THE INTERNATIONAL JOURNAL OF SCIENCE & TECHNOLEDGE

Design and FPGA Implementation of Reliable SHA-3 Algorithm

Tejaswini S.
M. Tech. Student, VLSI Design and Embedded System, SCE, VTU, Bangalore, India
Sudha M. S.
Assistant Professor, Department of ECE, SCE, VTU, Bangalore, India
B. N. Shobha
Associate Professor, Department of ECE, SCE, VTU, Bangalore, India

Abstract:
SHA-3 is one of the important cryptographic tools which is being used for system and information security. This cryptographic tool is used in assuring data integrity as changing one single bit in the input message can change about half of the output digest. The SHA-3 has been selected and will be used to provide security to applications requiring hashing, pseudo-random number generation and integrity checking. This algorithm has been chosen based on benchmarks such as security, performance and complexity. To provide reliable architectures for this algorithm an efficient concurrent error detection scheme is chosen for the proposed SHA-3 algorithm. The proposed error detection approach has less complexity and performance overheads while maintaining high error coverage. A low-complexity recomputing with RERO scheme reduces the hardware overhead of this error detection approach. This scheme is simulated on FPGA and that has less complexity and performance overheads. The development of high-performance concurrent error detection scheme is expected to provide more reliable and robust hardware implementation for the newly-standardized SHA-3 algorithm.

Keywords: *Secure hash algorithm 3, Image Conversion, Security, High performance, Field programmable gate array.*

1. Introduction

Data integrity and authenticity related security issues are very important in internet communications. SHA-3 is one of the important cryptographic tools which is being used for system and information security. This cryptographic tool is used in assuring data integrity as changing one single bit in the input message can change about half of the output digest. Authenticity and nonrepudiation is more efficiently achieved by using SHA-3 algorithm. The selection process for choosing a new secure cryptographic hash algorithm, i.e., SHA- 3, was initiated by the NIST to increase security and performance of hash functions.

There are some reasons for having error detection for such an important cryptographic tool. The digital circuits are prone to natural faults and this is the case for the hardware implementations of SHA-3 as well. Malicious attacks such as those based on fault injections can target the availability of this algorithm which, in turn, result in malfunctioning of the integrity-checking process. This can induce much overhead to the system, especially the constrained nodes such as those used in sensitive applications, e.g., security of implantable and wearable medical devices or constrained industrial setups. All these can cause the integrity check to fail and produce unreliable system. This unreliable system needs to restart or reinitialized several times. Such behaviour requires more energy consumption and results in higher cost. Concurrent error detection can help to detect the errors as much as possible and stop the system or process gracefully to resolve the issue.

The error detection approach based on the time-redundancy techniques, i.e., the RERO scheme has been used in this proposed method. This RERO-based approach reduces the hardware overhead of the original designs, suitable for light weight and low power implementations. Then, to evaluate the error detection capability of the proposed scheme in response to transient and permanent faults through the simulation. Through the simulations, the proposed scheme reaches high error coverage. Then the original SHA-3 and proposed error detection scheme are synthesized using a FPGA standard-cell library to obtain the area overheads and the performance metrics. The synthesis results will shows less overhead for the proposed technique, while achieving high error coverage.

2. Proposed Method

Cryptographic tool can be used in assuring data integrity, digital signature, military applications, medical, bank applications, companies, Radio frequency identification, Message authentication, Emerging Cryptographic Transformations and several resource constrained applications. All these applications need security, robust algorithm and reliable system. Earlier algorithms provides minimum security, performance and complexity. This SHA-3 algorithm provides high security but it produces unreliable system if

implemented on the hardware. Hence RERO based error detection time-redundancy scheme has been used that provides reliable system as compared to the previous scheme.

2.1. Block Diagram

In the Fig 1 shows the block diagram of the proposed method. Here the input is an image that can be of any size. This image is resized into 100*100 and converted into pixels, each pixel is 8 bits wide and it is passed to algorithm through the 8 bits to 64 bits converter. This 8 bits to 64 bits converter converts and gives 64 bits output and is fed to the algorithm. Then the algorithm generates 512 bits of hashed values according to inputs and fed to the 512 bits to 8 bit converter. This converter converts and gives 8 bits output and is stored in the text file form. This text form of output can be converted into image and displayed on the computer system by using MATLAB tool.

