TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KỸ THUẬT MÁY TÍNH

BÁO CÁO ĐỒ ÁN
THIẾT KẾ HỆ THỐNG SỐ HDL

ĐỀ TÀI

ADVANCED ENCRYPTION STANDARD

GV hướng dẫn: Ngô Hiếu Trường

Lớp: CE213.Q12

Nhóm sinh viên thực hiện:

Họ và Tên	*MSSV*
Vũ Thành Lam	**23520840**
Lê Trần Huỳnh Phong	**23521164**
Nguyễn Đức Toàn	**23521606**

Hồ Chí Minh, 2025

ACKNOWLEDGMENT

Trước hết,nhóm em xin gửi lời cảm ơn chân thành và sâu sắc nhất đến Thầy Ngô Hiếu Trường, giảng viên hướng dẫn đồ án, đã tận tình chỉ bảo, cung cấp kiến thức chuyên môn vững vàng cùng những góp ý kịp thời và định hướng quý báu xuyên suốt quá trình nghiên cứu và triển khai đề tài.

Nhóm em cũng xin gửi lời cảm ơn đến Khoa Kỹ thuật Máy tính và Trường Đại học Công nghệ Thông tin đã trang bị cơ sở vật chất và nguồn tài liệu học thuật cần thiết, tạo môi trường học tập và nghiên cứu tốt nhất.

Cuối cùng,em xin gửi lời cảm ơn đến các thành viên trong nhóm đã cùng nhau nỗ lực, chia sẻ kiến thức và hợp tác hiệu quả để vượt qua mọi thử thách, hoàn thành đồ án này.

Xin chân thành cảm ơn.

MỤC LỤC

MỤC LỤC HÌNH ẢNH

MỤC LỤC BẢNG

Chương 1

GIỚI THIỆU

1.1 Tổng quan

AES (Advanced Encryption Standard) là tiêu chuẩn mã hóa dữ liệu do Viện Tiêu chuẩn và Công nghệ Hoa Kỳ (NIST) ban hành năm 2001 nhằm thay thế thuật toán DES vốn đã lỗi thời. AES thuộc nhóm thuật toán mã hóa khối đối xứng, trong đó quá trình mã hóa và giải mã đều sử dụng cùng một khóa bí mật. Với độ an toàn cao, hiệu suất tốt và khả năng triển khai linh hoạt, AES đã nhanh chóng trở thành chuẩn mã hóa được sử dụng rộng rãi trong các hệ thống bảo mật hiện đại.

1.1.1 AES là gì

AES là thuật toán mã hóa khối (block cipher), hoạt động trên các khối dữ liệu có kích thước cố định 128 bit. Thuật toán hỗ trợ ba độ dài khóa gồm 128 bit, 192 bit và 256 bit, tương ứng với số vòng lặp xử lý khác nhau trong quá trình mã hóa/giải mã. AES được phát triển từ thuật toán Rijndael, là thuật toán chiến thắng trong cuộc thi lựa chọn chuẩn mã hóa mới do NIST tổ chức.

1.1.2 Lịch sử phát triển

Bối cảnh ra đời

Trong giai đoạn 1970–1990, thuật toán DES (Data Encryption Standard) được sử dụng rộng rãi như một chuẩn mã hóa dữ liệu. Tuy nhiên, với độ dài khóa chỉ 56 bit, DES dần trở nên không còn đảm bảo an toàn trước sự phát triển mạnh mẽ của năng lực tính toán. Đến cuối thập niên 1990, các cuộc tấn công brute-force đã có thể phá vỡ DES trong thời gian

ngắn, cho thấy nhu cầu cấp thiết về một thuật toán mã hóa mới mạnh hơn và linh hoạt hơn.

Quá trình tuyển chọn AES (1997–2000)

Năm 1997, NIST chính thức công bố kế hoạch xây dựng một chuẩn mã hóa mới để thay thế DES. Cuộc thi lựa chọn AES được tổ chức công khai với các tiêu chí:

- Thuật toán mã hóa khối đối xứng.
- Kích thước khối cố định 128 bit.
- Hỗ trợ các độ dài khóa 128, 192 và 256 bit.
- Bảo mật cao, hiệu suất tốt và khả năng triển khai rộng rãi.

Trong giai đoạn 1997–2000, cuộc thi diễn ra qua hai vòng với sự tham gia của nhiều nhóm nghiên cứu trên thế giới.

- Vòng 1 (1998): NIST nhận được 15 thuật toán đề cử.
- Vòng 2 (1999): 5 thuật toán được chọn vào vòng chung kết gồm: MARS, RC6, Rijndael, Serpent và Twofish.

Đến tháng 10/2000, NIST công bố Rijndael là thuật toán chiến thắng.

Lý do Rijndael được lựa chọn

Rijndael được đánh giá vượt trội nhờ các đặc điểm:

- Bảo mật tốt, không phát hiện điểm yếu nghiêm trọng trong quá trình đánh giá.
- Hiệu suất cao trên nhiều nền tảng phần mềm và phần cứng.
- Thiết kế đơn giản, dễ triển khai, dễ tối ưu hóa.
- Tính linh hoạt cao trong cấu trúc thuật toán.

Chính thức hóa và ứng dụng (2001–nay)

Tháng 11/2001, AES chính thức được ban hành trong tiêu chuẩn FIPS PUB 197. Kể từ năm 2002, AES trở thành chuẩn mã hóa liên bang Hoa Kỳ và được ứng dụng rộng rãi trong các lĩnh vực như ngân hàng, viễn thông, an ninh mạng và các giao thức bảo mật như VPN, TLS/SSL, WPA2/WPA3.
Năm 2010, Intel giới thiệu tập lệnh AES-NI, giúp tăng tốc đáng kể tốc độ mã hóa/giải mã trong phần cứng. Đến nay, AES được xem là thuật toán mã hóa đối xứng thông dụng và an toàn nhất, được khuyến nghị sử dụng cho nhiều ứng dụng bảo mật quan trọng.

1.1.3 Các đặc điểm của AES

Đặc điểm cơ bản

AES là thuật toán mã hóa khối đối xứng, trong đó dữ liệu được xử lý theo từng khối có kích thước cố định 128 bit. Thuật toán sử dụng cùng một khóa cho cả quá trình mã hóa và giải mã. AES hỗ trợ ba độ dài khóa là 128 bit, 192 bit và 256 bit. Tương ứng với đó, số vòng xử lý của thuật toán lần lượt là 10, 12 và 14 vòng.

Cấu trúc thuật toán

AES được thiết kế theo mô hình Substitution–Permutation Network (SPN), bao gồm nhiều vòng lặp xử lý. Mỗi vòng (trừ vòng cuối) gồm bốn bước chính:

- SubBytes: Thay thế từng byte trong khối dữ liệu bằng giá trị tương ứng trong bảng S-box nhằm tạo tính phi tuyến và tăng độ an toàn của thuật toán.
- ShiftRows: Dịch chuyển các hàng trong ma trận trạng thái để tạo sự khuếch tán dữ liệu theo hàng.
- MixColumns: Trộn dữ liệu theo từng cột bằng các phép toán trong trường Galois nhằm tăng khả năng khuếch tán (diffusion). Bước này được bỏ qua ở vòng cuối.
- AddRoundKey: Thực hiện phép XOR giữa dữ liệu và khóa vòng. Khóa vòng được sinh từ khóa chính thông qua quá trình Key Schedule.

Đặc điểm bảo mật

AES được xem là một trong các thuật toán mã hóa an toàn nhất hiện nay nhờ:

- Không tồn tại điểm yếu nghiêm trọng được công bố cho đến hiện tại.
- Khả năng chống tấn công brute-force rất cao do không gian khóa lớn (2^{128} với AES-128).
- Thiết kế kháng tốt các dạng tấn công phân tích như differential và linear cryptanalysis.
- Cấu trúc S-box và MixColumns được xây dựng dựa trên nền tảng toán học vững chắc, giúp hạn chế tấn công đại số.

Đặc điểm hiệu suất

AES có hiệu suất xử lý cao, phù hợp cho cả phần mềm lẫn phần cứng:

- Thuật toán hoạt động nhanh, tiết kiệm tài nguyên và phù hợp cho các ứng dụng thời gian thực.
- Tối ưu tốt trên CPU hiện đại, đặc biệt khi sử dụng tập lệnh tăng tốc phần cứng AES-NI, giúp cải thiện tốc độ từ 4 đến 10 lần.
- Yêu cầu bộ nhớ thấp, thích hợp cho thiết bị nhúng và hệ thống IoT.

Tính linh hoạt và khả năng triển khai

AES dễ triển khai trên nhiều nền tảng khác nhau nhờ cấu trúc thuật toán rõ ràng và đơn giản. Thuật toán tương thích với nhiều chế độ hoạt động (modes of operation) như:

- ECB: Đơn giản nhưng kém an toàn vì không che giấu cấu trúc dữ liệu.
- CBC: Bảo mật tốt hơn, yêu cầu vector khởi tạo (IV).
- CTR: Tốc độ cao, có thể xử lý song song.
- GCM: Hỗ trợ mã hóa đồng thời với xác thực (AEAD), rất phổ biến hiện nay.

Chuẩn hóa và ứng dụng

AES được chuẩn hóa trong FIPS PUB 197 và ISO/IEC 18033-3, trở thành thuật toán mã hóa đối xứng tiêu chuẩn trong nhiều hệ thống bảo mật. Thuật toán được hỗ trợ rộng rãi trong các thư viện mật mã như OpenSSL, CryptoAPI, BouncyCastle và tích hợp sẵn trên phần lớn các CPU hiện đại.

Một số hạn chế

Mặc dù có độ an toàn cao, AES vẫn cần được triển khai đúng cách:

- Quản lý và bảo vệ khóa là yếu tố quan trọng để đảm bảo an toàn hệ thống.
- Một số chế độ như CBC và CTR yêu cầu IV/nonce thích hợp để tránh rò rỉ thông tin.
- AES có thể bị tấn công kênh kề (side-channel) nếu phần cứng hoặc phần mềm triển khai không cẩn thận.
- Trong bối cảnh máy tính lượng tử, độ an toàn của AES giảm nhẹ (theo thuật toán Grover), tuy nhiên AES-256 vẫn được xem là đủ mạnh cho tương lai gần.

