HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Phạm Văn Lực

NGHIÊN CỨU GIẢI PHÁP NÂNG CAO HIỆU QUẢ SỬ DỤNG MẬT MÃ ĐƯỜNG CONG ELLIPTIC TRÊN CÁC THIẾT BỊ TÍNH TOÁN NHÚNG

Chuyên ngành: **Kỹ thuật Điện tử**

Mã số: **9.52.02.03**

TÓM TẮT LUẬN ÁN TIẾN SĨ KỸ THUẬT ĐIỆN TỬ

HÀ NỘI - 2022

Công trình được hoàn thành tại:
Học viện Công nghệ bưu chính viễn thông

Phản biện 1:…… ……..
Phản biện 2: …… ……..
Phản biện 3: …… ……..

Luận án được bảo vệ trước Hội đồng chấm luận án cấp Học viện tại Học viện Công nghệ Bưu chính viễn thông, 122 Hoàng Quốc Việt, Hà Nội.
Họp tại: …… ……...
Vào hồi.....giờ.....ngày.....tháng.....năm 2022

Có thể tìm hiểu luận án tại thư viện Học viện Công nghệ Bưu chính viễn thông.

MỞ ĐẦU

Tính cấp thiết của đề tài luận án

Các thiết bị tính toán nhúng (gọi tắt là các hệ thống nhúng) đang được ứng dụng rộng rãi trong nhiều lĩnh vực, điển hình như trong các thiết bị IoT (Internet of Things), thiết bị di động, thiết bị cầm tay, máy tính bảng, thẻ thông minh, các bộ điều khiển trong xe ô tô, các bộ cảm biến môi trường, các hộp kỹ thuật số,…Theo thống kê, các thiết bị sử dụng hệ thống nhúng chiếm tới 90% các thiết bị tính toán hiện nay. Do chủng loại và môi trường kết nối đa dạng, nhu cầu bảo đảm an toàn và bảo mật thông tin cho các thiết bị di động nói chung và các hệ thống nhúng nói riêng đang trở nên cấp thiết. Một lý do nữa là những thiết bị này thường là không dây, khả năng truy cập vật lý mọi lúc - mọi nơi, vấn đề an toàn bảo mật dữ liệu và bảo vệ truy cập để ngăn chặn tấn công nghe lén và rò rỉ thông tin cá nhân là rất quan trọng.

Đặc điểm chung của các hệ thống nhúng là có hạn chế về tài nguyên và năng lực xử lý. Việc triển khai các giải pháp an toàn bảo mật thông tin cho các hệ thống nhúng có thêm nhiều thách thức so với các hệ thống máy tính truyền thống.

Mục tiêu nghiên cứu

Nghiên cứu, đề xuất các giải pháp để nâng cao hiệu quả thực thi một số thuật toán mật mã đường cong elliptic trên nền hệ thống nhúng sử dụng vi xử lý ARM, đảm bảo hiệu quả về tốc độ tính toán, tài nguyên sử dụng và an toàn trong sử dụng.

Ý nghĩa khoa học và đóng góp

Các giải pháp được đề xuất trong luận án sẽ góp phần tăng cường khả năng bảo mật thông tin trên các thiết bị nhúng, đáp

ứng nhu cầu cấp thiết trong thực tiễn hiện nay về bảo mật dữ liệu, ngăn chặn tấn công nghe lén và rò rỉ thông tin cá nhân. Luận án gồm có 02 đóng góp sau:

1) Đề xuất phương pháp nhân phân tầng hai số hạng trên trường hữu hạn dựa trên hai thuật toán nhân cơ bản là thuật toán nhân theo phương pháp phổ thông và thuật toán nhân theo phương pháp Karatsuba. Với phương pháp phân tầng, luận án đã xây dựng được thuật toán nhân có chi phí tốt nhất trong các trường hợp cụ thể cũng như đưa ra công thức để xác định chi phí của thuật toán đó. Thuật toán đề xuất đã được kiểm chứng về tính hiệu quả trên nền vi xử lý nhúng ARMv7 và ARMv8.

2) Đề xuất, cải tiến thuật toán nhân vô hướng (nhân giữa một điểm và một số nguyên dương) của hệ mật đường cong Elliptic trên trường nguyên tố dựa trên đề xuất cải tiến thuật toán NAF và đề xuất nâng cao hiệu quả của các phép toán số học (cộng điểm, nhân đôi điểm) theo phương pháp song song hai phép nhân..

Bố cục của luận án

Nội dung luận án được trình bày trong 3 chương. Chương 1. Trình bày tổng quan các vấn đề nghiên cứu. Chương 2 trình bày Nâng cao hiệu quả của phép nhân số học trong trường nhị phân trên vi xử lý ARM. Chương 3 trình bày nâng cao hiệu quả phép nhân vô hướng của hệ mật ECC trong trường nguyên tố trên vi vử lý ARM.

CHƯƠNG 1. TỔNG QUAN CÁC VẤN ĐỀ NGHIÊN CỨU

1.1. Hệ thống nhúng

Hệ thống nhúng (Embedded System) là một thuật ngữ thường dùng để chỉ một thiết bị tính toán đặc biệt, có tích hợp cả phần cứng và phần mềm phục vụ các ứng dụng chuyên dụng trong nhiều lĩnh vực của đời sống hiện nay. Khác với các hệ thống phổ biến khác, ví dụ như các máy tính đa năng hay nhiều hệ thống máy tính trong điều khiển công nghiệp, các hệ thống nhúng thường có kích thước nhỏ hơn, khả năng xử lý yếu hơn, bộ nhớ nhỏ hơn và yêu cầu tiêu thụ ít năng lượng [10, 11].

Một hệ thống nhúng gồm ba thành phần chính: 1) Phần cứng; 2) Phần mềm ứng dụng; 3) Hệ điều hành hoặc một hệ thống thời gian thực (RTOS).

1.2. Bộ vi xử lý ARM trong hệ thống nhúng

1.2.1. Kiến trúc ARM (Advanced RISC Machine)

ARM là viết tắt của Advanced RISC machines, thực chất là một kiến trúc bộ xử lý sử dụng tập lệnh rút gọn (RISC = Reduced Instructions Set Computer). Kiến trúc RISC tối ưu hóa các đường dẫn trên mạch, giúp giảm mức tiêu thụ điện năng, tiết kiệm diện tích, yêu cầu tản nhiệt thấp, cung cấp mức hiệu suất vượt trội, lý tưởng cho các thiết bị nhỏ gọn có tài nguyên hạn chế.

