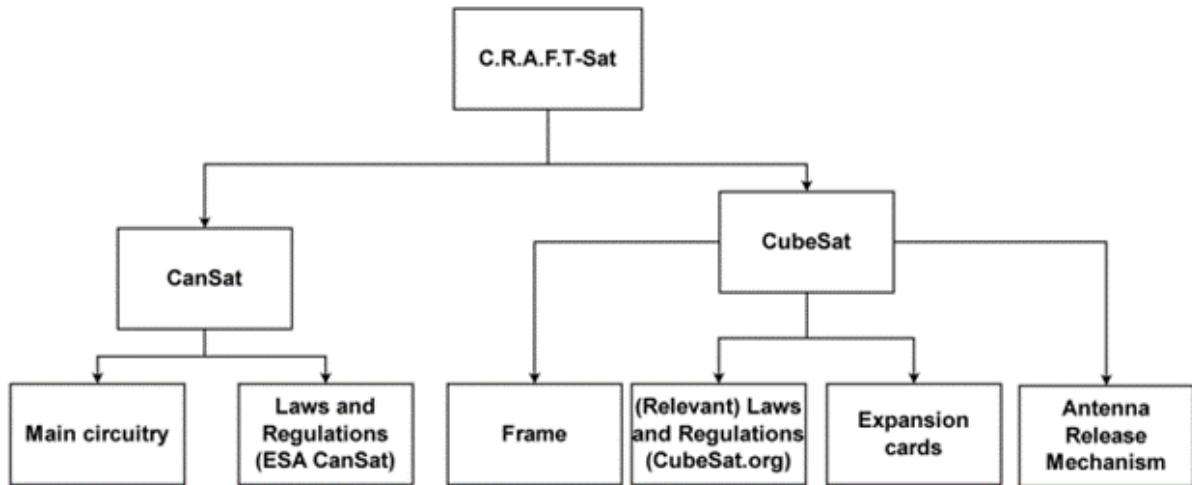


Figure 1: C.R.A.F.T-Sat | NNG

1.2.1 System Context

Our assignment is to **redesign the electronics from CanSat NAROM 2018**, so that it can be integrated in both a CanSat and 1U CubeSat frame. This system (from now on referred to as **C.R.A.F.T-Sat**), is **only** intended to be used by students (**from upper middle school to university level**) and for **educational purposes** only. This “hybrid” solution regarding the electronics is to make the C.R.A.F.T-Sat an intermediary between operating with CanSat and full-fledged satellites.

C.R.A.F.T-Sat is not intended to stay in orbit under any circumstances. It is, however, expected to operate at an altitude of approx. 26km(+/- 5km) above sea level using a balloon mechanism. The balloon in question has a **maximum load capacity of 500 grams**. The upward trajectory for the CubeSat can last for up to 1,5-3 hours before reaching its terminal height. Once the balloon has exploded (due to high pressure), the CubeSat can be in “free-fall” (without parachute) for approximately 73 seconds. The assignment involves **developing expansion cards** (i.e. sub modules) and a CanSat shield (i.e. circuit board that must fit inside both CanSat and CubeSat frame). These expansion cards are only intended for a CubeSat frame, not CanSat. The circuit board sub-modules from interim goal 4 [1.1.1] does **not** have to fit inside a CanSat frame.

C.R.A.F.T-Sat’s sensors should start measuring during upward *and* downward traversal. Once activated, the data collected by the sensors will be transmitted in **soft real-time** (i.e. no more than a **30 seconds for data to be received** to a Ground Station (another Teensy 4.1) through radio signals. In addition to this data transfer, data will also be **saved locally** to an SD card that has been inserted into the Teensy 4.1’s SD card slot and been placed inside the CubeSat frame. It is also expected that the data received by the Ground Station must be able to be saved locally as a text file. The telemetry data received by the Ground Station must be able to be viewed through a **GUI**. It is not a requirement for C.R.A.F.T-Sat to support uplink communication, but it would be a value-added feature.

It is intended that **students will program their own microcontroller using sensor**

libraries that have been provided by the development team. For plug-and-play functionality, it is expected for a default code configuration for the microcontroller to be provided by the software team in case the student does not want to tailor the program. It is not expected that the students will have to write code to plot themselves - the Ground Station Software must do it automatically for them, with option for custom filtering of data viewed. A desired feature is to have the ability to plot the satellite trajectory onto a map. There is network access for the students on Andøya Space Education (ASE)'s facilities. Thus, solutions involving use of external server is allowed, but it is still a requirement for the ground station to function offline as well.

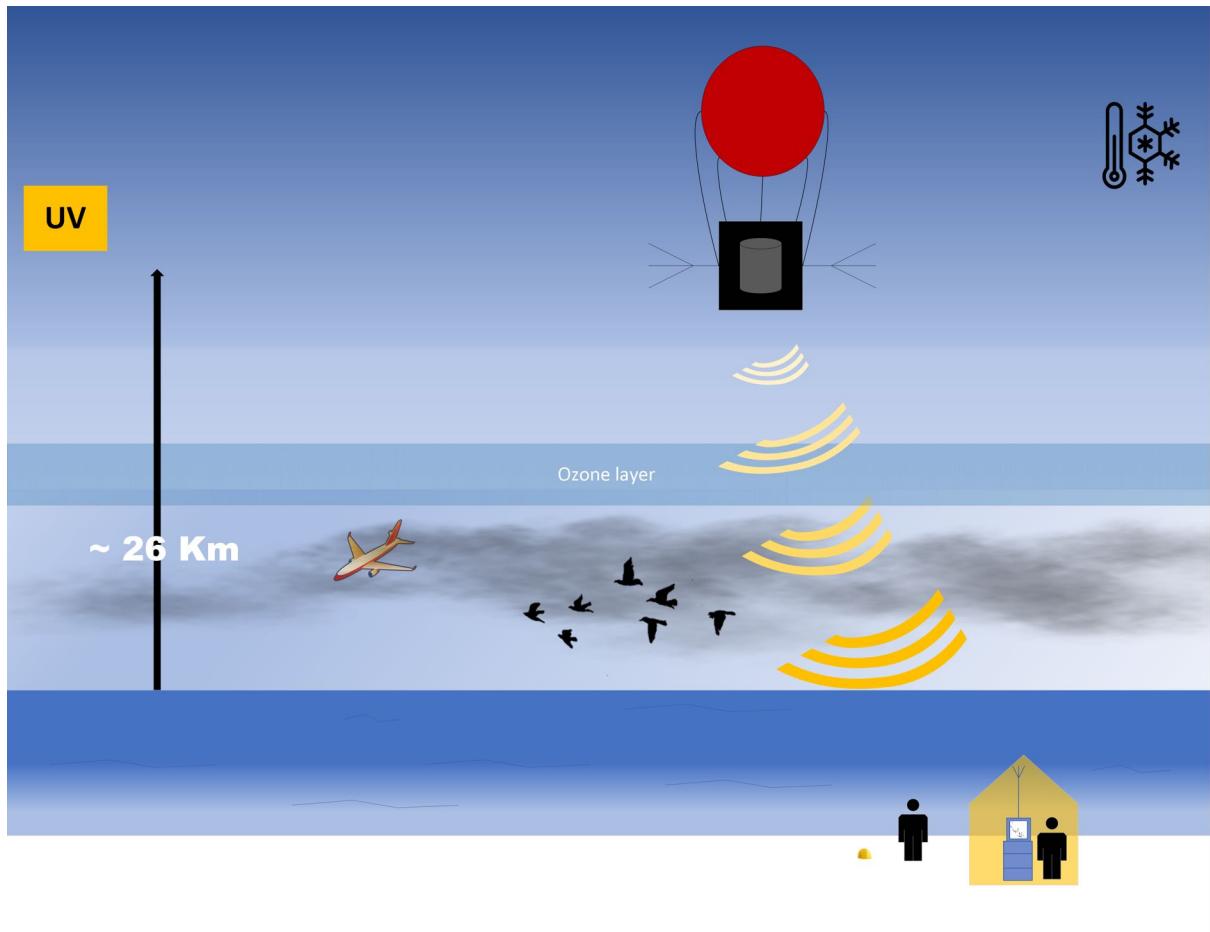


Figure 2: Context Diagram | NNG

1.3 Group's members

1.3.1 Role distributions

Nathanael Getaneh

- Group leader
- Product owner

-
- Developer (GUI)

23 years of age. Third year computer engineering student (Cyber Physical Systems). Have previous relevant experience from working at student projects and as an intern at the Norwegian Central Bank's IT department. Primarily a developer of the Ground Station software for this project.

Glenn Andre Lyster Karlsen

- Documentation lead
- Developer (Electrical)

I am 30 years old. I am born and raised in Kongsberg. My first education is being a skilled worker landscape gardener. My interests include music, electronic, photography, drawing & art, and motorcycles. Now I am studying mechatronics at Krona University in Kongsberg.

Daniel Østmoen

- Marketing (Website)
- Developer (GUI)

I am 23 years old and I'm from a town called Vestfossen, with the closest city being Hokksund. When I was younger I spent a lot of my time playing video games, and at high school I went on the "data og elektronikk" study in Kongsberg and been a sound technician for several years now. Which drove me to like computer and technology more and more. This resulted in when I decided to be an engineer, that my first choice was Cyber Physical System at USN Kongsberg. In my spare time I like to be active, where i like to workout and do all kinds of sports (volleyball, tennis, disc golf, ice hockey etc...).

Thomas Lunde

- Presentations
- Developer (Sensor programming)

I am 24 years old, from a small town called Ulefoss in Norway. I live with a significant other and a golden retriever. I previously worked as a construction machine operator. When I was finished with my apprenticeship I decided to take further education, to be able to work with something I was more passionate about. I have always been interested in computers and technology. For that reason, I decided to study Cyber Physical Systems at USN Kongsberg. In my spare time I like to workout, walk my dog and I have recently picked up 3D printing as a hobby.

Mohamed Reda El Meallem

-
- Health Safety and Environmental (HSE) lead
 - Report (LaTeX) lead
 - Developer(Electrical)

I am originally from Morocco. In 2019, I moved to Norway after living 2 years in Germany. In Norway, I have settled in the charming village of Revetal in Vestfold, a place I really enjoy. My fascination with Circuit Board Design, Signal Processing, and Cybernetics were the key factors in choosing Electronics and Cybernetics.

In my spare time, photography is my passion. I spend a lot of time in nature, capturing moments of beautiful wildlife and scenery with my camera. Photography has enhanced my respect for nature and increased my appreciation for all the living beings that are involved in it.

Sebastian Tveito Benjaminsen

- Marketing (SoMe)
- Developer (Mechanical)

My age are 24, and I'm from Porsgrunn, a city i Telemark. I started with a car mechanic education and ended after high school. I had a dream about being a mechanical engineer and started with pre-course to be qualified for my further education. From there, i started with the the study-line Product Development at University of Southeast - Norway (USN) campus Kongsberg, which gives me the title mechanical engineer.

During this study-line, i have fulfilled course's which gave me a lot of experience. Some of those topics includes topics equal simple electronics, calculus, product developing and finite element analysis, etc. Two of my most experienced courses was working tasks as an manufacturing engineer, and programming and micro-controller; a course where i developed a self-design "tanks" where you controlled the rolling belts, the shooting point, and launch of a toy missile, named Nerf. This was done by using mechanical engineering, electro engineering and software developing. Those engineering topics is my plan for further education after this assignment.

I don't have a lot of spare-time because of all the required work in my course's, but when i take brake's it's sometimes been used in a swimming pool or family visiting.

DØ, TL|DØ

1.4 Andøya Space Education

ASE is a non-commercial "ideal corporation (a non-profit limited company)" established by Andøya Space in 2000. They are supported by the Norwegian Ministry of education and research, where their primary objective is to increase the interest of space related activities for students [1]. Here are some different opportunities for courses and activities they offer to Teachers and Students [2]:

-
- “Fysikken i fokus ” Initiated in 2001. Is a recruitment program in collaboration with high schools, supported by the Research Council of Norway, the Norwegian Space Agency, and the National Center for STEM Education Recruitment.
 - “Teachers’ Space Camp” is a summer camp for teachers that integrates rocket building, basic sensors, and coding into the curriculum, aligning with the Norwegian National Curriculum Framework 2020.
 - “Vg2 Dronefag” certifies students in operating and maintaining diverse drones, emphasizing practical skills in planning, and executing drone operations.
 - “Vg3 Romteknologi” expands on “Vg2 Data og elektronikk” or “Vg2 Dronefag”, covering telemetry, remote sensing, geographic information systems, space technology, and satellite engineering to meet the growing demand in space-related industries.

These courses have cross-disciplinary educational projects encompass launching (CanSats), coding experiments for the International Space Station, lunar base construction, local climate project selection, and virtual space missions [3].

They also have a visitor center where they offer customized experiences such as a Virtual tour to Mars, escape room, exhibitions, films about the universe, climbing wall, and paper rocket workshop [4].

As of 2024, ASE collaborates with 60 different national and international schools, countries, and agencies. Additionally, they cooperate with 18 different institutions and organizations. [5].

TL|GALK

1.5 Rationale Behind Assignment

ASE’s focus is to teach students about space and space technology in a practical manner. For their education they use CanSat and CubeSat development as an educational platform. The assignment we received from ASE was to redesign a CanSat kit they developed in 2018. A requirement for this new kit is that it should be modular so it can be used in both CanSat education, but also fit within a CubeSat frame for other educational purposes.

This CanSat kit is used to introduce the Science, Technology, Engineering and Mathematics (STEM)-field to students ranging from 13 year old up to university students. This CanSat kit is also used in a competition for students in the age gap of 14-19 years old, where they compete in teams of 3-6 students. This kit is used as a learning platform where the students have the components and base software already developed for them, but it is up to the students to use their creativity to build their own unique CanSat.

Todays kit uses a Teensy 3.5 developed by PJRC, which is an microcontroller that uses a 90nm silicon transistors [6]. Due to the global chip shortage most semiconductor manu-

facturers decided to focus on 45nm and smaller, therefore PJRC has upgraded the Teensy to a newer platform, and the last batch of Teensy 3.5 was produced 15. August 2023 . PJRC recommends moving current projects to their newer and faster Teensy 4.1 [6]. The new CanSat kit will therefore use Teensy 4.1 as the microcontroller for the new CanSat Kit

By upgrading the microcontroller, a new circuit board needs to be developed. A requirement set by ASE was for the new circuit board, is there has to be breakout capabilities for I²c, SPI and UART protocols as well 5V and 3V3 power, to enable this circuit board to be used within a CubeSat system. ASE also want us to develop a ground station software with a GUI for the students to view real time telemetry data, and see the current status for the CanSat and CubeSat.

2 Work Method

For this project, we opted for an AGILE project method. The main reason for this is that an AGILE work philosophy allows for modifications to be made over multiple iterations, rather than persevering with sub-optimal decisions. Still, it is unwise to be indifferent regarding what AGILE methodology to use as it could greatly impact a project's success, depending on the nature of the project and the composition of project team members.

Being a cross-disciplinary ensemble, our group was faced with a dilemma: namely the great variance between members' preference for how they like to work. In general, software engineers prefer to work incrementally, this was also true for our group. Mechanical engineers, on the other hand, typically prefer to work evolutionary. Electrical engineers' style of work is usually a hybrid between the two. As a result, finding a common tool for task management that suited all team members was a challenge. After multiple discussions and trialing, we have (as things stand) landed on the following system:

- Software engineers and electrical engineers log tasks and user stories in Azure DevOps; a suite of tools provided by Microsoft [Subsection 2.1]
- The mechanical engineer on our team is using digital "post-it notes" with connecting lines between them for his task management [Subsection 2.3]

An obvious concern one might have with this solution is if not restricting the team members to the same tool could lead to miscommunication and disarray. However, measures have been put in place to reduce the risk of this happening. For one, the mechanical engineer is setting up his method to closely resemble the rest of the group's system in Azure DevOps. Second, all team members are restricted to the same length, start, and end date for an iteration. Lastly, all members are obliged to partake in all three of our daily stand ups, along with sprint reviews and retrospectives. Although demanding a more conscious effort concerning communication and synchronization, we now have a working system that shows appropriate sensitivity to each member's personality and work ethic.

2.1 Azure DevOps

Developed by Microsoft, Azure DevOps is a suite of services useful for project management and development. AD supports numerous services, but as of now, only two will be used: Azure Boards, and Azure Repos.

2.1.1 Azure Boards

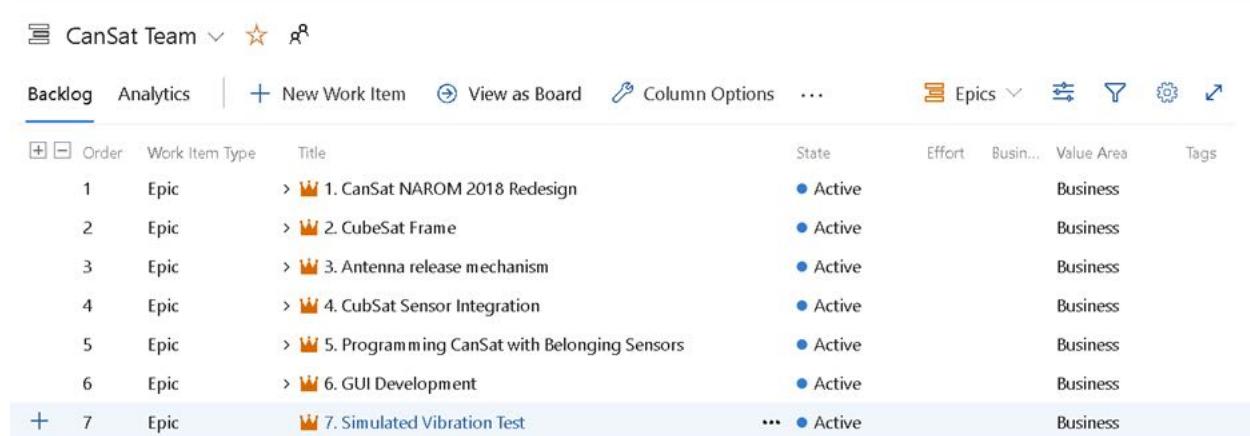
As implied by its name, Azure Boards is a digital task board for project management. One of Azure Boards' key features is its flexibility – being able to configure its setup to support practically any AGILE methodology. This outstanding freedom for customization allows us to create a project environment that is tailored to our needs.

The jargon and setup in Azure Boards does draw inspiration from AD (another AGILE project management method), but with some diverging definitions and implementation for the same terminology. Our employed setup resembles a folder-like structure:

- Epics
 - Features
 - * User stories
 - Tasks

2.1.2 Epics

Epics represent the highest level in our project setup. They are often regarded as the overall objectives for a project's end-product[7]. Our project has seven epics – each named after the seven objectives set by ASE in their assignment description. Epics can last over multiple iterations/sprints and can support multiple features.



A screenshot of the Azure DevOps Backlog board for the 'CanSat Team' project. The board shows a list of seven Epics, each with a title and a list of sub-items. The columns are: Order, Work Item Type, Title, State, Effort, Business Area, and Tags. All epics are currently active and belong to the 'Business' category.

Order	Work Item Type	Title	State	Effort	Business Area	Tags
1	Epic	> 1. CanSat NAROM 2018 Redesign	● Active		Business	
2	Epic	> 2. CubeSat Frame	● Active		Business	
3	Epic	> 3. Antenna release mechanism	● Active		Business	
4	Epic	> 4. CubSat Sensor Integration	● Active		Business	
5	Epic	> 5. Programming CanSat with Belonging Sensors	● Active		Business	
6	Epic	> 6. GUI Development	● Active		Business	
7	Epic	> 7. Simulated Vibration Test	... ● Active		Business	

Figure 3: Epics in Azure DevOps

2.1.3 Features

Features are a sub-category of Epics that represent the qualities of our system of interest (SoI). A feature can support multiple user stories [7].

Figure 4: Features in Azure DevOps

5	Epic	✓ 🏆 5. Programming CanSat with Belonging Sensors	● Active	Business
	Feature	> 🎮 GY-91 implemented	● Active	Business
	Feature	> 📡 RFM-96 implemented	● Active	Business
	Feature	> 📸 Camera Implemented	● Active	Business
	Feature	> 🛍 GPS implemented	● Active	Business
	Feature	> 🎵 Speaker implemented	● Active	Business
6	Epic	✓ 🏆 6. GUI Development	● Active	Business
	Feature	> 📡 Real-time data transfer	● Active	Business
	Feature	> 🖥 Intuitive layout	● Active	Business
	Feature	> 🌐 Plug-and-play usability	● Active	Business
	Feature	> 📈 Real-time data plotting	● Active	Business
+	Feature	🖨 Option to save data to file	... ● Active	Business

2.1.4 User stories

User stories represent the penultimate level in this hierarchical structure. Contrary to epics and features, user stories are generally intended to be completed within a single sprint.

For this project, we followed the standard naming convention for user stories:

“AS <role (often end-user)>, I WANT <objective> TO/FOR <motivation>”

[8]

The reason for opting for this wording is that it forces every developer on our team to evaluate the value this user story is bringing to the end user. Thus, if a user story cannot be written in the following manner, it can indicate that the task should not be a priority since its not bringing value to our consumers.

Figure 5: User story in Azure DevOps

170 As a student, I want the documentation to be complete, concise and clear, to be able to give the best feedback/opinion.

Nathanael Nega Getaneh 0 comments Add tag

Save & Close

State	Active	Area	CanSat	Updated by
Reason	Implementation s...	Iteration	CanSat\Sprint 3	D

Description

Context: For the 1st submission. Recipients are the external grader, internal and external supervisor and internal grader.

Constraint: Be completed by February 1st. at the very latest.

Acceptance Criteria

- The report and documentation MUST be equally understood by all team members.
- The report and documentation MUST be understood by at least one non-team member.
- All sources used MUST be evaluated and deemed proper by everyone in the group. If not, an attempt should be made to find better sources.
- The report and documentation MUST be correctly cited (IEEE) and referenced throughout.
- Abbreviations MUST be explained in the folder "Abbreviation list".

Classification

Deployment

To track r work item deploym... in your pi more abc reporting

Development

+ Add link

Link an A request c your devi branc tc

Related Work

+ Add link

Alongside a **description**, Azure DevOps allows users to set **acceptance criteria** for a user story. This feature is critical as it sets a clear benchmark for the team's perception of "done" [8]. For this reason, it is unacceptable for acceptance criteria to be vague. Below is an excerpt of acceptance criteria that have been written and used by us:

- All sources used MUST be evaluated and deemed proper by everyone in the group. If not, an attempt should be made to find better sources.
- The report and documentation MUST be correctly cited (IEEE) and referenced throughout.
- All abbreviations MUST be explained in the folder "Abbreviation list" and referenced in the main text when used.

2.1.5 Tasks

From a user story, an objective can be broken down into clear and concise tasks. A task must naturally be scoped down to the level where it can be completed within the length of an iteration.

In Azure DevOps, each task can be given a description. Our group has decided to utilize this feature by including the following:

- Description of the task itself.
- If relevant, link to any sources for context.
- Teams link to the documentation relevant to this task. This has to be added before a task can be closed because work that is not documented, is work not done. It is also a requirement that the documentation is read by and approved another team member (preferably one of a different discipline) before the task can be closed.

Figure 6: Task in Azure DevOps

The screenshot shows a task card in Azure DevOps. At the top, it says "TASK 149" and the title "149 Setup version control in Azure repos (1st iteration)". Below that, it shows the assignee "Nathanael Nega Getaneh" and "0 comments". The status is "Closed" (green dot), "Reason" is "Completed", "Area" is "CanSat", and "Iteration" is "CanSat\Sprint 2".

Description: Setup git version control in Azure repos. Decide how to separate and integrate GUI and sensor development.
[CRAFT VERSION CONTROL DOCUMENTATION 01.docx](#)

Planning: Priority 2, Activity.

Discussion: A note says "Add a comment. Use # to link a work item, ! to link a pull request, or @ to mention a person." There is a placeholder text area below.

Effort (Hours): Original Estimate 3, Remaining 0, Completed 3.

Implementation:

As seen in Figure 6, tasks include an original estimate for the hours expected to complete the task. This column is filled in before a task is set to active. Hours are then updated in the "Remaining" column, and the total number of hours actually used is entered into the "Completed" column/tab once a task has been closed. By filling in these "time-slots", it becomes easier for the product owner and sprint master to better make judgements regarding progress and if someone is having too much or too little work that sprint.

In Azure DevOps, tasks are assigned to one person only. If multiple people are working on the same task, then that task can either be duplicated for each team member, or simply specified in the task's description who else is also working on this task.

2.2 Secure Documentation and Development

As emphasised by our external supervisor in our first meeting with him, thorough documentation is an important demand from ASE (our primary stakeholder). It is equally important within our group to have a consistent and reliable framework/system for documenting and developing.

2.2.1 Azure Repos (Version control)

Consistent and detailed version control is essential for reliable documentation and collaboration. Good version control can also function as a safety net in case of human or technical error. Azure Repos is a service included in the Azure DevOps subscription and functions as a host for git repositories. For a detailed explanation and showcasing how we have implemented Azure Repos for version controlling code, please see subsection 5.13

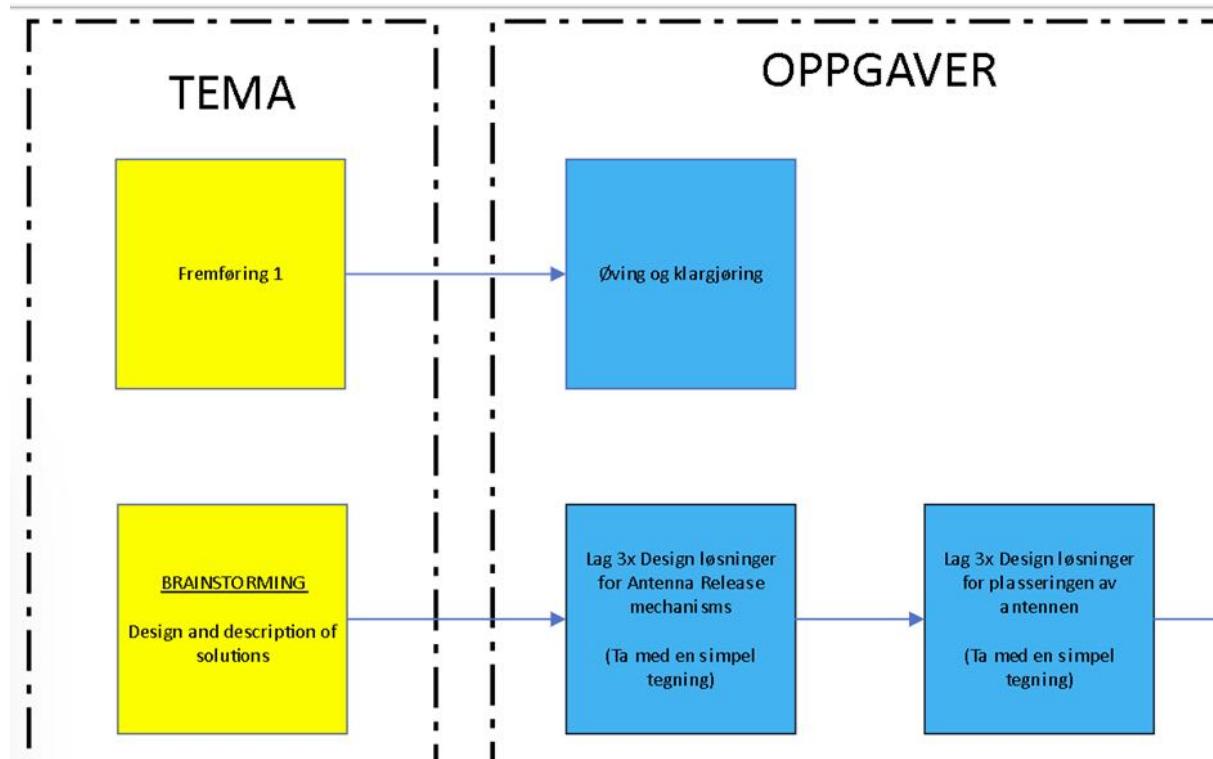
2.2.2 Regular Backups

All our documentation has currently been stored in Microsoft Teams. To add a layer of security, the documentation manager [reference to relevant subsection will be added at a later stage] is responsible for taking weekly backups of the documentation.

2.3 Visio Method

The yellow post-it notes represent the general themes that the tasks (i.e., blue post-it notes) fall under. The yellow post-it notes are analogous to user stories in Azure DevOps and the blue notes are analogous to the tasks in Azure DevOps. This close resemblance mitigates the risk of a potential mismatch in communication and work done between the mechanical student and the rest of the group. The directional arrows connecting the blue boxes together signals the order in which the tasks will (and sometimes must) be completed. Once a task has been completed, a diagonal line is draw across the vertices of the blue box. Initially, this system was originally made using physical post-it notes that were hung up to the wall. This was problematic due to poor readability, and accessibility (could only be viewed from the project room), and maintainability over time. It was, thus, decided to migrate this system onto a digital Visio file stored in Teams. Going digital also has the benefit of reducing the gap between users of Visio and those of Azure DevOps.

Figure 7: Project management method in Visio | STB



2.4 Length for each sprint

After thorough discussion amongst each other, we have decided (as things stand) to opt for a sprint duration of one week (Monday – Friday). The justification for this is that one week is deemed enough time to conduct and complete important sub-goals and tasks, while also providing a healthy degree of stress to reach deadlines. However, our group is open to adjusting the sprint length, if necessary.

2.5 Weekly schedule

During this project, our core hours have been from 9 AM to 3 PM (unless a member has a valid excuse). Core hours are a staple in many office spaces and refer to mandatory working hours. There is some freedom in wanting to work from home (if it is not too frequent an occurrence). However, home office workers still must work inside those core hours and must partake in all meetings and plenary sessions.

2.5.1 Ensuring good communication

To ensure sufficient communication and synchronization amongst all team members, each workday consists of three short obligatory meeting/stand-ups.

We are also currently experimenting with one-hour lunches every other Wednesday that is intended for casual discussion about the project in an enjoyable setting. We experienced early on that the three stand-ups, although useful, can be too short and "intense" for clarifying potential misunderstandings and airing out new ideas. Thus, we believe the one hour lunches will be a useful addition.

2.6 Timetable

(NB: This plan does not include short coffee breaks and such)

MONDAY– THURSDAY

09:00 AM: Morning stand-up (No more than 15 minutes)

09:15 AM – 11.30 AM: Project work

11:30 AM – 12:00 PM: Lunch¹

12:00 PM: Midday stand-up (No more than 5 minutes)

12:15 PM – 02:45 PM: Project work

(On MONDAYS, meeting with internal supervisor: 09:00 AM – 10:00 AM)

¹Every other Wednesday, lunch lasts one hour (11:30 AM - 12:30 PM)

02:45 PM – 03:00 PM: Wrap-up

FRIDAY

09:00 AM – 10:00 AM: Morning stand-up and project work

10:15 AM - 10:45 AM: Meeting with external supervisor.

10:45 AM - 11:30 AM: Project work

11:30 AM - 12:00 PM: Lunch

12:00 PM – 02:00 PM: Sprint Review + retrospective

02:00 PM – 03:00 PM: Sprint planning.

2.6.1 Terminology Explained

Morning stand-up: Each member swiftly informs the team the following:

- Status on yesterday's work.
- The focus and goal for today.
- If they are blocked from doing certain work

All meeting participants are obligated to stand to encourage fast progression of the meeting. At the end of each stand-up, the one leading the meeting (usually the group leader or sprint master), will ask if there is anyone on the team that needs help with finishing a task or troubleshooting.

Midday stand-up: Each member swiftly informs the team the following:

- Status on today's work
- Report unforeseen obstacles (if there are any)
- If they need help or more work (The one leading the meeting will ensure to ask the developer this if the developer himself has not done it).

This meeting is for identifying any potential bottlenecks before time has been inefficiently spent by the end of a workday. Everyone is also obliged to stand during this meeting.

Wrap-up: Session where the following three questions are asked:

- What went well today?
- What could have been better?
- How can we improve?

Project work: Includes both individual, group and plenary work.

Sprint Review + Retrospective: Meeting where the success of a sprint is evaluated [9]. Different from regular stand ups, this meeting focuses more on the progress of the project itself instead of the individual group member. Areas for improvement are highlighted and considered for the sprint thereafter.

2.6.2 Roles explained

The **product owner** is responsible for ensuring that the team is working towards a common goal that corresponds to the requirements that have been assigned to us[10]. To maintain a clear and accurate vision within the group. The product owner is also the main intermediary between our group and stakeholders.

The **documentation manager**'s main responsibility is managing the documentation that is on Teams. That involves seeing to it that the documentation is properly organized within our unanimously agreed structure. The documentation manager has the authority to point out mistakes made in that regard and demand the team member correct such.

The **HSE lead** has an higher focus on the health, safety and environmental impact that could be caused from project work. He is also primarily responsible with making and updating any risk analysis as the project progresses.

2.6.3 Sprint Master

Separate from the product owner, the sprint master is responsible for ensuring that everyone on the team is following our AGILE methodology for the entire duration of a sprint. This is heightened communication between the sprint master and the product owner to maximize efficient and highlight areas for improvement. Although not custom, we decided to let this be a **rotating role**; a new sprint master will be appointed for each iteration. An advantage for doing this, is that it evens out the workload amongst team members. It also encourages all team members to engage in our project management method.

2.7 Decision making

GALK|TL

The startup of a product development is a critical phase to have the right progression; to not focus on creating a solution to the issue at hand right away, and instead scope the problem with the eyes of an engineer.

Selecting the platforms and software for keeping track of group progress, files and communication was done during the first week and is an important step in making sure the group members are aware of all the common work method required for a multidisciplinary engineering team.

For the group to work efficiently, we are dependent on the members to keep up to date on all progress we are doing together, or each member are required to update others with

their knowledge and ideas underway. Though the idea is simple, the follow through is cumbersome to practise and the outcome will vary greatly on the understanding of the topic that each group member have acquired of the discussed material. Therefor are decisions where all members are to be affected, or are to follow a common work method should be done with all members to lessen the work after, where updating team members are required. The skill of the group members to accomplish an efficient communication grounds are a process where improvements come over time, as feedback about the topic in discussion will tell where lacking information for other discipline's.

3 Requirements

TL, NNG, STB|TL

3.1 Stakeholder Requirements

At the start of our project we made a list over our relevant stakeholders. These are all groups or parties that has an impact in our product. This was done in an attempt to get an overview over all the relevant actors we have to satisfy. These stakeholder are:

1. ASE
 - Owners and maintainers of C.R.A.F.T-Sat
2. Students
 - Users of C.R.A.F.T-Sat
 - Upper middle school-University level
3. Teachers and Educators
 - Supervise students
 - Educate students with the use of C.R.A.F.T-Sat
4. European Space Agency (ESA)
 - CanSat competition requirements
5. Norwegian Government
 - General laws and regulations
 - Nasjonal kommunikasjonsmyndighet (NKOM)
 - Civil Aviation Authority of Norway (Luftfartstilsynet)
6. European Union
 - General Data Protection Regulation (GDPR)

From our stakeholders we are able to set some stakeholder requirements that C.R.A.F.T-Sat must oblige. In an iterative dialogue with our main stakeholder ASE, we were able to set requirements our stakeholders need our product to satisfy.

1. All subsystems must be well documented this includes, but are not limited to:
 - (a) printed circuit board (PCB) Design
 - (b) CubeSat Frame
 - (c) Antenna Release mechanism
 - (d) GUI Source code
 - (e) C.R.A.F.T-Sat source code

-
- (f) Vibration test procedure
 - (g) NB! If acceptable documentation for subsystems and components already exist, a reference to this documentation is sufficient.
2. Easy to reproduce by other parties
 - (a) Affordable
 - (b) Most likely are in stock in the future
 - (c) Available from well known websites
 - (d) Well documented parts
 3. Easy to operate, assemble and disassemble
 - (a) CubeSat frame must be easily assembled/disassembled by the end user with the aid of an instruction manual by the end users.
 - (b) Circuit Board and components must be easy to solder by hand by the end users.
 - (c) Circuit board and expansion card must be easily installed within the CubeSat Frame by the end user.
 - (d) The software for the C.R.A.F.T-Sat must be easy to read and use for the end users.
 4. Safe to use by all susceptible parties.
 5. Software used for producing electronical and software components for C.R.A.F.T-Sat must be free and open source.
 6. The entire system should be made with longevity in mind [rule of thumb: "more than 10 years"]. This applies to, but are not limited to
 - (a) CubeSat Frame
 - i. 3D printing materials
 - ii. Mounting Mechanism
 - (b) PCB and Sensors
 - i. Tools availability
 - ii. PCB materials
 - iii. Components
 - (c) Software implementations
 - i. Packages and libraries
 - ii. language availability

While most of these stakeholder requirements come from ASE, our other stakeholders also has requirements our system must answer to. The **Norwegian Government** set the general laws and regulations for our system when it is under production and operation, the authorities responsible are NKOM, which are responsible for radio communication laws. They are the authority that decides the laws when it comes to radio communication from the C.R.A.F.T-Sat to the Ground Station. Our system must also answer to all related laws relative to our project.

Our stakeholders the **teachers** are the secondary end user for our system. They are responsible to teach our end users, the **students**, using our system. The teachers requires our system to be, Well documented subsystems, easy to reproduce, and safe to use by all susceptible parties. Since C.R.A.F.T-Sat is used by students for education, we can assume they do have little to no experience in many of the steps they need to go through when producing a CubeSat or a CanSat. They will require our system to be well documented, easy to reproduce and easy to assemble and disassemble.

TL, STB, NNG|TL

3.2 Functional requirements

Functional requirements are features that our system must do. These requirements are set from our stakeholder requirements, split into features our system must have to be a minimum viable product. Every functionality included in our system that are not a part of our functional requirements list, are extra features not needed, but accepted and approved by our main stakeholder ASE.

1. C.R.A.F.T-Sat frame must be able to be 3D printed.
2. C.R.A.F.T-Sat antenna must be able to be released at operation.
3. C.R.A.F.T-Sat antenna release mechanism must be re-usable, with allowance for disposable parts (e.g. Burn-wire).
4. C.R.A.F.T-Sat must be able to transfer data in soft real-time.
5. The CanSat shield must reuse sensors from CanSat NAROM 2018.
6. The CanSat shield must have breakout points for specific communication protocols and power 5
7. The CanSat shield must have the ability to connect to expansion circuit boards.
8. The CanSat shield and expansion board must fit within the CubeSat-Frame.
9. The ground station must have the ability to receive data from the C.R.A.F.T-Sat in soft real-time.
10. The ground station must have a GUI with the ability for students to view telemetry data in soft real-time.
11. The Ground Station must have the ability to save data to file.

12. C.R.A.F.T-Sat sensors must work at an altitude of approx. 26km (+- 5km).

13. Functional requirements not included in Minimum viable product includes:

- Micro controller must have the ability to save data locally to SD card
- Sensors and modules must be able to be hand soldered to the CanSat shield
- The microcontroller must be able to be programmed by end users by using the provided libraries

NB: All requirements mentioning real time have a max latency of 30 seconds.

TL, STB, NNG|TL

3.3 Design Requirements

By adhering to our stakeholder and functionally requirements we gain the ability to set design requirements our system must adhere to. The design requirements are meant to help us envision a solution when we are in the design phase. By having set requirements we limit ourselves to a solution that will be viable to our stakeholders. Some of these requirements are negotiated with ASE, but some design requirements are set by third parties, like the CubeSat design design specifications A.20. As with the functional requirements any additional design features included in our system are developed by us and accepted and approved by ASE, the only requirement our additional design do have is that they should not do so our set design requirements are not answered.

1. C.R.A.F.T-Sat must oblige to all relevant design requirements and regulations imposed by ESA (CanSat competition) A.1
2. CubeSat frame must fit within 1U test pod
3. The entire CRAFT-Sat system must not exceed 500 grams.
4. C.R.A.F.T-Sat frame must answer to relevant CubeSat design specifications A.20
5. The shield must have breakout points for SPI, UART, I²c and 3v3 & 5v power supply, with the ability to connect expansion circuit boards within the C.R.A.F.T – Sat.
6. GUI must be easily readable.
7. Software source code must be easy to read and well documented.

NNG|TL

3.4 Testing Requirements

If a requirement cannot be tested, then it is not a requirement. Thus, there has been careful deliberation on ways to verify whether C.R.A.F.T-Sat is fulfilling the requirements listed in subsections 3.1, 3.2 and 3.3. From our risk analysis [see Subsection 4], there is

a risk that some requirements will be extremely challenging to test. For instance, since we cannot test launch our C.R.A.F.T-Sat into the stratosphere, it makes it difficult to verify if all sensors operate with a 26km range. The suggested solutions for testing have consequently had to be creative in some respects. Even requirements that are presumably “simple” to test, can prove later to be difficult due to unforeseen circumstances. To rectify this, we have made a diligent attempt at simplifying our suggestions for testing, and when possible, provided alternative testing methods. This can be seen in Tables 1, 2 and 3.

3.4.1 26 km range

As already mentioned, lack of launch opportunities makes it harder for us to verify a range of approx. 26 km. Testing, however, does not always have to include prototyping; it can also involve the analysis of data. Thus, one way to test requirements linked to the range of our sensors is to **read the data sheets to all the sensors that will be used**, more concretely the range for each sensor. After all, the ideal testing that we are unable to do, has most likely already been conducted by the manufacturers of these sensors.

Another method for testing such a requirement that we have considered is having C.R.A.F.T-Sat active as far as possible from its Ground Station and observing if data is being transferred. The distance between these two subsystems could be strictly vertical (e.g., separated by multiple floors in a building), although the maximum distance would severely limited. Lifting the C.R.A.F.T-Sat with a drone has also been considered).

With horizontal distance, however, distances can be greater. The downside is that since C.R.A.F.T-Sat is intended to be vertically distant from the Ground Station, the placement of C.R.A.F.T-Sat’s antennas could potentially be sub-optimal for horizontal distances, giving it an inaccurate perception of its sensors’ range. Although larger distances are possible horizontally, the number of 26 km obstacle-free locations nearby is scarce. One location that is technically viable is a river passing from Gvarv (Norwegian village), to Skien (Norwegian city). Figure 8 shows the routes’ distance of approx. 26 km.

Still, testing a prototype in such manner can still prove useful as evidence to deductively reason whether this objective and its belonging requirements have been achieved.

3.4.2 Ease of Use and Documentation

Words such as “easy” and “well/good” can be undesirable in a requirement list due to them being subjective in nature. Given our system context, however, such terms are essential when describing the qualities of our system’s ease of use and documentation. Our end users consist of young students that have minimal (perhaps zero) experience with CanSat and CubeSat. However, the system’s simplicity must not compromise with the requirements that our system buyer has set regarding its functionality. Thus, the evaluation of “ease of use” and “well documented” shall primarily be from the view of our end user, and ASE.

To test requirements linked to this objective, we shall keep frequent communication with our external supervisor regarding their opinion on work, and eventually receive their final

approval on our system design. We will also create a case study targeted at upper middle school students, showcasing our system and letting them explore C.R.A.F.T-Sat and C.R.A.F.T-GUI without any guidance (only tool being the instruction manual that we will produce, along with other documentation). The students will then be asked to state their opinion regarding the system and its documentation, and whether it was simple to use. Ease of use regarding assembly, disassembly, and operation of our system can also be tested in a similar fashion.

3.4.3 Law and Regulations

For the main circuitry to be applicable for both the C.R.A.F.T-Sat and the updated CanSat NAROM 2018, it must comply with all design requirements imposed by ESA (CanSat competition). However, as clarified by our external supervisor, laws involving the use of parachutes should **not** be a requirement we set for ourselves. Thus, testing for such will not be conducted by us. Consequently, that also includes testing of laws regarding descent time and rate.

The testing of the weight requirement range of 300-350 grams (including parachute) for CanSat will be quite simple: Since we have been sent a replica of a fully configured CanSat NAROM 2018 by ASE, we can place it on a digital scale and note down the given value. Its old circuitry is then removed from the frame, and the weight of the CanSat is measured again. The difference in weight is then calculated to give an estimate of the technical budget (regarding weight) that the electronical engineering department must work with. To test the legality of our updated design, the updated circuitry can simply be placed inside the CanSat NAROM 2018 frame, and the entire system's weight is remeasured. If the measured value lies within 300-350 grams, then that verifies that this requirement has been met.

3.4.4 Accessibility and Longevity

Verifying a requirement can be much simpler when it is quantifiable rather than qualitative. For instance, requirement involving certain product standards and/or dimensions for usability (e.g. requirement requiring a minimum 0805 component size for hand soldered components), can simply be tested by **referring to the data sheets** for these components. The same principles apply for requirements regarding the **fitting and size** of certain components.

System longevity is more cumbersome to test, since it involves making an educated prediction on an uncertain future. Fortunately, the requirement involving a component/subsystem being supported (not necessarily last) for 10 years is more of a “rule of thumb” than a firm demand. Still, to verify this requirement, and other ones surrounding accessibility, certain criteria will be held against components/programs/services when considering their viability in our system design:

- Is this product available for download or purchase from reputable firms that support fast shipping to Norway?
- Is the product open source?

-
- Does the product comply with all laws and regulations in Norway?
 - How long has the product in question existed in the market? Does it have any viable competitors? If so, how is their prognosis looking with respect to long-term existence?
 - Are there any large and stable corporations that have invested in this product?
 - Is this component easy to use from the perspective of our end user? Does the end user or ASE have any prior experience with this product?
 - Are the potential complications/implications the product might have on the ease of use for the end-user outweighed by the advantages that it brings? If unsure, make a Pugh matrix or refer with our client.

3.4.5 Functional Requirement Testing

NNG|

Functional Requirement	Testing Method	Alternative Method
1	Create CAD of C.R.A.F.T-Sat frame. 3D print said model to verify that it can be 3D printed.	
2	Synthesise a technical budget which considers the physical size limitations and occupations within C.R.A.F.T-Sat's 1U CubeSat frame.	
3	Release antenna at least 20 times. After each release, analyse the antenna to see if it is a capable condition.	Conduct a FEM analysis.
4	Note the time a data packet is sent from a C.R.A.F.T-Sat with the time that said packet was received by the ground station. If this time difference is 30 seconds or less for at least 95 percent of all the packets, then this requirement is considered fulfilled.	
5	Mount reused sensors to CanSat shield.	
6	Use a voltmeter	
7	Synthesise a technical budget which considers the physical size limitations and occupations within C.R.A.F.T-Sat's 1U CubeSat frame.	
8	Synthesise a technical budget which considers the physical size limitations and occupations within C.R.A.F.T-Sat's 1U CubeSat frame.	
9 -10	Note the time a data packet is sent from a C.R.A.F.T-Sat with the time that said packet was received by the ground station. If this time difference is 30 seconds or less for at least 95 percent of all the packets, then this requirement is considered fulfilled.	
11	Navigate the file structure of the computer in which the data log was aimed to save locally to and verify whether the file exists and have data that is exact to that showed on log on our GUI	
12	3.4.1	
13a	Connect SD card to C.R.A.F.T-Sat's Teensy 4.1. Then activate the C.R.A.F.T-Sat's sensors to measure and store sensor data. Then remove said SD card and insert it into the Ground Stations SD card slot and see if its GUI can convert the bit-stream into understandable data.	
13b	Hand-solder all sensors and modules to the CanSat shield.	
13c	Code and test a default configuration for a C.R.A.F.T-Sat that was programmed using the same libraries that the end users will employ.	

Table 1: Functional Requirements Test

3.4.6 Design Requirement Testing

Design Requirement	Testing Method	Alternative Method
1	3.4.3	
2	Place C.R.A.F.T-Sat frame inside 1U test pod and see if it fits.	Measure dimensions of C.R.A.F.T-Sat frame and compare standardized specifications online from CubeSat.org.
3	Place C.R.A.F.T-Sat on a high accuracy digital scale.	
4	Compare C.R.A.F.T-Sat frame to design specifications.	
5	Use a voltmeter to verify the voltage from C.R.A.F.T-Sat's power supply.	
6	3.4.2	
7	3.4.2	

Table 2: Design Requirements Test

3.4.7 Stakeholder Requirement Testing

Stakeholder Requirement	Testing Method	Alternative Method
1	3.4.2	
2a-c	3.4.4	
2d	3.4.2	
3	3.4.2	
4	Evaluate system based on our risk assessment [Subsection 4].	Evaluate system based on relevant laws, regulations and safety standards. Refer with our stakeholders.
5	3.4.4	
6	3.4.4	

Table 3: Stakeholder Requirements Test



Figure 8: Gvarv to Skien (Google Maps)

4 Risk Analysis

RE|NNG

Risk analysis is an essential process that helps predict and prepare for possible difficulties during the project. Risk analysis is seen as a strategic approach that not only recognizes risks, but also assesses their likelihood and potential impact, and finds strategies to mitigate them. The methodology we follow is iterative, which means that the process starts with systematically identifying risks, analyzing potential impacts on the system, and preparing strategies for managing risks. This process is not a single event but a part of ongoing project management and is often updated with new discoveries and insights[11].

4.1 Risk Assessment

Each identified risk will be assessed based on its likelihood of occurrence (**probability**) and its potential **impact** on the project. The probability will be categorized as High Probability (HP), Medium Probability (MP), or Low Probability (LP). The impact will be categorized as High Impact (HI), Medium Impact (MI), and Low Impact (LI). The risk score will be determined by multiplying the probability and impact ratings, as shown in table (4). A higher score (from green to red) indicates a higher risk.

Risk Matrix		Impact		
		High	Medium	Low
Probability	High	HP and HI	HP and MI	HP and LI
	Medium	MP and HI	MP and MI	MP and LI
	Low	LP and HI	LP and MI	LP and LI

Table 4: Risk Assessment Matrix.

The determination of the risk potential level is based on three distinct factors. These factors include the frequency at which a common risk may occur, the timing of occurrence (early, middle, or late phase), and the complexity involved in mitigating the impact of such a risk. A table (5), (6), and (7) has been created to provide a clearer understanding of these factors.

Impact Level	Complexity to Mitigate	Description
High (H)	High complexity	Requires significant resources to mitigate
Medium (M)	Moderate complexity	Manageable with moderate resources
Low (L)	Low complexity	Easily managed with minimal resources

Table 5: The impact is Influenced by The Complexity to Mitigate The Risk.

Probability Level	Frequency of Occurrence
High (H)	More than twice
Medium (M)	Twice
Low (L)	less than twice

Table 6: Probability Boundaries.

Probability Level	Timing of Occurrence	Interval
High (H)	Late Phase	April - May
Medium (M)	Middle Phase	March
Low (L)	Early Phase	January - February

Table 7: Probability Boundaries Based on Timing of Occurrence.

4.1.1 Technical Risk Analysis: Electronics Field

4.1.2 Risk Assessment: Early Phase

In the field of electronics, numerous risks are often encountered throughout project development. Table (8) provides a summary of the most predicted risks and their descriptions. Tables (9), (10), and (11) detail the likelihood of these risks occurring and their impact on the early, middle, and late phase of the project, respectively.

Risks	Description
Missed External Supervisor Contact.	Independent decision-making due to external supervisor absence because of illness.
Unvalidated PCB Functionality.	Inability to validate the main PCB's functionality due the lack of communication with ASE.
Unverified Sensors chosen for Expansion Card.	Sensors chosen for the expansion card might not be of interest to Andøya Space Education, leading to wasting time and resources and increasing cost.
Unconfirmed Integrating of Solar Panel in C.R.F.T-Sat	Uncertain need for implementing solar panel, potentially wasting tame and resources.

Risks	Description
Changing in Design Without Approval from Stakeholders.	In case of lack of communication with the external supervisor, we could decide to make design changes without the Stakeholder's approval.
Unverified Battery Type and Battery Size.	The chosen Lithium polymer (LiPo) battery and battery size might not be the preferred type by ASE.
Incomplete Component Deliveries.	Potential Delays due to incomplete deliveries from the supplier.
Unverified Connection Methods between Main and Expansion Card	Relying solely on internal expertise for chosen connection methods without stakeholder confirmation might lead to compatibility issues.
Budget Exceedance.	Exceeding the allocated budget of 10,000 NOK.

Table 8: Risks and Their Description: Electronics Team

Risks	Probability	Impact	Risk Score
Missed External Supervisor Contact	LP	MP	LP and MI
Unvalidated PCB Functionality	LP	HI	LP and HI
Unverified Sensors chosen for Expansion Card	LP	HI	LP and HI
Unconfirmed Integrating of Solar Panel in C.R.F.T-Sat	LP	MI	LP and MI
Changing in Design without Approval from Stakeholders	LP	LI	LP and LI
Unverified Battery Type and Battery Size	LP	LI	LP and LI
Incomplete Component Deliveries	LP	LI	LP and LI
Unverified Connection Methods between Main and Expansion Card	LP	LI	LP and LI
Budget Exceedance	LP	MI	LP and MI

Table 9: Case Matrix for Electronics Team: Early Phase

4.1.3 Risk Assessment: Middle Phase

Risks	Probability	Impact	Risk Score
Missed External Supervisor Contact	MP	HI	MP and HI
Unvalidated PCB Functionality	MP	MI	MP and MI
Unverified Sensors chosen for Expansion Card	MP	HI	MP and HI
Unconfirmed Integrating of Solar Panel in C.R.F.T-Sat	MP	MI	MP and MI
Changing in Design without Improvement from Stakeholders	LP	MI	LP and MI
Unverified Battery Type and Battery Size	LP	LI	LP and LI
Incomplete Component Deliveries	MP	LI	MP and LI
Unverified Connection Methods between Main and Expansion Card	LP	MI	LP and MI
Budget Exceedance	MP	MI	MP and MI

Table 10: Case Matrix for Electronics Team: Middle Phase

4.1.4 Risk Assessment: Late Phase

Risks	Probability	Impact	Risk Score
Missed External Supervisor Contact	HP	HI	HP and HI
Unvalidated PCB Functionality	HP	MI	HP and MI
Unverified Sensors chosen for Expansion Card	HP	HI	HP and HI
Unconfirmed Integrating of Solar Panel in C.R.F.T-Sat	HP	MI	HP and MI
Changing in Design without Improvement from Stakeholders	HP	LI	HP and LI
Unverified Battery Type and Battery Size	HP	LI	HP and LI
Incomplete Component Deliveries	MP	HI	MP and HI
Unverified Connection Methods between Main and Expansion Card	HP	MI	MP and HI

Risks	Probability	Impact	Risk Score
Budget Exceedance	MP	MI	MP and MI

Table 11: Case Matrix for Electronics Team: Late Phase

4.1.5 Mitigation Strategies for Risks in Electronics Field

The mitigation strategies are crucial and must be carefully evaluated to ensure successful risk mitigation. Table (12) addresses the predicted risks and the strategies implemented by the electronics team to minimize their effects throughout this project.

Risks	Mitigation Strategies
Missed External Supervisor Contact	Attempt to re-establish communication with ASE as soon as possible. Document all decisions made independently and prepare justifications based on research, experiences, and Team member's competence.
Unvalidated PCB Functionality	Document the main PCB's design and testing procedures thoroughly. Prepare for potential redesign based on future communication with Andøya Space Education.
Unverified Sensors chosen for Expansion Card	Research CubeSat sensors typically used at the 26 km altitude range. Select sensors based on their relevance to educational purposes and alignment with common CubeSat applications at this altitude. Prioritize communication with ASE to confirm sensor selection as soon as possible.
Unconfirmed Integrating of Solar Panel in C.R.F.T-Sat	Making thorough research about implementing solar panels in CanSat and C.R.A.F.T-Sat, focusing on availability, efficiency, and cost-effectivity.

Risks: Continuous	Mitigation Strategies: Continuous
Changing in Design without Approval from Stakeholders	Making thorough documentation of each change we have made without approval from our stakeholders.
Unverified Battery Type and Battery Size	Research common battery types used in CanSat and CubeSat with similar specifications. Select a battery with a standard form factor and voltage output to allow for potential replacements if necessary.
Incomplete Component Deliveries	Identify alternative suppliers for critical components. Order components with sufficient lead time to account for potential delays. Maintain a buffer of essential components to mitigate delays. Prioritize local suppliers above online-stocks.
Unverified Connection Methods between Main and Expansion Card	Relying solely on internal expertise for chosen connection methods without stakeholder confirmation might lead to compatibility issues
Unverified Connection Methods between Main and Expansion Card	Research standard connection methods commonly used in CubeSats. Select connection methods with broad compatibility across various electronic components. If possible, design the PCB to accommodate alternative connection options based on potential future requirements.
Budget Exceedance	Track project expenses precisely. Prioritize essential components and functionalities within the budget constraints. Explore cost-saving alternatives for components where feasible. Clearly communicate any potential budget shortfalls to Andøya Space Education to explore possible solutions.

Table 12: Mitigation Strategies executed by Electronics Team

4.1.6 Effectiveness of Mitigation Strategies

Evaluation of Mitigation Strategies

1. Re-establishing Supervisor Contact

- **Effectiveness:** The strategy to document decisions made by own and provide a detailed justifications in the absence of our external supervisors was effective. It ensured continuity in project progress and enhanced the team's decision-making capabilities. Effort to re-establish communication with the external supervisor have been succeeded. The completed project was showcased, and the stakeholder was pleased with the results.

2. Validation of PCB Functionality

- **Effectiveness:** A detailed design of PCB-design and test procedures has allowed the electronics team to quickly realize every change that must be executed after each iteration. This strategy has minimized downtime and facilitated a more robust design process.

3. Sensor Selection for Expansion Card

- **Effectiveness:** The focused research on sensors based on relativity to C.R.A.F.T-Sat, applicability at high altitude with harsh cold temperature, and relevance to educational purposes, has led to the identification of the most precise sensors that private customers can purchase, which are also lightweight, compact, and cost-effective.

4. Solar Panel Integration:

- **Effectiveness:** The comprehensive analysis regarding the integration of solar panels into C.R.A.F.T-Sat and CanSat and designing a Do It Yourself (DIY) project have provided valuable insights into their achievability and utility.

5. Handling Design Changes

- **Effectiveness:** Each modification in the design has been thoroughly analyzed and documented. For instance, the rationale for decreasing the number of dipole antennas from two to one has been comprehensively documented. This adjustment has allowed for additional space for the implementation of camera and solar panels, while also lowering the expenses and complexity associated with the antenna design.

6. Battery Selection

- **Effectiveness:** Investigative studies on various battery types have provided the team with the necessary insights to select the optimal battery, guaranteeing both operational reliability and compliance with technical specifications.

7. Managing Component Deliveries

- **Effectiveness:** The strategy to utilize different suppliers to purchase the required components has reduced the risk and impact of delivery delays. How-

ever, we encountered a scenario where the GY-81 sensor could only be purchased by **Amazon**. The component did not arrive as quickly as anticipated, but we have sufficient sensors to proceed with the tests.

8. Connection Methods Verification

- **Effectiveness:** A detailed research of most effective connection methods that easily connects the expansion card with main card, can survive the vibration test, and reduce weight, have been succeeded in avoiding compatibility issues.

9. Budget Management

- **Effectiveness:** Accurate budget tracking and investigating cost-saving alternatives have been effective in preventing budget overruns.

4.1.7 Technical Risk Analysis: Data Field

The data aspect of the project is also exposed to several risks during the development of the project. Table (13) outlines the anticipated risks and their description, while Tables (14), (15), and (16) provide detailed information on the probability of these risks and their effects during the early, middle, and late stages of the project, respectively. We have also thoroughly examined mitigation strategies, which are described in Table (17).

Risks	Description
Missed External Supervisor Contact	Independent decision-making due to external supervisor absence because of illness.
Changing in Design Without Approval of Stakeholders	In case of lack of communication with the external supervisor, we could make our own decision to make changes in design without Stakeholder's Approval.
Technical problems	For instance, the probability that the access to the web server and/or the code integration between different tasks, failure.
Reliability and Relevance of Toolkits used during the project	Risks based on the relevance and reliability of essential toolkits provided by Tkinter for the project.
Budget Exceedance	Exceeding the allocated budget of 10,000 NOK

Table 13: Risks and Their Description: Data Team

Risks	Probability	Impact	Risk Score
Missed External Supervisor Contact	LP	MP	LP and MI
Changing in Design Without Approval of Stakeholders	LP	LI	LP and LI
Technical problems such as access on the web server and code integration	MP	LI	MP and LI
Reliability and Relevance of Toolkit, such as Tkinter	MP	LI	MP and LI
Budget Exceedance	LP	LI	LP and LI

Table 14: Case Matrix for Data Team: Early Phase

Risks	Probability	Impact	Risk Score
Missed External Supervisor Contact	MP	HI	MP and HI
Changing in Design Without Approval of Stakeholders	MP	LI	MP and LI
Technical problems such as access on the web server and code integration	MP	MI	MP and MI
Unreliability and Irrelevance of Toolkit, such as Tkinter	LI	MP	MP and LI
Budget Exceedance	LI	LI	LI and LI

Table 15: Case Matrix for Data Team: Middle Phase

Risks	Probability	Impact	Risk Score
Missed External Supervisor Contact	HI	HI	HI and HI
Changing in Design Without Approval of Stakeholders	HI	MI	HP and MI
Technical problems such as access on the web server and code integration	MP	MI	MP and MI
Unreliability and Irrelevance of Toolkit, such as Tkinter	LI	MP	MP and LI

Risks	Probability	Impact	Risk Score
Budget Exceedance	LI	LI	LI and LI

Table 16: Case Matrix for Data Team: Late Phase

4.1.8 Mitigation Strategies for Risks in Data Field

Risks	Mitigation Strategies
Missed External Supervisor Contact	Attempt to re-establish communication with ASE as soon as possible. Document all decisions made independently and prepare justifications based on research, experiences, and Team member's competence.
Changing in Design Without Approval of Stakeholders	Making thorough documentation of each change we have made without Approval from our stakeholders.
Technical problems in web server	Using Cloud for verifying that the code is able to communicate with the remote database.
Reliability and Relevance of Toolkits used during the project	Creating standard templates through the use of classes.
Budget Exceedance	As a requirement from ASE to utilize only free and open-source tools, budget exceedance was not a serious risk for data team.

Table 17: Mitigation Strategies Executed by Data Team

4.1.9 Effectiveness of Mitigation Strategies

Evaluation of Mitigation Strategies: Data Field

1. Re-establishing Supervisor Contact

- **Effectiveness:** The strategy to document decisions made by own and provide a detailed justifications in the absence of our external supervisors was effective. It ensured continuity in project progress and enhanced the team's decision-making capabilities. Effort to re-establish communication with the external

supervisor have been succeeded. The completed project was showcased, and the stakeholder was pleased with the results.

2. Handling Design Changes

- **Effectiveness:** Each modification in the design has been thoroughly analyzed and documented. For instance, the rationale for decreasing the number of dipole antennas from two to one has been comprehensively documented. This adjustment has allowed for additional space to implement a camera and solar panels, while also lowering the expenses and complexity associated with the antenna design.

3. Technical Problems in Web Server

- **Effectiveness:** By moving our **SQL database** to the Azure cloud, we have verified that our code is able to communicate with remote databases. **Flask** has not been moved to the cloud during the development, but research has been done on how this can be executed.

4. Reliability and Relevance of Toolkit Used During the Project

- **Effectiveness:** Since the tool is old and limited, we have, through the use of classes (object-oriented programming), created standard templates that can be used for several of our pages so that we only need to create one to two templates instead of a unique template for each page. Using **Customtkinter** makes the look more modern.

5. Budget Management

- **Effectiveness:** Accurate budget tracking and investigating cost-saving alternatives have effectively prevented budget overruns.

5 Preparatory Work

5.1 UV Radiation and Heat

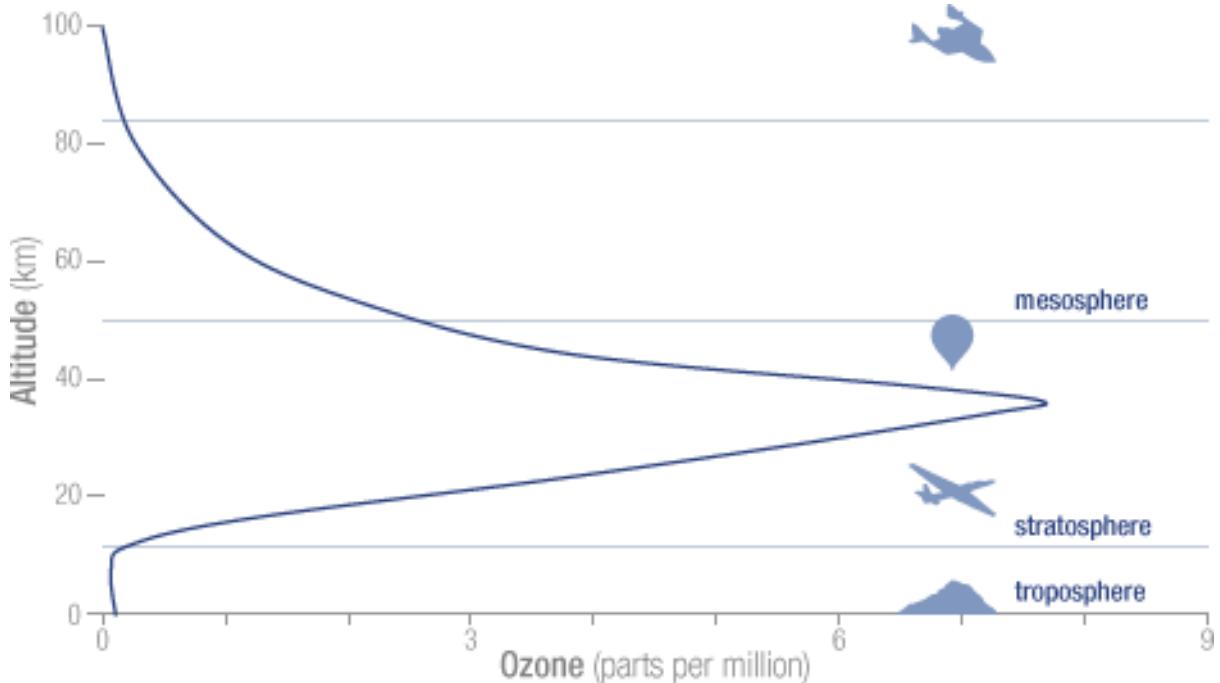
A major contender for viable battery types was Lithium Polymer batteries (LiPo), particularly due to its low price, wide accessibility and smaller mass than its counterparts. However, to verify whether this battery type would be suited for our system context, research was conducted on such.

5.1.1 Ozone Layer

Certain studies led to uncertainties among us regarding whether commercial batteries are equipped to handle UV radiation over extended periods of time. However, most of the UV-C and UV-B radiation (the type of UV that are generally considered the most harmful) is absorbed by the **ozone layer** [12]. Therefore, the peak concentration of ozone in the stratosphere was studied.

The maximum height of the ozone layer is not constant - there are small variations to it. The lower "limit" of 15km is a common denominator (Note: The term limit can be misleading - ozone is present below and above the listed thresholds below. The term limit in this context refers to areas of peak ozone concentration) However, the maximum height of the ozone layer ranges from 30-40 kilometers altitude according to reputable sources: NASA Ozone Watch claims the upper limit ranges between 30-35 km altitude[12] while European Environment Agency (EEA) claims it to be at 30km [13]. However, as Figure 9 shows, the ozone layer exceeds way beyond to higher altitudes.

Figure 9: Ozone Concentration According to Changes in Altitude - Courtesy of NASA Ozone Watch [12]



One could, thus, deductively reason that C.R.A.F.T-Sat's battery will not be exposed to critical levels of UV radiation. However, this may not be true for two reasons:

First, extremely cold temperatures have been a proven causation to the depletion of ozone in the stratospheric zone. According to the EEA, "once temperatures drop below -78°C, polar stratospheric clouds tend to form, which exacerbate ozone depletion" [13]. The is critical information as our external supervisor at ASE claimed that temperatures at C.R.A.F.T-Sat's maximum altitude is approximately can go as low as -85°C. A depleted ozone concentration at such altitudes could thus increase the level of UV radiation that the battery is exposed to. However, it is worth noting that EEA clearly states in their article that this phenomena is significantly more common in the southern hemisphere than in the northern hemisphere, due to differing meteorological behavior [13].

Second, the ozone layer is known for blocking UV radiation from objects and organisms that reside **below** it (i.e., at sea level). Since the C.R.A.F.T will for large periods reside **inside** the ozone layer, the degree of UV radiation should presumably be greater than at sea level. For these two reasons, **isolation of the battery could end up being a vital measure**.

Regarding the impact of UV exposure to the performance and chemistry of commercial batteries, some research do exist. According to an scientific report from professors at Ohio State University, "It has been found that the dimethyl carbonate (DMC), a major

component in commercial battery electrolytes, will decompose significantly under radiation". However, this was only true when moisture was present [14]. However, DMC is a chemical compound that is usually used in Lithium ion batteries (LiB), not LiPo. This study thus aid in solidifying our decision to use LiPo batteries. However, the studies on LiPo batteries is quite scarce relative to LiB. Any conclusions from such must, therefore, be taken in a careful manner. Still, the risk of **moisture** in the atmosphere and above is worthy of concern and consideration. This also further incentives isolation of the battery from these external factors.

Although there are relatively few studies on the impact of UV radiation on LiPo batteries, there is a decent amount of studies conducted on the individual chemicals that constitute said battery. The polymer electrolyte in LiPo batteries usually comprises of is Polyethylene Oxide (PEO). The studies shown, contrary to skepticism, actually show **positive side effects** on PEO's conductivity after prolonged UV exposure. A study published collaboratively by professors at the university of Shanghai and the University of Uppsala showcased a study in which a crystallized PEO (CPEO) electrolyte that was UV cured. The results showed a "much higher ionic conductivity "at room temperature than that of pure PEO solid state electrolyte" [15]. One theory behind these findings is that prolonged UV exposure reduces the crystallinity of the PEO's ethylene oxide (EO) chains which in return increases the transfer of Lithium ions [15].

One type of radiation that is almost unanimously negative on commercial batteries is gamma radiation [16]. Fortunately, that should not be a concern for our system scope. Still, if ASE at a later stage wants to launch C.R.A.F.T-Sat into low orbit, then further research would have to be conducted on gamma rays (due to higher concentrations of it in outer space), potentially opt for a different battery type that is equipped to handle such radiation.

5.2 Prototype material

The frame of CubeSat design has usually a relation of being produced with the material Aluminium or Titanium. Those materials could be used with 3D printing if we use the Selective Laser Melting (SLM) method (information about a SLM is described in appendix L). Since the prototype has to be 3D printed it's more reasonable if a user also understand of how the 3D prints produces the final design of the C.R.A.F.T-Sat (the final design is described in section 7.12). Given from ASE the printing method had to be Fused Deposition Modeling (FDM) (FDM method is described in appendix J). A FDM is a easy to use machine which could have a low cost, compared with SLM, and a user with non-experience could easily be taught how to use it.

The prototype developed and components produced, is was made by using a FDM printer two of the group members own. The materials used with FDM is defined as thermoplastic, we have therefore selected three thermoplastic polymers as the most reasonable prototype materials for this project (read appendix O.2 for a simple description about thermoplastics). The thermoplastic polymer we selected was Polyactic Acid (PLA), Polyethylene Terephthalate Glycol (PETG) and Polyether Ether Ketone (PEEK). It's important to note that the material which should be used for realistic use and is produced with 3D printing is described in appendix: I. Those materials will be evaluated with pugh-matrix and the selected material(s) will be used for a tension test to gain some of the necessary data for a Finite Element Analysis (FEA) and material evaluation of the prototype.

STB|

5.2.1 Polyactic Acid (PLA)

The polymer material PLA is one of easiest and one of the most common thermoplastic material used in the Additive Manufacturing environment. The reason PLA is common material are because PLA doesn't demand a lot of experience and skills to be used for Rapid Prototyping using the FDM method, and PLA doesn't cost a lot, and it's easy to be stored when it's not used. But, PLA has its disadvantages. Those disadvantages are related to low UV tolerance, liquid and chemical tolerance, and a low fatigue resistance. In total, PLA has disadvantages that limit its physical usability, and therefore results with a limit of where to use the material in other states than eye contact with a prototype.

STB|

5.2.2 Polyethylene Terephthalate Glycol (PETG)

The polymer material PETG is one of easiest and one of the most common thermoplastic material used in the Additive Manufacturing environment. PETG is material which needs a heated print-bed during the manufacturing with the FDM method. PETG are often used in developing a prototype with a tougher impact strength, better UV and a better chemical and liquid tolerance; compared with the PLA data.

STB|

5.2.3 Polyether Ether Ketone (PEEK)

The polymer material PEEK is a type of thermoplastic, known to be one of the highest performing in its type and is a very lightweight material, with a high specific strength (strength per unit weight). The advantages with PEEK is; a high heat resistance, a little too no reaction against harmful radiation and harmful chemicals, and a comparable tensile and load strength to titanium and steel, while being many times lighter.

The disadvantages of PEEK material in a study project is the high material price compared with the PLA and PETG. PEEK also demands a high price FDM printer because of it's high demands with heat room, heat bed, and nozzle heat.

STB|

5.3 Pugh-matrix and selection of prototype material

The pugh-matrix is chosen to decide the best prototype material(s) for the C.R.A.F.T-Sat. *The chosen material for the current solution is in this case related to the purchase and emptying aluminium soda cans.* In a pugh-matrix, values has to be given to each alternative related to each criteria. Therefore, question related to each criteria has been made:

- Yes; value 1.
- Equal; value 0.
- No; value -1.

In table 18 the alternatives of materials are; Alt. 1 the PLA, Alt. 2 the PETG and Alt. 3 the PEEK. Too chose the best material related to our upcoming print-produced C.R.A.F.T-Sat frame and other parts, these criteria questions have been made:

- Does the material cost less than the current soda-can solution.
- Is the material more robust in the working environment compared with the soda-can solution.
- Using the new material, is the operability better compared with the current solution.
- Is the production with the new material easier compared with the current solution.

Criteria	Current solution	Alt. 1	Alt. 2	Alt. 3
Marketing (cost)	0	1	1	-1
Robust	0	-1	0	1
Operability	0	-1	-1	1
Production	0	1	1	-1
Result	0	1	0	

Table 18: Pugh Matrix: prototype material | STB

The result from the table 18 results with PETG to be the most appropriate material for the prototype in our project. But, since PLA is easy to use on a FDM printer and the material cost is equal with a PETG it will also be considered to be used depending on the results from the final tension tests described in section 5.5

STB|

5.4 Polymer tension staff-test plan

To get a final conclusion about if it's PLA or PETG which are more appropriate for the C.R.A.F.T-Sat design and prototype production, a tension test was done. The tension test is also used to gain data about the chosen 3D printing settings in Cura (settings chosen in Cura is described in appendix K) and the influence of a temperature environment change compared with non-temperature change. The final test plan is chosen to be shown in fig 10, and is representing the group members as a test operator.

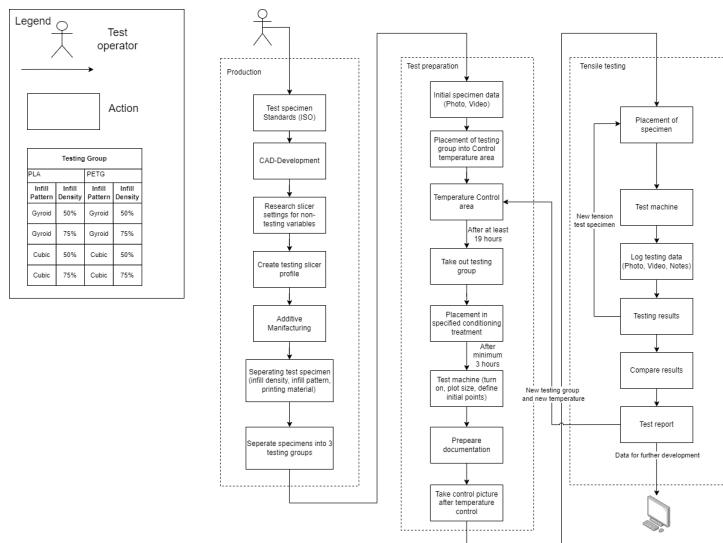


Figure 10: Material (tension data) testing plan | STB

The test plan is divided into three phases; the production phase, the test preparation phase and the tensile testing phase. Because of time, the test started after standard researching and test plan diagram making, but before the fully section writing of the phases. Therefore, the section writing describing the phases will represent the work before the test, but also how the test plan is used and the work during test.

STB|

5.4.1 The production phase

To simplify the necessary work we decided to find *NS-EN ISO standards* relating to documentation, modeling- & testing specimen, as the most effective solution to gain the data we are seeking. The chosen standards are described in appendix Q.

The test staff CAD is described as 1A in fig 559 designed on Solid Works (SW) to be mostly equal the design from *NS-EN ISO 527-2:2012, chapter 6.1 Shape and dimensions*.

The tension staff has a opening of re-design if the tensile-testing machines requires it. One of the reason of the tension test is to decide the slicer settings in a chosen slicer program; in our case the program is Cura. The chosen slicer settings for Cura is described in appendix K. Before the test preparation phase, the 3D printed staffs is dived into the testing groups related to the three temperatures before test.

STB|

5.4.2 The test preparation phase

It's important to document the necessary data of the initial specimens (*test staffs before tensile test*) before the tensile test. The important data documented before test was 3D printing slicer settings; because it had to be equal for all printing test staffs.

Since we didn't have the possibility to test the polymer material with the tension machine in a environment temperature equal the environment change due to altitude, we chose to gain test data equal as possible as if the C.R.A.F.T-Sat has stayed in an altitude with a specified temperature for a specified time and from there suddenly dropped to an altitude where the temperature is equal the ambient temperature described in *NS-EN ISO 291:2008*

The specified temperature control area is divided to represent three altitude sections (the specified temperature is an interval because of possibilities in temperature change in the fridge/machines during use);

- High troposphere/low stratosphere altitude (8 to 26 km over the sea) temperature: -38°C to -40°C
- Low/Middle troposphere altitude (0 - 8 km over the sea) temperature: -18°C to -20°C
- Low troposphere/ground altitude (room) temperature; 20°C to 24°C

Note: The chosen temperature lays between those altitudes and isn't a constant as an environment temperature during the increase and decrease in altitude, therefore, the temperature could be different and lower.

We decided to have the test staffs in the temperature control area for at least 19 hours; the C.R.A.F.T-Sat won't be in an altitude with the lowest temperature for 19 hours but we chose the amount of time to gain a possible data difference and a possibly clear difference in the test specimens result.

After the test specimens has been in the specified temperature, they were placed in the testing room and not used for at least three hours. During the waiting time preparation of report documentation was done. The reports as based on the standard *NS-EN ISO 527-1:2019*.

STB|

5.4.3 The tensile testing phase

It was important to place all the test specimens equally as possible in the tensile testing machine to get an approximate equal outcome, therefore measuring, writing, and pictures were taken before and after the test used for comparing and documenting all the given results.

Note: the friction coefficient on the test specimens was low (it was physically shown during testing) and required manually press on the clamp to prevent a failure in testing data. We therefore used manually press with 100, 300 and 500 newton, depending on the surface on test staff. The data from each test is stored on a memory pen divided into folders related to variable factors; polymer material, specified temperature, infill pattern and infill density. The reports from the testing is shortly texted and filled with pictures and results, and is described in the appendix with the reports and obtained testing data P.

Note: the diagram 11 shows that because of the chosen variables and selecting ISO standards to follow, it's necessary for 60 test specimens per polymer material. Therefore, it ended with 3D printing 120 test staffs (includes PLA & PETG) to get the necessary data before concluding the best material and slicer settings.

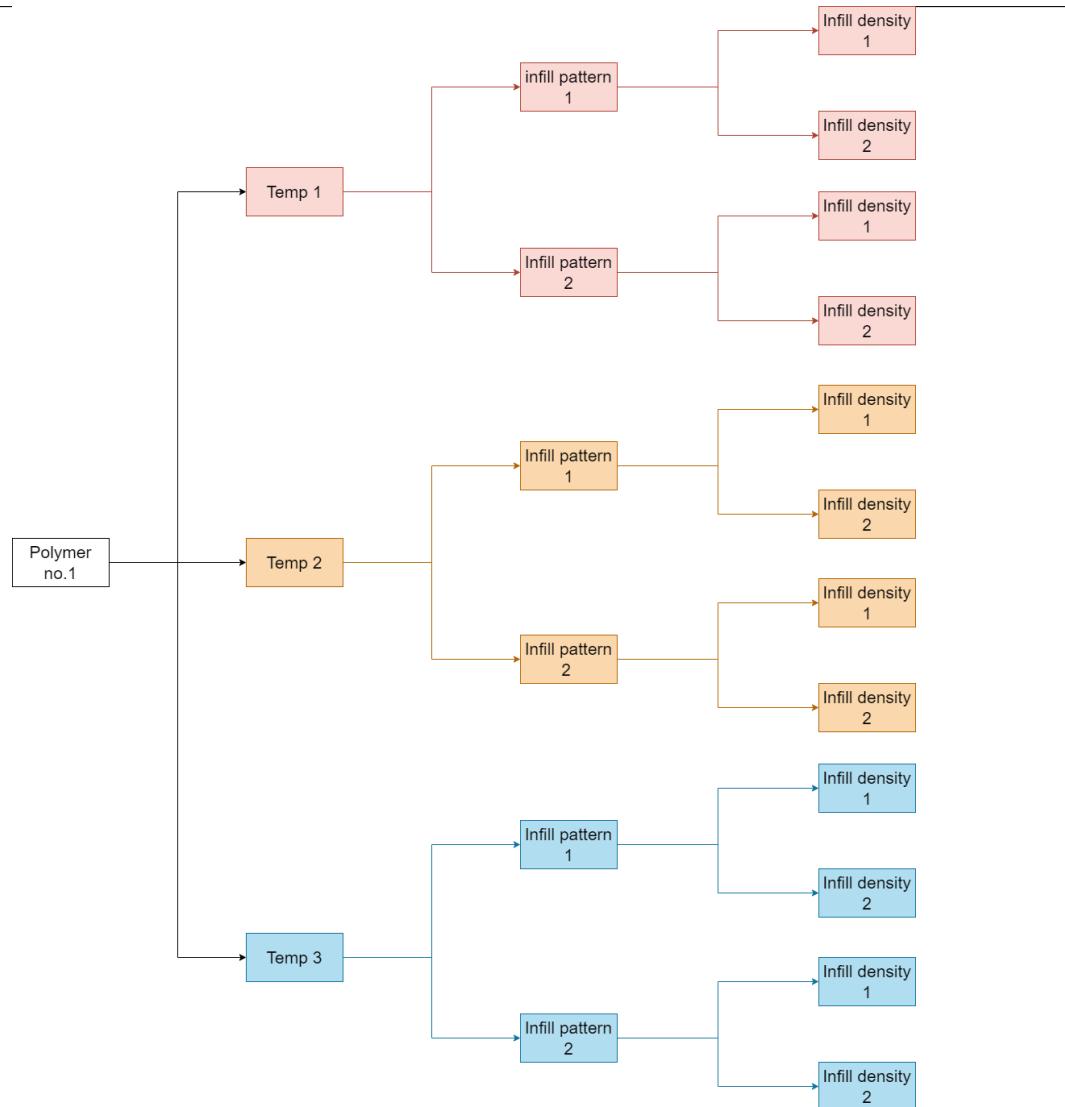


Figure 11: Testing factor diagram (per polymer material) x5 test per infill density | STB,TL

STB|

5.5 Results from tension test and final polymer conclusion

The tension test done gained important data to be used for FEA and other important factors related to material data. The fig 12 is table with a summary of the mean values from the tension test results appendix P and gives the possible decision of what polymer material and 3D printing settings in Cura should be used for FDM printing.

Note: since the test required manually pressure data was removed from the stress-strain curves described as possible failure data. Therefore, the tensile strength of the sections temperature 2. and temperature 3. is removed because the gained data was unrealistic. The strain data gained from the testing was chosen as not required for comparing in this case, but is documented in each test report and can be changed as required if it's

suddenly necessary. since it's difficult to use it to calculate the Poisson's ratio. The ratio and other necessary values is at this moment defined as material data from a browser and not testing.

A	B	C	D	E	F	G	H	I	J	K
1			3D print material							
2			PETG				PLA			
3			Gyroid		Cubic		Gyroid		Cubic	
4			50 %	75 %	50 %	75 %	50 %	75 %	50 %	75 %
Temperature & Values (Celsius)	Temp 1 (-38 to -40)	σ_y [MPa]	11,86	11,99	11,41	11,16	9,31	9,10	7,94	8,53
		E_t [MPa]	2301,45	3001,59	1028,13	2134,68	1613,40	1702,48	237,76	1202,37
		S_{ut} [MPa]	37,11	37,77	36,47	39,78	35,55	39,61	35,27	39,51
	Temp 2 (-18 to -20)	σ_y [MPa]	10,27	11,62	10,65	11,73	8,56	9,03	8,40	9,69
		E_t [MPa]	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
		S_{ut} [MPa]	35,88	36,16	34,50	106,63	33,56	38,33	33,19	37,49
	Temp 3 (20 to 22)	σ_y [MPa]	9,48	11,11	11,22	11,01	6,75	9,24	7,89	7,58
		E_t [MPa]	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
		S_{ut} [MPa]	34,92	38,67	37,36	38,90	22,69	39,14	36,25	38,86
14	Test staff types:		1	2	3	4	5	6	7	8
Mean values (temp)		σ_y [MPa]	10,54	11,58	11,10	11,30	8,20	9,12	8,08	8,60
		E_t [MPa]	2301,45	3001,59	1028,13	2134,68	1613,40	1702,48	237,76	1202,37
		S_{ut} [MPa]	35,97	37,53	36,11	61,77	30,60	39,03	34,90	38,62

Figure 12: Summary of the mean values from tension testing | STB

Since the table in fig 12 has a lot of values results in difficulty to define each outcome for comparing, a section in the table has been made using math and develop a new mean values based on only the section *3D print material*. This resulted in a easier method in comparing values and possibility to decide which printing sections and polymer should be used. The diagram in fig 13 includes the values of tensile modulus. It shows a big difference in the highest and lowest value under the comparing of which polymer material. According to the diagram, gyroid pattern has the highest tensile modulus values in both polymer material, and is therefore according to the infill pattern and tensile modulus the most appropriate for a prototype where we don't know the possible results but want to be safe. When we are comparing the infill density and using the possible chosen infill pattern, it's shown that using the percentage value of 75 has the highest stress value for the tensile modulus.

The diagram in fig 14 describes the proportional limit: yield strength. The data from the program developed by a student used for tensile testing didn't include the correct yield strength. Therefore, it's chosen to define the proportional limit yield strength as the yield strength. It's a possibility to gain error's in the analysing and will be considered when using the data during FEA. The diagram shows that the lowest stress value of PETG is higher than the highest stress value for PLA. It could therefore be considered that PETG is better to be used depending only on the stress and not data of ductility. An consideration is to chose the test type 6, which has the highest yield strength values for PLA if it's suddenly needed to use PLA.

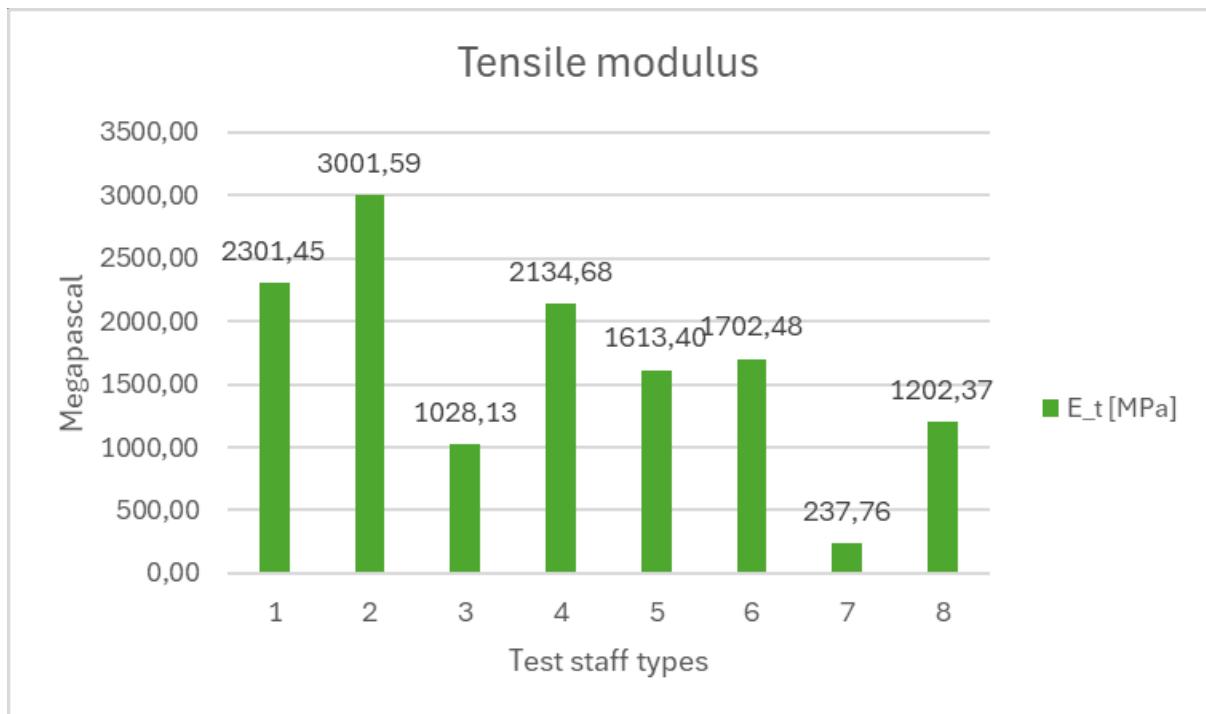


Figure 13: Tensile modulus | STB

According to the yield strength diagram and PETG polymer material, values with the infill pattern cubic gives almost identical values, but choosing a gyroid pattern results in either the highest or lowest stress value depending on the density percentage. The percentage value of 75% is most appropriate to get a stress-strain curve with a high "yield-strength" value and non-defined strain value.

The diagram in fig 15 describes the ultimate tensile strength (UTS). According to the diagram the UTS is mostly equal for both PLA & PETG except of test staff type 4 with a stress value approximately 1.5 – 2 times higher than the other tests. It can be concluded with PETG to be the best option according to the UTS but with the approval of equal values of PLA.

From all the data, the material, and printing variables, the best settings for 3D printing and material is the test staff type 2 (material: PETG, infill pattern: gyroid & infill density: 75%) with the material data:

- Assumed yield strength; $\sigma_y = 11.58$ MPa
- UTS; $\sigma_{uts} = 37.53$ MPa
- Tensile modulus; $E = 3001.49$ MPa

We accept the possibility to change the test staff type 6 (material: PLA, infill pattern: gyroid & infill density: 75%). The material data is:

- Assumed yield strength; $\sigma_y = 9.12$ MPa

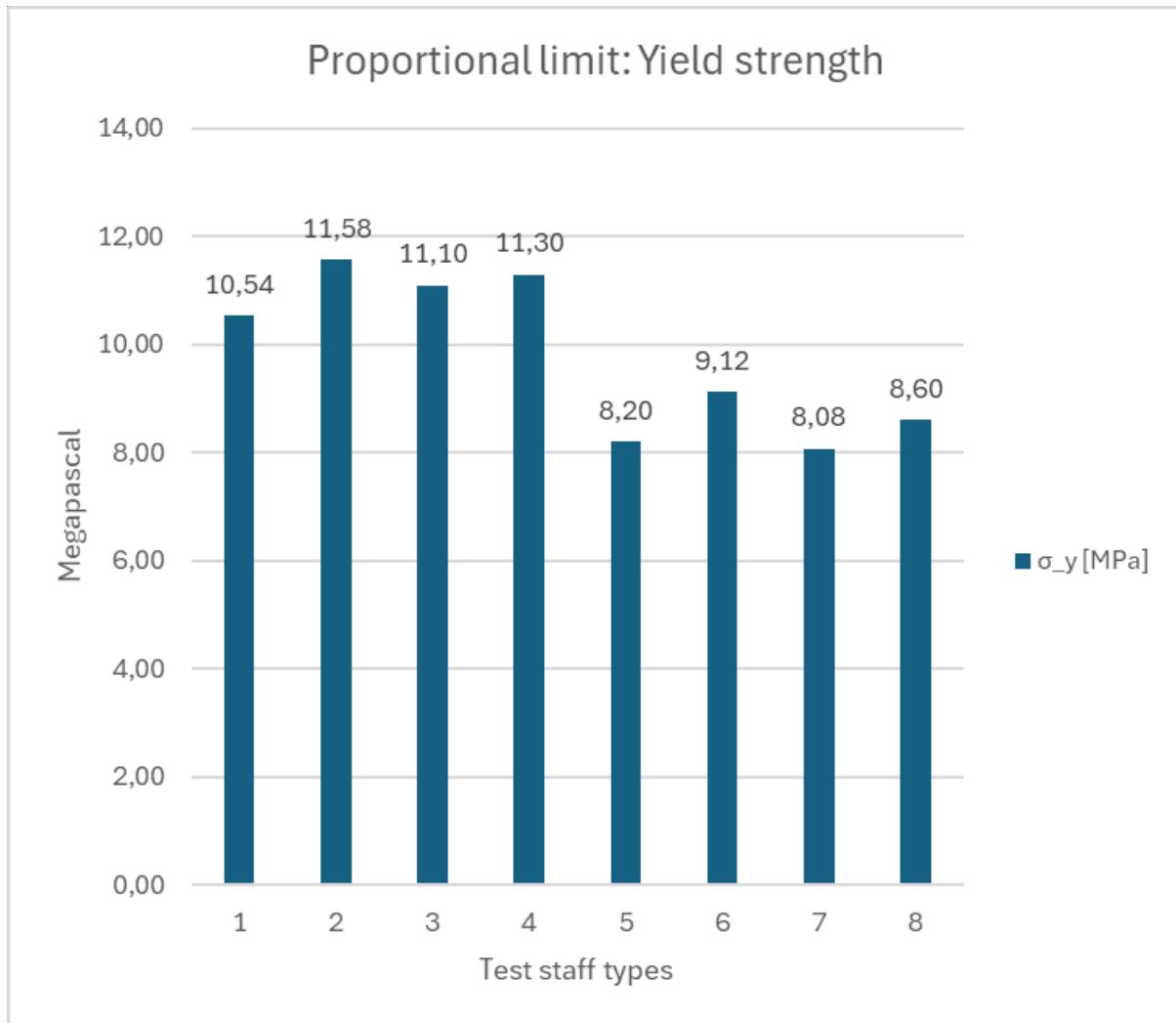


Figure 14: Proportional limit: Yield strength | STB

- UTS; $\sigma_{uts} = 39.03$ MPa
- Tensile modulus $E = 1702.48$ MPa

The product will mainly involve the PETG but PLA will be the back-up material if interactions happens. The concluded materials is only based on high values of stress (not strain) and could be changed if it requires reduction in the mass and the chosen settings could be a option in reduction. Strain values is documented in the test reports if it suddenly is necessary to be used, and it could possible change the chosen materials.

Note: the chosen materials of prototyping is not defined as the best option for our case because it's not usually used and since the testing was related to what we was possible to do on the university, we chose to have the chosen values as constants during the usement in altitude increasing, pressure decreasing and temperature changing for FEA.

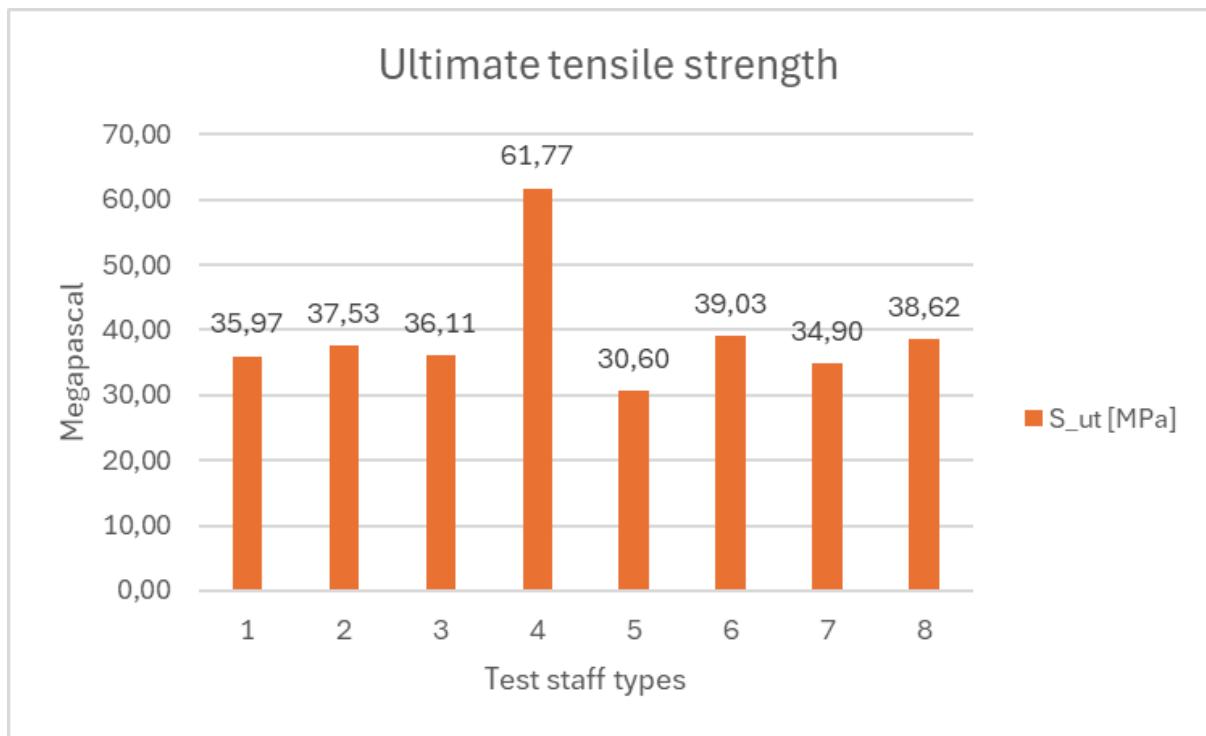


Figure 15: Material data: Ultimate Tensile Strength (UTS | STB

5.6 Laws and regulations regarding radio and sensors

When communicating using radios there are certain laws and regulations you need to follow. Different countries have different laws related to radio use. It is therefore important for us to get an insight of the different laws we need to follow, in order to answer our stakeholder requirements set by the Norwegian Government 3.1. In Norway we have "General Authorisations Regulations" [17]. The regulation controls which frequency bands are dedicated for free use. Since our radio use the 433Mhz radio frequency band our radio use falls under Chapter III. §8, Section 11,12 and 13 of "General Authorisations Regulations"[17]. Where the regulation states you are allowed to use the frequency band if you follow the EN 300 220-2 standard [18].

We want our satellite to have the capability to transmit pictures or videos to the ground station, we need to follow row H in Annex B, Table B.1 in the EN 300 220-2 standard[18]. Where it states that we can use a maximum effective radiated power, e.r.p. of 10mW and a duty cycle of 10%. We can however use a higher duty cycle if we implement Listen Before Talk (LBT) with Adaptive Frequency Agility (AFA). To reduce the complexity we have decided to use a duty cycle of a maximum of 10%. More details about the implementation of this can be seen in the section 7.2

Since we are planning to collect sensor data using an "airborne sensorsystem", we also have to follow the regulations regarding airborne sensor systems [19]. In Norway there are set "sensor prohibition areas", these areas can be found on NSM's website[19]. A screenshot of the current sensor prohibition areas can be seen in Fig.16. In Kongsberg there are currently 3 of these areas, these are Heistadmoen, Arsenalet, and Kongsberg Teknologipark. They do however not span across our place of work, which is Krona, meaning we are outside these areas and can collect sensor data. It is however important to check if you are within a sensor prohibition area before launching any CanSat or C.R.A.F.T-Sat since they do collect sensor data.



Figure 16: Sensor prohibition areas

Since both the CanSat and C.R.A.F.T-Sat are deployed using a weather balloon, they fall under Chapter VIII,§24 of the "regulation about air traffic rules and operative procedures"[20]. Since we are not able to test our system using a weather balloon, we have decided to not give much attention to this regulation since it involves the launch of the balloon and does not regard **the use** of our system to any extent, and it will be the launch providers responsibility to follow the regulation.

TL|TL

5.7 Pins used in software

To get an overview of which pins on the Teensy 4.1 are used for which modules we created a diagram over which communication protocol, can use which pins and which pins need to be used together, this was done so we would have a better insight into what pins can be changed in software and what choices need to be made in hardware. Looking at the diagram in Fig. 17 each pins sharing the same color, for the same protocol, share a

bus, meaning you need to use them together. Looking at the CS (Chip Select) pins for SPI pins 11,12,13 can use either 10, 36, or 37 as CS pins. The pins with a bright green color indicate that any digital pin can be used. However, it is good practice to use the dedicated pins since they have special hardware associated with them, and they will be faster than a software solution. The bright red box specifies that the digital pin must be interrupt-capable. This however is not a problem with the Teensy 4.1 since all digital pins are interrupt-capable.

UART		SPI				I ² C		RFM-96 Specific	
Rx	Tx	CS	MOSI	MISO	SCK	SDA	SCL	Reset	Callback
0	1	10	11	12	13	18	19	Any Digital PIN	
52	53	36				17	16	Needs interrupt: Any digital pin on teensy will do	
7	8	37				44	45		
15	14	0	26	1	27	24	25		
16	17	38							
21	20	44	43	42	45				
46	47		50	54	49				
25	24	Used for SD-Card-Reader							
28	29	Any Digital PIN							
34	35								
48									
Any Digital PIN, slower speed									

Figure 17: Diagram over pins used for different communication protocols

5.8 Deciding radio module library

TL|NNG

When deciding on what library to use for the RFM-96 radio module which is one of the modules used in the 2018 CanSat Kit, we needed to look at several options. ASE has developed their own library for the module. This library is documented with a basic library command set, found in the CanSat book [21]. However, on this page it states "*other commands will be very specific and is not documented*". The functions that are not documented are essential troubleshooting and tuning, without proper documentation available, we have decided not to use their implementation of the module but opted for an alternative library.

To find an alternative library we developed some requirements, we need the library to meet.

1. It needs to work with Semtech SX127x chip.
2. it needs to be well documented.
3. it has to have an easy interface to use.
4. it needs to have **at least** the same functionality as ASE's CanSat_rfm96 library.
5. it needs to have active users online for troubleshooting.

To find libraries to use for Teensy, Arduino's library section is the main place to start looking. Here we found two possible alternatives for libraries to use for the RFM-96 module, Sandeep Mistry's LoRa library and Chandra Wijaya Sentosa Lora-RF library. Mistry's library is more straightforward to use, whereas Sentosa's Library is more complex, offering more flexibility.

Both of these libraries are suitable for our needs, but we have opted for Mistry's LoRa library because it answers our requirements better. The most important requirement that made us choose this library was the documentation. It has easier-to-follow documentation, and Sentosa's LoRa-RF library lacks some documentation for key features like more advanced sending and receiving operations. When doing research for LoRa modules for Arduino, Mistry's Lora library seems like the de facto standard for these types of radio modules, making it a great choice for our application.

The only feature all these libraries lack are the ability to save data to an SD Card. This is something ASE has implemented to tailor their library. This functionality is implemented using the official SD library from Arduino A.10

5.9 Deciding GY-91 library

TL|NNG

When choosing what library to use for the sensor package GY-91 which is one of the sensors used in the 2018 CanSat kit. We went through the same steps as with the radio in sub-section 5.8. Since the GY-91 is a multi chip sensor G.3.2, we need a library for both the BMP-280 and MPU-9255, or a library that combines both. ASE has developed a library that combines both the chips. In the header of this library they reference that

they have resourced a MPU-9255 library developed by Krzysztof Adamkiewicz. The BMP-280 section of ASE's library states "the library is almost fully taken from Adafruit's library" They have however shortened the library since the chip will only communicate using the I²c protocol. The current GY-91 developed by ASE library would probably work fine, we have however decided not to use this library. The reason for this decision is that there lacks a date when the library was developed and/or updated. Since this information is not available, we are assuming this library was created somewhere around the same time their radio library was created, that is in 2018.

When looking at the commit logs for both of the libraries used in ASE's library we can see that both of them has had multiple bug fixes and updates after 2018. Our last stakeholder requirement 3.1, need the code to be maintainable in the future. By including the newest versions of these libraries, it is easier to troubleshoot, bug fix and update in the future since you can assure what version of the libraries that used.

The downside to doing it this way is that it can complicate things for our end users, since they need to differentiate between the different chips on the GY-91 Module, this is something we want to avoid. To circumvent our solution to make the end product more difficult to use, we have decided to make easier to use function's within the C.R.A.F.T-Sat class that use the included libraries, so the end users only need to think about the the sensor module and not individual chips.

NNG|TL

5.10 Impact of Quality of Data Transmission: Doppler Shifting

One physical phenomena that warranted concern early on was the *Doppler effect* (also commonly (and incorrectly) referred to as Doppler Shifting). The Doppler Effect can be defined as "an alteration in the observed frequency of a sound due to motion of either the source or the observer." A real-life example of such an event is the change in pitch from a passing train; when the train (moving source) is approaching the listener (the stationary object), the noise is perceived as much higher pitch than when the train recedes the listener; even if the velocity of the train and its distance to the listener in both instances are equal. This change in the relative frequencies is defined as the *Doppler shift*.

Although the Doppler Effect can still occur when both the observer and source are moving, the circumstance that is most relevant to our system context involves a stationary observer (i.e., Ground Station) and a moving source (i.e., C.R.A.F.T-Sat and CanSat). In such cases, the frequency received by the observer can be calculated using the formula below:

$$fo = fs \frac{v}{(v + vs)} \quad (1)$$

With the operations of satellites, the Doppler Effect has proven itself to be a threat to the validity of the data that a ground station receives. Thus, even slight changes in the frequency values can prove obstructing. This is particularly true if the satellite is fast-moving, such as if C.R.A.F.T-Sat was attached to a student rocket. To further add, Long Range (LoRa) technology (which both C.R.A.F.T-Sat and CanSat will use, does not explicitly feature built-in functionality to account for the Doppler effect. However, a study published by the IEEE in 2019 indicates that LoRa devices are reasonably resilient to Doppler shifting [22]. However, another IEEE-published study states that this is less true for *dynamic Doppler shifts*, i.e., when "the relative velocity between two communicating nodes" is changing [23]. This latter is relevant if it is detached from its launch mechanism and is traversing downwards, especially if a parachute is not attached. A document published collaboratively by the United Nations (UN) and the Japanese Aerospace Exploration Agency (JAXA) proposes two methods to "account for frequency variations due to the Doppler shift":

- Implement frequency tolerances: This could be achieved by providing the ground station transceiver(transceiver if up-link is supported) with a large enough range of acceptable receiving frequencies that deviate from the intended frequency. This is the simplest method to configure but at the negative cost of system performance. This is because a larger range can make the ground station more susceptible to noise and thus, lower the quality of the data received.
- Include automated "tracking of frequency generated by the Doppler Shift": This solution requires more time developing and tuning, but can be a better way of maintaining both the precision and accuracy of data transmission. By regularly checking and adjusting the reception frequency to align with changes caused by the Doppler shifting, a balance can be struck between maintaining data quality, and streamlining the data collection process. There are two methods for achieving this that were deemed plausible given our system scope and were considered: using dedicated software, or hard-coding automated tracking in the ground stations library.

5.10.1 Dedicated Software: SDRSharp

A common method for handling the Doppler effect is utilizing a Software Defined Radio (SDR) which provides automated Doppler shift correction. A popular Software SDR is SDRSharp [24]. This is a tool that is both open source and free. Being open source, SDRSharp is regularly being updated and supported by both the system developers and its community of users, satisfying requirement regarding a "10-year longevity". Being a dedicated software, it is feature-rich and abstracts its implementation from the end user which can make easier to use by ASE and their enrolled students. Unfortunately, there is no built in integration between SDRSharp and Arduino code (which our Ground Station is programmed in). This can make communication between this dedicated software and our ground station more complex and thus counteract any benefit provided by SDRSharps's abstraction and functionality.

5.10.2 Hard-coding Doppler Shift Correction

This method involves programming Teensy 4.1 with Doppler Shift Correction. Due to time constraints, we have been unable to implement such correct to our Teensy 4.1 library. However, it is the solution that we would recommend to ASE. The main advantage of this method is that it is built into the library that Team Software is producing instead of needing another separate subsystem. Having it hard-coded/available in the library can also be a learning opportunity for curious students who want to learn more and experiment with the Doppler effect, satisfying not necessarily any requirements, but is quite fitting to our system context and scope. Lastly, coding Doppler shift corrections ourselves provides more freedom and clarity for the system administrators, as they are in full control of how their software works and are not reliant on another software product whose implementations are more complex and perhaps not well documented or unclear.

5.11 Understanding our Target Audience - User Testing

On April 15th 2024, we arranged a user testing session at Vestsiden Middle School in Kongsberg, where six 10th grade students were showcased various of the subsystems that we had developed so far. The students were also able to explore some of these subsystems via prepared activities. For GUI used to plot telemetry data (C.R.A.F.T-GUI), each student was assigned blank, weighted Pugh matrices. The purpose in doing so is that by having the students fill out the matrices themselves, then higher trust can be placed on their results since the students have less of a preconceived bias than Team Software. The matrix forms were for evaluating three concept designs for C.R.A.F.T-GUI frontend, differing in both in layouts and/or color scheme. The designs themselves and the filled out Pugh matrices is available in Subsection 8.6.4. Perhaps more interesting than the designs themselves, however, where their importance rankings of various criterion of what makes a quality GUI. The five factors ranked were:

- Ease of use
- Intuitive
- Features
- Aesthetics
- Performance/Speed

Each student gave each factor an importance ranking of 1-3, where 3 is the highest.

Table 19: Pugh Matrix - Importance Rankings for a GUI. Taken from user testing at Vestsiden middle school

Criteria (1-3)	User 1	User 2	User 3	User 4	User 5	Total
Ease of use	2	3	3	3	3	14
Intuitive	2	2	2	3	2	11
Features	3	3	3	2	3	14
Aesthetics	3	1	2	3	3	12
Performance/Speed	2	2	3	3	3	13

Analysing Table 19, all five criterion scored relatively high. However, "features" and "ease of use" were the criteria which received on average the highest rating, with a total score of 14/15. Interestingly, "User 1", whom only ranked ease of use a 2, scored aesthetics a 3. This indicates (for some) an the acceptance of a trade-off between visuals and system simplicity. However, the majority of young students prioritize user-friendliness and function over visuals. "Intuitive" was the lowest scoring factor. However, some students revealed during the testing session that the term "intuitive" was new to them. There is a probability that this could have contributed to the low score. "Performance" ranked second with a score of 13/15.

5.12 Deciding Medium for Storage

5.12.1 Text File Types

The only mandatory storage medium regarding C.R.A.F.T-GUI is mentioned in objective no 6: "sensor data must be plotted in real-time, with the option to **save data to file**", more specifically, as a text file. There are three popular text file formats that we looked into, each with their unique perks:

Text File Document (.TXT) is the most basic file type there is; it stores data as a plain text file. .TXT files can store any value and its values are not separated by a fixed delimiter or values, making it more suitable for unstructured data. However, because it does not enforce a fixed structure, .TXT files are generally not used nor recommended for storing that that has to be organized in a consistent manner, such as users, sensors, and sensor data.

Tab Separated Values (.TSV) files is a text file type in which columns and rows that encompass data values are parsed/separated via a tab [25]. The extra space/padding created by the tab tends to make the data easier to read; .TSV is in many ways a file type that produces an output that is the most similar to most data tables that people are familiar with. This is an advantage considering our young target audience may not be comfortable with other data structures that may require an adaptation period and learning curve. Because of its tabular output, .TSV files are generally easy to import into Microsoft Excel and databases. The main argument against .TSV files and why it is not the most popular text file type is its higher risk of formatting issues caused by human error. Tab indentations can in some cases be mistaken for multiple spaces. Since the output of what a tab and space delimiter produce is so similar, troubleshooting can become a time-consuming nuisance. An appropriate analogy between .TSV files and Python code which also uses indentation to separate scopes rather than a semicolon. These can be difficult to detect and resolve for our young target audience and even C.R.A.F.T-GUI's administrators.

Comma Separated Values (.CSV) files is the most popular file type by far. They are nearly identical to .TSV files expect that the rows and columns are separated by a comma instead of a tab [26]. Although harder on the eyes than .TSV, the explicit use of a character rather than empty space makes it more obvious where values are separated, and is thus way faster to identify and resolve errors. This also enforces a clear structure to data which is useful for saving telemetry data as it adds a layer of predictability to our end users. Another major advantage in favor of .CSV is its great support by third-party applications and plug-ins. A primary reason for ASE requesting a feature of saving to text file is to be able to **view and process** such data via third-party applications such as Excel and Python. Python is a programming language that astonishes in data process and analysis and is often used as an introductory programming language for middle school

students and above in Norway. Pandas is as of today, the most popular software library to enable such functionality. **This tool (among others) is better tailored for .CSV files than .TSV.** For instance, Pandas has a built-in function called `read_csv()` that reads data from a .CSV [27]. There are no such built-in functions for .tsv files: Users have to utilize workarounds such as using `read_csv()` and specifying a tab as a separator. This is not intuitive, especially for someone with minimal programming experience. .CSV is also the file type that is preferred by ASE, and based on our research, we align with their viewpoint. **Thus, the telemetry saved locally to file from C.R.A.F.T-GUI will be in a .CSV format.**

5.12.2 Databases

Utilizing database(s) our data architecture for C.R.A.F.T-GUI is part of our MVP solution and was not that ASE initially asked for. However, we still believed that the inclusion of a database was something worth considering in our design and was thus pitched in to, and approved by, ASE. Although leveling up system complexity, there are certain facets to using a database that can add great value to both our system buyer and users:

Improved performance: Data retrieval from a database is often significantly faster than reading from a text file. Database superiority in this regard becomes even more evident as the amount of data stored increases. With proper indexing (i.e., "data structures that can increase a database's efficiency in accessing tables"), data retrieval can be rapidly fast compared to reading from a text file.

Predictable and greater scalability: Due to the storage of data being more efficient, storing data on a database can often prove to be more scaleable with an increase in data size. Furthermore, structured query languages (SQL) require the database to be normalized [see Subsection 6.35]. A database being normalized to the third normalization form or higher, assures that the database can be scaled up and its values modified and removed, without causing critical damage to its infrastructure. Given our system scope and context, the inclusion of a database could be used to generate and store user accounts, register student groups, sensors, and telemetry data to name a few.

Feature-rich: Support for databases is also integrated into many programming languages, including Python. This means that the database can quite seamlessly be integrated with our Python code (via APIs) and send user input from the GUI to the database and vice versa in a streamlined manner. This could in return be used to increase functionality such as having a satellite configuration page [see Subsection 129]. The API used to streamline communication between C.R.A.F.T-GUI and the database was MySQL.Connector.

Long-term storage (+ Web server): It is harder to delete a database row than a text file on accident. Deleting data from a database requires a query to be run; deleting a text file can be as simple as accidentally pressing 'delete' on your keyboard. Storing

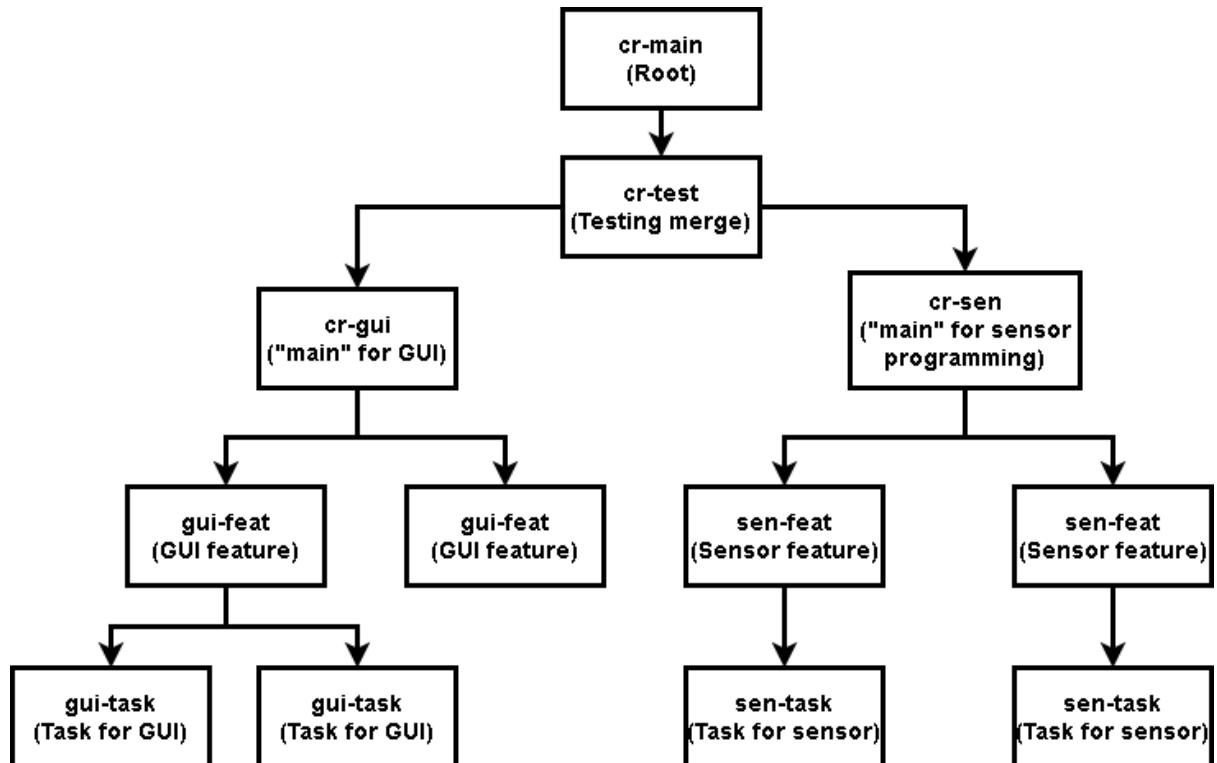
of logs in a database means that data can be accessible to others which is useful for students since the log can be accessed by all members instead of it just being stored on one user's computer, Storing in a database is also more resource-friendly as it frees up local storage.

NNG|

5.13 Setting up Coding Environment

For the software developers to steadily, efficiently and safely develop code, a simple structure for version control in Azure Repos has been designed.

Figure 18: Our Git Structure in Azure Repos



As shown in Figure 18, our solution consists of having both the GUI and sensor programming in the same repository. Hosting separate repositories for the GUI and sensor programming was seriously considered. However, further research revealed that this would make merging code together (which is inevitable since the GUI plots the programmed sensor data) would most likely become a complex, risky and tedious task [28]. Having two closely linked subsystems in separate repositories could also lead to code duplication, which would be time inefficient as well.

However, to still give a healthy sense of isolation between these two subsystems, the branches branching out from the main branch, will in practice function as their own separate main branches. “cr-gui” is the “main” branch for GUI development, while “cr-sen” is the “main” branch for sensor development. Underneath these branches are “-feat-”

(features), and below that “-task-”. As shown in the image above, the “department-prefix” (i.e., “gui-” and “sen-”) is for clarity and separating the folder that Azure Repos make in Git.

Still, there is no guarantee that an eventual merge of cr-gui and cr-sen will be seamless just because they work as expected by themselves: Passing tests and verification in isolation is no guarantee for flawless integration. Thus, an additional branch (cr-test) was added in between the root branch and the respective GUI and sensor branches (cr-sen and cr-gui). The merging is now tested in cr-test before it potentially moves up to the root of the Git tree. Another safety protocol that we have setup in Azure Repos is that any merge to cr-gui/cr-sen and above, requires the pull requests to be reviewed and approved by everyone on Team software before completing merge. This enforces synchronization between the development team and reduces risk of undesirable code going unnoticed. It has also been unanimously agreed that no one is allowed to complete a pull requests that contains merge conflicts (i.e., an event where “Git cannot automatically resolve code differences between two commits”); those must be resolved before completing merge.

5.14 Choosing GUI Library

DØ|TL

In a meeting with ASE, it was highlighted that several employees primarily utilize Python as their main programming language. Given the ease of learning and understanding Python compared to other languages, it was deemed desirable to adhere to the PEP 8 – Style Guide for Python Code [29], a convention widely adopted by Python programmers. The objective was to select a suitable GUI library that aligns with these preferences and requirements.

5.14.1 Tkinter

Tkinter commonly comes bundled with Python, uses Tk and is Python’s standard GUI framework [30]. It emerged as the preferred choice for developing graphical user interfaces (GUIs) due to its alignment with our objectives and the following key features:

- **Standard Library Integration:** Tkinter is seamlessly integrated into Python’s standard library, making it readily accessible and eliminating the need for additional installations.
- **Open Source and Freely Available:** Tkinter is open source and freely available [31], meeting the requirement outlined during discussions with ASE.
- **Basic GUI Components:** Tkinter offers a comprehensive set of basic features for creating windows, dialogs, buttons, and menus, making it suitable for rapid prototyping and development.

5.14.2 Comparison with Other GUI Libraries and Frameworks

While other options such as PyQt and Streamlit were considered, Tkinter emerged as the preferred choice for the following reasons:

PyQt Considerations

- **Feature-Rich:** PyQt, leveraging the Qt framework, offers advanced features and greater flexibility compared to Tkinter[32].
- **Licensing Complexity:** PyQt is available under both GPL and commercial licenses [33], introducing licensing complexities that Tkinter avoids [31].
- **Uncertainty of Complexity:** Despite its advantages, PyQt's complexity and uncertainty regarding its long-term suitability for our project led us to prioritize Tkinter's simplicity and stability.

Streamlit Considerations

- **Web-Based Nature:** Streamlit offers easy-to-use features for web application development but requires internet connectivity and uptime on webservers, which may not align with offline usage requirements [34].
- **Relative Newness:** Streamlit's recent release in 2019 [35] contrasts with Tkinter's longstanding presence since Python 1.0 in 1991 [36], suggesting Tkinter's maturity and longevity, which strengthens the fulfillment of the **Stakeholder Requirement "6.** The entire system should be made with longevity in mind. This applies to, but are not limited to: (c) Software implementations" 3.1.

5.14.3 Conclusion

In conclusion, Tkinter emerged as the preferred choice for GUI development due to its simplicity, stability, and alignment with the project's requirements. While other options offer advanced features and ease of use in certain contexts, Tkinter's robustness and offline capability make it a reliable choice for our current needs. Subsequent sections of this report will delve into the advanced GUI functionality achievable through Tkinter's simplicity, demonstrating its suitability for our project despite its basic features.

Note: How Tkinter is used is described in B.1.

6 Design Phase

STB|STB

6.1 Design process

The chosen design process is based on developing a frame and the antenna release mechanisms on the C.R.A.F.T-Sat. In order to properly understand how the design of the frame and antenna release components is developed, a engineering design process diagram has been made, fig 19.

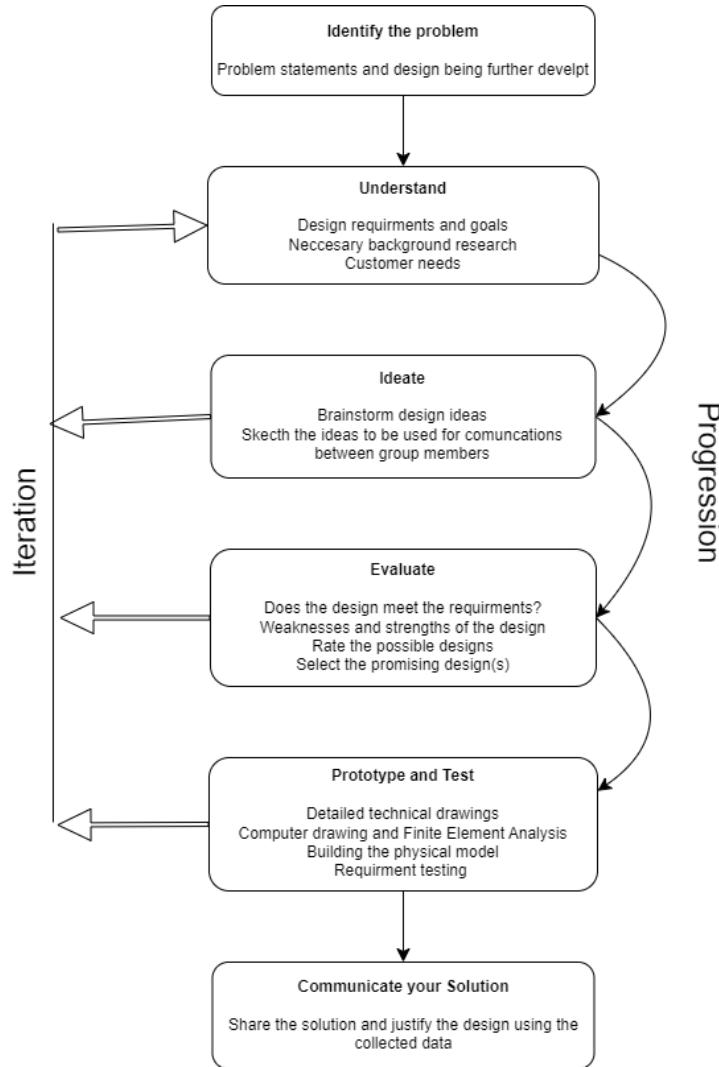


Figure 19: Engineering design process diagram. | STB

6.2 Creation of the C.R.A.F.T-Sat frame

During the planning on the creation of the frame, several thoughts about idealizing the frame fell down. Therefore, to find the most optimal solutions, we had to define the problem to use it for idealizing a start solution for the given task. Several methods has been used to idealize the solution possibilities for the given problem. Those methods includes brainstorming and intern communication between the group members.

STB|STB

6.2.1 Brainstorming - C.R.A.F.T-Sat frame

Before we start with the prototype design of the C.R.A.F.T-Sat frame, some points had to be considered. The chosen points is defined in fig 20, and was noted in a session of brainstorming about what we should be described before CAD modeling of a prototype.

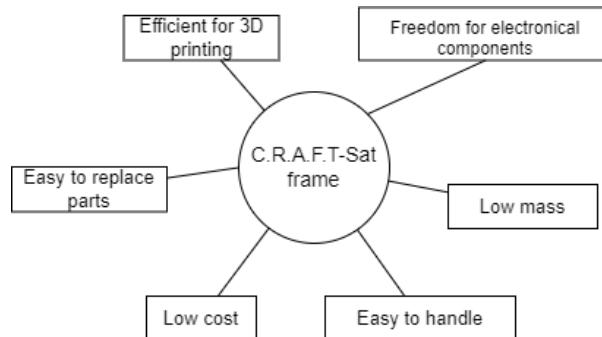


Figure 20: Brainstorming mind map of the frame. | STB

One of the most important topic is the combination between the electronical components and the frame itself. This topic is therefore leading to the freedom for the electronical components and the possibility of changes inside the frame and how we can combine the parts. This part is leading to the efficiency of 3D printing as a very important topic. This is because it gives prototypes which will be used to further investigate if there is any need of improvement in the design and bachelor assignment is describing the production of the frame is 3D printing. By using 3D-printing and efficiency, the topic; easy to replace parts and low cost, plays an important role. Those roles is describing that since we can 3D-print components, they still have to be in the state where the cost of production is not too high due to the advanced design and it has to be at a stage where it isn't too advanced to assemble parts together.

One of the requirements describes that the total mass of the C.R.A.F.T-Sat can't be higher than 500 grams. This gives limitations about the freedom of the design and gives the point; easy to handle, a big meaning.

6.3 Creation of the antenna release mechanism

The theme of creating a antenna release mechanism gives incredible many thoughts about optimal solutions. But, to find the most optimal solutions, we had to define the problem to use it for idealizing a start solution for the given task. Several methods has been used to idealize the solution possibilities for the given problem. hose methods includes brainstorming and intern communication.

STB|STB

6.3.1 Brainstorming - antenna release mechanism

Before we start with the prototype design of the antenna and it's release mechanism, some points had to be considered. The chosen points is defined in fig 21, and was noted in a session of brainstorming about what we should be described before CAD modeling of a prototype.

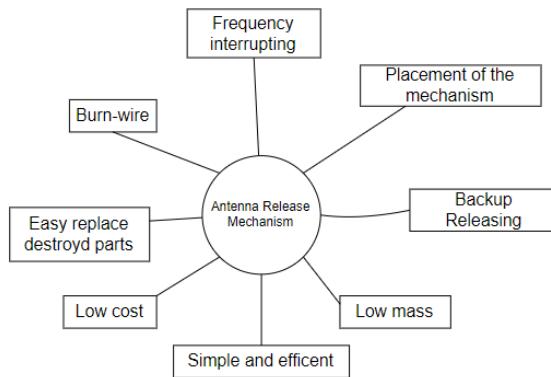


Figure 21: Brainstorming mind map of the antenna release mechanism. | STB

Interrupts on the frequency takes place at more than one reason. Those reasons can be defined as the self-interrupts and external interrupts. The self-interrupts could be based on what material used on components surrounding the antenna and damage the antenna or antenna release has been exposed to. External interrupts is based on factors such as suddenly interrupts with the signals between the ground station and C.R.A.F.T-Sat.

Further, the placement of the antenna release is a important factor. This is because of the creation in mass stability in the overall design and the signal-waves sent to the ground station. To have a most efficient antenna release, a backup releasing system is important. This system is based on obstacles and solutions off how the obstacles can be solved. It's off course other factors in the backup systems which can be more or less sufficient, but in the start stage this is what's expected later to be solved.

Update: Due to suddenly changes in the design, the backup releasing systems was more related to secure that the antenna will release when the burn-wire is used.

Since it's required that the amount of total mass of the C.R.A.F.T-Sat is not more than 500 grams, the components used for the release mechanism has to have a low mass. This gives limitations in what material to be used and the amount of components because it can result in a higher total mass. This brings up the important stage of being simple and efficient. Chosen points in being efficient and simple is:

- Less amount of advanced components.
- Simple design to secure the solutions for of obstacles between the stages not-released antenna and released antenna.
- A simple, less mass, system to control when the antenna should be released.

Those points results in how we can design this system including a low cost, easy replacement of destroyed parts, and burn-wire mechanism.

6.4 Evaluation & selection of the antenna release mechanism & C.R.A.F.T-Sat frame

After the brainstorming and intern communications, alternatives for designs has been made. The first topic is the frame, and it's alternatives of assembling and disassembling. This topic is important due to how we will input the components; teensy and breadboards, and all of the written brainstorming points. The start of the evaluation and selection will be on the frame and from the the antenna release mechanism.

All of the alternatives for the frame is not including the placement of the antenna release mechanism and are just considering how to mount the frame together, and other relative topics around just the frame. The first alternative, fig 22 includes four plates, where we have a big capability to manufacture simple parts few parts, and has the possibility to keep some parts equal which improves how effective the production time will be, and will result with a low cost of 3D printing. The frame itself will give the possibility to have a reduced mass, but a disadvantage is that it will require more working-time to design an assembly which has an appropriate stiffness.

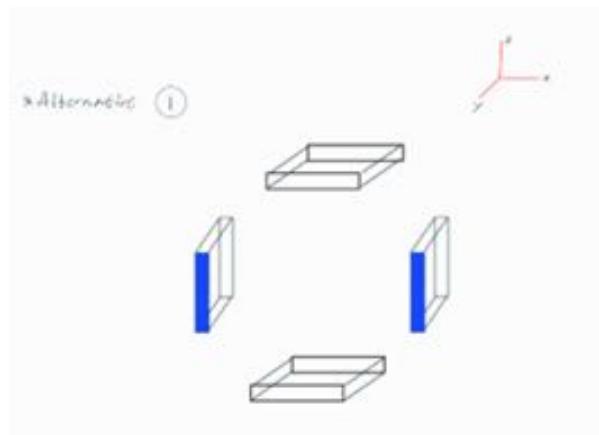


Figure 22: Design alternative 1 | STB

The second alternative, fig 23 includes three parts, where it has the purpose being more all-in on 3D print, and the only other components is the bottom and top part designed to keep all electronical components inside the middle printed part. This frame design gives the possibility to reduce mass, and has a lower requirement of the amount of mounting places on the frame itself. One of the disadvantages in this design is the production time of the middle component because it will require more complex production process of 3D-printing.

The last alternative, fig 24 is a simple two parts design, where the purpose is to close the components inside a "box" where it will be in advantage of securing the electronical components and stiffness of the frame. The big disadvantage will be the mounting process with all of the electronical components, and the mounting of the frame with all of the requirements due to fit inside the CubeSat dispenser and be as simple as possible.



Figure 23: Design alternative 2 | STB

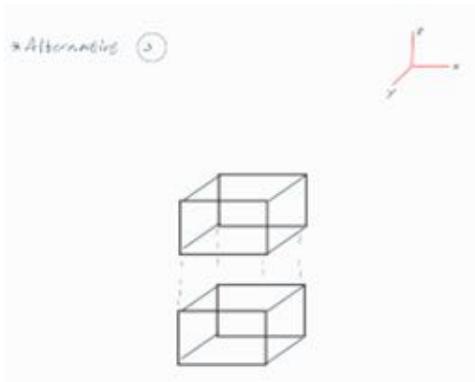


Figure 24: Design alternative 3 | STB

From this alternatives, deciding the design relative to our purpose is done by using the pugh-matrix. To use this chosen pugh-matrix, a description about the earlier solution had to be done. This description is relative to a CanSat design, and is written as:

The current solution is a CanSat with a frame design equal a simple soda-can.

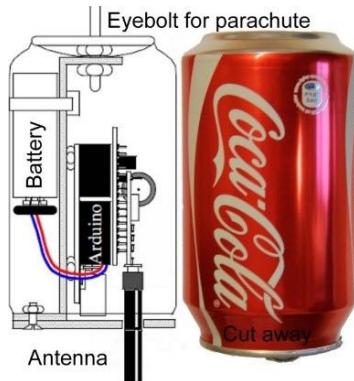


Figure 25: [21]

The four chosen criteria is decided as the most important factors controlling the design. To decide the pugh-matrix values four questions has been made for comparing. Those questions are:

1. Does the new frame cost less than current solution.
2. Is it easier to maintenance the new frame compared with the current solution.
3. Does it have a better usability than the current solution.
4. Does it take less time to produce the new frame, compared with the current solution.

To further decide which alternatives in the pugh-matrix is most optimal, values had to be made used for comparing between the criteria, alternative and current solution. The used values and description of each value is:

- Better than current solution, value 1
- Equal current solution, value 0
- Worse than current solution, value -1

Criteria	Current solution	Alt. 1	Alt. 2	Alt. 3
Cost	0	-1	-1	-1
Maintenance	0	1	0	-1
Usability	0	1	-1	0
Production - time	0	1	-1	0
Result	2	-3	-2	

Table 20: Pugh Matrix: C.R.A.F.T–Sat frame | STB

The result from the table 20 is concluding that fig 22 is the most appropriate design for our purpose. But, an important fact is that this is just a rough design and will be further developed during the sprint periods.

The placement of and how the antenna release mechanism is important. Therefore, those are the topics in the upcoming alternatives. We will first mention that burn-wire mechanism; description of this mechanism is in appendix H, has been accepted from ASE and is a simple alternative to implement into the design. We have also decided to use a measuring tape as the alternative antenna in the stage of developing the release mechanism.

The first alternative is based on a design where the dipole-antenna are placed inside the frame with double-wire holding it in place during the period when the antenna is not launched. When the period of antenna launch starts, the circuit for the burn-wire will start and the sealing will be opened by a physical mechanism controlled by the burn-wire. When the sealing open, the dipole-antenna will be released.

The second alternative is based on a design where the dipole-antenna is placed inside the frame and has some equal systems as the first alternative. The difference is that when the burn-wire system is used, the dipole-antenna will be released and self-open the sealing.

The third and last alternative is an antenna which has been mounted around the frame where the release mechanism is still using burn-wire method, but the antenna and circuit for burn-wire has a bigger possibility to fault due to other components and space.

From this alternatives, deciding the design relative to our purpose is done by using the pugh-matrix. To use this chosen pugh-matric, a description about the earlier solution had to be done. This description is relative to a CanSat design, and is written as:

The current solution is a simple UHF – Antenna placed outside of the soda can.

The four chosen criteria is decided as the most important factors controlling the design. To decide the values four questions has been made for comparing. Those questions are:

1. Does the new mechanism cost less than current solution.
2. Is it easier to maintenance the new mechanism compared with the current solution.
3. Does it have a better usability than the current solution.
4. Is the efficiency better than the current solution.

To further decide which alternatives in the pugh-matrix is most optimal, values had to be made used for comparing between the criteria, alternative and current solution. The used values and description of each value is:

- Better than current solution, value 1
- Equal current solution, value 0
- Worse than current solution, value -1

Criteria	Current solution	Alt. 1	Alt. 2	Alt. 3
Cost	0	-1	-1	0
Maintenance	0	-1	-1	1
Usability	0	1	1	-1
Efficiency	0	1	1	-1
	Result	0	0	-1

Table 21: Pugh Matrix: C.R.A.F.T-Sat antenna release mechanism | STB

The result from the table 21 is concluding first and second alternatives as the possible designs to be further developed. But in comparing between the first and second alternatives, the first is most optimal because it won't require investigation about if the force from the antenna is capable of open the sealing, with the effect from material type, and other factors affecting the antenna pushing the sealing open. The total design will therefore include a frame design equal alternative 1 and the antenna release equal alternative 1.

STB|

6.5 Timeline - iterations

The iteration work in developing the C.R.A.F.T-Sat frame and antenna release mechanism related to mechanical work was divided into three timeline-iterations and the final design. The timeline is shown in fig 26 including the dates/states where the iteration starts/ends and when the final design start and end. A further description about the work from iteration 1, 2 & 3 is described in appendix Z.

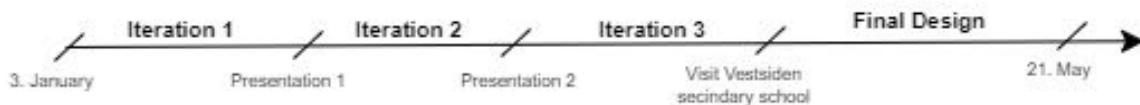


Figure 26: Iterations & final design

6.6 Battery Heat Flux

RE|GALK

Determining the **heat flux** of the battery is a crucial parameter for successful FEA simulation. Heat flux represents the energy flow per unit area, as shown in Equation (2)([37]):

$$\text{Heat Flux} = \frac{\text{Power}}{\text{Total Surface Area}} \quad (2)$$

where,

- Power is expressed in W (Watt)
- Total surface area is expressed in m^2
- Heat Flux is expressed in W/m^2

The dimensions of the battery are 62mmx32mmx18 mm:

The selected WLToys 7.4V 1500mAh LiPo battery for C.R.A.F.T-Sat has a voltage value of 7.4V and current of 0.15 A. Hence, the power can be calculated using Equation (3)

$$\text{Power}(W) = \text{Voltage}(V) \times \text{Current}(A) \quad (3)$$

$$\Rightarrow \text{Power} = 7.4 \text{ V} \times 0.15 \text{ A} = 1.11 \text{ W}$$

The dimensions of the battery are 62mmx32mmx18 mm, we select length $l = 62$, width $w = 32$, and height $h = 18$, the surface area can be calculated using Equation (4):

$$\text{Surface Area} = 2(lw + lh + wh) \quad (4)$$

$$\Rightarrow \text{Surface Area} = 2((62 \times 32) + (62 \times 18) + (32 \times 18)) \text{ mm}^2 = 7352 \text{ mm}^2$$

By converting the value from mm^2 to m^2 by multiplying by 1×10^{-6} , we get:

$$\text{Surface Area} = 0.00735 \text{ m}^2$$

The Heat Flux or Average Heat Flux: The heat flux or average heat flux can be determined using Equation (2)

$$\text{Average Heat Flux} = \frac{1.11 \text{ W}}{0.007352 \text{ m}^2} = 151 \text{ W/m}^2$$

Surface Specific Heat Calculations After determining the average heat flux of the battery, we can now calculate the heat estimated heat flux in each side of the battery.

Top and Bottom Surface of the battery (62mm x 32 mm):

$$\text{Area} = 62 \times 32 \text{ mm}^2 \times 10^{-6} = 0.001984 \text{ m}^2$$

$$\Rightarrow \text{Heat Flux} = \frac{1.11 \text{ W}}{2 \times 0.001984 \text{ m}^2} = 280 \text{ W/m}^2$$

Side surfaces (62mm x 18mm):

$$Area = 62 \times 18 \text{ mm}^2 \times 10^{-6} = 1.116 \cdot 10^{-3} \text{ m}^2$$

$$\Rightarrow \text{Heat Flux} = \frac{1.11 \text{ W}}{2 \times 0.001116 \text{ m}^2} = 497 \text{ W/m}^2$$

Side Surfaces (32mm x 18 mm):

$$Area = 32 \times 18 \text{ mm}^2 \times 10^{-6} = 0.00576 \text{ m}^2$$

$$\Rightarrow \text{Heat Flux} = \frac{1.11 \text{ W}}{2 \times 0.00576 \text{ m}^2} = 96.35 \text{ W/m}^2$$

6.7 Design Consideration: Electromagnetic Interference (EMI) and Thermal Management

Electromagnetic Interference (EMI) and thermal management are both crucial factors in PCB design, especially in compact systems like CanSats and C.R.A.F.T-Sat. These factors (in addition to components dimensions) are considered in our iterations for component placement in PCB for CanSat and C.R.A.F.T-Sat.

6.7.1 Electromagnetic Interference (EMI)

EMI refers to a disruption that impacts the electrical circuit as a result of either electromagnetic induction or electromagnetic radiation emitted from an external source[38].

Components in CanSat that can emit the EMI:

- **Radio Communication Module and Microcontroller:** These components can emit EMI that can negatively affect the performance of sensitive analog components. For example, a Teensy 4.1 microcontroller could rapidly turn on its internal transistors, creating a wide range of harmonics that may interfere with neighboring traces or components.
- The **GY-91**: gyroscope module includes an accelerometer, gyroscope, magnetometer, and barometer. It is important to note that the **GY-91** device may be susceptible to electromagnetic interference (EMI), leading to inaccurate readings or temporary operational problems. This susceptibility is especially significant for the magnetometer, as it measures magnetic fields and may be influenced by magnetic interference from surrounding components.
- **Antenna:** The antenna can also capture EMI from the other components, which could degrade the quality of the received or transmitted signal. Similarly, the antenna can also radiate Radio Frequency (RF) energy that could interfere with nearby sensitive components.

6.7.2 Thermal Management

Thermal management is another essential aspect to consider. In a small PCB like that of a CanSat, components need to be closely packed, leading to a higher amount of heat produced per area. This heat generation can lead to inaccuracies in the measurements taken by the components in the CanSat or, in more severe cases, cause them to malfunction[39].

In terms of components in CanSat:

- **Voltage Regulators:** These components can generate significant heat when they regulate the voltage, especially when they supply a large amount of current or when the input voltage is much higher than the output voltage.

-
- **Negative Temperature Coefficient (NTC) Temperature Sensor:** Placing a NTC sensor near components that generate heat could influence the temperature readings provided by the NTC sensor.

6.7.3 Mitigation Techniques

The strategies used by C.R.A.F.T to mitigate the issues related to EMI and heat are centered on aspects such as positioning components, their separation, grounding, thermal vias, and airflow.

For EMI Mitigation:

- **Separation:** Sensitive components are placed away from components that generate electromagnetic energy.
- **Grounding:** A well-designed ground plane can absorb and redirect EMI away from sensitive components.

For Mitigation of the Thermal Problems:

- **Proper Placement:** Components that generate significant heat should be spaced out and not placed near temperature sensitive components such as the NTC temperature sensor.
- **Thermal vias:** The vias, which are plated through holes, are positioned below the heat sources to create a low thermal resistance pathway from the upper copper layer to the bottom of the PCB[40].
- **Airflow:** If possible, we ensure that there is enough airflow around PCB, which can help with cooling.

6.8 Prework for Component Placement: EMI Theory and Simulation

RE|GALK

In today's technological landscape, with circuit boards and components shrinking in size yet increasing in speed, the significance of Electromagnetic Compatibility (EMC) has reached unprecedented levels. The potential for interference among devices has become critical, as it can significantly impair performance and even result in system failure if thorough simulations and tests are not conducted accurately. The complex interaction of electromagnetic fields at gigahertz frequencies requires an advanced understanding to maintain EMC in small but densely populated electronic environments.

In order to efficiently reduce EMI, it is essential to have a basic understanding of the propagation of **electric fields (E-fields)** and **magnetic fields (H-fields)**. This transmission, known as **electromagnetic radiation**[[41]], can affect the functionality of delicate electronic parts such as the GY-91 sensor module and affect the overall reliability of the system. Maxwell's equations play a key role in forecasting and comprehending these phenomena of wave propagation, which can help us to derive the expression or equation

of the electromagnetic waves. Electromagnetic waves are a type of radiation that propagates through space. They originate from the interaction between an electric field and a magnetic field([42]).

In the free space, which means that there are no charges q and no currents, Maxwell's equations can be written in terms of electric field E and magnetic field B as equations (5) to (8)[[43]] show:

$$\vec{\nabla} \cdot \vec{E} = 0 \quad (5)$$

$$\vec{\nabla} \cdot \vec{B} = 0 \quad (6)$$

$$\vec{\nabla} \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \quad (7)$$

$$\vec{\nabla} \cdot \vec{B} = \mu_0 \epsilon_0 \left(\frac{\partial \vec{E}}{\partial t} \right) \quad (8)$$

where $\nabla \cdot$ and $\nabla \times$ are the **divergence** and **curl** operators, respectively. The constants μ_0 and ϵ_0 called the permeability of the free space and the permittivity of the free space, respectively.

To derive electromagnetic waves, we need to condense the equations from (5) to (8) into a single equation for electromagnetic waves.

We start by multiplying equation 7 by ∇ :

$$\begin{aligned} \vec{\nabla} \times (\vec{\nabla} \times \vec{E}) &= -\vec{\nabla} \times \frac{\partial \vec{B}}{\partial t} \\ \Rightarrow \vec{\nabla} \times (\vec{\nabla} \times \vec{E}) &= -\frac{\partial \vec{\nabla}}{\partial t} \times \frac{\partial \vec{B}}{\partial t} \\ \Rightarrow \vec{\nabla} \times (\vec{\nabla} \times \vec{E}) &= -\frac{\partial}{\partial t} (\vec{\nabla} \times \vec{B}) \end{aligned} \quad (9)$$

The left side of equation (9) can be calculated using vector identity:

$$\vec{\nabla} \times (\vec{\nabla} \times \vec{E}) = \vec{\nabla} (\vec{\nabla} \cdot \vec{E}) - \nabla^2 \vec{E} \quad (10)$$

Applying Maxwell's equation (5), the equation (10) can be simplified to:

$$\vec{\nabla} \times (\vec{\nabla} \times \vec{E}) = -\nabla^2 \vec{E} \quad (11)$$

Similarly, applying Maxwell's equation (6), the right side of equation (9) can be simplified to the following.

$$-\frac{\partial}{\partial t}(\vec{\nabla} \times \vec{B}) = -\frac{\partial}{\partial t}(\mu_0 \epsilon_0 \frac{\partial \vec{E}}{\partial t}) \quad (12)$$

Based on the expression we get in Equation (11) and (12), we can write:

$$\begin{aligned} -\nabla^2 \vec{E} &= -\frac{\partial}{\partial t}(\mu_0 \epsilon_0 \frac{\partial \vec{E}}{\partial t}) \\ \Rightarrow \nabla^2 \vec{E} &= \frac{\partial}{\partial t} \frac{\partial \vec{E}}{\partial t} \mu_0 \epsilon_0 \\ \Rightarrow \nabla^2 \vec{E} &= \frac{\partial^2 \vec{E}}{\partial t^2} \mu_0 \epsilon_0 \end{aligned} \quad (13)$$

The wave equation is then illustrated in Equation (14)

$$\frac{\partial^2 \vec{E}}{\partial t^2} = \frac{1}{\mu_0 \epsilon_0} \nabla^2 \vec{E} \quad (14)$$

where the speed of light c can be expressed as:

$$c = \frac{1}{\sqrt{\mu_0 \epsilon_0}} \quad (15)$$

Equation (14) can be written as:

$$\frac{\partial^2 \vec{E}}{\partial t^2} = c^2 \nabla^2 \vec{E} \quad (16)$$

The general solution to the Electromagnetic Equation, represented in (16), in one dimension (for electric field \vec{E} along the y-axis and for magnetic field \vec{B} along the z-axis) is:

$$E_y(x, t) = E_{max} \cos(kx - \omega t) \quad (17)$$

$$B_z(x, t) = B_{max} \cos(kx - \omega t) \quad (18)$$

where k and ω represent the wave number and angular frequency, respectively. E_{max} and B_{max} are the amplitude (the highest value) for the electric field and the magnetic field, respectively[[44, pp. 1083- -1087]].

To understand how \vec{E} and \vec{B} propagate on the y and z axes, we developed a MatLab code illustrated in Listing (??)

Using the MATLAB, we obtain a visualization depicting the propagation of the electric field \vec{E} , and magnetic field, \vec{B} , as depicted in Figure (27). This is an animated graph which can be shown in an MP4 file named **Propagation of the Electric and Magnetic Field**. The cross product $\vec{E} \times \vec{B}$ represents the direction of the electromagnetic wave. The cross product $\vec{E} \times \vec{B}$ is also crucial for calculating **Poynting vector**, \vec{S} , which represents the directional energy flux (the rate of energy transfer per unit area) of an electromagnetic field[[45]]. The Poynting vector is expressed as:

$$\vec{S} = \frac{1}{\mu_0}(\vec{E} \times \vec{B}) \quad (19)$$

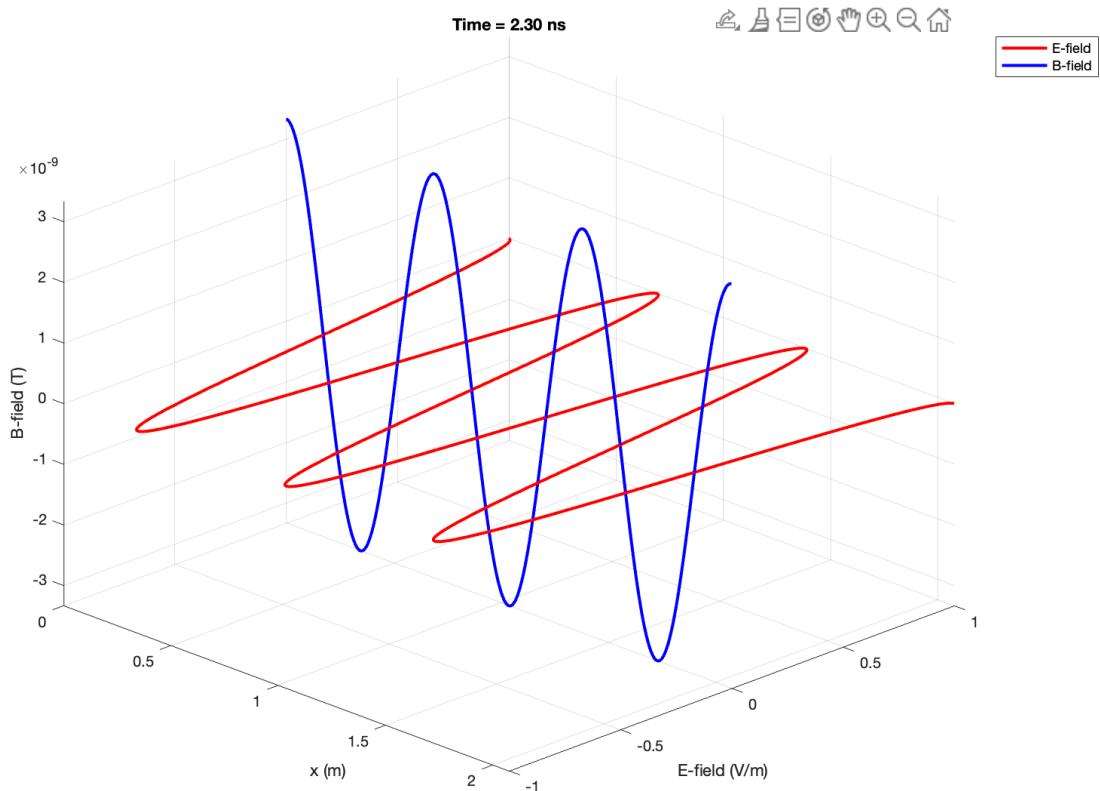


Figure 27: Electric and Magnertic Wave's Propagation

Detailed Analysis of Monopole Antenna Radiation Pattern

In order to gain a more accurate understanding of how the selected antenna for CanSat will emit radiation, it is necessary to investigate the radiation pattern of this particular type of antenna operating at the specified frequency of 434 MHz. The radiation pattern analysis will offer insights into the directions from which the wave will emanate. Conse-

quently, a Monopole antenna was developed using MatLab, and its specific characteristics are outlined in Table (22).

Table 22: Quarter Monopole Antenna's Specification

Parameter	Value
Frequency	$f = 434 \text{ MHz}$
Wave Length	$\lambda = \frac{c}{f} = 0.6912 \text{ m}$
Antenna Length	$\frac{\lambda}{4} = 0.1728 \text{ m}$
Ground Plane Length	$\frac{\lambda}{2} = 0.3456 \text{ m}$
Ground Plane Width	$\frac{\lambda}{2} = 0.3456 \text{ m}$

In addition to the radiation pattern, it is essential to consider two significant parameters, namely the **Azimuthal Plane Pattern (Horizontal)** and the **Elevation Plane Pattern (Vertical)**. The Azimuthal Plane Pattern illustrates how the antenna emits radiation when viewed from above, providing insight into how power is distributed horizontally around its axis. Analyzing an antenna's azimuthal plane pattern is crucial for understanding its horizontal coverage area. On the other hand, the Elevation Plane Pattern displays how the antenna radiates in a vertical plane that encompasses its axis, offering a perspective on the radiation distribution above and below the antenna. The primary purpose of studying the Elevation Plane Pattern is to assess how the antenna emits power vertically[[46]].

A MatLab script has been created as demonstrated in Listing (1) to design a monopole antenna with the frequency and dimensions specified in Table (22). The script generates visualizations of the radiation pattern, Azimuthal Plane Pattern, and Elevation Plane Pattern. The results are shown in Figures (28), (29), (30), and (31), corresponding to the quarter monopole antenna's design, the radiation pattern, the azimuthal plane pattern, and the elevation plane pattern, respectively.

```

1 % Quarter Monopole Antenna Design and Simulating
2
3
4 f = 434e6; % Operating frequency in Hz
5 c = 3e8; % Speed of light in m/s
6 lambda = c / f; % Wavelength in meters
7
8 % Creating a monopole antenna of height lambda/4
9 % The length of a monopole is (typically) a quarter of the
   wavelength, lambda.
10 monopoleAnt = monopole('Height', lambda/4, 'GroundPlaneLength',
    lambda/2, 'GroundPlaneWidth', lambda/2);
11
12 % Showing the Monopole antenna
13
14 figure();
15 show(monopoleAnt);

```

```

16
17 % Radiation Patern for Quarter Monopole Antenna
18 figure();
19 pattern(monopoleAnt, f);
20
21 % Azimuthal Plane Pattern (Horizontal)
22 figure();
23 patternAzimuth(monopoleAnt, f)
24
25 % Elevation Plane Pattern (Vertical)
26 figure();
27 patternElevation(monopoleAnt, f)

```

Listing 1: MatLab's Code for Designing Quarter Monopole Antenna and Illustrating its Radiation Pattern.

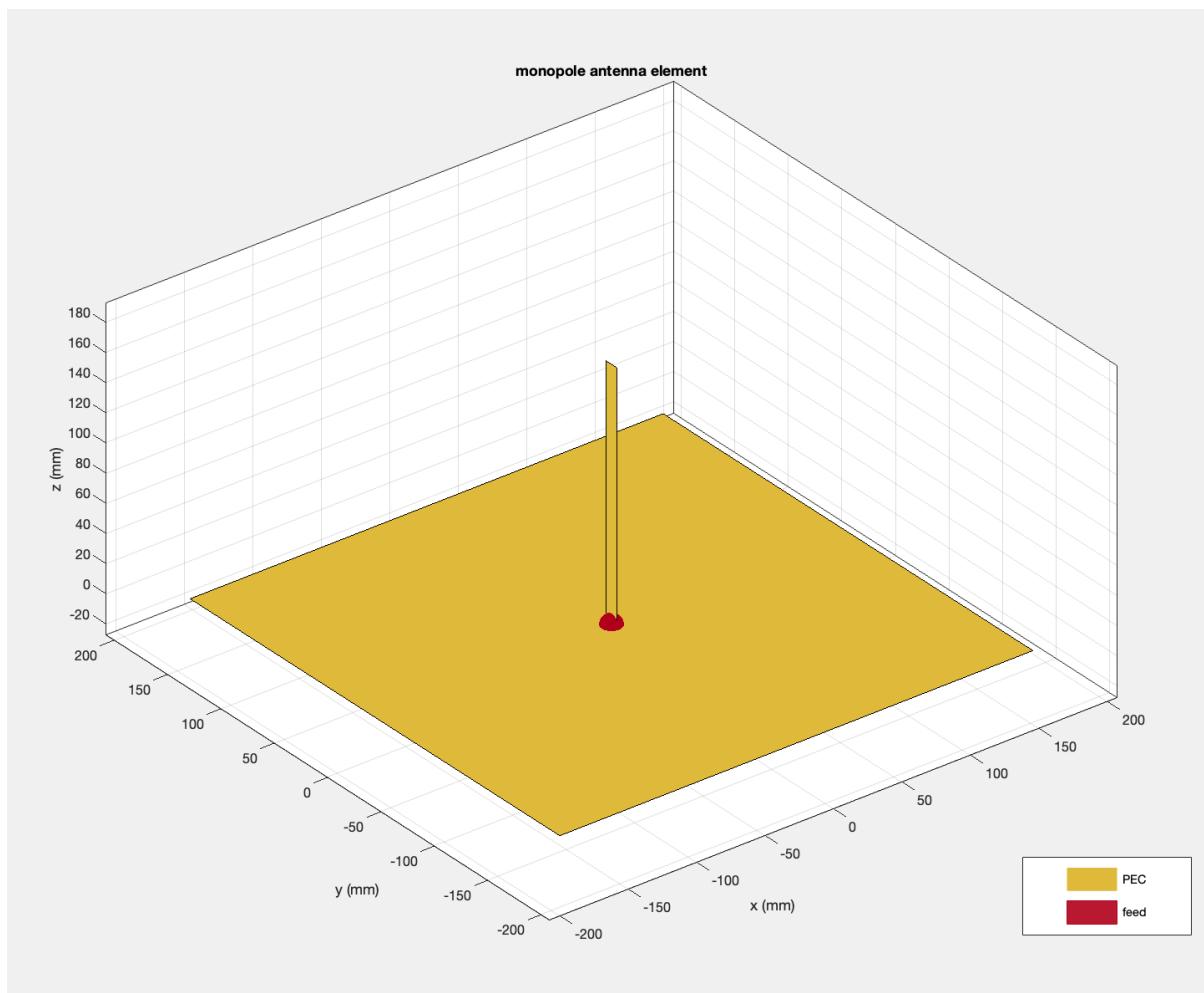


Figure 28: Design of Quarter Monopole Antenna

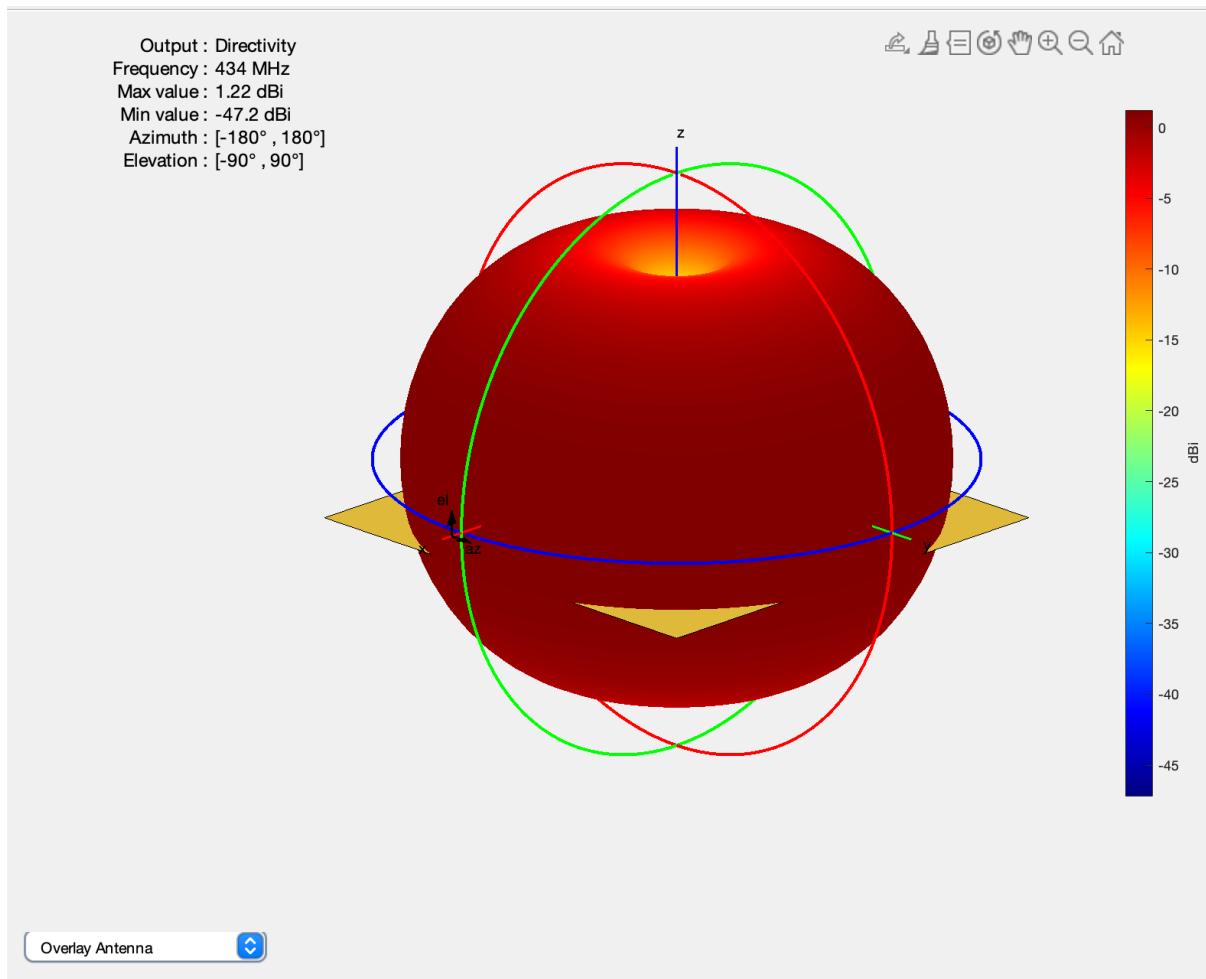


Figure 29: Quarter Monopole Antenna's Radiation Pattern

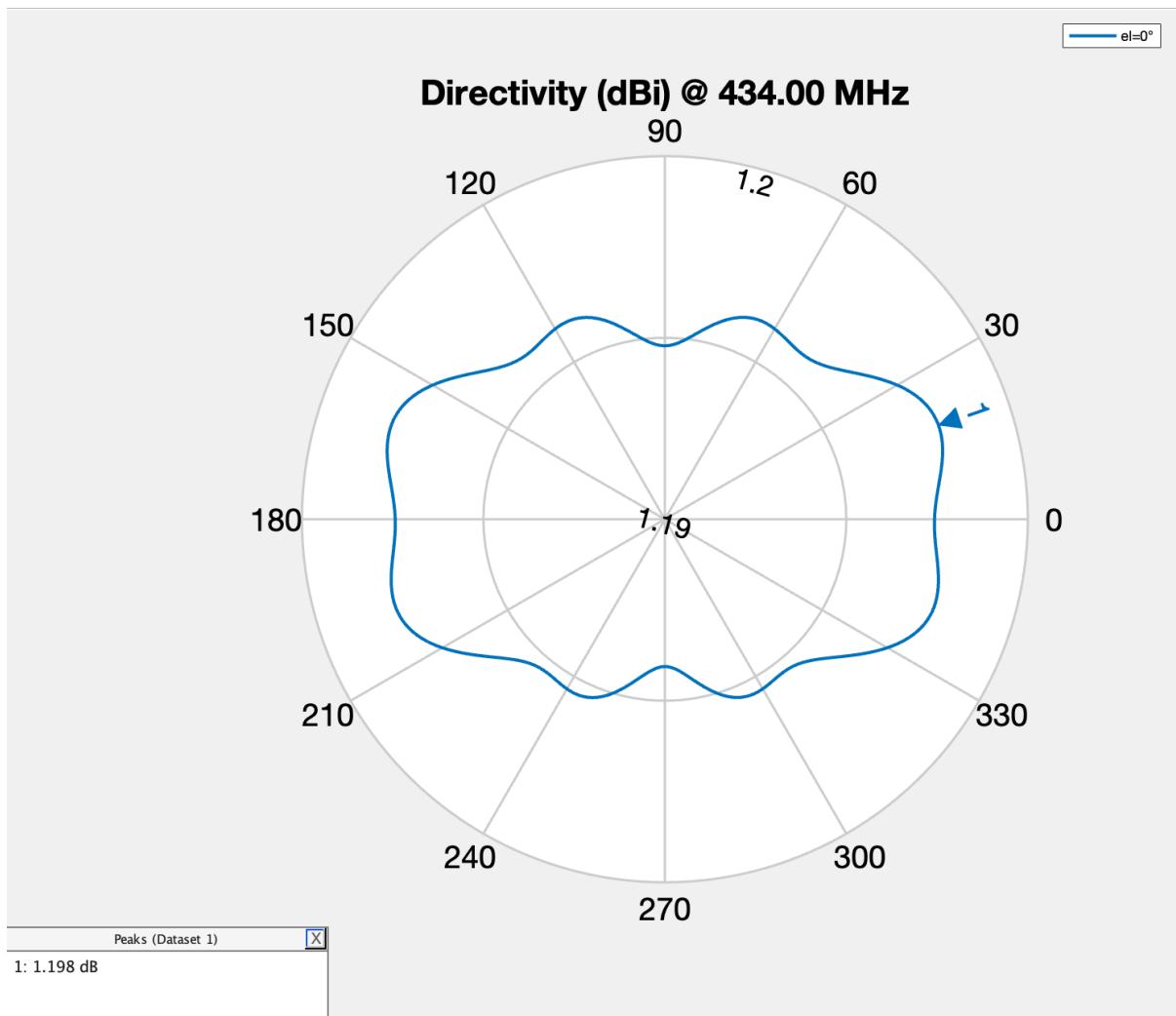


Figure 30: Azimuthal Plane Pattern (Horizontal)

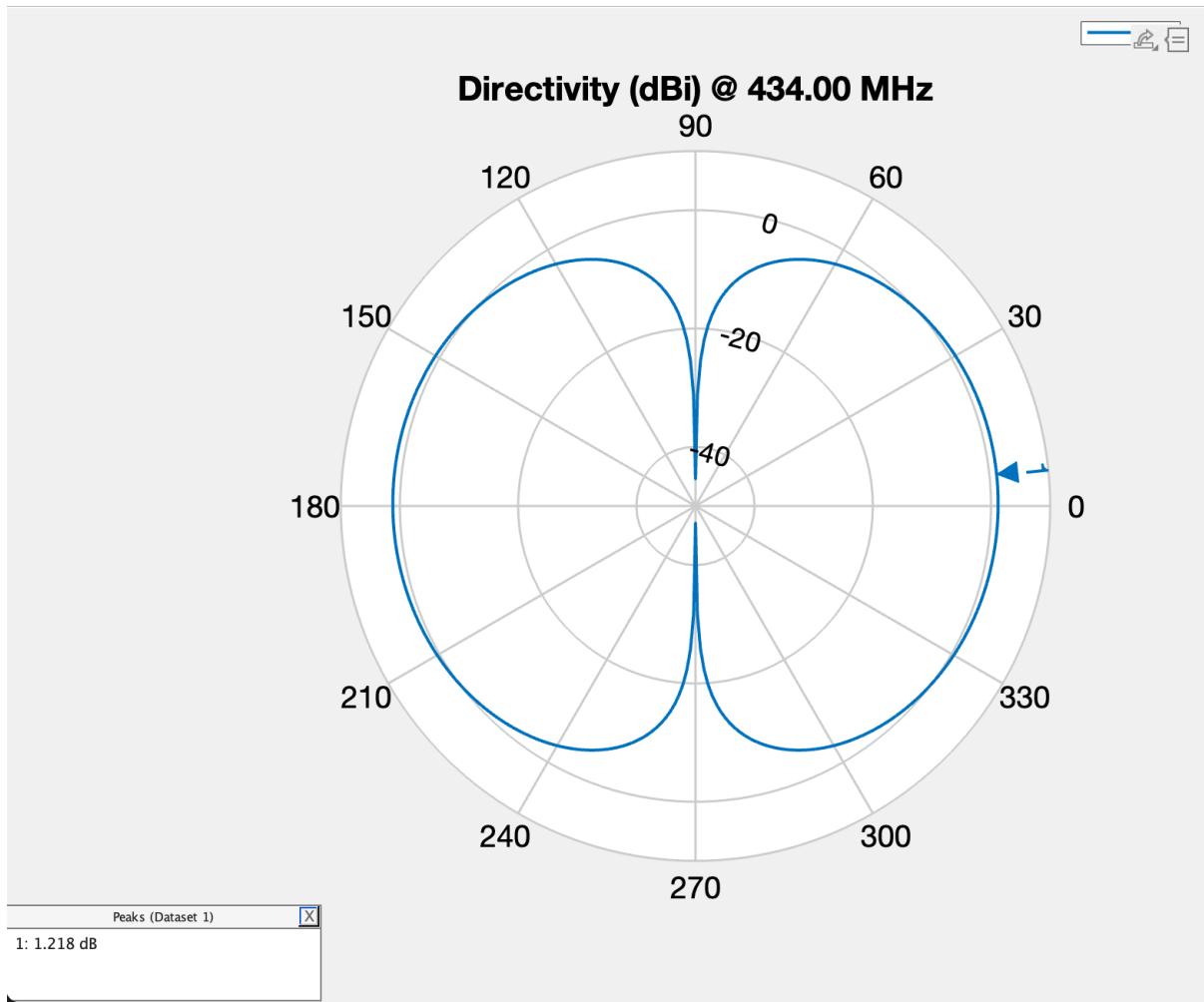


Figure 31: Elevation Plane Pattern (Vertical)

The radiation characteristics of the monopole antenna depicted in Figure (29) are characterized by its omnidirectional propagation. This means that the antenna emits waves with relatively consistent intensity in all directions, which is a common trait of monopole antennas. Despite this omnidirectional property, there is a noticeable decrease in the emitted power towards the zenith and nadir directions. The azimuthal pattern, as illustrated in Figure (30), confirms the omnidirectional radiation of the antenna by showing significant uniformity along the horizontal plane. The main lobes peak symmetrically to the right and left, with a distinct lack of substantial radiation in the directions of 0, 45, 90, 135, 180, 225, 270, and 315 degrees. By aligning the GY-91 Gyroscope with the vertical axis of the antenna, it is possible to minimize electromagnetic interference (EMI) effects since the radiation intensity along the antenna's axis is reduced. This placement strategy utilizes the area where the antenna's radiation pattern is weakest to shield the gyroscope from strong electromagnetic fields that could potentially introduce inaccuracies in the measurements.

Although the elevation pattern, shown in Figure (31), may be of lesser importance in CanSat projects where components are on the same PCB, it offers valuable insight for

scenarios where vertical separation is possible. It indicates that the radiation is least intense above and below the antenna, suggesting that components placed in these directions will experience lower exposure to the antenna's radiation.

Detailed Analysis of Dipole Antenna Radiation Pattern

Although the dipole antenna in C.R.A.F.T-Sat is not as crucial as the monopole antenna in CanSat due to the greater distance between the placement of PCB and the mounting of the antenna, the situation is mitigated because PCB with its components is placed outside the near-field region, which is the small area nearest to the antenna. Furthermore, the dipole antenna pattern closely follows that of the monopole antenna, allowing the comprehensive analysis of the monopole antenna to be relevant for the dipole antenna as well.

Figure (32) shows a dipole antenna designed using MatLab, while Figures (33), (34), and (35) display the dipole antenna's radiation pattern, Azimuthal Plane Pattern, and Elevation Plane pattern, respectively. A comparison of Figures (33), (34), and (35) with Figures (29), (30), and (31) reveals that the directivity values and patterns of the dipole and monopole antennas are nearly identical, with only slight variations in directivity values and pattern of the Azimuthal plan. This similarity confirms that the research conducted for CanSat with a monopole antenna is valid for C.R.A.F.T-Sat with a dipole antenna as well.

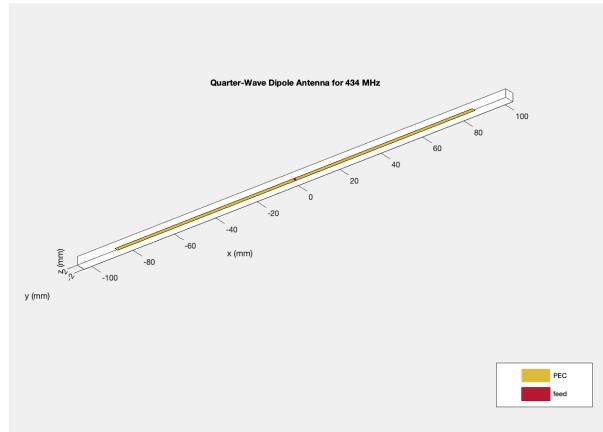


Figure 32: Design of Half-Wave Dipole Antenna

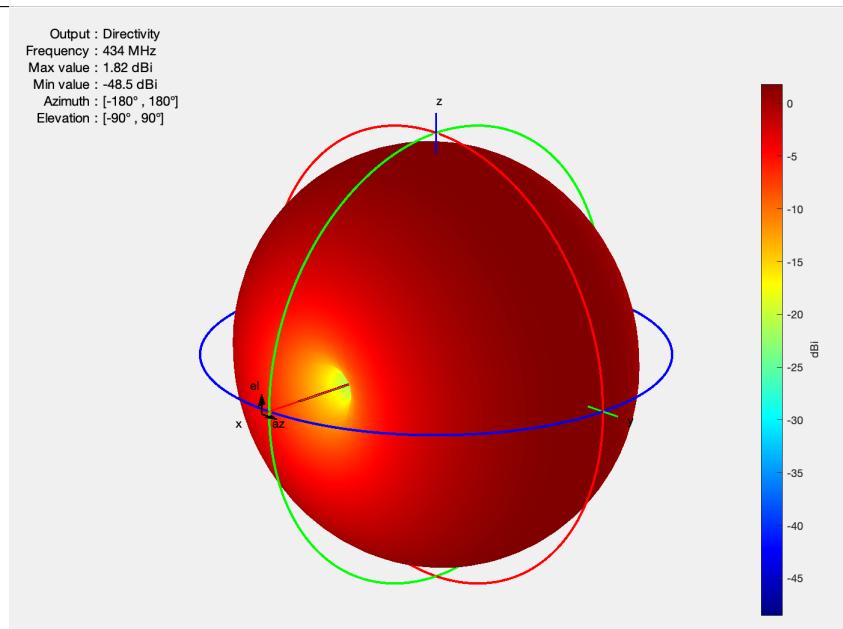


Figure 33: Dipole Antenna's Radiation Pattern

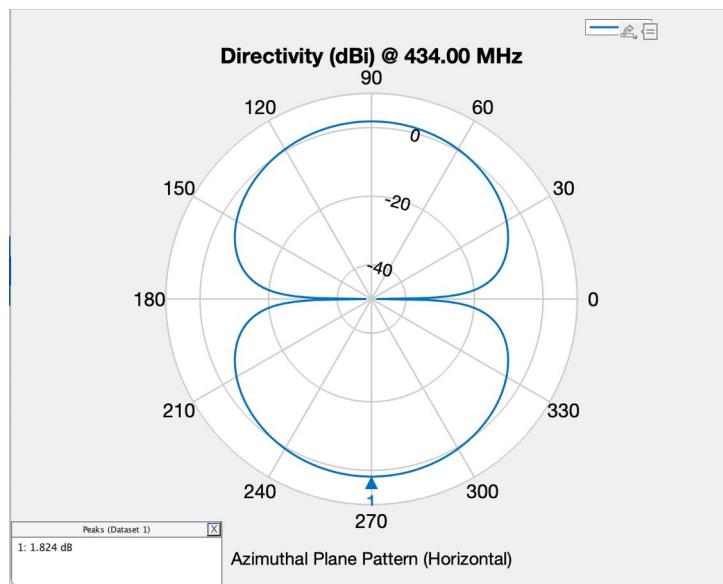


Figure 34: Dipole Antenna: Azimuthal Plane Pattern (Horizontal)

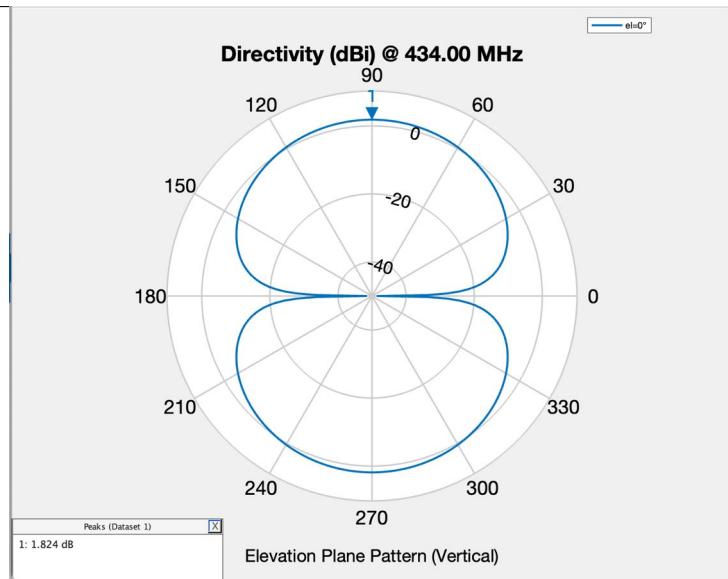


Figure 35: Dipole Antenna: Elevation Plane Pattern (Vertical)

6.9 Voltage Regulator: Thermal Distribution

The thermal distribution generated by the linear 5V and 3.3V voltage regulators is a crucial aspect in order to manage the thermal issue and to ensure that the voltage regulators' temperature does not affect the measured temperature by the NTC temperature sensor. The heat generated is due to the dissipated power, which can be calculated by using Equation (20)([47]):

$$P_{loss} = (V_{in} - V_{out}) \cdot (I_{out}) \quad (20)$$

where:

- P_{loss} is the power dissipated.
- V_{in} is the voltage provided by the battery and is equal to 7.4V.
- V_{out} is the voltage putout value from the voltage regulators, which is 5V and 3.3V for 5V and 3.3V regulator, respectively.

Power Dissipation for 5V Voltage Regulator

The power dissipation of the 5V voltage regulator can be calculated using the formula in Equation (20)

$$P_{loss} = (7.4V - 5V) \cdot (0.107A)$$

$$\Rightarrow P_{loss} = 0.2568 \text{ W}$$

Power Dissipation for 3.3V Voltage Regulator

Using the same formula shown in Equation (20) we can calculate the power dissipation of the 3.3V voltage regulator.

$$P_{loss} = (7.4V - 3.3V) \cdot (0.107A)$$

$$\Rightarrow P_{loss} = 0.4387 \text{ W}$$

The power dissipation calculations for the 5V and 3.3V voltage regulators indicate low values, suggesting that they do not dissipate a significant thermal risk. However, for safety reasons, these regulators are placed in the bottom layer of PCB, as far away as possible from the NTC temperature sensor. An infrared thermometer was used to determine the heat output of the voltage regulators, and the findings are shown in Figure (1) and (2) for the 5V and 3.3V voltage regulator, respectively, after one hour of operation.

6.10 PCBs Component Placement

RE|GALK

Optimal positioning of components is essential to reduce EMI, control heat spread, and improve the performance of the entire system. The arrangement of the components on the shield circuit board is determined by research on EMI, heat management on voltage regulators, and the dimensions of the components.

6.10.1 Component Placement

1. Teensy 4.1 Microcontroller

- **Placement:** Almost central location on the PCB.
- **Justification:** This placement allows for shorter trace distances to other components and sensors, reducing the potential for noise and signal degradation. A central position facilitates easier and more efficient routing of connection and use less numbers of vias, enhancing system overall performance.

2. RFM96 Radio

- **Placement:** Close to SMA antenna connector, located at the bottom of top layer PCB.
- **Justification:** Minimizing the trace length between the **RFM96 radio** and its antenna reduces signal loss and amount of EMI. Placing the antenna at the bottom of toplayer PCB helps in distancing it far from other sensitive components, as **GY-91 Sensor Module**. This placement reduces the risk of EMI that can affect these sensors.

3. GY-91 Sensor Module

- Aligned in the same vertical line with the antenna on the top layer of PCB.
- **Justification:** In this direction, the antenna exhibits the least directivity strength as Figure (31) shows. This is an advantageous position for the **GY-91**, which includes sensitive sensors like **magnetometer** and **accelerometer**. This placement ensures minimal EMI impact, maintaining the accuracy and reliability of the sensor readings.

4. Voltage Regulators

- **Placement:** On the bottom layer of the PCB.
- **Justification:** While these linear regulators are robust against thermal issues, they can generate heat in some scenarios. They are, therefore, placed solely in the bottom side of PCB to prevent the heat from affecting components placed on the top side, particularly sensitive elements as NTC sensor. In addition to that, several thermal vias are placed nearby the voltage regulators to improve heat dissipation.

5. NTC Temperature Sensor

-
- **Placement:** Positioned at the bottom part of the upper layer, maximizing the distance from the voltage regulators.
 - **Justification:** To ensure that the temperature readings are not affected by the heat generated by the voltage regulators, the NTC sensor is positioned far from these components. This setup ensures more accurate temperature readings by mitigating the impact of internal heat sources.

6. Buzzer

- **Placemet:** On the top of the top layer of PCB.
- **Justification:** Utilizes the available free space efficiently and facilitates easy integration within the C.R.A.F.T-Sat structure.

Furthermore, it is crucial to note that thermal vias surround all sensors to enhance heat dissipation across the **shield circuit board**. The maintenance of the weight equilibrium of C.R.A.F.T-Sat with all its components was also taken into account by the placement of the components.

6.11 Shield circuit board

6.11.1 Overview

The main shield for the CanSat and C.R.A.F.T-Sat is designed for the Teensy 4.1 micro controller with break out pins for easy expandability and pin accessibility in mind. It is made as a two sided PCB to keep cost low and troubleshooting easy.

6.11.2 Dimensions of the circuit board

The circuit board has a width of 62 mm and a height of 70 mm. We operate within the CanSat competition limits of 115 mm by 66 mm in diameter, and within the 1U CubeSat. The thickness of the PCB is 1,6 mm. The mounting holes are M3 sized and are spaced 4mm at the center from each side of the board edge.

Table 23: Circuit board dimensions

Board outline height	70 mm
Board outline width	62 mm
Hole spacing height	62 mm
Hole spacing width	54 mm
Hole size	M3
Board thickness	1,6 mm

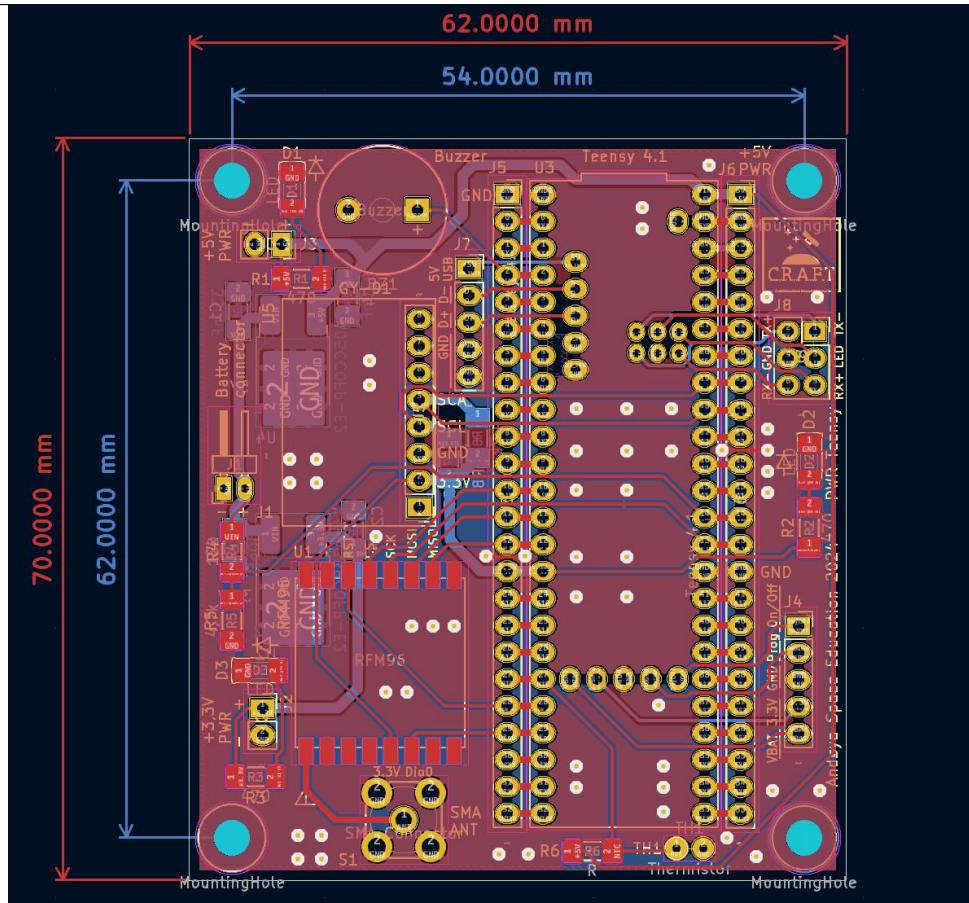


Figure 36: Circuit board dimensions

6.12 The size limit

The circuit board size is a key requirement to be able to achieve the goal for creating the CanSat with the right dimensions. The CanSat has size requirement of 115mm height by 66mm diameter in a cylinder form. That is the European soda can size, and is what the CanSat is to represent in form factor.

6.12.1 Break out pins

The circuit board for the micro processor has breakout pins accessible on the shield for easy connection of external sensors, components and modules. The breakout pins are connected in the same manner as the layout on the Teensy 4.1. To identify the 24 breakout pins next to the Teensy can the text on the Teensy 4.1 be read from the silkscreen on the micro controller.

The Teensy 4.1 has three additional pin groups which can be connected to the break out pins. It is possible the Teensy 4.1 does not come with soldered pins other than the 24 row on each side. If header J4, J7 and J8 are to be used does the pins need to be positioned between the micro controller and the PCB.

6.12.2 Components

The main circuit board consists of the Teensy 4.1 micro controller, a LoRa radio module, the GY-91 multi package, up to two volt regulators, NTC sensor, 2mm PHR connector for battery, as well resistors and capacitors.

Surface-mount devices (SMD) are soldered onto the top of the board at the exposed copper pads. These components only occupy one side of the circuit board and allows for two sided use of the PCB.

Thru pin components have a pin that requires a hole in the circuit board. It will allow for connection from both sides of the PCB but occupies both sides where the pin penetrates the board.

 Table 24: Bill of Materials for CanSat shield

Component	Reference	Model	Mounting type	PCB Layer
Radio	U1	RFM96W	SMD	Top
3,3 V regulator	U2	BA33BC0FP-E2	SMD	Bottom
Teensy	U3	4.1	Thru pin	Top
5 V regulator	U4	BA05CC0FP-E2	SMD	Bottom
GY-91	U5		Thru pin	Top
Buzzer	Buzzer		Thru pin	Top
2x1 connector	J1	2mm PHR	Thru pin	Top
2x1 Header	J2	2,54 mm pitch	Thru pin	Top
2x1 Header	J3	2,54 mm pitch	Thru pin	Top
5x1 Header	J4	2,54 mm pitch	Thru pin	Top
24x1 Header	J5	2,54 mm pitch	Thru pin	Top
24x1 Header	J6	2,54 mm pitch	Thru pin	Top
5x1 Header	J7	2,54 mm pitch	Thru pin	Top
3x2 Header	J8	2,54 mm pitch	Thru pin	Top
470 Ω	R1	1206	SMD	Top
470 Ω	R2	1206	SMD	Top
470 Ω	R3	1206	SMD	Top
12k k Ω	R4	1206	SMD	Top
4,7 k Ω	R5	1206	SMD	Top
12k k Ω	R6	1206	SMD	Top
4,7 k Ω	R7	1206	SMD	Bottom
4,7 k Ω	R8	1206	SMD	Bottom
NTC 10 k Ω	TH1	2,54 mm pitch	Thru pin	Top
Polarized capacitor 470 nF	C1	1206	SMD	Bottom
Capacitor 330 nF	C2	1206	SMD	Bottom
Polarized capacitor 470 nF	C3	1206	SMD	Bottom
Capacitor 330 nF	C4	1206	SMD	Bottom
LED	D1	1206	SMD	Top
LED	D2	1206	SMD	Top
LED	D3	1206	SMD	Top
SMA connector angled	S1		Thru pin	Top

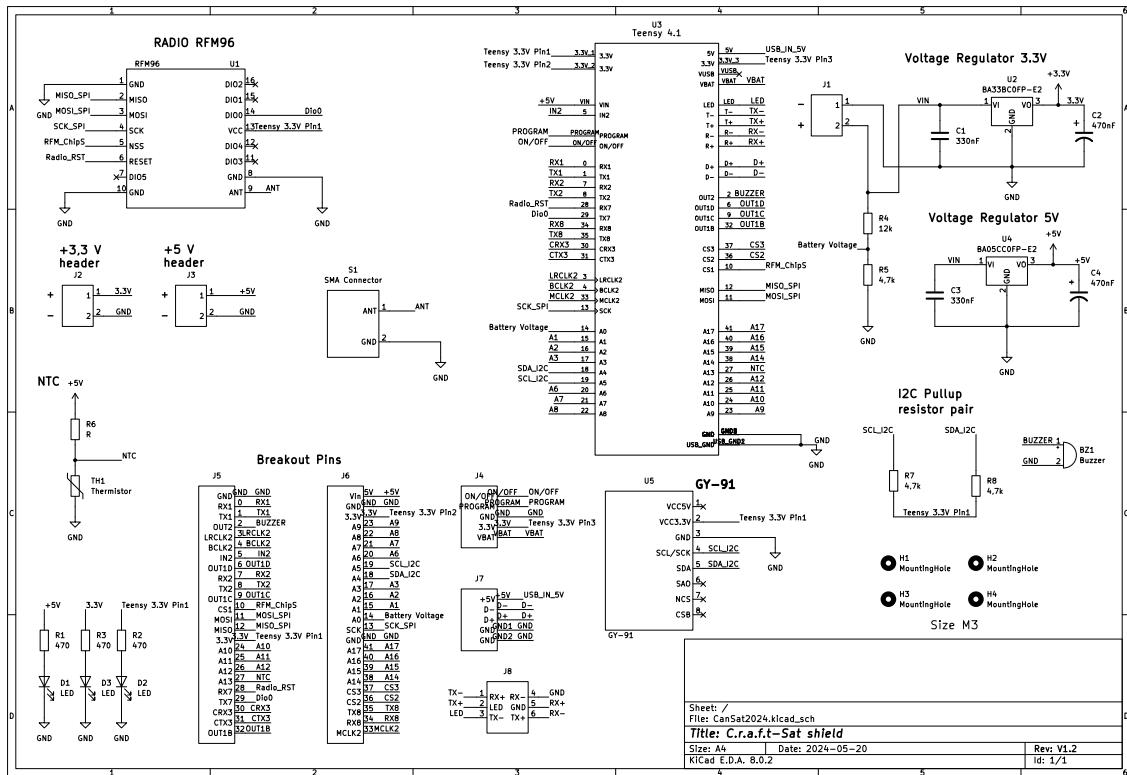


Figure 37: Shield schematic

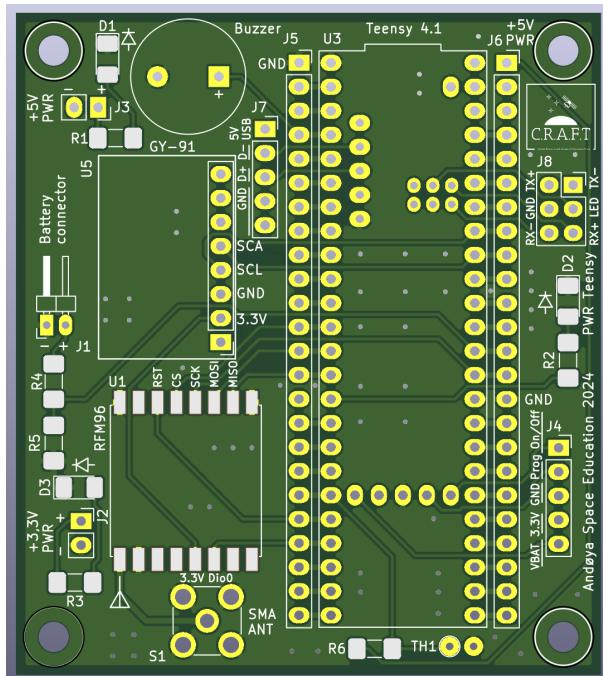


Figure 38: Circuit board top layer, 3D viewer

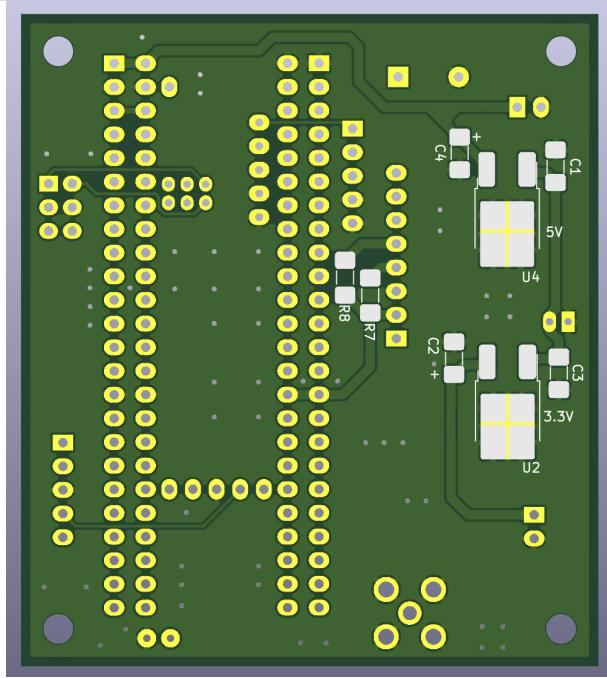


Figure 39: Circuit board bottom layer, 3D viewer

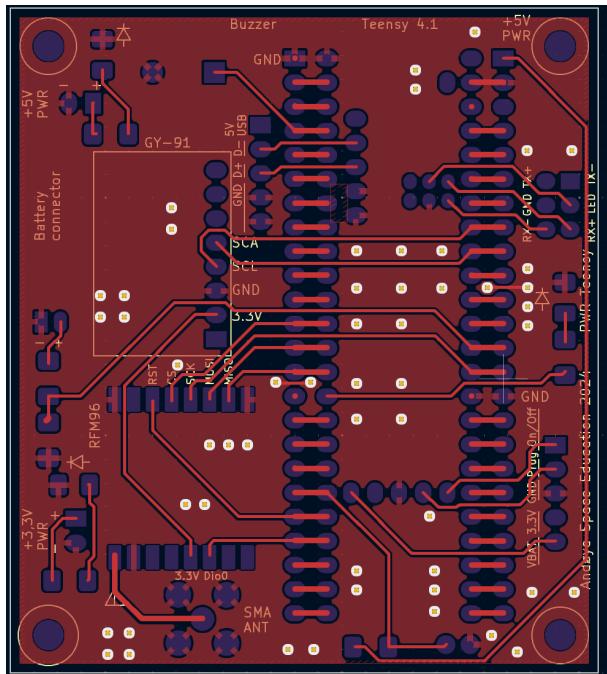


Figure 40: Circuit board top layer

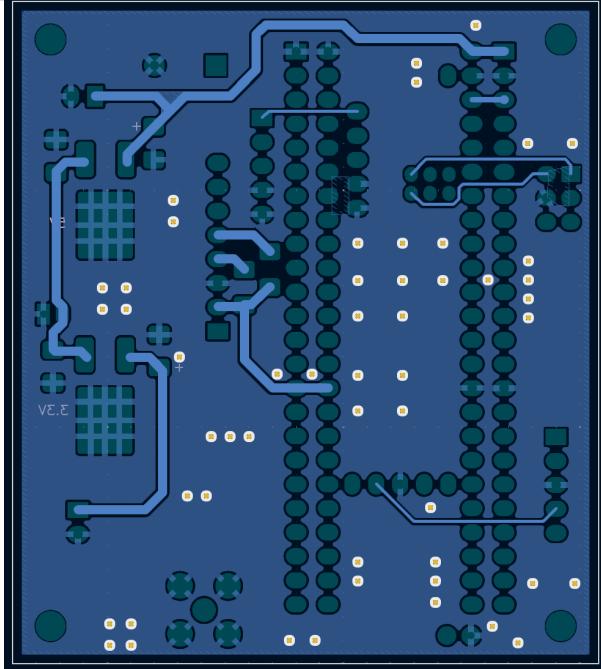


Figure 41: Circuit board bottom layer

6.12.3 Ground plane and trace angles - Noise prevention

Digital circuits can create electrical noise or interference to nearby components and traces because of the impulse of the digital signal. The zeros and ones in the computer code becomes short pulses of DC voltage, and the transition time between 0 V and 3,3 V when the electrons are accelerating. If the derivative of the current equals zero is there no magnetic field being created through the traces. The electricity would experience a great acceleration if the angels of the traces are bent with 90 degrees, so all traces are made sure to be 45 degrees only [48]. The ground plane is a filled zone of copper on the PCB where to grounding is connected on both sides of the board. The effect of the ground plane is to help prevent electrical noise from the traces sending digital signals because of how the magnetic waves are redirected into the circuit board instead of out into the surrounding space.

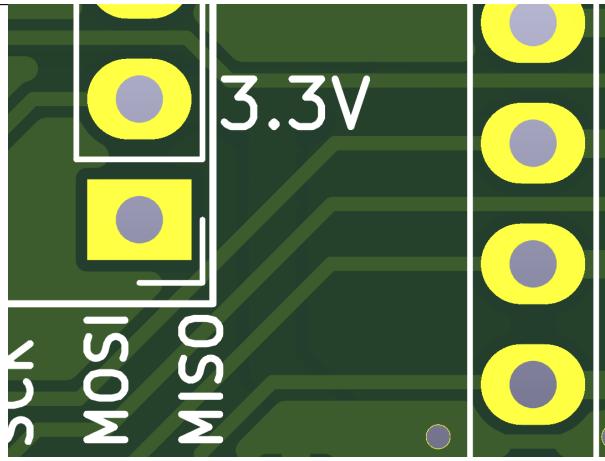


Figure 42: PCB traces at 45 degree angle

6.12.4 Ordering gerber files from manufacturer

The PCB is ordered from JLCPCB and the manufacturer specifications are taken from their website during the making of the circuit board in KiCad to select the minimum distance and trace size for the circuit board [49] and the constraints used in the board file is listed in table (25). When ordering the gerber files from another circuit board factory are these constraints required to be checked beforehand to ensure a working circuit board is possible to make for the factory.

Table 25: PCB ordering minimum capabilities

Subject	Clearance
Minimum clearance	0.127 mm
Minimum track width	0.127 mm
Minimum connection width	0.127 mm
Minimun annular width	0.3 mm
Minimun via diameter	0.5 mm
Copper to hole clearance	0.1 mm
Copper to edge clearance	0.3 mm
Minimum through hole	0.3 mm
Hole to hole clearance	0.5 mm
uVias	0.6 mm
Minimum uVia hole	0.3 mm

6.12.5 Ground plane testing

The pads must be probed to make sure no accidental short was created during the factory printing due to the area the grounding affects could possibly have left over copper be connected to the positive voltage or a micro controller pin.

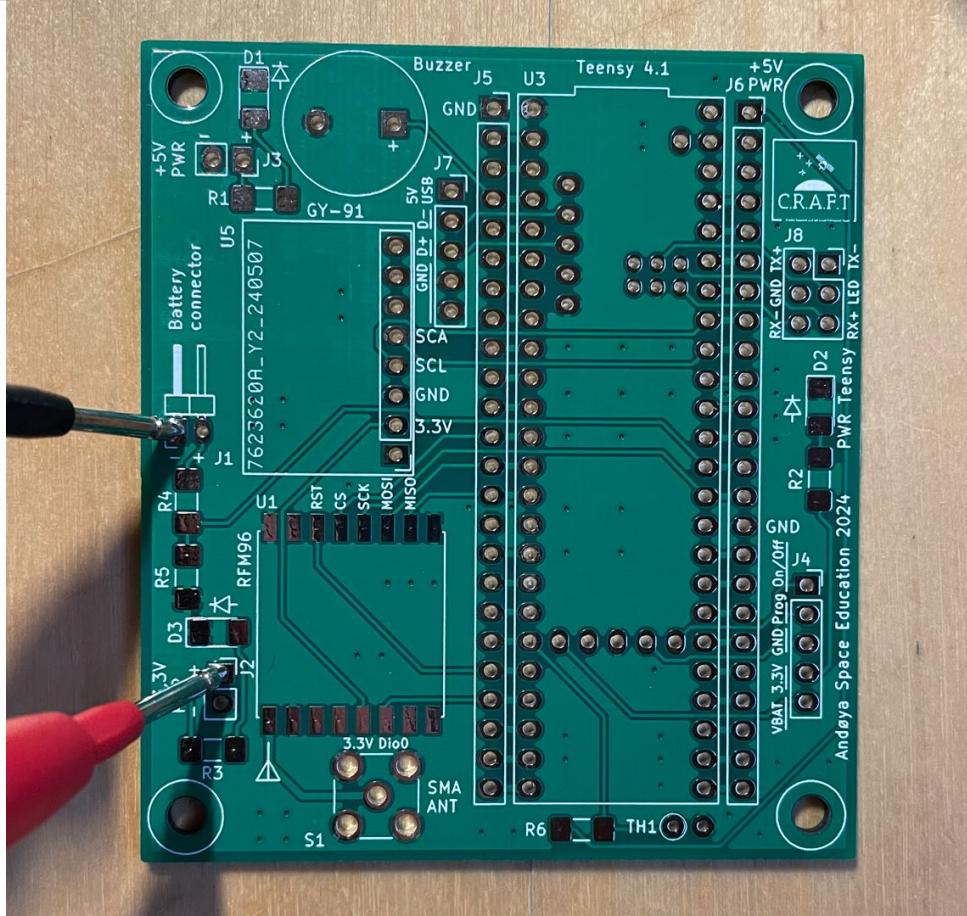


Figure 43: Ground test probing

6.12.6 Hand soldering the components to the PCB

The PCB is designed for easy hand soldering and all pads are oversized. An oval shape is used the header pins to utilize the space more optimally. The component references on the PCB silkscreen can be identified in the circuit schematic.

The radio must be soldered before the SMA connector when the PCB is to be used in the C.R.A.F.T-Sat frame due to the positioning of the angled SMA connector. Due to tight tolerances it is recommended to use a small piece of electrical tape to mask the co. It is important to solder the radio module to the right SMD pads and the antenna pin is marked by the antenna symbol. The radio antenna pin can be identified by reading the silkscreen on the bottom of the radio module. Due to tight tolerances it is recommended to solder the SMA angled connector a little higher above the PCB from the footprint for the C.R.A.F.T-Sat for easier connection of the coaxial antenna cable. When the circuit board is to be used in a CanSat does the SMA connector need to be positioned outwards from the circuit board to allow for the cable antenna to be connected.

A complete soldered shield PCB is seen in figure (44) where the white tape is covering the radio module. The bottom of the PCB is seen in figure(45). Some of the components have been re-soldered to reuse them from the first prototype board which can be seen in appendix F.

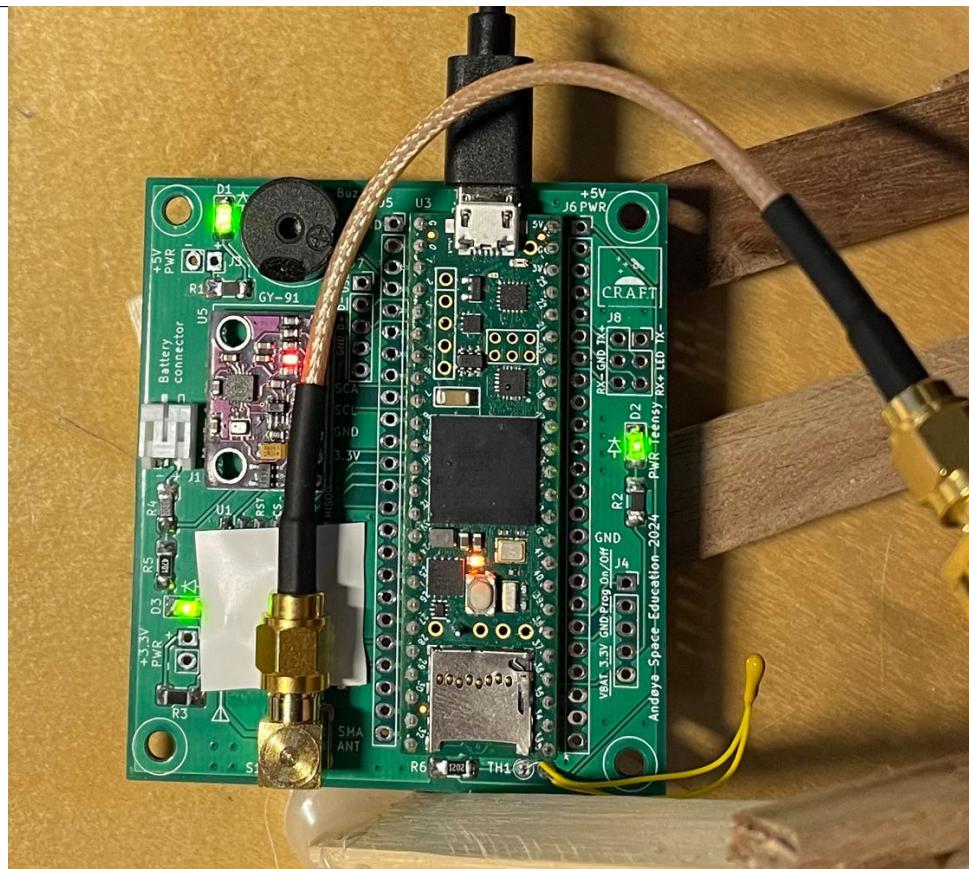


Figure 44: Shield with soldered components powered by USB

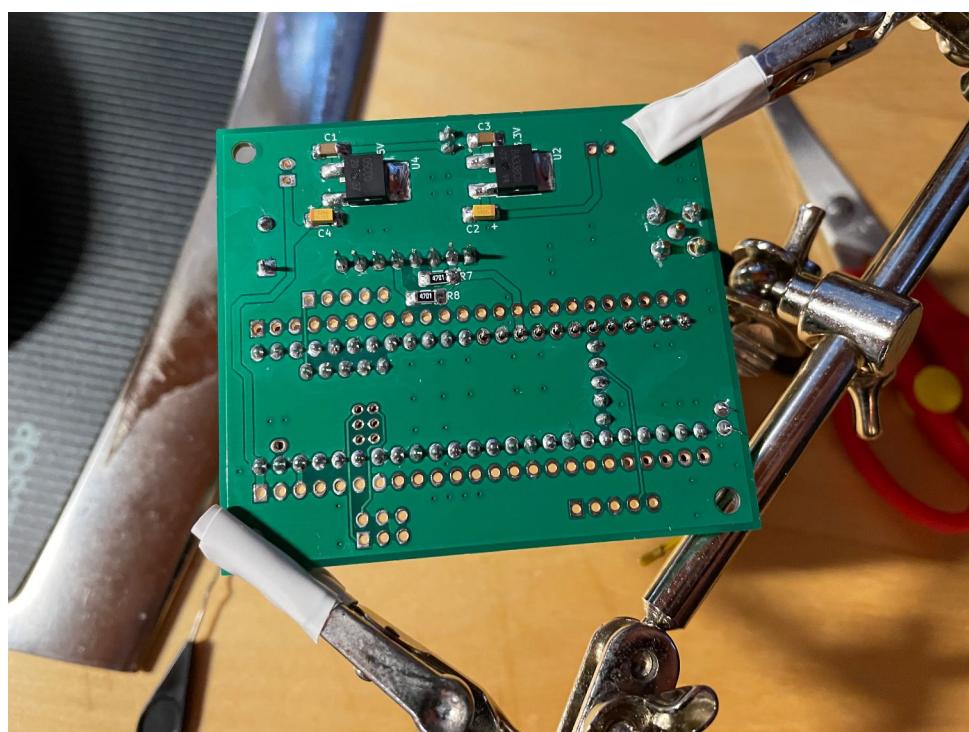


Figure 45: Circuit board bottom

6.13 Aerosols in the atmosphere

GALK|RE

Air is a mixture of many different substances floating in the layer of atmosphere covering the Earth. Many sources of pollution affect the levels of particles present in the atmosphere depending on your location or the path of winds bringing with the local particles afar from the source . The aerosols being suspended in the air can absorb the moisture in the air and fall downwards onto the electronics of the satellite. Also falling droplets can catch particles on the path downwards and bring contamination onto the electronics [50].

6.13.1 Water conductivity

When water get contaminated with ionised particles will the conductivity increase and electricity can travel through the water [51].

6.13.2 Protecting electronics against moisture

Electronics can be protected in several ways. Weather balloons use styrofoam to cover the electronics inside from temperature and keep moisture outside. The use of styrofoam protection in C.R.A.F.T-Sats can be troublesome as the space it occupies is limited and used for the frame, circuit boards and antenna. The use of silicone to cover the exposed copper pads can also be used to create a layer on top of the electronics for moisture or dust protection. This procedure will cause the components and board to be incapable of being soldered to unless the removal of the silicone layer is done. This means it should be preformed as the last step of preparing the satellite electronics. The use of atmospheric sensors are implemented in the C.R.A.F.T-Sat design. Covering the sensors with the silicone protective layer could disable them. Precaution must be taken to the sensors if silicone layer protection is used.

6.14 Positioning system

GALK|RE

6.14.1 The requirement for a tracking module

The CanSat can use a tracker for the satellite for which the students can learn about satellite tracking systems. A tracking system can also possibly help recover the satellite. The term Global Positioning System (GPS) is what most will associate with the positioning system. GPS is only one of the satellite positioning systems that older and cheaper modules are often restricted to use. The latest devices are capable of using all the available civil positioning satellite systems with one device. Therefore, we use the name Global Navigation Satellite System (GNSS) when we speak of the positioning system for CanSat and CraftSat.

6.14.2 GNSS in genereal

To be able to track your own position requires a reference point. The higher number of reference points will increase the accuracy of your location. Satellites in orbit around Earth send out radio signals at about 1.6 GHz from which the GNSS module receives and calculates the position of these signals.

6.14.3 The GNSS module

The GNSS module used for the prototyping of this project is a NEO-6 module developed by uBlox and can only use the GPS satellites. Never versions of the uBlox chip is developed and are available, but currently have a higher cost. We are using a complete module with integrated antenna connector, GPS chip type NEO-6, and supplementary resistors and capacitors.

6.14.4 Components

The components which makes up a complete GPS module can be sectioned into sub modules: Antenna, signal wire, and radio chip.

- Antennas used for 1.6 GHz are small due to the tiny wavelength. Therefore are the antennas seen used with GPS and GNSS modules just a few centimeters. These antennas have an impedance of 50Ω .
- Some modules have the antenna implemented on the board itself, and others have a separate antenna which require a cable.
- The integrated circuitry which does the calculations are put into a compact chip. It is responsible for also communicating with the micro controller, but it only sends out information about the location calculated based of the satellite signals. This radio chip is available for purchase as a separate unit, and as a chip on a board module.

6.14.5 NEO-6

Prototype GPS we are using in the development phase of the project is the NEO-6 uBlox module with active antenna[52].

6.14.6 NEO-M9N

The recommended upgrade from uBlox is to use the new NEO-M9N module which has GNSS capabilities[53].

GALK|RE

6.15 Voltage Regulator

The circuit board requires two voltage regulators to maintain a steady voltage level for the components from the battery to satisfy the requirements of the PCB design. The voltage regulator components we use are Integrated Circuit (IC) with the ability to take

a higher voltage input and maintain a lower voltage at the output in a controlled manner even with a variable input and output load. The internals of a voltage regulator is a combination of many transistors and resistors to create a sub circuitry that can maintain the output voltage and luckily are these components packed into a tiny IC to solder onto the circuitry board.

The voltage regulator has three connector pins: Input, output and ground. The input requires the external power source, and the output is where the electronics at the lower voltage will have to be connected. Ground, often seen abbreviated as GND, is the common connection point of all the components where the voltage potential is zero volts.

The system requirements for the voltage supplies are +3,3 volts and +5 volts. The voltage regulator IC will require a higher voltage at the input to maintain a constant voltage output and possibly a heat sink when higher currents are required [54], therefore when selecting an appropriate battery, the data sheet information about the components tells what voltage range the regulator requires.

The footprint of the component is what describes the size, shape and placement of the pin when it occupies a physical space on PCB. The voltage regulator previously used for the 3,3 volt supply uses the TO-252 footprint on the NAROM 2018 CanSat circuit board. A similar component with the same footprint with higher input voltage capabilities is reused in the redesign of the circuit board.

The additional components required for the voltage regulator to function optimally are the use of capacitors placed near the input and output of the voltage regulator.

Both voltage regulators use the same footprint for the component with the same input and output capacitors.

6.16 Antenna release electronics

GALK|RE

The antenna for the satellite will have a deploy-able function with a command from the micro controller. The burn-wire release electronics uses the expansion board to occupy space for the required components, making the expansion board necessary for the use of the antenna release mechanism.

6.16.1 Antenna activation

The deployment of the antenna does utilize the output of the micro controller to activate the electronic components and transistor which releases the folded antenna. The Teensy does not have a digital to analog converter built in to give a variable Direct Current (DC) output voltage from any pins. It only supports Pulse Width Modulation (PWM) out on the digital pins, but we take advantage of the PWM properties to control the pulse width of the output, thus controlling the transistor through the circuitry.

6.16.2 Current calculations

The current required to heat a thin wire to the melting temperature of a nylon fishing line was calculated using Matlab. The problem at hand was if melting the fishing line to release the antenna would require too much heat to cause material deformation to the surrounding PETG or PLA 3D printed materials used in the C.R.A.F.T-Sat design. We also have limited current available from the battery of 1500 mAh. This gives us only 1,5 Amps at hand.

The problem about the heat is looked at by comparing the energy required in Joules to heat up the thin Nickel Chromium wire compared to a small tiny area of air. If the air requires less energy to reach the melting point compared to the fishing line, would it be possible for the air to transfer the heat into the plastic frame during the antenna release sequence.

Table(26) lists the density for air [55] and Nickel Chromium[56]. We need to calculate the mass for the minimum volume of air and the mass of the metal wire to find the energy required to heat up the materials from the possible freezing temperature in the high atmosphere to the melting point of the fishing line.

Table 26: Material density

Material	Density
Air	1,2928
Nickel Chromium	8649,97022

The material melting points are listed in table(27) and we must stay below the lowest value, and not reach the Nickel Chromium melting point. If we do reach the melting point of the metal wire will the antenna release, but we lose the re-usability of the small wire.

Table 27: Material melting points

Material	Melting point
Nylon 6 - 6/6 (Fishing line)	220 - 265 ° C
Nickel Chromium	1475 ° C
PLA	60 ° C
PETG	80 ° C

The material heat capacity for air[55] and Nickel Chromium[56] is listed in table(28).

Table 28: Material heat capacity

Material	Heat capacity (J/kg*K)
Air	1005
Nickel Cromium	500

Table(29) lists the desired temperatures to be used for the current calculations.

Table 29: Burn-Wire temperature table

Material	Temperature before	Temperature after
Air	-60° C	60° C
Nickel Chromium wire	-60° C	265° C

We calculate the area of the end circles of the wire in Eq(21) and the wire length in Eq(22). The Nickel Chromium wire used in the design has a diameter of 0,2 mm.

$$A = \pi \times r^2 = \pi \times (0.1 \times 10^{-3})^2 = 3,1416 \times 10^{-3} m^2 \quad (21)$$

$$l_{Burn-Wire} = 2 * 10^{-2} m \quad (22)$$

Then we calculate the mass of the close surrounding air to know how large of a space was required of air around the wire to prevent heat transfer through the air. The mass of the wire based of the volume is calculated in Eq(23) and Eq(22), with the material density listed in table(26). We use in this calculations a volume of air of 2 cm by 2 cm by 2 cm.

$$Mass_{(Air)} = square\ meters \times Density_{(Air)} = (0.02\ m)^3 \times 1,2928 = 1.0342 \times 10^{-5}\ kg \quad (23)$$

We use the same equation for calculating the mass of the wire in Eq(24).

$$Mass_{(Wire)} = A \times l \times Density_{(Wire)} \quad (24)$$

And insert Eq(21) and Eq(22) into Eq(25)

$$Mass_{(Wire)} = 3,1416 \times 10^{-3} \text{ m}^2 \times 2 \times 10^{-2} \text{ m} \times 8649,97022 = 5.4349 \times 10^{-6} \text{ kg} \quad (25)$$

Now we need to calculate the delta energy required for heating up these elements. The required energy to be put into the materials to heat them up is found with Eq(26)[55].

$$dQ = c \times m \times dT \quad (26)$$

All temperatures calculated with Eq(26) is to be done in the unit Kelvin, so we add 273,15 to the temperature in Celsius in the calculations.

$$dQ_{Air} = c_{Air} \times Mass_{Air} \times (T_0 - T) <=> \quad (27)$$

$$1005J/(kg \times K) \times 1.0342 \times 10^{-5} \text{ kg} \times ([273,15 - 60] - [273,15 + 60]) <=>$$

$$dQ_{Air} = 1.2473$$

Then the wire calculations is done with Eq(26) as well.

$$dQ_{Wire} = c_{Wire} \times Mass_{Wire} \times (T_0 - T) <=> \quad (28)$$

$$500J/(kg \times K) \times 5.4349 \times 10^{-6} \text{ kg} \times ([273,15 - 60] - [273,15 + 265]) <=>$$

$$dQ_{Wire} = 0.8832$$

We can see from the results of Eq(27) and Eq(28) the energy it requires to heat up a 2 cm by 2 cm by 2 cm volume of air from the cold air temperatures to the melting point of the PLA material. The wire will require less energy to warm up the material which is contacting the fishing line, so reaching the heat of surrounding printed frame is not possible with the energy put into the wire.

Finally to verify the current requirements to heat the wire to the required temperature do we go backwards from the equation for the delta energy to the wire in Eq(28). The energy is calculated as Joules, which equals Watts × Second. Therefore can we divide the Joule by the second we want to have the antenna release to calculate how much power we need to put into the wire in a short or long time. We pick 1 second.

$$Watt_{Wire} = dQ_{AirWire}/second = 0.8424/1 = 0,8832 \text{ W} \quad (29)$$

Then we take the square rot of the watt in Eq(29) and divide the watt by the resistance in the burn-wire which has been measured to be 1,6 Ω.

$$\text{Required Current}_{\text{Wire}} = \sqrt{\text{Watt}_{\text{Wire}}/R} = \sqrt{0,8832/1,6\Omega} = 1,1494 \text{ A} \quad (30)$$

We can then see the current required in Eq(30) we only need a current of 1,15 A to heat up the nickel chromium wire to the required heat to melt the fishing line.

6.16.3 PWM to DC converter

The PWM function on the Teensy allows us to create a DC signal if the signal is transmitted through a low-pass filter circuitry on the PCB. The analog filter is made of a resistor in series and a capacitor in parallel. The resistor will slow down the current which comes out of the pin on the Teensy so controlled current spike occurs each time the pulse out of the pin is sent. The small current then charges up the capacitors during the "ON" state of the pulse. When the pulse is in "OFF" state does the capacitors discharge slowly. In this manner can we charge up the capacitors to a certain energy level and let it discharge just slightly before charging back them up to the set level. This way we can manage to reach the wanted voltage with the energy put into the capacitor. That is the concept of the PWM to DC converter to let us set a voltage level controlled by the less capable micro-controller.

6.16.4 DC Signal amplification

The Teensy digital out signal pins has max current draw of 4 mA which is the recommendation from the technical information supplied by the manufacturer website [57]. The power transistor requires up to 50 mA at the base pin to active the transistor according to the data sheet [58]. The Teensy digital output is 3,3 volts max and the transistor requires a voltage higher than that to draw the required current for the burn-wire to get the heat for melting the nylon fishing line. The transistor base pin voltage is shown in the PSpice simulation in figure(48) to require a voltage of 3,6V to provide the burn-wire with enough current. Therefore is an operational amplifier used for signal amplification of the DC signal. We can see the current output increase through each component in table(30) taken from the data sheets.

Table 30: Burn-Wire component requirements

Component	Current out max	Voltage out max	Current inn min	Voltage in min
Teensy 4.1	4 mA	3,3 V	-	-
Op-amp	80 mA	12 V, V_{CC}	15 nA	0 V
Transistor	2 A	100 V, V_{CC}	-	1,6 V

6.16.5 Components

We use as few components as possible to save space, weight and cost for the satellite. A few resistors, capacitors, one op-amp and one transistor is used to convert the PWM from the micro-controller to control the current through the transistor. The op-amp pin 1 is identified by the angled edge of the ceramic casing being on the same side as seen in figure(46).

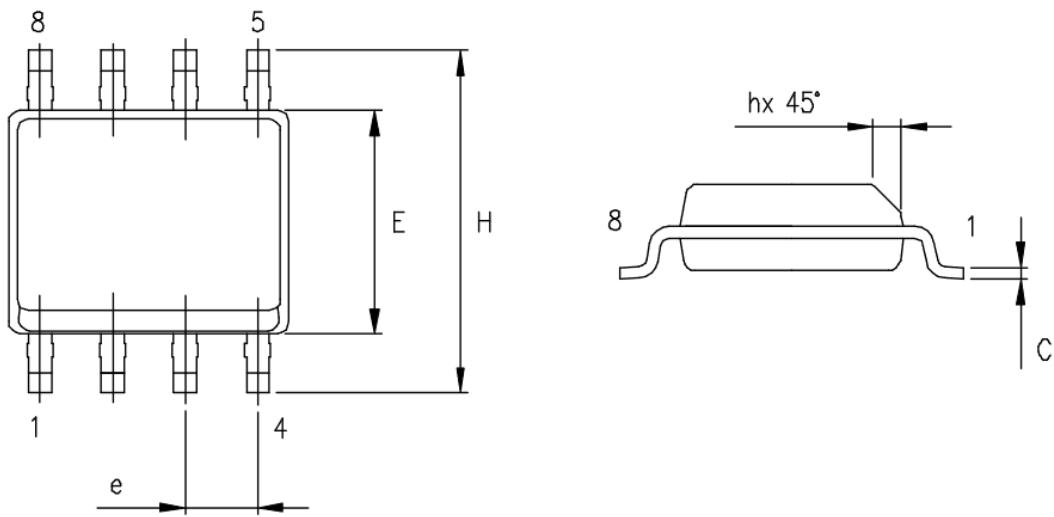


Figure 46: Op-amp pin identification

Table 31: Burn-wire components

Component	Value	Unit	Amount	Reference	Mounting type
Resistor	1k	Ω	1	R1	Surface mount
Resistor	1,5k	Ω	1	R2	Surface mount
Resistor	12k	Ω	1	R3	Surface mount
Capacitor	470n	F	3	C1, C2, C3	Surface mount
Op-Amp	TS921ID		1	U1	Surface mount
Transistor	TIP112		1	Q1	Thru pin
Terminal block	PCB to wire		1	J3	Thru pin
Battery connector	2mm PHR		1	J1	Thru pin
Pin connector	2,54mm pitch		1	J2	Thru pin

6.16.6 Burn-wire

A burn-wire is a metal wire that will heat up from a high current flowing across a tiny cross section. It can be used to melt materials under tension. We need to use easily accessible materials to keep cost low and production time short, therefore fishing line is the choice for this design for holding back the antenna release mechanism. The use of Nickel Chromium wire made for foam cutting tools is used for the design. The thin Nickel Chromium wire has a slightly higher resistance than copper, so it will generate more heat from the current available.

6.16.7 Simulation - PSpice

The burn-wire simulation is done with another CAD software not used for the making of the shield and expansion card. This software is called OrCad and is also made for

PCB production and allows for simulating components and circuits. All components for the simulation are used from the software Cadence library [59], except the op-amp TS921ID seen in figure(47) which is downloaded from the manufacturer's website[60]. The simulation schematic is seen in figure (47) and the component values are simulated with the use of PSpice models to test optimal component values and verify the design from using a PWM generator which can simulate the output of the micro controller. In the simulation are three probes used to measure the voltage at the input and output of the op-amp, which can be identified as the blue marker in figure(47) and green marker. The red marker measures the current through the burn-wire. The simulation output is seen in figure(48) and we can read the desired output values of the micro controller voltage of the blue marker and trace to be below 3,3 V. The green marker is the amplified output of the op-amp and goes above 3,3 V, which is more than the micro controller can handle. The red trace shows a current above 1.2 amps which is working fine.

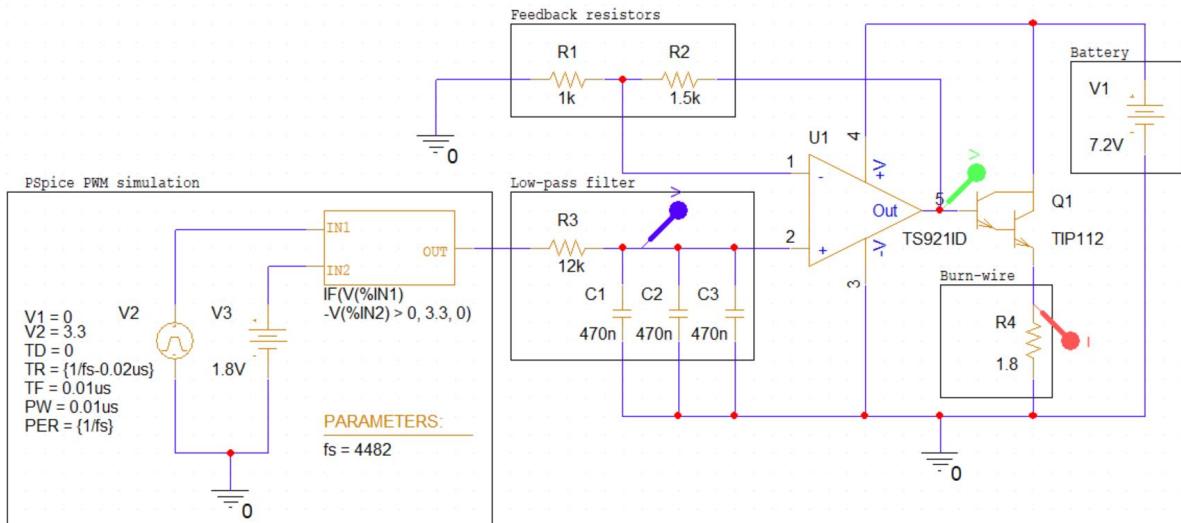


Figure 47: Burn-Wire PSpice schematic

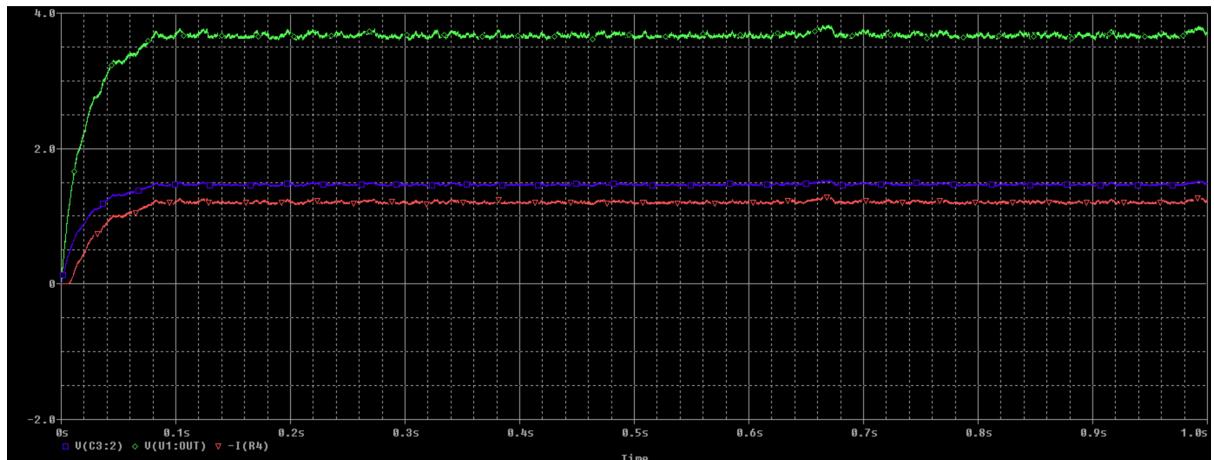


Figure 48: Burn-Wire PSpice simulation output

6.16.8 Connection and power

The burn-wire is connected to the battery from its own connection to avoid high current spikes in the common digital ground plane. The PWM pin use one input signal from the Teensy.

