Real-time DSP Implementations of Voice Encryption Algorithms

Cristina-Loredana Duta[1], Laura Gheorghe[2] and Nicolae Tapus[1]
[1]*Department of Computer Science and Engineering, University Politehnica of Bucharest, Bucharest, Romania*
[2]*Research and Development Department, Academy of Romanian Scientists, Bucharest, Romania*
cristina.duta.mapn@outlook.com, {laura.gheorghe, nicolae.tapus}@cs.pub.ro

Keywords: Digital Signal Processor, Voice Encryption, Blackfin Processor, TMS320C6X Processor, AES, RSA, NTRU.

Abstract: In the last decades, digital communications and network technologies have been growing rapidly, which makes secure speech communication an important issue. Regardless of the communication purposes, military, business or personal, people want a high level of security during their conversations. In this context, many voice encryption methods have been developed, which are based on cryptographic algorithms. One of the major issues regarding these algorithms is to identify those that can ensure high throughput when dealing with reduced bandwidth of the communication channel. A solution is to use resource constrained embedded systems because they are designed such that they consume little system resources, providing at the same time very good performances. To fulfil all the strict requirements, hardware and software optimizations should be performed by taking into consideration the complexity of the chosen algorithm, the mapping between the selected architecture and the cryptographic algorithm, the selected arithmetic unit (floating point or fixed point) and so on. The purpose of this paper is to compare and evaluate based on several criteria the Digital Signal Processor (DSP) implementations of three voice encryption algorithms in real time. The algorithms can be divided into two categories: asymmetric ciphers (NTRU and RSA) and symmetric ciphers (AES). The parameters taken into consideration for comparison between these ciphers are: encryption, decryption and delay time, complexity, packet lost and security level. All the previously mentioned algorithms were implemented on Blackfin and TMS320C6x processors. Making hardware and software level optimizations, we were able to reduce encryption/decryption/delay time, as well as to reduce the energy consumed. The purpose of this paper is to determine which is the best system hardware (DSP platform) and which encryption algorithm is feasible, safe and best suited for real-time voice encryption.

1 INTRODUCTION

Security and privacy represent a fundamental issue when transmitting information through insecure communication channels.

There are many channels available for transmitting speech signals, for instance telephone networks and private or public radio communication systems. Speech signal can be represented in two forms: *analogue* and *digital* form. In *analogue* representation, it is a waveform which describes the frequency and amplitude of the signal. In *digital* form, it is the numeric representation of the analogue form, where the signal is composed of zeros and ones.

There are some situations when the information transmitted has to be confidential, such as diplomatic and military communications during war and peace. Since speech has a lot of redundancy compared with written text, it becomes a very difficult task to provide security for it.

There are two distinct approaches to achieve speech security: *analogue* scrambling and *digital* ciphering. In the past, the researchers have been interested in speech scrambling because it uses small bandwidth, has simple implementations and good capabilities when dealing with asynchronous transmission.

The purpose of voice coders in digital telecommunication systems is to reduce the required transmission bandwidth. Several vocoders have been invented – LPC-10 (Linear Prediction Coding), CELP (Code Excited Linear Prediction), MELP (Mixed Excitation Linear Prediction) and so on. In general, secure communication systems are based on LPC techniques. The main reason is that LPC voice coding can ensure low bit rates and high voice intelligibility.

Applications that use cryptographic algorithms often demand a set of strict requirements for implementations, such as low resource consumption,

Duta, C-L., Gheorghe, L. and Tapus, N.
Real-time DSP Implementations of Voice Encryption Algorithms.
DOI: 10.5220/0006208304390446
In *Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP 2017)*, pages 439-446
ISBN: 978-989-758-209-7

reduced number of logic gates and memory, and efficient power consumption. In this context, designing implementations that fulfil all these requirements it's a very challenging task and represents a wide area of research.

