УДК 621.3

DOI: 10.18664/ikszt.v0i2.51936

MIROSHNIK M.A., Dr. of tech. science (Ukrainian State University of Railway Transport)

Implementation of cryptographic algorithms on FPGA-based digital distributed systems

This paper presents a single chip efficient FPGA implementation of RSA and DES for Digital Envelope Scheme that targets the Altera Apex 20KE EP20k200EBC356. Implementation of cryptographic algorithms on programmable devices like FPGAs run much faster than on software while preserving physical security of hardware solutions. The high throughput in the implementation of cryptographic algorithms for digital envelope scheme is achieved by means of exploiting the parallelism present in the DES and RSA operations as well as the features of Altera Apex 20KE device family which best suit for system-on-a-programmable-chip (SOPC) applications. The parallelized single chip implementation of DES and RSA for performing the Encryption/Decryption for the Digital envelope scheme offers a throughput of 3.5 Gbits/sec at a system clock rate of 54.7 MHz. This implementation even provides a means for using the DES and RSA separately.

Key words: *Encryption, Decryption, Plaintext, Public Key, Private Key, Digital Envelope.*

1 INTRODUCTION

Recently data security is becoming very important as more information is exchanged through the internet. One of the advantages of Internet is the open system architecture. On the other hand, the lack of privacy in Internet based information exchange is preventing its application any further for secure communication. One way of eliminating this weakness is through the introduction of Cryptography. Cryptography is the art or science of keeping messages secret. There are two kinds of cryptosystems symmetric and asymmetric. Symmetric cryptosystems use the same key (the secret key) to encrypt and decrypt a message, whereas the asymmetric cryptosystems use one key (the public key) to encrypt a message and a different key (the private key) to decrypt it. Asymmetric cryptosystems are also called public key cryptosystems. When using secret key cryptosystems, users must first agree on a session key, that is, a secret key to be used for the duration of one message or communication session. In completing this task there is a risk the key will be intercepted during transmission. This is part of the key management problem. Public key cryptography offers an attractive solution to this problem within a framework called a digital envelope.

The digital envelope consists of a message encrypted using secret key cryptography and a secret key (encrypted using public key cryptography). Not only do digital envelopes help solve the key management problem; they also increase the performance (relative to using a public key system for direct encryption of message data) without sacrificing security. The increase in performance is obtained by using a secret-key cryptosystem to encrypt the large and variably sized amount of message data, reserving public key cryptography for encryption of short length keys. Since secret key cryptosystems are much faster than public key cryptosystems.

The fig. 1 shows the digital envelope scheme. In this scheme the data to be transmitted is encrypted using secret key algorithm. This ensures fast processing. Establishment of the session key is done by transmitting the public key encryption of the secret key (session key).The symmetric key and asymmetric key encryption algorithm that is taken for implementation are DES and RSA respectively.

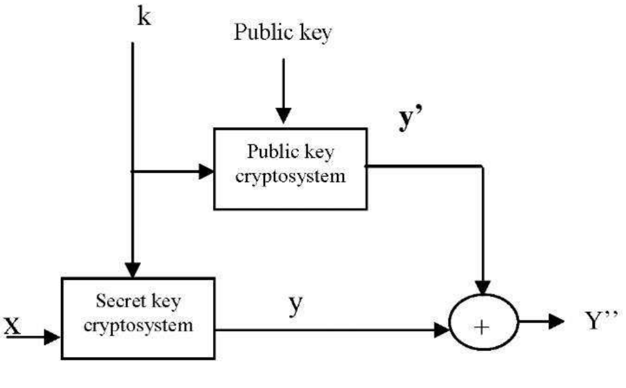


Fig. 1. Digital Envelope Scheme-Encryption

Where

$$y = e_{\sec ret}(x,k),$$

$$y' = e_{public}(k,_{public\_key}),$$

$$y'' = concatenation(y,y').$$

Hardware implementation of these algorithms is preferred now days because of their high performance, physical security and low power consumption. Though ASIC implementation of encryption algorithms are performing better, they are not flexible. FPGA provides the needed flexibility as well as the high speed of custom hardware implementations. In this paper the major focus is on efficient FPGA implementation of both DES and RSA on a single chip for high performance support for digital envelope scheme [1].