1.2 Ứng dụng thực tế của AES

AES là một trong những chuẩn mã hóa đối xứng được sử dụng rộng rãi nhất hiện nay nhờ tính bảo mật cao, hiệu năng tốt và khả năng triển khai linh hoạt trên nhiều nền tảng khác nhau. Dưới đây là các ứng dụng tiêu biểu của AES trong thực tế.

1.2.1 Bảo vệ dữ liệu cá nhân và tài khoản

AES được sử dụng trong hầu hết các hệ thống bảo vệ thông tin cá nhân:

- Mã hóa mật khẩu trong các cơ sở dữ liệu.
- Bảo vệ dữ liệu người dùng trong các dịch vụ email, mạng xã hội và ứng dụng web.
- Mã hóa tệp tin cá nhân trong các hệ điều hành như Windows (BitLocker), macOS (FileVault), Android và iOS.
- Bảo vệ dữ liệu nhạy cảm như ảnh cá nhân, danh bạ, tài liệu quan trọng.

Nhờ vào độ an toàn cao, AES giúp giảm nguy cơ rò rỉ dữ liệu và các cuộc tấn công đánh cắp thông tin cá nhân.

1.2.2 Bảo về dữ liệu trong lĩnh vực y tế

Trong lĩnh vực y tế, dữ liệu bệnh nhân rất nhạy cảm và phải tuân theo các tiêu chuẩn an toàn nghiêm ngặt. AES được sử dụng để:

- Mã hóa hồ sơ bệnh án điện tử (EHR/EMR).
- Bảo vệ dữ liệu xét nghiệm, lịch sử điều trị, chẩn đoán.
- Bảo mật các thiết bị IoT y tế như máy đo nhịp tim, thiết bị truyền dữ liệu theo dõi bệnh nhân.
- Ngăn chặn truy cập trái phép vào hệ thống bệnh viện.

Nhờ đó, AES giúp đảm bảo tính riêng tư và tuân thủ các tiêu chuẩn như HIPAA.

1.2.3 Bảo vệ dữ liệu trong lĩnh vực tài chính

Ngành tài chính – ngân hàng là nơi AES được ứng dụng nhiều nhất:

- Mã hóa giao dịch ngân hàng trực tuyến.
- Bảo vệ số tài khoản, số thẻ tín dụng và dữ liệu thanh toán.
- Bảo mật ví điện tử, mobile banking và các thiết bị POS.
- Sử dụng trong chuẩn PCI-DSS và các giao thức thanh toán điện tử.

AES giúp đảm bảo an toàn cho các giao dịch tài chính thời gian thực với nguy cơ tấn công rất cao.

1.2.4 Bảo vệ dữ liệu trong thiết bị di động và ứng dụng di động

Hầu hết hệ điều hành di động hiện nay đều tích hợp AES:

- iOS sử dụng AES-256 để mã hóa toàn bộ bộ nhớ.
- Android sử dụng AES trong File-Based Encryption (FBE).

- Ứng dụng OTT như Zalo, Messenger, Viber dùng AES trong quá trình truyền tải dữ liệu.

AES giúp ngăn chặn việc giải mã trái phép khi thiết bị bị mất hoặc bị đánh cắp.

1.2.5 Bảo mật trong các hệ thống đám mây (Cloud Computing)

Dịch vụ đám mây là nơi AES được sử dụng cực kỳ rộng rãi:

- Mã hóa dữ liệu lưu trữ trên AWS, Azure, Google Cloud.
- Mã hóa dữ liệu khi truyền tải giữa các server.
- Bảo vệ dịch vụ lưu trữ như Google Drive, OneDrive.
- Các hệ thống backup/restore cũng sử dụng AES để đảm bảo an toàn dài hạn.

AES giúp tăng cường tính toàn vẹn và bí mật cho dữ liệu trong môi trường phân tán.

1.2.6 Bảo mật trong các Internet of Things (IoT)

Với số lượng thiết bị IoT ngày càng tăng, AES đóng vai trò quan trọng:

- Mã hóa dữ liệu cảm biến (sensor data).
- Bảo vệ giao tiếp của smart home, smart city.
- Bảo mật camera IP, router và thiết bị gia dụng thông minh.
- Mã hóa dữ liệu truyền qua giao thức nhẹ như MQTT.

Nhờ yêu cầu tài nguyên thấp, AES phù hợp với các thiết bị IoT nhỏ.

1.2.7 Bảo mật trong các ứng dụng truyền thông và trò chơi điện tử

() AES còn được dùng trong các hệ thống truyền thông và giải trí:

- Bảo vệ dữ liệu trong game online, tránh gian lận hoặc sửa gói tin.
- Mã hóa nội dung video streaming (Netflix, YouTube).
- DRM (Digital Rights Management) bảo vệ bản quyền phần mềm và đa phương tiện.

- Mã hóa gói tin trong các phần mềm chat, VoIP, video call.

AES giúp đảm bảo chất lượng truyền thông và chống tấn công vào hệ thống.

1.3 Nhiệm vụ đề tài

Đề tài "Advanced Encryption Standard – AES256 và triển khai kiến trúc Pipeline" hướng đến việc nghiên cứu, thiết kế và đánh giá hoạt động của thuật toán mã hóa AES256 trên nền tảng phần cứng số. Các nhiệm vụ chính của đề tài bao gồm:

1.3.1 Nghiên cứu lý thuyết

- Tìm hiểu tổng quan về lịch sử phát triển, cấu trúc và cơ chế hoạt động của AES, đặc biệt là phiên bản AES256.
- Phân tích các thành phần chính của thuật toán gồm: SubBytes, ShiftRows, MixColumns, AddRoundKey và Key Expansion.
- Khảo sát các phương pháp tối ưu hóa thuật toán AES trong phần cứng, bao gồm thiết kế Pipeline, song song hóa và chia nhỏ đường dữ liệu.

1.3.2 Thiết kế và mô phỏng phần cứng AES256

- Xây dựng mô hình phần cứng AES256 sử dụng ngôn ngữ mô tả phần cứng Verilog.
- Thiết kế Pipeline cho từng giai đoạn của thuật toán nhằm tăng throughput.
- Thiết kế testbench và mô phỏng hoạt động thuật toán bằng ModelSim để kiểm tra tính đúng đắn của từng khối xử lý.

1.3.3 Tổng hợp và triển khai trên FPGA

- Tổng hợp thiết kế trên nền tảng FPGA thông qua công cụ Vivado hoặc Quartus Prime.
- Phân tích tài nguyên sử dụng (LUTs, Registers, DSPs, BRAMs) thông qua báo cáo tổng hợp.
- Đánh giá tốc độ hoạt động, tần số tối đa (Fmax) và mức tiêu thụ năng lượng.

1.3.4 Viết báo cáo và hướng phát triển

- Tổng hợp toàn bộ quá trình nghiên cứu và kết quả thành báo cáo hoàn chỉnh.
- Đề xuất cải tiến cho các hướng phát triển tiếp theo như tối ưu kiến trúc Pipeline, tích hợp AES vào hệ thống SoC, hoặc triển khai phiên bản AES-GCM.

1.4 Giới hạn của đề tài

Mặc dù đề tài hướng đến việc triển khai và tối ưu hóa thuật toán AES256 trên nền tảng phần cứng số, nhưng phạm vi nghiên cứu vẫn tồn tại một số giới hạn nhất định nhằm phù hợp với thời gian và yêu cầu của môn học. Các giới hạn cụ thể bao gồm:

1.4.1 Giới hạn về chức năng

- Đề tài chỉ tập trung vào quá trình mã hóa (encryption) của AES256; quá trình giải mã (decryption) không được triển khai.
- Các chế độ hoạt động của AES như ECB, CBC, CFB, OFB hoặc GCM không nằm trong phạm vi thực hiện; mô hình chỉ triển khai AES256 ở dạng khối chuẩn.
- Chỉ thiết kế kiến trúc Pipeline cho thuật toán AES, không xây dựng hệ thống hoàn chỉnh (ví dụ: tích hợp DMA, giao tiếp ngoại vi hoặc bus hệ thống).

1.4.2 Giới hạn về phần cứng

- Thiết kế được tổng hợp và kiểm tra trên FPGA, nhưng không triển khai thực nghiệm trên bo mạch thật do hạn chế thiết bị.
- Việc đánh giá tiêu thụ năng lượng thực tế của FPGA chỉ dựa trên báo cáo từ phần mềm tổng hợp (Power Analyzer), không đo đạc bằng thiết bị chuyên dụng.
- Thiết kế chưa tối ưu hóa chuyên sâu cho từng dòng FPGA hoặc ASIC.

1.4.3 Giới hạn về mô phỏng và kiểm thử

- Mô phỏng chỉ dừng lại ở testbench cơ bản, kiểm tra các vector đầu vào – đầu ra.

- Không thực hiện kiểm thử nâng cao như kiểm thử ngẫu nhiên (random test), kiểm thử lỗi hoặc phân tích side-channel (timing/power attack).
- Không so sánh hoạt động của mô hình với các chuẩn thương mại như OpenSSL, AES-NI hoặc module cryptography của ARM.

1.4.4 Giới hạn về tối ưu hóa

- Kỹ thuật Pipeline được áp dụng ở mức độ cơ bản; chưa triển khai các cải tiến nâng cao như:
  - Pipeline sâu (deep-pipeline)
  - Kỹ thuật retiming nâng cao
  - Tối ưu hóa song song hóa dữ liệu (parallel processing)
- Hiệu năng đạt được chủ yếu phụ thuộc vào cấu trúc Pipeline chuẩn, chưa tối ưu hóa cho mục tiêu cực đại hóa throughput.

1.4.5 Giới hạn về thời gian thực hiện

- Do thời gian thực hiện đồ án có hạn, đồ án tập trung vào đúng trọng tâm chính là tìm hiểu, mô phỏng và tổng hợp AES256 Pipeline.
- Một số phần mở rộng như tích hợp AES vào hệ thống SoC hoặc nghiên cứu AES-GCM chỉ được đề xuất ở phần định hướng phát triển.

