ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 1, January 2013

Design of a High Throughput Crypto Devices Based On AES

Lokesh Kumar.S[1], Arun Kumar.R[2], Vinoth Kumar.P[3], Santhosh Kumar.A[4], Sriram.A [5]
[1] [2] [3] [4] Final Year ECE, [5] Assistant Professor

Abstract*— In this project a design of a high throughput crypto devices based on AES algorithm has been presented. The bus width of the architecture is 128 bit. For the 128 bit input 128 bit key is used. The plain text and the key are used as a inputs. Sub Bytes method has been implemented using both composite field method and fixed Rom for further analysis and comparison of performance. By using S-box and key expansion the throughput has been achieved in the range of Gbps. To obtain a high speed pipelining method has been used in this design. The pipelined architecture of the AES algorithm is proposed in order to increase the throughput of the algorithm. The key schedule algorithm of the AES encryption to get the speedup. All the methods have been discussed with a proper screen shots.

***Index Terms*— RIP (Routing Information Protocol), OSPF (Open Shortest Path First), IGRP (Interior Gateway Protocol), Interior Gateway Protocol (IGP)**

I. INTRODUCTION

AES is short for Advanced Encryption Standard and is a United States encryption standard defined in Federal Information Processing Standard (FIPS) 192. AES is the most recent of the four current algorithms approved for federal us in the United States. The Advanced Encryption Standard (AES) is a algorithm used to encrypt and decrypt data for the purposes of protecting the data when it is transmitted electronically. AES is a symmetric encryption algorithm processing data in block of 128 bits. AES is symmetric since the same key is used for encryption and the reverse transformation, decryption. The only secret necessary to keep for security is the key. AES may configured to use different key-lengths, the standard define 3 lengths and the resulting algorithms are named AES-128, AES-192 and AES-256 respectively to indicate the length in bits of the key. The older standard, Data Encryption Standard. DES is up to 56 bits only. To overcome the disadvantages of the DES algorithm, the new standard is AES algorithm. This standard explicitly defines the allowed values for the key length (Nk), block size (Nb), and number of rounds (Nr). Basically the Advanced Encryption Standard (AES) has four blocks. By adding the 128 bit key with 128 bit input plain text is converting into the 128 bit cipher text. The AES algorithm has 10 rounds for a cycle for every round different keys are produced. To obtain a high throughput in a speed of Gbps, pipelining concept introduced in AES algorithm.

II. CRYPTOGRAPHY

Secrecy is at the heart of cryptography. Encryption is a practical means to achieve information secrecy. Modern encryption techniques are mathematical transformations (algorithms) with treat messages as numbers or algebraic elements in space and transform them between a region of " meaning full messages" and region of " unintelligible messages ".A message in meaning full region and input to an encryption algorithm is called clear text and unintelligible output from the encryption algorithm is called cipher text. It disregard the intelligible of a message then a message input to an encryption algorithm is conventionally called plaintext which may or may not be Intelligible for example, a plain text message can be random nonce or a cipher text message; seen such case in some protocols therefore plaintext and cipher text are a pair of respective notions: the former refer to messages input to, and the letter, output, an encryption algorithm.

III. ALGORITHM

The principle design of Advanced Encryption Standard (AES) is based on substitution permutation network, which can take a block of the plaintext and the key as inputs. AES consists of four separate blocks which are repeated for 10 rounds by applying the inputs in several alternative layers to produce the cipher text block. For the first nine rounds all four blocks are repeated but for the final round the Mix Columns block is excluded. Fig 1 shows the basic building block of the AES core which contains four separated blocks, Sub Bytes, ShiftRows, Mix Columns and

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 1, January 2013

AddRoundKey. There is a 32-bit pipelining register in between each of these blocks. This full block is repeated ten times in the AES core to get the whole result.

