Xây dựng bộ thư viện người dùng cho card tăng tốc mã hóa qua giao tiếp PCIe

Vũ Tá Cường, Phan Văn Kỷ

***Tóm tắt* — Thiết bị mã hóa tốc độ cao sử dụng giao tiếp PCIe được phát triển rất đa dạng và được thương mại hóa từ lâu với mục đích tăng tốc độ xử lý mã hóa/giải mã và che giấu thuật toán, tham số mật mã. Trong bài báo [1], nhóm tác giả đã giới thiệu giải pháp thiết kế card tăng tốc mã hóa PCIe sử dụng công nghệ FPGA. Trong bài báo này, nhóm tác giả sẽ giới thiệu giải pháp xây dựng bộ thư viện người dùng cho card tăng tốc mã hóa PCIe. Bộ thư viện này sẽ có các hàm chức năng giúp các ứng dụng, phần mềm trên máy tính sử dụng card tăng tốc PCIe một cách linh hoạt, thuận tiện.**

***Abstract*— High-speed encryption device using PCIe interface has been developed and commercialized for a long time with the purpose of speeding up encryption/decryption processing and hiding algorithms and cryptographic parameters. In the article [1], the authors introduced a solution to design a PCIe cryptographic accelerator using FPGA technology. In this paper, the authors introduce a solution to build a user library for PCIe cryptographic accelerator. This library set will have functions to help applications and software on computers use PCIe acceleration cards flexibly and conveniently.**

Từ khóa— PCIe; FPGA; tăng tốc phần cứng.

Keyword— PCIe; FPGA; hardware acceleration.

I. GIỚI THIỆU

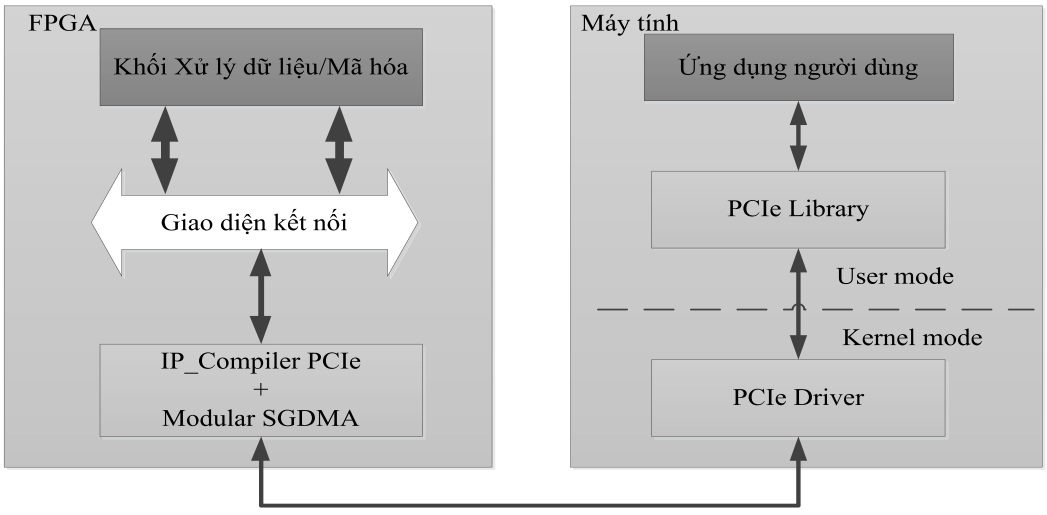
Hiện nay, phần lớn các hệ thống sử dụng mật mã đang được triển khai dân sự chủ yếu sử dụng các sản phẩm của nước ngoài. Do đặc điểm như vậy nên các nghiên cứu trực tiếp về mật mã và giao thức mật mã đa phần là về lý thuyết và các thuật toán phổ thông. Phần lớn các nghiên cứu xây dựng card tăng tốc xử lý thường chủ yếu tập trung vào nghiên cứu ứng dụng giao tiếp PCIe, để tăng tốc trao đổi giữa máy tính và các card phần cứng tăng tốc độ xử lý dữ liệu (xử lý hình ảnh, thuật toán trí tuệ nhân tạo). Trong lĩnh vực mật mã, việc tăng tốc mã hóa sử dụng các phần cứng chuyên dụng cũng được các nhà khoa học quan tâm và các hãng công nghệ lớn khác cũng cho ra đời các sản phẩm tăng tốc phần cứng chuyên dụng.

