ĐỀ XUẤT KIẾN TRÚC VÀ ĐÁNH GIÁ THỬ NGHIỆM KHỐI MÃ HÓA KHÔNG DÙNG SBOX CHO HOẠT ĐỘNG TRUYỀN THÔNG CỦA CÁC HỆ THỐNG IOT

TÓM TẮT

Hoạt động truyền thông luôn là thách thức chính của các thiết kế IoT hiện đại, giải quyết bài toán truyền thông hiệu quả cân bằng giữa chi phí (năng lượng, năng lực xử lý) và hiệu quả góp phần chính vào thành công chung. Vấn đề bảo mật IoT tuy không mới nhưng vẫn luôn là một thách thức lớn, nhất là với các hệ thống IoT trong không gian rộng khả năng bao quát của người dùng hạn chế. Bài báo này được phát triển theo xu thế tích hợp chức năng bảo mật vào bên trong các hệ thống IoT với ưu điểm tiêu thụ năng lượng thấp, không yêu cầu năng lực xử lý và hiệu quả truyền thông cao. Thuật toán bảo mật không sử dụng sbox cho phép người thiết kế tối ưu chi phí thiết kế thực thi để từ đó tối đa hóa hiệu quả chung của toàn hệ thống IoT.

Từ khóa: Bảo mật, truyền thông IoT, COMET, blockcipher.

1. MỞ ĐẦU

Hiện nay xu thế nghiên cứu và triển khai ứng dụng theo định hướng IoT đang là một hướng rất được quan tâm và cũng mang lại nhiều lợi ích to lớn. Các hệ thống IoT góp phần nâng cao hiệu quả cuộc sống, gia tăng năng xuất lao động và hơn thế nữa là góp phần quan trọng trong công cuộc phòng chống dịch bệnh. Các thiết bị IoT cho phép tạo lập dữ liệu sâu rộng và đa dạng và sau đó sẽ chuyển tải về trung tâm xử lý; đây là thành tố cơ bản để tạo nên dữ liệu lớn cho tương lai. Hoạt động truyền thông giữa các thành phần của hệ thống IoT cũng như giữa các hệ thống lại có những ràng buộc rất khác nhau mà một số tiêu chuẩn bảo mật đang có chưa thể đáp ứng một cách tối ưu với các nhu cầu năng lượng, hiệu quả xử lý, hiệu quả truyền thông.

Tiêu chuẩn mật mã hiện đang phổ biến nhất là thuật toán AES được NIST công bố trong [1]; nhưng chủ yếu vẫn dùng cho hoạt động truyền thông giữa các máy tính theo chuẩn Internet. Với các hệ thống IoT với ràng buộc về tiêu hao năng lượng, hiệu

năng xử lý không mạnh nhưng lại yêu cầu tính bảo mật lại rất cao thì AES chưa phải là một lựa chọn phù hợp. Do vậy, Viện tiêu chuẩn và công nghệ Mỹ - NIST đã và đang tiến hành các bước để tuyển chọn giải thuật mật mã phù hợp cho các hệ thống công suất thấp tương đồng với xu thế IoT[5].

Thuật toán COMET (*COunter Mode Encryption with authentication Tag*) là một trong những thuật toán được đề xuất cho việc xét tuyển chuẩn hóa của NIST nói trên với đặc điểm nổi trội là không dùng khối sbox mà đa phần các giải thuật bảo mật dạng khóa công khai vẫn đang dùng. Thuật toán COMET được phát triển dựa trên mô hình CHAM đã được công bố tại [3] với trọng tâm là hàm lặp cho phép xoắn dữ liệu đầu vào với khóa bảo mật hoặc vẫn sử dụng nền của thuật toán AES đã có.