More specifically, an implementation should be fast enough to make sure that the execution of the cryptographic algorithms doesn't slow down the system significantly. This can be achieved by hardware acceleration (because it has been demonstrated that software implementations cannot achieve the desired level of performance with reasonable costs (Pedre, 2016) and (Joao, 2009)). Regarding the available resources, only a small part of them are dedicated to cryptography, which makes implementing high-security algorithms very complicated.

Another important point for secure voice communications is *real-time processing*. In this case, the aspect of framing of the incoming data becomes an essential task. A good balance of the block size and all the parameters has to be found (short buffers can cause buffer overflow and large buffers can lead to delays depending on the sampling rate).

Because the cryptographic algorithms are complex, they need to be implemented on flexible platforms in order to meet real-time requirements for voice encryption. In this context, we have chosen for our implementations two DSP hardware platforms from Texas Instruments: *Blackfin ADSP-BF537* and *TMS320C6711*.

In this paper, we study implementations of cryptographic algorithms on existing embedded architectures. We took into consideration three algorithms, evaluate their performance (in terms of encryption/decryption/delay time), their complexity, packet loss, security level as well as their power consumption. We implemented symmetric and asymmetric algorithms on DSP platforms and explored how to make use of the existing architectural features to provide the best mapping between cryptographic processing and the target embedded systems and how to reduce the energy consumption. We performed step by step optimizations in order to meet real time requirements with the purpose to determine which encryption algorithm and which hardware platform is best suited for real time secure communications.

The remaining of the paper is organized as follows. Section 2 presents a brief overview of the cryptographic algorithms and of the DSP platforms considered in this paper. In Section 3 the related work is described. Section 4 includes our optimization approach and experimental setup as well as the implementation of the system in detail. In Section 5 we present the results of the real-time implementations of the algorithms and a comparison between them based on several criteria. The conclusions and future work are summarized in Section 6.

2 BACKGROUND

This section includes a brief description of the implemented cryptographic algorithms: symmetric cipher such as AES and asymmetric ciphers such as RSA and NTRU, and the description of general aspects of DSP architectures used: Blackfin ADSP-BF537 and TMS320C6711.

2.1 Speech Scrambling Techniques

There are several types of analogue voice scrambling, which are described further on.

Time domain scrambling – the voice is being recorded for some time and then is cut into small frames, which are transmitted in a different time order, based on a secret code. The main disadvantage of this technique is the fact that the signal has the same frequencies as before, which makes it easy to recover basic information.

Frequency domain scrambling – the frequencies of the voice are being inverted. The main problem is that the fundamental characteristics of the voice signal are not significantly changed which makes this technique vulnerable.

Amplitude domain scrambling – the amplitude of the signal is modified, but this doesn't really change the signal.

Compared with analogue scrambling, digital encryption is a much stronger method of protecting speech communications. The main advantage is that it doesn't matter what kind of signal is being encrypted (text, video, voice and so on). Moreover, there are little possibilities for cryptanalysis compared to analogue scrambling. On the other side, if the data is being corrupted, it will not be decrypted correctly (degradation of voice quality) and if the data are lost, then the synchronization and communication will be lost.

Cryptographic algorithms can be classified based on the number of keys that are used in the encryption/decryption process. *Secret Key Cryptography (SKC)* uses a single key for both encryption and decryption (AES, for example). *Public Key Cryptography (PKC)* uses one key for encryption and another for decryption and includes

algorithms such as: NTRU, RSA and Elliptic Curve Cryptography (ECC).

2.2 AES Block Cipher

AES (Daemen, 2001) is a block cipher which is included in the symmetric-key algorithms category and was designed as a replacement for Data Encryption Standard (DES). The block size is 64 bits, the key has variable length of 128, 192 and 256 bits and a variable number of rounds based on the key size (10, 12 or 14 rounds).

A round includes the following operations: substitution of bytes, shifting of rows, mixing of columns and XOR with the round key.

