ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Đồng Phạm Khôi

GIẢI PHÁP BẢO MẬT BẰNG PHẦN CỨNG CHO CÁC THIẾT BỊ INTERNET-OF-THINGS.

Chuyên ngành: Kỹ thuật Điện tử
Mã số: 9510302.01

TÓM TẮT LUẬN ÁN TIẾN SĨ
CÔNG NGHỆ KỸ THUẬT ĐIỆN TỬ - VIỄN THÔNG

HÀ NỘI – 2022

Công trình được hoàn thành tại: Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội

Người hướng dẫn khoa học: 1. PGS. TS. Trần Xuân Tú

2. TS. Nguyễn Kiêm Hùng

Phản biện: .....

.....

Phản biện: .....

.....

Phản biện: .....

.....

Luận án sẽ được bảo vệ trước Hội đồng cấp Đại học Quốc gia chấm luận án tiến sĩ họp tại .....

vào hồi giờ ngày tháng năm

Có thể tìm hiểu luận án tại:

- Thư viện Quốc gia Việt Nam
- Trung tâm Thông tin - Thư viện, Đại học Quốc gia Hà Nội

TÓM TẮT LUẬN ÁN

Internet vạn vật (*Internet of Things - IoT*) đã mở ra một cuộc cách mạng trong việc giao tiếp giữa con người - đồ vật và giữa các đồ vật với nhau. IoT là một xu hướng mới, tuy nhiên cũng tạo ra nhiều thách thức mới về an ninh và sự riêng tư thông qua việc kết nối với các thiết bị và dịch vụ phổ biến trên Internet. Để có thể khai thác được những tiềm năng to lớn mà IoT mang lại, còn nhiều vấn đề cần phải giải quyết đặc biệt là vấn đề bảo mật cho các thiết bị và hệ thống IoT.

Theo một nghiên cứu gần đây của Dự án an toàn ứng dụng web mở (Open Web Application Security Project - OWASP) thì có tới gần 3/4 thiết bị IoT có nguy cơ bị tin tặc tấn công và xâm hại. Nghiên cứu này cũng chỉ ra một số nguy cơ bảo mật phổ biến nhất đối với các thiết bị IoT. Bất kỳ người dùng nào truy cập vào thiết bị qua kết nối di động đều có thể bị tấn công bởi lỗ hổng này do tính không đảm bảo an toàn của các dịch vụ mạng, đặc biệt là các dịch vụ vô tuyến. Kẻ tấn công sử dụng các dịch vụ mạng dễ bị tấn công để tấn công vào thiết bị.

Chính vì vậy việc nghiên cứu các giải pháp bảo mật cho các thiết bị IoT là vấn đề cần được quan tâm nghiên cứu. Các thiết bị IoT thường rất đa dạng về chủng loại, có thiết bị có hệ điều hành, nhưng cũng có rất nhiều thiết bị là các hệ thống nhúng không có hệ điều hành do đó việc phát triển một phần mềm bảo mật chung cho tất cả các thiết bị IoT là điều rất khó thực hiện. Hơn nữa với các thiết bị IoT có nguồn tài nguyên hạn chế việc thực hiện mã hóa và giải mã bằng phần mềm là không phù hợp do chiếm nhiều tài nguyên của thiết bị, tiêu tốn năng lượng và có thể không đáp ứng được yêu cầu thời gian thực, việc sử dụng các phần mềm bảo mật cũng ẩn chứa nhiều nguy cơ bị tin tặc tấn công. Do đó việc phát triển các giải pháp bảo mật phần cứng cho thiết bị IoT là một giải pháp phù hợp.

Bảo mật an toàn thông tin khi truyền qua mạng gồm các vấn đề như: Tính bảo mật, tính xác thực, tính toàn vẹn dữ liệu… Trong đó bảo mật dữ liệu là một trong các vấn đề của bảo mật và an toàn thông tin. Các thuật toán mã hóa bảo mật dữ liệu tiêu biểu như DES (Data Encryption Standard), Triplle DES… Trong đó thuật toán AES (Advanced Encryption Standard) được ra đời năm 2000 và được chuẩn hóa bởi Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ đã được sử dụng rộng rãi trên phạm vi toàn thế giới và được sử dụng trong nhiều chuẩn truyền thông của IEEE. AES có nhiều ưu điểm: tính bảo mật, tính hiệu quả khi thực hiện trên phần mềm và phần cứng, tốc độ, độ chính xác khi mã hóa, giải mã và tính khả thi. Ngày nay các vấn đề nghiên cứu triển khai AES bằng phần cứng theo các hướng tối ưu hóa công suất tiêu thụ, vấn đề về băng thông vẫn được quan tâm nghiên cứu.

Từ những phân tích, đánh giá trên, luận án đặt ra mục tiêu là nghiên cứu như sau:

- Nghiên cứu các vấn đề cơ bản của bảo mật bằng phần cứng cho các thiết bị Internet of things. Nghiên cứu các phương pháp triển khai thuật toán AES bằng phần cứng. Nghiên cứu các kỹ thuật cơ bản để thiết kế các vi mạch tiêu thụ năng lượng thấp. Đề xuất các giải pháp, kỹ thuật thực thi bộ mã hóa AES bằng phần cứng có hiệu năng cao.

- Nghiên cứu các phương pháp tối ưu công suất tiêu thụ cho phần cứng đa lõi AES. Đề xuất phương án giảm công suất tiêu thụ của phần cứng đa lõi AES bằng kỹ thuật clock gating sử dụng mạng SNN. Đề xuất thuật toán tạo bộ dữ liệu huấn luyện mô hình SNN. Đề xuất kiến trúc phần cứng của nền tảng AES đa lõi và phần cứng bộ điều khiển công suất thấp SNN.

Để đưa ra giải pháp đúng đắn thực hiện mục tiêu nghiên cứu đề

ra, luận án sử dụng các phương pháp nghiên cứu như sau:

- Tìm hiểu tổng quan lý thuyết về thuật toán mã hóa bảo mật AES, các phương pháp triển khai AES bằng phần cứng và các công trình liên quan. Tổng quan lý thuyết về các kỹ thuật công suất thấp trong thiết kế các vi mạch tích hợp. Đề xuất các thiết kế phần cứng AES hiệu năng cao, công suất tiêu thụ thấp.

- Sử dụng các công cụ thiết kế, mô phỏng, thực thi phần cứng để triển khai các kiến trúc mã hóa AES từ đó phân tích các yếu tố về hiệu năng và công suất tiêu thụ của hệ thống. Đề xuất giải pháp cải tiến băng thông và công suất tiêu thụ của hệ thống bằng việc sử dụng kiến trúc đa lõi AES và phương pháp ngắt xung đồng hồ (clock gating) cho các lõi AES bằng phần cứng Spiking Neural Network.

