Low Power and Pipelined Secure hashing Algorithm-3(SHA-3)

Jayanti Sharma
Dept. of Electronics and Communication Engineering,
PESIT,
Bangalore-560085, India
jayanti.sharma15@gmail.com

Deepali Koppad, Professor
Dept. of Electronics and Communication Engineering,
PESIT,
Bangalore-560085, India
deepalikoppad@pes.edu

Abstract— **Cryptography plays an important role in the security of data. Even though the data is encrypted it can be altered while transmitting on the network so data should be verified using a digital signature. Hashing algorithms are used to create these digital signatures for verification of the data received. Hashing algorithm like Secure hash algorithm-3 SHA-3(512) (keccak) is designed which has a fixed output length of 512-bits. Then to improve on power a low-power technique such as latch based clock gating technique is used. After applying these techniques all the designs are compared in terms of power, delay and frequency. SHA-3 algorithm is designed using Verilog HDL and simulated in Xilinx ISE v14.2.**

Keywords— SHA-3; Hashing; low-power; pipelining; clock gating (key words)

I. INTRODUCTION

To prevent any data alteration while sending it on the network, data has to be digitally signed. If someone on the network tries to alter the data then its digital signature will also change. So at the receiver side the digital signatures will not match and data alteration can be detected.

Hash functions can be used to generate these digital signatures. Message which has to be transmitted is given as input to the hash function and it will generate the corresponding hash value or message digest

Most common hash fuctions in use are SHA-0 (Secure hash algorithm-0), SHA-1, MD4 (Message digest-4), MD5 and RIPEMD. But cryptanalysis of these algorithms is done and it was found that these algorithms are vulnerable to several attacks like collision resistance, birthday attack, etc so SHA-2 and SHA-3 algorithms came in to existence as till now no attacks have been reported against these algorithms. Since SHA-2 is algorithmically similar to SHA-1, there are chances of cracking SHA-2 in the near future. Therefore National Institute of Standards and technology (NIST), USA announced SHA-3 (keccak) as the new standard hashing algorithm. SHA3 is less prone to attacks like collision resistance and pre-image resistance because of its increased number of rounds as compared to other algorithms like SHA2, MD5 and SHA1. SHA3 is the most secure hashing algorithm so far [11].

On the basis of output length SHA-3 is further classified as SHA3-224, SHA3-256, SHA3-384, and SHA3-512 will give 224-bit, 256-bit, 384-bit, 512-bit hash values respectively and it is also classified as extendable-output functions (XOFs) which are SHAKE128 and SHAKE256 [4].

In this paper, a combinational Secure hashing algorithm SHA-3(512) (keccak) is designed which has a fixed output length of 512-bits and its power, delay and frequency is measured. Then to improve its performance in terms of frequency pipelining is done as explained in Section IV. To achieve low-power a latch based clock gating technique is used explained in Section V. Once this is done all the designs are compared in terms of power, delay and frequency.

The paper is organized as follows. Section II explains the SHA-3 algorithm, and Section III includes the design of combinational SHA-3 and Section IV includes pipelined SHA-3. Section V includes Clock gated and pipelined SHA-3 and Section VI includes comparison among the designs, while Section VII concludes the paper.

II. SECURE HASHING ALGORITHM-3(SHA-3)

Secure Hash Algorithm-3 (SHA-3) is chosen as the standard algorithm among various hashing algorithm and it is based on Keccak algorithm. Keccak has emerged as a new Secure Hash Algorithm-3. It consists of various SHA3 variants like SHA3-224-bit, SHA3-512-bit, SHA3-384-bit and SHA3-256-bit. It comprises of various rounds and each round includes some logical operations. Basically it is generated by using sponge function in which input is first absorbed and then squeezed to give desired output.

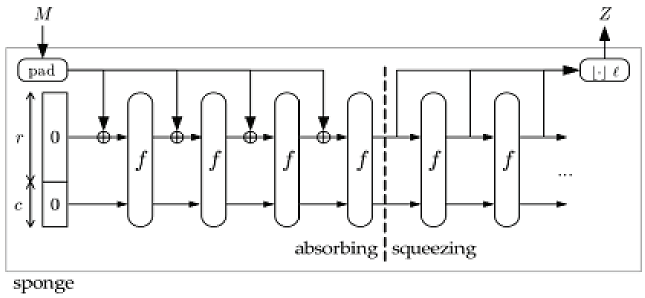


Fig.1. Sponge Function.