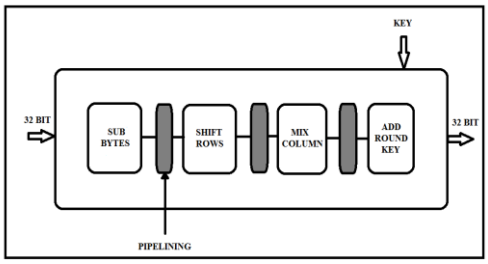

Fig 1. Block Diagram of AES Algorithm

IV. SPECIFICATION

For the AES algorithm, the length of the input block, the output block and the State is 128 bits. This is represented by Nb = 4, which reflects the number of 32-bit words (number of columns) in the State. An implementation of the AES algorithm shall support at least one of the three key lengths: 128, 192, or 256 bits (i.e., Nk = 4, 6, or 8, respectively). Implementations may optionally support two or three key lengths, which may promote the interoperability of algorithm implementations. For the AES algorithm, the length of the Cipher Key, K, is 128, 192 or 256 bits. The key length is represented by Nk = 4, 6, or 8which reflects the number of 32-bit words (number of columns) in the Cipher Key. For the AES algorithm, the number of rounds to be performed during the execution of the algorithm is dependent on the key size. The number of rounds is represented by Nr, where Nr = 10 when Nk = 4, Nr = 12 when Nk = 6, and Nr = 14 when Nk = 8. The only Key-Block-Round combinations that conform to this standard are given in Table.

Table 1. Key-Block-Round Combinations

TYPES	KEY LENGTH (Nk words)	BLOCK SIZE (Nb words)	NUMBER OF ROUNDS (Nr)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

For both its "Cipher and Inverse Cipher, the AES algorithm uses a round function that is composed of four different byte-oriented transformations:

1. Byte substitution using a substitution table(S –box)
2. Shifting rows of the State array by different offsets
3. Mixing the data within each column of the State array
4. Adding a Round Key to the State.

V. SUBBYTES TRANSFORMATION

The SubBytes transformation is a non-linear byte substitution that operates independently on each byte of the State using a substitution table (S-box). There are two approaches to implement the sub byte transform. One is by using look up table (LUT) to get the sub byte value for each input; the other is to calculate the sub byte value by mathematical equations. Due to all the operations are in finite field GF (2^8), there are 256 different sub byte values in total. All the values can be stored in a ROM as a table. When sub byte is in process, the replacement of original value is achieved by look up this table in rom. Therefore, sub byte with LUT is simple to design. If we think of one value only, the calculation method of the transform is slower than the LUT one. However, considering multiple values transform, only one value can be found by LUT at each time which is not suitable for mass data transform. Although multiple tables can be designed in the system, the resource cost is excessive. On the other hand, calculating method is more suitable for mass values transform. Taking vantage of pipeline structure, registers can easily be introduced between logic gates which means as long as the pipeline is full, the transform results can be received continuously at each clock cycle.

In matrix form, the affine transformation element of the S-box can be expressed as:

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 1, January 2013

$$\begin{pmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{pmatrix} = \begin{pmatrix} 1&0&0&0&1&1&1&1 \\ 1&1&0&0&0&1&1&1 \\ 1&1&1&0&0&0&1&1 \\ 1&1&1&1&0&0&0&1 \\ 1&1&1&1&1&0&0&0 \\ 0&1&1&1&1&1&0&0 \\ 0&0&1&1&1&1&1&0 \\ 0&0&0&1&1&1&1&1 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

The S-box used in the Subbytes() transformation is presented in hexadecimal. For example, if=1,1 x {53}, then the substitution value would be determined by the intersection of the row with index '5' and the column with index '3'.

Table 2. S-box values (In hexadecimal)

X \ Y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	1	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	O4	C7	23	C3	18	96	O5	9E	O7	12	80	E2	EB	27	B2	75
4	O9	83	2C	1A	1B	6E	5A	E0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	OO	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	O2	78	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	OC	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	O6	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7E	AE	O8
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	O3	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

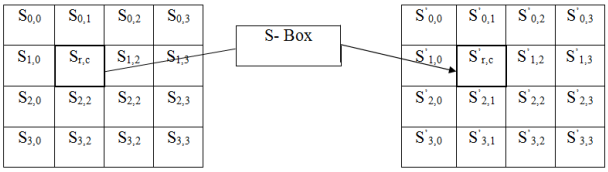

Fig 2. Sub Bytes Transformation

VI. SHIFTROWS TRANSFORMATION

ShiftRows is a simple shifting transformation. First row of the state is kept as it is, while the second, third and fourth rows cyclically shifted by one byte, two bytes and three bytes to the left, respectively. In the InvShiftRows, the first row of the State does not change, while the rest of the rows are cyclically shifted to the right by the same offset as that in the ShiftRows. In the Shift Rows transformation, the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes (offsets). The first row is not shifted at all the second row is shifted by one the third row by two, and the fourth row by three bytes to the left. It is a transposition step on the row of the state where each row of the state is shifted cyclically by a certain number of steps. The first row (row 0) is unaltered. The second row (row 1) is shifted by one byte, the third row is shifted by two bytes and the final row is shifted three bytes.

