

《现代密码学》第四讲

分组密码 (三)

上讲内容回顾

- DES算法的整体结构——Feistel结构
- DES算法的轮函数
- DES算法的密钥编排算法
- DES算法的解密变换

本节主要内容

- AES算法的整体结构
- AES算法的轮函数
- AES算法的密钥编排算法
- AES算法的解密变换

本节主要内容

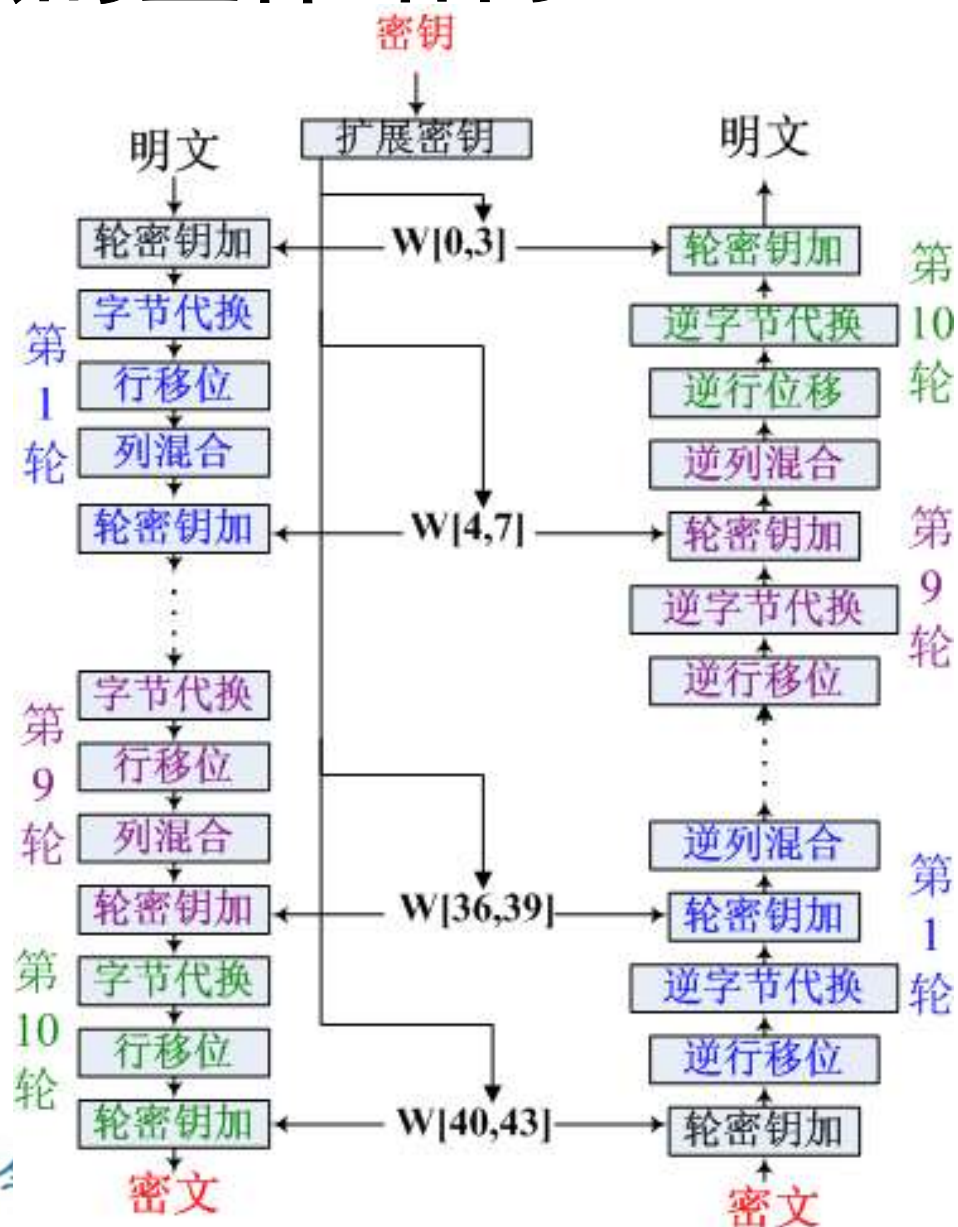
- AES算法的整体结构
- AES算法的轮函数
- AES算法的密钥编排算法
- AES算法的解密变换

AES算法的整体结构

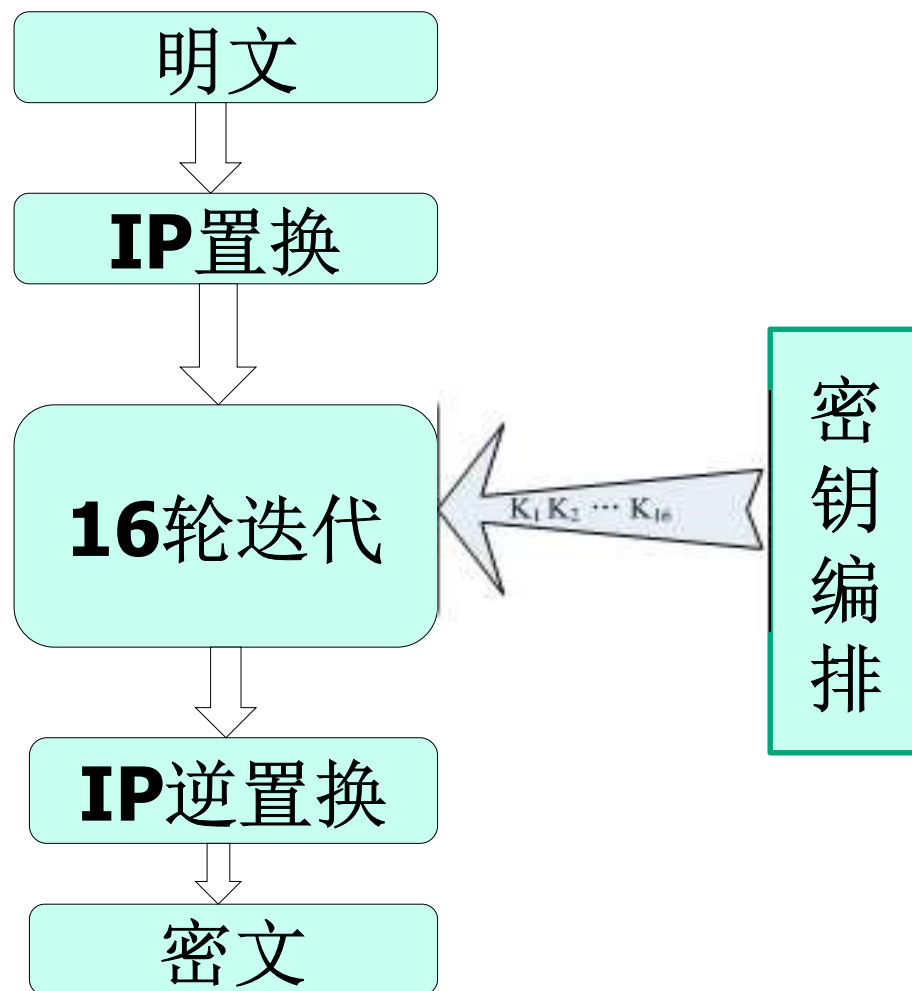
设计者: Joan Daemen和Vincent Rijmen
高级数据加密标准
(Advanced Encryption Standard, AES)
Rijndael → AES

版本	密钥长度 (Nk words)	分组长度 (Nb words)	迭代轮数 (Nr)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