Tuy nhiên, giải thuật COMET nói riêng và hầu hết các giải thuật tham gia các vòng lựa chọn để tạo thành chuẩn hóa của NIST vẫn ở dạng ý tưởng và đề xuất phương pháp xử lý số liệu ở mức hệ thống. Giải thuật thường được trình bày ở dạng toán có kết hợp với mã nguồn ngôn ngữ C. Đây là một thách thức lớn đối với định hướng nghiên cứu thiết kế phần cứng khi người dùng cần phải nắm bắt tốt các phương pháp xử lý số liệu cụ thể đồng thời phải có giải pháp chuyển hóa quy trình xử lý tuần tự thành hoạt động song song thời gian thực. Do vậy, trong khuôn khổ bài báo này chúng tôi tập trung vào hoạt động thực thi mã hóa khối dữ liệu của thuật toán COMET và từ đó đề xuất nên kiến trúc khả thi thực hiện chức năng mã hóa phù hợp và đề xuất giải pháp kiểm thử hoạt động của khối thiết kế.

Nội dung chính của bài báo được trình bày từ cơ bản về thuật toán COMET trong phần 2, phần 3 tập trung mô tả kiến trúc mà chúng tôi đề xuất để thực thi khối mã hóa, phần 4 là trình bày về kịch bản kiểm thử và kết quả kiểm thử đã thu được. Phần 5 chúng tôi đưa ra kết luận chung của công việc đã thực hiện.

2. Thuật toán mã hóa COMET

Thuật toán mã hóa COMET viết tắt từ cụm thuật ngữ *COunter Mode Encryption with authentication Tag* – Mã hóa chế độ đếm với thẻ xác thực là một thuật toán mật mã được đề xuất cho hoạt động truyền thông công suất thấp hướng đến các hệ thống IoT tiên tiến. Thuật toán COMET là mật mã khối hoạt động theo nguyên lý cung cấp xác thực mã hóa kết hợp với dữ liệu quan hệ - AEAD (*authenticated encryption with associated data*) [4]. Đây cũng là một trong những thuật toán được vào vòng 2 của quy trình lựa chọn bộ mã chuẩn hóa cho các liên kết trong hệ thống IoT của NIST. Thuật toán COMET được đề xuất theo cả hai nguyên lý là AES và CHAM; đồng thời cũng hỗ trợ hai dạng kích thước khối là khối 64bit và khối 128bit.

Tuy nhiên, trong bài viết này chúng tôi tập trung vào nghiên cứu giải thuật để xây dựng nên kiến trúc khối mã ứng dụng cho hoạt động bảo mật với khối 128bit nên

phần trình bày giải thuật chúng tôi cũng chỉ tập trung cho hoạt động mã hóa khối 128bit sử dụng nguyên lý CHAM128 [3].

2.1. Mô tả khối mã khóa - CHAM

CHAM được xây dựng dựa trên mô hình thiết kế ARX mà không sử dụng khối sbox như nhiều thuật toán bảo mật phổ biến hiện nay. CHAM có kích thước khối thường dùng là n = {64,128} cùng với kích thước khóa $k = \{n, 2n\}$, nhưng thuật toán COMET chỉ tập trung vào trường hợp $k = n$. CHAM hoạt động theo phương thức lặp vòng với số lần lặp là 80. Bên trong mô hình CHAM hoạt động đan chéo theo đơn vị từ, mỗi khối sẽ gồm 4 từ; nên với CHAM128/128 sẽ sử dụng từ 32bit và CHAM64/128 sẽ sử dụng từ 16bit. Số lượng vòng lặp và kích thước từ được ký hiệu lần lượt là r và w. Mô hình CHAM-n/k mã khóa một bản tin $P \in \{0,1\}^n$ thành một bản mã $C \in \{0,1\}^n$ sử dụng một khóa bảo mật là $K \in \{0,1\}^n$ bằng cách sử dụng r hàm vòng như sau:

1. P được phân chia thành 4 từ w-bit là $S_3^0, S_2^0, S_1^0, S_0^0$
2. Tại mỗi lần lặp thứ i, $i \in \{0, \cdots, r-1\}$ thì đầu ra là S^{i+1} được tính như sau
   Ứng với các lần lặp chẵn sẽ là:

$$S_3^{i+1} = \left((S_0^i \oplus i) \boxplus \left((S_1^i \lll 1) \oplus R_{i\%\frac{2k}{w}} \right) \right) \lll 8$$

$$S_j^{i+1} = S_{j+1}^i \; với \; 0 \le j \le 2$$

   Đối với các lần lặp lẻ

$$S_3^{i+1} = \left((S_0^i \oplus i) \boxplus \left((S_1^i \lll 8) \oplus R_{i\%\frac{2k}{w}} \right) \right) \lll 1$$

$$S_j^{i+1} = S_{j+1}^i \; với \; 0 \le j \le 2$$

   Trong đó, $\oplus$ là phép cộng module 2 cho từng bit
   $\boxplus$ là phép cộng module 2^w
   $\lll$ là phép dịch vòng về phía trái
3. Phân chia khóa của CHAM n/k được lấy từ khóa bảo mật $K \in \{0,1\}^n$ và tạo ra $2k/w$ khóa vòng w-bit ký hiệu lần lượt là $R_0, R_1, \cdots, R_{2k/n-1}$. Ở đầu vào mỗi lần thực hiện khóa mã sẽ được tạo ra như sau:

$$R_i = K_i \oplus (K_i \lll 1) \oplus (K_i \lll 8)$$

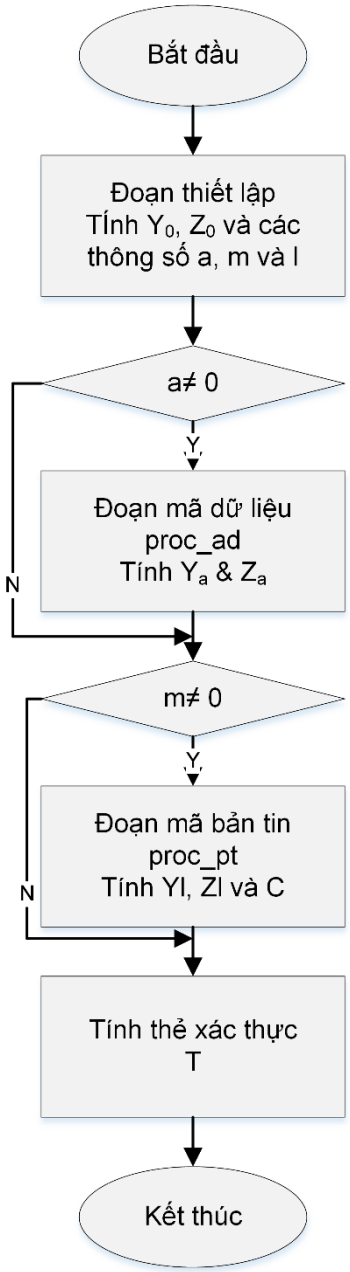
$$R_{(i+k/n)\oplus 1} = K_i \oplus (K_i \lll 1) \oplus (K_i \lll 11)$$

   Trong đó, $i \in \{0, \cdots, k/w\}$

2.2. Thuật toán mã hóa COMET

Thuật toán mã hóa COMET được đề xuất lấy khối mã khóa CHAM là trung tâm để tạo khóa mã từ khối khóa K và khối từ tin đầu vào PT như đã trình bày trong phần trên. Hoạt động mã hóa được chia thành ba đoạn gồm: đoạn thiết lập init_state; đoạn mã khóa dữ liệu và mã khóa bản tin.

Đoạn thiết lập – Init-state có chức năng xoắn trộn dữ liệu đầu vào bộ mã và tính toán các thông số phù hợp với từng trường hợp mã hóa.