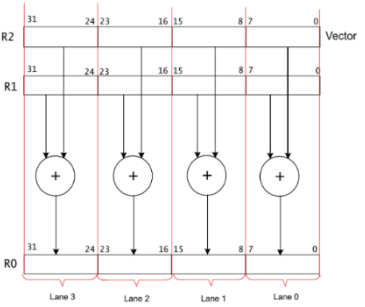
1.2.2. Các vi xử lý ARM trong thực tế

Các phiên bản ARM đã được phát triển trải rộng từ ARMv1 (năm 1985) đến ARMv8 (năm 2012) [5], và hiện nay mới nhất ARMv9 (năm 2021).

1.2.3. Kiến trúc mở rộng NEON cho ARM

Kiến trúc tính toán song song đơn lệnh-đa dữ liệu (SIMD = Single Instruction Multiple Data) được đưa vào các bộ vi xử lý

ARM từ phiên bản 6 (ARMv6). Từ phiên bản 7 (ARMv7), bộ vi xử lý ARM hỗ trợ engine tích hợp một số thuật toán đã được cứng hóa sẵn được gọi là thành phần NEON. NEON thực chất là thành phần đồng xử lý được tích hợp vào trong các dòng vi xử lý ARM [4], điển hình như Cortex-A8, Cortex-A9, Cortex-A15, Cortex-A53…



Hình 1.1: Phép toán cộng 8 bit trên 4 làn sử dụng 1 chỉ lệnh SIMD trên ARMv6

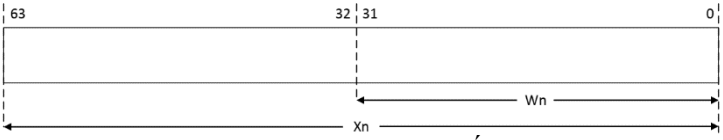
ARMv7 có 13 thanh ghi 32 bit cho mục đích chung được ký hiệu là từ R0 – R12. Thành phần NEON trong kiến trúc ARMv7 sử dụng các thanh ghi đặc biệt, những thanh ghi này tách biệt với các thanh ghi mục đích chung và được biểu diễn ở dạng thanh ghi D hai từ (64-bit) hoặc thanh ghi Q 128-bit (bốn từ). Hình dưới đây minh họa cấu trúc thanh ghi trong ARMv7.

Q0				Q1				Q2			
D0		D1		D2		D3		D4		D5	
S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11

Hình 1.2: Cấu trúc của các thanh ghi NEON trên ARMv7

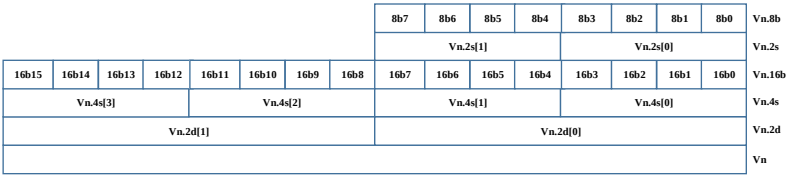
ARMv8 có kiến trúc 64-bit, kiến trúc này có tập chỉ lệnh mới (AARCH64). Kiến trúc ARMv8 có 31 thanh ghi mục đích chung

(X0-X30), đây là những thanh ghi 64-bit và nửa thấp của mỗi thanh ghi này được ký hiện là W0-W30, được biểu diễn như hình sau



Hình 1.3: Các thanh ghi trong kiến trúc ARMv8

Những vector này có thể có độ rộng 128-bit với hai hoặc nhiều phần tử; hoặc là thanh ghi có độ rộng 64-bit với một hoặc nhiều phần tử.



Hình 1.4: Truy nhập trên thanh ghi ARMv8

1.2.4. Lập trình NEON trên kiến trúc ARM

GNU GCC cung cấp các tùy chọn vector cho mã C để sinh ra mã NEON. Để cải thiện hiệu suất thực hiện của hệ thống dựa trên NEON, có hai phương pháp để viết mã lập trình là sử dụng chỉ lệnh nội tại NEON (intrinsic) hoặc lập trình trực tiếp bằng ngôn ngữ Assembly [19]. Các bước cơ bản để viết đoạn mã sử dụng tính toán trong NEON là: tải dữ liệu, thực hiện tính toán và lưu dữ liệu.

1.3. An toàn và bảo mật thông tin trên hệ thống nhúng

1.3.1 Các thách thức khi xây dựng hệ thống nhúng

So với các máy tính có năng lực xử lý mạnh, các hệ thống nhúng thường có kích thước nhỏ hơn, khả năng xử lý yếu hơn,

bộ nhớ nhỏ hơn và yêu cầu tiêu thụ ít năng lượng. Việc triển khai các giải pháp an toàn bảo mật thông tin cho các hệ thống nhúng có thêm nhiều thách thức so với các hệ thống máy tính truyền thống.

1.3.2 Mật mã trên hệ thống nhúng

Để bảo đảm an toàn và bảo mật dữ liệu cho hệ thống nhúng, các thuật toán mã hóa sử dụng mã khối và hệ mã hóa công khai cũng được sử dụng, song cần được phát triển phù hợp trong môi trường thiết bị nhúng.

Mật mã ECC đã có những tính chất ưu việt hơn về yêu cầu bộ nhớ, năng lực tính toán so với các hệ thống mật mã khác. Tuy nhiên, đối với các thiết bị nhúng có nền tảng phần cứng và tài nguyên hạn chế, việc nghiên cứu các giải pháp nhằm triển khai hiệu quả hệ thống mật mã ECC trên môi trường thực tế nói chung và trên nền các thiết bị tính toán nhúng nói riêng vẫn còn có những hạn chế và vẫn đang là chủ đề được quan tâm trong cộng đồng nghiên cứu cũng như trong thực tế triển khai trên thế giới.

1.4. Hệ mật đường cong Elliptic và ứng dụng

Hệ mật đường cong Elliptic (ECC - Ellicpic Curve Cryptography) là một hệ mật phổ biến được cài đặt trên các thiết bị tính toán nhúng do kích thước khóa nhỏ hơn nhiều so với các hệ mật khác [6]. Trong phần này, luận án tóm lược những đặc điểm cơ bản của hệ mật ECC

1.4.1. Cách biểu diễn điểm trên trường hữu hạn

Có ba dạng tạo độ cơ bản thường được sử dụng. Đó là tọa độ quan hệ (Affine coordinate), tọa độ chiếu (Projective coordinate) và tọa độ nén [7].