Một trong những ưu thế vượt trội của công nghệ FPGA là có khả năng xử lý dữ liệu song song, cho phép triển khai các thuật toán mật mã một cách nhanh chóng và hiệu quả. Với những ưu thế vượt trội đó, kết hợp với các ngoại vi tốc độ cao để thực hiện giao tiếp với các hệ thống bên ngoài như PCIe, USB 3.0 hay Ethernet cáp quang là tiền đề để phát triển các thiết bị mã hóa tốc độ cao.

Trên thực tế, đã có rất nhiều công ty lớn trên thế giới về lĩnh vực mật mã đã nghiên cứu và ứng dụng tính ưu việt của giao thức PCIe vào thiết bị của mình và được ứng dụng rất rộng rãi trong các lĩnh vực thương mại và dân sự. Các thiết bị này được tích hợp các thuật toán mật mã như AES hay DES; hỗ trợ tính toán ECC, RSA,... Mặc dù được thương mại hóa, có tốc độ xử lý và tính toán nhanh, tuy nhiên theo nghiên cứu của nhóm tác giả, các sản phẩm tăng tốc phần cứng chuyên dụng của các hãng công nghệ lớn [2-4] thường là các sản phẩm được đóng gói dưới dạng hộp đen, nhà sản xuất chỉ cung cấp các thông số kỹ thuật liên quan đến sản phẩm. Việc nghiên cứu tùy biến, tiến tới làm chủ thiết kế chế tạo các sản phẩm này là điều rất khó khăn.

Vì vậy, việc nghiên cứu thiết kế chế tạo card tăng tốc mã hóa sử dụng công nghệ FPGA là cần thiết. Với kinh nghiệm trong nghiên cứu, thiết kế và chế tạo các thiết bị mật mã sử dụng công nghệ FPGA, việc thiết kế card mã hóa tốc

Bài báo được nhận ngày 10/3/2022. Bài báo được nhận xét bởi phản biện thứ nhất ngày 17/3/2022 và được chấp nhận đăng ngày 30/3/2022. Bài báo được nhận xét bởi phản biện thứ hai ngày 12/4/2022 và được chấp nhận đăng ngày 28/4/2022.



Hình 1. Mô hình hoạt động chung của card tăng tốc mã hóa qua giao tiếp PCIe

độ cao sử dụng công nghệ FPGA qua giao tiếp PCIe với giải pháp đọc ghi trực tiếp, bóc tách, mã hóa/giải mã bằng ngôn ngữ mô tả phần cứng HDL trên FPGA là khả thi và có thể ứng dụng được trong thực tế.

Trong bài báo [1], nhóm tác giả đã trình bày giải pháp xử lý gói tin theo giao thức PCIe sử dụng công nghệ FPGA. Từ đó đã thực hiện can thiệp mật mã, làm cơ sở để xây dựng card tăng tốc mã hóa qua giao tiếp PCIe sử dụng công nghệ FPGA. Để giúp các ứng dụng và phần mềm máy tính có thể dễ dàng giao tiếp với card tăng tốc mã hóa qua giao tiếp PCIe, nhóm tác giả đã hướng tới xây dựng bộ thư viện người dùng với các hàm chức năng phục vụ quá trình mã hóa, giải mã dữ liệu.

Phần còn lại của bài báo được tổ chức như sau: Phần II trình bày mô hình làm việc của card tăng tốc mã hóa qua giao tiếp PCIe; Phần III trình bày giải pháp thiết kế, xây dựng bộ thư viện người dùng cho card tăng tốc mã hóa qua giao tiếp PCIe; Phần IV trình bày giải pháp và kết quả thử nghiệm sử dụng card tăng tốc mã hóa qua giao tiếp PCIe và Phần V là kết luận.

Ký hiệu: Trong toàn bộ bài báo các ký hiệu ECB, OFB, CTR, XTS là các chế độ mã hóa được sử dụng cho thuật toán mã khối; Một word trong tổ chức gói tin được thiết kế là 4 Byte; IV là vector khởi tạo được sử dụng cho các chế độ CTR và OFB.