Hình 1: Lưu đồ thực thi quá trình mã hóa COMET

Đoạn mã khóa dữ liệu thực hiện chức năng xoắn trộn dữ liệu đầu vào sau khi được chia thành các khối 128bit hoặc nhỏ hơn và sử dụng hàm lặp 80 lần để mã hóa dữ liệu cho từng khối dữ liệu đã được chuẩn bị. Sau đó dữ liệu ra được ghép nối trở lại thành chuỗi tin mã hóa phù hợp.

Đoạn mã hóa bản tin thực hiện tương tự như với mã khóa dữ liệu nhưng đầu vào là bản tin chưa mã hóa và dữ liệu đầu ra cũng là sự phối ghép các đoạn mã hóa theo khối tiêu chuẩn 128bit.

Hoạt động tính thẻ xác thực là hoạt động cuối cùng của chuỗi quá trình mã hóa để thu được thẻ xác thực là khối dữ liệu 128bit. Toàn bộ quá trình mã hóa COMET sử dụng CHAM được trình bày trong **Hình 1**.

Phần lõi của cả ba đoạn trên vẫn là hàm lặp vòng 80 lần CHAM đã được trình bày ở phần 2.1. Các hàm tính đều sử dụng hàm lặp vòng CHAM với các dữ liệu đầu vào thích hợp.

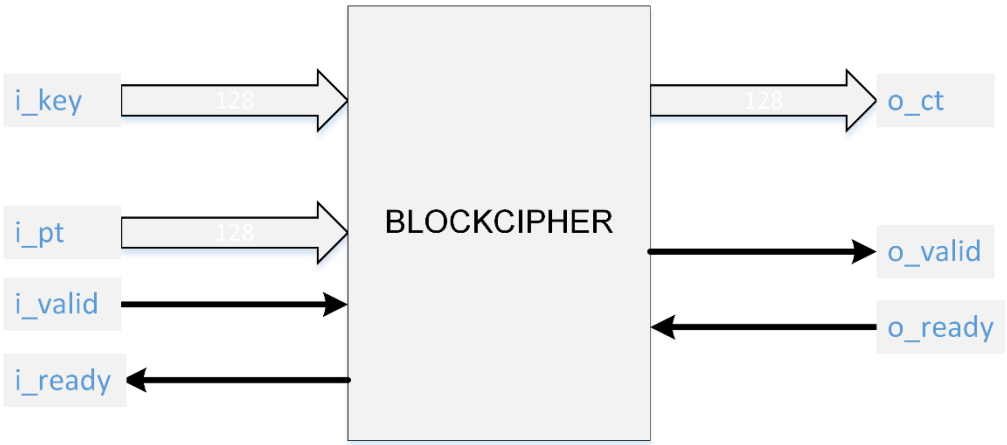
3. KIẾN TRÚC ĐỀ XUẤT CHO KHỐI MÃ HÓA

Giải thuật COMET hiện chỉ được giới thiệu chung và thực thi bằng mã nguồn mở trên nền ngôn ngữ C++ dùng chung cho các hệ thống tính toán đa dụng và còn chưa được phát triển thành các kiến trúc khả thi cho các ứng dụng cụ thể. Vì vậy, trong bài báo này chúng tôi hướng đến mục tiêu là đề xuất một kiến trúc khả thi cho cả công nghệ FPGA và ASIC. Hơn nữa, chúng tôi sử dụng công cụ mô tả phần cứng mới được phát triển với nhiều hứa hẹn với các tính năng nổi bật là CHISEL 3 [2].

Khối mã hóa được đề xuất với kích thước khối 128bit có chức năng xoắn dữ liệu giữa khối khóa 128bit và khối từ tin 128bit đầu vào để tạo nên khối dữ liệu đã khóa mã 128bit dựa trên mô hình CHAM đã trình bày trong phần 2 của bài báo này.

