International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-2, December, 2019

Implementation of AES for Encryption in Vertex- 3 of FPGA Environment for Security

B. Satyanarayana, M. Srinivasan

Abstract - Data transmission with protection is main concept which is getting demand now a days for which number of encryption of data techniques are developed and now in this paper Advanced Encryption Standard (AES) Algorithm is used and is implemented on FPGA kit using vertex-3 family. We use 128 bits consists of input, key data, output data for this design. It is called an iterative looping with replacement box, key, loop in this design for both encryption and decryption of data. We use Xilinx software platform for simulation of our design that is AES by which area utilization and throughput is increased for achieving low power consumption, high data security, reduced latency and easy architectural design. This data operation is applicable in many areas.

Keywords - AES, encryption, decryption, Latency, FPGA, Throughput.

I. INTRODUCTION

As we are familiar that digital world requires data transmission for information exchange where security and time deficiency are major two assets which never let go of deficiency. for overcoming this problem number of encryption techniques are developed among which AES (Advance Encryption Standerd) is the best efficient before which DES was present in which breaking the code was very easy and simple because its key length is very small which is of 56 bits only. So, this stood main reason for developing algorithm for AES called by National Institute of Standards and Technology (NIST) in 1997, immediately after call 15 applications were introduced with algorithms from which Rijndael algorithm stood as a winner. The key size is 128 bits as minimum where as 256 bits for maximum. There is one more advantage that it can be presided and transmitted as 32 bits of multiple data. So, ther is difficulty in key breaking in AES. So, we also call AES algorithm a Rijndael algorithm. In this we have fixed length of block i.e., 128 bits and length of key is 128, 192 & 256 accordingly varying.

For additional blocks and key sizes we can handle this Rijndael algorithm, anyhow the AES doesn't adopts the additional features. We can use both hardware and software for implementing this AES algorithm resourcefully. As the requirement increases such as high speed, high volume of communicational security is combined with physical security. Whereas for implementation in software very less expenditure is required with resources but we can say it's a slow process.

Revised Manuscript Received on December 30, 2019.
* Correspondence Author
B.Satyanarayana*, Department of ECE, Sri Satya Sai University of Technology and Medical Science, Sehore, Bhopal, Madhya Pradesh, India,
Dr M.Srinivasan, Department of ECE, Sri Satya Sai University of Technology and Medical Science, Sehore, Bhopal, Madhya Pradesh, India

American Standerd Codes for Information Interchange (ASCII) standards are there to maintain this encryption data where there are some fixed rules to be followed. Till today ASIC's were used or GPPs are used for implementation but there is much faster hardware is developed than these and that is FPGA. This has wide applications and faster solutions in recent technology. We can use functionally wide range support configurations which is supported by reconfigurable devices. In our paper we only deal with AES algorithm for both encryption and decryption of data by using 128 bits as key size for implementation. It is also using approach of iterative loops and LUTs for S-Box implementation or replacement. We use this AES design in this paper to achieve low power consumption, low latency, high throughput and area efficiency utilization on FPGA vertex kit.

II. LETERATURE SURVEY

The permutation network substitution is our AES proposed algorithm. The main concept is to convert simple data into a cipher text data which is arrange jumbled. Whereas decryption means converting cipher text complex data into simple text data form which is original form that is plain text. The needed data from the text can be encrypt or decrypt as needed because the AES is a symmetric cipher block. For performing this encryption and decryption we use only limited or less number of cycles for each 128 bits used for iterative looping approach.

