Analysis and Implementation of DVB-S2 in the UHF Band for STRATHcube Downlink Communications

***Abstract*—This paper outlines the downlink system design for STRATHcube. Performance and link analyses were conducted, analysing communication windows over the course of the mission and expected theoretical performance. The design was implemented in hardware using MathWorks HDL Coder and C++ code developed for packet handling in software. Software is being created for a basic ACM router and fragmentation using Generic Stream Encapsulation (GSE).**

Initial performance analysis was conducted in comparison to a typical system which uses Constant Coding and Modulation (CCM) and optimises for maximum availability and shows a significant uplift in data throughput over the course of the mission. Additionally, resource analysis of the target FPGA SoC and the implemented design, as well as timing analysis show that the system will be implementable in hardware.

***Index Terms*—CubeSat, Communications, DVB-S2, Software Defined Radio**

I. Introduction

A. CubeSat Communications

Efficient downlink of recorded telemetry is a critical challenge in CubeSat missions, constrained by power limitations, bandwidth restrictions, and dynamic channel conditions imposed during a ground station pass. Further, many CubeSats use the Ultra High Frequency (UHF) amateur band which introduces the further issue of in-band interference which cannot be accounted for in advance.

To maximise throughput under these conditions, Adaptive Coding and Modulation (ACM) must be used. Digital Video Broadcasting – Satellite Second Generation (DVB-S2) is one such ACM system, with near Shannon limit performance and a modular standard allowing it to be matched to the use case. Despite this, there are no space rated UHF DVB-S2 transmitters currently on the market which necessitates its implementation on a Software Defined Radio (SDR). While integrating an SDR can be costly and complex for many missions, the design of STRATHcube already includes one, minimising additional expenses.

In [1] a design for an SDR based communication system in the 915 MHz UHF band leveraging ACM is presented. The system used similar modulation to DVB-S2, with a different coding method. Their analysis showed an almost doubling in throughput compared to CCM. This proved that ACM systems for CubeSats are feasible and offer large performance benefits.

Thanks are given to the sponsors of the STRATHcube, without whom this project would not be possible: The University of Strathclyde Alumni Fund, the Institute of Mechanical Engineers, the Royal Aeronautical Society, the University of Strathclyde Mechanical and Aerospace Engineering Department, and the University of Strathclyde Aerospace Centre of Excellence.

B. STRATHcube Mission

Given the constraints and trade-offs involved in telemetry downlink design, particularly in small satellite missions, emerging CubeSat projects increasingly explore novel approaches to balance performance, complexity, and cost.

STRATHcube is a 2U CubeSat developed at the University of Strathclyde which is part of the *ESA Fly Your Satellite! Design Booster* programme. It has two payloads, each targeting a key area of space sustainability.

The primary payload is a technology demonstrator of a Passive Bistatic Radar (PBR) using an SDR and communicating in the UHF band. Consequently, there will already be a powerful System on Chip (SoC) Field Programmable Gate Array (FPGA) based SDR included on the mission allowing a DVB-S2 transmitter to be implemented as a “piggy-back“ on the primary payload using spare resources.

The secondary payload aims to measure the aerothermal effects leading to solar panel fragmentation by recording and transmitting sensor data during re-entry. Due to this, direct communications with a ground station is infeasible and are instead conducted using the Iridium communications network.

C. Mission Phases

The mission is structured into distinct phases, each imposing unique constraints and opportunities on the communication architecture.

- **Early Operations & Commissioning (EOC) (10 days):** Deployment from International Space Station (ISS) at 415 km, system activation
- **Primary Operations Phase (180 days):** Alternates PBR measurements with UHF downlinks until 170 km altitude
- **Transition Phase:** Reconfiguration for re-entry
- **Secondary Phase:** Re-entry data collection and downlink via Iridium

UHF communications take place during the EOC phase, Primary Operations Phase and Transition phase, driving the need for a robust communication system.

