WEP Warwick Evans Publishing

International Journal of Computer Science and Information Technology

ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 2, Number 2, Year 2024
DOI: https://doi.org/10.62051/ijcsit.v2n2.29
Journal homepage: https://wepub.org/index.php/IJCSIT/index

Model-based Design for RS Encoder and Decoder

Yang Nie

School of Physics and Electronic Information Engineering Jining Normal University, Ulanqab, China

nieyangwork@163.com

ABSTRACT

In this paper, an efficient design methodology of the Reed-Solomon (RS) encoder/decoder is proposed via Model-Based Design (MBD), which is focused on the development process from requirements development, design, implementation and testing. The design code can be automatically generated, and the testing-bench for RS encoders/decoders can also be automatically created. Compared to traditional design methods, the MBD approach can improve product quality and reduce development time.

KEYWORDS

Reed-solomon Code; Encoder; Decoder; MBD

1. INTRODUCTION

To evaluate the performance of new communication algorithms and architectures, engineers must create conceptual validation prototypes and design systems for field trials [1].

Typically, FPGA hardware with embedded processors is used to construct prototypes, as these platforms are known for their ability to support rapid prototyping and testing of new technologies and design modifications. Without external assistance, a team may struggle to implement communication prototypes and testing platforms based on FPGA. Although the engineers possess a profound knowledge of signal processing and communication algorithm development, they have less experience in hardware implementation. This experience gap is further compounded by the lack of tools and workflow. Although engineers typically use advanced languages such as MATLAB, hardware engineers use their own design tools and hardware description language (HDL).

It has proven difficult to adapt the traditional manual of the development process to the current requirements. To address this issue, it is necessary to introduce circuit design into the model-based development process [2]. Model-based design ensures that the final product meets the system requirements. Model-based design can enable engineering teams with different specializations to work together efficiently and to communicate between people working at different stages of the design process. It can detect and correct errors early in the development process, and can automatically generate robust, efficient and high-quality embedded software code and HDL code.

In this paper, the Reed-Solomon (RS) encoder / decoder is designed to show the important role of this method in the design of communication system through the MBD method. The structure of the paper is the following. The MBD methodology is introduced in Section 2. The theory of RS encoder / decoder is illustrated in Section 3. The main contribution of this work is presented in Section 4, where RS encoder / decoder are designed and tested by MBD. Finally, Section 5 summarizes the main conclusions of this work.

2. MODEL-BASED DESIGN

Model-Based Design improves design quality and accelerates design and verification tasks by employing an executable specification. This executable specification is elaborated to create hardware and software partitioning, automatically create hardware and software implementation code, and verify the hardware and software implementations in the context of the complete system. In the MBD [3,4], a system model is at the center of the development process, from requirements development, through design, implementation, and testing. The development flow by MBD is shown in Fig. 1. Through the establishment of floating point model, the fixed-point model and the system-level model for presenting a complete system design requirement, different professional engineers work effectively, and can communicate at different stages of the development flow. It enforces continuous testing and verification throughout the design process. Using MBD related tools, it can ensure that the design of the various stages of continuous testing, thereby ensuring the correctness of the design

