THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Department of
Electronics & Communication Engineering

Efficient Hardware Implementation of AES

Name: Sachleen S. Chani **Mentor: Dr. Vijaypal Singh Rathor**

Roll No. : 101506143

Abstract

- Hardware implementation of cryptographic algorithms is inherently more secure.
- An effective way to implement the AES algorithm in FPGA is explored in this project.
- Higher speeds and lesser area could be achieved by designing the SubBytes and MixColumns layers as a Look Up Tables (LUTs) and ROMs.

Introduction

- To securely transmit data, cryptography is used. There are two types of cryptographic systems: Symmetric and Asymmetric.
- Symmetric cypher algorithms use identical keys for the sender and the receiver.
- Rijndael algorithm , which is a symmetric, block cryptosystem, was chosen as the Advanced Encryption Standard (AES) by the NIST on 2nd Oct 2000.
- Hardware implementation provides greater speeds and higher reliability with large amount of data [1].

Advanced Encryption Standard

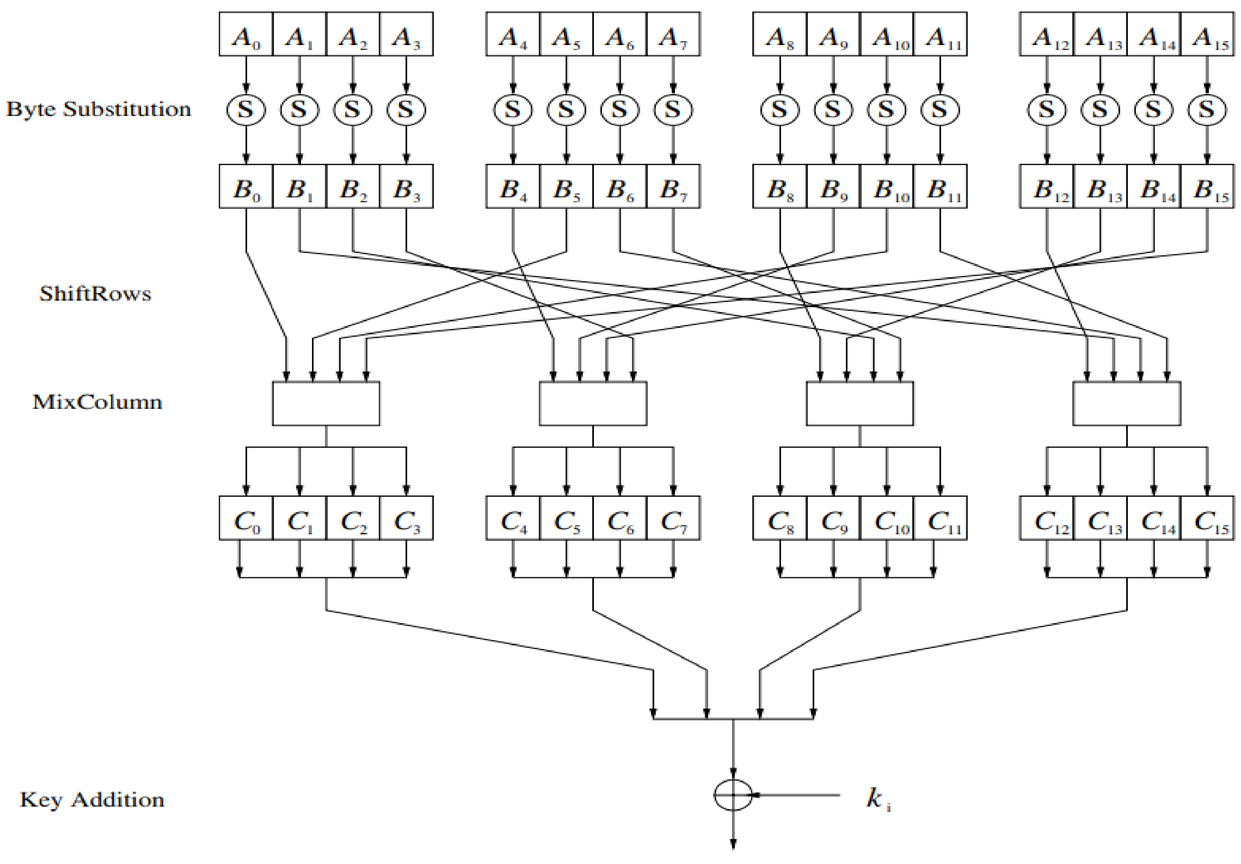
- AES algorithm is very robust because of the different key lengths, which results in AES-128, AES-192 and AES-256 format encryption for a 128-bit (16 word) block data.