II. Giới thiệu mô hình làm việc của card tăng tốc mã hóa qua giao tiếp PCIe

Mô hình hoạt động chung của card tăng tốc mã hóa qua giao tiếp PCIe được thể hiện như trên Hình 1. Mô hình này được chia làm hai phần chính, trong đó:

- Máy tính thực hiện giao tiếp với các ứng dụng người dùng thông qua bộ thư viện người dùng và driver PCIe. Driver của thiết bị được cài vào hệ điều hành máy tính và sẽ được tải khi hệ điều hành khởi động. Khi đó hệ điều hành sẽ nhận biết thiết bị PCIe như một ngoại vi đã được đăng ký, cho phép trao đổi dữ liệu giữa máy tính và thiết bị. Để ứng dụng người dùng có thể làm việc với thiết bị PCIe, cần có thư viện người dùng. Thư viện này cung cấp các hàm API để ứng dụng người dùng giao tiếp với thiết bị PCIe.

- Bo mạch FPGA thực hiện giao tiếp với máy tính qua PCIe bằng cách sử dụng IPCore PCIe Compiler và khối SGDMA. Dữ liệu từ IPCore này sẽ được chuyển tiếp tới khối xử lý dữ liệu được thiết kế trên FPGA bằng ngôn ngữ mô tả phần cứng HDL, thông qua các giao diện bus tương ứng với các bus dữ liệu vào/ra của IPCore PCIe.

III. Giải pháp thiết kế, xây dựng bộ thư viện người dùng cho card tăng tốc mã hóa qua giao tiếp PCIe

Nhóm tác giả hướng tới thiết kế card tăng tốc mã hóa tốc độ cao qua giao tiếp PCIe có thể tích

hợp cho nhiều ứng dụng người dùng, do đó sẽ đi sâu vào việc thiết kế bộ thư viện người dùng dành riêng cho card tăng tốc mã hóa qua giao tiếp PCIe.

Bộ thư viện người dùng sẽ giao tiếp với thiết bị PCIe thông qua driver của thiết bị. Quy trình thiết kế bộ thư viện người dùng được thực hiện dựa trên các tài liệu do Intel và Xilinx cung cấp [5-8]. Khi ứng dụng người dùng gửi yêu cầu xuống bộ thư viện API, một thread sẽ được gửi cho kernel driver và dữ liệu sẽ được trao đổi giữa thiết bị và ứng dụng người dùng thông qua các bộ đệm được phân bổ trước đó. Việc trao đổi dữ liệu sẽ được thực hiện thông qua hàm `IOCTL` trên Linux hoặc `DeviceIoControl` trên Windows.

Bộ thư viện này cho phép các ứng dụng người dùng tích hợp card tăng tốc mã hóa qua giao tiếp PCIe vào ứng dụng của mình nhằm tăng tốc độ mã hóa. Giải pháp thiết kế bộ thư viện trên máy tính mà chúng tôi xác định là yêu cầu các ứng dụng người dùng muốn tích hợp card mã hóa để tăng tốc độ xử lý, sẽ phải thực hiện gửi các tham số mật mã như khóa, IV, chế độ mã hóa,… xuống cho thiết bị bằng cách gửi kèm theo dữ liệu cần được mã hóa và giải mã.

Bộ thư viện người dùng được thiết kế gồm 02 nhóm hàm chính:

- Các hàm API cơ bản:

+ Hàm `pcie_open()`: Thực hiện mở thiết bị.

+ Hàm `pcie_send()`: Thực hiện truyền dữ liệu xuống thiết bị.

+ Hàm `pcie_recv()`: Thực hiện nhận dữ liệu từ thiết bị.

+ Hàm `pcie_close()`: Thực hiện đóng thiết bị.

- Các hàm API chức năng:

+ Các hàm xác thực thiết bị.

+ Các hàm truyền tham số mật mã xuống thiết bị.

+ Các hàm phục vụ mã hóa/giải mã.

A. Giải pháp thiết kế các hàm API cơ bản

Các hàm API cơ bản có các hàm chính là: hàm đóng và mở thiết bị; hàm truyền và nhận dữ liệu. Các hàm API chức năng khác sẽ được thiết kế dựa trên cơ sở hai API này. Việc thiết kế như vậy nhằm đơn giản hóa và tiện lợi nhất cho người sử dụng. Hai hàm API truyền và nhận dữ liệu lấy mảng byte làm tham số, các mảng byte chứa dữ liệu cần gửi hoặc dữ liệu nhận được.

1. Hàm mở thiết bị

Hàm này giúp người dùng truy cập đến thiết bị PCIe, khởi tạo bộ nhớ trước khi làm việc với thiết bị. Nguyên mẫu hàm mở thiết bị được xây dựng như sau:

```
pcie_t * PCIE_open(int id)
```

- Với tham số:

`id:` ID của thiết bị PCIe.

