SOMAIYA VIDYAVIHAR UNIVERSITY
K J Somaiya College of Engineering

Modern Block Ciphers- DES & AES

Ms. Swati Mali
swatimali@somaiya.edu
Somaiya Vidyavihar University

Module 3: Symmetric Key Cryptography

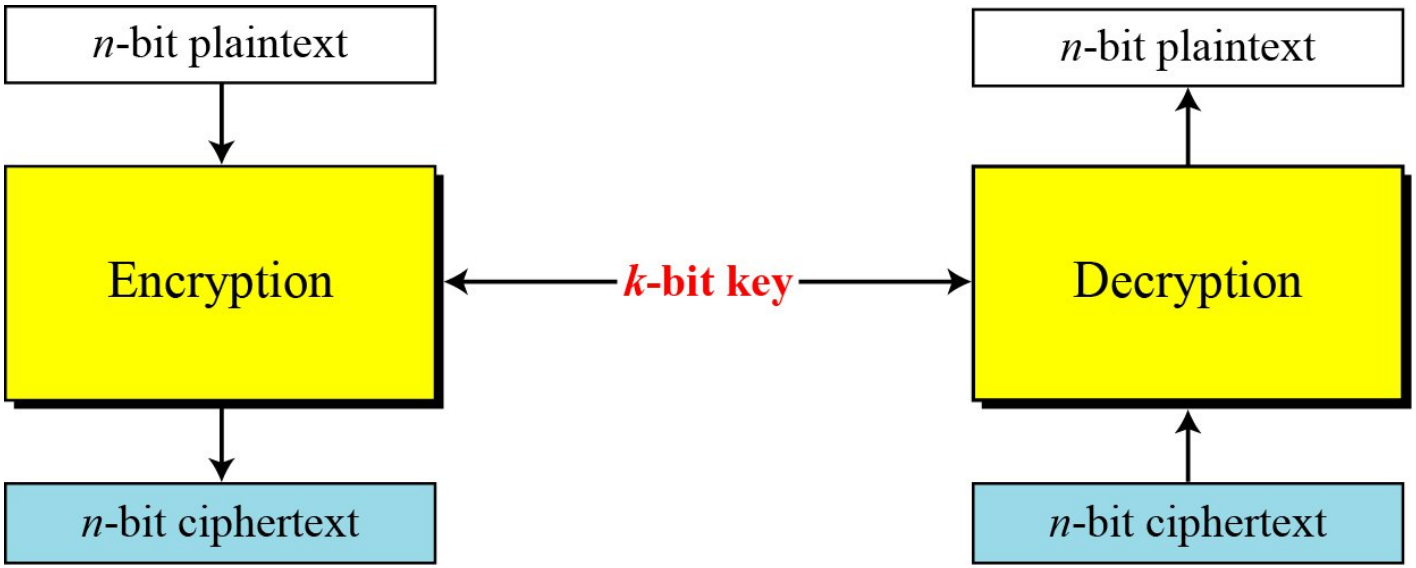
3.1 Building blocks of modern and classical Block Ciphers: P box, S Box, EX-OR operations, circular shifts, swaps, split and combine, Rounds, Initialization vectors, Confusion, Diffusion, Fiestel Ciphers, Non-Fiestel ciphers

3.2 Introduction to classical block ciphers as means of confusion and diffusion techniques : Initialization vector, Electronic code book[ECB], Cipher feedback[CFB], cipher block chaining[CBC], output feedback[OFB], propagating cipher block chaining[PCBC], counter[CTR]

3.3 DES: DES Structure, DES Analysis: Properties, Design Criteria, DES Strength and Weaknesses, DES Security, Multiple DES, 3DES, AES vs DES

MODERN BLOCK CIPHERS

- A symmetric-key modern block cipher :
  - encrypts an n-bit block of plaintext or
  - decrypts an n-bit block of ciphertext.
  - The encryption or decryption algorithm uses a k-bit key.



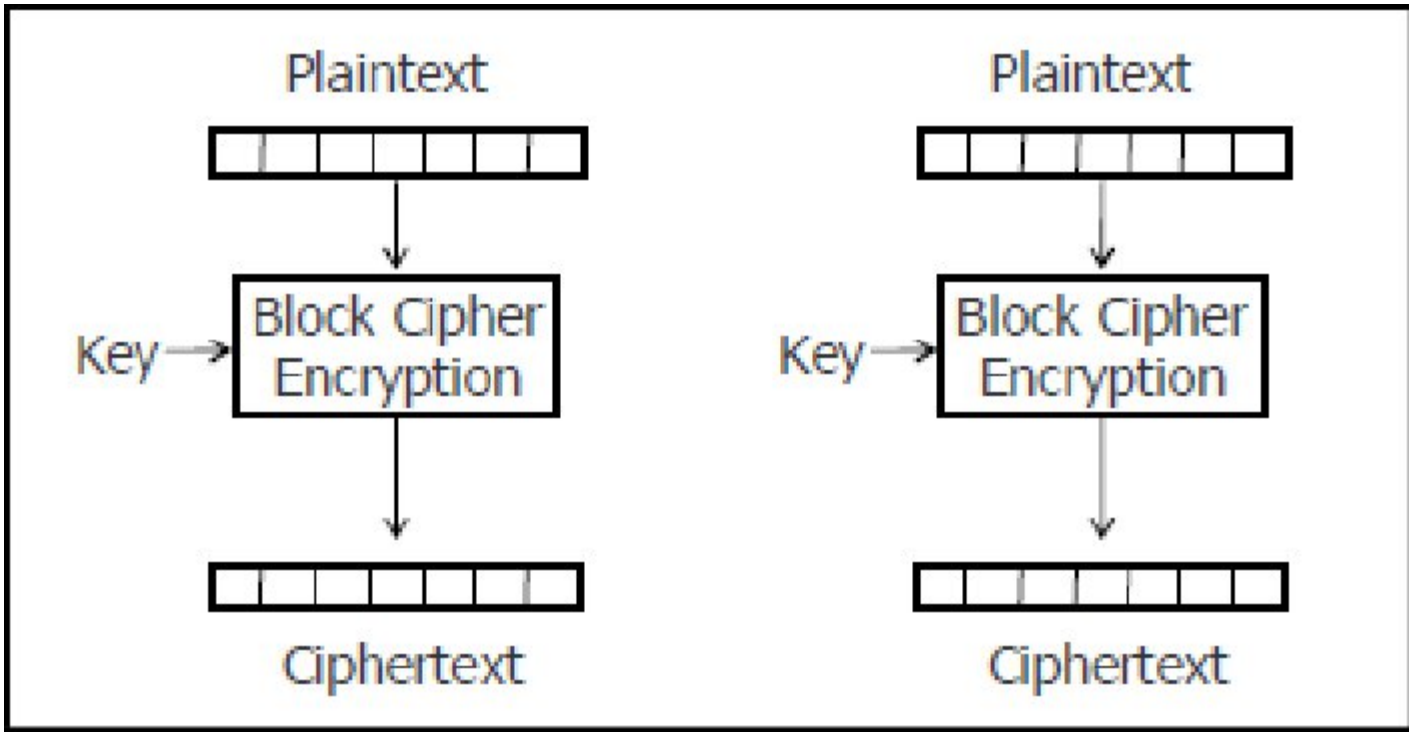
“must-know” vocabulary terms

- **Avalanche Effect** - Small change in input causes large change in output (important in ciphers & hashes)
- **Initialization Vector (IV)** – Random/non-repeating input ensuring same plaintext doesn’t give same ciphertext.
- **Padding** – Adding extra data to fit block size.
- **Rounds** – Repeated transformations in encryption (e.g., multiple rounds in AES, DES).

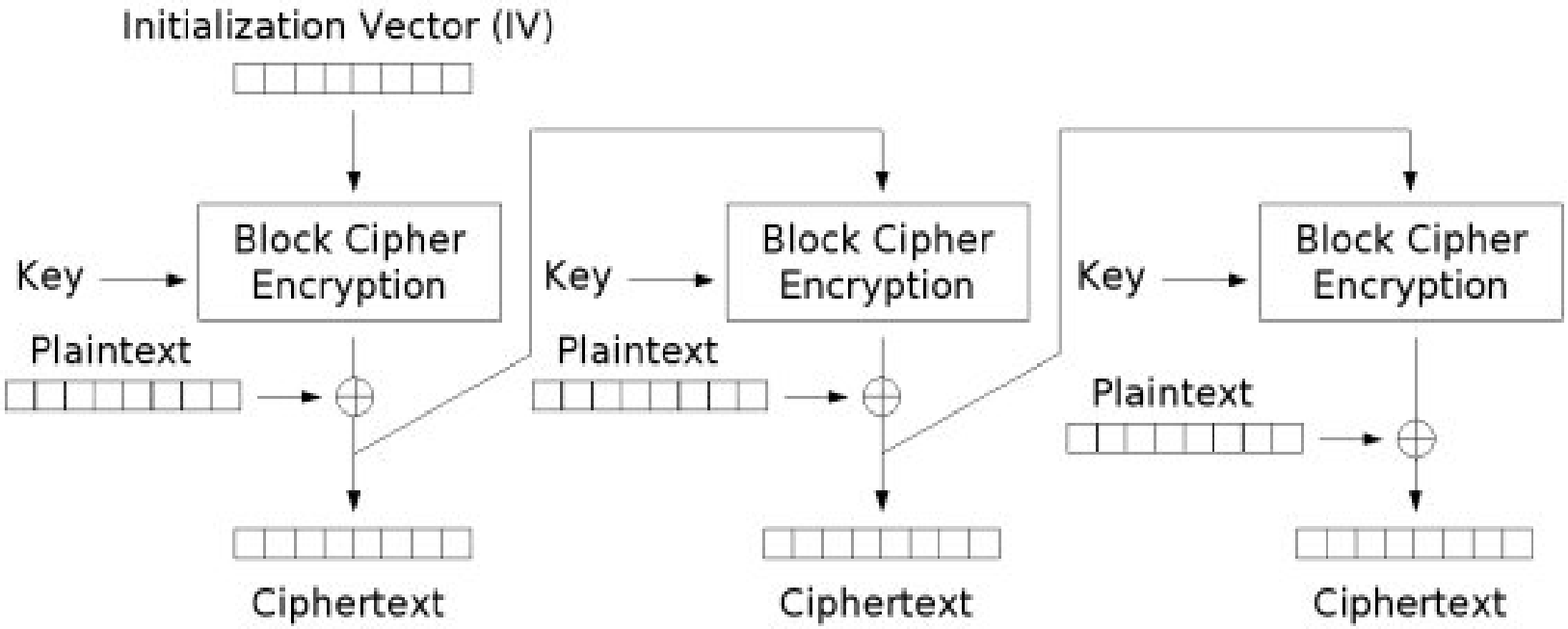
3.2 Classical block ciphers

- Electronic codebook mode
- Cipher feedback mode
- Output feedback mode
- Cipher block chaining with checksum

