Chapter 12

Cryptographic Hash Functions

Chapter 12

Objectives

- To introduce general ideas behind cryptographic hash functions
- To discuss the Merkle-Damgard scheme as the basis for iterated hash functions
- To distinguish between two categories of hash functions:
- To discuss the structure of SHA-512.
- To discuss the structure of Whirlpool.

12-1 INTRODUCTION

A cryptographic hash function takes a message of arbitrary length and creates a message digest of fixed length. The ultimate goal of this chapter is to discuss the details of the two most promising cryptographic hash algorithms—SHA-512 and Whirlpool.

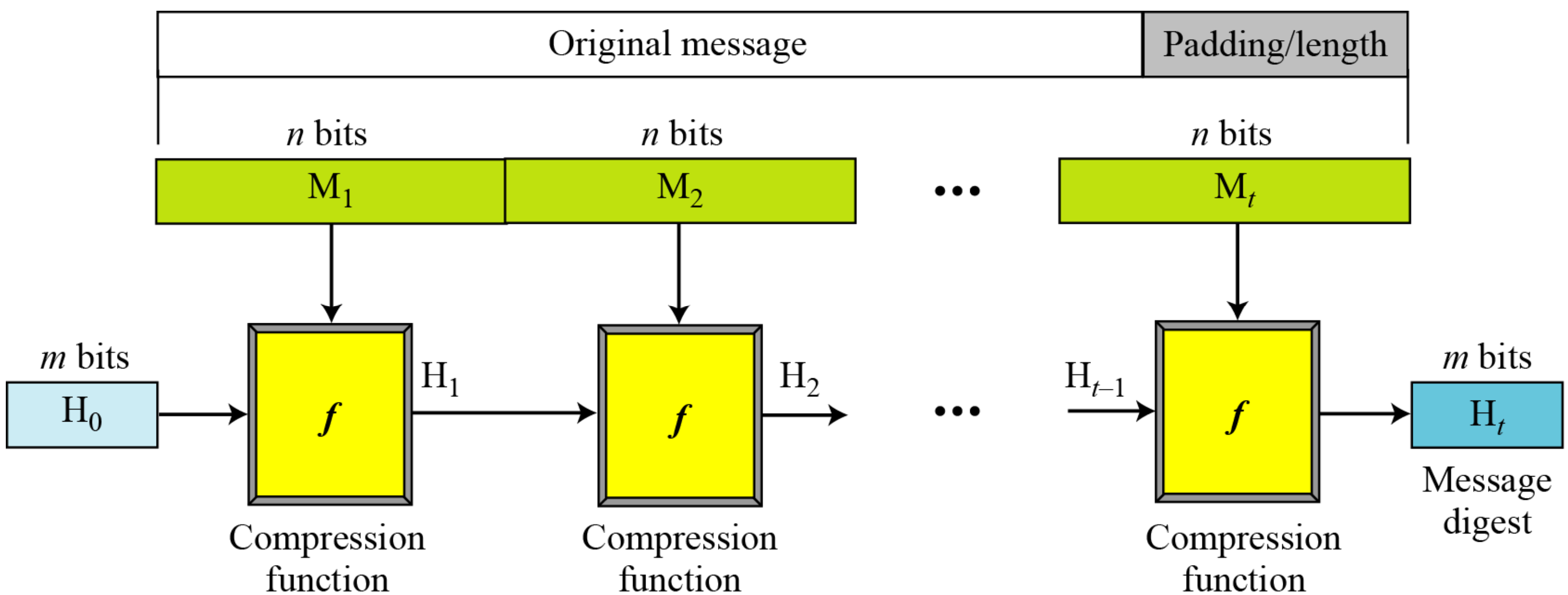
Topics discussed in this section:

12.1.1 Iterated Hash Function
12.1.2 Two Groups of Compression Functions

12.1.1 Iterated Hash Function

Merkle-Damgard Scheme

Figure 12.1 ***Merkle-Damgard scheme***



12.1.2 Two Groups of Compression Functions

1. The compression function is made from scratch.

Message Digest (MD)

2. A symmetric-key block cipher serves as a compression function.

Whirlpool

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

	MD5	SHA-1	RIPEMD-160
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	∞	$2^{64} - 1$ 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

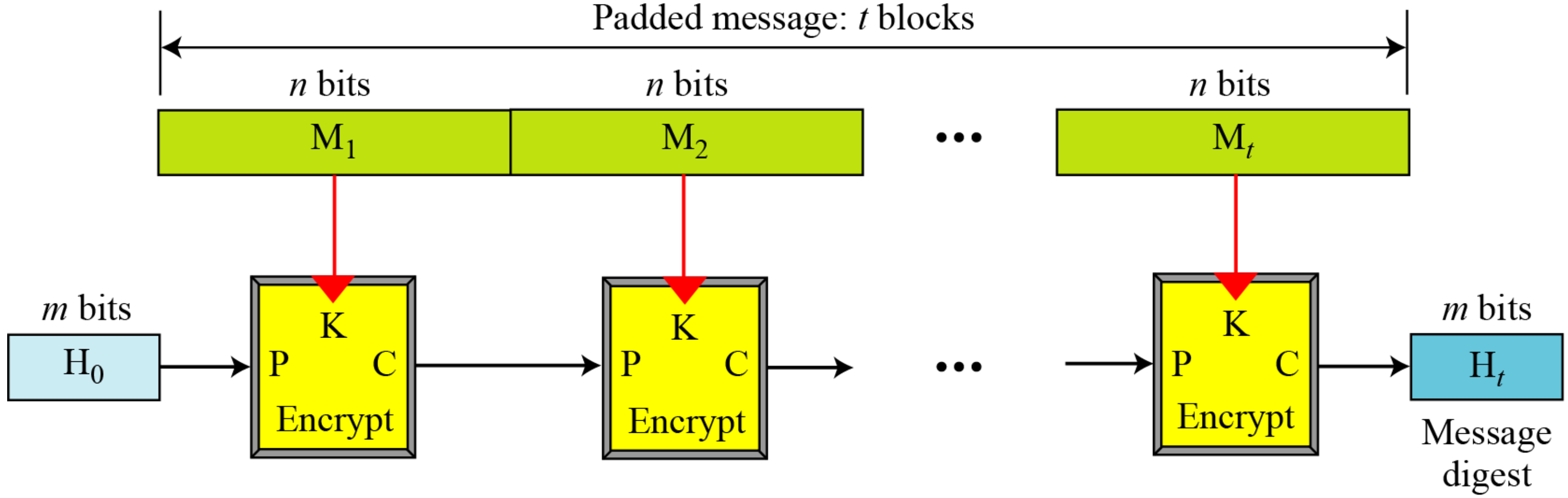
12.1.2 Continued

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

Characteristics	*SHA-1*	*SHA-224*	*SHA-256*	*SHA-384*	*SHA-512*
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

