Hindawi
Journal of Sensors
Volume 2019, Article ID 8520562, 9 pages
https://doi.org/10.1155/2019/8520562

Research Article

Influence of Encryption Algorithms on Power Consumption in Energy Harvesting Systems

Ljubomir M. Vračar[1], **Milan D. Stojanović**[1], **Aleksandar S. Stanimirović**[2] and **Zoran D. Prijić**[1]

[1]*Department of Microelectronics, University of Niš, Faculty of Electronic Engineering, Niš, Serbia*
[2]*Department of Computer Science, University of Niš, Faculty of Electronic Engineering, Niš, Serbia*

Correspondence should be addressed to Ljubomir M. Vračar; ljubomir.vracar@elfak.ni.ac.rs

Received 15 November 2018; Revised 8 March 2019; Accepted 21 March 2019; Published 8 April 2019

Academic Editor: Jaime Lloret

Autonomous energy harvesting sensors present one of the most attractive areas of microelectronics at the moment. They are a part of Internet of Things (IoT) systems so the data need to be protected across transmission. One way for data protection is encryption and the other way is digital signature. However, energy consumption of those systems is increased using protections algorithms, and it should be considered because these are energy harvesting systems. The paper describes the ways in which data encryption and digital signature algorithms can be implemented in resource limited systems based on an 8-bit microcontroller. Alongside the implementation method, the paper deals with the energy demands of the selected encryption algorithms and digital signatures. The execution time, energy consumption, and memory consumption will be considered.

1. Introduction

Continuous reduction of the microcontrollers' and sensors' energy consumption has boost development of various energy harvesting systems. Those are autonomous systems supplied by the energy gathered from their surroundings. The most common energy sources are the ambient light, thermal gradient, airflow, vibrations, and electromagnetic radiation. These devices have either a single power management circuit in order to reduce the complexity and dissipation or an optimized power management circuit for each energy source to increase efficiency of the whole system. They tend to work in low consumption mode whenever it is possible in order to increase their autonomy [1, 2]. Energy harvesting sensor nodes are organized in a sensor field around a base station that represents some kind of a data sink. Usually, every WSN node has just enough energy to periodically collect measured data and transfer collected data to the nearest base station in a single burst, using the single-hop infrastructureless architecture. Thus, an operating cycle of the node consists of the two operating modes: active and inactive. During the active mode, a node performs data acquisition and transmission. Usually, these operations last very short, around a few tens of milliseconds. WSN node spends much more time in the inactive mode in order to preserve energy and this period lasts from few seconds to few minutes. Ratio of the active and inactive periods represents an operating duty cycle of the node, in range approximately from 0.005% to 5%.

In addition, there is a need for data protection during transmission. Encryption and digital signature can be used for it [3–7]. Data are protected in this way, but energy consumption of system is increased because program complexity rises. Implementation of encryption algorithms in energy harvesting systems is mainly driven by WSN node energy limitations and characteristics of the system microcontroller. Energy consumption depends on the algorithm which has been used and the quantity of data transferred.

The lack of data and energy storage and low levels of electrical and computational performance in a node represent major obstacles to the implementation of established security techniques in wireless sensor network. Wireless sensor nodes often have very limited processing capabilities and the trend

in manufacturing is to reduce the cost of nodes, not to increase computing power. The unreliable communication channel and unattended operation make effective security even more difficult to achieve.

Concerning security aspects, the problem of optimizing resources used for security, yet providing an adequate level of protection, is a hot topic at the moment [8, 9]. Researchers have made considerable efforts to develop lightweight cryptographic algorithms suitable for resource-constrained devices. In particular, the trade-off between energy and performance requirements of security solutions is of utmost relevance for energy harvesting systems. Each adopted security solution should be a good compromise between factors that are conflicting in nature such as power consumption and performances.

2. Algorithms and Measurement Methods

Several algorithms of encryption can be implemented in autonomous sensors [10, 11]. It mainly depends on characteristics of microcontroller this is the part of system, in order to make the encryption run as efficiently as possible. Some of the algorithms for this application are TEA [12], XTEA [13, 14], and SKIPJACK. All listed are symmetric algorithms.

The name TEA is acronym of Tiny Encryption Algorithm, and adjective tiny indicates the simple way of doing and implementation. TEA algorithm encrypts data that are divided into 64-bit blocks using 128-bit key, and it was based on Feistel's network (Feistel cipher). It was initially proposed to carry 64 rounds within the TEA algorithm; however, implementation with 32 rounds is common. Disadvantage of TEA is simple distribution of keys. Equivalent keys are the consequence of that; each key is equal to the other three.

XTEA (eXtended Tiny Encryption Algorithm) presents advanced version of TEA developed by David Wheeler and Roger Needham in 1997. Differences between TEA and XTEA are in the order of performing operations and in the way the keys are distributed. XTEA requires a small memory space so it is often used in systems whose hardware features are limited.

SKIPJACK algorithm was developed in the early 1980s by the American National Security Agency for encryption of Government's documents. It runs within 32 rounds and encrypts 64-bit data blocks using an 80-bit key.

