现代密码学

第四十三讲 SHA-1算法

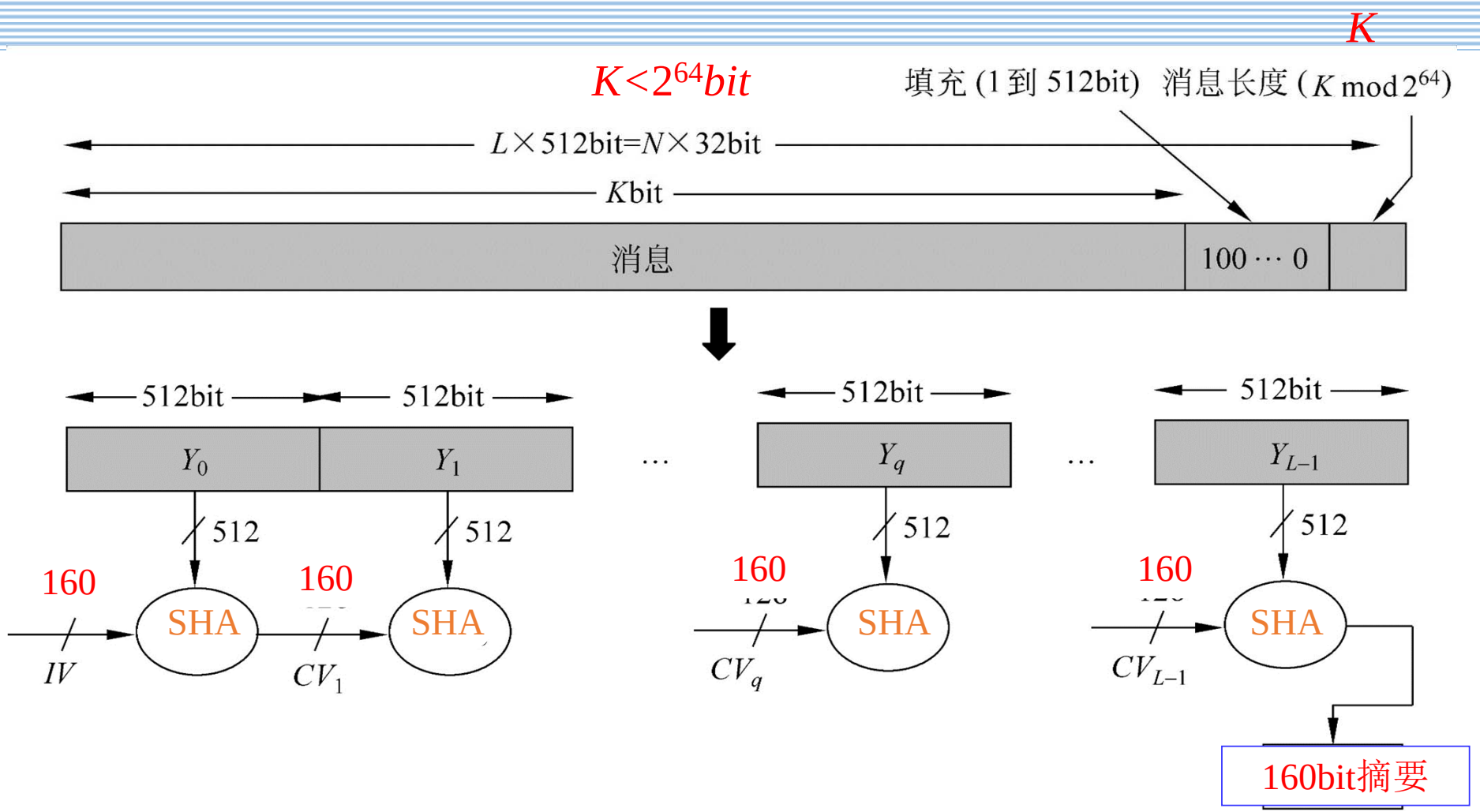
信息与软件工程学院

安全杂凑算法SHA-1

- 安全杂凑算法(Secure Hash Algorithm, SHA)由美国NIST设计，于1993年作为联邦信息处理标准公布。SHA是基于MD4的算法，其结构与MD4非常类似

