A Performance Benchmark of Image Encryption Algorithms in RISC-V, ARM and x86 Architectures

Brendan John*, Naman Arora*

Abstract—Image encryption is becoming a necessary security measure for video streams for current technology, as they can contain sensitive information. Biometric used to identify an individual can be extracted, such as in eye tracking devices or webcams. Our work provides a comparative study to understand the latency introduced by image encryption algorithms using RISC-V, ARM, or x86 ISAs.

Index Terms—Image Encryption, Benchmark, ARM, RISC-V, Eye Tracking, Iris Biometric

1 Introduction

The goal of our project is to perform a comparative study between RISC-V, ARM, and x86 ISAs for the task of image encryption. Our experiment also benchmarks the performance of each ISA performing an image encryption task, which has not previously been performed for RISC-V. This task was selected as protecting the content of video streams is a growing research topic [22]. A compromised stream could contain personally identifiable information through a biometric, or reveal intellectual property. We focus on the case of encrypting images that could contain a user biometric, i.e., images containing the iris [9], eyebrow [10], and other regions of the face [8]. The outcome of our benchmark will be to evaluate the latency added to an eye tracking pipeline if an image encryption and decryption step were included. Certain applications, such as foveated rendering [27], cannot tolerate latency up to 60ms in worst case scenarios [2]. Other applications can tolerate higher latency values.

Our goal is to setup a testing environment where performance on encrypting and decrypting grayscale images from the eye camera of a video-based eye tracker [17] can be measured. These images contain the iris biometric, putting the user at risk [15], and are currently being integrated into the next generation of virtual reality (VR) and augmented reality (AR) devices. We plan to evaluate three different ISAs, RISC-V, ARM, and x86. Due to the low latency constraints of these eye tracking systems, we hypothesize that the architecture will play a role in performance, as measured in simulation or in hardware.

For our literature review we surveyed research papers using Google Scholar to identify a range of approaches used to encrypt images, such that we could select a secure method with the lowest expected runtime. We also searched for research papers and articles related to RISC-V and how it is impacting low-power embedded systems, such as those found in current IoT devices. We also identified tools and applications that would allows us to compare the performance of each ISA, including simulation and dynamic binary translation. These results will allow us to identify which tools, target hardware, and systems should be used for our comparative study and benchmark. Individual papers are briefly summarized in Section 2.

*These two authors contributed equally

- *Brendan John is a PhD student at the University of Florida. E-mail: brendanjohn@ufl.edu.*
- *Naman Arora is a Masters student at the University of Florida. E-mail: naman.arora@ufl.edu*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

2 Background

2.1 Image Encryption Algorithms

Towghi et al. [29] propose using a double phase encoding approach to encrypt an image. The image is converted to the Fourier domain, multiplied with a mask derived from one of the keys, then convolved using filter that is also based off a random number key. The implementation was found less sensitive to noise, and had lower error when reconstructing the image compared to amplitude based encryption. Runtimes for each algorithm were not provided in their evaluation.

Zhange and Karim [32] present a method to use double phase encoding for color images with an existing grayscale encryption system. The contribution of this work has more impact in the case of optical systems that implement encryption with a lens, instead of digital images captured using a camera, and thus computing performance is not discussed.

Regarding the security of the double random phase key algorithms, Carnicer et al. [3] exposed a chosen-cyphtertext attack for the encryption scheme. If an attacker can continuously run the decryption routine they will be able to input different cyphertexts and begin to determine bits of the shared encryption key. Even if the full key isn't estimated, the image can be reconstructed with a partial key, as missing certain bits only results in a low-pass filtering of the plaintext image. Thus, using these types of encryption schemes still place the plaintext image at risk.

Kachris et al. [16] present an logic-based processor that implements the SCAN encryption algorithm with the goal of real-time encryption, and is not concerned about decryption. SCAN is a block cipher with a block size of 64x64 pixels in this evaluation. A specialized FPGA is used to implement the operations with a frequency of 60 MHz. They reported performance results in terms of throughput, as they targeted video streaming applications. They achieved real-time performance (0.5 MB/s) for 1024x1024 gray scale images at three frames per second, and color images at one frame per second.

