A Review of Multi-Core Implementation of Secure Hash Function Algorithm

[1]Amit Patel, [2]Dr. Bharti Chourasia

[1]M.Tech Scholar, [2]Professor & HOD
[1]Department of Electronics & Communication,
[1]RKDF Institute of Science & Technology Bhopal, India.

Abstract : A cryptographic hash work is a phenomenal class of hash work that has certain properties which make it fitting for use in cryptography. It is a numerical figuring that maps information of emotional size to a bit string of a settled size (a hash) and is expected to be a confined limit, that is, a limit which is infeasible to adjust. Hash Functions are significant instrument in information security over the web. The hash functions that are utilized in different security related applications are called cryptographic hash functions. This property is additionally valuable in numerous different applications, for example, production of digital signature and arbitrary number age and so on. The vast majority of the hash functions depend on Merkle-Damgard development, for example, MD-2, MD-4, MD-5, SHA-1, SHA-2, SHA-3 and so on, which are not hundred percent safe from assaults. The paper talks about a portion of the secure hash function, that are conceivable on this development, and accordingly on these hash functions additionally face same attacks.

IndexTerms – Secure, function, MD-2, MD-4, MD-5, SHA-1, SHA-2, SHA-3.

I. INTRODUCTION

A cryptographic hash function is an extraordinary class of hash function that has certain properties which make it reasonable for use in cryptography. It is a scientific algorithm that maps information of self-assertive size to a bit string of a fixed size (a hash) and is intended to be a single direction function, that is, a function which is infeasible to reverse. The best way to reproduce the information from a perfect cryptographic hash function's yield is to endeavor an animal power search of potential contributions to check whether they produce a match, or utilize a rainbow table of coordinated hashes.

The perfect cryptographic hash function has five primary properties:

- It is deterministic so a similar message dependably results in a similar hash
- It rushes to figure the hash an incentive for some random message
- It is infeasible to create a message from its hash an incentive aside from by attempting every single imaginable message
- A little change to a message should change the hash esteem so broadly that the new hash worth seems uncorrelated with the old hash esteem.
- It is infeasible to discover two distinct messages with similar hash esteem.

Cryptographic hash functions have numerous information-security applications, remarkably in digital signatures, message confirmation codes (Macintoshes), and different types of validation. They can likewise be utilized as common hash functions, to list information in hash tables, for fingerprinting, to distinguish copy information or particularly recognize documents, and as checksums to identify unintentional information defilement. For sure, in information-security settings, cryptographic hash esteems are in some cases called (digital) fingerprints, checksums, or simply hash esteems, despite the fact that every one of these terms represent progressively broad functions with rather various properties and purposes.

SHA-3 (Secure Hash Algorithm 3) is the most recent individual from the Secure Hash Algorithm group of models, discharged by NIST on August 5, 2015. Albeit part of a similar arrangement of benchmarks, SHA-3 is inside not the same as the MD5-like structure of SHA-1 and SHA-2. SHA-3 is a subset of the more extensive cryptographic crude family Keccak structured by Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche, expanding upon RadioGatún. Keccak's creators have proposed extra uses for the function, not (yet) institutionalized by NIST, including a stream figure, a confirmed encryption framework, a "tree" hashing plan for quicker hashing on certain architectures,and AEAD figures Keyak and Ketje. NIST does not at present intend to pull back SHA-2 or expel it from the amended Secure Hash Standard. The motivation behind SHA-3 is that it very well may be directly substituted for SHA-2 in current applications if important, and to altogether improve the strength of NIST's general hash algorithm toolbox. SHA-3 utilizes the wipe development, wherein information is "assimilated" into the wipe, at that point the outcome is "squeezed" out.

