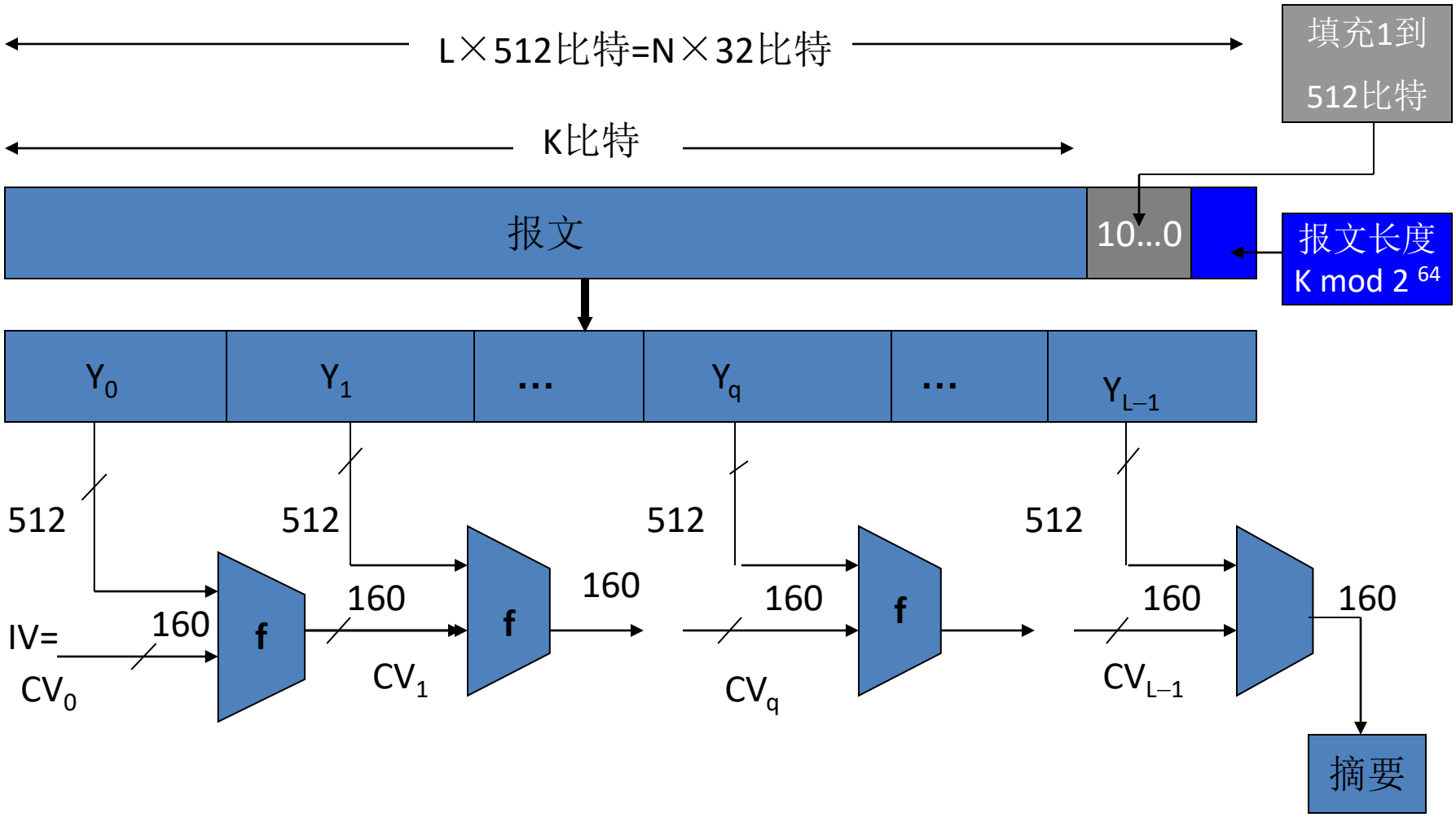
SHA1

- 概述
  - SHA: Secure Hash Algorithm——安全杂凑函数
  - 美国国家标准和技术协会(NIST)提出，93年发布，95年修订
  - 符合Merkle-Damgard结构，基于和模仿MD4
  - 输入任意长度报文，输出160比特的摘要
  - 输入分组长度为512比特，处理过程与MD5相似