1.5 Phân chia công việc nhóm

Bảng 1.1: Bảng phân công công việc

Thành viên	Công việc chi tiết
Vũ Thành Lam - 23520840	• Thiết kế pipeline cho kiến trúc • Key Expandsion • Mixcolumns • AddRounKey • Thuyết trình • Hỗ trợ viết báo cáo
Lê Trần Huỳnh Phong - 23521164	• Sbox • Viết báo cáo • Chạy 100 testcase • Tìm kiếm tài liệu • Hỗ trợ mô phỏng
Nguyễn Đức Toàn - 23521606	• Post simulation • Pre-simulation. • Shiftrow • Slide. • Chạy 100 testcase. • Hỗ trợ mô phỏng

Chương 2

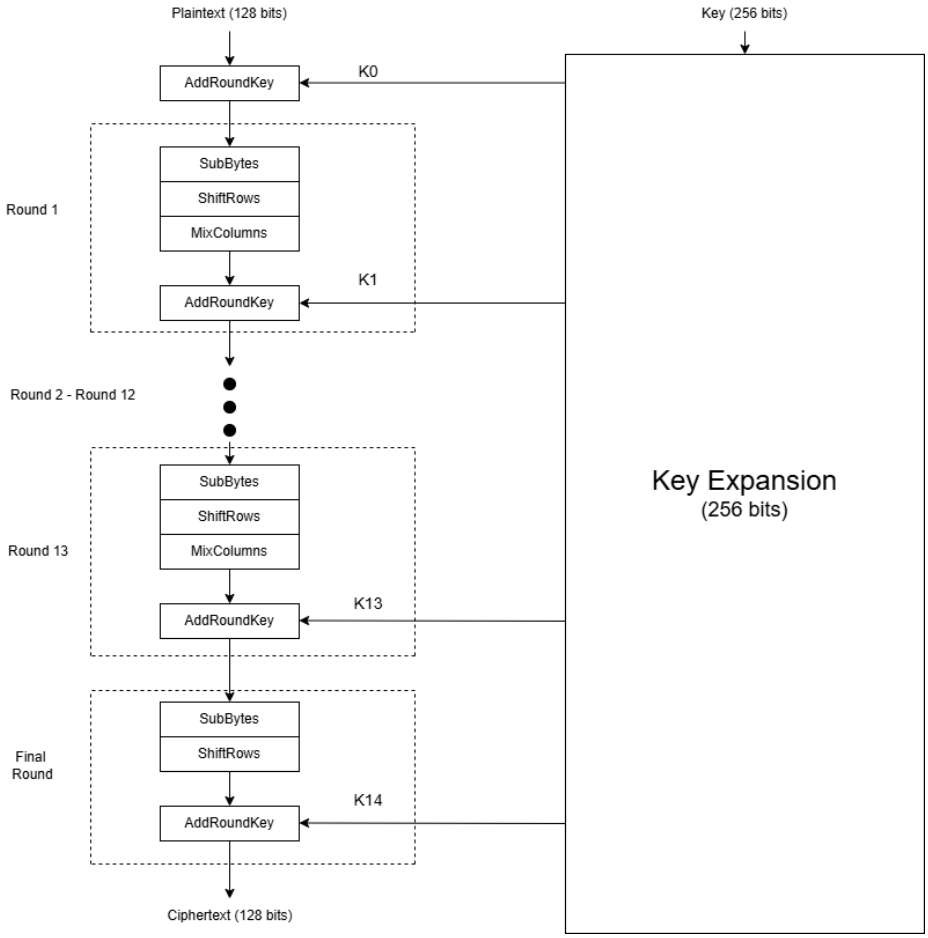
KIẾN TRÚC CỦA CHUẨN MÃ HÓA NÂNG CAO (AES256)

AES256 là phiên bản mạnh nhất trong bộ tiêu chuẩn AES với độ dài khóa 256 bit và 14 vòng xử lý. Kiến trúc AES256 dựa trên mô hình Substitution–Permutation Network (SPN), gồm nhiều phép biến đổi tuyến tính và phi tuyến được lặp lại theo từng vòng. Chương này trình bày chi tiết quy trình mã hóa và kiến trúc Pipeline của AES256 dưới góc độ triển khai phần cứng.

2.1 Quy trình mã hóa

AES256 hoạt động trên khối dữ liệu 128 bit và sử dụng khóa 256 bit để tạo ra các khóa vòng thông qua thuật toán Key Expansion. Toàn bộ quá trình mã hóa gồm:

- 1 vòng AddRoundKey ban đầu
- 13 vòng gồm đầy đủ 4 phép biến đổi
- 1 vòng cuối không có MixColumns



Hình 2.1: Cấu trúc mã hóa tổng quát

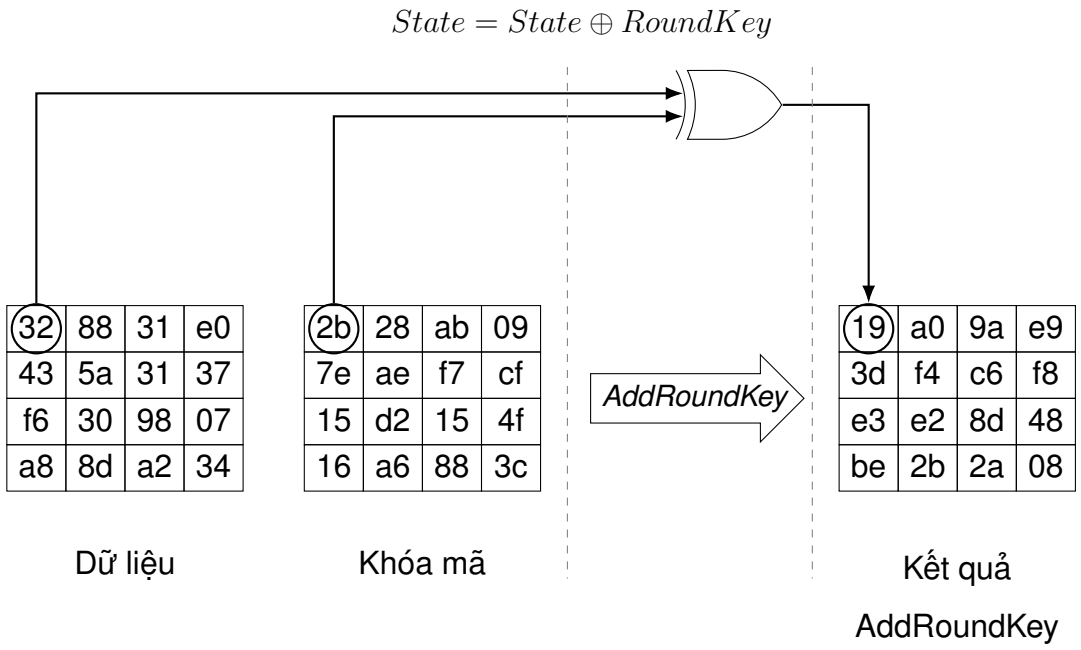
2.1.1 AddRoundKey

AddRoundKey là phép biến đổi đơn giản nhưng quan trọng nhất của AES. Ở bước này, khối dữ liệu trạng thái (State) được XOR trực tiếp với khóa vòng tương ứng.

• Chức năng

- Trộn khóa vào dữ liệu.
- Là bước duy nhất đưa yếu tố bí mật của thuật toán vào mỗi vòng mã hóa.
- Tăng độ an toàn bằng việc thay đổi dữ liệu liên tục theo khóa.

• **Mô tả**



Trong triển khai phần cứng, XOR được thực hiện song song trên 128 bit, giúp đạt tốc độ cao.

2.1.2 SubBytes Transformation

SubBytes là phép biến đổi phi tuyến duy nhất trong AES.

Bảng 2.1: Bảng S-box AES

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

•Chức năng

- Thay thế mỗi byte bằng giá trị trong S-box.
- Tạo tính phi tuyến, tăng khả năng chống các cuộc tấn công phân tích như differential hoặc linear cryptanalysis.

• Mô tả

Một bảng S-box 16×16 được dùng để ánh xạ 8-bit vào 8-bit.

Các bước tạo S-box dựa trên:

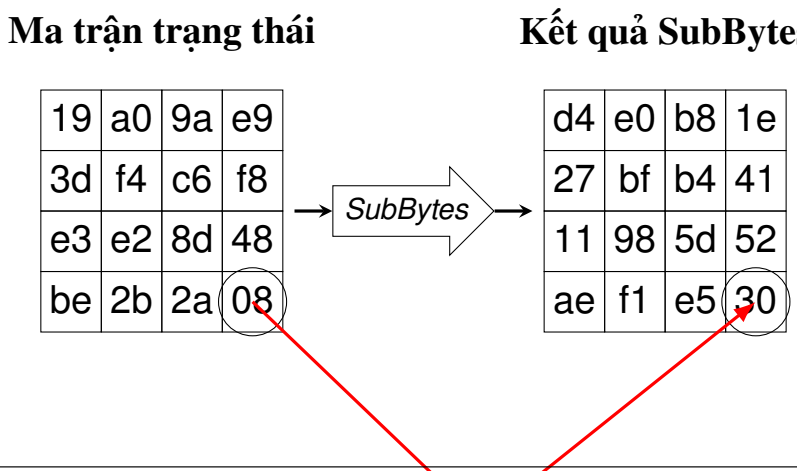
- Nghịch đảo trong trường GF (2^{128}).
- Ánh xạ affine để tăng tính bảo mật.

• Đặc điểm phần cứng

- Có thể triển khai bằng ROM 256 byte hoặc logic tổ hợp.
- Trong Pipeline, mỗi byte có thể xử lý độc lập $\rightarrow$ dễ song song hóa.

