International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2016): 79.57 | Impact Factor (2017): 7.296

Implement a Digital Speech Encoder, Decoder, Encryption and Decryption on an FPGA

Vishnupriya S Devarajulu[1], Bharath Ganineni[2], Sri Prudvi Raj Godthi[3], Siva Sudhakar Doddapaneni[4], Mohana Sudha Bandi[5]

Email: *vishnupriyasupriya[at]gmail.com*

Abstract: ***Increasing need of data protection in computer networks led to the development of several cryptographic algorithms hence sending data securely over a transmission link is critically important in many applications. To achieve higher performance in today's heavily loaded communication networks, hardware implementation is a wise choice in terms of better speed and reliability. This paper presents the Advanced Encryption Standard (AES) algorithm using Xilinx– virtex7 Field Programmable Gate Array (FPGA). To achieve higher speed and lesser area, Sub Byte operation, Inverse Sub Byte operation, Mix Column operation and Inverse Mix Column operations are designed as Look Up Tables and Read Only Memories. In this paper we presented a description of the components of a speech encoding and how to compress those speech signals. The ISE design suite 14.4 by Xilinx has been used for programming in VHDL for all the blocks of encoder, decoder, encryption and decryption.***

Keywords: AES; Rijndael; Cryptography; FPGA; Encryption; Decryption; Encoder; Decoder

1. Introduction

Cryptography allows people to carry over the confidence found in the physical world to the electronic world. The importance of cryptography is constantly increasing since the amount of sensitive data being transmitted over an open environment is also increasing day by day. The more information that is transmitted in computer - readable form, the more vulnerable we become to automated spying. Cryptography is not only important in defense applications but also important in real world applications such as E - commerce, E - mail etc.

Encryption and decryption use the AES algorithm. The AES was published by National Institute of Standards and Technology (NIST) in 2001. Later Rijndael algorithm was selected as AES algorithm. Rijndael algorithm can have key length of 128, 192 and 256 bits while block size must be 128 bit [2]. G.711 is used for performing the encoding and decoding part. G.711 is an ITU - T standard for audio compounding, It is primarily used in telephony. The standard was released in 1972. Its formal name is Pulse code modulation (PCM) of voice frequencies. It is a required standard in many technologies, for example in H.320 and H.323 specifications. It can also be used for fax communication over IP networks. G.711, also known as Pulse Code Modulation (PCM), is a very commonly used waveform codec. G.711 is a narrowband audio codec that provides toll - quality audio at 64 kbit/s. G.711 passes audio signals in the range of 300–3400 Hz and samples them at the rate of 8, 000 samples per second, with the tolerance on that rate 50 parts per million (ppm). The following block has been used in this present work.

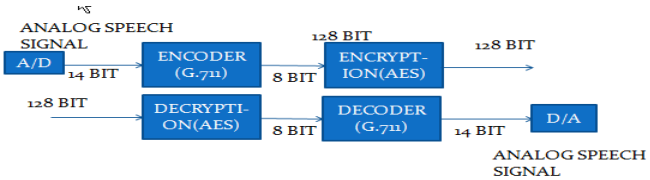
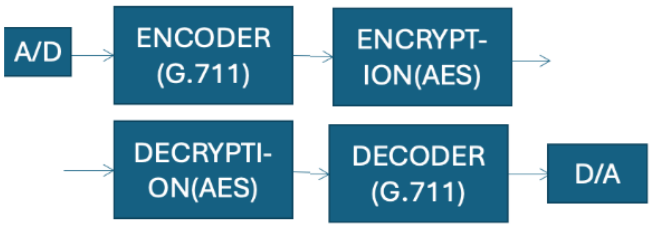


Figure 1: Block diagram for the overall view



2. Related Work

Significant work has been carried out in cryptography and error correction, although in the separate implementation of AES and convolutional codes. In [2], a joint encryption and error correction scheme based on the McEliece public - key cryptosystem using algebraic codes was proposed. They designed and synthesized speech encoding and encryption in a system - on - chip. Their design was intended for implementation with application - specific integrated circuits. In this paper the software implementations of encoder, encryption, decoder and decryption have been performed in VHDL

3. Encoder

3.1 Speech Encoding:

Speech coding is the application of data compression of digital audio signals holding speech. Speech coding utilizes speech particular parameters estimation utilizing audio indicator preparing procedures to model the speech sign, consolidated with generic data compression calculations to represent to the ensuing demonstrated parameters in a minimized bit stream.