3.1. Mô tả kiến trúc chung

Từ hoạt động vào/ra dữ liệu của khối mã CHAM như trình bày ở trên được kế thừa từ mã nguồn mở C++ thì theo định hướng phát triển kiến trúc khả thi cho việc tổng hợp thiết kế trên các nền tảng công nghệ cần được đặc tả chi tiết hơn. Trong công trình này chúng tôi tập trung cho mục tiêu chứng minh tính khả thi của thuật toán COMET đã được đề xuất nên hạn chế tối đa các kỹ thuật chuyên sâu trong hoạt động thiết kế phần cứng. Vậy nên khối mã hóa được đề xuất với hai BUS dữ liệu vào có độ rộng 128bit tương ứng với các đầu vào bản tin và đầu vào khối khóa, bên cạnh đó chúng tôi bổ sung thêm hai tín hiệu bắt tay dùng chung cho cả hai BUS đầu vào là *i_ready* và *i_valid* có chức năng tương ứng là báo trạng thái sẵn sàng và nhận xác thực dữ liệu vào.



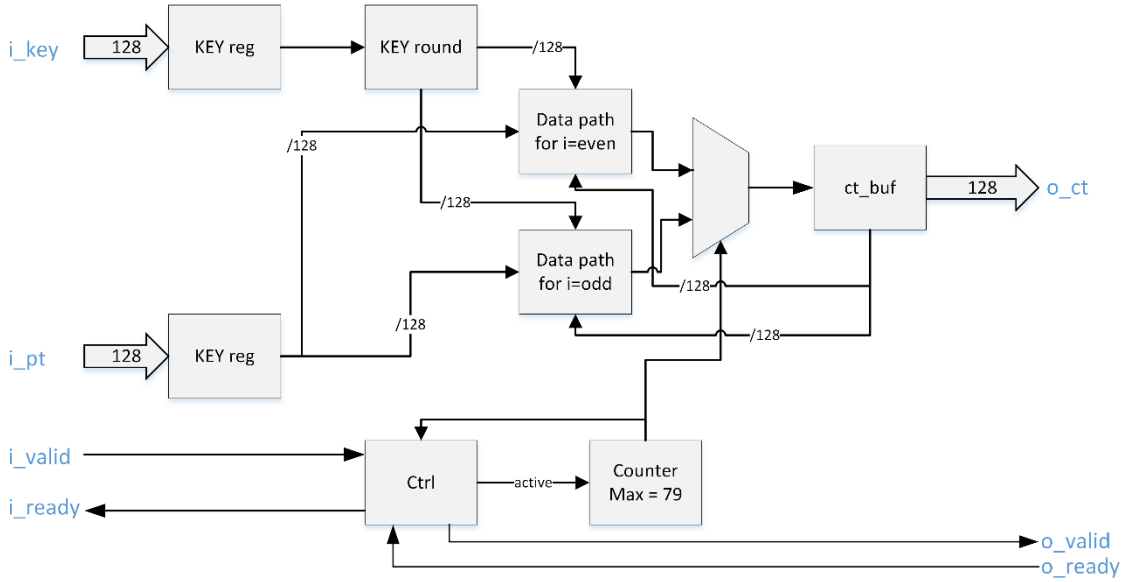
***Hình 2**: Kiến trúc chung cho khối mã hóa*

Ở phía đầu ra chúng tôi chỉ đề xuất một BUS dữ liệu ra có độ rộng 128bit kết hợp với hai tín hiệu giao tiếp là *o_valid* và *o_ready*. Trong đó *o_valid* là tín hiệu hướng ra có chức năng báo hiệu dữ liệu ra đã sẵn sàng để khối tiếp theo có thể nhận được dữ liệu tin cậy. Tín hiệu *o_ready* là tín hiệu hướng vào nhằm kiểm tra trạng thái nhận dữ liệu của khối liền sau, nếu tín hiệu sẵn sàng không tích cực thì dữ liệu ra phải được lưu giữ ổn định chờ khối liền sẵn sàng đọc vào. Do đó, trong quá trình chờ khối liền sau đọc dữ liệu thì tín hiệu trạng thái trả về cho khối liền trước cũng phải ở trạng thái không tích cực.