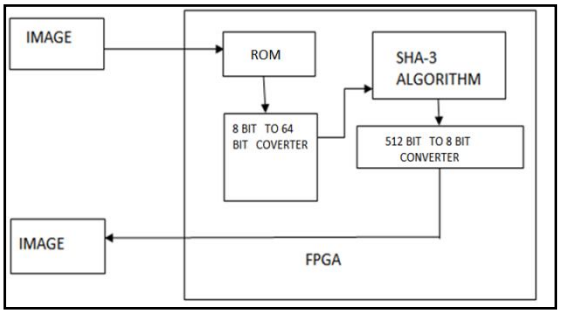

Figure 1: Block diagram of proposed scheme

2.1.1. Keccak Algorithm

The core of the Keccak algorithm is the permutation function which is repeatedly applied to a fixed-length state of b = r + c bits, where r and c are bit rate and capacity of the algorithm. Higher values of r improve the speed whereas higher values of c correspond to higher security level. The input message is first padded to get a length in multiples of r. Then, through five internal steps for each round, the absorbing phase is performed. Finally, the squeezing phase occurs in which the first r bits of the state are returned as the output block. There are seven possible types for Keccak and for the sake of brevity and focus on the recommended type, namely Keccak-f[r + c = 1600] (c = 1024 and r = 576).

In Keccak-f[1600], 1600 is the width of the underlying permutation. For this type of Keccak, the state consists of an array of 5 × 5 lanes, each of length w = 64 bits. The recommended number of rounds for Keccak-f[1600] is 24 [1].

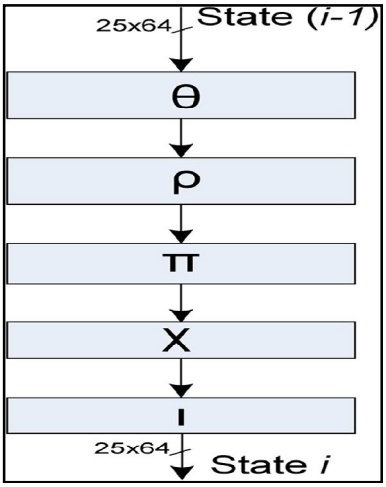

Figure 2: Five internal steps in the Keccak algorithm[12].

The five internal steps of each of the 24 rounds of Keccak-f[1600] is shown in Fig 2 where $0 \leq x, y \leq 4$. The first internal step is θ as follows:

$c[x] \leftarrow A[x,0] \oplus A[x,1] \oplus A[x,2] \oplus A[x,3] \oplus A[x,4]$ 1

$D[x] \leftarrow C[x-1] \oplus rot(C[x+1],1)$..... 2

$A[x,1] \leftarrow A[X,Y] \oplus D[X]Z$ 3

In equation 3, A[x, y] denotes the state A, C and D being internal states and Bitwise XOR, modulo-5 addition and subtraction operations are performed in this equation and it can be represented as □, + and −, respectively. Finally, rot(C[x + 1],1) is the bitwise cyclic shift operation, moving bit at position i into position i + 1modulo the lane size.

The next two steps are ρ and π as follows: $B[y, 2 \times x + 3 \times y] \leftarrow rot(A[x, y], r[x, y])$.The constants $r[x, y]$ are the rotation offsets. The second to last step is x, i.e.,$A[x, y] \leftarrow B[x, y] \oplus \sim(B[x + 1, y]).B[x + 2, y])$. in this equation ~, . ,⊕ symbols represents NOT, AND and XOR operations. In the last step, $A[0,0] \leftarrow A[0,0] \oplus RC$, where RC is the round constant specific for each of the 24 rounds of Keccak-f[1600]. In the final squeezing phase, the first r bits of the state are returned as the output block.

2.1.2. RERO Method

The proposed RERO-based error detection method for Keccak-f[1600] can also be applicable to other variants of Keccak. A simplified structure RERO-based error detection approach is shown in Fig. 5. A pipeline-register has been placed after the π step compare in Fig. 4 and 5. The location for placing the registers is chosen to break the timing path into approximately equal halves. It is denoted as the two halves of pipelined stages by H1 and H2.

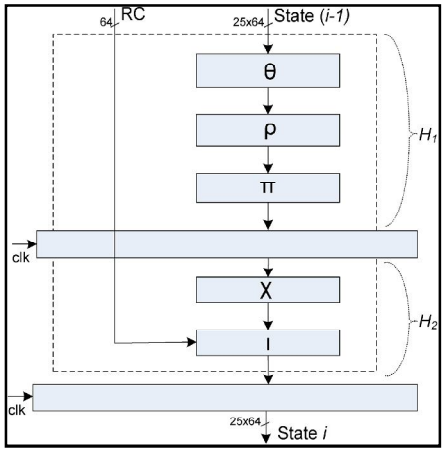

Figure 3: Keccak algorithm modified for the presented RERO-based approach[12].