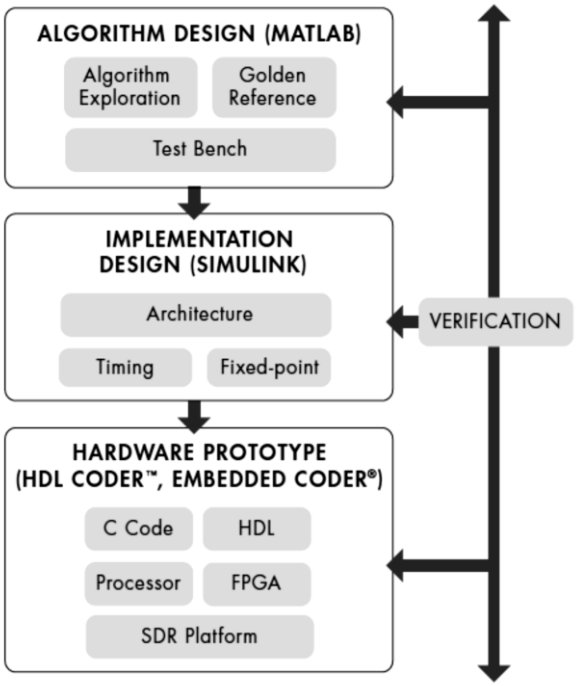

Fig.1 The Workflow of MBD

MBD is a process that enables fast and cost-effective development of dynamic systems, including control systems, signal processing, and communications systems. The model is an executable specification that you continually refine throughout the development process. After model development, simulation shows whether the model works correctly. All aspects of the design can be tested and verified according to the model of the system level and the demand. Fig. 2 is the joint development flow of MBD. System model is performed by MATLAB. HDL tool provides a workflow advisor that automates the programming of Xilinx® and Altera® FPGAs. It can control HDL architecture and implementation, highlight critical paths, and generate hardware resource utilization estimates. HDL tool generates VHDL and Verilog test benches for rapid verification of generated HDL code, automatically generates two types of simulation models. HDL simulation model is applied for performing HDL simulation with Simulink and an HDL simulator, such as Cadence Incisive or Mentor Graphics ModelSim. FPGA-in-the-loop (FIL) simulation model is applied for verifying design with Simulink and an FPGA board.

