

# **Cryptographic Hash Functions**

**Hàm băm mật mã**

# Nội dung

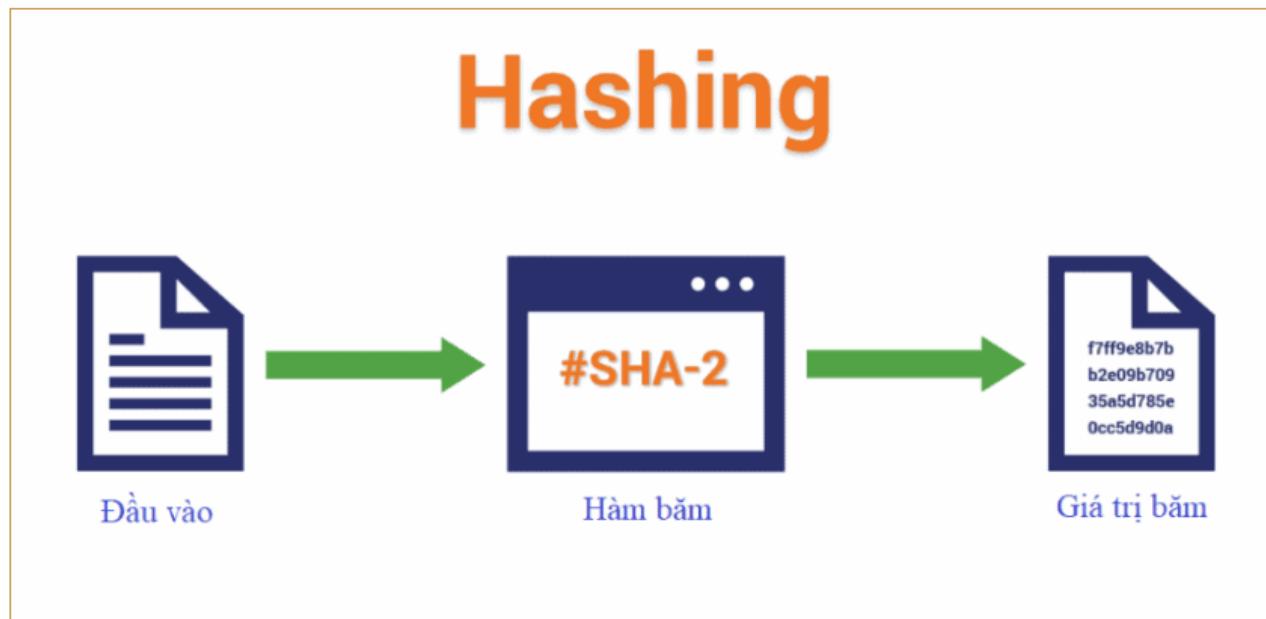
- Giới thiệu những ý tưởng chung đằng sau các hàm băm mật mã
- Thảo luận về lược đồ Merkle-Damgard làm cơ sở cho các hàm băm được lắp lại
- Phân biệt giữa hai loại hàm băm:
  - Cấu trúc của SHA-512.
  - Cấu trúc của Whirlpool.



# 1 GIỚI THIỆU

## Hash là gì?

- ❖ Về cơ bản hashing là quá trình biến một dữ liệu đầu vào có độ dài bất kỳ thành một chuỗi đầu ra đặc trưng có độ dài cố định. Hashing được thực hiện thông qua hàm băm (hash function).
- ❖ Một cách tổng quát hàm băm là bất kỳ hàm nào có thể được sử dụng để ánh xạ dữ liệu có kích thước tùy ý thành các giá trị kích thước cố định. Các giá trị được trả về bởi hàm băm được gọi là giá trị băm, mã băm, thông điệp băm, hoặc đơn giản là “hash”.



# 1 GIỚI THIỆU

- ❖ Ví dụ, khi bạn download một video trên YouTube có dung lượng **50 MB** và thực hiện hashing trên nó bằng **thuật toán băm SHA-256**, thì đầu ra bạn thu được sẽ là một giá trị băm có độ dài **256 bit**. Tương tự, nếu bạn lấy một tin nhắn văn bản có dung lượng **5 KB**, để hashing bằng **SHA-256** thì giá trị băm đầu ra bạn thu được vẫn sẽ là **256 bit**.
- ❖ Trong blockchain, các giao dịch có độ dài khác nhau sẽ được băm thông qua một thuật toán băm nhất định và tất cả đều cho đầu ra có độ dài cố định bất kể độ dài của giao dịch đầu vào là bao nhiêu. Chẳng hạn, Bitcoin sử dụng thuật toán **SHA-256** để băm các giao dịch cho kết quả đầu ra có độ dài cố định là **256 bit (32 byte)** cho dù giao dịch chỉ là một từ hoặc giao dịch phức tạp với lượng dữ liệu khổng lồ.
- ❖ Kỹ thuật hashing thường được sử dụng và có ứng dụng rộng rãi nhất trong việc đảm bảo tính toàn vẹn cho dữ liệu trong blockchain là các hàm băm mật mã (*cryptographic hash function*) chẳng hạn như **SHA-1. SHA-2. SHA-3, SHA-256...** Sỡ dĩ như vậy là do các hàm băm mật mã có một số tính chất quan trọng phù hợp cho việc đảm bảo an toàn dữ liệu.

# 1 GIỚI THIỆU

*Một hàm băm mật mã nhận một thông điệp có độ dài tùy ý và tạo ra một thông báo có độ dài cố định. Mục tiêu của chương này là thảo luận chi tiết về hai thuật toán băm mật mã hứa hẹn nhất – **SHA-512** and **Whirlpool***

**Chủ đề trong phần này gồm:**

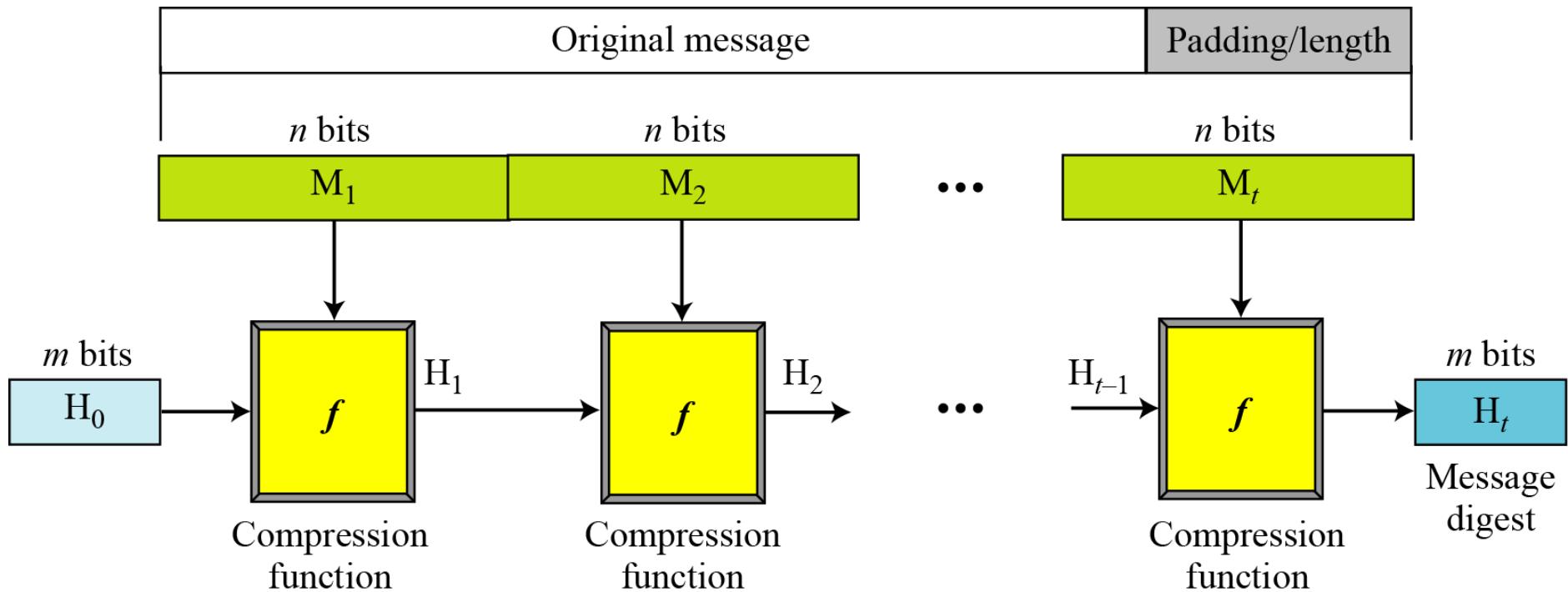
- 1.1 Hàm băm được lặp lại
- 1.2 Hai nhóm chức năng nén

# 1.1 Hàm băm được lắp lại

## Iterated Hash Function

### Lược đồ Merkle-Damgard

Figure 1 Lược đồ Merkle-Damgard



$H_0$ : là giá trị / vec tơ khởi tạo

$H_i = f(H_{i-1}, M_i)$ , trong đó  $f$  là hàm nén.

$H_t$  là hàm băm mã hóa của bản tin gốc, tức là  $h(M)$ .

## *1.2 Hai nhóm hàm nén*

### *Two Groups of Compression Functions*

*1. Chức năng nén được thực hiện từ đầu.*

*Tóm lược thông điệp  
Message Digest (MD)*

*2. Mật mã khôi khóa đối xứng đóng vai trò như một hàm nén.*

*Xoáy nước  
Whirlpool*

## 1.2 Hai nhóm hàm nén

### Two Groups of Compression Functions

1. *Chức năng nén được thực hiện từ đầu.*

**Tóm lược thông điệp Message Digest  
(MD)**

Có nhiều phiên bản MD như MD2, MD4, **MD5** được thiết kế bởi Ron Rivest. MD5 chia bản tin thành các khối 512 bits để tạo thành một digest 128 bit.

SHA (*secure hash algorithm*): *chuẩn băm bảo mật* được NIST công bố theo FIP 180. SHA dựa trên cấu trúc MD5 bao gồm có SHA-1, SHA-224, SHA-256, SHA-384, và SHA-512.

## 1.2 *Continued*

**Table 12.1** *Characteristics of Secure Hash Algorithms (SHAs)*

<i>Characteristics</i>	<i>SHA-1</i>	<i>SHA-224</i>	<i>SHA-256</i>	<i>SHA-384</i>	<i>SHA-512</i>
Maximum Message size	$2^{64} - 1$	$2^{64} - 1$	$2^{64} - 1$	$2^{128} - 1$	$2^{128} - 1$
Block size	512	512	512	1024	1024
Message digest size	160	224	256	384	512
Number of rounds	80	64	64	80	80
Word size	32	32	32	64	64

**Table 12.8 A Comparison of MD5, SHA-1, and RIPEMD-160**

	<b>MD5</b>	<b>SHA-1</b>	<b>RIPEMD-160</b>
Digest length	128 bits	160 bits	160 bits
Basic unit of processing	512 bits	512 bits	512 bits
Number of steps	64 (4 rounds of 16)	80 (4 rounds of 20)	160 (5 paired rounds of 16)
Maximum message size	$\infty$	$2^{64} - 1$ bits	$2^{64} - 1$ bits
Primitive logical functions	4	4	5
Additive constants used	64	4	9
Endianness	Little-endian	Big-endian	Little-endian

