International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

# COPY RIGHT

ELSEVIER SSRN



IJIEMR Transactions, online available on 24th Jan 2023.

Link : https://ijiemr.org/downloads/Volume-12/Issue-01

**Title: Hardware Accelerator Formulated Password recovery in Hybrid CPU-FPGA Devices**

volume 12, Issue 01, Pages: 873-878

Paper Authors: **Amshiya Mohammedkasim & Dr. S. Senthilkumar**


USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

# Hardware Accelerator Formulated Password recovery in Hybrid CPU-FPGA Devices

[1]Amshiya Mohammedkasim & [2]Dr. S. Senthilkumar

[1]PhD Scholar, Dept.:- ECE, SVS College of Engineering and Technology, Coimbatore, Tamilnadu, India

[2]Professor, Dept.:- ECE, SVS College of Engineering and Technology, Coimbatore, Tamilnadu, India

**Abstract** Password recovery tools are needed to recover lost and forgotten passwords so as to regain access to valuable information. As the process of password recovery can be extremely compute-intensive, hardware accelerators are often needed to expedite the recovery process. This paper thus presents a high performance, energy-efficient accelerator built upon modern hybrid CPU-FPGA SoC devices. The proposed password recovery accelerator relies on the development of a set of intellectual property (IP) cores for implementing variety of encryption algorithms with vastly different characteristics and complexities. To keep the resource requirements of each IP core running on a resource-strapped FPGA to the minimum, while achieving the highest throughput possible, the most performance critical computational hash functions are mapped to the FPGA with two specific optimization techniques, namely the fixed message padding for hashing and loop transformation for deep pipelining. The proposed password recovery accelerator implements a non-blocking deep pipeline design that does not incur any data and structural hazards, which is made possible by applying a task scheduling scheme through the use of block RAMs. Synchronization between tasks that are mapped to run separately on CPU and FPGA is achieved through task reordering and a communication protocol for maximum parallelism and low overhead. The proposed design is evaluated on Xilinx XC7Z030-3 device, and it is compared much favourably with other known implementations. The proposed hardware accelerator design is found 12.5 and 3.1 times more resource-efficient than the pure FPGA-based password recovery accelerators for TrueCrypt and WPA-2, respectively. The proposed implementation also shows a remarkable >200 percent improvement in terms of energy efficiency over a state-of-the-art implementation on NVIDIA GTX 750 Ti GPU.

## 1. Introduction

With hundreds of thousands of passwords lost or forgotten every year, valuable protected information becomes unavailable to the legitimate owner or authorized law enforcement personnel. In this case, regaining the control of precious, encrypted data relies on the use of password recovery tools by means of password cracking. Brute-force password cracking, attempts to recover a password by simply trying all possible passwords until the correct one is eventually hit. The amount of computation time needed to find a correct password depends on the password length and complexity of the character set, as well as computational complexity of the encryption algorithm. Although dictionary-based Markov model theory and hints or probability provided by users can help reduce the search space, password recovery by nature is a time-consuming process.

Since large amount of computation is required during the brute-force password recovery process, many hardware accelerators have been reported and the most widely used hardware platforms are field programmable gate array (FPGA) and graphic processing unit (GPU). One big problem of most FPGA-based accelerators is that they only focus on optimizing their throughput for single hash algorithm (e.g., SHA-256) or implementing for one particular encryption algorithm, making them inflexible in handling many other algorithms that have been developed for cryptography. GPU-based solutions, on the other hand, are flexible enough to handle different types of cryptography algorithms, but they tend to have high energy footprint, as a result of sequential execution of cryptographic algorithms and GPU's fixed-width data path architectures with limited bandwidths.