II. Objectives

This work focused on STRATHcube’s downlink system design, with objectives to:

- Develop detailed downlink system design

- Build and validate an engineering model using development boards

The scope was limited to downlink design (excluding uplink), with hardware implementation as an optional goal. Due to this, the implementation was designed for modularity in order to preserve flexibility for future development.

III. Adaptive Coding and Modulation Analysis

Proof-of-concept analysis was conducted to assess whether the performance gains of Adaptive Coding and Modulation (ACM) justified its implementation complexity. The initial analysis assumed a fixed 409 km orbit altitude, propagated over the 180-day mission duration using ISS TLE data through MATLAB's Satellite Communications Toolbox. [2] As the altitude of the satellite lowers over time, the slant range lowers and throughput increases, therefore the analysis provides a rough lower bound on data throughput for the mission.

The methodology comprised three key steps:

1) **Data Rate Precomputation**: Maximum achievable data rates were calculated for discrete 5 °elevation increments (0 °– 90 °) using link budget calculations and required carrier-to-noise ratios from [3], as shown in Fig. 1.
2) **Two-Phase Simulation**:
   - Coarse orbital analysis (100 s timestep) identified ground station access windows via Toolbox `accessIntervals()` function
   - Fine-grained propagation (1 s timestep) generated precise elevation-angle time histories. A histogram of which is depicted in Fig. 2.
3) **Performance Analysis**: Time-at-elevation statistics (binned at 5 °resolution) were combined with precomputed rate tables to estimate total ACM throughput. The CCM baseline scenario assumed conservative operation at 10 °elevation capacity throughout all passes.

The ACM system achieved over 4 Gb of data transfer over the course of the mission, whereas the CCM strategy achieved only 2.4 Gb. These results suggest a 69% increase in throughput over the course of the mission, indicating that the performance uplift is worth the increased complexity requirements. Additionally, if the ground station is located at a lower latitude, higher order modulations could be used and the performance further increased relative to CCM.

It is worth noting that interference was not considered in this analysis and is a major factor in planned UHF band, further research will be required to identify the impact of this.

IV. System Design and Implementation

Following the ACM analysis, it was decided to implement a DVB-S2 compliant system with ACM capabilities. As flight hardware could not be sourced for this investigation, the target platform was a combination of a Digilent Ze dBoard Zynq 7020 development board attached to a FMCOMMS AD936x series development board. As the interface of the AD936x series transceivers are the same, the system could be verified with a different transceiver in the same family.

TABLE I
STRATHCUBE COMMISSIONING PHASE LINK BUDGET

Name	Adverse	Nominal	Favourable
Altitude	409 km		
Elevation	10 °	20 °	40 °
Slant Range	1463 km	1001 km	611 km
FSPL	148.5 dB	145.2 dB	140.9 dB
CNR	10.9 dB	14.1 dB	19.1 dB
Highest Achievable MODCOD w. 10 dB Margin	QPSK 2/5	8PSK 3/5	16PSK 3/4
MODCOD CNR Required	-0.3 dB	4.0 dB	9.0 dB
Bitrate	115.6 kbps	217.8 kbps	386.2 kbps

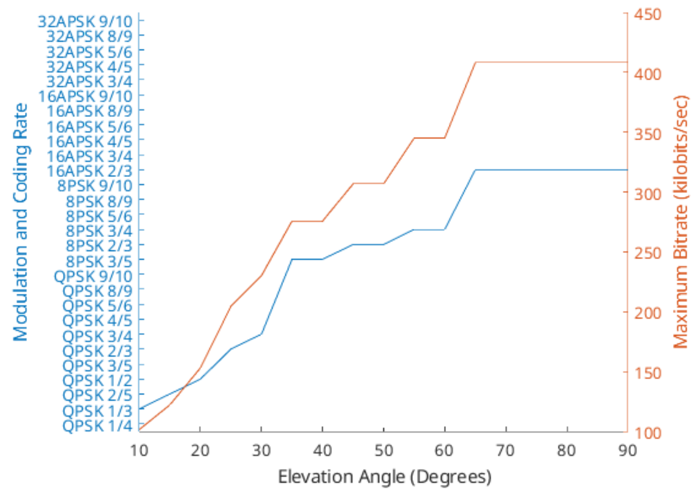

Fig. 1. Highest throughput by elevation angle. The highest achievable modulation and coding rate is shown in blue, with the corresponding throughput in orange.