It also ensures that each byte in each row does not interact solely with their corresponding bytes. There are two schemes to execute the ShiftRows block. The first one is shown in Fig 3 where a mod 4 counter and two 128-bit registers are used. Each of E0 to E15 stands for 8-bit data element.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 1, January 2013

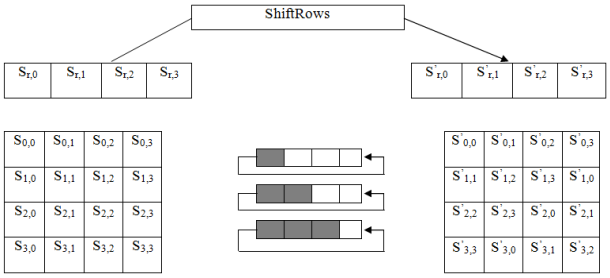

Fig 3. ShiftRows Transformation

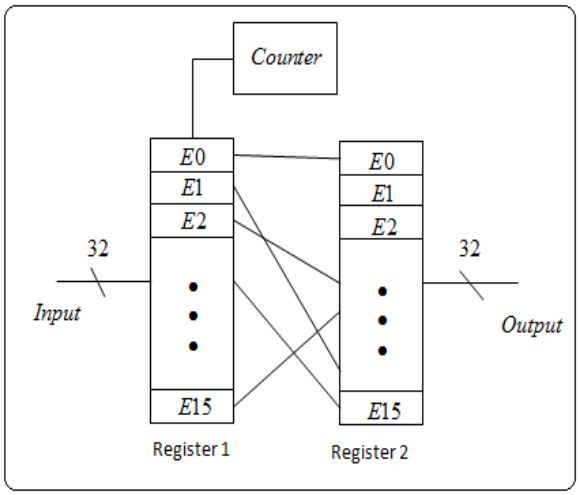

Fig 4. Shift rows block using a counter

The data comes into the ShiftRows block in the form of 32 bits and thus it takes 4 clock cycles to get one set of data. It requires a mod 4 counter to identify which column is coming into the ShiftRows block so that the first column can be marked as 00, the second one as 01, and so on. The data comes into the Register 1 in 4 clock cycles. In the fifth clock cycle, the elements in register 1 would be put into the corresponding position in register 2 according to the principle of ShiftRows. At the same time (the fifth clock cycle) first 32 bits of next 128-bit data would be read into E0-E3 again. At the sixth clock cycle, first 32 bits of the register 2 can be taken out. In general, there need 4 clock cycles to put data into register 1, 4 clock cycles to get out of data from register 2, 1 cycle for "shifting", and 6 clock cycles latency to get the first 32 bit output. So the counter is not only for identifying the data but also for notifying the registers to get in and output data and shift.

VII. MIXCOLUMNS TRANSFORMATION

Mix Columns a mixing operation which operates on the columns of the state, combining the four bytes in each column. Mix Columns can be expressed as modular multiplication with constant polynomials and constant matrix multiplication. We merged the two circuits (MixColumns) into one because inverse MixColumns matrix contains a full MixColumns matrix. Through matrix manipulation it is possible to show that the inverse MixColumn is just addition (XOR) of MixColumns and element matrices. In this merged version, the numbers of XOR logic gates are decreased by 2/3 (from 592 XORs to 195 XORs) with only 2 XOR gates of additional delay. But the question is whether this additional delay is affordable with our high throughput part of fully unrolled version of AES. The comparison shows that the normal MixColumns and inverse MixColumns in separate gives 15% less delay with consuming 30% more area. Therefore it is easy to decide to use this merged version.