The following sessions of the paper is organized as follows: session 2 discusses the Choice of algorithms for digital envelope and recalls Data Encryption Standard (DES) and RSA algorithms. Session 3 is oriented towards the scope for parallelism in both DES and RSA. Session 4 is oriented towards the implementation issues. They focus on the practical issues that should be considered while implementing in FPGAs. Session 5 describes the results and discussion of the implementation through resource utilization and speedup in terms of throughput. Finally, the concluding remarks are given.

2 CHOICE OF ALGORITHMS FOR DIGITAL ENVELOPE

Because of the wide popularity of the DES in the symmetric and RSA in the asymmetric crypto algorithm category, these two algorithms are widely used for the implementation of digital envelope scheme.

2.1 Data Encryption Standard (DES).

DES is a private key (symmetric) algorithm [2 - 3]. It comes under the category of block cipher operating on 64-bit blocks of plaintext utilizing a 64-bit key.

Every eighth bit of the 64-bit key is used for parity checking or otherwise ignored. In DES key controls exactly how this process works. By doing these operations repeatedly and in a non-linear manner you end up with a result which can not be used to retrieve the original without the key. The fig. 2 shows the block diagram of DES, in which each 64 bits of data is iterated on 16 times. For each iteration a 48 bit subset of the 56 bit key is fed into the encryption block represented by the dashed rectangle below.

Decryption is the inverse of the encryption process. The Function "F" shown in the diagram is the heart of DES. It actually consists of several different transforms and non-linear substitutions [4 - 5].

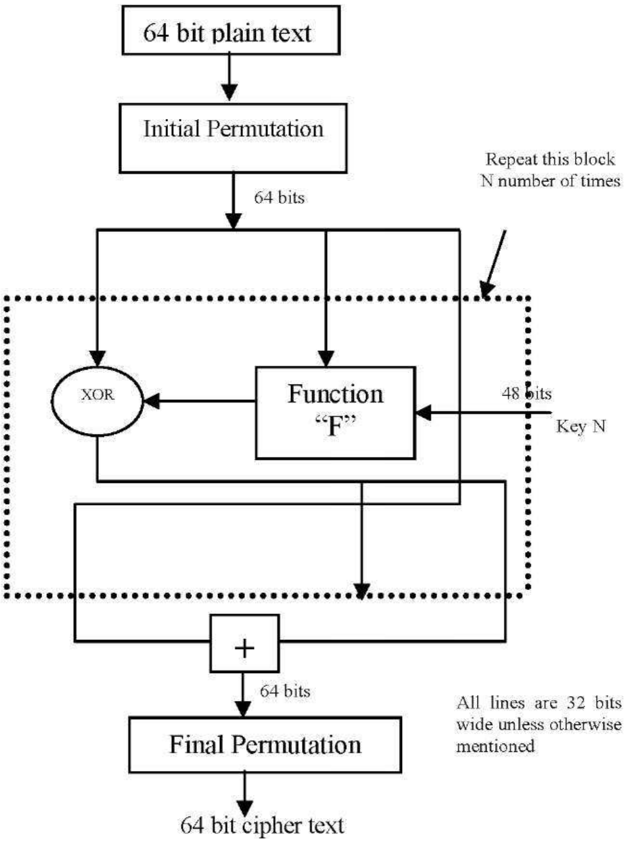


Fig. 2. Block Diagram of DES

DES is a private key (symmetric) algorithm [3]. It comes under the category of block cipher operating on 64-bit blocks of plaintext utilizing a 64-bit key.

Every eighth bit of the 64-bit key is used for parity checking or otherwise ignored. In DES key controls exactly how this process works. By doing these operations repeatedly and in a non-linear manner you end up with a result which can not be used to retrieve the original without the key. The fig. 2 shows the block diagram of DES, in which each 64 bits of data is iterated on 16 times. For each iteration a 48 bit subset of the 56 bit key is fed into the encryption block represented by the dashed rectangle below.

Decryption is the inverse of the encryption process. The Function "F" shown in the diagram is the heart of DES. It actually consists of several different transforms and non-linear substitutions [5].

2.2 RSA Algorithm

The following is the relatively easy to understand math behind RSA public key algorithm [3].

1. Find *P* and Q, two large (e.g., 1024-bit) prime numbers.

2. Choose E such that E is greater than 1, E is less than N = PQ, and E and (P-1) (Q-1) are relatively prime, which means they have no prime factors in common. E does not

have to be prime, but it must be odd. (P-1)(Q-1) can't be prime because it's an even number.