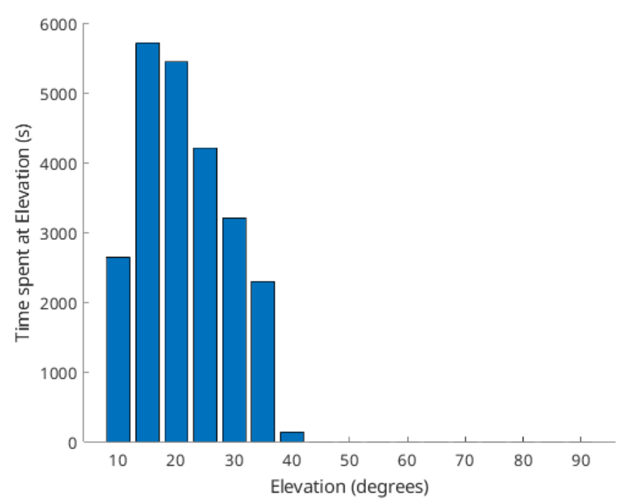

Fig. 2. Histogram of time spent at each simulated pass elevation, binned into 5 °increments. The majority of time is spent in the range of 15 to 25 °.

The designed system had a modular structure, allowing unit testing of each component and facilitating changes in the future as the satellite is developed further.

1) **Processing System**:
   - **System Interface**: Aggregate packets from all external input sources.
   - **Packet Parser**: Parse the input packets to determine their priority level.
   - **ACM Router**: Assign packets into buffers according to Quality of Service (QoS) requirements and encapsulation using Generic Stream Encapsulation (GSE) for efficient framing.
   - **PS-PL Interface**: Break up packets for transfer using an Advanced eXtensible Interface (AXI) and manage control signals for the DVB-S2 Transmitter system.
2) **Programmable Logic**:
   - **DVB-S2 Interface**: Manage interface from DVB-S2 transmitter block to PS AXI-Stream interface.
   - **DVB-S2 Transmitter**: Manages framing, coding, modulation and filtering at baseband.
   - **AD936x Controller**: Manages interface with transceiver.
3) **External Hardware**:
   - **AD936x Transceiver**: Upconversion and transmission of signal.

The FPGA implementation was accomplished using MathWorks HDL Coder [4] due to ease of implementation and ease of testing. Further, using the *Hardware Software Codesign* [5] methodology, both the transceiver interface and PS - PL interface bindings could be automatically generated.

A reference DVB-S2 HDL Coder implementation by MathWorks [6] was selected to reduce technical risk, as it was pretested, allowing faster development of the rest of the system. Further logic was added around this block to manage the AXI-Stream interface and to convert the output for the 12 bit DAC interface. The transmitter portion is shown in Fig. 3.

The implementation of packet handling is yet to be completed, although relevant libraries have been identified and work begun on a C++ implementation of GSE. Further, the packet handling system shall be designed to work with XTCE schemas for packet definition to reduce the difficulty of modification as the satellite design is updated.

V. Simulation & Implementation Results

MATLAB code was created to generate synthetic AXI packets to test the system in Simulink. The resulting output spectrum was then compared against the ITU out of band emissions mask for the amateur and amateur satellite service [7], as shown in Fig. 4.

The system was then exported and an IP block design generated using Vivado. The resource usage was then analysed by subsystem and resource type as shown in Fig. 5. Following this, a timing report was generated, finding a worst negative slack of 0.126 ns, a worst hold slack of 0.016 ns and a worst pulse width slack of 0.264 ns.

VI. Discussion

The simulated output spectrum successfully passes the ITU spectral mask, meaning that it complies with emissions requirements. This requires further testing in hardware to verify that the implemented design aligns with simulation results.