1.4.2 Ứng dụng của hệ mật dựa trên đường cong Elliptic

Có nhiều ứng dụng được phát triển trên nền mật mã đường cong elliptic, trong số đó phải kể đến là thuật toán chữ ký số ECDSA, thuật toán trao đổi khóa ECDH, ECMQV, ECHMQV, thuật toán mã hóa ECIES... Các ứng dụng này đều xây dựng trên nền ECC với phép tính mật mã cơ bản là phép nhân vô hướng của một điểm trên đường cong với một số nguyên.

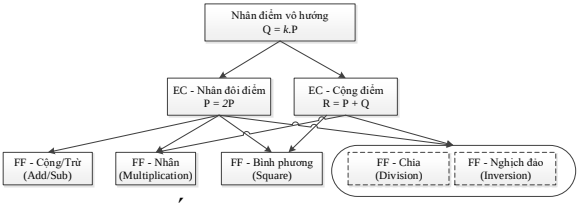
1.4.3 Ứng dụng ECDH và ECDSA trong bảo mật truyền dữ liệu trên thiết bị nhúng.

IKEv2 [57] và TLS [58] là hai giao thức trao đổi khóa phổ biến hiện nay mà sử dụng giao thức trao đổi khóa ECDH và giao thức ký số ECDSA. IKEv2 là một giao thức trao đổi khóa cho IPSec VPN, trong khi đó TLS được sử dụng làm giao thức trao đổi khóa cho OpenVPN.

1.5. Hiệu quả sử dụng mật mã đường cong Elliptic trên thiết bị nhúng và các nghiên cứu liên quan

1.5.1. Sử dụng mật mã đường cong Elliptic trên thiết bị nhúng

Phân cấp của các phép toán số học cơ bản trên EC như hình dưới đây:



Hình 1.5: Cấu trúc của phép toán trên EC

1.5.2. Các nghiên cứu liên quan

Các nghiên cứu liên quan có thể được phân chia thành 3 nhóm chính là: 1) Cải tiến thuật toán và lựa chọn tham số phù hợp; 2)

Áp dụng các kỹ thuật đặc thù cho thiết bị nhúng; 3) Giải pháp sử dụng phần cứng đặc thù trên thiết bị nhúng.

1.5.3. Đánh giá, nhận xét

Một số hạn chế của những phương pháp nghiên cứu trước đó là:

a) Đánh giá hiệu quả thuật toán về mặt toán học:

Theo hiểu biết của chúng tôi thì cho đến nay việc đánh giá các thuật toán nhân số lớn chỉ dựa vào tiêu chí *"Thuật toán được coi là hiệu quả hơn nếu có số phép nhân cơ bản ít hơn"*. Do vậy, việc phân rã, đánh giá bài toán thường chỉ tính đến phép nhân mà bỏ qua phép cộng (nguyên nhân do các vi xử lý trước đây có thời gian thực hiện phép nhân lớn hơn nhiều lần so với thời gian thực hiện phép cộng).

b) Về mặt công nghệ

+ Nhóm 1: Cải tiến thuật toán và lựa chọn tham số phù hợp, có hạn chế: Tham số tính nhanh thường có độ an toàn thấp hơn, thuật toán chưa phù hợp với thiết bị nguồn tài nguyên hạn chế.

+ Nhóm 2: Áp dụng các kỹ thuật lập trình đặc thù cho thiết bị nhúng (ngôn ngữ bậc thấp ASM, kỹ thuật song song trên nhiều CPU). Hạn chế: Các trình biên dịch mới đã tương đối tối ưu; các nghiên cứu còn rời rạc chưa tích hợp đầy đủ vào ECDH, ECDSA.

+ Nhóm 3: Giải pháp sử dụng phần cứng đặc thù trên thiết bị nhúng: co-processors: GPU; FPGA (Field Programmable Gate Arrays); DSP; NEON. Hạn chế: Thuật toán cứng nhắc; khó tùy biến, bản địa hóa thuật toán.

Định hướng nghiên cứu, phát triển của luận án:

Để nâng cao hiệu quả cho các thuật toán ECDH, ECDSA trên thiết bị nhúng, luận án định hướng nghiên cứu, đề xuất một số giải pháp để nâng cao hiệu suất, tốc độ tính toán gồm:

+ Đề xuất phương pháp đánh giá chi phí thuật toán chính xác hơn

+ Cải tiến, nâng cao hiệu quả thuật toán với tham số thay đổi được, và có thể hoạt động trên nền tảng ARM

+ Có thể sử dụng tùy biến, bản địa hóa thuật toán một cách dễ dàng (đặc biệt trong lĩnh vực an ninh quốc phòng); an toàn trong sử dụng

+ Các nguyên thủy được tích hợp đầy đủ vào trong giao thức ECDH, ECDSA

1.5.4 Các nền tảng phần cứng sử dụng trong luận án

Hầu hết các sản phẩm nhúng hiện nay đều sử dụng vi xử lý ARM phiên bản v7 (ARMv7) và v8 (ARMv8), do vậy để triển khai thực nghiệm các kết quả đề xuất trong luận án chúng tôi chọn hai Kit phát triển đại diện mà hỗ trợ hai dòng vi xử lý này. Cụ thể, chúng tôi chọn ZesBoard (ZYNQ 7000) để mô phỏng kết quả trên dòng vi xử lý ARMv7 và Kit NXP IMX8M để mô phỏng kết quả trên dòng vi xử lý ARMv8.

1.6. Kết luận chương 1

Chương 1 đã đi sâu phân tích các yêu cầu cần bảo mật dữ liệu trên nền thiết bị nhúng, các giải pháp nâng cao hiệu quả hệ mật khóa công khai Elliptic trên nền các thiết bị nhúng (cải tiến thuật toán, sử dụng các tham số đặc biệt; sử dụng các thành phần tính toán nhanh GPU, DSP, FPGA; sử dụng các đồng xử lý ...). Trong các giải pháp này, luận án cũng đã phân tích những ưu điểm và hạn chế của từng phương pháp. Trên cơ sở đó luận án sẽ tập trung vào giải pháp cải tiến thuật toán kết hợp với sử dụng đồng xử lý

(NEON) mà được tích hợp sẵn trên các vi xử lý từ ARMv7 trở đi.