Giá trị trả về: Cấu trúc `pcie_t` lưu các tham số thiết bị PCIe hoặc `NULL` nếu không mở được thiết bị.

2. Hàm đóng thiết bị

Hàm đóng thiết bị sẽ được gọi khi thực hiện xong các chức năng của thiết bị như: mã hóa, giải mã,... Hàm này giúp giải phóng bộ nhớ, đóng thiết bị. Nguyên mẫu hàm đóng thiết bị được xây dựng như sau:

Nguyên mẫu:

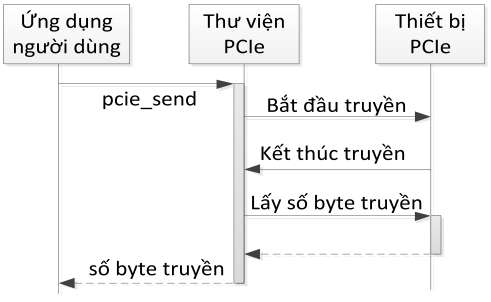
```
void  PCIE_close(pcie_t *pcie)
```

- Với tham số:

`Pcie:` Cấu trúc lưu tham số thiết bị PCIe.

3. Hàm truyền dữ liệu

Đối với hàm truyền dữ liệu `pcie_send()` sẽ được thiết kế theo sơ đồ như Hình 2.



Hình 2. Lưu đồ hoạt động của hàm truyền dữ liệu

Hoạt động của hàm này được mô tả như sau: khi ứng dụng người dùng gọi đến hàm truyền dữ liệu `pcie_send()`, thư viện PCIe sẽ gửi lệnh bắt đầu truyền xuống thiết bị PCIe. Thiết bị PCIe sau khi nhận được lệnh này sẽ bắt đầu nhận dữ liệu từ trên máy tính. Khi nhận hết dữ liệu thiết bị sẽ gửi chỉ thị báo cho thư viện biết quá trình truyền đã kết thúc. Tiếp đó, thư viện PCIe sẽ gửi lệnh yêu cầu lấy số byte đã truyền xuống thiết bị, thiết bị nhận được lệnh này sẽ gửi trả lại cho thư viện. Thư viện nhận được sẽ gửi trả lại cho ứng dụng người dùng số byte truyền.

Nguyên mẫu hàm truyền dữ liệu được tổ chức như sau:

```
int pcie_send(pcie_t * pcie, void * data, int len, long timeout)
```

- Với tham số:

`pcie:` Cấu trúc lưu tham số thiết bị PCIe.

`data:` Dữ liệu cần ghi xuống thiết bị PCIe.

`len:` Độ dài của dữ liệu ghi xuống (4-byte word).

`timeout:` Thời gian chờ trong khi truyền, đặt bằng 0 thì phần mềm sẽ chờ vô thời hạn.

Giá trị trả về: Số từ ghi được.

4. Hàm nhận dữ liệu

Sơ đồ hoạt động của hàm nhận dữ liệu được thực hiện như trong Hình 3.

Hoạt động của hàm này được mô tả như sau: khi ứng dụng người dùng gọi đến hàm truyền dữ liệu `pcie_recv()`, thư viện PCIe sẽ chờ dữ liệu gửi lên từ thiết bị PCIe. Khi thiết bị bắt đầu truyền dữ liệu lên máy tính, thư viện sẽ nhận và gửi cho ứng dụng người dùng. Khi nhận hết dữ liệu thiết bị sẽ gửi chỉ thị báo cho thư viện biết quá trình truyền đã kết thúc. Tiếp đó, thư viện PCIe sẽ gửi lệnh yêu cầu lấy số byte đã truyền xuống thiết bị, thiết bị nhận được lệnh này sẽ gửi trả lại cho thư viện. Thư viện nhận được sẽ gửi trả lại cho ứng dụng người dùng số byte truyền.

Nguyên mẫu hàm nhận dữ liệu được tổ chức như sau:

```
int pcie_recv(pcie_t * pcie, void * data, int len , long timeout)
```

- Với tham số:

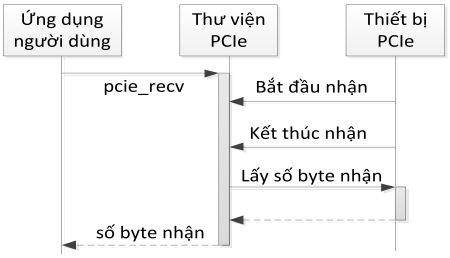
`pcie:` Cấu trúc lưu tham số thiết bị PCIe.