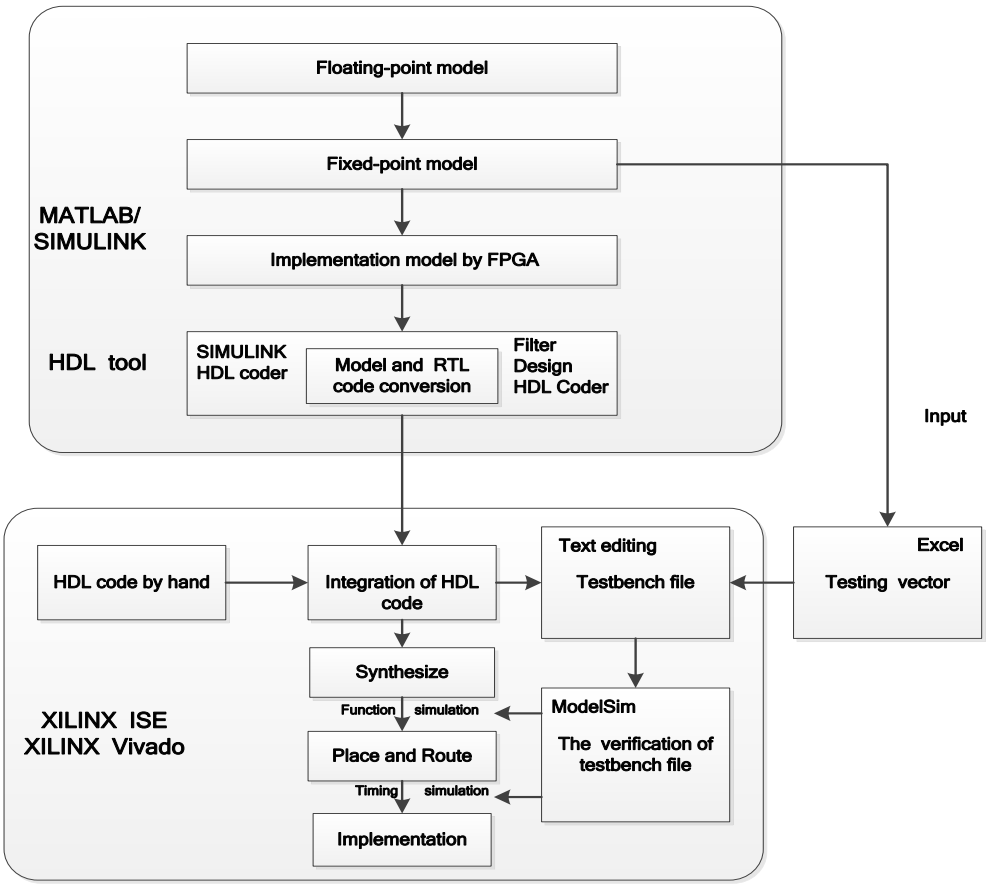

Fig. 2 The Workflow of MBD based on XILINX development tools

3. REED–SOLOMON CODES

Reed–Solomon codes named after Reed and Solomon [5] following their publication in 1960 have been used together with hard decision decoding in a wide range of applications. Reed–Solomon codes are maximum distance separable (MDS) codes and have the highest possible minimum Hamming distance. The codes have symbols from F_q with parameters $(q-1, k, q-k)$. They are not binary codes but frequently are used with $q = 2^m$, and so there is a mapping of residue classes of a primitive polynomial with binary coefficients [6] and each element of F_{2^m} is represented as a binary-tuple. Thus, binary codes with code parameters $\left(m(2^m - 1), km, 2^m - k\right)$ can be constructed from Reed–Solomon codes. Reed–Solomon codes can be extended in length by up to two symbols and in special cases extended in length by up to three symbols. In terms of applications, they are probably the most popular family of codes.

The IEEE 802.16 Broadband Wireless Access standard [6] employs a shortened version of the RS $(N = 255, K = 239, T = 8)$ code generated on GF(256), where N is the byte length of the coded code word, K is the byte length of the input information before the encoding, and T is the maximum number of bytes that can be corrected. The code primitive polynomial is

$$p(x) = x^8 + x^4 + x^3 + x^2 + 1 \tag{1}$$

The code generator polynomial is

$$g(x) = (x+\alpha)(x+\alpha^2)(x+\alpha^3)\cdots(x+\alpha^{2T}) \tag{2}$$

Where $\alpha^i \left(1 \le i \le 2T\right)$ are nonzero elements of GF$\left(2^m\right)$.

If the input information is $m(x)$, the code word can be obtained by using the coding formula of the cyclic code.

$$c(x) = m(x)x^{N-K} + \left[x^{2T} m(x)\right] \bmod g(x) \tag{3}$$

The code word gets the RS code after truncating the bit of information and deleting the check bit. In the IEEE 802.16d protocol, the supported RS codes include RS (32, 24, 4), RS (40, 36, 2), RS (64, 48, 8), RS (80, 72, 4), RS (108, 96, 6), RS (120, 108, 6). RS encoder introduces parity symbols, which are used by the RS decoder to detect and correct symbol errors. The code can correct up to symbol errors in each code word.

4. THE DESIGN RS ENCODER / DECODER BY MBD

This section shows how to implement encoder and decoder for the IEEE 802.16 standard using the MBD method, which includes the encoder and decoder design of RS code. Integer-Input RS encoder block and integer-input RS decoder block of Simulink library. In Fig. 3, it shows the model diagram of the entire design, which includes the source, the RS subsystem and destination. The RS subsystem is composed of the RS encoder module and the RS decoder module, and the structure is shown in Fig. 4. The ErrorGen subsystem adds noise to the RS encoded message.

The source repeatedly transmits the message followed by a guard interval. The model has parameters message length, for the number of symbols in the message to encode; and period, which includes the message length and the length of the guard interval. The guard interval between messages accommodates the latency of the encoder adding parity check symbols to the message, and the decoder performing a Chien search. In the initFcn callback of the model, the message length is set to 36 and period is set to 236 (which suggest that the guard interval has a length of 200 symbols).

