A Simple Power Analysis of an FPGA implementation of a polynomial multiplier for the NTRU cryptosystem

Eros Camacho-Ruiz*[1], Santiago Sánchez-Solano[1], Macarena C. Martínez-Rodríguez[1],
Erica Tena-Sanchez[1,2], Piedad Brox[1]
[1]*Instituto de Microelectrónica de Sevilla (CSIC / University of Seville)*
[2]*Department of Electronics Technology, Escuela Politécnica Superior, University of Seville*
Seville, Spain
*camacho@imse-cnm.csic.es

***Abstract*—As quantum computing technology advances, the security of traditional cryptographic systems is becoming increasingly vulnerable. To address this issue, Post-Quantum Cryptography (PQC) has emerged as a promising solution that can withstand the brute force of quantum computers. However, PQC is not immune to attacks that exploit weaknesses in implementation, such as Side Channel Attacks (SCAs). SCAs can extract secret keys by analyzing the physical characteristics such as power consumption of the device while performing cryptographic operation. Simple Power Analysis (SPA) is a type of SCA that uses power consumption measurements to extract sensitive information. By applying SPA to a specific hardware implementation of a PQC algorithm such as the NTRU, potential vulnerabilities can appear in the Arithmetic Unit (AU) in charge of the multiplication operation. The effectiveness of this analysis to extract sensitive information has been evaluated through extensive experiments in which different countermeasures and strategies have been proposed, as well as an accelerated algorithm has been implemented. The results demonstrate that SPA can point out security breaches in the NTRU implementation, indicating an issue that can affect the PQC in the future.**

***Index Terms*—Post-Quantum Cryptography, NTRU, Simple Power Analysis**

I. Introduction

The technological advances are approaching quantum computers with enough computing power to threaten current cryptography. As was demonstrated in Shor's algorithm, current public-key cryptosystems based on the hardness of integer factorization and discrete logarithms will become vulnerable to attacks using a sufficiently large quantum computer. Mathematically, Post-Quantum Cryptography (PQC) has emerged to face it, but unfortunately, its integration on classical computers is not always enough to achieve a full-protection.

This research was supported in part by the SPIRS Project with Grant Agreement No. 952622 under the EU H2020 research and innovation programme, QUBIP Project with Grant Agreement No. 101119746 under the EU Horizon Europe research and innovation programme, SCAROT project 1380823-US/JUNTA/FEDER, and the ARES Project PID2020-116664RB-100 funded by MCIN/AEI/10.13039/501100011033 and the EU NextGeneration EU/PRTR. M.C.M.R. holds a postdoc fellowship from the Andalusia Government with support from PO FSE of EU. E.C.R. is supported by the FPU20/03008 predoc grant from the Spanish government.

The transition towards a quantum-safe era is not a straightforward way and several actions are identified that appoint to different strategies. At the industrial level, standardization is a key factor to introduce PQC. The standards for PQC are fostered by NIST which launched a public competition to request, evaluate, and standardize one or more quantum-resistant public-key and signature algorithms [1]. Other activities are oriented to develop efficient implementations of PQC on embedded systems compatible with tight constraints on power consumption and memory size, as well as limitations on time and processing power. Additionally, one crucial aspect is to guarantee that no sensitive information is leaked in the implementation of a post-quantum algorithm. Additionally, several post-quantum constructions are particularly vulnerable to Side-Channel Attacks (SCAs) that exploit specifically chosen ciphertexts to amplify the observed leakage, known as Chosen-Ciphertext Side-Channel Analysis (CC-SCA), [2]. The security evaluation of post-quantum schemes against attacks and the development of effective countermeasures to mitigate them are open research topics [3].

In passive attacks, sensitive information is inferred by observing physical characteristics of the device such as execution time, power consumption, or electromagnetic emanation during the normal operation mode [4]. In the case of power SCAs, two categories are distinguished Simple Power Analysis (SPA) and Differential Power Analysis (DPA) attacks. SPA is based on visual inspection of one or few power consumption measurements collected over a period of time. DPA is a method for analyzing large number of power consumption traces using statistical analysis and error correction techniques. This approach takes multiple power consumption power traces during the normal operation of the device for different input plaintexts for the same key. Calculating the correlation between the hypothetical power consumption and the measured power consumption for an attacked intermediate value, the secret key can be obtained with the maximum correlation value. Given enough traces, even tiny correlations can be detected. Using DPA, an adversary can obtain secret keys by analyzing correlations obtained from multiple cryptographic operations.