AES算法的整体结构



AES算法的整体结构

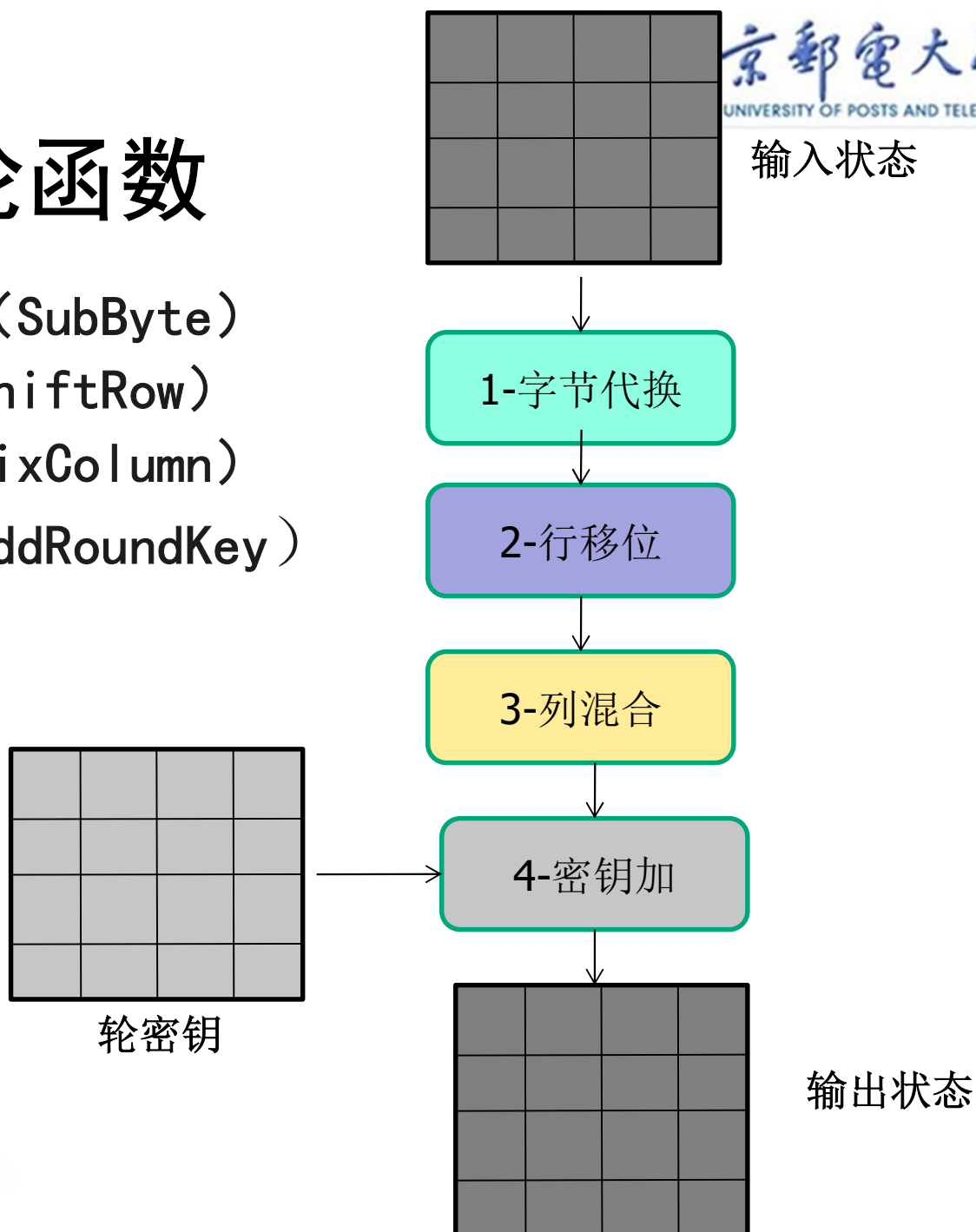


本节主要内容

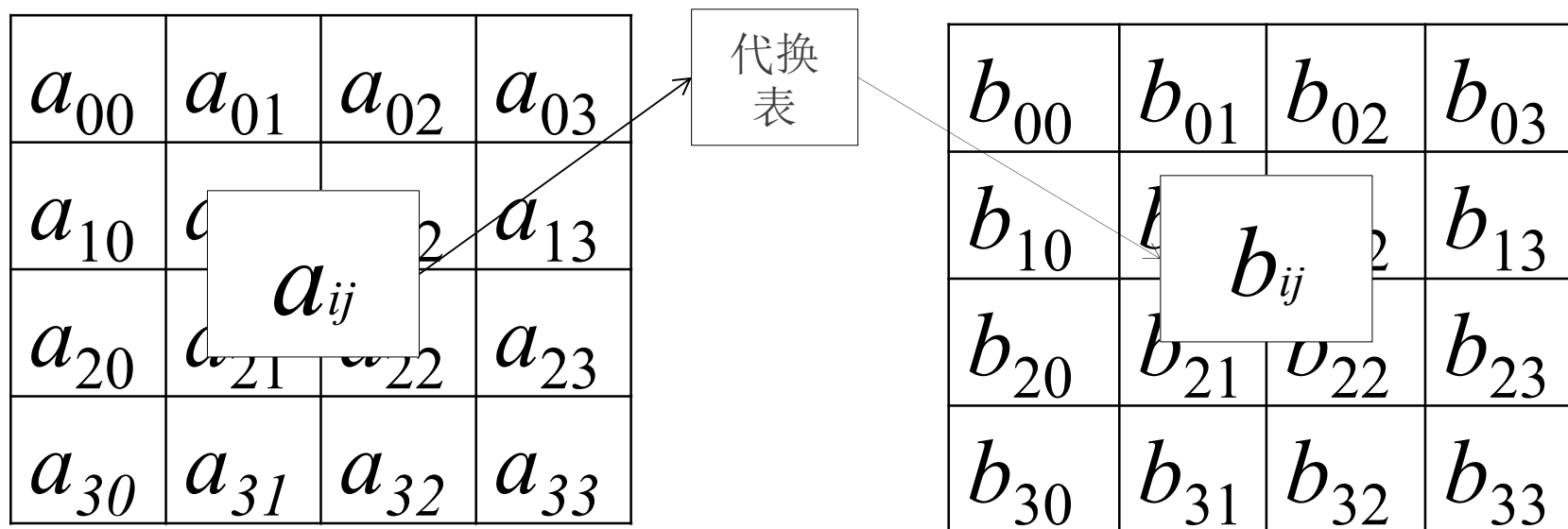
- AES算法的整体结构
- AES算法的轮函数
- AES算法的密钥编排算法
- AES的解密变换

AES算法的轮函数

- 1) 字节代换 (SubByte)
- 2) 行移位 (ShiftRow)
- 3) 列混合 (MixColumn)
- 4) 密钥加 (AddRoundKey)



