Transport and Communications Science Journal

IMPLEMENTATION OF AUTHENTICATED ENCRYPTION WITH ASSOCIATED DATA GRAIN-128AEAD ALGORITHM ON STM32F400 PROCESSOR FAMILY

Nhu Quynh Luc*, Thi Nga Tran, Cong Khanh Ngo, Huy Duc Tran, Van Chien Nguyen, Tien Anh Tran

Academy of Cryptography Techniques, 141 Chien Thang Road, Tan Trieu, Thanh Tri, Hanoi, Vietnam

ARTICLE INFO

TYPE: Research Article
Received: 15/11/2021
Revised: 25/01/2022
Accepted: 30/01/2022
Published online: 15/05/2022
https://doi.org/10.47869/tcsj.73.4.7

* *Corresponding author*
Email: quynhln@actvn.edu.vn; Tel: +84 984180146

Abstract. An embedded device is becoming popular in daily life thanks to their low power consumption and multiprocessing capability. In particular, the security of embedded devices has been a big issue of concern to academic and industrial communities. This study aims at the Grain 128-AEAD authenticated encryption with associated data algorithm embedded on low-power and resource-constrained devices. This stream cipher belongs to the Grain family developed from the Grain-128a algorithm, and it has the advantage of not only providing security, but also adding authenticity to the associated data to ensure the authenticity, integrity and confidentiality of the data. It is also considered suitable for IoT (Internet of Thing) platforms and embedded device applications with limited resources and low power consumption. In this study, the algorithm was implemented on STM32 processor family. The resulting code size is only 832 bytes, and the total execution time for a 128-byte input block of Grain-128AEAD algorithm (Encryption and Decryption) takes 30 µs, which is better than previous implementations on various hardware platforms. The compiled file size is only 54kB, which makes the algorithm fit embedded applications.

Keywords: Light-weight cryptography, IoT security, stream ciphers, Grain-128AEAD algorithm.

1. INTRODUCTION

Nowadays, embedded devices become popular in daily life because of their reasonable prices, low power consumption and multiprocessing capability [1], [2], [3], [4]. However, the security of embedded devices in recent years has been an issue of concern to researchers [5]. Many stream ciphers or block ciphers have entered competitions to select the most suitable algorithms that meet the following criteria: low power consumption and execution capability with limited resources and memory [1], [3]. In [6], Alexander Maximov and Martin Hell showed that lightweight stream ciphers are more appropriate than lightweight block ciphers to optimize energy when encrypting longer messages, for the execution can be sped up without increasing hardware costs.

The Grain-128AEAD (Authenticated Encryption with Associated Data) algorithm is currently the 2nd round candidate of the selection contest as a lightweight cryptographic complying to standards by NIST (National Institute of Standards and Technology) [1], [3], [7], [8]. Grain-128AEAD, which is based on the Grain-128a algorithm [1], [9], belongs to the Grain family and was published on eSTREAM by Martin Hell, Thomas Johansson and Willi Meier in 2004 with the first version Grain v0 [10]. Later it was developed into Grain v1, which is one of seven projects that eSTREAM catalogued for continue development since September 9, 2008, with a key length of 128 bits and an initialization vector IV (96-bit). However, all three versions only support encryption without an authentication mechanism. It was not until Grain-128a version that started to support authentication, and the last version was Grain-128AEAD which can be said to be the complete version of the Grain family. The Grain-128AEAD Algorithm is a stream cipher that supports authenticated encryption of associated data, also resistant to the attacks shown in earlier version [3], [6], [11], [12].

The main idea of this study is to embed the Grain 128-AEAD algorithm on resource-constrained devices. The rest of the study will be divided as follow: Section 2 will discuss previous works related to authenticated encryption and decryption with associated data, then the 128AEAD algorithm design will be presented. The algorithm implementation will be demonstrated in section 3, including comparison between results on the computer, on the STM32 microprocessor and on other hardware platforms. The final part will summarize the achieved results and point out directions for further study.

2. RELATED WORKS

2.1. AUTHENTICATED ENCRYPTION AND DECRYPTION WITH ASSOCIATED DATA

In [1], Hell, M., Johansson et al. have detailed the mathematical proof for the Grain-128AEAD algorithm and the corresponding algorithm schema. Figure 1 shows that the algorithm consists of two main functional blocks: the first block generates a random bit-stream used for encryption and authentication code generation; the second block is used to generate the token for authentication. The first block consists of two 128-bit registers, a Linear-Feedback Shift Register (LFSR) and a Nonlinear-Feedback Shift Register (NFSR), then a Boolean function to combine the output of LFSR and NFSR. The second block consists of a 64-bit shift register and a 64-bit accumulator. By using both LFSR and NFSR, the nonlinearity of the key stream generator can be increased, which made it an advantage over other stream ciphers' in terms of security and execution speed [8], [15], [11].