SHA1产生报文摘要的过程



SHA1操作过程

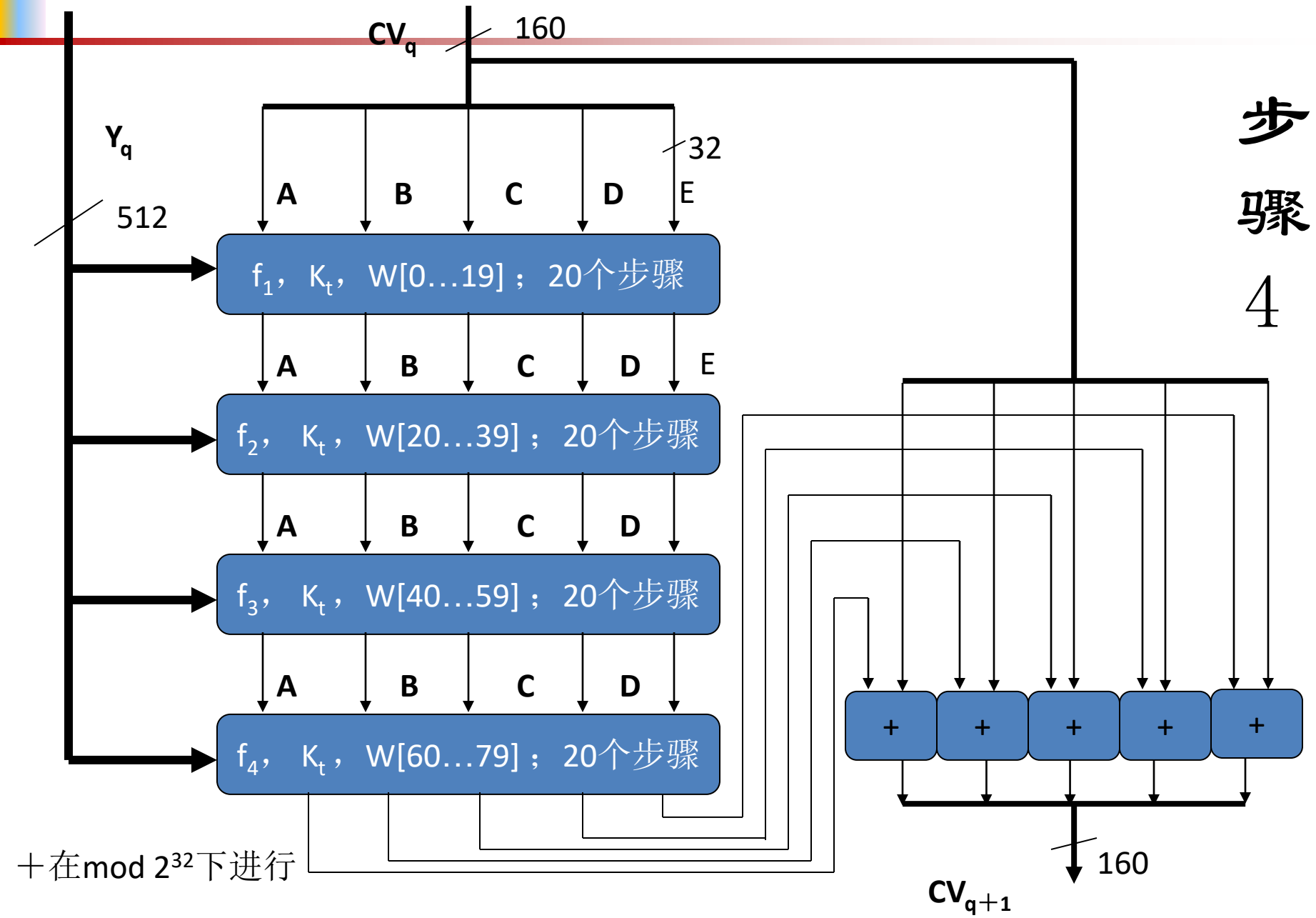
- 步骤1：附加填充比特。对报文进行填充，使得填充后的比特数与448 mod 512同余。例如，如果报文是448比特，则填充512比特形成960比特的报文。填充比特首位为1，其余位全0。
- 步骤2：附加报文长度值。用64比特表示初始报文的长度，长度值的高位优先，然后附加在步骤1填充的结果之后。

SHA1操作过程

- 步骤3：初始化MD缓存。使用一个160比特缓存存放杂凑的中间和最后结果。缓存表示为5个32比特的缓存器(A,B,C,D,E)
  - 初始化值(16进制表示)：A=67452301, B=EFCDAB89, C=98BADCFE, D=10325476, E=C3D2E1F0.
  - 初始化格式：小数在前的格式存储，即字的低位字节放在高地址字节上，像32位的比特串：A: 01 23 45 67, B: 89 AB CD EF, C: FE DC BA 98, D: 76 54 32 10, E: F0 E1 D2 C3