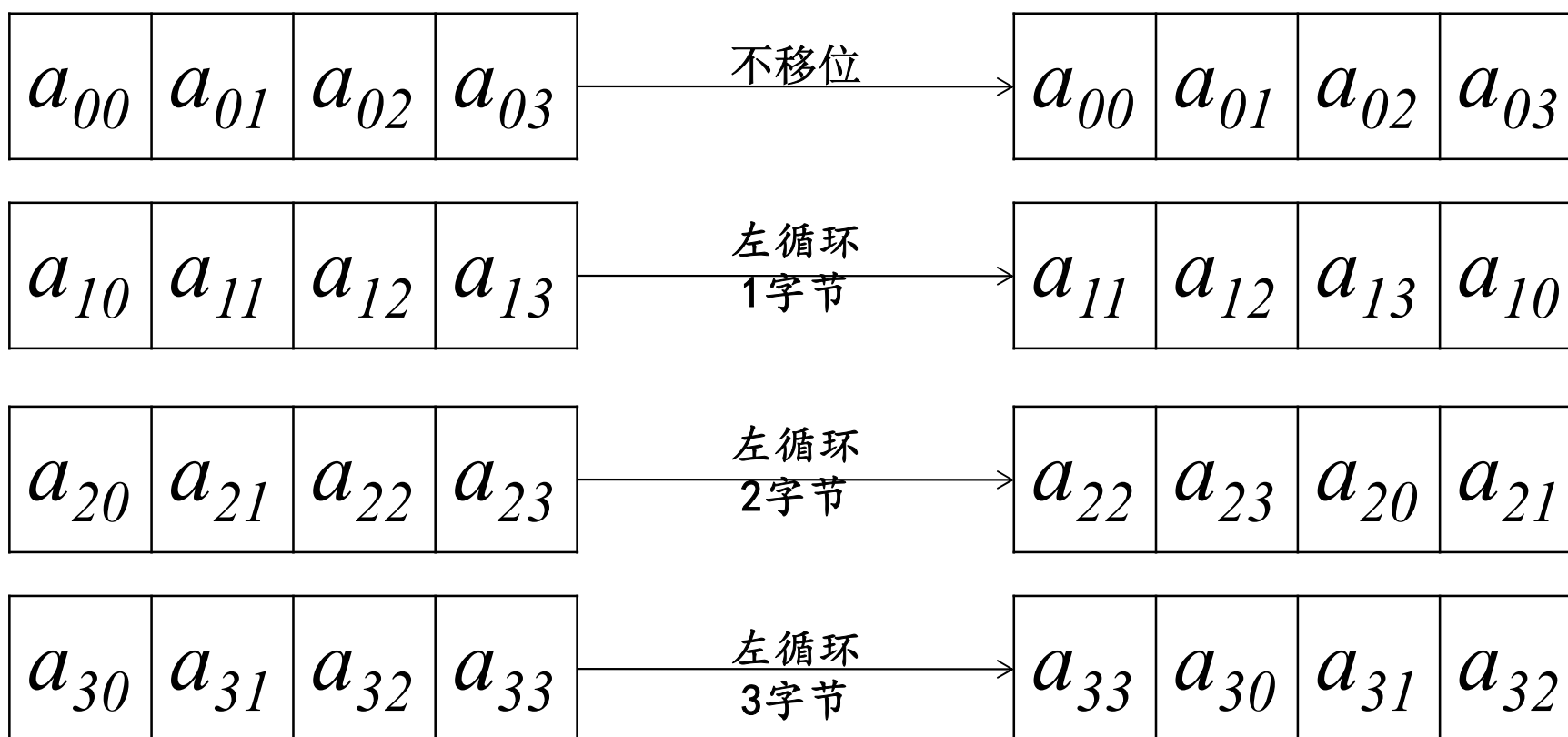
1-字节代换 (ByteSub)



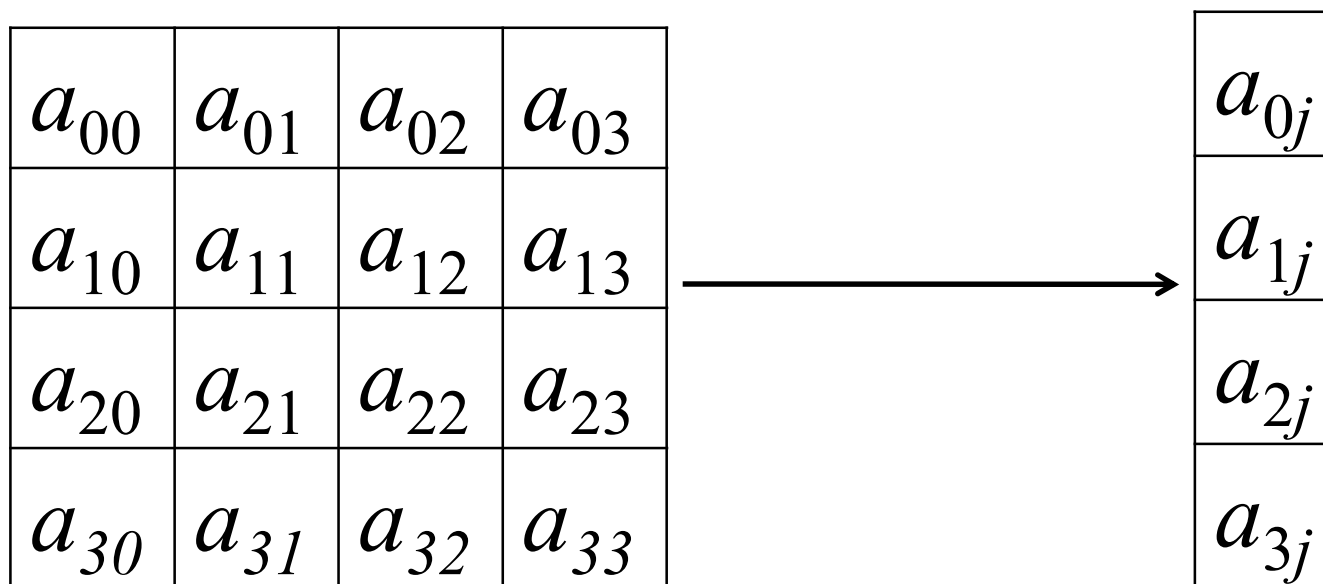
1-字节代换 (ByteSub)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

2-行移位 (ShiftRow)



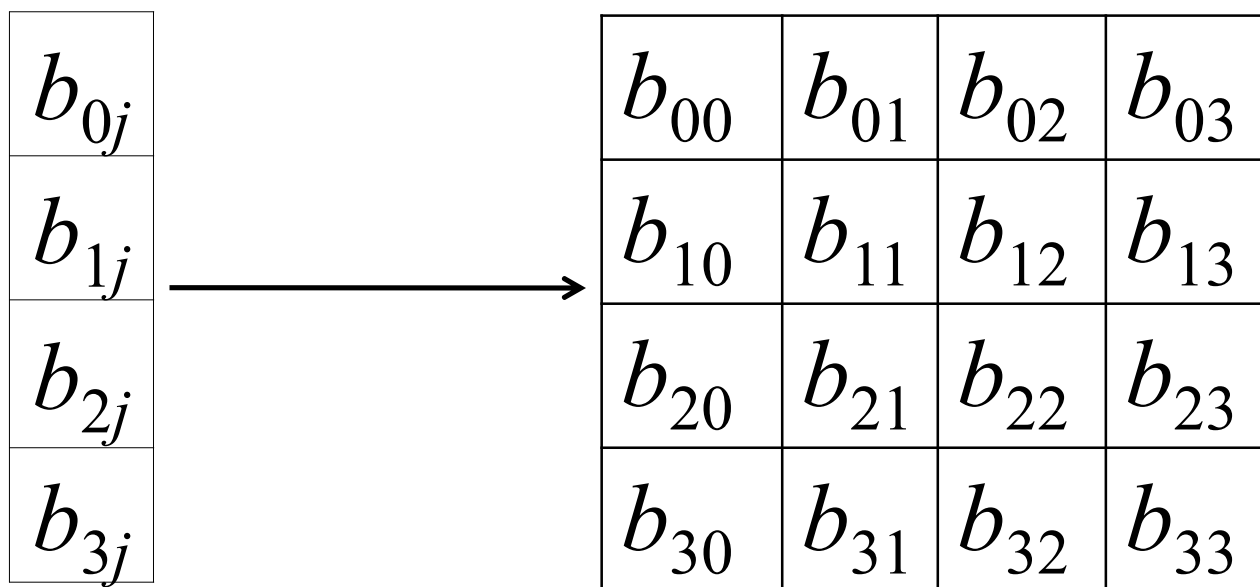
3-列混合 (MixColumn)