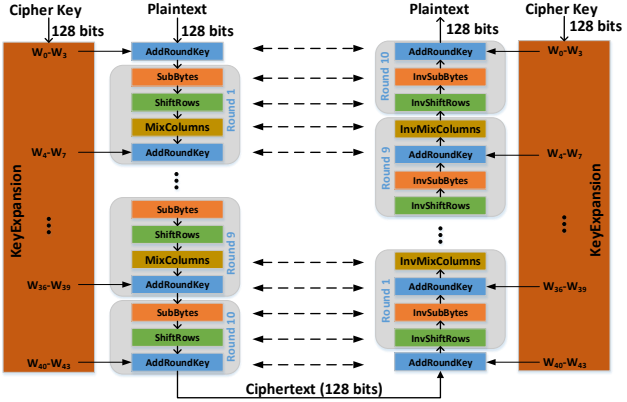
- Sử dụng các công cụ thiết kế, mô phỏng để triển khai các kiến trúc phần cứng AES. Phân tích hiệu năng và công suất tiêu thụ của hệ thống. Đề xuất giải pháp cải tiến băng thông và công suất tiêu thụ của hệ thống như sử dụng kiến trúc đa lõi AES và phương pháp ngắt xung đồng hồ (clock gating) cho các lõi AES bằng phần cứng Spiking Neural Network.

Chương 1. TỔNG QUAN

Trong chương này tác giả đã trình bày một số khái niệm về bảo mật và an toàn thông tin, lịch sử ra đời của thuật toán mã hóa bảo mật AES. Mô hình thuật toán và các phép biến đổi cũng được trình bày một cách chi tiết. Tác giả cũng tập trung phân tích các tham số đánh giá trong quá trình thực thi chuẩn mã AES bằng phần cứng như băng thông, độ trễ, công suất tiêu thụ, chi phí diện tích…Luận án cũng phân tích và đánh giá hiệu quả của các công trình thực thi AES bằng phần cứng, phân tích các ưu, nhược điểm để làm cơ sở định hướng nghiên cứu cho các phần tiếp theo. Các kiến trúc và các phương án thực thi

AES bằng phần cứng cũng được phân tích và tổng hợp. Trong đó triển khai kiến trúc AES thông lượng cao sử dụng các kiến trúc mã hóa song song, kỹ thuật đường ống bên trong, bên ngoài các tầng mã hóa, kiến trúc đa lõi. Các kỹ thuật để giảm công suất thiết kế như phân chia miền clock, giảm tần số xung nhịp, tối ưu hóa tài nguyên phần cứng, tối ưu hóa thuật toán, ngắt xung đồng hồ, power gating… cũng được đề cập để tìm ra một hướng tiếp cận phù hợp cho định hướng nghiên cứu của luận án.

1.1. Thuật toán AES



Hình 1.1. Mô hình thuật toán AES-128.

Hình 1.1 mô tả mô hình thuật toán AES-128. Mã hóa AES là bộ mã hóa theo khối 128-bit và là một thuật toán mã hóa khối đối xứng, nghĩa là thuật toán mã hóa và giải mã sử dụng chung một khóa, thuật toán giải mã là phép toán ngược của thuật toán mã hóa. AES dùng 4 phép biến đổi chính để mã hóa một khối dữ liệu là: *Add Round Key*, *Substitute Bytes*, *Shift Rows*, *Mix Columns* với các phép biến đổi ngược tương ứng là *Inverse Sub Bytes*, *Inverse Shift Rows*, *Inverse Mix Columns*. Riêng phép biến đổi *Add Round Key* đơn giản chỉ là phép

XOR nên phép biến đổi ngược cũng là *Add Round Key*. Mỗi phép biến đổi nhận tham số đầu vào có kích thước 128-bit và cho ra kết quả cũng có kích thước 128-bit. Trong AES, số vòng mã hóa phụ thuộc vào kích thước của khóa. Tương ứng bằng 10, 12 hoặc 14 cho các khóa 128-, 192- hoặc 256-bit.

Vận dụng các phép biến đổi ngược trên, thuật toán giải mã AES cũng gồm 10 vòng thực hiện theo chiều ngược lại. Kích thước khóa ban đầu là 128 bit (gồm 16 byte). AES dùng hàm *Expand Key* để mở rộng kích thước khóa thành 44 từ (word) 32 bit. 44 từ này được chia thành 11 cụm khóa con, mỗi khóa con 4 từ làm tham số cho 11 thao tác *Add Round Key*.

1.2. Các phương án thực thi AES bằng phần cứng

Kiến trúc thông lượng cao

- Sử dụng các kiến trúc mã hóa song song, các kỹ thuật đường ống
- Kiến trúc đa lõi

Công suất tiêu thụ thấp

- Kiến trúc: 8, 16, 32, 64 đường dữ liệu (datapath);
- Sử dụng kiến trúc vòng lặp
- Kỹ thuật thiết kế công suất thấp

Tối ưu tài nguyên phần cứng, công suất tiêu thụ

- Sử dụng các kiến trúc lặp 1 phần
- Sử dụng các cổng logic thay cho bảng tra cứu S-Box

Tái cấu hình: Tái cấu hình AES-128, 192, 256

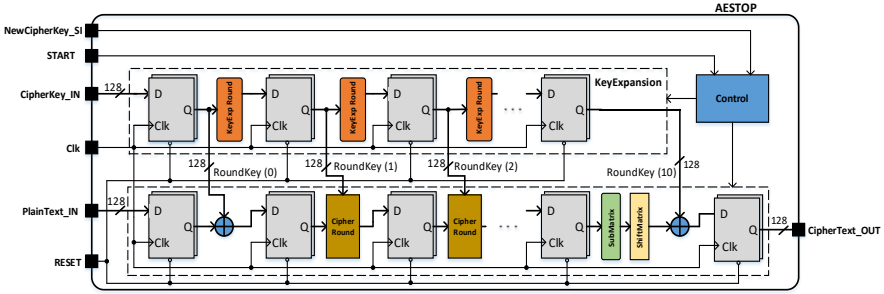
Chương 2. THIẾT KẾ PHẦN CỨNG ĐƠN LÕI AES

Nội dung chính trong Chương 2 là đề xuất kiến trúc AES đơn lõi, đi sâu vào chi tiết thiết kế kiến trúc phần cứng của lõi AES. Kiến trúc lõi đơn AES được thiết kế với ngôn ngữ mô tả phần cứng VHDL, mô

phỏng và kiểm chứng các chức năng trên ModelSim và tổng hợp phần cứng với Design Compiler của Synopsys với thư viện NAND GATE 45nm.

2.1. Đề xuất kiến trúc phần cứng đơn lõi AES

Kiến trúc phần cứng AES được đề xuất trong Hình 2.1. Để đạt được thông lượng cao, kỹ thuật đường ống bên ngoài các tầng mã hóa được thực hiện bằng cách chèn các thanh ghi vào giữa các tầng mã hóa.



Hình 2.1. Đề xuất kiến trúc phần cứng của AES.

Kiến trúc gồm 11 tầng mã hóa. Mỗi tầng mã hóa được thiết kế là các mạch lo-gic tổ hợp, trong đó tầng đầu tiên chỉ thực hiện phép XOR 128 bit dữ liệu với 128 bit khóa chính, 9 tầng tiếp theo giống hệt nhau (Cipher Round) gồm 4 phép biến đổi *SubMatrix*, *ShifMatrix*, *MixMatrix* và *AddRoundKey*, riêng tầng cuối cùng chỉ có 3 phép biến đổi là *SubMatrix, ShifMatrix và AddRoundKey*. Kiến trúc đường ống bên ngoài các tầng mã hóa đảm bảo khi dữ liệu điền đầy trong các tầng đường ống thì mỗi chu kỳ xung nhịp mã hóa được một khối dữ liệu 128 bit. Phần tiếp theo trình bày chi tiết kiến trúc phần cứng và các mô-đun của AES và kết quả mô phỏng trên công cụ ModelSim.