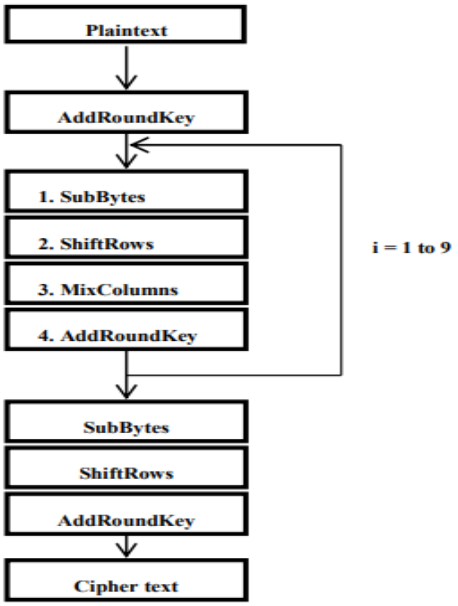

Figure.1: AES OPERATION

Encryption and decryption are two fundamental operations in AES algorithm functions.

III. ENCRYPTION OPERATION

We execute the AES algorithm for N-1 times of loops it has a fixed size of blocks 128 bits and this N is represented as length of key depended rounds that seems varying every time such as 128, 186, 256,512 bits in rounds and key length respectively. The major parameter N is having 10, 14 or 12 rounds which is dependent on the size of keys. There is a difference between last and first round of algorithm is varying from remaining all in AES algorithm. We don't perform the mixed column operation at the last but at the beginning of initial round the add round key of AES algorithm is added. The iteration count is of total 10 for loop to the length of key 128 bits but the size of key and block in our proposed paper is of same size. There are 4 transformations for the AES encryption to undergo.

Transform Of Round Key

Ciphers add round key is similar exactly to the general add round key. As the operation of XOR is inversed to itself it matches to the cipher text add round key. For both the transmission operations such as encryption and decryption the architecture is same as modules of encryption. Depending on the type of work modules the decryption is done which is on speed and size of physical environment basis here we have three architectures which are totally different. To the module of encryption the key Expansion is same without ant change we consists of these modules such as:

1. Block of input
2. Expansion of key
3. Block of inverse data
4. Block of output
5. Controller.

The expiration of input key block then our output block and controller will be similar but the changing block in these is the data block. It's an inverse of operation of transform of data blocks which happened in inverse block of data. As shown in below figure the basic iterative architecture of flow diagram is used to explain the decryption blocks similarity module decryption. We can avoid the complexity of implementation of hardware can be neglected with this approach and in a single cycle block we can do the calculations of S-box with which latency is reduced.

Table.1: S Box Table

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6	90	d8	ab	0	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

IV. PROPOSED ARCHITECTURE OPTIMIZATION

The Rijndael algorithm has a straightforward structure and can be executed effectively on a wide scope of microprocessors, a significant element when a quick software module is required and when the algorithm is coded for implanted platforms. There are a ton of regularities in the design of Rijndael algorithm.

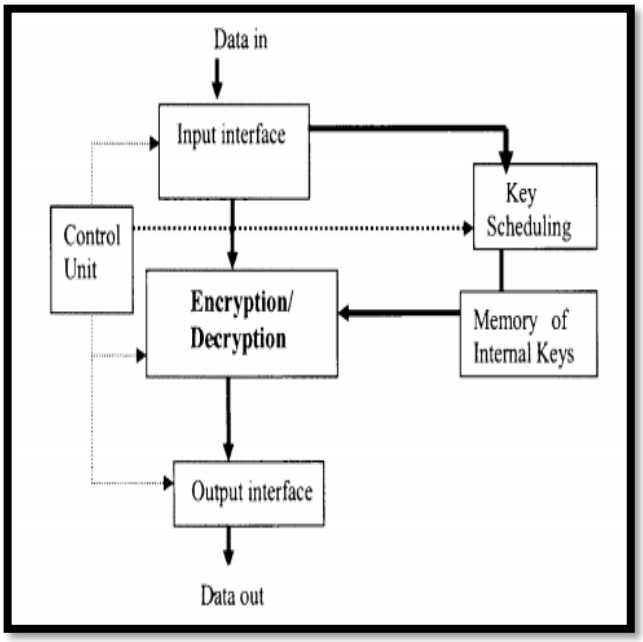

Figure.2: Block diagram of hardware implementation of block cipher

Consequently the basic way just as the general area can be minimized, with impeccable VLSI design. In this area, the four Rijndael transformations concentrating on their conceivable hardware acknowledgment is introduced. Since the Add Round Key transformation is a straightforward bit insightful XOR between the round key and the cipher state, the hardware usage comprises of 128 two-inputs XOR doors.