算法描述

- 算法的输入：小于2^{64}比特长的任意消息，分为512比特长的分组。
- 算法的输出：160比特长的消息摘要。
- 算法的框图与MD5一样，但杂凑值的长度和链接变量的长度为160比特


K
K<2^64bit
填充 (1 到 512bit)
消息长度 (K mod 2^64)
L×512bit=N×32bit
Kbit
消息
100 … 0
512bit
512bit
512bit
512bit
Y0
Y1
Yq
YL–1
512
512
512
512
160
160
160
160
SHA
SHA
SHA
SHA
IV
CV1
CVq
CVL–1
160bit摘要

算法处理步骤---①对消息填充

- 对消息填充，使得其比特长在模512下为448，即填充后消息的长度为512的某一倍数减64，留出的64比特备第② 步使用。
- 步骤①是必需的，即使消息长度已满足要求，仍需填充。例如，消息长为448比特，则需填充512比特，使其长度变为960，因此填充的比特数大于等于1而小于等于512。
- 填充方式是：第1位为1，其后各位皆为0。

算法处理步骤--- ②附加消息的长度

- 留出的64比特以来表示消息被填充前的长度。如果消息长度大于2^{64}，则以2^{64}为模数取模。

- big-endian模式是指数据的高字节保存在内存的低地址中，反之为little-endian模式

算法处理步骤--③对MD缓冲区初始化

- 使用160比特长的缓冲区存储中间结果和最终杂凑值。

缓冲区为5个32比特以big-endian方式存储数据的寄存器(A，B，C，D，E)