SHA1操作过程

- 步骤4：处理512比特(16个字)报文分组。核心是一个包含4个循环的压缩函数f，由20个步骤组成。4个循环结构相似，但每次使用的原始逻辑函数不同，分别记为f_1，f_2，f_3，f_4。
  - 输入：当前处理的512分组(Y_q)和160比特缓存值ABCDE(即上一次迭代的输出CV_q)。
  - 循环：每次循环分别使用一个额外的常数K_t，即$[2^{30}\times\sqrt{2}]$(0～19)，$[2^{30}\times\sqrt{3}]$(20～39)，$[2^{30}\times\sqrt{5}]$(40～59)，$[2^{30}\times\sqrt{10}]$(50～79)，这里有个取整运算。
  - 输出：第4次循环输出加到第1次循环的输入上产生CV_{q+1}。相加是缓存中5个字分别与CV_q中对应的5个字以模2^{32}相加。



步骤4

＋在mod 2^{32}下进行

SHA1操作过程

- 步骤5：输出。所有L个512比特的分组处理完成后，第L个阶段的输出作为报文的160比特摘要。总结MD5操作如下：
  - $CV0=IV$
  - $CV_{q+1}= CV_q + SUM_{32}(CV_q, ABCDE_q)$
  - $SHA1=CV_L$