Electronic codebook mode

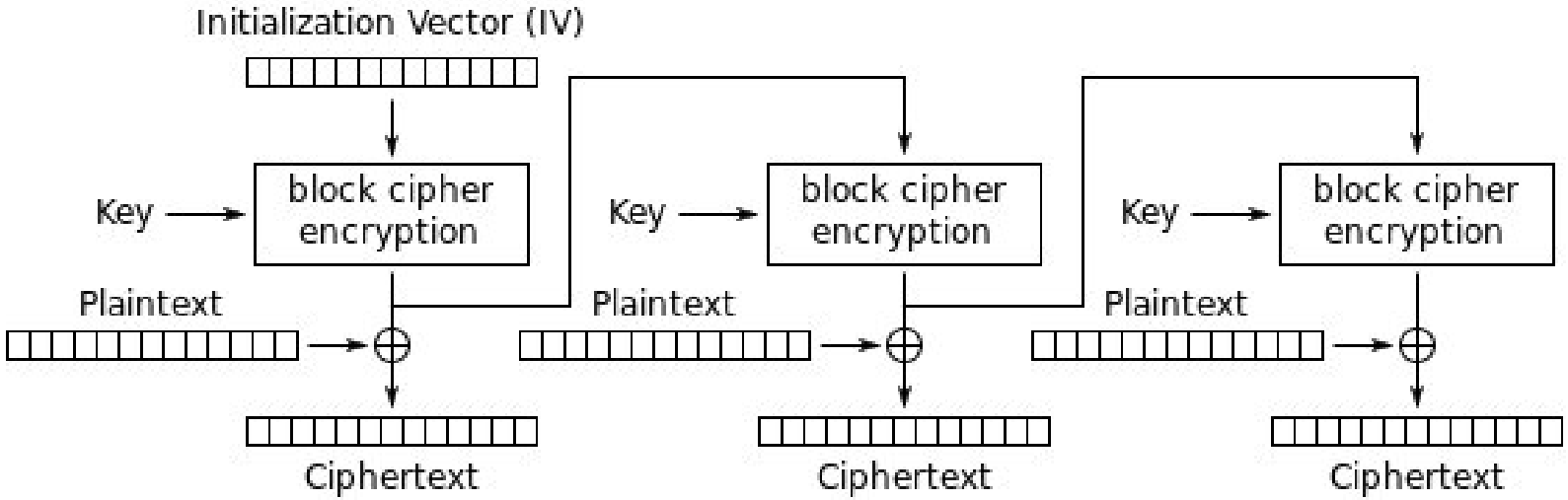


Cipher Feedback Mode



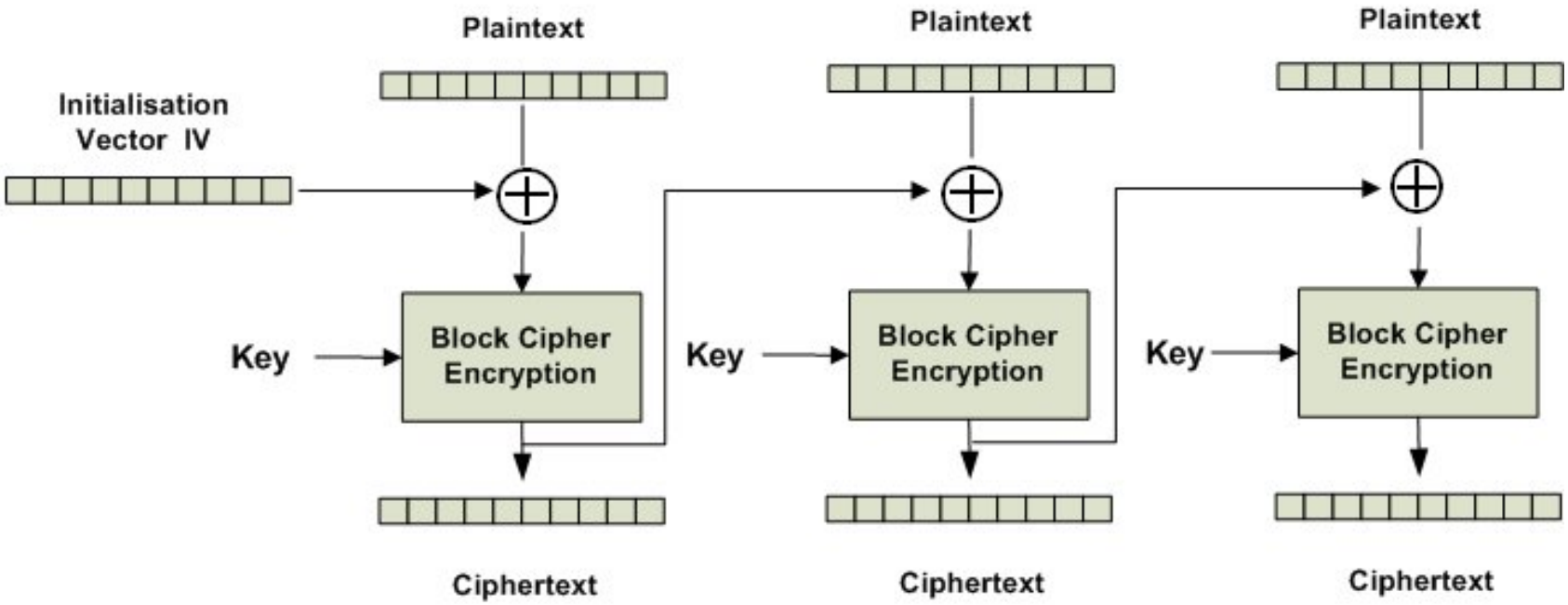
Cipher Feedback (CFB) mode encryption

Output Feedback Mode



Output Feedback (OFB) mode encryption

Cipher block chaining mode



Summary : Classical block cipher modes

ECB:	CBC:
Security:	***Security***:
- Plaintext patterns are not concealed.	+ Plaintext patterns are concealed by XORing with previous ciphertext block.
- Input to the block cipher is not randomized; it is the same as the plaintext.	+Input to the block cipher is randomized by XORing with the previous ciphertext block.
+More than one message can be encrypted with the same key.	+ More than one message can be encrypted with the same key.
- Plaintext is easy to manipulate, blocks can be removed, repeated, or interchanged.	+/- Plaintext is somewhat difficult to manipulate; blocks can be removed from the beginning and end of the message, bits of the first block can be changed, and repetition allows some controlled changes.
Efficiency:	***Efficiency***:
+ Speed is the same as the block cipher.	+ Speed is the same as the block cipher.
Ciphertext is up to one block longer than the plaintext, due to padding.	- Ciphertext is up to one block longer than the plaintext, not counting the IV.
- No preprocessing is possible.	- No preprocessing is possible.
+ Processing is parallelizable.	+/- Encryptions not parallelizable; decryption is parallelizable and has a random-access property.
Fault-tolerance:	***Fault-tolerance***:
- A ciphertext error affects one full block of plaintext.	- A ciphertext error affects one full block of plaintext and the corresponding bit in the next block.
- Synchronization error is unrecoverable.	- Synchronization error is unrecoverable.

CFB:	OFB/Counter:
Security:	***Security***:
+ Plaintext patterns are concealed.	+ Plaintext patterns are concealed.
+ Input to the block cipher is randomized.	+ Input to the block cipher is randomized.
+ More than one message can be encrypted with the same key provided that a different IV is used.	+ More than one message can be encrypted with the same key, provided that a different IV is used.
+/- Plaintext is somewhat difficult to manipulate;blocks can be removed from the beginning and end of the message, bits of the first block can be changed, and repetition allows some controlled changes.	- Plaintext is very easy to manipulate, any change in ciphertext directly affects the plaintext.
Efficiency:	***Efficiency***:
+ Speed is the same as the block cipher.	+ Speed is the same as the block cipher.
- Ciphertext is the same size as the plaintext, not counting the IV.	- Ciphertext is the same size as the plaintext, not counting the IV.
+/- Encryption is not parallelizable; decryption is parallelizable and has a random-access property.	+ Processing is possible before the message is seen.
- Some preprocessing is possible before a block is seen; the previous ciphertext block can be encrypted.	-/+ OFB processing is not parallelizable; counter processing is parallelizable.
+/- Encryption is not parallelizable; decryption is parallelizable and has a random-access property.	
Fault-tolerance:	***Fault-tolerance***:
- A ciphertext error affects the corresponding bit of plaintext and the next full block.	+ A ciphertext error affects only the corresponding bit of plaintext.
+Synchronization errors of full block sizes are recoverable. 1-bit CFB can recover from the addition or loss of single bits.	-Synchronization error is unrecoverable.

- Reference: Bruce Schneier, “Applied Cryptography”, 2nd Edition

Modern Block Cipher techniques

Modern Block Cipher techniques

- Confusion – substitution – S Box
- Diffusion – transposition – P Box
- Ex-OR operations
- Circular Shift
- Swap
- Split and combine
- Product cipher - complex cipher combining substitution, permutation, and other components (introduced by Shannon)
- rounds

Diffusion

- ***The idea of diffusion is to hide the relationship between the ciphertext and the plaintext.***

Diffusion hides the relationship between the ciphertext and the plaintext.

- ***Frustrate the adversary who uses ciphertext statistics to find plaintext***
- ***If a single symbol in plaintext is changes, several or all symbols in the ciphertext will also be changed***

Confusion

- ***The idea of confusion is to hide the relationship between the ciphertext and the key.***

Confusion hides the relationship between the ciphertext and the key.

- ***Frustate the adversary who uses ciphertext statistics to find key***
- ***If a single bit in the key is changed, most or all bits in the ciphertext will also be changed***

S-Box

- An S-box (substitution box) can be thought of as a miniature substitution cipher

An S-box is an $m \times n$ substitution unit, where m and n are not necessarily the same.

- *S Box can be keyed or keyless*
- *Modern Block ciphers normally use keyless S Box where the mapping from inputs to outputs is predetermined*

S-Box

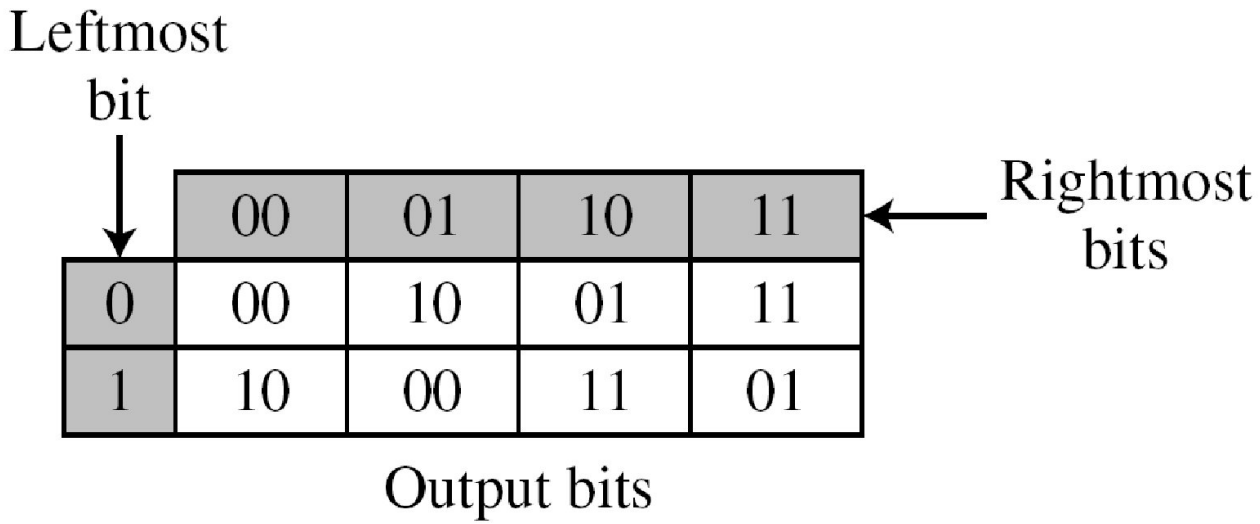
- S Box are substitution ciphers in which relationship between input and output is defined by
  - a table or
  - mathematical relation
    - E.g

$$y_1 = x_1 \oplus x_2 \oplus x_3 \qquad y_2 = x_1$$

If the input is 110 then the output y1= 0 and y2= 1
If the input is 001 then the output y1= 1 and y2= 0.

