Vietnam National University Ho Chi Minh City, University of Science
Department of Information Technology

Topic 4:

Data Encryption Standard và Advanced Encryption Standard

Assoc. Prof. Trần Minh Triết
PhD. Trương Toàn Thịnh

TRUONG DAI HOC KHOA HOC TU NHIEN
KHTN
TP. HO CHI MINH

cdio
4.0

fit@hcmus

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Contents

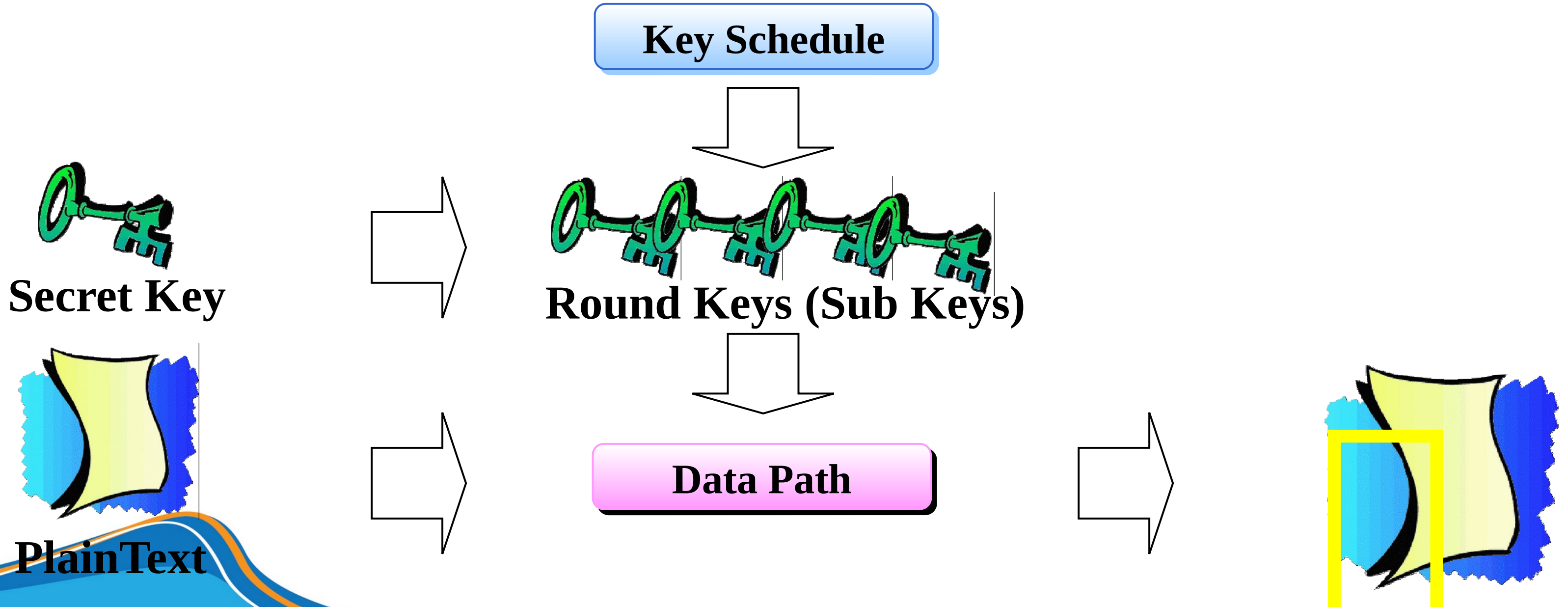
- **D**ata **E**ncryption **S**tandard (DES)
- **A**dvanced **E**ncryption **S**tandard (AES)
  - Guidance of AES128

Product Cipher

- Encryption using only substitution or transposition is not safe (due to the nature of the language)
- Consecutive use of simple encryption operations creates a more secure way of encrypting information
  - **Substitution** combined with **Substitution** is more secure than one **Substitution**
  - **Transposition** combined with **Transposition** is more secure than one **Transposition**
  - **Substitution** combined with **Transposition** is much safer than using only one type of operation (replace or swap)
- This is the prelude to modern encryption methods

Block-cipher

- Data Path: Usually, the encryption process consists of several consecutive encryption cycles (rounds); Each cycle consists of many encryption operations
- Key Schedule: From a secret-key, produce (with rule) key values to be used in each encryption cycle (round key)



Cipher architecture

- Common architectures:
  - Fiestel: for example, Blowfish, Camellia, CAST-128, DES, FEAL, KASUMI, LOKI97, Lucifer, MARS, MAGENTA, MISTY1, RC5, TEA, Triple DES, Twofish, and XTEA
  - SPN: for example, Rijndael – AES, Anubis…
- Data Encryption Standard: block-cipher
  - Ideas: product cipher with key-size 56 bits and block-size: 64 bits
  - IBM developed from **Lucifer** and published in **1975**
  - Chosen to be Federal Information Processing Standard-FIPS) in **1976**
  - Encryption and decryption algorithm published
  - Mathematical and cryptographic basis of DES design: secret information

Feistel architecture - encryption

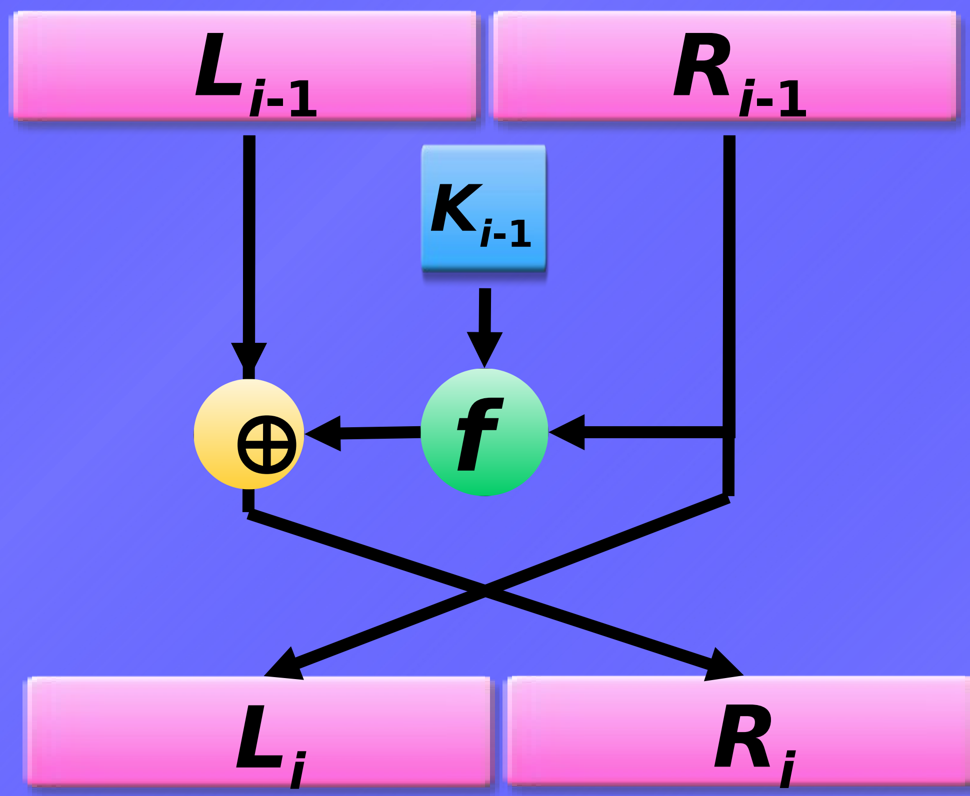
Round 1

...

Round *i*

...