**Table 12.9 Relative Performance of Several Hash Functions  
(coded in C++ on a 850 MHz Celeron)**

Algorithm	MBps
MD5	26
SHA-1	48
RIPEMD-160	31

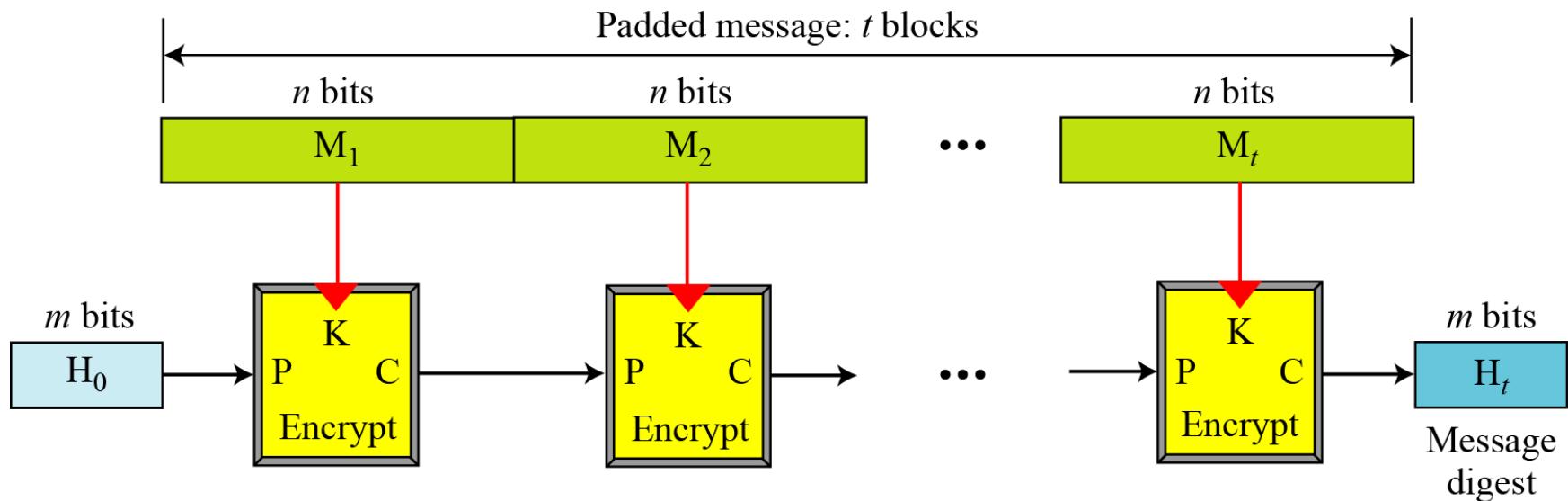
Note: Coded by Wei Dai; results are posted at <http://www.eskimo.com/~weidai/benchmarks.html>

## 1.2 Continued

### Rabin Scheme: Lược đồ Rabin

Lược đồ đơn giản dựa trên Merkle-Damgard: Hàm nén được thay thế bởi một hệ mật mã nào đó, các khối bản tin sử dụng như là Khóa (K), bản tin digest trước mỗi hàm là bản tin gốc (P) của nó. Bản tin C là bản tin digest mới tạo ra bởi hàm đó. Chú ý: kích thước của digest bằng kích thước của mã khối dữ liệu. Ví dụ trong hệ mật DES thì nó (digest) chỉ bằng 64 bit.

Figure 2 Rabin scheme

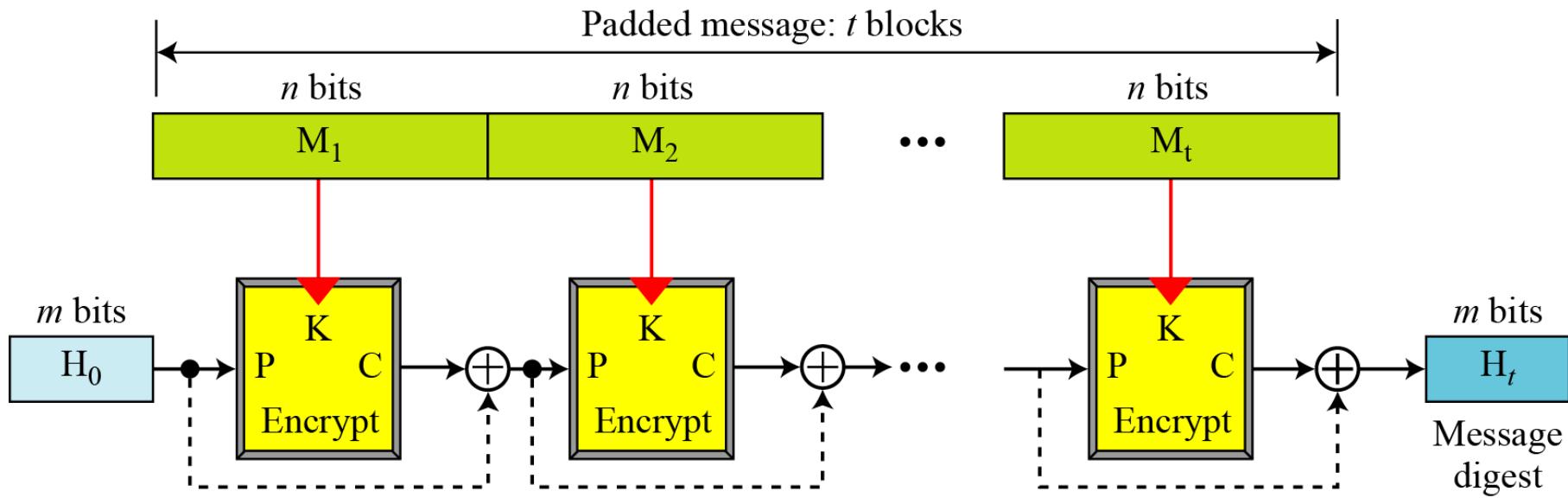


# 1.2 Continued

## Lược đồ Davies-Meyer:

Lược đồ này dựa trên lược đồ Rabin, có thêm phần *XOR* dữ liệu giữa  $C$  và  $P$  sau mỗi hàm băm  $\rightarrow$  nhằm tăng độ bảo mật cho tấn công vào quá trình trung gian (tấn công **meet-in-the-middle**)

Figure 3 Lược đồ Davies-Meyer

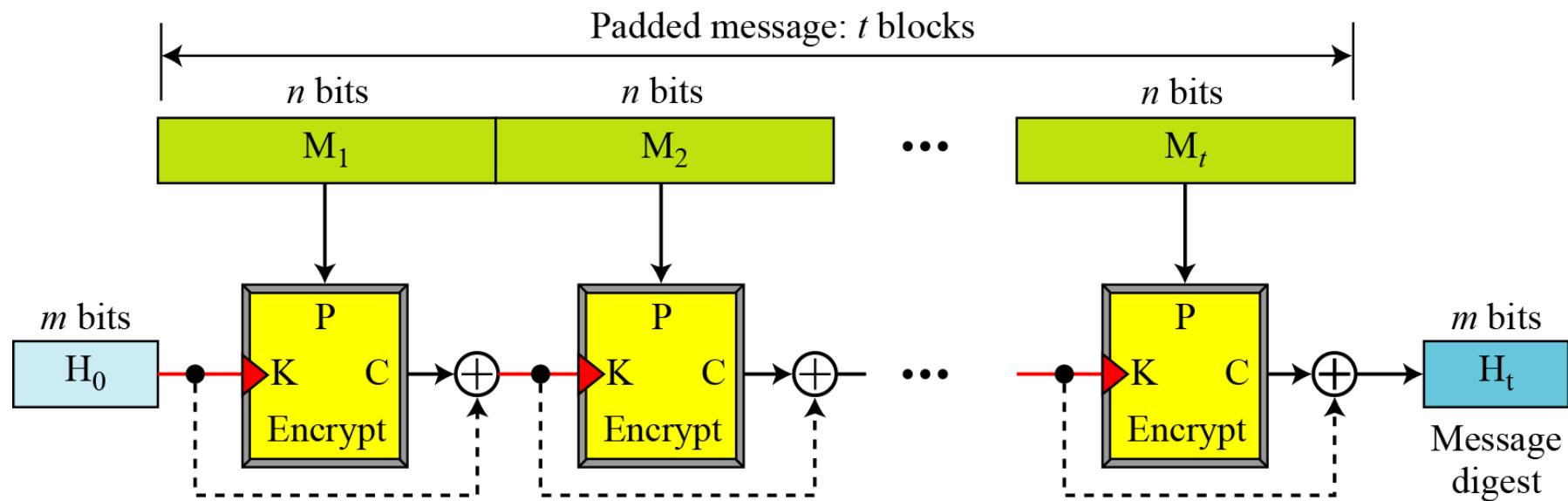


## 1.2 Continued

### Lược đồ Matyas-Meyer-Oseas

Đây là phiên bản kép của Davis-Meyer; các khối bản tin sử dụng cho khóa mật. Lược đồ này sử dụng tốt nhất khi khóa mật và khối bản tin cùng kích thước. Khi đó hệ AES được sử dụng là tốt nhất.

Figure 4 Lược đồ Matyas-Meyer-Oseas scheme

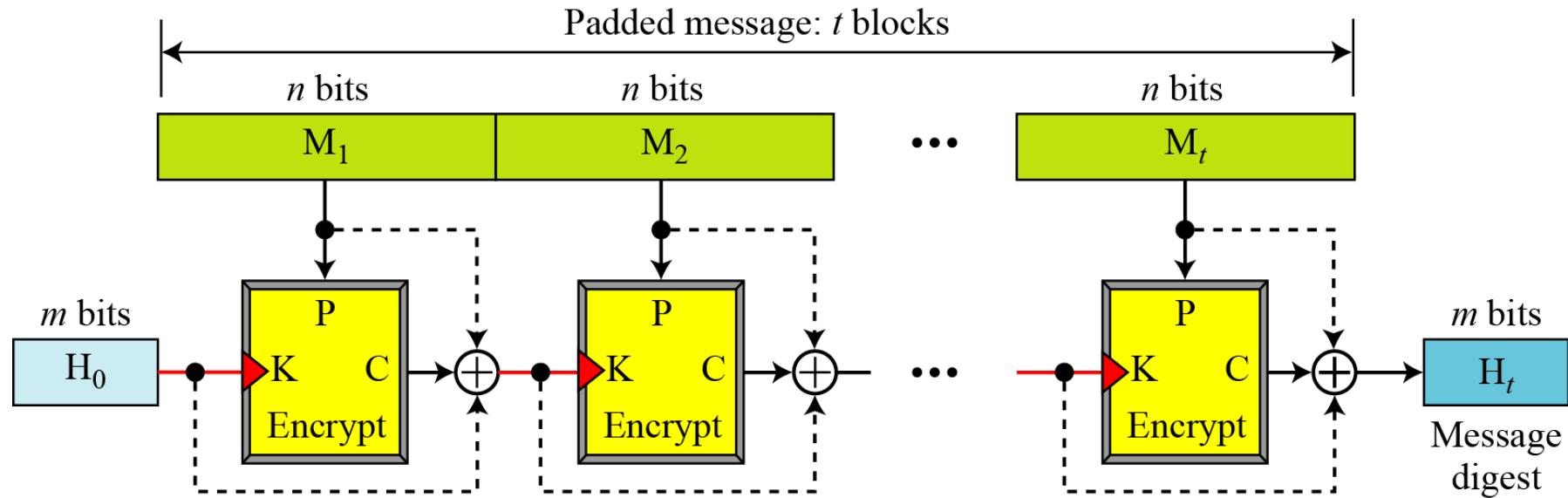


## 1.2 Continued

### Lược đồ Miyaguchi-Preneel:

Là phiên bản mở rộng của lược đồ Matyas-Meyer-Oseas; ở đây phép XOR bởi 3 luồng bản tin  $\{M_t, C, H_{t-1}\}$  tạo ra bản tin digest mới ( $H_t$ )  $\rightarrow$  tăng cường tính bảo mật cho tấn công trung gian.