Sponge function consists of three operations- initializing, absorbing and squeezing as shown in Fig.1 where input message 'M' is given to padding module and at the output side hash value of the input is obtained as 'Z'

- During initialization phase input matrix is initialized with zeros and input padding is done to make an arbitrary input block equal to 1600 bits.
- Then in absorbing phase input matrix is XORed and all 24 rounds of computation are carried out.
- During squeezing operation, desired length of output is obtained by truncating the input matrix.

III. Design of Combinational SHA-3

SHA-3(512) algorithm is designed which will have the fixed output length of 512-bits. Input can be of any length, it can be 128bit, 1600bit, or 5bit. Here, input is taken as 5 bit and 1600 bit.

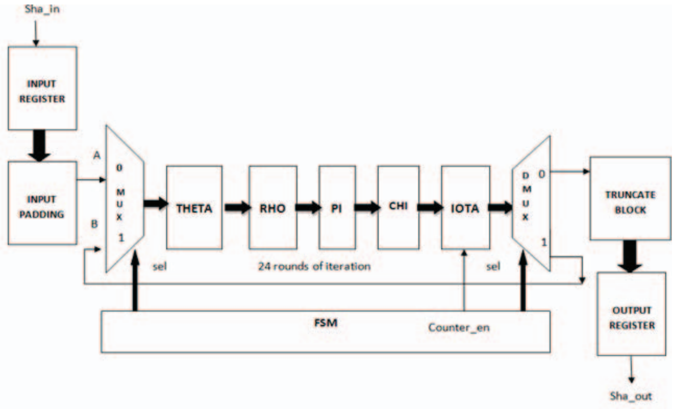


Fig.2. SHA-3 Combinational is kept between 2 registers.

Sha_in is the 5-bit input or 1600-bit given to input register as shown in Fig.2, input padding module will pad the input with required number of zeros to make it 1600 bit. Now this input of 1600 bit is given to the MUX. When sel signal for MUX is 0 this will connect the input 1600 bit to theta input via MUX. Then this input will pass through theta, rho, pi, chi and iota modules. For iota module there is XORing of input with Round constants which have different values for different rounds. So these values of Round constant for each round are provided by an external counter.

Finite state machine (FSM) is designed which controls the input of MUX and DMUX. FSM has two states and it has 3 outputs mux_sel, dmux_sel and counter_en that means its mux_sel, dmux_sel output is connected to select line of MUX, DMUX respectively and counter_en is connected to iota module. Each time the iota output is feedback the counter increments by 1. During this time the sel signal of MUX and DMUX will be 1, this will ensure the output of iota stage will be fed back to the theta stage for the next round computations.

Once the counter completes 24 rounds then the sel signal of MUX and DMUX will be set to 0 thereby the output will propagate as valid output to the truncation block. After removing the extra bits from the output, the desired 512-bit hash output is stored in output register and given as Sha_out as shown in Fig.2.

Since this is a combinational logic so when static timing analysis tools perform timing analysis they evaluate only a register to register path and thus it is necessary to keep the combinational logic between two registers. In Fig.3 Circle depicts combinational logic which is kept between two registers.

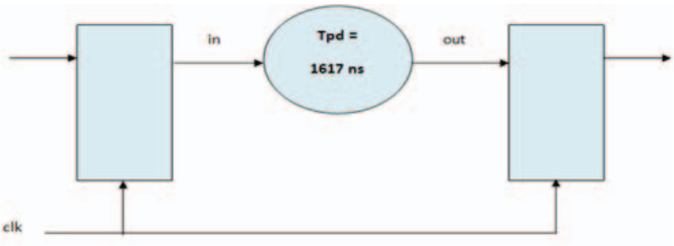


Fig.3.SHA-3 Combinational logic kept between 2 registers.

