NANOLINK: A ROBUST AND EFFICIENT PROTOCOL FOR SMALL SATELLITE RADIO LINKS

Conference Paper · May 2016

CITATIONS
14

READS
1,885

3 authors:

Nicolas Appel
Technische Universität München
10 PUBLICATIONS **54** CITATIONS
SEE PROFILE

Sebastian Rückerl
Technische Universität München
14 PUBLICATIONS **47** CITATIONS
SEE PROFILE

Martin Langer
Technische Universität München
56 PUBLICATIONS **506** CITATIONS
SEE PROFILE

NANOLINK: A ROBUST AND EFFICIENT PROTOCOL FOR SMALL SATELLITE RADIO LINKS

Nicolas Appel, Sebastian Rückerl, Martin Langer
Institute of Astronautics, Technische Universität München, Germany
{nicolas.appel, sebastian.rueckerl, martin.langer}@tum.de

Abstract

We present Nanolink, a data link layer protocol for CubeSats and other spaceborne assets with similar bandwidth and hardware resources. The protocol is designed to operate with high efficiency and high reliability over links with a small bandwidth-delay product and moderate to weak signal quality. A type-I hybrid automatic repeat request (ARQ) scheme and an extensible header structure reduce the overhead added by unused protocol features, thus minimizing the overhead added on the return channel by the ARQ. Simulations show a good performance of the protocol, despite high bit error rate on the channel. Furthermore, the return channel bandwith efficiency of the protocol allows its use on asymmetric links.

1 Introduction

Since their introduction, CubeSats have evolved from purely educational missions to spacecraft with a broad variety of scientific and commercial applications. Currently implemented communication systems on CubeSats are constrained by their low-to-medium up- and downlink rates as well as their limited reliability. Partly responsible for this is the frequent use[1] of AX.25[2], an amateur radio version of X.25, which is used by amateur radio enthusiasts for packet radio. AX.25 is inherently ill-suited for space applications, since it lacks efficient error control and generally introduces high overhead. This can result in problems during mission operations[3, 4].

Most radio links in commercial or government space applications use implementations of protocols of the Consultative Committee for Space Data Systems (CCSDS). These protocols provide a wide range of capabilites that cover the needs of standard LEO, GEO and deep space missions. There has been a mission with CCSDS compatible radios on a CubeSat[5]. However, in the discussed application the CCSDS protocols were used on broadband S- and X-Band links. The typical CubeSat radio links use narrowband links in the amateur radio frequency bands. For these links, the protocols generate too much overhead, since most features are not required. Additionally, CCSDS compatibility requires a compatible physical layer. This is problematic since many commercial-off-the-shelf CubeSat transceivers do not provide this capability and suitable ground stations are costly and often exceed the budget of the CubeSat project[1].

However, CCSDS protocols such as Proximity-1[6] provide many features that are relevant to CubeSat radio communications, such as channel multiplexing, hybrid automatic repeat request (ARQ) and in-orbit transceiver control mechanisms. The Satellite Transport Protocol by Henderson and Katz[7]

is an interesting approach to develop a transport protocol capable of handling the high delay and asymmetry of earth to GEO links. For this purpose, they adapted the ARQ mechanism of SSCOP[8], an ATM transport protocol. A special property of this mechanism is its very economical use of the return link.

For future applications a reliable protocol tailored to the needs of CubeSats is needed to support basic functions as debugging and software updates as well as potential scientific missions requiring deep space or satellite-to-satellite communication. For this reason, we developed the protocol Nanolink. The protocol is designed to resolve all reliability issues associated with amateur radio protocols, but also created to be more lightweight and efficient than CCSDS protocols. Additionally, we believe that extensibility is an important aspect of a protocol (i.e. new features can be added, without redesigning the protocol).

2 Basic Operation

Nanolink is a reliable, packet oriented, connection based data link layer protocol. The main function of the protocol is to reliably transfer data packets of variable length to another node. To ensure reliability, the protocol uses a combination of forward error correction (FEC) and automatic repeat request (ARQ).