Round *Nr*



$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus f\,(R_{i-1}, K_{i-1})$$

Feistel architecture - decryption

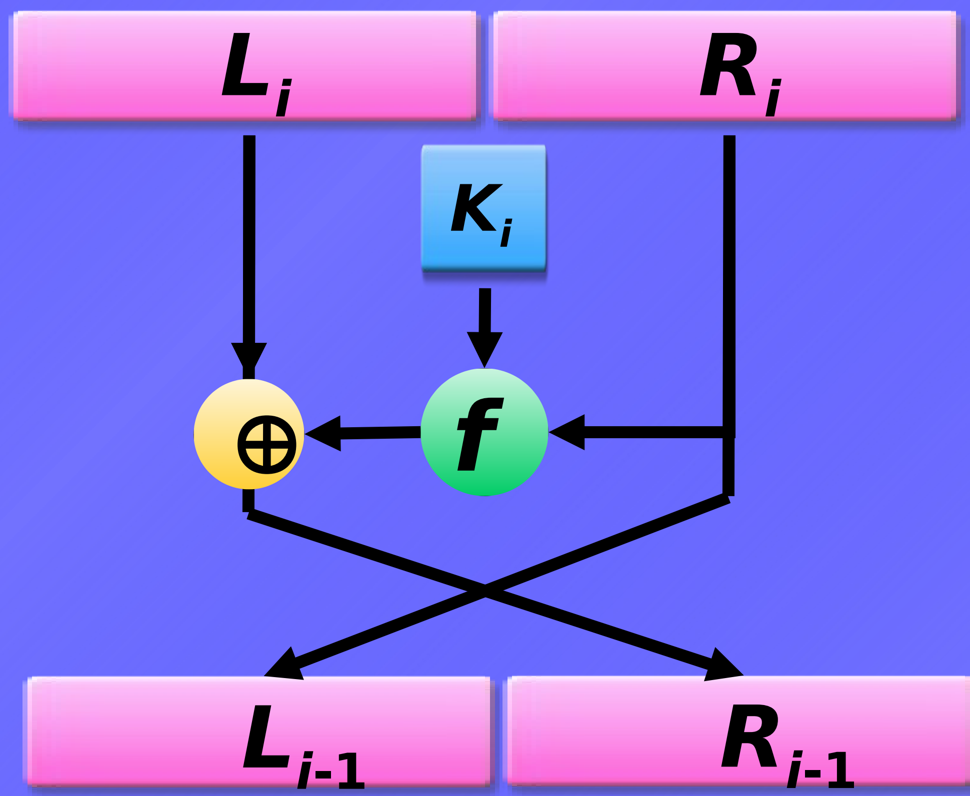
Round *Nr*

...

Round *i*

...

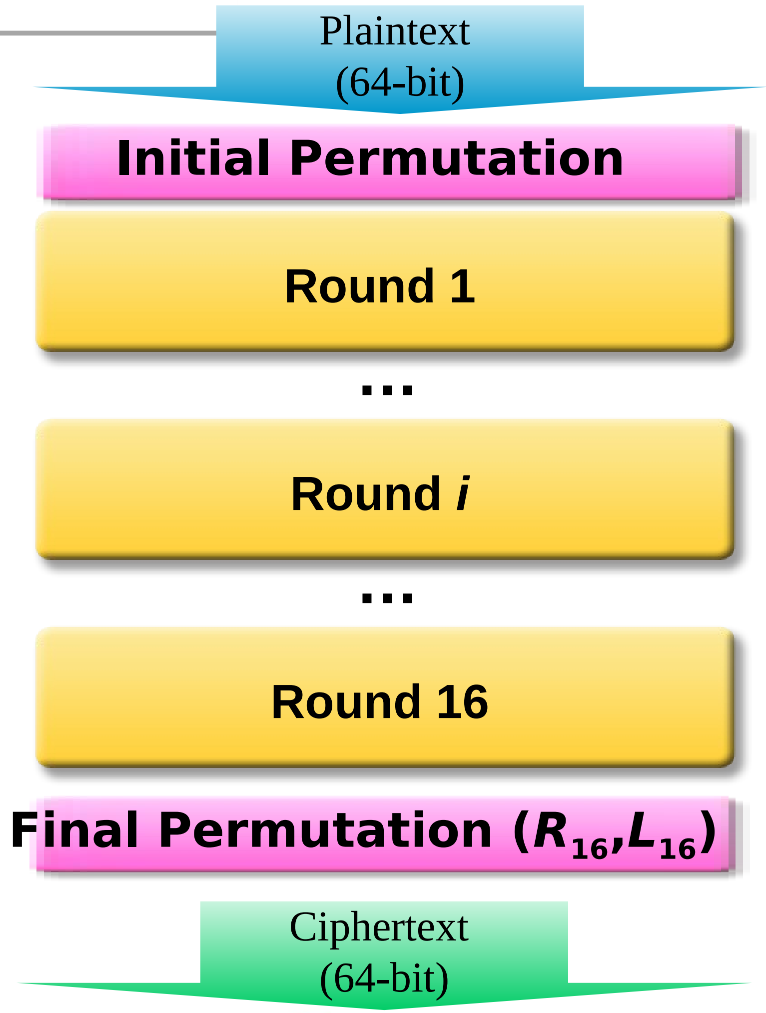
Round *1*



$$R_{i\text{-}1} = L_i$$

$$L_{i\text{-}1} = R_i \oplus f\,(L_i, K_i)$$

The Encryption Process of the DES



IP: Initial Permutation

FP: Final Permutation

FP = IP^{-1}

<u>Notes</u>:

FP and IP have no cryptographic significance, only for loading data into and out of data blocks (in the mid-1970s hardware mechanism!!!)

Initial Permutation

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Example: 58^{th} bit of x (input) becomes the 1^{st} bit of IP(x) (output)
50^{th} bit of x (input) becomes the 2^{nd} bit of IP(x) (output)

fit@hcmus

Final Permutation

FP = IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Example: The 1st bit of x (input) becomes 58th bit of IP$^{-1}(x)$ (output)
The 2nd bit of x (input) becomes 50th bit of IP$^{-1}(x)$ (output)

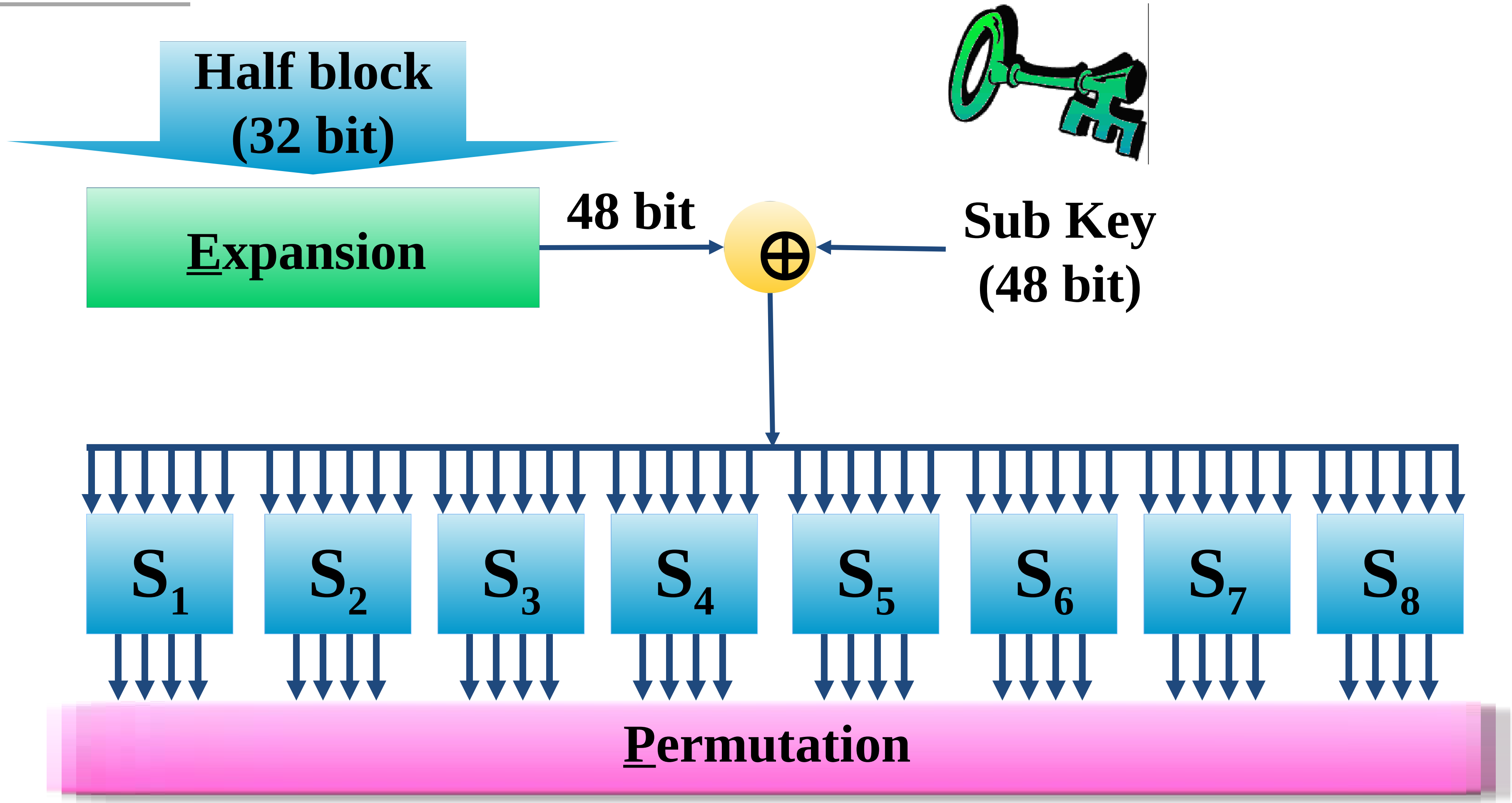
Expansion

Table E: extend from 32 bits to 48 bits

Table of choosing bit					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Example: 32nd bit of x (input) becomes the 1st bit of E(x) (output)