The Shift Rows transformation comprises of a permutation of the bytes inside the cipher State. Thus, it doesn't expend any silicon area and is adequate to switch the wires relating to the right bytes. So as to design the hardware cryptosystem, there were two potential architectures: iterative architecture or pipelined architecture. The iterative architecture is increasingly worried about the size of the physical hardware hence relinquishing the speed. Iterative architecture requires an output from past advance as contribution for the following stage. The pipelined architecture can build the speed of encryption/decryption by processing numerous blocks of data at the same time. It is acknowledged by inserting rows of registers among combinational rationale. Portions of rationale between two continuous registers structure pipeline stages. Every pipeline stage is one round unit for this situation. During each clock cycle, the in part processed data moves to the following stage and its place is taken by the resulting data block. Much of the time it is critical to create versions that maximize execution for minimum cost. In this work, it is planned to accomplish a high speed in throughput, so the pipelined architecture was chosen. The appropriateness of different design focuses depends intensely on how the cipher will be utilized. The full blended inner-and outer-round pipelining ought to be the architecture of decision for contrasting hardware execution of the AES up-and-comers in non-criticism cipher modes.

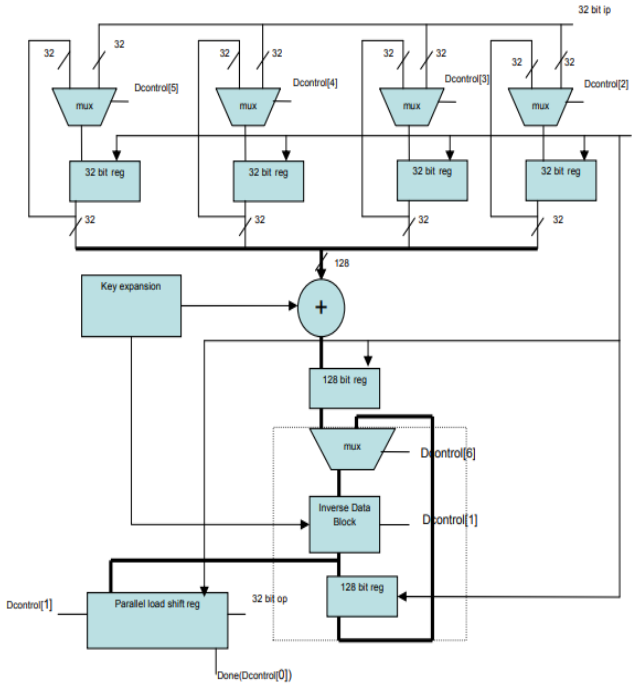

Figure.3: AES Algorithm Architecture

V. AES DECRYPTION

For encryption operation its whole inverse the decryption. In reverse operation we can obtain the text of plain which is original by decryption to the cipher text of encryption. As the schedule of key is same we apply the backwards operation to all. Only sub bytes, inverse row shift and inverse column mix operations is remain to implement, where as the specific add round key remains same.

Table.2: Inverse S-Box Table

Inverse S-box

x \ y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

AES key operation

For generating a scheduled key the expansion of cipher key routine operation is done taken by AES algorithm. Total of Nb (Nr+1) words are generated with the key expiation. key data of Nb words are required for N operation of rounds where the Nb is required by the algorithm only initial set of words. The key schedule expansion of the input keys in according to the pseudo code requires process of 4 bytes word. Basically we need two things of expiation key operation that is key of output expanded and input cipher key. For stretching a short to a long key the main responsible thing is key scheduling we have different sizes of each key expiation such as 128,192 and 256 bits which is cipher key dependent.