S-Box

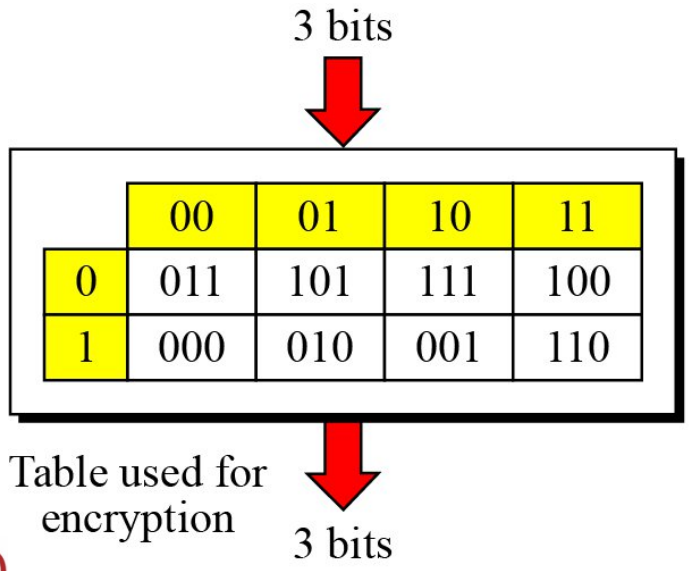
- Tabular s box example (non-invertible)

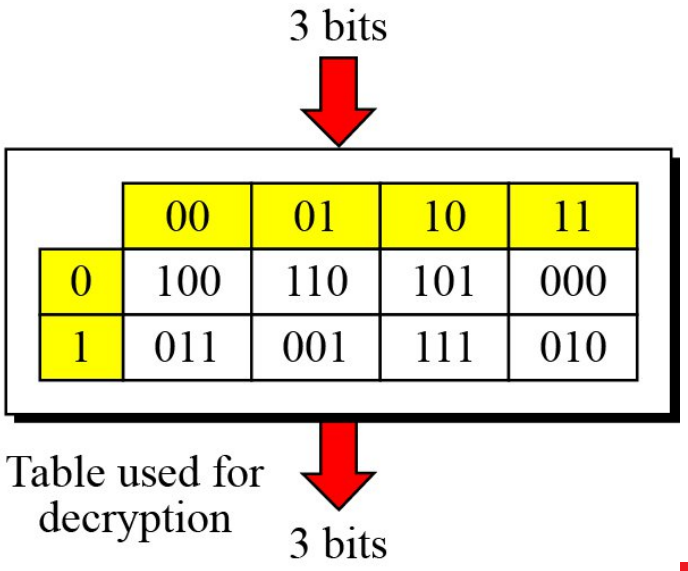


- For i/p = 001, o/p = 10
- First 1 bit makes row, rest two bits make column

Invertability of S-Box

- An S-box may or may not be invertible.
- In an invertible S-box, the number of input bits should be the same as the number of output bits
- Example : invertible S box

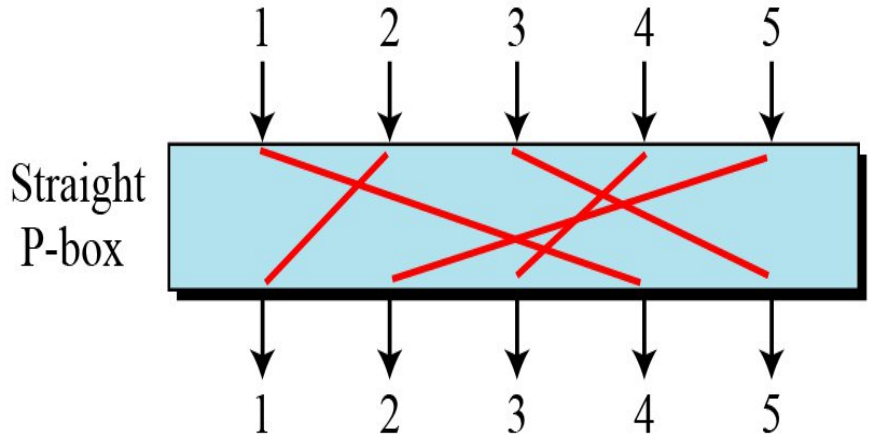


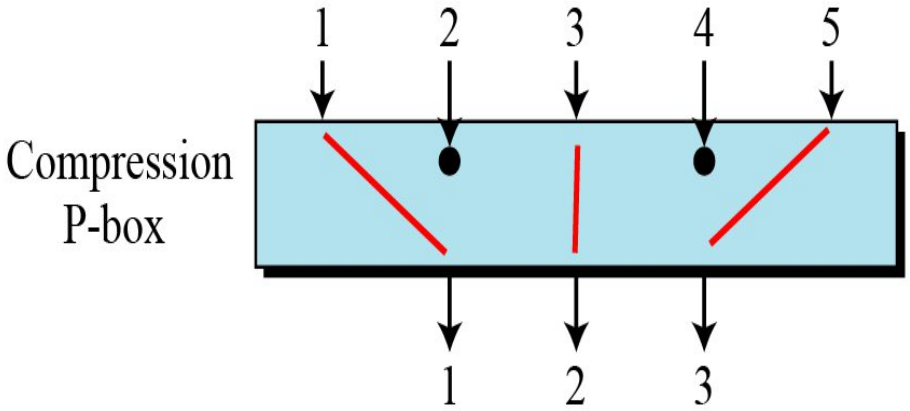


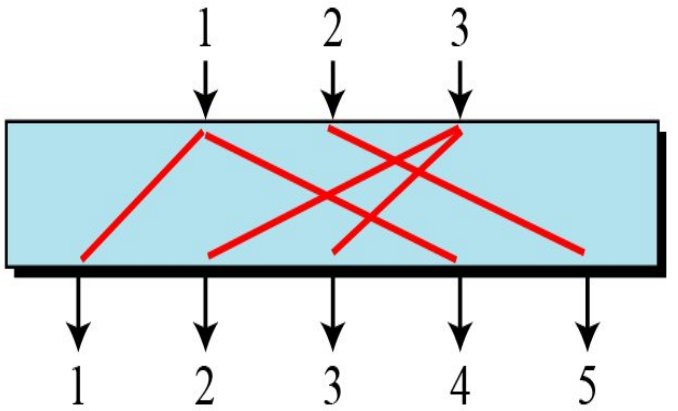
P-Box

- Permutation Box parallels the traditional transposition cipher for characters
- It transposes bits
- Types
  - Straight P box
  - Compression P box
  - Expansion P box

Types of P Boxes







Straight P box

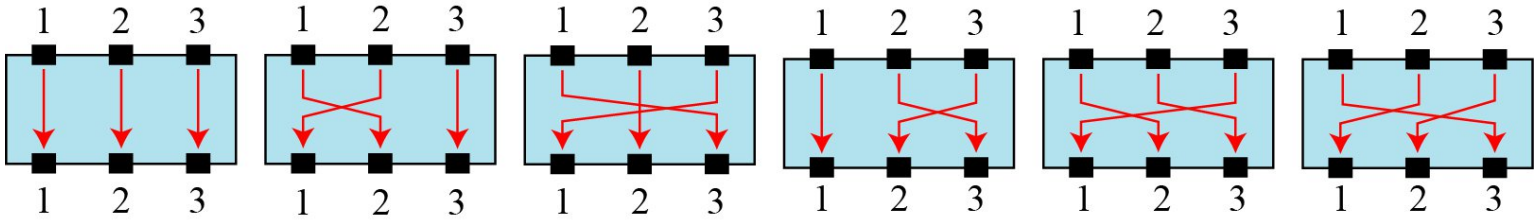
- A straight P-box is a P-box with n inputs and n outputs
- n! possible mappings
- P Box is normally Keyless
- Mapping is predetermined
- If implemented in Hardware, then Prewired
- IF implemented in software, Permutation table shows rule of mapping

Straight P-Boxes

64 Inputs
64 Outputs
The index of the entry corresponds with the output
First entry is 58=>First Output comes from 58th Input
Last Entry is 07=>64th Output comes from 7th Input

58	50	42	34	26	18	10	02	60	52	44	36	28	20	12	04
62	54	46	38	30	22	14	06	64	56	48	40	32	24	16	08
57	49	41	33	25	17	09	01	59	51	43	35	27	19	11	03
61	53	45	37	29	21	13	05	63	55	47	39	31	23	15	07

All possibilities of a 3 × 3 straight P-box.



Example

Design an 8 × 8 permutation table for a straight P-box that moves the two middle bits (bits 4 and 5) in the input word to the two ends (bits 1 and 8) in the output words. Relative positions of other bits should not be changed.

Solution

We need a straight P-box with the table [4 1 2 3 6 7 8 5]. The relative positions of input bits 1, 2, 3, 6, 7, and 8 have not been changed, but the first output takes the fourth input and the eighth output takes the fifth input.

Compression P Box

- *A compression P-box is a P-box with n inputs and m outputs where m < n.*
- *Some of the inputs are blocked and do not reach the output.*

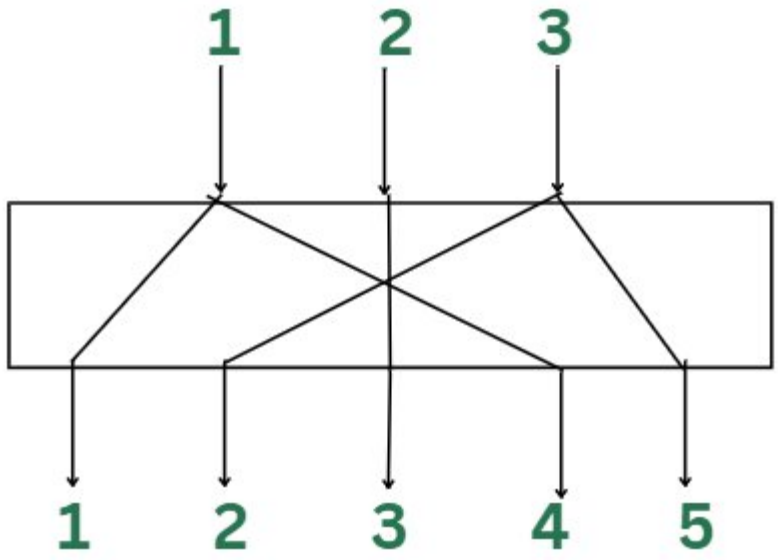
Example of a 32 × 24 permutation table

01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	06	10	11	12	30	31	32

- *Inputs 7,8,9,15,16,23,24 and 25 are blocked*
- *Used when we need to permute bits and the same time decrease the number of bits for next stage*

Expansion P-box

- An expansion P-box is a P-box with n inputs and m outputs where m > n.
- Some of the inputs are connected to more than one output



Invertability of P Box

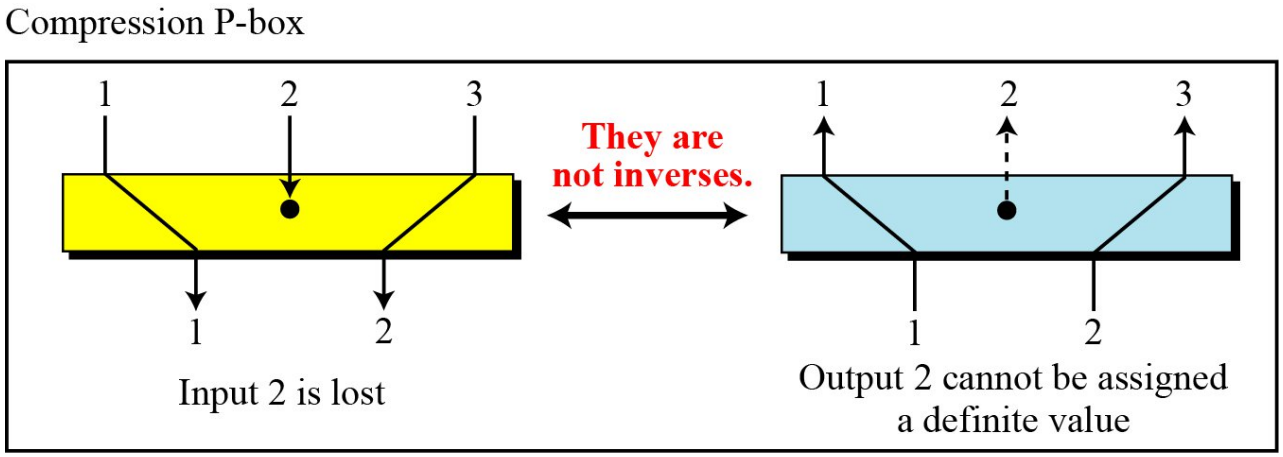
- A straight P-box is invertible,
- but compression and expansion P-boxes are not.

