Title	Interceptive side channel attack on AES-128 wireless communications for IoT applications(Main Article)
Author(s)	Pammu, Ali Akbar; Chong, Kwen-Siong; Ho, Weng-Geng; Gwee, Bah Hwee
Citation	Pammu, A. A., Chong, K.-S., Ho, W.-G., & Gwee, B. H. (2016). Interceptive side channel attack on AES-128 wireless communications for IoT applications. 2016 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), 650-653.
Date	2016
URL	http://hdl.handle.net/10220/42163
Rights	© 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at: [https://doi.org/10.1109/APCCAS.2016.7804081].

Interceptive Side Channel Attack on AES-128 Wireless Communications for IoT Applications

Ali Akbar Pammu*, Kwen-Siong Chong, Weng-Geng Ho and Bah-Hwee Gwee
School of Electrical and Electronic Engineering
Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798
Email: *ali1@e.ntu.edu.sg

***Abstract*–We propose wireless interceptive side-channel attack (SCA) technique to reveal the 16-byte secret key of the AES-128 encryption algorithm in wireless communications through Correlation Electromagnetic Analysis (CEMA) for Internet of Things (IoT) applications. The encrypted wireless communication link is established using two ATmega-processor based Arduino boards. There are two key features in our proposed interceptive SCA technique. First, we identify the sensitive modules, which emit significant EM signal (physical leakage information) of the ATmega processor during the encryption process. The significant EM signals are highly correlated with processed data to reveal the secrete key. Second, we investigate the resistance of AES-128 encryption algorithm implementation on ATmega processor against CEMA based SCA. The wireless signal is intercepted and correlated with EM signals generated during the encryption process. Based on our experimental results, the correlated EM signals leak out at the three modules - FLASH memory, data bus and SRAM modules during the encryption process are 101.56 dBμV, 105.34 dBμV and 121.79 dBμV respectively. In addition, we perform the CEMA attacks on the AES-128 implementation on the ATmega processor and the secret key is successfully revealed at 20,000 EM traces.**

***Keywords*–Electromagnetic Attack, Arduino, ATmega, SCA**

I. Introduction

Advanced technology, nowadays, enables the integration of wireless communication devices with multiple sensors which is well-known as Internet-of-Things (IoT). However, the wireless communication platform used in IoT such as Wi-Fi, Bluetooth and Radio Frequency (RF) are easily intercepted by unauthorized party (i.e. adversary) as depicted in Fig. 1. The intercepted communication can be abused for another purpose and eventually make the IoT applications such as E-Healthcare system [1], Micro Energy Harvesting [2], Smart Building [3] and Vehicle-to-Vehicle (V2V) communication [4] unsecured and vulnerable. In addition, the adversary is not only able to intercept the communication signals but also able to manipulate it. Therefore, the wireless communication in IoT may no longer be trustfully due to malicious attack.

In order to address the aforementioned issues, the wireless communication system in IoT must be capable to communicate with other devices over an air interface and provide privacy and security capabilities. In this context, it is equipped with security module, which can protect the necessary information sent to and received from an intended party. The security module is embedded with encryption algorithm such as Advanced Encryption Standard (AES) which is typically transform the message (i.e. plaintext) into encrypted information (i.e. ciphertext) using a secret key (known only by intended party). Therefore, any unauthorized party that intercept the wireless signals is almost impossible to interpret it without the secret key of the cryptographic algorithms (i.e. AES).

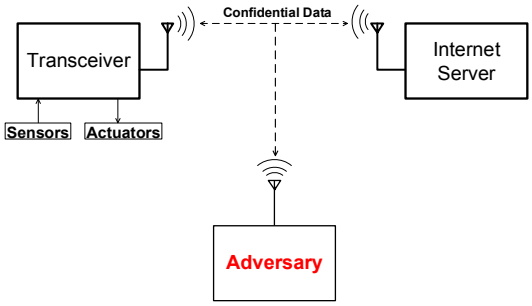


Fig. 1: Interceptive of IoT system by Adversary

The AES algorithm has been proven to be more effective compared with other symmetric algorithm such as Data Encryption Standard (DES) in protecting the plaintext [5]. In 1998, the key of DES (2^{56} possible keys) has been successfully broken by brute force attack. In addition, the AES algorithm can process large block size of plaintext such as 128-bit, 192-bit and 256-bit with dissipate low power (i.e. in FPGA, dissipate about 10.5mW) and fast process (i.e. processing 128-bit requires only 50μs) [5]. Therefore, the AES has high potential to be implemented in the future IoT system, particularly at sensor based application with powered by battery at high-end user systems.