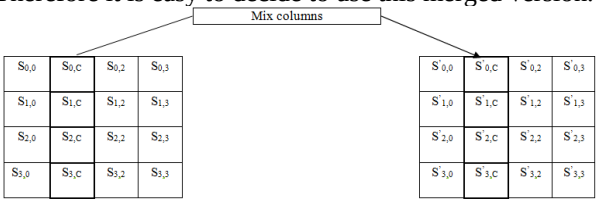

Fig 5. MixColumns Transformation

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 1, January 2013

VIII. ADDROUNDKEY TRANSFORMATION

In this block each byte of states is combined with the subkey where each subkey is derived from the cipher key using key schedule. The subkey and the state are of the same size. The subkey is added by combining each byte of the state with the corresponding byte of the sub key through bitwise XOR manipulation. In the Add Round Key transformation, a Round Key is added to the State by a simple bitwise XOR operation. Each Round Key consists of Nb words from the key schedule. Those Nb words are each added into the columns of the State, such that [wi] are the key schedule words, and round is a value in the range 0 round Nr. In the Cipher, the initial Round Key addition occurs when round = 0, prior to the first application of the round function. The application of the Add Round Key transformation to the Nr rounds of the Cipher occurs when 1<round <Nr. The action of this transformation is illustrated in Fig. 3.6, where l = round * Nb.

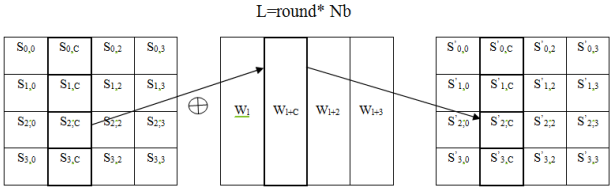

Fig 6. Add Round Key Transformation.

IX. KEY EXPANSION

128 bit key is taken as input in this block and expanded for all the rounds and stored .The keys are then used for every round. Key schedule part is dependent on the sub bytes. The sub byte is calculated both using composite filed and LUT (look up table) method. The LUT is definitely not area efficient rather time efficient whereas the composite field sub bytes technique is just opposite. The area delay curve comes up with the right solution to be chosen. Without making any decision beforehand on which key expansion should be used, both the key expansion have been used in the core AES in different combination with different sub bytes. This makes it easy to analyze the performance of each combination. The key expansion in total takes 12 clock cycles to be completed but data encryption is possible to start right after 4th cycle because of the availability of first few round's keys. All the statistics are shown in table 3.

Table 3. The Statistics of Key Expansion Block

METHOD	256 x 8 BIT ROM	XORs	SLICES	MINIMUM FIELD
MEMORY BASED	40	1312	5092	6.279ns
COMPOSITE FIELD	NO	3790	2335	22ns

X. PIPELINING

Basically pipelining means to process the data that is given as input in a continuous manner without having to wait for the current process to get over. This pipelining concept is seen many processors. In the architecture in figure the register are used to store the current output of the round that is being executed. Now instead of passing the output of each round to the passed at current round's value is stored in the next input to current round can directly be given as soon as the current output is obtained. And the input to next round is given from the register thus avoiding a direct contact between the two rounds. This is not possible in the iterative looping architecture because the next input can be given only when the whole round based processing is over since the same hardware is used over again in the process of obtaining the cipher text. Thus, the pipelined architecture increases the speed of execution for obtaining the chipper text but at a cost of increased hardware. In the substitute bytes we use a look up table based S-box. This contributes for the some of the hardware in the form of block RAMs. With the help of a search based look up table (LUT) we can reduce the hardware cost to a considerable extent. The need for Number of block RAMs which are important resources in any chips are completely eliminated when a search based S-box is used in AES pipelined algorithm. There is 2% consumption of block RAMs in iterative architecture without search based s-box and AES pipelined architecture on a vertex 5 board. Also the Number of fully used Bit Slices is substantially reduced in *AES* pipelined architecture with a search based memory which is even lower than in the iterative

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 1, January 2013

architecture. The input/output device utilization is constant in all the three architectures. All other devices such as slice registers, flip-flops and LUTs are understandably lower in case of iterative architecture. Thus with the help of search based s-box in AES pipelined algorithm some of the key resource utilization is reduced. Trade-off between iterative architecture and pipelined architecture and at the same time ensuring the tremendous throughput as in pipelined feature.