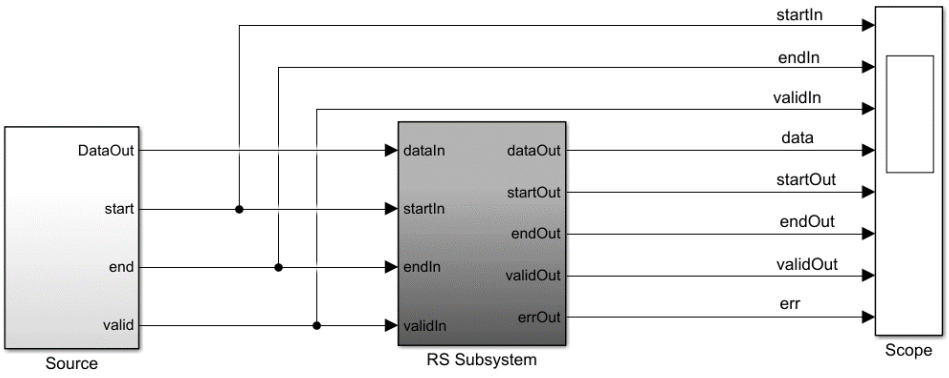

Fig. 3 The block diagram of the whole design model

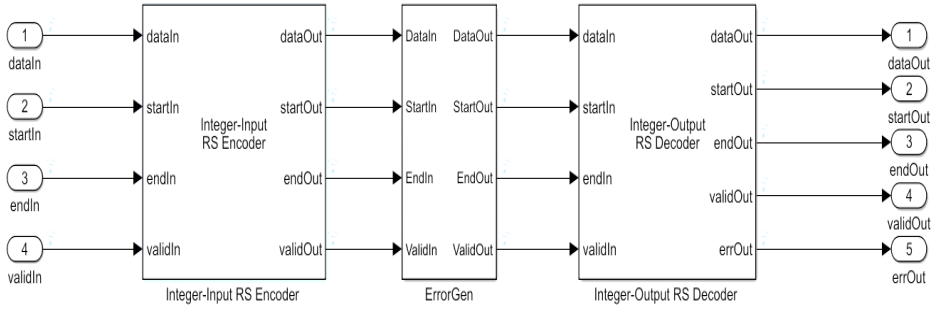

Fig. 4 The block diagram of the whole design model

4.1. The design of RS encoder by MBD

The RS encoder is designed by the Integer-Input RS Encoder block, which encode data using a Reed-Solomon encoder. The Integer-Input RS Encoder block creates a Reed-Solomon code with message length K and code word length N. We can specify N and K directly in the block dialog. The symbols for the code are integers between 0 and 2M-1, which represent elements of the finite field GF(2M)[7]. The default value of M is the smallest integer that is greater than or equal to log2(N+1), that is, ceil(log2(N+1)). We can change the value of M from the default by specifying the primitive polynomial for GF(2M). An (N, K) Reed-Solomon code can correct up to floor((N-K)/2) symbol errors (not bit errors) in each code word[8,9]. The configuration parameters of the Integer-Input RS Encoder block are shown in Fig. 5, where primitive polynomial parameter is determined by (1).