A physical-attack approach which is known as side-channel attack (SCA) [6] has been reported to successfully extract the secret key of the encryption algorithm by analyzing the physical parameters and the processed data. The physical parameters such as power dissipation [7], electromagnetic (EM) interference [8] and timing [9] are generated by electronic devices during the encryption process. Power analysis is the most common SCA where it analyzes the power dissipation profile to extract the secret key. However, power analysis is more invasive compared with EM analysis [10]. This is due the adversary is required to modify the circuit in order to obtain a valid power measurement. In this context, the adversary should identify the V_{DD} and put the resistor between the V_{DD} and main supply of encrypted devices (i.e. crypto-chip). The current across the resistor is measured during the encryption process. Correlation Electromagnetic Analysis (CEMA) is one of the popular techniques in EM based attack, which is implemented to reveal the secret key of the AES implementation.

The AES-128 algorithm, which processes 128-bit key size, requires 10 rounds of iterations to encrypt the plaintext into ciphertext. The 10 rounds of iterations make AES-128 highly effective when compared to other encryption algorithms such as DES and triple-DES (3-DES). However, due to the leaking of physical parameters correlate with intermediate data, the AES-128 implementation may still be vulnerable against SCA, particularly against the CEMA attack, as depicted in Fig. 2.

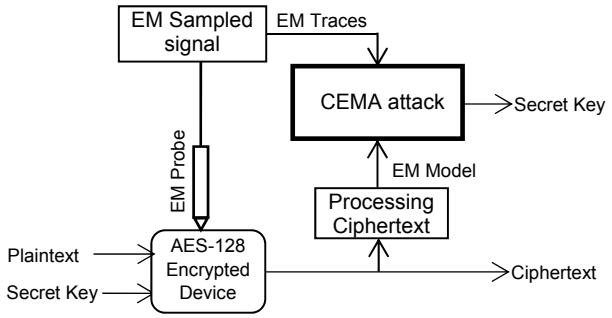


Fig. 2: General scheme of CEMA based SCA on AES-128

The SCA on small embedded devices, such as microcontrollers and cryptographic co-processors, are popular nowadays due to its ubiquitous application at high-end devices of IoT. However, there are only few literatures and studies of SCA on these systems (i.e. microcontroller and programmable chip) due to complexity on the program implementation. In the near future, there will be high chance to shift towards moving the cryptographic operation to the microcontroller, as implemented in IoT, mobile payments and host card emulation [10].

In this paper, we propose a wireless interceptive SCA technique to reveal the secret key of the AES-128 encryption algorithm. There are two key features of our proposed wireless interceptive SCA technique. First, we analyze the physical leakage information, the correlated EM signals, of the processed data generated during the encryption process based on the circuit architecture of the ATmega processor. Therefore, our analysis is able to identify the particular module of the processor leaks the correlated EM signals. Second, we investigate the resistance of AES-128 encryption algorithm in ATmega processor implementation against CEMA based SCA. Thus, our investigation is able to evaluate the security level of AES implementation in ATmega processor against CEMA attack. Based on the experimental result, the correlated EM signals, corresponded with the processed data, noticeably leak out at FLASH memory, data bus and SRAM module during the encryption process. In addition, based on CEMA attack, the secret key is successfully revealed by 20,000 EM measurements.

This paper is organized as follows. Section II presents the ATmega structure in Arduino for high-end user application. Section III presents the proposed investigation of CEMA attack on Arduino ATmega based on AES-128 implementation. Section IV presents the measurement results and finally, conclusions are drawn in Section V.

II. ATMEGA ARCHITECTURE

The basic circuits design architecture of the processor can determine the robustness and resistance against SCA. Cell logic, switching activities, placement and routing of the modules and wiring structure are the key parameters which can potentially leak information out during the encryption process. In the cell logic perspective, the binary input (0 or 1) can be identified from variant of power dissipation measurement, hence the attacker can easily identify the internal processed data of the processor. For the switching activities, placement and wiring, its EM signal can be identified and measured during the transferred binary value (0 or 1) in the wire. The EM signal generated by the processor during the encryption is relatively easy to measure by placing the EM probe close to the chip.