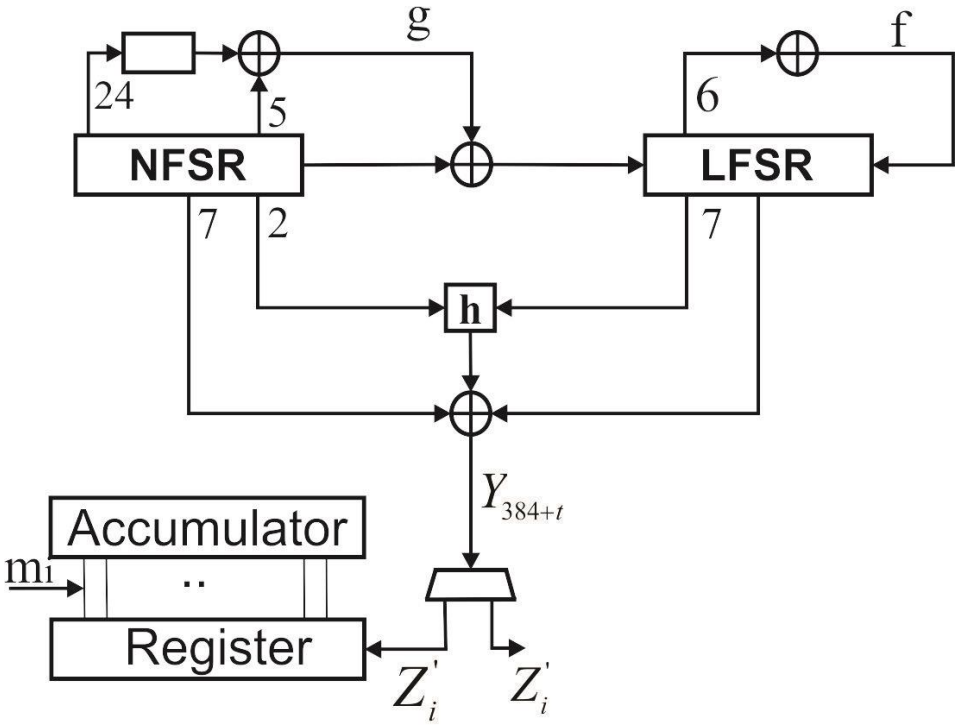

Figure 1. Function blocks of the Grain-128AEAD algorithm [1].

Figure 2 shows the initialization process scheme in the Grain-128AEAD algorithm. Before the pre-output is used as the keystream for encryption/decryption and for authentication, the internal states of the pre-output generator and the authentication generator registers are initialized with key and nonce.

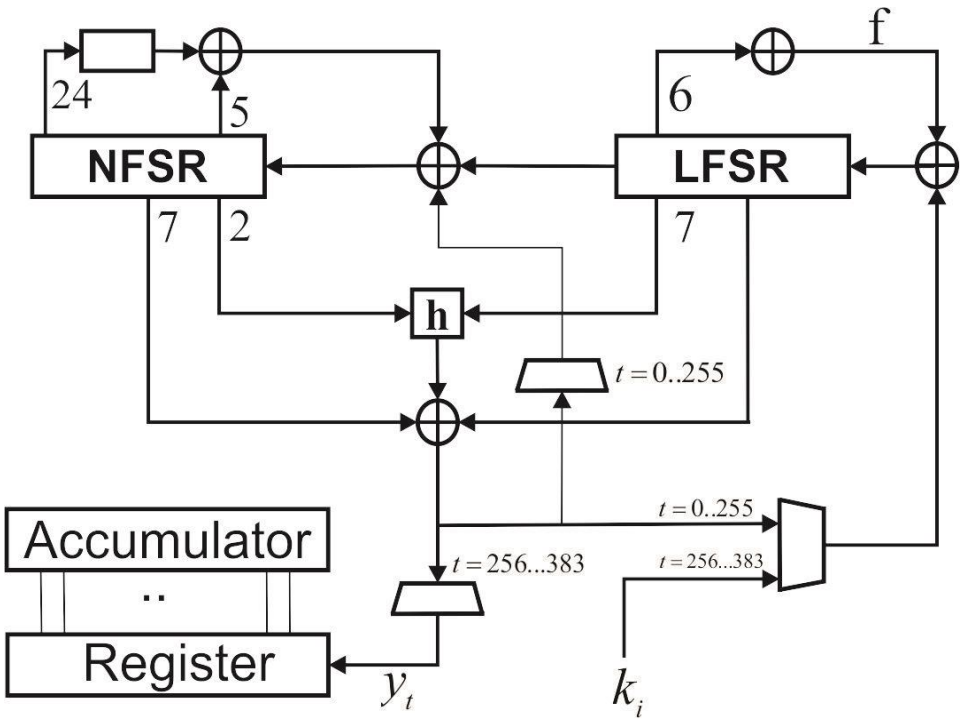

Figure 2. Initialization process of Grain-128AEAD [1].

For initialisation, the first 128-bits of the NFSR register are used to initialise the key, where the first 96-bits of the LFSR register are used to initialise the nonce, the next 31 bits are filled with 1s, and the last bit is bit 0. Next, the encryption algorithm is executed 256 times and each return output will be XOR-ed with the inputs of LFSR and NFSR. After the initialization of previous output set, the authentication module is initialised by having the first 128 bits of the output block generated from the first block loaded to the 64 bits of the shift register and the 64 bits of the accumulator, where the first 64 bits are loaded into the adder and the last 64 bits goes to the shift register. The last 128 bits of the output block are used for encryption and authentication.

2.2. DESIGN AND ANALYSIS OF THE GRAIN-128AEAD ALGORITHM FOR AUTHENTICATED ENCRYPTION AND DECRYPTION WITH ASSOCIATED DATA

Authenticated Encryption with associated data (denoted AEAD) [16] is a form of symmetric key cryptosystem that ensures confidentiality, integrity, and data authenticity at every step. In which, the encryption will combine with the AEAD mask block for integrity check, while the decryption process will check the received AEAD mask block. Confidentiality protects information by converting the input plaintext into independent random bits, while authenticity ensures the integrity and originality of the data by detecting any changes to the data [1].