6.16.9 Circuit Schematic - Burn-Wire

The schematic for the burn-wire is seen in figure(49). Each section of the system is identified by the boxes and text. The low-pass filter consist of one $12\text{ k}\Omega$ and three 470 nF capacitors. The high resistor value will prevent high current draw from the micro controller digital pins, and allows to reuse the same resistor value as used on the CanSat shield circuit board for the NTC resistor. Putting capacitors in parallel simply sums the values of all capacitors, and the simulations resulted in the value of the three capacitors in Eq(31).

$$C = C1 + C2 + C3 \Leftrightarrow 470\text{ nF} + 470\text{ nF} + 470\text{ nF} = 1,41\text{ }\mu\text{F} \quad (31)$$

We can see from the simulation results in figure(48) the rise-time of the filter to be quick, but not so quick we get fast spikes which will cause noise and an unstable transistor output. This value from equation(31) and the resistor value of the low-pass filter will give the cutoff frequency of the low-pass filter.

$$f_c = \frac{1}{2\pi RC} \Leftrightarrow \frac{1}{2\pi \times 12k \times (1,41 \times 10^{-6})} = 9,4\text{ Hz} \quad (32)$$

We are only interested in DC voltage levels, so the a low cutoff frequency is not a concern for activating the antenna release mechanism and will help lower the noise output generated from alternating voltage levels .

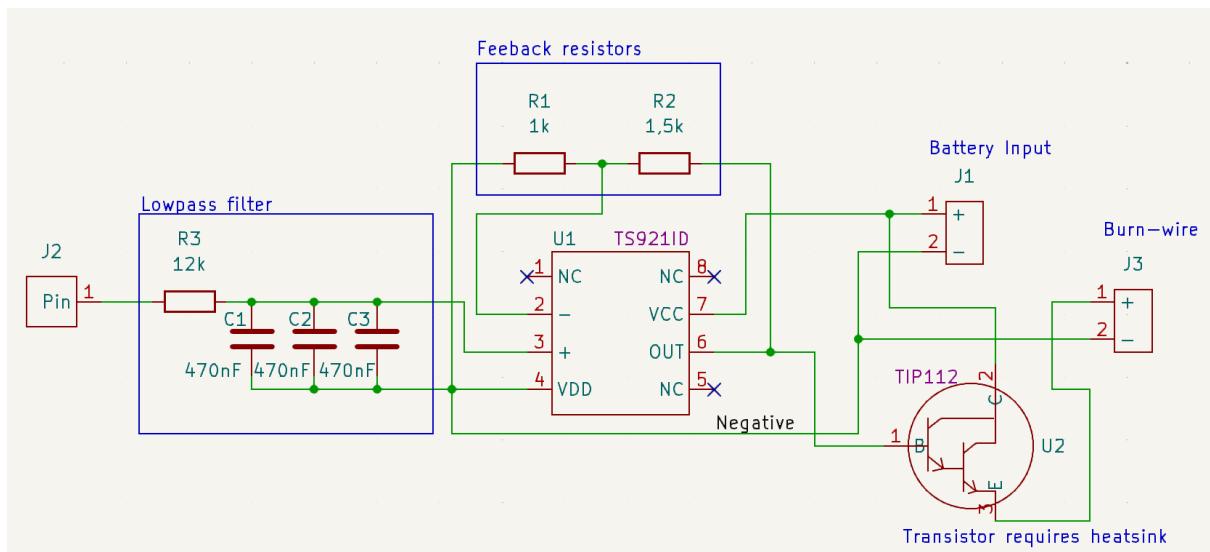


Figure 49: KiCad Burn-Wire schematic

6.17 Test Results

The antenna release mechanism has been tested with up-link activation from the ground station to the C.R.A.F.T-Sat for the burn-wire circuit. The electronics stay idle when waiting with the micro controller with the PWM pin at 0 duty cycle. The antenna *will* release if: the wrong PWM pin is used, if the burn-wire is connected before the Teensy is on and ready, or if the Teensy is reprogrammed with the battery connector. Connect the burn-wire battery connector as a final step.

The burn-wire is seen connected to the antenna frame in figure(50) and completely integrated in the C.R.A.F.T-Sat frame in figure(51).

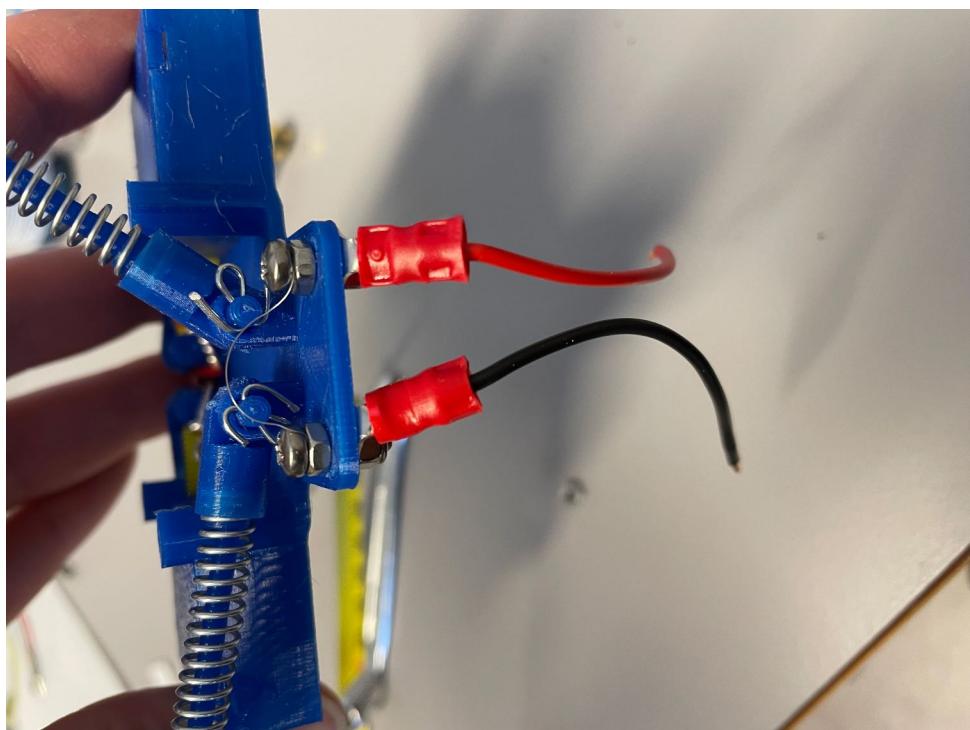


Figure 50: Burn-Wire connection

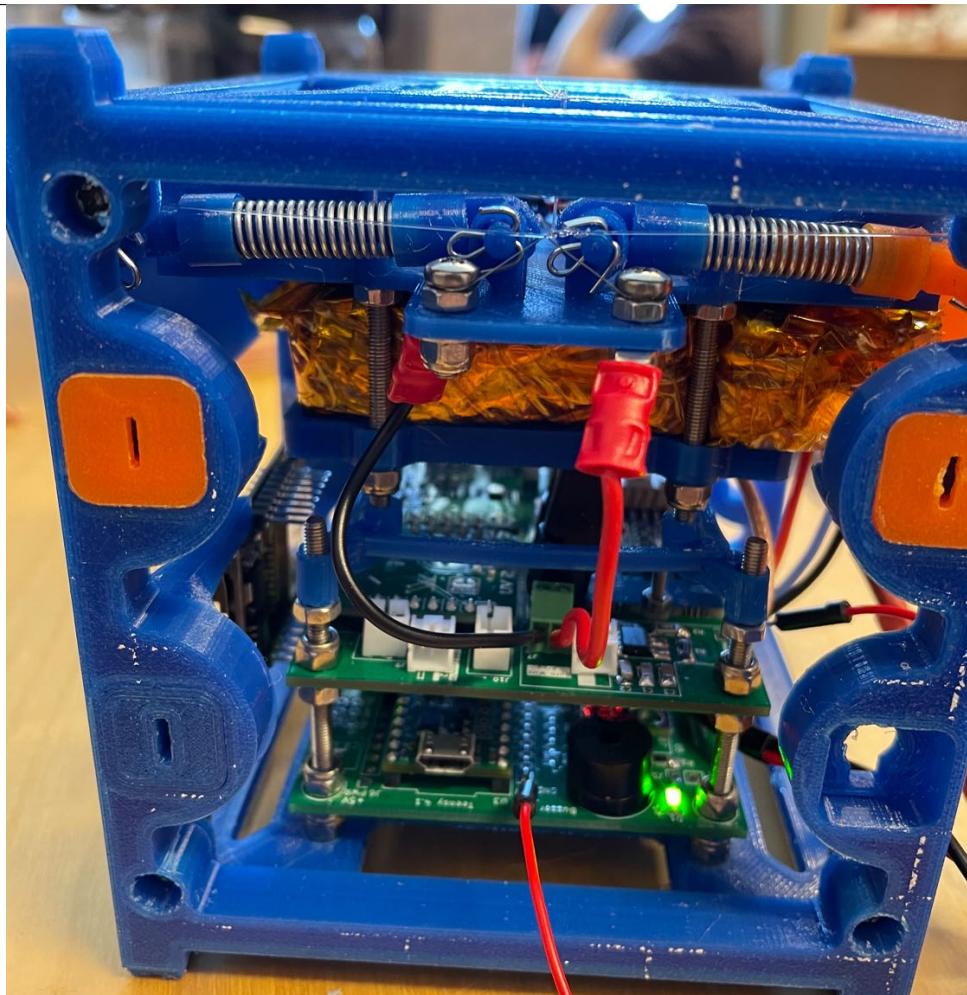


Figure 51: Burn-Wire connected in CraftSat

6.18 Connection of antenna to SMA connector

GALK|RE

6.18.1 Quick overview

Antennas are used for wireless communication between electronics and is used in our project to transfer the measured data to the ground station.

6.18.2 Connection

The antenna used for the satellite is a dipole design and is connected to the shield SMA connector by a coaxial cable. The antenna connection use a coaxial cable. This is so any magnetic fields are countered within the cable itself and stopping the cable as acting as antenna. The end of the cable has one lead of the dipole antenna attached to the positive terminal of the coaxial cable, while the second antenna piece is connected to the ground. The antenna is connected in the C.R.A.F.T-Sat. The coaxial cable used is 15 cm long and extended through the plastic frame with positive and negative wires. If the use of a

longer coaxial cable is used can the red and black wire seen in figure(52) be shorter. The antenna connection will alter the effect of the transmission of the antenna. The wires are soldered to the metal measuring tape used as antenna to ensure good connection. The surface paint on the measuring tape is scraped off where to solder meets.

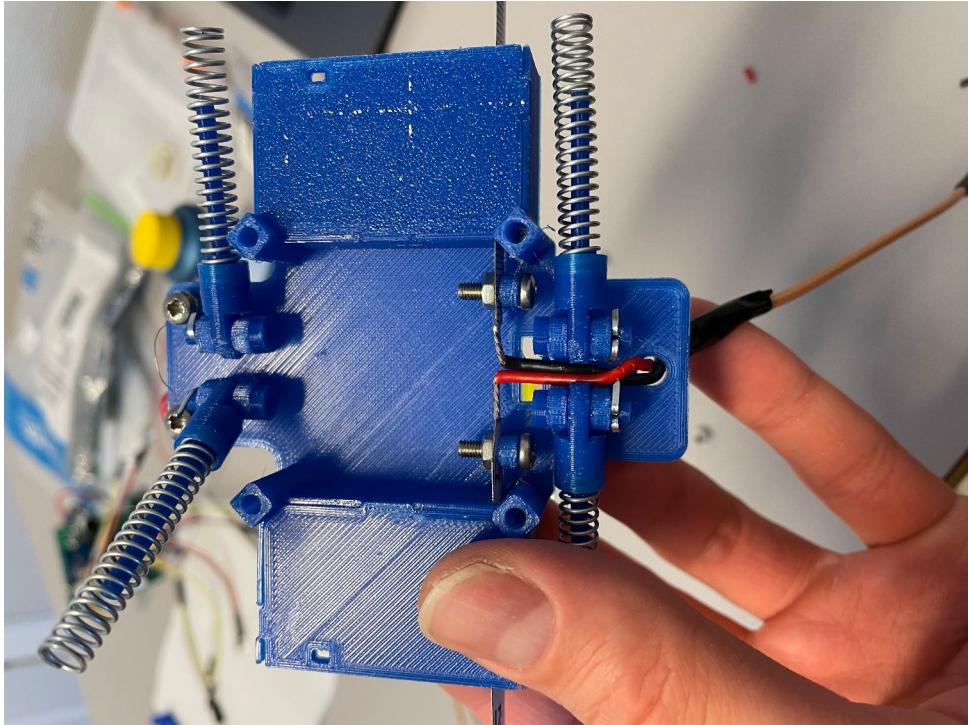


Figure 52: Coaxial cable connected with small signal cables to the antenna

6.18.3 Material

For the creation of the radio waves does the electricity from the radio module require a metallic structure to allow small currents to accelerate of a sinusoidal manner. Only the acceleration of electrons will generate the radio waves. The material used in this design is from a metallic measuring tape. The materials with low resistivity works great for sending radio waves, and the resistance can easily be tested using a multi-meter. The metal measuring tape over about 1 meter was measured to be only $0,8 \Omega$, thus being a suitable material for antennas.

6.18.4 Testing

The antenna test done in the EMC (Electromagnetic Compatibility) lab was preformed using a frequency analyzer and a closed metal room to prevent outside signals to give false readings. The C.R.A.F.T-Sat antenna can be seen located on the table in figure(53). The antenna is cut to be half the wavelength of the 433 MHz frequency of the radio signals over the total length, and each pole is one fourth of the wavelength. The signal spike read from the frequency analyzer showed a strong signal transmitted at 433 MHz. A short lab report explaining the testing procedure in appendix E.

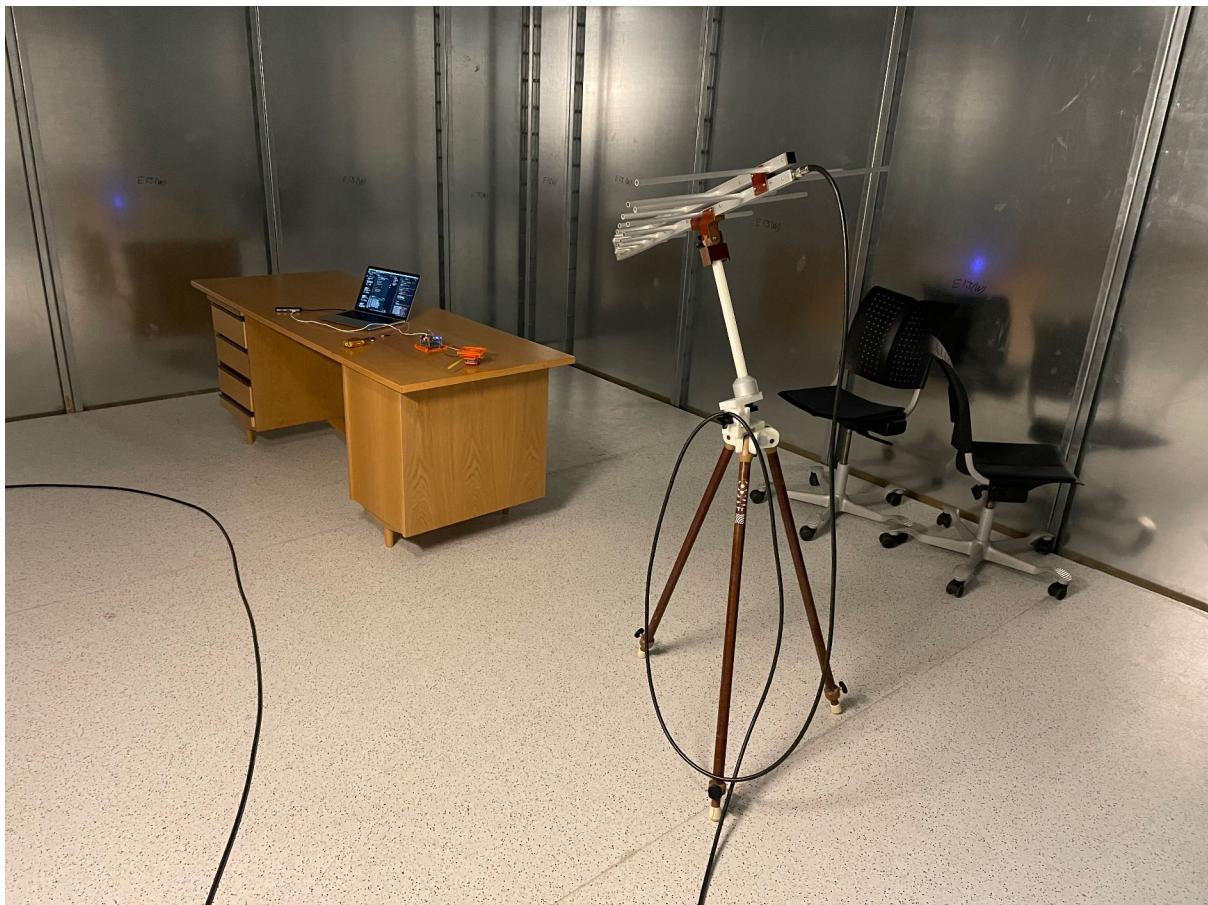


Figure 53: EMC lab antenna testing

6.18.5 Balun

In antenna design are the impedance an important thing to check if it will match the antenna to the radio. The radio signals travel as an alternating current through the metallic structure of the antenna material, which causes the generation of the magnetic field that will convey the data to be transferred across airspace. The coaxial cable between the radio module and the metallic leads of the dipole antenna is created this way to prevent unwanted signal generation. This coaxial cable will cause a capacitance build up during the radio wave generation. The effect of this will cause a phase shift in the signal travelling out of the radio module and into the antenna. The effect of this can be countered with the use components which has reactance properties to change the phase of the alternating current creating the magnetic field. Implementing antenna matching components into the design is called a Balun, and can often be seen as a small box near the middle of the antenna. The matching components are not necessary if the matching of the antenna is good or good enough, but a "good enough" situation can always be improved.

6.18.6 Antenna matching

Antenna matching is done using a Smith chart[61] on a handheld vector network analyzer[62] seen in figure(54), and is how the antenna can be tested to get a good match to the radio. The radio output requires 50Ω impedance to get the most of the signal transferred into the antenna. The capacitive or inductive properties of the connector, wire and antenna will affect the antenna resistance and how well the antenna does perform. The modern vector network analyzer can handle the Smith chart calculations on its own and we can simply read the recommended capacitor and/or coil values to create a better antenna match. Figure(55) shows the Smith chart being the circles and lines in the background, and the green marker being the one we are interested in for the antenna matching. The closer the green marker is at center, the better the transferred signal will be. Hand made antennas will easily be altered from minor alterations during making and getting a good match requires tuning.



Figure 54: Vector Network Analyzer with automatic matching test



Figure 55: Vector Network Analyzer testing signal loss in cable

6.19 Components Weight

RE|GALK

6.19.1 Weight Components: Estimation from Datasheet

It is essential to maintain the weight of both the CanSat shield and the expansion card within the weight requirement. The weight of the CanSat shield must not exceed 330g, and the weight of the C.R.A.F.T-Sat with both PCB cards must not exceed 500g to fulfill the ASE requirement for the maximum allowable weight. Table (32) details the components of the CanSat shield along with their estimated weights and the CanSat's total weight after assembling. Table (33) shows the weight of the components used in the expansion card, total estimated weight, and C.R.A.F.T-Sat's total weight after assembling. As indicated in Tables (32) and (33), both CanSat and C.R.A.F.T-Sat weigh less than the maximum allowable weight requirement, thus complying with the ASE requirement regarding the weight limit in CanSat and C.R.A.F.T-Sat.

Table 32: Estimated Component Weight and Total Estimated and Measured Weight of CanSat.

Component	Weight in grams
Teensy 4.1 Microcontroller	10
RFM96 Radio	1.092
GY-91 Sensor Module	3
Voltage Regulator 3.3V	0.325
Voltage Regulator 5V	1.404
Buzzer	1.6
NTC Temperature Sensor	0.004
Antenna	14
Edge Connector	10
Capacitors and Resistors	0.297
Pin Headers and Connectors	3.85
Circuit Board	14.256
WLToys 7.4V Battery	95
Total Estimated Weight	170.4
CanSat' Total Measured Weight	205

Table 33: Estimated Component Weight and Total Estimated and Measured Weight of Expansion Card.

Sensor	Weight in gram
MikroElektronika UV 2 Click	17
MikroElektronika Environment 3 Click	16
Seeed Studio Grove-VOC and eCO2 Gas Sensor	9
ESP32-CAM Camera	8
Voltage Regulators	8
Burn Wire Components	8
Connectors	11
Circuit Board	15.2
Total Estimated Weight	92.2
C.R.A.F.T-Sat's Total Measured Weight	410

RE|GALK

6.20 Sensors Selection for Expansion Card

6.20.1 Advantages for Adding Expansion Card in C.R.A.F.T-Sat

The expansion card in C.R.A.F.T-Sat is an important subsystem designed to interface with the main card, which is operated with **Teensy 4.1** microcontroller. This expansion card is specially designed to host additional sensors that the main card cannot house due to space constraints. By integrating the expansion card with the main card, we can significantly improve data collection by C.R.A.F.T-Sat.

6.20.2 Advantages for Implementing Expansion Card

1. Increase Data Collection:

- **More Sensors Integration:** The expansion card allows us to include additional sensors that the main card cannot house due to limited size. Adding more sensors can collect diverse data during a single mission. Multiple data collection can be useful for further studies and research and for a better understanding of our environment.

2. Modular and Adjustable Design:

- **Customization:** Depending on the specific mission objectives of the C.R.A.F.T-Sat, various expansion cards with the same size but different sensors can be selected, making the C.R.A.F.T-Sat suitable and adaptable for various missions.

3. Smart Design:

- **Efficient of Use of Space:** Projects as C.R.A.F.T-Sat can suffer from limited place. Integrating an expansion card is a smart step that can help to take advantage of all areas inside the frame of C.R.A.F.T-Sat.

6.20.3 Sensors Selection for Expansion Card

The selection of sensors for C.R.A.F.T-Sat is crucial in order to enhance the ability of C.R.A.F.T-Sat to a new level. After thorough and careful research, the expansion card in C.R.A.F.T-Sat will house both the ultraviolet (UV) sensor (Figure (56)), the carbon dioxide (CO₂) sensor (Figure (58)), the humidity sensor (Figure (57)), and the camera (Figure (59)). Table (34) summarizes the sensors used on the expansion card, detailing their names, dimensions, and weight.

Table 34: Sensors Implemented on Expansion Card

Sensor	Diemension in mm	Weight in gram
MikroElektronika UV 2 Click	28.6x25.4	17
MikroElektronika Environment 3 Click	28.6x25.4	16.5
Seeed Studio Grove-VOC and eCO ₂ Gas Sensor	40x20	9
ESP32-CAM Camera	40x30x10	8

The exact selection of these sensors is based on several factors. These factors include compatibility with **Teensy 4.1 microcontroller**, the ability to operate at extreme temperatures, the quality and precision of data collection and the availability of the sensors.

Compatibility and Communication Interface

The sensors in the expansion card must be fully compatible with the **Teensy 4.1 microcontroller**, which supports a variety of communication protocols, including I²c, UART, SPI, and PWM. Ensuring compatibility simplifies the complete integration with the microcontroller.

- **MikroElektronika UV 2 Click Sensor:** Supports I²c, UART, and SPI, offering flexibility in interfacing options.
- **MikroElektronika Environment 3 Click for Humidity:** Supports I²c and SPI, ensuring flexibility in interfacing options.
- **Seeed Studio Grove-VOC and eCO₂ Gas Sensor:** Supports I²c, which is essential for easy communication with **Teensy 4.1**.
- **ESP32-CAM Camera:** Supports I²c, UART, SPI, and PWM, making it versatile for various communication needs.

Temperature of Operation

The C.R.A.F.T-Sat can face extreme cold temperature under mission. The sensors equipped in it must therefore withstand this extreme temperatures. The operating tem-

perature ranges of the selected sensors ensure their functionality and reliability in harsh conditions.

- **MikroElektronika UV 2 Click Sensor:** -40°C to +85°C.
- **MikroElektronika Environment 3 Click for Humidity:** -40°C to +85°C.
- **Seeed Studio Grove-VOC and eCO2 Gas Sensor:** -40°C to +85°C.
- **ESP32-CAM Camera:** -20°C to +85°C.

Quality and Precision

- All selected sensors are designed by known manufactures that are known of quality product, ensuring that the data collected are faithful and can be used for further studies and research.

Availability

The availability and short delivery times of the components are crucial to keep the project on schedule. The selected sensors were easily available and purchased from reliable suppliers known for quality products.

- Sensors for CO2, UV, and humidity measurements were purchased from **RS Online**, a reputable supplier known for high-quality electronic components and fast shipping time.
- The ESP32-CAM was purchased from **elkim** store in Norway, ensuring quick delivery time.

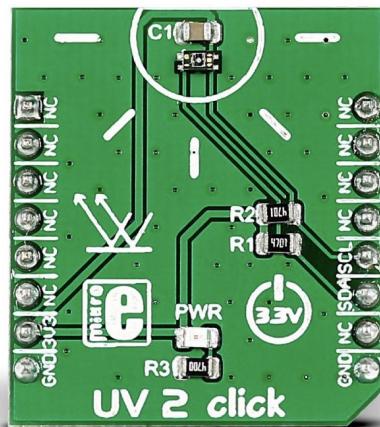


Figure 56: MikroElektronika UV 2 Click Sensor

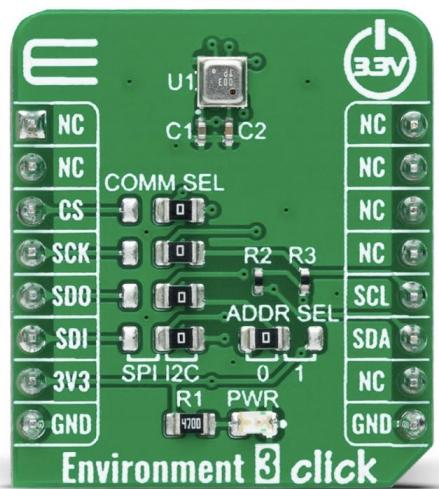


Figure 57: MikroElektronika Environment 3 Click

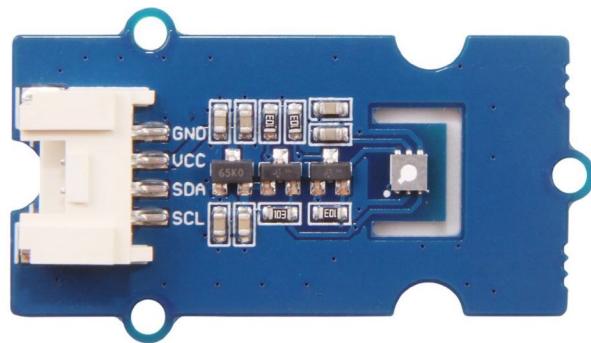


Figure 58: Seeed Studio Grove-VOC and eCO2 Gas Sensor

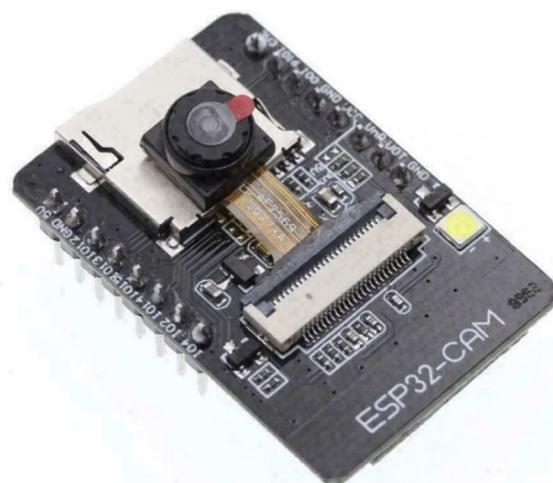


Figure 59: ESP32-CAM Camera

6.20.4 Significance of Measuring UV, CO₂, and Humidity, and Capturing Images

1. **UV Radiation Measurement:** Monitoring the UV values is crucial for studying the Ozone layer on Earth. The ozone layer is important because it absorbs most of the sun's harmful UV radiation. Understanding and monitoring UV radiation levels at different altitudes can help monitor Ozone's health and detect changes due to climate change and pollution([63]).

UV radiation can be harmful to people. Tracking its level can help us to understand the impact of UV on human health and explore new methods to reduce these effects([63]).

2. **CO₂ Measurement:** CO₂ has recently gained obvious attention due to its contribution to global warming. Monitoring CO₂'s level will help track pollution and manage a better solution to mitigate its impact on Earth. This sensor will also measure volatile organic compounds (VOC), providing valuable data on air quality. Its low power consumption and compact size make it suitable for C.R.A.F.T-Sat applications.
3. **Humidity Measurement:** Humidity data is valuable for understanding weather. Humidity data can enhance our understanding of cloud formation and weather forecasting ([64]). This sensor also consumes low power and has an impact size, making it suitable for C.R.A.F.T-Sat applications.
4. **Image Capturing:** Capturing images from high altitudes provides visual data that can be used for Earth observation and environmental monitoring. Capturing images can also be a fun task for students that ASE will work with with our C.R.A.F.T-Sat.

6.21 Design of Expansion Card

The dimension of the expansion card PCB is set to 62x70 mm to ensure it fits within the dimension C.R.A.F.T-Sat of 100x100 mm. This size allows for efficient utilization of space while leaving room for camera implementation and wiring between the expansion card and shield circuit board. The mounting holes are placed at a distance of 54x62mm to match the shield circuit board, ensuring compatibility with the C.R.A.F.T-Sat mounting mechanism using M3 screws. This design avoids the need for separate mounting solutions, simplifying the integration process. Table (35) and Figure (60) show the dimensions of the expansion card.

Table 35: Circuit Board Dimensions for Expansion Card.

Board Side	Value
Board outline height	70 mm
Board outline width	62 mm
Vertical distance between holes	62 mm
Horizontal distance between holes	54 mm
Hole size	M3
Board thickness	1.6 mm

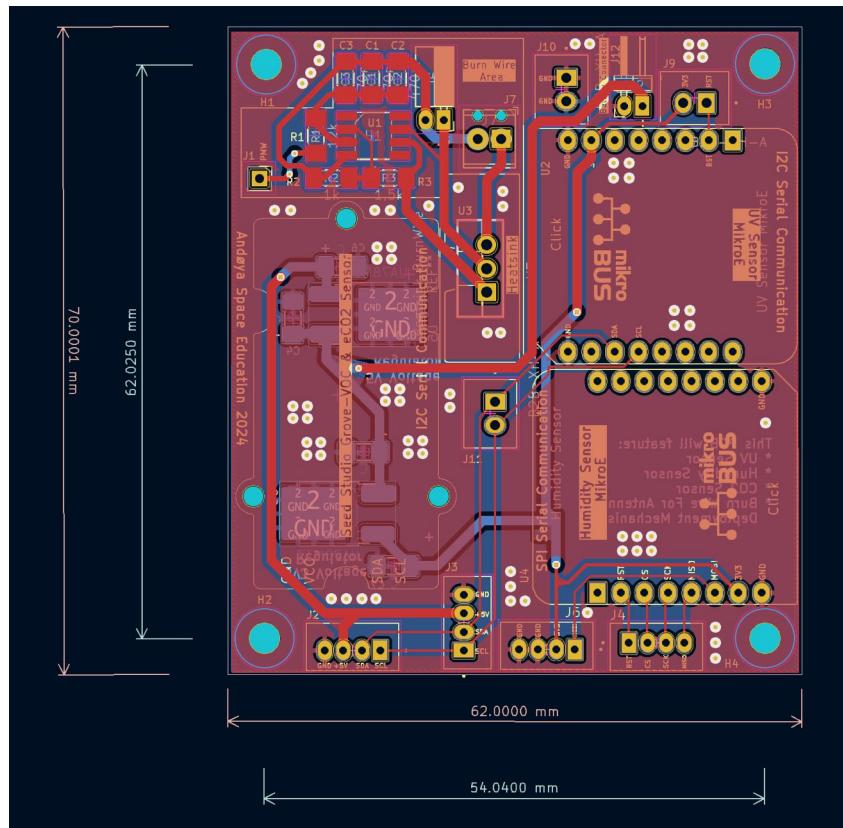


Figure 60: Circuit Board Dimensions for Expansion Card.

6.21.1 Components Implemented in Expansion Card

The expansion card is equipped with several sensors and key components necessary for the mission of C.R.A.F.T-Sat. All components of the expansion card are listed in Table (36).

Table 36: Expansion Card's Bill of Material Without Burn-Wire Components

Component	Reference	Model	Mounting Type	PCB Layer
UV Sensor	U2	MikroElektronika UV Click	Thru Pin	Top
Humidity Sensor	U4	MikroElektronika Environment 3 Click	Thru Pin	Top
CO2 Sensor		Seeed Studio Grove-VOC and eCO2 Gas Sensor	Thru Pin	Top
3,3 V Regulator	U6	BA33BC0FP-E2	SMD	Bottom
5V Regulator	U5	BA05BC0FP-E2	SMD	Bottom
2x1 Header	J9	2.00 mm pitch	Thru pin	Top
2x1 Header	J10	2.54 mm pitch	Thru pin	Top
2x1 Header	J11	2.54 mm pitch	Thru pin	Top
2x1 Header	J12	2.00 mm pitch	Thru pin	Top
4x1 Header	J2	2.00 mm pitch	Thru pin	Bottom
4x1 Header	J3	2.00 mm pitch	Thru pin	Top
4x1 Header	J4	2.00 mm pitch	Thru pin	Top
4x1 Header	J6	2.00 mm pitch	Thru pin	Top
470 nF	C6	1206	SMD	Bottom
470 nF	C7	1206	SMD	Bottom
330 nF	C4	1206	SMD	Bottom
330 nF	C5	1206	SMD	Bottom

The burn-wire components used in Expansion Card is illustrated in Table (31).

The arrangement of the components on the expansion card is depicted in Figures (61) and (62), which only show the pads on the top and bottom layers, respectively. Figures (63) and (64) present a 3D visualization of the top and bottom layers, respectively.

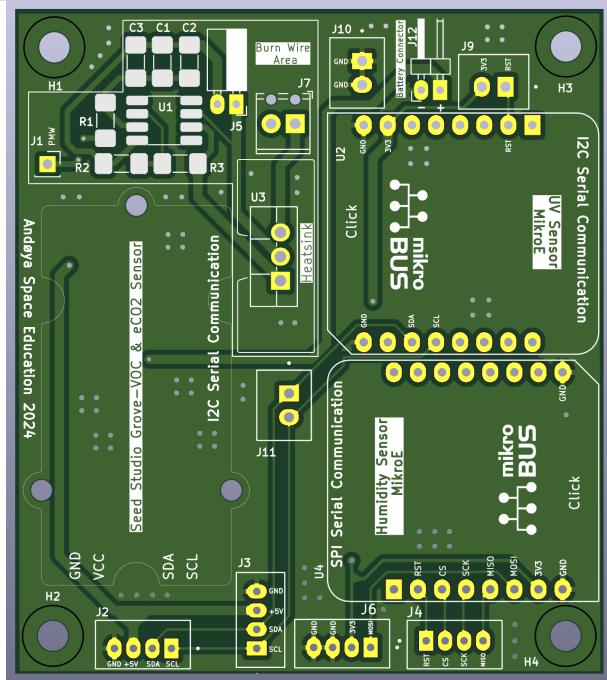


Figure 61: Three Dimensional Visualisation of Expansion Card Without Components: Top Layer.

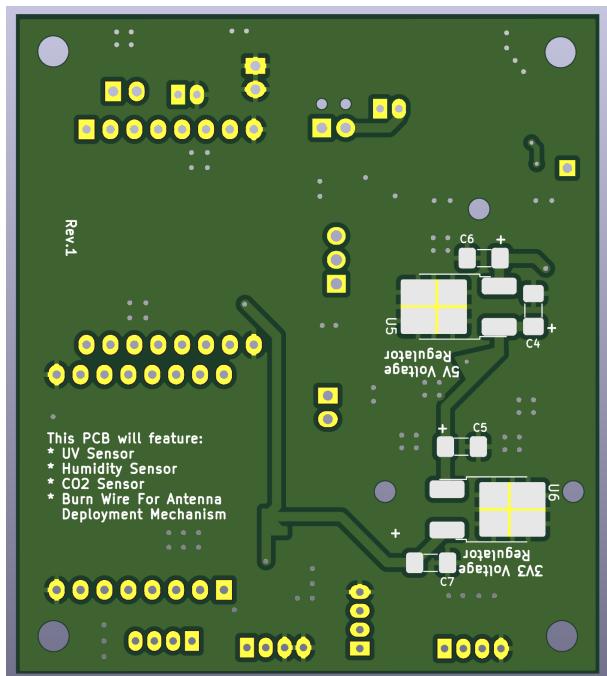


Figure 62: Three Dimensional Visualisation of Expansion Card Without Components: Bottom Layer.

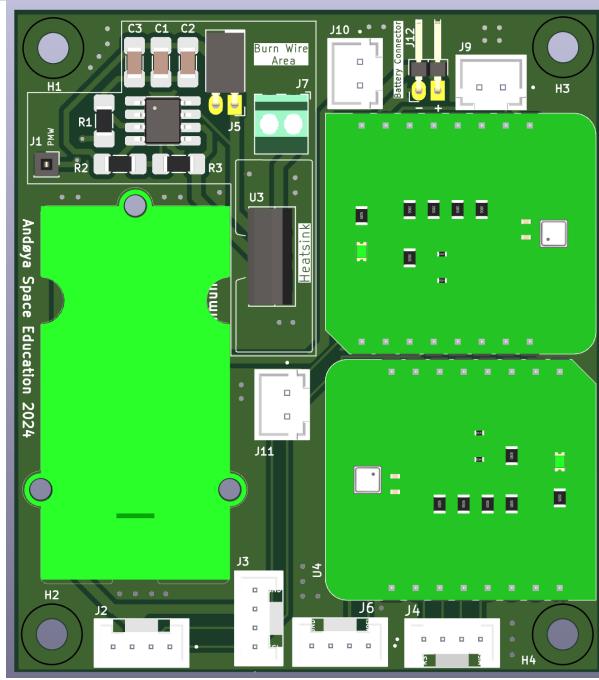


Figure 63: Three Dimensional Visualisation of Expansion Card With Components: Top Layer.

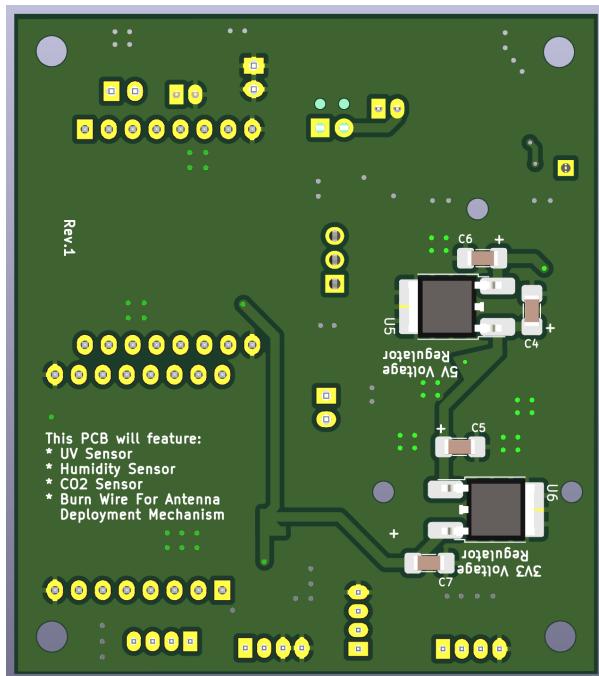


Figure 64: Three Dimensional Visualisation of Expansion Card With Components: Bottom Layer.

6.21.2 Ground Plane and Trace Angle

1. **Ground Plane:** A ground plane is implemented on the top and bottom layers on the expansion PCB card in order to minimize the electrical noise in the same was as main shield used.
2. **Trace Angle:** 90-degree trace angles are avoided because it can cause signal reflection and impedance discontinuities([48]).

6.21.3 Increased Pad Size

To make soldering easier for students with limited experience, the pad size of all components on the expansion card has been increased. This change allows for easier placement and soldering of components by hand.

6.21.4 Ground Plane Testing

A detailed ground plane test is performed to ensure that there are no short circuits or connectivity issues. This test is crucial to verify that the PCB manufacturing process has not made any production faults that could impact the functionality of the expansion card.

6.21.5 Ordering The Expansion Card

The PCB expansion card is ordered from **JLCPCB**, using the same specifications as the main shield. These specifications are presented in Table (25).

RE|GALK

6.22 Battery Selection for CanSat and C.R.A.F.T-Sat Mission

The selection of the optimal battery for CanSat and C.R.A.F.T-Sat is crucial. The battery must not only power all the systems inside reliably but also meet rigorous specifications for size, weight, durability, stability, and efficiency, all within the harsh environment in which the C.R.A.F.T-Sat will be placed. Given that CanSat has a much nicer environment, we can restrict our research to C.R.A.F.T-Sat, where the results can be applied to both.

Due to ASE requirement that all components must belong to consumer products, we must exclude all special batteries that cannot be obtained by ordinary people. We need, therefore, to restrict our research to the two most popular batteries today, LiPo and lithium-ion (Li-ion) batteries.

6.22.1 Li-ion Batteries

Li-ion batteries: These are rechargeable batteries where lithium ions travel back and forth between the negative (anode) and positive (cathode) electrodes during the charging and discharging processes ([65]).

Advantages of Li-ion batteries:

-
1. **Cost:** Li-ion batteries have been used for a very long time, leading to reducing manufacturing costs significantly.
 2. **High Power Density:** Li-ion can store more charge compared with LiPo batteries with similar size.
 3. **Battery Life:** Li-ion batteries tend to last longer compared with LiPo batteries.

Constant Voltage: Li-ion batteries are capable to maintain a constant voltage.

Disadvantages of Li-ion batteries:

1. **Efficinecy Decrease:** Despite Li-ion batteries last longer, the efficiency decreases with time.
2. **Harsh Usage:** Li-ion batteries are not suitable for rough usage as they have liquid chemical electrolytes inside them, making them easily to take fire by failure or rough usage.
3. **Safety:** Li-ion batteries are unsafe in higher temperature. This is why we hear various episodes where battery mobile takes fire when it gets warm.

6.22.2 LiPo Batteries

LiPo batteries represent an alternative type of rechargeable batteries, utilizing polymer electrolytes rather than liquid electrolytes. LiPo batteries have been invented much lately compared with Li-ion batteries but have gained more popularity lately in various technologies([66]).

Advantages of LiPo batteries([67]):

1. **Safety:** LiPo batteries are considered safe batteries and safe to use. They are more robust against explosion thanks to flecible packaging with aluminium that protects them from explosion or other dangerous situation.
2. **Weight:** LiPo batteries are considered lighter than Li-ion batteries because of a lack of liquid electrolyte, making them suitable for critical projects.
3. **Low Self-Discharge Level:** LiPo batteries will not discharge quicker when the battery is not in use.

Disadvantages of LiPo batteries([67]):

1. **Life Span:** LiPo batteries usually have shorter life span compared with Li-ion batteries.
2. **Energy Density:** LiPo batteries have less energy density compared with Li-ion batteries, making them store less amount of energy relative to their weights and/or size.
3. **Cost:** LiPo batteries are considered more pricy compared with Li-ion batteries with same size and specifications.

Table (37) summarizes the main differences between the batteries Li-ion and LiPo:

Table 37: Comparison Between Li-ion and LiPo Batteries

Features	Li-ion Battery	LiPo Battery
Battery Life	Longer battery life than LiPo	Shorter battery life than Li-ion
Safety	Higher risk of explosion under hazardous conditions	Safer to use and less likely to explode.
Efficiency by Time	Loses actual charging capacity over time	Maintain charging capacity better than Li-ion batteries.
Energy Density	High energy density	Lower energy density than Li-ion batteries
Weight	Heavier than LiPo battery	Lighter than Li-ion battery
Charging Speed	It can take longer charging time	It normally takes shorter charging time than Li-ion battery
Cost	Less expensive than LiPo battery	More expensive than Li-ion battery

6.22.3 Battery Selection: Risk Analysis

Assessment of both Li-ion and LiPo batteries reveals that each possesses significant features, complicating the decision-making process. However, the importance of these features varies. Table (38) presents these features and their respective importance in the design of C.R.A.F.T-Sat, rated on a scale of 1 to 5, with 5 being the most significant and 1 the least.

Although the price difference between Li-ion and LiPo batteries is minimal, particularly for smaller batteries, this does not significantly affect their importance in decision making. Battery longevity is undoubtedly critical, as a longer battery life reduces the need for replacements and minimizes environmental impact. Over time, the lifespan of LiPo batteries has improved, narrowing the gap with Li-ion batteries. Although energy density and charging speed are significant, the safety of operating the C.R.A.F.T-Sat for extended periods without risk of taking fire leads the importance scale. This concern for safety, which can pose a risk to people, animals, and the environment, makes it the most critical factor, leading us to favor the LiPo battery over the Li-ion battery.

Table 38: Weightning Battery Features From 1 to 5.

Features	Scale
Battery Life	3
Safety	5
Efficiency by Time	4
Energy Density	4
Weight	5
Charging Speed	4
Cost	2

6.22.4 Battery Selection Process for C.R.A.F.T-Sat

Choosing the right power source is essential for C.R.A.F.T-Sat to successfully complete its mission. The C.R.A.F.T-Sat, which is designed according to ASE standards, the battery must meet various electrical and physical criteria.

Initial Design Requirements

Essential criteria for the C.R.A.F.T-Sat battery include:

1. **Voltage Output:** Battery's voltage output must be sufficient to operate **Teensy 4.1** microcontroller with 5V voltage. This means that the battery voltage output must at least be have an output of 5V.
2. **Physical Constraints:** The battery must be compact enough to fit within the C.R.A.F.T-Sat's frame, with minimum possible height in order to have place for both main PCB and expansion card, and all necessary wiring between them.
3. **Environmental Conditions:** The battery should function effectively in extreme cold temperatures, providing reliable power
4. **Availability:** Because of the project's tight timeline, the batteries that can be shipped from Norway, was more prioritized.

Battery Selection and Justification of Selection: WLToys 7.4V 1500mAh LiPo

Given the tense requirements, the battery selected after several iterations is a WLToys LiPo battery with 7.4V voltage output and a capacity of 1500 mAh, shown in Figure (65). This battery was chosen based on several critical factors:

- **Size and Integration:** The smaller dimensions facilitated easier integration of battery and other electrical components in C.R.A.F.T-Sat.
- **Sufficient Power Delivery:** A capacity of 1500 mAh will be sufficient to supply both CanSat and C.R.A.F.T-Sat with adequate power.
- **Safety and Reliability:** As mentioned before LiPo batteries are preferred for their safety features, especially in terms of reduced risk of explosion or fire contra Li-ion batteries.
- **Cold Weather Performance:** LiPo batteries are recognized for their excellent performance in cold conditions, which is the reason why they are mostly used in drones today.

Availability: The selection was also influenced by the availability of the battery for quick delivery within Norway.



WLtoys

Figure 65: WLToys 7.4V 1500mAh LiPo Battery

6.23 Power Consumption for CanSat and C.R.A.F.T-Sat

After assembling all components in main PCB card and successfully gathered the data, we measured a maximum power consumption of 150 mA per hour. This indicates by using a battery with capacity of 1500 mAh, we can estimate a power supply in around:

$$\text{Power Supply Duration} = \frac{1500 \text{ mAh}}{150 \text{ mA}} = 10 \text{ hours}$$

6.24 Battery Test

The power supply provided by the selected battery, the WLToys 7.4V 1500mAh LiPo battery, is a crucial aspect. The C.R.A.F.T-Sat mission must be designed to last 3 hours, ascending to approximately 26 km in altitude before returning to the ground. During this mission, C.R.A.F.T-Sat will experience a significant temperature decrease, which could affect battery performance by accelerating its discharge rate. Therefore, it is crucial to perform comprehensive testing of the battery in similar thermal environments to determine whether the current battery meets the requirements or if a new battery selection process is necessary.

In this experiment, our objective is to expose the battery to severe low temperatures to:

1. **Ensure Complete Functionality:** The C.R.A.F.T-Sat will face several temperature decrease when it ascents. Testing the battery's performance under these conditions is vital to ensure that it can maintain its charge and effectively power the C.R.A.F.T-Sat without discontinuities.
2. **Test Sensitivity of Chemistry:** Despite LiPo batteries being known for handling extreme temperatures very well compared with Li-ion batteries, the extremely low temperature still can significantly slow chemical reactions within the battery impacting its ability to deliver power efficiently. This test will determine the battery's capability to maintain optimal discharge rates.
3. **Ensure Safety:** The extreme cold could change the structural integrity of the battery, potentially leading to failures such as short circuit or thermal issues. By testing the battery at low temperatures, we aim to confirm its durability and safety.
4. **Test Duration and Energy Requirements:** It is crucial to verify that the battery's capacity of 1500 mAh can meet the C.R.A.F.T-Sat's operational needs for the full duration of its mission.

6.24.1 Prework: Battery Isolation Before Test Execution

To simulate harsh environmental conditions at an altitude of 26 km, where temperatures can drop to -85°C, it is critical to employ robust insulating materials that can effectively

shield the battery while maintaining its performance. Given the limitations in reaching such extreme temperatures directly in our testing environment, we used both freeze spray, a freezer at home that can offer constant temperature at -18°C, and a freezer at **Kongsberg Automotive** that can supply a constant temperature below to -40°C. We also used specific insulating materials to create a more realistic test that can approach the real-world situation as closely as possible.

Materials and Tools Descriptions and Functionalities

1. Household Freezer

- Temperature Capability: It can maintain a constant temperature of -18°C.
- Role in Testing: This freezer was primarily used for testing the battery in subzero environments that are not too extremely cold as a start phase of testing. It provided a stable temperature for initial observations of battery performance.

2. Industrial Freezer from Kongsberg Automotive:

- Temperature Capability: Capable of reaching temperatures as low as -40°C.
 - Role in Testing: The industrial freezer allowed for more extreme testing conditions. At -40°C, we could stress the battery even closer to the expected lower temperature under the mission. This freezer was crucial for identifying any potential issues that might arise under harsh cold stress.
3. **Freeze Spray:** Freeze spray is used to rapidly lower the area it is applied to, in this case, around the battery. This spray can achieve temperatures down to -55°C.
4. Role in Testing: Initially, Freeze spray was used to evaluate the insulation properties during the probe test, and subsequently on the battery after insulation by comparing the measured temperature before and after insulation.

5. Heat Insulating Tape (Polyimide Film):

6. Properties: Polyimide tape is recognized for its superior thermal stability and finds use in numerous advanced applications. It can operate effectively within a temperature range of -73 °C to +260°C, making it suitable for insulating and protecting the battery under temperature changes.
7. Role in Testing: This tape was used to wrap the battery, providing a thermal barrier that mitigates the rapid temperature changes and maintains a more stable internal temperature. The tape's high lower- and upper-temperature threshold ensures that it does not lose its properties during the test.

8. Heat Insulating Thermpfoil:

- Properties: This Thermofoil is a lightweight reflective insulation material that is effective from -46°C to +86°C. It can reflect both heat and cold, making it suitable for battery insulation.

-
- Role in Testing: Thermofoil was applied around the battery over the polyimide tape; this layer is meant to provide extra protection and insulation by reflecting back the outside temperature changes and keeping the internal temperature more stable.

6.24.2 Battery Testing: Process and Results

Test Procedure and Insulation Verification

1. Temperature Sensor Calibration

- **Procedure:** The test began by calibrating the temperature sensor using freeze spray to ensure accuracy.
- **Outcome:** The sensor correctly measured -50°C, confirming its reliability for testing.

2. Insulation Capability Test on a Non-Battery Material:

- **Procedure:** Before starting test with the battery, a simple insulation was applied to another material to avoid battery damage, battery explosion, and to evaluate the effectiveness of insulation against freeze spray.
- **Outcome:** The test was successful. The insulation could block the cold as long as the insulation was perfectly sealed.

3. Battery Insulation Effectiveness

- **Procedure:** The battery was then insulated using Polyime tape, providing a thermal barrier to slow temperature changes and stabilize the internal temperature.
- **Additional Inuslation:** Thermofoil was applied over the Polymide tape to enhance protection by reflecting external temperature and maintain internal temperature stable.

4. Freezer Test for Insulation Effectiveness:

- **Procedure:** The insulated battery was placed in a household freezer to monitor the temperature change.
- **Outcome:** The internal temperature dropped from +17°C to -18°C in approximately 40 minutes, instead of immediately in case with no insulation. This confirms the effectiveness of the insulation. However, a thorough sealing at all corners was required to prevent entre of air, suggesting the need for specialized tool to extract all the air between the battery and insulation layer and perfect sealing.

battery Performance Test

5. **Setup:** The insulated battery was connected to the main PCB shield with all components to determine the the duration of power supply.
6. **Procedure:**

-
- Startet in an industrial freezer from 0°C, quickly decreasing to -40°C. Test duration is 4 hours.
 - Continued at room temperature (between 15-18°C) for two hours.
 - Completed in household freezer for five hours.

7. **Results:** The battery voltage dropped from 8.4V (fully charged battery) to 6V (empty battery) after 11 hours throughout these procedures. A further drop in voltage within an hour for the discharging test. Table (39) shows the development of the voltage throughout the test.

It is important to mention that letting the battery discharge at a voltage below the allowable value (around 6V) is not recommended because it can permanently damage the battery. This procedure was performed for experimental purposes and because of the presence of devices that can protect the battery from permanent discharging and damage.

Table 39: Voltage Development under Battery Test.

Test Duration	Voltage Drop
After 0 hours	8.4V (Fully charged)
After 4 hours	8.1V
After 6 hours	7.42V
After 7 hours	7.39V
After 8 hours	7.13V
After 9 hours	6.9V
After 10 hours	6.46V
After 11 hours	6.15V (Empty)
After 12 hours	3.26V (Not recommended)

6.24.3 Vacuum Test for Battery

To test the reliability for the battery even more, the battery was exposed to several low pressure levels in laboratory. The test duration was 60 minutes, five minutes for each low pressure value. The battery survived the low pressure test and supplied the main PCB card with all sensors with the power. The voltage was measured after the test and the value was in normal level. Additionally, the shape of the battery was compared to another battery of the same type, as shown in Figure (66), showing no alterations. These findings suggest that the test was successful.

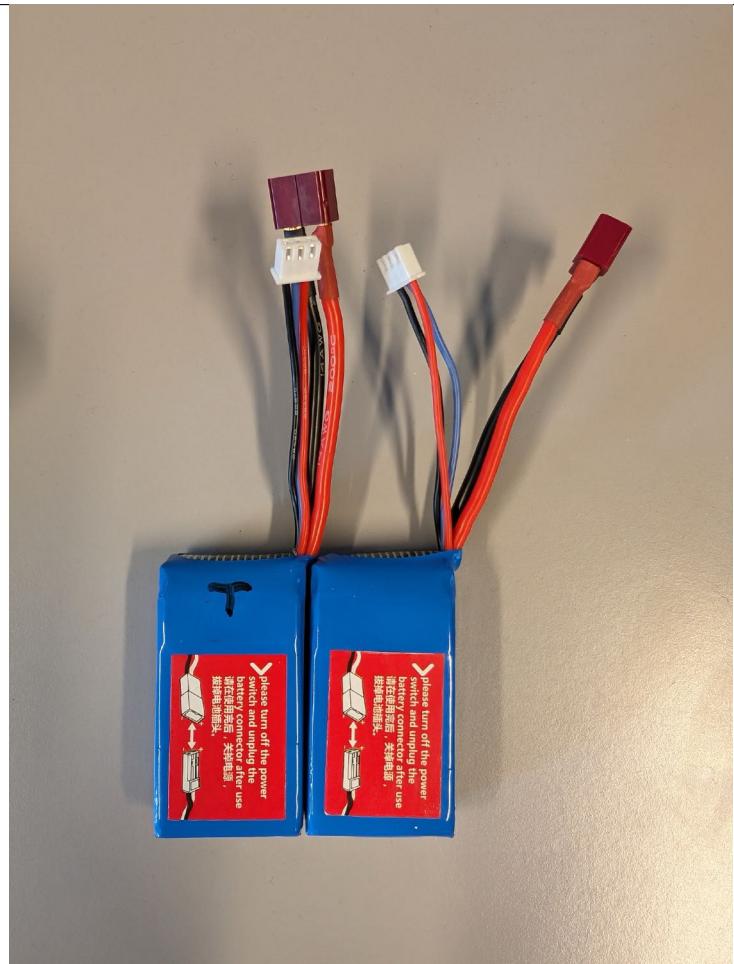


Figure 66: Shape Comparison between the Battery Used in Vacuum Test with Another Battery.

6.25 Solar Cell Panels for Battery Charging in C.R.A.F.T-Sat

Solar energy today stands as a cornerstone among renewable energy sources and has recently gained increasing attention. Notably, its application in space technology, as in CubeSats and other types of satellites, emphasizes its potential. CubeSats, including many 1U models developed by National Aeronautics and Space Administration (NASA), typically integrate solar panels as their primary energy source. The effectiveness of solar panels in space is significantly enhanced by the absence of atmospheric interference, such as clouds and nighttime shadows, allowing them to operate at peak efficiency and generate more power than their earthly equivalents.

Although C.R.A.F.T-Sat will not enter outer space, it aims to ascend approximately 26 km above the Earth's surface. This altitude surpasses the typical height of cloud formations, which generally do not exceed 18 km (<https://www.noaa.gov/jetstream/clouds/four-core-types-of-clouds>), except for rare instances like polar stratospheric clouds that form around 30 km (<https://scied.ucar.edu/image/polar-stratospheric-clouds>). At this elevation, solar panels have more efficiency and generate more power, making them a vital energy source for powering parts of C.R.A.F.T-Sat.

6.25.1 Purpose of Integration of Solar Panels in C.R.A.F.T-Sat

The integration of solar panels in C.R.A.F.T-Sat serves a dual purpose: primarily, it powers an expansion card entirely with clean energy, demonstrating the similarity of renewable sources in near-space conditions. Secondly, it provides an educational platform for students, offering practical insight into how solar panels work, the real CubeSat design, and operations that closely mimic real-world space environments.

6.25.2 Seeed Studio 0.5W Kit solar panel: Technical and Physical Specification

Physical Constraints and Selection Criteria

Choosing a suitable solar panel is not achievable without reviewing the design constraints for C.R.A.F.T-Sat. The size and weight requirement for C.R.A.F.T-Sat forces us to work within the maximum dimensions for C.R.A.F.T-Sat, which is 100x100x100mm, and with a total weight that must not exceed 500g. This requirement requires the selection of solar panels that are both compact and lightweight.

Selected Solar Panel Specifications

After considering various options, a **Seeed Studio 0.5W Kit solar panel** was chosen based on its dimensions of 70x55x3mm, weight of only 15g, availability in the marked, delivery time, and power supply (addressed in the next section), fitting well within the project's physician requirements and meeting at the same time the technical requirements for charging both types of batteries. Table (40) summarizes the physical and technical specifications of the selected solar panel.

Table 40: Seeed Studio Solar Panel Specification

Specification	Value
Dimensions	70 x 55 x 3mm
Weight	15g
Typical voltage	5.5 V
Typical current	100 mA
Maximum load voltage	6.4 V
Maximum Charging Current	600 mA

6.25.3 Design Considerations For Solar Panels in C.R.A.F.T-Sat

The primary role of solar panels in the C.R.A.F.T-Sat project is to charge the batteries efficiently, rather than directly powering the C.R.A.F.T-Sat components. This function requires careful consideration of the voltage output of the solar panels relative to the battery requirements.

Voltage Requirements

For effective charging, the voltage output from the solar panels must exceed the battery voltage. This requirement is important to ensure that the battery can be charged sufficiently. If the voltage in the solar panel is the same as the voltage in the battery, then no current flows and the battery will not charge. The figure (67) shows a simulation of a circuit with two equal voltage sources, represents a solar panel and a battery. The simulation clearly shows a current flow of 0 A on both sides, emphasizing that there is no battery charging/discharging in such a scenario.

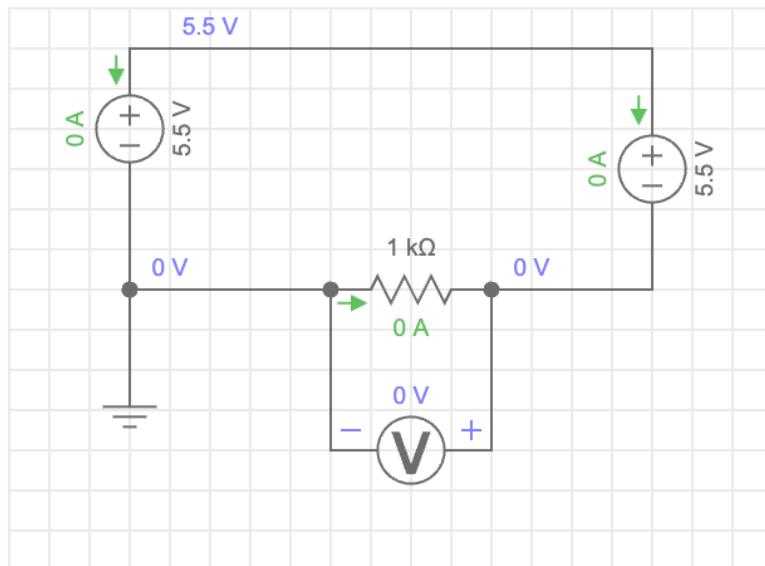


Figure 67: Current Flow Simulation with Two Equal Voltage Sources. | RE

In the worst case, if the solar panel has a lower voltage than the battery, the current will flow in the opposite direction, also from the battery to the solar panel, causing discharging of the battery. The simulation in Figure (68) shows a 7.4 V battery voltage source connected together with a 5.5 V solar panel voltage source. The figure clearly shows the current flow in an undesirable direction from the battery towards the solar panel, which causes the battery to discharge.

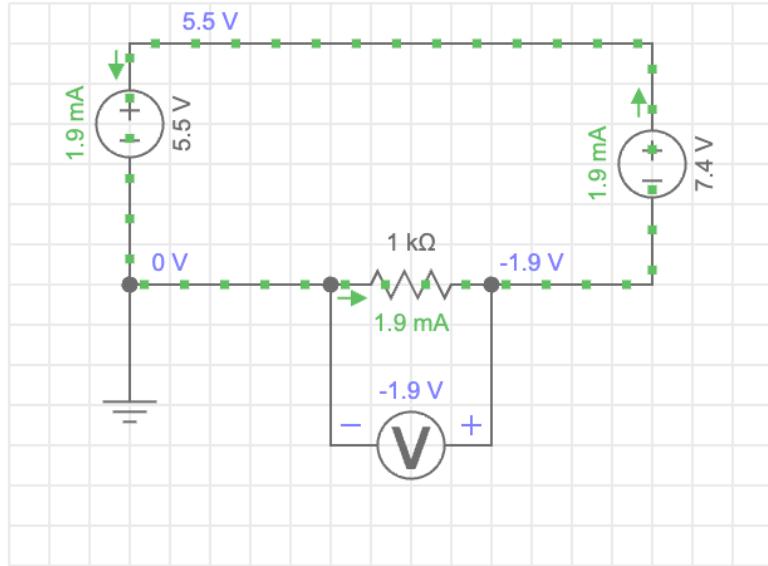


Figure 68: Current Flow Simulation When the Battery Voltage is Greater than The Solar Panel's Voltage. | RE

We must therefore ensure that the voltage of solar panel always is greater than the battery voltage in order to charge the battery correctly and ensuring current flow in the right direction. Figure (69) shows a circuit with a desired current flow when the voltage of the solar panel is greater than the voltage of the battery.

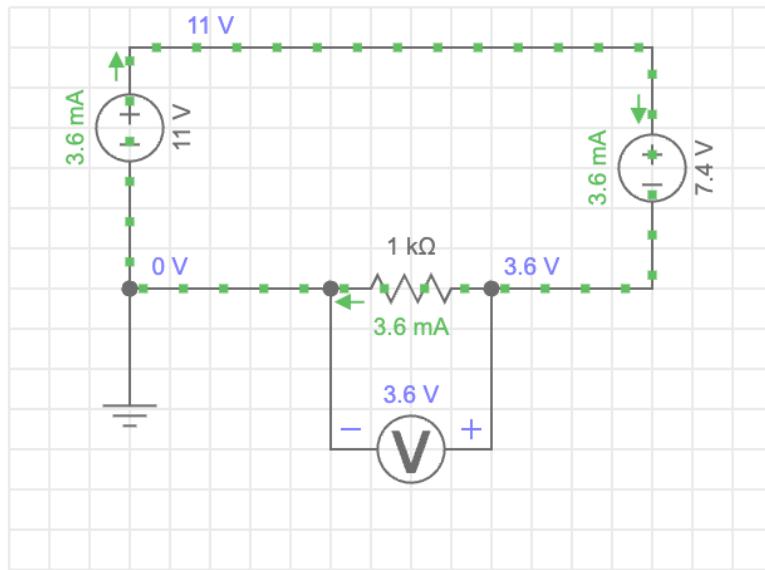


Figure 69: Current Flow Simulation When the Solar Panel's Voltage is Greater than The Battery Voltage. | RE

Voltage Requirements: 3.7 V Battery Charging

The solar panels selected for C.R.A.F.T-Sat typically produce a voltage of 5.5V, which is sufficient to charge a 3.7V lithium-ion battery. Each panel has a maximum output of 100 mA. Connecting the two panels in parallel to each other as shown in Figure (70), the total available current will double to 200 mA, as Equation (33) shows. This configuration will take advantage of the maximum current charge of 600 mA that the battery can receive, enhancing the charging rate which will reduce the charging time. However, the voltage value will remain unchanged at the value of 5.5V, as shown in Equation (34), but is sufficient to charge a 3.7 V battery.

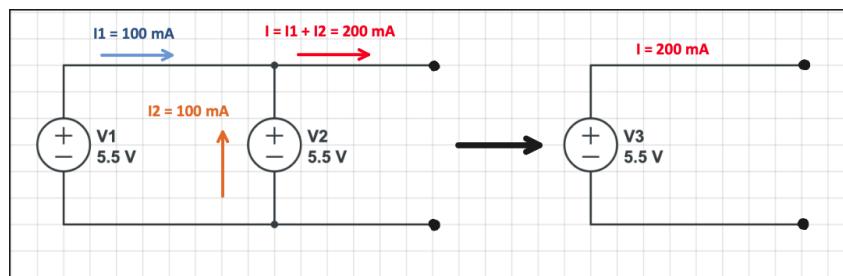


Figure 70: Voltage Sources in Parallel Configuration: Simple Illustration. | RE

$$I = I_1 + I_2 \quad (33)$$

$$\Rightarrow I = 100 \text{ mA} + 100 \text{ mA}$$

$$\Rightarrow I = 200 \text{ mA}$$

$$V_3 = V_1 = V_2 \quad (34)$$

To confirm the accuracy of Equations (33) and (34), a detailed simulation is conducted in **TinkerCad**. In this simulation, both solar panels are arranged as depicted in Figure (70). The maximum current and voltage values in the simulation are chosen to match those typically observed in the selected real solar panel, ensuring a realistic approximation of actual conditions.

Figures (71) and (72) show two solar panels set in a parallel arrangement, illustrating the respective current and voltage values. These figures validate the computations detailed in Equations (33) and (34) concerning the total current and voltage generated when two voltage sources are connected in parallel configuration.

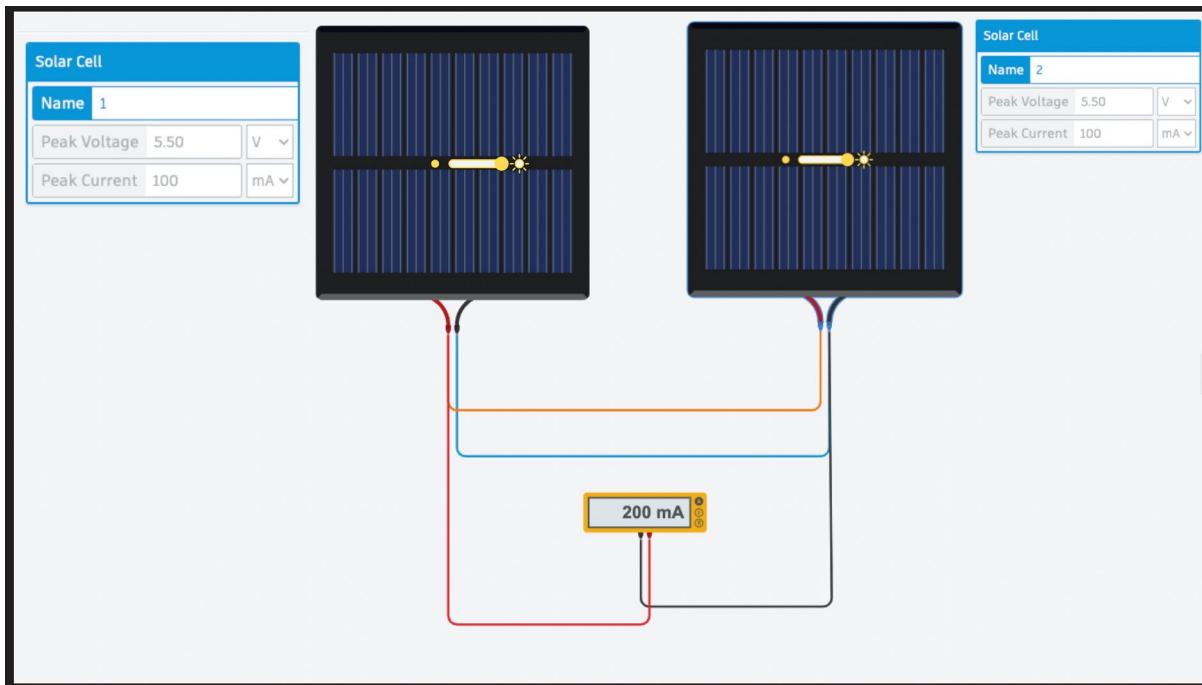


Figure 71: Solar Panels Arranged in Parallel Configuration: Total Current Generated. | RE

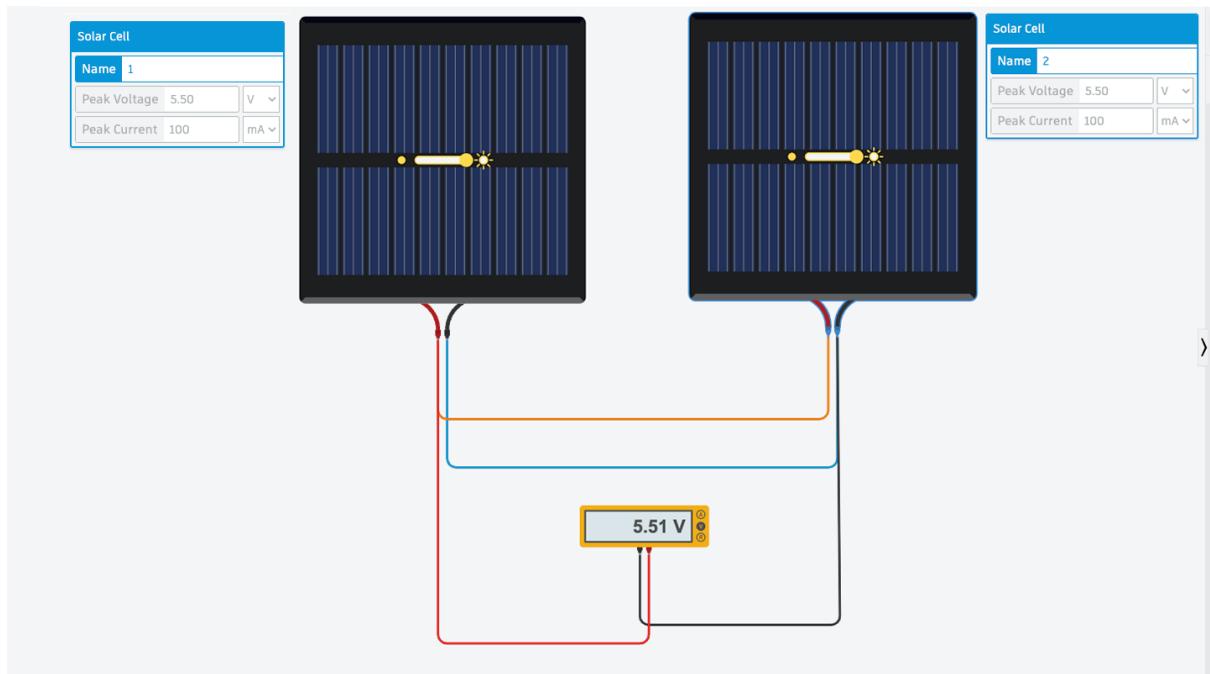


Figure 72: Solar Panels Arranged in Parallel Configuration: Total Voltage Generated. | RE

Voltage Requirements: 7.4V Battery Charging

To charge a 7.4V battery that requires a voltage higher than 7.4V, simply using one or connecting two identical solar panels in a parallel configuration will not increase the voltage output. Consequently, the battery will discharge instead of charging, with the current moving from the battery to the solar panels. The correct approach is to connect the two solar panels in a series configuration, thereby raising the voltage to 11V, as demonstrated in Figure (73) and Equations (35) and (36).

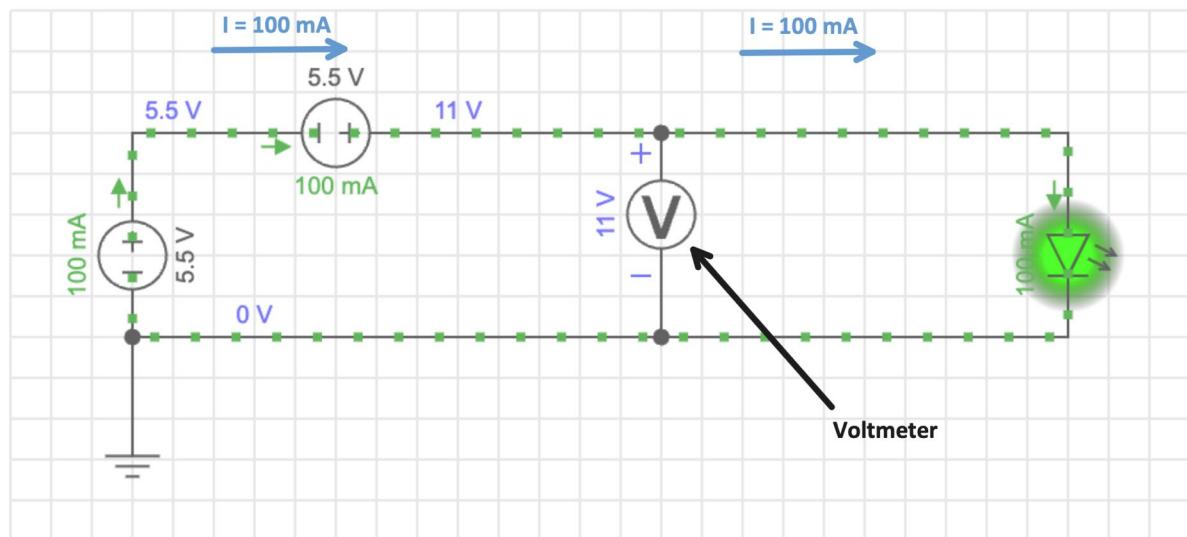


Figure 73: Voltage Sources in Series Configuration: Simple Illustration. | RE

$$V = V_1 + V_2 \quad (35)$$

$$\Rightarrow V = 5.5 \text{ V} + 5.5 \text{ V}$$

$$\Rightarrow V = 11 \text{ V}$$

$$I = 100 \text{ mA} \quad (36)$$

To verify the accuracy of Equations (35) and (36), another detailed simulation is performed in **TinkerCad**. In this simulation, the solar panels are configured in series, as depicted in Figure (73). Like the simulation involving two solar panels in a parallel setup, the peak current and voltage levels in this simulation are selected to align with those typically seen in the chosen real solar panel, ensuring a realistic representation of actual conditions.

Figures (74) and (75) depict a series configuration of two solar panels, demonstrating the resulting voltage and current outputs. These illustrations confirm the calculations presented in Equations (35) and (36), which address the voltage and current produced when two voltage sources are connected in series configuration.

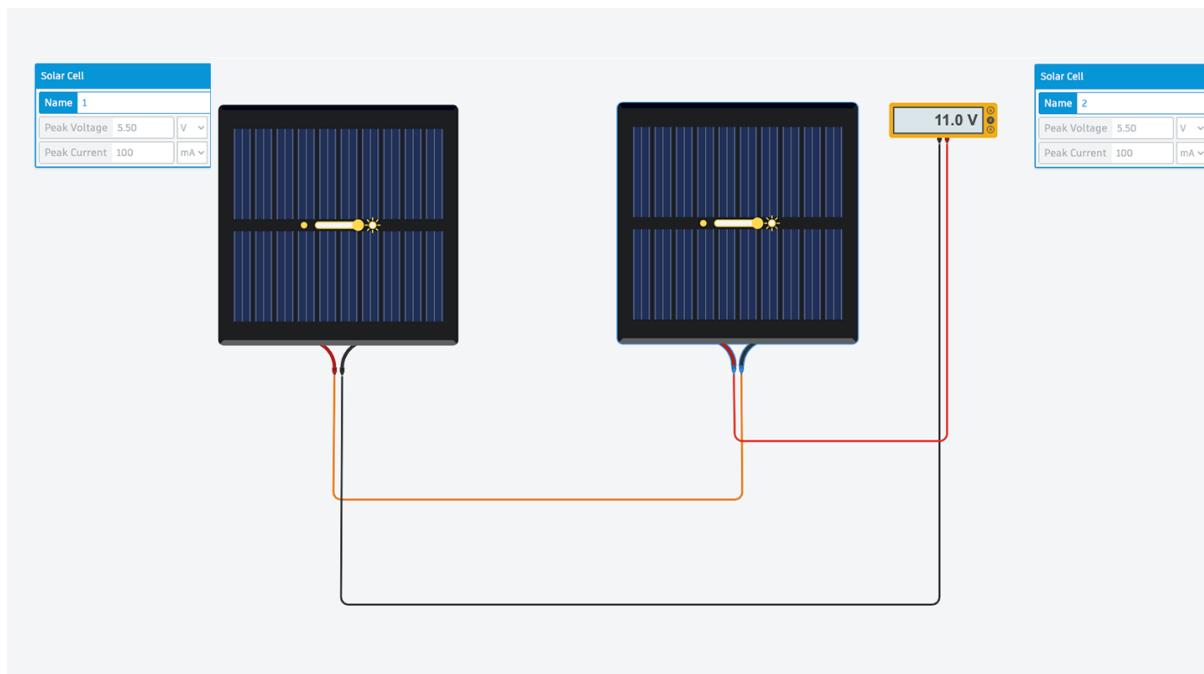


Figure 74: Solar Panels Arranged in Series Configuration: Total Voltage Generated. | RE

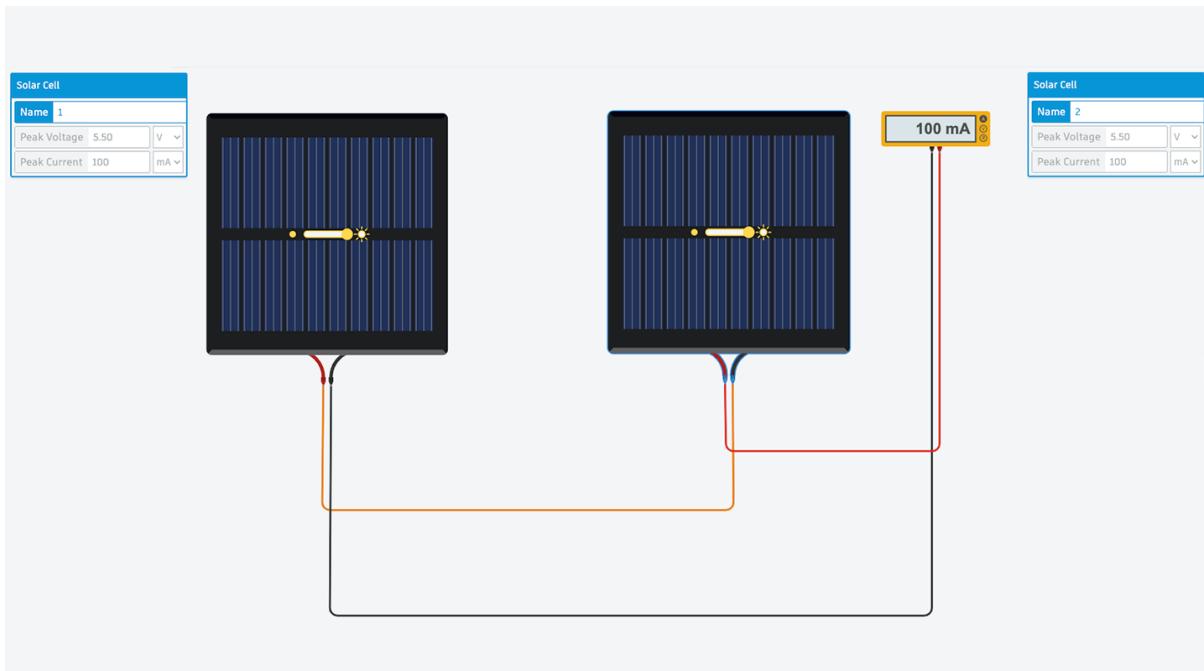


Figure 75: Solar Panels Arranged in Series Configuration: Total Current Generated. | RE

As shown in Figure (74), the voltage produced exceeds indeed the 7.4V needed. Although the series configuration maintains the charging current unchanged at the peak value of 100 mA, the increased voltage ensures that the battery can be charged, even if it will take a longer time compared to the 200 mAh charging current one gets in parallel configuration.

As illustrated in Figure (74), the generated voltage indeed exceeds the required 7.4V. Although the series configuration keeps the charging current constant at a peak of 100 mA, the higher voltage guarantees that the battery can be charged, although at a slower rate than the 200 mA charging current achieved in a parallel configuration.

Choice of Configuration Design: Series or Parallel Configuration

Deciding between a Series or Parallel Configuration is straightforward. If the voltage of the battery to be charged is lower than the output voltage of the solar panels, choosing a series configuration of the solar panels is the optimal choice, as it reduces the charging time. Conversely, if the battery voltage is equal to or higher than that of the solar panels, a parallel configuration becomes essential. This setup enables effective charging of the battery and avoids discharging, although it results in a longer charging duration.

It is crucial to recognize that the values discussed thus far reflect ideal conditions on a warm, sunny day without any shadows blocking the solar panels. On cloudier days, both voltage and current are likely to decrease. At night, these values can drop to zero. Figures (76) and (77) illustrate the reductions in voltage (7V instead of 11V) and current (698 uA instead of 100 mAh) on a cloudy or less sunny day, respectively, while figures (78) and (79) depict how both voltage and current fall to zero in the absence of sunlight or at

night. You might be concerned that, during cloudy days or at night, the voltage could drop below the battery's voltage, leading to unwanted battery discharge. To prevent this and other issues, C.R.A.F.T-Sat will employ a so-called **solar charge controller**, the details of which will be explained in the following section.

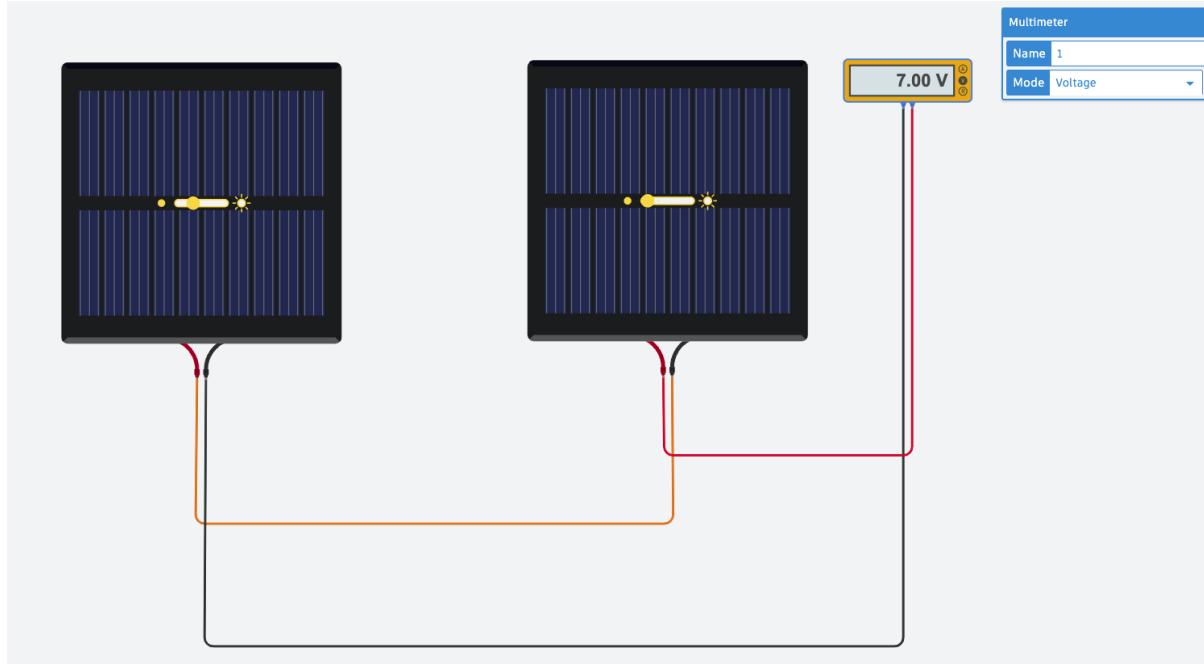


Figure 76: Voltage Reduction on a Cloudy Day: Simulation | RE

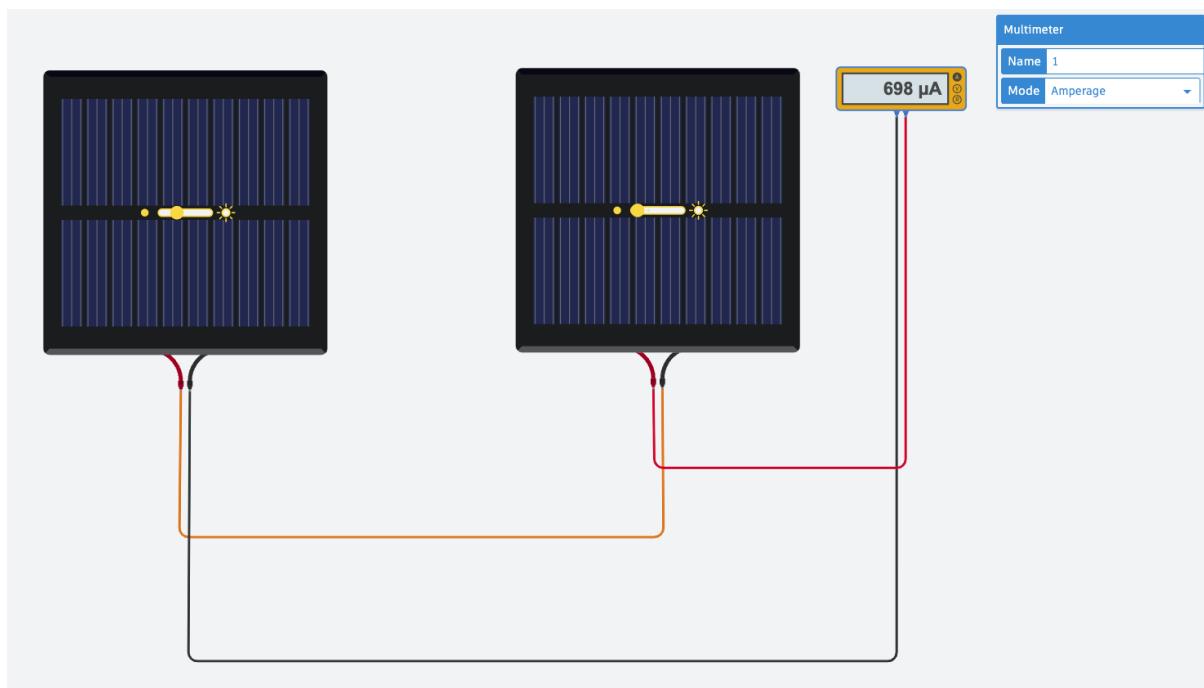


Figure 77: Current Reduction on a Cloudy Day: Simulation | RE

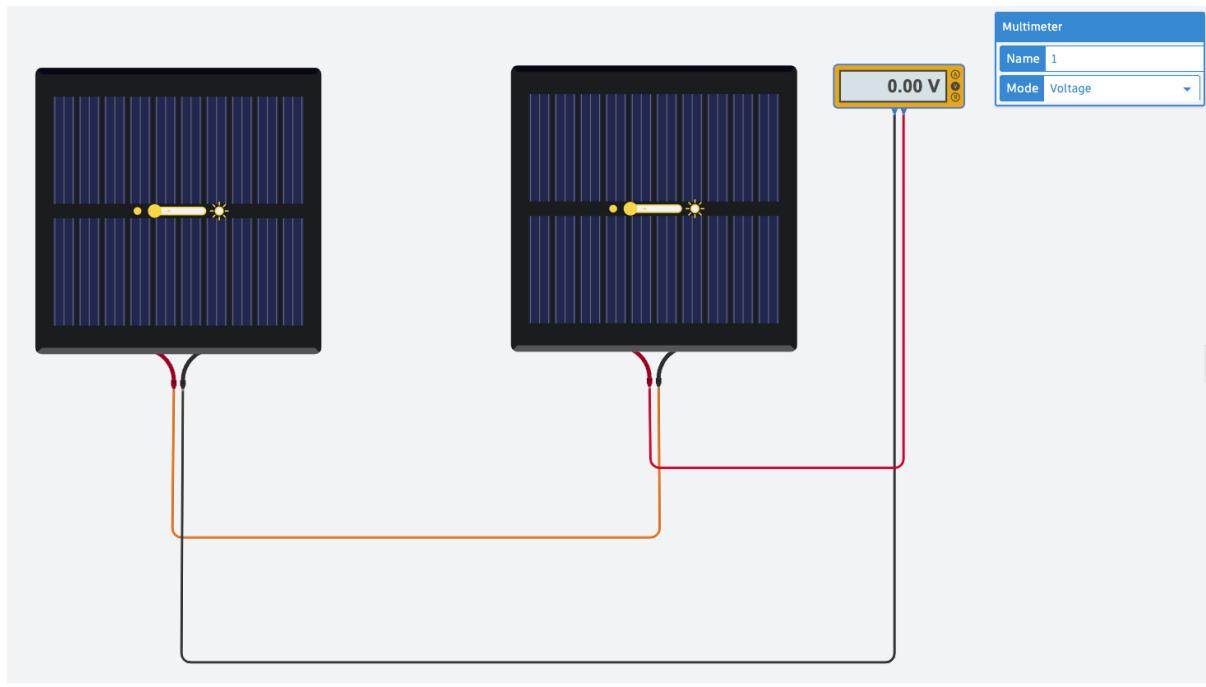


Figure 78: No Voltage Production in Case of Sun Absence: Simulation | RE

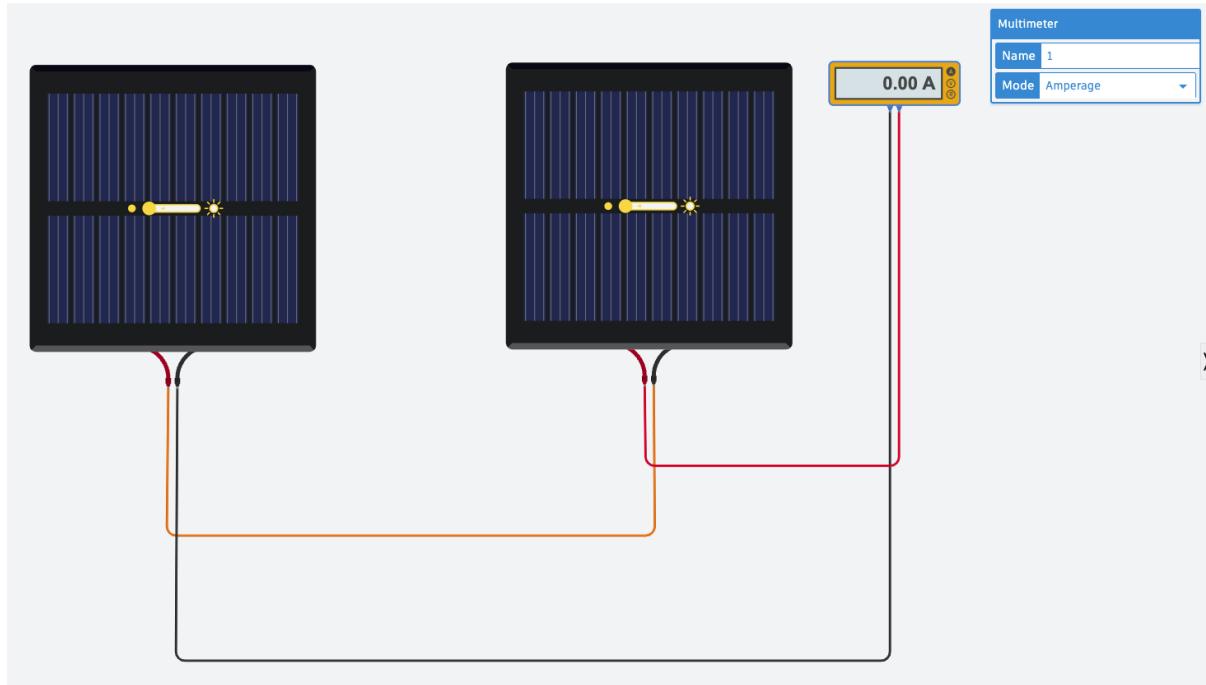


Figure 79: No Current Production in Case of Sun Absence: Simulation | RE

6.25.4 Solar Panels in Series: Voltage Output

Figure (80) illustrates the voltage output of two solar panels linked in series. The individual voltages of the panels were 5.71V and 5.17V as shown in Figures (81) and (81),

respectively. In a series setup, the combined voltage output is increased to 11.81V, as shown in Figure (80) , verifying the configuration setup for connecting two solar panels in series.



Figure 80: Solar Panels Arranged in Series Configuration: Total Voltage Outout in Test.

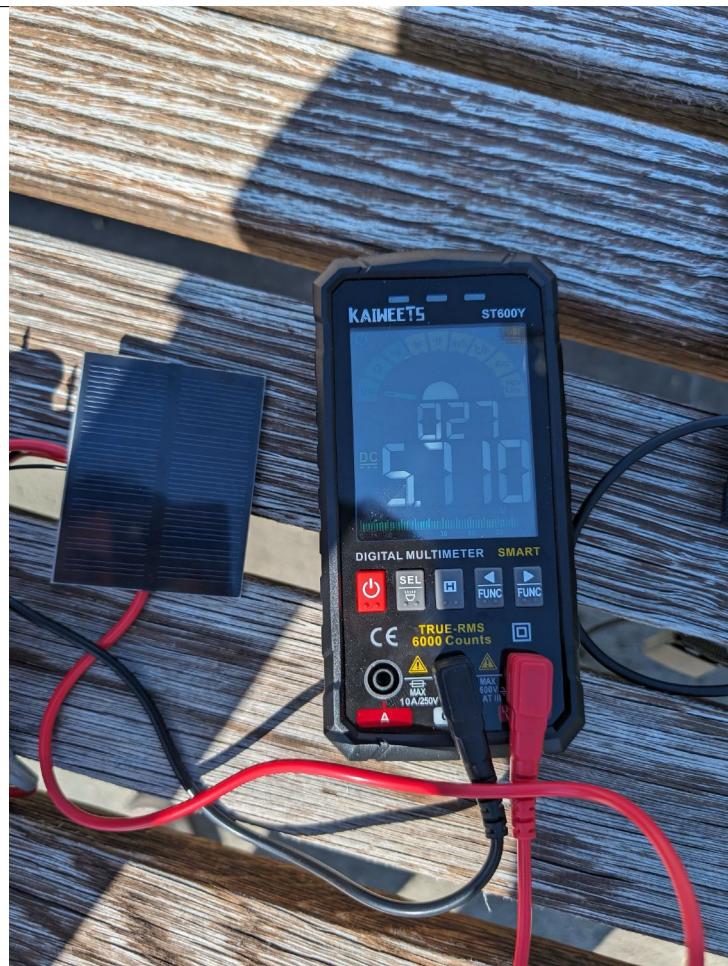


Figure 81: Voltage Output of Solar Panel 1 for Testing



Figure 82: Voltage Output of Solar Panel 2 for Testing

Similarly, following the same procedure as shown in Figure (72) to test two solar panels

in parallel configuration.

6.25.5 Solar Charge Controller

The solar charge controller plays a crucial role in solar powered systems, where it protects batteries from being overcharged or discharged excessively, which could cause permanent damage to batteries. The solar charge controller regulates the voltage from the solar panels and controls the current flow from the solar panels to the batteries.

Overcharging is a frequent problem in LiPo batteries, leading to swelling or, in extreme cases, to fire. In addition to that, LiPo batteries naturally discharge over time. If left uncharged for extended periods or not charged at the correct voltage, they may drain to a voltage level that is harmful to the battery's health, making it difficult or impossible to recover.

Selection of DFRobot Solar Power Manager 5V as a Solar Charger Controller

In a project like C.R.A.F.T-Sat, which faces numerous physical and technical limitations, the choice of an appropriate solar charge controller is essential for proper battery management and protection. After a comprehensive evaluation of different models, the **DFRobot Solar Power Manager 5V**, shown in Figure (83), was selected as the optimal solution. This selection was influenced by the controller's functionality, availability in the market, and its unique features.

DFRobot Solar Power Manager 5V: Selection Criteria

The choice of the DFRobot Solar Power Manager 5V was influenced by several factors.

1. **Functionality:** This controller offers a robust set of features that are essential for managing the energy gained by a solar panel. It not only regulates the charging of the battery to maximize efficiency, but also ensures that the power supply remains stable under varying conditions.

2. Protection Features:

- **Overcharge Protection:** Safeguard the battery from being charged beyond its maximum voltage, which is crucial for preventing damage and extending battery's lifespan.
- **Overdischarge Protection:** Ensures that the battery voltage does not drop below a safe threshold, protecting it from deep discharge damage. **Reverse Connection Protection:** Protects the electronics from damage in case of incorrect wiring during setup.
- **Overcurrent Protection:** Monitors and limits the current to prevent enormous draw that could lead to overheating and potential hardware failure.
- **Maximum Power Point Tracking (MPPT):** It includes MPPT which enhances the efficiency of the solar panel by continuously adjusting the load

to ensure that the solar panel operates at its optimum power output under different sunlight conditions.

3. **Availability:** The DFRobot Solar Power Manager 5V is readily available in the Norwegian Market, which ensures that there are no delays in project timelines due to supply issues.
4. **Size and Weight:** By size of **33.0mm × 63.0mm** and weight of **20 g**, the controller is considered as a compact and light weight unit, which makes it ideal for C.R.A.F.T-Sat's projects.

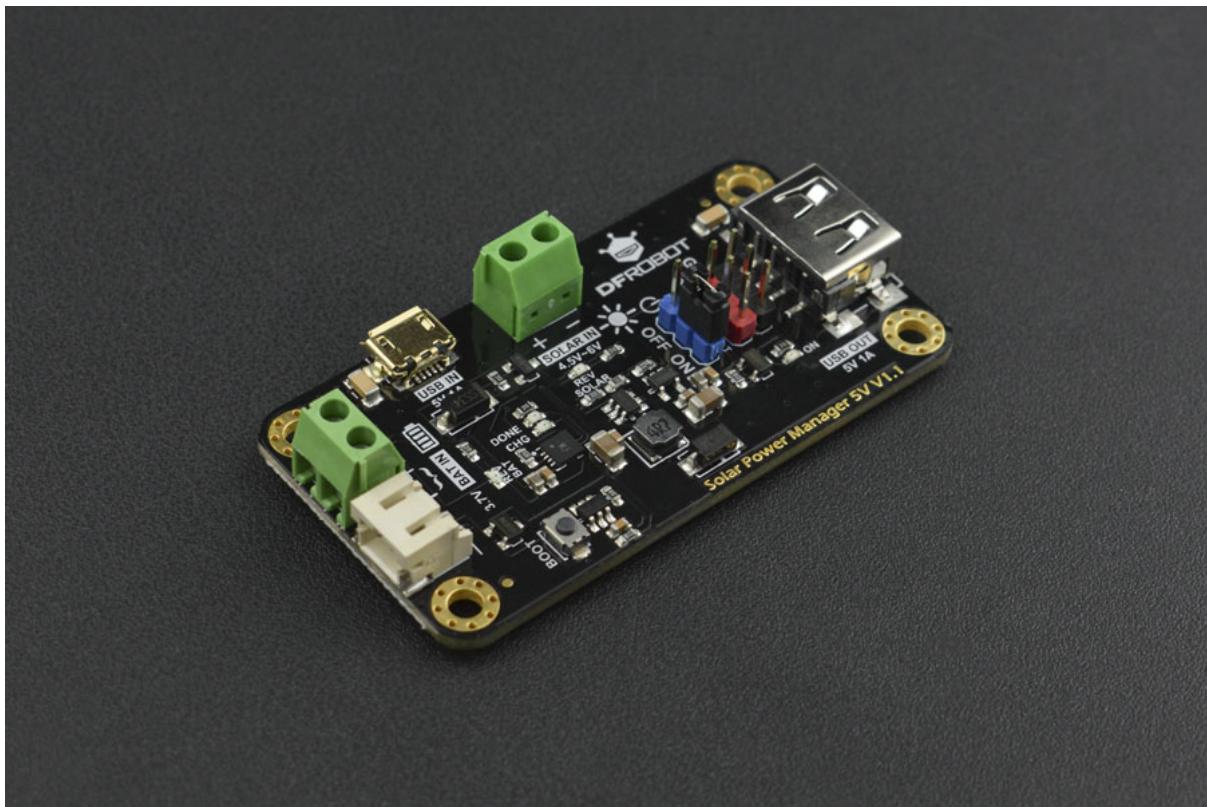


Figure 83: DFRobot Solar Power Manager 5V

Comparison With Other Solar Charge Controllers

Other solar charge controllers such as the **TP4056** shown in Figure (84) and **DFRobot DFR0264** shown in Figure (85) were also considered. The **TP4056** is known for its simplicity and effectiveness in charging LiPo batteries but lacks features such as MPPT. The **DFRobot DFR0264** does not provide the same level of integrated protection and the MPPT functionality as the 5V Solar Power Manager.

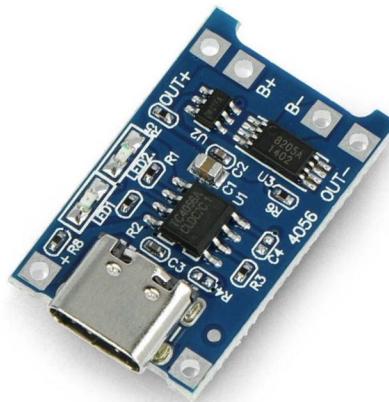


Figure 84: TP4056 from SEEED Studio



Figure 85: DFR0264 Solar Charger Controller

DFRobot Solar Power Manager 5V: Constraints

Although the chosen solar charge controller offers various safety features, there are certain compromises involved in its selection. The primary limitation is that the DFRobot Solar Power Manager 5V is only compatible with single-cell batteries and is specifically recommended for use with 3.7V lithium batteries. This implies for C.R.A.F.T-Sat that it cannot be used with a two-cell 7.4V battery. Such limitations are typical for solar charge controllers of this size and weight, which are designed to work exclusively with 3.7V LiPo batteries. The plan is to use a 3.7 LiPo battery to operate the expansion card that contains the solar charge controller. However, it is crucial to note that this integration requires a redesign of the C.R.A.F.T-Sat battery to accommodate the 3.7 LiPo battery.

6.25.6 DFRobot Solar Power Manager 5V: Main Features Explained Simply

Understanding some main features and their functionalities of DFRobot Solar Power Manager 5V is an excellent educational tool, particularly for college students. Before presenting some simple circuits that perform some of the functions selected Solar Charge Controller (SCC), we introduce several key purposes and benefits of creating and using such simplified circuits in educational content.

Educational Objectives and Benefits

1. **Hands-On Learning:** Building and testing simple circuits provides hands-on experience that complements theoretical learning. By observation can the students gain a better understanding of electronic components and circuit design.
2. **Understanding Basic Electronics Principles:** By designing a circuit that includes fundamental features like charge protection and power indication, students can learn about basic electronics functions such as voltage regulation, current control, and circuit safety mechanisms.
3. **Inspiration for Innovation:** Engaging with simple circuits can inspire students to think creatively about how to improve technologies for new uses. This type of innovation is a milestone of engineering.
4. **Cost-Effective Learning Tool:** Simple circuits are often more cost-effective compared to more complex projects.
5. **Preparation for More Complex Projects:** Starting with simpler circuits prepares students for more complex electronics projects.

Indicator for Battery Charging and Full Charge Status)

Figures (86) and (87) illustrate a basic battery indicator circuit for charging and fully charged states, respectively. We will examine the key components of these circuits in detail:

Circuit Main Components:

- **Solar Cell Panel 5V:** Is represented with 5V voltage source,
- **Battery 3.7V:** Is represented with 3V when it is empty and with 4.2V voltage source when it is fully charged.
- **Red LED** Indicates charging status.
- **Green LED:** Indicated fully charged status.
- **Resistors:** Control the current through each LED to prevent damage.
- **Diodes:** To prevent reverse current.
- **Variable Resistor:** Simulates variable solar intensity affecting the charging current.
- **NPN Transistor:** Used as a switch to control the activation of the green LED, indicating that the battery is fully charged.

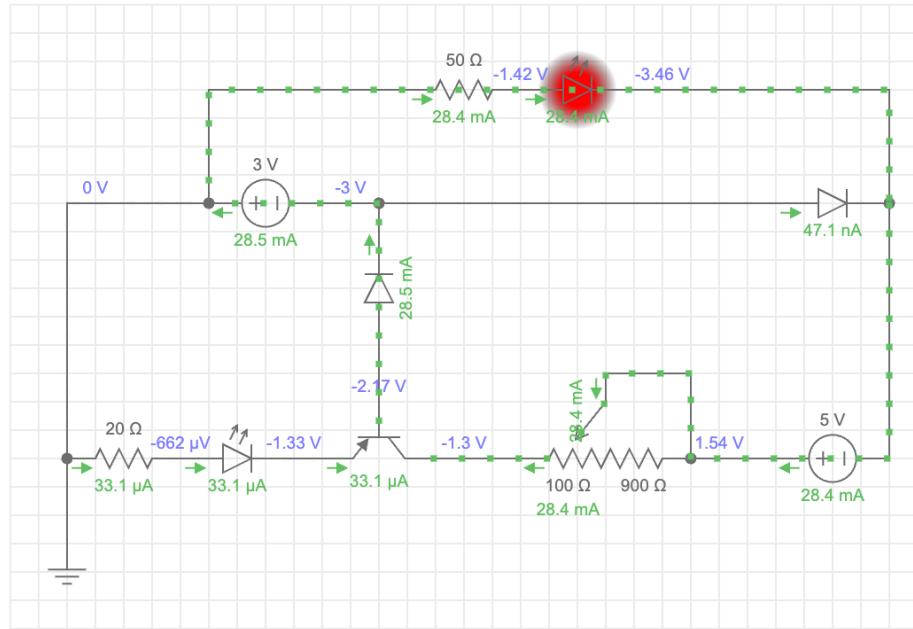


Figure 86: Simple Circuit: Battery Charging Indicator with Red Light Emitted Diode

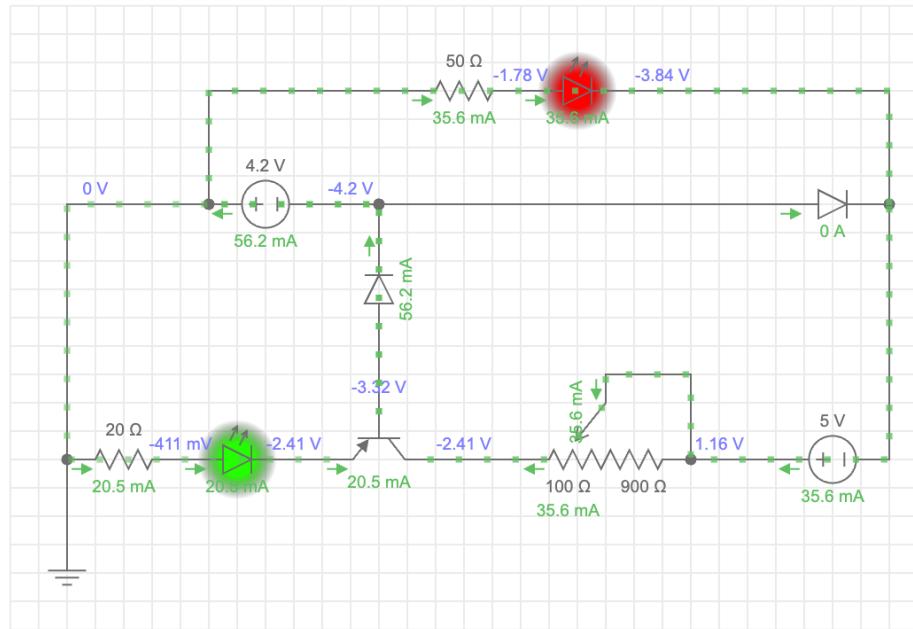


Figure 87: Simple Circuit: Full Charged Battery Indicator with Green Light Emitted Diode

Circuit Work Method

-
1. When the battery voltage is around 3V, indicating an empty battery, as shown in Figure (86), the circuit allows current to flow through the battery and the red LED, which illuminates the red LED, indicating that the battery is currently charging.
 2. As the battery voltage rises and nears 4.2V, as illustrated in Figure (87), the green LED fully illuminates once the NPN transistors permit current flow through the green LED, indicating that the battery is completely charged.

Battery Protection Again Discharging

The circuit illustrated in Figure (88) shows a low voltage cut-off mechanism during discharge to protect the battery from permanent damage. The circuit deactivates the load at 3.3V and reinstates operation at around 3.6V.

Circuit Main Components:

- Voltage Sources:
 - Main Battery Voltage Source: This represents the battery. It is modeled as 4V voltage source in Figure (88).
 - Reference Voltage Source (2.5V): This voltage source is used to provide a reference voltage to the comparator. It helps in setting the threshold at which the circuit acts to disconnect the load.
- **Comparator:** It compares the battery voltage to the reference voltage. If the battery voltage after went through resistor falls below the reference voltage, the comparator's output changes state.
- **NPN Transistor:** The transistor functions as a switch controlled by the output of a comparator. When the transistor detects that the battery voltage is higher than the reference voltage, it allows current to flow to the load (lamp). If the voltage falls below the reference level, the comparator output deactivates the transistor, stopping power to the load, and preventing additional battery discharge.
- **Capacitor:** Stabilized the comparator's input by filtering out voltage spikes.
- **Ramp Voltage:** is used to simulate varying battery voltage to test how the circuit responds as the battery discharges.

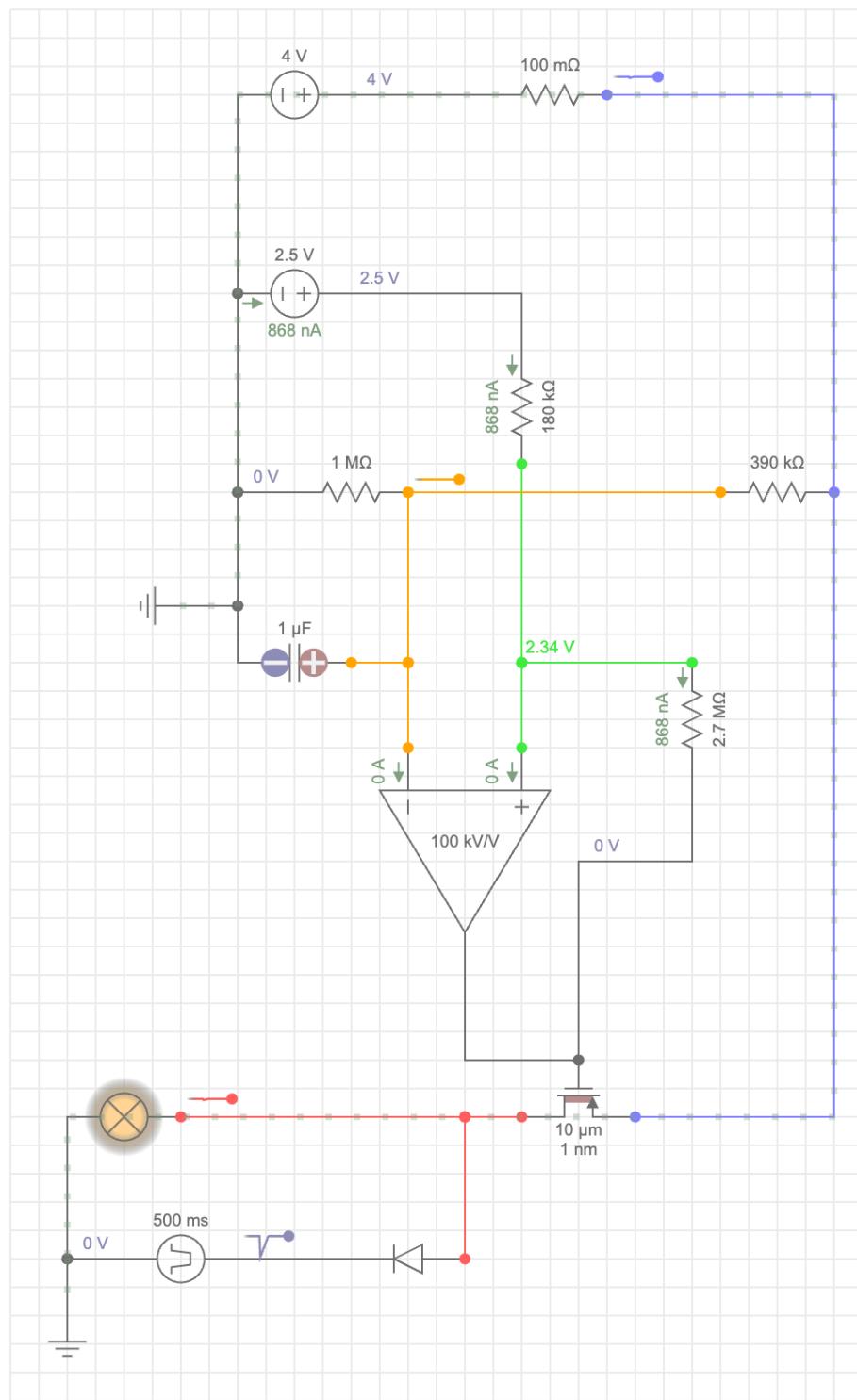


Figure 88: Simple Circuit: Battery Protection Again Discharging

It is important to mention that the DFRobot Solar Power Manager 5V includes additional safety features such as overcharging protection. We believe that it is sufficient to present

a circuit for battery indicator and discharging protection because of its importance.

6.25.7 Using Solar Charge Controller to Charge 3.7V Battery With Solar Panel

To test the reliability of the solar charge controller, both the solar panel and the 3.7V battery are connected as shown in Figure (89). Figure (90) shows a red led light that indicates that the battery is currently charging with the solar panel.

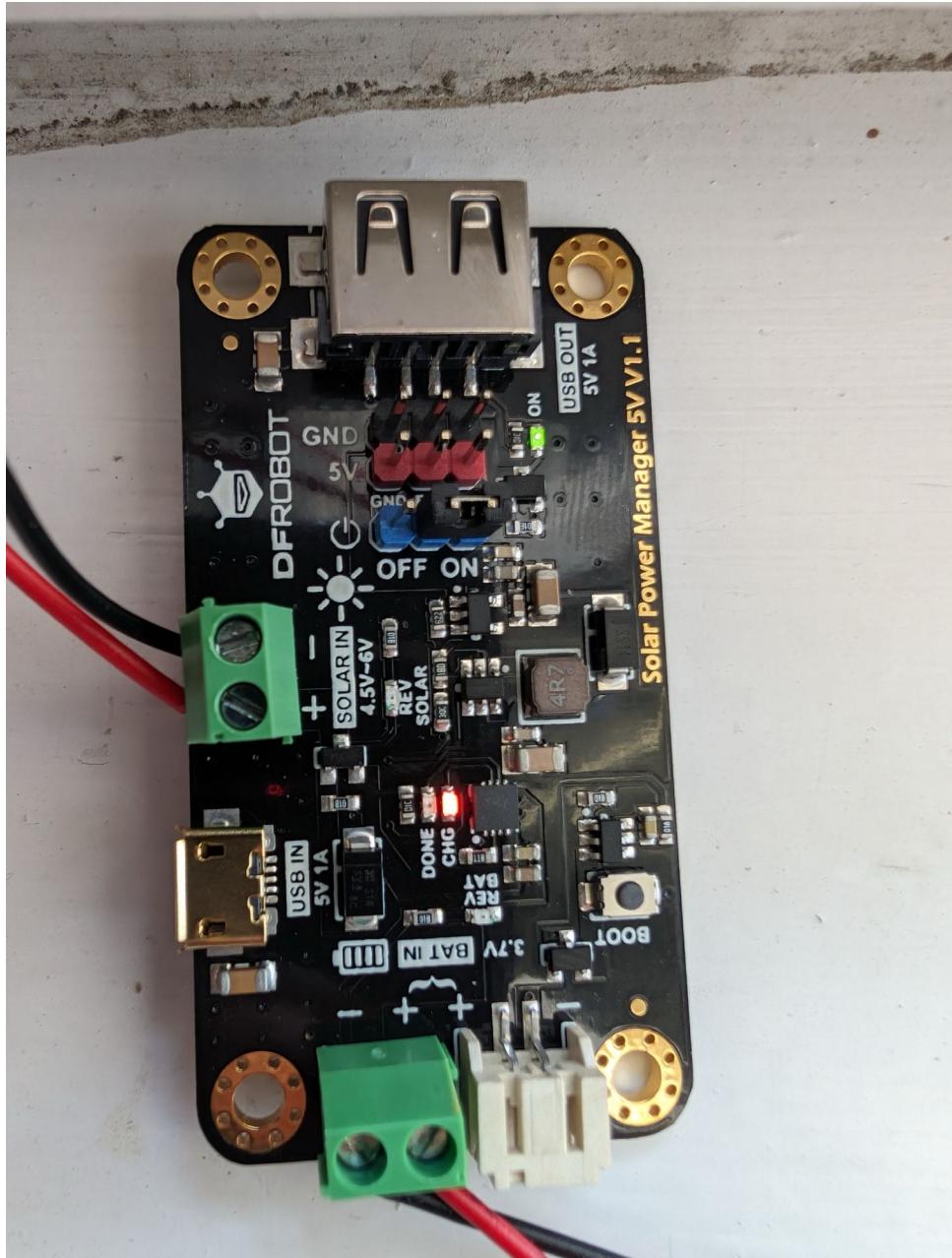


Figure 89: Charging Indicator using Solar Charge Controller

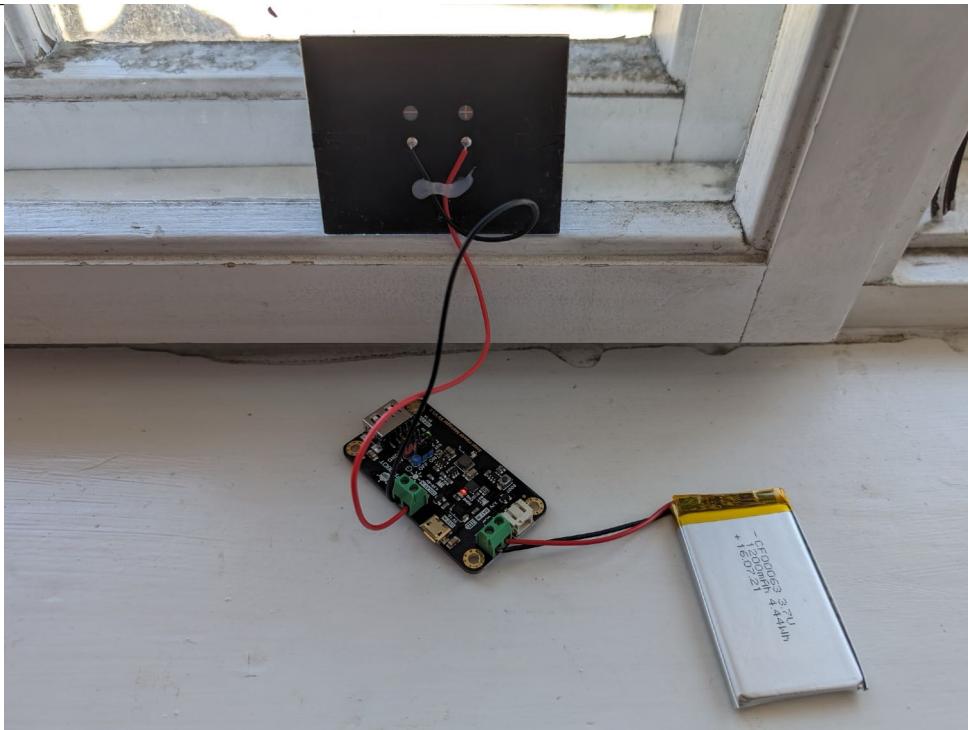


Figure 90: Charging Battery With Solar Panels Using Solar Charge Controller

6.25.8 Simple Design Illustration of C.R.A.F.T-Sat with Solar Panels

To provide a clearer presentation of the solar panels used in C.R.A.F.T-Sat, two basic models of C.R.A.F.T-Sat created in **TinkerCad** are displayed in Figures (91), (92), and (93). These figures demonstrate the implementation of a single and a dual solar panel on each side of C.R.A.F.T-Sat, respectively. Figure (91) features a single solar panel that simplifies design and minimizes costs. In the best-case scenario, the maximum current achievable is limited to 100 mA, requiring a lengthy charge time of about 12-14 hours for a 1200 mAh battery. In contrast, Figure (92) and (93) illustrate a set-up with two solar panels, one on each side. By arranging these panels in a parallel configuration, as depicted in Figures (70) and (71), the current capacity increases to 200 mA. This enhancement reduces the charging time to approximately 6-7 hours, offering significant advantages in certain scenarios.

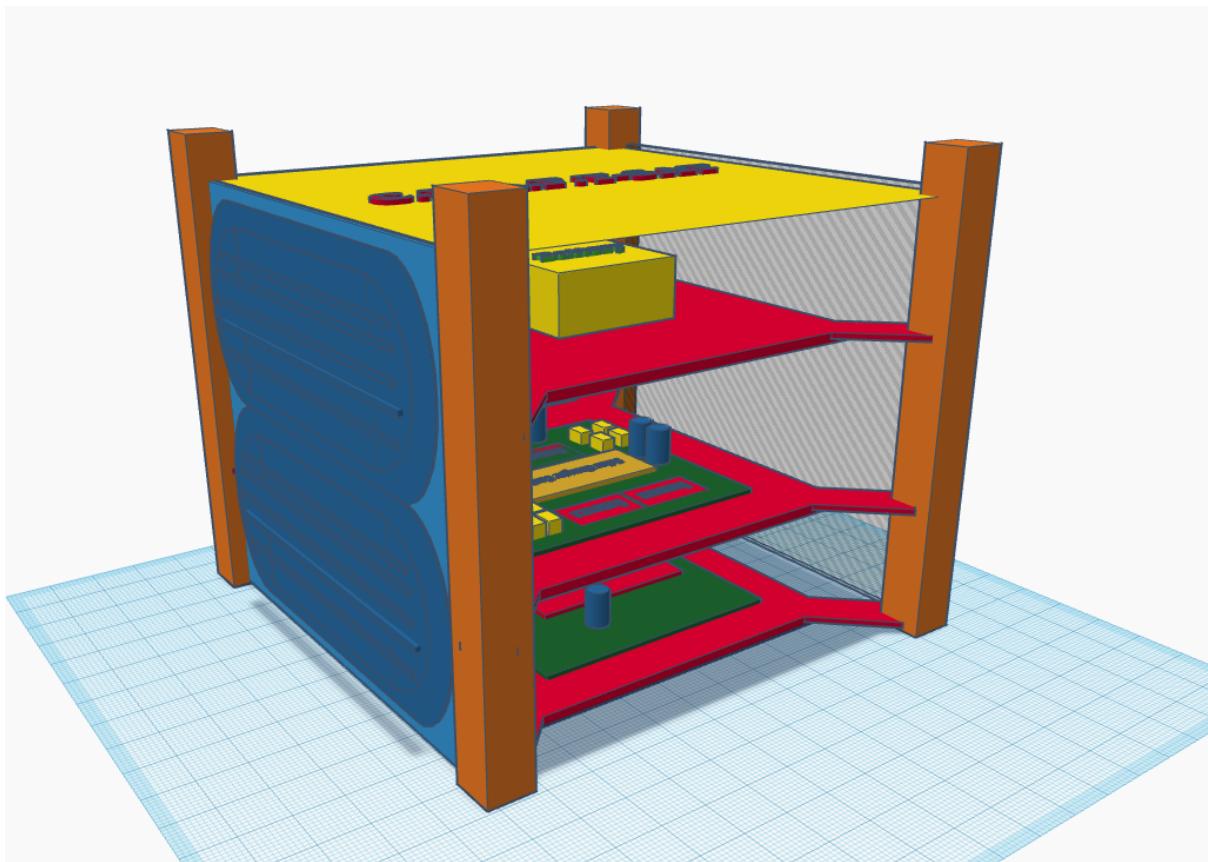


Figure 91: Simple Illustration of C.R.A.F.T-Sat features a single solar panel. | RE

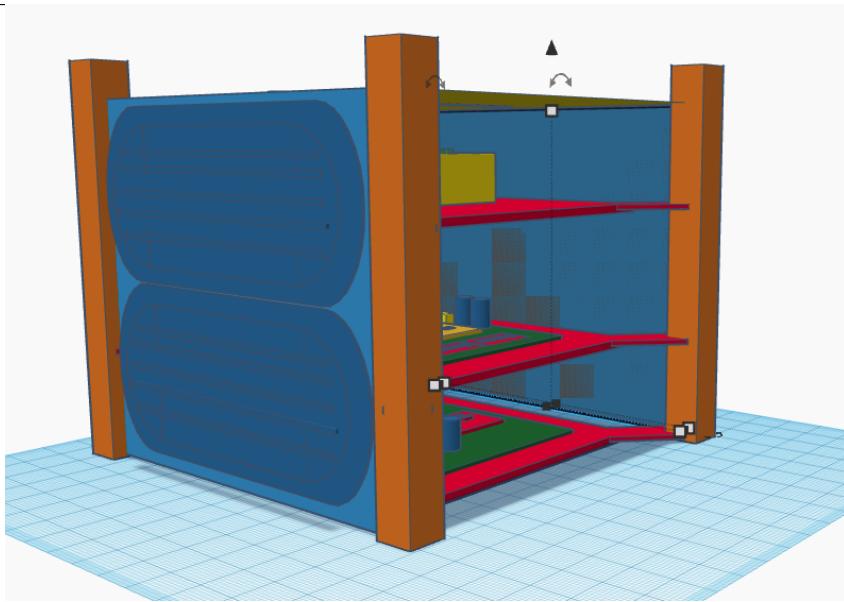


Figure 92: Simple Illustration of C.R.A.F.T-Sat features two solar panels: Left Side. | **RE**

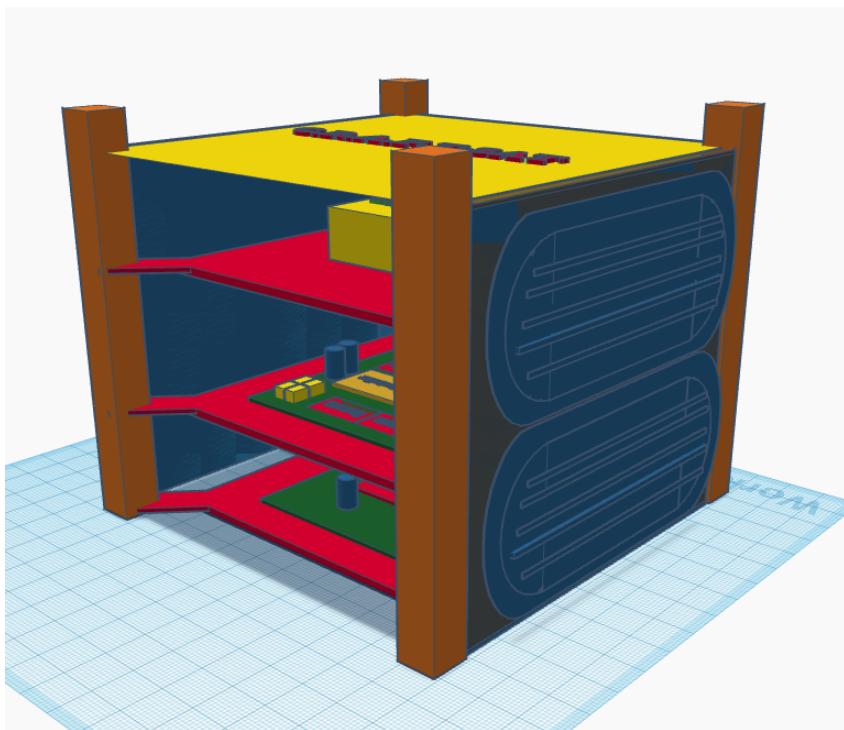


Figure 93: Simple Illustration of C.R.A.F.T-Sat features two solar panels: Right Side. | **RE**

6.25.9 Solar Panel Exclusion and Future Opportunities

Despite comprehensive and detailed planning for the integration of solar panels into the project C.R.A.F.T-Sat, the final system will not include solar panels. This decision is primarily driven by stringent weight constraints and specific power requirements as set

by ASE.

Weight and Power Requirements The total weight of C.R.A.F.T-Sat must not exceed 500 grams, a limit that requires careful consideration of the mass of every component. Furthermore, the main PCB card in CanSat and C.R.A.F.T-Sat requires to be able to be operated by 5V, in addition to 3.3V, forcing us to use a 7.4V battery to fulfill this requirement.

Documentation Value and Future Implementation

The detailed documentation created during this project phase serves several important purposes.

1. **Implementation Guide for Electronics:** The implementation guide for integrating solar panels provides a clear step-by-step structure that can be easily followed by ASE.
2. **Educational Resource:** This documentation serves as a solid educational resource for students in the college. It offers insight into the design considerations, technical challenges, and potential benefits of using solar panels in CubeSat.

6.26 Solar Cell Panels for Battery Charging in CanSat

6.26.1 Solar Panel Integration In CanSat

The incorporation of solar panels requires careful consideration of the CanSat's design and requirement, including:

- Panel Positioning: It is crucial to optimize the placement of solar cells to maximize exposure to sunlight while not interfering with CanSat's operational components.
- Weight and Size Constraints: The addition of solar panels might not impact the weight and size significantly, preserving breaking the weight and size requirement.
- Cost: The cost element is crucial in capturing the stakeholders' attention towards this solution.

6.26.2 Challenges and Possible Solutions

The benefits of incorporating solar panels into CanSat are substantial and appealing. However, there are significant challenges to overcome for successful integration. CanSat's weight is constrained to a range of 300 to 350 grams, making it difficult to accommodate all components such as sensors, microcontroller, solar panel, and battery. Furthermore, the solar conditions in countries like Norway are highly variable and not always conducive to the efficient operation of solar-powered devices. Thus, the primary goal of the battery is not to entirely replace conventional batteries but to supplement them with environmentally friendly and renewable energy sources, offering the convenience of battery charging without the need for disconnecting it. This integration is expected to inspire other innovators to devise new and more sustainable ideas, not only for CanSat but for various satellite applications. Another obstacle is the limited availability of solar panels that meet the specified size and weight criteria. Although some options are found in the Chinese market, doubts remain regarding their performance compared to the claims in the documentation. In the event that obtaining a suitable solar panel becomes unfeasible during the project timeline, we will ensure the provision of comprehensive documentation and visual representations. This will enable ASE to seamlessly take over the project, requiring only the procurement and installation of a solar panel. Nevertheless, our ultimate objective remains to complete the CanSat project with a functioning solar panel.

6.26.3 Strategic Placement of Solar Cells on CanSat

In the search for efficient charging solutions for CanSat systems, the strategic placement of solar cell panels is central. Three innovative configurations have been developed to maximize energy capture from the Sun, ensuring a steady charging current to the CanSat battery system. Each configuration presents its own advantages and is designed to suit different mission profiles and environmental conditions.

First Iteration: Top-Mounted Solar Panel

The initial design, shown in Figure (94), includes a solar panel positioned on the upper side of the CanSat. This setup leverages the CanSat's upward orientation, which is frequently in direct sunlight. However, due to limitations on charging capacity imposed by the mission, incorporating a parachute may not significantly impact the CanSat's maximum flight duration of up to 170 seconds, especially when the battery is adequately charged. Moreover, the mounted area can be adjusted to be opened to receive more sunlight when not in operation, and closed during operation.

Advantages:

- Maximizing exposure occurs when the CanSat maintains a stable vertical orientation both during the mission and while outside.
- Facilitates wiring and installation because of its fixed mounting.
- Reduces the risk of damage under the mission because of its fixed mounting.

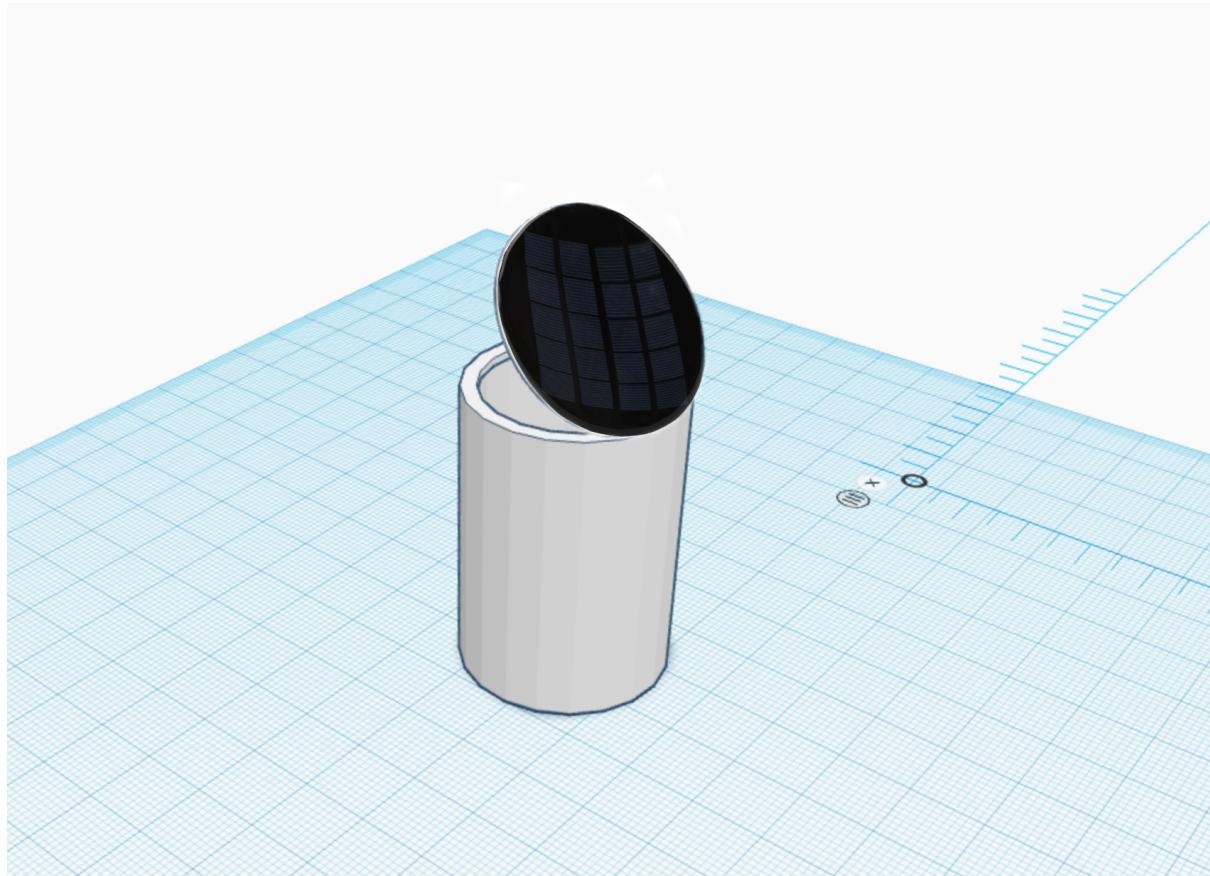


Figure 94: Top-Mounted Solar Panel in CanSat

Second Iteration: Elastic Solar Panel Around The CanSat

The next idea involves wrapping the cylindrical body of the CanSat with an elastic solar cell panel, as illustrated in Figure (95). This approach guarantees that regardless of how the CanSat is positioned upon landing, a section of the solar array will probably receive sunlight. In this scenario, the exposed area is expected to be larger as well.

Advantages:

- Offers a greater surface area for energy collection.
- Provide continuous charging capability regardless of CanSat's orientation.
- Possibility of utilizing the entire body of the CanSat, maximizing space efficiency.

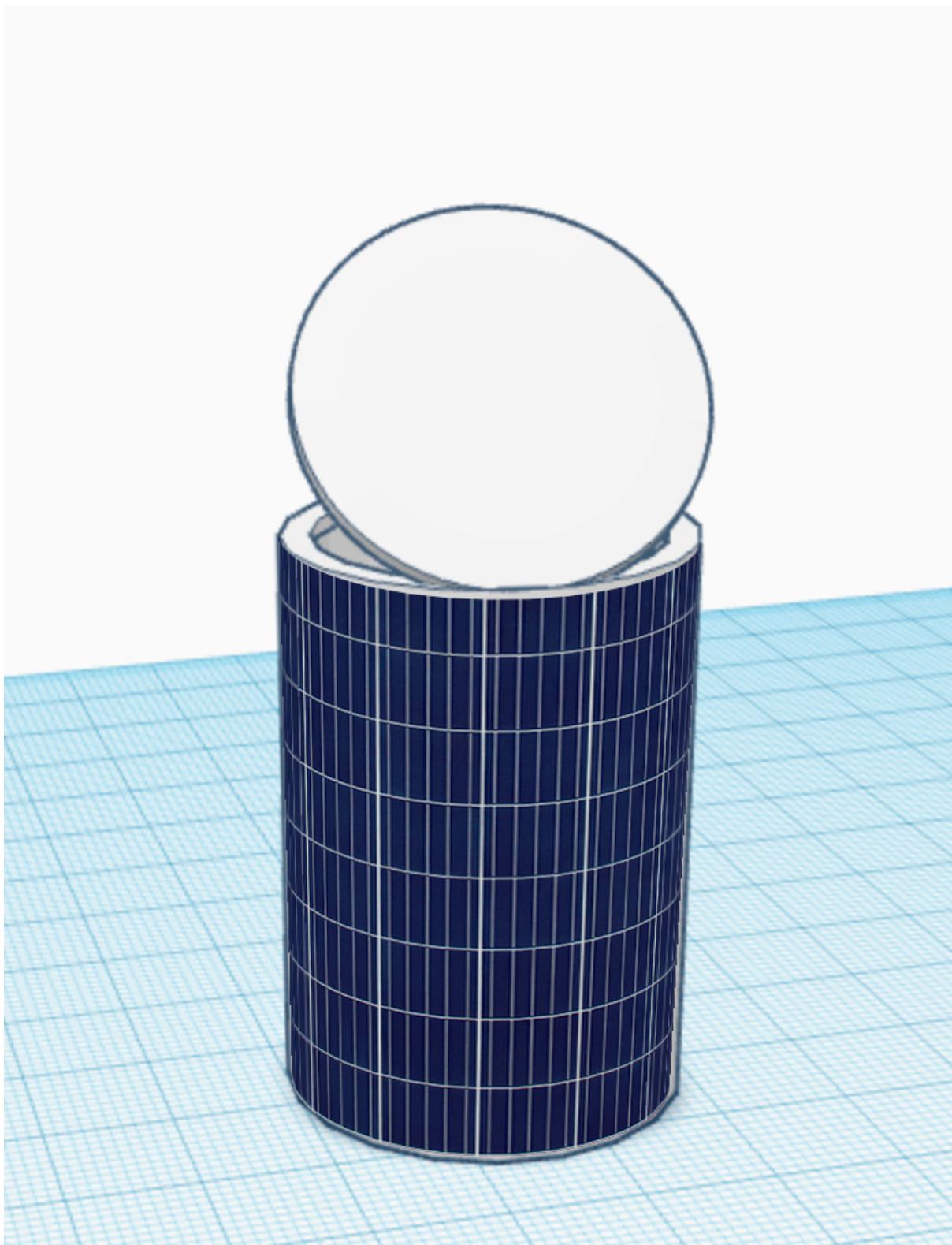


Figure 95: Elastic Solar Panel Around The CanSat

Third Iteration: Hybrid Dual Configuration

By integrating the advantages of the initial two concepts, the third version incorporates

both the upper panel and the cylindrical solar panel that wraps around as shown in Figure (96). This combination strategy guarantees optimal power production, as the upper panel ensures a consistent charge when the CanSat is in an upright position, while the wrap-around panel enhances energy absorption, particularly beneficial during tumbling or when the CanSat is not ideally positioned. The only limitation that could restrict these solutions is the weight factor. One way to address this is to employ a smaller solar panel that does not need to completely encircle the CanSat frame.

Advantages:

- Guarantees the best possible sunlight exposure in different orientations and locations.
- Enhances overall power generation, ensuring a consistent supply of energy in various situations.
- Improves the redundancy of the power system, guaranteeing that in case one panel is blocked or harmed, the other panel can still operate.

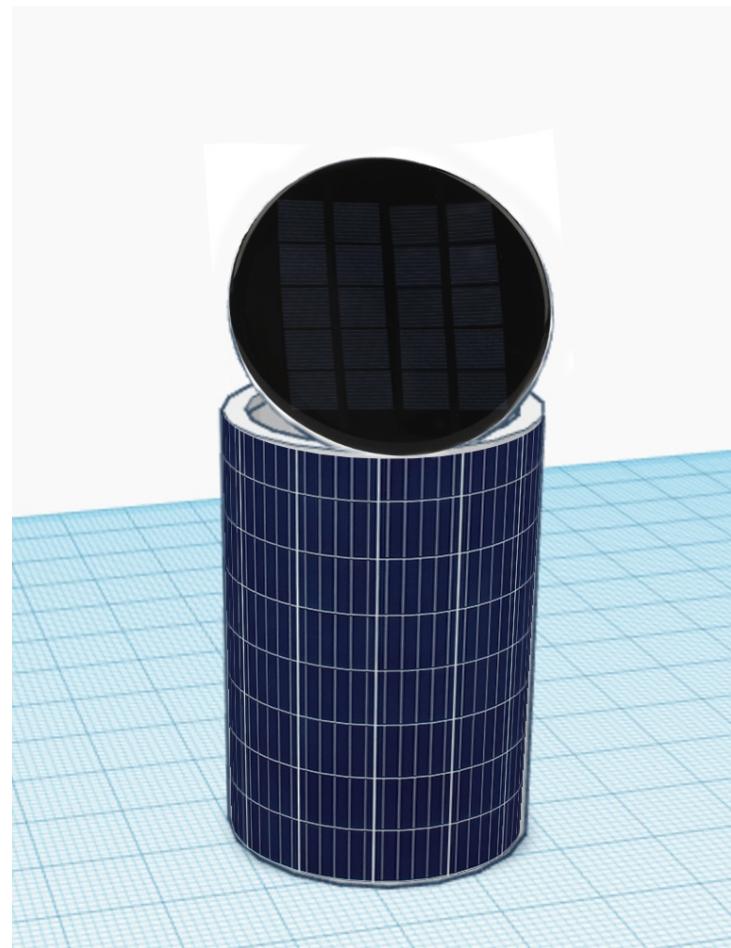


Figure 96: Solar Cell Panel: Hybrid Dual Configuration

6.26.4 Flexible Solar Panel from DFRobot

A flexible solar panel is a suitable option for CanSat due to its adaptable shape. However, finding these panels in the appropriate size for CanSat can be challenging and typically costly. A cost-effective solution could be a flexible solar panel from DFRobot (Figure (97)), which serves as an excellent starting point. This panel can be integrated with the selected SCC to power the 3.7V LiPo battery. The installation process remains identical to that of the 5V solar panel discussed in subsection (6.25.7).

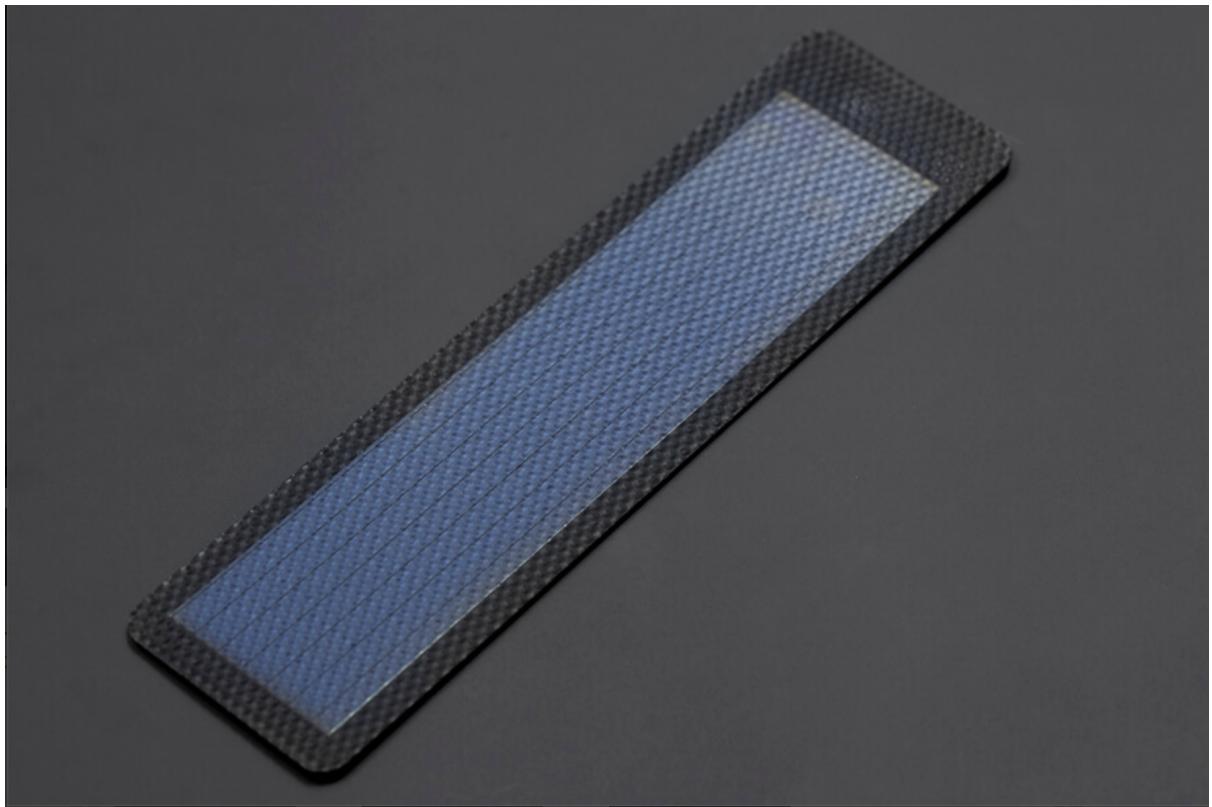


Figure 97: Flexible Solar Panel from DBRobot

RE|GALK

6.27 Antenna Research For C.R.A.F.T-Sat

To design the appropriate antenna release mechanism for C.R.A.F.T-Sat, the selection of the antenna type is crucial. We need first to address the antenna that ensures reliable communication for the mission that does not reach space but operates within Earth's atmosphere, approximately 26 km, above the sea level. The research carried out to identify the most suitable antenna type for C.R.A.F.T-Sat intended to operate at a frequency of 433 MHz and/or 866 Mhz corresponds to a wavelength of 69.28 cm and 34.64 cm, respectively, determined by applying the formula (37):

$$\lambda = \frac{c}{f} \quad (37)$$

where,

- λ is the wavelength of the antenna.
- c is the speed of light in vacuum (approximately $3 \times 10^8 \text{ m/s}$).
- f is the frequency in Hertz.

6.27.1 Antenna Comparison

Our initial research focused on identifying commonly used antennas in CubeSats, especially in 1U CubeSats. The evaluation criteria included simplicity of the design, ease of purchase, feasibility of deployment, gain, bandwidth, and compatibility with a 1U CubeSat frame, and eventually integration of solar panels. The findings are addressed in the following:

Dipole Antenna

- Advantages:
 - Simple to design or to purchase, which facilitates rapid development and testing.
 - High gain, which makes it suitable for C.R.A.F.T-Sat's mission.
 - Wide bandwidth which offers flexibility in communication frequencies.
 - Most widely used in 1U CubeSat due simplicity.
- Disadvantages:
 - The C.R.A.F.T-Sat frame and solar panel integration must be carefully considered as they can affect the radiation pattern.
 - The length of the antenna could be nearly the same as the C.R.A.F.T-Sat's frame, which can affect the antenna performance.
- Suitability for C.R.A.F.T-Sat:
 - Yes, given its performance and size, it is compatible to be used for C.R.A.F.T-Sat.

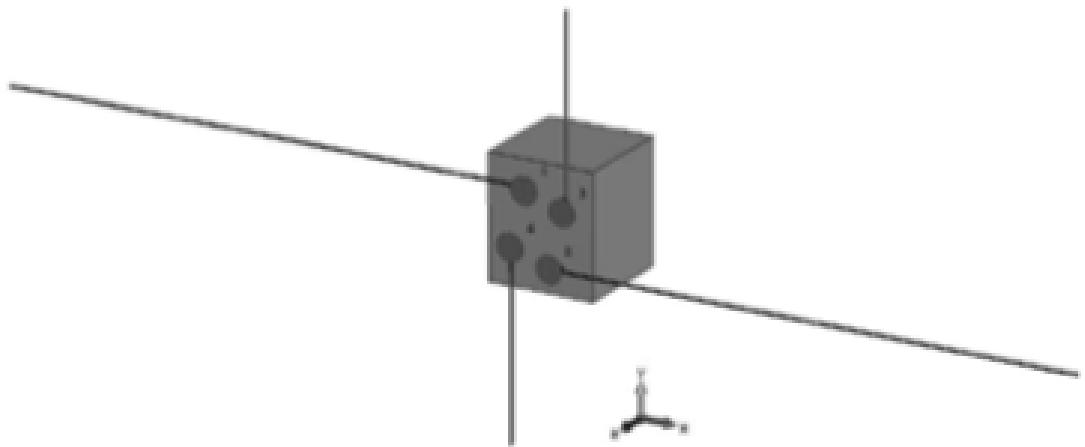


Figure 98: Dipole Antenna[68, p. 108]

Helical Antenna

- Advantages:
 - High gain, which is beneficial for long-distance communication.
 - Wide bandwidth, which offers flexibility in communication frequencies.
- Disadvantages:
 - Due to size constraints, it is not suitable for 1U CubeSat. It is typically used for larger CubeSats than 1U type.
- Suitability for C.R.A.F.T-Sat:
 - No, due to its size factor.

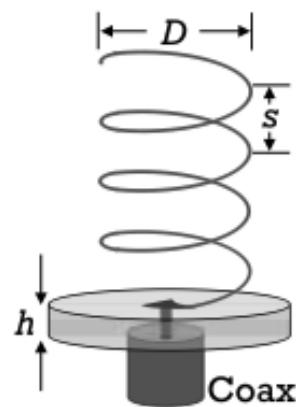


Figure 99: Helical Antenna[68, p. 109]

Quadrifilar Helix Antenna

- Advantages:
 - High gain, which enhances the range and quality of communication.
 - Offers a wide bandwidth, beneficial for versatile communication needs.
- Disadvantages:
 - More complex to design and not as widely available in the market compared to the dipole antenna
 - The antenna must be mounted at the bottom side of C.R.A.F.T-Sat and must point to the receiver in the ground.
- Suitability for C.R.A.F.T-Sat:
 - Yes, particularly when high gain and bandwidth are prioritized.



Figure 100: Quadrifilar Helix Antenna[68, p. 114]

Microstrip Patch Antenna

- Advantages:
 - Easy to design, allowing for straightforward manufacturing and integration.
- Disadvantages:
 - Less suitable for command and control uplink.
 - Deployment complexity, which could introduce additional challenges for designing an effective antenna deployment mechanism.
- Suitability for C.R.A.F.T-Sat:
 - Yes, but it can be too difficult to design a reliable antenna deployment mechanism.

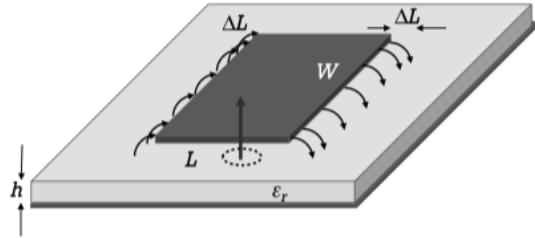


Figure 101: Microstrip Patch Antenna[68, p. 120]

Horn Antenna

- Advantages:
 - Medium gain, which can provide a respectful performance.
- Disadvantages:
 - Large in volume, making it impractical for limited space, as in case of C.R.A.F.T-Sat.
 - Medium gain does not justify the volume and design factor.
- Suitability for C.R.A.F.T-Sat:
 - No, because of its size and impractical for a small size such as C.R.A.F.T-Sat.

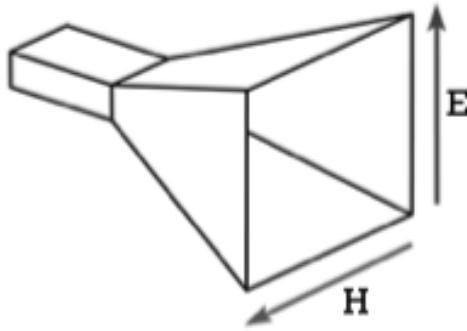


Figure 102: Horn Antenna[68, p. 131]

6.27.2 Reasons for Selecting the Dipole Antenna

Considering all the factors analyzed regarding the antennas mentioned, it can be inferred that the Dipole Antenna and Quadrifilar Helix Antenna are suitable options for use in C.R.A.F.T-Sat and are advantageous for developing the deployment mechanism.

However, the decision to proceed with the dipole antenna for C.R.A.F.T-Sat was made after in-depth discussion in the group and careful consideration of various factors. The underlying reasons for this choice are as follows:

-
- Availability: The dipole antenna is readily to design and readily available on the market, which simplifies the design and/or achievement process and testing.
 - The dipole antenna is less dependent on mounting placement and pointing to the ground station than the Quadrifilar helix antenna.
 - Weight consideration: The dipole antenna is lighter than the quadrifilar helix antenna, which helps to fulfill the weight requirement of 500 g.
 - The performance of both antennas is nearly identical, providing sufficient gain for this type of communication.

6.27.3 The Length of the Dipole Antenna

A dipole antenna can be designed in different lengths, depending on the application. However, the half-wave dipole and the quarter-wave dipole are the most common dipole antennas used in different applications, including a CubeSat.

For a half-wave dipole antenna, the length of the antenna corresponds to half of the wavelength, which depends on the frequency operated for communication, while when using the quarter-wave dipole antenna, the length of the antenna is the quarter of the wavelength.

Frequency of 433 MHz:

We apply the formula (37) to calculate the wavelength, λ

$$\Rightarrow \lambda = \frac{3.10^8 \text{ m/s}}{433 \text{ Hz}}$$

$$\Rightarrow \lambda \approx 692.84 \text{ mm}$$

The length of the antenna, L1, for a half-wave dipole can be calculated using Equation (38):

$$L1 = \frac{\lambda}{2} \tag{38}$$

$$\Rightarrow L1 = \frac{692.84 \text{ mm}}{2}$$

$$\Rightarrow L1 = 346.42 \text{ mm} \tag{39}$$

The length of the antenna, L2, for a dipole of quarter wavelength can be determined using the formula (40):

$$L2 = \frac{\lambda}{4} \tag{40}$$

$$\Rightarrow L2 = \frac{692.84 \text{ mm}}{4}$$

$$\Rightarrow L2 = 173.21 \text{ mm} \quad (41)$$

Frequency of 866 MHz:

In the same way, we can determine the antenna's length when using a frequency of 866 MHz. The wavelength, λ , can be calculated by applying equation (37):

$$\Rightarrow \lambda = \frac{3.10^8 \text{ m/s}}{866 \text{ Hz}}$$

$$\Rightarrow \lambda \approx 346.42 \text{ mm}$$

The length of the antenna, L1, for the half-wave dipole antenna can be determined using Equation (38):

$$\Rightarrow L1 = \frac{346.42 \text{ mm}}{2}$$

$$\Rightarrow L1 = 173.21 \text{ mm} \quad (42)$$

Using the formula (40), we can calculate the length of the antenna, L2, for a dipole of quarter wavelength:

$$\Rightarrow L2 = \frac{346.42 \text{ mm}}{4}$$

$$\Rightarrow L2 = 86.60 \text{ mm} \quad (43)$$

RE|GALK

6.28 Dipole Antenna: Design, Simulation, and Test

6.28.1 Dipole Antenna: Design in MatLab

After selecting the type of antenna that will be mounted in C.R.A.F.T-Sat and in order to understand how a quarter-wave dipole antenna will radiate, it is necessary to investigate the radiation pattern of this particular type of antenna, the main and side lobe where it shows the direction of maximum radiation or less radiation, respectively (Book Antenna Theory and Micro Side 15).

Table (41) presents the parameters used for simulation in MatLab and for design and testing in the EMC laboratory. It is also important to mention that Section (6.27.3) shows details on the derivation of the antenna length for quarter-wave dipole antenna.

Table 41: Parameters used for Simulation, Design and Testing The Quarter-wave Dipole Antenna

Parameter	Value
Frequency of Operation	434 MHz
Wave Length	692.84 mm
Antenna Length	173.21 mm

To design a quarter-wave dipole antenna with parameters illustrated in Table (41) and in the same direction as it will be assembled in C.R.A.F.T-Sat, a MatLab script has been created, as demonstrated in Listing (2).

The script generates a visualization of the quarter-wave dipole antenna, the radiation pattern, the azimuthal plane pattern, and the elevation plane pattern which are illustrated in Figures (103), (104), (105), and (106), respectively.

```

1
2 f = 434e6; % Operating frequency in Hz
3 c = 3e8; % Speed of light in m/s
4 lambda = c / f; % Wavelength in meters
5
6 % Length of quarter-wave dipole antenna
7 L = lambda/4;
8
9 % Create the dipole antenna
10 dipoleAnt = dipole('Length',L, 'Width',L/100, 'Tilt',[90], 'TiltAxis',[0 1 0]);
11
12 % Display the antenna
13 disp(dipoleAnt);
14
15 % Visualize the antenna
16 figure;
17 show(dipoleAnt);
18 title('Quarter-Wave Dipole Antenna for 434 MHz');
19
20 % Plot the radiation pattern of the antenna
21 figure;
22 pattern(dipoleAnt, f);
23 title('Radiation Pattern of the Quarter-Wave Dipole Antenna')
24
25
26 % Azimuthal Plane Pattern (Horizontal)
27 figure();
28 patternAzimuth(dipoleAnt, f)
29 title('Azimuthal Plane Pattern (Horizontal)')
30

```

```
31 % Elevation Plane Pattern (Vertical)
32 figure();
33 patternElevation(dipoleAnt, f)
34 title('Azimuthal Plane Pattern (Horizontal)')
```

Listing 2: MatLab's Code for Designing Quarter-Wave Dipole Antenna and Illustrating its Radiation Pattern.

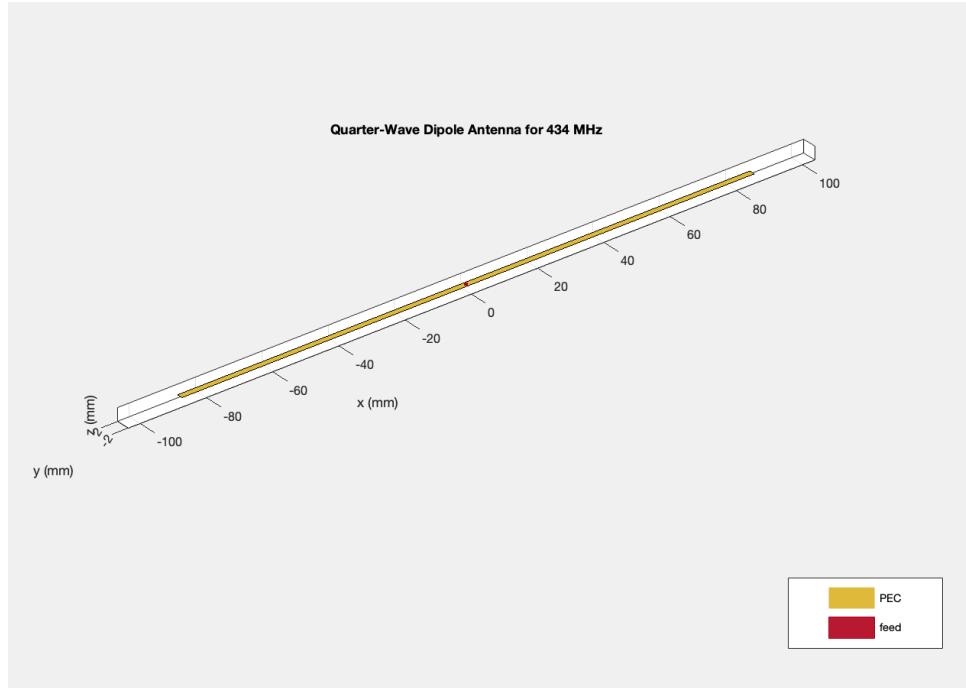


Figure 103: Design of Quarter-Wave Dipole Antenna

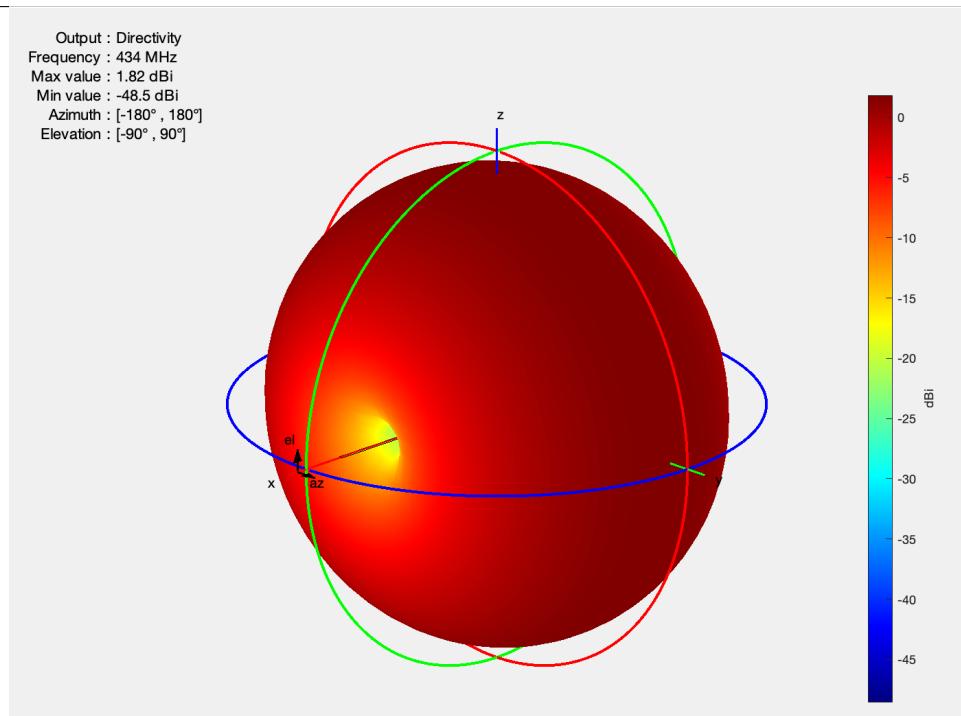


Figure 104: Quarter-Wave Dipole Antenna's Radiation Pattern

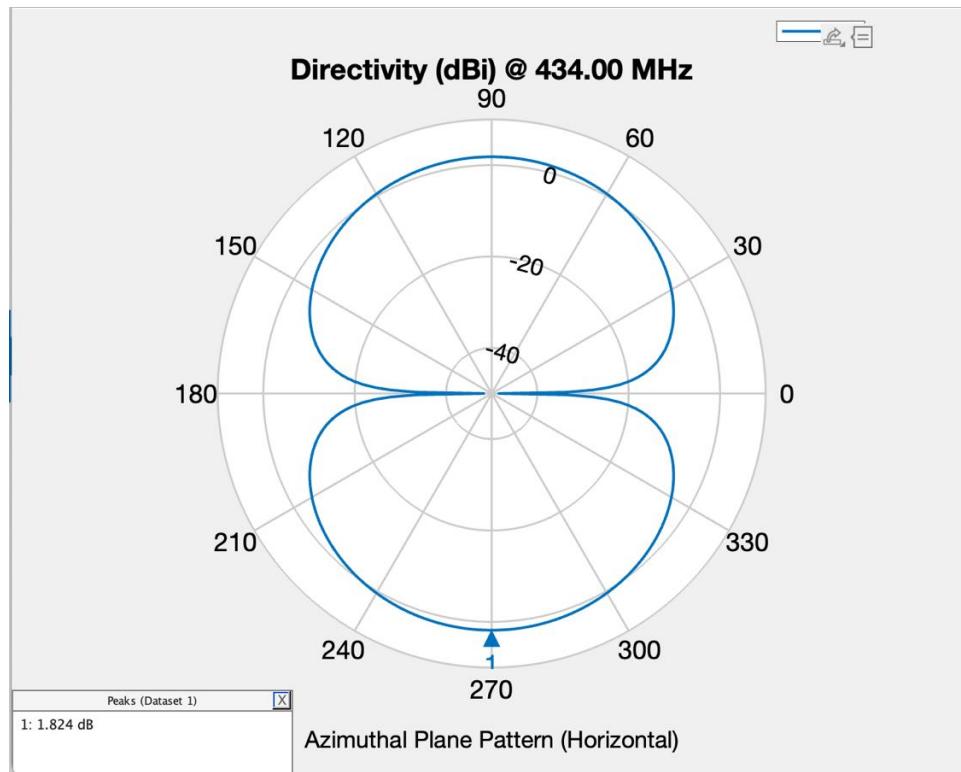


Figure 105: Quarter-Wave Dipole Antenna: Azimuthal Plane Pattern (Horizontal)

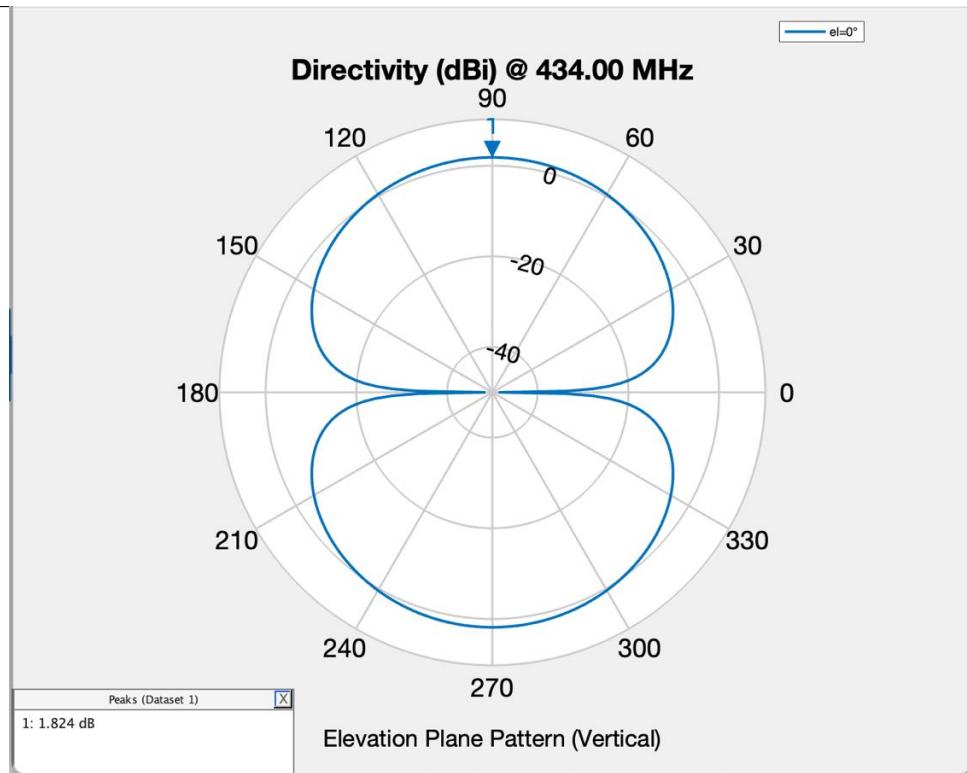


Figure 106: Quarter-Wave Dipole Antenna: Elevation Plane Pattern (Vertical)

6.28.2 Analyzing The Results Obtained in MatLab Simulation

The radiation characteristics of the quarter-wave dipole antenna shown in Figure (104) are characterized by its omnidirectional propagation. This means that the antenna emits waves with relatively consistent intensity in all directions, which is a common trait of dipole antennas. Despite this omnidirectional property, there is a noticeable decrease in the emitted power towards the center.

The azimuthal and elevation plane patterns, as illustrated in Figures (105) and (106), confirm the omnidirectional radiation of the antenna. They show main lobes in both the zenith and nadir, indicating more radiation towards the z- and y-axes in three-dimensional presentation.

An important parameter from the simulation that we need to use in the testing phase is the directivity, measured in **dB_i** (decibels relative to isotropic) to describe the amount of power the antenna can transmit in a specific direction compared to a standard isotropic antenna. As shown in Figure (104), the optimal **directivity** of a dipole antenna is equal to:

$$\text{Directivity} = 1.82 \text{ dB} \quad (44)$$

6.29 Real World Antenna Test and Analysis

To validate the performance of the designed quarter-wave dipole antenna, a thorough testing in EMC laboratory is executed. The designed antenna parameters mirrored these shown in Table (41). The resulting graph is visualized in Figure (??), which depicts a measured signal strength of:

$$\text{Signal Strength} = -10.39 \text{ dBm} \quad (45)$$

where dBm represents decibel milliwatts, a unit used to measure signal strength or power level.

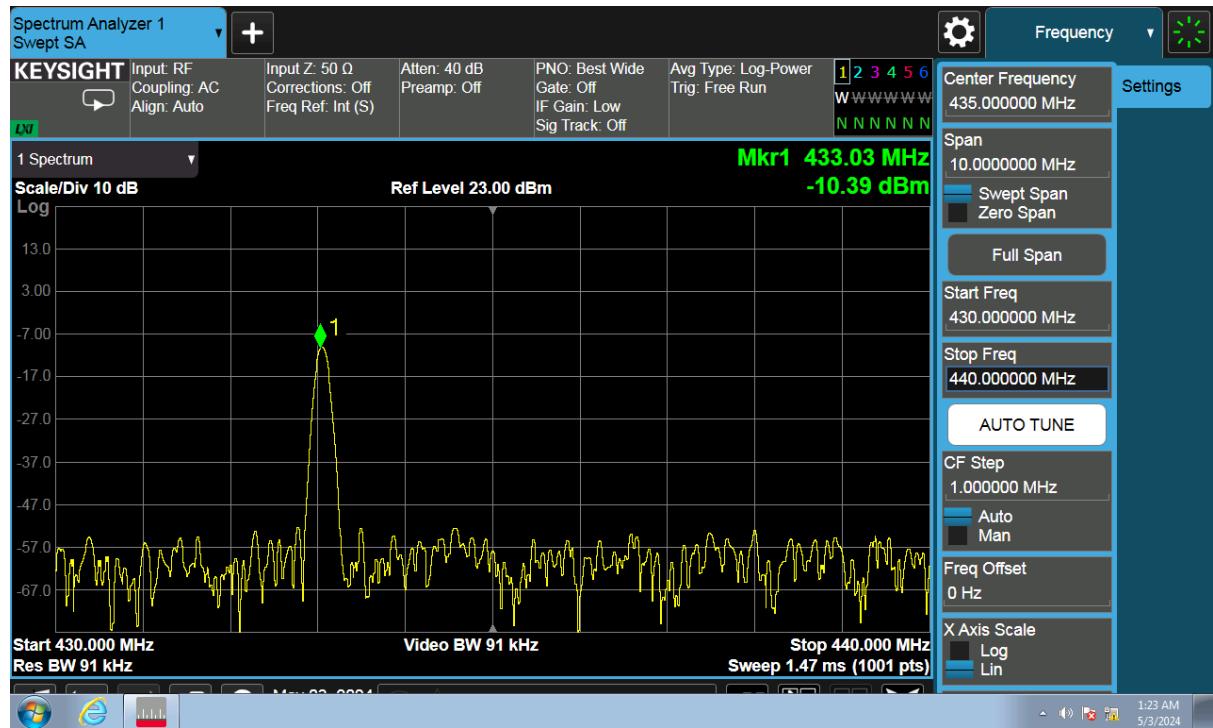


Figure 107: Quarter-Wave Dipole Antenna: Signal Strength in Frequency of Operation (434 MHz)

A crucial point is to determine whether this signal strength is sufficient for reliable communication from around 26 km altitude to a ground station. An online tool has been used to calculate the maximum distance that can be achieved in a communication system. The required parameters, their value, and description are shown in Table(42).

Table 42: Parameters for Calculating the Maximum Achievable Distance in Communication System

Parameter	Value
Transmitter Power	-10.39 dBm
Receiver Sensitivity	-125 dBm
Frequency of operation	434 MHz
Transmit Antenna Gain	1.82 dBi
Receive Antenna Gain	1.82 dBi
Total Cable Losses	1.2 dB

where:

- **Transmitter Power:** refers to the measured signal strength is shown in (45) and in Figure (107).
- **Receiver Sensitivity:** The value is assumed based on the typical value of receiver sensitivity in LoRa communication.
- **Transmit Antenna Gain:** is the same as the directivity obtained in simulation with MatLab and is shown in (44).
- Receive Antenna Gain: is assumed to be the same is the value in transmit antenna gain because of antenna reciprocity theorem. The value is then set to 1.82 dBi
- **Total Cable Losses:** is the power loss due the cable length. The value is measured by using **Antenna Analyzer**

Using the **dBm to Distance Calculator** from **3ROAM** by the given parameters values shown in Table (42), the estimated maximum achievable distance for the communication system is approximately **39 km** as shown in Figure (108).

Transmit Power (dBm)	
-10,39	
Receiver Sensitivity (dBm)	
-125	
Frequency	Units
434	MHz
Transmitting Antenna Gain (GTx)	
1.82	
Receiving Antenna Gain (GRx)	
1.82	
Cable Loss (dB)	
1,2	
Maximum Path Loss (dB)	
113.41	
Distance	Units
39.14	Kilometers

Figure 108: Maximum Distance the Designed Quarter-Wave Dipole Antenna Can Achieve in Communication System.

It is essential to mention that the maximum achievable distance is dependent on the specific type of receiver with the same specific value as shown in Table (42) for receiver sensitivity. The use of another type of receiver with different receiver sensitivity could improve or worsen the final result. This is therefore crucial for ASE to utilize a particular receiver that is typically used in LoRa communication for effective low-signal detection.

6.30 C.R.A.F.T-Sat Library Design

TL|TL

Referring to our 5th interim goal 1.1.1, ASE wants us to develop a "plug-and-play" configuration, as well as a library used to program the C.R.A.F.T-Sat. Due to time constraints in our bachelors project, a decision was made to not develop code for the different components located on the C.R.A.F.T-Sat shield, but instead depend on already developed code as described in subsection 5.8 and 5.9. Adhering to our 3rd stakeholder requirement 3.1, we need the library to be easy to read and use. Therefore we have decided to implement a class to encapsulate all the functions needed by the library. By doing this we can separate the functions within the library, keeping them out of the global namespace, meaning to access the functions the user need to create an object of the library. This makes the code safer and less prone to errors when being used by users with little to no experience.

With the decision to not develop code for the components located on the C.R.A.F.T-Sat shield, it requires some setup before being able to use the library. A guide can be seen in section 174. The first part of the stakeholder requirement states the library should be easy to read 3.1. A design decision was made to not use any abbreviation in the function calls. It is therefore easy to read exactly what the function does i.e. "read_all_sensors()" does exactly what the function states. Acronyms on the other hand can be used since using the entire word may cause more confusion.

To make troubleshooting for the students easier, the "begin" function will display any potential fixes to errors that might occur during the start of the program. This will halt the program if there are problems with the startup of the C.R.A.F.T-Sat, an example of this can be seen in Fig. 109.

```
Starting C.R.A.F.T-Sat
SD Card initialisation failed! Is the SD card inserted?
```

Figure 109: Example error output

6.31 Real-Time Data Transfer

C.R.A.F.T-Sat must be able to transfer data in soft real-time.

Achieving sub-30-second data transfer time is relatively simple and seamless when the amount of sensor data or the distance between sender and receiver is minimal, according to our research. An experiment/study titled "A Real-Time Data Acquisition System for Monitoring Sensor Data" successfully managed to transfer data from **one** sensing unit on a microcontroller to a computer via USB and then to a Python GUI in real-time [69]. Although the meaning of "real-time" was not specified, similar experiments that we did in the subject course "Programming and Microcontrollers" showed data being transferred almost faster than what eyes could perceive.

The problem arises when multiple sensor data are being received simultaneously by a microcontroller and those data have to travel sub 26 kilometers to reach their destination. A paper published by the American Journal of Engineering Research (AJER) showcased an experiment that showed a wireless RF communication between two microcontrollers (Arduino UNO's) using the 433 MHz radio frequency, the same one that C.R.A.F.T-Sat will use [70]. The radio module used was also of comparable specifications and quality to the RFM-96 module that C.R.A.F.T-Sat will use. A major difference, however, is that the distance between the transmitter and receiver was minimal and the study itself claims that an optimistic maximum range would be a range of 300-500 meters. There are, however, slight differences in our system context that could yield significantly better ranges than those achieved in that study.

- 1. Lack of hindrances:** If the distance between the transmitter and receiver is horizontal rather than vertical, there will naturally be more obstacles between those two components [see subsection 3.4.1]. Objects such as trees, vehicles, and buildings can greatly weaken the maximum range. As illustrated in our system context diagram, the distance between the C.R.A.F.T-Sat and the Ground Station will be mostly vertical where there is less impedance.
- 2. Stronger antenna:** Second, the antenna that will be connected to the Ground Station's receiver, will be much more powerful than the "simple wire" that the AJER study used. These two differences help explain why ASE in previous tests has managed to achieve a range of up to 56 kilometers.

Thus, objective 4 seems more plausible thanks to the review of existing research. But data transfer is not just about sending data; it is about sending the **right data**, in the **right format**, at the **right time**. To achieve this, extensive research has been conducted to understand how wireless communication between two Teensys works, and more importantly, map out weaknesses in existing architecture to potentially find ways to improve

reliability and accuracy.

NNG|TL

6.31.1 MVP Solution (Offline mode)

Preface: This diagram represents an MVP solution, i.e., our system in “offline mode”. If the web server is down or network speed is critically low, then this solution should always be a backup.

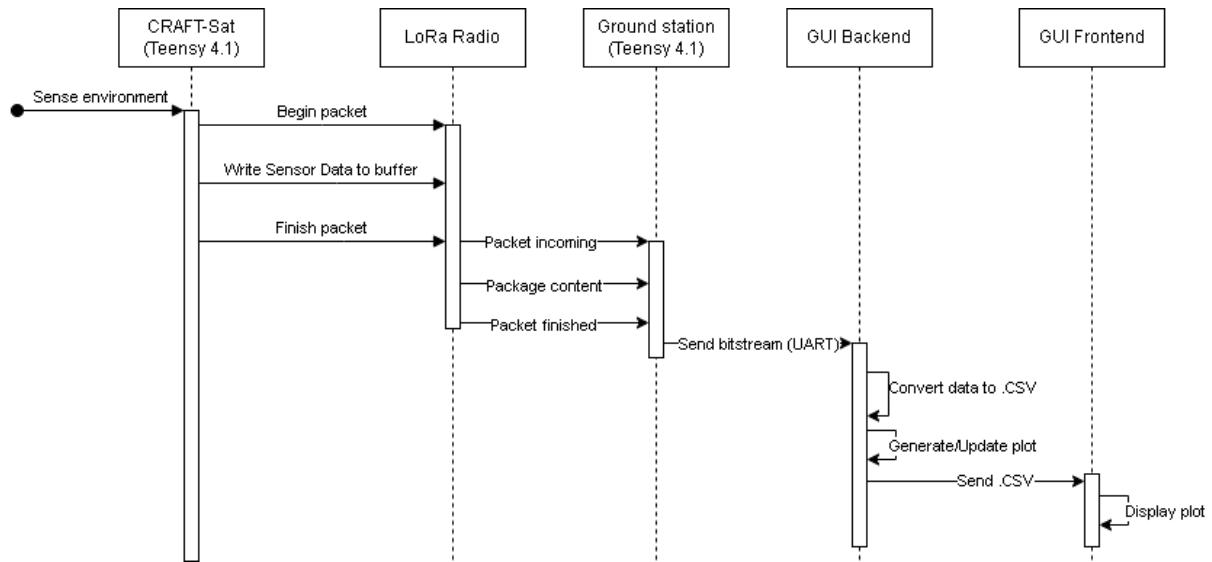


Figure 110: Sequence Diagram - MVP Solution (TL, NN G)

Things to note:

- All the arrows are pointing rightwards. This is due to the current lack of uplink communication functionality. A disadvantage from this is that it seems impossible to establish "handshakes" between subsystems as well as informing the sender whether the package was successfully received. We, thus, have to be creative with alternative solutions in future iterations.
- “Send bitstream (UART)” is easily achieved using built in Arduino function `Serial.Print()`.
- In this solution, bitstream is converted to a .CSV file. The reason for this is that it can easily be imported into Microsoft Excel (in case log is saved locally to machine) and pandas (python) for live plotting. .

TL|NNG

6.31.2 Uplink Solution

Preface: This diagram represents a solution with uplink capabilities. Where the user can send commands to the C.R.A.F.T-Sat.

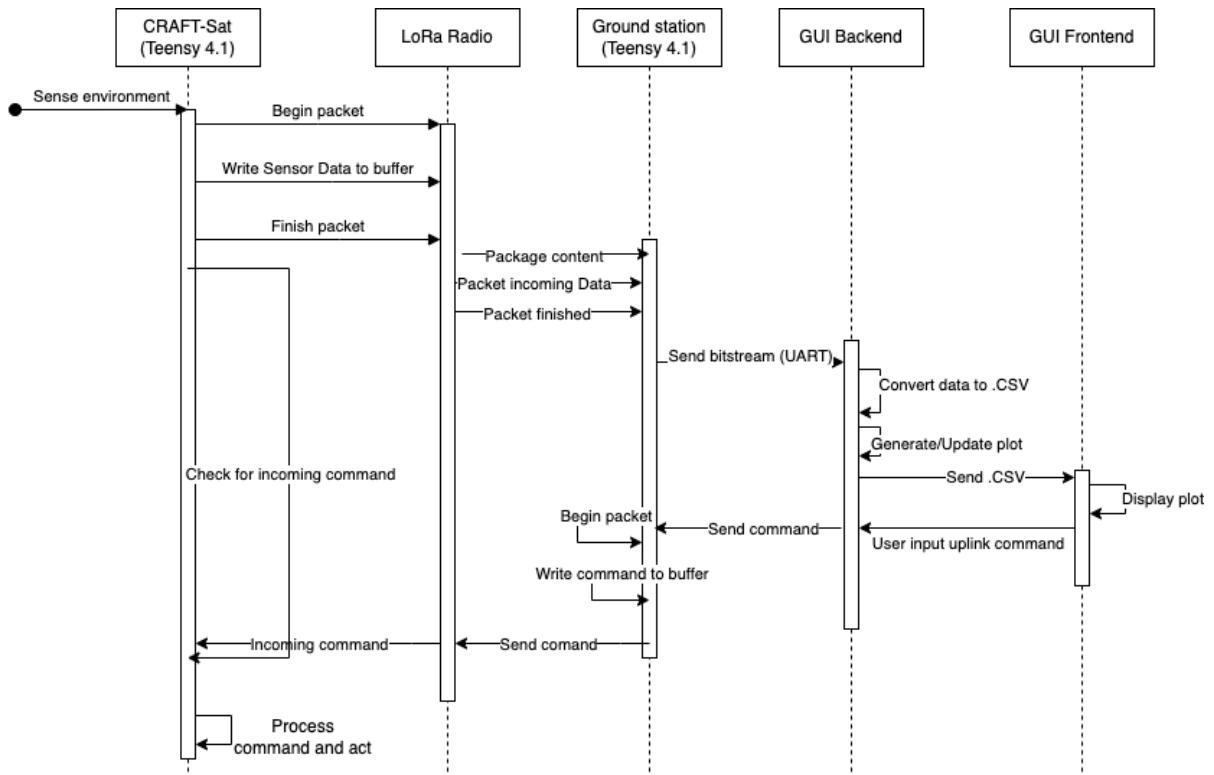


Figure 111: Sequence Diagram - Uplink Solution (TL)

Things to note:

- There is two-way communication with this solution, but there is no "handshake", like in the previous example.
- “Send bitstream (UART)” is easily achieved using the built-in Arduino function `Serial.Print()`.
- In this solution the user can input a command into the GUI frontend. The frontend sends the command to the backend which gives input to the ground station. The ground station writes the command to a packet and sends it to the C.R.A.F.T-Sat. The C.R.A.F.T-Sat will interpret the command and act accordingly.

NNG|TL

6.32 Data architecture

To create a subsystem as complex as C.R.A.F.T-GUI that achieves the functionality and longevity desired by ASE, designing a comprehensive and understandable data architecture was essential. Furthermore, a good data architecture aids in the documentation of our product which is another requirement 3.1. As a final point, a well-documented data architecture makes it easier for ASE to maintain, modify, and expand C.R.A.F.T-GUI if they so desire. The primary requirements that molded our final design were:

- Requirements such as :

-
- Easy to use
 - Longevity: Last for 10 years
 - Free and open source
 - System context

6.32.1 Easy to use

For the majority of this report, "ease of use" has referred to the employment of our system. However, usability should also be viewed from the perspective of the system's **caretakers**. It is crucial to show consideration to this stakeholder as they play a significant role in the longevity and safety of our end product. This aspect is especially important for us since once deployed, we will not maintain C.R.A.F.T-GUI for our client; they will require another custodian for that.

6.32.2 Longevity: Last for 10 years

Few things are as constantly evolving as software technology. New software products and tools emerge while others are phased out or get discontinued support and updates. For reasons related to security and stability, longevity was one of, if not the biggest incentives when designing C.R.A.F.T-GUI's data architecture and selecting what tools and products to be included inside it. Although it is mostly impossible to predict the existence of a product in 10 years, assumptions based on thorough research, group discussion, and referrals with ASE were made to the best of our abilities.

6.32.3 Free and open source

3D printing and electronic assembly are activities that can be costly, both concerning raw materials and tools required for production. For instance, CanSat competition requirements set an upper budget limit of 500 Euros. C.R.A.F.T-Sat will naturally be more expensive due to the additional components that it has. This could compound to a significant cost for ASE as the number of enrolled student groups increases. Thus, it is desirable to reduce costs, especially when doing so does not negatively impact system performance. Software programming is an area where this is possible. There is an adequate amount of free GUI toolkits and Python frameworks that can be used to create C.R.A.F.T-GUI according to our needs and wants.

Free, however, is not sufficient by itself. Licensing rights are something often neglected, but can later be detrimental to the distribution of a software system. Some tools are free to download and use, but later require part ownership of the end product, or impose mandatory fees for publishing and distribution. This is not desirable, neither by us nor ASE. Therefore, part of the design requirement 3.3 was that all software tools used must be "open-source". In the context of software development, open source is defined as "software with source code that anyone can inspect, modify, and enhance" [71]. (Make sure to compare and contrast popular open-source licenses). However, numerous types of open source licenses can vary greatly concerning legal use. Some licenses require that

changes made to the source code must be documented and published. Other licenses demand that the source code is also included. To reduce the risk of inappropriate upkeep with legalities, we steered from licenses that held such demands whenever possible. That left us with three licenses that qualified: MIT, Boost Software License 1.0, and Unilicense [72]. All of these licenses are similar, but Unlicensed does not require a copy of the license, nor a copyright notice to be included in the distributed product [72]. Demanding free and open source source for all aspects of C.R.A.F.T-GUI ensures ASE complete, long-term ownership of our product.

6.32.4 System context

The following aspects from our system context [Subsection 1.2.1] were especially significant when evaluating designs for C.R.A.F.T-GUI's data architecture:

Age group: Young people tend to explore a system to its absolute limits, sometimes to their detriment as weaknesses in the system's design are explored.

Security: Security is desired, but the sensor data transmitted and collected is not classified information. However, we were informed by ASE that their web server that C.R.A.F.T-GUI could utilize does not require encryption. This information influenced our data architecture regarding the storage of user information. Further explanation regarding the latter can be found in Subsection 6.33.

Location: Since C.R.A.F.T's ground station will be used indoors [Subsection 1.2.1], it is assumed that ASE's internet strength is strong and stable. However, small dips in signal quality is a risk, especially if there are multiple student groups present simultaneously,

NNG|TL

6.32.5 Using an External Server

As mentioned in sub-subsection 6.32.4, ASE has granted permission to utilize one of their external servers for C.R.A.F.T-GUI, but we were not required to use it. Not involving an external server and keeping C.R.A.F.T-GUI strictly is by far the least complex solution and it would satisfy an MVP solution for Objective 6 [Subsection 1.1.1]. The reduced complexity would also improve ease of use and maintenance. However, a strictly offline application is the least feature-dense solution by far. No online connectivity means no remote database which makes our ASE-approved idea (add reference) of a configuration page less streamlined [See sub-subsections 6.36]. It would also require more disk space on the end users computer to store satellite logs. Considering that features are important to our target demographic [See subsection 5.11], we decided to utilize an external server for C.R.A.F.T-GUI. However, the extent to which said server is employed had to also be decided.

Deploying our entire application onto ASE's server has several benefits: For one, users would only need a URL to access C.R.A.F.T-GUI from anywhere, via a web browser. There is no secondary storage needed at all, only primary storage (e.g., RAM). A pure web application is also agnostic to the hardware and operating system of the computer trying to access it; administrators could in theory administer C.R.A.F.T-GUI from any computer or even a tablet. Users would also be spared from manually installing and updating the application; these changes are done on the backend, abstracted from the user. The major fallacy point that deviated us from this concept was that entire application would be down if the server is inoperable, temporarily destroying all customer value. Another risk with such a design is linked to the safety. When utilizing a external server, measures have to be taken in order to prevent unsolicited and undesirable activity on there such as malware and eavesdropping. The risks for such can be reduced via technologies such as demilitarized zones (DMZ), firewall policies and intrusion detection and protection systems (IDS/IPS). That is, however, at expense of high costs for ASE. Considering that that our system is intended for low budget, educational projects, these investments seem disproportionate to our system scope and context. We were unable to gather enough information from ASE regarding their existing measures revolving cyber security, which further steered us away from this concept.

6.32.6 Using Databases

For the storing and retrieval of data, there are two possible solutions: storing the data locally, or on a database that is running on an external server. The latter solution was selected by us. For one, storing data remotely is safer than local storage as the data is protected regardless of whether the end user's ground station malfunctions. Second, storing data remotely frees up storage on the ground stations computer, which is of benefit to the end user. Lastly, using a remote database means that multiple persons (with the right privileges, see Subsection 6.33.1] can access and store data on there. A **milestone decision** we made was to use *both* a relational *and* non-relational database for C.R.A.F.T-GUI.

Relational databases (commonly referred to as SQL databases) are data sets that are structured according to standardized normalization rules. In the context of Database Management Systems (Database Management Systems (DBMS)), database normalization refers to "the process of organizing data in a database" [73]. As of today, there are five different normalization forms (Normalization Form (NF)), where 5NF is the most normalized. However, a database that achieves 3NF is considered normalized [73]. For further information about the first three normalization forms and how they were achieved for C.R.A.F.T-GUI, see Subsection 6.35.

The main **advantage** of relational databases is that its rigid structure makes it efficient at organizing and querying stored data. This is relevant for C.R.A.F.T-GUI as a short response time is valuable due how important system performance is for our end users [See subsection 5.11]. Thus, if users are forced to wait for significant periods of time for data to be handled, their incentive to use C.R.A.F.T-GUI is diminished. Another ad-

vantage of a rigid and standardized design pattern is that it enforces consistency, which creates **predictability** for the end user and the custodians of C.R.A.F.T-GUI. This is particularly useful for data sets that prefer a consistent arrangement. In our final design, we have thus used a MySQL database to for "inventory" (i.e., users, sensors, satellites etc. Complete list of data can be found in Figure 119). Relational databases are also easier for long term documentation since its structure is less likely to be altered over time. This ensures that the maintainers of C.R.A.F.T-GUI can better trust the provided documentation and require minimal alterations from them to said documentation. Thus, choosing an SQL database would fulfill our stakeholder requirement regarding "thorough documentation" [Subsection 2.2].

Non-relational databases (also referred to as NoSQL databases) are data sets that does **not** enforce any fixed structure by default [74]. This type of database is particularly useful when there is no natural way to normalize the data of interest. A relevant example is the sensor data that is being received by the ground station. There is no intuitive way to normalize such data. To add, the transmission of data from satellite to ground station can be unpredictable due to student group making alterations to our satellite design and our Teensy 4.1 library. This means that even if the database was structured to store sensor data, there is a considerable risk that the data forwarded by the ground station would not fit inside. Since a "faulty" log from a satellite launch should not damage the entire infrastructure, a non-relational database has been dedicated to store mission data. By doing so, C.R.A.F.T-GUI is functional regardless of the quality of logs.

NNG|NNG

6.32.7 Final Design: Flask and databases on Remote server

Based on the aforementioned criterion and our preparatory work, C.R.A.F.T-GUI's data architecture is modelled in Figures 112, 113, and 114.

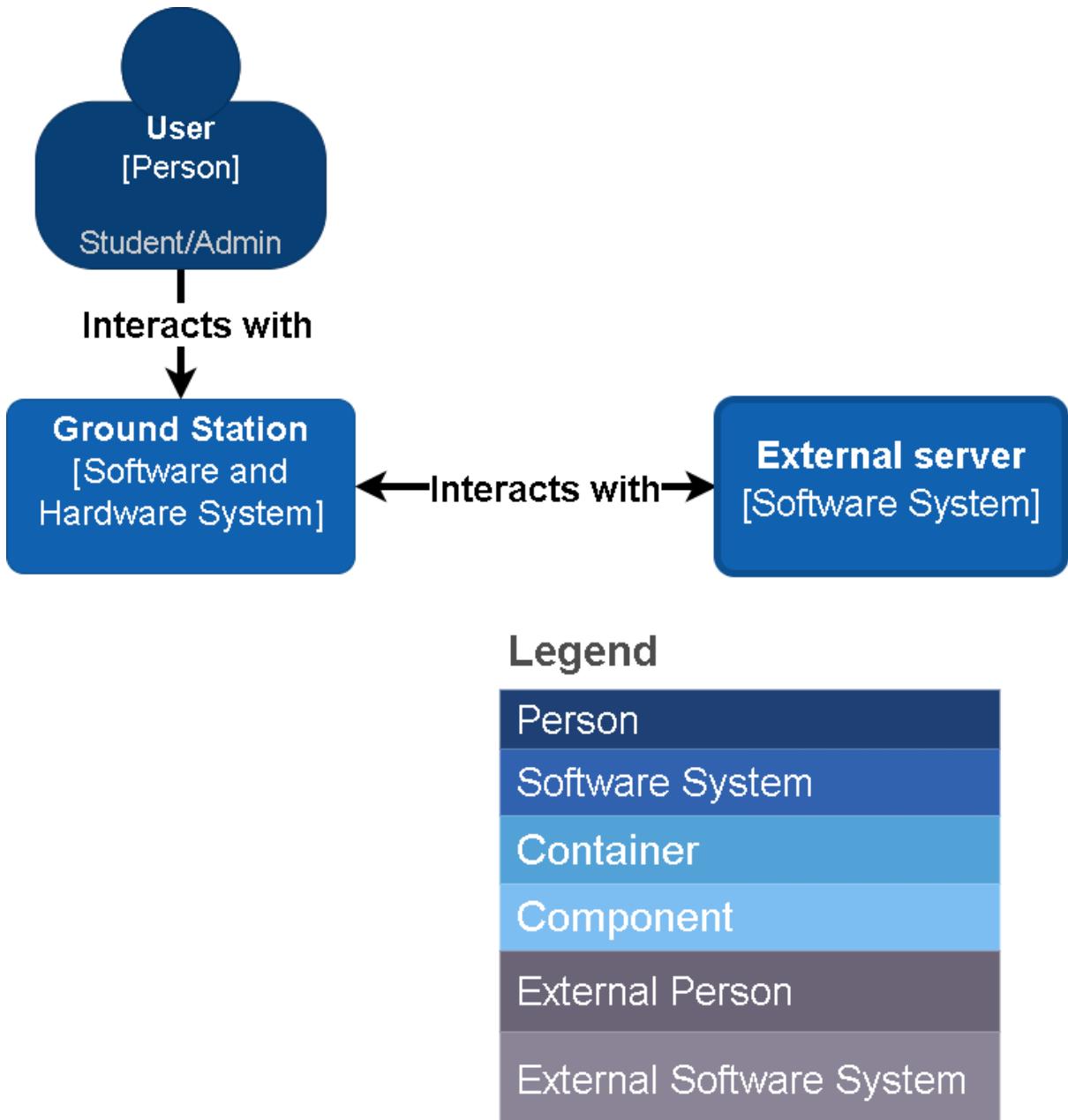


Figure 112: C4 system context diagram (C.R.A.F.T-GUI)

As Figure 112 shows, the user interacts with the ground station, and the the ground station communicates with ASE's external server. Observing Figure 113, the ground station includes the GUI itself and the local Python backend. Locally running these sub systems/containers preserves independence from an internet connection. For functionality that requires access to our remote databases (e.g. user authentication), the local backend can forward user requests to the external server where the database is located. From the perspective of the end user, this positional change would make C.R.A.F.T-GUI resemble more of a applications similar to Microsoft OneDrive: data can be stored remotely once logged in.

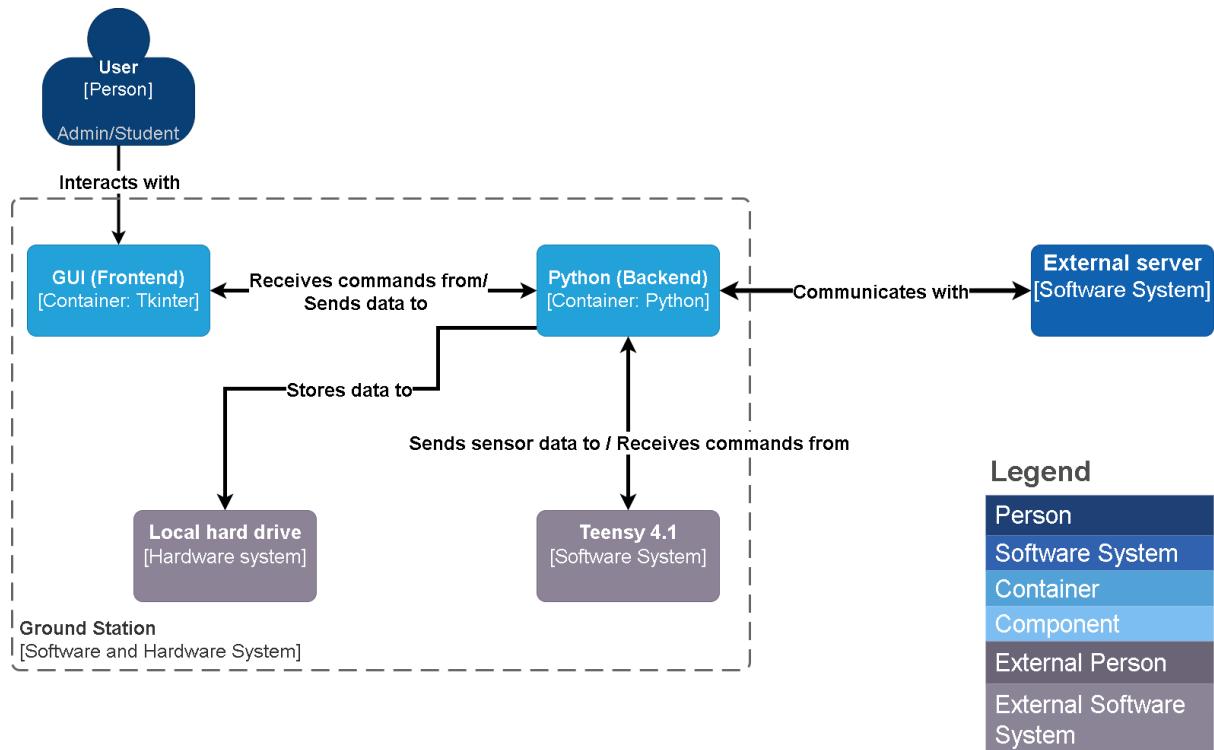
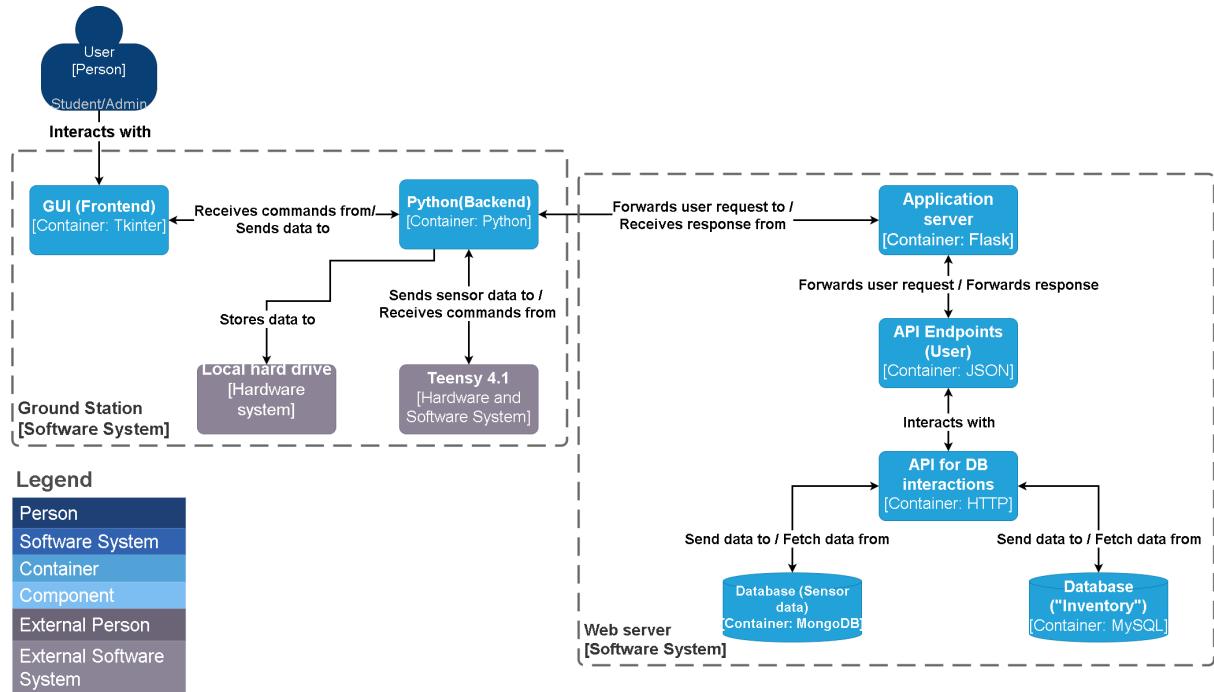


Figure 113: C4 container diagram (Ground station)

The container "GUI (Frontend)" refers to the interface that users interact with. "Python (Backend)" encompasses code abstracted from the user (e.g., UI layouts) along with code that handles data to/from our Teensy 4.1. "Teensy 4.1" is labeled a hardware **and** software system as it also includes its programmed library.

Figure 114: C4 container diagram (Web server)



An intermediary component is necessary to communicate with the remote databases, namely an **application server**. This server type is responsible for enabling "interaction between user clients and server-side application code [75]". For C.R.A.F.T-GUI, Flask has been our application server of choice since it is well supported for Python, is free and open source.

Flask has been running locally on our personal computer during the development process of C.R.A.F.T-GUI, for simplicity's sake. Although running Flask locally on each end user's computer is technically possible, it is not recommended as each user needs to have the right version of Flask installed. This would reduce our product's ease of use. Therefore, it is better practice to transfer Flask to an external server. Another major advantage of running Flask on an external instance is **improved traceability**. Since Flask is centralized to one location instead of an undefined amount, it is much simpler for the administrators to monitor traffic and attempts at contacting the databases. This would in practice add another layer of safety to our GUI as data breaches can be faster identified and handled. The same can also be similarly said regarding system malfunctions as time spent on troubleshooting by ASE's IT department can be significantly reduced. If the centralized Flask application server is undergoing downtime (both planned and unplanned by ASE), then all the users are blocked from data backup and other additional functionalities. However, one could argue that this is actually an advantage as it is often not desired by application maintainers that an uncontrollable amount of users can still access the database when it is perhaps undergoing changes during downtime. With this design, there is nothing that prevents the Flask server and the databases to be running on separate external servers; the setup should be exactly the same. This provides ASE

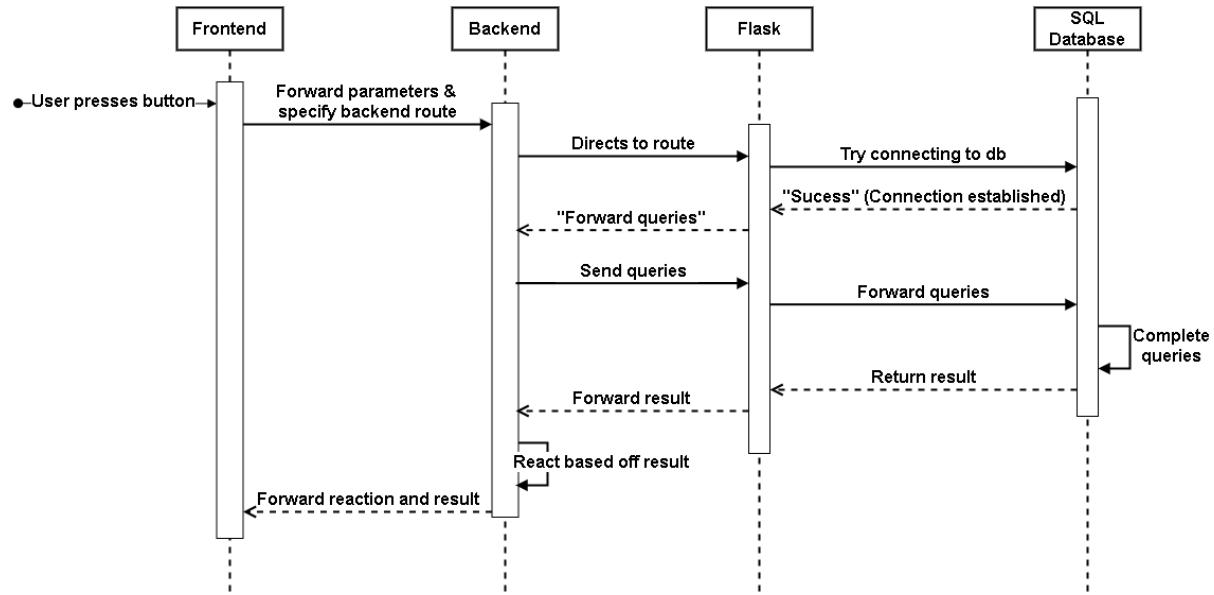
the freedom to decide how they want to setup these two subsystems.

Analysing Figure 114, Flask accepts user input and redirects to relevant API endpoints. API (Application Programming Interface) endpoints refer to available destination points for user requests to be routed to. The Listing below shows an example of API endpoints.

```
1 @app.route("/login_otp", methods=["POST"])
2 def login_otp():
3     data = request.json
4     email = data["email"]
5     otp_candidate = data["otp"]
6
7     result = logintodb_otp(email, otp_candidate)
8     return jsonify(result)
9
10 def logintodb_otp(email, otp_candidate): # parametrizes login credentials for input sanitization
11     cursor = None
12     db = None # Initialize db outside the try-except block
13     try: # where in the function should database connection occur? Write about it in documentation
14         db = mysql.connector.connect(
15             host="craftsql.mysql.database.azure.com",
16             user="nathanael_getaneh",
17             password="Craft2024",
18             database="craftsql",
19         )
20         cursor = db.cursor()
21
22         # Sending SQL query to remote database
23         query = "SELECT user_type, secret_key FROM users WHERE email = %s"
24         cursor.execute(query, (email,)) # expects tuple, comma is thus needed.
25         user = cursor.fetchone()
```

The first line "@app.route()" corresponds to the "API Endpoints (User)" container in Figure 114 where user requests are accepted. C.R.A.F.T-GUI administrators have a few more available API endpoints than students since they have access to more pages and functionality. This endpoint forwards user request by calling the function "logintodb_otp()" corresponds to the container "API for DB interactions". This endpoint sends a HTTP (Hypertext Transfer Protocol) request to our SQL database along with a query. The result of the query is then returned back to "API for DB Interaction" and from there to "API Endpoints (User)". From this endpoint, the query result is formatted as a JSON (Javascript Object Notation) file and forwards back linearly (through each container) back to "GUI (Frontend)". The process of handling user requests is also visualized as a sequence diagram in Figure 115.

Figure 115: Abstracted sequence diagram showing handling of a user request in C.R.A.F.T-GUI



NOTE: The diagram in figure 115 is abstracted. A concrete version of such a diagram is available in Figure 117

6.32.8 Deploying C.R.A.F.T-GUI - Pyinstaller

Packaging a Python application for reuse is surprisingly difficult compared to other programming languages.

"One of Python's biggest shortcomings is that it's hard, often unreasonably so, to take a Python program you've written and just give it to someone else. Languages like C and C++, and now Go and Rust, have build tools that generate a standalone binary artifact. Python, an interpreted language, requires its runtime in order to do anything. But bundling the runtime with your code is at best a clumsy process." [76]

Pyinstaller is a popular open source tool (GNUPV3) that bundles the python application as an executable (.exe) file [77]. Our end users can start C.R.A.F.T-GUI just as they would any other desktop application; by clicking on the file. This drastically reduces the complexity for our target audience. The users also does not have to worry about having the right python interpreter installed; they do not even need to have python installed on the computer at all [77]. The students and administrators do not have to be concerned about installing the right dependencies to use the GUI; those are also included in the bundle. It also bundles the application files (ease of use) Another benefit is that Pyinstaller does not set any limitation on network connectivity to external components. Thus, it is important to emphasize that the inclusion of Pyinstaller does not alter any of the previously discussed data architectural designs and that all of these solutions are able to

"include" Pyinstaller (although it would make little sense to use Pyinstaller with solution no. 1, as it clicking on the executable would just direct to a web browser). Pyinstaller does not really affect the architecture diagram, but is the gateway to said architecture.

Disadvantages: Pyinstaller is OS dependent. A bundle for Windows and MacOS/Linux would have to be configured separately [78]. Fortunately, ASE has confirmed that the students will only use Windows 11 on their sites so that is less of a concern. However, this could be a problem if students outside ASE (e.g., working at their own school) wants to use the application with a non windows computer. Pyinstaller does not support ChromeOS (ChromeOS does not support .exe files in general) which is unfortunate as Chromebooks are popular to students due to their low prices. There has been reported a risk of the .exe file created by Pyinstaller to trigger a false alarm to certain antivirus software, although those risks can be mitigated by tinkering the settings in Pyinstaller [76] (i.e., custom compiling the .exe -briefly describe what that means)

NNG|TL

6.33 Safety Implementations for C.R.A.F.T-GUI

Introduced by the European Union (EU) in 2016, the General Data Protection Regulation (GDPR) is an extensive privacy and security law for the handling of private data [79]. As an European Economic Area (EEA) member, Norway's privacy laws consist of GDPR [80]. Thus, C.R.A.F.T-GUI must comply to this legal text, especially since it stores user data [Sub-subsection 6.32.6].

C.R.A.F.T-GUI does not store any sensitive data about its users (e.g., health history, race, sex etc.). However, it does store **personal data**. GDPR defines personal data as "any information which are related to an identified or identifiable natural person." [81]. Although the EU does not provide a complete list of what classifies as personal data, the definition's inclusion of "**'any information'**", one must assume that the term 'personal data' should be as **broadly interpreted as possible**." [81]. Storing personal data is not forbidden by GDPR, but it "places the responsibility on the controller and the processor in Art. 32(1) of the General Data Protection Regulation to implement appropriate technical and organizational measures to secure personal data" [82]. Once deployed, ASE is considered the "**controller and processor**" of C.R.A.F.T-GUI. A list of mandatory encryption techniques is (again) not included by GDPR, to "accommodate individual factors" [82].

For these reasons, the following decisions were made regarding what sorts of personal data C.R.A.F.T-GUI will store, and how said data should be protected:

- **Username instead of email.** It is generally easier to identify a person via their email than their username, since they are used in multiple areas of their lives (Social media, work, etc.). Without confirmation emails, a person's email can be used without their knowing, which is arguably unethical. Verifying that an email address is valid (i.e., sending a confirmation email) would require an extra server inside C.R.A.F.T-GUI's data architecture [Figure 114]. Complying with GDPR,

this server should also have security measures set in place, disproportionately increasing system complexity and vulnerability. The same reasoning is also why phone numbers are not stored in C.R.A.F.T-GUI.

- **Keep storing user's forename and surname.** Although categorized as personal data, full names are not as unique as phone number nor email as multiple persons can share the same full name. We also considered full names to be of importance for C.R.A.F.T-GUI administrators as full names are less cryptic than usernames. Thus, full names help administrators separate users apart so as not to delete the wrong user account by accident. However, if ASE decides to remove forename and surname from their Structured Query Language (SQL) database, they are able to do so without causing damage to its data architecture. This is because these attributes are not foreign keys inside any of said database's concepts.

6.33.1 User Authentication - TOTP

To protect the personal data stores by C.R.A.F.T-GUI Time-based One-Time Passwords (Time-based One-Time Passwords (TOTP)) has been integrated. TOTP are one-time passwords based on a secret key in combination with the time of a system clock [83]. Secret keys are unique to each user, and has to be stored in at least two places; the user's phone and a database. Upon account registration, the secret key can be packaged as a quick-response (QR) (quick-response) code which the user can then scan on a mobile authenticating app (such as Google Authenticator) [83]. From a computer algorithm involving the secret key and the phone's clock, a one-time password is generated. The same process occurs on the external server where the same secret key is stored, using the clock of said server. The TOTPs from the server and the user's phone are compared to each other. If equal, the user is successfully authenticated. The entire process is visualized in Figure 116.

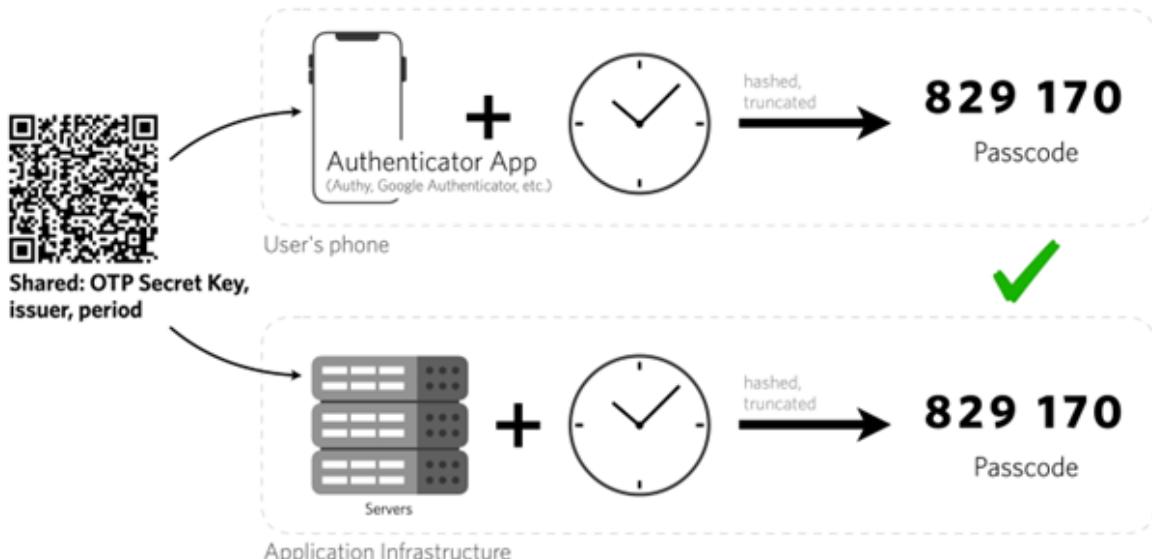
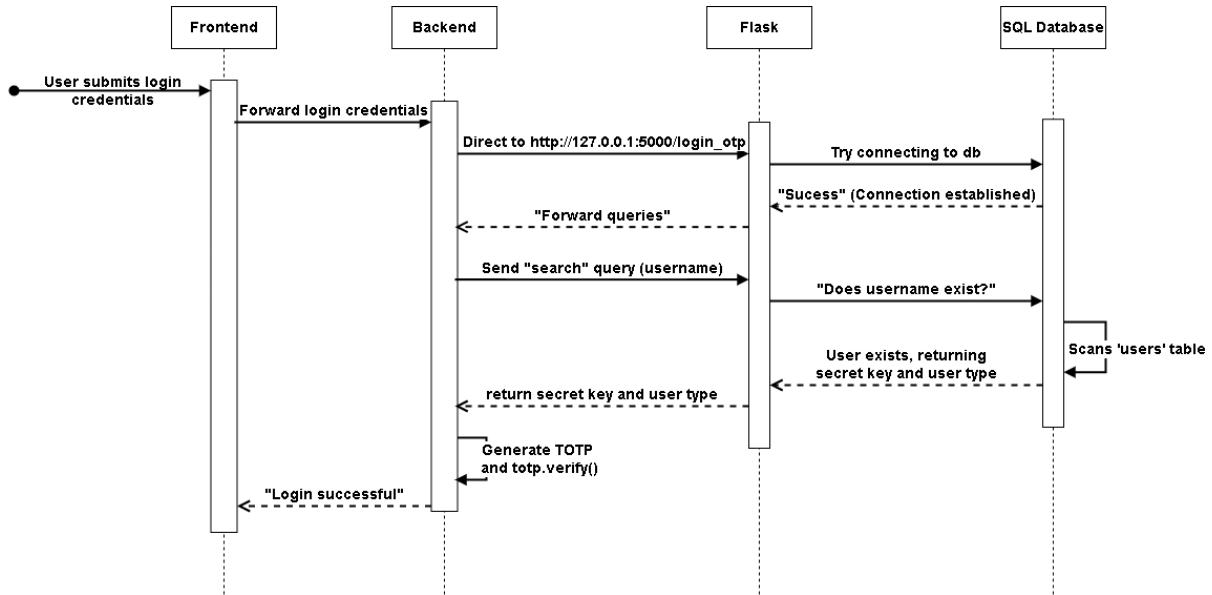


Figure 116: TOTP process [83]

The generated TOTPs are only active for a short time period (30 seconds for C.R.A.F.T-GUI). Once that time window is closed, the algorithm repeats itself and a new code is generated. **PyOTP** is the Python library used to configure Time-based One-Time Passwords. The implementation of TOTP for C.R.A.F.T-GUI is visually described in a sequence diagram, found in Figure 117

Figure 117: Sequence diagram: TOTPs in C.R.A.F.T-GUI



NOTE: `totp.verify()` is a function built into PyOTP which verifies that the user-inputted password is correct.

The main advantages of equipping TOTPs instead of passwords/passphrases is that the secret keys are harder to assume, since they are randomly generated. Also since the user will only have to type in the TOTP (which is only temporarily valid), any bystanders or eavesdroppers are unable to view secret key, only a code that will be useless within a few seconds.

One risk regarding TOTPs is the application server (i.e., Flask) operating at a different clock time than that of the user's phone. For instance, if the user is at a different time zone than the application server wants to login. Fortunately, Norway is not split into multiple time zones. Although it would be nice to consider edge cases where the user is abroad, that scenario is considered to be quite a rarity. To account for small misalignments in clock readings, the time window for TOTPs can be adjusted if need be. Another downside is that if the user lose access to authenticating app (e.g., Forgot their phone), then they are unable to log into C.R.A.F.T-GUI. To manage these risks, an alternative authentication method has also been integrated, namely encrypted passwords [Sub-subsection 6.33.2]

6.33.2 User Authentication - Password Hashing

As an alternative, C.R.A.F.T-GUI users are able authenticate themselves using a password that is permanently stored inside the SQL database. However, if an intruder manages to access the database, or if the password was fetched in-transfer, then said password could be read by others in plain text, which is not respectful to GDPR legislation. To rectify this, **encryption** of backup passwords has been configured. Although synchronous ciphering (i.e., allows encrypted data to be deciphered) is sufficient, we decided to use **hashing** as our encryption method instead. Once a password has been hashed, the process is irreversible, adding another layer of safety. Rather, the user inputs a candidate password which is hashed and then compared with the hashed password stored in the database [84]. Flask-Bcrypt (a Flask extension) is our selected tool for password hashing [84].

NNG|TL

6.34 Setting up SQL database - Database Modelling

Before synthesising a relational database, it had to first be modelled and normalized. There are four laws in database modelling that must be followed in order for said database to be considered architecturally sound.

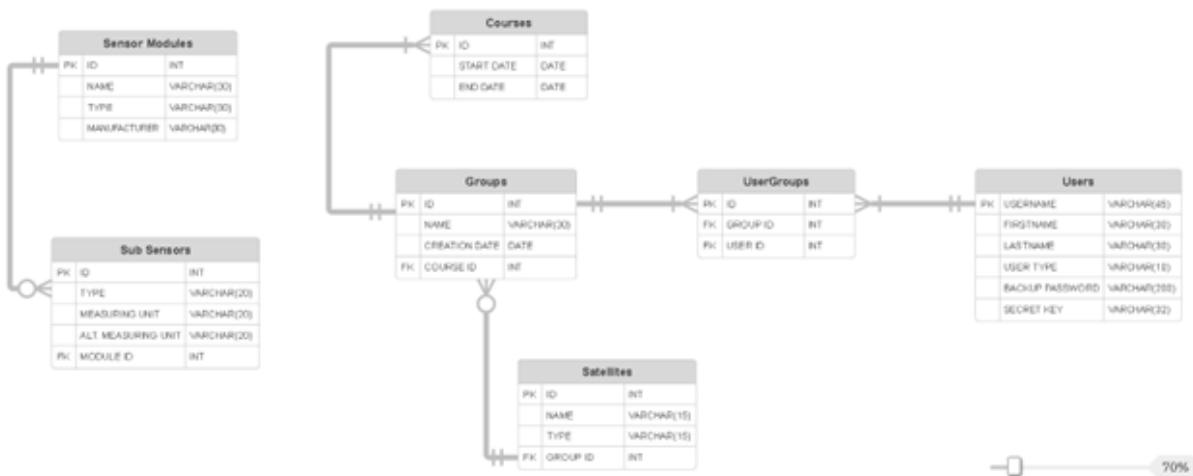
Rule 1: Attributes are grouped within belonging concepts [85]. In the context of C.R.A.F.T-GUI's SQL database, *concepts* refers to the tables that constitute a database, while attributes refers to the columns in these tables. To clarify this rule, an excerpt from our Entity-Relationship (ER) diagram representing the table "users" has been visualized in Figure 118.

As seen from Figure 118, it is apparent that all these attributes are relevant for a user. Therefore, it is appropriate that they are "defined" inside the "users" table. The column "user type" was deemed a necessary addition to separate system administrators from regular users as their privileges are different. All the concepts in C.R.A.F.T-GUI's SQL database are represented in the Entity-Relationship (ER) diagram in Figure 119.

Figure 118: DBMS modelling rule no. 1: Table "users" (ER diagram)

Users		
PK	USERNAME	VARCHAR(45)
	FIRSTNAME	VARCHAR(30)
	LASTNAME	VARCHAR(30)
	USER TYPE	VARCHAR(10)
	BACKUP PASSWORD	VARCHAR(200)
	SECRET KEY	VARCHAR(32)

Figure 119: ER Diagram of C.R.A.F.T-GUI's SQL Database



Rule 2: One-to-many relationships between two entities are grouped on the "many side" [85]. To explain this rule, two terms must be properly defined: **primary**

keys (**Primary key (PK)**), and foreign keys (**Foreign Key (FK)**)).

A primary key is an **unique** identifier for each row inside a concept. For instance, a social security number uniquely identifies a specific individual. It is a value that can never be duplicated nor instantiated more than once. Foreign keys are attributes within a table that are primary keys inside another table. FK's are useful for establishing a link between multiple tables within a database.

In the context of database modelling, a one-to-many relationship (One-to-many (1:m)) means that one entity can have multiple instances of another entity, but not vice versa. In other words, rule 2 states that the foreign keys should be stored on the "many-side".

Based on information that we received from ASE during our weekly meetings, each student-composed group are expected to operate and launch *only one* C.R.A.F.T-Sat during their partaking in ASE's course. Therefore, restricting a group to only own one satellite was a viable option. However, we considered the risk of a group starting with a CanSat and then transitioning to C.R.A.F.T-Sat. In that case, each group would own more than one satellite. Thus, a longer-term approach was favoured. A 1:m relationship between student groups and satellites was established: A group can own multiple satellites, but a satellite can only belong to one group.

In general, there are two methods for linking the tables "groups" and "satellites" together:

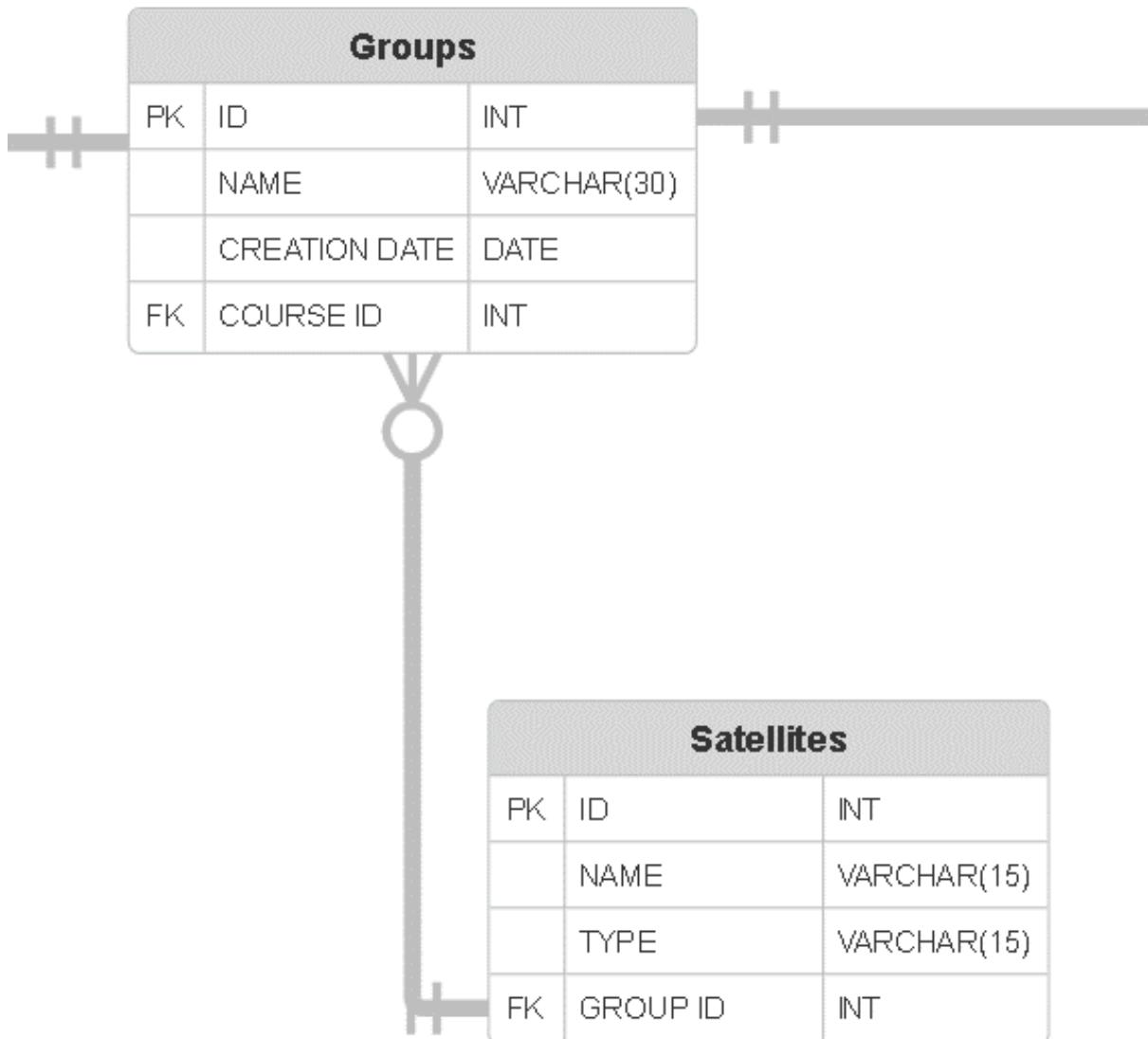
1. Each satellite contains the ID of the Group that it belongs to (i.e., "Group ID" is inside the table "Satellites").
2. Each group contains the ID of every satellite that they "own" (i.e., The table "Groups" contains multiple columns of Satellite ID's).