II. LITERATURE REVIEW

A. Alzahrani et al.,[2018] Embedded multi-core systems are implemented as systems-on-chip that rely on packet storeand-forward networks-on-chip for communications. These systems do not use buses or global clock. Instead routers are used to move data between the cores, and each core uses its own local clock. This implies concurrent asynchronous computing. Implementing algorithms in such systems is very much facilitated using dataflow concepts. In this work, it is propose a methodology for implementing algorithms on dataflow platforms. The methodology can be applied to multi-threaded, multi-core platforms or a combination of these platforms as well. This methodology is based on a novel dataflow graph representation of the algorithm. it is applied the proposed methodology to obtain a novel dataflow multi-core computing model for the secure hash algorithm-3. The resulting hardware was implemented in field-programmable gate array to verify the performance parameters. The proposed model

of computation has advantages, such as flexible I/O timing in term of scheduling policy, execution of tasks as soon as possible, and self-timed event driven system. In other words, I/O timing and correctness of algorithm evaluation are dissociated in this paper. The main advantage of this proposal is ability to dynamically obfuscate algorithm evaluation to thwart side-channel attacks without having to redesign the system. This has important implications for cryptographic applications. [1]

A. Aghaie et al.,[2018] Achieving different security properties through lightweight cryptography has been the focus of recent research efforts. A wide range of criteria should be taken into account when designing lightweight ciphers, such as both the linear and non-linear properties, and resilience to active side-channel attacks (SCA), e.g., fault analysis attacks mounted on VLSI implementations. This work motivates the urgency and highlights the importance of considering reliability and error detection as a design factor beforehand rather than an afterthought. In this work and through case studies, it is motivate the urgency of "design-for low-cost reliability and fault diagnosis" for future work to thwart fault analysis attacks "before" the algorithms are designed. In this work, it is focus on the design criteria of S-boxes transformations (we note that for different ciphers, these transformations might be denoted by other yet similar terms), which provide diffusion and confusion properties in block ciphers, stream ciphers, and hash functions. it is also present the results of our implementations for a set of S-boxes used in various block ciphers benchmarked on application-specific integrated circuit (ASIC). The proposed assessments for reliability and error detection of lightweight ciphers is a step forward towards design for error detection while achieving secure implementations. [2]

X. Qiuyun, et al.,[2017] In this work, UVM (Universal Verification Methodology) is adopted to build the SHA-256 IP verification platform. The generic code of the verification platform is automatically generated by the Perl script. That is the semi-automatic UVM platform. It forms the verification structure of the module level and the top level. Based on the platform, the remaining of the core code is added to the testbench. Then according to the verification scheme, a large number of test cases produce the stimulus to the DUT, which make the functional coverage achieve 100%, the code coverage and the waveform meet the requirements. The results show the SHA-256 IP design successfully and the semi-automatic UVM platform is usable. [3]

I. A. Landge et al.,[2016] Many modern electronic systems like PDAs, cell phones, network routers, smart cards, and networked sensors are in need to access, store, manipulate, or communicate sensitive information, making security a serious concern in their design. Embedded and VLSI systems, which account for a wide range of products from the electronics, semiconductor, telecommunications, and networking industries, requires some of the most demanding security concerns. Embedded and VLSI based systems are mainly resource constrained and frequently need to operate in physically insecure environments. Whenever personal or confidential data is processed or stored security objectives like authenticity and integrity have to be fulfilled. This security objective must be ensured by the application or underlying hardware. Hardware implementation aspects of the MD5 hash algorithm using VHDL are discussed in this paper. [4]

S. Koranne, et al.,[2015] Modern very large-scale integration (VLSI) layout databases routinely consist of 10e15 edges, and thus problems of information retrieval, intellectual property (IP) inventory control, tampering detection, IP infringement detection, data tagging, and database version control, are extremely computationally intensive. All these tasks can be reduced to the problem of copy detection, and in this work, it is propose a canonical hash function for VLSI layout datasets which can be used for efficient copy detection and signature generation. The proposed signature is independent of the ordering of the layout elements, their tessellation, resolution, and even vertex count. These parameters, which do not contribute to the final wafer image, increase the entropy of the data and thus standard hash functions such as message digest (MD5) or secure hash algorithm (SHA), are not suitable for this problem of VLSI layout hashing. In this work, a novel, entropy reduced hash function is developed which can be used to alleviate the above mentioned problems of physical IP management. The proposed method has O(n logn + k) time complexity, and O(√n) memory complexity, where n is the number of edges in the input layout, and k is the number of intersections between edges. The proposed system has been implemented, and computational results validating our approach are also provided. [5]