3-列混合 (MixColumn)

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

3-列混合 (MixColumn)



有限域上的字节运算

- AES选择的既约多项式为

$$m(x)=x^8+x^4+x^3+x+1$$

- GF(2⁸) 上的元素表示方法:

字节表示 Byte: $a_7a_6a_5a_4a_3a_2a_1a_0$

多项式表示 $s(x)$:

$$s(x)=a_7x^7+a_6x^6+a_5x^5+a_4x^4+a_3x^3+a_2x^2+a_1x+a_0$$

$m(x)$ 字节表示: 00000001 00011011
(0x011B)

有限域上的字节运算

例：0x57 + 0x83 = ?

$$(x^6+x^4+x^2+x+1) + (x^7+x+1)=x^7+x^6+x^4+x^2$$

$$0x57 \oplus 0x83 = 0xD4$$

例：0x57 × 0x83 = ?

$$\begin{aligned} & (x^6+x^4+x^2+x+1) \times (x^7+x+1) \\ &= (x^{13}+x^{11}+x^9+x^8+x^7) + (x^7+x^5+x^3+x^2+x) + (x^6+x^4+x^2+x+1) \\ &= x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1 \\ &= x^7+x^6+1 \bmod m(x) = x^8+x^4+x^3+x+1 \end{aligned}$$

有限域上的字节运算

例： $0x57 + 0x83 = ?$

$$(x^6+x^4+x^2+x+1) + (x^7+x+1) = x^7+x^6+x^4+x^2$$

$$0x57 \oplus 0x83 = 0xD4$$

例： $0x57 \times 0x83 = ?$

$$\begin{aligned} & (x^6+x^4+x^2+x+1) \times (x^7+x+1) \\ &= (x^{13}+x^{11}+x^9+x^8+x^7) + (x^7+x^5+x^3+x^2+x) + (x^6+x^4+x^2+x+1) \\ &= x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1 \\ &= x^7+x^6+1 \pmod{m(x)} = x^8+x^4+x^3+x+1 \end{aligned}$$

有限域上的字节运算

例：0x41 \times 0x02 = ?

$$(x^6+1) \times x \\ = x^7+x$$

例：0xC1 \times 0x02 = ?

$$(x^7+x^6+1) \times x \\ = x^8+x^7+x \bmod m(x) \\ = x^7+x^4+x^3+1$$

有限域上的字节运算

GF(2⁸) 上的快速乘法:

$$1) a_7a_6a_5a_4a_3a_2a_1a_0 \times 0x02 \quad / \quad a(x) \times x$$

- 当字节的最高位为0时 ($a(x)$ 的7次项系数为0)，左移补0;
- 当字节的最高位为1时 ($a(x)$ 的7次项系数为1)，左移补0，再按位模0x011B

有限域上的字节运算



GF(2⁸) 上的快速乘法:

$$2) \quad a_7a_6a_5a_4a_3a_2a_1a_0 \times b_7b_6b_5b_4b_3b_2b_1b_0 \\ / \quad \mathbf{a(x)} \times \mathbf{b(x)}$$

$$[\mathbf{a(x)} \times \mathbf{b(x)}] \bmod \mathbf{m(x)} = \\ \{ [\mathbf{a(x)} \bmod \mathbf{m(x)}] \times [\mathbf{b(x)} \bmod \mathbf{m(x)}] \} \bmod \mathbf{m(x)}$$

$$\mathbf{a(x)} \times [\mathbf{b(x)} + \mathbf{c(x)}] = \mathbf{a(x)} \times \mathbf{b(x)} + \mathbf{c(x)} \times \mathbf{a(x)}$$

$$a(x) \times b(x) =$$

$$[b_0 \times a(x)]$$

$$+ [b_1 x \times a(x)] \quad A^{(0)} = a(x) \bmod m(x)$$

$$+ [b_2 x^2 \times a(x)] \quad A^{(1)} = x \times a(x) = x \times A^{(0)} \bmod m(x)$$

$$+ [b_3 x^3 \times a(x)] \quad A^{(2)} = x^2 \times a(x) = x \times A^{(1)} \bmod m(x)$$

$$+ [b_4 x^4 \times a(x)] \quad A^{(3)} = x^3 \times a(x) = x \times A^{(2)} \bmod m(x)$$

$$+ [b_5 x^5 \times a(x)] \quad A^{(4)} = x^4 \times a(x) = x \times A^{(3)} \bmod m(x)$$

$$+ [b_6 x^6 \times a(x)] \quad A^{(5)} = x^5 \times a(x) = x \times A^{(4)} \bmod m(x)$$

$$+ [b_7 x^7 \times a(x)] \quad A^{(6)} = x^6 \times a(x) = x \times A^{(5)} \bmod m(x)$$

$$A^{(7)} = x^7 \times a(x) = x \times A^{(6)} \bmod m(x)$$

$$a(x) \times b(x) = b_0 A^{(0)} + b_1 A^{(1)} + b_2 A^{(2)} + b_3 A^{(3)} + b_4 A^{(4)} \\ + b_5 A^{(5)} + b_6 x^6 A^{(6)} + b_7 A^{(7)}$$

3-列混合 (MixColumn)

课堂练习：列混合运算(128比特分组)

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} CD \\ 00 \\ 00 \\ 00 \end{pmatrix}$$

b3: 4C b2: CD b1: CD b0: 81

4—密钥加 (AddRoundKey)

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}
a_{30}	a_{31}	a_{32}	a_{33}

 \oplus

k_{00}	k_{01}	k_{02}	k_{03}
k_{10}	k_{11}	k_{12}	k_{13}
k_{20}	k_{21}	k_{22}	k_{23}
k_{30}	k_{31}	k_{32}	k_{33}

 $=$

b_{00}	b_{01}	b_{02}	b_{03}
b_{10}	b_{11}	b_{12}	b_{13}
b_{20}	b_{21}	b_{22}	b_{23}
b_{30}	b_{31}	b_{32}	b_{33}

本节主要内容

- AES算法的整体结构
- AES算法的轮函数
- AES算法的密钥编排算法
- AES算法的解密变换

AES算法的密钥编排算法

密钥编排指从种子密钥得到轮密钥的过程，AES的密钥编排由密钥扩展和轮密钥选取两部分组成，其基本原则如下：

- 1) 轮密钥的总比特数等于轮数加1再乘以分组长度；如128比特的明文经过10轮的加密，则总共需要 $(10+1) * 128 = 1408$ 比特的密钥。
- 2) 种子密钥被扩展成为扩展密钥；
- 3) 轮密钥从扩展密钥中取，其中第1轮轮密钥取扩展密钥的前 N_b 个字，第2轮轮密钥取接下来的 N_b 个字，依次类推。

AES算法的密钥编排算法

1) 扩展密钥

扩展密钥是以4字节字为元素的一维阵列，表示为 $W[Nb * (N_r + 1)]$ ，其中前 N_k 个字取为种子密钥，以后每个字按递归方式定义。扩展算法根据 $N_k \leq 6$ 和 $N_k > 6$ 有所不同。