Although both solutions could technically work, the latter is by far the least desirable: If the foreign key is on the many side, it is impossible to estimate the number of columns needed to contain all the satellite IDs inside the Groups table. Therefore, following modelling rule no 2, the attribute "Group ID" functions as a foreign key and is placed inside the "Satellites" table. This is represented in Figure 120 using ER notation; the "crow-foot" with a vertical line across represents "many", and the vertical line on the opposing end represents "one".

Rule 3: Many-to-many relationships between two entities are placed in a separate table instead of being grouped [85]. This rule is to prevent values from repeating themselves unnecessarily, also coined as **redundancy**. Take for instance the interconnection between the tables "groups" and "users".

It is obvious that a group can (and arguably must) have multiple members, but a user should be able to join multiple groups. The latter is something that was decided by us when designing C.R.A.F.T-GUI. This decision was motivated by requirements regarding

Figure 120: Rule 2: 1:m relationship between groups and satellites (ER diagram)



system longevity [Sub-subsection 3.4.4]; a rule of thumb of 10 years. When designing a database with a strict structure, it is virtually impossible to accommodate for every single edge case: Edge cases usually have to be modified to adapt to the structure, not the other way around. However, it is usually best to strive for a design minimizes the amount of such unfortunate scenarios to improve efficiency, user satisfaction, and the maintainability of the database. Still, some assumptions have had to be made regarding the operation and future scaling of C.R.A.F.T-GUI; one of them being that a user cannot be in multiple groups **simultaneously**. A group should be allowed to join multiple groups at different

Figure 121: DBMS modelling rule no. 3: Creating associative table "UserGroups" (ER diagram)



time intervals. However, it is not desirable to overwrite "Group ID" if it was an entity inside the table "users". This is because that would prevent C.R.A.F.T-GUI from preserving historical data which could be of interest for ASE to keep, and for the students who might want evidence of them partaking in ASE's student programme. A direct connection between students and groups would then create a many-to-many (many-to-many (m:m)) relationship which is not desirable to as there is no predictable way to store them. Rule 3 solves this dilemma by creating a separate table that handles these m:m relationships. This table is widely recognized in the field of DBMS **associative tables**. [86]. By doing so, a m:m relationship is treated as two 1:m relationships.

Rule 3 was followed by placing a table between groups and users called UserGroup, as seen in Figure 121. This associative table contains the different combinations of group IDs and User IDs, thus solving our dilemma. In practice, this would be handled in an SQL query using INNER JOIN to merge Users and Groups together [See user manual for an explanation of INNER JOIN and how it has been used in C.R.A.F.T-GUI's code].

Rule 4: One always-constraint: Group on the side where the always-constraint is. Or: Combine them to one table, or group it where you deem to be the best [85]. Constraints are limitations that are set on columns inside a table, or *in between* tables. This rule is only intended for one-to-one One-to-one (1:1) relationships, but always-constraints can be applied to 1:m relationships [87].

An example of an always constraint used inside C.R.A.F.T-GUI is the handling of the relationship between Groups and Satellites. Here, it is most appropriate to place the always-constraint on the "Satellite"'s foreign key (i.e., Group ID). This is because we believe it is more logical that a satellite must be assigned an owner upon initialization and not afterwards. In SQL, this would be handled in a query which enforces that Group ID (Satellites table) is occupied: "**GROUP ID IS NOT NULL**", meaning that the GROUP ID column must contain actual values. "**NOT NULL**" can also be manually toggled from a SQL management tool such as MySQLWorkbench, as seen in Figure 122. To provide flexibility, administrators should be allowed to attach satellites to a group **after** the group has been created.

Another example of an always-constraint in MySQL used for C.R.A.F.T-GUI is **ON**

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	C
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>						
name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>						
type	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>						
owner	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>						

Figure 122: DBMS modelling rule no. 4: Establishing constraints inside "Satellites" table

CASCADE DELETE which "specify whether you want rows deleted in a child table when corresponding rows are deleted in the parent table." [88]. This keyword has been used for instance on the table "Groups" since when a group is deleted from the database, so should entries in the tables Satellites and UserGroups, which uses Group ID as a foreign key. If a user is removed from a group or deleted, any entries in UserGroups containing the deleted User ID should also be deleted.

```

1 ALTER TABLE `usergroups`
2 ADD CONSTRAINT `fk_usergroups_group_id` FOREIGN KEY (`group_id`)
3 REFERENCES `groups`(`id`)
4 ON DELETE CASCADE;

```

A complete list of instances describing all always-constraints applied to C.R.A.F.T-GUI's SQL database are included inside our user guide to ASE

NNG|TL

6.35 Setting up SQL database - Database Normalization

As discussed in Subsection 6.32.6, 3NF is considered sufficient for database normalization. If a database is modelled, then it is also normalized.

For a database to reach first normalization form (1NF), it must fulfill the following criteria [89]:

- No repeating groups
- Only one value per cell
- A primary key is present

As evident in Figure 123, the first two requirements are accomplished: There are no duplicate groups (i.e., columns) present, and none of the table's cells contain multiple values. However, this database is still not in 1NF due to a missing primary key. The necessity of PKs is to prevent duplicate rows and improve flexibility for making modifications to a table, even though it is linked to other concepts. For instance, if a C.R.A.F.T-GUI

First Name	Last Name	User type
Nathanael	Getaneh	Admin
Thomas	Lunde	Student
Sebastian	Benjaminsen	Student
Daniel	Østmoen	Student

Figure 123: C.R.A.F.T-GUI's SQL database in unnormalized form (UNF) - missing primary key

administrator makes changes to a sensor module (except its PK) in C.R.A.F.T-GUI. Doing so should **not** automatically influence said module's sub sensors in the table "Sub Sensors" [See Figure 119]. A table can only have one PK [89]. In general, there are two ways to include a primary key to a table:

1. Add a column which contains ID's which have been generated solely to be primary keys.
2. Use an existing column (or a combination of columns) as a primary key.

Although the former is a perfectly viable solution, it is generally better to use existing columns as primary identifiers when it is not intuitive to refer to an entity as an arbitrary value, as is the case with a user. For these reasons, the second solution was considered more appropriate.

USERS			
Username	First name	Last name	User type
ngetaneh	Nathanael	Getaneh	Admin
tlunde	Thomas	Lunde	Student
spatent	Sebastian	Benjaminsen	Student
dmoen	LeBron	James	Admin

Green = PK

Figure 124: C.R.A.F.T-GUI's SQL database in 1NF

In Figure 124, each user assigns themselves a unique username. Ideally, we wanted to use email as the user's primary key since they are distinctive by default. However, this was not viable for reasons explained in Subsection 6.33. With these changes, our database was in 1NF.

For a database to reach second normalization form (2NF), the following criterion must be met [89]:

- The database is already in 1NF

-
- No partial dependencies present

Partial dependency is a type of dependency where only parts of a primary key are needed to identify a non-key attribute/column. This dependency type can only occur when the primary key is a composite key (i.e., a combination of columns). Since all of C.R.A.F.T-GUI's database tables contain single-column PKs, this was not a concern. Thus, our database model was already in 2NF.

For a database to reach third normalization form (3NF), the following criterion must be met [89]:

- The database is already in 2NF
- No transitive dependencies present

Transitive dependencies is a type of dependency where two or more non-key attributes are not dependent on the primary key at all, but rather each other. Transitive dependencies occur if there is a 1:1 relationship between these columns. This is not the case for C.R.A.F.T-GUI. Therefore, our database had achieved 3NF could be considered normalized. C.R.A.F.T-GUI's normalized database can be visualized in Figure 125.

Figure 125: C.R.A.F.T-GUI's SQL database in 3NF

SATELLITES			
Satellite ID	Name	Type	Group ID
1	Apollo 1	CanSat	1
2	Apollo 2	CRAFT-SAT	1
3	Apollo 3	CanSat	3

GROUPS				
Group ID	Size	Group name	Creation date	Course ID
1	3	gr1	02-12-2024	1
2	4	gr2	02-12-2024	1

USERS			
Username	First name	Last name	User type
ngananh	Narhanaf	Gethaneh	Admin
lunde	Thomas	Lunde	Student
sparen	Sebastian	Benjaminsen	Student
dmoen	LeBron	James	Admin

COURSES		
CourseID	Start date	End date
1	02/12/2024	31/12/2024
2	04/01/2025	01/04/2024

SENSOR MODULES			
Module ID	Name	Type	Producer
1	GT-81	Thermostat	Adafruit
2	RFM-98	Radio	Arduino

SUB SENSORS				
Sub sensor ID	Name	Module ID	Measuring Unit	Alt. Measuring Unit
1	thermometer	1	Celsius	Kelvin
2	pressure	1	Pascal	
3	gyroscope	1	deg/s	rev/s
4	accelerometer	1	mm/s ²	G

UserGroup		
UserGroup ID	Group ID	Username
1	1	ngananh
2	1	dmoen

GREEN = PK

TURQUOISE = FK

6.36 GUI Frontend Design

NNG|TL

When designing C.R.A.F.T-GUI, there were four requirements in particular that steered our designed choices:

Functional requirement: The ground station must have a GUI with the ability for students to view telemetry data in soft real-time.

Design requirement: GUI must be easily readable. 7. Software source code must be easy to read and well documented.

Stakeholder requirement: The entire system should be made with longevity in mind [rule of thumb: "more than 10 years"].

As previously briefed in Subsection 3.4.2, the term "easy" and "well" are from the perspective of our end user, namely students enrolled in ASE's programme, and C.R.A.F.T-GUI administrators. Fortunately, every team member on this project is in the upper echelon of our user base; university students. This made it easier for us to grasp the needs and wants that such students commonly seek in a GUI. The most notable of which is that a "quality" GUI should:

- support an **intuitive** layout that is easy to navigate through.
- at the very least host all **features** that are necessary for the program to achieve the functionality that it promotes.
- provide **guidance** in case a user is disoriented in using the application.
- provide **consistency** in its design and stability in the application itself and the data it manages.

It is fortunate that all these criteria are also important for upper middle school and high school students; the other two-thirds of our user. However, the degree for which these features must be implemented, is generally significantly higher. A main cause of this is this age group's natural lack of experience with "complex" computer programs. This, in return, can make them less tolerant to "poorly" designed GUI's compared to university which most have prior experience with such.

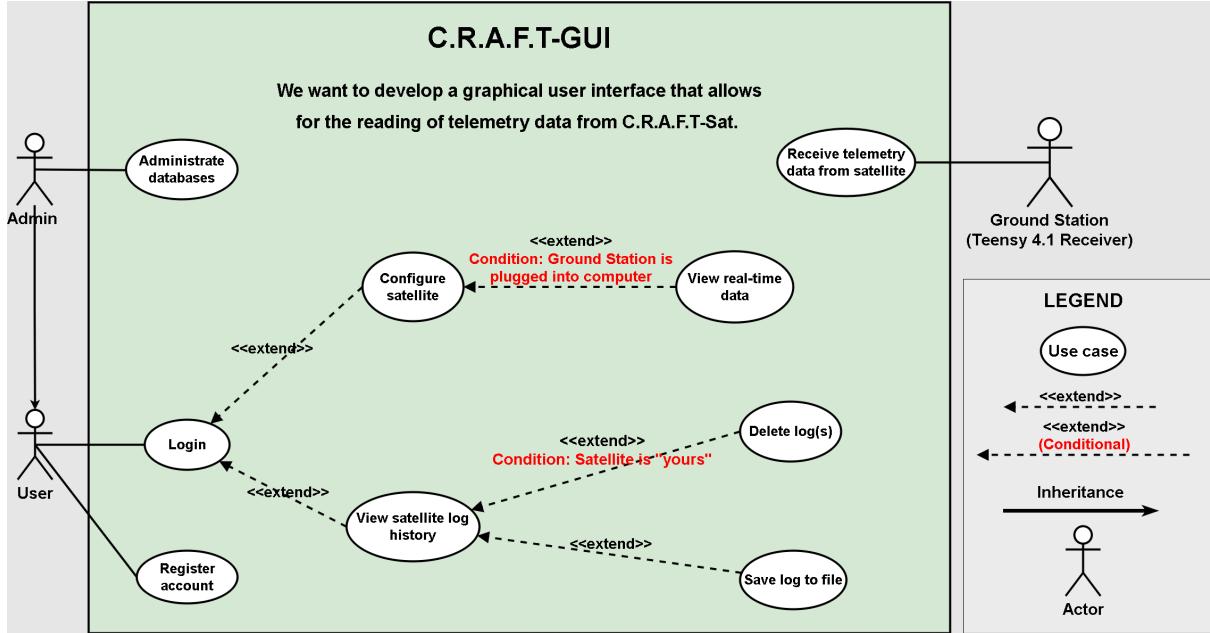
Their young age also indicate that they have less experience with troubleshooting error messages and such compared to university students. This is particularly of concern for the software team since our GUI might not be accessible to the public, and thus not support guidance from online forums and similar mediums. Consequently, this incentivises us to place greater emphasis on designing an **instruction manual** for the GUI and making it **easy to follow and access**.

However, since C.R.A.F.T-GUI will be used as an educational tool (an environment for intuitive thinking), the students shall not feel overly reliant on its documentation. Additionally, almost all students have a natural intrigue for exploring beyond what is necessary. Thus, C.R.A.F.T-GUI should instead aid to promote independent thinking and self-exploration.

Another challenge that we faced when designing C.R.A.F.T-GUI was **striking a balance** between being easy to use for middle school students while simultaneously not have university students perceive our application as too childlike. Through multiple design iterations, our current vision consist of strong colors, but in conservative amounts and subtly placed, for this exact reason.

Before one can code an intuitive piece of software, it should be modelled first. From these factors, criterion and requirements, a Unified Modelling Language (UML) use case diagram visualising C.R.A.F.T-GUI's functionalities was synthesised:

Figure 126: Use case diagram - C.R.A.F.T-GUI



Referring to Figure 126, users should be able to configure their satellite. This use case involves informing the GUI the sensors to be plotted, along with their prefixes. The design for this page can be found in Figure 129². Although not represented in Figure 126, user should be able to bypass login and still plot telemetry data. This feature has been coined by us as "offline-mode"³ and can be observed in Figure 127. However, if logged in, users are able to access and save past C.R.A.F.T-Sat and CanSat mission logs. They should also be able to delete logs, **but only if** the satellite belongs to one of the user's groups. This caveat is represented in Figure 126 as a conditional extend arrow pointing from "Delete log(s)" to "View satellite log history". In UML, «extend» specifies that a use case is optional extension of the use case it is pointing to. An administrator in C.R.A.F.T-GUI should share the same functionalities as a regular user. This is symbolized through the inheritance arrow pointing from "Admin" to user. In addition, administrators should be able to administer C.R.A.F.T-GUI's databases. A concept design for these administrative pages can be seen in Figure 131.

²The designs in Figures 127-131 are concept designs and can therefore, slightly differ from the final product, both in features and aesthetics

³In this design, an internet connection is required for users to configure their satellites, even in "offline-mode".

Figure 127: Login page - C.R.A.F.T-GUI

C.R.A.F.T
CanSat Reserach and Advanced Fabrication Team

E-mail

OTP

Password

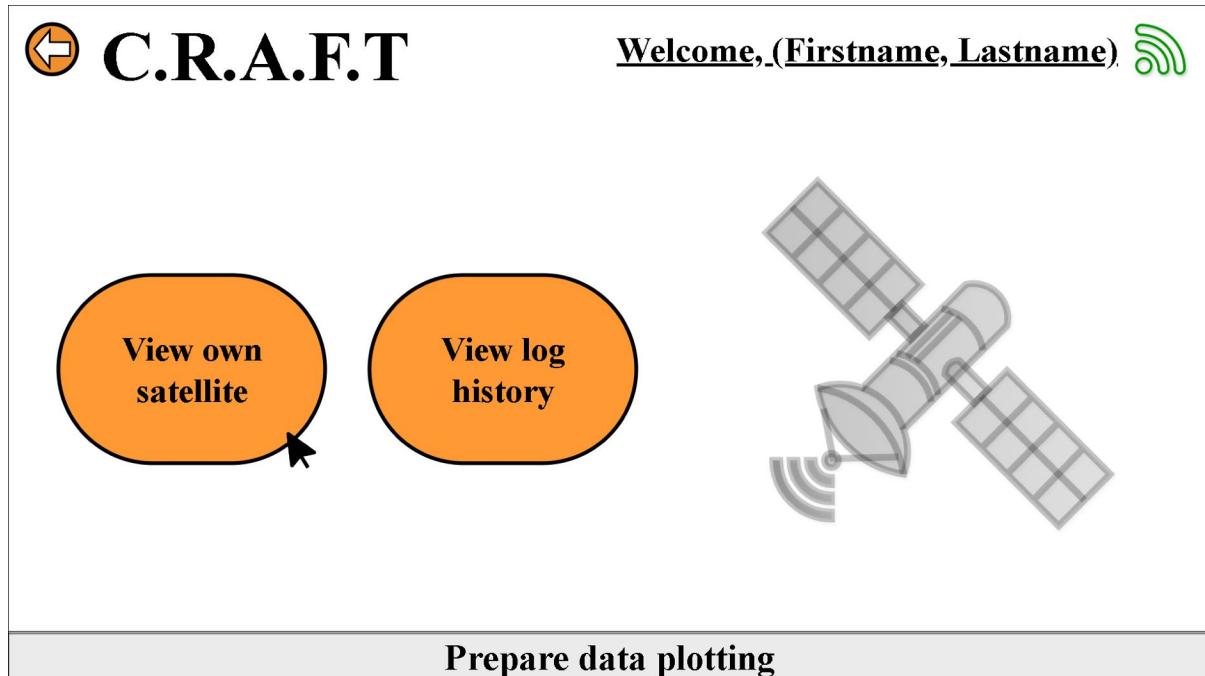
Login

Use offline Swap to OTP Register user

Press circle to login

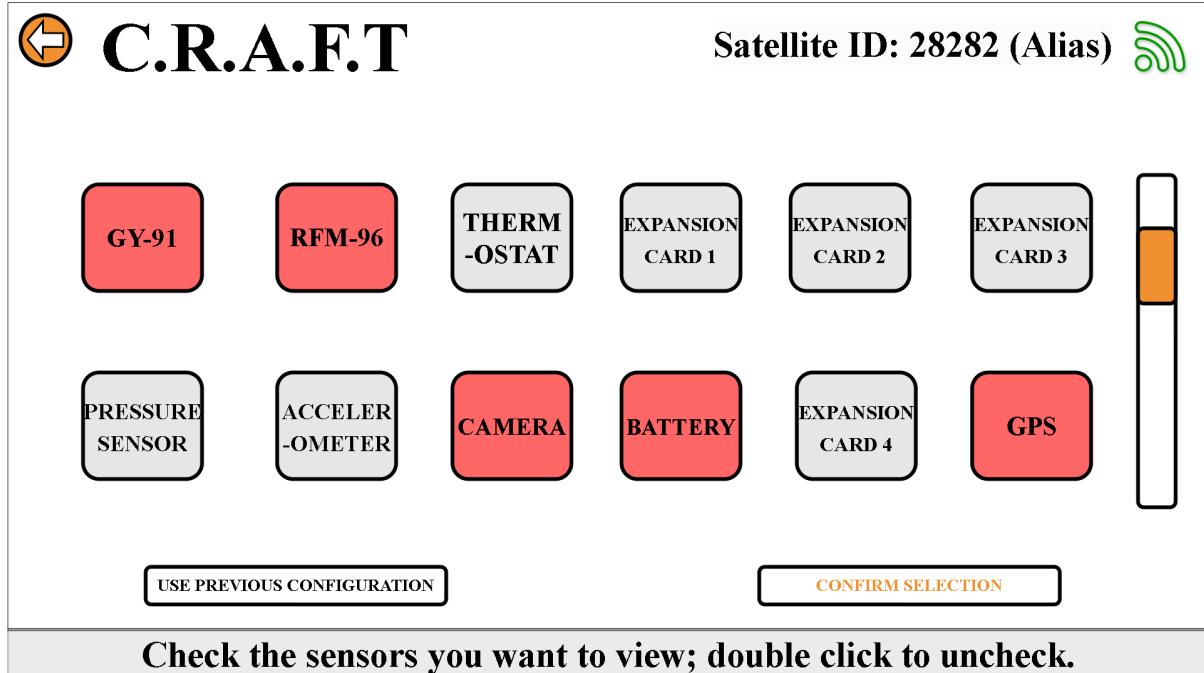
- User specifies the authentication method to employ.
- User registration is explained in sub-subsection 6.33.1

Figure 128: (Home page (regular user) - C.R.A.F.T-GUI



- Decision is to use this layout was motivated by the results from our user testing.

Figure 129: (Configuring satellite - C.R.A.F.T-GUI



Ideally, to prevent tedium, there should be a button that allows the user to save and reuse a satellite configuration. However, this has currently not been implemented due to time constraints.

Figure 130: Log histories - C.R.A.F.T-GUI

SORT BY:	MOST RECENT	OLDEST	FILE SIZE
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			
DATE: dd.mm.yyyy, LOCATION: latitude/longitude, DURATION: hr:min.sec, SATELLITE: ip_address ("Name")			

Click log to view more

- The rows in Figure 130 contain some general information about the satellite mission. The intention behind this feature is that users can better identify their log of interest to download, rather than accidentally downloading the "wrong" log.

Figure 131: Administrator page (Groups) - C.R.A.F.T-GUI

Group ID	Name	Size	Course ID
js	John	Smith	1
ab	Ali	Baba	2

REGISTRATION FORM



Hover over a button

-
- Administrators have access to the following pages (each with similar design as Figure 131):
 - Groups
 - Sensors
 - Users
 - Courses
 - Satellites
 - Sensors
 - The "REGISTRATION FORM" is different for each administrator page and has, therefore, been abstracted in Figure 131

Other facets to note regarding this design:

- An instruction widget is placed at the bottom of every single page, spanning the application window's entire x-axis. When hovering over a button, the text inside this widget shall contain a brief, yet insightful description of what the button does. This feature is appropriate as including some of C.R.A.F.T-GUI's documentation within the application itself improves user accessibility and user friendliness. The consistent placement and omnipresence of this instruction widget also improve ease of use which is one of our mandatory design requirements.
- White background: Improves readability that account for low brightness displays and poor lighting conditions. Decision is also based on results from our user testing [Subsection 8.6.4]
- No hidden menus for improved ease of use.
- The “Go back” button (along with other widgets) are iconized rather than containing plain text. This is because icons draw more attention and subconscious understanding compared to plain text which requires reading to grasp its meaning.
- Vertical scroll bars are in the Configuration page and Admin pages to compensate for potential lack of screen real estate.

NNG|TL

6.36.1 C.R.A.F.T-GUI - Code Structure

One major benefit of employing an object oriented philosophy for C.R.A.F.T-GUI's code is that it can make code modular, reducing the program's lines of code. Another benefit of instantiating base class is **improved safety**. This is because when reusing a class instead of duplicating code explicitly for every object, then the system coder/maintainers *only* have to configure that one base class correctly, and after that it can be reused multiple times. This also makes troubleshooting easier since there are less code to parse through and a lot of functionality is centralized to one area/class, creating a single potential point

of failure rather than multiple.

For instance, the top layout of C.R.A.F.T-GUI is **almost identical** for each page [Sub-section 6.36]. Another staple for every single page is the inclusion of an instruction label at the page's bottom.

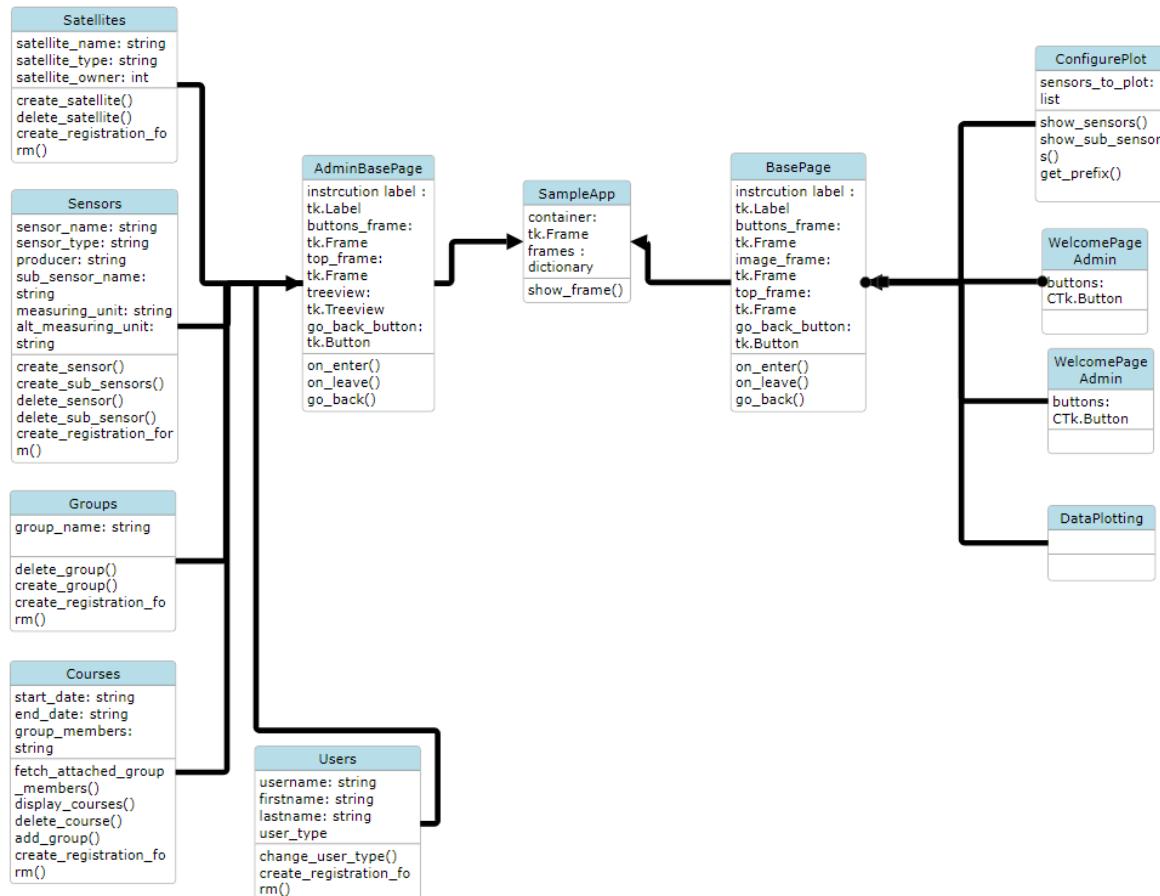
Without OOP the same lines would have to be pasted into every single init() function for each page. If a single accidental modification is made to the code, but is **not fatal** (i.e., does not cause an error message), then that modification could be virtually impossible to identify as there are too many similar initialization functions to parse through. However, if centralized to one class, then this is easier. To achieve this, layouts that are almost guaranteed to be in every page was transferred to a class called BasePage(). This is visualized in Figure 132

Now, other pages/classes just have to pass BasePage as a parameter to inherit said layout. This is visualized in the class diagram in Figure 132. The same principles also applied for admin-only pages. For that, the class AdminBasePage has been created, which also inherits from SampleApp().

If there are minor differences that one want to establish while still inheriting from BasePage, then that can easily be accomplished by overriding functions and variables from BasePage. For instance, Login page should not have a Go Back button since Login is the start page of C.R.A.F.T-GUI. That can easily be accounted for by destroy redundant widgets using Tkinter's built-in .destroy() function.

By simply implementing BasePage to our frontend code, the number of lines in our frontend code was lowered by approx. 500 lines which is immense. Complexity is reduced, increasing stakeholder satisfaction from ASE and their IT department. This is because long term costs with regards to system maintenance, expansion and troubleshooting is reduced. Lowered complexity also simplifies the documentation drastically and makes it more plausible to document our system well, which is one of our primary design and stakeholder requirements. More detailed documentation which more importantly describes a **systematic structure**, to our system, can make C.R.A.F.T-GUI easier for ASE to expand if they want to.

Figure 132: Class Diagram Visualizing C.R.A.F.T-GUI's Infrastructure



6.37 DIY Vibration Test

One facet of Objective 7 is to design a procedure for a vibration test that ASE's enrolled students can follow to conduct their own vibration tests on C.R.A.F.T-Sat and CanSat. To show ample consideration to all ages in our target audiences, the following design requirements molded our decision. Similar experiments have been conducted, with good results [90]

- System must be easy to use
- System must be affordable (since its target for young students)
- System context: End users range from upper middle school, to university students.
- system must be **safe to use**

Ease of use also involves ease of initialization/setup and understanding the system. Industrial shakers are often too expensive and large to be a realistic purchase for a educational institution, let alone a single student. These shakers can also involve a lot of components that make assembly and maintenance by an amateur unrealistic and violate or design requirement of being safe to use. An industrial shaker would also diverge from our system scope: the purpose of C.R.A.F.T-Sat is to act as an affordable and user friendly stepping stone to learning about and potentially mastering industrial satellites. Recommending the enrolled students to use an industrial shaker would thus violate the principles that are our system is scoped around. Therefore, other sources of vibrations were considered.

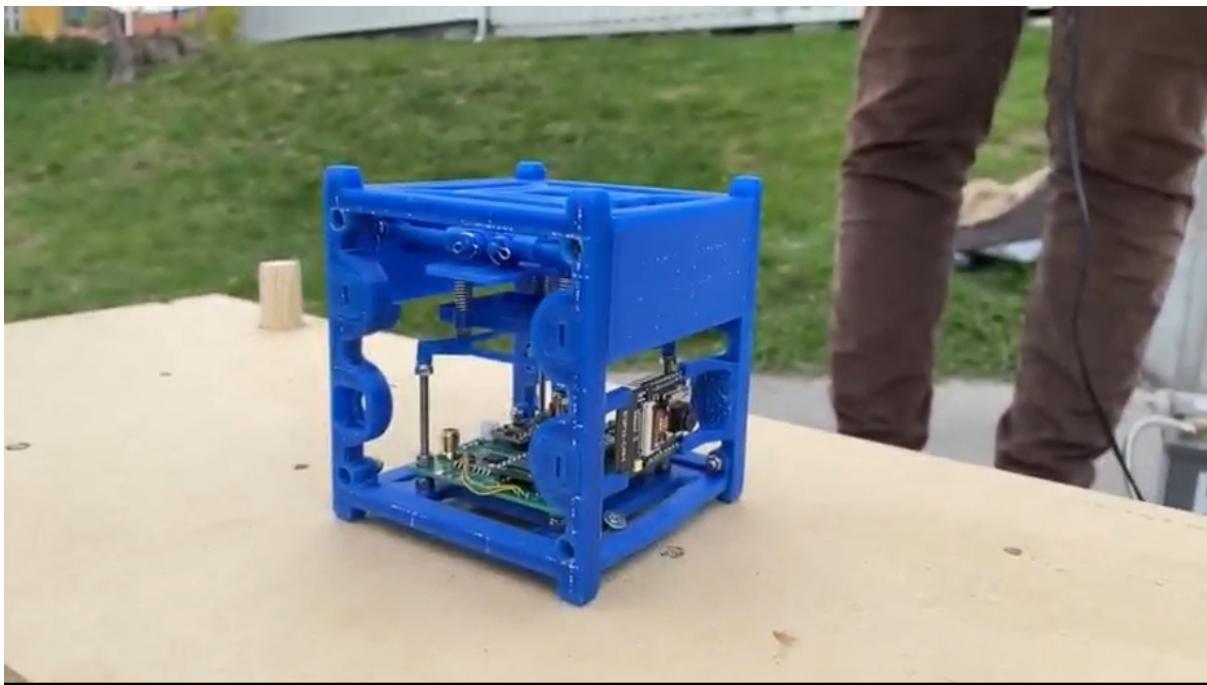
6.37.1 Solution - Homemade Shaker using a Speaker

Concept: 3D-printed rings are placed on top of a speaker. A flat surface is then mounted on top of the rings. A C.R.A.F.T-Sat or CanSat is then mounted to the surface. The speaker will cause the rings to vibrate, which causes the flat surface to vibrate which consequently causes the satellite to vibrate. A demo of this setup can be seen in Figures 133 and 134. Similar experiments have been conducted, with good results [90]

Figure 133: Homemade Shaker: Simple 3D of Rings placed on Top of Speaker



Figure 134: Homemade Shaker: Test Run Outside



Pros:

- Most commercial speakers are able to vibrate at much higher frequencies than commercial power tools, better reflecting the high frequencies that can be present during actual satellite missions.
- Speakers and amplifiers can be found second hand for cheap, sometimes even for free. Most schools in Norway already have speakers.
- This solution works for both CanSat and C.R.A.F.T-Sat as there is no specific mounting required for them; they just need to fit on top of the flat plane. Mount to speaker requires only simple 3D print.
- Frequency can easily be adjusted in real time via free web applications or audio sweep videos on YouTube. Amplitude can easily be adjusted by increasing the speaker volume on the computer.

Cons:

- Requires an amplifier to generate enough amplitude to cause visible vibration. Bigger speakers are more power hungry.
- Speakers can be very loud and reach high frequencies that can cause discomfort and damage to hearing if not wearing ear plugs. This can also cause noise complaints by bystanders.

-
- Bass speakers and sub woofers cause the most vibration, but cannot reach frequencies as high as regular speakers. Bass frequencies occur at around 30-250 Hz.

7 Production

7.1 Sensor Array

TL|DØ

To answer our 9th and 10th Functional requirements 3.2 we need to limit the size of our radio packets being sent. A larger packet will require a longer sending time and are more prone to errors. Although we cannot account for legal restrictions regarding duty cycles 7.2, we can reduce the packet size, to reduce sending time.

For the GUI to display the data, it needs a way to differentiate the incoming data string. With the use of a database as seen in section 5.12.2. We decided to differentiate data based on IDs located in the database.

In earlier versions of our C.R.A.F.T-Sat library, we separated data with the use of sensor names, the problem with this was it required much more data to be sent wireless, thus reducing how often we can send and increasing the amount of data.

ID is index

One approach to connect IDs to their respective data is to have an array with sensor data, where the index of the array is the same as the ID. With this approach, you have an easy way to identify data, by just inputting the ID to whichever data you want.

This does however have some downsides. If IDs are large, the array would need to be as large as the largest ID. If there are large gaps in ID ranges it would also waste a lot of space. There is also no easy way to connect a name to the respected ID and data.

Array of pointers

Another approach is to create a sensor class or a struct with template types. Where the sensor data can be stored as any type that you want. The collection of the object can be stored in an array of pointers to the sensor class. This gives the highest flexibility and might give better memory efficiency since pointers take up a constant space. This also enables a way to tie ID, name, and data together, without being dependant on the location in the array.

The downside to this approach is complexity. Since this is a library used for Arduino/-teensy, there are no simple ways to use smart pointers, like in normal C++. The use of raw pointers can lead to memory leaks, if not handled correctly. The use of this also forces the students that are going to use the library to think about custom data types created by sensor library developers which takes away the simplicity the library is supposed to have.

Array of sensors

A third approach for this problem is to reduce the complexity of the last suggestion. Create a sensor class or struct without template types. Cast all sensor data to the same object type and put the objects in an array of sensors.

The downside to this is that all sensor data has to be casted to the same datatype, and it uses a lot of memory if the amount of sensors increases.

We decided to implement an array of sensors. Where sensors are implemented as a struct, containing **id**, **data**, and **name**. Objects of this struct are then placed into an array of sensors, where the index is independent of the sensor ID. With this method, we get an easy way to encapsulate IDs, names, and data. The reason for choosing the third approach is because of our end users. ASE has expressed a desire for the code to be easily

read, with easy-to-follow lines and simple instructions. The first approach would probably not work since IDs could not be a large number, and the ability to send custom sensor data would be problematic if the students would be limited to the id length. The second approach could work with the right abstraction, but as mentioned would require the users to use custom datatype provided by developers, making it more complex.

To keep the IDs in the database and library the same, the library defines them at the top of the header file using #define [Listing. 7.1] . This makes the ID similar to a constant, without taking space as a variable in memory. This also makes it easy to change the IDs in the library if needed.

```
1 //DATABASE SENSORVALUE ID'S
2 #define MPU_ACCELEROMETER_X 2
3 #define MPU_ACCELEROMETER_Y 3
4 #define MPU_ACCELEROMETER_Z 4
5 #define MPU_GYROSCOPE_X 1
6 #define MPU_GYROSCOPE_Y 5
7 #define MPU_GYROSCOPE_Z 6
8 #define MPU_MAGNETOMETER_X 7
9 #define MPU_MAGNETOMETER_Y 8
10 #define MPU_MAGNETOMETER_Z 9
11 #define MPU_TEMPERATURE 10
12 #define BMP280_PRESSURE 12
13 #define BMP280_TEMPERATURE 11
14 #define NTC_TEMPERATURE 13
15 #define RSSI_VALUE 14
16 #define PACKET_SNR 15
17 #define PACKET_FREQUENCY_ERROR 16
```

To keep track of each sensor's index we have created two methods. A function with O(n) runtime complexity, where you input the ID of the sensor you want the index to. To reduce runtime we decided to create a O(1) method of finding the indexes, although this will only work for the standard sensors on the C.R.A.F.T-Sat shield. This method uses a struct which holds information about all the indexes. The indexes in the struct is set at run time adding the index to the sensors enabled by the constructor. This is done by the "add_sensor" A.17 function which returns the index of where the sensor was added.

7.2 Honoring duty cycles

TL|TL

For our system to answer the requirements set by the Norwegian Government 3.1, we needed to implement a way for our system to honor the duty cycle as described in section 5.6. A duty cycle refers to how long a device is on and off during a defined time period. In our case this how much time the radio is allowed to send data before needing a period of time off, before being allowed to send again. As stated in 5.6, we need to honor a

10% duty cycle, which means the transmitter on the RFM-96 G.3.6 can be on 10% of the time, the rest it needs to either receive or be idling.

To be able to honor a 10% duty cycle we need a way of knowing how long the transmitter is on. Luckily semtech, the developer of the SX1276 chip located on the RMF96 G.3.6 has created a "Time on air" formula where you can input your radio settings and output how much time the transmitter will use to send the packet. The radio settings that affect sending time is:

- Bandwidth
- Preamble size
- Spreading Factor
- Payload Length
- Cyclic Redundancy Check
- Low Data Rate optimization
- Explicit or Implicit Header mode

Since the library should be "plug and play" 1.1.1, we have decided to implement a function that calculates the "Time on air" and prohibits any sending of data before a period of 90% is over. The implementation is based on the formula provided by Semtech SX127X datasheet [91]. A LoRa Air-time calculator developer has performed their own validation of the formula provided by Semtech, which concludes that the formula matches real life transmitting results [92]. Due to time constraints and equipment availability, we have decided not to test their conclusion, and assume their validation is correct.

The equations below are taken directly from the SX127X datasheet[91]:

- PL is the number of Payload bytes (1 to 255)
- SF is the spreading factor (6 to 12)
- IH=0 when the header is enabled, IH=1 when no header is present
- DE=1 when LowDataRateOptimize=1, DE=0 otherwise
- CR is the coding rate (1 corresponding to 4/5, 4 to 4/8)

Definition of symbol rate:

$$Rs = \frac{BW}{2^SF} \quad (46)$$

Definition of symbol time:

$$T_s = \frac{1}{Rs} \quad (47)$$

Preamble time, where $n_{preamble}$ is the preamble size, 4.25 symbols is added by LoRa.

$$T_{preamble} = (n_{preamble} + 4.25)T_s \quad (48)$$

Payload symbols is calculated:

$$n_{payload} = 8 + \max(\text{ceil}[\frac{8PL - 4SF + 28 + 16CRC - 20IH}{4(SF - 2DE)}](CR + 4), 0)) \quad (49)$$

Payload time

$$T_{payload} = n_{payload} \times T_s \quad (50)$$

Total Time on Air:

$$T_{packet} = T_{preamble} + T_{payload} \quad (51)$$

Our implementation of can be seen in Listing 3. Our function does not have any inputs, because it uses the currently used radio settings and calculates payload size depending on the amount of data in the sensor array.

```
1 double Craft_sat::calculate_time_on_air()
2 {
3     double symbol_rate = BW/(pow(2,SF));
4     double symbol_time = 1000/symbol_rate;
5     double preamble_time = (preamble_size + 4.25)*symbol_time;
6     double tmp = ((8*payload_length) - (4*SF) + 28 + ((CRC_enabled
7         ? 16 : 0)) - (explicit_header ? 0 : 20))/(4*(SF - (LDRO ? 2
8         : 0)));
9     double payload_symbols = 8 + max(ceil(tmp)*(CR),0);
10    double payload_time = payload_symbols * symbol_time;
11    double packet_time = preamble_time + payload_time;
12    return packet_time;
13 }
```

Listing 3: LoRa Air Time Calculator in Arduino/C++

7.3 Setting up SQL Database

7.4 Connecting SQL database to Microsoft Azure

Since our SQL database will be running on a external server once deployed, it was appropriate to establish such a connection already in the developmental stages. The most ideal setup would of course be connecting to ASE's server, but that was unfortunately not possible due to a lack of authorisation credentials. Instead, we connected our SQL database to a cloud based instance hosted by Microsoft Azure. Azure is a cloud subscription service by Microsoft that allows for quick and simple cloud based storage and services, ideal for our testing purposes [93]. The process of connecting a Python to an SQL is quite agnostic to what server the database runs on. Thus, the integration procedure used can easily be extrapolated to any server instance.

Figure 135: C.R.A.F.T-GUI's SQL Database Hosted on Microsoft Azure

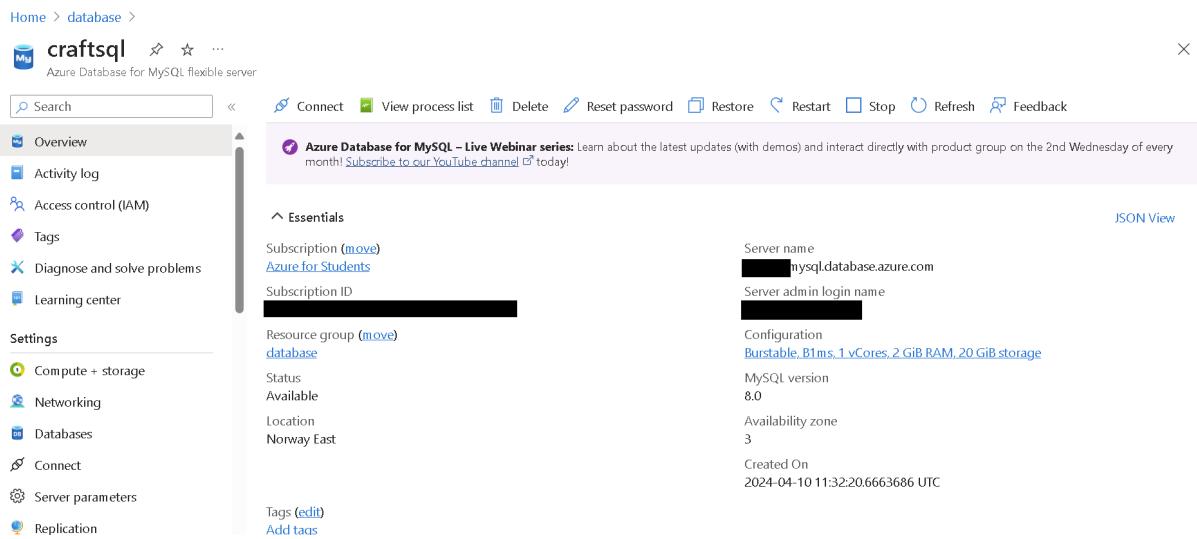


Figure 136: Connecting Remote Database to MySQLWorkbench

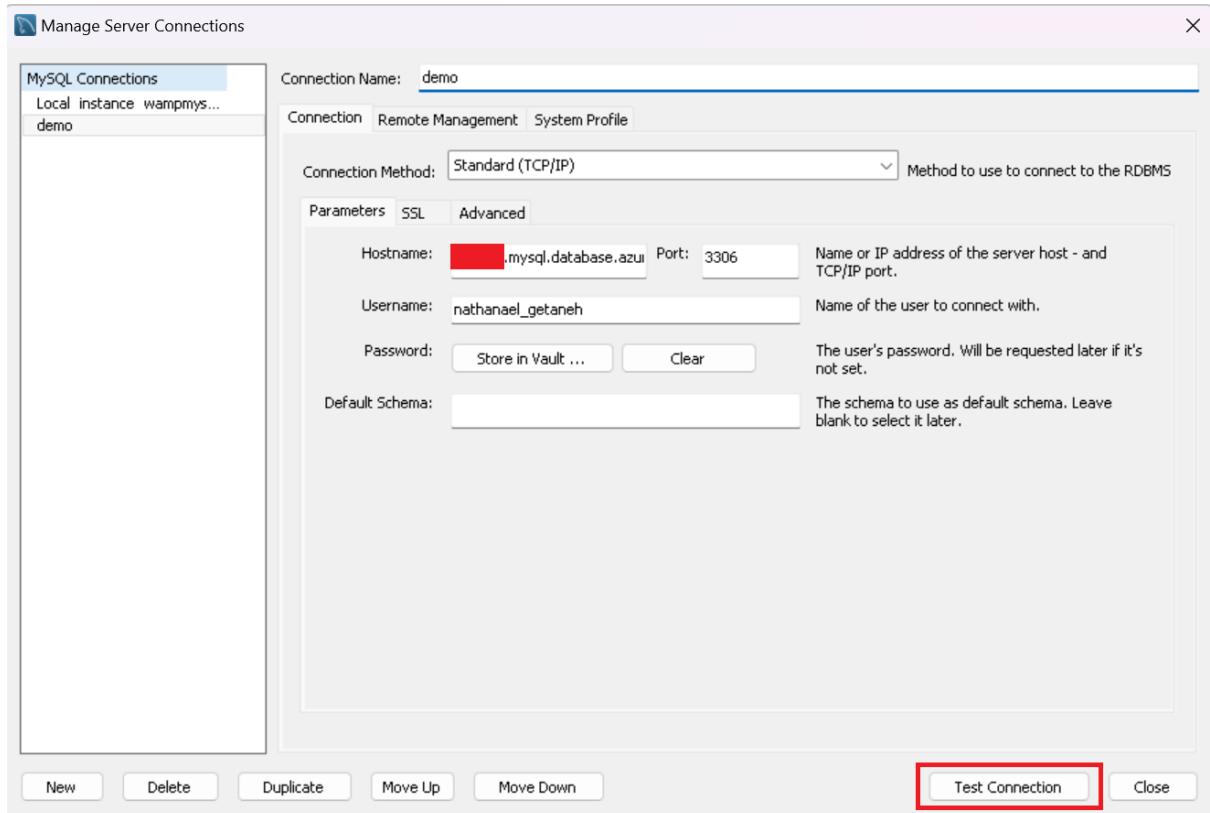


Figure 137: Caption

Welcome to MySQL Workbench

MySQL Workbench is the official graphical user interface (GUI) tool for MySQL. It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

[Browse Documentation >](#)

[Read the Blog >](#)

[Discuss on the Forums >](#)

MySQL Connections ⊕ ↻

Local instance wampmysqld64	demo
└ root	
└ localhost:3306	

To establish a communication between the external server to our backend, one only needs to insert the name of the server and database, along with username and server password. For complete control over the entire SQL database, MySQLWorkbench (a free, dedicated

GUI) was used. As Figure 136 shows, only four credentials are needed: the host name, username, password and port number of the remote database. NOTE: 3306 is port being used as that port is the default port used for SQL databases.

7.5 Protecting SQL Database

7.5.1 Storing Database Credentials

It is not safe to directly insert the access credentials to our Azure SQL database as plain text as anyone that has access to the source can easily break into craftsSQL with it. Instead, C.R.A.F.T-GUI stores these credentials inside a separate Python script [94].

```
1 db_credentials.py
2
3     credentials= {
4         'host': "hostname",
5         'user': "username",
6         'password': "password",
7         'database': "database_name",
8     }
```

In 7.5.1, the login credentials to our database is formatted as a dictionary; a data structure that is indexed by keys rather than a range of numbers [95]. Now, the credentials inside db_credentials.py are indirectly called by the application server, referencing its key instead of actual value.

```
1 try:
2     db = mysql.connector.connect
3     (
4         host=credentials["host"],
5         user=credentials["user"],
6         password=credentials["password"],
7         database=credentials["database"],
8     )
```

7.5.2 Protecting SQL Database - Input Sanitation

In computer programming, **input sanitation** is a "cybersecurity measure of checking, cleaning, and filtering data inputs from users, Application Programming Interface (API)s, and web services of any unwanted characters and strings to prevent the injection of harmful codes into the system" [96]. Malicious attackers may try to access the database by formatting their entries so that they can be wrongly misinterpreted by the application as an SQL query. This technique is commonly referred to as **SQL injections** [96].

To mitigate the risk of SQL injections, the following methods have been implemented to C.R.A.F.T-GUI's code:

- .strip(), a built-in Python function, has been used to remove white spaces "at the

beginning and end of a string" [97]

- Instead of being passed to "API for DB Interactions" [114], user request is handled by API endpoints. The request is first formatted into a dictionary. If the data cannot be parsed as such, then the user request is rejected and unable to continue towards the SQL database. See Subsection 6.32.7.

7.5.3 Programming SQL Database

To create our database and the schema inside our database (which is from now on referred to as "craftsql"), MySQLWorkbench was used. This is a free MySQL editor which functions as a user interface for MySQL. To export craftsql, MySQLWorkbench features an Data Export Wizard which generates the SQL queries needed to create said database which is then saved as a file [98]. To import the database, ASE would simply have to paste in this file contents into an MySQL editor of their choice. We recommend Andøya Space Education to refer themselves to MySQLWorkbench documentation for using Data Export, since it is the most up to date. Link to said documentation is included inside C.R.A.F.T's user manual to ASE

NNG|TL

7.6 Using Tkinter and CustomTkinter - Frontend

7.6.1 Loading up C.R.A.F.T-GUI - SampleApp()

The first class that was written code for was SampleApp as all other Classes are dependent on it [Figure 132]. It is the class responsible for loading all of the pages that the user will interact with via our user interface.

```
1  class SampleApp(tk.Tk): # intermediary when transitioning between frames/pages.
2      def __init__(self, *args, **kwargs):
3          tk.Tk.__init__(self, *args, **kwargs)
4
5          container = tk.Frame(self)
6          container.pack(side="top", fill="both", expand=True)
7          container.grid_rowconfigure(0, weight=1)
8          container.grid_columnconfigure(0, weight=1)
9
10         self.frames = {} # dictionary
11
12         for F in (
13             Login,
14             RegisterUser,
15             WelcomePage,
16             ViewOwnSatellites,
17             ConfigurePlot,
18             AdminEntry,
19             Courses,
20             Groups,
```

```

21         Users,
22         Sensors,
23         Satellites,
24     ):
25         frame = F(container, self)
26         frame.configure(background="white")
27         self.frames[F] = frame
28         frame.grid(row=0, column=0, sticky="nsew")
29
30     self.show_frame(Login)
31
32 def show_frame(self, cont): # container
33     # Raise the new frame to be displayed
34     frame = self.frames[cont]
35     frame.tkraise()
36     print(f"{cont} frame raised")

```

The for-loop in lines 12-28 loads up each page. Loading up all pages upon opening C.R.A.F.T-GUI can be slower than instantiating each page when their are needed. There are two major benefits, however, loading all pages at once: First, loading all pages upon opening C.R.A.F.T-GUI means that all the most resource-heavy processes is done in the beginning. Once the pages have been loaded, transitioning between pages can be much faster since the page was created upon opening the app. This means that the only function that has to run to change page is show_frame(self, cont). This function is simple and not resource intensive, so it will not take much time to complete. Another major benefit is if all pages are successfully spun at application execution, then that is a good indicator that the user will not experience any unexpected crashes when using C.R.A.F.T-GUI. As a user, it could be quite frustrating if a page was functioning fine but then crashed when moving to a different page. This could damage the trust that our user base has on our application. It is much better to know that something does not work as early as possible until later down the line.

```

1 if __name__ == "__main__":
2     app = SampleApp()
3     app.geometry("1920x1080")
4     app.mainloop()

```

```

1 if __name__ == "__main__":
2     app.run(host="localhost", port = 5000, debug=True)

```

It is in app.run() where ASE can set the "host" and "port" parameters to IP address and port of the external server that they want to run Flask on.

7.6.2 Implementing Layout - Frames

Referring to our front-end design in Subsection 6.36 and code structure in Sub-subsection 7.7, the layout for almost all pages can be abstracted as shown in Figure 139.

Figure 138: General Layout of C.R.A.F.T-GUI Page

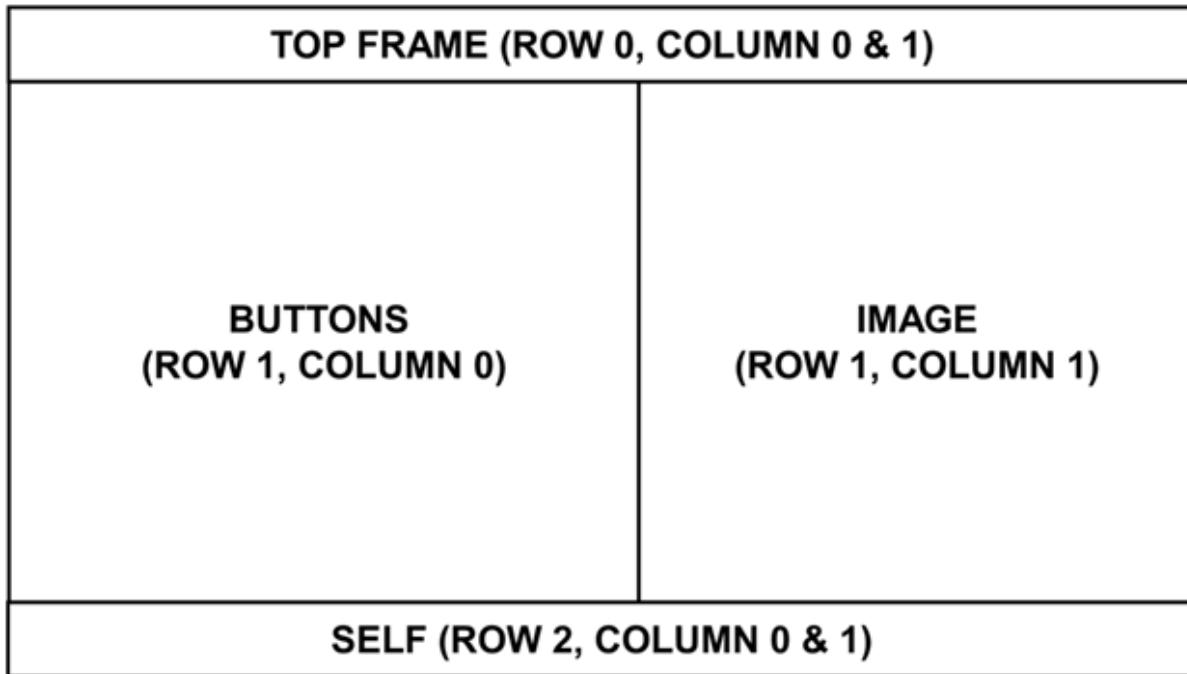
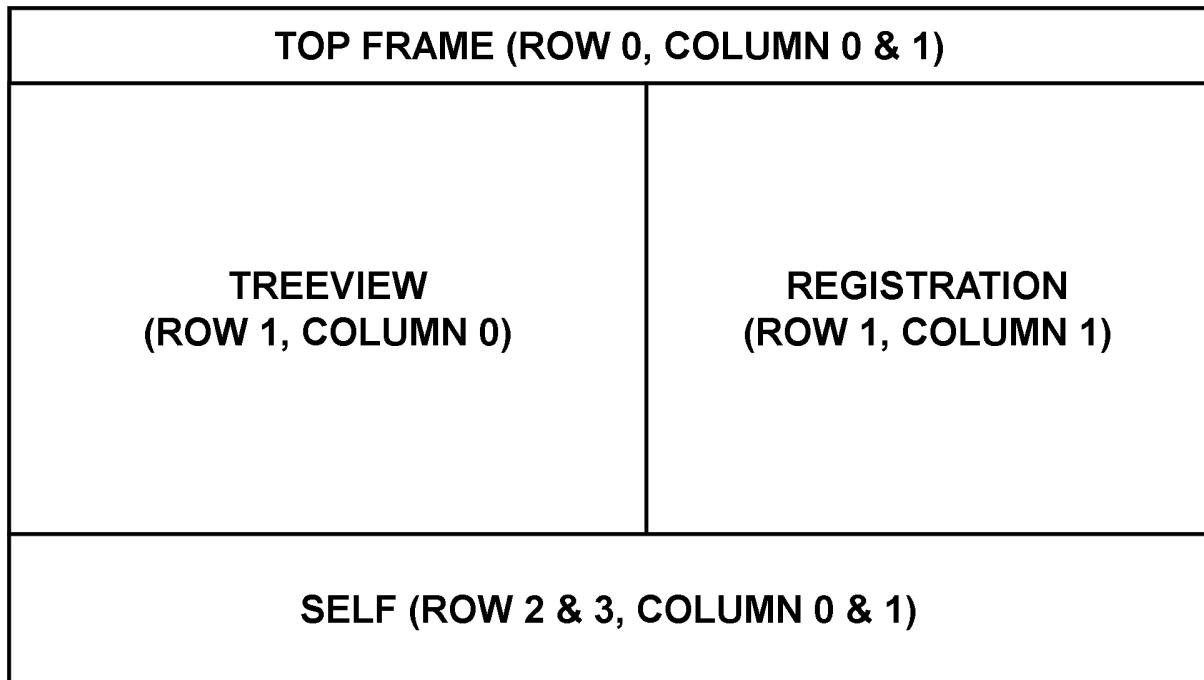


Figure 139: Layout for Admin pages in C.R.A.F.T-GUI



In Tkinter, there are three ways to position widgets in certain locations: place(), pack(), and grid(). Place() is the most straightforward as it accepts the exact x, and y coordinates

of where you want the widget to be placed on the screen. Although the simplest, we avoided place() completely as it does flexible to different screen sizes, and can require a lot of fine tuning. This lowers end product value for our client and end user as it violates requirements regarding usability as widgets can be placed inconsistently for each user, depending on the screen size, aspect ratio and resolution of each monitor. .pack() is a better approach as it packs the widget inside the desired frame, and accepts the "side" as a parameter where the programmer can specify where the button should be positioned relative to the frame that it exists in ("top", "right", "bottom", "left", and "center"). Since it is easier to use than .grid(), .pack() has been used inside frames where there are minimal number of widgets inside said frame, and if the desired position of the widgets can be categorized to one of the aforementioned options inside the "side" parameter. An example of this would be the frame to store the background image [See Sub-subsection 7.6.4]. However, for frames containing multiple widgets or widgets spanning the entire screen (e.g., Top Frame, Buttons, and Instruction Label), .grid() was used. As its name implies, .grid() positions widgets on a virtual grid (which is defined and configured in code), accepting row and column as parameters. For instance, referring to Figure 139, the buttons frame can be stationed at row=1, and column=0. These exact parameter values can be set relative the frames surrounding it: E.g., top frame is one row above Buttons Frame and there are no frames above it, so Top Frame's is at relatively speaking at the zeroth row. Observing Figure 139 again, Top Frame and the Instruction Label (inside Self frame) span across multiple columns. To accommodate for this, .grid() has a parameter called "columnspan" which defines for how many columns the frame or widget should span across.

```
1 # Top frame for logo and go back button
2 self.top_frame = tk.Frame(self, bg="white")
3 self.top_frame.grid(row=0, column=0, sticky="ew", columnspan=2)
4 self.top_frame.grid_columnconfigure(1, weight=1)
```

Inside the listing above, a Tkinter Frame (named top_frame) is initialized. The parameter accepted by this Frame is its master frame (i.e., the frame that the new frame is positioned inside). The second parameter, "bg", sets the background color of the frame (bg). "self" is inputted as top frames master as self represents the application window. In line 3, top frame is pinned using .grid() at row=0, and column=0. Columnspan has been set equal to 2, meaning that top frames column should span across two columns from its starting point, meaning the top frame occupies columns 0 and 1, at the zeroth row. Another parameter used in line 3 is "sticky" which determines how a frame or widget should be elongated/stretched. Setting sticky equal to "ew" (East-West) means that the top frame should only be stretched across the entire x-plane, but not in the y-direction. Another major benefit of using .grid() is that each frame can be assigned a weight, which decides how much space a frame should occupy when the application window is resized. Here, column 1 of top frame (i.e., where the welcome (Firstname, Lastname) and the WiFi status symbol) has the lowest possible weight of 0 and it is not desirable that these widgets should have priority space accommodation compared to others larger and more important widgets such as the buttons and background image.

7.6.3 Creating Buttons

By default, buttons in Tkinter is solid-colored, rectangular-shaped with no rounded corners. This created an aesthetic that is generally not considered modern by today's standards. They are also not as attention grabbing and describing as what you would want from a button. Fortunately, Tkinter allows for buttons to import images and modify their shape that way. The Listing below shows how the "go back" button from Figure 140 was imported as a button.

```
1 # Load images
2 self.go_back_img = tk.PhotoImage(file="craft_gui_go_back_button.png")
3
4 # Go back button
5 self.go_back_button = tk.Button(
6     self.top_frame,
7     image=self.go_back_img,
8     borderwidth=0,
9     highlightthickness=0,
10    command=self.go_back,
11 )
12 self.go_back_button.grid(row=0, column=0, padx=20, pady=10, sticky="w")
13 self.go_back_button.image = self.go_back_img # Keep a reference
```

In line 2, the image file is imported in to Python file for the front end code. PhotoImage is a package that has been imported from Tkinter and handles the image file so that it can be utilized by other Tkinter widgets, such as buttons. Lines 6-12 instantiates a button widget. The "self." prefix simply allows for a function or variable to be reused outside of its original scope. Tkinter's built in button (tk.Button) has a parameter called image which in Line 8 has been set equal to the imported go back image. Line 12 places this button on the top-left corner.

Still, there are some buttons that we want to contain text, be "normal-shaped", and modifiable. The problem with standard Tkinter's outdated button appearance remains. To remedy this, CustomTkinter buttons was used instead as they allow for buttons to have rounded corners. Additionally, CustomTkinter buttons have built-in "Hover colour" parameter that changes the colour of the button when the mouse pointer is hovering on it, which is a nice feature as it gives visible response to the user.

```
1 self.view_own_button = customtkinter.CTkButton
2 (
3     buttons_frame,
4     text="View own satellites",
5     font=("Helvetica", 20),
6     fg_color="orange",
7     text_color="black",
8     corner_radius=8,
9     border_width=3,
10    border_color="black",
```

```
11     hover_color="DarkOrange2",
12     anchor="center",
13     command=self.view_own,
14 )
```

The corner radius parameter determines how rounded the corners should be. The "command" parameter defines what action should be triggered when the button is pressed: Here, the function view_own() is triggered which makes a callback the the function show_frames() with the target page (ViewOwnSatellites) as its parameter. The "Hover Color" parameter means that the colour of the button should change to "DarkOrange2" when hovering over it [99].

7.6.4 Adding Background Image

Similar to buttons, labels are also a image-compatible widget. The main difference between labels and buttons is that a label is non clickable and this cannot be a catalyst for change.

```
1 # Create a frame for the background image
2 self.image_frame = tk.Frame(self, bg="white")
3 self.image_frame.grid(row=1, column=1, sticky="nsew")
4
5 # Load the background image and place it in the image frame
6 self.background_image = tk.PhotoImage(file="satellite_bg.png")
7 self.background_label = tk.Label(
8     self.image_frame, image=self.background_image, bg="white"
9 )
10 self.background_label.pack(side="right", fill="both", expand=True)
```

First, a dedicated frame for the background image is created and positioned to the right (Lines 2-3). Since there is only one widget inside image frame (namely the background image), pack() was suitable for positioning the image to the right (Line 10). The fill="both" parameter instructs the label to fill in any free space inside the image frame. Setting "expand" equal to True enforces that the background image is allowed to scale up and down automatically if the application window is resized.

7.6.5 Instruction Label

The reason why the instruction label is inside the master frame rather than creating a new frame dedicated to it (similar to Top Frame), is that this label will be used for *every single page* with *no modifications* made to it. Since instruction label shall always be there, it made more sense to position it inside the master frame.

```
1     self.instruction_label = tk.Label(
2         self,
3         text="Hover over a button",
4         font=("Helvetica", 14),
5         bd=1,
```

```

6         relief="sunken",
7         anchor="center",
8     )
9
10    # Instruction label should span the entire width and should be at the bottom
11    self.instruction_label.grid(
12        row=3, column=0, columnspan=2, sticky="ew", padx=0, pady=0
13    )

```

NOTE: Although pinning the instruction label to the second row like shown in Figure 139, setting sticky="ew" leaves gaps at the bottom since it is being stretched vertically. For this reason, the instruction label is pushed down to third row. To make the instruction label provide information once that button is hovered over, .bind(), a function built into Tkinter was utilized. Take for instance the button in the code Listing below:

```

1   self.view_own_button = customtkinter.CTkButton(
2       buttons_frame,
3       text="View own satellites",
4       font=("Helvetica", 20),
5       fg_color="orange",
6       text_color="black",
7       corner_radius=8,
8       border_width=3,
9       border_color="black",
10      hover_color="DarkOrange2",
11      anchor="center",
12      command=self.view_own,
13  )
14
15  self.view_own_button.bind(
16      "<Enter>",
17      lambda event, msg="View own satellites": self.on_enter(event, msg),
18  )
19
20  self.view_own_button.bind(
21      "<Leave>",
22      self.on_leave
23  )

```

Here, two parameter types are accepted; **event and handler**. The event parameter defines what action the button should be binded/respond to. The button widget is then binded to the event <Enter>. In Tkinter, this event corresponds to when "the mouse pointer is moved into a visible part of a widget". In the second button binding, view_own_button linked to the event <Leave> which corresponds to when "the mouse pointer is moved out of a widget". A list of all the event types that Tkinter accepts are included in C.R.A.F.T-GUI's user manual. Handler is a parameter that instructs what action should occur when the event is present. For the first button binding, the <Enter> event triggers a lambda function which makes a function call to self.on_enter() forwards

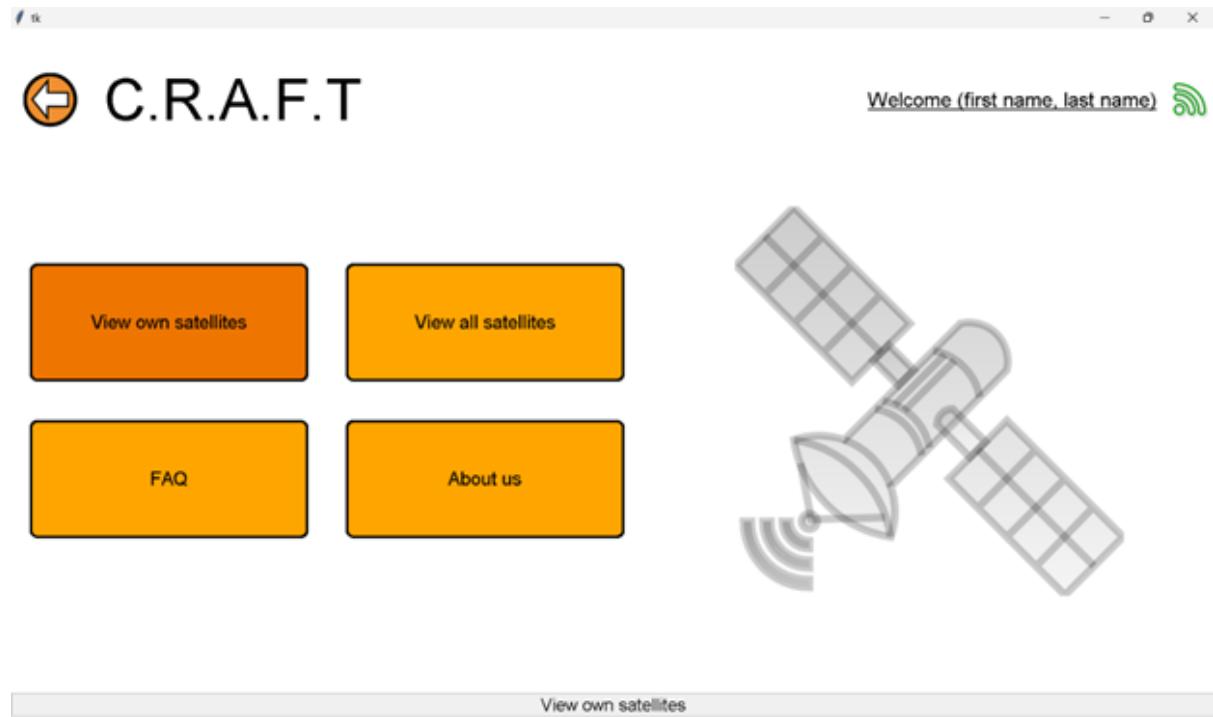
the event and instruction message "msg" as parameters. In Python, lambdas are anonymous functions useful for handling a **singular expression**. They are considered safer since its scope is minimal and only active for a short period of time, compared to regular functions that can be activated from anywhere if one is not careful with their scoping. The functions being called by the handler can be seen in the following Listing:

```
1 def on_enter(self, event, message):
2     self.instruction_label.config(text=message)
3
4 def on_leave(self, event):
5     self.instruction_label.config(text="Hover over a button")
6
7 def go_back(self):
8     return
```

The .config() function allows for Tkinter widgets to be modified post creation. Inside on_enter(), the text inside the instruction label updated to to the message passed by a lambda function. Inside on_leave(), the text is reverted back to default.

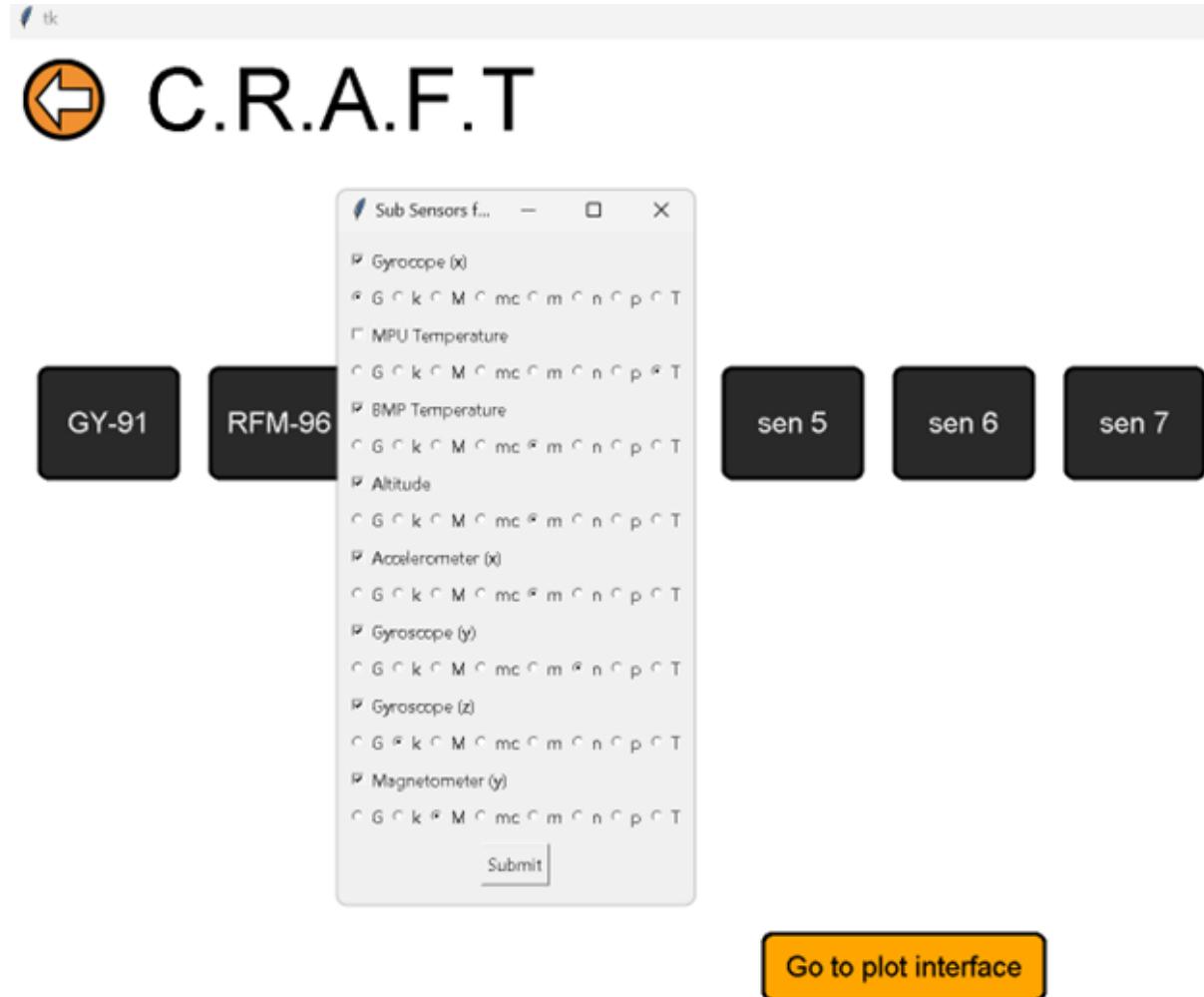
The outputs from Sub-subsections 7.6.3, 7.6.4, and 7.6.5, can be seen in Figure 140

Figure 140: Home Page for Regular Users - End Product



7.6.6 Configuration Page

Figure 141: Page for Configuring Satellite - End Product



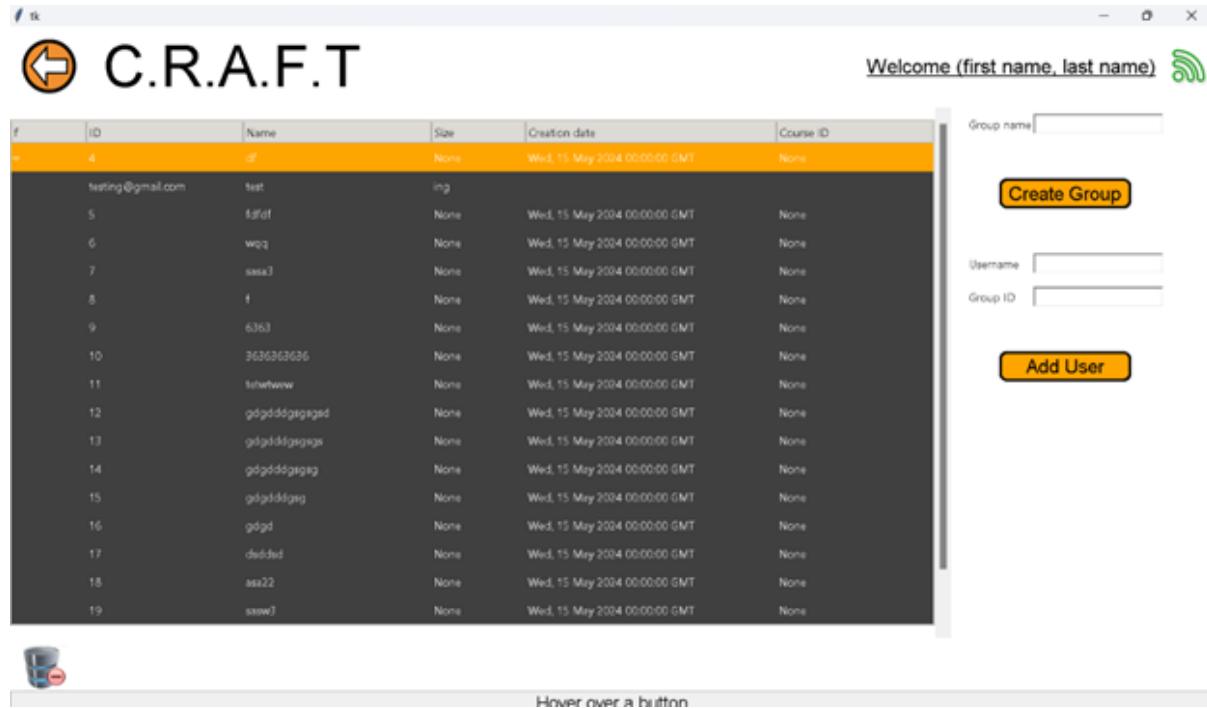
Due to the complexity of the code behind the page for configuration a satellite, a full explanation will be added to our user manual to ASE. Simplifying the explanation, most important things to note about this page is the following:

- Configuration page has integrated automatic scaling in response to resizing of the application window.
- Scroll bar has been implemented: It activates itself automatically once the number of sensor buttons exceed the screen size.
- Once a sensor module button has been pressed, a pop-up windows appears listing of the sub sensors contained inside this module as a check boxes. Beneath each check box, there are radio buttons where users can select their desired prefix to their measurement. Once selection has been confirmed, the selection is added to a list of already selected sensors. This list is then forwarded to C.R.A.F.T-GUI's data

plotting page where the selected sensors appear, ready to be added to the plotting grid.

7.6.7 Administrators - GUI

Figure 142: Admin Page (Overview of Users) - End Product



- Treeview, a Tkinter widget has been used. The advantage of Treeviews is that they can contain sub-entries under a parent row. This has been utilized by C.R.A.F.T-GUI by including child nodes when relevant to the parent node. In Figure 142, each group has its members as sub-entries.
- Orange color to highlight selected item from tree.
- Registration form on the right side. This is true for every administrator page for consistency.

NNG|NNG

7.7 Cleaning up code

One major benefit of employing an object oriented philosophy for C.R.A.F.T-GUI's code is that it can make code modular, reducing the program's lines of code. Another benefit of instantiating base class is **improved safety**. This is because when reusing a class instead of duplicating code explicitly for every object, then the system coder/maintainers *only* have to configure that one base class correctly, and after that it can be reused multiple times. This also makes troubleshooting easier since there are less code to parse through

and a lot of functionality is centralized to one area/class, creating a single potential point of failure rather than multiple.

For instance, the top layout of C.R.A.F.T-GUI is **almost identical** for each page. Another staple for every single page is the inclusion of an instruction label at the page's bottom. (see image). The code to implement both "entities" can be seen below:

Without OOP philosophy implementation, these exact lines would have to be pasted into every single init() function to each page. If a single non unwanted modification is made to the code, but is **not fatal** (i.e., does not cause an error message), that modification could be virtually impossible to identify as there are too many similar initialization functions to parse through. However, if centralized to one class, then this is easier. To achieve this, layouts that are almost guaranteed to be in every page was transferred to a class called BasePage()

Now, other pages/classes just have to pass BasePage as a parameter to inherit said layout (see snippet)

If there are minor differences that you want to establish while still inheriting from BasePage, then that can easily be accomplished by overriding functions and variables from BasePage. For instance, Login page should not have a Go Back button since Login is the start page of C.R.A.F.T-GUI. That can easily be accounted for by destroy redundant widgets using Tkinter's built-in .destroy() function.

By simply implementing BasePage to our frontend code, the number of lines in our frontend code was lowered by approx. 500 lines which is immense. Complexity is reduced, increasing stakeholder satisfaction from ASE and their IT department as reduced long term costs in regard to system maintenance, expansion and troubleshooting. Reduced complexity also simplifies the documentation drastically and makes it more plausible to document our system well, which is one of our primary design and stakeholder requirements. More detailed documentation which more importantly describes a **systematic structure**, to our system, can make C.R.A.F.T-GUI easier for ASE to continue expanding if that would ever be of interest to them (e.g., add more functionality)

DØ|DØ

7.8 Logging Page

7.8.1 General

- **Refresh Function:** This function refreshes the page when it is initiated and when the refresh, delete, or save button is pressed.
- **Canvas with Scrollbar:** The canvas at the bottom contains save instances information on buttons. These buttons are created from save instances retrieved from

the MongoDB database.

- **Delete Button:** Deletes the instance and collection when clicked. It displays the save name, start, stop, and elapsed time. The button turns green when pressed, indicating the selected save. After deletion, the page refreshes to reflect the changes.
- **Save to Disk Button:** When a save button is clicked, a popup allows the user to choose where to save the file. Upon saving, the refresh method is executed.
- **Refresh Button:** Refreshes the page.

7.8.2 User Capabilities

- See save instances of the group the user belongs to.
- Delete save instances, save them to disk, and refresh the page.



Figure 143: Logging page (SaveHistory) from an user perspective

7.8.3 Admin Capabilities

- See all groups in a dropdown menu.
- When logged in, the admin can use all MongoDB functions [100].
- Upon pressing "OK" after selecting a group, all save instances for that group are displayed.



Figure 144: Logging page (SaveHistory) from an admin perspective

7.8.4 SaveHistory Class Overview

The `SaveHistory` class consists of the following components:

- `__init__` Method: Defines the initial state, including database connection (`self.db` and `self.my_collection`), image assets, layout of frames and widgets (buttons, labels, dropdown lists, scrolling canvases), and instance variables. It also calls the `refresh` method initially.
- `refresh` Method: Updates the interface based on available saved instances by fetching records and generating buttons for each instance.
- `get_button_press_command` Method: Returns a wrapper function around `button_press`.
- `button_press` Method: Invoked when a save instance button is pressed. Changes all buttons back to orange and the selected button to green.
- `show_group_dropdown` Function: Displays a dropdown for the admin to choose which MongoDB collection (group) to view.
- `ok_pressed` Method: Validates the dropdown selection.
- `connect_to_mongodb` Method: Connects to a MongoDB server and sets up the database collection.
- `save_mongodb_dialog` Method: Opens a file dialog box for the user to choose where to save a MongoDB database export.
- `export_from_mongo_db` Method: Exports a chosen collection from the MongoDB database as a CSV file.
- `delete_from_mongo_db` Method: Deletes a specified collection from the MongoDB database.

-
- `get_all_saved_instances` Method: Fetches all saved instances from the MongoDB database collection if defined.
 - `fetch_group_names` Method: Sends a GET request to a local server to retrieve all group names.
 - `go_back` Method: Changes the current frame of the controller to `AdminEntry` (for admin) or `ViewOwnSatellites` (for users), depending on the existence of `PlotInterface` in `controller.frames`.

7.9 The Plot Interface

DØ|TL

Reaching user-friendly interfaces for our GUI is the highest goal in our project that meets **Stakeholder Requirement** "3. Easy to assemble and disassemble: (d) The software for the C.R.A.F.T-Sat must be easy to read and use for the end users." 3.1. The aim was to make the design modular and flexible of our interface that underscores the commitment to adaptability and future integration of new modules. Our GUI aims to be useful for many years to come, adhering to a rule of thumb that ensures its longevity and meets the **Stakeholder Requirement** "6. The overall system shall be designed for longevity. This applies to, but is not limited to: (c) Software implementations" 3.1. Below, we present a use case diagram that illustrates the interactions between users and the plot interface.

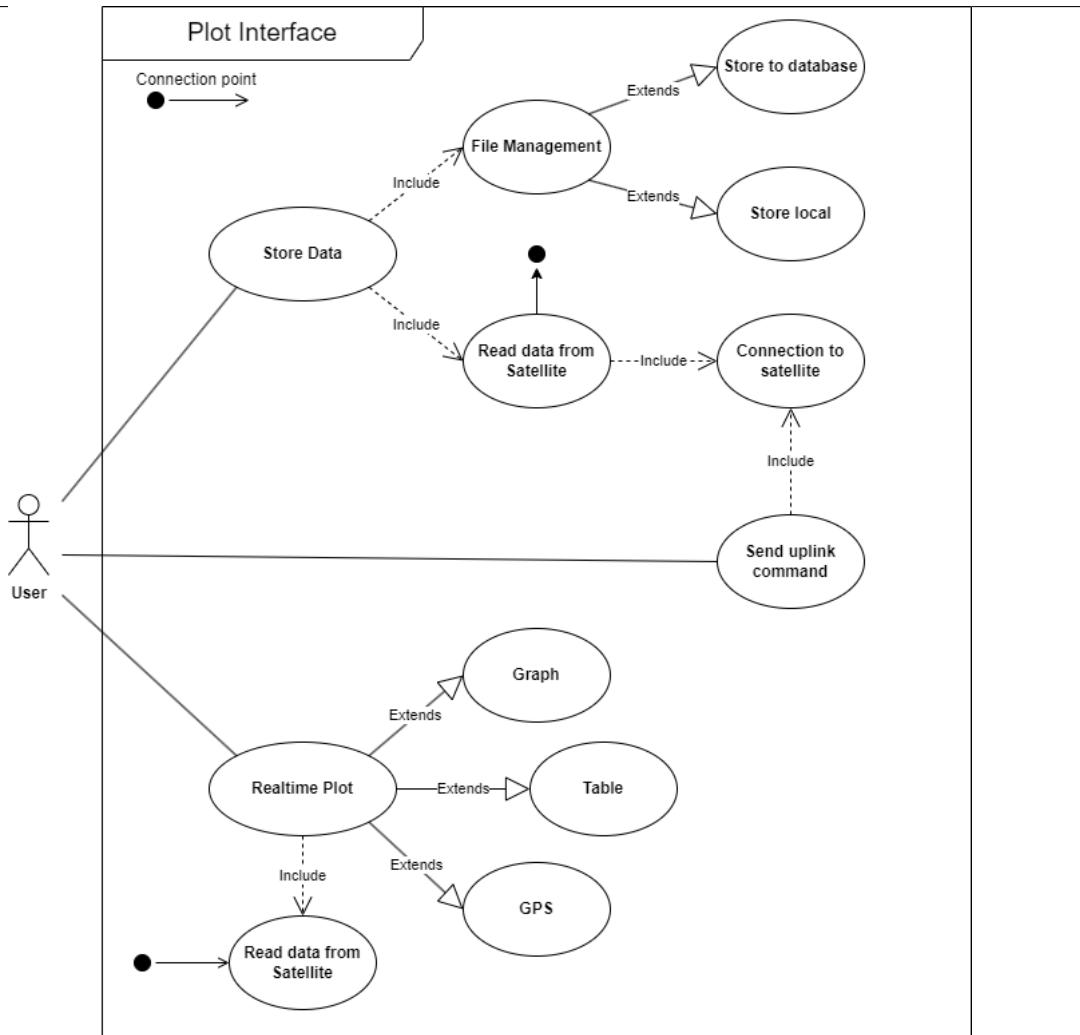


Figure 145: Use Case Diagram for the Plot Interface

The use case diagram, as presented in Figure 145, illustrates the interactions between the users and the plot interface system.

The main use cases represented in the diagram are:

- **Store Data:** A central function where the user stores data retrieved from the satellite. This use case includes "Read data from Satellite" and connects to "File Management."
- **File Management:** Manages data storage options including "Store to database" and "Store local," each extending the fundamental functionality of the file management system.
- **Connection to Satellite:** Encompasses satellite communication functionalities, including "Send uplink command," and is an extension of reading data from the satellite.
- **Realtime Plot:** Allows for the visualization of data in real-time and extends to

generate "Graph," "Table," and integrates "GPS" data to enrich the visual presentation.

Class Diagram Integration

Following the use case diagram, the class diagram provides a deeper insight into the structure of the system, illustrating the relationships between various components critical to the plot interface's functionality. As shown in Figure 146, the class diagram details the modular architecture of the system, facilitating easy updates and maintenance. Key classes such as *FileManagement*, *PlotInterface*, and *SerialPort* form the backbone of the system, supporting extensive data manipulation and interaction capabilities. These classes are designed to be robust and flexible, accommodating future enhancements and ensuring the system's long-term viability in line with our stakeholder requirements.

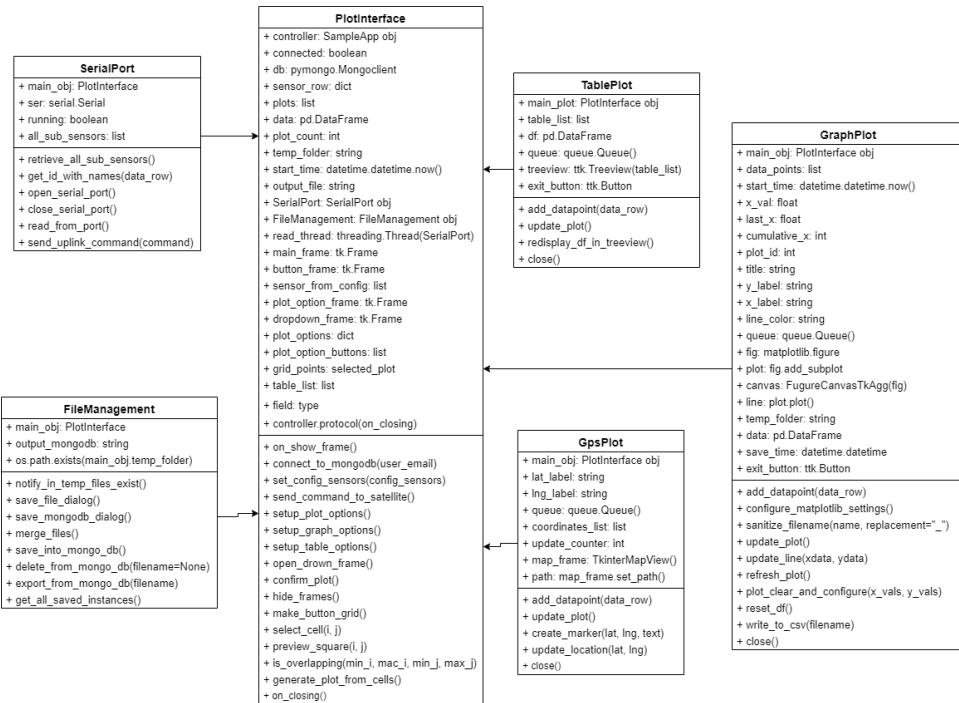


Figure 146: Class Diagram showing the structure of the plot interface code

7.9.1 File Management System in the Plot Interface

The Plot Interface incorporates a file Management system designed to enhance data accessibility and integrity. As depicted in the class diagram (see Figure 146) and the use case diagram (see Figure 145), the system seamlessly integrates functionalities for local and MongoDB data storage, enabling users to manage and analyze their data efficiently.

Local Storage Mechanism The implementation of local data storage revolves around the capacity to save all plot-related data into a .csv file. This process begins with the creation of a temporary folder that dynamically stores all plot data, continuously updated via the 'write_to_csv' function. This function leverages the robust capabilities of the csv [101] and os [102] libraries to manage file writing and directory operations effectively.

The datetime [103] library enriches each entry in the .csv file with precise timestamps, including dates, clock time, and elapsed time, enhancing the informational value of the data for post-runtime analysis.

A user-centric feature within this system is the 'Choose Save Location' button, which activates a popup window allowing users to specify the filename and the desired location for saving the file on their local machine. Once the user confirms the save action, the system merges the data stored in the temporary folder into a single .csv file using the pandas library [104]. This file is structured into separate tables to facilitate easier analysis and interpretation.

A	B	C	D	E	F	G	H	I	J	K	L	M
Start Date	Start Time	End Date	End Time	Total Elapse	Date Time	Clock Time	Elapsed Tim	Time (sec)	ID: 2 Acceler	ID: 3 Acceler	ID: 4 Acceler	ID: 13 Altitud
09.05.2024	15:35:12	09.05.2024	15:37:11	00:02:00	09.05.2024	15:35:13	00:00:02		2 209.0	408.0	190.0	963.0
				09.05.2024	15:35:14	00:00:03		3 233.0	514.0	987.0	830.0	
				09.05.2024	15:35:15	00:00:04		4 410.0	254.0	130.0	416.0	
				09.05.2024	15:35:16	00:00:05		5 612.0	976.0	420.0	565.0	
				09.05.2024	15:35:17	00:00:06		6 138.0	65.0	15.0	345.0	
				09.05.2024	15:35:18	00:00:07		7 293.0	62.0	808.0	794.0	
				09.05.2024	15:35:19	00:00:08		8 578.0	668.0	89.0	576.0	
				09.05.2024	15:35:20	00:00:09		9 799.0	78.0	104.0	887.0	
				09.05.2024	15:35:21	00:00:10		10 221.0	903.0	253.0	372.0	
				09.05.2024	15:35:22	00:00:11		11 537.0	148.0	141.0	327.0	
				09.05.2024	15:35:23	00:00:12		12 497.0	886.0	979.0	64.0	
				09.05.2024	15:35:24	00:00:13		13 978.0	371.0	704.0	647.0	
				09.05.2024	15:35:25	00:00:14		14 834.0	798.0	405.0	484.0	
				09.05.2024	15:35:26	00:00:15		15 806.0	634.0	326.0	710.0	
				09.05.2024	15:35:27	00:00:16		16 695.0	249.0	6.0	685.0	
				09.05.2024	15:35:28	00:00:17		17 900.0	687.0	731.0	486.0	
				09.05.2024	15:35:29	00:00:18		18 19.0	194.0	449.0	480.0	
				09.05.2024	15:35:30	00:00:19		19 855.0	219.0	102.0	244.0	
				09.05.2024	15:35:31	00:00:20		20 824.0	774.0	824.0	4.0	
				09.05.2024	15:35:32	00:00:21		21 361.0	763.0	343.0	960.0	

Figure 147: Content of the merged .csv stored locally

MongoDB Integration Simultaneously with local storage, the system is configured to store data in MongoDB, a document-based database, enhancing data management and retrieval capabilities. This integration allows for real-time data storage in MongoDB Atlas [100], which handles large datasets efficiently and provides robust data retrieval mechanisms. The FileManagement class includes methods such as 'save_into_mongo_db' and 'export_from_mongo_db', facilitating seamless data synchronization between the local system and the cloud database. This dual-storage approach ensures data redundancy and enhances data security, providing a backup mechanism in case of local data loss. For further details on MongoDB integration and logging functionalities, see Subsection C.22 in the Appendix.

A	B	C	D	E	F	G	H	I	J	K	L	M
Filename	Start Date	Start Time	End Date	End Time	Total Elapse	Date Time	Clock Time	Elapsed Tim	Time (sec)	ID: 2 Acceler	ID: 3 Acceler	l
jippi	09.05.2024	15:09:04	09.05.2024	15:19:50	00:10:46	09.05.2024	13:32:21	00:01:54	114	889.0	626.0	9
					09.05.2024	13:32:22	00:01:55	115	445.0	904.0	4	
					09.05.2024	13:32:23	00:01:56	116	343.0	994.0	9	
					09.05.2024	13:32:24	00:01:57	117	822.0	492.0	8	
					09.05.2024	13:32:25	00:01:58	118	876.0	633.0	1	
					09.05.2024	13:32:26	00:01:59	119	252.0	854.0	6	
					09.05.2024	13:32:27	00:02:00	120	146.0	834.0	7	
					09.05.2024	13:32:28	00:02:01	121	805.0	363.0	7	
					09.05.2024	13:32:29	00:02:02	122	286.0	485.0	9	
					09.05.2024	13:32:30	00:02:03	123	758.0	760.0	4	
					09.05.2024	13:32:31	00:02:04	124	270.0	773.0	9	
					09.05.2024	13:32:32	00:02:05	125	410.0	396.0	1	
					09.05.2024	13:32:33	00:02:06	126	815.0	89.0	3	
					09.05.2024	13:32:34	00:02:07	127	339.0	613.0	6	
					09.05.2024	13:32:35	00:02:08	128	334.0	954.0	4	
					09.05.2024	13:32:36	00:02:09	129	44.0	350.0	9	
					09.05.2024	13:32:37	00:02:10	130	7.0	143.0	1	
					09.05.2024	13:32:38	00:02:11	131	282.0	298.0	1	
					09.05.2024	13:32:39	00:02:12	132	497.0	196.0	5	
					09.05.2024	13:32:40	00:02:13	133	267.0	676.0	1	

Figure 148: File with the name "jippi" export from MongoDB, and stored locally on the computer as a .csv file

Data Consistency and Integrity To address potential issues with NaN values and inconsistencies in data types during the merging process, the system pre-constructs a DataFrame based on the maximum number of rows and columns needed, initializing placeholders accordingly. This meticulous approach ensures that all column headers are unique and that data from different plots are aligned correctly without unintended data type conversions or the introduction of NaN values.

This sophisticated File Management system, as illustrated in the use case and class diagrams, provides the Plot Interface with a robust framework for managing plot data, ensuring high usability and accessibility while maintaining data integrity and security.

7.9.2 Serial Port Integration

The Plot Interface successfully integrates serial port functionality to effectively receive and process data transmitted over the radio from a ground station, specifically using a Teensy microcontroller. This integration leverages the Python serial [105] library to establish a robust connection with the ground station through a designated COM-Port, ensuring reliable data acquisition.

Storing and Handling Sensor Data Further enhancing its capabilities, the system ensures that all sensor data read from the serial port is meticulously captured and stored, whether or not it is immediately displayed in plots. This functionality is described in depth in Subsection C.15 in the Appendix. The data, comprising various sensor readings transmitted as a continuous string, requires precise parsing to separate sensor labels/-names from their values. The parsed data is then organized into a structured format, facilitating easy access and manipulation for potential analysis and reporting.

Plug and Play Functionality The Plot Interface incorporates plug-and-play capabilities to enhance user experience and meet the dynamic needs outlined in the Stakeholder Requirements 3.1. This feature allows the system to automatically detect and configure the connection with various ground stations' COM ports without user intervention C.21. This automated detection and configuration significantly streamline the setup process for users, ensuring that the system adheres to the "Easy to operate, assemble and disassemble" requirement stipulated by our stakeholders. This approach not only simplifies the initial setup but also reduces potential errors during system deployment, emphasizing ease of use and reliability.

Notification and Data Integrity A notable implementation within this framework is the proactive alert system that enhances data integrity and user convenience. This system alerts the user upon program startup if temporary files exist, suggesting actions to either save or delete the data. This feature is crucial for recovering data that was not properly saved due to unexpected shutdowns or crashes, thereby safeguarding valuable data against potential loss.

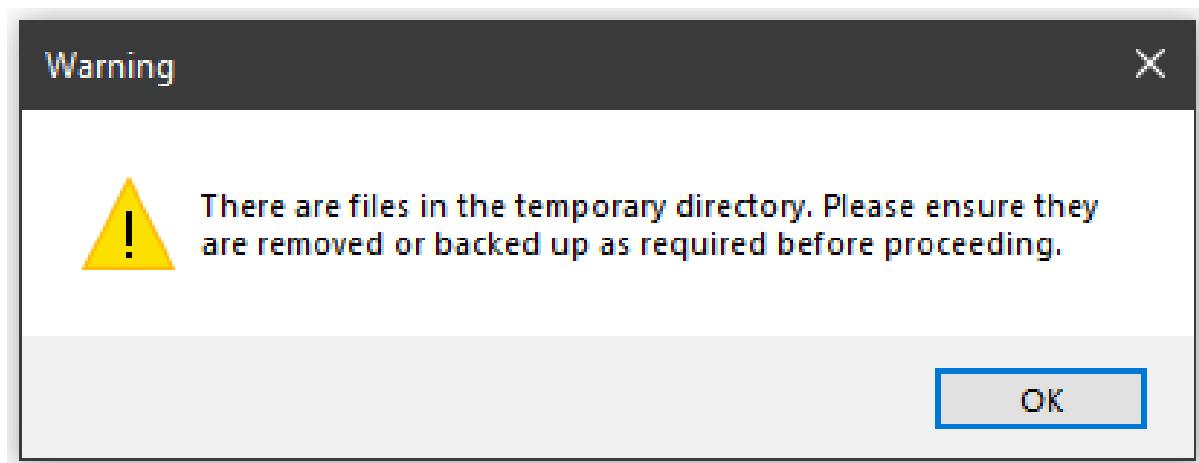


Figure 149: Notification alert if temporary files exist

This section, along with the detailed handling of sensor data storage and parsing, ensures that the Plot Interface not only meets but exceeds expectations in terms of data handling and operational integrity, as further elaborated in Subsection C.15 in the Appendix.

7.9.3 Uplink to Satellite

One of the key enhancements in the Plot Interface is the integration of functionality to send uplink commands to the satellite via the serial port. This feature significantly improves the interface's user interaction and expands its capabilities to include direct communication with satellites, enabling real-time command execution and control.

Command Transmission Functionality The uplink command functionality allows users to input commands directly from the interface, which are then transmitted to the satellite. The format for these commands is standardized to ensure consistency and

ease of use. Commands such as "BEEP, 300;" are sent to perform specific functions on the satellite, like activating a buzzer at a specified frequency. This command structure is detailed in the C.R.A.F.T-Sat Library API documentation A.17, which outlines how commands are structured and processed.

Reference to Command Formats The command formats used in the uplink functionality adhere to the definitions set out in the C.R.A.F.T-Sat Library APIA.17. This ensures that all interactions with the satellite through these commands are predictable and effective. The API documentation specifies various commands under the "Remote Control" section, where each command and its expected parameters are clearly defined, ensuring users have a clear guide on how to interact with the satellite system efficiently.

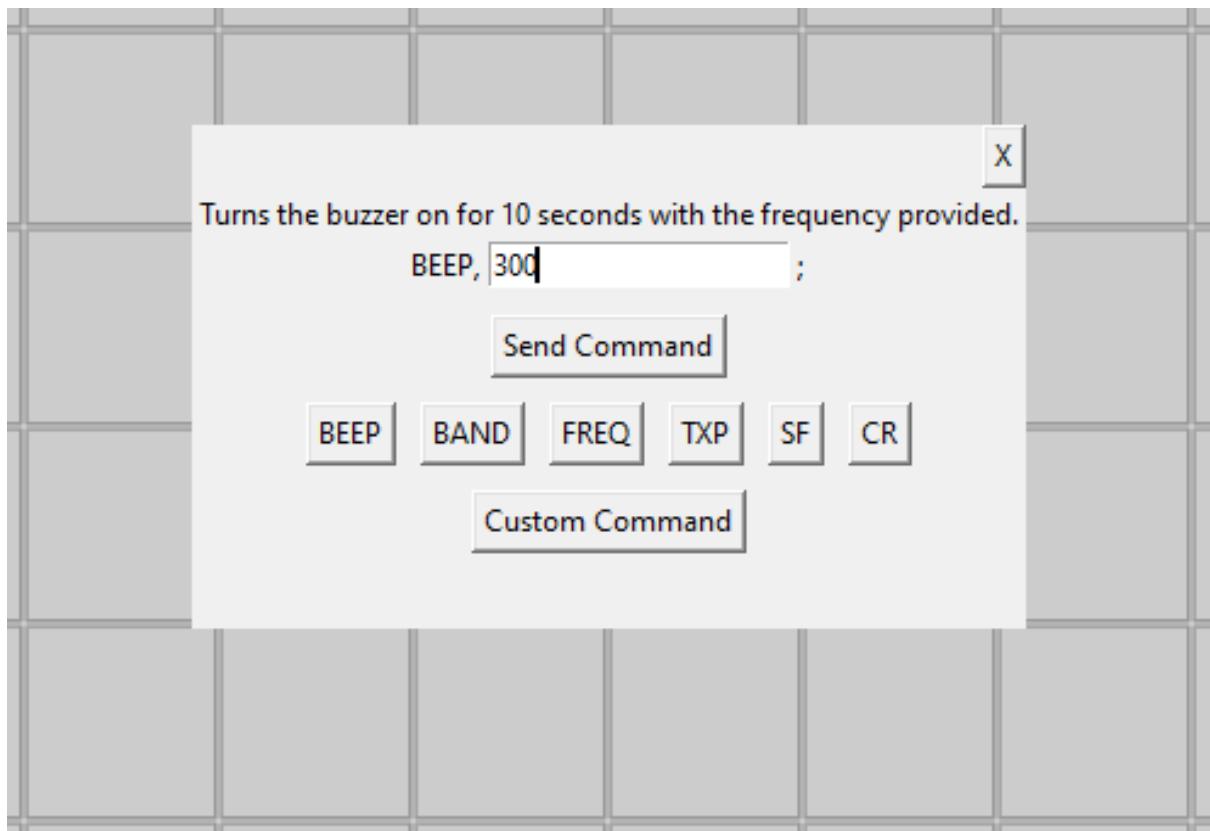


Figure 150: Uplink field to send commands to the satellite. Commands are formatted as "BEEP, 300;" for operational clarity.

The addition of this uplink command functionality not only enhances the Plot Interface's usability but also aligns with the goals of increasing the system's interactivity and user engagement. By integrating direct satellite communication capabilities, the interface provides users with a powerful tool for satellite operations, reflecting a comprehensive approach to user interface design and functionality.

7.9.4 Graph-, Table-, and GPS-Plot in Real-Time

The Plot Interface offers a comprehensive suite of visualization options tailored to a variety of data analysis needs, integrating Graph-, Table-, and GPS-plot capabilities. Each option is designed to enhance the user experience by providing specific insights suitable for different types of data and analysis requirements, aligning with the system's objectives to provide an accessible educational tool.

Graph Plot The Graph Plot functionality within the PlotInterface provides real-time visualization of streaming data, crucial for monitoring dynamic parameters and capturing temporal trends necessary for immediate data analysis.

Table Plot The Table Plot organizes data into a structured tabular format, which is crucial for detailed statistical analysis, enabling users to efficiently sort, compare, and analyze datasets. This helps in uncovering underlying patterns in the data.

GPS Plot The GPS Plot supports geographical data visualization, vital for projects that require spatial tracking or analysis. This plot type utilizes GPS data to render positional information on maps, providing enhanced analytical utility for geographical and movement-based studies.

The implementation of these plotting options exemplifies the Plot Interface's commitment to adaptability and educational value, ensuring it remains a relevant and powerful tool for data analysis in various educational and experimental settings. The interface design is intuitive, aligning with the stakeholder requirement for an easy-to-use system [3.1], enhancing user engagement and ensuring operational efficiency.

Plot on Button-Grid Inspired by the experience of selecting seats online for a movie theater, similar to functionality found on platforms like Nordisk Film Kino [[106]], a user-friendly interface was developed for selecting and marking areas for plots to enhance user experience and modularity. Users can select and visualize plot areas by clicking and moving the mouse within a button-grid, simplifying the interaction model and enhancing the structured organization of the interface.

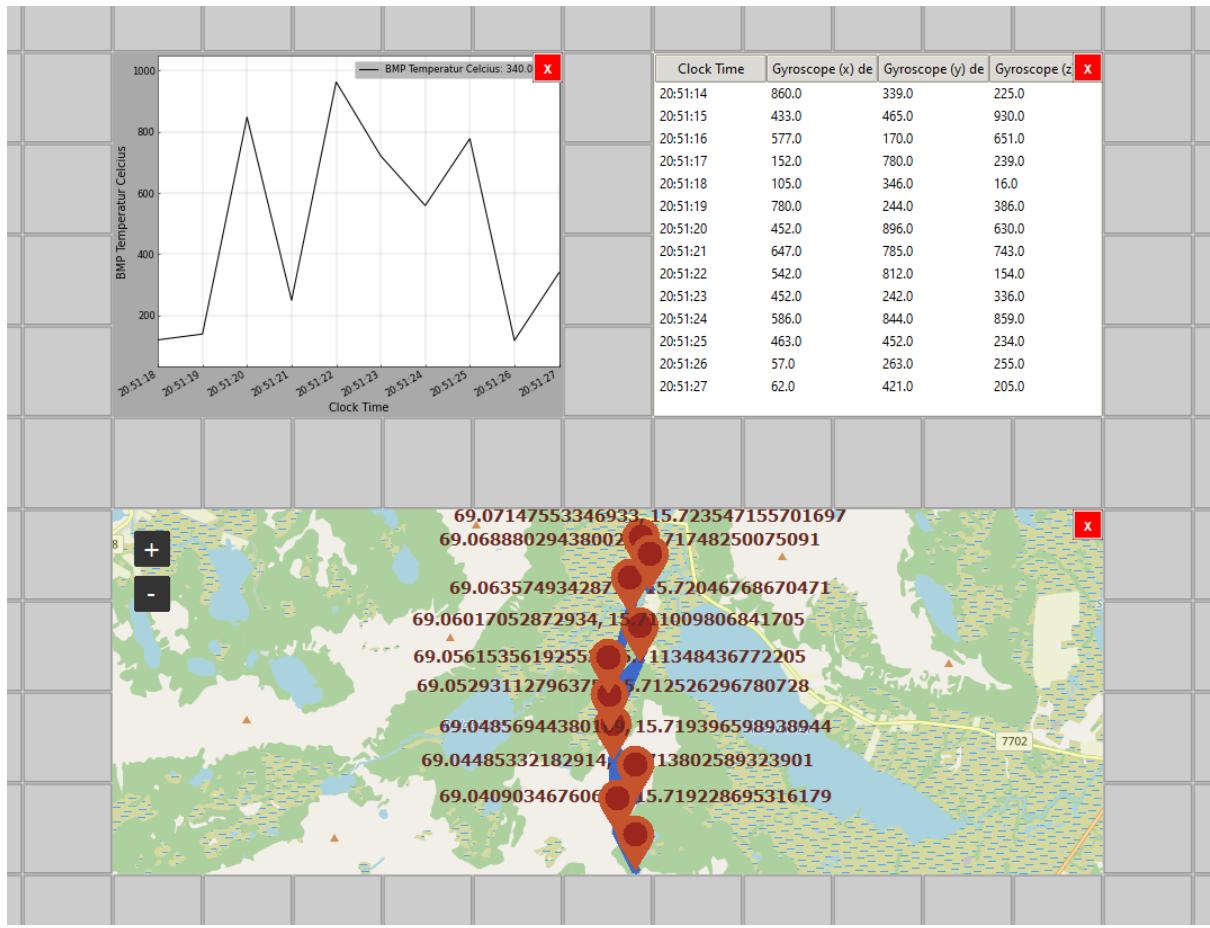


Figure 151: User interface displaying options for Graph, Table, and GPS plotting in real-time.

This solution replaces traditional drag-and-drop features with a more structured and straightforward interface, maintaining advanced modularity and likely to be enhanced in future iterations. Such developments ensure the Plot Interface remains aligned with evolving user needs and technological advancements, fostering a continuously improving user experience.

Note: For a detailed account of the developmental process behind the plot interface, including step-by-step implementation details and challenges encountered, please refer to Section C in the Appendix.

DØ|TL

7.10 Deploying a Flask Application to Google Cloud

Here are details described for the process of deploying a Flask application to Google Cloud and configuring it to run concurrently with a Tkinter GUI application. The steps involved in this process include containerizing the Flask application, configuring Google Cloud Build, granting necessary permissions, modularizing the application, and enabling concurrent execution using threads.

The deployment process was divided into several key tasks, as described below.

7.10.1 Dockerizing the Flask Application

To deploy the Flask application on Google Cloud Run, the application was first containerized using Docker. The Dockerfile used for this purpose is shown below:

```
1 FROM python:3.9-slim
2
3 WORKDIR /app
4
5 COPY . /app
6
7 RUN pip install --no-cache-dir -r requirements.txt
8
9 EXPOSE 8080
10
11 ENV PORT 8080
12
13 CMD ["python", "backend.py"]
```

7.10.2 Configuring cloudbuild.yaml

The `cloudbuild.yaml` file was created to define the steps for Google Cloud Build to build [107], push, and deploy the application to Google Cloud Run [108]. The configuration is as follows:

```
1 steps:
2   - name: 'gcr.io/cloud-builders/docker'
3     args: ['build', '-t', 'gcr.io/$PROJECT_ID/flask-app', '.']
4   - name: 'gcr.io/cloud-builders/docker'
5     args: ['push', 'gcr.io/$PROJECT_ID/flask-app']
6   - name: 'gcr.io/google.com/cloudsdktool/cloud-sdk'
7     entrypoint: 'gcloud'
8     args: ['run', 'deploy', 'flask-app', '--image', 'gcr.io/$PROJECT_ID/flask-app',
9            '--platform', 'managed', '--region', 'europe-west1', '--allow-unauthenticated']
10 images:
11   - 'gcr.io/$PROJECT_ID/flask-app'
12 logsBucket: gs://your-bucket-name
```

7.10.3 Granting Necessary Permissions

The Cloud Build service account required several roles to deploy the application to Cloud Run. The following roles were granted:

- `roles/run.admin`
- `roles/iam.serviceAccountUser`

-
- roles/cloudbuild.builds.builder

The permissions were granted using the following commands:

```
1 gcloud projects add-iam-policy-binding [PROJECT_ID] \
2   --member=serviceAccount:[PROJECT_NUMBER]@cloudbuild.gserviceaccount.com \
3   --role=roles/run.admin
4
5 gcloud projects add-iam-policy-binding [PROJECT_ID] \
6   --member=serviceAccount:[PROJECT_NUMBER]@cloudbuild.gserviceaccount.com \
7   --role=roles/iam.serviceAccountUser
8
9 gcloud projects add-iam-policy-binding [PROJECT_ID] \
10  --member=serviceAccount:[PROJECT_NUMBER]@cloudbuild.gserviceaccount.com \
11  --role=roles/cloudbuild.builds.builder
```

7.10.4 Modularizing the Flask Application

The Flask application was refactored to use a factory function, allowing it to be imported and used in different scripts. This modularization facilitated concurrent execution with a Tkinter application [109].

7.10.5 app_backend.py

The Flask application was defined in 'app.py' as follows:

```
1 import os
2 from flask import Flask, request, jsonify
3 import mysql.connector
4 from flask_bcrypt import Bcrypt
5 import pyotp
6 from datetime import datetime
7
8 def create_app():
9     app = Flask(__name__)
10    bcrypt = Bcrypt(app)
11
12    @app.route("/login_otp", methods=["POST"])
13    def login_otp():
14        data = request.json
15        email = data["email"]
16        otp_candidate = data["otp"]
17
18        result = logintodb_otp(email, otp_candidate)
19        return jsonify(result)
20
21    def logintodb_otp(email, otp_candidate):
22
23        # Other routes and functions here...
```

```
24
25     return app
```

7.10.6 backend.py

The `backend.py` file was updated to run the Flask application [110]:

```
1 from app import create_app
2
3 app = create_app()
4
5 if __name__ == "__main__":
6     port = int(os.environ.get("PORT", 8080))
7     app.run(host="0.0.0.0", port=port)
```

7.11 Running Flask and Tkinter Concurrently

To run the Flask application concurrently with a Tkinter GUI application, threading [111] was used. The concurrent execution was managed in `main.py`.

7.11.1 gui_poc.py

```
1 import threading
2 import os
3 from app import create_app
4 import tkinter as tk
5 from tkinter import ttk
6
7 def run_flask():
8     app = create_app()
9     app.run(host='0.0.0.0', port=8080)
10
11
12 def flask_process_shutdown(pid):
13     main_module = importlib.import_module("__main__")
14     sample_app = getattr(main_module, "app", None)
15
16     if sample_app is not None and PlotInterface in sample_app.frames:
17         sample_app.frames[PlotInterface].on_closing()
18
19     os.kill(pid, signal.SIGINT) # or signal.SIGTERM
20
21 def run_tkinter():
22     global app # Now we're setting app as global, not sample_app
23     app = SampleApp()
24     pid = os.getpid()
25     app.protocol("WM_DELETE_WINDOW", lambda:
```

```
26     # override closing
27     (flask_process_shutdown(pid), app.quit()))
28     width = app.winfo_screenwidth()
29     height = app.winfo_screenheight()
30     app.geometry(f"{width}x{height}+0+0")
31     app.mainloop()
32
33 if __name__ == "__main__":
34     flask_thread = Process(target=run_flask)
35     tkinter_thread = threading.Thread(target=run_tkinter)
36
37     flask_thread.start()
38     tkinter_thread.start()
```

7.11.2 Conclusion

The deployment process involved containerizing the Flask application, configuring Google Cloud Build, and setting up concurrent execution with Tkinter using threading. The Flask application was successfully deployed to Google Cloud Run, and both the web and GUI applications were run concurrently.

7.12 The final C.R.A.F.T-Sat frame and release mechanism

The C.R.A.F.T-Sat is designed to be 3D printed with a simple FDM printer controlled by a user. The final prototype, fig 152 requires specified angles, supports and non-supports when the parts used to assemble the C.R.A.F.T-Sat are printed. Therefore, a simple printing manual, appendix M.3 is made. The manual describes the method to change sizes due to the print results and ends for every part file with a description of the recommended print direction, support, and other procedures to gain a product equal to our prototype.

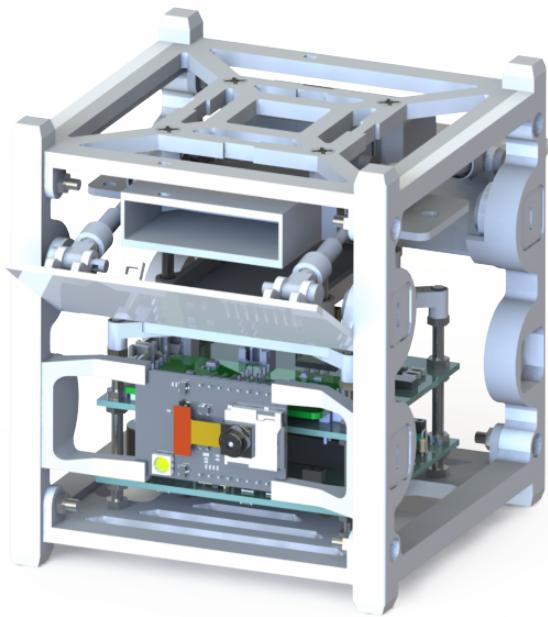


Figure 152: Final prototype (without antennas, springs, and cables)

Since the prototype is designed to be manufactured using a simple FDM printer, the probability of getting sizes on the prototype equal to the CAD measurement isn't high (we don't have a specific value but by 3D printing, through the project, we have never got a specific equal decimal value as the value from the CAD model).

In SW the product is divided into one main assembly and multiple sub-assemblies. The main assembly, *Main_asm* 152 represents the final prototype and has a *ZY_wall* design (fig 237) referred to as the frame, frame design contains simplified cuts used for a connection method between the *ZY_walls & Sealings* (fig - *Sealing_PositiveY* 266, fig *Sealing_NegativeY* 265), and *ZY_walls & Camera_Holder* (fig 284) or other boards/-components mounted with an equal method as the *Camera_Holder*. The design chosen is described as a reusable, and easy re-producible product. The reason we decided on those descriptions is because the 3D printed parts designed and used have a design related to

not too difficult to 3D print, and also easy to quickly print parts if fracture happens on different components. We decided to have the opportunity to easily re-print as a solution due to the risk of suddenly destroying 3D-printed parts during testing and use.

To connect the *sealings* with the *ZY_wall* the *HingeLockPart* (fig 261) is designed to be placed between the cuts equal it's shaft design with the head inside the C.R.A.F.T-Sat. To lock the *sealings*, turn the *HingeLockPart* counterclockwise or clockwise with an angle of 90° and see from outside the C.R.A.F.T-Sat on the *ZY_wall* that the shaft is prevented to fall back out. The *HingeLockPart* is only designed to connect and not directly rotate during use, so, due to friction, it's easy for the *HingeLockPart* to turn and fall out without something that prevents it to turn. We have therefore designed *Lock1* (fig 256) with a cut for the *HingeLockPart* as a cover and stopper for rotation of the *HingeLockPart*. The *Camera_Holder* has an equal design as the *HingeLockPart* and to prevent it from rotation use *Lock1*.

Connected with the main assembly is the sub-assembly, *Sub_MainAsm* (fig 153) and *Spring_asm*. The subassembly's is designed to be easily assembled and disassembled with 3D printed parts and M3 bolts and locking nuts. The release mechanism is mainly referred to the *Sub_MainAsm* which is divided into the upper part and a lower part. The upper part is mainly the area where the battery is placed and where the antennas are mounted. The lower part is where the electrical circuit boards with teensy and shield are mounted. A further description of how to mount together this product is found in appendix M.

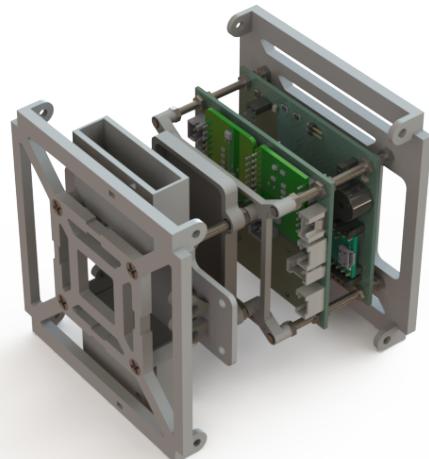


Figure 153: The final prototype - *Sub_MainAsm* | STB

7.13 Finite Element Analysis (FEA) on the final design

The FEA uses a Finite Element Method (FEM) which is an effective method for verifying a CAD with a lower cost compared to prototype testing related to a requirement of multiple tests if the test object has requirements such as our tensile test staff with at least 5 staffs per test. Therefore, we have decided to use the FEA to verify the structure related to the frame and antenna release mechanism before we approve further realistic testing with a 3D-printed prototype. It's important to understand that the test we are providing with the FEA is not realistically equal to the physical final product testing due to the amount of memory/data required to design/produce a FEA of the C.R.A.F.T-Sat and the lack of specific material data on the electronic devices designed by the team and imported from a CAD software.

Note: FEM is the analysing method of FEA.

We have therefore provided a simplified analysis with the fixture, loads, and other necessary data required for a simplified analysis which has enough data to give a result with the possibility to be changed when other material parts are included in the analysis. We know that a flow and a thermal vacuum analysis could give good data regarding the environment when the altitude increases and the temperature, and pressure changes. Also, the air could affect its position, structure, or electronic devices. Therefore, a flow analysis should be related to how the air will flow around and between the C.R.A.F.T-Sat. This could provide information about how the C.R.A.F.T-Sat will change its position during launch. The Solid Works Simulation (SWS) will instead be based on mostly the *main_asm* from SW and have a thermal analysis, static analysis (without thermal data due to the required time for the analysis and a computer with not enough memory), frequency analysis regarding the natural frequency, harmonic dynamic analysis, and a drop test.

Note: We decided to mostly use the *main_asm* to gain mostly equal data about the realistic use of the C.R.A.F.T-Sat and included simulations of part-files at the point we decided it as necessary and had enough time to do so. Also, the material we have decided to be used on the chosen analysis is for the 3D printing structure; polymer material PETG and the bolts/nuts AISI 316 Stainless Steel Bar (fig with material data taken from SW 154).

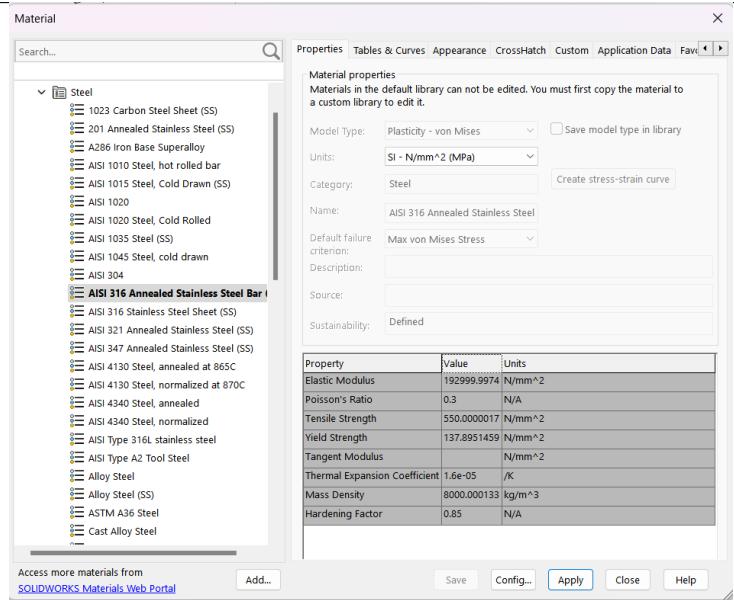


Figure 154: SW: AISI 316 Stainless Steel Bar

STB| TL

7.13.1 FEA: Thermal analysis

We decided to provide a thermal analysis due to the environmental change when thermal exposure is on the C.R.A.F.T-Sat changes when altitude increases (it's important to note that the temperature will be constant in the low stratosphere region). Since the analysis required a lot of data, we could only make one specific thermal analysis. We decided that should represent when the C.R.A.F.T-Sat will be exposed for the lowest temperature, -85°C given from ASE. The report for the thermal analysis is included in appendix V.2.

Note: it would be better if temperature -40°C , -20°C and 20°C also were used in the analysis, and it could be included on part files, but due to the restricted time, it was only considered and not done. It's also important to note that the chosen interactions in the FEA represent the interactions necessary when the FEA was made and tested, and new interactions have been discovered later but not included in this analysis. This does not give the correct answer in the analysis compared with reality, but the answer is correct due to what is chosen as interactions.

The thermal analysis on the C.R.A.F.T-Sat represents a thermal load exposure of a temperature equal $-85^{\circ}\text{C} = 233.15$ Kelvin on the outside surfaces and a heat flux from the battery on the surfaces the battery is touching. The heat flux values are manually calculated, section 6.6 and placed on the areas where the different calculated value is represented. Due to the result from the calculation of the heat flux it would be more reasonable to use the average value, but when we decided to take the analysis it was considered to use the values representing the different areas.

It's also important to inform you that the calculated values don't include the effect a

temperature cover on the battery could give on the values for the heat flux. The reason why the heat flux is not reduced is to provide a high-risk possibility.

The included parts of the assembly used in the analysis are represented as the model. According to the analysis, the highest temperature on the model analyzed with the effect from outside temperature and the chosen heat flux is equal 54.65°C . The area with this temperature is on the surfaces with the polymer material PETG. According to the material data for the polymer material PETG, appendix O.2.4 the lowest glass transition temperature, 79.0°C is higher than the highest temperature on the polymer material. Therefore, we will still be in the temperature area with glass transition temperature for all the PETG components.

Thermal conclusion:

The polymer is therefore usable in our case, but with a big risk in the areas where we have loads that could provide a fracture, or changes of the structure due to stiffness. Such areas are where bolts/nuts are placed, and the areas with the springs that bend the *sealing* and could provide a sudden fracture due to less ductility.

Note: We have not given or have found the glass transition temperature for the bolts/nuts that are included in the analysis (material: *AISI 316 Stainless Steel Sheet*). Therefore, we have decided it there could be two possibilities related to the temperature effect on those components.

If it's in the glass-transition temperature, it will affect the material equal the PETG, but if it's not in the glass-transition temperature, the nuts/bolts will be more ductile and have a larger possibility of strain but with a lower value with stress (compared if it was in the glass-transition temperature).

STB|TL

7.13.2 FEA: Static analysis

First, the static analysis should include thermal data as a load, but due to the required memory for including the thermal analysis and providing many thermal analyses, we haven't inputted the thermal data in the static simulation. The reports for the static analysis are included in appendix V.1.

In our product, the stress affecting the design comes from the spring load, clamping force on the surfaces between the nuts and bolts, and the internal loads between surfaces due to gravitation. We have decided to not make the assembly advanced, therefore, the amount of clamp force affecting the structure where bolts & nuts are mounted will be neglected in the analysis. The analysis will instead include bolts and nuts without threads and have bonded surfaces.

Sealing:

The sealing is one of our biggest interests regarding the spring force. The sealing has three static stages (not including thermal analysis) that should be considered for testing:

1. The *sealing* is closed and the "stick" providing the sealing to not open (fixtures) when

the spring force (neglected force antenna provides the sealing when it is closed due to it's more effective to define the force from the springs and its risks) is at its lowest value.

2. The *sealing* is closed and the burn-wire (with an equal amount of spring force in the opposite direction) is keeping the sealing closed (fixture) when the spring force (neglected force antenna provides on the sealing when it is closed due to it's more effective to define the force from the springs and its risks) is at its lowest value.
3. The *sealing* is open, resulting when the maximum spring force pushing the sealing (load) when it's held back by a new contact surface (the surface is not touching the sealing when it's closed) on the *ZY_wall* (fixture).

We did provide a simulation for steps (1) (2) and (3), but due to time, step (3) was the only *sealing* analysis report regarding static simulation which was finished.

Note: Pressure is important to consider during the static analysis, due to its effect on the electronic devices and since our product is 3D printed without 100% infill. This results in a possibility of pressure differences inside and outside the walls of the print and could provide material interactions and possible catastrophic failures. However, due to time, a simulation regarding the pressure effect was not made, but a realistic vacuum test was provided instead.

According to the simulation on the *sealing* at static stage (3) we gained a result where the maximum vonMises stress was at 69.82 MPa, and the area with high values is at the area where the part is designed to be used for mounting the sealing with the *ZY_wall*. The mesh we gained around those areas was good enough according to mesh control with Aspect Ratio and Jacobian Ratio. The highest displacement value is 2.09 mm and shows a curve where the *Spring_asm* is mounted. Due to the high-stress value, the factor of safety was simulated to represent 0.17. This is too small and will result in a catastrophic failure. Therefore, a simulation regarding the *Main_asm* will be important to also provide to see if the result is equally matched.

Main_asm:

The fixture on the *Main_asm* was decided to represent a realistic use of C.R.A.F.T-Sat when it's connected to the balloon at a specific altitude and will experience thermal exposure as a load value in a static analysis. It's therefore correct to first make a thermal analysis and include the thermal data as a load in the static analysis. In our case, we decided to make a thermal analysis regarding the heat flux from the battery at a specific steady-state temperature. But since we can't include the thermal data, we will only use the load exposing the C.R.A.F.T-Sat at the specific height. According to the static simulation of the *Main_asm* with the antenna released (but not included), our product's only force is from the springs inside the *Spring_asm* with the value 16.86 N. The model (in this case; the part files included in this simulation) is exposed to a maximum displacement value equal to 0.142 mm. The highest vonMises stress is 7.40 MPa, which is lower than the material data; yield strength of the PETG material. It's important to note that the result would have changed if the thermal analysis at temperature -85°C was included.

Due to material data changing with the temperature, a change of polymer temperature provides a change in the stiffness, the temperature in the areas with almost equal value as the glass transition temperature. We can't confirm without further evidence, and we don't have enough data regarding the other temperatures affecting the material data during use in the different environments.

Static conclusion:

Since both analysis uses the same state of open sealing, we can compare the results. According to the *Main_asm* analysis, our *sealing* won't bend with an equal displacement value as in the *Sealing* analysis. The biggest difference is also regarding the result; the factor of safety. According to the data from the *Main_asm* we will have a value higher than the accepted value 1, but the *sealing* analysis gives a value much lower than 1, resulting in a non-useable product with those circumstances and material data. Since both of them are trusted, a realistic test will provide a safer conclusion about our design regarding the spring force.

STB|TL

7.13.3 FEA: Natural frequency analysis

When the C.R.A.F.T-Sat is used, frequency due to a vibration test, or other devices providing vibration could give the C.R.A.F.T-Sat a catastrophic fracture. We have therefore decided that a FEA regarding the natural frequency is important. The reports for the natural frequency analysis are included in appendix V.3.

By using the engineer octave rule, a mechanical engineer can simply decide if the natural frequency of the design is in a risk zone of resonance from a vibration applied to the model. The octave rule says you are unsafe for resonance if the natural frequency, f_n from the model is inside the interval $\{\frac{f_n}{2}, 2f_n\}$.

The frequency we used with the simplified vibration test will be 30 – 60 Hz. Therefore, by using the octave rule, the natural frequency's values has to be outside of the interval

$$\left\{ \frac{30}{2}, 2 \cdot 60 \right\} = \{15, 120\} \text{ Hz}$$

We decided to use the *main_asm* in the analysis to provide a result regarding the final design, but a simulation regarding the *XY_wall_Positive_Z* has also been reported and included in the appendix.

Note: It's important to understand that our result is related to the components used, if an assembly regarding every component was used, the result would change due to changes in mass and stiffness.

We also decided to provide a harmonic dynamic analysis regarding forced vibration, but due to it being less sufficient in gaining a result regarding the data we need, We decided to include a reference to the report in this section, appendix V.4. **Frequency conclusion:**

According to the result we have gained from the analysis our highest calculated frequency value (60 Hz) is lower than the values from the frequency simulation. Therefore, we can conclude that our product with the assembly used in the FEA will not resonate.

Note: when FEA was made, it represents when it's mounted to the balloon or a device with an equal mounting method.

We can not give a direct conclusion about the frequency values if all parts in the assembly were included, but we can give an assumption that the frequency value would mostly increase due to the mass requirement providing a minimal increment in the total mass from what's already used, and the circuit boards removed would provide a stiffer area inside the frame regarding the bolts used for mounting.

STB|TL

7.13.4 FEA: Drop test

The drop test is chosen as a dynamic simulation, giving possibilities to gain data from a non-equilibrium state. The report for the drop test is included in appendix V.5.

A drop test is also referred to as the possibility to gain data at a state where a sudden opposite force occurs. In our case, the drop test of the *Main_asm* is decided to be important to gain the necessary data regarding the possibility of the balloon suddenly cracking and the C.R.A.F.T-Sat falling straight down to the ground. We don't include temperature effects when it suddenly changes and could cause scenarios such as deformation/melting during fall, and other scenarios which as changing design and material hitting the ground.

In the realistic scenario, a possibility such as the prototype won't fall straight down because of possible changes in the structure such as wind or center of gravity, making the prototype spin or not fall with the chosen surface against the ground. But, since we don't have enough data regarding a realistic analysis and enough information about all the material used on the circuit boards, we decided to simplify it by assuming it falls straight down, non-electronic devices are necessary for the analysis and the solution time after impact is 125.4 microseconds. It's also important to note that the realistic critical scenario is in the case where it falls straight down from the altitude of 26,000 meters above sea level. The height required a lot of data and a conclusion of using 10 meters was made.

The part we decided as important after the impact is *XY negative Z* because it will hit the surface first. It's important to note the chosen stiffens are rigid, and the other *setup information* is shown in the drop test of *Main_asm*. Since the assembly file is large (large data capacity) the analysis had to be simplified with a non-used mesh quality plot for Aspect ratio and Jacobian ratio (but chosen values for mesh are equal in all *Main_asm* analysis).

Note: The assembly doesn't have the nuts on the bolts, which also results in non-stress on the simplified bolts used in the analysis gained from a realistic use of the locking nuts. Also, the bolts don't have threads in the analysis and the results won't include data about a real bolt and nut with threads.

The vonMises stress result in the last time-lap is enormous and is mostly shown on the illustrated bolts between the parts antenna board and the XY Negative Z. Since the material used for the bolts isn't polymer the yield strength value isn't equal the PETG yield strength value. In our case, the yield strength value of the bolts and nuts is higher than the calculated vonMises stress on the bolts, therefore, the bolts won't stay deformed if no other states on the bolts have occurred. The frame will gain cracks and be non-usable after this scenario. The reason is because of the stress in the area where its polymer material is much higher than the UTS.

By reading the data gained from the displacement, and knowing that the falling direction is the negative z-axis in the displacement fig, the areas that first get the shock/first hitting the ground (*ZY_wall* end surfaces) will have the displacement values approximate 0.1 mm and spreads to the center between the walls symmetrically to a value approximate 2.4 mm.

Drop test conclusion: Since the data gained from a low-height drop test gives values and results with a non-reusable product. An equal drop test with the only change in height is unnecessary and a change in the design or material should be chosen instead if the product must be re-usable if a scenario of suddenly falling from a height higher than 10 meters occurs. We will therefore conclude that the design is not suitable and has to be re-designed to be reused after a scenario equal to the drop test occurs, but we accept the design since the stakeholder requirements don't include re-usability after a drop test.

Note: This doesn't include the thermal effect on the material changing the brittleness before it hits the ground.

STB| TL

7.14 Conclusion about the design related to FEA

Since our product is only developed to be mounted with a balloon and the effect from the vibration test gives us only trustful data due to illustration of a CubeSat testing, but also possible vibrations due to different scenarios such as vibration movement. But by the drop test regarding only 10 meters. The 3D-printed parts have to be re-produced and mounted. And a higher drop could lead to a more catastrophic scenario.

The effect from the spring force is more important than the drop test, and since the *Main_asm* analysis showed trustful data about the effect of springs on the *sealings*, we will accept this design as a reliable structure and the final design of our project.

We will conclude with the acceptance that our results show us that the 3D printed parts in the C.R.A.F.T-Sat are only concluded as re-usable if it's on the ground.

8 Testing Phase

8.1 PCB Vibration Test

RE|GALK

The reliability and robustness of the onboard electronics is crucial in small satellites such as C.R.A.F.T-Sat. A PCB that connects all components and sensors must withstand harsh environmental conditions, including intense vibrations at the time of launch. This section outlines the significance, approach, and expected results of performing thorough vibration tests on our newly developed PCB. This PCB is made from Flame Retardant 4 (FR-4) material and carries several crucial components such as a Teensy 4.1 microcontroller, GY-91, NTC temperature sensor, and RFM96 radio, all of which are carefully hand-soldered to guarantee connectivity and operation.

What is a vibration test for PCB:

A PCB vibration test is a method used to assess the durability, reliability, and performance of electrical parts soldered to the PCB when subjected to dynamic forces. This testing is particularly vital for space hardware, such as CubeSat, as it replicates the intense mechanical shocks and vibrations experienced during the launch. The test generally utilizes a vibration test, most commonly a **shaker table**, that can generate a broad spectrum of vibration frequencies and intensities. The PCB is mounted on this table, which then exposes it to controlled vibrations that simulate real-world conditions. For our PCB, the test parameters are set to an intensity of approximately 14 Grms (root mean square of gravitational force (Grms)), covering a frequency range from 5 to 2000 Hz, for a duration of approximately two minutes. This setting is chosen to reflect the harsh vibrational environment experienced during the rousted launch, ensuring that all potential vibration-induced failure modes can be identified, addressed, and improved in a new iteration.

Purpose of Vibration Testing for PCB:

The primary goal of vibration tests for PCB, especially in the aerospace context, is to validate the structural and functional integrity of the ship under simulated launch conditions. This process helps identify potential vulnerability in the board design, choice of material, selection of components, and assembly methods, including solder points and components mounting.

Structural Integrity: Vibration tests help to ensure that the selected PCB material (FR-4 in our case) and the physical structure can withstand stresses without fracturing or deforming.

Component Security: Testing also verifies that all components remain securely attached to the board and maintain electrical connectivity throughout and after being subjected to vibrations.

Operational Reliability: By simulating operational stresses, we can forecast and improve the operational reliability of PCB in the actual mission environment, thereby lowering the possibility of failures during the mission.

Design Validation: Successful test results validate our design, confirming that the selected materials, component layout, and soldering techniques are appropriate for the mission requirement. Without such a comprehensive testing, undiscovered defects could endanger the mission.

8.2 Freeze Test Electronics

A cold test of electronic components is a environmental simulation to lower the air temperature to a very cold temperature. A low energy state in the surrounding environment will change the materials which electronic components are made off. Change in temperature will change the resistance in materials and affect small currents, and in worst case make signals too weak to properly transmit. Poor soldier connections could disconnect from the material properties to retract in change in temperature.

To reach low temperatures of around -20°C can your standard freezer be used in summer or at warm locations. Ultra low temperature freezers and shock freezers can reach extreme temperatures of -80°C. Freeze sprays are a cheap option for simple component verification to test data sheet minimum operational temperature and can reach a temperature of about -50°C.

The purpose of a freeze test is to increase the success of your system in cold environments and to figure out if there are any faulty electronics beforehand the mission occurs. A cold test is done to prevent failure during missions in extreme conditions. There is always a chance your components have manufacturing differences to the data sheet values, so testing the components beforehand is important for crucial missions.

GALK|STB

8.3 Noise testing of electronics

Electronics will create magnetic waves when change in current speed occurs. Any digital output and input sending or reading a signal in wires or circuit board traces will create a magnetic field which can accelerate nearby free electrons in conducting materials, here as connected traces and wires on the circuit board.

Noise can have many different frequencies depending on the source for the electronic interference. We have designed the circuit board to prevent loss in signal integrity and cause errors or missing bits during data transfer of digital circuit. For testing of the system for electronic interference is measuring the voltage traces on the circuit board with a oscilloscope to see if any ripples occurs on the voltage output of the regulators.

To check for magnetic interference caused by change in current speed is possible by looping a probe on the oscilloscope to detect a voltage created in the wires which would also affect the traces on the circuit board. The looped probes is simply hovered near the electronic components and traces on the circuit board, and the read voltage spike will tell how much voltage the traces can pick up.

A noisy circuit board is also an unpredictable one. If the interference is external can the problem come and go. If the problem is the circuitry itself will the problem be constant

and one part can affect the other parts of the components near the source. For critical missions is testing mandatory, as components could have issues not known before testing the components before using them. The noise test can also be done with other sources affecting your components.

STB|NNG

8.4 Re-usability of antenna release mechanism

Since we decided to use a burn wire-mechanism it was important to test if it's re-usable due to the functional requirement 3. The testing was divided to gain a final conclusion. The divided testing is;

- Test if the polymer won't melt the area where burn-wire is placed.
- Test the spring and it's force on the sealing and if the chosen rope (in our case, a fishing rope) is usable to release a spring
- Test if the fishing rope could easily be mounted to the sealings
- Test if the sealing opens when the spring is released by a burn-wire mechanism.

STB|NNG

8.4.1 Burn-wire placement

It was important to provide enough information regarding the heat and it's effect on the polymer material. We have therefore tested if the *Antenna_ReleaseBoard* starts melting if a current with less than two seconds opening provides a heat used to melt the fishing rope. We can conclude from testing and a non-destructive test (observation) of the *Antenna_ReleaseBoard* that the heat will not give an destructive result, and due to given data is accepted to be used on the polymer

Note: the testing was used on a 3D printed part made of PLA which has a lower melting temperature compared with PETG

STB|NNG

8.4.2 Spring releasing

It's important to test if the springs is an reliable opening method of the sealing. Therefore, we have multiple times assembled the C.R.A.F.T-Sat with springs, closed the sealing, and removed the object interfering the the releasing and observed that non-failure has occurred relating to the amount of force the spring is providing on the sealing. But, it has been observed that by using the "stick" or manually force on the sealing in the opposite but parallel direction, displacement occurs due to the amount of forces from the spring on the sealing. Spring release has been tested with burn-wire but outside of the C.R.A.F.T-Sat. We accept the displacement due to the possibility of not being able to prevent the forces equally as the fishing rope. and concludes with these tests as accepted as a control of spring releasing.

8.4.3 Fishing rope mounting

The design has small holes meant to the fishing rope, and it could be difficult to mount the fishing rope when the C.R.A.F.T-Sat is not disassembled. Therefore, it's been tested the method of how it's done. The method decided to be used is by placing the rope inside the holes when the sealing is open. Next; close the sealing and place the "stick" to keep it closed. The last step was to tie the fishing rope in without slack. It has been tested, and accepted, but an improvement of the mounting method is accepted as an possibility.

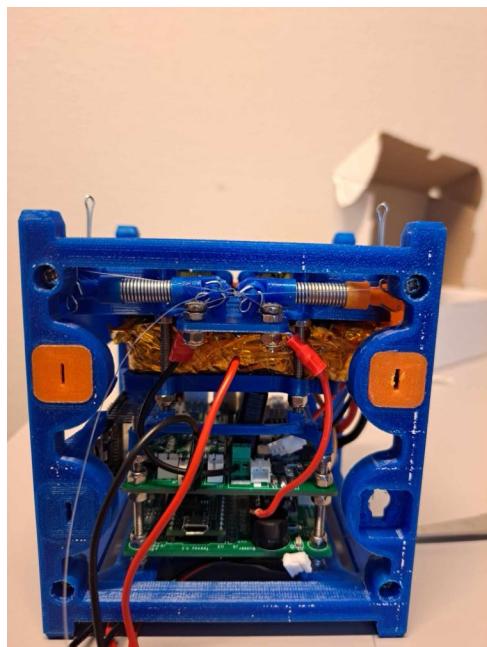


Figure 155: Fishing rope tied - Final design | STB

8.4.4 Sealing & burn-wire releasing

When the C.R.A.F.T-Sat was mounted together, 5 to 6 hours was used, and a new sufficient method should be considered related to which tools to be used. The sealing will be closed by using the fishing rope and is an sufficient method in the use of burn-wire.

The test in releasing the antenna was first a failure due to pin connection, but tests shows that our product could easily be reset for a new burn-wire usement and releasing of antennas.

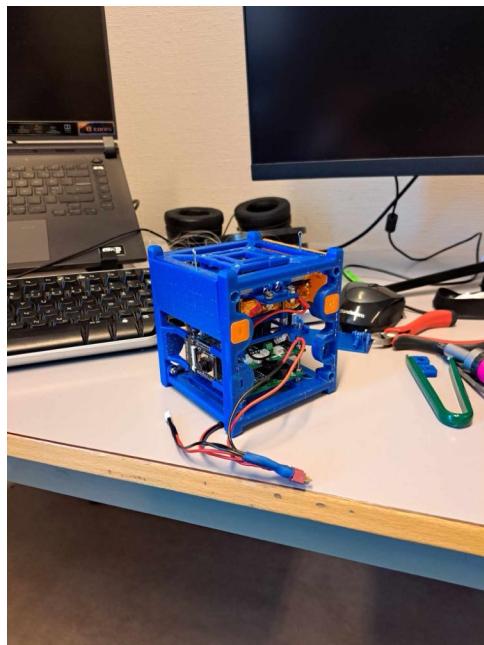


Figure 156: Final design - preloaded for antenna release | STB

STB|

8.4.5 Test conclusion

We will conclude that the burn-wire as the release mechanism works and is an efficient method in releasing the antenna using up-link to control the burn-wire. The final design should be re-updated due to the amount of work in mounting the design but is accepted as it is.

STB|NNG

8.5 3D printed prototype

Since it was required to 3D print the frame, functional requirement 1, printing the antenna release mechanism components was also considered and used. To gain an acceptance of this requirement, the design has to be accepted with a parts regarding an effective

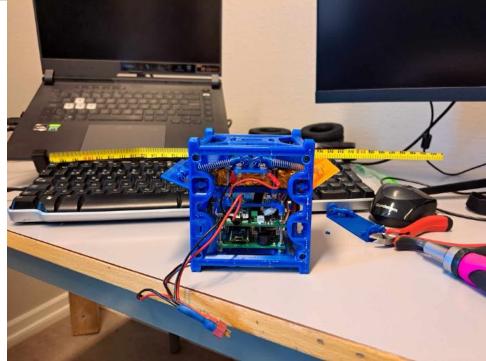


Figure 157: Final design - antenna released | STB

3D printing method. Also, since it's 3D printed sizes will be changed during printing (depending on the accuracy of the FDM printer used) the parts has to be printed for controlling the result sizes. To provide an acceptance, the testing was divided to:

- Test if the frame designed to be equal a CubeSat fits inside a 1U test pod; design requirement 2;
- Test if the 3D printed prototype answers relevant design specifications, appendix A.20; design requirement 4

STB|NNG

8.5.1 1U test pod - frame size

It's required that the C.R.A.F.T-Sat has the equal frame size as an CubeSat. Therefore, a test pod prototype was designed and produced with a laser cut using plywood sheet as the prototype material. The sizes on this prototype is designed to be equal an realistic test pod. The reason plywood sheet is used is because the thickness is already given as one specific value (we assume a specific value since the interval value of the product thickness is not given and it's not measured due to the amount of work). Therefore the thickness is a smaller consideration, and the accuracy of lase cutting (depending on if a plate is curved or not when it's used) the planned sizes gives a more reliable prototype used as a testing object.

The test pod testing with the final design, fig 158 resulted with an acceptance that the outside measurements of the frame (100×100 mm).

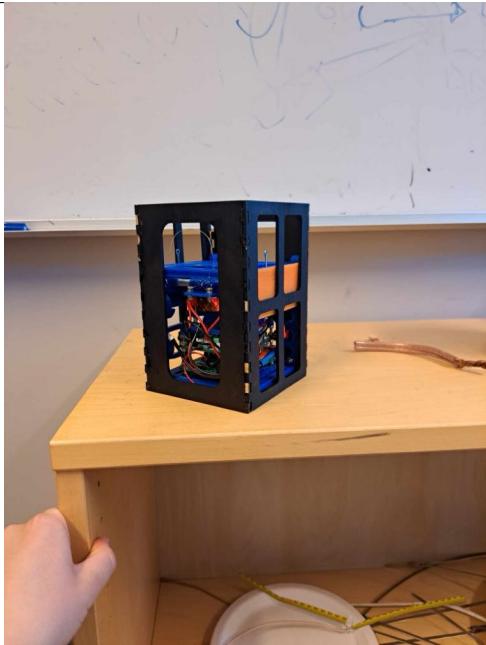


Figure 158: TEST POD & Final design

STB|NNG

8.5.2 Design requirements approved

Control if that chosen requirement regarding the design is approved.

8.6 C.R.A.F.T-Sat library testing

TL|TL

During the development phase of the C.R.A.F.T-Sat library extensive testing has been done throughout the entire process. When we have implemented certain features these have been tested, before deeming them viable. This has been done through acceptance criteria as discussed in section 2.1.4, but also testing to see if the implementation works as intended

8.6.1 Sending and receiving

To answer Functional Requirements 4 and 9 3.2, we needed test sending and receiving. To do this we set up two microcontrollers with radios. One of them also had sensors that could output data to send. For the receiver to receive the data, it needs to "listen" for incoming data on the right frequency. It is therefore important for the satellite and the ground station to use the same frequency. This was done using the LoRa library as mentioned in 5.8.

When the radios could communicate, we needed to manipulate the data at the grounds station so the GUI got the right format on the data string as discussed in section C.15. We decided the simplest method would be to send the string as the GUI need the data to be

received, reducing complexity at the ground station (This is further discussed in Section REF). When the ground station transferred the data in the right format we connected the ground station to the GUI to make sure it worked as intended. By setting up the satellite to print out the sensor data, and then send the data, we could compare the data being sent and the data received to see if they matched. Although we did not take radio noise and distance into consideration we could establish that the sending and receiving worked.

8.6.2 Sensor testing

8.6.3 User Testing

Since we do have experience in programming it is easy to overlook what the end users find easy and what they find difficult. To get a better insight we got help from 6 students attending Vestsiden secondary school. They got to try out the library with the help of the included guide A.14, while we observed and answered questions during the process. The students did not have any prior programming experience, making them the perfect focus group.

Observations during testing revealed that what they struggled the most with was where the function calls should be used. The earlier guide specified in the explanatory text where they should put the different functions, therefore we updated the guide to include the setup and loop part into the guide, making it easy to see where the functions should be used.

Another part they found difficult was the use of the serial monitor, the early guide only specified in the terminology section that they should start the serial monitor. We decided it should also be included in the user guide.

During their programming, we asked control questions to see if they knew what the functions did, and the function name alone was enough to provide the user with enough information to explain what the function did, as was our theory as discussed in section 6.30.

The focus group was successful in programming their own satellite with the help of the user guide and some help from us. Their feedback was that they thought that if they had more time, they would be able to set up the satellite and ground station with minor adjustments to the guide and without our assistance.

8.6.4 GUI Designs

During user testing, five of the students were shown the progress that had been made to C.R.A.F.T-GUI so far. The students were assigned blank Pugh matrices and asked to evaluate three different GUI concept designs. These designs are found in Figures 159, 160 and 161

It is worth emphasizing that the demo that was shown used design in Figure 159 as reference. However, a conscious attempt was made in appearing indifferent/welcoming to the opinions of, even if they preferred a design which would require more drastic changes.

Figure 159: DESIGN 1 (GUI) - USER TESTING

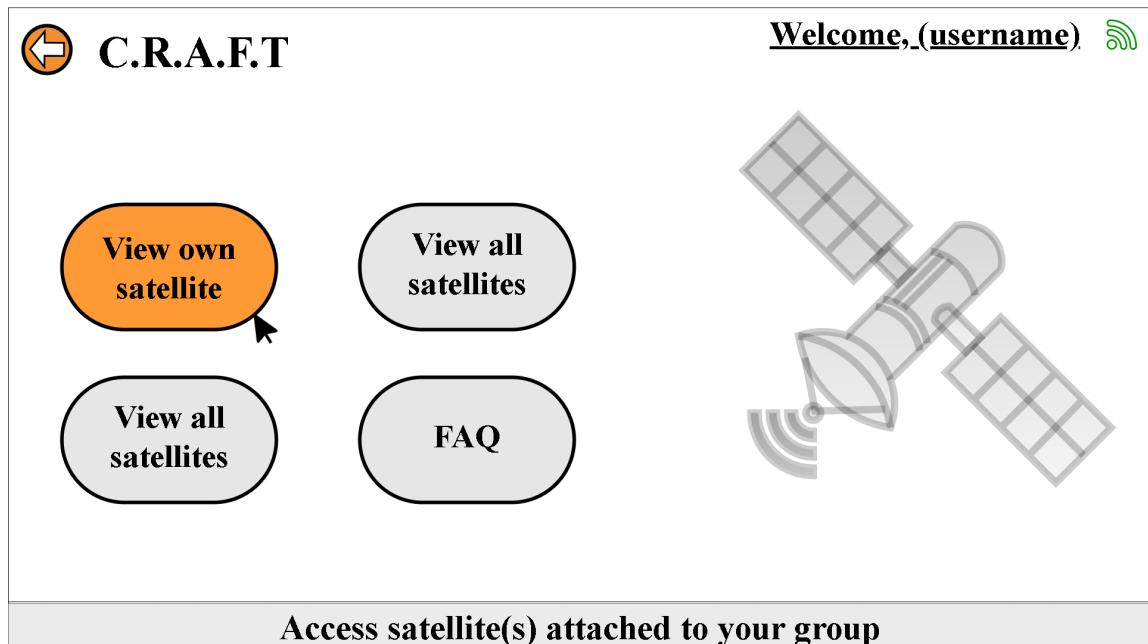


Figure 160: DESIGN 2 (GUI) - USER TESTING

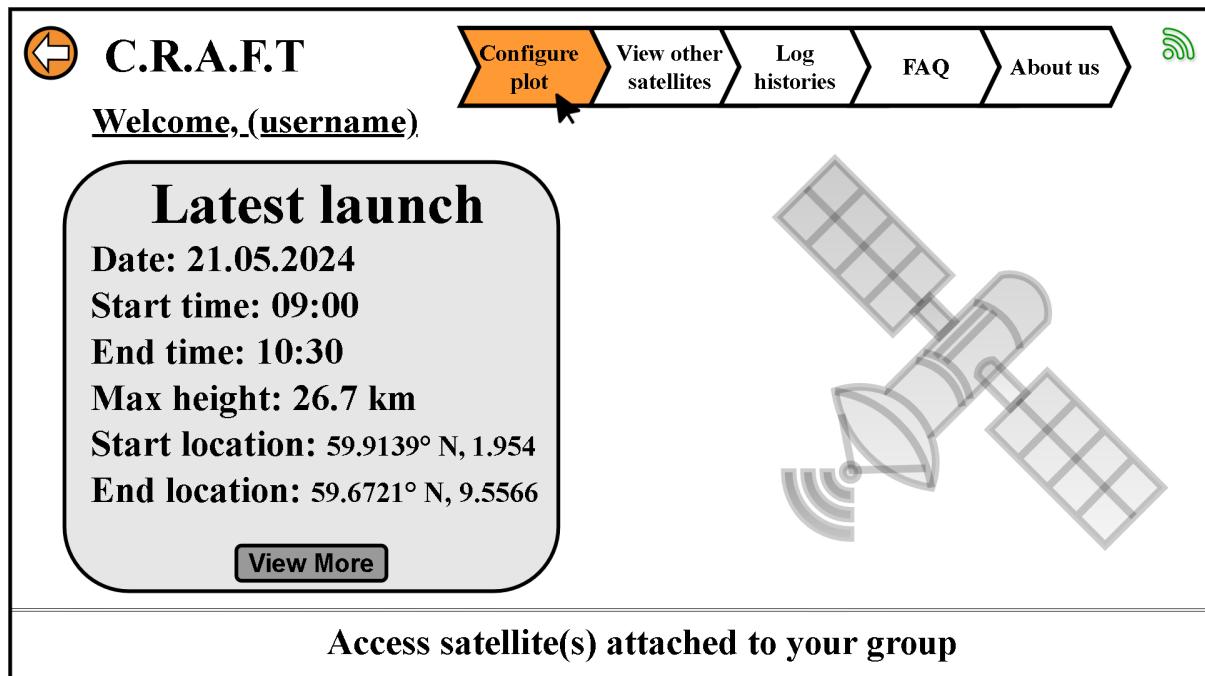
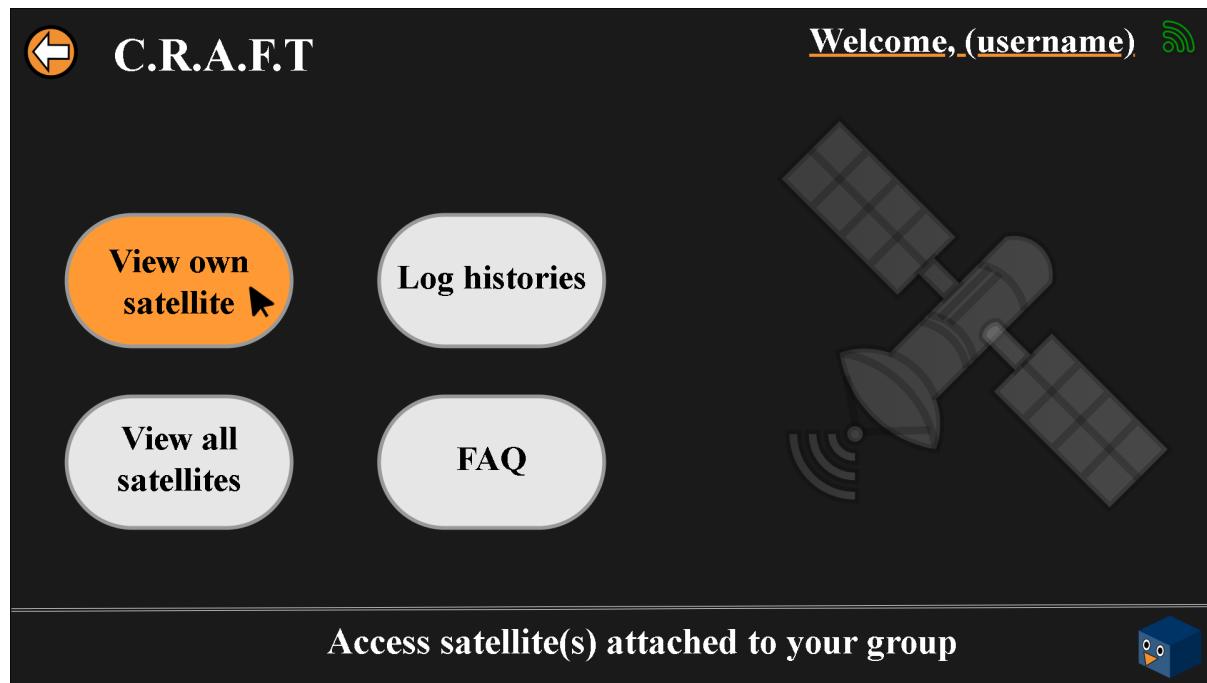


Figure 161: DESIGN 3 (GUI) - USER TESTING



Criteria	Weight (1-3)	Design 1 (1, 0, -1)	Design 2 (1, 0, -1)	Design 3 (1, 0, -1)
Ease of use	2	0	1	0
Intuitive	2	1	1	1
Features	3	0	1	0
Aesthetics	3	1	1	1
Performance/Speed	2			
Total		5	10	5

Criteria	Weight (1-3)	Design 1 (1, 0, -1)	Design 2 (1, 0, -1)	Design 3 (1, 0, -1)
Ease of use	3	1	0	1
Intuitive	2	1	0	1
Features	3	1	1	1
Aesthetics	1	0		0
Performance/Speed	2			
Total		8	3	8

Criteria	Weight (1-3)	Design 1 (1, 0, -1)	Design 2 (1, 0, -1)	Design 3 (1, 0, -1)
Ease of use	3	1	0	1
Intuitive	2	1	0	1
Features	3	1	0	1
Aesthetics	2	1	0	1
Performance/Speed	3			
Total		10	0	10

Criteria	Weight (1-3)	Design 1 (1, 0, -1)	Design 2 (1, 0, -1)	Design 3 (1, 0, -1)
Ease of use	3	0	1	1
Intuitive	3	0	1	1
Features	2	1	0	1
Aesthetics	3	1	1	1
Performance/Speed	3			
Total		5	9	11

Criteria	Weight (1-3)	Design 1 (1, 0, -1)	Design 2 (1, 0, -1)	Design 3 (1, 0, -1)
Ease of use	3	1	-1	1
Intuitive	2	1	-1	1
Features	3	1	1	0
Aesthetics	3	0	0	-1
Performance/Speed	3			
Total		8	-2	3

As evident from these five tables, design no. 1 and 3 edged out as the winner. However, the gap between those two, and design no 2, is minimal. In retrospect, it would have perhaps been wiser to increase the weighting range (i.e., importance rankings) for each criterion in the Pugh matrices. Still, there were common denominators among each result which can be extrapolated to draw a more nuanced understanding of our target audiences wants and needs.

When designing a GUI, it is generally desired for its design to not be controversial according to out intended demographic. Although there were 2 of the students which really liked design no 2, the other three showed varying disdain for it, receiving total scores of -2, 0 and 3 respectively. This observation did incline us more towards design 1 or 3.

These Pugh matrices also revealed an interesting observation regarding design 1 and 3. These two designs have the exact same layout. The only difference between them is the colour theme; dark mode vs. light mode. Before testing, we hypothesised that dark mode would be the preferred choice due to it having a reputation for being more

popular among younger generations. However, more students leaned towards light mode on average. What was most surprising was the student which ranked design no 3 higher than design no 1 for "ease of use", considering that the layouts are identical. To ensure that it was not a mistake, this student was confronted about it and he insisted that he meant dark mode increased the GUI's ease of use. This indicates that there is some correlation between how appearance can impact ones perception of other qualities that are not directly linked together.

9 Verification Phase

STB|NNG

9.1 Objective 2 & 3

To verify that our system answer interim goal 2 & 3 1.1.1 we have tested the following requirements.

1. C.R.A.F.T-Sat must be easy for a student to operate, assemble and disassemble
2. The frame has to be 3D printed
3. The antenna must be able to be released at operation
4. The antenna release mechanism must be re-usable
5. The frame must fit within 1U test pod
6. The frame must answer relevant CubeSat design specifications.

The relevant CubeSat design specifications C.R.A.F.T-Sat is up-filling is:

- 2.1.1 (A.20)
- 2.2.1 (A.20)
- 2.2.1.2 (A.20)
- 2.2.2 (A.20)
- 2.2.4 (A.20)
- 2.2.5 (A.20)
- 2.2.7 (A.20)
- 2.2.8 (A.20)
- 2.2.9 (A.20)
- 2.2.10 (A.20)
- 3.1.1 (A.20)

TL|TL

9.2 Objective 5

To verify that our system answer interim goal 5 1.1.1 we have tested the following requirements.

1. Must be well documented 3.1

-
2. Must be easy to operate 3.1
 3. Must use open source software 3.1
 4. Must follow radio communication laws 3.1
 5. Must be able to transfer data in soft real-time. 3.2
 6. Ground station must receive data from C.R.A.F.T-Sat 3.2
 7. *Ground station must have the ability to save data to file 3.2

Note: * = not strictly related to objective 5

To verify we have answered requirement 1. and 2. We arranged user testing discussed in section 8.6.3. The students used the included C.R.A.F.T-Sat Library User guide A.14 to program their own C.R.A.F.T-Sat. The students was also asked control questions to see if they knew what the line of code did, giving us an indication if we had answered the Stakeholder requirement 3d 3.1, and if our assumption in design section 6.30 was correct. The outcome from the user test was that the students was able to use the library and the included guide, meaning we have sufficiently answered stakeholder requirement 3d 3.1 for interim goal 5. They understood what all the functions did and gave feedback to what was difficult so we could improve the guide. The Stakeholder 1e 3.1 is not sufficiently answered from the user testing since they only used the guide.

We have however documented all the parts that can cause confusion, making further development of the library possible. We have also commented the code explaining what the function does and before parts that may be difficult to understand.

To verify the we have followed the relevant laws and regulations from our stakeholders 3.1. We have implemented duty cycles to answer to the laws regarding radio use as mentioned in section 7.2. We have used the LoRa documentation [112] which states, the time you have to wait is "Time On Air" multiplied with "duty cycle". For our case where we need respect a 10% duty cycle wait time is given by:

$$waitTime = TimeOnAir \cdot 9 \quad (52)$$

We have not been able to verify the transfer of data in soft real time 3.2, since the C.R.A.F.T-Sat does not have any method to track time. So we are not able add the time a packet is sent to the packet itself. Although we have been able to verify the sending and receiving of data we have not been able to time the packet.

The verification has been done through sending sensor data to the GUI. This way we can see that packets arrive correctly.

The ability for the groundstation to save data to file is mostly meant for the GUI saving implementation. As a backup we have implemented the saving of each incoming packets as well. This will save all the raw incoming data to an SD card. By doing this we can make sure that unexpected errors to the GUI does not mean data is lost. Verification has been done by receiving packets and saving them using the implemented functions. Then we have taken out the SD card from the groundstation and used an SD card reader to

view the data stored.

9.3 Object 6

- Database and Flask run on an external server
- GUI is able to send up-link signal
- Data saves as a .CSV file

10 Validation Phase

Due to difficulty in establishing contact with ASE at the later stages in development, we have been unable to validate our final product to a desirable extent. On May 16th 2024, we had our final meeting with our interim external supervisor, whom described our DIY Vibration Shaker solution as "elegant" and was pleased with the results there. The remaining subsystems will have to be validated by ASE post-submission.

The user testing session that has been previously discussed also gave us some insight and some form of validation. Comparing the design of C.R.A.F.T-GUI to most voted for design in Sub-subsection 8.6.4, shows a resemblance. One may therefore, assume that the design of C.R.A.F.T-GUI is somewhat validated to the similarities of the winning concept during our user testing.

11 Conclusion

The assignment was divided into tasks resulted with a normal CanSat and the new updated satellite developed by combining the CanSat and CubeSat; the C.R.A.F.T-Sat. The work done had some difficulties due to the given requirements and the amount of work necessary to 'develop this product- But due to the teams effort, a final product was developed with approves of the requirements written for each task.

We developed two satellites but due to the amount of tasks related to the C.R.A.F.T-Sat, it was the product mostly worked on and discussed in the group. The GUI is developed to work as planned with our product both for a CanSat and a C.R.A.F.T-Sat and was a station related to both products.

The products and report shows the large effort of all the team members in developing the final product. We will conclude that our final product is working as planned, has user manual for students, and has the possibilities to be updated later.

References

- [1] Andøya Space, *Om andøya space education*, Norwegian. [Online]. Available: <https://andoyaspace.no/what-we-do/space-education/om-andoya-space-education/> (visited on 01/19/2024).
- [2] Andøya Space, *Kurs og aktiviteter*, Norwegian. [Online]. Available: <https://andoyaspace.no/what-we-do/space-education/kurs-og-aktiviteter/> (visited on 01/26/2024).
- [3] Andøya Space, *Undervisningsprosjekter*, Norwegian. [Online]. Available: <https://andoyaspace.no/what-we-do/space-education/undervisningsprosjekter/> (visited on 01/26/2024).
- [4] Andøya Space, *Slipp løs din indre utforsker*, Norwegian. [Online]. Available: <https://spaceship-aurora.andoyaspace.no/> (visited on 01/26/2024).
- [5] Andøya Space, *Samarbeidsavtaler*, Norwegian. [Online]. Available: <https://andoyaspace.no/what-we-do/space-education/om-andoya-space-education/samarbeidsavtaler/> (visited on 01/19/2024).
- [6] PJRC, *Teensy® 3.5 Development Board*, English, Aug. 2023. [Online]. Available: <https://www.pjrc.com/store/teensy35.html> (visited on 01/21/2024).
- [7] Mike Cohn, *Epics, features and user stories*, English. [Online]. Available: <https://www.mountaingoatsoftware.com/blog/stories-epics-and-themes> (visited on 11/08/2022).
- [8] Ramon Felip, *How to write good user stories*, English. [Online]. Available: <https://apiumhub.com/tech-blog-barcelona/how-to-write-good-user-stories/> (visited on 03/21/2020).
- [9] Clair Maynard, *Learn scrum with jira software*, Norwegian. [Online]. Available: <https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software>.
- [10] Acekube, *Scrum master vs. product owner: Unlocking the secrets of effective collaboration*, English. [Online]. Available: <https://www.linkedin.com/pulse/scrum-master-vs-product-owner-unlocking-secrets-effective-collaboration/> (visited on 01/20/2023).
- [11] U.S DEPARTMENT of STATE, *Risk Analysis*, English, Accessed: 2024-01-30. [Online]. Available: <https://www.state.gov/risk-analysis/>.
- [13] “What is the current state of the ozone layer?” European Environment Agency. (), [Online]. Available: <https://www.eea.europa.eu/en/topics/in-depth/climate-change-mitigation-reducing-emissions/current-state-of-the-ozone-layer> (visited on 12/12/2023).
- [14] C. Tan, D. J. Lyons, K. Pan, *et al.*, “Radiation effects on the electrode and electrolyte of a lithium-ion battery,” *Journal of Power Sources*, vol. 318, pp. 242–250, 2016, ISSN: 0378-7753. DOI: <https://doi.org/10.1016/j.jpowsour.2016.04.015>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0378775316303573>.
- [15] F. Lv, K. Liu, Z. Wang, J. Zhu, Y. Zhao, and S. Yuan, “Ultraviolet-cured polyethylene oxide-based composite electrolyte enabling stable cycling of lithium battery at low temperature,” *Journal of Colloid and Interface Science*, vol. 596, pp. 257–266, 2021, ISSN: 0021-9797. DOI: <https://doi.org/10.1016/j.jcis.2021.02.095>.

-
- [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0021979721002472>.
- [16] J. Qiu, D. He, M. Sun, *et al.*, “Effects of neutron and gamma radiation on lithium-ion batteries,” *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, vol. 345, pp. 27–32, 2015, ISSN: 0168-583X. DOI: <https://doi.org/10.1016/j.nimb.2014.12.058>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0168583X1401088X>.
- [17] Digitaliserings- og forvaltningsdepartementet, *Forskrift om generelle tillatelser til bruk av frekvenser (fribruksforskriften)* - Lovdata, Norwegian, Laws, Jan. 2012. [Online]. Available: <https://lovdata.no/dokument/SF/forskrift/2012-01-19-77/> (visited on 04/30/2024).
- [18] ETSI EN 300 220-2 V3.2.1 (2018-06), Standard, 650 Route des Lucioles, France, Jun. 2018. [Online]. Available: https://www.etsi.org/deliver/etsi_en/300200_300299/30022002/03.02.01_60/en_30022002v030201p.pdf (visited on 04/30/2024).
- [19] Luftbårne sensorsystemer - Nasjonal sikkerhetsmyndighet, no, May 2020. [Online]. Available: <https://nsm.no/fysisk/luftbarne-sensorsystemer/> (visited on 04/30/2024).
- [20] Samferdselsdepartementet, *Forskrift om lufttrafikkregler og operative prosedyrer* - Lovdata, Norwegian, Dec. 2016. [Online]. Available: <https://lovdata.no/dokument/SF/forskrift/2016-12-14-1578> (visited on 04/30/2024).
- [21] Andøya Space Education, *The CanSat book*, nb-NO, Nov. 2016. [Online]. Available: <https://learn.andoyaspace.no/ebook/the-cansat-book/> (visited on 02/28/2024).
- [22] A. A. Doroshkin, A. M. Zadorozhny, O. N. Kus, V. Y. Prokopyev, and Y. M. Prokopyev, “Experimental study of lora modulation immunity to doppler effect in cubesat radio communications,” *IEEE Access*, vol. 7, pp. 75 721–75 731, 2019.
- [23] T. Ameloot, M. Moeneclaey, P. Van Torre, and H. Rogier, “Characterizing the impact of doppler effects on body-centric lora links with sdr,” *Sensors*, vol. 21, no. 12, p. 4049, Jun. 12, 2021. DOI: 10.3390/s21124049.
- [24] D. Samarasekara. “2. getting started with sdrsharp,” Radio Society of Sri Lanka. (), [Online]. Available: <https://rssl.lk/2023/07/13/getting-started-with-sdrsharp/#:~:text=SDRSharp%20is%20a%20popular%20software,relatively%20easy%20to%20start%20with> (visited on 07/13/2023).
- [25] T. L. of Congress, *TSV, Tab-Separated Values — loc.gov*. [Online]. Available: [https://www.loc.gov/preservation/digital/formats/fdd/fdd000533.shtml#:~:text=A%20tab%2Dseparated%20values%20\(TSV,are%20separated%20by%20tab%20characters.%7D](https://www.loc.gov/preservation/digital/formats/fdd/fdd000533.shtml#:~:text=A%20tab%2Dseparated%20values%20(TSV,are%20separated%20by%20tab%20characters.%7D).
- [26] Flatfile, *What is a CSV file: A comprehensive guide — flatfile.com*. [Online]. Available: <https://flatfile.com/blog/what-is-a-csv-file-guide-to-uses-and-benefits/#:~:text=csv%20file%20extension%20stands%20for,as%20data%20separated%20by%20commas.%7D>.
- [27] The Matplotlib development team, *Matplotlib: Visualization with python*, English, Feb. 27, 2024. [Online]. Available: <https://matplotlib.org/>.

-
- [28] D. Doomen, *Merging git repositories without losing history*, English. [Online]. Available: <https://www.continuousimprover.com/2023/02/merging-repositories.html> (visited on 02/03/2023).
 - [29] Guido van Rossum, Barry Warsaw, Alyssa Coghlan, *Pep 8 – style guide for python code*, English. [Online]. Available: <https://peps.python.org/pep-0008/> (visited on 01/22/2024).
 - [30] Aditya Sharma, *Introduction to gui with tkinter in python*, English. [Online]. Available: <https://www.datacamp.com/tutorial/gui-tkinter-python> (visited on 02/25/2024).
 - [31] Vibhutig13, *Mit license*, English. [Online]. Available: <https://github.com/PacktPublishing/Python-GUI-Programming-with-Tkinter/blob/master/LICENSE> (visited on 02/10/2024).
 - [32] Martin Fitzpatrick, Khumbo Klein, Puniton Chaetognathan, Leo Well, *Pyqt vs. tkinter — which should you choose for your next gui project?* English. [Online]. Available: <https://www.pythonguis.com/faq/pyqt-vs-tkinter/> (visited on 02/17/2024).
 - [33] Riverbank Computing Limited, *License faq*, English. [Online]. Available: <https://riverbankcomputing.com/commercial/license-faq> (visited on 02/10/2024).
 - [34] Snowflake Inc., *A faster way to build and share data apps*, English. [Online]. Available: <https://streamlit.io/> (visited on 02/15/2024).
 - [35] Arvindra Sehmi, *Introduction to streamlit and streamlit components*, English. [Online]. Available: <https://auth0.com/blog/introduction-to-streamlit-and-streamlit-components/> (visited on 02/27/2024).
 - [36] Bernd Klein, *Tkinter - the python interface for tk*, English, Feb. 27, 2024. [Online]. Available: <https://python-course.eu/tkinter/>.
 - [37] *Heat Flux Formula: Meaning, Methods, Formulas, Examples*, en-US, Section: Physics Formulas, Apr. 2020. [Online]. Available: <https://www.toppr.com/guides/physics-formulas/heat-flux-formula/> (visited on 05/20/2024).
 - [38] C. Trick, *What is Electromagnetic Interference (EMI)?* en-us. [Online]. Available: <https://www.trentonsystems.com/en-us/resource-hub/blog/what-is-electromagnetic-interference> (visited on 03/04/2024).
 - [39] fx-builder, *5 PCB Thermal Management Techniques*, en-US, Jul. 2022. [Online]. Available: <https://www.mclpcb.com/blog/thermal-pcb-management-techniques/> (visited on 03/04/2024).
 - [40] ADMIN, *Thermal Vias: Heat Dissipation through Vias in PCB / MADPCB*, en-US, Aug. 2021. [Online]. Available: <https://madpcb.com/glossary/thermal-vias/> (visited on 03/04/2024).
 - [41] *Electromagnetic Radiation - an overview / ScienceDirect Topics*. [Online]. Available: <https://www.sciencedirect.com/topics/neuroscience/electromagnetic-radiation> (visited on 04/10/2024).
 - [42] *Electromagnetic waves / National Oceanic and Atmospheric Administration*, en. [Online]. Available: <https://www.noaa.gov/jetstream/satellites/electromagnetic-waves> (visited on 04/10/2024).
 - [43] M. Ozols, “Differential geometric formulation of Maxwell’s equations,” en,

-
- [44] H. D. Young, R. A. Freedman, and A. L. Ford, *University physics: with modern physics*, eng, Fifteenth edition in SI units. Harlow: Pearson Education Limited, 2020, OCLC: 1380676054, ISBN: 978-1-292-31473-0.
- [45] *Electromagnetic waves*. [Online]. Available: <http://electron9.phys.utk.edu/optics421/modules/m1/emwaves.htm> (visited on 04/10/2024).
- [46] *How To Read An Antenna Chart - Gristle King - A Guide to DePIN*. [Online]. Available: <https://gristleking.com/all-about-antennas-radiation-patterns/comment-page-1/> (visited on 04/10/2024).
- [47] *Linear Regulators*, en-US. [Online]. Available: <https://practicalee.com/linear-regulators/> (visited on 05/20/2024).
- [48] *The Challenges of Right-Angle PCB Traces*, en-US, Aug. 2022. [Online]. Available: <https://resourcespcb.cadence.com/blog/2022-the-challenges-of-right-angle-pcb-traces> (visited on 05/20/2024).
- [52] *Neo-6*, English. [Online]. Available: <https://www.u-blox.com/en/product/neo-6-series> (visited on 02/23/2024).
- [53] *Neo-m9n*, English. [Online]. Available: <https://www.u-blox.com/en/product/neo-m9n-module?legacy=Current#Product-Selection> (visited on 02/23/2024).
- [54] T. L. Floyd, *Electronic devices*, eng, Tenth edition, Conventional current version. NY, NY: Pearson, 2018, OCLC: 1051432254, ISBN: 978-0-13-441444-7.
- [57] *Teensy 4.1 technical information*, English. [Online]. Available: <https://pjrc.com/store/teensy41.html#digital> (visited on 05/12/2024).
- [63] *Ozone*, en, Text.Article, Publisher: NASA Earth Observatory, Jul. 1999. [Online]. Available: https://earthobservatory.nasa.gov/features/Ozone/ozone_2.php (visited on 05/20/2024).
- [64] *The Role of Temperature and Humidity Sensors in Weather Stations*. [Online]. Available: <https://www.rikasensor.com/the-role-of-temperature-and-humidity-sensors-in-weather-stations.html> (visited on 05/20/2024).
- [65] D. Murden, *What is a Lithium Battery: Definition, Technology & Work Process*, en-GB, Mar. 2022. [Online]. Available: <https://ecotreelithium.co.uk/news/what-are-lithium-batteries/> (visited on 05/21/2024).
- [66] *What is lithium polymer battery (LiPo)? / Definition from TechTarget*, en. [Online]. Available: <https://www.techtarget.com/whatis/definition/lithium-polymer-battery-LiPo> (visited on 05/21/2024).
- [67] *The advantages and disadvantages of lithium polymer battery / Shenzhen AES Power battery*. [Online]. Available: <https://www.szaspower.com/industry-news/The-advantages-and-disadvantages.html> (visited on 05/21/2024).
- [68] R. Baktur, *Antenna design for CubeSats*. Boston ; London: Artech House, 2022, OCLC: on1268325109, ISBN: 978-1-63081-785-5.
- [69] T. B. Pratiksha Sarma Hidam Kumarjit Singh, *A real-time data acquisition system for monitoring sensor data*, English. [Online]. Available: https://www.researchgate.net/profile/Pratiksha-Sarma-2/publication/326398953_A_Real-Time_Data_Acquisition_System_for_Monitoring_Sensor_Data/links/5b4a05b7aca272c609409d43/A-Real-Time-Data-Acquisition-System-for-Monitoring-Sensor-Data.pdf (visited on 06/30/2018).
- [70] M. K. S. Fahmida Ahmed1 Shakh Md. Alimuzjaman Alim, *433 mhz (wireless rf) communication between two arduino uno*, 2016.

-
- [71] opensource. “What is a open source,” opensource. (), [Online]. Available: <https://opensource.com/resources/what-open-source#:~:text=Doesn%27t%20%22open%20source%22,or%20to%20which%20they%20contribute..>
- [72] C. a License. “Licenses,” Choose a License. (), [Online]. Available: <https://choosealicense.com/licenses/>.
- [73] Microsoft. “Description of the database normalization basics,” Microsoft. (), [Online]. Available: <https://learn.microsoft.com/en-us/office/troubleshoot/access/database-normalization-description> (visited on 07/14/2023).
- [74] Microsoft. “Non-relational data and nosql,” Microsoft. (), [Online]. Available: <https://learn.microsoft.com/en-us/azure/architecture/data-guide/big-data/non-relational-data>.
- [75] IBM. “Non-relational data and nosql,” IBM. (), [Online]. Available: Web%20server%20versus%20application%20server:%20What's%20the%20difference?
- [76] S. Yegulalp. “6 ways to package python apps for re-use,” InfoWorld. (), [Online]. Available: <https://www.infoworld.com/article/3656628/6-ways-to-package-python-apps-for-re-use.html> (visited on 04/13/2022).
- [77] Pyinstaller. “What pyinstaller does and how it does it,” Pyinstaller. (), [Online]. Available: <https://pyinstaller.org/en/latest/operating-mode.html>.
- [78] Pyinstaller. “Common issues and pitfalls,” Pyinstaller. (), [Online]. Available: <https://pyinstaller.org/en/latest/common-issues-and-pitfalls.html>.
- [79] B. Wolford. “What is gdpr, the eu’s new data protection law?” GDPR EU. (), [Online]. Available: <https://gdpr.eu/what-is-gdpr/>.
- [80] Datatilsynet. “Om personopplysningsloven med forordning og når den gjelder,” Datatilsynet. (), [Online]. Available: <https://www.datatilsynet.no/regelverk-og-verktøy/lover-og-regler/om-personopplysningsloven-og-nar-den-gjelder/> (visited on 11/23/2023).
- [81] I. C. title = GDPR - Personal Data, Intersoft Consulting. (), [Online]. Available: <https://gdpr.eu/what-is-gdpr/>.
- [82] I. Consulting. “Gdpr - encryption,” Intersoft Consulting. (), [Online]. Available: <https://gdpr-info.eu/issues/encryption/>.
- [83] Twilio. “What is a time-based one-time password (totp)?” Twilio. (), [Online]. Available: <https://www.twilio.com/docs/glossary/totp>.
- [84] M. Countryman. “Flask-bcrypt.” (), [Online]. Available: <https://flask-bcrypt.readthedocs.io/en/1.0.1/>.
- [87] K. Moholt, Personal communication, unpublished, May 16 2024.
- [88] IBM. “Using the on delete cascade option,” IBM. (), [Online]. Available: <https://www.ibm.com/docs/en/informix-servers/14.10?topic=clause-using-delete-cascade-option>.
- [89] K. Moholt, “Modellering,” Lecture, Univeristy of South-East Norway, Kongsberg, 2022.
- [90] Wattnotions. “Making a vibration table using a subwoofer,” Wattnotions. (), [Online]. Available: <https://www.wattnotions.com/making-a-vibration-table-using-a-subwoofer/>.
- [91] Semtech, *SX1276/77/78/79 - 137 MHz to 1020 MHz Low Power Long Range Transceiver*, English, Datasheet, May 2020. [Online]. Available: <https://semtech.com>.

-
- my.salesforce.com/sfc/p/#E0000000JelG/a/2R0000001Rbr/6EfVZUorrpoKFFvaF_Fkpgp5kzjiNyiaBqcpqh9qSjE (visited on 05/12/2024).
- [92] Yung-Hsiang Hu, *ifTNT/lora-air-time: A Calculator that calculate the Time on Air (ToA) of LoRa packet. Based on the formula in Semtech's datasheet.* Feb. 2024. [Online]. Available: <https://github.com/ifTNT/lora-air-time> (visited on 05/12/2024).
- [94] GeeksForGeeks. “How to hide sensitive credentials using python,” GeeksForGeeks. (), [Online]. Available: <https://www.geeksforgeeks.org/how-to-hide-sensitive-credentials-using-python/> (visited on 09/14/2021).
- [96] W. Staff. “Input sanitization,” Webopedia. (), [Online]. Available: <https://www.webopedia.com/definitions/input-sanitization/> (visited on 06/03/2023).
- [99] Tom Schimansky, *Customtkinter*, English, Apr. 20, 2024. [Online]. Available: <https://github.com/TomSchimansky/CustomTkinter?tab=readme-ov-file>.
- [100] © 2024 MongoDB, Inc., *Loved by developers. built for*, English, Apr. 25, 2024. [Online]. Available: <https://www.mongodb.com/>.
- [101] Python Software Foundation, *Csv — csv file reading and writing*, English, Mar. 7, 2024. [Online]. Available: <https://docs.python.org/3/library/csv.html>.
- [102] Python Software Foundation, *Os — miscellaneous operating system interfaces*, English, Mar. 7, 2024. [Online]. Available: <https://docs.python.org/3/library/os.html>.
- [103] Python Software Foundation, *Datetime — basic date and time types*, English, Mar. 7, 2024. [Online]. Available: <https://docs.python.org/3/library/datetime.html>.
- [104] NumFOCUS, Inc, *Pandas*, English, Mar. 10, 2024. [Online]. Available: <https://pandas.pydata.org/>.
- [105] Chris Liechti, *Pyserial*, English, Mar. 12, 2024. [Online]. Available: <https://pyserial.readthedocs.io/en/latest/pyserial.html>.
- [106] Nordisk Film Kino, *Nordisc film kino*, Norwegian, Mar. 2, 2024. [Online]. Available: <https://www.nfkino.no/?city=drammen>.
- [107] Google, *Cloud build documentation*, English, May 19, 2024. [Online]. Available: <https://cloud.google.com/build/docs>.
- [108] Google, *Cloud run documentation*, English, May 19, 2024. [Online]. Available: <https://cloud.google.com/run/docs>.
- [109] Python Software Foundation, *Tkinter — python interface to tcl/tk*, English. [Online]. Available: <https://docs.python.org/3/library/tkinter.html> (visited on 01/16/2024).
- [110] Pallets, *Flask web development, one drop at the time*, English, May 19, 2024. [Online]. Available: <https://flask.palletsprojects.com/en/2.0.x/>.
- [111] Python Software Foundation, *Threading — thread-based parallelism*, English, Mar. 14, 2024. [Online]. Available: <https://docs.python.org/3/library/threading.html>.
- [112] E. B, *LoRa Documentation*, English, Documentation, May 202. [Online]. Available: <https://lora.readthedocs.io/en/latest/> (visited on 05/20/2024).
- [113] Bosch Sensortec, “BST-BMP280-DS001-26,” English, Data sheet BST-BMP280-DS001-26. [Online]. Available: <https://www.bosch-sensortec.com/media/>

-
- boschsensor.com/downloads/datasheets/bst-bmp280-ds001.pdf (visited on 01/24/2024).
- [114] InvenSense Inc, “MPU-9250 Product Specification Revision 1.1,” English, 1745 Technology Drive, San Jose, CA 95110 U.S.A, Data sheet PS-MPU-9250A-01, Jun. 2016, p. 42. [Online]. Available: <https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf> (visited on 01/24/2024).
- [115] InvenSense Inc, “MPU-9255 Product Specification,” English, 1745 Technology Drive, San Jose, CA 95110 U.S.A, Data sheet DS-000007, Sep. 2014, p. 42. [Online]. Available: <https://stanford.edu/class/ee267/misc/MPU-9255-Datasheet.pdf> (visited on 01/24/2024).
- [116] The European Space Agency, *What is a CanSat?* English. [Online]. Available: https://www.esa.int/Education/CanSat/What_is_a_CanSat (visited on 01/20/2024).
- [117] NASA, *What are SmallSats and CubeSats?* English, Aug. 2023. [Online]. Available: <https://www.nasa.gov/what-are-smallsats-and-cubesats/> (visited on 01/20/2024).
- [118] Semtech, *LoRa (PHY)*, English. [Online]. Available: <https://www.semtech.com/lora/what-is-lora> (visited on 01/20/2024).
- [119] Semtech, *LoRaWAN® Standard*, English. (visited on 01/20/2024).
- [120] David Meaney, *LoRa and LoRaWAN Timing*, English. [Online]. Available: <https://ecsxtal.com/lora-lorawan-timing/> (visited on 01/20/2024).
- [121] Eric Peña, Mary Grace Legaspi, *UART: A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter / Analog Devices*, English, Dec. 2020. [Online]. Available: <https://www.analog.com/en/resources/analog-dialogue/articles/uart-a-hardware-communication-protocol.html> (visited on 02/29/2024).
- [122] S. Campbell, *Basics of the SPI Communication Protocol*, en-US, Feb. 2016. [Online]. Available: <https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/> (visited on 03/02/2024).
- [123] S. Campbell, *Basics of the I2C Communication Protocol*, en-US, Feb. 2016. [Online]. Available: <https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/> (visited on 03/02/2024).
- [124] T. Axsom, *3D Printing Inserts: Types, Installation & Best Practices*, en-US. [Online]. Available: <https://www.fictiv.com/articles/solving-common-problems-with-3d-printed-inserts> (visited on 05/10/2024).
- [125] techstudycell, *Program ESP32-CAM using Arduino UNO*, en-US, Jun. 2020. [Online]. Available: <https://easyelectronicsproject.com/esp32-projects/program-esp32cam-arduino/> (visited on 05/15/2024).
- [126] *ESP32-CAM Pinout Reference*, en-US, Mar. 2023. [Online]. Available: <https://lastminuteengineers.com/esp32-cam-pinout-reference/> (visited on 05/15/2024).
- [127] H. Tai, *Hideakitai/ESP32SPISlave*, original-date: 2021-07-26T09:59:46Z, May 2024. [Online]. Available: <https://github.com/hideakitai/ESP32SPISlave> (visited on 05/15/2024).
- [128] KiCad, *About KiCad*, English, Accessed: 2024-01-30. [Online]. Available: <https://www.kicad.org/about/kicad/>.

-
- [129] KiCad, *Schematic Capture*, English, Accessed: 2024-01-30. [Online]. Available: <https://www.kicad.org/discover/schematic-capture/>.
 - [130] KiCad, *Spice Simulation*, English, Accessed: 2024-01-30. [Online]. Available: <https://www.kicad.org/discover/spice/>.
 - [131] KiCad, *Pcb Design in PcbNew*, English, Accessed: 2024-01-30. [Online]. Available: <https://www.kicad.org/discover/pcb-design/>.
 - [132] KiCad, *3d PCB Viewer Features*, English, Accessed: 2024-01-30. [Online]. Available: <https://www.kicad.org/discover/3dviewer/>.
 - [133] KiCad, *Gerber Features*, English, Accessed: 2024-01-30. [Online]. Available: <https://www.kicad.org/discover/gerber-viewer/>.
 - [134] *Teensy® 4.1*. [Online]. Available: <https://www.pjrc.com/store/teensy41.html#tech> (visited on 01/31/2024).
 - [135] *GY-91*, nb-NO, Apr. 2018. [Online]. Available: <https://learn.andoyaspace.no/ebook/the-cansat-book/v6-2/sensors-for-primary-mission/gy-91/> (visited on 01/31/2024).
 - [136] *LoRa*, en, Page Version ID: 1190225333, Dec. 2023. [Online]. Available: <https://en.wikipedia.org/w/index.php?title=LoRa&oldid=1190225333> (visited on 01/31/2024).
 - [137] R. Bridge, *8 Key Features of LoRa® Technology*, en, Dec. 2021. [Online]. Available: <https://radiobridge.com/lora-key-features> (visited on 01/31/2024).
 - [138] *Teensy Technical Specs Comparision Table*. [Online]. Available: <https://pjrc.com/teensy/techspecs.html> (visited on 01/31/2024).
 - [139] *The RFM96 radio*, nb-NO, Jul. 2018. [Online]. Available: <https://learn.andoyaspace.no/ebook/the-cansat-book/v2018/the-rfm96-radio/> (visited on 01/31/2024).
 - [140] *Battery Life Calculator / DigiKey Electronics*. [Online]. Available: <https://www.digikey.no/en/resources/conversion-calculators/conversion-calculator-battery-life> (visited on 02/18/2024).
 - [141] *Functions of Solar Charge Controller*, en. [Online]. Available: <https://www.inverter.com/functions-of-solar-charge-controller> (visited on 03/01/2024).
 - [142] *1904 / Adafruit Mikrolitiumpolymerbatteri-lader Micro USB / Elfa Distrelec Norge*, no. [Online]. Available: <https://www.elfadistrelec.no/no/mikrolitiumpolymerbatteri-lader-micro-usb-adafruit-1904/p/30091160> (visited on 03/01/2024).
 - [143] Gantt.com, *What is a gantt chart?* English, Accessed: 2024-01-31. [Online]. Available: <https://www.gantt.com/>.
 - [144] W. Bolton, *Mechatronics: electronic control systems in mechanical and electrical engineering*, Seventh edition. Harlow, England ; New York: Pearson Education Limited, 2019, ISBN: 978-1-292-25097-7.
 - [145] Ken Shirriff, *Reverse engineering a 7805 voltage regulator*, English, Accessed: 2024-01-31. [Online]. Available: <https://www.righto.com/2014/09/reverse-engineering-counterfeit-7805.html>.
 - [146] T. Instrument, *Mua78mxx positive-voltage regulators*, English, UA78M33 Datasheet, JUNE 1976 –REVISED JANUARY 2015.
 - [147] *Buzzer*. [Online]. Available: <https://no.rs-online.com/web/c/passive-components/sounder-buzzer-microphone-components/piezo-buzzers/?searchTerm=buzzer> (visited on 01/31/2024).

-
- [148] *Gnss systems*, English. [Online]. Available: <https://www.gps.gov/systems/gnss/> (visited on 02/24/2024).
 - [149] *Pcb trace width calculator*, English. [Online]. Available: <https://www.4pcb.com/trace-width-calculator.html> (visited on 02/29/2024).
 - [155] Christian Koch, *Learn python by creating 10 apps with tkinter*, English. [Online]. Available: <https://www.udemy.com/course/learn-python-by-creating-10-apps/> (visited on 01/17/2024).
 - [156] israel-dryer, *Ttkbootstrap*, English. [Online]. Available: <https://ttkbootstrap.readthedocs.io/en/latest/>.
 - [157] Lore-Gaviano, Neoakay005, *Unsupported locale setting on module import for norwegian locale*, English. [Online]. Available: <https://github.com/israel-dryer/ttkbootstrap/issues/505> (visited on 01/16/2024).
 - [158] JetBrains s.r.o., *Jetbrains ai*, English. [Online]. Available: <https://www.jetbrains.com/ai/#plans-and-pricing> (visited on 01/19/2024).
 - [159] Google Cloud, *Large language models powered by world-class google ai*, English. [Online]. Available: <https://cloud.google.com/ai/l1ms> (visited on 02/11/2024).
 - [160] JetBrains s.r.o., *Jetbrains ai terms of service*, English. [Online]. Available: <https://www.jetbrains.com/legal/docs/terms/jetbrains-ai-service/> (visited on 01/23/2024).
 - [161] Martin Fitzpatrick, *Drag & drop widgets with pyqt5*, English, Feb. 27, 2024. [Online]. Available: <https://www.pythonguis.com/faq/pyqt-drag-drop-widgets/>.
 - [162] Riverbank Computing Limited, *Project description*, English, Feb. 27, 2024. [Online]. Available: <https://pypi.org/project/PyQt5/>.
 - [163] Python Software Foundation, *Shutil — high-level file operations*, English, Mar. 12, 2024. [Online]. Available: <https://docs.python.org/3/library/shutil.html>.
 - [164] Python Software Foundation, *Concurrent.futures — launching parallel tasks*, English, Mar. 14, 2024. [Online]. Available: <https://docs.python.org/3/library/concurrent.futures.html>.
 - [165] Python Software Foundation, *Queue — a synchronized queue class*, English, Mar. 14, 2024. [Online]. Available: <https://docs.python.org/3/library/queue.html>.
 - [166] Python Software Foundation, *Re — regular expression operations*, English, Mar. 16, 2024. [Online]. Available: <https://docs.python.org/3/library/re.html>.
 - [167] The Matplotlib development team, *Matplotlib.dates*, English, Mar. 27, 2024. [Online]. Available: https://matplotlib.org/stable/api/dates_api.html.
 - [168] Tom Schimansky, *Tkintermapview*, English, May 10, 2024. [Online]. Available: <https://github.com/TomSchimansky/TkinterMapView>.
 - [169] A. Johnstone, “CubeSat design specification Rev. 14.1 the CubeSat program,” *Cal Poly SLO*, pp. 1–34, 2022.
 - [172] *What Is FDM 3D Printing? – Simply Explained*, en, Feb. 2024. [Online]. Available: <https://all3dp.com/2/fused-deposition-modeling-fdm-3d-printing-simply-explained/> (visited on 05/12/2024).
 - [173] S. Roland, *1.9: Hydrostatic Equilibrium*, en, Jan. 2020. [Online]. Available: [https://geo.libretexts.org/Bookshelves/Meteorology_and_Climate_Science/Practical_Meteorology_\(Stull\)/01%3A_Atmospheric_Basics/1.09%3A_Hydrostatic_Equilibrium](https://geo.libretexts.org/Bookshelves/Meteorology_and_Climate_Science/Practical_Meteorology_(Stull)/01%3A_Atmospheric_Basics/1.09%3A_Hydrostatic_Equilibrium) (visited on 05/05/2024).
 - [174] S. Kalpakjian, *Manufacturing engineering and technology*, eng, Harlow, 2023.

-
- [177] J. A. V. Vilán, M. L. Estévez, and F. A. Agelet, “Antenna Deployment Mechanism for the Cubesat Xatcobeo. Lessons, Evolution and Final Design,” en,
- [178] T. Xometry, *All About PLA 3D Printing Filament: Composition, Properties, Differences*, en-us. [Online]. Available: <https://www.xometry.com/resources/3d-printing/pla-3d-printing-filament/> (visited on 01/13/2024).
- [179] *Overview of materials for PETG Copolyester*, English, Material Property Data. [Online]. Available: <https://matweb.com/search/DataSheet.aspx?MatGUID=4de1c85bb946406a86c52b688e3810d0&ckck=1> (visited on 05/04/2024).
- [180] *All About PETG 3D Printing Filament: Materials, Properties, Definition*, en-us. [Online]. Available: <https://www.xometry.com/resources/3d-printing/petg-3d-printing-filament/> (visited on 02/21/2024).
- [181] A. Crass, *PEEK® for Aerospace/Defense Applications*, en-US, Section: Aerospace & Defense Industries Blogs, Sep. 2022. [Online]. Available: <https://performanceplastics.com/blog/peek-for-aerospace-defense-applications/> (visited on 02/21/2024).
- [182] I. Kauppila, *Selective Laser Melting (SLM 3D Printing) – The Ultimate Guide*, en, Feb. 2023. [Online]. Available: <https://all3dp.com/1/selective-laser-melting-guide/> (visited on 04/10/2024).
- [183] *Overview of materials for Polylactic Acid (PLA) Biopolymer*, English, Material Property Data. [Online]. Available: <https://matweb.com/search/DataSheet.aspx?MatGUID=ab96a4c0655c4018a8785ac4031b9278> (visited on 05/11/2024).
- [184] *NS-EN ISO 527-2:2012*, nb, May 2012. [Online]. Available: <https://online.standard.no/nb/ns-en-iso-527-2-2012> (visited on 04/10/2024).
- [185] *NS-EN ISO 527-1:2019*, en, Dec. 2019. [Online]. Available: <https://online.standard.no/en/ns-en-iso-527-1-2019> (visited on 05/13/2024).
- [186] *NS-EN ISO 291:2008*, en, Sep. 2008. [Online]. Available: <https://online.standard.no/en/ns-en-iso-291-2008> (visited on 05/13/2024).
- [187] L. Physics. “The doppler effect,” LibreTexts Physics. (), [Online]. Available: [https://phys.libretexts.org/Bookshelves/University_Physics/University_Physics_\(OpenStax\)/Book%3A_University_Physics_I_-_Mechanics_Sound_Oscillations_and_Waves_\(OpenStax\)/17%3A_Sound/17.08%3A_The_Doppler_Effect](https://phys.libretexts.org/Bookshelves/University_Physics/University_Physics_(OpenStax)/Book%3A_University_Physics_I_-_Mechanics_Sound_Oscillations_and_Waves_(OpenStax)/17%3A_Sound/17.08%3A_The_Doppler_Effect).
- [188] Microsoft. “Quickstart: Create an instance of azure database for mysql - flexible server by using the azure portal,” Microsoft. (), [Online]. Available: <https://learn.microsoft.com/en-us/azure/mysql/flexible-server/quickstart-create-server-portal> (visited on 03/13/2024).
- [189] E. Fernandez. “How to use mysql workbench to connect to databases,” Nexcess. (), [Online]. Available: <https://www.nexcess.net/help/how-to-use-mysql-workbench-to-connect-to-a-database/> (visited on 07/12/2022).
- [191] D. H. Litwhiler. “A custom vibration test fixture using a subwoofer,” Penn State University. (2022), [Online]. Available: https://ijme.us/cd_11/PDF/Paper%2068%20ENT%202025.pdf.
- [192] PyOTP. “Pyotp - the python one-time password library,” PyOTP. (), [Online]. Available: <https://pyotp.readthedocs.io/en/stable/>.

Bibliography

- [193] C. E. Yoe, *Principles of risk analysis: decision making under uncertainty*, Second edition. Boca Raton: CRC Press/Taylor & Francis Group, 2019, ISBN: 978-0-429-66761-9 978-0-429-66489-2 978-0-429-66217-1.
- [194] D. Vose, *Risk Analysis: a Quantitative Guide*, eng, 3rd ed. Somerset: Wiley, 2014, OCLC: 1020026709, ISBN: 978-1-119-95947-2.
- [195] E. J. Anderson, *Business risk management: models and analysis*. Chichester, West Sussex, UK: Wiley, 2014, ISBN: 978-1-118-74936-4 978-1-118-74941-8.
- [196] JetBrains s.r.o., *Pycharm*, English. [Online]. Available: <https://www.jetbrains.com/pycharm/> (visited on 01/16/2024).
- [197] C. Stausland, B. Jensen, and N. K. Rossing, “CanSat - Teensy 3.5 En veiledning,” no, p. 150, Nov. 2018. [Online]. Available: <https://www.ntnu.no/documents/2004699/12108297/CanSat+Teensy+3.5+-+en+veiledning.pdf/b29ce49b-0868-42c6-b441-67f5d7a4c29e?t=1573659315473>.

A Appendices

A.1 CanSat Competition Requirements

These requirements are taken from ESA [116]

1. All the components of the CanSat must fit inside a standard soda can (115 mm height and 66 mm diameter), with the exception of the parachute. Radio antennas and GPS antennas can be mounted externally (on the top or bottom of the can, but not on the sides), depending on the design. N.B. The rocket payload area has 4.5 cm of space per CanSat available, along the can's axial dimension (i.e. height), which must accommodate all external elements including: parachute, parachute attachment hardware, and any antennas.
2. The antennas, transducers and other elements of the CanSat cannot extend beyond the can's diameter until it has left the launch vehicle.
3. The mass of the CanSat must be between 300 and 350 grams. CanSats that are lighter must take additional ballast with them to reach the 300 grams minimum mass limit required.
4. Explosives, detonators, pyrotechnics, and inflammable or dangerous materials are strictly forbidden. All materials used must be safe for the personnel, the equipment, and the environment. In case of doubt, Material Safety Data Sheets (MSDS) will be requested.
5. The CanSat must be powered by a battery and/or solar panels. It must be possible for the systems to be switched on for four continuous hours.
6. The battery must be easily accessible in case it has to be replaced/recharged.
7. The CanSat must have an easily accessible master power switch.
8. Inclusion of a retrieval system (beeper, radio beacon, GPS, etc.) is recommended.
9. The CanSat should have a recovery system, such as a parachute, capable of being reused after launch. It is recommended to use bright coloured fabric, which will facilitate recovery of the CanSat after landing.
10. The parachute connection must be able to withstand up to 1000 N of force. The strength of the parachute must be tested to ensure that the system will operate nominally.
11. For recovery reasons, a maximum flight time of 120 seconds is recommended. If attempting a directed landing, then a maximum of 170 seconds flight time is recommended.
12. A descent rate between 8 and 11 m/s is recommended for recovery reasons. In case of a directed landing, a lower descent rate of 6m/s is recommended.
13. The CanSat must be able to withstand an acceleration of up to 20 g.
14. The total budget of the final CanSat model should not exceed 500€. Ground Stations (GS) and any related non-flying item will not be considered in the budget.

More information regarding the penalties in case the teams exceed the stated budget can be found in the next section.

15. In the case of sponsorship, all items obtained should be specified in the budget with the actual corresponding costs on the market.
16. The CanSat must be flight-ready upon arrival at the launch campaign. A final technical inspection of the CanSats will be done by authorised personnel before launch.

A.2 CanSat Input/Output blackbox diagram and GY-91 block diagram documentation

TL|TL

On January 20th we created an blackbox diagram for Inputs and Outputs for the CanSat 162. ASE expressed their desire to keep the existing sensors and components from the 2018 version of the CanSat kit. To gain a better overview we decided to create a black box diagram that visualizes the outputs and their measurement units. This was done in an attempt to give a better understanding on what the GUI might would need to visualize and what data the ground station might need to process.

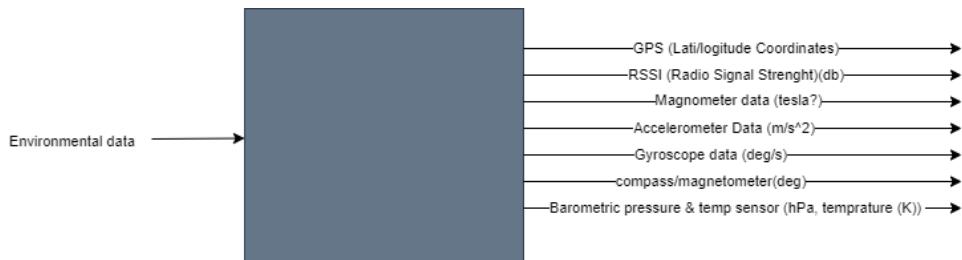


Figure 162: Black Box Diagram for Inputs and Outputs for CanSat

The components the 2018 CanSat kit uses are a RFM-96 LoRa module and a GY-91 Sensor package. The LoRa module is used for wireless communication between devices and the GY-91 is the package that contains multiple different sensors.

To again get a better overview over the GY-91 we created a block diagram over the package to show the different chips and their internal sensors 163.

To find the different chips located on the GY-91 sensor ASE has specified in the CanSat book for the 2018 kit that their GY-91 uses an MPU-9255 and a BMP-280 chip, however trying to find this package with these specific chips showed that the available GY-91 chips uses an older version called the MPU-9250 and a BMP-280. To find the differences between the old and the new chips the data sheet for both the MPU-9250 and MPU-9255, were compared by Daniel Østmoen using a PDF compare function in Adobe Acrobat, this highlights the differences between the two documents. This resulted in an unclear conclusion, but the two chips both contain the same internal chips called AK-8963 and MPU-6515. An assumption was made that the software would be identical, with possible differences in the register map. This would need to be verified when we receive the package, and can test the assumption, since at the time of writing we do not have the components at our disposal. The data sheet for both MPU-9250 and MPU-9255 revealed that the AK-8963 is a 3-Axis magnetometer and the MPU-6515 is a 3-Axis Gyroscope and 3-Axis Accelerometer.[115, pp. 5–6], [114, pp. 5–6]

The datasheet for BMP-280 revealed that this chip is a digital pressure sensor. [113, p. 2] That outputs the pressure in hectopascal, the pressure sensor can also be used to calculate the current temperature readings.

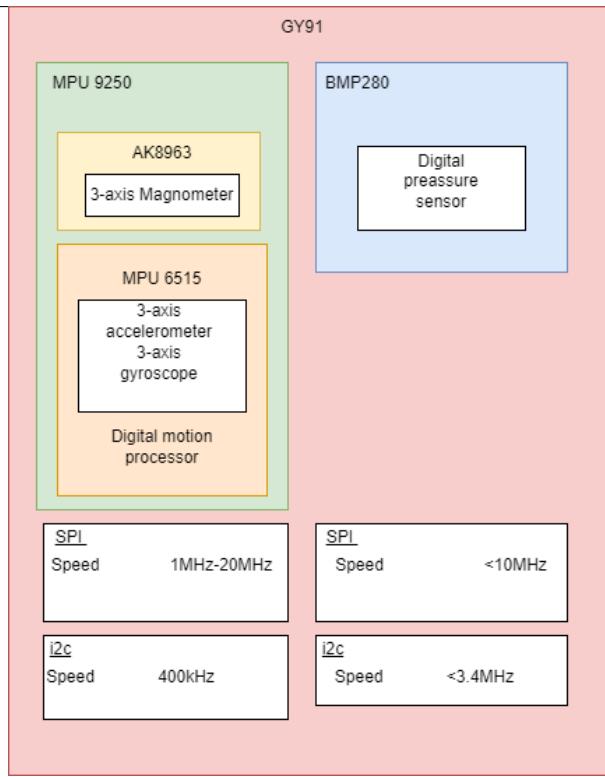


Figure 163: Block diagram for GY-91

The data sheet for the MPU-9250 and MPU-9255 does not clearly specify the output units for what the magnetometer, gyroscope and accelerometer are. By searching on the internet the sensor type and output unit the result was that, the output unit for the magnetometer was tesla, accelerometer was m/s² and gyroscope was tesla. This datasheet does however specify that the magnetometer is used as a compass, where google revealed the output is degrees, i chose to include both tesla and degrees since we might use both in the final system.

ASE also has a concerned their interest in using a GPS for plotting and recovery purposes. This has not been the main focus at this stage, but the output is still specified in the black box diagram since the plan is to implement a GPS module at a later stage. The output is latitude and longitude coordinates.

By looking at existing Arduino libraries for LoRa modules revealed that the signal strength from the radio module is given in Received Signal Strength Indicator (RSSI) where the unit is decibels.

A.3 What is LoRa?

LoRa stands for Long Range, and is a wireless communication technology designed to allow low powered, long range communication between multiple devices. LoRa is a spread spectrum modulation technique patented by Semtech [118]. Spread spectrum modulation is

LoRa is most notably used in IoT devices, allowing farther communication between devices than the typical Wi-Fi network. The networking of LoRa devices is called LoRaWAN. LoRaWAN is built upon Semtech's LoRa platform and is a standard developed by The LoRa Alliance. LoRaWAN is specifically designed to allow battery-powered devices to communicate wirelessly. [119] LoRaWAN uses the licence-free ISM band, which is portions of the radio spectrum reserved for ISM purposes. The frequency range used by LoRa and LoRaWAN is 169MHz, 433MHz (Asia), 868MHz (Europe) and 915 MHz (North America) [120]. In Norway, the 433MHz band can also be used as explained in 5.6.

A.4 CanSat

TL|TL

A CanSat is a small representation of a real satellite, which is used specifically to teach about space technology to students. The CanSat major components should fit within the space of a 330ml soda can. During its operation it is dropped from a altitude of a few hundred meters up to 1.5km, and during its descend it uses sensors to gather research data. [116] The CanSat are used in projects where students compete in teams of 3 to 6. All the students are between 14 to 19 years old. The Norwegian CanSat competition is arranged by European Space Education Resource Office (ESRO), ASE and ESA where ESA has set the CanSat requirements A.1

A.5 CubeSat

CubeSat is a nanosatellite where their size is based on a standard size called Unit (U)[117]. One U is approximate 10cm x 10cm x 11cm. CubeSat's are a versatile satellite that can be used for a lot of different space missions. They were primarily developed to provide a platform for students to learn about space technology, but they are also used by national space agencies to perform complex space missions and demonstration of new technology. The main reason for using a CubeSat over a traditional satellite is cost, versatility and scalability.

A.6 CanSat Package Diagram 1. iteration Documentation

TL|TL

On 28. January we created a package diagram for the CanSat libraries. we first created a class diagram for the same purposes, but this was not the most suited diagram to show libraries. we discovered UML's package diagram that is used to simplify class diagrams, or to display package dependencies. we made the diagram with the same structure as the class diagram Fig. 165.

The diagram starts with an actor called CanSat user. The user imports the CanSat library which is used to program the CanSat. The CanSat Library access's multiple libraries, for the radio, the GY-91 sensor package, and a GPS library.

The GY-91 library merges two other packages called AK8963 and MPU-6515. This is just to display that the sensor package does use two different chips containing multiple different sensors, in reality GY-91 is one library.

This is the 1. iteration of this design and will probably change in later sprints. This is created to give a visual representation of the initial design of the software for the CanSat.

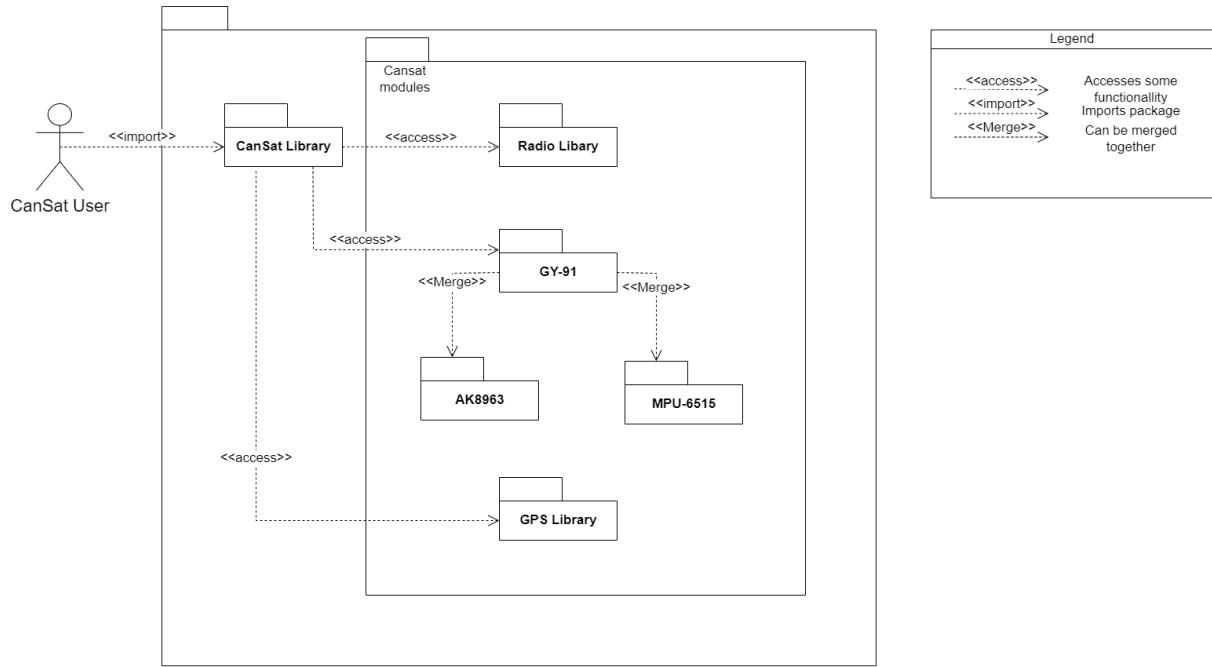


Figure 164: Package Diagram

A.7 Class diagram 1. iteration documentation

January 23. the first design for the Arduino program running on the CanSat was created. This design is an early development prototype and is not how the end product will look like, this diagram is just to get a basic overview of how the different libraries and software can be implemented.

Our proposed solution would be to have one CanSat class that the end user would see. This class would function as a wrapper for the different libraries used for the CanSat. This will be used to simplify the programming experience for the students, but will also add the ability for further extension of the software if necessary. Since this is in early development the functions and variables are not what the end user will use, it is just an example on what the final product could look like.

From the research done when creating the blackbox diagram for the CanSat and block diagram for the GY-91 in section A.2 We could use some information from different libraries that already exist to display some functions that probably will be used in the end product. ASE has, for the CanSat kit from 2018, developed a RFM-96 and a GY-91 Library. The GY-91 library is based upon a library created for the MPU-9255 and Adafruit's BMP280 library with some of their own code as well. This is documented in the header of the library and a comment .cpp file for the library. This library does have some comments, but no official documentation that I could find. The RFM96 library is only slightly commented on, with most of the comments being to the programmers writing the library. This library however does have some documentation in the form of a library command set, that instructs the end user on some of the most used functions the end user will see.

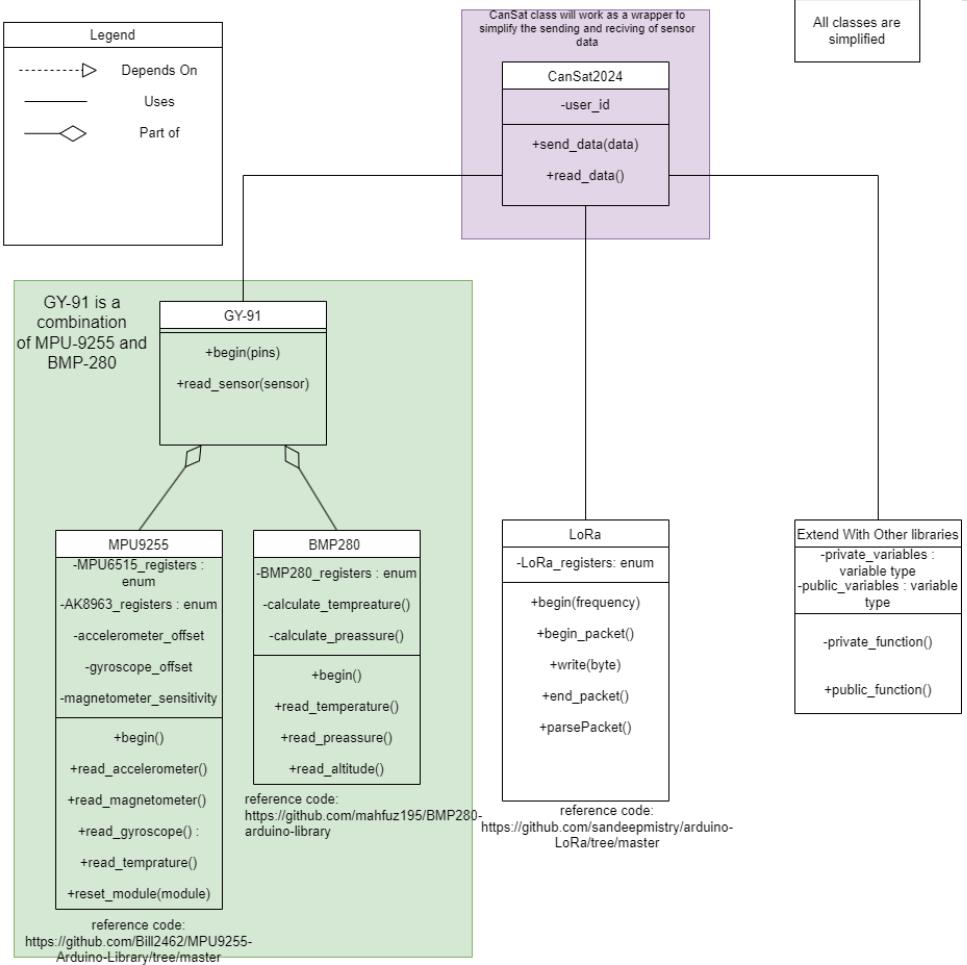


Figure 165: Class diagram

A.8 Harsh top lines in 3D print Model

On 1. February we started 3D printing the initial design for the CubeSat frame, Sebastian has designed. This design was printed to get something physical to show the audience during our 1. presentation, a decision was made, that we were going to print this at 0.32mm (max layer height for 0.4mm nozzle) to improve printing time. This model was sliced with Ultimaker Cura, but settings might apply to other slicers. During this print, the surface level became uneven with harsh lines in seemingly random places, as seen in Fig. 168. This was not a problem with the design itself, but rather with slicer settings. We continued to print with these imperfections so it would be done before our presentation. We tried to find the cause of this problem so that in later iterations for the CubeSat design this would not be a problem.

To find the cause of these lines we look at the sliced 3D model file, where these lines occurred. Cura has by default *Monotonic Top/Bottom Order*, turned off. This means the slicer does not take into consideration how the surface finish looks like and just continue printing with a pattern, that prioritizes minimum travel distance. By turning *Monotonic Top/Bottom Order* on you can decide how the top and bottom layers will look like. This setting forces the infill for the top and bottom layers to fill at a set pattern and creates a smoother and more even finishing pattern. However, turning this setting on does increase print time. Fig. 166 shows how Cura would fill inn the top surface with *Monotonic Top/Bottom order* turned off. 167 shows filling from the lower left corner to the upper right corner, with *Monotonic Top/Bottom order* turned on.

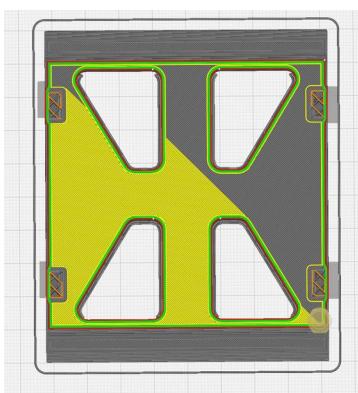


Figure 166: Monotonic order turned on

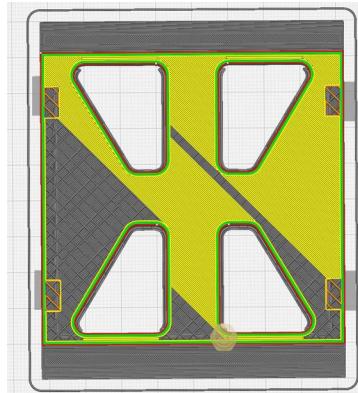


Figure 167: Monotonic order turned off

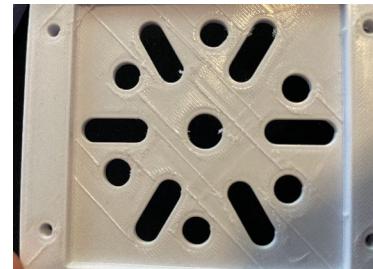


Figure 168: Harsh Top Lines

A.9 Activity diagram documentation

On 14. February an activity diagram was created to visualize what the base software and components do during a simple CanSat primary operation. The solid black node indicates the starting position. The first action is to turn the CanSat on, this is done either by the operator to collect data on the way up or with a mechanical mechanism when the CanSat is released. The first step for the software is to initialize the radio and GY-91 sensor module. The sensors then start reading data and write it to a variable. The radio module will then take this variable and write it to its buffer. This buffer is where the data is stored before it is either written to an SD card, sent over using the LoRa technology, or both. The radio and the sensor modules will continue to read and write data to the buffer until the modules are turned off. The CanSat operator will be able to view this data on the GUI.

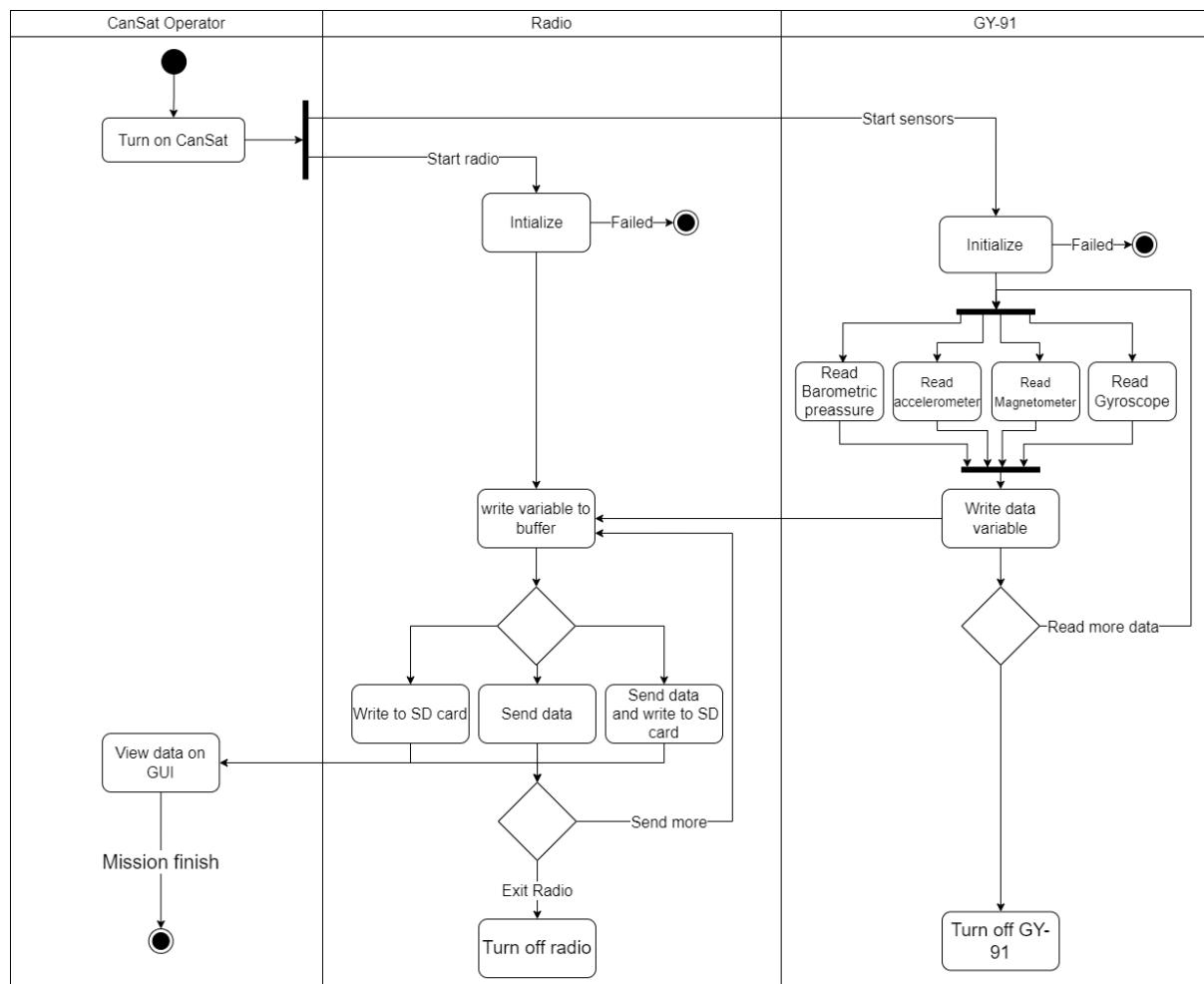


Figure 169: Activity diagram for CanSat operation

A.10 Implementing SD Card

TL|DØ

With our decision to not use ASE's RFM-96 library 5.8, we need to implement the ability to save to an SD card ourselves. The advantage to doing it this way we gain the ability to save data that are not being transmitted to the ground station as well as saving received packets to an SD card.

Saving and writing to an SD card uses the communication protocol SPI A.12. To read and write to SD cards, using Arduino like microcontrollers, Arduino has developed an official SD library that makes this process easier.

Since Teensy 4.1 has a built-in SD card reader we need to find the right pins used for SPI communication. PJRC has created a pin diagram as seen in Fig. 170. that shows this clearly. The pins used are 42,43,44,45. However, the official SD library "ReadWrite example" does state that Teensy 4.1 uses "BUILTIN_SDCARD" as the input for SD card initializing, so this is what we will use.

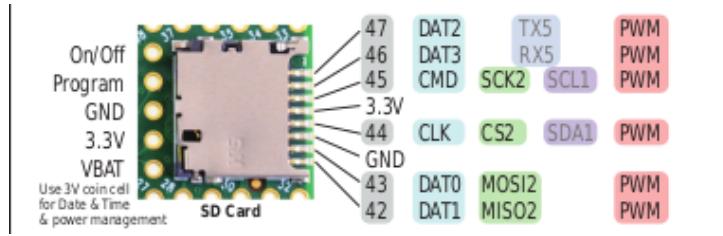


Figure 170: Teensy 4.1 Pin diagram for extra pins

For our implementation, we wanted to save all the sensor data that has been read. To do this we use the the SD library and the sensor array as discussed in 7.1. We want to save the data with the sensor name as the header. To do this we first check if the file we are trying to save exists. If it does we write the data. But if we need to create the file, we loop trough all the sensor names located in the sensor array to the file. This makes the first column of the file the sensor name.

Since we are using the sensor array we know that the name index is the same as the data index, so we can use a *for loop* to loop through the sensor array and write the data and it will be placed under the right header.

A.11 GPS implementation

TL|DØ

To implement a GPS with a microcontroller there are a few questions to ask before having the necessary information;

1. What communication protocol does the GPS use?
2. Do you have any available pins for the protocol?
3. What sort of library do you need to use the GPS?

Many of the popular GPS modules use the UART communication protocol, like the *NEO-6M* or *Adafruit Ultimate GPS*.

These modules do not need a library or any code to make it work with a microcontroller, but it makes the programming experience much easier. In this case, the GPS will send raw GPS data to the microcontroller and the programmer can do what they want to do with this data. A library for a GPS using UART, takes this raw data and encapsulates it with easy-to-use functions. This makes it easy to read the data you care about, without seeing the data you do not care about.

When using a GPS module with a SPI or I²c communication protocols, you do need a library, as a minimum you would need the official SPI library from Arduino or the official Wire library from Arduino for I²c connection. However, using these low-level libraries requires significant experience in programming modules. For the most common GPS modules like the *Adafruit MINI GPS PA1010D*, there exist libraries written specifically for the GPS module, making communicating using SPI or I²c much easier.

Since a GPS module is not used in a primary CanSat mission. We have not included a GPS implementation to the library. What we have done is implemented a way to send "custom" data to the ground station, enabling the sending of any sensors the user may want to use. An example of sending custom GPS data can be seen in section A.15.1.

A.12 Communication Protocols

When implementing sensors or other modules with a microcontroller it is valuable to know a bit about different protocols used for communication between these devices. In our project the different protocols used are UART, SPI and I²c. These are all common and widely used protocols in micro controller programming. This section will aims to give an introduction to these protocols, to give a brief overview over how the communication works in hardware, and the different names associated with the the corresponding protocol.