A key feature of Nanolink is the ability to multiplex several frame streams into one physical channel through 'virtual channels' (see figure 1). These virtual channels can have different priorities, latency requirements and generally facilitate the implementation of traffic classes. Additional features can be added to the protocol using extension headers, which can be appended to the frame header.

At the beginning of each Nanolink session, the connection is set up with a simple two way handshake. Afterwards, the transmitter sends frames of variable length to the receiver. The frames are stored at the transmitter side for possible retransmission using the ARQ mechanism. Individual frames are identified by a sequence number in the frame header. The receiver uses selective acknowledgement (SACK) to notify the transmitter of the reception of frames and request the retransmission of missing frames. Sessions are terminated by explicitly closing the connection or by a timeout after losing the physical connection.

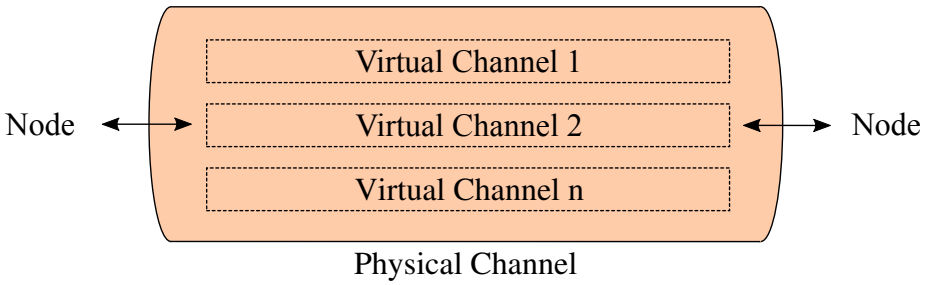


Figure 1: Virtual Channel multiplexing

2.1 Protocol operation

Nanolink offers two types of service quality: unreliable and reliable transmission, which will be discussed later. Protocol frames can be excluded from the ARQ mechanism by modifying a header flag. All protocol frames are secured by forwared error correction.

Figure 2 illustrates how FEC and framing work together. For bit synchronization, each code block is preceded by a codeword sync marker (CSM). Frames are placed into the code blocks, but not aligned to the code blocks. Frames are synchronized using the attached sync marker (ASM). Once a code block can not be decoded correctly, e.g. due to errors, at least one frame is lost. This frame is then retransmitted, but not the entire code block. This hybrid ARQ scheme works well under various bit error rate conditions, since the FEC can be adjusted to the expected environment. For lower signal-to-noise ratio (SNR), powerful codes like Gallager codes (LDPC) are sensible, while for high SNR or bursty error patterns Reed-Solomon codes are more efficient.

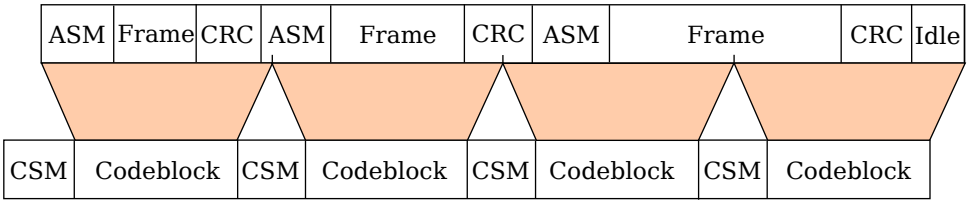


Figure 2: Frames are protected from errors using FEC

Reliable service

The reliable service ARQ mechanism was adapted from STP[7]/SSCOP[8]. It uses sequence numbers to identify frames within a sliding window of constant size. Transmitter and receiver exchange "STAT(us)" and "POLL" messages to synchronize their protocol state. In contrast to STP, there are no USTAT messages for requesting retransmissions for single frames. Also, POLL and STAT messages do not use time tags, since they are not needed on point-to-point links. POLL messages are sent by the transmitter for status polling and contain no further information. The receiver responds with a STAT message which contains the highest in-order received sequence number, the highest correctly received sequence number and the sequence numbers within that range, which were not received at the time. The layout of such a STAT message is illustrated in figure 3. STAT messages can also be sent without prior polling by the transmitter, e.g. after detection of a missing frame.