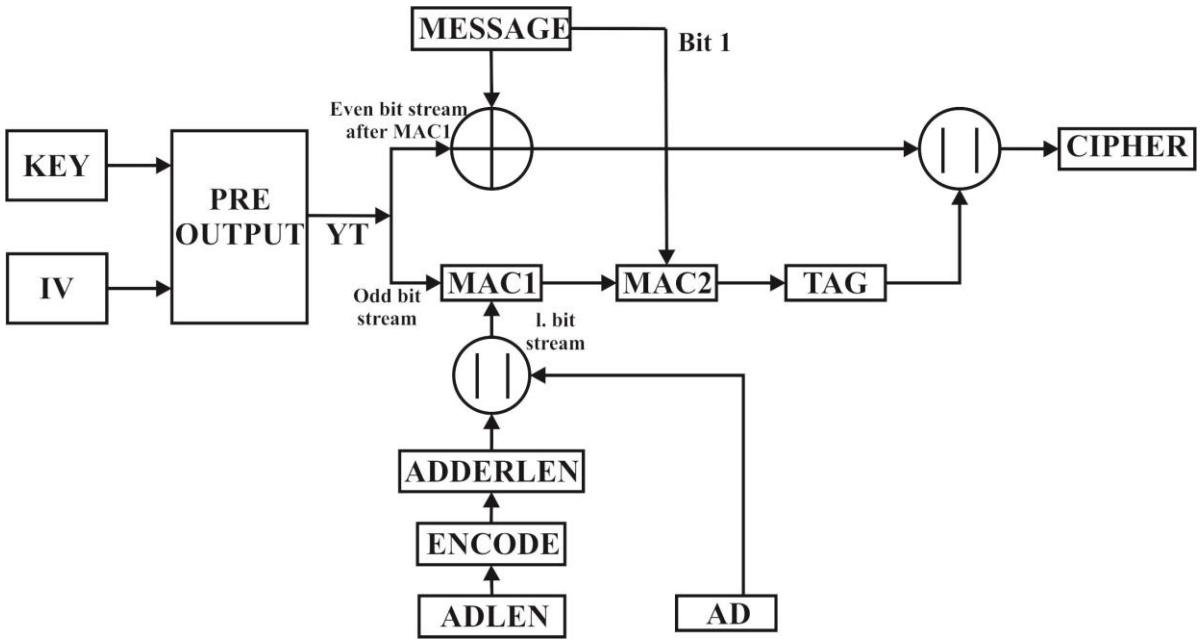

Figure 3. Authentication encryption scheme with associated data of Grain-128AEAD.

Figure 3 shows the authenticated encryption mode with AEAD associated data of the Grain-128AEAD algorithm. In the encryption mode the input of the algorithm includes *ad, adlen, m, mlen, k, nonce*; the output of the process is *ciphertext c*. Similarly, the decryption mode of Grain-128AEAD algorithm is shown in Figure 4.

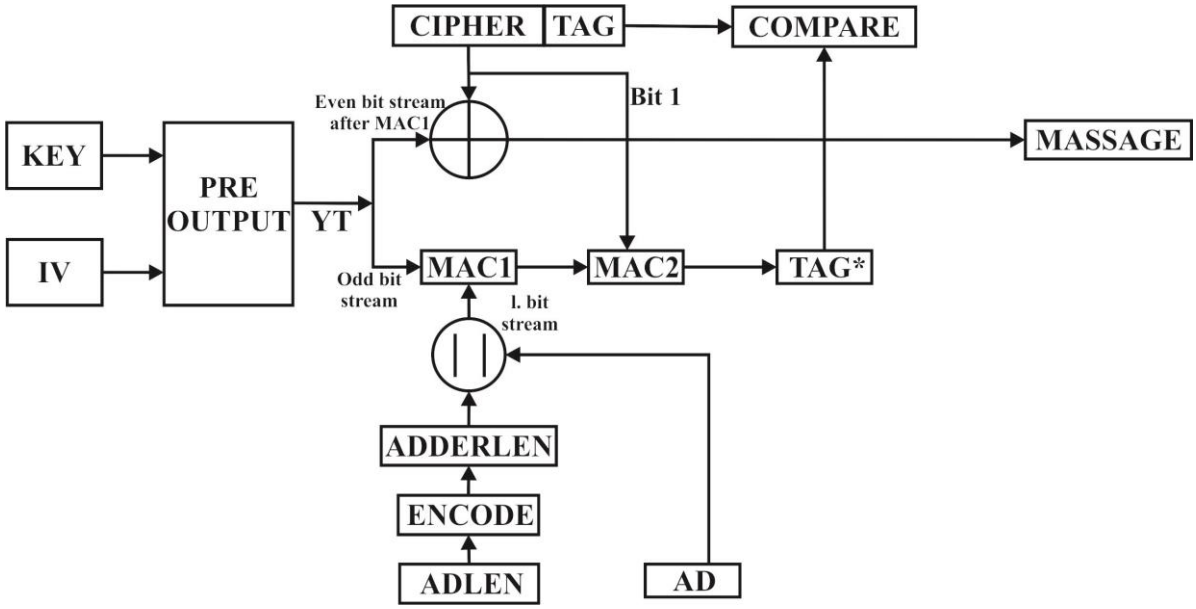

Figure 4. Authentication decryption diagram with associated data of Grain-128AEAD.