Salleh et al. [24] provide a 2-D chaotic map algorithm that is able to process images of any resolution using Baker's map. The mapping is used to generate random numbers that influence how an image is shuffled using permutations. Prior methods relied on a square image size for input. Chen et al. [6] implement a 3-D chaotic map algorithm, using a cat map. The performance results were good for the time, with encryption and decryption of a 512x512 grayscale image each taking one second on a single core 1GHz Intel Pentium IV.

Moon et al. [23] propose a fingerprint image based encryption protocol. The algorithm is optimized for fingerprint features, i.e. ridges, and achieves superb performance ranging from 380 ms to 730 ms depending on the fingerprint sensor used. The processor ran at 400MHz with 16MB of RAM and 2MB ROM. The algorithm is only applicable to fingerprint biometric images.

Machhout et al. [21] utilize Cellular Automata (CA) generated from logic comparisons based on 2-D key patterns. A processor was designed based on this algorithm, as it only performs logic comparisons and integer arithmetic, and achieved encryption times around 0.78 ms. The algorithm was not implemented in hardware for evaluation.

Torres-Huitzil [28] presented results from an FPGA implementation in hardware, achieving an encryption time of 560 microseconds for a 512x512 RGB image. These encryption algorithms are fast, using dedicated encryption hardware in their design and implementation.

Cheepchol and San-Um [5] present their own efficient encryption based on CA that is targeted towards images that contain the face biometric. The proposed solution is designed for RGB images and is implemented in MATLAB, however no runtime values were reported.

Huang et al. [13] discuss the security of a compressing sensing-based approach, where the image stream is encrypted and compressed at the same time while being streamed over a network. Traditional algorithms of this kind are susceptible to a chosen-plaintext attack. The authors propose a block cipher to further encrypt the data that is easily parallelized. A performance of 317ms is achieved using a 1.8 GHz Athlon dual-core processor with 2 GB of RAM to encrypt 512x512 grayscale images.

Chang et al. [4] implement image encryption and decryption in hardware, designing an FPGA optimal for applying a 32 bit key AES encryption. The hardware component is small in size, using only 110 slices. While low in area and optimal in throughput, this solution is not real-time, as encryption took 3.2 seconds and decryption took 5.1 seconds.

Mondal et al. [22] integrate a 2D Discrete Wavelet Transform (DWT) with Arnold's cat map by first permuting the image with the cat map, applying a wavelet transform, and then utilizing linear piece-wise chaotic mapping to generate the final encrypted image. Performance with an Intel 2.4GHz i5 dual-core Mp450 processor was 0.98 for a 512x512 image. The units for runtime were not provided, but were assumed to be in seconds. It also is not clear if this computation time included just encryption, or encryption then decryption.

Khalaf [18] proposes an efficient algorithm using random permutations using a DWT. Runtimes were presented without specifications for the hardware tested on, and without units for resulting values. If they are assumed to be in seconds, the encryption takes about 0.13 seconds and decryption about 0.11 seconds.

Hafsa et al. [12] present an FPGA design that can encrypt and decrypt images using AES with a 256 bit key. A single 50MHz NIOS II processor is used for implementation. Runtime was computed for a 256x256 grayscale image and came in at 4ms.

Lan et al. [19] build off the of the idea of chaotic mappings but proposing an integrated chaotic system (ICS) encryption. Additional chaos map structures are created using three basic 1D chaos maps and a series of operations, resulting in a more robust encryption than previously proposed methods. An average encryption time of 444ms was computed using Matlab code on an Intel i7-6700 running at 3.6GHz with 8GB of RAM.

Verma et al. [30] propose using a binary tree generating from one shared 32 bit key in an algorithm targeting real-time performance. The binary tree is unique for each key, and is then used to permute the pixels of the image, similar to other algorithms. Runtime for encryption on a 2.5GHz 7th generation Intel i5 with 8GB of RAM took 288ms.

Jain et al. [14] provide a survey of image encryption algorithms circa 2016. They compared state-of-the-art algorithms including the chaotic map approach along with established block and stream ciphers including DES, triple DES, AES, RC4, IDEA, and Visual Cryptography (VC). An Intel i5 2.2GHz processor was used to run Matlab code with grayscale images. The best performance was achieved by the Vigenére encryption (46 ms for 384x384 image), however the security of this particular method is ranked as poor or very poor in several categories. The algorithms that provided low runtimes while also preserving other security factors, such as keyspace security, are VC, Vigenère Cryptography, and RC4. The runtimes for VC and RC4 were 792ms and 570ms for a 384x384 image.