Function *f* of DES



S-box

S_1															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

Example: $B_j = b_1b_2b_3b_4b_5b_6$ then $S_j(B_i) = S_j[b_1b_6][b_2b_3b_4b_5]$

S-box

S_3															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S-box

S_5															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S-box

S_7															
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Permutation table

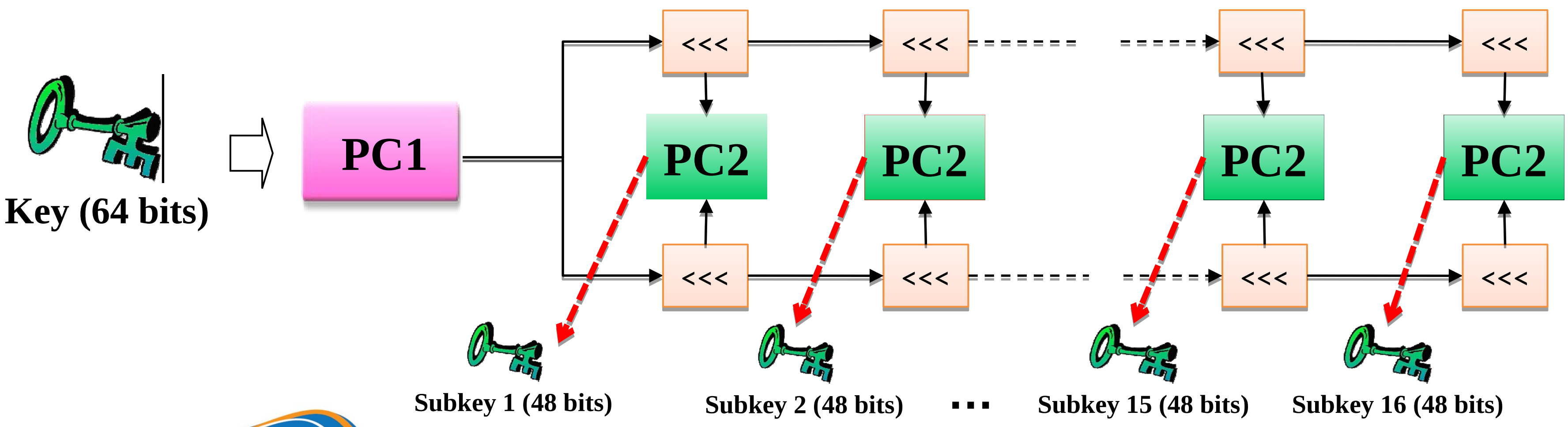
P			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Notes:

- Input of 32 bits (includes 8 S-box, each S-box has 4 bits)
- Output 32 bits are permuted
- Example: the 1^{st} bit of **P**(x) (output) is 16^{th} bit of x (input)

Key Schedule

- Bit rotation operation
  - <<<: Rotate to the left
  - >>>: Rotate to the right
- With subkey in {1, 2, 9, 16}: rotate to the left one position
- With remaining subkey: rotate to the left two positions



Permutations in Key Schedule

PC-1

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Choose 56 bits

(leave 8, 16, 24, 32, 40, 48, 56, 64)

PC-2

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Choose 48 bits

(leave 9, 18, 22, 25, 35, 38, 43, 54)

Example: 50^{th} bit of x (input) becomes the 1^{st} bit of PC-1(x) (output)

Explain: $\mathbf{57} = 8 \times \mathbf{7} + 1 \rightarrow \mathbf{50} = \mathbf{57} - \mathbf{7}$

Comments

- 4 weak keys:
  - All bits 0
  - All bits 1
  - ½ are bit 0 (consecutive), ½ are bit 1 (consecutive)
- 12 semi-weak keys
  - Property: $\text{Encrypt}_k(P) = P$
  - Form of such key: 7 bit 0 (consecutive), 7 bit 1 (consecutive)
- Complement key
  - $\text{Encrypt}_k(P) = C \rightarrow \text{Encrypt}_{k^*}(P^*) = C^*$
  - x^* is formed by reversing bits of x

Pseudocode DES

- Encryption(K, M) // $|K| = 56$ and $|M| = 64$
  - $(K_1, \ldots, K_{16}) \leftarrow \mathit{CreateSubKey}(K)$ // $|K_i| = 48$
  - $M \leftarrow \mathbf{IP}(M)$
  - $M \rightarrow L_0 \,\|\, R_0$
  - Loop $r = 1, \ldots, 16$
    - $L_r \leftarrow R_{r-1}$; $R_r \leftarrow \boldsymbol{f}(K_r, R_{r-1}) \oplus L_{r-1}$
  - $C \leftarrow \mathbf{FP}(L_{16} \,\|\, R_{16})$
  - Return C

- CreateSubKey(K) // $|K| = 56$ bits
  - $K \leftarrow \textbf{PC-1}(K)$
  - $K \rightarrow C_0 \,\|\, D_0$
  - Loop $r = 1, \ldots, 16$
    - If $r \in \{1, 2, 9, 16\}$ then $j \leftarrow 1$, else $j \leftarrow 2$
    - $C_r \leftarrow C_{r-1} \lll j$; $D_r \leftarrow D_{r-1} \lll j$
    - $K_r \leftarrow \textbf{PC-2}(C_r \,\|\, D_r)$
  - Return $(K_1, \ldots, K_{16})$

Rijndael method

- Rijndael method proposed by Vincent Rijmen and Joan Daeman
- The National Institute of Standards and Technology (NIST) has adopted it as the Advanced Encryption Standard since October 2, 2000.
- The block cipher method has a flexible block size and key cipher with values of 128, 192 or 256 bits.
- This method is suitable for applications on a variety of systems from smart cards to personal computers.

Some Math Concepts

- Unit of information processed in Rijndael algorithm is the byte
- Each byte is treated as an element of the field Galois $GF(2^8)$ equipped with addition (denote $\oplus$) and multiplication (denote $\bullet$)
- Each byte is represented in different ways:
  - Binary: $\{b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0\}$ and hexadecimal: $\{h_1 h_0\}$
  - Polynomial with binary coefficients: $\sum_{i=0}^{7} b_i x^i$
- **Addition in $GF(2^8) = \{0, 1, \ldots, 255\}$**
  - $\{a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0\} \oplus \{b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0\} = \{c_7 c_6 c_5 c_4 c_3 c_2 c_1 c_0\}$
  - for $c_i = a_i \oplus b_i$, $0 \le i \le 7$
- Multiplication in $GF(2^8)$:
  - $a(x) \bullet b(x) \equiv a(x) \times b(x) \mod (x^8 + x^4 + x^3 + x + 1)$

Some Math Concepts

- Example of multiplication of {53} = 01010011 & {CA} = 11001010
  - $(x^6 + x^4 + x + 1) \bullet (x^7 + x^6 + x^3 + x)$
  - $= x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + x^2 + x \bmod x^8 + x^4 + x^3 + x + 1$
  - $= 1$
  - = {01}
- Example of addition of {53} = 01010011 & {CA} = 11001010
  - $(x^6 + x^4 + x + 1) \oplus (x^7 + x^6 + x^3 + x)$
  - $= x^7 + x^4 + x^3 + 1$
  - = {99}

Polynomials with coefficients in $GF(2^8)$

- Addition of two polynomials:
  - $a(x) = \sum_{i=0}^{3} a_i x^i = a_0 + a_1 x + a_2 x^2 + a_3 x^3$
  - $b(x) = \sum_{i=0}^{3} b_i x^i = b_0 + b_1 x + b_2 x^2 + b_3 x^3 \; (a_i, b_i \in GF(2^8))$
  - $a(x) + b(x) = \sum_{i=0}^{3} (a_i \oplus b_i) x^i$

$$= (a_0 \oplus b_0) + (a_1 \oplus b_1) x + (a_2 \oplus b_2) x^2 + (a_3 \oplus b_3) x^3$$

- Multiplication of 2 polynomials: $c(x) = a(x) \otimes b(x) = a(x) \bullet b(x) \bmod x^4 + 1$

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Polynomials with coefficients in $GF(2^8)$