The techniques utilized as a part of speech coding are like that in audio data compression and audio coding where information in psychoacoustics is utilized to transmit just data that is pertinent to the human auditory system. For instance, in narrowband speech coding, just data in the recurrence band 400 Hz to 3500 Hz is transmitted however

the remade sign is still sufficient for clarity. So, for the compression of the low frequency signals and to attain the above characters we can use G711 as the algorithm to encode the speech signal

G.711 is the default pulse code modulation (PCM) standard for Internet Protocol (IP) private branch exchange (PBX) vendors, as well as for the public switched telephone network (PSTN). G.711 digitizes analog voice signals producing output at 64 kilobits per second (Kbps). G711 is also known as Pulse Code Modulation (PCM). G711 mu - law encoding composed of three steps

- Sampling
- Quantization
- Coding

PCM is made of three progressive steps: sampling, quantizing also coding. Sampling is the determination of a signal's amplitude at normal time intervals. Since the phone system has a transmission capacity of 4 KHz, for faultless reproduction, a voice signal must be inspected at a rate of no less than 8 KHz, as stated by Nyquist's hypothesis. That is, the amplitude of the sign is inspected each 125 ms. When the signal's amplitude is acquired, it is quantized into a discrete set of amplitude levels for representation as an advanced signal. Quantization is attained by separating the data transmission of the framework into quantization intervals, also known as bins. All signal amplitudes falling inside a bin are represented to by the midpoint of the quantization interval. The quantization procedure presents quantization error into the digital signal; however, the presented error may be minimized by minimizing the width of the bins with respect to the number of bits required to remarkably distinguish the quantization bins. Finally, coding of the signal is performed by converting over the midpoint of every quantization level to a codeword. After the quantization, coding must be implemented on the PCM signal which the compression has to be done.

General PCM has a 14 - bit codeword once the quantization was done. To transmit this 14 - bit code the transmission cost would be more and more complicated. So, this 14 - bit codeword has to be compressed. So, for compression two international compounding standards that retain up to 5 bits of precision by encoding signal data into 8 bits are m - law and A - law. Mu - law is the accepted standard of the U. S. and Japan, while A - law is the European accepted standard. In this project we used mu law to compress

The equation for the mu law is

F (x) = sgn (x) ln (1 + □|x|) / ln (1 + □) - 1□□x □□1

Where m is the compression parameter (m=255 for the U. S. and Japan), and x is the normalized integer to be compressed.

4. Encryption

4.1 Encryption using Aes algorithm (Dijndeal Algorithm)

AES is a symmetric encryption and has a fixed block size of 128 bits. AES is symmetric since the same key is used for encryption and the reverse transformation, decryption. First the 128 - bit key is expanded into eleven so - called round keys, each of them 128 bits in size. After an initial round, during which the first - round key is XOR ed to the plain text, nine equally structured rounds follow. Each round consists of the following operations:

- Substitute bytes
- Shift rows
- Mix columns
- Add round key

4.2 Substitute bytes Operation:

The operation Substitute bytes is a nonlinear substitution operation. This operation adds security to AES. The operation can be considered as a lookup in the lookup table provided in the FIPS 197 guide and then substituting the corresponding value in the 16 bytes of the state. The operation can represent as below