DPA is a more complex and powerful technique than SPA, but the use of SPA is very useful when data-dependent features in the power traces are significant. Based on the SPA, a designer can make decisions to achieve an efficient and secure implementation of a cryptographic algorithm.

The study presented in this paper is focused on the NTRU (Nth Degree Truncated Polynomial Ring Unit) submitted to the NIST post-quantum standardization project. NTRU belongs to the group of post-quantum algorithms based on lattice-based cryptography. Some of the proposed NTRU implementations in the literature are vulnerable to both power [5] and timing analysis [6]. A simple and effective countermeasure to timing attacks is to perform implementations that run in constant time, whereas the protection against power analysis attacks is usually more complex.

Several power analysis attacks using various techniques have been reported on the different NIST PQC candidates. A side-channel assisted chosen ciphertext attack on KYBER is presented in [7], a side-channel attack on the masked implementation of SABER is detailed in [8], and two vulnerabilities in the NTRU algorithm and a side-channel assisted attack to exploit them are reported in [9]. Finally, single-trace side-channel attacks on the message encoding of several of the NIST PQC finalists are described in [10].

This work addresses the issue of SPA performed on PQC, with a particular focus on the NTRU cryptosystem. The main objective of the article is to analyze the effectiveness of these information gathering measures in the context of PQC. The study includes experimental analysis of polynomial multiplication based on the NTRU cryptosystem, which can be susceptible to side-channel attacks using power measurements. Several countermeasures and strategies are also suggested to reduce potential information leakage in this scenario. Overall, this work aims to focus on the importance of securing systems that use PQC that are still vulnerable to side-channel attacks.

This paper is organized as follows. The NTRU is mathematically described in Section II.A, while its hardware implementation is detailed in Section II.B. Pointing out the possibility of a SPA on this implementation, Section III describes the experimental setup used for data acquisition. Section IV exposes the results obtained from the power data acquired. The possible consequences of this study lead to the proposal of several countermeasures in terms of Arithmetic Unit design in Section V, and in terms of modification in the control logic in Section VI. Finally, the conclusions of this work are shown in Section VII.

II. NTRU Implementation

A. Mathematical Background

NTRU's cryptography system relies on a specific mathematical structure known as polynomial convolution rings or quotient rings, which allow performing polynomial operations [11]. The quotient ring shown in (1) represents a truncated polynomial ring of degree N. The set of polynomials whose integer coefficients are reduced module t is denoted by $\mathbb{Z}_t[X]$, while the polynomial $(x^N - 1)$ is used to obtain the modulus of the polynomial arithmetic operations employed by various cryptographic primitives in the current standard version.

$$R_{N,t} = \frac{\mathbb{Z}_t[x]}{(x^N - 1)} \tag{1}$$

A polynomial $P(x)$ is defined by a set of integer coefficients denoted by p_i, that represents the coefficient of $P(x)$ of degree i, as it is shown in (2)

$$P(x) = p_0 + p_1 \cdot x + p_2 \cdot x + \ldots + p_{N-1} \cdot x^{N-1} \tag{2}$$

The multiplication over quotient rings forms the basis of operation of cryptographic algorithms such as the NTRU standardized in IEEE Std 1363.1 [12]. In this context, both encryption and decryption schemes require polynomial multiplication. The multiplication of two polynomial, which generates another polynomial, can be described as: $C(x) = A(x) \times B(x)$. The coefficients of this polynomial are calculated according to (3).

$$c_k = \sum_{j=0}^{N-1} \sum_{i=0}^{N-1} \sum_{i+j=k \bmod N} (a_i \cdot b_j) \bmod t \tag{3}$$

Although recent versions of NTRU have increased the security in the encryption and decryption process, the polynomial multiplication is still a fundamental part of the crypto scheme. For the sake of illustration, focusing on encryption and following the initial NTRU proposal [11], the encrypted message, $c(x) \in R_{N,q}$, is obtained as shown in (4).