IV=缓存ABCDE的初值，第3步定义

$ABCDE_q$ =第q报文分组最后一次循环输出

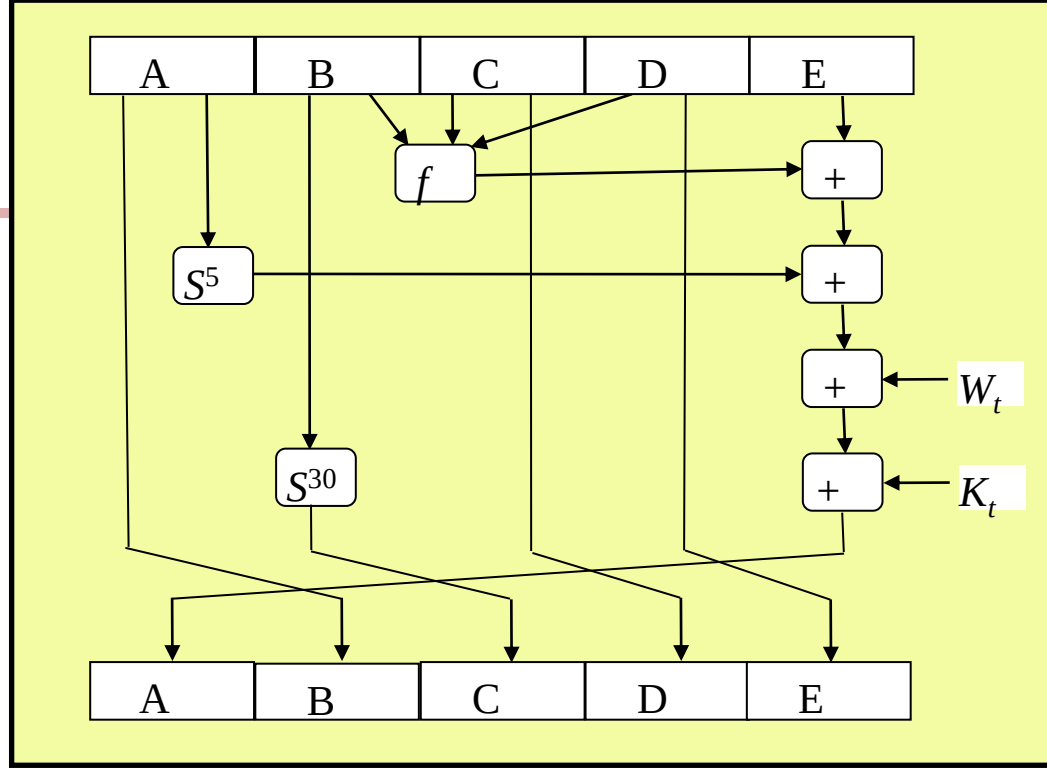
L=报文的分组数

SUM_{32} =对输入对中的每个字执行模2^{32}加

SHA1=最终的报文摘要

SHA1的压缩函数f

- f由4个循环组成
- 每个循环包括20步操作
- 每一步的基本形式如下



$$A,B,C,D,\ E \leftarrow (E+f(B,C,D)+S^5(A)+W_t+K_t),A,S^{30}(B),C,D$$

A,B,C,D,E =缓存中的5个字　　S^i ＝32比特常数循环左移i位

f(t,B,C,D) =步骤t的原始逻辑函数　　t =步骤数；0到79

K_t = 一个额外的常数，前面有定义

W_t ＝当前512比特输入报文分组导出的一个32比特字

＋ =在mod 2^{32}下进行的加法

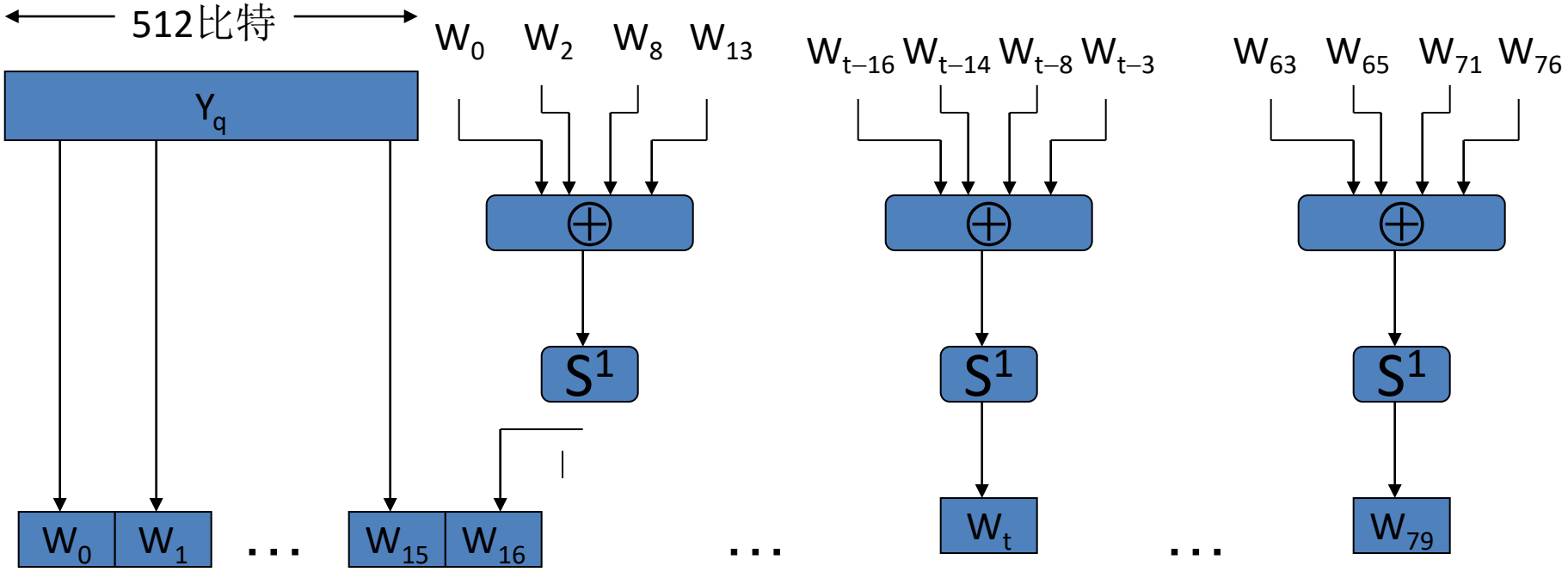
SHA1的压缩函数f

- 原始逻辑函数

循环	原始函数	函数值
1	f_1=F(t,B,C,D)	$(B\wedge C)\vee(\neg B\wedge D)$
2	f_2= F(t,B,C,D)	$B\oplus C\oplus D$
3	f_3= F(t,B,C,D)	$(B\wedge C)\vee (B\wedge D)\vee (C\wedge D)$
4	f_4= F(t,B,C,D)	$B\oplus C\oplus D$

SHA1的压缩函数f

- W_t的定义；由512比特报文分组导出32比特字， W_t的前16个字直接取自当前分组中的16个字，余下的字定义为$W_t=S^1(W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3})$