The advantage of the Grain-128AEAD algorithm is that its encryption mode is simple by encoding each bit stream of the message while a key bit is input at the same time. Furthermore, data-binding authenticity is also ensured with the MAC1 and MAC2 hash values (the hash values are calculated from the message). The Grain-128AEAD stream cipher has taken advantage of symmetric key cryptography during the encryption of an authenticated message with associated data. The decryption procedure is more complicated than the encryption due to additional comparisons to check the integrity of the message. This ensures confidentiality, integrity and authenticity of associated data after decryption.

Table 1. Execution speed and power consumption comparison between Grain algorithm and other lightweight ciphers [8], [19], [20].

Cryptosystem	Number of key bits	Number of block bits	Clock cycle per block	Throughput at 100 MHz (Kbps)	Processor logic	Area (Ges)
Block cipher system						
Pressent	80	64	32	200	0.18	1.570
Hight	128	64	34	188	0.25	3.048
mCrypton	96	64	13	492	0.13	2.681
Stream cipher system						
Trivium	80	1	1	100	0.13	2.599
Grain	80	1	1	100	0.13	1.294

The design is similar to Grain-128a, which is an ISO standard for RFID systems (ISO/IEC 29167-13:2015) [13]. In [14], results of memory-optimized implementations of Grain-128a requiring 84 bytes RAM bytes on ARM Cortex-M3 are presented. In [17], Dibyendu Roy et al. presented a design method for NFRS of the Grain-128AEAD algorithm with the objective of improving the execution speed of Grain-128AEAD while ensuring the security of the stream ciphers. Soon after, Bijoy Das et al. showed that the weakness of the attacked Stream Ciphers was in the Linear-Feedback Shift Register (LFSR) and Nonlinear-Feedback Shift Register (NFSR) blocks, then he developed the Attack on Linear Scan Chains

method for Stream Ciphers [18]. This is completely explicable as associated data was not authenticated in the schema for versions prior to Grain-128AEAD.

Table 1 shows that the GRAIN-128AEAD algorithm has better encryption/decryption processing speed than the Trivium stream cipher and other block ciphers applied in lightweight cryptography [8]. Furthermore, Grain-128AEAD can provide authenticated encryption at the expense of modest resources and power, which makes it suitable for embedded application [8], [10], [12]. Most of the current publications focus on the evaluation of Grain 128-AEAD based on mathematics and the implementation of algorithms on computers, there are little studies towards the design of Grain 128-AEAD algorithm on devices with limited resources such as FPGA [9], [15], [11], [19].

To improve the performance and examine the ability of the Grain-128AEAD algorithm on low-power and resource-constrained hardware for embedded application, the authors implemented the algorithm on STM32F400 series microprocessor, specifically STM32F407IGHx, which is a 32-bit chip family of STMicrochip using ARM CortexTM-M4 technology. It is a series designed for medical, industrial and consumer applications that provide high levels of integration and performance, rich embedded memory and peripherals. For simplicity, the authors used the development board MCBSTM32F400 board for testing. The implementation results will be discussed in section 3.

While the Grain-128AEAD works fast and effectively on STM32F400 processor with input data less than 2kB, the execution speed of the algorithm decreases dramatically with input data larger than 2kB. The reason is that the encryption/decryption process of Grain-128AEAD used up most of hardware resources on STM32 chip while generating the large key stream corresponding to the input data, which slowed down the operation of the processor. To overcome these limitations, a processing method for Grain-128AEAD algorithm on the STM32 chip is proposed as following:

Input data processing for Grain-128AEAD: The input data is divided into blocks of 128 bytes for encryption. This encryption process will be executed sequentially from the first block to the last block. In case the last block has less than 128 bytes, it will be zero-padded.

Generating keystream data: After initialization, instead of generating a keystream corresponding to the length of the whole message, the keystream is generated for each 128-byte block, which is also the case for encryption. Finally, the authentication encryption message with associated data will be generated at the last block of the message.

3. RESULTS AND DISCUSSION

3.1. GRAIN-128AEAD ALGORITHM ON STM32F400 CHIP IMPLEMENTATION

The data will be packed into frame for transfer between the microcontroller and the computer. This frame will start with the "*start"* byte, then the data to be processed ends when the "*end"* byte is encountered. In this study, '*start' byte is fixed at 0x2a while the end byte is fixed at 0x2f.*

When data is transmitted from the computer, if the *"start"* byte is received, the microcontroller will receive the message in a byte-by-byte stream and then call an interrupt for processing, which will encapsulate the received message until it encounters *"end"* byte, and passed on the message to the next processing step.

After receiving the frame, it will proceed to remove the *"start"* byte and *"end*" byte to filter the necessary data. The commands to transmit and receive data from the computer to the STM32F400 of this Grain-128AEAD are shown in Table 2.

The authors have built the Grain-128AEAD algorithm on STM32F400, UART protocol was used to transmit data between the micro-controller and the computer while the results were stored on SD card via SDIO interface, which is shown on Figure 5.