$$c(x) = h(x) \times r(x) + m(x) \bmod q \tag{4}$$

Where $h(x) \in R_{N,q}$ comes from the secret key of the cryptosystem, and $r(x) \in R_{N,p}$ is the so-called obfuscation polynomial. In all versions of NTRU, the parameter q that is used in $c(x)$ and $h(x)$ is set to 2048. While the other parameter p is set to 3. Thus, the $r(x)$ polynomial is formed by the coefficients $-1, 0$ and 1, forming a ternary polynomial. Focusing on the multiplication, this can be described as shown in (5), or in terms of the coefficients in (6).

$$e(x) = h(x) \times r(x) \bmod q \tag{5}$$

$$e_k = \sum_{j=0}^{N-1} \sum_{i=0}^{N-1} \sum_{i+j=k \bmod N} (h_i \cdot r_j) \bmod q \tag{6}$$

Both $h(x)$ and $r(x)$ coefficients are secret and cannot be disclosed to a third party. In the NTRU standardized in IEEE Std 1363.1, there is a parameter that set the number of non-zero and zero elements within $r(x)$, but not the exact position. Therefore, the most vulnerable operation seems to be the multiplication itself where information leakage can occur, especially by the reduced elements that form $r(x)$, being one of them 0.

B. Hardware implementation

Although different works such as [13] or [14] follow a full-parallel design, the hardware implementation of this work has followed a serial architecture, applying the same strategy that is used in [15]. Fig. 1 shows the multiplier schematic. The Control Unit block manages the generation of indices i, j, and k, which are used as memory addresses in the Memory block during the operation. It also holds the coefficients of the input polynomials $r(x)$ and $h(x)$. To simplify the study of the multiplication operation, these coefficients are stored in advance within the Memory Block. The Memory block also stores the partial results and the resulting polynomial, $e(x)$, during the operation. The total clock cycles required to perform the complete operation will be $N \cdot N$, being N the number of coefficients of $h(x)$ and $r(x)$.

Previously published works [13], [14] and [15] introduced an Arithmetic Unit (AU), that instead of perform a multiplication between coefficients, uses the reduced number of elements presented in $r(x)$ to perform an addition (for $r_j = 1$) or a subtraction (for $r_j = -1$). This work undergoes some changes to this AU, which instead of using a multiplexer, it now utilizes a logic-gate AND as illustrated in Fig. 2. The r_i coefficient controls the operations of the AU, whose function is summarized in Table I. To hardware description, the Verilog Hardware Description Language (HDL) was employed, while the RTL-based design flow provided by Xilinx Vivado tools was utilized in developing the multiplier module. In terms of occupation, the AU implementation requires 22 LUTs in Xilinx Spartan6 family series.

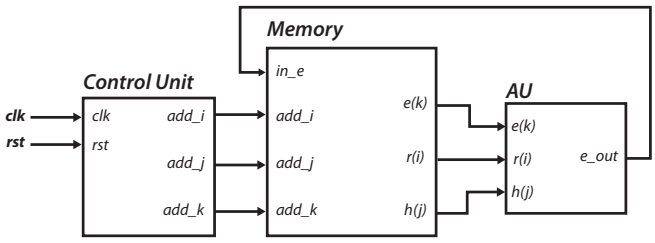

Fig. 1. Block diagram of the hardware implementation.

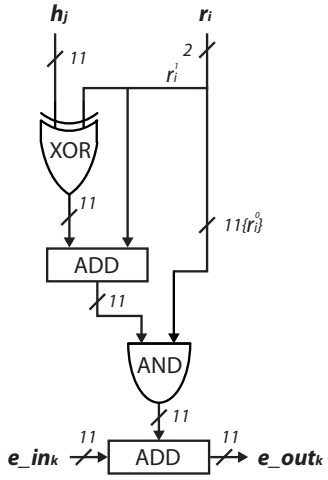

Fig. 2. Arithmetic Unit (AU) used in this work.

