Enhancement in the Protection of Transmitted Data

Ing. Vítězslav Křivánek,

Department of Telecommunications, Brno University of Technology,
Purkynova 118, 612 00 Brno, Czech Republic

Summary
Nowadays and undoubtedly also in the future, modern communication technologies are marked with a tendency towards constant increase in the speed of data transmission. During such transmissions new problems originate which need to be eliminated. An ever more pressing issue are burst-forming errors caused by increases in the transmission speed. One of the main tasks is to find adequate coding and error correction. The presented article focuses on the enhancement of protection of the transmitted data and on the search for alternative solutions to the correction of burst errors with respect to the existing methods.

Key words:
Forward error correction, convolution coding, burst error, Simulation, Matlab.

1. Introduction

A constantly growing integration of services requires considerable versatility from communication terminal units. Current terminal units must be able to cope with diverse technologies of the access networks. This has the logical consequence of different levels of resistance towards damage to the content of the transmitted messages. The process protecting the digital signal from damage consists in suitable arrangement of the protecting elements. But since the protection is performed by an increase in redundancy at the expense of the actual transmission of the useful information, it is very important to find the most suitable solution which however differs from case to case.

2. Simulation

Evolution of systems dealing with the protection of useful information against transmission errors is closely related with the development of all types of communication networks. At present, security codes are employed in digital networks as a standard and quite systematically. Individual phases of coding and decoding procedures which are required to assure the necessary protection can be very clearly shown by a simulation which facilitates easier understanding of the issue at hand.

There are plenty of programmes appropriate for the simulation of correction codes: WinSpice, Micro-Cap, Matlab system upgrade Simulink. Basic requirements for the programme are stability, extensive element library, user-friendly graphic environment, simple and intuitive control, analysis of potential errors and detailed documentation. For this reason the next demand is mainly laid on simple and intuitive control. The mathematical programme Matlab was selected as the best suited option for the coding process.

Particularly the specialized Simulink libraries, which serve for modelling and simulation of dynamic systems, can be used. Dynamic system models are created interactively in the form of block diagrams and connections between them. Simulink is used for a time solution by way of a simulation of the dynamic system behaviour considered that we know its mathematical description. It enables observation of the variables in almost any part of the formed model [2].

A remarkable advantage of the specified programme lies in the possibilities of graphic representation which are important for learning the function and creation of the security codes which are of considerable complexity. Apart from higher clarity, these better quality security codes presentation methods also enable improving the perspective on the issues in question [1]. Other simulation applications exist but their intuitive control, clarity and arrangement of results is lower than with the selected Matlab mathematical programme.

3. Convolutional codes codec model – parallel connection

Security codes can be divided into two main groups: block and convolutional. Convolutional encoders are characterized as sources of messages with a memory. Hagelbarger was the first to introduce convolutional codes for burst error correction. Later Iwadare and Massey, independently of each other, designed more efficient codes

Manuscript received July 5, 2008
Manuscript revised July 20, 2008

of the same type. Having the same correction capability, their codes require shorter security intervals than Hagelbarger codes. Afterwards, optimal codes for correcting sequential burst errors were independently invented by Berlekamp and Preparate [3]. All above mentioned convolutional codes have a lot in common. These common characteristics can be well utilized in the course of model formation.

Fig. 3.1 shows a general connection block diagram for a simulated codec. Apart from the encoder and decoder blocks, also generator of the input data stream, transmission channel and display of the output data stream blocks are used.

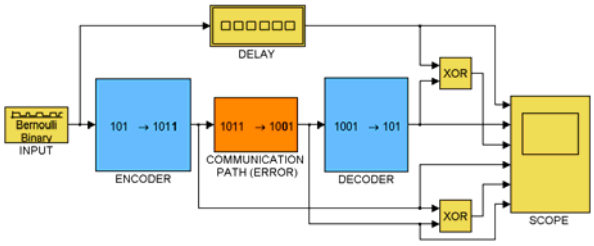

Fig. 3.1: General codec connection block diagram.

