ISSN No:-2456-2165

FPGA Implementation of AES Key Expansion Algorithm in Fully Pipelined and Loop Unrolled Architectures

Kiran P V, Naveen Kumar Kanavi
Department of Electronics and Communication Engineering,
Proudhadevaraya Institute of Technology,
Hosapete, India

Abstract:- This paper has an aim of comparing the implementation and performance of key expansion algorithm of Advanced Encryption Standard (AES) in fully pipelined and loop unrolled modes. Using Xilinx 14.1 ISE for simulation and synthesis, fully pipelined mode achieves a throughput of 51.65 Gbps. The loop unrolled mode achieves throughput of 20.84 Gbps and has very less device utilization and is suitable for implementation on low end FPGAs.

Keywords:- *AES, Key Expansion, Fully Pipelined, Loop Unrolled, Throughput.*

I. INTRODUCTION

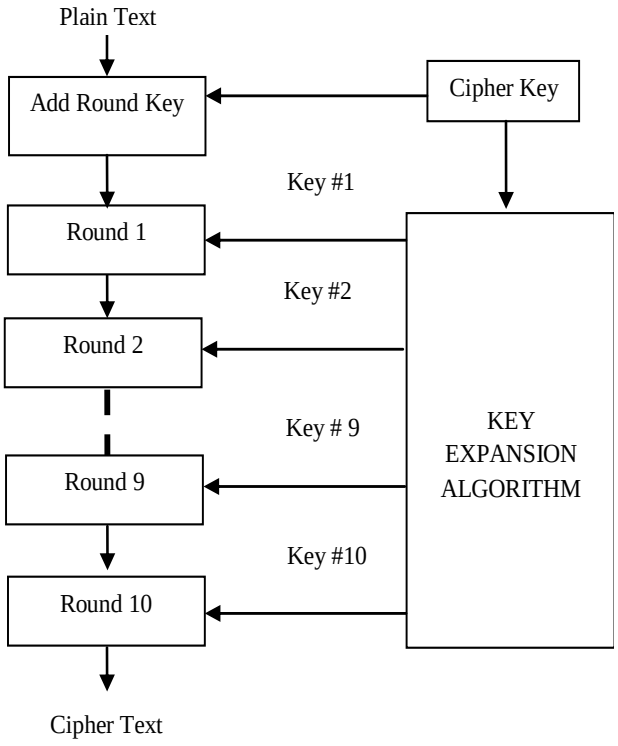
Modern digital communications rely on block cipher encryption techniques to ensure data integrity. Advanced encryption standard algorithm is one such algorithm that has proven its effectiveness in that direction. The algorithm processes data in three different lengths namely 128, 192 and 256 bits and makes use of same length cipher key to produce encrypted data [1]. One of the key aspects of the algorithm is key expansion through which encryption process goes through many iterations before producing the cipher text. The synthesis and simulation of the algorithm can be done on logic devices such as Field Programmable Gate Arrays (FPGA) because of the reconfigurability and flexibility they offer. FPGA implementation of the key expansion algorithms can be done using two architectures such as Loop-unrolled, pipelined techniques [2], [3], [4]. The first technique loop unrolled provides area efficient design where as second method provides high throughput at the cost of hardware area.

In this paper we present the possibilities of implementing the Key expansion algorithm with loop unrolling and pipelined architectures on Xilinx FPGA and compare the performances of both architectures in terms of area occupation and speed. The paper is organized as follows: Section 2 describes the AES encryption and key expansion algorithms. Section 3 describes the architectures for both pipelined and loop unrolled techniques. Section 4 describes implementation of both architectures on FPGA. Section 5 presents results and comparisons. And last section presents conclusions and scope for extension of the implementation.

II. AES ENCRYPTION AND KEY EXPANSION ALGORITHMS

A. AES-128 Encryption Algorithm

The Encryption has nine rounds of similar transformations and tenth one with slight modifications. The original cipher key undergoes initial transformation before it is given to first round. An encryption round has four sequential transformations internally called **SubBytes, ShiftRows, MixColumns and AddRoundKey** Transformations [1]. Last round excludes MixColumns Transformation. The text from each round is passed to the next round to undergo transformations. Each Round makes use of different key. Cipher Text is produced after tenth round as shown in Fig. 1.