In this section, feature of Arduino based on ATmega processor (specifically on ATmega328P) is discussed to investigate the possibility of leaking the SCA information. The processor technology is based on 8-bit AVR in which embed on-chip flash memory for program storage. As tabulated in Table I, size of the FLASH memory is 32KB used to store the program such as AES which occupies only 4KB. Temporary state of 16-byte of each rounds AES (totally 10 rounds iteration) is stored in data SRAM. The changes of the value for each state transitions in data SRAM is possible to be corresponded with the Hamming Distance (HD), EM model, which is derived from ciphertext. The switching activities with frequency of 20MHz generates EM signals which depends on the value of the processed information.

TABLE I. SPECIFICATION OF ATMEGA PROCESSOR BASED ARDUINO

ATmega328P	
Processor Technology	8-bit AVR
Operating Voltage	1.8 – 5.5 V
Switching Frequency	20MHz
Power Dissipation*	0.36mW
FLASH Memory	32Kbyte
EEPROM	1Kbyte
SRAM	2Kbyte

*at voltage 1.8V and Frequency 1MHz

In order to protect the sensitive information, the 8-bit AVR microcontroller is implemented for high-performance encryption and decryption engine which can support 128-bit, 192-bit and 256-bit as lengths size key for AES. All AVR microcontrollers contain lock mechanisms to prevent reading and copying the program stored in on-chip Flash memory [11]. The lock mechanism program is stored in EEPROM. The intermediate result of AES, which is the output of each round is stored in data SRAM before sending to I/O lines and I/O modules as depicted in Fig. 3.

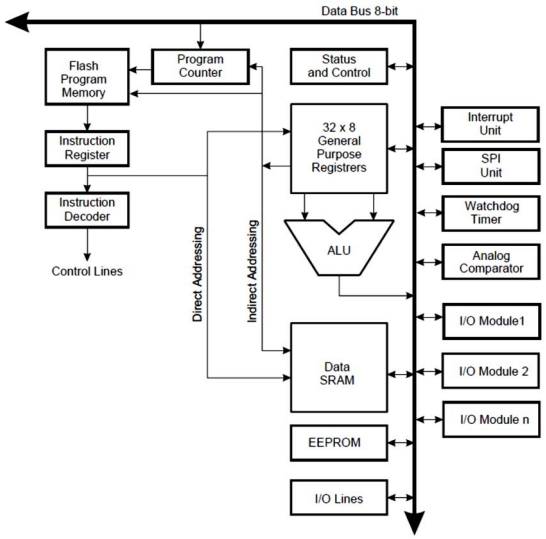


Fig. 3: Block diagram of 8-bit AVR architecture ATmega328P

The instruction from Flash memory during the encryption is sent to 32×8 General Purpose Register (GPR) and Arithmetic Logic Unit (ALU) to process the temporary stored information (i.e. plaintext) in the data SRAM. The result of the encryption is sent back to data SRAM by overwriting the previous result. The interface from GPU and ALU to SRAM is using data bus 8-bit as well as to the I/O modules. The EM of data flowing in data bus is emitted can be corresponded with processed data. In this context, the processed data is derived from ciphertext by employing HD EM model. The

HD EM model is correlated with EM measurement and analyze the correct key. Since the ciphertext is more accessible than plaintext in practice, the HD model in this context is the changes between input and output of the last round. The key obtained at this round is reversed back using reverse expansion key AES to get the original key.

III. CEMA ON AES-128 BASED ARDUINO IMPLEMENTATION

The CEMA is the most prevalent type of EM analysis attack against encrypted devices [8] based on the EM model. An attacker exploits the correlation between the EM emanation by the device and the intermediate data (EM model) generated during the encryption process. In the AES-128 encryption process, there are 10 rounds of operations and at the last round consists of three operations such as S-Box, ShiftRow and AddRoundKey (XOR). As depicted in Fig. 4, the HD of the input (output of round 9) and the output (ciphertext) of the last round (round 10) is obtained to generate the HD EM model. The correlation coefficient based on the correlation between the EM model and the EM traces can then be determined to reveal the secrete key.