B. Alomair et al.,[2014]In cryptography, secure channels enable the confidential and authenticated message exchange between authorized users. A generic approach of constructing such channels is by combining an encryption primitive with an authentication primitive MAC. In this work, it is introduce the design of a new cryptographic primitive to be used in the construction of secure channels. Instead of using general purpose MACs, it is propose the deployment of special purpose MACs, named ε-MACs. The main motivation behind this work is the observation that, since the message must be both encrypted and authenticated, there might be some redundancy in the computations performed by the two primitives. Therefore, removing such redundancy can improve the efficiency of the overall composition. Moreover, computations performed by the encryption algorithm can be further utilized to improve the security of the authentication algorithm. In particular, it is will show how ε-MACs can be designed to reduce the amount of computation required by standard MACs based on universal hash functions, and show how ε-MACs can be secured against key-recovery attacks.[6]

M. Zhang, et al.,[2013] Sensor networks are frequently deployed in physically insecure environments and capture sensitive data, making security a paramount challenge. Cryptographic techniques, such as encryption and hashing, are useful in addressing these concerns. However, the use of these schemes greatly increases the energy consumption of sensor nodes and thus shortens their lifetime. To address this challenge, it is propose encompression (encryption + compression) as a strategy to achieve low-energy

secure data transmission in sensor networks. Our proposal combines, for the first time, compressive sensing (CS), a powerful and general approach for exploiting sparsity of sensor data, with encryption and integrity checking of the compressively sensed data. While encompression can be realized using any compression technique, CS is particularly well suited since it can be realized with a very low computational and energy footprint that is compatible with the constraints of sensor nodes. it is present an evaluation of a hardware implementation of encompression, wherein the CS, encryption, and integrity checking algorithms are realized using a 65-nm CMOS technology. it is also present a system-level evaluation of encompression by realizing it in software on a commercial embedded sensor platform and measuring the energy consumption.[7]

I. Algredo-Badillo, et al.,[2012] In order to design efficient hardware implementations of cryptographic algorithms for a particular application, it is often required to explore several architectures in order to select the one that offers the appropriate trade-off between throughput and hardware resources. A natural choice for performing a design space exploration are the Field Programmable Gate Arrays (FPGAs) for being reconfigurable, flexible and physically secure devices. In this work it is explore several architectures for implementing the SHA-512 algorithm based on the loop unrolling technique and analyze their area-performance trade-offs. The analysis consists on unrolling at different levels the main loop which is the most costly part in the SHA-512 algorithm. The resulting hardware architectures are implemented and analyzed in order to identify the critical path and make decisions on the architectural design. The obtained results provide a practical guide to understand the effect of introducing different levels (1, 2, 4, 5, 8) of unrolling in terms of throughput and hardware resources. The hardware architecture 4x that partially unrolls four iterations of the main loop of the SHA-512 algorithm reports the best performance compared against related works, while the 1x architecture exhibits the best efficiency. [8]

Table 1: Summary of literature survey

Sr No	Author Name	Publish Details	Proposed Work	Outcome
1	A. Alzahrani	IEEE, 2018	A methodology for implementing algorithms on dataflow platforms	Dynamically obfuscate algorithm evaluation to thwart side-channel attacks.
2	A. Aghaie	IEEE, 2018	Considering reliability and error detection as a design factor beforehand	A set of S-boxes used in various block ciphers benchmarked on ASIC.
3	X. Qiuyun	IEEE, 2017	UVM is adopted to build the SHA-256 IP verification platform.	The SHA-256 IP design successfully and the semi-automatic UVM platform.
4	I. A. Landge	IEEE, 2016	Embedded and VLSI systems, which account for a wide range of products from the electronics	Hardware implementation aspects of the MD5 hash algorithm using VHDL are discussed .
5	S. Koranne,	IEEE, 2015	A novel, entropy reduced hash function is developed alleviate the above mentioned problems of physical IP management	Proposed System has been implemented, and computational results validating our approach are also provided.
6	B. Alomair	IEEE, 2014	It is introduce the design of a new cryptographic primitive.	Amount of computation required by standard MACs based on universal hash functions.
7	M. Zhang,	IEEE, 2013	Decompression (encryption + compression) low-energy secure data transmission in sensor networks.	CS may be a game changer in enabling state-of-the-art cryptography to be employed in highly energy-constrained sensor networks.
8	I. Algredo-Badillo	IEEE, 2012	The SHA-512 algorithm based on the loop unrolling technique and analyze.	Hardware architectures are implemented and analyzed in order to identify the critical path and make decisions.