## 2 Password-Based Cryptography And Password Recovery

In this section, we will first give a brief introduction to password- based cryptography, including both the encryption and decryption processes. If a password is not

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

available to allow direct access to password-encrypted data/file, password recovery has to take place, and the processes and methods for password cracking are reviewed in Section 2.2. Different platforms, hardware, software, or mixture of both, can be used for the purpose of password recovery, and pros and cons of these platforms are assessed in this section as well

2.1 Password based Cryptography

2.2 Password Recovery/Cracking

2.3 Hardware for Password Recovery

## Algorithm Modification For FPGA-Friendly Implementation

For a higher throughput and better resource utilization, two algorithm modifications are proposed in this section, including fixed message padding for hashing and loop transformation for deep pipelining. For the sake of presentation, cryptographic application of RAR5 will be taken as an example. These two modifications can be well applied to all other cryptographic applications

## Encryption-specific Fixed Message Padding for Hashing

The encryption algorithm for RAR5 is illustrated in Fig. 4, where SHA-256 is employed as its cryptographic core. Inside the encryption algorithm, SHA-256 is performed twice to obtain two 256-bit long numbers (referred to as state-I and state-O in Fig. 4) and these two numbers serve as the initial state inputs of the remaining SHA-256 operations. The 128-bit salt is padded with a fixed 384-bit message to form a 512-bit message block for the first hash message authentication code (HMAC) iteration which needs to perform SHA-256 twice (referred to as in-HMAC and out-HMAC in Fig. 4). For the following 32,799 times of HMAC iterations, the 512-bit message blocks for the input are generated by padding the 256-bit digest output from the last SHA-256 operation with a fixed 256-bit message (0x80000000,0,0,0,0,0,0,0x300).

Inside the SHA-256 hash function, each 512-bit input message block is expanded and fed to the 64 rounds of data scrambling in words of 32-bit each (denoted by Wt, $0 \le t \le 63$). Since the input message block only has 16 32-bit words (denoted Mt, $0 \le t \le 15$), the remaining 48 32-bit words are obtained from message expansion, according to the following equations:

Equation 1

$\sigma_o(x) = ROT\ R(x, 7) \oplus ROT\ R(x, 18) \oplus SH\ R(x, 3)$

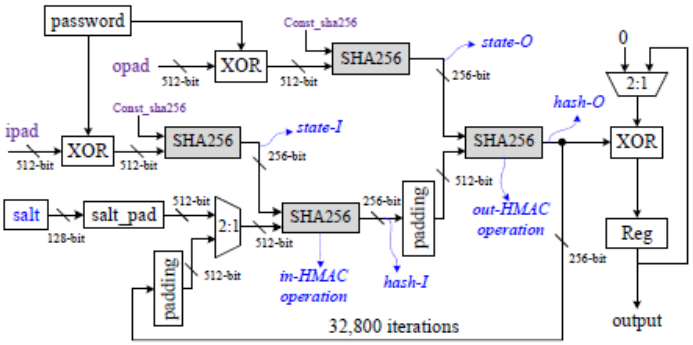


Figure 1 The encryption built into RAR5 format with multiple iterations of SHA-256

Equation 2

$\sigma_1(x) = ROT\ R(x, 17) \oplus ROT\ R(x, 19) \oplus SH\ R(x, 10)$

Equation 3

$W_t = \{M_t \qquad 0 \le t \le 15$

$\{\sigma_1(W_{\square -2}) + \sigma_o(W$

where ROT R (x, n) is a circular shift right function that shifts binary x by n positions, and SHR (x, n) is a logic shift right function that shifts x by n positions. All additions in the SHA- 256 algorithm are modulo 232.

A pipelined SHA-256 design requires 48 message expansion modules, each of which consists of two shift-transform modules (referred as σ0 and σ1 in Fig. 5) and three 32-bit adders. To reduce the resource consumption introduced by the message expansion, a fixed message padding method is proposed. As shown in Fig. 5, since most of the SHA-256 computation tasks in the RAR5 encryption algorithm share a fixed 256-bit message padding that has many zeros, significant number of operations can actually be eliminated to reduce the hardware cost. For example, as the calculation of W16 is related to W9 and W14 whose values are zeros, equation (3) can be simplified as