初始值分别为A=67452301，

B=EFCDAB89，

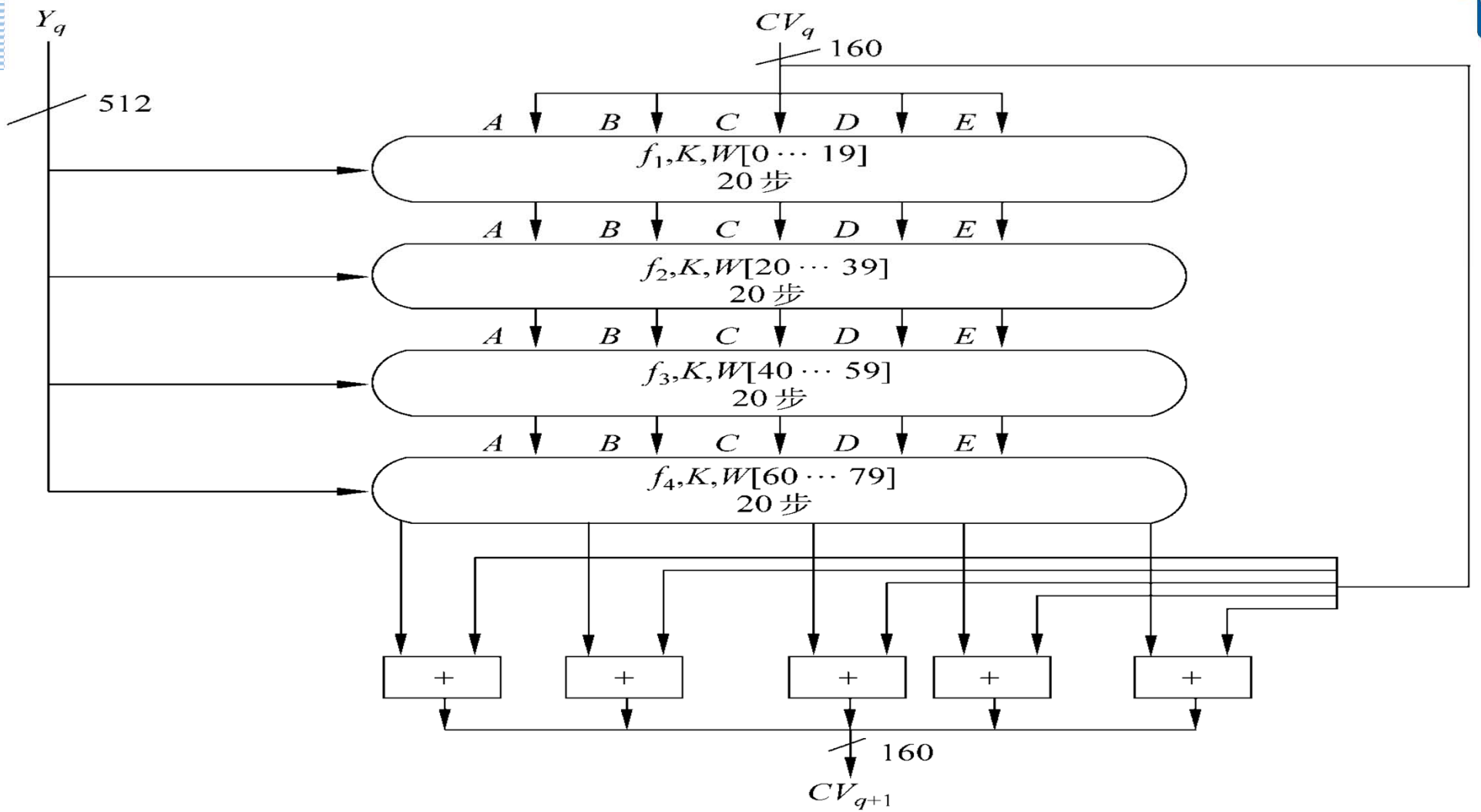
C=98BADCFB，

D=10325476，

E=C3D2E1F0。

算法处理步骤--④以分组为单位对消息进行处理

- 每一分组Y_q都经一压缩函数处理，压缩函数由4轮处理过程（如图所示）构成，每一轮又由20步迭代组成。
- 第4轮的输出（即第80步迭代的输出）再与第1轮的输入CV_q相加，以产生CV_{q+1}，其中加法是缓冲区5个字中的每一个字与CV_q中相应的字模2^{32}相加。