By using the XOR of 32 bits, S- Box and rotation of cyclic byte we implement the key expansion operation mostly. In the AES algorithm another main segment is key management which contains distribution of key, backup and key storage, change of key and disposal of key.

VERTEX-3 FPGA IMPLEMENTATION

We implement this AES algorithm on Vertex-3 FPGA design by verilog algorithm the family if vertex version is XC3S1600E in Xilinx ISE 14.7 design tool. We use 128 bit AES algorithm length of text plane as input, but only when inputs are 116 bytes then only we can sort out the bit plane. We treat the 8 bits data as single entity sequence in AES. Byte is the only processing unit for AES algorithm. From the keyboard we give the input to this design which will be in hexadecimal form encoded. The figure as shown below gives the experimental setup. Sending the program of plain text in input block is by using the visual basic design help which is having 4096 bytes of data with key data cipher of vertex-3 version with device of number family XC3S1600E-FG320-5. For putting the graphical data together we use the visual basics which is very efficient for graphical user interface and it also can give the faster program of windows than a C program.

For both the operations of encryption and decryption of AES algorithm we use the FPGA kit then by using language of verilog coding is done.

The architecture of AES is implemented on FPGA with Vertex-3 of EDA tool Xilinx. To the parallel port of pc the parallel cable of FPGA is interfaced by using a connector of on board JTAGA which is provided for configuring. Whereas for interfacing vertex and pc we use serial port and serial cables. O the FPGA input is sent with cipher text in serial manner from PC to FPGA.

VI. RESULT

American Standard Code for Information Interchange (ASCII) data which is of 4096 bytes is sended to the Vertex-3 of 1600E-FS320-5 version as input. But as a single entity it takes the serial data transform all at time after which only 16 bytes is transformed to FPGA then encryption of AES algorithm is done to first 128 bits data next to remaining 128 bits data till it reaches the 4096 bytes count and we implement the whole text of data on Vertex-3 1600Efs320-5. We use this terminal of TMFT for data transform in FPGA which is by using UART of 4096 bytes data for transfer that acts as interface for computer and vertex-3 kit. In terminal of TMFT we can see both input and output software at a time.

Table 3: Parameters

Kit usage				
Logic utilization		**Used**	**Availability**	**Utilized**
Number of bonded IOBs	Encryption	6	250	2%
	Decryption	6		2%
Number of BRAMs	Encryption	16	36	44 %
	Decryption	20		55 %
Number of GCLKs	Encryption	2	24	8%
	Decryption	2		8%

By text input cipher of decryption we can use the same key of cipher text for algorithm of encryption also. Where we do the decryption on our personal computers, when we get decryption output if decryption is done then it is called original decryption data. As shown in below figure the exact data is obtained after 10 rounds of rotation of AES algorithm. The implementation design is as shown in figure below which is vertex-3 FPGA kit.

Figure.4: Vertex-3 Kit

Before the calculation of first design round the each round key completed with its clock cycle which is in single clock cycle.

VII. SYNTHESIS AND POWER DISTRIBUTION REPORT

Verilog code is used for our design and we implement it on vertex-3 0f 1360 FPGA design with high scope of simulation and efficient implementation in Xilinx 14.7 version with synthesized process. We implement the algorithm of AES on Vertex-3 which can efficiently do baud rate calculation, efficient area utilization, high throughput and major is memory. The table below shows the hardware overall utility criteria of FPGA where as timing report is nothing but the device utilization time or simply it defines the latency.

Table 4: Utilization of Resources

Target FPGA device	Virtex-5 XC5VLX50
Encryption throughput	3.09 Gbps
Timing Report	
Speed grade	-3
Max. clock frequency	242.153 MHz
Min. period	4.130 ns
Min. input arrival time before clock	2.254 ns
Max. output required time after clock	2.622 ns
Device Utilization	
Number of Slice LUTs	5256 / 28800 18%
Number of occupied Slices	1745 / 7200 24%
Number of fully used LUT-FF pairs	260 / 5258 4%
Total equiv. gate count for design	44689
Block RAMS	Zero

The baud rate for both encryption and decryption is same, whereas allocation of memory is different with totally different gain in throughput.