Equation 4

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

$$W_{16} = \sigma_1(W_{14}) + W_9 + \sigma_0(W_1) + W_0 = \sigma_0(W_1) + W_0$$

where one shift-transform module (σ1) and two adders are eliminated (as shown in the upper part of Fig. 5). For another example, as the calculation of W25 is related to W9 and W10 whose values are also zeros, equation (3) can be simplified as Equation 5

$$W_{25} = \sigma_1(W_{23}) + W_{18} + \sigma_0(W_{10}) + W_9 = \sigma_1(W_{23}) + W_{18}$$

**EVALUATION**

In order to evaluate the performance and cost of the proposed accelerator for password recovery, we have built the hardware prototype based on the Xilinx Zynq 7000 series FPGA (XC7Z030-3), which comes with programmable fabric and two ARM Cortex-A9 processors clocked at 500 MHz (the highest clocking frequency can be scaled to 1 GHz). The system performance is evaluated on an FPGA cluster with 16 FPGA chips connected by a Gigabit Ethernet interface. IP cores located in accelerator library are synthesized and implemented using the Xilinx VIVADO (v2016.3) tool set. To verify the resilience of the pr co W Tr rel dif the pa an dir ex pa us is pa po is su

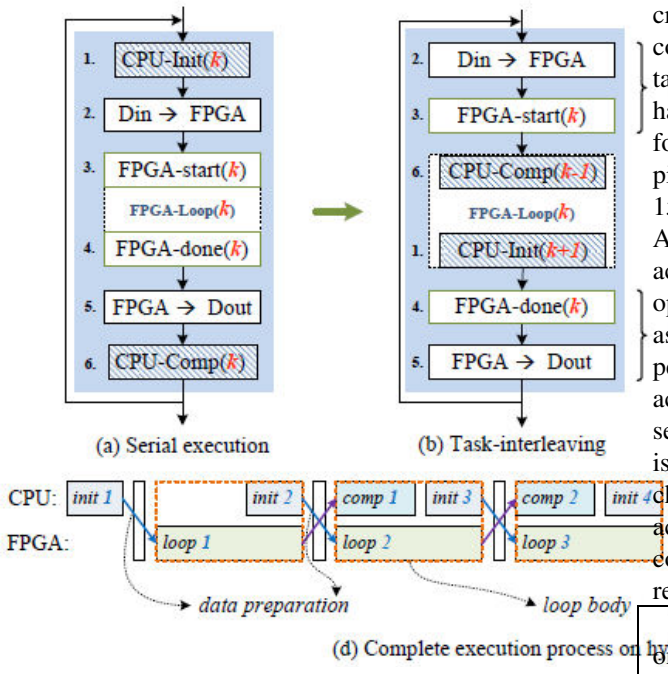


Figure 2 Synchronization in the proposed accelerator system

| Applications | Cluster Speed ( passwords/s ) | Single XC7Z030-3 FPGA | | |
|---|---|---|---|---|
| | | Speed (passwords/s | Measurement Power( | Energy Effi. ( passw |
| | | | W ) | ords/J ) |
| RAR5 | 91,286 | 5,711 | 12.95 | 441 |
| WAP-2 | 924,007 | 57,780 | 14.89 | 3,880 |
| Office 2007 | 312,917 | 19,635 | 12.35 | 1,590 |
| Office 2010 | 156,619 | 9,837 | 12.55 | 784 |
| Office 2013 | 15,619 | 976 | 10.65 | 92 |
| True Crypt | 325,123 | 20,443 | 7.54 | 2,711 |

Table 1 System performance of the proposed accelerator among different applications (a cluster with 16XC7Z030-3 FPGA)