Summary Based on our survey of the literature the majority of algorithms either use a chaos map for image permutations, or a standard symmetric key approach like AES is used with reasonable results for our evaluation. It is worth noting that the lowest runtimes were recorded using dedicated hardware, which may not be installed in VR and AR devices in the near future. We chose to implement the approaches of Vigenère Cryptography [20], RC4 [11], and a Chirikov chaotic map algorithm [25], as they represent secure state-of-the-art algorithms that cover a range of approaches and would be simple and fast enough to include as part of an eye tracking pipeline. A stretch goal would be to implement the efficient binary tree based algorithm [30] as well.

2.2 IoT Architectures & Simulation

Shuja et al. [26] focus on ARM to x86 emulation techniques and benchmark to outline the overhead that comes with it. ARM was assumed as a standard for mobile and IoT devices as processor architecture. The bench marking strategy mainly utilized two open source emulators, Qemu and gem5. Through rigorous testing, it was concluded that ISA emulation from ARM to x86 can have very high system and application virtualization overhead.

Yi et al. [31] develop a embedded microprocessor based on ARMv4 architecture. ARM is highlighted as the most popular processor architecture. ARM is also presented as a super set of Reduced Instruction Set Computer.

The article [1] exhibits how the open RISC-V ISA for hardware industry is analogous to the open software revolution in the past decade. The pitfalls associated with the architecture design are analyzed keeping the IoT and mobile industry in perspective. Various cost associated with royalties and licensing are also taken into account which finally leads to a conclusion that RISC-V is the future of IoT processor architecture.

Clark et al. [7] present rv8, a RISC-V architecture simulation suit targeting x86 platform. Computationally intensive benchmarking was run to compare Qemu, the best emulator open source has to offer, with rv8 on various optimization levels. It was concluded from the said tests that dynamic binary translation was about 75% faster on rv8 when compared to Qemu.

Summary Based on our survey we choose to evaluate the RISC-V, ARM, and x86 processors. For evaluation of RISC-V, the tool rv8 [7] will be used on an x86 system (Dual-core Intel i5-7200U at 2.5GHz with 8GB of RAM) to measure runtime of the encryption and decryption codes.

3 Methodology

Generally our approach is to compute execution time of several image encryption (and decryption) routines. Several hardware platforms will be used, with several different ISAs tested using simulation. Our results will be tabulated to compare the performance of each ISA for each configuration and determine which performed best. We are also interested in the absolute runtime of these methods, to determine if they could be implemented in real-time for a stream of eye tracking data using the hardware configurations we selected.

3.1 Crypto Algorithms

3.1.1 RC4 Implementation

RC4 is simply applied to the 2D grayscale image by treating it as one long one-dimensional string that is XOR'd with the generated key. RC4 relies on two key mechanisms, the Key Scheduling Algorithm (KSA) and Pseudorandom Generation Algorithm (PRGA). These components are used for encryption and decryption as shown in Figure 1. The shared private key, K, is known by the sender and receiver, and through the use of KSA generate random bit sequence that encrypts the plain text through an XOR operation. Pseudo-code for the KSA and PRGA functions that will be implemented are provided by Jain et al. [14]. Decryption is performed analogous to encryption, except the XOR operation is applied byte by byte to the cipher text to generate the original plain text.

3.1.2 Vigenère Cryptography Implementation

Vigenère cryptography is based on alphabetic substitution [14], not unlike that originally implemented by Caesar or ROT13 ciphers. However, dealing with images we instead substitute numbers in the range 0 to 255. Given a key, K with B bytes, we treat this as B different numbers.

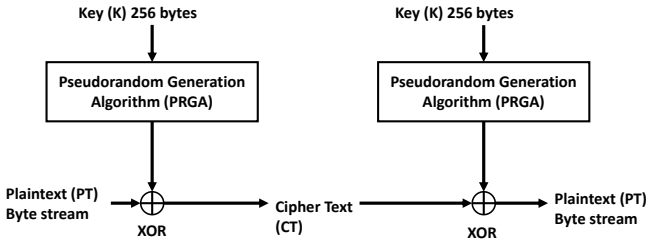

Fig. 1: Block diagram for RC4 encryption.