`data:` Dữ liệu đọc lên từ thiết bị PCIe.

`len:` Độ dài của dự liệu đọc lên (4-byte word).

`timeout:` Thời gian chờ trong khi truyền, đặt bằng 0 thì phần mềm sẽ chờ vô thời hạn.

Giá trị trả về: Số từ nhận được.



Hình 3. Lưu đồ hoạt động của hàm nhận dữ liệu

B. Giải pháp thiết kế các hàm API chức năng

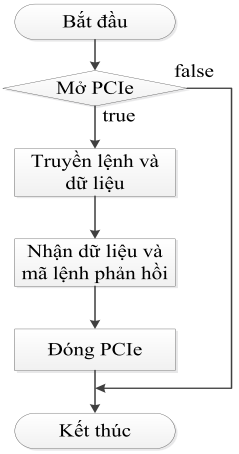
Giải pháp thiết kế chung của các hàm API chức năng được thể hiện như trong Hình 4. Cụ thể, tất cả các hàm API chức năng đều hoạt động theo các bước sau:

- Bước 1: Mở thiết bị PCIe. Nếu thành công chuyển sang Bước 2, ngược lại kết thúc.

- Bước 2: Truyền lệnh và dữ liệu xuống thiết bị PCIe.

- Bước 3: Nhận dữ liệu và mã lệnh phản hồi từ thiết bị PCIe.

- Bước 4: Đóng thiết bị PCIe.



Hình 4. Mô hình hoạt động chung của các hàm API chức năng

Tất cả các hàm API chức năng đều được tổ chức dữ liệu truyền xuống có header có độ dài 16 bytes và mã lệnh phản hồi có độ dài 4 bytes. Cụ thể tổ chức header và mã lệnh phản hồi được tổ chức như sau:

1. Các hàm xác thực thiết bị

Tổ chức header:

Thứ tự bit	Giá trị	Chức năng
7:0	0x02	Xác thực, hủy xác thực cho admin và người dùng
15:8	0x00	Xác thực thiết bị bởi mã pin admin
	0x01	Xác thực thiết bị bằng mã pin người dùng
	0x02	Hủy xác thực (khóa thiết bị) bằng mã pin admin
	0x03	Hủy xác thực (khóa thiết bị) bằng mã pin người dùng
	0x04	Thay mã pin mới cho admin
	0x05	Thay mã pin mới cho người dùng
	0x06	Xác thực để ghi và thay khóa trên SRAM
	0x07	Xác thực để ghi và thay Sbox
79:16		Báo hiệu độ dài của mã Pin (tính theo đơn vị Byte)

Tổ chức mã lệnh phản hồi:

Thứ tự bit	Giá trị	Ý nghĩa
7:0	0x01	Lỗi xác thực
	0x02	Xác thực thiết bị thành công
	0x03	Hủy xác thực (khóa thiết bị) thành công
	0x04	Hủy xác thực (khóa thiết bị) lỗi
15:8	Số lần lỗi	Số lần lỗi xác thực (tối đa 10 lần)

2. Các hàm truyền tham số mật mã

Tổ chức header:

Thứ tự bit	Giá trị	Chức năng
7:0	0x01	Báo hiệu truyền khóa
	0x0e	Báo hiệu truyền Sbox
15:8		Địa chỉ ghi khóa
23:16	0x01	Báo hiệu khóa lưu trực tiếp vào RAM trên FPGA
	0x02	Báo hiệu khóa lưu vào SRAM

Tổ chức mã lệnh phản hồi:

Thứ tự bit	Giá trị	Ý nghĩa
7:0	0x01	Lỗi
	0x02	Truyền thành công

3. Các hàm phục vụ mã hóa/giải mã

Tổ chức header:

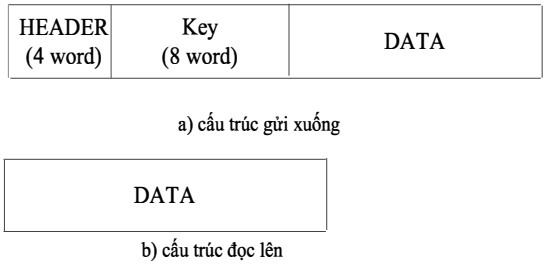
Thứ tự bit	Giá trị	Chức năng
7:0	0x03	Mã hóa theo chế độ ECB sử dụng khóa ở SRAM
	0x04	Giải mã theo chế độ ECB sử dụng khóa ở SRAM
	0x05	Mã hóa theo chế độ CTR sử dụng khóa ở SRAM
	0x07	Mã hóa theo chế độ OFB sử dụng khóa ở SRAM
	0x08	Giải mã theo chế độ ECB sử dụng khóa từ ứng dụng
	0x09	Mã hóa theo chế độ ECB sử dụng khóa từ ứng dụng
	0x0b	Mã hóa theo chế độ CTR sử dụng khóa từ ứng dụng
	0x0d	Mã hóa theo chế độ OFB sử dụng khóa từ ứng dụng
	0x0e	Giải mã theo chế độ XTS
	0x0f	Mã hóa theo chế độ XTS
15:8		Địa chỉ ghi khóa sử dụng để mã hóa/giải mã theo các chế độ
23:16	0x00	Mã hóa với IV từ thiết bị (áp dụng cho CTR và OFB)
	0x01	Mã hóa với IV từ ứng dụng (áp dụng cho CTR và OFB)

Tổ chức mã lệnh phản hồi:

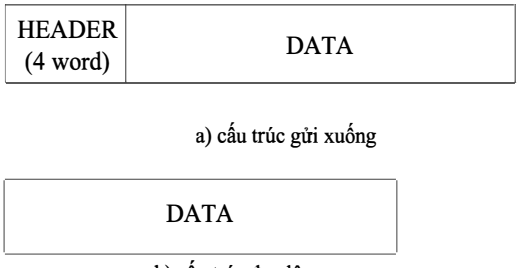
- Đối với mã hóa/giải mã theo chế độ CTR và OFB (các chế độ sử dụng IV), khi thực hiện mã hóa, sẽ sử dụng IV được cấp bởi thiết bị, quá trình đọc lên sẽ thực hiện đọc dữ liệu IV lên cùng với dữ liệu đã được mã hóa.

- Đối với quá trình giải mã, dữ liệu đọc lên sẽ trừ đi dữ liệu IV.

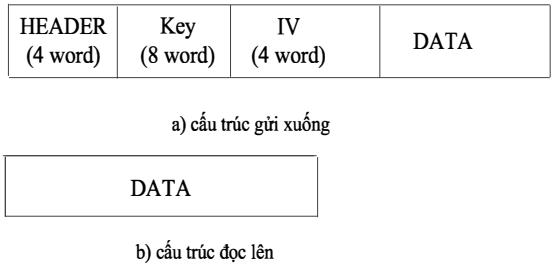
Cụ thể cấu trúc gói tin sử dụng trong các hàm mã hóa/giải mã được thể hiện như trong Hình 5 đến Hình 11.



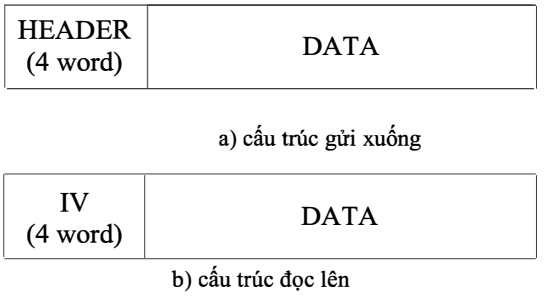
Hình 5. Cấu trúc gói tin mã hóa/giải mã theo chế độ ECB sử dụng khóa từ ứng dụng



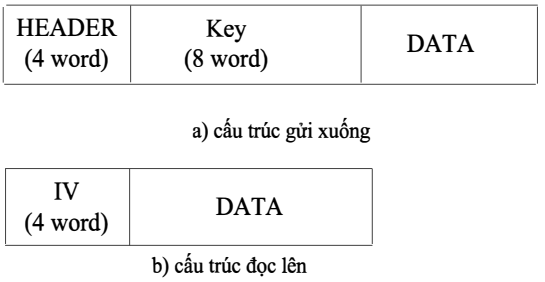
Hình 6. Cấu trúc gói tin mã hóa/giải mã theo chế độ ECB sử dụng khóa từ thiết bị



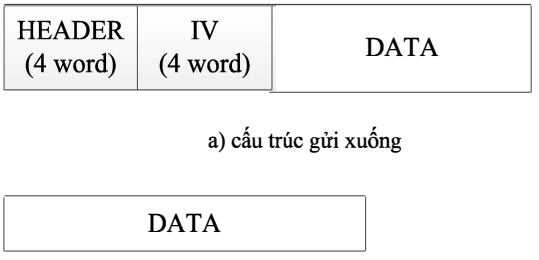
Hình 7. Cấu trúc gói tin mã hóa/giải mã theo chế độ OFB/CTR sử dụng khóa và IV từ ứng dụng