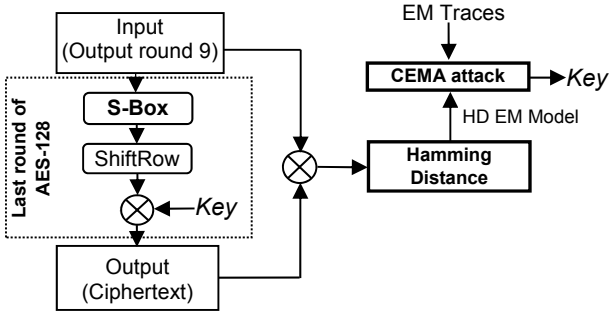


Fig. 4: CEMA attack is based on HD EM model of the last round AES

The CEMA attack is performed based on the analysis of correlation coefficient. The highest value of correlation coefficient occurs at the highest variance (σ^2) of EM traces [6]. The variance of EM traces is defined as the distribution of the EM emanation generated from different values of the processed intermediate data. Highest variance corresponds to sampled EM traces (sampling points) which have highest probability of leaking the information of the secret key.

The AES-128 algorithm is implemented on Arduino microcontroller which are the random plaintext and key pre-programmed in the flash memory. In this context, the AES-128 is based on *T-Table* approach [10], where the internal operations (i.e. S-Box, MixColumn and ShiftRow) are merged in one Look-Up-Table (LUT), hence the implementation dissipates low power and compatible with high-end user (i.e. IoT application). There are two Arduinos employed in this experiment, Arduino_1 to encrypt the plaintext and send it in the form of ciphertext and Arduino_2 intercepts the ciphertext in wireless communication.

Our main objective is to reveal the stored and programmed key on unprotected Arduino_1 by analyzing the ciphertext and the EM signals. In Arduino_1, it requires two sub-modules, AES-128 and RF with antenna A_1, which are processing the plaintext and the key into a form of ciphertext and send it through wireless channel. In Arduino_2, the ciphertext is received by RF antenna A_2 and convert the analog signal into digital by means of Analog to digital converter (ADC) module. The EM signal is measured together with ciphertext during the encryption process and analyzed both parameters in Personal Computer (PC) as depicted in Fig. 5.

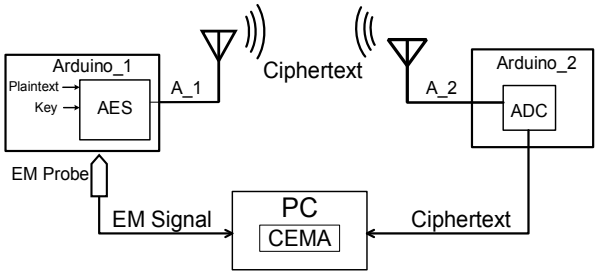


Fig. 5: CEMA attack scenario on Arduino

The CEMA attack is a byte-based EM analysis attack. Each byte of key (sub-key) is estimated by means of 256 possible values (1 byte = 8 bits and possibility is $2^8 = 256$), hence the correct sub-key is one of the 256 sub-key candidates. The CEMA attack is performed by analyzing the correlation coefficient ($r_{i,j,t}$) of two variables, EM model ($X_{i,j,m}$) and EM traces ($Y_{t,m}$), for $i = 1, \cdots, 16$ sub-keys, $j = 1, \cdots, 256$ sub-key candidates, $t = 1, \cdots, 1000$ sampling points, as follows:

$$r_{i,j,t} = \frac{\sum_{m=1}^{n}(X_{i,j,m}-\bar{X}_{i,j})(Y_{t,m}-\bar{Y}_t)}{\sqrt{\sum_{m=1}^{n}(X_{i,j,m}-\bar{X}_{i,j})^2}\cdot\sqrt{\sum_{m=1}^{n}(Y_{t,m}-\bar{Y}_t)^2}} \quad (1)$$

The correct sub-key, i, corresponds to the highest $r_{i,j,t}$ at particular sub-key candidate, j, and sampling point of power traces, t. For instance, in attacking the first sub-key (i =1), with 1,000 EM traces (n = 1,000), the highest correlation coefficient ($r_{i,j,t}$ = 0.9) occurs at sampling point 61 (t = 61) for sub-key candidate 45 (j = 45), thus the correct first sub-key is 45.