Figure 5 Lược đồ Miyaguchi-Preneel



## 2 SHA-512

*SHA-512 is the version of SHA with a 512-bit message digest. This version, like the others in the SHA family of algorithms, is based on the Merkle-Damgard scheme.*

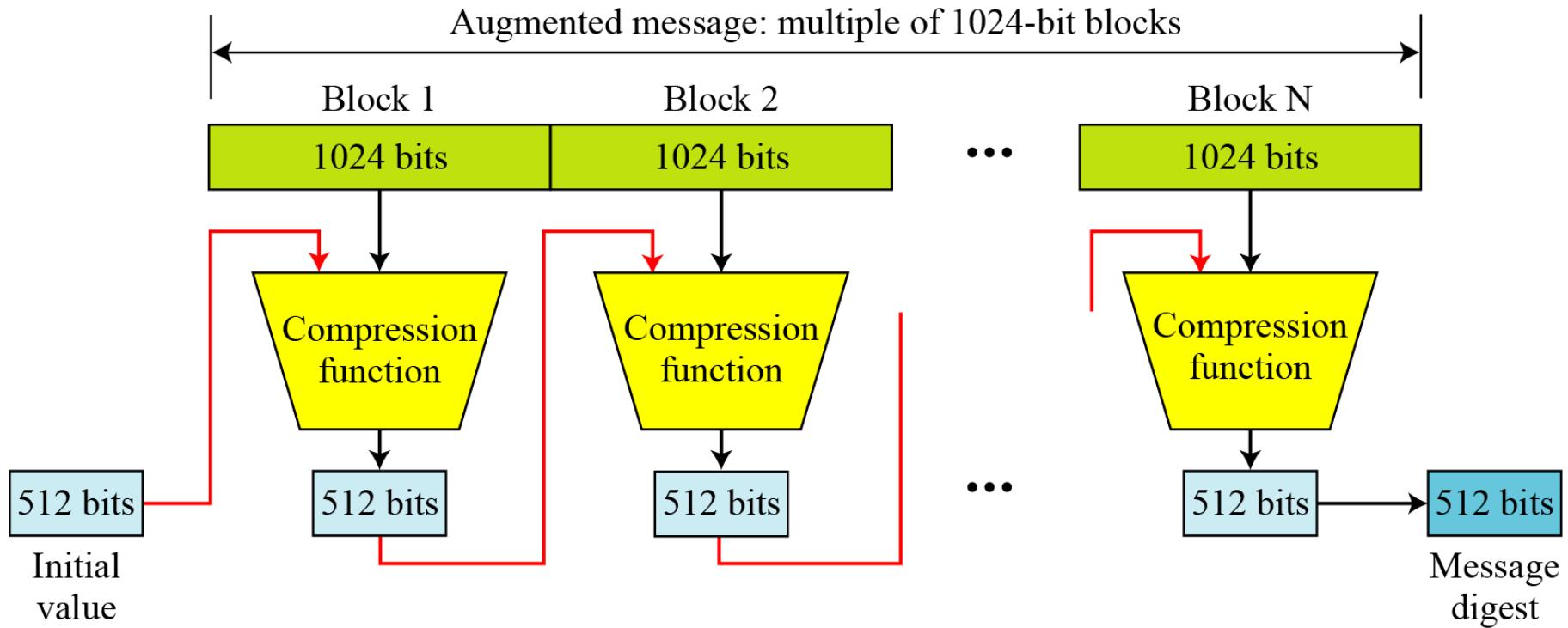
*SHA-512 là phiên bản của SHA với bản tóm tắt thông báo 512 bit. Phiên bản này, giống như các phiên bản khác trong họ thuật toán SHA, dựa trên lược đồ Merkle-Damgard.*

### *Topics discussed in this section:*

- 2.1      Introduction**
- 2.2      Compression Function**
- 2.3      Analysis**

## 2.1 Introduction

**Figure 6 Message digest creation SHA-512**



## 2.1 Continued

### *Message Preparation*

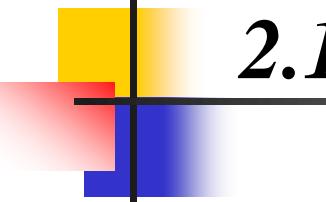
*SHA-512 insists that the length of the original message be less than  $2^{128}$  bits.*

*SHA-512 khẳng định rằng độ dài của bản tin gốc phải nhỏ hơn  $2^{128}$  bit.*

#### **Note**

**SHA-512 creates a 512-bit message digest out of a message less than  $2^{128}$ .**

*SHA-512 tạo bản tóm tắt tin nhắn 512 bit từ một bản tin nhỏ hơn  $2^{128}$ .*



## 2.1 *Continued*

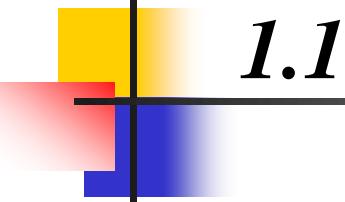
### Example 1

This example shows that the message length limitation of SHA-512 is not a serious problem. Suppose we need to send a message that is  $2^{128}$  bits in length. How long does it take for a communications network with a data rate of  $2^{64}$  bits per second to send this message?

Ví dụ này cho thấy rằng giới hạn độ dài tin nhắn của SHA-512 không phải là một vấn đề nghiêm trọng. Giả sử chúng ta cần gửi một tin nhắn có độ dài  $2^{128}$  bit. Mất bao lâu để một mạng truyền thông với tốc độ dữ liệu  $2^{64}$  bit / giây gửi thông điệp này?

### Solution

A communications network that can send  $2^{64}$  bits per second is not yet available. Even if it were, it would take many years to send this message. This tells us that we do not need to worry about the SHA-512 message length restriction.



## **1.1   Continued**

### **Example 1**

**This example shows that the message length limitation of SHA-512 is not a serious problem. Suppose we need to send a message that is  $2^{128}$  bits in length. How long does it take for a communications network with a data rate of  $2^{64}$  bits per second to send this message?**

**Ví dụ: giới hạn độ dài tin nhắn của SHA-512 không phải là một vấn đề nghiêm trọng. Giả sử chúng ta cần gửi một tin nhắn có độ dài  $2^{128}$  bits. Mất bao lâu để một mạng truyền thông với tốc độ dữ liệu  $2^{64}$  bit / giây gửi thông điệp này?**

## 2.1 *Continued*

### Example 2

Hỏi cần bao nhiêu pages dùng cho bản tin có độ dài  $2^{128}$  bits

#### Solution

Suppose that a character is 32, or  $2^6$ , bits. Each page is less than 2048, or approximately  $2^{12}$ , characters. So  $2^{128}$  bits need at least  $2^{128} / 2^{18}$ , or  $2^{110}$ , pages. This again shows that we need not worry about the message length restriction.

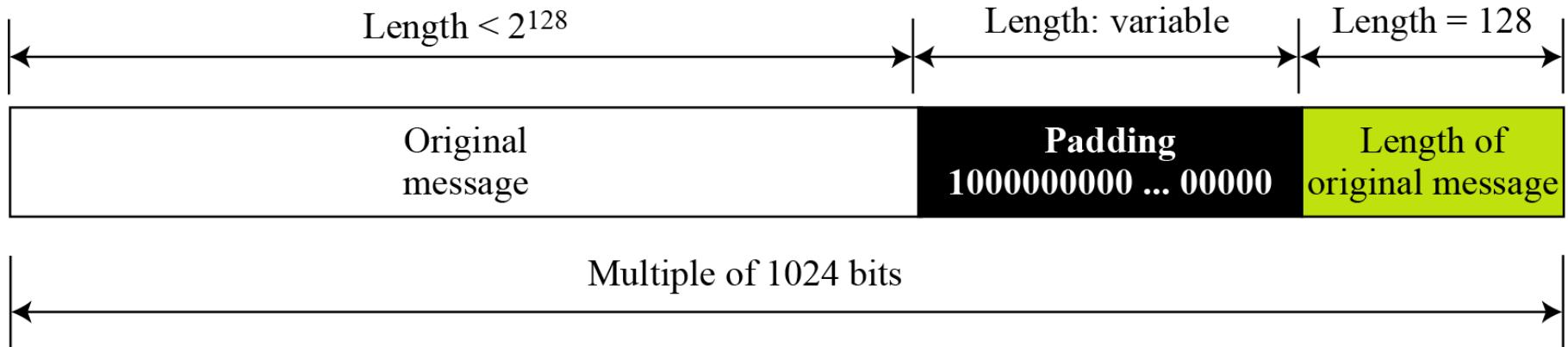
Giả sử rằng một ký tự là 32, hoặc  $2^6$ , bit. Mỗi trang ít hơn 2048, hoặc khoảng  $2^{12}$ , ký tự. Vì vậy,  $2^{128}$  bit cần ít nhất  $2^{128} / 2^{6+12}$  hoặc  $2^{110}$ , trang. Điều này một lần nữa cho thấy rằng chúng ta không cần phải lo lắng về giới hạn độ dài tin nhắn.

## 2.1 Continued

### Trường Padding và length

**Figure 7 Padding and length field in SHA-512**

**Trường đệm và độ dài trong SHA-512**



$$\begin{aligned} & (M+P+128) = 0 \bmod 1024 \\ \rightarrow & P = (-M-128) \bmod 1024 \end{aligned}$$

## 2.1 *Continued*

### Example 3

What is the number of padding bits if the length of the original message is 2590 bits?

*Số bit đệm Padding là bao nhiêu nếu độ dài của bản tin gốc là 2590 bit?*

#### Solution

Chúng ta có thể tính số bit đệm như sau:

$$|P| = (-2590 - 128) \bmod 1024 = -2718 \bmod 1024 = 354$$

The padding consists of one 1 followed by 353 0's.

Phần đệm bao gồm một số 1 sau là 353 số 0.

## 2.1 *Continued*

### Example 4

**Do we need padding if the length of the original message is already a multiple of 1024 bits?**

**Chúng ta có cần đệm không nếu độ dài của tin nhắn gốc đã là bội số của 1024 bit?**

### Solution

**Yes we do, because we need to add the length field. So padding is needed to make the new block a multiple of 1024 bits.**

**Có, bởi vì chúng ta cần thêm trường độ dài. Vì vậy, cần đệm để tạo khối mới là bội số của 1024 bits.**

## 2.1   Continued

### Example 5

What is the minimum and maximum number of padding bits that can be added to a message?

Số bit đệm tối thiểu và tối đa có thể được thêm vào bản tin là bao nhiêu?

### Solution

- a. The minimum length of padding is 0 and it happens when  $(-M - 128) \bmod 1024$  is 0. This means that  $|M| = -128 \bmod 1024 = 896 \bmod 1024$  bits. In other words, the last block in the original message is 896 bits. We add a 128-bit length field to make the block complete.

Độ dài tối thiểu của padding là 0 và nó xảy ra khi  $(-M - 128) \bmod 1024$  bằng 0. Điều này có nghĩa là  $|M| = -128 \bmod 1024 = 896 \bmod 1024$  bit. Nói cách khác, khối cuối cùng trong thông điệp gốc là 896 bit. Chúng ta thêm trường độ dài 128 bit để làm cho khối hoàn chỉnh.

## 2.1 *Continued*

### Example 5

### *Continued*

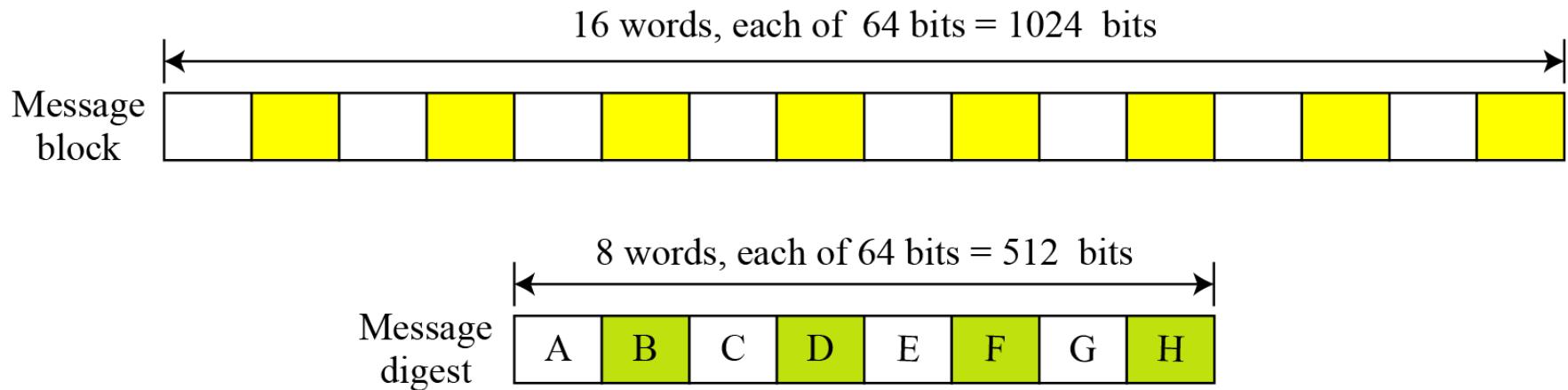
- b) The maximum length of padding is 1023 and it happens when  $(-|M| - 128) = 1023 \bmod 1024$ . This means that the length of the original message is  $|M| = (-128 - 1023) \bmod 1024$  or the length is  $|M| = 897 \bmod 1024$ . In this case, we cannot just add the length field because the length of the last block exceeds one bit more than 1024. So we need to add 897 bits to complete this block and create a second block of 896 bits. Now the length can be added to make this block complete.