- Two polynomials: $a(x) = \{3\}x^3 + \{1\}x^2 + \{1\}x + \{2\}$ and $b(x) = \{1\}x^2 + \{8\}x + \{1\}$
- Example of addition:
  - $a(x) + b(x) = \{3\}x^3 + \{9\}x + \{3\}$
- Example of multiplication:
  - $a(x) \otimes b(x) = a(x) \bullet b(x) \bmod x^4 + 1$
  - Consider multiplication step:
    - $a(x) \bullet b(x) = (\{3\}x^3 + \{1\}x^2 + \{1\}x + \{2\}) \bullet (\{1\}x^2 + \{8\}x + \{1\})$
    - $= (\{3\}\bullet\{1\})x^5 + (\{3\}\bullet\{8\})x^4 + (\{3\}\bullet\{1\})x^3 + (\{1\}\bullet\{1\})x^4 + (\{1\}\bullet\{8\})x^3 + (\{1\}\bullet\{1\})x^2 + (\{1\}\bullet\{1\})x^3 + (\{1\}\bullet\{8\})x^2 + (\{1\}\bullet\{1\})x + (\{2\}\bullet\{1\})x^2 + (\{2\}\bullet\{8\})x + (\{2\}\bullet\{1\})$
    - $= (\{3\}\bullet\{1\})x^5 + (\{3\}\bullet\{8\} \oplus \{1\}\bullet\{1\})x^4 + (\{3\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{1\}\bullet\{1\})x^3 + (\{1\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{2\}\bullet\{1\})x^2 + (\{1\}\bullet\{1\} \oplus \{2\}\bullet\{8\})x + (\{2\}\bullet\{1\})$

Polynomials with coefficients in $GF(2^8)$

- Example of multiplication: $a(x) \otimes b(x) = a(x) \bullet b(x) \bmod x^4 + 1$
  - Consider modulo step:
    - $c(x) = (\{3\}\bullet\{1\})x^5 + (\{3\}\bullet\{8\} \oplus \{1\}\bullet\{1\})x^4 + (\{3\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{1\}\bullet\{1\})x^3 + (\{1\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{2\}\bullet\{1\})x^2 + (\{1\}\bullet\{1\} \oplus \{2\}\bullet\{8\})x + (\{2\}\bullet\{1\})$
    - $c(x) \bmod x^4 + 1 = (\{3\}\bullet\{1\})x^{5 \bmod 4} + (\{3\}\bullet\{8\} \oplus \{1\}\bullet\{1\})x^{4 \bmod 4} + (\{3\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{1\}\bullet\{1\})x^{3 \bmod 4} + (\{1\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{2\}\bullet\{1\})x^{2 \bmod 4} + (\{1\}\bullet\{1\} \oplus \{2\}\bullet\{8\})x + (\{2\}\bullet\{1\})$
    - $c(x) \bmod x^4 + 1 = (\{3\}\bullet\{1\})x + (\{3\}\bullet\{8\} \oplus \{1\}\bullet\{1\}) + (\{3\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{1\}\bullet\{1\})x^3 + (\{1\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{2\}\bullet\{1\})x^2 + (\{1\}\bullet\{1\} \oplus \{2\}\bullet\{8\})x + (\{2\}\bullet\{1\})$
    - $c(x) \bmod x^4 + 1 = (\{3\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{1\}\bullet\{1\})x^3 + (\{1\}\bullet\{1\} \oplus \{1\}\bullet\{8\} \oplus \{2\}\bullet\{1\})x^2 + (\{3\}\bullet\{1\} \oplus \{1\}\bullet\{1\} \oplus \{2\}\bullet\{8\})x + (\{3\}\bullet\{8\} \oplus \{1\}\bullet\{1\} \oplus \{2\}\bullet\{1\})$

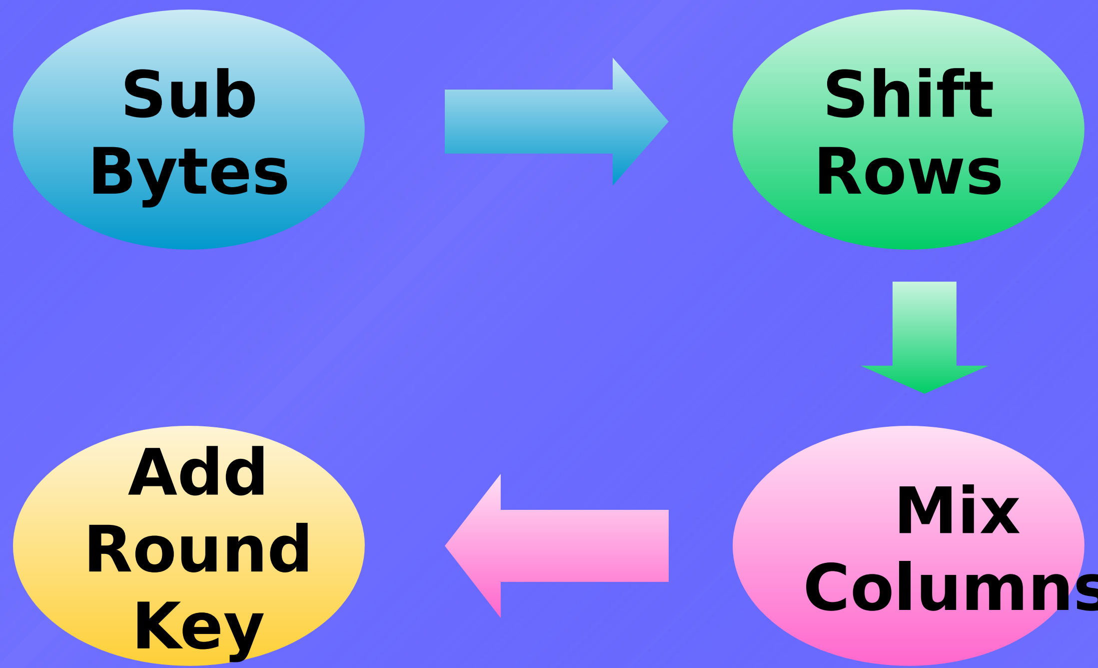
Data block representation and key code

- Intermediate result between the transformation steps is called state.
- A state is represented as a matrix of 4-rows and *Nb*-columns with *Nb* = block-size/32
- Cipher-key is represented as a matrix of 4-rows and *Nk*-columns with *Nk* = key-size/32
- Number of cycles ***Nr*** = *max*{*Nb*, *Nk*} + 6