2.1. Digital Signatures Algorithms. Unlike encryption, in digital signature, the data transmitted can be encrypted or may remain transparent and it is possible to determine the sender of the message. The digital signature algorithm consists of three parts: keys generating, signing, and signature verification. Some of digital signature algorithms are RSA [15], ElGamal [16], Rabin, Schnorr, and Nyberg-Rueppel. In this paper RSA and ElGamal algorithms were used.

RSA digital signature algorithm is based on RSA asymmetric encryption algorithm [17]. Today, this algorithm is most used. Key generating in this algorithm is based on calculating with large prime numbers p and q. It is recommended that p and q be the size of 512 to 1536 bits, to ensure proper security. Of course, this applies to computer systems that do not have memory constraints. When the RSA digital signature algorithm is implemented on a microcontroller, the sizes of the keys expressed in bits must be considerably smaller (the numbers p and q are 16-bit in size).

ElGamal's digital signature algorithm was developed in 1985. The working principle of this algorithm is based on calculating discrete logarithms. The size of initial parameter p for keys generation is 16 bits like the parameter in RSA algorithms. Digital signature is based on asymmetric encryption algorithms and implementation of those algorithms on microcontroller is more complicated than symmetric ones because they are based on complex mathematical computation. However, we decided to use two digital signatures algorithms and three symmetric encryption algorithms in order to explore implementation possibilities of both algorithms. Before encryption the data are divided into 64-bits block for all three encryption algorithms. Only one of these blocks was used in the research because it is sufficient to represent data obtained from sensor (temperature sensor LM35). For sensors with the amount of output data (encryption input data) larger than 64 bits, it is necessary to divide data in the blocks and perform encryption for every block separately. Total execution time and system energy consumption in that case would be increased linearly.

Figure 1 shows algorithms that present ways in which a digital signature can be implemented in a microcontroller. These algorithms can be applied to any digital signature algorithm.

The most simplified algorithm is presented on the right side of Figure 1. For this reason, this digital signature algorithm is implemented on a microcontroller when research was made. Size of firmware memory of the selected microcontroller is 32 kB, and therefore firmware should be optimized before uploading. Keys for both RSA and ElGamal digital signatures were generated externally, in order to optimize size of firmware, total execution time, and system energy consumption. Generated keys are written in transmitter and receiver. The algorithm on the right and the algorithm in the middle of Figure 1 have equal level of security, but the right one has certain advantages: shorter execution time and smaller energy consumption.

3. Implementation Case Study

To evaluate experimental data, a microcontroller system based on PIC18F45K22 [18] with an external ST-TX03-ASK 434 MHz radio module has been constructed. Current consumption was measured with custom made current probe based on the integrated circuit INA212 [19] using oscilloscope. One of the microcontroller pins was used as a "trigger" pin during firmware execution, providing accurate time stamp for measurements.

Figure 2 presents simplified schematics of the system which was used for the energy consumption measurements. Firmware for the microcontroller was written in Proton Basic Compiler.

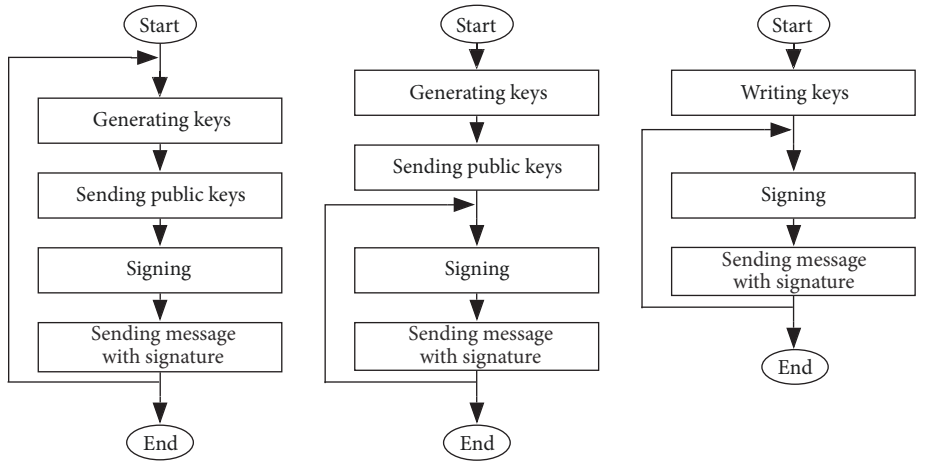


Figure 1: Implementation of digital signature algorithm in PIC microcontroller.