*Độ dài tối đa của vùng đệm là 1023 và điều đó xảy ra khi  $(- | M | - 128) = 1023 \bmod 1024$ . Điều này có nghĩa là độ dài của thông điệp gốc là  $| M | = (-128 - 1023) \bmod 1024$  hoặc độ dài là  $| M | = 897 \bmod 1024$ . Trong trường hợp này, chúng ta không thể chỉ thêm trường độ dài vì độ dài của khối cuối cùng vượt quá một bit nhiều hơn 1024. Vì vậy, chúng ta cần thêm 897 bit để hoàn thành khối này và tạo khối thứ hai gồm 896 bit. Bây giờ chiều dài có thể được thêm vào để làm cho khối này hoàn chỉnh.*

## 2.1 Continued

### Words

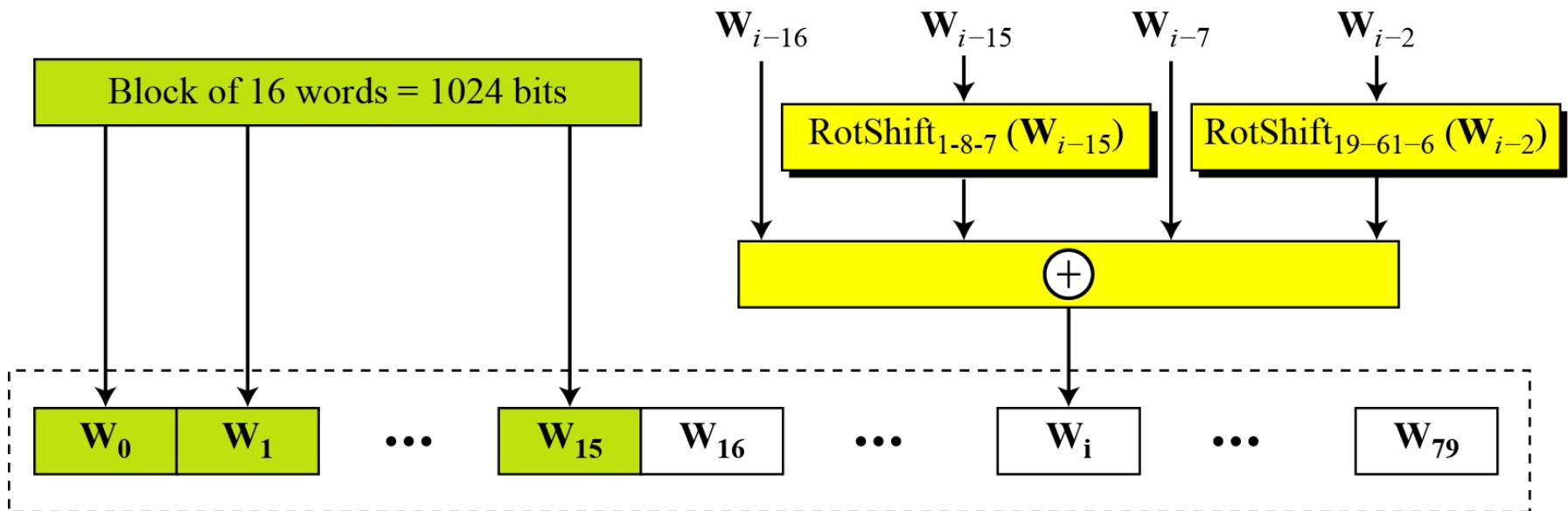
**Figure 12.8 A message block and the digest as words**  
*Một khối bản tin và thông báo dưới dạng các từ*



## 2.1 Continued

### Word Expansion: Mở rộng từ

Figure 9 Word expansion in SHA-512



$\text{RotShift}_{l-m-n}(x)$ :  $\text{RotR}_l(x) \oplus \text{RotR}_m(x) \oplus \text{ShL}_n(x)$

$\text{RotR}_i(x)$ : Right-rotation of the argument  $x$  by  $i$  bits

$\text{ShL}_i(x)$ : Shift-left of the argument  $x$  by  $i$  bits and padding the left by 0's.

## 2.1 *Continued*

### Example 6

Show how W60 is made?

Cho biết W60 được tạo ra như thế nào.

#### Solution

Each word in the range W16 to W79 is made from four previously-made words. W60 is made as

Mỗi từ trong phạm vi W16 đến W79 được tạo từ bốn từ đã tạo trước đó. W60 được làm như

$$W_{60} = W_{44} \oplus \text{RotShift}_{1-8-7}(W_{45}) \oplus W_{53} \oplus \text{RotShift}_{19-61-6}(W_{58})$$

## 2.1 Continued

### Message Digest Initialization

**Table 12.2** Values of constants in message digest initialization of SHA-512

Buffer	Value (in hexadecimal)	Buffer	Value (in hexadecimal)
A <sub>0</sub>	<b>6A09E667F3BCC908</b>	E <sub>0</sub>	<b>510E527FADE682D1</b>
B <sub>0</sub>	<b>BB67AE8584CAA73B</b>	F <sub>0</sub>	<b>9B05688C2B3E6C1F</b>
C <sub>0</sub>	<b>3C6EF372EF94F828</b>	G <sub>0</sub>	<b>1F83D9ABFB41BD6B</b>
D <sub>0</sub>	<b>A54FE53A5F1D36F1</b>	H <sub>0</sub>	<b>5BE0CD19137E2179</b>

Các giá trị khởi tạo trên được tính toán từ 8 số nguyên tố sau :  
2, 3, 5, 7, 11, 13, 17, và 19.

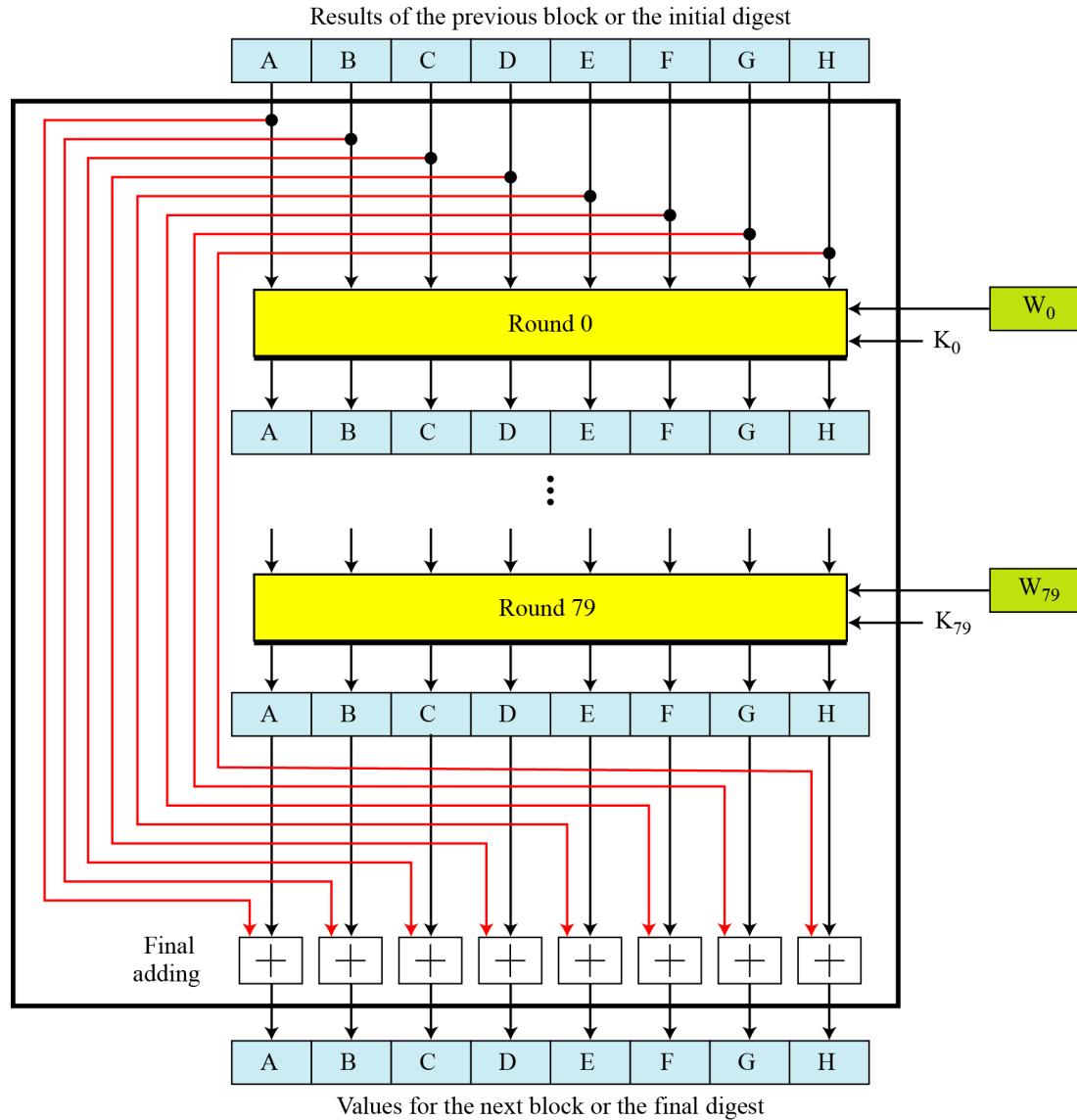
Sau đó **mũ ½** → chuyển đổi thành nhị phân và chỉ giữ lại 64 bits đầu.

Ví dụ:  $\sqrt{19}=4.35889894354\dots$

→0100.0010 0101 ....