A straight P-box is invertible=>We can use Straight box in the encryption cipher and its inverse in the decryption cipher

Permutation tables need to be inverses of each other

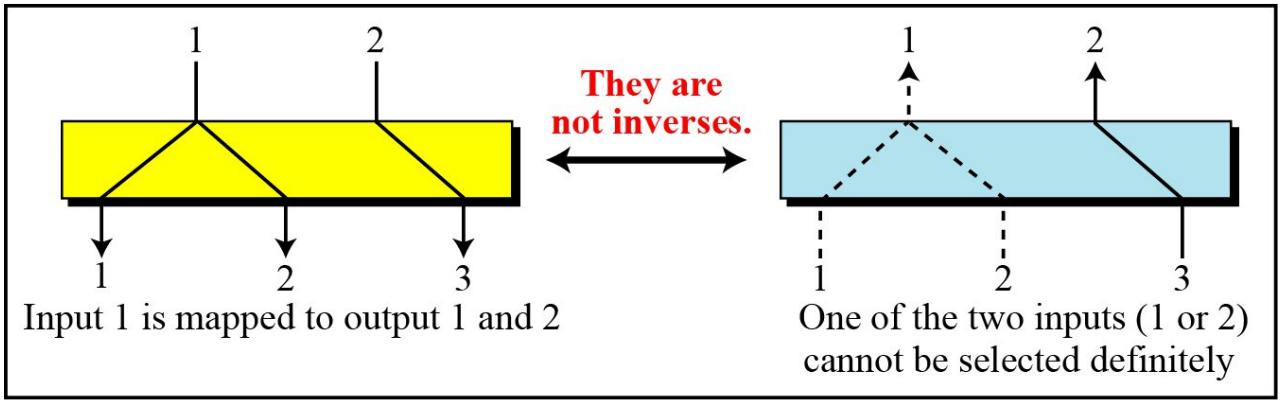
Invertability of P Box

In Compression P Box-
An input can be dropped During Encryption .
The Decryption Algorithm doesn't have a clue how to replace the dropped bit.



Invertability of P Box

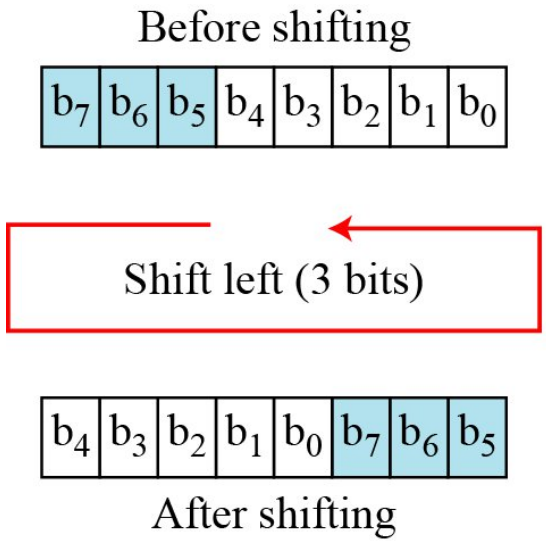
In Expansion P Box-
An input may be mapped to more than one output during Encryption. The Decryption algorithm does not have a clue which of the several inputs are mapped to an output.



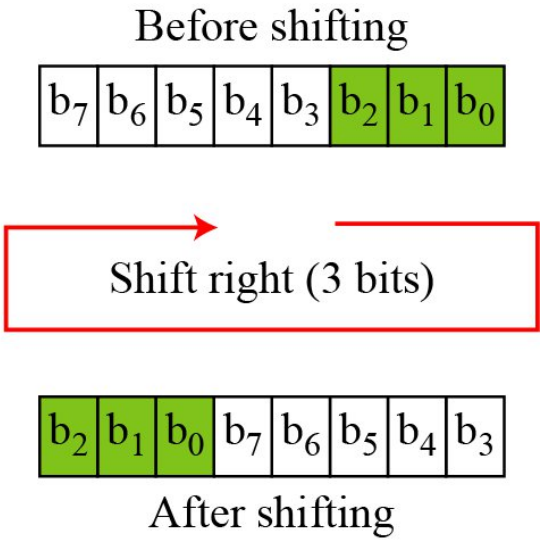
Expansion P-box

Circular Shift

- Mixes bits in a word and helps hide the patterns in the original word
- Circular left shift operation shifts each bit in a n bit word, k positions left.
- The leftmost k bits are removed from the left and become the rightmost bits



- Circular right shift operation shifts each bit in a n bit word ,k positions right.
- The rightmost k bits are removed from the right and become the leftmost bits
-



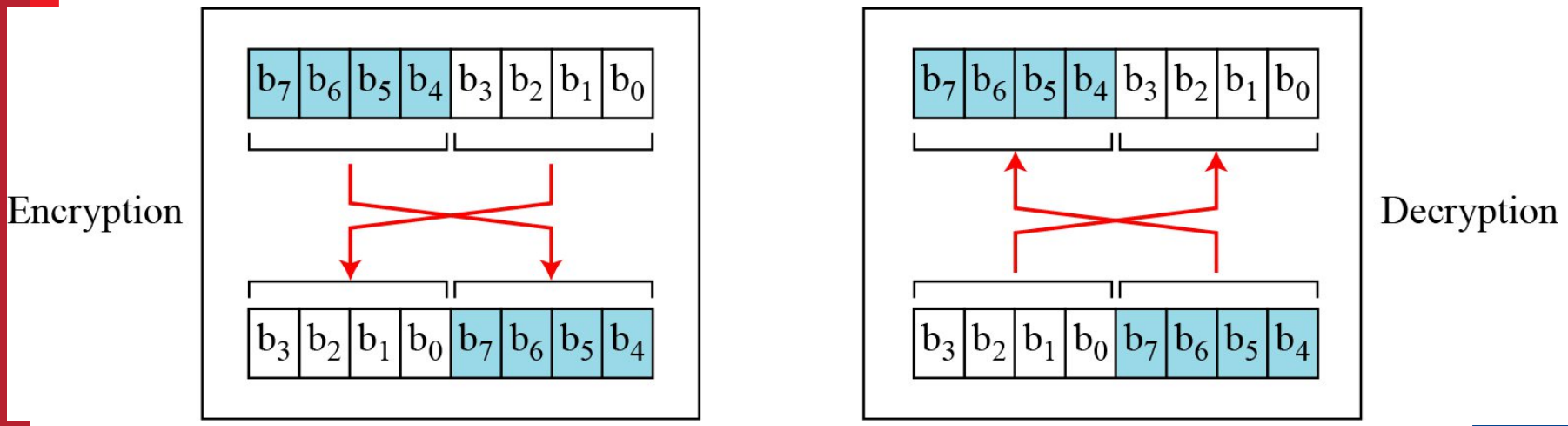
Invertibility of circular shift

- Invertibility- A circular left shift operation is the inverse of the circular right shift operation
- If one is used in encryption cipher, The other can be used in the Decryption cipher

Swap

The swap operation is a special case of the circular shift operation where k = n/2.

Swap operation on an 8-bit word



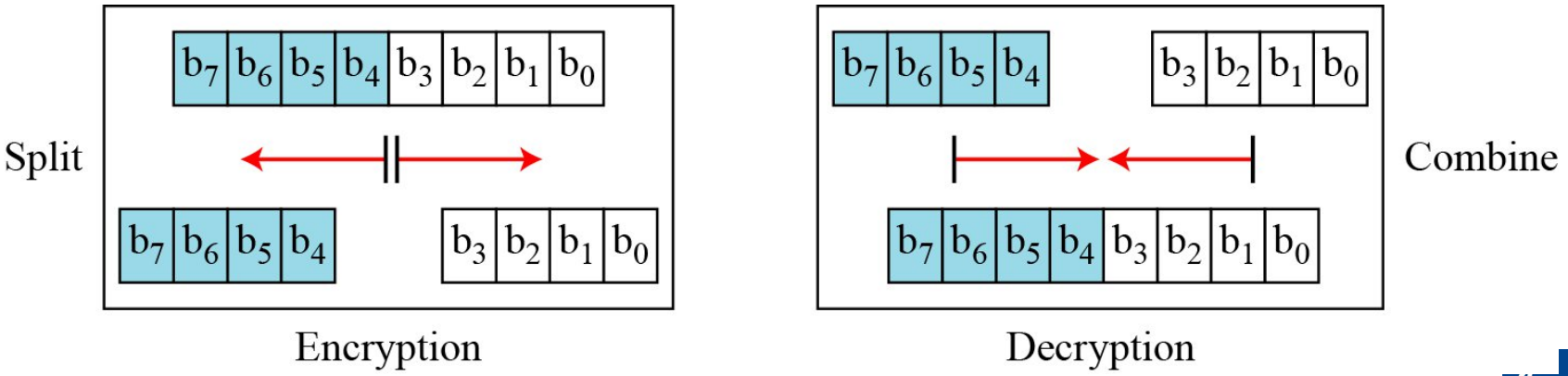
Swap operation invertability

- Self Invertible-A swap operation in Encryption cipher can be totally cancelled by a swap operation in the decryption cipher

Split and Combine

Two other operations found in some block ciphers are split and combine.

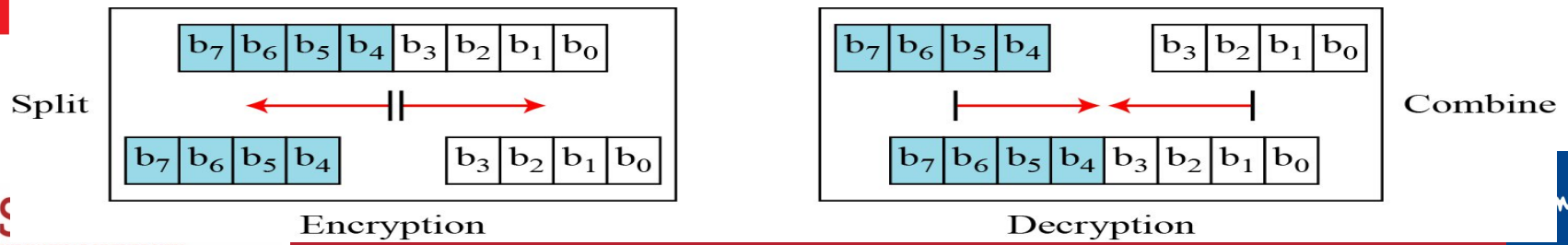
Split and combine operations on an 8-bit word



Split and Combine

- *Split operation splits an n-bit word in the middle creating two equal length words*
- *Combine operation normally concatenates two equal length words to create an n bit word.*
- *Inverse of each other*
- *Used as a pair to cancel each other out.*