2.2. Kết quả tổng hợp phần cứng và thảo luận

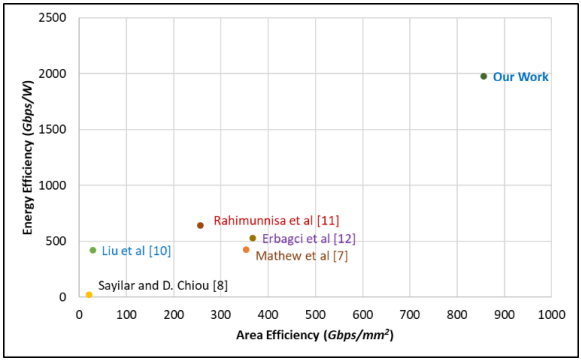
Chi tiết các kết quả thực thi phần cứng ở tần số hoạt động 870 *MHz*

được trình bày trong Bảng 2.1, thiết kế này đạt được thông lượng cao 111,3 *Gbps*, với mức tiêu thụ điện năng 56,3 *mW*. Chi phí diện tích là 0,13 mm^2 (164,5 *kGates*). Từ đây có thể tính được hiệu quả sử dụng phần cứng là 856 $Gbps/mm^2$.

Bảng 2.1. Kết quả thực thi phần cứng.

Công nghệ (*nm*)	**45**
Tần số xung nhịp (*MHz*)	870
Diện tích (mm^2)	0,13
Độ trễ (*ns*)	12,6
Diện tích (*kGate*)	164,5
Số xung nhịp cần thiết để mã hóa một khối dữ liệu	11
Băng thông (*Gbps*)	111,3
Công suất tiêu thụ (*mW*)	56,3
Hiệu quả sử dụng diện tích (*Mbps/kGate*)	676,6
Hiệu quả sử dụng diện tích ($Gbps/mm^2$)	856,1
Hiệu quả sử dụng năng lượng (*Gbps/W*)	1977

Kết quả triển khai kiến trúc phần cứng trên công nghệ ASIC 45*nm* được so sánh với các công trình liên quan trong Bảng 2.2. Thiết kế được đề xuất không chỉ có độ trễ thấp mà còn hiệu quả cao trong việc sử dụng tài nguyên phần cứng. Trên cùng công nghệ 45*nm*, thiết kế của luận án đạt được thông lượng gấp đôi so với thiết kế của Mathew và các cộng sự. Hiệu quả sử dụng diện tích cũng lớn hơn 2,4 lần, độ trễ thấp hơn 2 lần, mặt khác, công suất tiêu thụ cũng thấp hơn 2,2 lần. Trong kiến trúc AES của Sayilar và D. Chiou, mặc dù thông lượng cao hơn nhưng hiệu quả sử dụng tài nguyên phần cứng thấp hơn 400 lần và mức tiêu thụ công suất cao hơn109 lần so với thiết kế được đề xuất. So sánh về diện tích, kiến trúc AES được đề xuất nhỏ hơn 48 lần so với thiết kế của Sayilar và D. Chiou.



Hình 2.2. So sánh hiệu quả sử dụng diện tích và năng lượng.

Về độ trễ, kiến trúc được đề xuất có độ trễ thấp nhất so với các công trình trong Bảng 2.2. Do đó, kiến trúc này phù hợp với các ứng dụng thời gian thực. Hình 2.2 thể hiện hiệu quả sử dụng năng lượng và hiệu quả sử dụng diện tích của thiết kế được đề xuất cao hơn so với các công trình liên quan khác.

Bảng 2.2. So sánh kết quả thực thi phần cứng trên công nghệ ASIC.

Công trình	CLK (MHz)	Số xung nhịp mã hóa	Công nghệ (nm)	Diện tích (mm^2)	Diện tích (kGate)	Công suất (mW)	Thông lượng (Gbps)	Độ trễ (ns)	Hiệu quả sử dụng năng lượng (Gbps/W)	Hiệu quả sử dụng diện tích (Gbps/mm^2)
Mathew et al.	2100	55	45	0,15	-	125	53	26,2	424	353
Sayilar and D. Chiou	1000	20	45	6,32	-	6179	128	20	20,7	20,3
Ali et al.	1015	21	180	-	-	-	130	20,7	-	-
Liu et al.	255	-	90	0,104	-	7,1	2,97	-	418	28,6
Rahimunnisa et al.	200	55	130	0,1	-	40	25,6	275	640	256
Erbagci et al.	2200	44	65	0,75	-	523	275,2	20	526	367
Hodjat et al. ver.1	234	11	180	-	180	-	30	47	-	-
Hodjat et al. ver.2	547	41	180	-	275	-	70	74,9	-	-
Our Work	**870**	**11**	**45**	**0,13**	**164,5**	**56,3**	**111,3**	**12,6**	**1977**	**856,1**

2.3. Kết luận chương

Chương này đã đề xuất kiến trúc phần cứng AES đơn lõi phù hợp với các ứng dụng thông lượng cao và yêu cầu thời gian thực. Kiến trúc mã hóa song song và kỹ thuật đường ống được sử dụng để tăng tốc độ mã hóa và giảm độ trễ. Kết quả thực thi phần cứng trên công nghệ ASIC 45*nm* cho thấy thiết kế có thể hoạt động ở tần số tối đa 870 *MHz*

và đạt được thông lượng cao 111,3 *Gbps* và có độ trễ thấp (12,6 *ns*) trong khi có hiệu quả sử dụng diện tích (856 *Gbps/mm*^{2}) và hiệu quả sử dụng năng lượng (1977 *Gbps/W*) cao hơn một số công trình liên quan khác.

Mặc dù kiến trúc AES lõi đơn được đề xuất trong chương này đạt được thông lượng tương đối cao, hiệu quả sử dụng diện tích và năng lượng khá tốt. Tuy nhiên vẫn chưa đáp ứng được tốc độ của một số chuẩn truyền thông thế hệ mới (ví dụ *IEEE P802.3bs 2017* quy định tốc độ dữ liệu tối đa lên đến 400 *Gbps*). Hướng phát triển tiếp theo là đề xuất kiến trúc AES đa lõi có hiệu quả sử dụng năng lượng và hiệu quả sử dụng tài nguyên phần cứng tốt hơn với thông lượng lên đến vài trăm *Gbps*, đáp ứng các tiêu chuẩn truyền thông băng thông rộng hiện tại và tương lai.