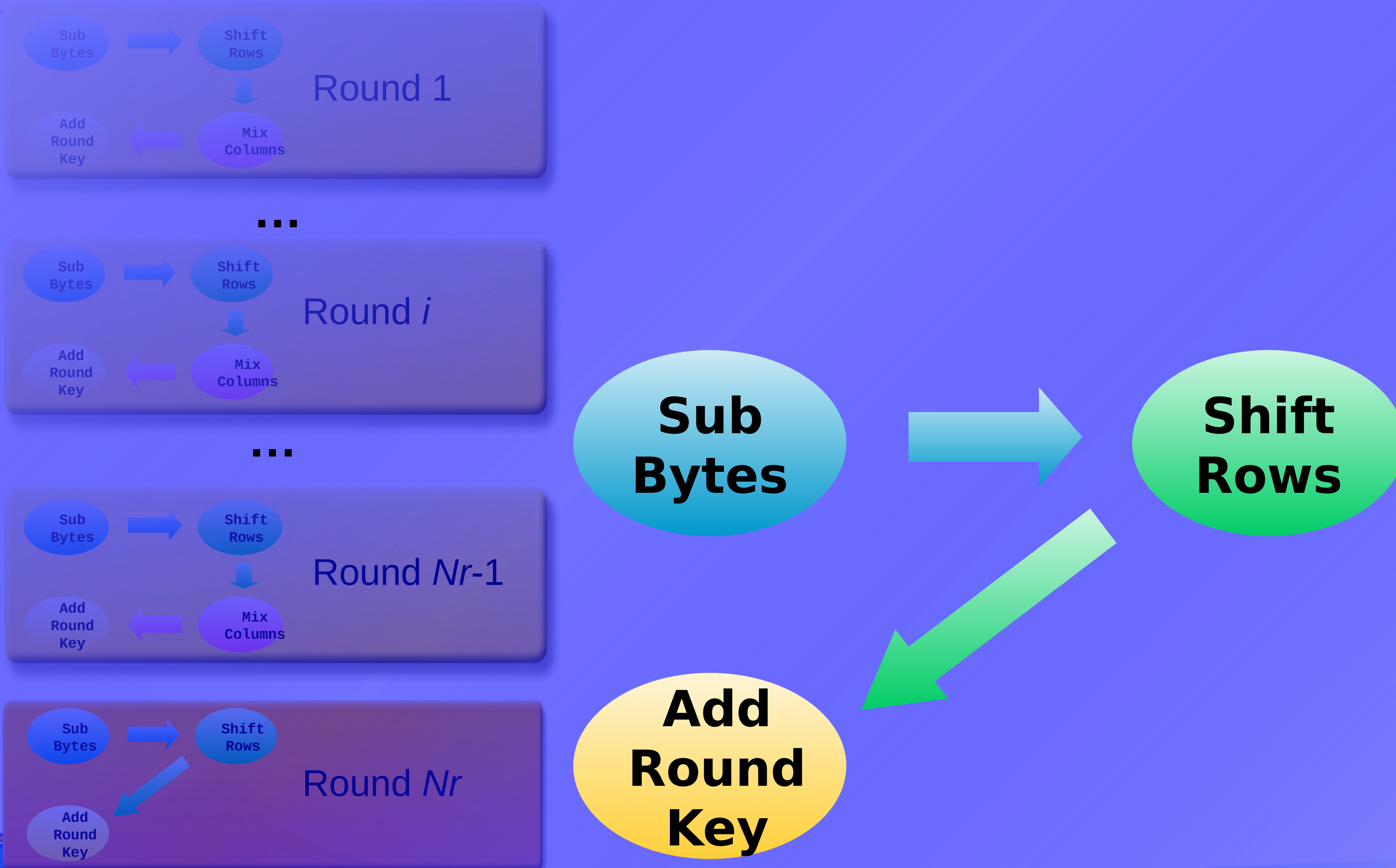
Nb=8

S_{00}	S_{01}	S_{02}	S_{03}	S_{04}	S_{05}	S_{06}	S_{07}
S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}	S_{16}	S_{17}
S_{20}	S_{21}	S_{22}	S_{23}	S_{24}	S_{25}	S_{26}	S_{27}
S_{30}	S_{31}	S_{32}	S_{33}	S_{34}	S_{35}	S_{36}	S_{37}

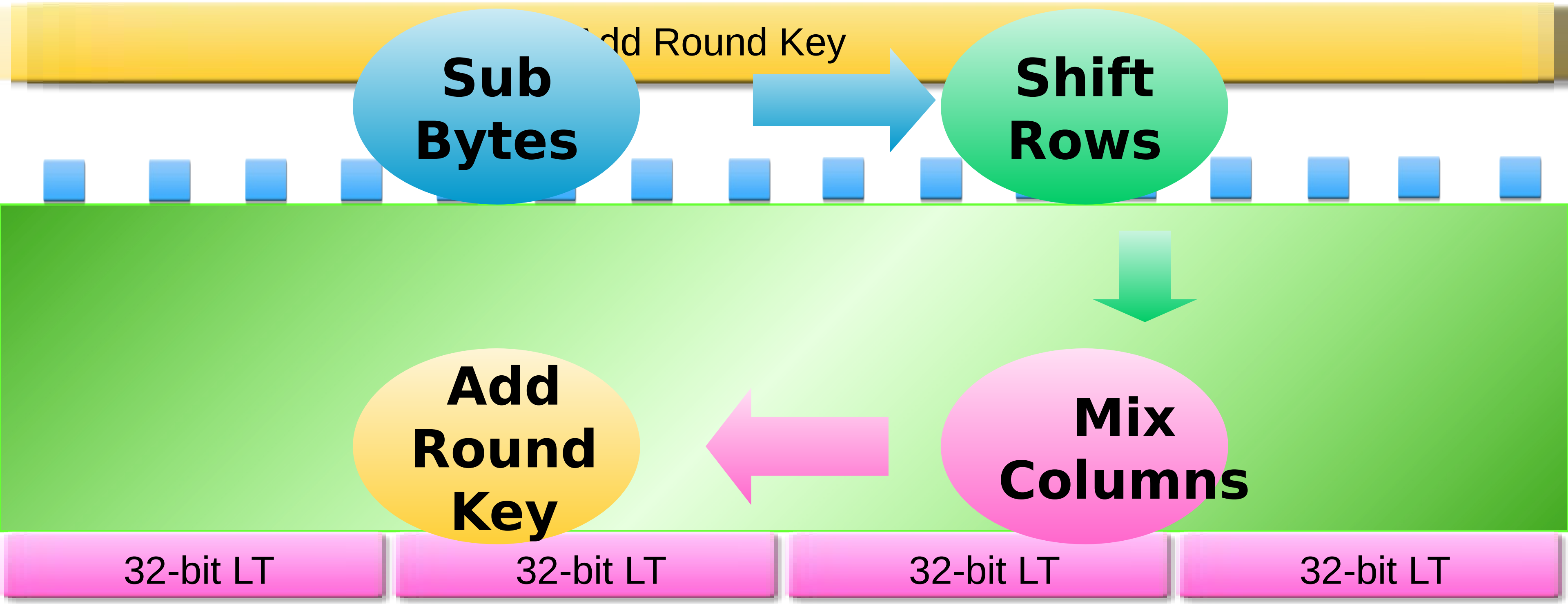
Normal cycles



Last cycle


Sub Bytes
Shift Rows
Mix Columns
Add Round Key
Round 1
...
Sub Bytes
Shift Rows
Mix Columns
Add Round Key
Round *i*
...
Sub Bytes
Shift Rows
Mix Columns
Add Round Key
Round *Nr*-1
Sub Bytes
Shift Rows
Add Round Key
Round *Nr*
Sub Bytes
Shift Rows
Add Round Key

Substitution-Permutation Network architecture

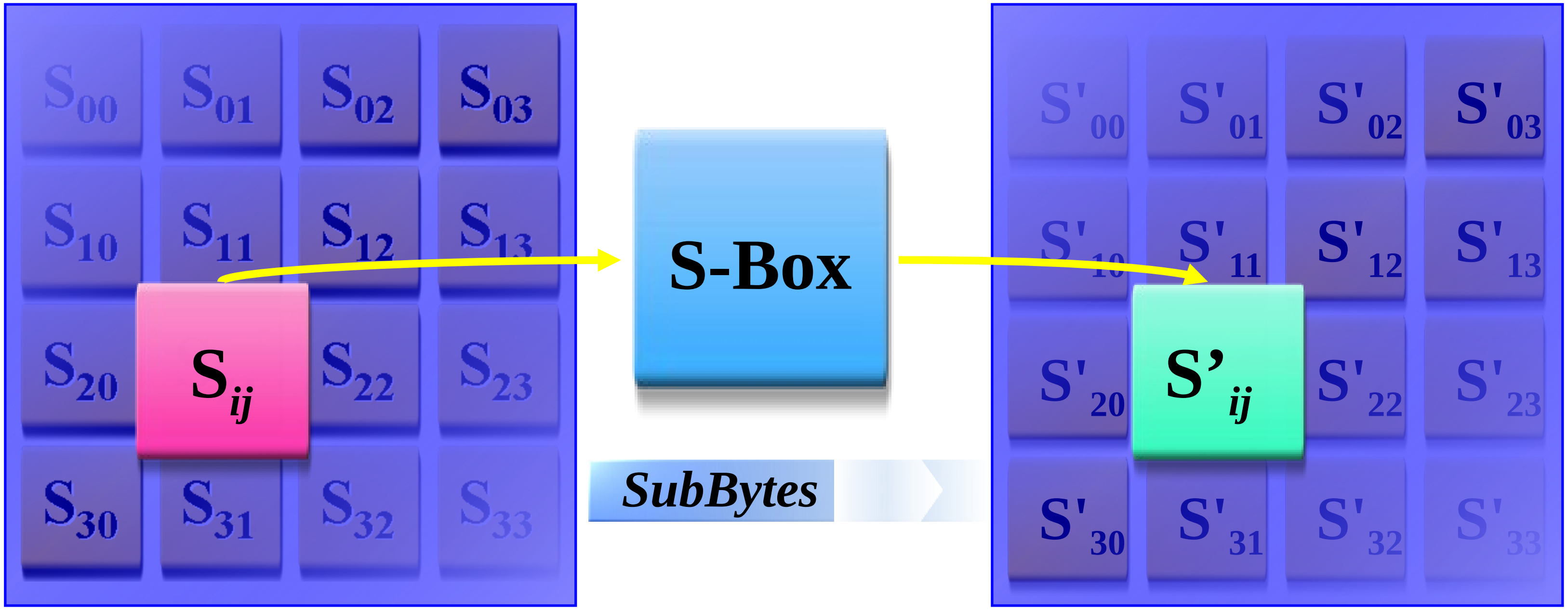


Rijndael algorithm encryption

```
Cipher( byte in[4 * Nb], byte out[4 * Nb], word w[Nb*(Nr + 1)])
begin
    byte      state[4,Nb]
    state =   in
    AddRoundKey(state, w)
    for round = 1 to Nr – 1
        SubBytes(state)
        ShiftRows(state)
        MixColumns(state)
        AddRoundKey(state, w + round * Nb)
    end for
    SubBytes(state)
    ShiftRows(state)
    AddRoundKey(state, w + Nr * Nb)
    out = state
end
```