Encryption Part

Fig. 1: AES 128 Encryption block diagram

ISSN No:-2456-2165

B. Key Expansion Algorithm

The AES algorithm makes use of a Cipher Key for Key Expansion. A new key is formed after every key expansion in the form of words. 40 words are generated by Key Expansion. The key expansion performs operations **Temp, Subword(), Rotword(), Rcon[i], w[i-Nk]** in each round [1].

The function **SubWord()** uses the S-box to replace four bytes with new bytes. The S-Box is shown in the Fig. 2.

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	01	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	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	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	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	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: S-box: Substitution values for the byte xy

The function **RotWord()** uses the word {b0,b1,b2,b3} as input, performs a cyclic rotation, to produce a word {b1,b2,b3,b0}. The **Rcon[i]** is an array of constant values. From figure 3, w[**i**], is XOR of the previous word, **w**[**i-1**] and **w**[**i-4**]. The following example shows the details.

Cipher Key = 2b7e1516 28aed2a6 abf71588 09cf4f3c
Where w[0] = **2b7e1516,** w[1] = **28aed2a6,** w[2] = **abf71588,** w[3] = **09cf4f3c**

Fig.3, shows the results after initial round.

i (dec)	temp	After RotWord()	After SubWord()	Rcon[i/Nk]	After XOR with Rcon	w[i-Nk]	w[i]= temp XOR w[i-Nk]
4	09cf4f3c	cf4f3c09	8a84eb01	01000000	8b84eb01	2b7e1516	a0fafe17
5	a0fafe17					28aed2a6	88542cb1
6	88542cb1					abf71588	23a33939
7	23a33939					09cf4f3c	2a6c7605

w5=88542cb1, w6=23a33939, w7=2a6c7605, the key 1 is formed by concatenating the above four words.

Key1={w4,w5,w6,w7}=a0fafe1788542cb123a339392a6c7605

The algorithm performs same calculations for another nine times to generate remaining keys [1]. The Key generated from one round is used for generating next key. The block diagram of entire algorithm is shown in Fig. 4.

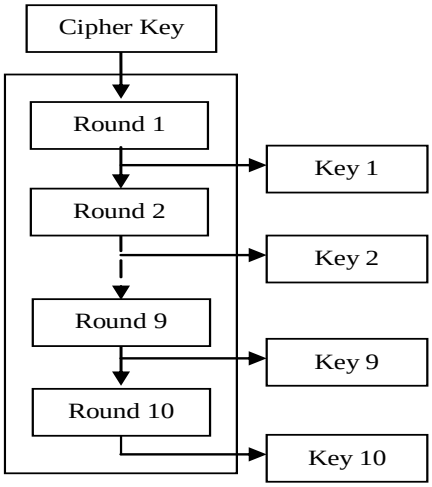


Fig. 4: Key Expansion Algorithm

III. ARCHITECTURES FOR KEY EXPANSION ALGORITHM

A. Fully Pipelined Architecture

In fully pipelined architecture storage registers are included between key expansion rounds to store the output from a round. The contents of the register are then given as an input to next key expansion round. The architecture contains replication of a round many times and uses lot of silicon area while providing a good throughput. The circuit produced for a round is active only during the key expansion for that round and remains inactive rest of the times. Hence there is a lot of wastage of silicon area. The architecture is shown in Fig. 5.

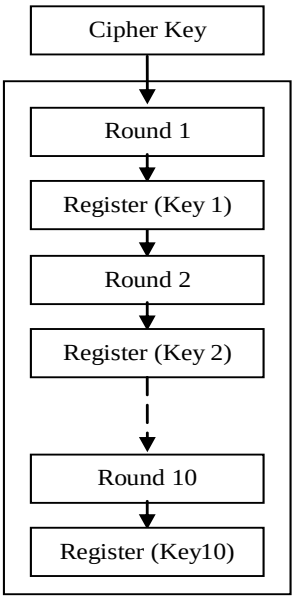


Fig. 5: Pipelined Architecture for Key Expansion

B. Loop Unrolled Architecture

The loop unrolled architecture has an objective of reducing silicon area by replacing all similar rounds by only one round. The output from the round is again given back to the same round with the help of key selection mechanism that act something like a multiplexer. This architecture saves enormous amounts of memory as ten similar rounds are replaced by a single round at the same time maintaining similar throughput. The Fig. 6, shows working principle of this architecture.