Hardware Implementation Analysis

The detailed hardware implementations for different cryptographic applications are summarized in Table 3. The construction of each accelerator core is dependent on the target encryption applications, and each core contains one hash function pipeline. In the design of accelerator IP cores for TrueCrypt, apart from customizing a RipeMD-160 pipeline to accelerate the repeated run of RipeMD-160 for 15,996 times in the looping phase for each password, an AES-256 core is instantiated on the FPGA to further accelerate the AES-256 encryption and decryption operations. The throughput of an accelerator core is defined as the number of hashing operations that can be completed per second. For example, the throughput of a RAR5 accelerator core is 189 million SHA-256 operations per second, while the throughput of a WPA-2 accelerator core is 247.2 million SHA-1 operations per second. To characterize the computing power of the reconfigurable accelerated device, the maximum number of accelerator cores that can be placed on one XC7Z030-3 FPGA is also reported in column 5 of Table 3.

| Applications | Construction of each accelerator core | Freq (MHz) | Throughput per core (MOPS/s) | Max NUM. |
|---|---|---|---|---|
| RAR5 | SHA-256 pipeline | 190 | 189.0 | 2 |
| WAP-2 | SHA-1 pipeline | 250 | 247.2 | 4 |
| Office 2007 | SHA-1 pipeline | 250 | 247.3 | 4 |


Volume 12,Issue 01 Jan 2023 ISSN : 2456- 5083

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

| | | | | |
|---|---|---|---|---|
| Office 2010 | SHA-1 pipeline | 250 | 247.3 | 4 |
| Office 2013 | SHA-512 pipeline | 100 | 98.7 | 1 |
| True Crypt | Ripe MD-160 pipeline | 190 | 188.4 | 2 |
| | AES-256 Module | 190 | 1.79 | |

Table 1 Hardware implementation on one XC7Z030-3 FPGA

**Pipeline with fixed message padding**

Pipelined implementations with the fixed message padding technique can significantly reduce the resource consumption and improve the performance of hash function units. Defined as bits of data that can be processed per slice per second, throughput per area is often used to evaluate the performance of hardware hash function unit. In our proposed design, it takes only 8,153 slices to construct a deep SHA-256 pipeline with the 256-bit fixed message padding, clocked at 190 MHz in the case of RAR5, reaching a performance of 5.9 Mbps per slice (this SHA-256 pipeline can process 256-bit of data for every clock cycle). Co
on
sli

**I**
**ba**

bu
ha
pi
thr
va
inc
fil

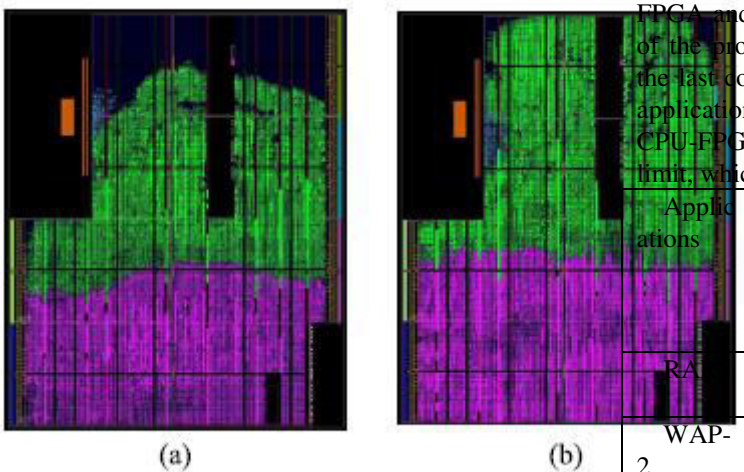


Fig. 14. The RAR5 designs after place and route: (a) two 64-stage SHA-256 pipelines clocked at 114 MHz, and (b) two 106-stage SHA-256 pipelines clocked at 190 MHz.

Trade-Off Between Depth Of Hash Function Pipeline And Resource Utilization

This technique helps reduce the unused resources in the FPGA and improve the operating frequency of hash function pipeline, thus the throughput. Two RAR5 designs after place and route are shown in Fig. 14. When the SHA-256 pipeline is implemented with 64 stages, two accelerator cores each clocked at 114 MHz can be accommodated by the target FPGA; one can see significant amounts of resource is wasted. By applying the depth of hash function pipeline exploring technique, the alternative design with a deeper pipeline can operate on the highest achievable frequency of 190 MHz. In this particular example, the performance improvement
is 1.67 folds.