III. TYPES OF CRYPTOGRAPHIC HASH ALGORITHMS

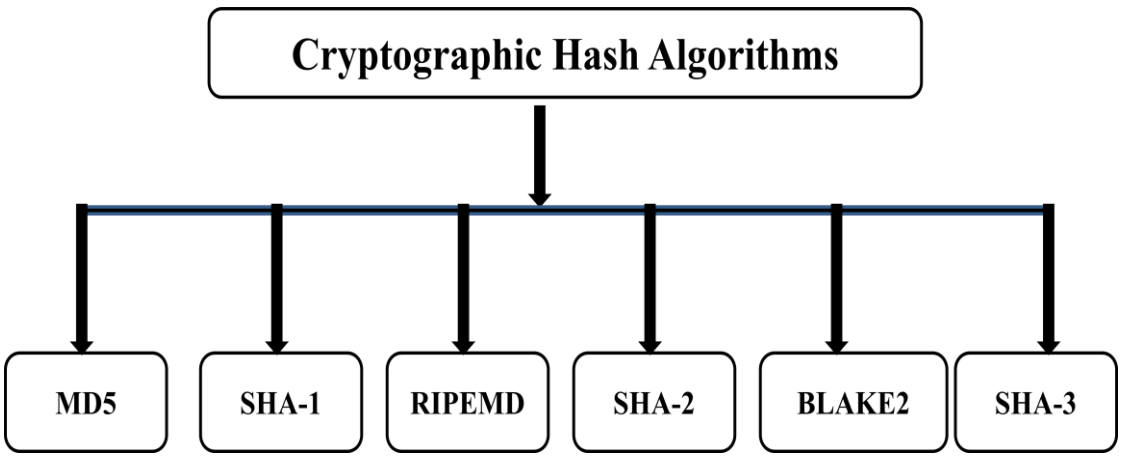

Figure 1: Types of crypto hash function

Hashing methods are categorized into two groups:

1. Data-oriented hashing versus security-oriented hashing

(i) Data-Oriented Hashing Data-oriented hashing refers to methods techniques that expect to utilize hashing to accelerate information recovery or examination, where a hash table is regularly kept up for an inquiry.

(ii)Security-Oriented Hashing Security-oriented hashing refers to methods that utilization hashing for confirmation or approval. For instance, a client may download programming from an open web server yet is stressed whether the product hosts been changed by a third gathering.

2. SHA-3

In SHA-3, the state S comprises of a 5 × 5 exhibit of w-bit words (with w=64), b = 5 × 5 × w = 5 × 5 × 64 = 1600 bits all out. Keccak is likewise characterized for littler intensity of-2 word sizes wdown to 1 bit (complete condition of 25 bits). Little state sizes can be utilized to test cryptanalytic assaults, and middle of the road state sizes (from w = 8, 200 bits, to w = 32, 800 bits) can be utilized in down to earth, lightweight applications.

For SHA-3-224, SHA-3-256, SHA-3-384, and SHA-3-512 occurrences, r is more noteworthy than d, so there is no requirement for extra square stages in the pressing stage; the main dbits of the state are the ideal hash. Be that as it may, SHAKE-128 and SHAKE-256 permit a discretionary yield length, which is valuable in applications, for example, ideal lopsided encryption cushioning.