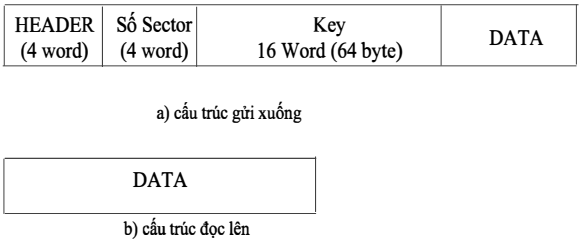
Hình 8. Cấu trúc gói tin mã hóa/giải mã theo chế độ OFB/CTR sử dụng khóa và IV từ thiết bị



Hình 9. Cấu trúc gói tin mã hóa/giải mã theo chế độ OFB/CTR sử dụng khóa từ ứng dụng và IV từ thiết bị



Hình 10. Cấu trúc gói tin mã hóa/giải mã theo chế độ OFB/CTR sử dụng khóa từ thiết bị và IV từ ứng dụng



Hình 11. Cấu trúc gói tin mã hóa/giải mã theo chế độ XTS

IV. THỬ NGHIỆM

Sau khi xây dựng thành công bộ thư viện người dùng dành cho card tăng tốc mã hóa qua

giao tiếp PCIe, nhóm tác giả đã xây dựng một chương trình để thử nghiệm. Nhóm tác giả đã sử dụng bo mạch DE4 với vai trò là card tăng tốc mã hóa kết nối với máy tính qua giao tiếp PCIe. Trong đó, máy tính sẽ chạy ứng dụng mã hóa/giải mã dữ liệu có sử dụng card tăng tốc mã hóa. Đầu tiên phần mềm máy tính sẽ khởi tạo dữ liệu đầu vào ngẫu nhiên, gửi xuống thiết bị để mã hóa, dữ liệu mã hóa nhận được sẽ tiếp tục được gửi xuống thiết bị để giải mã. Kết thúc giải mã, phần mềm sẽ so sánh kết quả giải mã với dữ liệu ban đầu. Kết quả của quá trình thử nghiệm được thể hiện như trong Hình 12 và Hình 13.

```
Select E:\2021\PCIe_Card\test_ky - Copy\x64\Debug\CLK_PCIE_test.exe

Che do doc ghi ro:
Toc do: 2.4467 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200

Che do ECB:
   Che do su dung khoa tu ung dung:
Toc do ma hoa: 2.3090 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 2.4071 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200
   Che do su dung khoa tu thiet bi:
Toc do ma hoa: 2.4467 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 2.4120 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200

Che do CTR:
   Che do su dung khoa va IV tu thiet bi:
Toc do ma hoa: 1.9687 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 2.2956 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200
   Che do su dung khoa tu thiet bi, IV tu ung dung:
Toc do ma hoa: 2.1234 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 2.1046 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200
   Che do su dung khoa va IV tu ung dung:
Toc do ma hoa: 2.1387 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 2.2521 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200
   Che do su dung khoa tu ung dung, IV tu thiet bi:
Toc do ma hoa: 2.2268 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 2.2693 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200
```

Hình 12. Kết quả thử nghiệm với chế độ đọc ghi Rõ, ECB và CTR

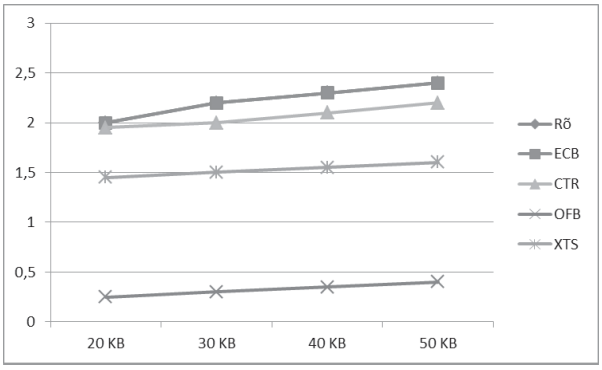
```
Select E:\2021\PCIe_Card\test_ky - Copy\x64\Debug\CLK_PCIE_test.exe

Che do OFB:
   Che do su dung khoa va IV tu thiet bi:
Toc do ma hoa: 0.4058 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 0.4112 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200
   Che do su dung khoa tu thiet bi, IV tu ung dung:
Toc do ma hoa: 0.4079 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 0.4126 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200
   Che do su dung khoa va IV tu ung dung:
Toc do ma hoa: 0.4082 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 0.4110 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200
   Che do su dung khoa tu ung dung, IV tu thiet bi:
Toc do ma hoa: 0.3997 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 0.4113 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200

Che do XTS:
Toc do ma hoa: 1.4864 Gb/s, dung luong file: 0.0488 MB
Toc do giai ma: 1.6585 Gb/s, dung luong file: 0.0488 MB
        So tu sai: 0/51200
```

