Round Rescheduling Technique for Hash Function Blake using Carry Select Adder with Binary to Excess Converter

S.Jananee, N.Annu, K.Jayapriyadharshini

Abstract— Cryptographic hash functions are used widely in many applications mainly for its high speed and security. NIST organized SHA-3 competition and the final round candidates are BLAKE, KECCAK, SKEIN, JH AND GROSTL. Among the five finalists design and architecture of BLAKE is analyzed in this paper. Hash function BLAKE is the one-way cryptography which means no key is used while sending and receiving the message. In the field of cryptography speed and secrecy are the tradeoffs. To obtain high speed and efficiency, Round Rescheduling Technique is incorporated. To make BLAKE more efficient, modular addition is replaced with Carry Select Adders (CSA) using Binary to Excess Converter (BEC). The Existing and Proposed architecture of BLAKE is designed using CSA whereas Modified BLAKE is designed using CSA with BEC. So that the Area and Power consumed in Proposed method is less compared with Existing methods. BLAKE-32 and BLAKE-64 are coded in VHDL language and simulated in Mentor Graphics Front-End tool - Modelsim. Area and Power results are shown in Xilinx ISE simulator.

Keywords-- Hash Function, SHA-3, NIST CSA, BEC, VHDL, Xilinx ISE simulator.

I. INTRODUCTION

Cryptography is probably the most important aspect of communications security and is becoming increasingly important as a basic building block for computer security [1]. The increased use of computer and communication systems by industry has increased the risk of theft of proprietary information. Although these threats may require a variety of countermeasures, encryption is a primary method of protecting valuable electronic information [7]-[9]. It is an art of codifying messages, so that they become unreadable. In field of cryptography hashes have further applications including the creation of authentication codes, digitally signing documents and generation of statically random data streams. Hash functions have no key since the plaintext is not recoverable from cipher text [4]. A function that maps a variable –length data or message into a fixed –length valve called a hash code.

Manuscript Received on February, 2013.

S. Jananee, Electronics and Communication Engineering, Avinashilingam Institute for Home Science and Higher Education for Women University, Coimbatore, India.

N.Annu, Electronics and Communication Engineering, Avinashilingam Institute for Home Science and Higher Education for Women University, Coimbatore,India.

K.Jayapriyadharshini, Electrical and Electronics Engineering, Annamalai University, Coimbatore, India.

The function is designed such a way that when protected it provides an authenticator to the data or message [2]. Also referred to as a message digest. For example person's name having a variable length could be hashed to a single integer. The ultimate purpose of a hash function is to produce a "finger print" of a file, message or other blocks of data and it should be collision free. The National Institute of Standards and Technology (NIST) have established a set of standardized hash functions called the Secure Hash Algorithm (SHA)[3]. The first version is SHA-0 and then consecutively SHA-1, SHA-2 and SHA-3 versions are introduced to provide additional security [5]. NIST announced a global competition to find a new SHA function in 2007 and submissions were accepted for approximately one year [11],[12]. Totally 64 submissions are received out of which 51 were accepted as first round candidates and 14 as second round candidates in 2009[1]. They are BLAKE, Blue midnight wish, Cube hash, Echo, Fugue, Grostl, Hamsi, JH, Keccak, Luffa, Shabal, SHA vite-3, SIMD, Skein[8]. The final round candidates are announced in 2010 as BLAKE, Grostl, JH, Keccak and Skein [10]. The algorithms are being analyzed and narrowed down through elimination rounds, based on Security, Performance and Design of each function.

In this paper SHA-3 finalists BLAKE is analyzed with its design, Architecture and its Performance. BLAKE obtains high speed and security apart from other finalists. Our work extends by replacing the modular addition in rescheduled G function with carry select adders using BEC ,so that the power and area occupancy is less when compared with the existing work.

The rest of the paper is structured as follows. Section II gives the complete specification of the BLAKE hash function. Section III describes round rescheduling technique. Section IV discuses the carry select adders using BEC. Section V deals with the simulation work. Section VI estimates the comparison results. Finally the work is concluded in section VII.