Kiến trúc chung cho khối mã hóa được mô tả chi tiết trong **Hình 2**

3.2. Mô tả chi tiết

Thuật toán COMET được đề xuất dưới dạng mã nguồn ngôn ngữ C++ cho các vi xử lý 8bit nên chủ yếu sử dụng các kỹ thuật xử lý thông tin của ngôn ngữ C++ có kết hợp kỹ thuật con trỏ. Kỹ thuật con trỏ và truyền vị trí nhớ của các biến cho phép vi xử lý tính toán nhanh và hiệu quả. Tuy nhiên, hoạt động xử lý thông tin dựa trên các kiến trúc phần cứng thì không thể sử dụng kỹ thuật con trỏ mà phải được đặc tả chi tiết hoạt động vào/ra và chuyển hóa dữ liệu ở dạng song song đồng bộ.



***Hình 3**: Kiến trúc chi tiết khối mã hóa*

Thuật toán COMET thường dùng các kỹ thuật ghi dịch dữ liệu trong toàn quá trình tính toán của mình, nhưng ở ngôn ngữ C++ nhóm tác giả chủ yếu sử dụng các từ 8bit nên là cho hoạt động xoắn dữ liệu phức tạp và ít hiệu quả. Trên nền tảng Chisel chúng tôi sử dụng kỹ thuật cắt ghép với từ 32bit khá đơn giản như

$$Y = Cat(I(23,0), I(31,24))$$

Thì tương đương với việc xử lý ghi dịch vòng phía trái 8bit của một từ 32bit đầu vào là I thành từ đầu ra 32bit là Y.

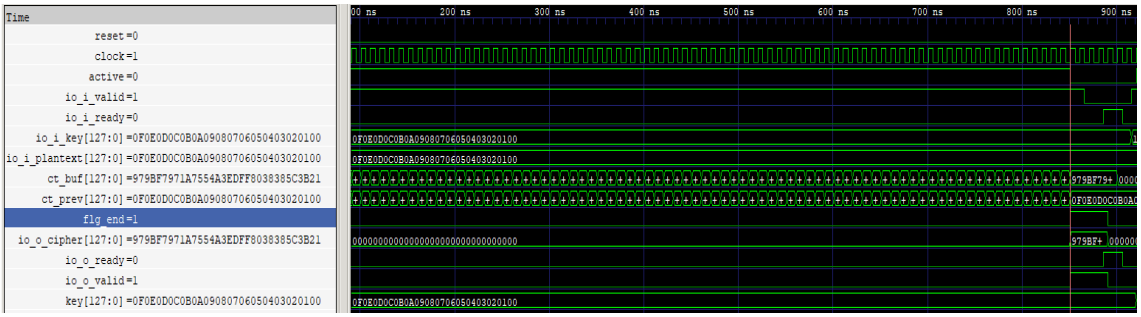
Kiến trúc chi tiết của khối mã hóa được mô tả như trong **Hình 3**

Toàn bộ kiến trúc này đã được chúng tôi đặc tả bằng ngôn ngữ CHISEL 3 kèm theo đặc tả đánh giá cũng được viết bằng ngôn ngữ CHISEL 3.

4. MÔ PHỎNG ĐỂ ĐÁNH GIÁ THỬ NGHIỆM

4.1. Đánh giá trong một vòng lặp

Kiến trúc đề xuất đã được chúng tôi tiến hành mô phỏng đánh giá một cách chi tiết các hoạt động xoắn dữ liệu trong vòng lặp với r = 80. Kết quả được cho trong Hình 4 là giản đồ xung của một chu trình lặp theo đúng như hàm lặp của thuật toán COMET đã công bố trước đó.