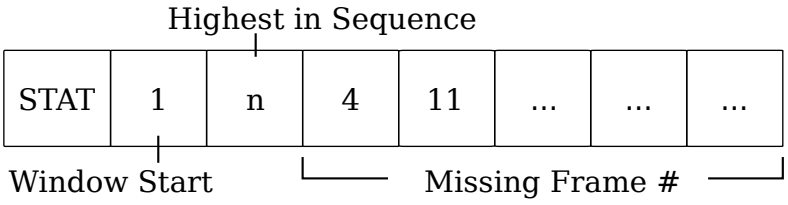


Figure 3: STAT message layout

POLL messages may be issued to control the size of the sent queue. The advantage of this modification is that STAT messages can now be sent independently from POLL messages, and be used to control the receiver queue size. STAT messages are sent after a timer expires or the threshold for the

size of the receive buffer is attained or the loss of a consecutive number of frames exceeding a certain threshold is detected.

The detection of missed frames is based on the sequence numbers of the last two correctly received frames, for the sake of this example called S_1 and S_2. If $S_2 = S_1 + 1$ no miss is detected. If $S_2 = S_1 + 2$, a single miss is detected and can be handled by issuing a STAT message, if it has not been received before. The case $S_2 \geq S_1 + 3$ is handled by comparing the sequence numbers in the interval with the receive buffer and sending a STAT message if necessary. Frames are discarded if S_2 is not within the window, or $S_2 = S_1$, or a frame with sequence number S_2 is already in the receive buffer.

Unreliable service

The unreliable service is the alternative to ARQ controlled transmissions. It is primarily thought for frames with expendable information, such as control messages, or telemetry data. Secondary users are upper-layer protocols that implement their own ARQ and packet reordering mechanisms. Packets sent via the unreliable service are not buffered by the transmitter and cannot be retransmitted. Unreliable frames have higher priority over reliable frames. The reason for this is that real-time data expires quickly and swiftness is preferred over reliability.

2.2 Framing

Nanolink transmits data in frames of variable size. The structure of these frames is illustrated in figure 4. A frame is preceded by a 24 bit synchronization marker. The marker is succeeded by a 3 byte frame header, up to 1021 bytes of payload and finally a CRC16 checksum, which is calculated over the entire frame.

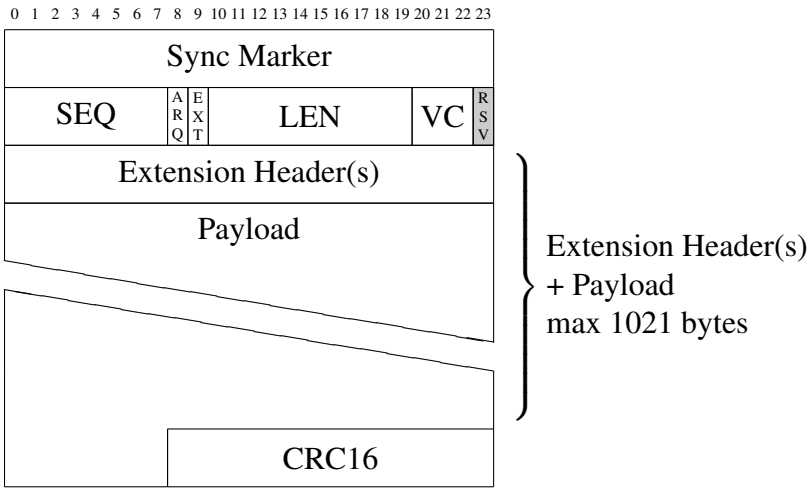


Figure 4: Nanolink frame

The Nanolink header contains a sequence number (SEQ), an ARQ and extension Header (EXT) flag, a length field (LEN), a virtual channel identifier (VC), and an reserved bit (RSV). The SEQ field is 8 bit long and thus allows a maximum window size of 127 frames. The relatively small size of 8 bit was choosen because ressources for buffering frames are limited on small satellites, especially when

using FPGA or microcontroller based implementations. Thus the number of frames in transmission is limited the same way. Hence, a window size of 127 seems sufficient. The ARQ flag indicates whether or not the frame is to be handled by the ARQ mechanism. The EXT flag indicates the presence of extension headers (discussed in more detail in 2.3). The following 10 bit length field allows the frames to be up to 1024 bytes in total length. The virtual channel identifier allows up to eight different data streams to be multiplexed into one physical channel. Different implementations may use these data streams for different traffic classes or other special cases like telemetry transmission. The last bit of the Nanolink frame header is reserved for future use. The header may be followed by one or more extension headers (see section 2.3) and the actual payload.