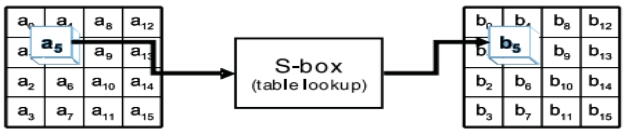


Figure 2: Substitute bytes operation

4.3 Shift Rows Operation

Here, the rows are shifted cyclically left leaving the first row unchanged. The second row is shifted one byte position to the left in the matrix, the third row is shifted two - byte positions to the left, and the fourth row is shifted three - byte positions to the left. The working of the shift rows operation can be explained pictorially as follows

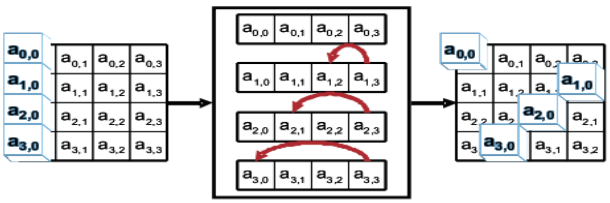


Figure 3: Shift rows operation

4.4 Mix Columns Operation

The mix columns operation is considered as the complex operation. Here, the normal addition operation is replaced by the exclusive OR operation. The mix columns operations is depicted as below

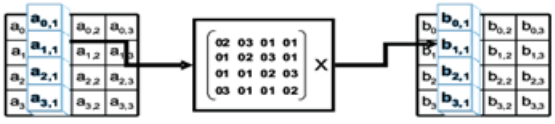


Figure 4: Mix Columns Operation

4.5 Add Round Key Operation:

The Add round key operation is nothing but a simple exclusive OR operation in between the input data (state) and the key (Round key). The expanded key bytes are never used. The operation can be represented as below

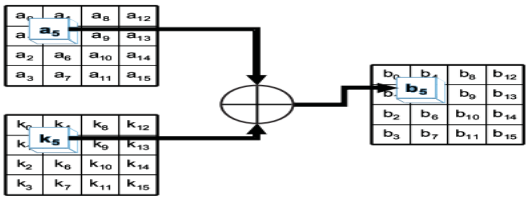


Figure 5: Add Round Key Operation

4.6 Key Expansion Operation:

The main operations or functions which are used in this key expansion routine are

- Sub word ()
- Rot word ()
- Rcon ()

The sub word () and Rot word () are just the substitution and rotation operations correspondingly on the 4 bytes. Rcon is the function which provides some constant values.

5. Decryption

This process is direct inverse of the Encryption process. All the transformations applied in Encryption process are inversely applied to this process.

5.1 Process for Decryption

5.1.1 The Inverse Cipher:

The inversion of the cipher code presented is straightforward and is just the reverse process of encryption. Decryption undergoes the following sequence

1) Add round key
2) Inverse sub bytes transformation
3) Inverse shift row
4) Inverse mix column
5) Inverse sub key

5.2 Add Round Key

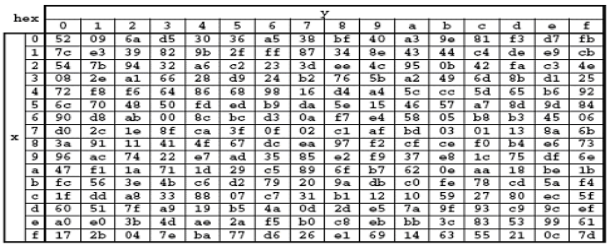
In decryption the key is generation is the inverse of the encryption process. As per Rijndael decryption goes through 10 rounds of decryption process.

5.3 Inverse Bytes Substitution Transformation

Inverse Byte Substitution Transformation is the inverse of the byte substitution transformation in Encryption, in which the inverse S - Box is applied to each byte of the State. This is obtained by applying the inverse of the affine transformation followed by taking the multiplicative inverse in GF (2^8).

5.4 Inverse Sub bytes transformation using S - Box

The inverse sub bytes transformation uses the inverse S - Box table provided in the figure



hex	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

x

Figure 6: Inverse subbytes transformation

5.5 Inverse Shift Rows Transformation

Inverse Shift Rows Transformation is the inverse of the Shift Rows transformation in Encryption. The bytes in the last three rows of the State are cyclically shifted over different numbers of bytes. The first row, $r = 0$, is not shifted. The bottom three rows are cyclically shifted by Nb - Shift (r, Nb) bytes.