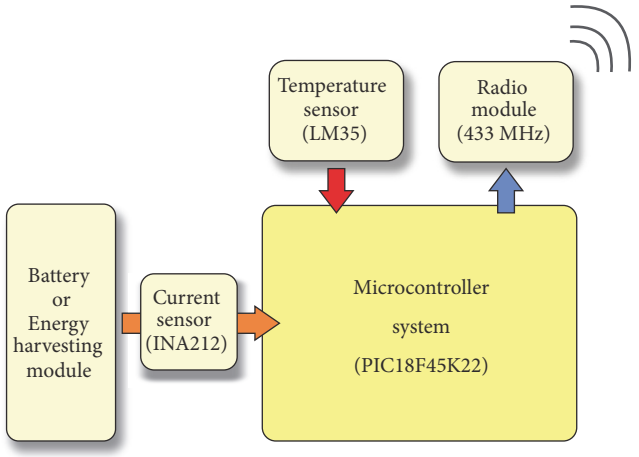


Figure 2: Simplified schematic of the system.

The system presented in Figure 2 is an autonomous system and it should perform encryption data before sending. Receiver is at the other side of communication channel. It receives data and performs decryption of them. In our research only the execution time and system energy consumption of transmitter side is of interest. The measurement set-up also provides the duration of time intervals in which encryption algorithms were executed. Figure 3 presents the example of captured waveforms recorded by an oscilloscope during the experiment.

The red signal (the upper one, CH2) is the voltage on the trigger pin and it changes its state every time the new operation within the software is executed. The blue signal presents the current and is recorded by the current probe. It is obvious that current consumption is the highest during the sending of the data. It is necessary to separate encryption from the sending signal and to determine the number of measurings for each operation. As an example, Figure 4 shows a part of the signal where we can see that the encryption starts after the 467^{th} and ends before the 480^{th} sample of measuring.

To estimate the energy consumption, it is necessary to determine the surface below the given curve. In other words, it is necessary to calculate the integral (1) for the captured waveforms, where the supply voltage is constant and its value is 3.3 V.

$$W = \int V(t)\, I(t)\, dt \ [J] \tag{1}$$

$$V = const = 3.3\, V \tag{2}$$

$$W = 3.3 \int I(t)\, dt \ [J] \tag{3}$$

3.1. Influence of Encryption Algorithms on Execution Time. Measuring time helps determining which encryption algorithm and which digital signature take the shortest execution time. Encryption time is measured for the implementation of the three previously named algorithms. For the same algorithms the time of data sending was also measured. Figure 5 shows the signals on which the measuring was based.

The Figure 5 shows that the firmware execution time of these encryptions is similar, especially comparing TEA and XTEA algorithms (the scale of the time axis of the oscilloscope is the same for each algorithm). Figure 6 shows the results of these measurings.

According to the values measured, the process of data encryption takes much less time than data sending. The size of input data, keys, and output data is always the same. Therefore, the conclusion is that the encryption time for the same algorithm depends neither on the key values nor on the input data value. When there is no encryption, only the measured temperature value is sent, which is an eight-bit variable. On the other hand, when the measured value of temperature is encrypted, the output of the given encrypted algorithms is 64-bit data. The encryption time and the data sending time make the total software execution time (Figure 7).

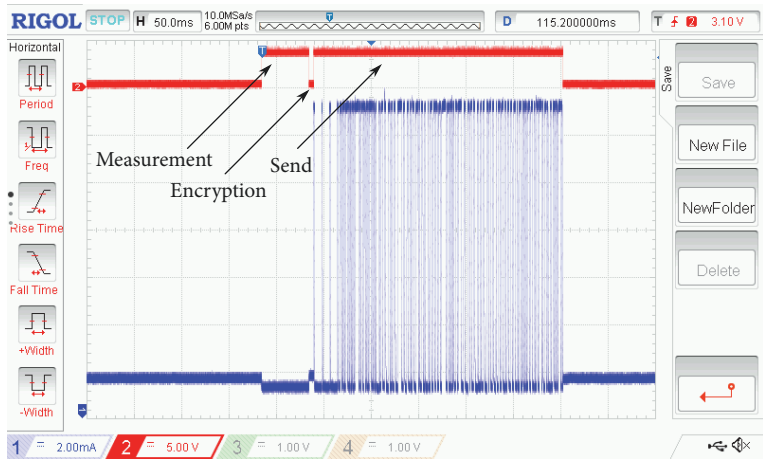


Figure 3: Captured waveforms: data preparation, encryption, and data sending.

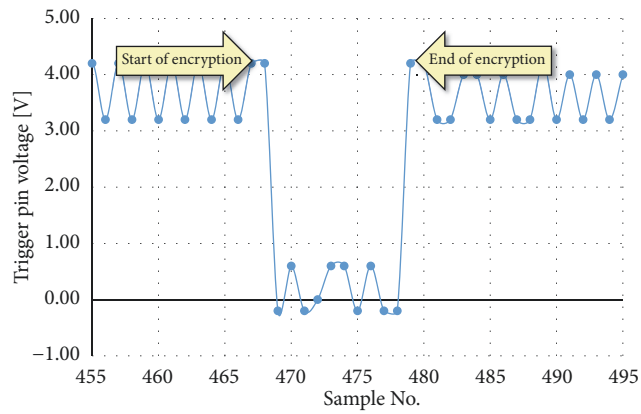


Figure 4: State of trigger pin at start and at the end of encryption.

