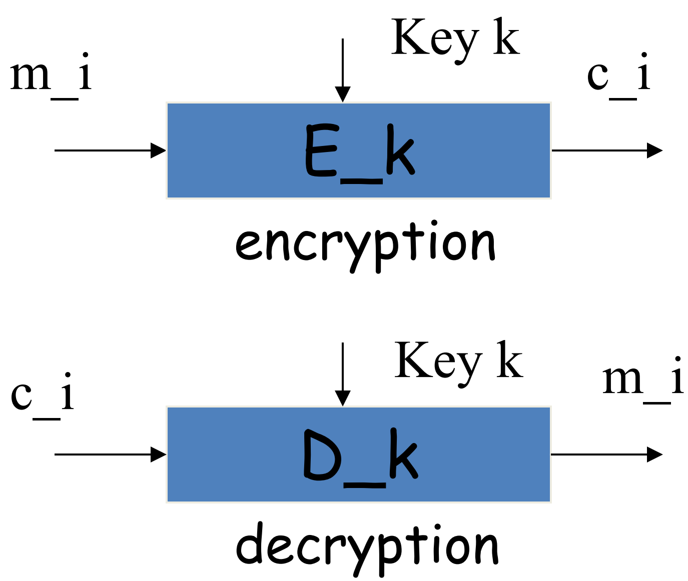
Advanced Encryption Standard

Cunsheng Ding
HKUST, Hong Kong, CHINA

Block Ciphers

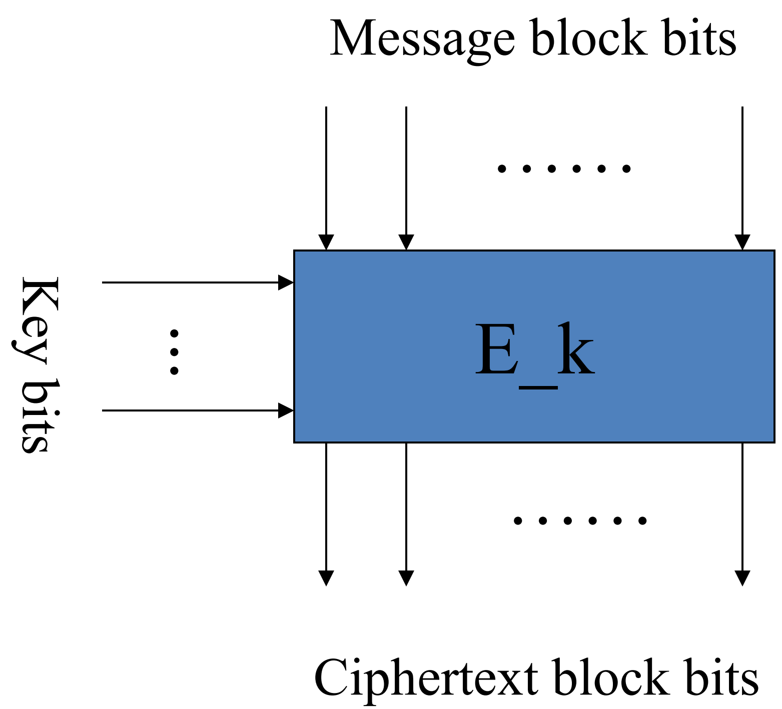
Pictorial discription



- Definition: the ciphertext $c=E_k(m)$ is time-independent, and depends only on m and k.
- Example:
  - $E_k(m_i) = m_i + k$, where + is the bitwise exclusive-or operation.
- How to design a secure block cipher?
  - Two basic security requirements.