Hình 4: Giản đồ xung mô tả hoạt động mã với vòng lặp r=80

Kết quả kiểm chứng với giá trị vào ra trên nền code C++ đã có

```
mlen = 0 - adlen = 0 - mlen_blocks = 0 - adlen_blocks = 0
message:  0
ad:  0
npub:  f e d c b a 9 8 7 6 5 4 3 2 1 0
Blockcipher key =  f e d c b a 9 8 7 6 5 4 3 2 1 0
Blockcipher pt =  f e d c b a 9 8 7 6 5 4 3 2 1 0
Blockcipher ct =  97 9b f7 97 1a 75 54 a3 ed ff 80 38 38 5c 3b 21
Blockcipher key =  17 9b f7 97 1a 75 54 a3 db ff 0 70 70 b8 76 59
Blockcipher pt =  f e d c b a 9 8 7 6 5 4 3 2 1 0
Blockcipher ct =  24 2c c8 65 b b7 d7 30 f4 d5 b6 aa 36 4f 74 4The end encrypt for adlen = 0
Blockcipher key =  f e d c b a 9 8 7 6 5 4 3 2 1 0
Blockcipher pt =  f e d c b a 9 8 7 6 5 4 3 2 1 0
Blockcipher ct =  97 9b f7 97 1a 75 54 a3 ed ff 80 38 38 5c 3b 21
Blockcipher key =  17 9b f7 97 1a 75 54 a3 db ff 0 70 70 b8 76 59
Blockcipher pt =  f e d c b a 9 8 7 6 5 4 3 2 1 0
Blockcipher ct =  24 2c c8 65 b b7 d7 30 f4 d5 b6 aa 36 4f 74 4
---------------------------------
```

***Hình 5**: Giá trị vào ra khối mã hóa tương ứng trong code C++*

Từ hai kết quả cho thấy, với các giá trị đầu vào là:

- Khóa KEY=0F0E0D0C0B0A09080706050403020100 và

- Khối tin vào pt = 0F0E0D0C0B0A09080706050403020100

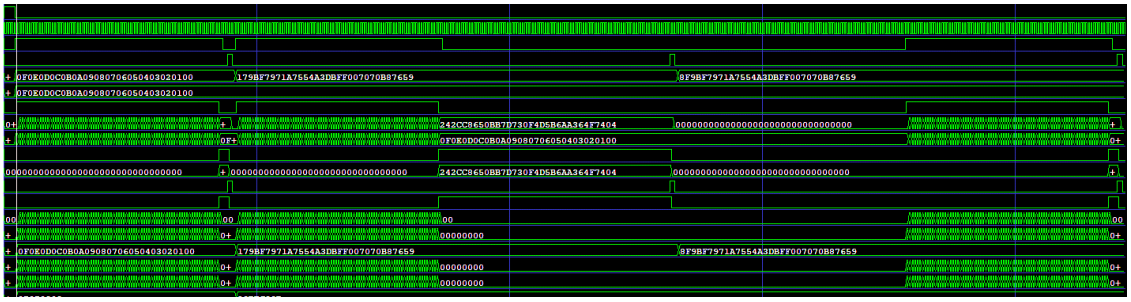
- Khối mã ra ct = 97 9b f7 97 1a 75 54 a3 ed ff 80 38 38 5c 3b 21

Điều này khẳng định được tính chính xác của khối mã chúng tôi đã tiến hành xây dựng.

4.2. Đánh giá hoạt động trao đổi dữ liệu với đầu ra

Hoạt động đánh giá còn được chúng tôi thử nghiệm trong một số trường hợp đầu ra đã sẵn sàng nhận vừa chưa sẵn sàng nhận. Hoạt động đánh giá này là cần thiết để chứng minh được khả năng thực thi lên phần cứng của giải thuật từ ý tưởng ban đầu chỉ trên nền C++ sẽ rất khác so với việc triển khai trên thực tế phần cứng.