Fig. 3.2 shows a general internal connection of an encoder block. It is obvious that it is composed of three sub-blocks. The first subblock is the serial-parallel inverter inverting the input serial data into three parallel data flows for the encoder. In the encoder itself, the input data are equipped with a safety bit and then progress to the block of the parallel-serial inverter that inverts parallel data back to the serial flow suitable for the transmission through the transmission channel.

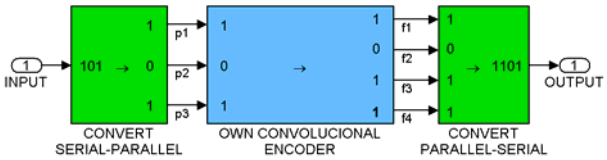

Fig. 3.2: General connection block diagram for an encoder with speed adjustment.

It is visible from Fig. 3.2 that there are two different transmission rates in the encoder block connection. Three parallel bits enter the encoder block which are inverted from the input serial data with the transmission rate v_1 by the serial-parallel inverter. Yet in the output we get four parallel bits inverted back to the serial data with the transmission rate v_2 by the parallel-serial inverter. Therefore the following ratio is valid for the transmission rates: $v_1 / v_2 = 3/4$.

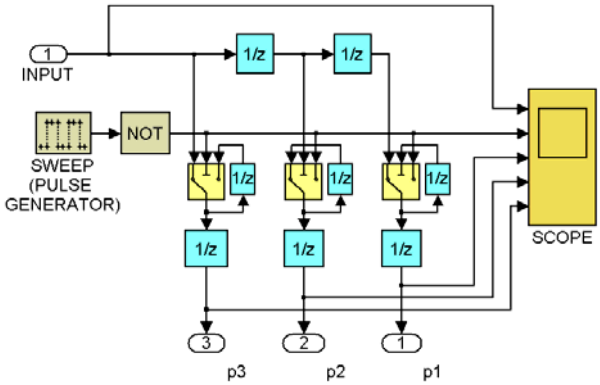

Fig. 3.3: Serial-parallel inverter.

Fig. 3.3 shows the internal diagram of the serial-parallel inverter. The input data with the rate v_1 go to the shift register made of two memory cells. When the third signal element comes to the inverter input, controlled switches are switched using the time-base impulse and then logical values of the memory cells content are sampled. After switching back the switches, the sample thus taken is stored in a loop consisting of the output and the second input of the switch until a subsequent sample is taken. Thus the connection performs the function of a demultiplexer.

The internal connection of the encoder itself is determined by the generator matrix [4]. Three partial parallel signal flows are brought to the encoder input and samples of logical values are taken from them – Fig 3.4. A security bit for the given three input bits is generated by means of mod2 sums represented by XOR (Exclusive OR) blocks and of seven memory cells represented by time lag elements 1/z.

The parallel-serial inverter block shall invert four output bits from the encoder into the serial flow of bits with the rate v_2 suitable for the transmission. Fig. 3.5 shows its internal connection.