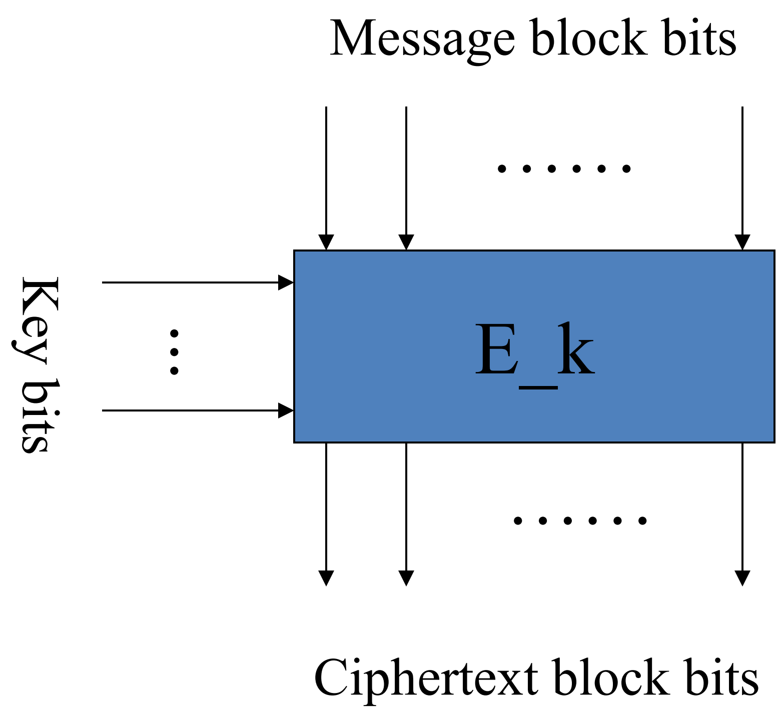
Shannon's Idea of Diffusion

- Good diffusion
  - Every message block bit is involved in many ciphertext block bits.
  - Every key bit is involved in many ciphertext block bits.
  - Linear functions for diffusion purpose



Shannon's Idea of Confusion

- Good confusion
  - Every ciphertext bit is a function of message block bits and key bits.
  - These functions should be very complex in format.
  - Nonlinear functions for confusion



Design Ideas of Block Ciphers

- Use linear functions for diffusion purpose
- Use nonlinear functions for confusion purpose
- Iterate a simple round function a number of times.
- Almost all block ciphers follow these design strategies
- In this course, we introduce only
  - The Data Encryption Standard
  - The Advanced Encryption Standard

The Data Encryption Standard (DES)

- It is a “block” cipher with key length 56 bits.
- It was designed by IBM in 1976 for the National Bureau of Standards (NBS), with approval from the National Security Agency (NSA).
- It had been used as a standard for encryption until 2000.
- A new encryption standard was adopted in 2000, as a replacement of DES.

The Advanced Encryption Standard (AES)

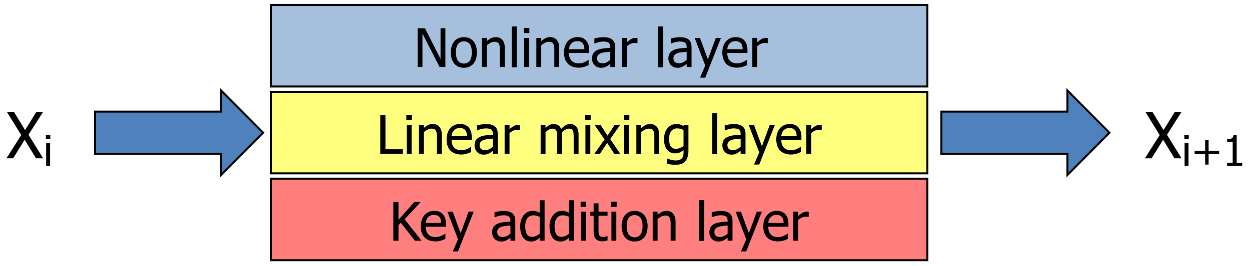
- A replacement for DES was needed because DES is subject to exhaustive key search attacks.
- US NIST issued a call for ciphers in 1997
- 15 candidates accepted in Jun 98
- 5 were shortlisted in Aug-99
- Rijndael was selected as the AES in Oct-2000
- Issued as FIPS PUB 197 standard in Nov-2001

AES Design Requirements

- A symmetric block cipher
- 128-bit plaintext block
- 128/192/256-bit keys
- Stronger & faster than “Triple-DES”
- Active life of 20-30 years
- Efficient in both software and hardware implementations
- Simple in design
- Suitable for smart cards (memory requirement)

Rijndael: Design Techniques

- Shannon's idea of diffusion and confusion
- Iterated block cipher
- The round function has the form



Rijndael: Key and Block Size

Key Size (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Plaintext block size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Number of rounds	10	12	14
Round key size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Expanded key size (words/bytes)	44/176	52/208	60/240