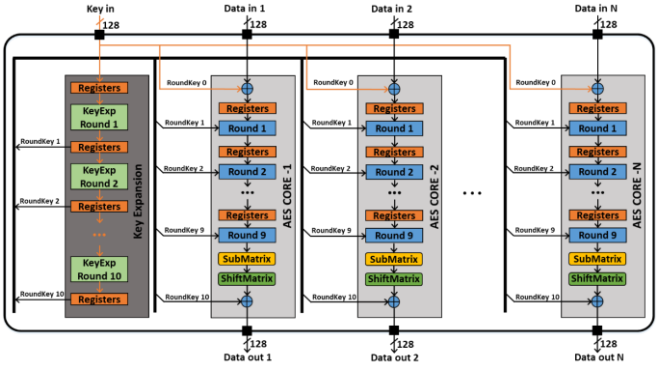
Chương 3. THIẾT KẾ PHẦN CỨNG ĐA LÕI MCRYPTCORES

Chương này đề xuất kiến trúc phần cứng mã hóa AES đa lõi (MCryptCores) để đạt được bộ mã hóa thông lượng cao, độ trễ nhỏ đáp ứng yêu cầu mã hóa bảo mật của các chuẩn truyền thông mới. Để giảm chi phí diện tích và công suất tiêu thụ, các lõi AES sẽ dùng chung khối Tạo khóa con *(KeyExpansion).* Kỹ thuật mã hóa song song và kỹ thuật đường ống *(pipeline)* áp dụng bên ngoài các vòng AES cũng được áp dụng cho kiến trúc xuất để tăng thông lượng và giảm độ trễ. Thiết kế của MCryptCores được mô hình hóa ở mức RTL *(Register-Transfer-Level)* sử dụng ngôn ngữ VHDL sau đó được tổng hợp trên công nghệ CMOS 45 *nm* sử dụng công cụ *Design Compiler* của *Synopsys*.

3.1. Thiết kế kiến trúc phần cứng đa lõi MCryptCores

Kiến trúc AES đa lõi MCryptCores được đề xuất được mô tả trong

Hình 3.1. Kiến trúc này bao gồm *N* lõi AES đơn, hoạt động song song để tăng tốc độ mã hóa. Vì vậy, mỗi chu kỳ xung nhịp mã hóa được $128 \times N$ bit dữ liệu đầu vào. Thông thường, mỗi lõi AES có một khối Tạo khóa con *(KeyExpansion)* được sử dụng để tạo các khóa con cho mỗi tầng mã hóa AES. Kiến trúc của các lõi đơn AES được kế thừa từ kiến trúc được đề xuất trong Chương 2. Tuy nhiên, để giảm diện tích và công suất tiêu thụ, chương này đề xuất kiến trúc McryptCores với các lõi AES dùng chung khối Tạo khóa con *(KeyExpansion).*

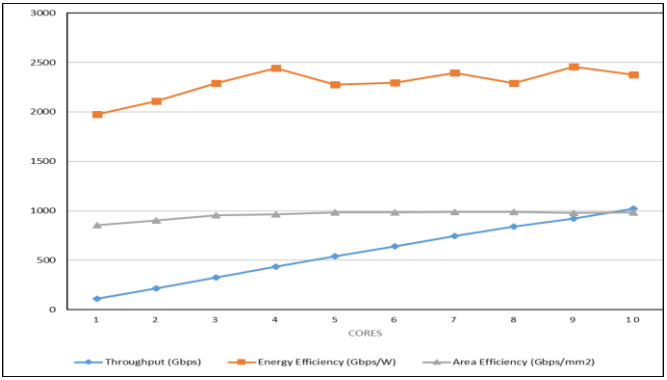


Hình 3.1. Kiến trúc phần cứng đa lõi MCryptCores.

3.2. Kết quả tổng hợp phần cứng và thảo luận

Công suất tiêu thụ và chi phí diện tích tỉ lệ thuận với số lõi AES trên chip. Tuy nhiên, vì kiến trúc đa lõi MCryptCores dùng chung Khối Tạo khóa con *(KeyExpansion),* nên tiết kiệm năng lượng hơn và có hiệu quả sử dụng diện tích cao hơn so với kiến trúc lõi đơn. Với một lõi trên chip, hiệu suất năng lượng là 1977 *Gbps/W* và hiệu quả diện tích là 956 *Gbps/mm*^{2}. Với 10 lõi trên chip, hiệu suất sử dụng năng lượng và diện tích của MCryptCores cao hơn hẳn, lần lượt 2377 *Gbps/W* và 983 *Gbps/mm*^{2}. Do đó, kiến trúc 10 lõi MCryptCores tiết kiệm năng lượng hơn 20% và hiệu quả sử dụng diện tích cao hơn 28%

so với kiến trúc lõi đơn. Mặt khác, so với các công trình liên quan, kiến trúc đa lõi được đề xuất tiết kiệm hơn về diện tích và công suất tiêu thụ.



Hình 3.2. So sánh thông lượng và hiệu năng của các kiến trúc đa lõi.

Với kiến trúc lõi đơn *(N=1)*, thông lượng là 111,3 *Gbps*, nhưng với kiến trúc 10 lõi AES *(N=10)*, thông lượng lên đến 1 *Tbps*. Như vậy các kiến trúc này có thể đáp ứng các yêu cầu về tốc độ mã hóa dữ liệu đối với các tiêu chuẩn truyền thông hiện tại và trong tương lai. Ví dụ kiến trúc 4 lõi *(N=4)* có thông lượng 433,7 *Gbps* đáp ứng yêu cầu truyền dữ liệu 400 *Gbps* của *P802.3bs 2017*.

Bảng 3.1. Kết quả thực thi kiến trúc MCryptCores trên công nghệ CMOS 45 nm.

Số lõi trên chip	**F_{max} *(MHz)***	**Diện tích *(mm^2)***	**Diện tích *(kGate)***	**Công suất *(mW)***	**Thông lượng *(Gbps)***	**Độ trễ *(ns)***	**Hiệu quả năng lượng *(Gbps/W)***	**Hiệu quả diện tích *(Gbps/mm^2)***	**Thông lượng/lõi *(Gbps/core)***
1 lõi (N=1)	870	0,13	164,5	56,3	111,3	12,6	1977	856	111,3
2 lõi (N=2)	847	0,24	303,6	102,7	216,8	13,0	2111	903	108,4
3 lõi (N=3)	847	0,34	431,9	142,0	325,2	13,0	2289	956	108,4
4 lõi (N=4)	847	0,45	561,1	177,6	433,7	13,0	2442	964	108,4
5 lõi (N=5)	847	0,55	690,9	238,1	542,1	13,0	2277	986	108,4
6 lõi (N=6)	833	0,65	815,2	278,7	639,7	13,2	2296	983	106,6
7 lõi (N=7)	833	0,75	945,7	311,9	746,4	13,2	2393	989	106,6
8 lõi (N=8)	820	0,85	1062,6	366,4	839,7	13,4	2292	990	105,0
9 lõi (N=9)	800	0,94	1178,5	374,8	921,6	13,8	2459	980	102,4
10 lõi (N=10)	800	1,04	1305,9	430,9	1024,0	13,8	2377	983	102,4

Bảng 3.1 so sánh thông lượng của kiến trúc AES 10 lõi MCryptCores với các công trình liên quan. Kiến trúc đề xuất có thông

lượng 1024 *Gbps* thấp hơn so với công trình [7] (sử dụng GPU Tesla V100), nhưng cao hơn các công trình sử dụng GPU, FPGA và ASIC khác.

