IMPLEMENTATION OF A HDL-CODER BASED TELECOMMAND RECEIVER APPLICATION FOR MICROSATELLITE COMMUNICATION

Jan Budroweit, Ferdinand Stehle, Christopher Willuweit[2], Dirk Wübben[2]

German Aerospace Center (DLR), Institute of Space Systems, 28359 Bremen, Germany
[2]Department of Communications Engineering, University of Bremen, 28359 Bremen, Germany

ABSTRACT

In this paper the development and implementation of a Telecommand (TC) receiver application for microsatellite communication is presented. The TC receiver application is executed and operated by a highly integrated Generic Software-Defined Radio (GSDR) platform. This platform architecture is designed for the reliable operation of multiple radio frequency applications on spacecraft. For the development and implementation process of the TC receiver application, a new model-based development workflow by Matlab/Simulink is used and evaluated.

Index Terms— SDR, Model-based design, VHDL, Telecommand, Microsatellite communication

1. INTRODUCTION

The demand of scalability and flexibility has been increased for microsatellite missions in the past decade and there are many activities on-going to provide satellites with a tall envelope in terms of provided power, computing resources and communication links [1][2]. A proposed solution for a flexible and scalable communication system is the Generic Software-Defined Radio (GSDR) of the German Aerospace Center, which provides the operation of multiple radio frequency (RF) applications on a spacecraft, almost without frequency band limitations [3]. To improve the flexibility and reduces the development and verification time of applications for the GSDR, model-based design workflows are promising, which for example are offered by Mathworks with their Matlab and Simulink tools [4]. In this paper the Mathsworks model-based design workflow is used to develop, implement and test a Telecommand (TC) receiver application on the GSDR platform architecture.

In section 2, the system overview, including the application related requirements and important design constraints are presented. Section 3 describes the model-based design workflow and the GSDR system architecture. The implementation of the TC application is presented in section 4 and the associated validation and test results are discussed in section 5. The final conclusion is given in section 6.

2. SYSTEM OVERVIEW

For satellite communication it is often mandatory to be compliant with standards and recommendations of the Consultative Committee for Space Data Systems (CCSDS). Major reason is that many ground stations are following their recommendation and thus, the communication link related specifications. For communication subsystems on a spacecraft it is often only required to cover and handle the signal processing related to the physical layer of the ISO/OSI reference model. In particular, the communication subsystem is performing the analog to digital conversion (and vice-versa), the modulation/demodulation and related RF signal processing (e.g. filtering, amplifying and RF converting). Additional signal processing, like coding and encoding, is then performed by the on-board computing and data handling subsystem. For the proposed TC receiver application in this paper, part of the data link layer relevant signal processing (decoding) is also performed. In the following Tab. 1, the requirements and application specifications are presented.

Tab. 1: TC receiver application related requirements and specifications

Parameter	Value
Carrier frequency	2081.2MHz
Occupied bandwidth	153kHz
Doppler offset	+/-65kHz
Doppler rate	< 1kHz/s
Modulation	BPSK
Data rate	64kBit/s
Coding	Expurgated BCH (63,56)
PLOP-Mode	PLOP-2

The specifications have been selected from a reference mission of the Germany Aerospace Center, Institute of space system, which covers almost the major mission requirements. The selected frequency band is S-band in the uplink (2081.2MHz carrier frequency). Due to a selected modulation scheme of Binary Phase Shift Keying (BPSK) and a data rate of 64kBit/s, an occupied bandwidth of approx. 153kHz is mandatory. For error detection and correction purposes, an expurgated Bose-Chaudhuri-Hocquenghem (BCH) code is used. Fig. 1 shows the signal

processing flow for the TC application and illustrates relevant signal processing parts which are required to implement.