SubBytes

- Nonlinear byte substitution via substitution table (S-box)
- Acts independently on each byte in the current state



SubBytes

- Process of replacing byte x in **SubBytes**:
  - Determine the inverse element x^{-1} (Binary form is $\{x_7x_6x_5x_4x_3x_2x_1x_0\}$), with $\{00\}^{-1} = \{00\}$
  - Affine: $y_i = x_i \oplus x_{(i+4) \bmod 8} \oplus x_{(i+5) \bmod 8} \oplus x_{(i+6) \bmod 8} \oplus x_{(i+7) \bmod 8} \oplus c_i$
  - with $\{c_7c_6c_5c_4c_3c_2c_1c_0\} = \{63\}$

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

InvSubBytes

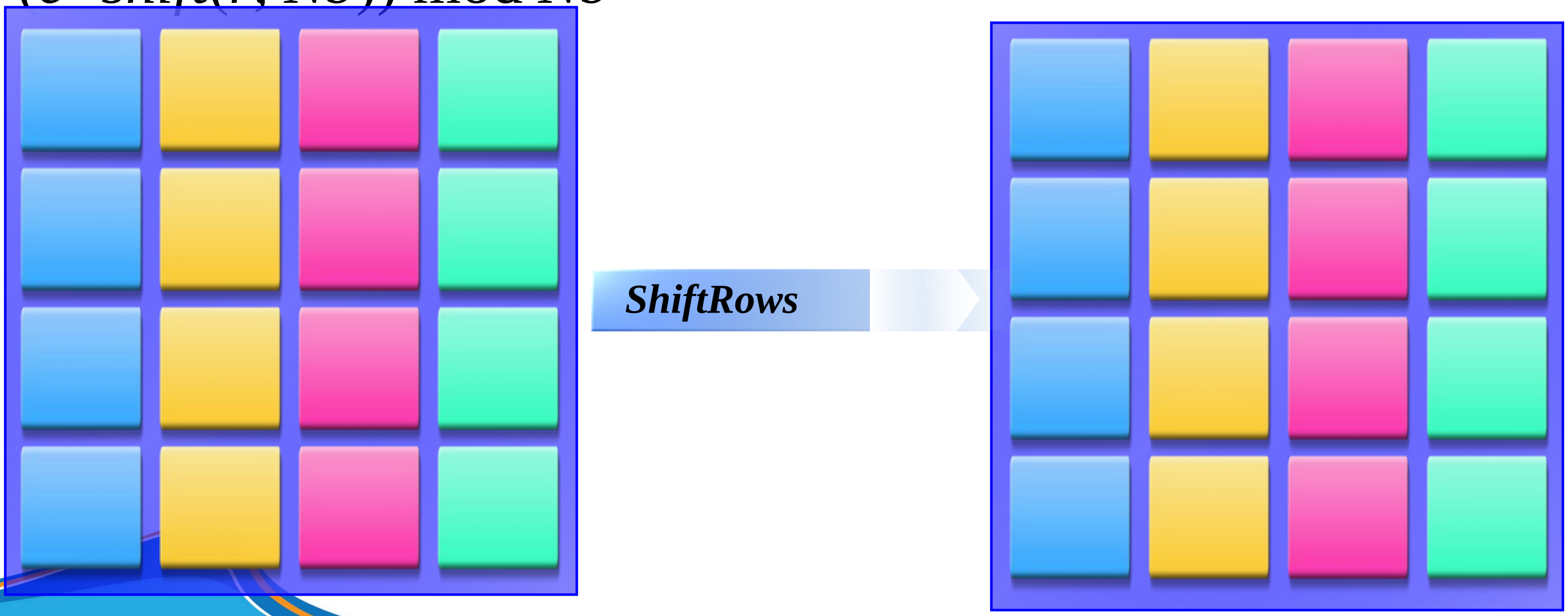
- Process of replacing byte y in **InvSubBytes**:
- Affine:
  - $x_i = y_{(i+2) \bmod 8} \oplus y_{(i+5) \bmod 8} \oplus y_{(i+7) \bmod 8} \oplus d_i$ với $\{d_7d_6d_5d_4d_3d_2d_1d_0\} = \{05\}$
- Determine the inverse element $x^{-1} \in GF(2^8)$ of x ($\{00\}^{-1} = \{00\}$)

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

ShiftRows

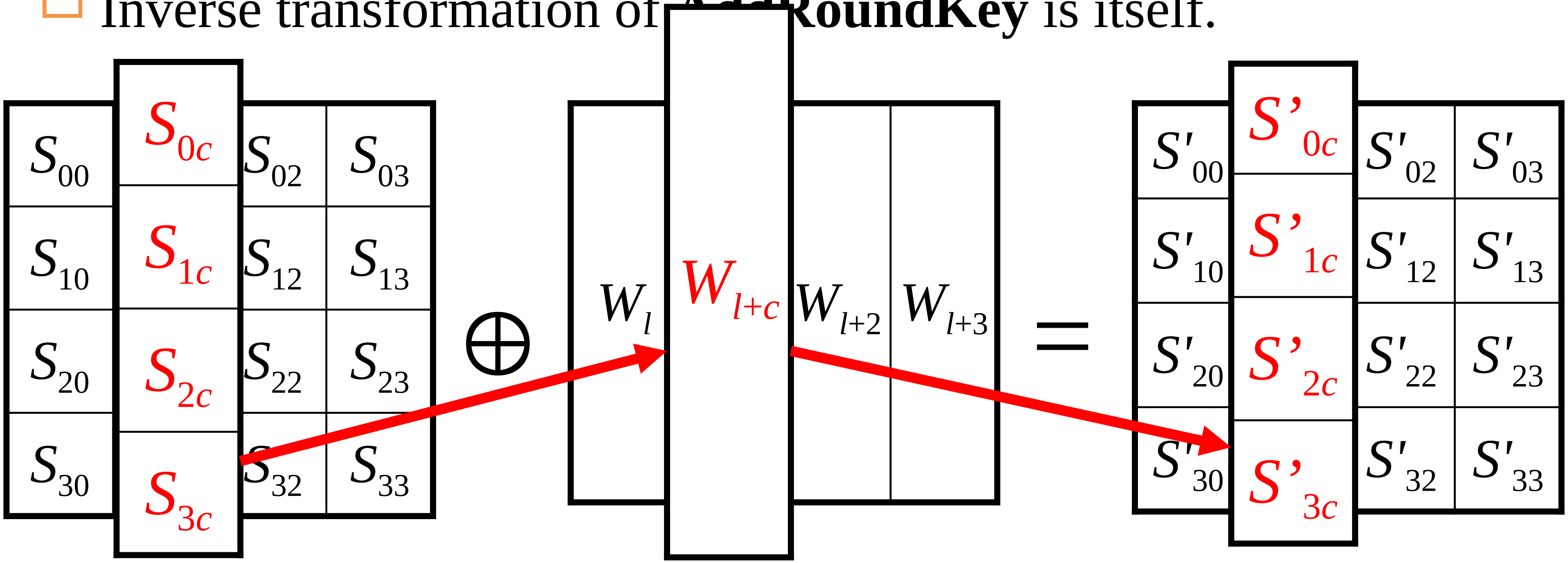
***shift*(*r*, *Nb*)**		***r***		
		1	**2**	**3**
Nb	**4**	**1**	**2**	**3**
	6	**1**	**2**	**3**
	8	**1**	**3**	**4**

- Each line of the current state is rotated by a number of places
- Byte $s_{r,c}$ at row r column c changes to column (c + *shift*(r, Nb)) mod Nb
- **InvShiftRows**: Byte $s_{r,c}$ at row r column c changes to column (c- *shift*(r, Nb)) mod Nb



