AES Implementation

EE370 Project

Anurag Sai-Y9645   *Shantanu Chopra-Y9537*   *Neeraj Kulkarni-Y9292*

Introduction:

- The Advanced Encryption Standard (AES) specifies a FIPS (Federal Information Processing Standards)-approved cryptographic algorithm that is used to protect electronic data.
- The AES algorithm is asymmetric block cipher that can encrypt and decrypt information.
- Encryption converts data to an unintelligible form called cipher text; decrypting the cipher text converts the data back into its original form, called plaintext.

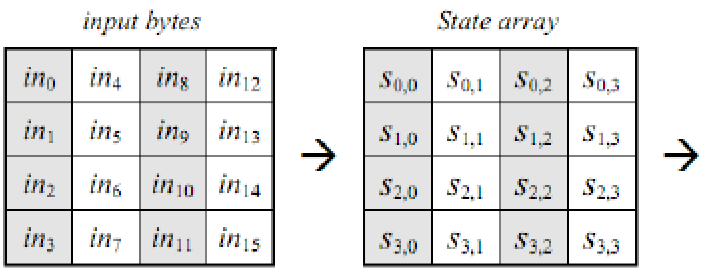
Problem Statement:

It involved implementation of the following modules of AES Algorithm:

- *Implementation of Cipher module (Encryption):*
  -Series of transformations that converts plaintext to cipher text using the cipher key.
- *Implementation of Key expansion:*
  -Used to generate a series of Round Keys from the Cipher Key.
- *Implementation of Inverse Cipher (Decryption):*
  -Finally obtain original text from cipher text.

AES Algorithm:

1. **Initialisation:** 128 bit data is received as input which is converted to a state matrix as follows:



implimentation by[1]

2. **Encryption:**

Pseudo Code for encryption:

```
Cipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
   byte state[4,Nb]
   state = in
   AddRoundKey(state, w[0, Nb-1])
   for round = 1 step 1 to Nr-1
         SubBytes(state)
         ShiftRows(state)
         MixColumns(state)
         AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
   end for
   SubBytes(state)
   ShiftRows(state)
   AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
   out = state
end
```

3. **Key Expansion:**

Psuedo code for key expansion is as follows:

```
KeyExpansion(byte key[4*Nk], word w[Nb*(Nr+1)], Nk)
begin
   word temp
   i = 0
   while (i < Nk)
         w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
         i = i+1
   end while
   i = Nk
   while (i < Nb * (Nr+1)]
         temp = w[i-1]
         if (i mod Nk = 0)
               temp = SubWord(RotWord(temp)) xor Rcon[i/Nk]
               else if (Nk > 6 and i mod Nk = 4)
               temp = SubWord(temp)
         end if
         w[i] = w[i-Nk] xor temp
         i = i + 1
   end while
 end
```

4. **Decryption :**

   Pseudo code for Decryption:

```
InvCipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
      byte state[4,Nb]
      state = in
      AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1]) // See Sec. 5.1.4
      for round = Nr-1 step -1 downto 1
            InvShiftRows(state) // See Sec. 5.3.1
            InvSubBytes(state) // See Sec. 5.3.2
            AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
            InvMixColumns(state) // See Sec. 5.3.3
      end for
      InvShiftRows(state)
      InvSubBytes(state)
      AddRoundKey(state, w[0, Nb-1])
      out = state
end
```