5.8 SHA的分组处理框图

算法处理步骤--⑤输出消息

- 输出消息的L个分组都被处理完后，最后一个分组的输出即为160比特的消息摘要。

总结---③到⑤的处理过程

$$CV_0=IV;$$

$$CV_{q+1}=SUM_{32}(CV_q,\ ABCDE_q);$$

$$MD=CV_L$$

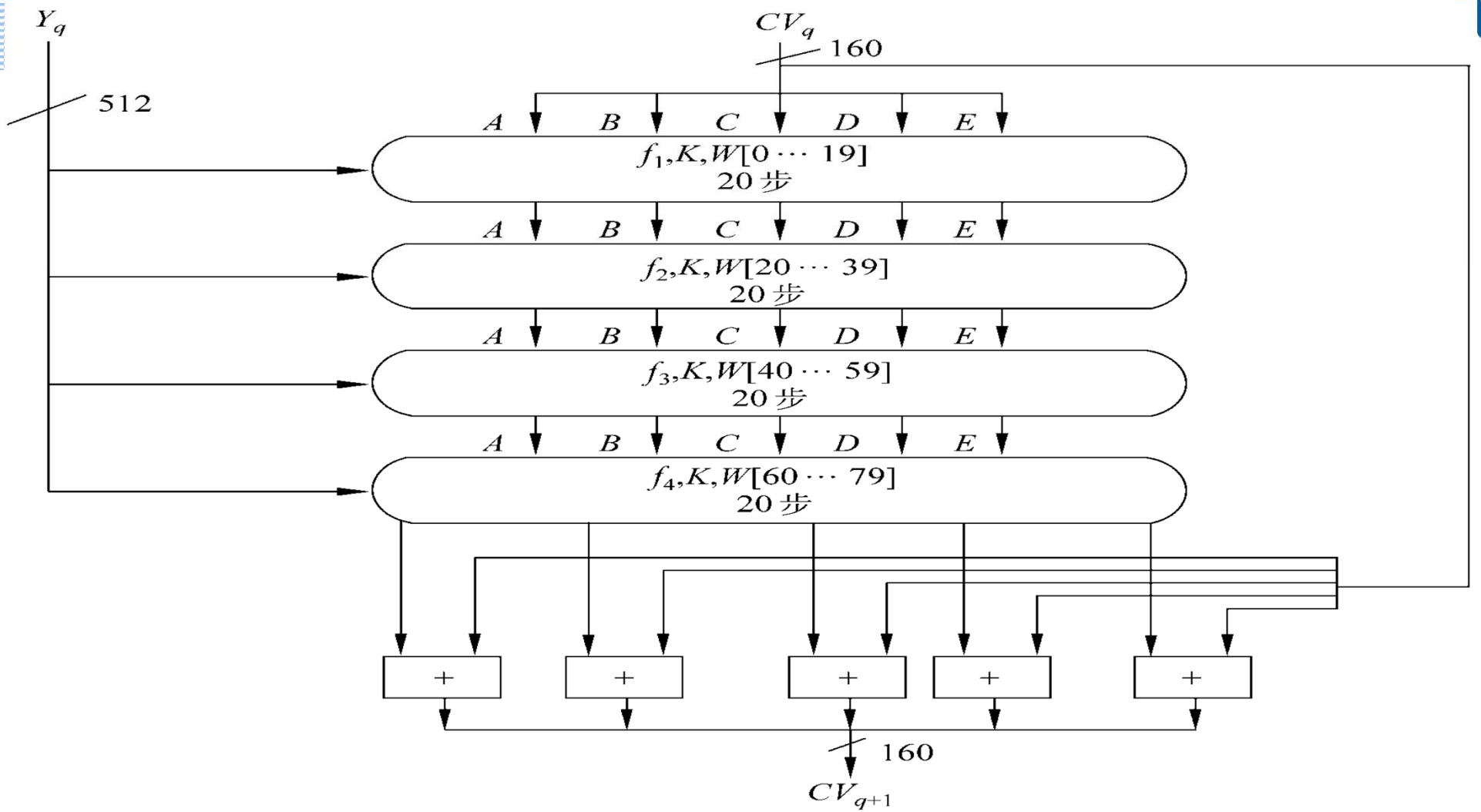
IV：缓冲区ABCDE的初值。

$ABCDE_q$：第q个消息分组经最后一轮处理过程处 理后的输出

L：消息(包括填充位和长度字段)的分组数

SUM_{32}：对应字的模2^{32}加法

MD：最终的摘要值。



5.8 SHA的分组处理框图

SHA的压缩函数

- SHA的压缩函数由4轮处理过程组成，每轮处理过程20步迭代运算组成，每一步迭代运算的形式为

$$A,B,C,D,E \leftarrow (E + f_t(B,C,D) + CLS_5(A) + W_t + K_t),\ A,\ CLS_{30}(B),\ C,\ D$$