CHƯƠNG 2. NÂNG CAO HIỆU QUẢ CỦA PHÉP NHÂN SỐ HỌC TRONG TRƯỜNG NHỊ PHÂN TRÊN VI XỬ LÝ ARM

Chương 2 của luận án trình bày về giải pháp để nâng cao hiệu quả của phép nhân số học trên vi xử lý ARM bao gồm: nghiên cứu mô hình, thuật toán nhân hiệu quả mới với chi phí thấp nhất và đề xuất các thuật toán cải tiến nhằm tăng hiệu năng xử lý dựa trên sự kết hợp của thuật toán đề xuất và sự hỗ trợ tính toán song song của thành phần NEON trên trường nhị phân. Các kết quả chính của chương 2 được trình bày trong ba công trình công bố [J1, C1, C2].

2.1. Phép nhân và phép cộng số học cơ bản trong trường hữu hạn

2.1.1. Phép nhân phổ thông

+ Nhân bằng phương pháp quét số hạng (Operand-Scanning)

+ Nhân bằng phương pháp quét tích (Product-Scanning)

2.1.2. Phép nhân số nguyên lớn trong trường nguyên tố

Trong cài đặt phần mềm, phép nhân trong trường nguyên tố được thực hiện trong các bộ nhân hỗ trợ sẵn trong vi xử lý (ALU) hoặc bộ bằng phần cứng được nhúng vào trong các vi xử lý nhúng

2.1.3 Phép nhân trong trường nhị phân

Trên phần mềm, phép nhân trong trường nhị phân sử dụng kết hợp giữa phép toán EOR (exclusive-or) và phép toán dịch bit. Do các vi xử lý không hỗ trợ các bộ nhân nhị phân chuyên dụng, nên

thực hiện phép toán nhân trên trường nhị phân sẽ chậm hơn thực hiện phép nhân trên trường nguyên tố.

2.1.4. Xác định tỷ số giữa phép nhân và phép cộng trên vi xử lý ARMv7/v8

Mục này sẽ xác định tỷ số giữa chỉ lệnh nhân và chỉ lệnh cộng trên vi xử lý ARM. Việc xác định được tỷ số giữa phép nhân và phép cộng giúp cho việc xây dựng được thuật toán nhân phân tầng với chi phí thấp nhất.

2.2. Nhân phân tầng trong trường hữu hạn

2.2.1. Số nguyên lớn và việc xử lý trên các số nguyên lớn

Thuật toán cộng và thuật toán trừ hai số *t* ký tự thực hiện theo Algorithm 2.5 và Algorithm 2.6 trang 30 trong [25]. Cả hai thuật toán trên đều cần đến *t* phép cộng có nhớ hoặc tương tự *t* phép trừ có mượn (hai phép toán trên có chi phí như nhau).

Ký hiệu *a là thời gian trung bình để thực hiện một phép cộng hai ký tự với nhau* và $A(t)$ là chi phí tính toán của phép cộng (trừ) hai số t ký tự theo hai thuật toán này thì

$$A(t) = ta. \tag{2.1}$$

2.2.2. Thuật toán nhân số lớn

Tính toán chi phí cả phép nhân và phép công cho hai thuật toán cơ bản:

a) Thuật toán nhân theo phương pháp phổ thông

$$M_S(t) = t^2 m + 2(t^2 - t)a. \tag{2.2}$$

b) Thuật toán nhân theo phương pháp của Karatsuba

$$\begin{aligned} M_K(t) &= t^2 m + 2(t^2 - t)a \\ &\quad + \frac{t(t-1)}{2}(-m + 4a) \\ &= \frac{t^2 - t}{2} m + 4(t^2 - t)a \end{aligned} \tag{2.3}$$

Bổ đề 1. *Ta có:*

1. *Với mọi* $t > 1$ *ta có* $M_S(t) > 4A(t), M_K(t) > 4A(t)$.
2. *Thuật toán nhân theo phương pháp Karatstuba chỉ hiệu quả hơn thuật toán nhân phổ thông khi và chỉ khi* $\frac{m}{a} > 4$.

2.2.3. Mô tả thuật toán phân tầng

Với cách biểu diễn một số $t = u \cdot v$ ký tự B-phân như một số gồm u ký tự B^v-phân, trong đó mỗi ký tự này lại là một số gồm v ký tự B-phân, ta có thể xây dựng phép nhân hai số nguyên t ký tự B-phân thông qua thuật toán nhân hai số có u ký tự B^v-phân và thuật toán nhân hai số có v ký tự B-phân. Chúng tôi gọi phương pháp này là phương pháp nhân "phân 2 tầng u: v", trong đó phần tính toán với các số có u ký tự B^v-phân được gọi là tầng cao, và phần tính toán còn lại được gọi là tầng thấp.

2.2.4. Một số tính chất về chi phí của thuật toán phân tầng

Thuật toán phân tầng chúng tôi trình bày ở mục trước có một số tính chất thể hiện qua mệnh đề sau.

Mệnh đề 1. *Cho* $t = u \cdot v$ *với* $u, v > 1$. *Khi đó ta có*

1. $M_S(u): M_S(v) = M_S(s): M_S(u) = M_S(t)$.
2. $M_K(u): M_K(v) = M_K(v): M_K(u) \leq M_K(t)$.
3. $M_K(u): M_S(v) - M_K(v): M_S(u) = \frac{t(u-v)}{2}(-m + 4a)$.
4. $M_S(u): M_K(v) - M_S(v): M_K(u) = \frac{t(u-v)}{2}(-m - 4a)$.
5. $M_S(u): M_S(v) - M_K(u): M_S(v) = \frac{t(u-v)}{2}m + t(u - 1)(v - 3)a \geq 0$.

2.2.5. Thuật toán nhân số nguyên lớn có chi phí thấp nhất

Với thuật toán phân tầng và tính chất của nó trình bày trong hai mục trước chúng tôi chứng minh một kết quả quan trọng sau.

Định lý 1. *Cho* $t = t_{s-1} \dots t_i \dots t_0$ với $t_i, 0 \le i \le s-1$ *là các số nguyên tố và* $t_{i-1} \ge t_i$. *Khi đó thuật toán nhân hai số nguyên t ký tự có chi phí thấp nhất là*

(i) $M_K[t_{s-1}]: \dots : M_K[t_0]$ *nếu* $m > 4a$.

