TU Darmstadt Space Technology

TUDSaT

Development of a CubeSat communications system based on CCSDS and ECSS standards

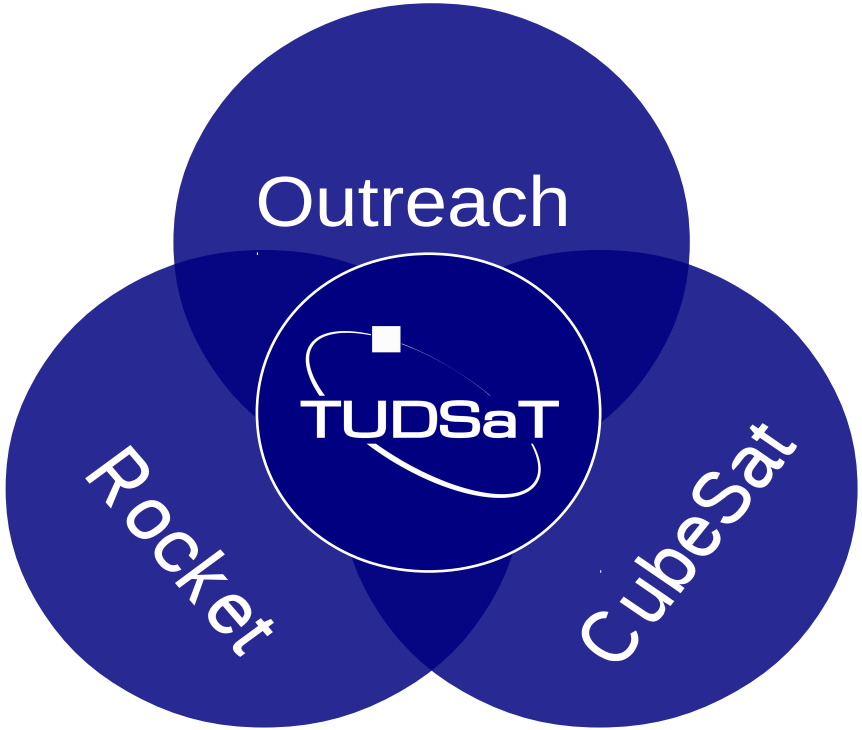
Open Source CubeSat Workshop 2018
Madrid 25.09.2018

Milenko Starcik
TU Darmstadt Space Technology e. V.

Our Team

TUDSaT

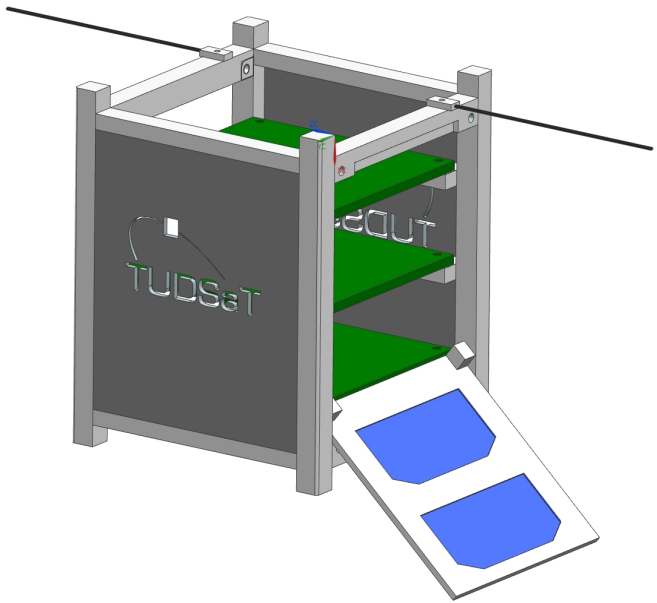
Our Mission



Our CubeSat

Open Source

1U - CubeSat



Technology Demonstrator

ECSS & CCSDS Standards

Reflectarray Antenna

CCSDS and ECSS standards

Global

Blue: Recommended Standards

Red: Draft Recommended Standard

Magenta: Recommended Practices

Green: Informational Reports

CCSDS and ECSS standards

OSI Layers	CCSDS Standards
7 Application	Mission Operation (MO) Services
6 Presentation	Message Abstraction Layer (MAL)
5 Session	Message Abstraction Layer Space Packets Binding
4 Transport	Space Packets
3 Network	
2 Data Link	Unified Space Data Link Telemetry Synchronisation and Channel Coding Telecommand Synchronisation and Channel Coding
1 Physical	Radio Frequency and Modulation System

User friendly?