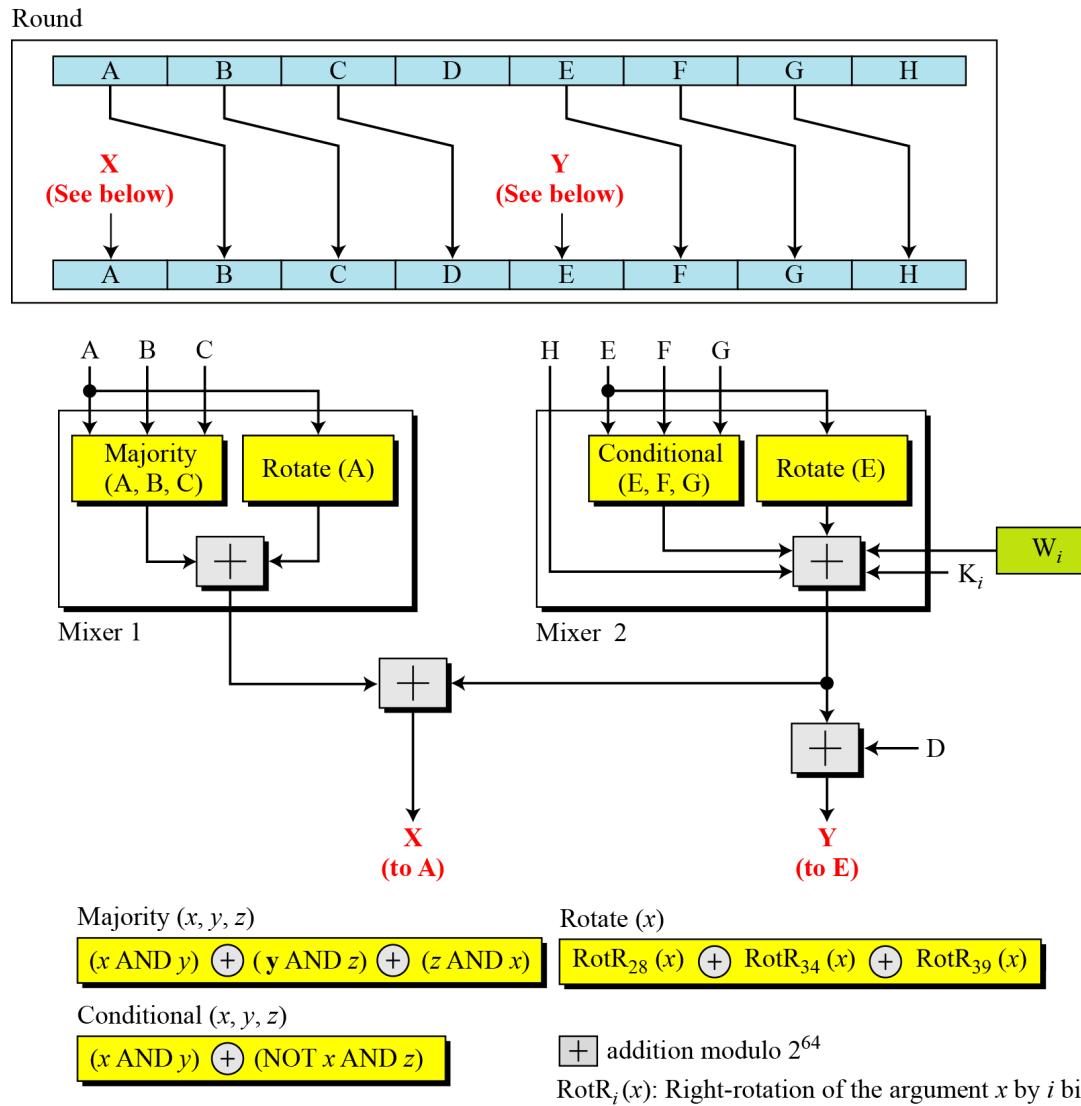
## 2.2 Compression Function: *Chức năng nén*

Figure 10 *Compression function in SHA-512*  
*Chức năng nén trong SHA-512*



## 2.2 Continued

**Figure 11** Structure of each round in SHA-512



## 2.2 *Continued*

### *Majority Function*

$$(A_j \text{AND } B_j) \oplus (B_j \text{AND } C_j) \oplus (C_j \text{AND } A_j)$$

### *Conditional Function*

$$(E_j \text{AND } F_j) \oplus (\text{NOT } E_j \text{AND } G_j)$$

### *Rotate Functions*

**Rotate (A):**  $\text{RotR}_{28}(A) \oplus \text{RotR}_{34}(A) \oplus \text{RotR}_{29}(A)$

**Rotate (E):**  $\text{RotR}_{28}(E) \oplus \text{RotR}_{34}(E) \oplus \text{RotR}_{29}(E)$

## 2.2 *Continued*

**Table 12.3** Eighty constants used for eighty rounds in SHA-512

428A2F98D728AE22	7137449123EF65CD	B5C0FBCFEC4D3B2F	E9B5DBA58189DBBC
3956C25BF348B538	59F111F1B605D019	923F82A4AF194F9B	AB1C5ED5DA6D8118
D807AA98A3030242	12835B0145706FBE	243185BE4EE4B28C	550C7DC3D5FFB4E2
72BE5D74F27B896F	80DEB1FE3B1696B1	9BDC06A725C71235	C19BF174CF692694
E49B69C19EF14AD2	EFBE4786384F25E3	0FC19DC68B8CD5B5	240CA1CC77AC9C65
2DE92C6F592B0275	4A7484AA6EA6E483	5CB0A9DCBD41FBD4	76F988DA831153B5
983E5152EE66DFAB	A831C66D2DB43210	B00327C898FB213F	BF597FC7BEEF0EE4
C6E00BF33DA88FC2	D5A79147930AA725	06CA6351E003826F	142929670A0E6E70
27B70A8546D22FFC	2E1B21385C26C926	4D2C6DFC5AC42AED	53380D139D95B3DF
650A73548BAF63DE	766A0ABB3C77B2A8	81C2C92E47EDAEE6	92722C851482353B
A2BFE8A14CF10364	A81A664BBC423001	C24B8B70D0F89791	C76C51A30654BE30
D192E819D6EF5218	D69906245565A910	F40E35855771202A	106AA07032BBD1B8
19A4C116B8D2D0C8	1E376C085141AB53	2748774CDF8EEB99	34B0BCB5E19B48A8
391C0CB3C5C95A63	4ED8AA4AE3418ACB	5B9CCA4F7763E373	682E6FF3D6B2B8A3
748F82EE5DEFB2FC	78A5636F43172F60	84C87814A1F0AB72	8CC702081A6439EC
90BEFFFA23631E28	A4506CEBDE82BDE9	BEF9A3F7B2C67915	C67178F2E372532B
CA273ECEEA26619C	D186B8C721C0C207	EADA7DD6CDE0EB1E	F57D4F7FEE6ED178
06F067AA72176FBA	0A637DC5A2C898A6	113F9804BEF90DAE	1B710B35131C471B
28DB77F523047D84	32CAAB7B40C72493	3C9EBE0A15C9BEBC	431D67C49C100D4C
4CC5D4BECB3E42B6	4597F299CFC657E2	5FCB6FAB3AD6FAEC	6C44198C4A475817

## 2.2 Continued

*There are 80 constants,  $K_0$  to  $K_{79}$ , each of 64 bits. Similar These values are calculated from the first 80 prime numbers (2, 3,..., 409). For example, the 80th prime is 409, with the cubic root  $(409)^{1/3} = 7.42291412044\dots$  Converting this number to binary with only 64 bits in the fraction part, we get*

$$(111.0110\ 1100\ 0100\ 0100\ \dots\ 0111)_2 \rightarrow (7.6C44198C4A475817)_{16}$$

*The fraction part:  $(6C44198C4A475817)_{16}$*

## 2.2 *Continued*

### Example 7

We apply the Majority function on buffers A, B, and C. If the leftmost hexadecimal digits of these buffers are 0x7, 0xA, and 0xE, respectively, what is the leftmost digit of the result?

### Solution

The digits in binary are 0111, 1010, and 1110.

- a. The first bits are 0, 1, and 1. The majority is 1.
- b. The second bits are 1, 0, and 1. The majority is 1.
- c. The third bits are 1, 1, and 1. The majority is 1.
- d. The fourth bits are 1, 0, and 0. The majority is 0.

The result is 1110, or 0xE in hexadecimal.

## 2.2 *Continued*

### Example 8

We apply the Conditional function on E, F, and G buffers. If the leftmost hexadecimal digits of these buffers are 0x9, 0xA, and 0xF respectively, what is the leftmost digit of the result?

### Solution

The digits in binary are 1001, 1010, and 1111.

- a. The first bits are 1, 1, and 1. The result is  $F_1$ , which is 1.
- b. The second bits are 0, 0, and 1. The result is  $G_2$ , which is 1.
- c. The third bits are 0, 1, and 1. The result is  $G_3$ , which is 1.
- d. The fourth bits are 1, 0, and 1. The result is  $F_4$ , which is 0.

The result is 1110, or 0xE in hexadecimal.

## 2.3 Analysis

*With a message digest of 512 bits, SHA-512 expected to be resistant to all attacks, including collision attacks.*

*Với bản tóm tắt thông điệp 512 bit, SHA-512 dự kiến có khả năng chống lại tất cả các cuộc tấn công, bao gồm cả các cuộc tấn công va chạm.*

### 3 WHIRLPOOL

*Whirlpool is an iterated cryptographic hash function, based on the Miyaguchi-Preneel scheme, that uses a **symmetric-key block cipher** in place of the **compression function**. The block cipher is a **modified AES** cipher that has been tailored for this purpose.*

*Whirlpool là một hàm băm mật mã được lặp đi lặp lại, dựa trên lược đồ Miyaguchi-Preneel, sử dụng mật mã khối khóa đối xứng thay cho hàm nén. Mật mã khối là một mật mã AES đã được sửa đổi đã được điều chỉnh cho mục đích này. Whirlpool được thiết kế bởi Vincent Rijmen và Paulo S.L.M. Barreto.*

**Topics discussed in this section:**

**12.3.1 Whirlpool Cipher**

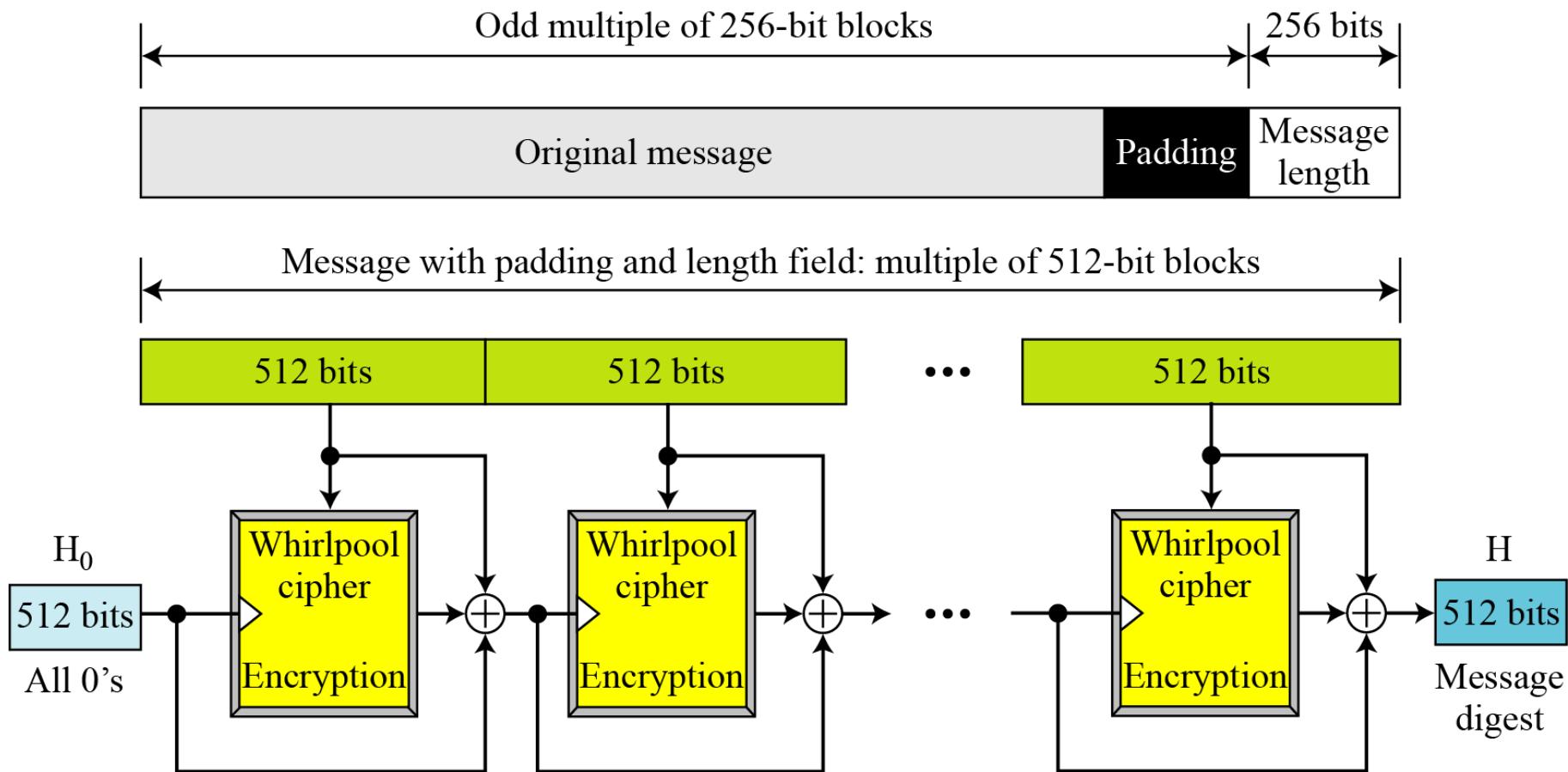
**12.3.2 Summary**

**12.3.3 Analysis**

### 3 Continued

Padding: phần các bit đệm, bắt đầu bằng bit 1, còn lại là các bit 0, cuối cùng là khối 256 bits chỉ độ dài bản tin gốc (Original message).