Each number, K_i where i is one of B bytes, is used to substitute the value for one pixel, as shown in the equation:

$$Pixel_{cipher} = (Pixel_{plain} + K_i) mod 255 \quad (1)$$

After all B bytes have been used for the first B pixels, we circle back to the start of the key and apply each number again to the remaining of the pixels in the image. Decryption is performed by modular subtraction to the cipher image pixels with the same key. This method is quite efficient, as it requires only a modular addition (or subtraction), but the key is sensitive to attack.

3.1.3 Chirikov Chaos Map Implementation

Image encryption with a chaos map is performed using an arbitrary length key, K, whose decimal value will be used during the mapping process. For our implementation the Chirikov chaos map will be used [25]. For each pixel in the input image at row i and column j, we map the grayscale value into a new location using the following equations:

$$\begin{aligned} i' &= (i - 1 + j - 1) mod N \\ j' &= \left\lfloor j - 1 + K * sin(\tfrac{2\pi i'}{N}) \right\rfloor mod N, \end{aligned} \quad (2)$$

where N is the height of the image. Using a nested for loop over the image pixels this process randomizes the pixel values, which are placed into their new row i' and column j'. Decryption performs an inverse mapping using K. Note that more complex algorithms have been proposed that utilize information diffusion and multiple mappings with different functions to further enhance security, however we chose to implement a simple, yet still secure, version with one mapping to serve as baseline performance for the chaos map-based approaches.

3.2 Security Evaluation Metrics

While we have some understanding of the security of each method is from past work, we will still use two metrics to evaluate how secure the encryption of each algorithm is. The most common metrics are to compare histograms of all grayscale values in the plain text image next to that of the cypher image, and to compute correlation between adjacent pixels [14]. The plain text will vary based on input, however the cypher text should vary significantly from the input as well. Histograms show that the distribution of values did in fact change, and correlation shows that patterns within locality of the input image are no longer present in the cypher text.

3.3 Testing Apparatus

3.3.1 Testing Input

Our focus for evaluation will be with grayscale images, where one byte is allocated for each pixel providing a range of values from 0 to 255. The most common image resolution for the eye tracking use case is 320x240 [15], however we will also compute performance for images that are 640x480, and 800x600. These are larger than most currently deployed systems, but could be present in future iterations of eye tracking technology as camera sensors become more efficient. The selected crypto algorithms do not depend on the image content, so one test image will be used as captured by a Pupil Labs Pro eye tracker [17] configured for each resolution.

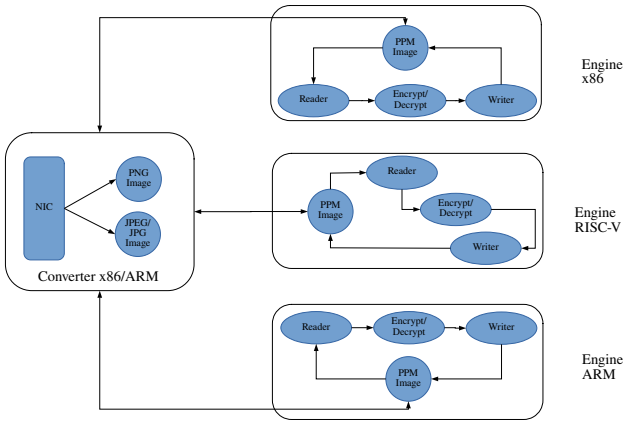

Fig. 2: Flow diagram for testing apparatus used to measure performance of a simulated ISA with C code for image encryption and decryption. Note that images are converted to .ppm format via the native instruction set as part of the rv8 interface, and then are forwarded to encryption and decryption components that are being executed through the simulated instruction set, isolating the performance of each ISA.