Split and combine operations on an 8-bit word

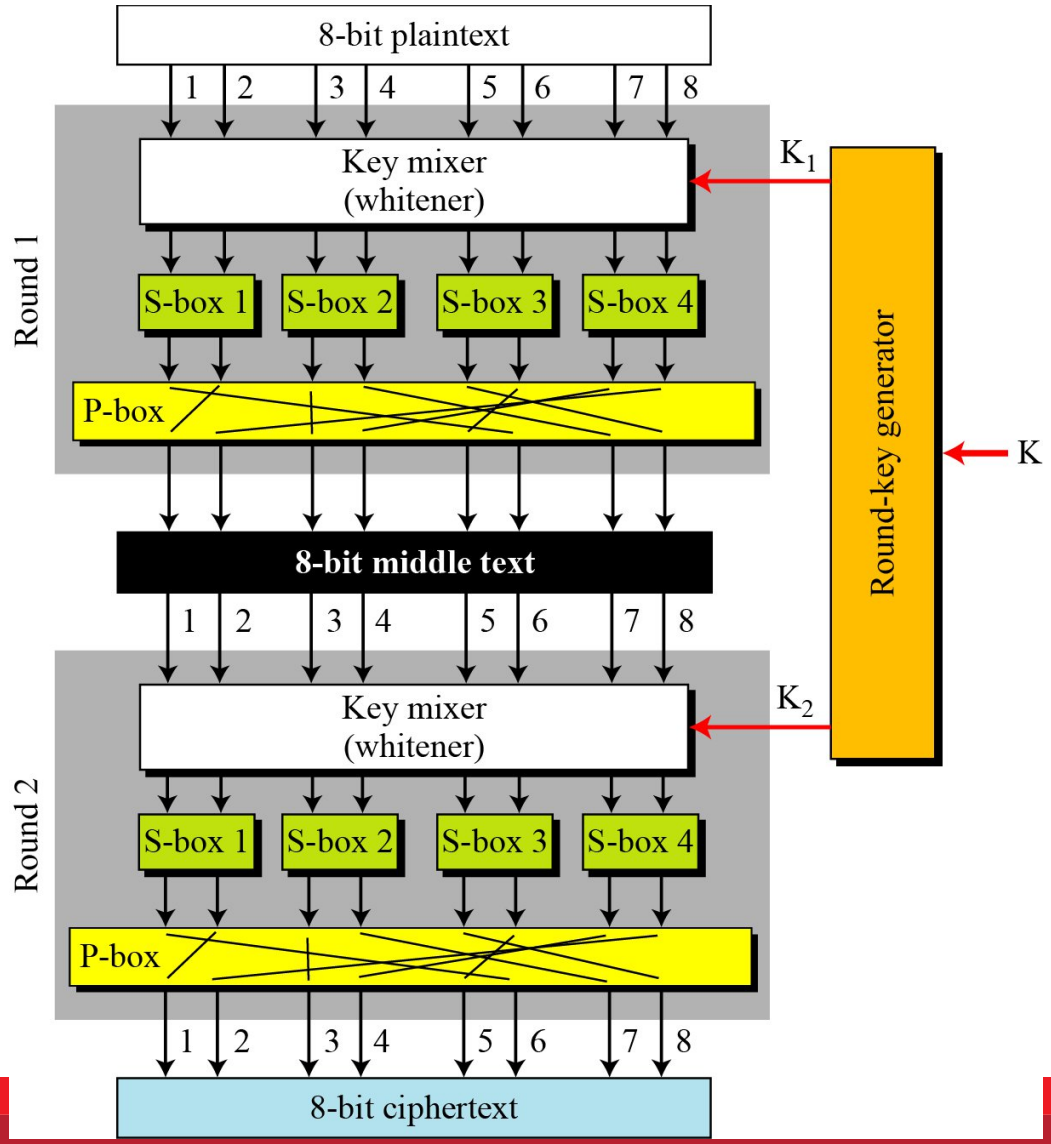


Rounds

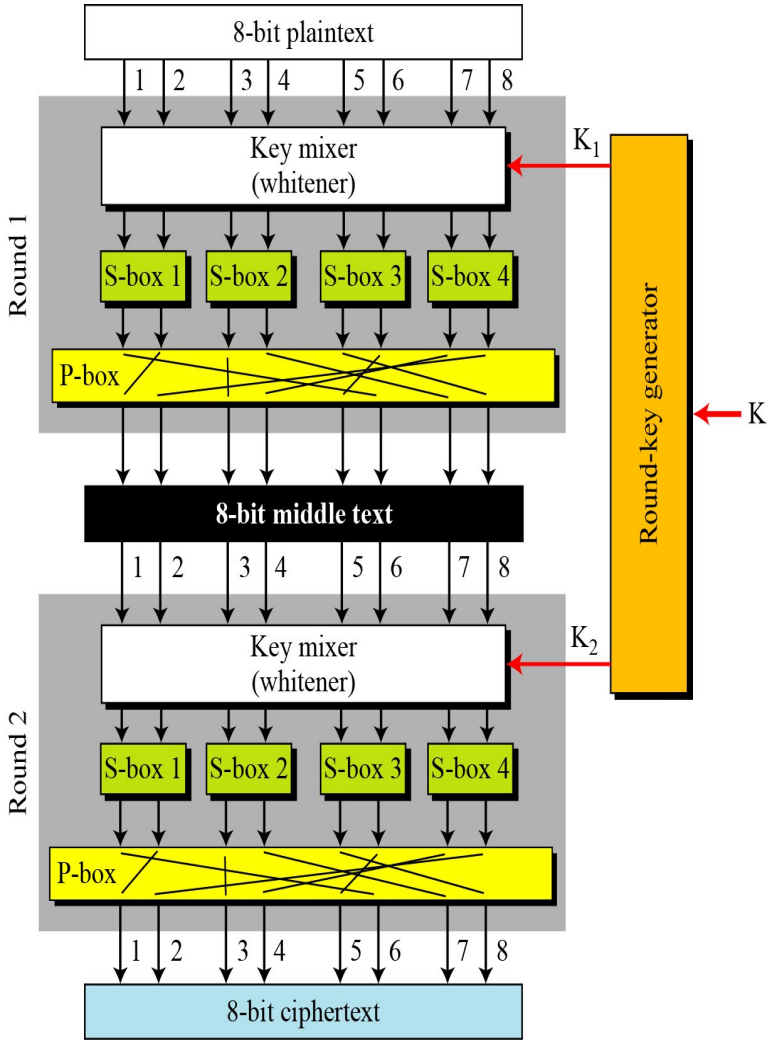
Diffusion and confusion can be achieved using iterated product ciphers where each iteration is a combination of S-boxes, P-boxes, and other components.

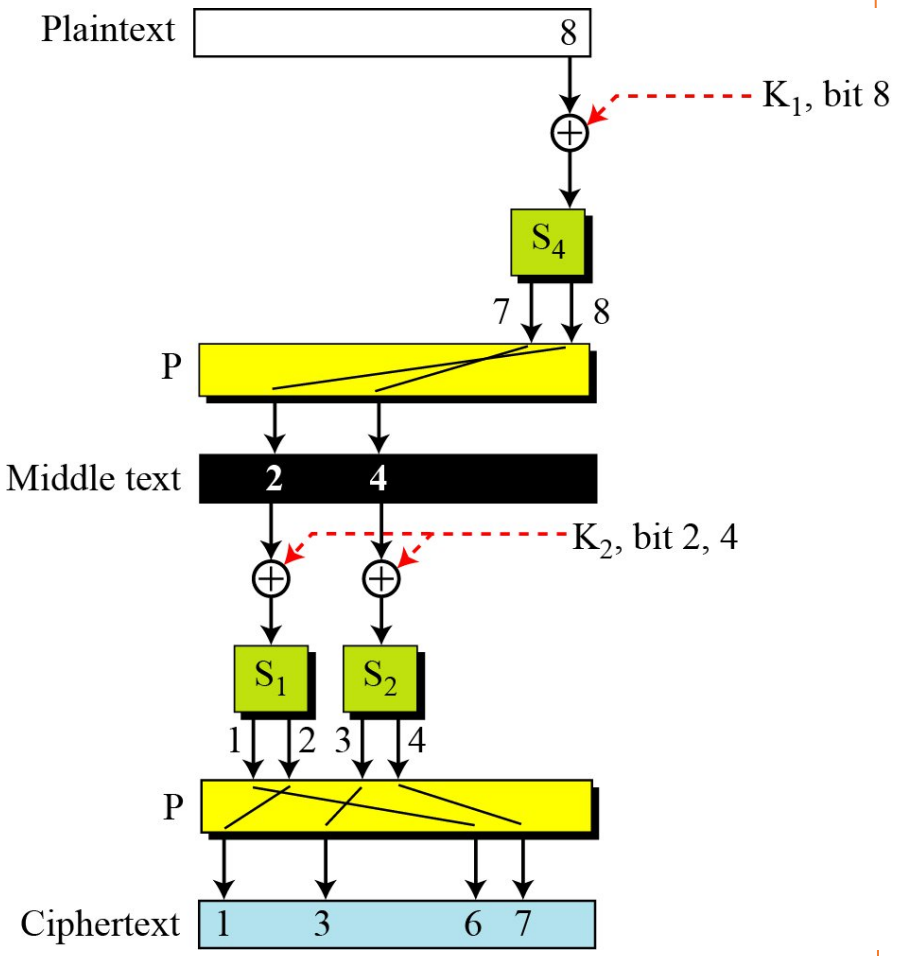
- *The block cipher uses a Key schedule or Key generator that creates different keys for each round.*
- *In a N-round cipher, the plaintext is encrypted N times to create the ciphertext.*
- *The ciphertext is decrypted N times to create the plaintext*

A product cipher with two rounds



How Changing a single bit in the plain text affects many bits in the ciphertext





Classes of Product Ciphers

Modern block ciphers are all product ciphers, but they are divided into two classes.

1. *Feistel ciphers*
   - Uses both invertible and non-invertible components
   - A Feistel cipher can have three types of components: self-invertible, invertible, and noninvertible.

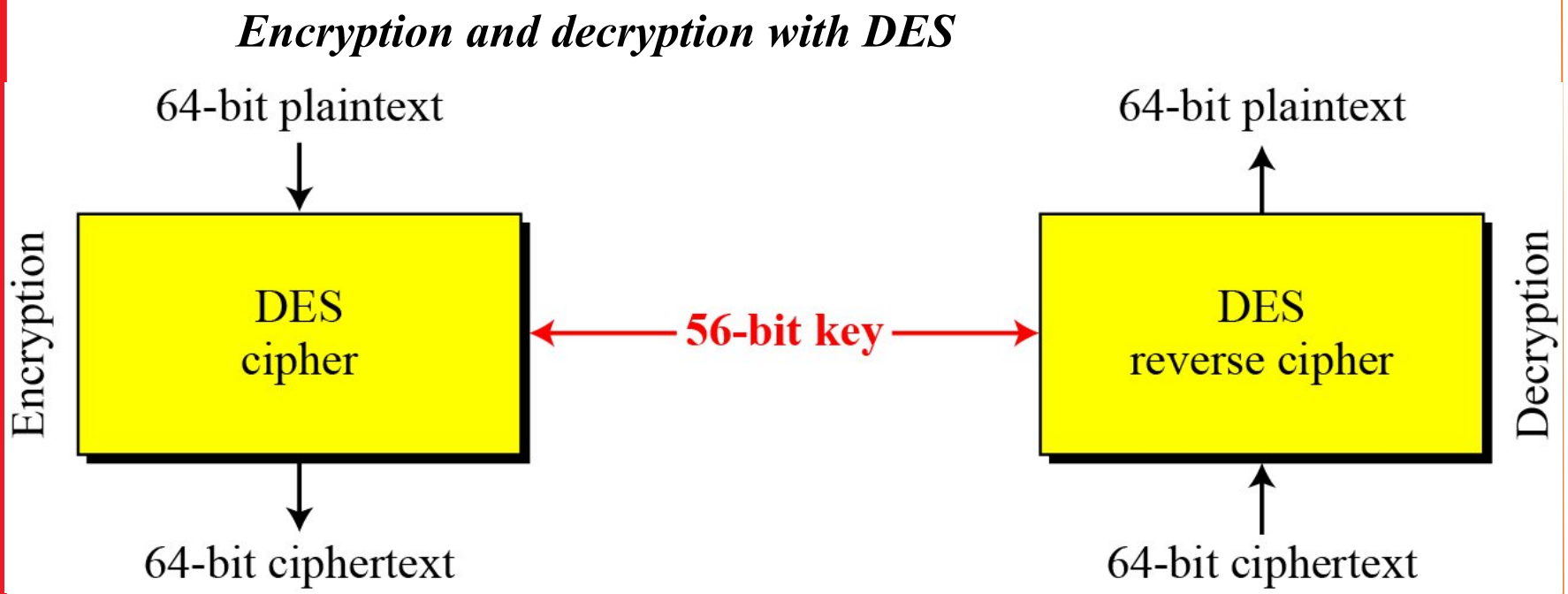
2. *Non-Feistel ciphers*
   - Non-Feistel ciphers-Uses only invertible components
   - A non-Feistel cipher uses only invertible components.
   - A component in the encryption cipher has the corresponding component in the decryption cipher.