Rabin Scheme

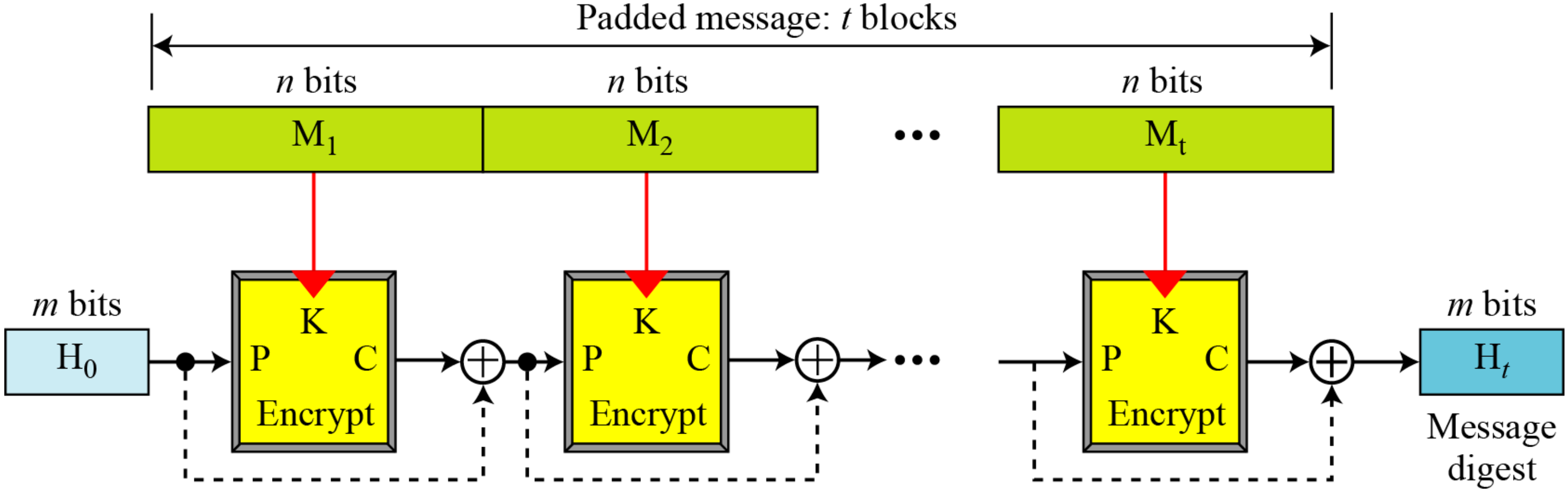
Figure 12.2 ***Rabin scheme***



12.1.2 Continued

Davies-Meyer Scheme

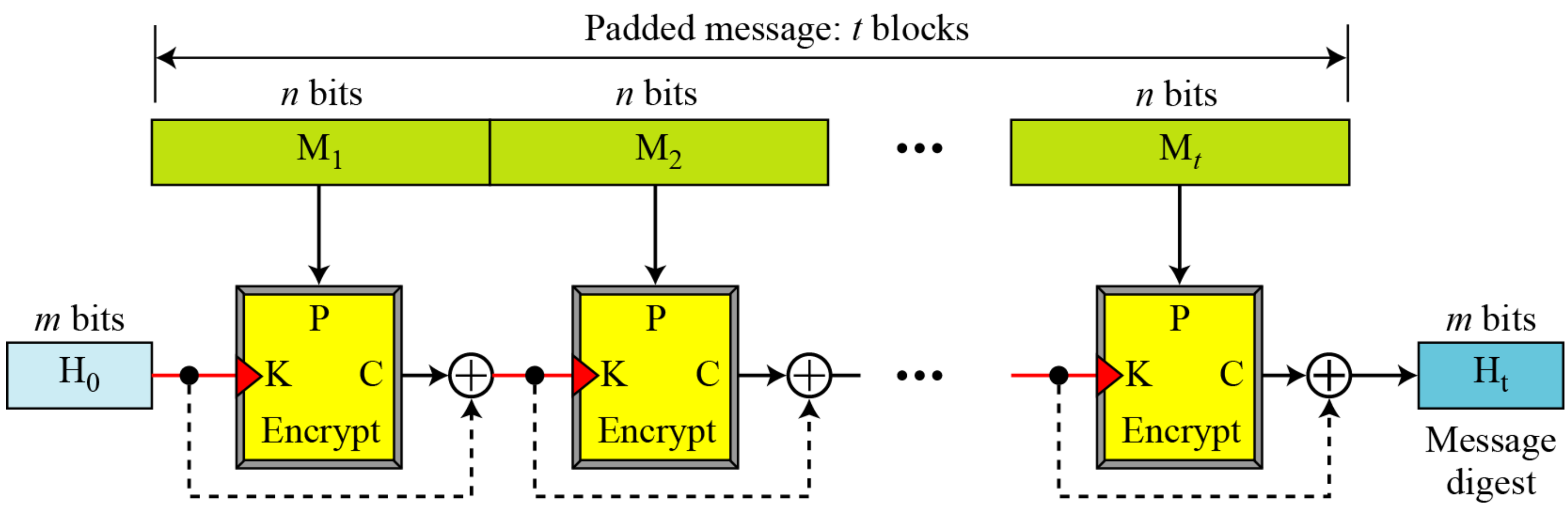
Figure 12.3 ***Davies-Meyer scheme***



12.1.2 Continued

Matyas-Meyer-Oseas Scheme

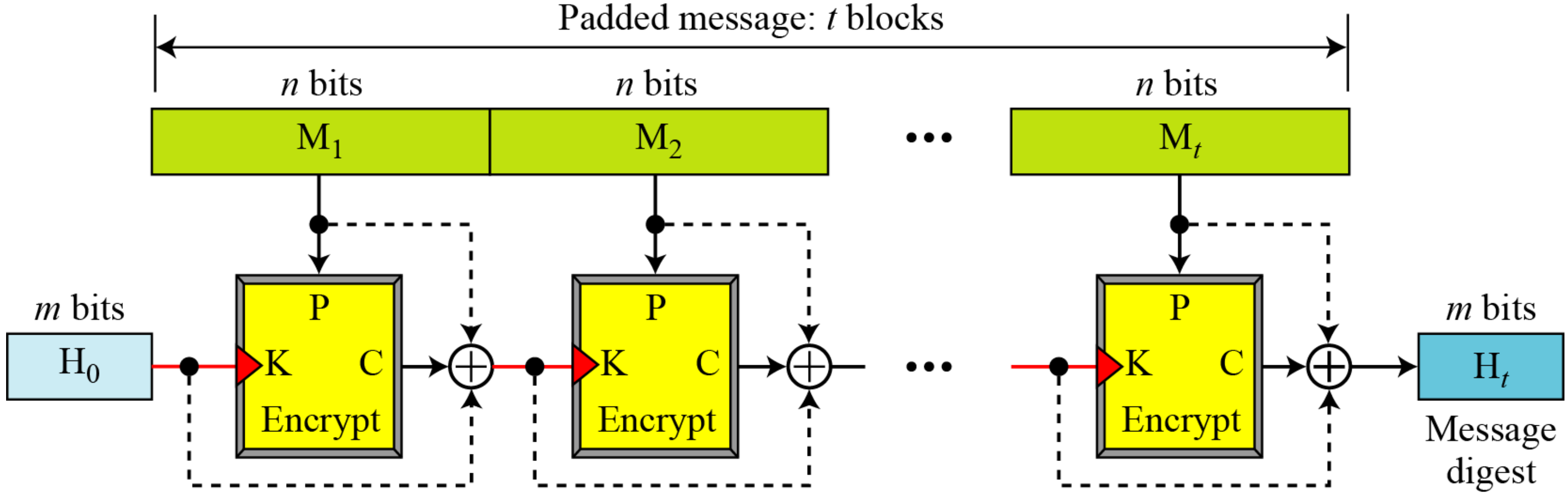
Figure 12.4 ***Matyas-Meyer-Oseas scheme***



12.1.2 Continued

Miyaguchi-Preneel Scheme

Figure 12.5 *Miyaguchi-Preneel scheme*



12-2 SHA-512

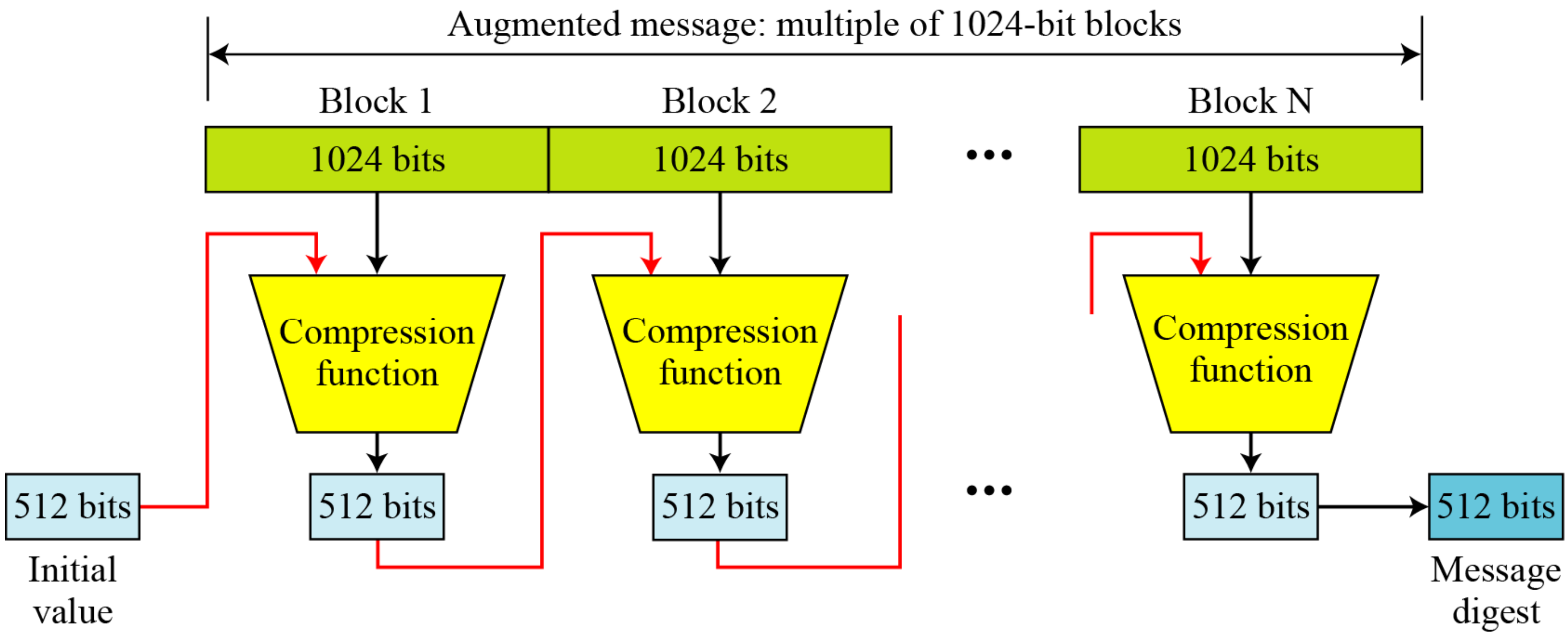
SHA-512 is the version of SHA with a 512-bit message digest. This version, like the others in the SHA family of algorithms, is based on the Merkle-Damgard scheme.