Regarding its security, until 2009 the only successful attacks against full AES were side-channel attacks. After that, several other attacks were developed such as related-key (Biryukov, 2009) and biclique (Bogdanov, 2011).

2.3 RSA Cipher

RSA (Rivest, 1997) represent the first developed public-key cryptosystem that was used to securely transmit information. The key size is variable, starting from 512 bits until 2048 bits.

The security of RSA cryptosystem relies on the problem of factoring large numbers and the RSA problem (Rivest, 2003). In 2009, RSA with 512 bits key was broken in 73 days and in 2010, a RSA number with 768 bits was factorized. With the recent emerge of quantum computer, many concerns appeared regarding the possibility to break RSA using it, but no practical attacks have been found until now.

2.4 NTRU Cipher

NTRU (US Patent, 1997), an alternative to RSA and ECC, is a public-key cryptosystem based on the shortest vector problem in a lattice (which is considered to be unbroken when using quantum computers) (Sakshaug, 2007). It includes two algorithms: *NTRUEncrypt* and *NTRUSign*.

The main advantage of it is the fact that it is resistant to attacks which use Shor algorithm. Due to the fact that the encryption and decryption processes are based on a simple polynomial multiplication, this cipher is much faster compared with RSA.

NTRUEncrypt algorithm was standardized for data encryption in 2011 by the Accredited Standards Committee X9 and is widely used in financial services industry.

2.5 DSP Platforms

Because intensive processing operations are performed during speech encryption algorithms (analysis, synthesis and encryption/decryption), it is best suited to implement them in dedicated DSPs. An important issue when dealing with DSPs, it to decide between floating point and fixed point computational core. Floating-point processors are not bit-exact, but they provide faster implementations that fixed-point processors. In this context, we have chosen *Blackfin ADSP-BF537*, which is fixed-point and *TMS320C6x*, which is floating-point.

The reason for choosing these DSPs were: high speed arithmetic, robust data transfer to and from real word, multiple access memory structure, less power and cheap.

2.5.1 Blackfin Processor

Blackfin processor (Reference Manual, 2013) has a high performance 32-bit embedded processor core, with a ten stage RISC pipeline and full SIMD (Single Instruction Multiple Data) support with instructions for accelerated and multimedia processing. Two 32-bit values can be read and written in a single clock cycle.

Other advantages of *Blackfin* processor are: supports instructions performed in parallel, multiple power-down modes for periods where there is little CPU activity and enables dynamic power management.

Blackfin contains an internal Analog-to-Digital Converter (ADC) and is much faster than microcontrollers. Also, we have chosen this processor due to its versatility in programming code, which means we can write code in *C/C++* and *LabVIEW*.

2.5.2 TMS320C6711 Processor

DSPs from *TMS320C6x* family (Reference Manual, 2005) are fast special-purpose microprocessors with specialized architecture and an instruction set dedicated for signal processing.

An advantage is the fact that the processor has integrated peripherals (host interface, external memory interface, multi-channel buffer serial ports, and memory direct interface). It includes the VLIW (Very-Long-Instruction-Word) technology, which means that the CPU fetches in advance very-long instruction words (256 bits) to provide eight 32-bit instructions during every clock cycle.

The DSP consists of eight parallel-operation functional units including two 16-bit multiplication units, and has a performance of 1600 MIPS at 200

MHz. The instruction processing system is of the VLIW pipeline type and can execute conditional operations and the maximum instruction code size is 64 Kbytes.

3 RELATED WORK

For a better understanding of the importance of our work, in this section the results obtained by other researchers are presented, as well as other speech encryption algorithm implementations.

Implementing cryptographic algorithms using dedicated VLSI (Very Large Scale Integration) hardware even though it provides high computing power, lacks flexibility and involves high investment costs. Other embedded hardware, such as DSPs represent a better solution because they offer higher flexibility compared with VLSI chips, provide more computing power than normal microprocessors and have a low cost due to mass production.