2.3 Extension Headers

The extension headers are a core feature of the Nanolink Protocol. The concept is already known from protocols such as IPv6[9]. Extension headers are optional protocol control elements that allow increased versatility and loose coupling without introducing significant constant overhead. Figure 5 shows one full extension header, including the optional length field and up to 256 bytes of payload.

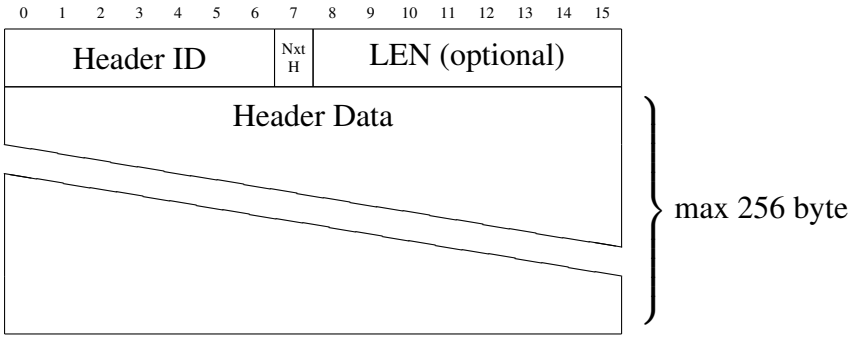


Figure 5: Nanolink Extension Header

Because the protocol has been designed with low bandwidth radio links in mind, an important requirement is low protocol overhead. Not all headers are required in the standard cases and therefore removed from the frame header, but not moved into special frames, as is usually done. Control frames, although possibly small, also need a complete frame header and sync word to detect the frame, and to distinguish it from data frames. The minimum possible size of a frame header is one byte, due to byte alignment, and therefore just as long as a minimum extension header. Due to the additional sync word, this solution offers high overhead while adding no value for low bandwidth-delay products.

Instead, the concepts of control frames and data frames are merged, so that only one frame format exists within Nanolink. This may cause additional delay, since long frames increase the delay for the receipt of control information. However, this is not a concern, since the bandwidth-delay product is assumed to be very small. Additionally, higher bandwidths/delays or channel asymmetries can be accommodated since it is possible to send frames without payload, containing only control information.

Extension headers are designed to be daisy chained, so that multiple extension header packets can be placed within one frame. The presence of a further extension header is indicated by setting the NxtH flag of the extension header. The advantage of this method is that it allows a very high number of extension headers inside a frame without the permanent overhead of a dedicated header counter field. The byte alignment requires the header packets to be at least 8 bit in length. A 8 bit ExtID field could

distinguish 256 different header types, an excessive amount, which is not required. Repurposing 1 bit of the header byte brings therefore no considerable limitations. However, this bit can be used as NxtH flag for header chaining. Therefore, only minimal overhead is introduced. A header counter would add permanent overhead, even if no extension header is used. Also, it would either limit the possible number of extension headers in a frame severely if the reserved bits of the frame header were used, or would require an entire additional byte. In any case, this solution is inferior to the chaining method.

As for most extension headers the exact size of the header data is constant and known a priori, a length field is not required, and all communications endpoints know these statically defined sizes. Only for some special cases, e.g. the ARQ acknowledgement for multiple frames at once, the size varies. Therefore the first byte of the header data is used as length field in these cases. This technique is obviously superior in terms of overhead to both chaining multiple headers of the same size or defining an arbitrary large but constant size. The variable size range is 1 to 256 bytes indicated by a length field with values of 0 to 255 respectively.