3. Compute D such that (DE - 1) is evenly divisible by (P-1) (Q-1). It can be written as DE = 1 (mod (P-1) (Q-1)), and D is called the multiplicative inverse of E.

4. The encryption function is C = ME mod N, where C is the cipher text (a positive integer), M is the plaintext (a positive integer).The message being encrypted, M, must be less than the modulus, N.

5. The decryption function is M = CD mod N, where C is the cipher text (a positive integer), M is the plaintext (a positive integer).

Your public key is the pair (N, E). Your private key is the number D. The product PQ = N is the modulus. E is the public exponent. D is the secret exponent. You can publish your public key freely, because there are no known easy methods of calculating D, P, or Q given only (N, E) (your public key).

3 SCOPE FOR PARALLELISM IN DES AND RSA

The main purpose of this design is to implement both DES and RSA on a single FPGA for a high performance support to the digital envelope scheme which can be adopted in practical encryptor. The following sections explain the implementation and optimization steps taken to improve performance.

The iterative nature of DES makes it more suitable for partial pipelining. Pipelining usually replicates the hardware needed for a single round and introduces data storage between each round. Though this technique increases the number of data blocks encrypted concurrently, this design does not go exactly with partial pipelining because of the speed limitation of RSA algorithm that is supported in the same chip. So a fine blend of iterative loops and sub pipelining inside each round to achieve less hardware requirement and high speed that will match DES with RSA on a single chip. Since the DES design has number of constant S- boxes the system level memory are configured as ROM to store these S- box values. This type of ROM implementation of S- Boxes is the most efficient way [4] of implementing DES on FPGA. In the implementation of DES, key scheduling involves carrying out a shift operation during each round of the 16-round DES to generate the sub key. Figure 3 show the key processor that is used in the implementation.

In each round the 56-bit key is divided into two 28 bit halves and each halve is independently rotated left either one or two positions, depending on the round. The 56-bit result is used as the input for the next round and to select the 48 bits that make up the key for the current round. These operations are carried out in parallel to the encryption/decryption operation. With the DES the same function F is used forward to encrypt or backwards to decrypt. The only change is that keys must be taken in reverse order *(k16, k15 ..., k1).*

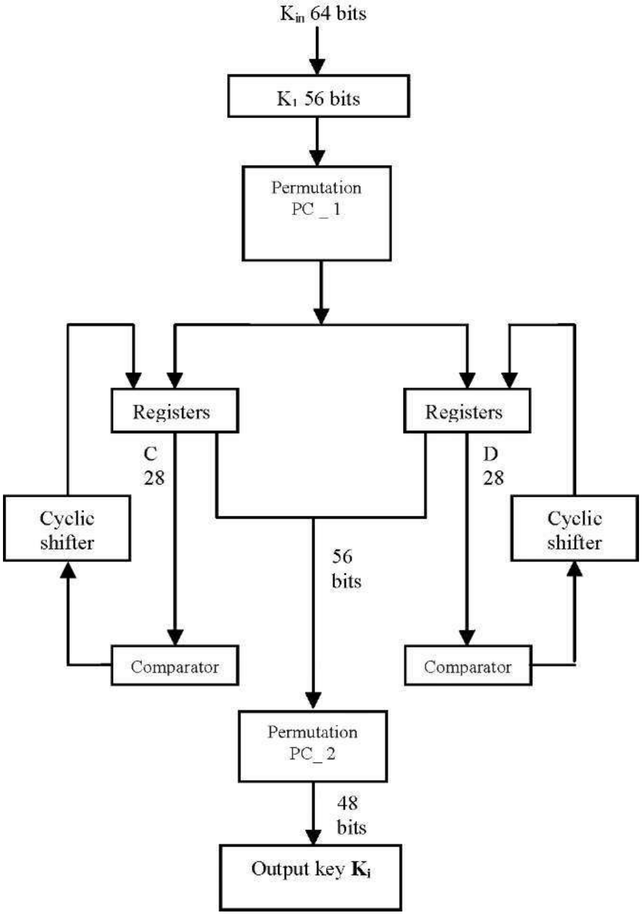


Fig. 3. Key Processor

To this extent, several algorithms have been proposed [6], which try to reduce the time needed for these exponentiations. One of the most widely used is the square and multiply algorithm. This method performs modular multiplication and squaring, as follows:

```
$Y : -1; \quad Z : -M;$
$For\ i : 0\ \ to\ \ k-1\ \ do\ \ \{k\ \ is\ \ the\ \ number\ \ of\ \ bits\}$
$Ifr\ \ ei-1\ \ then$
$Y : Y \times Z \bmod N; \quad \{Modularmultiplication\}$
$End\{If\}$
$Z : -Z^{2} \bmod N; \quad \{Modularmultiplication\}$
$End;\{For\}$
$\mathrm{Re}turn\,?;$
```