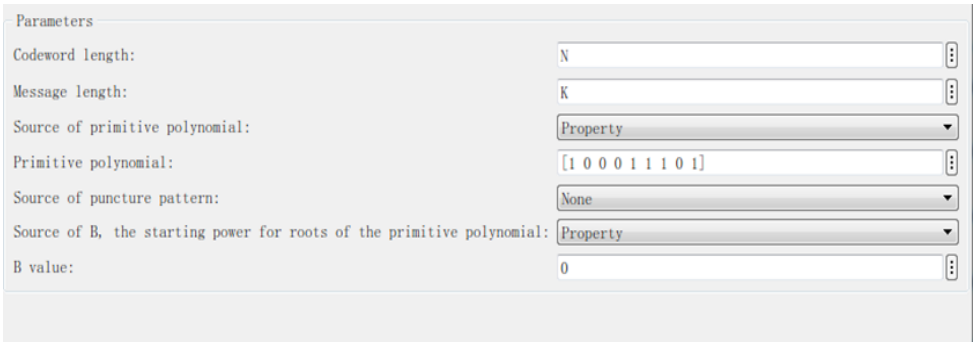

Fig. 5 The parameter configuration of the Integer-Input RS Encoder

The following Fig. 6 illustrates possible input and output signals to this block when code word length N is set to 7, message length K is set to 5, and the default primitive and generator polynomials are used. Suppose M = 3, N = 23-1 = 7, and K = 5. Then a message is a vector of length 5 whose entries are integers between 0 and 7. A corresponding code word is a vector of length 7 whose entries are integers between 0 and 7.

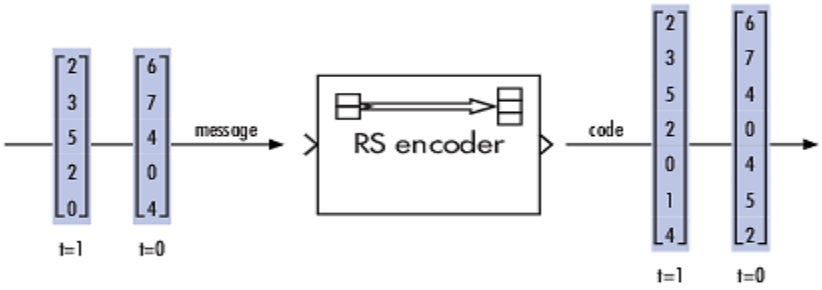

Fig. 6 The relationship between the input and output of the RS encoder

4.2. The design of RS decoder by MBD

The RS decoder is designed by the Integer-Input RS DEcoder block, which decode Reed-Solomon code to recover integer vector data. This block uses the Berlekamp-Massey decoding algorithm[10]. The Integer-Output RS Decoder block recovers a message vector from a Reed-Solomon codeword vector. For proper decoding, the parameter values in this block must match those in the corresponding Integer-Input RS Encoder block. If the decoder is processing multiple codewords per frame, then the same puncture pattern holds for all codewords.

The block can output shortened codewords when the Shortened message length S is specified. In this case, the codeword length N and message length K should specify the full-length (N, K) code that is shortened to an (N–K+S, S) code. The second output is the number of errors detected during decoding of the codeword. A -1 indicates that the block detected more errors than it could correct using the coding scheme. An (N, K) Reed-Solomon code can correct up to floor((N-K)/2) symbol errors (not bit errors) in each codeword. The configuration parameters of the Integer-Input RS DEcoder block are shown in Fig. 7.