A.12.1 Universal asynchronous receiver-transmitter

UART is communication protocol used in micro controllers and computers, and one of the communication protocols used in C.R.A.F.T-Sat. UART is one of the most widely used hardware communication protocols [121]. Many sensors used in C.R.A.F.T-Sat can be used with the UART protocol, in our product our GPS is connect through UART.

How UART works: UART's main advantage is it's simplicity. To form a connection between two devices you need to wires, one transmit wire and one receive wire, often labeled RX and TX. These wires are connected to opposite pins as seen in Fig. 171. When a transmitter is ready to send data to the receiver, a so called *start bit* is sent. Then a collection of bits in the range of 5 to 9 bits, often 8 bits (1 byte) is sent, the collection of bits are the actual data the device wants to send. When the data is finished transmitting a *stop bit* is sent indicating the transmission is finished [121]. To put this into perspective, the letter "A" takes exactly 8 bits of space. Sending multiple lines of text over UART will therefore repeat the steps listed above for each number of characters there are in those lines of text.

Fig.172 Shows how this is done over the wire. The receiver knows that a start bit is coming by always keeping the wire at a high voltage by the transmitter, when the voltage in the wire drops to 0, this is when the data is being sent.

The rate of how fast the data is being sent is called the baud-rate, this is what you might be familiar with if you have programmed the Teensy or any Arduino micro controller, and used the Serial.begin(9600) command. 9600 is the baud rate used in this example, the number means means *number of bits per second*. When setting the baud-rate you tell the micro controller to use its internal clock to generate a pulse that matches the send speed, likewise when you open the serial monitor in the Arduino IDE, your telling the computer to use its internal clock to receive bits at the required speed.

The main disadvantage to UART is the use of pins on a micro contorller. Each module using this protocol will use 2 pins each. With a limited amount of pins you cannot run many modules before running out. However one workaround to this problem is to use Arduino official SoftwareSerial library, making it possible to use any digital pins for UART communication.

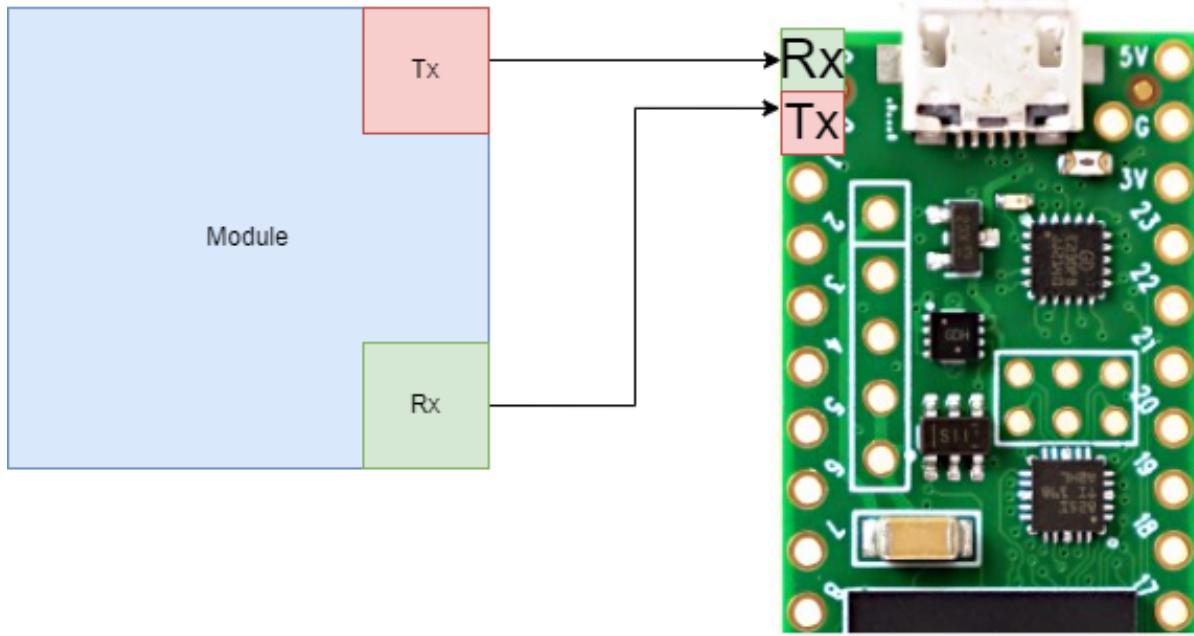


Figure 171: UART Connection to Teensy 4.1

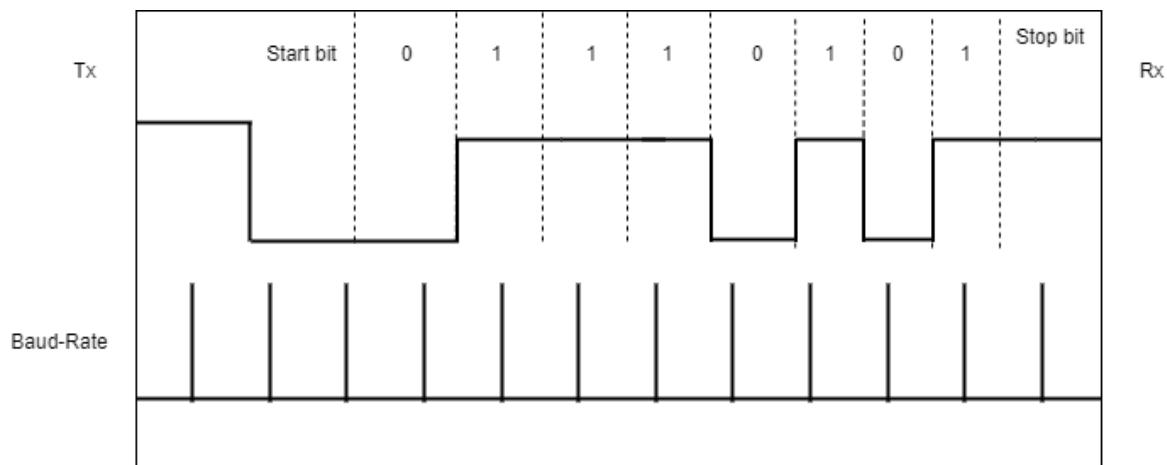


Figure 172: Sending of the letter "u" using UART

A.12.2 Serial peripheral interface

SPI is another communication protocols used in computers and micro controllers. In C.R.A.F.T-Sat the RFM-96 LoRa radio uses SPI to communicate with the Teensy 4.1, aswell as the in built SD-card reader.

How SPI works works: SPI works on the master-slave principle. The master is the main device giving instructions to the slave, in our case the master is the Teensy 4.1 and the slave is the RFM-96. SPI uses 4 pins to communicate, these are called: [122]

- MOSI (Master output, Slave input)
- MISO (Master input, Slave Output)

-
- SCLK (Source Clock)
 - CS/SS (Chip select/Slave select)

The MOSI and MISO Pins are where the data between the devices are being sent, while the SCLK synchronizes the sending, meaning while the data transfer is ongoing, each clock cycle one bit is passed through the MOSI or MISO pin. Similarly to UART A.12.1, the CS pin are kept high all the time. When the master wants to transfer data to or from the slave, the CS is turned low indicating to the slave the master is ready to receive data [122]. One feature of SPI is that you can connect multiple slaves to the same SPI bus, only differentiating them with the CS pin, so if you want four sensors connected through SPI you need MOSI, MISO, SCLK, and 4 CS pins.

When devices communicate using SPI, 5 steps happen:

1. Master sends out SCLK signal.
2. CS turns low.
3. Master sends out bits.
4. Slave responds.
5. CS turns high indicating communication is finished.

The main advantages for SPI speed and data length. SPI generally offers greater speeds than both UART and I²C. The protocol is also not restrained by a set packet length. As long as the CS pin is turned low the communication can be as short or as long as the devices want. The protocol also allows for "serial" communication meaning devices can send and receive data at the same time.

The main disadvantage is that if you want to only connect one device to SPI, you need four wires compared to UART and I²C which uses two. [122] More information and more indept examples can be found on: Circuit-Basics

A.12.3 Inter-Integrated Circuit

I²C is another communication protocol, used in micro controllers and computers and is the last communication protocols used in C.R.A.F.T-Sat. Our C.R.A.F.T-Sat uses I²C to communicate with the sensor module GY-91.

How I²C works: I²C works on the master and slave principle like SPI. Compared to SPI, I²C can use multiple masters, aswell as multiple slaves. In our case the Teensy 4.1 is the master and the sensors within the GY-91 is the slaves. I²C uses two wires to communicate, these are called SDA (Serial data) and SCLK (Serial clock). Like UART A.12.1 data are sent in packets. These packets however are more complicated, containing a start condition, a stop condition, an 7-10 bit address, a read/write bit, an ACK/NACK (Acknowledge/Negative Acknowledge) bit, and the data itself.

The start condition indicates data is about to be transferred. This works by switching the wire from high to low, during a clock cycle over the SCLK wire. Similarly stop condition is when SDA wire goes from low to high during a clock cycle indicating the data is

finished. I²c protocol separates different slaves with the use of addressing. Like a home address, chips have a unique address that separates where they live in the circuit. The read/write bit indicates if the master wants to receive data from the slave or send data to the slave.

The ACK/NACK bit stand for acknowledge and negative acknowledge. This is used for data checking. When data is received from one device, it returns an acknowledge if the receive was successful. Similarly to UART the data is sent in 8 bit segments, right after these 8 bits are sent an ACK/NACK bit is sent back to the sender, indicating if the data arrived correctly. The sender must receive an ACK before the next 8 bit segments is sent.

The biggest advantage to I²c protocol is the addressing scheme. This allows the use of only 2 pins on the microcontroller, with a 7 bit address you can have up to 128 slaves connected to two pins, and with a 10 bit address you can have up to 1024 slaves. Compared to both UART and SPI, this protocol is the only protocol that has error checking, in the form of ACK/NACK creating a more reliable communication. The downside to using I²c is that the speed is much slower compared to SPI since the data is split into 8 bits segment and sending ACK/NACK bit after each segment creating much more traffic. For more information about this visit: [Circuit-Basics](#)

A.13 LoRa Packet

A Lora packet contains three elements, A Preamble, An optional header and data payload. [91] When the optional header is included it is called explicit mode, when it is excluded it is called implicit mode. The structure of a lora packet can be seen in Fig. 173.

- Preamble
 - Is used to synchronize the receiver with the incoming data
 - Symbols represent bits as tones
 - Contains $8 + 4.25$ symbols by default but is variable from $6 + 4.25$ to $65535 + 4.25$ symbols
 - A longer preamble improve synchronization, but increase transmission time
- Header
 - Explicit header
 - * Contains information about the payload
 - * Information about the coding rate
 - * an optional 16-bits of cyclic redundancy check for the payload
 - Implicit Header If you wish to use a spreading factor of 6, implicit header is required. [91]
 - * Reduces transmission time by removing the header
 - * Requires manual configuration on transmitter and receiver
 - *
- Payload contains the actual data sent by the receiver. Is variable in size from 0-255 bytes.

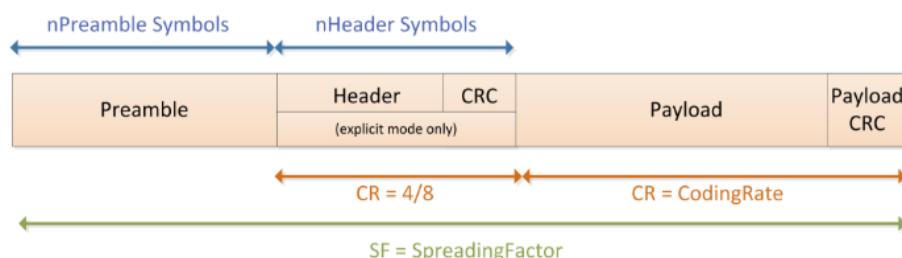


Figure 173: LoRa Packet

A.14 Using the C.R.A.F.T-Sat Library

The C.R.A.F.T-Sat library is a library used for both CanSat and C.R.A.F.T-Sat missions. The library gives an easy-to-use interface for developers with different experience levels. This section will explain how to use the library and its included functionality.

A.14.1 terminology

This section will use some terminology that will be necessary to understand to program your satellite.

- Library
 - A library contains already defined functions that allow users to use functionality without needing to know how it works behind the scenes.
- Object
 - An object is an instance of a class or library for using the functions defined in the library.
- Serial Monitor
 - The serial monitor is used to read outputs from the microcontroller on your computer. To output information you use `Serial.print("output")` or `Serial.println("output")`.
- Comments
 - All lines beginning with `//` is comments. This will be ignored by the microcontroller and are used for explanation to you as the user.
- void setup & void loop
 - void setup will run once when the microcontroller gets power. This is used to run code that only need to run once, like begin functions. The void loop will run indefinitely, you put the code that need to run often inside the loop, like reading sensors and sending data.

A.15 C.R.A.F.T-Sat library User guide

Include library

Before you can use the library you need to include it in your Arduino project. This is done at the top of your project with the command below.

```

1 #include <craft_sat.h>
2
3 void setup()
4 {
5

```

```
6 }
7
8 void loop()
9 {
10
11 }
```

Instantiate Object

Before you can use any functionality within the library you will need to instantiate an object of the library. This is done by using **Craft_sat object_name**. The object name can be whatever you want, we have decided to call the satellite, *sat* and the ground station, *gs*. It is a good idea to call it something relevant and easy to type.

The instantiating is done differently for the satellite and the ground station since the ground station does not use any sensors. There are also a third option which you can enable/disable any standard sensors you want. This is done by entering *true* or *false* within the parentheses. Examples of this can be found in Section A.15.1.

```
1
2 #include <craft_sat.h>
3
4 Craft_sat sat;
5 Craft_sat gs(backup); //Set backup to true or false if you want
   to save receiving data
6 Craft_sat gs(radio, ntc, BMP280, MPU9255, backup, file_name); // set
   sensors to true or false
7
8 void setup()
9 {
10
11 }
12
13 void loop()
14 {
15
16 }
```

Start sensors and modules

In the setup part of your Arduino project you will need to use the *begin* function. This will initialize all the sensors and modules used for your C.R.A.F.T-sat shield. You will need to input the radio frequency given out by ASE or teacher in both your satellite and ground station to make sure they communicate on the same frequency. Since we are using an RFM-96 G.3.6 the frequency will be within 433.050MHz and 434.790 MHz. *frequency* in the example below will need to be changed with a frequency in the range of 433050000-434790000

```
1 if (!sat.begin(freq)) //Change out frequency
2 {
3     Serial.println("Something is wrong: See output");
4     while(true);
5 }
```

A good practice with most libraries using a *begin* function is to put it within a test, to see if the modules start correctly. The *begin* function returns true if everything started successfully and false if something went wrong.

The "!" before the function is called the NOT operator and returns the opposite of what the function returns. The *if* test enters the code within the curly brackets, **if** whatever inside the test is true. The *while(true)* simply means loop indefinitely and do nothing. If something is wrong you will get an output indicating what might be wrong and fixes to try.

Read sensors

To be able to use the sensors located on the C.R.A.F.T Shield you must use any of these functions. In most cases these functions should be put within the loop for continuous readings.

```
1 sat.read_all();
2 sat.read_GY91();
3 sat.read_ntc();
4 sat.read_accelerometer();
5 sat.read_gyroscope();
6 sat.read_magnetometer();
7 sat.read_pressure();
8 sat.read_mpu_temperature();
9 sat.read_bmp_temperature();
```

sat.read_all() does as the name suggests and read all sensors located on the C.R.A.F.T shield. You can also use the individual functions to read individual sensors or modules.

Send sensor data

To send sensor data to the ground station you use the function located below. You need to read data before you can send anything.

```
1 sat.send_sensor_data();
```

Receive

To setup the ground station is made to be as simple as possible using the *receive()* function. It is important to begin the serial monitor before you can receive anything.

```
1 gs.receive();
```

Remote control

To setup remote control of the satellite you need to enable it in both the ground station and satellite. On the ground station you need input **true** into the receive function. To accept incoming messages on the satellite you need to use the Allow_remote_control function.

```
1     receive(true); //enables remote control capability.  
1     allow_remote_control(); //enables remote control capability.
```

To send commands to the satellite you input these commands into the serial monitor.

A.15.1 Example programs

```
1 #include <craft_sat.h>  
2  
3 Craft_sat sat; //Instantiate object called sat, no input means  
    all sensors are active  
4 void setup()  
5 {  
6     Serial.begin(9600); //For debug output  
7     if(!sat.begin(433500000)) //start with radio frequency 433.5  
        Mhz  
8     {  
9         Serial.print("Something went wrong"); //debug output  
10        while(true);  
11    }  
12 }  
13  
14 void loop() {  
15     sat.read_all_sensors(); //read all sensors, must be called  
        before sending data  
16     sat.send_sensor_data(); //send sensor data to groundstation  
17     sat.save_data(); //save data to SD card  
18     //Seems like nothing happened? Flash  
        Basic_Ground_Station_Setup to your groundstation and open the  
        serial monitor to see what the satelitte is sending  
19 }
```

Listing 4: Simple C.R.A.F.T-Sat program

```
1 #include <craft_sat.h>  
2  
3 Craft_sat gs(true); //Turn on SD Card saving  
4 void setup() {  
5     Serial.begin(9600);
```

```

6 if (!gs.begin(433500000)) //Start radio with 433.5Mhz
7 {
8     Serial.println("Something went wrong!");
9     while(true);
10 }
11 }
12 void loop() {
13     gs.receive(); //receive data from satellite
14 }
```

Listing 5: Simple Ground Station program

```

1 #include <craft_sat.h>
2
3 Craft_sat sat;
4 void setup() {
5     Serial.begin(9600);
6     if (!sat.begin(433500000)) //Start radio with 433.5Mhz
7     {
8         Serial.println("Something went wrong!");
9         while(true);
10    }
11    sat.set_accelerometer_scale(scale_16g);
12 }
13
14 void loop() {
15     sat.read_all_sensors();           //read all sensors on the C.R.A.F
16     .T-Sat shield
17     sat.send_sensor_data();          //Send sensor data to ground
18     station
19     sat.allow_remote_control();      //Allow satellite to be controlled
20     by the ground station
21 }
```

Listing 6: Satellite that allows incoming commands

```

1 #include <craft_sat.h>
2
3 Craft_sat gs(false); //False = no SD card
4 void setup() {
5     Serial.begin(9600);
6     if (!gs.begin(433500000)) //Start radio with 433.5Mhz
7     {
8         Serial.println("Something went wrong!");
9         while(true);
10    }
11 }
12 void loop() {
```

```
13     gs.receive(true); //True = Remote control enabled  
14 }
```

Listing 7: Ground Station with Uplink (send commands to satellite)

```
1 #include <craft_sat.h>  
2 #include <TinyGPSPlus.h>  
3  
4 Craft_sat sat; //Satellite with all sensors attached  
5 TinyGPSPlus gps;  
6  
7 double interval = 1000; //GPS reading interval  
8  
9 //indexes for custom data in sensor array  
10 int latitude_index;  
11 int longitude_index;  
12 int time_index;  
13 int kmph_index;  
14  
15 void setup() {  
16     Serial.begin(115200);  
17  
18     //Serial1 = UART bus 1, pin 0 and 1 on teensy 4.1  
19     Serial1.begin(9600); //Swap out 9600 for the baudrate of your  
      GPS  
20  
21     if (!sat.begin(433500000)) //Start radio with 433.5Mhz  
22     {  
23         Serial.println("Something went wrong!");  
24         while(true);  
25     }  
26     //Add GPS ID and name to Sensor array  
27     latitude_index = sat.add_sensor(123, "Latitude"); //ID 123  
      with sensor name "Latitude"  
28     longitude_index = sat.add_sensor(124, "Longitude"); //ID 124  
      with sensor name "Longitude"  
29     time_index = sat.add_sensor(125, "Time"); //ID 125  
      with sensor name "Time"  
30     kmph_index = sat.add_sensor(126, "Kmph"); //ID 126  
      with sensor name "Kmph"  
31 }  
32  
33 void loop() {  
34     double start = millis();  
35     do  
36     {  
37         while (Serial1.available())
```

```

38     gps.encode(Serial1.read());
39 }while (millis() - start < interval);
40
41 sat.read_all_sensors(18); //read all sensors with 18 as
   temperature offset for NTC sensor
42 //add GPS data to sensor array.
43 sat.add_sensor_data(latitude_index, gps.location.lat()); // 
   add latitude to sensor array
44 sat.add_sensor_data(longitude_index, gps.location.lng()); // 
   add longitude to sensor array
45 sat.add_sensor_data(time_index, gps.time.value()); // 
   add time to sensor array
46 sat.add_sensor_data(kmph_index, gps.speed.kmph()); // 
   add kmph to sensor array
47 sat.allow_remote_control(); // 
   Allow remote control of C.R.A.F.T-Sat
48 sat.send_sensor_data(); // 
   Send data to ground station
49 //No need for delay since send_sensor_data adds time on air as
   delay
50 }

```

Listing 8: Sending custom GPS data

A.16 Library Installation Guide

The C.R.A.F.T-Sat library is developed to gather all the individual components on the C.R.A.F.T-Sat shield into an entire system. The library is a wrapper for other harder to use libraries, to make the programming of the shield easier. Before you can use the C.R.A.F.T-Sat library you need to import some other files it depends on.

The library dependencies can be seen in Fig. 174. If you are using the Arduino IDE to program your C.R.A.F.T-Sat shield which is highly recommended, you do not need to download the dependencies in blue in Fig 174. This guide assumes you have already installed the "Teensyduino" add-on.

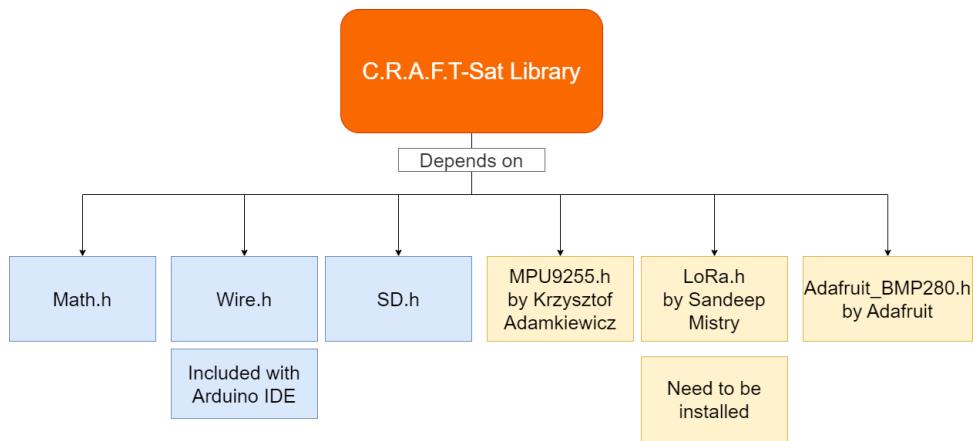


Figure 174: C.R.A.F.T-Sat library dependencies

On the left side of the Arduino IDE, click on the "*library manager*" icon. Then search for "*adafruit_bmp280*" and click install as seen in Fig. 175. The current version of the C.R.A.F.T-Sat library uses Version 2.6.8.

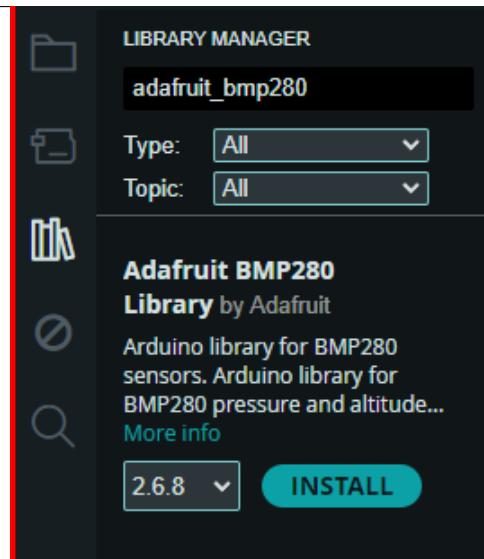


Figure 175: BMP280 Library by Adafruit

For the next two libraries you would have to download the libraries from GitHub, and import it manually. The LoRa library can be found in the Arduino Library manager, but it does not have the latest commit which enables *Low Data rate optimization* which we use to calculate "Time on air" 7.2. To access these libraries go to <https://github.com/sandeepmistry/arduino-LoRa/> and <https://github.com/Bill2462/MPU9255-Arduino-Library>, press *Code* and *Download ZIP*, as seen in Fig. 176

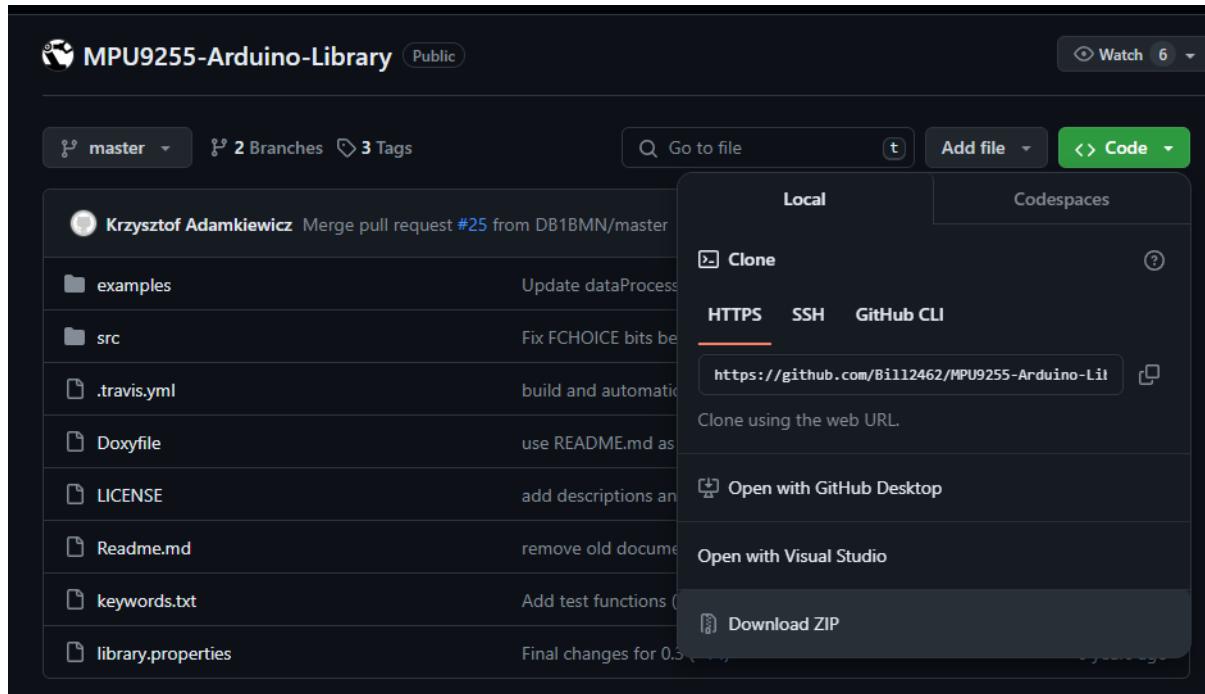


Figure 176: Download MPU9255 library from GitHub

Open the Arduino IDE and in the top left bar press, *Sketch->Include Library->Add .ZIP*

Library and select the .zip files downloaded from GitHub, as seen in Fig.177. Repeat this procedure for the MPU9255 Library, the LoRa Library, and the C.R.A.F.T-Sat library.

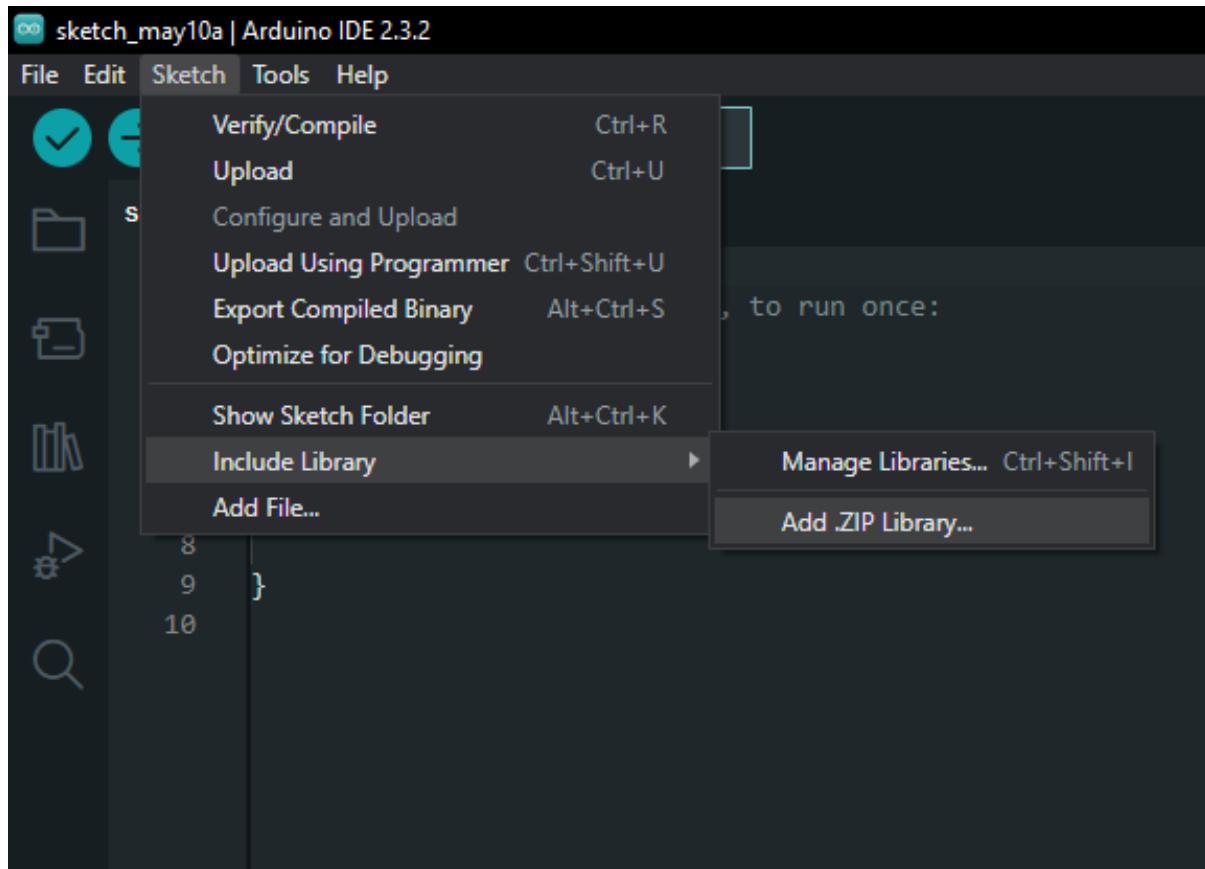


Figure 177: Include libraries in the Arduino IDE

A.17 C.R.A.F.T-Sat Library API

Constructors

The library uses 3 constructor. The default constructor sets all the sensors on the C.R.A.F.T-Sat shield to active.

The full control constructor gives the ability to turn on and off all the sensors and set a custom name for the sensor data.

the ground station constructor activates the radio and gives the ability to turn on and off backup to SD card, by default this is turned off;

```
1 craft_sat sat; //default
2 craft_sat sat(radio, ntc, bmp280, mpu9255, backup, filename); //Full
control
3 craft_sat gs(backup); //groundstation
```

begin

The begin function starts all the sensors and sets the right settings for the radio.

Allowed frequencies: All SX127X frequencies

Allowed Transmission power: 2-20. Always uses PA Boost. Defaults to 10

Allowed bandwidths: 7.8E3, 10.4E3, 20.8E3, 31.25E3, 41.7E3, 62.5E3 (Default), 125E3, and 250E3

Allowed spreading factors: 6-12

Allowed coding rates: 5-8, default 5.

```
1 begin(frequency, TX_power, bandwidth, spreading_factor,
coding_rate);
```

Sensor array

Used to hold sensor information. Contains sensor name, sensor id, and sensor data. All standard sensor modules get added automatically. ID is used to differentiate sensors at groundstation/GUI.

Adding sensors

Add non standard sensor to sensor array. Returns index of sensor array where new sensor is located. ID is used in the GUI to differentiate sensor values. Name is used when saving to SD card.

```
1 add_sensor(id, name);
```

Adding sensor data

Add sensor data to sensor array.

```
1 add_sensor_data(index, data);
```

Print sensors

Prints out all data located in the sensor array to the serial monitor.

```
1     print_sensor();
```

Print sensors

Returns the index of where the sensor with Input id is located. Returns -1 if not found.

```
1     get_index(id);
```

Send sensor data

Send sensor data located in the sensor array. *Header* indicates implicit (false) or explicit header (true), default is explicit. *Start* indicates which index of the sensor array to start from, defaults to 0 and is used if packet is larger than 255 bytes. Sensors need to be read before anything is sent.

```
1     send_sensor_data(header, start);
```

Receive

Receives data and outputs to the serial monitor for the GUI to read. remote control is used for sending commands to the satellite remotely. Defaults to false. See Remote control for more information.

```
1     receive(remote_control);
```

Save data

Saves sensor data located in the sensor array to SD card. Must have backup enabled (see constructor). Creates name as headers when file is created. File name is set with the full control constructor. Default name is "Sensor_data.csv".

```
1     save_sensor_data();
```

Read sensors

These functions are used to read temperature and add them to the sensor array.

```
1     read_all_sensors();
2     read_GY91();
3     read_accelerometer();
4     read_gyroscope();
5     read_magnetometer();
6     read_mpu_temperature(); // returns temperature
7     read_bmp_temperature(); // returns temperature
8     read_ntc();           // returns temperature
9     read_pressure();      // returns pressure
```

Temperature alarm

Used to activate the buzzer when a certain temperature is above threshold. Reverse Input and max temperature to use alarm for negative degrees.

```
1     temperature_alarm(input_temperature, max_temperature);
```

Advanced features

Advanced features are functions that give more control over the radio and sensors. Use with caution.

Remote control

Allows the satellite to be controlled via the ground station by inputting commands into the serial monitor. Commands follow this structure, "COMMAND, VALUE;". Implemented commands:

BEEP, Frequency

Turns the buzzer on for 10 seconds with the frequency provided.

RST

Resets all the sensor modules if no value is specified. Allowed values:

- ACC = Accelerometer
- GYR = Gyroscope
- MAG = Magnetometer
- TEMP = MPU9255 Temperature
- SIG = MPU9255
- BMP = BMP280

BAND, Frequency

Sets the radio bandwidth on the satellite. For allowed frequencies see begin function

FREQ, Frequency

Sets the radio frequency on the satellite. All SX127X frequencies allowed.

TXP, power

Sets the radio transmission power on the satellite. Allowed values: 2-20dBm.

SF, spreading factor

Sets the radio spreading factor on the satellite. Allowed values: 6-12

CR, coding rate

Sets the radio coding rate on the satellite. Allowed values: 5-8

```
1     allow_remote_control();
```

Radio settings

Functions are used to adjust the radio settings.

```
1     use_CRC(bool);                                // affects Time on
      air
2     use_low_data_rate(bool);                      // affects Time on
      air
3     set_preamble_length(preamble_length);        // affects Time on
      air
4     set_bandwidth(frequency);                    // affects Time on
      air
5     set_spreading_factor(spreading_factor); // affects Time on
      air
```

```
6     set_coding_rate( rate );                                // affects Time on
    air
7     set_TX_power( power );
```

Radio settings

Used to adjust the scale of the accelerometer and gyroscope scale in the sensor array

```
1     set_accelerometer_scale( scale );
2     set_gyroscope_scale( scale );
```

Miscellaneous

Private functions used by other public functions **Time on Air**

Used for calculating "Time on air". Returns the calculated sending time in milliseconds with current payload size. Used for honoring duty cycles. Default duty cycle is 10%.

```
1     calculate_time_on_air();
```

Processing

Used by functions for reading MPU9255. Turns raw data into human readable data.

```
1     process_acceleration( input , scale ); //outputs m/s^2
2     process_angular_velocity( input , scale ); //outputs deg/s
3     process_magnetic_flux( input , sensitivty ); //outputs microT
```

Accept command

Used for accept incoming command from the serial monitor and send it to the satellite.
Used in the receive function.

```
1     accept_command();
```

Read to send

Used to check if wait time of duty cycle is over. Returns true if the transmitter is allowed to send

```
1     ready_to_send();
```

A.18 ESP32 Camera Experiment

TL|TL

As mentioned in section 1.1.1, ASE want the ability to connect a camera to the C.R.A.F.T-Sat. We wanted to explore the possibility to connect a camera and transfer pictures or videos to the Teensy 4.1 microcontroller. By enabling this feature we could lay the groundwork for image and video handling, making it possible to further transfer or process images on the microcontroller.

We decided to try out this experiment with an ESP32 Camera since the ESP32 is in itself a microcontroller developed by Espressif Systems. Communication between microcontrollers can also be relatively uncomplicated, with the use of UART, SPI, or I²C A.12. The camera is also widely used in many different projects, and is quite common and readily accessible so it answers our Stakeholder requirement 6b 3.1.

The first task we needed to do, was test out the ESP32 and see if it worked as expected. Problems occurred when trying to upload a program to the chip. The ESP32 camera does not have a USB connector, meaning we required an FTDI adapter [125]. We could also use another microcontroller to program the ESP32[125].

We went with the second option since we have microcontrollers at our disposal. We first tried to connect the ESP32 camera with a Teensy 4.1 as seen in fig 178. This did not work, so we tried to use an Arduino Mega 2560 which did not work either. The problem is that the ESP32 Camera does not get programmed by the microcontroller, but the microcontroller act as a USB to serial convertor for the ESP32. Neither Teensy 4.1 or Arduino Mega 2560 has this capability, but the Arduino Uno does, as shown in Fig.178. We borrowed an Arduino Uno from USN and manged to upload a test program to the camera and confirmed the ESP32 camera worked.

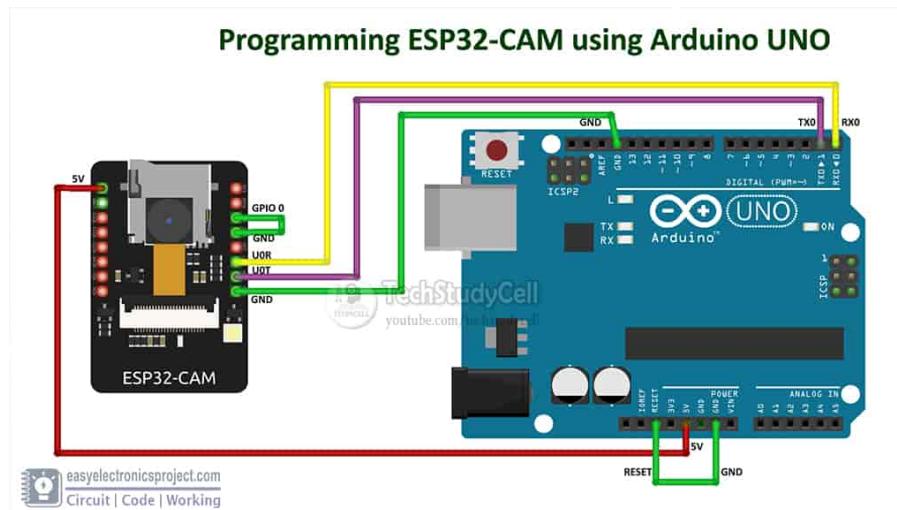


Figure 178: ESP32 Camera connection to microcontroller [125]

The second task was to transfer a picture from the camera to the Teensy 4.1. Most we project we could find either used the cameras built-in SD card reader or the built in WiFi

module to save or send pictures. We instead wanted to use the ESP32's hardware UART, SPI or I²c pins to transfer the picture to the Teensy 4.1 so it could have the possibility to send the picture via the radio.

Even though the ESP32 has all the aforementioned communication protocols available. The ESP32 camera does not have breakout for these pins, making them inaccessible. The camera does have 10 breakout pins, but all of them, except one is not safe to use. as seen in Fig. 179. The only pin that is entirely safe to use is an Rx pin for UART communication, but without an available TX pin, communication with this UART bus is impossible.

The yellow warning symbols in fig 179, are breakout for the SD card. Since SD cards use

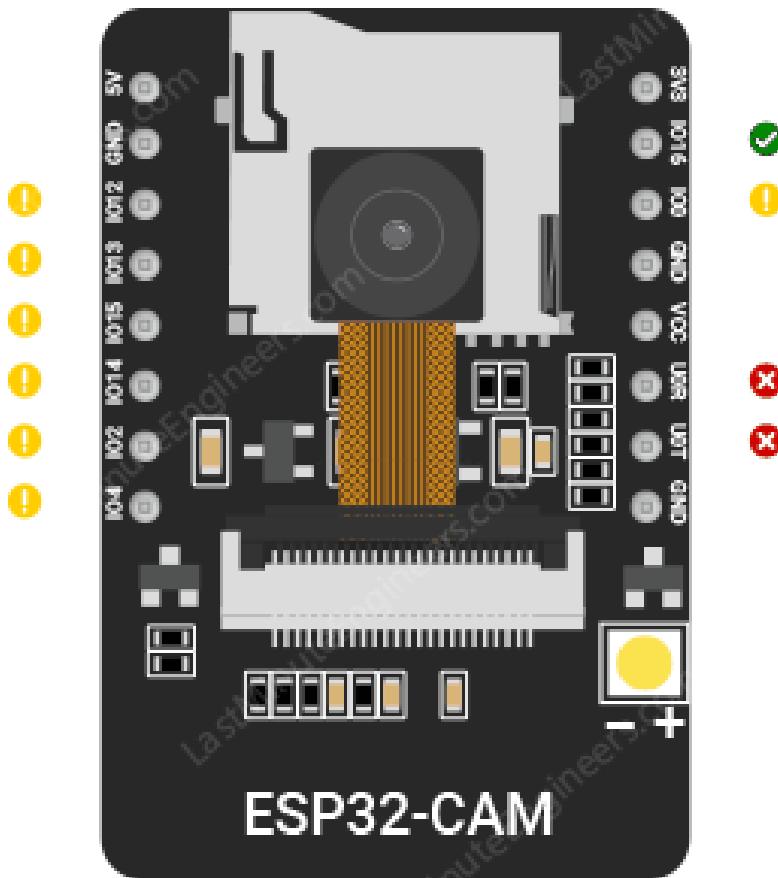


Figure 179: ESP32 Camera pins [126]

SPI communication A.10, an attempt was made to use the SPI capable pins to transfer the image. The problem with this strategy was that the ESP32 does not directly support SPI slave mode [127]. With the use of a new library, limited time left in the project, and no examples to build upon. We did not have the capacity to continue with this solution. A last attempt was made to use the UART "red" UART pins in fig 179. By using "Serial.write" function to write the contents of the image to the pins. These pins also output debug messages for troubleshooting, so they are already in use. With the last attempt, we were able to transfer the picture to the Teensy 4.1 and save the contents to the SD card located on the Teensy 4.1. This method is slow and requires manual parsing of the data to remove any debug messages. We did not manage to get any usable

information from the picture data. By comparing a downloaded JPEG image from the internet and a jpeg image captured by the camera and transferred to the Teensy 4.1, then saved to an SD card, we found out that characters nonprintable ASCII characters are used in a normal jpeg image. When transferring these characters using UART the nonprintable characters resulted in a "?" which made the image unreadable.

To conclude the experiment, multiple solutions were tried to connect the ESP32 camera to the Teensy 4.1 to give the ability to transfer pictures or videos to the ground station. None of the different solutions was successful, and options are limited using the ESP32 camera, because of the lack of breakout pins located on the module.

On the other hand, the ESP32 is a powerful microprocessor on its own, so by using the built-in SD card reader, it is possible to save pictures locally to the module. The microcontroller also has a built-in WiFi module, which is capable of transmitting pictures or videos to another WiFi module, making remote picture or video transfer possible if a long-range transmitter with WiFi capabilities is used.

A.19 CanSat frame documentation

To physically show our audience at the last presentation and USN Expo what a CanSat is, we decided to design and produce a CanSat frame, where we can install the C.R.A.F.T-Sat shield. The frame was developed using Fusion 360, and is designed to be produced by a FDM 3D printer without supports. The frame is made up of two parts, making it simple to put together. This also makes it easier to reproduce by students, if they wish to use it for their project. The frame uses 3D printable M3 threads and two M3x10-12mm button head screws to attach the top part to the bottom part. To make lining up the top part to the bottom part easier, we added "Pegs". These are not meant to hold the parts together but act as an aligner. This can be seen in Fig 180.

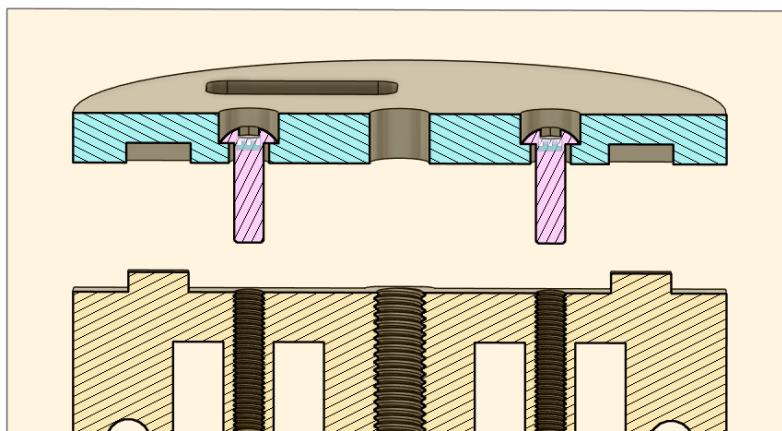


Figure 180: Section analysis of top and bottom parts

We decided to use a M5 eyebolt for the parachute mount. This is also attached using 3D printable threads. The parachute mount also attaches the soda can to the frame with an M5 nut, making the can physically connected to the frame as seen in Fig REF BILDE. The idea came from the CanSat book made by ASE and NTNU [197]. The C.R.A.F.T-Sat shield is attached to the frame using M3 threaded fitting inserts and should fit most M3 inserts under 10mm in length and bigger than 4.5mm in width. The reason for using inserts for the shield is they are stronger than 3D printable threads and offer a more secure attachment to the frame [124].

The 1. requirement from the CanSat competition requirements A.1, states that radio antennas and GPS antennas can be mounted externally. Therefore we have a hole for an antenna with an SMA connector to stick out the bottom. The bottom also has room to mount a ceramic GPS patch antenna as seen in Fig REF KOMMER

A.20 Relevant CubeSat Design Specification (1U – 12U) REV 14.1 CP-CDS-R14.1.

2.1.1 All parts shall remain attached to the CubeSats during launch, ejection, and operation. [169]

2.2.1 The CubeSat shall use the coordinate system as defined in fig. 181. The origin of the CubeSat coordinate system is located at the geometric center of the CubeSat.[169]

2.2.1.2 Note: The standoff length dimension [“(0.5 – 0.7) 0.1 MIN +/- Z Faces”], specified in fig 181, exist to prevent interfaces with potential neighbouring CubeSats and dispenser interfaces.[169]

2.2.2 The -Z face of the CubeSat will be inserted first into the dispenser.[169]

2.2.3 No components on the yellow shaded sides (fig 181) shall protrude farther than 6.5 mm normal to the surfaces from the plane of the rail. [169]

2.2.4 Deployable shall be constrained by the CubeSat, not the dispenser. [169]

2.2.5 Rails shall have a minimum width of 8.5 mm measured from the edge of the rail to the first prostration on each face, ref 183. [169]

2.2.6 Rails should have a surface roughness less than $1.6 \mu m$. [169]

2.2.7 The edges of the rails should be rounded to a radius of at least 1 mm. [169]

2.2.8 The ends of the rails on the +/- Z face shall have a minimum surface area of 6.5 mm x 6.5 mm contact area with neighbouring CubeSat rails.[169]

2.2.9 At least 75% of the rail should be in contact with the dispenser rails and 25% of the rails may be recessed. [169]

2.2.10 The typical maximum mass for a 1U configuration is 2.00 kg. [169]

2.2.11 The CubeSat center of gravity shall fall within the ranges defined in fig 182. Note: The acceptable center of gravity location is measured from the geometric center on each major axis. [169]

3.1.1 Random vibration testing shall be performed to the levels and duration as defined by the launch provider. [169]

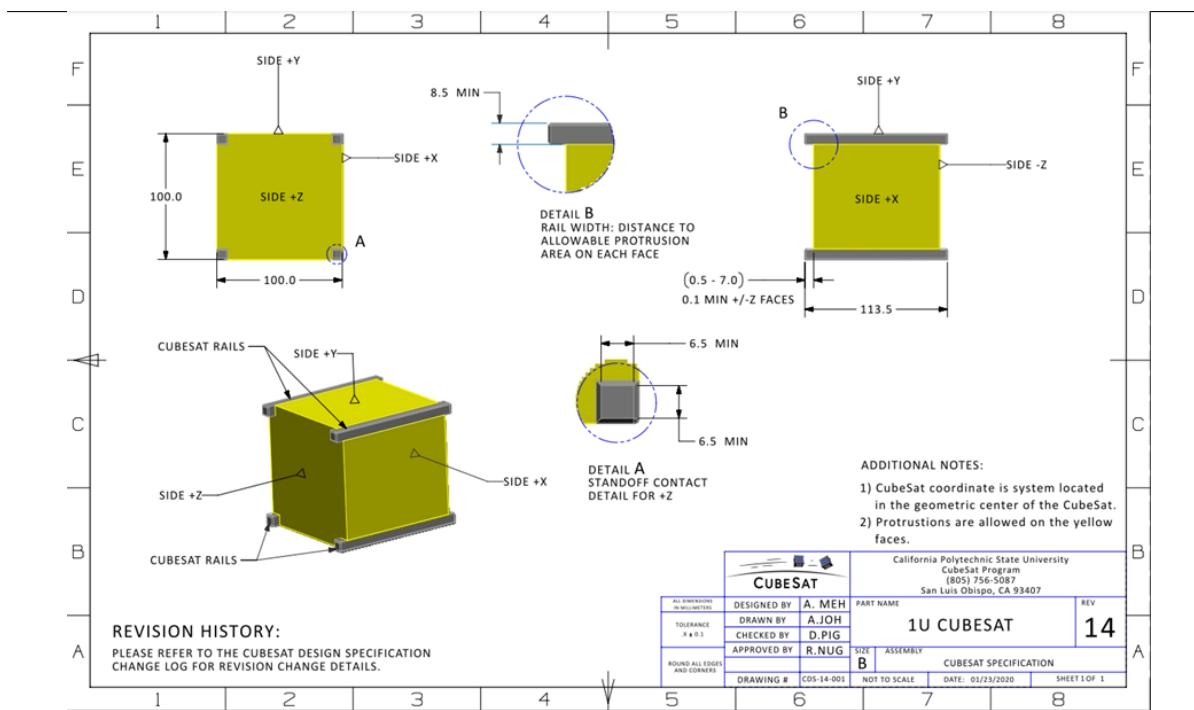


Figure 181: 1U CubeSat Sheet [169]



Figure 182: Acceptable center of gravity [169]

~~SHOW ALL FIGURES~~



Figure 183: Distance - edge to edge [169]

B Tools for GUI

DØ|TL

B.1 Use of Tkinter

When learning Tkinter, a course for creating apps was followed [155]. In the following, you can see some important points extracted from the tutorial.

The "window.mainloop()" function in Tkinter is essential for event handling, registering interactions like button clicks and mouse movements until the application window is closed. Code lines after "mainloop()" do not execute during application runtime but only after closure.

To retrieve data from widgets, Tkinter provides methods such as Tkinter variables or the `get()` function, primarily for Entry widgets. Labels are updated using `"label.config(text="sometext")"` or `'label['text'] = 'some other text'` for consistency.

Inbuilt Tkinter variables facilitate automatic updates between widgets, storing basic data like strings, integers, and Booleans. `StringVar` features functions like `"get()"` and `"set("text")"` for real-time updates.

Tkinter offers three main types of buttons-Button, Checkbutton, and Radiobutton. All require Tkinter variables for proper usage. The instantiation of buttons using `"ttk.Button()"` and `"ttk.Checkbutton()"` is interchangeable.

These concise guidelines provide a foundation for effective GUI development with Tkinter and Ttkbootstrap, emphasizing consistent coding practices and widget management.

Ttkbootstrap is a package that offers a modern alternative to the traditional Tkinter package. It offers an expanded selection of widgets and layout [156].

B.1.1 Ttkbootstrap error

DØ|TL

When trying to compile a code with ttkbootstrap package included an error occurred [157]:

```
File "C:\Users\anonym\AppData\Local\Programs\
Python\Python312\
Lib\site-packages\ttkbootstrap\dialogs\dialogs.py", line 566, in DatepickerDialog
locale.setlocale(locale.LC_ALL, locale.setlocale(locale.LC_TIME, ""))
```

Why this error occurred is explained by **Lore-Gaviano** "When working on a PC with Norwegian locale, DatepickerDialog throws a "local.Error", failing the import of ttkbootstrap. Norwegian locale contains non-ASCII characters "å", which is not accepted in setting

the locale in line 566 of \ttkbootstrap\dialogs\dialogs.py".

A fix to the error was found and explained by **Neoakay005**:

1. Just open up the dialogs.py in folder: "C:\Users"user"\AppData\Local\Programs\Python\Python312\Lib\site-packages\ttkbootstrap\dialogs\dialogs.py", line 566
 2. Overwrite this line `locale.setlocale(locale.LC_ALL, locale.setlocale(locale.LC_TIME, "")`)
- with "tr_TR.utf8" (but you can overwrite as default with "en_EN.utf8") `locale.setlocale(locale.LC_ALL, locale.setlocale(locale.LC_TIME, "tr_TR.utf8"))`

and everything runs without any problem.

B.2 PEP 8 – Style Guide for Python Code

DØ|TL

It is a programming style that is used by most Python developers, to have good readability, maintenance, and good documented code. In the rest of this subsection can you see some important points that are taken from the '**PEP 8 – Style Guide for Python Code**' document [29]:

A Foolish Consistency is the Hobgoblin of Little Minds

A style guide is about consistency. Consistency with this style guide is important. Consistency within a project is more important. Consistency within one module or function is the most important.

However, know when to be inconsistent – sometimes style guide recommendations just are not applicable. When in doubt, use your best judgment. Look at other examples and decide what looks best. And don't hesitate to ask!

In particular: do not break backwards compatibility just to comply with this PEP!

Some other good reasons to ignore a particular guideline:

1. When applying the guideline would make the code less readable, even for someone who is used to reading code that follows this PEP.
2. To be consistent with surrounding code that also breaks it (maybe for historical reasons) – although this is also an opportunity to clean up someone else's mess (in true XP style).
3. Because the code in question predates the introduction of the guideline there is no other reason to be modifying that code.
4. When the code needs to remain compatible with older versions of Python that don't support the feature recommended by the style guide.

Use of mathematical Operators:

```
1 # Correct:  
2 # easy to match operators with operands  
3 income = (gross_wages  
4         + taxable_interest  
5         + (dividends - qualified_dividends)  
6         - ira_deduction  
7         - student_loan_interest)
```

UTF-8 Standard and non-ASCII table (as Norwegian ‘ÆØÅ’): All identifiers in the Python standard library MUST use ASCII-only identifiers and SHOULD use English words wherever feasible (in many cases, abbreviations and technical terms are used which

are not English). **Note:** Thats why error occurred using ttkbootstrap B.1.1

Group imports: Imports should be grouped in the following order:

5. Standard library imports.
6. Related third-party imports.
7. Local application/library-specific imports.

You should put a blank line between each group of imports.

Comments

Comments that contradict the code are worse than no comments. Always make a priority of keeping the comments up-to-date when the code changes!

Comments should be complete sentences. The first word should be capitalized unless it is an identifier that begins with a lowercase letter (never alter the case of identifiers!).

Ensure that your comments are clear and easily understandable to other speakers of the language you are writing in.

Python coders from non-English speaking countries: please write your comments in English, unless you are 120

Inline Comments

```
1 Inline comments are unnecessary and in fact distracting
2 if they state the obvious. Do not do this:
3 x = x + 1           # Increment x
4
5 But sometimes, this is useful:
6 x = x + 1           # Compensate for border
7
8 Do not compare boolean values to True or False using ==:
9 # Correct:
10 if greeting:
11
12 # Wrong:
13 if greeting == True:
```

B.3 JetBrains AI Service

DØ|TL

AI Assistant will be used by some team-members for writing code, documentation and for designing our code (This will only be for GUI programming)[158]. What it does, is that it learns your IDE (which package it has installed etc..) and uses the LLMs (Large Language models) to retrieve data [159]. After reading **JetBrains AI Terms of Service** Some important points were highlighted to keep in mind when using this AI Assistant tool [160]:

b) You own Your Inputs and Your Data

As between You and Us and to the extent permitted by applicable law, You own the Inputs and Your Data shared with JetBrains AI and keep all proprietary rights, including intellectual property rights to them.

Every time You submit an Input to JetBrains AI, You confirm that You have the right to submit the Input together with Your Data and understand that You are doing so at Your own risk and that You are solely responsible for this step and all consequences of its use in JetBrains AI. You also indemnify Us from any liability relating to this Input and Your Data (see the ‘Indemnification’ Section).

c) Your rights to use Outputs and Suggestions

Subject to Your compliance with these Terms, JetBrains hereby assigns to You all its right to, title to, and interest in JetBrains AI’s Outputs and Suggestions. You understand that the Outputs and Suggestions are sometimes a result of third-party large language models and as such can be subject to third-party rights, including open-source licenses. You also acknowledge that Outputs and Suggestions are generated as a non-exclusive response to Your Inputs or certain context of Your work with JetBrains Products, so the same or similar Outputs or Suggestions can also be generated to other JetBrains AI customers, based on their inputs or context of their use.

B.3.1 Map Plot with coordinates

Here is a procedure to get a Plot of coordinate points on a map by mainly using AI Assistant (NOTE: it is not been taken into consideration of using the PEP 8 - Style):

Question: make a code that plots map positions with the tkinter and ttkbootstrap package: Received data is latitude and longitude

Answer: Here’s an example code snippet that plots positions on a map using Python’s tkinter and ttkbootstrap packages. It should be noted that tkinter does not inherently support map plotting. If you want to plot actual geographical data, it would be more suitable to use dedicated libraries such as Matplotlib’s Basemap toolkit or Folium. However, if you only need to plot generic points on a canvas (which could represent a map

for some purpose), here's how you can use Tkinter. Because of the restrictions, I use custom map data points and custom map layouts for the demonstration. Please replace the dummy_data placeholder with your actual data.

Done: Installed Matplotlib package

No code was used after the last input/question, but found out that the Matplotlib package was needed. The question was formulated to get a better answer that fit for our purpose:

Question: make a code that plot map positions in a GUI interface with the tkinter, ttkbootstrap, and matplotlib package: Received data is latitude and longitude

Answer: Sure, assuming you want to embed a Matplotlib figure that plots map positions inside a Tkinter window, you could use the matplotlib's FigureCanvasTkAgg class to place the plot on a tkinter widget. Here is a basic example:

```
import tkinter as tk
from tkinter import ttk
from ttkbootstrap import Style

import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from mpl_toolkits.basemap import Basemap

def draw_map_on_canvas(canvas, latitude_list, longitude_list):
    # Set your plot inside a figure
    figure = plt.Figure(figsize=(5, 4), dpi=100)

    m = Basemap(projection='cyl', llcrnrlat=-90, urcrnrlat=90, llcrnrlon=-180,
                urcrnrlon=180, resolution='c', ax=figure.add_subplot(1, 1, 1))

    m.drawmapboundary(fill_color="#A6CAE0", linewidth=0)
    m.fillcontinents(color='green', alpha=0.3)
    m.drawcoastlines(linewidth=0.1, color="white")

    # plot points on the map
    for lon, lat in zip(longitude_list, latitude_list):
        x, y = m(lon, lat)
        m.plot(x, y, 'ro', markersize=5)

    map_plot = FigureCanvasTkAgg(figure, master=canvas)
```

```
map_plot.draw()
map_plot.get_tk_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=True)

def main():
    app = tk.Tk()
    style = Style('lumen') # using the 'lumen' theme
    style.configure(app)
    canvas = ttk.Frame(app, width=500, height=500)
    canvas.pack()

    longitude_list = [77.209021, 72.877655, 88.363895, 91.736237]
    latitude_list = [28.613939, 19.075984, 22.572646, 25.578773]

    draw_map_on_canvas(canvas, latitude_list, longitude_list)

    app.mainloop()

if __name__ == "__main__":
    main()
```

This assumes that you can install and use the Basemap package on your system. Note that Basemap does require you to have a way to compile Fortran code to be installed from the source, which might be cumbersome. Furthermore, the Basemap package is being deprecated and functionality is being moved to Cartopy. So for modern and more robust geographical plotting, consider using Cartopy in the future. Either way, this code creates a new Tkinter window, plots your map on a Matplotlib canvas using Basemap, then embeds this Matplotlib canvas in the Tkinter window using FigureCanvasTkAgg. Please replace the longitude_list and latitude_list with your actual data.

Done: Installed basemap and ttkbootstrap package

After small adjustments as replacing one of the longitude and latitude coordinates with Andøya's coordinates, we get this output (shown in 184) when running the code:

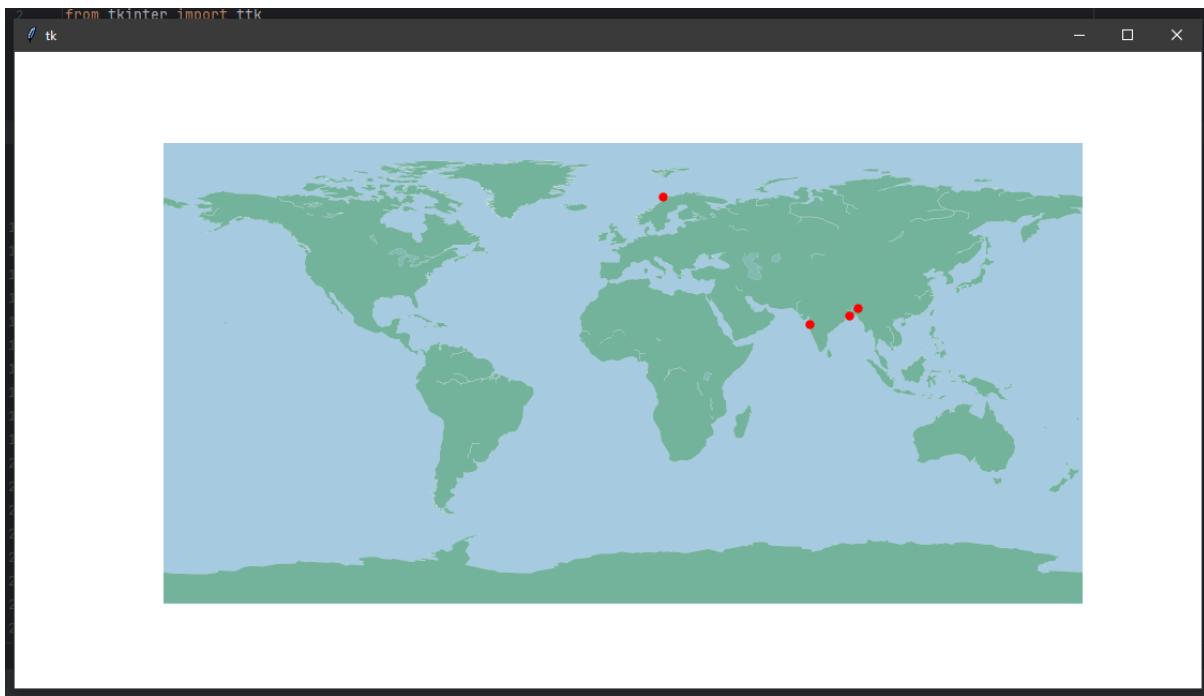


Figure 184: Map plot with coordinate points

B.3.2 RSSI Plot

Graph plot is something that will be used to display sensor data. This shows the procedure of using AI Assistant to display RSSI (Radio Signal Strength)(dB). :

Keep in mind that randomly generated values were used to reproduce the values of the RSSI data for testing purposes.

Question: Generate code to show five points at the same time on the graph in real time. Distance in (km) at the x-axis and RSSI (dB) at the y-axis.

Code generated:

```
import tkinter as tk
import matplotlib
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.figure import Figure
import numpy as np
import ttkbootstrap as ttk
import random
import time

matplotlib.use('TkAgg')
#distance = 0.0

class RealtimePlot(tk.Tk):
```

```

def __init__(self, *args, **kwargs):
    self.distance = 0.0
    # self.distances = [] # List to store distances
    tk.Tk.__init__(self, *args, **kwargs)
    tk.Tk.wm_title(self, 'Real-time RSSI Plot')
    self.style = ttk.Style()
    self.style.theme_use('cosmo')

    self.fig = Figure(figsize=(5,5), dpi=100)
    self.plot = self.fig.add_subplot(111)

    self.canvas = FigureCanvasTkAgg(self.fig, master=self)
    self.canvas.draw()
    self.canvas.get_tk_widget().pack()

    self.update_plot()

def update_plot(self):
    rssi = random.uniform(-100, -30) # generate random rssi
    # distance = random.uniform(0, 100) # generate random distance
    self.distance = self.distance + 0.5

    self.plot.cla()
    self.plot.plot(self.distance, rssi, marker='o')
    self.plot.set_xlabel('Distance (Km)')
    self.plot.set_ylabel('RSSI (dB)')

    self.canvas.draw()

    # update every second
    self.after(1000, self.update_plot)

if __name__ == '__main__':
    root = RealtimePlot()
    root.mainloop()

```

But realize it displayed 5 points over eachother as shown in 185:

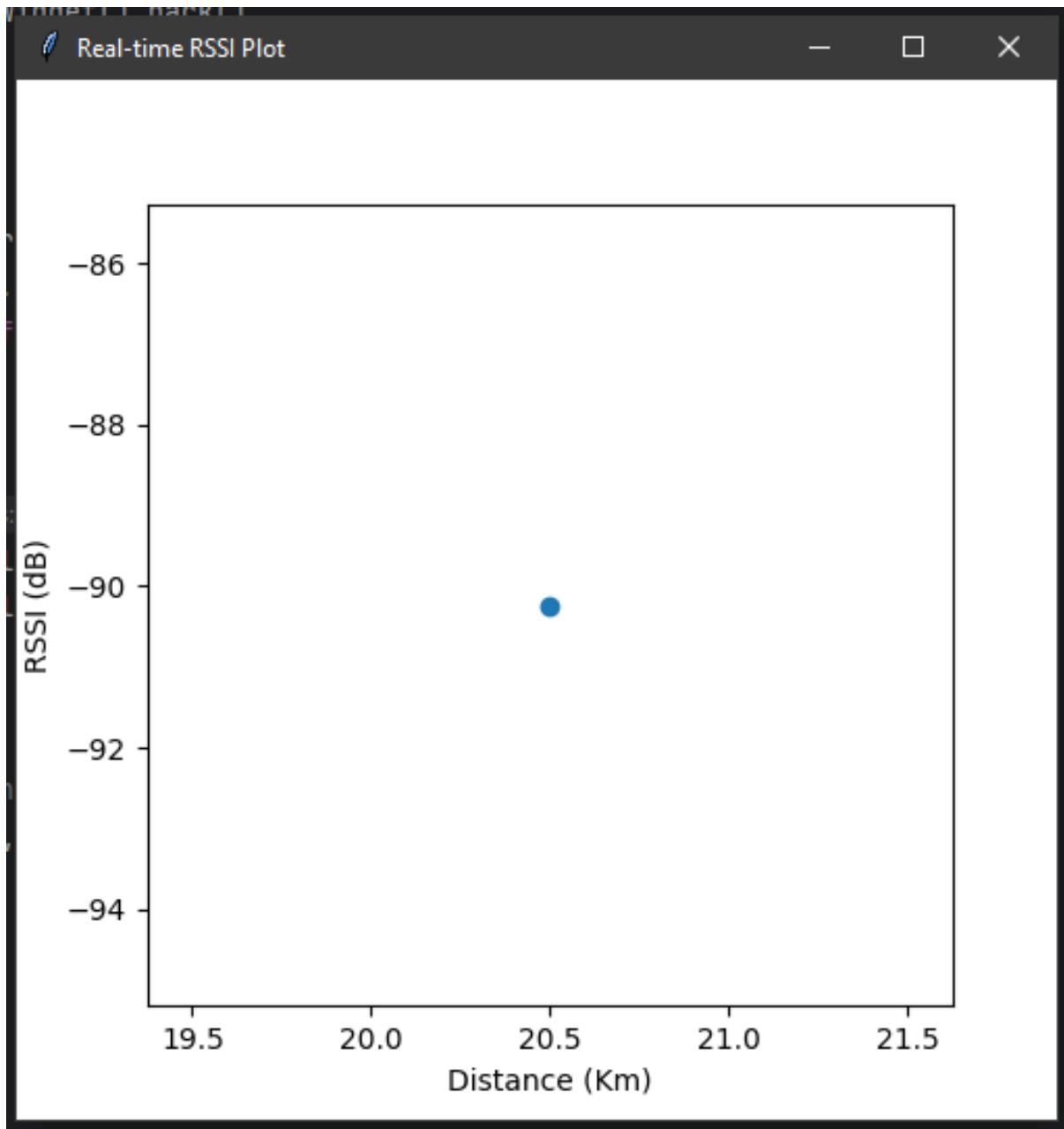


Figure 185: RSSI plot with no lines between points

By rephrasing the question, a much better code was suggested:

Question: Generate code to show five points at the same time on the graph in real time. There should be a line between each point. Distance (km) at the x-axis and RSSI (dB) at the y-axis.

Generated code:

```
import tkinter as tk
import matplotlib
```

```
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.figure import Figure
import numpy as np
import ttkbootstrap as ttk
import random
import time
from ttkbootstrap import Style

matplotlib.use('TkAgg')

class RealtimePlot(tk.Tk):
    def __init__(self, *args, **kwargs):
        self.distances = [] # To store the 'distance'-'rssи' pairs
        self.dist_val = 0.0 # Dette ble lagt til
        tk.Tk.__init__(self, *args, **kwargs)
        tk.Tk.wm_title(self, 'Real-time RSSI Plot')
        self.style = ttk.Style()
        self.style.theme_use('darkly')

        self.fig = Figure(figsize=(5,5), dpi=100)
        self.plot = self.fig.add_subplot(111)

        self.canvas = FigureCanvasTkAgg(self.fig, master=self)
        self.canvas.draw()
        self.canvas.get_tk_widget().pack()

        self.update_plot()

    def update_plot(self):
        rssи = random.uniform(-100, -30) # generate random rssи
        # generate dist incrementally
        self.dist_val = self.dist_val + 0.5

        self.distances.append((self.dist_val, rssи)) # Save the 'dist'-'rssи' pair

        if len(self.distances) > 5:
            distances, rssis = zip(*self.distances[-5:]) # Get last 5 pairs only
        else:
            distances, rssis = zip(*self.distances) # Get all pairs

        self.plot.cla()
        self.plot.plot(distances, rssis, marker='o') # Plot
        self.plot.set_xlabel('Distance (Km)')
        self.plot.set_ylabel('RSSI (dB)')
```

```

        self.canvas.draw()

    self.after(1000, self.update_plot) # Update every second

if __name__ == '__main__':
    root = RealtimePlot()
    root.mainloop()

```

By running this code you will see that 5 points is displayed and a new point comes after every second. Note that these RSSI values is not realistic where the data have random values generated by the code 186:

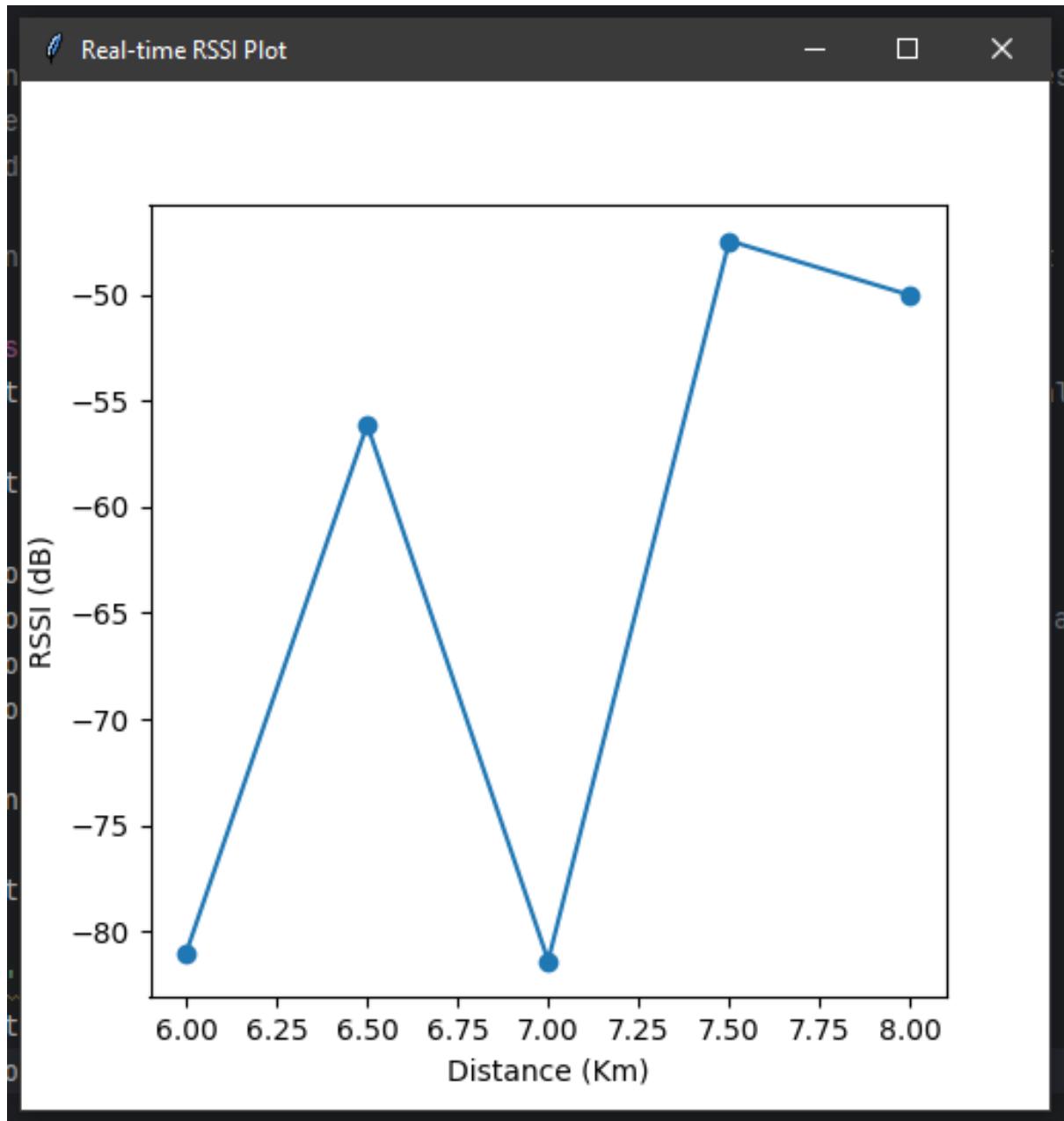


Figure 186: RSSI plot with lines between points

C Making the plot interface

DØ|TL

Achieving a user-friendly interface for our students is a goal in our project that meets the **Stakeholder Requirement** "3. Easy to assemble and disassemble: (d) 3.1 The software for the C.R.A.F.T-Sat must be easy to read and use for the end users." 3.1. The aim is to make a modular and flexible design of our interface that underscores the commitment to adaptability and future integration of new modules. And our GUI to be usable for several years in the future, adhering to a rule of thumb that ensures its longevity that meets the **Stakeholder Requirement** "6. The entire system should be made with longevity in mind. This applies to but is not limited to: (c) Software implementations" 3.1. Below, we detail the progress (in iterations) made thus far in developing the plot interface.

C.1 Real-time RSSI Plotting

Our objective was to develop a real-time plot of Received Signal Strength Indication (RSSI) values using Tkinter and Matplotlib. The primary aim was to visualize the variation in RSSI (dB) concerning distance (Km) in a graphical representation. To be able to see real-time data of RSSI plotted on a graph.

We utilized the following packages:

Tkinter: for creating the GUI window [109].

Matplotlib: for generating the plot within the Tkinter window [27].

RealtimePlot Class: To achieve real-time plotting functionality, we designed a RealtimePlot class. This class incorporated the following key components:

We implemented a method named `update_plot()` within the RealtimePlot class. This method was configured to update the plot at regular intervals, precisely every 1000 milliseconds (1 sec). The update was achieved using the `self.after(1000, self.update_plot)` function call, ensuring the continuous refreshment of the plot with new data points.

The plot consisted of 10 data points. The x-axis represented the distance (Km) while the y-axis represented the RSSI (dB) values. Each point was connected by lines to visualize the trend over time.

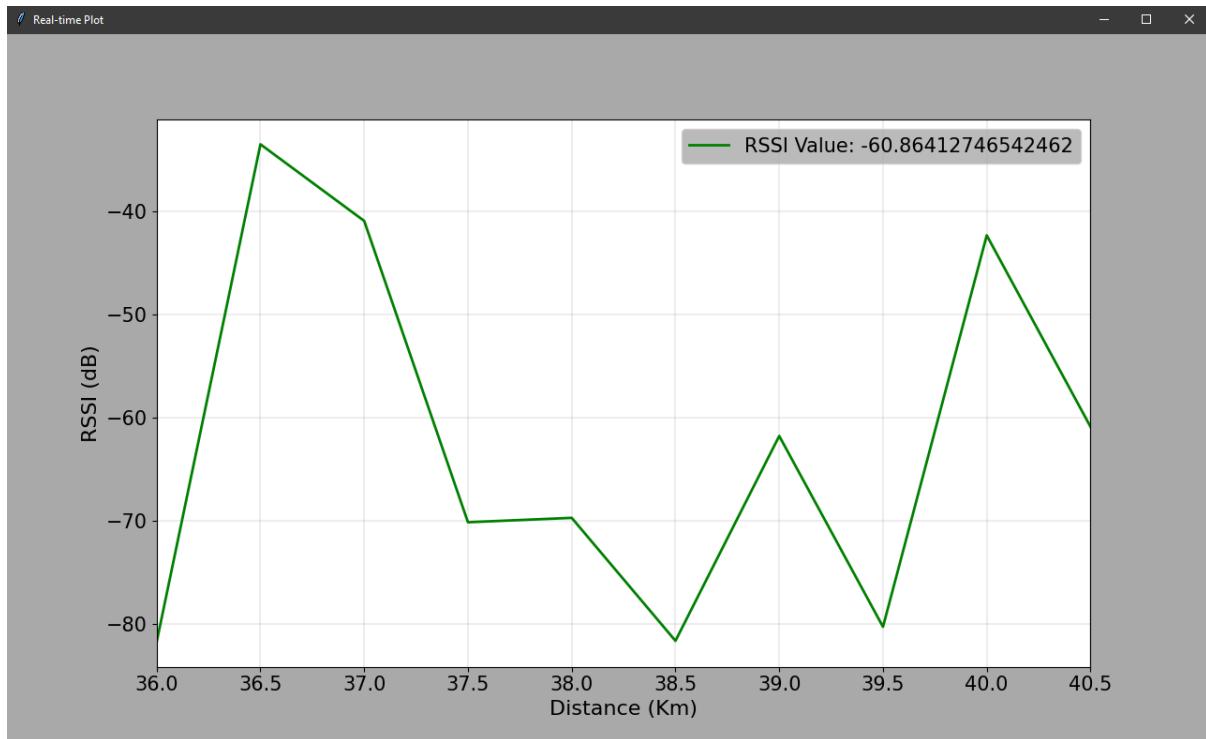


Figure 187: RSSI Plot using tkinter and matplotlib

Through the integration of Tkinter and Matplotlib, we successfully developed a real-time plot of RSSI values against distance. The implementation of the RealtimePlot class facilitated the automatic updating of the plot at one-second intervals, ensuring the dynamic visualization of RSSI trends. This project demonstrates an effective method for monitoring and analyzing radio signal strength variations in real-time scenarios.

C.2 Plot Pop-Up Buttons

The objective was to develop and implement a user interface featuring plot-pop-up buttons for data visualization. The aim was to create a user-friendly interface with four distinct buttons, each corresponding to a specific plot type: RSSI Plot, Acceleration Plot, Magnetometer Plot, and Custom Plot. To be able to choose what plot to display.

Core functionality was encapsulated within the RealtimePlot class, located in the file plotmodule.py. This class was designed to facilitate real-time plotting with customizable parameters.

```
# code: plotmodule.py
import tkinter as tk

class RealtimePlot(tk.Tk):
    def __init__(self, data_label, x_label, y_label,
                 y_lower, y_higher, x_update_val, *args, **kwargs):
        # Initialization code here
```

Buttons for selecting different plot types were implemented in the file `plotmain.py`. Each button was associated with a specific plot type and triggered the display of the corresponding plot when clicked.

```
# code: plotmain.py
from plotmodule import RealtimePlot
import tkinter as tk

# Button configurations
btn1 = tk.Button(root, text="RSSI Plot", command=lambda: display_plot(
    RealtimePlot('RSSI Value', 'Distance (Km)', 'RSSI (dB)', -100, -30, 0.5)))
btn2 = tk.Button(root, text="Acceleration Plot", command=lambda: display_plot(
    RealtimePlot('Acceleration Value', 'Time (sec)', 'Acceleration (m/s^2)', -10, 10, 1)))
btn3 = tk.Button(root, text="Magnetometer Plot", command=lambda: display_plot(
    RealtimePlot('Magnetometer Value (nT)', 'Time (sec)', 'nanoTesla', -1000, 1000, 1)))
btn4 = tk.Button(root, text="Custom Plot", command=lambda: display_plot(
    RealtimePlot('Custom Value', 'Custom X-axis', 'Custom Y-axis', -1000, 1000, 1)))
```

The implementation emphasized flexibility and modularity, allowing for easy extension and customization of plot types in the future. By utilizing the `RealtimePlot` class, additional plot types can be integrated seamlessly into the interface.

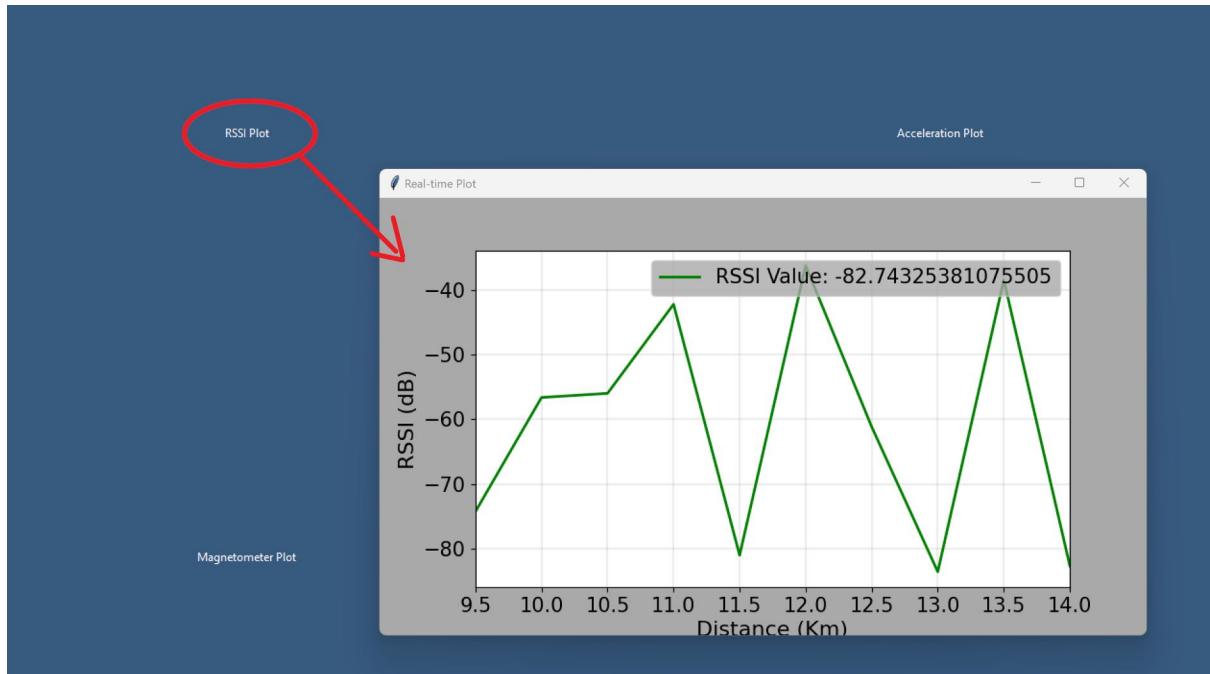


Figure 188: Shows pop-up window of RSSI Plot after clicking 'RSSI Plot' button

It achieved the desired functionality, where clicking on each button resulted in the display of the relevant plot in a new window. This intuitive interface enables users to visualize data according to their specific requirements.

C.3 ‘Add Plot’ buttons with ‘plot options’ popup

The goal was to implement four ‘Add Plot’ buttons that, when clicked, display a ‘plot options’ popup. Users can select a plot type from the popup, which replaces the corresponding ‘Add Plot’ button, offering flexibility in plot placement. To be able to choose what to plot what is to be plotted at a chosen location.

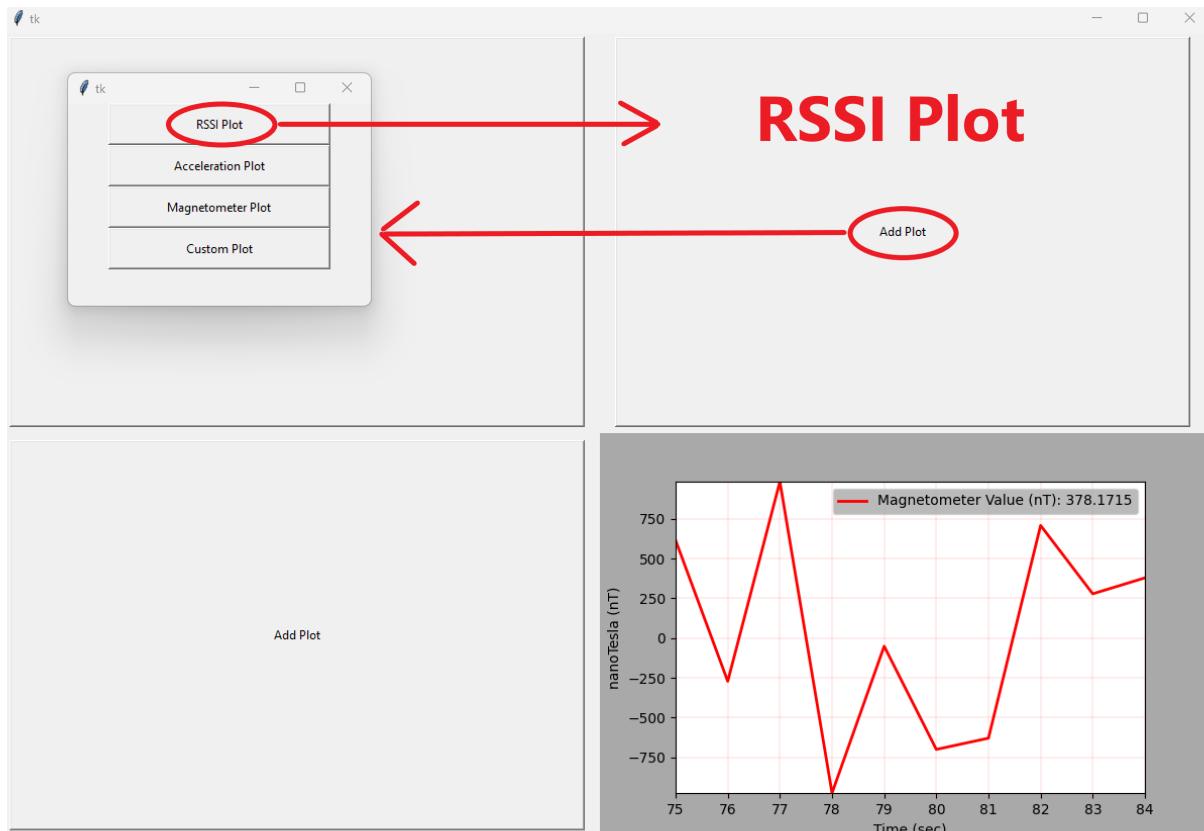


Figure 189: Procedure of clicking ‘Add Plot’ button and choosing RSSI Plot

Chosen plot replaces the clicked ‘Add Plot’ button, and enables users to customize plot placement.

C.4 ‘Plot option’ replaces ‘Add Plot’ button

The aim is to document the implementation of the ‘Plot Option’ feature in the software interface. The objective of this feature is to replace the existing ‘Add Plot’ button with a ‘Plot Option’ frame that allows users to configure plot settings directly. To be able to interact with the ‘plot options’ in a more intuitive way.

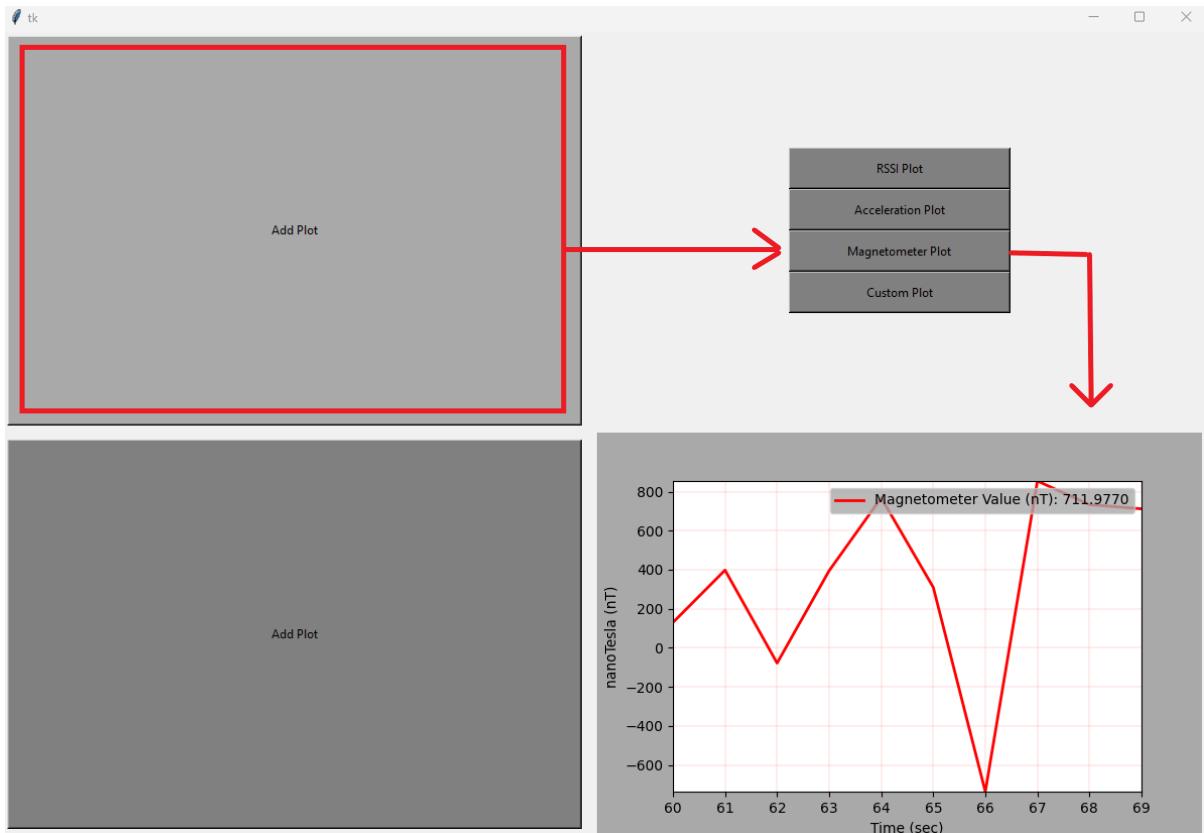


Figure 190: 'Plot Options' and Plot appears inside 'Add Plot' button

A hover effect was incorporated to enhance user feedback. When the mouse pointer hovers over the 'Plot Option' frame buttons, it turns lighter (light grey) to indicate interactivity.

```

def on_enter(e):
    e.widget['background'] = 'darkgrey'

def on_leave(e):
    e.widget['background'] = 'grey'

# Button Creation with Configurations
rss_i_button = tk.Button(frame, text="RSSI Plot", height=2, width=30, bg='grey',
                        command=lambda: display_plot(
                            RealtimePlot(root, 'blue', 'RSSI Value',
                                         'Distance (Km)', 'RSSI (dB)', -100, -30, 0.5),
                            row, col))
rss_i_button.bind("<Enter>", on_enter)
rss_i_button.bind("<Leave>", on_leave)

```

C.5 New ‘Add Plot’ button appears after plot

The objective is to have the same functionality as in **1.4 ‘Plot option’ replaces ‘Add Plot’ button**, but that you start with one ‘Add Plot’ button and a new one appears after a new plot is displayed. To have space for a more modular interface.



Figure 191: New ‘Add Plot’ after a new Plot is displaying

Optimization Case:

Initially, the interface allowed for real-time plotting with an update rate set at 1000ms (1 plot per second). However, when the update rate was increased to 100ms (10 plots per second), and attempts were made to display more than two plots simultaneously, the program crashed.

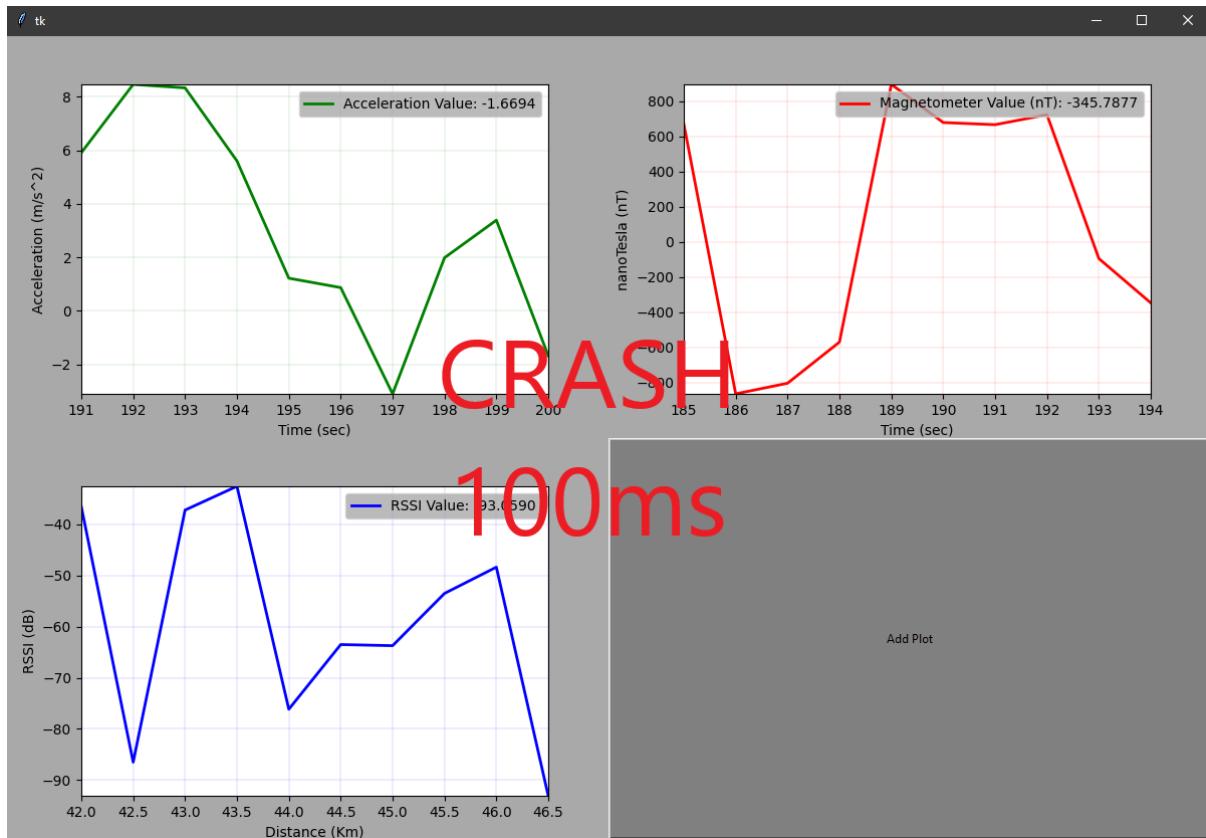


Figure 192: Program crashes when displaying 2 or more plots at the same time

Upon examining the `RealtimePlot` class in `plotmodule.py`, it was discovered that the entire graph was redrawn with each update, leading to performance degradation and eventual crashes.

```
class RealtimePlot:
    self.after(100, self.update_plot)

def update_plot:
    # Draw graph and update line
```

Solution:

To address the performance issue, the approach was modified to draw the graph only once upon instantiation of a new `RealtimePlot` object, updating only the line at each iteration.

```
class RealtimePlot:
    # Draw graph
    self.after(100, self.update_plot)

def update_plot:
    # Update line
```

New Objective The primary objective evolved to enhance modularity and flexibility within the interface, allowing users to customize plot layouts, resize plots, and enable auto-scaling.

Problem Encountered

Efforts to implement drag and drop functionality within the tkinter library proved challenging, as available options such as PanedWindows and non-frame-based plots did not yield the desired results.

Conclusion and Future Directions

Given the limitations of tkinter, exploration of alternative frameworks became necessary. After evaluating PyQt, a GUI framework built around the C++ QT framework [32], its diverse features and built-in drag and drop capabilities [161] were deemed promising for achieving the desired interface enhancements.

C.6 Solution with PyQt

In our pursuit of enhancing functionality within our data visualization application, we endeavored to explore the PyQt5 framework as an alternative to Tkinter. Our primary goal was to replicate the existing features while integrating the drag-and-drop functionality available in PyQt5 [162].

The objective was to attain the same level of functionality as achieved with Tkinter while leveraging the drag-and-drop feature inherent in PyQt5. To be able to change plot position and size with a drag-and-drop feature.

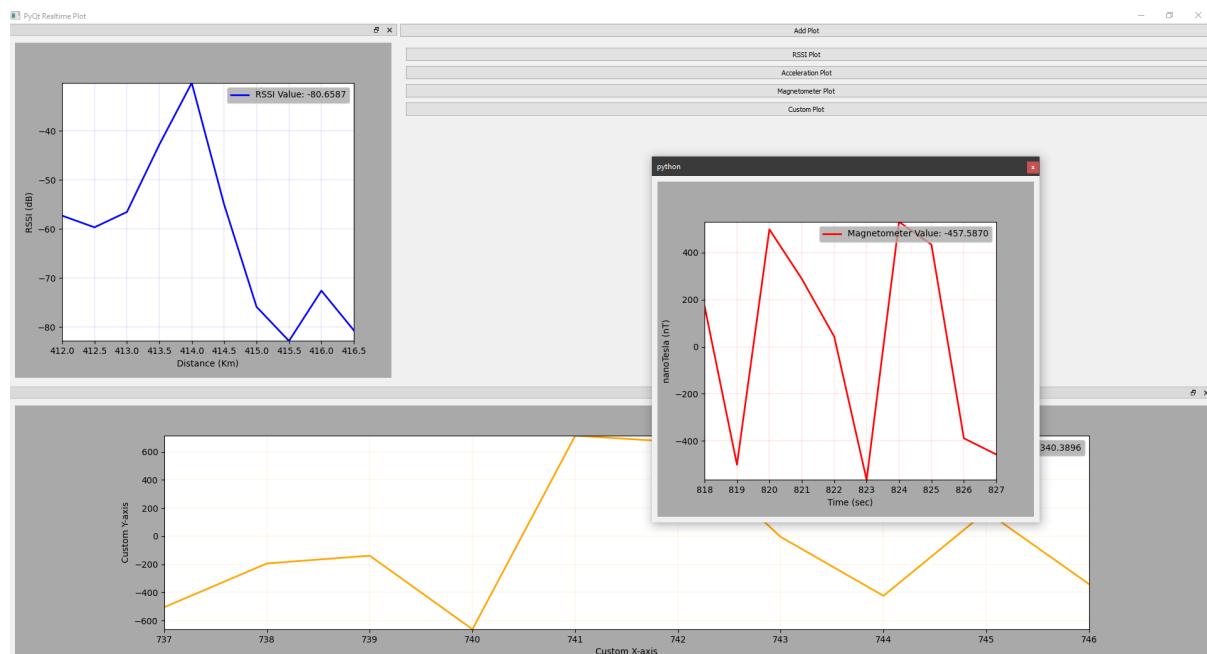


Figure 193: Plot Interface program using PyQt with 'drag and drop' feature

The drag-and-drop feature caused unexpected behavior where the plot/widget would occasionally pop out of the window during manipulation. The application exhibited noticeable lag and buggy behavior when attempting to move the plot around, impacting user experience.

Back to tkinter

We decided to move back to tkinter because:

- First of all. This was stated in section 5.14 "Tkinter is open source and freely available [31], meeting the requirement outlined during discussions with ASE." and "PyQt is available under both GPL and commercial licenses [33], introducing licensing complexities that Tkinter avoids [31]." 5.14.
- The uncertain behavior and complexity of handling certain features within PyQt5.
- Addressing the issues encountered with PyQt5 would require significant effort and expertise.

Despite the absence of the drag-and-drop feature in Tkinter, the team remains committed to finding alternative workarounds to enhance user interaction and experience.

C.7 Tkinter with inspiration from PyQt program

The main goal was to incorporate an 'Add Plot' feature at the top level of the Tkinter window, taking to the functionality observed in the PyQt program. To be able to add a new plot without having a dedicated 'Add Plot' button to each plot placement.

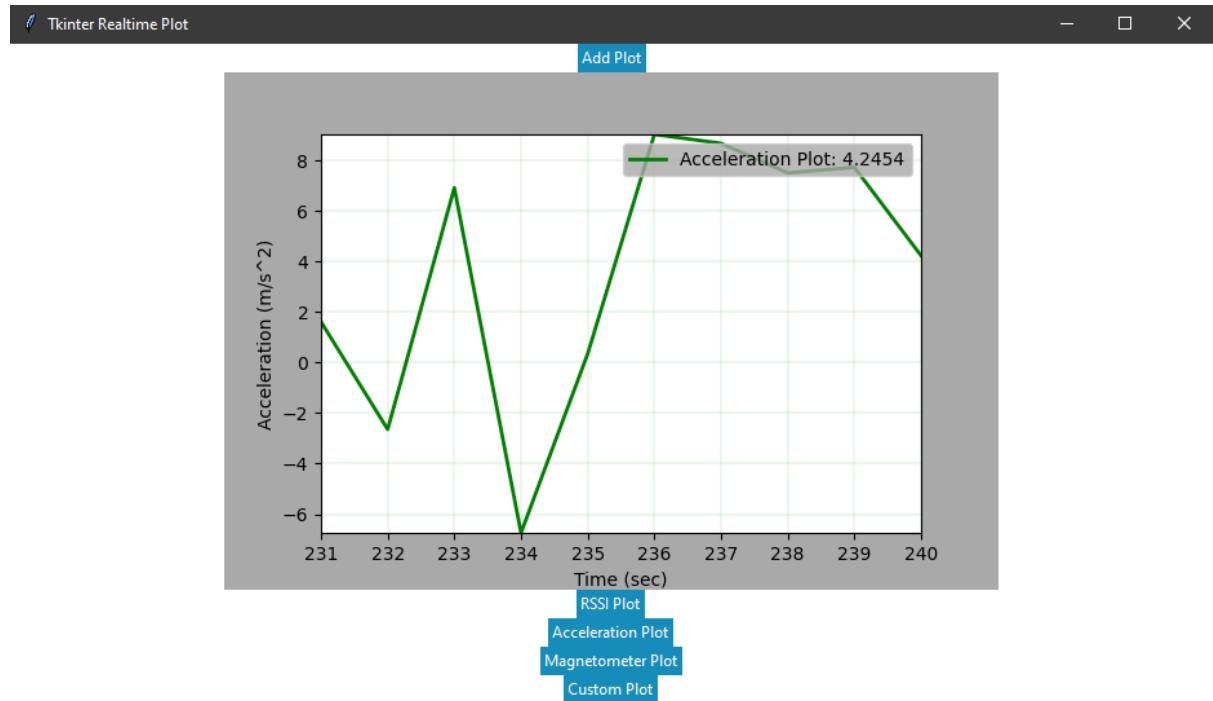


Figure 194: Using tkinter with the same functionality as PyQt (except drag and drop)

Attempts were made to integrate the drag and drop feature on the frame rather than a panedWindow, yet this approach did not yield the desired outcome. After trying several options, it came to a dead end. A conclusion came that we set a stopper to the drag and drop feature, and try to find a workaround that replaces it.

C.8 Plot on a grid

The primary objective of the Grid Plotting Feature was to enable users to organize plots systematically on a grid layout within the software interface. A new mouse interaction mechanism was introduced, allowing users to click, move the mouse pointer, and click again to define the dimensions of a window for displaying a plot. To be able to see and place the plot in a more structured way.

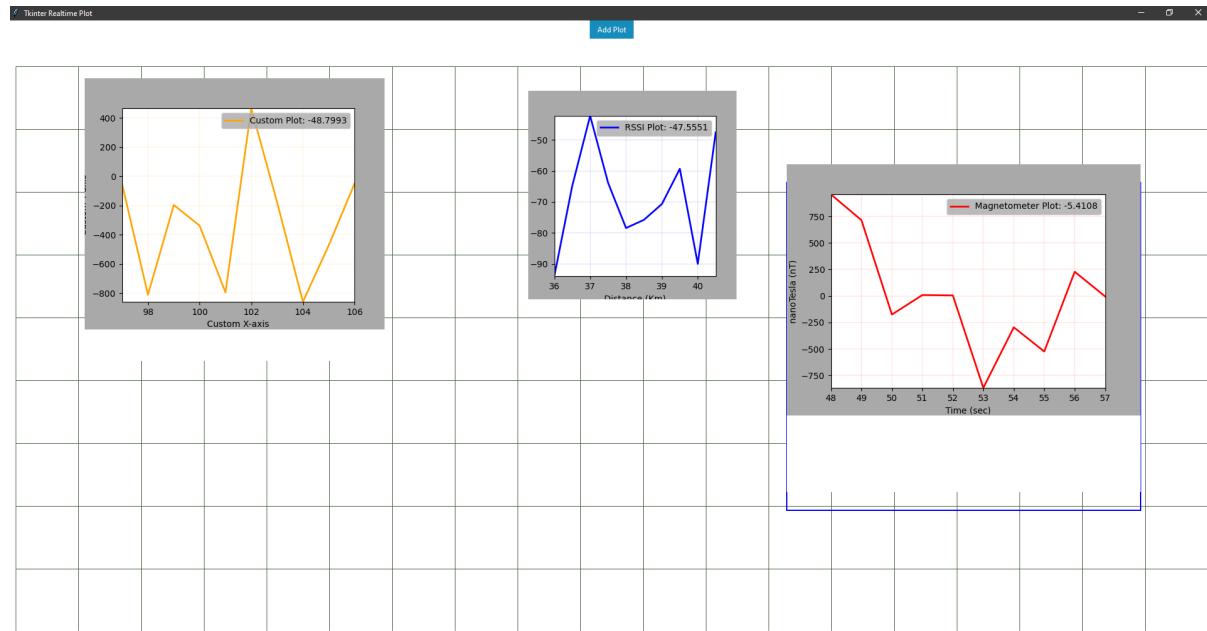


Figure 195: Click move click to display on grid

Despite initial setbacks where the system did not perform as intended, the plot interface exhibited significant bugs, occasionally failing to plot data in the designated locations. However, the implementation of a grid in the background marks the inception of a more structured and organized plot interface. This refinement represents a crucial step forward in rectifying the issues encountered and advancing the functionality of the system.

C.9 Plot on button-grid

The objective of this project was to bring to life an inspiration drawn from the experience of ordering seats online for a movie theatre, taking to the functionality found on platforms such as Nordisk Film Kino [106]. The focus was on implementing a user-friendly interface for selecting and marking areas for plots, to enhance user experience and modularity. To be able to see where the plot will be placed, and to organize the interface in a structured

way.

Implementation involved creating a button-grid interface where users could select and mark areas for plots. Upon selecting a plot, users would click on a button within the grid, and then move the mouse pointer to visualize the area to be marked. A second click would confirm the selection, replacing the marked area with the chosen plot.

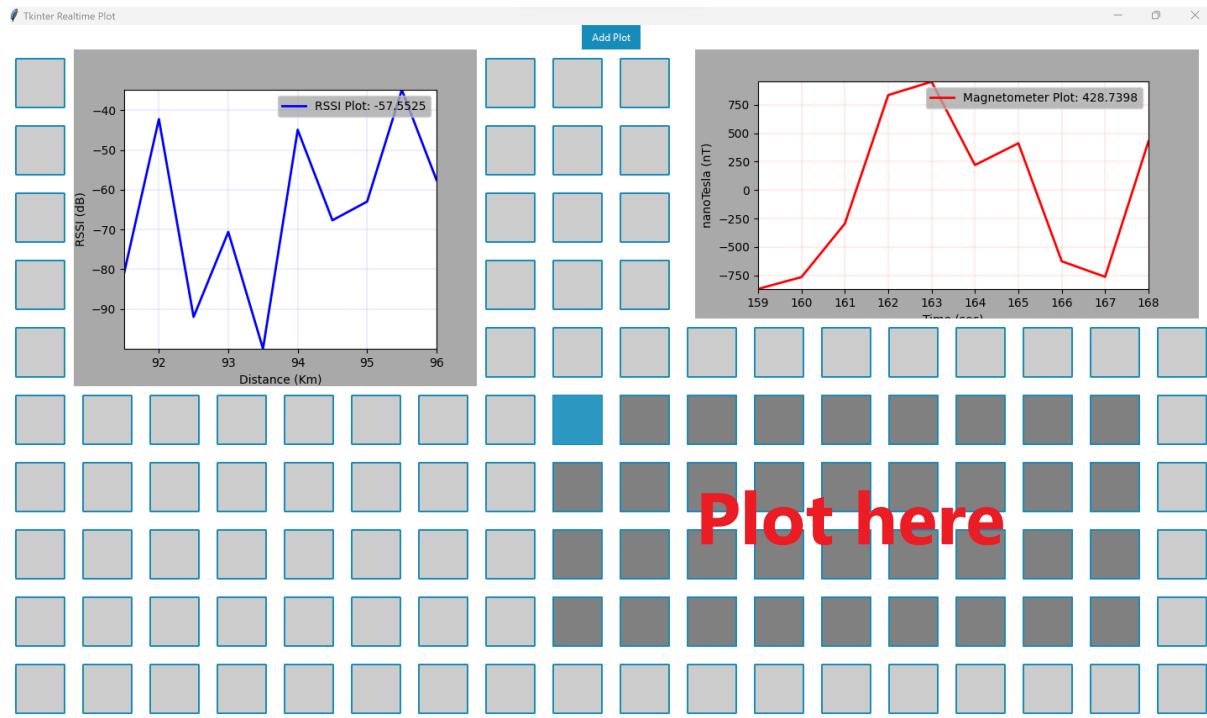


Figure 196: Click move click on button-grid

This solution is a great replacement for drag-and-drop feature and can be seen as having more structure and an easier to it's interface, while it keeps advanced modularity. This solution will most likely be worked on more.

C.10 Save to file

The objective was to enable the capability to save all plot data to a .csv file. This functionality allows users to analyze the data post-runtime or after conducting tests, ensuring data accessibility and usability beyond immediate application use.

The implementation involved the creation of a temporary folder that stores all plot data. This data is updated continuously through the `write_to_csv(self, filename)` function. This function utilizes both the `csv` [101] and `os` [102] libraries to manage file writing and directory operations, respectively. To ensure compatibility with Excel and to prevent the automatic conversion of data into dates, all dots ('.') in numerical values are replaced with commas (','):

```
# plotmodule.py
def write_to_csv(self, filename):
```

```

file_exists = os.path.isfile(filename)
with open(filename, 'a', newline='') as file:
    writer = csv.writer(file, delimiter=';')
    if not file_exists:
        # only write the header if the file did not exist before
        writer.writerow([f'{self.x_label}', f'{self.y_label}', 'Date',
                        'Clock', 'Duration', ])
    # generated the row data and datetime once so they are the same
    row_data = [f'{val:.4f}'.replace('.', ',') for val in self.x_vals[-1]]
    # note: only the last datapoint
    row_data.append('')
    # append the datetime
    row_data.append(datetime.datetime.now().strftime("%H:%M:%S"))
    row_data.append(str(self.elapsed_time))
    writer.writerow(row_data) # write the csv row with the datetime

```

To provide robust tracking and context for the plot data, the `datetime` [103] library is employed. This allows the application to append relevant timestamps such as dates, clock time, and elapsed time to each entry in the .csv file, enhancing the data's informational value.

A user-friendly interface feature was added in the form of a 'Choose Save Location' button. Clicking this button triggers a popup window where users can specify both the filename and the desired save location on their local machine. If a user opts to save the file, the data stored in the temporary folder is merged into a single file using `pandas` library [104]. The temporary folder is then deleted using `shutil` [163]. This merged file contains all plot data, organized into separate tables to facilitate easier analysis:

```

# plotmain.py

def merge_files(self):
    if not self.output_file:
        print("No output file selected.")
        return

    # create a temporary folder to hold single plot files
    temp_folder = "temporary_plot_files"
    os.makedirs(temp_folder, exist_ok=True) # ensure the folder exists

    all_dfs = []
    for plot in self.plots:
        # For each file, read the dataframe and make a copy
        file_path = os.path.join(
            temp_folder,
            f"{'plot['plot'].title_nr}.csv"
        ) # temporary folder added to file path
        df = pd.read_csv(file_path, delimiter=';',

```

```

names=[f'{plot["plot"].title}_{plot["plot"].plot_id}',

        '',
        plot["plot"].start_time.strftime('%Y.%m.%d'),
        plot["plot"].start_time.strftime('%H:%M:%S'),
        f'{plot["plot"].elapsed_time}', ''])

all_dfs.append(df.copy())

# combine all the plot .csv files to one merged file
combined_df = pd.concat(all_dfs, axis=1)
combined_df.to_csv(self.output_file, index=False, sep=';')

# delete all files in the temporary folder
shutil.rmtree(temp_folder)

# recreate the temporary folder
os.makedirs(temp_folder, exist_ok=True)

```

	A	B	C	D	E	F	G	H	I	J	K
1	RSSI Plot_0		14.03.2024	15:34:58	00:00:27		Magnetometer Plot_1		14.03.2024	15:35:02	00:00:22
2	Distance (Kn	RSSI (dB)	Date	Clock	Duration		Time (sec)	nanoTesla (r	Date	Clock	Duration
3	0,5	-53,9228		15:34:58	00:00:00		1	-887,7676		15:35:02	00:00:00
4	1	-56,4002		15:34:59	00:00:01		2	722,4746		15:35:03	00:00:01
5	1,5	-93,6759		15:35:00	00:00:02		3	-51,4741		15:35:04	00:00:02
6	2	-70,5926		15:35:01	00:00:03		4	287,2648		15:35:05	00:00:03
7	2,5	-89,4779		15:35:02	00:00:04		5	359,3645		15:35:06	00:00:04
8	3	-92,9064		15:35:03	00:00:05		6	-17,6406		15:35:07	00:00:05
9	3,5	-57,0285		15:35:04	00:00:06		7	54,1377		15:35:08	00:00:06
10	4	-81,4554		15:35:05	00:00:07		8	-917,7513		15:35:09	00:00:07
11	4,5	-50,4716		15:35:06	00:00:08		9	-738,3345		15:35:10	00:00:08
12	5	-56,8539		15:35:07	00:00:10		10	779,6451		15:35:11	00:00:10
13	5,5	-82,3959		15:35:08	00:00:11		11	192,499		15:35:12	00:00:11
14	6	-82,49		15:35:09	00:00:12		12	-825,3214		15:35:13	00:00:12
15	6,5	-95,3621		15:35:10	00:00:13		13	450,5089		15:35:14	00:00:13
16	7	-70,5474		15:35:11	00:00:14		14	-827,4132		15:35:15	00:00:14
17	7,5	-48,7909		15:35:12	00:00:15		15	113,1108		15:35:16	00:00:15

Figure 197: The content of a merged/saved file

C.11 Serial port

The objective was to effectively plot data received via the serial port from a ground station. The focus was on handling data transmitted over radio and received by the ground station hardware, specifically a Teensy microcontroller.

The implementation leveraged the Python serial [105] library to establish a connection with the ground station via a designated COM-Port. The data received through this connection was directly utilized within the generate_new_data(self) function:

```

# plotmodule.py

# outside RealtimePlot class
import serial

```

```

ser = serial.Serial('COM3', 9600) # this is an example, use your port and baud rate

# ...
def generate_new_data(self):
    while ser.inWaiting():
        try:
            val = ser.readline().decode().strip() # assuming values are sent line by line
            self.x_val += self.x_update_val
            y_val = float(val)
        except ValueError:
            pass
# ...

```

The data of interest during this iteration was the RSSI (Received Signal Strength Indicator) measured in dBm, which reflects the signal strength received by the antenna at the ground station. The tests were successful, confirming that the system could reliably receive and visually represent signal strength data without delays, thereby achieving the set objectives for this development phase.

C.12 Thread v1

The objective was to enhance the GUI's capability to display multiple plots simultaneously. This feature allows for the real-time visualization of either different datasets or the same dataset across several plots, providing a comprehensive view of data dynamics.

The implementation was primarily integrated into the main window code of the GUI. A function, open_serial_port(self), was developed to handle the initiation of data communication through the serial port. Data points read from the serial port are then distributed as inputs to various plot objects within the application. This process is managed by the plotmodule.py within the realtime class:

```

# plotmain.py

# ...
def open_serial_port(self):
    try:
        # Which COM-Port and baudrate to use
        self.ser = serial.Serial('COM3', 9600)
    except serial.SerialException as e:
        print("Could not open serial port: {}".format(e))

def read_from_port(self):
    """Read from serial port and enqueue the messages"""

    if self.ser is None:
        print("No open serial port.")
        return

    while self.running: # Only runs while self.running is True
        val = self.ser.readline().decode().strip()

```

```

try:
    y_val = float(val)
except ValueError:
    y_val = 0
for plot_info in self.plots: # For each plot
    # Set the data point to the plots
    plot_info['plot'].add_datapoint(y_val)

self.ser.close()
# ...

# plotmodule.py

# ...
def add_datapoint(self, val):
    self.queue.put(val) # Add the datapoint to the plot's queue
# ...

```

Given the limitation that serial data can be read in only one place at a time, the implementation required the use of multi-threading to manage simultaneous data streams effectively. This was accomplished using the threading [111] library and the concurrent.futures [164] library. Each plot object is instantiated as a separate thread in the generate_plot_from_cells(self) function. These threads receive data from the main thread, which targets the window.read_from_port function in the `__main__` section of the code:

```

# plotmain.py
# ...
def generate_plot_from_cells(self):
    # ...
    if not self.is_overlapping(min_i, max_i, min_j, max_j):
        plot_queue = queue.Queue() # Create a separate queue for each plot
        new_plot = RealtimePlot(self.grid_frame, self.plot_count,
                               title, color, label)
        # ...
        future = self.executor.submit(self.update_plot_queue, plot_queue)
        plot_thread = threading.Thread(target=new_plot.update_plot)
        plot_thread.start()
        plot_info = {
            'plot': new_plot,
            'coordinates': (min_i, max_i, min_j, max_j),
            'thread': plot_thread,
        }
        # store the plot along with its coordinates for future reference.
        self.plots.append(plot_info)
        # ...
# ...

```

```

if __name__ == "__main__":
    window = MainWindow()
    window.open_serial_port()
    # Start a new thread for reading from port
    read_thread = threading.Thread(target=window.read_from_port)
    read_thread.start() # Start reading from the port
    window.run() # Start UI

```

C.13 Queue (dead end)

The objective was to remain consistent with the previous version: to enable multiple plots to visualize data simultaneously in real time. The approach explored in this iteration involved using a queue to centralize data storage, allowing all plot objects to fetch data from a single source.

The implementation strategy shifted to utilizing a queue mechanism, aimed at storing all incoming data in one accessible location. In the main window code, data received from the serial port was placed into a queue using the self.serial_data.put(y_val) method provided by the .put() function of the queue [165] library. This method was incorporated within the read_from_port(self) function, ensuring that all incoming data was queued for subsequent use by plot objects:

```

# plotmain.py

class MainWindow(tk.Tk):

    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        # ...
        self.serial_data = queue.Queue() # Create a queue to hold the serial data
        # ...
    # ...

    def read_from_port(self):
        if self.ser is None:
            print("No open serial port.")
            return

        while self.running: # Only runs while self.running is True
            val = self.ser.readline().decode().strip()
            try:
                y_val = float(val)
            except ValueError:
                y_val = 0
            self.serial_data.put(y_val) # Put serial data into shared queue

        self.ser.close()
    # ...

```

In the plotmodule_queue, the functionality was structured such that each plot object retrieves data from the shared queue. This was achieved using the .get() function, which allows plots to access the next available data point. The queue was maintained as an attribute within the constructor of the plotting module. During each update cycle, data was fetched from the queue and used as the value for the y-axis on each plot:

```
# plotmodule.py

class RealtimePlot(tk.Frame):

    def __init__(self, ..., serial_data, ...):
        # ...
        self.serial_data = serial_data # save it as an queue attribute
        # ...
    def generate_new_data(self):
        while True:
            val = self.serial_data.get() # Fetch data from the shared queue
            # ...
        # ...

# ...
```

C.14 Thread v2

The objective was to refine the data management system by incorporating a method to count and track each data read from the serial port since the start of the program. This feature ensures that every y-value plotted is associated with a corresponding read number, facilitating detailed analysis and review of the stored data.

The add_datapoint function was reintroduced to manage the assignment of data to plots. In this setup, the data_counter for each plot is synchronized with the value of self.read_counter at the exact moment a y-value is received. This ensures that each data point stored in the queue is tagged with its respective read number. Unlike the previous iteration where a shared queue was utilized, this version employs individual queues for each plot:

```
# plotmain.py

    # ...
    def read_from_port(self):
        """Read from serial port and enqueue the messages"""
        # ...
        while self.running: # Only runs while self.running is True
            # ...
            for plot_info in self.plots: # For each plot
                plot_info['plot'].add_datapoint(y_val)
                plot_info['plot'].data_counter = self.ser_read_counter
            # ...
        # ...
```

C.15 Read and store all sensor data

The objective was to ensure that all sensor data read from the serial port is stored, regardless of whether it is used in a plot. This involves capturing and saving the data with appropriate sensor headers in a .csv file, allowing for comprehensive tracking of all sensor information.

Given the nature of the data stream received from the ground, which was transmitted as a continuous string of sensor values, it was necessary to devise a method for separating sensor labels/names from their corresponding values. The adopted approach involved using semicolons (';') to separate different sensor data pairs and commas (',') to distinguish between sensor labels and their values. An example of the received string format would be “;sen1_name, sen1_value; sen2_name, sen2_value; etc..”

To effectively parse this string, the re [166] library was utilized, particularly the re.findall() function. This function helped in isolating characters and numbers to form key-value pairs. These pairs were then stored in a dictionary called data_row, where the sensor names served as keys linked to their respective values.

The data_row dictionary was subsequently converted into a pandas [104] DataFrame. In this DataFrame, the keys became the headers, and their corresponding values populated the rows beneath, ensuring that values were stored in designated columns. At the end of the read_from_port(self) function, this DataFrame was converted into a .csv file. This file, along with others containing plot data, was stored in the same temporary folder, with files named in a sequence that facilitated easy organization and access:

```
# plotmain.py

# ...
def read_from_port(self):
    """Read from serial port and enqueue the messages"""

    if self.ser is None:
        print("No open serial port.")
        return

    temp_file_name = "temp.csv"
    temp_file_path = os.path.join(self.temp_folder, temp_file_name)

    while self.running: # Only runs while self.running is True
        raw = self.ser.readline().decode().strip()

        # Extract variable-value pairs from raw, using regex
        var_val_pairs = re.findall(';([a-zA-Z]+),(-?[0-9]+)', raw)
        # Initialize dictionary with ser_read_counter
        data_row = {'Data_counter': self.ser_read_counter}

        for var, val in var_val_pairs:
```

```
try:
    # Convert the value to float and store it in the dictionary
    data_row[var] = float(val)
except ValueError:
    print(f"Could not convert value: {val} to float.")

print(data_row)

# Convert the data_row to a DataFrame
data_row_df = pd.DataFrame(data_row, index=[0])

# Only concatenate if data_row_df is not empty
if not data_row_df.empty:
    # Check if DataFrame is None (i.e., this is the first row of data)
    if self.data is None:
        # If so, initialize DataFrame with variable names from data_row
        self.data = data_row_df
    else:
        # Concatenating the new DataFrame with the existing DataFrame
        self.data = pd.concat([self.data, data_row_df],
                             ignore_index=True)

# Save data to temporary CSV file with separator as ;
temp_data = self.data.copy()
#temp_data.to_csv(temp_file_path, index=False, sep=";")
temp_data = temp_data.apply(lambda x: x.map(lambda v:
str(v).replace('.', ',')) if pd.notnull(v) else ''))
temp_data.to_csv(temp_file_path, index=False, sep=";")

# Don't forget to increment your counter
self.ser_read_counter += 1

self.ser.close()
# ...
```

F	G	H	I	J	K	L	M	N	O	P
1_Magnetom	Data_count	altitude	temp		Date	Data_count	Data_time	rssi	altitude	temp
2024,04,05	45.0	0	14		2024:04:05	0	12:04:52	-86	21	10
	46.0	1	10			1	12:04:53	-70	17	8
	47.0	5	8			2	12:04:54	-64	15	4
	48.0	7	4			3	12:04:55	-77	25	15
	49.0	17	15			4	12:04:56	-60	23	17
	50.0	19	17			5	12:04:57	-72	28	19
	51.0	24	19			6	12:04:58	-80	26	24
	52.0	26	24			7	12:04:59	-86	28	23
	53.0	28	23			8	12:05:00	-87	29	24
	54.0	29	24			9	12:05:01	-90	26	27
	55.0	32	27			10	12:05:02	-74	29	30
	56.0	35	30			11	12:05:03	-96	27	34
	57.0	37	34			12	12:05:04	-97	35	35
	58.0	45	35			13	12:05:05	-87	30	36
	59.0	50	36			14	12:05:06	-78	30	27
	60.0	50	27			15	12:05:07	-83	32	34

Figure 198: The content of a merged/save file with all the sensor data in the right end

A notable feature implemented was the `notify_if_temp_files_exist(self)` function. This function triggers a popup notification when the program runs, offering the user the option to either save or delete data stored in the temporary folder. This feature is particularly important for scenarios where the program might crash or shut down unexpectedly, allowing users to recover data that was not properly saved before the disruption:

```
class MainWindow(tk.Tk):

    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        # ...
        self.temp_folder = "temp_files"
        if not os.path.exists(self.temp_folder):
            # create a temporary folder if it doesn't exist
            os.makedirs(self.temp_folder, exist_ok=True)
        else:
            # Notify user if there are existing files in temporary directory
            self.notify_if_temp_files_exist()
        # ...
    def notify_if_temp_files_exist(self):
        if os.listdir(self.temp_folder):
            messagebox.showwarning(
                "Warning",
                "There are files in the temporary directory. Please ensure "
                "they are removed or backed up as required before proceeding.")
        self.save_file_dialog()
    # ...
```

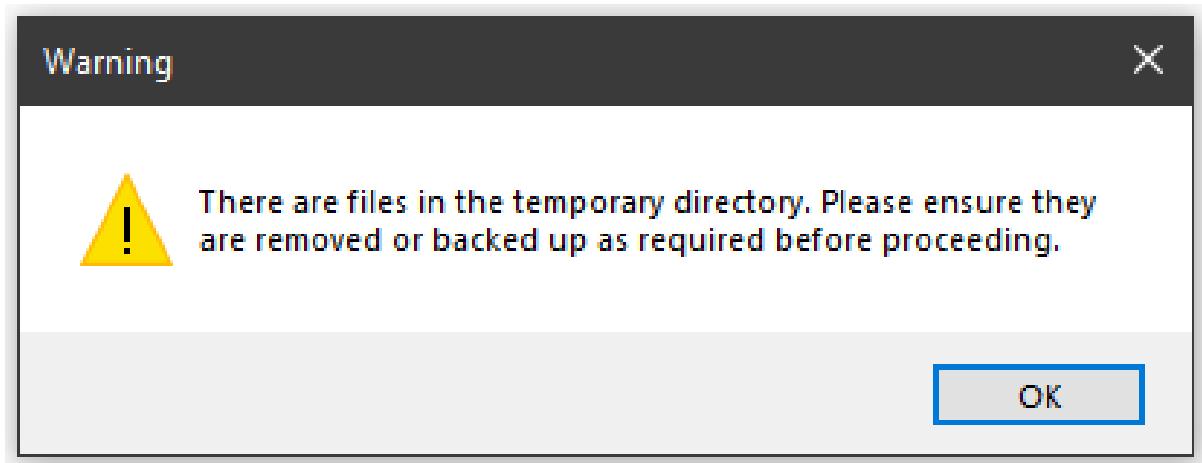


Figure 199: Notify that appears if temporary files exist. If you click ok, then you have the option to save the backup data to file

C.16 Dropdown, datapoint and merged file format

The objective was to provide users with the flexibility to customize the x and y axes of their plots. This feature allows users to select specific data points to compare and visualize against each other, enhancing the analytical capabilities of the GUI.

The functionality to select data for the x and y axes was implemented through drop-down menus. These menus appear when a user clicks on one of the plot-options buttons. The dropdowns are dynamically updated based on the content available in the data DataFrame, which is constructed from the data_row dictionary. Once the axes are selected, users can confirm their choices and proceed to place their plot on the GUI:

```
# plotmain.py
#
def accept_plot_creation(self, title, color, label, x_label, y_label,
    min_val, max_val, interval):
    for button in self.plot_option_buttons: # hide plot options
        button.pack_forget()

    if self.data is not None:
        # calling setup_axis_dropdown() after a plot option is clicked
        self.setup_axis_dropdown()
        columns = list(self.data.columns)
        print(columns)
        self.x_axis_var.set(columns[0])
        self.y_axis_var.set(columns[0])
        self.x_axis_dropdown['values'] = columns
        self.y_axis_dropdown['values'] = columns
        self.x_axis_dropdown.config(state='readonly')
        self.y_axis_dropdown.config(state='readonly')
```

```

self.confirm_button =
ttk.Button(self.plot_option_frame,
text="Confirm Plot", command=self.confirm_plot)

# initially hide the confirm button
self.confirm_button.pack_forget()

# when a plot option is selected, show the labels,
# dropdown menus and confirm button
self.x_label_dropdown.pack()
self.x_axis_dropdown.pack()
self.y_label_dropdown.pack()
self.y_axis_dropdown.pack()
self.confirm_button.pack()

self.selected_plot = (
title, color, label, x_label, y_label, min_val, max_val, interval)

```

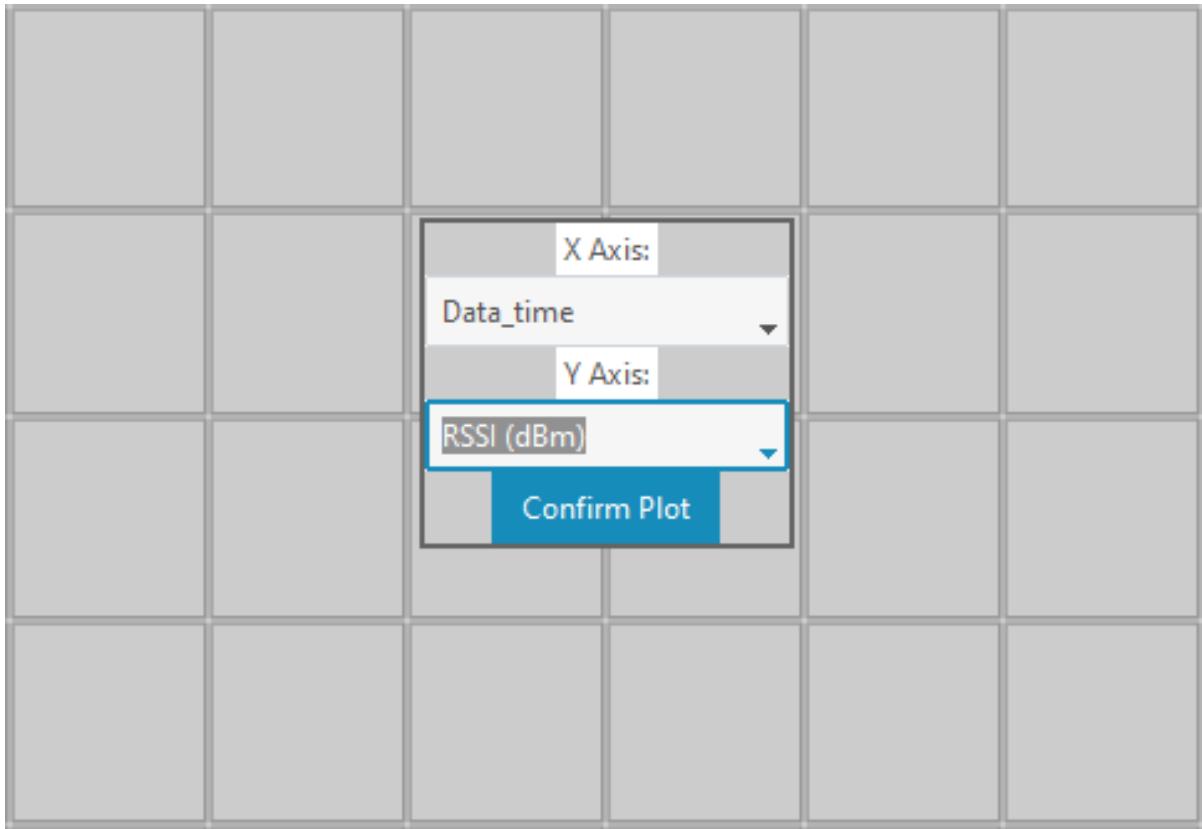


Figure 200: Customize the plot by choosing what data to show on the x and y-axis. Here on the y-axis, the sensor with the id 1 is shown

Within the `read_from_port(self)` function, the `data_row` dictionary is passed to each plot object. Each plot object then checks this dictionary against its specified x and y labels to fetch the appropriate data for visualization:

```
# plotmain.py
# ...
def read_from_port(self):
    """Read from serial port and enqueue the messages"""
    # ...
    while self.running: # Only runs while self.running is True
        # ...
        # Add data_row to each plot's queue
        for plot_info in self.plots:
            plot = plot_info['plot']
            plot.add_datapoint(data_row)
        # ...
    # ...
```

In the RealtimePlot class, the add_datapoint(self, data_row) function includes a check to retrieve the intended values for the x and y axes. The packet_counter variable is incorporated to facilitate data tracking across both the plots and the comprehensive data table. Due to uncertainties in the sequential values that might be input for the x-axis, a cumulative approach is used to manage the x-values, ensuring smooth visual progression without overlaps or misalignments: self.cumulative_x += abs(current_x - self.last_x):

```
# plotmodule.py
# ...
def add_datapoint(self, data_row):
    if self.y_label in data_row and self.x_label in data_row:
        y_val = data_row[self.y_label]
        # This line sets 'packet_counter' value
        packet_counter = data_row.get('packet_counter', None)
        if self.x_label in data_row:
            current_x = data_row[self.x_label]

            # If this is the first datapoint, add it directly.
            if self.last_x is None:
                # Adding 'packet_counter' value
                self.queue.put((current_x, y_val, packet_counter))
                self.cumulative_x = current_x
            else:
                # Increment cumulative value
                self.cumulative_x += abs(current_x - self.last_x)

                # Add the cumulative value to the plot's queue
                self.queue.put((self.cumulative_x, y_val, packet_counter))

            # Now, set 'last_x' to current x
            self.last_x = current_x
    # ...
```

Modifications were also made to the merge_files(self) function to address issues with NaN values and data type inconsistencies in the merged DataFrame. Prior iterations faced challenges when aligning tables of differing lengths, which resulted in unintended conversions to float types and pervasive NaN values. To mitigate this, the DataFrame is now pre-constructed based on the maximum number of rows and total columns needed, with placeholders initialized accordingly. Ensuring that all column headers are unique across data and plot data sets is crucial for this configuration to function correctly:

```
# plotmain.py
# ...
def merge_files(self):
    # ...
    all_dfs = []
    column_headers = [] # List to hold all column headers

    for filename in os.listdir(self.temp_folder):
        if filename.endswith('.csv'):
            file_path = os.path.join(self.temp_folder, filename)
            df = pd.read_csv(file_path, delimiter=';')
            all_dfs.append(df)
            # Prefix columns with filename to ensure uniqueness
            prefixed_columns = [col for col in df.columns]
            column_headers.extend(prefixed_columns)

    if not all_dfs:
        return

    # Find the maximum number of rows and total columns needed
    max_rows = max(df.shape[0] for df in all_dfs)
    total_columns = sum(df.shape[1] for df in all_dfs)

    # Initialize a DataFrame with placeholders
    combined_df = pd.DataFrame(index=range(max_rows),
                                columns=range(total_columns), dtype=str).fillna('')

    # Populate the combined DataFrame
    current_col = 0
    for df in all_dfs:
        rows, cols = df.shape
        combined_df.iloc[:rows, current_col:current_col + cols] = df.values
        current_col += cols

    # reset self.data
    self.data = None
    # ...
    # Assign the collected column headers to the combined DataFrame
```

```
combined_df.columns = column_headers
# ...
```

F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1_Accelerati	1_packet_co	1X_Time (sec)	Y_8		Date	Total_Elapse	Data_time	Elapsed_time	Time (sec)	1	2	3	4	
22.04.2024		30 119.0			22.04.2024	00:00:48	20:45:17	00:00:01	1 778.0	898.0	719.0	206.0		
		31 628.0					20:45:18	00:00:02	2 298.0	789.0	604.0	83.0		
		32 267.0					20:45:19	00:00:03	3 851.0	828.0	930.0	299.0		
		33 197.0					20:45:20	00:00:04	4 784.0	249.0	598.0	769.0		
		34 831.0					20:45:21	00:00:05	5 372.0	306.0	287.0	460.0		
		35 820.0					20:45:22	00:00:06	6 7.0	97.0	891.0	79.0		
		36 186.0					20:45:23	00:00:07	7 205.0	584.0	181.0	679.0		
		37 797.0					20:45:24	00:00:08	8 741.0	741.0	277.0	729.0		
		38 733.0					20:45:25	00:00:09	9 255.0	947.0	553.0	834.0		
		39 574.0					20:45:26	00:00:10	10 584.0	225.0	889.0	811.0		
		40 337.0					20:45:27	00:00:11	11 228.0	125.0	245.0	664.0		
		41 237.0					20:45:28	00:00:12	12 972.0	305.0	159.0	267.0		
		42 779.0					20:45:29	00:00:13	13 411.0	840.0	534.0	714.0		
		43 132.0					20:45:30	00:00:14	14 914.0	597.0	451.0	380.0		
		44 437.0					20:45:31	00:00:15	15 92.0	321.0	286.0	651.0		
		45 72.0					20:45:32	00:00:16	16 956.0	439.0	874.0	375.0		
		46 327.0					20:45:33	00:00:17	17 907.0	386.0	654.0	259.0		
		47 648.0					20:45:34	00:00:18	18 91.0	722.0	488.0	22.0		
		48 463.0					20:45:35	00:00:19	19 310.0	968.0	45.0	750.0		

Figure 201: The merged/saved file with unique column headers

C.17 ID check

The objective was to seamlessly integrate the configuration settings from the GUI with the data received at the ground station via the serial port. This setup enables users to selectively plot data from specific sensors and sub-sensors based on their unique identifiers.

Each sub-sensor is stored in a MySQL database with unique identifiers. Users can select desired sensors for plotting, which are then compiled into a list of tuples, each containing elements like ('id', 'name', 'prefix', 'unit'). It's important to note that some sub-sensors might be sent without a 'prefix', requiring special handling in the system.

Data received from the ground station is formatted using the string structure ;id,data;, where traditional name/label identifiers are replaced with their corresponding IDs. This adjustment simplifies the data matching process between the configuration interface and the actual data received.

Upon initialization, the list of tuples is converted into a dictionary called self.sensor_row, where each sensor ID serves as a key. During data handling, precautions are taken to accommodate tuples that may lack a 'prefix' using conditional checks within the set_sensor_dict(self) function. This setup also concatenates various attributes into a single string to serve as a label for data identification:

```
# plotmain.py
#
class MainWindow(tk.Tk):

    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        # ...
```

```

        self.sensor_from_config = [
            ('1', 'RSSI', '(dBm)'),
            ('2', 'Distance', 'k', 'm'),
            ('3', 'Temp', '*C')
        ]
        self.sensor_row = {}

        self.set_sensor_dict()
        # ...
    # ...
def run(self):
    self.mainloop()

def set_sensor_dict(self):
    for sensor in self.sensor_from_config:
        if len(sensor) == 4:
            sensor_id, name, prefix, unit = sensor
            self.sensor_row[sensor_id] = f'{name} {prefix}{unit}'
        elif len(sensor) == 3:
            sensor_id, name, unit = sensor
            self.sensor_row[sensor_id] = f'{name} {unit}'
        else:
            print("Invalid sensor")
    # ...

```

The setup_plot_options(self) function enables dynamic creation of plot buttons based on the self.sensor_row dictionary. All x-labels default to 'Data_time', which represents the actual time data was received. However, users can customize x-axis labels and values in the 'Custom Plot' option:

```

# plotmain.py
# ...
def setup_plot_options(self):

    if self.sensor_row is not None:
        columns = list(self.sensor_row.values())
        for col in columns:
            self.plot_options.update({col: ["blue", 'Data_time', col]})

    self.plot_options.update({"Custom Plot":
        ["orange", 'Custom X-axis', 'Custom Y-axis']})
    # ...
# ...
def accept_plot_creation(self, title, color, x_label, y_label,):
    if title != "Custom Plot":
        self.selected_plot = (
            title, color, x_label, y_label)

```

```
    self.confirm_plot()
    return
# ...
# ...
```

Data matching is executed by identifying common keys between the database configuration IDs and the IDs from the received data. A dictionary, matched_rows, is then formed, incorporating 'Data_time' as the default x-axis and looping through matched keys to append corresponding sensor names/labels and values. This dictionary is then dispatched as datapoints to the plot objects. If 'Data_time' is used as the x-label, the cumulative adjustment typically employed in the add_datapoints function is omitted to maintain accuracy:

```
# plotmain.py
# ...
def read_from_port(self):
    """Read from serial port and enqueue the messages"""
    # ...
    while self.running:  # Only runs while self.running is True
        # ...
        data_row_keys = set(data_row.keys())
        sensor_row_keys = set(self.sensor_row.keys())

        common_keys = data_row_keys & sensor_row_keys

        matched_row = {'Data_time': data_row['Data_time']}

        matched_row.update({self.sensor_row[key]: data_row[key] for
                           key in common_keys})

        # Add data_row to each plot's queue
        for plot_info in self.plots:
            plot = plot_info['plot']
            plot.add_datapoint(matched_row)
        # ...
    # ...

# plotmodule.py
# ...
def add_datapoint(self, data_row):
    if self.y_label in data_row and self.x_label in data_row:
        # ...
        if self.x_label in data_row:
            # ...
        else:
            if self.x_label == 'Data_time':
                # Adding 'packet_counter' value
```

```

        self.queue.put((current_x, y_val, packet_counter))
        return
    # ...
# ...

```

To ensure compatibility and legality in file naming, especially given that sensor names might contain special characters, a sanitize_filename function was introduced. This function replaces any illegal characters with underscores ('_') before saving the data to a temporary plot file:

```

# plotmodule.py
# ...
def sanitize_filename(self, name, replacement='_'):
    """
    Sanitize the filename by replacing all invalid characters.
    :param name: Original filename
    :param replacement: Character to replace invalid filename characters to
    :return: Sanitized filename
    """

    # Define a list of invalid characters.
    # You may need to modify this for OS compatibility
    invalid_chars = r'[\/*?"<>|]'
    return re.sub(invalid_chars, replacement, name)

def update_plot(self):
    while not self.queue.empty():
        # ...
        # check to replace illegal special characters
        safe_title = self.sanitize_filename(self.title_nr)
        self.write_to_csv(os.path.join(self.temp_folder, f'{safe_title}.csv'))
        self.after(100, self.update_plot)
# ...

```

C.18 Plot and button design

The objective was to refine the design of the plot interface, making it intuitive and user-friendly, to ensure that students feel they are engaging with a polished and easy-to-use program.

An initial step involved addressing a longstanding error that, while not previously disruptive to functionality, could lead to error messages that might confuse users. The issue was related to the plotting of clock time, which needed to be formatted using mdates from the matplotlib.dates [167] package. Fixing this error was crucial for maintaining the professional quality and reliability of the application:

```
# plotmodule.py
# ...
```

```

def add_datapoint(self, data_row):
    if self.y_label in data_row and self.x_label in data_row:
        #
        if self.x_label in data_row:
            if self.x_label == 'Data_time':
                self.save_time = data_row[self.x_label]
                current_x = datetime.datetime.strptime(data_row[self.x_label],
                                                       "%H:%M:%S")
            #
        #
    #
# ...

def plot_clear_and_configure(self, x_vals, y_vals):
    if x_vals and y_vals: # check if x_vals and y_vals are not empty
        #
        if self.x_label == 'Data_time':
            # This line is important for dealing with dates on x-axis
            self.plot.xaxis_date()
            # Convert dates into numbers matplotlib can understand
            min_x, max_x = mdates.date2num(min(x_vals)),
                           mdates.date2num(max(x_vals))
            if min_x != max_x:
                #
            # Format x-axis data as time
            self.plot.xaxis.set_major_formatter(
                mdates.DateFormatter('%H:%M:%S'))
            #
        #
    #
# ...

```

To enhance readability and the overall visual appeal of the plots, `self.fig.tight_layout()` was utilized. This method automatically adjusts subplot parameters to give the plot elements sufficient room, preventing the plots from appearing cramped or overlapping. Additionally, switching from the pack to grid layout manager in the GUI design helped anchor plots firmly within their designated spaces, effectively eliminating unwanted white space around the plot canvas:

```

# plotmodule.py
# ...
class RealtimePlot(ttk.Frame):
    def __init__(self, **kwargs):
        ttk.Frame.__init__(self, parent, *args, **kwargs)
        #
        # Use grid instead of pack and make it expand in all directions
        self.canvas.get_tk_widget().grid(row=0, column=0, sticky="nsew")
        #
    #

```

```

def update_line(self, xdata, ydata):
    # ...
    self.fig.tight_layout(pad=0.5)
    # ...
# ...
# ...

# plotmain.py
# ...
class MainWindow(tk.Tk):

    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        # ...
        # Create a style for button hover and apply on ttk Buttons
        self.selected_button_style = Style()
        self.selected_button_style.configure("Selected.TButton",
                                            foreground="white",
                                            background="#FAE5D3",
                                            bordercolor="#F5CBA7",
                                            lightcolor="#F5CBA7",
                                            darkcolor="#F5CBA7")

        self.selected_button_style.map("Selected.TButton",
                                      # color when you hover over the button
                                      background=[('active', '#F0B27A')],
                                      bordercolor=[('active', '#F39C17')],
                                      lightcolor=[('active', '#FCA31A')],
                                      darkcolor=[('active', '#DE8E13')])
        # ...
# ...
# ...

```

The dpi (dots per inc) in the constructor of RealtimePlot was changed from 100 to 50 to get rid of tight_layout error "self.fig = Figure(figsize=(6, 4), dpi=50)". It made the size of the graph smaller/tighter, but the resolution is lowered. All labels and tick sizes had to be changed. You can see the final design of this iteration below:

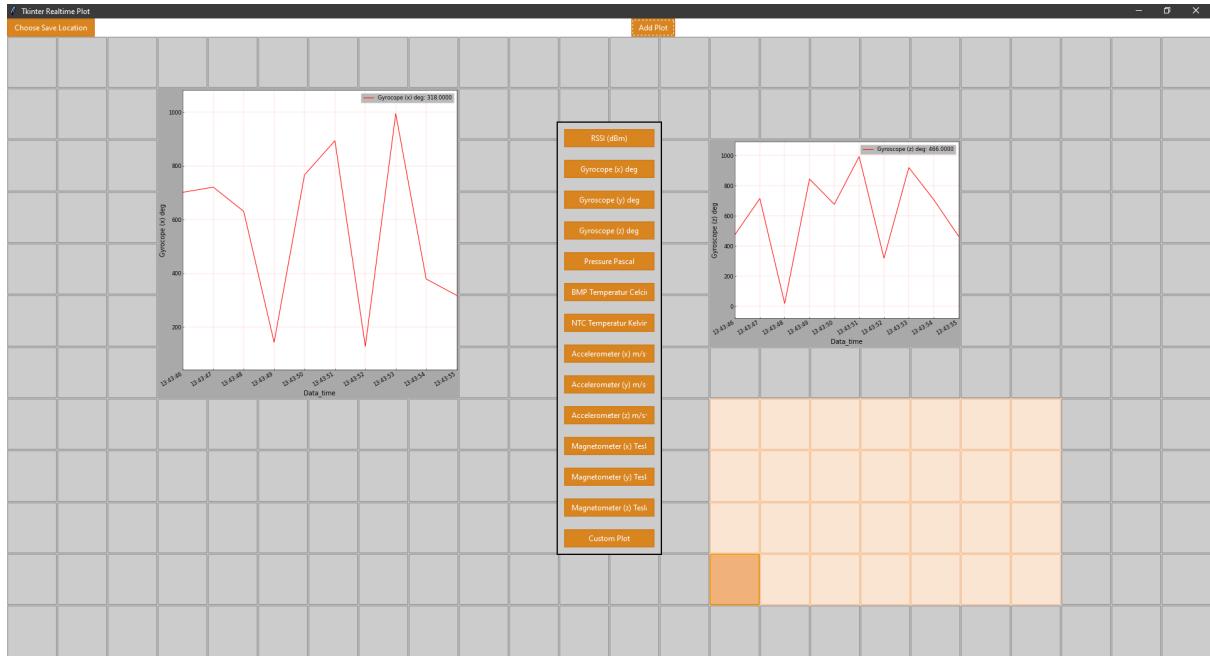


Figure 202: This is the newest design until now. The light orange area is when you choose where to plot

C.19 Converting from ttkbootstrap to CustomTkinter

The objective was to integrate code with another module (`gui_poc`), which initially presented several debugging challenges. It was discovered that `ttkbootstrap` and `CustomTkinter` [99] could not be used simultaneously within the same codebase, leading to the removal of all `ttkbootstrap` components and a switch to `CustomTkinter` or standard `tktinter`. A 'Plot Interface' button was added to the existing module to provide direct access to the new plotting features.

Furthermore, it was necessary to align the code with the main loop of `gui_poc`, for which a specific solution was developed. Upon launching the module, serial reading from the port and data saving began immediately. Additionally, a feature was implemented on the configuration page allowing users to select specific subsensors, generating a list of tuples that are sent to the `PlotInterface` class to create labels and buttons for plotting. This integration streamlined the codebase and improved the system's operational flow.

```
# gui_poc.py
# class ConfigurePlot
# after pressing "Go to plot interface" button
def accept_plot_interface(self):
    self.controller.set_config_sensors(self.sensors_to_plot)
    self.controller.show_frame(PlotInterface)

# class SampleApp
def set_config_sensors(self, config_sensors):
    self.config_sensors = config_sensors
```

```
def show_frame(self, cont): # container
    frame = self.frames[cont]
    frame.tkraise()
    if cont == PlotInterface:
        frame.set_config_sensors(self.config_sensors)
        frame.on_show_frame()
    print(f"{cont} frame raised")
```

C.20 Nested frames for plot buttons

The objective was to set up nested frames stacked vertically, each containing a row of four buttons. These frames and buttons are updated automatically based on the selected number of subsensors or sensors, preventing the list from extending too long vertically and disappearing off the screen. Additionally, a bug with the dropdown menu that interfered with the plot_option frame was resolved by destroying frames when they were no longer needed and recreating them as necessary. This setup ensured a more organized and accessible user interface.

```
# plotmain.py
def setup_graph_options(self):
    ...
    for key, value in self.plot_options.items():
        if key != "Custom Plot" and key != "Clock Time":
            ...
            i += 1

            if i % 4 == 0:
                row_frame.pack(pady=10)
                row_frame = tk.Frame(self.plot_option_frame)
                row_frame.config(background="grey80")
            ...
        ...
```

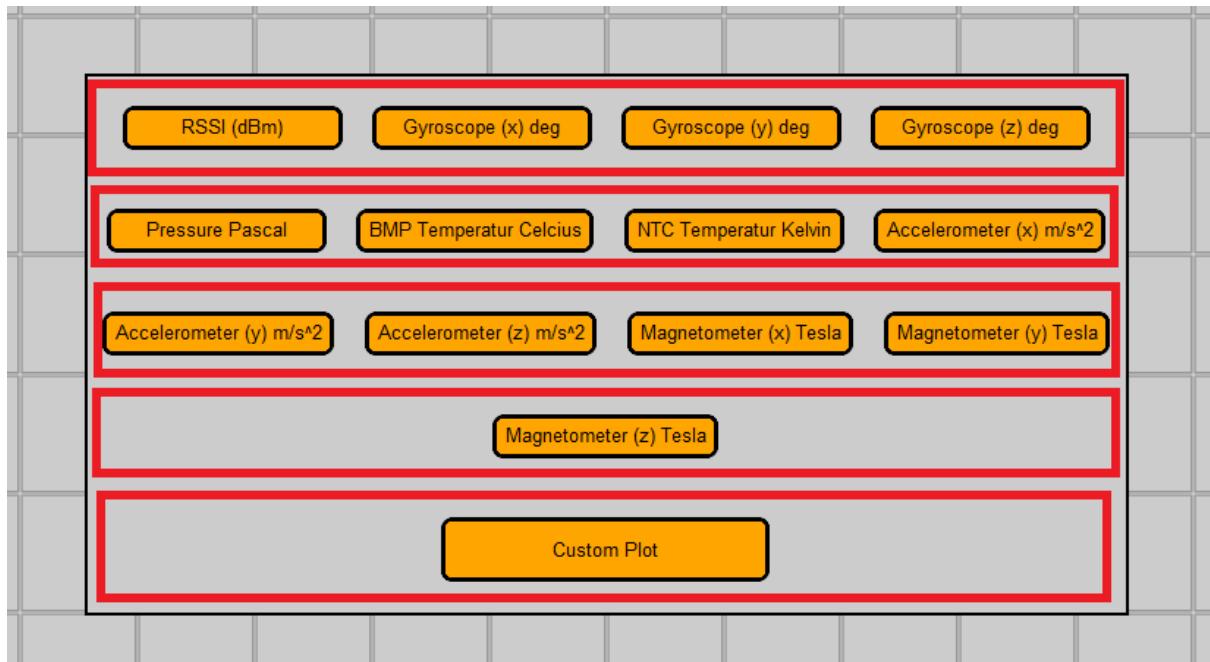


Figure 203: Showing the interface of choosing what graph to plot. The red squares is the nested frames

C.21 Plug and play with auto-detection for serial port

The objective was to implement plug-and-play functionality with auto-detection for the serial port. The program was designed to prevent crashes when a COM port is plugged in or out by handling exceptions. Within these exceptions, function calls were used to iteratively open the COM port using `open_serial_port`. The code was also modified to loop through all connected COM ports until it finds one that is connected and in the correct format, locking onto it until disconnected, then rechecking all connected ports. This enhancement streamlined the GUI for better plug-and-play capability.

```
# serial_save.py
# class SerialPort
    def open_serial_port(self):
        ports = serial.tools.list_ports.comports() # get list of available ports
        for port in ports:
            while True: # loop until we successfully connect or exhaust tries
                try:
                    self.ser = serial.Serial(port.device, 9600, timeout=1)
                    print(f"Testing serial port: {port.device}")
                    raw = self.ser.readline().decode("utf-8", "ignore").strip()

                    match = re.search(r";([^\x2C]+),(-?[0-9]*\.[0-9]+)", raw)
                    if match is not None:
                        print(f"Successfully opened serial port: {port.device}")
                        return
                else:
```

```

        print(f"Data from {port.device} does not match the expected f
        self.ser.close()
        break # wasn't break before, and checked same over and ove

    except (serial.serialutil.SerialException, AttributeError):
        print(f"Could not open port {port.device}. Retrying in 5 seconds.
        time.sleep(1) # wait for 1 second
        continue # retry opening the serial port

    except Exception as e:
        print(f"Error reading from port {port.device}: {str(e)}")
        self.ser.close()
        break # Exit the while loop and try next port

    self.ser = None # Exception("No compatible serial port found.")

```

C.22 Integrated MongoDB and added Logging page

The objective was to integrate MongoDB Atlas [100] for data management and add a logging page. The logging side was implemented and MongoDB database was utilized to successfully store documents and retrieve them as needed. However, difficulties were encountered in retrieving intermediate data, such as sensor data and time. Attempts to fetch an ID from previous storage to then retrieve intermediate data were not entirely successful.

After trying various queries, it was discovered that placing datetime.datetime objects first in the database, then converting them to strings in the correct time format in the DataFrame for storage, was effective. The same approach was applied when exporting data from MongoDB to a string before saving it to a .csv file. Now, all entries can be efficiently retrieved from the database, where the file's destination and name can be specified, enhancing the system's data handling capabilities.

```

# serial_save.py
# class SerialPort
    def read_from_port(self):
        try:
            if connected:
                # data_row_df is a dataframe
                data_dict = data_row_df.to_dict("records")
                # Insert collection
                my_collection.insert_many(data_dict)
        except Exception as e:
            print(f"Error occurred while inserting data into MongoDB: {st

# class FileManagement
    def save_into_mongo_db(self):

```

```

print("Saving data to MongoDB...")
filename = os.path.splitext(os.path.basename(self.main_obj.output_file))[0]
print(filename)
# Create a new document
doc = {
    "Filename": filename,
    "Start Date": self.main_obj.start_time,
    "End Date": datetime.now(),
    "Total Elapse": str(self.main_obj.elapsed_time) # Saving elapsed_time as
}

# Insert the document into the collection
my_collection.insert_one(doc)

def export_from_mongo_db(self, filename):
    # ...
    start_date = name_and_date_doc["Start Date"]
    end_date = name_and_date_doc["End Date"]

    between_data = list(my_collection.find({"Date Time": {"$gte": start_date, "$lt": end_date}}))
    # ...

```

C.23 Table- and GPS-Plot implementation

The objective was to implement Table- and GPS-Plot functionalities in the plotting interface, allowing users to select between graph plot, table plot, or GPS plot. Checkbuttons were added to the interface for users to choose what data to include in table plots. In both the TablePlot and GpsPlot classes, update_plot() and add_datapoint() methods were implemented to ensure these plotting options functioned correctly, enhancing the flexibility and utility of the plot interface. The TkinterMapView [168] library was used to visualize tracking points with coordinates on a map. This development provides users with more tailored plotting capabilities based on their specific data visualization needs.

```

# plotmodule.py
# in class GraphPlot, TablePlot and GpsPlot
def close(self):
    # Find the corresponding plot_info for the current plot
    plot_info_to_remove = next((plot_info for plot_info in self.main_obj.plots if
                                plot_info['type'] == self.type))

    # If the corresponding plot_info is found, remove it
    if plot_info_to_remove is not None:
        self.main_obj.plots.remove(plot_info_to_remove)

    # Retrieve the coordinates of the removed plot
    min_i, max_i, min_j, max_j = plot_info_to_remove['coordinates']

```

```

# Display the corresponding grid buttons in the main class
for i in range(min_i, max_i + 1):
    for j in range(min_j, max_j + 1):
        self.main_obj.cell_buttons[i][j].grid()

self.destroy() # destroys the frame that holds the plot and the exit button

```

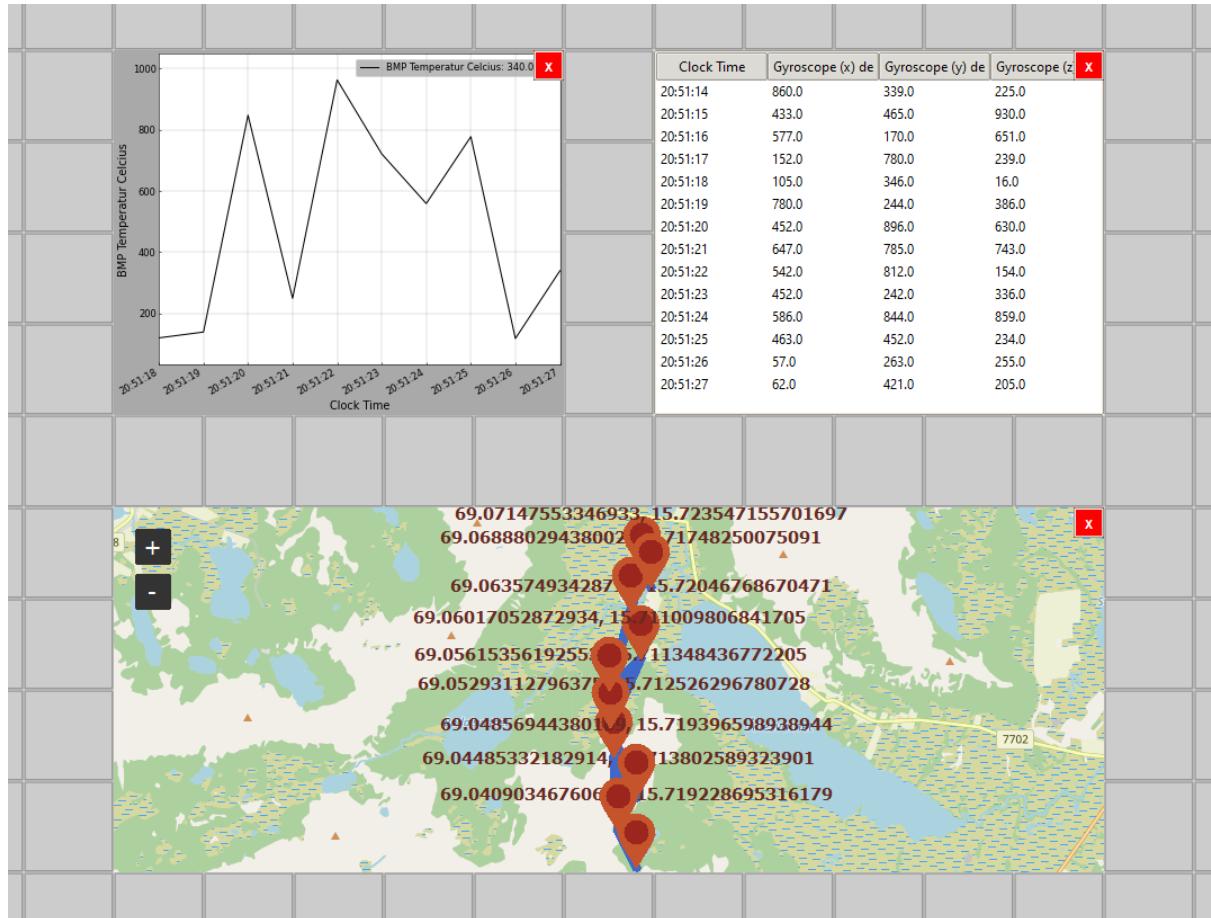


Figure 204: Realtime plot on graph, table and gps at the same time. Values is random generated.

C.24 Exit button for plot frames and Uplink to satellite

The objective was to enhance the plot frames and satellite communication functionalities. An exit button was added to plot frames, necessitating the storage of plotting coordinates before deleting the frame. This facilitated the recreation of buttons in the "button grid" using the stored coordinates. Additionally, functionality was included in the PlotInterface to send uplink commands to the satellite via the serial port through an input field. This integration significantly improved the interface's user interaction and expanded its capabilities to include direct communication with satellites.

```

# serial_save.py
# class SerialPort

```

```

def send_uplink_command(self, command):
    """
    Sends an uplink command to the satellite via the serial port.

    Args:
        command (str): The command to be sent.
    """
    if self.ser is None or not self.ser.isOpen():
        raise Exception('Serial port is not open')
    command_bytes = bytes(command, 'utf-8')
    print("The entered command is: ", command)
    self.ser.write(command_bytes)

```

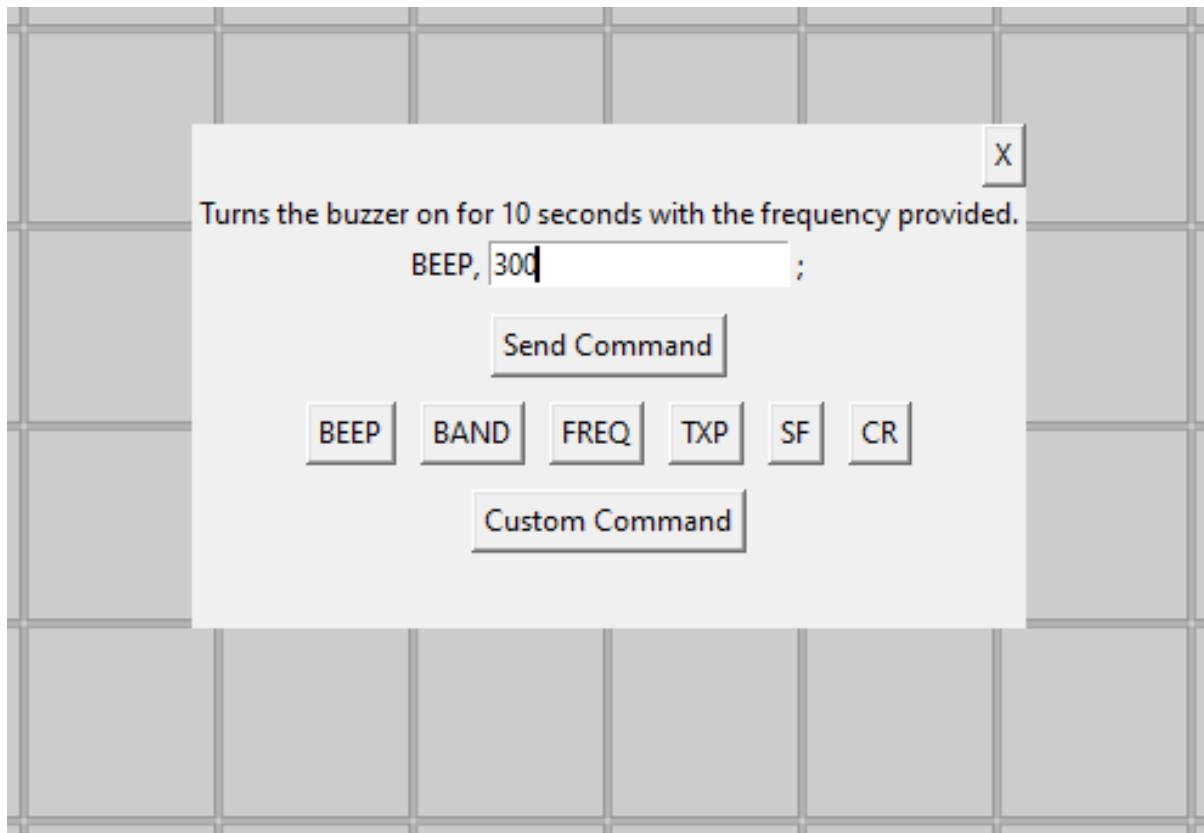


Figure 205: Uplink fra to send command to satellite. This will be sendt in the format "BEEP, 300;".

D Lab report voltage regulator testing

GALK|

D.1 Quick overview

The voltage regulators are tested in a lab. We have tested their ability to power to digital circuit and verify the capability of the regulator components to handle noise and possibly power surges.

D.2 Equipment

We have used the following equipment for the volt regulator testing:

1. +3,3 V regulator: BA33BC0FP-E2
2. +5 V regulator: BA05CC0FP-E2
3. breadboard
4. Capacitors: polarized 470 nF & 330 nF
5. Multimeter (Volt meter)
6. Power supply
7. Oscilloscope

D.3 Procedure

The volt regulators needs to maintain the voltage output during idle and load of the electronics. We test the voltage output with no load and with load in this lab test. The volt regulators are connected with quick clamp connectors for the pins, and the breadboard are used for connecting the thru pin capacitors we had available in the lab at hand. We are looking for noise in the oscilloscope display or wrong output voltage. As a final test did we use the oscilloscope with a probe that has a ground connector at the end and we hook it in a loop. This way we can listen to noise generated in the circuitry.

D.4 Results

The test results are shown in figure(206) where the breadboards, oscilloscope, power supply, and components are seen connected at the lab bench. We can see in figure(207) the voltage being measured to be 3,18 V and 4,99 V, which is well within acceptable levels. The power supply is running at 7,2 volts and no current limit. The current is drawn at 0.13 A according to the power supply internal Ampere-meter

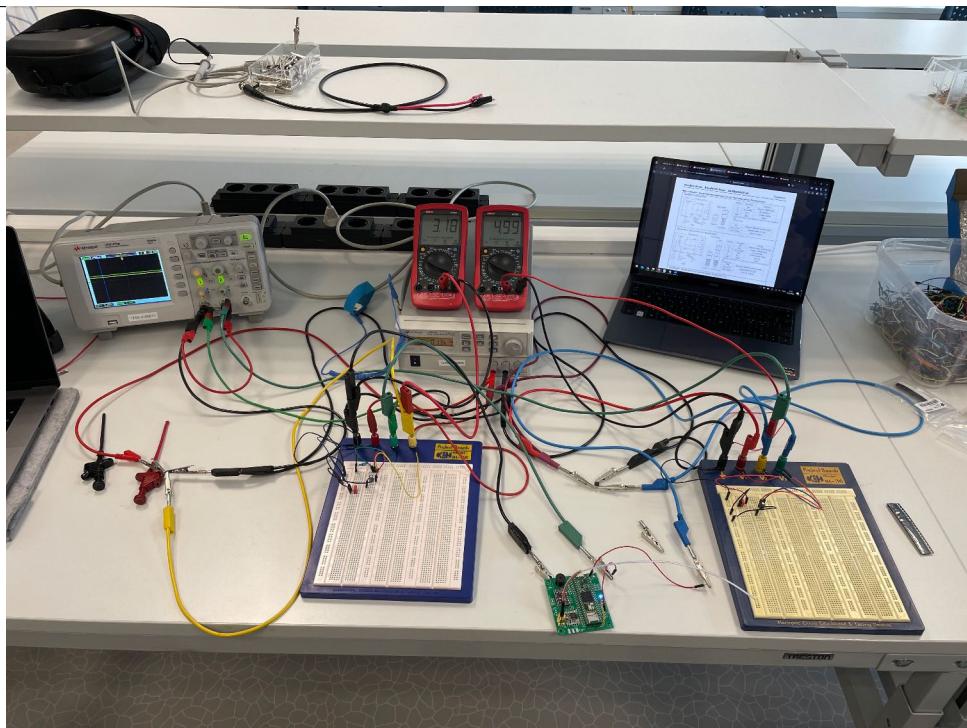


Figure 206: Lab setup for volt regulator testing

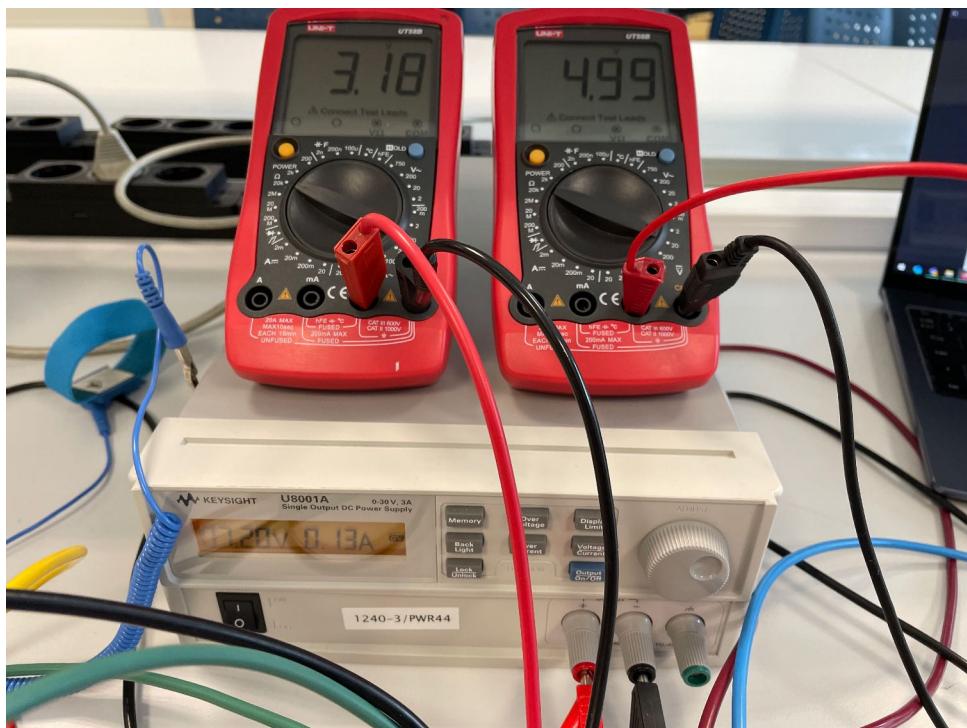


Figure 207: Volt regulator output measurement

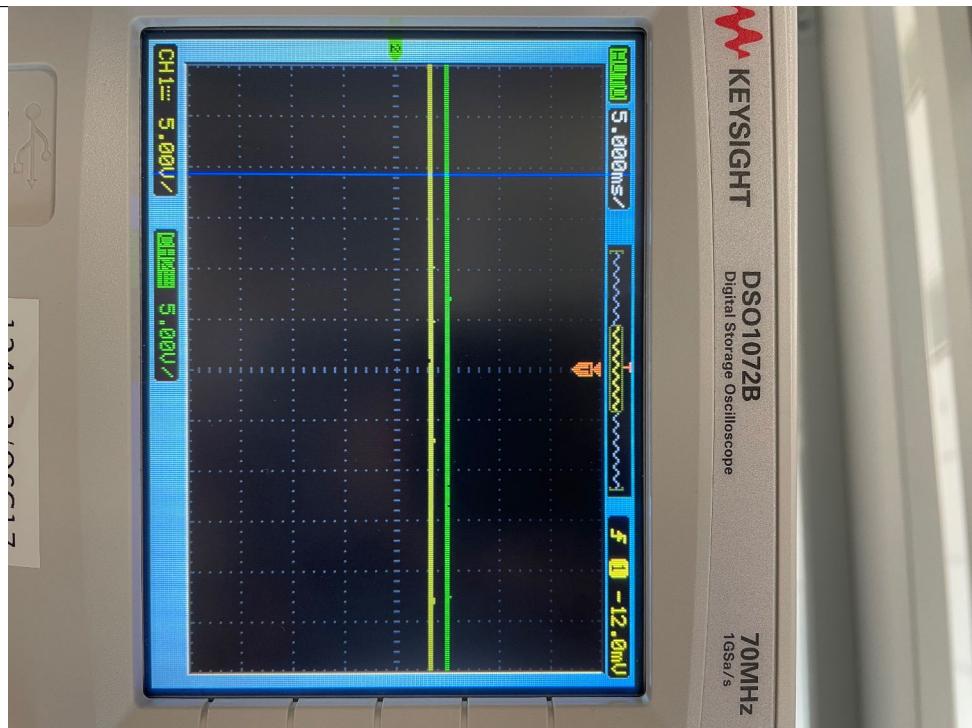


Figure 208: Oscilloscope readout

E Lab report antenna testing

GALK|

E.1 Quick overview

The C.R.A.F.T-Sat antenna has been tested in a lab to verify the antenna design, material, transmitted frequency, connection, and signal strength. This is done by transmitting a signal at the desired frequency and listen to the signal with a spectrum analyzer.

The following equipment is used in the testing:

1. Spectrum analyzer
2. Receiver antenna
3. EMC Faraday cage (Optional)
4. CraftSat antenna
5. Shield with Teensy 4.1 (Or similar micro controller)

E.2 Procedure

The C.R.A.F.T-Sat antenna is seen if figure(209) and is connected to the SMA connector on the PCB.

To test the antenna did we run the code on the micro controller to transmit at the desired signal. The receiver antenna listens to the surrounding radio waves, so the spectrum analyzer is set to have a narrow bandwidth of frequencies we want to detect. In figure(212) is the spectrum analyzer set to listen from 430 MHz to 440 MHz.

This lab test was done using a Faraday cage to prevent external interference to the test results and the radio module. The Faraday cage blocks radio waves from entering and creating false signal spikes and can be seen as the metal walls in figure(210). We did tests at the end with an omnidirectional antenna which will receive signals from all angles in figure(211). The energy the omnidirectional antenna receive is less than the directional antenna in figure(210) and we detected a slightly weaker signal.

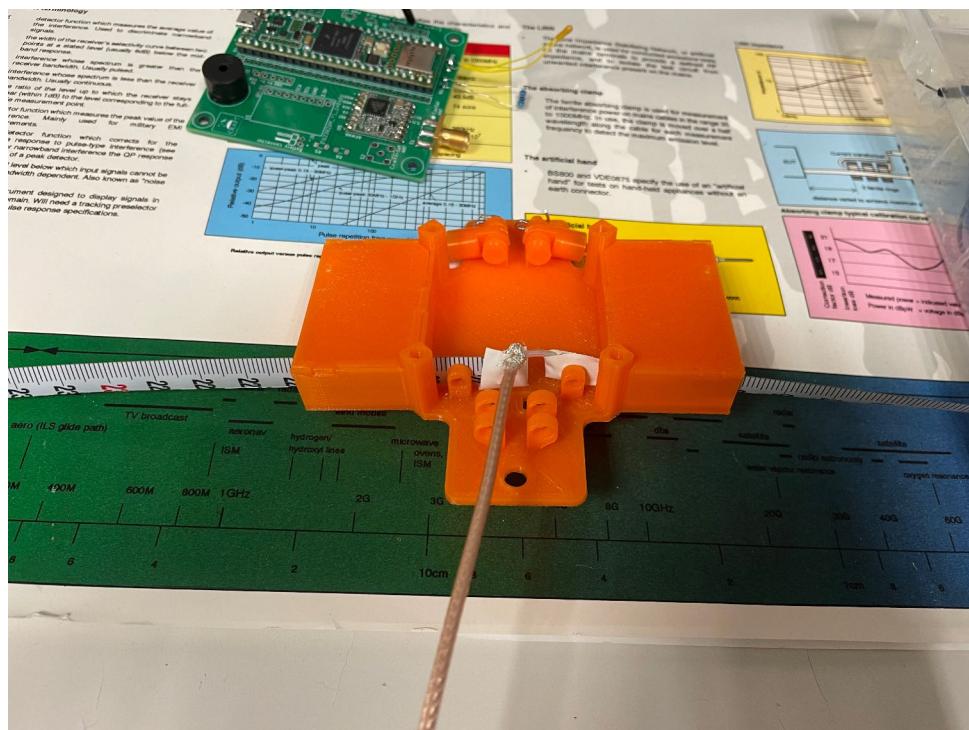


Figure 209: Antenna mounted in CraftSat for lab testing

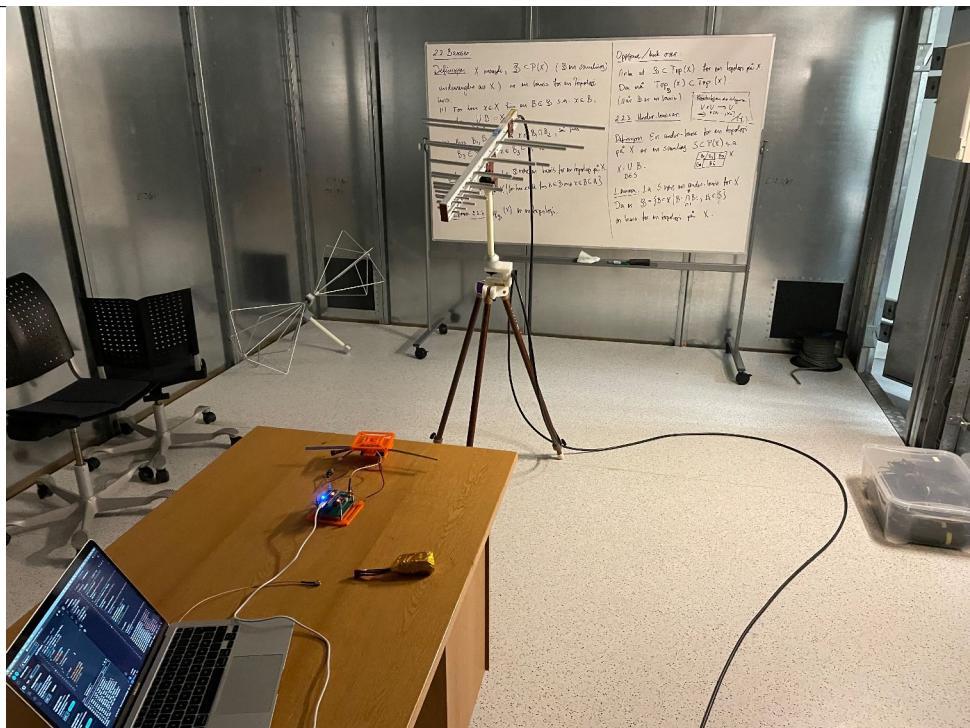


Figure 210: Antenna testing in Faraday cage

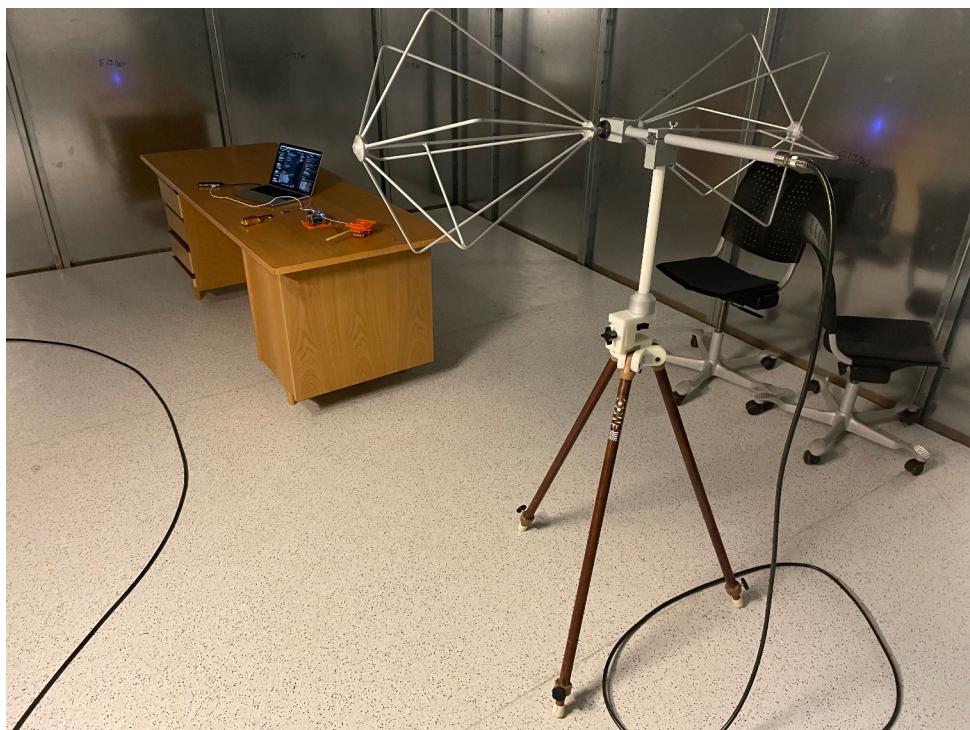


Figure 211: Omnidirectional listening antenna in Faraday cage

E.3 Results

The C.R.A.F.T-Sat antenna is transmitting near the receiver antenna and the read out on the display of the spectrum analyzer shows the signal spike at the transmitted frequency seen in figure(212). The spectrum analyzer seen in figure(213) is located outside the Faraday cage and we can read on the display the signal being transmitted. The antenna is working properly and is suitable for use in the C.R.A.F.T-Sat design.



Figure 212: Frequency analyzer read out

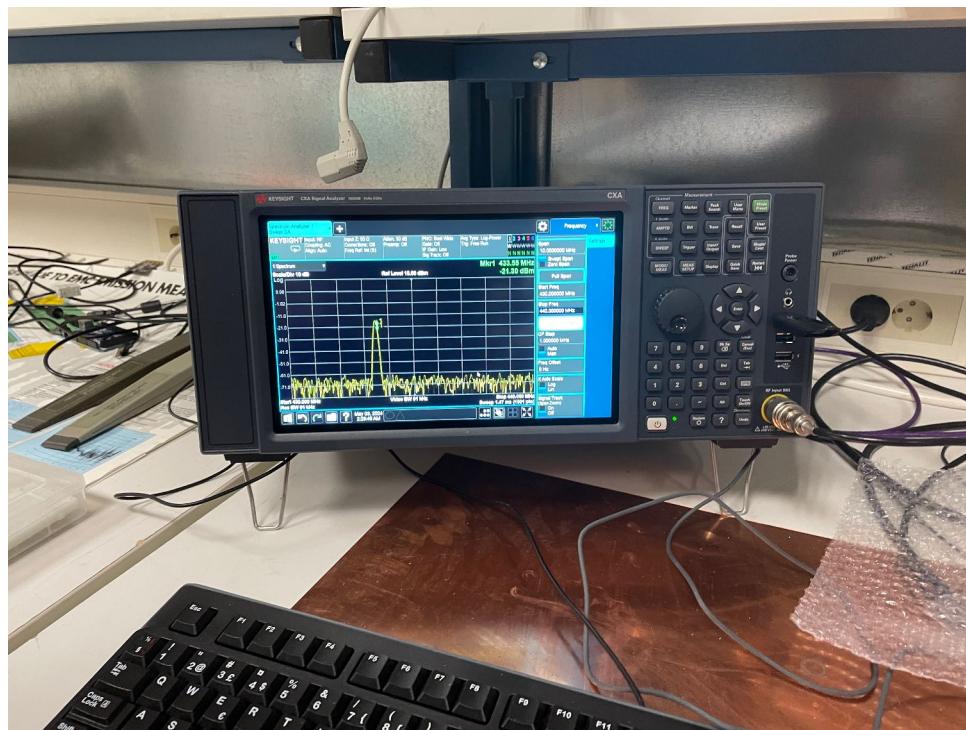


Figure 213: Spectrum analyzer used for antenna testing

F Desoldering of components

We have had a limited number of components, so the reuse of the main modules were necessary during the production of the shield PCB. The use of a vacuum pump desoldering station was required to clear the pins thoroughly of solder to remove the non functioning Teensy of the PCB. The desoldering station can be seen in figure(214) and being used in figure(215)



Figure 214: Desoldering station



Figure 215: Desoldering pistol being used to remove left over pins of non functioning Teensy

The process of desoldering the pin is quick with the vacuum pistol and is seen in figure(216), figure(217), and figure(218).

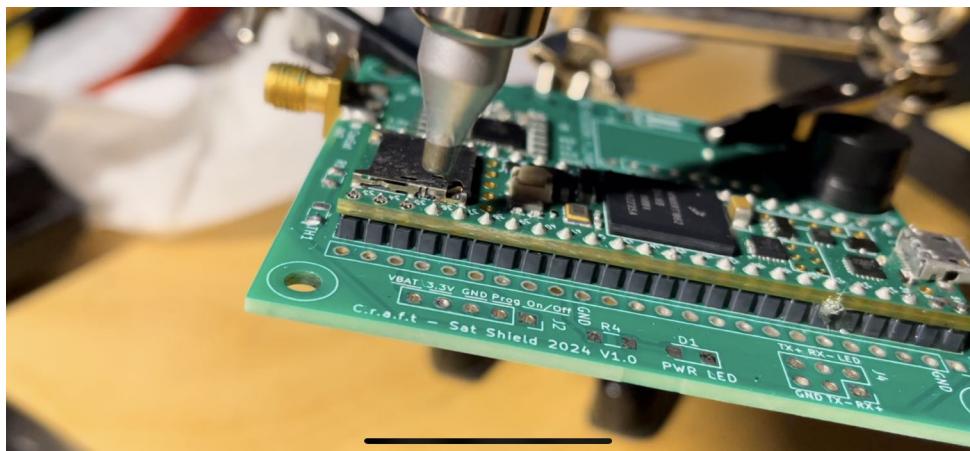


Figure 216: Desoldering of pins before

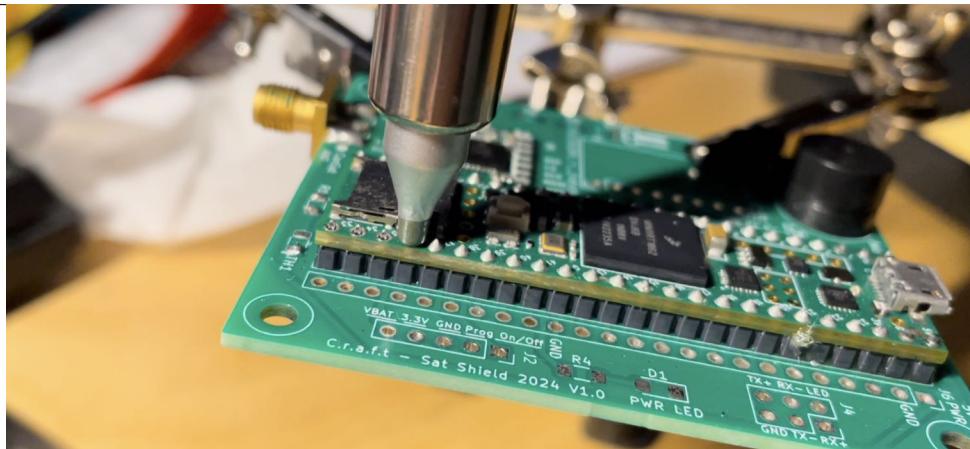


Figure 217: Desoldering of pins during

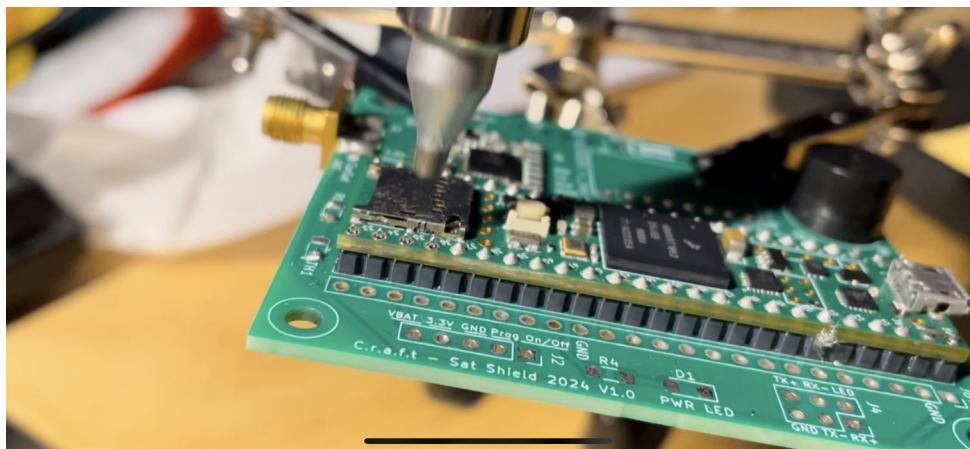


Figure 218: Desoldering of pins after

The desoldered Teensy is powered for testing by the computer USB in figure(219). The radio which is seen on the PCB in figure(219) is desoldered from an earlier prototype board seen in figure(220).

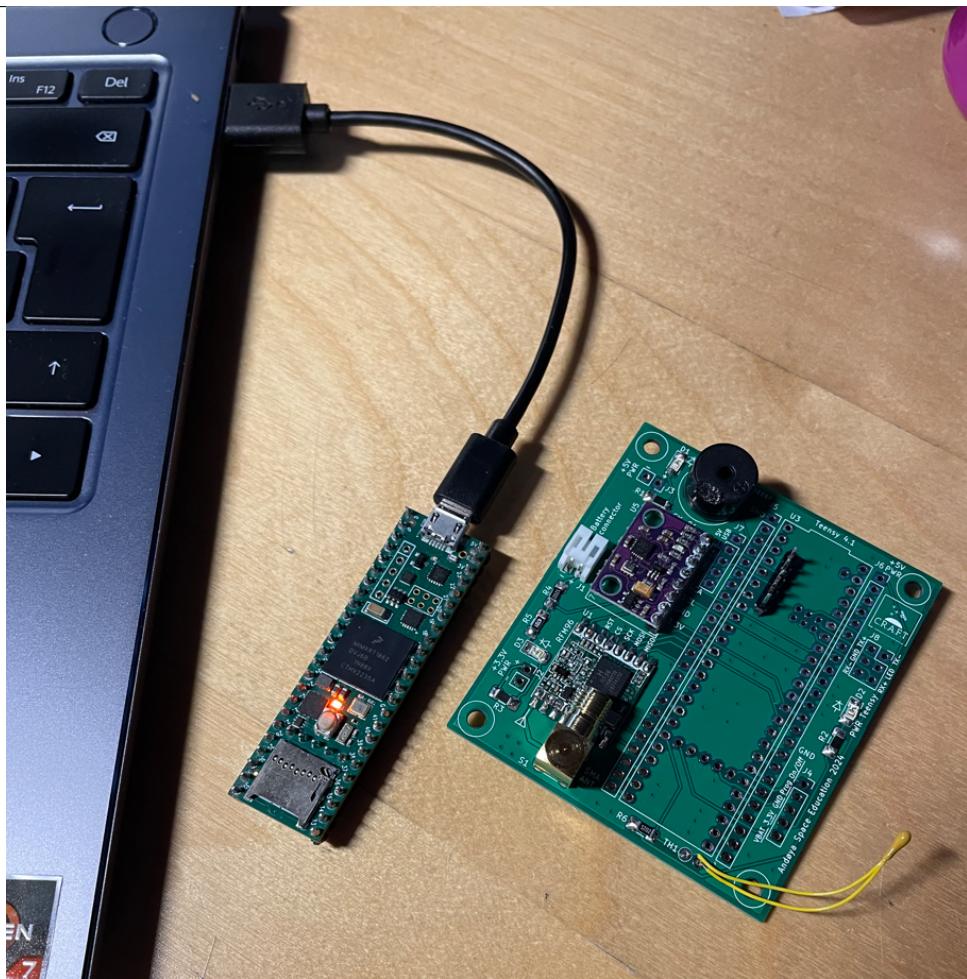


Figure 219: Non functioning Teensy removed for testing and replacement

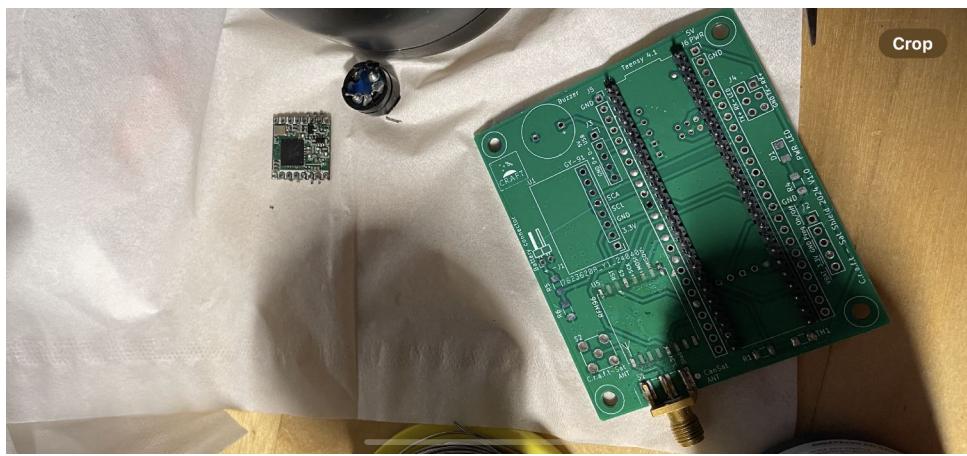


Figure 220: Radio module after desoldering from prototype board

GALK|

G Burn-Wire Prototype

This appendix section is a quick overview of the burn-wire development before verifying the design for PCB ordering

The development of the burn-wire circuit was done through theory, breadboard testing and simulation. The built breadboard prototype is both preformed early with a potentiometer, and later with soldered components to wires for verification. Figure(221) shows the bread board prototype being built with the use of a potentiometer to control a DC voltage directly to the base pin of the transistor for current control. Figure(222) shows the op-amp being prepared with wires to connect it to bread board. Figure(223) shows the pre loaded spring connected between a fishing line and the nickel chromium wire. Figure(224) shows the components ready and connected at the breadboard. Figure(225) shows the burn-wire being ran by an Arduino Mega to generate the PWM output, where the multimeter is reading the desired current output of 1,167 Amperes.

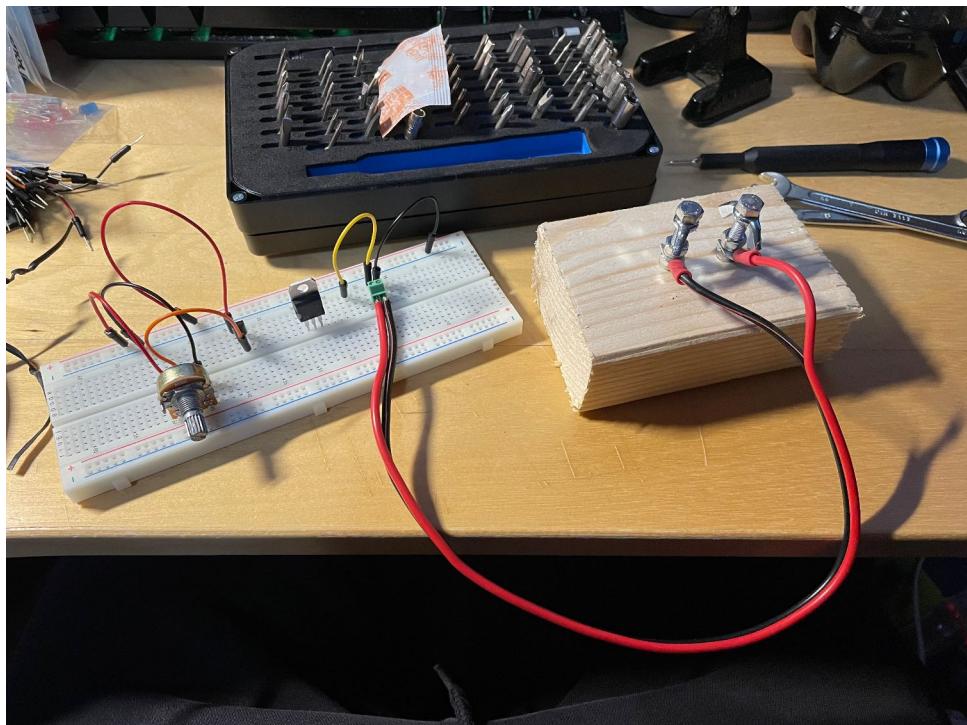


Figure 221: Burn-wire prototype with potentiometer for DC volt controll being built.

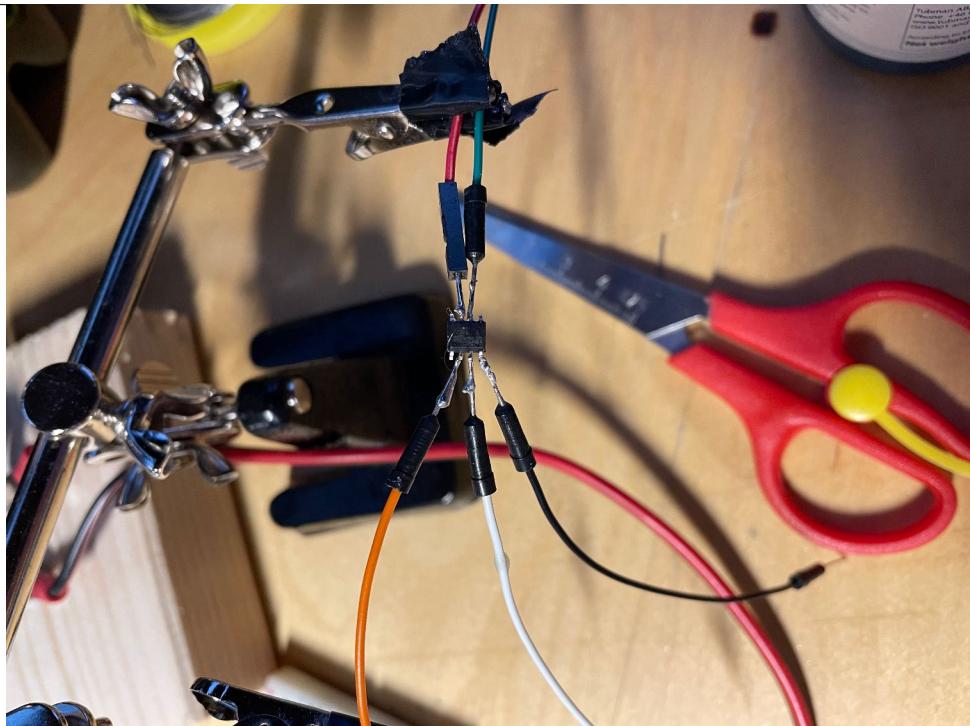


Figure 222: Op-amp with SMD pins soldered to wires for breadboard testing

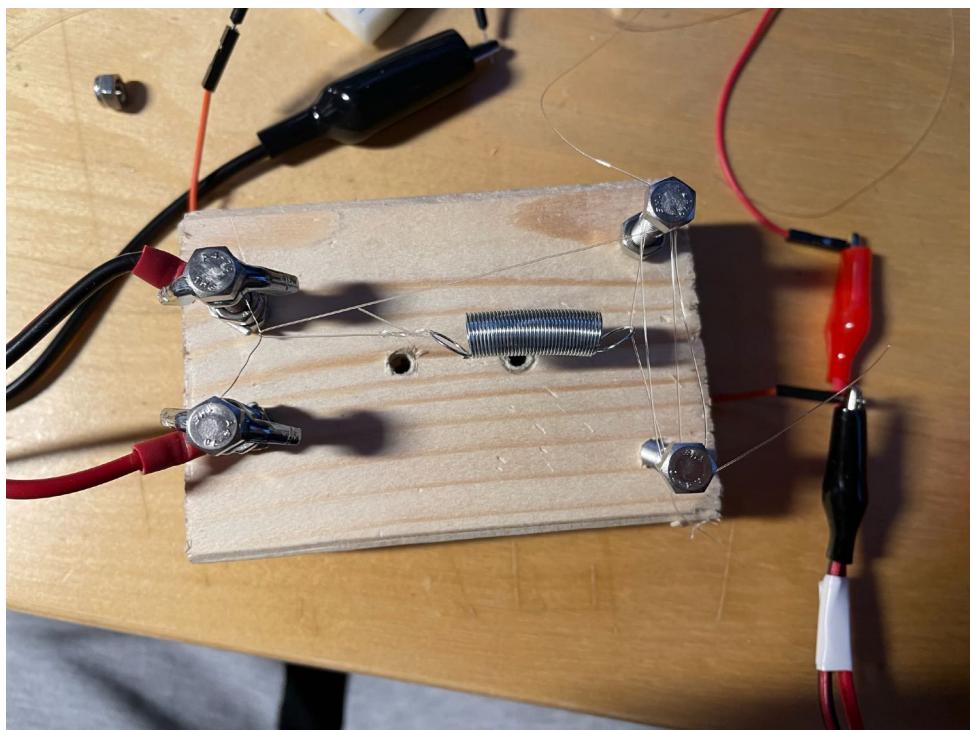


Figure 223: Burn-Wire test release mechanism

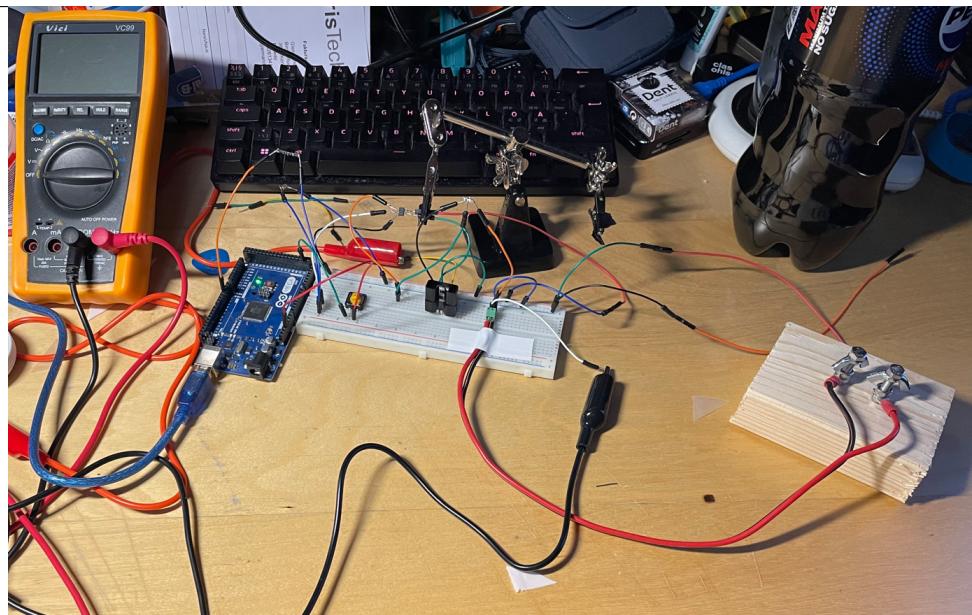


Figure 224: Burn-Wire bread board prototype with components

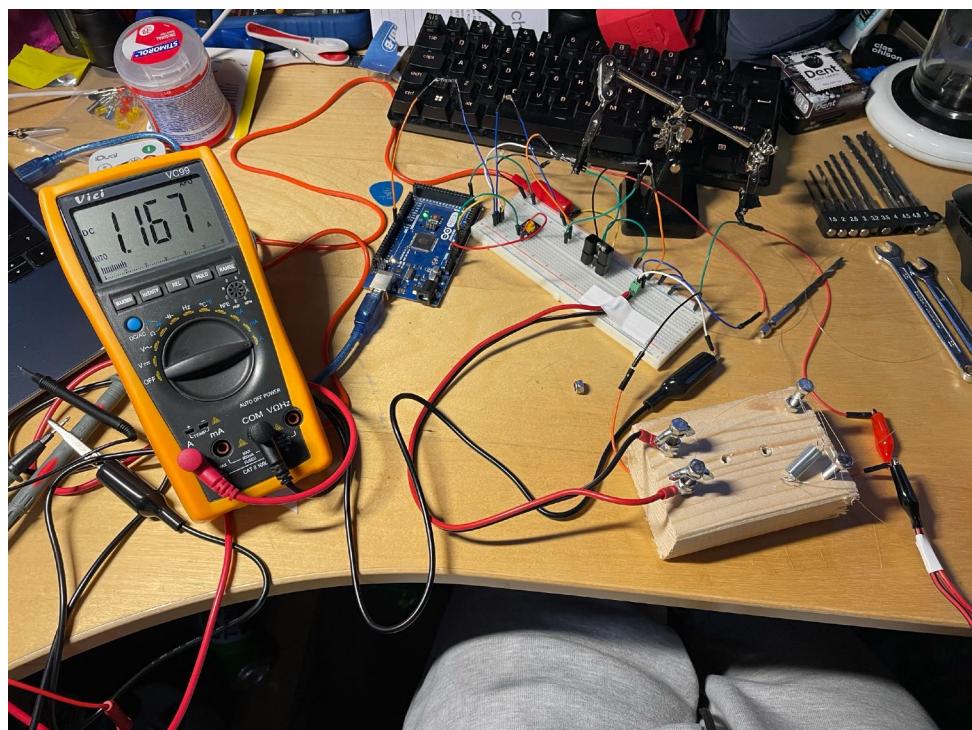


Figure 225: Burn-Wire in active state

G.1 KiCad

KiCad is a software for electronic design automation that was first released in 1992 by its original author Jean-Pierre Charras. It is now managed by the KiCad Development Team and is available as a free and open-source tool. KiCad allows users to design schematics for electronic circuits and convert them into PCB. It is designed to be user-friendly for inexperienced users but also offers advanced features for more experienced users and complex projects. KiCad is accessible to anyone who wants to use it[128].

G.1.1 Software Structure

Schematic Capture: KiCad enables users to create schematic diagrams of electronic circuits at various complexity levels. Users have the flexibility to customize the components available in libraries or design their own components and libraries. This is particularly useful for designing components according to specific standards, such as Norwegian standards for resistors, capacitors, and conductors. Once all the necessary components are drawn and connected, the schematic can be taken to the PCB layout stage[129].

Spice Simulation: KiCad includes a Spice simulation feature that allows users to analyze and test the behavior of the circuits they have created. This simulation can be done before physically building the circuits, providing a way to verify their expected behavior[130].

PCB Layout: After completing the schematics and verifying the desired behavior through simulations, the next step is to design the physical layout of the PCB using the PcbNew tool in KiCad[131].

Footprint: The footprint is a crucial aspect of PCB design as it represents the physical outline and pin configuration of electronic components placed on the PCB. KiCad provides tools for creating and managing footprints to ensure accurate placement of components.

3D Viewer: KiCad includes a 3D viewer tool that allows users to visualize their projects in a physical manner when needed[132].

Gerber Viewer: The Gerber viewer tool in KiCad is used to view Gerber files, which are the standardized format for describing PCB layouts in manufacturing. By using the Gerber viewer, users can examine their Gerber files and verify their accuracy before sending them for manufacturing[133].

G.2 Teensy 4.1

G.2.1 Description of Teensy 4.1

The latest microcontroller developed by PJRC is called the Teensy 4.1, which is a continuation of the Teensy 4.0 platform. This board is known for its compact size and impressive features, which will be discussed in more detail later on. The Teensy 4.1 offers a significant improvement in processing power, making it one of the fastest microcontrollers available today. As a result, it is suitable for a wide range of applications, both for beginners and advanced users. Despite its small size and light weight, the Teensy 4.1 offers enhanced connectivity options and memory capabilities. This makes it an excellent choice for projects like CanSat, where size and weight are important considerations[134][138].

G.2.2 Key Features of Teensy 4.1

1. Processor and Speed:

- The **Teensy 4.1** is equipped with an ARM Cortex-M7 processor running at 600 MHz, making it one of the fastest microcontrollers available on the market[134][138].
- It offers high-performance computing power suitable for various applications[134][138].

2. Memory:

- It has 1024 kB of RAM (2x512 kB tightly coupled) and 8 MB of flash memory[134][138].
- The memory is expandable with an SD card slot for additional storage, making it suitable for storing data (pictures, videos, etc.)[134][138].
- Provide locations to solder on two additional memory chips, one for PS RAM and the other for Flash[134][138].

3. Ethernet Capability:

- If desirable, it can feature 10/100 Mbit Ethernet capability for network-supported projects. This feature is helpful for IoT (Internet of Things) applications[134][138].

4. Programmability:

- It is compatible with the Arduino IDE, which makes programming accessible to a wide range of users[134][138].
- It supports Python programming languages through CircuitPython. It is important to notice that CircuitPython does not completely support all Teensy 4.1's hardware[134][138].

5. Audio Capability

-
- **Teensy 4.1** offers an advanced processing capability with a dedicated audio library designed by PJRC. This can be suitable for audio synthesizing projects and real-time audio processing[134][138].

6. Size and Form Factor:

- It has a compact size (60.96 mm x 15.24 mm), making it a perfect choice for projects where space is a constraint (optimal for CanSat)[134][138].

G.3 Buzzer

GALK|TL

The buzzer is a small component with the ability to take an electrical signal and convert said signal to sound waves for humans to hear [147]. The buzzer will have the best capability to be heard when placed near an opening of the CanSat and CubeSat.

G.3.1 Results from low pressure testing

The buzzer was exposed to the low atmospheric pressure in a test chamber to verify its survival capability from the mission. No deformation occurred at the

G.3.2 Gyroscope GY-91

RE|TL

The GY-91 module is a popular choice for a wide range of electronics and robotics applications due to its ability to integrate multiple sensors. It includes the BMP-280 barometric pressure sensor and the MPU-9255 sensor package[135].

G.3.3 BMP-280 Barometric Pressure Sensor

The BMP-280 Barometric Pressure Sensor, developed by Bosch Sensortech, is a highly accurate sensor specifically designed for measuring barometric pressure. It is commonly utilized in various applications that necessitate precise atmospheric pressure measurements, including altitude estimation and weather monitoring[113].

1. Pressure Measurement:

- It provides highly accurate pressure readings, making it useful for altitude tracking and weather prediction applications[113].

2. High Precision:

- It offers high precision with low absolute accuracy error[113].

3. Low Power Consumption:

- It is designed for low power consumption, making it suitable for battery-powered devices[113].

4. I²c and SPI Interferences:

- It supports I²c and SPI communications, ensuring compatibility with various microcontrollers (including **Teensy 4.1**)[113].

G.3.4 MPU-9255 Multi-Sensor Package:

The MPU-9255 is a multi-sensor package that combines two chips (MPU-6515 and AK8963) that integrate a 3-axis gyroscope, a 3-axis accelerometer, an advanced onboard Digital Motion Processor, and a 3-axis magnetometer motion tracking[114].

1. 3-Axis Gyroscope:

- It measures rotational motion and changes in orientation along the x, y, and z axes[114].

2. 3-Axis Accelerometer:

- It detects linear acceleration in three dimensions, providing data on motion, shock, and vibration[114].

3. 3-Axis Magnetometer:

- It offers compass-like functionality, detecting magnetic fields in three directions[114].

4. Digital Motion Processor:

- It includes an onboard Digital Motion ProcessorTM (DMPTM) that is capable of processing complex algorithms[114].

5. I²c and SPI Interface:

- It is compatible with both I²c and SPI interfaces for flexible communication with various microcontrollers, including Teensy 4.1[114].

G.3.5 NTC Temperature Sensor

GALK|GALK

The ability for electronics to measure the ambient air temperature can be done with a thermistor [144]. A thermistor is an oxide semiconductor where temperature changes will affect the resistance properties. The type we are asked to use by ASE is called NTC. When temperature in the air varies does the resistance value change over time, and we are able to measure these changes with the micro controller to get a very accurate measurement of the air near the sensor. The placement of the sensor is therefore important so electronics which create heat doesn't affect the readings from the sensor.

The sensor requires a power source to have the temperature resistance read by the micro controller. The sensor might not have linear properties, but adjustments to the measurement code and calibrations can increase the accuracy of the sensor.

G.3.6 RFM96 Radio

The **RFM96 Radio** is a wireless transceiver that is designed for long-range wireless applications, such as Internet of Things (IoT), remote control systems, and other projects. It is known for its ability to maintain a reliable wireless communication over long distances, potentially spanning several kilometers, depending on factors like the environment and antenna configuration[139][136].

1. Chip:
 - The RFM96 utilizes the **SX127X** by Semtech the developers of LoRa.
2. Frequency Range:
 - **RFM96 Radio** typically operates in different frequency bands, but the unit utilized in the project will be restricted to 433 MHz by various filters. This feature makes it eligible for use in various countries, including Norway.
3. LoRaTM Technology:
 - It employs LoRaTM (Long Range) modulation technology, providing significant advantages in long-range capability and power consumption.
 - Using LoRaTM technology also enhances its ability to resist interference and maintain a stable connection in a noisy environment.
4. Transmission Power and Sensitivity:
 - It offers relatively low transmission power (+20 dBm, and 100 mW maximum) and high receiver sensitivity, enabling long-range communication.
5. Low Power Consumption:
 - It is designed especially for low-power operations, making it an excellent choice for battery-powered applications.
 - It offers Sleep-mode for even less power consumption in inactivity case.
6. Data Transfer and Modulation:
 - It supports a wide range of data transfer rates and other types of modulations, such as frequency shift keying (FSK), gaussian frequency shift keying (GFSK), minimum shift keying (MSK), gaussian minimum shift keying (GMSK), and on-off keying (OOK).
7. Programmable Parameters:
 - It allows for programmable bandwidth and coding rate, giving users control over various aspects of communications.
8. Interface and Programmability:

-
- It typically interfaces with microcontrollers (such as Teensy 4.1) via SPI, which is common and easy to use.
 - It can be programmable through simple commands, allowing for easy integration into a wide range of electronic projects.

9. Applications:

- It is ideal for applications that require long-range wireless communication, such as weather monitoring and remote sensing [139].

G.4 Backup antenna

While the dipole antenna offers numerous advantages over other options, its implementation is not without risks. The worst-case scenario involves a malfunction or complete failure of the release mechanism. In such a situation, it is crucial to ensure the continuous transmission of signals. Therefore, a backup solution must be in place to guarantee signal transmission even under adverse conditions. Following a detailed discussion, one potential solution is to use an antenna whose length corresponds to the Craftsat frame. Based on results in (41) and (42), the antenna length for quarter-wave dipole antenna would be 173.21 mm and 86.60 mm for frequencies of 433 MHz and 866 MHz, respectively. For the half-wave dipole antenna, the antenna length, based on results in (39) and (42) is 346.42 mm and 173.21 mm for frequencies of 433 MHz and 866 MHz, respectively.

G.5 Gantt diagram

GALK|GALK

A gantt diagram is a chart where the plan for a system under development will track the previous, current and future phases of the progression. [143] A need for a gantt diagram can be not always so clear when we use an agile development method, due to the issue with misunderstanding the use of the diagram and it having a waterfall type of development plan without reiterations. Our goal is not to make a firm decision and stick with it through the development, but work agile and make changes as more knowledge is uncovered about the product to mend a suitable final product. Therefore our project will not live by the gantt diagram, but utilize it for the overview and time schedules we are following.

A few examples from our gantt diagram are "make budget plan", "order equipment", and "recieve feedback from stakeholder". Making a budget plan in the beginning where many unknowns about the project are still present, will not be a correctly planned budget. Reiterating a few times down the line will be necessary to get a more precise cost for all parts, yet the gantt diagram is written to have the budget plan early. Ordering equipment is most likely also not a single action where everything is ready right at the same time, as overlooked areas for the project can create the need for more components and parts. The budget and ordering equipment are two topics closely related when planning the project, yet keeping to the plan firm could prove to be difficult. Receiving feedback from stakeholders is a repeating process which appears multiple times through the gantt diagram as a final step in the different phases.

[NB! Gantt diagram is currently too large to fit inside this document. Will be resolved at a later stage]

G.6 Project Progress Report

RE|RE

This section summarizes the advancements achieved by the entire group. It emphasizes the interdisciplinary team's collaborative work in **design**, **development**, and **preparation** for implementation. We have adopted an **Agile** approach to ensure a flexible and responsive process that effectively addresses the project's demanding requirements.

G.6.1 System overview

We have created a basic description of the CanSat's system as shown in Figure 226 made in Microsoft Visio, which showcases the structure of our CanSat under operation. It highlights the essential components of the system and their interaction with the ground station, underscoring the significance of communication and testing throughout the CanSat's life cycle.

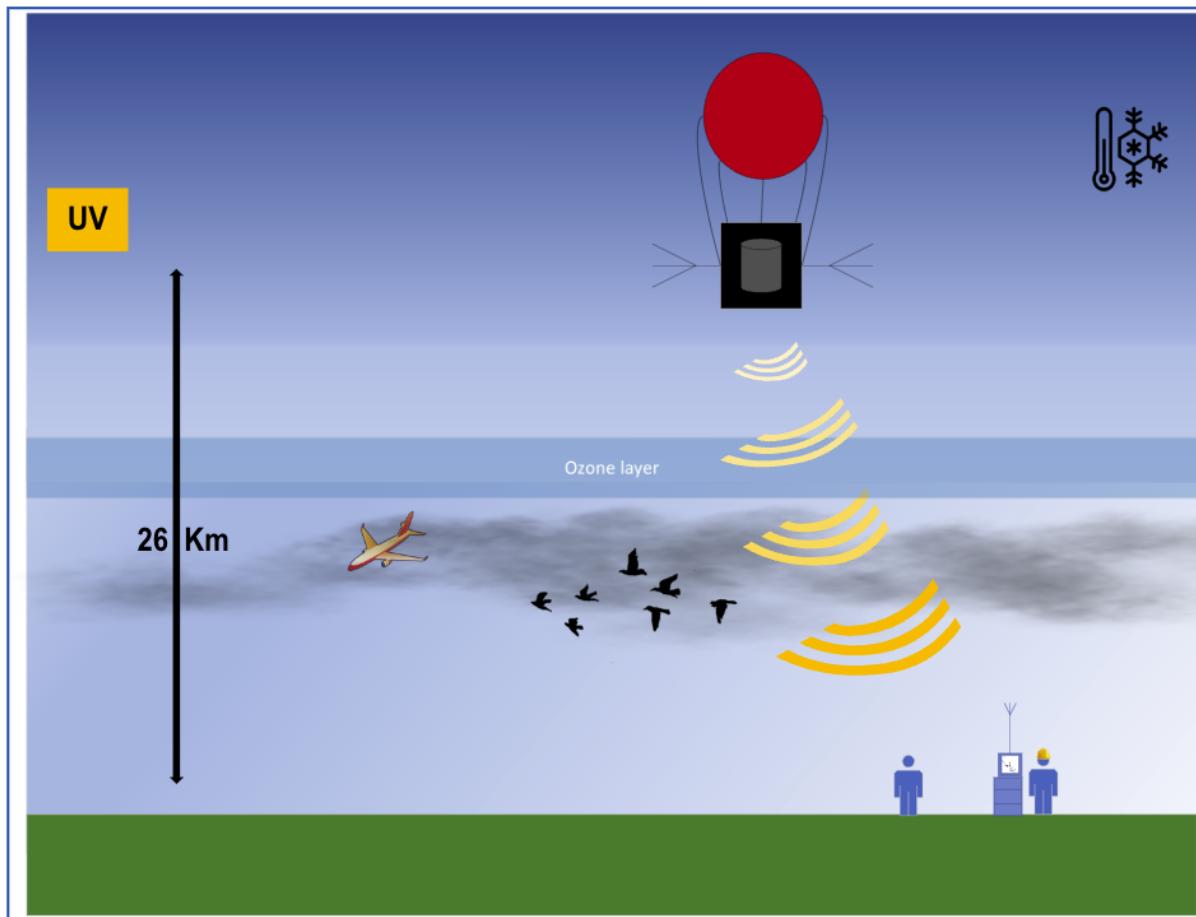


Figure 226: CanSat's Overview Under Operation | NNG

G.6.2 Project Management and Planning

A comprehensive project plan has been created using the **Gantt diagram**, which visually represents the different stages and significant events in the project. The utilization of

this tool plays a vital role in coordinating interdisciplinary tasks and maintaining project progress.

G.6.3 The Progress in Electronic System

We have addressed ASE's request to reuse the sensors from the CanSat 2018 project by conducting a thorough analysis of the main components. To demonstrate the connections between these components, we have created a block diagram as shown in Figure 227. Additionally, our report provides detailed technical specifications for each component and includes the physical dimensions of all components which summarized in Figure 228 which also illustrates the potential locations of these components on the planned circuit board. Furthermore, to assist our mechanical engineer in making a decision regarding the antenna release mechanism, it was necessary to find a UHF antenna that could serve as a foundation for this mechanism. After evaluating and eliminating several antenna options, we utilized **MatLab** software to design and analyze the remaining antennas. Our focus during this process was primarily on studying their properties, such as gain and radiation patterns.

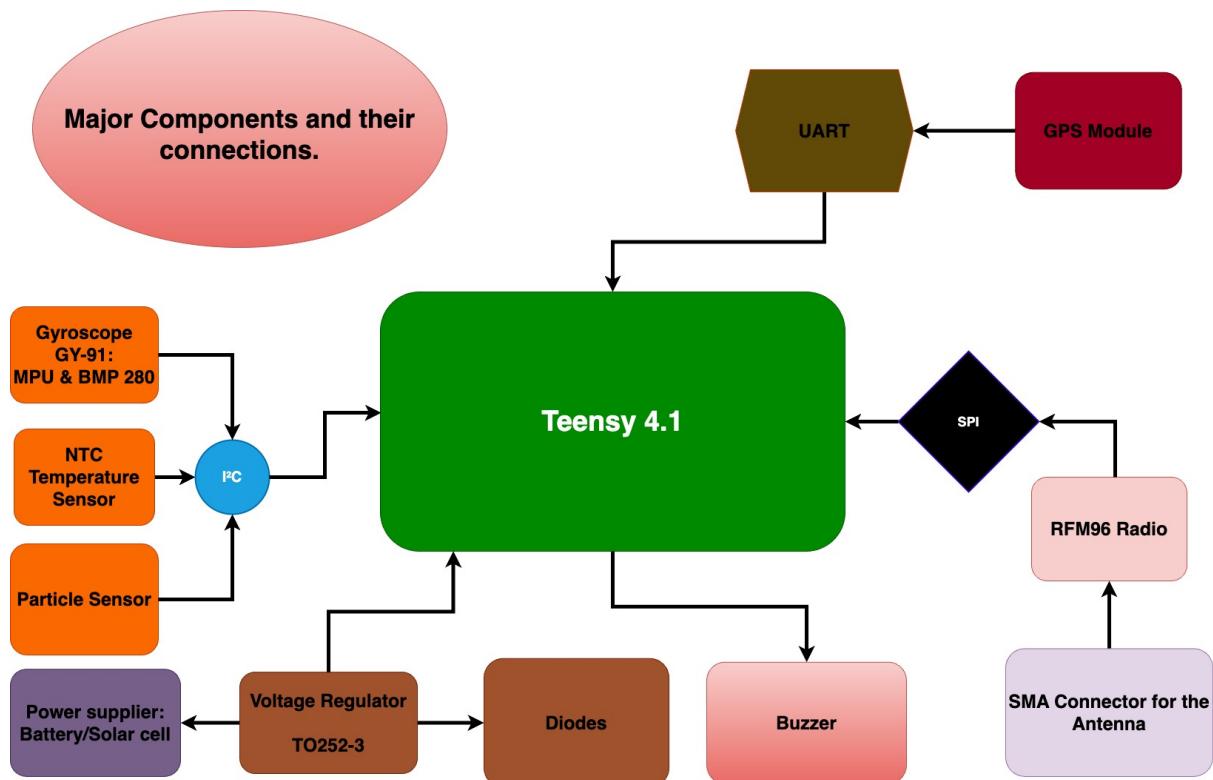


Figure 227: Major Components and Their Connections.

G.6.4 The Progress in Data System

Regarding the data, we have expanded our understanding of **tKinter**, a Python library utilized for constructing **GUI**. Our specific focus was on the functionalities that are essential for our project, such as real-time data plotting, displaying data streams, and managing file storage. These features are crucial in facilitating the interaction between

Technical Budget – Circuit board size

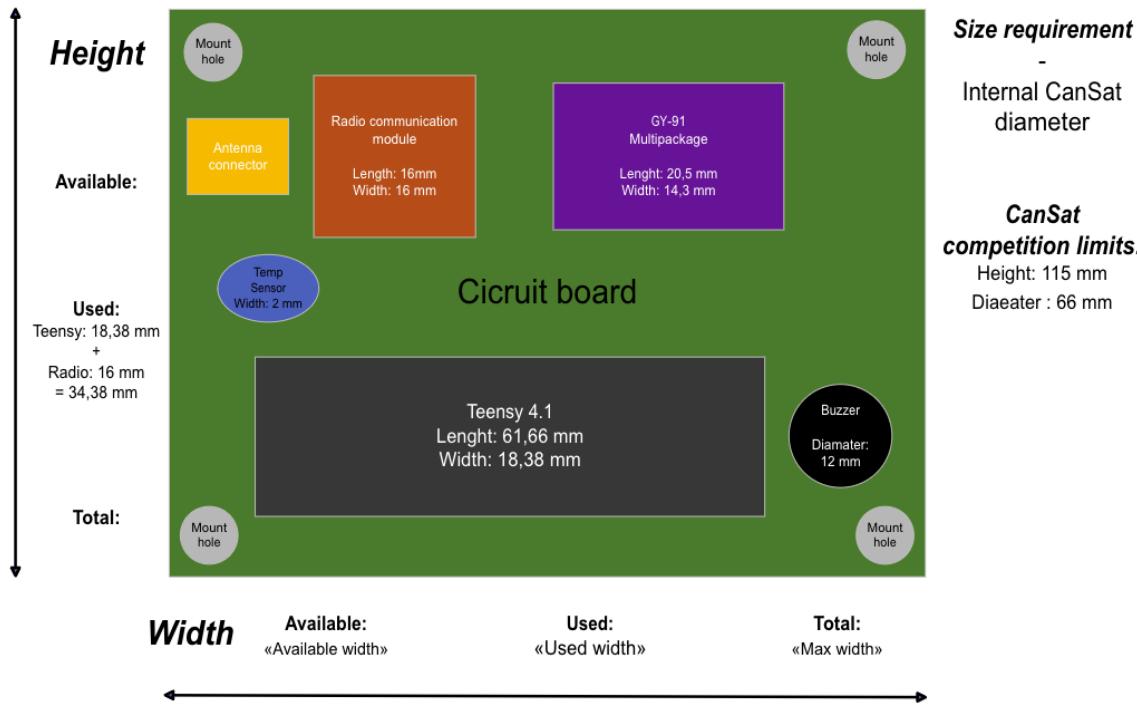


Figure 228: Circuit Board And Sensors With Physical Dimensions and Placement.

ground station operators and the CanSat, as well as monitoring its status. Additionally, we aimed to enhance the processing and analysis of the data collected by the CanSat's sensors. Initially, we concentrated on investigating the components of the GY-91 gyroscope sensor, which are instrumental in providing the CanSat with knowledge of its surroundings and position. Furthermore, we explored different methods of integrating the sensors with the primary microcontroller platform, namely the **Teensy 4.1**. This integration is vital to ensure the CanSat's ability to acquire accurate and valuable data throughout its mission.