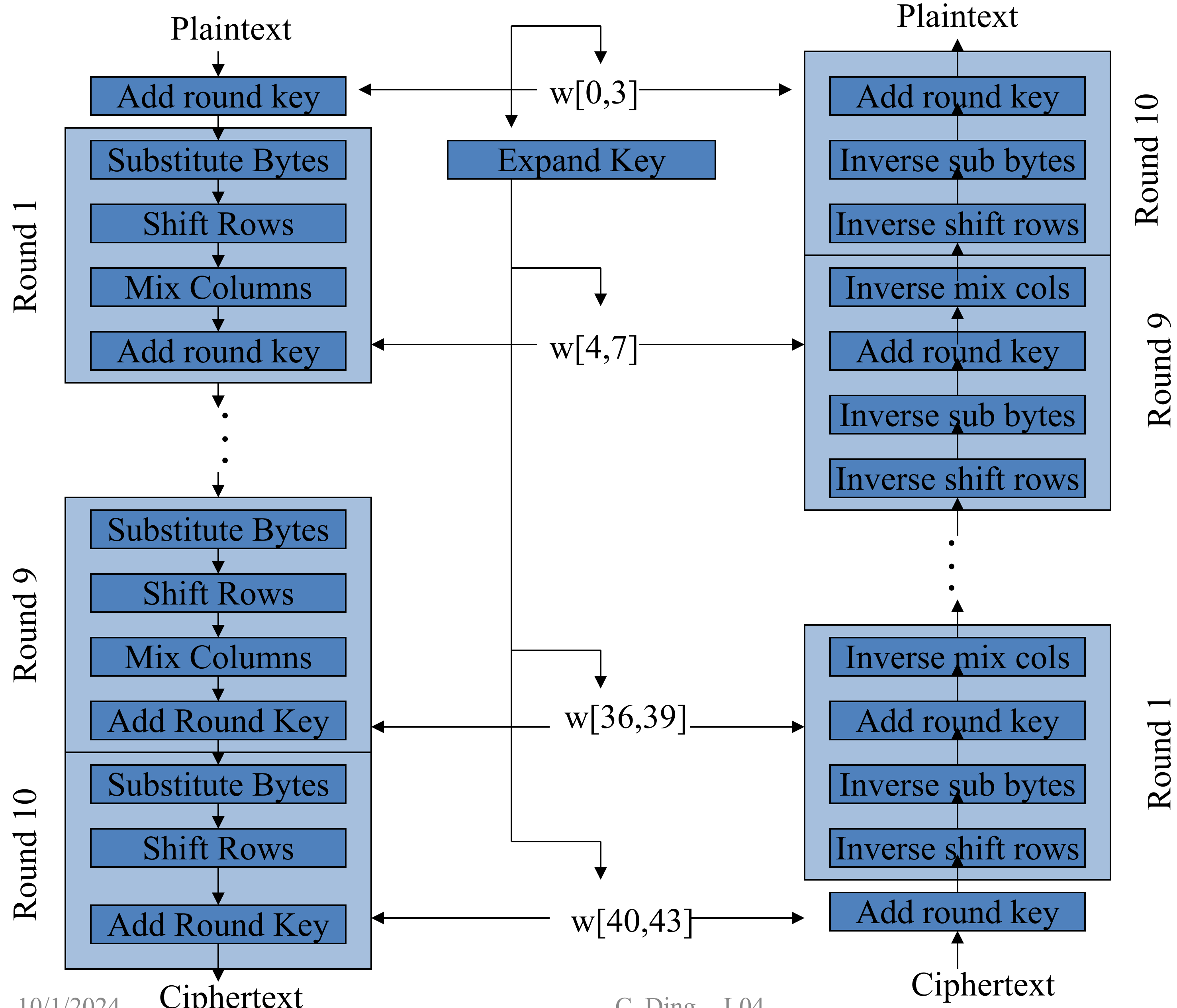
Main Steps

- An initial round-key addition
- 9/11/13 rounds, corresponds to 128/192/256 bit keys
- A final round, similar to other round, but without mixed column operations
- In summary, 10/12/14 rounds, corresponds to 128/192/256 bit keys