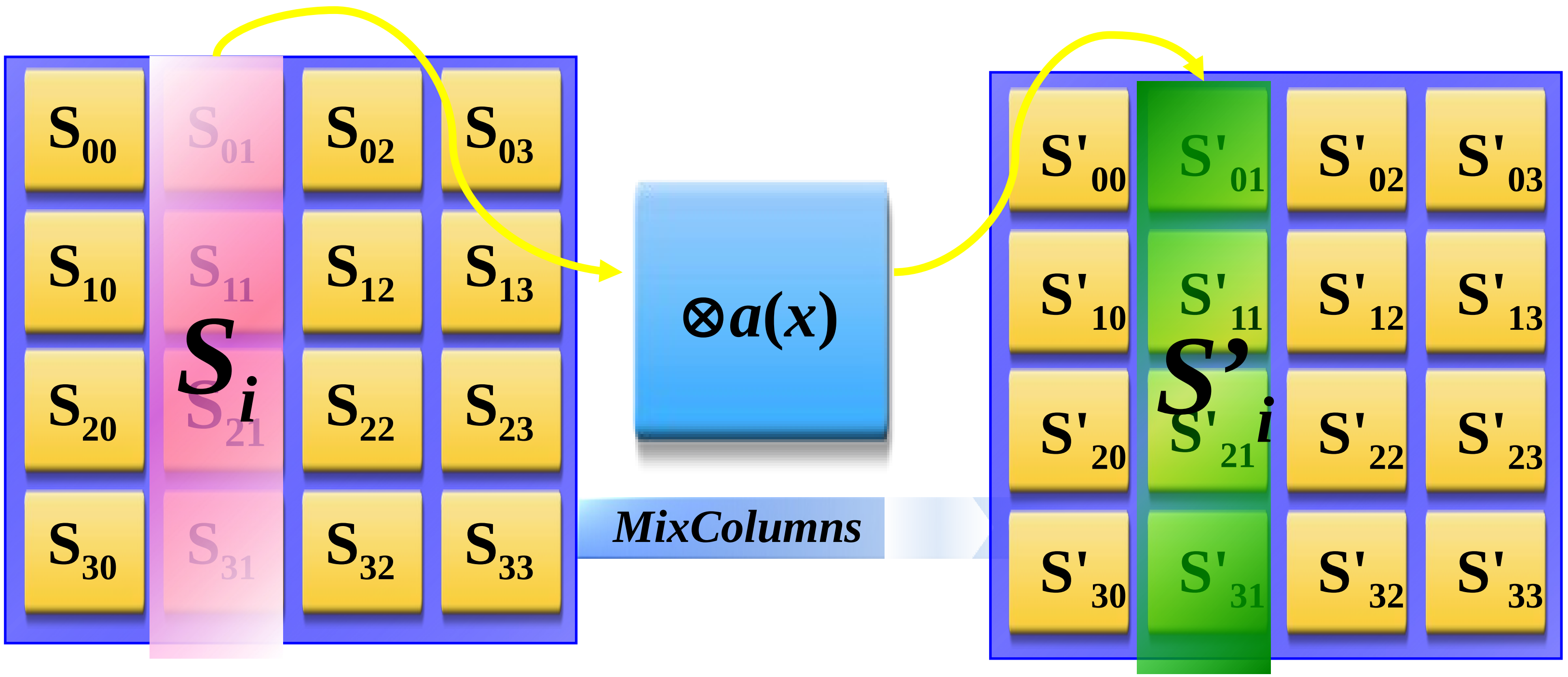
AddRoundKey

- Each byte of the status will XOR with corresponding byte in key-cipher of the current cycle: $s'_{r,c} = k_{r,c} \oplus s_{r,c}$, $0 \le r < 4$, $0 \le c < Nb$
- Inverse transformation of **AddRoundKey** is itself.



$l = round*Nb$

MixColumns


S00
S01
S02
S03
S10
S11
S12
S13
S20
S21
S22
S23
S30
S31
S32
S33
Si
⊗a(x)
MixColumns
S'00
S'01
S'02
S'03
S'10
S'11
S'12
S'13
S'20
S'21
S'22
S'23
S'30
S'31
S'32
S'33
S'i

MixColumns

- Each column of the current state is represented as a polynomial s(x) with coefficients in $GF(2^8)$.
- Perform multiplication: $s'(x) = a(x) \otimes s(x)$
- With: $a(x) = \{03\}x^3+\{01\}x^2+\{01\}x+\{02\}$

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

InvMixColumns

- Each column of the current state is represented as a polynomial s(x) with coefficients in GF(2^8).
- Perform multiplication: $s'(x) = a^{-1}(x) \otimes s(x)$
- With: $a^{-1}(x) = \{0b\}x^3+\{0d\}x^2+\{09\}x+\{0e\}$

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Generate a key-cipher for each cycle

- The extended key table is a 1-dimensional array of words (4 bytes in length).
- Function that generates the extended key cipher table depends on the value *Nk*, that is, on the length of the primary key

4 bytes

w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	w_{11}	w_{12}	w_{13}	w_{14}	w_{15}	w_{16}	w_{17}	…
Key for cycle 0						Key for cycle 1						Key for cycle 2						…

Extended key cipher table and how to determine key of the period (*Nb* = 6 (192-*bit* data), *Nk* = 4 (128-*bit* key))

Generate a key-cipher for each cycle

```
KeyExpansion(byte key[4 * Nk], word w[Nb * (Nr + 1)], Nk)
begin
    i=0
    while (i < Nk)
        w[i] = word[key[4*i], key[4*i+1],  key[4*i+2],key[4*i+3]]
        i = i + 1
    end while
    i = Nk
    while (i < Nb * (Nr + 1))
        word temp = w[i - 1]
        if  (i mod Nk = 0) then
            temp = SubWord(RotWord(temp)) xor Rcon[i / Nk]
        else
            if (Nk = 8) and (i mod Nk = 4) then temp = SubWord(temp)
        end if
        w[i] = w[i - Nk] xor temp
        i = i + 1
    end while
end
```

Generate a key-cipher for each cycle

- The set of constants is generated according to the following rule
  - $Rcon[i] = (RC[i], \{00\}, \{00\}, \{00\})$ với $RC[i] \in GF(2^8)$
  - $RC[i] = \{02\} \bullet RC[i - 1] = x^{(i-1)}$
  - Example
    - $Rcon[1] = (x^0 = \{01\}, \{00\}, \{00\}, \{00\})$
    - $Rcon[2] = (x^1 = \{02\}, \{00\}, \{00\}, \{00\})$
    - $Rcon[3] = (x^2 = \{04\}, \{00\}, \{00\}, \{00\})$
    - …
    - $Rcon[8] = (x^7 = \{80\}, \{00\}, \{00\}, \{00\})$
    - $Rcon[9] = (x^8 = \{1B\}, \{00\}, \{00\}, \{00\})$
      - Explain: $x^8 = \{100000000\} \bmod x^8 + x^4 + x^3 + x + 1 = \{100011011\} = \{1B\}$
    - $Rcon[10] = (x^9 = \{36\}, \{00\}, \{00\}, \{00\})$
      - Explain: $x^9 = \{1000000000\} \bmod x^8 + x^4 + x^3 + x + 1 = \{100011011\} = \{36\}$
    - …

S-box

Sbox		y															
		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
x	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

fit@hcmus

S-Box^{-1}

Sbox^{-1}		*y*															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Guidance of AES128

- AES128(M, K) // $|K| = |M| = 128$ bits
  - $(K_0, K_1, \ldots, K_{10}) \leftarrow$ expand(K) // $K_0 \leftarrow K$
  - $s \leftarrow M \oplus K_0$
  - Loop $r = 1 \rightarrow 10$
    - $s \leftarrow$ s-box(s)
    - $s \leftarrow$ shift-rows(s)
    - If $r \leq 9$ then $s \leftarrow$ mix-cols(s)
    - $s \leftarrow s \oplus K_r$
  - Return s

- expand(K) // $|K| = 128$ bits
  - $K_0 \leftarrow K$
  - Loop $i = 1 \rightarrow 10$
    - $K_i[0] \leftarrow K_{i-1}[0] \oplus$ s-box($K_{i-1}[3]$ <<< 8) $\oplus C_i$
    - $K_i[1] \leftarrow K_{i-1}[1] \oplus K_i[0]$
    - $K_i[2] \leftarrow K_{i-1}[2] \oplus K_i[1]$
    - $K_i[3] \leftarrow K_{i-1}[3] \oplus K_i[2]$
  - Return $(K_0, K_1, \ldots, K_{10})$

Guidance of AES128 (expand)