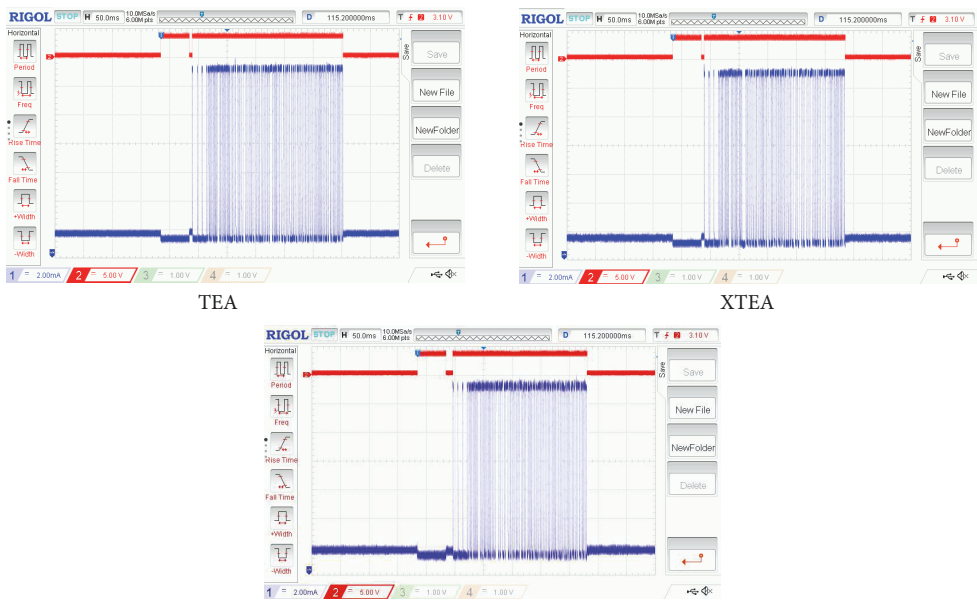


Figure 5: Captured waveforms during execution of encryption algorithms.

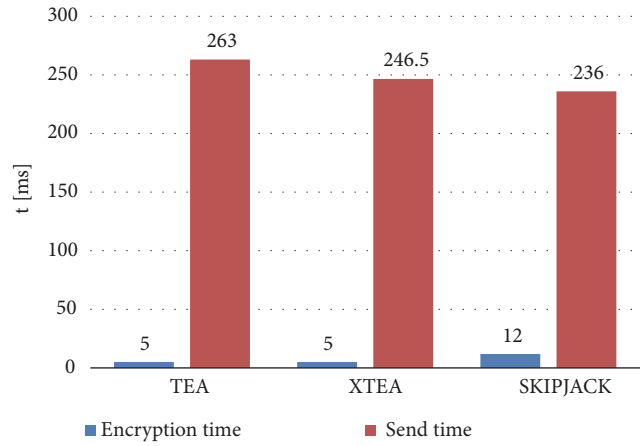


Figure 6: Firmware execution time of encryption and data sending (in ms).

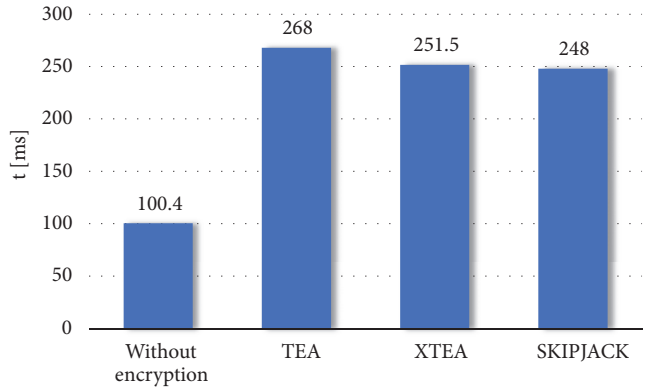


Figure 7: Total firmware execution time (in ms).

3.2. Influence of Digital Signature on Execution Time. The first step in keys generation for digital signature is choosing random prime numbers. The keys for RSA and ElGamal algorithm were generated by choosing the value for number p, while other values were calculated. Figure 8 presents the signals recorded at the lowest and the highest values of p. For the RSA digital signature, the value on the time axis of the oscilloscope, when p=17, is 20 ms, and when p=991, it is 100 ms. The picture also shows that the software execution time is much longer in the second case. For the ElGamal digital signature the execution time is also longer for higher key values. However, this difference is less obvious compared to the RSA algorithm.

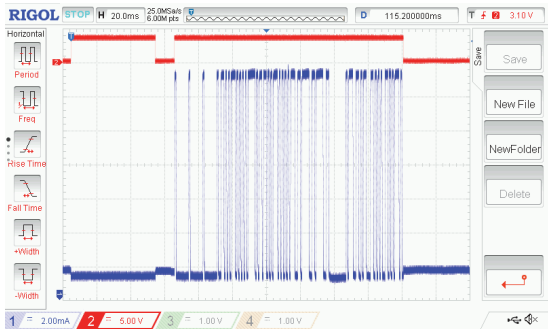
Figure 9 presents dependence of signing time on the value of the initial p parameter, for RSA and ElGamal algorithm.