XI. SUB PIPELINING

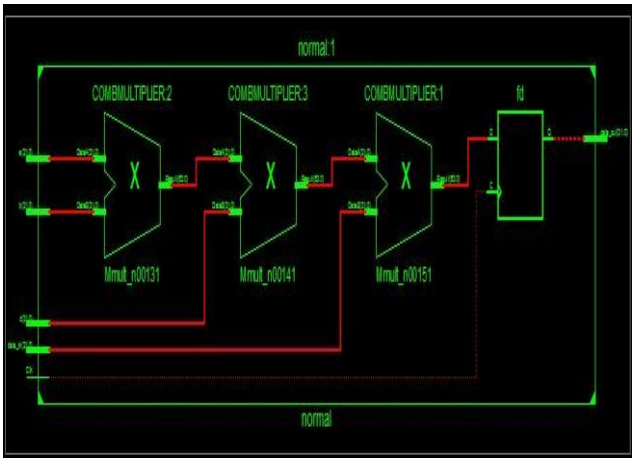

Fig 7. Waveform for normal code-no delay at all

As you can see, the normal code is implemented by connecting 3 multipliers in a cascaded fashion with a flip flop at the end stage. For the pipelined code, we have flip flops after each multiplier. What does this mean? The extra flip flops reduce the delay through the combinatorial logic and hence pipelined code can operate at a higher frequency than the normal code.

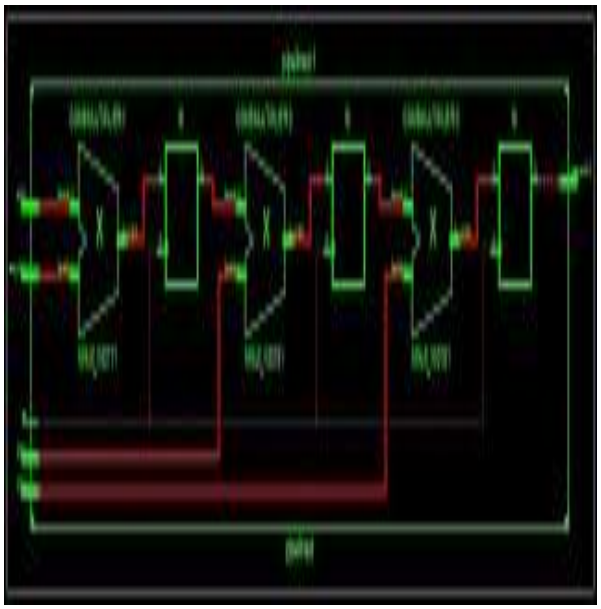

Fig 8. Waveform for pipelined code-3 clock cycle delay

The 'normal' code takes less time to write and is mostly straight forward. But if you want your design to offer the highest speed possible, you have to think out of the box! The 'pipelined' code is little bit complicated to write. In this case we had to use case statements and a for loop to implement a small equation. But it gives higher speed. In large projects pipelined designs are very important for some blocks since it may act as a bottleneck for the performance of the whole design. They introduce a small number of delay between input and output, in terms of clock cycle. For instance we have 3 stages in the pipelined code and hence the output comes only after 3 clock cycles, after the input is applied.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 1, January 2013