In (Wollinger, 2003) is presented an overview of cryptography in embedded systems. Moreover, in (Fiskiran, 2002), implementations of cryptographic algorithms in assembly and their optimizations using RISC instructions are described in details.

In (Wollinger, 2000) implementations of AES finalists (Towfish, RC6, Rijndael, Mars and Serpent) on TMS320C6201 processor is discussed and a comparison is made by taking into consideration criteria such as: total number of cycles and number of Mbit/sec for DSP multi-block mode and for DSP single-block mode. They obtained the best result for Twofish, an encryption speed of 139.1 Mbit/sec and a decryption speed of 148.8 Mbit/sec.

In (Thulasimani, 2010), AES implementations for keys of 128, 192 and 256 bits are presented for a single hardware unit.

In (Verna, 2012), the performance analysis (in terms of execution time and resource utilization) for three cryptographic algorithms: RC6, Twofish and Rijndael (the predefined key length for all of them is 128 bits) is described.

In (Itoh, 1999), public key algorithms such as RSA, Digital Signature Algorithm (DSA) and ECDSA were implemented on TMS320C6201 and a performance of 11.7 msec was obtained for 1024-bit RSA signing, 14.5 msec for 1024-bit DSA verification and 3.97 msec for 160-bit ECDSA verification.

Energy evaluation of software implementations of block ciphers is presented in (Grobschadl, 2007).

In our paper, we have developed hardware implementations for three cryptographic algorithms targeting embedded architectures and we have optimized the implementations for execution time and for power consumption.

To implement RSA on DSP is not trivial, because RSA is based on the theory of large prime factorization, which requires intensive modulo computations and also a large storage for big number processing. There are few papers in the specialized literature that describe the implementation of RSA on DSP hardware, such as (Yen, 2003), (Er, 1991). In (Yen, 2003), in order to improve the performance, additional DSP chips can be added and the application is controlled by a PC application through UART serial channels. In (Er, 1991), the authors have developed a RSA encryption module using Motorola 56300 DSP family, solution which is hard to integrate with existing e-commerce systems.

The main goal of (Ambika, 2012) is to illustrate some widely-used methods for secure speech communication systems starting with speaker identification and speech coding and ending with voice encryption and decryption (using symmetric or asymmetric algorithms).

In (Bassalee, 2008), the implementations of AES, DES, SHA1, TDEA, and ECDSA on Blackfin DSP are presented. The authors analyse their performance and try to reduce the encryption/decryption time and the energy consumed, by taking advantage of several architectural features that are available on the platform. They were able to reduce the energy consumption with almost 90% and to improve the execution time by a factor of 4.

4 IMPLEMENTATION

In this section we present the details regarding real-time and offline implementations of three speech encryption algorithms: AES, NTRU and RSA on two different DSP platforms: Blackfin ADSP-BF537 and TMS320C6711.

For the implementation of the cryptographic algorithms we have used as a baseline *VisualDSP++* software development environment. The main advantages are: optimizing C/C++ compiler, enhanced user-interface, statistical profiling tool, built-in performance analysis capabilities. Moreover, VisualDSP++ can be used to estimate with accuracy the energy consumption at instruction level.

At the beginning, the algorithms were tested offline, using a single processor so that we could verify if the implementations remain functional even when they are included in this VisualDSP++ environment from Microsoft Visual Studio 2015.

After thorough optimizations were performed at hardware and software level (to include all computation processes within the frame duration of 22.5 milliseconds), we were able to develop a real-time secure communication system. The block diagram of our system is presented in Figure 1.

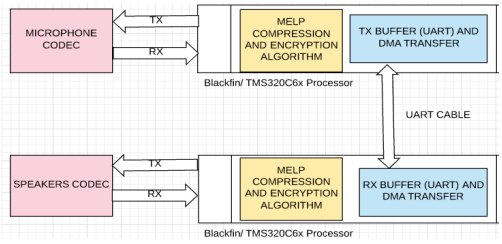

Figure 1: Block diagram of real-time secure communication system.