As can be observed in Table I, when the AU is operating using the coefficients of $r(x)$, $r_i = 01$ and $r_i = 11$, it performs the addition or the subtraction of the coefficient of $h(x)$, h_j, respectively. However, when a zero coefficient of $r(x)$ ($r_i = 00$) acts as input of the AU, no operation is performed, maintaining at the output the same result as at the input. Therefore, it is evident that, since there is a clear difference in the behavior of the AU, there must also be a difference in the power consumption of the arithmetic module depending on the coefficient of $r(x)$ it has at its input. This may constitute an input vector for an attacker who wants to obtain information regarding the obfuscation polynomial, $r(x)$. To do this, a SPA of the power consumption of the arithmetic circuit would be sufficient to see if there are differences in power consumption corresponding to different values of the coefficients of $r(x)$.

TABLE I
AU OPERATION IN FUNCTION OF r_i.

r_i	Operation
00	$e_{out_k} = e_{in_k}$
01	$e_{out_k} = e_{in_k} + h_j$
11	$e_{out_k} = e_{in_k} - h_j$

III. Experimental Setup

To analyze the SPA flaws of different hardware proposals, the design has been implemented in FPGA. It has been used the SAKURA-G board [16], specifically designed to test cryptographic hardware implementations against SCAs. The SAKURA-G board, is composed by 2 FPGAs, namely controller FPGA and crypto FPGA (both Xilinx Spartan6 FPGA), where the cryptographic module can be isolated from the rest of the design to measure its power consumption with high precision. SAKURA-G has specific circuitry to measure the power consumption traces, which, along with a PC and an oscilloscope, allows the SPA attack to be carried out.

Fig. 3 shows an scheme of the used experimental setup. The Device Under Test (DUT) is the attacked hardware module implemented on the crypto Spartan6 FPGA of SAKURA-G borad. The used PC (i7, 64 RAM, Windows 10) runs Matlab and the software to control the oscilloscope. The used oscilloscope is the PicoScope 3406D, with bandwidth of 200MHz and 250MS/s.

Several power-consumption traces have been experimentally measured for each implementation and then processed under Matlab to perform the SPA attack.

IV. SPA of the NTRU AU

The results of the power analysis of the AU used in this work are shown in Fig. 4, where the analyzed power trace is shown in blue in the upper graph. These data correspond to a

selection of a few cycles of operation. Specifically, about 45 coefficients of $r(x)$ being operated each of them N times by the coefficients of $h(x)$. The bottom graph is used to clarify the data presented in the power trace using `findchangepts` Matlab function. The orange line represents the jumps that are occurring in the power consumption of the module. Just below this line are represented the r_i coefficients that are being used in each cycle (framed by vertical black lines) to operate in the arithmetic module. As can be seen, firstly, there is a difference in the power consumption of the multiplication module and secondly, this difference is exclusively due to when null coefficients in $r(x)$ are being used.

The above can be summarized as follows: two thresholds of power consumption corresponding to null elements and non-null elements could be clearly established. This is a clear vulnerability that allows knowing not only the number of null elements, but also their position in $r(x)$, which would mean a reduction of the search space for a possible attack on the cryptosystem. Therefore, several countermeasures are proposed that could mitigate the problem in advance.

V. SPA Of The Countermeasures Proposed

The design of countermeasures in this work seeks to mitigate the differences in power consumption that affect the current AU by maintaining the functionality. The first proposed countermeasure tries to obfuscate the power consumption instantiating a dummy AU, without considering its output as can be observed in Fig. 5. The function of this AU is to operate in the opposite way than the original one, thereby maintaining equal power consumption when working with 0 (then the dummy would act as a -1) as when not (then the dummy would act as a 0). The second proposed countermeasure tries to mitigate the differences in the power consumption instantiating 3 AUs whose inputs are the three possible values of the coefficients of $r(x)$, as it can be observed in Fig. 6. The output of the operation is controlled by a multiplexer that uses the coefficient r_i to select between inputs. In this case, since the 3 AUs are operating at the same time, the power consumption should not suffer great differences beyond the difference in consumption that may occur in the selection of the multiplexer. In terms of occupation, the first proposed countermeasure requires 45 LUTs, while the second one 110 LUTs in Xilinx Spartan6 family series.