**Synchronization between FPGA and CPU**

As discussed before, the computation tasks of all the cryptographic applications are divided into three phases, where the tasks in the initialization phase and the comparison phase are assigned to the CPU, while the tasks in the looping phase are assigned to the FPGA. For the convenience of analysis, we gather the timing information of each phase by using the timers available in the ARM core. Table 4 lists the required times for task execution and data transmission per password during different phases on a single FPGA chip, where columns Init, Loop, and Comp report the computation time for different applications running on corresponding devices (the CPU and the FPGA), while the data transfers between the CPU and the FPGA are given as T1 and T2.

Generally speaking, the execution time of a heterogeneous system is longer than the execution time of any individual unit of the system. As a result, the parallel efficiency in our case is determined as Tfpga=Tsystem, where Tfpga and Tsystem represent the time spent on the FPGA and the system, respectively. The parallel efficiency of the proposed synchronization mechanism is reported in the last column of Table 4. One can see that across all the applications, the parallel efficiency of the hybrid CPU-FPGA device is at least 96.0 percent of the theoretical limit, which is the case for application of WPA-2.

| Applications | Time for execution and transmission (μs/password) | | | | | | Parallel Effi. |
|---|---|---|---|---|---|---|---|
| | Init. | T I | Loop | T 2 | Comp. | Sys. | |
| RA | 19.1 | 1.01 | 173.5 | 0.08 | 0.4 | 175.1 | 99.3% |
| WAP-2 | 2.9 | 0.32 | 16.6 | 0.26 | 13.8 | 17.3 | 96.0% |
| Office 2007 | 3.0 | 0.16 | 50.6 | 0.16 | 21.7 | 51 | 99.2% |
| Office 2010 | 3.2 | 0.16 | 101.1 | 0.16 | 21 | 101.7 | 99.4% |
| Office 2013 | 46.5 | 1.28 | 1,012.2 | 1.28 | 169.6 | 1,027.5 | 99.1% |
| True Crypt | 46.8 | 1.26 | 47.2 | 0.00 | 0.0 | 48.9 | 96.5% |

Table 2 Parallel efficiency of the proposed synchronization mechanism

| Application | TrueCrypt | WAP-2 |
|---|---|---|


Volume 12,Issue 01 Jan 2023 ISSN : 2456- 5083

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

| s | | | | |
|---|---|---|---|---|
| Implementation | [8] | This work | [9] | This work |
| FPGA type | Spartan-6 I.X150 | XC7Z030-3 | Spartan-6 LX150 | XC7Z030-3 |
| FPGA Number | 128 | 16 | 36 | 16 |
| Resource efficiency comparison on single FPGA | | | | |
| Speed (passwors/s) | 1,917 | 20,443 | 21,871 | 57,780 |
| Resources (slices) | 23,038 | 19,650 | 23,038 | 19,650 |
| Resources efficiency (passwords/s/slice) | 0.083 | 1,040 | 0.949 | 2.940 |
| Speedup | Ix | 12.5x | 1x | 3.10x |
| Speed comparison on FPGA cluster | | | | |
| Speed (passwors/s) | 245.397 | 325,123 | 741,200 | 924,007 |
| Speedup | 1x | 1.32x | 1x | 1.25x |
| Applications | TrueCrypt | | WAP-2 | |
| Implementation | [8] | This work | [9] | This work |
| FPGA type | Spartan-6 I.X150 | XC7Z030-3 | Spartan-6 LX150 | XC7Z030-3 |
| FPGA Number | 128 | 16 | 36 | 16 |
| Resource efficiency comparison on single FPGA | | | | |
| Speed (passwors/s) | 1,917 | 20,443 | 21,871 | 57,780 |
| Resources (slices) | 23,038 | 19,650 | 23,038 | 19,650 |
| Resources efficiency (passwords/s/slice) | 0.083 | 1,040 | 0.949 | 2.940 |
| Speedup | Ix | 12.5x | 1x | 3.10x |
| Speed comparison on FPGA cluster | | | | |
| Speed (passwors/s) | 245.397 | 325,123 | 741,200 | 924,007 |
| Speedup | 1x | 1.32x | 1x | 1.25x |