The voice analysis and synthesis algorithms based on LPC methods have been deeply tested and simulated before passing them to the DSP implementation phase. We have chosen MELP speech compression algorithm because it has some additional features, compared to LPC such as: *mixed-excitation* (reduces the buzz), *pulse dispersion* (disperses the excitation energy with a pitch period), *adaptive spectral enhancement* (provides a more natural quality to the speech signal) and *aperiodic pulses* (useful for transitions between unvoiced and voiced segments of signal).

The steps performed to ensure real-time secure communications are the following. The voice signal is taken from the microphone and converted into frames using the built-in ADC of the DSP, compressed and stored in the buffering memory. The buffered frame is encrypted and stored in another buffering memory and then is transmitted to the receiver. At the receiver end, the frame is decrypted and stored in a buffering memory. The decrypted frame is decompressed and then passed byte by byte to the DAC at a rate of 8 kHz and outputted to the speakers.

5 OPTIMIZATION AND EXPERIMENTAL RESULTS

In order to reduce the power consumption, we took into consideration optimizing the code in terms of execution speed. We have used a C compiler as a first step, because writing the entire program in assembly language will exceed the performance gain (little flexibility, lot of code and a large amount of time necessary to implement the cryptographic algorithms using assembler). After this, we created a detailed profile and identified which were the time critical code sections and wrote these in assembly language.

We developed C implementations for all mentioned cryptographic algorithms. We then used VisualDSP++ simulator to profile the execution of the implementations and to correctly and efficiently identify the code sections that can be improved using different optimization techniques.

For AES cipher, we have used a key of 128 bits and CBC mode of operation and for RSA, a key of 1024 bits.

We were able to observe from the beginning that the complexity and packet lost are approximately the same for all the algorithms and that the encryption/decryption/delay time varies in a way dependent of the number of bits per second. In this context, in order to determine a real difference between the algorithms' implementations, we only took into consideration the encryption time necessary for one frame, which is approximately the same as the decryption time.

Table 1 and Table 2 show the execution time in milliseconds, when running the implementations for the first time for compressing and encrypting a single frame on both DSP platforms.

Table 1: Execution time per frame before code optimization on Blackfin ADSP-BF537 processor.

Encryption Alg.	Execution time/frame (ms)
AES-128	140.125
RSA-1024	175.753
NTRUEncrypt	173.344

Table 2: Execution time per frame before code optimization on TMS320C6711 processor.

Encryption Alg.	Execution time/frame (ms)
AES-128	128.111
RSA-1024	157.220
NTRUEncrypt	153.488

At the beginning of the optimization, we studied the cryptographic routines starting with the algorithmic level, before going into low-level target-specific optimizations.

For instance, for *AES*, taking into consideration that the main part of the cipher is the *round transformation*, we simplified it to save execution time for real time implementation. More exactly, all operations of the round (substitution of bytes, shifting of rows, mixing of columns), were combined into a single set of look-up tables. We also stored the possible resulting terms after pre-computing the finite field multiplications (using only 1 Kbyte of memory).

In the key expansion function, most operations were implemented by 32-bit word exclusive OR.

The next step was to apply different optimization techniques at C level such as: using pragmas for optimizing loops, different memory banks, data alignment, no alias and speed. From the VisualDSP++ environment we enabled the optimization for C code (automatic inlining and interprocedural optimization). Also, in the implementations we have used volatile and static data types, arithmetic data types (int, short, char, unsigned int, unsigned char, unsigned short), as well as runtime C/C++ and DSP libraries, intrinsic functions and inline assembly.

An advantage of the inline assembly is that it can be used to rewrite a subroutine that involves overhead. The intrinsic functions allow to reduce the resource consumption because they are predefined functions, which are managed differently that other functions at compile time.