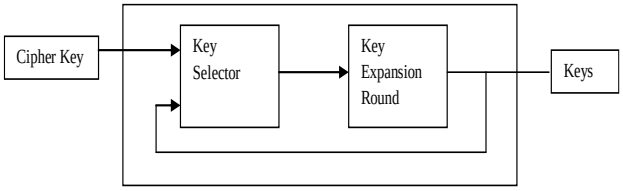


Fig. 6: Loop Unrolled Architecture for Key Expansion

IV. IMPLEMENTATION OF BOTH ARCHITECTURES IN XILINX FPGAS.

A. Fully Pipelined Architecture

The fully pipelined architecture was implemented in VHDL and synthesized and simulated in Xilinx 14.1 ISE Design Suite. The architecture has ten rounds and registers in between as shown RTL schematic in Fig.7.

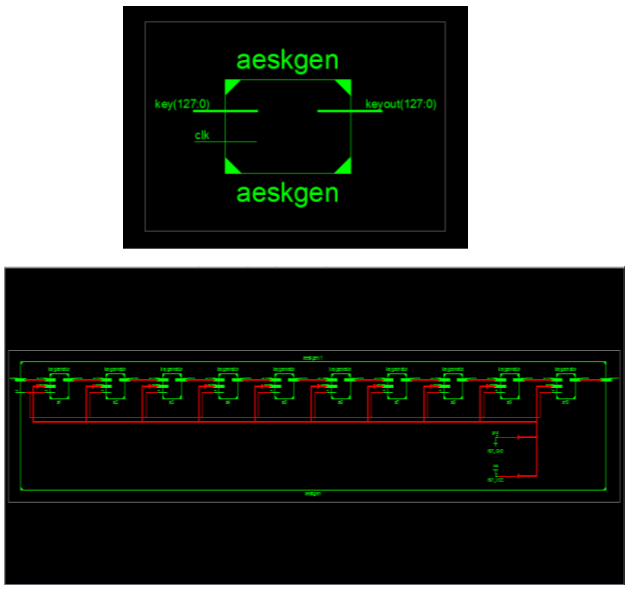


Fig. 7: Entity and internal architecture of Key Expansion module

B. Loop Unrolled Architecture

This architecture has only one round instead of ten different rounds and the single round can produce all ten keys. The feedback from output of a round to the key selector input is provided for selection. The implementation of entity and its architecture is shown in RTL schematic Fig. 8.

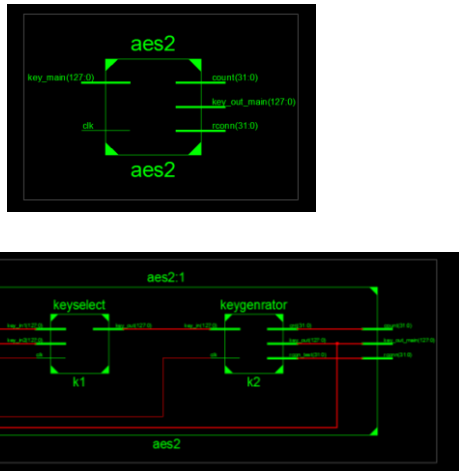


Fig. 8: Entity and internal architecture of Key Expansion module (Loop Unrolled)

V. PERFORMANCE AND COMPARISONS.

A. Output Waveforms and Throughput.

The simulation was carried out using Xilinx ISim simulator. The Fully pipelined architecture was able to produce all the ten keys in 20 clock cycles. So is the Loop Unrolled architecture. The Fig. 9 and Fig. 10, shows outputs produced by both implementations.