Bảng **Error! No text of specified style in document.**.1. So sánh thông lượng của kiến trúc AES đa lõi với các công trình liên quan.

Công trình	**Nền tảng**	**Số lõi**	**Thông lượng *(Gbps)***
[1] 2019	65 *nm* CMOS	9 cores AES CCM	13,54
[2] 2015	multiple FPGAs	20 core AES GCM	883
[3] 2010	FPGA Xilinx Virtex-5	4 cores AES GCM	119,3
[4] 2012	Intel® Xeon® X7560 Processors	32 cores	6,6
[5] 2017	NVIDIA GeForce GTX 1080 GPU	8 cores AES-ECB	279,86
[6] 2017	NVIDIA Tesla P100-PCIe	AES-ECB	605,9
[7] 2019	Tesla V100 GPU	AES-ECB	1380
[7] 2019	Tesla V100 GPU	AES- CTR	1470
[8] 2014	Radeon HD 7970	AES-ECB	205
Our work	**45 *nm* CMOS**	**10 cores AES-ECB**	**1024**

3.3 Kết luận chương

Chương này đã đề xuất kiến trúc MCryptCores có thông lượng mã hóa cao. Kết quả thực thi phần cứng chứng minh rằng kiến trúc MCryptCores đạt được thông lượng lên tới 1 *Tbps* với 10 lõi AES trên chip.

Các lõi AES sử dụng chung khối Tạo khóa con *(KeyExpansion)*, do đó tiết kiệm diện tích và điện năng tiêu thụ hơn. Với 10 lõi AES, MCryptCores có hiệu quả sử dụng năng lượng lớn hơn 20% và hiệu quả sử dụng diện tích lớn hơn 28% so với kiến trúc lõi đơn. Kết quả tổng hợp phần cứng cũng được so sánh với các công trình khác sử dụng FPGA, ASIC, GPU... Với 10 lõi trên chip, kiến trúc MCryptCores đạt được thông lượng lớn, hiệu quả cao về diện tích và công suất tiêu thụ.

Kiến trúc MCryptCores có độ trễ thấp là 13,8 *ns* (với 10 lõi), vì vậy

nó phù hợp với các ứng dụng thời gian thực. Mặt khác, thông lượng cao trong thiết kế cũng đáp ứng các yêu cầu bảo mật dữ liệu trong các tiêu chuẩn truyền thông mới như *IEEE P802.3bm 2015*, với tốc độ 100 *Gbps* hoặc IEEE *P802.3bs 2017* có tốc độ truyền dữ liệu lên đến 400 *Gbps.*

Chương 4. THIẾT KẾ PHẦN CỨNG ĐA LÕI SPIKE-MCRYPTCORES

Trong chương này, luận án đề xuất kiến trúc Spike-MCryptCores, đây là kiến trúc AES đa lõi với bộ điều khiển công suất thấp lấy cảm hứng từ não bộ. Spike-MCryptCores sử dụng nhiều lõi AES để tăng tốc độ mã hóa. Để giảm công suất tiêu thụ, Spike-MCryptCores sử dụng chiến lược điều khiển ngắt xung đồng hồ để bật/tắt xung nhịp của mỗi lõi. Để điều khiển ngắt xung đồng hồ, Spike-MCryptCores sử dụng spiking neural networks (SNN), được lấy cảm hứng từ não bộ. SNN gần đây đã trở nên phổ biến do phần cứng tương đối đơn giản và tốn ít năng lượng. Mặc dù có rất nhiều mô hình và mạch nơ-ron, nhưng một mô hình cho phần cứng SNN với chi phí diện tích nhỏ và công suất thấp là mục tiêu của luận án. Mô hình Leaky-Integrated-and-Fire được lựa chọn làm mô hình nơ-ron do có phần cứng đơn giản trong khi vẫn duy trì các tính năng hợp lý về mặt sinh học của nó.

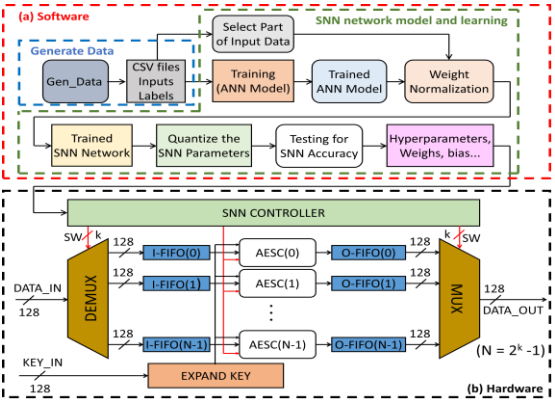
4.1. Kiến trúc Spike-McryptCores

Phần này trình bày kiến trúc tổng quan của hệ thống AES đa lõi công suất thấp với bộ điều khiển lấy cảm hứng từ não bộ (nền tảng Spike-MCryptCores). Sau đó mô tả chi tiết từng mô-đun của hệ thống này.

4.1.1. Kiến trúc tổng quan

Spike-MCryptCores bao gồm hai phần chính: Phần mềm và Phần

cứng như trong Hình 4.1. Hình 4.1(a) mô tả quá trình được thực hiện trên phần mềm, bao gồm các bước sau: (1) tạo dữ liệu training dựa trên các kịch bản khác nhau của dữ liệu đầu vào và lưu nó vào file .csv; (2) huấn luyện mô hình SNN bằng cách sử dụng các kịch bản dữ liệu được tạo trong giai đoạn 1, đánh giá mô hình và tính toán tham số (hyperparameter), trọng số (weight) và độ lệch (bias) cho phần cứng Spike-MCryptCores. Vai trò chính của Phần mềm là tạo ra một số kịch bản dữ liệu chung và sử dụng chúng để đào tạo SNN. Với mô hình SNN đã được đào tạo, hệ thống có thể dự đoán sự thay đổi của tốc độ dữ liệu và đưa ra cách điều chỉnh phù hợp. Nói cách khác, mô hình SNN sẽ dự đoán số lõi AES cần được bật để mã hóa dữ liệu đến. Nếu quá nhiều lõi được kích hoạt, có những lõi không sử dụng và nó sẽ tiêu tốn điện năng. Mặt khác, nếu quá ít lõi được kích hoạt, nó sẽ tạo ra sự cố nút chai (bottleneck) vì thông lượng Spike-MCryptCores nhỏ hơn tốc độ dữ liệu đến. Do đó, việc quyết định số lượng lõi được kích hoạt là rất quan trọng.



Hình 4.1. Nền tảng Spike-McryptCores.