VisualDSP++ offers the option to use *Profile Guided Optimization (PGO)* which allows to collect data while the program is executing in order to identify code sections which are called most frequently. After using PGO, the execution time for all the implemented encryption algorithms decreased with approximately 50 milliseconds.

Table 3 and Table 4 show the execution time after applying optimization techniques at algorithm level and at C level.

Table 3: Execution time per frame after algorithm and C level optimizations on Blackfin ADSP-BF537 processor.

Encryption Alg.	Execution time/frame (ms)
AES-128	52.340
RSA-1024	66.977
NTRUEncrypt	63.893

Table 4: Execution time per frame after algorithm and C level optimizations on TMS320C6711 processor.

Encryption Alg.	Execution time/frame (ms)
AES-128	43.051
RSA-1024	57.115
NTRUEncrypt	54.209

As it can be seen, the smallest execution time is obtained for AES algorithm, which is followed by NTRUEncrypt algorithm. The highest execution time is obtained for RSA (67 ms for ADSP-BF537 and 58 ms for TMS320C6711) which is expected, taken into consideration that we have 1024-bit keys and we are doing multiplication operations.

Regarding the hardware level optimizations, we have rewritten time consuming code sections using assembly language, used hardware loops and parallel instructions, took advantage of the software pipeline provided by the DSP platforms and used special addressing modes (different data sections). Moreover, the less frequently accessed data was kept in SDRAM and the rest of the functions were cached.

Also, the energy consumed by a multi-issue parallel instruction is less than the energy consumed by the individual instructions that compose the multi-issue instruction. In this context, we tried to use parallel instructions as often as possible.

Table 5 and Table 6 show the execution time after applying optimization techniques at hardware level.

Table 5: Execution time per frame after hardware level optimizations on Blackfin ADSP-BF537 processor.

Encryption Alg.	Execution time/frame (ms)
AES-128	7.893
RSA-1024	10.644
NTRUEncrypt	8.053

Table 6: Execution time per frame after hardware level optimizations on TMS320C6711 processor.

Encryption Alg.	Execution time/frame (ms)
AES-128	5.662
RSA-1024	8.771
NTRUEncrypt	6.308

According to the results, after hardware level optimizations, AES still has the best execution speed (5.6 ms) and RSA still is the slowest (9 ms).

To be able to optimize the implementations of the encryption algorithms previously described, we identified which were the most time-consuming functions. The results are shown in Table 7 and Table 8, which specify the number of cycles consumed by each of the function before and after the optimization.

Table 7: CPU time before and after optimizations on Blackfin ADSP-BF537 processor.

Encryption Algorithm	Time functions	No opt.	With opt.
AES-128	Key expansion	3.54 Mcycles	120 Kcycles
RSA-1024	Key generation	5.67 Mcycles	202 Kcycles
	Modular multiplications	4.51 Mcycles	184 Kcycles
NTRU Encrypt	Key generation	5.43 Mcycles	188 Kcycles
	Polynomial multiplications	4.12 Mcycles	162 Kcycles

Table 8: CPU time per frame before and after optimizations on TMS320C6711 processor.

Encryption Algorithm	Time functions	No opt.	With opt.
AES-128	Key expansion	3.34 Mcycles	102 Kcycles
RSA-1024	Key generation	5.41 Mcycles	188 Kcycles
	Modular multiplications	4.23 Mcycles	165 Kcycles
NTRU Encrypt	Key generation	5.22 Mcycles	172 Kcycles
	Polynomial multiplications	3.89 Mcycles	148 Kcycles

For AES-128, the function which provides the key expansion consumes the most, more than 3 Mcycles. After all the optimizations, the results changed significantly to approximately less than 125 Kcycles.

In case of RSA and NTRU encryption algorithms, the most time-consuming functions are the key generation and the multiplication operations. After applying the optimizations, the number of cycles decreased with more than 5 Mcycles for both algorithms.