Topics discussed in this section:

12.2.1 Introduction
12.2.2 Compression Function
12.2.3 Analysis

12.2.1 Introduction

Figure 12.6 *Message digest creation SHA-512*



12.2.1 Continued

Message Preparation

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

Note

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

12.2.1 Continued

Example 12.1

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?

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.

12.2.1 Continued

Example 12.2

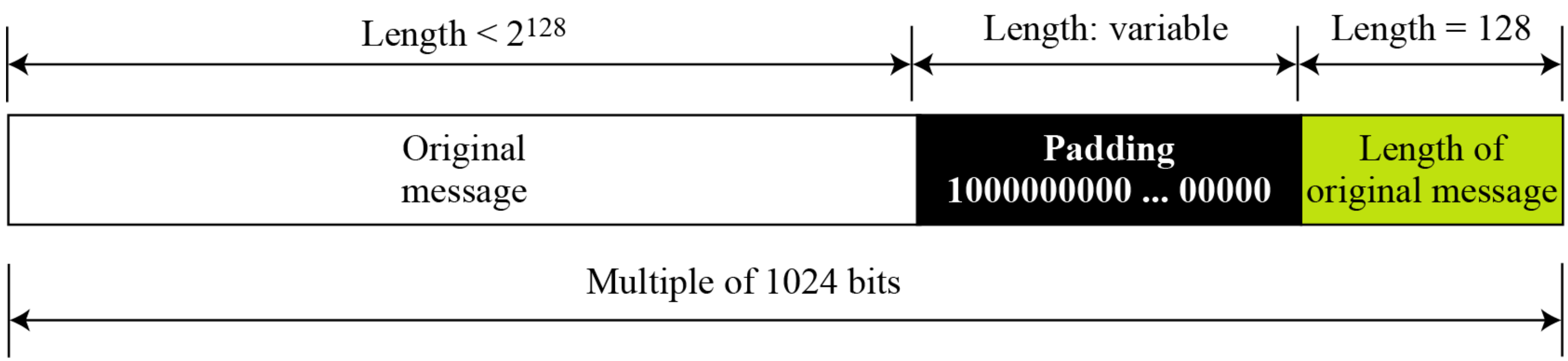
This example also concerns the message length in SHA-512. How many pages are occupied by a message of 2^{128} bits?

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.

12.2.1 Continued

Figure 12.7 ***Padding and length field in SHA-512***



12.2.1 Continued

Example 12.3

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

Solution

We can calculate the number of padding bits as follows:

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

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

12.2.1 Continued

Example 12.4

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

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.

12.2.1 *Continued*

Example 12.5

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

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.

12.2.1 Continued

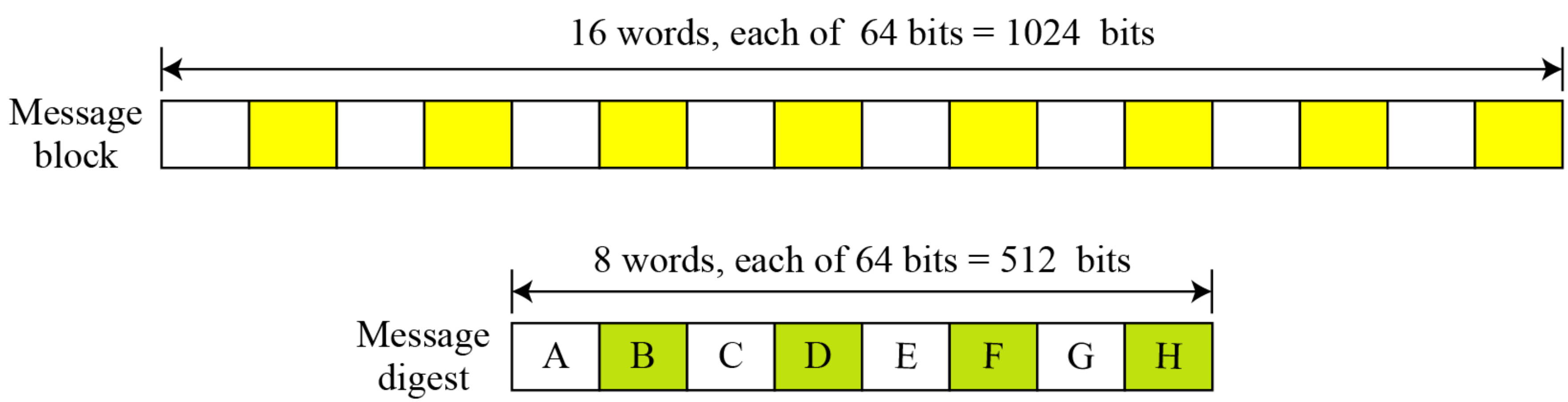
Example 12.5 *Continued*

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.**

12.2.1 Continued

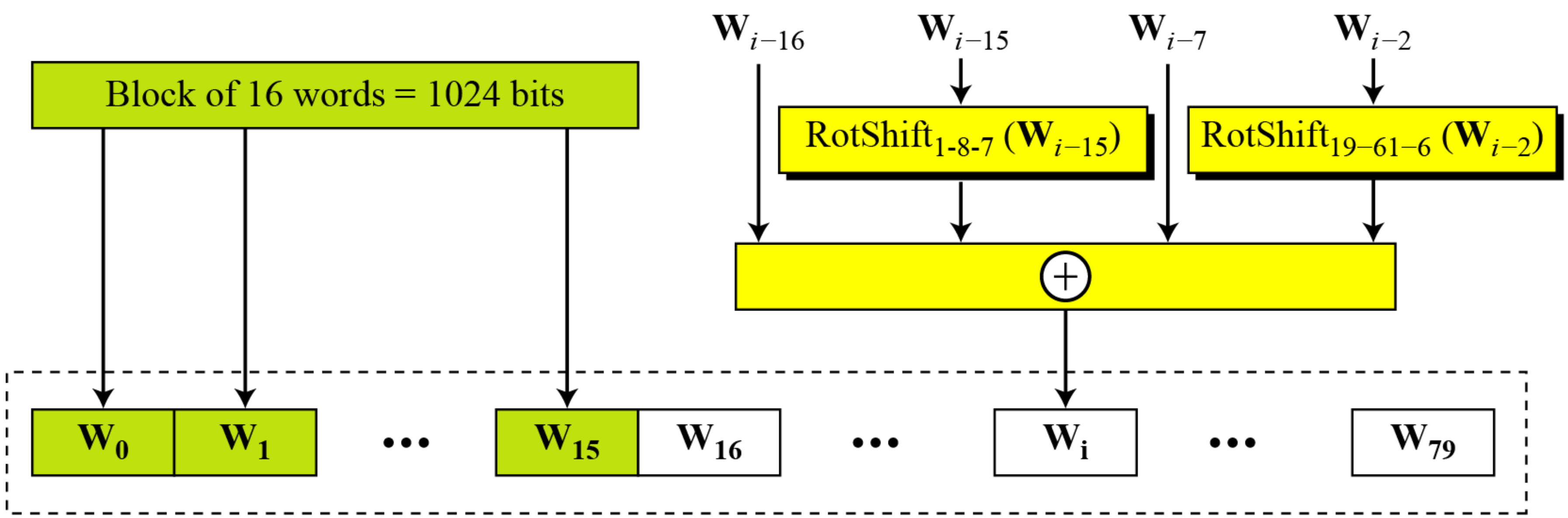
Words

Figure 12.8 *A message block and the digest as words*



Word Expansion

Figure 12.9 *Word expansion in SHA-512*



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

RotR$_i$ (x): Right-rotation of the argument x by i bits
ShL$_i$ (x): Shift-left of the argument x by i bits and padding the left by 0's.

12.2.1 Continued

Example 12.6

Show how W60 is made.