Ví dụ:

Hình 2.2: Trực quan hóa Phép toán SubBytes



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

2.1.3 ShiftRows

ShiftRows là phép biến đổi dịch vòng theo hàng của ma trận trạng thái 4×4 byte.

$$\begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} \Rightarrow \begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,1} & a_{1,2} & a_{1,3} & a_{1,0} \\ a_{2,2} & a_{2,3} & a_{2,0} & a_{2,1} \\ a_{3,3} & a_{3,0} & a_{3,1} & a_{3,2} \end{bmatrix}$$

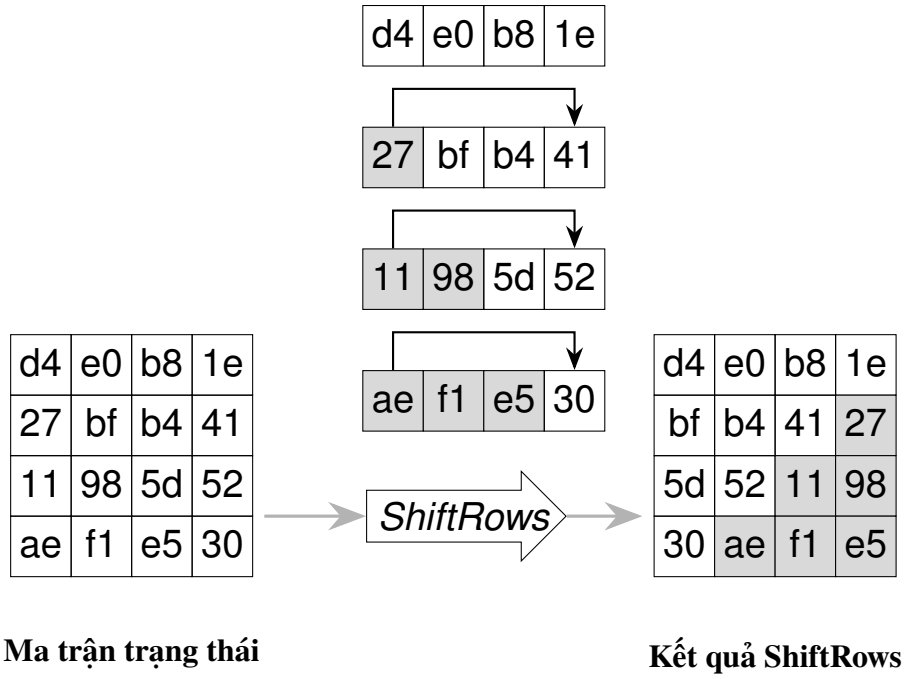
• Chức năng

- Tăng mức độ khuếch tán theo chiều ngang.
- Kết hợp với MixColumns để lan truyền thay đổi trên toàn khối dữ liệu.

• Quy tắc dịch

- Hàng 0: giữ nguyên
- Hàng 1: dịch trái 1 byte
- Hàng 2: dịch trái 2 byte
- Hàng 3: dịch trái 3 byte

Hình 2.3: Trực quan hóa Phép toán ShiftRows



• Đặc điểm phần cứng

• Chỉ là thao tác định tuyến (wiring). → Không tiêu tốn tài nguyên logic. → Thực hiện cực nhanh.

2.1.4 MixColumns

MixColumns thực hiện biến đổi tuyến tính trên từng cột của ma trận trạng thái.

• Chức năng

- Tăng độ khuếch tán theo chiều dọc.
- Kết hợp với ShiftRows để đảm bảo mỗi byte đầu vào ảnh hưởng đến toàn bộ khối đầu ra sau vài vòng.

• Mô tả toán học

Mỗi cột được nhân với ma trận cố định (Constant Matrix) trong trường $GF(2^8)$. Các giá trị trong ma trận này là hằng số được biểu diễn dưới dạng hệ cơ số 16:

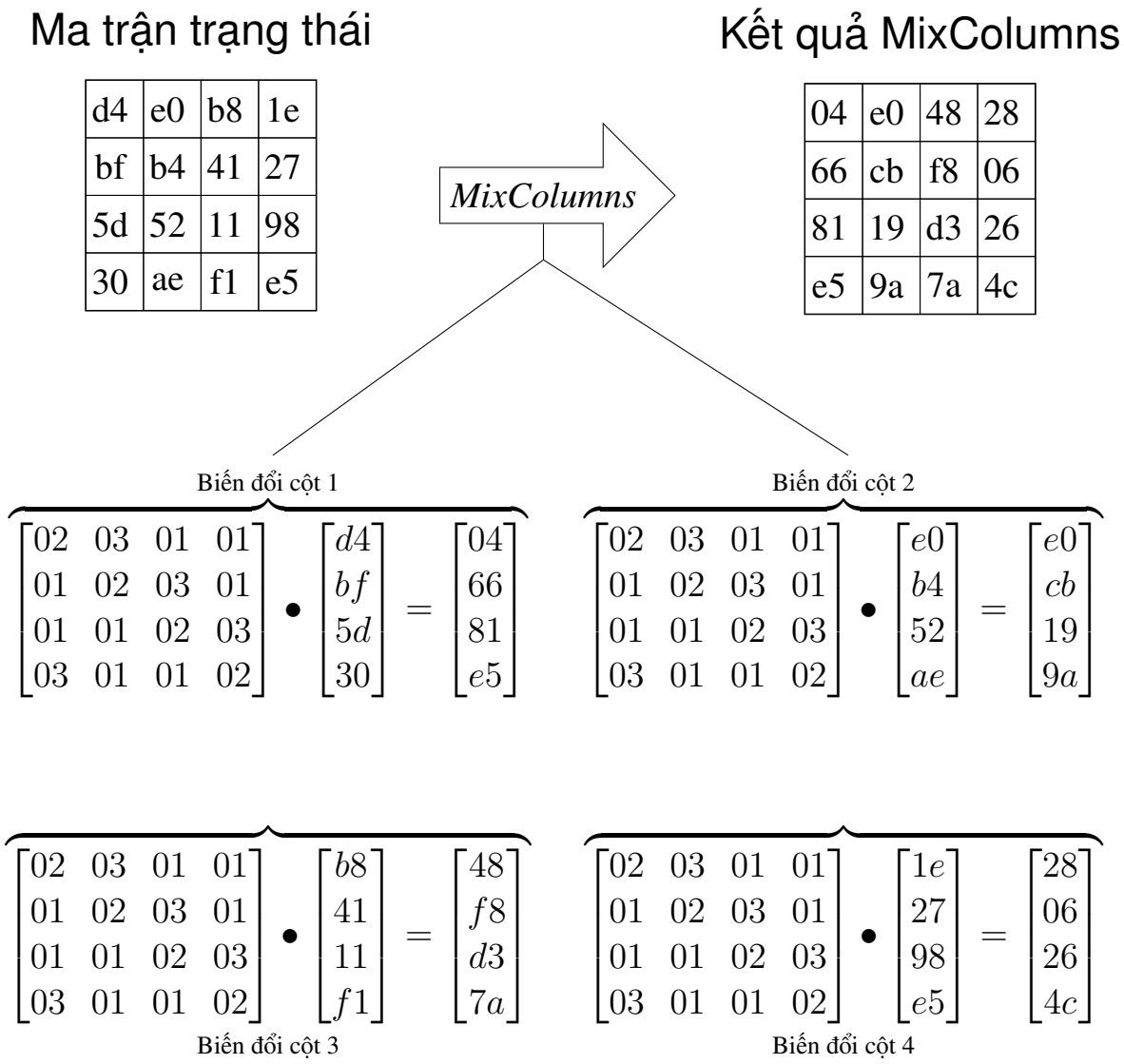
$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$$

Phép nhân ma trận này đảm bảo mỗi byte đầu ra phụ thuộc vào tất cả 4 byte đầu vào của cột, tạo ra khả năng khuếch tán dữ liệu (diffusion) mạnh mẽ.

• Đặc điểm phần cứng

- Phép nhân trong GF(2^{128}) cần các phép dịch và XOR.
- Có thể tối ưu bằng cách dùng LUT hoặc cấu trúc logic chuyên dụng.

Hình 2.4: Trực quan hóa phép toán MixColumns



2.1.5 Key Expansion

Key Expansion là quá trình mở rộng khóa từ khóa gốc 256-bit thành 15 round keys, mỗi round key có kích thước 128-bit, phục vụ cho 15 vòng mã hóa của AES-256.

Kiến trúc thiết kế

Module `Key_Expansion` nhận đầu vào:

- `in[255:0]`: Khóa gốc 256-bit
- `clk`: Xung clock đồng bộ

Xuất ra 15 round keys: `Key_0` đến `Key_14`, mỗi key 128-bit.

Quy trình mở rộng khóa

Bước 1: Khởi tạo Words

- Chia khóa 256-bit thành 8 words ban đầu (Word[0] đến Word[7])
- Mỗi word có kích thước 32-bit
- Cần sinh thêm 52 words (Word[8] đến Word[59]) để tạo đủ 60 words cho 15 round keys

Bước 2: Sinh Words mới

Với mỗi word thứ i ($i \geq 8$):

Trường hợp 1: $i \bmod 8 = 0$

$$Word[i] = \text{SubWord}(\text{RotWord}(Word[i-1])) \oplus Rcon[i/8 - 1] \oplus Word[i-8]$$

- RotWord: Xoay trái 1 byte (shift left circular)
- SubWord: Thay thế từng byte qua S-box
- Rcon: Round constant theo chỉ số vòng

Trường hợp 2: $i \bmod 8 = 4$

$$Word[i] = \text{SubWord}(Word[i-1]) \oplus Word[i-8]$$

Trường hợp 3: Các trường hợp còn lại

$$Word[i] = Word[i-1] \oplus Word[i-8]$$

Bước 3: Tổ chức Round Keys

- Mỗi Round Key được tạo từ 4 words liên tiếp
- `Key_0` = Word[0], Word[1], Word[2], Word[3]
- `Key_1` = Word[4], Word[5], Word[6], Word[7]
- ...