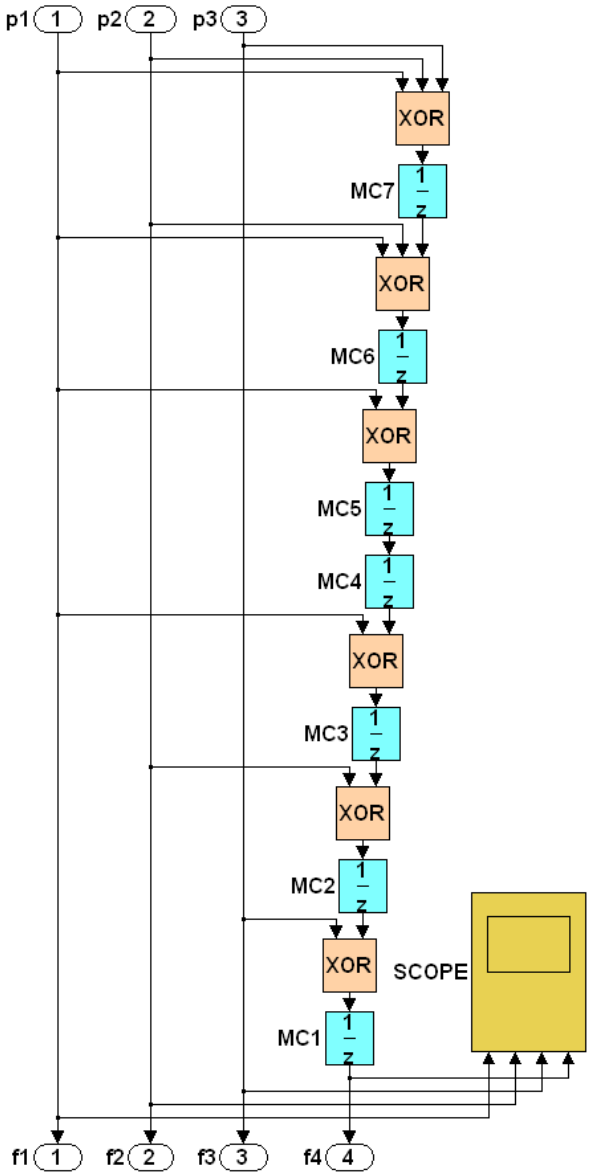

Fig. 3.4: Block diagram of encoder connection Berlekamp – Preparate code.

The inverter function is based on the gradual reading of samples of signals brought to the f1 – f4 inputs. The currently read input is determined by switching the relevant switch controlled by the comparator. The comparator compares the value from the 0 - 3 counter controlled by the time base operating with the rate v_2 and the value of the invariable particular for each input flow. If the invariable value is identical with the value from the counter, the respective switch is switched and the sample goes to the disjunction gate. The output bit flow with the rate v_2 is obtained by the logical sum of partial flows using the four-input gate OR. Due to this procedure, the input flows are multiplexed into the output flow.

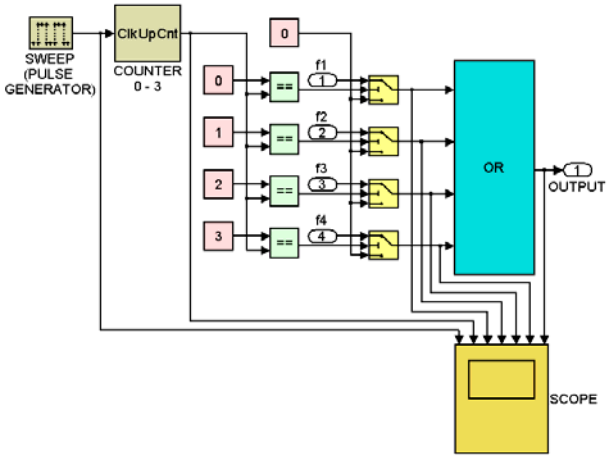

Fig. 3.5: Parallel-serial inverter.

All of the modified data are sent over the transmission channel where damage can occur. Digital transmission is represented by a bit sequence – data transmission segments of constant length and specific level that can have two values (either 0 or 1) if the binary code is used. An error in the data channel can come about due to varied influences but only by a change in the bit value – either from 0 to 1 or from 1 to 0. The error generator block simulates burst errors which can originate in the real transmission channel. The error can be easily modeled by a mod2 adder, or in a more sophisticated manner by the error generator block shown on Fig. 3.6, that adds an error vector. The possible length of a burst error and the distance between errors can thus be simply modified.

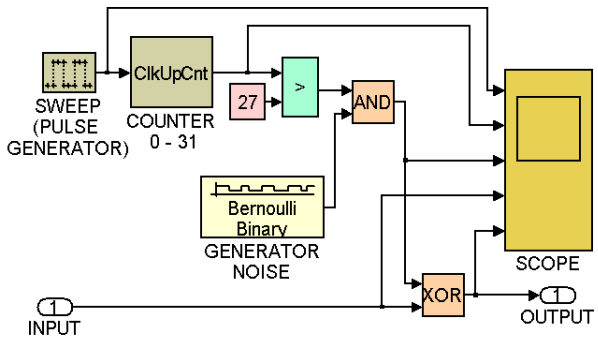

Fig. 3.6: Burst error generator.