Solution

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

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

12.2.1 Continued

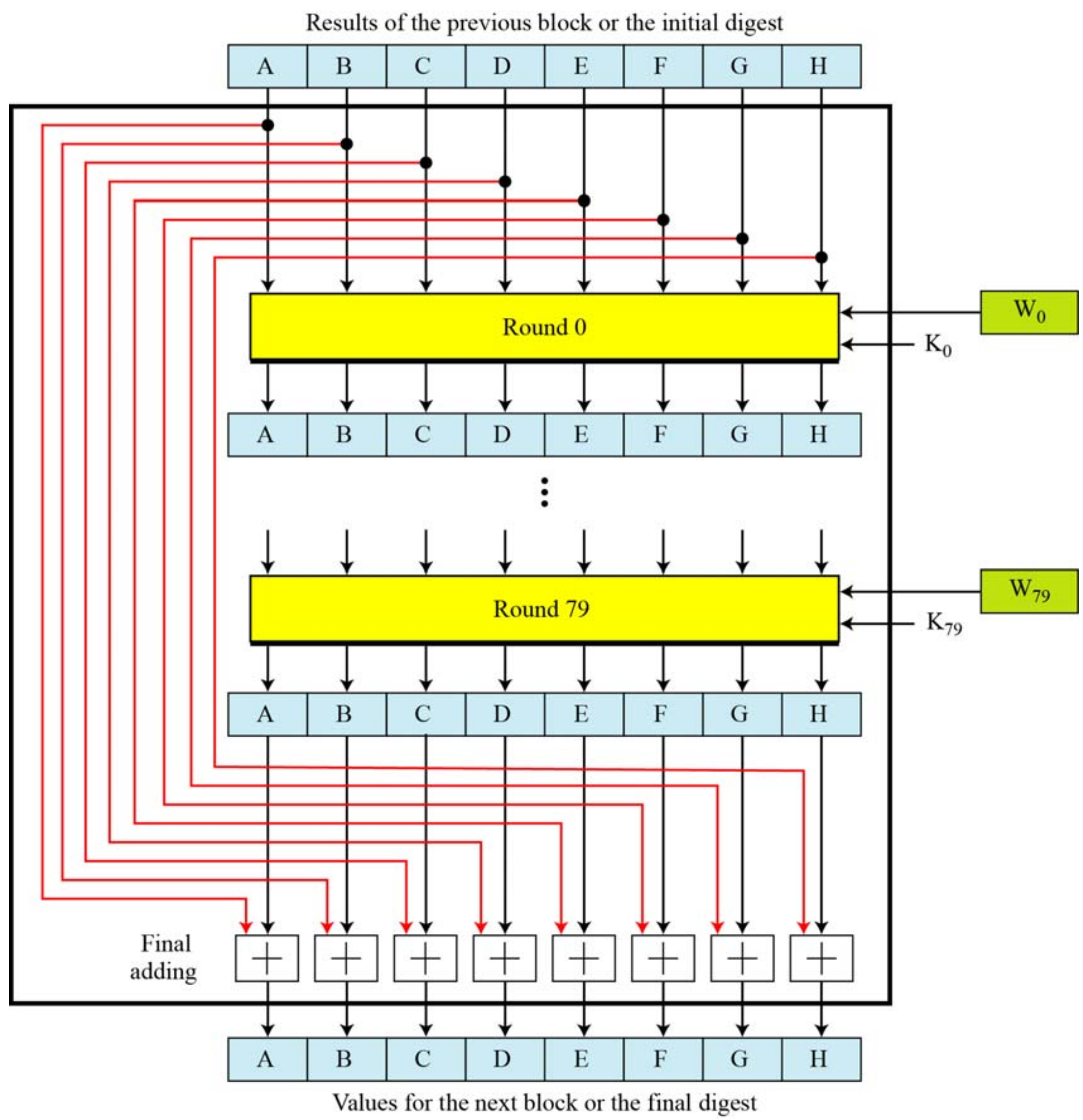
Message Digest Initialization

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

Buffer	*Value (in hexadecimal)*	*Buffer*	*Value (in hexadecimal)*
A_0	**6A09E667F3BCC908**	E_0	**510E527FADE682D1**
B_0	**BB67AE8584CAA73B**	F_0	**9B05688C2B3E6C1F**
C_0	**3C6EF372FE94F82B**	G_0	**1F83D9ABFB41BD6B**
D_0	**A54FF53A5F1D36F1**	H_0	**5BE0CD19137E2179**

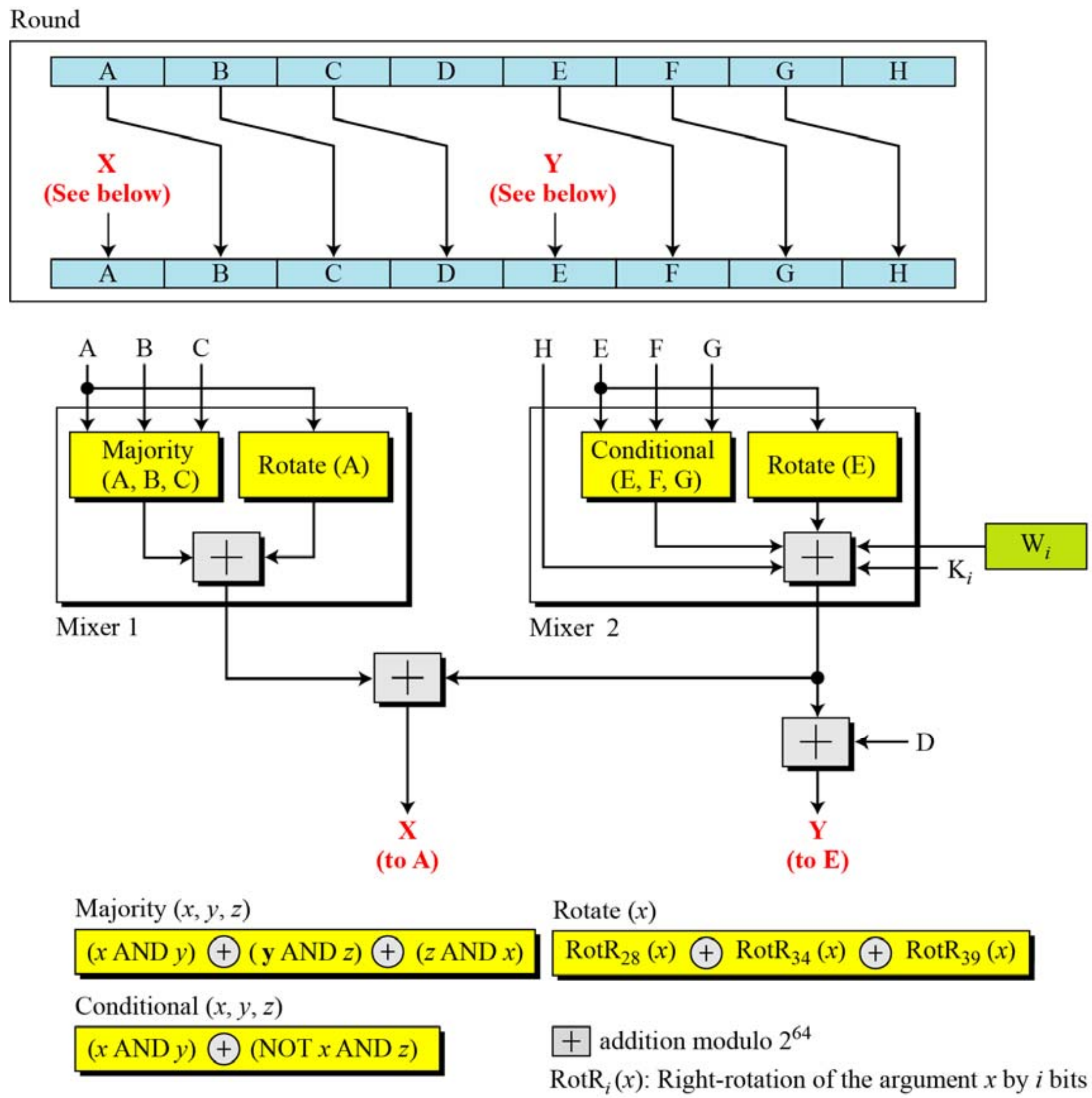
12.2.2 Compression Function

Figure 12.10 *Compression function in SHA-512*



12.2.2 Continued

Figure 12.11 *Structure of each round in SHA-512*



12.2.2 Continued

Majority Function

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

Conditional Function

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

Rotate Functions

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

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

12.2.2 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

12.2.2 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 $(409)^{1/3} = 7.42291412044$. Converting this number to binary with only 64 bits in the fraction part, we get

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

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

12.2.2 Continued

Example 12.7

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.