The picture shows that, with the RSA algorithm, signing time rises exponentially with the increase of the p parameter, while the change of the signing time with the ElGamal algorithm is considerably smaller, depending on the same parameter. The other important parameter is time needed to send the data and signatures. Figure 10 presents the dependence of data and signature sending time on p parameter.

The change in data and signature sending time is much smaller compared to the signing time at both algorithms. Signing time and sending time make the total software execution time. The values gained are presented in the Figure 11.

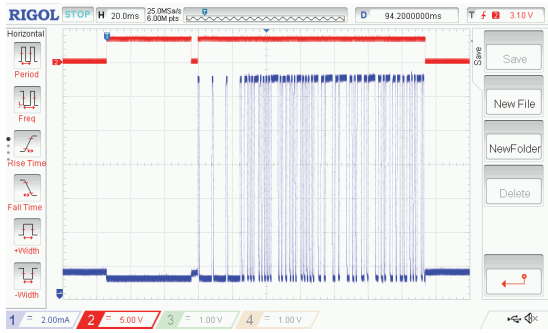
3.3. Influence of Encryption Algorithms on Current Consumption. Calculation of the current consumption was done in order to determine which of the given algorithms has the smallest influence on the high energy consumption. The calculation was done by the previously described method (1), and therefore, all of the following values will be presented in μJ. When the system executes software without algorithm encryption or signing, the consumption is 3352 μJ. Figures 12 and 13 present the calculated values of energy consumption for the encrypting and data sending, respectively.

The diagrams show that the consumption value of each algorithm depends on its duration time; thus, the algorithm that lasts longest needs most energy. When the consumption values for encrypting and sending are compared, the results show that the system needs much more energy to perform data sending because the process of sending takes more time than encrypting. By adding these two values total system energy consumption is obtained. This value is of high importance when choosing the encryption algorithm which is to be implemented in the system. Figure 14 presents the values of

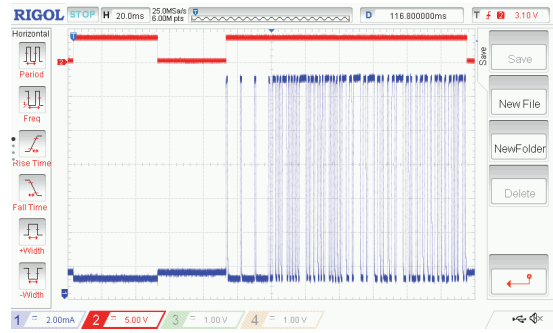


RSA_p=17

RSA_p=991



ELGamal_p=17



ELGamal_p=991

Figure 8: Captured waveforms during digital signature.

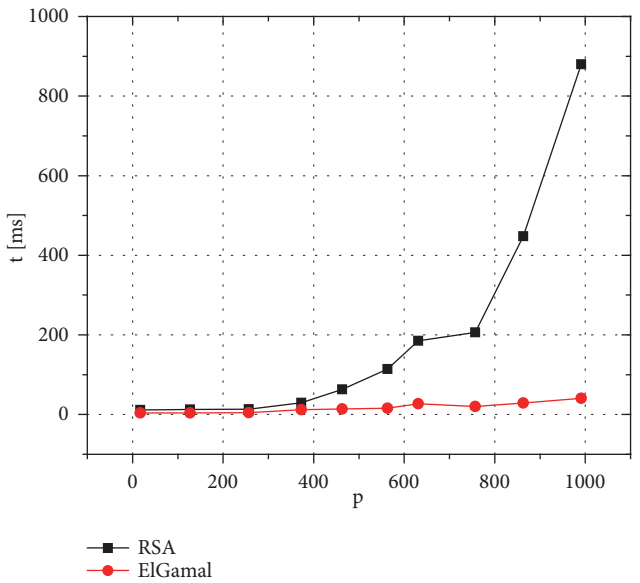


Figure 9: Signing firmware execution time (in ms).

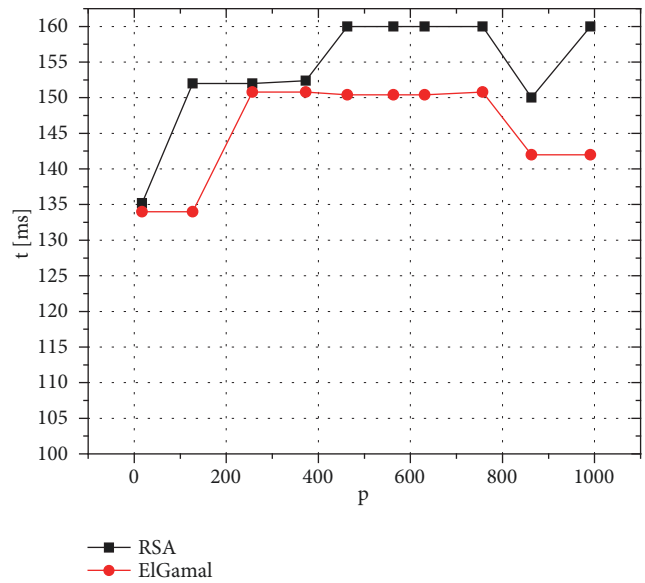


Figure 10: Firmware execution time (in ms) of data and signature sending.

total consumption for each of the given algorithms, as well as the values of consumption when these algorithms are not included in the software implemented in the microcontroller.