Table 3 Comparison of hardware accelerators based on FPGA cluster

**Comparison with Other Schemes**

**Comparison with FPGA-only implementations**

To facilitate a fair comparison, we consider two figures of merit, namely the resource efficiency and the password recovery speed. The resource efficiency is defined as the system speed normalized with respect to available hardware resources (i.e., the number of slices in FPGA). To the best of our knowledge, FPGA-based accelerators for password recovery on the cryptographic applications of RAR-5, office 2007, office 2010, and office 2013 have not been disclosed. As a result, we can only compare our implementations of TrueCrypt and WPA-2 with the existing studies, as listed in Table 5. Note that all the existing designs can only deal one specific type of cryptographic application.

An implementation based on a single FPGA cluster with 128 Xilinx Spartan-6 LX150 FPGA chips was reported for TrueCrypt acceleration, and each FPGA consists of eight RipeMD-160 pipelines and two AES-256 XTS cores clocked at 66 MHz, reaching a speed of 245,397 passwords per second. However, the implementation of RipeMD-160 is not resource-efficient, as its pipeline contains only five rounds and it takes 16 cycles to finish one time of RipeMD-160 hashing, resulting in a resource efficiency of only 0.083 passwords/s/slice, far less than the resource efficiency of our implementation. Even though we have only 16 FPGA chips in our proposed implementation and only two RipeMD-160 pipelines are instantiated on one FPGA, the throughput of 325,123 passwords per second is 1.32 times faster than the design reported.

Another implementation for password recovery of WPA-2 was also based on a single FPGA cluster. With 36 Spartan- 6 LX150 FPGA chips, each of which consists of two SHA-1pipelines operating at 187 MHz, that design reaches performance of 741,200 passwords per second. However, since a standard SHA-1 pipeline is adopted in that scheme, only two SHA-1 pipelines can be placed on a single FPGA. In our design, we customize the SHA-1 pipeline with the fixed message padding technique, which makes it possible to place four SHA-1 pipelines working at

250 MHz on a single FPGA chip. The resource efficiency of our proposed design is thus 3.1 times better than the design reported.

| Applications | RAR5 | WPA-2 | Office 2007 | Office 2010 | Office 2013 | True Crypt |
|---|---|---|---|---|---|---|
| System throughput (passwords/s) | | | | | | |
| This work | 5,711 | 57,780 | 19,635 | 9,837 | 979 | 20,443 |
| Hashcat | 5,384 | 61,883 | 21,195 | 10,559 | 1,374 | 40,365 |
| Speedup | 1.1 | 0.9 | 0.9 | 0.9 | 0.7 | 0.5 |
| Power (Watt) | | | | | | |



Volume 12,Issue 01 Jan 2023 ISSN : 2456- 5083

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

| | | | | | | |
|---|---|---|---|---|---|---|
| This work | 12.95 | 14.89 | 12.35 | 12.55 | 10.653 | 7.5434.60 |
| Hashcat | 33.54 | 38.05 | 38.08 | 37.92 | 4.95 | |
| Energy Efficient (passpordws/J) | | | | | | |
| This work | 441.00 | 3,880.46 | 1,589.88 | 783.82 | 91.92 | 2,711.27 |
| Hashcat | 160.52 | 1,626.36 | 556.59 | 278.45 | 39.31 | 1,166.62 |
| Speed up | 2.75 | 2.39 | 2.86 | 2.81 | 2.34 | 2.32 |

Table 4 Comparison against the design using Nvidia GTX750 Ti GPU

**Comparison with GPU scheme**

Table 6 compares the throughput and energy efficiency between the Hashcat v3.6 software implementations on G PU and the proposed accelerator on hybrid CPU-FPGA cluster. The energy consumption of GPU is reported by the NVIDIA System Management Interface.