XII. RESULTS

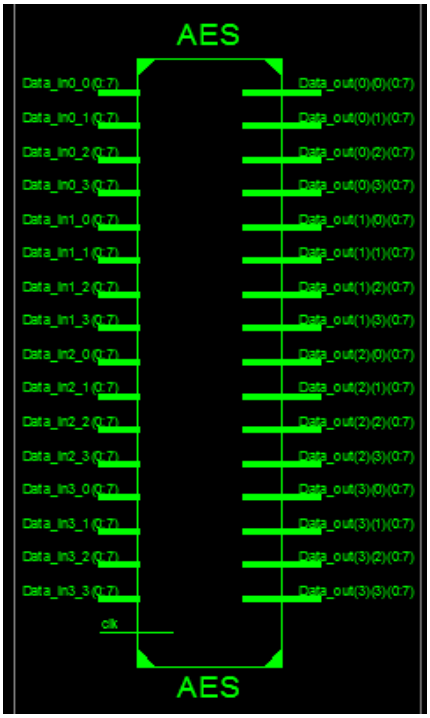

Fig 9. RTL view AES core

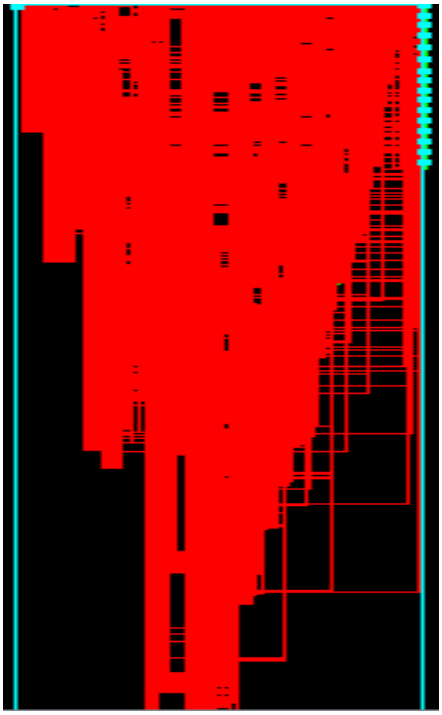

Fig 10. Schematic view of AES crypto core

XIII. CONCLUSION

The detail analysis on using different SBox method has made the decision easy to follow the right one. The speed has been increased substantially through sub pipelining in the Sub Bytes block. In the ShiftRows part, comparatively simpler method is followed to implement. However, further research can be done to implement it by using an improved shift-register. Pipeline and sub-pipeline structure has increased the maximum frequency significantly. The average delay in pipeline should be controlled by sub-pipelining to reduce the maximum delay to average latency time. In this project, the maximum frequency increased from 35.411 MHz to 103.061MHz by inserting 4 registers. Therefore, taking the vantage of pipeline design AES can be implemented in FPGA for high throughput purpose. Comparing with a design without sub-pipeline, the sub-pipeline design improves the performance remarkably.

REFERENCES

[1] Ahmad Karim, Minhaj Ahmad Khan (2011). "Behavior of Routing Protocols for Medium to Large Scale Networks", Australian Journal of Basic and Applied Sciences, 5(6): 1605-1613.

[2] Bertsekas.D and R. Gallager (1992), Data Network. Prentice-Hall.

[3] Bruno, Anthony & Kim, Jacqueline (2005), CCDA Self-Study: RIP, IGRP, and EIGRP Characteristics and Design .Retrieved.

[4] Bernard Fortz, Jennifer Rexford and MikkelThorup.(2002), Traffic Engineering With Traditional IP Routing Protocols." IEEE Communications Magazine.

[5] Cisco EIGRP: Enhanced Interior Gateway Routing Protocol Overview (2005). Retrieved

[6] Davis, David. (2005) Cisco administration 101: What you need to know about EIGRP.

[7]Guang Yang, "Introduction to TCP/IP Network Attacks" white paper available a seclab.cs.sunysb.edu/sekar/papers/netattacks.pdf.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 1, January 2013

[8] Garcia-Lunes-Aceves and Murthy.S,(1995) "A Loop-Free Path-Finding Algorithm: specification, Verification and Complexity," in Proceedings of the IEEE INFOCOM.

[9] IP Routing: From Basic Principles to Link State Protocols. Retrieved November 27, (2005).

[10]Ittiphonkrinpayorm and SuwatPattaramalai, (2012)"Link Recovery Comparison between OSPF & EIGRP ", International Conference on Information and Computer Networks (ICICN 2012) IPCSIT vol. 27 (2012) IACSIT Press, Singapore.

[11] Malkin.G. (1998) Enhanced Interior Gateway Routing Protocol. (2005).

[12] Mehboob Nazism Shehzad, Najam-Ul-Sahar, "Simulation of OSPF Routing Protocol Using OPNET Module"(A Routing Protocol Based on the Link-State Algorithm).

[13] Malkin.G, (1998) "Routing Information Protocol Version 2," SRI Network Information Center, RFC 2453.

[14] McQuillan, G. Falk, and I. Richer,(1978) "A Review of the Development and Performance of the ARPANET Routing Algorithm," IEEE Transactions on Communications, vol. 26, no. 12, pp. 1802–1811.