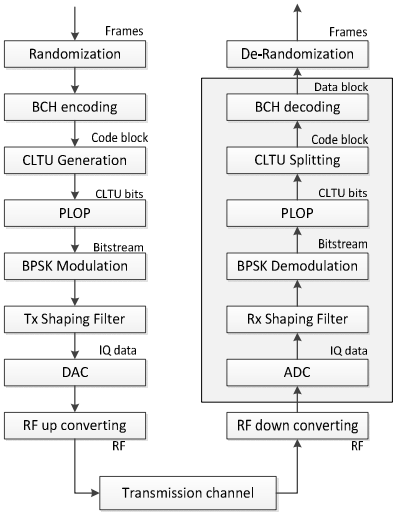

Fig. 1: Transmission flow. The gray highlighted bocks are part of the TC receiver application

The data to be transmitted are organized in frames. A frame contains information about the addressing, sequencing, user data and error detection. Fig. 2 shows the structure of a transmission frame.

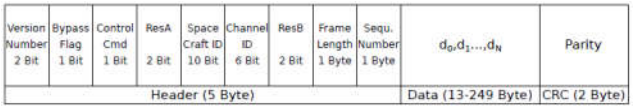

Fig. 2: Structure of the transmission frame

One frame includes a header with a length of five byte, up to 249 byte of user data, as well as a CRC-Checksum with a length of two byte. One or multiple frames are transmitted inside of The Communication Link Transmission Unit (CLTU). Beginning with the acquisition and IDLE sequence (a series of 0x55 byte), the starting sequence (0xEB90) flags the start of CLTU. Afterwards, the segmented block codes and error corrected frames are following. The maximum length of the CLTU is specified by each mission.

The channel coding adds redundancy in terms of control bits to the user data. Thus, errors in the transmission could be detected and corrected. The selected expurgated BCH code is derived from a (63,56) hamming code and could detect 2 bits and is able to correct 1 bit per code block [5].

The Physical Layer Operation Procedure (PLOP) specifies the order of states during a CLTU transmission. While at a PLOP-1 configuration the transmitter is sending only the carrier between a CLTU transmission sequences, the PLOP-2 mode specifies a transmission of IDLE frames between CLTU transmissions. Due to the IDLE frame transmission sequence, modulated data are available and the transmission channel can be established. Major advantage of the PLOP-2 mode is that the receiver does not need to re-synchronize every time a CLTU is transmitted and received. On the other hand, the receiver needs to track and recover the carrier frequency (e.g. due to Doppler Effect) and timing of the received signal.

The BPSK is one of the simplest digital modulation schemes and modulates the information into the phase of the carrier signal. In case of a BPSK, only two phase conditions are defined, which are called symbols. Each symbol contains one bit. A series of modulated symbols with a given period is passed through pulse shaping filter to generate a continuous waveform. This shaping filter reduces the required bandwidth and improves the noise sensitivity.

3. DEVELOPMENT ENVIRONMENT AND HARDWARE ARCHITECTURE

In this section the selected model-based design workflow of Mathsworks is briefly described. Additionally, the GSDR system architecture is presented on which the TC receiver application through the model-based design needs to be implemented.

3.1. Mode-based design workflow

Model-based design is a development tool that is based on a model of the target environment. The model represents the environment and the target system and is also the specification, test bed and the fundamental of the prototype. This approach should combine much information, prevent duplications, generated replicable results through automatic processes and allows a detailed focus of the design towards the problem statement [6]. In the beginning, the model is initially designed with the goal of optimizing the desired behavior. In the further process, the model is refined, adapted in terms of implementation ability and constantly tested by simulation. Prototypes based on hardware-in-the-loop simulations lead the model closer to the real applicability. The final goal is a model out of which, through code generation, a real, real-time capable system with equivalent behavior can generate.

3.2. GSDR system architecture

The GSDR system basically consists of two major components, which are relevant for the model-based design implementation workflow.