Provided that the modular squaring and multiplication are performed in parallel, only log2k steps are required to calculate the exponentiation. At the same time this implementation is based on Montgomery method for computing both modular multiplications and modular squaring as Montgomery scheme is very efficient [7 - 8].

4 IMPLEMENTATION ISSUES

The obvious way of improving the performance of cryptographic algorithms in the digital envelope scheme is to implement them in hardware. This implementation has improved the performance of DES in two ways. The key scheduling part of the DES scheme is improved by computing the sub keys needed for each stage in parallel. Then the iterative looping and sub pipelining inside each round of the DES has given good performance characteristics for implementing both DES and RSA on the same FPGA.

The primary factor that determines the performance of the public key crypto systems (RSA) is the implementation efficiency of the modular multiplication and exponentiation. There are various algorithms available for implementing modular multiplications like Barrett's and Booth's methods and Brickell's algorithm. But this design considers Montgomery's algorithm as it is considered the most popular and more efficient. The implementation of Montgomery's algorithm for modular multiplications in RSA algorithm of the Digital envelope has shown a better time response [8].

5 RESULTS AND DISCUSSION

This design has implemented both DES and RSA cryptographic algorithms on a single FPGA for a high speed encryption/decryption support for Digital envelope scheme of data security. The throughput for the implementation is calculates as follows

$$Throughput(Mbit/\sec) - 64bits * clock$$
$$frequency(MHz)$$

This design and implementation of DES for the digital envelope achieve a very good encryption/decryption rate of 6.4 Gbit/s, which is faster [8] than many equivalent implementations. It also compares promisingly with existing ASIC implementations. Also the RSA implementation with the parallel Montgomery scheme achieves a reasonable encryption/decryption rate of 3.5 Gbit/s this result is also better than many such implementations in the literature [8].

When both DES and RSA algorithms are implemented in a single chip FPGA then the encryption/decryption rate of DES is limited by the RSA scheme, but the resultant through put that is achieved is better than their software versions that is used in digital envelope scheme at present. So this hardware accelerator will be a great boon for digital envelope scheme that is practiced today.

The simulation results for this implementation are achieved in ALTERA Quartus II 4.0 environment [8]. The synthesis results are achieved using Mentor Graphics Leonardo Spectrum. The Table 1 and Table 2 shows the throughput results achieved in the sequential and parallel implementation of DES and RSA. The parallel implementation is approximately twice faster than its sequential counterpart in both the cases.

Table 1

Throughput comparison for DES

DES	Clock Frequency (MHz)	Throughput (Mbits/sec)
Sequential	49.6	3174.4
Parallel	100.5	6432

Table 2

Throughput comparison for RSA

DES	Clock Frequency (MHz)	Throughput (Mbits/sec)
Sequential	25.5	1632
Parallel	54.7	3501

The following session Table 3 and Table 4 compares the resource utilized in implementing the RSA algorithm in a sequential and parallel way. The results show that more resources are occupied by the sequential implementation.

RSA - Sequential Implementation. Resources Utilization: Device: Altera Apex20KE Family: EP20K400EBC652 Clock Frequency: 25.5 MHz

Table 3

RSA - Sequential Implementation

Resources	Available	Used	Percentage of Usage
IOS	488	72	14.75
LCS	16640	11824	71.06

RSA - Parallel Implementation. Clock Frequency**:** 54.7 MHz

Table 4

RSA - Parallel Implementation

Resources	Available	Used	Percentage of Usage
IOS	488	21	4.3
LCS	16640	2198	13.21

Based on all the implementation results achived the parallel implementation of DES and RSA is combined for a suitable implementation of the system for Digital envelope scheme. This implementation has shown an improvement in both throughput and resource usage.