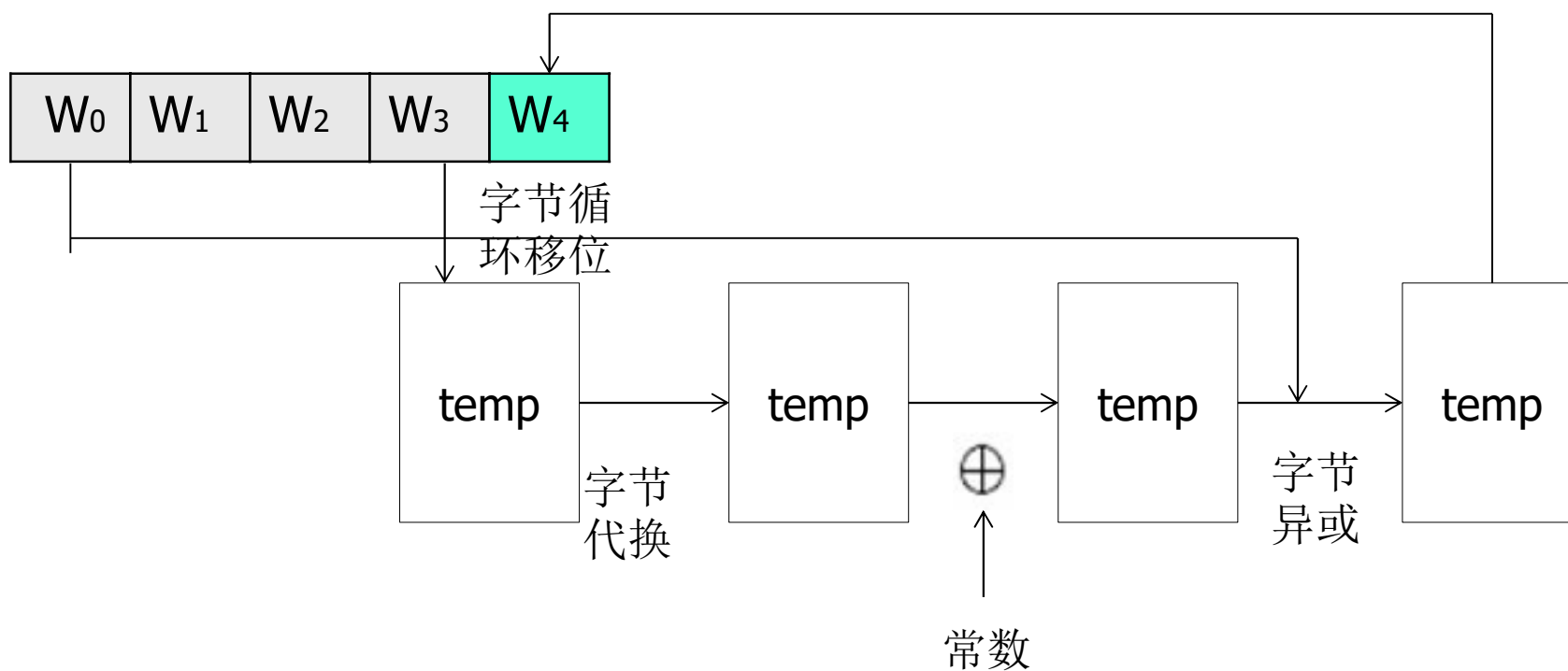
AES算法的密钥编排算法

当 $N_k \leq 6$ 时，扩展算法如下：

```
KeyExpansion (byteKey[4*Nk] , W[Nb*(Nr+1)])
{
    for (i =0; i < Nk; i ++ )
        W[i]=(Key[4* i],Key[4* i +1],Key[4* i +2],Key[4* i +3] );
    for (i =Nk; i <Nb*(Nr+1); i ++ )
    {
        temp=W[i-1];
        if (i % Nk== 0)
            temp=SubByte (RotByte (temp))^Rcon[i /Nk];
        W[i]=W[i-Nk]^ temp;
    }
}
```