3.3.2 Hardware Configurations

For our performance evaluation we will compare execution times for encryption and decryption. As we focus on the impact of ISA on performance, we will evaluate three hardware platforms:

1. ARM - Raspberry PI Model B+ (BCM2837B0, 1.4GHz).
2. x86 - Dell Inspiron Laptop 3000 (i5-7200U, 2.5GHz)
3. x86 - Lenovo Legion Y530 (i7-8750H, 2.2GHz)

The goal is to cover a range of hardware specs, from low power (Raspberry Pi), mid-range (i5-7200U), and high power (i7-8750H). Docker (version 19.03) is used to set up the environment for simulating over these configurations. The RISC-V toolchain and RV8 [7] simulator are configured once and converted to a portable image to provide a standardized environment. RV8 is only compatabile with Linux, MacOS, and OpenBSD, therefore we will use a Linux distrubution, such as ArchLinux[1], to implement our evaluation.

RV8 provides a way to compile C code for encryption and decryption in simulation, however limits the use of non-standard libraries, such as OpenCV. To accommodate this in our image based evaluation, we propose a pipeline that deals with image conversion and data transfer that runs on the native ISA, but executes encryption and decryption on the simulated ISA.

3.3.3 Instrumentation

Figure 2 outlines our general approach for simulating the different ISAs we will explore (x86,RISC-V,ARM). RV8 simulation allows us to compile and execute C code, but limits the libraries used. This includes image I/O, and working with different standard image formats like JPG and PNG. Instead, we note that we can convert images to a uncompressed format that can be used without such libraries, such as PPM. Generally, each byte of the grayscale image is stored sequentially, and during encryption or decryption can be processed pixel by pixel. This provides us the advantage of isolating our encryption and decryption routines and computing execution time for each ISA. Execution time can also be measured for the converter code, but we note that in our

[1] https://www.archlinux.org/

proposed use case the eye tracking device encryption could be applied directly to sensor data, which would not be compressed into a particular format, but already loaded into memory.

4 Results

5 Conclusion

Acknowledgments

TODO

References

[1] Open source risc-v architecture is changing the game for iot processors. https://www.embedded-computing.com/guest-blogs/open-source-risc-v-architecture-is-changing-the-game-for-iot-processors. Accessed: 2019-09-22.

[2] E. Arabadzhiyska, O. T. Tursun, K. Myszkowski, H.-P. Seidel, and P. Didyk. Saccade landing position prediction for gaze-contingent rendering. *ACM Transactions on Graphics (TOG)*, 36(4):50, 2017.

[3] A. Carnicer, M. Montes-Usategui, S. Arcos, and I. Juvells. Vulnerability to chosen-cyphertext attacks of optical encryption schemes based on double random phase keys. *Optics letters*, 30(13):1644–1646, 2005.

[4] K.-H. Chang, Y.-C. Chen, C.-C. Hsieh, C.-W. Huang, and C.-J. Chang. Embedded a low area 32-bit aes for image encryption/decryption application. In *2009 IEEE International Symposium on Circuits and Systems*, pp. 1922–1925. IEEE, 2009.

[5] S. Cheepchol, W. San-Um, S. Kiattisin, and A. Leelasantitham. Digital biometric facial image encryption using chaotic cellular automata for secure image storages. In *The 4th Joint International Conference on Information and Communication Technology, Electronic and Electrical Engineering (JICTEE)*, pp. 1–5. IEEE, 2014.

[6] G. Chen, Y. Mao, and C. K. Chui. A symmetric image encryption scheme based on 3d chaotic cat maps. *Chaos, Solitons & Fractals*, 21(3):749–761, 2004.

[7] M. Clark and B. Hoult. rv8: a high performance risc-v to x86 binary translator. In *First Workshop on Computer Architecture Research with RISC-V (CARRV). Boston, MA, USA*, 2017.

[8] J. Daugman. Face and gesture recognition: Overview. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (7):675–676, 1997.

[9] J. Daugman. How iris recognition works. In *The essential guide to image processing*, pp. 715–739. Elsevier, 2009.

[10] Y. Dong and D. L. Woodard. Eyebrow shape-based features for biometric recognition and gender classification: A feasibility study. In *2011 International Joint Conference on Biometrics (IJCB)*, pp. 1–8. IEEE, 2011.

[11] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the key scheduling algorithm of rc4. In *International Workshop on Selected Areas in Cryptography*, pp. 1–24. Springer, 2001.