3. MD5

MD5 was planned by Ronald Rivest in 1991 to supplant a previous hash function MD4, and was determined in 1992 as RFC 1321. Impacts against MD5 can be determined inside seconds which makes the algorithm unsatisfactory for most use situations where a cryptographic hash is required. MD5 produces a review of 128 bits (16 bytes).

4. SHA-1

SHA-1 was created as a major aspect of the U.S. Government's Capstone venture. The first determination - presently normally called SHA-0 - of the algorithm was distributed in 1993 under the title Secure Hash Standard, FIPS Bar 180, by U.S. government benchmarks organization NIST (National Foundation of Norms and Innovation). It was pulled back by the NSA not long after production and was supplanted by the changed form, distributed in 1995 in FIPS Bar 180-1 and usually assigned SHA-1. Impacts against the full SHA-1 algorithm can be delivered utilizing the shattered assault and the hash function ought to be viewed as broken. SHA-1 creates a hash condensation of 160 bits (20 bytes).

5. RIPEMD-160

RIPEMD (RACE Uprightness Natives Assessment Message Overview) is a group of cryptographic hash functions created in Leuven, Belgium, by Hans Dobbertin, Antoon Bosselaers and Bart Preneel at the COSIC research bunch at the Katholieke Universiteit Leuven, and first distributed in 1996. RIPEMD depended on the plan standards utilized in MD4, and is comparable in execution to the more famous SHA-1. RIPEMD-160 has anyway not been broken. As the name suggests, RIPEMD-160 produces a hash summary of 160 bits (20 bytes).

6. SHA-2

SHA-2 (Secure Hash Algorithm 2) is a lot of cryptographic hash functions planned by the US National Security Office (NSA), first distributed in 2001.They are manufactured utilizing the Merkle–Damgård structure, from a single direction pressure function itself fabricated utilizing the Davies–Meyer structure from a (characterized) specific square figure. SHA-2 essentially comprises of two hash algorithms: SHA-256 and SHA-512. SHA-224 is a variation of SHA-256 with various beginning qualities and truncated yield. SHA-384 and the lesser known SHA-512/224 and SHA-512/256 are on the whole variations of SHA-512. SHA-512 is more secure than SHA-256 and is ordinarily quicker than SHA-256 on 64 bit machines, for example, AMD64. The yield estimate in bits is given by the expansion to the "SHA" name, so SHA-224 has a yield size of 224 bits (28 bytes), SHA-256 produces 32 bytes, SHA-384 produces 48 bytes lastly SHA-512 produces 64 bytes.

7. BLAKE2

An improved variant of BLAKE called BLAKE2 was declared in December 21, 2012. It was made by Jean-Philippe Aumasson, Samuel Neves, Zooko Wilcox-O'Hearn, and Christian Winnerlein with the objective to supplant generally utilized, yet broken MD5 and SHA-1 algorithms. At the point when kept running on 64-bit x64 and ARM models, BLAKE2b is quicker than SHA-3, SHA-2, SHA-1, and MD5. In spite of the fact that BLAKE nor BLAKE2 have not been institutionalized as SHA-3 it has been utilized in numerous conventions including the Argon2 secret key hash for the high effectiveness that it offers on current CPUs. As BLAKE was a contender for SHA-3, BLAKE and BLAKE2 both offer a similar yield sizes as SHA-3 - including a configurable yield measure.

8. HASH vs AES

SHA represents Secure Hash Algorithm while AES represents Propelled Encryption Standard. So SHA is a suite of hashing algorithms. AES then again is a figure which is utilized to scramble. SHA algorithms (SHA-1, SHA-256 etc...) will take an info and produce a summary (hash), this is regularly utilized in a digital marking process (produce a hash of certain bytes and sign with a private key). SHA is a hash function and AES is an encryption standard. Given an information you can utilize SHA to deliver a yield which is in all respects probably not going to be created from some other information.