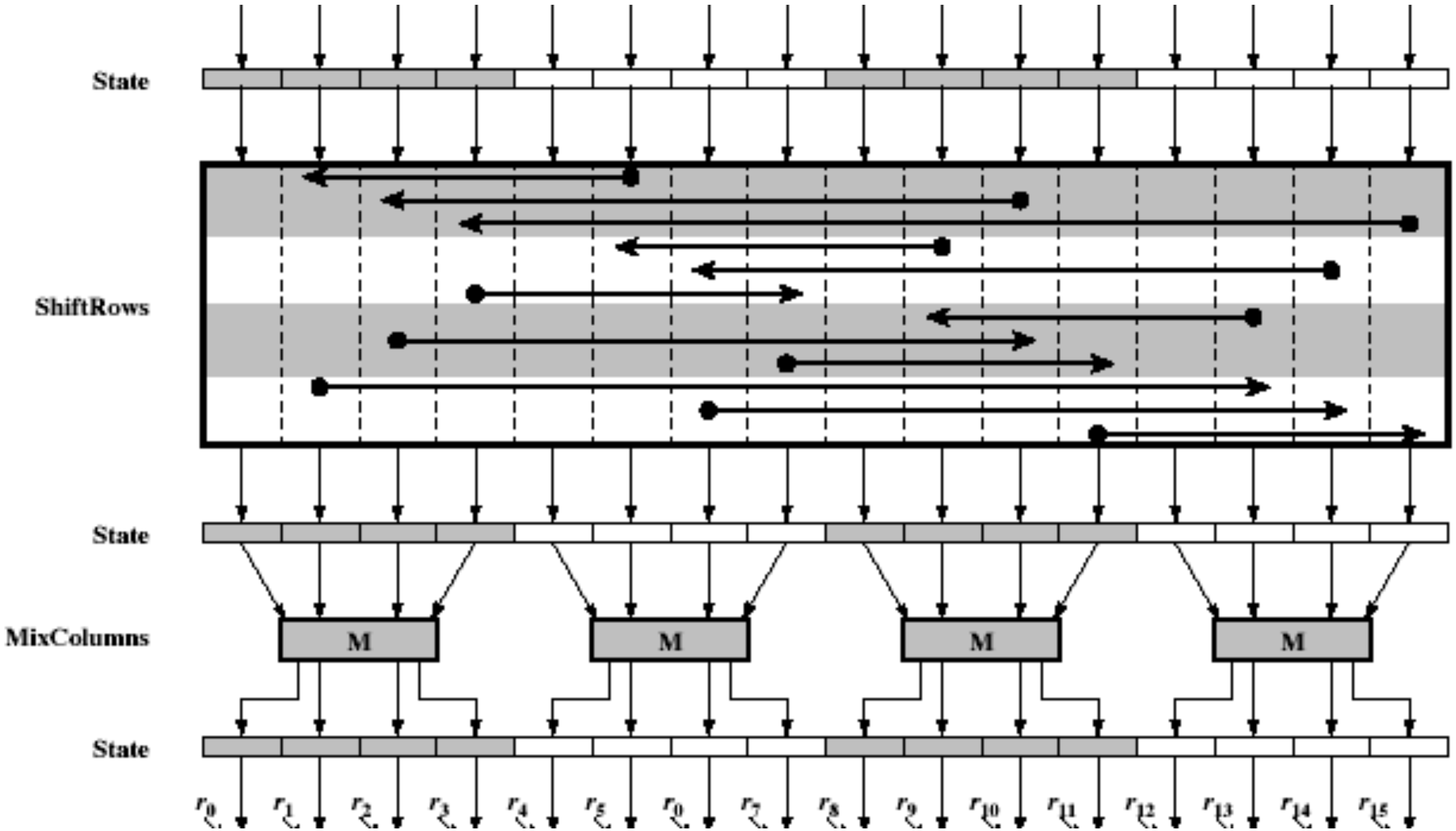
The Internal Structure of AES

- We will now consider the default case, i.e., the secret key has 128 bits.
- In this case, there are an initial round and 10 normal rounds of iteration.
- We will introduce all building blocks except one.

AES Encryption & Decryption



AES Round Function



Key and State Bytes in Rectangular Arrays

$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	$k_{0,6}$	$k_{0,7}$
$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	$k_{1,6}$	$k_{1,7}$
$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$	$k_{2,4}$	$k_{2,5}$	$k_{2,6}$	$k_{2,7}$
$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$	$k_{3,4}$	$k_{3,5}$	$k_{3,6}$	$k_{3,7}$

Variable Key size :

16, 24 or 32 bytes

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$	$a_{0,6}$	$a_{0,7}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$	$a_{1,6}$	$a_{1,7}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$	$a_{2,6}$	$a_{2,7}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$	$a_{3,6}$	$a_{3,7}$

Variable State size:

16, 24 or 32 bytes

Round Function : ByteSub

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

$a_{i,\,j}$ → S-box → $b_{i,\,j}$

$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$
$b_{1,0}$	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$
$b_{2,0}$	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$
$b_{3,0}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$

ByteSub acts on individual bytes of the State.