**Figure 12 Whirlpool hash function  
Hàm băm Whirlpool**

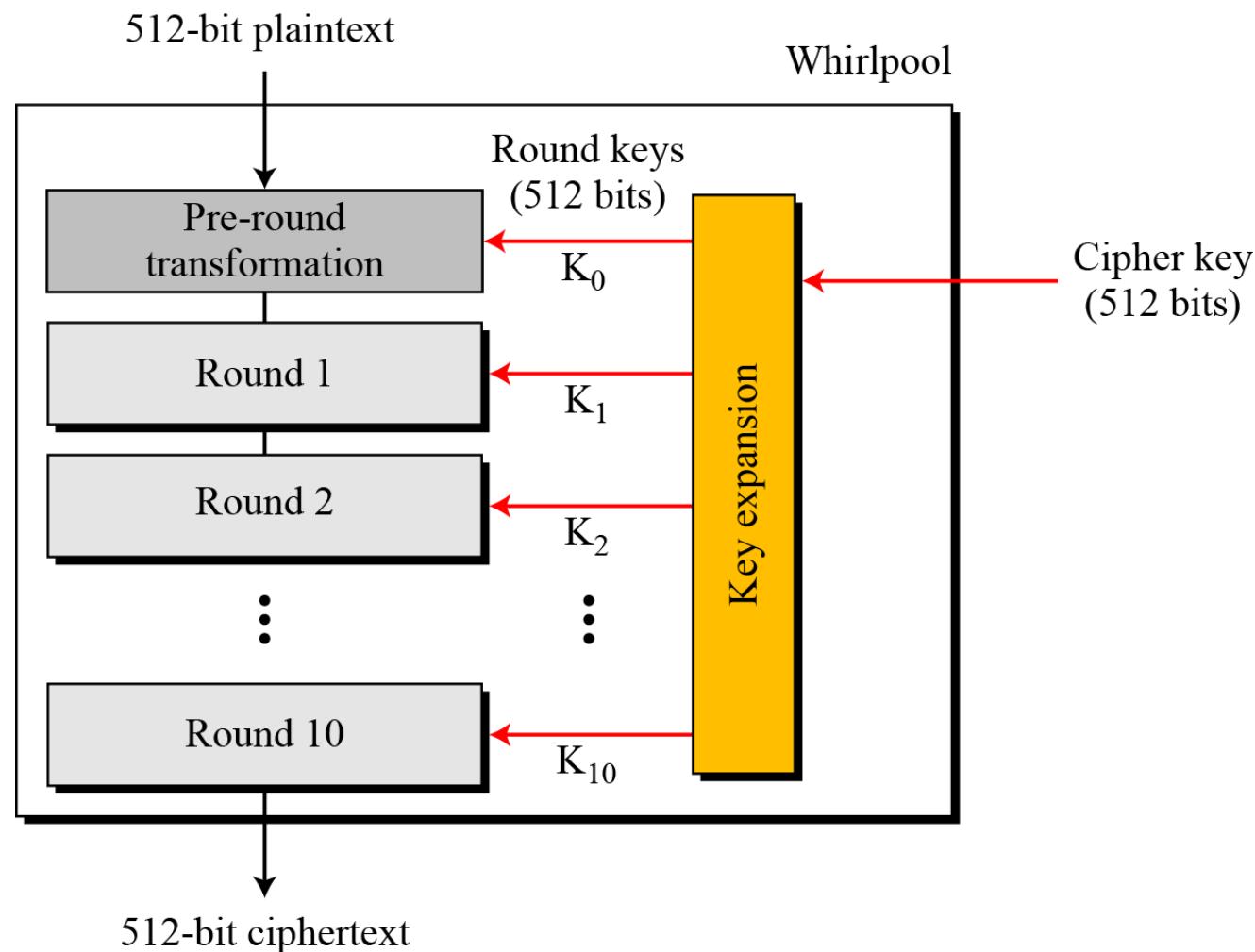


## 3.1 Whirlpool Cipher

Whirlpool dựa trên hệ mật kiều non-Feistel giống như AES để thực hiện phép mã hóa băm. Whirlpool gồm 10 round, mỗi khối có kích thước khối và khóa là 512 bits được đánh số từ K0 đến K10.

**Figure 13 General idea of the Whirlpool cipher**

*Ý tưởng chung về mật mã Xoáy nước*

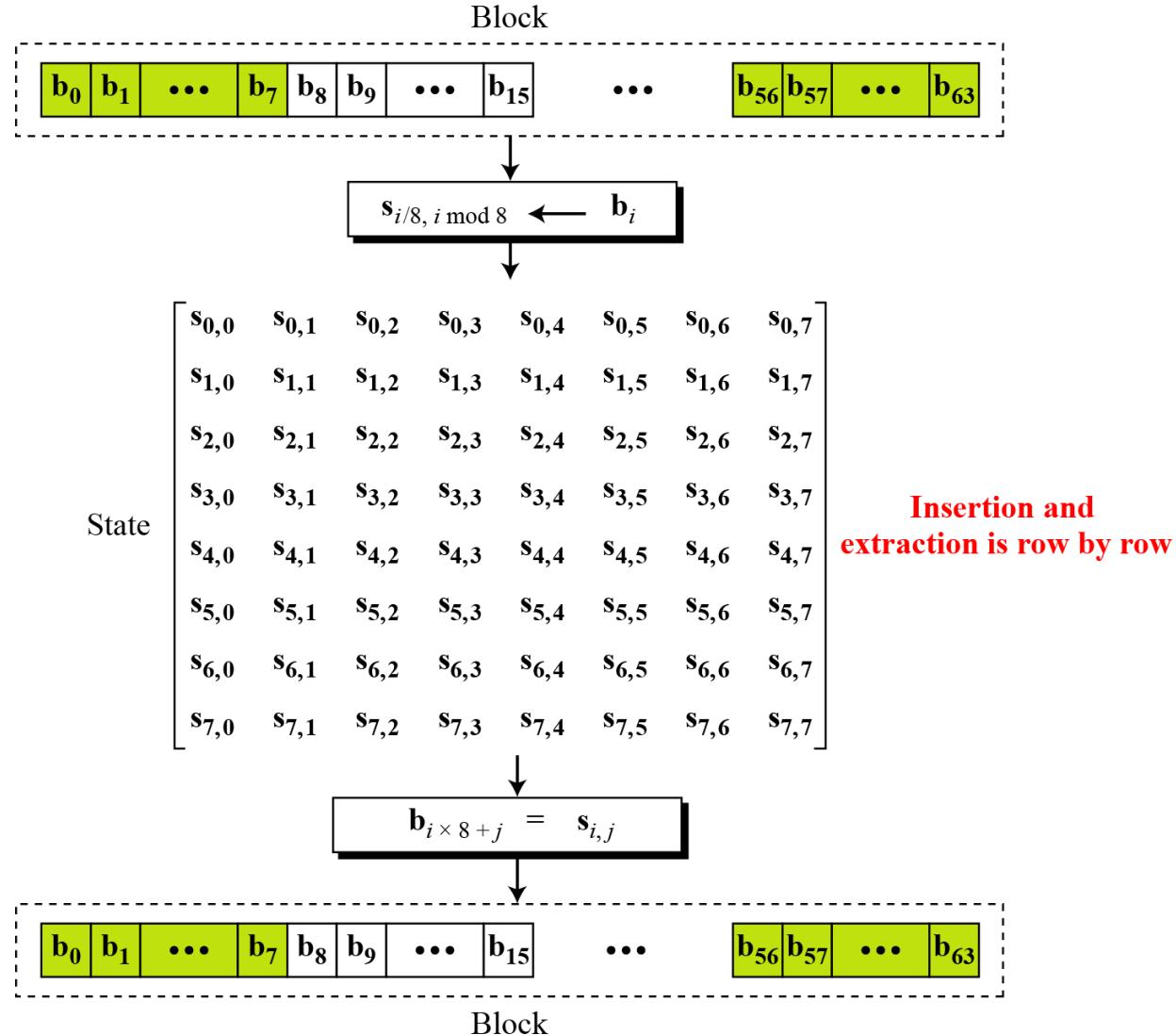


### 3.1 Continued

**Figure 14 Block and state in the Whirlpool cipher**  
*Khối và trạng thái trong mật mã Whirlpool*

Giống như AES, Whirlpool sử dụng các block và state. Một block xem như là 1 ma trận hàng 64 bytes; một state là ma trận vuông kích thước  $8 \times 8$  bytes.

Tuy nhiên, không giống như AES, việc viết block vào state là theo hàng.