2.4 Mode Switching

To enable the change of certain protocol parameters during runtime, Nanolink supports mode switching. The procedure was adapted from Proximity-1[6] for the use in Nanolink. It is illustrated in Figure 6. The mode switch is depicted to affect both up- and downlink transmitter-receiver pairs, however this is not necessarily the case as a mode switch might only affect the uplink and not the downlink. The process is explained using the example below.

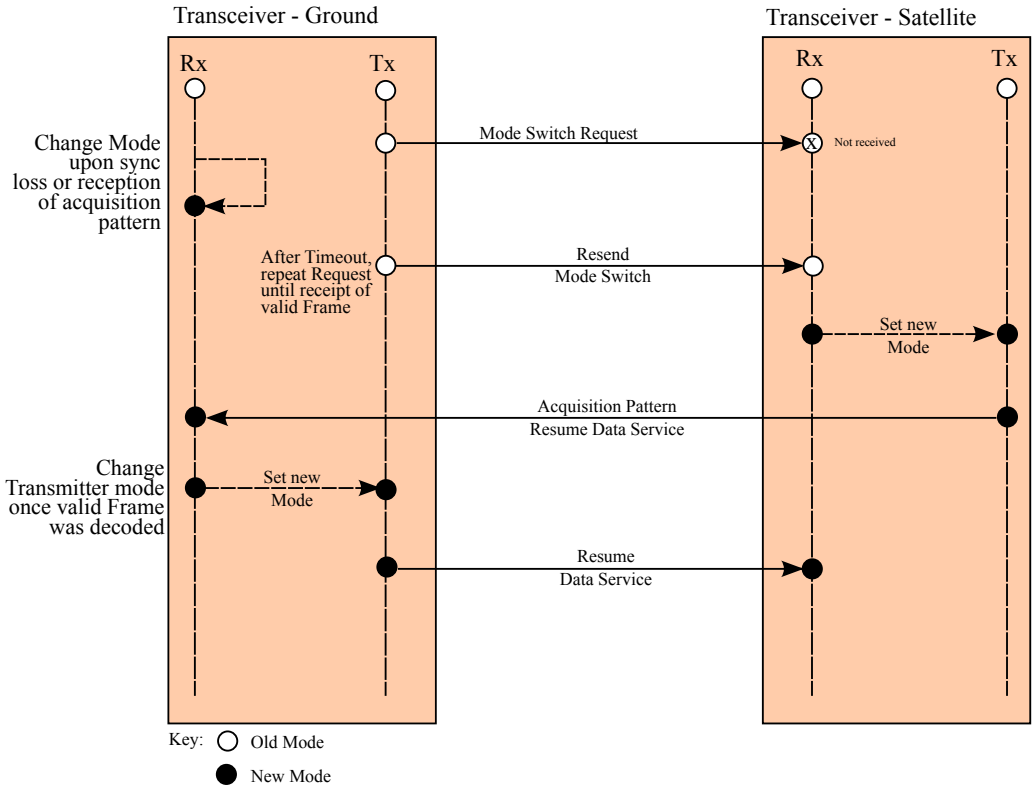


Figure 6: Parameter switching procedure

At the beginning, satellite and ground are assumed to be successfully connected. Ground control

transmits a mode switch request frame to the satellite containing the new parameters (e.g. FEC method, code rate, etc.). The frame contains no payload data. Afterwards the transmitter radiates an idle pattern (i.e. an alternating sequence of 1 and 0). In our example, the satellite did not receive the mode switch correctly and therefore continues operation without change. Upon carrier or synchronization loss or the reception of a sufficiently long acquisition sequence, the ground receiver applies the new parameters and performs the mode switch. After a timeout, the ground transmitter interrupts the idle pattern and repeats the mode switch request. Upon arrival at the satellite receiver, the new parameters are applied to both receiver and transmitter. The satellite transmitter resumes operation by sending the acquisition pattern followed by codeblocks. The first frame in the first codeblock is a non-sequence controlled frame containing a poll request. The ground receiver synchronizes with the acquisition sequence and receives the valid poll frame. Subsequently, the transmitter executes the mode switch. Afterwards, it resumes operation by transmitting the acquisition pattern followed by codeblocks. The first frame sent contains a STAT message as response to the POLL.