Key size (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Plaintext block size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Number of rounds	10	12	14
Round key size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Expanded key size (words/bytes)	44/176	52/208	60/240

AES Parameters

- For this project, the implementation is using a key length of 128 bits.
- The Key is expanded into 44-word key.

AES round description



AES Round Functions

- SubBytes uses a LUT to substitute input 8-bits.
- ShiftRows rotates the rows of the 16-byte block through a fixed pattern.
- In the MixColumns layer multiplication of the block data is performed in Galois field GF(2^8).

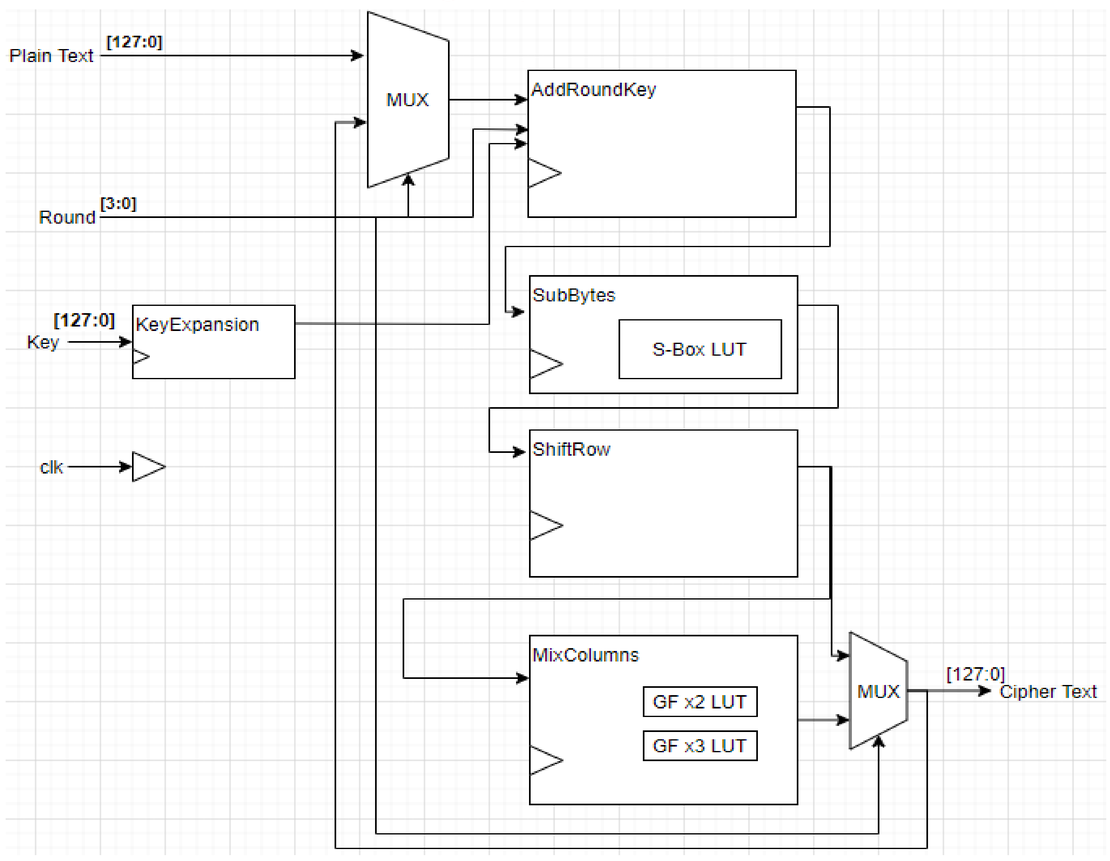
Look Up Tables and ROM Implementation

- Implementation utilizes LUTs (ROM) for the S-box SubBytes layer, multiplication in MixColumns layer [2].
- This reduces the computation cost otherwise needed for matrix multiplication in Galois field GF(2^8).