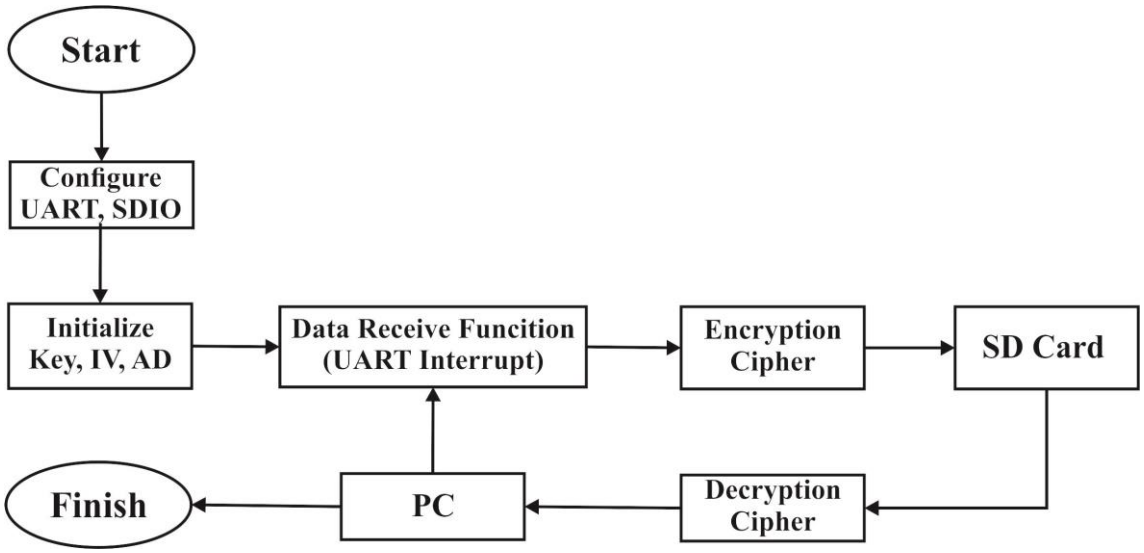

Figure 5. Procedure for implementing Grain-128AEAD algorithm on MCBSTM32F400 Kit using UART interface combined with SDIO interface.

Configure algorithm parameters: The initialization process is performed by 3 commands (set key, set nonce and set up ad data for the Grain-128AEAD). The commands to load parameters for Grain-128AEAD in the STM32F400 chip and the design of these commands are shown in Table 2. Figure 6 shows the process of setting the parameters (as shown in Table 2) and running the data encryption mode of the Grain-128AEAD.

Table 2. Design of command architectures for Grain-128AEAD on STM32F400.

Commands	Functions for Grain-128AEAD
Processing data transferred	
start = 0×2a	Receive the message byte-by-byte and call the interrupt for processing
end = 0×2f	Pack the data and transmit it to the next processing step
Configuring parameters for Grain-128AEAD	
@setkey	Set *key* value for system
@setiv	Set *nonce* value for system
@setad	Set AD data for the system
Run Grain-128AEAD's encryption mode	
**data/*	Encrypt data
Run Grain-128AEAD's decryption mode	
!	Decrypt data stored in SD Card

Encryption mode: The authors designed it with the *"*data/"* command. By this instruction, the Grain-128AEAD algorithm on the STM32F400 chip will encrypt the message transmitted to the chip and save the encrypted result as a file on the SD Card.

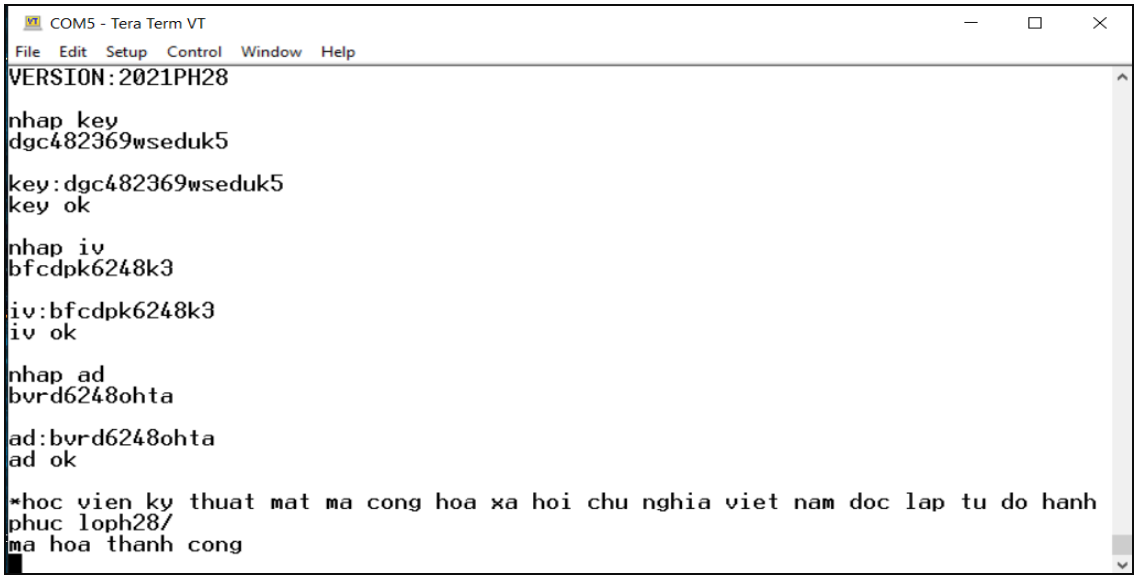

Figure 6. Setting parameters and running encryption of Grain-128AEAD on KIT STM32F400.

To test the encryption function of Grain-128AEAD on the microcontroller, the authors chose the ciphertext "*hoc vien ky thuat mat ma cong hoa xa hoi chu nghia viet nam doc lap tu do hanh phuc loph28"* to perform encryption by command *"*data/"*. The generated ciphertext will be saved in the SD Card as shown in Figure 7.