II. BLAKE SPECIFICATION

BLAKE has two main versions such as BLAKE-32 and BLAKE-64. Complete specifications are given below.

A. BLAKE-32 and BLAKE-64

The BLAKE-32 algorithm operates on 32-bit words and returns a 256-bit hash value, whereas BLAKE-64 operates on 64- bit words and returns a 512-bit hash value. Length of all variables is doubled when compared to BLAKE-32. The compressive function of BLAKE-64 is similar to BLAKE-32. That it makes 14 rounds instead of ten, and uses rotation distances as 32,25,16,11. After ten rounds, the round function uses the permutations $\sigma_0 \ldots \sigma_3$ for the last four rounds. It's based on the compressive function. This function is decomposed into 3 main parts such as Initialization, Round function and Finalization. The

compressive function of BLAKE -32 has four values as its input

- Chaining value $h = h_0 \ldots\ldots h_7$
- Message block $m = m_0 \ldots.. m_{15}$
- Salt $s = s_0 \ldots.. s_3$
- Counter $t = t_0, t_1$

The input salt is an optional one which is used only for specific applications such as randomized hashing [6]. The output obtained from the compressive function is new chaining value $h' = h'_0 \ldots\ldots h'_7$ of 256 bits.

a) Initialization

$V_0 \ldots.. V_{15}$ initial state is represented in a 4 *4 matrix of 32-bit words.

$$\begin{matrix} & V_0 & V_1 & V_2 & V_3 \\ V_4 & V_5 & V_6 & V_7 & \\ V_8 & V_9 & V_{10} & V_{11} & \\ V_{12} & V_{13} & V_{14} & V_{15} & \end{matrix}$$

Initialization vector used by BLAKE-256 is given below.

IV = 6A09E667 IV = BB67AE85
V = 3C6EF372 IV = A54FF53A
IV = 510E527F IV = 9B05688C
IV = 1F83D9AB IV = 5BE0CD19

Within the compress function, a 512-bit state v is maintained, treated as a 4x4 matrix of 32-bit words. This state is initialized from the current hash, salt value, timer value t, and a 256-bit constant c. The initial state of the compression function is given

$$\begin{matrix} h_0 & h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 & h_7 \\ s_0 \oplus c_0 & s_1 \oplus c_1 & s_2 \oplus c_2 & s_3 \oplus c_3 \\ t_0 \oplus c_4 & t_0 \oplus c_5 & t_1 \oplus c_6 & t_1 \oplus c_7 \end{matrix}$$

where $c_0 \ldots. c_{15}$ are predefined word constants.

The digits of constant c are directly taken from the hexadecimal representation of π, chosen for its irrational nature as shown below

$c_0 =$ 243F6A88 $c_1 =$ 85A308D3
$c_2 =$ 13198A2E $c_3 =$ 03707344
$c_4 =$ A4093822 $c_5 =$ 299F31D0
$c_6 =$ 082EFA98 $c_7 =$ EC4E6C89
$c_8 =$ 452821E6 $c_9 =$ 38D01377
$c_{10} =$ BE5466CF $c_{11} =$ 34E90C6C
$c_{12} =$ C0AC29B7 $c_{13} =$ C97C50DD
$c_{14} =$ 3F84D5B5 $c_{15} =$ B5470917

After initializing the state matrix, it is iteratively processed through 14 rounds. In designing BLAKE less complicate rounds are suggested and absolutely proven to provide greater security [1]. Such as security which means it's difficult to invert such that function inputs cannot be determined from function outputs.

b) Round Function

Each round consists of eight state transformations, labeled G_0 through G_7. These are responsible for the confusion (changes to data) and diffusion (dispersion of data) of the BLAKE algorithm. A round is a transformation of the state that computes

$G_0(V_0, V_4, V_8, V_{12})$ $G_1(V_1, V_5, V_9, V_{13})$

$G_2(V_2, V_6, V_{10}, V_{14})$ $G_3(V_3, V_7, V_{11}, V_{15})$

$G_4(V_0, V_5, V_{10}, V_{15})$ $G_5(V_1, V_6, V_{11}, V_{12})$

$G_6(V_2, V_7, V_8, V_{13})$ $G_7(V_3, V_4, V_9, V_{14})$

Each operates on and modifies only by 4 of the 16 state words which is generalized as a, b, c, and d. this transformation consists of addition , bit rotation and XOR operations. The main architecture of BLAKE is shown in Fig 1.The general G transformation for BLAKE-64 is given by