Hình 4.1(b) minh họa sơ đồ khối phần cứng của nền tảng Spike-

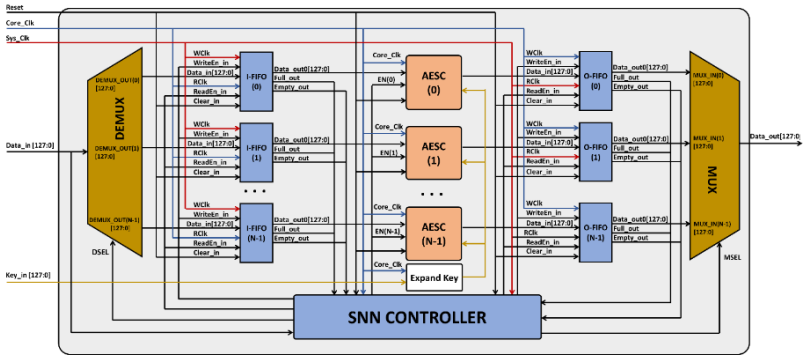
MCryptCores, bao gồm: (1) Phần cứng SNN: điều khiển quá trình tách/ghép dữ liệu (MUX, DEMUX) và bật/tắt các lõi AES; (2) DEMUX: Bộ phân kênh cho dữ liệu đầu vào; (3) MUX: Bộ ghép kênh cho dữ liệu đầu ra; (4) I-FIFO và O-FIFO lần lượt là Bộ đệm dữ liệu đầu vào và Bộ đệm dữ liệu đầu ra; (5) AESCs: gồm N lõi AES (AESC) hoạt động song song. Lưu ý rằng số lượng lõi AES (N) có thể được cấu hình ở giai đoạn thiết kế. Với cấu hình phần cứng do Phần mềm tạo ra, kiến trúc phần cứng của SNN có thể thực hiện dự đoán số lõi cần bật và đưa ra các tín hiệu để kích hoạt chúng. Vì toàn bộ dự đoán của SNN có thể được thực hiện song song trên phần cứng nên hệ thống không cần bất kỳ CPU chuyên dụng nào để thực hiện dự đoán. Hơn nữa, nhờ có chi phí phần cứng thấp và tiêu thụ ít năng lượng của SNN, bộ điều khiển có khả năng tiết kiệm năng lượng và không tạo ra chi phí đáng kể về điện năng và diện tích.

4.1.2. Thiết kế phần mềm cho nền tảng

Hình 4.1(a) là một sơ đồ thể hiện quá trình thực hiện bằng phần mềm, bao gồm hai giai đoạn chính là giai đoạn tạo dữ liệu huấn luyện và giai đoạn huấn luyện SNN. Mục đích của phần mềm là tạo ra các kịch bản có thể xảy ra của dữ liệu đầu vào. Các kịch bản này được sử dụng để huấn luyện mô hình SNN và sau đó tạo ra cấu hình phần cứng SNN.

4.1.3. Kiến trúc phần cứng của nền tảng

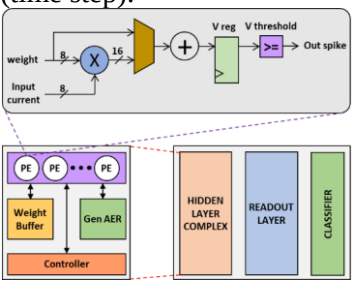
Hình 4.2 trình bày kiến trúc phần cứng chi tiết của nền tảng Spike-McryptCores. Bộ điều khiển SNN: Mô-đun này sẽ thực hiện dự đoán số lượng lõi được bật / tắt (bằng cách bật/tắt xung nhịp của mỗi lõi AES). Mô-đun cũng tạo ra các tín hiệu điều khiển cho DEMUX, I-FIFO, AESC, O-FIFO và MUX.



Hình 4.2. Kiến trúc phần cứng của Spike-MCryptCores.

4.1.4. Kiến trúc phần cứng cho SNN

Hình 4.3 trình bày kiến trúc phần cứng của một Phần tử xử lý (PE) và sơ đồ khối của SNN. Động lực của các nơ-ron LIF được xử lý bởi PE. Mỗi PE bao gồm một bộ Nhân và Cộng (Multiply-and-Accumulate MAC) đơn giản sẽ tích hợp các đầu vào cho nơ-ron trong mỗi bước thời gian (time step).



Hình 4.3. Kiến trúc phần cứng của SNN.

4.2. Kết quả và thảo luận

Trong phần này trình bày kết quả đánh giá của nền tảng Spike-MCryptCores. Bao gồm đánh giá kết quả huấn luyện với mô hình SNN. Kết quả triển khai phần cứng trên công nghệ CMOS 45*nm*. Công suất tiêu thụ của phần cứng Spike-MCryptCores cũng được so sánh

với công suất tiêu thụ của McryptCores (đã trình bày trong **Error! Reference source not found.**) để làm nổi bật vai trò của Bộ điều khiển SNN trong hệ thống.

4.2.1. Đánh giá kết quả phần cứng

Kiến trúc phần cứng của Spike-MCryptCores được thiết kế bằng ngôn ngữ mô tả phần cứng VHDL, được mô phỏng và kiểm chứng trên ModelSim. Các công cụ Synopsys Design Compile, Prime Time và Cadence Innovus được sử dụng để phân tích công suất tiêu thụ và thiết kế layout với thư viện CMOS NANGATE 45*nm*.

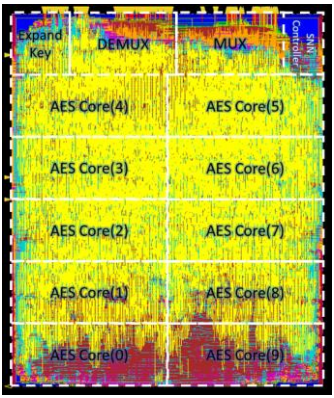
Bảng 4.1 trình bày chi phí phần cứng của các mô-đun trong Nền tảng Spike-MCryptCores. Với cấu hình N = 10 AESC, chi phí diện tích của nền tảng Spike-MCryptCores là 0,992 mm^2. Trong đó chi phí diện tích cho các AESC là 0,898 mm^2, chiếm 90,5%. Chi phí phần cứng cho các mô-đun DEMUX và MUX chiếm 7,2%. Trong khi bộ điều khiển SNN chỉ chiếm 2,3%. Có thể thấy rằng với chi phí diện tích rất nhỏ nhưng bộ điều khiển SNN có thể dự đoán số lõi và có thể bật/tắt xung nhịp của các AESC phù hợp với tốc độ dữ liệu đến để tiết kiệm công suất của hệ thống.

Bảng 4.1. Kết quả tổng hợp phần cứng của Spike-McryptCores.

Module	Absolute Total (mm^2)	Percent (%)
Spike-MCryptCores	0,992	100
AESCs	0,898	90,5
DEMUX + I-FIFO	0,036	3,6
MUX + O-FIFO	0,036	3,6
SNN CONTROLLER	0,021	2,3

Hình 4.4 là Layout và Floorplan của Spike-MCryptCores với kích thước 1200x1425 μm bao gồm các mô-đun chính sau: AESC (10

AESC) chiếm phần lớn diện tích chip (90,5%), DEMUX+I-FIFO và MUX+O-FIFO chiếm 7,2%, và phần còn lại là SSN Controller, chỉ chiếm một phần nhỏ (2,3%) trong chip.



Hình 4.4. Spike-MCryptCores Layout& Floorplan.

4.2.2. Đánh giá kết quả huấn luyện

Dữ liệu chuẩn bị cho quá trình training gồm 2500 mẫu data và label. Dữ liệu này được chia thành 2 phần. Phần 1: chọn ngẫu nhiên 400 mẫu để kiểm tra độ chính xác của mô hình huấn luyện. Phần 2: gồm 2100 mẫu còn lại dùng để huấn luyện.