Data Encryption Standard

History of DES

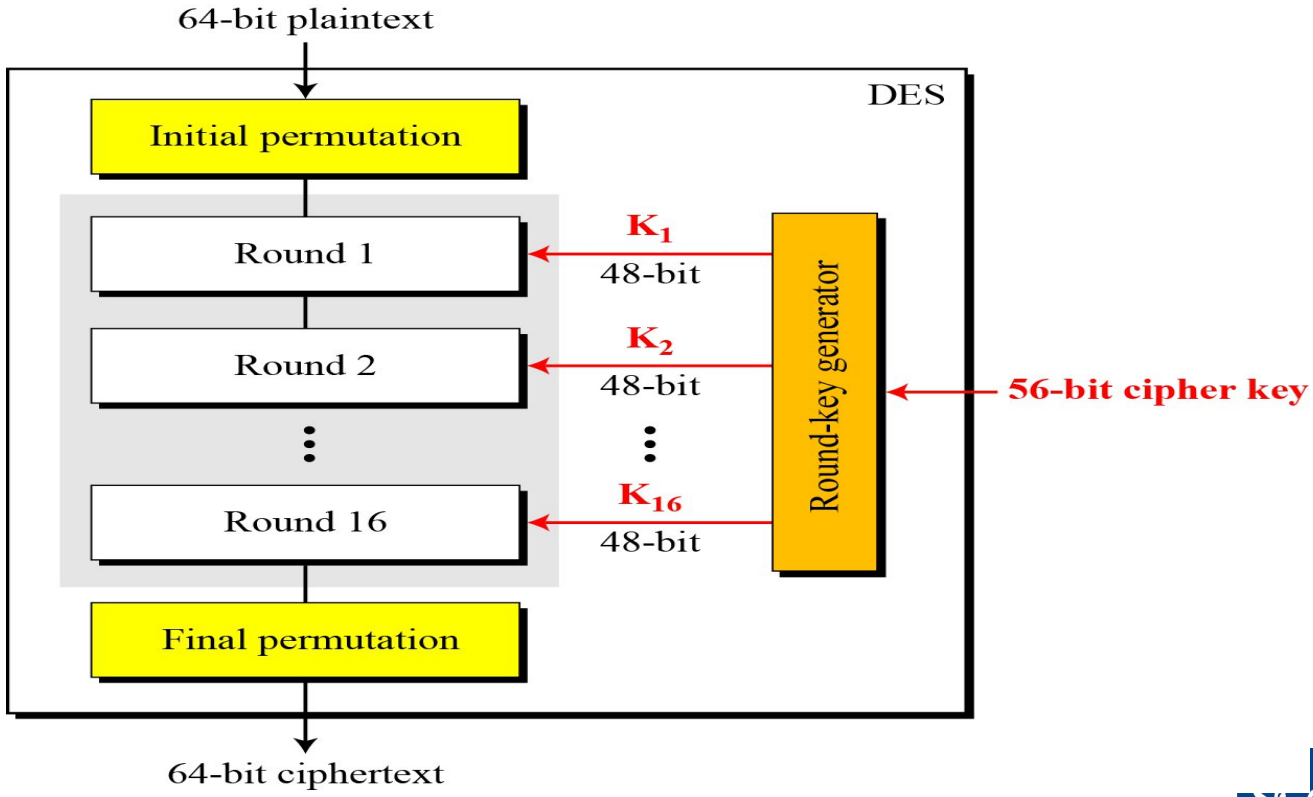
- In 1973, NIST published a request for proposals for a national symmetric-key cryptosystem.
- A proposal from IBM, a modification of a project called Lucifer, was accepted as DES.
- DES was published in the Federal Register in March 1975 as a draft of the Federal Information Processing Standard (FIPS).
- Finally published as FIPS 46 in the Federal Register in January 1977
- A new standard FIPS 46-3 recommended the use of Triple DES- repeated DES cipher three times
- AES, The recent standard is supposed to replace DES in the long run

DES - a block cipher



DES Structure

- The encryption process is made of two permutations (P-boxes), which we call initial and final permutations, and sixteen Feistel rounds.



DES Keys and General structure

- Each round uses a different 48 bit round key generated from the cipher key according to a predefined algorithm

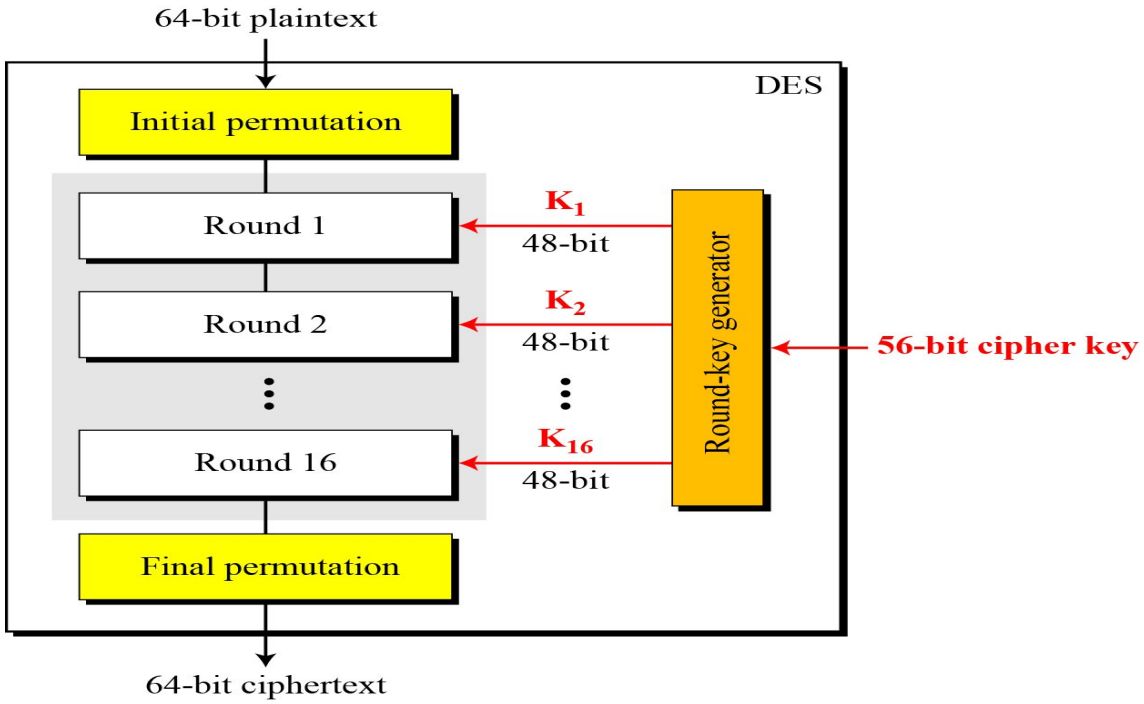
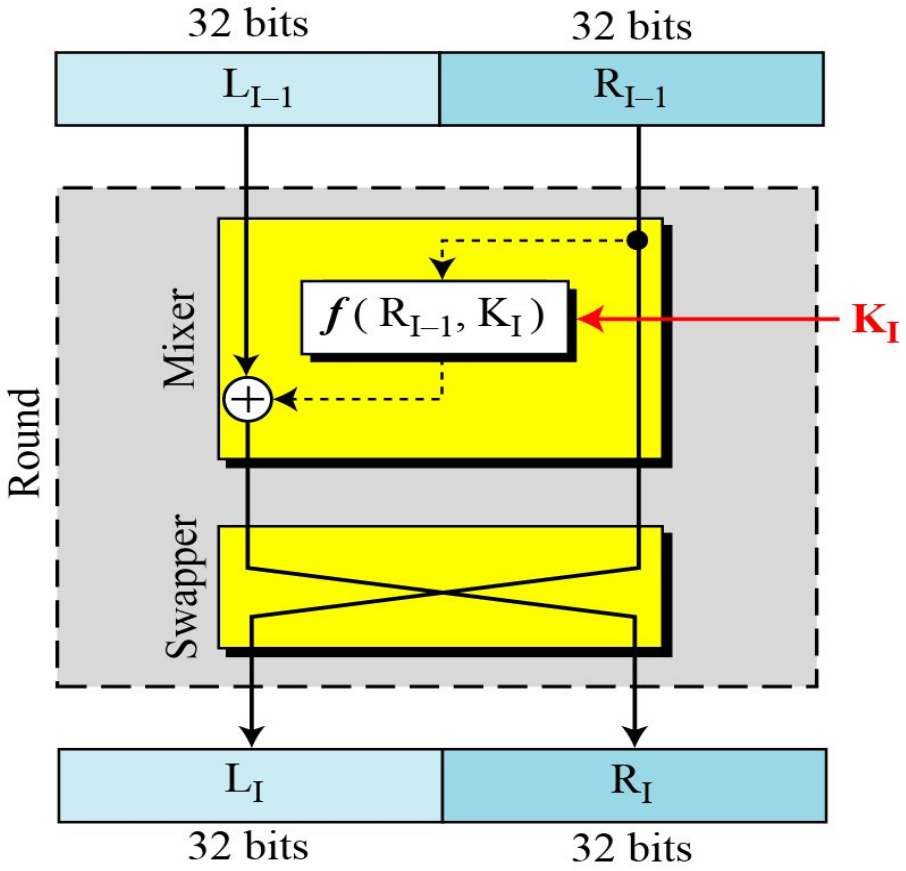


Fig: General structure of DES

DES Rounds

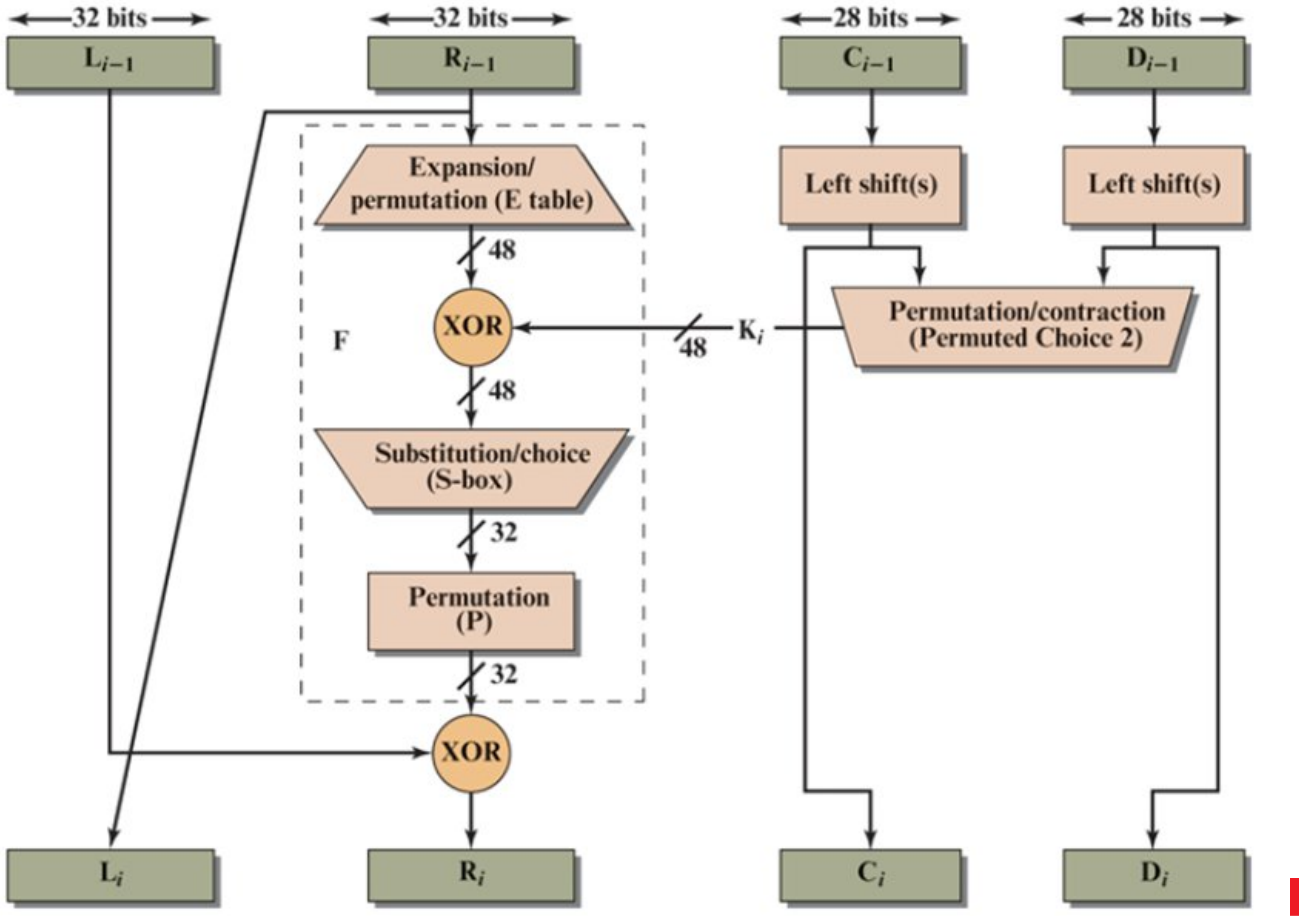


- *Swapper is invertible*
- *Mixer is invertible because of EXOR operation*
- *All non invertible elements are collected inside the function f*

A round in DES

The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.