- `Key_14` = Word[56], Word[57], Word[58], Word[59]

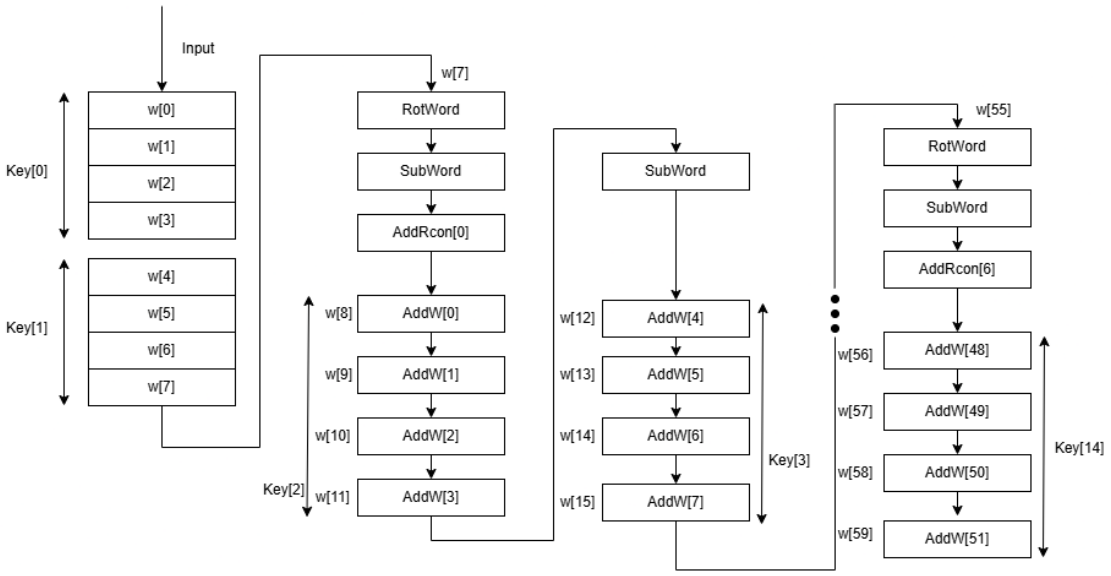
Round Constants (Rcon)

Round constants được sử dụng trong AES-256:

- Rcon[0] = 0x01000000
- Rcon[1] = 0x02000000
- Rcon[2] = 0x04000000
- Rcon[3] = 0x08000000
- Rcon[4] = 0x10000000
- Rcon[5] = 0x20000000
- Rcon[6] = 0x40000000

Các giá trị này được sinh theo quy tắc nhân 2 trong trường Galois $GF(2^8)$.

Đồng bộ hóa



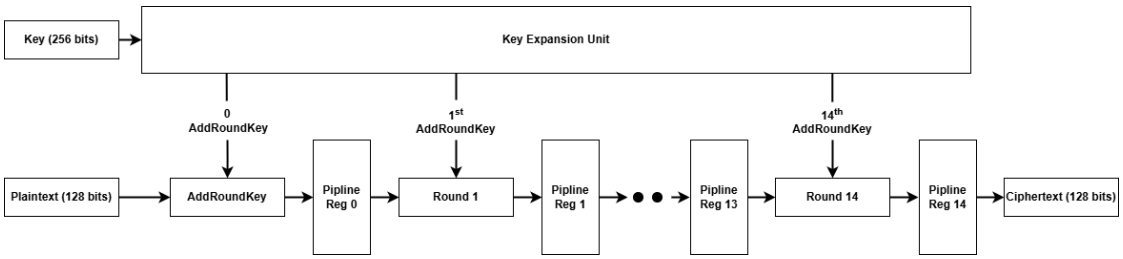
Hình 2.5: Trực quan phép toán Key Expansion

2.2 Kiến trúc Pipeline AES

2.2.1 Mô hình Pipeline tổng quát

Thiết kế AES256 Pipeline tối ưu hiệu suất thường là Pipeline Toàn phần (Full Pipelining).

- Cấu trúc: Kiến trúc Pipeline toàn phần nhân bản toàn bộ 15 vòng xử lý của AES256 (Round 0 đến Round 14) thành 15 giai đoạn logic tuần tự.
- Thanh ghi Pipeline: Các thanh ghi 128 bit (Register) được chèn vào giữa các vòng (Round N và Round $N+1$) để đồng bộ hóa dữ liệu và lưu trữ dữ liệu cho xung clock kế tiếp. Nhờ các thanh ghi này, ngay khi Round N hoàn thành, Round $N+1$ có thể bắt đầu xử lý, trong khi Round N chuẩn bị nhận khối dữ liệu mới.
- Round Block: Mỗi khối Round (trừ Round cuối) gồm 4 phép biến đổi: SubBytes $\rightarrow$ ShiftRows $\rightarrow$ MixColumns $\rightarrow$ AddRoundKey. Đây là Critical Path của mạch quyết định tần số hoạt động (F_{max}) của toàn bộ mạch. Tần số hoạt động tối đa mà mạch có thể hoạt động mà không bị fail timing hay nói cách khác là Slack ở tần số này thì nó 0, muốn mạch hoạt động chính xác thì $Slack \geq 0$.



Hình 2.6: Sơ đồ minh họa kiến trúc AES256 Pipeline toàn phần

2.2.2 Quản lý Khóa trong Pipeline (Key Path Management)

Trong kiến trúc Pipeline toàn phần, thách thức lớn nhất là đảm bảo mỗi vòng mã hóa (Round Block) nhận được Khóa Vòng (Round Key) tương ứng đúng vào chu kỳ đồng hồ (Clock Cycle) mà khối dữ liệu đi vào Round đó.

A. Mô hình Tuần tự (Sequential Key Generation)

Thay vì triển khai Key Expansion dưới dạng logic tổ hợp tốn kém (tính toán 13 khóa cùng lúc, Delay sẽ là từ Word[7] đến Word[59]), đồ án này sử dụng mô hình tuần tự để giảm

độ delay của mạch và tăng (F_{max}):

- Tính toán tuần tự: Thuật toán Key Expansion (2.1.5) được triển khai trên một khối logic lặp, chỉ tính toán 4 Word($W[i]$) hoặc một Khóa Vòng (K_i) sau mỗi chu kỳ clock.
- Thời gian sinh Key: Với 13 Khóa Vòng, sẽ mất khoảng 14 chu kỳ clock để sinh ra toàn bộ các khóa cần thiết (K_2 đến K_{14}) từ khóa gốc 256 bit.

B. Quản lý Khóa trong Pipeline

Chiến lược quản lý khóa

- Module `Key_Expansion` sinh tất cả 13 round keys song song với dữ liệu.
- Các round keys được kết nối trong wire array RoundKey[0:14].
- Mỗi stage pipeline truy cập trực tiếp vào round key tương ứng thông qua kết nối wire.
- Roundkey được lấy ra tương ứng cho từng Round, giảm độ phức tạp thiết kế.

C. Lợi ích

- Sau 14 clock cycles đầu tiên, tất cả round keys đã sẵn sàng và ổn định
- Không có hazard liên quan đến khóa trong quá trình hoạt động ổn định
- Kiến trúc đơn giản với kết nối wire trực tiếp

2.2.3 Ưu điểm của Pipeline (Tăng cường)

- Tăng Throughput (Thông lượng) Hệ thống: Dữ liệu mới có thể được nạp vào hệ thống mỗi chu kỳ clock (với Full Pipeline)3. Do đó, tốc độ mã hóa lý thuyết bằng:

$$Throughput = F_{\max} \times \frac{\text{số tầng} \times 128 \text{ bits}}{\text{latency}} = [\text{Giá trị}] \text{ Gbps}$$

- Tăng Tần số Hoạt động (F_{max}): Bằng cách chia quá trình tính toán của 14 vòng thành 14 giai đoạn, thời gian trễ lớn nhất (Critical Path) của toàn mạch giảm xuống chỉ còn bằng độ trễ của một vòng AES.

2.2.4 Thách thức khi triển khai Pipeline (Chi tiết hóa)

- Tăng Tài nguyên Logic: Việc nhân bản 14 khối Round Block và chèn các thanh ghi Pipeline làm tăng đáng kể mức sử dụng tài nguyên logic (LUTs và Registers) trên FPGA.

- Độ trễ (Latency) Ban đầu: Mặc dù Throughput cao, khối dữ liệu đầu tiên vẫn cần 14 chu kỳ clock để đi qua toàn bộ Pipeline và xuất ra Ciphertext cuối cùng.

- Độ phức tạp của Key Path: Việc xử lý Key Expansion song song để cung cấp 15 Round Key đồng thời yêu cầu mạch logic phức tạp hơn và tiêu thụ nhiều tài nguyên hơn cho việc lưu trữ và phân phối khóa.

Chương 3

TỔNG QUAN VỀ PHẦN MỀM MÔ PHỎNG VÀ PHẦN MỀM THIẾT KẾ

Chương này giới thiệu tổng quan về các công cụ phần mềm và ngôn ngữ mô tả phần cứng được sử dụng xuyên suốt quá trình thực hiện đồ án, từ khâu thiết kế logic, mô phỏng chức năng đến tổng hợp và đánh giá hiệu năng trên nền tảng FPGA.

3.1 Tổng quan về phần mềm thiết kế và mô phỏng Vivado

Vivado Design Suite là một môi trường phát triển tích hợp (Integrated Development Environment - IDE) mạnh mẽ do Xilinx phát triển, được sử dụng để thiết kế, tổng hợp và triển khai các hệ thống dựa trên FPGA (Field-Programmable Gate Array) và Zynq SoC (System on Chip) của hãng.

Vai trò trong Đồ án

- Tổng hợp (Synthesis): Chuyển đổi mã mô tả phần cứng Verilog thành mạch logic cổng (gate-level netlist) và ánh xạ nó lên các tài nguyên chuyên biệt của FPGA (LUTs, Flip-Flops, DSPs).
- Tiển khai (Implementation): Thực hiện quá trình Placement (đặt các khối logic vào vị trí vật lý) và Routing (kết nối các khối) trên chip FPGA mục tiêu.
- Phân tích Thời gian Tĩnh (Static Timing Analysis - STA): Xác định các đường trễ quan trọng (Critical Path) của thiết kế Pipeline, từ đó tính toán tần số hoạt động tối đa (F_{max}) mà mạch có thể đạt được.

- Báo cáo Hiệu năng và Tài nguyên: Tạo ra các báo cáo chi tiết về mức độ sử dụng tài nguyên (Utilization Report) và thực hiện phân tích năng lượng tiêu thụ mô hình (Power Analysis) để đánh giá chi phí phần cứng của thiết kế.

- Mô phỏng Pre-Synthesis (Pre-Simulation): Kiểm tra tính đúng đắn về mặt logic và chức năng của mã Verilog trước khi tổng hợp. Điều này bao gồm việc kiểm tra từng khối chức năng (SubBytes, MixColumns, Key Expansion) và toàn bộ kiến trúc AES256 Pipeline.

- Testbench và Vector Kiểm thử: ModelSim được sử dụng để chạy Testbench, trong đó các khối dữ liệu đầu vào (Plaintext) và khóa (Key) chuẩn được cung cấp. Kết quả đầu ra của mô hình được so sánh với các Vector Kiểm thử (Test Vectors) đã được tính toán sẵn từ các công cụ mã hóa online hoặc chuẩn NIST.