A，B，C，D，E：缓冲区的5个字

t：迭代的步数（0≤t≤79）

$f_t(B,C,D)$：第t步迭代使用的基本逻辑函数

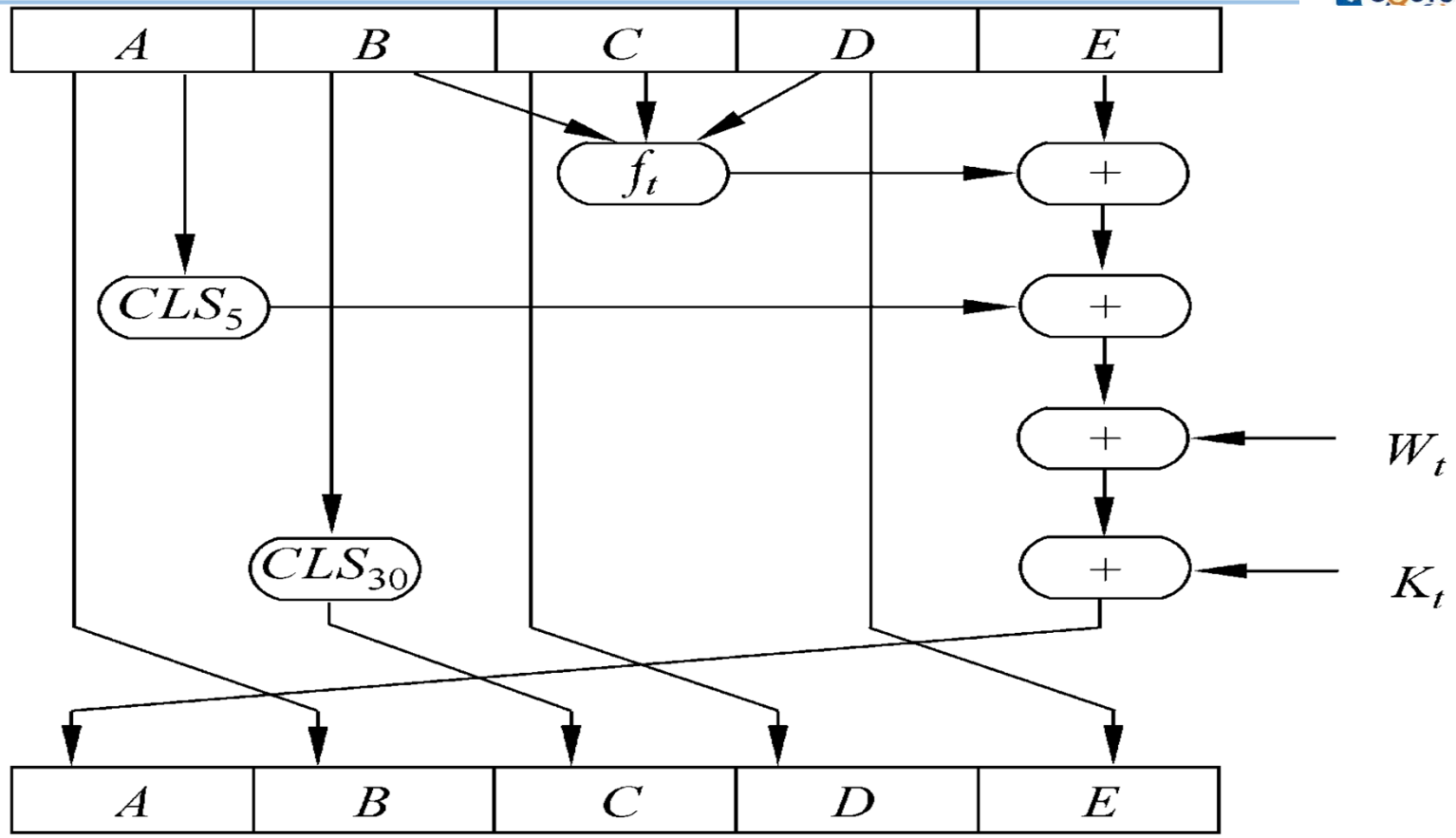
CLSs：左循环移s位

W_t：由当前512比特长的分组导出的一个32比特长的字

K_t：加法常量

+：模2^{32}加法。

一步迭代示意图



$$A, B, C, D, E \leftarrow (E + f_t(B, C, D) + CLS_5(A) + W_t + K_t),\ A,\ CLS_{30}(B),\ C,\ D$$

基本逻辑函数f_t

迭代的步数	函数名	定义
$0 \le t \le 19$	$f_1 = f_t(B,C,D)$	$(B \wedge C) \vee (\overline{B} \wedge D)$
$20 \le t \le 39$	$f_2 = f_t(B,C,D)$	$B \oplus C \oplus D$
$40 \le t \le 59$	$f_3 = f_t(B,C,D)$	$(B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$
$60 \le t \le 79$	$f_4 = f_t(B,C,D)$	$B \oplus C \oplus D$