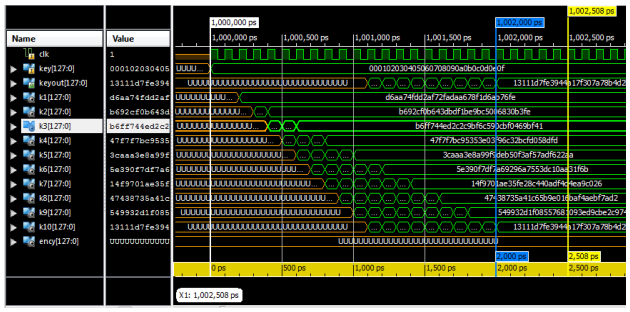


Fig. 9: Fully pipelined output

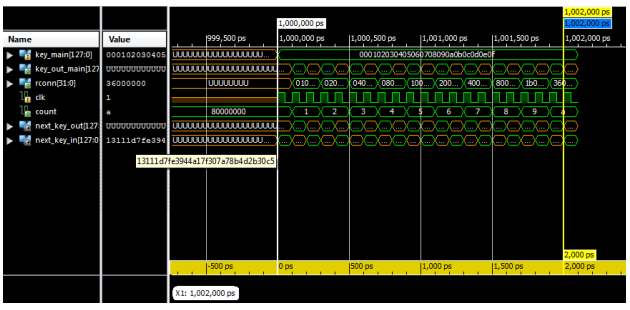


Fig. 10: Loop Unrolled output

Throughput is a measure of performance of an algorithm. Throughput of architecture depends on maximum frequency. Throughput can be calculated as below [2].

Throughput = 128 * Maximum Frequency **(2)**

ISSN No:-2456-2165

Maximum frequency can be obtained from Timing Summary after synthesis. This is an estimated by synthesis tool and can be found in synthesis report. The performances both implementations are shown in Table I.

TABLE I. PERFORMANCE COMPARISON OF BOTH ARCHITECTURES

Architecture	Max. Frequency (MHz)	Throughput (Gbps)	Clock cycles
Fully Pipelined	403.551	51.65	20
Loop Unrolled	162.829	20.84	20

Fully pipelined architecture achieves a very high throughput of 51.65 Gbps whereas loop unrolled achieves a low throughput of 20.84 Gbps, which is due to feedback mechanism in the architecture as shown in the Table I.

B. Device utilization.

The device utilization is found by Xilinx Synthesis tool. The results of both architectures are compared in Table II. From Table II it is observed that Loop Unrolled architecture consumes far less resources than the Fully Pipelined architecture. Hence using Loop Unrolled architecture we can choose a lower end FPGAs having fewer resources on board than Vertex 4 device which we have chosen for synthesis.

TABLE II. DEVICE UTILIZATION COMPARISON OF BOTH ARCHITECTURES

Resources used	Fully Pipelined	Loop Unrolled
Selected Device	4vfx40ff672-11	4vfx40ff672-11
Slices	877 out of 18624	293 out of 18624
Slice Flip Flops	1280 out of 37248	361 out of 37248
4 input LUTs	1600 out of 37248	547 out of 37248
IOs	257	321
Bonded IOBs	257 out of 352	321 out of 352
FIFO16/RAMB16s	20 out of 144	2 out of 144
RAMB16s	20	2
GCLKs	1 out of 32	1 out of 32

VI. CONCLUSIONS

In this paper we presented implementation of AES 128 key expansion algorithm in fully pipelined and loop unrolled architectures and their performances are compared in terms of device utilization and throughput. Fully pipelined architecture achieved a throughput of 51.65 Gbps and poor device utilization on FPGA. Whereas loop unrolled architecture achieved a throughput of 20.84 Gbps but has shown very efficient device utilization. This advantage could help in implementing the algorithm in low end FPGAs and eventually bring the cost down if throughput is not the criteria. Key expansion occupies significant portion of area in encryption or decryption. If we combine loop unrolled key expansion module with fully pipelined encryption or decryption, we can achieve tradeoff between device utilization and throughput for entire encryption or decryption.

REFERENCES

[1]. National Institute of Standards and Technology (NIST), Advanced Encryption Standard (AES), Federal Information Processing Standards Publications 197 (FIPS197), Nov. 2001, available at https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

[2]. H. Qin, T. Sasao, Y. Iguchi, “An FPGA Design of AES Encryption Circuit with 128-bit Keys”, Proceedings of the 15th ACM Great Lakes Symposium on VLSI (GLSVLSI), Illinois, April 17-19, 2005

[3]. K. Rahimunnisa, P. Karthigaikumar, N. Anitha Christy, S. Suresh Kumar, J. Jayakumar, “PSP: Parallel sub-pipelined architecture for high throughput AES on FPGA and ASIC”, Central European Journal of Computer Science, 2013

[4]. Abhijith P. S, Dr. Manish Goswami, S. Tadi, Kamal Pandey, “Optimized Architecture for AES”, available at https://eprint.iacr.org/2014/540.pdf