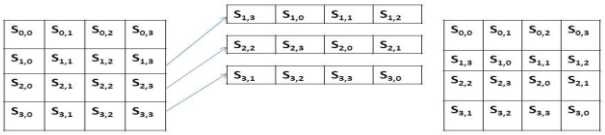


Figure 7: Inverse shift row

5.6 Decryption Block

The encryption process for 128 - bit data size goes through 10 rounds. The initial round only adds the starting key and the input data, and the result is the input of round 1. Round 1 through round 10 the starting data goes through sub - byte transformation, shift rows transformation, mix column transformation and then added with the specific round key generated for each round from previous round key. The flowchart of the top - level module is as shown below.

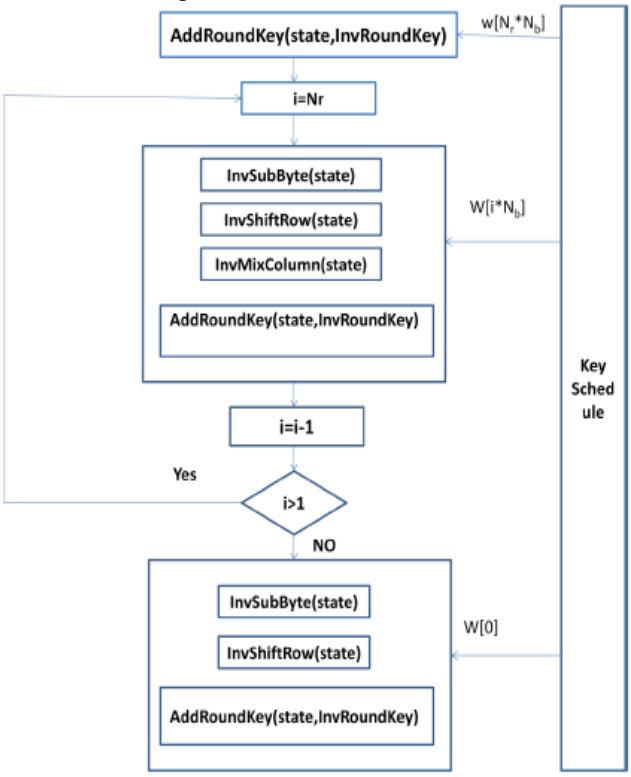


Figure 8: AES block controlling decryption

6. Decoder

6.1 Speech Decoding

Speech decoding is the application of data compression of digital audio signals holding speech. Decoding is the process by means of which the samples are reconstructed, from the numerical signal. This process is made in a device called decoder.

6.2 Decoding Process

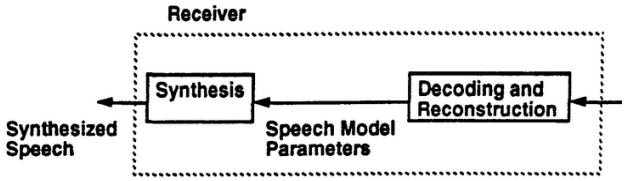


Figure 8: process of decoding

Decoding the m - law encoded data is essentially a matter of reversing the steps in the encoding. Table below illustrates the m - law decoding table, applied after reversing the inversion pattern. Before expansion, the m - law code is inverted again to restore the original code. During expansion, the discarded least significant bits are approximated by the median of the interval, to reduce the loss in accuracy. The m - law binary decoding table used for expansion is given in Table below.