12.2.2 Continued

Example 12.8

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.

12.2.3 Analysis

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

12-3 WHIRLPOOL

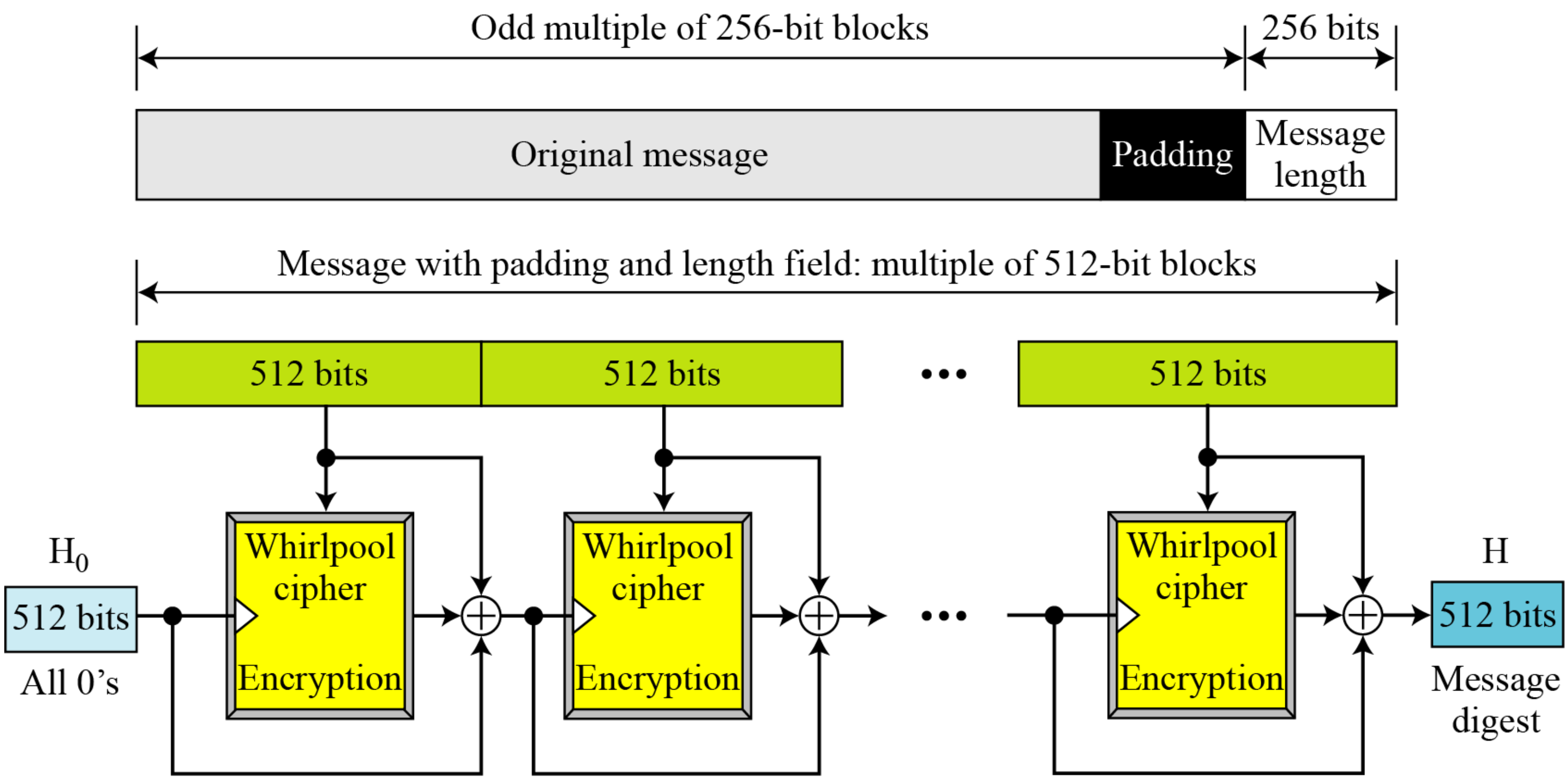
Whirlpool is an iterated cryptographic hash function, based on the Miyaguchi-Preneel scheme, that uses a symmetric-key block cipher in place of the compression function. The block cipher is a modified AES cipher that has been tailored for this purpose.

Topics discussed in this section:

12.3.1 Whirlpool Cipher
12.3.2 Summary
12.3.3 Analysis

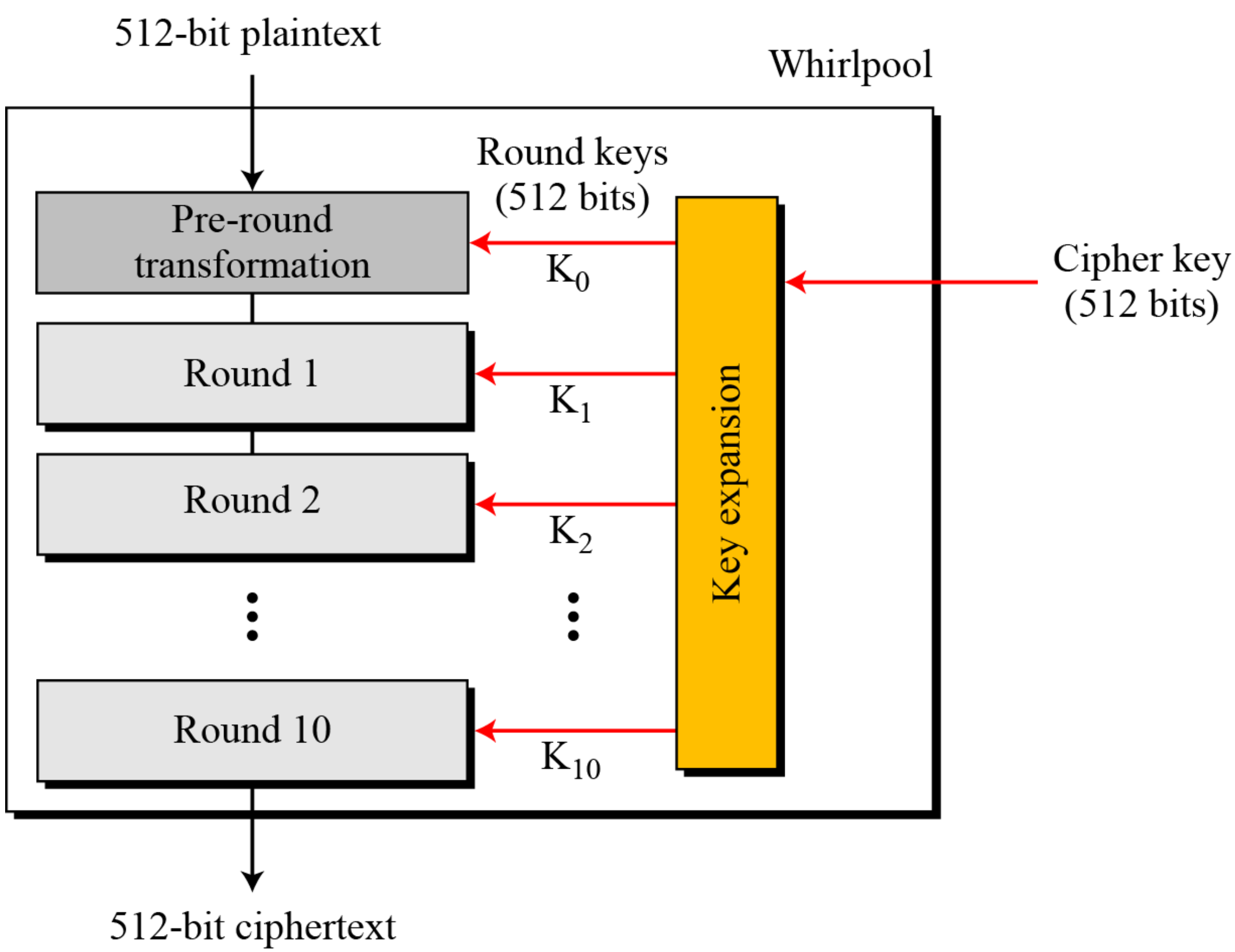
12-3 Continued

Figure 12.12 ***Whirlpool hash function***



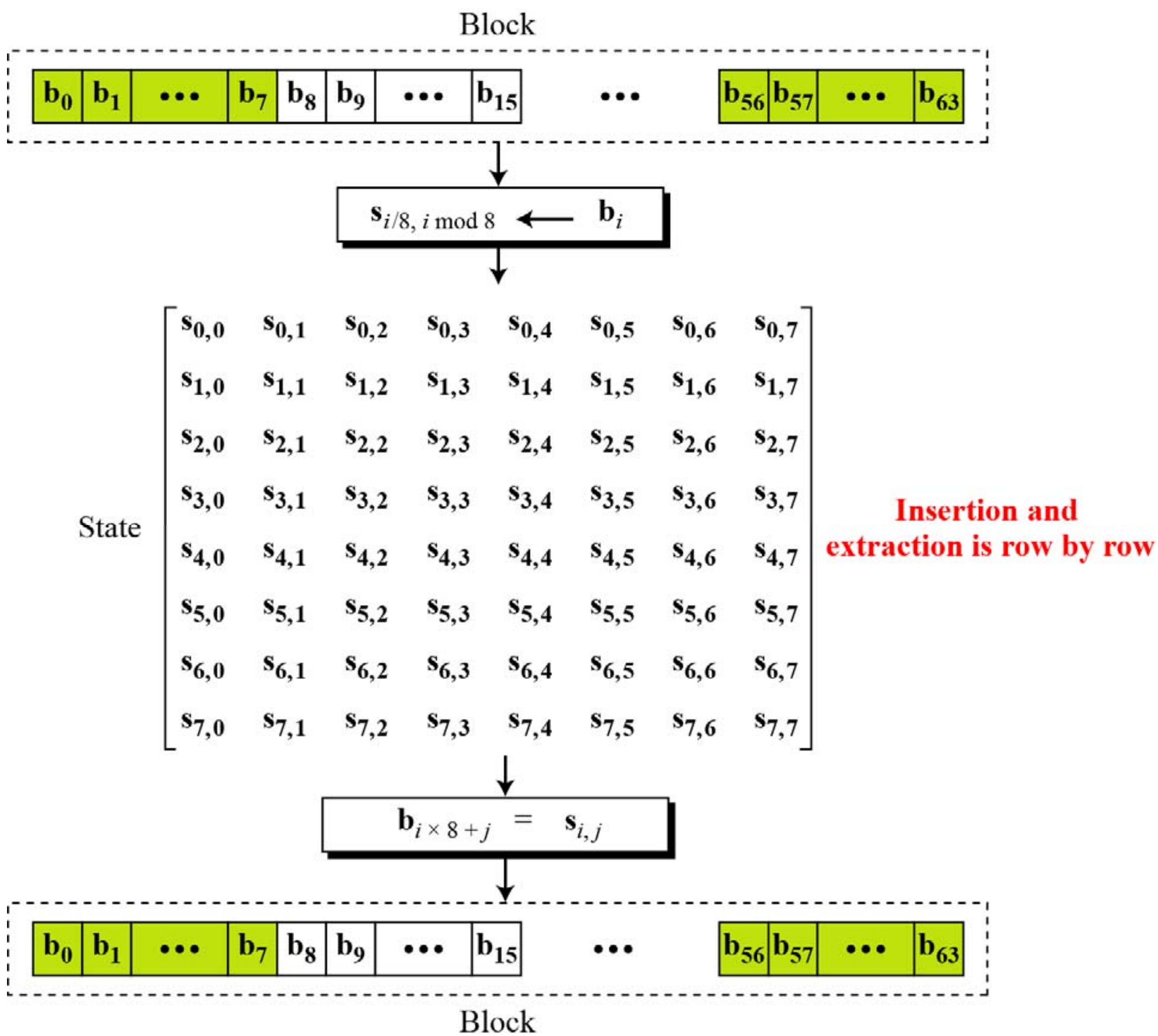
12.3.1 Whirlpool Cipher

Figure 12.13 *General idea of the Whirlpool cipher*



12.3.1 Continued

Figure 12.14 *Block and state in the Whirlpool cipher*



12.3.1 Continued

Structure of Each Round

Each round uses four transformations.