Every byte is identified as an element in in GF(2^8)

$S(y) = y^{254}$

Round Function : ShiftRow

$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$	no shift	$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$
$b_{1,0}$	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$	cyclic shift by 3	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$	$b_{1,0}$
$b_{2,0}$	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$	cyclic shift by 2	$b_{2,2}$	$b_{2,3}$	$b_{2,0}$	$b_{2,1}$
$b_{3,0}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$	cyclic shift by 1	$b_{3,3}$	$b_{3,0}$	$b_{3,1}$	$b_{3,2}$

ShiftRow operates on the rows of the State.
The purpose is to provide inter column diffusion.

Round Function : MixColumn

$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$
$b_{1,1}$	$b_{1,2}$	$b_{1,3}$	$b_{1,0}$
$b_{2,2}$	$b_{2,3}$	$b_{2,0}$	$b_{2,1}$
$b_{3,3}$	$b_{3,0}$	$b_{3,1}$	$b_{3,2}$

$\times$ C(x)

$d_{0,0}$	$d_{0,1}$	$d_{0,2}$	$d_{0,3}$
$d_{1,0}$	$d_{1,1}$	$d_{1,2}$	$d_{1,3}$
$d_{2,0}$	$d_{2,1}$	$d_{2,2}$	$d_{2,3}$
$d_{3,0}$	$d_{3,1}$	$d_{3,2}$	$d_{3,3}$

MixColumn operates on the columns of the State.

The columns of the State are considered as polynomials of degree 3 over $GF(2^8)$ and multiplied module x^4+1 with a fixed polynomial c(x) :
$c(x) = 03x^3 + 01x^2 + 01x + 02$
Each element in $GF(2^8)$ is expressed as uv, where u, v in
{0, 1, …, 9, A, B, C, D, E, F}, where A=:10, …, F=:15.
MixColumn is implemented using operations of XOR, conditional bit-shifts.
Purpose – inter-byte diffusion within columns (based on ECC theory)
Together with ShiftRow, it ensures that after a few rounds, all output bits depend on all input bits. Coefficients of the matrix were also chosen for efficient implementation.