The signal processing unit is based on a fully programmable Xilinx ZYNQ System on Chip (SoC) which combines an ARM Cortex A9 processor and a Field Programmable Gate Array (FPGA) fabric. Thus, soft-cores representing CPUs inside of an FPGA can be prevent and improve the overall performance. The FPGA is directly connected to the processor and allows seamless dynamic hardware extension from the software perspective of view. The second important system part is the Radio Frequency Integrated Circuit

(RFIC) Transceiver, AD9361, of Analog Devices (ref. Fig.3).

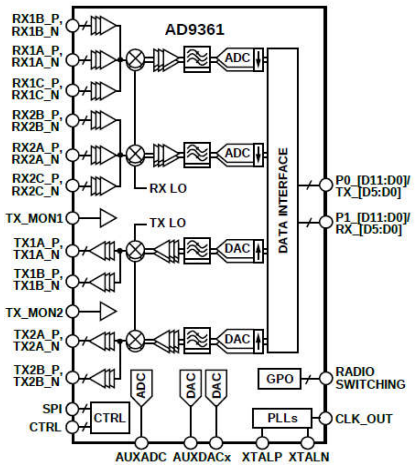

Fig. 3: Schematic if the AD9361 RFIC

The RFIC itself consist of two independent transmitter and receiver chains, sharing a common clock source. The integrated analog digital converter (ADC) and the digital analog converter (DAC) sample rate and the local oscillator control to mix the baseband signal into RF band (70MHz to 6GHz) are provided by this reference clock. For each chain, a stack of amplifiers and filters allows a narrow band selection and RF signal processing. The RFIC can be operated in time division duplex and frequency division duplex mode. Each receiver and transmitter chain consists of selectable (by multiplexing) RF input and RF outputs.
Both devices, the AD9361 and the Zynq SoC are supported by the selected model-based design workflow of Mathworks.

4. IMPLEMENTATION

In this section the receiver components and their implementation are presented. The received signal is passing multiple processing stages for error correction in frequency, phase and timing. The sample rate of the AD9361 is set to 16 times the required symbol rate of 64kBit/s. With the receivers shaping filter and the timing recovery, the sample rate will be reduced to the symbol rate.

4.1. Frequency and phase correction

Since the center frequency of the received signal generally mismatches to the down converted frequency of the receiver, frequency shift compensation is required. Due to the Doppler Effect a frequency offset of +/-65kHz is additionally expected. The offset correction is performed after digitization of the input signal. For coarse frequency offset compensation a Fast Fourier Transformation (FFT) based approach is promising since a phased locked loop is not required (estimation of frequency offset out of calculated spectrum). In a first step, the incoming signal is multiplied by itself to double the frequency. In case of a BPSK-modulated signal, this has a superposition of the two possible phase angles, and thus, a single strong signal at twice the center frequency result. Through the FFT, this signal can be estimated to the power spectral density. With the frequency index at the maximum power level, the offset can be calculated and corrected. The HDL compatible block design for the frequency correction is presented in Fig. 4.

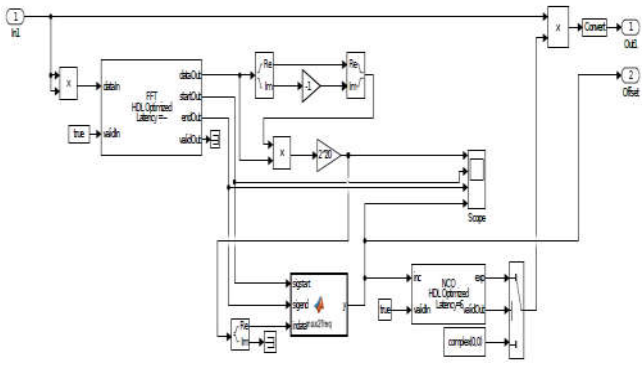

Fig. 4: HDL compatible frequency offset compensation block design

By the transmission of the signal a phase error has to be assumed which causes a constant rotation of the symbols in the constellation diagram. The implemented phase correction is based on a discrete Phased Locked Loop (PLL) [7], in which the deviation is detected by a phase error detector und passes through a loop filter. A controller finally ensures a proportional rotation of the phase.