Measuring proved that the consumption depends on encryption time. Time for performing algorithm encryption does not depend on the values of the data which are encrypted, nor on the key values. Therefore, the conclusion is made that the encryption consumption has the constant value for the given algorithm.

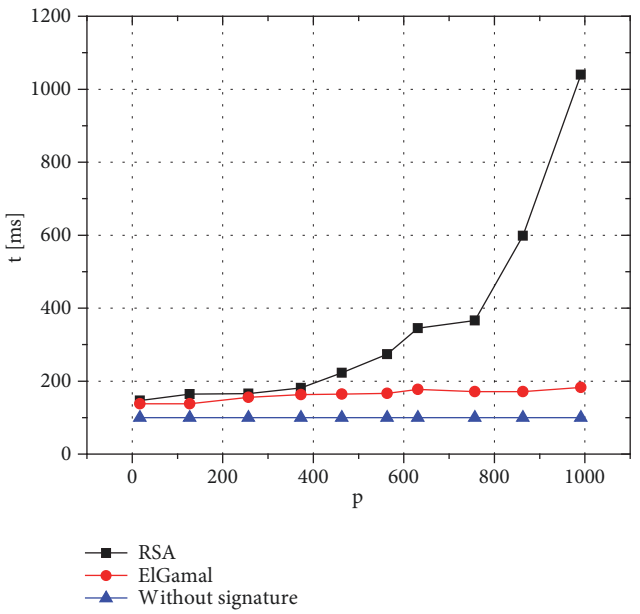


Figure 11: Total firmware execution time (in ms) with signature.

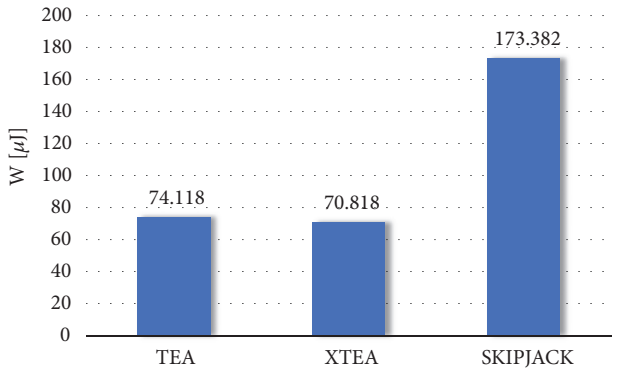


Figure 12: Energy consumption during encryption (in μJ).

3.4. Influence of Digital Signature on Current Consumption. Considering digital signature, the consumption was calculated for the same key values of RSA and ElGamal algorithms, for which the execution time was measured. Figure 15 presents dependence of consumption at calculating signatures on the key values, i.e., on the initial p parameter. Time of data sending with the signature depending on the same parameter p is presented in Figure 16.

Energy consumption needed for signing has the same dependence on the p parameter as the firmware execution time. The same dependence can be found regarding the energy consumption for sending data and signatures. Thus, just like with the encrypting algorithms, consumption depends on the operation execution time. Total system consumption is obtained by adding signature consumption and sending consumption (Figure 17).

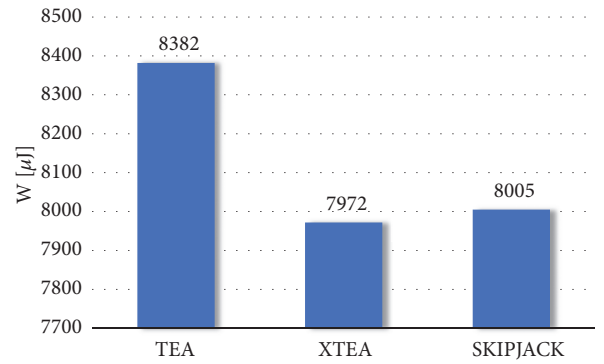


Figure 13: Energy consumption during data sending (in μJ).

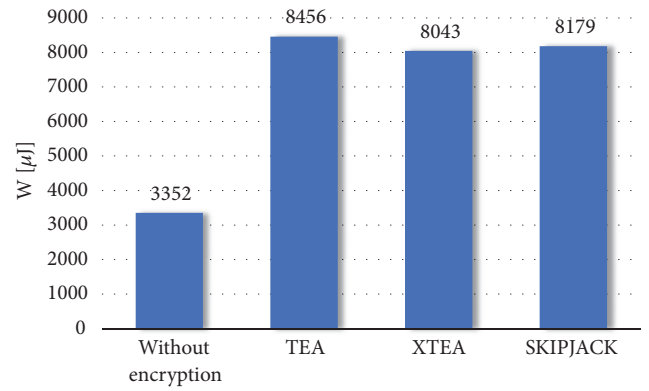


Figure 14: Total system energy consumption depending on encryption algorithm (in μJ).