CCSDS and ECSS standards

OSI Layers	CCSDS Standards	Pages
7 Application	Mission Operation (MO) Services	65+62+71+290
6 Presentation	Message Abstraction Layer (MAL)	178
5 Session	Message Abstraction Layer Space Packets Binding	72
4 Transport	Space Packets	49
3 Network		
2 Data Link	Unified Space Data Link	168
	Telemetry Synchronisation and Channel Coding	92
	Telecommand Synchronisation and Channel Coding	43
1 Physical	Radio Frequency and Modulation System	280
		= 1370

Cooperation

Pros and Cons

TUDSaT

Pro:

- Everything for free
- Robust
- Cooperation
- Reusability

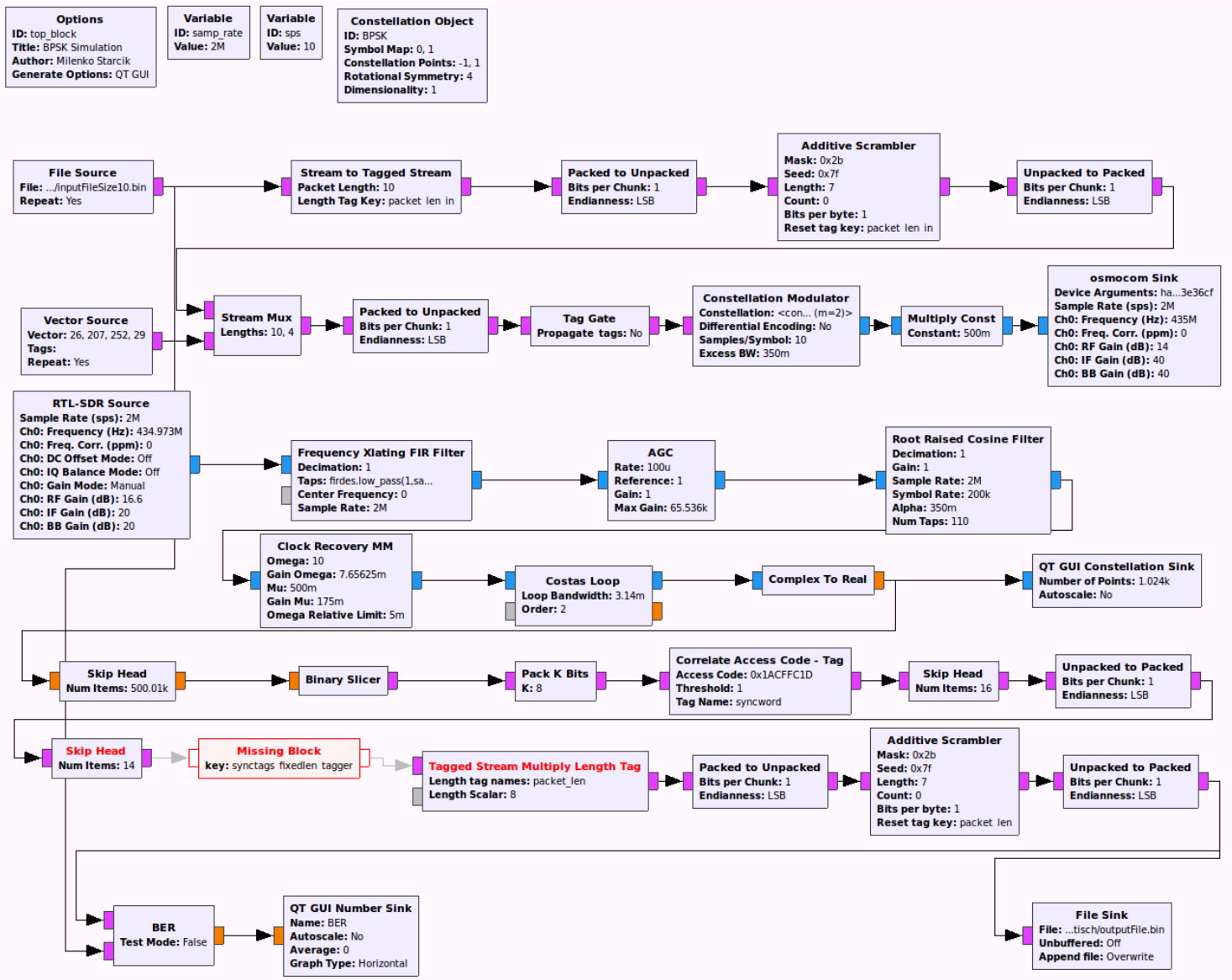
Contra:

- Difficult to understand
- Overhead
- Unused functionality

TUDSaT

Summer of Code in Space 2017

“CCSDS frame exchange over Radio Link in GNURadio”


Options
ID: top_block
Title: BPSK Simulation
Author: Milenko Starcik
Generate Options: QT GUI
Variable
ID: samp_rate
Value: 2M
Variable
ID: sps
Value: 10
Constellation Object
ID: BPSK
Symbol Map: 0, 1
Constellation Points: -1, 1
Rotational Symmetry: 4
Dimensionality: 1
File Source
File: .../inputFileSize10.bin
Repeat: Yes
Stream to Tagged Stream
Packet Length: 10
Length Tag Key: packet_len_in
Packed to Unpacked
Bits per Chunk: 1
Endianness: LSB
Additive Scrambler
Mask: 0x2b
Seed: 0x7f
Length: 7
Count: 0
Bits per byte: 1
Reset tag key: packet_len_in
Unpacked to Packed
Bits per Chunk: 1
Endianness: LSB
Vector Source
Vector: 26, 207, 252, 29
Tags:
Repeat: Yes
Stream Mux
Lengths: 10, 4
Packed to Unpacked
Bits per Chunk: 1
Endianness: LSB
Tag Gate
Propagate tags: No
Constellation Modulator
Constellation: <con... (m=2)>
Differential Encoding: No
Samples/Symbol: 10
Excess BW: 350m
Multiply Const
Constant: 500m
osmocom Sink
Device Arguments: ha...3e36cf
Sample Rate (sps): 2M
Ch0: Frequency (Hz): 435M
Ch0: Freq. Corr. (ppm): 0
Ch0: RF Gain (dB): 14
Ch0: IF Gain (dB): 40
Ch0: BB Gain (dB): 40
RTL-SDR Source
Sample Rate (sps): 2M
Ch0: Frequency (Hz): 434.973M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Off
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 16.6
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20
Frequency Xlating FIR Filter
Decimation: 1
Taps: firdes.low_pass(1,sa...
Center Frequency: 0
Sample Rate: 2M
AGC
Rate: 100u
Reference: 1
Gain: 1
Max Gain: 65.536k
Root Raised Cosine Filter
Decimation: 1
Gain: 1
Sample Rate: 2M
Symbol Rate: 200k
Alpha: 350m
Num Taps: 110
Clock Recovery MM
Omega: 10
Gain Omega: 7.65625m
Mu: 500m
Gain Mu: 175m
Omega Relative Limit: 5m
Costas Loop
Loop Bandwidth: 3.14m
Order: 2
Complex To Real
QT GUI Constellation Sink
Number of Points: 1.024k
Autoscale: No
Skip Head
Num Items: 500.01k
Binary Slicer
Pack K Bits
K: 8
Correlate Access Code - Tag
Access Code: 0x1ACFFC1D
Threshold: 1
Tag Name: syncword
Skip Head
Num Items: 16
Unpacked to Packed
Bits per Chunk: 1
Endianness: LSB
Skip Head
Num Items: 14
Missing Block
key: synctags fixedlen tagger
Tagged Stream Multiply Length Tag
Length tag names: packet_len
Length Scalar: 8
Packed to Unpacked
Bits per Chunk: 1
Endianness: LSB
Additive Scrambler
Mask: 0x2b
Seed: 0x7f
Length: 7
Count: 0
Bits per byte: 1
Reset tag key: packet_len
Unpacked to Packed
Bits per Chunk: 1
Endianness: LSB
BER
Test Mode: False
QT GUI Number Sink
Name: BER
Autoscale: No
Average: 0
Graph Type: Horizontal
File Sink
File: ...tisch/outputFile.bin
Unbuffered: Off
Append file: Overwrite