$a \leftarrow a+b(m_{\sigma r(2i)} \oplus c_{\sigma r(2i+1)})$
$d \leftarrow (d \oplus a) >>> 32$
$c \leftarrow c+d$
$b \leftarrow (b \oplus c) >>> 25$
$a \leftarrow a+b+ (m_{\sigma r(2i+1)} \oplus c_{\sigma r(2i)})$ -----(5)
$d \leftarrow (d \oplus a) >>> 16$
$c \leftarrow c+d$
$b \leftarrow (d \oplus a) >>> 11$

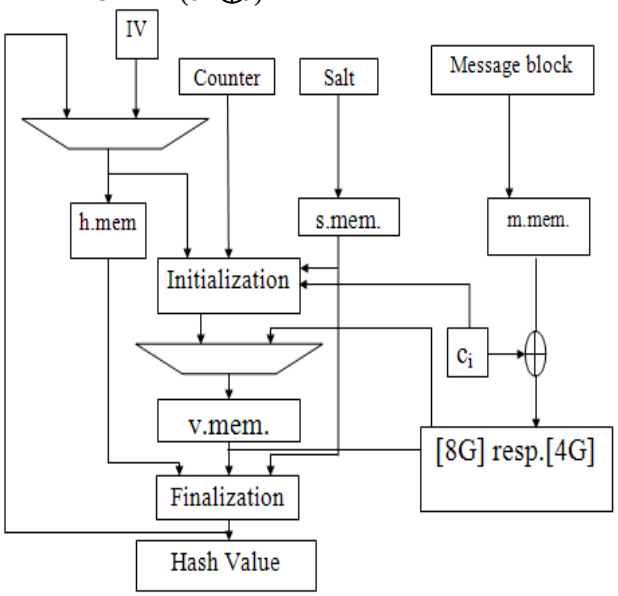

Fig.1 Main Architecture of BLAKE

σ_r refers to permutation s from 0 to 9. The permutation table for BLAKE is shown in following table respectively.

TABLE I

σ_0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
σ_1	14	10	4	8	9	15	13	6	1	12	0	2	11	7	5	3
σ_2	11	8	12	0	5	2	15	13	10	14	3	6	7	1	9	4
σ_3	7	9	3	1	13	12	11	14	2	6	5	10	4	0	15	8
σ_4	9	0	5	7	2	4	10	15	14	1	11	12	6	8	3	13
σ_5	2	12	6	10	0	11	8	3	4	13	7	5	15	14	1	9
σ_6	12	5	1	15	14	13	4	10	0	7	6	3	9	2	8	11
σ_7	13	11	7	14	12	1	3	9	5	0	15	4	8	6	2	10
σ_8	6	15	14	9	11	3	0	8	12	2	13	7	1	4	10	5
σ_9	10	2	8	4	7	6	1	5	15	11	9	14	3	12	13	0

The state words on which each G transformation operates is that first 4 calls $G_0 \ldots.. G_3$ are computed in parallel and from G_4 to G_7 are computed

diagonally. The order $G_0 \ldots G_3$ is called as column step and $G_4 \ldots G_7$ as diagonal step. The parallelization is shown in the Fig.2

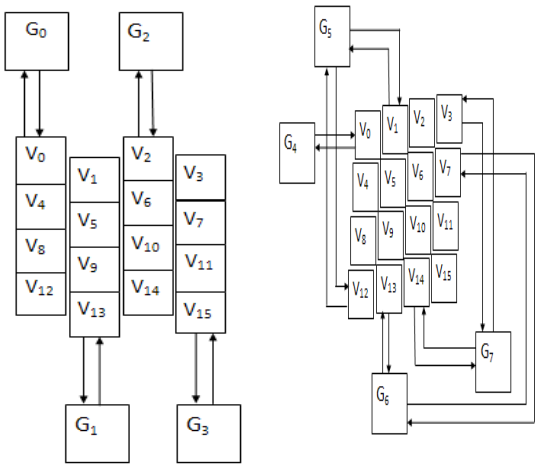

Fig.2. The parallelized column step and diagonal step

c) Finalization

After ten /14 rounds of G transformation the compressive function performs final step. Thus the new chaining value h' is extracted from state $V_0 \ldots V_{15}$ with input of the chaining value h and salt S.