Digital Envelope Scheme. Device: Altera Apex20KE Device Family: EP20K200EBC356 Clock Frequency: 54.7 MHz

Table 5

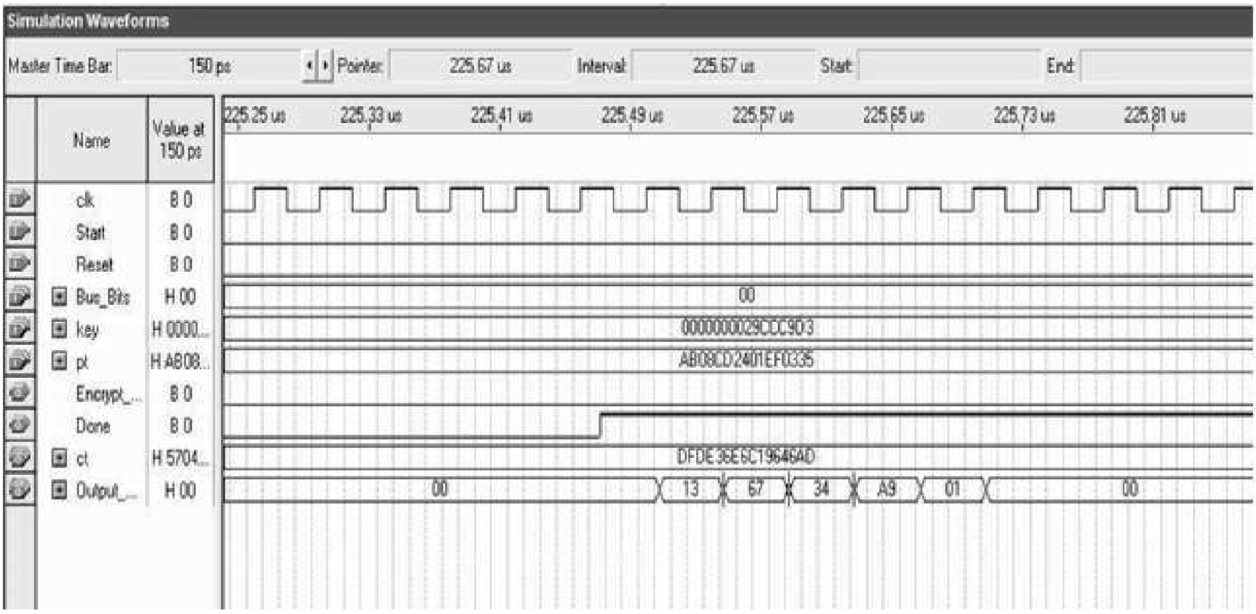
Digital Envelope Scheme

Resources	Available	Used	Percentage of Usage
IOS	273	213	78.02
LCS	8320	3161	37.99
Memory bits	106496	32768	30.77
Throughput = 3.5 Gbits/sec			

The fig. 4 shows the simulation result achieved for this implementation in ALTERA Quartus II 4.0 environment. It shows the encrypted output both for DES and RSA from the same chip.

CONCLUSION

This reconfigurable single chip design and implementation of DES and RSA has significantly increased the throughput of the digital envelope scheme. In order to achieve best Performance, this method has exploited the parallelism in the encryption pipe and key scheduling pipe of DES and modular squaring and multiplication process in RSA. The proposed implementation achieves an encryption rate of 3.5 Gbits/sec at 54.7 MHz. Upon comparison, this implementation offers better results than previously reported in literature [1 - 8].


Master Time Bar: 150 ps
Pointer: 225.67 us
Interval: 225.67 us
Start:
End:

Name	Value at 150 ps
clk	B 0
Start	B 0
Reset	B 0
Bus_Bits	H 00
key	H 0000...
pt	H AB08...
Encrypt_...	B 0
Done	B 0
ct	H 5704...
Output_...	H 00