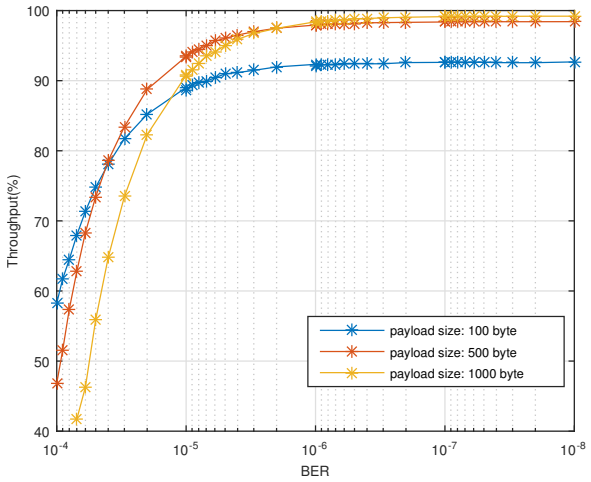
The advantage of this method is its robustness to signal impairments or losses of the physical connection e.g. due to line-of-sight obstructions. Mode switches are persistent and affect all following connections. The reason behind this is that mode switches are intended to allow for changes in the mission requirements or adjustments to changes in the environment. Thus, switching modes for every connection is not desirable.

3 Performance

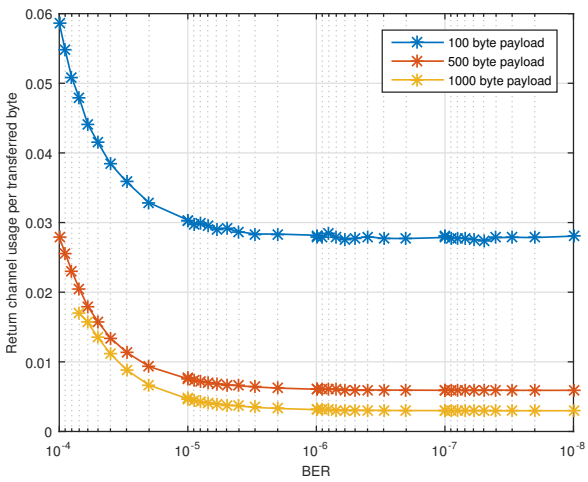
To demonstrate the performance of Nanolink, a simulation of the expected throughput at different bit error rates (BER) was performed. The simulation assumed a window size of 20 frames, a reasonably small size suitable for FPGA or microcontroller implementations. This assures a result close to the real application on small satellites. Figure 7a shows the results of this simulation for frames containing 100, 500, and 1000 bytes of payload each. The throughput is shown in percent of payload data relative to the total transferred data on the link layer.

Due to the required constant size fields, i.e. sync pattern, header, and checksum, the maximum reachable throughput depends on the size of the actual payload per frame. For 100 bytes of payload the maximal theoretical throughput is 92.6%, for 500 bytes of payload 98.5%, and 99.2% for 1000 bytes of payload. The simulation demonstrates that these ideal values are almost reached, even though the POLL and STAT extension headers add some extra overhead. Disabling the extension headers and adding all the required field to the general header would drastically reduce the maximal reachable throughput, as most of the time the additional header fields would be transferred, but are actually unused. Obviously reducing the header size further would increase maximum throughput even more, but cannot be done as the header does not contain any information that is not required within each single frame.

Another issue is the bandwidth occupation of the reverse channel. Depending on the BER, more frames need to be requested for retransmission, thus putting a load on the reverse channel. The load on the reverse channel is illustrated in figure 7b. On the horizontal axis is the BER on the forward channel, on the vertical axis is the number of bytes required on the reverse channel, required to transmit one byte on the forward channel. It is assumed that the ARQ information is incorporated in frames that carry payload data. Thus, only the overhead of the ARQ information is included, since the frames would have been sent nevertheless. It can be seen, that the protocol can work with very



(a) BER vs throughput for different frame sizes



(b) BER vs reverse channel occupation

Figure 7: Simulation results

asymmetric channels, since the load on the reverse channel is very low, despite the small window size, so that asymmetries of 1:100 are possible. Larger window sizes would reduce the bandwidth on the return channel even further.