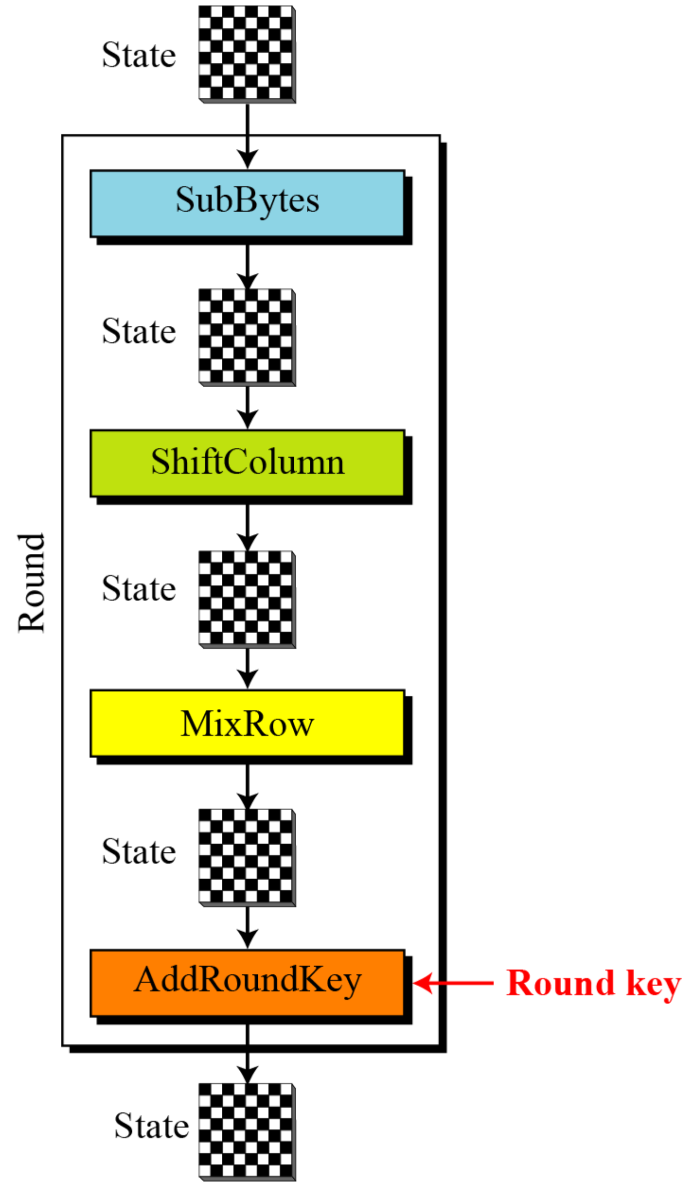


Figure 12.15 ***Structure of each round in the Whirlpool cipher***

12.3.1 Continued

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

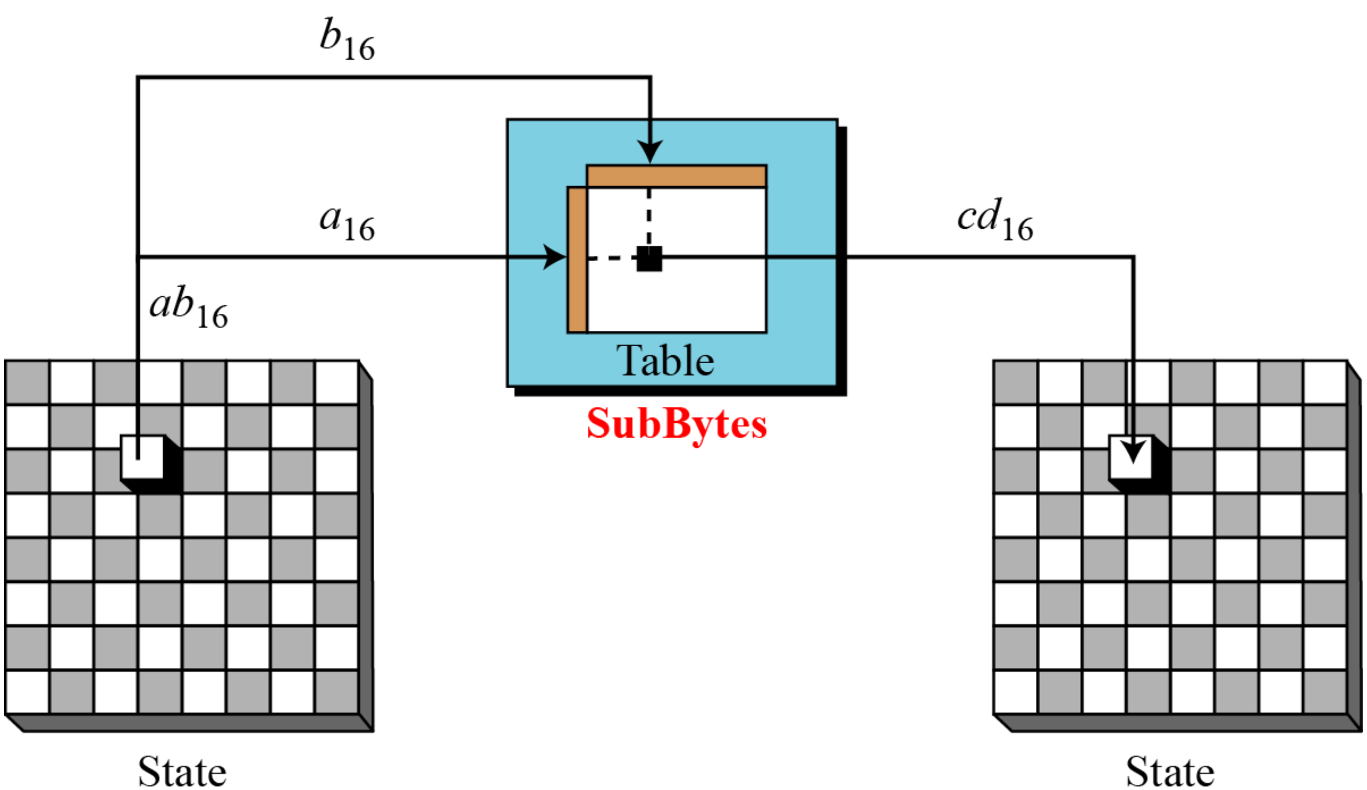


Figure 12.16 *SubBytes transformations in the Whirlpool cipher*

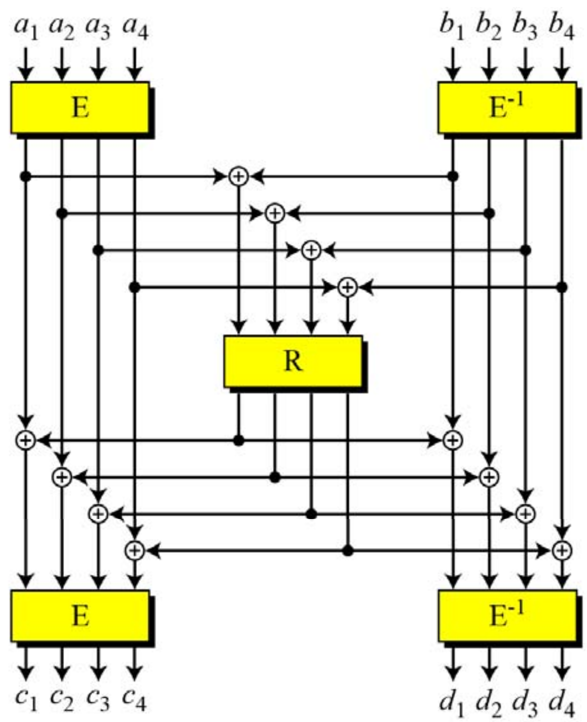
12.3.1 Continued

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

12.3.1 Continued

Figure 12.17 ***SubBytes in the Whirlpool cipher***



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

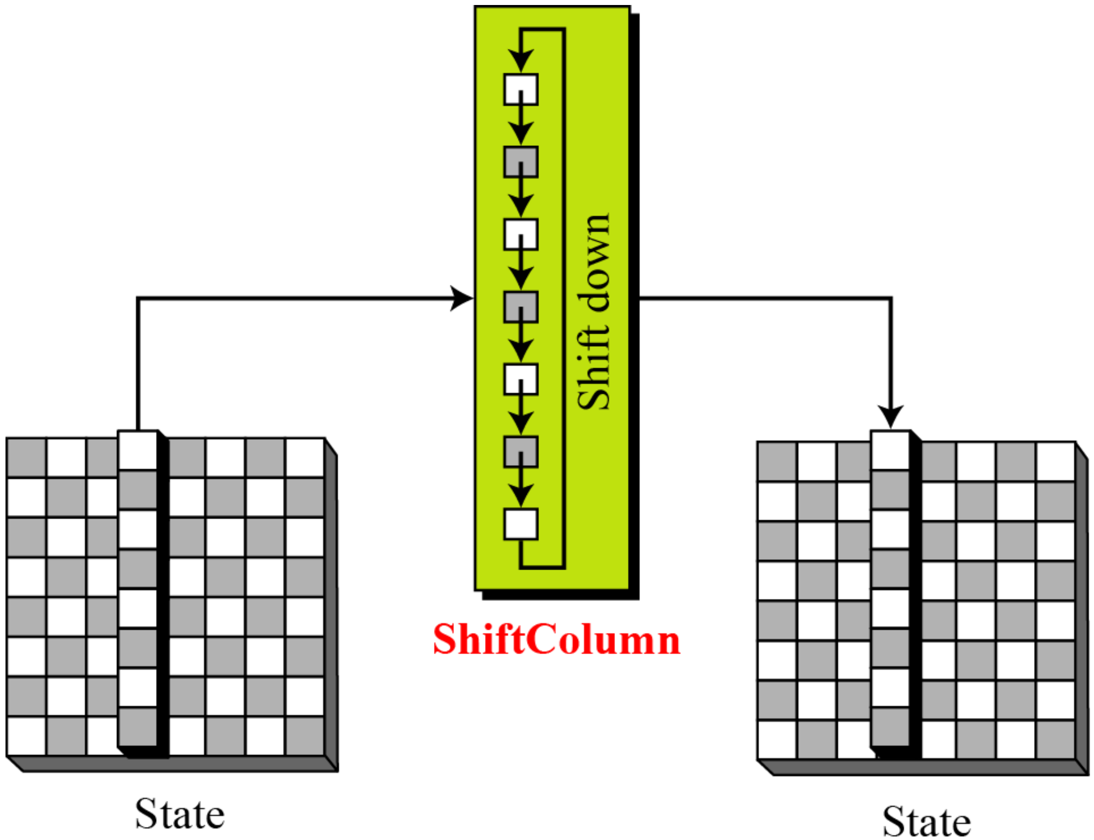
Input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	E^{-1} box
Output	F	0	D	7	B	E	5	A	9	2	C	1	3	4	8	6	