G.6.5 The Progress in Mechanical System

As a disciplinary group, we have taken significant measures to solidify our approach and planning. Initially, we have created a simplified version of the **Concept of Operation**, which provides an overview of how our system will operate within the defined environment and accomplish the intended mission. This simplified version will serve as a fundamental guide for all subsequent development phases. Additionally, we have established the **Scope of Work**, which clearly depicts the project's boundaries and specifies what is to be delivered. This document is crucial in ensuring that the team remains focused and avoids **scope creep**, which can lead to unnecessary complexity and delays. In the spirit of interdisciplinary collaboration within our team, we have engaged in discussions with one representative from each discipline to explore the **potential functional requirements**.

This collaborative effort has resulted in a more comprehensive understanding of the needs and possible solutions that may arise in different domains of the project. We have begun formulating the design requirements and to ensure successful integration and deployment in space, we have carefully selected specific points from the CubeSat design specifications to integrate into our design requirements. Lastly, we have initiated the development of the production systems, which will facilitate an efficient workflow from design to prototype completion and ultimately production.

H Burn-wire description "Final design"

The burn-wire mechanism is an easy method and normally used for realising the antennas on a CubeSat. We decided that burn-wire is the most appropriate function for releasing the antennas and a easy interface of work between engineering teams. Therefore, this appendix will describe a reason why we decided that burn-wire mechanism is a possible solution of antenna releasing under the definition of reusable, and how the burn-wire mechanism works on the final design.

H.1 Why is a burn-wire mechanism defined as reusable

When a burn wire is used, material will be removed. We define the material removed as an easy to replace and buy material. The definition about reusable doesn't include that every component inside a function has to be a infinite use and never replaced. We define the function of burn-wire and reusable because of easy less replacement of used components in early time. The only material which has to be replaced is a rope. To make this replacement accepted inside a function of reusable, we decided that the rope has to be cheap and it's the only component which has to be replaced during use in a short time. This is of cures a withstanding function about re-usability, To accept it, we use this quote as an description of accepting

Something reusable will always need to be replaced during a specified time, to give the producer a possibility to earn a reason of product. A telephone which is daily used will always need to be filled up with energy.

We can use the rope as an definition of a *energy*. When the rope is gone "*energy is used*", the rope has to be replaced "*energy has to be filled up*". The replacement cost a low amount of money and that is the reason we are using it as an energy.

H.2 How does it work with the sealings

The 266 is mounted on the *ZY walls - ref drawing* by using the *HingeLockPart* and *Lock1*. The sealings gets pushed open by the *Spring_asm*. To keep the sealing closed, a force parallel but in the opposite direction of the forces pushing the sealing open (the antenna and *Spring_asm* is the components pushing the sealing open) had to be made. Therefore, a rope with enough strength to withstand the pushing force is chosen and goes trough both holes on the sealings and meets at the burn-wire. The rope is tied on the burn wire. When the burn wire has a equal temperature to a specific temperature the rope is burned and the area where it's tided to the wire is gone. Since no force is holding the rope and according to the third newton law

For every action (force) there is an equal and opposite reaction

the sealing which was closed by ropes opposite force will open and the antennas will pop out. To use the release-mechanism one more time, place a new fishing rope inside the area where its designed to place it. Then, fold the antenna and push the sealing back to

the position where its closed. Use a *antenna stick* and place it on the trough the cuts on the sealing and antenna board to keep the sealing closed. Then, tie the ropes to the burn-wire (**it's a small area so take time don't rush.**

Note: keep the *antenna stick* on the C.R.A.F.T-Sat until it's cleared for launch. The reason are because it will withstand forces and make sure the antenna won't be released before use.

I The optimal material for the C.R.A.F.T-Sat using Additive Manufacturing method

Additive Manufacturing technologies has highly increased during the last decades. This results in many manufacturing devices for 3D printing. Since we don't develop a frame with new material, we searched on 3D printing materials related to CubeSat. There is a lot of factors to consider when we are selecting a material for the Additive Manufacturing method. The factors we have chosen for consideration is strength, weight, thermal properties, cost, and the requirements described in chapter 3. Since metal parts can offer high strength and stiffness we considering them to be suitable for the structural components, such as the frame, in the C.R.A.F.T-Sat.

To 3D printing with a metal material, we can't use the same FDM 3D printing which we are using for the prototype. The SLM (appendix L) is therefore a Rapid Prototyping machine we define as a more suitable 3D printing for metal materials such as aluminium alloys and titanium. Since we are not developing a frame with new material, we searched on printable materials for a CubeSat, we selected aluminium and titanium as two possible solutions using a SLM. One of the biggest difference between titanium and aluminium is the cost, and therefore, we will conclude that aluminum is most suited for 3D printing the frame.

J Fused Deposition Modeling (FDM)

FDM is a method of material extrusion where the material is extruded through a single or multiple nozzles where layer-by-layer joined together creates a 3D object. The definition of printing with layer-by-layer is that during printing, filament will go through an extrude where gears will control the feeding of filament. When the material is fed, it will go through a heating area and out of a nozzle. The filament which goes through the nozzle has a diameter depending on the nozzle diameter, and the velocity of the amount of filament goes through is decided of the printing program on the printing machine. Other settings such as printing velocity, bed velocity, etc. is decided on *the printing program, which in our case on the Elegoo Neptune 4 pro is fluid*. The object will be produced when the bed changing its position in the XY-axis and when a layer is finished the bed or nozzle head will change its position in the Z-axis to a new layer start position. This will be repeated until the sliced 3D object is finished. The printing method is an easy and dominates the society in 3D printing. [172]

The material used is thermoplastics (description about thermoplastics O.2) which is easy to get and polymer material such as PETG and PLA is an non-advanced printing material which only needs a printer with low-cost heated bed (its accepted with non-heating bed for PLA but it's recommended) and could be placed in an environment without ventilation, where if a thermoplastic such as ABS (Acrylonitrile butadiene styrene) which is the next easy printing polymer where a user needs ventilation because of toxic when printing.

K Cura; Slicer settings

The group has two different FDM printers categorized as easy-to-use and non-expensive printers. To produce a 3D printed prototype with a FDM printer, a G-code file has to be made to give the FDM machine information about each layer. The G-code is made with a slicer program in our case Cura, which converts a STL, OBJ, X3D, or 3MF file into a G-code. We are only using STL files when we are 3D printing.

Cura has a lot of settings that are categorized between "basic" and "all". Since a user of the C.R.A.F.T-Sat may have little experience with a general slicer program used for 3D printing; we have decided to inform settings we self-defined as constants in the production of the C.R.A.F.T-Sat and settings chosen as test-variables due to the effect of the material strength on the product.

K.1 Cura; slicer settings decided to be tested - variable

Since basic settings is chosen, the settings decided to be tested is the *infill pattern* and the *infill density*. The *infill pattern* describes how the internal structure (structure between the walls, and top/bottom walls) will be printed. By editing the infill pattern an effect on how the structure will absorb the stresses occurs due to how the design between walls is chosen. By researching we have decided that a gyroid and cubic pattern is more effective in gaining a higher strength related to the *infill pattern*. [175]

The *infill density* describes the amount of the volume between the walls the infill pattern will use. If the density is equal to zero, fig 229 there is no *infill* between the walls. If we decide to have a density equal to 50%, fig 230 shows that we have both mass and empty space between the walls. If we decide to have density equal 100%, fig 231 shows no empty space between the walls.

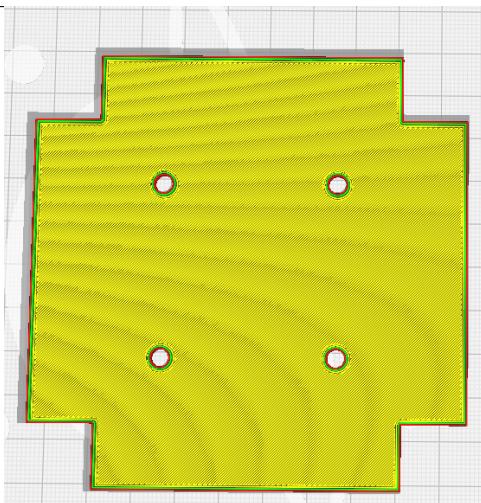


Figure 229: Density = 0% - Infill pattern = Cubic | STB

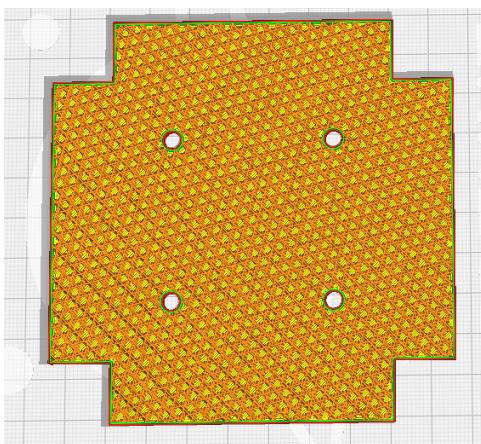


Figure 230: Density = 50% - Infill pattern = Cubic | STB

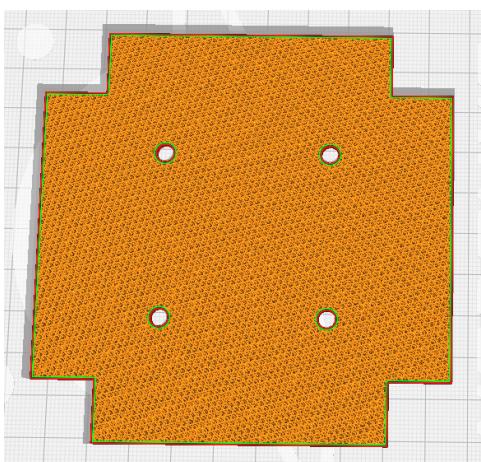


Figure 231: Density = 100% - Infill pattern = Cubic | STB

K.2 Cura; slicer settings decided to be constant

Since the slicing setting is decided to be basic, there is a small number of settings to choose from, but those settings are related to a simple 3D printed product. The first setting we decided to constant is the layer height. A layer height describes the thickness of each layer printed. By configuration testing of what values to use, we have decided to use 0.15 mm. The print result of the tension staffs, using this layer height resulted very well and it's the reason we have decided to use it as a constant.

The wall thickness is the distance between the outside surface and the infill pattern area. By selecting the thickness value for the wall, it will give an effect of at which point the stress will damage or break the product. If it's too thin it will be easier to break it, but if it's too thick it can lead to wastage of material and an increase of printing time. We decided to use a wall thickness of 1.2 mm due to print tests through the bachelor period and the given results.

The top/bottom thickness is the thickness before and after infill. You could call it; *the area where the product is "solid"* independent of the density value. Again, by manually testing, we have decided that the top/bottom thickness should be equal 1.2 mm (equal wall thickness). By calculation, we can find out how many layers the product will contain before it starts printing the infill pattern, which is equal to the amount of layers from the end of the printing the infill pattern to the end of printing product. The formula group made due to the given information is:

$$\text{Topp/bottom thickness} = \text{Thickness per layer} \cdot \text{amount of layer}$$

By editing this equation:

$$\text{Amount of layer} = \frac{\text{Topp/bottom thickness}}{\text{Thickness per layer}} = \frac{1.2 \text{ mm}}{0.15 \text{ mm}} = 8$$

This gives information about that the FDM will need to print 8 layers before it starts with the infill pattern at the chosen setting. We accept this amount of layers between the top/bottom surface and the infill surface.

The given information will be used for the polymer material PLA and PETG. The difference will be the temperature on the heat bed and the nozzle. When we are printing with PLA, the print bed temperature chosen is 60°C and a nozzle temperature of 205°C. For the PETG the bed temperature will be 70°C and a nozzle temperature of 235°C. Those temperatures are chosen due to different prints during the bachelor assignment. Supports with either normal or tree structure will be used in the production of the prototype components, but is decided to not be defined as a constant or tested. The support settings will therefore only be used when it's necessary.

L Selective Laser Melting (SLM)

SLM is Additive Manufacturing method involving high-powered laser to selectively melt a metal powder. The parts melted will fuse together layer-by-layer until the homogeneous model is complete. SLM includes a packed powder placed on the printing bed which will provide support to the model during the Additive Manufacturing process. Materials SLM process operates with are pure metals and alloys as aluminum, stainless steels, titanium, etc. [182]

An important notice during the Rapid Prototyping process; high thermal gradient between the molten part and the surrounding powder could result in stresses causing distortion and warpage. Therefore, during printing, support structure are often needed and used in order to dissipate heat from critical areas and fix/hold the part securely onto the build plate. The thumb rule is that overhangs or hollow structures which are angled between 0 to 45 degrees should be supported. The surface finish on a SLM part is rough and may need post-processing to achieve the surface desired, depending on the requirement for the product. [182]

M User manual - Assembly & disassembly the C.R.A.F.T-Sat

M.1 Description about this manual

This appendix includes the necessary items, 3D printing methods, and assembly sheets of the final product illustrating how you should mount the product.

When you need to disassembled the product, follow assembling in opposite steps.

M.2 Items needed to assembly C.R.A.F.T-Sat

- (a) Pozidriv screwdriver (PZ1)
- (b) Wrench; size 5.5 mm
- (c) Super glue (all super glue is accepted at this state).
- (d) 3D printed components
- (e) M3 bolts and locking nuts

M.3 3D printing: Manual and explanation - Final design

When we started with 3D printing the final prototype, different difficulties occurred. It was often based on the sizes from the CAD compared with the size the FDM is capable to use in the production of prototype parts. To succeed in Additive Manufacturing of the prototype parts we had to cut the larger parts in CAD to create new parts including only the areas where we want to control fits and printing method. Also, by removing mass of the large parts just to check specific points reduced the printing time and made print control more effective. We will therefore define the material cut for 3D printing control of specified areas as mass cut.

Note: PLA was used during the period of changing sizes and test if the chosen sizes of components is possible to produce. The settings used is equal the settings chosen for PETG. The chosen slicer settings and values for the not tested setting is described in appendix K.

It could be more effective if we had printed an object, measured and documented the values, calculated a scale difference in all three axis, and used the values instead of multiple prints. The scale is different from printer to printer since it depends on the accuracy in all axis of the printer used. But, since we want to have a more controlled production with fit, 3D print and check method worked with a idea of a better precision but with a consequence in a longer prototyping time before conclusion of the final prototype.

Part name: ZY_wall

The first component printed with mass cut was the *ZY_wall*. We printed five mass cuts of it's design and one final design without a mass cut. The reason we decided to mass cut this object is because the *ZY_wall* gave difficulties due to how it should be printed, since a area on the component is cut for the *Lock1*. The first version of *ZY_wall*, fig 232, has a specific cut for a *HingeLockPart*. The SW function extruded cut was also used to remove the mass but at the time it was done, it was difficulties in removing more mass. Therefore, normal supports was used on the sides.

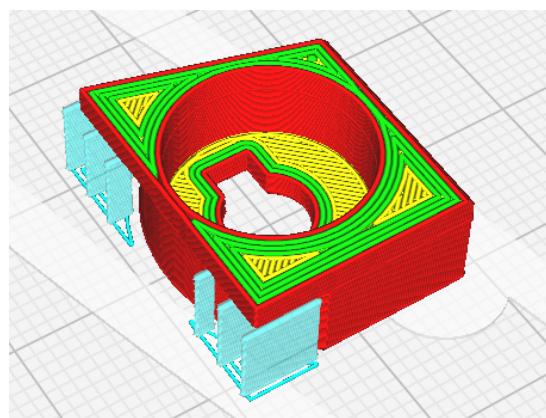


Figure 232: The first *ZY_wall* cut - version | STB

The second mass cut version, fig 233 regards change the volume of the mass cut on the *ZY_wall*. The reason was because print with the first mass cut ended with failures in the area with supports and at this state we decide to not print it in the opposite direction equal the real print. The reason was because the design of the *ZY_wall* was changed suddenly. The second cut still was designed for a circular *Lock1*.

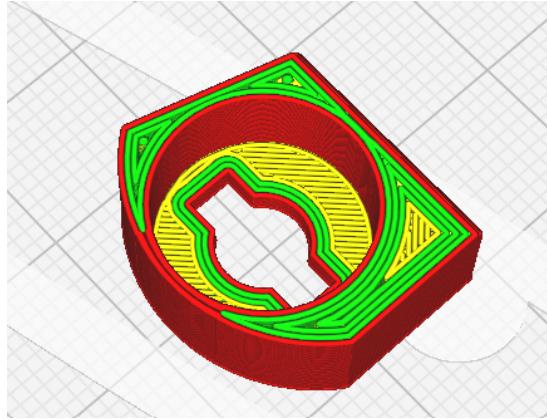


Figure 233: The second *ZY_wall* cut - version | STB

The third, fourth and fifth *ZY_wall* cut's versions was related to the thickness change of the first layers from $1 \rightarrow 3$ mm. The amount of layers necessary for printing in the area with the large cylinder was found and made the next version related to a cut on the part to stop *Lock1* for rotation.

The sixth mass cut version, fig 234 is designed with a rectancuglar cut for a *Lock1* design without fillets. The reason we chose it rectangular is because a *HingeLockPart* will be placed through the lower face and turned 90° clock wise or counter clock wise, then, the *Lock1* will be placed over it and secure that the *HingeLockPart* will not rotate and fall out in the same direction it was placed.

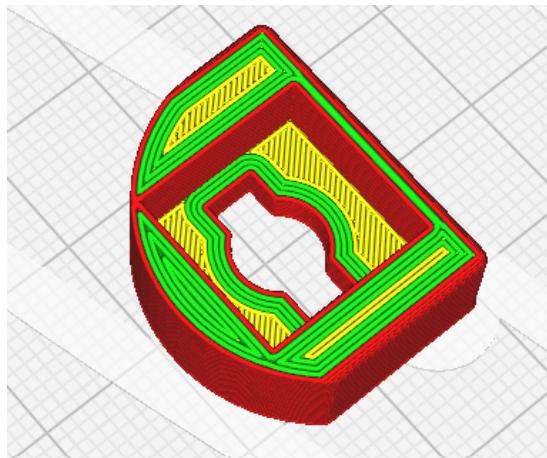


Figure 234: The sixth *ZY_wall* cut - version | STB

The sixth mass model version showed that it's good but should be re-modeled because of the edges close to the edges where no print is placed and gives a risk of a area with less/no filament/or a opening on the sides.

To solve the problem with edges the seventh design, fig 235 has fillets inside the rectangular area. This gives the area between the walls inside the rectangular hole and the outside wall a small space. It was also printed in the opposite direction of the other

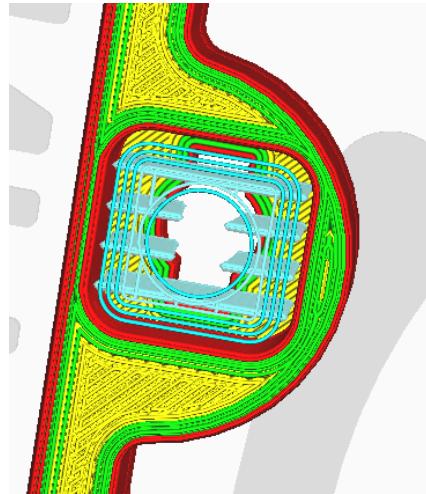


Figure 235: The seventh *ZY_wall* cut - version | STB

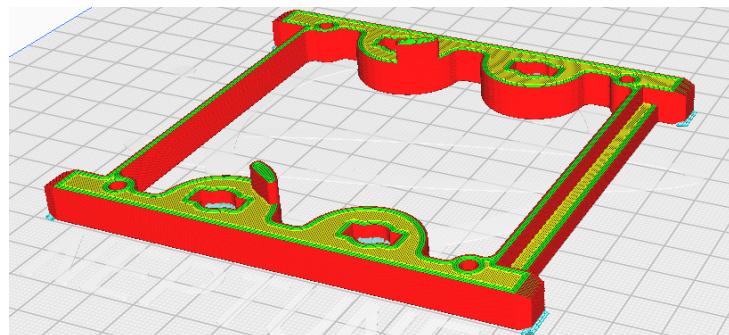


Figure 236: Final *ZY_wall* - 3D printed | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: Print the *ZY_wall*, fig 236 in a equal direction as the figure and use supports (Cura setting; support structure normal - touching build plate).

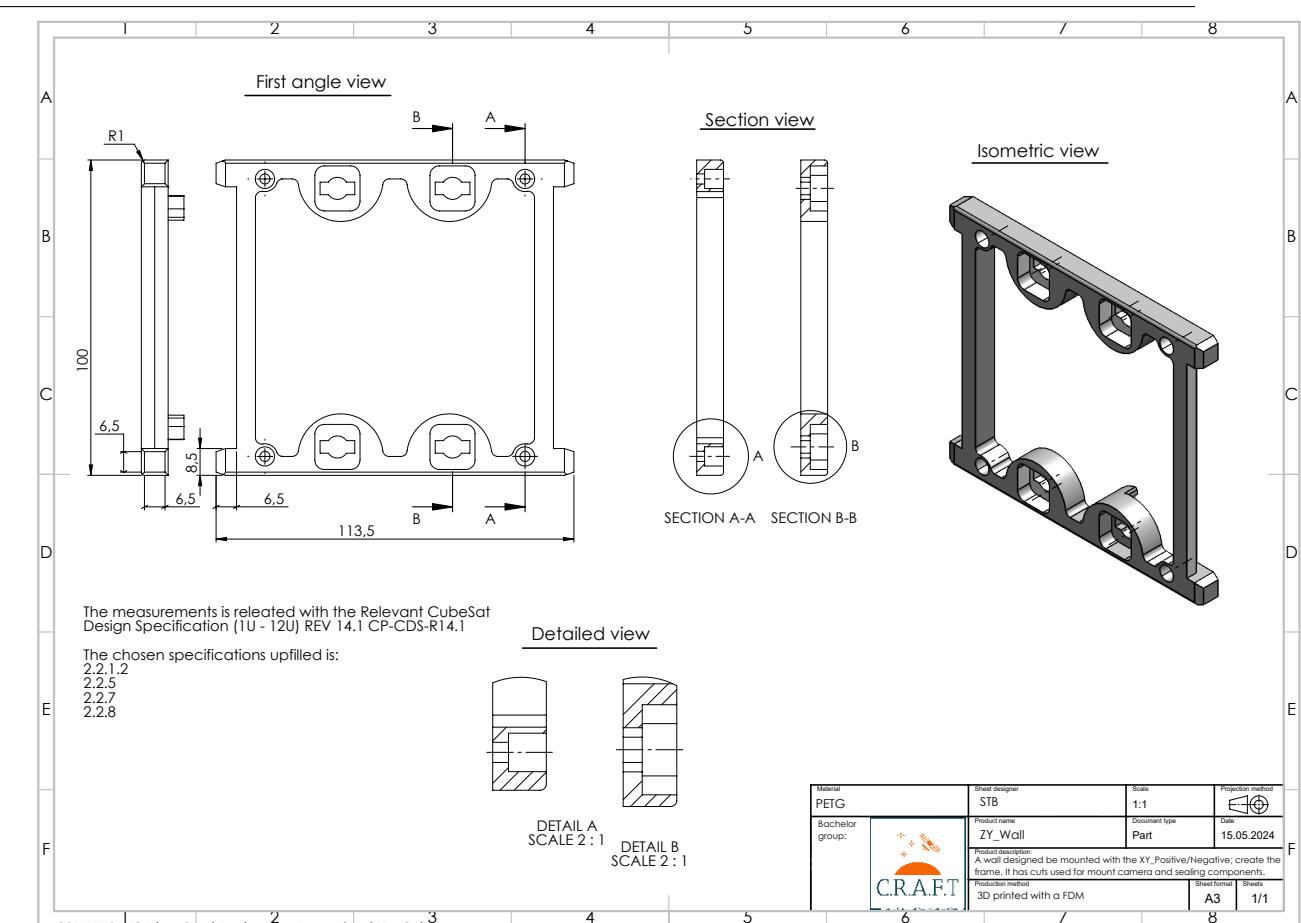


Figure 237: ZY_Wall - sheet | STB

Part name: SpringHold_Part2

The first version of the *SpringHold_Part2*, fig 238 started with a realistic control which printing direction is most appropriate for the component. It was also used to test if a version of the *InternalStampPin* with the major diameter Ø3 mm and M3x0.5 threads would fit with the first version of the *SpringHold_Part2*. It ended with a failure due to the small threads when it printed the *SpringHold_Part2*.

Note: the component does not have fillets on the edge where the body changes from a cylinder to a hinge because the fillet created an not possible print because of it's low radius value. It was tested with the first version.

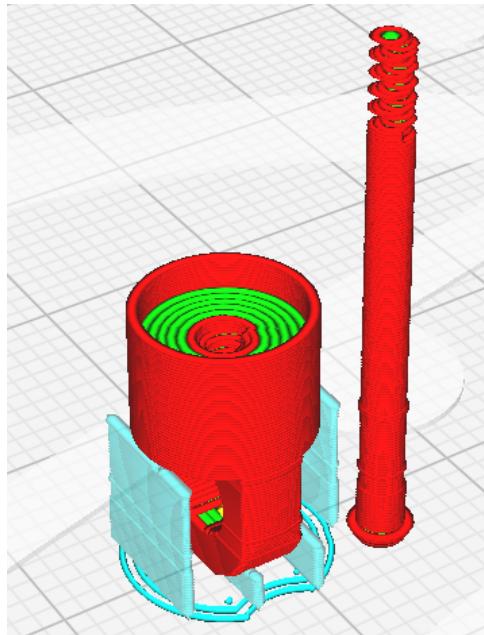


Figure 238: Left; first version of *SpringHold_Part2* - Right; first version of *InternalStampPin* | STB

Since the first version ended with a non-usuable part. A higher metric thread value and diameter for the *SpringHold_Part2* had to be used. The second version, fig 239 shows a the new version with a larger major diameter and thread value M5x0.8. The printing was accepted and gave us a conclusion of the surface where supports is placed. The surface is under the threads (equal the first version figure). The version had risks with failure in printing because of the small area on the printing bed which was not supported but we accepted the risk because of we didn't want to interact with supports in the area where the threads is placed.

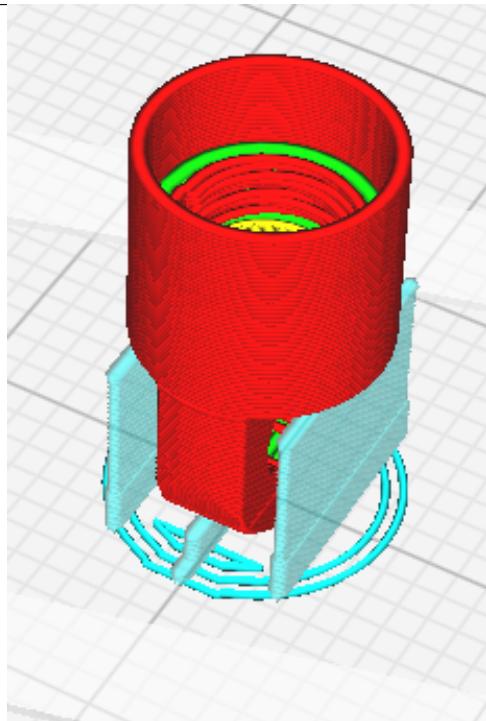


Figure 239: Second version - *SpringHold_Part2*

The third (& final) version, fig 240 represents a change in the diameter, height and thickness where the *Pin_Sealing* is placed due to changes in the *Pin_Sealing* production.

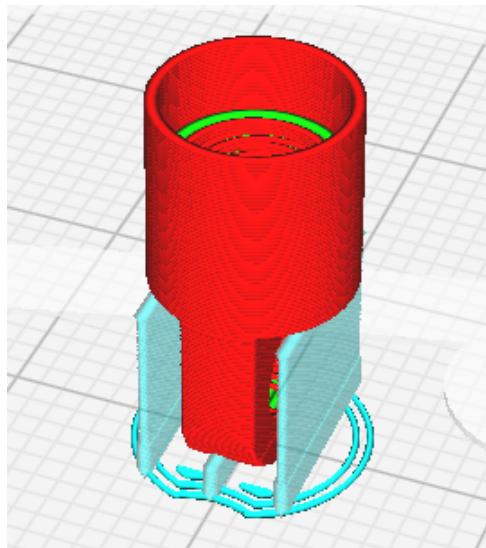


Figure 240: Third version - *SpringHold_Part2*

Print recommendation of the used component in the C.R.A.F.T-Sat: Print the *SpringHold_Part2* with the surface on the print bed equal fig 240 and use support setting; normal support on the surfaces between the bed and the overhangs.

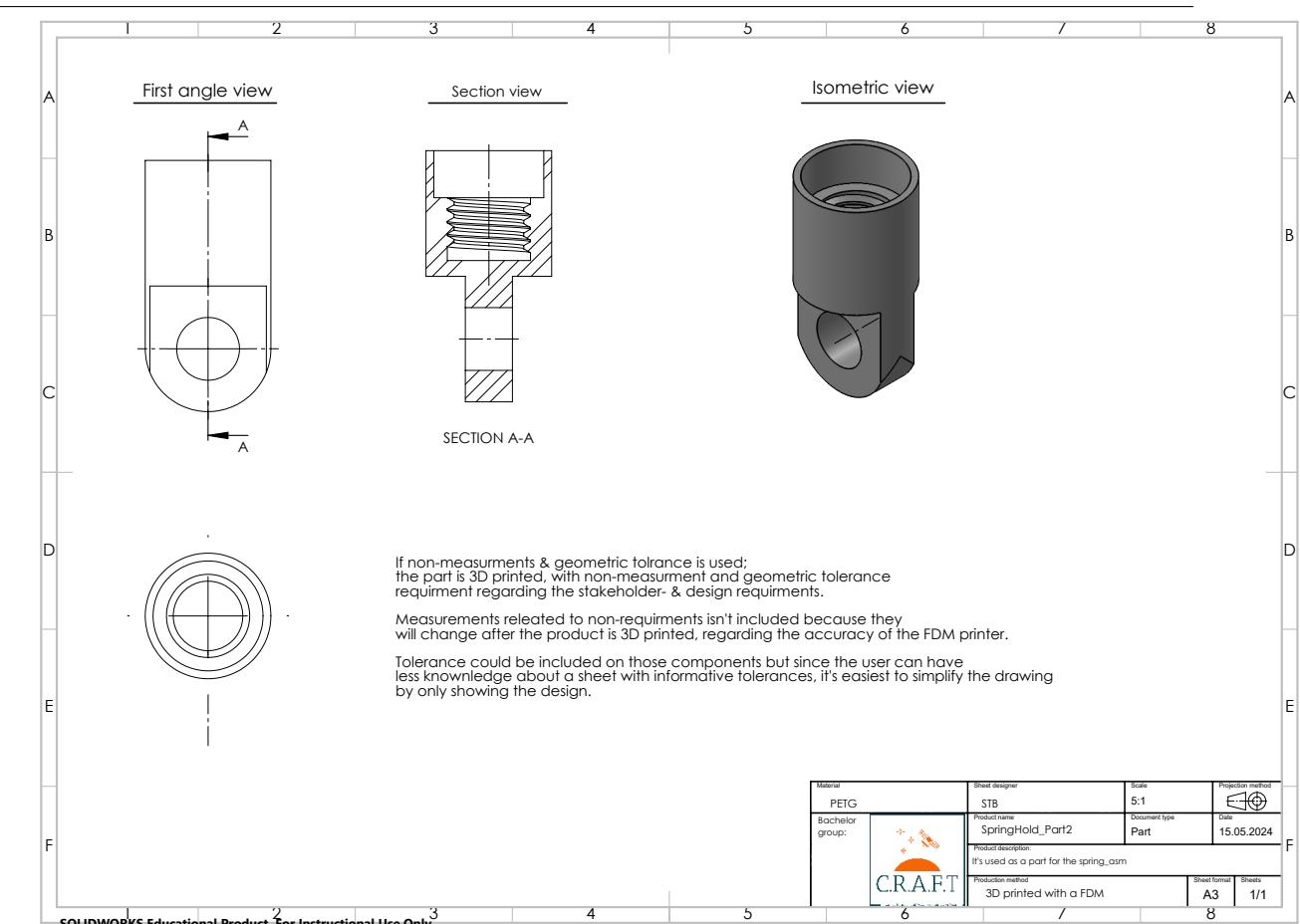


Figure 241: *SpringHold_Part2* - sheet | STB

Part name: SpringHold_Part1

The first version of the *SpringHold_Part1*, fig 242 started with a test of how many layers is necessary for threads, which surface on the part should be on the bed, and if a threaded hole with M3x0.5 thread resulted. We printed without supports and result ended with a failure with too few layers and the major diameter was too small. It was also considered to split the printing, but it was assumed to be not sufficient due to the sizes resulted after it was printed.

Note: the largest inside diameter on the *SpringHold_Part1* & *SpringHold_Part2* is determined and never changed since it's chosen for the springs used.

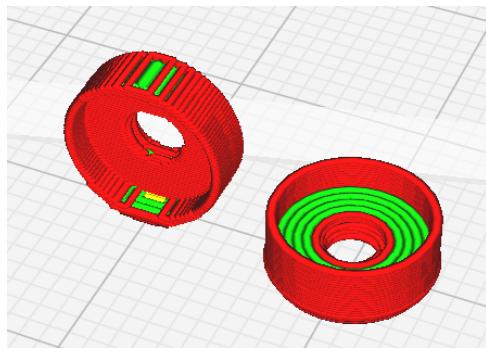


Figure 242: First version - *SpringHold_Part1*

The second (& final) version, fig 243 had changes in the major diameter from M3x0.5 to M5x0.8, more of layers with threads and without threads. We decided (without testing) that printing the part flat, without supports, and the pitch diameter normal to the printing bed gave the threads and part we needed.

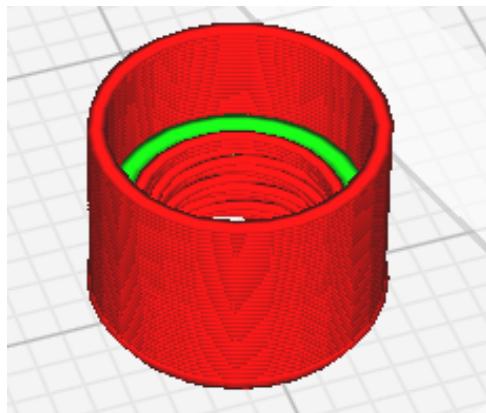


Figure 243: Second version - *SpringHold_Part1*

Print recommendation of the used component in the C.R.A.F.T-Sat: print the Second version of the *SpringHold_Part1*, fig 243 with non-supports and place the part in a equal direction as in the figure.

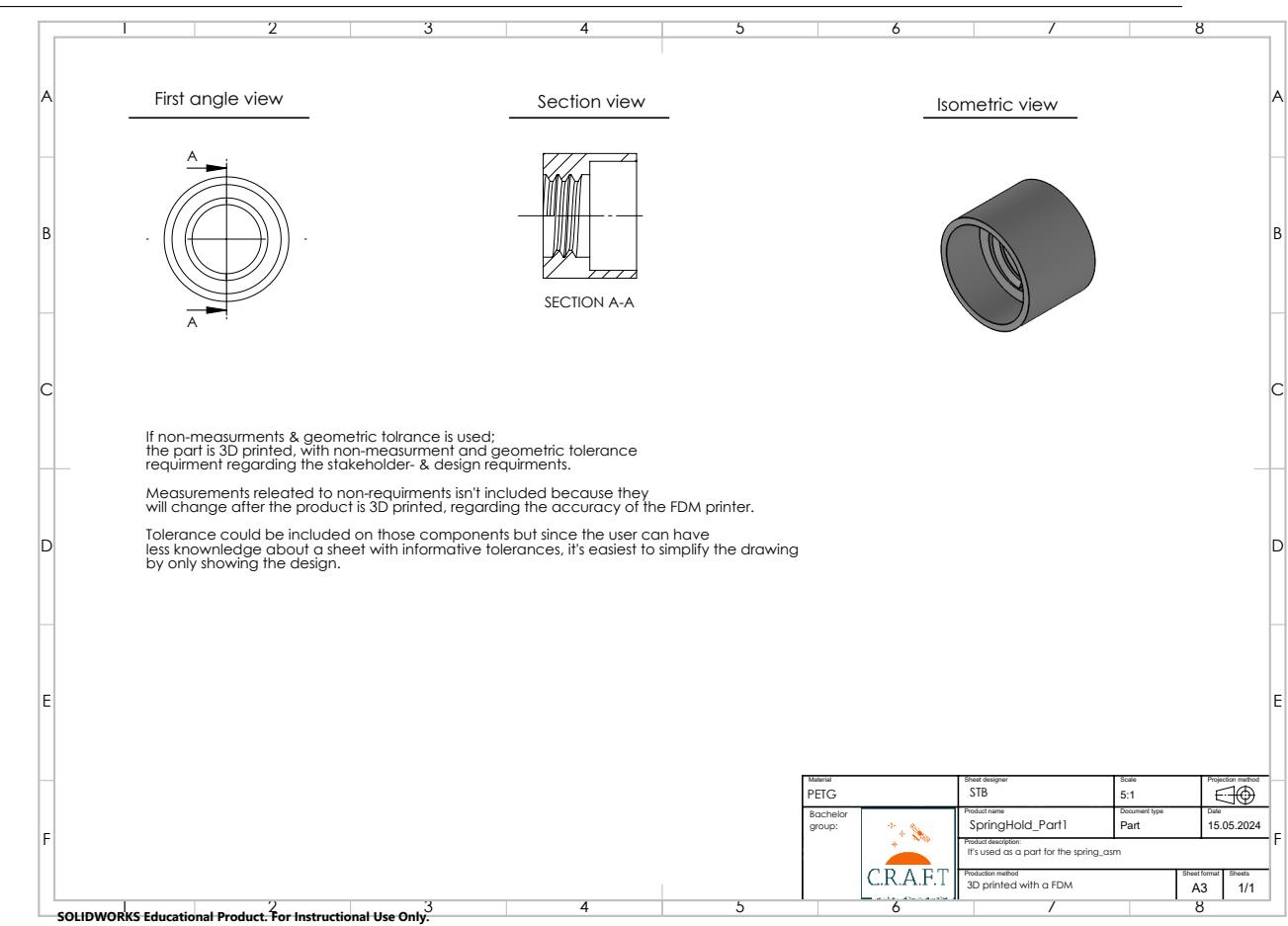


Figure 244: *SpringHold_part1* - sheet | STB

Part name: ShockPart

The first version of the *ShockPart*, fig 245 is a test of how many layers is necessary before the hole used for a *Pin_Sealing*, the printing direction due to threads, and internal diameter used for a *InternalStampPin* and where supports is necessary to be placed. Since it was determine from the last version of the *SpringHold_Part1* that thread M5x0.8 was appropriate to use, we did not test the thread size on the *ShockPart*. Also, the diameter used in the area where the *InternalStampPin* diameter is placed was not changed, and was defined after test printing of the *InternalStampPin* to define a realistic diameter. The print of the first version of the *ShockPart* resulted in failure due to non-supports inside the *Pin_Sealing* hole.

Note: the amount of layers was equal 1 mm before internal face of the hole used for the *Pin_Sealing*.

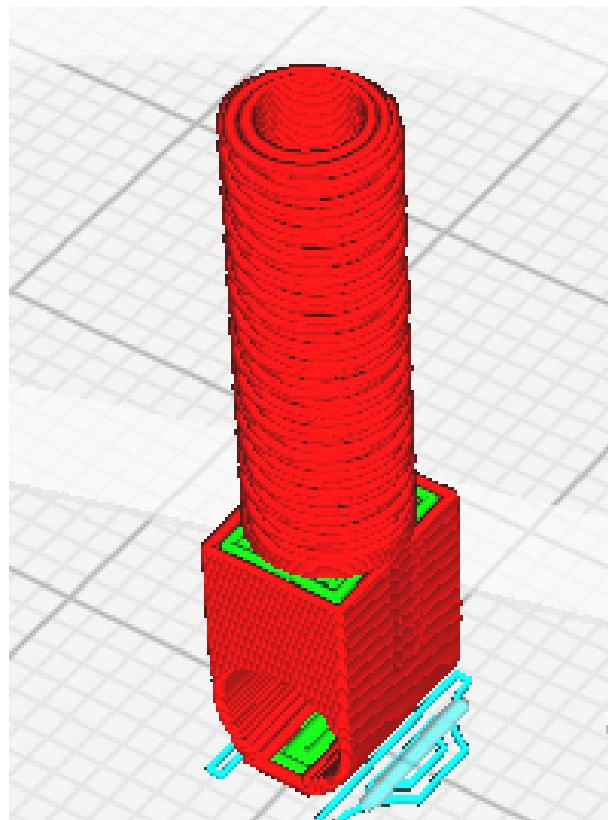


Figure 245: First version - *ShockPart*

Since it required supports inside the hole designed for *Pin_Sealing*, the second version of the *ShockPart*, fig 246 is test of the result using supports inside the hole and the method of removing the supports.

Note: tool; side cutters is used for removing the supports inside the *Pin_Sealing* hole on the *ShockPart*.

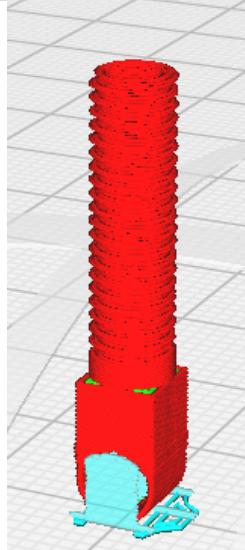


Figure 246: Second version *ShockPart*

The diameter on the *Pin_Sealing* was changed during 3D printing, therefore, the third (& final) version, fig 247 had only changes in the diameter used for the *Pin_Sealing*.

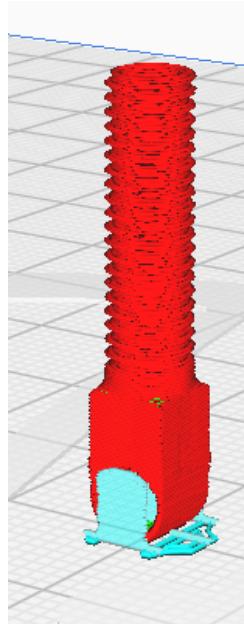


Figure 247: Third version - *ShockPart*

Print recommendation of the used component in the C.R.A.F.T-Sat: print the third version of the *ShockPart*, fig 247 with the threads placed in the positive Z-axis, equal figure. Use supports (Cura setting; support structure normal - everywhere) but be secure that the threads does not have any supports. Use support blocker to prevent supports on the threads.

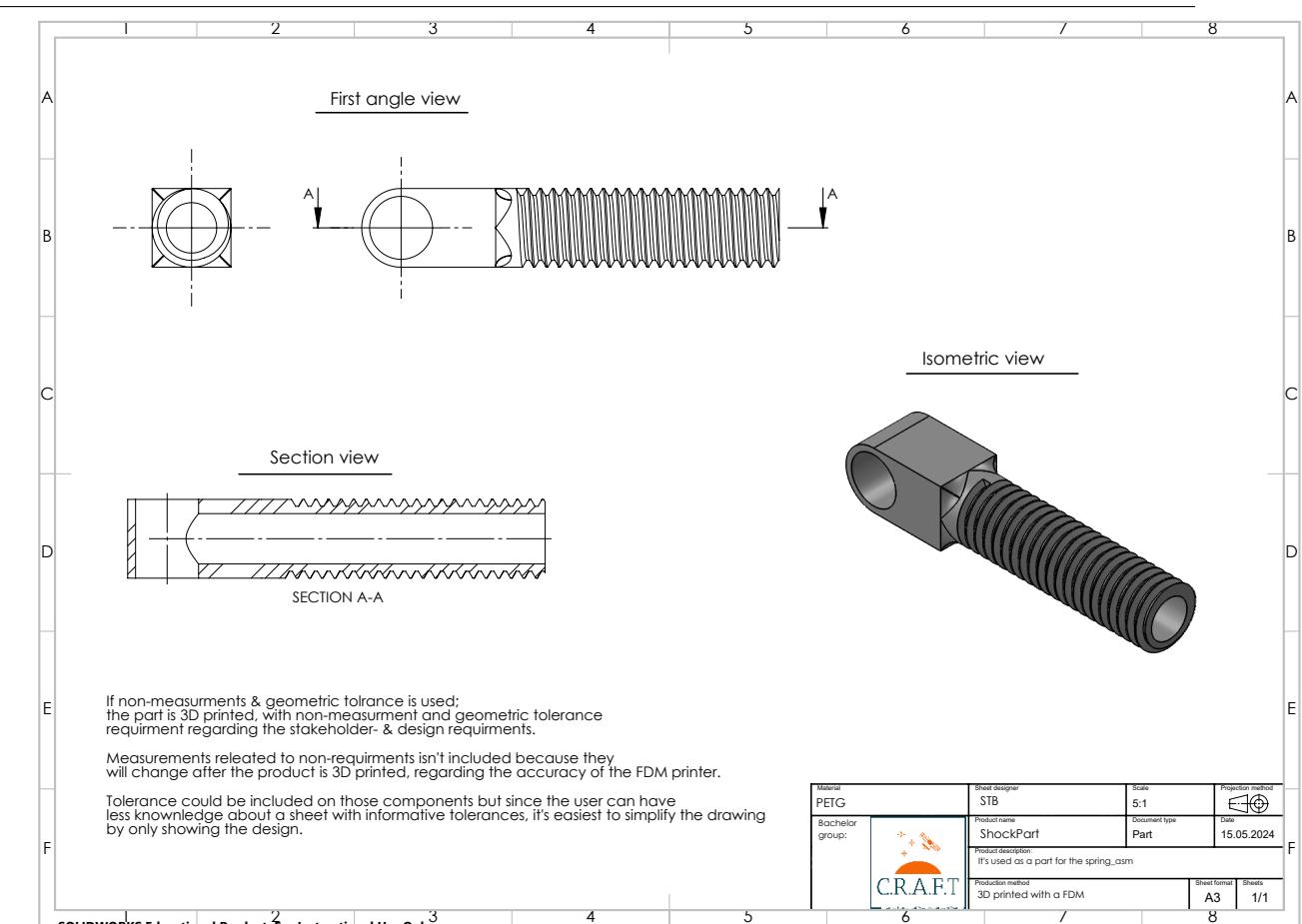


Figure 248: *ShockPart* - sheet | STB

Part name: InternalStampPin

The first version of the *InternalStampPin*, fig 238 was designed as a shaft with M3x0.5 mm threads and a possible solution to secure the shaft will not fall out. Due to the thread diameter and failure with the printing, a second version was designed

Note: the design of not falling out on the shaft was thrown and a solution with longer shaft with the solution of none disassembling during use was considered and designed instead.

The second version of the *InternalStampPin*, fig 249 had changes in the major diameter of the area with threads from M3x0.5 to M5x0.8, and a smaller increasing in the diameter without threads. The result was a good print but difficult to mount on the other threaded components due to the first print layer on the bed is equal the area where the threads is starting.

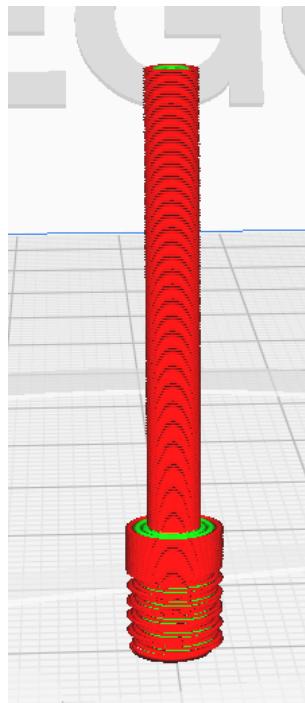


Figure 249: Second version - *InternalStampPin* | STB

To solve the printing problem, the third (& final) version of the *InternalStampPin*, fig 250 was made. It uses normal supports from the printing bed on the lower surface on the larger cylindrical area where we have threads.

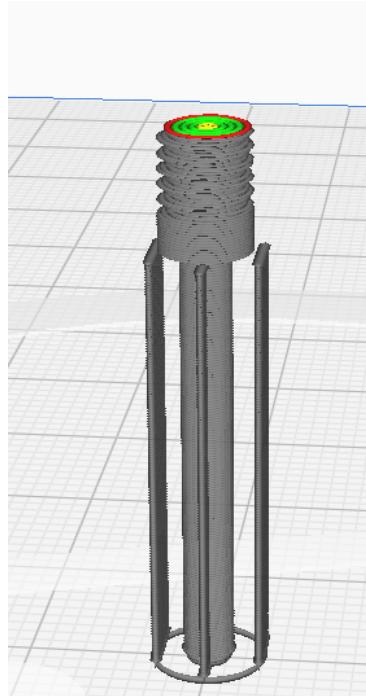


Figure 250: Third version - *InternalStampPin* | STB

Print recommendation of the used component of in the C.R.A.F.T-Sat: print the third version of *InternalStampPin*, fig 250 with the small cylindrical surface on the print bed and use supports (Cura setting; support structure normal - touching build plate) on the large cylindrical - threaded region between the first layer of the larger diameter and the print bed.

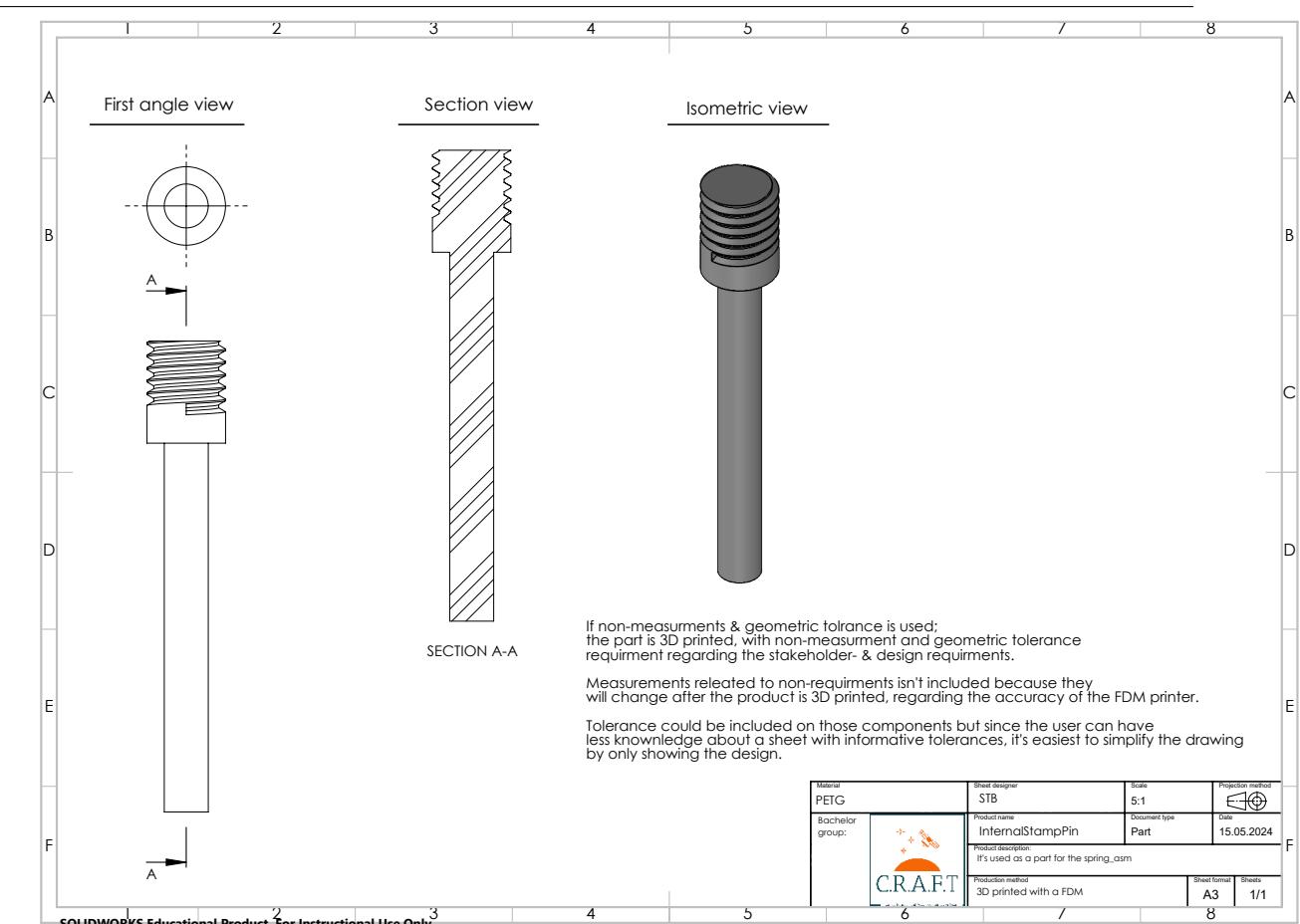


Figure 251: *InternalStampPin* - sheet | STB

Part name: Lock1

The first version of the *Lock1*, fig 252 was designed as small cylindrical body with a cut used to check if the final size of the *HingeLockPart* fits. The next state was the re-drawing of the outside body of *Lock1* to fit inside the *ZY_wall* cut.

Note: sizes used for the *HingeLockPart* was considered before printing. Therefore, it fits with the first print.

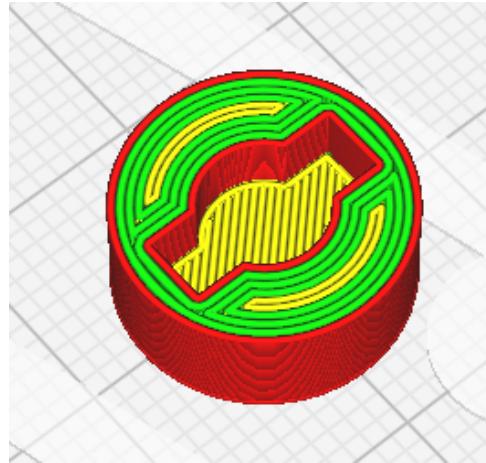


Figure 252: First version - *Lock1* | STB

The second version, fig 253 has a rectangular cut instead of a circular. This shape was made to preventen the *Lock1* to rotate inside the area of *ZY_wall* cut. Also, a rectangular hook cut on the version was made. The cut on the lower surface is made to make it easier to pull out the *Lock1* from *ZY_wall* using a hook. Since the hook area was small, a new version was made.

Note: the outside sizes on the rectangular body was considered during CAD to prevent unnecessary printing time and measurement checking.

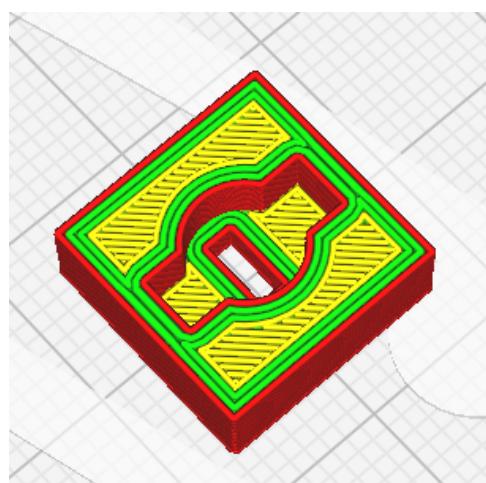


Figure 253: Second version - *Lock1* | STB

The third version, fig 254 has a design where the rectangular cut for a hook is changed to cut removing material from circularar edge to edge. This gave the opportuniti for a "larger" tool used as a hook. Also, the hook area was made thicker. This resulted in a print where the risk is lower in destroying the hook area during dragging.

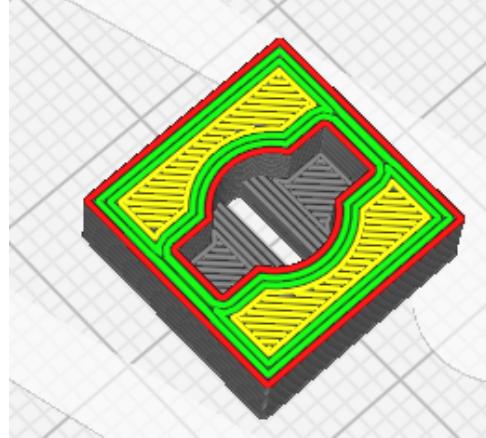


Figure 254: Third version - *Lock1* | STB

It was a printing problem on the *ZY_wall* part (the sixt *ZY_wall* cut design). Therefore, the fourth (& final) version, fig 255 was regarding fillets on the edges.

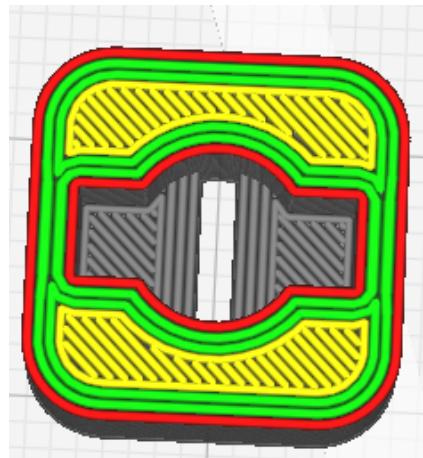


Figure 255: Fourth version - *Lock1* | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: print the fourth version of *Lock1*, fig 255 with the surface containing the hook cut on the printing bed to prevent use of supports.

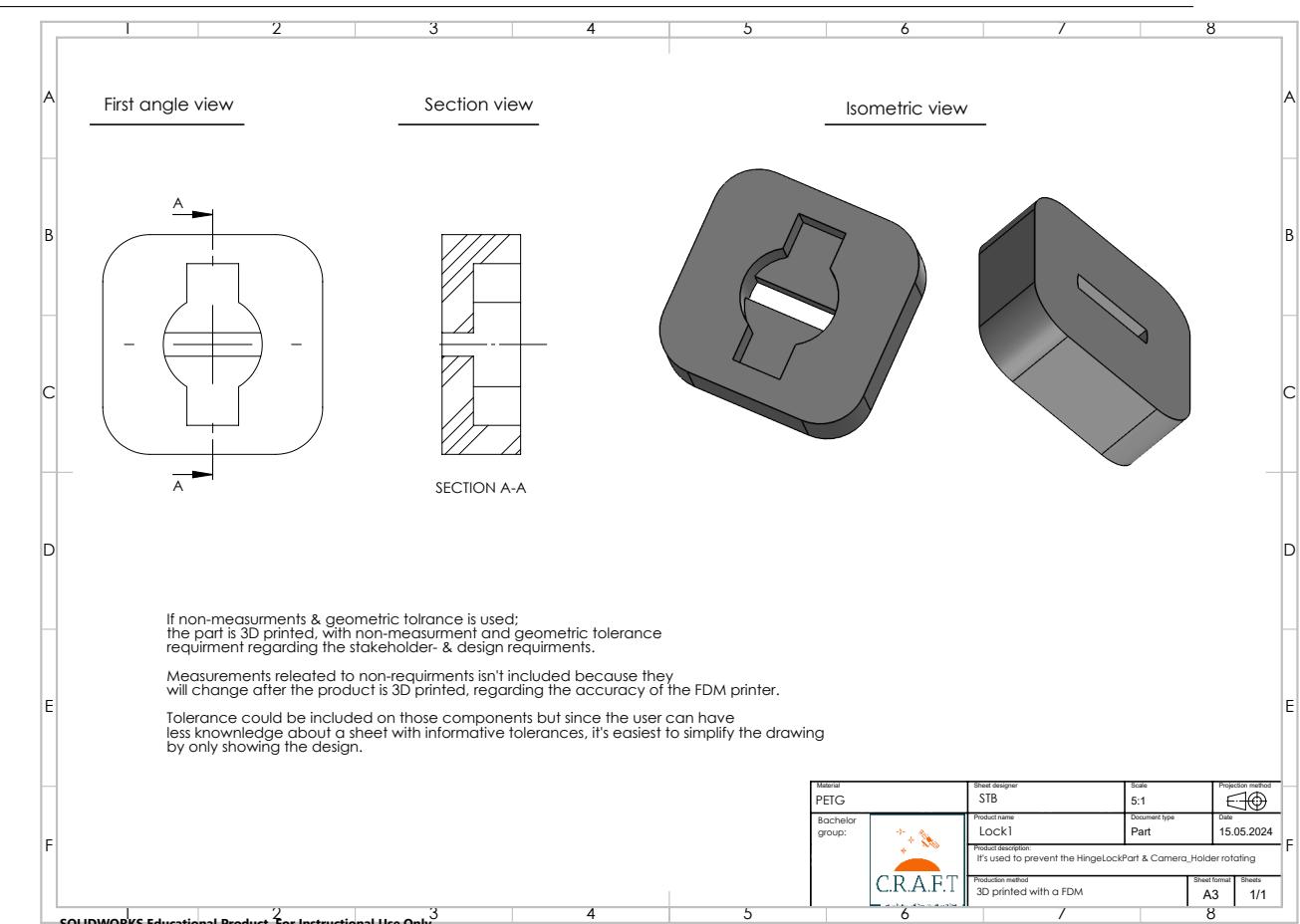


Figure 256: *Lock1* - sheet | STB

Part name: HingeLockPart

The first version of the *HingeLockPart*, fig 257 was designed with the thought of keeping everything small due to the total mass of the C.R.A.F.T-Sat. But, when it was placed in the slicer program, it shows that the key was not equal the CAD. Therefore, we didn't print the first version.

Note: we will in this section use the name key for the area on the body designed to fit with a *Lock1* part.

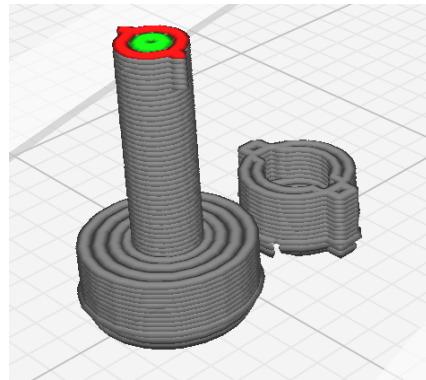


Figure 257: First version - *HingeLockPart* | STB

The second version of the *HingeLockPart*, fig 258 increased the diameter of the longest cylinder and increased the size of the key. This also showed an unrealistic print with a non-possibility to get an key usable in our product.

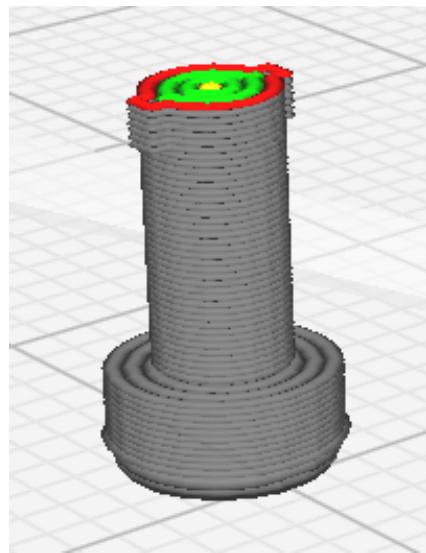


Figure 258: Second version - *HingeLockPart* | STB

The third version of the *HingeLockPart*, ref 259 increased the length of the key, to a length where it's more usable for the C.R.A.F.T-Sat. It didn't have any supports and was therefore not printed, but was tested with the fourth and fifth version. The thickness of the cut for key on *Lock1* was changed, therefore the height of the *HingeLockPart* had to be changed.

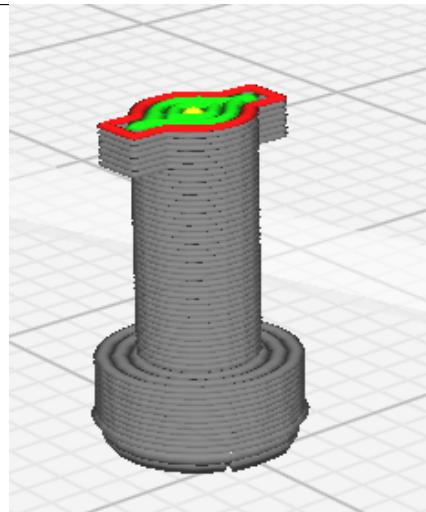


Figure 259: Third version - *HingeLockPart* | STB

The sixth (& final) version of the *HingeLockPart*, fig 260 represents the height reduction and the component used. We decided that supports was necessary to use because of a possible risk in a angled/rough surface. The reason is that it's planned to be place the lowest surface of the key on the *ZY_wall* and be hindered in rotation-movement with the *Lock1*.

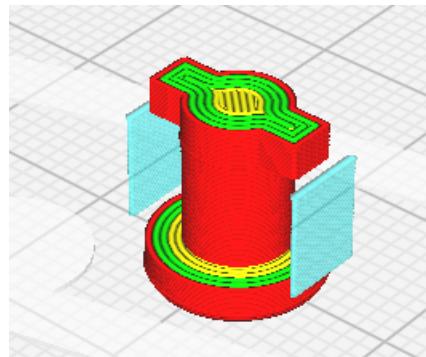


Figure 260: Sixth version - *HingeLockPart* | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: print the sixth version of *Pin_Sealing*, fig 260 with supports (Cura setting; support structure normal - everywhere) and place the cylinder with the larges surface area on the print bed (equal fig 260).

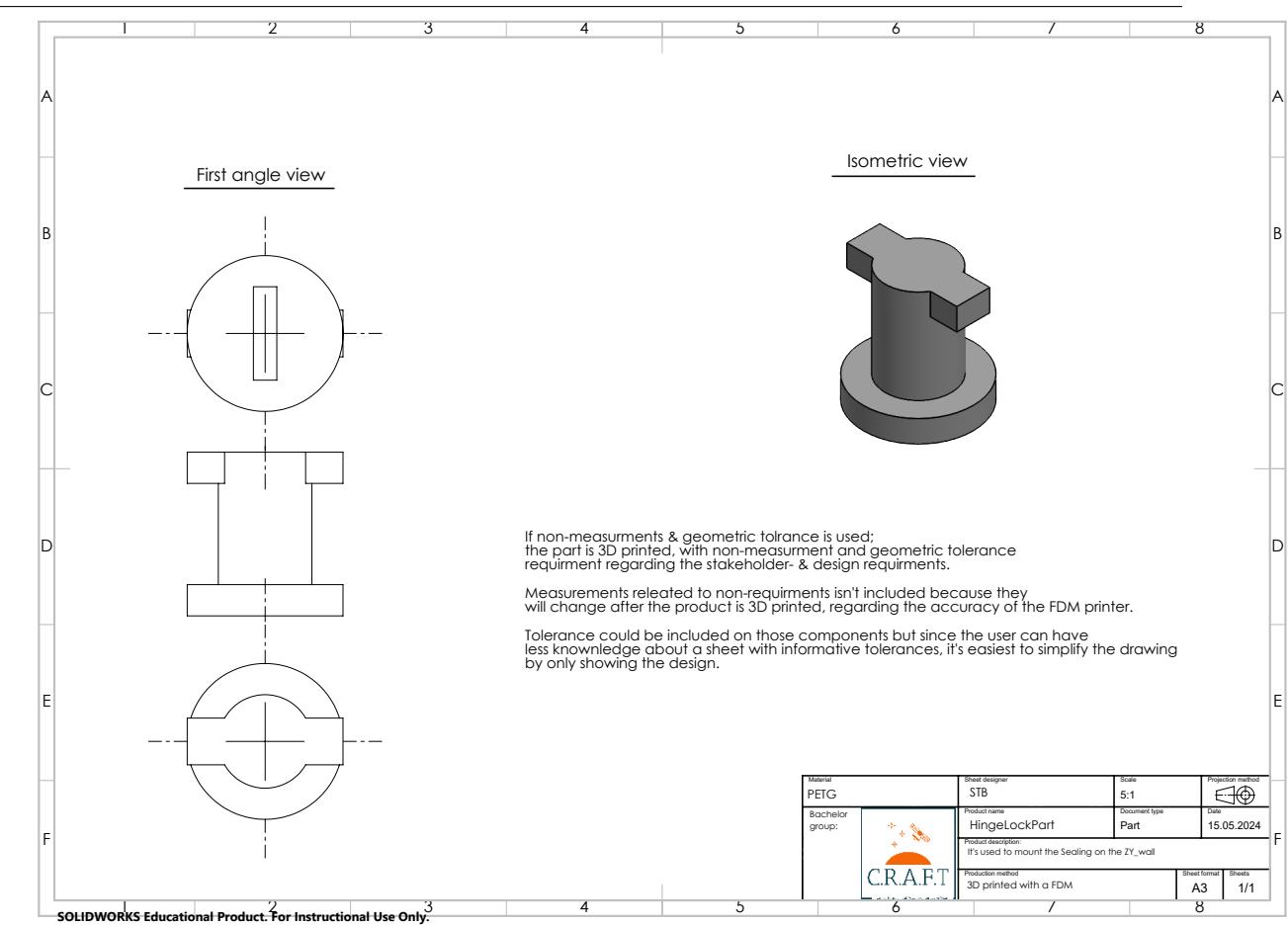


Figure 261: *HingeLockPart* - sheet | STB

Part name: SealingAntenna

Since the *SealingAntenna* is a large component, we made a mass cut to get the possibility to only print a region where we have attach the fishing rope and *HingeLockPart*. The first version of the *SealingAntenna* cut, fig 262 was used to check how the component should be printed with supports. It was some problems in support removal in the area where the fishing rope is attached. We also noticed the possibilities off fracture due to the small thickness and surface area where the *Pin_Sealing* is mounted.

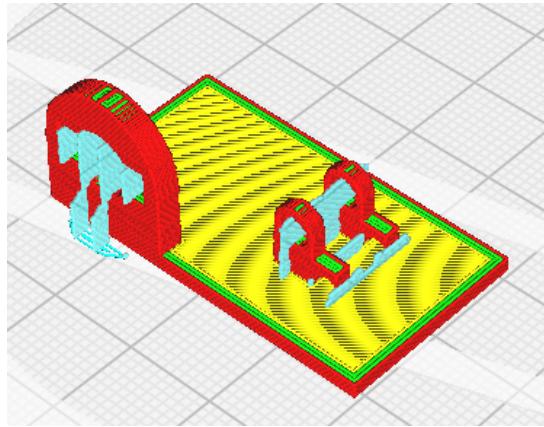


Figure 262: First version - *SealingAntenna* cut | STB

The second version of the *SealingAntenna* cut, fig 263 represents a change of the diameter of the holes where the *Surface_Pin* is attached. The new diameter made it easier mount the new *Pin_Sealing*. Since it was a small opening between the surface of the sealing wall and of the mounting placement for the fishing rope, it resulted with considerations if the area where the fishing rope is attached could be larger and have no openings beside a half circle cut straight through the walls.

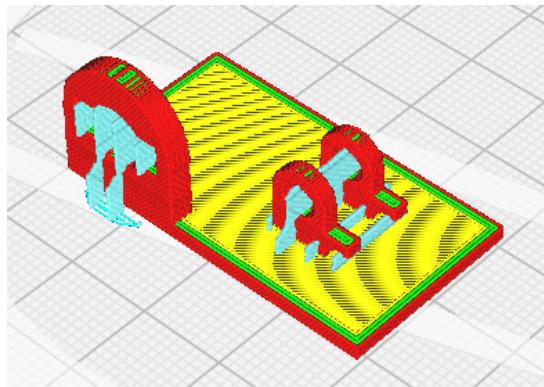


Figure 263: Second version - *SealingAntenna* cut | STB

The first (& finish) version of the *SealingAntenna*, fig 264 represents a change of the fishing rope design and change, the areas where supports is placed. It's important to understand that areas without supports in this design didn't need support, and that the support could give a consequence of a bad printed component due to the small dimensions.

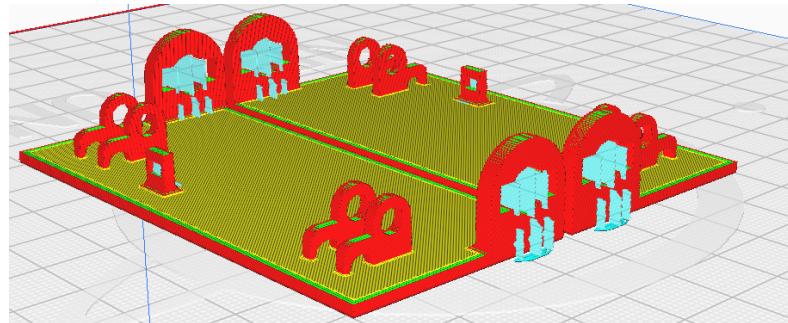


Figure 264: *SealingAntenna* | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: print the *SealingAntenna*, fig 264 flat with the all the mounting parts faced in the positive Z-axis. Only use supports (Cura setting; support structure normal - everywhere, and place support blocker on the other overhang faces) on the holes made for the *HingeLockPart*.

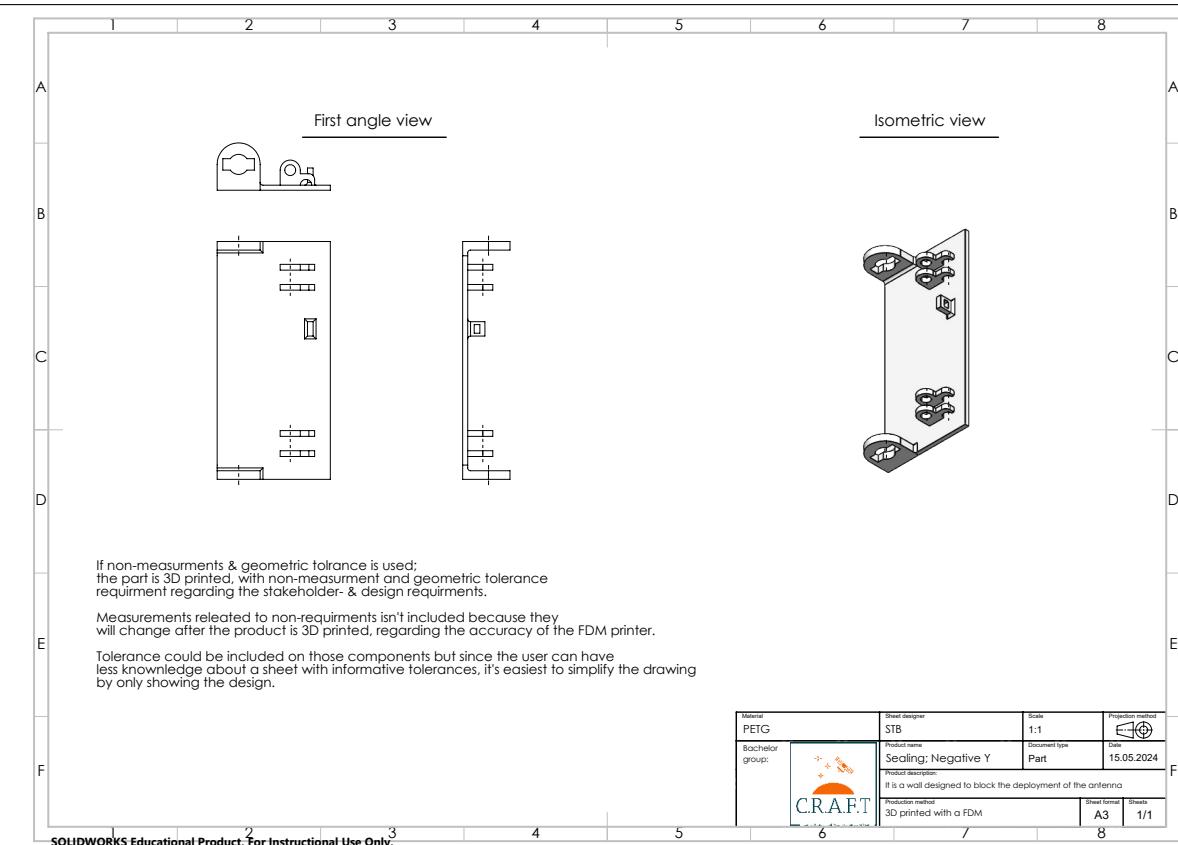


Figure 265: *Sealing_Negative Y* - sheet | STB

Part name: Antenna_ReleaseBoard

The first (& final) version of the *Antenna_Releasebord*, fig 267 doesn't need any supports during printing. It had changes during the design related to the attachment of the antennas and *SpringHold_Part2* but we have non-figures of the work. It's important to print this component flat on the equal surface as in the figure.

Print recommendation of the used component in the C.R.A.F.T-Sat: print the *Antenna_ReleaseBoard*, fig 267 flat with the all the mounting parts faced in the positive Z-axis. Don't use supports.

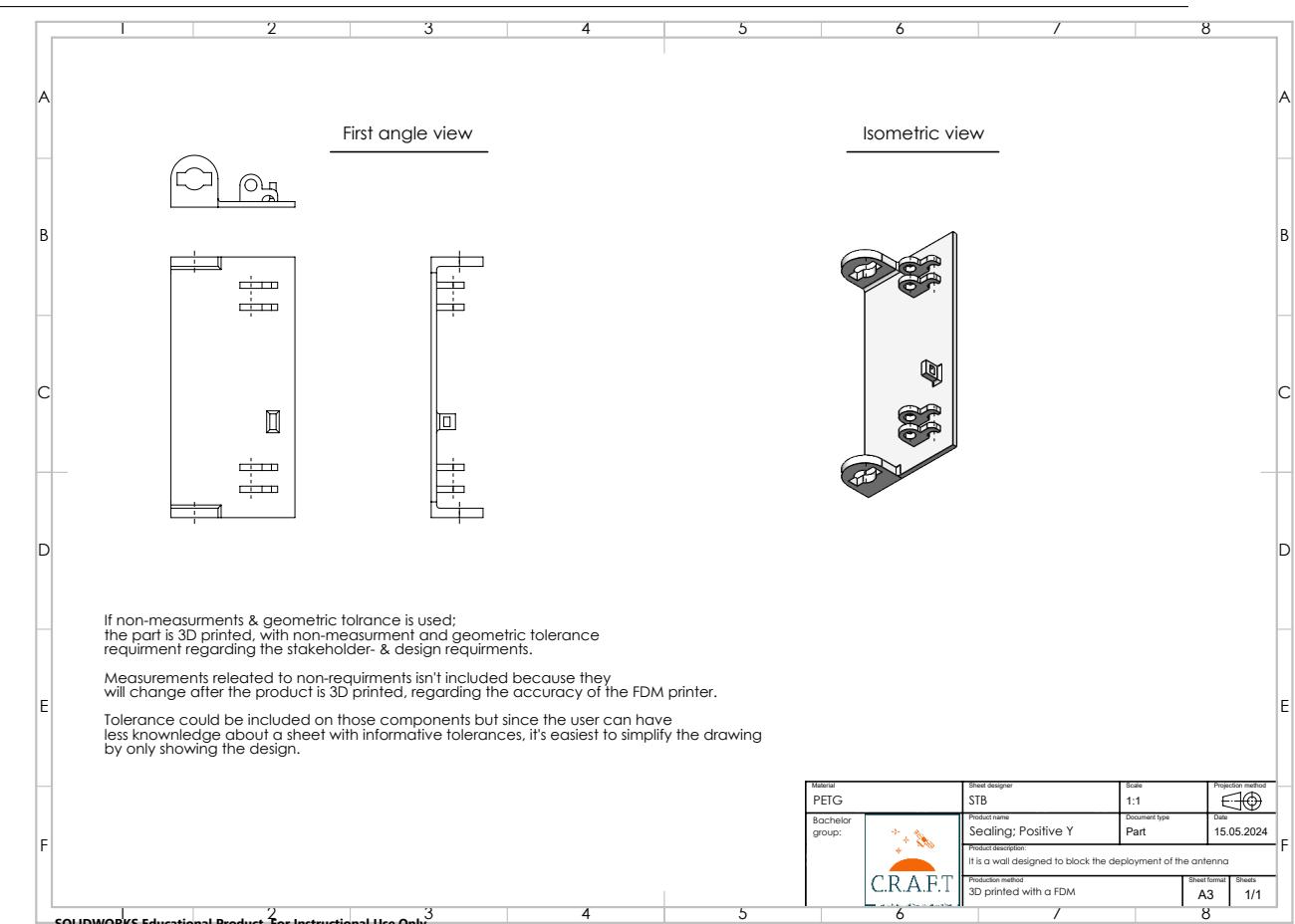


Figure 266: *Sealing_Positive Y* - sheet | STB

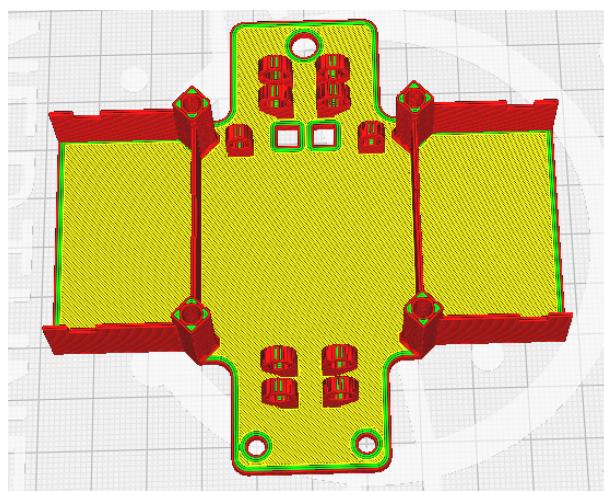


Figure 267: *Antenna_ReleaseBoard* | STB

Part name: XY_wall_Negative_Z

When the *XY_wall_Negative_Z* was designed, it was some difficulties around the amount of layers (thickness) of the area with between the top surface and the surface where the

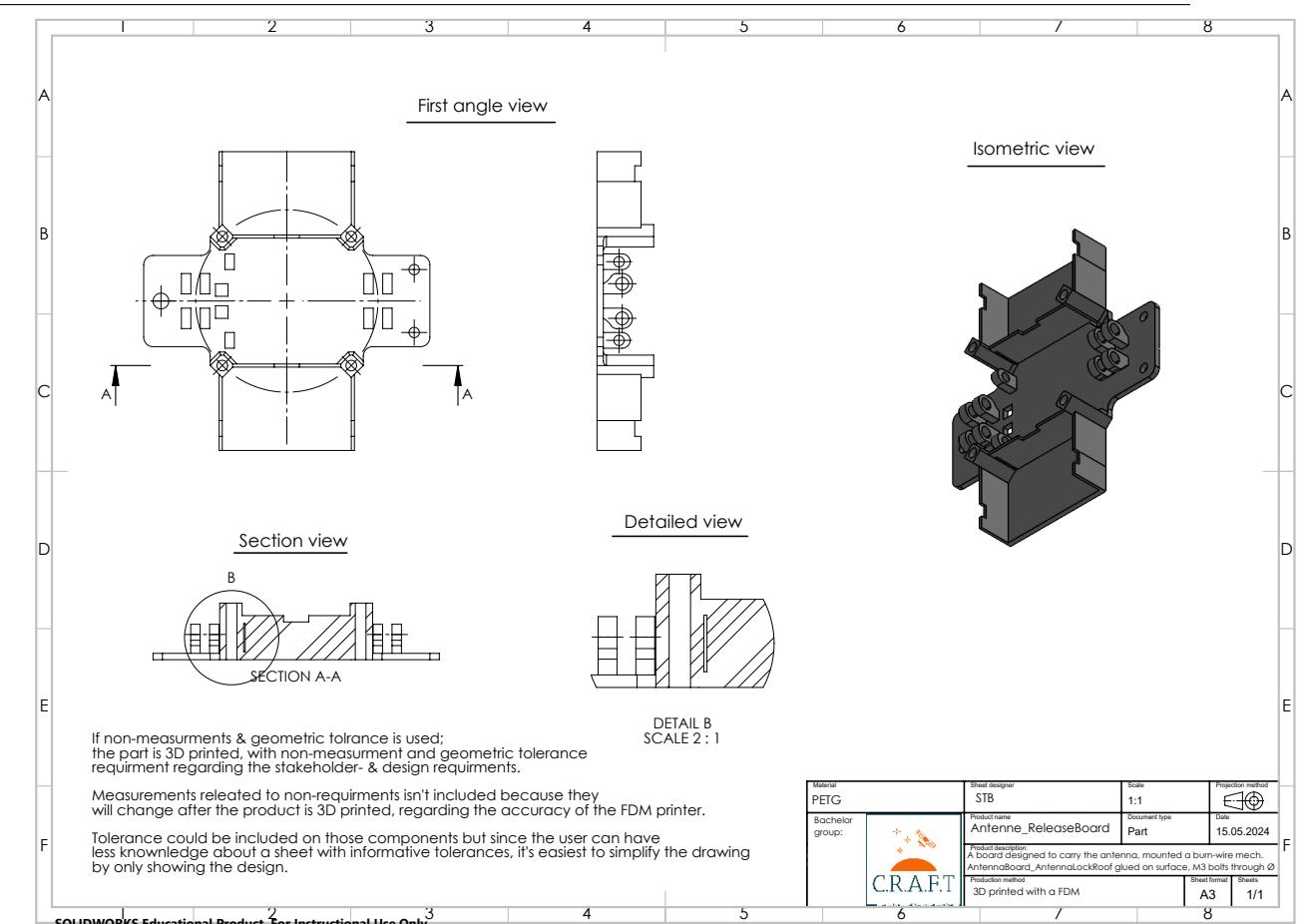


Figure 268: *Antenne_ReleaseBoard* - sheet | STB

internal diameter changes from large to small on the mounting area for a M3 bolt. Therefore, the *XY_wall_Negative_Z* was cut to test the dimensions. The first *XY_wall_Negative_Z* cut version, fig 269 ended as the solution of the the dimensions where the bolt is mounted.

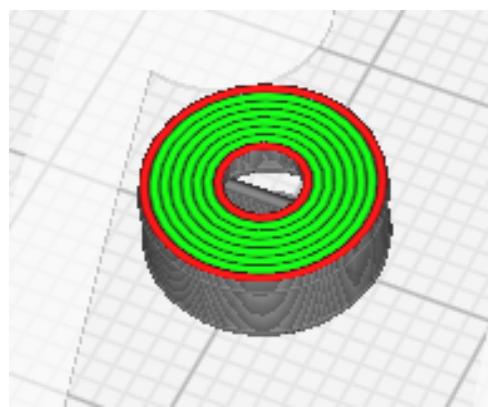


Figure 269: *XY_wall_Negative_Z* cut | STB

The first *XY_wall_Negative_Z* version, fig 270 is the solution of the the dimensions and

how to print the wall correctly.

Note: only use supports on the areas on the printing bed with overhang parallel with the printing bed.

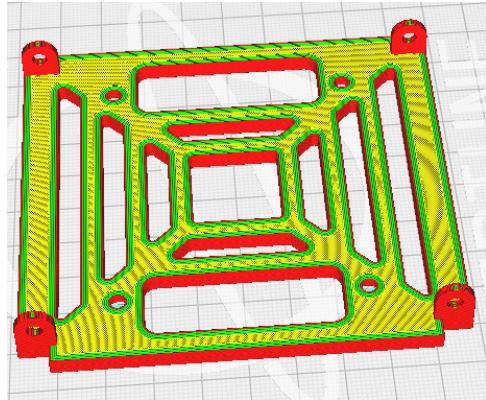


Figure 270: *XY_wall_Negative_Z* | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: print the *XY_wall_Negative_Z*, fig 270 flat with the mounting for *ZY_wall* faced in the positive Z-axis. Only use supports (Cura setting; support structure normal - touching build plate).

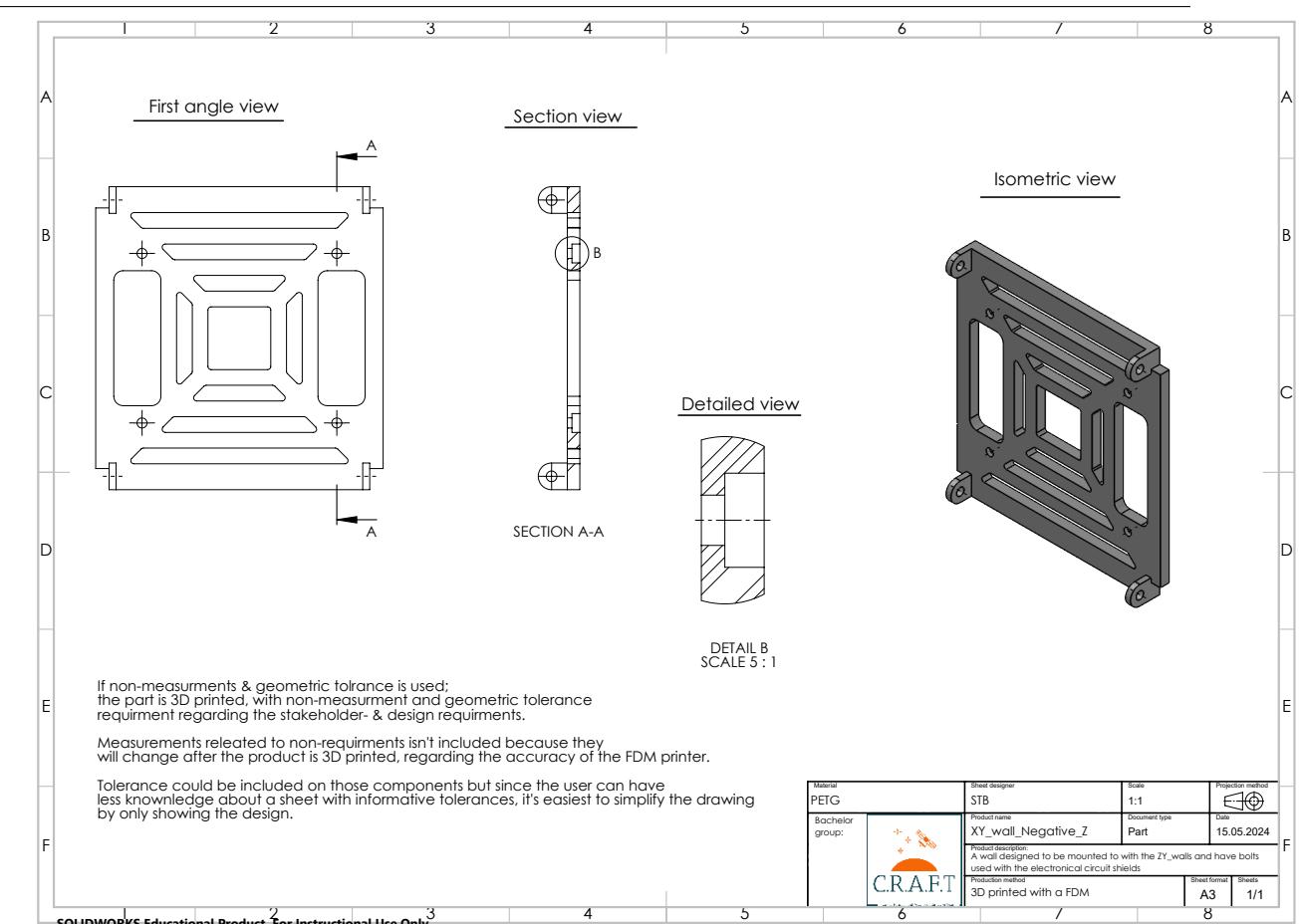


Figure 271: *XY_wall_Negative_Z* - sheet | STB

Part name: XY_wall_Positive_Z

The sizes regarding the circular cut parallel with the printing board used for mounting with screws has the same dimensions as the equal cut in *XY_wall_Negative_Z*. Print this part with the same supports as the settings used for *XY_wall_Negative_Z*.

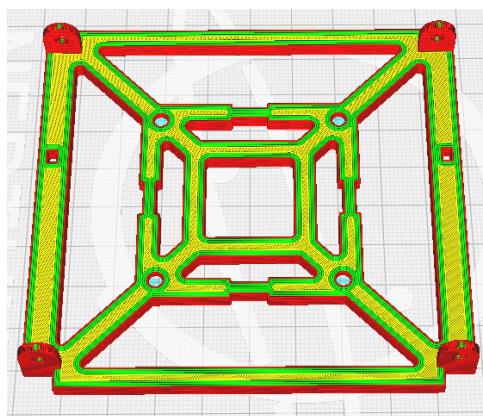


Figure 272: *XY_wall_Positive_Z* | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: print the *XY_wall_Positive_Z*, fig 272 flat with the mounting for *ZY_wall* faced in the positive Z-axis. Only use supports (Cura setting; support structure normal - touching build plate).

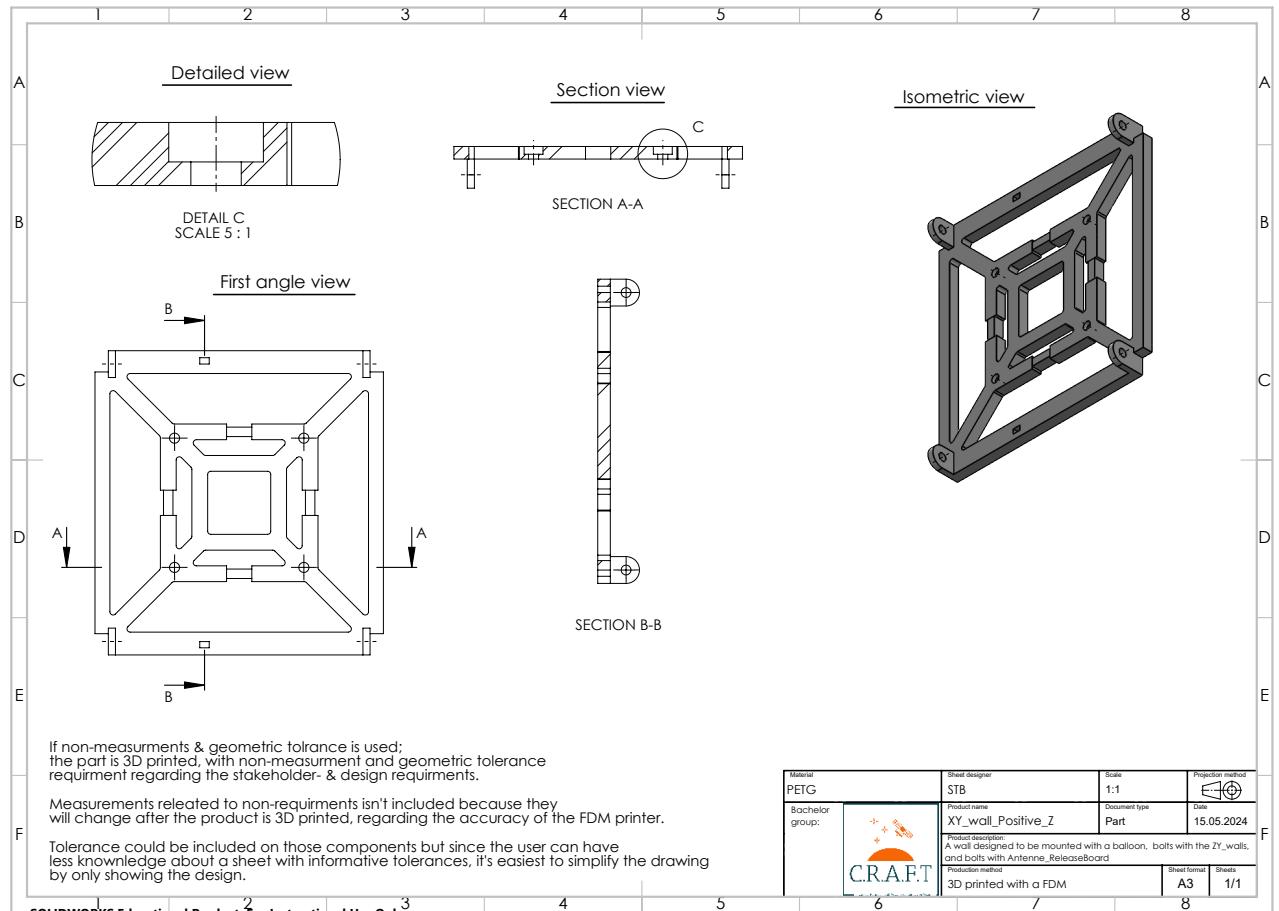


Figure 273: *XY_wall_Positive_Z* - sheet | STB

Part name: BatterySpringPlate

The *BatterySpringPlate*, fig 274 is designed to fit a specific battery with tape folding made of temperature resistance material. The holes on the side is designed to place M3 bolts used as cylinders through the component.

Print recommendation of the used component in the C.R.A.F.T-Sat: print the *BatterySpringPlate*, fig 274 with non-supports placed flat on face with the largest surface area placed on the printing bed.

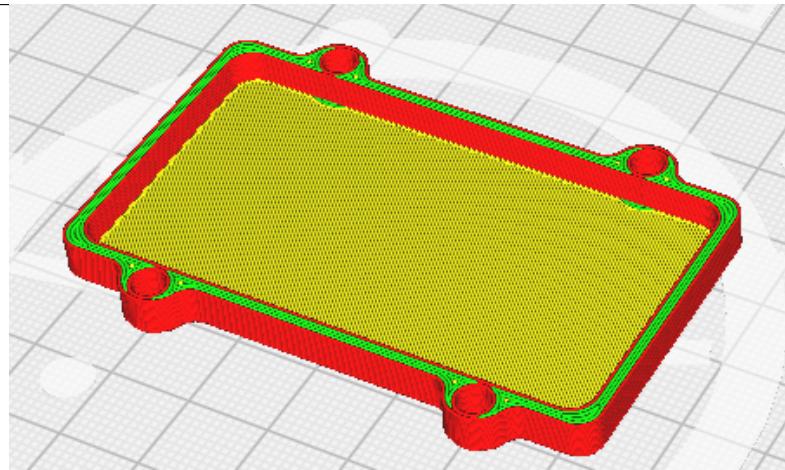


Figure 274: *BatterySpringPlate* | STB

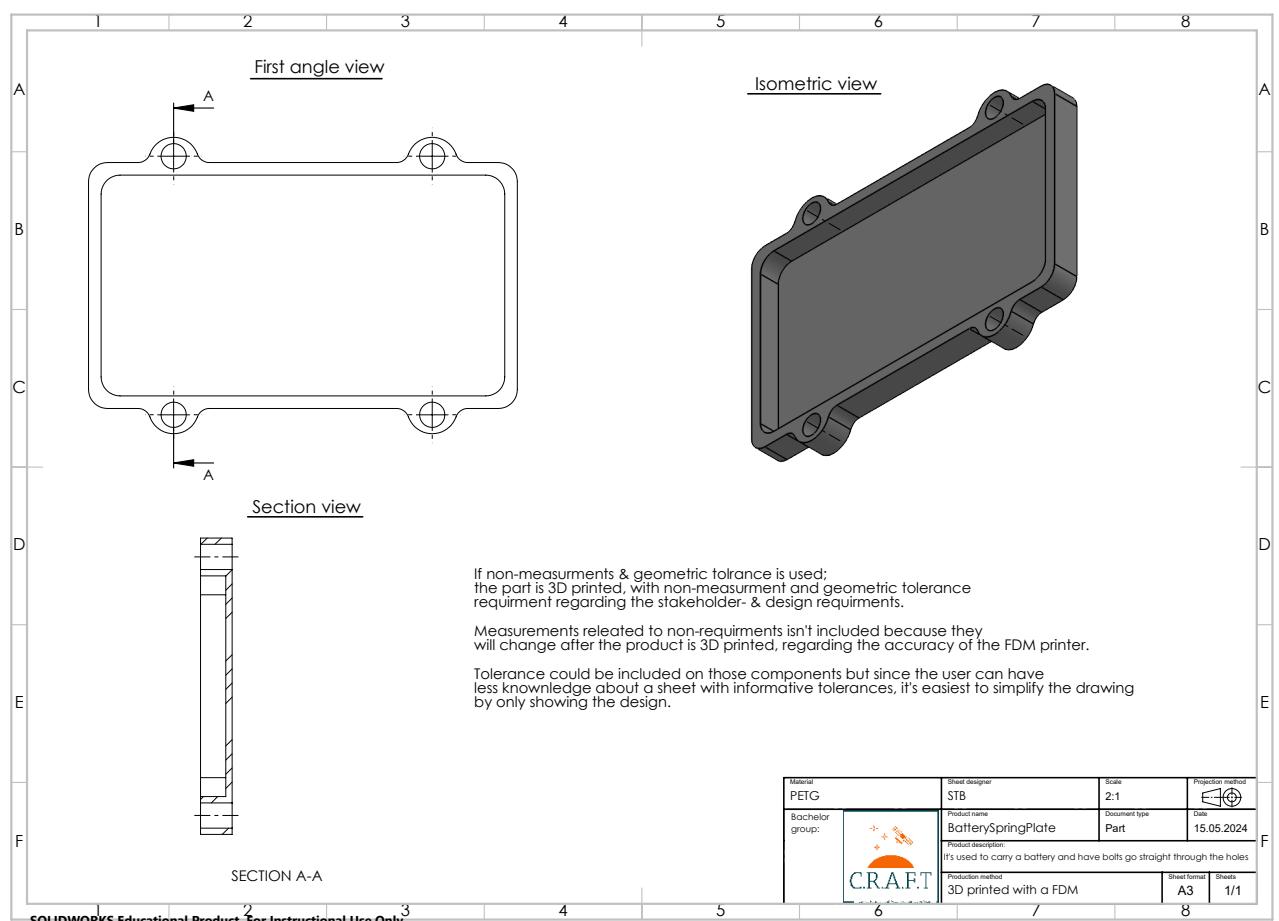


Figure 275: *BatterySpringPlate* - sheet | STB

Part name: Pin_Sealing

The work started with defining the diameter's on the component and the hole size where a split pin is placed. Some of version of *Pin_Sealing* had to small walls between the small

diameter long cylinder part. We haven't pictures of those versions, therefore, they are not described further. Instead, a further version which we have a figure of is described as the first version. The first version of the *Pin_Sealing*, fig 276 was designed to have a split pin placed between the top layers on the *Pin_Sealing*. When this version was printed, failure occurred in the area of the pin placement.

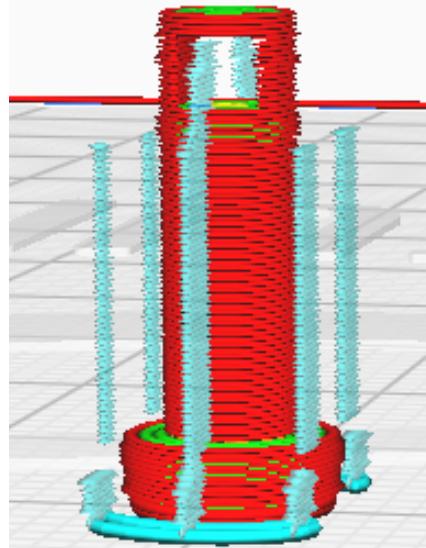


Figure 276: First version - *Pin_Sealing* | STB

The second *Pin_Sealing* version, fig 277 had the same length but be changed the position to a state where the component doesn't need supports. This also ended with print failure.

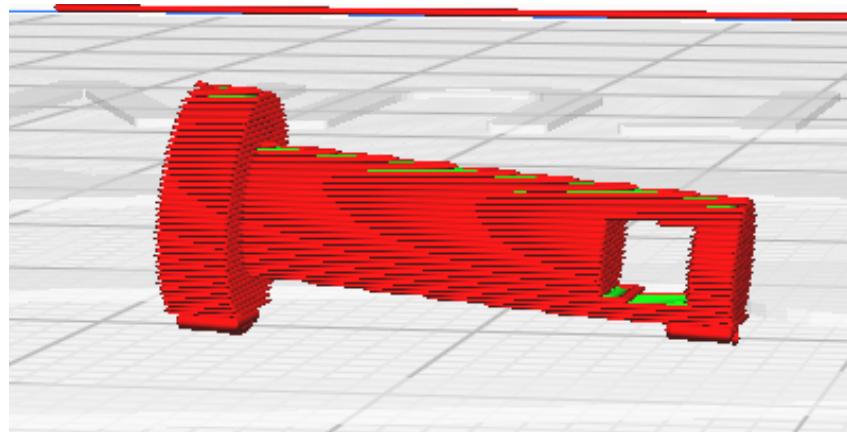


Figure 277: Second version - *Pin_Sealing* | STB

The third *Pin_Sealing* version, fig 278 was a change of the amount of layers between the top layer of the hole used for a split pin and the top layer of the *Pin_Sealing*.

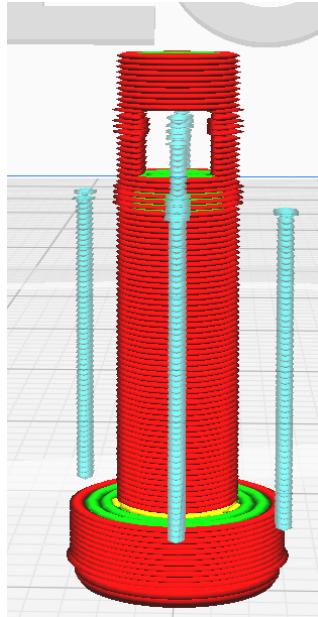


Figure 278: Third version - *Pin_Sealing* | STB

The fourth (& final) *Pin_Sealing* version, fig 279 was a change of the amount of layers between the top layer of the hole used for a split pin and the top layer of the *Pin_Sealing*.

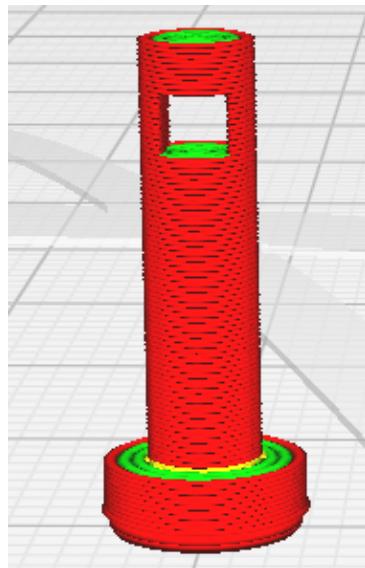


Figure 279: Fourth version - *Pin_Sealing* | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: print the fourth version of *Pin_Sealing*, fig 279 with non-supports and with the large-diameter cylinder on to the print bed surface (equal figure).

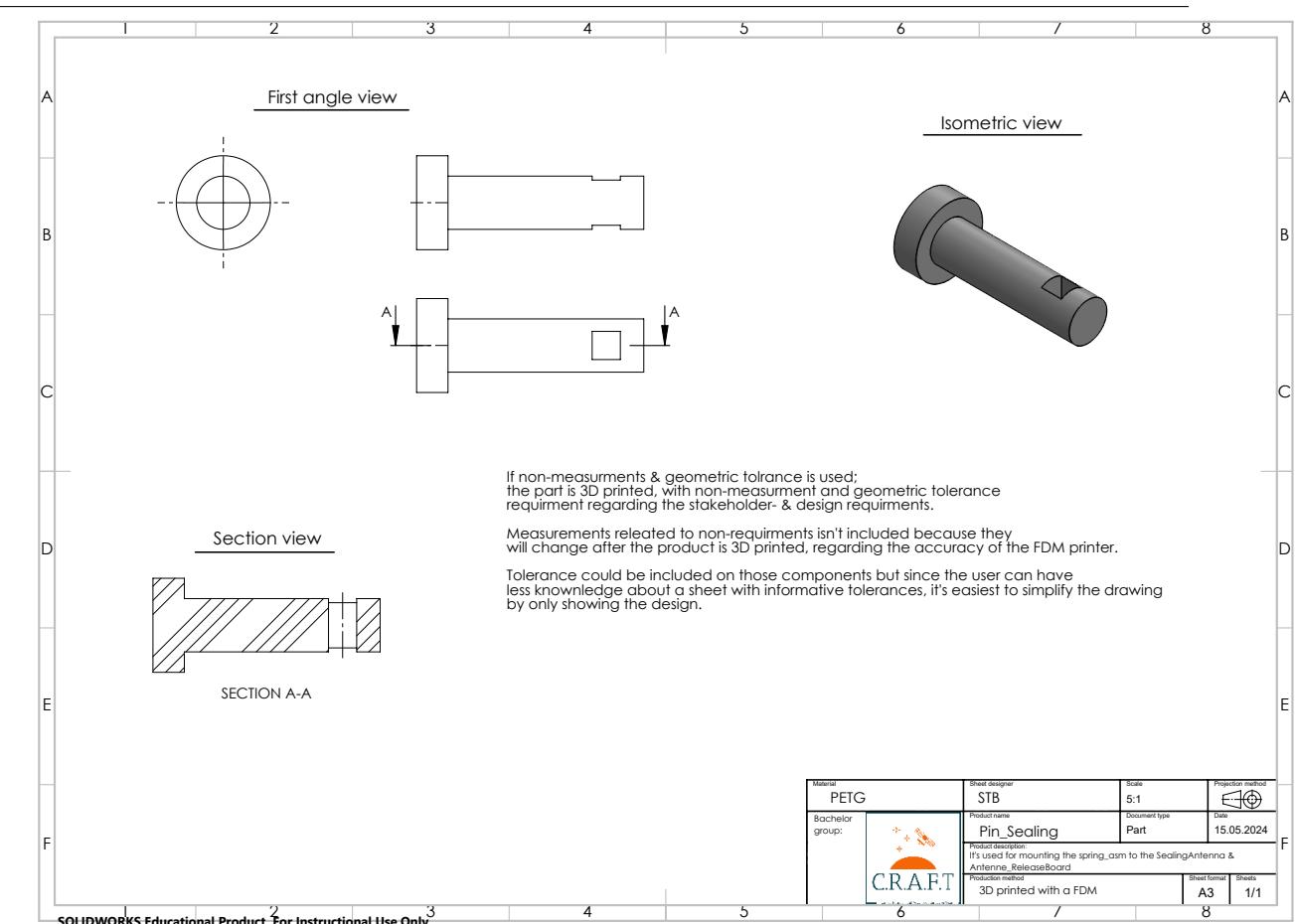


Figure 280: *Pin_Sealing* - Sheet | STB

Part name: AntennaBoard_AntennaLockRoof

The *AntennaBoard_AntennaLockRoof*, fig 281 is designed to be glued on the *Antenna_ReleaseBoard* and have a stick drag through the rectangular cut area. This stick represents a system before the launch (a system meant for non-antenna releasing).

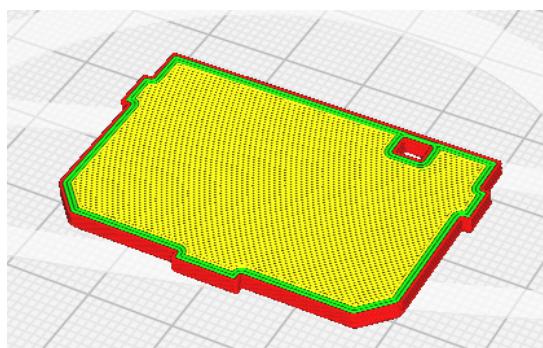


Figure 281: *AntennaBoard_AntennaLockRoof* | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: print the *AntennaBoard_AntennaLockRoof*, fig 281 flat with no supports.

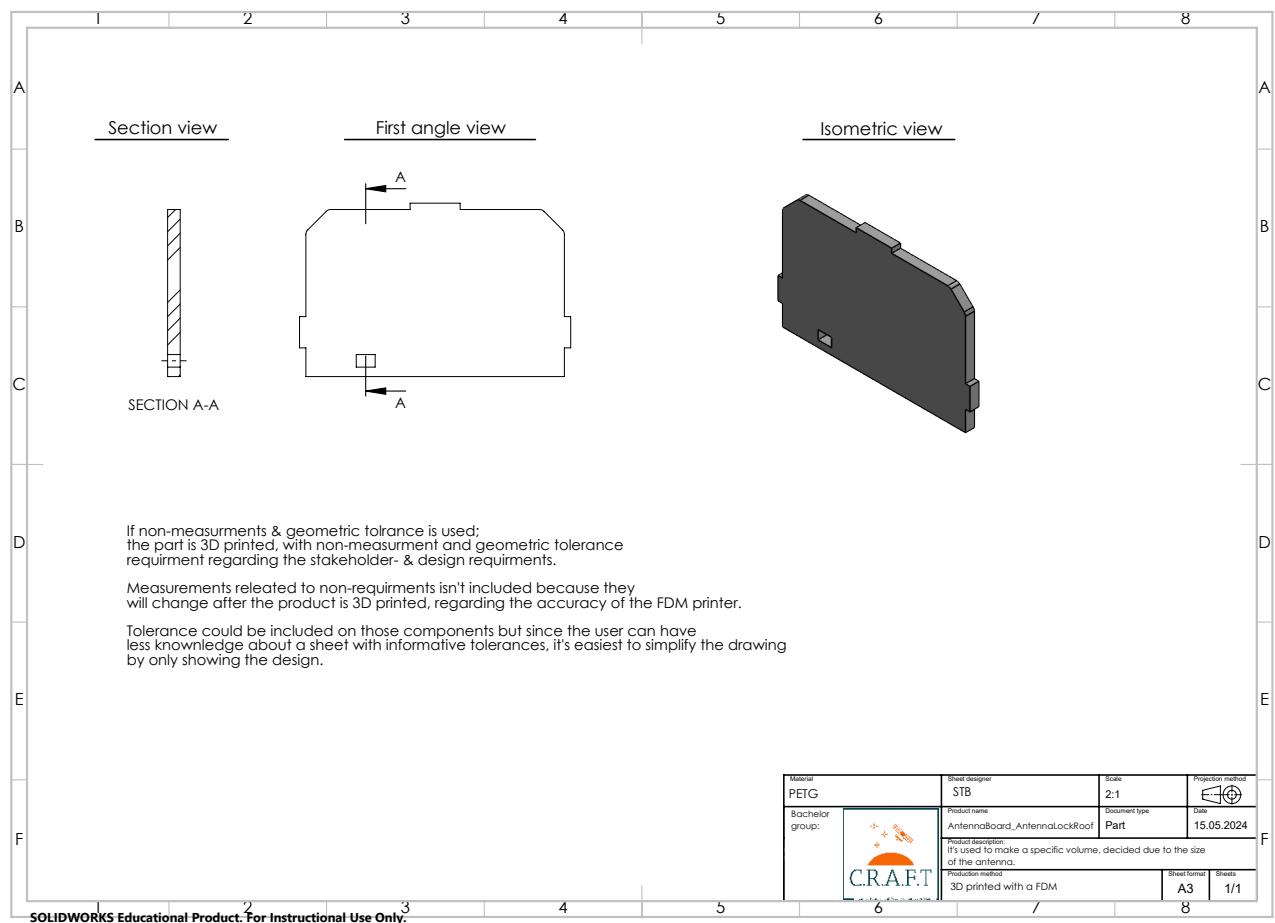


Figure 282: *AntennaBoard_AntennaLockRoof* - sheet | STB

Part name: Camera_Holder

The *Camera_Holder*, fig 283 is designed to hold the *ESP3-Camera shield* and be mounted equally as a *HingeLockPart* on the *ZY_wall*.

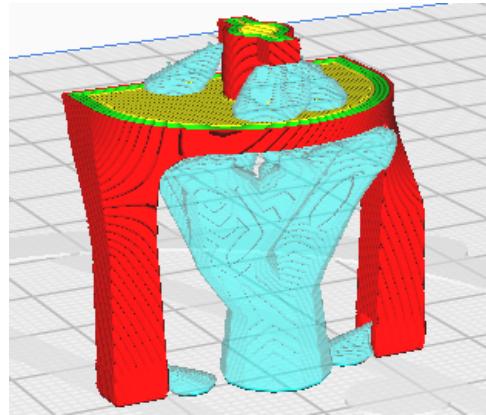


Figure 283: *ESP3-Camera shield* holder | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: print the *Camera_Holder*, fig 283 with supports (Cura setting; support structure tree - everywhere) from the printing bed to the overhang surfaces of the structure.

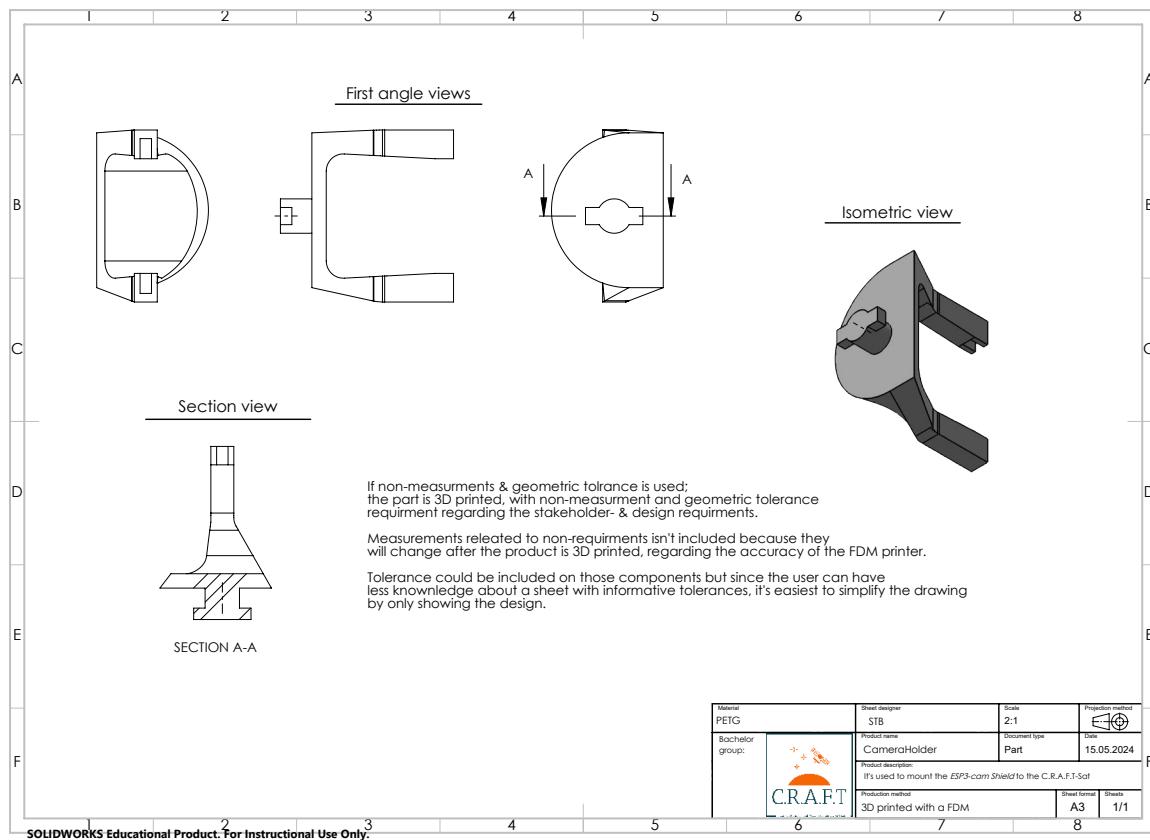


Figure 284: *Camera_Holder* - sheet| STB

Part name: Hook_Plate or Hook_Parts

The *Hook_Plate*, fig 285 is designed with a slot for the *Plate_Lock* and holes for M3 bolts.

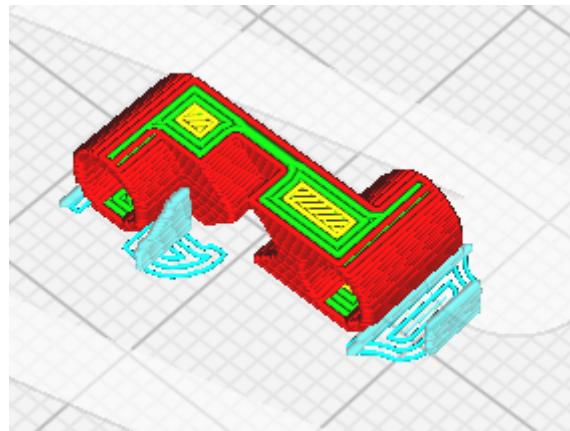


Figure 285: Hook_Plate | STB

Print recommendation of the used component in the C.R.A.F.T-Sat: print the *Hook_Plate* with the side surface parallel and on to the bed, equal fig 285. It's possible to use supports (Cura setting; support structure normal - touching build plate) on the outside surface but isn't a must.

Part name: Plate_Lock

The *Plate_Lock*, fig 287 is designed with a specified curvature meant to fit the 4x*Hook_Plate* and stiff them for rotation in the XY-axis of the C.R.A.F.T-Sat.

Print recommendation of the used component in the C.R.A.F.T-Sat: print the *Plate_Lock* with no supports, and flat (equal fig 287) on the print bed.

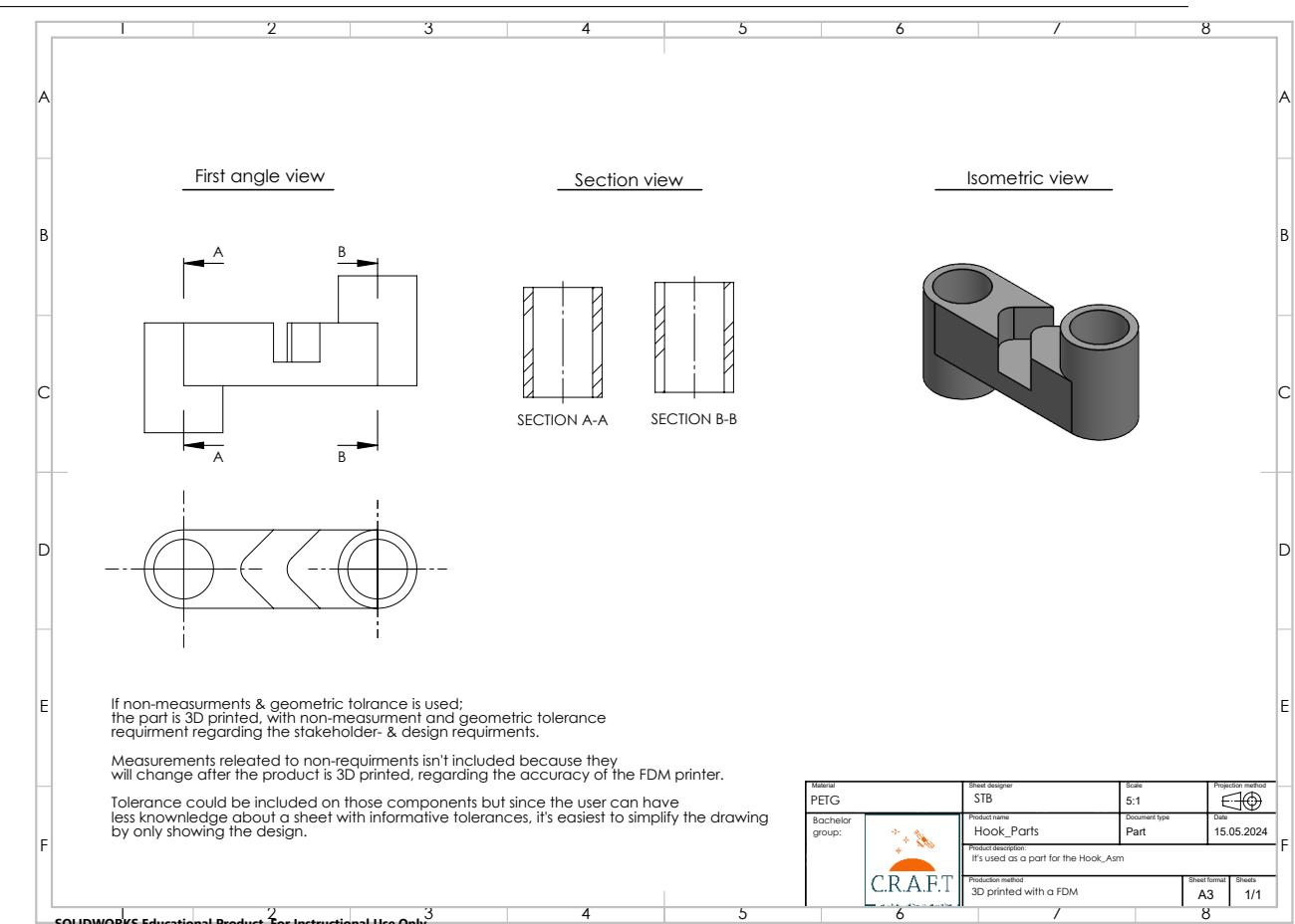


Figure 286: *Hook_Parts* - sheet | STB

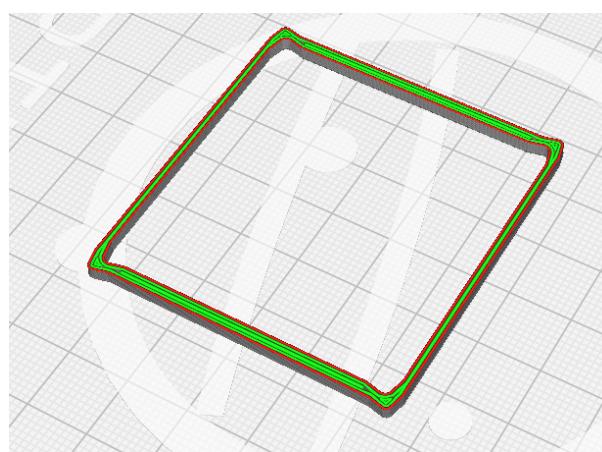


Figure 287: *Plate_Lock* | STB

M.4 Circuit boards - sheet

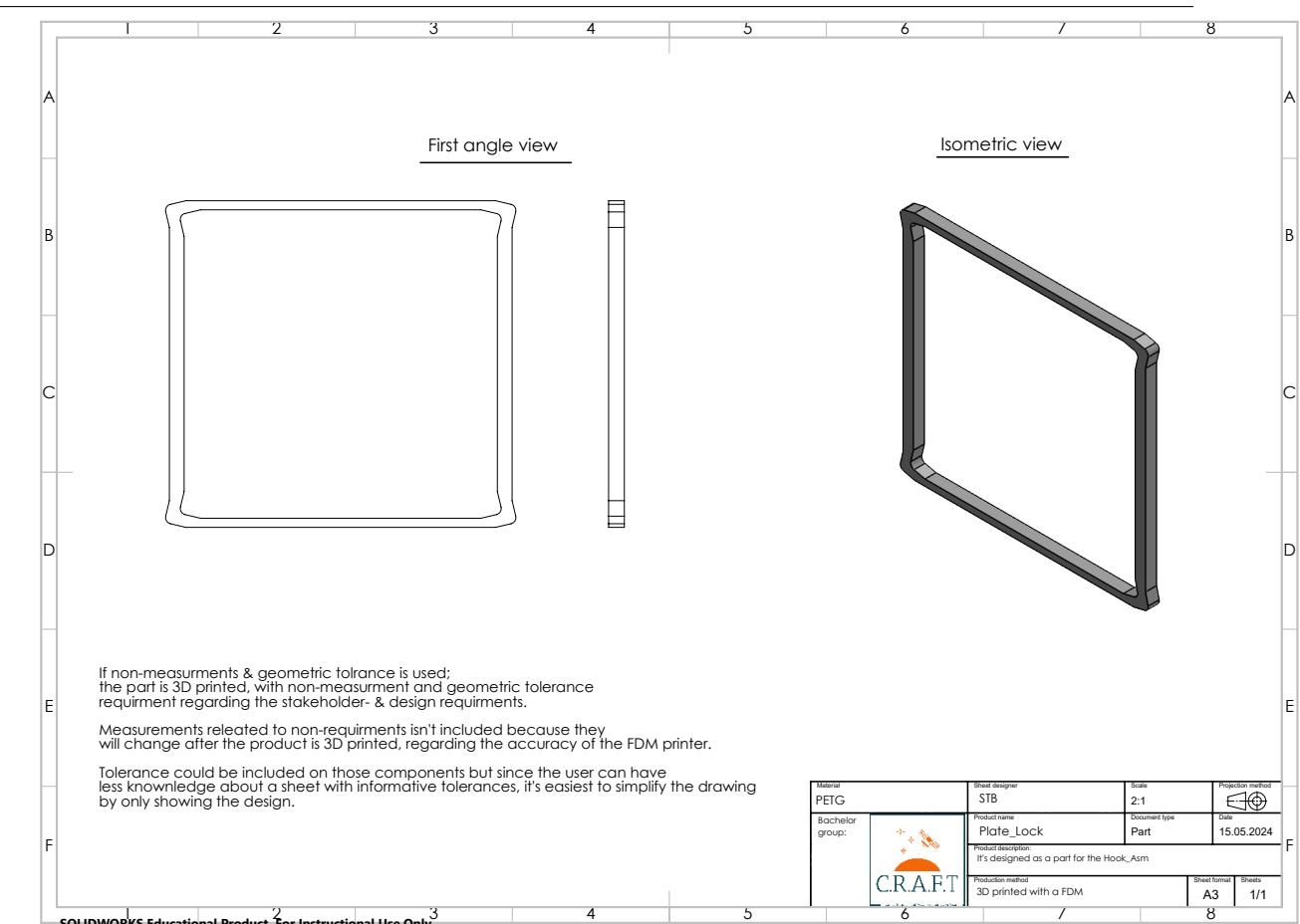


Figure 288: *Plate_Lock* - sheet | STB

STB|STB

M.5 Assemblies

Mount the component equal the BOM (Bills Of Material) sheets.

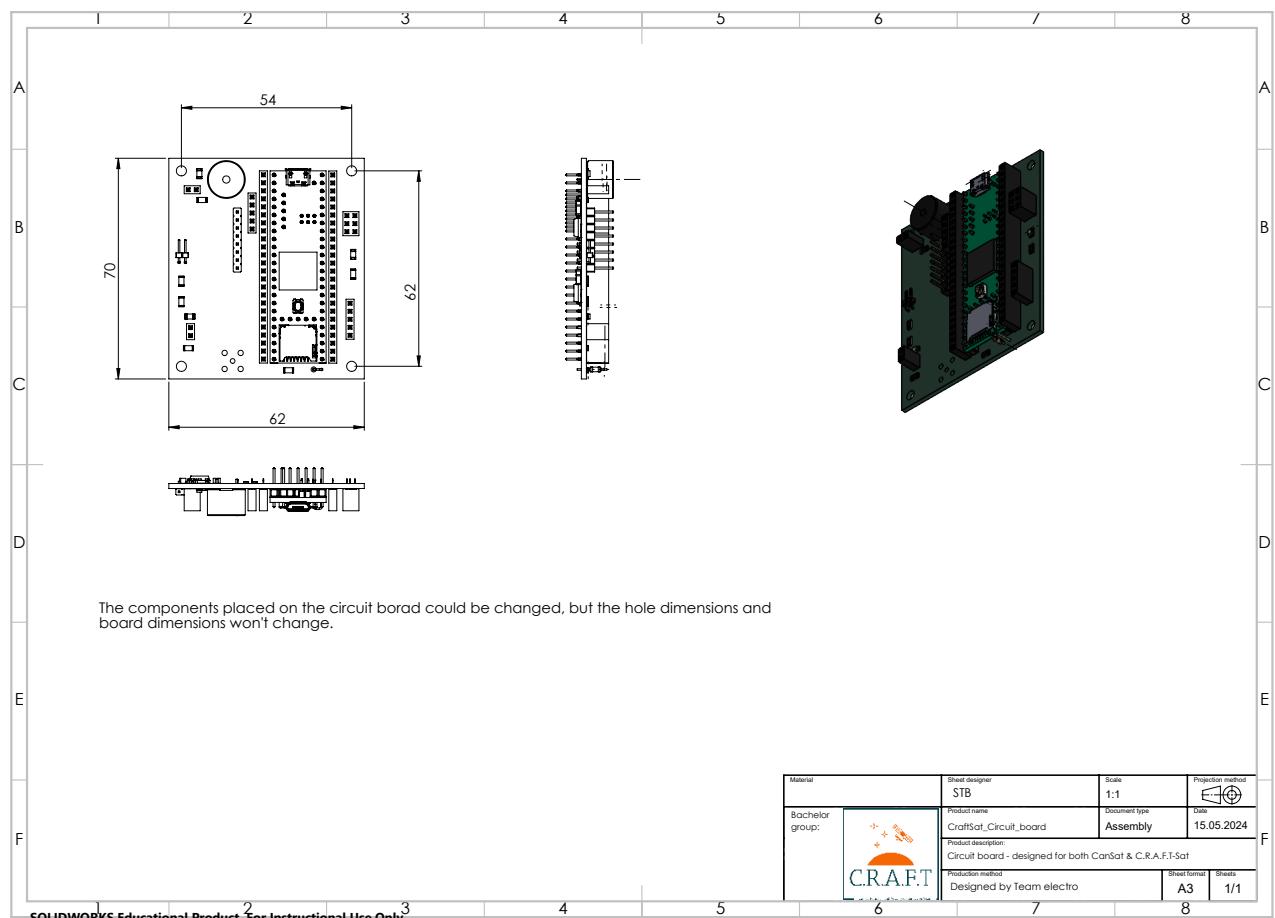


Figure 289: Circuit board | STB

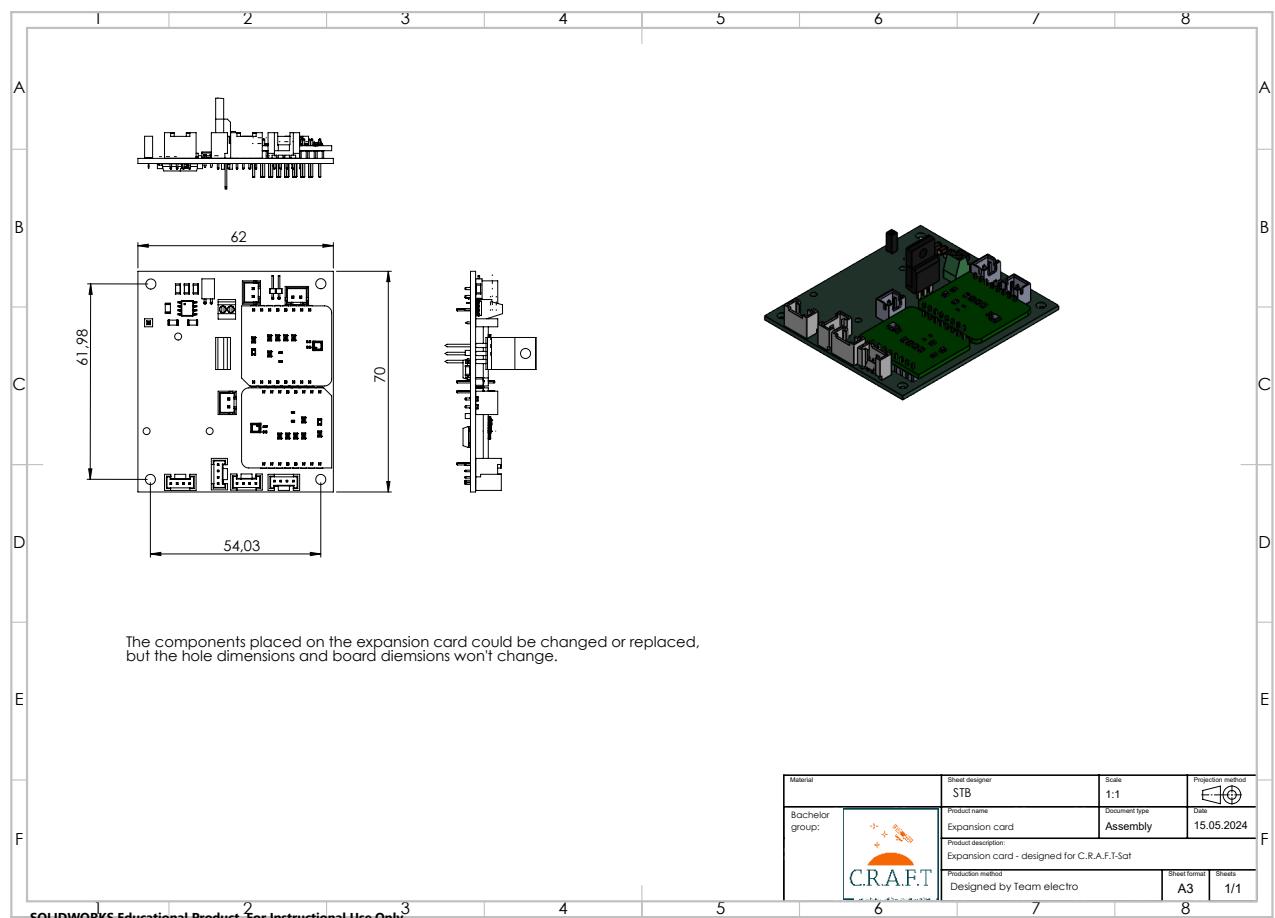


Figure 290: Expansion Card | STB

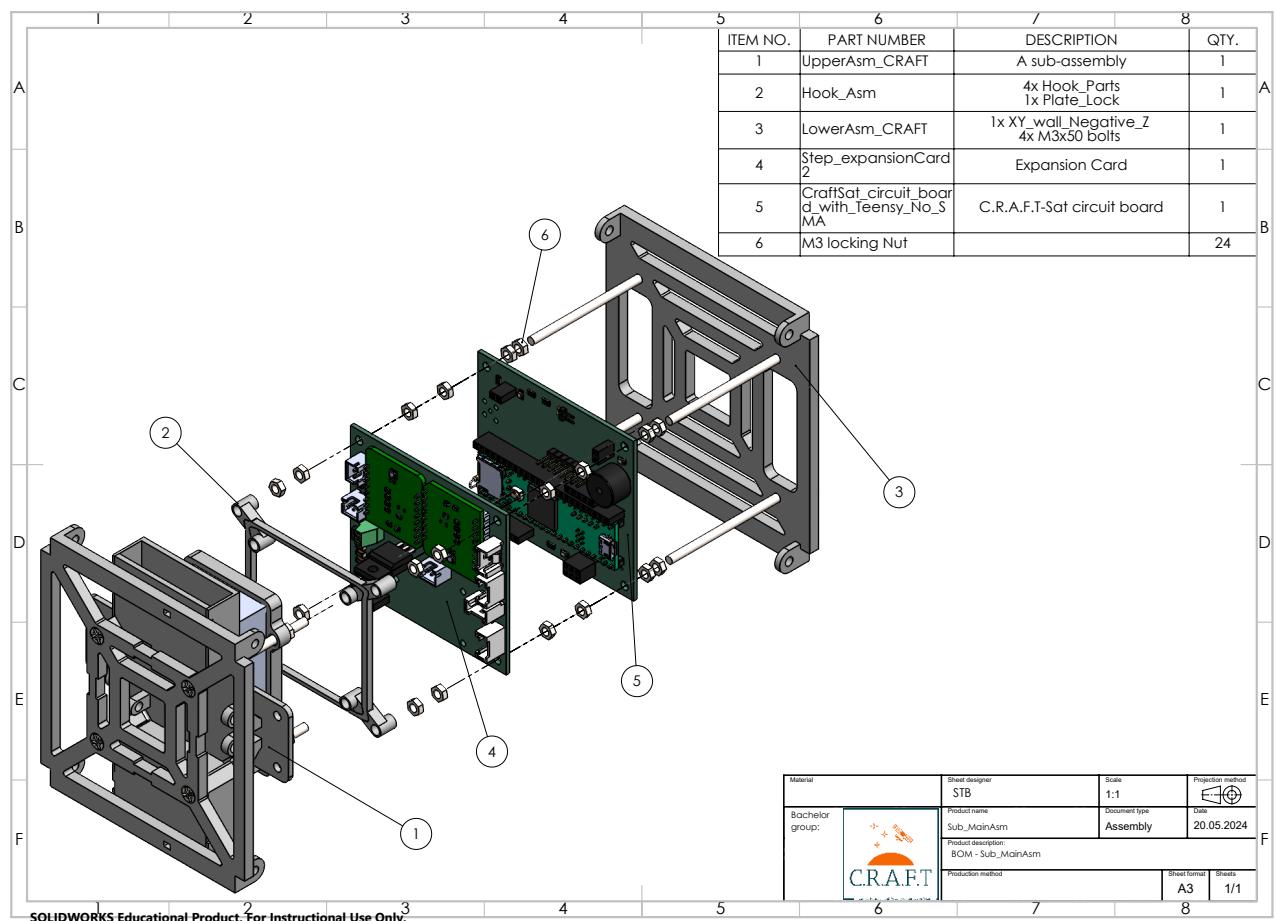


Figure 291: *Sub_MainAsm | STB*

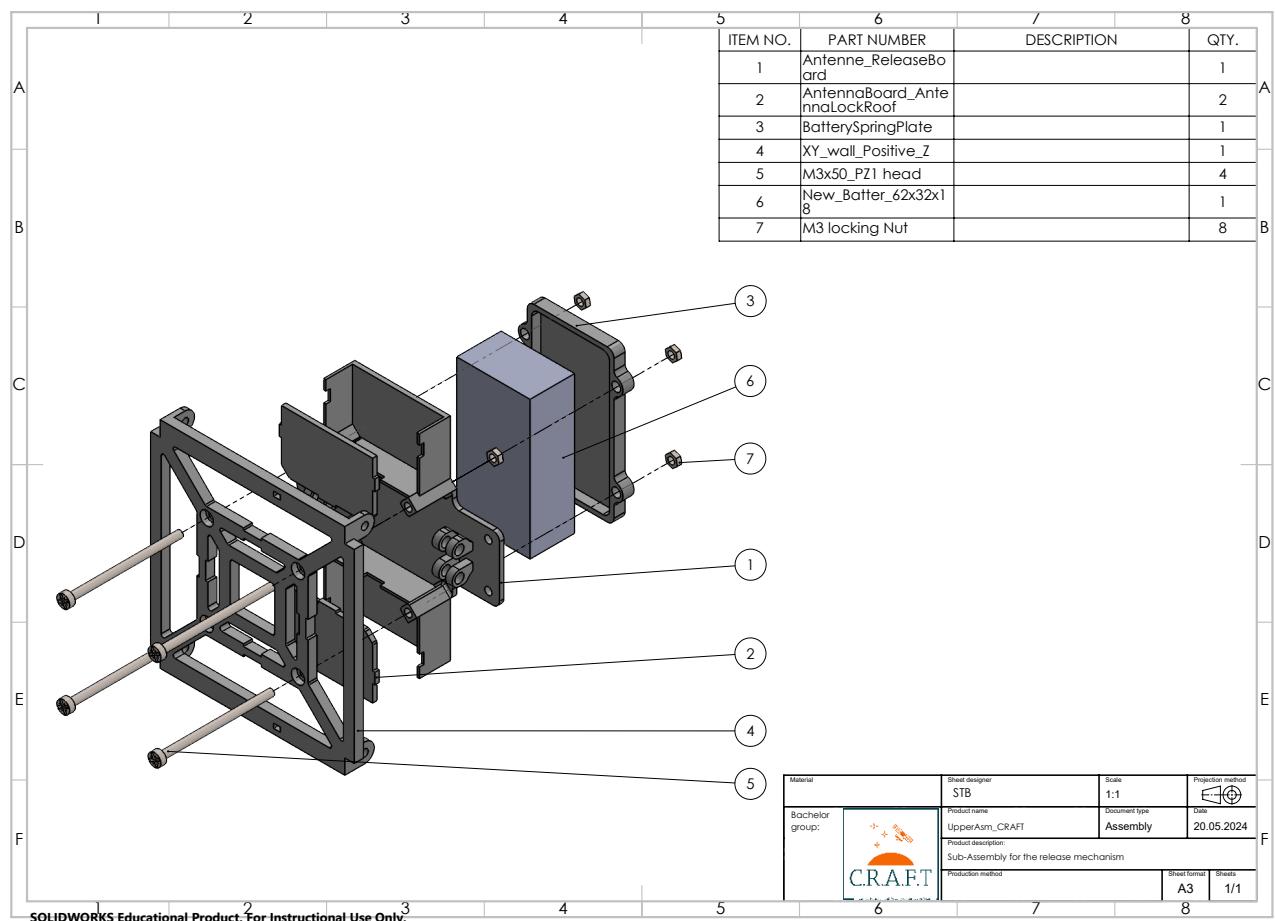


Figure 292: *UpperAsm_CRAFT* | STB

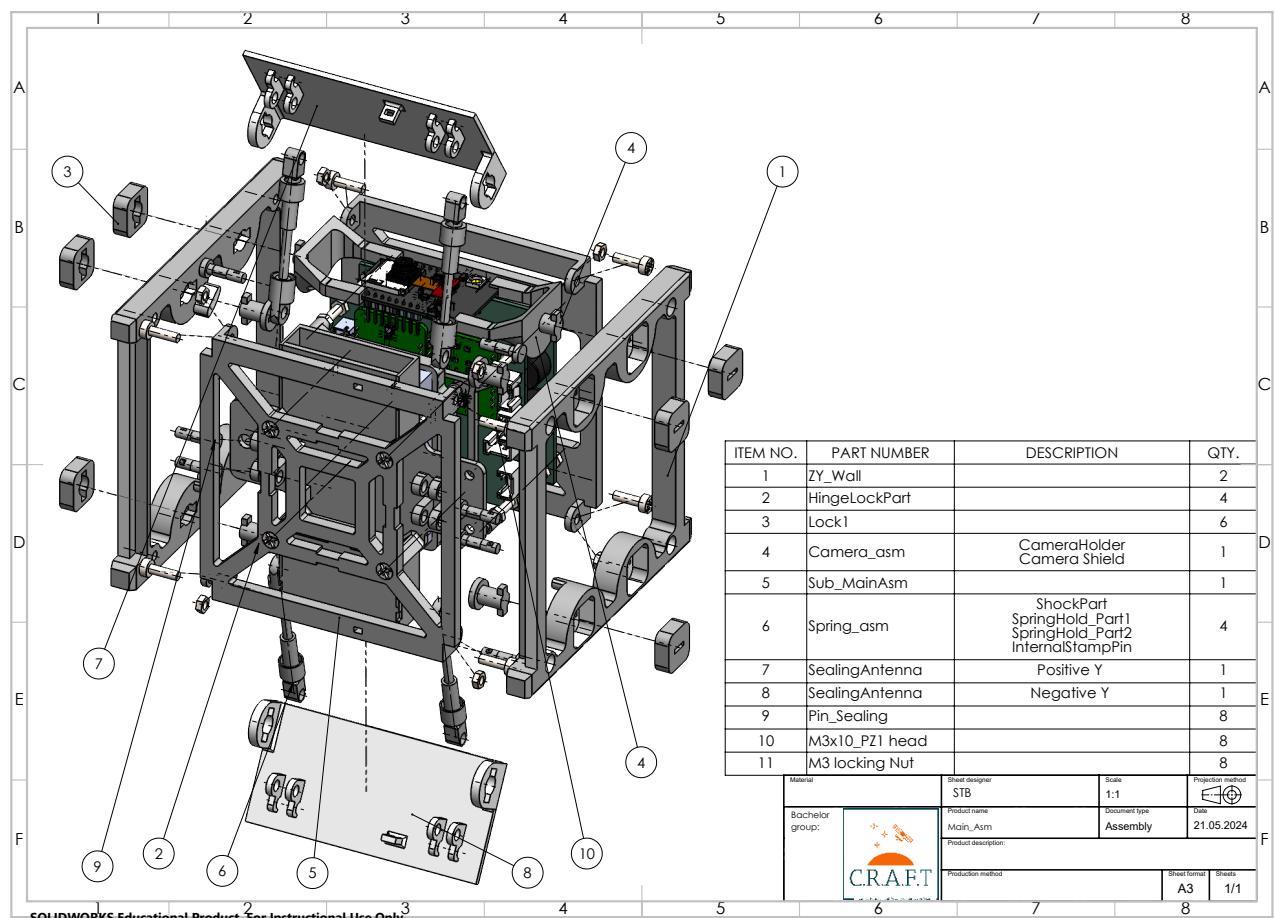


Figure 293: *Main_Asm | STB*

N Rough Mechanical Test Plan

This will be a rough test plan off what and how we can test the frame and the antenna release mechanism.

This should be tested:

- The stability and usability of the frame.
- If the antenna release mechanism is working as planned
- The assembly/interfaces between the frame and:
 - Antenna release mechanism
 - CanSat components
 - The stability in a vibration-test.

This is how they will be tested:

- Assembly and disassembly the frame, and check if it's optimal for our product.
- Assembly the mechanical, electronical, and computer components; check if the design for the antenna release mechanism is working as planned.
- Check if the planned interfaces mountings is working between the frame and the antenna release mechanisms, and the frame and CanSat components.
- Check with a vibration test, if all the components used is safe to use, based on a vibration test.

O Documentation about polymer

This document is designed to be a short description about polymer material relative to the polymer material chosen in the C.R.A.F.T bachelor assignment.

O.1 Glass-transition temperature

In a amorphous polymer, the polymer chains exist without long-range order. The amorphous polymer doesn't have a specific melting point, but they undergo a distinct changes in their mechanical behaviour across a narrow range of temperatures. When the temperature is lower or equal to the materials glass temperature, the material become hard, rigid, brittle, and glassy. But when the temperature is higher than the material glass temperature the material become rubbery or leathery.[174].

Glass-transition temperature , glass point, or glass temperature is the definition of temperature, (T_g), when the transition occurs. This behavior is something many amorphous polymers experiencing. An exception is polycarbonate, which is neither rigid nor brittle below its glass-transition temperature. [174].

To determine the T_g , the specific volume of the polymer need to be plotted as a function of temperature. The glass point occurs at the point where there is a sharp change in the slope of the curve, fig 294. Unlike the amorphous polymer, partly crystalline polymers have a distinct meting point, T_m .

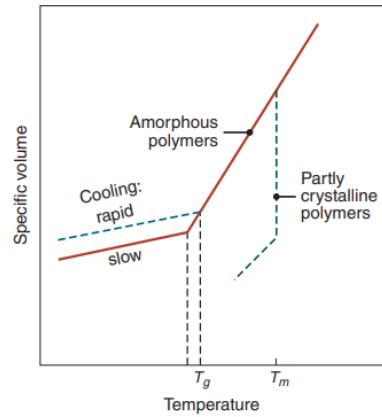


Figure 294: Specific volume of polymers as a function of temperature.

Heads up; The slope in a highly cross-linked polymer changes gradually near the glass point, making it difficult to determine the glass-transition temperature. The glass-transition temperature varies with the type of polymer, and can be above or below the room temperature.[174].

O.2 Thermoplastics

Thermoplastic has the behaviour of when the temperature is raised above a glass temperature, T_g , or melting point, T_m , the polymers become easier to shape or to mold into desired shapes. When the polymer is cooled, it returns to its original strength and hardness. Their behavior also depends on variables such as their structure and composition; among the most important is the temperature and deformation rate. If the temperature is below the glass-transition temperature, some polymers are brittle, such as glass and behave like an elastic solid. [174].

O.2.1 Effects of temperature

When the temperature of the thermoplastic polymer is raised above the glass temperature, it will first become leathery and then with further temperature increasing, rubbery. When the temperature is above the melting point for crystalline thermoplastic, it becomes a viscous fluid with the factor of decreasing the viscosity when the temperature is further increased. Since it is a viscous fluid, it can be softened, it can be used to molding into other shapes, re-solidified, and remelted. This can be done several times, however, repeated heating and cooling causes thermal aging or degradation of thermoplastics. [174].

O.2.2 Water Absorption

An important factor of some polymers is their ability to absorb water. When the moisture absorption is increased, the glass temperature, yield stress, and the elastic modulus of the polymer typically become rapidly lower; changes in the dimensions also occur. [174].

O.2.3 Thermal and Electrical Properties

Generally is polymer is characterized by, [174].:

- Low thermal and electrical conductivity.
- Low specific gravity (0.90 to 2.2).
- High coefficient of thermal expansion.

O.2.4 Material data from a digital database

Some data was difficult to gain for the thermoplastic by testing. Therefore, we decided to use a material database and note important values used for FEA for both PLA and PETG. We used [179] to gain the data for PETG:

-
- Glass transition temperature: $79.0 - 85.0^{\circ}C$
 - Density: 1370 kg/m^3
 - Thermal conductivity: 0.225 W/(mK)
 - Specific heat capacity: 1300 J/(kgK)
 - Poisson's Ratio: 0.38
 - Thermal Expansion Coefficient. $3.8 /K$

and [183] to get data about the PLA:

- Glass transition temperature: $-43.0 - 120.0^{\circ}C$
- Density: 3410 kg/m^3
- Thermal conductivity: 0.170 W/(mK)

where K: kelvin

P Tension testing results

P.1 Material: PETG x Infill pattern: Cubic Infill density: 50% Preparation temp.: -18 to -20 Celsius

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -18°C to -20°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test



Figure 295: Before tensile test 1 | STB



Figure 296: After tensile test 1 | STB

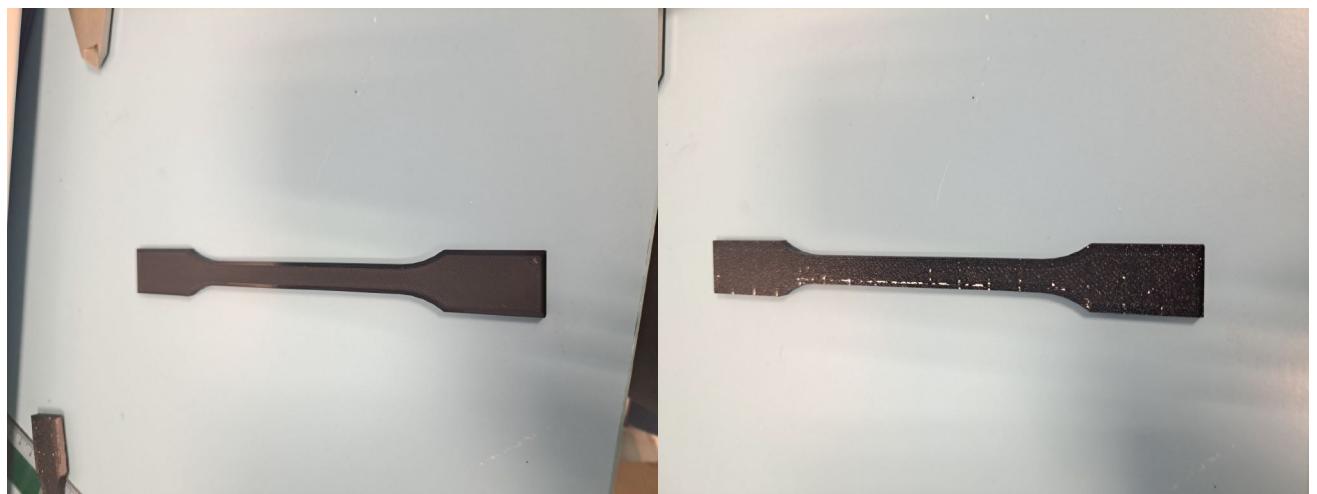


Figure 297: Before tensile test 2 | STB



Figure 298: After tensile test 2 | STB

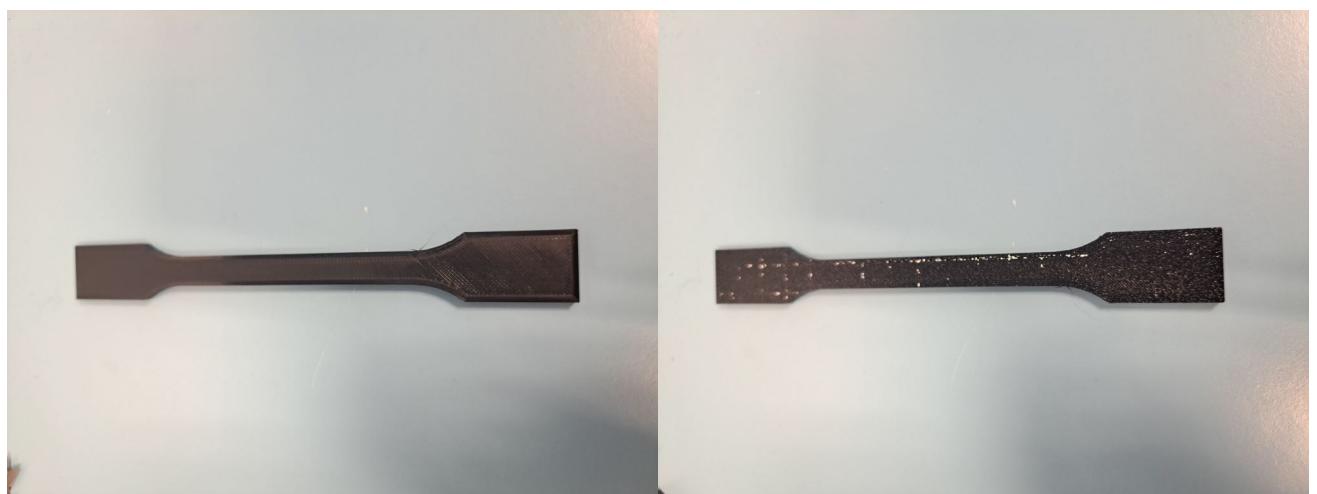


Figure 299: Before tensile test 3 | STB

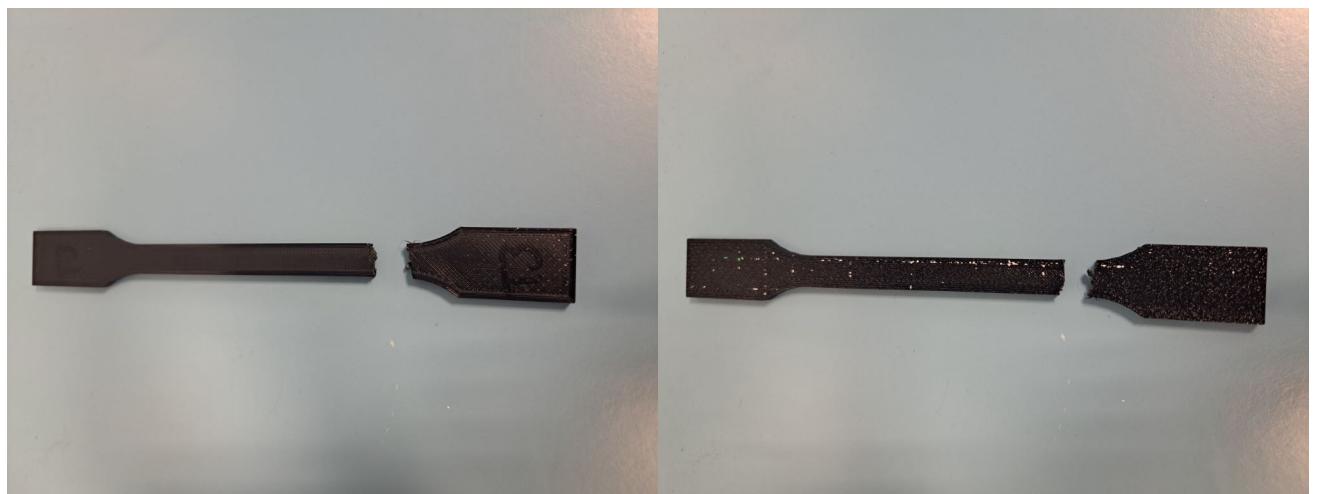


Figure 300: After tensile test 3 | STB

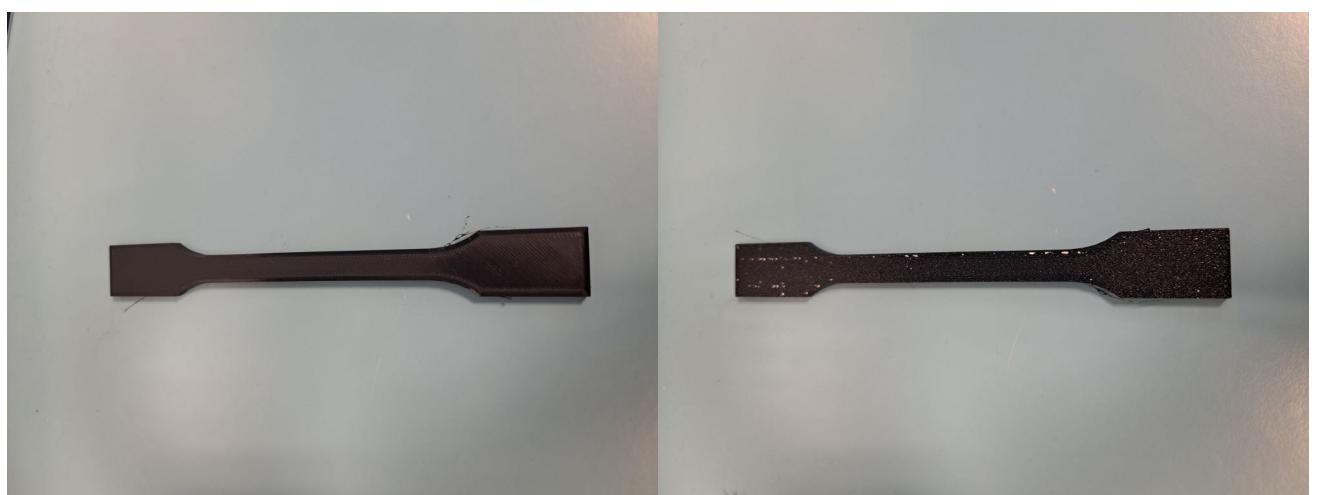


Figure 301: Before tensile test 4 | STB

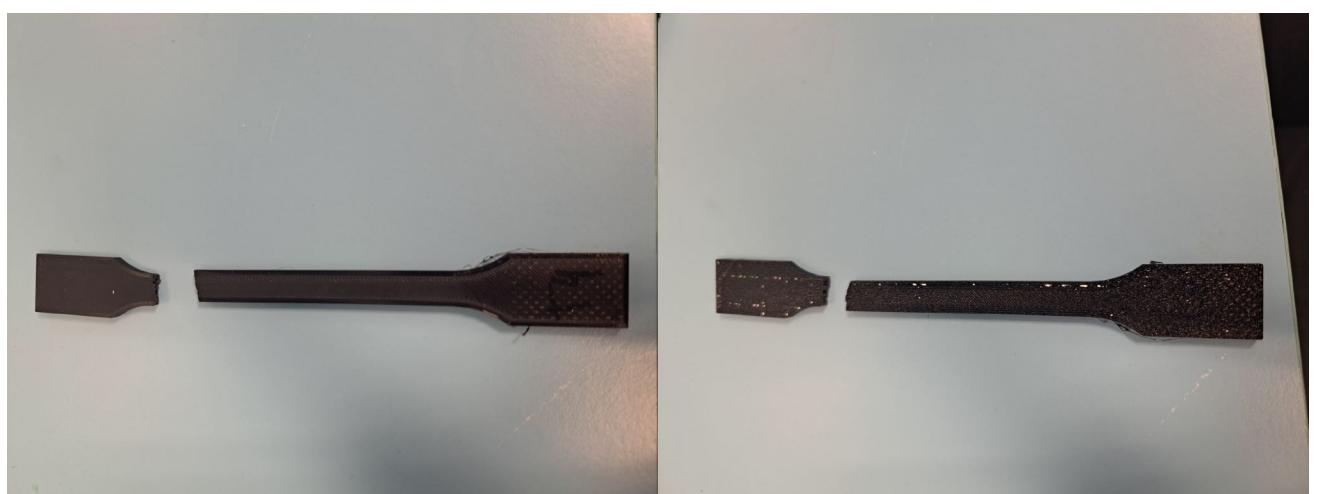


Figure 302: After tensile test 4 | STB

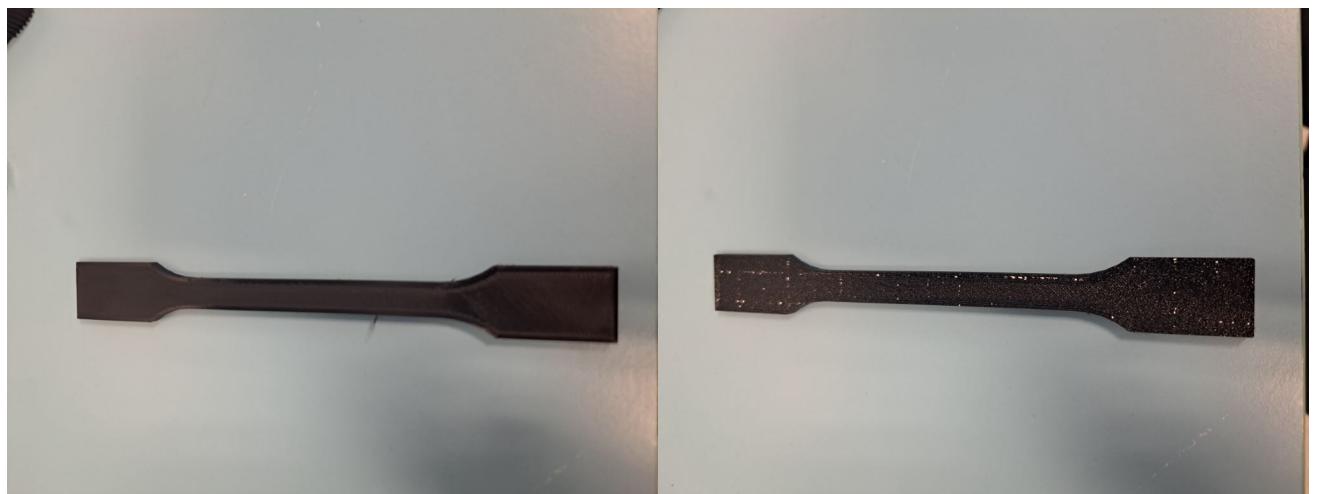


Figure 303: Before tensile test 5 | STB

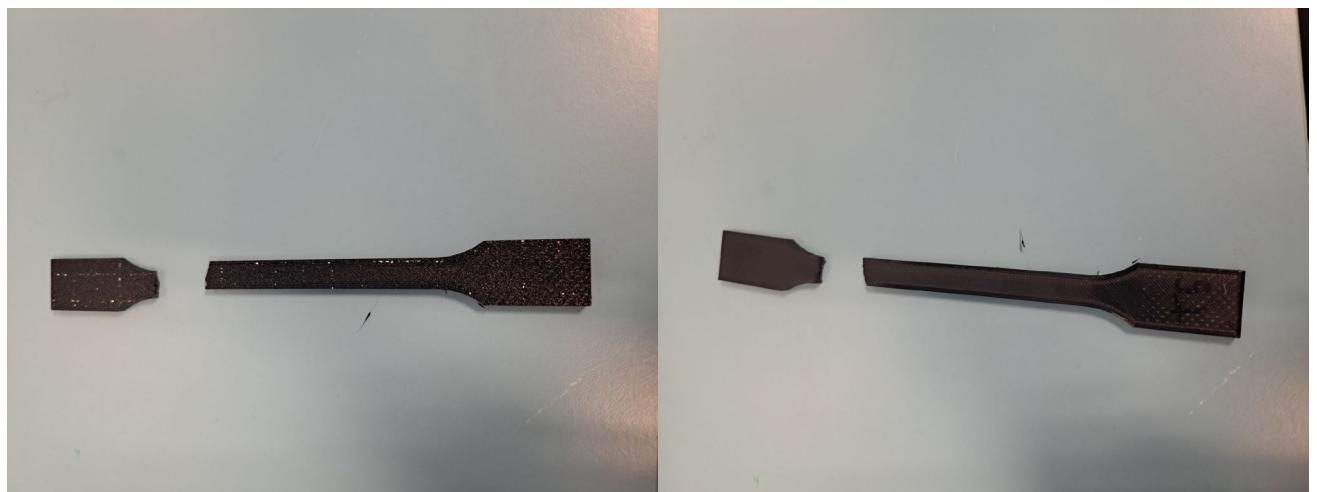


Figure 304: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 305.

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

16. April 2024

STB|

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
DATA	Initial values	L_0	Chosen start length [mm]	108,00	109,00	109,00	109,00	109,00	N/A
		h	Test staff thickness [mm]	4,13	4,12	4,10	4,12	4,14	4,12
		b_1	Width at narrow portion [mm]	9,99	10,00	10,01	10,10	9,99	10,02
Value (Settings/test plan)	A		Cross-section [mm ²]	41,26	41,20	41,04	41,61	41,36	N/A
		L	Gripping distance	108,00	109,00	109,00	109,00	109,00	N/A
		N/A	Amount of test	5	5	5	5	5	N/A
		v	Test Speed [mm/min]	60	60	60	60	60	N/A
Proportional limit	σ_y		Nominal Stress [MPa]	9,857	10,870	10,913	10,931	10,663	10,647
		ε_nom	Nominal strain [%]	0	0	0	0	0	0
		σ_true	True stress [MPa]	9,857	10,871	10,912	10,931	10,663	10,647
		ε_true	True strain [%]	0	0	0	0	0	0
		E_t	Tensile modulus [MPa]	N/A	394954,69	N/A	N/A	232455,63	125482,06
UTS	S_ut		Ultimate tensile strength [MPa]	34,132	37,678	35,197	32,769	32,699	34,495
		ε_nom	Nominal strain [%]	3,61	4,40	3,52	3,07	3,12	3,54
		σ_true	True stress [MPa]	35,364	39,335	36,437	33,774	33,72	35,726
		ε_true	True strain [%]	3,54	4,31	3,46	3,02	3,07	3,48
Fracture	σ_b		Stress at break [MPa]	21,306	35,109	34,719	13,45	4,384	21,794
		ε_b	Strain (engineering) at break [%]	3,65	5,00	3,57	3,12	3,17	3,70
		σ_true	True stress [MPa]	22,085	36,863	35,958	13,869	4,523	22,660
		ε_true	True strain [%]	3,59	4,88	3,51	3,07	3,12	3,63

Figure 305: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

P.2 Material: PETG x Infill pattern: Cubic Infill density: 50% Preparation temp.: -38 to -40 Celsius

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based **NS-EN ISO 527-1:2019**, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -38°C to -40°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between 20°C and 24°C before tensile test.

Pictures - before & after tensile test



Figure 306: Before tensile test 1 | STB



Figure 307: After tensile test 1 | STB



Figure 308: Before tensile test 2 | STB



Figure 309: After tensile test 2 | STB

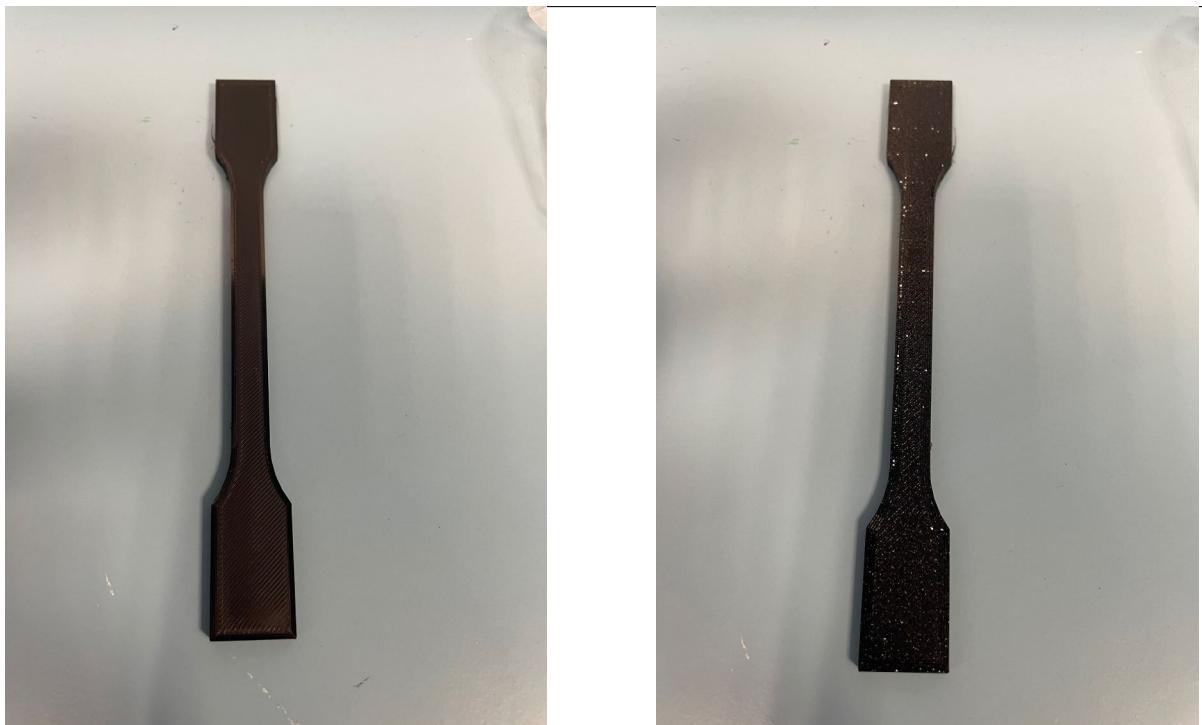


Figure 310: Before tensile test 3 | STB



Figure 311: After tensile test 3 | STB



Figure 312: Before tensile test 4 | STB

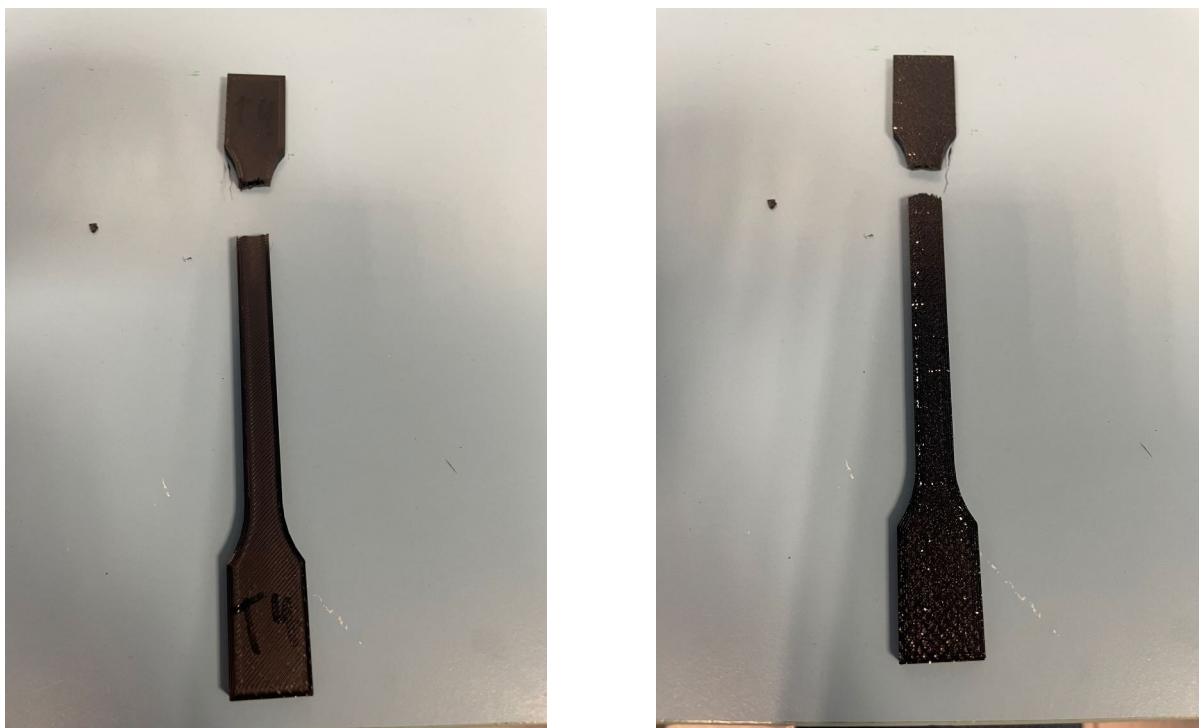


Figure 313: After tensile test 4 | STB



Figure 314: Before tensile test 5 | STB



Figure 315: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the tensile test is shown in fig 316.

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
DATA	Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
		h	Test staff thickness [mm]	4,10	4,10	4,12	4,09	4,09	4,10
		b_1	Width at narrow portion [mm]	9,99	10,00		10,03	9,99	8,00
Value (Settings/test plan)	A		Cross-section [mm ²]	40,96	41,00	0,00	41,02	40,86	N/A
		L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
	N/A		Amount of test	5	5	5	5	5	N/A
	v		Test Speed [mm/min]	60	60	60	60	60	N/A
Proportional limit	σ_y		Nominal Stress [MPa]	11,078	11,497	10,892	11,730	11,873	11,414
	ε_{nom}		Nominal strain [%]	1,1	1,15	1,16	1,03	1,12	1,112
	σ_{true}		True stress [MPa]	11,200	11,629	11,018	11,851	12,005	11,541
	ε_{true}		True strain [%]	1,1	1,14	1,15	1,03	1,1	1,104
	E_t		Tensile modulus [MPa]	1005,41	1000,14	940,02	1133,45	1061,61	1028,13
UTS	S_{ut}		Ultimate tensile strength [MPa]	35,555	35,830	35,298	37,172	38,496	36,470
	ε_{nom}		Nominal strain [%]	4,58	4,63	4,64	4,89	5,29	4,81
	σ_{true}		True stress [MPa]	37,185	37,489	36,937	38,988	40,531	38,226
	ε_{true}		True strain [%]	4,48	4,53	4,54	4,77	5,15	4,69
Fracture	σ_b		Stress at break [MPa]	18,663	35,785	34,822	36,861	36,289	32,484
	ε_b		Strain (engineering) at break [%]	4,63	4,68	4,73	3,98	5,84	4,77
	σ_{true}		True stress [MPa]	19,527	37,431	36,469	38,695	38,408	34,106
	ε_{true}		True strain [%]	4,53	4,57	4,62	4,86	5,68	4,85

Figure 316: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

18. April 2024

STB|

P.3 Material: PETG x Infill pattern: Cubic Infill density: 50% Preparation temp.: 20 to 24 Celsius

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019

-
- NS-EN ISO 291:2008
 - NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed room for 19 hours with the *preparation temperature* 20°C to 24°C. The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C.

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between 20°C and 24°C before tensile test.

Pictures - before & after tensile test



Figure 317: Before tensile test 1 | STB



Figure 318: After tensile test 1 | STB

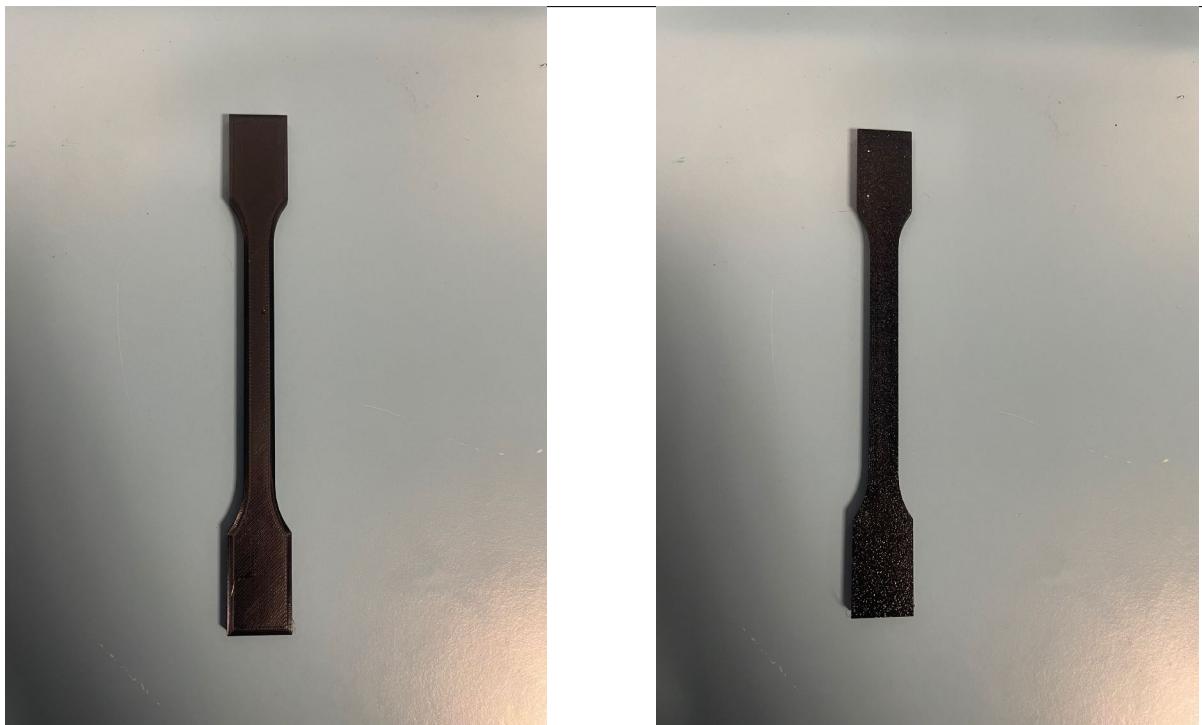


Figure 319: Before tensile test 2 | STB



Figure 320: After tensile test 2 | STB



Figure 321: Before tensile test 3 | STB

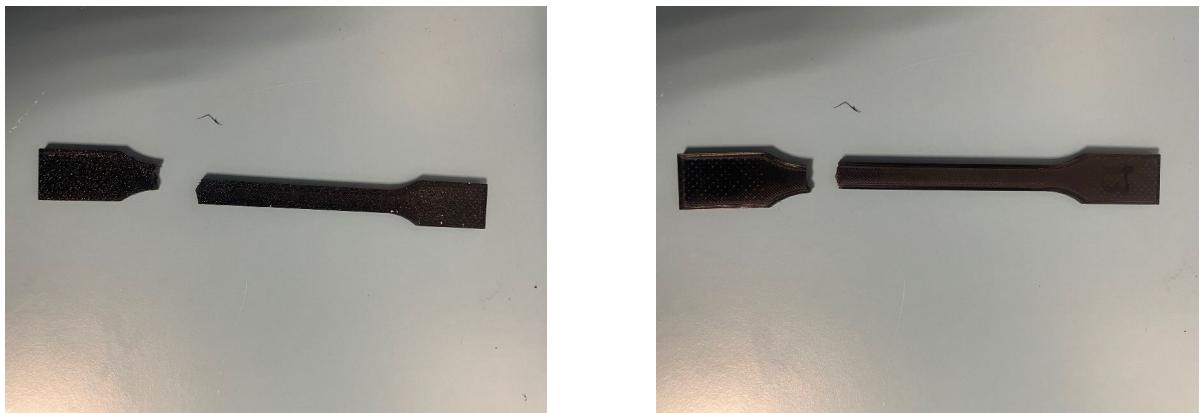


Figure 322: After tensile test 3 | STB

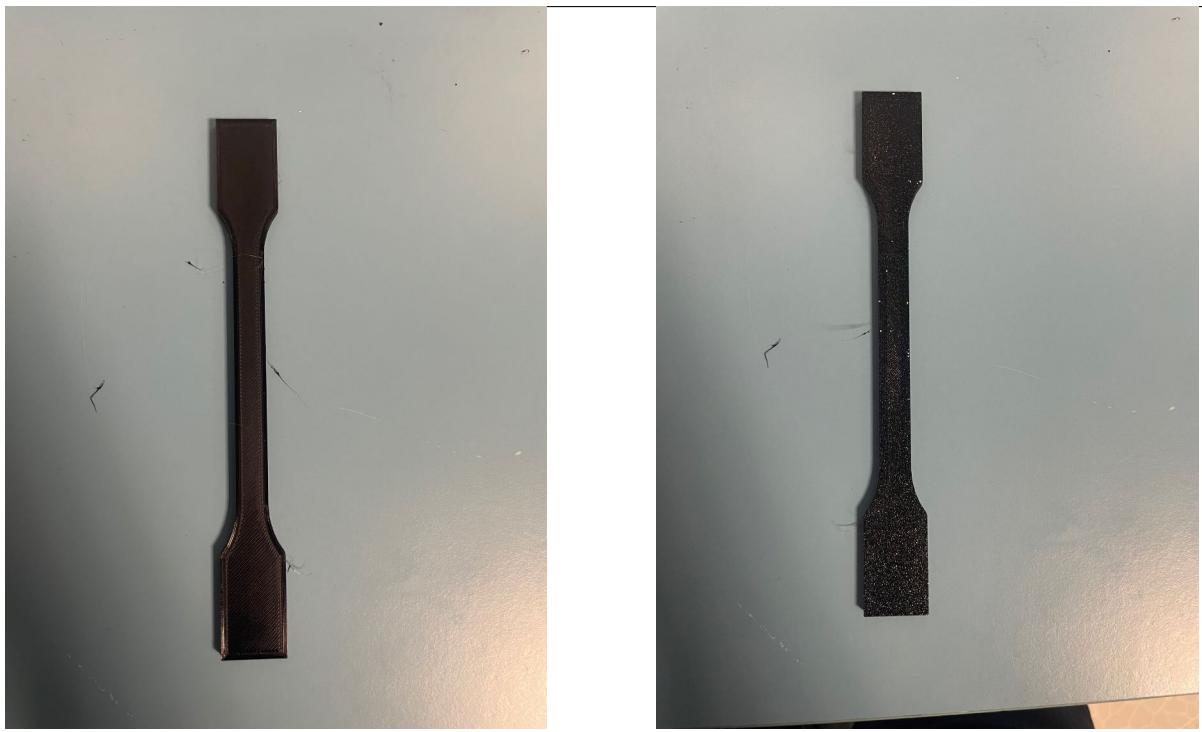


Figure 323: Before tensile test 4 | STB

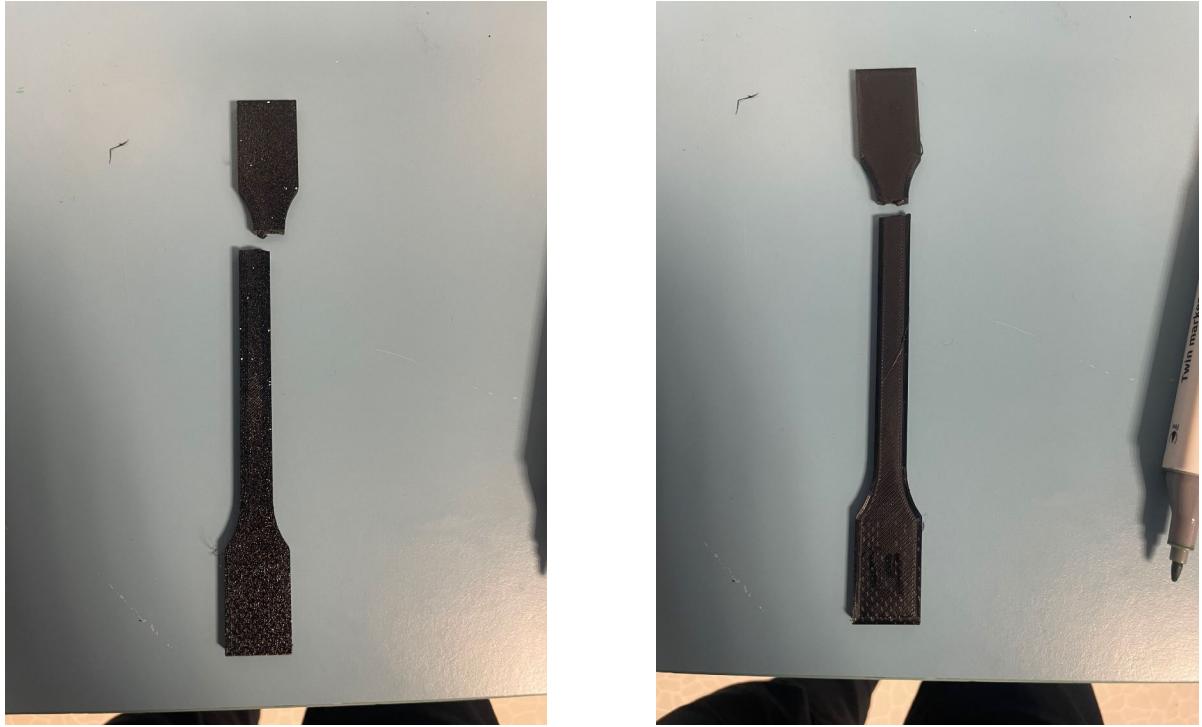


Figure 324: After tensile test 4 | STB



Figure 325: Before tensile test 5 | STB

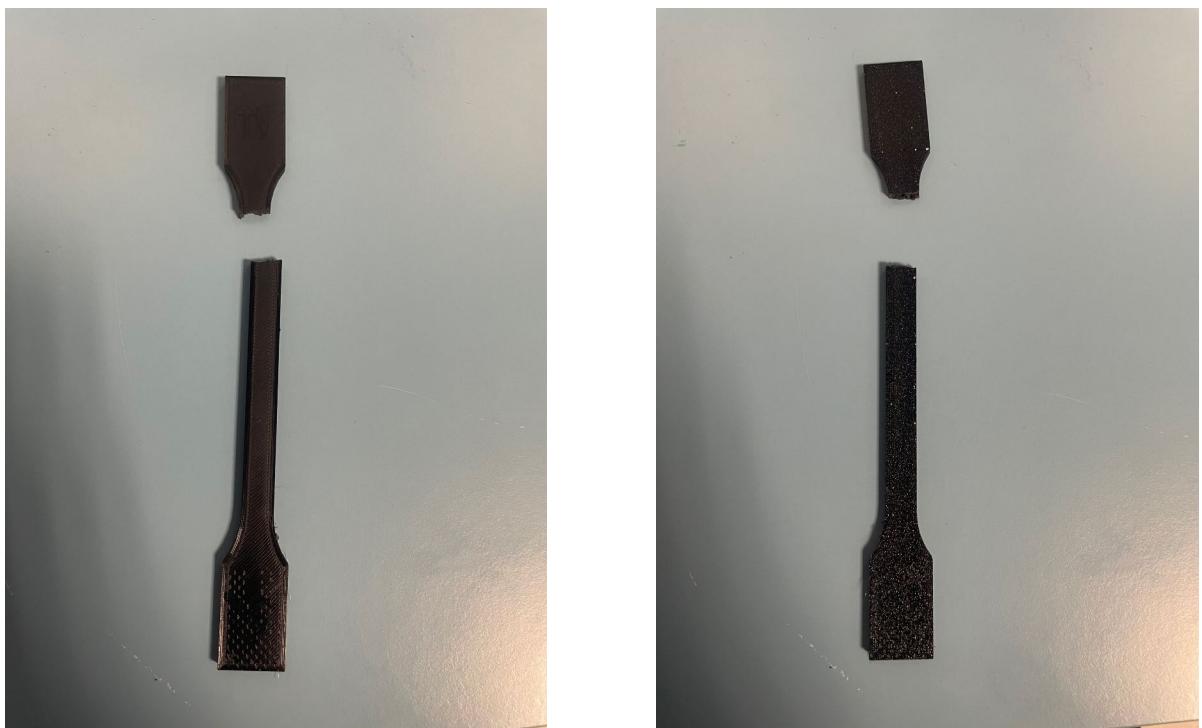


Figure 326: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the tensile test is shown in fig 327.

	A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values	
2		Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
3			h	Test staff thickness [mm]	4,10	4,10	4,07	4,10	4,07	4,09
4			b_1	Width at narrow portion [mm]	9,99	10,02	10,08	10,02	10,00	10,02
5		Value (Settings/test plan)	A	Cross-section [mm ²]	40,96	41,08	41,03	41,08	40,70	N/A
6			L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
7			N/A	Amount of test	5	5	5	5	5	N/A
8			v	Test Speed [mm/min]	60	60	60	60	60	N/A
9		Proportional limit	σ_y	Nominal Stress [MPa]	12,274	10,043	11,251	11,236	11,317	11,224
10			ϵ_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0
11			σ_{true}	True stress [MPa]	12,274	10,043	11,251	11,236	11,317	11,224
12			ϵ_{true}	True strain [%]	N/A	N/A	N/A	N/A	N/A	0
13		UTS	E_t	Tensile modulus [MPa]	N/A	N/A	408223,13	411179,75	163880,58	
14			S_{ut}	Ultimate tensile strength [MPa]	35,339	40,171	38,531	36,522	36,238	37,360
15			ϵ_{nom}	Nominal strain [%]	3,02	4,44	3,66	3,39	3,21	3,54
16			σ_{true}	True stress [MPa]	36,407	41,957	39,941	37,761	37,402	38,694
17		Fracture	ϵ_{true}	True strain [%]	2,98	4,35	3,60	3,34	3,16	3,49
18			σ_b	Stress at break [MPa]	35,172	39,074	38,459	25,071	1,348	27,825
19			ϵ_b	Strain (engineering) at break [%]	3,07	4,72	3,71	3,44	3,26	3,64
20			σ_{true}	True stress [MPa]	36,25	40,918	39,884	25,933	1,392	28,875
21			ϵ_{true}	True strain [%]	3,02	4,61	3,64	3,38	3,21	3,57

Figure 327: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

17. April 2024

STB|

P.4 Material: PETG x Infill pattern: Cubic Infill density: 75% Preparation temp.: -18 to -20 Celsius

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008

-
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -18°C to -20°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between 20°C and 24°C before tensile test.

Pictures - before & after tensile test



Figure 328: Before tensile test 1 | STB

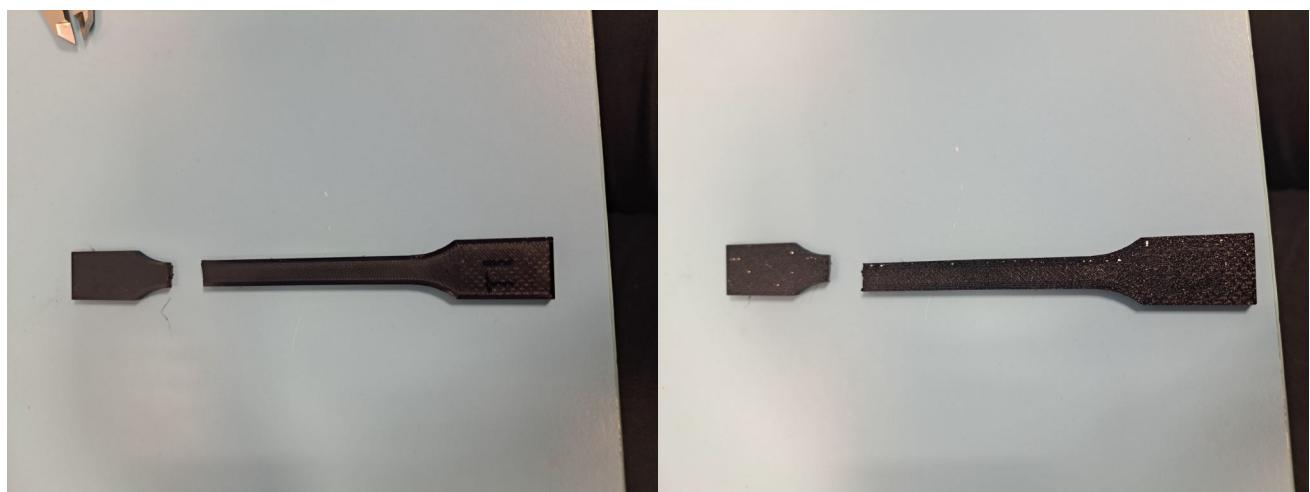


Figure 329: After tensile test 1 | STB

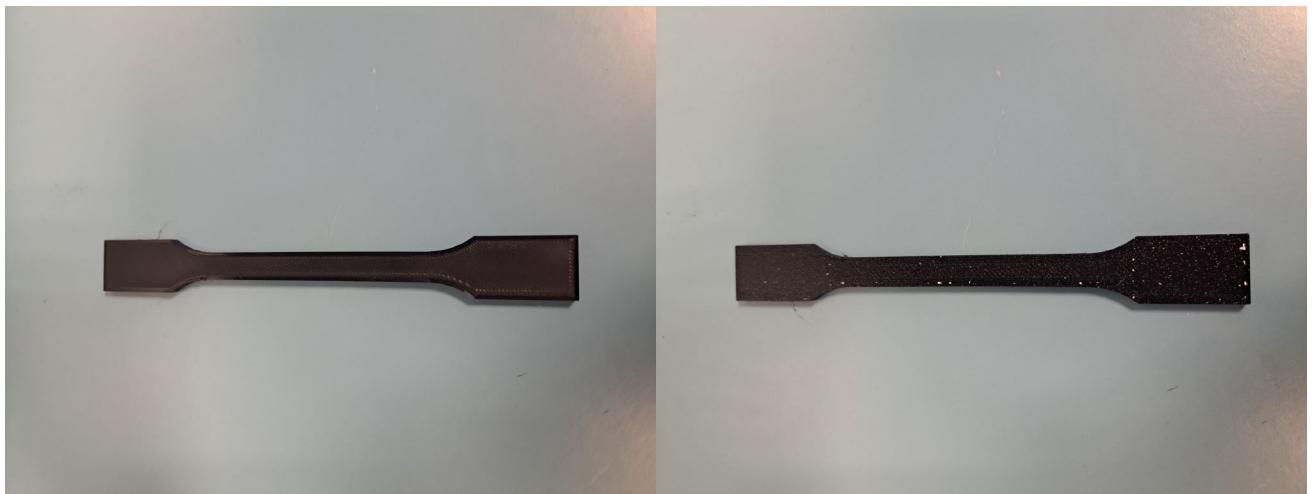


Figure 330: Before tensile test 2 | STB



Figure 331: After tensile test 2 | STB

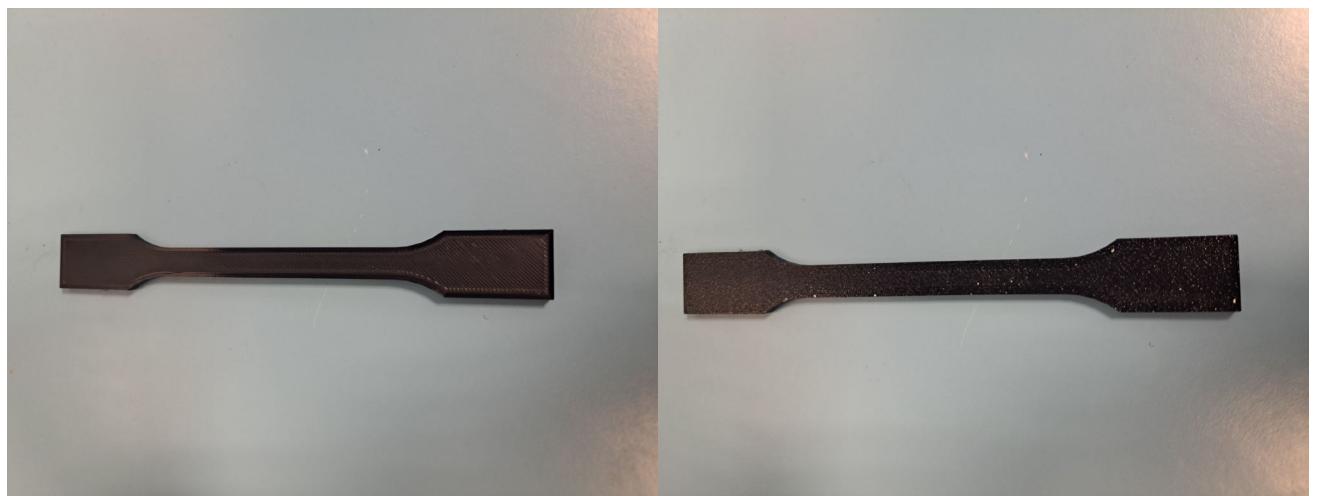


Figure 332: Before tensile test 3 | STB



Figure 333: After tensile test 3 | STB

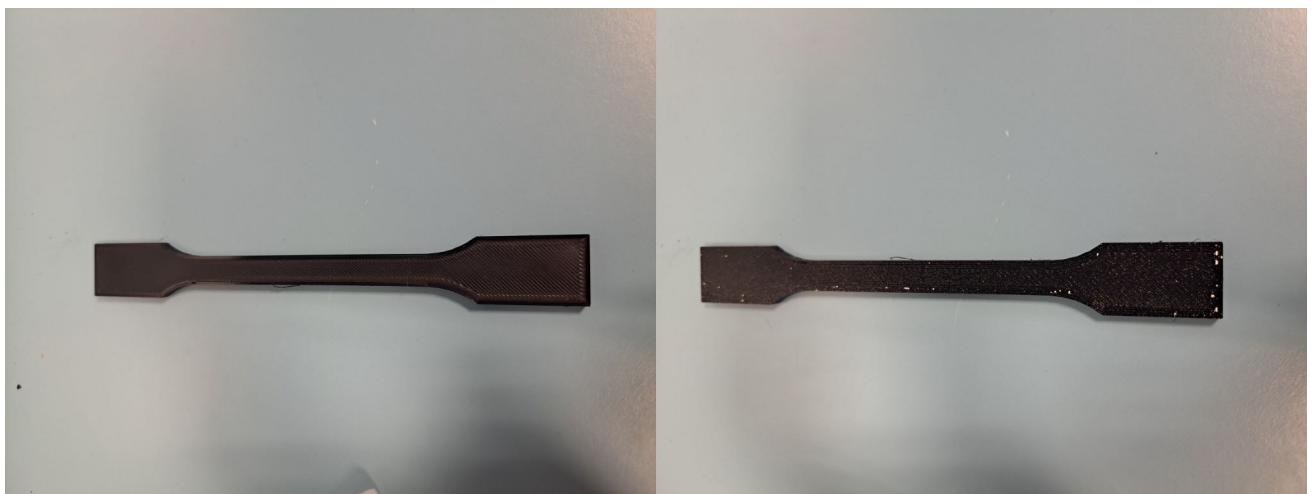


Figure 334: Before tensile test 4 | STB

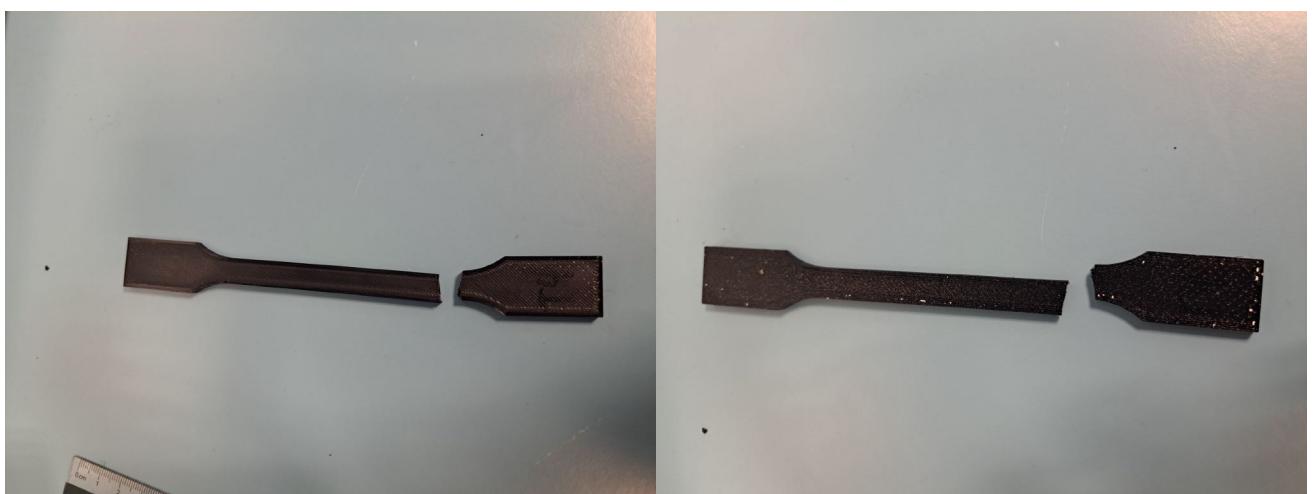


Figure 335: After tensile test 4 | STB

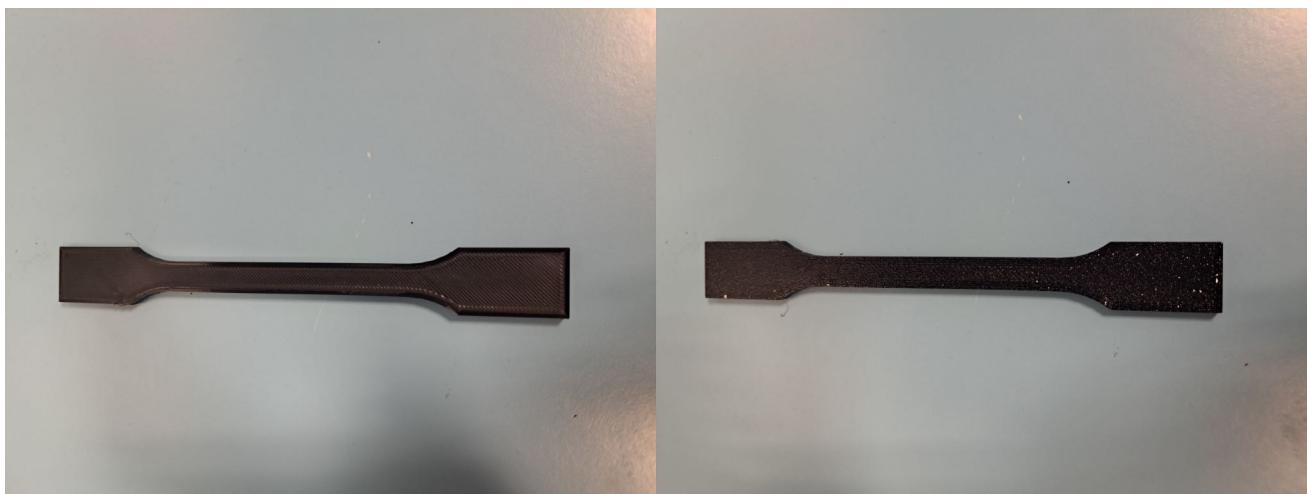


Figure 336: Before tensile test 5 | STB

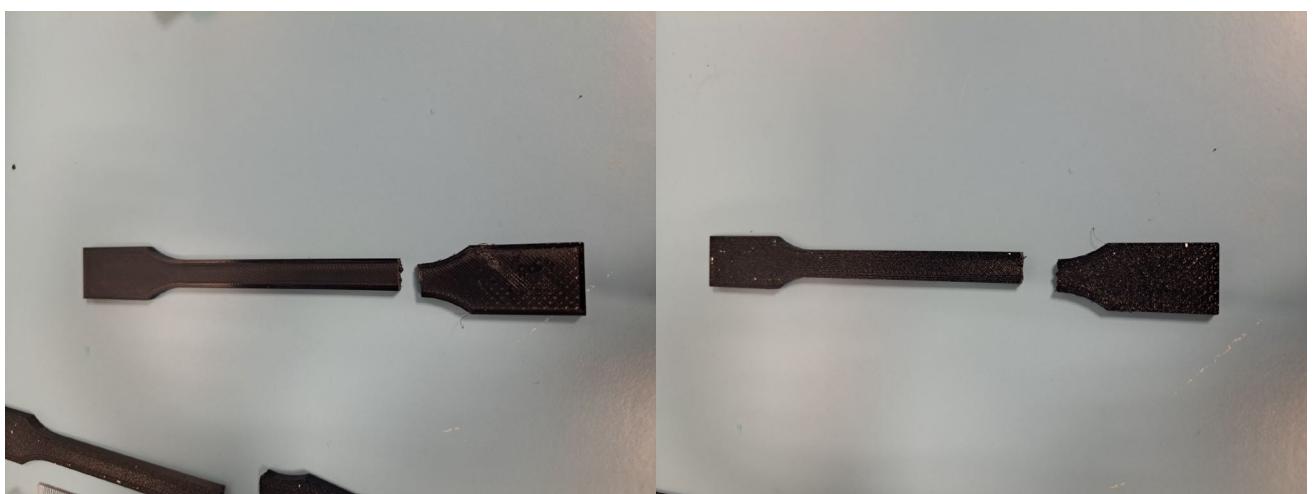


Figure 337: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the tensile test is shown in fig 338.

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
DATA	Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
		h	Test staff thickness [mm]	4,10	4,09	4,09	4,14	4,14	4,11
		b_1	Width at narrow portion [mm]	10,00	10,00	10,00	10,01	10,03	10,01
Value (Settings/test plan)	A		Cross-section [mm ²]	41,00	40,90	40,90	41,44	41,52	N/A
		L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
	N/A		Amount of test	5	5	5	5	5	N/A
	v		Test Speed [mm/min]	60	60	60	60	60	N/A
Proportional limit	σ_y		Nominal Stress [MPa]	12,931	12,244	11,717	10,878	10,880	11,730
		ϵ_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0
	σ_{true}		True stress [MPa]	12,931	12,244	11,717	10,879	10,880	11,730
		ϵ_{true}	True strain [%]	N/A	N/A	N/A	N/A	N/A	0
	E_t		Tensile modulus [MPa]	N/A	N/A	N/A	237146,52	N/A	47429,30
UTS	S_{ut}		Ultimate tensile strength [MPa]	36,834	38,194	40,997	377,624	39,484	106,627
		ϵ_{nom}	Nominal strain [%]	3,07	3,44	4,12	3,67	4,12	3,68
	σ_{true}		True stress [MPa]	37,964	39,506	42,687	39,005	41,112	40,055
		ϵ_{true}	True strain [%]	3,02	3,38	4,04	3,61	4,04	3,62
Fracture	σ_b		Stress at break [MPa]	18,907	0,168	39,344	6,834	38,918	20,834
		ϵ_b	Strain (engineering) at break [%]	3,12	3,53	4,53	3,72	4,26	3,83
	σ_{true}		True stress [MPa]	19,496	0,174	41,128	7,088	40,576	21,692
		ϵ_{true}	True strain [%]	3,07	3,47	4,44	3,65	4,17	3,76

Figure 338: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

16. April 2024

STB|

P.5 Material: PETG x Infill pattern: Cubic

Infill density: 75%

Preparation temp.: -38 to -40 Celsius

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008

-
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -38°C to -40°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between 20°C and 24°C before tensile test.

Pictures - before & after tensile test



Figure 339: Before tensile test 1 | STB



Figure 340: After tensile test 1 | STB



Figure 341: Before tensile test 2 | STB



Figure 342: After tensile test 2 | STB



Figure 343: Before tensile test 3 | STB



Figure 344: After tensile test 3 | STB



Figure 345: Before tensile test 4 | STB



Figure 346: After tensile test 4 | STB



Figure 347: Before tensile test 5 | STB



Figure 348: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the tensile test is shown in fig 349.

	A	B	C	D	E	F	G	H	I	J
1	DATA	Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values	
2		L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A	
3		h	Test staff thickness [mm]	4,05	4,08	4,05	4,08	4,06	4,06	
4		b_1	Width at narrow portion [mm]	9,99	9,96	10,00	9,98	10,00	9,99	
5		A	Cross-section [mm ²]	40,46	40,64	40,50	40,72	40,60	N/A	
6		L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A	
7		N/A	Amount of test	5	5	5	5	5	N/A	
8		v	Test Speed [mm/min]	60	60	60	60	60	N/A	
9		Proportional limit	σ_y Nominal Stress [MPa]	10,222	10,780	11,566	10,758	12,479	11,161	
10			ε_{nom} Nominal strain [%]	0,51	0,51	0,46	0,6	0,56	0,528	
11		UTS	σ_{true} True stress [MPa]	10,273	10,835	11,619	10,823	12,549	11,220	
12			ε_{true} True strain [%]	0,5	0,51	0,46	0,6	0,56	0,526	
13			E_t Tensile modulus [MPa]	2022,05	2113,32	2521,48	1790,31	2226,26	2134,68	
14		Fracture	S_ut Ultimate tensile strength [MPa]	42,776	38,803	41,910	37,161	38,234	39,777	
15			ε_{nom} Nominal strain [%]	5,41	4,36	5,04	4,13	4,00	4,59	
16			σ_{true} True stress [MPa]	45,090	40,494	44,024	38,695	39,763	41,613	
17			ε_{true} True strain [%]	5,27	4,27	4,92	4,05	3,92	4,49	
18			σ_b Stress at break [MPa]	41,613	38,296	39,805	5,271	14,628	27,923	
19			ε_b Strain (engineering) at break [%]	5,82	4,45	5,50	4,18	4,04	4,80	
20			σ_{true} True stress [MPa]	44,036	40,001	41,995	5,491	15,219	29,348	
21			ε_{true} True strain [%]	5,66	4,35	5,36	4,09	3,96	4,68	

Figure 349: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

18. April 2024

STB|

P.6 Material: PETG x Infill pattern: Cubic

Infill density: 75%

Preparation temp.: 20 to 24 Celsius

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008

-
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed room for 19 hours with the *preparation temperature* 20°C to 24°C. The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C.

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between 20°C and 24°C before tensile test.

Pictures - before & after tensile test



Figure 350: Before tensile test 1 | STB



Figure 351: After tensile test 1 | STB



Figure 352: Before tensile test 2 | STB



Figure 353: After tensile test 2 | STB

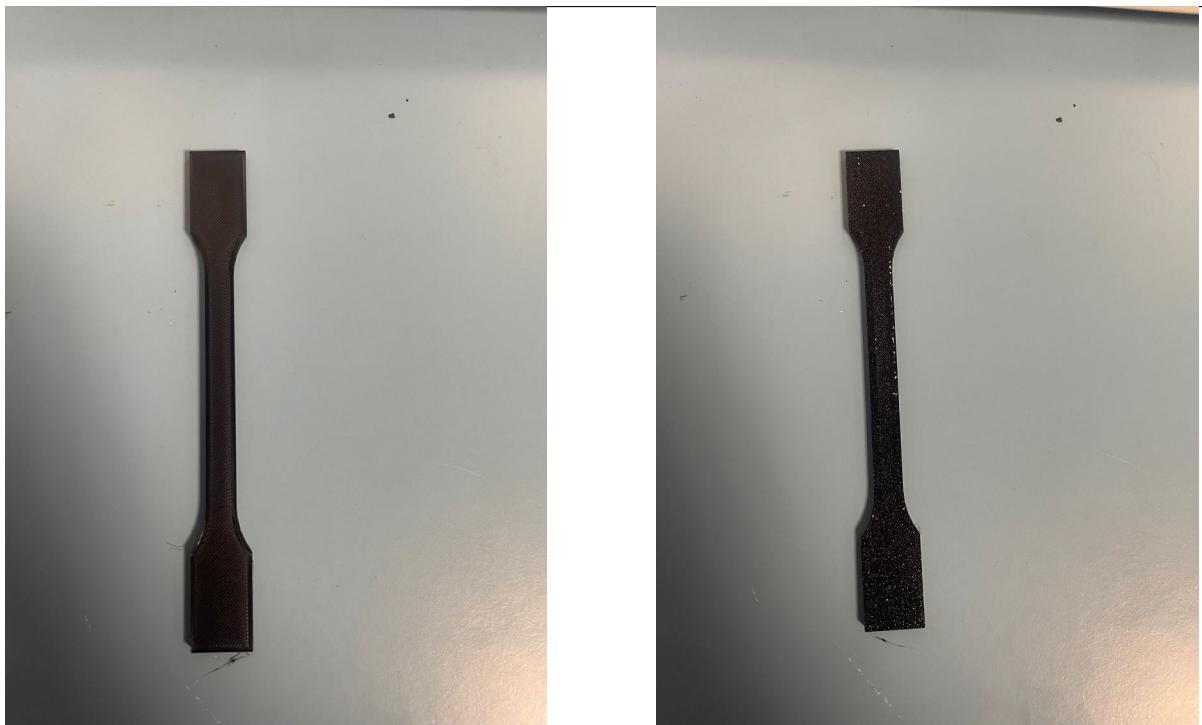


Figure 354: Before tensile test 3 | STB



Figure 355: After tensile test 3 | STB



Figure 356: Before tensile test 4 | STB



Figure 357: After tensile test 4 | STB



Figure 358: Before tensile test 5 | STB



Figure 359: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 360.

	A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values	
2	Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A	
3		h	Test staff thickness [mm]	4,15	4,15	4,14	4,14	4,15	4,15	
4		b_1	Width at narrow portion [mm]	10,00	9,99	10,01	10,01	10,01	10,00	
5	Value (Settings/test plan)	A	Cross-section [mm ²]	41,50	41,46	41,44	41,44	41,54	N/A	
6		L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A	
7		N/A	Amount of test	5	5	5	5	5	N/A	
8		v	Test Speed [mm/min]	60	60	60	60	60	N/A	
9	Proportional limit	σ_y	Nominal Stress [MPa]	10,556	11,394	10,783	11,185	11,111	11,006	
10		ϵ_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0	
11		σ_{true}	True stress [MPa]	10,555	11,394	10,783	11,186	11,111	11,006	
12		ϵ_{true}	True strain [%]	N/A	N/A	N/A	N/A	N/A	0	
13	UTS	E_t	Tensile modulus [MPa]	N/A	620994,13	N/A	1219300,63	N/A	368058,95	
14		S_{ut}	Ultimate tensile strength [MPa]	38,822	41,059	40,627	36,725	37,250	38,897	
15		ϵ_{nom}	Nominal strain [%]	3,57	4,31	4,17	3,16	3,34	3,71	
16		σ_{true}	True stress [MPa]	40,207	42,829	42,320	37,887	38,495	40,348	
17	Fracture	ϵ_{true}	True strain [%]	3,51	4,22	4,08	3,11	3,29	3,64	
18		σ_b	Stress at break [MPa]	15,349	35,032	39,799	29,205	28,686	29,614	
19		ϵ_b	Strain (engineering) at break [%]	3,61	4,95	4,35	3,21	3,39	3,90	
20		σ_{true}	True stress [MPa]	15,904	36,767	41,530	30,142	29,658	30,800	
21		ϵ_{true}	True strain [%]	3,55	4,83	4,26	3,16	3,33	3,83	

Figure 360: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

18. April 2024

**P.7 Material: PETG x Infill pattern: Gyroid
Infill density: 50%
Preparation temp.: -18 to -20 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -18°C to -20°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test



Figure 361: Before tensile test 1 | STB



Figure 362: After tensile test 1 | STB

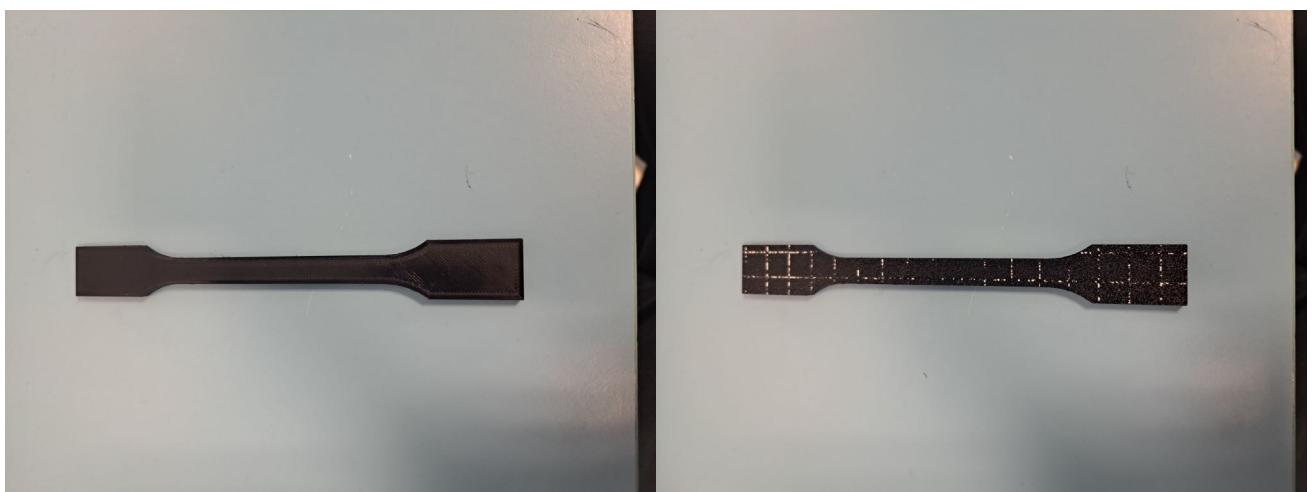


Figure 363: Before tensile test 2 | STB

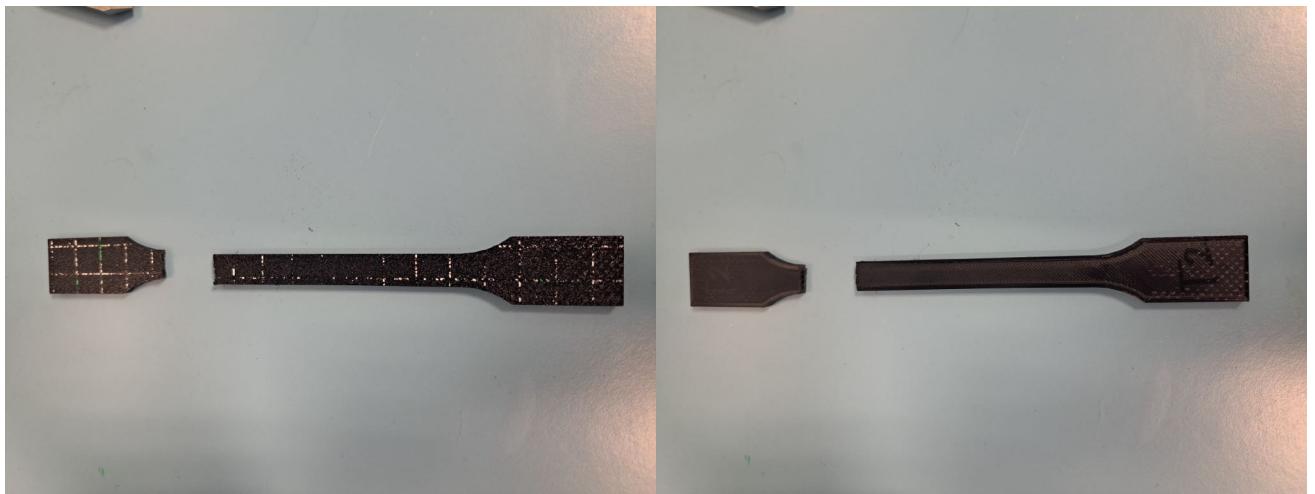


Figure 364: After tensile test 2 | STB

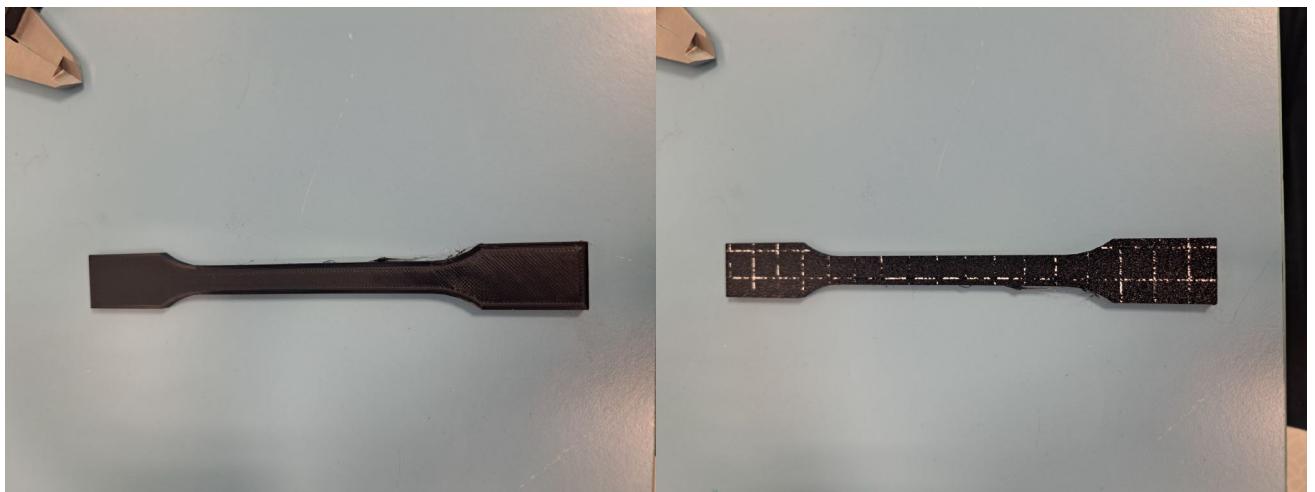


Figure 365: Before tensile test 3 | STB

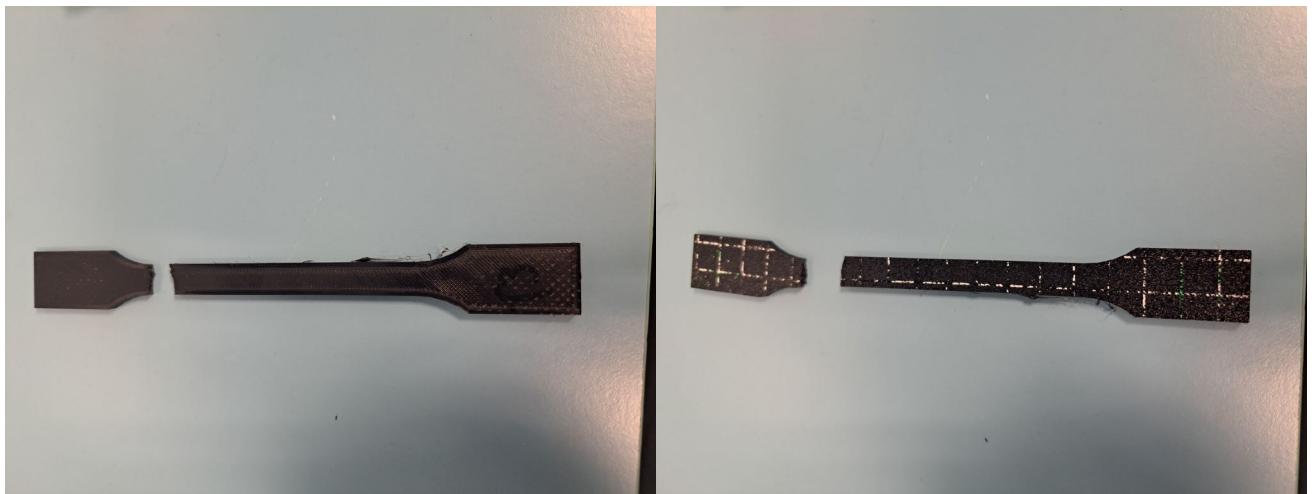


Figure 366: After tensile test 3 | STB

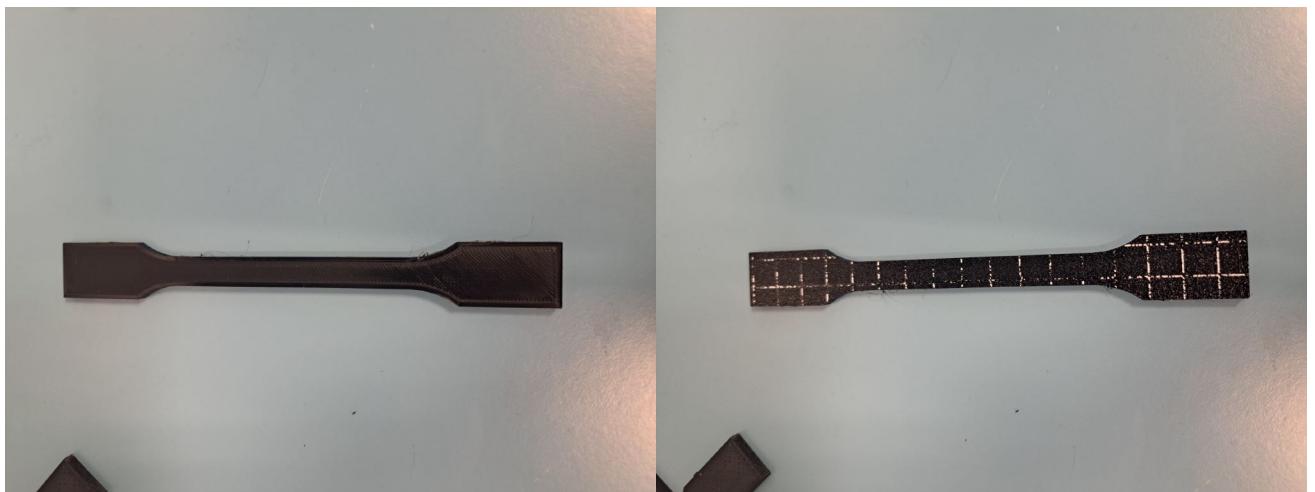


Figure 367: Before tensile test 4 | STB

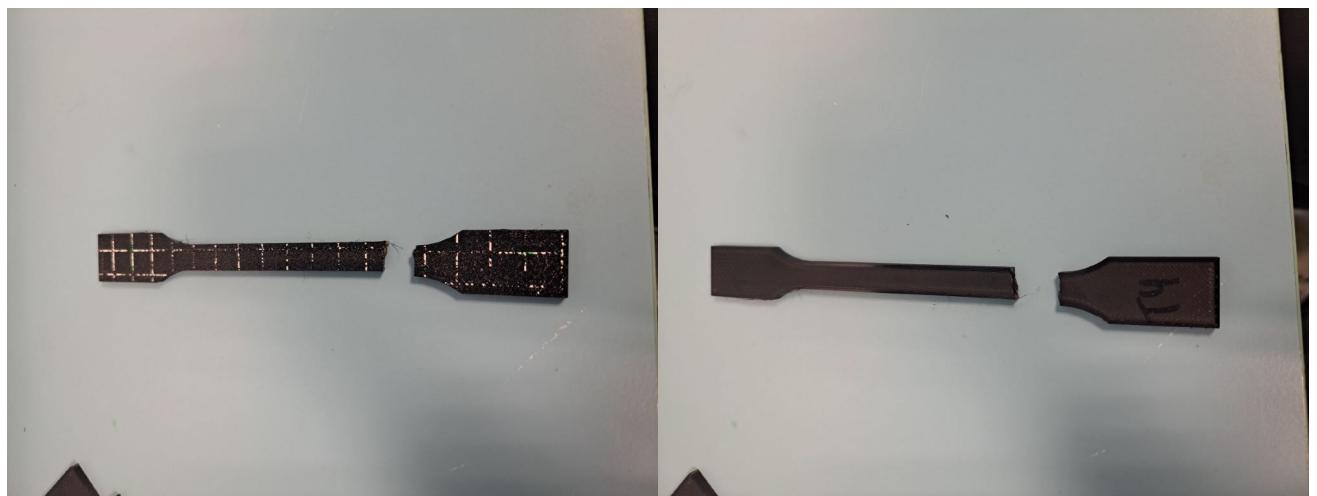


Figure 368: After tensile test 4 | STB

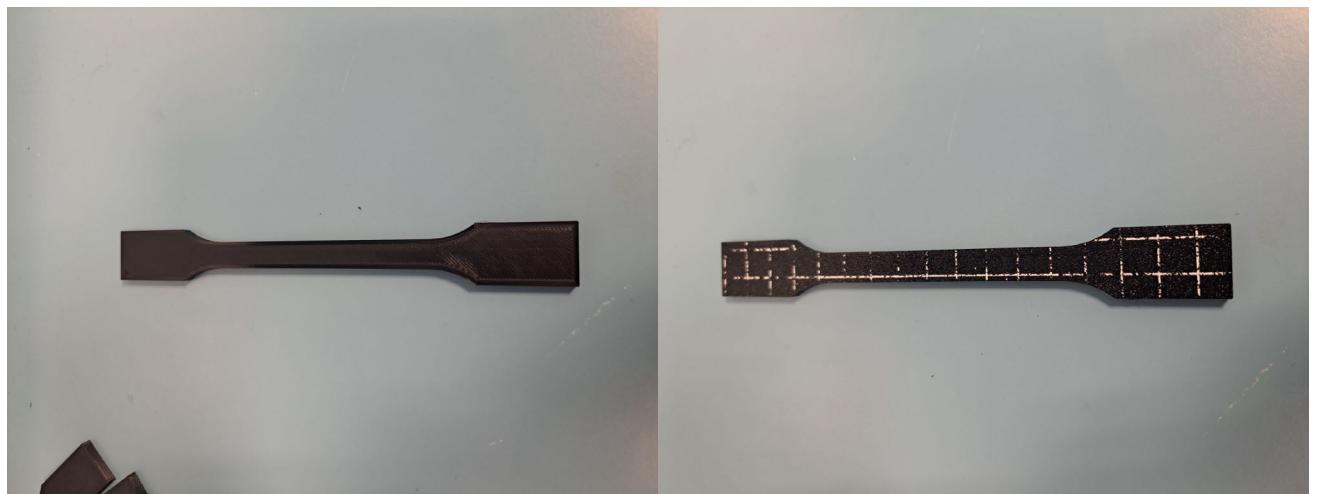


Figure 369: Before tensile test 5 | STB



Figure 370: After tensile test 5 | STB

STB|

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 371.