AES算法的密钥编排算法

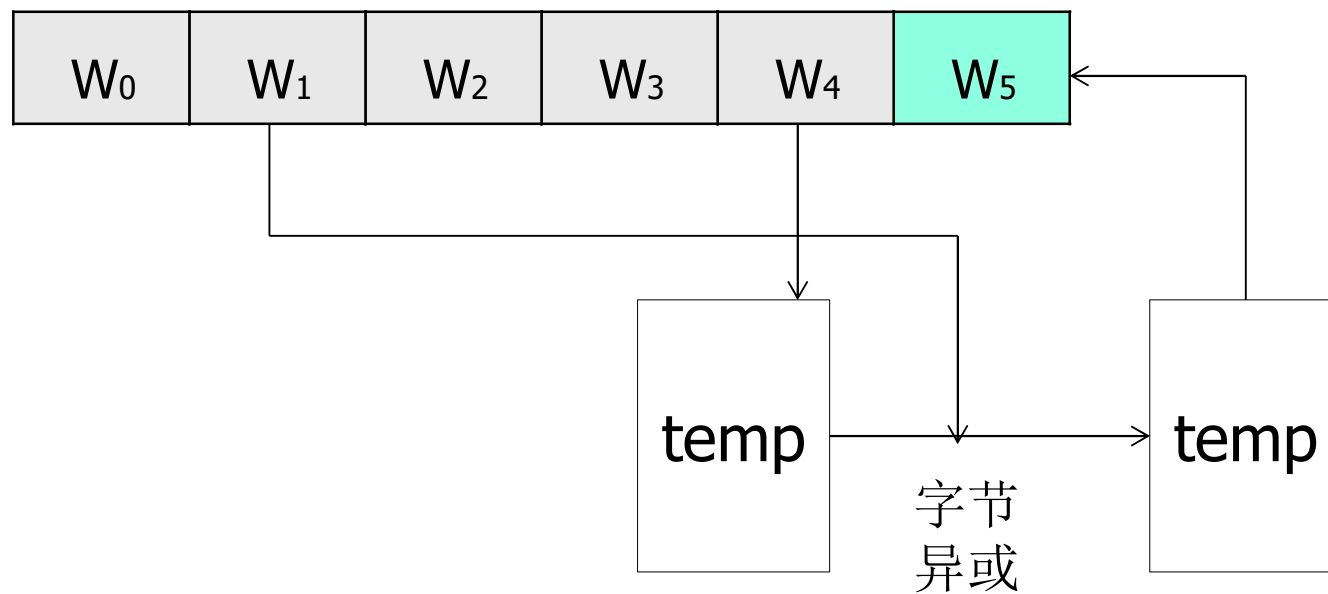
$i=4$ $i\%4==0$



$Nk=4$

AES算法的密钥编排算法

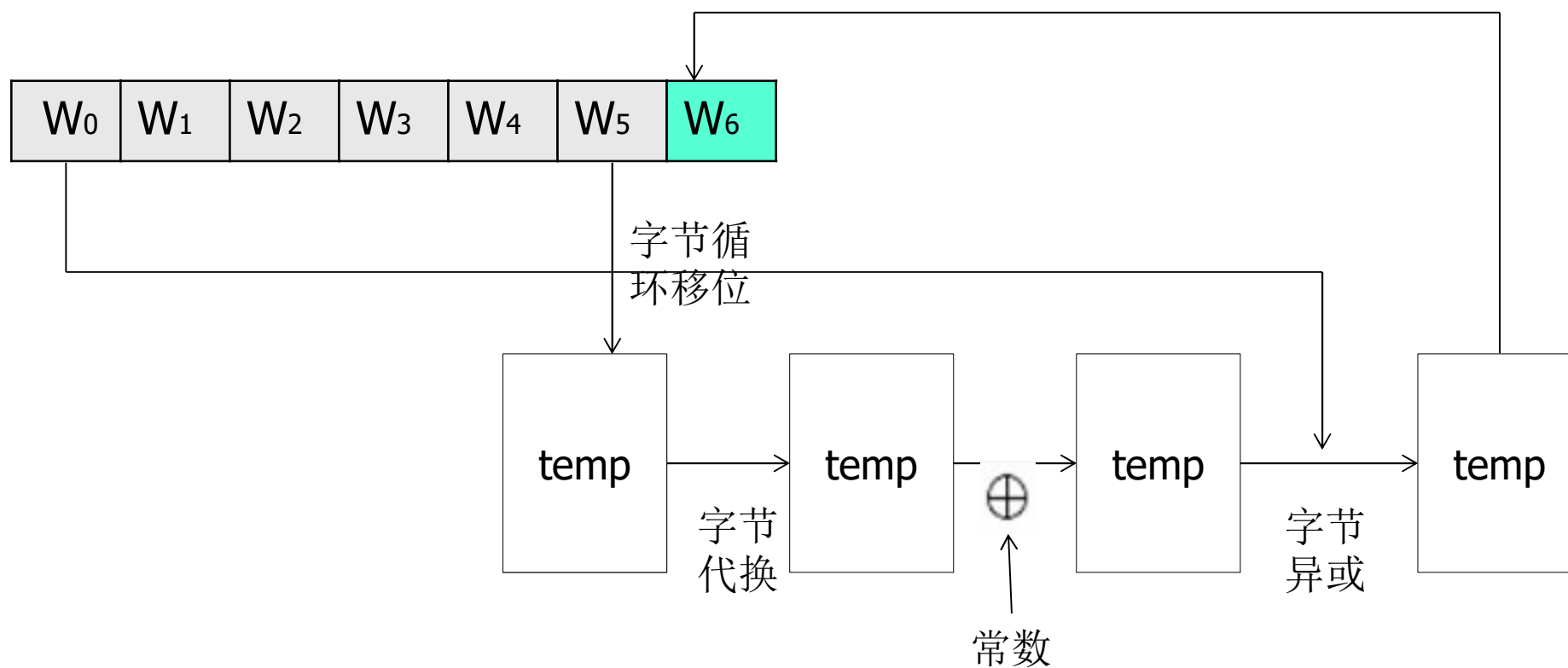
$i=5$ $5\%4 \neq 0$



$Nk=4$

AES算法的密钥编排算法

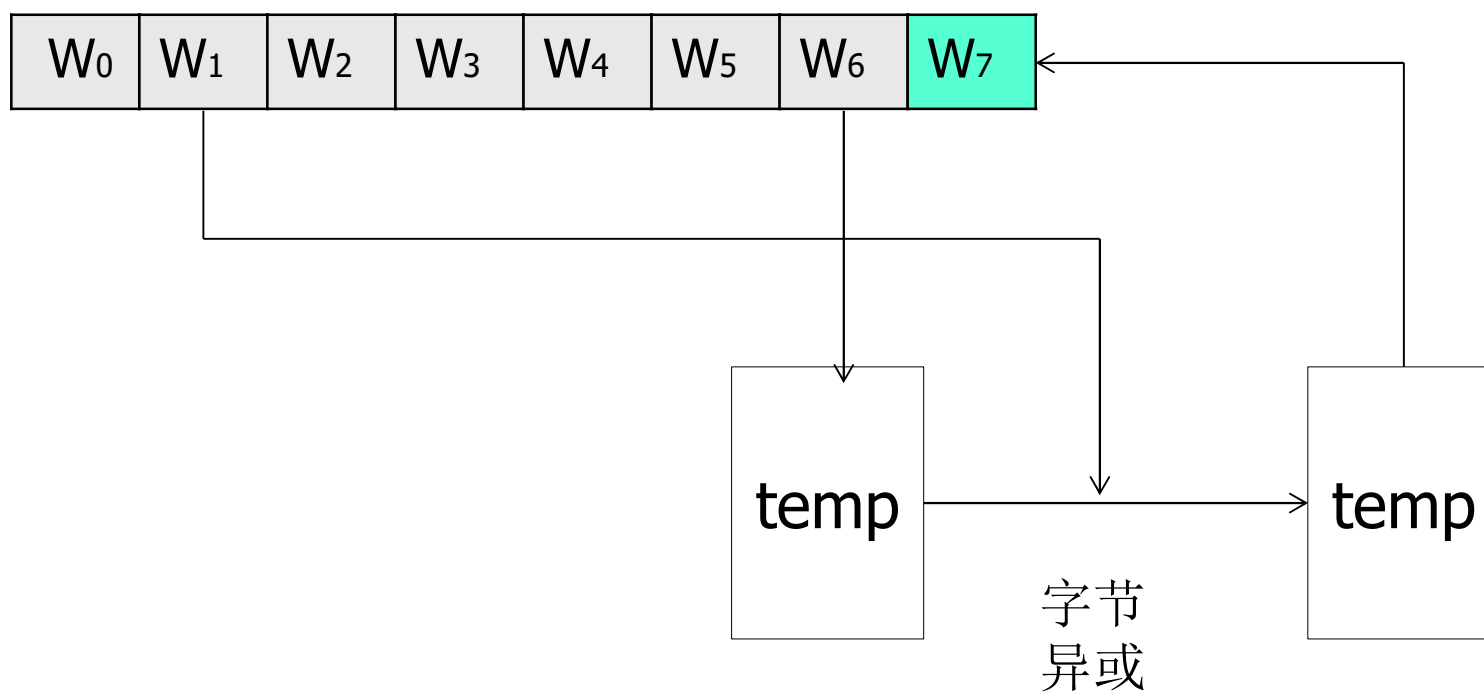
$i=6$ $i\%6==0$



$Nk=6$

AES算法的密钥编排算法

$i=7$ $i\%6 \neq 0$



$Nk=6$

AES算法的密钥编排算法

- $\text{Key}[4*N_k]$ 为种子密钥，看作以字为元素的一维阵列；
- 函数 $\text{SubByte}()$ 返回4字节字，其中每一个字节都是用 Rijndael 的 S 盒作用到输入字对应的字节得到；
- 函数 $\text{RotByte}()$ 也返回4字节字，该字由输入的字循环移位得到，即当输入字为 (a, b, c, d) 时，输出字为 (b, c, d, a) 。