$h_0' <= h_0 \text{ xor } s_0 \text{ xor } v_0 \text{ xor } v_8$
$h_1' <= h_1 \text{ xor } s_1 \text{ xor } v_1 \text{ xor } v_9$
$h_2' <= h_2 \text{ xor } s_2 \text{ xor } v_2 \text{ xor } v_{10}$
$h_3' <= h_3 \text{ xor } s_3 \text{ xor } v_3 \text{ xor } v_{11}$
$h_4' <= h_4 \text{ xor } s_0 \text{ xor } v_4 \text{ xor } v_{12}$
$h_5' <= h_5 \text{ xor } s_1 \text{ xor } v_5 \text{ xor } v_{13}$
$h_6' <= h_6 \text{ xor } s_2 \text{ xor } v_6 \text{ xor } v_{14}$
$h_7' <= h_7 \text{ xor } s_3 \text{ xor } v_7 \text{ xor } v_{15}$

III. ROUND RESCHEDULING TECHNIQUE

The G function of BLAKE is modified with round rescheduling technique. The introduction of the addition with message constant pair in the G function leads to an increment of the propagation delay. If in single call of G, each update of the state has been conceived to operate sequentially, the MC-pair addition can be shifted within the computations. It is thus possible to anticipate it, reducing the critical path of G. the rescheduled G_i (a*,b, c ,d) computes

$a := a^* + b$
$d := (d \oplus a) >>> r_0$
$c := c + d$
$b := (b \oplus c) >>> r_1$
$a := a + b + (m_{\sigma r(2i+1)} \oplus c_{\sigma r(2i)})$
$d := (d \oplus a) >>> r_2$
$c := c + d$
$b := (d \oplus a) >>> r_3$
$a^* := a + (m_{\sigma r+1\,(2i)} \oplus c_{\sigma r+1\,(2i+1)})$

where r_i are the rotation indices for BLAKE-32 and a *corresponds to the modified first input/output variable after the MC addition. To keep the correct functional behavior, a two-input MUX should be inserted before the sequential logic, hence allowing the record of a instead of a* in the last round. This is the main reason why the rescheduling optimization could not be carried out automatically by the synthesis tool and must be instantiated at code level. Anticipated computation takes place in message and constant pair. Adder change makes the hash function even more efficient. Speed of the operation also partially relay on the adder change. Based upon the rounds both the bit words differ and area occupied is also different. To exploit the full parallelizability of BLAKE two types of design have been coded in VHDL.Transformations for BLAKE-32 is doubled and used for BLAKE-64. Furthermore, the complete execution of initialization and finalization can be performed in the same clock cycle, if the new message blocks are given. BLAKE also uses some constant values. The rescheduled G transformation for BLAKE-64 is visually shown in following Fig.3.

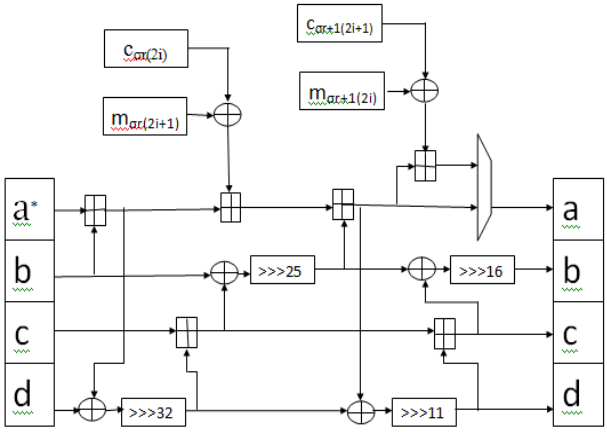

Fig.3. Block Diagram of the Rescheduled G function

In this paper instead of normal modular addition carry select adder with BEC is used. So that the area occupied is less and power consumption is also less. When compared with other adders this CSA suits this technique and proves to be more efficient. Total rounds obtained for BLAKE-64 is 14 and for BLAKE-32 are 10. Round rescheduling technique is provided to speed up the process with high security and delay reduction.