Compressed Code Word							Biased Output Values											
Chord			Step															
bit: 6	5	4	3	2	1	0	bit: 12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	a	b	c	d	0	0	0	0	0	0	0	1	a	b	c	d	1
0	0	1	a	b	c	d	0	0	0	0	0	0	1	a	b	c	d	1	0
0	1	0	a	b	c	d	0	0	0	0	0	1	a	b	c	d	1	0	0
0	1	1	a	b	c	d	0	0	0	0	1	a	b	c	d	1	0	0	0
1	0	0	a	b	c	d	0	0	0	1	a	b	c	d	1	0	0	0	0
1	0	1	a	b	c	d	0	0	1	a	b	c	d	1	0	0	0	0	0
1	1	0	a	b	c	d	0	1	a	b	c	d	1	0	0	0	0	0	0
1	1	1	a	b	c	d	1	a	b	c	d	1	0	0	0	0	0	0	0

Figure 9: Values of the biased output

Step 1: Read the data from the file where the amplitudes are stored.
Step 2: extract the integer part alone from the data and save the sign magnitude of it.
Step 3: add 33 to the input data by using the simple full adder logic for biasing the sample.
Step 4: now assign the segment code of the data by checking the data from its MSB.
STEP 5: By checking the MSB alone we can decide the segment code. For segment 6 By checking the two bits from the MSB we decide the segment 6 and so on.
Step 6: After assigning the segment code we must assign the code value by checking the segment number as shown in the previous table.
Step 7: This compressed data will be converted again to decimal equivalent and then stored in another file with all the sign bits and decimal point.

7. Simulation Results

For testing, the digital input data and digital key of 128 - bit length has been considered. The test bench was created by these random input data and key with the help of which corresponding cipher text was generated. This cipher text would be the input for the decryption block. The results obtained are very satisfactory. One of the only difficulties was that the state had to match the operations done, that problem was solved using conditional statements. The change in frequency to a shorter clock period resulted in diminished performance that does not perform adequately

8. Hardware Implementation

The FPGA board used should be in compatibility with the requirements of all the blocks which include speech encoding, decoding encryption and decryption. Hardware implementation of cryptographic algorithms is physically secure than software implementations since outside attackers cannot modify them. Encryption standard has been commercially available and implemented for many years and offers some insights into the use of FPGA and custom hardware instructions to decipher coded DES messages.

9. Conclusion

The software implementations of encoder, encryption, decoder and decryption have been performed in VHDL. All the functions and transformations mentioned above in the report are also defined and used in the program. We used G7.11 standards for encoder and decoder. AES algorithm is used for encryption and decryption. The ISE design suite 14.4 by Xilinx has been used for programming in VHDL. By implementing the code for encoder and decoder we will be able to compress the 14 - bit PCM into 8 - bit codeword. The encryption and decryption blocks are designed and simulated, and the results are as expected.

10. Future Developments

At first, **Spartan 3E** starter board in the design suite software was taken for programming. The Input - Output pin utilization was found to be **110%** which was more than the availability. Thus, **Virtex7** board was considered for the software implementation and pin utilization problem was solved. So, Virtex7 board could be suggested for the future hardware implementation.

References

[1] M. Goswami and S. Kannojiya, “High Performance FPGA Implementation of AES Algorithm with 128 - Bit Keys, ” Proc. IEEE Int. Conf. Advances Computing Comm., vol.1, Himarpur, India, 2011, pp.281 - 286.
[2] FIPS - 197, NIST - National Institute of Standards and Technology, “Announcing the ADVANCED ENCRYPTION STANDARD (AES), ” ttp: //csrc. nist. gov/publications/fips/fips197/fips - 197. pdf, 2001.
[3] "A Simplified AES Algorithm. " N. p., 20 Jan.2010. Web. http: //www.rose - hulman. edu/~holden/Preprints/s - aes. pdf

[4] "AES128 Implementation for Encryption and Decryption. " *Texas Instruments*. N. p., Mar.2009. Web. <http: //www.ti. com/lit/an/slaa397a/slaa397a. pdf>.
[5] "Announcing the ADVANCED ENCRYPTION STANDARDS (AES). " *Federal Information Processing Standards Publication 197*. N. p., 26 Nov.2001. Web. <http: //csrc. nist. gov/publications/fips/fips197/fips - 197. pdf