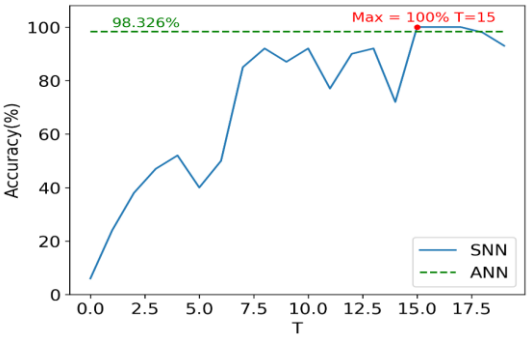
Bảng 4.2. Kết quả huấn luyện.

Network	8-3-11	8-5-11	8-10-11	8-15-11
ANN	95,82%	98,326%	95,81%	100%
SNN controller (32-bit)	89,29%	97,72%	95,67%	97,72%
SNN controller (8-bit)	89,29%	95,44%	96,58%	97,27%
Min Diff. (Prediction vs Label)	-1	-1	-1	-1
Max Diff. (Prediction vs Label)	+1	+1	+1	+1

Theo Bảng 4.2, độ chính xác của bộ điều khiển SNN (8-bit) với cấu hình 8-5-11 là 95,44%, do đó tỷ lệ lỗi của mô hình này là 4,56%.

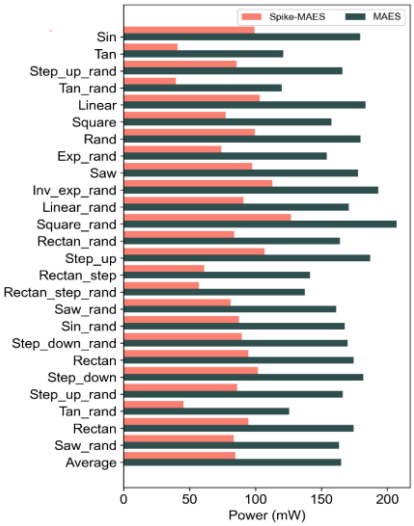
Tuy nhiên, trong những trường hợp dự đoán sai, sự chênh lệch giữa nhãn và dự đoán chỉ là 1 đơn vị. Nghĩa là, trong số 4,56% dự đoán không chính xác thì độ lệch cũng rất nhỏ (bộ điều khiển SNN dự đoán ít hơn một lõi hoặc nhiều hơn một lõi so với nhãn). Phần cứng Spike-AES có các mô-đun I-FIFO và O-FIFO ở đầu vào và đầu ra của AESC, vì vậy có thể giải quyết được các vấn đề tắc nghẽn do những sai lệch trong dự đoán của SNN. Có thể chọn mô hình 8-10-11 hoặc mô hình 8-15-11 để cải thiện độ chính xác lên 96,58% hoặc 97,27%; tuy nhiên, do độ phức tạp của phần cứng cũng tăng lên, luận án đã sử dụng 8-5-11 vì nó mang lại sự cân bằng tốt nhất giữa độ chính xác và chi phí diện tích.

Hình 4.5 là biểu đồ thể hiện kết quả huấn luyện với mô hình mạng nơ-ron 8-5-11. Mô hình có 8 đầu vào, 5 lớp ẩn và 11 đầu ra. Với mô hình ANN, độ chính xác đạt 98,326%, với mô hình SNN, độ chính xác tăng lên theo bước thời gian T và đạt đến độ bão hòa với độ chính xác lên đến 100% ở $T = 16$. Vì các spike được tạo ra trong quá trình huấn luyện SNN được tạo ngẫu nhiên theo quá trình Poisson, độ chính xác có thể thay đổi trong quá trình suy luận. Lưu ý rằng mức 100% có thể là do tác động của một nhiễu ngẫu nhiên vì nó giảm xuống thấp hơn sau 17 timestep.



Hình 4.5. Độ chính xác của ANN và SNN sử dụng mô hình 8-5-11.

4.2.3. Đánh giá hiệu năng của SNN Controller



Hình 4.6. Công suất tiêu thụ trung bình của Spike-MCryptCores và MCryptCores.

Hình 4.6 trình bày công suất trung bình của Spike-MCryptCores và MCryptCores cho tất cả 24 kịch bản. Công suất tiêu thụ trung bình của Spike-MCryptCores trong 24 kịch bản dữ liệu là 84,85 *mW*, trong khi với MCryptCores là 164,93 *mW*. Do đó, mức tiêu thụ năng lượng trung bình của Spike-MCryptCores trong 24 kịch bản dữ liệu bằng 51,4% so với của MCryptCores. Spike-MCryptCores đạt được khả năng điều khiển như mong đợi. Độ chính xác điều khiển lên đến 95,44%. Trong 4,56% số mẫu lỗi, sự sai lệch chỉ là ± 1 lõi.

Bảng 4.3 so sánh giữa giải pháp của luận án và các giải pháp tiêu thụ công suất thấp khác cho các Hệ thống trên chip. Spike-MCryptCores tiết kiệm được tối đa 67% trong khi các phương pháp sử sụng DVFS khác chỉ có thể tiết kiệm tối đa 51%. Công trình [11]

sử dụng kỹ thuật ngắt xung đồng hồ và thu được kết quả có thể so sánh được. Tóm lại, kiến trúc của lận án cho thấy một cách tiếp cận để giảm tiêu thụ điện năng với hiệu quả tương đương. Cũng xin lưu ý rằng so với các phương pháp khác, SNN cho phép linh hoạt hơn với khả năng đào tạo lại để thích ứng với các tình huống mới.

Bảng 4.3. So sánh mức độ tiết kiệm năng lượng giữa các mô hình thuật toán.

Work	**Technique**	**Algorithm**	**Platform**	**Energy Reduction (%)**
C. Ababei et al. [9]	Dynamic voltage and frequency scaling (DVFS)	Distributed (DVFS) algorithm	CMOS-65nm (TSMC)	- Maximum 50,0%
H. Zakaria et al. [10]	Dynamic voltage and frequency scaling (DVFS)	Programmable self-timed ring (PSTR)	CMOS-45nm (ST Microelectronics)	- Maximum 51,4%
W. Chouchene et al. [11]	Clock Gating	Open Compute Project (OCP)	FPGA - Xilinx Virtex5	- Maximum 63,0% - Minimum 37,0%
P. Pande et al. [12]	Dynamic voltage and frequency scaling (DVFS)	Producer - Consumer FIFO	CMOS-90nm (TSMC)	- Maximum 32,2%
H.-P. Phan et al. [13]	Dynamic voltage and frequency scaling (DVFS)	Fuzzy lo-gic algorithm	CMOS-65nm (TSMC)	- Maximum 43,0%
This work	Clock Gating	Spiking Neural Network (SNN) controller	CMOS-45nm	- Maximum 67,0% - Minimum 39,0% - Average 48,6%