Based on the results in Table 7 and 8, we calculated the Clock Rate Reduction (CRR), which can be expressed as a percent of: the number of clock cycles before optimizations minus the number of cycles after optimizations and divided by the number of cycles after optimizations. The CRR values for the implemented encryption algorithms are presented in Table 9.

Table 9: CRR values for all speech encryption algorithms on both DSP platforms.

Encryption Algorithm	CRR ADSP-BF537	CRR TMS320C6711
AES-128	28.5%	31.74%
RSA-1024	27.06%	27.77%
NTRUEncrypt	27.88%	29.34%

In addition to all the previous described experiments, we performed a subjective analysis for the real-time implementations of the speech encryption algorithms, as it can be seen in Figure 2.

In this scenario, the quality of the signal is based on the listeners' opinion. We took into consideration 20 listeners, which had 15 distinct audio files encrypted for each algorithm and they had to give grades from 0 to 10. The best results were actually obtained by AES-12 8 cipher (8.35) and the worst results were obtained for RSA-1024 (7.65).

Figure 2: Subjective analysis scores for real-time implementations.

6 CONCLUSIONS

The main goal of the paper was to compare several speech encryption algorithms (symmetric and asymmetric ciphers) using two DSP platforms, one fixed-point processor (Blackfin ADSP-BF537) and one floating-point processor (TMS320C6711), in order to identify which is best suited for a real-time secure communication system.

We began with writing the implementations in C and then we ported the code on DSP processors. We started the optimization at algorithm level, continued at C level and ended with hardware optimizations, all of them being necessary to fulfil real-time requirements (the execution time per frame had to be smaller than the threshold 22.5 ms).

After thorough analysis, we can conclude that the block cipher AES is suited for real-time applications, because they have similar number of cycles and the smallest execution time per frame.

The public-key ciphers, RSA and NTRU, are also reliable and can be used, even though they are a little bit slower than the symmetric algorithms. Moreover, RSA and NTRU have a higher security level, so a trade-off between performance and security has to be made, taking into consideration the purpose of the real-time secure communication system that is being developed.

Based on the results of the subjective analysis, all the algorithms implemented have a good audio quality.

Referring to the choice of DSP platform, there are some differences between the two processors, but the execution time for the implemented algorithms doesn't change very much, so any choice is good for developing real-time secure communication applications.

We implemented an entire working system, which is not restricted to any specific medium and we respected all the security properties defined in the standards for the implemented cryptographic algorithms.

Our future work will include, modifying the filters in the compression function (using faster filters or filters that provide less losses) to decrease the execution time for the already implemented algorithms. Moreover, we intend to implement other voice encryption algorithms, with the purpose to provide a DSP platform that has security functions integrated and that can be used with trust to secure real-time sensitive communications.

ACKNOWLEDGEMENTS

This work has been funded by program Partnerships in priority areas – PN II carried out by MEN-UEFISCDI, project No. 47/2014.

REFERENCES

Pedre, S., Krajnik, T., Todorovich, E., Borensztejn, P., 2016. *Accelerating embedded image processing for real time: a case study*. In *Journal of Real-Time Image Processing*, Vol. 11, No. 2, pp. 349-374.

Joao, J. A., Mutlu, O., Patt, Y. N., 2009. Flexible Reference-Counting-Based Hardware Acceleration for Garbage Collection. In *Proceedings of ISCA 2009*, pp. 1-11.

Daemen J., Rijmen V., 2001. *Rijndael: The Advanced Encryption Standard*. In D r. Dobb's Journal, pp. 137-139.

Biryukov, A., Khovratovich, D., 2009. *Related-key Cryptanalysis of the Full AES-192 and AES-256*. In *Cryptology ePrint Archive: Report 2009/317*.