SPAs are also performed to corroborate if there are still differences in the power consumption of these two proposed countermeasures. The results are shown in Fig. 7. In these graphs it is possible to visualize the power traces in blue, as well as, over them, an orange line that allows to clarify and observe the existence of changes in the power due to changes in the coefficients of $r(x)$, for which the same Matlab function used in Fig. 4 has been applied. In order from top to bottom,

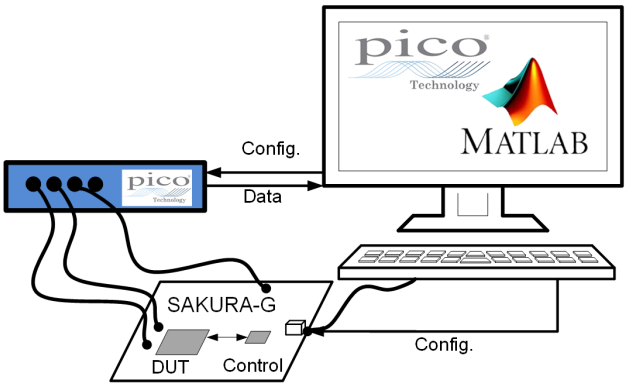

Fig. 3. Experimental setup scheme.

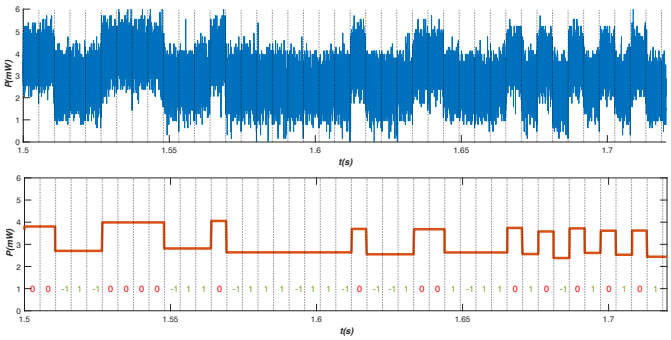

Fig. 4. SPA of the NTRU multiplication module

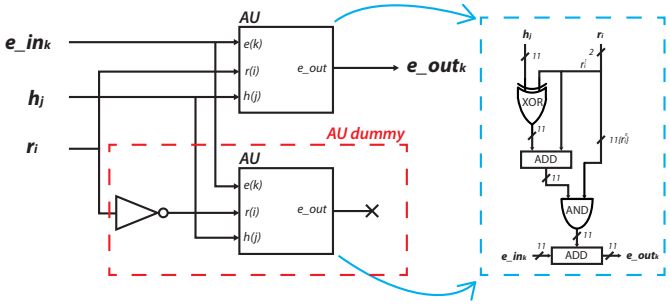

Fig. 5. First countermeasure proposed.

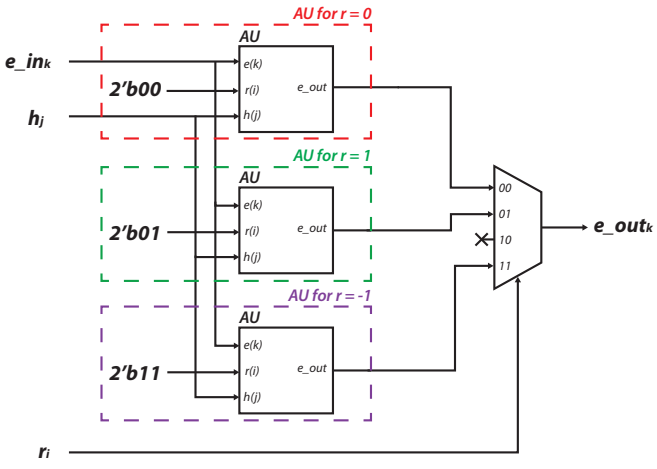

Fig. 6. Second countermeasure proposed.

the power consumption of the NTRU multiplication module with the original AU, with the first countermeasure and, with the second countermeasure are shown. The last graph shows the comparison between the different power consumption of the three designs. As can be observed, there is hardly any difference in the power consumption either between different countermeasures or between the countermeasures and the original case. This supposes that the proposed countermeasures are continuing to produce a noticeable impact on the power usage, even when a null coefficient is employed. As a result, it is still feasible to determine the quantity and location of the zero coefficient of $r(x)$, not being effective in mitigating the differences in power consumption, and therefore, generating information leaks. This may be because the extra circuitry added (the inverter in one case and the multiplexer in another) is very sensitive to changes in the input bits and therefore their inclusion still leaks information. The control logic could also have some effect when operating with zero elements. At this point, from the algorithmic point of view, it is interesting to study the SPA of the recently-published accelerated version of the NTRU cryptosystem.