After measuring the frequency of the above design, it was found that it has a propagation delay of 1617 ns.

The combinational logic has a propagation delay of 1617 ns and gives a frequency of 0.618MHz which is very low frequency. One method to increase the frequency is to make use of pipelining.

Initially input is zero and reset is high. When reset is 0, output out_2 is obtained after a single clock cycle. Since it’s a combinational logic so output is obtained instantly just after a single clock cycle delay as shown in the simulation result Fig.4. The clock is used here only for clocking the FSM and is not used in the SHA3 step mapping computations.

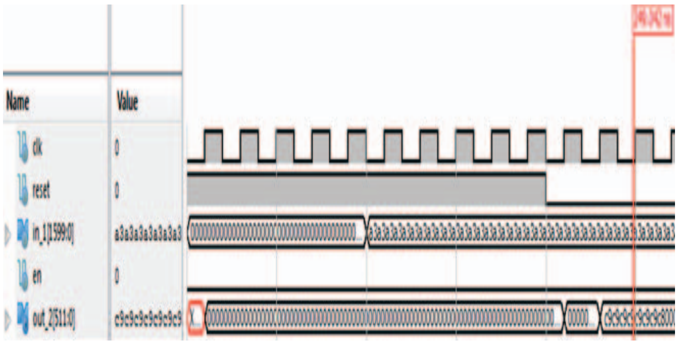
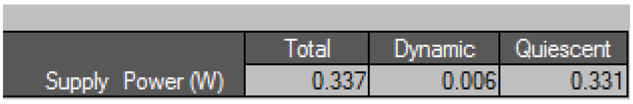


Fig.4.Simulation result of SHA-3 combinational with 1600-bit input.

Total power of combinational SHA-3 is calculated using Xpower Analyzer and shown in Fig.5. Total power is 0.337W out of which static power is 0.331W and dynamic power is 0.006W.



	Total	Dynamic	Quiescent
Supply Power (W)	0.337	0.006	0.331

fig 5. Power results of Combinational SHA-3.

IV. Design of Pipelined SHA-3 Algorithm

Pipelining is done to increase the performance of SHA3 algorithm. Registers are inserted in between step mapping rounds of SHA-3 [2]. First register is inserted between theta and rho module, second in between rho and pi module, third register is inserted between pi and chi module and the last register is inserted between chi and iota module named as w,x,y,z respectively as shown in Fig.6. These registers are controlled by the inputs like clock signal and clock enable signal.

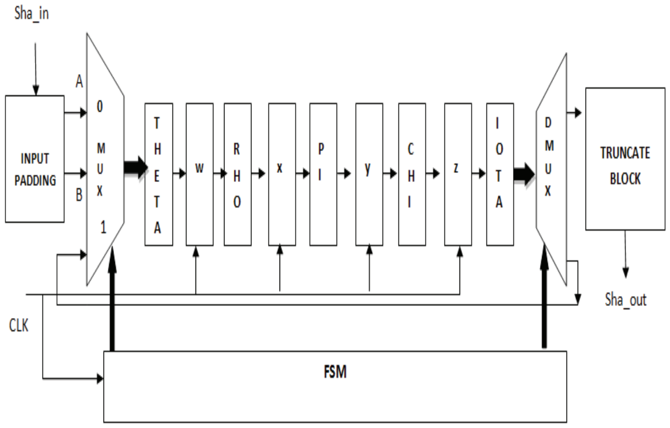


Fig.6.Pipelined SHA-3.

In order to increase the clock frequency the large data path of 1617 ns is broken or split in between at one or multiple places to increase the clock frequency. This can be achieved by using the technique of pipelining. After splitting the combinational logic, the circuit obtained will be similar to the one shown in Fig 7.