Fig. 4. Simulation result of Digital envelope References

REFERENCES

1. Мирошник М.А. Разработка средств защиты информации от в распределенных компьютерных системах и сетях. Інформаційно-керуючі системи на залізничному транспорті. – 2015. – №1. – с. 18-25.
2. Мирошник М.А. Методы защиты цифровой информации в распределенных компьютерных сетях. Інформаційно-керуючі системи на залізничному транспорті. – 2014. – №5. – с. 66-70.
3. Miroshnik M. Investigation of the influence of generator signal higher order propagation components and ways of its compensating in the multiprobe microwave multimeter/ / Miroshnik M.A., Zaichenko O. B., Kovalenko M.A. // Інформаційно-керуючі системи на залізничному транспорті. – 2014. – №3, 78-83.
4. Miroshnik M. Model of influences of sensor reflections on the accuracy of microwave reflectometer. / Miroshnik Marina, Klyuchnyk Igor, Tsekhmistro Roman, Warsza Zygmunt, Zaichenko Olga // PAK (Pomiary Automatyka Kontrola) vol. 60, nr /2014, p.223-225.
5. Miroshnik M. Uses of programmable logic integrated circuits for implementations of data encryption standard and its experimental linear cryptanalysis. / Miroshnik M., Kovalenko M. // Інформаційно-керуючі системи на залізничному транспорті. – 2013. – №6, с.36-45.
6. Miroshnik M. Application of software complex for query processing in the database management system with a view of dispatching problem solving in Grid systems. / Miroshnik M.A., Kotukh V.G., Selevko S.N. // Telecommunications and radio engineering. – 2013. Vol.27, № 10, P. 875-891.

7. Мирошник М.А. Диагностические эксперименты в системах защиты информации на сетях клеточных автоматов. / М.А. Мирошник, Я.Ю.Королева // Інформаційно-керуючі системи на залізничному транспорті. – 2009. – №4. – С. 142–145.
8. Мирошник М.А. Методы эффективного кодирования внутренних состояний микропрограммных автоматов. / М.А. Мирошник, Я.Ю.Королева, // Технология приборостроения. – 2011. – №1. – С. 12–16.

Мирошник М.А. Реализация криптографических алгоритмов на основе ПЛИС для цифровых распределенных систем. В статье представлена однокристальная эффективная реализация в виде ПЛИС RSA и DES для цифровой конвертации, которая предназначена для *Altera Apex 20KE EP20k200EBC356.* Реализация криптографических алгоритмов на основе программируемых устройств, таких как ПЛИС, работает намного быстрее, чем на программном обеспечении, сохраняя физическую безопасность аппаратных решений. Высокая пропускная способность в реализации криптографических алгоритмов для схем цифровой конвертации достигается путем использования параллелизма, присущего операциям DES и RSA, а также особенности устройств семейства *Altera Apex 20KE*, которые лучше всего подходят для приложений систем на программируемой логической матрице (SOPC). Распараллеленная однокристальная реализация DES и RSA для выполнения шифрования / дешифрования для схемы цифровой конвертации предлагает пропускную способность 3,5 Гбит/с при тактовой частоте системы 54.7MHz. Эта реализация даже предоставляет средство для использования DES и RSA отдельно.

Ключевые слова: шифрование, дешифрование, незашифрованный текст, открытый ключ, секретный ключ, цифровой конверт.

Мірошник М.А. Реалізація криптографічних алгоритмів на основі ПЛІС для цифрових розподілених систем. У статті представлена однокристальна ефективна реалізація у вигляді ПЛІС RSA і DES для цифрової конвертації, яка призначена для *Altera Apex 20KE EP20k200EBC356*. Реалізація криптографічних алгоритмів на основі програмованих пристроїв, таких як ПЛІС, працює набагато швидше, ніж на програмному забезпеченні, зберігаючи фізичну безпеку апаратних рішень. Висока пропускна здатність у реалізації криптографічних алгоритмів для схем цифрової конвертації досягається шляхом використання паралелізму, притаманного операціям DES і RSA, а також особливості пристроїв сімейства *Altera Apex 20KE*, які найкраще підходять для додатків систем на програмованої логічної матриці (SOPC). Розпаралелена однокристальна реалізація DES і RSA для виконання шифрування / дешифрування для схеми цифрової конвертації пропонує пропускну здатність 3,5 Гбіт/с при тактовій частоті системи 54.7MHz. Ця реалізація навіть надає засіб для використання DES і RSA окремо.

Ключові слова: шифрування, дешифрування, незашифрований текст, відкритий ключ, секретний ключ, цифровий конверт.

Рецензент д.т.н., профессор, профессор кафедры СКС Листровой С.В. (УкрГУЖТ)

Поступила 30.03.2015 г.