VI. SPA Of The Accelerated Algorithm

Since the previously formulated countermeasures have not been successful in reducing the differences in power consumption that may lead to information leakage, a new analysis is proposed to corroborate if the latest version of the hardware implementation of the algorithm first presented in [17] still produces this type of information leakage. The idea behind this countermeasure is to hide the most sensitive coefficients without affecting the operation. As previously mentioned, the number of cycles required to perform the multiplication operation would be $N \cdot N$, however, when $r_i = 0$ no operation is performed, so at the algorithmic level not operating or operating with 0 it is exactly the same. This was already applied in [15] to reduce the number of cycles in the complete operation of the NTRU cryptosystem. Thus, if a jump is considered in the multiplication operation when a null coefficient is found in $r(x)$, the pseudocode of the operation could be written as shown in Algorithm 1.

Algorithm 1 The polynomial multiplication using only non-zero elements

```
for i = 0 : N − 1 do
    if r_i ≠ 0 then
        for j = 0 : N − 1 do
            k = mod(j + i, N)
            e_k = e_k + (h_j · r_i)
        end for
    end if
end for
```

Once the Control block is modified to carry out the algorithm modification, the SPA can corroborate if there are still differences in the power consumption. The results can be shown in Fig. 8. In the same way as in the previous figures, the power traces are in blue and a line is shown in orange that makes it easier to see if there are jumps in power consumption. In the upper graph, we have again the power consumption of the NTRU multiplier with the original AU presented in Fig. 2 while, in the middle graph, we have the application of the algorithm acceleration. Although the accelerated algorithm must be applied from the start of the multiplication operation, in this case, to illustrate the change in power consumption, such acceleration is applied to the middle of the display window.

When applying the acceleration, it can be observed how

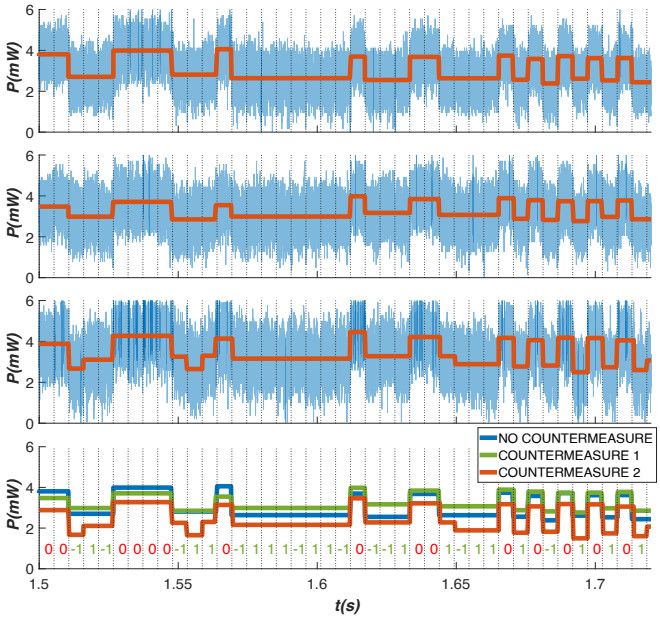

Fig. 7. SPA of the NTRU multiplication module with the original AU and the two countermeasures proposed.