4.3. Kết luận chương

Trong chương này, nghiên cứu sinh đã đề xuất hệ thống Spike-MCryptCores với bộ điều khiển nơ-ron công suất thấp. Spike-MCryptCores bao gồm phần mềm cho phép thiết kế, huấn luyện và kiểm tra bộ điều khiển SNN và phần cứng bao gồm nhiều lõi AES được điều khiển bởi phần cứng SNN. Phần mềm thiết kế để huấn luyện cho SNN đạt độ chính xác trên 95% chỉ với một lớp ẩn duy nhất gồm 5 nơ-ron và chỉ sai khác 1 lõi so với nhãn trong trường hợp có lỗi. Nền tảng được đề xuất chỉ chiếm 7,6% chi phí diện tích cho bộ ghép kênh,

bộ phân kênh và FIFO không đồng bộ. Hơn nữa, bộ điều khiển SNN chỉ chiếm 2,3% diện tích của hệ thống, con số này là không đáng kể. Với bộ điều khiển SNN, hệ thống có thể giảm tiêu thụ năng lượng từ 39% đến 67% so với McryptCores không có bộ điều khiển SNN. Với Spike-McryptCores, luận án đã giới thiệu một phương pháp mới để thiết kế và điều khiển các hệ thống đa lõi với chi phí cực nhỏ, độ chính xác cao và tiết kiệm năng lượng hiệu quả.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Mật mã đóng một vai trò quan trọng trong việc bảo mật dữ liệu chống lại các cuộc tấn công và giảm nguy cơ bị đánh cắp thông tin. AES là một trong những thuật toán mã hóa khóa đối xứng phổ biến nhất. AES đã nhận được sự quan tâm đáng kể của các nhà nghiên cứu trong những năm gần đây do có nhiều ứng dụng trong truyền thông, quân sự, ngân hàng điện tử, v.v. Triển khai AES có thể được phân loại rộng rãi thành triển khai phần mềm và phần cứng. So với việc triển khai phần mềm, việc triển khai AES bằng phần cứng đã được chứng minh là cung cấp khả năng bảo mật vật lý tốt hơn, tốc độ cao hơn và tiêu thụ năng lượng thấp hơn.

Từ những nghiên cứu, tìm hiểu về các phương pháp thiết kế vi mạch hiệu năng cao, luận án đã đề xuất kiến trúc phần cứng đơn lõi cho thuật toán mã hóa AES và đạt được băng thông 119,3 Gbps. Tuy nhiên để đáp ứng nhu cầu truyền thông băng rộng theo các chuẩn truyền thông mới như *IEEE P802.3bs 2017* với tốc độ dữ liệu lên tới 400 Gbps, luận án đã xây dựng kiến trúc phần cứng đa lõi AES. Kiến trúc phần cứng đa lõi gồm 10 lõi AES hoạt động song song, tuy nhiên do được thiết kế sử dụng chung khối Key Expansion nên tăng được hiệu quả sử dụng diện tích và hiệu quả sử dụng năng lượng. Mặc dù kiến trúc đa lõi đã đáp ứng được nhu cầu về băng thông ngày càng cao

của các chuẩn truyền thông hiện đại, tuy nhiên thiết kế AES đa lõi vẫn chưa được tối ưu hóa về năng lượng tiêu thụ. Do tốc độ dữ liệu đầu vào tại các thời điểm khác nhau có thể khác nhau, vì vậy tại nhiều thời điểm sẽ có các lõi không sử dụng đến nhưng vẫn tiêu thụ năng lượng vô ích. Để khắc phục nhược điểm đó, luận án đã đề xuất bộ điều khiển SNN- một mô hình tính toán mạng nơ-ron thế hệ thứ 3 và được mô tả là giống với hoạt động của não bộ để bật/tắt các xung nhịp tới từng lõi AES phù hợp với tốc độ dữ liệu đầu vào. Kết quả là bộ điều khiển SNN có thể tiết kiệm công suất tiêu thụ cho hệ thống từ 39% đến 67%. Các đóng góp chính của luận án là:

- Đề xuất và thực thi kiến trúc phần cứng AES đơn lõi cho các ứng dụng thông lượng cao và thời gian thực. Kiến trúc song song và kỹ thuật đường ống được sử dụng để tăng tốc độ mã hóa và giảm độ trễ. Kết quả triển khai phần cứng trên công nghệ ASIC 45nm cho thấy thiết kế đạt được thông lượng cao 111,3 Gbps và có độ trễ thấp (12,6 ns). Các kết quả này đã được công bố tại Hội nghị ISCIT 2019.

- Đề xuất và thực thi kiến trúc phần cứng AES đa lõi song song có thông lượng mã hóa cao. Để giảm thiểu chi phí về diện tích và công suất tiêu thụ, khối KeyExpansion được chia sẻ giữa các lõi AES. Kết quả thực thi phần cứng chứng minh rằng kiến trúc đạt được thông lượng lên tới 1 Tbps với 10 lõi AES trên chip. Các kết quả đã được công bố tại Hội nghị APCCAS 2020 và trên tạp chí JCSCE.

- Đề xuất và thực thi kiến trúc phần cứng Spike-MCryptCores với bộ điều khiển nơ-ron công suất thấp dùng để điều khiển bật/tắt xung nhịp của các lõi AES (clock gating). Spike-MCryptCores bao gồm phần mềm để thiết kế, huấn luyện và kiểm tra bộ điều khiển SNN và phần cứng bao gồm nhiều lõi AES và bộ điều khiển chặn xung nhịp SNN. Bộ điều khiển SNN có thể giúp hệ thống giảm tiêu thụ năng

lượng từ 39% đến 67%. Với Spike-McryptCores, luận án đã giới thiệu một phương pháp mới để thiết kế và điều khiển các hệ thống đa lõi với chi phí nhỏ, độ chính xác cao và tiết kiệm năng lượng. Các kết quả này được trình bày trên tạp chí IEEE Access (in review).

Hướng phát triển:

Mặc dù SNN điều khiển clock-gating có thể giảm tới 67% công suất tiêu thụ, nhưng cũng có các kỹ thuật tiêu thụ công suất thấp khác như power gating hoặc phân chia tỷ lệ tần số điện áp động (DVFS). Có thể dùng SNN điều khiển Power gating để ngắt nguồn của các mô-đun hoặc dùng SNN để điều khiển DVFS để tối ưu hóa năng lượng tiêu thụ của hệ thống. Cả hai cách tiếp cận đều được coi là các công việc trong tương lai và SNN có thể là giải pháp phù hợp.

Trong luận án này, sử dụng phương pháp huấn luyện ngoại tuyến cho SNN do đó vẫn không thể thực hiện điều chỉnh trong quá trình hoạt động (tức là nhận ra các trường hợp kém hiệu quả và điều chỉnh SNN). Nếu các nhà thiết kế muốn huấn luyện trực tuyến, có hai giải pháp: (1) sử dụng CPU chuyên dụng để thu thập dữ liệu và đào tạo SNN, và (2) sử dụng phương pháp học trực tuyến đàn hồi xung thời gian (STDP). Cách tiếp cận thứ hai là một trong những công việc trong tương lai của nhóm nghiên cứu nhằm cung cấp khả năng điều chỉnh hệ thống trực tuyến.