Table 5: Timing Report

Designs		**Our Design**
FPGA Vendor		Xilinx
FPGA Chip		Spartan 3E 1600
Baud Rate		9600
Memory	Encryption	179720 Kbytes
	Decryption	183816 Kbytes
Throughput (Mbps)	Encryption	2269 Mbps
	Decryption	1132 Mbps

Table.6: simulation report

kit utilization				
Logic utilization		**Used**	**Availablity**	**Utilized**
Number of Slices	Encryption	947	14752	6%
	Decryption	1170		7%

Number of Slice Flip Flops	Encryption	829	29504	2%
	Decryption	628		2%
Number of 4 Input LUTs	Encryption	1589	29504	5%
	Decryption	2252		7%

As we are comparing with different authors results of practical type should be satisfying. Test bench operation is used to test the system. For activating the system a test bench is applied by decryption or encryption of input pulse. 197 FIPS is designed which is a simple vector to test. Here by using this throughput is high and very low usage of memory is achieved than others.

VIII. CONCLUSION

The AES definitely uses the cipher text algorithm and data of length 128 bits each. We do implementation of this design on Xilinx 13.1i software platform and vertex-3 family is used this AES arithmetic implementation represents the complexity reduction operation where as within single cycle number of positive outputs are transmitting means throughput is increased and simultaneously can say latency decreased which gives high performance architecture. This algorithm achieved throughput for encryption is 3000 Mbps i.e., the rate of encryption where as 1200 Mbps is for Decryption i.e., rate of decryption. Even memory is optimized in our design as well as power consumption also which is because of less area utilization on the FPGA kit where there is a feature scope. Its better applicable in any situation such as defense, smart cards which is for top level security purpose.

REFFERENCES

1. Ahmad, N.; Hasan, R.; Jubadi, W.M; "Design of AES S-Box using combinational logic optimization", IEEE Symposium on Industrial Electronics & Applications (ISIEA), pp. 696-699, 2010.
2. Mr. Atul M. Borkar, Dr. R. V. Kshirsagar and Mrs. M.Vyawahare, "FPGA Implementation of AES Algorithm", International Conference on Electronics Computer Technology (ICECT), pp. 401-405, 2011
3. G. Rouvroy, F. X. Standaert, J. J. Quisquater, J. D. Legat, Compact and efficient encryption/decryption module for FPGA implementation of the AES Rijndael very well suited for small embedded applications, Procedings of the international conference on Information Technology: Coding and Computing 2004 (ITCC 2004), pp. 583 – 587, Vol. 2, April 2004
4. [8] K. Chapman, PicoBlaze 8-bit Microcontroller, Xilinx, 2002http://www.xilinx.com/products/design_resources/proc_ central/ grouping/picoblaze.html
5. [9] N. Pramstaller and J. Wolkerstorfer, A Universal and efficient AES coprocessor for Field Programmable Logic Arrays, FPL 2004, LNCS Vol. 3203, pp. 565-574, SpringerVerlag, 2004
6. P. Chodowiec, K. Gaj, Very Compact FPGA Implementation of the AES Algorithm, Cryptographic Hardware and Embedded Systems (CHES 2003), LNCS Vol. 2779, pp. 319 – 333, Springer-Verlag, October 2003.

AUTHOR PROFILE

B. satyanarayana, is a research scholar at SSSUTMS University, Bhopal, India. Has done number of publications in national, international & scopus journals with conferences with having 12 years experience as assistant professor.

Dr. M. Srinivasan, working as Associate professor in SSSUTMS with more than 18 years experience having interest in VLSI. Had published more than 14 papers in nation, international and scopus journals.