IV. MEASUREMENT RESULTS

The experiment is conducted based on two Arduino microcontrollers, which are implementing AES-128 algorithm. The EM emanation signal of Arduino_1 generated during the encryption process is measured and recorded in oscilloscope (2.5GS/sec) as depicted in Fig. 6. The encrypted data, ciphertext, intercepted by Arduino_2, which is sent from Arduino_1 through wireless communication (i.e. RF).

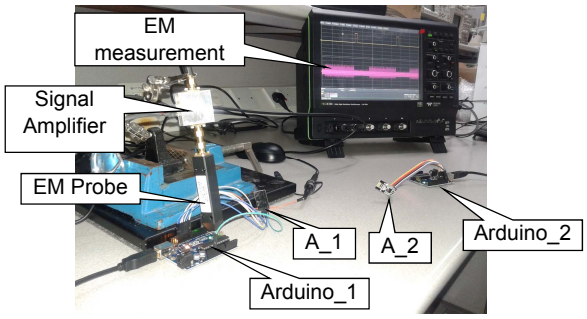


Fig. 6: EM measurement of AES-128 based Arduino implementation

The encryption process is initiated by loading the plaintext from I/O module to SRAM through data bus 8-bit. Subsequently, the data in the SRAM is processed in 10 round iterations with the instruction from FLASH memory to GPR and ALU through data bus 8-bit. During the encryption process, the EM probe is used to detect the EM signal. The EM probe is carefully place on particular location of the module and measure the corresponding EM signals. Although the EM is detected on whole surface of the

processer as depicted in Fig. 7, however the maximum EM is generated at particular location, which are FLASH memory, SRAM and data bus 8-bit. This is due to the temporary output which is stored in SRAM from flash memory through data bus 8-bit generate EM during the encryption process. The measurement result EM generated during the encryption process is tabulated in Table II. The maximum and minimum of EM for each module is clearly identified during the encryption. This implies that each module potentially leaks information (i.e. secrete key) through EM signal while performing the encryption.

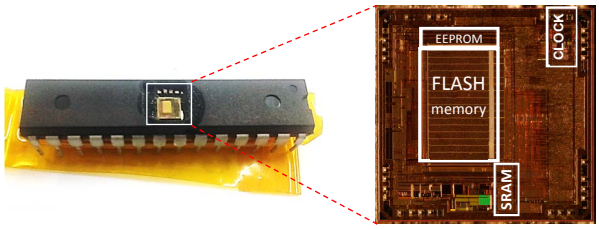
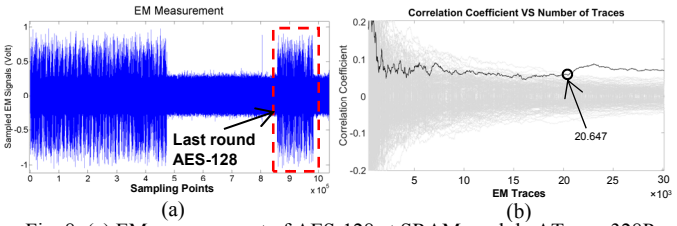


Fig. 7: Layout view of the ATmega328P

TABLE II. EM MEASUREMENT RESULTS OF MODULES IN ATMEGA328P DURING THE ENCRYPTION PROCESS

Module	EM measurement		
	Max (dBµV)	Min (dBµV)	Difference
32×8 GPR	96.74	95.41	1.33
ALU	100.23	96.51	3.72
I/O Modules	85.21	84.27	0.94
FLASH memory	101.56	96.33	5.23
Instruction Register	98.63	97.67	0.96
EEPROM	75.44	70.14	5.3
SRAM	**121.79**	**85.32**	**36.47**
8-bit Data Bus	105.34	95.33	10.01

The 30,000 random plaintext are processed during the encryption process and the corresponding EM signals is measured. The SRAM module generate high EM signals and difference between maximum and maximum EM measurement is highest. This implies that the EM signals is highly corresponded with the processed data and vulnerable against CEMA attacks. In this experiment, the EM measurement is focused on SRAM module during the encryption process. The EM measurement is depicted in Fig. 8(a). The HD EM model is employed in this experiment, measuring the changes of input and output of the last round in SRAM of the AES-128 implementation. The CEMA result shows that the secret key is start to reveal at 20,647 EM traces as depicted in Fig. 8(b), where the black and grey color are denoted as correct and the wrong key respectively.