(ii) $M_K[t_{s-1}]: \dots : M_K[t_1]: M_S[t_0]$ *trong trường hợp ngược lại.*

Tiếp sau đây chúng tôi đưa ra một kết quả bổ trợ để xác định chi phí một thuật toán nhân các số t ký tự với t có dạng $t = 2^r \times d$. Ký hiệu $(M_K[2])^r$ là thuật toán phân r tầng $M_K[2]: \dots : M_K[2]$.

Mệnh đề 2. *Cho* $d > 1$ *là một số nguyên. Khi đó thuật toán* $(M_K[2])^r : M[d]$ *có chi phí tính toán là*

$$(M_K[2]))^r : M[d]) = 3^r M(d) + 8(3^r - 2^r)A(d), \qquad (2.4)$$

trong đó M(d) là chi phí của thuật toán nhân hai số d ký tự M[d] và A(d) là chi phí cộng hai số d ký tự.

2.2.6. Tìm thuật toán nhân tối ưu cho một số giá trị t với giả thiết m = 2a.

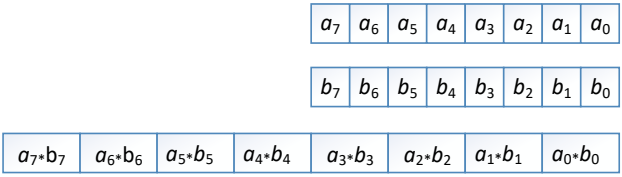
Trong mục này trước hết chúng tôi đưa ra thuật toán nhân tối ưu trong phạm vi các thuật toán được trình bày trong các mục trước và trên giả thiết **m** = 2**a** cho các số t ký tự 32-bít cho các giá trị t ≤ 16 và một số t riêng rẽ khác thường được sử dụng trong các ứng dụng mật mã.

2.3. Nhân phân tầng trong trường nhị phân trên vi xử lý ARMv7

2.3.1 Chỉ lệnh nhân nhị phân trong thành phần NEON trên vi xử lý ARMv7

Một sự tiện lợi rõ ràng của NEON SIMD cho mật mã đó là nó cung cấp nhiều không gian trong các thanh ghi, giảm số lần tải (load) và lưu (store) từ bộ nhớ. Thay vì 4 phép toán số học 32-bit

thì với NEON SIMD có thể thực hiện chỉ với 1 phép toán số học 128-bit.



Hình 2.1: Hoạt động của lệnh VMULL.P8.

2.3.2. Tham số của đường cong NIST trên các trường nhị phân Sử dụng đường cong elliptic của NIST trên $\mathbb{F}_{2^{283}}$

2.3.3. Một phương pháp mới để nhân nhanh đa thức trên vi xử lý ARMv7

Trong mục này, chúng tôi đề ứng dụng thuật toán nhân phân tầng đề xuất ở **mục 2.2** để nhân nhanh hai đa thức trên trường nhị phân GF(2^{283}) trên vi xử lý ARMv7. Phương pháp đề xuất mới ở đây (MNKv7) là kết hợp giữa thuật toán Karatsuba với bộ nhân cơ bản 64bit và 32bit trên nền vi xử lý ARMv7 để tạo thành thuật toán nhân đầy đủ trên hai trường $\mathrm{GF}(2^{283})$.

2.3.4 Thực nghiệm, đánh giá thuật toán đề xuất

Chúng tôi sử dụng ngôn ngữ Assembly/C và thư viện mật mã GMP để cài đặt thuật toán nhân 32bit và 64bit đã đề xuất. Sau đó kết hợp giữa thuật toán nhân 64bit, thuật toán 32 bit cơ bản với thuật toán Karasuba (cũng được cài đặt bằng ngôn ngữ Assembly) để thực hiện phép nhân hai đa thức đầy đủ trên $GF(2^{283})$ – thuật toán đầy đủ này gọi là MNKv7. Để đánh giá được tốc độ thực hiện của thuật toán cải tiến, chúng tôi có so sánh sự thực thi giữa thuật toán cải tiến với sự thực thi tương ứng khi sử dụng các hàm mặc định của thư viện GMP [61] và thư viện OpenSSL [55].

Bảng 2.1. Kết quả đánh giá trên Xilinx Zynq-7000

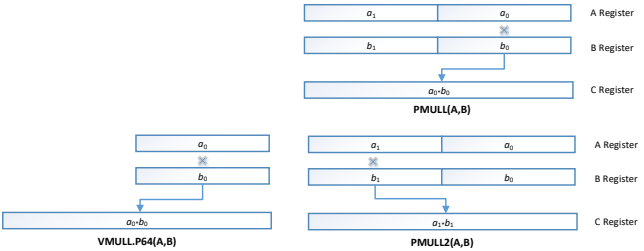
Thuật toán	**Thời gian thực hiện (nano giây)**		
	OpenSSL-1.1.1	**GMP-5.1.3**	**MNKv7**
Ký ECDSA (NIST-B283)	34771922	4521351	973160
Kiểm tra chữ ký ECDSA (NIST-B283)	67183606	21786237	2517141
Ký ECDSA (NIST-K283)	31328528	4494423	913152
Kiểm tra chữ ký ECDSA (NIST-K283)	60183606	10760133	1273322

2.4. Nhân phân tầng trong trường nhị phân trên vi xử lý ARMv8

Trong mục này, chúng tôi đề xuất thuật toán nhân phân tầng trên trường nhị phân GF(2^{283}), GF(2^{409}), GF(2^{571}) dựa trên thuật toán nhân phân tầng đề xuất ở **mục 2.2** ở trên, bằng việc kết hợp giữa ba thành phần chính: thuật toán Karatsuba, bộ nhân 128-bit và bộ nhân 192-bit trên nền vi xử lý ARMv8 64-bit (AArch64) và 32-bit (AArch32).

2.4.1 Hỗ trợ nhân đa thức nhị phân trên vi xử lý ARMv8

AArch64 cung cấp hai chỉ lệnh là PMULL và PMULL2, cả hai chỉ lệnh đều thực hiện phép nhân 64x64-bit. Hoạt động của PMULL và PMULL2 được minh họa trong hình 2.2.