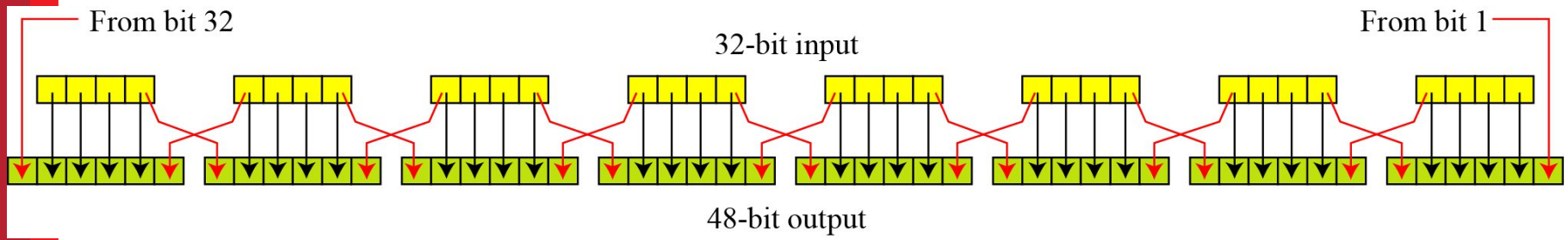
Figure C.2 **Single Round of DES Algorithm**



Expansion P-box

Since R_{I-1} is a 32-bit input and K_I is a 48-bit key, we first need to expand R_{I-1} to 48 bits.

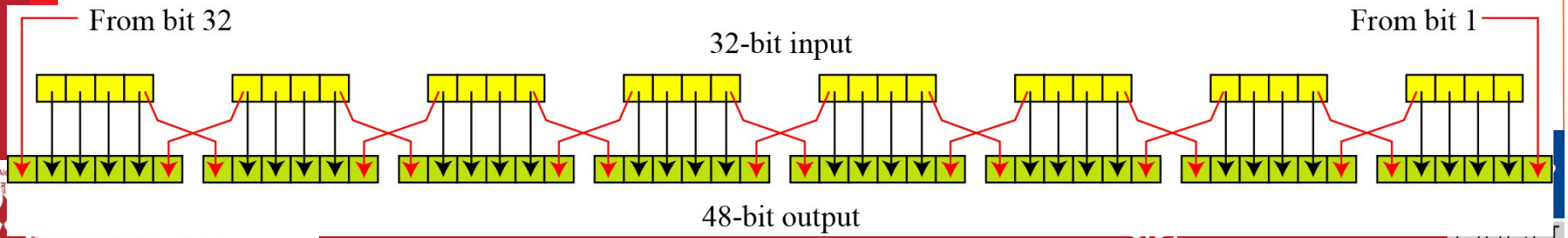
Figure 6.6 ***Expansion permutation***



Expansion P-box

- *For each section, I/P bits 1,2,3 and 4 are copied to O/P bits 2,3,4 and 5 respectively.*
- *Output Bit 1 comes from bit 4 of previous section*
- *Output bit 6 comes from bit 4 of next section*
- *Sections 1 and 8 are considered adjacent, Same rule applies to bits 1 and 32*

Expansion permutation



Expansion P-box expressed as table

Although the relationship between the input and output can be defined mathematically, DES uses Table to define this P-box.

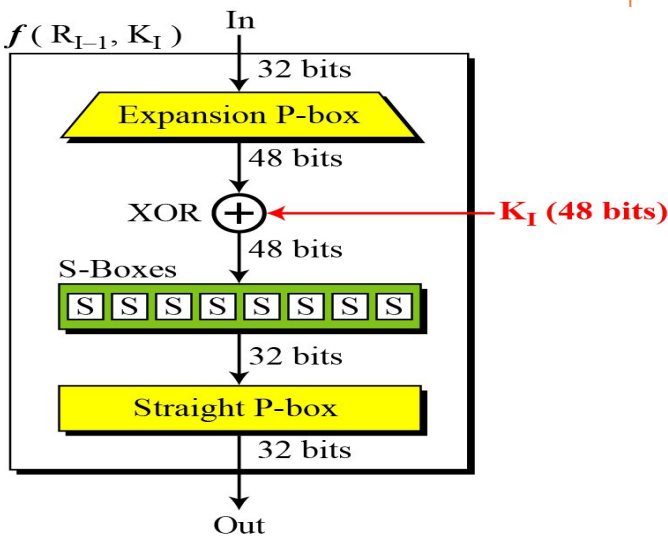
A 6 X 8 Table

Expansion P-box table

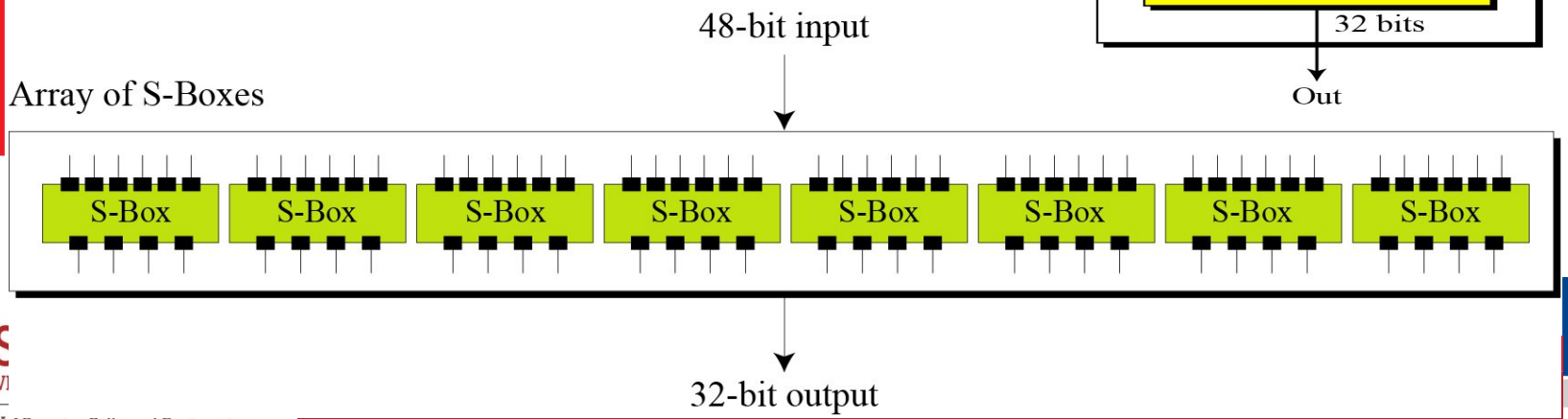
32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	31	31	32	01

S-Boxes

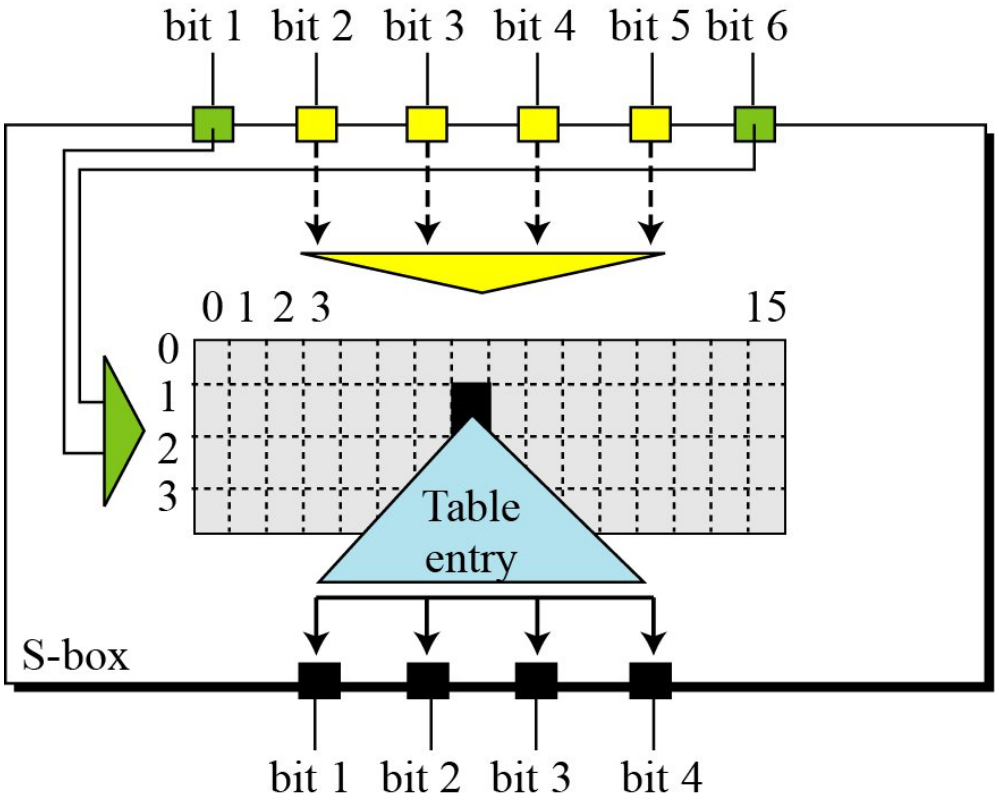
- *The S-boxes do the real mixing (confusion).*
- *48 bit output from Whitener is divided into Eight 6-bit chunks*
- *DES uses 8 S-boxes,*
- *Each 6-bit chunk as input and a 4-bit output.*



S-boxes



S-box rule: Bit 1 and 6 make the row, bits 2 to 5 make the column



Example

- *Table 6.3 shows the permutation for S-box 1.*
- *For the rest of the boxes see the textbook.*
- *The row number and column no, output are given as decimal to save space*
- *These need to be changed to binary*

Table 6.3 *S-box 1*

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

Example

The input to S-box 1 is 100011. What is the output?

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

Solution

- If we write the first and the sixth bits together, we get 11 in binary, which is 3 in decimal.
- The remaining bits are 0001 in binary, which is 1 in decimal.
- We look for the value in row 3, column 1, in Table (S-box 1).
- The result is 12 in decimal, which in binary is 1100. So the input 100011 yields the output 1100.