(a) (b)

Fig. 8: (a) EM measurement of AES-128 at SRAM module ATmega328P, (b) Correlation coefficient vs EM Traces of the secrete key AES-128 key based Arduino implementation

V. CONCLUSION

We have proposed a wireless interceptive SCA technique to reveal the secret key of the AES-128 encryption algorithm in wireless communications through CEMA for IoT applications. The physical leakage information, the correlated EM signals, of the processed data generated during the encryption process based on the circuit architecture of the ATmega processor has been analyzed. Therefore, our analysis is able to identify the particular module of the processor leaks the correlated EM signals. We have also investigated the resistance of AES-128 encryption algorithm implemented on ATmega processor against CEMA based SCA. Based on the experimental result, the correlated EM signals, corresponded with the processed data, noticeably leak out at data bus processor and SRAM module during the encryption process. In addition, we have performed the CEMA attack against AES-128 algorithm based on ATmega processor implementation and the secret key is successfully revealed by 20,000 EM measurements.

ACKNOWLEDGMENT

This research work was supported by Agency for Science, Technology and Research, Singapore, under SERC 2013 Public Sector Research Funding, Grant No: SERC1321202098. The authors thank A*STAR for the kind support in funding this research. We also thank Mr. Chua Chung Tah for his help to delayer the ATmega328P, which enable us to analyze the internal architecture.

REFERENCES

[1] Boyi Xu, Li Da Xu, Hongming Cai, Cheng Xie, Jingyuan Hu and Fenglin Bu, "Ubiquitous Data Accessing Method in IoT-Based Information System for Emergency Medical Services," in *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1578-1586, May 2014.

[2] R. Gomez Cid-Fuentes, A. Cabellos-Aparicio and E. Alarcon, "Area Model and Dimensioning Guidelines of Multisource Energy Harvesting for Nano–Micro Interface," in *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 18-26, Feb. 2016.

[3] F. Zafari, I. Papapanagiotou and K. Christidis, "Microlocation for Internet-of-Things-Equipped Smart Buildings," in *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 96-112, Feb. 2016.

[4] L. Du and H. Dao, "Information Dissemination Delay in Vehicle-to-Vehicle Communication Networks in a Traffic Stream," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 66-80, Feb. 2015.

[5] Ali Akbar Pammu, Kwen-Siong Chong, Kyaw Zwa Lwin Ne and Bah-Hwee Gwee, "High Secured Low Power Multiplexer-LUT Based AES S-Box Implementation," *IEEE International Conference on Information System Engineering (ICISE) 2016*, Los Angeles, Apr 2016, pp. 3-7

[6] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Advances in Cryptology — CRYPTO' 99*. vol. 1666, M. Wiener, Ed., ed: Springer Berlin Heidelberg, 1999, pp. 388-397.

[7] Kwen-Siong Chong *et al*., "Counteracting differential power analysis: Hiding encrypted data from circuit cells," *IEEE International Conference on Electron Devices and Solid-State Circuits (EDSSC) 2015*, Singapore, 2015, pp. 297-300.

[8] Y. Hori, T. Katashita, A. Sasaki and A. Satoh, "SASEBO-GIII: A hardware security evaluation board equipped with a 28-nm FPGA," *The 1st IEEE Global Conference on Consumer Electronics 2012*, Tokyo, 2012, pp. 657-660.

[9] B. Coppens, I. Verbauwhede, K. D. Bosschere and B. D. Sutter, "Practical Mitigations for Timing-Based Side-Channel Attacks on Modern x86 Processors," *30th IEEE Symposium on Security and Privacy*, Berkeley, CA, 2009, pp. 45-60.

[10] J. Balasch, B. Gierlichs, O. Reparaz, and I. Verbauwhede, "DPA, Bitslicing and Masking at 1 GHz," In *Cryptographic Hardware and Embedded Systems - CHES 2015*, Lecture Notes in Computer Science 9293, T. GÃneysu, and H. Handschuh (eds.), Springer-Verlag, pp. 599-619, 2015.

[11] Amel-8271J-AVR-ATmega-Datasheet_11/2015: ATmega48A/ PA/ 88A/ PA/ 168A/ PA/ 328/P