- 基本逻辑函数的输入为3个32比特的字，输出是一个32比特的字。表中$\wedge$，$\vee$，$-$，　　分别是与、或、非、异或4个逻辑运算。

字W_t

- 下面说明如何由当前的输入分组（512比特长）导出$W_0, W_1, \ldots, W_{15}, W_{16}, W_{17}, \ldots, W_{79}$）

$$W_t = CLS_1(W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3})$$

$$W_t = CLS_1(W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3})$$

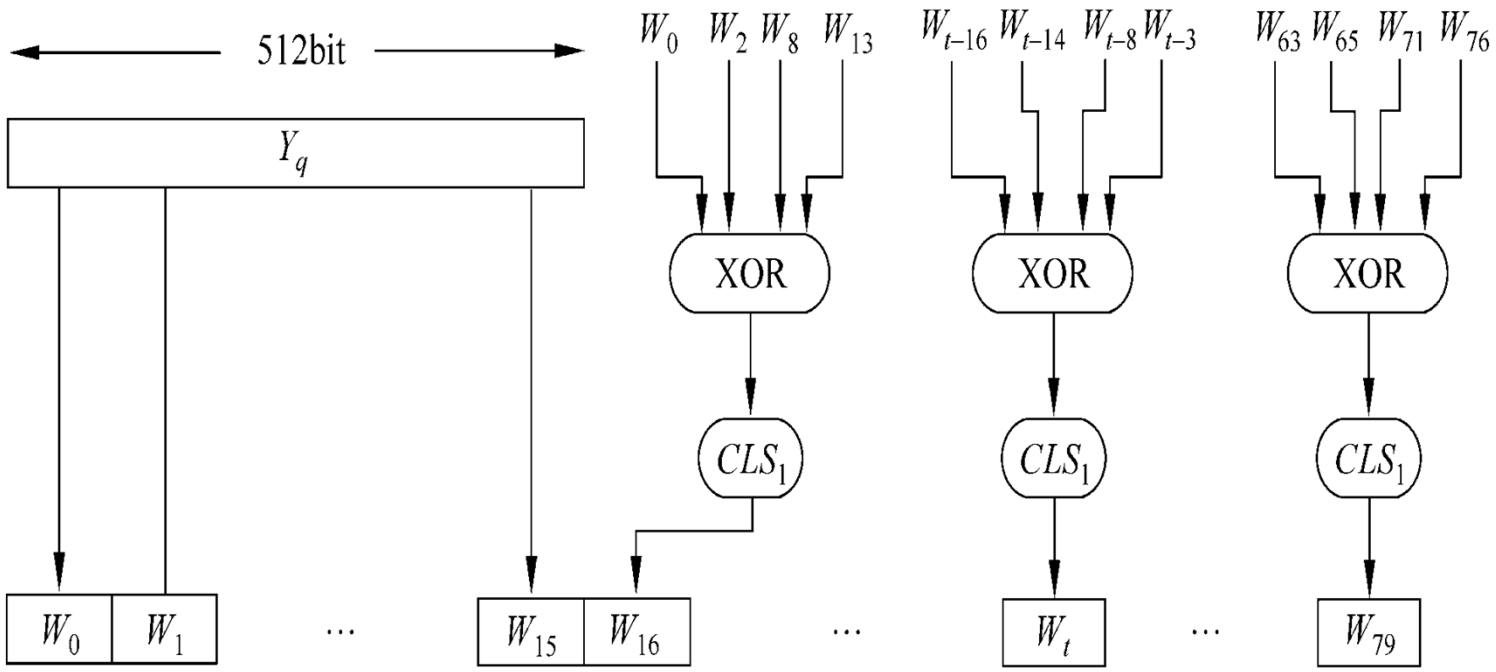


图 SHA分组处理所需的80个字的产生过程

常量字K_t

- 常量字K_0, K_1,...,K_{79}, 如果以16进制给出。它们如下：

$$K_t = 0x5A827999 \quad (0 <= t <= 19)$$
$$K_t = 0x6ED9EBA1 \quad (20 <= t <= 39)$$
$$K_t = 0x8F1BBCDC \quad (40 <= t <= 59)$$
$$K_t = 0xCA62C1D6 \quad (60 <= t <= 79).$$