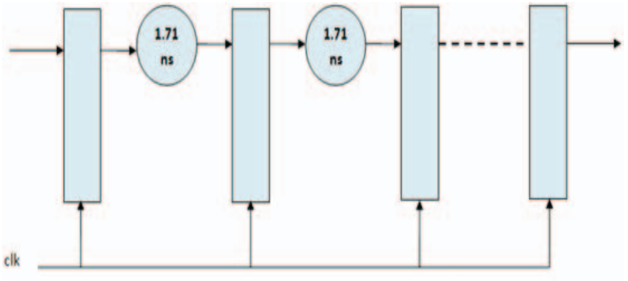


Fig.7.Pipelining registers inserted showing the delay

This would help to reduce the total path between two registers thus now instead of covering a very long data path of 1617 ns the logic signals have to cover only a path of 1.71 ns between each register slice as obtained in the synthesis report. Thus this enables the circuit to work at a much higher frequency of 581MHz.

Fig.8 shows the simulation result of Pipelined SHA-3. Initially input is zero, reset is high and enable is zero. When reset is 1 output is 0, as reset becomes 0 and enable signal goes high and the output is obtained after 12 clock cycles. Clock is enabled throughout the operation as shown in Fig.8.

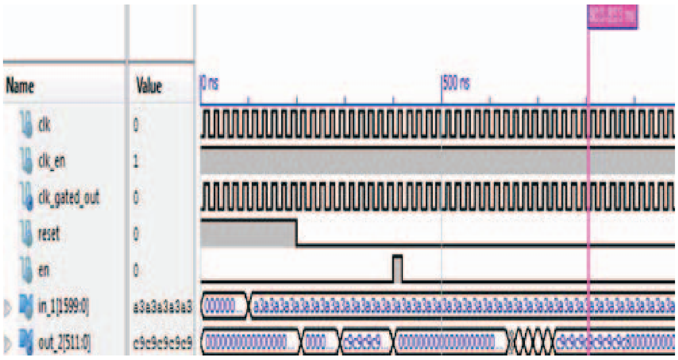
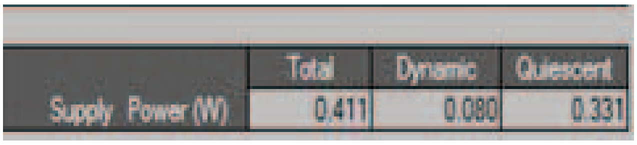


Fig.8.Pipelined SHA-3 with 1600-bit input.

Power is calculated using X power analyzer tool. Total Power of pipelined SHA-3 computed and it is equal to 0.411W out of which 0.331W is static power and 0.080W is the dynamic power as shown in Fig.9.



	Total	Dynamic	Quiescent
Supply Power (W)	0.411	0.080	0.331

Fig.9.Power results of Pipelined SHA-3 algorithm

V. Clock Gated and Pipelined SHA-3 Algorithm

Clock gating is implemented as a part of pipelining registers. A latch based clock gating is applied to the pipelined SHA-3 as shown in Fig.10. In this design Clock and clock_enable are the two inputs to the latch and clock_gated_output is the output of the latch based clock gating. Clock is controlled by a clock enable signal, whenever clock is required clock enable will be 1 otherwise clock enable will be 0 which will switch off the unnecessary switching of clock.

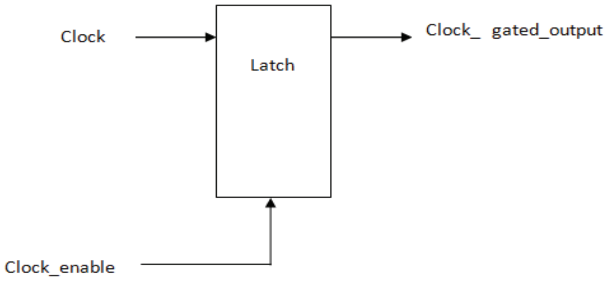


Fig.10. Latch based clock gating.

The presence of latch based clock gating prevents glitches from propagating to the clock input of the register as whenever the clock enable goes low the latch will latch the previous stae of the output and not allow propagation of glitches to the clock input of the gated register.

Clock gated architecture is shown in Fig.11 in which Clock gating is implemented as a part of pipelining registers. So whenever clock is not required at the registers it can be switched off using this clock enable signal.