- expand(K) // $|K|$ = 128 bits
  - $K_0 \leftarrow K$
  - Loop $i = 1 \rightarrow 10$
    - $K_i[0] \leftarrow K_{i-1}[0] \oplus$ s-box($K_{i-1}[3]$ <<< 8) $\oplus C_i$
    - $K_i[1] \leftarrow K_{i-1}[1] \oplus K_i[0]$
    - $K_i[2] \leftarrow K_{i-1}[2] \oplus K_i[1]$
    - $K_i[3] \leftarrow K_{i-1}[3] \oplus K_i[2]$
  - Return $(K_0, K_1, \ldots, K_{10})$
- Notes:
  - The input is usually a 16-byte key sequence, ví dụ "Thats my Kung Fu"
  - The algorithm deals with integers, each subkey consists of 4 integers
    - Each integer 4 bytes
    - Use type int

Guidance of AES128 (expand)

- expand(K) // $|K|$ = 128 bits
  - $K_0 \leftarrow K$
  - Loop $i = 1 \rightarrow 10$
    - $K_i[0] \leftarrow K_{i-1}[0] \oplus$ s-box($K_{i-1}[3]$ <<< 8) $\oplus C_i$
    - $K_i[1] \leftarrow K_{i-1}[1] \oplus K_i[0]$
    - $K_i[2] \leftarrow K_{i-1}[2] \oplus K_i[1]$
    - $K_i[3] \leftarrow K_{i-1}[3] \oplus K_i[2]$
  - Return (K_0, K_1, …, K_{10})
- Table of constants C:

i	**Values**
C_0	0x00000000
C_1	0x01000000
C_2	0x02000000
C_3	0x04000000
C_4	0x08000000

C_5	0x10000000
C_6	0x20000000
C_7	0x40000000
C_8	0x80000000
C_9	0x1b000000
C_{10}	0x36000000

Guidance of AES128 (expand)

- Some notes
  - Convert from 4 bytes to 1 int: char *s*[] = "abcd" → int *a* = 0x61626364
  - Convert from 1 int to 4 bytes: int *a* = 0x61626364 → char *s*[] = "abcd"
  - Same for transferring 16 bytes → 4 ints and 4 ints → 16 bytes
- Example to test expand:

Keys	Values
K_0	54 68 61 74 73 20 6d 79 20 4b 75 6e 67 20 46 75
K_1	e2 32 fc f1 91 12 91 88 b1 59 e4 e6 d6 79 a2 93
K_2	56 08 20 07 c7 1a b1 8f 76 43 55 69 a0 3a f7 fa
K_3	d2 60 0d e7 15 7a bc 68 63 39 e9 01 c3 03 1e fb
K_4	a1 12 02 c9 b4 68 be a1 d7 51 57 a0 14 52 49 5b
K_5	b1 29 3b 33 05 41 85 92 d2 10 d2 32 c6 42 9b 69
K_6	bd 3d c2 87 b8 7c 47 15 6a 6c 95 27 ac 2e 0e 4e
K_7	cc 96 ed 16 74 ea aa 03 1e 86 3f 24 b2 a8 31 6a
K_8	8e 51 ef 21 fa bb 45 22 e4 3d 7a 06 56 95 4b 6c
K_9	bf e2 bf 90 45 59 fa b2 a1 64 80 b4 f7 f1 cb d8
K_{10}	28 fd de f8 6d a4 24 4a cc c0 a4 fe 3b 31 6f 26

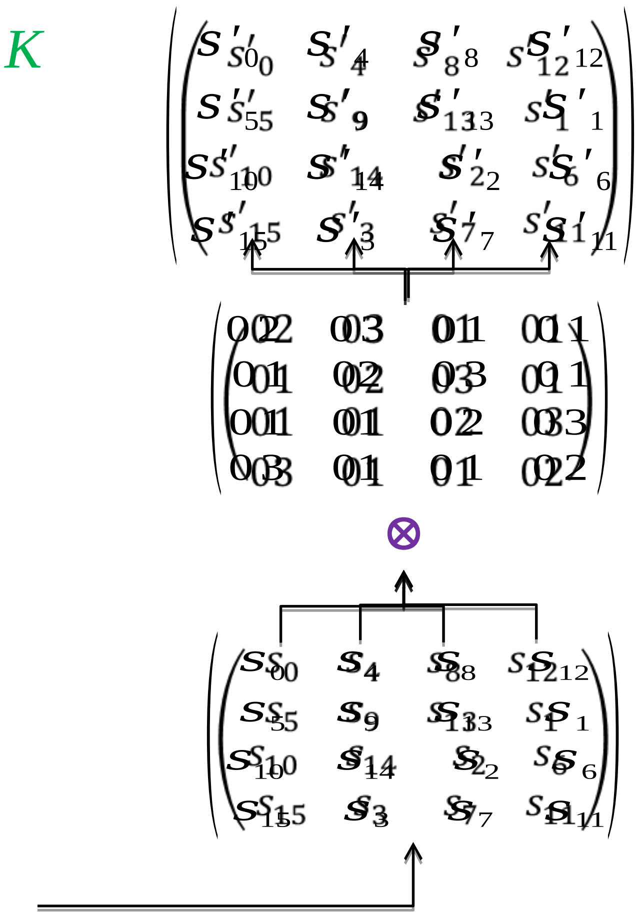
Thats my Kung Fu

Guidance of AES128 (AES128)

- AES128(M, K) // $|K| = |M| = 128$ bits
  - $(K_0, K_1, \ldots, K_{10}) \leftarrow$ expand(K) // $K_0 \leftarrow K$
  - $s \leftarrow M \oplus K_0$
  - Loop $r = 1 \rightarrow 10$
    - $s \leftarrow$ s-box(s)
    - $s \leftarrow$ shift-rows(s)
    - if $r \leq 9$ then $s \leftarrow$ mix-cols(s)
    - $s \leftarrow s \oplus K_r$
  - Return s
- Notes:
  - Value s: $s_0s_1s_2s_3s_4s_5s_6s_7s_8s_9s_{10}s_{11}s_{12}s_{13}s_{14}s_{15}$ stands for a 16-byte string ($s = M \oplus K_0$)
  - Function s-box: $[0, 255] \rightarrow [0, 255]$ follows S-box table

Guidance of AES128 (AES128)

- AES128(M, K) // $|K| = |M| = 128$ bits
  - (K_0, K_1, …, K_{10}) ← expand(K) // $K_0 \leftarrow K$
  - $s \leftarrow M \oplus K_0$
  - Loop $r = 1 \rightarrow 10$
    - $s \leftarrow$ s-box(s)
    - $s \leftarrow$ shift-rows(s)
    - if $r \leq 9$ then $s \leftarrow$ mix-cols(s)
    - $s \leftarrow s \oplus K_r$
  - Return s
- Notes:
  - shift-rows
    - Input: $s_0s_1s_2s_3s_4s_5s_6s_7s_8s_9s_{10}s_{11}s_{12}s_{13}s_{14}s_{15}$
    - Output: $s_0s_5s_{10}s_{15}s_4s_9s_{14}s_3s_8s_{13}s_2s_7s_{12}s_1s_6s_{11}$



Guidance of AES128 (AES128)

- Test-vector for AES128
  - Key: 54|68|61|74|73|20|6d|79|20|4b|75|6e|67|20|46|75 (“Thats my Kung Fu”)
  - Message: 54|77|6f|20|4f|6e|65|20|4e|69|6e|65|20|54|77|6f (“Two One Nine Two”)
  - Cipher-text:
    - C_0: 00 1f 0e 54 3c 4e 08 59 6e 22 1b 0b 47 74 31 1a
    - C_1: 58 47 08 8b 15 b6 1c ba 59 d4 e2 e8 cd 39 df ce
    - C_2: 43 c6 a9 62 0e 57 c0 c8 09 08 eb fe 3d f8 7f 37
    - C_3: 78 76 30 54 70 76 7d 23 99 3c 37 5b 4b 39 34 f1
    - C_4: b1 ca 51 ed 08 fc 54 e1 04 b1 c9 d3 e7 b2 6c 20
    - C_5: 9b 51 20 68 23 5f 22 f0 5d 1c bd 32 2f 38 91 56
    - C_6: 14 93 25 77 8f a4 2b e8 c0 60 24 40 5e 0f 92 75
    - C_7: 53 39 8e 5d 43 06 93 f8 4f 0a 3b 95 85 52 57 bd
    - C_8: 66 25 3c 74 70 ce 5a a8 af d3 0f 0a a3 73 13 54
    - C_9: 09 66 8b 78 a2 d1 9a 65 f0 fc e6 c4 7b 3b 30 89
    - C_{10}: 29 c3 50 5f 57 14 20 f6 40 22 99 b3 1a 02 d7 3a

Intermediate cipher-text of AES128

Final cipher-text of AES128