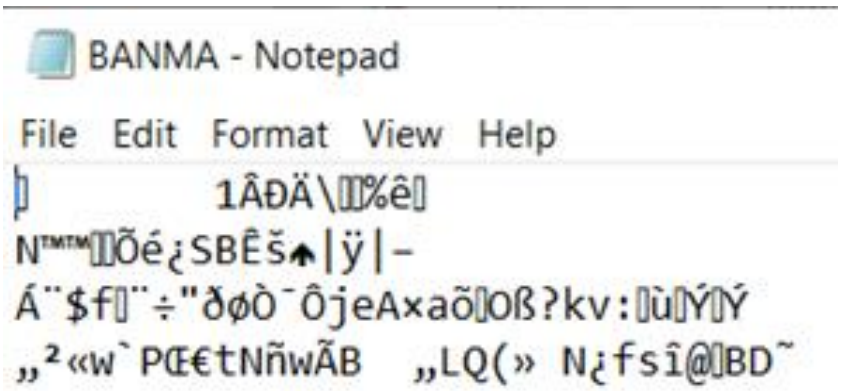

Figure 7. Detailed results of ciphertext when executed by Grain-128AEAD stored in SD card

Decryption mode: The command "!" is set for decryption. If the microcontroller receives the command from the computer, data retrieved from the SD will be decrypted. The decryption results are shown in Figure 8, which indicated successful operation of the Grain-128AEAD algorithm on STM32F400 series microcontroller. Different string and data were used for algorithm evaluation, presented in section B.

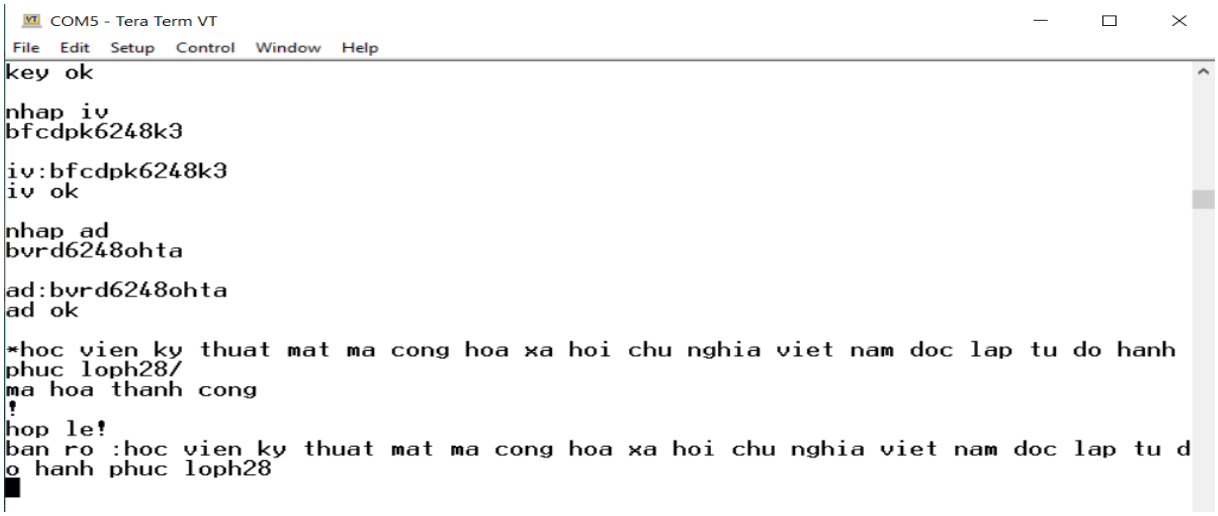

Figure 8. Plaintext file of ciphertext when decrypting Grain-128AEAD on STM32F400 chip.

3.2. RESULTS ACHIEVED BY GRAIN-128AEAD EVALUATION

The algorithm was designed to run on a Dell Inspiron 7559 computer with core i7, 8Gb ram running WIN 10-64 bit. The Grain-128AEAD algorithm is implemented in C programming language with Visual Code Studio 2015.