In Fig.3 the original input is first applied to the architecture in the first cycle. In the second cycle, while the second half of the circuit *H*2 executes this first input, the rotated variant of the first input is fed to the first half of the circuit H1. This trend is consecutively executed until the last rotated input is derived for second runs, each of 26 input words are rotated by a number between 1 and 63, where each word has 64 bits for detecting the errors, the outputs of the runs with the rotated-inputs are rotated back and compared against the original inputs and any mismatch indicates an error. the advantage of subpipelining is to reduce the throughput degradation of the proposed scheme. Hence, the order of applying the inputs is managed so that the advantage of concurrent executions taken. Although the added subpipelining registers slightly increase the induced hardware overhead, it is more preferable to use 100% time-redundancy schemes which introduce much more overall design overhead. Time-redundancy techniques inherently tend to increase the number of cycles needed for computations. This reduces the throughput of the hardware implementations.

48 cycles are needed for the original pipelined architecture of Keccak-f[1600] to execute 24 rounds of five steps each, the number of cycles is intact when the RERO based approach is utilized. This is because of the feedback structure of the rounds of Keccak-f[1600] as shown Fig. 3. Consequently, when derive the overhead is derived of the proposed approach through FPGA syntheses, low degradation in throughput is observed compared to the original Keccak-f[1600]. Finally, the proposed approach is capable of detecting both transient and permanent errors with very high coverage. Due to its low hardware overhead, the presented approach is suitable for the applications requiring high performance and low complexity.

2.1.3. SHA-3/Keccak algorithm

SHA-3 is one of the important cryptographic tools which is used for system and information security. SHA-3/Keccak algorithm consists of padding module and permutation module.

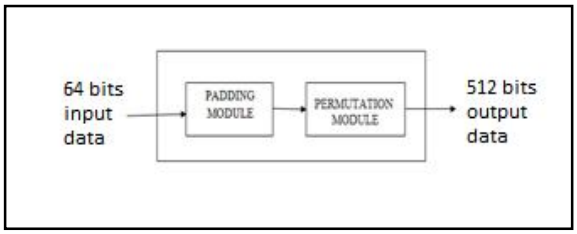

Figure 4: block diagram of SHA-3/ keccak Algorithm.

In the fig 4, the block diagram of SHA-3/keccak algorithm is presented. It consists of padding module and permutation module. The padding module converts 64 bit user data into 576 bit data because of r=576 value in NIST standard. The output of padding module is fed to the permutation module. This permutation module generates 512 bits of hash values.

2.1.4. Architecture of the Padding Module

Padding module converts 64 bit input data into 576 bit data because the width of the user input is less than 576 bit. So the padding module convert the user data into 576 bit.

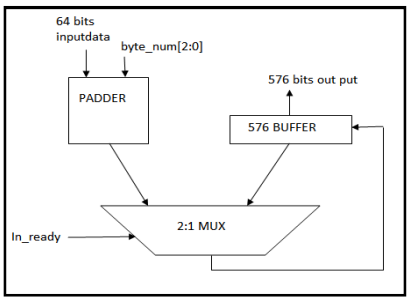

Figure 5: Architecture of the padding module.

fig 5 shows the architecture of padding module. The width of the user input is less than 576 bit, the padding module converts the user data into 576 bit because r value is 576 according to the NIST standard. Initially user data is 64 bits the padder concatenate the zeros with the input data based on the byte_num[2:0] and fed to the mux. This byte_num[2:0] generates 8 states the padder madule concatenate 8 bits input data with the zeros in each state. After 8 states the padder generates 64 bits. In the padding module internally has 8 padder modules. Finally, padding module generates 512 bits output data and 64 bit zeros and fed into the mux. The mux produces the output based on the IN_READY state. The output of mux is fed to the buffer. If the buffer grows full IN_READY pin goes low, then the permutation module begins the calculation and the padder module takes new input data.

2.1.5. Permutation Module

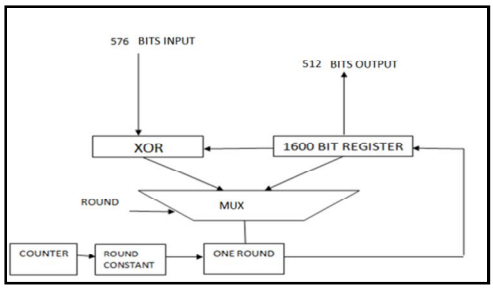

Figure 6: Architecture of permutation module.