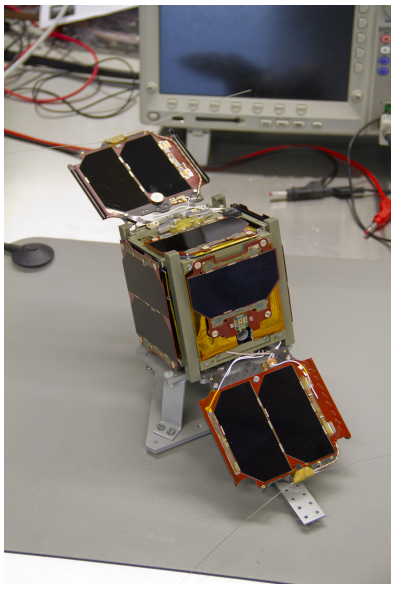
4 Future Work

Nanolink will be implemented on the radio hardware of the TDP-3[1] experiment for Bexus 22[2], which will launch in October 2016. This first experiment is expected to yield important data on the performance in a near space environment. Following this, Nanolink will also be used for the communication of the MOVE-II satellite[3]. The Munich Orbital Verification Experiment (MOVE) satellite program of the Technical University of Munich was initiated in 2006. The main goal of the program since then has been the hands-on education of undergraduate and graduate students. The programs' first CubeSat, called First-MOVE (fig. 8a), was launched in late 2013 and operated in space for a month[3]. Currently the second satellite, called MOVE-II (fig. 8b), is under development[10]. Again designed as a 1 Unit CubeSat (10x10x10 cm^3), with a total mass of 1.3 kg it is planned to launch MOVE-II in early 2018 into a 500-550 km sun-synchronous orbit (SSO). The satellite will incorporate a UHF/VHF transceiver for telemetry and low-to-mid data rate transmission. Furthermore, an experimental S-Band transceiver will enhance the capabilities of the satellite towards higher data rates. The achievable data rate on the S-Band link is expected to be more than 1 MBit/s. It is planned to use the downlink capability in both bands for payload data transmission.

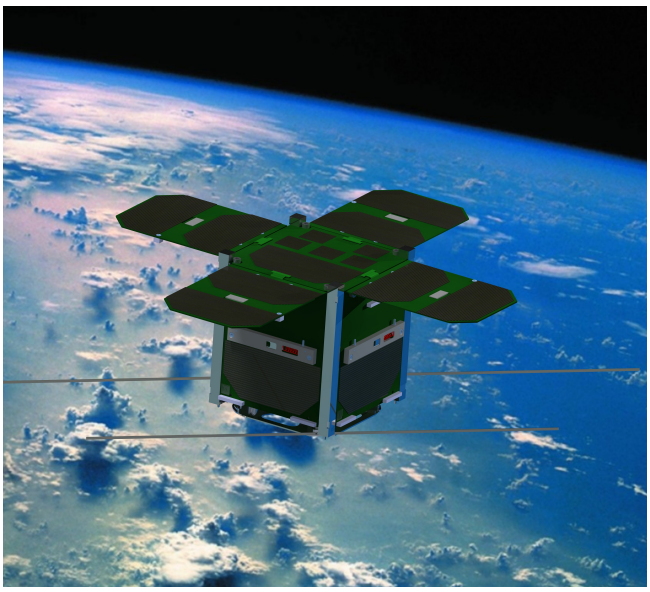
An independent implementation for the software-defined-radio framework GNURadio[4] is in progress and will allow the construction of easy-to-use, cost-efficient ground stations.

Furthermore, we are working on an RFC-like specification of the protocol. We believe that the harmonization of nanosatellite communications will facilitate shared ground station resources and thus

[1] http://www.spacedetectors.de/
[2] http://rexusbexus.net
[3] http://www.move2space.de
[4] http://gnuradio.org/



(a) First-MOVE



(b) MOVE-II (artist's impression)

Figure 8: Images of the two MOVE satellites

improve satellite operations. With this context, we will also focus on the inter-satellite communication capabilites of the protocol and multi-groundstation setups and will develop the required extensions.