4. Conclusion

The paper describes the ways in which data encryption and digital signature algorithms can be implemented in resource limited systems based on an 8-bit microcontroller. Alongside the implementation method, the paper deals with the energy demands of the selected encryption algorithms and digital signatures. The execution time, energy consumption, and memory consumption have been considered. All the selected algorithms can be applied in WSNs unless the memory deficiency makes it impossible.

Memory usage for the implementation of encryption and digital signature algorithms is as follows: TEA, 6.07%; XTEA, 5.15%; SKIPJACK, 5.26%; RSA, 13.81%; and ElGamal, 19.34%. XTEA symmetric encryption is the best solution of the three considered memory usage. Total firmware execution time is the shortest when SKIPJACK algorithm (248 ms) is implemented and the largest with TEA encryption algorithm (268 ms). XTEA algorithm consumes the smallest amount of energy (8043 μJ) compared to TEA (8456 μJ) and SKIPJACK (8179 μJ) for encryption and sending data of the same size and same initial values.

Security of digital signatures algorithms depends on keys size. In this case, keys size for both RSA and ElGamal algorithms are equal. Thus, we can say that implementations of these digital signatures in microcontroller PIC18F45K22

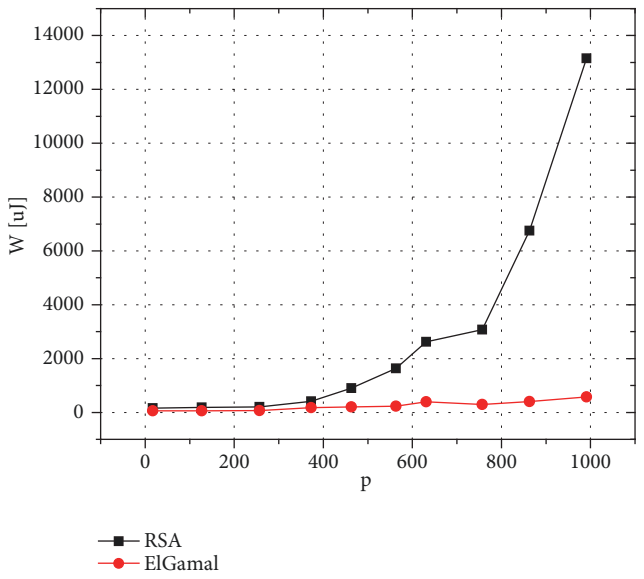


Figure 15: System energy consumption for digital signature algorithms (in μJ).

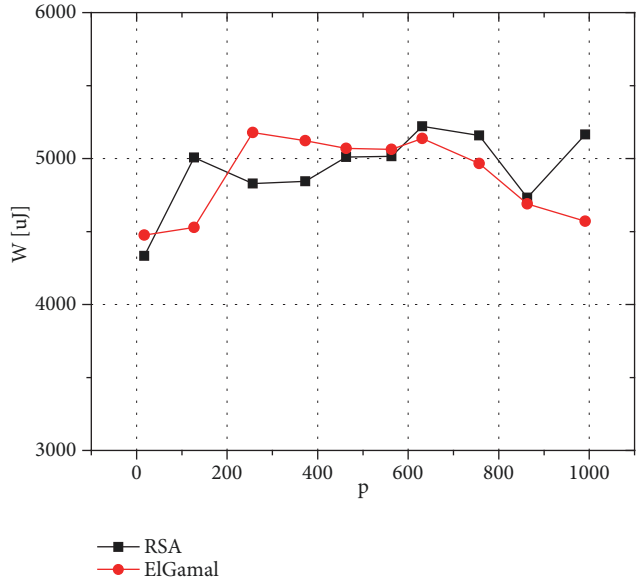


Figure 16: System energy consumption during data sending with signature (in μJ).

have equal level of security, but the execution time and the energy consumption make the ElGamal algorithm more advantageous.

Further research could be based on advanced microcontrollers that provide the higher operating frequency and lower energy consumption, which is necessary in case of the most advanced security algorithms.

Data Availability

As corresponding author, I can provide original measurements data captured with oscilloscope (Figures 5 and 8) upon request. All the other figures are simple representation of the calculated data. I cannot provide complete source code for embedded software because it will be implemented in some commercial product that we are developing.