IV. CARRY SELECT ADDERS USING BEC

The design of BLAKE-32bit using CSA is shown in Fig.4. Here it shows an adder with block sizes of 2-2-3-4-5-6-7-3 bit. Each block has 2-2-3-4-5-6-7-3 bit Ripple Carry Adder (RCA) respectively and a multiplexer. Since carry-in is known at the beginning of computation, a carry select block is not needed a mux for the first two bits. The problem of ripple carry adder is that each adder has to wait for the arrival of its carry-input signal before the actual addition can start [22]. The basic idea of the Carry-Select Adder is to use blocks of two Ripple-Carry Adders, one of which is fed with a constant 0 carry –in while the other is fed with a constant 1 carry-in. Therefore, both blocks can calculate in parallel. When the actual carry-in signal for the block arrives, multiplexers are used to select the correct one of both pre calculated partial sums. Also, the resulting carry-out is selected and propagated to the next carry-select block.

However, the CSA is not area efficient because it uses multiple pairs of ripple carry adders(RCA) to generate partial sum and carry by considering carry input cin=1 and cin=0,then the final sum and carry are selected by the multiplexers (mux) which means, the whole blocks has contains so many full adders of RCA. The ripple carry adder has requires more number of logic gates for addition operation. Area for each one of full adder is 13 and area for each one of 2:1 mux is 4.

So that its occupies more area in the regular CSA. The CSA have also required more power and it is not area efficient. In order to overcome these demerits, modification CSA is designed [25]. That is add a BEC –1 instead of RCA with cin=1. Carry select adder is not area efficient; because of number of ripple carry adders are present in each block. Therefore, the carry select adder is modified by replacing BEC structure instead of RCA with cin=1. The logic gates are reduced by this modification. By this modification we may be achieve reduce delay and total area size.

BEC is a non-weighted code. It is also a self-complementing BCD code used in decimal arithmetic units. The Excess-1 code for the decimal number is performed in the same manner as BCD except that decimal number 1 is added to the each decimal unit before encoding it to binary. The main idea of this work is to use BEC instead of the RCA with cin=1 in order to reduce the area and power consumption of the regular CSA [23].

The importance of the BEC logic stems from the large silicon area reduction when the CSA with large number of bits are designed. To replace the n-bit RCA, an n+1 bit BEC is required [12]. A structure and the function table of a n-bit BEC are shown in Fig.5. The Boolean expressions of the 4-bit BEC is listed as (note the functional symbols ~NOT, &AND, ^XOR).