[12] A. Hafsa, A. Sghaier, W. E. Yousef, M. Machhout, and J. Malek. Image encryption/decryption design using niosii soft core processor. In *2017 International Conference on Engineering & MIS (ICEMIS)*, pp. 1–5. IEEE, 2017.

[13] R. Huang, K. Rhee, and S. Uchida. A parallel image encryption method based on compressive sensing. *Multimedia tools and applications*, 72(1):71–93, 2014.

[14] Y. Jain, R. Bansal, G. Sharma, B. Kumar, and S. Gupta. Image encryption schemes: a complete survey. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 9(7):157–192, 2016.

[15] B. John, S. Koppal, and E. Jain. Eyeveil: degrading iris authentication in eye tracking headsets. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*, p. 37. ACM, 2019.

[16] C. Kachris, N. Bourbakis, and A. Dollas. A reconfigurable logic-based processor for the scan image and video encryption algorithm. *International Journal of Parallel Programming*, 31(6):489–506, 2003.

[17] M. Kassner, W. Patera, and A. Bulling. Pupil: an open source platform for pervasive eye tracking and mobile gaze-based interaction. In *Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing: Adjunct publication*, pp. 1151–1160. ACM, 2014.

[18] A. D. Khalaf. Fast image encryption based on random image key. *International Journal of Computer Applications*, 975:8887, 2016.

[19] R. Lan, J. He, S. Wang, T. Gu, and X. Luo. Integrated chaotic systems for image encryption. *Signal Processing*, 147:133–145, 2018.

[20] S. Li and Y. Zhao. Image scrambling based on chaos theory and vigenère cipher. In *2011 Seventh International Conference on Computational Intelligence and Security*, pp. 555–558. IEEE, 2011.

[21] M. Mohsen, G. Zied, Z. Medien, and T. Rached. Design of reconfigurable image encryption processor using 2-d cellular automata generator. *International Journal of Computer Science and Applications*, 6(4):43–62, 2009.

[22] B. Mondal, T. Mandal, D. A. Khan, and T. Choudhury. A secure image encryption scheme using chaos and wavelet transformations. *Recent Patents on Engineering*, 12(1):5–14, 2018.

[23] D. Moon, Y. Chung, S. B. Pan, K. Moon, and K. I. Chung. An efficient selective encryption of fingerprint images for embedded processors. *ETRI journal*, 28(4):444–452, 2006.

[24] M. Salleh, S. Ibrahim, and I. F. Isnin. Enhanced chaotic image encryption algorithm based on baker's map. In *Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS'03.*, vol. 2, pp. II–II. IEEE, 2003.

[25] N. Sethi. A new image encryption method using chirikov and logistic map. *International Journal of Computer Applications*, 59(3), 2012.

[26] J. Shuja, A. Gani, A. Naveed, E. Ahmed, and C.-H. Hsu. Case of arm emulation optimization for offloading mechanisms in mobile cloud computing. *Future Generation Computer Systems*, 76:407–417, 2017.

[27] Q. Sun, A. Patney, L.-Y. Wei, O. Shapira, J. Lu, P. Asente, S. Zhu, M. Mcguire, D. Luebke, and A. Kaufman. Towards virtual reality infinite walking: dynamic saccadic redirection. *ACM Transactions on Graphics (TOG)*, 37(4):67, 2018.

[28] C. Torres-Huitzil. Hardware realization of a lightweight 2d cellular automata-based cipher for image encryption. In *2013 IEEE 4th Latin American Symposium on Circuits and Systems (LASCAS)*, pp. 1–4. IEEE, 2013.

[29] N. Towghi, B. Javidi, and Z. Luo. Fully phase encrypted image processor. *JOSA A*, 16(8):1915–1927, 1999.

[30] A. Verma, P. Gupta, and M. Deshmukh. An efficient encryption technique for images using symmetric key cryptography and binary trees. In *Proceedings of 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT)*, pp. 26–27, 2018.

[31] Q.-M. Yi, M. Shi, M.-M. Chen, and G. Wang. Research and design of embedded microprocessor based on arm architecture. In *2016 13th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, pp. 463–467. IEEE, 2016.

[32] S. Zhang and M. A. Karim. Color image encryption using double random phase encoding. *Microwave and optical technology letters*, 21(5):318–323, 1999.