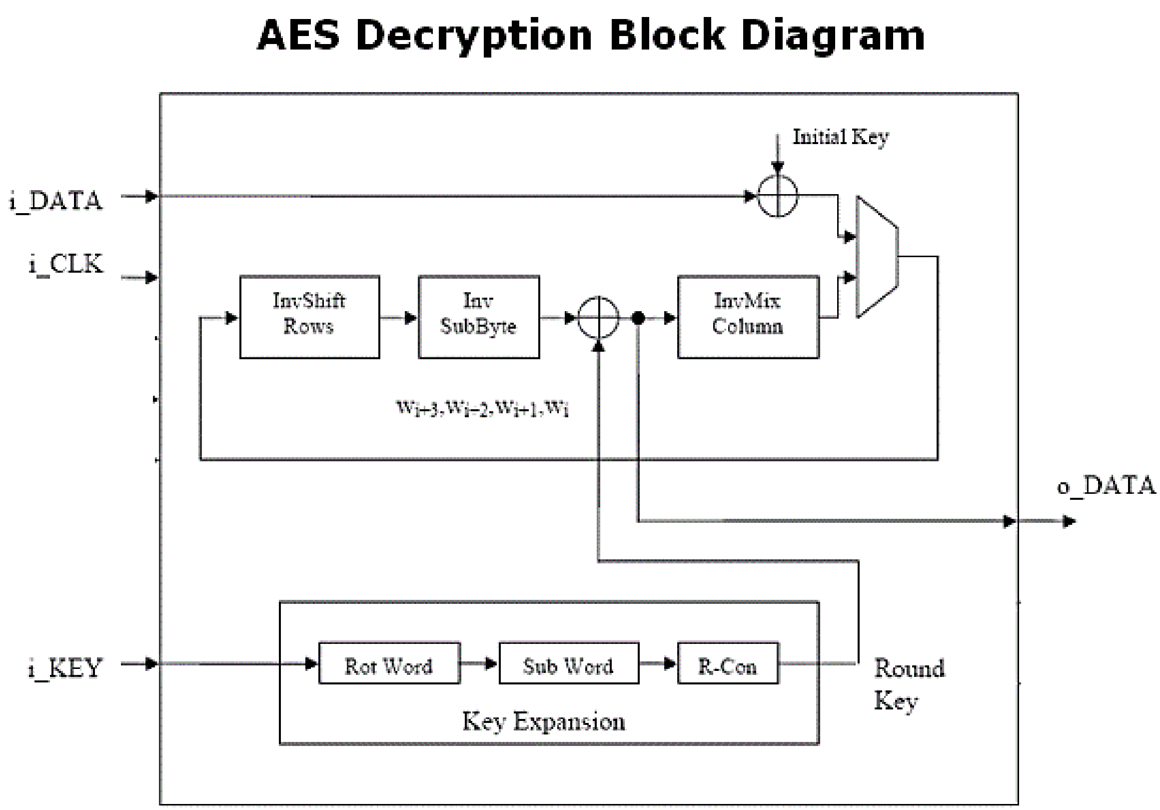
Implementation of Algorithm:

The following submodules were used in the implementation of the above pseudo codes:

1. <u>*Subbytes & INV Subbytes:*</u> *These modules substitute each byte in the state matrix using non linear transformation. We* essentially implemented the transformation using Sbox look up table.
2. <u>*Shiftrows & INV Shiftrows:*</u> *In these modules rows of the state matrix are shifted cyclically.*
3. <u>*Mixcolumns & INV Mixcolumns:*</u> *Performs a matrix multiplication on the state matrix.*
4. <u>*Round Key generation:*</u> For the generation of round keys AES algorithm takes the Cipher Key, K, and performs a Key Expansion routine to generate a key schedule. The Key Expansion generates a total of Nb (Nr + 1) words: the algorithm requires an initial set of Nb words, and each of the Nr rounds requires Nb words of key data. The resulting key schedule consists of a linear array of 4-byte words, denoted w[i], with i in the range 0 < i < Nb(Nr + 1).

5. *<u>AddRoundKey:</u> In this transformation, a Round Key is added to the State by a simple bitwise XOR operation.*
6. <u>Modules of KeyExpansion:</u>
   a. SubWord() -A function that takes a four-byte input word and applies the S-box to each of the four bytes to produce an output word.
   b. RotWord()-The function takes word [a0,a1,a2,a3] as input, performs a cyclic permutation, and returns the word[a1,a2,a3,a0].
   c. Rcon-The round constant word array, Rcon[i], contains the values given by [xi-1,{00},{00},{00}], with x^i-1 being powers of x (x is denoted as {02}) in the field GF(2^8).

Block Diagram of Circuit:



Block Diagram for Encryption is same as above the difference being in the use ShiftRows, SubBytes, Mixcolumns modules instead of Inverses of them.

References:

1. National Institute of Standards and Technology, Advanced Encrytion Standard, Federal Information Processing Standards 197, November 2001
2. A compact pipeline hardware implementation of the AES 128 Cipher, *Nadia Nedjah, Luiza de Macedo Mourelle, Marco Paulo Cardoso,* IEEE Computer society.