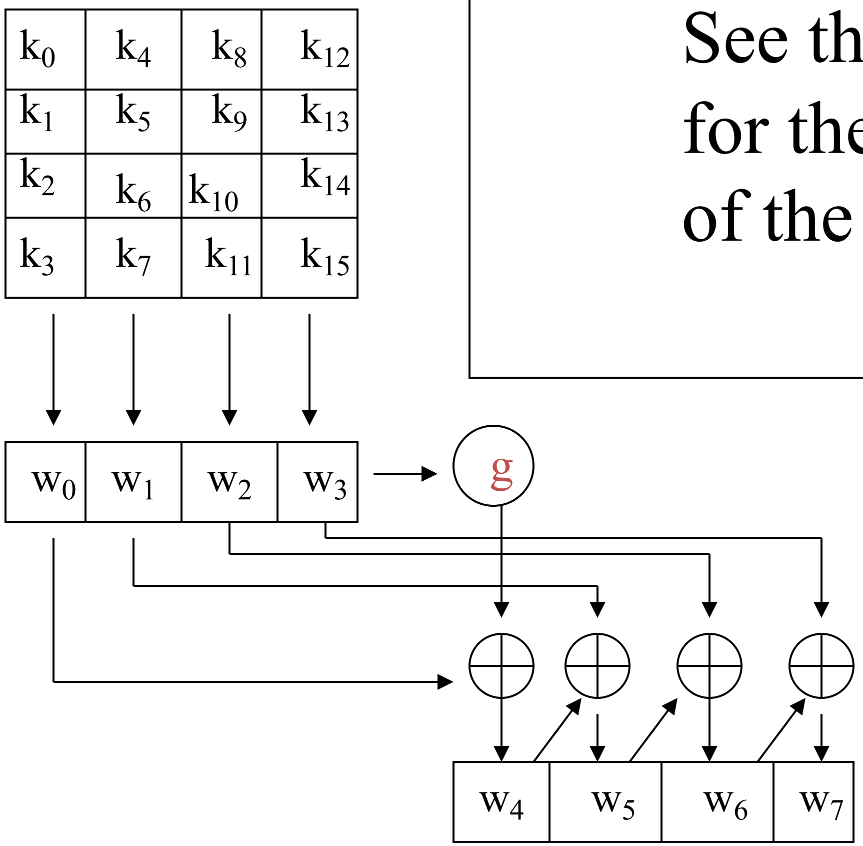
Round Function : AddRoundKey

$$\begin{array}{|c|c|c|c|}\hline d_{0,0} & d_{0,1} & d_{0,2} & d_{0,3} \\ \hline d_{1,0} & d_{1,1} & d_{1,2} & d_{1,3} \\ \hline d_{2,0} & d_{2,1} & d_{2,2} & d_{2,3} \\ \hline d_{3,0} & d_{3,1} & d_{3,2} & d_{3,3} \\ \hline \end{array} \oplus \begin{array}{|c|c|c|c|}\hline k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ \hline k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ \hline k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ \hline k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} \\ \hline \end{array} = \begin{array}{|c|c|c|c|}\hline e_{0,0} & e_{0,1} & e_{0,2} & e_{0,3} \\ \hline e_{1,0} & e_{1,1} & e_{1,2} & e_{1,3} \\ \hline e_{2,0} & e_{2,1} & e_{2,2} & e_{2,3} \\ \hline e_{3,0} & e_{3,1} & e_{3,2} & e_{3,3} \\ \hline \end{array}$$

In AddRoundKey, the Round Key is bitwise XORed to the State.
Purpose is to make round function key-dependent.

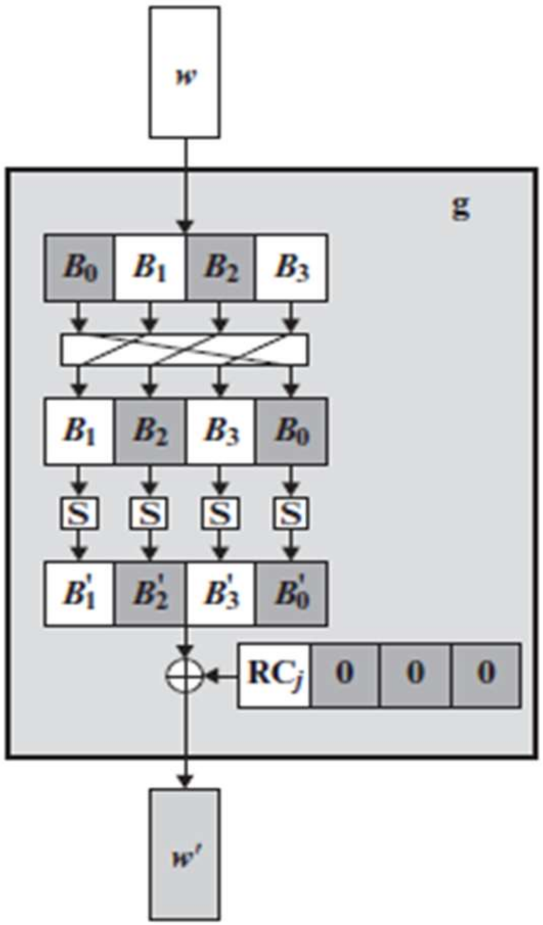
Key-XORing with plaintext or ciphertext is sometimes called **whitening**. This is a cheap way of adding to the security of the cipher by preventing the collection of plaintext-ciphertext pairs.

AES Key Expansion



See the next page for the definition of the function g

The Function g



- Each RC_j is a round constant
- RC_j has one byte
- For definition of RC_j, see the official documentation or a textbook on cryptography

Decryption

- Not identical to encryption
- May need different implementations if encryption and decryption are needed
- Quite often only encryption needed
  - Digest

Conclusions

- The design of the round function is simple
- Symmetric and parallel structure
- Efficient implementation in both software and hardware
- Secure against all known cryptanalytic attacks
- Suitable modern processors
- Suitable for smart cards (8-bit processor)
- It is widely used in real-world security systems