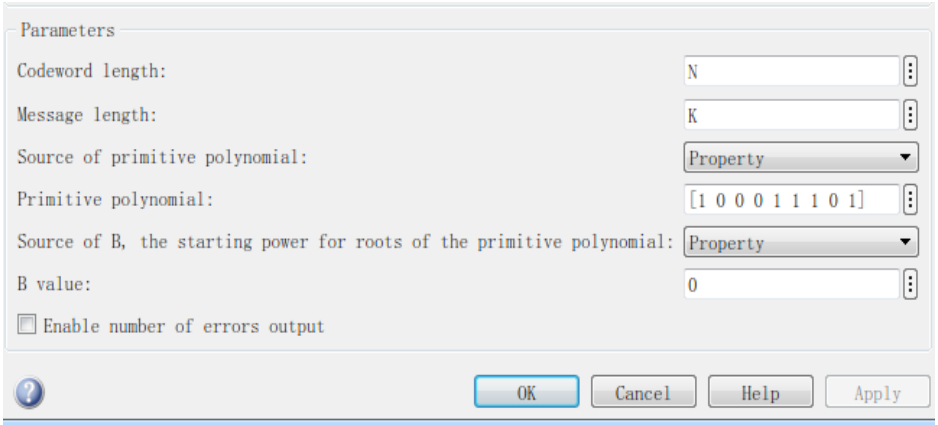

Fig. 7 The parameter configuration of the Integer-Input RS Decoder

4.3. The Simulation of RS encoder/decoder by MBD

The Logic Analyzer can be used to view multiple signals in one window and viewing signals this way makes it easier to observe transitions. The simulation results of Fig. 8 show that, in the Logic Analyzer output the input data signal represents the input of the RS encoder block and this is the 36 byte message given in the IEEE 802.16 specification. The encoded data shows the output of the RS encoder block. Note that the IEEE 802.16 specification performs puncturing of the parity bytes and retains

only the first four bytes of the 16 bytes. In this demo all 16 bytes of parity are used and the first four bytes of parity are 49, 31, 40, and BF, matching the IEEE 802.16 specification.

The errata signal represents the encoded data with noise added in the specified noise locations. These noise locations are marked with 1s in the inserter signal. The decoded and corrected message out of the RS decoder block is shown by the output data signal. Note that the RS decoder block introduces about 3 period lengths of latency. Observe output data to see that the errors induced by noise are corrected.

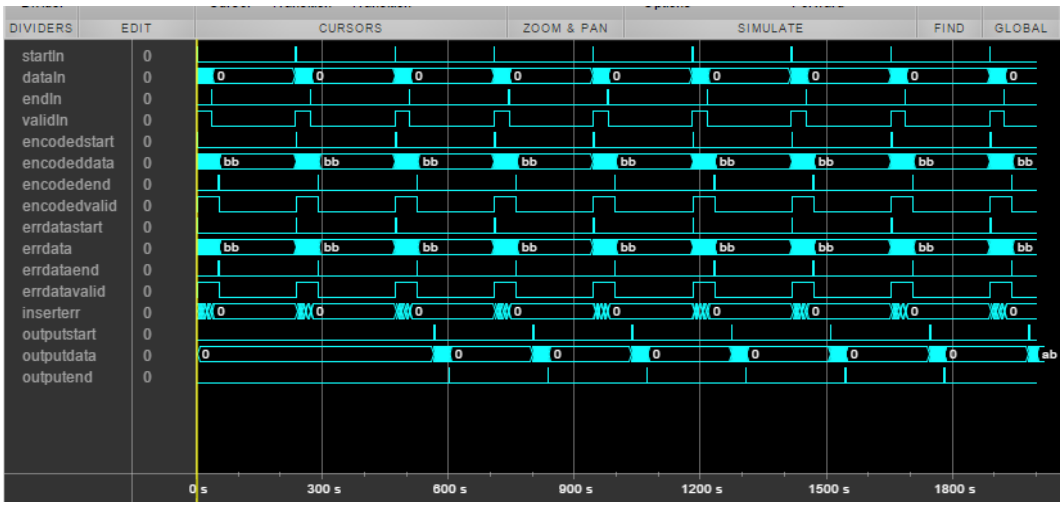

Fig. 8 The simulation results of the RS encoder/decoder

In the development flow based on MBD, HDL Workflow Advisor is a MATLAB tools for supporting the FPGA design, which verifies the model, automatically generates HDL code. Using XILINX FPGA development ISE as synthesis tool, the HDL code automatically generated of RS encoder/decoder is synthesized and implement by XILINX FPGA chip. The following simulation result shows the ModelSim HDL simulator after running the generated, which is shown in Fig. 9. It can be seen from the simulation results that it is not only accurate, but also fast and effective .