4.2. Timing recovery

In the timing recovery, the optimum timing for sampling is determined. Since the symbol clock of the transmitter on the receiver side is not explicitly available, it needs to be reconstructed out of the signal. In the constellation diagram, a timing error causes a spread of values by the optimal symbol positions. The timing recovery is designed comparable to a PLL. The timing error detector generates an error value, which is forwarded to a control block after grading with a loop filter.

4.3. Demodulation, decoding and bit error correction

The BPSK demodulator receives the corrected IQ symbols from the timing recovery and determines the corresponding data bits. The demodulator makes a hard decision on the corresponding bit per input value.
The CLTU detection is required to detect the start and stop of a frame in a continuous bitstream and to determine the including code blocks. This is done by parallelizing the incoming bits to a shift register and performing continuous comparing with the start sequence (0xEB90).
With the implemented BCH code, single bits can be detected and corrected. The implementation of the BCH decoder is realized with shift registers, as presented in Fig. 5 on the following page [5].

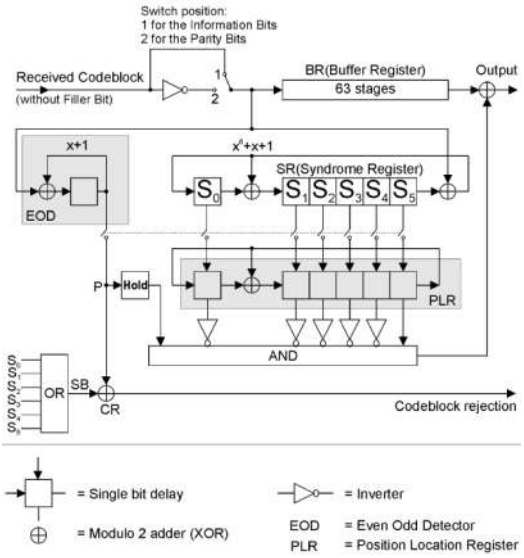

Fig. 5: Implementation on a HDL compatible BCH decoder [5]

5. VALIDATION AND TEST

To test the receiver, a reference transmitter is used, which is part of the Electronical Ground Support Equipment (EGSE) of a DLR satellite mission. The used part of the EGSE represents the ground station for such mission (same specification) and generates the required data and frames for the receiver's evaluation.

Firstly, the receiver is tested without any interference in a hardware-in-the-loop setup, in which IQ-data are captured and then used as input for the HDL-optimized simulation model (Simulink). In a series of simulations, those IQ data are passing through a simulated AWGN channel with different noise energies. The results for uncorrectable, correctable and loss of blocks over Eb/N0 are presented in Fig. 6.

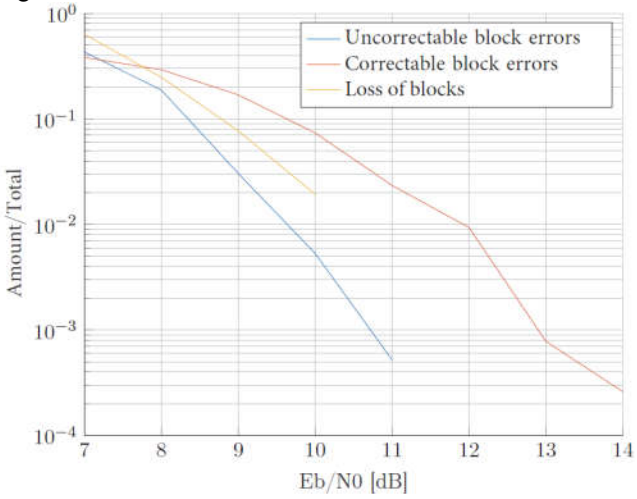

Fig. 6: Block errors with captured IQ data in a HDL-optimized simulation model

The block losses are given relatively to the total number of code blocks being sent. As expected, the errors increase with decreasing signal-to-noise ratio. Below 11dB, the block losses rise very sharply. For the values 13dB and at 14dB, the simulation is performed with twice the number of code blocks being sent in order to be able to resolve the low error rate.