The decoder is also composed of three sub-blocks. Serial to parallel converter transforms the input data of v_2 speed into four parallel sub-streams. Error correction takes place in the actual decoder on the basis of syndromes acquired

from the security bits. The internal connection diagram of the decoders corresponds to the respective applied convolutional code [2], [4]. Individual corrected sub-streams are finally multiplexed into an output bit stream of v_1 speed by means of a parallel to serial converter. The basic internal connection of the multiplexor and the demultiplexor is identical with the encoder.

A complete simulation of the codec of the given model together with a graphic output (see Fig 3.7) can be performed by connecting all of the sub-parts. The figure shows the following graphs of Berlekamp – Preparate code for the correction of 4 error bursts. First of the charts shows the unsecured input bit sequence with the rate v_1, delayed by 27 times of the first time base. The lag is introduced in order to make the chart visualization clearer (corresponding signal elements are one under the other). On the second chart, we can see the decoded output bit sequence with the same transmission rate. The third chart shows the difference between input data and decoded data. The zero course represents the match of signals. The fourth chart demonstrates data encoded by the encoder that enter into the simulated transmission channel with the transmission rate v_2. The second but last chart shows burst errors modifying the transmitted signal. On the very last chart there are transmitted data with inserted error bursts entering the decoder circuit. Since the input data sequence equals the output data sequence (see the third course), the correction of existing errors has been carried out and the simulation has confirmed the proper function of the codec.

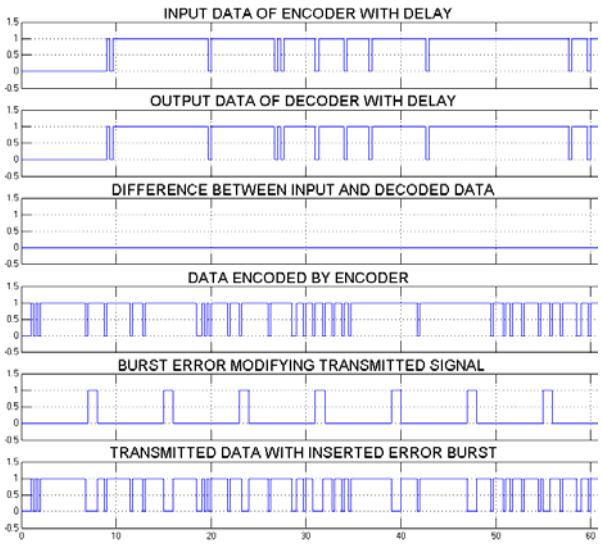

Fig. 3.7: Graphical output of simulation

4. Conclusion

We deal with the system error rate only when the occurrence of different types of errors exceeds the tolerable limit. Burst errors present a problem in the field of space miniaturization for data recording and acceleration of data transmission which entirely transforms the perspective of the protection of a useful information and enhancement in the protection of transmitted data from undesirable effects of noise. It shows that the now frequently used systems for the correction of contingent errors are no longer very suitable for this group of errors.

The presented article addresses the utilization of the Matlab mathematical programme for the elaboration of detailed simulation models. Higher quality presentation methods of security correction code offer not only better illustration but also improvement of the perspective on the issues in question and better conditions for a research concerning alternative solutions.

References

[1] BRUEN, A., FORCINITO, A.: *Cryptography, information theory, and error-correction: a handbook for the 21st century*. Hoboken, N.J.: Wiley-Interscience, 2005, 468 p., ISBN: 0-471-65317-9.

[2] KŘIVÁNEK, V.: *The Use of Matlab for the Simulation of the Burst Error Correction. International Journal of Computer Science and Network Security*, 2006, vol. 6, no. 7B, p. 141-145, ISSN: 1738-7906.

[3] MORELOS-ZARAGOZA, R.: *The Art of Error Correcting Coding*. 2nd ed., John Wiley & Sons Ltd., 2006, 384 p., ISBN 0-470-44782-4.

[4] RON, M.: *Introduction to coding theory*. Cambridge, Cambridge University Press 2006, 566 p., ISBN 0-521-84504-1.