Computing performance of our hardware accelerator with one embedded FPGA is on a par with the implementation using an NVIDIA GTX 750 Ti GPU. The energy efficiency of our proposed accelerator is 2.32~2.86 times better than that of implementation using an NVIDIA GTX 750 Ti GPU.

## 3. Conclusion

In this paper, we have proposed a high-speed and energy efficient accelerator for password recovery. For the construction of hardware accelerator with high throughput and low resource consumption, optimizations were performed at both algorithm and hardware implementation levels, including deep pipelining with the fixed message padding for hashing and loop transformation based on BRAM-based scheduler, and exploration between depth of hash function pipeline and FPGA resource constraints, as well as synchronization between the FPGA and the CPU for maximum parallelism.

Compared with the FPGA-only password recovery accelerators, the proposed design improves the resource efficiency by 12.5 and 3.1 times for encryption applications of TrueCrypt and WPA- 2, respectively. Compared with the GPU-based password recovery implementation Hashcat with nearly identical performance, our FPGA-based accelerator increases the energy efficiency by 2.32~2.86 times.

## Acknowledgements

The success of this phase of the project required great amount of guidance and assistance from many people, and I and extremely fortunate to have got this all along this completion of this phase.

I convey my sincere thanks to all the staff members who guide me throughout my course. My sincere thanks to **GOD, PARENTS** and **FRIENDS** who were helped encouraged me during the entire course of this project

We are very grateful to experts for their appropriate and constructive suggestions to improve this template.

## REFERENCES

[1] [1] R. Morris and K. Thompson, “Password security: A case history,” Commun. ACM, vol. 22, no. 11, pp. 594–597, Nov. 1979.

[2] [2] P. Oechslin, “Making a faster cryptanalytic time-memory trade-off,” in Proc. Int. Conf. Adv. Cryptol. (CRYPTO), 2003, pp. 617–630.

[3] [3] A. Narayanan and V. Shmatikov, “Fast dictionary attacks on passwords using time-space trade off,” in Proc. ACM Conf. Comput. Commun. Secur. (CCS), Nov. 2005, pp. 364–372.

[4] [4] M. Weir, S. Aggarwal, B. d. Medeiros, and B. Glodek, “Password cracking using probabilistic context-free grammars,” in Proc. IEEE Symp. Secur. Privacy (SP), May 2009, pp. 391–405.

[5] [5] R. P. McEvoy, F. M. Crowe, C. C. Murphy, and W. P. Marnane,“Optimisation of the SHA-2 family of hash functions on FPGAs,” in Proc. IEEE Symp. Emerg. VLSI Technol. Archit. (ISVLSI), Mar. 2006, pp. 1–6.

[6] [6] R. Chaves, G. Kuzmanov, L. Sousa, and S. Vassiliadis, “Cost-efficient SHA hardware accelerators,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 16, no. 8, pp. 999–1008, Jul. 2008.

[7] [7] M. D. Rote, N. Vijendran, and D. Selvakumar, “High performance SHA- 2 core using the round pipelined technique,” in Proc. IEEE Int. Conf.Electron., Computing, Commun. Technol. (CONECCT), Jul. 2015, pp. 1–6.

[8] [8] A. Abbas, R. Vo_, L. Wienbrandt, and M. Schimmler, “An efficient implementation of PBKDF2 with RIPEMD-160 on multiple FPGAs,” in Proc. IEEE Int. Conf. Parallel Distrib. Syst. (ICPADS), Dec. 2014, pp. 454–461.

[9] [9] M. Kammerstetter, M. Muellner, D. Burian, C. Kudera, and W. Kastner, “Efficient high-speed WPA2 brute force attacks using scalable low-cost FPGA clustering,” in Proc. Int.

Conf. Cryptographic Hardware Embedded Syst. (CHES), Aug. 2016, pp. 559–577.