TUDSaT

Documentation

Version Control

Problems

Installation

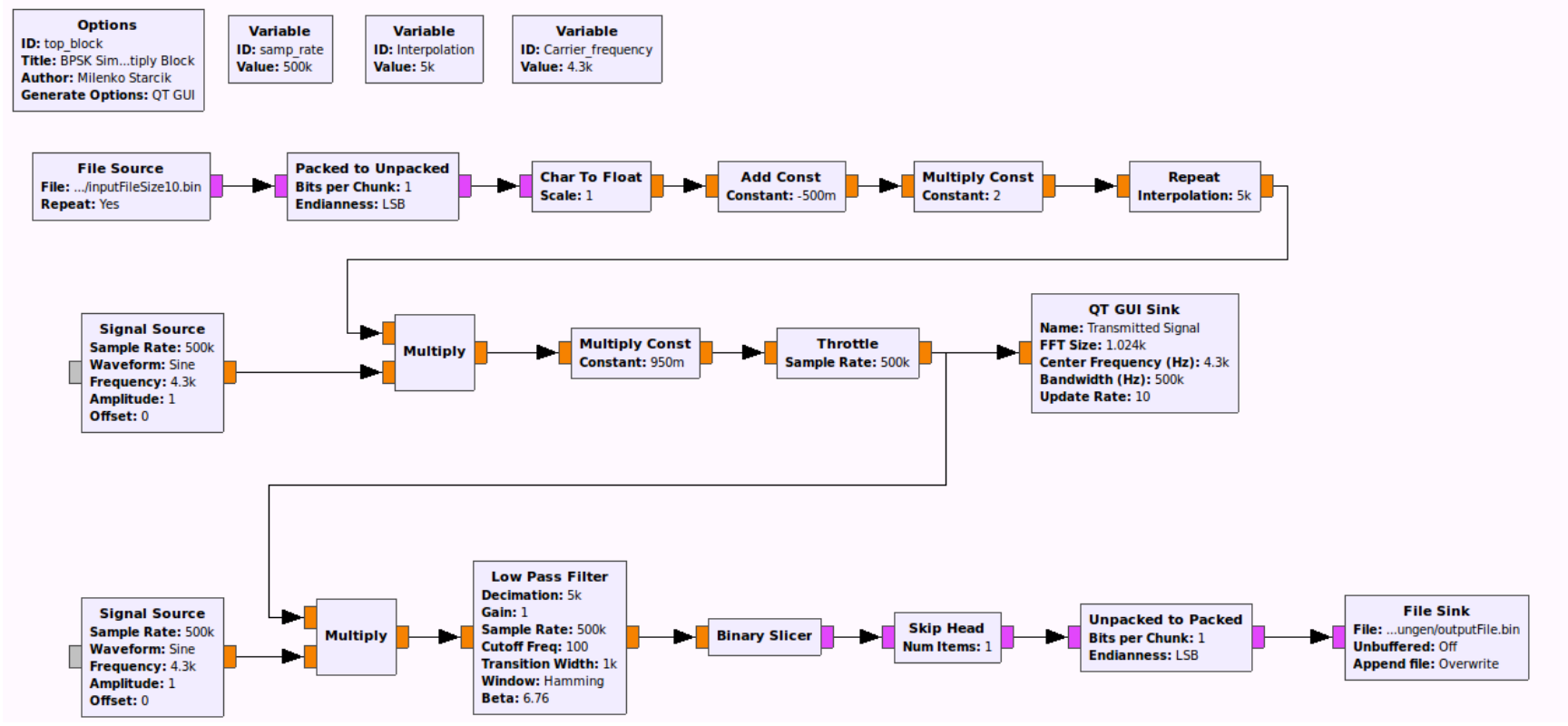
Bugs

hackRF-one

Keep it simple

Solutions

Summer of Code in Space



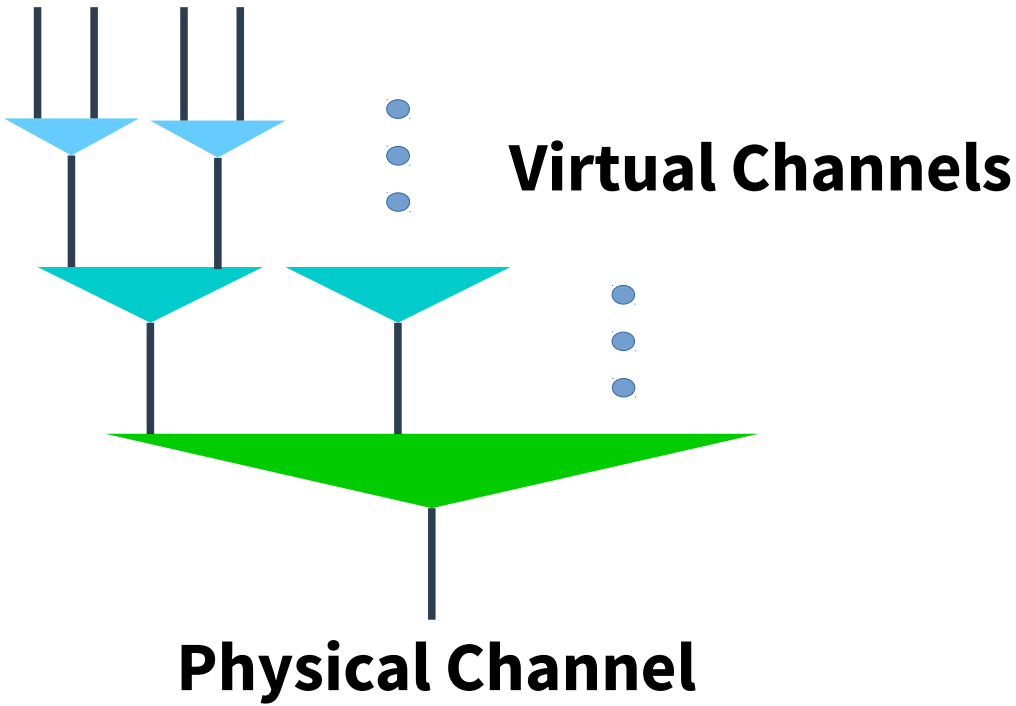
Synchronization and Channel Coding

TUDSaT

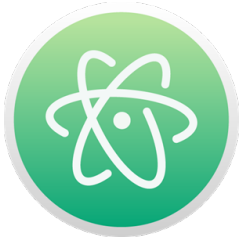
Telemetry

- Error-control coding
  - Reed-Solomon coding
- Frame validation
- Synchronization
- Pseudo-randomizing

Unified Space Data Link



Unified Space Data Link



Project

- USLP
  - __pycache__
  - documentation
  - __init__.py
  - TransferFrame.py
  - TransferFrameDataField.py
  - TransferFrameDataFieldHeader.py
  - TransferFramePrimaryHeader.py
  - TruncatedTransferFrame.py
  - TruncatedTransferFramePrimaryHeader.py
  - utils.py

TransferFrame.py

```python
class TransferFrame:
    """
    A Transfer Frame is used to carry the data in the Unified Space Data Link Protocol.
    """
    def __init__(self,
                 transfer_frame_primary_header,
                 transfer_frame_insert_zone,
                 transfer_frame_data_field,
                 operational_control_field,
                 frame_error_control_field):

        '''
        :param transfer_frame_primary_header: Contains information about the Frame
        :type transfer_frame_primary_header: TransferFramePrimaryHeader
        :param transfer_frame_insert_zone: Is used by the Insert Service
        :type transfer_frame_insert_zone:
        :param transfer_frame_data_field: Contains the data of the Frame
        :type transfer_frame_data_field: TransferFrameDataField
        :param operational_control_field: Provides a mechanism to report some real-time functions
        :type operational_control_field:
        :param frame_error_control_field: Provides the capability to detect errors
        :type frame_error_control_field:
        '''

        self.transfer_frame_primary_header = transfer_frame_primary_header
        self.transfer_frame_insert_zone = transfer_frame_insert_zone
        self.transfer_frame_data_field = transfer_frame_data_field
        self.operational_control_field = operational_control_field
        self.frame_error_control_field = frame_error_control_field

```

Welcome to Unified Space Data Link Protocol's documentation!