Hình 13. Kết quả thử nghiệm với chế độ OFB và XTS

Nhóm tác giả đã tiến hành thử nghiệm nhiều lần, kết quả cho thấy bộ thư viện người dùng dành cho card tăng tốc mã hóa qua giao tiếp PCIe hoạt động ổn định, đạt được yêu cầu đặt ra. Kết quả thử nghiệm cụ thể được thể hiện trong Hình 14.



Hình 14. Kết quả thử nghiệm tốc độ mã hóa/giải mã (tính theo Gb/s)

V. KẾT LUẬN

Bài báo đã trình bày việc nghiên cứu giải pháp thiết kế, xây dựng bộ thư viện người dùng dành cho card tăng tốc mã hóa qua giao tiếp PCIe. Bộ thư viện này giúp các ứng dụng người dùng sử dụng card tăng tốc mã hóa một cách dễ dàng, linh hoạt. Các kết quả thử nghiệm trên bo mạch DE4 cho thấy, bộ thư viện người dùng dành cho card tăng tốc mã hóa qua giao tiếp PCIe đã hoạt động ổn định, đáp ứng các yêu cầu đặt ra. Với giải pháp này, nhóm tác giả đã làm chủ hoàn toàn bộ thư viện người dùng, giúp hoàn thiện bộ thư viện sát nhất với yêu cầu thực tế.

TÀI LIỆU THAM KHẢO

[1] Ky, P. V., Cuong, V. T., & Phuc, L. H. (2021). Solution for Cryptographic Intervention in PCI-Express Data Transmission on FPGA Board. Journal of Science and Technology on Information Security, 2(12), 59-68.

[2] IBM, "IBM CEX6S (4768) PCIe Cryptographic Coprocessor (HSM)", URL: http://www.ibm.com/security/cryptocards, [14-12-2021].

[3] UTIMACO, "SecurityServer Se-Series Gen2". URL: https://hsm.utimaco.com, [14-12-2021].

[4] Thales Luna PCIe HSM 7. URL: https://cpl.thalesgroup.com/encryption/hardware-security-modules/pcie-hsms, [14-12-2021].

[5] Altera Corp, "DE4 User manual", URL: ftp://ftp.intel.com/Pub/fpgaup/pub/Intel_Materi

al/13.0/Tutorials/Using_PCIe_on_DE4.pdf, [14-12-2021].

[6] Altera, “Using PCI Express on DE4 Boards”, URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_pci_express.pdf, [14-12-2021].

[7] Altera, “IP Compiler for PCI Express User Guide”, URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_pci_express.pdf, [14-12-2021].

[8] Xilinx “PCI Express PHY LogiCORE IP Product Guide”, URL: https://www.xilinx.com/support/documentation/ip_documentation/pcie_phy/v1_0/pg239-pcie-phy.pdf, [14-12-2021].

SƠ LƯỢC VỀ TÁC GIẢ

Vũ Tá Cường

Đơn vị công tác: Viện Khoa học - công nghệ mật mã, Ban Cơ yếu Chính phủ.

Email: vutacuong109@gmail.com

Quá trình đào tạo: Nhận bằng Đại học năm 2011; Thạc sĩ năm 2013 và Tiến sĩ năm 2016 chuyên ngành Vô tuyến kỹ thuật tại Đại học Hàng không vũ trụ Kharkov, Ukraina.

Hướng nghiên cứu hiện nay: Kỹ thuật mật mã.

Phan Văn Kỷ

Đơn vị công tác: Viện Khoa học - công nghệ mật mã, Ban Cơ yếu Chính phủ.

Email: pvk.hvktqs@gmail.com

Quá trình đào tạo: Nhận bằng Đại học năm 2013 và Thạc sĩ năm 2017 chuyên ngành Vô tuyến kỹ thuật tại Đại học tổng hợp kỹ thuật Điện St. Peterburg, Liên Bang Nga.

Hướng nghiên cứu hiện nay: Công nghệ vi mạch; FPGA.