For most resource types, there is enough below 50% usage, leaving plenty for the PBR implementation, however the transmitter design uses almost all of the Block RAM available on the device, which would severely restrict the PBR subsystem. The Block RAM usage of the final system will require optimisation in the future, as well as analysis of the PBR implementation to identify the required resources.

The timing report showed that the system is implementable in hardware, however the margins are quite slim. This may be related to how spread out the implemented design was on the silicon, as so many Block RAMs were required. An optimisation of resource requirements could therefore improve the timing slack available.

VII. Conclusion

The benefits of ACM for the mission has been analysed using orbital simulation and link budget analysis, indicating that a substantial uplift in throughput could be possible. With this in mind, the system architecture was designed, with all key blocks identified for packet handling and transmission using DVB-S2. A transmitter was created building upon a MathWorks reference implementation using MathWorks HDL Coder. This hardware implementation was then synthesised using Vivado and found to meet both resource and timing requirements.

The packet handling systems are still to be implemented, although work has begun on GSE.

References

[1] E. Grayver, A. Chin, J. Hsu, S. Stanev, D. Kun and A. Parower, "Software defined radio for small satellites," 2015 IEEE Aerospace Conference, Big Sky, MT, USA, 2015, pp. 1-9, doi: 10.1109/AERO.2015.7118901.

[2] MathWorks, Satellite Communications Toolbox (R2024b). (2024). The MathWorks Inc., Natick, Massachusetts, United States. [Online]. Available: https://www.mathworks.com

[3] ETSI, EN 302 307-1 - V1.4.1 - Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broa dBand satellite applications; Part 1: DVB-S2, EN 302 307-1, Jul. 2014.

[4] MathWorks, HDL Coder (R2024b). (2024). The MathWorks Inc., Natick, Massachusetts, United States. [Online]. Available: https://www.mathworks.com

[5] MathWorks, 'Hardware-Software Co-Design Workflow for SoC Platforms', The MathWorks Inc., Accessed: Mar. 31, 2025. [Online]. Available: https://uk.mathworks.com/help/hdlcoder/ug/hardware-software-codesign-workflow-for-soc-platforms.html

[6] MathWorks, DVB-S2 HDL Transmitter. (2024). MATLAB. The MathWorks Inc. Accessed: Dec. 11, 2024. [Online]. Available: https://uk.mathworks.com/help/satcom/ug/dvbs2-hdl-transmitter.html

[7] ITU, Unwanted emissions in the out-of-band domain, SM.1541-7, Sep. 2024.