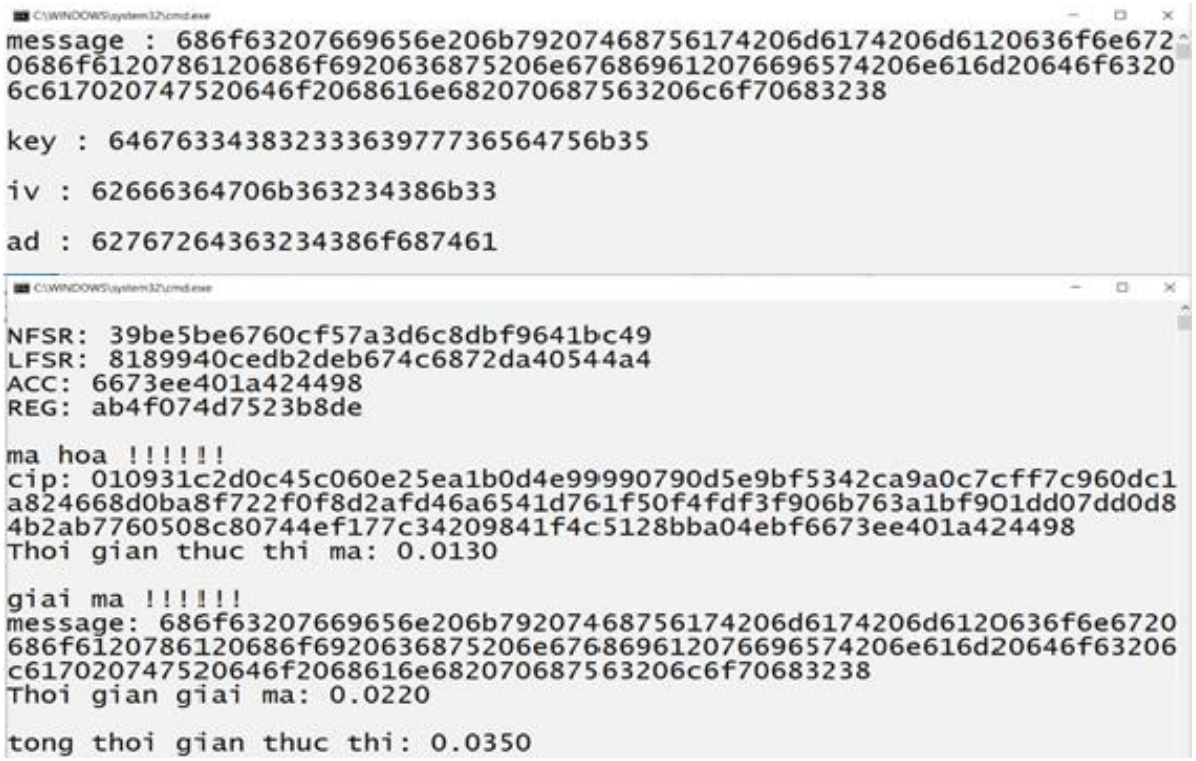

Figure 9. Results of running Grain -128AEAD algorithm on computer.

To check the execution time for the Grain-128AEAD algorithm on the STM32F400, the author used the Oscilloscope GDS-2304 available at the Academy of Cryptography Techniques. The measured execution time of Grain-128AEAD on the STM32F407 chip and on the computer are shown on Table 3 and Figure 9, respectively.

Table 3. Comparison of run-time results of Grain-128AEAD on Kit STM32F400 and on Computer.

Input data (bytes)	Computer		STM32F400	
	Encryption(ms)	Decryption(ms)	Encryption (μs)	Decryption (μs)
90	13	22	12.89	22.11
128	10	20	10.77	19.23
500	60	90	59.57	91.43
1000	90	210	91.21	211.79
1500	120	300	119.67	302.33
2600	220	480	221.15	482.85

In Table 3, the experimental results with different input data for Grain-128AEAD algorithm with various input data size are shown. With the operating frequency of the STM32F400 processor is 48MHz, the execution speed of the algorithm on the microprocessor is approximately 1000 times faster than that on the setup computer.

In particular, the obtained code size of Grain-128AEAD algorithm implementation on STM32F400 is 832 bytes. The number of cycles for encryption and decryption is 517 and 923, respectively, which makes total time for both encryption and decryption of Grain-128AEAD on the STM32F400 1440 (cycles) (Table 4). Furthermore, the resulting compiled (.hex) file for generated Grain-128AEAD program is only 54kB. It implies that the application can be easily loaded into smaller and lower cost microcontrollers, or integrated with other programs.

Table 4. Comparison of run-time results of Grain-128AEAD on Kit STM32F400 and other hardware platforms.

Crypto system	Version	Code size (byte)	Total time (cycles)	Ref.
ARM balanced choice (MCBSTM32F400 Kit)				
Grain	Grain-128AEAD	832	1440	**This work**
AVR balanced choice (AVR ATmega 128 Kit)				
ACORN	v1	3024	396798	[6], [15], [20], [21], [22]
AES-GCM	v2	2338	1460677	
ASCON	v2	24590	53272	
(Alt.) ASCON	v3	3724	362589	
Grain	v2	1734	263101	
Ketje-Jr	v2	5156	311949	
NORX	v3	5028	124062	
MSP balanced choice (MSP430F1611 Kit)				
ACORN	v2	1750	676228	[6], [15], [20], [21], [22]
AES-GCM	v2	1952	2174330	
ASCON	v3	5572	417711	
Grain	v4	1358	184104	
Ketje-Jr	v2	6248	335624	
NORX	v3	4216	71419	

Table 4 presents resulting code size of various crypto algorithm implementations on different platforms. On both AVR ATmega 128 (8-bit) MSP430F1611 (16-bit), Grain implementation generated the smallest code size of 1734 bytes and 1358 bytes, respectively. Meanwhile, this study showed that Grain-128AEAD implementation on STM32F400 only produced the code size of 832 bytes. It's also shown that the total execution time of Grain-

128AEAD is 1440 (cycles), which is much lower than the results obtained using various crypto system proposed in publications [6], [15], [20], [21], [22]. In [15], hardware implementations targeting ASICs were presented and discussed. The code was implemented in VHDL, and synthesized using a 65 nm library from ST Microelectronics. With both high speed and low power implementations developed, Grain-128AEAD is proved to be suitable for IoT devices and lightweight cryptographic devices. The results in Table 4 indicated that the algorithm implemented in this study is better than other implementations in terms of execution time and code size. The 128-byte block processing immensely decreased computational complexity for the algorithm, thus lead to small code size and execution time while still preserving data confidentiality and authentication.

4. CONCLUSION

The study implements Grain-128AEAD algorithm on ARM's STM32F400 series microcontroller, particularly data processing and data transmission interfaces. The authenticated encryption and decryption results have been measured with the associated data of Grain-128AEAD. The implemented program of Grain-128AEAD on STM32F400 have a small code size of 832 bytes, and the compiled (.hex) file is of 54kB. The total execution time for a 128-byte input block of Grain-128AEAD algorithm (Encryption and Decryption) on STM32F400 takes 30 μs, which is equivalent to 1440 (cycles). With the results measured, it can be inferred that the encryption/decryption execution speed of Grain-128AEAD on the STM32F400 chip is fast, consuming little power, once again proving the previous assertion that Grain-128AEAD is suitable for embedded devices, IoT devices and lightweight cryptographic devices.