Hình 2.2: Hoạt động của VMULL.P64, PMULL và PMULL2 trên vi xử lý ARMv8

2.4.2. Xây dựng thuật toán nhân đa thức nhị phân phân tầng trên vi xử lý ARMv8

Trong nội dung này, chúng tôi đề xuất thuật toán nhân phân tầng trên trường nhị phân GF(2^{283}), GF(2^{409}), GF(2^{571}) bằng việc kết hợp giữa ba thành phần chính (gọi là MKNv8): Thuật toán Karatsuba, bộ nhân 128-bit và bộ nhân 192-bit trên nền vi xử lý ARMv8 32-bit (AArch32) và 64-bit (AArch64).

Giới hạn cho các thuật toán nhân nhị phân ở đây là hai thuật toán, một theo phương pháp nhân phổ thông (ký hiệu là $M_S[t_i]$) và một theo phương pháp Karatsuba ($M_K[t_i]$) cho tầng thứ t_i. Phân tích cụ thể cho các trường cần khảo sát như sau:

- Đối với GF(2^{283}) thì mỗi đa thức bậc ≤ 282 được tách thành 5 đa thức bậc 63. Khi này phân tầng tốt nhất có thể được xét đến là 2×3 ký hiệu là $M_K[2]: M_S[3]$ hoặc 3x2 ký hiệu là $M_K[3]: M_S[2]$
- Đối với GF(2^{409}) thì mỗi đa thức bậc ≤ 408 được tách thành 7 đa thức bậc 63. Khi này phân tầng tốt nhất là 2×2×2, ký hiệu là $M_K[2]: M_K[2]: M_S[2]$
- Đối với GF(2^{571}) thì mỗi đa thức bậc ≤ 570 được tách thành 9 đa thức bậc 63. Khi này phân tầng tốt nhất là 3×3, ký hiệu là $M_K[3]: M_S[3]$.

Như vậy, để thực hiện phép nhân phân tầng trong trường nhị phân GF(2^{283}), GF(2^{409}) và GF(2^{571}) thì ta cần xây dựng 2 bộ nhân là: bộ nhân 128-bit và bộ nhân 192-bit bằng phương pháp nhân phổ thông.

2.4.3. Thực nghiệm và đánh giá thuật toán đề xuất

Chúng tôi sử dụng thư viện RELIC [54] để tích hợp cài đặt thuật toán đề xuất (MKNv8) mô tả ở trong mục trên. Tiếp theo

để đánh giá tính hiệu quả của MKNv8, chúng tôi sử dụng thuật toán mật mã là ECDSA, ECDH và ECMQV. Các kết quả lần lượt được thể hiện trong bảng dưới đây.

Bảng 2.2. Kết quả đánh giá ECDSA trên NXP IMX8M Kit

Algorithm	**LOHAD [29] (nanosec)**	**MKNv8 (nanosec)**
Sign ECDSA (NIST-B283)	2157937	504856
Verify signature ECDSA (NIST-B283)	10463561	2426928
Sign ECDSA (NIST-B409)	4740125	1042230
Verify signature ECDSA (NIST-B409)	22585278	4920991
Sign ECDSA (NIST-B571)	9947431	2051932
Verify signature ECDSA (NIST-B571)	46434713	9791679

2.5. Kết luận chương 2

Các kết quả chính của chương 2 được trình bày trong ba công trình công bố [J1, C1, C2] với hai kết quả chính là:

- Đề xuất thuật toán nhân phân tầng với chi phí thấp nhất và phương pháp đánh giá hiệu quả thuật toán nhân này.
- Áp dụng thuật toán nhân phân tầng kết hợp với bộ nhân trên trong thành phần NEON để nâng cao hiệu quả của phép nhân số học trong trường nhị phân trên vi xử lý ARMv7 và ARMv8.

Kết quả thực nghiệm cho thấy thuật toán đề xuất nhanh hơn các kỹ thuật trước đó lần lượt là 4, 5, 5 lần trong trường GF(2^{283}), GF(2^{409}), GF(2^{571}) khi thực hiện trên cùng một nền tảng phần cứng ARMv7 hoặc ARMv8.

CHƯƠNG 3. NÂNG CAO HIỆU QUẢ PHÉP NHÂN VÔ HƯỚNG CỦA HỆ MẬT ECC TRONG TRƯỜNG NGUYÊN TỐ TRÊN VI XỬ LÝ ARM

Trong chương 3, luận án sẽ đề xuất một số giải pháp nhằm cải tiến và nâng cao hiệu quả thực hiện phép nhân điểm trên đường cong ECC trong trường nguyên tố dựa trên kết hợp giữa cải tiến NAF và cải tiến phép cộng và nhân đôi điểm nhờ sử dụng phép nhân kép. Các kết quả chính của chương 3 được trình bày trong hai công trình công bố [J2, J3].

3.1. Cơ sở để nâng cao hiệu quả của phép nhân điểm vô hướng trên ECC trong trường nguyên tố.

3.1.1 Nâng cao hiệu quả của phép tính số học trong trường nguyên tố trên vi xử lý nhúng

Trong trường nguyên tố, việc thay đổi thuật toán số học dựa trên khai thác các tính năng mới (ví dụ như ứng dụng NEON trong vi xử lý ARM Cortex-A) sẽ gặp phải khó khăn trong vấn đề ***xử lý lan truyền phần dư***.

3.1.2 Nâng cao hiệu quả của các phép toán số học điểm

Phép toán cơ bản của ECC là phép nhân điểm vô hướng kP, trong đó k là một số nguyên, P là một điểm trên đường cong elliptic. Để thực hiện phép nhân điểm vô hướng, ta có thể sử dụng các phép cộng điểm (point addition) và nhân đôi điểm (point doubling multiplication).

Trong chương 3 của luận án, sẽ kết hợp cả 3 phương pháp này để nâng cao hiệu quả của phép tính số học điểm cụ thể:

a) Trong mục 3.3 sẽ trình bày về phương pháp biểu diễn eNAF để giảm số phép cộng và nhân đôi điểm

b) Trong mục 3.4 sẽ kết hợp cả phương pháp 2 và 3 đó là nâng cao hiệu quả của phép toán nhân số học (phép nhân kép) và đề xuất tích hợp triển khai cho phép cộng điểm và nhân đôi điểm.