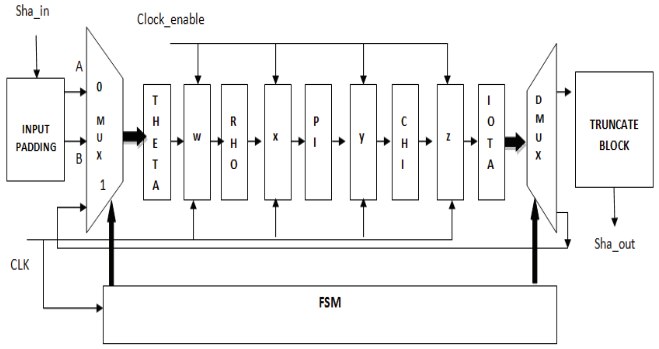


Fig.11. Clock gated pipelined architecture.

Pipelining registers w,x,y,z consists of following structure shown in Fig.12. Clock gated signal is the output of latch which can switch off the pipeline registers when not required.

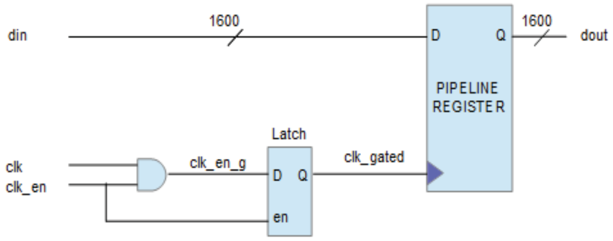


Fig.12.Clock gating technique implemented as a part of Pipelining register.

Simulation result of clock gated and pipelined design is shown in Fig.13 where clock is controlled by a clock enable signal, whenever clock is required clock enable will be 1 otherwise clock enable will be 0 which will switch off the unnecessary switching of clock. Clock and clock_enable are the two inputs to the pipelining registers.

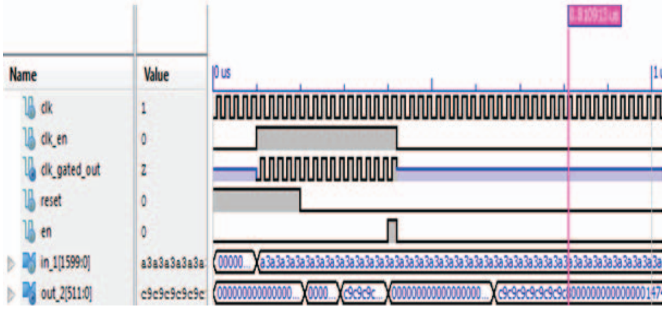
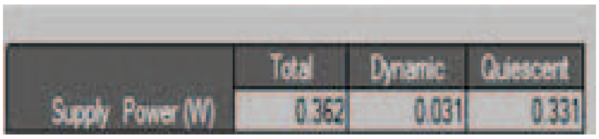


Fig.13.Simulation result of Pipelined and clock gated SHA-3

Total power of pipelined SHA3 is computed when clock signal is disabled when not required (when clock gating is applied) is shown in Fig.14.



	Total	Dynamic	Quiescent
Supply Power (W)	0.362	0.031	0.331

Fig.14. Power results of Clock gated and pipelined SHA-3.

After applying clock gating total power is computed as 0.362W out of which 0.331W is static power and 0.031W is the dynamic power as shown in Fig.14.

VI. COMPARISON

Comparison of all the designs is done and shown in table 1. It can be seen that using the pipelining technique, the frequency of the design is improved from 0.618 MHz to 581.937 MHz, delay also got reduced from 1617ns to 1.71ns. Total power of combinational SHA-3 is 0.337W while the power after applying pipelining is 0.411W.

Using clock gating with pipelining maintains the same frequency and delay as design without clock gating. The advantage of this design is the reduction in the dynamic power to 0.031W from 0.080W.

.

TABLE I.COMPARISON OF ALL THE DESIGNS IN TERMS OF FREQUENCY AND POWER

S. No .	Design	Frequency (MHz)	Delay (ns)	Power (W)	No. of LUTs out of 612000	Device
1	Combinational SHA-3	0.618	1617	0.337	6389	Virtex-7
2	Pipelined SHA-3	581.937	1.718	0.411	5370	Virtex-7
3	Clock gated and Pipelined SHA-3	581.937	1.718	0.362	5370	Virtex-7

IV. CONCLUSIONS

The SHA-3 design was implemented using only combinational circuits. It was seen that the frequency of this design is very less. Hence an improvement was suggested to use the pipelining technique. The overall frequency of the design was improved from 0.618MHz to 581MHz.