X0=~B0
X1=B0^B1
X2=B2^(B0&B1)
X3=B3^(B0&B1&B2)
C1=B3&B2
C2=B0&B1
C0=C1&C2

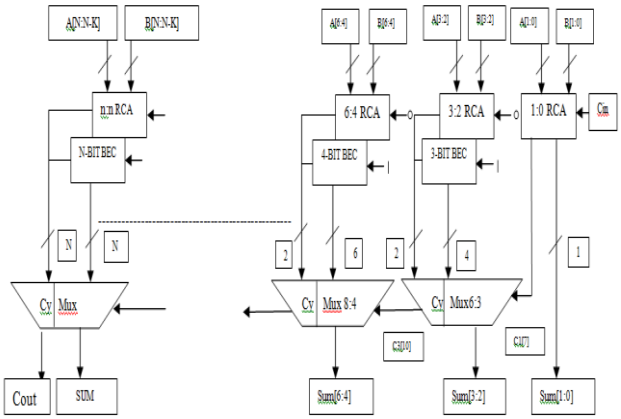

Fig.4.Design of BLAKE using CSA with BEC

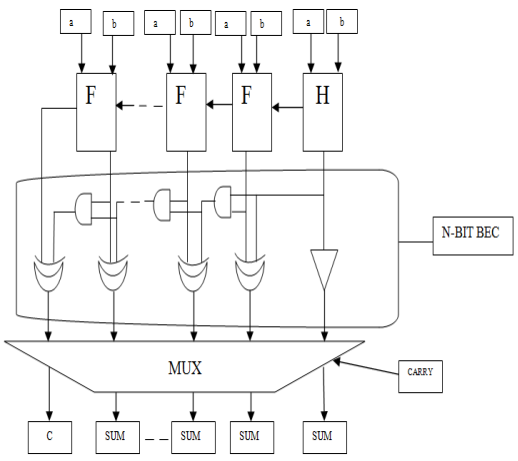

Fig.5. Structure of n – bit BEC

V. SIMULATION WORK

For the simulation work in this paper we adopt VHDL language and simulated in Modelsim and also synthesized in XILINX ISE simulator. The Area and Delay, Power values were obtained from the XILINX ISE simulator.

The design of BLAKE-32 and BLAKE-64 is simulated in Modelsim 10.a. Modelsim is a powerful HDL simulation tool that allows you to simulate the inputs of our modules and view both outputs and internal signals and performance of logic circuits. It allows doing both behavioral and timing simulation. A project is a collection mechanism for an HDL design under specification or test. It may ease interaction with the tool and are useful for organizing files and specifying simulation settings. Fig 6, and 7 shows the snapshots of simulation window. Message, hash, salt, counter are the inputs. hd_0 ,…..hd_7 are the final hash valve. Remaining values shown in the snapshots are the signals.clk is included to obtain the power report. Modified BLAKE-32,and 64 using carry select adders with BEC are shown below..

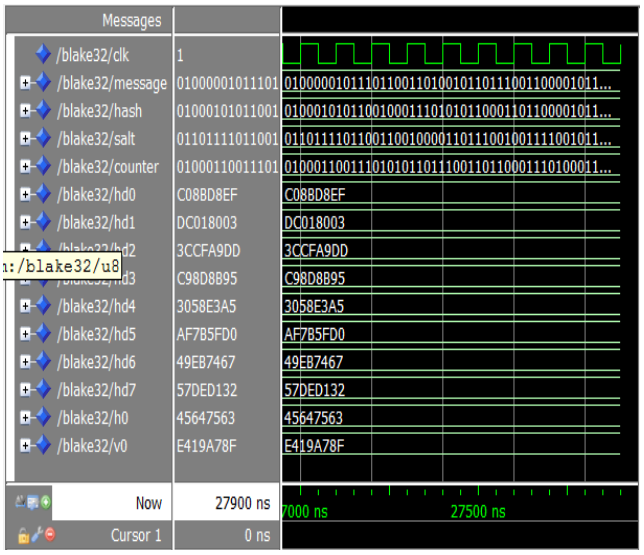

Fig 6. Modified BLAKE-32 using CSA with BEC

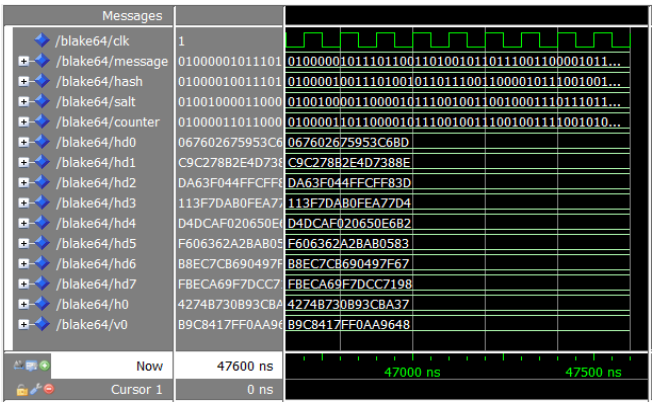

Fig.7. Modified BLAKE-64 using CSA with BEC

VI. COMPARISON RESULTS

The comparative results of BLAKE-32 and 64 using CSA with BEC for Existing, Proposed and Modified are depicted clearly in the following table. The total cell area for Modified BLAKE-32 is 6282 (μm^2) and the cell area for BLAKE-64 is 9757 (μm^2).