- Gỡ lỗi (Debugging): Cung cấp giao diện dạng sóng (waveform) trực quan, cho phép nhà thiết kế quan sát sự thay đổi giá trị của các tín hiệu bên trong mạch tại từng chu kỳ clock. Đây là công cụ thiết yếu để tìm và sửa lỗi trong logic tuần tự (Sequential Logic) của Pipeline và Key Schedule.

3.2 Tổng quan về phần mềm tổng hợp Quartus II và Quartus Prime Lite

Vai trò trong Đồ án

- Công cụ Tổng hợp Thay thế/Đối chứng: Quartus Prime được sử dụng để tổng hợp thiết kế AES256 trên các dòng FPGA của Intel, nhằm đối chứng (cross-check) mức độ sử dụng tài nguyên và hiệu năng của cùng một mã Verilog trên các kiến trúc FPGA khác nhau (Xilinx và Intel).

- Tối ưu hóa: Bộ công cụ này cung cấp các tính năng tối ưu hóa chuyên biệt cho kiến trúc logic của chip Intel, giúp nhà thiết kế tinh chỉnh cấu trúc pipeline để đạt F_{max} cao hơn hoặc giảm thiểu tài nguyên sử dụng.

- Phân tích Chi tiết: Giống như Vivado, Quartus Prime cung cấp báo cáo Utilization và Timing Analysis để xác định chính xác hiệu quả của kiến trúc Pipeline đã triển khai.

3.3 Tổng quan về ngôn ngữ Verilog

Verilog HDL (Hardware Description Language) là một ngôn ngữ mô tả phần cứng tiêu chuẩn được sử dụng rộng rãi trong thiết kế các hệ thống điện tử số (Digital Systems).

Vai trò trong Đồ án

- Mô hình hóa Kiến trúc: Verilog được sử dụng để mô hình hóa toàn bộ kiến trúc AES256 Pipeline, bao gồm 15 giai đoạn xử lý dữ liệu và khối Key Schedule tuần tự.
- Triển khai Logic Tổ hợp: Các phép biến đổi nhanh như AddRoundKey (XOR) và ShiftRows (Wiring) được mô tả bằng logic tổ hợp thuần túy (Combinational Logic).
- Triển khai Logic Tuần tự: Các thanh ghi Pipeline và Bộ điều khiển (Controller) được mô tả bằng logic tuần tự (Sequential Logic), cho phép chúng lưu trữ trạng thái và đồng bộ hóa hoạt động của các khối trong môi trường Pipeline dựa trên tín hiệu clock.
- Tối ưu hóa Phần cứng: Việc sử dụng Verilog cho phép nhà thiết kế kiểm soát chi tiết cách thức mạch logic được ánh xạ lên tài nguyên FPGA, giúp tối ưu hóa thiết kế Pipeline để đạt được tần số hoạt động cao nhất (F_{max}) và thông lượng (Throughput) mong muốn.

Chương 4

TRIỂN KHAI THIẾT KẾ PHẦN CỨNG AES256 VỚI KỸ THUẬT PIPELINE

Chương 4 trình bày quá trình mô hình hóa thuật toán AES256 thành mã Verilog và đánh giá kết quả trên nền tảng FPGA. Thiết kế được triển khai theo kiến trúc Full Pipeline 15 giai đoạn (Round 0 đến Round 14) nhằm tối đa hóa thông lượng (throughput)

4.1 Triển khai kiến trúc Pipeline AES

Kiến trúc được triển khai trong module `top.v` là sự kết hợp của 15 khối xử lý vòng (Round Block) độc lập, được nối tiếp bằng các thanh ghi `State_Matrix`.

4.1.1 Add Round Key

Phép biến đổi này được thực hiện bằng cách XOR logic 128 bit giữa đầu ra của giai đoạn trước đó và Khóa Vòng tương ứng.

- **Triển khai Phần cứng:** Khối AddRoundKey được mô tả hoàn toàn bằng **logic tổ hợp** (combinational logic) sử dụng phép toán XOR (`^`) trên 128 bit, giúp đạt tốc độ xử lý tức thời.
- **Ví dụ Verilog:**

```
assign State_Matrix_In[N] = after_MixColumns[N-1] ^ RoundKey[N];
```

- **Vị trí trong Pipeline:** Đây là bước cuối cùng trong mỗi Round Block trước khi dữ liệu được ghi vào các thanh ghi Pipeline (`State_Matrix`) ở chu kỳ tiếp theo.

4.1.2 SubBytes Transformation

Phép biến đổi phi tuyến SubBytes được thực hiện song song trên 16 byte (128 bit) của khối dữ liệu.

Triển khai: Sử dụng 16 module sbox độc lập (logic tổ hợp), mỗi module thực hiện tra cứu 8-bit trên Bảng S-box được cài đặt dưới dạng Lookup Table (LUT). Việc sử dụng LUT đảm bảo tốc độ xử lý nhanh nhất.

4.1.3 ShiftRows

Phép dịch vòng theo hàng được thực hiện để khuếch tán dữ liệu theo chiều ngang.

- **Triển khai:** Đây là thao tác định tuyến (Wiring) và không sử dụng logic tính toán nào (không tiêu tốn LUTs), giúp thực hiện cực kỳ nhanh.
- **Cấu trúc dịch:** Dữ liệu được sắp xếp lại bằng cách gán dây (wire assignment) theo quy tắc dịch 0, 1, 2, 3 byte cho các hàng 0, 1, 2, 3 tương ứng.

4.1.4 MixColumns

Phép biến đổi tuyến tính này được thực hiện trên 4 cột song song. Đây là khối logic phức tạp nhất.

- **Triển khai:** Khối `MixColumns` được xây dựng hoàn toàn bằng logic tổ hợp. Phép nhân ma trận trong trường $GF(2^8)$ được phân tách thành các phép toán dịch trái, XOR và điều chỉnh theo đa thức rút gọn (8'h1b).
  - Sử dụng module `Mul_By_2` và `Mul_By_3`. Cụ thể, `Mul_By_3` được triển khai dưới dạng $3x = 2x \oplus x$
- **Critical Path:** Do tính phức tạp của phép nhân, khối MixColumns là thành phần có độ trễ lớn nhất trong mỗi Round Block, do đó nó là yếu tố chính giới hạn tần số hoạt động tối đa (F_{max}) của toàn bộ thiết kế.

4.1.5 Key Expansion

Thiết kế sử dụng một kiến trúc tổ hợp kết hợp tuần tự cho Key Expansion.

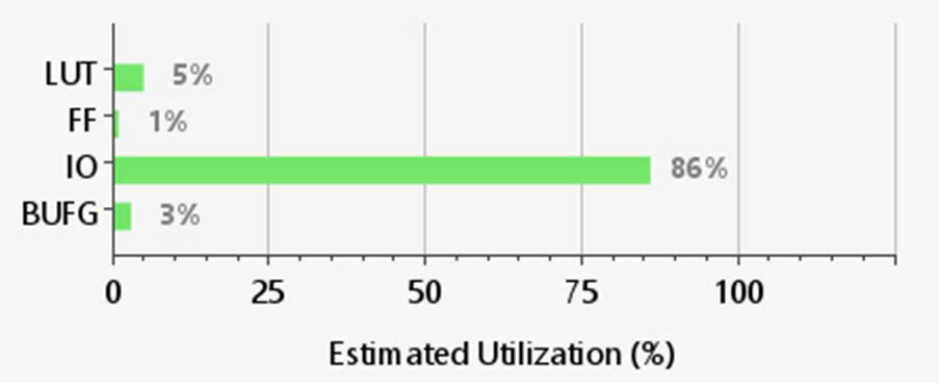
- **Cấu trúc triển khai:** Toàn bộ 60 từ 32 bit ($W[0]$ đến $W[59]$) được tính toán bằng logic tổ hợp và lưu trữ trong một mảng thanh ghi Word 32-bit (chuyển đổi từ `Word_in` sang Word sau mỗi posedge clk).
- **Đầu ra Khóa Vòng (Round Keys):**
  - K_2 đến K_{14}: Lấy từ giá trị thanh ghi Word đã được tính toán ở chu kỳ trước.
  - K_0 **và** K_1 **(Khóa gốc):** Lấy trực tiếp từ đầu vào in (256-bit Key).

4.2 Kết quả tổng hợp của thiết kế

Phần này trình bày các số liệu thực tế thu được từ quá trình tổng hợp (Synthesis) thiết kế AES256 Full Pipeline trên nền tảng FPGA thiết bị mục tiêu Xilinx Virtex-7. Các kết quả này phản ánh chi phí phần cứng và hiệu suất năng lượng của kiến trúc đã triển khai.

4.2.1 Mức độ sử dụng tài nguyên

Kiến trúc Full Pipeline là thiết kế tối đa hóa thông lượng (throughput) bằng cách nhân bản 15 khối xử lý vòng (Round Block) độc lập. Do đó, mặc dù tỷ lệ sử dụng tổng thể trên chip Virtex-7 thấp (do kích thước chip lớn), mức tiêu thụ các tài nguyên logic cụ thể vẫn phải được phân tích chi tiết.