MAC码

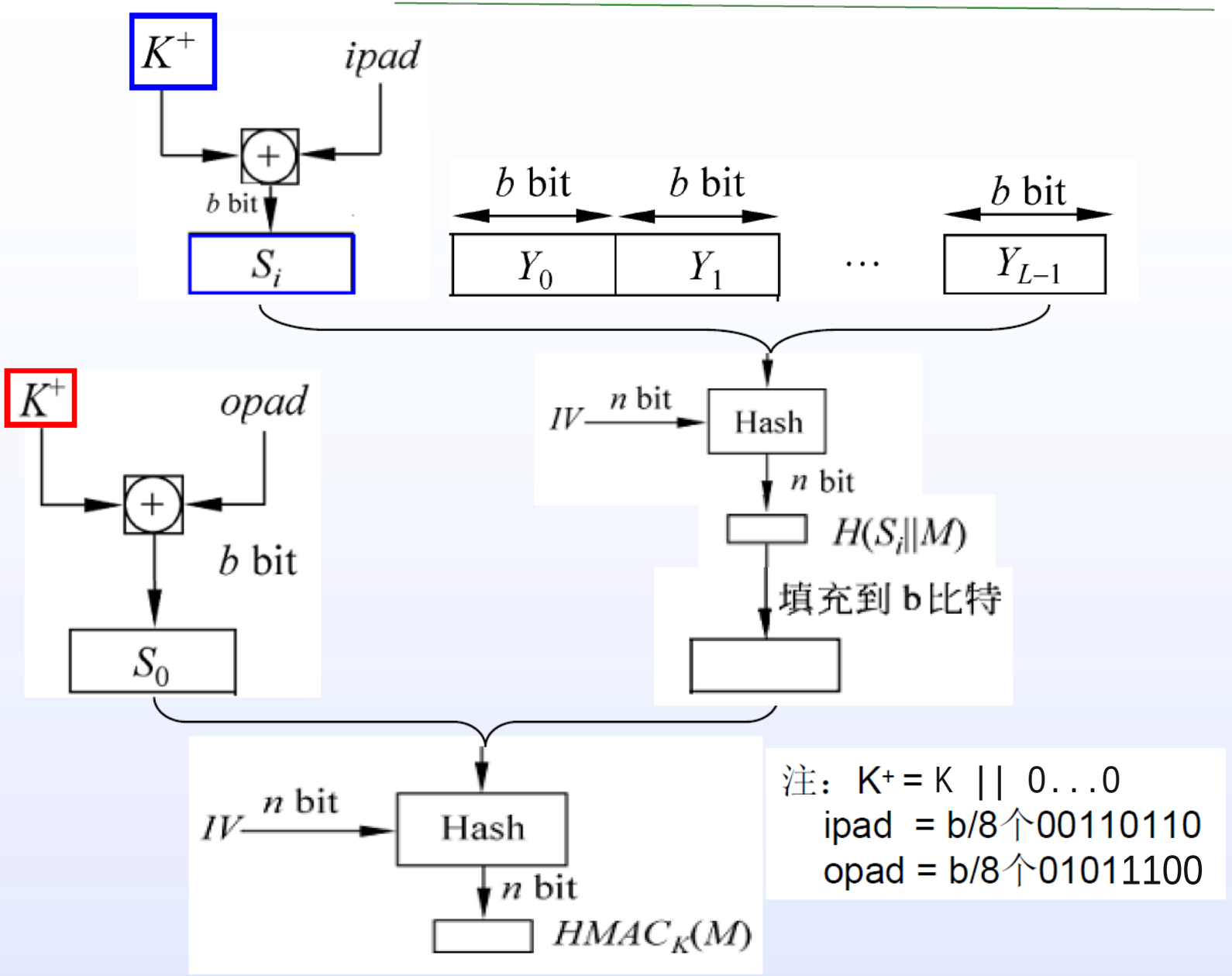
- MAC：Message Authentication Code报文鉴别码
- 安全目标：确认报文在发送过程中没有被篡改过，确认报文是正确的发送者发送
- 实现：一种编码方法，输入报文和一个密钥，输出一个摘要，发送报文和摘要给收端
- 验证：收端可以验证(比如共享密钥)摘要的正确性

HMAC码

- 用杂凑函数实现MAC码
  - HMAC=H[(K⁺⊕opad)||H((K⁺⊕ipad)||M)]
    1. K的右边填充0以产生一个b比特长的K⁺
    2. K⁺与ipad逐比特异或产生一个b比特长的分组S_i，ipad为b/8个00110110
    3. 将M链接到S_i后面
    4. 将H作用于3产生的数据流
    5. K⁺与opad逐比特异或产生一个b比特长的分组S_0，opad为b/8个01011100
    6. 将步骤4产生的数据流链接到S_0后面
    7. 将H作用于步骤6产生的数据流并输出结果

$$HMAC_K(M) = H[\ (K^+ \oplus opad)\ \|\ H[(K^+ \oplus ipad)\|M]\ ]$$

HMAC 算法框图



注：K+ = K || 0...0
ipad = b/8个00110110
opad = b/8个01011100