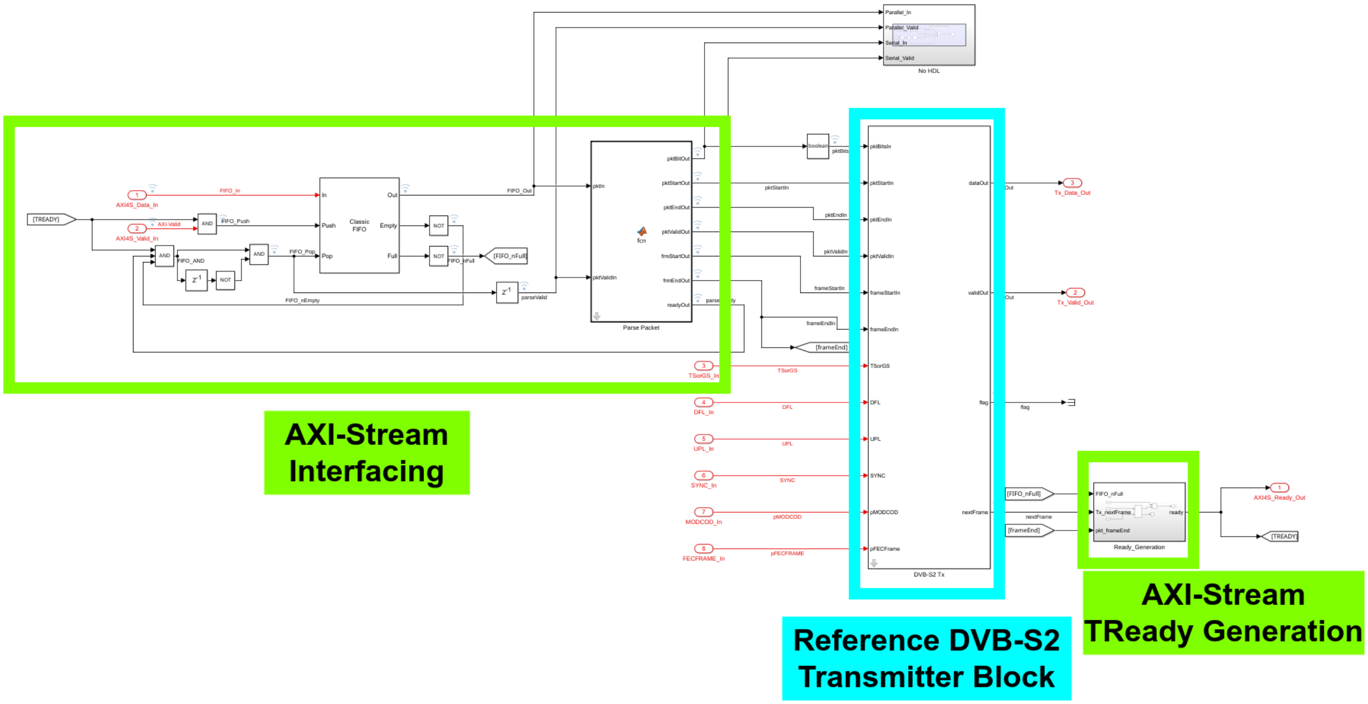

Fig. 3. MathWorks HDL Coder transmitter blocks. Highlighted in green are the blocks used for the AXI-Stream interface. In Cyan is the reference DVB-S2 transmitter block created by MathWorks

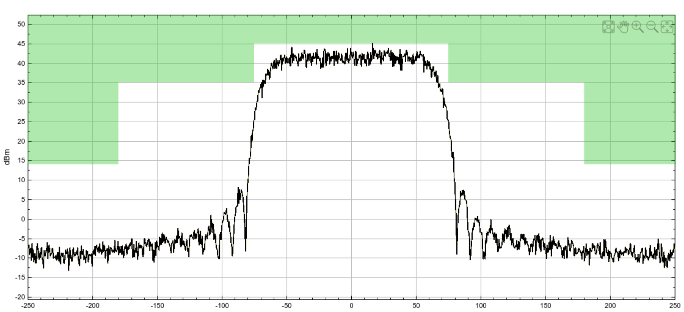

Fig. 4. Simulated output spectrum with ITU spectral mask. The mask is shown as the coloured section at the top, with green indicating that the spectrum is passing. Note that the power axis is not representative of the actual output power, as this has been scaled for interfacing with the DAC.

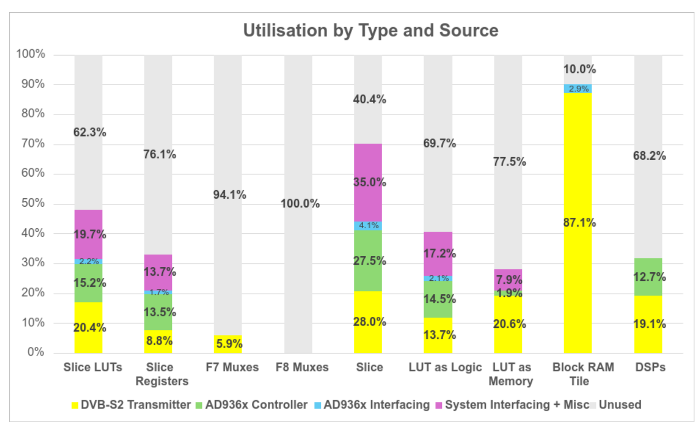

Fig. 5. Implemented design utilisation by resource type and subsystem. Most resource types are below 50% usage, with the exception of Block RAM, of which only 10% is free.