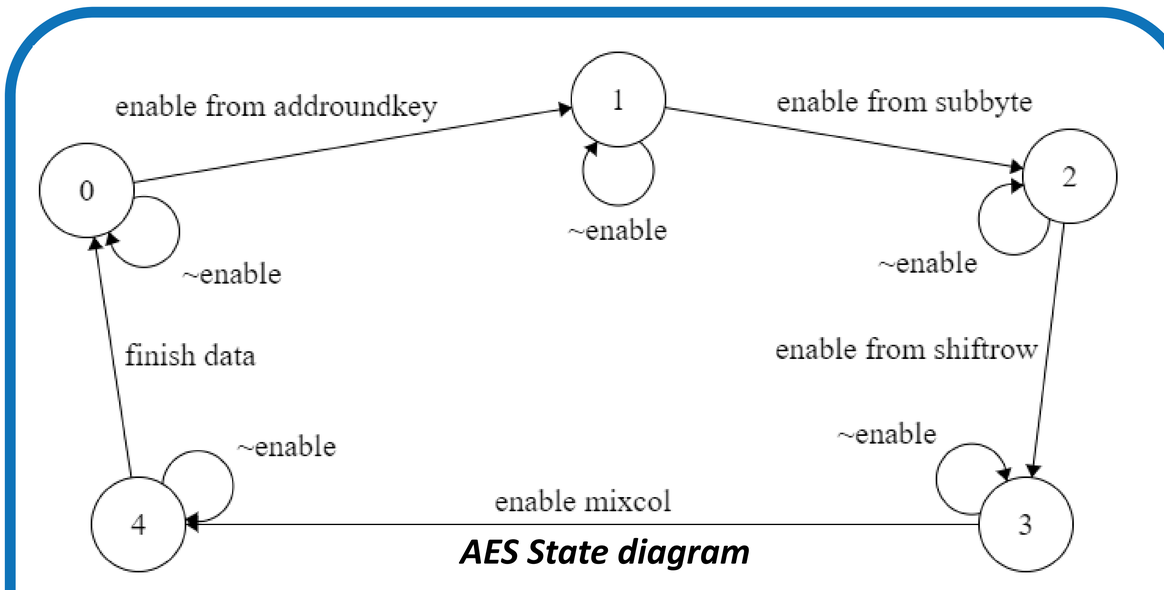
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	02	04	06	08	0A	0C	0E	10	12	14	16	18	1A	1C	1E
1	20	22	24	26	28	2A	2C	2E	30	32	34	36	38	3A	3C	3E
2	40	42	44	46	48	4A	4C	4E	50	52	54	56	58	5A	5C	5E
3	60	62	64	66	68	6A	6C	6E	70	72	74	76	78	7A	7C	7E
4	80	82	84	86	88	8A	8C	8E	90	92	94	96	98	9A	9C	9E
5	A0	A2	A4	A6	A8	AA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

Galois Multiplication by 2 LUT

Verilog Implementation



Top Level block diagram



AES State diagram

- This is a Moore FSM; the states depend on the internal enable.
- This implementation represents only first round for the AES. FSM stays in a state until the calculation of the state is not finished.

References

[1] P. S. Abhijith, M. Srivastava, A. Mishra, M. Goswami, and B. R. Singh, "High performance hardware implementation of AES using minimal resources," pp. 338-343, 2013, doi: 10.1109/ISSP.2013.6526931.

[2] W. McLoone and J. V. McCanny, "Rijndael FPGA implementation utilizing look-up tables," vol. 34, pp. 349-360, 2001, doi: 10.1109/SIPS.2001.957363.