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

12.3.1 Continued

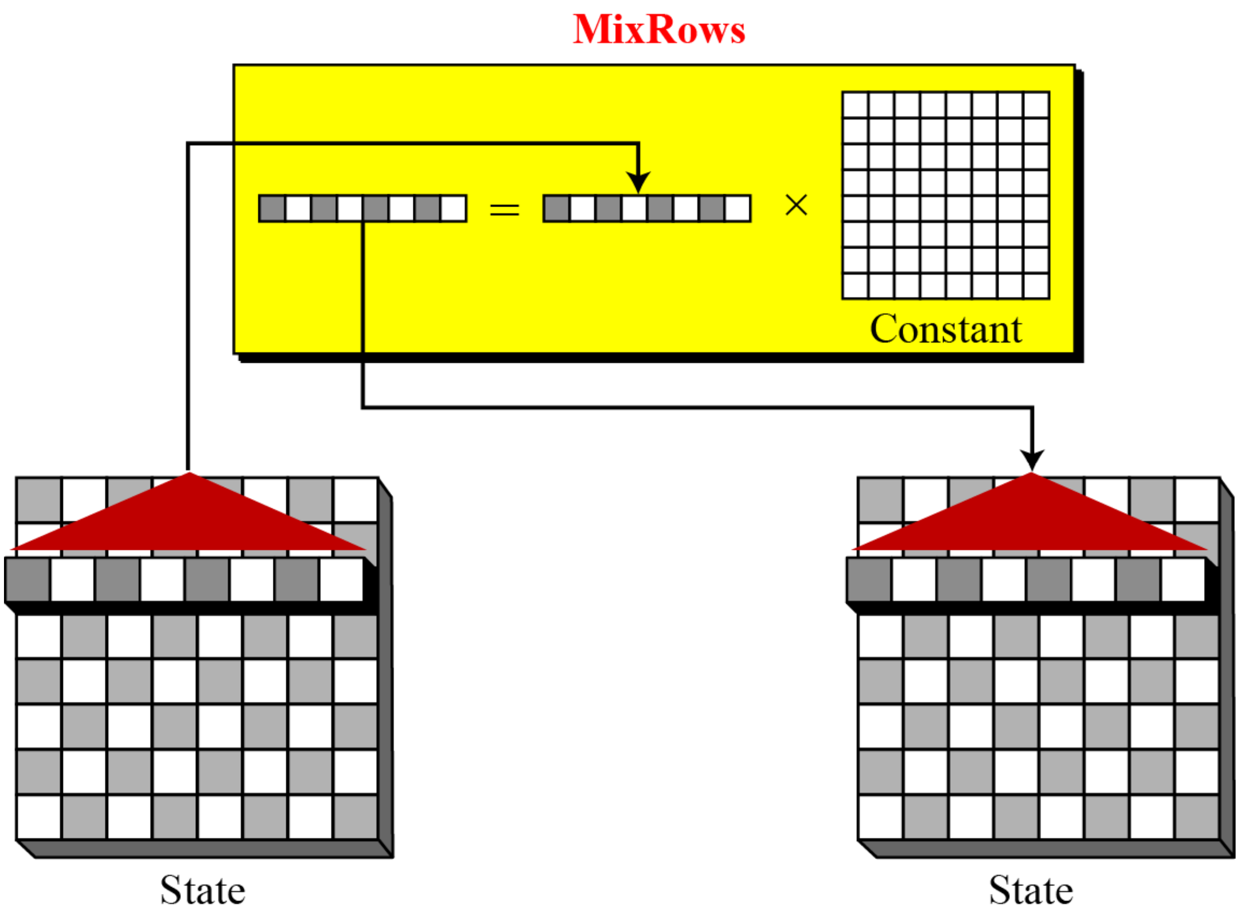
ShiftColumns

Figure 12.18 ***ShiftColumns transformation in the Whirlpool cipher***



12.3.1 Continued

Figure 12.19 *MixRows transformation in the Whirlpool cipher*

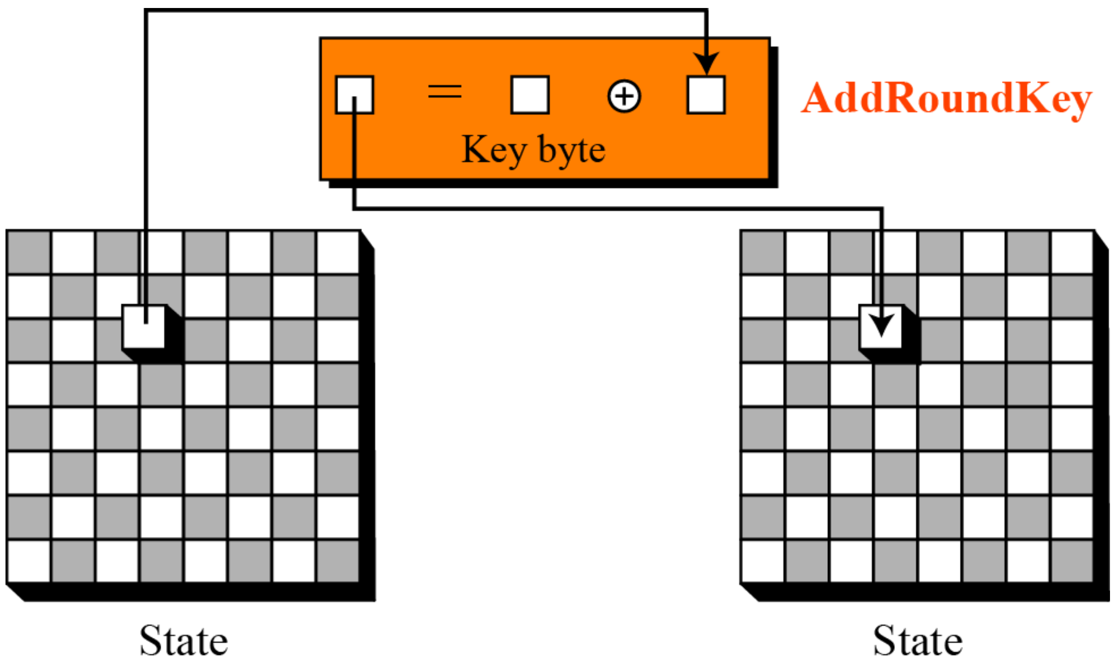


$$
\begin{bmatrix}
01 & 01 & 04 & 01 & 08 & 05 & 02 & 09 \\
09 & 01 & 01 & 04 & 01 & 08 & 05 & 02 \\
02 & 09 & 01 & 01 & 04 & 01 & 08 & 05 \\
05 & 02 & 09 & 01 & 01 & 04 & 01 & 08 \\
08 & 05 & 02 & 09 & 01 & 01 & 04 & 01 \\
01 & 08 & 05 & 02 & 09 & 01 & 01 & 04 \\
04 & 01 & 08 & 05 & 02 & 09 & 01 & 01 \\
01 & 04 & 01 & 08 & 05 & 02 & 09 & 01
\end{bmatrix}
$$

Constant matrix

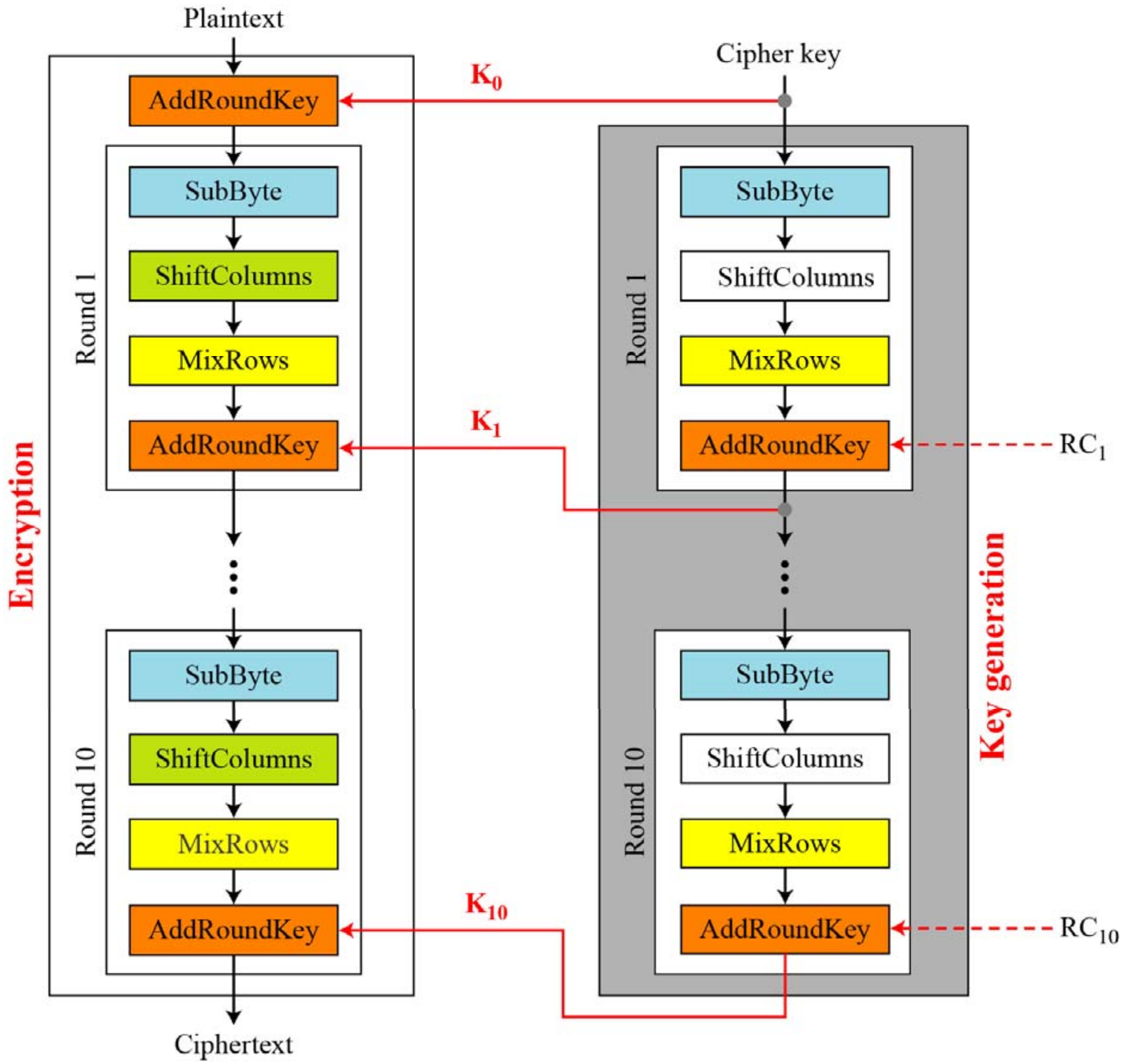
12.3.1 Continued

Figure 12.20 *AddRoundKey transformation in the Whirlpool cipher*



12.3.1 Continued

Figure 12.21 *Key expansion in the Whirlpool cipher*



12.3.1 Continued

Figure 12.22 *Round constant for the third round*

$$RC_3 = \begin{bmatrix} \mathbf{1D} & \mathbf{E0} & \mathbf{D7} & \mathbf{C2} & \mathbf{2E} & \mathbf{4B} & \mathbf{FE} & \mathbf{57} \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 & 00 & 00 & 00 & 00 \end{bmatrix}$$

12.3.2 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

12.3.3 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.