SHA and AES fill various needs. SHA is utilized to produce a hash of information and AES is utilized to scramble information. Here's a case of when a SHA hash is valuable to you. Let's assume you needed to download a DVD ISO picture of some Linux distro. This is a huge document and now and again things turn out badly - so you need to approve that what you downloaded is right. What you would do is go to a confided in source, (for example, the offical distro download point) and they ordinarily have the SHA hash for the ISO picture accessible. SHA has was utilized to approve information that was not ruined. AES, then again,

is utilized to scramble information, or keep individuals from survey that information with knowing some mystery. AES utilizes a shared key which implies that a similar key (or a related key) is utilized to encoded the information as is utilized to unscramble the information. For instance in the event that I scrambled an email utilizing AES and I sent that email to you then you and I would both need to realize the shared key used to encode and decode the email.

.

IV. CONCLUSION

There are many algorithm developed for security enhancement. Each algorithm have own strength and weakness. Except SHA approach all algorithm have encryption and decryption process on the other hand security method can be decoded by any menas. SHA is only approach which cannot be reverse through any mathematical function or reverse algorithm. In this survey talk about various secure cryptographic algorithm. Here included MD, SHA-1, SHA-2, SHA-3 and so on and discover SHA-3 is most recent planned algorithm which is increasingly reasonable and helpful for secure message in web applications. Various researchers have proposed their own algorithms anyway none of them are time gainful as SHA-3 and besides there are chances of upgrading the internal nature of these algorithms.

REFERENCE

1. A. Alzahrani and F. Gebali, "Multi-Core Dataflow Design and Implementation of Secure Hash Algorithm-3," in *IEEE Access*, vol. 6, pp. 6092-6102, 2018.
2. A. Aghaie, M. M. Kermani and R. Azarderakhsh, "Design-for-Error-Detection in Implementations of Cryptographic Nonlinear Substitution Boxes Benchmarked on ASIC," *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, Windsor, ON, Canada, 2018, pp. 574-577.
3. X. Qiuyun, H. Ligang, L. Qiming, G. Shuqin and W. Jinhui, "The Verification of SHA-256 IP using a semi-automatic UVM platform," *2017 13th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)*, Yangzhou, 2017, pp. 111-115.
4. I. A. Landge and B. K. Mishra, "Hardware based MD5 implementation using VHDL for secured embedded and VLSI based designs," *2016 International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, 2016, pp. 1-6.
5. S. Koranne, "DÉJÀ VU: An Entropy Reduced Hash Function for VLSI Layout Databases," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 11, pp. 1798-1807, Nov. 2015.
6. B. Alomair and R. Poovendran, "E-MACs: Toward More Secure and More Efficient Constructions of Secure Channels," in *IEEE Transactions on Computers*, vol. 63, no. 1, pp. 204-217, Jan. 2014.
7. M. Zhang, M. M. Kermani, A. Raghunathan and N. K. Jha, "Energy-efficient and Secure Sensor Data Transmission Using Encompression," *2013 26th International Conference on VLSI Design and 2013 12th International Conference on Embedded Systems*, Pune, 2013, pp. 31-36.
8. I. Algredo-Badillo, M. Morales-Sandoval, C. Feregrino-Uribe and R. Cumplido, "Throughput and Efficiency Analysis of Unrolled Hardware Architectures for the SHA-512 Hash Algorithm," *2012 IEEE Computer Society Annual Symposium on VLSI*, Amherst, MA, 2012, pp. 63-68.
9. N. Sklavos, "Multi-module Hashing System for SHA-3 & FPGA Integration," *2011 21st International Conference on Field Programmable Logic and Applications*, Chania, 2011, pp. 162-166.
10. A. Shahmoradi and M. Masoumi, "A new nanoelectronic based approach for efficient VLSI realization of SHA-512 algorithm," *IEEE EUROCON 2009*, St.-Petersburg, 2009, pp. 1206-1213.
11. R. Chaves, G. Kuzmanov, L. Sousa and S. Vassiliadis, "Cost-Efficient SHA Hardware Accelerators," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, no. 8, pp. 999-1008, Aug. 2008.
12. Dan Cao, Jun Han and Xiao-yang Zeng, "A reconfigurable and ultra low-cost VLSI implementation of SHA-1 and MD5 functions," *2007 7th International Conference on ASIC*, Guilin, 2007, pp. 862-865.