A	B	C	D	E	F	G	H	I	J
	Units	Description		Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
Initial values	L_0	Chosen start length [mm]		105,00	105,00	105,00	105,00	105,00	N/A
	h	Test staff thickness [mm]		4,10	4,10	4,10	4,10	4,10	4,10
	b_1	Width at narrow portion [mm]		10,01	10,05	10,00	10,00	10,00	10,01
Value (Settings/test plan)	A	Cross-section [mm^2]		41,04	41,21	41,00	41,00	41,00	N/A
	L	Gripping distance		105,00	105,00	105,00	105,00	105,00	N/A
	N/A	Amount of test		5	5	5	5	5	N/A
Proportional limit	v	Test Speed [mm/min]		60	60	60	60	60	N/A
	σ_y	Nominal Stress [MPa]		11,939	10,203	8,796	6,286	14,150	10,275
	ϵ_{nom}	Nominal strain [%]		1,10	0,27	0,59	0,65	1,22	0,766
UTS	σ_{true}	True stress [MPa]		12,071	10,231	8,848	6,327	14,323	10,360
	ϵ_{true}	True strain [%]		1,09	0,27	0,59	0,65	1,21	0,762
	E_t	Tensile modulus [MPa]		1085,39	3732,85	1482,49	970,69	1160,76	1686,44
Fracture	S_ut	Ultimate tensile strength [MPa]		39,209	36,817	34,993	33,320	35,085	35,881
	ϵ_{nom}	Nominal strain [%]		6,67	4,89	6,97	5,98	4,36	5,77
	σ_{true}	True stress [MPa]		41,824	38,617	37,433	35,312	36,594	37,956
	ϵ_{true}	True strain [%]		6,46	4,77	6,74	5,81	4,27	5,61
	σ_b	Stress at break [MPa]		35,961	36,722	3,561	31,862	33,44	28,309
	ϵ_b	Strain (engineering) at break [%]		7,53	4,94	7,02	6,17	4,41	6,01
	σ_{true}	True stress [MPa]		38,668	38,535	3,812	33,827	34,914	29,951
	ϵ_{true}	True strain [%]		7,26	4,82	6,79	5,99	4,31	5,83

Figure 371: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

STB|

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens from tensile test 1 was rejected and no data was gained. No time to re-print and test again.

STB|

Date of measurement

v1 16. April 2024 - Rejected and replaced

v2 19. April 2024

STB|

P.8 Material: PETG x Infill pattern: Gyroid

Infill density: 50%

Preparation temp.: -38 to -40 Celsius

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -38°C to -40°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between $20^{\circ}C$ and $24^{\circ}C$.

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test

Figure 372: Before tensile test 1 | STB



Figure 373: After tensile test 1 | STB



Figure 374: Before tensile test 2 | STB

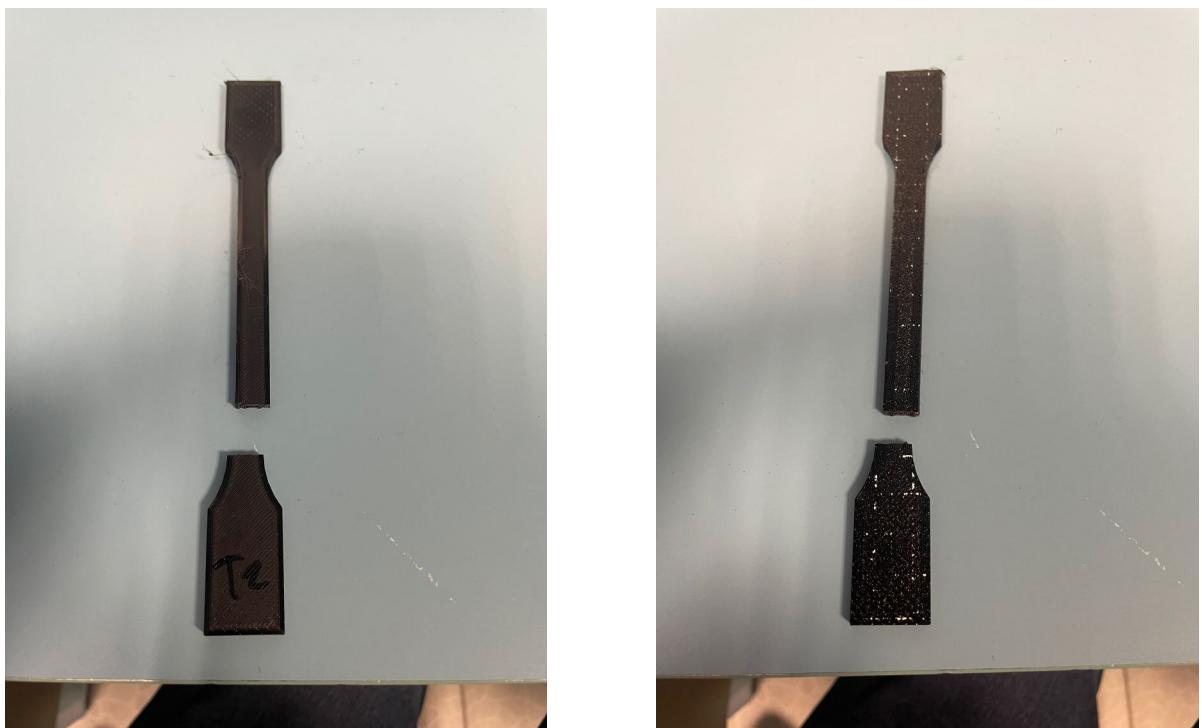


Figure 375: After tensile test 2 | STB



Figure 376: Before tensile test 3 | STB



Figure 377: After tensile test 3 | STB

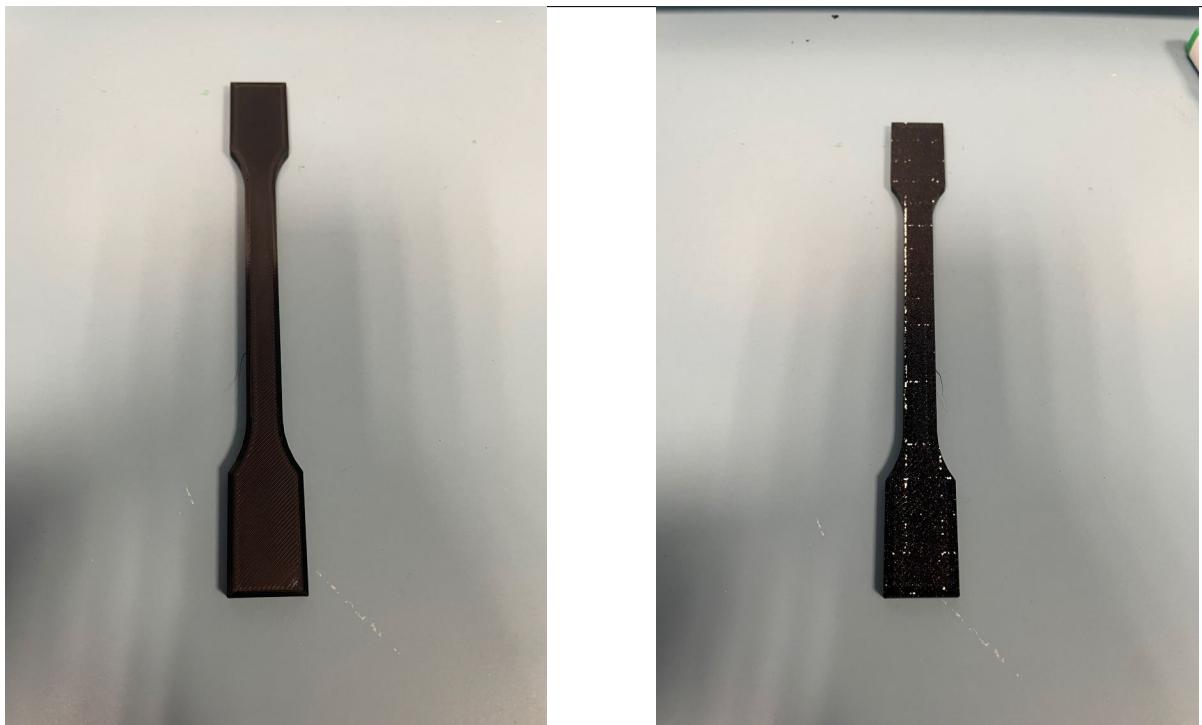


Figure 378: Before tensile test 4 | STB



Figure 379: After tensile test 4 | STB



Figure 380: Before tensile test 5 | STB

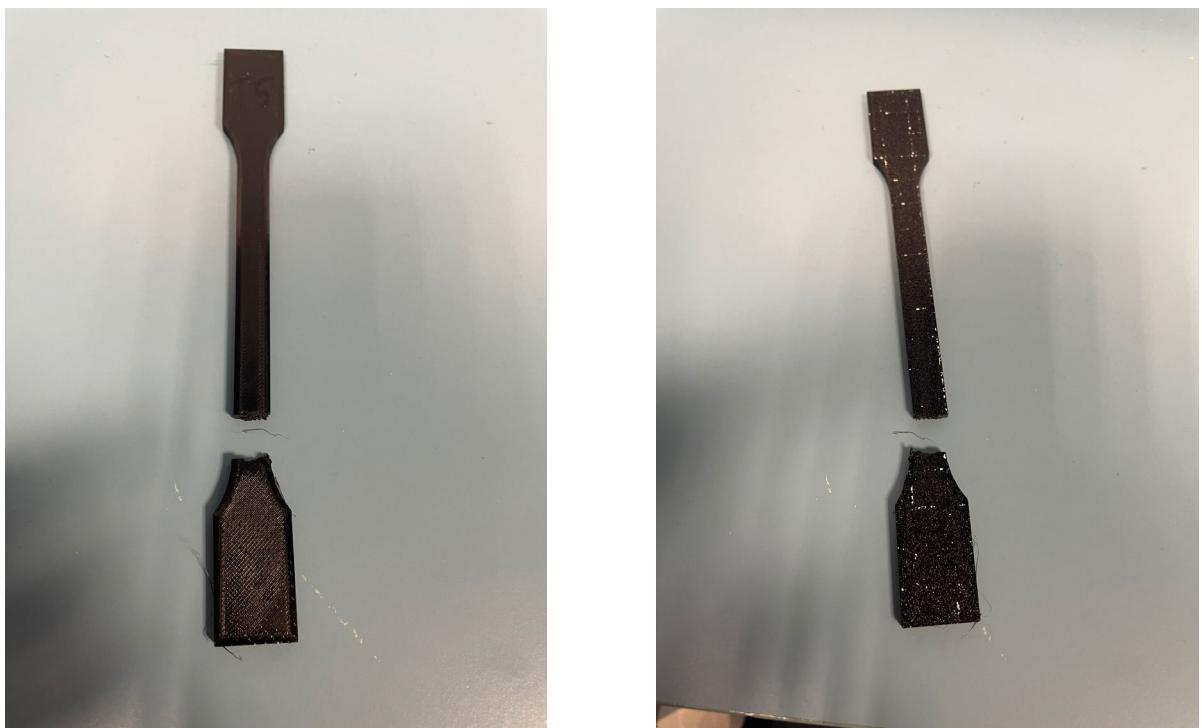


Figure 381: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 382.

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
2	Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
		h	Test staff thickness [mm]	4,07	4,07	4,05	4,06	4,08	4,07
		b_1	Width at narrow portion [mm]	10,00	10,00	10,00	10,00	9,99	10,00
5	Value (Settings/test plan)	A	Cross-section [mm ²]	40,70	40,70	40,50	40,60	40,76	N/A
		L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
		N/A	Amount of test	5	5	5	5	5	N/A
		v	Test Speed [mm/min]	60	60	60	60	60	N/A
9	Proportional limit	σ_y	Nominal Stress [MPa]	11,654	12,256	11,808	11,707	11,854	11,856
		ε_{nom}	Nominal strain [%]	0,58	0,61	0,59	0,41	0,45	0,528
		σ_{true}	True stress [MPa]	11,722	12,331	11,878	11,755	11,907	11,919
		$\varepsilon_{\text{true}}$	True strain [%]	0,58	0,61	0,59	0,41	0,45	0,528
14	UTS	E_t	Tensile modulus [MPa]	2000,46	1999,86	2007,98	2867,53	2631,43	2301,45
		S_{ut}	Ultimate tensile strenght [MPa]	35,347	36,864	39,732	38,814	34,815	37,114
		ε_{nom}	Nominal strain [%]	3,93	4,19	4,85	4,40	3,66	4,21
		σ_{true}	True stress [MPa]	36,736	38,408	41,661	40,521	36,089	38,683
18	Fracture	$\varepsilon_{\text{true}}$	True strain [%]	3,85	4,10	4,74	4,30	3,59	4,12
		σ_b	Stress at break [MPa]	12,738	36,861	36,901	38,717	34,431	31,930
		ε_b	Strain (engineering) at break [%]	3,98	4,23	5,36	4,44	3,80	4,36
		σ_{true}	True stress [MPa]	13,244	38,375	38,878	40,437	35,738	33,334
21		$\varepsilon_{\text{true}}$	True strain [%]	3,90	4,15	5,22	4,35	3,73	4,27

Figure 382: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

18. April 2024

**P.9 Material: PETG x Infill pattern: Gyroid
Infill density: 50%
Preparation temp.: 20 to 24 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed room for 19 hours with the *preparation temperature* 20°C to 24°C. The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C.

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test



Figure 383: Before tensile test 1 | STB



Figure 384: After tensile test 1 | STB



Figure 385: Before tensile test 2 | STB



Figure 386: After tensile test 2 | STB



Figure 387: Before tensile test 3 | STB

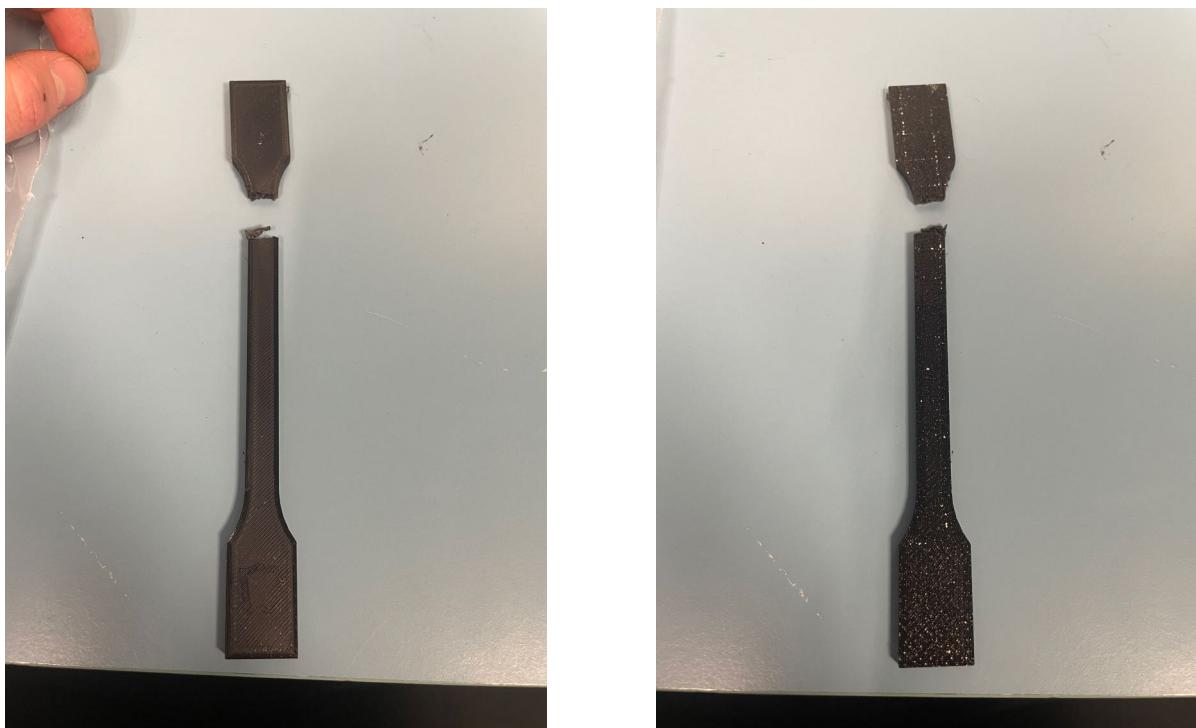


Figure 388: After tensile test 3 | STB



Figure 389: Before tensile test 4 | STB



Figure 390: After tensile test 4 | STB



Figure 391: Before tensile test 5 | STB



Figure 392: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 393

	A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values	
2		Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
3			h	Test staff thickness [mm]	4,08	4,08	4,10	4,10	4,10	4,09
4			b_1	Width at narrow portion [mm]	10,01	10,01	9,98	10,04	9,99	10,01
5		Value (Settings/test plan)	A	Cross-section [mm ²]	40,84	40,84	40,92	41,16	40,96	N/A
6			L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
7			N/A	Amount of test	5	5	5	5	5	N/A
8			v	Test Speed [mm/min]	60	60	60	60	60	N/A
9		Proportional limit	σ_y	Nominal Stress [MPa]	9,070	9,574	9,820	9,190	9,762	9,483
10			ϵ_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0
11			σ_{true}	True stress [MPa]	9,071	9,574	9,820	9,190	9,762	9,483
12			ϵ_{true}	True strain [%]	N/A	N/A	N/A	N/A	N/A	0
13		UTS	E_t	Tensile modulus [MPa]	197738,34	1043420,00	535145,375	333885,88	1064130,50	527834,94
14			S_{ut}	Ultimate tensile strength [MPa]	34,002	34,674	34,536	33,830	37,564	34,921
15			ϵ_{nom}	Nominal strain [%]	3,85	3,90	3,58	3,81	4,63	3,95
16			σ_{true}	True stress [MPa]	35,312	36,025	35,772	35,118	39,304	36,306
17		Fracture	ϵ_{true}	True strain [%]	3,78	3,82	3,51	3,74	4,53	3,88
18			σ_b	Stress at break [MPa]	33,378	34,65	33,291	0,27	35,913	27,500
19			ϵ_b	Strain (engineering) at break [%]	3,94	3,94	3,71	3,90	5,09	4,12
20			σ_{true}	True stress [MPa]	34,694	36,016	34,528	0,223	37,742	28,641
21			ϵ_{true}	True strain [%]	3,87	3,87	3,65	3,83	4,97	4,04

Figure 393: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

17. April 2024

**P.10 Material: PETG x Infill pattern: Gyroid
Infill density: 75%
Preparation temp.: -18 to -20 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -18°C to -20°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test

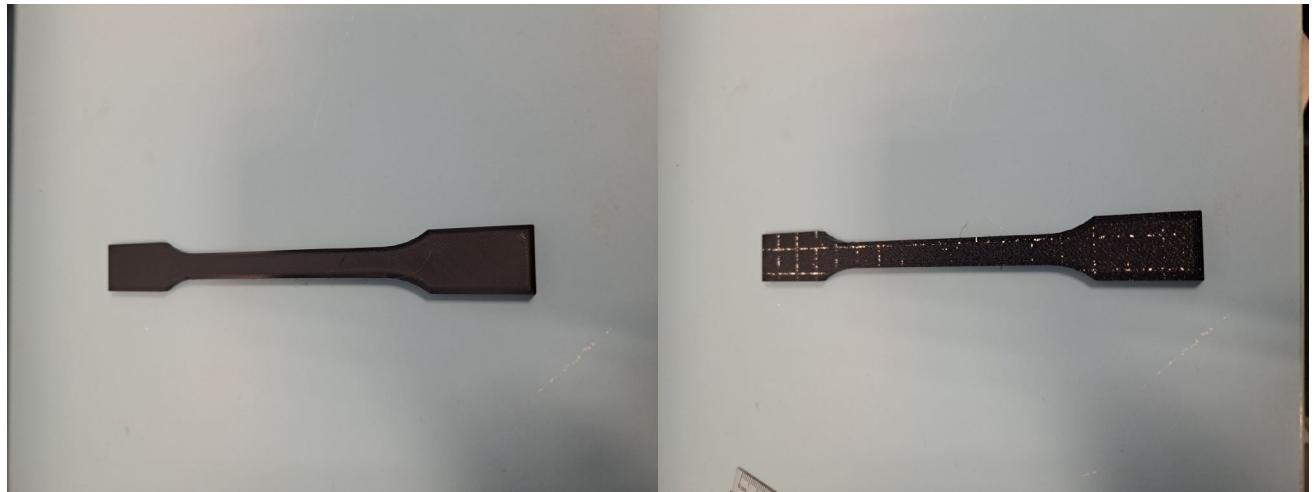


Figure 394: Before tensile test 1 | STB

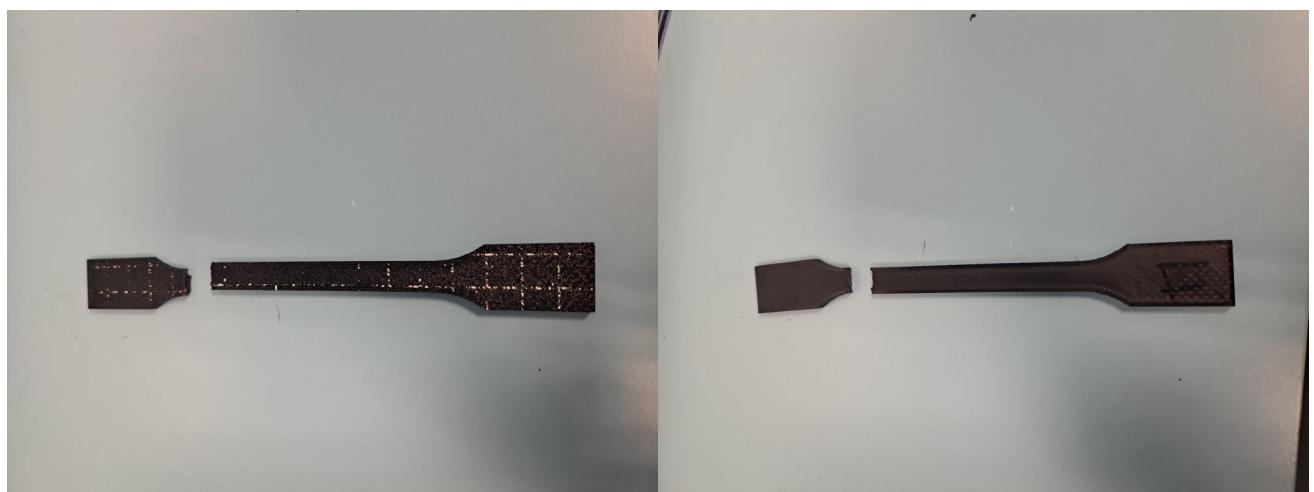


Figure 395: After tensile test 1 | STB



Figure 396: Before tensile test 2 | STB



Figure 397: After tensile test 2 | STB



Figure 398: Before tensile test 3 | STB

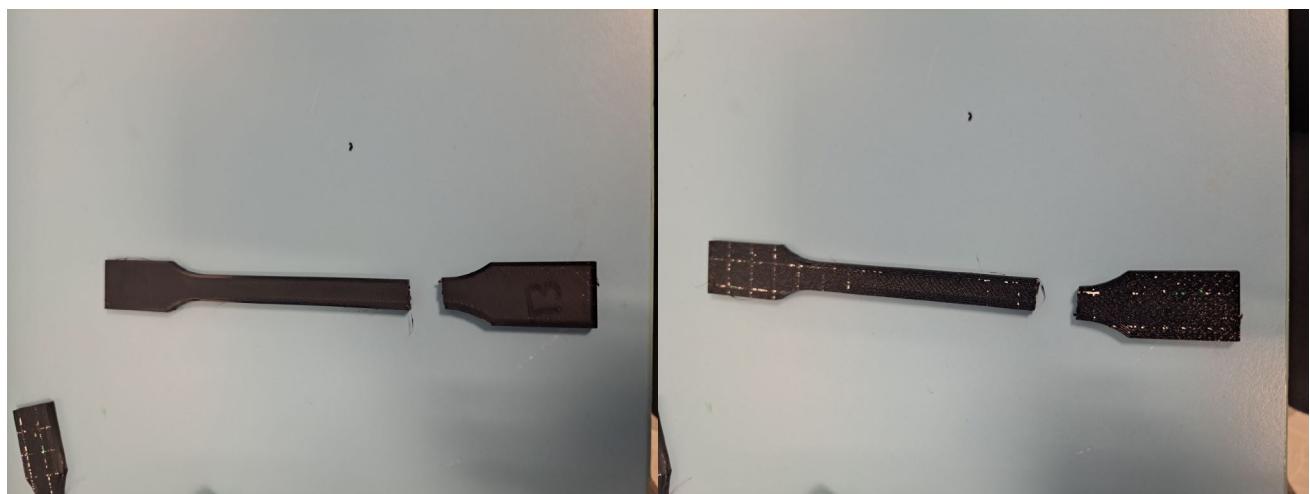


Figure 399: After tensile test 3 | STB

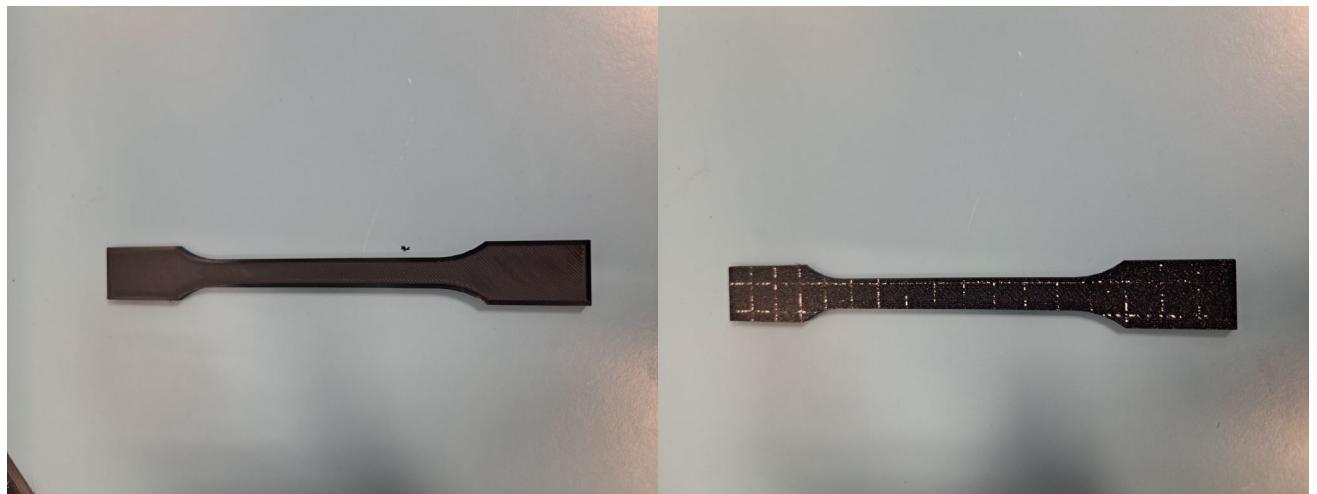


Figure 400: Before tensile test 4 | STB

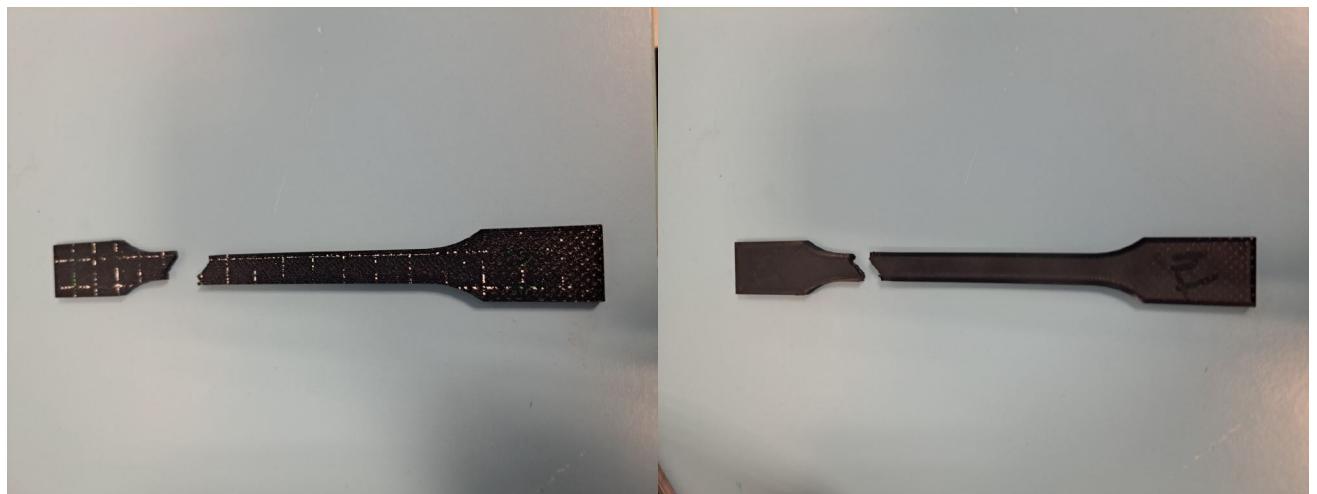


Figure 401: After tensile test 4 | STB

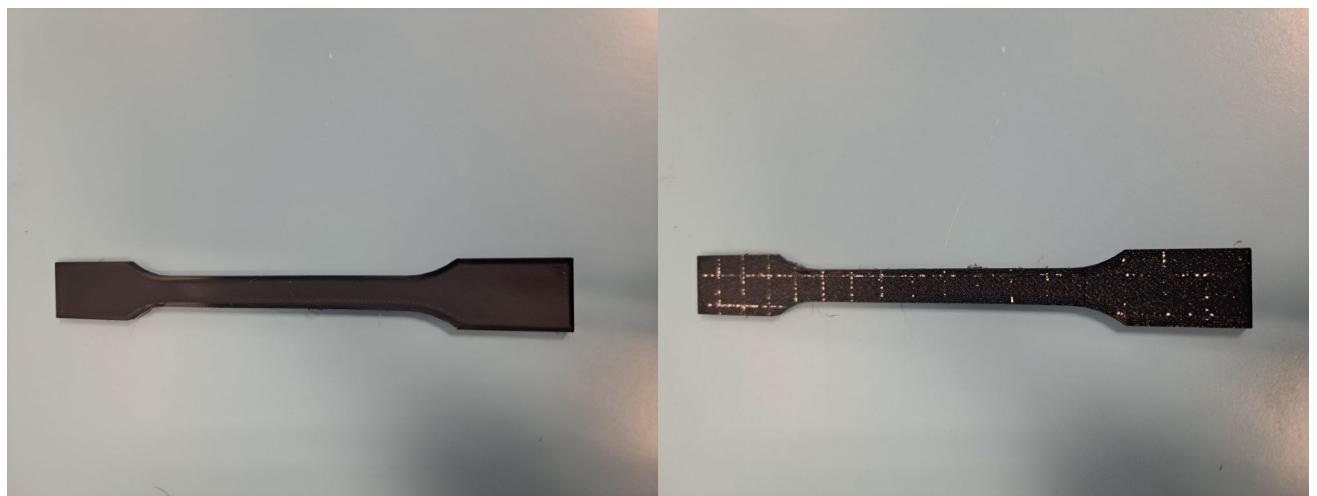


Figure 402: Before tensile test 5 | STB

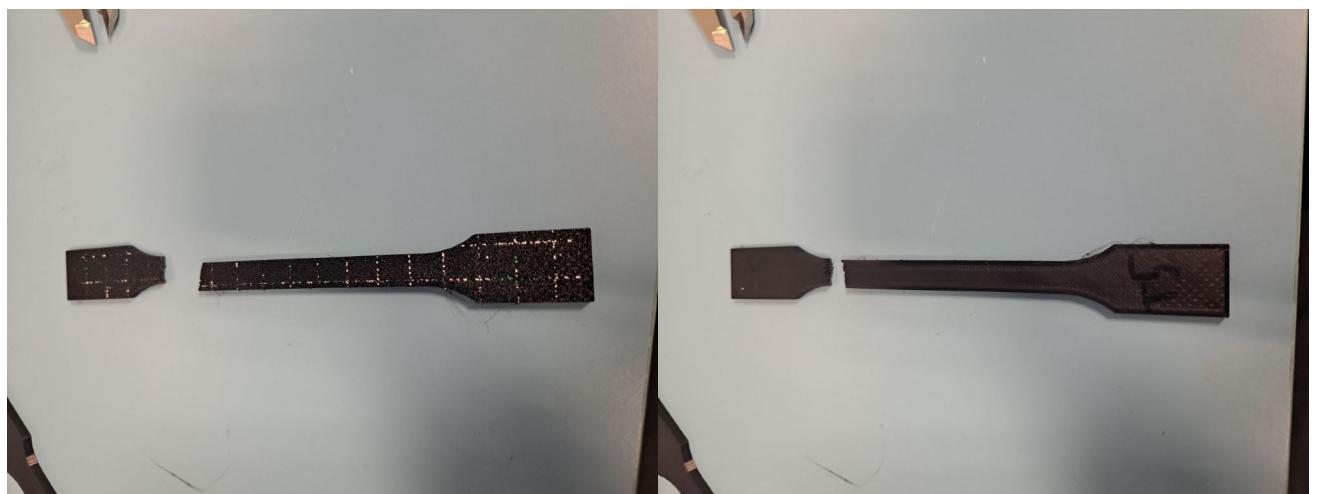


Figure 403: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 404.

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
2	Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
		h	Test staff thickness [mm]	4,08	4,07	4,13	4,12	4,20	4,12
		b_1	Width at narrow portion [mm]	10,10	10,02	10,00	10,00	10,00	10,02
5	Value (Settings/test plan)	A	Cross-section [mm ²]	41,21	40,78	41,30	41,20	42,00	N/A
		L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
		N/A	Amount of test	5	5	5	5	5	N/A
9	Value (Settings/test plan)	v	Test Speed [mm/min]	60	60	60	60	60	N/A
		σ_y	Nominal Stress [MPa]	11,629	11,631	11,360	12,583	10,920	11,625
		ϵ_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0
11	Proportional limit	σ_{true}	True stress [MPa]	11,629	11,631	11,366	12,583	10,920	11,626
		ϵ_{true}	True strain [%]	N/A	N/A	N/A	N/A	N/A	0
		E_t	Tensile modulus [MPa]	N/A	633924,06	N/A	N/A	N/A	126784,81
14	UTS	S_{ut}	Ultimate tensile strenght [MPa]	36,053	33,907	35,000	41,055	34,767	36,156
		ϵ_{nom}	Nominal strain [%]	3,21	2,71	2,93	4,21	3,07	3,23
		σ_{true}	True stress [MPa]	37,209	34,825	36,026	42,785	35,833	37,336
		ϵ_{true}	True strain [%]	3,16	2,67	2,89	4,13	3,02	3,17
18	Fracture	σ_b	Stress at break [MPa]	0,761	2,643	1,068	39,033	7,933	10,288
		ϵ_b	Strain (engineering) at break [%]	3,26	2,76	3,03	4,76	3,12	3,39
		σ_{true}	True stress [MPa]	0,786	2,716	1,100	40,893	8,181	10,735
		ϵ_{true}	True strain [%]	3,21	2,72	2,98	4,65	3,07	3,33

Figure 404: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

16. April 2024

**P.11 Material: PETG x Infill pattern: Gyroid
Infill density: 75%
Preparation temp.: -38 to -40 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -38°C to -40°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test



Figure 405: Before tensile test 1 | STB



Figure 406: After tensile test 1 | STB



Figure 407: Before tensile test 2 | STB



Figure 408: After tensile test 2 | STB



Figure 409: Before tensile test 3 | STB



Figure 410: After tensile test 3 | STB

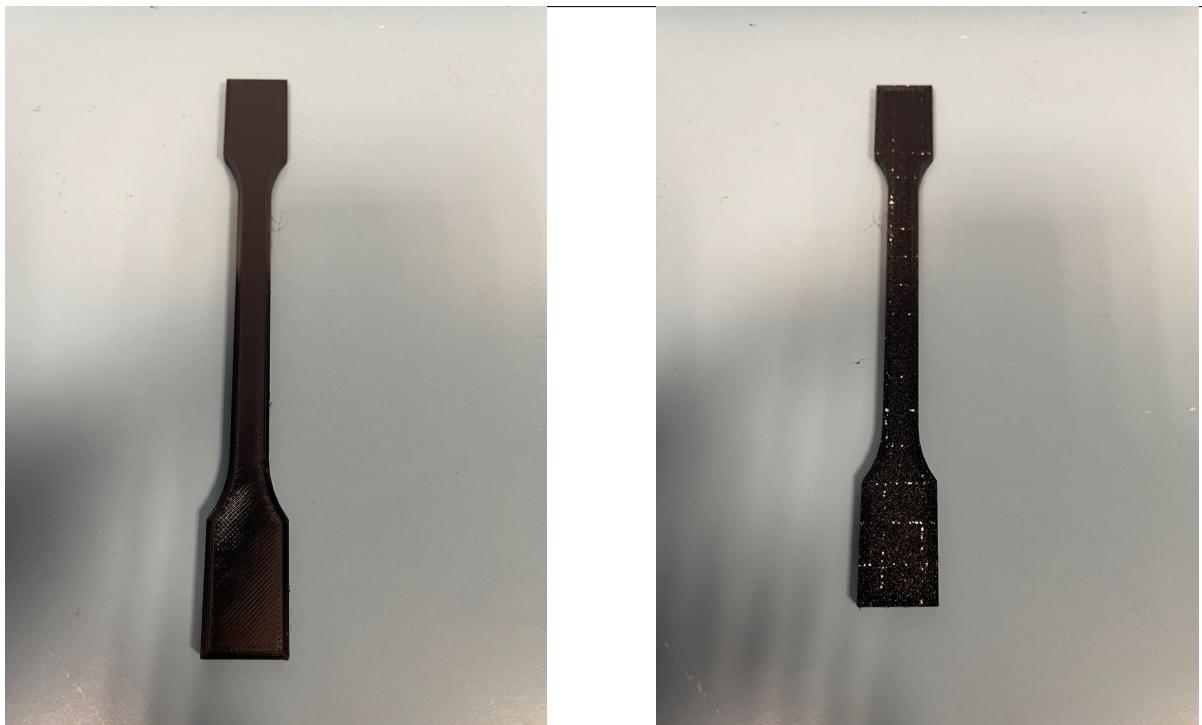


Figure 411: Before tensile test 4 | STB



Figure 412: After tensile test 4 | STB



Figure 413: Before tensile test 5 | STB



Figure 414: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 415.

	A	B	C	D	E	F	G	H	I	J
1			Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
2			L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
3			h	Test staff thickness [mm]	4,10	4,13	4,10	4,11	4,10	4,11
4			b_1	Width at narrow portion [mm]	9,99	9,97	10,00	9,99	10,00	9,99
5			A	Cross-section [mm^2]	40,96	41,18	41,00	41,06	41,00	N/A
6			L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
7			N/A	Amount of test	5	5	5	5	5	N/A
8			v	Test Speed [mm/min]	60	60	60	60	60	N/A
9			σ_y	Nominal Stress [MPa]	12,298	12,019	12,381	11,600	11,664	11,992
10			ε_{nom}	Nominal strain [%]	0,35	0,63	0,61	0,65	0,2	0,488
11			σ_{true}	True stress [MPa]	12,341	12,095	12,457	11,676	11,688	12,051
12			$\varepsilon_{\text{true}}$	True strain [%]	0,35	0,63	0,6	0,65	0,2	0,486
13			E_t	Tensile modulus [MPa]	3527,63	1909,74	2041,72	1775,83	5753,03	3001,59
14			S_ut	Ultimate tensile strength [MPa]	41,129	36,843	38,602	35,683	36,595	37,770
15			ε_{nom}	Nominal strain [%]	4,66	3,84	4,04	3,77	3,55	3,97
16			σ_{true}	True stress [MPa]	43,044	38,256	40,164	37,027	37,893	39,277
17			$\varepsilon_{\text{true}}$	True strain [%]	4,55	3,76	3,96	3,70	3,49	3,89
18			σ_b	Stress at break [MPa]	38,330	36,034	37,957	35,349	36,045	36,743
19			ε_b	Strain (engineering) at break [%]	5,25	3,93	4,14	3,81	3,68	4,16
20			σ_{true}	True stress [MPa]	40,344	37,449	39,527	36,697	37,373	38,278
21			$\varepsilon_{\text{true}}$	True strain [%]	5,12	3,85	4,05	3,74	3,62	4,08

Figure 415: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

18. April 2024

**P.12 Material: PETG x Infill pattern: Gyroid
Infill density: 75%
Preparation temp.: 20 to 24 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PETG - read section 5.2.2 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed room for 19 hours with the *preparation temperature* 20°C to 24°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test



Figure 416: Before tensile test 1 | STB



Figure 417: After tensile test 1 | STB

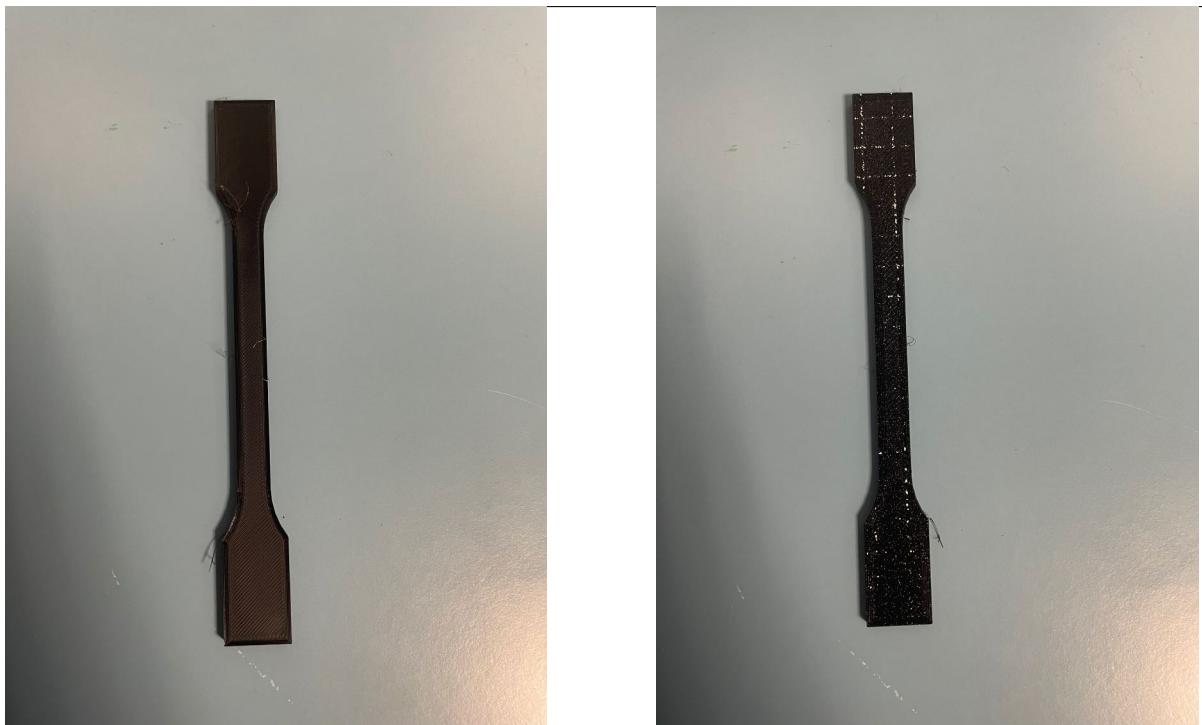


Figure 418: Before tensile test 2 | STB



Figure 419: After tensile test 2 | STB



Figure 420: Before tensile test 3 | STB



Figure 421: After tensile test 3 | STB

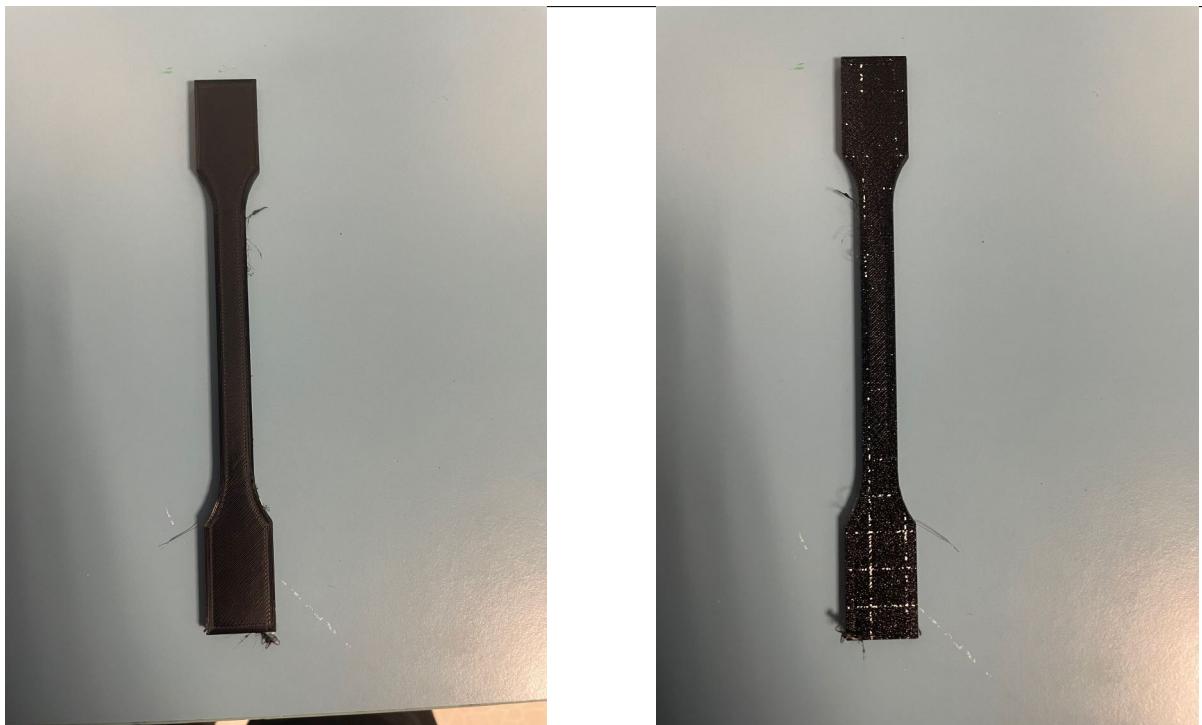


Figure 422: Before tensile test 4 | STB



Figure 423: After tensile test 4 | STB

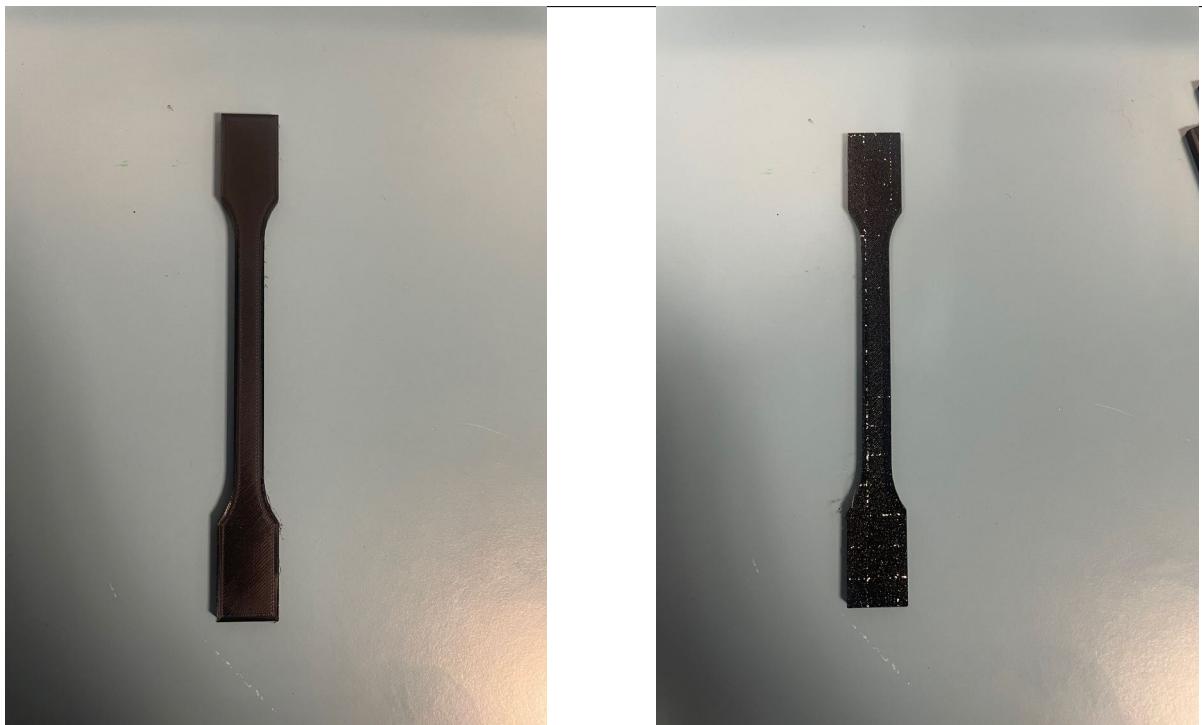


Figure 424: Before tensile test 5 | STB

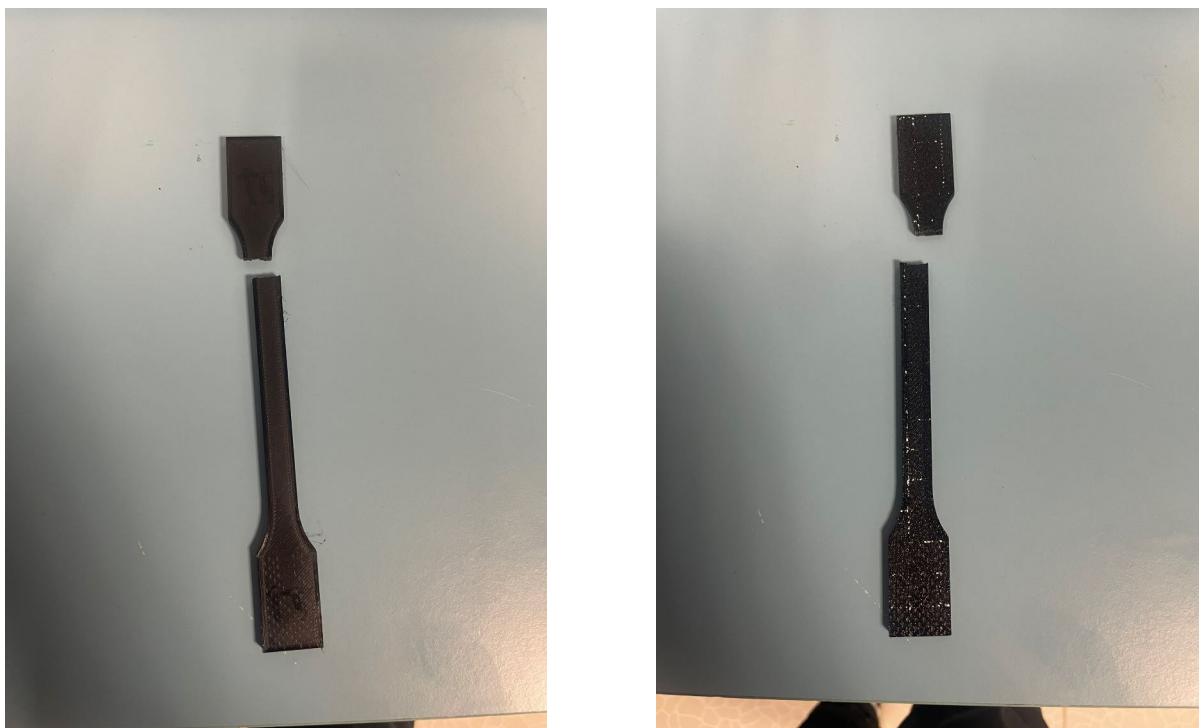


Figure 425: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 426.

	A	B	C	D	E	F	G	H	I	J
1	DATA	Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values	
2		Initial values	L_0 Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A	
3			h Test staff thickness [mm]	4,10	4,08	4,10	4,12	4,09	4,10	
4			b_1 Width at narrow portion [mm]	9,98	10,01	9,98	10,00	10,00	9,99	
5		Value (Settings/test plan)	A Cross-section [mm ²]	40,92	40,84	40,92	41,20	40,90	N/A	
6			L Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A	
7			N/A Amount of test	5	5	5	5	5	N/A	
8			v Test Speed [mm/min]	60	60	60	60	60	N/A	
9		Proportional limit	σ_y Nominal Stress [MPa]	10,850	10,630	10,730	11,917	11,429	11,111	
10			ε_nom Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0	
11			σ_true True stress [MPa]	10,850	10,630	10,730	11,917	11,429	11,111	
12			ε_true True strain [%]	N/A	N/A	N/A	N/A	N/A	0	
13			E_t Tensile modulus [MPa]	591270,38	1158760,13	N/A	649445,63	N/A	479895,23	
14		UTS	S_ut Ultimate tensile strenght [MPa]	38,608	39,209	41,793	35,870	37,858	38,668	
15			ε_nom Nominal strain [%]	3,62	3,67	4,49	3,02	3,30	3,62	
16			σ_true True stress [MPa]	40,006	40,647	43,670	36,955	39,108	40,077	
17			ε_true True strain [%]	3,56	3,60	4,39	2,98	3,25	3,56	
18		Fracture	σ_b Stress at break [MPa]	6,946	38,777	38,297	30,542	1,677	23,248	
19			ε_b Strain (engineering) at break [%]	3,67	3,76	5,09	3,07	3,35	3,79	
20			σ_true True stress [MPa]	7,201	40,234	40,244	31,48	1,733	24,178	
21			ε_true True strain [%]	3,60	3,69	4,96	3,02	3,30	3,71	

Figure 426: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

17. April 2024

P.13 Material: PLA x Infill pattern: Cubic
Infill density: 50%
Preparation temp.: -18 to -20 Celsius

The documentation and testing is based on sections/requirements from standards:

-
- NS-EN ISO 527-1:2019
 - NS-EN ISO 291:2008
 - NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -18°C to -20°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between 20°C and 24°C before tensile test.

Pictures - before & after tensile test

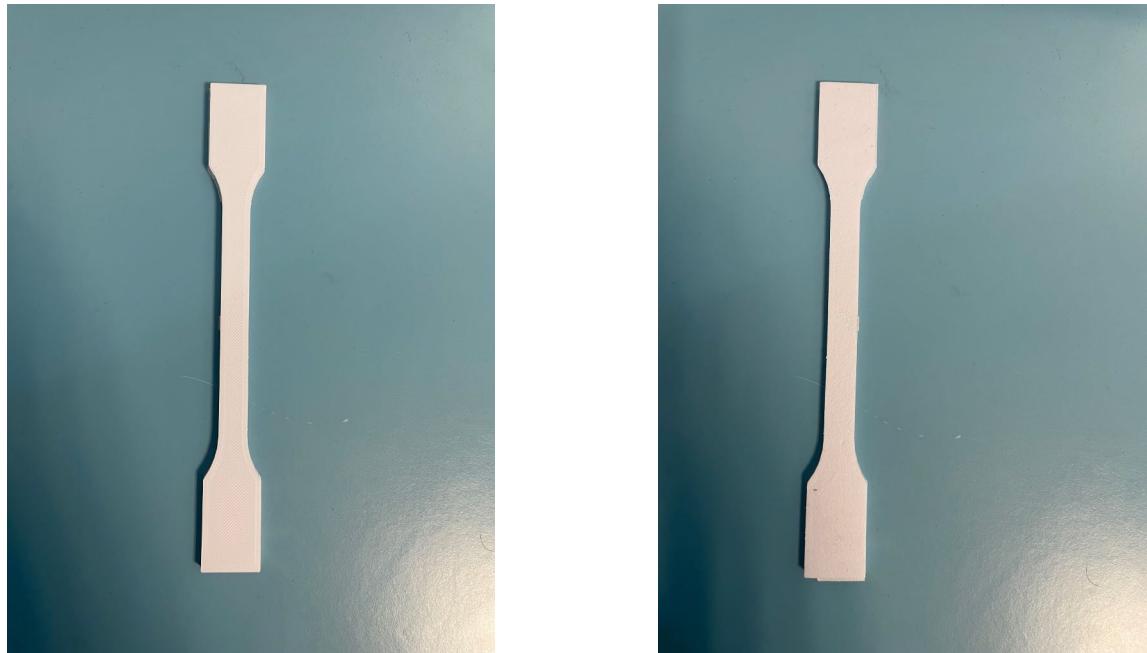


Figure 427: Before tensile test 1 | STB

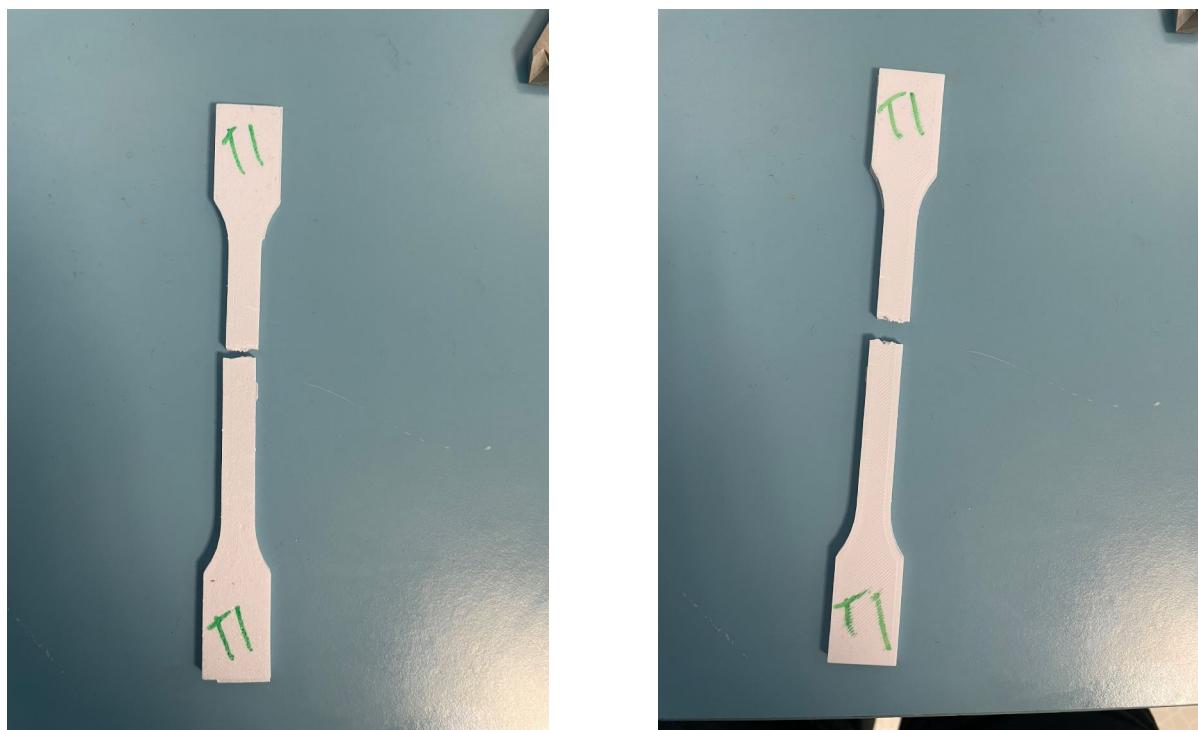


Figure 428: After tensile test 1 | STB

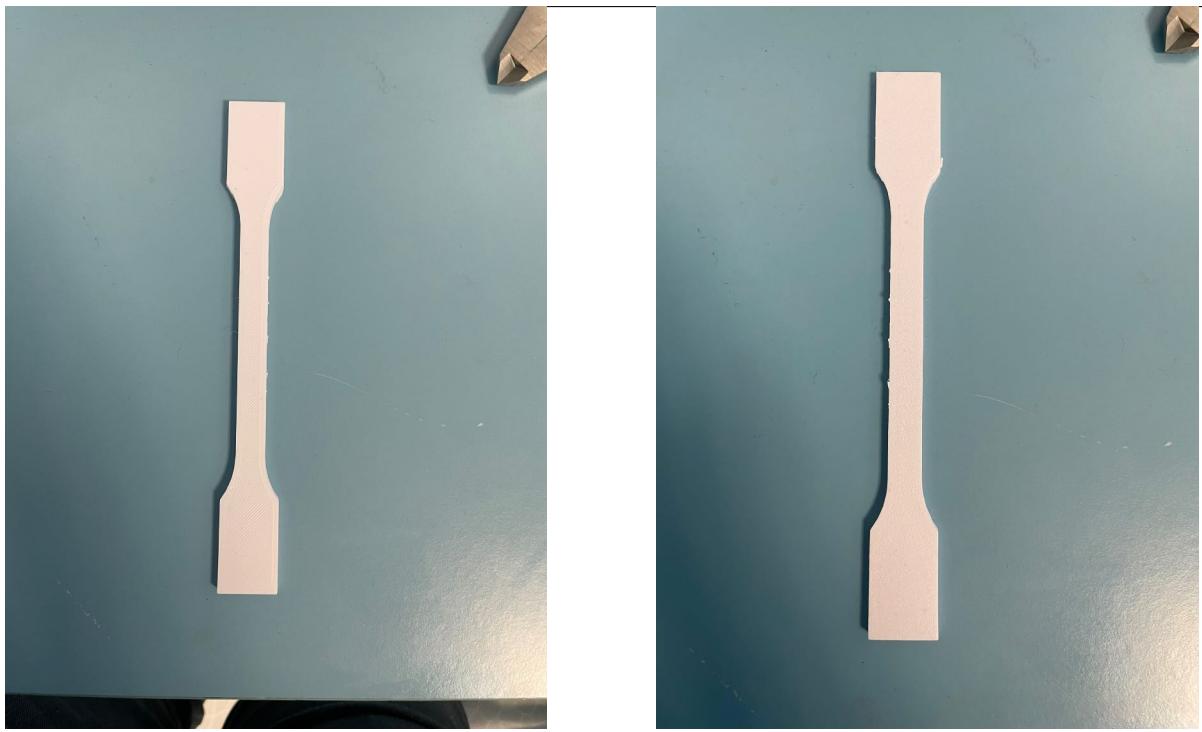


Figure 429: Before tensile test 2 | STB

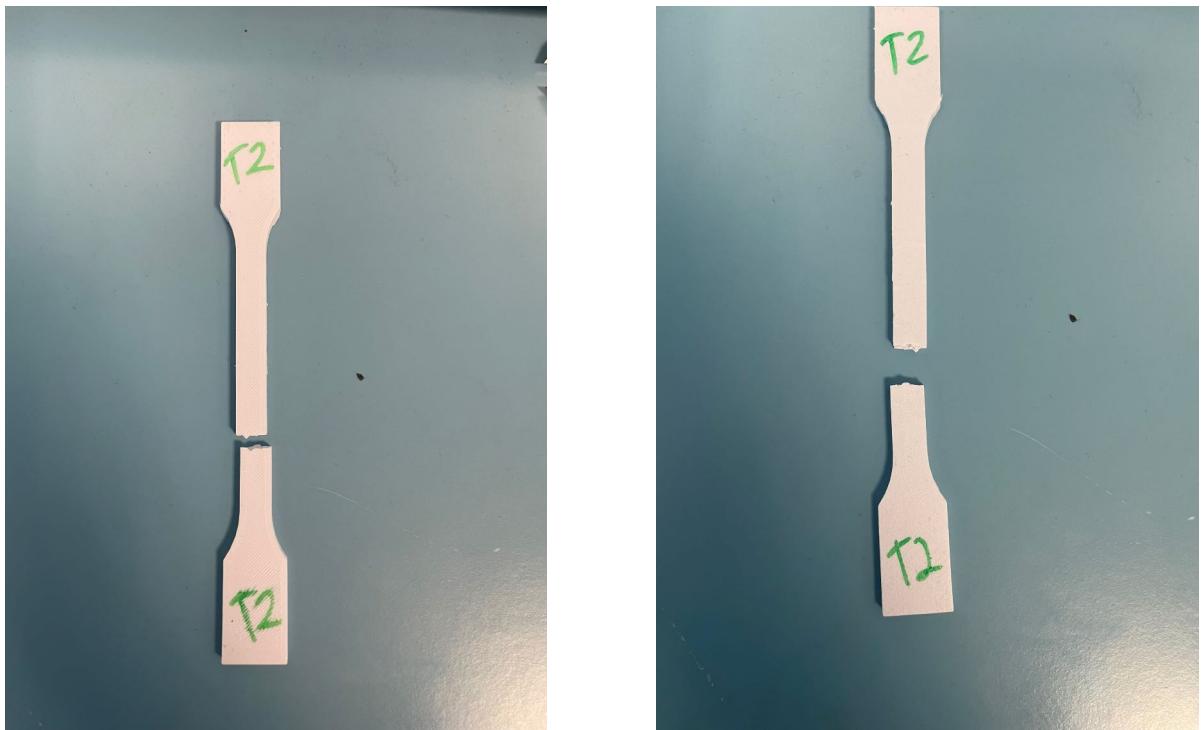


Figure 430: After tensile test 2 | STB

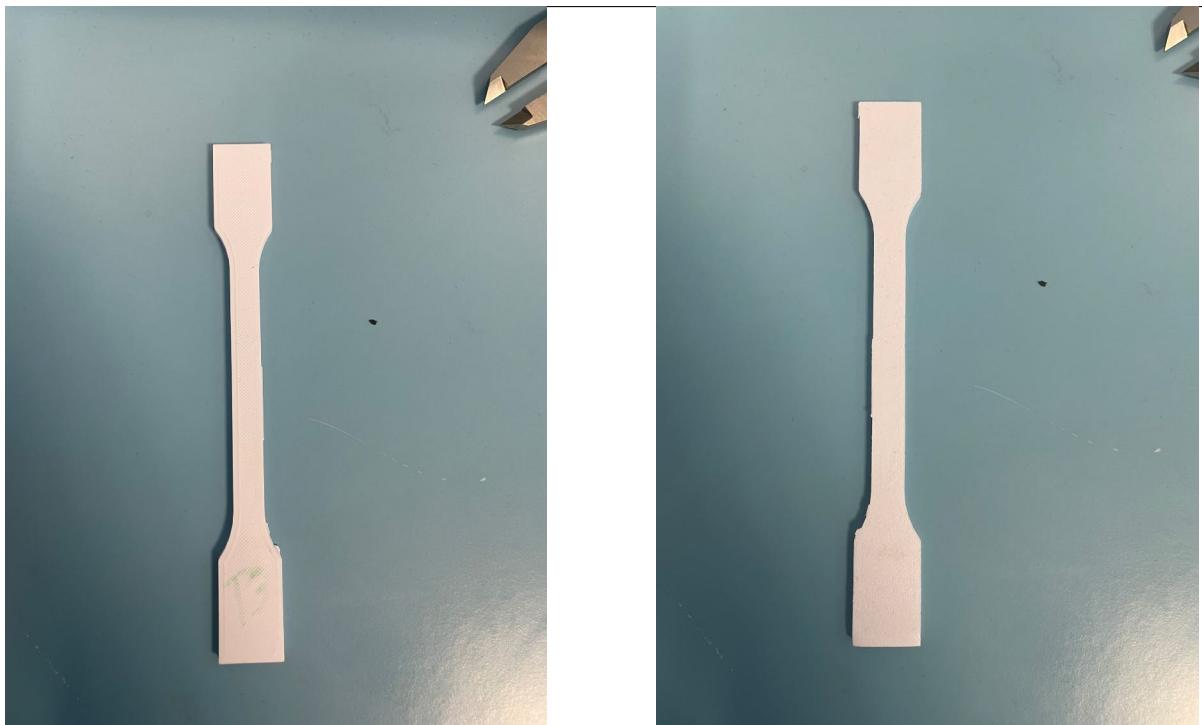


Figure 431: Before tensile test 3 | STB

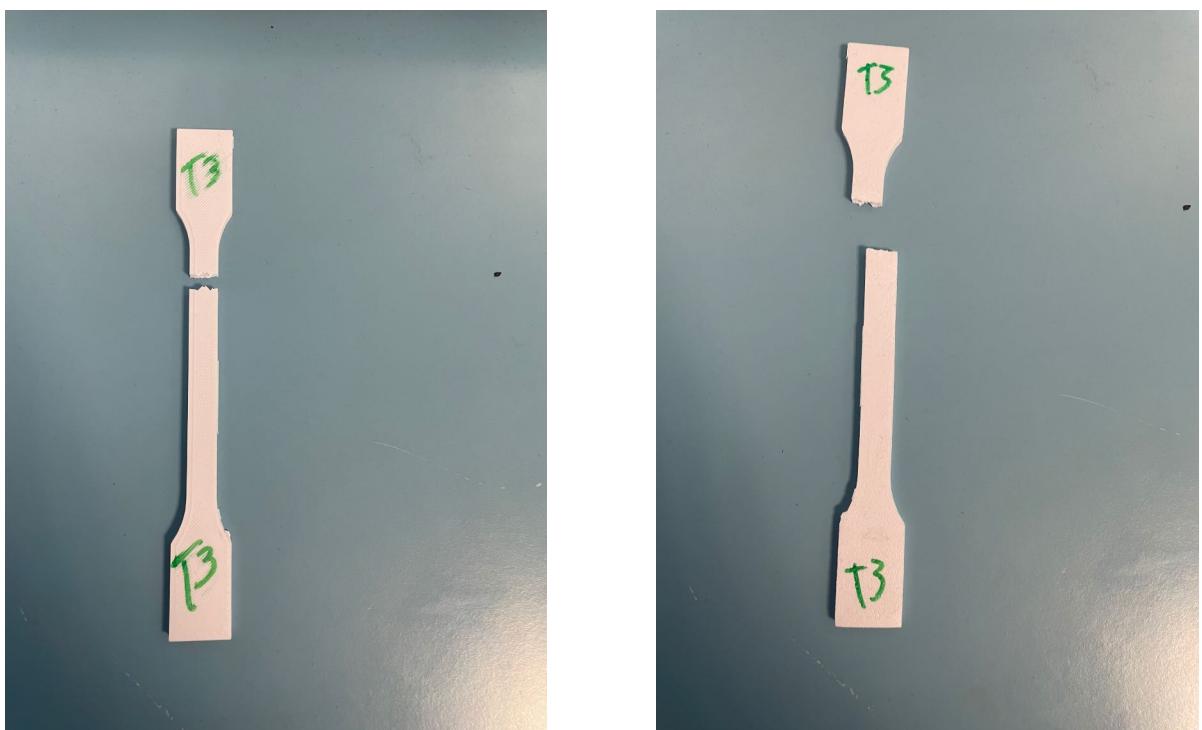


Figure 432: After tensile test 3 | STB

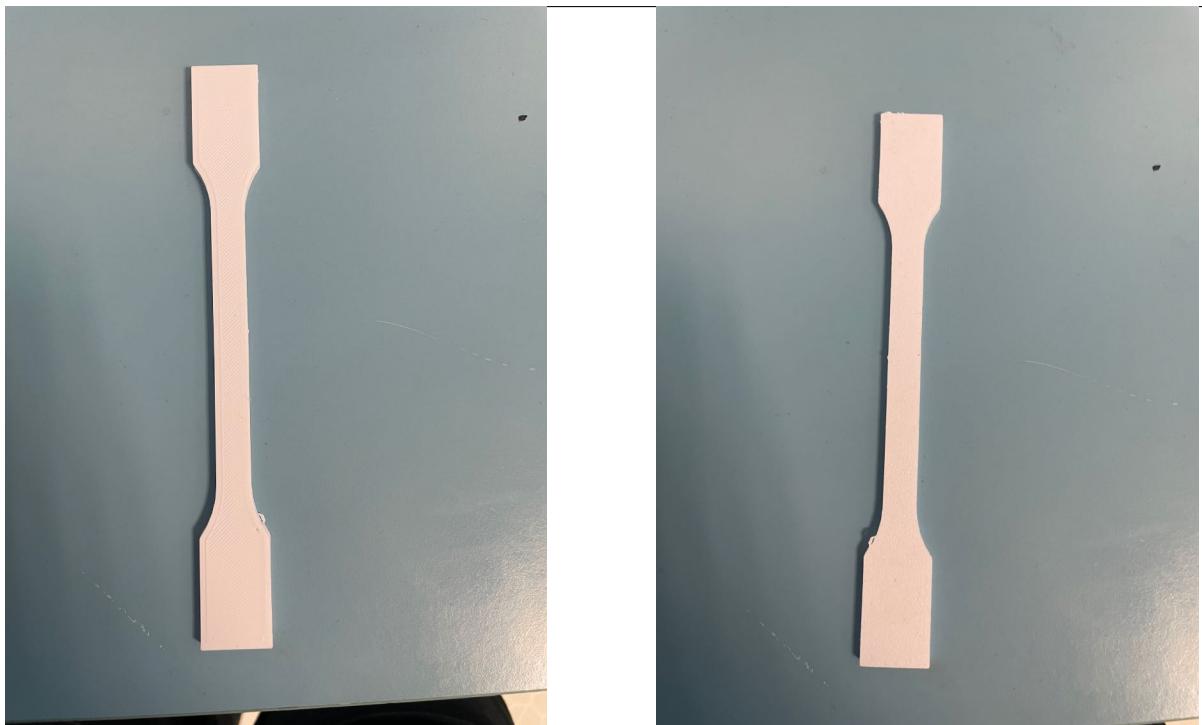


Figure 433: Before tensile test 4 | STB

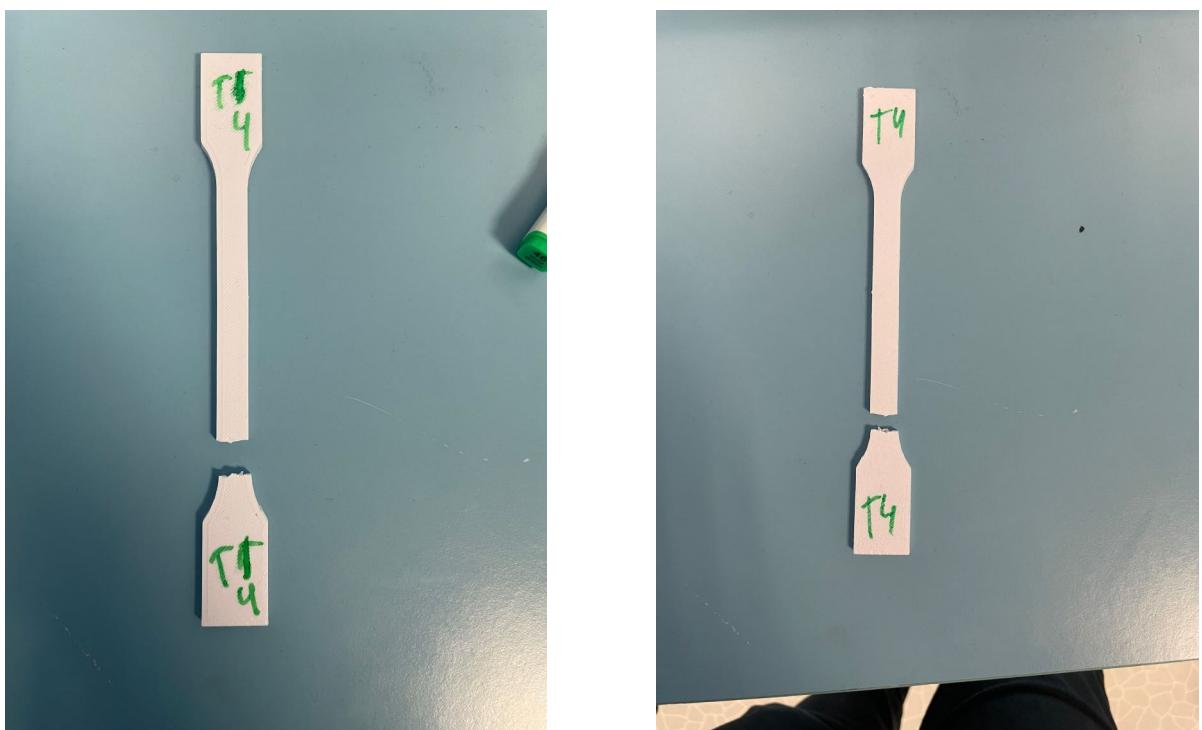


Figure 434: After tensile test 4 | STB

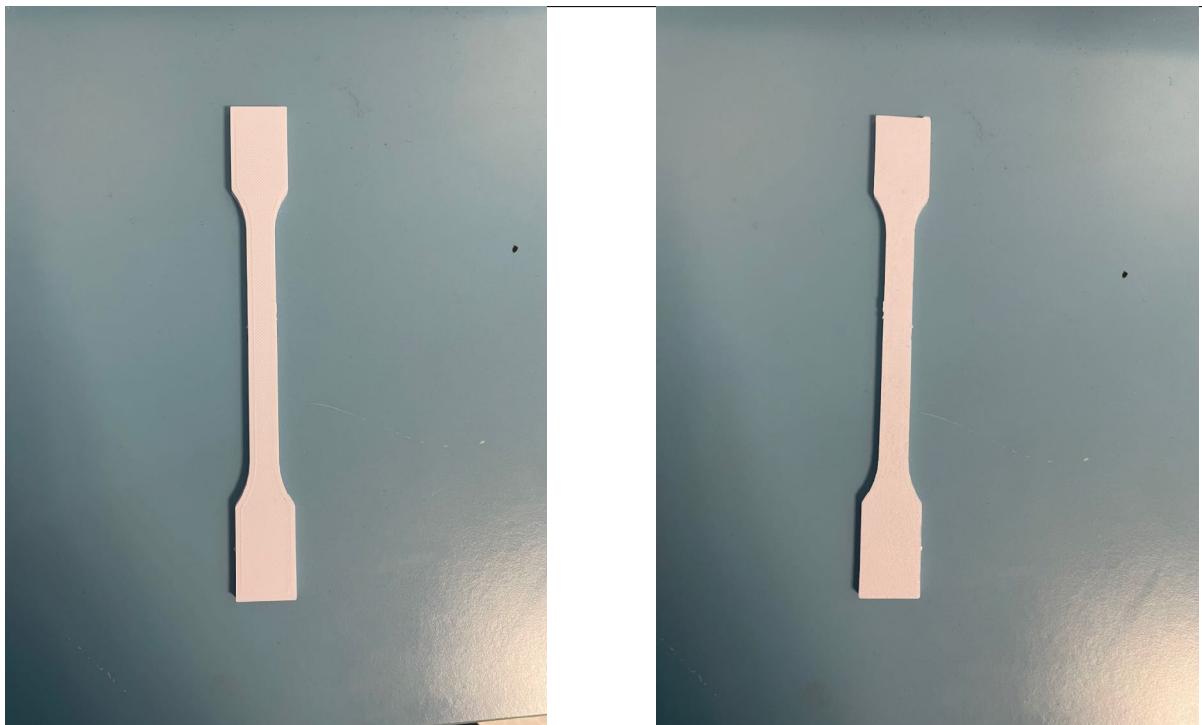


Figure 435: Before tensile test 5 | STB

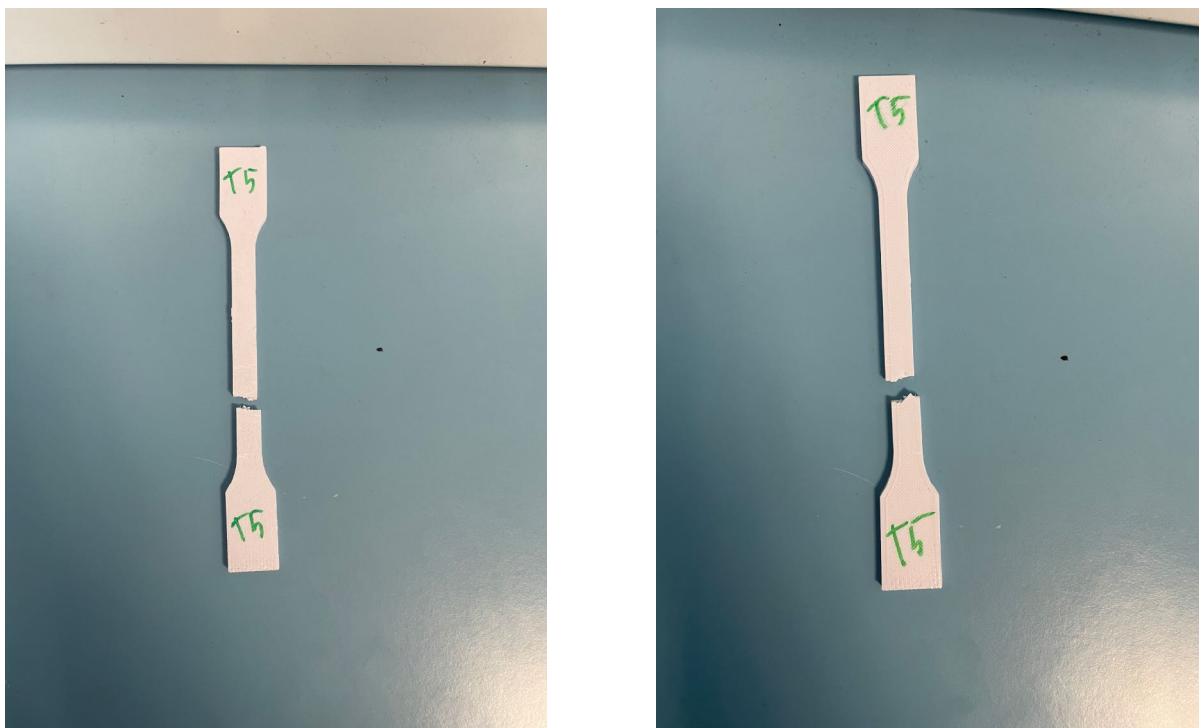


Figure 436: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 437.

	A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values	
2		L_0	Chosen start length [mm]	108,00	108,00	108,00	108,00	108,00	N/A	
3		h	Test staff thickness [mm]	3,85	3,85	3,93	3,88	3,83	3,87	
4		b_1	Width at narrow portion [mm]	10,03	10,03	10,09	10,07	10,08	10,06	
5	DATA	Initial values	A	Cross-section [mm ²]	38,62	38,62	39,65	39,07	38,61	N/A
6			L	Gripping distance	108,00	108,00	108,00	108,00	108,00	N/A
7		Value (Settings/test plan)	N/A	Amount of test	5	5	5	5	5	N/A
8			v	Test Speed [mm/min]	60	60	60	60	60	N/A
9		Proportional limit	σ_y	Nominal Stress [MPa]	9,212	8,984	9,218	8,152	6,448	8,403
10			ε_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0
11			σ_{true}	True stress [MPa]	9,213	8,984	9,218	8,152	6,448	8,403
12			ε_{true}	True strain [%]	N/A	N/A	N/A	N/A	N/A	0
13			E_t	Tensile modulus [MPa]	331642,75	N/A	N/A	N/A	174088,41	101146,23
14	UTS	S_{ut}	Ultimate tensile strength [MPa]	33,931	33,626	33,314	32,456	32,644	33,194	
15		ε_{nom}	Nominal strain [%]	4,07	4,31	3,74	4,49	17,46	6,81	
16		σ_{true}	True stress [MPa]	35,313	35,041	34,562	33,912	38,343	35,434	
17		ε_{true}	True strain [%]	3,99	4,12	3,68	4,39	16,09	6,45	
18	Fracture	σ_b	Stress at break [MPa]	30,682	31,565	31,510	31,879	30,969	31,321	
19		ε_b	Strain (engineering) at break [%]	4,77	4,67	4,16	4,67	18,01	7,26	
20		σ_{true}	True stress [MPa]	32,145	33,072	32,821	33,369	36,547	33,591	
21		ε_{true}	True strain [%]	4,66	4,57	4,07	4,57	16,56	6,89	

Figure 437: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

16. April 2024

**P.14 Material: PLA x Infill pattern: Cubic
Infill density: 50%
Preparation temp.: -38 to -40 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -38°C to -40°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test

Figure 438: Before tensile test 1 | STB

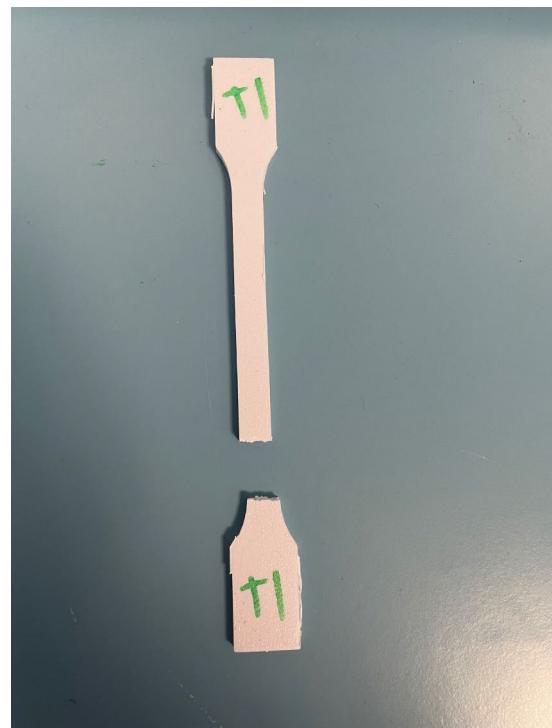


Figure 439: After tensile test 1 | STB

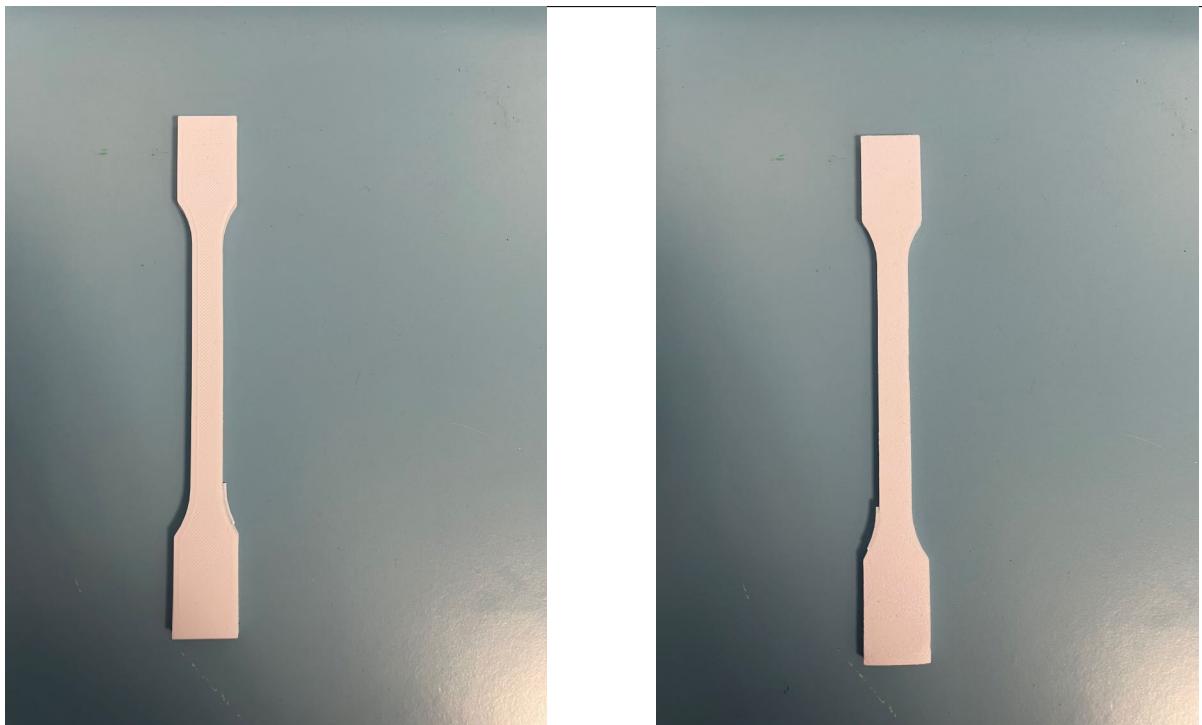


Figure 440: Before tensile test 2 | STB

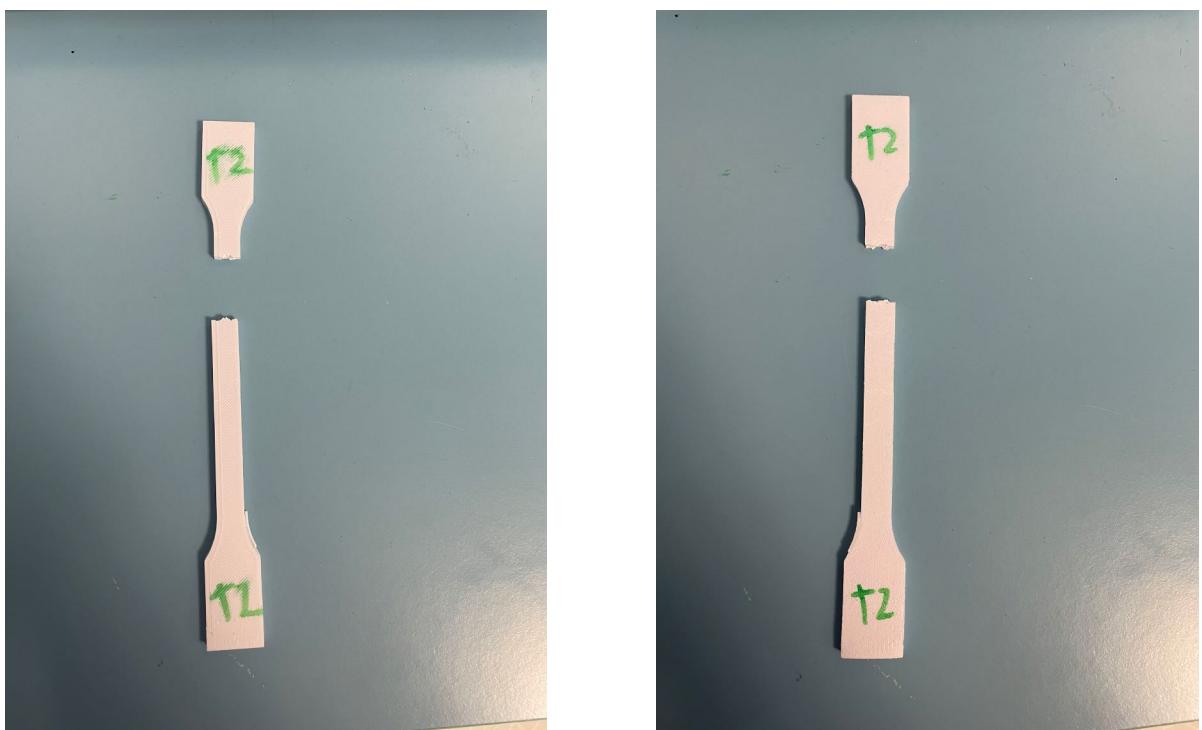


Figure 441: After tensile test 2 | STB

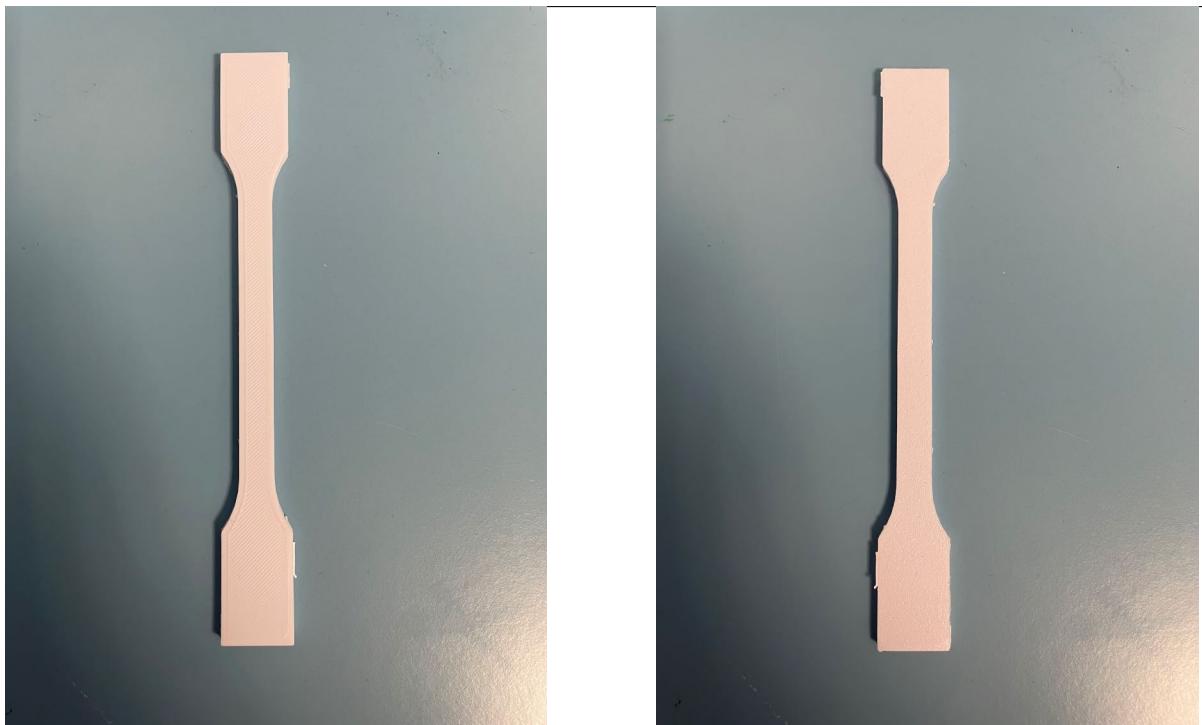


Figure 442: Before tensile test 3 | STB

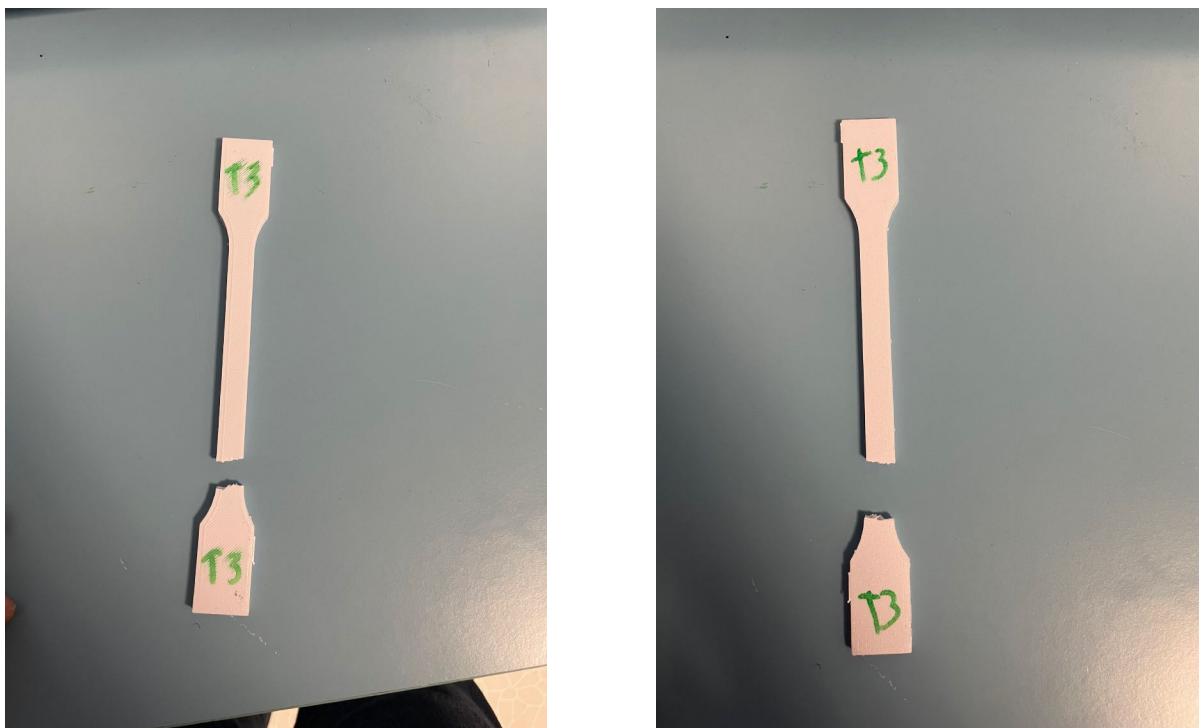


Figure 443: After tensile test 3 | STB

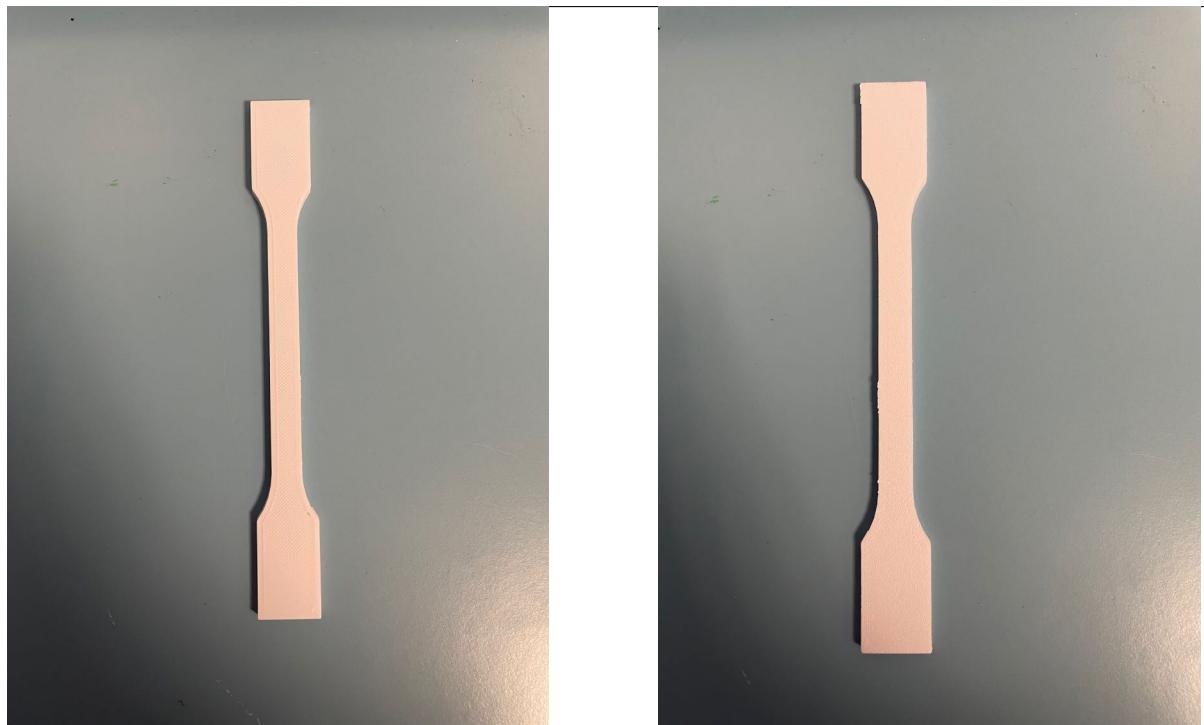


Figure 444: Before tensile test 4 | STB

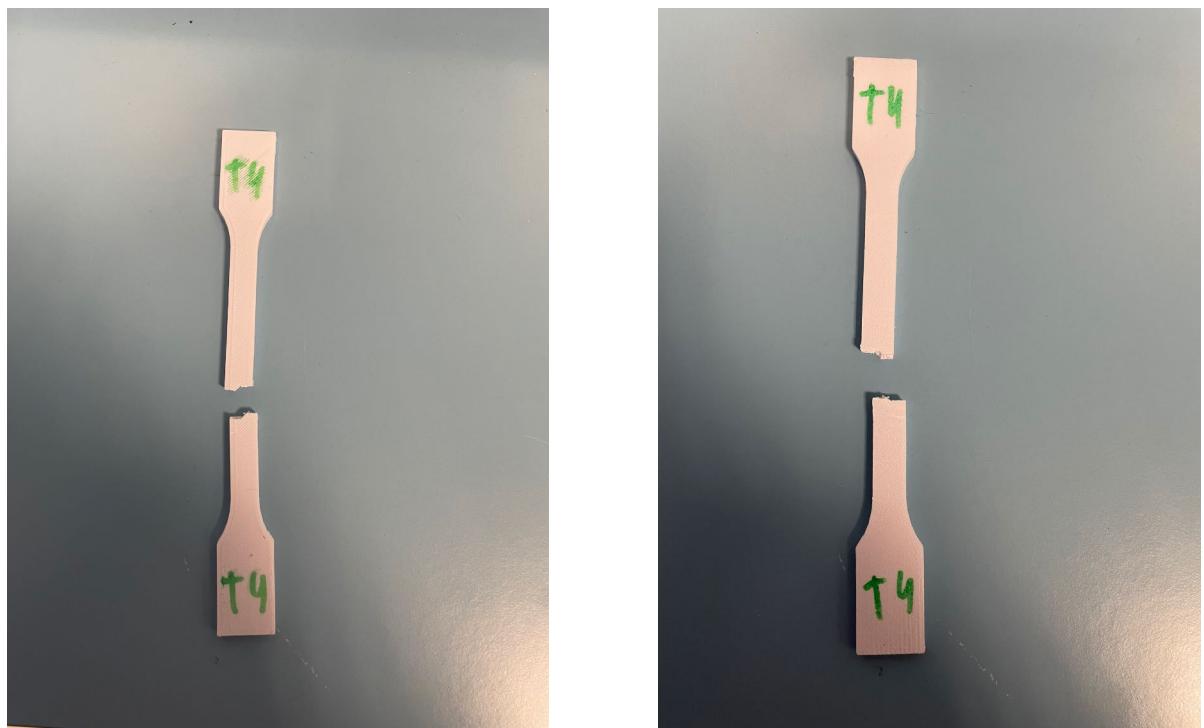


Figure 445: After tensile test 4 | STB

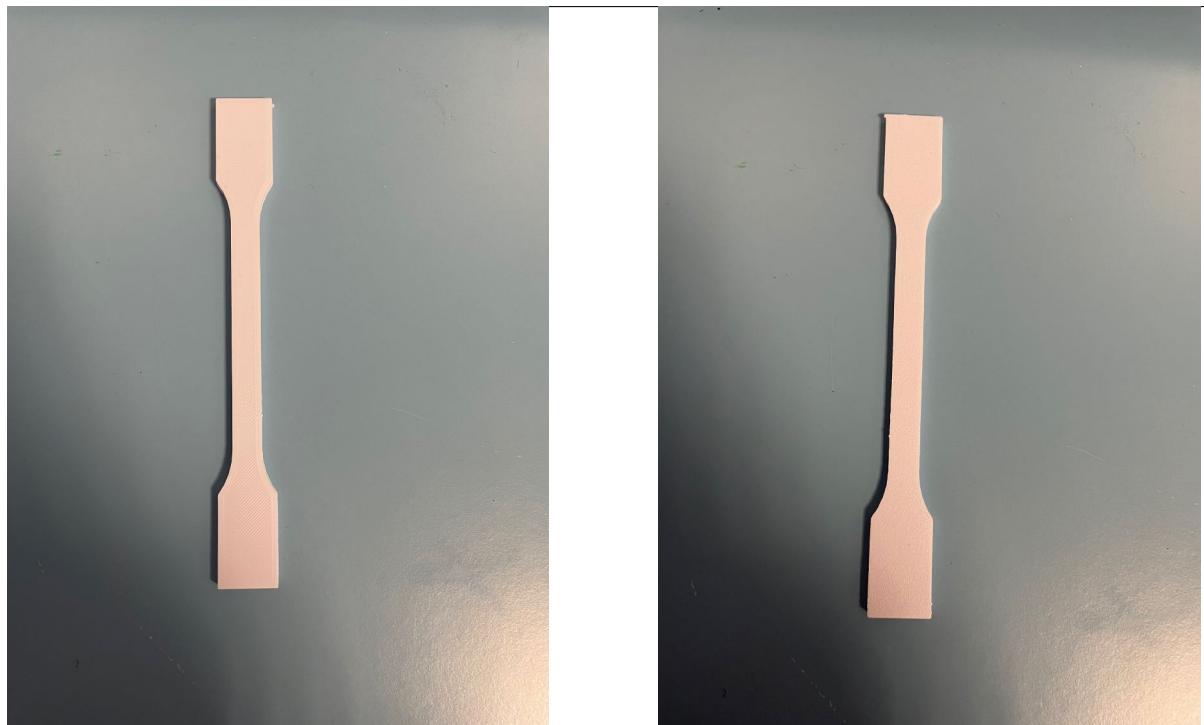


Figure 446: Before tensile test 5 | STB



Figure 447: After tensile test 5 | STB

STB|

Properties from each tensile test specimen & mean values

The result from the tensile test is shown in fig 448

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
2	Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
		h	Test staff thickness [mm]	3,83	3,86	3,85	3,83	3,82	3,84
		b_1	Width at narrow portion [mm]	9,98	9,91	9,97	10,30	10,05	10,04
5	Value (Settings/test plan)	A	Cross-section [mm ²]	38,22	38,25	38,38	39,45	38,39	N/A
		L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
		N/A	Amount of test	5	5	5	5	5	N/A
		v	Test Speed [mm/min]	60	60	60	60	60	N/A
9	Proportional limit	σ_y	Nominal Stress [MPa]	5,743	6,046	6,562	10,036	11,334	7,944
		ϵ_{nom}	Nominal strain [%]	0	0	0	0,95	0,95	0,38
		σ_{true}	True stress [MPa]	N/A	N/A	N/A	1057,980	11,442	2,288
		ϵ_{true}	True strain [%]	5,743	6,046	6,561	10,131	0,95	5,8862
14	UTS	E_t	Tensile modulus [MPa]	0,00	0,00	0,00	0,94	1187,88	237,76
		S_{ut}	Ultimate tensile strength [MPa]	35,894	36,507	34,493	35,028	34,410	35,266
		ϵ_{nom}	Nominal strain [%]	6,64	6,09	5,22	4,48	4,80	5,45
		σ_{true}	True stress [MPa]	38,279	38,732	36,294	36,596	36,063	37,193
18	Fracture	ϵ_{true}	True strain [%]	6,43	5,92	5,09	4,38	4,69	5,30
		σ_b	Stress at break [MPa]	35,612	33,638	34,288	32,916	1,378	27,566
		ϵ_b	Strain (engineering) at break [%]	6,78	6,74	5,27	5,03	5,08	5,78
		σ_{true}	True stress [MPa]	38,028	35,904	36,095	34,57	1,449	29,209
20		ϵ_{true}	True strain [%]	6,56	6,52	5,13	4,90	4,96	5,61

Figure 448: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

STB|

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

STB|

Date of measurement

18. April 2024

STB|

P.15 Material: PLA x Infill pattern: Cubic
 Infill density: 50%
 Preparation temp.: 20 to 24 Celsius

The documentation and testing is based on sections/requirements from standards:

-
- NS-EN ISO 527-1:2019
 - NS-EN ISO 291:2008
 - NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed room for 19 hours with the *preparation temperature* 20°C to 24°C. The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C.

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between 20°C and 24°C before tensile test.

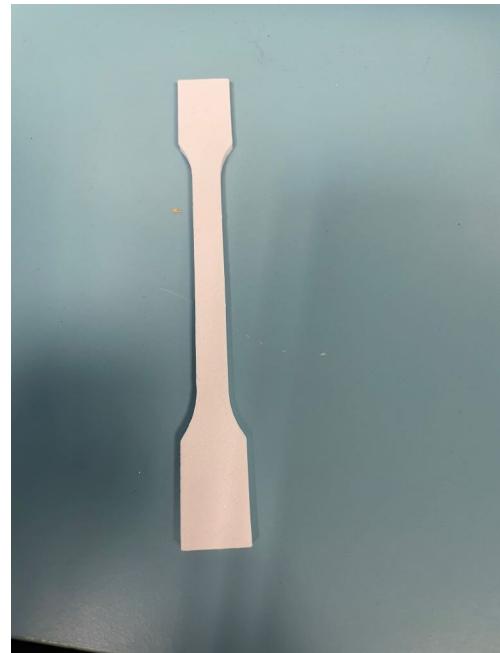
Pictures - before & after tensile test

Figure 449: Before tensile test 1 | STB

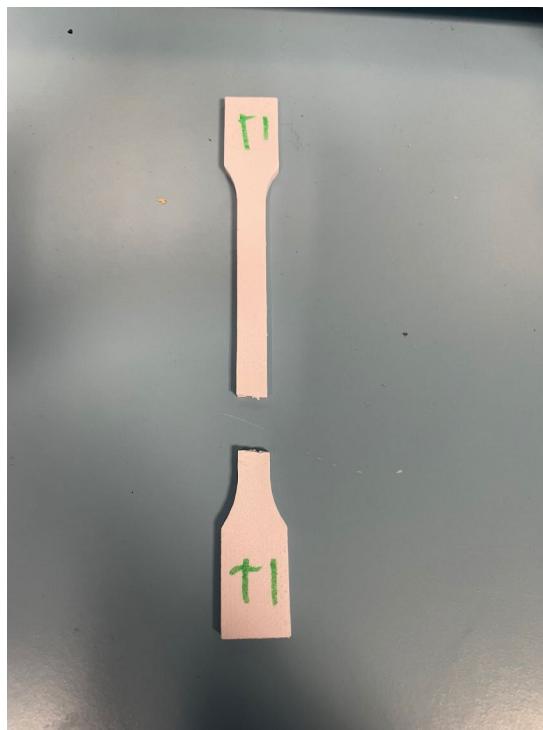


Figure 450: After tensile test 1 | STB

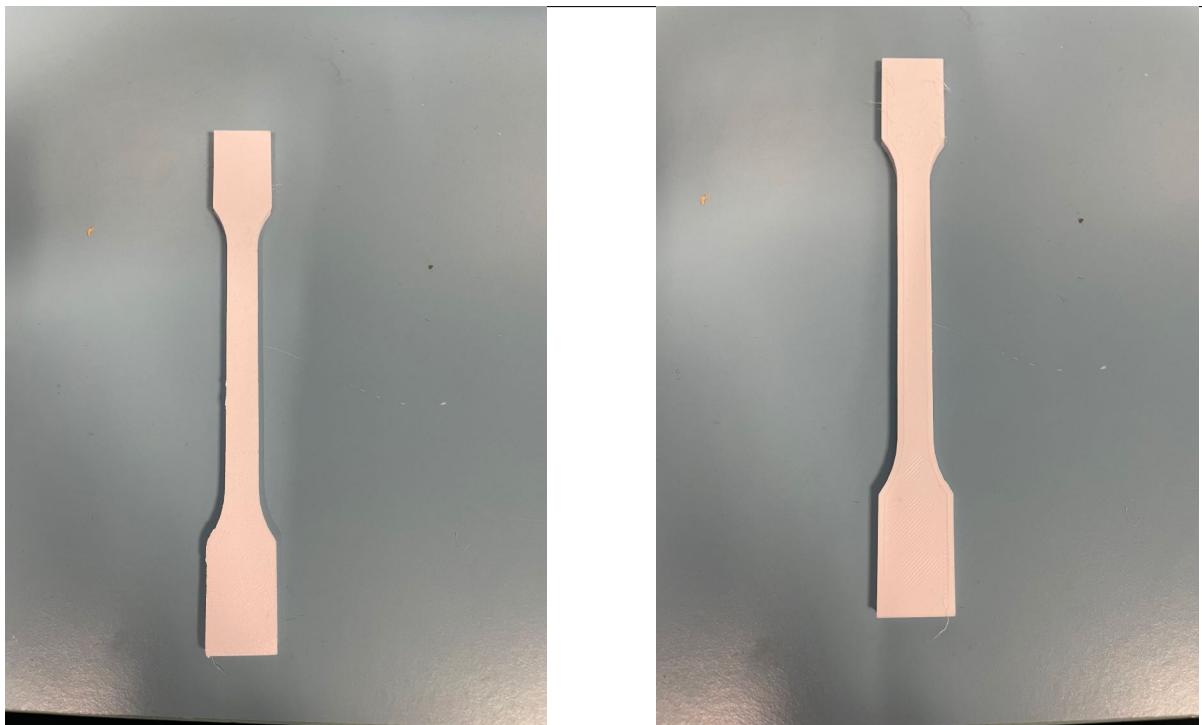


Figure 451: Before tensile test 2 | STB

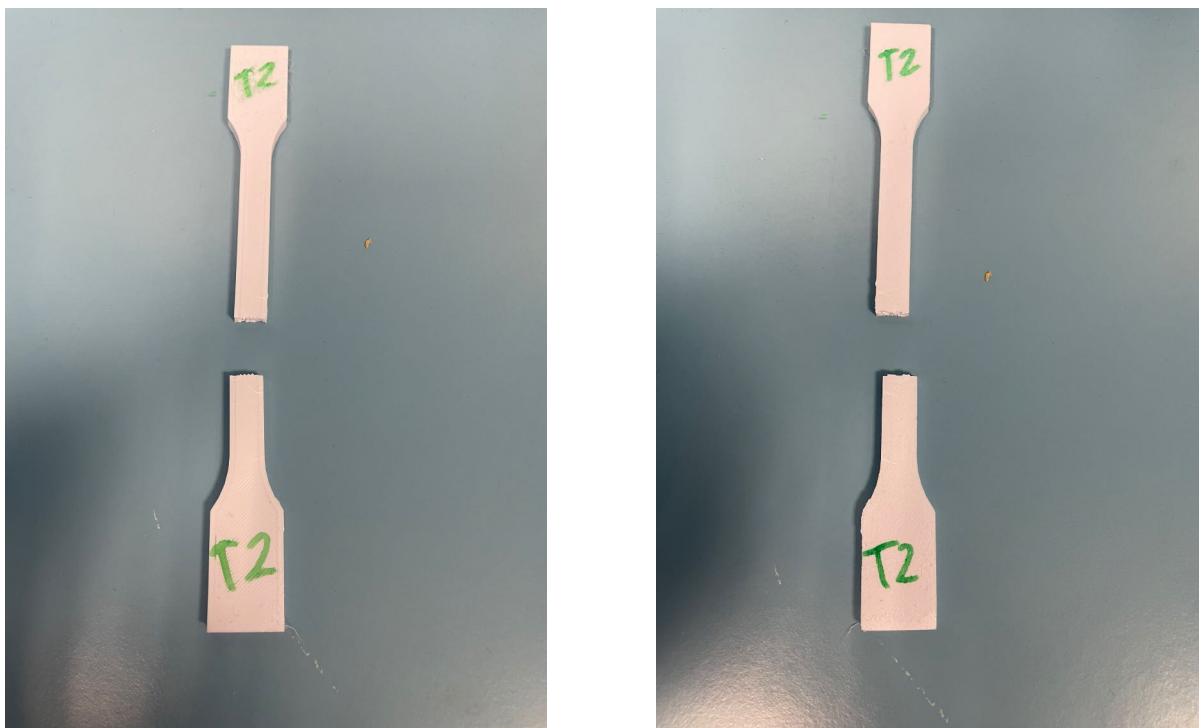


Figure 452: After tensile test 2 | STB

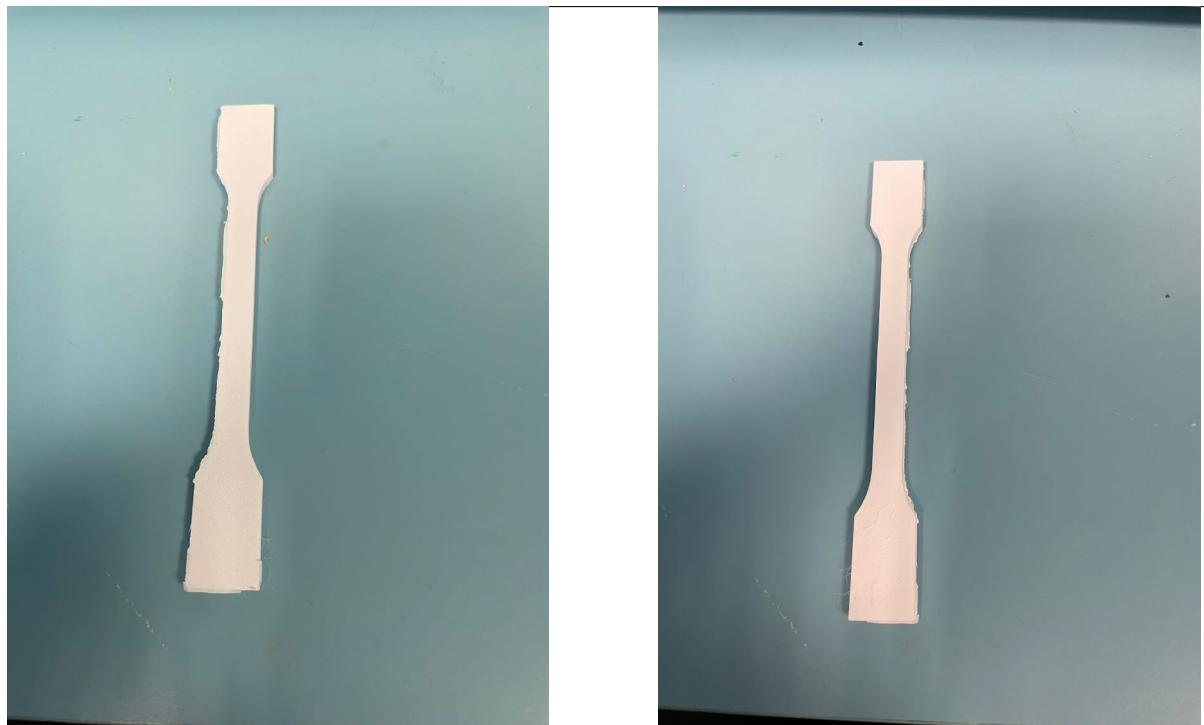


Figure 453: Before tensile test 3 | STB

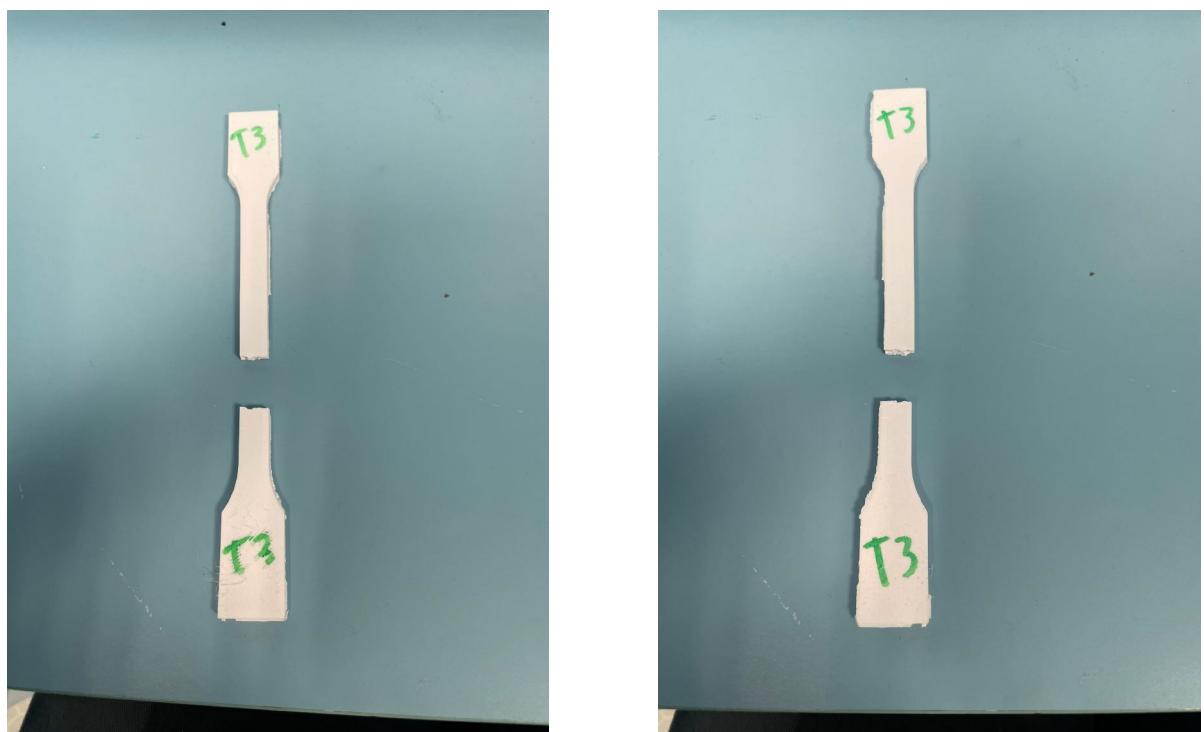


Figure 454: After tensile test 3 | STB

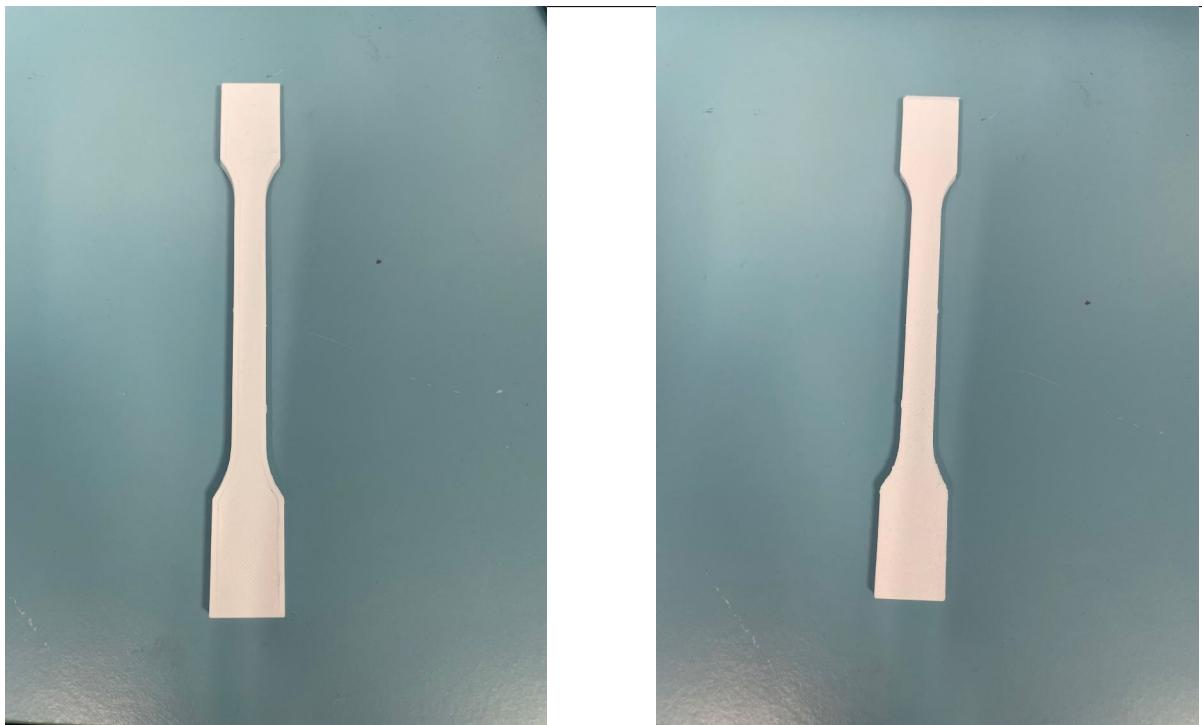


Figure 455: Before tensile test 4 | STB

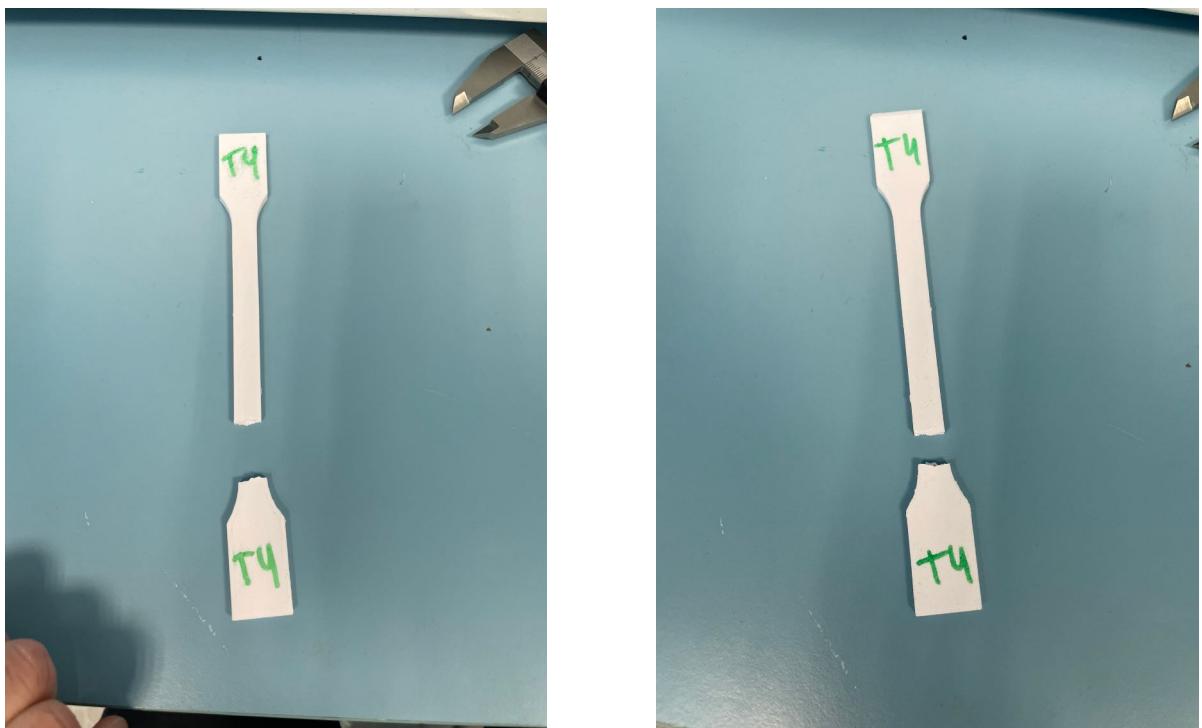


Figure 456: After tensile test 4 | STB

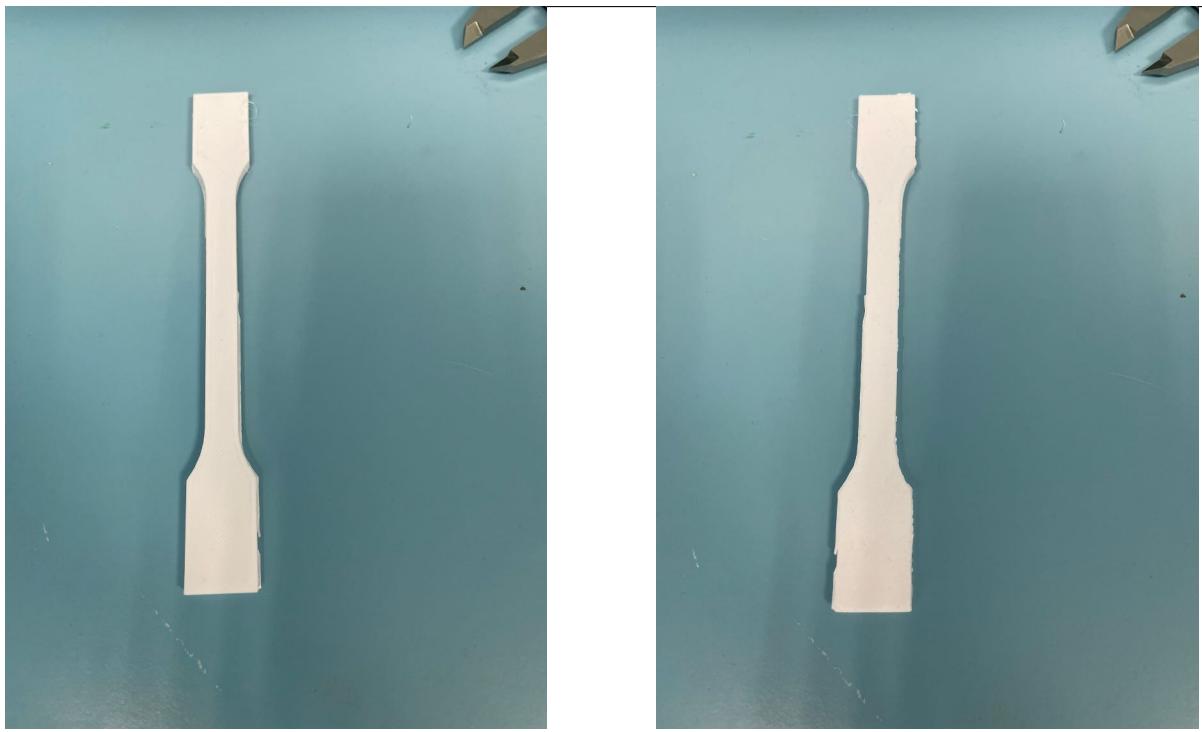


Figure 457: Before tensile test 5 | STB

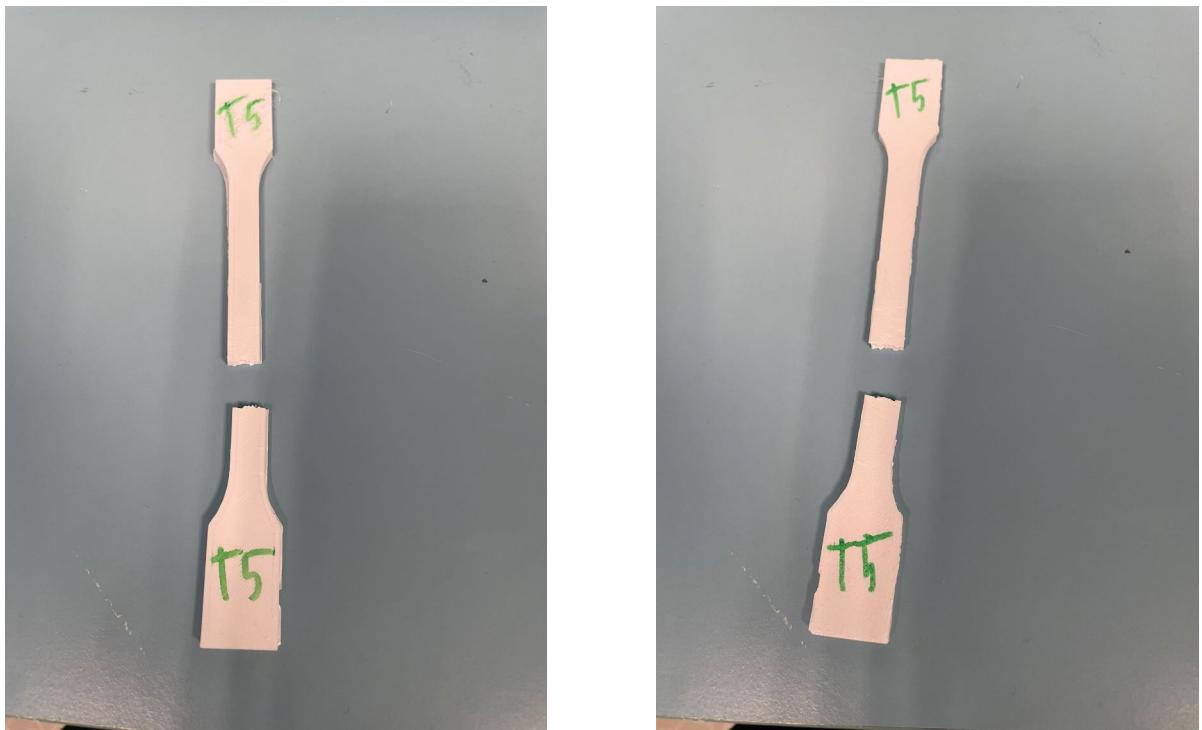


Figure 458: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 459.

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
2		L_0	Chosen start length [mm]	108,00	108,00	108,00	108,00	108,00	N/A
3		h	Test staff thickness [mm]	3,81	3,85	3,83	3,80	3,83	3,82
4		b_1	Width at narrow portion [mm]	10,00	10,05	9,97	10,05	10,03	10,02
5		Value (Settings/test plan)	A	Cross-section [mm ²]	38,10	38,69	38,19	38,19	38,41
6			L	Gripping distance	108,00	108,00	108,00	108,00	108,00
7			N/A	Amount of test	5	5	5	5	N/A
8			v	Test Speed [mm/min]	60	60	60	60	N/A
9		Proportional limit	σ_y	Nominal Stress [MPa]	9,311	9,523	6,929	5,517	8,155
10			ε_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	0
11			σ_{true}	True stress [MPa]	9,311	9,523	6,929	5,517	8,155
12			ε_{true}	True strain [%]	N/A	N/A	N/A	N/A	0
13			E_t	Tensile modulus [MPa]	N/A	1028590,19	N/A	297947,75	N/A
14		UTS	S_ut	Ultimate tensile strength [MPa]	36,499	36,371	36,752	35,284	36,364
15			ε_{nom}	Nominal strain [%]	4,35	4,35	4,90	5,93	5,51
16			σ_{true}	True stress [MPa]	38,086	37,952	38,554	37,375	38,367
17			ε_{true}	True strain [%]	4,26	4,26	4,79	5,76	5,36
18		Fracture	σ_b	Stress at break [MPa]	35,136	35,003	35,443	35,002	34,98
19			ε_b	Strain (engineering) at break [%]	4,67	4,67	5,23	6,06	5,88
20			σ_{true}	True stress [MPa]	36,778	36,639	37,296	37,124	37,036
21			ε_{true}	True strain [%]	4,57	4,57	5,10	5,89	5,71
22									5,17

Figure 459: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

17. April 2024

**P.16 Material: PLA x Infill pattern: Cubic
Infill density: 75%
Preparation temp.: -18 to -20 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -18°C to -20°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test

Figure 460: Before tensile test 1 | STB

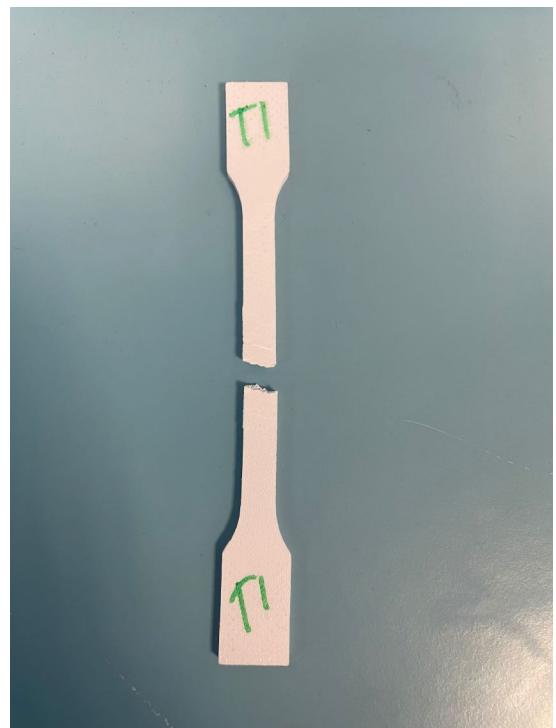


Figure 461: After tensile test 1 | STB

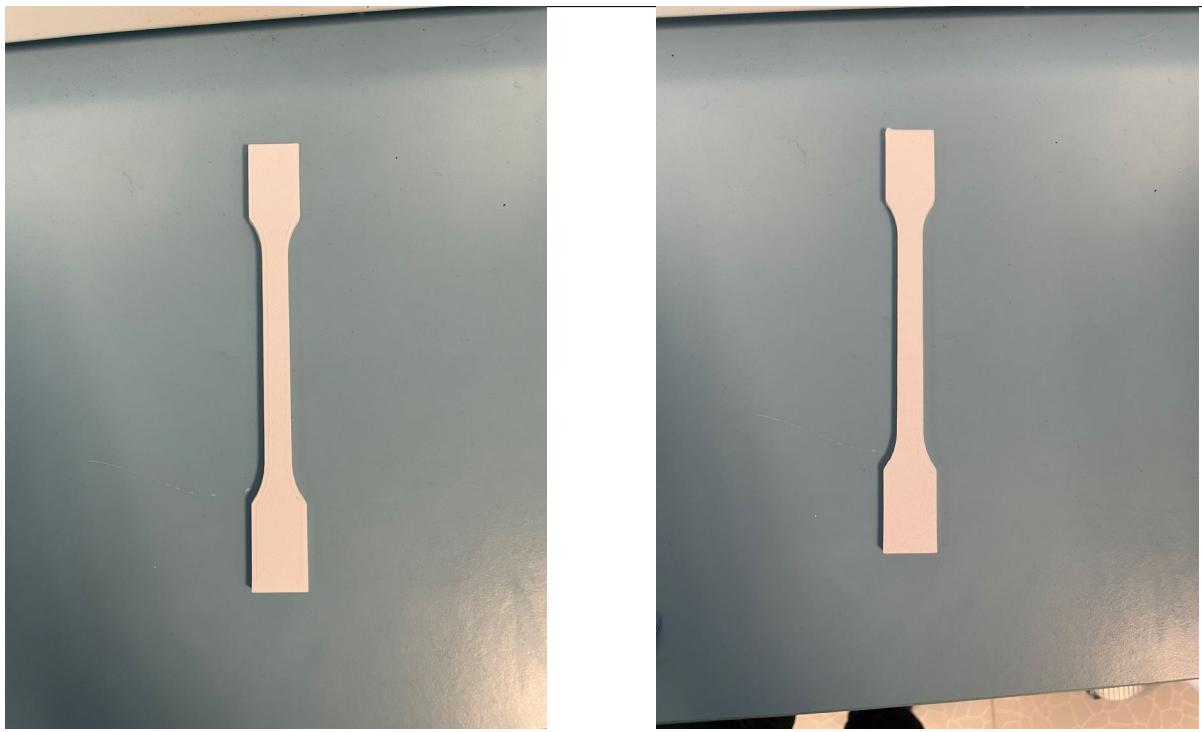


Figure 462: Before tensile test 2 | STB

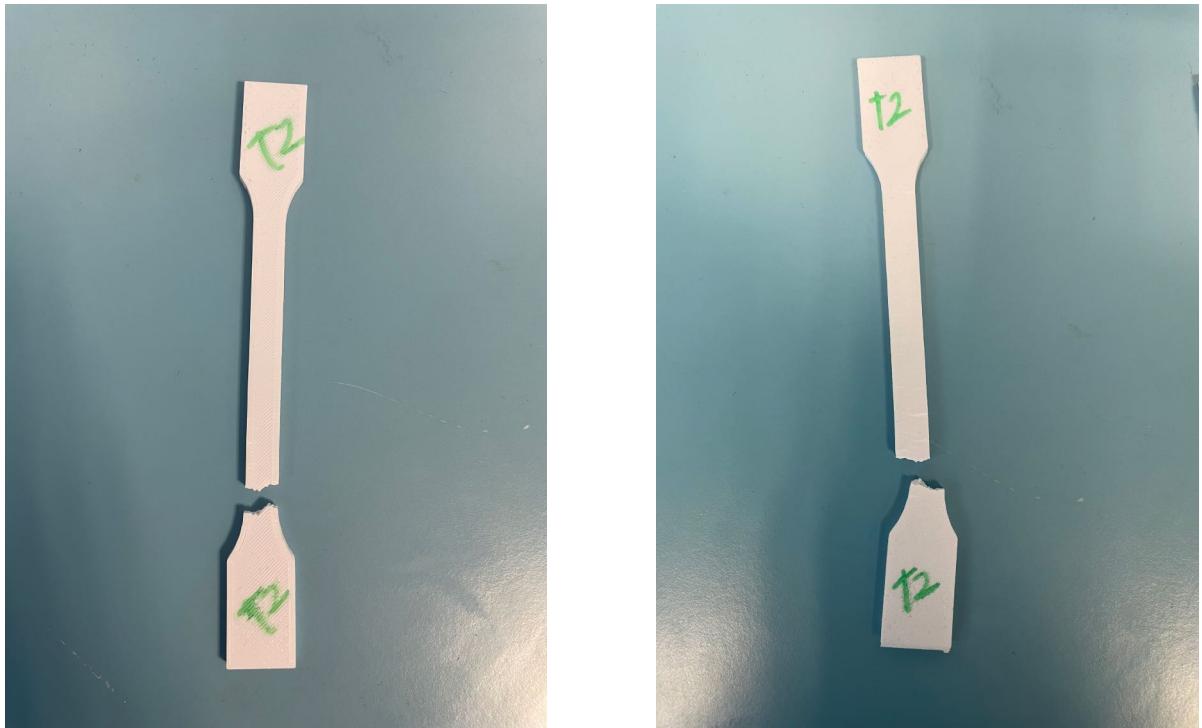


Figure 463: After tensile test 2 | STB

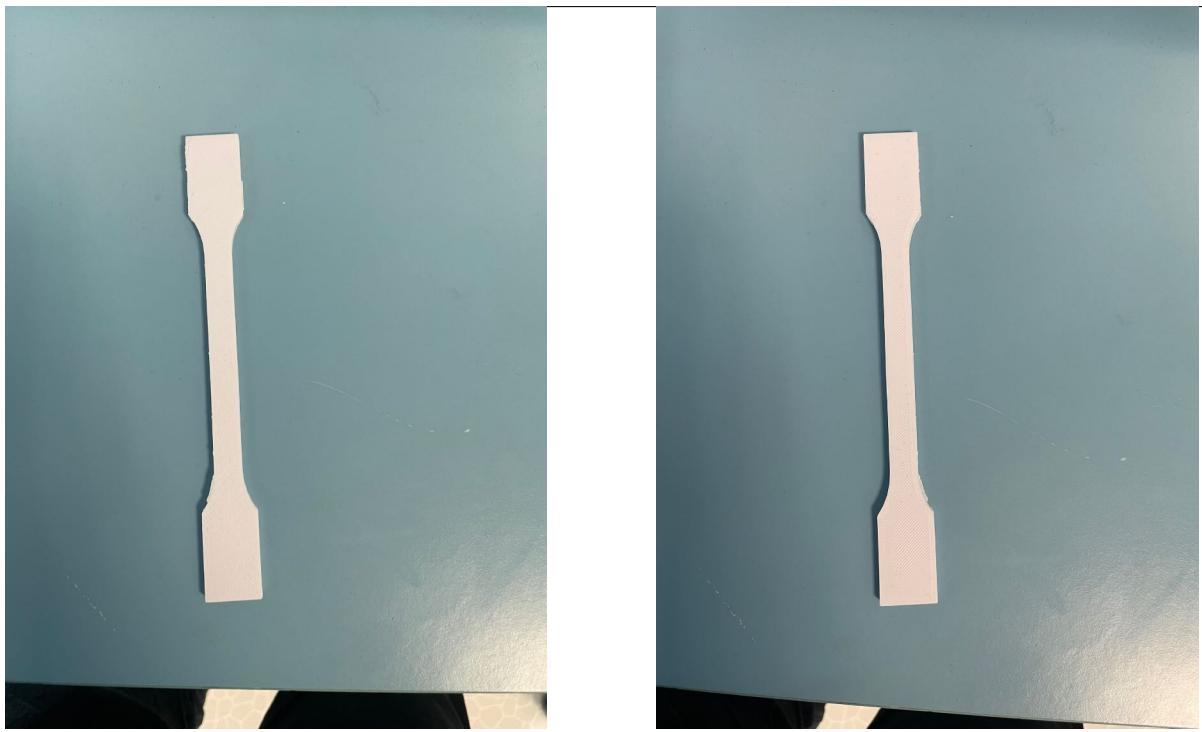


Figure 464: Before tensile test 3 | STB

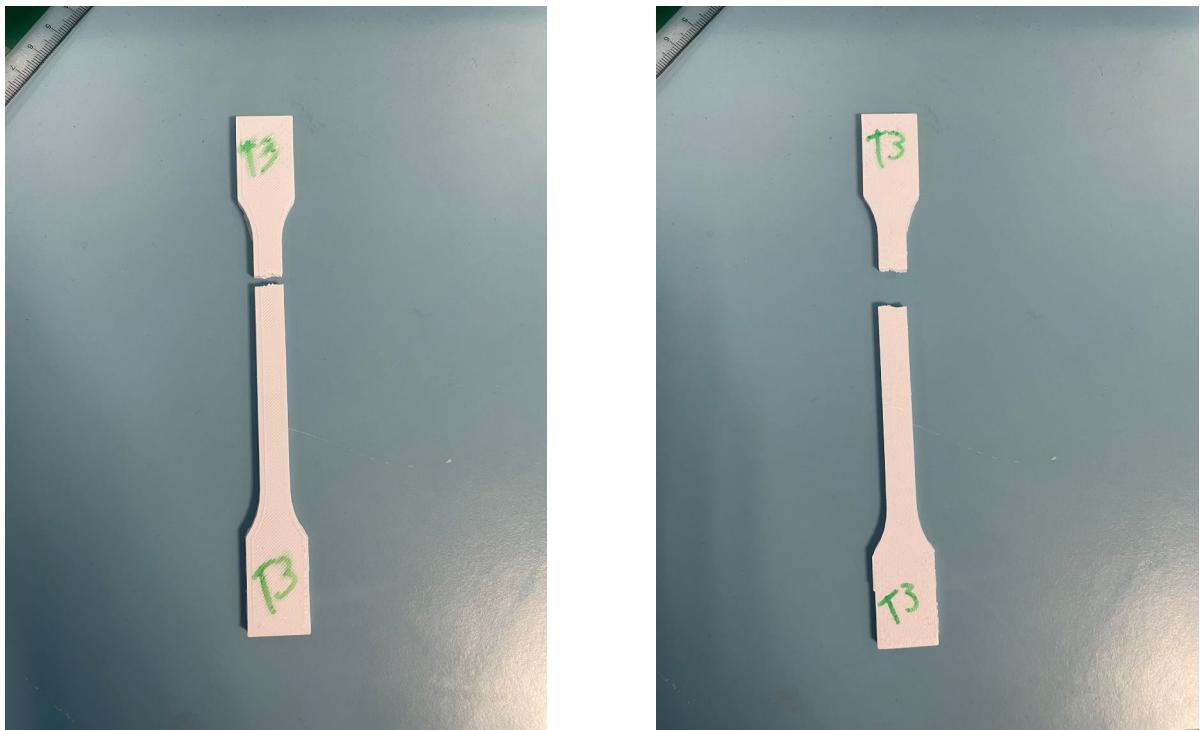


Figure 465: After tensile test 3 | STB

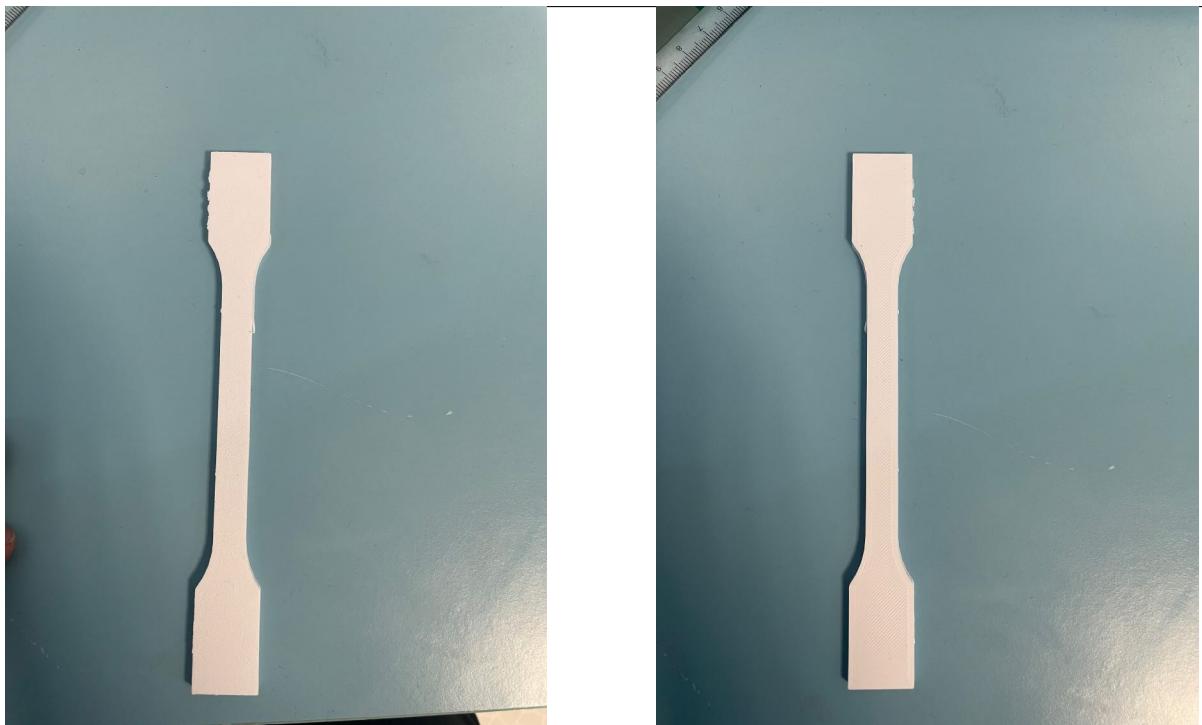


Figure 466: Before tensile test 4 | STB

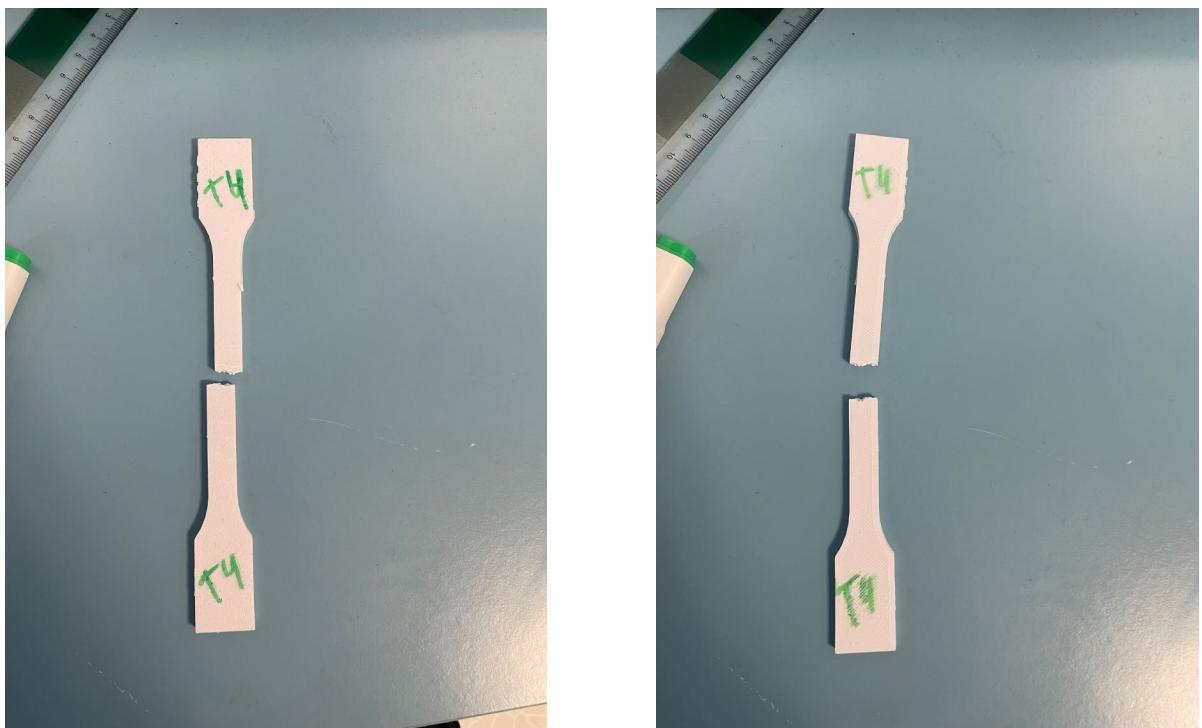


Figure 467: After tensile test 4 | STB

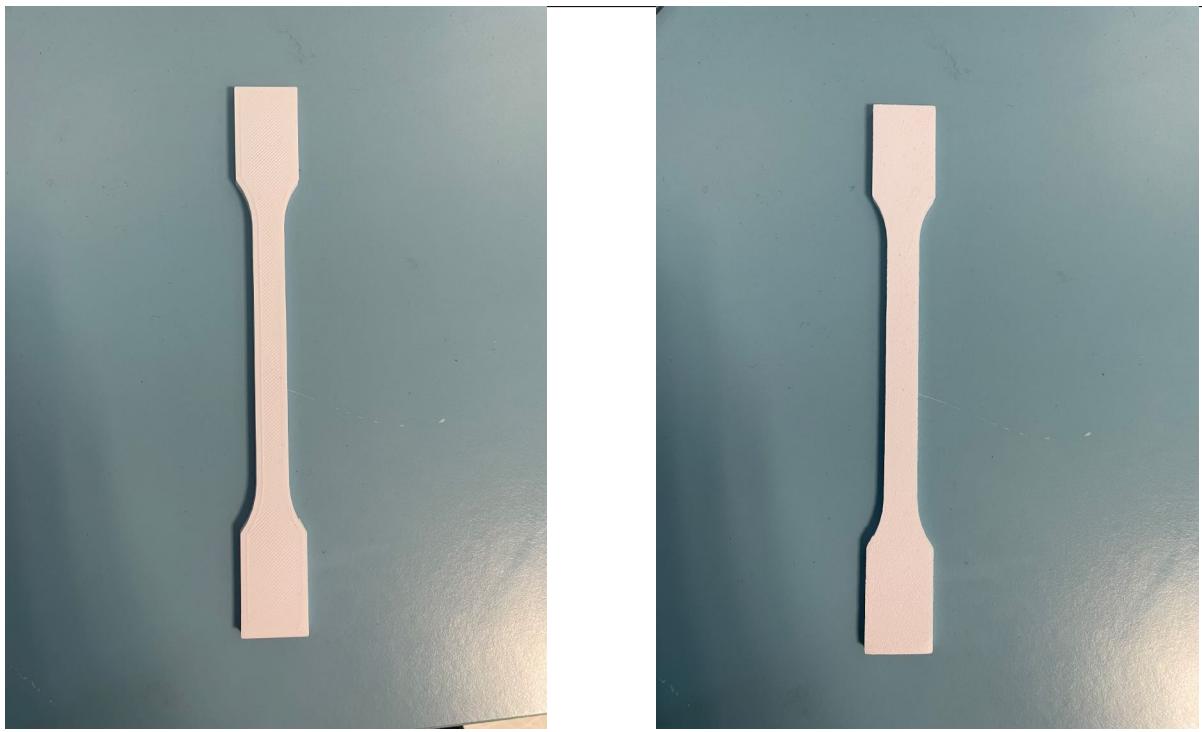


Figure 468: Before tensile test 5 | STB

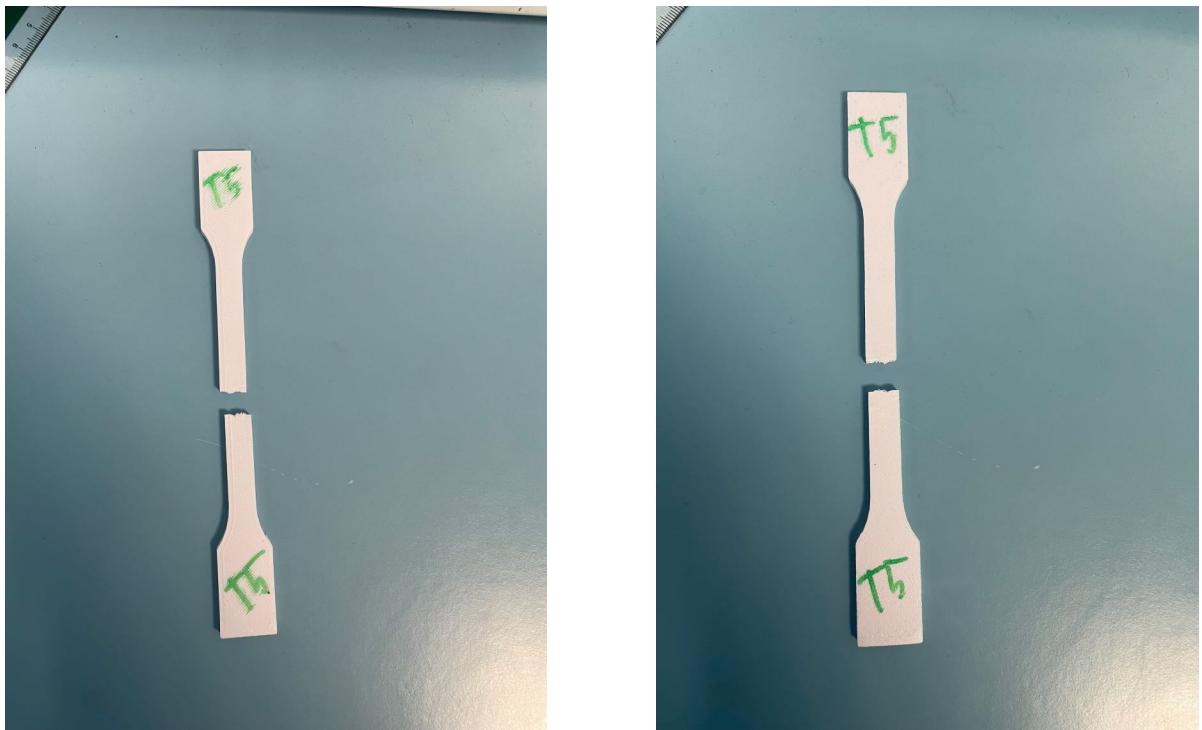


Figure 469: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 470.

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
2	Initial values	L_0	Chosen start length [mm]	108,00	108,00	108,00	108,00	108,00	N/A
3		h	Test staff thickness [mm]	3,86	3,83	3,86	3,86	3,88	3,86
4		b_1	Width at narrow portion [mm]	10,06	10,03	10,00	10,00	10,02	10,02
5	Value (Settings/test plan)	A	Cross-section [mm^2]	38,83	38,41	38,60	38,60	38,88	N/A
6		L	Gripping distance	108,00	108,00	108,00	108,00	108,00	N/A
7		N/A	Amount of test	5	5	5	5	5	N/A
8	Proportional limit	v	Test Speed [mm/min]	60	60	60	60	60	N/A
9		σ_y	Nominal Stress [MPa]	11,130	10,026	9,419	8,658	9,226	9,692
10		ε_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	0	0
11	UTS	σ_{true}	True stress [MPa]	11,130	10,026	9,419	8,658	9,226	9,692
12		$\varepsilon_{\text{true}}$	True strain [%]	N/A	N/A	N/A	N/A	N/A	0
13		E_t	Tensile modulus [MPa]	601005,88	216490,34	N/A	187006,92	N/A	200900,63
14	Fracture	S_{ut}	Ultimate tensile strength [MPa]	37,073	36,991	38,032	37,423	37,937	37,491
15		ε_{nom}	Nominal strain [%]	3,80	3,66	3,88	3,75	3,83	3,78
16		σ_{true}	True stress [MPa]	38,381	38,343	39,508	38,826	39,391	38,890
17		$\varepsilon_{\text{true}}$	True strain [%]	3,73	3,59	3,81	3,68	3,76	3,71
18	Fracture	σ_b	Stress at break [MPa]	33,162	33,317	35,265	33,741	32,643	33,626
19		ε_b	Strain (engineering) at break [%]	4,72	4,58	4,44	4,58	4,30	4,52
20		σ_{true}	True stress [MPa]	34,728	34,844	36,830	35,288	34,046	35,147
21		$\varepsilon_{\text{true}}$	True strain [%]	4,61	4,48	4,34	4,48	4,21	4,42

Figure 470: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

16. April 2024

**P.17 Material: PLA x Infill pattern: Cubic
Infill density: 75%
Preparation temp.: -38 to -40 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -38°C to -40°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

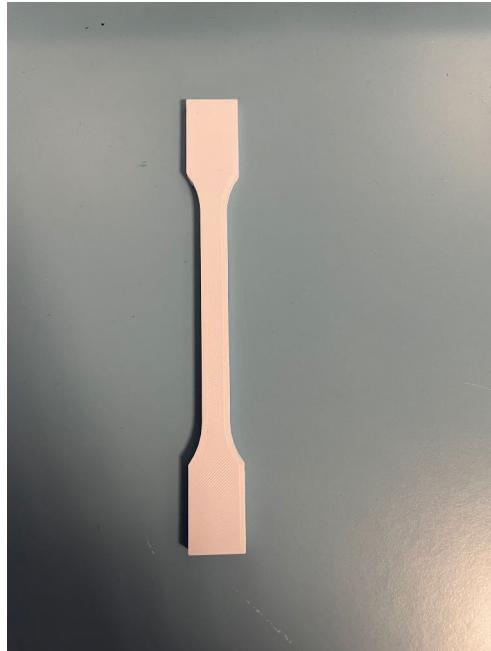
Pictures - before & after tensile test

Figure 471: Before tensile test 1 | STB

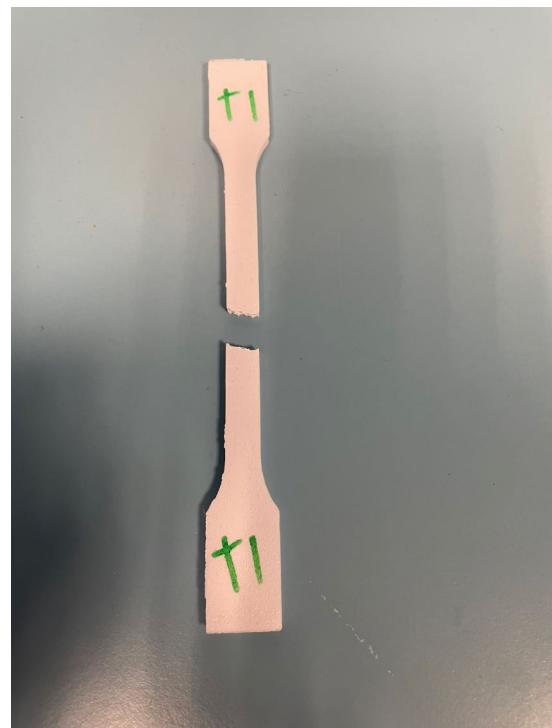


Figure 472: After tensile test 1 | STB

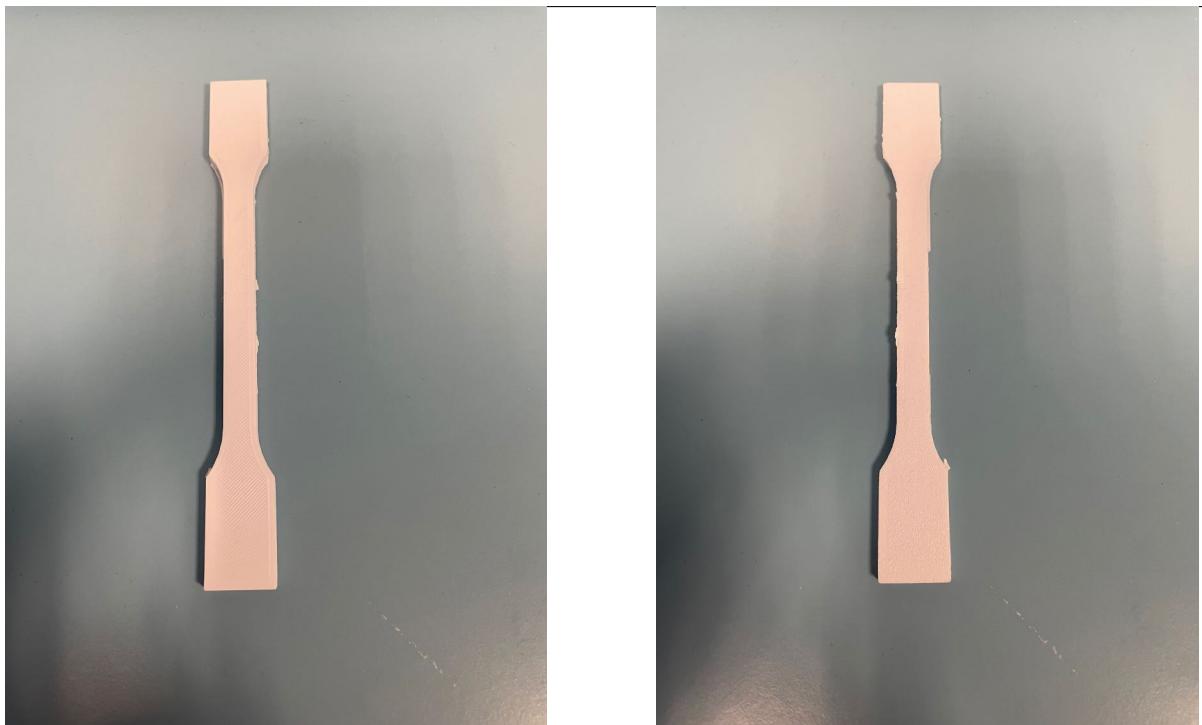


Figure 473: Before tensile test 2 | STB

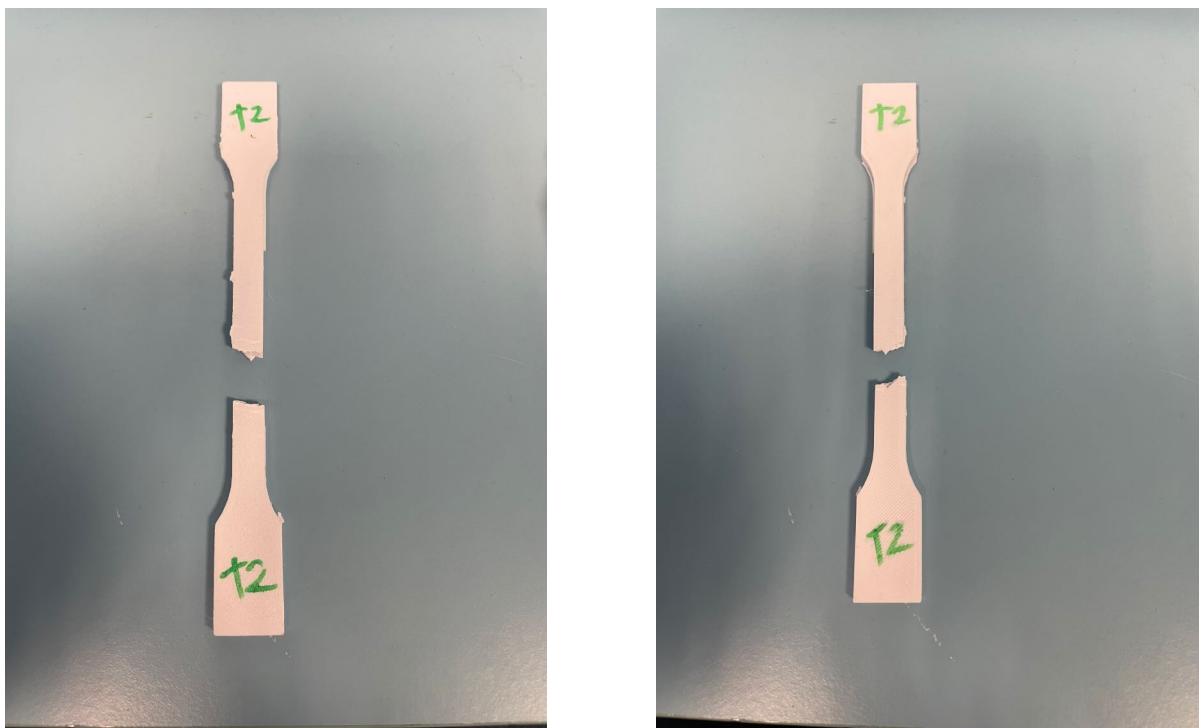


Figure 474: After tensile test 2 | STB

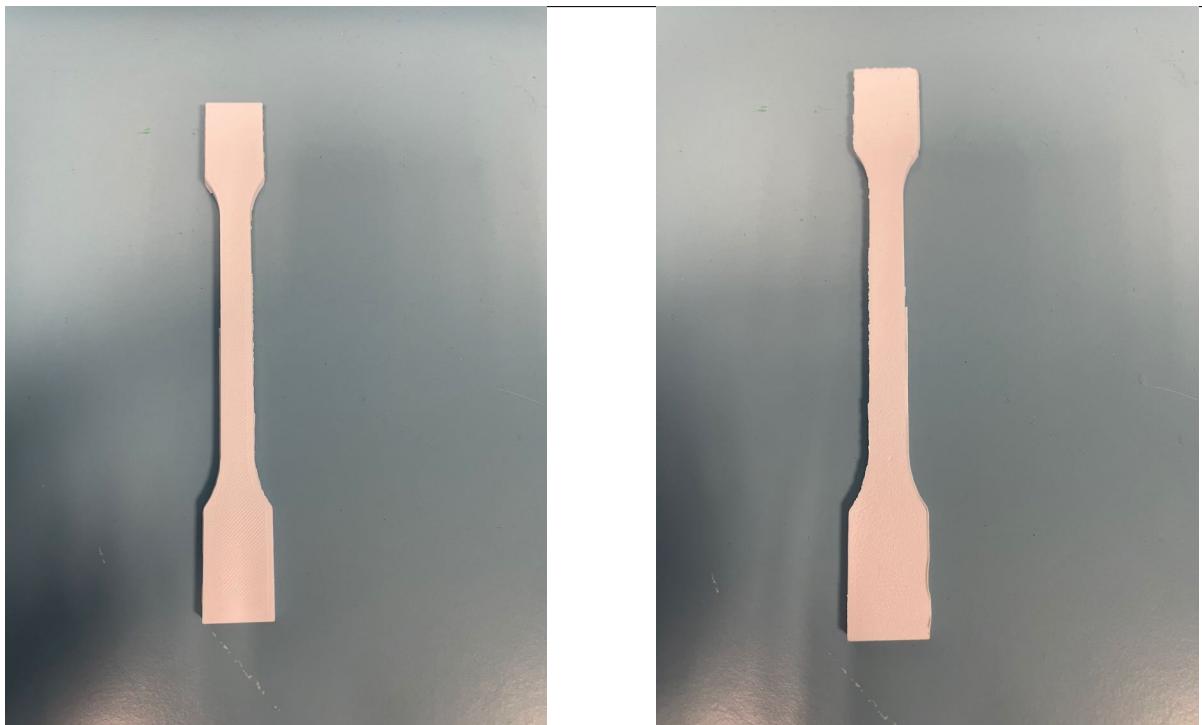


Figure 475: Before tensile test 3 | STB

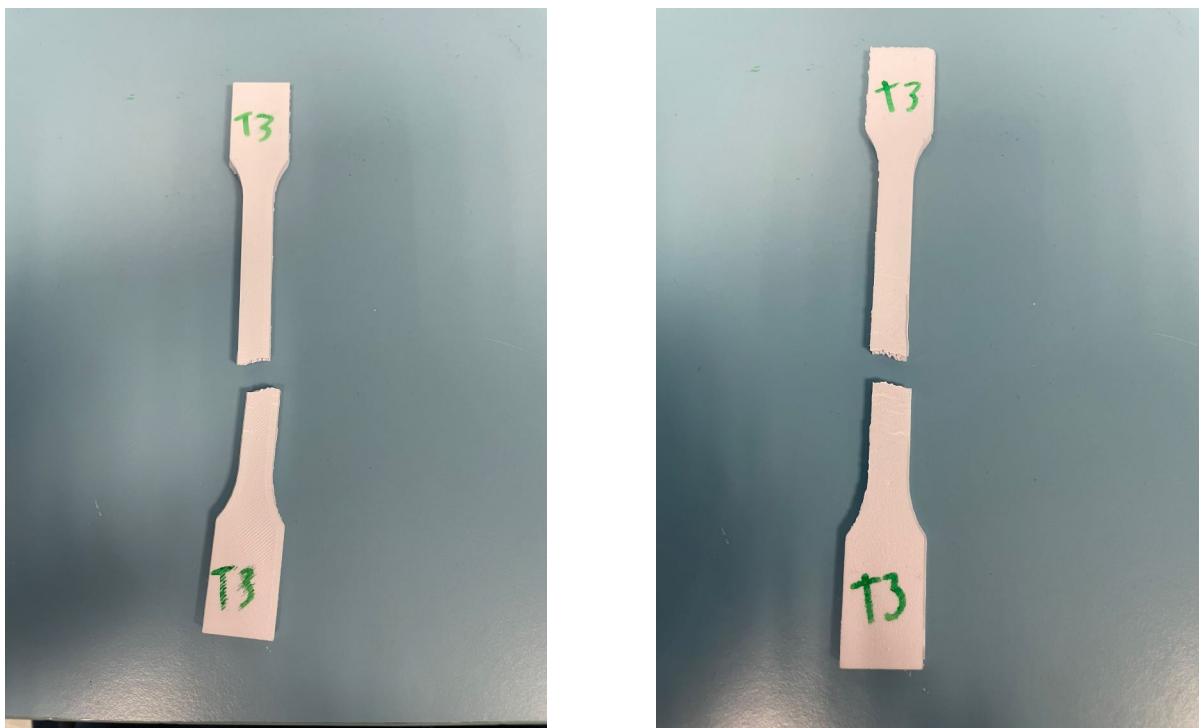


Figure 476: After tensile test 3 | STB

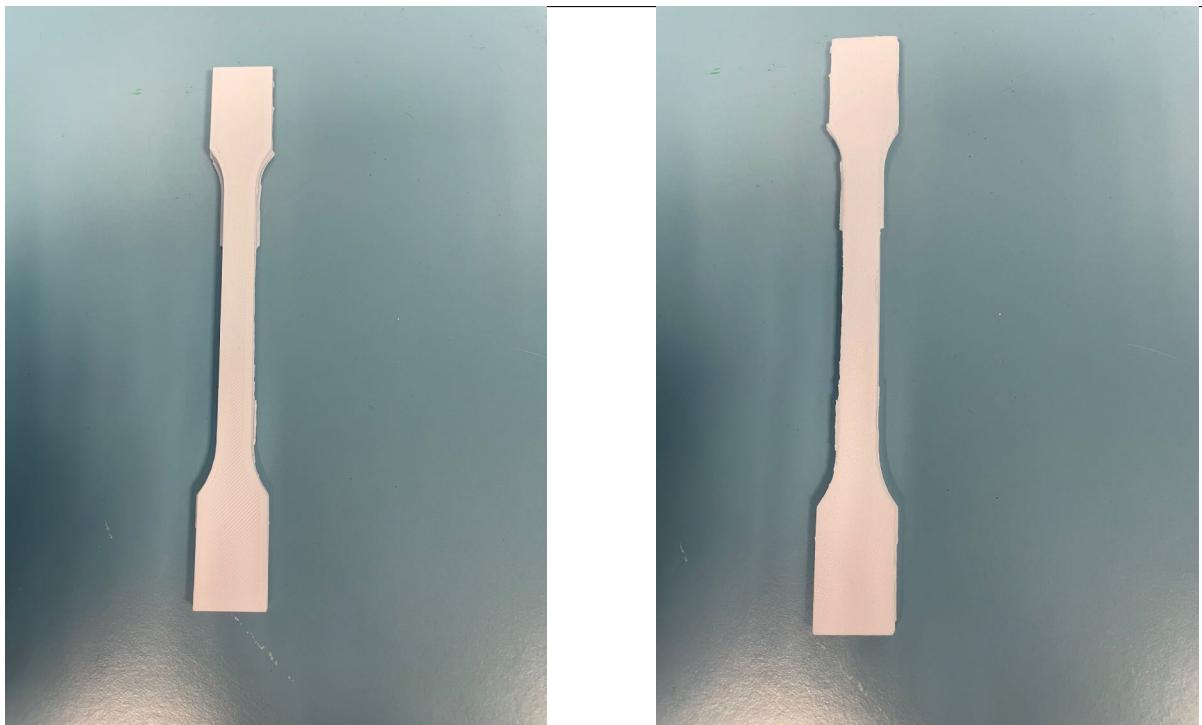


Figure 477: Before tensile test 4 | STB

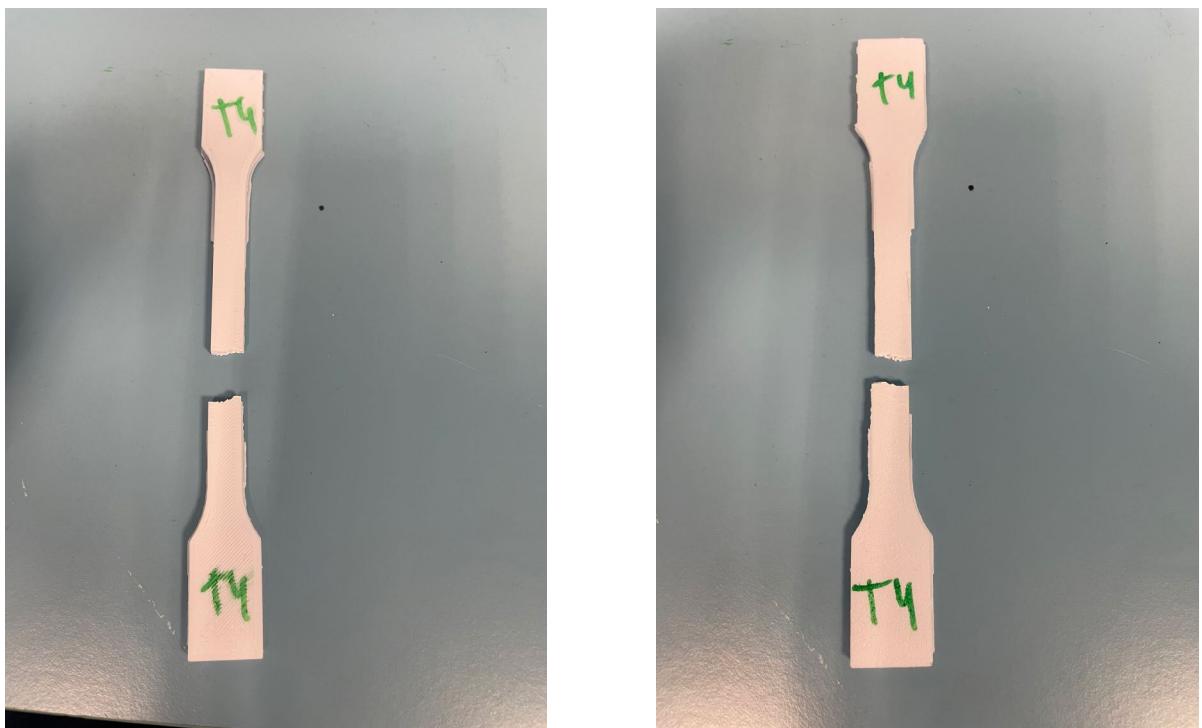


Figure 478: After tensile test 4 | STB

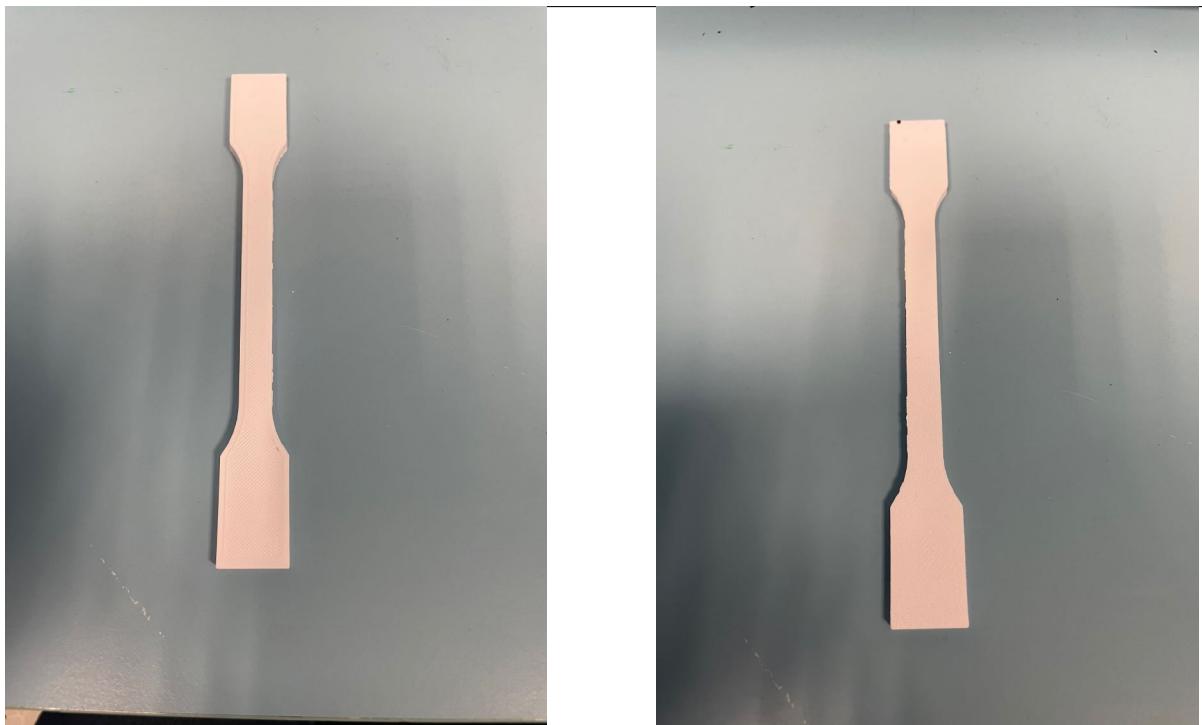


Figure 479: Before tensile test 5 | STB

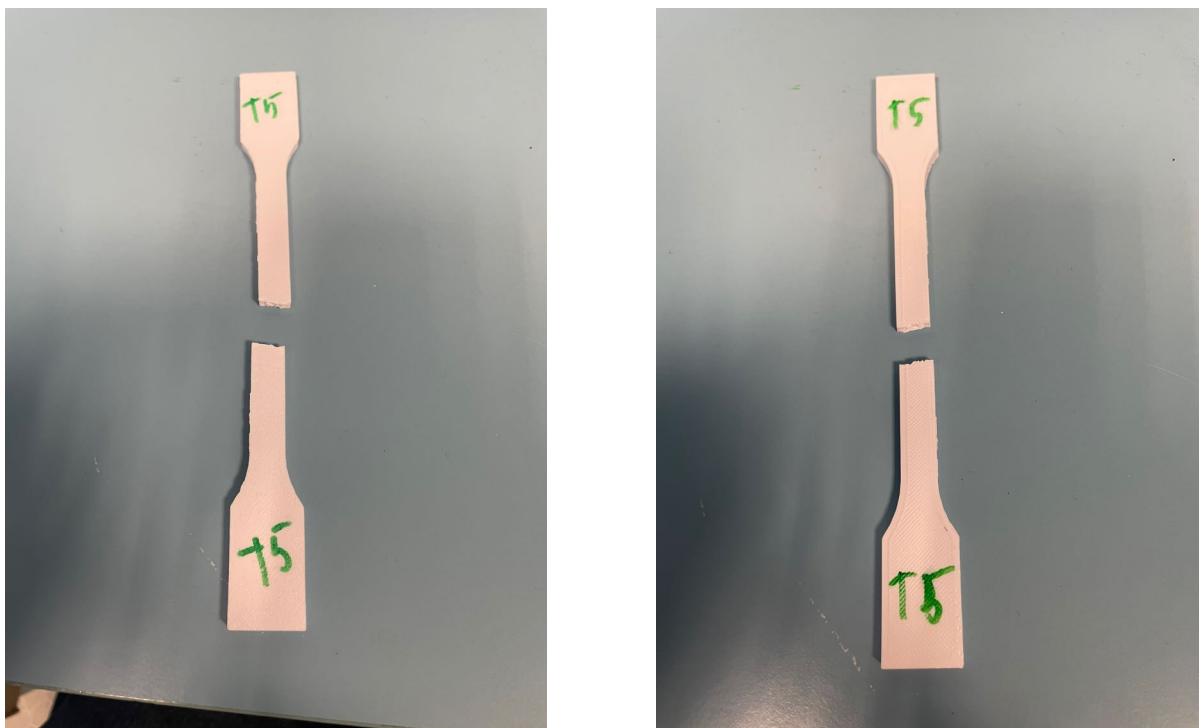


Figure 480: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 481

	A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values	
2	Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A	
3		h	Test staff thickness [mm]	3,87	3,82	3,84	3,85	3,82	3,84	
4		b_1	Width at narrow portion [mm]	9,97	10,06	10,05	10,00	9,97	10,01	
5	Value (Settings/test plan)	A	Cross-section [mm ²]	38,58	38,43	38,59	38,50	38,09	N/A	
6		L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A	
7		N/A	Amount of test	5	5	5	5	5	N/A	
8		v	Test Speed [mm/min]	60	60	60	60	60	N/A	
9	Proportional limit	σ_y	Nominal Stress [MPa]	8,483	8,772	8,101	8,884	8,414	8,531	
10		ε_{nom}	Nominal strain [%]	0,65	0,94	0,63	0,68	0,7	0,72	
11		σ_{true}	True stress [MPa]	8,539	8,855	8,152	8,944	8,473	8,593	
12		ε_{true}	True strain [%]	0,65	0,94	0,63	0,68	0,7	0,72	
13		E_t	Tensile modulus [MPa]	1302,37	931,06	1277,82	1303,25	1197,33	1202,37	
14	UTS	S_{ut}	Ultimate tensile strenght [MPa]	40,334	39,145	39,183	38,971	39,910	39,509	
15		ε_{nom}	Nominal strain [%]	5,33	5,11	4,89	4,71	4,92	4,99	
16		σ_{true}	True stress [MPa]	42,482	41,146	41,101	40,807	41,872	41,482	
17		ε_{true}	True strain [%]	5,19	4,99	4,78	4,60	4,80	4,87	
18	Fracture	σ_b	Stress at break [MPa]	38,073	35,83	35,374	37,24	38,16	36,935	
19		ε_b	Strain (engineering) at break [%]	5,74	5,71	5,68	4,99	5,28	5,48	
20		σ_{true}	True stress [MPa]	40,259	37,875	37,382	39,098	40,177	38,958	
21		ε_{true}	True strain [%]	5,58	5,55	5,52	4,87	5,15	5,33	

Figure 481: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

18. April 2024

**P.18 Material: PLA x Infill pattern: Cubic
Infill density: 75%
Preparation temp.: 20 to 24 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed room for 19 hours with the *preparation temperature* 20°C to 24°C. The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C.

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test



Figure 482: Before tensile test 1 | STB



Figure 483: After tensile test 1 | STB

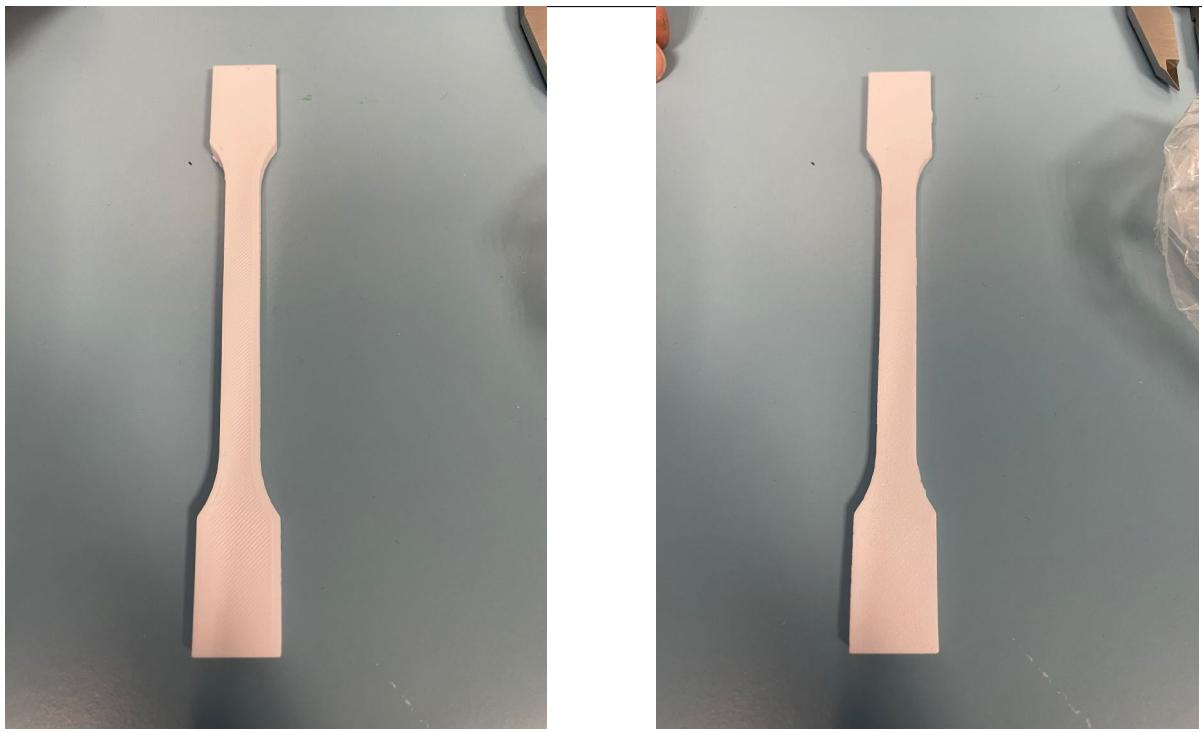


Figure 484: Before tensile test 2 | STB

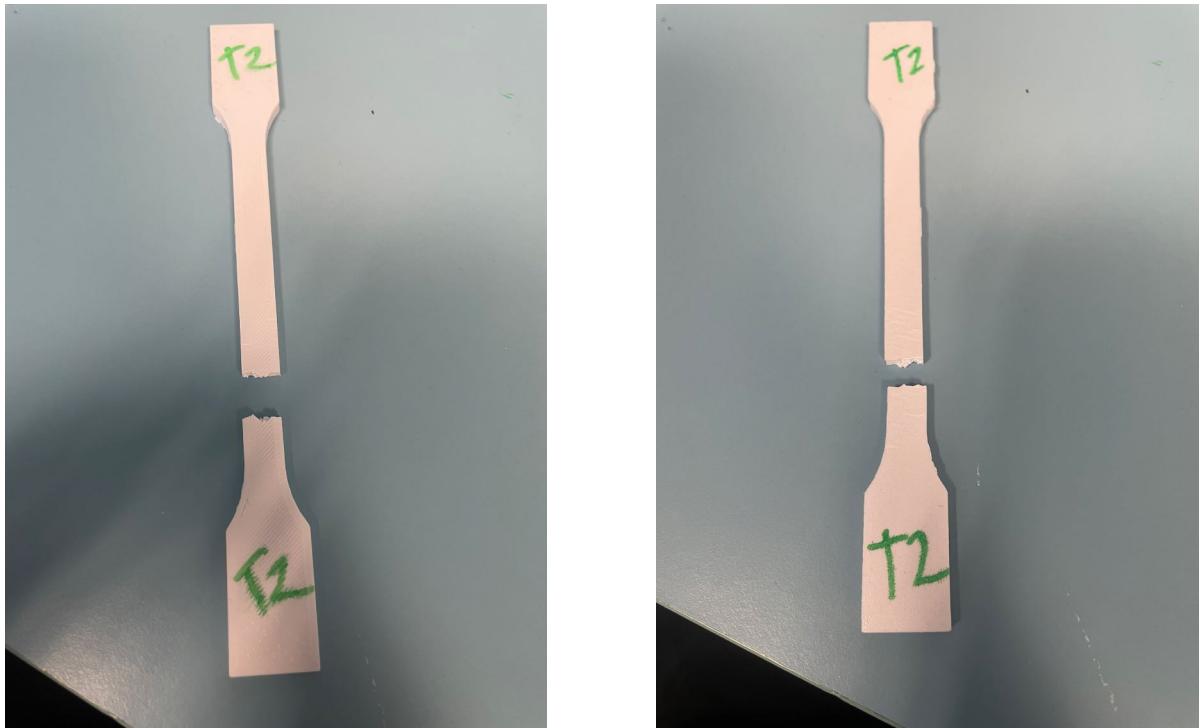


Figure 485: After tensile test 2 | STB

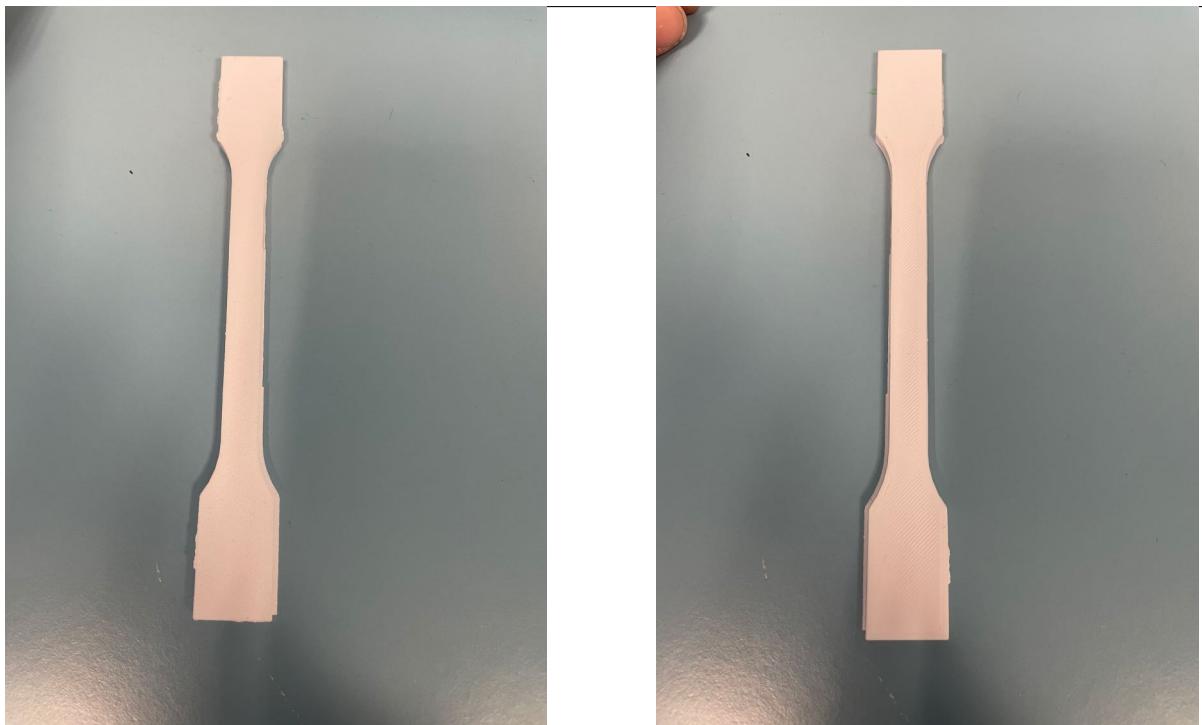


Figure 486: Before tensile test 3 | STB

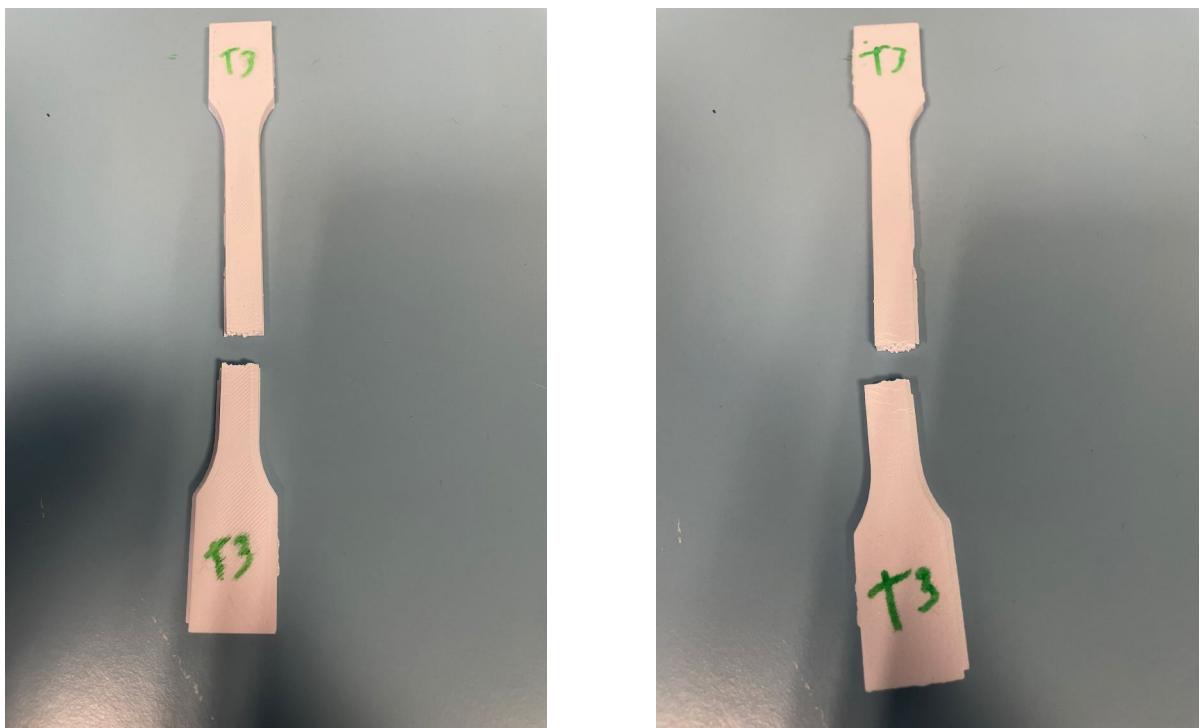


Figure 487: After tensile test 3 | STB

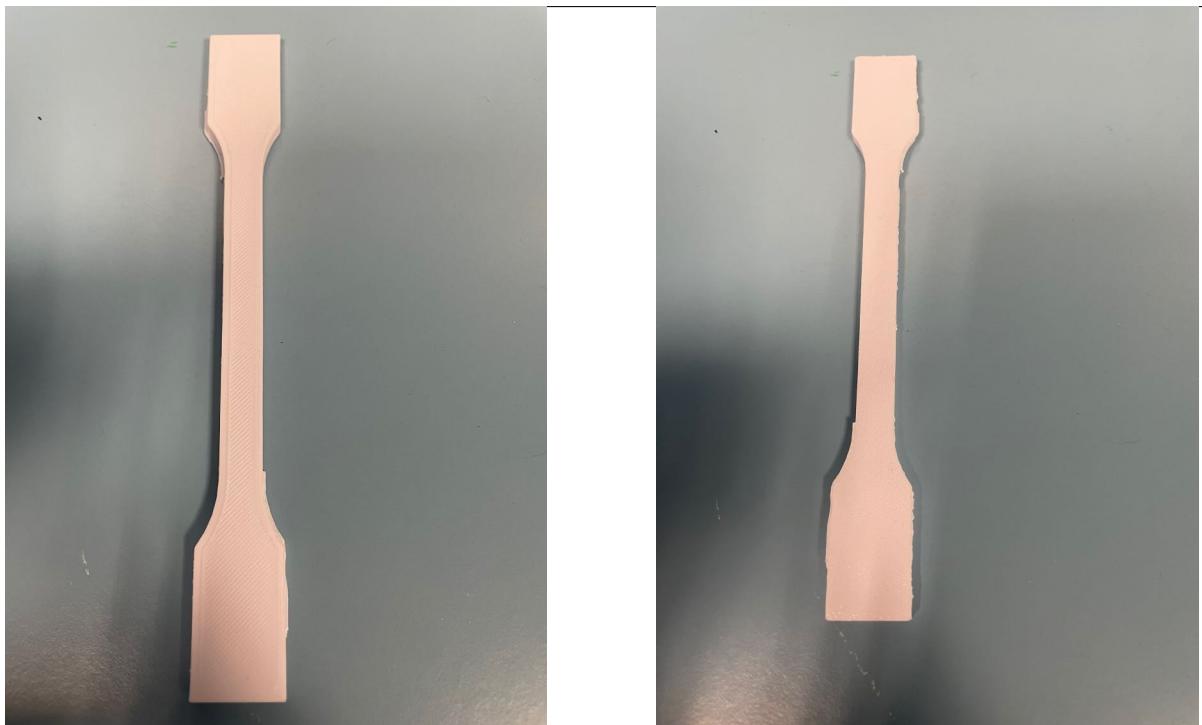


Figure 488: Before tensile test 4 | STB

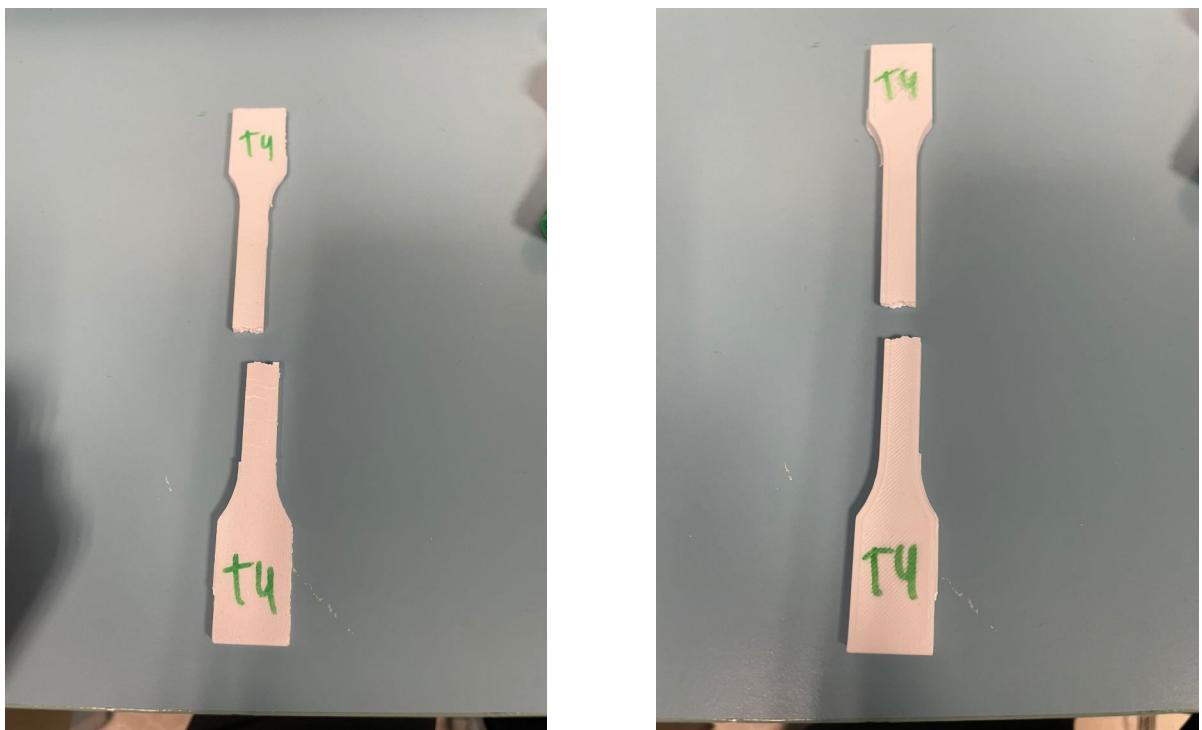


Figure 489: After tensile test 4 | STB

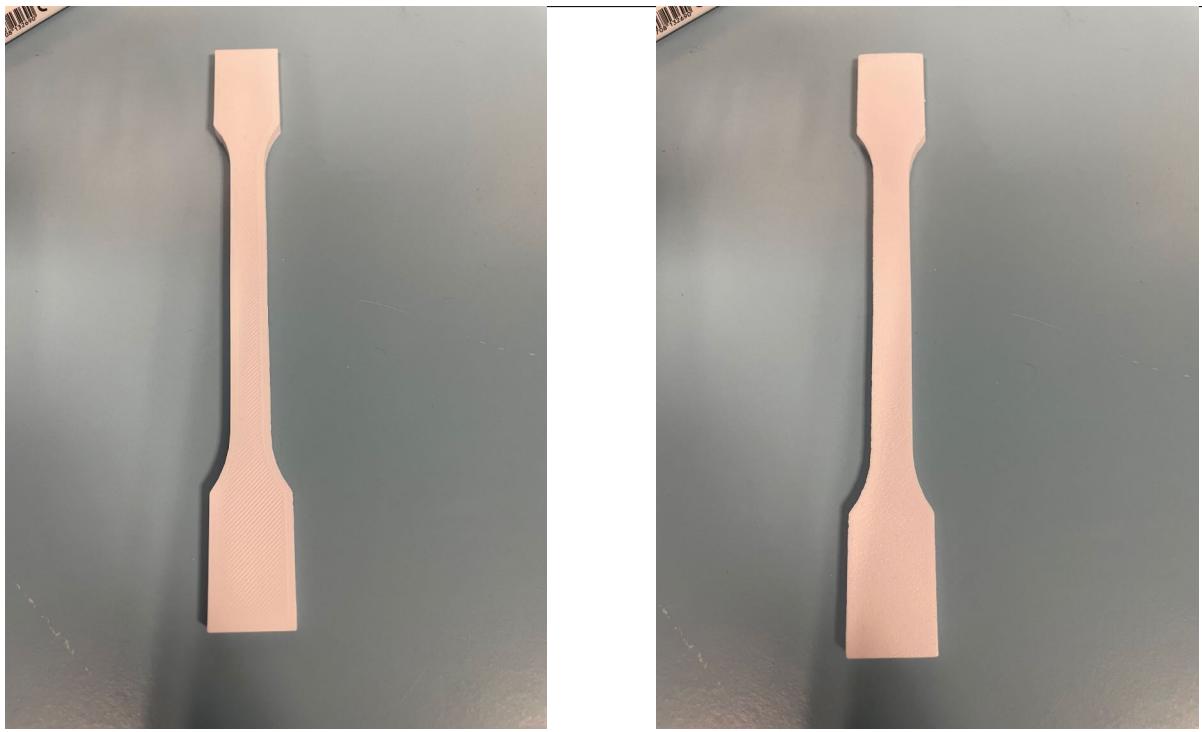


Figure 490: Before tensile test 5 | STB

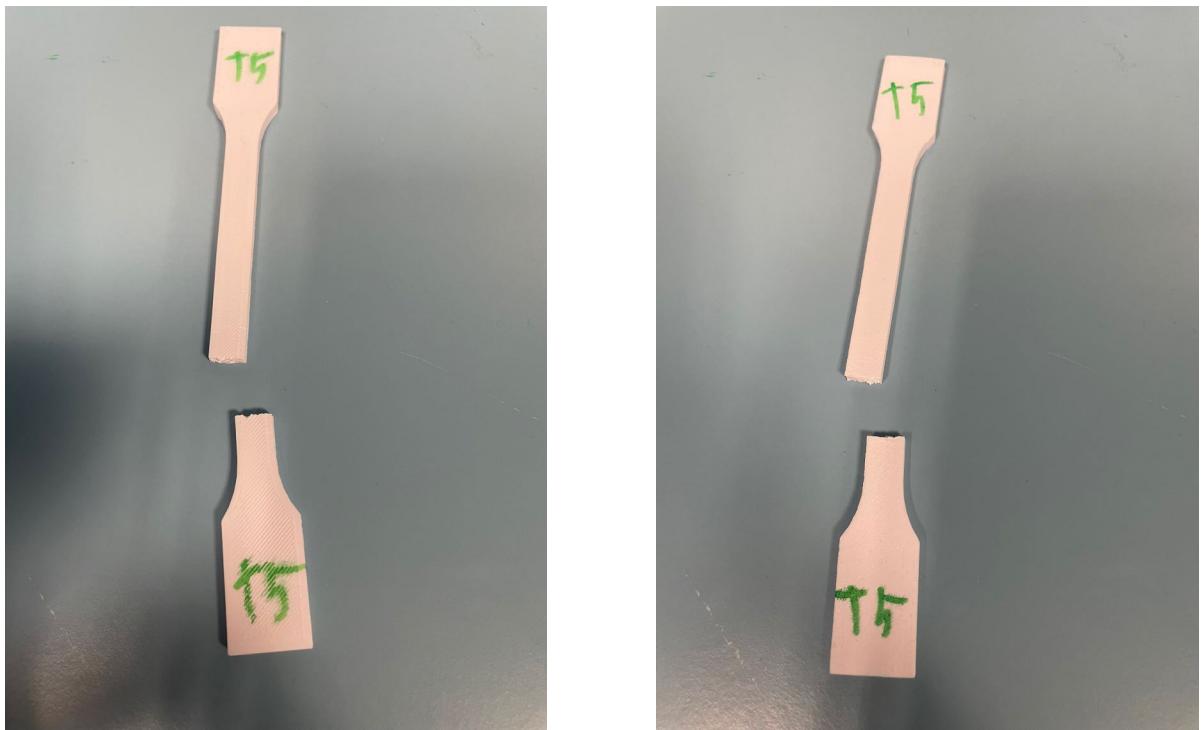


Figure 491: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 492

A	B	C	D	E	F	G	H	I	J
	Units	Description		Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
1	DATA	Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	N/A
2			h	Test staff thickness [mm]	3,82	3,82	3,84	3,92	3,86 3,85
3			b_1	Width at narrow portion [mm]	10,05	10,00	10,03	9,92	9,96 9,99
4		Value (Settings/test plan)	A	Cross-section [mm ²]	38,39	38,20	38,52	38,89	38,45 N/A
5			L	Gripping distance	109,00	109,00	109,00	109,00	109,00 N/A
6			N/A	Amount of test	5	5	5	5	5 N/A
7			v	Test Speed [mm/min]	60	60	60	60	60 N/A
8		Proportional limit	σ_y	Nominal Stress [MPa]	8,654	6,439	7,455	7,031	8,310 7,578
9			ε_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	0
10			σ_{true}	True stress [MPa]	8,653	6,439	7,455	7,031	8,310 7,578
11			ε_{true}	True strain [%]	N/A	N/A	N/A	N/A	0
12			E_t	Tensile modulus [MPa]	N/A	N/A	N/A	766279,13	N/A 153255,83
13	UTS		S_ut	Ultimate tensile strenght [MPa]	38,597	39,200	38,396	38,735	39,383 38,862
14			ε_{nom}	Nominal strain [%]	4,26	5,04	4,86	4,77	4,77 4,74
15			σ_{true}	True stress [MPa]	40,239	41,176	40,261	40,581	41,26 40,703
16			ε_{true}	True strain [%]	4,17	4,92	4,74	4,66	4,66 4,63
17	Fracture		σ_b	Stress at break [MPa]	37,167	35,865	35,750	35,055	37,089 36,185
18			ε_b	Strain (engineering) at break [%]	4,58	5,87	5,27	5,50	5,18 5,28
19			σ_{true}	True stress [MPa]	38,869	37,969	37,633	36,984	39,009 38,093
20			ε_{true}	True strain [%]	4,48	5,70	5,14	5,36	5,05 5,15

Figure 492: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

18. April 2024

**P.19 Material: PLA x Infill pattern: Gyroid
Infill density: 50%
Preparation temp.: -18 to -20 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* -18°C to -20°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

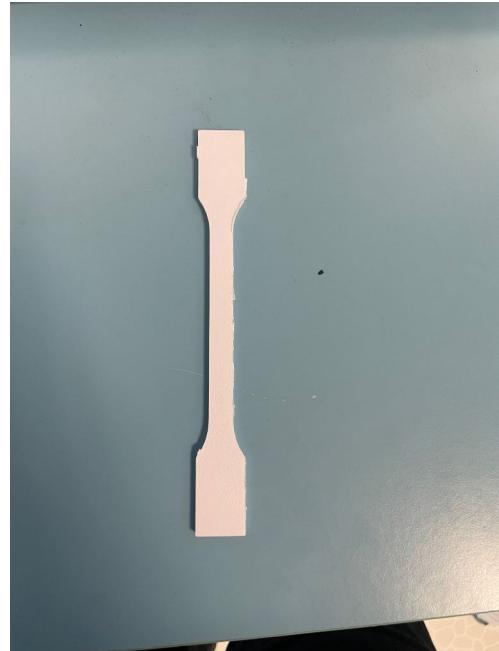
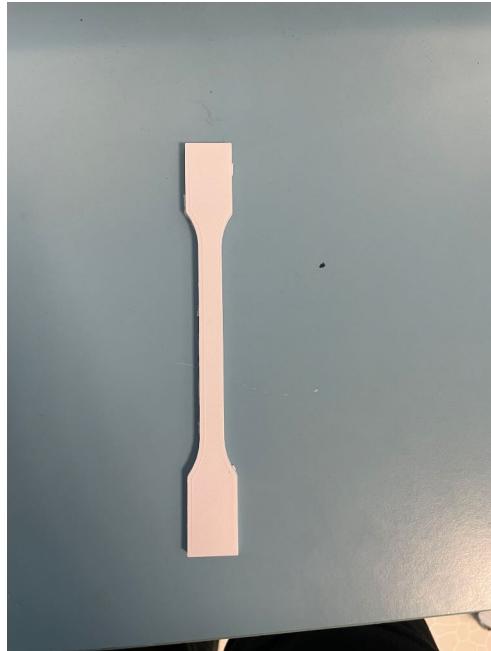
Pictures - before & after tensile test

Figure 493: Before tensile test 1 | STB

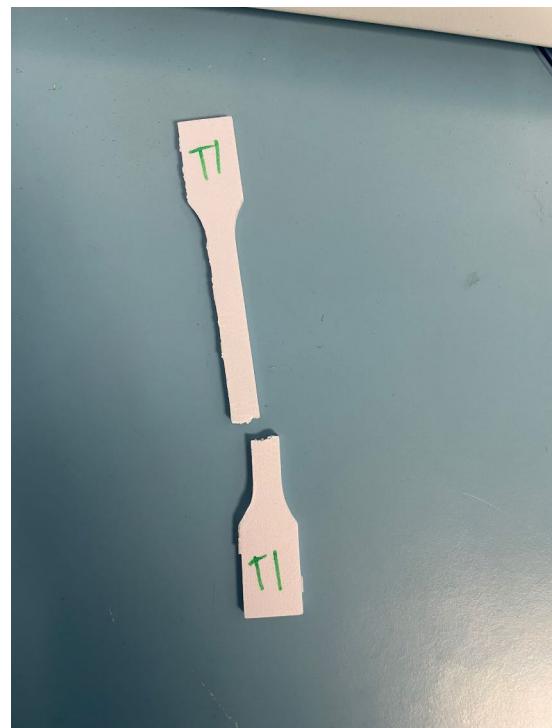
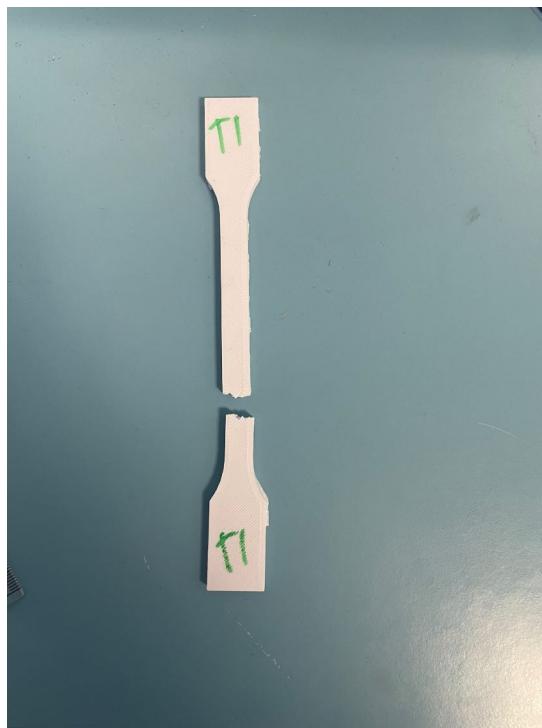


Figure 494: After tensile test 1 | STB

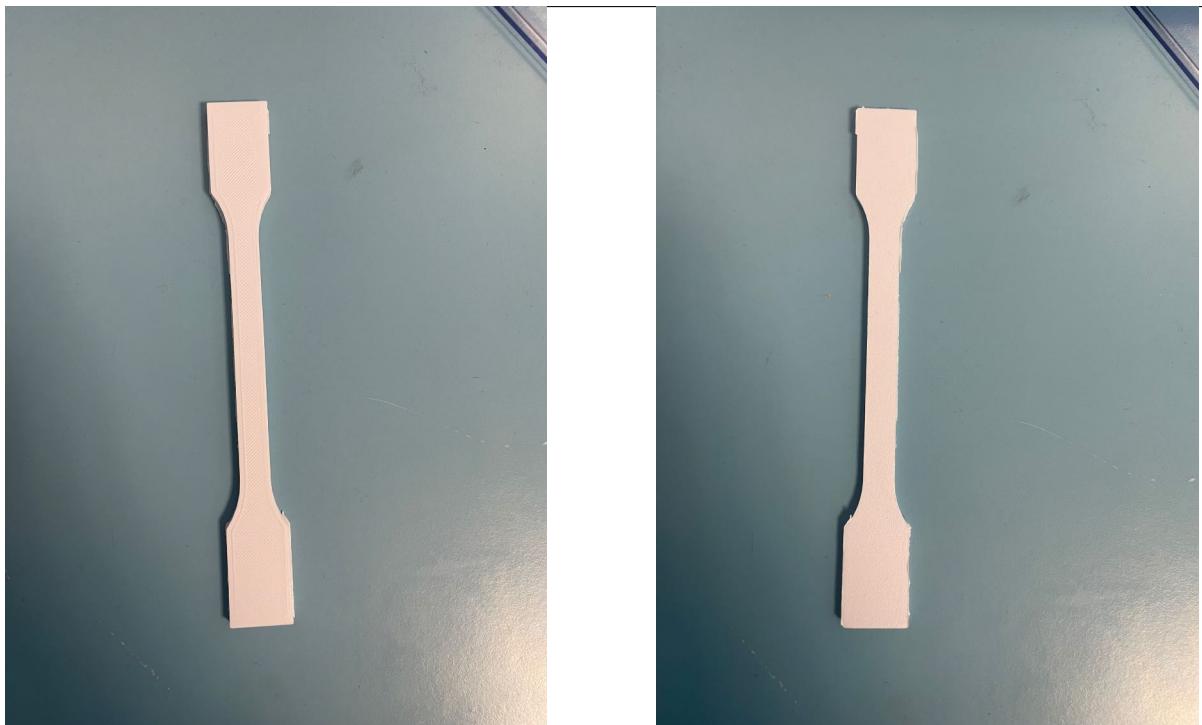


Figure 495: Before tensile test 2 | STB

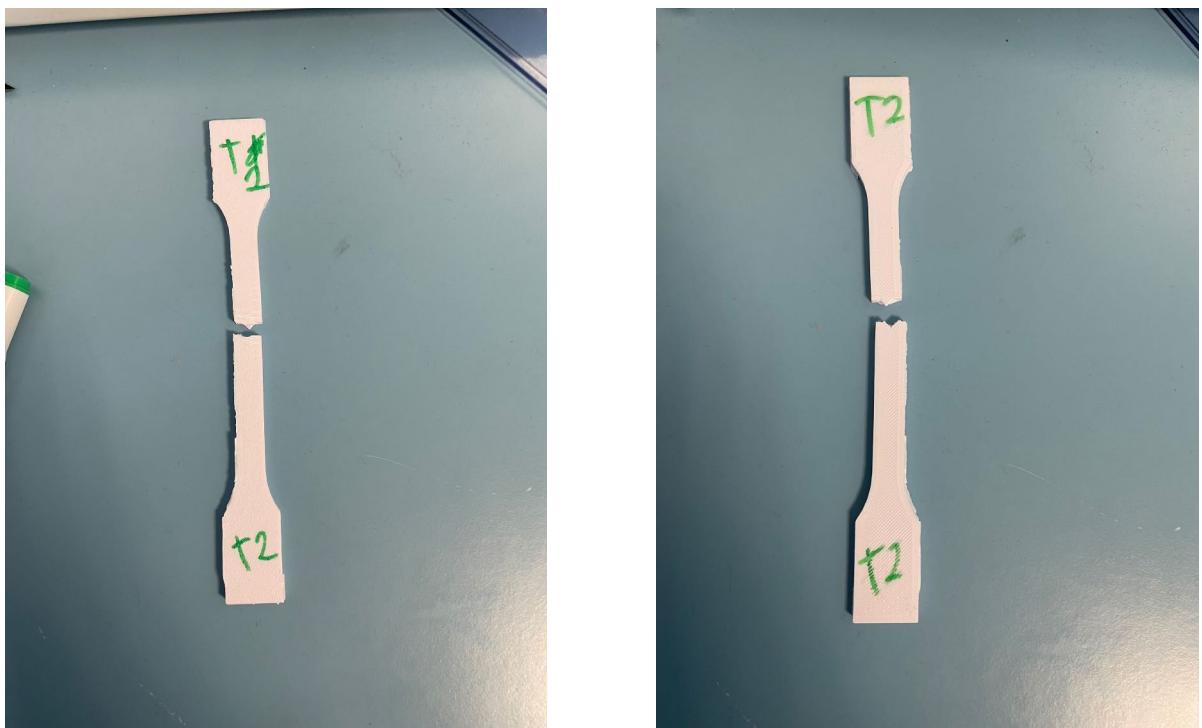


Figure 496: After tensile test 2 | STB

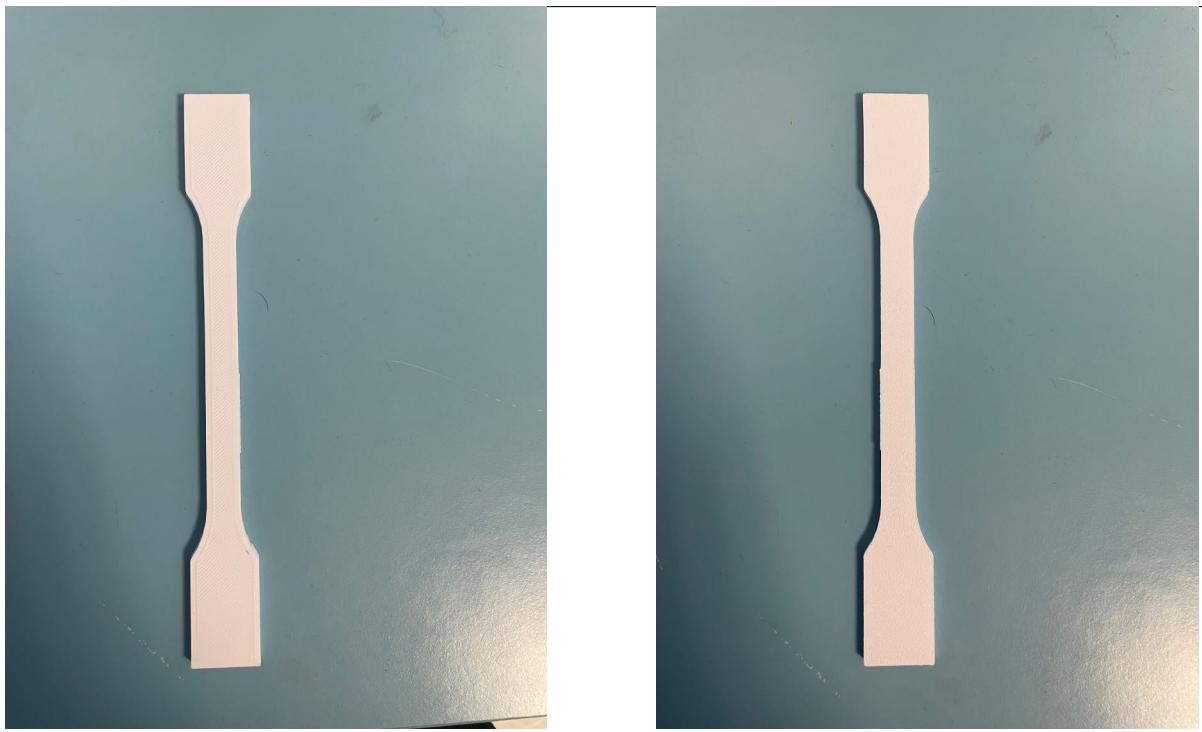


Figure 497: Before tensile test 3 | STB

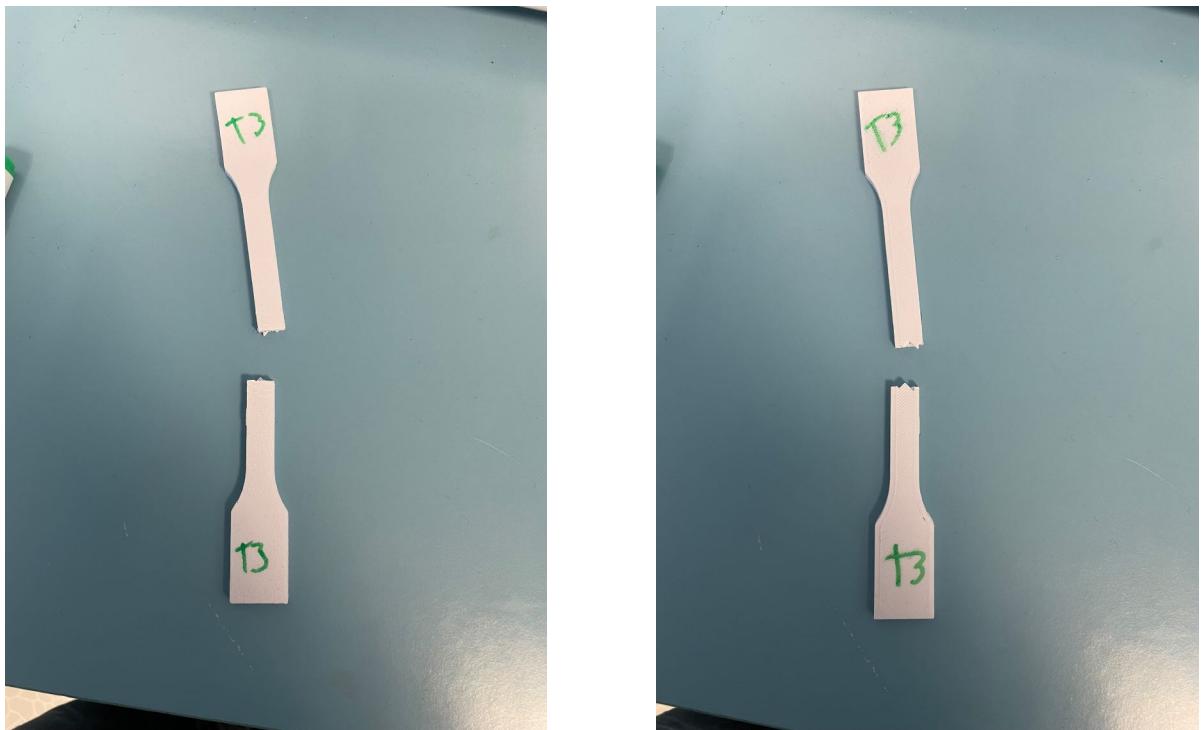


Figure 498: After tensile test 3 | STB



Figure 499: Before tensile test 4 | STB

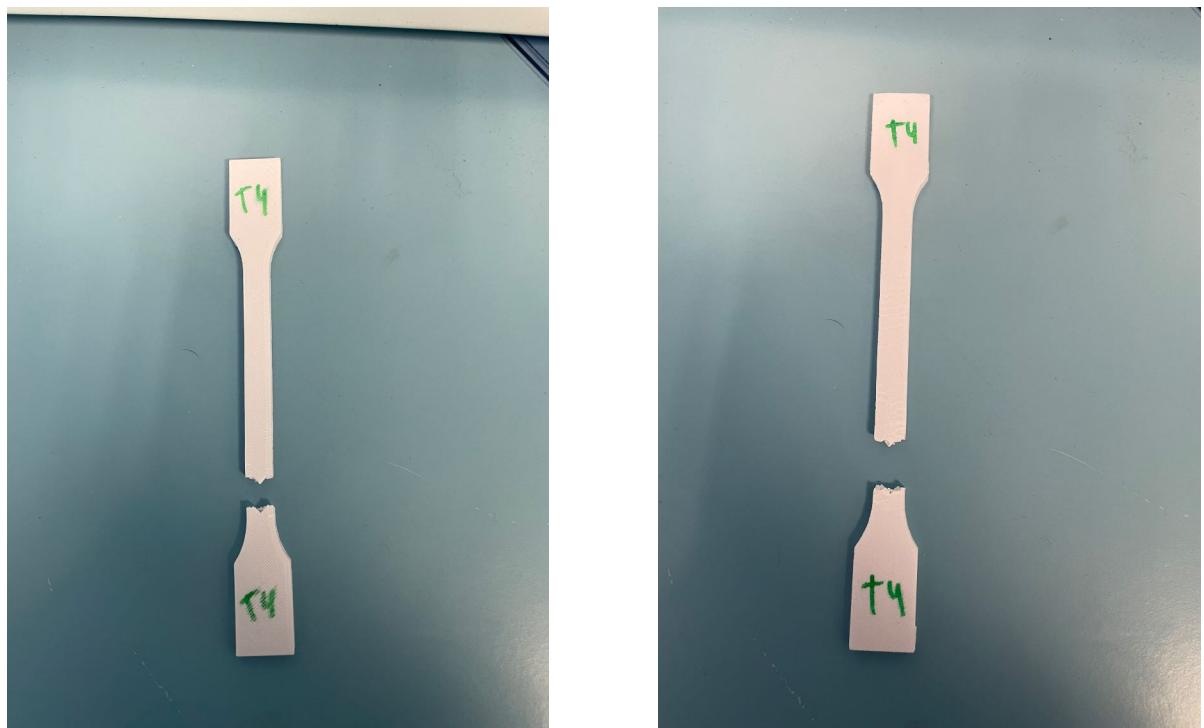


Figure 500: After tensile test 4 | STB

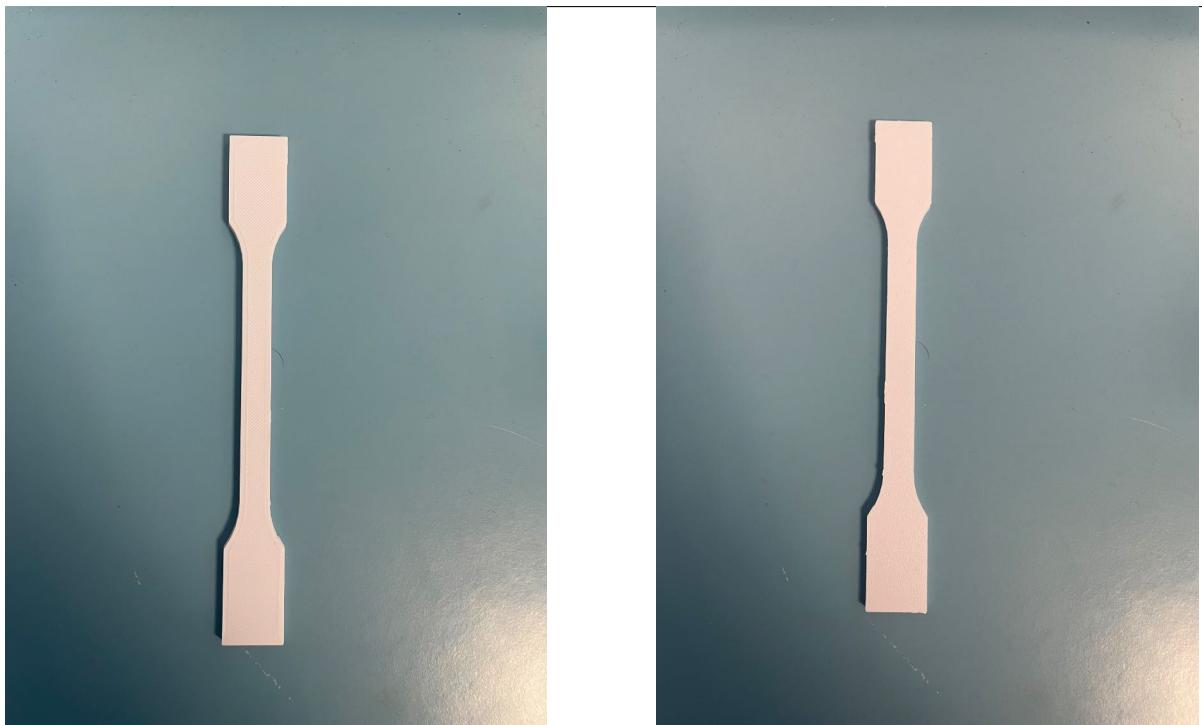


Figure 501: Before tensile test 5 | STB

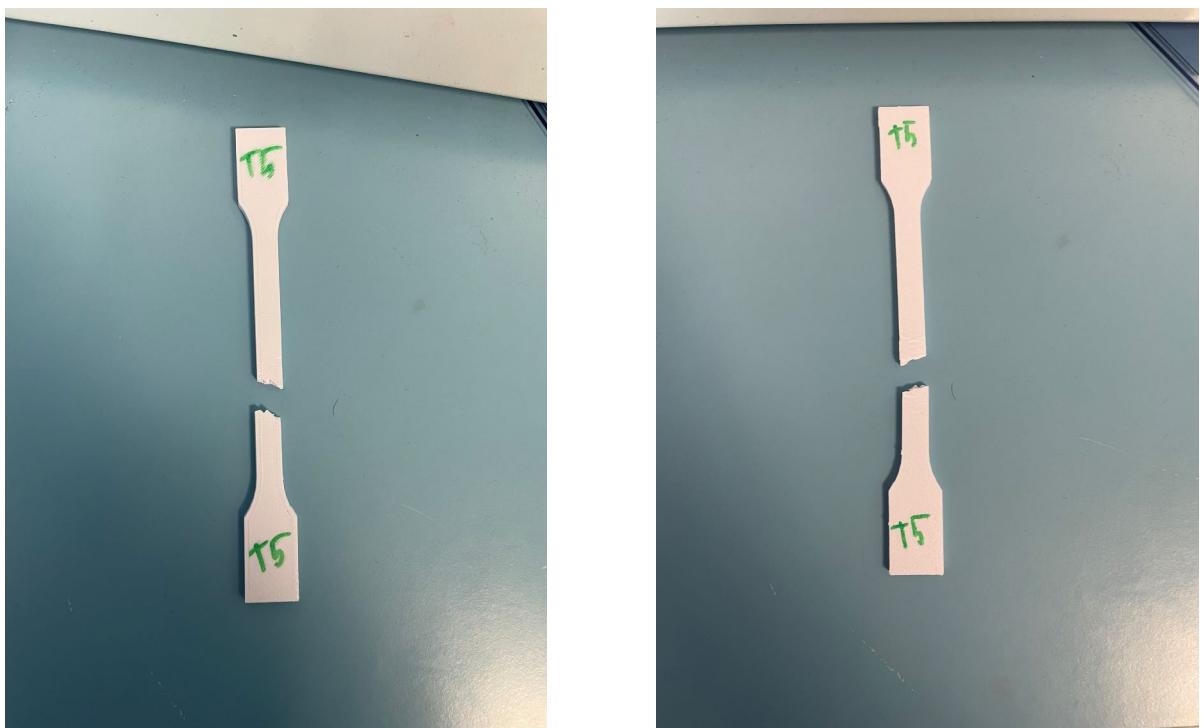


Figure 502: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 503.