Power of all the designs is compared and it was found that the total power of combinational SHA-3 was 0.337W out of which 0.331W was static power and 0.006W was dynamic power. Then after the application of pipelining total power increased to 0.411W but at the cost of increased performance. And after applying clock gating to the design static power remains same as 0.331W for both pipelined and clock gated designs but dynamic power got reduced from

0.080W to 0.031W after applying clock gating. Hence 0.049W or 49mW of dynamic power of the design is saved after applying clock gating.

References

[1] Alia Arshad, Dur-e-Shahwar kundi, Arshad Aziz, "Compact Implementation of SHA3-512 on FPGA",*IEEE Conf. on Information Assurance and Cyber Security (CIACS*), Rewalpindi , Pakistan, pp. 29-33, 12-13 June, 2014.

[2] George S. Athanasiou, George-Paris Makkas, Georgios Theodoridis, "High throughput pipelined FPGA implementation of the new SHA-3 cryptographic hash algorithm", *6th IEEE Int. Symp. on Communications, Control and Signal Processing (ISCCSP)*, Athens, Greece, pp. 538-541, 21-23 May 2014.

[3] Lenos Ioannou, Harris E. Michail, Artemios G. Voyiatzis, "High Performance Pipelined FPGA Implementation of the SHA-3 Hash Algorithm", *4th IEEE Mediterranean Conf. on Embedded Computing (MECO)*, Budva, Montenegro, pp.68-71, 14-18 June 2015.

[4] Muzaffar Rao, Thomas Newe and Ian Grout, "Secure Hash Algorithm-3(SHA-3) implementation on Xilinx FPGAs, Suitable for IOT Applications", *IEEE Int. Conf. on Computer and Information Technology, Ubiquitous Computing and Communications*, *Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM)*, Liverpool, England, pp. 2212-2216, 26-28 Oct. 2015

[5] Meera.k, Krishna sankar.p, Sriram kumar.k, "Redundant file finder, remover in mobile environment through SHA-3 algorithm", *2nd IEEE Int. Conf. on Electron. and Commun. Systems (ICECS)*, Coimbatore, India, pp. 1440-1447, 26-27 Feb 2015.

[6] Ashish Kumar, Vishal Arora, "Analyzing the Performance and Security by using SHA3 in WEP", *IEEE Int. Conf. on Engineering and Technology (ICETECH)*, Coimbatore, India, pp. 1-4, 20 March 2015.

[7] Liang Han, Bai Guoqiang , "Hardware implementation analysis of SHA-3 candidates algorithms", *10th IEEE Int. Conf. on Solid-State and Integrated Circuit Technology (ICSICT)*, Shanghai, China, pp. 266-268, 1-4 Nov. 2010.

[8] Pei Luo, Liwei Zhang Yunsi Fei, A. Adam Ding, "Towards Secure Cryptographic Software Implementation Against Side-Channel Power Analysis Attacks", *26th IEEE Int. Conf. Application-specific Systems, Architectures and Processors (ASAP)*,Toronto, pp. 144-148, 27-29 Jul. 2015.

[9] FIPS 202, *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions,* National Institute of Standards and Technology (NIST),5August,2015. [online].Available:http://csrc.nist.gov/publications/drafts/fips-202/fips_202_draft.pdf.

[10] Jagrit Kathuria, M. Ayoubkhan, Arti Noor, "A Review of Clock Gating Techniques", Int. J. of Electron. and Commun. Eng., vol.1, pp. 106-114, Aug 2011.

[11] Christof Paar, *Understanding Cryptography – SHA3 and The Hash Function Keccak.* Springer.

[12] Samir Palnitkar, *Verilog HDL-A guide to digital design and synthesis*. Prentice Hall PTR, 2nd ed., California: Sun Microsystems, 2003.

[13] Nazeih M. Botros, *HDL Programming VHDL and Verilog*. Dreamtech Press, 2006.