Resource	Estimation	Available	Utilization %
LUT	13145	257600	5.10
FF	3944	515200	0.77
IO	515	600	85.83
BUFG	1	32	3.13

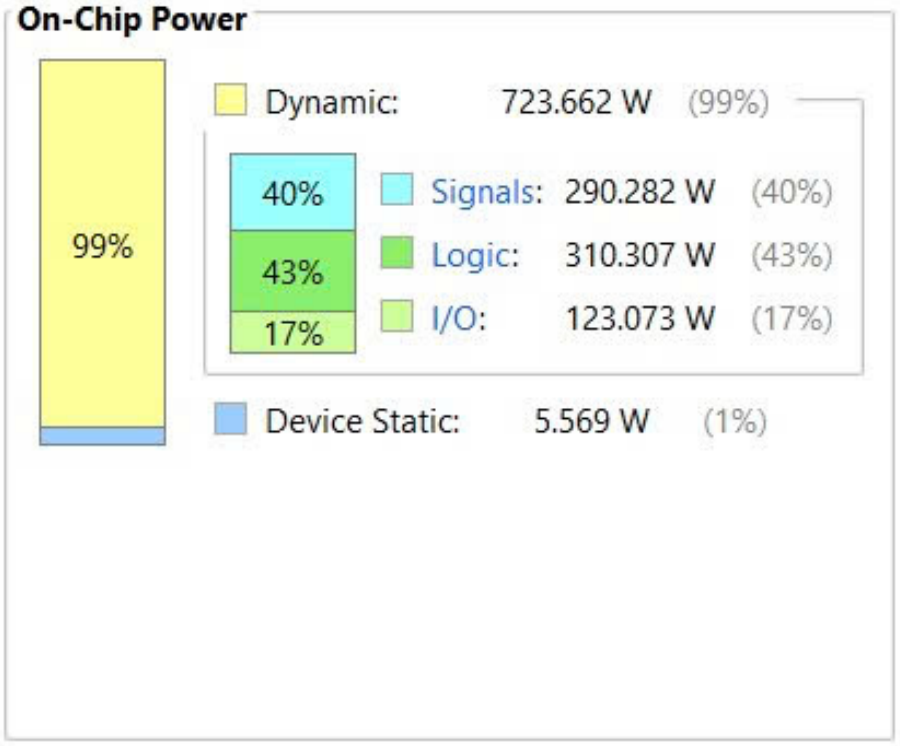
Hình 4.1: Bảng so sánh mức sử dụng tài nguyên

Phân tích Tài nguyên

- Sử dụng Logic (LUT và FF): Tỷ lệ sử dụng LUT (5.10%) và FF (0.77%) trên chip Virtex-7 cho thấy thiết kế sử dụng tài nguyên logic hiệu quả.
  - Việc tiêu thụ 13,145 LUT chủ yếu đến từ các phép toán phức tạp (như MixColumns và SubBytes) được nhân bản 15 lần trong kiến trúc Pipeline, cùng với logic Key Expansion bán tổ hợp.
  - Sử dụng 3,944 FF là chi phí cần thiết để triển khai các thanh ghi Pipeline (Register Array) giữa các giai đoạn, đảm bảo đồng bộ hóa luồng dữ liệu và duy trì Throughput là 1 khối/chu kỳ.
- Chân I/O (Input/Output): Tỷ lệ sử dụng IO cao (85.83%) là hợp lý và cần thiết cho thiết kế. Điều này phản ánh yêu cầu truyền tải đồng thời khóa 256-bit (Key), khối dữ liệu 128-bit (Plaintext) và các tín hiệu điều khiển cần thiết cho hoạt động song song của mạch.
- Bộ đệm Clock (BUFG): Thiết kế chỉ sử dụng 1 BUFG (3.13%), chứng tỏ tín hiệu clock được quản lý và phân phối ổn định, hiệu quả đến toàn bộ 15 giai đoạn Pipeline, một yếu tố quan trọng để đạt F_{max} cao.

4.2.2 Phân tích năng lượng tiêu thụ

Kết quả phân tích năng lượng (Power Analysis) được ước tính bằng công cụ tổng hợp (Vivado Power Analyzer) cho mô hình Full Pipeline khi hoạt động liên tục ở tần số tối đa.



Hình 4.2: Bảng kết quả phân tích năng lượng (Power Analysis)

Phân tích Năng lượng Tiêu thụ

- Đặc trưng của Dynamic Power: Dynamic Power (723.662 W) chiếm 99% tổng công suất. Đây là đặc điểm nổi bật của các thiết kế ASIC/FPGA tốc độ cao và logic tổ hợp lớn như Full Pipeline, nơi các cổng logic chuyển mạch liên tục ở tần số cao để duy trì thông lượng.
- Phân bổ Năng lượng Động: Công suất động chủ yếu tập trung vào các khối tính toán logic:
  - Logic (43%) và Signals (40%): Tổng cộng 83% công suất động được dành cho Logic tính toán (các khối MixColumns và SubBytes) và việc chuyển đổi tín hiệu trên các đường dây kết nối (Signals). Điều này phản ánh chi phí lớn nhất của việc duy trì hoạt động song song của 15 Round Block.
  - I/O (17%): Chi phí I/O tương đối cao là do tần số giao tiếp I/O cao và sự cần thiết phải truyền dữ liệu 128-bit liên tục vào/ra.
- Năng lượng Tĩnh (Static Power): Mức năng lượng tĩnh 5.569 W (1% tổng công suất) là thấp, cho thấy công nghệ chip Virtex-7 hiệu quả trong việc duy trì trạng thái chờ.

4.3 Kết quả Simulation của thiết kế

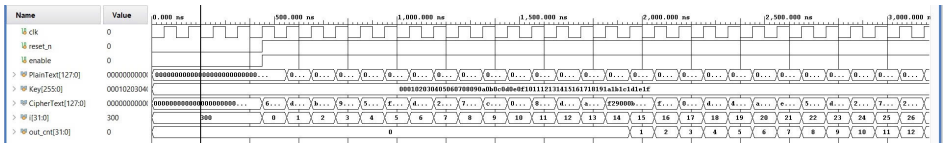
Mục này trình bày kết quả xác minh chức năng (functional verification) của thiết kế AES256 Pipeline thông qua mô phỏng. Các mô phỏng được thực hiện trên ModelSim/Vivado Simulator, sử dụng Test Vector chuẩn NIST (PlainText: 00...FF, Key: 00...0F cho AES-128) để kiểm tra tính đúng đắn và hiệu suất thời gian của mạch.

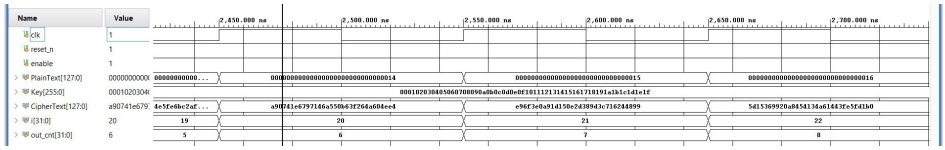
4.3.1 Mô phỏng Pre-Simulation

Xác nhận tính đúng đắn của logic trước khi tổng hợp bằng Vivado.

- Kiểm tra luồng dữ liệu: Xác nhận khối dữ liệu đầu tiên được nạp và đi qua 15 thanh ghi Pipeline tuần tự.

Testcase



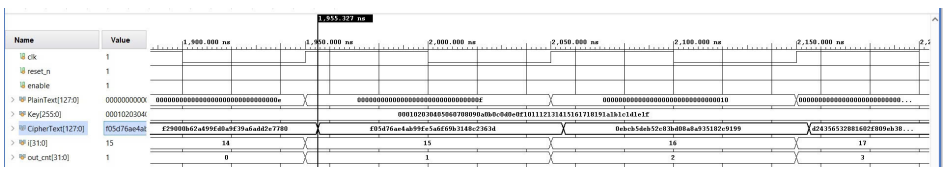


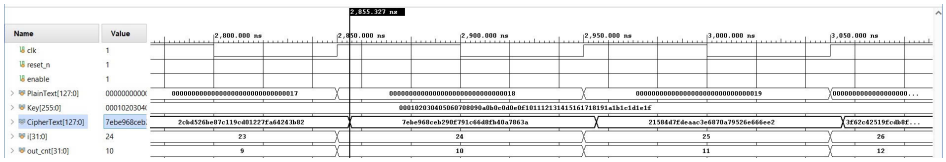
4.3.2 Mô phỏng Post-Simulation

Kiểm tra hoạt động của mạch sau khi tính toán trễ (Timing Back-Annotation).

Mục tiêu: Đảm bảo thiết kế vẫn hoạt động chính xác ở việc có thêm delay.