Transfer Frame

class `TransferFrame.TransferFrame`(*transfer_frame_primary_header, transfer_frame_insert_zone, transfer_frame_data_field, operational_control_field, frame_error_control_field*)

A Transfer Frame is used to carry the data in the Unified Space Data Link Protocol.

__init__(*transfer_frame_primary_header, transfer_frame_insert_zone, transfer_frame_data_field, operational_control_field, frame_error_control_field*)

Parameters:
- **transfer_frame_primary_header** (*TransferFramePrimaryHeader*) – Contains information about the Frame
- **transfer_frame_insert_zone** – Is used by the Insert Service
- **transfer_frame_data_field** (*TransferFrameDataField*) – Contains the data of the Frame
- **operational_control_field** – Provides a mechanism to report some real-time functions
- **frame_error_control_field** – Provides the capability to detect errors

Transfer Frame Data Field

class `TransferFrameDataField.TransferFrameDataField`(*transfer_frame_data_field_header, transfer_frame_data_zone*)

The Transfer Frame Data Field contains the data that should be delivered by the frame.

__init__(*transfer_frame_data_field_header, transfer_frame_data_zone*)

Parameters:
- **transfer_frame_data_field_header** (*TransferFrameDataFieldHeader*) – Contains information about the Transfer Frame Data Field
- **transfer_frame_data_zone** (*str*) – Contains the real data

Transfer Frame Data Field Header

class `TransferFrameDataFieldHeader.TransferFrameDataFieldHeader`(*TFDZ_construction_rules,*

Active Prototypes

OSI Layers	CCSDS Standards
7 Application	Mission Operation (MO) Services
6 Presentation	Message Abstraction Layer (MAL)
5 Session	Message Abstraction Layer Space Packets Binding
4 Transport	Space Packets
3 Network	
2 Data Link	Unified Space Data Link Telemetry Synchronisation and Channel Coding Telecommand Synchronisation and Channel Coding
1 Physical	Radio Frequency and Modulation System


$\Delta V = V_e \ln \frac{m_0}{m_B}$
$E = mc^2$
Rocket (engl. Version)
42
Curious?
Web: tudsat.space Twitter: TUDSaT
Facebook: TUDarmstadtSpaceTechnology
TUDSaT

milenko Mar 21

Hey all,

I am Milenko from TU Darmstadt Space Technology, a stundent group from Darmstadt, Germany.
We are currently beginning to develop a CubeSat and a Rocketry programm which both should be made open source from the very beginning.
Before setting up the gits and so on we want to make sure that the licencing is correct.
It would be interesting to us to know how you are managing intellectual property that is produced for the Libre Space Foundation.
Which licences do you prefer for Software and for Hardware and why?
An other Question is, under which name is the Software/Hardware published?
Is it enough to set up a repository with the licence in it, write our groups name on it and anyone contributing to it is accepting it that way?

Greetings from Darmstadt,

Milenko

1

created	last reply	replies	views	users	likes	links
Mar 21	Apr 18	3	206	2	7	2

elkos Mar 27

Some basic stuff. The larger a project gets it is more common to use several licenses. There are many open-source licenses that a project can pick according to it's needs and licensing can be an issue of debate (although we had consensus early on when we started). We wanted to provide our users with a license that will cover the following liberties for our users:
0 freedom to run
1 freedom to study
2 freedom to share
3 freedom to modify

We wanted a license that has a strong copyleft, meaning that if you do modify our source code and distribute it (or use it on your website for the AGPLv3 parts) you should share your modifications with the same license.

GNU GPLv3 for software. This license is a copyleft. Keep in mind that the v3 has several provisions regarding to against software patents, DRM and more. Also keep in mind that GPLv3 is recognized as a legal document in several jurisdictions.
GNU AGPLv3,for websites actually is a special version of the GPL,that covers the application layer service loophole
LGPL is a "weaker copyleft" license designed for libraries.

Hardware is licensed CERN Open Hardware Licence v1.2 which provides a strong copyleft and is in my opinion closer to the copyleft paradigm on GPL.

Hope that helped.

Mar 21

1 / 4
Mar 22

Apr 18