[15] Mittal.V and G. Vigna,(2002) "Sensor-Based Intrusion Detection for Intra- Domain Distance-Vector Routing," in ACM CCS's 02.

[16] Nordstr¨om and Dovrolis.C (2004), "Beware of BGP attacks," SIGCOMM Computer. Communication. Rev., vol. 34, no. 2, pp. 1–8.

[17] Office of the President of the United States, "Priority II: A National Cyberspace Security Threat and Vulnerability Reduction Program," 2004.

[18] Pethe.R.M., Miss S.R.Burnase (2011) TECHNICAL ERA LANGUAGE OF THE NETWORKING - EIGRP International Journal of Engineering Science and Technology (IJEST) NCICT Special Issue.

[19] Perlman.R, (1988) "Network layer protocols with byzantine robustness," Ph.D. dissertation, MIT Lab. for Computer Science.

[20] Pei.D, X. Zhao, L. Wang, D. Massey, A. Mankin, F. S. Wu, and L. Zhang,(2002) "Improving BGP Convergence Through Assertions Approach," in Proceedings of the IEEE INFOCOM.

[21] Pei.D, D. Massey, and L. Zhang,(2004) "A Framework for Resilient Internet Routing Protocols," IEEE Network Special Issue on Protection, Restoration and Disaster Recovery.

[22] Postel.J, "User Datagram Protocol, (1980)" SRI Network Information Center, RFC 768.

[23] Padmanabhan.V.N. and D. R. Simon,(2002) "Secure Trace route to Detect Faulty or Malicious Routing," in Proceeds of HOTNETS.

[24] Rajesh. R, Lakshmanan. M and Noor Mohammed (2011), ": Implementation of Networked Control Systems using TCP/IP", International Journal of Computer Applications 18(2):1-5.

[25] Rekhter.Y, Li.T, and Hares. S (2006), "A Border Gateway Protocol 4 (BGP- 4)," RFC 4271 (Draft Standard), Internet Engineering Task Force.

[26] Rekhter.Y, "Experience with the BGP Protocol," (1991) RFC 1266 (Informational), Internet Engineering Task Force.

[27] Rekhter.Y and T. Li,(1995) "Border Gateway Protocol 4," SRI Network Information Center, RFC 1771.

[28] RIP Version 2. Retrieved November 27, (2002), from http://rfc.net/std0056.html Osterloh, Heather.

[29] Resource Reservation Protocol. (2002). Retrieved November 27, 2005.

[30] Smith.B. R, S. Murphy, and J. J. Garcia-Luna-Aceves, (1997) "Securing distance-vector routing protocol," in Global Internet'96.

[31] Terzis.A , K. Nikoloudakis, L. Wang, and L. Zhang,(2000) "IRLSim: A General Purpose Packet Level Network Simulator," in Proceedings of the 33rd ACM-SIAM Symposium on Discrete Algorithms..

[32] Vishal Sharma, Rajneesh Narula and Sameer khullar (2012) "Performance Analysis of IEEE 802.3 using IGRP and EIGRP Routing Protocols" International Journal of Computer Applications (0975 – 8887) Volume 44– No13.

[33] Wu, Bing, "Simulation Based Performance Analyses on RIP, EIGRP and OSPF Using OPNET".

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 2, Issue 1, January 2013

[34] YoungmiJoo, VinayRibeirot, AnjaFeldmannt, Anna C.Gilbertand Walter Willinger Guang Yang,", "TCP/IP traffic dynamics and network performance: A lesson in workload modeling, flow control and trace-driven simulations", ACMSIGCOM.

AUTHOR BIOGRAPHY

S.LOKESHKUMAR is doing final B.E year Electronics and communication His area of interest is VLSI, Image Processing, and embedded System.

A.Sriram is working in SNS College of engineering, Coimbatore as Assistant professor in the department of ECE. He received the M.E degree in communication Systems from Cape Institute of Technology, Tirunelveli under Anna University Chennai in 2012 and B.E degree in Electronics and Communication Engineering from Cape Institute of Technology, Tirunelveli under Anna University, and Chennai in 2009. He had presented 3 national conferences in various fields and published 2 papers in international Journals. He has done the project in the area of embedded systems, VLSI. His areas of interests are Digital communication, Micro-Processor, Micro-controller and Digital Electronics.

.