# IMPLEMENTING AES-256 ON FPGA

19th May 2021

Ho Hai Cong Thuan
Student, Computer Engineering, University of Information Technology, HCMC, Vietnam
hohaicongthuan@gmail.com

***Abstract*—In the modern days, the amount of data grow exponentially, including classified and sensitive data that need to be kept secured. For this reason, many cryptographic techniques have been invented for the purpose. AES is one of them. It provides fast and secure data encryption which are the reasons this algorithm is chosen for this project.**

**The goal of this project is to implement a fully functional AES encryption and decryption system using 256-bit key on FPGA.**

***Keywords*—AES-256; cryptography; data security; FPGA; encryption; decryption.**

# 1 INTRODUCTION

*The Advanced Encryption Standard (AES)*, also known as *Rijndael* is a specification for encrypting electronic data first introduced by the *U.S. National Institute of Standards and Technology (NIST)* in 2001. It provides a fast and secure way to encrypt data and uses symmetric keys encryption which means both the encryption and decryption processes using the same key. The key length for AES could be 128, 192 and 256 bits. This paper concentrates on AES using 256-bit key which will be refered to as AES-256 for the rest of this paper.

## 1.1 Concepts used in AES-256

| | |
|---|---|
| ***Key expansion*** | Routine used to generate a series of Round Keys from the Cipher Key. |
| ***State*** | Intermediate Cipher result that can be pictured as a rectangular array of bytes, having four rows and ***Nb*** columns. |
| ***S-box*** | Non-linear substitution table used in several byte substitution transformation and in the Key Expansion routine to perform a one-for-one substitution of a byte value. |
| ***Word*** | A group of 32 bits that is treated either as a single entity or as an array of 4 bytes. |

## 1.2 Abbreviations and Symbols used in AES-256

| | |
|---|---|
| ***Nb*** | Number of columns (32-bit words) comprising the State. For this standard, $\boldsymbol{Nb} = 4$. |
| ***Nk*** | Number of 32-bit words comprising the Cipher Key. For this standard, $\boldsymbol{Nk} = 8$. |
| ***Nr*** | Number of rounds, which is a function of ***Nk*** and ***Nb*** (which is fixed). For this standard, $\boldsymbol{Nr} = 14$. |
| XOR | Exclusive-OR operation |
| $\oplus$ | Exclusive-OR operation |
| $\otimes$ | Multiplication of two polynomials (each with degree $< 4$) modulo $x^4 + 1$ |
| $\bullet$ | Finite field multiplication |

# 2 AES-256

## 2.1 Key Expansion

The Key Expansion routine in AES-256 takes a 256-bit cipher key and generate a set of $Nb(Nr+1)$ (which is 60) words. These words are smaller parts that make up round keys, each round key has four words. These round keys involve in the *Add Round Key* in the encryption and decryption process.

There are 3 functions that participate in the key scheduling process:

| | |
|---|---|
| ***RotWord*** | Takes a four-byte word $[a_0, a_1, a_2, a_3]$ and performs rotation one byte to the left and returns $[a_1, a_2, a_3, a_0]$. |
| ***SubWord*** | Takes a four-byte word and substitute each byte with the corresponding byte in the S-box. |
| ***Rcon*** | The round constants, which is given in the form $[rc_i, 00_{16}, 00_{16}, 00_{16}]$ with $i$ starts from 1. $rc_i$ is defined as in (1) |

$$rc_i = \begin{cases} 1 & \text{if } i = 1 \\ 2 \cdot rc_{i-1} & \text{if } i > 1 \text{ and } rc_{i-1} < 80_{16} \\ (2 \cdot rc_{i-1}) \oplus 11\text{B}_{16} & \text{if } i > 1 \text{ and } rc_{i-1} \geq 80_{16} \end{cases} \quad (1)$$

### 2.2 Algorithm

In AES-256, the first two round keys (first 8 words) are filled with the cipher key. For the rest, $w[i]$ word is generated using $w[i-1]$ word.

Loop through the following steps until we have generated $Nb(Nr+1)$ words.

If $i$ is divisible by ***Nk***, $w[i-1]$ is rotated by the function ***RotWord*** and then substituted by the function ***SubWord***. The final result is ***XOR***-ed with ***Rcon[i/Nk]*** and assigned to $w[i]$. Otherwise, if $i$ dividing by ***Nk*** results in 4 as the remainder, only ***SubWord*** is performed on $w[i-1]$.

$w[i]$ will then be ***XOR***-ed with $w[i-Nk]$ and the result is assigned back to itself. $i$ is incremented by 1.

After finishing the algorithm, a set of $Nb(Nr+1)$ words is generated. Round Keys are created by grouping four words each sequentially. At this point, we have one round key for the initial round and 14 round keys for 14 rounds during the encryption or decryption processes, with the total of 15 round keys.

# 3 IMPLEMENTATION

# 4 RESULTS

# 5 CONCLUSION

# References


[1] Federal Information Processing Standard (FIPS) 197. *Advanced Encryption Standard (AES)* 26 November 2001.

[2] Sam Trenholme. *Rijndael's key schedule. https://samiam.org/key-schedule.html.*