In the next step, the receiver is implemented in hardware and interfaces the EGSE via RF cabling. Through the EGSE, interferences like noise and attenuation are then added to the signal. Additionally, the receivers performance has been evaluated with respect to Doppler shift in (sweep from +/125kHz with 1kHz/s) on different level of signal input power. Results therefore are given in Fig. 7

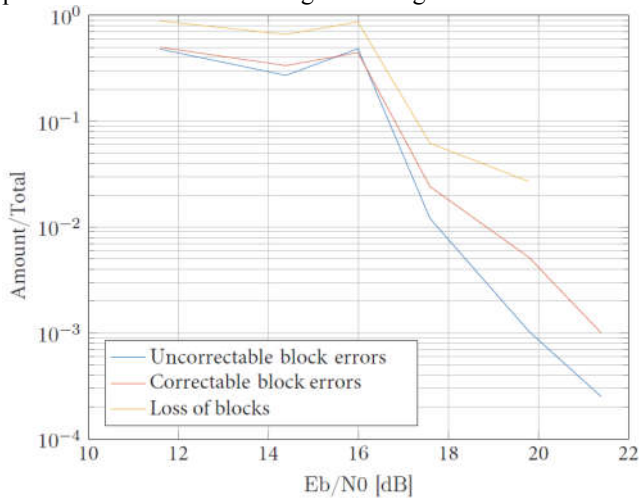

Fig. 7: Measured block errors with implemented hardware connected to EGSE.

The results are showing the different kind of block errors vs Eb/N0. Compared to the simulation results with captured IQ data, a lightly higher Eb/N0 is noted, which is explained by the noise figure of the receiver hardware in the RF input.

6. CONCLUSION

In this paper we presented the development, implementation and verification of a CCSDS compatible receiver application using the model-based design workflow of Mathworks. In principle, the workflow is a very powerful tool that allows rapid development of embedded applications. In practice, it has been shown that this tool requires a lot an additional work, since many functions (for this application) are not provided by Mathworks and needed to be designed and implemented separately. Nevertheless, the mix of auto-code generated and manually written functions to VHDL, out of one system model, was successfully implemented and tested through this workflow and the results are showing good performances.

REFERENCES

[1] F. Dannemann, M. Jetzschmann, “Technology-driven design of a scalable small satellite platform,” *4S Symposium 2016 proceedings*, ESA, Malta, May 2016.

[2] F. Dannemann, M. Jetzschmann, “Scalability and modularity as dimension of flexibility of a microsatellite platform,” *68th International Astronautically Congress 2017, Symposium on small satellite missions (B4)*, IAC, Australia, September 2017.

[3] J. Budroweit, “Design of a Highly Integrated and Reliable SDR Platform for Multiple RF Applications on Spacecraft,” *GLOBECOM 2017 - 2017 IEEE Global Communications Conference,* Singapore, 2017, pp. 1-6. doi: 10.1109/GLOCOM.2017.8255087

[4] D. Pu, A. Cozma and T. Hill, “Four Quick Steps to Production: Using Model-Based Design for Software-Defined Radio (Part 1),” *Analog Dialogue 49-09,* USA, September 2015.

[5] CCSDS, “TC Synchronization and Channel Coding—Summary of Concept and Rationale,” *CCSDS*, 2012.

[6] R. Aarenstrup, “Managing Model-Based Design,” *The MathWorks Inc*., 2015.

[7] M. Rice, “Digital Communications: A Discrete Time Approach,” *Pearson*, 2011.