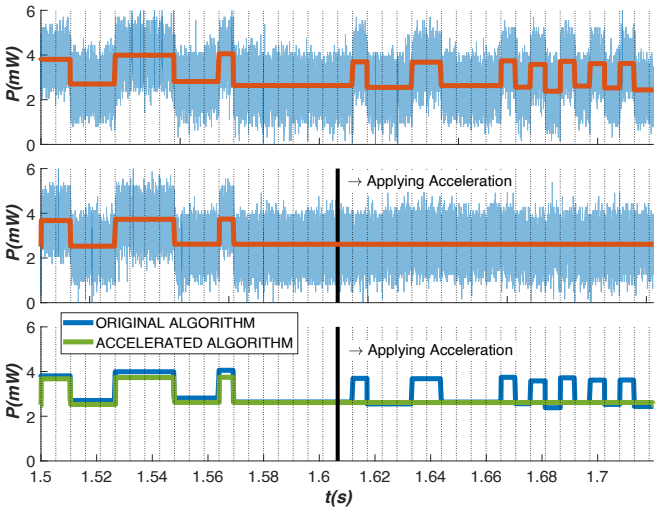

Fig. 8. SPA of the NTRU multiplication module with the original algorithm versus the application of the accelerated version.

when jumping the null coefficients there is no longer a difference in power between different values of r_i. Apart from that, Fig. 9 shows one of the operation cycle in detail from the previous analysis. Since a jump is occurring when a null-coefficient appears in the accelerated version of the algorithm, it is conceivable that the null-coefficient jump itself may cause some information leakage. In the detail it can be seen how it is impossible to locate where the null-coefficient jump is occurring in the accelerated version. In this way, an attacker who is carrying out a SCA attack, through the use of a SPA, will not be able to obtain any type of internal information, either the number or the position of the null coefficients. This solution, therefore, is not only faster but also avoids information leaks caused by AU.

VII. Conclusions

In conclusion, this work highlights the importance of addressing the vulnerabilities posed by SCAs in the NTRU cryptosystem, that is among the so-called PQC algorithms. These attacks attempt to breach the security of cryptographic systems by exploiting information gained from the physical implementation of the system, such as power consumption through SPAs. These analysis have demonstrated that the initial AU proposed can be susceptible to information leakage, which underscores the need for implementing robust countermeasures. For that, several countermeasures have been proposed, whose SPAs have also demonstrated that their effectiveness is limited to mitigate the risk of presenting security breaches. Additionally, the accelerated version of the algorithm is also analyzed. The results of this analysis suggest that it does not present any information leakage, which is encouraging for its practical deployment in secure communication systems. Ultimately, this work attempts to focus on the importance of power analysis and by extension the possibilities of SCA in PQC, whose implementation is currently under development.

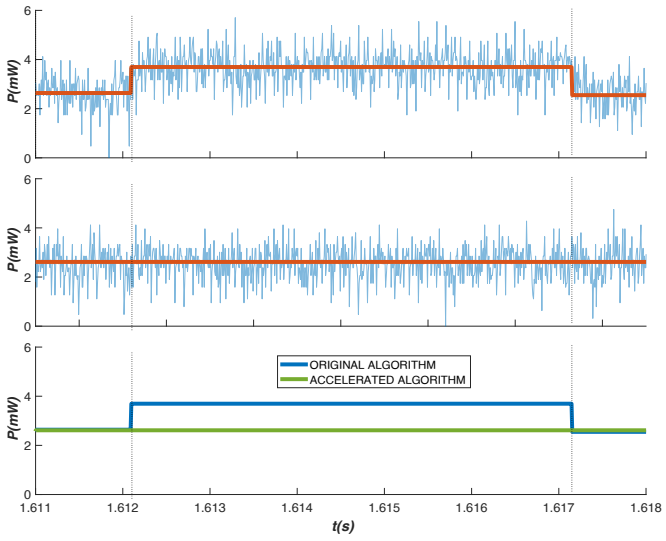

Fig. 9. Detail of the SPA of the NTRU multiplication module with the original algorithm versus the application of the accelerated version.

References

[1] NIST, "Post-Quantum Cryptography," July 2022, https://csrc.nist.gov/projects/post-quantum-cryptography.