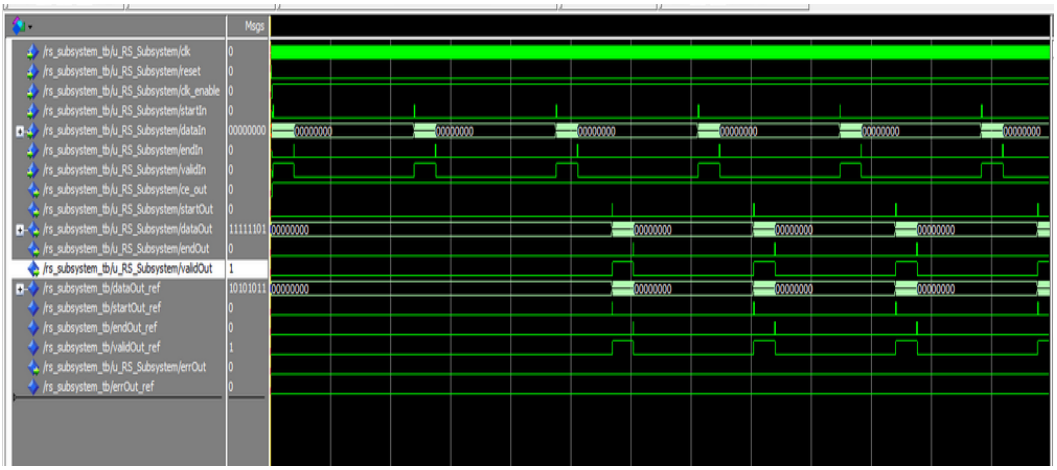

Fig. 9 The ModelSim simulation results of the RS encoder/decoder

5. CONCLUSIONS

The design and implementation of the communication algorithm is a very complex process, and the hardware implementation is very difficult. In order to meet the fast hardware implementation of communication algorithms, this paper proposed an optimized approach of the RS encoder/decoder using MBD workflow. By the method of MBD, the complex communication algorithm can automatically generate HDL code, and quickly complete the functional validation of FPGA design. The described methodology allows accelerating the design process of communication systems. Compared with the traditional design methods, the MBD method can improve product quality and reduce development time.

REFERENCES

[1] Nie Y, Ge H, Jing L. Model-Based Design Methodology for Digital Up and Down Conversion of Software Defined Radio[J]. International Journal of Multimedia and Ubiquitous Engineering, 2016, 11(4): 27-36.

[2] Chatterjee S, Kleijn W B. Auditory Model-Based Design and Optimization of Feature Vectors for Automatic Speech Recognition[J]. IEEE Transactions on Audio Speech & Language Processing, 2011, 19(6):1813-1825.

[3] Wang S, Shin K G. Task construction for model-based design of embedded control software[J]. IEEE Transactions on Software Engineering, 2006, 32(4):254-264.

[4] Costabile M F, Fogli D, Mussio P, et al. Visual Interactive Systems for End-User Development: A Model-Based Design Methodology[J]. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 2007, 37(6):1029-1046.

[5] Wicker S B, Bhargava V K. Reed-Solomon Codes and Their Applications[M]. John Wiley & Sons, Inc. 1999.

[6] IEEE 802.16: IEEE Standard for Air Interface for Broadband Wireless Access Systems(Revision of IEEE Std 802.16-2009). IEEE-SA. 8 June 2012.

[7] Berlekamp E. Bit-serial Reed - Solomon encoders[J]. IEEE Transactions on Information Theory, 2003, 28(6):869-874.

[8] Hsu I S, Reed I S, Truong T K, et al. The VLSI Implementation of a Reed—Solomon Encoder Using Berlekamp's Bit-Serial Multiplier Algorithm[J]. IEEE Transactions on Computers, 2006, C-33(10):906-911.

[9] Dash A R, Lenka T R. VLSI implementation of Reed-Solomon encoder algorithm for communication systems[J]. Radioelectronics & Communications Systems, 2013, 56(9):441-447.

[10] Clark, George C. Jr., and J. Bibb Cain, Error-Correction Coding for Digital Communications, New York, Plenum Press, 1981.