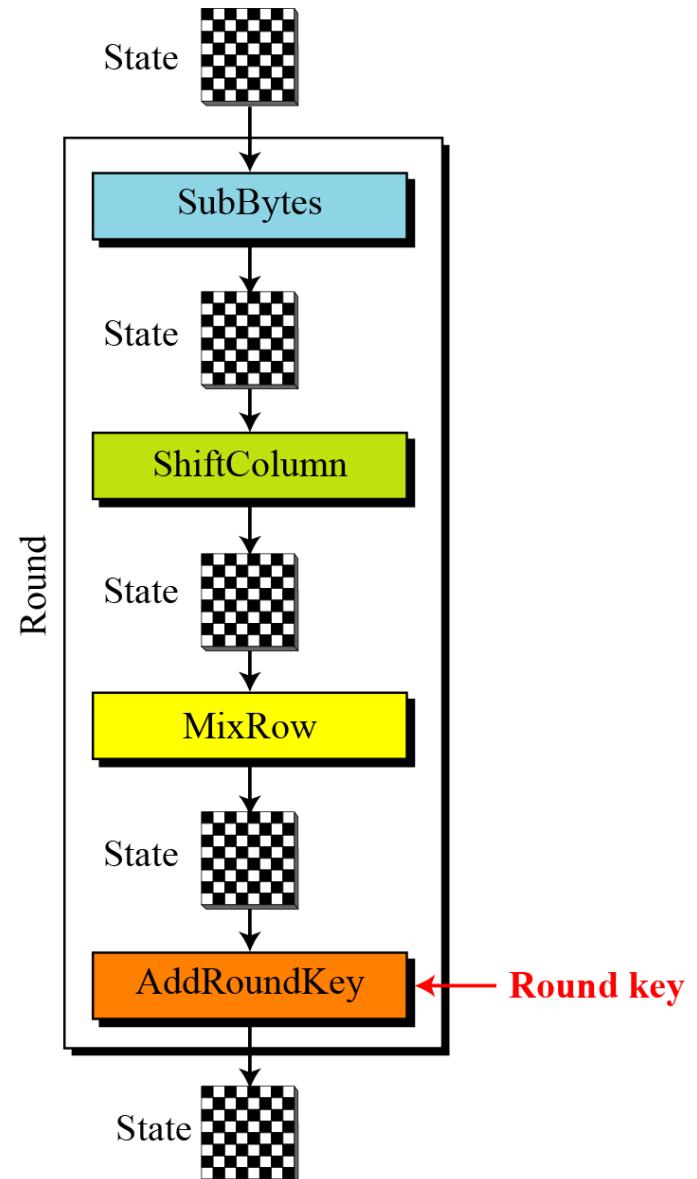


## 3.1 Continued

### *Structure of Each Round*

*Each round uses four transformations.*

*Mỗi vòng sử dụng bốn phép biến đổi.*



**Figure 15** *Structure of each round in the Whirlpool cipher*

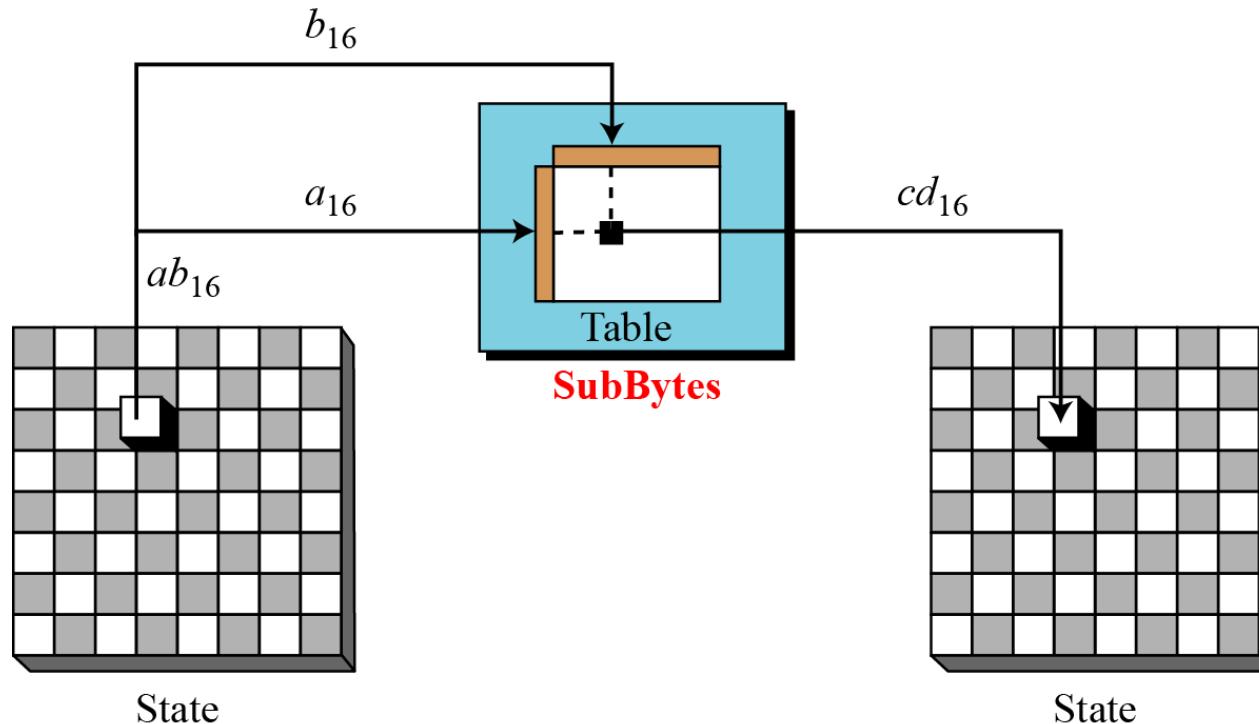
### 3.1 Continued

*SubBytes* Like in AES, SubBytes provide a nonlinear transformation.

*Giống như trong AES, SubBytes cung cấp một phép biến đổi phi tuyến.*

**Figure 16** SubBytes transformations in the Whirlpool cipher

Các phép biến đổi SubBytes trong mật mã Whirlpool



## 12.3.1 Continued

**Table 12.4** SubBytes transformation table (S-Box)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	18	23	C6	E8	87	B8	01	4F	36	A6	D2	F5	79	6F	91	52
1	16	BC	9B	8E	A3	0C	7B	35	1D	E0	D7	C2	2E	4B	FE	57
2	15	77	37	E5	9F	F0	4A	CA	58	C9	29	0A	B1	A0	6B	85
3	BD	5D	10	F4	CB	3E	05	67	E4	27	41	8B	A7	7D	95	C8
4	FB	EF	7C	66	DD	17	47	9E	CA	2D	BF	07	AD	5A	83	33
5	63	02	AA	71	C8	19	49	C9	F2	E3	5B	88	9A	26	32	B0
6	E9	0F	D5	80	BE	CD	34	48	FF	7A	90	5F	20	68	1A	AE
7	B4	54	93	22	64	F1	73	12	40	08	C3	EC	DB	A1	8D	3D
8	97	00	CF	2B	76	82	D6	1B	B5	AF	6A	50	45	F3	30	EF
9	3F	55	A2	EA	65	BA	2F	C0	DE	1C	FD	4D	92	75	06	8A
A	B2	E6	0E	1F	62	D4	A8	96	F9	C5	25	59	84	72	39	4C
B	5E	78	38	8C	C1	A5	E2	61	B3	21	9C	1E	43	C7	FC	04
C	51	99	6D	0D	FA	DF	7E	24	3B	AB	CE	11	8F	4E	B7	EB
D	3C	81	94	F7	9B	13	2C	D3	E7	6E	C4	03	56	44	7E	A9
E	2A	BB	C1	53	DC	0B	9D	6C	31	74	F6	46	AC	89	14	E1
F	16	3A	69	09	70	B6	C0	ED	CC	42	98	A4	28	5C	F8	86

# 3.1 Continued

**Figure 17 SubBytes in the Whirlpool cipher**

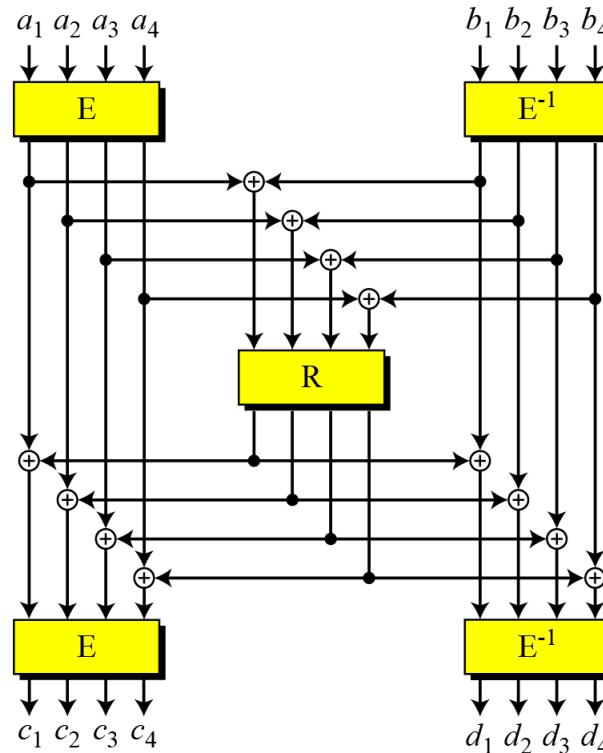
Các phần tử trong bảng 12.4 ở slide trước, bangr SubBytes, có thể tính toán sử dụng trường GF(2<sup>4</sup>) với đa thức tối giản  $x^4 + x + 1$ :

$$E(\text{input}) = (x^3 + x + 1)^{\text{input}} \quad \text{mod}$$

(x<sup>4</sup>+x+1) nếu input khác 0xF

$$E(0xF) = 0.$$

$E^{-1}$  là phép nghịch đảo của E.



Input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Output	1	B	9	C	D	6	F	3	E	8	7	4	A	2	5	0
Input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Output	F	0	D	7	B	E	5	A	9	2	C	1	3	4	8	6
Input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Output	7	C	B	D	E	4	9	F	6	3	8	A	2	5	1	0