Bogdanov, A., Khovratovich, D., Rechberger, C., 2011. *Biclique Cryptanalysis of the Full AES*. In *Advances in Cryptology – ASIACRYPT 2011*, Vol. 7073, pp. 344-371.

Rivest, R., Shamir, A., Adleman, L., 1978. *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*. In *Communications of the ACM*, Vol. 21, No. 2, pp. 120-126.

Rivest, L. R., Kaliski, B., 2003. *RSA Problem*. In *Encyclopedia of Cryptography and Security, chapter RSA Problem.*

US Patent 6081597, 1996. *Public key cryptosystem method and apparatus*. In *Google Patents*.

Sakshaug, H., 2007. *Security Analysis of the NTRUEncrypt Public Key Encryption Scheme.* Available at https://brage.bibsys.no/xmlui/bitstream/handle/11250/258846/426901_FULLTEXT01.pdf (Last Accessed: October 2016).

ADSP-BF537 Blackfin Processor Hardware Reference Manual, Revision 3.4, 2013. Available at http://www.analog.com/media/en/dsp-documentation/processor-manuals/ADSP-BF537_hwr_rev3.4.pdf (Last Accessed: October 2016).

TMS320C6711, Floating-Point Digital Signal Procesors, 2005. Available at http://www.ti.com/lit/ds/symlink/tms320c6711c.pdf (Last Accessed: October 2016).

Wollinger, T., Guajardo, J., Paar, C., 2003. *Cryptography in embedded systems: An overview.* In *Proceedings of the Embedded World 2003.*

Fiskiran, M., Lee, R.B., 2002. *Workload characterization of elliptic curve cryptography and other network security algorithms for constrained environments.* In *Proceedings of the IEEE International Workshop on Workload Characterization (WWC-5)*, pp. 127-137.

Wollinger, T., Wang, M., Guajardo, J., Paar, C., 2000. *How well are high-end dsps suited for the aes algorithms? aes algorithms on the tms320c6x dsp*. In *AES Candidate Conference*, pp. 94-105.

Thulasimani, L., Madheswaran, M., 2010. *Design And Implementation of Reconfigurable Rijndael Encryption algorithms for Reconfigurable Mobile Terminals*. In *International Journal on Computer Science and Engineering*, Vol. 2, No. 4, pp. 1003-1011.

Verna, H.K., Singh, R. K., 2012. *Performance Analysis of RC6, Twofish and Rijndael Block Cipher Algorithms*. In *International Journal of Computer Applications,* Vol. 42, No. 16, pp. 1-7.

Itoh, K., Takenaka, M., Torii, N., Temma, S., Kurihara, Y., 1999. *Fast implementation of public-key cryptography on a dsp tms320c6201*. In *CHES '99: Proceedings of the First International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 61–72.

Grobschadl, J., Tillich, S., Rechberger, C., Hofmann, M., Medwed, M., 2007. *Energy evaluation of software implementations of block ciphers under memory constraints*. In *DATE*, pp. 1110–1115.

Yen, SM., Kim, S., Lim, S., Moon, S. J., 2003. *RSA speedup with Chinese remainder theorem immune against hardware fault cryptanalysis*. In *IEEE Transactions on Computers*, Vol. 52, No.4, pp. 461--472.

Er, M. H., Wong, DJ., Sethu, A., Ngeow, KS., 1991. *Design and implementation of RSA cryptosystem using multiple DSP chips*. In *IEEE International Symposium on Circuits and Systems*, Vol. 1, pp. 49--52.

Ambika, D., Radha, V., 2012. *Secure Speech Communication- A Review*. In *International Journal of Engineering Research and Applications*, Vol. 2, No. 5, pp. 1044-1049.

Bassalee W., Kaeli D., 2008. *Resource-Conscious Optimization of Cryptographic Algorithms on an Embedded Architecture*. Available at http://www.ece.neu.edu/groups/nucar/publications/ODSPES08bassalee.pdf (Last Accessed: October 2016).