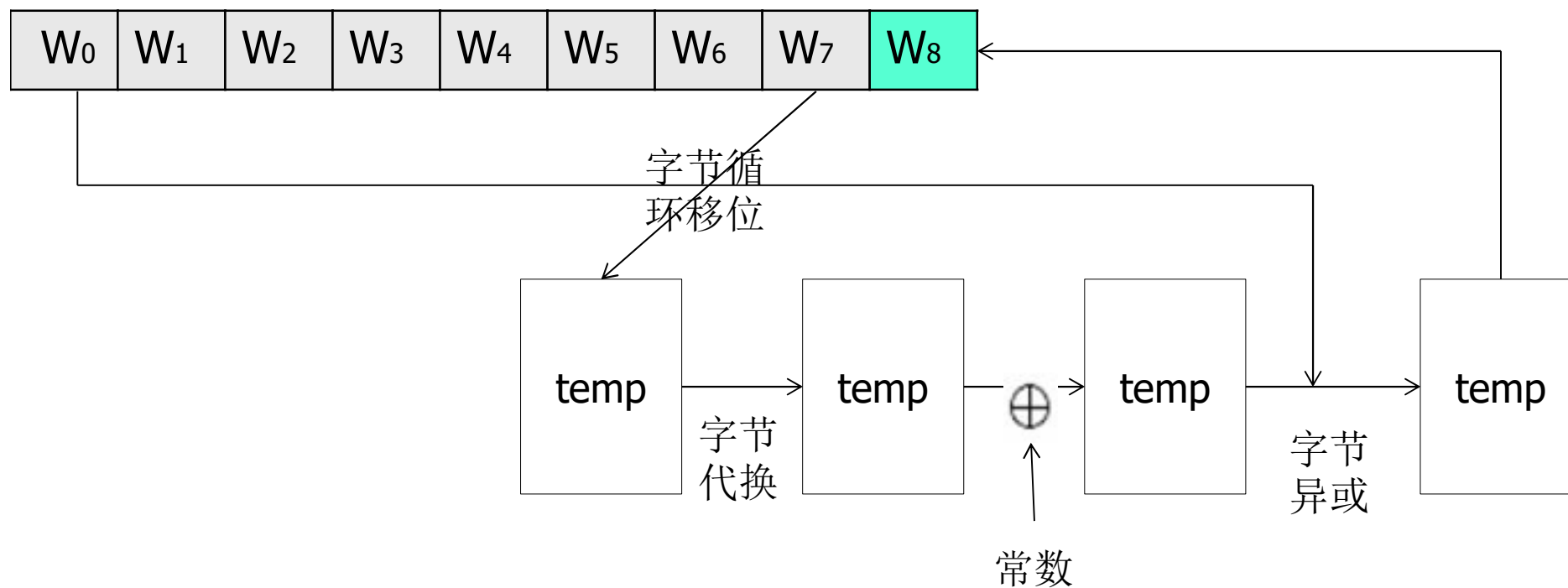
AES算法的密钥编排算法

当 $N_k > 6$ 时，扩展算法如下：

```
KeyExpansion (byte Key[4*Nk] , W[Nb*(Nr+1)])  
{  
    for (i=0; i < Nk; i ++)  
        W[i]=(Key[4* i], Key[4* i +1], Key[4* i +2], Key[4* i  
+3] );  
    for (i =Nk; i <Nb*(Nr+1); i ++)  
    {  
        temp=W[i -1];  
        if (i % Nk==0)  
            temp=SubByte (RotByte (temp))^Rcon[i /Nk];  
        else if (i % Nk==4)  
            temp=SubByte (temp);  
        W[i]=W[i - Nk]^ temp;  
    }  
}
```