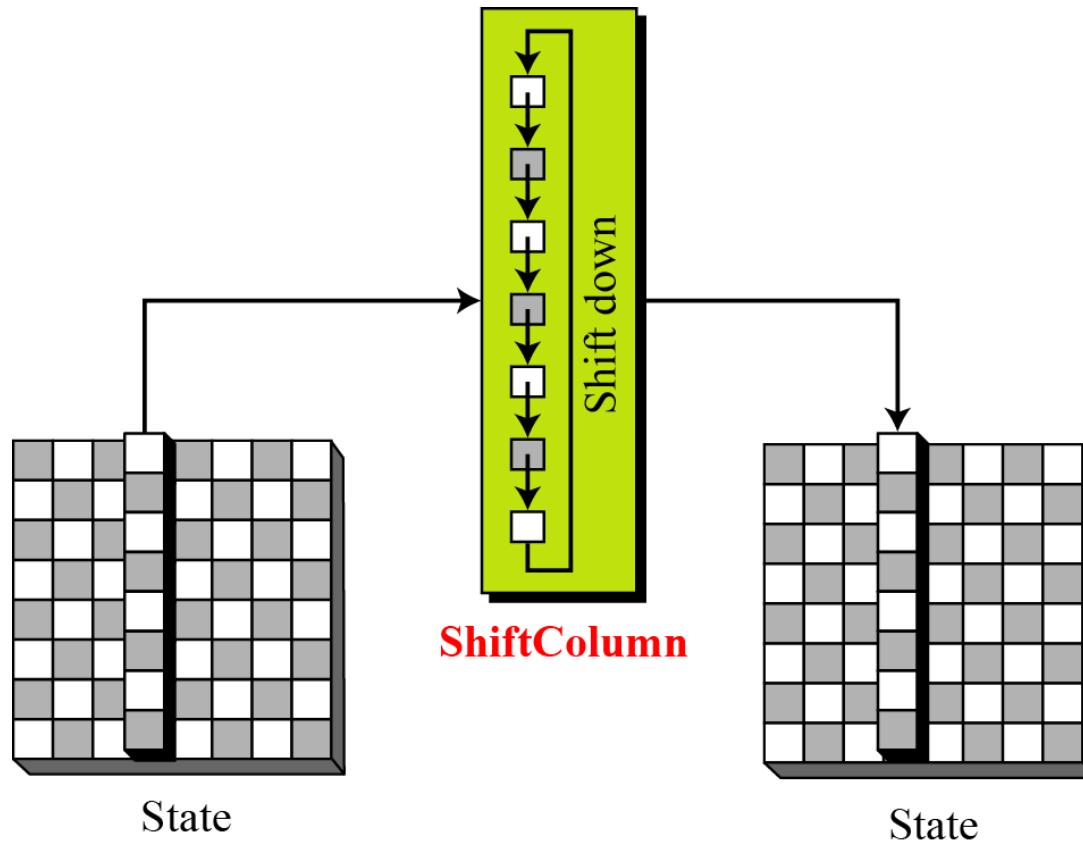
### 3.1 Continued

*ShiftColumns: phép dịch cột*

*Giống phép dịch hàng shiftRows trong AES.*

**Figure 18** ShiftColumns transformation in the Whirlpool cipher

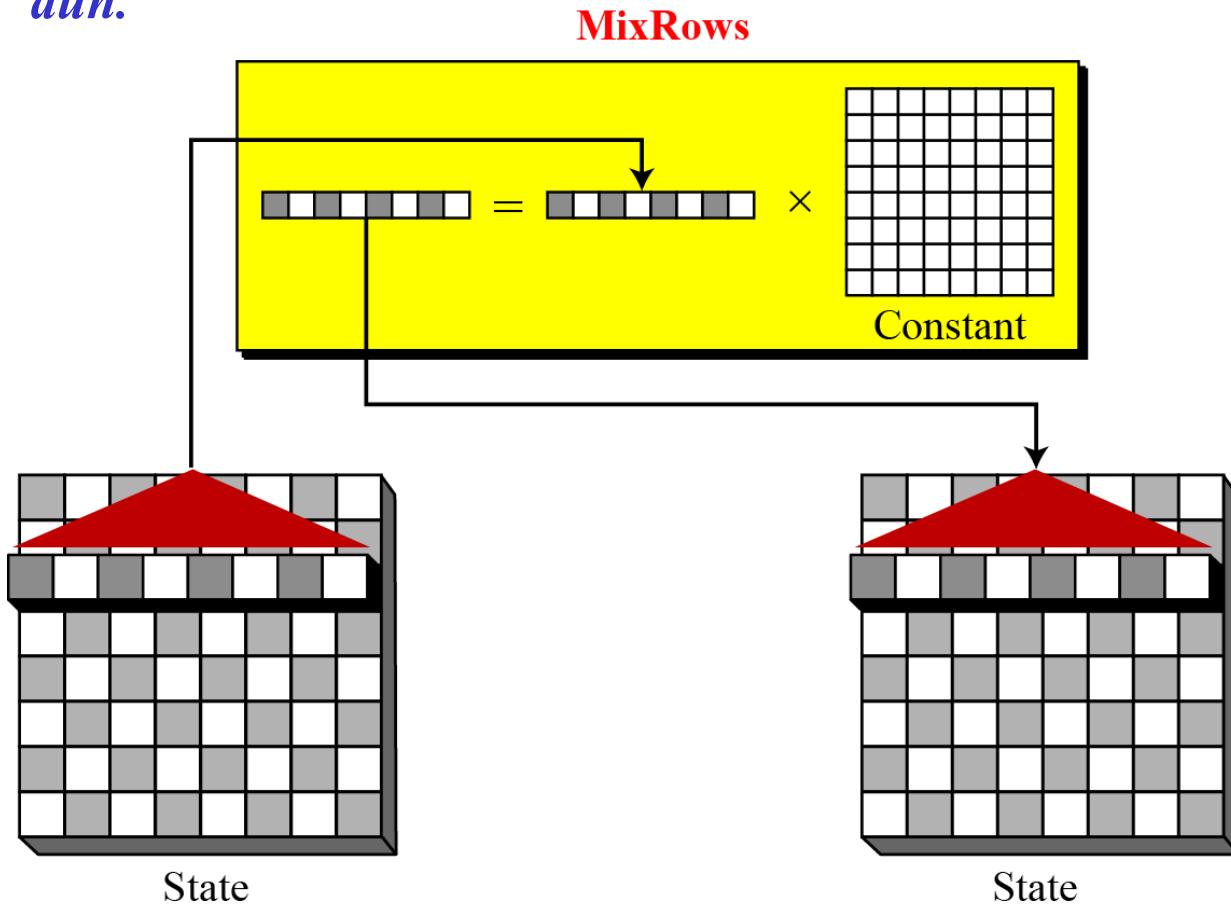
*Chuyển đổi ShiftColumns trong mật mã Xoáy nước*



# 3.1 Continued

## MixRows: phép trộn

Giống như AES, phép nhân byte được thực hiện trong  $GF(2^8)$ , đa thức tối giản  $x^8+x^4+x^3+x^2+1$  trong phép biến đổi mỗ dun.



**Figure 19** MixRows transformation in the Whirlpool cipher

Phép biến đổi MixRows trong mật mã Xoáy nước

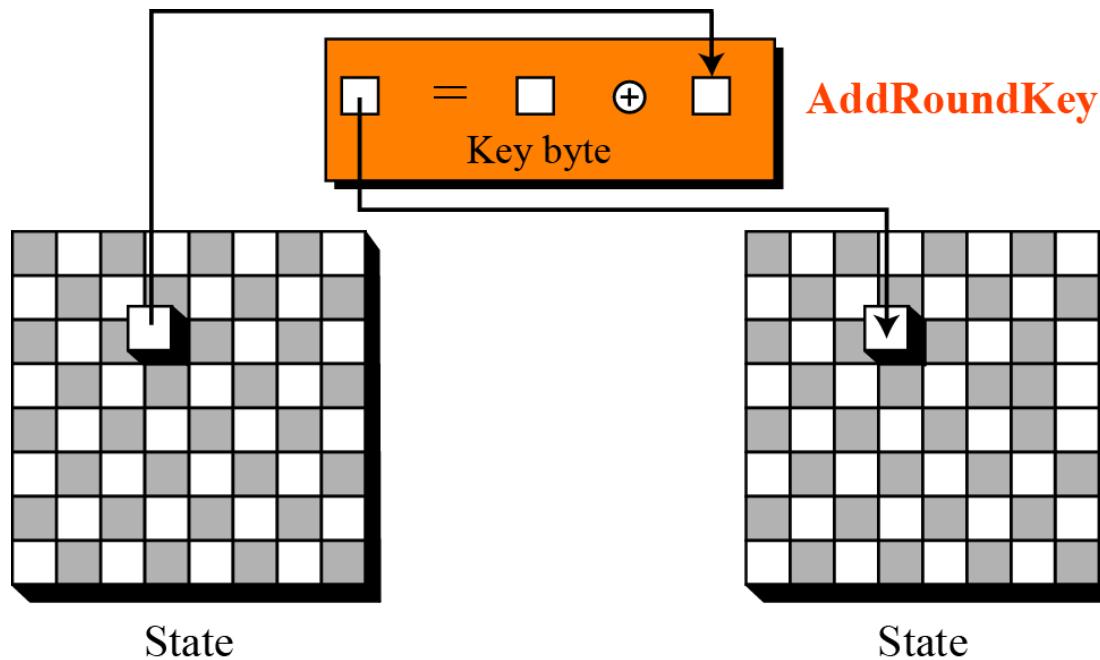
01	01	04	01	08	05	02	09
09	01	01	04	01	08	05	02
02	09	01	01	04	01	08	05
05	02	09	01	01	04	01	08
08	05	02	09	01	01	04	01
01	08	05	02	09	01	01	04
04	01	08	05	02	09	01	01
01	04	01	08	05	02	09	01

Constant matrix

### 3.1 Continued

*AddRoundKey: được thực hiện theo byte trong trường  $GF(2^8)$*

**Figure 20** *AddRoundKey transformation in the Whirlpool cipher*  
*Phép biến đổi AddRoundKey trong mật mã Whirlpool*



# 3.1 Continued

**Figure 21** Key expansion in the Whirlpool cipher

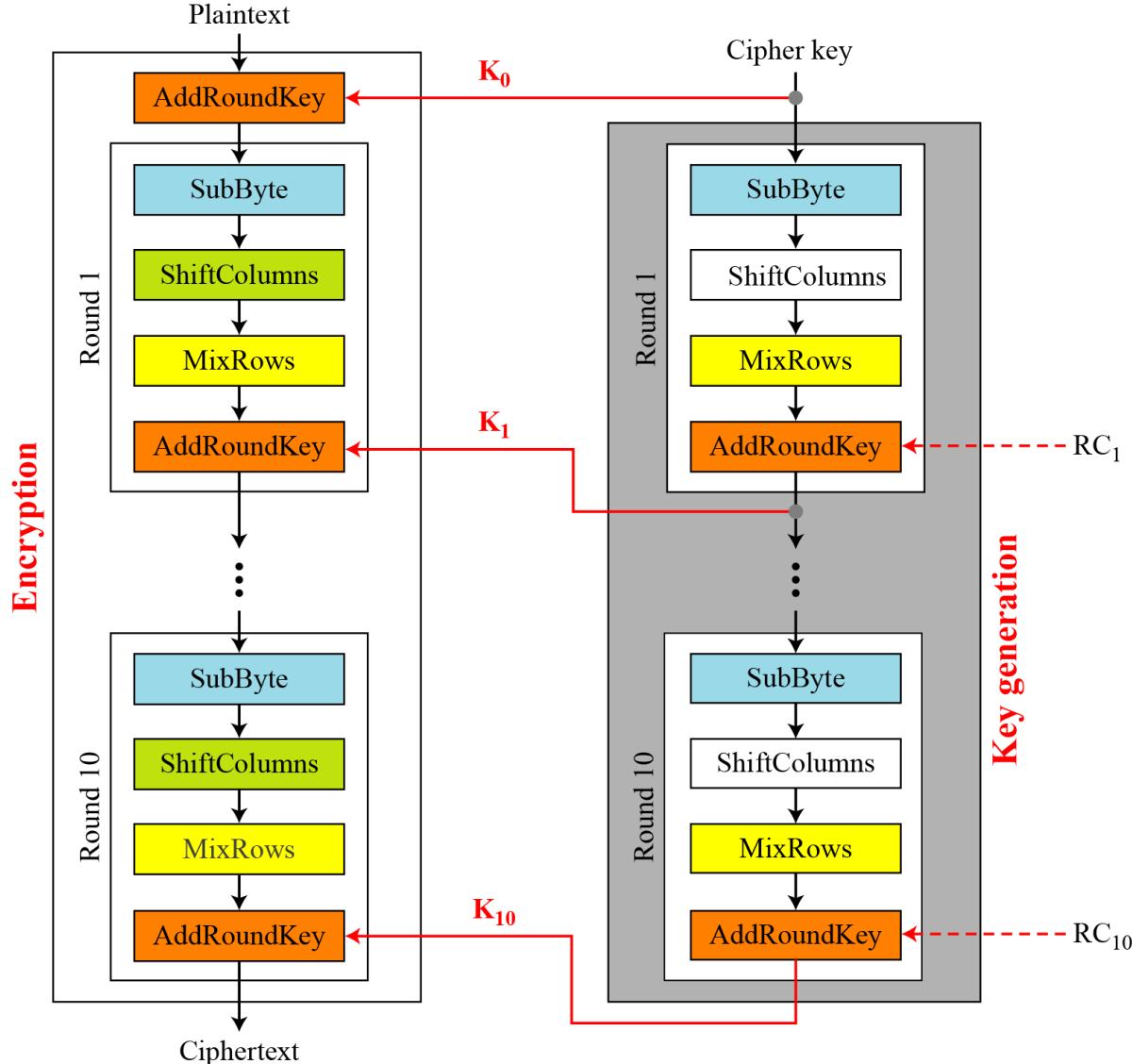
Mở rộng khóa trong mật mã Whirlpool

Mở rộng khóa trong mật mã Whirlpool cơ bản khác với mở rộng khóa của AES.

RC: hằng số vòng (Round Constant)

Thuật toán mở rộng khóa sử dụng các hằng số như là các khóa vòng và thuật toán mã hóa sử dụng đầu ra của mỗi vòng của thuật toán tạo khóa đó.

Thuật toán tạo khóa xem khóa mật như là bản tin plaintext để mã hóa nó.



### **3.1 *Continued***

## *Round constants*

**Figure 22** Round constant for the third round

## **VÍ dụ giá trị RC3 kích thước 8x8**

*Đối với mỗi vòng RCx là ma trận  $8 \times 8$ , trong đó chỉ có hàng đầu tiên có các giá trị khác không, các hàng còn lại bằng 0*

**Giá trị của hàng đầu tiên được thực hiện sử dụng phép biến đổi SubBytes theo bảng 12.4**

**RC\_round** [row, column] = SubBytes (8 (round-1)+column) nếu row =0;  
**RC round** [row, column] = 0, nếu row khác 0.

## 3.2 Summary

**Table 12.5** *Main characteristics of the Whirlpool cipher*

Block size: 512 bits
Cipher key size: 512 bits
Number of rounds: 10
Key expansion: using the cipher itself with round constants as round keys
Substitution: SubBytes transformation
Permutation: ShiftColumns transformation
Mixing: MixRows transformation
Round Constant: cubic roots of the first eighty prime numbers

### *3.3 Analysis*

*Although Whirlpool has not been extensively studied or tested, it is based on a robust scheme (Miyaguchi-Preneel), and for a compression function uses a cipher that is based on AES, a cryptosystem that has been proved very resistant to attacks. In addition, the size of the message digest is the same as for SHA-512. Therefore, it is expected to be a very strong cryptographic hash function.*

*Mặc dù Whirlpool chưa được nghiên cứu hoặc thử nghiệm rộng rãi, nhưng nó dựa trên một sơ đồ mạnh mẽ (Miyaguchi-Preneel) và đối với chức năng nén sử dụng mật mã dựa trên AES, một hệ thống mật mã đã được chứng minh là có khả năng chống lại các cuộc tấn công. Ngoài ra, kích thước của bản tin digest giống như đối với SHA-512. Do đó, Whirlpool được kỳ vọng là một hàm băm mật mã rất mạnh.*