In the Fig 6, shows that the architecture of permutation module with initially 576 bit data is to XORed with the default value in register and fed to the mux. The mux generates the output based on the round control pin. This pin goes low if the round value is 23. The round constant module is implemented by combinational logic which saves resource compared to the RAM, because most bits in the round constant is zero.

3. Simulation Results

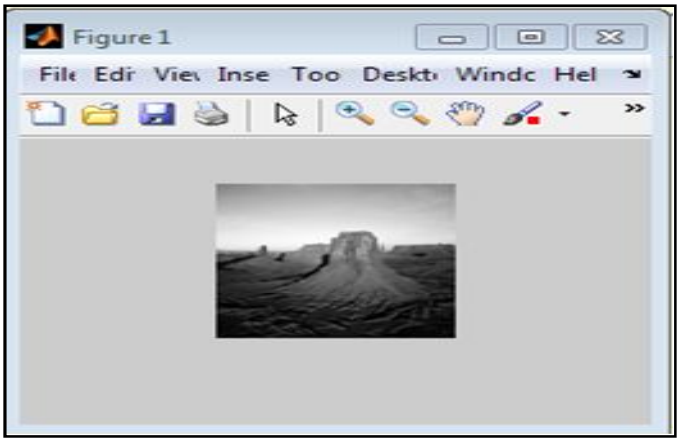

Figure 7: Input image

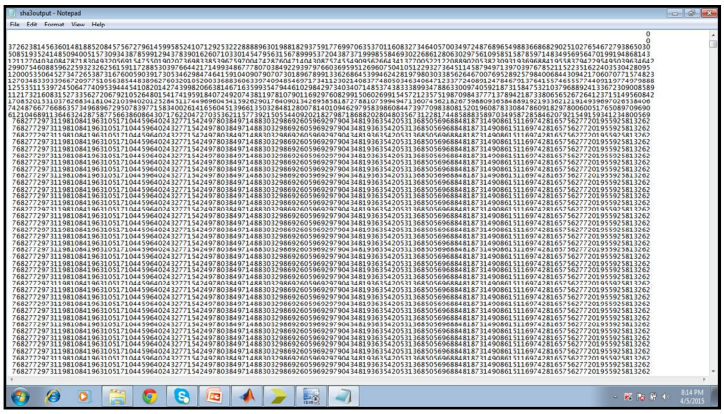

Figure 8: Hashed values of the image.

Figure 9: Final simulation results of algorithm by passing image.

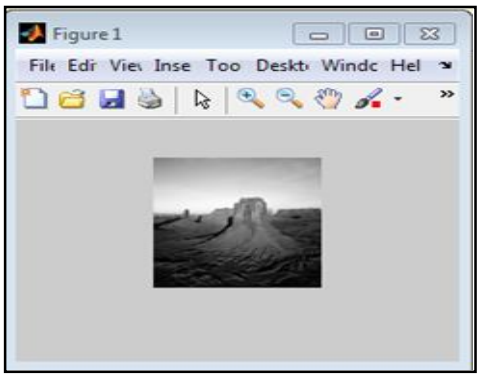

Figure 10: Output image.

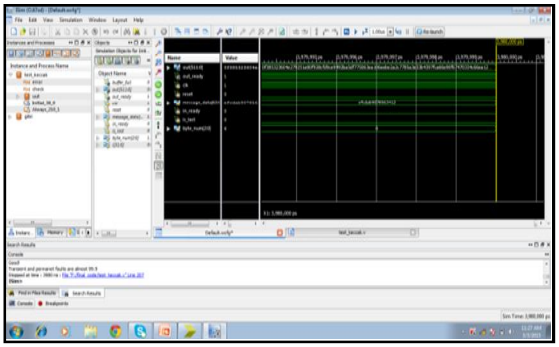

Figure 11: Simulation results of RERO method.

In simulation results, it shows error capability of the Keccak algorithm. In the console it shows good or error. If it shows good in Fig 11 then there is no error in the algorithm or else it shows error as shown in Fig 12 below and also it shows error detection capability which corresponds to the transient and permanent faults is 99.99%.