3.2. Một số thuật toán nhân điểm vô hướng trên đường cong Elliptic.

Mục này trình bày một số thuật toán cơ bản đển nhân một số nguyên với một điểm gồm: 1) Thuật toán nhị phân Right – to – Left; 2) Thuật toán NAF (non-adjacent form); 3) Thuật toán NAF cửa sổ trượt cho tính phép nhân vô hướng trên đường cong Elliptic

3.3. Mở rộng cho dạng biểu diễn NAF của số nguyên dương

Trong mục này, chúng tôi đề xuất một thuật toán cho việc tìm dạng biểu diễn không liền kề NAF của một số nguyên k cho trước. Thuật toán này tiết kiệm được ba phép tính số học so với thuật toán nguyên thủy. Tiếp theo, với dạng biểu diễn hầu không liền kề aNAF mới của số k, chúng ta đã rút gọn một phép tính bình phương khi thực hiện phép "bình phương – nhân" đối với phép tính lũy thừa k trên các trường hữu hạn, hoặc tương ứng một phép tính gấp đôi khi thực hiện phép "gấp đôi - cộng" trong phép nhân điểm trên đường cong elliptic.

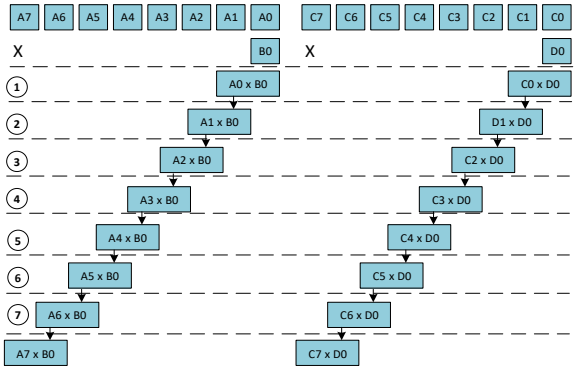
3.4. Nâng cao hiệu quả của phép toán số học điểm trên hệ mật ECC trong trường nguyên tố

Trong phần này, sẽ trình bày một phương pháp nâng cao hiệu quả các phép toán số học trên đường cong elliptic. Phương pháp này tập trung khai thác tối đa đặc điểm phần cứng SIMD trên nền vi xử lý ARM có tích hợp thành phần NEON nhằm tăng tốc độ các phép toán số học. Các đặc trưng chính của phương pháp gồm:

1) kết hợp phép nhân thông thường (phương pháp quét số hạng) với thuật toán nhân Karatsuba cho các số hạng dài; 2) thực hiện nhân song song (nhân kép) trên trường số nguyên tố kết hợp với ghép cặp để giảm thiểu chi phí đọc, ghi dữ liệu giữa bộ nhớ và thành phần NEON.

3.4.2. Đề xuất thuật toán song song hai phép nhân trên trường GF(p)

Đối với kiến trúc SIMD, do đặc điểm hỗ trợ tính hai phép nhân 32-bit sử dụng một chỉ lệnh đơn, chúng tôi ứng dụng phép toán này trong thuật toán minh họa như hình sau.



Hình 3.1: Thực hiện nhân song song hai phép nhân

3.4.3. Cải tiến thuật toán số học trên đường cong Elliptic

Như đã trình bày ở các mục trên, bất cứ ở đâu xuất hiện cặp phép nhân trên F_p mà dữ liệu của chúng không phụ thuộc vào nhau thì ta có thể sử dụng các thuật toán nhân kép đã trình bày ở trên để thực hiện nhân song song cặp phép nhân này. Nguyên lý này cũng có thể áp dụng cho phép bình phương.

Ta ký kiệu M, S là chi phép của phép nhân và phép bình phương thông thường; Mt, St là chi phí của phép nhân kép (thực hiện hai phép nhân song song) theo mô tả ở trong mục 3.4.2.

Bảng 3.1. So sánh chi phí của thuật toán đề xuất

Thuật toán	Phép toán trên điểm	
	Cộng điểm	Nhân đôi điểm
Thuật toán tuần tự [56]	11M+5S	3M + 5S
Thuật toán song song theo đề xuất	5Mt + 2Mt + 1M +1S	4Mt

3.5. Thử nghiệm và đánh giá

3.5.1 Mô tả phương thực nghiệm, kịch bản

Sau khi cài đặt thuật toán đề xuất, chúng tôi tiến hành đánh hiệu quả của thuật toán so với thuật toán chưa cải tiến (thuật toán mặc định trong thư viện RELIC). Các so sánh được lần lượt chạy trên hai nền vi xử lý ARMv7 và ARMv8.

3.5.2 Thử nghiệm trên ARMv7

Các kết quả thử nghiệm sau đây được thực hiện trên nền tảng phần cứng: Xilinx Zynq Kit [49] với vi xử lý ARMv7 có tốc độ 1.3 GHz, chạy hệ điều hành Linux nhúng.

Bảng 3.2. Thời gian thực hiện của tính toán trong giao thức ECDSA và ECDH trên ARMv7

Độ dài bit	Giao thức dựa trên $E(F_p)$	Thời gian (10^6 nano giây)		Tỷ số (nhân kép / nhân tuần tự)
		Nhân kép	Nhân tuần tự	
256	ECDH	9.0	11.2	0.8
	ECMQV	12.1	14.2	0.85
	ECDSA Sig	4.9	5.3	0.92
	ECDSA Ver	12.6	14.7	0.86
384	ECDH	23.4	29.3	0.80

	ECMQV	31.2	36.7	0.85
	ECDSA Sig	12.6	13.6	0.93
	ECDSA Ver	32.5	37.3	0.87
521	ECDH	69.4	86.9	0.80
	ECDSA Sig	36.1	39.3	0.92
	ECDSA Ver	74.8	85.0	0.88

3.5.3 Thử nghiệm trên ARMv8

Các kết quả thử nghiệm sau đây được thực hiện trên nền tảng phần cứng: NXP IMX8M Kit [47] (vi xử lý ARMv8 tốc độ 1.5 GHz) chạy hệ điều hành Linux nhúng.

Trên nền ARMv8, các tính hợp thuật toán đề xuất cho giao thức ECDH và ECDSA tăng khoảng từ 10% đến 20% so với nguyên thủy.