[10] [10] X. Li, C. Cao, P. Li, S. Shen, Y. Chen, and L. Li, "Energy-efficient hardware implementation of LUKS PBKDF2 with AES on FPGA," in Proc. IEEE Trustcom/BigDataSE/ISPA, Aug. 2016, pp. 402–409.

[11] [11] C. Wang, L. Gong, Q. Yu, X. Li, Y. Xie, and X. Zhou, "DLAU: A scalable deep learning accelerator unit on FPGA," IEEE Trans. Comput.- Aided Design Integr. Circuits Syst., vol. 36, no. 3, pp. 513–517, Mar. 2017.

[12] [12] W. Shi, X. Li, Z. Yu, and G. Overett, "An FPGA-based hardware accelerator for traffic sign detection," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 25, no. 4, pp. 1362–1372, Apr. 2017.

[13] [13] K. Malvoni, S. Designer, and J. Knezovic, "Are your passwords safe: Energy-efficient bcrypt cracking with low-cost parallel hardware," in Proc. 8th USENIX Workshop on Offensive Technologies (WOOT), 2014. [Online]. Available: https://www.usenix.org/conference/woot14/ workshop-program/presentation/malvani

[14] [14] "Advanced encryption standard (AES)," Nov. 2011. [Online]. Available: https://csrc.nist.gov/csrc/media/publications/fips/197/final/documents/fips-197.pdf

[15] [15] D. Eastlake and T. Hansen, "US secure hash algorithms SHA and SHA-based HMAC and HKDF," May 2011. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6234.txt

[16] [16] H. Dobbertin, A. Bosselaers, and B. Preneel, "RIPEMD-160: A strengthened version of RIPEMD," in Proc. Int. Workshop on Fast Software Encryption (FSE), 1996, pp. 71–82.

[17] [17] B. Ur, S. M. Segreti, L. Bauer, N. Christin, L. F. Cranor, S. Komanduri, D. Kurilova, M. L. Mazurek, W. Melicher, and R. Shay, "Measuring realworld accuracies and biases in modeling password guessability," in Proc. USENIX Secur. Symp., 2015, pp. 463–481.

[18] [18] J. Galbally, I. Coisel, and I. Sanchez, "A new multimodal approach for password strength estimation - part I: Theory and algorithms," IEEE Trans. Inf. Forensics Security, vol. 12, no. 12, pp. 2829–2844, Dec 2017.

[19] [19] B. Hitaj, P. Gasti, G. Ateniese, and F. P´erez-Cruz, "PassGAN: A deep learning approach for password guessing," 2017. [Online]. Available: http://arxiv.org/abs/1709.00440

[20] [20] "hashcat: Advanced password recovery," 2017. [Online]. Available: https://hashcat.net/hashcat/

[21] [21] OpenWall, "John the Ripper password cracker," 2017. [Online]. Available: http://www.openwall.com/john/

[22] [22] L. Bossuet, M. Grand, L. Gaspar, V. Fischer, and G. Gogniat, "Architectures of flexible symmetric key crypto engines a survey: From hardware coprocessor to multi-crypto-processor system on chip," ACM Computing Surveys, vol. 45, no. 4, p. 41, Aug. 2013.

[23] [23] "Zynq-7000 all programmable SoC," 2016. [Online]. Available: https://www.xilinx.com/support/documentation/product-briefs/ zynq-7000-product-brief.pdf

[24] [24] "Stratix 10 SoC: Highest performance and most power efficient processing," 2013. [Online]. Available: https://www.altera.com/products/ soc/portfolio/stratix-10-soc/overview.html

[25] R. F. Voss, J. Clarke. Algorithmic Musical Composition, Silver Burdett Press, Londyn, 1986.

[26] W. Zabierowski, A. Napieralski. Chords classification in tonal music, Journal of Environment Studies, Vol.10, No.5, 50-53.

[27] A. Abiewskiro,. Z. Moplskiiera. The Problem Of Grammar Choice For Verification, TCSET of the International Conference , House of Lviv Polytechnic National University, 19-23, 2008.

[28] Farquhar C, Protein and DNA Music, Online available from http://www.hrpub.org