5 Conclusion

In the past, most CubeSats used AX.25 for communication, which is inherently unreliable. Hence, we developed a new protocol which offers a high degree of reliability and efficiency. A hybrid ARQ scheme based on the existing protocols SSCOP and STP is used to provide robustness to noisy channels. An extensible header structure is used to reduce the overhead which is introduced by the ARQ mechanism and other protocol features. Simulations show that the protocol remains very efficient, despite high BER on the channel. It also performs well on asymmetric links, which is important since CubeSats usually exhibit a high degree of asymmetry between up- and downlink bandwidth.

We envision Nanolink to be used by other CubeSat missions for TT&C and data transmission. Future research will focus on the use of Nanolink for inter-satellite connections and multi-ground station setups. The protocol will first be used on the TDP-3 experiment onboard of BEXUS 22, and on the MOVE-II satellite of TUM.

Acknowledgments

The authors acknowledge the funding of MOVE-II by the Federal Ministry of Economics and Energy, following a decision of the German Bundestag, via the German Aerospace Center (DLR) with funding grant number 50 RM 1509.

The REXUS/BEXUS program is realized under a bilateral agency agreement between the German

Aerospace Center (DLR) and the Swedish National Space Board (SNSB). The Swedish share of the payload has been made available to students from other European countries through a collaboration with the European Space Agency (ESA).

The authors like to thank Christian Fuchs for his valuable feedback during the conception of Nanolink.

References

[1] B. Klofas, "CubeSat Radios: From Kilobits to Megabits," in *Ground Systems Architecture Workshop*, 2014.

[2] "AX.25 Link Access Protocol for Amateur Packet Radio," https://www.tapr.org/pdf/AX25.2.2.pdf, 1998, revision 2.2.

[3] M. Langer, C. Olthoff, J. Harder, C. Fuchs, M. Dziura, A. Hoehn, and U. Walter, "Results and lessons learned from the cubesat mission first-move," in *Small Satellite Missions for Earth Observation*, R. Sandau, H.-P. Roeser, and A. Valenzuela, Eds. Berlin, Heidelberg: Springer, 2015.

[4] S. Busch, P. Bangert, S. Dombrovski, and K. Schilling, "Uwe-3, in-orbit performance and lessons learned of a modular and flexible satellite bus for future pico-satellite formations," *Acta Astronautica*, vol. 117, pp. 73 – 89, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0094576515003185

[5] J.-L. Issler, A. Gaboriaud, F. Apper, A. Ressouche, D. Evans, O. Koudelka, P. Romano, M. Unterberger, T. Dehaene, B. Lechvalier, G. Guillois, and M. Fernandez, "CCSDS Communication products in S and X Band for Cubesats," in *Proceedings of the AIAA/USU Conference on Small Satellites*, 2014.

[6] "Proximity-1 Space Link Protocol — Data Link Layer," Blue Book, Consulative Committee for Space Data Systems (CCSDS), Newport Beach, CA, Recommended Standard 211.0-B-5, Dec. 2013, issue 5. [Online]. Available: http://public.ccsds.org/publications/archive/211x0b5.pdf

[7] T. R. Henderson and R. H. Katz, "Satellite Transport Protocol (STP): An SSCOP-based Transport Protocol for Datagram Satellite Networks," in *2nd International Workshop on Satellite-based Information Services*. Berkeley: University of California in Berkeley, 1997.

[8] "B-ISDN ATM adaptation layer - Service specific connection oriented protocol (SSCOP)," ITU-T Recommendation Q.2110, Geneva, Standard, 1994.

[9] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460 (Draft Standard), Internet Engineering Task Force, Dec. 1998, updated by RFCs 5095, 5722, 5871, 6437, 6564, 6935, 6946, 7045, 7112. [Online]. Available: http://www.ietf.org/rfc/rfc2460.txt

[10] M. Langer, N. Appel, M. Dziura, C. Fuchs, J. Gutsmiedl, M. Losekamm, D. Meßmann, T. Pöschl, and C. Trinitis, "MOVE-II - der zweite Kleinsatellit der Technischen Universität München," in *Deutscher Luft- und Raumfahrtkongress 2015*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., Bonn, 2015.