[2] M. Azouaoui, Y. Kuzovkova, T. Schneider, and C. van Vredendaal, "Post-quantum authenticated encryption against chosen-ciphertext side-channel attacks," Cryptology ePrint Archive, Paper 2022/916, 2022, https://eprint.iacr.org/2022/916. [Online]. Available: https://eprint.iacr.org/2022/916

[3] J. Park, N. N. Anandakumar, D. Saha, D. Mehta, N. Pundir, F. Rahman, F. Farahmandi, and M. M. Tehranipoor, "Pqc-sep: Power side-channel evaluation platform for post-quantum cryptography algorithms," *IACR Cryptol. ePrint Arch.*, vol. 2022, p. 527, 2022.

[4] Y. Zhou and D. Feng, "Side-channel attacks: Ten years after its publication and the impacts on cryptographic module security testing," Cryptology ePrint Archive, Paper 2005/388, 2005, https://eprint.iacr.org/2005/388. [Online]. Available: https://eprint.iacr.org/2005/388

[5] P. Ravi, A. Chattopadhyay, and S. Bhasin, "Practical side-channel and fault attacks on lattice-based cryptography," in *2021 IFIP/IEEE 29th International Conference on Very Large Scale Integration (VLSI-SoC)*, 2021, pp. 1–2.

[6] B. Liu and H. Wu, "Efficient multiplication architecture over truncated polynomial ring for ntruencrypt system," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, pp. 1174–1177.

[7] P. Ravi, S. Sinha Roy, A. Chattopadhyay, and S. Bhasin, "Generic side-channel attacks on cca-secure lattice-based pke and kems," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 3, p. 307–335, Jun. 2020. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/8592

[8] K. Ngo, E. Dubrova, Q. Guo, and T. Johansson, "A side-channel attack on a masked ind-cca secure saber kem," Cryptology ePrint Archive, Paper 2021/079, 2021, https://eprint.iacr.org/2021/079. [Online]. Available: https://eprint.iacr.org/2021/079

[9] A. Askeland and S. Rønjom, "A side-channel assisted attack on ntru," Cryptology ePrint Archive, Paper 2021/790, 2021, https://eprint.iacr.org/2021/790. [Online]. Available: https://eprint.iacr.org/2021/790

[10] B.-Y. Sim, J. Kwon, J. Lee, I.-J. Kim, T. Lee, J. Han, H. Yoon, J. Cho, and D.-G. Han, "Single-trace attacks on the message encoding of lattice-based kems," Cryptology ePrint Archive, Paper 2020/992, 2020, https://eprint.iacr.org/2020/992. [Online]. Available: https://eprint.iacr.org/2020/992

[11] J. Hoffstein, J. Pipher, and J. H. Silverman, "Ntru: A ring-based public key cryptosystem," in *Algorithmic Number Theory*, J. P. Buhler, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 267–288.

[12] "Ieee standard specification for public key cryptographic techniques based on hard problems over lattices," *IEEE Std 1363.1-2008*, pp. 1–81, 2009.

[13] E. Camacho-Ruiz, M. C. Martínez-Rodríguez, S. Sánchez-Solano, and P. Brox, "Accelerating the development of ntru algorithm on embedded systems," in *2020 XXXV Conference on Design of Circuits and Integrated Systems (DCIS)*, 2020, pp. 1–6.

[14] E. Camacho-Ruiz, S. Sánchez-Solano, P. B. Jiménez, and M. C. Martínez-Rodríguez, "Timing-optimized hardware implementation to accelerate polynomial multiplication in the ntru algorithm," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 17, pp. 35:1–35:16, 2021.

[15] S. Sánchez-Solano, E. Camacho-Ruiz, M. C. Martínez-Rodríguez, and P. Brox, "Multi-unit serial polynomial multiplier to accelerate ntru-based cryptographic schemes in iot embedded systems," *Sensors*, vol. 22, no. 5, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/5/2057

[16] H. Guntur, J. Ishii, and A. Satoh, "Side-channel attack user reference architecture board sakura-g," in *IEEE 3rd Global Conference on Consumer Electronics (GCCE'14)*. IEEE, 2014, pp. 271–274.

[17] X. Zhan, R. Zhang, Z. Xiong, Z. Zheng, and L. Zhenglin, "Efficient implementations of ntru in wireless network," *Communications and Network*, vol. 05, pp. 485–492, 01 2013.