ACKNOWLEDGMENT

The authors thank Academy of Cryptography Techniques for supporting this work.

REFERENCES

[1]. M. Agren, M. Hell, T. Johansson, W. Meier, Grain-128a: a new version of Grain-128 with optional authentication, Int. J. Wirel. Mob. Comput., 5 (2011) 48. https://doi.org 10.1504/IJWMC.2011.044106

[2]. Y. Todo, T. Isobe, W. Meier, K. Aoki, B. Zhang, Fast Correlation Attack Revisited, Annual International Cryptology Conference, (2018) 129–159. https://doi.org/10.1007/978-3-319-96881-0_5

[3]. M. Hell, T. Johansson, W. Meier, J. Sonnerup, H. Yoshida, An AEAD Variant of the Grain Stream Cipher, International Conference on Codes, Cryptology, and Information Security, (2019) 55–71. https://doi.org/10.1007/978-3-030-16458-4_5

[4]. V. A. Thakor, M. A. Razzaque, M. R. A. Khandaker, Lightweight Cryptography Algorithms for Resource-Constrained IoT Devices: A Review, Comparison and Research Opportunities, IEEE Access, 9 (2021) 28177–28193. https://doi.org/10.1109/ACCESS.2021.3052867

[5]. S. S. Dhanda, B. Singh, P. Jindal, Lightweight Cryptography: A Solution to Secure IoT, Wirel. Pers. Commun., 112 (2020) 1947–1980. https://doi.org/10.1007/s11277-020-07134-3

[6]. A. Maximov, M. Hell, Software Evaluation of Grain-128AEAD for Embedded Platforms, 2020, https://eprint.iacr.org/2020/659.pdf

[7]. M. Sonmez Turan et al., Status Report on the Second Round of the NIST Lightweight Cryptography Standardization Process, Gaithersburg, MD, Jul. 2021. https://doi.org/10.6028/NIST.IR.8369

[8]. NIST, Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process, pp. 1–17, 2018. https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf.
[9]. S. S. Mansouri, E. Dubrova, An Improved Hardware Implementation of the Grain Stream Cipher, in 2010 13th Euromicro Conference on Digital System Design: Architectures, Methods and Tools, (2010) 433–440. https://doi.org/10.1109/DSD.2010.49
[10].M. Robshaw, The eSTREAM Project, (2008) 1–6. https://doi.org/10.1007/978-3-540-68351-3_1.
[11].I. Salam, T. H. Ooi, L. Xue, W.-C. Yau, J. Pieprzyk, R. C.-W. Phan, Random Differential Fault Attacks on the Lightweight Authenticated Encryption Stream Cipher Grain-128AEAD, IEEE Access, 9 (2021) 72568–72586. https://doi.org/10.1109/ACCESS.2021.3078845
[12].M. U. Bokhar, S. Alam, S. H. Hasan, A Detailed Analysis of Grain family of Stream Ciphers, Int. J. Comput. Netw. Inf. Secur., 6 (2014) 34–40. https://doi.org/10.5815/ijcnis.2014.06.05
[13].D. Matrix, R. Etendu, International standard ISO / IEC Information technology — Automatic identification and data capture, (2011) 2011.
[14].Y. Watanabe, H. Yamamoto, H. Yoshida, Towards Minimizing RAM Requirement for Implementation of Grain-128a on ARM Cortex-M3, IEICE Trans. Fundam. Electron. Commun. Comput. Sci., E103.A (2020) 2–10. https://doi.org/10.1587/transfun.2019CIP0025
[15]. J. Sönnerup, M. Hell, M. Sönnerup, R. Khattar, Efficient Hardware Implementations of Grain-128AEAD, (2019) 495–513. https://doi.org/10.1007/978-3-030-35423-7_25
[16]. J. C. Hernandez, Software implementation of authenticated encryption algorithms on ARM processors, 2018.
[17].D. Roy, D. K. Dalai, An Observation of Non-randomness in NFSR-Based Stream Ciphers with Reduced Initialization Round, J. Hardw. Syst. Secur., 5 (2021) 89–102. https://doi.org/10.1007/s41635-021-00113-5
[18].B. Das, A. Sardar, S. Maiti, A. Das, D. R. Chowdhury, An Attack on Linear Scan Chains for Stream Ciphers and the Impossibility of Simple Countermeasures, J. Hardw. Syst. Secur., 5 (2021) 191–207. https://doi.org/10.1007/s41635-021-00118-0
[19]. B. Li, M. Liu, D. Lin, FPGA implementations of Grain v1, Mickey 2.0, Trivium, Lizard and Plantlet, Microprocess. Microsyst., 78 (2020) 103210. https://doi.org/10.1016/j.micpro.2020.103210
[20].P. Kitsos, N. Sklavos, G. Provelengios, A. N. Skodras, FPGA-based performance analysis of stream ciphers ZUC, Snow3g, Grain V1, Mickey V2, Trivium and E0, Microprocess. Microsyst., 37 (2013) 235–245. https://doi.org/10.1016/j.micpro.2012.09.007
[21].M. Hell, T. Johansson, A. Maximov, W. Meier, S. Jonathan, and E. Ab, Grain-128AEAD - Status Document, pp. 1–5, 2020.
[22].M. Hell, T. Johansson, W. Meier, S. Jonathan, M. Hell, Grain-128AEAD - A lightweight AEAD stream cipher Cover sheet Backup point of contact :, pp. 1–37, 2019. https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/grain-128aead-spec-final.pdf.