The delay report for BLAKE is obtained from Xilinx ISE simulator. The

maximum combinational path delay obtained for BLAKE-32 is 10.162 ns and for BLAKE-64 is 9.532ns. The Maximum input arrival time before clock and Maximum output required time after clock can also be calculated in Xilinx ISE simulator. Comparatively reduction in area and power is obtained effectively with regular and modified structures. The area occupancy is measured by the total equivalent gate counts for the design.

TABLE II

WORD SIZE	HASH FUNCTION BLAKE	AREA (μm^2)	POWER (mW)
32	EXISTING CSA	15,762	73
	PROPOSED CSA	13,100	68
	MODIFIED CSA USING BEC	6282	52
64	EXISTING CSA	17,550	84
	PROPOSED CSA	18,136	81
	MODIFIED CSA USING BEC	9757	60

VII. CONCLUSION

In this paper the Performance and Design of BLAKE is analyzed thoroughly. Also how round rescheduling technique is incorporated in hash function BLAKE is shown clearly. The comparative study of area and power report is shown in table. To make BLAKE more efficient carry select adders are used so that the power consumption is low when compared to other adders. In case of binary to excess converter reduces the gate counts and occupies less area. The major concerns of the VLSI designer were area, performance, cost and reliability. Power consideration was mostly of one secondary importance. Designers of next generation systems want to integrate more features and get higher performance within the same or smaller area and power budget. Furthermore, some applications like signal processing, wireless network, intelligent control, etc have specific power requirements that must be adhered to in order to be compliant with system specifications and standards