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
	Initial values	L_0	Chosen start length [mm]	108,00	108,00	108,00	108,00	108,00	N/A
		h	Test staff thickness [mm]	3,85	3,86	3,85	3,89	3,84	3,86
		b_1	Width at narrow portion [mm]	10,05	10,09	10,03	10,03	10,08	10,06
	Value (Settings/test plan)	A	Cross-section [mm ²]	38,69	38,95	38,62	39,02	38,71	N/A
		L	Gripping distance	108,00	108,00	108,00	108,00	108,00	N/A
		N/A	Amount of test	5	5	5	5	5	N/A
		v	Test Speed [mm/min]	60	60	60	60	60	N/A
	Proportional limit	σ_y	Nominal Stress [MPa]	8,130	8,429	8,654	9,042	8,532	8,557
		ε_{nom}	Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0
		σ_{true}	True stress [MPa]	8,130	8,429	8,654	9,043	8,533	8,558
		ε_{true}	True strain [%]	N/A	N/A	N/A	N/A	N/A	0
	UTS	E_t	Tensile modulus [MPa]	878131,44	N/A	233662,03	244144,75	230374,08	317262,46
		S_{ut}	Ultimate tensile strength [MPa]	33,635	33,390	33,220	33,883	33,673	33,560
		ε_{nom}	Nominal strain [%]	4,35	4,02	4,21	3,75	5,00	4,27
		σ_{true}	True stress [MPa]	35,098	34,734	34,620	35,154	35,357	34,993
		ε_{true}	True strain [%]	4,26	3,95	4,13	3,68	4,88	4,18
	Fracture	σ_b	Stress at break [MPa]	26,468	28,735	29,820	28,358	27,774	28,231
		ε_b	Strain (engineering) at break [%]	5,65	5,04	4,91	5,51	6,67	5,56
		σ_{true}	True stress [MPa]	27,962	30,184	31,284	29,92	29,626	29,795
		ε_{true}	True strain [%]	5,49	4,92	4,79	5,36	6,46	5,40

Figure 503: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile est specimens was not rejected and replaced, but during the test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

16. April 2024

P.20 Material: PLA x Infill pattern: Gyroid
Infill density: 50%
Preparation temp.: -38 to -40 Celsius

The documentation and testing is based on sections/requirements from standards:

-
- NS-EN ISO 527-1:2019
 - NS-EN ISO 291:2008
 - NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* $-38^{\circ}C$ to $-40^{\circ}C$. The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between $20^{\circ}C$ and $24^{\circ}C$.

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test

Figure 504: Before tensile test 1 | STB

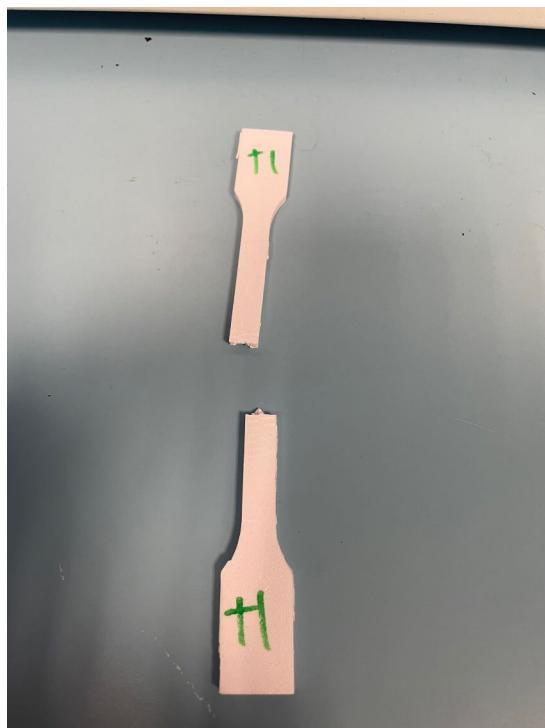


Figure 505: After tensile test 1 | STB

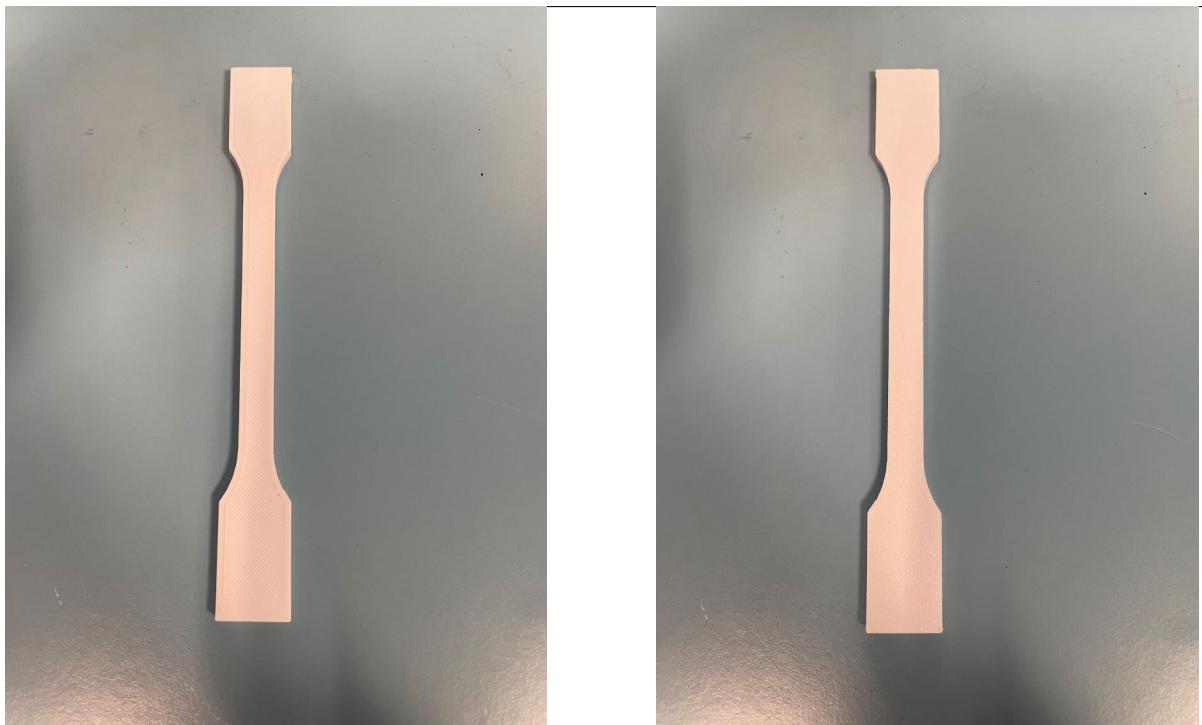


Figure 506: Before tensile test 2 | STB

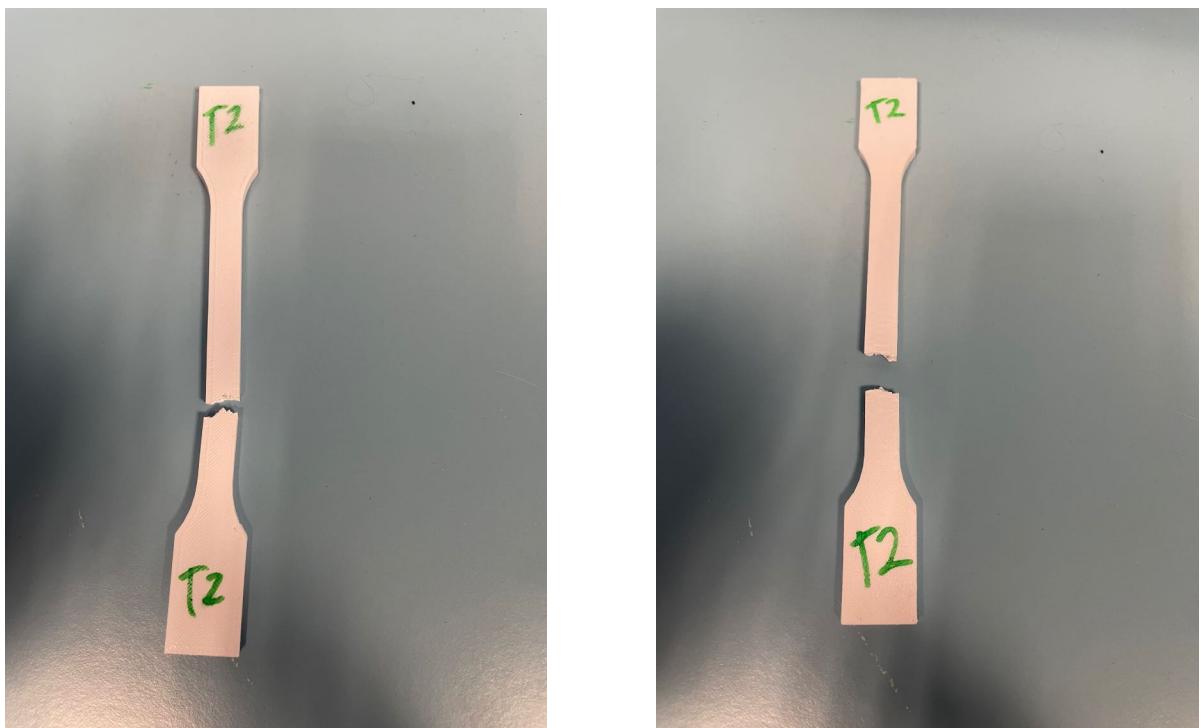


Figure 507: After tensile test 2 | STB

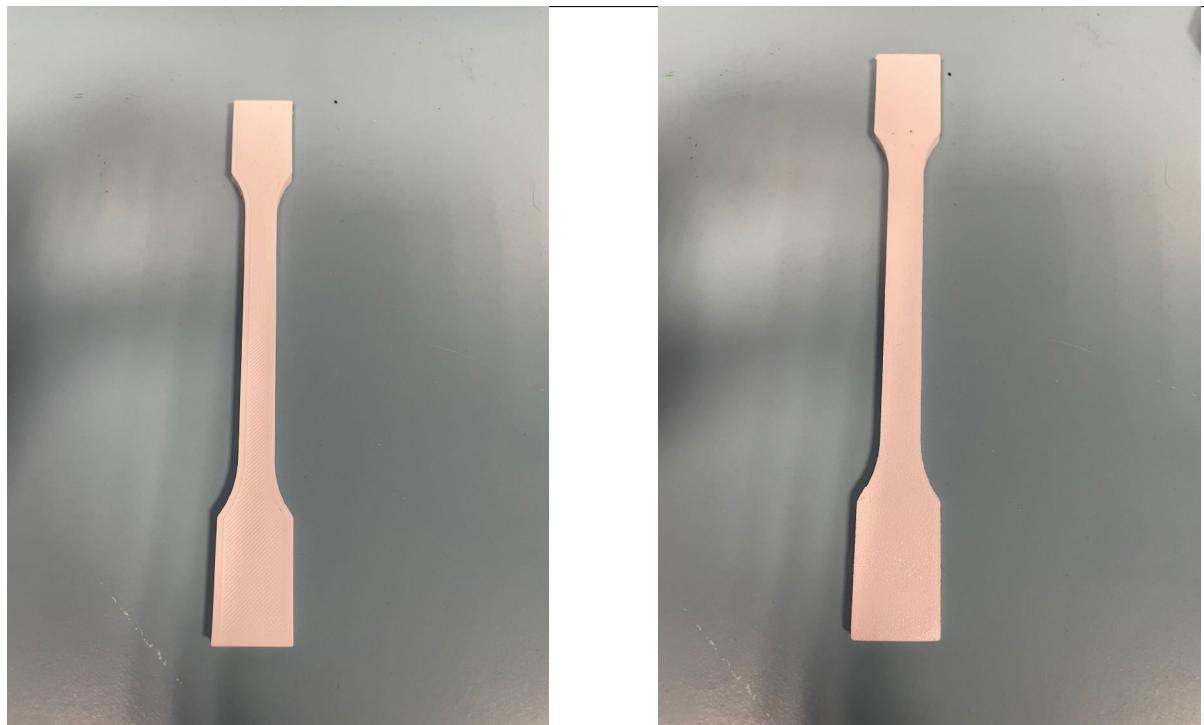


Figure 508: Before tensile test 3 | STB

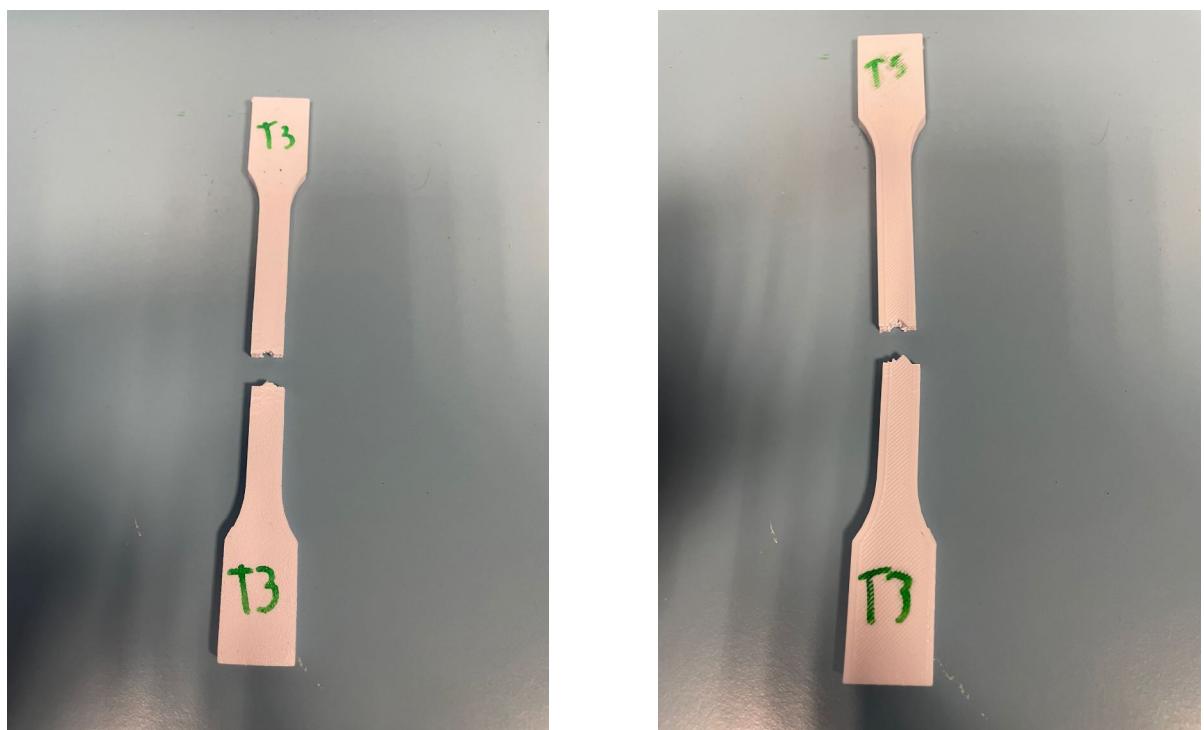


Figure 509: After tensile test 3 | STB

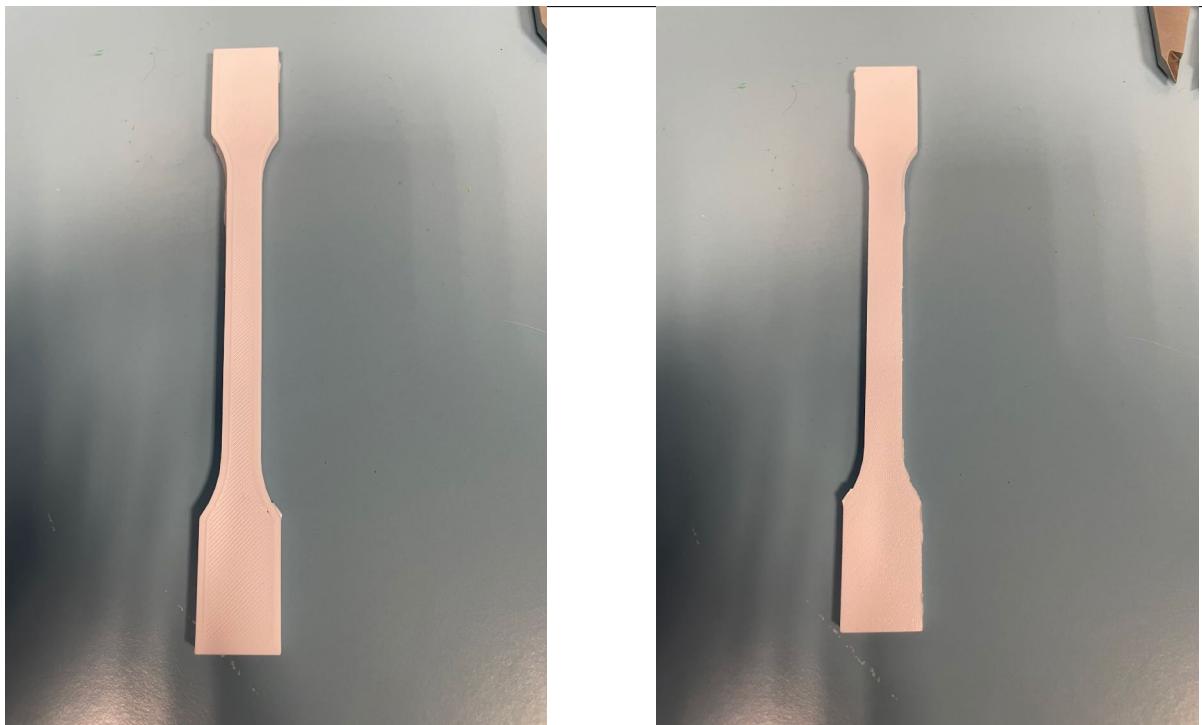


Figure 510: Before tensile test 4 | STB

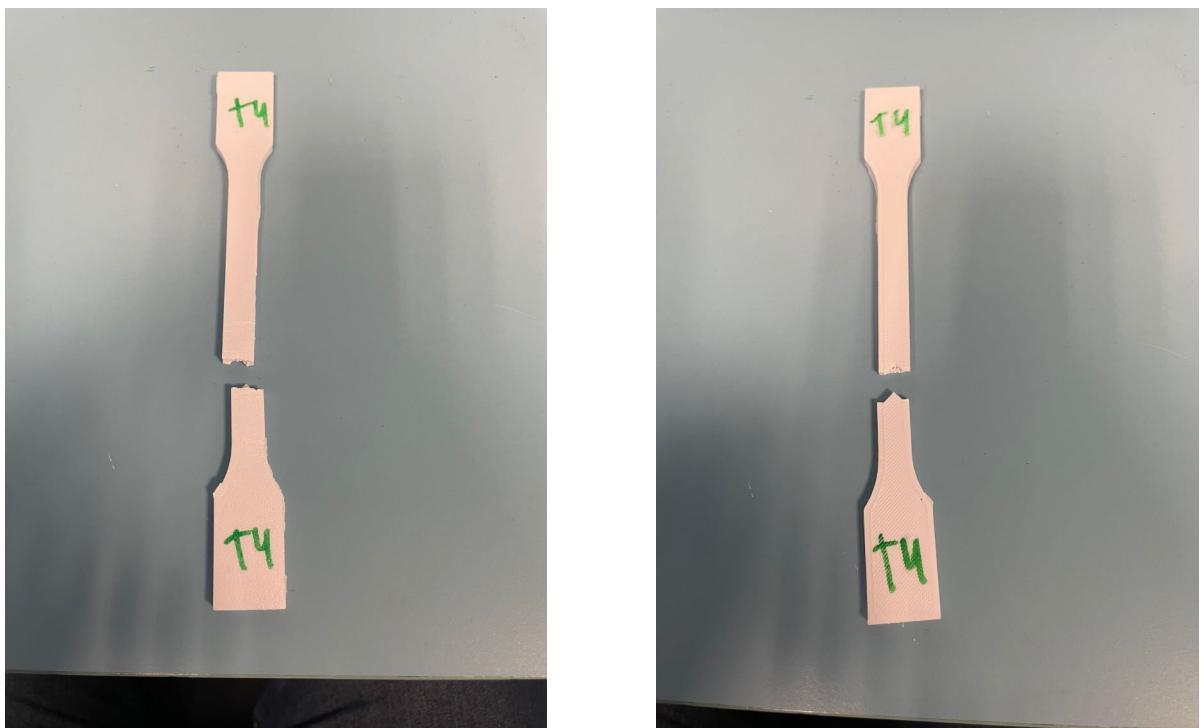


Figure 511: After tensile test 4 | STB

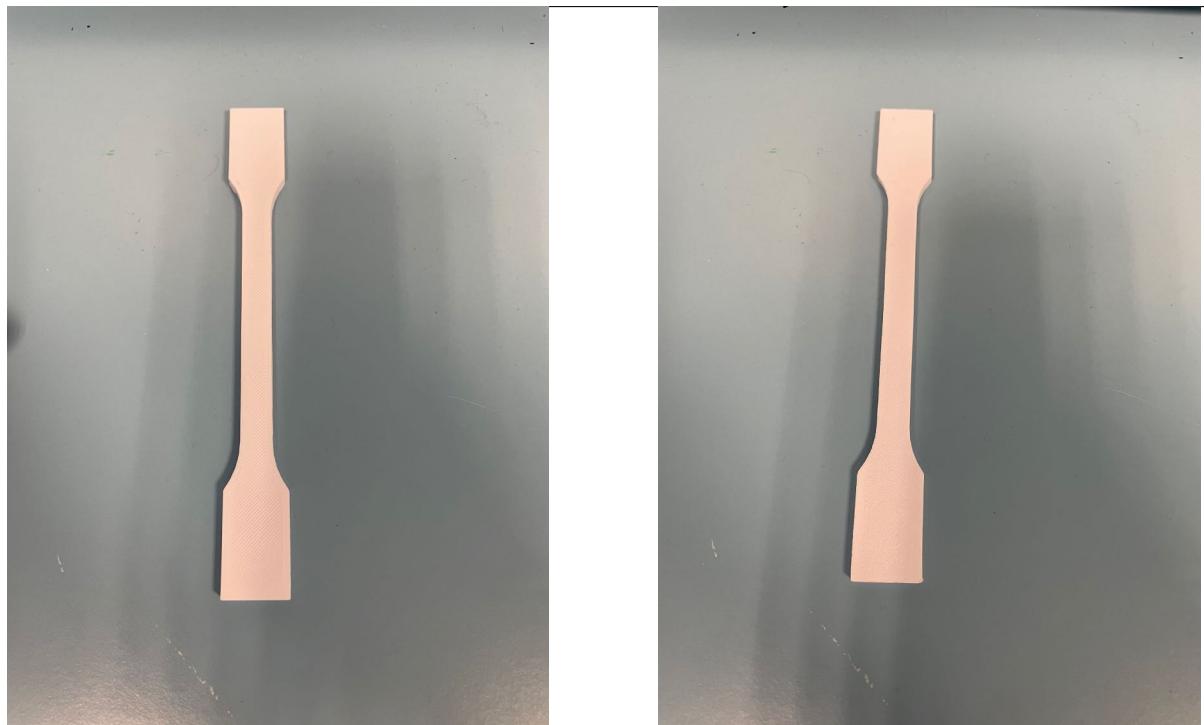


Figure 512: Before tensile test 5 | STB

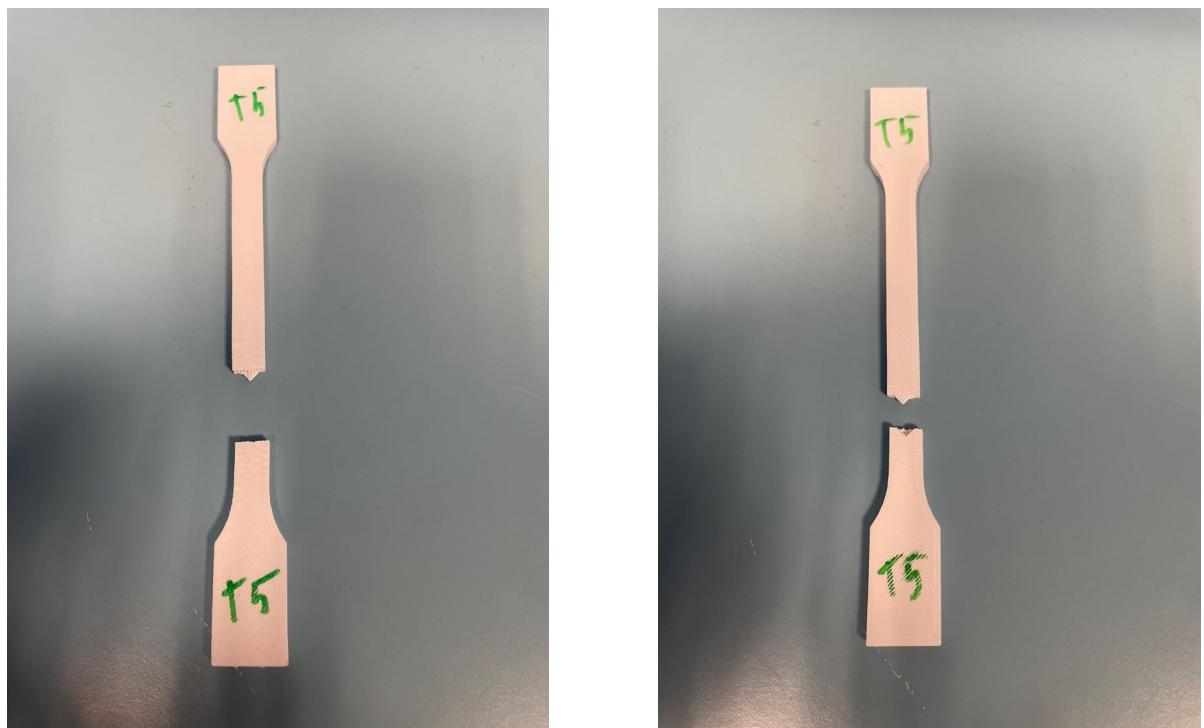


Figure 513: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 514

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
DATA	Initial values	L_0	Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
		h	Test staff thickness [mm]	3,85	3,84	3,90	3,85	3,83	3,85
		b_1	Width at narrow portion [mm]	9,96	9,94	9,97	9,97	9,96	9,96
Value (Settings/test plan)	A		Cross-section [mm ²]	38,35	38,17	38,88	38,38	38,15	N/A
		L	Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
	N/A		Amount of test	5	5	5	5	5	N/A
	v		Test Speed [mm/min]	60	60	60	60	60	N/A
Proportional limit	σ_y		Nominal Stress [MPa]	10,632	7,189	9,653	8,757	10,302	9,307
	ϵ_{nom}		Nominal strain [%]	1,04	0,68	0,72	0,34	0,5	0,656
	σ_{true}		True stress [MPa]	10,742	7,238	9,722	8,787	10,354	9,369
	ϵ_{true}		True strain [%]	1,03	0,67	0,72	0,34	0,5	0,652
	E_t		Tensile modulus [MPa]	1025,53	1061,79	1343,79	2586,81	2049,07	1613,40
UTS	S_ut		Ultimate tensile strength [MPa]	36,163	35,431	35,260	35,182	35,735	35,554
	ϵ_{nom}		Nominal strain [%]	4,66	5,63	4,47	4,74	4,35	4,77
	σ_{true}		True stress [MPa]	37,847	37,425	36,838	36,849	37,29	37,250
	ϵ_{true}		True strain [%]	4,55	5,47	4,38	4,62	4,26	4,66
Fracture	σ_b		Stress at break [MPa]	33,249	32,171	31,908	32,373	31,548	32,250
	ϵ_b		Strain (engineering) at break [%]	5,53	6,50	5,21	5,47	5,82	5,71
	σ_{true}		True stress [MPa]	35,088	34,261	33,570	34,145	33,384	34,090
	ϵ_{true}		True strain [%]	5,38	6,30	5,08	5,33	5,66	5,55

Figure 514: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the tensile test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

18. April 2024

**P.21 Material: PLA x Infill pattern: Gyroid
Infill density: 50%
Preparation temp.: 20 to 24 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed room for 19 hours with the *preparation temperature* 20°C to 24°C. The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C.

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test

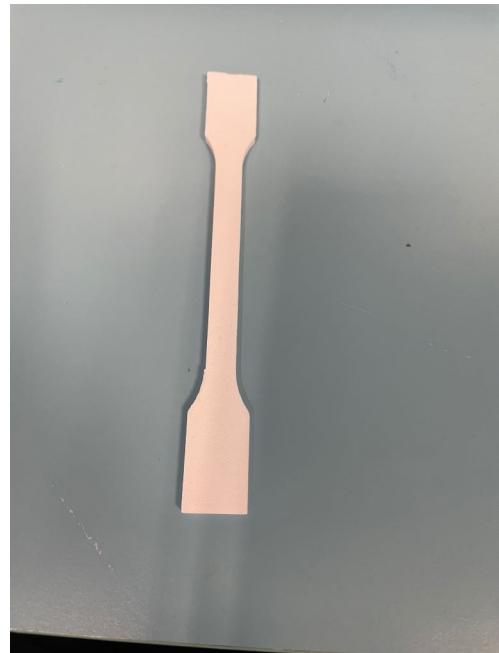
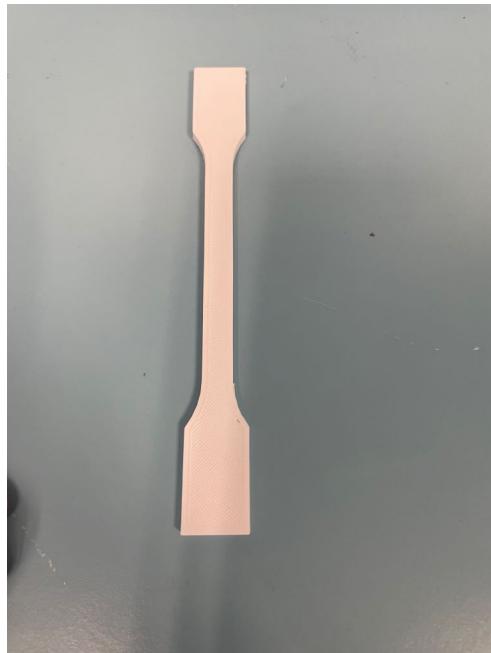


Figure 515: Before tensile test 1 | STB



Figure 516: After tensile test 1 | STB

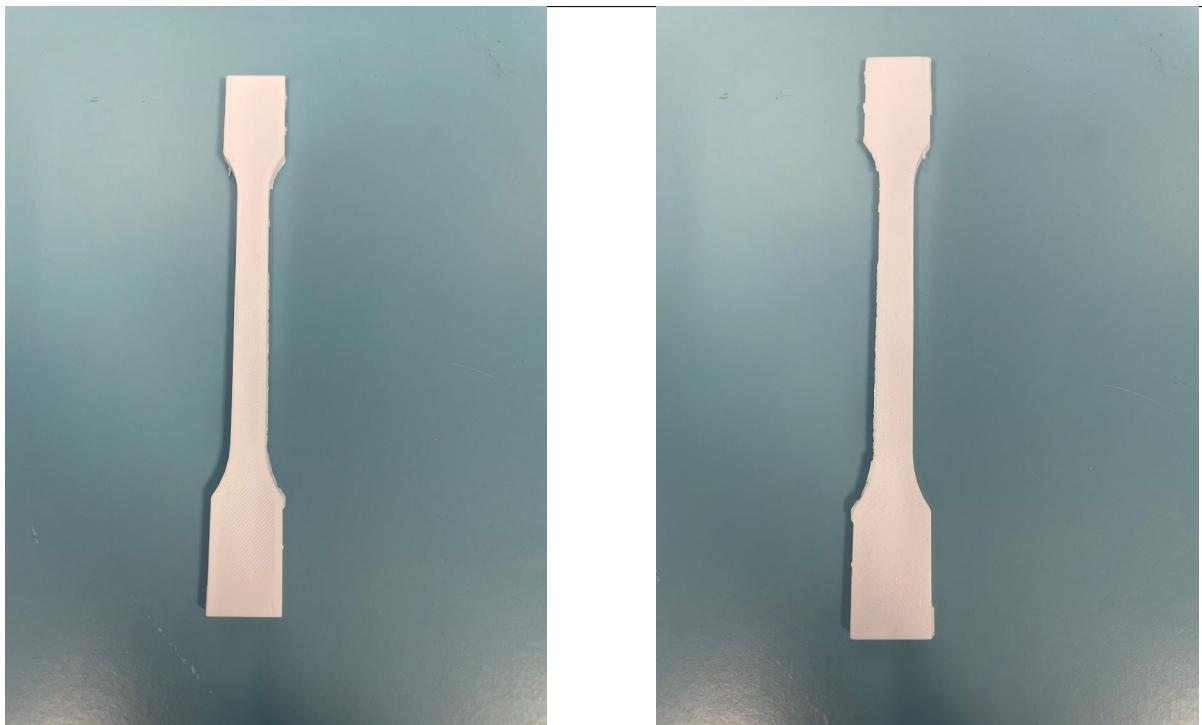


Figure 517: Before tensile test 2 | STB

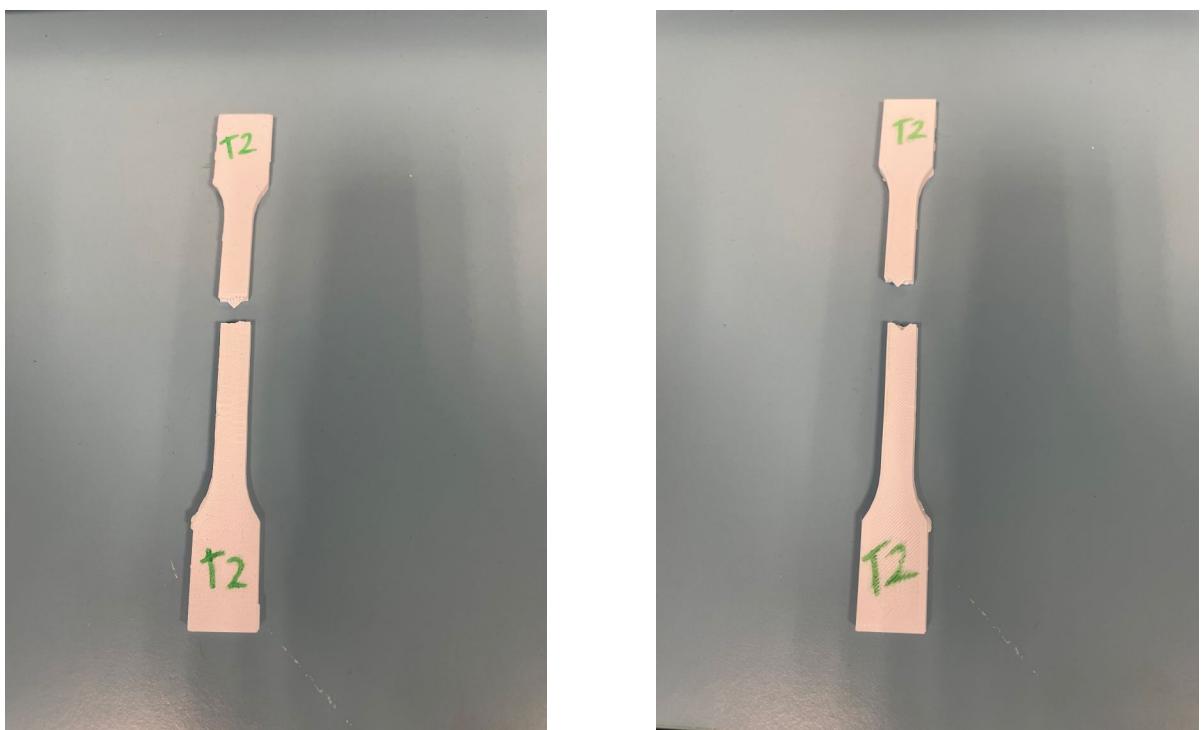


Figure 518: After tensile test 2 | STB

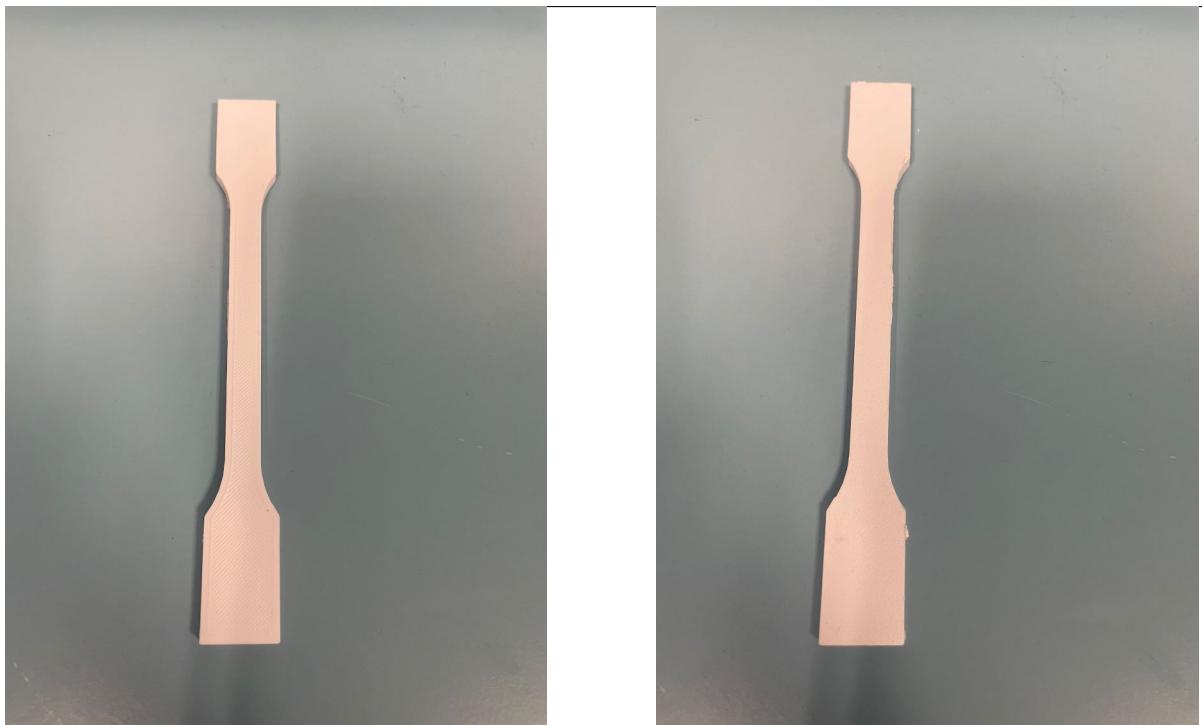


Figure 519: Before tensile test 3 | STB

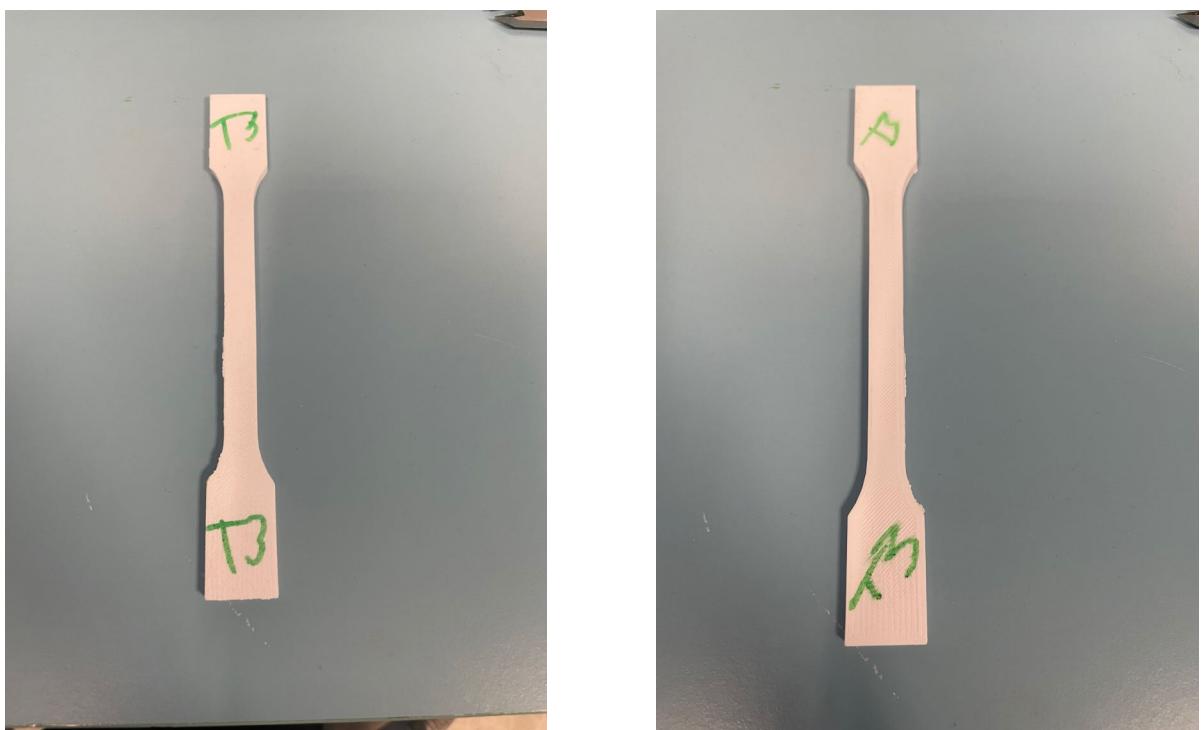


Figure 520: After tensile test 3 | STB

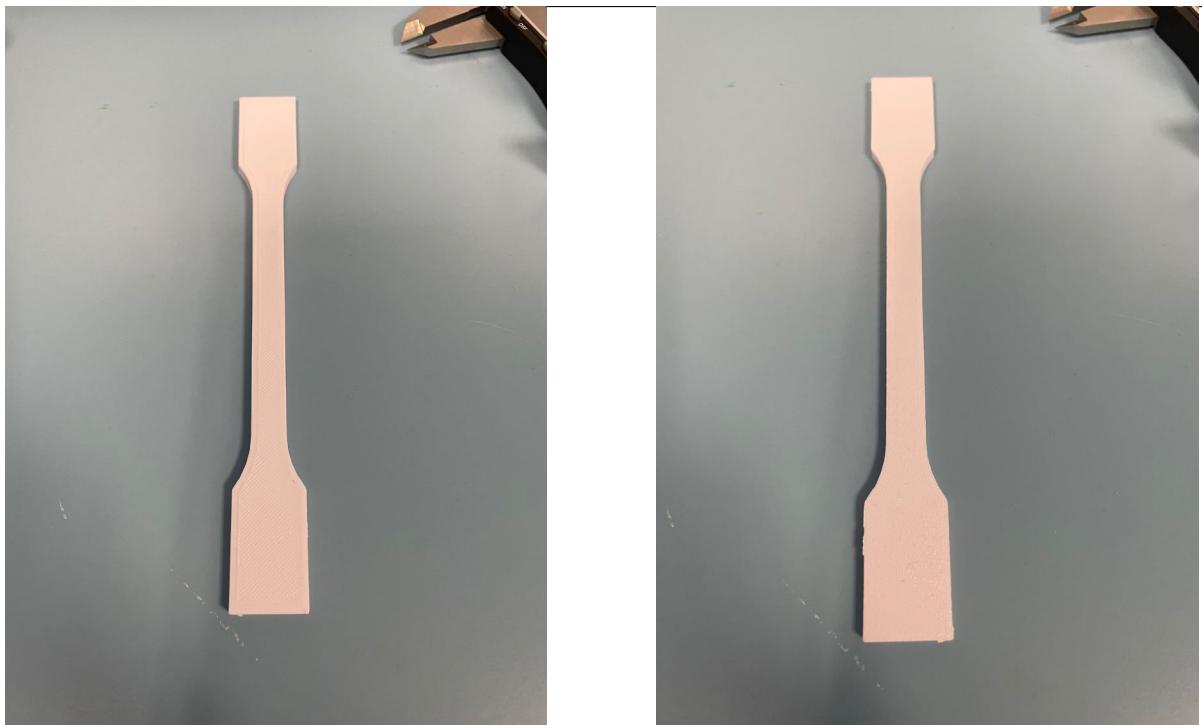


Figure 521: Before tensile test 4 | STB

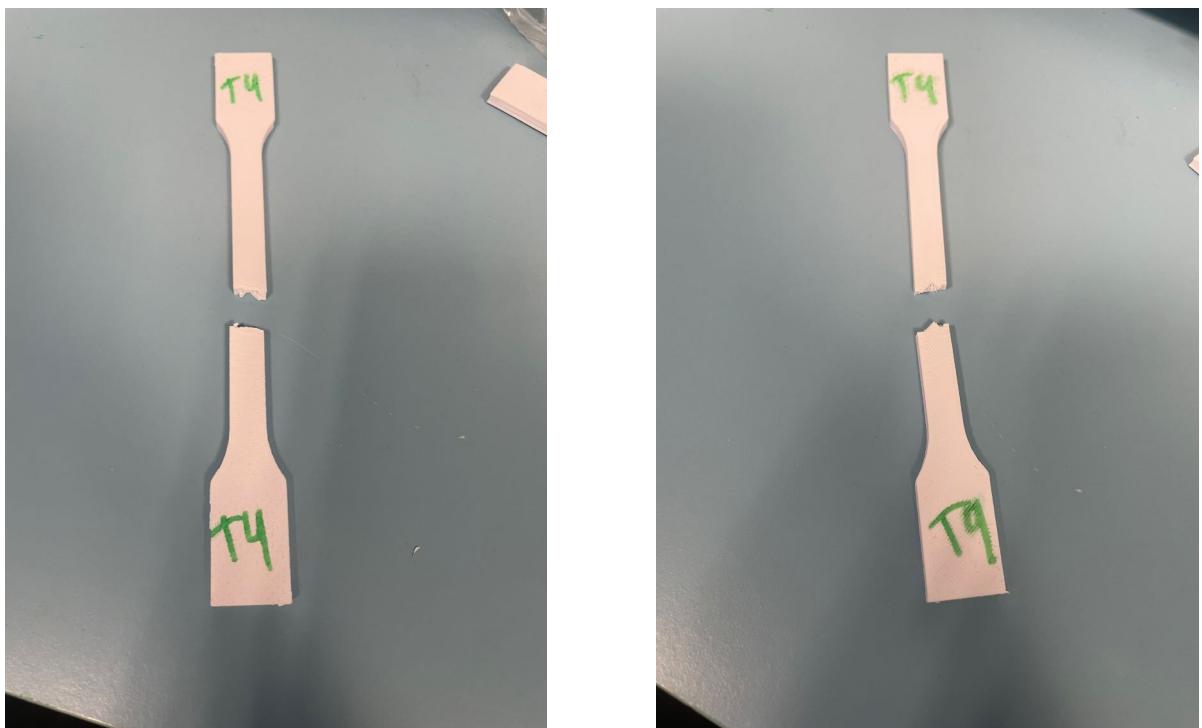


Figure 522: After tensile test 4 | STB

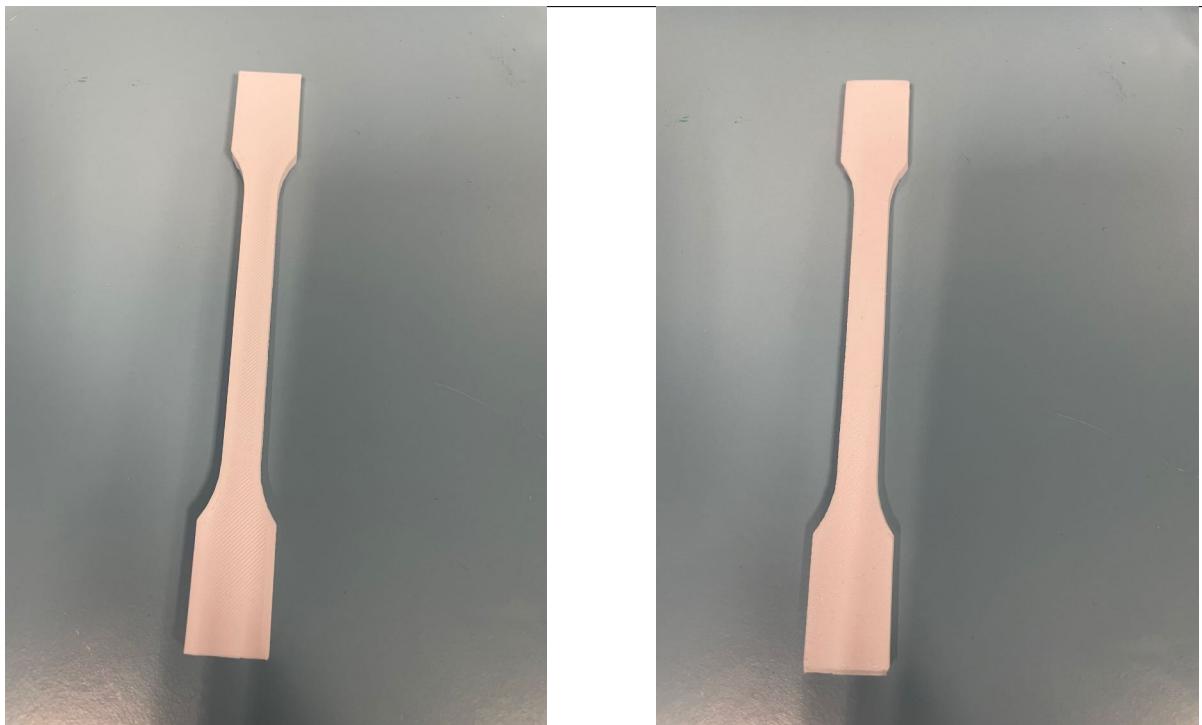


Figure 523: Before tensile test 4 | STB

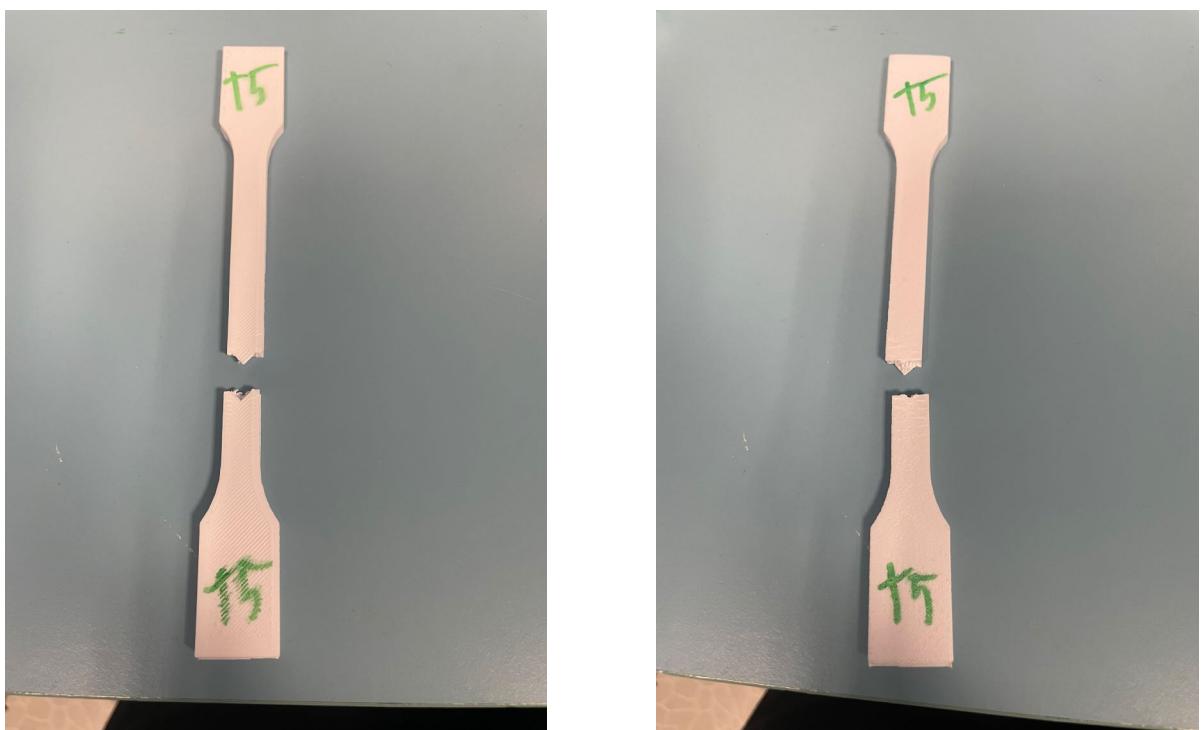


Figure 524: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 525.

	A	B	C	D	E	F	G	H	I	J
1	DATA	Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values	
2		L_0	Chosen start length [mm]	N/A	109,00	109,00	109,00	109,00	N/A	
3		h	Test staff thickness [mm]	N/A	3,86	3,88	3,86	4,02	3,12	
4		b_1	Width at narrow portion [mm]	N/A	9,98	9,90	9,99	9,88	7,95	
5		Value (Settings/test plan)	A	Cross-section [mm ²]	#VALUE!	38,52	38,41	38,56	39,72	N/A
6			L	Gripping distance	N/A	109,00	109,00	109,00	109,00	N/A
7			N/A	Amount of test	5	5	5	5	N/A	
8			v	Test Speed [mm/min]	60	60	60	60	N/A	
9		Proportional limit	σ_y	Nominal Stress [MPa]	N/A	9,133	6,200	10,903	7,501	6,747
10			ε_nom	Nominal strain [%]	N/A	N/A	1,94	N/A	N/A	0,388
11			σ_true	True stress [MPa]	N/A	9,133	320,265	10,902	7,501	69,560
12			ε_true	True strain [%]	N/A	N/A	6,32	N/A	N/A	1,264
13			E_t	Tensile modulus [MPa]	N/A	N/A	1,92	N/A	N/A	0,38
14		UTS	S_ut	Ultimate tensile strength [MPa]	N/A	35,157	9,414	34,919	33,952	22,688
15			ε_norm	Nominal strain [%]	N/A	5,04	3,49	3,62	4,53	3,34
16			σ_true	True stress [MPa]	N/A	36,930	9,743	36,181	35,491	23,669
17			ε_true	True strain [%]	N/A	4,92	3,43	3,55	4,44	3,27
18		Fracture	σ_b	Stress at break [MPa]	N/A	31,748	N/A	32,682	30,078	18,902
19			ε_b	Strain (engineering) at break [%]	N/A	5,96	N/A	4,17	5,50	3,13
20			σ_true	True stress [MPa]	N/A	33,641	N/A	34,045	31,732	19,884
21			ε_true	True strain [%]	N/A	5,79	N/A	4,08	5,35	3,04

Figure 525: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

Test specimen 1 was rejected but we didn't replace it because of time. Physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen during the other tensile tests.

Date of measurement

17. April 2024

**P.22 Material: PLA x Infill pattern: Gyroid
Infill density: 75%
Preparation temp.: -18 to -20 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* $-18^{\circ}C$ to $-20^{\circ}C$. The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between $20^{\circ}C$ and $24^{\circ}C$.

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test



Figure 526: Before tensile test 1 | STB

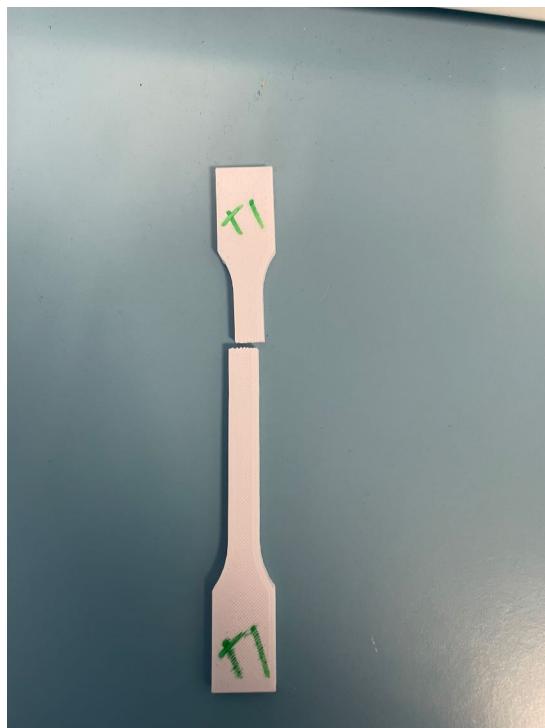


Figure 527: After tensile test 1 | STB

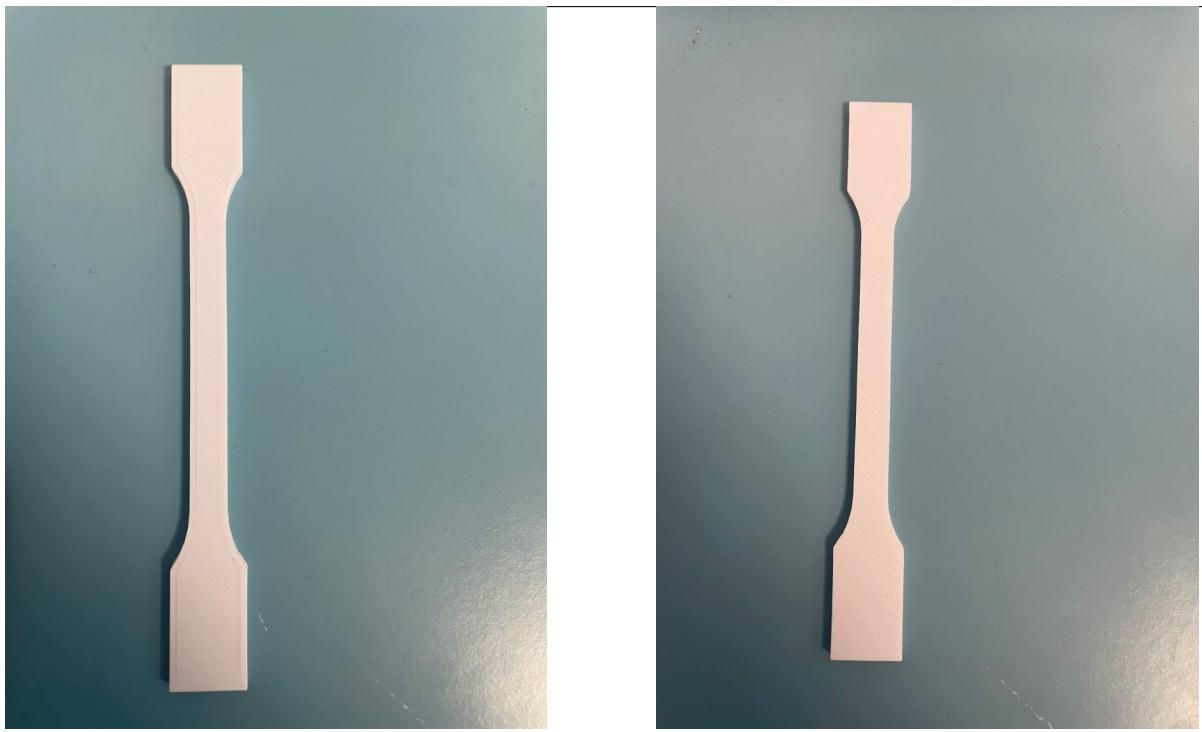


Figure 528: Before tensile test 2 | STB

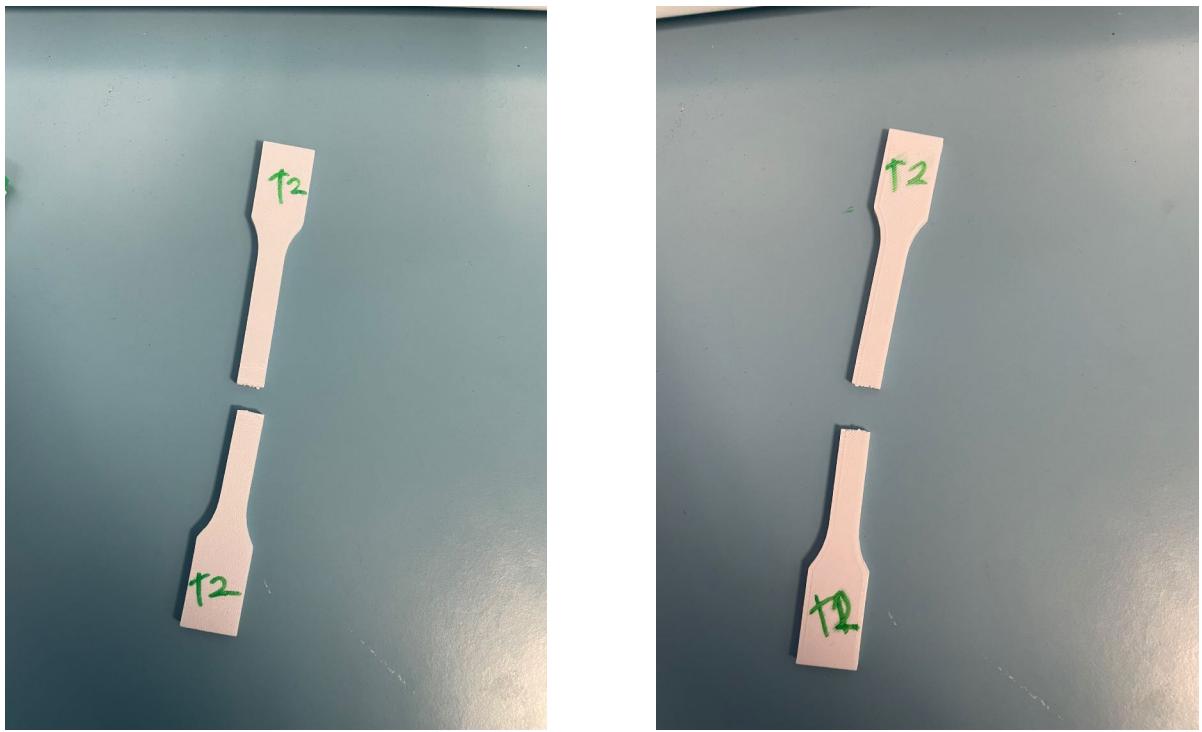


Figure 529: After tensile test 2 | STB

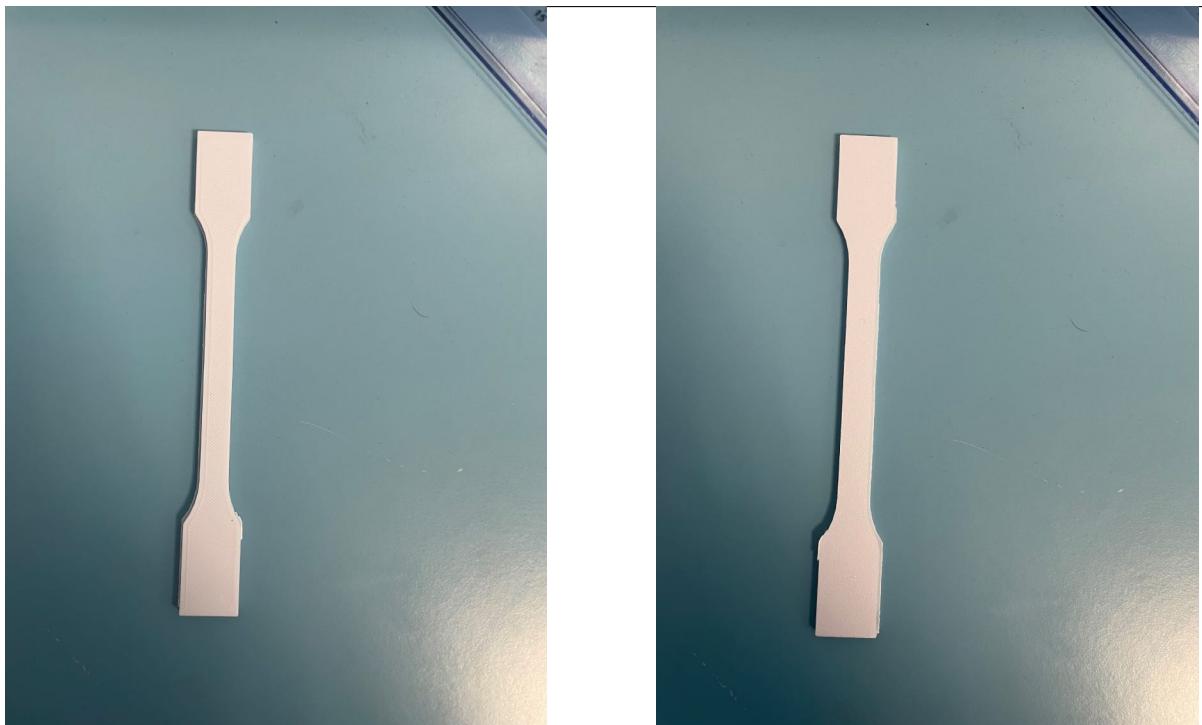


Figure 530: Before tensile test 3 | STB

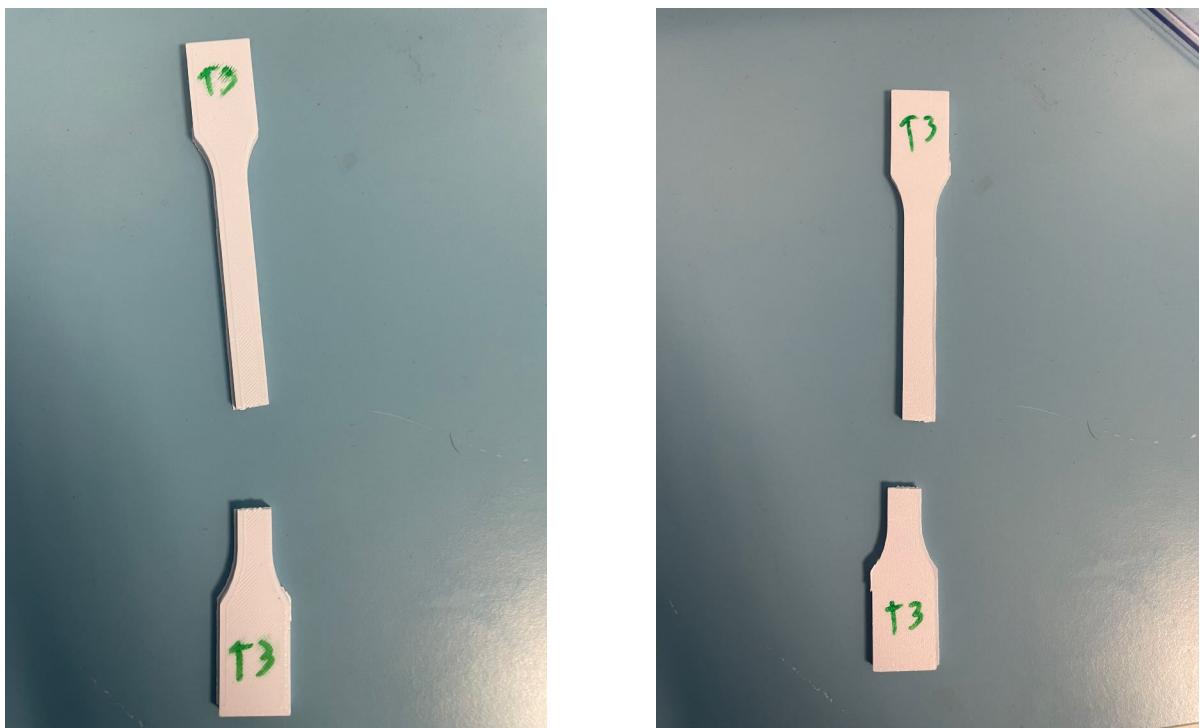


Figure 531: After tensile test 3 | STB

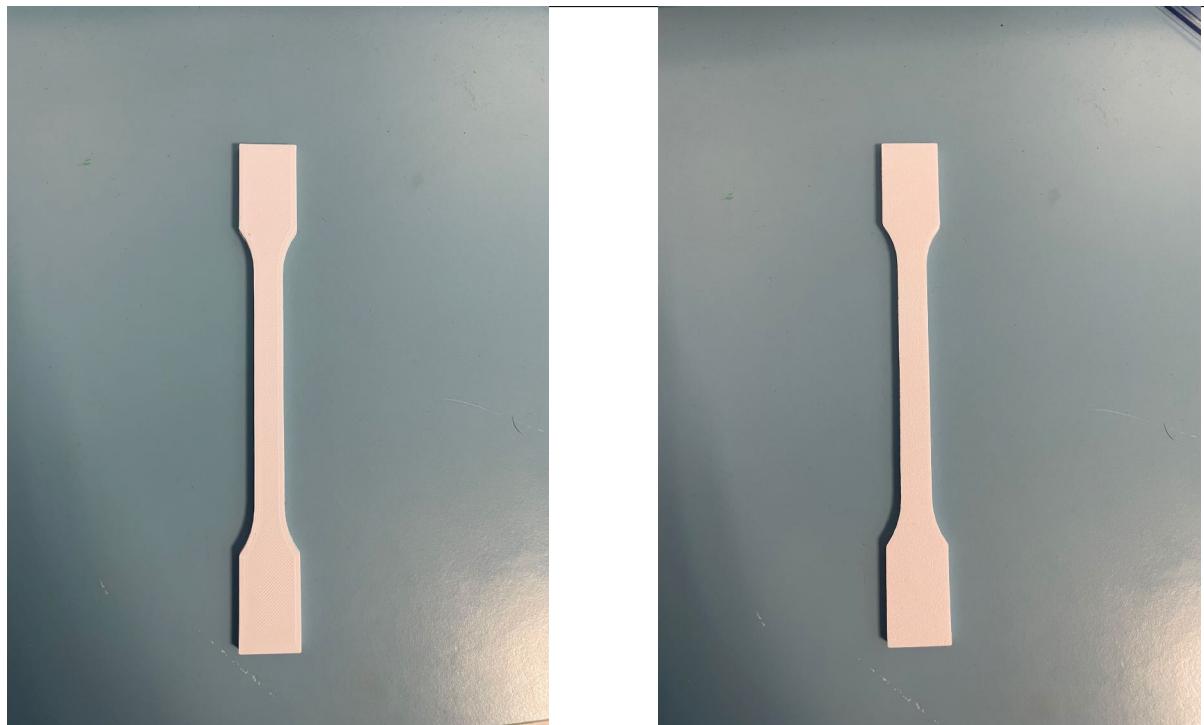


Figure 532: Before tensile test 4 | STB

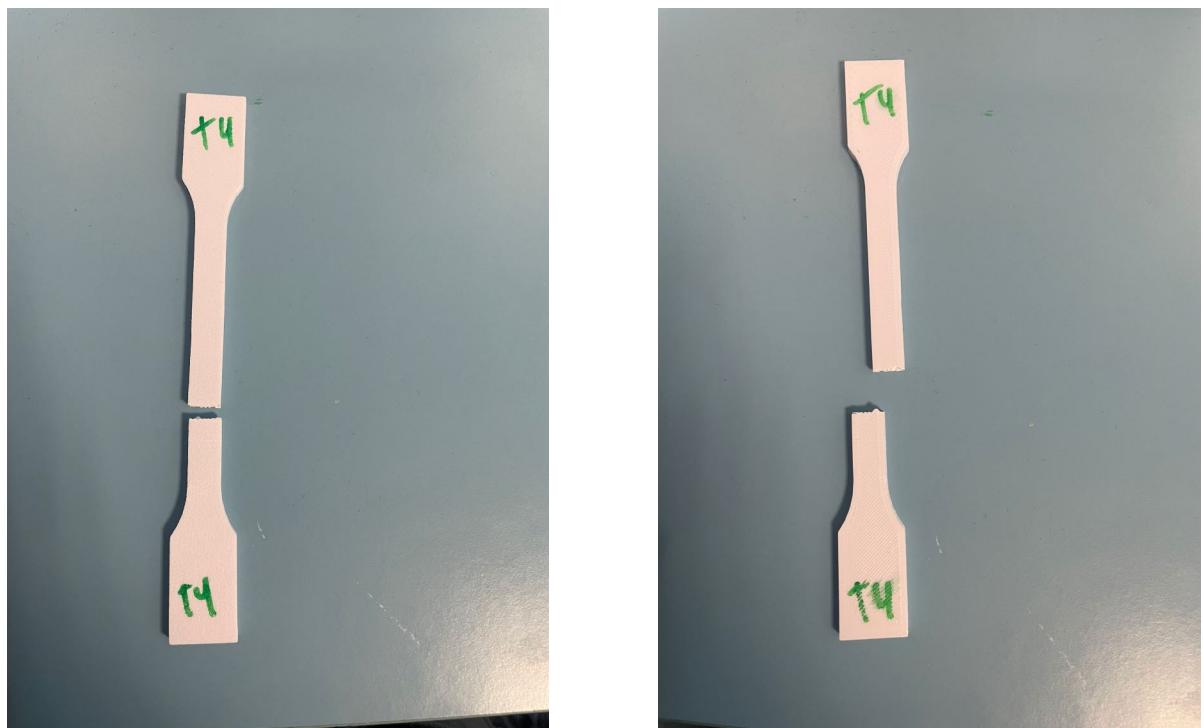


Figure 533: After tensile test 4 | STB

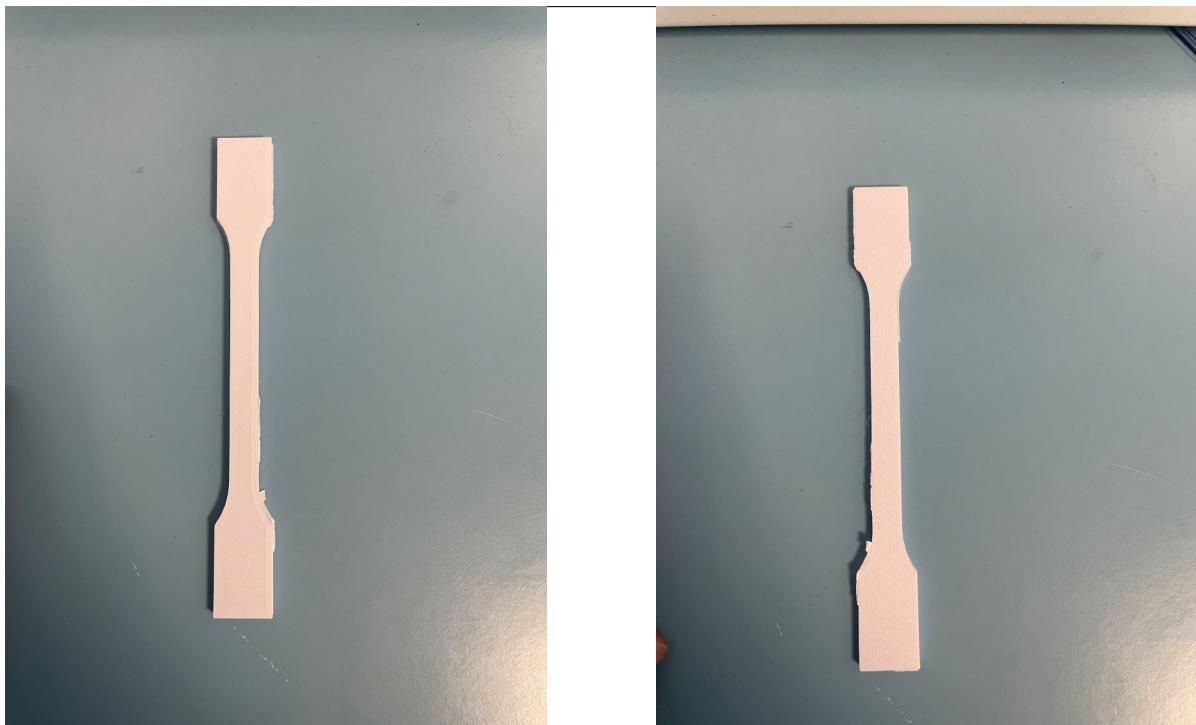


Figure 534: Before tensile test 5 | STB

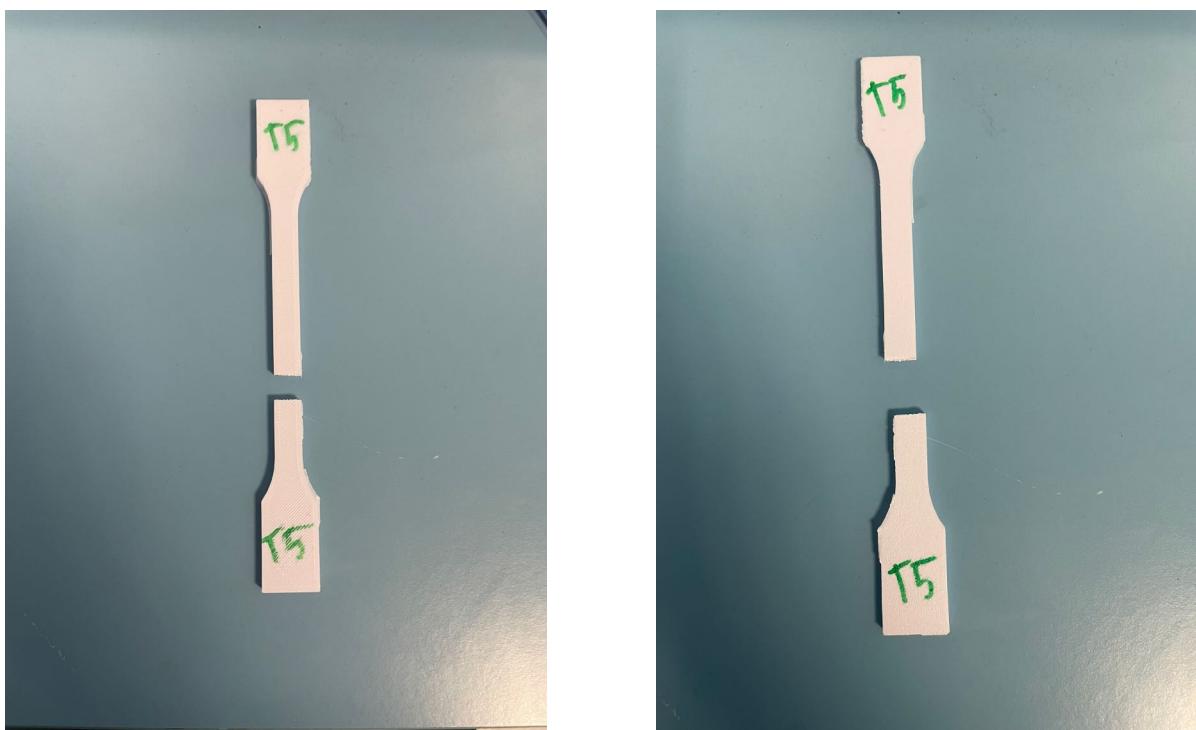


Figure 535: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 536.

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
2		Initial values	L_0 Chosen start length [mm]	108,00	108,00	108,00	108,00	108,00	N/A
3			h Test staff thickness [mm]	3,90	3,85	3,87	3,87	3,88	3,87
4			b_1 Width at narrow portion [mm]	10,06	10,03	9,97	10,00	10,00	10,01
5		Value (Settings/test plan)	A Cross-section [mm ²]	39,23	38,62	38,58	38,70	38,80	N/A
6			L Gripping distance	108,00	108,00	108,00	108,00	108,00	N/A
7			N/A Amount of test	5	5	5	5	5	N/A
8			v Test Speed [mm/min]	60	60	60	60	60	N/A
9		Proportional limit	σ_y Nominal Stress [MPa]	8,493	8,553	9,702	9,800	8,613	9,032
10			ε_nom Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0
11			σ_true True stress [MPa]	8,493	8,553	9,702	9,800	8,613	9,032
12			ε_true True strain [%]	N/A	N/A	N/A	N/A	N/A	0
13			E_t Tensile modulus [MPa]	183445,44	184739,13	N/A	N/A	186042,95	110845,50
14		UTS	S_ut Ultimate tensile strength [MPa]	38,766	38,448	38,099	37,807	38,518	38,328
15			ε_nom Nominal strain [%]	4,44	4,31	3,79	3,88	4,35	4,15
16			σ_true True stress [MPa]	40,489	40,104	39,543	39,276	40,194	39,921
17			ε_true True strain [%]	4,35	4,22	3,72	3,81	4,26	4,07
18		Fracture	σ_b Stress at break [MPa]	35,319	33,398	34,137	34,92	34,906	34,536
19			ε_b Strain (engineering) at break [%]	5,09	5,70	4,62	4,63	5,19	5,05
20			σ_true True stress [MPa]	37,118	35,301	35,715	36,536	36,716	36,277
21			ε_true True strain [%]	4,87	5,54	4,52	4,52	5,06	4,90

Figure 536: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the tensile test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

16. April 2024

**P.23 Material: PLA x Infill pattern: Gyroid
Infill density: 75%
Preparation temp.: -38 to -40 Celsius**

The documentation and testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed cooling-room for 19 hours with the *preparation temperature* $-38^{\circ}C$ to $-40^{\circ}C$. The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between $20^{\circ}C$ and $24^{\circ}C$.

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test

Figure 537: Before tensile test 1 | STB

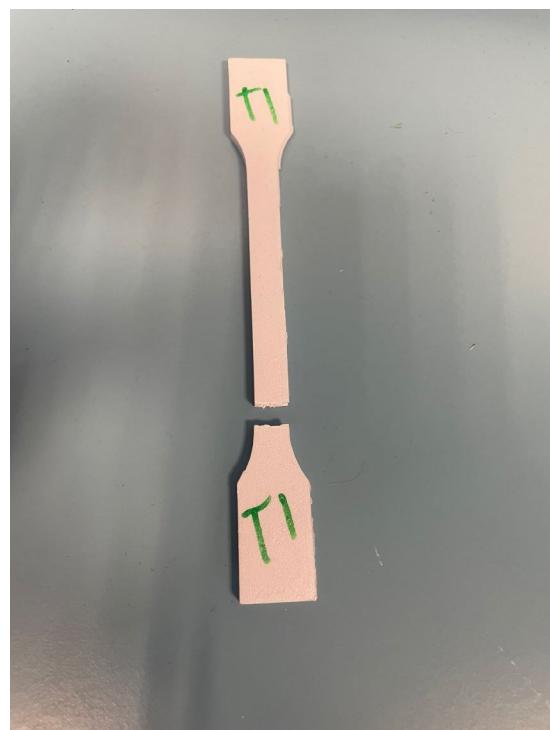
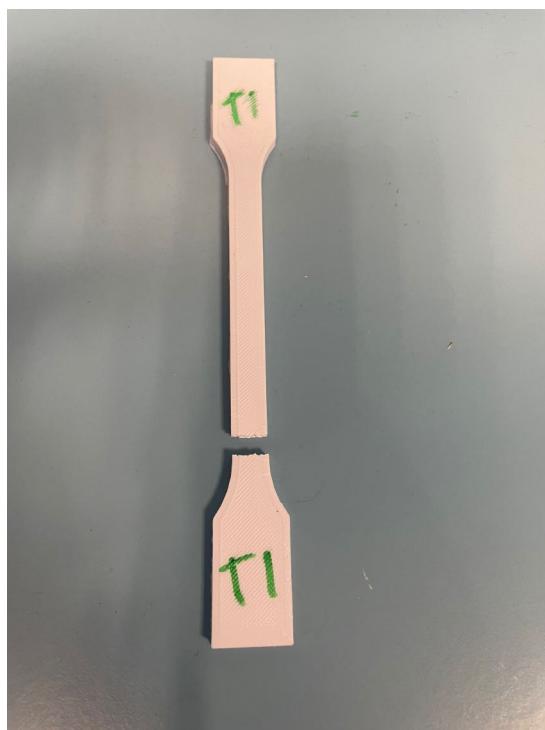


Figure 538: After tensile test 1 | STB

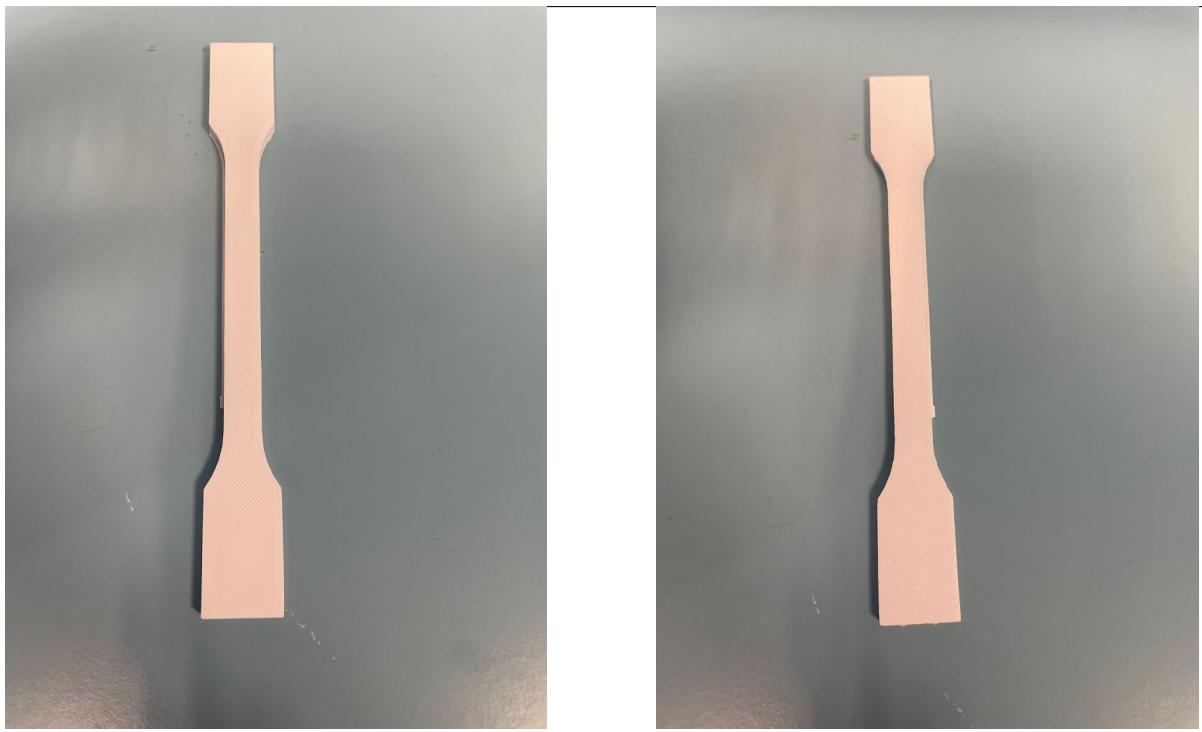


Figure 539: Before tensile test 2 | STB

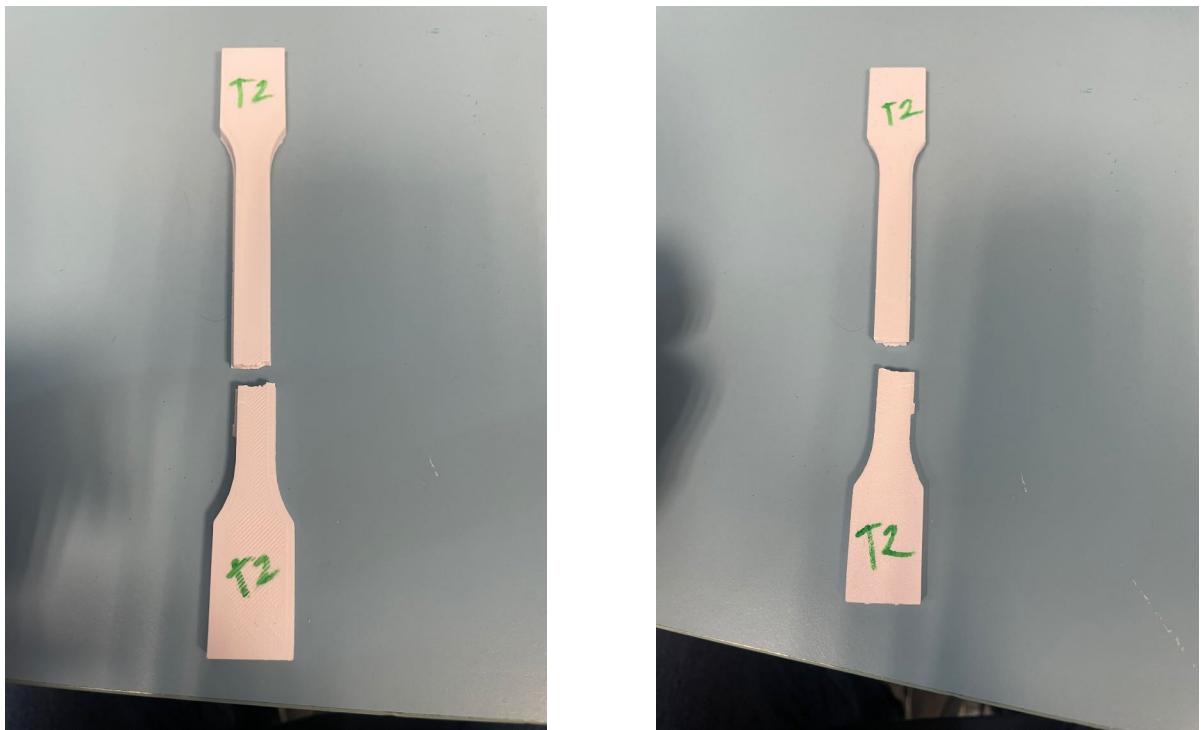


Figure 540: After tensile test 2 | STB

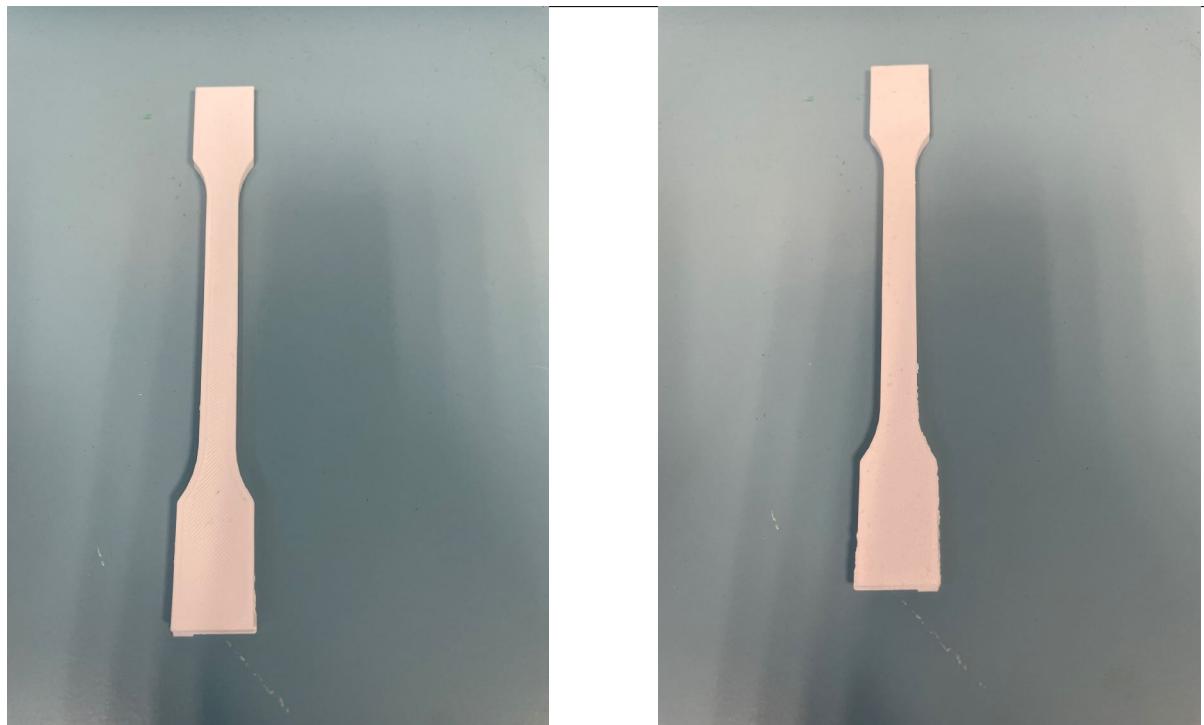


Figure 541: Before tensile test 3 | STB

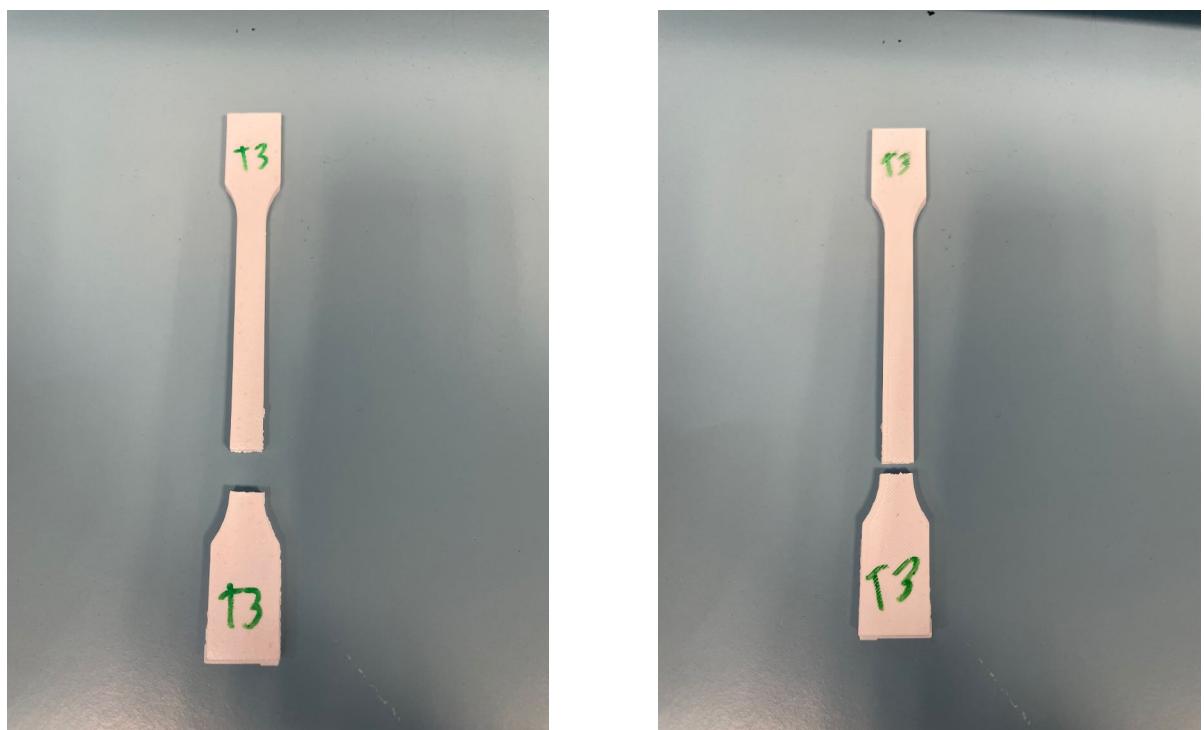


Figure 542: After tensile test 3 | STB

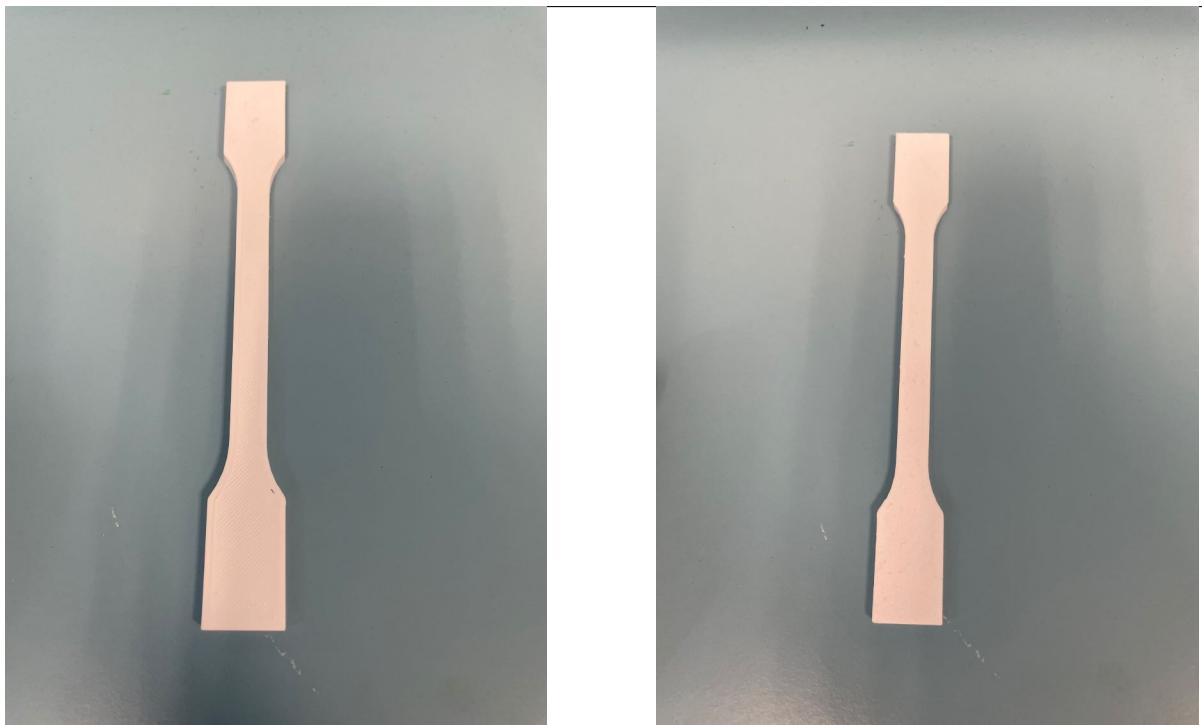


Figure 543: Before tensile test 4 | STB

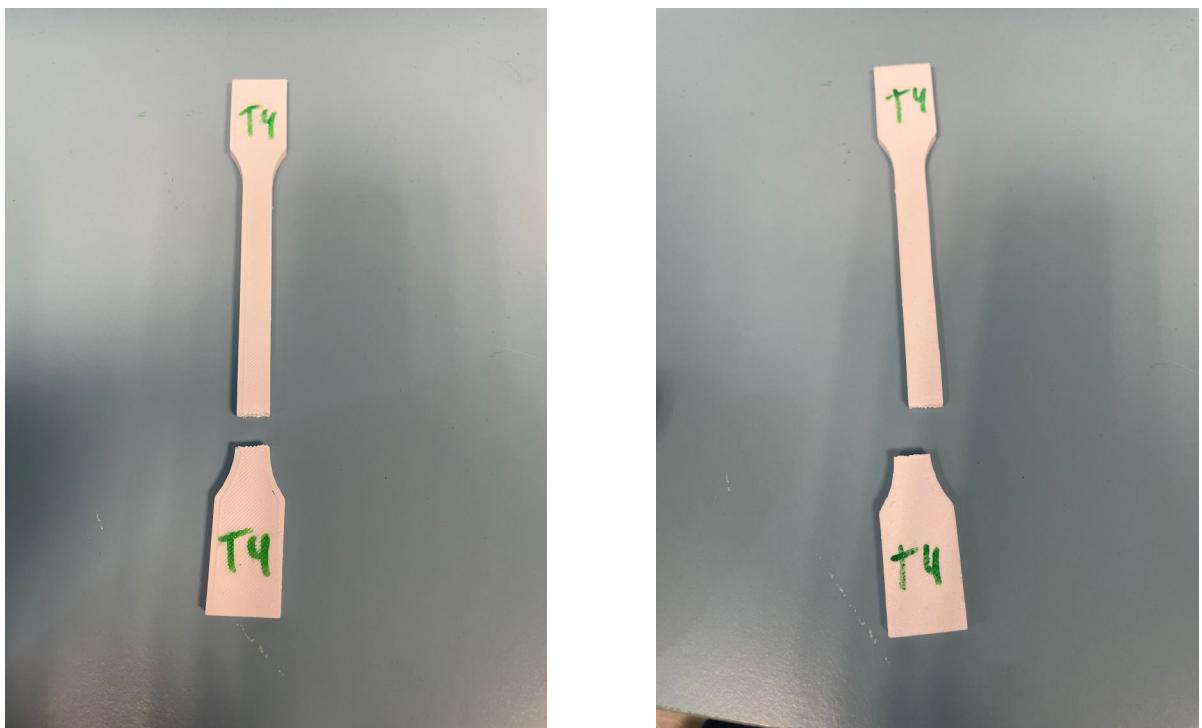


Figure 544: After tensile test 4 | STB

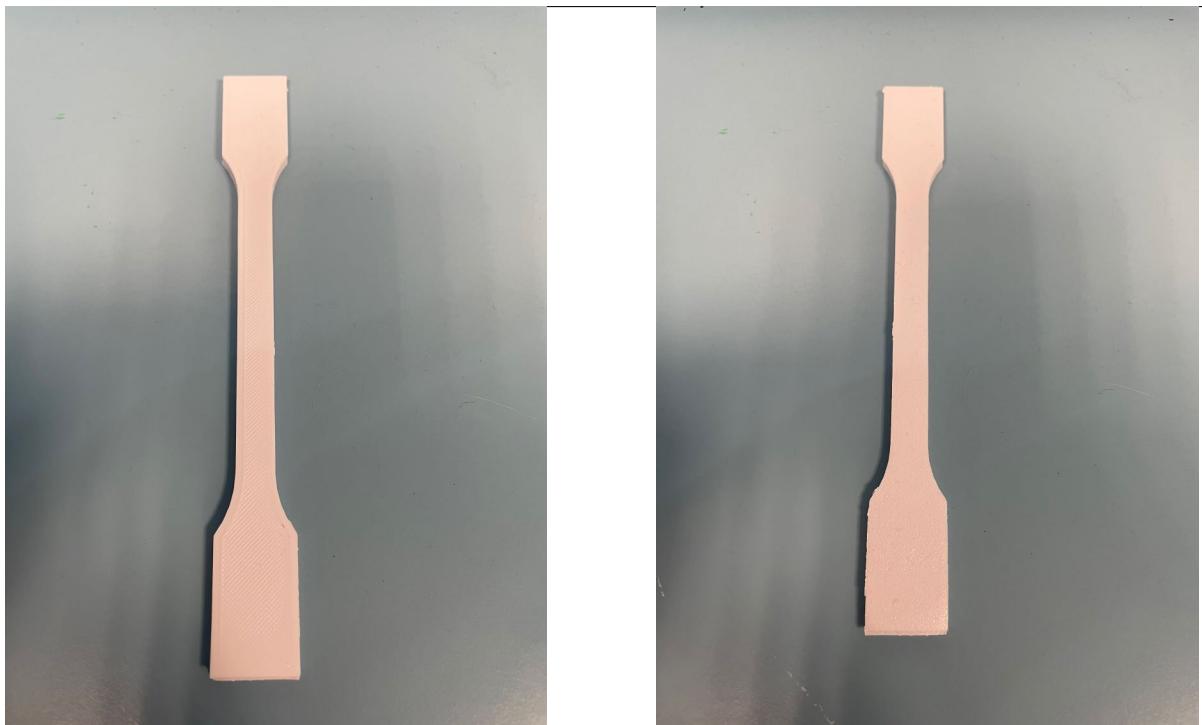


Figure 545: Before tensile test 5 | STB

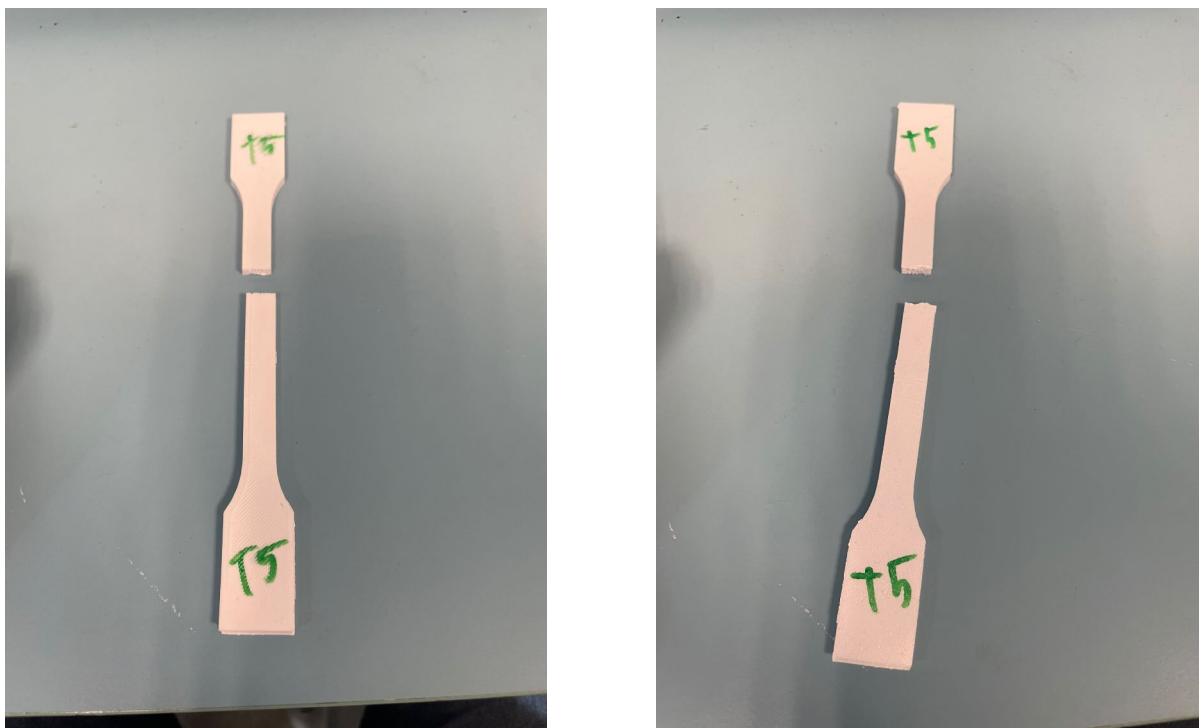


Figure 546: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 547.

	A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values	
2		Initial values	L_0 Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A	
3			h Test staff thickness [mm]	3,83	3,84	3,87	3,87	3,89	3,86	
4			b_1 Width at narrow portion [mm]	9,98	10,05	9,96	9,96	9,90	9,97	
5		Value (Settings/test plan)	A Cross-section [mm ²]	38,22	38,59	38,55	38,55	38,51	N/A	
6			L Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A	
7			N/A Amount of test	5	5	5	5	5	N/A	
8			v Test Speed [mm/min]	60	60	60	60	60	N/A	
9		Proportional limit	σ_y Nominal Stress [MPa]	8,820	9,573	9,483	8,695	8,907	9,096	
10			ε_nom Nominal strain [%]	0,68	0,53	0,36	0,79	0,59	0,59	
11			σ_true True stress [MPa]	8,875	9,624	9,517	8,764	8,959	9,148	
12			ε_true True strain [%]	0,62	0,53	0,36	0,78	0,58	0,574	
13			E_t Tensile modulus [MPa]	1420,02	1808,510	2657,40	1107,22	1519,27	1702,48	
14		UTS	S_ut Ultimate tensile strenght [MPa]	40,766	39,132	38,951	38,696	40,512	39,611	
15			ε_nom Nominal strain [%]	4,88	4,65	4,30	5,00	4,62	4,69	
16			σ_true True stress [MPa]	42,756	40,953	40,624	40,632	42,234	41,440	
17			ε_true True strain [%]	4,77	4,55	4,21	4,88	4,52	4,59	
18		Fracture	σ_b Stress at break [MPa]	38,509	36,389	36,535	37,323	37,077	37,167	
19			ε_b Strain (engineering) at break [%]	5,43	5,16	4,89	5,28	5,40	5,23	
20			σ_true True stress [MPa]	40,602	38,266	38,323	39,293	39,079	39,113	
21			ε_true True strain [%]	5,29	5,03	4,78	5,14	5,26	5,10	

Figure 547: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the tensile test, a physical clamp-force from the test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

18. April 2024

**P.24 Material: PLA x Infill pattern: Gyroid
Infill density: 75%
Preparation temp.: 20 to 24 Celsius**

The documentation and tensile testing is based on sections/requirements from standards:

- NS-EN ISO 527-1:2019
- NS-EN ISO 291:2008
- NS-EN ISO 527-2:2012

The report setup is based NS-EN ISO 527-1:2019, but won't contain every point required from the standard due to time, less informative information from testing, and it's not required for further developing of the C.R.A.F.T-Sat.

STB|

Material data

- Material type: PLA - read section 5.2.1 for information about the polymer.
- Source: Clas Ohlson

STB|

Test specimen type & testing speed

The tensile test specimen is designed and based on: ISO 527-2/1A/60. The 1A design is shown in fig 559.

STB|

Method of preparing the tensile test specimens and details of the manufacturing method

The tensile test specimens was placed in a closed room for 19 hours with the *preparation temperature* 20°C to 24°C . The overall slicer settings chosen to be variable and constants on the Cura program is described in section K.

STB|

The atmosphere for conditioning testing

The ambient temperature of the environment during testing was between 20°C and 24°C .

STB|

Special conditioning treatment

Before testing the tensile test specimen was taken out of the closed-room and placed in the testing room with a minimum 3 hour waiting time with temperature between $20^{\circ}C$ and $24^{\circ}C$ before tensile test.

Pictures - before & after tensile test



Figure 548: Before tensile test 1 | STB

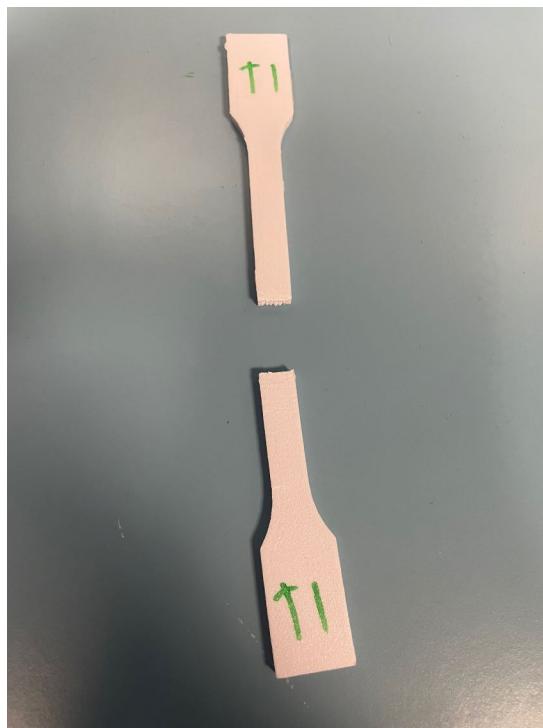


Figure 549: After tensile test 1 | STB

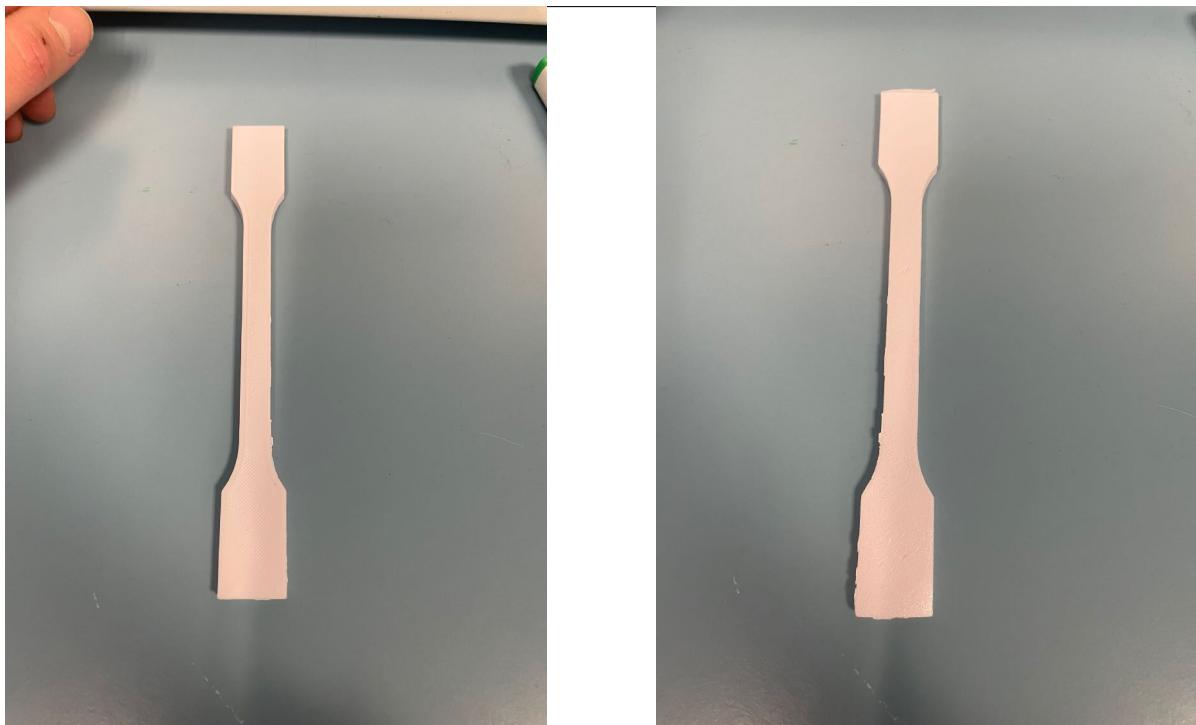


Figure 550: Before tensile test 2 | STB

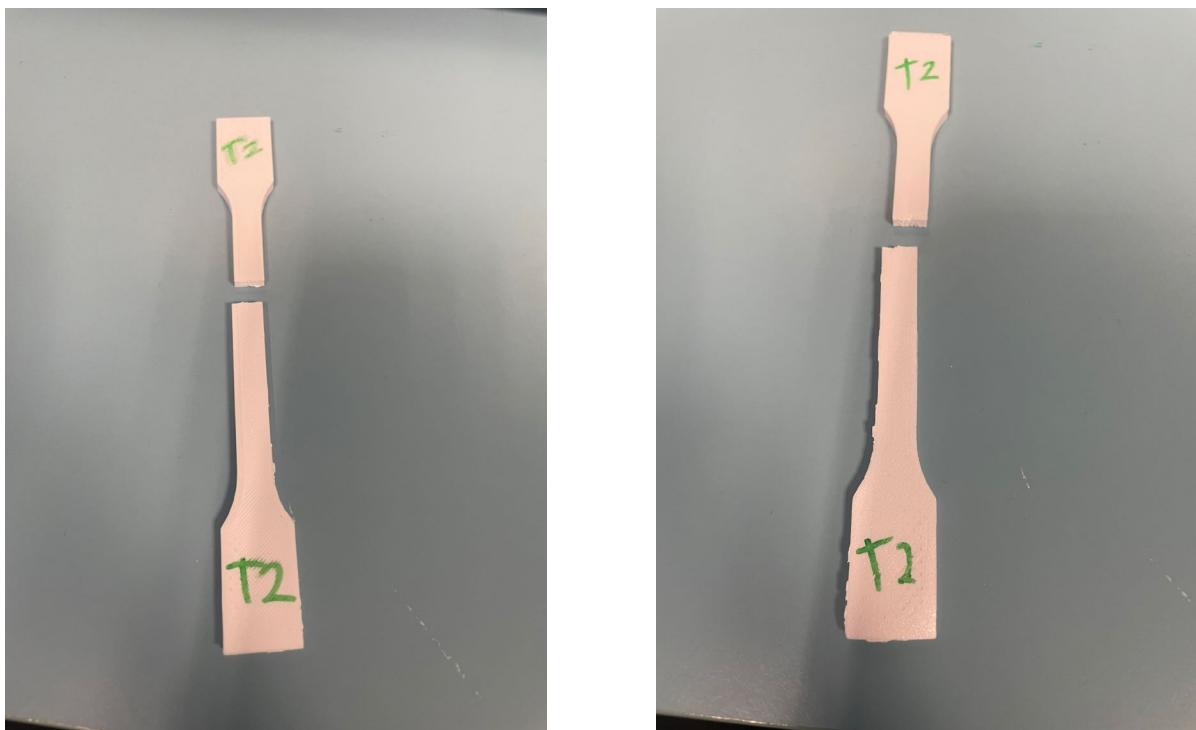


Figure 551: After tensile test 2 | STB

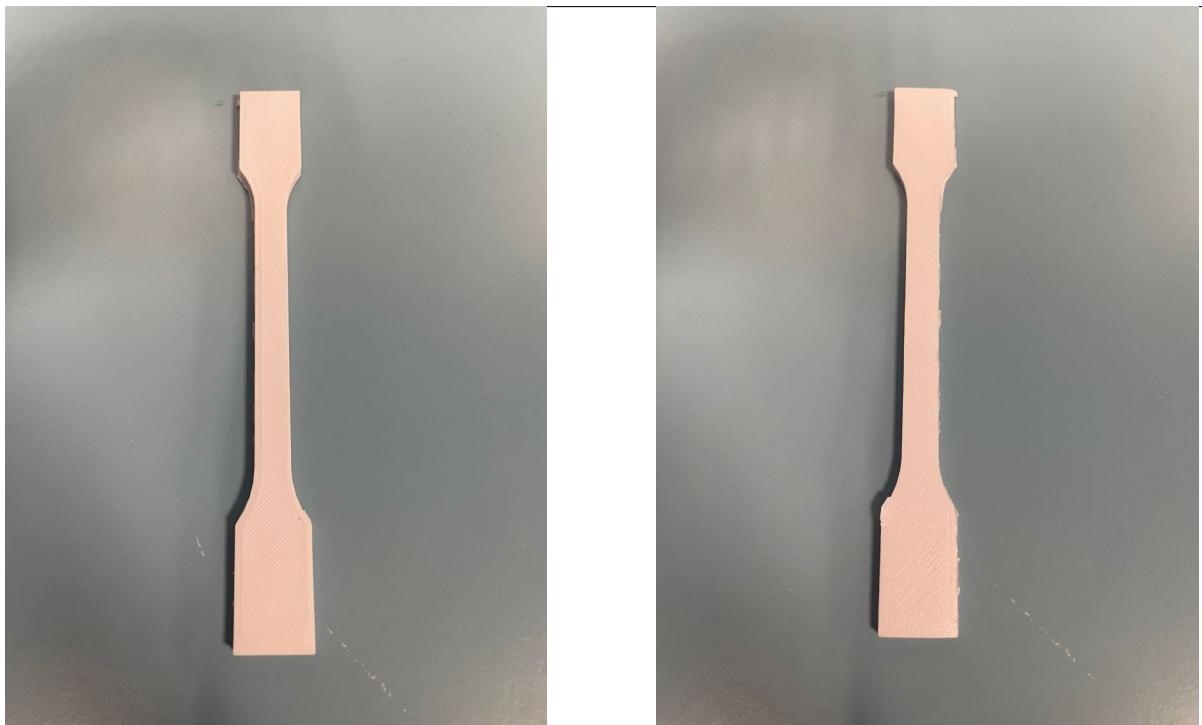


Figure 552: Before tensile test 3 | STB

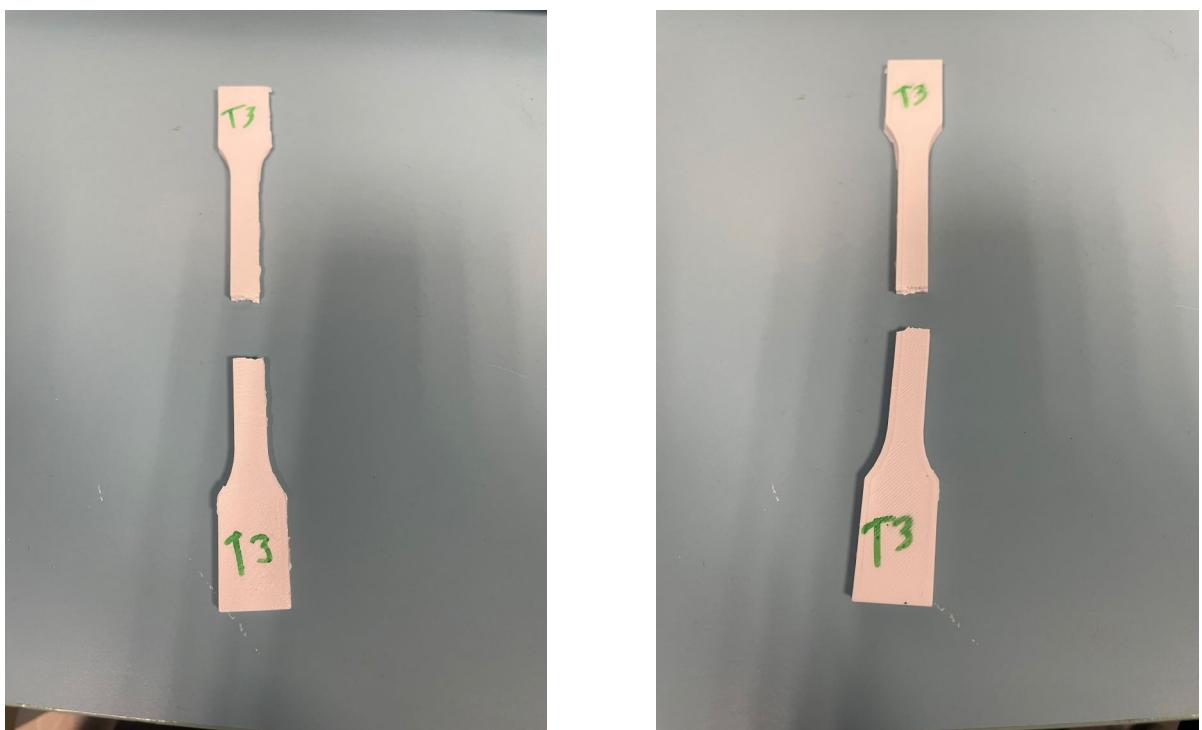


Figure 553: After tensile test 3 | STB

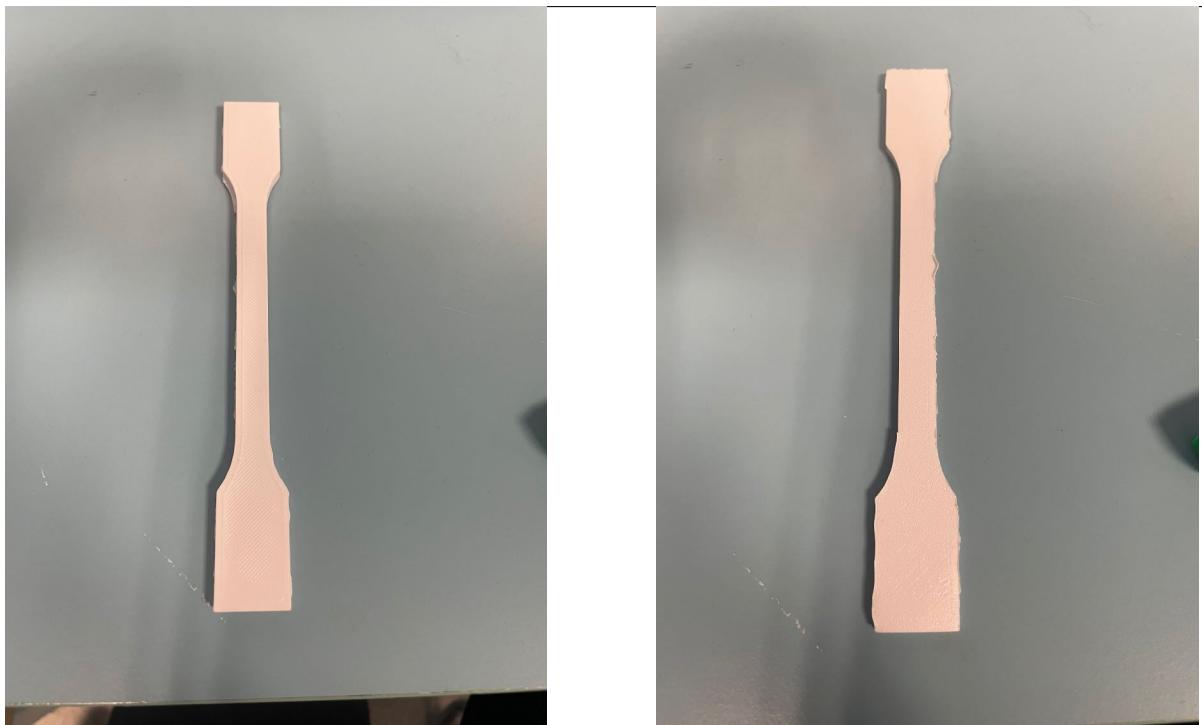


Figure 554: Before tensile test 4 | STB

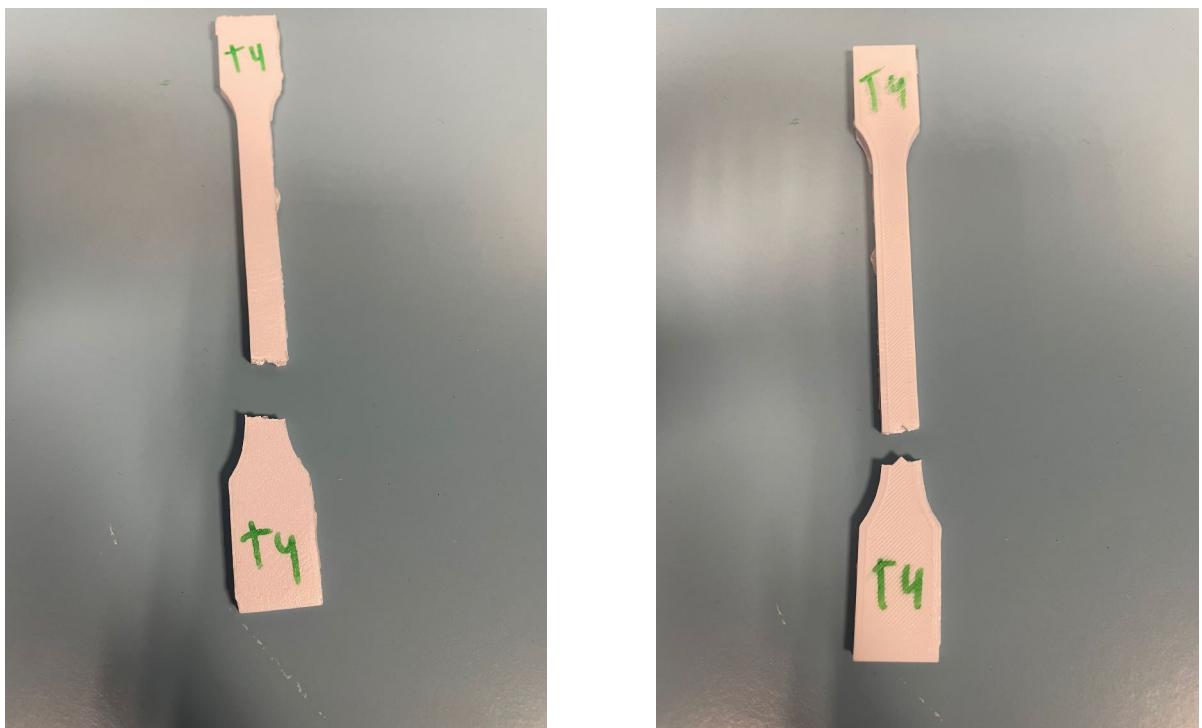


Figure 555: After tensile test 4 | STB

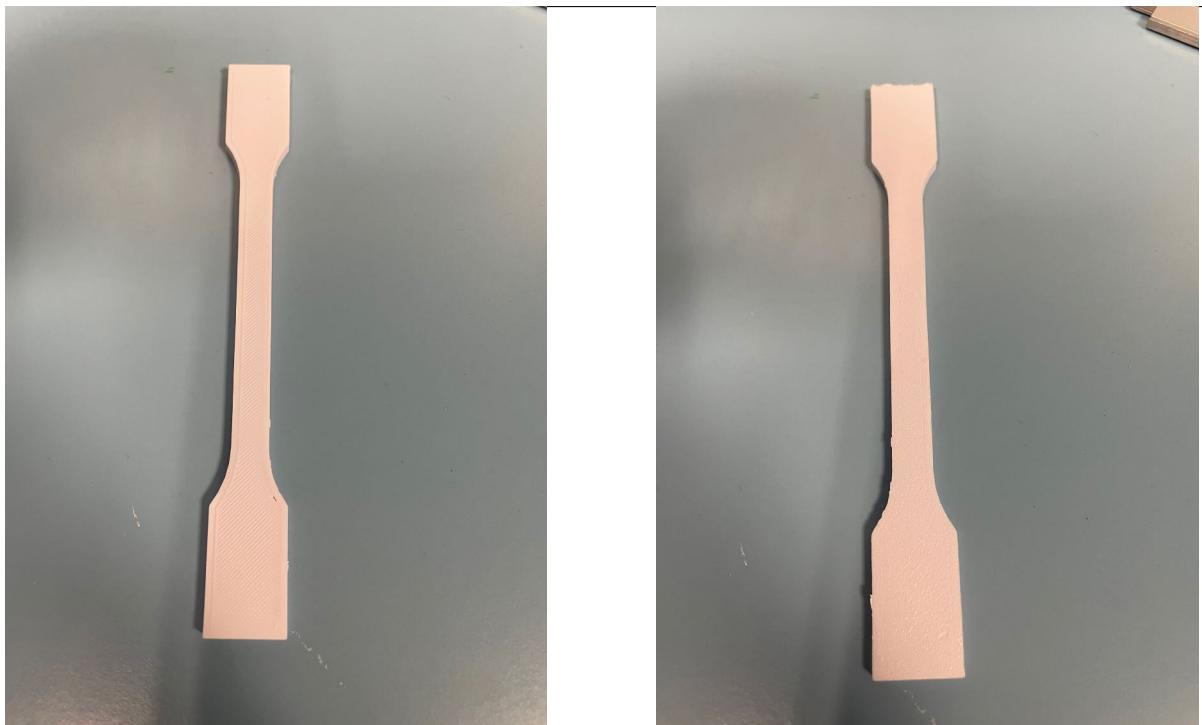


Figure 556: Before tensile test 5 | STB

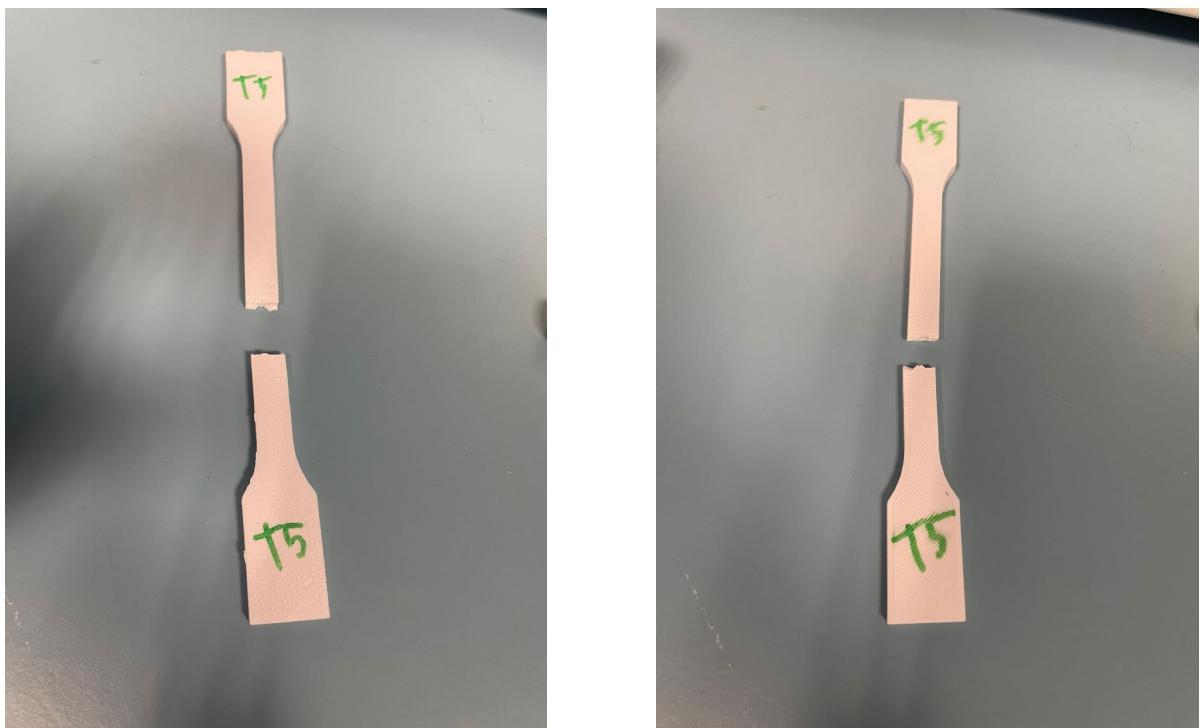


Figure 557: After tensile test 5 | STB

Properties from each tensile test specimen & mean values

The result from the the tensile test is shown in fig 558.

A	B	C	D	E	F	G	H	I	J
1		Units	Description	Specimen 1	Specimen 2	Specimen 3	Specimen 4	Specimen 5	Mean values
2		Initial values	L_0 Chosen start length [mm]	109,00	109,00	109,00	109,00	109,00	N/A
3			h Test staff thickness [mm]	3,82	3,89	3,83	3,86	3,85	3,85
4			b_1 Width at narrow portion [mm]	10,10	9,96	10,04	10,00	10,00	10,02
5		Value (Settings/test plan)	A Cross-section [mm ²]	38,58	38,74	38,45	38,60	38,50	N/A
6			L Gripping distance	109,00	109,00	109,00	109,00	109,00	N/A
7			N/A Amount of test	5	5	5	5	5	N/A
8			v Test Speed [mm/min]	60	60	60	60	60	N/A
9		Proportional limit	σ_y Nominal Stress [MPa]	7,925	6,753	9,761	8,988	12,778	9,241
10			ε_nom Nominal strain [%]	N/A	N/A	N/A	N/A	N/A	0
11			σ_true True stress [MPa]	7,925	6,753	9,761	8,988	12,778	9,241
12			ε_true True strain [%]	N/A	N/A	N/A	N/A	N/A	0
13			E_t Tensile modulus [MPa]	215957,38	N/A	354645,59	326545,94	N/A	179429,78
14		UTS	S_ut Ultimate tensile strength [MPa]	38,329	39,712	39,528	39,479	38,665	39,143
15			ε_nom Nominal strain [%]	4,72	4,44	4,04	4,13	3,35	4,14
16			σ_true True stress [MPa]	40,140	41,475	41,123	41,108	39,959	40,761
17			ε_true True strain [%]	4,62	4,34	3,96	4,04	3,29	4,05
18		Fracture	σ_b Stress at break [MPa]	35,205	39,231	36,215	35,29	34,465	36,081
19			ε_b Strain (engineering) at break [%]	5,41	4,62	4,68	5,23	4,31	4,85
20			σ_true True stress [MPa]	37,11	41,044	37,909	37,135	35,951	37,830
21			ε_true True strain [%]	5,27	4,52	4,57	5,10	4,22	4,74

Figure 558: Data from tensile test; standard - NS-EN ISO 527-1:2019 | STB

Statement about the tensile test specimens; rejected and replaced

The tensile test specimens was not rejected and replaced, but during the tensile test, a physical clamp-force from the tensile test team had to be used due to the smooth surface on the tensile test specimen.

Date of measurement

17. April 2024

Q ISO Standards: Documentation, modeling- & test specimen

NS-EN ISO is described as the instruction manual of in our case the design, documentation, and test method for tension data gaining for polymer material. We therefore agreed in using those standards to get a approved and more sufficient method in gaining the data we need. The NS-EN ISO standards we have used is written as the title in the subsection. We haven't used all data from the standards, and have therefore written what we have used.

STB|

Q.1 NS-EN ISO 527-1:2019 Determination of tensile properties

The NS-EN ISO standard are related to determination tensile properties for plastics and plastic composites under defined conditions. The investigation methods used are related to the tensile behaviour of the test specimens and determine the tensile strength, tensile modulus, and other aspects related with the relationship between the tensile strain/stress under defined defined conditions. [185]

According to section 4, principle and methods, subsection 4.1 Principle; "*the test staff specimen will be extended along it's major longitudinal axis at a constant test speed until the specimen fractures or until the stress (load) or the strain (elongation) reaches some predetermined value. During the tensile measurement procedure, the load sustained by the specimen and the elongation are measured*". [185]

According to subsection 4.2 Method, sub-subsection 4.2.1 & 4.2.2; "*the methods are applied using specimens which may be either moulded to the chosen dimensions or machined, cut or punched from finished and semi-finished products, such as mouldings, laminates, films and extruded or cast sheet. The methods specify preferred dimensions for the test specimens. Tests which are carried out on specimens of different dimensions, or on specimens which are prepared under different conditions, may produce results which are not comparable. Other factors, such as the speed of testing and the conditioning of the specimens, can also influence the results. Consequently, when comparative data are required, these factors shall be carefully controlled and recorded*" [185]

According to section 10, Calculation and expression of results, subsection 10.1, Stress; "*It's important in determination of factors to calculate the tensile stress. Every stress related stress related to normal force is calculated using formula:*

$$\sigma = \frac{F}{A} \quad (53)$$

In formula 53, σ is the tensile stress expressed in megapascals (MPa), F is the force

normal to the cross-section area expressed in newton (N) and A is the cross-section area expressed in square millimetres (mm^2)". [185]

According to section 12, Test report; "*The test report shall include the information specified in items (a) to (q)".* We have decided to not use all the items since some items wasn't decided as equal important with the short time in the assignment. The items we decided to use is:

- (a) a reference to the relevant part of NS-EN ISO 527;
- (b) all the data necessary for identification of the material tested, including type, source, manufacturer's code number and history, where these are known;
- (d) type of test specimen;
- (e) method of preparing the test specimens, and any details of the manufacturing method used;
- (g) number of the test specimen tested;
- (h) standard atmosphere for conditioning and testing, plus any special conditioning treatment, if required by the relevant standard for the material or product concerned;
- (i) accuracy grading of the test machine and extensometer (see ISO 7500-1, ISO 9513, and *NS-EN ISO 527-1:2019, p. 8, cp. 5.1.5*);
- (j) The gauge length L_0 ; it's referred as the chosen start length in the data sheets.
- (k) type of gripping device, the gripping distance L ;
- (l) testing speeds;
- (m) for each test specimen width, thickness and test results of the properties defined in [185], p. 2, ch. 3;
- (n) mean value(s) of width, thickness and of the measured property(ies);
- (p) statement as to whether any test specimens have been rejected and replaced, and, if so, the reasons, and reasons for testing non-conforming specimens;
- (q) date of measurement".

Q.2 NS-EN ISO 291:2008 Plastics - Standard atmospheres for conditioning and testing.

The NS-EN ISO standard are related to the conditioning and testing of all plastics and all types of test specimen at a constant atmospheric conditions. We have chosen some sections inside the standard as important. Those section is written in this subsection.

Note: "*Special atmospheres applicable to a particular test or material or simulating a particular climatic environment are not included in this International Standard*".[186]

According to section 3, Subsection 3.6, ambient temperature; "*environmental conditions corresponding to the usual atmospheric conditions in laboratories with uncontrolled temperature and humidity. Ambient temperature is define as temperature in the range from 18°C to 28°C*". [186]

According to section 8, Procedure, subsection 8.2, testing; "*unless otherwise specified, specimens shall be tested in the same atmosphere or at the same temperature in which they have been conditioned. In all cases, the test shall be carried out immediate after the removal of the specimens from the conditioning enclosure*". [186].

Q.3 NS-EN ISO 527-2:2012 Determination of tensile properties - Part 2: Test conditions for moulding and extrusion of plastics.

The NS-EN ISO standard describes what design we should use for the tensile testing staff. According to NS-EN ISO 527-2:2012 6.1 Shape and dimensions; the test staff we are using is a type 1A. A sheet with the dimensions was redrawn in the group with equal properties as the drawing from the standard - document. The dimensions of the test staff is in fig 559. The letters including values are described in table 43. [184]

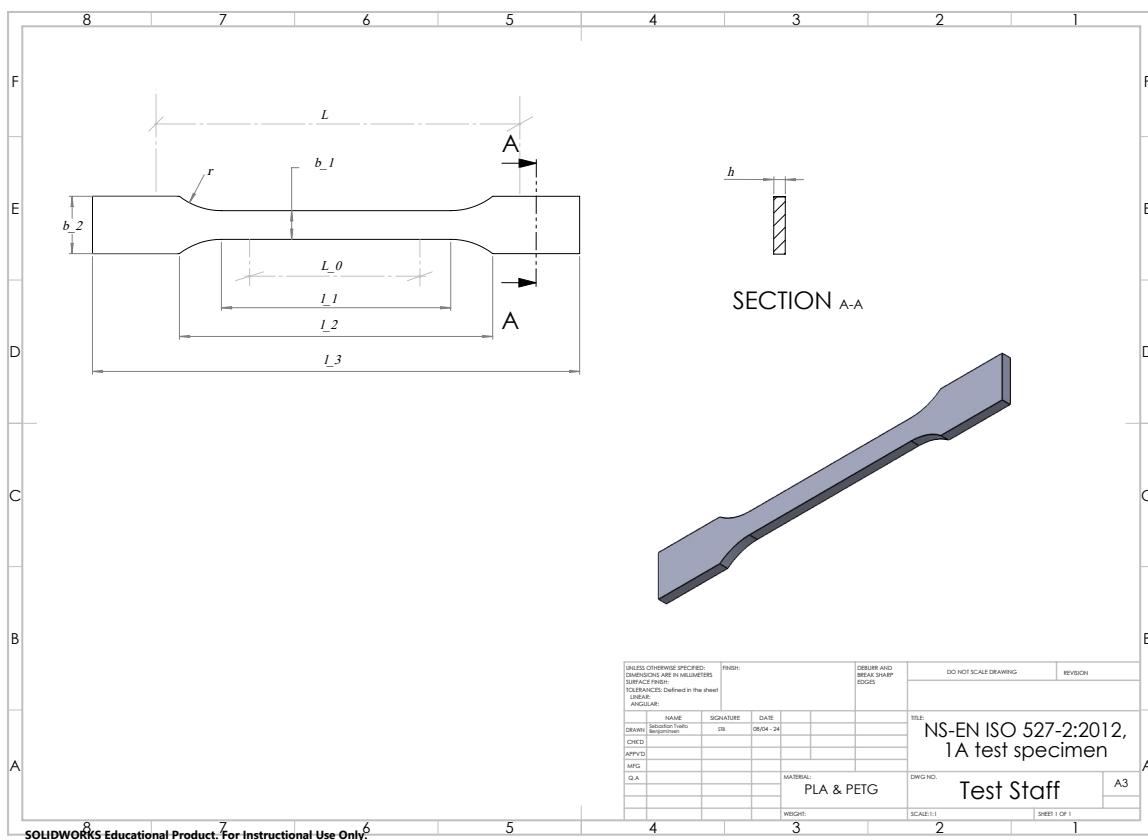


Figure 559: Sketch of the test specimen | STB

	Specimen type	1A
l_3	Overall length ^a	170
l_1	Lenght of narrow parallel-sided portion	80 ± 2
r	Radius	24 ± 1
l_2	Distance between broad parallel-sided portion ^b	109.3 ± 3.2
b_2	Width at ends	20.0 ± 0.2
b_1	Width at narrow portion	10.0 ± 0.2
h	Preferred thickness	4.0 ± 0.2
L_0	Gauge length	75.0 ± 0.5
L	Initial distance between grips	115 ± 1

Table 43: Description of the letters in fig 559 (every measurement is in millimeter) | STB

We have chosen that the test report shall include the information; "*Tensile test: ISO 527-2/type of test specimen/testing speed in millimeters per minute*".

R Simple CAD

During the first week of the bachelor assignment excitement was increased. Because of the excitement, the machine team started the Additive Manufacturing process creating a simple prototype, fig 560, using SW and FDM.

The prototype made the group visually from the machine team perspective prepared for the possible outcome in the project.

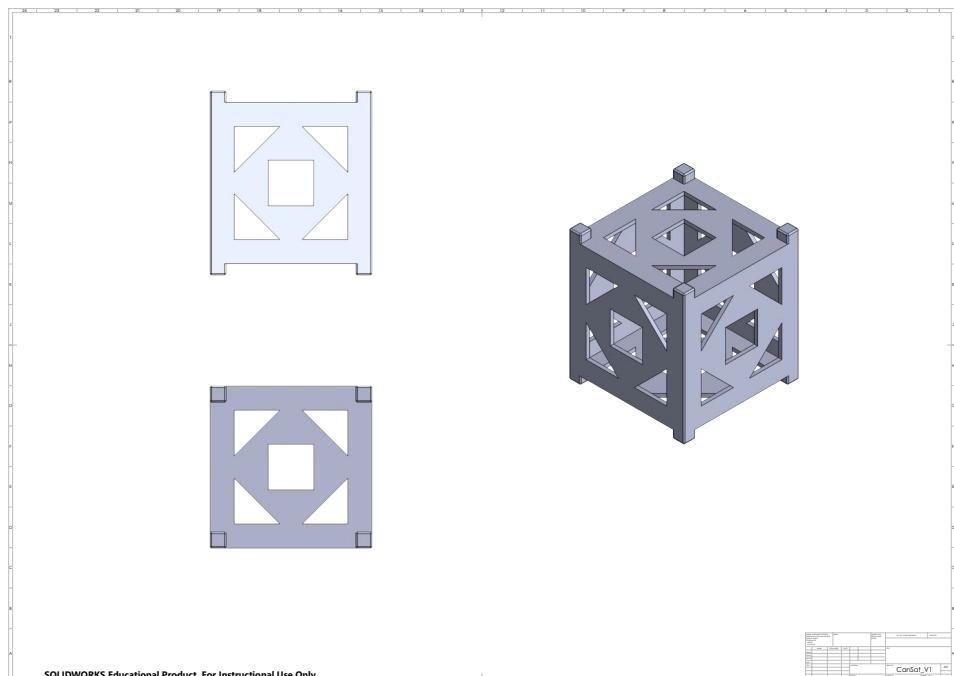


Figure 560: Simple CAD design | STB

An important note; the prototype described in this appendix was only used for this case.

S Hydrostatic pressure distance in gasses

In the troposphere, the temperature and pressure will decrease when the altitude increase. To get a mathematical equation, we assume the pressure gradient force to be approximately balancing the gravity force. [173]

Therefore, equation used for calculation the pressure surrounding the C.R.A.F.T-Sat before it's in the lower stratosphere is the hydro-static equitation (with the assumption of only one variable and it's a ideal gas) for fluid at rest:

$$\frac{dP(z)}{dz} = -\rho g$$

Where:

- $\frac{dP}{dz}$ is a derivitve function of pressure describing the change with respect to vertical distance (altitude)
- ρ is the density of the fluid
- g is the acceleration due to gravity

It's also easier to use the ideal gass law:

$$P = \rho RT \rightarrow \rho = \frac{P}{RT}$$

Where

- R is a gass constant and
- T is the absolute temperature in Kelvin; $T = T_{\text{in celcius}} + 273.15$

By combining the formulas the new equation is:

$$\frac{dP(z)}{dz} = \left(\frac{-P}{RT}\right) g$$

By using the new equation, integral for the pressure with an assumption that the temperature of the atmosphere is uniform, *isothermal atmosphere*, the integral can be solved as:

$$\int \frac{dP}{P} = \frac{-g}{R} \int \frac{dz}{T} \rightarrow \ln|P_1| - \ln|P_0| = \frac{-g}{RT}(z_1 - z_0) + C$$

$$\rightarrow P_1 = e^{\ln|P_0| - \frac{g}{RT}(z_1 - z_0) + C} = P_0 \cdot e^{\frac{-g}{RT}(z_1 - z_0)} C$$

By assuming the constant value C, won't interact in the equation ($C = 1$), the new equation is

$$P_1 = P_0 \cdot e^{\frac{-g}{RT}(z_1 - z_0)}$$

Since z_0 represents the height at sea level and z_1 the height in the troposphere ,the equation can change $z_1 - z_0 = h$ where the h stands for the height over sea level. The P_0 represents the absolute pressure at sea level and is equal $P_0 = P_{\text{gauge}} + P_{\text{atm}}$. Since we won't have any pressure beside the atmospheric pressure, gauge pressure is equal zero and the pressure we are using is the atmospheric pressure:

$$P(h) = P_{\text{atm}} \cdot e^{\frac{-g}{RT}h} \quad (54)$$

T System Interface diagram

In the start of the bachelor assignment it was important to understand the work between the group members and which group work necessary before important conclusion in the group related to common tasks was made. The *system interface diagram*, fig 561 describes systems related to the three team groups in the bachelor group. The team groups are *electrionical development*, *software development* and *mechanical product development*. The subsystems was not finished, but gave a illustration about the necessary work between the group members and at which stage a group verification had to be used before buying components.

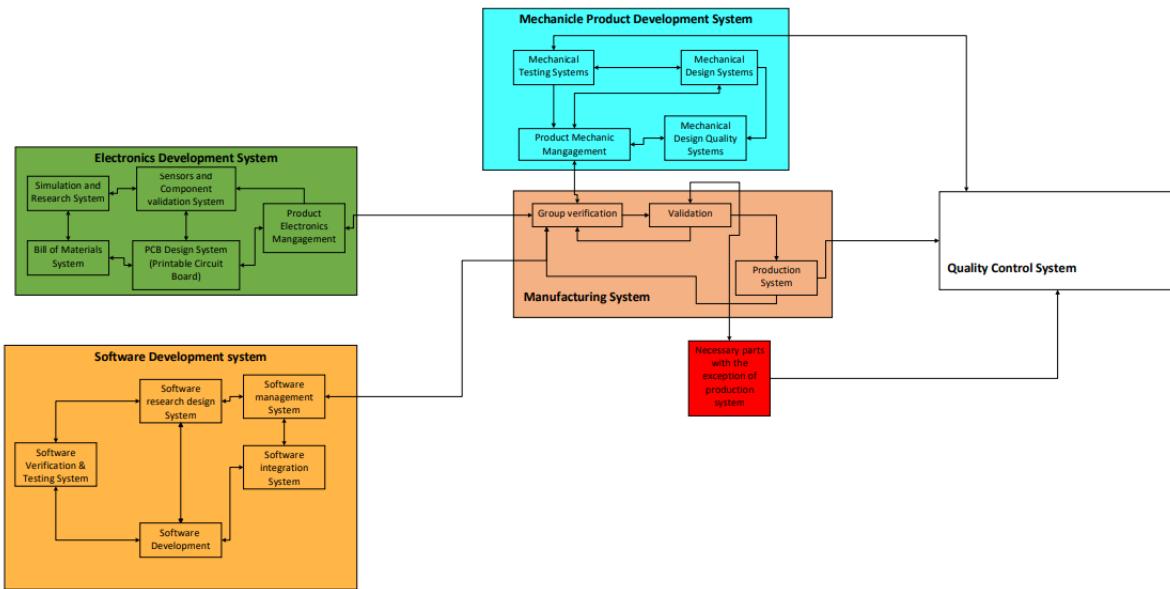


Figure 561: System Interface Diagram | STB

One of the important systems which was designed by a team member from all of the team groups was the production system, fig 562. The system gives a illustration of the work required between approved production and quality check

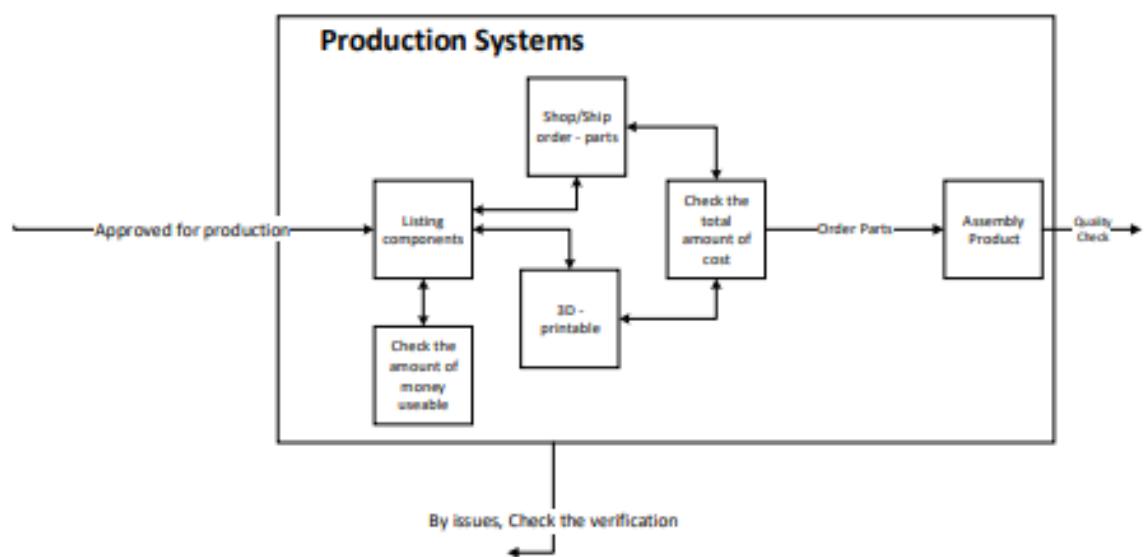


Figure 562: Production system | STB

U Simple FEA guide used with Vestsiden secondary school

The document contains a description about what FEM chosen as an desired FEA for the students at Vestsiden secondary school. The FEA is made using SWS and the C.R.A.F.T-Sat design illustrated in fig 563. FEA report is included under this appendix.

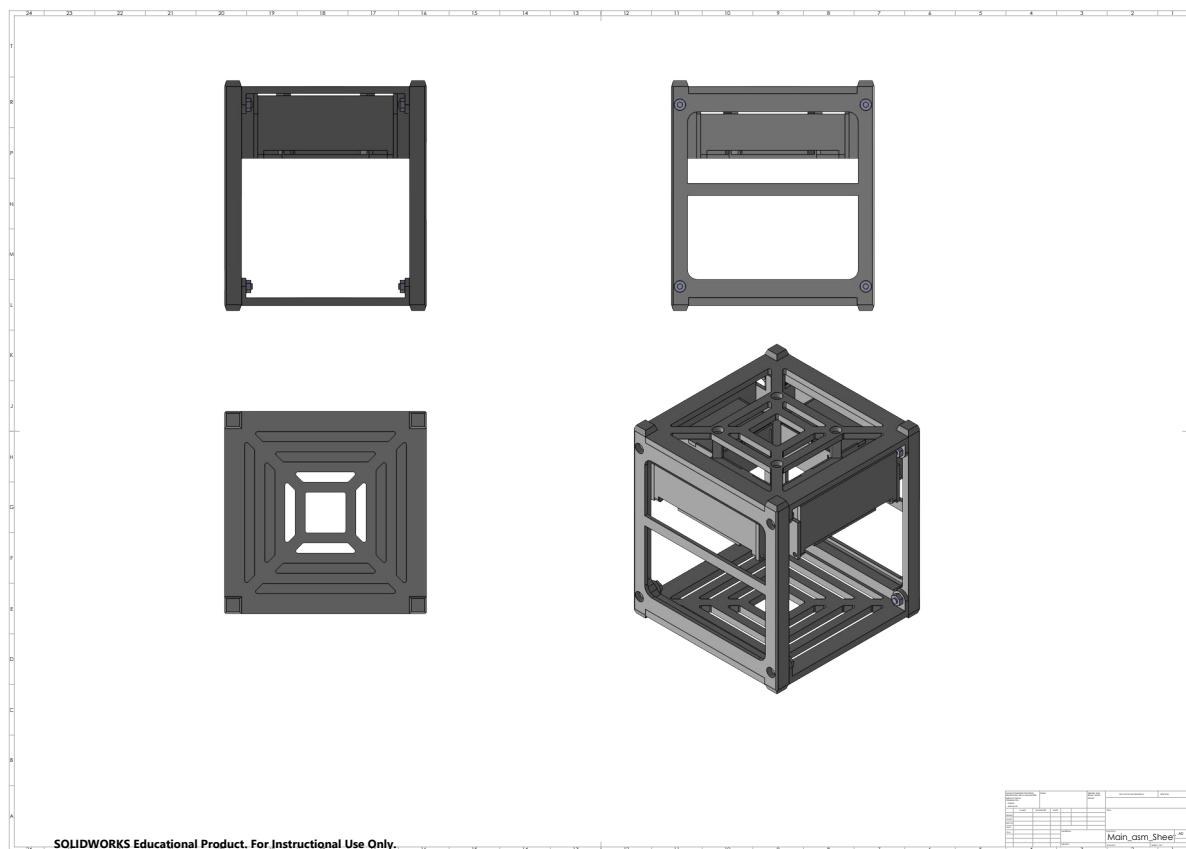


Figure 563: Simple sheet of the C.R.A.F.T-Sat design at date: 08. April 2024 | STB

To design a manual for a student group with less or no background with FEA simple parts are used (not assembly) with a sufficient description required for those students to understand what's done one the selected parts, why it's done on those parts and how a FEA-static is done on the selected components.

Fig 564 shows all of the part on the current state of C.R.A.F.T-Sat with a Bill of material (BOM). This gives the opportunity to select the parts we assume as critical components and will use for a FEA.

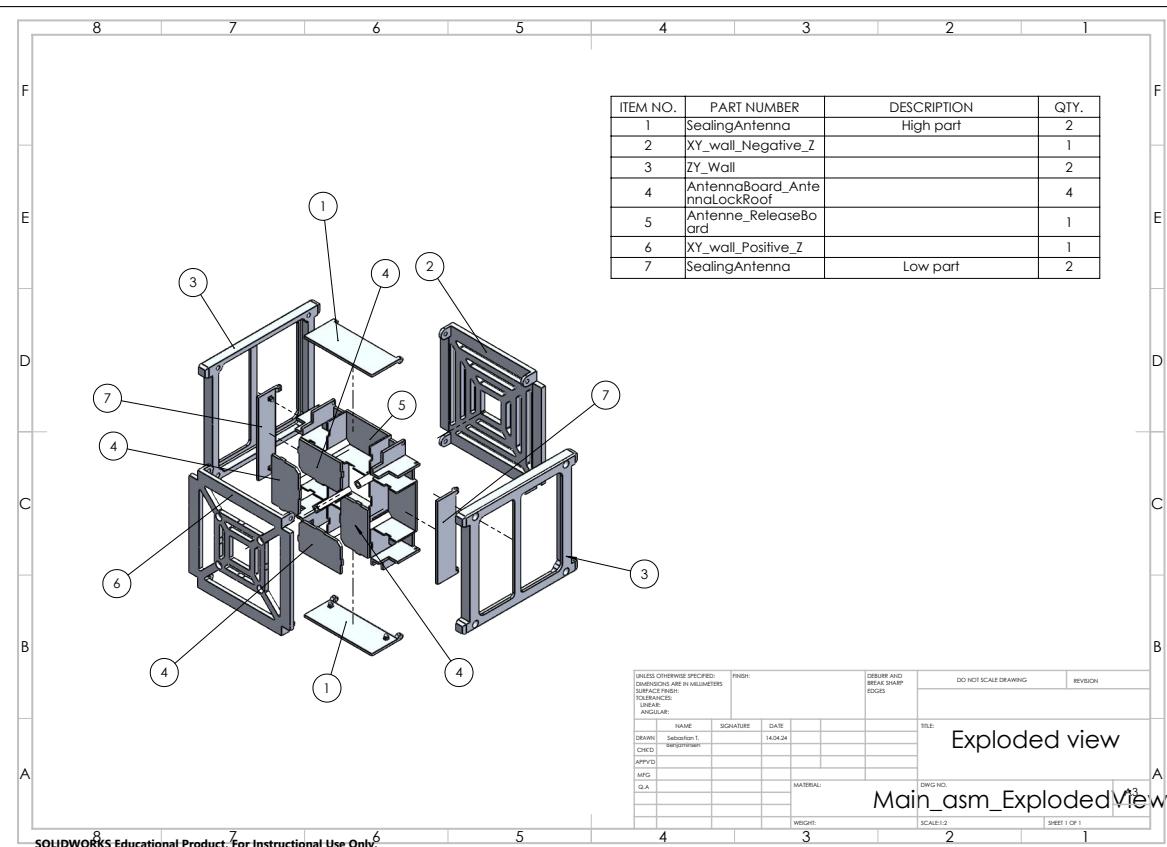


Figure 564: Simple exploded view of the C.R.A.F.T-Sat design at date: 08. April 2024 | STB

STB|

U.1 What need's to be tested

The C.R.A.F.T-Sat will experience different environments which is determined as factors deciding if the frame and release mechanism design is operative during a realistic test. But, at this date; 08. April 2024, the design and data gathering system placed in the frame is not finished and therefore, some of required FEA will not be displayed. Instead of showing a reliable thermal, vacuum and frequency FEA, a simple static FEA with no disturbing of temperature, pressure, and other variable factors. Therefore, the result from this test is not equal the test of the finished product, but will be a simple description of how a FEA is done with the C.R.A.F.T-Sat as the simple test product.

The components we select as critical at this state and a simple sheet of those components are XY Wall-Negative Z axis (fig 565), XY Wall-Positive Z axis (fig 566) and ZY Wall (fig 567).

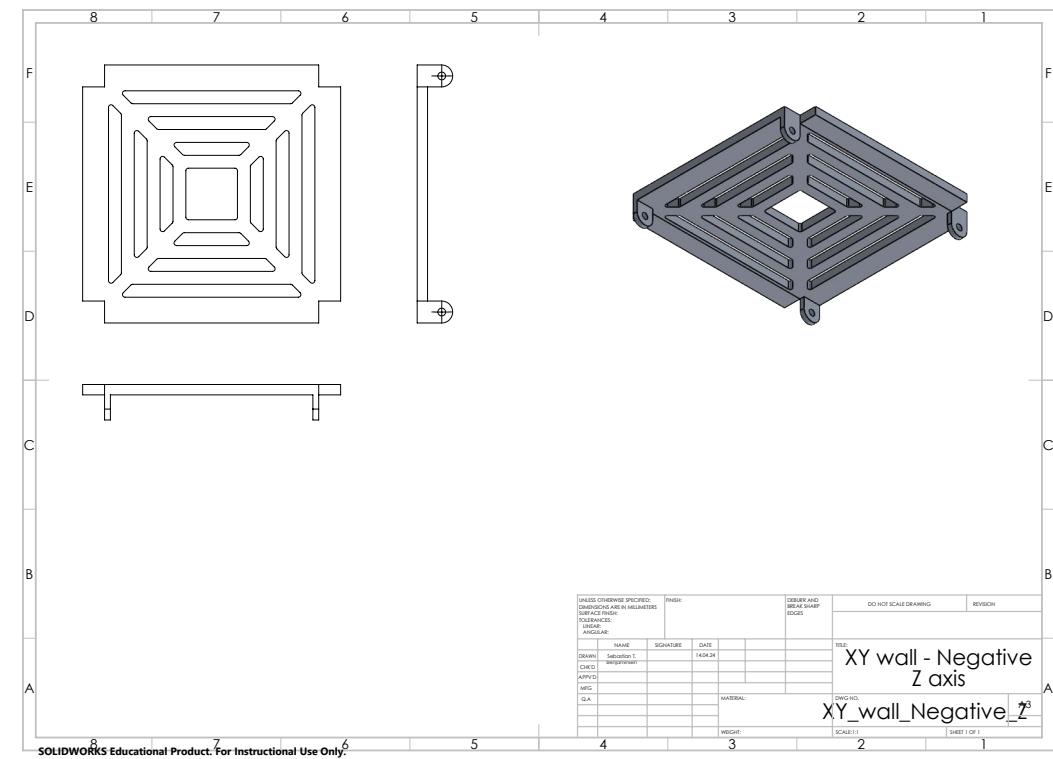


Figure 565: Date 08. April 2024; XY Wall-Negative Z axis | STB

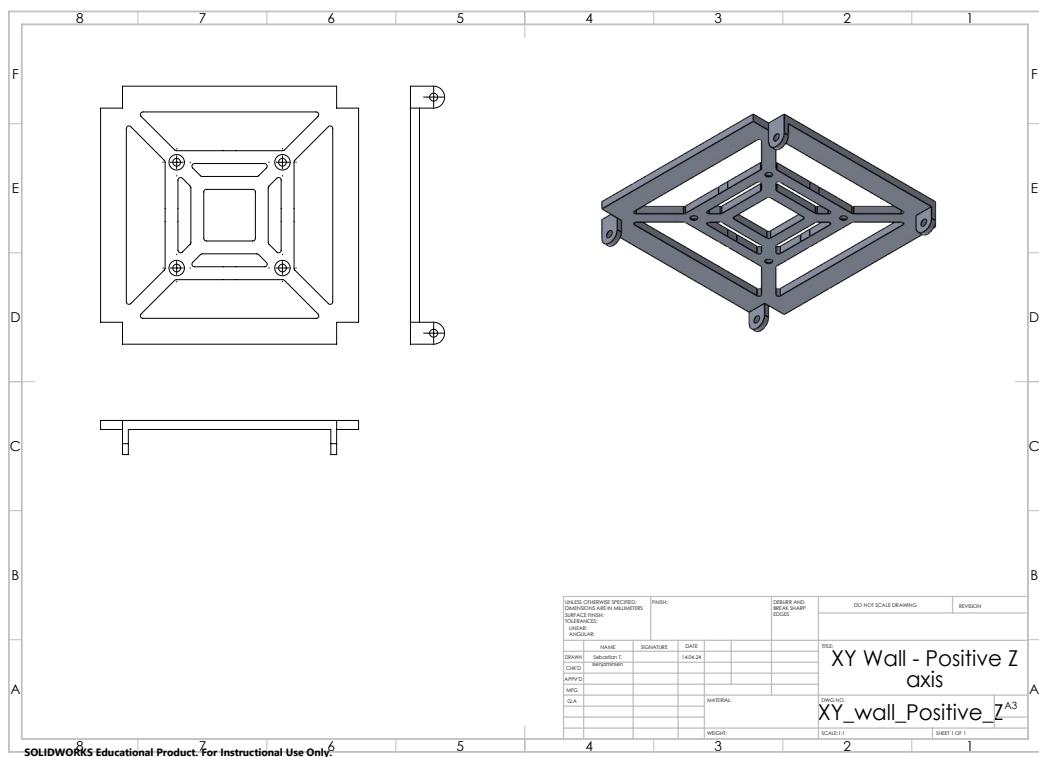


Figure 566: Date 08. April 2024; XY Wall-Positive Z axis | STB

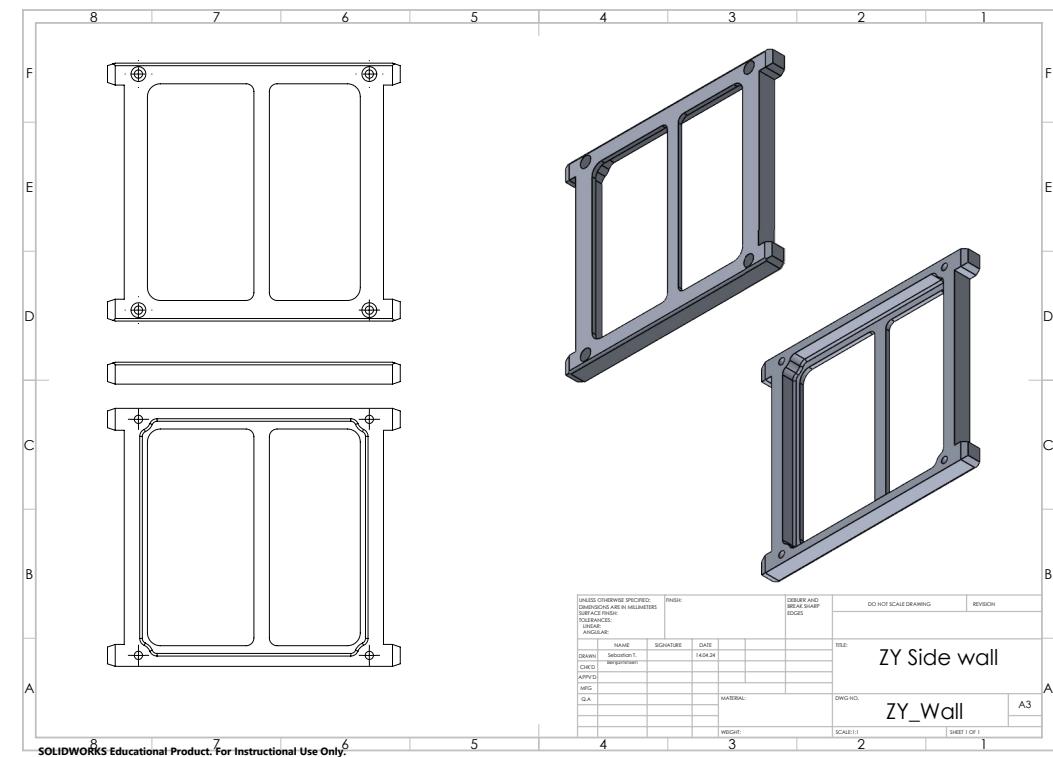


Figure 567: Date 08. April 2024; ZY Wall | STB

STB|

U.2 How can we test it

The static FEM on this product is decided to illustrate the face when it's standing still in the air, fig 568, it's of course not realistic since the product is designed to increase it's upward movement until the balloon cracks. But to decide if it's capable of being in the air connected with a balloon and internal equipment's giving weight inside the frame, we chose simple connections on the frame to illustrate the connection to the these cases.

A important state; the analyse related to this guide will not include factors such as side forces related to wind-cast and other effects during the real use of C.R.A.F.T-Sat.

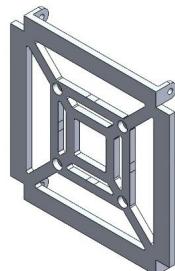
The balloon connections will be defined as fixtures on the frame/there balloon thread will be connected. And the force will be placed in the center of gravity location of the frame; both with and without the electronical devices (this is decided if the data related to mass from the electronical devices are available)

U.3 Why was it tested

The reason a static analysis is a necessary FEA are to decide if the C.R.A.F.T-Sat design are capable to withstand the simple outcomes of hanging on a balloon and carrying a specific amount of weight.

If the result shows a outcome of not being in the state we have decided as necessary a decision of how to fix the design will be the result.

C.R.A.F.T



Description

No Data

Simulation of XY_wall_Positive_Z

Date: mandag 15. april 2024

Designer: Sebastian Tveito Benjaminsen

Study name: Simple Static Analysis

Analysis type: Static

Table of Contents

Description	1
Assumptions.....	2
Model Information	2
Study Properties.....	3
Units	3
Material Properties	4
Loads and Fixtures	4
Connector Definitions	5
Interaction Information	5
Mesh information.....	5
Sensor Details.....	6
Resultant Forces	6
Beams.....	6
Study Results	7
Conclusion.....	10



SOLIDWORKS

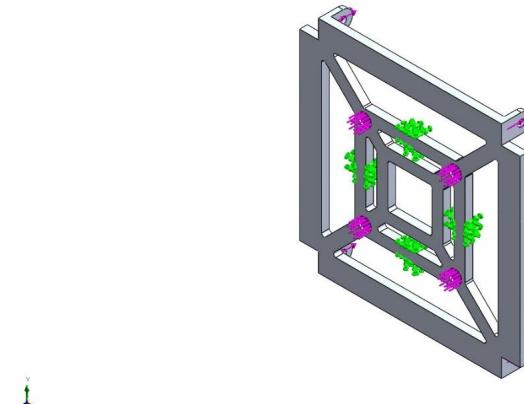
Analyzed with SOLIDWORKS Simulation

Simulation of XY_wall_Positive_Z 1

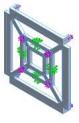
Assumptions

The component is placed in a stable condition hanging on to a balloon. Also, it's obstructed to all possible motion in all axes to simplify the analysis.

Model Information



Model name: XY_wall_Positive_Z
Current Configuration: Default

Solid Bodies			
Document Name and Reference	Treated As	Volumetric Properties	Document Path/Date Modified
Split Line2 	Solid Body	Mass:0,051453 kg Volume:1,90567e-05 m^3 Density:2 700 kg/m^3 Weight:0,50424 N	C:\Users\sebas\Downloads\FEM_SW-filer_Vestsiden_U_Skole\XY_wall_Positive_Z.SLDPRT Apr 15 01:01:57 2024



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of XY_wall_Positive_Z

2

Study Properties

Study name	Simple Static Analysis
Analysis type	Static
Mesh type	Solid Mesh
Thermal Effect:	On
Thermal option	Include temperature loads
Zero strain temperature	298 Kelvin
Include fluid pressure effects from SOLIDWORKS Flow Simulation	Off
Solver type	Automatic
Inplane Effect:	Off
Soft Spring:	Off
Inertial Relief:	Off
Incompatible bonding options	Automatic
Large displacement	Off
Compute free body forces	On
Friction	Off
Use Adaptive Method:	Off
Result folder	SOLIDWORKS document (C:\Users\sebas\Downloads\FEM_SW-filer_Vestsiden_U_Skole)

Units

Unit system:	SI (MKS)
Length/Displacement	mm
Temperature	Kelvin
Angular velocity	Rad/sec
Pressure/Stress	N/mm ² (MPa)



Material Properties

Model Reference	Properties	Components
	<p>Name: 6061 Alloy Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 55,1485 N/mm² Tensile strength: 124,084 N/mm² Elastic modulus: 69 000 N/mm² Poisson's ratio: 0,33 Mass density: 2,7 g/cm³ Shear modulus: 26 000 N/mm² Thermal expansion coefficient: 2,4e-05 /Kelvin</p>	SolidBody 1(Split Line2)(XY_wall_Positive_Z)
Curve Data:N/A		

Loads and Fixtures

Fixture name	Fixture Image	Fixture Details															
Fixed-1		Entities: 8 face(s) Type: Fixed Geometry															
Resultant Forces																	
<table border="1"> <thead> <tr> <th>Components</th><th>X</th><th>Y</th><th>Z</th><th>Resultant</th></tr> </thead> <tbody> <tr> <td>Reaction force(N)</td><td>2,20537e-06</td><td>-7,45058e-07</td><td>38,2852</td><td>38,2852</td></tr> <tr> <td>Reaction Moment(N.m)</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </tbody> </table>			Components	X	Y	Z	Resultant	Reaction force(N)	2,20537e-06	-7,45058e-07	38,2852	38,2852	Reaction Moment(N.m)	0	0	0	0
Components	X	Y	Z	Resultant													
Reaction force(N)	2,20537e-06	-7,45058e-07	38,2852	38,2852													
Reaction Moment(N.m)	0	0	0	0													

Load name	Load Image	Load Details
Force-1		Entities: 8 face(s) Reference: Edge< 1 > Type: Apply force Values: ---; ---; -0,488 kgf

Connector Definitions

No Data

Interaction Information

No Data

Mesh information

Mesh type	Solid Mesh
Mesher Used:	Standard mesh
Automatic Transition:	Off
Include Mesh Auto Loops:	Off
Jacobian points for High quality mesh	16 Points
Element Size	5 mm
Tolerance	0,25 mm
Mesh Quality	High

Mesh information - Details

Total Nodes	8653
Total Elements	4105
Maximum Aspect Ratio	18,708
% of elements with Aspect Ratio < 3	70,5
Percentage of elements with Aspect Ratio > 10	0,195
Percentage of distorted elements	0
Time to complete mesh(hh:mm:ss):	00:00:02
Computer name:	



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of XY_wall_Positive_Z

5

Sensor Details

No Data

Resultant Forces**Reaction forces**

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N	2,20537e-06	-7,45058e-07	38,2852	38,2852

Reaction Moments

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N.m	0	0	0	0

Free body forces

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N	2,03231e-06	4,05206e-06	-0,000100455	0,000100557

Free body moments

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N.m	0	0	0	1e-33

Beams

No Data



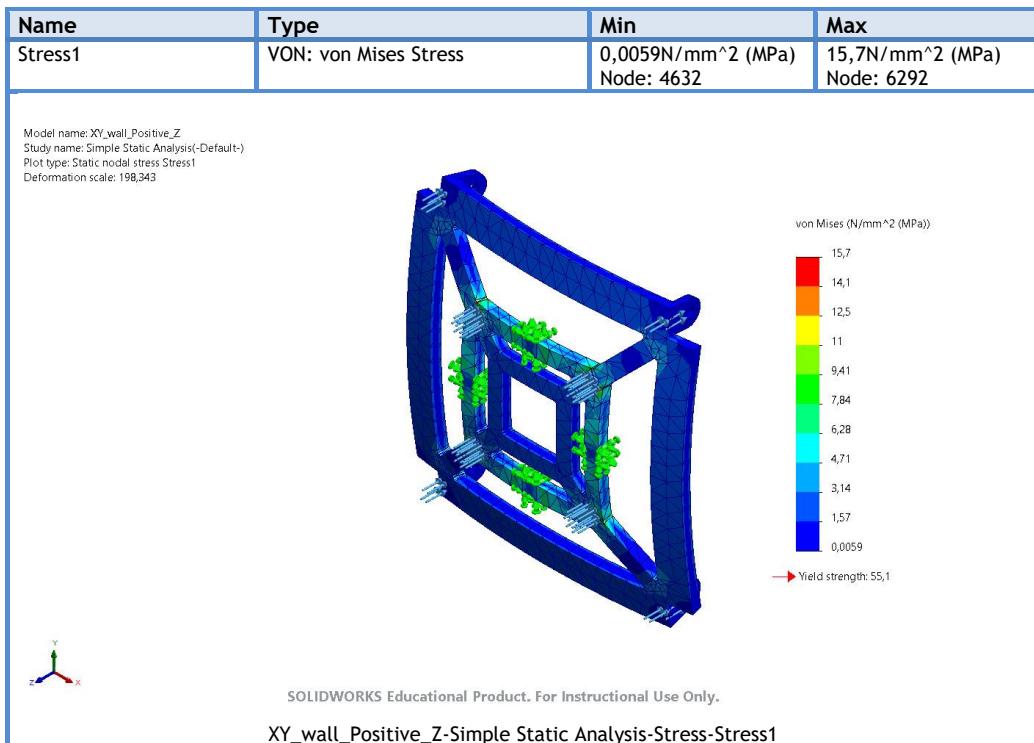
SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

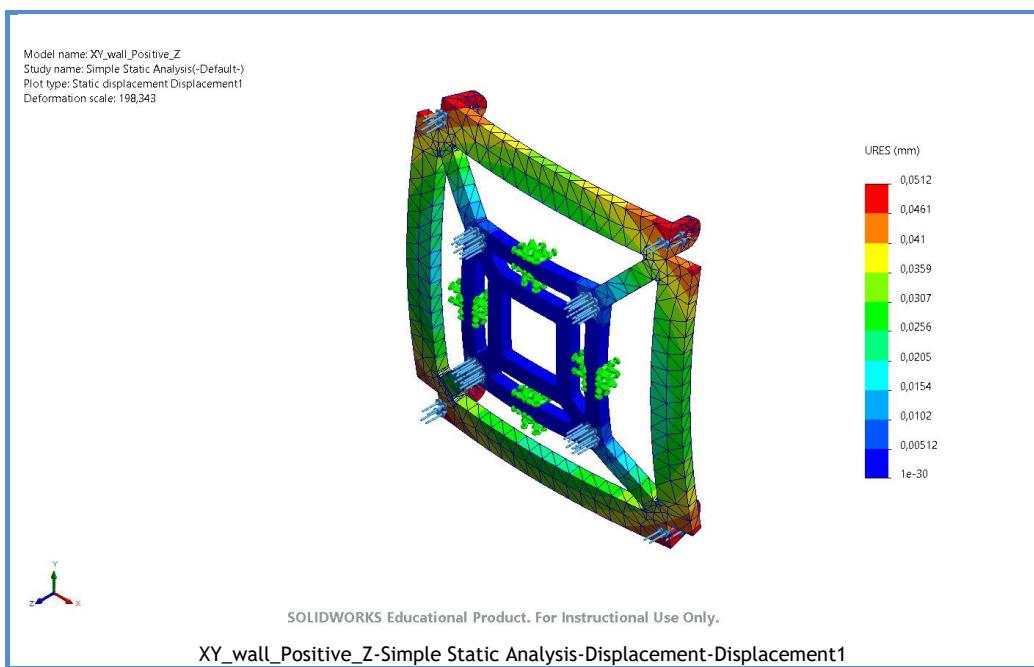
Simulation of XY_wall_Positive_Z

6

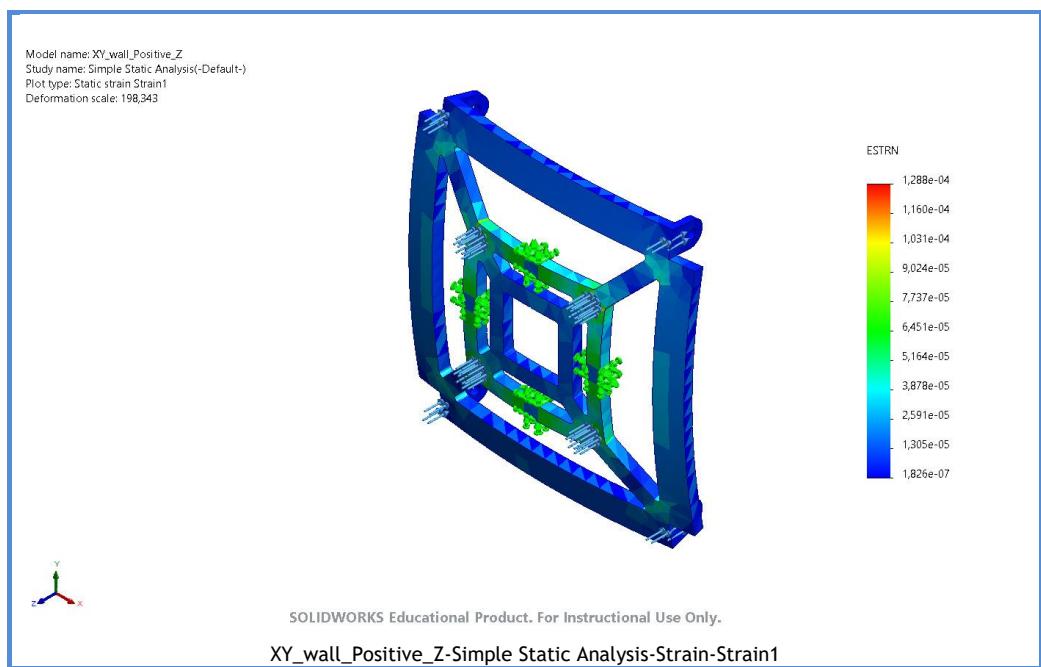
Study Results



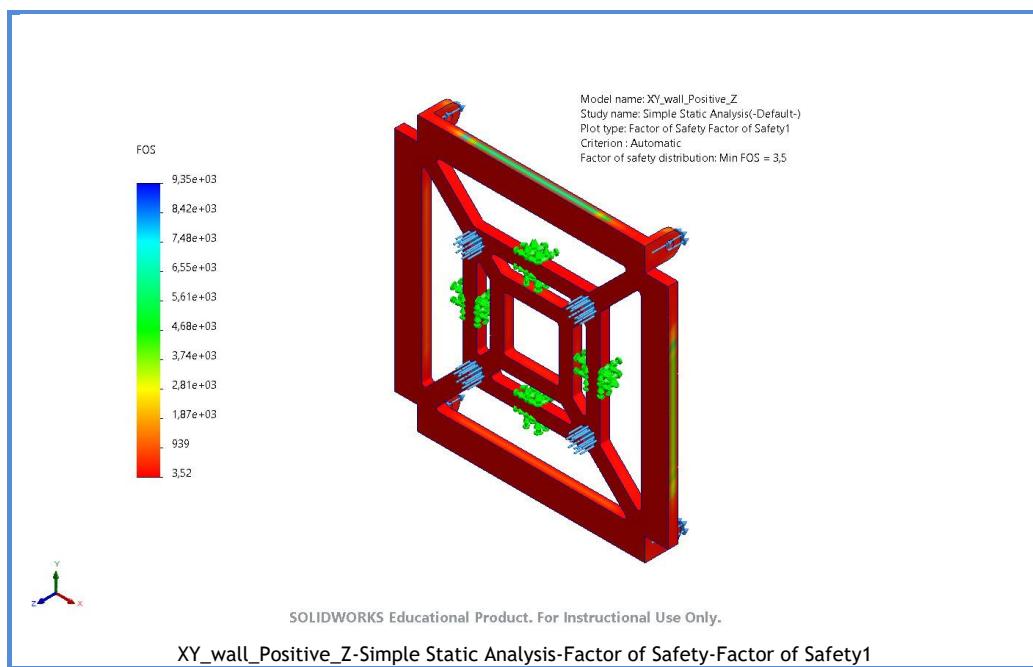
Name	Type	Min	Max
Displacement1	URES: Resultant Displacement	0mm Node: 289	0,0512mm Node: 6920



Name	Type	Min	Max
Strain1	ESTRN: Equivalent Strain	1,826e-07 Element: 1586	1,288e-04 Element: 1950



Name	Type	Min	Max
Factor of Safety1	Automatic	3,52 Node: 6292	9,35e+03 Node: 4632



Conclusion

According to the result of von Mises; the maximum stress occurred on the component is 15.7 MPa which is much less than the yield strength (**Note:** when the stress on a material is higher than yield strength the change refer as displacement will be constant when the force is removed and the product will be weaker) of the material. From this; the analyzed factor of safety gives a the lowest value 3.52 which also shows that current design, with chosen material could easily withstand the given forces and gives possibilities of material removal if the cost at this point is high and a requirement is removal of cost in material.



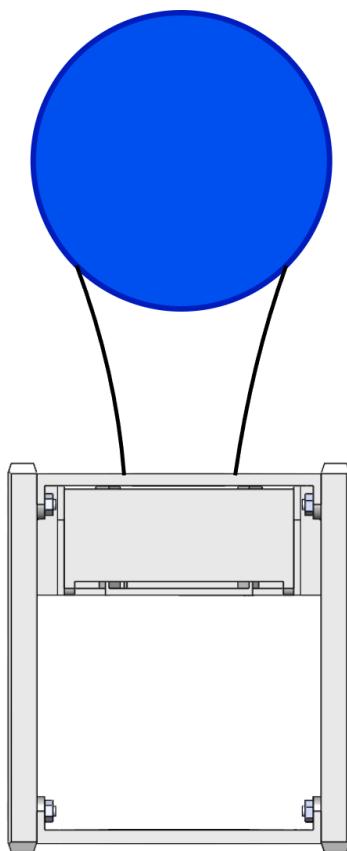


Figure 568: C.R.A.F.T-Sat with balloon| STB

V Finite Element Analysis - Reports

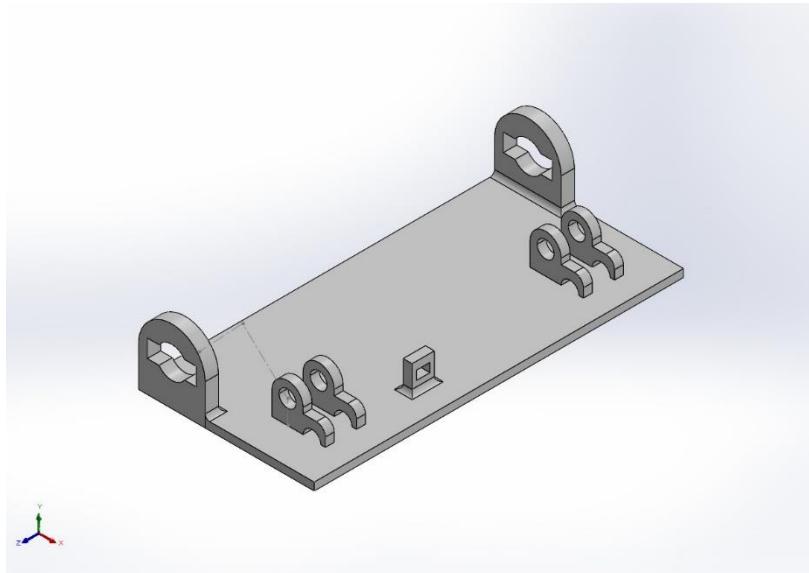
This appendix section represents all the FEA made for the final C.R.A.F.T-Sat.

V.1 Static analysis

Under this section is a descriptions and result of static analyis with the *SealinAntenna* and the *main_asm*.



C.R.A.F.T



Description

The simulation represents a sealing with the interacting interfaces when it's open. The material used is PETG. The load used is representing springs with an angle pushing a component against the wall inside two holes. Since the given data about the springs is in British units, the chosen spring force is in lbf, but the other data is in SI units.

Simulation of part: Sealing

Date: Wednesday 8. May 2024

Designer: STB

Study name: Static - Open sealing - pin - Positive Y

Analysis type: Static

Table of Contents

Description	1
Assumptions	2
Model Information	3
Study Properties	4
Units	4
Material Properties	5
Loads and Fixtures	6
Connector Definitions	7
Interaction Information	7
Mesh information	8
Sensor Details	9
Resultant Forces	9
Beams	9
Study Results	10
Conclusion	14



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of SealingAntenna 1



Assumptions

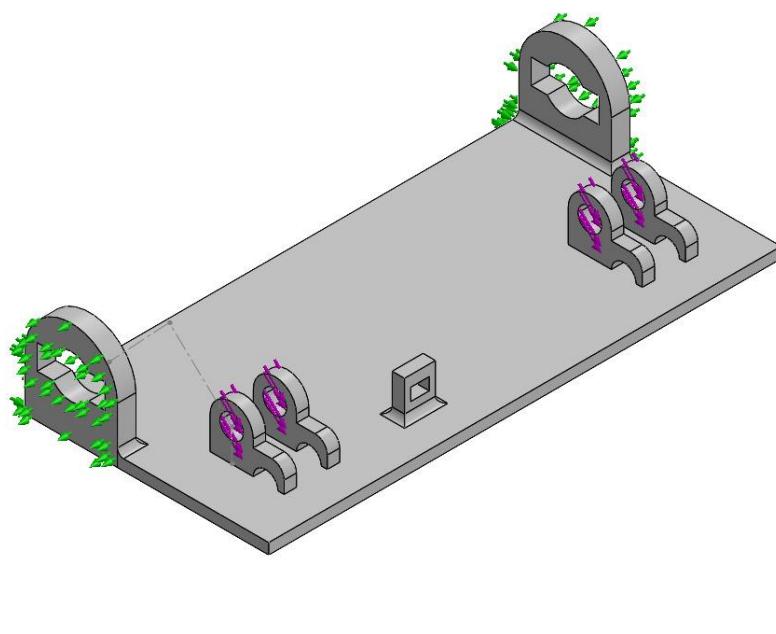
We assume this component to be one of the crucial parts in the C.R.A.F.T-Sat assembly. The fixture's used is decided as important and because of the amount of data used in a FEA chosen as necessary. Other fixtures could be placed but isn't chosen as important because of the data required.

It's also assumed the most important load are from the springs, therefore, no other loads are used. The 3D printed component is not a solid body (100% infill) but we assume that the values found in the analysis is approximately the same as in real life.





Model Information



Model name: SealingAntenna
Current Configuration: Sealing - Positive Y

Solid Bodies

Document Name and Reference	Treated As	Volumetric Properties	Document Path/Date Modified
Split Line3 	Solid Body	Mass:0,00973089 kg Volume:7,10284e-06 m^3 Density:1 370 kg/m^3 Weight:0,0953628 N	C:\Users\sebas\OneDrive - USN\TheLast_SW_DesignInterval\SealingAntenna.SLDprt May 8 11:20:40 2024





Study Properties

Study name	Static - Open sealing - pin - Positive Y
Analysis type	Static
Mesh type	Solid Mesh
Thermal Effect:	On
Thermal option	Include temperature loads
Zero strain temperature	298 Kelvin
Include fluid pressure effects from SOLIDWORKS Flow Simulation	Off
Solver type	Automatic
Inplane Effect:	Off
Soft Spring:	Off
Inertial Relief:	Off
Incompatible bonding options	Automatic
Large displacement	Off
Compute free body forces	On
Friction	Off
Use Adaptive Method:	Off
Result folder	SOLIDWORKS document (C:\Users\sebas\OneDrive - USN\TheLast_SW_Design\Intervall)

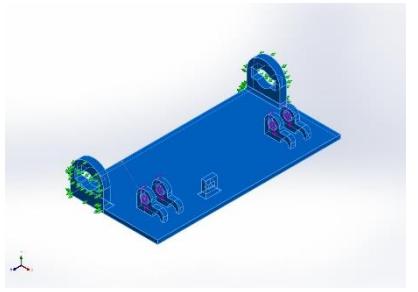
Units

Unit system:	SI (MKS)
Length/Displacement	mm
Temperature	Kelvin
Angular velocity	Rad/sec
Pressure/Stress	N/mm ² (MPa)





Material Properties

Model Reference	Properties	Components
	<p>Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 11,58 N/mm² Tensile strength: 37,53 N/mm² Elastic modulus: 3 001,59 N/mm² Poisson's ratio: 0,38 Mass density: 1,37 g/cm³</p>	SolidBody 1(Split Line3)(SealingAntenna)
Curve Data:N/A		





Loads and Fixtures

Fixture name	Fixture Image	Fixture Details
Fixed Hinge-1		Entities: 4 face(s) Type: Fixed Hinge
Resultant Forces		
Components	X	Y
Reaction force(N)	-71,4595	23,0719
Reaction Moment(N.m)	0	0
Resultant Forces		
Components	X	Y
Reaction force(N)	23,5284	0
Reaction Moment(N.m)	0	0
Resultant Forces		
Components	X	Y
Reaction force(N)	-4,399	2,36966
Reaction Moment(N.m)	0	0





Load name	Load Image	Load Details
Force-1		Entities: 4 face(s) Type: Apply force Values: ---; ---; 3,79 lbf
Force-2		Entities: 4 face(s) Type: Apply force Values: ---; ---; 3,79 lbf

Connector Definitions

No Data

Interaction Information

No Data



Mesh information

Mesh type	Solid Mesh
Mesher Used:	Blended curvature-based mesh
Jacobian points for High quality mesh	16 Points
Maximum element size	1,91982 mm
Minimum element size	0,639934 mm
Mesh Quality	High

Mesh information - Details

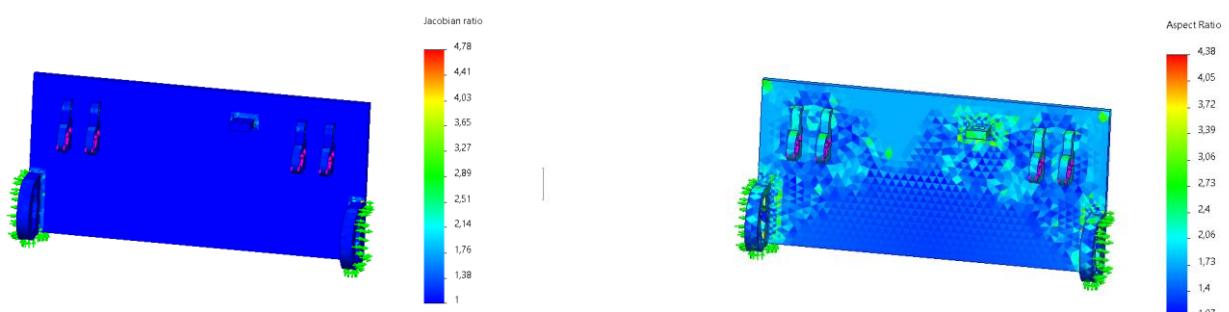
Total Nodes	21232
Total Elements	11041
Maximum Aspect Ratio	4,3846
% of elements with Aspect Ratio < 3	98,9
Percentage of elements with Aspect Ratio > 10	0
Percentage of distorted elements	0
Time to complete mesh(hh:mm:ss):	00:00:12
Computer name:	

Mesh Quality Plots

Both Aspect ratio and Jacobian ratio percentage value is less than 10%. Therefore, we conclude that the mesh quality is useable for the static analysis.

Model name: SealingAntenna
Study name: Static - Open sealing - pin - Positive Y(-Sealing - Positive Y-)
Plot type: Jacobian Ratio Element Distribution Quality2

Model name: SealingAntenna
Study name: Static - Open sealing - pin - Positive Y(-Sealing - Positive Y-)
Plot type: Aspect ratio Quality3





Sensor Details

No Data

Resultant Forces

Reaction forces

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N	-24,5856	23,0719	-2,31266e-05	33,716

Reaction Moments

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N.m	0	0	0	0

Free body forces

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N	0,00127124	0,0317797	-0,00749714	0,0326768

Free body moments

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N.m	0	0	0	1e-33

Beams

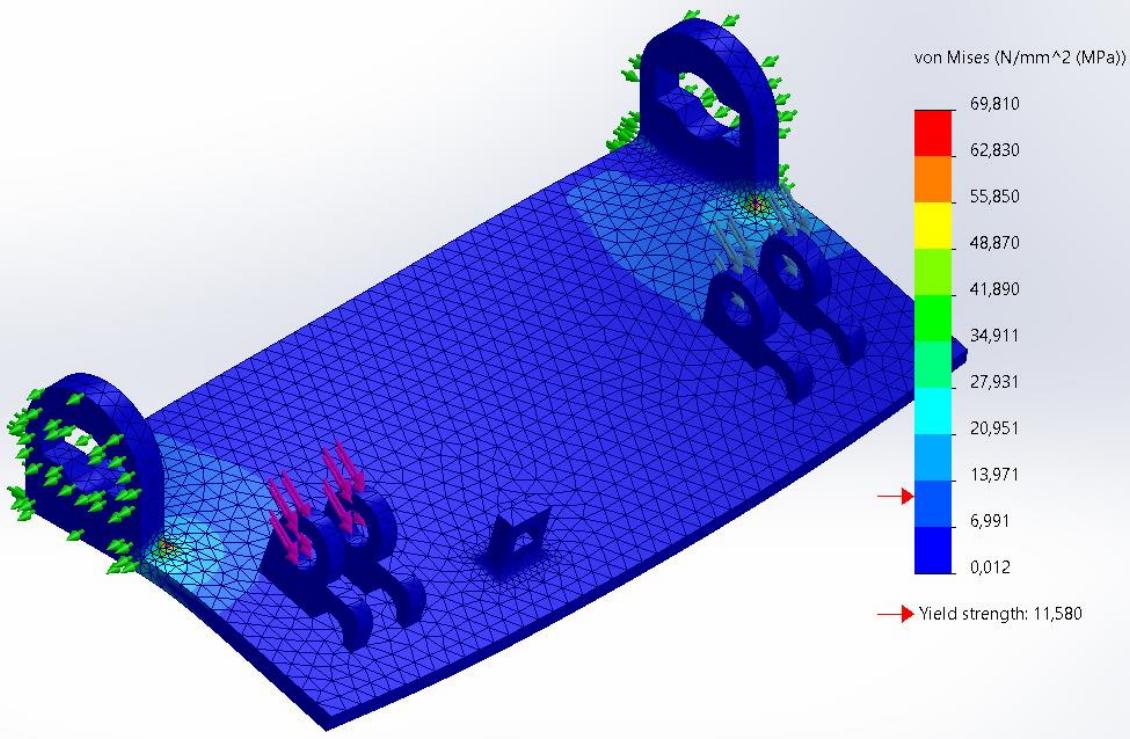
No Data



Study Results

Name	Type	Min	Max
Stress1	VON: von Mises Stress	0,012N/mm ² (MPa) Node: 7350	69,810N/mm ² (MPa) Node: 3055

Model name: SealingAntenna
 Study name: Static - Open sealing - pin - Positive Y(-Sealing - Positive Y-)
 Plot type: Static nodal stress Stress1
 Deformation scale: 3,96928

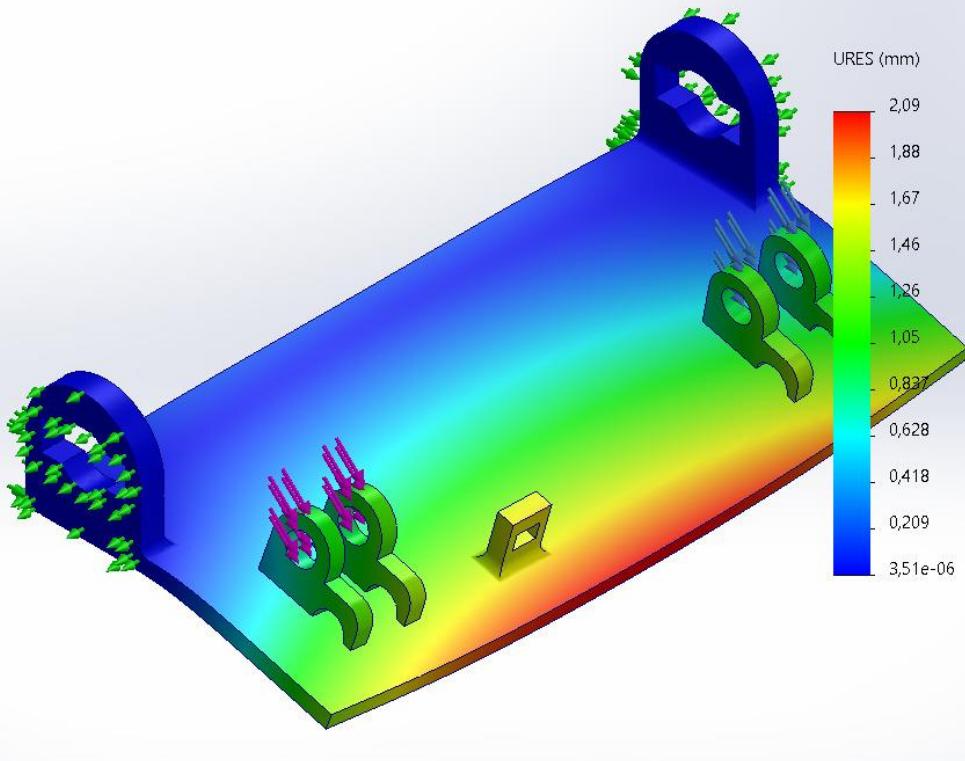


SOLIDWORKS Educational Product. For Instructional Use Only.

SealingAntenna-Static - Open sealing - pin - Positive Y-Stress-Stress1

Name	Type	Min	Max
Displacement1	URES: Resultant Displacement	3,51e-06mm Node: 1669	2,09mm Node: 455

Model name: SealingAntenna
 Study name: Static - Open sealing - pin - Positive Y(-Sealing - Positive Y-)
 Plot type: Static displacement Displacement1
 Deformation scale: 3,96928



SOLIDWORKS Educational Product. For Instructional Use Only.

SealingAntenna-Static - Open sealing - pin - Positive Y-Displacement-Displacement1

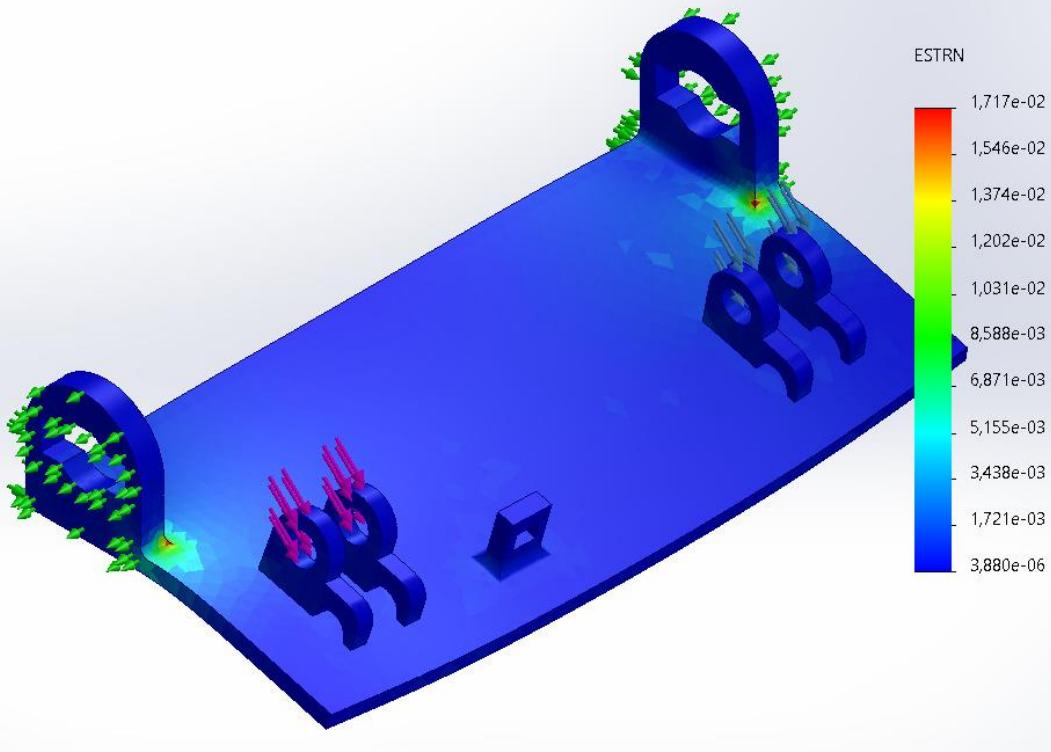
Name	Type	Min	Max
Strain1	ESTRN: Equivalent Strain	3,880e-06 Element: 1808	1,717e-02 Element: 5774

Model name: SealingAntenna

Study name: Static - Open sealing - pin - Positive Y(-Sealing - Positive Y-)

Plot type: Static strain Strain1

Deformation scale: 3,96928



SOLIDWORKS Educational Product. For Instructional Use Only.

SealingAntenna-Static - Open sealing - pin - Positive Y-Strain-Strain1

Name	Type	Min	Max
Factor of Safety1	Automatic	0,166 Node: 3055	990 Node: 7350

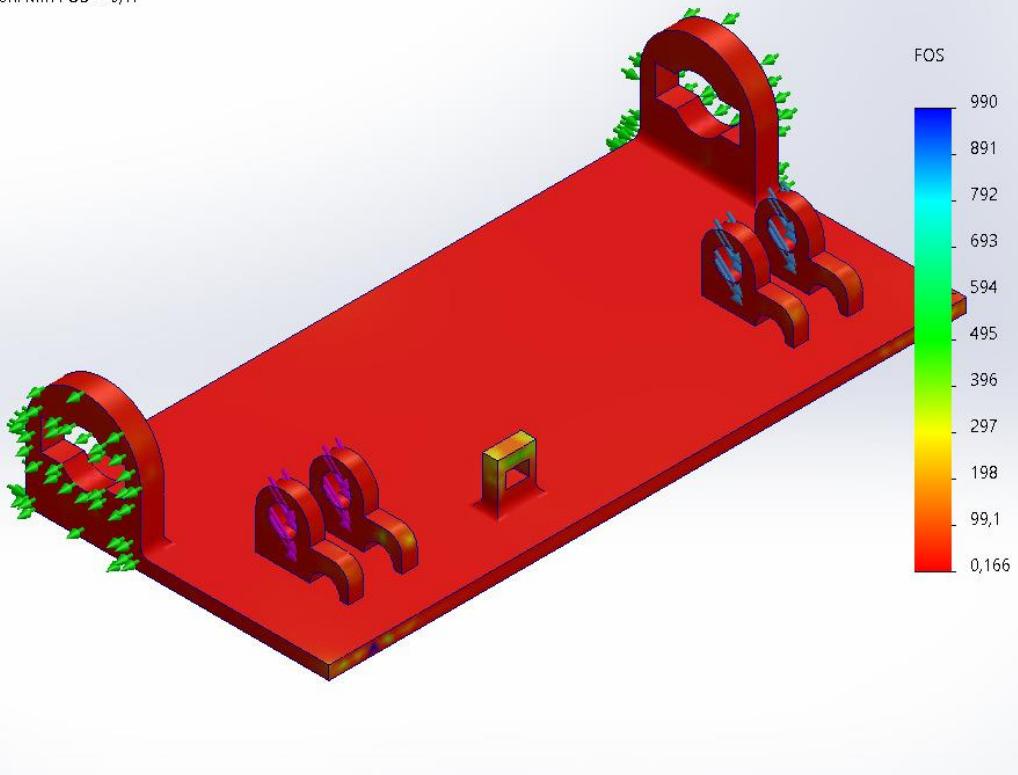
Model name: SealingAntenna

Study name: Static - Open sealing - pin - Positive Y(-Sealing - Positive Y-)

Plot type: Factor of Safety/Factor of Safety1

Criterion: Automatic

Factor of safety distribution: Min FOS = 0,17



SealingAntenna-Static - Open sealing - pin - Positive Y-Factor of Safety-Factor of Safety1

Conclusion

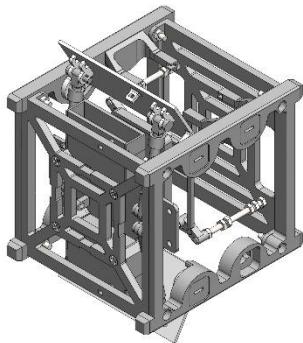
According to a calculation using vonMises theorem, the *spring force* on the *sealing* with the fixtures representing the sealing interfacing the frame components when it's open will have stresses up to six times higher than the yield strength. Compared with tensile testing data; tensile strength equal 37.53 MPa, the structure will possibly have cracks in those areas. Also, the factor of safety "FOS" is at the lowest value less than one, which is equal a state where it will be permanently deformed.

An important difference between the stress and deformation is that the placement of where the structure will deform most is also the placement where the stress is low/less than PETG yield strength. Therefore, if the load on the area with the highest deformation is removed, the structure will theoretically return to be equal with the structure before load occurred. But, since the stress is high at a point close to those areas, it could be cracks reaching those places with high deformation and could theoretically make the cracks larger, depending on the direction of the crack and the leading possibilities with the infill pattern on the component. The places where it will be permanent deformation is at the hinges which gives a large risk of the hinge and plate will be teared apart when the sealing is open with the spring pushing on it.

Therefore, because of the large risks with tearing the design of the sealing is not suited to be used without knowledge of the risks and that the product will be permanently deformed and have large possibilities of cracks leading to destruction during use at the state where the sealing is open, and the springs is pushing the sealing at against the component ZY-Wall.



C.R.A.F.T



1

Description

The analysis illustrates a steady state simulation with non-temperature related interactions in the simulation. The simulation has an Aspect ratio with a small amount percent higher than 10, but with mostly under 3 percent. Because the simulation required a large amount of data, we have chosen to simplify it, and accept the small amount of percent higher than 10 but the location of the values higher than 10 percent has been checked. Components such as electronical circuit boards and cables, bolts, and nuts with threads, have been removed. Some bolts have been included but they don't have threads. The model has only a design representing balloon mounting and not parachuting. The model can be re-designed, but it's not our object at this moment to do so.

Simulation of assembly: main_asm

Date: Sunday 12. May 2024

Designer: STB

Study name: Static Analysis

Analysis type: Static

Table of Contents

Description	1
Assumptions & model information	2
Study Properties.....	3
Units	3
Material Properties	4
Loads and Fixtures	11
Connector Definitions	12
Interaction Information	13
Mesh information.....	18
Sensor Details.....	19
Resultant Forces	19
Beams	20
Study Results	21
Conclusion.....	24



SOLIDWORKS

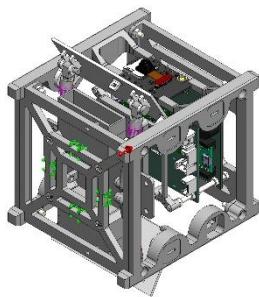
Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 1

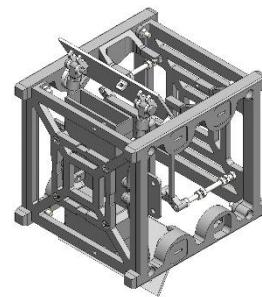
Assumptions & model information

We assume the model has four ropes mounted on the XY Positive Z part, and there are non-environmental forces interacting the model (example of environmental force; wind blowing on the sealing pushing the spring). We have also assumed the antenna is released and the fishing rope burned has disappeared. It's also assumed that the only force from the spring is on the spring part mounted on the sealing. Compared with the antenna board, the sealing is the only component, beside the assembly spring_asm which is rotating and is direct mounted to the spring. It's also assumed that the calculated value for the spring force is the correct/equal spring force from the bought springs.

The model is analyzed to represent a circumstance with none-electronical components and with an environment where none disturbing objects occurs. Since the model is a solid tight fitted with free components local interactions was made at the spring areas where components not touching each other due to none spring and thread parts. The bonded global interaction is used on the other components in the model. The model has components with threads and small sizes; the threads are removed, and some mechanical components (such as bolts, nuts, and springs) are not included in the analysis. The model represents the real sizes of the product, with PETG as the material for all the 3D printing parts. The total mass of the model analyzed is higher (we don't have the specific value at this moment) than the real product because we will not use 100% infill in 3D printing. This results with a higher gravitational force, $G = mg$, where m is the mass and g are the gravitational coefficient (a constant value).



Original Model



Model Analyzed



Study Properties

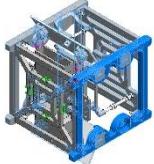
Study name	Static Analysis
Analysis type	Static
Mesh type	Solid Mesh
Thermal Effect:	On
Thermal option	Include temperature loads
Zero strain temperature	298 Kelvin
Include fluid pressure effects from SOLIDWORKS Flow Simulation	Off
Solver type	Automatic
Inplane Effect:	Off
Soft Spring:	Off
Inertial Relief:	Off
Incompatible bonding options	Automatic
Large displacement	Off
Compute free body forces	On
Friction	Off
Use Adaptive Method:	Off
Result folder	SOLIDWORKS document (C:\Users\235474\Downloads)

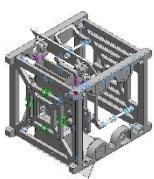
Units

Unit system:	SI (MKS)
Length/Displacement	mm
Temperature	Kelvin
Angular velocity	Rad/sec
Pressure/Stress	N/m ²



Material Properties

Model Reference	Properties	Components
	<p>Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m² Tensile strength: 3,753e+07 N/m² Elastic modulus: 3,00159e+09 N/m² Poisson's ratio: 0,38 Mass density: 1 370 kg/m³</p>	SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-1), SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-2), SolidBody 1(Cut-Extrude1)(HingeLockPart-6), SolidBody 1(Cut-Extrude1)(HingeLockPart-7), SolidBody 1(Cut-Extrude1)(HingeLockPart-8), SolidBody 1(Cut-Extrude1)(HingeLockPart-9), SolidBody 1(Fillet1)(Lock1-11), SolidBody 1(Fillet1)(Lock1-5), SolidBody 1(Fillet1)(Lock1-6), SolidBody 1(Fillet1)(Lock1-7), SolidBody 1(Fillet1)(Lock1-8), SolidBody 1(Fillet1)(Lock1-9), SolidBody 1(Split Line1)(Pin_Sealing-1), SolidBody 1(Split Line1)(Pin_Sealing-2), SolidBody 1(Split Line1)(Pin_Sealing-3), SolidBody 1(Split Line1)(Pin_Sealing-4), SolidBody 1(Split Line1)(Pin_Sealing-5), SolidBody 1(Split Line1)(Pin_Sealing-6), SolidBody 1(Split Line1)(Pin_Sealing-7), SolidBody 1(Split Line1)(Pin_Sealing-8), SolidBody 1(Split Line3)(SealingAntenna-1), SolidBody 1(Split Line3)(SealingAntenna-2), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-1),

		<p>SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-2), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-3), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-4), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Plate_Lock-1), SolidBody 1(Fillet2)(Sub_MainAsm-1/LowerAsm_CRAFT-2/XY_wall_Negative_Z-1), SolidBody 1(Split Line1)(Sub_MainAsm-1/UpperAsm_CRAFT-1/BatterySpringPlate-1), SolidBody 1(Fillet5)(ZY_Wall-1), SolidBody 1(Fillet5)(ZY_Wall-2)</p>
Curve Data:N/A		
	<p>Name: AISI 316 Annealed Stainless Steel Bar (SS) Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,37895e+08 N/m² Tensile strength: 5,5e+08 N/m² Elastic modulus: 1,93e+11 N/m² Poisson's ratio: 0,3 Mass density: 8 000 kg/m³ Thermal expansion coefficient: 1,6e-05 /Kelvin</p>	<p>SolidBody 1(Cut-Extrude1)(M3 locking Nut-1), SolidBody 1(Cut-Extrude1)(M3 locking Nut-2), SolidBody 1(Cut-Extrude1)(M3 locking Nut-3), SolidBody 1(Cut-Extrude1)(M3 locking Nut-4), SolidBody 1(Cut-Extrude1)(M3 locking Nut-5), SolidBody 1(Cut-Extrude1)(M3 locking Nut-6), SolidBody 1(Cut-Extrude1)(M3 locking Nut-7), SolidBody 1(Cut-Extrude1)(M3 locking Nut-8), SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-1),</p>



SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-10),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-11),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-12),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-13),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-14),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-15),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-16),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-17),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-18),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-19),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-20),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-21),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-22),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-23),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-24),





C.R.A.F.T

STB
12.05.2024

SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-25),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-26),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-27),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-28),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-29),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-3),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-30),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-31),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-32),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-4),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-5),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-6),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-7),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-8),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-9)

Curve Data:N/A

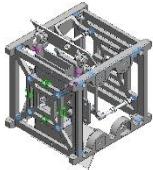


SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm

7

	<p>Name: AISI 316 Stainless Steel Sheet (SS)</p> <p>Model type: Linear Elastic Isotropic</p> <p>Default failure criterion: Unknown</p> <p>Yield strength: 1,72369e+08 N/m²</p> <p>Tensile strength: 5,8e+08 N/m²</p> <p>Elastic modulus: 1,93e+11 N/m²</p> <p>Poisson's ratio: 0,27</p> <p>Mass density: 8 000 kg/m³</p> <p>Thermal expansion coefficient: 1,6e-05 /Kelvin</p>	<p>SolidBody 1(M3_Thread)(M3x10_PZ1 head-1), SolidBody 1(M3_Thread)(M3x10_PZ1 head-2), SolidBody 1(M3_Thread)(M3x10_PZ1 head-3), SolidBody 1(M3_Thread)(M3x10_PZ1 head-4), SolidBody 1(M3_Thread)(M3x10_PZ1 head-5), SolidBody 1(M3_Thread)(M3x10_PZ1 head-6), SolidBody 1(M3_Thread)(M3x10_PZ1 head-7), SolidBody 1(M3_Thread)(M3x10_PZ1 head-8), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-1), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-2), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-3), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-4), SolidBody 1(M3_Thread)(Sub_MainAsm-1/UpperAsm_CRAFT-1/M3x50_PZ1 head-1), SolidBody 1(M3_Thread)(Sub_MainAsm-1/UpperAsm_CRAFT-1/M3x50_PZ1 head-2), SolidBody 1(M3_Thread)(Sub_MainAsm-</p>
---	---	--

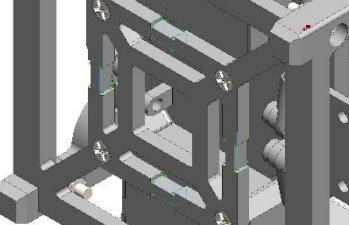
		1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-3), SolidBody 1(M3_Thread)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-4)
Curve Data:N/A		
	Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m ² Tensile strength: 3,753e+07 N/m ² Elastic modulus: 3,00159e+09 N/m ² Poisson's ratio: 0,38 Mass density: 1 370 kg/m ³ Thermal expansion coefficient: 3,8 /Kelvin	SolidBody 1(Outer_Sylinder)(Spring_asm-1/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-1/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-1/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-1/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-2/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-2/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-2/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-2/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-4/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-4/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-4/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-4/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-6/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-6/ShockPart-1),

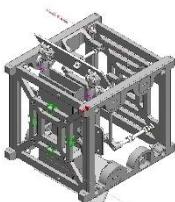
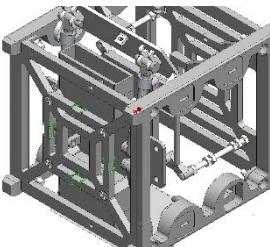


		SolidBody 1(Boss-Extrude2)(Spring_asm-6/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-6/SpringHold_part2-1), SolidBody 1(SplitLine4)(Sub_MainAsm-1/UpperAsm_CRAFT-1/Antenne_ReleaseBoard-1), SolidBody 1(SplitLine1)(Sub_MainAsm-1/UpperAsm_CRAFT-1/XY_wall_Positive_Z-1)
Curve Data:N/A		
	<p>Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m² Tensile strength: 3,753e+07 N/m² Elastic modulus: 3,0059e+08 N/m² Poisson's ratio: 0,38 Mass density: 1 240 kg/m³</p>	SolidBody 1(Cut-Extrude2)(Sub_MainAsm-1/UpperAsm_CRAFT-1/AntennaBoard_AntennaLocRoof-1), SolidBody 1(Cut-Extrude2)(Sub_MainAsm-1/UpperAsm_CRAFT-1/AntennaBoard_AntennaLocRoof-2)
Curve Data:N/A		



Loads and Fixtures

Fixture name	Fixture Image	Fixture Details		
Roller/Slider-1		Entities: 12 face(s) Type: Roller/Slider		
Resultant Forces				
Components	X	Y	Z	Resultant
Reaction force(N)	-0,000330508	0,000878483	15,9157	15,9157
Reaction Moment(N.m)	0	0	0	0

Load name	Load Image	Load Details
Gravity-1		Reference: Front Plane Values: 0 0 -9,81 Units: m/s ²
Force-1		Entities: 4 face(s) Type: Apply normal force Value: 3,79 lbf



C.R.A.F.T

STB
12.05.2024

Connector Definitions

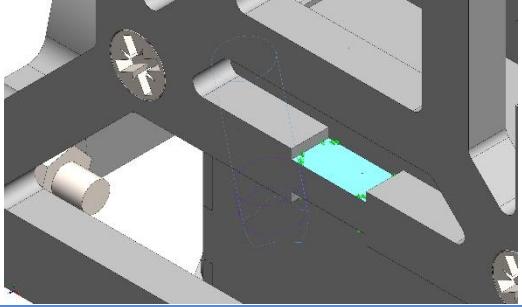
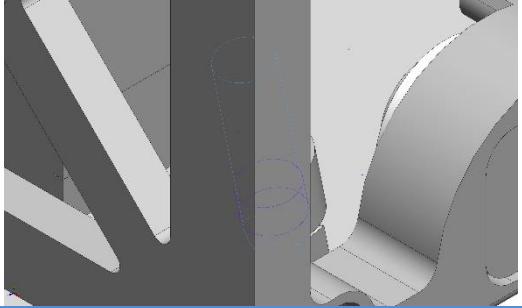
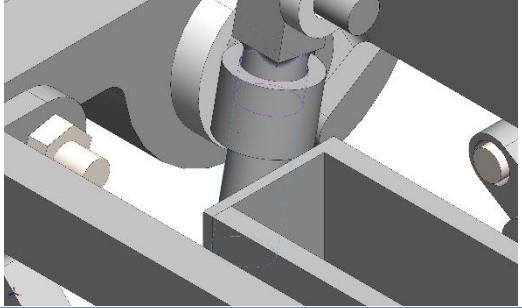
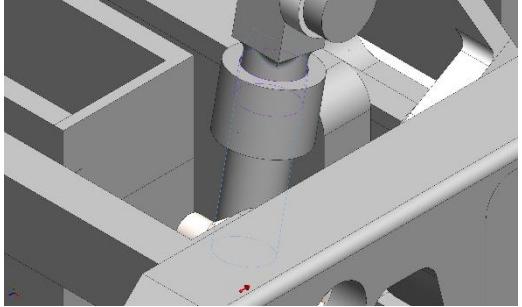
No Data



SOLIDWORKS Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 12

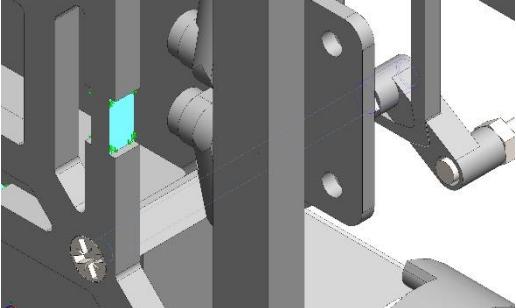
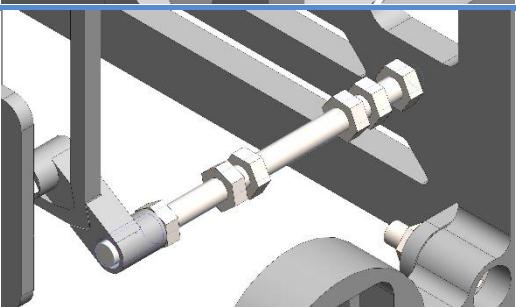
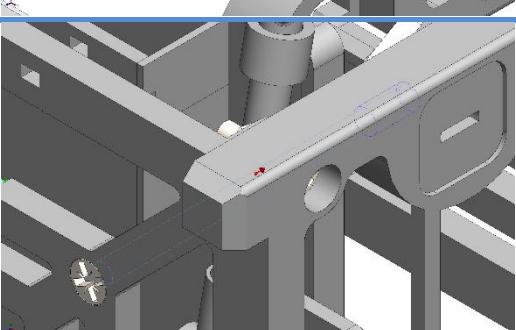
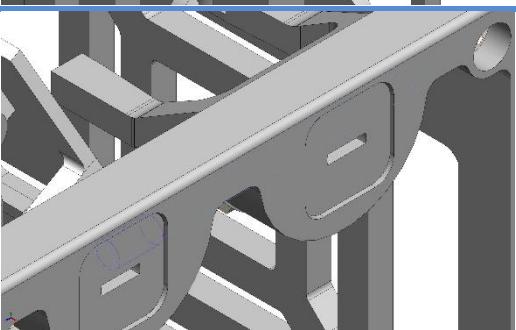
Interaction Information

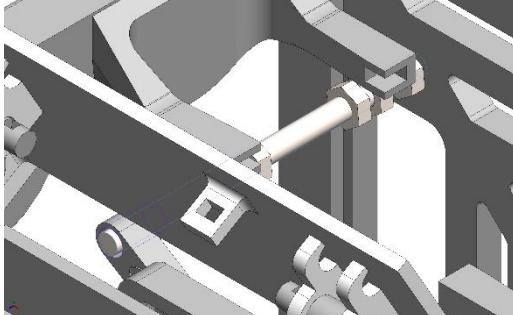
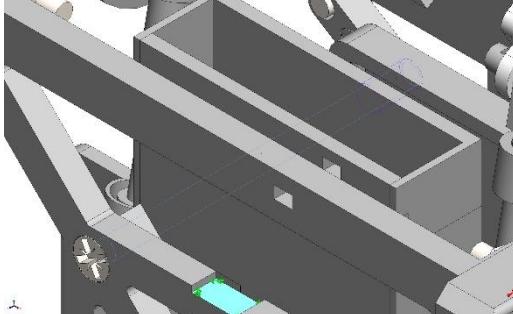
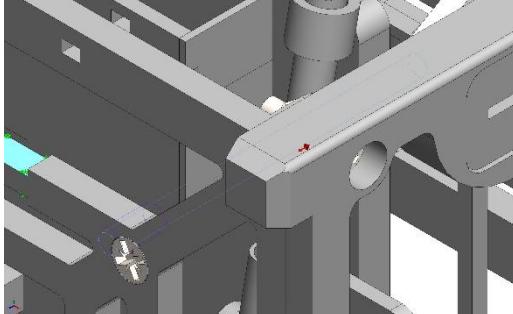
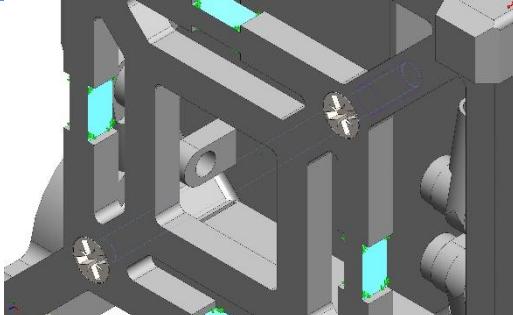
Interaction	Interaction Image	Interaction Properties
Local Interaction-3890		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-3891		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-3892		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-3893		Type: Bonded interaction pair Entities: 2 face(s)



Local Interaction-4153		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4154		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4155		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4156		Type: Bonded interaction pair Entities: 2 face(s)



Local Interaction-4416		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4417		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4418		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4419		Type: Bonded interaction pair Entities: 2 face(s)

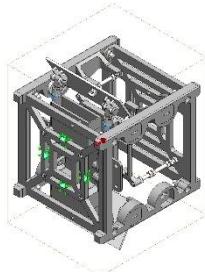
Local Interaction-4420		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4421		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4422		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4423		Type: Bonded interaction pair Entities: 2 face(s)



C.R.A.F.T

STB
12.05.2024

Global Interaction



Type: Bonded
Components: 1 component(s)
Options: Independent mesh



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 17



Mesh information

Mesh type	Solid Mesh
Mesher Used:	Blended curvature-based mesh
Jacobian points for High quality mesh	16 Points
Maximum element size	30,6206 mm
Minimum element size	1,53103 mm
Mesh Quality	High
Remesh failed parts independently	Off

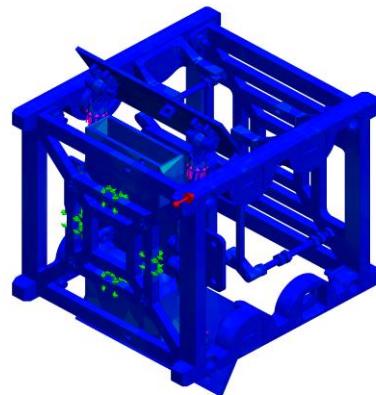
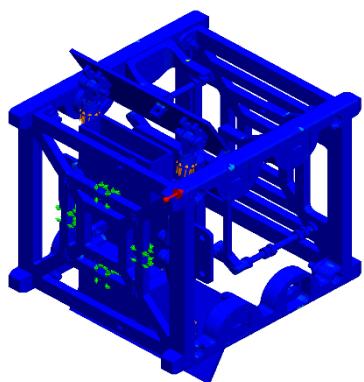
Mesh information - Details

Total Nodes	160954
Total Elements	80414
Maximum Aspect Ratio	88,119
% of elements with Aspect Ratio < 3	85,9
Percentage of elements with Aspect Ratio > 10	0,78
Percentage of distorted elements	0
Time to complete mesh(hh:mm:ss):	00:00:38
Computer name:	S-K-1230-06

Mesh Quality Plots

Because of the amount of data required in the analysis, didn't make a mesh quality change on the areas with a Aspect & Jacobian Ratio higher than 10. We will therefore accept a non-accurate value in the analysis but define the results using this mesh as a "real" result related to this mesh. If we had a computer with more memory, a mesh change in the areas with high ratios would be done to gain a more predictable value.