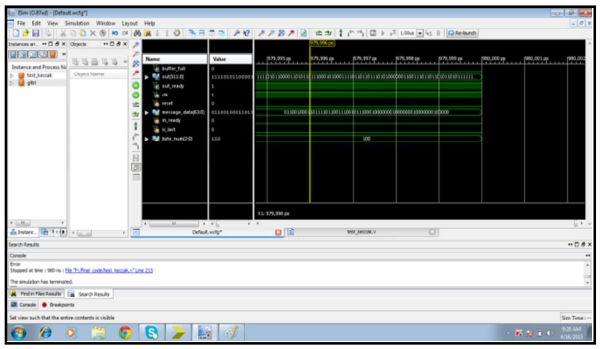

Figure 12: Simulation results of RERO method.

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	2428	28800	8%
Number of Slice LUTs	5313	28800	18%
Number of fully used LUT-FF pairs	1726	6015	28%
Number of bonded IOBs	521	480	108%
Number of Block RAM/FIFO	3	60	5%
Number of BUFG/BUFGCTRLs	1	32	3%

Figure 13: Design summary of the proposed method.

4. Conclusion

In this proposed scheme time-redundancy method has been presented for error detection of the recently-standardized secure cryptographic SHA-3 algorithm, i.e., Keccak. The proposed approach is based on the low hardware overhead RERO-based method. The simulation results shows the error detection capability of the algorithm is 99.99% by passing 64 bits input data. Based on the reliability requirements and available resources the RERO based error detection scheme is introduces for making the hardware implementations of reliable secure cryptographic SHA-3 algorithm.

5. References

i. Keccak Hash Function, NIST (National Institute of Standards and Technology), 2014, Mar, Available: http://csrc.nist.gov/groups/ ST/hash/sha-3.

ii. Luca Henzen, 2012, Developing a Hardware Evaluation Method for SHA-3 Candidatest, e-Print [Online], Available: http://eprint.iacr.org/2012/004.pdf.

iii. Miroslav Kneˇzevic, S. Tillich, M. Feldhofer and A.Szekely, Uniform evaluation of hardware implementations of the round-two SHA-3 candidates, in Proc. Conf, 2010 SHA-3 Candidate, pp.1-16.

iv. Elhadi, E. M. Shakshuki, N. Kang and T. R. Sheltami, EAACK–A secure intrusion detection system for MANETs, IEEE Trans, Ind. Electron., Mar. 2013,vol. 60, no. 3, pp. 1089–1098.

v. M. Mozaffari-Kermani, M. Zhang, A. Raghunathan and N. K. Jha, Emerging frontiers embedded security, in Proc, Conf, VLSI Design, Jan. 2013, pp. 203–208.

vi. A. Reyhani-Masoleh and M. Mozaffari-Kermani, A fault detection scheme for the FPGA implementation of SHA-1 and SHA-512 round computations, J. Electron, Test. , 2011, vol. 27, no. 4, pp. 517–530.

vii. M. Mozaffari-Kermani and R. Azarderakhsh, Efficient fault diagnosis schemes for reliable lightweight cryptographic ISO/IEC standard CLEFIA benchmarked on ASIC and FPGA, IEEE Trans, Ind, Electron, Dec. 2013,vol. 60, no. 12, pp. 5925–5932.

viii. A.Reyhani-Masoleh and M. Mozaffari-Kermani ,Reliable hardware architectures for the third-round SHA-3 finalist Grøstl benchmarked on FPGA platform, in Proc,Conf. Defect Fault Tolerance VLSI Syst, Oct. 2011, pp. 325–331.

ix. E.Homsirikamol, J. Li and E. E. Swartzlander, Concurrent error detection in ALUs by recomputing with rotated operands, in Proc, Conf. Defect Fault Tolerance VLSI Syst., Oct. 1992, pp. 109–116.

x. R. Karri, K. Wu, P. Mishra and Y. Kim, Concurrent error detection schemes of fault based side-channel cryptanalysis of symmetric block ciphers, IEEE Trans. Computer-Aided Design Integr, Circuits Syst., Dec. 2002, vol. 21, no. 12, pp. 1509–1517.

xi. D.-J. Bernstein and T. Lange, 2012, The new SHA-3 software shootout. E-Print [Online], Available: http: //eprint.iacr.org/2012/004.pdf.

xii. Siavash Bayat-Sarmad, Mehran Mozaffari-Kermani and Arash Reyhani-Masoleh, Efficient and Concurrent Reliable Realization of the Secure Cryptographic SHA-3 Algorithm, IEEE Trans, computer-aided design of integrated circuits and systems, vol. 33, no. 7, july 2014.