AES算法的密钥编排算法

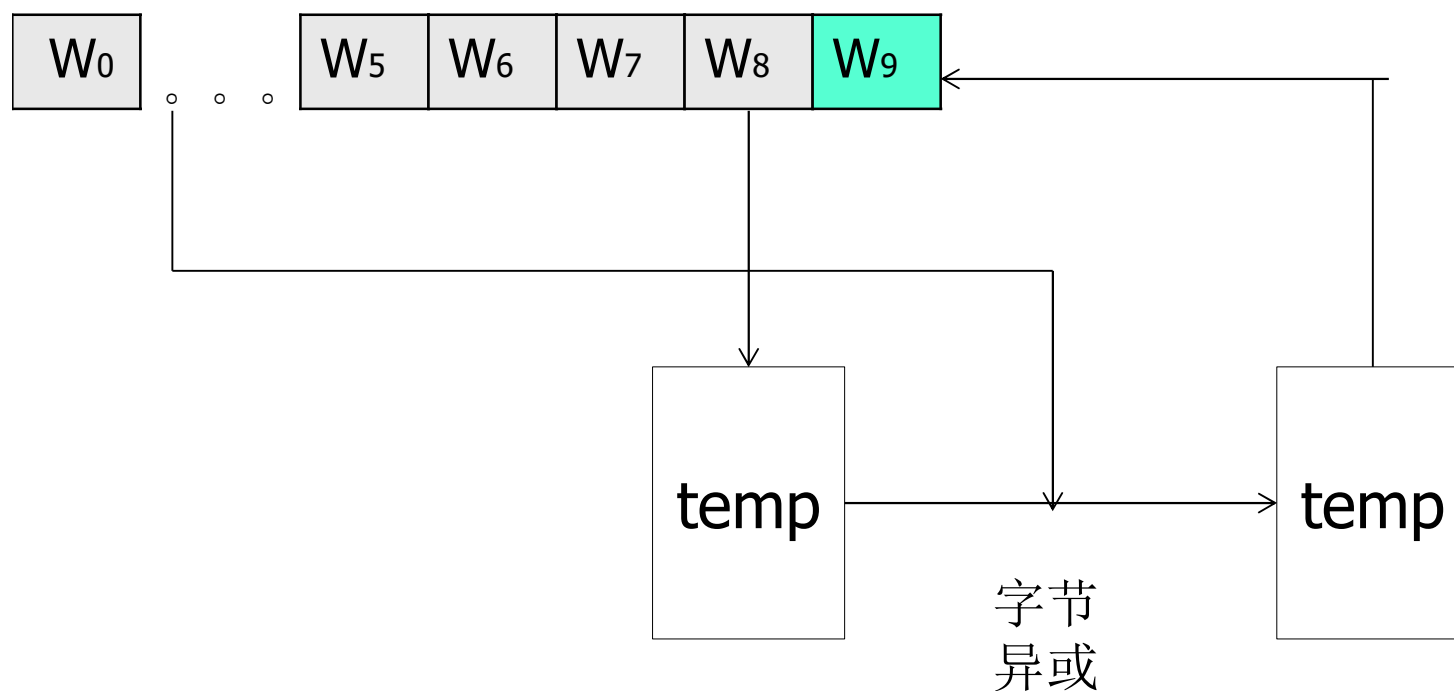
$i=8 \quad i\%8==0$



$Nk=8$

AES算法的密钥编排算法

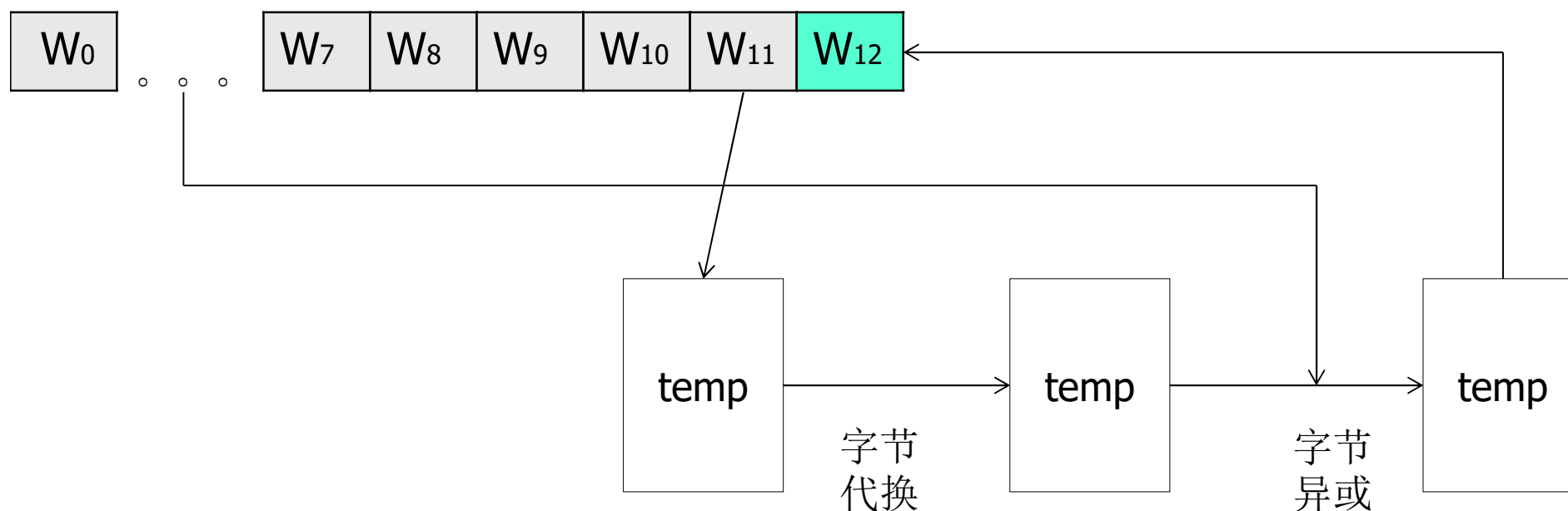
$i=9$ $i\%8 \neq 0$ or 4



$Nk=8$

AES算法的密钥编排算法

$i=12$ $i\%8==4$



$Nk=8$

AES算法的密钥编排算法

$Rcon[i/Nk]$ 为轮常数，其值与 Nk 无关，定义为（字节用十六进制表示，同时理解为 $GF(2^8)$ 上的元素）：

$$Rcon[i] = (RC[i], 00, 00, 00)$$

其中 $RC[i]$ 是 $GF(2^8)$ 中值为 x^{i-1} 的元素，因此

$$RC[1] = 1 \text{ (即 '01') }$$

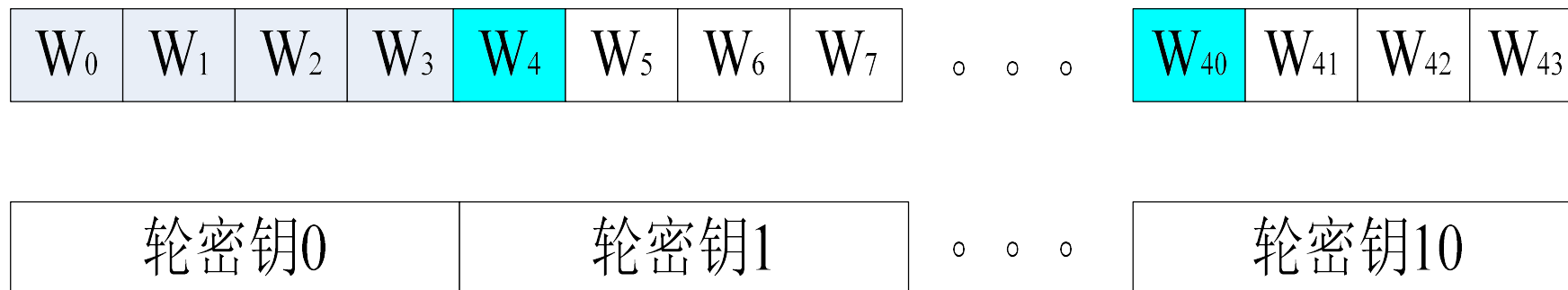
$$RC[2] = x \text{ (即 '02') }$$

$$RC[i] = x \cdot RC[i-1] = x^{i-1}$$

AES算法的密钥编排算法

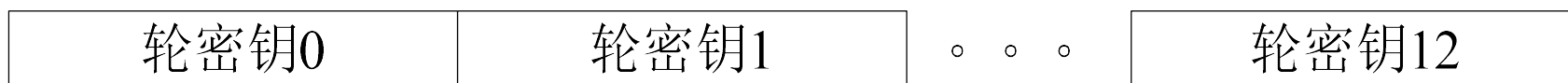
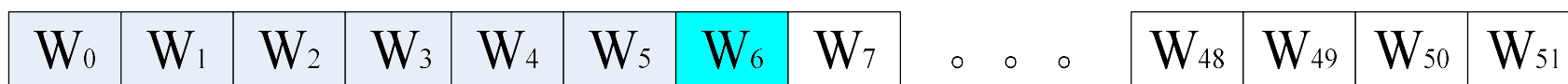
2) 轮密钥选取

轮密钥 i （即第 i 个轮密钥）由轮密钥缓冲字 $W[Nb * i]$ 到 $W[Nb * (i+1)]$ 给出：

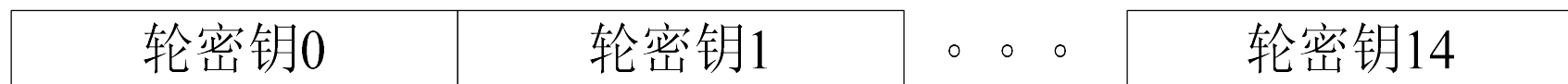
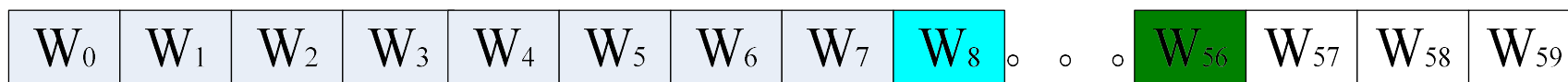


Nb=4及Nk=4时的密钥扩展与轮密钥选取

AES算法的密钥编排算法



Nb=4及Nk=6时的密钥扩展与轮密钥选取



Nb=4及Nk=8时的密钥扩展与轮密钥选取

本节主要内容

- AES算法的整体结构
- AES算法的轮函数
- AES算法的密钥编排算法
- AES算法的解密变换

AES 的解密变换

AES解密运算是加密运算的逆运算，其中轮函数的逆为：

1) ByteSub的逆变换由代换表的逆表做字节代换，也可通过如下两步实现：首先进行仿射变换的逆变换，再求每一字节在 $GF(2^8)$ 上逆元。

2) 行移位运算的逆变换是循环右移，位移量与左移时相同。

AES的解密变换

3) 列混合运算的逆运算是类似的，即每列都用一个特定的多项式 $d(x)$ 相乘， $d(x)$ 满足

$$(03x^3+01x^2+01x+02)*d(x)=01$$

由此可得

$$d(x)=0Bx^3+0Dx^2+09x+0E$$

4) 密钥加运算的逆运算是其自身。

主要知识点小结

- AES算法的整体结构
- AES算法的轮函数

扩散与混淆

00000000	1110	00000011	000000	10000000	01000000	10000000	11111111			
00000110	0100	00000111	111111	10000000	00000101	10000001	11100000			
00011000	1101	00011001	001111	10000000	11110100	10000100	11000000			
00011110	0001	00011111	001000	10001000	10000000	10000111	10000010			
00110000	0010	00110001	111100	10000000	11101010	10100000	10100000			
00110110	1111	00110111	000100	10000000	01110100	10101010	11000001			
00111000	1011	00111001	110011	10010000	00000100	10100100	10000001			
00111110	1000	00111111	000001	10011000	10101111	10101111	10101111			
01000000	0011	01000001	100100	11000000	11111111	11000000	10101010			
01000110	1010	01000111	001100	11000000	11100000	11000001	11010111			
01001000	0110	01001001	110000	11000000	10000101	11010100	10000011			
01001110	1100	01001111	100111	11001000	00111111	11010111	11101010			
01100000	0101	01100001	100001	11100000	00000111	11100000	11010100			
01100110	1001	01100111	000001	11100000	10000100	11101010	10000000			
01101000	0000	01101001	000111	11110000	00000101	11110100	10100100			
01101110	0111	01101111	100000	11111000	00000000	11111111	11110100			

THE END !