Quá trình khảo sát đánh giá được trình bày trong Hình 6. Trong mô phỏng này chúng tôi đưa ra ba trường hợp truyền thông chính gồm: khối đầu ra luôn sẵn sàng nhận dữ liệu; khối đầu ra chưa sẵn sàng nhận dữ liệu và dữ liệu đầu vào chưa có tín hiệu cập nhật để khối mã hóa chấp nhận dữ liệu.



Hình 6: Giản đồ xung cho khảo sát trao đổi dữ liệu

5. KẾT LUẬN

Sau thời gian nghiên cứu thuật toán mã hóa mà trọng tâm là hàm lặp vòng theo mô hình CHAM chúng tôi đã bước đầu đề xuất được kiến trúc khả thi cho khối hàm lặp thực hiện hoạt động xoắn dữ liệu và chìa khóa với kích thước khối là 128bit.

Đồng thời chúng tôi cũng đã bổ sung và hoàn thiện được cơ chế trao đổi thông tin hai chiều giữa khối mã hóa với các thành phần khác của hệ thống nhằm góp phần khẳng định tính khả thi của giải thuật COMET trong hoạt động bảo mật cho truyền thông trong các hệ thống IoT hiện đại.

Trong thời gian tới thiết kế này cần được hoàn thiện hơn với các chức năng đệm dữ liệu và bổ sung chức năng điều khiển chung cho toàn bộ khối mật mã theo đúng giải thuật COMET đã đề xuất.

LỜI CẢM ƠN

Nghiên cứu này được thực hiện trong khuôn khổ Đề tài cấp Đại học Huế mã số DHH2019-01-146.

TÀI LIỆU THAM KHẢO

[1]. NIST(2001): Announcing the ADVANCED ENCRYPTION STANDARD (AES). Fedral Information Processing Standards Publication FIPS 197, National Institute of Standards and Technology, U. S. Department of Commerce

[2]. Bachrach, J., Vo, H., Richards, B., Lee, Y., Waterman, A., Avižienis, R., . . . Asanović, K. (2012). Chisel: Constructing hardware in a Scala embedded language. DAC Design Automation Conference 2012, (pp. 1212-1221). doi:10.1145/2228360.2228584

[3]. Bonwook Koo, D. R.-G. (2017). CHAM: A family of lightweight block ciphers for resource-constrained devices. ICISC 2017 - 20th International Conference, (pp. 3-25). Seoul, South Korea.

[4]. Shay Gueron, A. J. (2019). COMET: COunter Mode Encryption with authentication Tag. USA: National Institute of Standards and Technology, USA.

[5]. NIST. (n.d.). https://csrc.nist.gov/projects/lightweight-cryptography.

PROPOSED ARCHITECTURE AND TESTBENCH A BLOCKCIPHER WITHOUT SBOX USING FOR COMMUNICATION IN IOT SYSTEMS

ABSTRACT

. Communication activities are the main issue when we design advantage IoT systems; solving the communication effective with trade-off design costs (energy, processing cost) with performance is always a key issue. Although IoT cryptography is not new, but it is always a big challenge; especially for IoT systems used in large space with limited user coverage. The article is developed according to the trend of integrated cryptography function into IoT systems with the advantage of low power consumption, no processing capacity and high communication efficiency. The cryptography algorithm without-sbox allows designers to optimize implement cost to maximize the efficiency of IoT systems.

Keywords: COMET, cryptography, blockcipher, IoT

GHI CHÚ:

Sau khi bài viết đã được Hội đồng biên tập chọn đăng, đề nghị tác giả cung cấp 1 file ảnh chân dung và một số thông tin lý lịch ngắn gọn bằng tiếng Việt, bao gồm: họ và tên, ngày sinh, nơi sinh, chức danh, chức vụ, sơ lược quá trình học tập và công tác (những mốc thời gian chính, năm và nơi cấp học hàm, học vị, …), lĩnh vực nghiên cứu và thành tích...

Những thông tin về tác giả sẽ được in vào cuối mỗi bài viết.