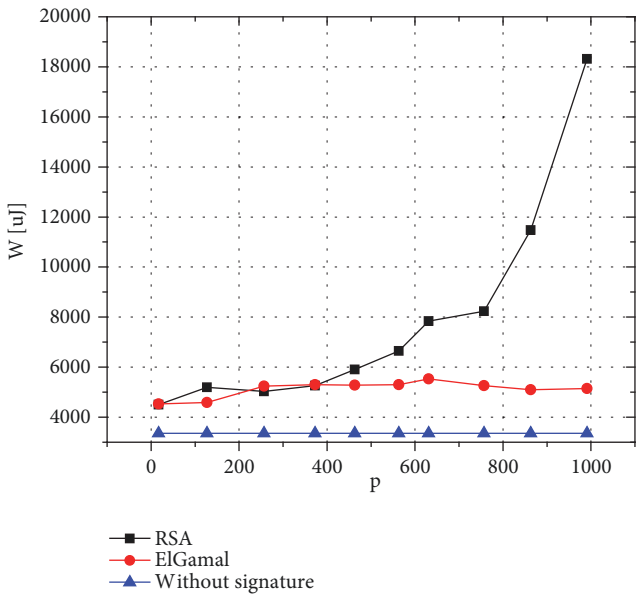


Figure 17: System energy consumption for digital signature algorithms (in μJ).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the Serbian Ministry of Education, Science and Technological Development under Grant TR32026 and in part by Ei PCB Factory, Niš, Serbia.

References

[1] X.-F. Zhang and C.-C. Yin, "Energy harvesting and information transmission protocol in sensors networks," *Journal of Sensors*, vol. 2016, Article ID 9364716, 5 pages, 2016.

[2] F. Engmann, F. A. Katsriku, J. Abdulai, K. S. Adu-Manu, and F. K. Banaseka, "Prolonging the lifetime of wireless sensor networks: a review of current techniques," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 8035065, 23 pages, 2018.

[3] O. Alfandi, A. Bochem, A. Kellner, C. Göge, and D. Hogrefe, "Secure and authenticated data communication in wireless sensor networks," *Sensors*, vol. 15, no. 8, pp. 19560–19582, 2015.

[4] J. M. Kim, H. S. Lee, J. Yi, and M. Park, "Power adaptive data encryption for energy-efficient and secure communication in solar-powered wireless sensor networks," *Journal of Sensors*, vol. 2016, Article ID 2678269, 9 pages, 2016.

[5] Y. Lu, J. Zhai, R. Zhu, and J. Qin, "Study of wireless authentication center with mixed encryption in WSN," *Journal of Sensors*, vol. 2016, Article ID 9297562, 7 pages, 2016.

[6] S.-H. Ju, H.-S. Seo, S.-H. Han, J.-C. Ryou, and J. Kwak, "A study on user authentication methodology using numeric password and fingerprint biometric information," *BioMed Research International*, vol. 2013, Article ID 427542, 7 pages, 2013.

[7] I. Froiz-Míguez, T. M. Fernández-Caramés, P. Fraga-Lamas, and L. Castedo, "Design, implementation and practical evaluation of an iot home automation system for fog computing applications based on MQTT and ZigBee-WiFi sensor nodes," *Sensors*, vol. 18, no. 8, 2018.

[8] R. Lacuesta, J. Lloret, M. Garcia, and L. Penalver, "Two secure and energy-saving spontaneous ad-hoc protocol for wireless mesh client networks," *Journal of Network and Computer Applications*, vol. 34, no. 2, pp. 492–505, 2011.

[9] R. Lacuesta, G. Palacios, and L. P. Herrero, "A protocol to the auto-configuration of securing and distributing spontaneous networks," in *Proceedings of the International Conference on Wireless Networks (ICWN '10)*, vol. 2, Las Vegas, NV, USA, July 2010.

[10] R. Lacuesta, J. Lloret, M. Garcia, and L. Peñalver, "A secure protocol for spontaneous wireless Ad Hoc networks creation," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 4, pp. 629–641, 2013.

[11] N. A. Alrajeh, S. Khan, J. Lloret, and J. Loo, "Artificial neural network based detection of energy exhaustion attacks in wireless sensor networks capable of energy harvesting," *Ad Hoc & Sensor Wireless Networks*, vol. 22, no. 1-2, pp. 109–133, 2014.

[12] D. J. Wheeler and R. M. Needham, "TEA, a tiny encryption algorithm," in *Fast Software Encryption: Second International Workshop*, vol. 1008, pp. 363–366, Leuven, Belgium, December 1994.

[13] R. M. Needham and D. J. Wheeler, *TEA Extensions*, Computer Laboratory Cambridge University England, 1997.

[14] R. M. Needham and D. J. Wheeler, *Correction to XTEA*, Computer Laboratory Cambridge University England, 1998.

[15] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[16] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469–472, 1985.

[17] B. Kadri, M. Feham, and A. M'hamed, "Lightweight PKI for WSN uPKI," *International Journal of Network Security*, vol. 10, no. 2, pp. 135–141, 2010.

[18] "PIC18F45K22," Datasheet, Microchip Tecnology Inc., 2010, https://ww1.microchip.com/downloads/en/DeviceDoc/40001412G.pdf.

[19] "INA212," Datasheet, Texas Instruments., 2017, http://www.ti.com/lit/ds/sbos437j/sbos437j.pdf.