Testcase

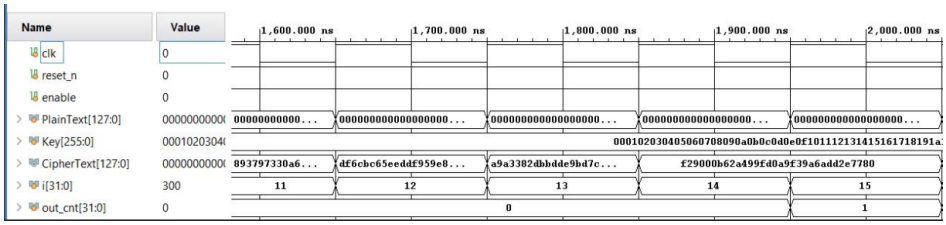


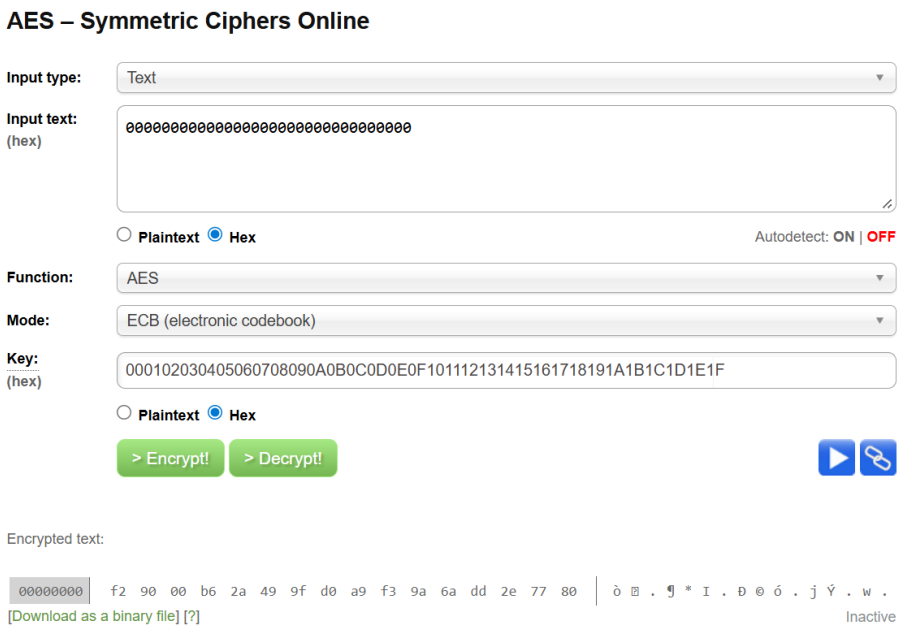


4.3.3 Kiểm tra kết quả mô phỏng với công cụ tính toán Online

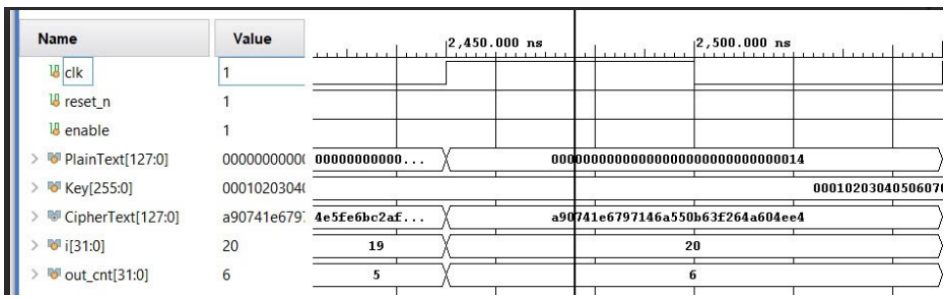
Testcase

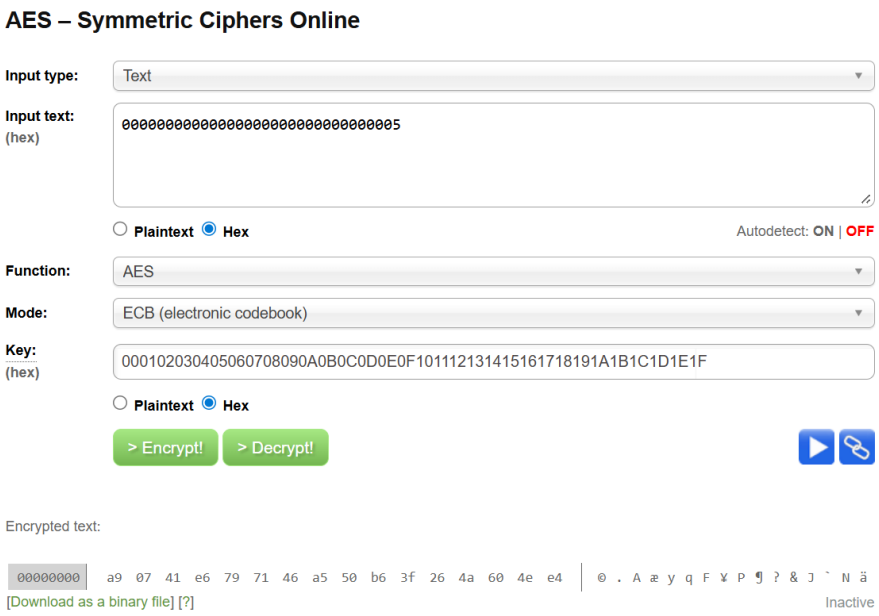
Test 1:





Test 2:





Bảng 4.1: Bảng so sánh kết quả mô phỏng và công cụ Online

Plaintext (Hex)	**Key (256-bit Hex)**	**Ciphertext Chuẩn (Hex)**	**Ciphertext Mô phỏng (Hex)**
[Input 1]	[Key 1]	[Output 1]	[Output 1]
000000000000 000000000000 000000000000	0001020304050607080 90A0B0C0D0E0F_101112 131415161718191A1B1C 1D1E1F	f29000b62a499f d0a9f39a6add2 e7780	f29000b62a499fd0a9f39a6 add2e7780
[Input 2]	[Key 2]	[Output 2]	[Output 2]
000000000000 000000000000 000000000005	0001020304050607080 90A0B0C0D0E0F_101112 131415161718191A1B1C 1D1E1F	a90741e6797146a550b63f2 64a604ee4	a90741e6797146a550b63f2 64a604ee4

4.4 Đánh giá thời gian hoạt động tĩnh của mạch

4.4.1 Kiểm tra độ trễ của mạch

- **Tần số Tối đa ($F_{\max}$):** $F_{\max}$ được giới hạn bởi thời gian trễ lớn nhất của một giai đoạn xử lý (Round Block), thường là khối MixColumns.

$$F_{\max} = \frac{1}{T_{\text{critical\_path}}} = [\text{Giá trị}]\ \text{MHz}$$

Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	139.06 MHz	139.06 MHz	clk	

Hình 4.3: Fmax của mô hình AES256 được tổng hợp bằng Quatus II

Clock Summary

Name	Waveform	Period (ns)	Frequency (MHz)
clk	{0.000 1.420}	2.840	352.113

Hình 4.4: Fmax của mô hình AES256 được tổng hợp bằng Vivado

- **Thông lượng (Throughput) được tính theo Fmax trên Quartus II:**

$$Throughput = F_{\max} \times \frac{\text{số tầng} \times 128 \text{ bits}}{29} = 139MHz \times \frac{15 \times 128 \text{ bits}}{\text{latency}} = 9203 \text{ Gbps}$$

- **Thông lượng (Throughput) được tính theo Fmax trên Vivado:**

$$Throughput = F_{\max} \times \frac{\text{số tầng} \times 128 \text{ bits}}{29} = 352MHz \times \frac{15 \times 128 \text{ bits}}{\text{latency}} = 23304 \text{ Gbps}$$

- **Độ trễ (Latency):** Latency là số chu kỳ clock cần thiết để khối dữ liệu đầu tiên đi qua toàn bộ 15 tầng Pipeline.

$$Total\ Latency = 29 \text{ chu kỳ clock}$$

Chương 5

TỔNG KẾT VÀ ĐỊNH HƯỚNG PHÁT TRIỂN

5.1 Tổng kết

Đồ án đã thành công trong việc nghiên cứu chuyên sâu thuật toán mã hóa Advanced Encryption Standard (AES), tập trung vào phiên bản khóa AES256, và áp dụng kiến trúc Pipeline trên nền tảng phần cứng số.

Thành quả chính:

- Mô hình hóa thành công các phép biến đổi cốt lõi của AES256 (SubBytes, ShiftRows, MixColumns, AddRoundKey, Key Expansion) bằng ngôn ngữ Verilog HDL.
- Triển khai kiến trúc Full Pipeline 15 giai đoạn để tối đa hóa thông lượng (Throughput) của mạch mã hóa, vượt trội so với kiến trúc lặp (Iterative) truyền thống ở chỗ có thể xử lý 15 PlainText cùng lúc.
- Thiết kế hệ thống Key Expansion theo mô hình tổ hợp kết hợp tuần tự, giảm thiểu mức sử dụng tài nguyên logic trên FPGA, đồng thời đảm bảo việc phân phối khóa song song cho Pipeline.
- Xác minh tính đúng đắn của thiết kế thông qua mô phỏng (ModelSim) và đối chiếu với các vector kiểm thử chuẩn.

5.2 Hạn chế của đề tài

Mặc dù đã đạt được mục tiêu về hiệu năng, đồ án vẫn tồn tại một số giới hạn như đã được xác định ngay từ đầu:

- Hạn chế về Chức năng: Đồ án chỉ tập trung vào quá trình Mã hóa (Encryption) của AES256. Quá trình giải mã (Decryption) chưa được triển khai do thời gian yêu cầu gấp rút.
- Hạn chế về Chế độ Hoạt động: Mô hình chỉ triển khai AES256 ở dạng khối chuẩn ECB, không tích hợp các chế độ hoạt động phức tạp như CBC, CFB, hoặc GCM.
- Hạn chế về Tối ưu hóa:
- Chưa triển khai các cải tiến nâng cao như Pipeline sâu (thay vì dùng LUT trong sbox thì sẽ tính sbox bằng logic và dùng 7 stage pipeline trong Sbox).
- Hạn chế về Kiểm thử: Không thực hiện kiểm thử nâng cao như kiểm thử ngẫu nhiên hoặc phân tích kênh kề (Side-Channel Analysis).

5.3 Định hướng phát triển

Dựa trên những hạn chế và tiềm năng của kiến trúc Pipeline đã triển khai, các hướng phát triển trong tương lai có thể bao gồm:

- Triển khai Chức năng Giải mã (Decryption): Bổ sung quá trình giải mã (Inverse Cipher) để hoàn thiện chức năng của bộ mã hóa/giải mã AES256.
- Tích hợp Chế độ Hoạt động (Modes of Operation): Nâng cấp kiến trúc để hỗ trợ các chế độ mã hóa phổ biến như AES-GCM (Galois/Counter Mode), cung cấp cả Mã hóa và Xác thực Dữ liệu.
- Tối ưu hóa Hiệu năng Chuyên sâu: Nghiên cứu áp dụng kỹ thuật Pipeline Sbox 7 stage thay cho LUT) hoặc Tối ưu hóa song song hóa dữ liệu (parallel processing) để tăng Throughput hơn nữa.

- Triển khai trên ASIC và Tích hợp SoC: Đề xuất nghiên cứu triển khai ở cấp độ ASIC hoặc tích hợp module AES vào một hệ thống trên chip (SoC) hoàn chỉnh, kết nối với bus hệ thống.
- Phân tích Bảo mật Phần cứng: Thực hiện các nghiên cứu và kiểm thử chuyên sâu về khả năng chống lại tấn công kênh kề (Side-Channel Attacks) để đảm bảo tính bảo mật của thiết kế.

TÀI LIỆU THAM KHẢO

[1] National Institute of Standards and Technology (NIST). *Advanced Encryption Standard (AES)*. Federal Information Processing Standards Publication 197, 2001.

[2] MIT 6.111 Staff. *Pipelining & Verilog (Lecture 9)*. Course Lecture Notes, 2016.

[3] Kool, M. *Design and Implementation of a 256-bit AES-GCM encryption device for network traffic using an FPGA-based SmartNIC*. Master's Thesis, Eindhoven University of Technology, 2024.

[4] Lin, S. H. et al. *Hardware Implementation of High-Throughput S-Box in AES for Information Security*. IEEE Access, Vol. 11, pp. 59050–59058, 2023.

[5] Gokul, R., Swarnalatha, A. *Pipelined AES-128 Encryption and Decryption Design Using Verilog HDL*. International Journal for Multidisciplinary Research (IJFMR), Vol. 3, Issue 7, pp. 216–220, 2021.

[6] Mishra, A. *Pipeline Implementation of AES Algorithm for Improved Resource Management and Higher Throughput*. Master's Thesis, Dalhousie University, 2023.