Bảng 3.3. Thời gian thực hiện của tính toán trong giao thức ECDSA và ECDH trên ARMv8

Độ dài bit	Giao thức dựa trên $E(F_p)$	Thời gian (10^6 nano giây)		Tỷ số (nhân kép / nhân tuần tự)
		Nhân kép	Nhân tuần tự	
256	ECDH	3.4	4.2	0.81
	ECMQV	4.6	5.3	0.86
	ECDSA Sig	1.8	2.0	0.90
	ECDSA Ver	4.7	5.5	0.85
384	ECDH	9.7	12.6	0.77
	ECMQV	12.8	15.8	0.81
	ECDSA Sig	5.1	5.8	0.88
	ECDSA Ver	13.0	16.0	0.81
521	ECDH	28.6	34.8	0.82
	ECMQV	37.8	43.1	0.88
	ECDSA Sig	14.7	16.0	0.92
	ECDSA Ver	37.1	42.9	0.86

3.6. Kết luận chương 3

Chương 3 của luận án tập trung cải tiến tốc độ cho các thuật toán mật mã dựa trên đường cong Elliptic trên trường số nguyên tố dựa trên đề xuất cải tiến thuật toán NAF và khai thác đặc điểm phần cứng trên nền vi xử lý ARM (cả ARMv7 và ARMv8) để nâng cao hiệu quả của các phép toán số học (cộng, nhân đôi và nhân vô hướng) trên đường cong Elliptic trong trường số nguyên tố. Các kết quả chính của chương 3 được trình bày trong hai công trình công bố [J2, J3]

Dựa trên đặc điểm SIMD trên vi xử lý ARM, luận án đã đề xuất cải tiến thuật toán cộng điểm và nhân đôi điểm nhờ phương pháp nhóm theo từng cặp phép nhân, theo cặp phép bình phương hoặc kết hợp cặp của phép nhân và phép bình phương. Các kết quả thực nghiệm đã chứng tỏ hiệu quả của thuật toán đề xuất như sau:

- Tăng khoảng từ 20% đến 30% đối với phép toán cơ bản (cộng, nhân đôi, nhân vô hướng) so với thuật toán mặc định trong thư viện RELIC
- Tăng từ 10% đến 20% trong các tính toán của giao thức ECDH và ECDSA so với các tính toán mặc định trong thư viện RELIC

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

A-Hai đóng góp khoa học của luận án

1) Đề xuất phương pháp nhân phân tầng hai số hạng trên trường hữu hạn dựa trên hai thuật toán nhân cơ bản là thuật toán nhân theo phương pháp phổ thông và thuật toán nhân theo phương pháp Karatsuba. Với phương pháp phân tầng, luận án đã xây dựng được thuật toán nhân có chi phí tốt nhất trong các trường hợp cụ thể cũng như đưa ra công thức để xác định chi phí

của thuật toán đó. Thuật toán đề xuất đã được kiểm chứng về tính hiệu quả trên nền vi xử lý nhúng ARMv7 và ARMv8.

2) Đề xuất, cải tiến thuật toán nhân vô hướng (nhân giữa một điểm và một số nguyên dương) của hệ mật đường cong Elliptic trên trường nguyên tố dựa trên đề xuất cải tiến thuật toán NAF và đề xuất nâng cao hiệu quả của các phép toán số học (cộng điểm, nhân đôi điểm) theo phương pháp song song hai phép nhân**.**

B-Ưu nhược điểm của các đề xuất trong luận án và hướng phát triển tiếp theo của đề tài

Ưu điểm: Các đề xuất nâng cao hiệu quả về sử dụng mật mã đường cong elliptic trên thiết bị nhúng trong luận án sẽ góp phần tăng cường khả năng bảo mật thông tin trên các thiết bị nhúng. Đặc biệt có ý nghĩa trong lĩnh vực an ninh quốc phòng: Khả năng mềm dẻo ở tùy biến tham số, bản địa hóa thuật toán, khả năng đáp ứng thời gian thực đối với các phần mềm bảo mật.

Nhược điểm và hạn chế: Thuật toán nhân phân tầng được xây dựng trên trên cơ sở hai thuật toán cơ bản là thuật toán theo phương pháp phổ thông và thuật toán nhân theo phương pháp Karatsuba. Tuy nhiên, cần có những nghiên cứu đề xuất thuật toán phân tầng dựa trên các thuật toán cơ sở khác cũng như cho phép toán khác như: modulo, bình phương..

Đề xuất hướng nghiên cứu tiếp theo: Nghiên cứu đề xuất thuật toán phân tầng dựa trên các thuật toán cơ sở khác cũng như cho phép toán khác trong trường hữu hạn: modulo, bình phương...; Nâng cao hiệu quả các các lược đồ, thuật toán mật mã dựa trên ECC cho các hệ vi xử lý nhúng khác như dòng ARM Cortex-M; Nghiên cứu đảm bảo an toàn chống lại tấn công kênh kề cho các thuật toán mật mã của hệ mật ECC hoạt động trên vi xử lý ARM

DANH MỤC CÔNG TRÌNH KHOA HỌC ĐÃ CÔNG BỐ

[**J1**] Phạm Văn Lực, Võ Tùng Linh, "*Phép nhân số nguyên lớn*", Tạp chí Nghiên cứu KH&CN quân sự, Số 63, 2019, pp.111-120.

[**C1**] Pham Van Luc, Vo Tung Linh, Hoang Dang Hai, Leu Duc Tan, "*Fast Binary Field Mutiplication on ARMv7 Embedded Processors*", in Proc. of 4th IEEE Int. Conf. on Recent Advances in Signal Processing, Telecommunications & Computing (SigTelCom 2020), Hanoi, Vietnam, Aug. 2020.

[**C2]** Pham Van Luc, Hoang Dang Hai, Leu Duc Tan, "*Multi-layer Multiplication in Binary Field on ARMv8 Processors*", in Proc. of IEEE Int. Conf. on Advanced Technologies for Communications (ATC 2020), Nhatrang, Vietnam, Oct. 2020.

[**J2**] Pham Van Luc, Leu Duc Tan, "*Extensions for NAF representation of positive integers*", Journal of Science in Information Technology and Communications, vol. CS.01, No.1, 2021, pp.62-66.

[**J3**] Pham Van Luc, Hoang Dang Hai, Leu Duc Tan, "*Improving the Efficiency of Point Arithmetic on Elliptic Curves using ARM Processors and NEON*", ISI Q2, International Journal of Network Security, Vol. 24, No. 2, 2022, pp. 364-376.