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, Avionic Systems, 28359 Bremen, Germany, email: jan.budroweit@dlr.de
[2]Department of Communications Engineering, University of Bremen, 28359 Bremen, Germany

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.

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 on the physical layer of the ISO/OSI reference model. 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.

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

Tab. 1: TC receiver application related requirements and specifications

Fig. 1 shows the signal processing flow for the TC application and illustrates relevant signal processing parts which are required to implement.

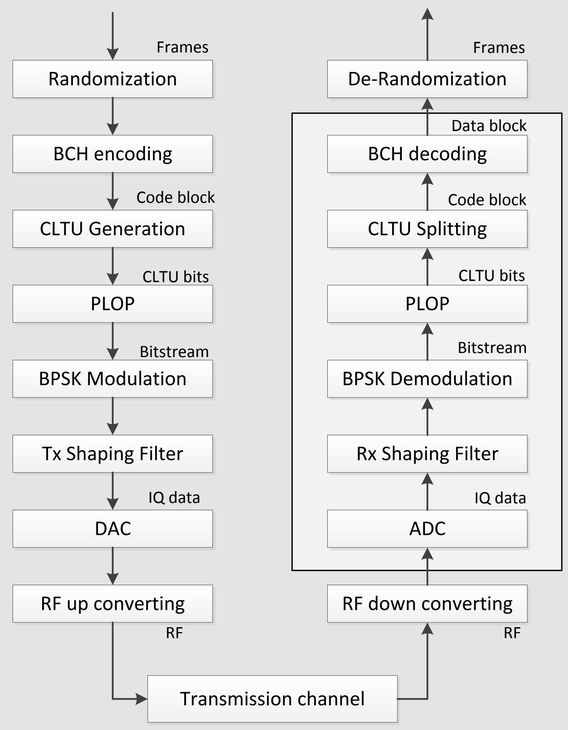

Fig. 1: Transmission flow. The gray highlighted bocks are part of the TC receiver application.

The data to be transmitted are organized in frames. A frame contains information about the addressing, sequencing, user data and error detection. 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 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.

The Physical Layer Operation Procedure (PLOP1/2) specifies the order of states during a CLTU transmission.

On BPSK, only two phase conditions are defined, which are called symbols. Each symbol contains one bit.

Development Environment

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 many information, prevent duplications, generated replicable results through automatic processes and allows a detailed focus of the design towards the problem statement. 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.

The platform for this model-based implementation workflow is a Generic Software-Defined Radio based on a Xilinx ZYNQ SoC.

Implementation

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.

Frequency and phase correction

Since the center frequency of the received signal generally mismatches to the down-converted frequency of the receiver, a frequency shift compensation is required. Due to the Doppler Effect a frequency offset of +/-65kHz is additionally expected. 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.

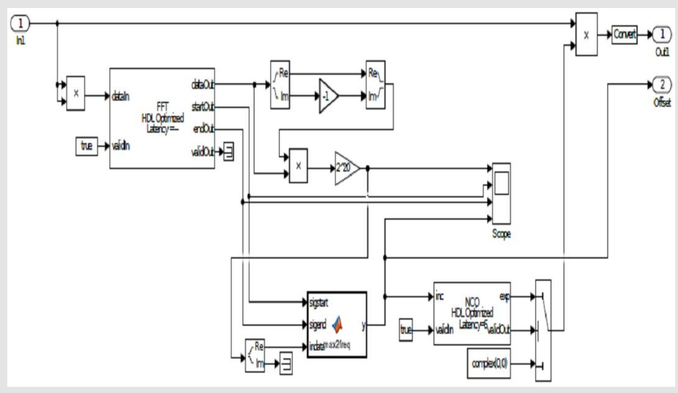

Fig. 2: HDL compatible frequency offset compensation block design

The implemented phase correction is based on a discrete Phased Locked Loop (PLL), 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.

Timing recovery

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.

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.

Validation and Test

To test the receiver, and its TC application, 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. 3.

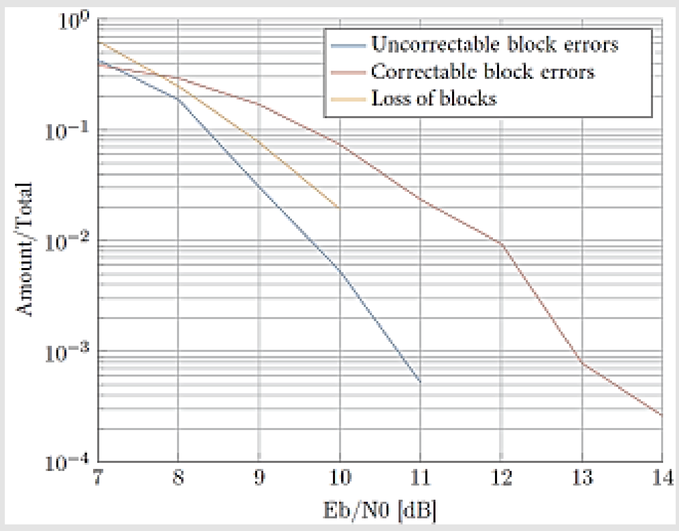

Fig. 3: AUT at the qualification test for random vibration (z-axis)

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. 4

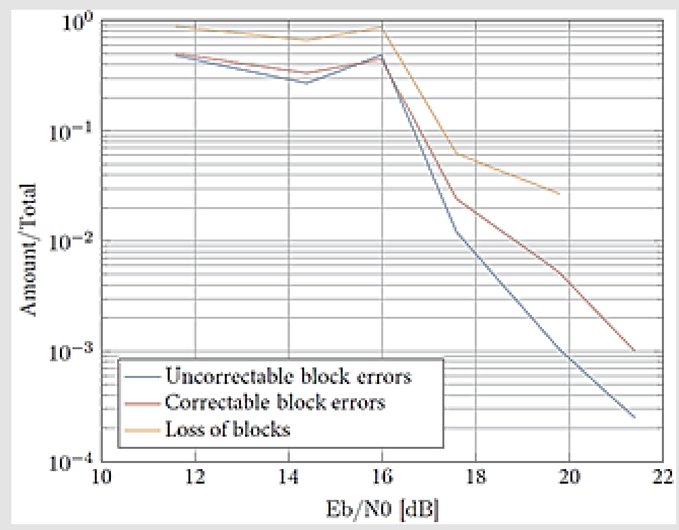

Fig. 4: Measured spectrum of the random vibration qualification test run in z-axis for the AUT.

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.

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.

Universität Bremen
Arbeitsbereich Nachrichtentechnik
DLR Deutsches Zentrum für Luft- und Raumfahrt

DLR at a Glance

DLR is the national aeronautics and space research centre of the Federal Republic of Germany. Its extensive research and development work in aeronautics, space, energy, transport and security is integrated into national and international cooperative ventures. In addition to its own research, as Germany's space agency, DLR has been given responsibility by the federal government for the planning and implementation of the German space programme. DLR is also the umbrella organisation for the nation's largest project management agency.

DLR has approximately 8000 employees at 16 locations in Germany: Cologne (headquarters), Augsburg, Berlin, Bonn, Braunschweig, Bremen, Goettingen, Hamburg, Juelich, Lampoldshausen, Neustrelitz, Oberpfaffenhofen, Stade, Stuttgart, Trauen, and Weilheim. DLR also has offices in Brussels, Paris, Tokyo and Washington D.C.