REFERENCES

[1] J.-P. Aumasson, L. Henzen, W. Meier, and R. C.-W Phan, "SHA-3 Proposal BLAKE, submission to NIST," 2008. [Online]. Available:http://131002.net/blake/
[2] Cryptanalysis of Dynamic SHA 2 by Jean-philipe, Orr Dunklemen, Bart Preneel Sabastian Indesteege.
[3] SHA-1 Standard, National Institute of Standards and Technology (NIST), Secure Hash Standard, FIPS PUB 180-1,*International Journal of Computer Trends and Technology- volume3Issue2- 2012*, ISSN: 2231-2803 Page 272 1995, available on line at www.itl.nist.gov/fipspubs/ fip180-1.htm
[4] X. Wang and H. Yu, "How to break MD5 and other hash functions," in *Advances in Cryptology—EUROCRYPT 2005, ser. Lecture Notes in Computer Science*. Berlin, Germany: Springer, 2005, vol. 3494, pp.19–35.
[5] NIST, Gaithersburg, MD, "Announcing the secure hash standard," FIPS180-2, 2002.
[6] NIST, Gaithersburg, MD, "SP 800-106, Randomized hashing Digital Signatures," 2007.
[7] L. Ji and X. Liangyu, "Attacks on round-reduced BLAKE," 2009.
[8] Xu Guo , Meeta Srivastav , Sinan Huang , Leyla Nazhandali and Patrick Schaumont Silicon Implementation of SHA-3 Final Round Candidates: BLAKE, Gr_stl, JH, Keccak and Skein dec. 2009,
[9] Ryan Toukatly SHA-3: The BLAKE Hash Function – 2009
[10] Luca Henzen, Pietro Gendotti, Patrice Guillet, Enrico Pargaetzi, Martin Zoller, and Frank K. Gurkaynak. Developing a Hardware evaluation method for sha-3candidates. In Mangard and Standaert [21], pages 248{263.
[11] National Institute of Standards and Technology (NIST). Cryptographic Hash Algorithm CompetitionWebsite.
[12] J.Sklansky, "Conditional-sum Addition Logic", IRE Transactions on Electronic Computers, EC-9,p 226-231, 1960
[13] O.J Bedrij, "Carry-select adder," IRE Transactions on Electronic Computers,pp. 340- 344, 1962.
[14] R.P Brent and H.T.Kung, "A Regular Layout for Parallel Adders", IEEE Transactions on Computers,vol. C-31, No.3,p.260-264, march, 1982.
[15] Tyagi, "A reduced are scheme for carry- select adders", IEEE Transactions on Computers, vol.42, pp.1163-1170, 1993.
[16] Paul F.Stelling, "Design strategies for optimal hybrid final adders in parallel multiplier", Journal of VLSI signal processing, vol 14, pp. 321-331, 1996.
[17] T.Y.Ceiang and M.J.Hsiao, "Carry-select adder using single Ripple carry adder," Electron .Lett., vol.34, no.22,pp.2101-2103, oct. 1998. K.Rawwat, T.Darwish and M. Bayoumi, "A low power Carry select adder with reduces area," proc. of Midwest Symposium on Circuits and Systems, pp.218-221, 2001.
[18] A.Shams, T.Darwish and M.Byoumi,"performance analysis of low power 1-bit CMOS full adder cells," IEEE Trans.VLSI Systems, vol.10,pp.20-29, Feb.2002.
[19] Massimo Alioto and Gaetano Palumbo, "Analysis and Comparison on Full Adder Block in Submicron Technology", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 10, No.6, December 2002.
[20] Y.He ,C.H.Chang and J.Gu, "An area efficient 64-bit square root carry-elect adder for low power applications," in Proc. IEEE Int . Symp.Circuits Syst.,2005, vol. 4,pp.4082-4085.
[21] Behnam Amelifard, Farzan Fallah and Massoud Pedram, "Closing the gap between carry select adder and ripple carry adder: a new class of low-power high performance adders", in Proc. of IEEE International Symposium on Quality Electronic Design (ISQED), 2005.
[22] Yan Sun, Xin Zhang, Xi Jin "High performance Carry select adder using fast All-one Finding Logic", IEEE Int conf modelling and simulation , pp 1012-1014, 2008.
[23] Padma Devi,Ashima Girdhar and Balwinder Singh "Improved Carry select adder with Reduced Area and low power consumption" Int Jou of Com Appl(0975-8887) vol3-no.4,June 2010.
[24] St_ephanie Kerckhof1, Fran_cois Durvaux, Nicolas Veyrat-Charvillon, Francesco Regazzoni, Guerric Meurice de Dormale2, Fran_cois-Xavier Standaert. University catholique de Louvain, UCL Crypto Group,B-1348 Louvain-la-Neuve, Belgium on Compact FPGA Implementations of the Five SHA-3 Finalists.
[25] St_ephanie Kerckhof1, Fran_cois Durvaux, Nicolas Veyrat-Charvillon, Francesco Regazzoni, Guerric Meurice de Dormale2, Fran_cois-Xavier Standaert. University catholique de Louvain, UCL Crypto Group,B-1348 Louvain-la-Neuve, Belgium on Compact FPGA Implementations of the Five SHA-3 Finalists.

S.Jananee received the B.E degree in Electrical and Electronics Engineering from V.L.B Janakiammal College of Engineering and Technology, Coimbatore. Affiliated to Anna University Chennai. And worked as a lecturer for a period of 1 year in Tamilnadu College of Engineering. She presented two paper in National Conference. Participated and Presented ISTE workshop on "Introduction to Research Methodologies" conducted by Indian Institute of Technology, Bombay under National Mission on Education through ICT (MHRD). She is currently pursuing her final year M.E- VLSI Design in Avinashilingam Institute for Home Science and Higher Education for Women University, Coimbatore, India.

N.Annu received the B.E degree in Electronics and Communication Engineering from Sri Ramakrishna Institute of Technology, Coimbatore. Affiliated to Anna University,Chennai. She presented two papers in National level Conference. She is currently pursuing her final year M.E- Medical Electronics. in Avinashilingam Institute for Home Science and Higher Education for Women University, Coimbatore, India.

K.Jayapriyadharshini received the B.E degree in Electrical and Electronics Engineering and M.E degree in Power Systems Engineering, both from Ananmalai University, India. She is currently working as an Assistant Professor in Tamilnadu

College of Engineering. Completed various projects and has professional exposure towards the related concepts and theories of designing.