Sensor Details

No Data

Resultant Forces

Reaction forces

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N	-0,000330508	0,000878483	15,9157	15,9157

Reaction Moments

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N.m	0	0	0	0

Free body forces

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N	0,019454	-0,00681183	1,4905	1,49065

Free body moments

Selection set	Units	Sum X	Sum Y	Sum Z	Resultant
Entire Model	N.m	0	0	0	1e-33



C.R.A.F.T

Beams

No Data

STB
12.05.2024



SOLIDWORKS

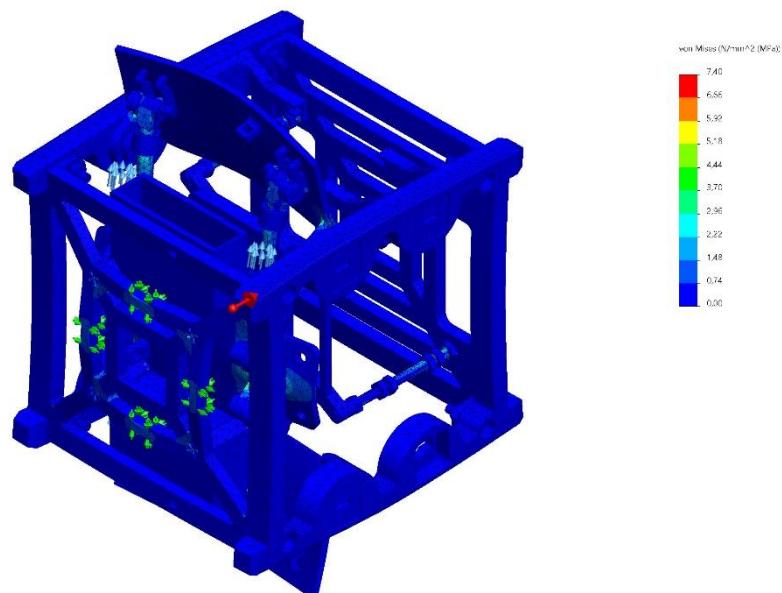
Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 20

Study Results

Name	Type	Min	Max
Stress1	VON: von Mises Stress	0,00N/mm ² (MPa) Node: 20955	7,40N/mm ² (MPa) Node: 44851

Mode name: Main_asm
Study name: Static Analysis (Default)
Plot type: Static von Mises Stress1
Deformation scale: 1197.24



SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Static Analysis-Stress-Stress1



C.R.A.F.T

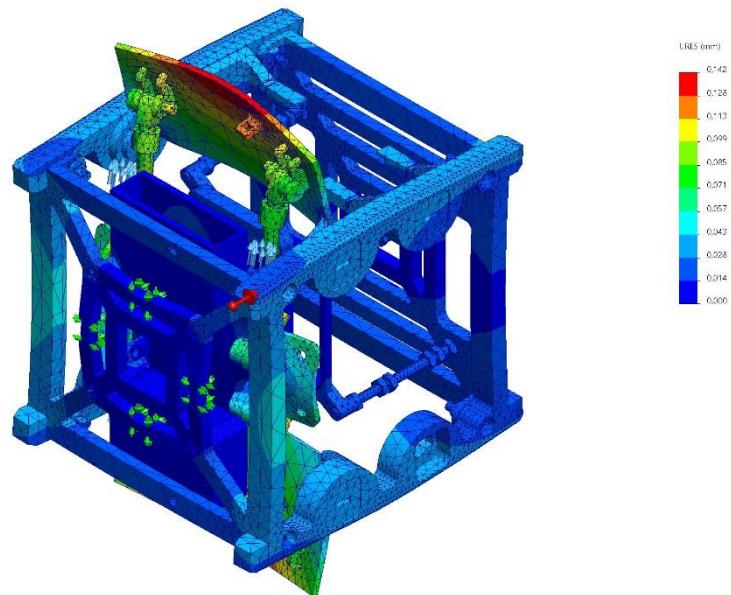
STB
12.05.2024

Name	Type	Min	Max
Displacement1	URES: Resultant Displacement	0,000mm Node: 116231	0,142mm Node: 26367

Model name: Main_asm
Study name: Static Analysis (Dehnf)

Plot type: Static displacement Displacement1

Deformation scale: 116:124



SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Static Analysis-Displacement-Displacement1



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm

22

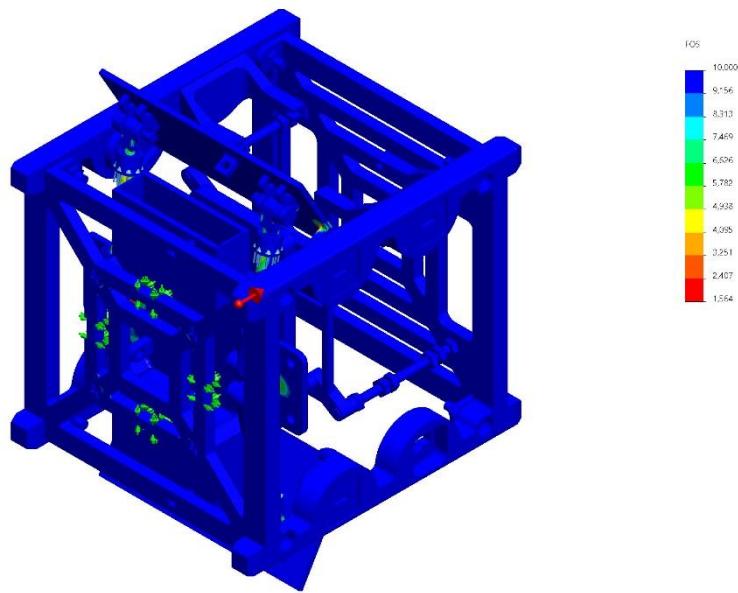


C.R.A.F.T

STB
12.05.2024

Name	Type	Min	Max
Factor of Safety1	Automatic	1,564 Node: 44851	10,000 Node: 1

Model name: Main_asm
Study name: Static Analysis (Deflult)
Plot type: Factor of Safety/Factor of Safety1
Orientation: Automatic
Factor of safety distribution: Min FOS = 1.6



SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Static Analysis-Factor of Safety-Factor of Safety1



SOLIDWORKS Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 23

Conclusion

The vonMises sheet show that the highest stress value is less than the yield strength value of PETG, which compared with the bolts (the bolts represent the classification A4) made of 316 Stainless steel is the weaker material in stress value. *This is don't conclude if the PETG is overall weaker than 316 stainless steel.*

The displacement shows (with high deformation scale) the force from the spring on the sealings and the curvature it makes. Bolts also gives stresses around the threads and spreads, but in this case the only chosen force used in the analysis beside the gravitation is from the springs. By using this, it's much easier to evaluate the sealings, and by this case they show the highest displacement values. The sealings will bend in the direction away from the antennas, which gives the secure state of not interacting/bending the antennas when the sealing is already open, and it's not pushed back to the state where it's closed.

Since the factor of safety is greater than one, we will conclude that the bending will return to the state before it's bend when the chosen loads is removed. Also, we have not been given a minimum factor of safety for a static analysis, we conclude that by the chosen case with only spring force and gravitational force, the model is approved to be considered with a dynamic loading with the springs to decide how many possible cycles is approved before the spring or other components is destroyed.

The dynamic loading is considered but not used. The dynamic loading plan is: sealings open and closed during a time interval with a change in spring force due to the difference in deflection value to calculate the spring force. The fixtures and position of gravitational load is equal this analysis.

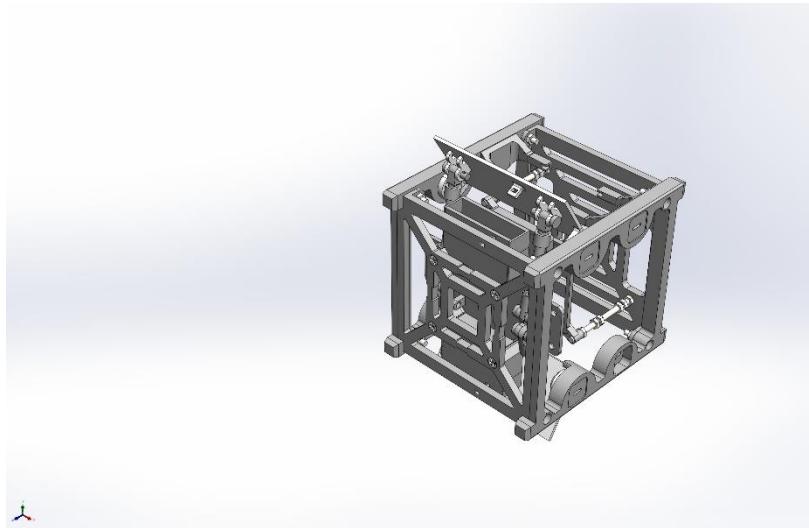
V.2 Thermal Analysis

Under this section is a description and a result of a thermal analysis with the *main_asm*.



CanSat Research and Advanced Fabrication Team

C.R.A.F.T



Description

The analysis illustrates a steady state simulation with a overall temperature of the lowest value given. This value is chosen to illustrate a critical state due to temperature. Antenna release board is the only area which don't have the temperature. The antenna release board, and the battery holder has the heat flux given from the battery. The simulation has an Aspect ratio with a small amount percent higher than 10, but with mostly under 3 percent. Because the simulation required a large amount of data, we have chosen to simplify it, and accept the small number of percent higher than 10 but the location of the values higher than 10 percent has been checked. Components such as electronical circuit boards and cables, bolts, and nuts with threads, have been removed. Some bolts have been included but they don't have threads. The model has only a design representing balloon mounting and not parachuting. The model can be re-designed, but it's not our object at this moment to do so.

Simulation assembly: Main_asm

Date: Friday 10. May 2024

Designer: STB

Study name: Thermal analysis

Analysis type: Thermal (Steady state)

Table of Contents

Description	1
Assumptions & model information	2
Study Properties.....	17
Units	17
Material Properties	18
Thermal Loads	25
Interaction Information	26
Mesh information.....	27
Sensor Details.....	27
Study Results	28
Conclusion.....	29



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

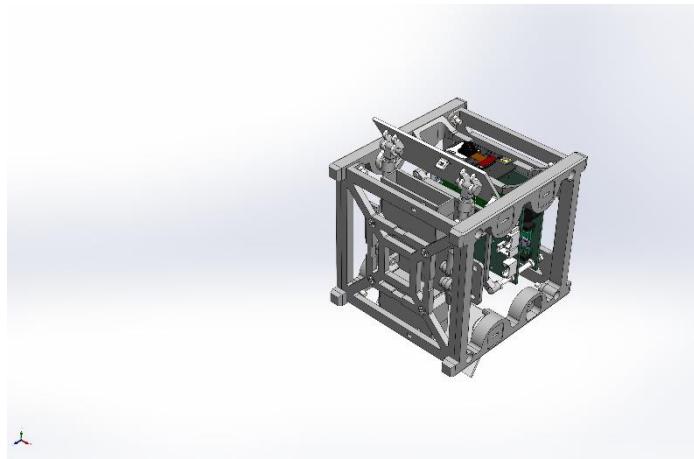
Simulation of Main_asm 1

Assumptions & model information

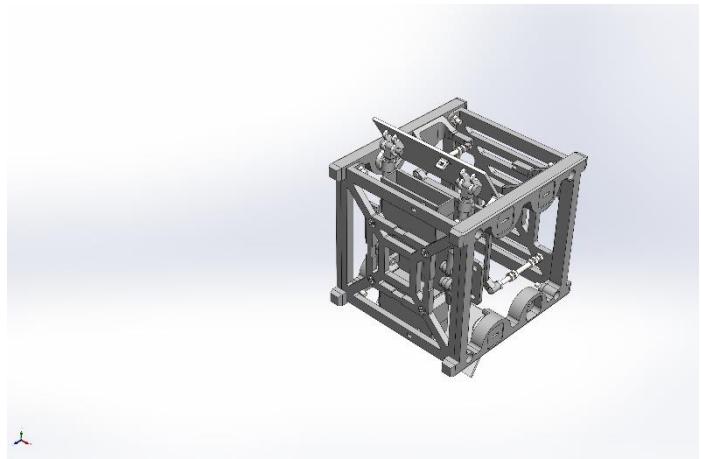
We assume heat gained from the battery on the polymer components is important to consider, and therefore is the only chosen thermal load beside an overall temperature of -85°C on the model. Since time lap simulation requires a large amount of data, we assume steady analysis representing the lowest temperature is enough for the main_asm. Since the model is a solid tight fitted with non-free components, non-local interactions are decided and only the bonded global interaction is used. The model has components with threads and small sizes; the threads are removed, and some mechanical components (such as bolts, nuts, and springs) are not included in the analysis.

Note: since the FEA required a large amount of data, it's difficult to gain other pictures and change the temperature scale without constantly crash the computer. Therefore, temperature grade in the analysis is in kelvin but we are mostly using Celsius. The convert equation from kelvin Celsius is:

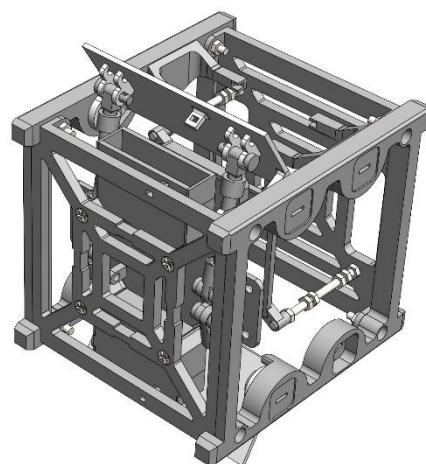
$$T_{\text{Celsius}} = T_{\text{Kelvin}} - 273.15$$



Original Model



Model Analyzed



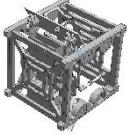
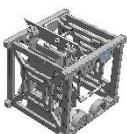
Model name: Main_asm
Current Configuration: Default

Solid Bodies

Document Name and Reference	Treated As	Volumetric Properties	Document Path/Date Modified
Split Line3	Solid Body	Mass: 0,00290598 kg Volume: 2,12136e-06 m^3 Density: 1 369,87 kg/m^3 Weight: 0,0284786 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Camera_Holder.SLDPR T May 10 08:37:58 2024
Split Line3	Solid Body	Mass: 0,00290598 kg Volume: 2,12136e-06 m^3 Density: 1 369,87 kg/m^3 Weight: 0,0284786 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Camera_Holder.SLDPR T May 10 08:37:58 2024
Cut-Extrude1	Solid Body	Mass: 0,000567144 kg Volume: 4,13973e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDPR T May 10 08:37:59 2024

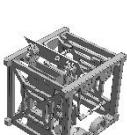




Cut-Extrude1		Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDprt May 10 08:37:59 2024
Cut-Extrude1		Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDprt May 10 08:37:59 2024
Cut-Extrude1		Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDprt May 10 08:37:59 2024
Fillet1		Solid Body	Mass:0,00107758 kg Volume:7,86552e-07 m^3 Density:1 370 kg/m^3 Weight:0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 08:37:59 2024
Fillet1		Solid Body	Mass:0,00107758 kg Volume:7,86552e-07 m^3 Density:1 370 kg/m^3 Weight:0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 08:37:59 2024
Fillet1		Solid Body	Mass:0,00107758 kg Volume:7,86552e-07 m^3 Density:1 370 kg/m^3 Weight:0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 08:37:59 2024





Fillet1 	Solid Body	Mass:0,00107758 kg Volume:7,86552e-07 m^3 Density:1 370 kg/m^3 Weight:0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 08:37:59 2024
Fillet1 	Solid Body	Mass:0,00107758 kg Volume:7,86552e-07 m^3 Density:1 370 kg/m^3 Weight:0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 08:37:59 2024
Fillet1 	Solid Body	Mass:0,00107758 kg Volume:7,86552e-07 m^3 Density:1 370 kg/m^3 Weight:0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 08:37:59 2024
Cut-Extrude1 	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 08:37:59 2024
Cut-Extrude1 	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 08:37:59 2024
Cut-Extrude1 	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05





C.R.A.F.T

STB
10.05.2024

		Density: 8 000 kg/m ³ Weight: 0,00228647 N	05\ M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m ³ Density: 8 000 kg/m ³ Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m ³ Density: 8 000 kg/m ³ Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m ³ Density: 8 000 kg/m ³ Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m ³ Density: 8 000 kg/m ³ Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
M3_Thread 	Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m ³ Density: 8 000 kg/m ³ Weight: 0,00954021 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1 head.SLDPRT May 10 08:37:59 2024
M3_Thread	Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m ³ Density: 8 000 kg/m ³	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-



SOLIDWORKS Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm

6



		Weight: 0,00954021 N	05\ M3x10_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	 Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downlo ads\FEA_BackUp 08-34-10- 05\M3x10_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	 Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downlo ads\FEA_BackUp 08-34-10- 05\M3x10_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	 Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downlo ads\FEA_BackUp 08-34-10- 05\M3x10_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	 Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downlo ads\FEA_BackUp 08-34-10- 05\M3x10_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	 Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downlo ads\FEA_BackUp 08-34-10- 05\M3x10_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	 Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downlo ads\FEA_BackUp 08-34-10- 05\M3x10_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	 Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downlo ads\FEA_BackUp 08-34-10- 05\M3x10_PZ1 head.SLDPR May 10 08:37:59 2024





C.R.A.F.T

STB
10.05.2024

		Weight:0,00954021 N	05\M3x10_PZ1 head.SLDPR May 10 08:37:59 2024
Split Line1 	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 08:37:59 2024
Split Line1 	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 08:37:59 2024
Split Line1 	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 08:37:59 2024
Split Line1 	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 08:37:59 2024
Split Line1 	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 08:37:59 2024
Split Line1 	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 08:37:59 2024
Split Line1 	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 08:37:59 2024



SOLIDWORKS Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm

8



		Weight:0,00214227 N	May 10 08:37:59 2024
Split Line1 	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 08:37:59 2024
Split Line1 	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 08:37:59 2024
Split Line3 	Solid Body	Mass:0,00973089 kg Volume:7,10284e-06 m^3 Density:1 370 kg/m^3 Weight:0,0953628 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SealingAntenna.SLDPR T May 10 08:38:00 2024
Split Line3 	Solid Body	Mass:0,00973089 kg Volume:7,10284e-06 m^3 Density:1 370 kg/m^3 Weight:0,0953628 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SealingAntenna.SLDPR T May 10 08:38:00 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m^3 Density:1 370 kg/m^3 Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLDPR T May 10 08:37:59 2024
Fillet1	Solid Body	Mass:0,000428203 kg Volume:3,12546e-07 m^3 Density:1 370,05 kg/m^3 Weight:0,00419639 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLDPR T May 10 08:38:00 2024





Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m^3 Density:1 370 kg/m^3 Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 08:38:00 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m^3 Density:1 370 kg/m^3 Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 08:38:00 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m^3 Density:1 370 kg/m^3 Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLD PRT May 10 08:37:59 2024
Fillet1	Solid Body	Mass:0,000428204 kg Volume:3,12546e-07 m^3 Density:1 370,05 kg/m^3 Weight:0,0041964 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLD PRT May 10 08:38:00 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m^3 Density:1 370 kg/m^3 Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 08:38:00 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m^3 Density:1 370 kg/m^3 Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 08:38:00 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m^3 Density:1 370 kg/m^3 Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLD PRT May 10 08:37:59 2024
Fillet1	Solid Body	Mass:0,000428203 kg Volume:3,12546e-07 m^3 Density:1 370,05 kg/m^3 Weight:0,00419639 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLD PRT May 10 08:38:00 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m^3 Density:1 370 kg/m^3 Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 08:38:00 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m^3 Density:1 370 kg/m^3	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-





		Weight:0,00442236 N	05\SpringHold_part2.SLD PRT May 10 08:38:00 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m^3 Density:1 370 kg/m^3 Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLD PRT May 10 08:37:59 2024
Fillet1	Solid Body	Mass:0,000428204 kg Volume:3,12546e-07 m^3 Density:1 370,05 kg/m^3 Weight:0,0041964 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLD PRT May 10 08:38:00 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m^3 Density:1 370 kg/m^3 Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 08:38:00 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m^3 Density:1 370 kg/m^3 Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 08:38:00 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 08:37:59 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 08:37:59 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 08:37:59 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 08:37:59 2024
Fillet1	Solid Body	Mass:0,00132768 kg Volume:9,69113e-07 m^3 Density:1 370 kg/m^3 Weight:0,0130113 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Plate_Lock.SLD PRT May 10 08:37:59 2024





M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 08:37:59 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 08:37:59 2024
Fillet2	Solid Body	Mass:0,0297987 kg Volume:2,17509e-05 m^3 Density:1 370 kg/m^3 Weight:0,292027 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\XY_wall_Negative_Z.S LDPRT May 10 08:38:01 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024





		Weight:0,00228647 N	05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPRT May 10 08:37:59 2024





Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 08:37:59 2024





		Weight:0,00228647 N	05\ M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 08:37:59 2024
Cut-Extrude2	Solid Body	Mass:0,00269269 kg Volume:2,17153e-06 m^3 Density:1 240 kg/m^3 Weight:0,0263884 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ AntennaBoard_AntennaLockRoof.SLDPR May 10 08:37:58 2024





Cut-Extrude2	Solid Body	Mass:0,00269269 kg Volume:2,17153e-06 m^3 Density:1 240 kg/m^3 Weight:0,0263884 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\AntennaBoard_AntennaLockRoof.SLDprt May 10 08:37:58 2024
Split Line4	Solid Body	Mass:0,0206986 kg Volume:1,51084e-05 m^3 Density:1 370 kg/m^3 Weight:0,202846 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Antenne_ReleaseBoard.SLDprt May 10 08:37:58 2024
Split Line1	Solid Body	Mass:0,00625083 kg Volume:4,56265e-06 m^3 Density:1 370 kg/m^3 Weight:0,0612582 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\BatterySpringPlate.SLDprt May 10 08:37:58 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1_head.SLDprt May 10 08:37:59 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1_head.SLDprt May 10 08:37:59 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1_head.SLDprt May 10 08:37:59 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1_head.SLDprt May 10 08:37:59 2024
Split Line1	Solid Body	Mass:0,0222911 kg Volume:1,62708e-05 m^3 Density:1 370 kg/m^3 Weight:0,218452 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\XY_wall_Positive_Z.SLDprt May 10 08:38:10 2024
Fillet5	Solid Body	Mass:0,0318527 kg Volume:2,32501e-05 m^3 Density:1 370 kg/m^3 Weight:0,312156 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ZY_Wall.SLDprt May 10 08:38:10 2024





Fillet5		Solid Body	Mass: 0,0318527 kg Volume: 2,32501e-05 m^3 Density: 1 370 kg/m^3 Weight: 0,312156 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ZY_Wall.SLDprt May 10 08:38:10 2024
---------	---	------------	--	---

Study Properties

Study name	Thermal analysis
Analysis type	Thermal(Steady state)
Mesh type	Solid Mesh
Solver type	FFEPlus
Solution type	Steady state
Contact resistance defined?	No
Result folder	SOLIDWORKS document (C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05)

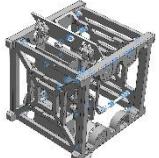
Units

Unit system:	SI (MKS)
Length/Displacement	mm
Temperature	Kelvin
Angular velocity	Rad/sec
Pressure/Stress	N/m^2



Material Properties

Model Reference	Properties	Components
	<p>Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Thermal conductivity: 0,2256 W/(m.K) Specific heat: 1 300 J/(kg.K) Mass density: 1 370 kg/m³</p>	SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-1), SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-2), SolidBody 1(Cut-Extrude1)(HingeLockPart-6), SolidBody 1(Cut-Extrude1)(HingeLockPart-7), SolidBody 1(Cut-Extrude1)(HingeLockPart-8), SolidBody 1(Cut-Extrude1)(HingeLockPart-9), SolidBody 1(Fillet1)(Lock1-11), SolidBody 1(Fillet1)(Lock1-5), SolidBody 1(Fillet1)(Lock1-6), SolidBody 1(Fillet1)(Lock1-7), SolidBody 1(Fillet1)(Lock1-8), SolidBody 1(Fillet1)(Lock1-9), SolidBody 1(Split Line1)(Pin_Sealing-1), SolidBody 1(Split Line1)(Pin_Sealing-2), SolidBody 1(Split Line1)(Pin_Sealing-3), SolidBody 1(Split Line1)(Pin_Sealing-4), SolidBody 1(Split Line1)(Pin_Sealing-5), SolidBody 1(Split Line1)(Pin_Sealing-6), SolidBody 1(Split Line1)(Pin_Sealing-7), SolidBody 1(Split Line1)(Pin_Sealing-8), SolidBody 1(Split Line3)(SealingAntenna-1), SolidBody 1(Split Line3)(SealingAntenna-2), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-1),

		SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-2), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-3), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-4), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Plate_Lock-1), SolidBody 1(Fillet2)(Sub_MainAsm-1/LowerAsm_CRAFT-2/XY_wall_Negative_Z-1), SolidBody 1(Split Line1)(Sub_MainAsm-1/UpperAsm_CRAFT-1/BatterySpringPlate-1), SolidBody 1(Fillet5)(ZY_Wall-1), SolidBody 1(Fillet5)(ZY_Wall-2)
Curve Data:N/A		
	Name: AISI 316 Annealed Stainless Steel Bar (SS) Model type: Linear Elastic Isotropic Default failure criterion: Unknown Thermal conductivity: 16,3 W/(m.K) Specific heat: 500 J/(kg.K) Mass density: 8 000 kg/m ³	SolidBody 1(Cut-Extrude1)(M3 locking Nut-1), SolidBody 1(Cut-Extrude1)(M3 locking Nut-2), SolidBody 1(Cut-Extrude1)(M3 locking Nut-3), SolidBody 1(Cut-Extrude1)(M3 locking Nut-4), SolidBody 1(Cut-Extrude1)(M3 locking Nut-5), SolidBody 1(Cut-Extrude1)(M3 locking Nut-6), SolidBody 1(Cut-Extrude1)(M3 locking Nut-7), SolidBody 1(Cut-Extrude1)(M3 locking Nut-8), SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-1),



C.R.A.F.T

STB
10.05.2024

SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-10),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-11),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-12),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-13),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-14),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-15),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-16),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-17),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-18),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-19),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-20),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-21),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-22),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-23),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-24),



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm

20



C.R.A.F.T

STB
10.05.2024

SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-25),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-26),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-27),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-28),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-29),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-3),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-30),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-31),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-32),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-4),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-5),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-6),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-7),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-8),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-9)

Curve Data:N/A

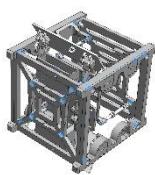


SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm

21



	Name: AISI 316 Stainless Steel Sheet (SS) Model type: Linear Elastic Isotropic Default failure criterion: Unknown Thermal conductivity: 16,3 W/(m.K) Specific heat: 500 J/(kg.K) Mass density: 8 000 kg/m^3	SolidBody 1(M3_Thread)(M3x10_PZ1 head-1), SolidBody 1(M3_Thread)(M3x10_PZ1 head-2), SolidBody 1(M3_Thread)(M3x10_PZ1 head-3), SolidBody 1(M3_Thread)(M3x10_PZ1 head-4), SolidBody 1(M3_Thread)(M3x10_PZ1 head-5), SolidBody 1(M3_Thread)(M3x10_PZ1 head-6), SolidBody 1(M3_Thread)(M3x10_PZ1 head-7), SolidBody 1(M3_Thread)(M3x10_PZ1 head-8), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-1), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-2), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-3), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-4), SolidBody 1(M3_Thread)(Sub_MainAsm-1/UpperAsm_CRAFT-1/M3x50_PZ1 head-1), SolidBody 1(M3_Thread)(Sub_MainAsm-1/UpperAsm_CRAFT-1/M3x50_PZ1 head-2), SolidBody 1(M3_Thread)(Sub_MainAsm-
--	--	--

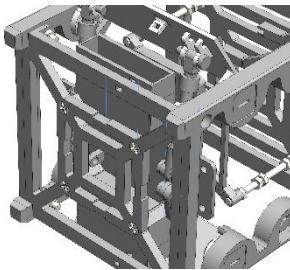
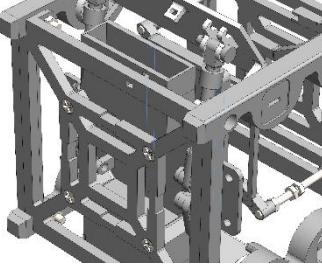
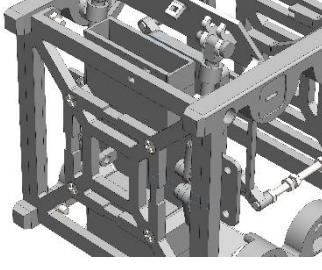
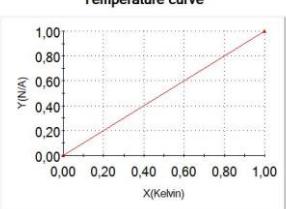
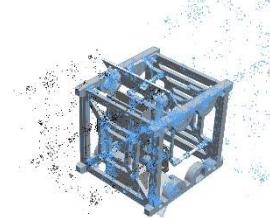


		1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-3), SolidBody 1(M3_Thread)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-4)
Curve Data:N/A		
	Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Thermal conductivity: 0,2256 W/(m.K) Specific heat: 1 300 J/(kg.K) Mass density: 1 370 kg/m^3	SolidBody 1(Outer_Sylinder)(Spring_asm-1/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-1/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-1/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-1/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-2/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-2/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-2/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-2/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-4/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-4/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-4/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-4/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-6/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-6/ShockPart-1),



		SolidBody 1(Boss-Extrude2)(Spring_asm-6/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-6/SpringHold_part2-1), SolidBody 1(SplitLine4)(Sub_MainAsm-1/UpperAsm_CRAFT-1/Antenne_ReleaseBoard-1), SolidBody 1(SplitLine1)(Sub_MainAsm-1/UpperAsm_CRAFT-1/XY_wall_Positive_Z-1)
Curve Data:N/A		
	Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Thermal conductivity: 0,2256 W/(m.K) Specific heat: 1 386 J/(kg.K) Mass density: 1 240 kg/m^3	SolidBody 1(Cut-Extrude2)(Sub_MainAsm-1/UpperAsm_CRAFT-1/AntennaBoard_AntennaLocRoof-1), SolidBody 1(Cut-Extrude2)(Sub_MainAsm-1/UpperAsm_CRAFT-1/AntennaBoard_AntennaLocRoof-2)
Curve Data:N/A		

Thermal Loads

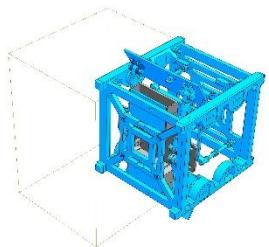
Load name	Load Image	Load Details	Function Curve
Heat Flux-1		Entities: 2 face(s) Heat Flux: 280 W/m ² Value:	
Heat Flux-2		Entities: 6 face(s) Heat Flux: 497 W/m ² Value:	
Heat Flux-3		Entities: 6 face(s) Heat Flux: 964 W/m ² Value: Temperature variation: on	 Temperature curve 0, 0
Temperature-1		Entities: 16368 face(s) Temperature: 188,15 Kelvin	



C.R.A.F.T

STB
10.05.2024

Interaction Information

Interaction	Interaction Image	Interaction Properties
Global Interaction		Type: Bonded Components: 1 component(s) Options: Independent mesh



SOLIDWORKS Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 26



Mesh information

Mesh type	Solid Mesh
Mesher Used:	Blended curvature-based mesh
Jacobian points for High quality mesh	16 Points
Maximum element size	15,3777 mm
Minimum element size	0,768884 mm
Mesh Quality	High
Remesh failed parts independently	Off

Mesh information - Details

Total Nodes	285841
Total Elements	153704
Maximum Aspect Ratio	88,119
% of elements with Aspect Ratio < 3	92,5
Percentage of elements with Aspect Ratio > 10	0,271
Percentage of distorted elements	0
Time to complete mesh(hh:mm:ss):	00:01:32
Computer name:	S-K-1230-10

Sensor Details

No Data

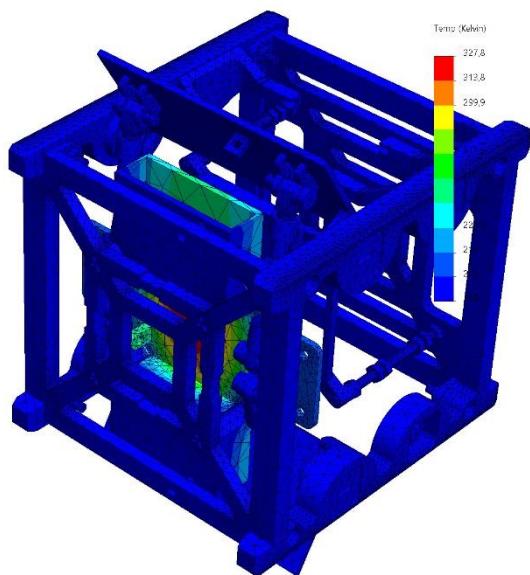




Study Results

Name	Type	Min	Max
Thermal1	TEMP: Temperature	188Kelvin Node: 140575	327,8Kelvin Node: 141786

Model name: Main_asm
Study name: Thermal analysis(Default)
Plot type: Thermal_Thermal1
Time step: 1



SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Thermal analysis-Thermal-Thermal1



Conclusion

The highest temperature on the antenna board surface is 327.8 Kelvin (54,65 Celsius). Since this temperature is lower than the glass transition temperature; the parts made of PETG will still brittle and less ductile. The sheet is difficult to read, but I assume the bolts has a low temperature (less than -60°C) which gives them the circumstance to also be in the glass transition temperature.

We don't have material data about the *glass-temperature* of the AISI 316 Stainless sheet bar, since the data for a metal is different compared with a polymer. We can't conclude the glass temperature regarding the steel and will instead conclude by the polymer material with the possibility for a new conclusion when data about the glass temperature of AISI 316 is provided.

We will therefore conclude with this analysis that the model will have a high strength, but a small ductility compared with PETG if the temperature was higher than the glass temperature. The structure due to glass temperature will be easy to fracture if a shock-load suddenly appeared on the structure. We will also confirm that it is possible to use the PETG material with calculated heat transfer and environmental temperature affection the structure, but with including the risk of change in conclusion if a new and possible more informative decision takes place.

It's important to note that it's not a analyze which describes if the electronical components is capable of being used in this environmental temperature.

Update: When other main_asm analysis was made, local interaction had to be made, but due to the amount of data required for the analysis, a new updated analysis isn't possible at this moment. The results are therefore related to the data gained from the analysis without local interactions in the spring assembly.

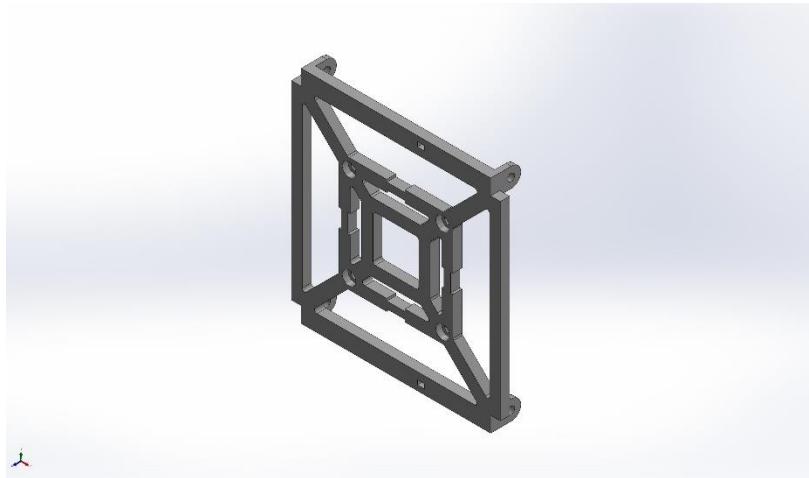
V.3 Natural frequency

Under this section is a descriptions and result of natural frequenzy analysis of the *XY_wall_Positive_Z* and the *main_asm*.



CanSat Research and Advanced Fabrication Team

C.R.A.F.T



Description

The analysis represents a simulation of the natural frequency from the model. The model is made of PETG, but the product is solid. Compared with a 3D printed part, this simulation would not be accurate because of the variables such as total mass but will illustrate how we can confirm if resonance occurs.

The analysis is illustrating that the normal frequency the model is dealing with is between 30 - 60 Hz. Therefore, by using the octave rule to prevent resonance we can't have a value inside the interval {15 - 120} Hz.

Simulation of part: XY_wall_Positive_Z

Date: Thursday 9. May 2024

Designer: STB

Study name: Frequency analysis

Analysis type: Frequency

Table of Contents

Description	1
Assumptions	2
Model Information	3
Study Properties	4
Units	4
Material Properties	5
Loads and Fixtures	6
Connector Definitions	7
Interaction Information	7
Mesh information	8
Sensor Details	9
Study Results	10
Conclusion	15



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of XY_wall_Positive_Z 1



Assumptions & model information

We assume that using a fixed fixture illustrates bolts and nuts connection with other parts. We also assume that a rope used for lifting with a balloon is mounted on the model by using roller/slider fixture on the surfaces. The rope is straight and therefore the model won't move in the direction normal to the surface where the ropes are placed. The model is in a state with only gravitation coefficient as an important load to use for simulating the natural frequency.

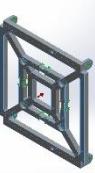




C.R.A.F.T

STB
09.05.2024

Model Information

Solid Bodies			
Document Name and Reference	Treated As	Volumetric Properties	Document Path/Date Modified
Split Line1 	Solid Body	Mass:0,0222911 kg Volume:1,62708e-05 m^3 Density:1 370 kg/m^3 Weight:0,218452 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\XY_wall_Positive_Z.SLDPRT May 9 12:53:17 2024



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of XY_wall_Positive_Z

3

Study Properties

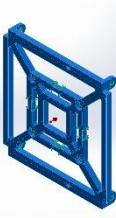
Study name	Frequency analysis
Analysis type	Frequency
Mesh type	Solid Mesh
Number of frequencies	5
Solver type	FFEPlus
Soft Spring:	Off
Incompatible bonding options	Automatic
Thermal option	Include temperature loads
Zero strain temperature	298 Kelvin
Include fluid pressure effects from SOLIDWORKS Flow Simulation	Off
Result folder	SOLIDWORKS document (C:\Users\235474\Downloads\BackUp_08-55-09-05)

Units

Unit system:	SI (MKS)
Length/Displacement	mm
Temperature	Kelvin
Angular velocity	Rad/sec
Pressure/Stress	N/m ²

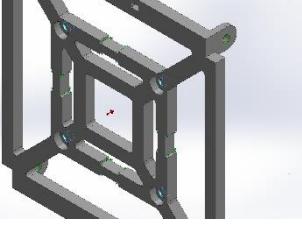
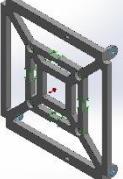
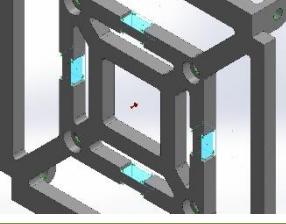


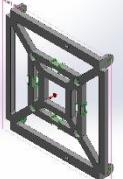
Material Properties

Model Reference	Properties	Components
	<p>Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m² Tensile strength: 3,753e+07 N/m² Mass density: 1 370 kg/m³ Elastic modulus: 3,00159e+09 N/m² Poisson's ratio: 0,38 Thermal expansion coefficient: 3,8 /Kelvin</p>	SolidBody 1(Split Line1)(XY_wall_Positive_Z)
Curve Data:N/A		



Loads and Fixtures

Fixture name	Fixture Image	Fixture Details
Fixed-1		Entities: 4 face(s) Type: Fixed Geometry
Fixed-2		Entities: 4 face(s) Type: Fixed Geometry
Roller/Slider-1		Entities: 12 face(s) Type: Roller/Slider

Load name	Load Image	Load Details
Gravity-1		Reference: Front Plane Values: 0 0 -9,81 Units: m/s ²



C.R.A.F.T

STB
09.05.2024

Connector Definitions

No Data

Interaction Information

No Data



SOLIDWORKS Analyzed with SOLIDWORKS Simulation

Simulation of XY_wall_Positive_Z

7

Mesh information

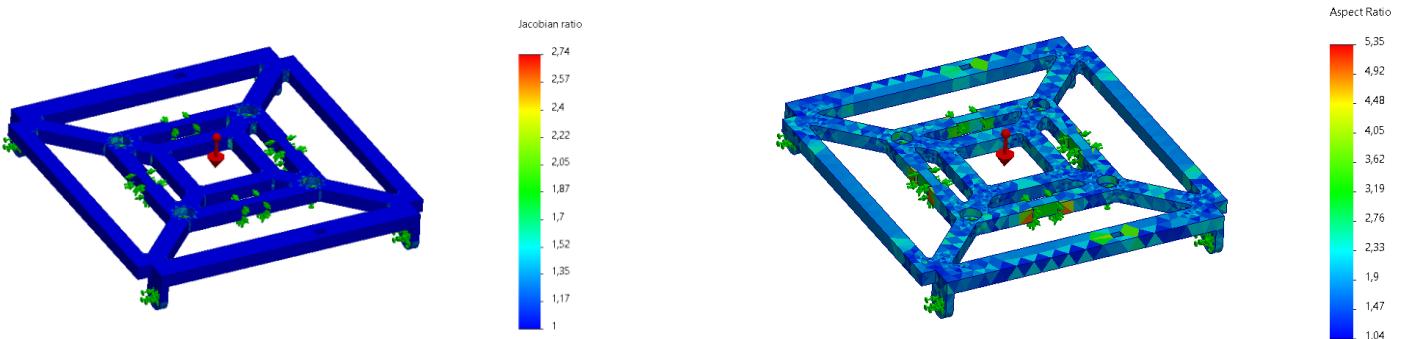
Mesh type	Solid Mesh
Mesher Used:	Blended curvature-based mesh
Jacobian points for High quality mesh	16 Points
Maximum element size	5,06983 mm
Minimum element size	0,765367 mm
Mesh Quality	High

Mesh information - Details

Total Nodes	31362
Total Elements	17573
Maximum Aspect Ratio	5,3462
% of elements with Aspect Ratio < 3	98,7
Percentage of elements with Aspect Ratio > 10	0
Percentage of distorted elements	0
Time to complete mesh(hh:mm:ss):	00:00:06
Computer name:	S-K-1230-10

Mesh Quality Plots

Since Jacobian- and Aspect ratio has percentage values less than 10, we accept this mesh for analysis.





C.R.A.F.T

STB
09.05.2024

Sensor Details

No Data



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

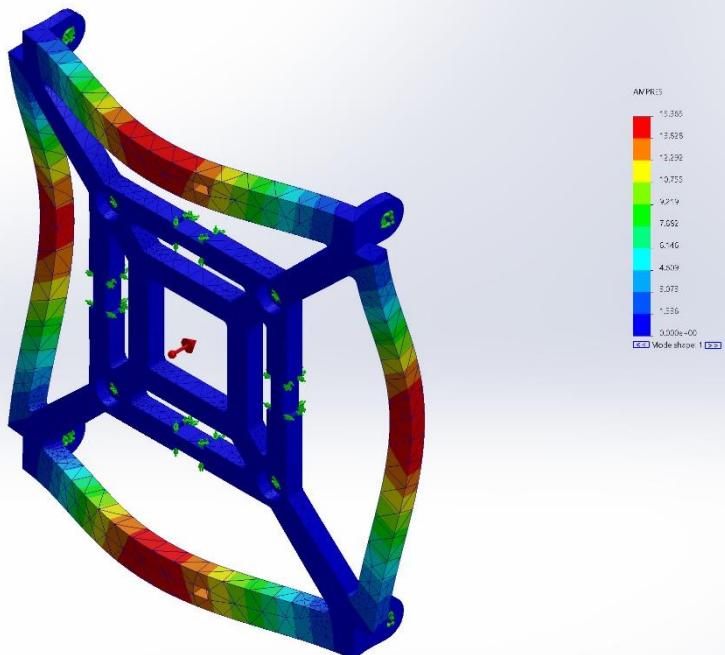
Simulation of XY_wall_Positive_Z

9

Study Results

Name	Type	Min	Max
Amplitude1	AMPRES: Resultant Amplitude Plot for Mode Shape: 1 (Value = 782,588 Hz)	0,000e+00 Node: 57	15,365 Node: 30283

Model name: XY_wall_Positive_Z
Study name: Frequency analysis-XY_wall_Positive_Z
Plot type: Frequency Analysis-XY_wall_Positive_Z
Mode Shape: 1 Value = 782,588 Hz
Deformation scale: 0,000033233



SOLIDWORKS Educational Product. For Instructional Use Only.

XY_wall_Positive_Z-Frequency analysis-Amplitude-Amplitude1

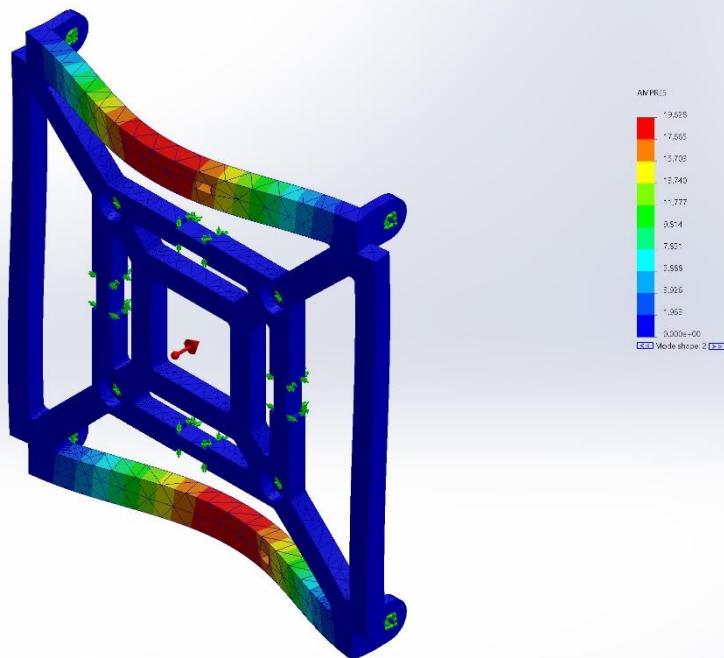


C.R.A.F.T

STB
09.05.2024

Name	Type	Min	Max
Amplitude2	AMPRES: Resultant Amplitude Plot for Mode Shape: 2(Value = 818,578 Hz)	0,000e+00 Node: 57	19,628 Node: 10000

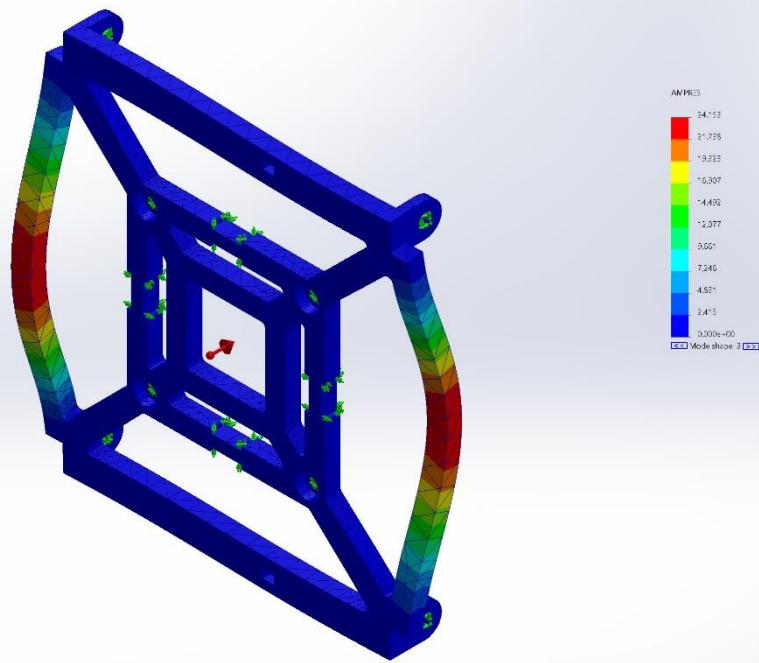
Model name: XY_wall_Positive_Z
Study name: Frequency analysis-Default-
Plot type: Frequency Amplitude<2
Mode Shape: 2 Value = -818.58 Hz
Deformation scale: 0.000/91672



XY wall Positive Z-Frequency analysis-Amplitude-Amplitude2

Name	Type	Min	Max
Amplitude3	AMPRES: Resultant Amplitude Plot for Mode Shape: 3 (Value = 836,787 Hz)	0,000e+00 Node: 57	24,153 Node: 28111

Mode name: XY_wall_Positive_Z
Study name: Frequency analysis (default)
Plot type: Frequency Analysis
Mode Shape: 3 Value = 836,787 Hz
Deformation scale: 0,000/007/8

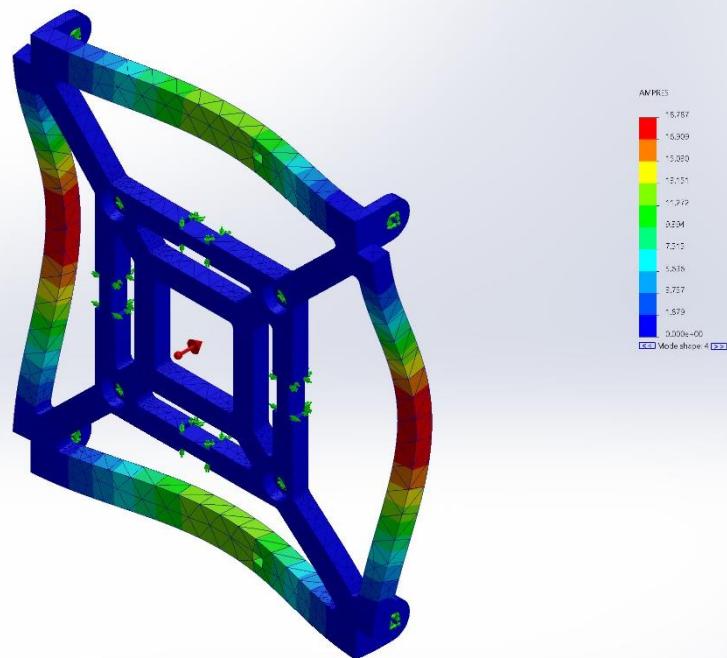


SOLIDWORKS Educational Product. For Instructional Use Only.

XY_wall_Positive_Z-Frequency analysis-Amplitude-Amplitude3

Name	Type	Min	Max
Amplitude4	AMPRES: Resultant Amplitude Plot for Mode Shape: 4 (Value = 879,885 Hz)	0,000e+00 Node: 57	18,787 Node: 29318

Mode name: XY_wall_Positive_Z
Study name: Frequency_wx_yzdc-(Inbuilt)
Plot type: Frequency_A
Mode Shape: 4 Value = 879,885 Hz
Deformation scale: 0,000517323

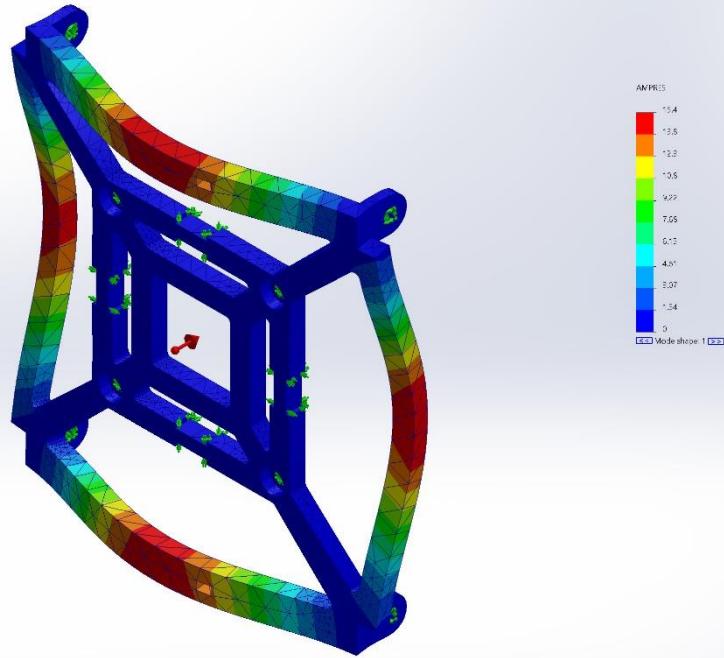


SOLIDWORKS Educational Product. For Instructional Use Only.

XY_wall_Positive_Z-Frequency analysis-Amplitude4

Name	Type	Min	Max
Amplitude5	AMPRES: Resultant Amplitude Plot for Mode Shape: 1 (Value = 782,588 Hz)	0 Node: 57	15,4 Node: 30283

Mode name: XY_wall_Positive_Z
Study name: Frequency_wx_yz_zy_zx_zw_zw_zx
Plot type: Frequency Analysis
Mode Shape: 1 Value = 782,591 Hz
Deformation scale: 0,000033233



SOLIDWORKS Educational Product. For Instructional Use Only.

XY_wall_Positive_Z-Frequency analysis-Amplitude5

Mode List

Frequency Number	Rad/sec	Hertz	Seconds
1	4 917,1	782,59	0,0012778
2	5 143,3	818,58	0,0012216
3	5 257,7	836,79	0,001195
4	5 528,5	879,88	0,0011365
5	6 520,8	1 037,8	0,00096356



Mass Participation (Normalized)

Mode Number	Frequency(Hertz)	X direction	Y direction	Z direction
1	782,59	2,6519e-08	1,0619e-08	0,012362
2	818,58	2,2272e-09	0,00053972	1,336e-06
3	836,79	0,003116	8,5364e-08	1,2793e-05
4	879,88	5,3755e-08	7,2146e-08	0,33092
5	1 037,8	5,2921e-06	1,1087e-08	0,0028582
		Sum X = 0,0031214	Sum Y = 0,0005399	Sum Z = 0,34615

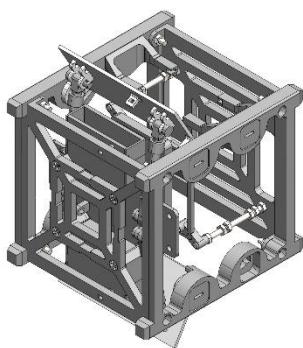
Conclusion

The *mode list* does not have a frequency value inside the interval decided with the octave rule. Therefore, the model is safe for resonance from the interval. Also, it's important to know that this includes with the gravitation force as the only load and the chosen fixtures.





C.R.A.F.T



1

Description

The analysis illustrates a natural frequency simulation. The simulation has an Aspect ratio with a small amount percent higher than 10, but with mostly under 3 percent. Because the simulation required a large amount of data, we have chosen to simplify it, and accept the small amount of percent higher than 10 but the location of the values higher than 10 percent has been checked. Components such as electronical circuit boards and cables, bolts, and nuts with threads, have been removed. Some bolts have been included but they don't have threads. The model has only a design representing balloon mounting and not parachuting. The model can be re-designed, but it's not our object at this moment to do so.

Simulation of assembly: main_asm

Date: Sunday 12. May 2024

Designer: STB

Study name: Frequency Analysis

Analysis type: Frequency

Table of Contents

Description	1
Assumptions & model information	2
Study Properties.....	17
Units	17
Material Properties	18
Loads and Fixtures	25
Connector Definitions	25
Interaction Information	26
Mesh information.....	31
Sensor Details.....	32
Study Results	33
Conclusion.....	38



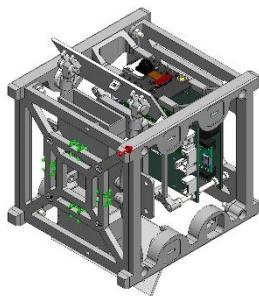
SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

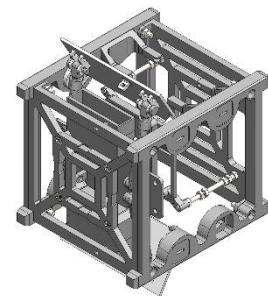
Simulation of Main_asm 1

Assumptions & model information

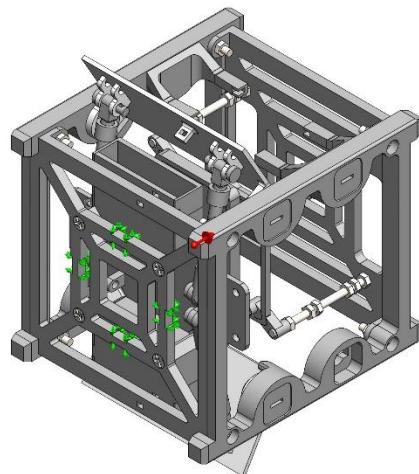
This analysis represents a natural frequency analysis with the gravitation coefficient as the only load. We assume the model is hanging on the balloon with four ropes. The model is analyzed to represent a circumstance with none-electronical components and with an environment where none disturbing objects occurs. Since the model is a solid tight fitted with free components local interactions was made at the areas where components not touching each other due to none spring and thread parts. The bonded global interaction is used on the other components in the model. The model has components with threads and small sizes; the threads are removed, and some mechanical components (such as bolts, nuts, and springs) are not included in the analysis. The model represents the real sizes of the product, with PETG as the material for all the 3D printing parts. The total mass of the model analyzed is higher (we don't have the specific value at this moment) than the real product because we will not use 100% infill in 3D printing. This results with a higher gravitational force, $G = mg$, where m is the mass and g are the gravitational coefficient (a constant value).



Original Model



Model Analyzed

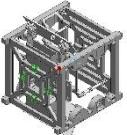
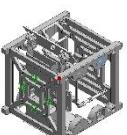


Model name: Main_asm
Current Configuration: Default

Solid Bodies

Document Name and Reference	Treated As	Volumetric Properties	Document Path/Date Modified
Split Line3	Solid Body	Mass:0,00290598 kg Volume:2,12136e-06 m^3 Density:1 369,87 kg/m^3 Weight:0,0284786 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Camera_Holder.SLDPR T May 10 15:28:12 2024
Split Line3	Solid Body	Mass:0,00290598 kg Volume:2,12136e-06 m^3 Density:1 369,87 kg/m^3 Weight:0,0284786 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Camera_Holder.SLDPR T May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDPR T May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDPR T May 10 15:28:12 2024



			
Cut-Extrude1 	Solid Body	Mass: 0,000567144 kg Volume: 4,13973e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDprt May 10 15:28:12 2024
Cut-Extrude1 	Solid Body	Mass: 0,000567144 kg Volume: 4,13973e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDprt May 10 15:28:12 2024
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Fillet1	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024

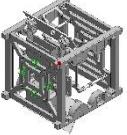




Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024





			
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
M3_Thread	Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1 head.SLDprt May 10 15:28:12 2024
M3_Thread	Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1 head.SLDprt May 10 15:28:12 2024

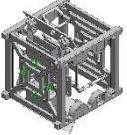




			
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024





			
Split Line1	Solid Body	<p>Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	<p>Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	<p>Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	<p>Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	<p>Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	<p>Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024



			
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line3	Solid Body	Mass: 0,00973089 kg Volume: 7,10284e-06 m^3 Density: 1 370 kg/m^3 Weight: 0,0953628 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SealingAntenna.SLDPR T May 10 15:28:14 2024
Split Line3	Solid Body	Mass: 0,00973089 kg Volume: 7,10284e-06 m^3 Density: 1 370 kg/m^3 Weight: 0,0953628 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SealingAntenna.SLDPR T May 10 15:28:14 2024
Outer_Sylinder	Solid Body	Mass: 0,000373091 kg Volume: 2,72329e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLDPR RT May 10 15:28:12 2024
Fillet1	Solid Body	Mass: 0,000428203 kg Volume: 3,12546e-07 m^3 Density: 1 370,05 kg/m^3 Weight: 0,00419639 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLDPR May 10 15:28:14 2024
Boss-Extrude2	Solid Body	Mass: 0,000164337 kg Volume: 1,19954e-07 m^3	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-



		Density:1 370 kg/m ³ Weight:0,0016105 N	05\SpringHold_part1.SLD PRT May 10 15:28:14 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m ³ Density:1 370 kg/m ³ Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 15:28:14 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m ³ Density:1 370 kg/m ³ Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000428204 kg Volume:3,12546e-07 m ³ Density:1 370,05 kg/m ³ Weight:0,0041964 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLD.PRT May 10 15:28:14 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m ³ Density:1 370 kg/m ³ Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 15:28:14 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m ³ Density:1 370 kg/m ³ Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 15:28:14 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m ³ Density:1 370 kg/m ³ Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000428203 kg Volume:3,12546e-07 m ³ Density:1 370,05 kg/m ³ Weight:0,00419639 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLD.PRT May 10 15:28:14 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m ³ Density:1 370 kg/m ³ Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 15:28:14 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m ³ Density:1 370 kg/m ³ Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 15:28:14 2024





Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m^3 Density:1 370 kg/m^3 Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000428204 kg Volume:3,12546e-07 m^3 Density:1 370,05 kg/m^3 Weight:0,0041964 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLD PRT May 10 15:28:14 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m^3 Density:1 370 kg/m^3 Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 15:28:14 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m^3 Density:1 370 kg/m^3 Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 15:28:14 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,00132768 kg Volume:9,69113e-07 m^3 Density:1 370 kg/m^3 Weight:0,0130113 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Plate_Lock.SLD PRT May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-





		Weight:0,0317073 N	05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
Fillet2	Solid Body	Mass:0,0297987 kg Volume:2,17509e-05 m^3 Density:1 370 kg/m^3 Weight:0,292027 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\XY_wall_Negative_Z.S LDPRT May 10 15:28:15 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024





Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024





		Weight:0,00228647 N	05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024





Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude2	Solid Body	Mass:0,00269269 kg Volume:2,17153e-06 m^3 Density:1 240 kg/m^3 Weight:0,0263884 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\AntennaBoard_AntennaLockRoof.SLDPR May 10 15:28:12 2024
Cut-Extrude2	Solid Body	Mass:0,00269269 kg Volume:2,17153e-06 m^3 Density:1 240 kg/m^3	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-





		Weight:0,0263884 N	05\AntennaBoard_Antenn aLockRoof.SLDPR May 10 15:28:12 2024
Split Line4	Solid Body	Mass:0,0206986 kg Volume:1,51084e-05 m^3 Density:1 370 kg/m^3 Weight:0,202846 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\Antenne_ReleaseBoar d.SLDPR May 10 15:28:12 2024
Split Line1	Solid Body	Mass:0,00625083 kg Volume:4,56265e-06 m^3 Density:1 370 kg/m^3 Weight:0,0612582 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\BatterySpringPlate.SL DPRT May 10 15:28:12 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	Mass:0,0222911 kg Volume:1,62708e-05 m^3 Density:1 370 kg/m^3 Weight:0,218452 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\XY_wall_Positive_Z.SL DPRT May 10 15:28:24 2024
Fillet5	Solid Body	Mass:0,0318527 kg Volume:2,32501e-05 m^3 Density:1 370 kg/m^3 Weight:0,312156 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\ZY_Wall.SLDPR May 10 15:28:24 2024
Fillet5	Solid Body	Mass:0,0318527 kg Volume:2,32501e-05 m^3 Density:1 370 kg/m^3	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\ZY_Wall.SLDPR



		Weight: 0,312156 N	May 10 15:28:24 2024
---	--	--------------------	----------------------

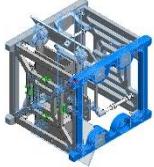
Study Properties

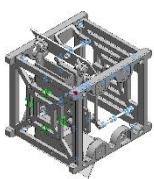
Study name	Frequency Analysis
Analysis type	Frequency
Mesh type	Solid Mesh
Number of frequencies	5
Solver type	FFEPlus
Soft Spring:	Off
Incompatible bonding options	Automatic
Thermal option	Include temperature loads
Zero strain temperature	298 Kelvin
Include fluid pressure effects from SOLIDWORKS Flow Simulation	Off
Result folder	SOLIDWORKS document (C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05)

Units

Unit system:	SI (MKS)
Length/Displacement	mm
Temperature	Kelvin
Angular velocity	Rad/sec
Pressure/Stress	N/m ²

Material Properties

Model Reference	Properties	Components
	<p>Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m² Tensile strength: 3,753e+07 N/m² Mass density: 1 370 kg/m³ Elastic modulus: 3,00159e+09 N/m² Poisson's ratio: 0,38</p>	SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-1), SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-2), SolidBody 1(Cut-Extrude1)(HingeLockPart-6), SolidBody 1(Cut-Extrude1)(HingeLockPart-7), SolidBody 1(Cut-Extrude1)(HingeLockPart-8), SolidBody 1(Cut-Extrude1)(HingeLockPart-9), SolidBody 1(Fillet1)(Lock1-11), SolidBody 1(Fillet1)(Lock1-5), SolidBody 1(Fillet1)(Lock1-6), SolidBody 1(Fillet1)(Lock1-7), SolidBody 1(Fillet1)(Lock1-8), SolidBody 1(Fillet1)(Lock1-9), SolidBody 1(Split Line1)(Pin_Sealing-1), SolidBody 1(Split Line1)(Pin_Sealing-2), SolidBody 1(Split Line1)(Pin_Sealing-3), SolidBody 1(Split Line1)(Pin_Sealing-4), SolidBody 1(Split Line1)(Pin_Sealing-5), SolidBody 1(Split Line1)(Pin_Sealing-6), SolidBody 1(Split Line1)(Pin_Sealing-7), SolidBody 1(Split Line1)(Pin_Sealing-8), SolidBody 1(Split Line3)(SealingAntenna-1), SolidBody 1(Split Line3)(SealingAntenna-2), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-1),

		<p>SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-2), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-3), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-4), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Plate_Lock-1), SolidBody 1(Fillet2)(Sub_MainAsm-1/LowerAsm_CRAFT-2/XY_wall_Negative_Z-1), SolidBody 1(SplitLine1)(Sub_MainAsm-1/UpperAsm_CRAFT-1/BatterySpringPlate-1), SolidBody 1(Fillet5)(ZY_Wall-1), SolidBody 1(Fillet5)(ZY_Wall-2)</p>
Curve Data:N/A		
	<p>Name: AISI 316 Annealed Stainless Steel Bar (SS) Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,37895e+08 N/m² Tensile strength: 5,5e+08 N/m² Mass density: 8 000 kg/m³ Elastic modulus: 1,93e+11 N/m² Poisson's ratio: 0,3 Thermal expansion coefficient: 1,6e-05 /Kelvin</p>	<p>SolidBody 1(Cut-Extrude1)(M3 locking Nut-1), SolidBody 1(Cut-Extrude1)(M3 locking Nut-2), SolidBody 1(Cut-Extrude1)(M3 locking Nut-3), SolidBody 1(Cut-Extrude1)(M3 locking Nut-4), SolidBody 1(Cut-Extrude1)(M3 locking Nut-5), SolidBody 1(Cut-Extrude1)(M3 locking Nut-6), SolidBody 1(Cut-Extrude1)(M3 locking Nut-7), SolidBody 1(Cut-Extrude1)(M3 locking Nut-8), SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-1),</p>



SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-10),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-11),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-12),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-13),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-14),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-15),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-16),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-17),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-18),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-19),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-20),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-21),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-22),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-23),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-24),





C.R.A.F.T

STB
12.05.2024

SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-25),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-26),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-27),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-28),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-29),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-3),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-30),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-31),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-32),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-4),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-5),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-6),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-7),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-8),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-9)

Curve Data:N/A

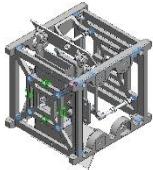


SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm

21

	<p>Name: AISI 316 Stainless Steel Sheet (SS)</p> <p>Model type: Linear Elastic Isotropic</p> <p>Default failure criterion: Unknown</p> <p>Yield strength: 1,72369e+08 N/m²</p> <p>Tensile strength: 5,8e+08 N/m²</p> <p>Mass density: 8 000 kg/m³</p> <p>Elastic modulus: 1,93e+11 N/m²</p> <p>Poisson's ratio: 0,27</p> <p>Thermal expansion coefficient: 1,6e-05 /Kelvin</p>	<p>SolidBody 1(M3_Thread)(M3x10_PZ1 head-1), SolidBody 1(M3_Thread)(M3x10_PZ1 head-2), SolidBody 1(M3_Thread)(M3x10_PZ1 head-3), SolidBody 1(M3_Thread)(M3x10_PZ1 head-4), SolidBody 1(M3_Thread)(M3x10_PZ1 head-5), SolidBody 1(M3_Thread)(M3x10_PZ1 head-6), SolidBody 1(M3_Thread)(M3x10_PZ1 head-7), SolidBody 1(M3_Thread)(M3x10_PZ1 head-8), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-1), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-2), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-3), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-4), SolidBody 1(M3_Thread)(Sub_MainAsm-1/UpperAsm_CRAFT-1/M3x50_PZ1 head-1), SolidBody 1(M3_Thread)(Sub_MainAsm-1/UpperAsm_CRAFT-1/M3x50_PZ1 head-2), SolidBody 1(M3_Thread)(Sub_MainAsm-</p>
---	---	--

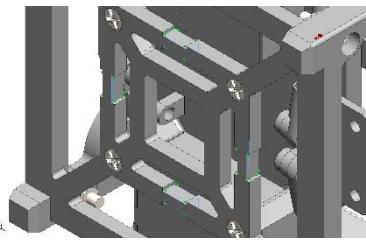
		1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-3), SolidBody 1(M3_Thread)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-4)
Curve Data:N/A		
	Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m ² Tensile strength: 3,753e+07 N/m ² Mass density: 1 370 kg/m ³ Elastic modulus: 3,00159e+09 N/m ² Poisson's ratio: 0,38 Thermal expansion coefficient: 3,8 /Kelvin	SolidBody 1(Outer_Sylinder)(Spring_asm-1/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-1/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-1/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-1/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-2/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-2/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-2/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-2/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-4/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-4/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-4/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-4/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-6/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-6/ShockPart-1),

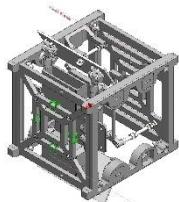


		SolidBody 1(Boss-Extrude2)(Spring_asm-6/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-6/SpringHold_part2-1), SolidBody 1(SplitLine4)(Sub_MainAsm-1/UpperAsm_CRAFT-1/Antenne_ReleaseBoard-1), SolidBody 1(SplitLine1)(Sub_MainAsm-1/UpperAsm_CRAFT-1/XY_wall_Positive_Z-1)
Curve Data:N/A		
	Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m^2 Tensile strength: 3,753e+07 N/m^2 Mass density: 1 240 kg/m^3 Elastic modulus: 3,0059e+08 N/m^2 Poisson's ratio: 0,38	SolidBody 1(Cut-Extrude2)(Sub_MainAsm-1/UpperAsm_CRAFT-1/AntennaBoard_AntennaLocRoof-1), SolidBody 1(Cut-Extrude2)(Sub_MainAsm-1/UpperAsm_CRAFT-1/AntennaBoard_AntennaLocRoof-2)
Curve Data:N/A		



Loads and Fixtures

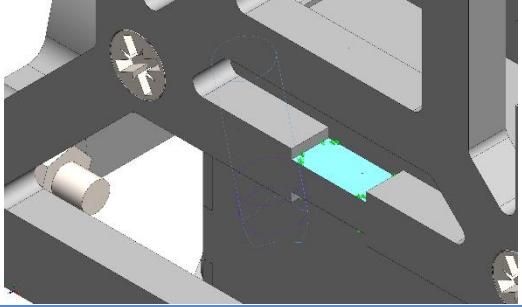
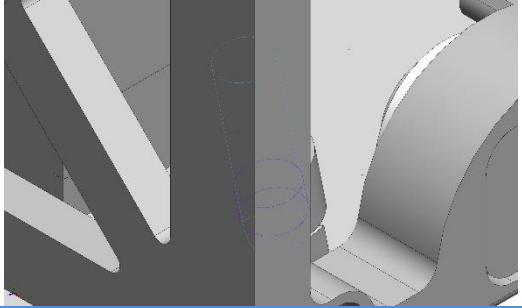
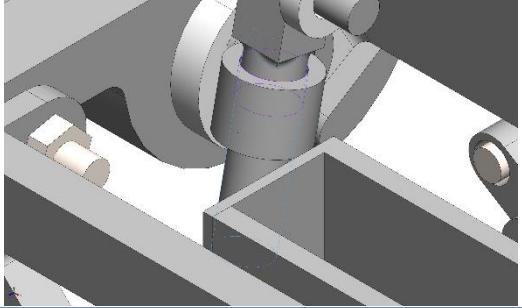
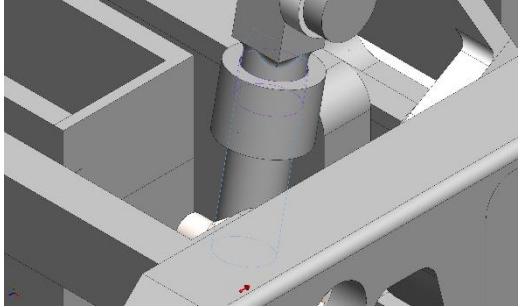
Fixture name	Fixture Image	Fixture Details
Roller/Slider-1		Entities: 12 face(s) Type: Roller/Slider

Load name	Load Image	Load Details
Gravity-1		Reference: Front Plane Values: 0 0 -9,81 Units: m/s ²

Connector Definitions

No Data

Interaction Information

Interaction	Interaction Image	Interaction Properties
Local Interaction-3890		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-3891		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-3892		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-3893		Type: Bonded interaction pair Entities: 2 face(s)



C.R.A.F.T

STB
12.05.2024

Local Interaction-4153		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4154		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4155		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4156		Type: Bonded interaction pair Entities: 2 face(s)



SOLIDWORKS

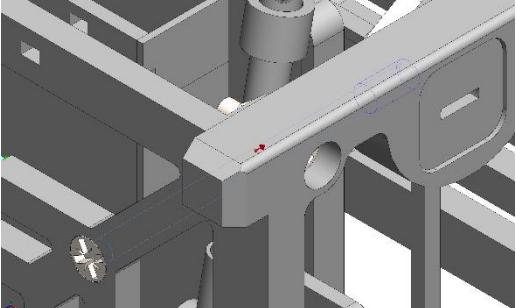
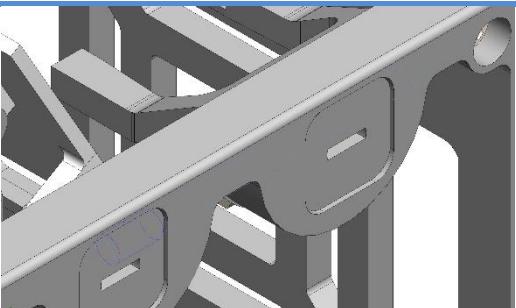
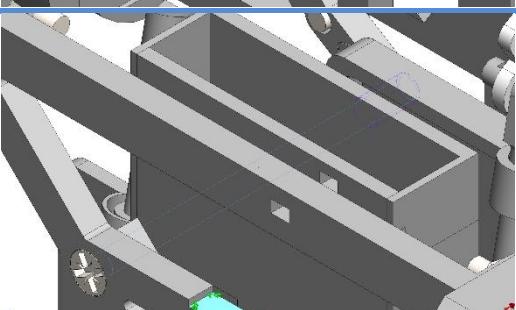
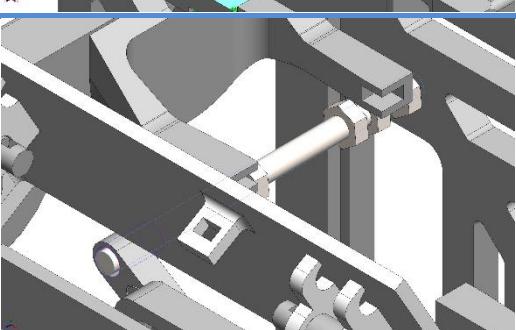
Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 27



C.R.A.F.T

STB
12.05.2024

Local Interaction-4416		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4417		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4418		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4419		Type: Bonded interaction pair Entities: 2 face(s)



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 28



C.R.A.F.T

STB
12.05.2024

Local Interaction-4420		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4421		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4422		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4423		Type: Bonded interaction pair Entities: 2 face(s)



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

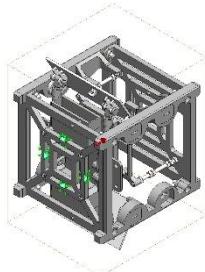
Simulation of Main_asm 29



C.R.A.F.T

STB
12.05.2024

Global Interaction



Type: Bonded
Components: 1 component(s)
Options: Independent mesh



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 30



Mesh information

Mesh type	Solid Mesh
Mesher Used:	Blended curvature-based mesh
Jacobian points for High quality mesh	16 Points
Maximum element size	15,3777 mm
Minimum element size	0,768884 mm
Mesh Quality	High
Remesh failed parts independently	Off

Mesh information - Details

Total Nodes	285780
Total Elements	153686
Maximum Aspect Ratio	88,119
% of elements with Aspect Ratio < 3	92,4
Percentage of elements with Aspect Ratio > 10	0,269
Percentage of distorted elements	0
Time to complete mesh(hh:mm:ss):	00:00:37
Computer name:	S-K-1230-06

Mesh Quality Plots

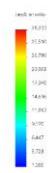
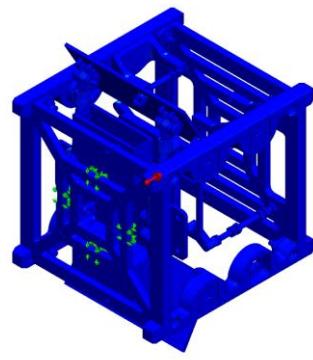
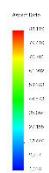
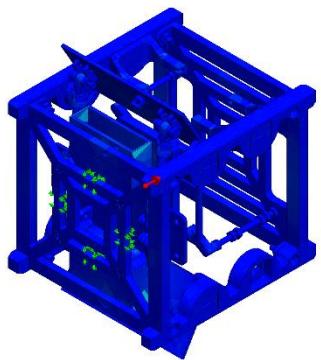
Because of the amount of data required in the analysis, didn't make a mesh quality change on the areas with a Aspect & Jacobian Ratio higher than 10. We will therefore accept a non-accurate value in the analysis but define the results using this mesh as a "real" result related to this mesh. If we had a computer with more memory, a mesh change in the areas with high ratios would be done to gain a more predictable value.





C.R.A.F.T

STB
12.05.2024



Sensor Details

No Data



SOLIDWORKS

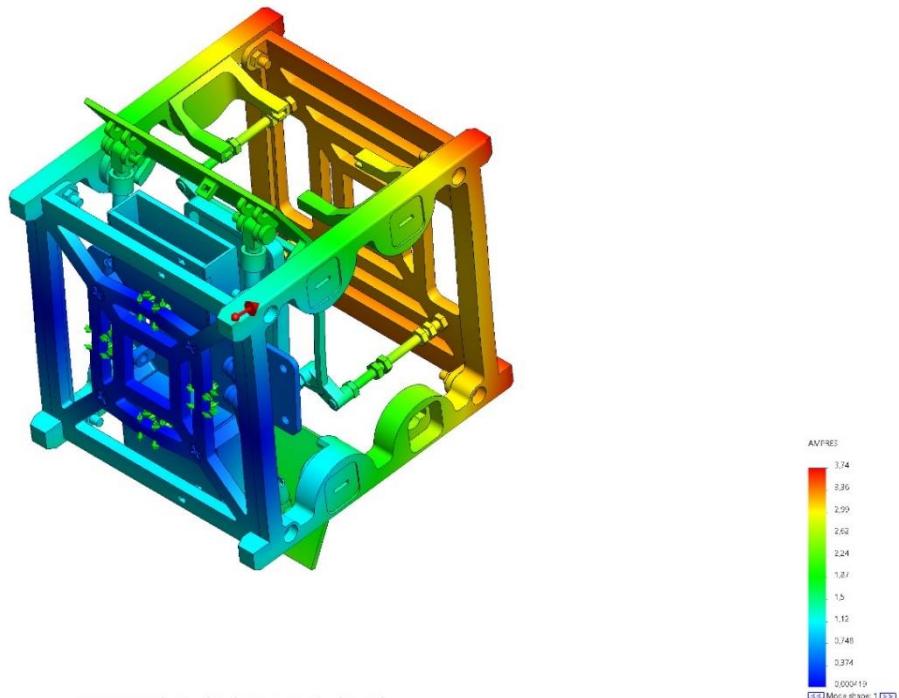
Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 32

Study Results

Name	Type	Min	Max
Amplitude1	AMPRES: Resultant Amplitude Plot for Mode Shape: 1 (Value = 60,5733 Hz)	0,000419 Node: 192279	3,74 Node: 211301

Mode name: Mode1
Study name: Frequency Analysis (Default)
Plot type: Frequency Amplitude
Mode Shape 1 Value = 60,5733 Hz
Deformation scale: 0,00416437

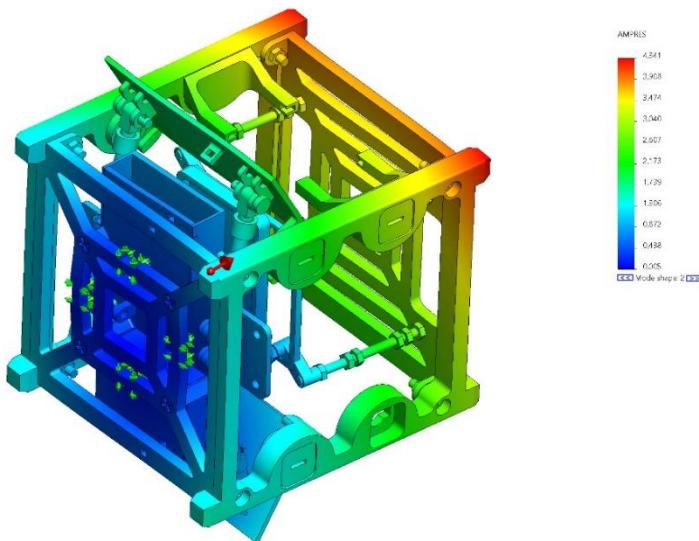


Main_asm-Frequency Analysis-Amplitude-Amplitude1



Name	Type	Min	Max
Amplitude2	AMPRES: Resultant Amplitude Plot for Mode Shape: 2 (Value = 61,8998 Hz)	0,005 Node: 138814	4,341 Node: 248680

Mode Name: Main
Study Name: Main (using Analysis Default)
Plot type: Frequency Amplitude2
Mode Shape: 2 Value = 61.9 Hz
Deformation scale: 0.005/82.71



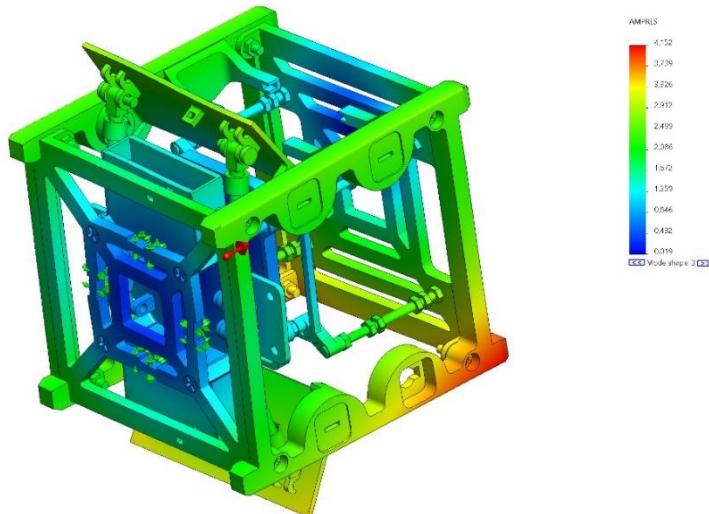
SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Frequency Analysis-Amplitude-Amplitude2



Name	Type	Min	Max
Amplitude3	AMPRES: Resultant Amplitude Plot for Mode Shape: 3 (Value = 65,4303 Hz)	0,019 Node: 155040	4,152 Node: 248699

Mode name: Main
Step by Step Frequency Analysis (Default)
Plot type: Frequency Amplitude3
Mode Shape: 3. Value = 65,4319
Deformation scale: 0,00425438

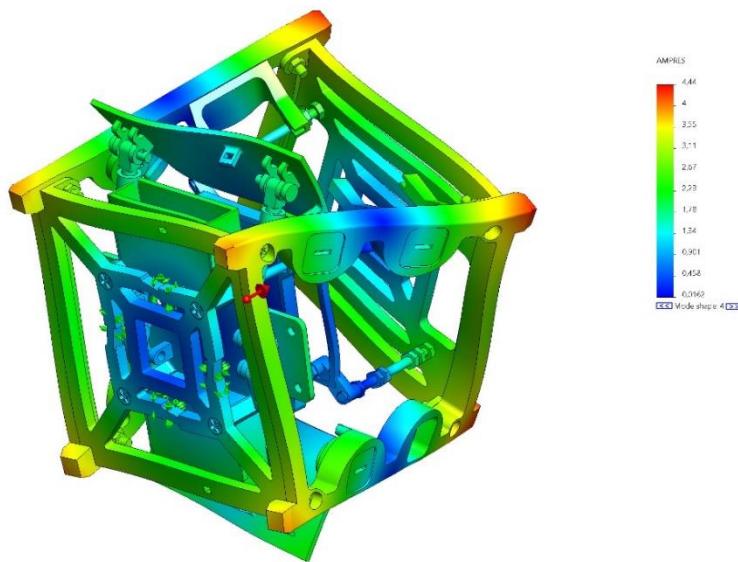


SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Frequency Analysis-Amplitude-Amplitude3

Name	Type	Min	Max
Amplitude4	AMPRES: Resultant Amplitude Plot for Mode Shape: 4 (Value = 248,786 Hz)	0,0162 Node: 134296	4,44 Node: 248673

Model name: Main.asm
Study name: Frequency Analysis (Default)
Plot type: Frequency Amplitude.cis
Mode Shape: 4 Value = 248,79 Hz
Deformation scale: 0,00460/82

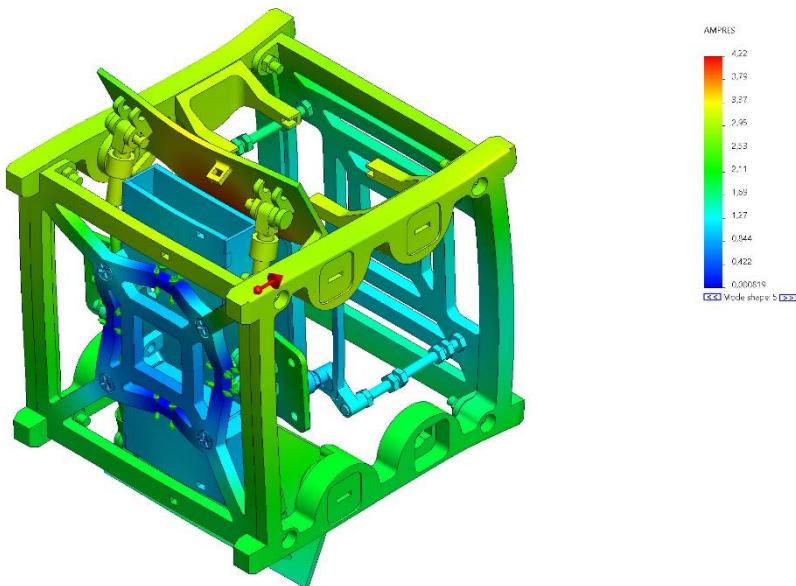


SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Frequency Analysis-Amplitude-Amplitude4

Name	Type	Min	Max
Amplitude5	AMPRES: Resultant Amplitude Plot for Mode Shape: 5 (Value = 256,377 Hz)	0,000819 Node: 182159	4,22 Node: 36480

Mode name: Main_asm
Study name: Frequency Analysis (Default)
Plot type: Frequency Amplitude
Mode Shape: 5 Value = 256,377 Hz
Deformation scale: 0,00050419



SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Frequency Analysis-Amplitude-Amplitude5

Mode List

Frequency Number	Rad/sec	Hertz	Seconds
1	380,59	60,573	0,016509
2	388,93	61,9	0,016155
3	411,11	65,43	0,015283
4	1 563,2	248,79	0,0040195
5	1 610,9	256,38	0,0039005

Mass Participation (Normalized)

Mode Number	Frequency(Hertz)	X direction	Y direction	Z direction
1	60,573	0,00018159	0,49996	0,00028921
2	61,9	0,44415	0,0002236	3,0419e-07
3	65,43	0,042941	5,2969e-06	2,0248e-08
4	248,79	5,7978e-05	4,9728e-05	4,0394e-06
5	256,38	1,8657e-07	0,01492	0,85433
		Sum X = 0,48733	Sum Y = 0,51515	Sum Z = 0,85463

Conclusion

According to the data about the natural frequency from five amplitudes, we will gain a resonance if the model from the analysis experiencing a vibration-frequency with a value equal:

- 60.573 Hz,
- 61.9 Hz,
- 65.43 Hz,
- 248.79 Hz or
- 256.38 Hz.

It's important to note that the result would change if more components were included. The frequency due to vibration, is calculated with formula:

$$f_n = \frac{1}{2\pi} \sqrt{\frac{k}{m}}$$

Where the only variables changing the frequency values defined for resonance is the k , the stiffness, and m , the total mass. Since our product has mass requirements, it would be easier to change the value of k by increasing or decreasing it the stiffness on the area with a high amplitude value. In our case, it would be more likely to increase it to gain a higher node value, since the simplified vibration test will use a frequency equal 60 Hz.

V.4 FEA: Harmonic dynamic analysis

The harmonic analysis represents a simulation of the structures response to the forced vibration.

To create the analysis, material damping ratio is required. But, during web-searching of this value, non-data was found and descriptions about the value needed to be tested to gained resulted with simplifications instead. The formula for the the damping ratio is:

$$\zeta = \frac{c}{c_c} = \frac{\text{actual damping}}{\text{critical damping}}$$

where actual damping represents the damping in a real-world system, and the critical damping represents the value for damping calculated by using formula

$$c_c = 2\sqrt{km}$$

or

$$c_c = 2m\sqrt{\frac{k}{m}} = 2m\omega_n$$

where m is the mass, k is the stiffness and ω_n is the natural rotational velocity. To define the natural rotational velocity, the natural frequency could be used:

$$f_n = \frac{1}{2\pi}\omega_n \rightarrow \omega_n = 2\pi f_n$$

By combining all equations, the new formula for the damping ratio is:

$$\zeta = \frac{c}{c_c} = \frac{c}{(2m)(2\pi f_n)} = \frac{c}{4m\pi f_n} \quad (55)$$

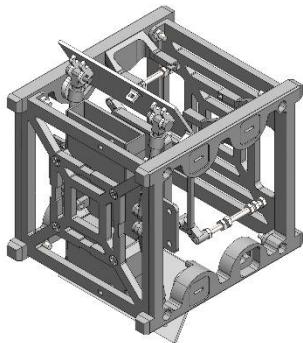
The value for actual damping requires testing to gain the correct value. Since it's not possible to gain the data for the actual damping, assumption about it being equal the critical damping was chosen. Therefore, the the value for damping ratio, $\zeta = 1$.

Note: when the actual damping is equal the critical damping, $\zeta = 1$, the system will return to equilibrium without oscillation.

Under this section is the result of this section title analysis.



C.R.A.F.T



1

Description

The analysis illustrates a harmonic simulation, using linear dynamic analysis. The simulation has an Aspect ratio with a small amount percent higher than 10, but with mostly under 3 percent. Because the simulation required a large amount of data, we have chosen to simplify it, and accept the small amount of percent higher than 10 but the location of the values higher than 10 percent has been checked.

Components such as electronical circuit boards and cables, bolts, and nuts with threads, have been removed. Some bolts have been included but they don't have threads. The model has only a design representing balloon mounting and not parachuting. The model can be re-designed, but it's not our object at this moment to do so.

Simulation of assembly: main_asm

Date: Monday 13. May 2024

Designer: STB

Study name: Harmonical Dynamical Analysis

Analysis type: Linear dynamic analysis
(Harmonic)

Table of Contents

Description	1
Assumptions & model information	2
Study Properties.....	17
Units	18
Material Properties	19
Loads and Fixtures	26
Connector Definitions	27
Interaction Information	28
Mesh information.....	33
Sensor Details.....	33
Study Results	34
Conclusion.....	38



SOLIDWORKS

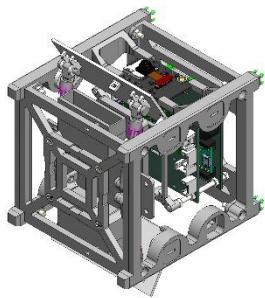
Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 1

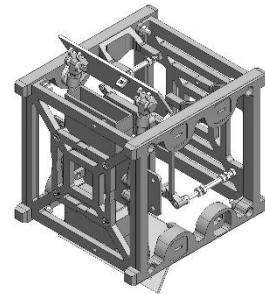
Assumptions & model information

We couldn't use the gravitational load in this analysis (without making a specified method by creating effected area of the force and place it with an equal position of the mass center of the model) and decided to assume that the force from the springs could give a data proven as informative. We also decided to assume that the fixtures used would illustrate the model be placed on the plate used for the realistic simplified vibration test. The model is analyzed to represent a circumstance with none-electronical components and with an environment where none disturbing objects occurs.

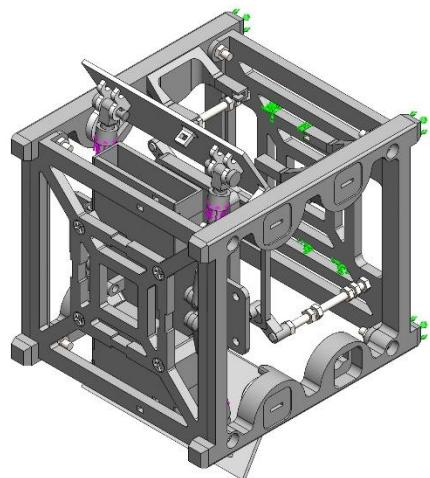
The reason is that including electronical devices required material definition on every part for the circuit board which we don't have. Since the model is a solid tight fitted with free components local interactions was made at the areas where components not touching each other due to none spring and thread parts. The bonded global interaction is used on the other components in the model. The model has components with threads and small sizes; the threads are removed, and some mechanical components (such as bolts, nuts, and springs) are not included in the analysis. The model represents the real sizes of the product, with PETG as the material for all the 3D printing parts. The total mass of the model analyzed is higher (we don't have the specific value at this moment) than the real product because we will not use 100% infill in 3D printing.



Original Model



Model Analyzed

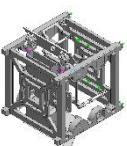
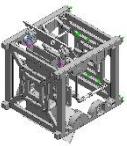
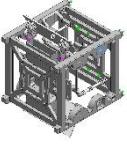
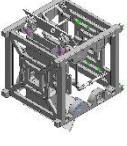
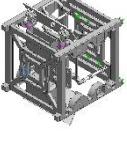


Model name: Main_asm
Current Configuration: Default

Solid Bodies

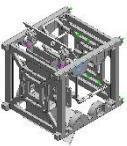
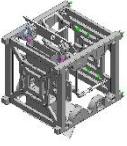
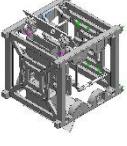
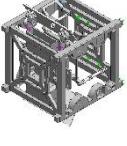
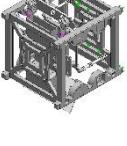
Document Name and Reference	Treated As	Volumetric Properties	Document Path/Date Modified
Split Line3	Solid Body	Mass:0,00290598 kg Volume:2,12136e-06 m^3 Density:1 369,87 kg/m^3 Weight:0,0284786 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Camera_Holder.SLDPR T May 10 15:28:12 2024
Split Line3	Solid Body	Mass:0,00290598 kg Volume:2,12136e-06 m^3 Density:1 369,87 kg/m^3 Weight:0,0284786 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Camera_Holder.SLDPR T May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDPR T May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDPR T May 10 15:28:12 2024



			
Cut-Extrude1 	Solid Body	Mass: 0,000567144 kg Volume: 4,13973e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDprt May 10 15:28:12 2024
Cut-Extrude1 	Solid Body	Mass: 0,000567144 kg Volume: 4,13973e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00555801 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\HingeLockPart.SLDprt May 10 15:28:12 2024
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Fillet1	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024

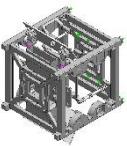




			
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Lock1.SLDprt May 10 15:28:12 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
Cut-Extrude1 	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDprt May 10 15:28:12 2024

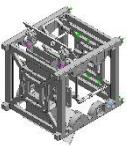




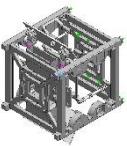
			
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR ^T May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR ^T May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR ^T May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass: 0,000233313 kg Volume: 2,91641e-08 m^3 Density: 8 000 kg/m^3 Weight: 0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR ^T May 10 15:28:12 2024
M3_Thread	Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1 head.SLDPR ^T May 10 15:28:12 2024
M3_Thread	Solid Body	Mass: 0,000973491 kg Volume: 1,21686e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,00954021 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1 head.SLDPR ^T May 10 15:28:12 2024

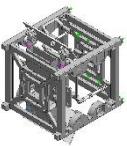




			
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024
M3_Thread	Solid Body	<p>Mass:0,000973491 kg Volume:1,21686e-07 m^3 Density:8 000 kg/m^3 Weight:0,00954021 N</p>	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x10_PZ1_head.SLDPRT May 10 15:28:12 2024



			
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024

			
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Pin_Sealing.SLDPR May 10 15:28:13 2024
Split Line3	Solid Body	Mass: 0,00973089 kg Volume: 7,10284e-06 m^3 Density: 1 370 kg/m^3 Weight: 0,0953628 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SealingAntenna.SLDPR T May 10 15:28:14 2024
Split Line3	Solid Body	Mass: 0,00973089 kg Volume: 7,10284e-06 m^3 Density: 1 370 kg/m^3 Weight: 0,0953628 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SealingAntenna.SLDPR T May 10 15:28:14 2024
Outer_Sylinder	Solid Body	Mass: 0,000373091 kg Volume: 2,72329e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLDPR RT May 10 15:28:12 2024
Fillet1	Solid Body	Mass: 0,000428203 kg Volume: 3,12546e-07 m^3 Density: 1 370,05 kg/m^3 Weight: 0,00419639 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLDPR May 10 15:28:14 2024
Boss-Extrude2	Solid Body	Mass: 0,000164337 kg Volume: 1,19954e-07 m^3	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-



		Density:1 370 kg/m ³ Weight:0,0016105 N	05\SpringHold_part1.SLD PRT May 10 15:28:14 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m ³ Density:1 370 kg/m ³ Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 15:28:14 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m ³ Density:1 370 kg/m ³ Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000428204 kg Volume:3,12546e-07 m ³ Density:1 370,05 kg/m ³ Weight:0,0041964 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLD.PRT May 10 15:28:14 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m ³ Density:1 370 kg/m ³ Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 15:28:14 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m ³ Density:1 370 kg/m ³ Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 15:28:14 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m ³ Density:1 370 kg/m ³ Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000428203 kg Volume:3,12546e-07 m ³ Density:1 370,05 kg/m ³ Weight:0,00419639 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLD.PRT May 10 15:28:14 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m ³ Density:1 370 kg/m ³ Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 15:28:14 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m ³ Density:1 370 kg/m ³ Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 15:28:14 2024





Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m^3 Density:1 370 kg/m^3 Weight:0,00365629 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\InternalStampPin.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000428204 kg Volume:3,12546e-07 m^3 Density:1 370,05 kg/m^3 Weight:0,0041964 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ShockPart.SLD PRT May 10 15:28:14 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m^3 Density:1 370 kg/m^3 Weight:0,0016105 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part1.SLD PRT May 10 15:28:14 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m^3 Density:1 370 kg/m^3 Weight:0,00442236 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\SpringHold_part2.SLD PRT May 10 15:28:14 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Hook_Parts.SLD PRT May 10 15:28:12 2024
Fillet1	Solid Body	Mass:0,00132768 kg Volume:9,69113e-07 m^3 Density:1 370 kg/m^3 Weight:0,0130113 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\Plate_Lock.SLD PRT May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-





		Weight:0,0317073 N	05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
Fillet2	Solid Body	Mass:0,0297987 kg Volume:2,17509e-05 m^3 Density:1 370 kg/m^3 Weight:0,292027 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\XY_wall_Negative_Z.S LDPRT May 10 15:28:15 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024





Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024





		Weight:0,00228647 N	05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\ M3 locking Nut.SLDPR May 10 15:28:12 2024





Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude1	Solid Body	Mass:0,000233313 kg Volume:2,91641e-08 m^3 Density:8 000 kg/m^3 Weight:0,00228647 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\M3 locking Nut.SLDPR May 10 15:28:12 2024
Cut-Extrude2	Solid Body	Mass:0,00269269 kg Volume:2,17153e-06 m^3 Density:1 240 kg/m^3 Weight:0,0263884 N	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05\AntennaBoard_AntennaLockRoof.SLDPR May 10 15:28:12 2024
Cut-Extrude2	Solid Body	Mass:0,00269269 kg Volume:2,17153e-06 m^3 Density:1 240 kg/m^3	C:\Users\235474\Downloads\FEA_BackUp 08-34-10-

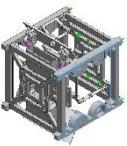




		Weight:0,0263884 N	05\AntennaBoard_Antenn aLockRoof.SLDPR May 10 15:28:12 2024
Split Line4	Solid Body	Mass:0,0206986 kg Volume:1,51084e-05 m^3 Density:1 370 kg/m^3 Weight:0,202846 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\Antenne_ReleaseBoar d.SLDPR May 10 15:28:12 2024
Split Line1	Solid Body	Mass:0,00625083 kg Volume:4,56265e-06 m^3 Density:1 370 kg/m^3 Weight:0,0612582 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\BatterySpringPlate.SL DPRT May 10 15:28:12 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\M3x50_PZ1 head.SLDPR May 10 15:28:13 2024
Split Line1	Solid Body	Mass:0,0222911 kg Volume:1,62708e-05 m^3 Density:1 370 kg/m^3 Weight:0,218452 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\XY_wall_Positive_Z.SL DPRT May 10 15:28:24 2024
Fillet5	Solid Body	Mass:0,0318527 kg Volume:2,32501e-05 m^3 Density:1 370 kg/m^3 Weight:0,312156 N	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\ZY_Wall.SLDPR May 10 15:28:24 2024
Fillet5	Solid Body	Mass:0,0318527 kg Volume:2,32501e-05 m^3 Density:1 370 kg/m^3	C:\Users\235474\Download s\FEA_BackUp 08-34-10-05\ZY_Wall.SLDPR





		Weight: 0,312156 N	May 10 15:28:24 2024
---	--	--------------------	----------------------

Study Properties

Study name on SWS	Harmonical Dynamical Analysis
Analysis type	Linear dynamic analysis (Harmonic)
Mesh type	Solid Mesh
Number of frequencies	15
Solver type	FFEPlus
Soft Spring:	Off
Incompatible bonding options	Automatic
Thermal option	Include temperature loads
Zero strain temperature	298 Kelvin
Include fluid pressure effects from SOLIDWORKS Flow Simulation	Off
Lower frequency limit	0 Hz
Upper frequency limit	1 000 Hz
Number of frequencies	15
Interpolation	Logarithmic
Result folder	SOLIDWORKS document (C:\Users\235474\Downloads\FEA_BackUp 08-34-10-05)





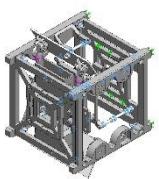
Units

Unit system:	SI (MKS)
Length/Displacement	mm
Temperature	Kelvin
Angular velocity	Rad/sec
Pressure/Stress	N/m ²



Material Properties

Model Reference	Properties	Components
	<p>Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m² Tensile strength: 3,753e+07 N/m² Elastic modulus: 3,00159e+09 N/m² Poisson's ratio: 0,38 Mass density: 1 370 kg/m³</p>	SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-1), SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-2), SolidBody 1(Cut-Extrude1)(HingeLockPart-6), SolidBody 1(Cut-Extrude1)(HingeLockPart-7), SolidBody 1(Cut-Extrude1)(HingeLockPart-8), SolidBody 1(Cut-Extrude1)(HingeLockPart-9), SolidBody 1(Fillet1)(Lock1-11), SolidBody 1(Fillet1)(Lock1-5), SolidBody 1(Fillet1)(Lock1-6), SolidBody 1(Fillet1)(Lock1-7), SolidBody 1(Fillet1)(Lock1-8), SolidBody 1(Fillet1)(Lock1-9), SolidBody 1(Split Line1)(Pin_Sealing-1), SolidBody 1(Split Line1)(Pin_Sealing-2), SolidBody 1(Split Line1)(Pin_Sealing-3), SolidBody 1(Split Line1)(Pin_Sealing-4), SolidBody 1(Split Line1)(Pin_Sealing-5), SolidBody 1(Split Line1)(Pin_Sealing-6), SolidBody 1(Split Line1)(Pin_Sealing-7), SolidBody 1(Split Line1)(Pin_Sealing-8), SolidBody 1(Split Line3)(SealingAntenna-1), SolidBody 1(Split Line3)(SealingAntenna-2), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-1),

		<p>SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-2), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-3), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-4), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Plate_Lock-1), SolidBody 1(Fillet2)(Sub_MainAsm-1/LowerAsm_CRAFT-2/XY_wall_Negative_Z-1), SolidBody 1(Split Line1)(Sub_MainAsm-1/UpperAsm_CRAFT-1/BatterySpringPlate-1), SolidBody 1(Fillet5)(ZY_Wall-1), SolidBody 1(Fillet5)(ZY_Wall-2)</p>
Curve Data:N/A		
	<p>Name: AISI 316 Annealed Stainless Steel Bar (SS) Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,37895e+08 N/m² Tensile strength: 5,5e+08 N/m² Elastic modulus: 1,93e+11 N/m² Poisson's ratio: 0,3 Mass density: 8 000 kg/m³ Thermal expansion coefficient: 1,6e-05 /Kelvin</p>	<p>SolidBody 1(Cut-Extrude1)(M3 locking Nut-1), SolidBody 1(Cut-Extrude1)(M3 locking Nut-2), SolidBody 1(Cut-Extrude1)(M3 locking Nut-3), SolidBody 1(Cut-Extrude1)(M3 locking Nut-4), SolidBody 1(Cut-Extrude1)(M3 locking Nut-5), SolidBody 1(Cut-Extrude1)(M3 locking Nut-6), SolidBody 1(Cut-Extrude1)(M3 locking Nut-7), SolidBody 1(Cut-Extrude1)(M3 locking Nut-8), SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-1),</p>



SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-10),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-11),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-12),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-13),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-14),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-15),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-16),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-17),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-18),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-19),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-20),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-21),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-22),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-23),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-24),





C.R.A.F.T

STB
13.05.2024

SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-25),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-26),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-27),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-28),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-29),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-3),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-30),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-31),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-32),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-4),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-5),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-6),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-7),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-8),
SolidBody 1(Cut-Extrude1)(Sub_MainAsm-1/M3 locking Nut-9)

Curve Data:N/A

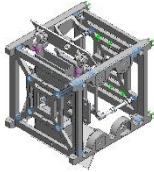


SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm

22

	<p>Name: AISI 316 Stainless Steel Sheet (SS)</p> <p>Model type: Linear Elastic Isotropic</p> <p>Default failure criterion: Unknown</p> <p>Yield strength: 1,72369e+08 N/m²</p> <p>Tensile strength: 5,8e+08 N/m²</p> <p>Elastic modulus: 1,93e+11 N/m²</p> <p>Poisson's ratio: 0,27</p> <p>Mass density: 8 000 kg/m³</p> <p>Thermal expansion coefficient: 1,6e-05 /Kelvin</p>	<p>SolidBody 1(M3_Thread)(M3x10_PZ1 head-1), SolidBody 1(M3_Thread)(M3x10_PZ1 head-2), SolidBody 1(M3_Thread)(M3x10_PZ1 head-3), SolidBody 1(M3_Thread)(M3x10_PZ1 head-4), SolidBody 1(M3_Thread)(M3x10_PZ1 head-5), SolidBody 1(M3_Thread)(M3x10_PZ1 head-6), SolidBody 1(M3_Thread)(M3x10_PZ1 head-7), SolidBody 1(M3_Thread)(M3x10_PZ1 head-8), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-1), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-2), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-3), SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-4), SolidBody 1(M3_Thread)(Sub_MainAsm-1/UpperAsm_CRAFT-1/M3x50_PZ1 head-1), SolidBody 1(M3_Thread)(Sub_MainAsm-1/UpperAsm_CRAFT-1/M3x50_PZ1 head-2), SolidBody 1(M3_Thread)(Sub_MainAsm-</p>
---	---	--

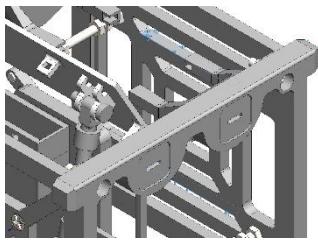
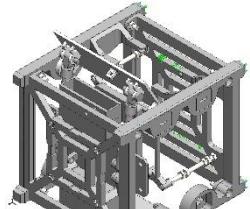
		1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-3), SolidBody 1(M3_Thread)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-4)
Curve Data:N/A		
	Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m ² Tensile strength: 3,753e+07 N/m ² Elastic modulus: 3,00159e+09 N/m ² Poisson's ratio: 0,38 Mass density: 1 370 kg/m ³ Thermal expansion coefficient: 3,8 /Kelvin	SolidBody 1(Outer_Sylinder)(Spring_asm-1/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-1/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-1/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-1/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-2/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-2/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-2/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-2/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-4/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-4/ShockPart-1), SolidBody 1(Boss-Extrude2)(Spring_asm-4/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-4/SpringHold_part2-1), SolidBody 1(Outer_Sylinder)(Spring_asm-6/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm-6/ShockPart-1),

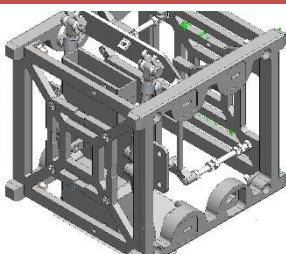


		SolidBody 1(Boss-Extrude2)(Spring_asm-6/SpringHold_part1-1), SolidBody 1(Cut-Extrude4)(Spring_asm-6/SpringHold_part2-1), SolidBody 1(SplitLine4)(Sub_MainAsm-1/UpperAsm_CRAFT-1/Antenne_ReleaseBoard-1), SolidBody 1(SplitLine1)(Sub_MainAsm-1/UpperAsm_CRAFT-1/XY_wall_Positive_Z-1)
Curve Data:N/A		
	Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m^2 Tensile strength: 3,753e+07 N/m^2 Elastic modulus: 3,0059e+08 N/m^2 Poisson's ratio: 0,38 Mass density: 1 240 kg/m^3	SolidBody 1(Cut-Extrude2)(Sub_MainAsm-1/UpperAsm_CRAFT-1/AntennaBoard_AntennaLocRoof-1), SolidBody 1(Cut-Extrude2)(Sub_MainAsm-1/UpperAsm_CRAFT-1/AntennaBoard_AntennaLocRoof-2)
Curve Data:N/A		



Loads and Fixtures

Fixture name	Fixture Image	Fixture Details															
Fixed-1		Entities: 2 edge(s), 2 face(s) Type: Fixed Geometry															
Resultant Forces																	
<table border="1"> <thead> <tr> <th>Components</th><th>X</th><th>Y</th><th>Z</th><th>Resultant</th></tr> </thead> <tbody> <tr> <td>Reaction force(N)</td><td>1,6486</td><td>1,18058</td><td>1,44412</td><td>2,4894</td></tr> <tr> <td>Reaction Moment(N.m)</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </tbody> </table>			Components	X	Y	Z	Resultant	Reaction force(N)	1,6486	1,18058	1,44412	2,4894	Reaction Moment(N.m)	0	0	0	0
Components	X	Y	Z	Resultant													
Reaction force(N)	1,6486	1,18058	1,44412	2,4894													
Reaction Moment(N.m)	0	0	0	0													
Roller/Slider-2		Entities: 4 face(s) Type: Roller/Slider															
Resultant Forces																	
<table border="1"> <thead> <tr> <th>Components</th><th>X</th><th>Y</th><th>Z</th><th>Resultant</th></tr> </thead> <tbody> <tr> <td>Reaction force(N)</td><td>8,45072e-05</td><td>3,42591e-05</td><td>3,14313</td><td>3,14313</td></tr> <tr> <td>Reaction Moment(N.m)</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </tbody> </table>			Components	X	Y	Z	Resultant	Reaction force(N)	8,45072e-05	3,42591e-05	3,14313	3,14313	Reaction Moment(N.m)	0	0	0	0
Components	X	Y	Z	Resultant													
Reaction force(N)	8,45072e-05	3,42591e-05	3,14313	3,14313													
Reaction Moment(N.m)	0	0	0	0													

Load name	Load Image	Load Details
Force-1		Entities: 4 face(s) Type: Apply normal force Value: 3,79 lbf



C.R.A.F.T

STB
13.05.2024

Connector Definitions

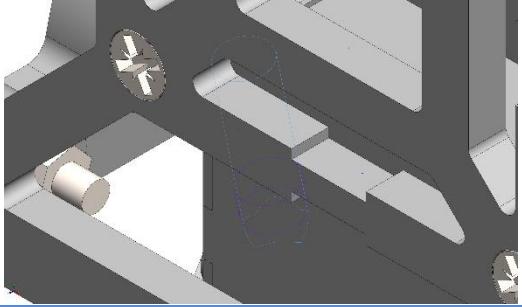
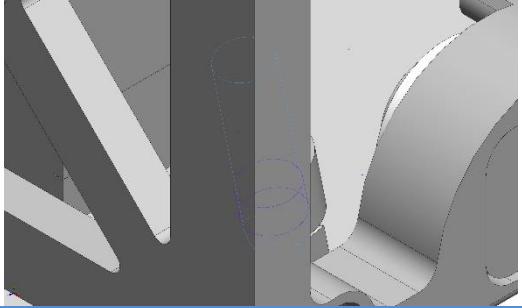
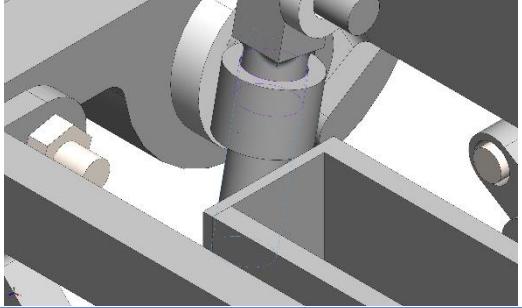
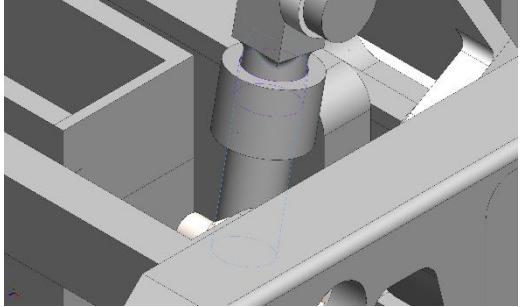
No Data



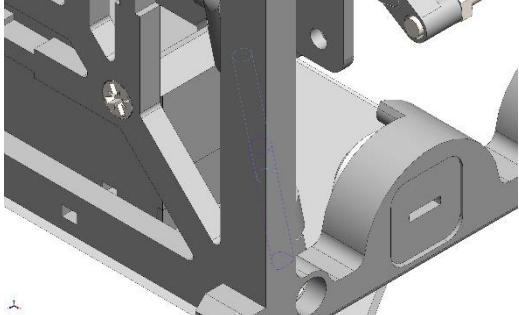
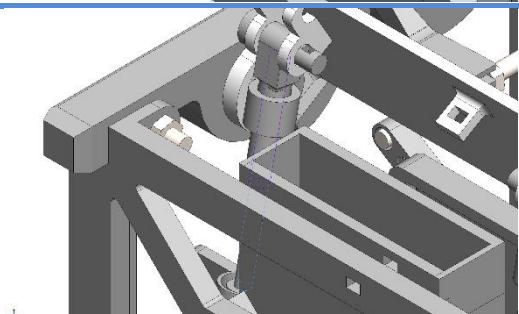
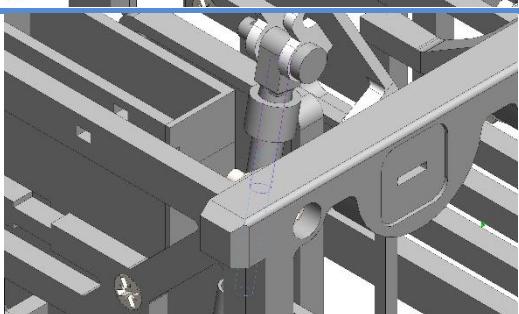
SOLIDWORKS Analyzed with SOLIDWORKS Simulation

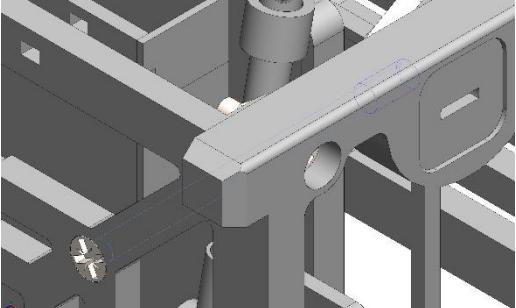
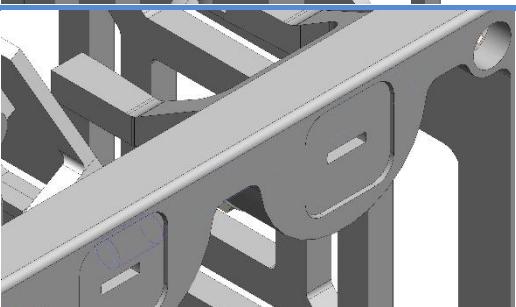
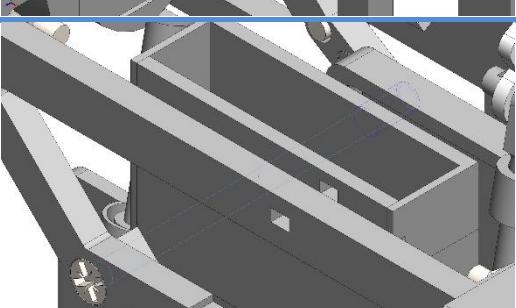
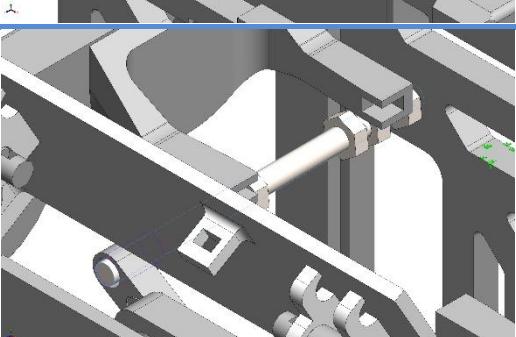
Simulation of Main_asm 27

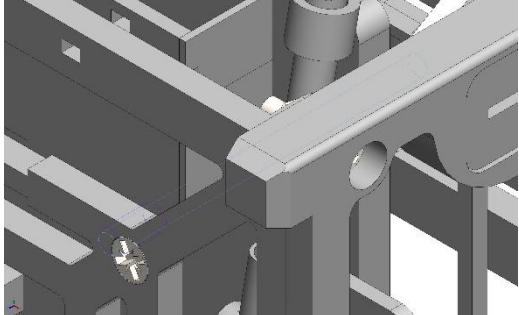
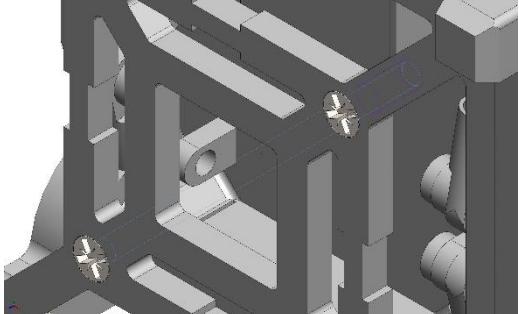
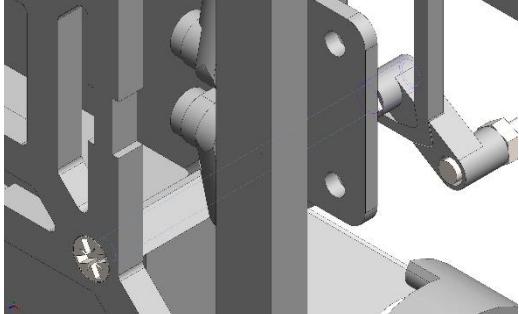
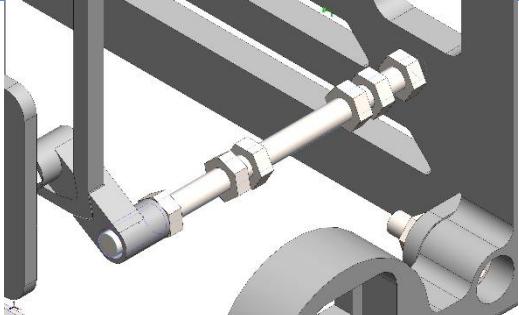
Interaction Information

Interaction	Interaction Image	Interaction Properties
Local Interaction-3890		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-3891		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-3892		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-3893		Type: Bonded interaction pair Entities: 2 face(s)



Local Interaction-4153		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4154		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4155		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4156		Type: Bonded interaction pair Entities: 2 face(s)

Local Interaction-4416		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4417		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4418		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4419		Type: Bonded interaction pair Entities: 2 face(s)

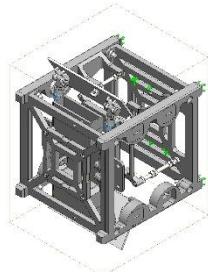
Local Interaction-4420		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4421		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4422		Type: Bonded interaction pair Entities: 2 face(s)
Local Interaction-4423		Type: Bonded interaction pair Entities: 2 face(s)



C.R.A.F.T

STB
13.05.2024

Global Interaction



Type: Bonded
Components: 1 component(s)
Options: Independent mesh



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 32

Mesh information

Mesh type	Solid Mesh
Mesher Used:	Blended curvature-based mesh
Jacobian points for High quality mesh	16 Points
Maximum element size	15,3777 mm
Minimum element size	0,768884 mm
Mesh Quality	High
Remesh failed parts independently	Off

Mesh information - Details

Total Nodes	285780
Total Elements	153686
Maximum Aspect Ratio	88,119
% of elements with Aspect Ratio < 3	92,4
Percentage of elements with Aspect Ratio > 10	0,269
Percentage of distorted elements	0
Time to complete mesh(hh:mm:ss):	00:00:39
Computer name:	S-K-1230-06

Mesh Quality Plots

Because of the amount of data required to open the simulation file, we couldn't open it to show the values of the Aspect Ratio and Jacobian Ratio. But, since less than 1% of the aspect ratio is higher than 10, we accept the analysis.

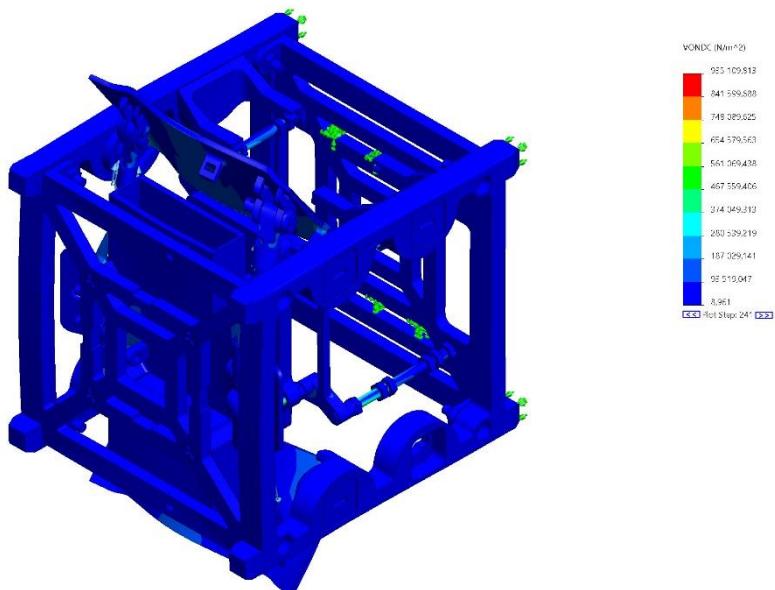
Sensor Details

No Data

Study Results

Name	Type	Min	Max
Stress1	VONDC: von Mises (Directional Components) at Step No: 241(Frequency = 1 000 Hz)	8,961N/m^2 Node: 4746	935 109,813N/m^2 Node: 130241

Mode name: Main_asm
Study name: Harmonical Dynamical Analysis Default
Plot type: Linear Dynamic Nodal stress Stress
Plot exp: 241 Frequency: 1 000 Hertz
Deformations scale: 764.259



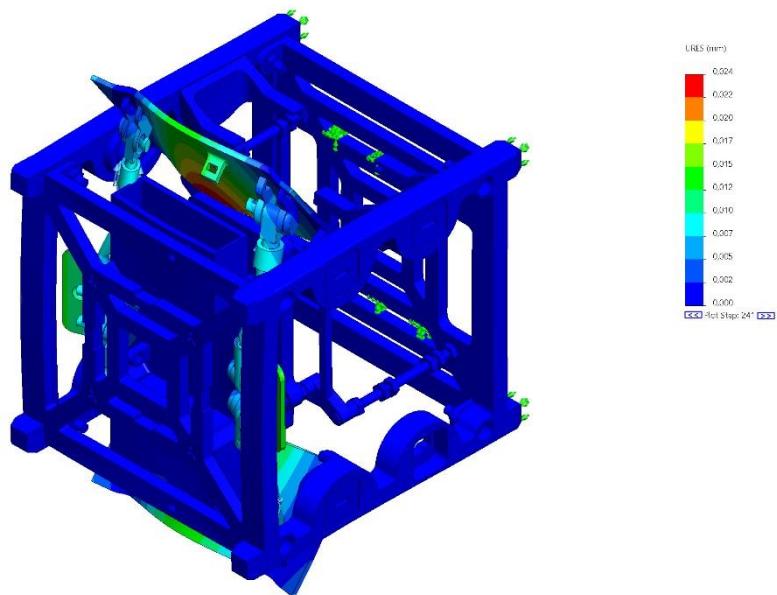
SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Harmonical Dynamical Analysis-Stress-Stress1



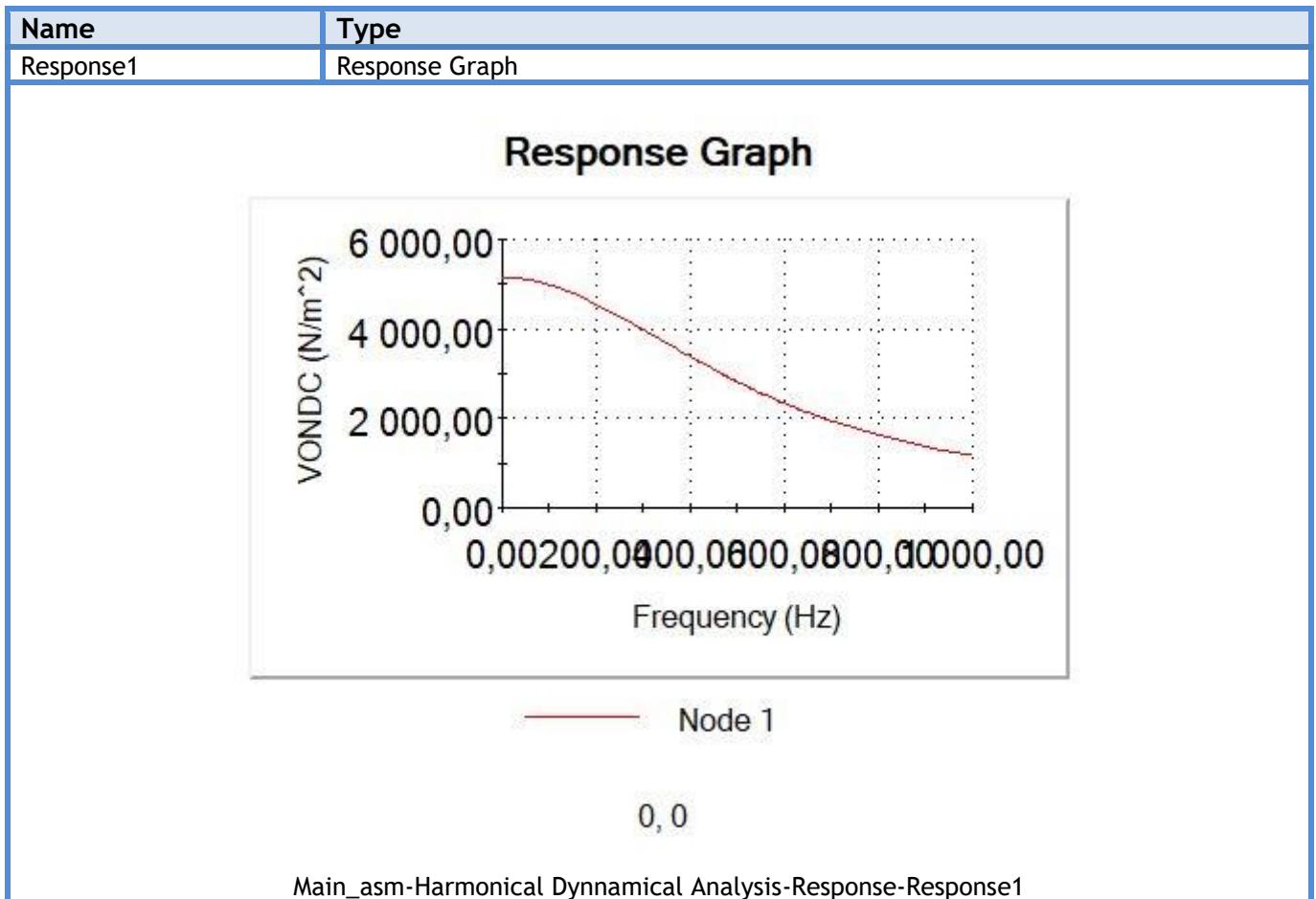
Name	Type	Min	Max
Displacement1	URES: Resultant Displacement at Step No: 241(Frequency = 1 000 Hz)	0,000mm Node: 95722	0,024mm Node: 27856

Model name: Main_asm
Study name: Harmonical Dynamical Analysis Default1
Plot type: Linear Dynamic Displacement Displacement1
Plot step: 241 Frequency: 1 000 Hertz
Deformation scale: 764.239



SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Harmonical Dynamical Analysis-Displacement-Displacement1





Mass Participation (Normalized)

Mode Number	Frequency(Hertz)	X direction	Y direction	Z direction
1	125,15	4,9951e-06	0,91367	0,003881
2	138,26	0,0050231	0,0024595	0,00026111
3	141,67	0,67883	0,020002	0,0003795
4	411,56	2,4064e-07	1,1746	0,37729
5	475,22	1,9191e-06	1 772,3	7 073
6	480,44	0,021038	1,3854	1,4243
7	491,99	2,7261e-05	0,12113	0,99227
8	497,39	0,0018391	10,444	4,0127
9	502,34	0,00012545	5 831,8	1 530,9
10	527,61	0,0081804	20,965	2,7169
11	548,38	1,0078e-05	278,84	150,84
12	590,6	0,0011845	2,0505	0,005495
13	605,12	0,00089406	4,58	0,022258
14	612,28	3,1257e-07	5,8251	2,7936
15	633,76	9,5521e-05	2,9724	0,13721
		Sum X = 0,71725	Sum Y = 7 933,3	Sum Z = 8 767,3





Conclusion

We decided to use vibration values 0 – 1000 Hz due to less information about what specific frequency we should use. This test ended with 241 steps, and with a vonMises value equal $935109.813 N/m^2 \approx 0.95 N/mm^2 = 0.95 MPa$ in the last step; the step where the stress is possibly low. The definition of low is referred to an amount of stress less than the yield strength of PETG and AISI 316 Annealed Stainless-Steel Bar. According to the given result and response graph at one specific node, the vonMises stress value has decreased due to the frequency increasing. We don't have the possibility to check the other nodes and their changes of stress with frequency due to the amount of memory required to open the SolidWorks simulation file, so our conclusions will therefore be that by increasing the vibration from 0 to 1000 Hz, the tension will be decreased in node 1, from 5 kPa to 1 kPa with the possibility of also decreasing or increasing stress values in the other nodes.

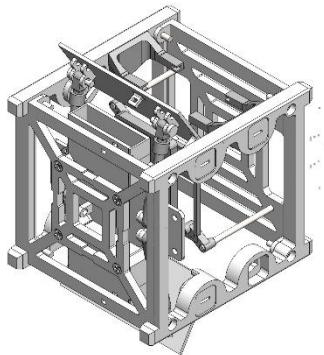
Note: the node from the response graph is not equal the node with the highest amount of stress in step 241 and was randomly selected to show the effect from an harmonical vibration on the stress at one randomly selected node.

V.5 Drop test

Under this section is a description and a result of a drop test with the *main_asm*.



C.R.A.F.T



3

Description

The analysis illustrates an impact on the assembly from a drop test after 125.4 microseconds when it has hit the ground. The simulation has an Aspect ratio with a small amount percent higher than 10, but with mostly under 3 percent. Because the simulation required a large amount of data, we have chosen to simplify it, and accept the small amount of percent higher than 10 but the location of the values higher than 10 percent has been checked.

Since the electronical assemblies don't include materials, they have been removed from the analysis. Also, some components such as bolts, nuts, and thread on those and other components have been removed. Some bolts have been included but they don't have threads. The model has only a design representing balloon mounting and not parachuting. The model can be re-designed, but it's not our object at this moment to do so. Therefore, it's interesting to see what's happening with our product.

Simulation of assembly: main_asm

Date: Thursday 9. May 2024

Designer: STB

Study name: Drop Test

Analysis type: Drop Test

Table of Contents

Description	1
Assumptions & model information	2
Study Properties.....	11
Units	11
Material Properties	12
Interaction Information	16
Mesh information.....	16
Study Results	17
Conclusion.....	21



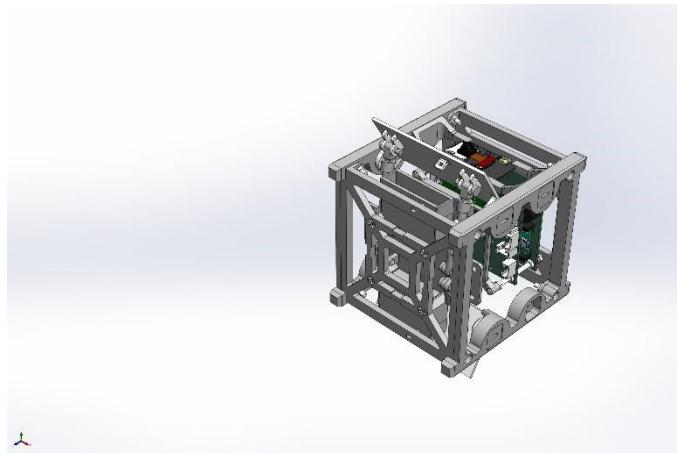
SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

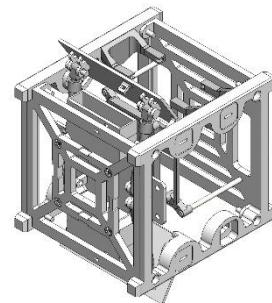
Simulation of Main_asm 1

Assumptions & model information

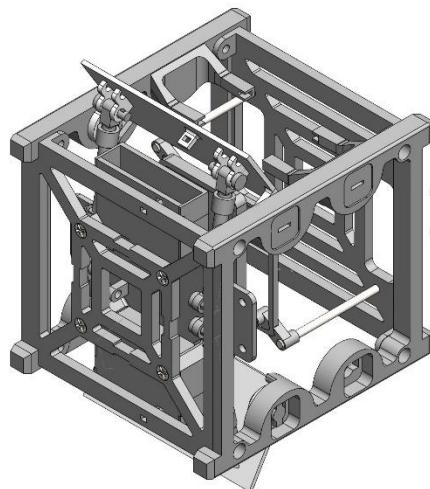
We assume center of gravity is in the chosen area in the analysis and there is none interacting forces or other circumstances disturbing a drop from the sky against the ground with a height of 10 meter. The model is analyzed to represent a circumstance with none-electronical components and with an environment where none disturbing objects occurs. The model will fall straight down with a gravitational coefficient equal 9.81 m/s^2 and hit a flat ground parallel to the XY-plane from the assembly. We have not chosen the coefficient of friction value, so it's assumed to be equal zero on the ground. Since the model is a solid tight fitted with no-free components, none-local interactions are decided and only the bonded global interaction is used. The model has components with threads and small sizes; the threads are removed, and some mechanical components (such as bolts, nuts, and springs) are not included in the analysis. The model represents the real sizes of the product, with PETG as the material for all the 3D printing parts. The total mass of the model analyzed is higher (we don't have the specific value at this moment) than the real product because we will not use 100% infill in 3D printing. This results with a higher gravitational force, $G = mg$, where m is the mass and g are the gravitational coefficient (a constant value).



Original Model



Model Analyzed



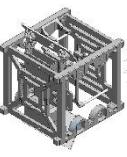
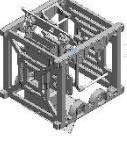
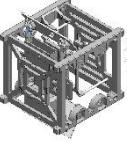
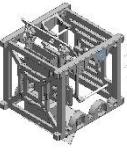
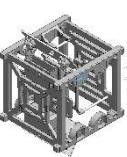
Model name: Main_asm
Current Configuration: Default

Solid Bodies

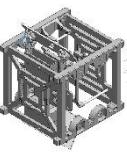
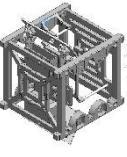
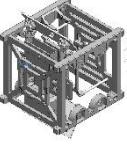
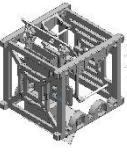
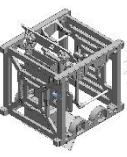
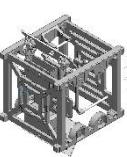
Document Name and Reference	Treated As	Volumetric Properties	Document Path/Date Modified
Split Line3	Solid Body	Mass:0,00290598 kg Volume:2,12136e-06 m^3 Density:1 369,87 kg/m^3 Weight:0,0284786 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Camera_Holder.SLDPR T May 9 09:34:21 2024
Split Line3	Solid Body	Mass:0,00290598 kg Volume:2,12136e-06 m^3 Density:1 369,87 kg/m^3 Weight:0,0284786 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Camera_Holder.SLDPR T May 9 09:34:21 2024
Cut-Extrude1	Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\HingeLockPart.SLDPR T May 9 09:34:22 2024





Cut-Extrude1 	Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\HingeLockPart.SLDprt May 9 09:34:22 2024
Cut-Extrude1 	Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\HingeLockPart.SLDprt May 9 09:34:22 2024
Cut-Extrude1 	Solid Body	Mass:0,000567144 kg Volume:4,13973e-07 m^3 Density:1 370 kg/m^3 Weight:0,00555801 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\HingeLockPart.SLDprt May 9 09:34:22 2024
Fillet1 	Solid Body	Mass:0,00107758 kg Volume:7,86552e-07 m^3 Density:1 370 kg/m^3 Weight:0,0105602 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Lock1.SLDprt May 9 09:34:22 2024
Fillet1 	Solid Body	Mass:0,00107758 kg Volume:7,86552e-07 m^3 Density:1 370 kg/m^3 Weight:0,0105602 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Lock1.SLDprt May 9 09:34:22 2024
Fillet1 	Solid Body	Mass:0,00107758 kg Volume:7,86552e-07 m^3 Density:1 370 kg/m^3 Weight:0,0105602 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Lock1.SLDprt May 9 09:34:22 2024



Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Lock1.SLDprt May 9 09:34:22 2024
Fillet1 	Solid Body	Mass: 0,00107758 kg Volume: 7,86552e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,0105602 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Lock1.SLDprt May 9 09:34:22 2024
Split Line1 	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Pin_Sealing.SLDprt May 9 09:34:22 2024
Split Line1 	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Pin_Sealing.SLDprt May 9 09:34:22 2024
Split Line1 	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Pin_Sealing.SLDprt May 9 09:34:22 2024
Split Line1 	Solid Body	Mass: 0,000218599 kg Volume: 1,59561e-07 m^3 Density: 1 370 kg/m^3 Weight: 0,00214227 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Pin_Sealing.SLDprt May 9 09:34:22 2024



Split Lie1	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Pin_Sealing.SLDPR May 9 09:34:22 2024
Split Line1	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Pin_Sealing.SLDPR May 9 09:34:22 2024
Split Line1	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Pin_Sealing.SLDPR May 9 09:34:22 2024
Split Line1	Solid Body	Mass:0,000218599 kg Volume:1,59561e-07 m^3 Density:1 370 kg/m^3 Weight:0,00214227 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Pin_Sealing.SLDPR May 9 09:34:22 2024
Split Line3	Solid Body	Mass:0,00973089 kg Volume:7,10284e-06 m^3 Density:1 370 kg/m^3 Weight:0,0953628 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\SealingAntenna.SLDPR T May 9 09:34:23 2024
Split Line3	Solid Body	Mass:0,00973089 kg Volume:7,10284e-06 m^3 Density:1 370 kg/m^3 Weight:0,0953628 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\SealingAntenna.SLDPR T May 9 09:34:23 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m^3	C:\Users\235474\Downloads\BackUp_08-55-09-





		Density:1 370 kg/m ³ Weight:0,00365629 N	05\InternalStampPin.SLDPart May 9 09:34:22 2024
Fillet1	Solid Body	Mass:0,000428203 kg Volume:3,12546e-07 m ³ Density:1 370,05 kg/m ³ Weight:0,00419639 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\ShockPart.SLDPart May 9 09:34:23 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m ³ Density:1 370 kg/m ³ Weight:0,0016105 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\SpringHold_part1.SLDPart May 9 09:34:23 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m ³ Density:1 370 kg/m ³ Weight:0,00365629 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\InternalStampPin.SLDPart May 9 09:34:22 2024
Fillet1	Solid Body	Mass:0,000428204 kg Volume:3,12546e-07 m ³ Density:1 370,05 kg/m ³ Weight:0,0041964 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\ShockPart.SLDPart May 9 09:34:23 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m ³ Density:1 370 kg/m ³ Weight:0,0016105 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\SpringHold_part1.SLDPart May 9 09:34:23 2024
Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m ³ Density:1 370 kg/m ³ Weight:0,00365629 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\InternalStampPin.SLDPart May 9 09:34:22 2024
Fillet1	Solid Body	Mass:0,000428203 kg Volume:3,12546e-07 m ³ Density:1 370,05 kg/m ³ Weight:0,00419639 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\ShockPart.SLDPart May 9 09:34:23 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m ³ Density:1 370 kg/m ³ Weight:0,0016105 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\SpringHold_part1.SLDPart May 9 09:34:23 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m ³ Density:1 370 kg/m ³ Weight:0,00442236 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\SpringHold_part2.SLDPart May 9 09:34:23 2024





Outer_Sylinder	Solid Body	Mass:0,000373091 kg Volume:2,72329e-07 m^3 Density:1 370 kg/m^3 Weight:0,00365629 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\InternalStampPin.SLD PRT May 9 09:34:22 2024
Fillet1	Solid Body	Mass:0,000428204 kg Volume:3,12546e-07 m^3 Density:1 370,05 kg/m^3 Weight:0,0041964 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\ShockPart.SLD PRT May 9 09:34:23 2024
Boss-Extrude2	Solid Body	Mass:0,000164337 kg Volume:1,19954e-07 m^3 Density:1 370 kg/m^3 Weight:0,0016105 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\SpringHold_part1.SLD PRT May 9 09:34:23 2024
Cut-Extrude4	Solid Body	Mass:0,000451261 kg Volume:3,29388e-07 m^3 Density:1 370 kg/m^3 Weight:0,00442236 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\SpringHold_part2.SLD PRT May 9 09:34:23 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Hook_Parts.SLD PRT May 9 09:34:22 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Hook_Parts.SLD PRT May 9 09:34:22 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Hook_Parts.SLD PRT May 9 09:34:22 2024
Fillet1	Solid Body	Mass:0,000339322 kg Volume:2,4768e-07 m^3 Density:1 370 kg/m^3 Weight:0,00332535 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Hook_Parts.SLD PRT May 9 09:34:22 2024
Fillet1	Solid Body	Mass:0,00132768 kg Volume:9,69113e-07 m^3 Density:1 370 kg/m^3 Weight:0,0130113 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Plate_Lock.SLD PRT May 9 09:34:22 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3	C:\Users\235474\Downloads\BackUp_08-55-09-

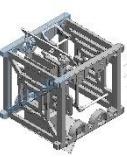




		Weight:0,0317073 N	05\M3x50_PZ1 head.SLDPR May 9 09:34:22 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\M3x50_PZ1 head.SLDPR May 9 09:34:22 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\M3x50_PZ1 head.SLDPR May 9 09:34:22 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\M3x50_PZ1 head.SLDPR May 9 09:34:22 2024
Fillet2	Solid Body	Mass:0,0297987 kg Volume:2,17509e-05 m^3 Density:1 370 kg/m^3 Weight:0,292027 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\XY_wall_Negative_Z.S LDPR May 9 09:34:24 2024
Cut-Extrude2	Solid Body	Mass:0,00269269 kg Volume:2,17153e-06 m^3 Density:1 240 kg/m^3 Weight:0,0263884 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\AntennaBoard_Antenn aLockRoof.SLDPR May 9 09:34:21 2024
Cut-Extrude2	Solid Body	Mass:0,00297499 kg Volume:2,17153e-06 m^3 Density:1 370 kg/m^3 Weight:0,0291549 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\AntennaBoard_Antenn aLockRoof.SLDPR May 9 09:34:21 2024
Split Line4	Solid Body	Mass:0,0206986 kg Volume:1,51084e-05 m^3 Density:1 370 kg/m^3 Weight:0,202846 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\Antenne_ReleaseBoar d.SLDPR May 9 09:34:21 2024
Split Line1	Solid Body	Mass:0,00625083 kg Volume:4,56265e-06 m^3 Density:1 370 kg/m^3 Weight:0,0612582 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\BatterySpringPlate.S LDPR May 9 09:34:21 2024
M3_Thread	Solid Body	Mass:0,00323544 kg Volume:4,0443e-07 m^3 Density:8 000 kg/m^3 Weight:0,0317073 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\M3x50_PZ1 head.SLDPR May 9 09:34:22 2024





M3_Thread	Solid Body	Mass: 0,00323544 kg Volume: 4,0443e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,0317073 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\M3x50_PZ1_head.SLDprt May 9 09:34:22 2024
M3_Thread	Solid Body	Mass: 0,00323544 kg Volume: 4,0443e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,0317073 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\M3x50_PZ1_head.SLDprt May 9 09:34:22 2024
M3_Thread	Solid Body	Mass: 0,00323544 kg Volume: 4,0443e-07 m^3 Density: 8 000 kg/m^3 Weight: 0,0317073 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\M3x50_PZ1_head.SLDprt May 9 09:34:22 2024
Split Line1	Solid Body	Mass: 0,0222911 kg Volume: 1,62708e-05 m^3 Density: 1 370 kg/m^3 Weight: 0,218452 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\XY_wall_Positive_Z.SLDprt May 9 16:37:00 2024
Fillet5 	Solid Body	Mass: 0,0318527 kg Volume: 2,32501e-05 m^3 Density: 1 370 kg/m^3 Weight: 0,312156 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\ZY_Wall.SLDprt May 9 09:34:24 2024
Fillet5 	Solid Body	Mass: 0,0318527 kg Volume: 2,32501e-05 m^3 Density: 1 370 kg/m^3 Weight: 0,312156 N	C:\Users\235474\Downloads\BackUp_08-55-09-05\ZY_Wall.SLDprt May 9 09:34:24 2024





Study Properties

Study name	Drop Test
Analysis type	Drop Test
Mesh type	Solid Mesh
Large displacement	On
Result folder	SOLIDWORKS document (C:\Users\235474\Downloads\BackUp_08-55-09-05)

Setup Information

Type	Drop height
Drop Height from Centroid	10 m
Gravity	9,81 m/s ²
Gravity Reference	Front Plane
Coefficient of friction	0
Target Stiffness	Rigid target
Critical Damping Ratio	0

Result Options

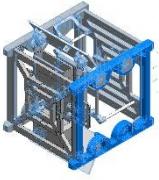
Solution Time After Impact	125,4 microsec
Save Results Starting From	0 microsec
No. of Plots	25
No. of Graph Steps Per Plot	20
Number of vertex	0

Units

Unit system:	SI (MKS)
Length/Displacement	mm
Temperature	Kelvin
Angular velocity	Rad/sec
Pressure/Stress	N/m ²



Material Properties

Model Reference	Properties	Components
	<p>Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m² Tensile strength: 3,753e+07 N/m² Elastic modulus: 3,00159e+09 N/m² Poisson's ratio: 0,38 Mass density: 1 370 kg/m³</p>	SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-1), SolidBody 1(Split Line3)(Camera_asm-1/Camera_Holder-2), SolidBody 1(Cut-Extrude1)(HingeLockPart-6), SolidBody 1(Cut-Extrude1)(HingeLockPart-7), SolidBody 1(Cut-Extrude1)(HingeLockPart-8), SolidBody 1(Cut-Extrude1)(HingeLockPart-9), SolidBody 1(Fillet1)(Lock1-11), SolidBody 1(Fillet1)(Lock1-6), SolidBody 1(Fillet1)(Lock1-7), SolidBody 1(Fillet1)(Lock1-8), SolidBody 1(Fillet1)(Lock1-9), SolidBody 1(Split Line1)(Pin_Sealing-1), SolidBody 1(Split Line1)(Pin_Sealing-2), SolidBody 1(Split Line1)(Pin_Sealing-3), SolidBody 1(Split Line1)(Pin_Sealing-4), SolidBody 1(Split Line1)(Pin_Sealing-5), SolidBody 1(Split Line1)(Pin_Sealing-6), SolidBody 1(Split Line1)(Pin_Sealing-7), SolidBody 1(Split Line1)(Pin_Sealing-8), SolidBody 1(Split Line3)(SealingAntenna-1), SolidBody 1(Split Line3)(SealingAntenna-2), SolidBody 1(Fillet1)(Sub_MainAsm-1/Hook_Asm-1/Hook_Parts-1),



	SolidBody 1(Fillet1)(Sub_MainAsm- 1/Hook_Asm-1/Hook_Parts- 2), SolidBody 1(Fillet1)(Sub_MainAsm- 1/Hook_Asm-1/Hook_Parts- 3), SolidBody 1(Fillet1)(Sub_MainAsm- 1/Hook_Asm-1/Hook_Parts- 4), SolidBody 1(Fillet1)(Sub_MainAsm- 1/Hook_Asm-1/Plate_Lock- 1), SolidBody 1(Fillet2)(Sub_MainAsm- 1/LowerAsm_CRAFT- 2/XY_wall_Negative_Z-1), SolidBody 1(Cut- Extrude2)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/AntennaBoard_AntennaLoc- kRoof-2), SolidBody 1(Split Line1)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/BatterySpringPlate-1), SolidBody 1(Fillet5)(ZY_Wall- 1), SolidBody 1(Fillet5)(ZY_Wall- 2)
--	---

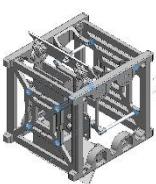
Curve Data:N/A

	Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m^2 Tensile strength: 3,753e+07 N/m^2 Elastic modulus: 3,00159e+09 N/m^2 Poisson's ratio: 0,38 Mass density: 1 370 kg/m^3 Thermal expansion coefficient: 3,8 /Kelvin	SolidBody 1(Outer_Sylinder)(Spring_as- m-1/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm- 1/ShockPart-1), SolidBody 1(Boss- Extrude2)(Spring_asm- 1/SpringHold_part1-1), SolidBody 1(Outer_Sylinder)(Spring_as- m-2/InternalStampPin-2), SolidBody 1(Fillet1)(Spring_asm- 2/ShockPart-1),
--	--	--



SolidBody 1(Boss-Extrude2)(Spring_asm-2/SpringHold_part1-1),
 SolidBody 1(Outer_Sylinder)(Spring_asm-4/InternalStampPin-2),
 SolidBody 1(Fillet1)(Spring_asm-4/ShockPart-1),
 SolidBody 1(Boss-Extrude2)(Spring_asm-4/SpringHold_part1-1),
 SolidBody 1(Cut-Extrude4)(Spring_asm-4/SpringHold_part2-1),
 SolidBody 1(Outer_Sylinder)(Spring_asm-6/InternalStampPin-2),
 SolidBody 1(Fillet1)(Spring_asm-6/ShockPart-1),
 SolidBody 1(Boss-Extrude2)(Spring_asm-6/SpringHold_part1-1),
 SolidBody 1(Cut-Extrude4)(Spring_asm-6/SpringHold_part2-1),
 SolidBody 1(SplitLine4)(Sub_MainAsm-1/UpperAsm_CRAFT-1/Antenne_ReleaseBoard-1),
 SolidBody 1(SplitLine1)(Sub_MainAsm-1/UpperAsm_CRAFT-1/XY_wall_Positive_Z-1)

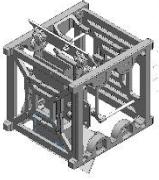
Curve Data:N/A



Name:	AISI 316 Stainless Steel Sheet (SS)
Model type:	Linear Elastic Isotropic
Default failure criterion:	Unknown
Yield strength:	1,72369e+08 N/m ²
Tensile strength:	5,8e+08 N/m ²
Elastic modulus:	1,93e+11 N/m ²
Poisson's ratio:	0,27
Mass density:	8 000 kg/m ³
Thermal expansion coefficient:	1,6e-05 /Kelvin

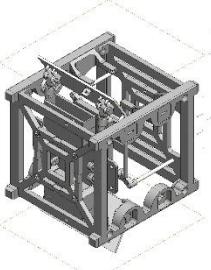
SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-1),
 SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-2),
 SolidBody 1(M3_Thread)(Sub_MainAsm-1/LowerAsm_CRAFT-2/M3x50_PZ1 head-3),



		SolidBody 1(M3_Thread)(Sub_MainAsm- 1/LowerAsm_CRAFT- 2/M3x50_PZ1 head-4), SolidBody 1(M3_Thread)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-1), SolidBody 1(M3_Thread)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-2), SolidBody 1(M3_Thread)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-3), SolidBody 1(M3_Thread)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/M3x50_PZ1 head-4)
Curve Data:N/A		
	Name: PETG - CRAFT Model type: Linear Elastic Isotropic Default failure criterion: Unknown Yield strength: 1,158e+07 N/m^2 Tensile strength: 3,753e+07 N/m^2 Elastic modulus: 3,0059e+08 N/m^2 Poisson's ratio: 0,38 Mass density: 1 240 kg/m^3	SolidBody 1(Cut-Extrude2)(Sub_MainAsm- 1/UpperAsm_CRAFT- 1/AntennaBoard_AntennaLoc kRoof-1)
Curve Data:N/A		



Interaction Information

Interaction	Interaction Image	Interaction Properties
Global Interaction		<p>Type: Bonded Components: 1 component(s) Options: Continuous mesh</p>

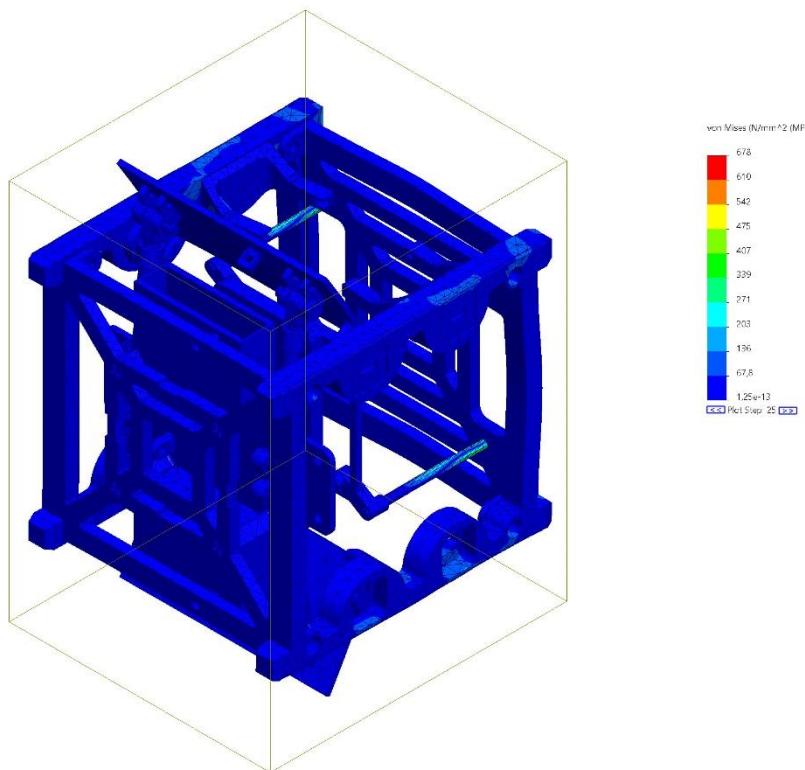
Mesh information

Mesh type	Solid Mesh
Mesher Used:	Curvature-based mesh
Jacobian points for High quality mesh	16 Points
Maximum element size	15,0678 mm
Minimum element size	3,01355 mm
Mesh Quality	High

Study Results

Name	Type	Min	Max
Stress1	VON: von Mises Stress	1,25e-13N/mm ² (MPa) Node: 25052	678N/mm ² (MPa) Node: 36945

Model name: Main_asm
Study name: Drop Test (Default)
Post type: Stress1
Post step 25 - time: 125.4 Microseconds
Deformation scale: 1



SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Drop Test-Stress-Stress1

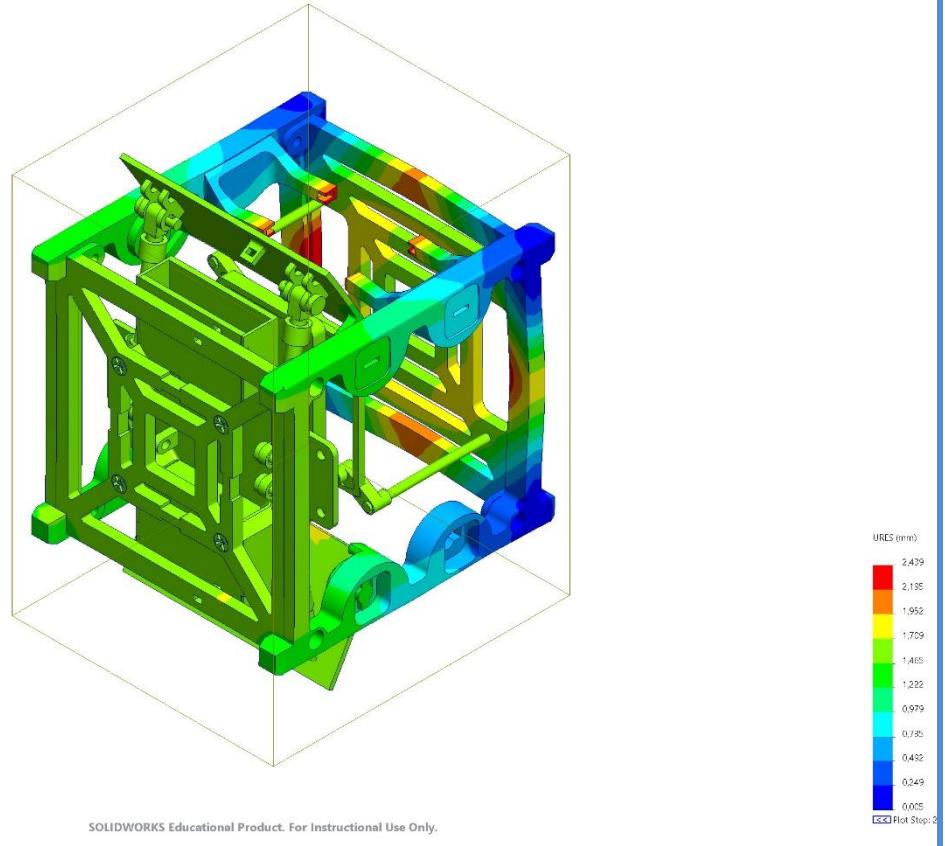


C.R.A.F.T

STB
09.05.2024

Name	Type	Min	Max
Displacement1	URES: Resultant Displacement	0,005mm Node: 86330	2,439mm Node: 45673

Model name: Main_asm
Study name: Drop (test Default)
Plot type: Displacement
Plot step: 24 time: 120.283 Microseconds
Deformation scale: 1



SOLIDWORKS Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 18

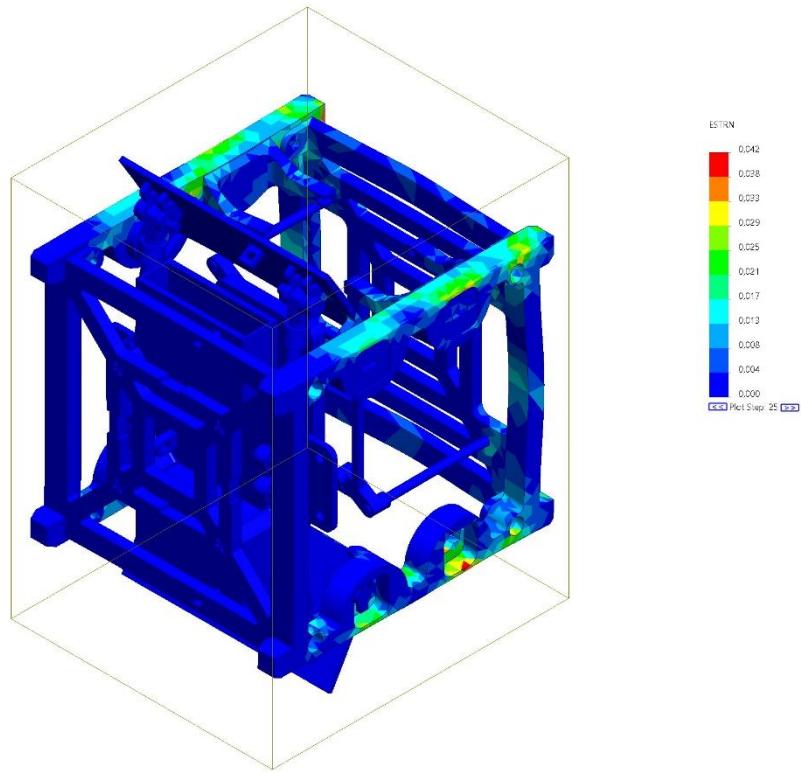


C.R.A.F.T

STB
09.05.2024

Name	Type	Min	Max
Strain1	ESTRN: Equivalent Strain	0,000 Element: 17293	0,042 Element: 54021

Model name: Main_asm
Study name: Drop (test-Default)
Plot type: Strain1
Plot step: 25 (time: 125.4 Microseconds)
Deformation scale: 1



SOLIDWORKS Educational Product. For Instructional Use Only.

Main_asm-Drop Test-Strain-Strain1



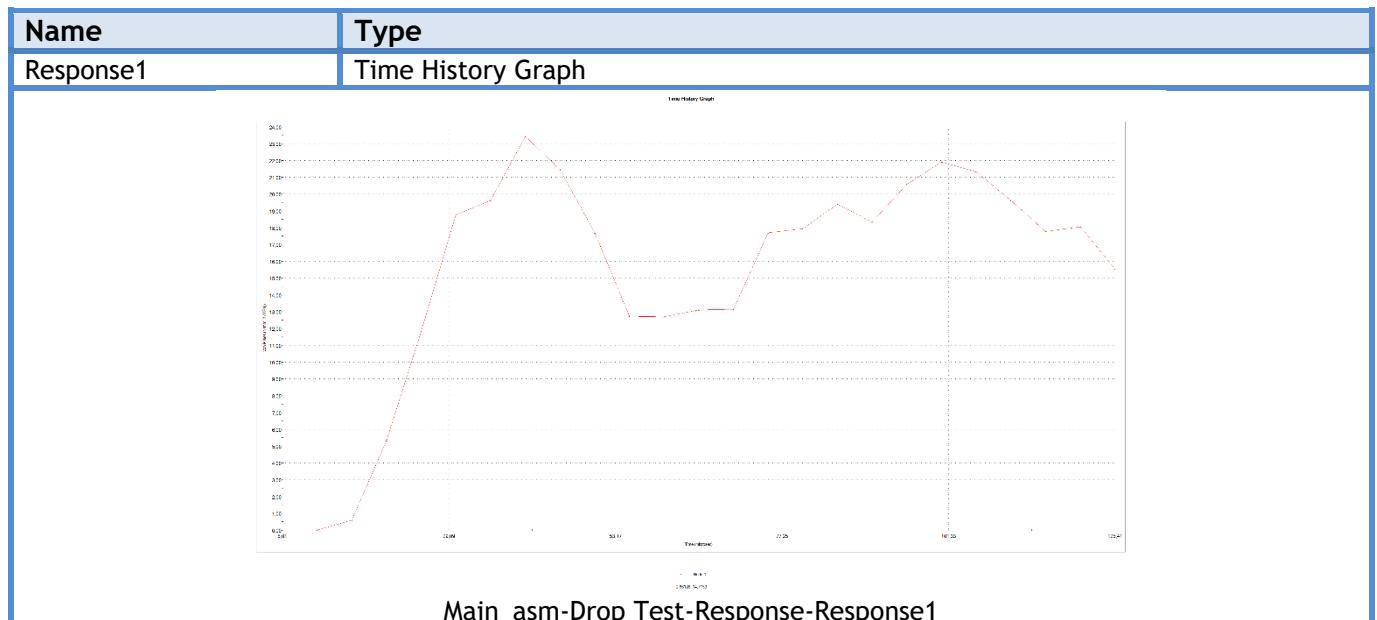
SOLIDWORKS Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm 19



C.R.A.F.T

STB
09.05.2024



SOLIDWORKS

Analyzed with SOLIDWORKS Simulation

Simulation of Main_asm

20

Conclusion

The sheet vonMises shows the results with the highest stress (in plot 25) mostly on the bolts used for mounting the electronical circuit boards and antenna release board. The bolts represent the classification A4 (material: 316 Stainless steel) and has a tensile strength value higher than the stress on the bolts. But the yield strength is less than the stress on the bolts, therefore, they will get a permanent deformation making the product work/strain-hardened (no-heat related hardening) but also less ductile the next time stress occurs on the bolts. By observing the resultant displacement sheet (in plot 25), the highest displacement will not occur on the areas first hitting the ground, but by the shock from hitting the ground, the displacement close to the center will be high. As it shows, the displacement will increase as it heading to the center of the lower section on the ZY-part and be highest at the center. The area where it's highest on the ZY-part spreads on the XY-part and decrease as it heads to the center of the ZY-part. The diagram shows the displacement will be small in the top parts not hitting the ground.

The time-graph gives a good description about the vonMises stress at a *node* from plot 1 to 25 and the changes from it first hits the ground into a chosen time after it has hit ground. Since we have a problem with the analysis file, we can't locate the specific location of where the stress is highest, and we can't make a factor of safety sheet with this analysis. But from observing the vonMises sheet, the stress will be high overall in the product and much higher than the tensile strength of PETG. We will therefore conclude with a permanent destruction of the frame when a 10-meter drop occurs, and the model hits the ground.

Update: When other main_asm analysis was made, local interaction had to be made, but due to the amount of data required for the analysis, a new updated analysis isn't possible at this moment. The results are therefore related to the data gained from the analysis without local interactions in the spring assembly.

W Vacuum Test

It was a simplified vacuum test. A report regarding the vacuum test, including description, risk, test data and a conclusion is included under this section.



Vacuum test

Simplified vacuum test on the C.R.A.F.T-Sat with electronical components measuring the pressure.

Writer: STB

Test plan description:

The plan is to test whether the components measure the vacuum pressure compared to the real pressure from a manometer. Also, to get a more realistic test, the components are mounted on the C.R.A.F.T-Sat. We will give an estimation of what pressure the object will be exposed to, define possible risk on the testing objects during test, estimated pressure (vacuum pressure planned for the tests), pressure from the manometer, pressure from the sensor BMP280, test time and the results from the tests.

The test is done in the composite lab at the University of Southeast Norway campus Kongsberg with acceptance of collaboration with a lab manager from the vocational school.

Possible risks:

- The manometer is defect before or during the test.
- Errors/bug with the code reading the pressure values.
- The BMP280 is defect before, during or after test.

Test data:

Decrease pressure from 1100 hPa to 300 hPa.

	Test No.	Estimated time	Estimated pressure	Pressure value per min (manometer)	Pressure value per min (BMP280)
Decrease pressure	1	5 min	1100 hPa	I. -0.2 bar II. -0.2 bar III. -0.2 bar IV. -0.2 bar V. -0.2 bar	Difficult to read the pressure at the specific time.
	2	5 min	940 hPa	I. -0.9 bar II. -0.9 bar III. -0.9 bar IV. -0.9 bar V. -0.9 bar	Difficult to read the pressure at the specific time.
	3	5 min	780 hPa	I. -0.8 bar II. -0.8 bar III. -0.8 bar IV. -0.8 bar V. -0.8 bar	Difficult to read the pressure at the specific time.
	4	5 min	620 hPa	I. -0.6 bar II. -0.6 bar III. -0.6 bar IV. -0.6 bar V. -0.6 bar	Difficult to read the pressure at the specific time.
	5	5 min	460 hPa	I. -0.46 bar II. -0.46 bar III. -0.46 bar IV. -0.46 bar V. -0.46 bar	Difficult to read the pressure at the specific time.
	6	5 min	300 hPa	I. -0.3 bar II. -0.3 bar III. -0.3 bar IV. -0.3 bar V. -0.3 bar	Difficult to read the pressure at the specific time.

Comment: 30 min test

Increase pressure from 300 hPa to 1100 hPa (Start at 460 hPa since decrease test ended at 300 hPa).

	Test No.	Estimated time	Estimated pressure	Pressure value per min (manometer)	Pressure value per min (BMP280)
Increase pressure	1	6 min	460 hPa	I. -0.46 bar II. -0.46 bar III. -0.46 bar IV. -0.46 bar V. -0.46 bar VI. -0.46 bar	Difficult to read the pressure at the specific time.
	2	6 min	620 hPa	I. -0.6 bar II. -0.6 bar III. -0.6 bar IV. -0.6 bar V. -0.6 bar VI. -0.6 bar	Difficult to read the pressure at the specific time.
	3	6 min	780 hPa	I. -0.8 bar II. -0.8 bar III. -0.8 bar IV. -0.8 bar V. -0.8 bar VI. -0.8 bar	Difficult to read the pressure at the specific time.
	4	6 min	940 hPa	I. -0.9 bar II. -0.9 bar III. -0.9 bar IV. -0.9 bar V. -0.9 bar VI. -0.9 bar	Difficult to read the pressure at the specific time.
	5	6 min	1100 hPa	I. -0.2 bar II. -0.2 bar III. -0.2 bar IV. -0.2 bar V. -0.2 bar VI. -0.2 bar	Difficult to read the pressure at the specific time.

Comment: 30 min test



Pressure data from the *BMP280*. The data values are plotted with the time interval per minute.

	Pressure	
1	[hPa]	37
2	1004.70	38
3	1004.74	39
4	1004.74	40
5	884.21	41
6	786.70	42
7	754.77	43
8	754.41	44
9	790.10	45
10	791.01	46
11	792.19	47
12	793.45	48
13	258.75	49
14	78.52	50
15	79.93	51
16	81.27	52
17	82.56	53
18	83.86	54
19	201.93	55
20	201.79	56
21	202.90	57
22	204.08	58
23	205.18	59
24	206.27	60
25	390.39	61
26	390.76	62
27	391.71	63
28	392.64	64
29	393.59	65
30	506.34	66
31	541.71	67
32	542.40	68
33	543.19	69
34	543.96	70
35	544.78	71
36	692.93	72
		73
		692.63
		693.16
		693.68
		694.26
		542.71
		557.54
		558.43
		559.29
		560.07
		560.88
		561.67
		394.11
		395.35
		396.35
		397.38
		398.39
		399.39
		195.48
		199.25
		200.50
		201.71
		202.87
		204.02
		205.13
		88.12
		89.40
		90.67
		91.88
		93.12
		94.33
		727.06
		784.62
		784.32
		784.51
		784.85
		785.23
		785.54



Test conclusion

The *BMP280* suddenly documented a pressure value less than its lowest recommended value of 300 hPa. Therefore, we consider the data to be wrong due to the low values. Since the test suddenly resulted with misunderstanding in the values gained from the manometer used which started at -1 bar, the test ended as a failure in data comparing and testing. A new and better prepared test should be done, but due to the amount of time before assignment deadline it's not enough time to take a new equal test.

X C.R.A.F.T-Sat mounting description used during visit: Vestsiden secondary school

The manual is designed to give a group of secondary school students a good and simple information about of how to assembly the C.R.A.F.T-Sat frame and antenna release mechanism after the parts has been produced with 3D printing. It's recommended to follow this manual, step-by-step, to assembly it correct. If the user isn't following the manual step-by-step the C.R.A.F.T-group takes no responsibility if the product get's any form of fracture or is totally destroyed.

To disassembly the C.R.A.F.T-Sat follow the mounting manual X.2, starting with **Task (4)**. and follow it backwards.

X.1 Items needed to assembly C.R.A.F.T-Sat

- (a) Pozidriv screwdriver (PZ1)
- (b) Slotted screwdriver (V4)
- (c) Wrench; size 5.5 mm
- (d) Super glue (all super glue is accepted at this state).
- (I) 8x M3 locking nut
- (II) 8x M3x10 bolt (PZ1 head)
- (III) 4x M3 nut
- (IV) 4x M3x40 bolt (V4 head)

3D printed component and amount print per component:

- 2x Side wall - ZY Plane, fig 569.
- 1x Negative Z - XY Side wall, fig 570.
- 1x Positive Z - XY Side wall, fig 571.
- 1x Release board, fig 572.
- 4x Lock plate - antenna board, fig 573.

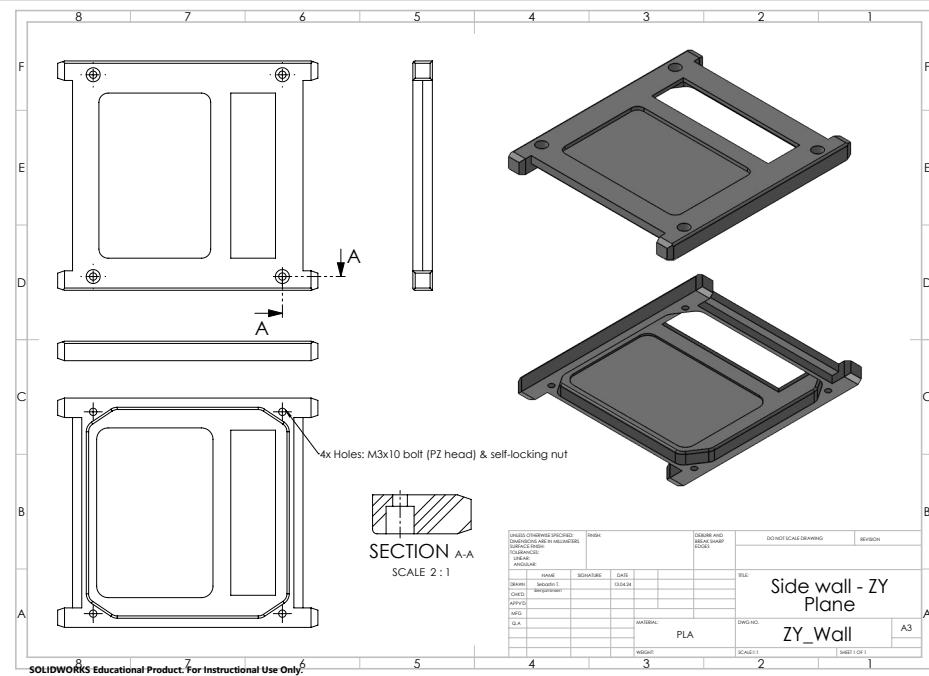


Figure 569: Side wall - ZY Plane | STB

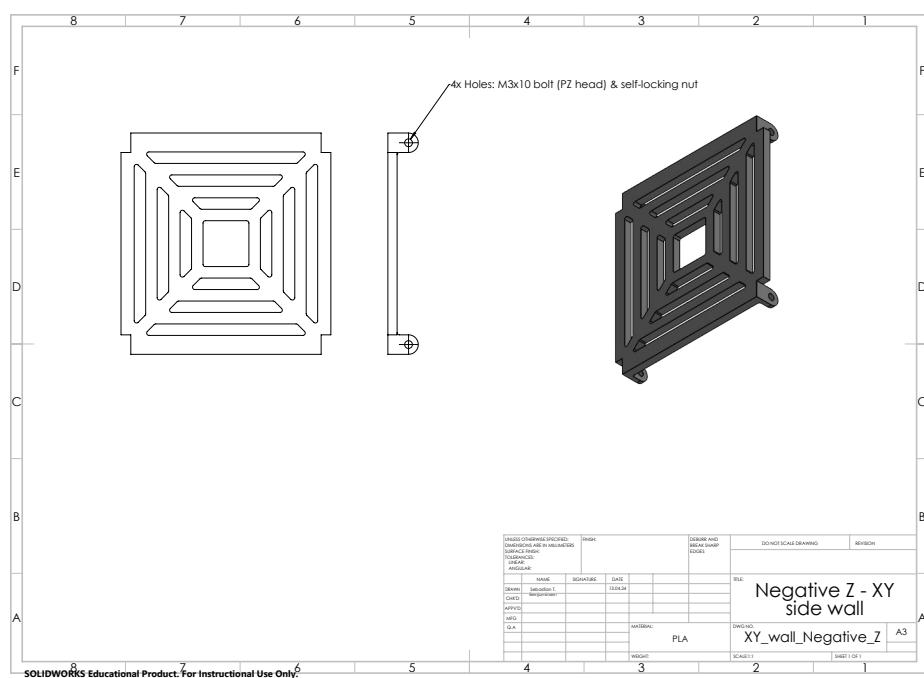


Figure 570: Negative Z - XY Side wall | STB

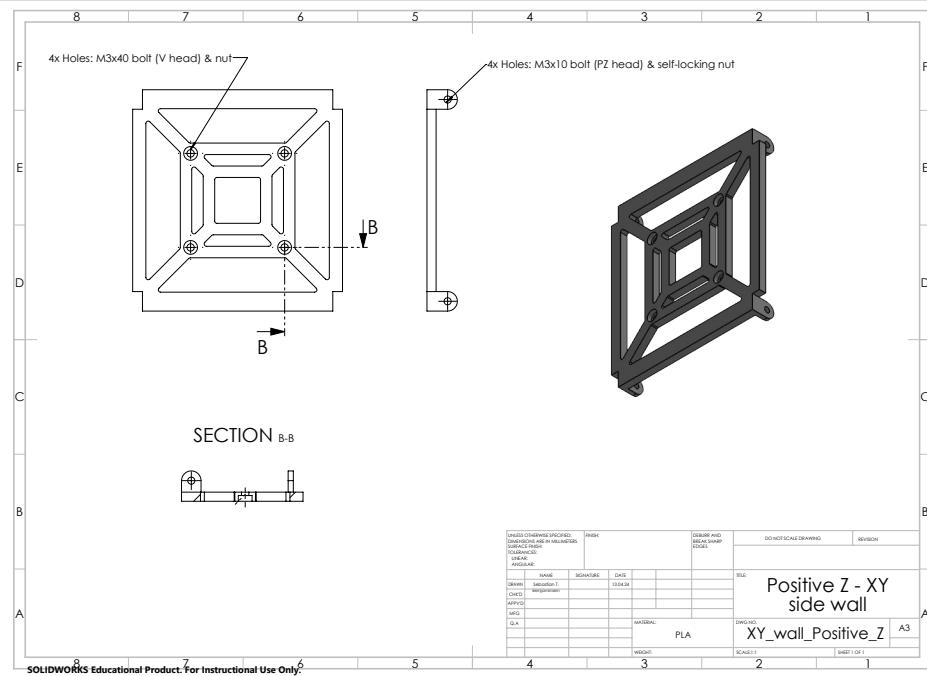


Figure 571: Positive Z - XY Side wall | STB

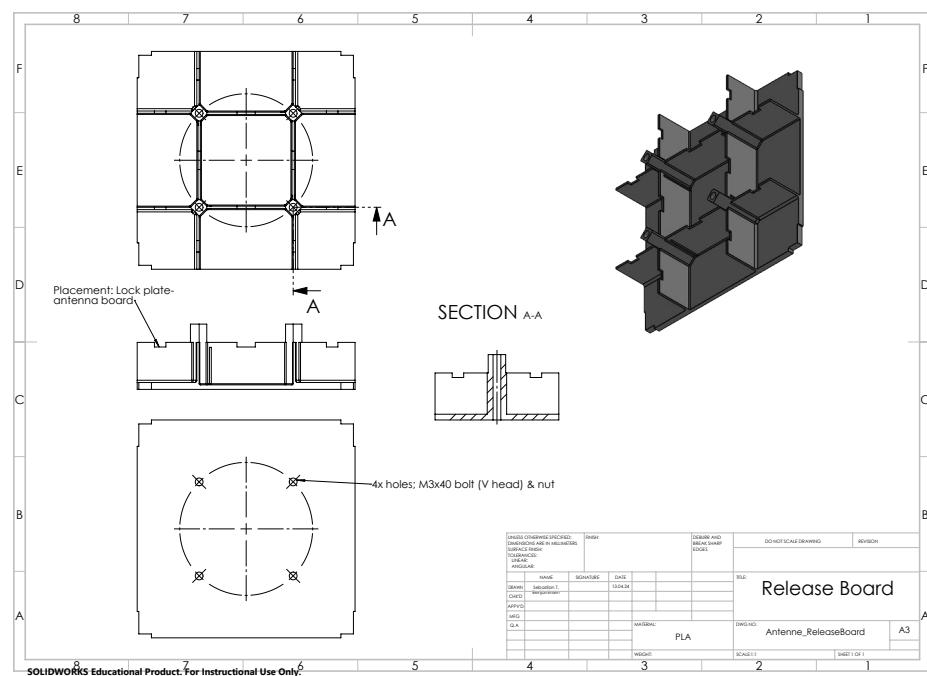


Figure 572: Release board | STB

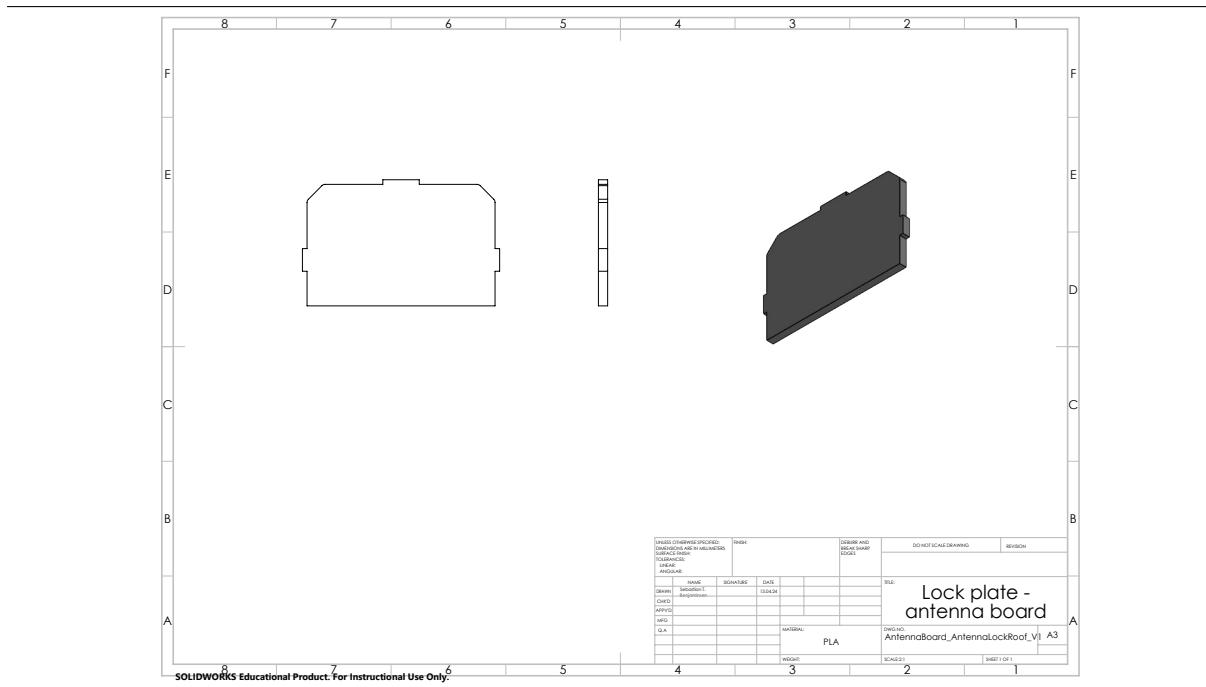


Figure 573: Lock plate - antenna board | STB

X.2 Mounting manual of C.R.A.F.T-Sat frame

Task (1).

Items used:

- (d) super glue

Instruction:

Use item (d) on the upcoming interface edges (edges two parts is touching each other) on the *1xlock plate* (fig 573) and place it on the *release board* (fig 572) where it has cut's for the edges from the *lock plate*. **Do this 4x times.** The result of task 1. shall be equal fig 574.

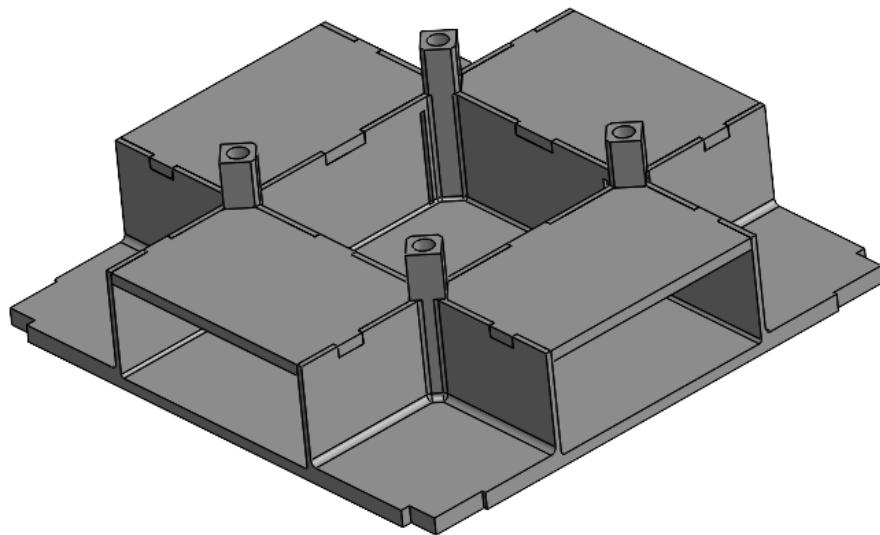


Figure 574: Result; mounting manual task 1. | STB

Note: Don't get any glue inside the critical area, fig 575

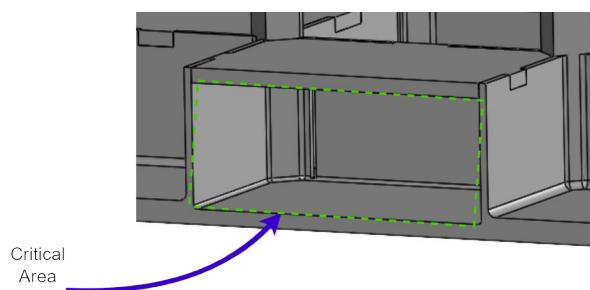


Figure 575: Critical area - antenna area | STB

Task (2).

Items used:

- (b) Slotted screwdriver (V4)
- (c) Wrench; size 5.5 mm
- (III) 4x M3 nut
- (IV) 4x M3x40 bolt (V4 head)

Instruction:

Place *positive Z - XY Side wall* (fig 571) on top of the mounting from task (1) (fig 574).

Note: the upcoming interface area (between *Side wall - ZY Plane* and *positive Z - XY Side wall*) on the *positive Z - XY Side wall* has to be parallel with the large cut area, fig 576.

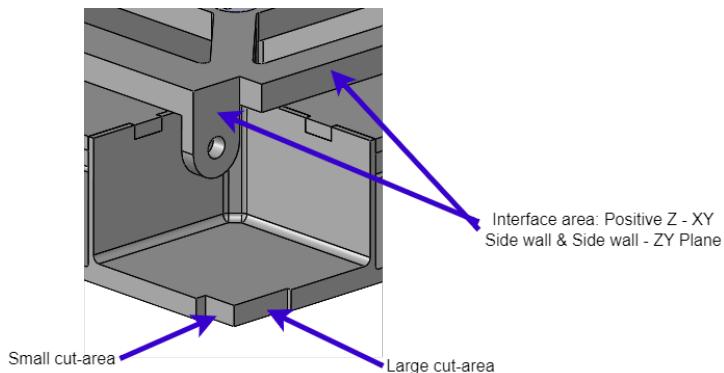


Figure 576: Interface area | STB

Place 1x of item (IV) on the top of the *positive Z - XY Side wall* and push it carefully down until the thread part pops out from the backside of the *release board*. Screw manually 1x of item (III) on the item (IV) until the item (III) interfacing with the backside of the *release board*. From there, use item (b) & (c) and tighten the (III) & (IV) until it's not possible to manually (without out tool (b) & (c)) unscrew them. **Do this 4x times.** The result of task 2. shall be equal fig 577.

Note: Don't use too much torque with the tool; only to the state where you can't use your hands to disassemble the parts.

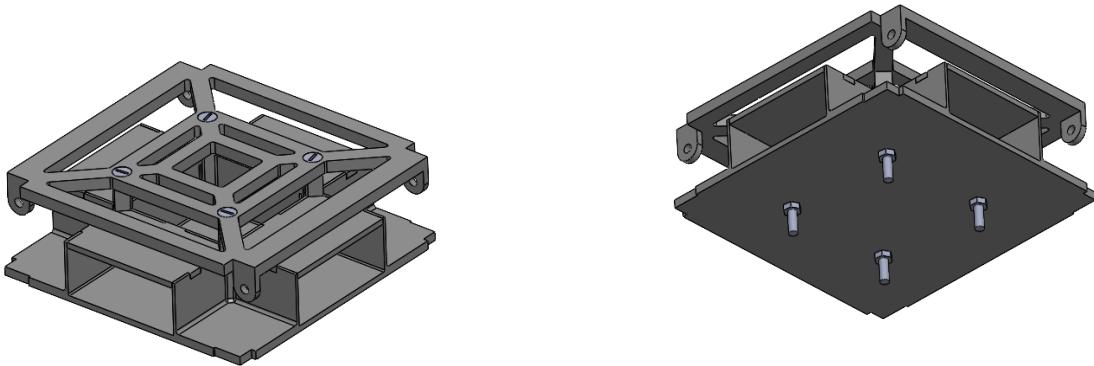


Figure 577: Result; mounting manual task 2. | STB

Task (3).

Items used:

- (a) Pozidriv screwdriver (PZ1)
- (c) Wrench; size 5.5 mm
- (I) 4x M3 locking nut
- (II) 4x M3x10 bolt (PZ1 head)

Instruction:

Place 1x*Side wall - ZY plane* (fig 569) on the mounting from task (2). The placement is described as interfacing in fig 576. Push 2x of item (II) inside the hole's designed for the item, and manually screw item 2x of item (I) on the threads on item (II) so it's simply placed on the locked on thread. From there, use tools (a) & (b) to mount together the components. The result of mounting this is shown in fig 578.

Note: only tighten *side wall - ZY plane* & and the mounting from task (2) to the point where they not wobbly.



Figure 578: Interface between *Side wall - ZY plane* & *Positive Z - XY Side wall* | STB

Note: the head on bolt in fig 578 is not equal item (II) but only a illustration of bolt with other measurements equal item (II).

Do this 2x times. The result of task 3. shall be equal fig 579.

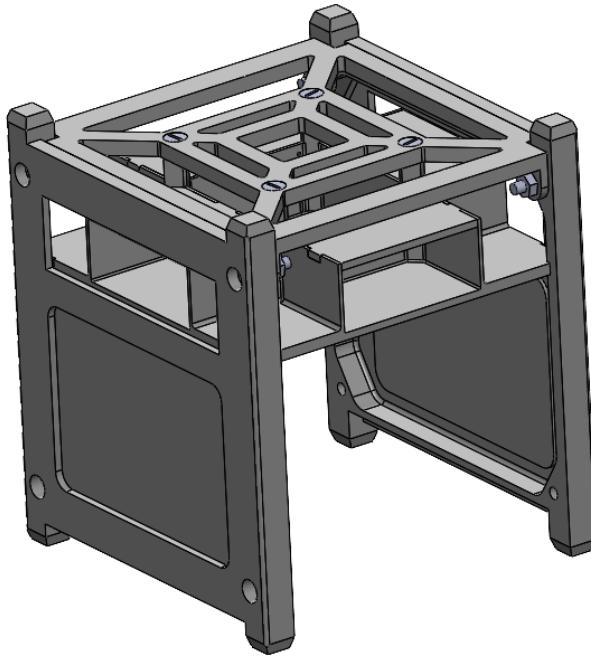


Figure 579: Result; mounting manual task 3. | STB

Task (4).

Items used:

- (a) Pozidriv screwdriver (PZ1)
- (c) Wrench; size 5.5 mm
- (I) 4x M3 locking nut
- (II) 4x M3x10 bolt (PZ1 head)

Instruction:

Place the *Negative Z - XY Side wall* (fig 570) on the mounting from task (3). The placement is described as interfacing, but with a *Positive Z - XY side wall* in fig 576 instead of *Negative Z - XY Side wall*.

Push 2x of item (II) inside the hole's designed for the item, and manually screw item 2x of item (I) on the threads on item (II) so it's simply placed on the locked on thread. From there, use tools (a) & (b) to mount together the the components. The result of mounting this is shown in fig 580.

Note: only tighten *side wall - ZY plane* & and the mounting from task (2) to the point where they not wobbly.



Figure 580: Interface between *Side wall - ZY plane* & *Negative Z - XY Side wall* | STB

Note: the head on bolt in fig 580 is not equal item (II) but only a illustration of bolt with other measurements equal item (II)

Do this 2x times. The result of task 4. shall be equal fig 581.

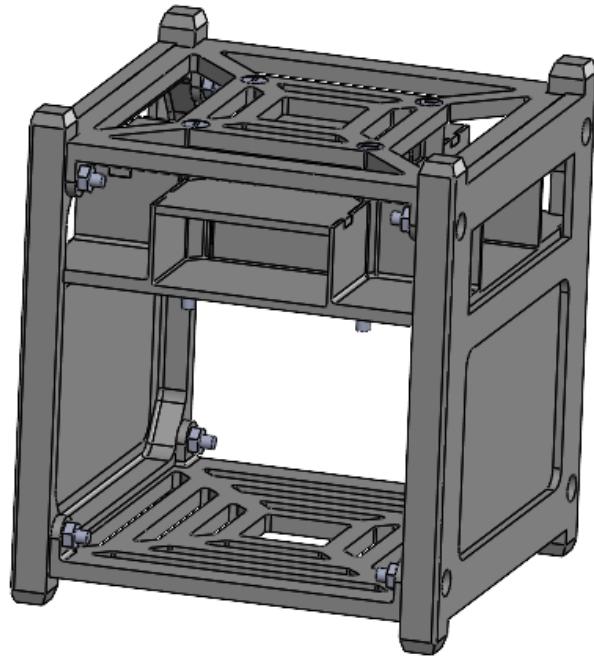


Figure 581: Result; mounting manual task 4. | STB

X.3 Risks

Possible risks during the presentation of current prototype is:

- User breaks the tools used to mount together the prototype.
- The user destroys 3D printed parts.
- The user destroys the assembly guide.
- The user misunderstand the mounting manual.

Y Helical Spring - Antenna release mechanism calculation

The spring chosen to be used on the C.R.A.F.T-Sat antenna release mechanism is a right-end compression spring with squared and ground ends. The given data from the shop about the spring used is:

- Outside diameter: $OD = 0.250$ in
- Free length: $L_0 = 1.50$ in
- Wire diameter: $d = 0.028$ in
- Number of total coils: $N_t = 20$
- Material: Pre-galvanised music wire

According to reference; "the spring material has the highest tensile strength and can withstand higher stresses under repeated loading than other spring material. The spring should not be used in temperature above $120^{\circ}C$ ($250^{\circ}F$) or at subzero temperatures" [171].

Table 44: Source: [171], N_a = number of active coils | STB

Term	Plain and Ground
End coils, N_e	2
Total coils, N_t	$N_a + 2$
Free length, L_0	$pN_a + 2d$
Solid length, L_s	dN_t
Pitch, p	$(L_0 - 2d)/N_a$

In the design of C.R.A.F.T-Sat antenna release mechanism there will be a situation related to the static loading between the sealing and antenna release board related to the spring if the sealing is and isn't open. It's therefore chosen to calculate the force given from the spring on the sealing (it's important due to it's the only part rotating compared with the release board) to be used in the FEA.

The average diameter of the spring chosen to be used is:

$$D = OD - d = 0.250 - 0.028 = 0.222 \text{ in} \quad (56)$$

With the the table ... the number of active coils in the chosen compression spring is:

$$N_a = N_t - 2 = 20 - 2 = 18 \quad (57)$$

The solid length of the spring (the length of the spring when it's fully compressed) is

$$L_s = dN_t = (0.028)(20) = 0.56 \text{ in} \quad (58)$$

The length of the spring in the CAD is:

-
- When the sealing is open: $L_{\text{open}} = 17.5 \text{ mm} = 0.69 \text{ in}$
 - When the sealing is closed: $L_{\text{closed}} = 32.4 \text{ mm} = 1.28 \text{ in}$

Since the length of the spring when it's fully compressed is less than the length when the sealing is open and closed, the chosen spring is closer to be operative for our case.

The deflection value is calculated by using formula:

$$y = L_0 - L \quad (59)$$

where $L_0 > L \geq L_s$. The deflection value of the spring when it's when the sealing is open is:

$$y = 1.5 - 0.69 = 0.81 \text{ in}$$

When the sealing is closed, the deflection value is:

$$y = 1.5 - 1.28 = 0.22 \text{ in}$$

Since it's a pre-galvanised music wire with no more information from in the product we assume it's equal a Music wire A228 to be used for further calculation. Also assume the surface is smooth, free of defects, and has a bright lustrous finish. The modulus of rigidity is $G = 12.0 \text{ Mpsi}$. [171]

The spring rate from the chosen spring is:

$$K = \frac{d^4 G}{8D^3 N_a} = \frac{(0.028^4)(12.0 \cdot 10^6)}{8(0.222^3)(18)} = 4.68 \text{ lbf/in} \quad (60)$$

The equation to calculate the spring force, due to the amount of deflection is:

$$F = yK \quad (61)$$

where:

- y is the deflection value and
- K is the spring rate

The spring force when the sealing is open is $F = 3.79 \text{ lbf} = 16.86 \text{ N}$, and when the sealing is closed $F = 1.03 \text{ lbf} = 4.58 \text{ N}$.

Z Timeline-iterations

The content will be about the timeline-iterations and include pictures, documents, and drawing. A important part to notice is that this chapter is designed to fully describe every faults and will include both documentation about why a chosen designs which worked as planned and designs which did not work as planned. This chapter is divided in three main topics:

- The first timeline-iteration describes the design work done and results between the bachelor start-date; 3. January to the first presentation date; 6. February.
- The second timeline-iteration describes the design work and results after the first presentation; 7. February to the second presentation; 8. Mars.
- The third timeline-iteration describes the design work and results between the second presentation; 8. Mars to the visit at Vestsiden secondary school.

The document is mostly referred as notes due to the time before deadline of bachelor report.

Z.1 First Timeline Iteration

The first week, a first simple design was made; an description about this design is explained in appendix R. The fig 582, is the final first timeline-iteration CAD design of C.R.A.F.T-Sat, using SW.

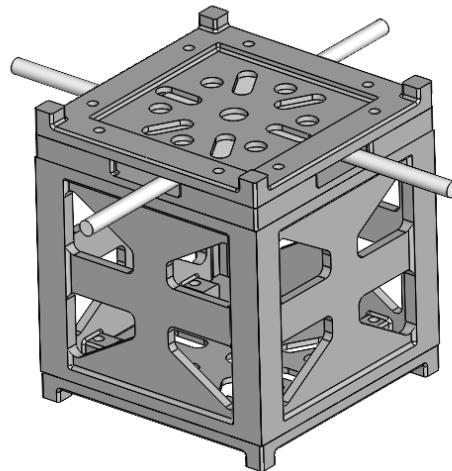


Figure 582: First timeline-iteration C.R.A.F.T-Sat design | STB

In the design of fig 582; their was a large amount of **design requirements**, some had to be considered and chosen to be used in the early stage of developing. The chosen requirements was 2. CubeSat must fit within a 1U test pod (at this early stage, the design was based on the CAD fit within, and not manufactured product fit within), and 4. C.R.A.F.T-Sat frame must answer to relevant CubeSat design specifications, A.20. The design was mostly aimed at the frame, but included consideration of the antenna and it's release mechanism. The CAD part files of the frame and antenna was converted to STereoLithography (STL), used in Rapid Prototyping, fig 583. During the printing of the Rapid Prototyping, harsh top lines occurred, and solutions had to been made. Those solutions is based on cura settings; the description of the solution is described in A.8.

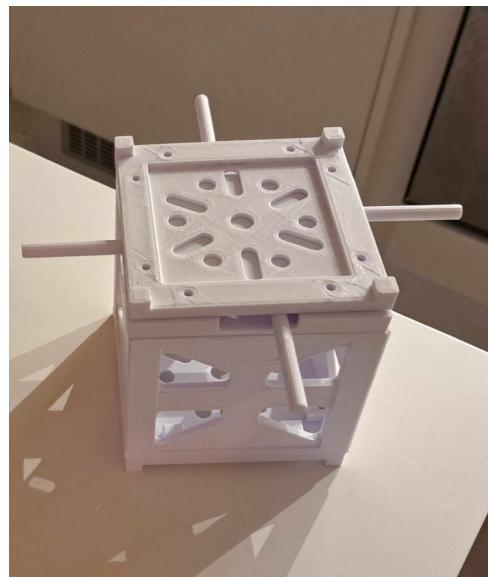


Figure 583: Rapid Prototyping of first timeline-iteration C.R.A.F.T-Sat design | STB

Frame design

The definition of side walls on the frame design is divided into two components, the ZY-wall and XY-wall. The name of those components are decided based on the coordinate system used in 181. This is because it make's the further developing and checking with design requirements related to 181 easier. The design of the first timeline-iteration XY-wall, fig 584, was made by consideration of the length from edge to edge in all axis and consideration of hole-placement, amount of holes, and mass removal on the face.

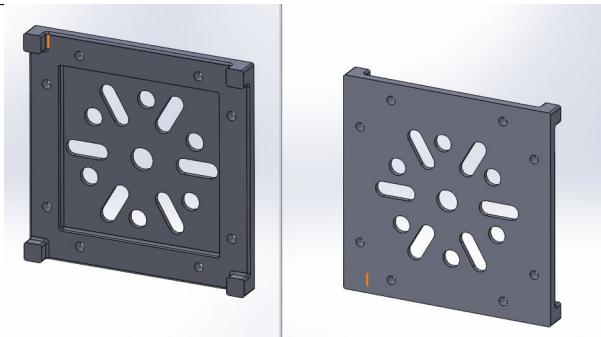


Figure 584: First timeline-iteration design; XY Wall | STB

The design of the first timeline-iteration ZY-wall, fig 585, was made by consideration of how the interfaces with the 584 and **antenna release board**.

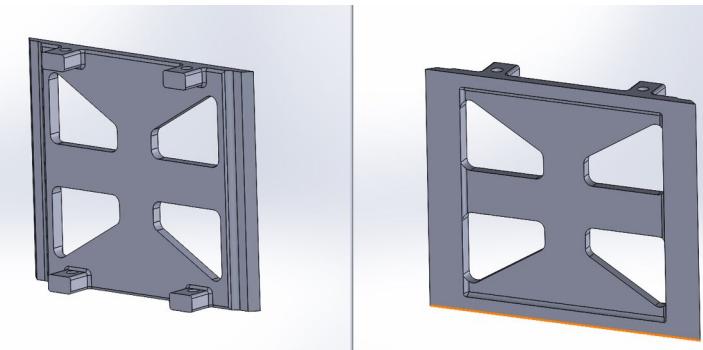


Figure 585: First timeline-iteration design; ZY Wall | STB

The physical interfaces in the design related only to the frame, was face to face interface between the fig 585 components and bolt/screw/pin interface between the fig 585 components and fig 584 components.

The chosen cut-design in fig 585 was decided due to mass removal and location of interfaces between components. The cut shape, and amount of mass removal will be considered in a **later timeline-iteration**

Antenna release mechanism

The first design, fig 587, illustrating the placement of the antenna. This illustration is excluding how the antenna mechanism could be used. This design will help for further investigation about the release mechanism due to the possible amount of space in the C.R.A.F.T-Sat and design of the frame.

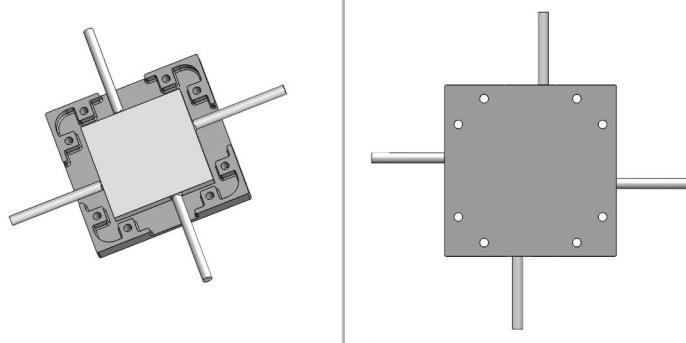


Figure 586: First timeline-iteration design; antenna board and simplified antenna | STB

The CAD file was converted to a STL file for Rapid Prototyping; a picture of the 3D printed antenna release design, fig 586.

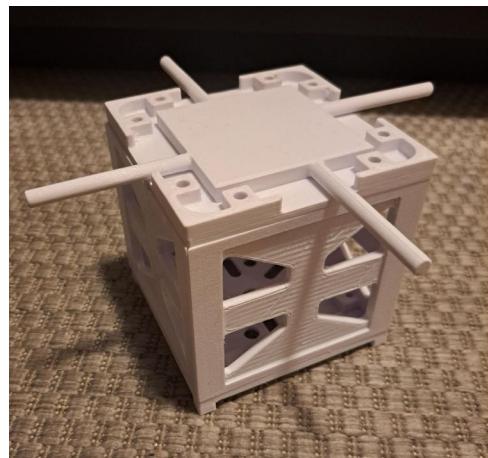


Figure 587: Rapid Prototyping of first timeline-iteration - antenna release design | STB

Z.2 Second Timeline Iteration

The fig 588, is the final second timeline-iteration CAD design of C.R.A.F.T-Sat, using SW.

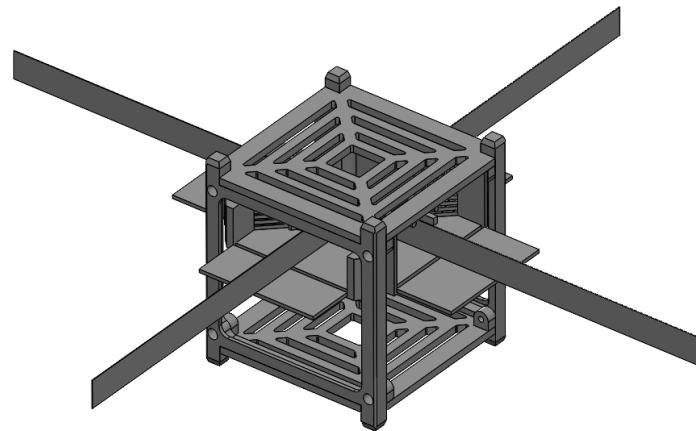


Figure 588: Second timeline-iteration C.R.A.F.T-Sat design | STB

Frame design

- Design requirement 3.3
- During the iterations, stakeholder gave a maximum mass

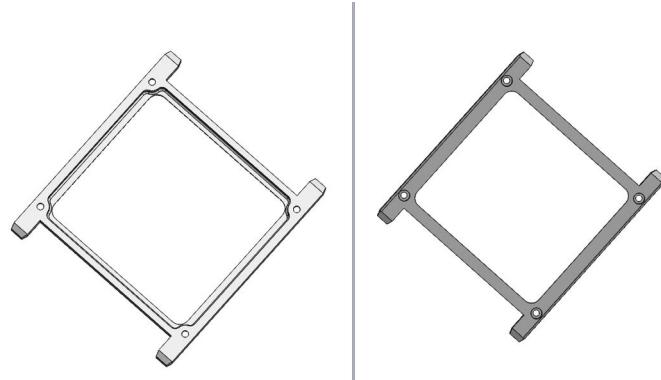


Figure 589: Second timeline-iteration design; ZY Wall | STB

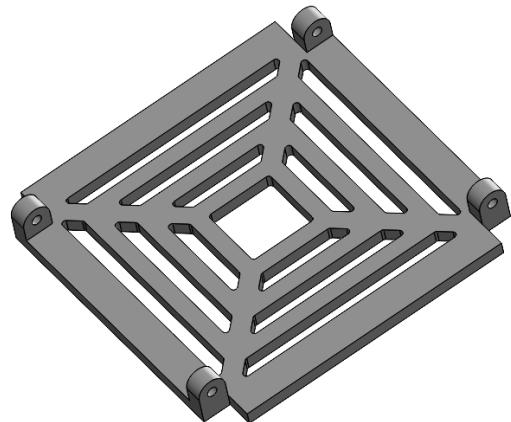


Figure 590: Second timeline-iteration design; XY Wall | STB

Antenna release mechanism

To further develop the release mechanism, the antenna release board; defined as the antenna board in Z.1, CAD model had to be re-made. The work started with an overall design of how the mechanism should work. Due to the curiosity and to have a design to be optimal for our purpose, investigation was done in searching after documents with different release mechanisms to get an idea of what is most reliable.

After hours with web-searching, the report document, ref. [177] , was discovered and gave the mechanical design team a idea to be further investigated. The idea was related to a solution of a simple release mechanisms with the possibility for dipole antenna with a large length. A start design where made, fig: 591, and was further investigated because of the possibilities of interacting with the signal waves.

After a conversation with the team members, the design ended with a conclusion of not optimal to our purpose because of a investigation about if the antenna is curved, the signal-waves will not be in the stage where the GUI-station will be capable to receive signals from the C.R.A.F.T-Sat.

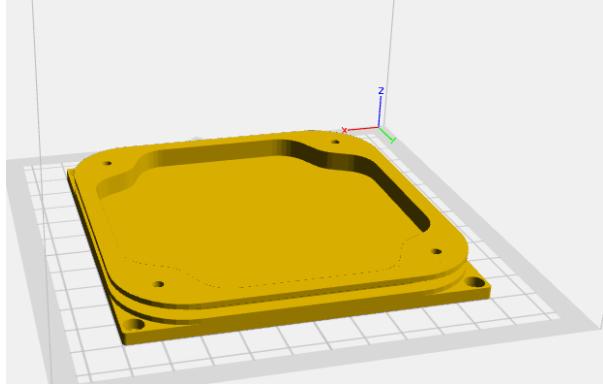


Figure 591: First design - antenna board | STB

A new plan had to be made. It started with stage; how should the backup design be made, due to the requirement of sending signals when the antenna mechanism break, appendix: G.4, and finding information about the frame material and if it will interact with signal waves, appendix: O.

The decision was at first decided to use the frequency equal 868 MHz, but it had the risk that the frequency 433 MHz was equal what ASE planned to used. The new design was a re-draw of the first design, ending with; fig 592. This was relative with a plan of using a frequency equal 868 MHz. The further work-plan from there was to continue with the design and find more information about these topics:

- The antenna's has to be placed on a flat surface, where the surface have at least a length of 86 mm.
- One of the following points has to be filled:
 - Material surrounding the antenna must not interact with the signals between the GUI/software.

-
- The physical interacting parts, who has a material interacting with the signals, has to be designed to minimize the interacting with the signals.

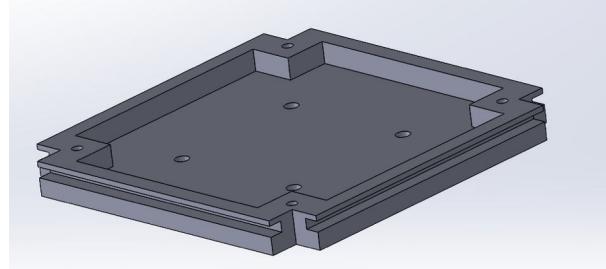


Figure 592: Second design - antenna board | STB

After a meeting with ASE it was described that the stakeholders was using frequency's equal 433 MHz, and made the second design not appropriate for further investigation.

A new design: the problem was solved with accepting not using up-link , and instead develop a timer-released system. Therefore, a new design with the name antenna release board, fig 593, was developed. This product will at first be developed for 3D printing and not massive production.

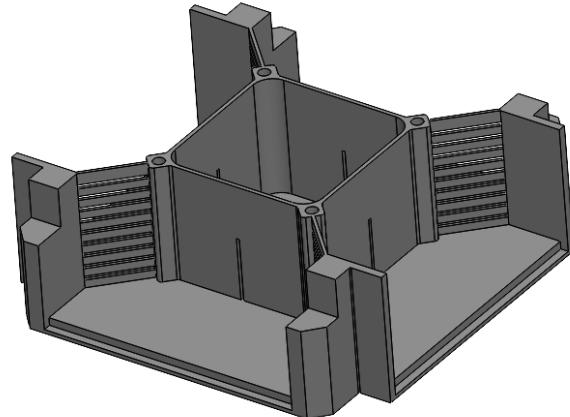


Figure 593: Second timeline-iteration design; Antenna release board | STB

The design was developed for interfacing with an antenna. This antenna has to have a stiffness and design to get the opportunity of being curve when folded and stiff when released. Fig 594 reveals a simple design of the release mechanism open.

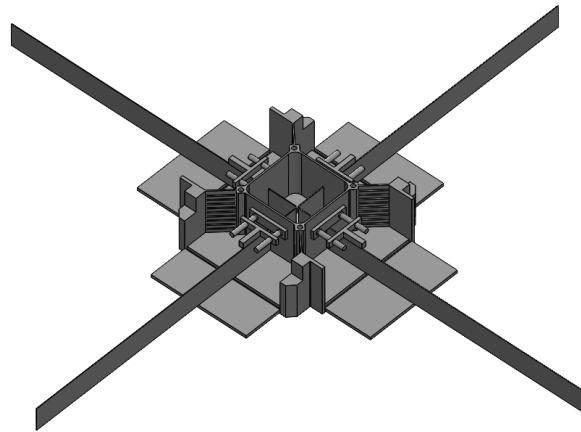


Figure 594: Second timeline-iteration design; Antenna release mechanism open | STB

The fig 595 reveals the simple design of folded. Both figures show's the spring board, fig596, rough developed to be used for opening the sealing with a spring mechanism.

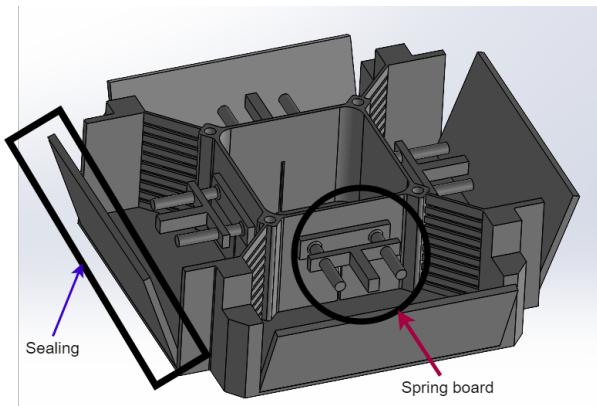


Figure 595: Second timeline-iteration design; Antenna release mechanism folded | STB

The first plan was based on developing a spring board, interfacing with the antenna release board. This product has a spring area where there is placed a compression spring; the material, size, and other data, was at this moment not specified. An burn-wire mechanism was connected to sealing interface part, used to compress the compressing spring. When the burn-wire was released, the compressed springs was released and the interface sealing area will push-open the sealing, fig 594

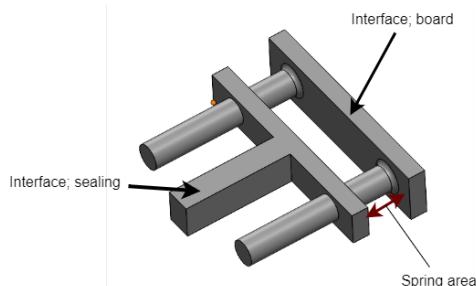


Figure 596: Second timeline-iteration design; | STB

Production of a Rapid Prototyping

During the Rapid Prototyping production problems occurred. Those problems resulted in poor products, fig: 597, and had to be solved to gain a passage for further developing and stop problems due to only Rapid Prototyping.



Figure 597: Poor 3D printed frame parts | STB

Comparing the Rapid Prototyping production before calibration, fig 597 and after calibration, fig 598; this proves calibration work was necessary for further Rapid Prototyping production.



Figure 598: Good 3D printed frame parts. | STB

During further CAD, the antenna release area has reached a further developed stage where the area for the antenna's is more visible and a prototype is 3D printed to show the team members the possible area, fig 599, for the planned antennas. This is also showing a further stage of the releasing mechanism using springs. The design with springs will be further developed in the third timeline-iteration.

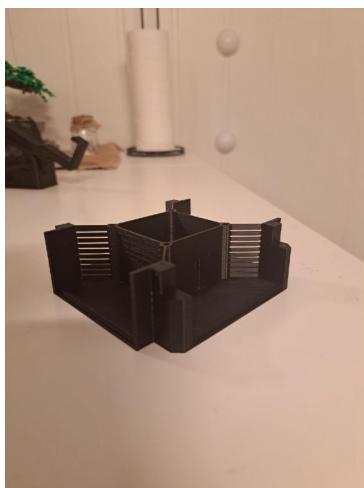


Figure 599: Design view - antenna release area | STB

All of the 3D printed components at this stage has a infill value equal 15%. This results in a mass equal a low mass value, but it will increase when further developing is happening in the next timeline-iteration.



Figure 600: Measuring mass of second timeline-iteration of C.R.A.F.T-Sat without bolts
- infill 15% | STB

The last design of C.R.A.F.T-Sat in the second timeline-iteration is shown in fig 601.



Figure 601: Second timeline-iteration of C.R.A.F.T-Sat - side view | STB

Z.3 Third Timeline Iteration

Due to time, the iteration has notes from work

Notes - iteration:

- It has been a further design of the positive and negative XY-Plane walls, representing the negative and positive Z-axis; the location of the design-center coordinate (X-, Y-, and Z-axis) is placed in the center of the product.

The negative XY-Plane wall will be updated later with location of interface between the electronical and negative XY-plane.

Material has been removed on the positive XY-Plane wall. A surface after removed material shows a simple knot can placed to interface with the balloon. The interface between the antenna release board and XY-Plane wall has been located, and will be analyzed to conclude if it is a optimal solution.

Because of to much upcoming unnecessary work with planning the physical combination of a 3D print and placement of a threaded nut during production. An solution has been made; the thickness of earlier threading location has been minimized and a self-lock nut will be considered if it's optimal in our case. The chosen standards of threaded item's is ISO 4762 - Hexagon socket Head Cap Screws, and ISO 4032 - Hexagon Nuts (not self-lock). More data, including calculation will be placed, to prove the chosen items is optimal for this design.

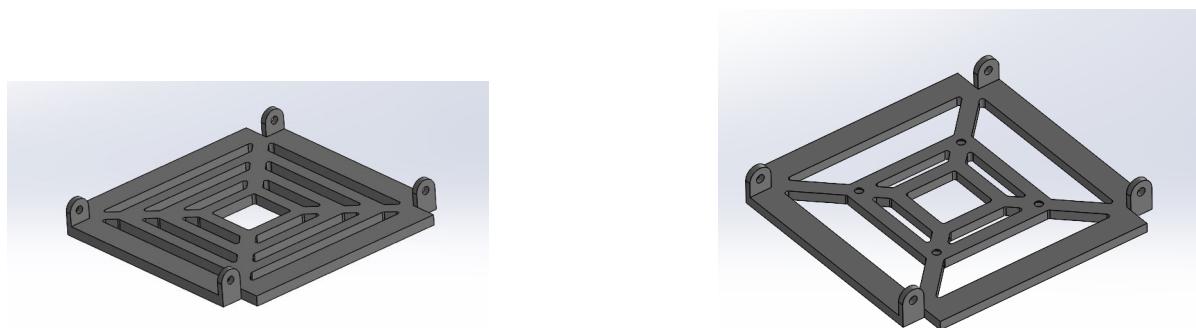


Figure 602: | STB

- In the second timeline-iteration, the antenna release area on the antenna release board, had a area for the antenna included with the release mechanism, but a further developing of this placement has been made. The design has mostly been between two areas, the sealing opening mechanisms and antenna "room". A rough value of the inside-volume of the antenna "room" is $V_{\text{Antenne "room"}} = (41) \cdot (18) \cdot (26.5) =$

19,557 mm³; this volume value doe's not include the cut inside the room, and changes in value due to 3D print manufacturing. In fig 603 , antenna hole, is where the antenna will be placed to interface with the electronical equipment.

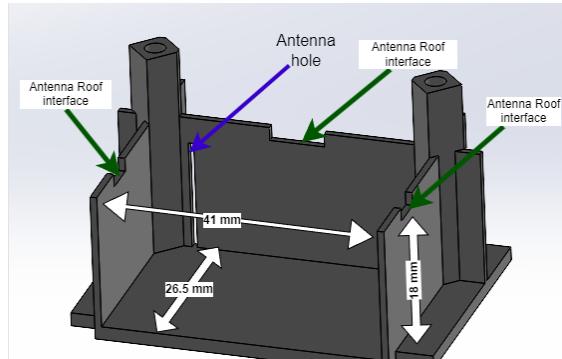


Figure 603: Inside volume | STB

The antenna roof interface is based on tight fit between the parts, and will be considered as an critical part, due to it's purpose of keeping the antenna inside a safe room during the period of when the antenna is not released. Material has

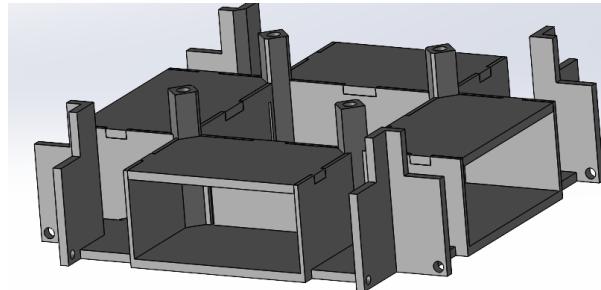


Figure 604: Antenna release board | STB

been removed on the antenna release board, and placement of holes and further consideration of material removal is not finished. This part is designed to be 3D printed.

- The sealing has been updated to include a four antennas (changes to two in final iteration) This is how the sealing will be closed and keep the antenna inside it's antenna "room" during a specified period. It will be further designed; considering the diameter of the hole to be used for a product connecting the sealing and release board.

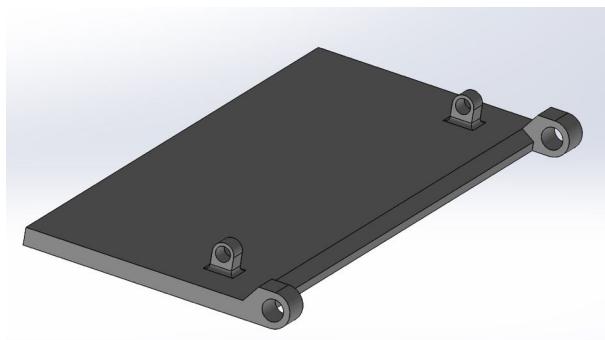


Figure 605: Sealing | STB