Straight Permutation

- ***Last operation in DES round***
- ***Permutation with 32 bit input and 32 bit output***

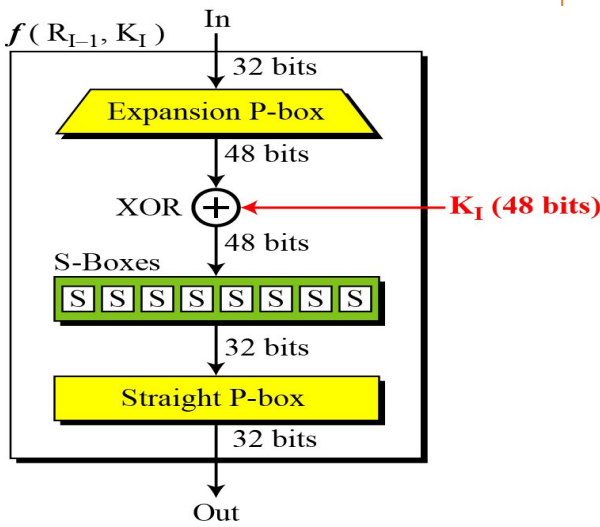


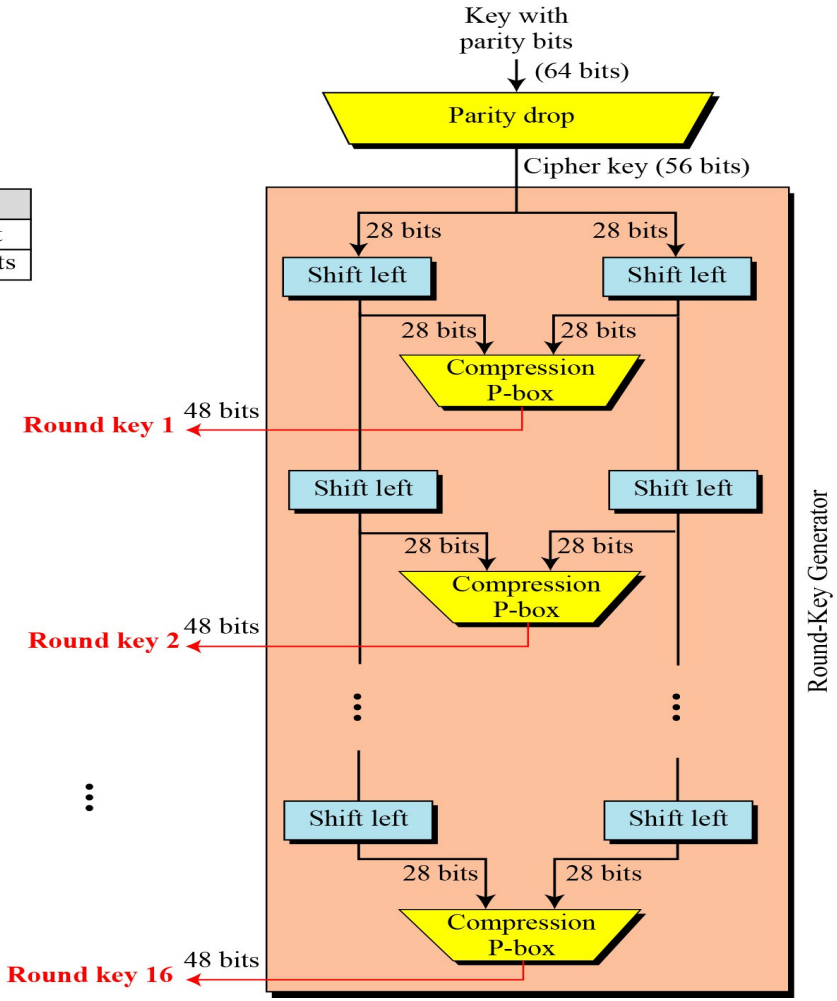
Table 6.11 ***Straight permutation table***

16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

Key Generation

Shifting

Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits



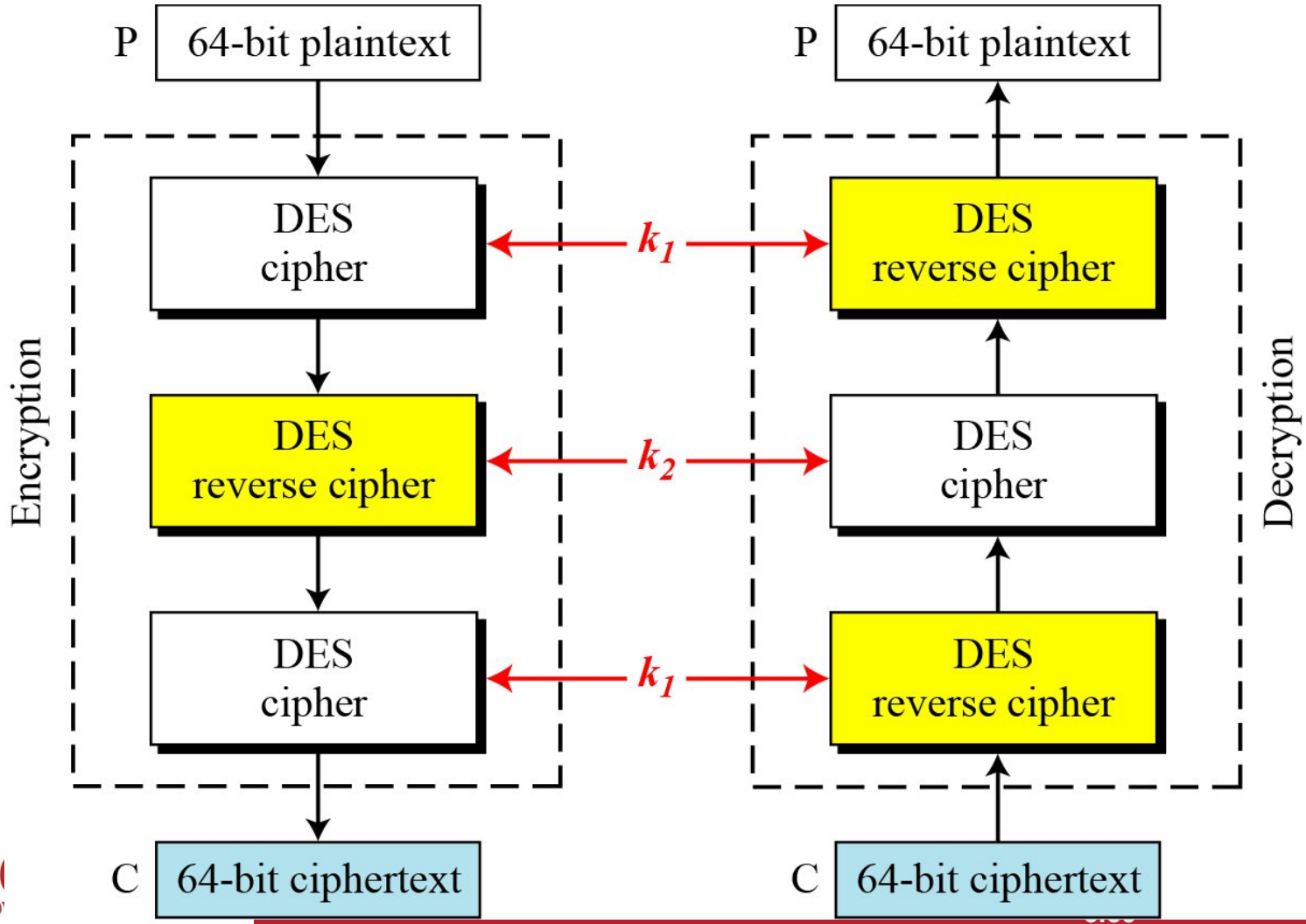
The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key.

Cipher key is normally given as a 64bit key in which 8 extra bits are parity bits, which are dropped off before the actual key generation

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bit shifts	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Triple DES

Triple DES with two keys



DES Analysis

- Properties
  - Avalanche effect
  - completeness
- Design Criteria
- DES Weaknesses

DES Properties

Avalanche effect

- *A small change in the Plain text (or key) should create a significant change in the ciphertext*
- *DES has been proved to be strong with respect to this property*

Completeness effect

- *Completeness effect means that each bit of the ciphertext needs to depend on many bits on the plaintext.*
- *Diffusion and Confusion produced by P Boxes and S Boxes in DES show a very strong completeness effect*

Avalanche effect

To check the avalanche effect in DES, let us encrypt two plaintext blocks (with the same key) that differ only in one bit and observe the differences in the number of bits in each round.

Plaintext: 0000000000000000 Key: 22234512987ABB23
Ciphertext: 4789FD476E82A5F1

Plaintext: 0000000000000001 Key: 22234512987ABB23
Ciphertext: 0A4ED5C15A63FEA3

- Experiments have found that DES versions with less than 16 rounds are even more vulnerable to known plain text attacks than brute force attack , That justifies the use of 16 rounds
- Weaknesses in Cipher Design
  - Weaknesses in S-boxes
  - Weaknesses in P-boxes
  - Weaknesses in Key

Applications of DES

- **Data Security in Banking**
  - Secure fund transfers (e.g., ATM transactions, SWIFT network)
  - Encrypting PINs and cardholder information
- **Secure Communication**
  - Encrypting emails and messages
  - Securing VoIP and telecommunication networks
- **File and Disk Encryption**
  - Encrypting sensitive files and documents
  - Used in early disk encryption tools

Applications of DES

- **Identity Verification & Authentication**
  - Used in challenge-response authentication systems
  - Biometric data encryption
- **Smart Cards and Embedded Systems**
  - Secure authentication in access control systems
  - Encrypting stored data on smart cards
- **Wireless and Network Security**
  - Used in older VPNs and network security protocols
  - Secure key exchanges
- **Digital Rights Management (DRM)**
  - Protecting copyrighted digital media
  - Encryption in early DVD protection systems

Limitations of DES

1. Small Key Size (56-bit)

- **Vulnerability:** The 56-bit key is too short by today's standards, making it susceptible to **brute-force attacks**.
- **Example:** In 1998, the **EFF (Electronic Frontier Foundation)** built a machine that cracked DES in under 3 days. Modern systems can do it even faster.

2. Susceptible to Brute-Force Attacks

- DES relies solely on the strength of its key. With advancements in computing power, brute-forcing DES is now feasible for even moderately resourced attackers.

3. Slow Performance in Software

- DES was optimized for hardware, not software.
- In software implementations, especially on general-purpose processors, it performs inefficiently compared to algorithms like AES.

4. Vulnerable to Cryptanalytic Attacks

- **Differential and Linear Cryptanalysis:** Techniques discovered in the 1990s can analyze DES more effectively than brute force, reducing its effective security.
- **Meet-in-the-Middle Attacks:** Particularly relevant for DES variants like Double DES.

Limitations of DES

5. Insecure in Certain Modes of Operation

- When used in modes like **ECB (Electronic Codebook)**, patterns in the plaintext can still be visible in the ciphertext, compromising data confidentiality.

6. Lack of Flexibility

- Fixed block size of 64 bits, which is small for modern data encryption needs.
- Key schedule is relatively weak, making the key generation process predictable in some cases.

7. Legacy Dependency

- Despite its weaknesses, some systems still use DES due to legacy constraints, posing security risks if not properly updated or managed.

Advanced Encryption Standard

- Introduction to AES
- History and Development
- Key Features of AES
- How AES Works (Encryption Process)
- AES Structure (Rounds and Key Sizes)
- Modes of Operation
- Applications of AES
- Advantages of AES
- Limitations of AES
- AES vs DES Comparison
- Conclusion

Introduction to AES

- What is AES?
  - AES (Advanced Encryption Standard) is a symmetric-key encryption algorithm.
  - Used to secure data through encryption and decryption.
- Developed by: Vincent Rijmen and Joan Daemen.
- Origin: Developed as the successor to DES.
- Adopted by: U.S. National Institute of Standards and Technology (NIST) in 2001.
- Also known as: Rijndael algorithm (its original name).

Introduction to AES

- In February 2001, NIST announced that a draft of the Federal Information Processing Standard (FIPS) was available for public review and comment.
- Finally, AES was published as FIPS 197 in the Federal Register in December 2001.

Introduction to AES

- AES is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits.
- It uses 10, 12, or 14 rounds.
- The key size, which can be 128, 192, or 256 bits, depends on the number of rounds

Key Features of AES

- **Block Cipher** – Operates on 128-bit data blocks.
- **Key Sizes** – Supports **128, 192, and 256-bit** keys.
- **Fast & Efficient** – Works well in both hardware and software.
- **Strong Security** – Resistant to cryptanalysis attacks.
- **Widely Used** – Found in VPNs, banking, military, and more.

AES Structure (Rounds and Key Sizes)

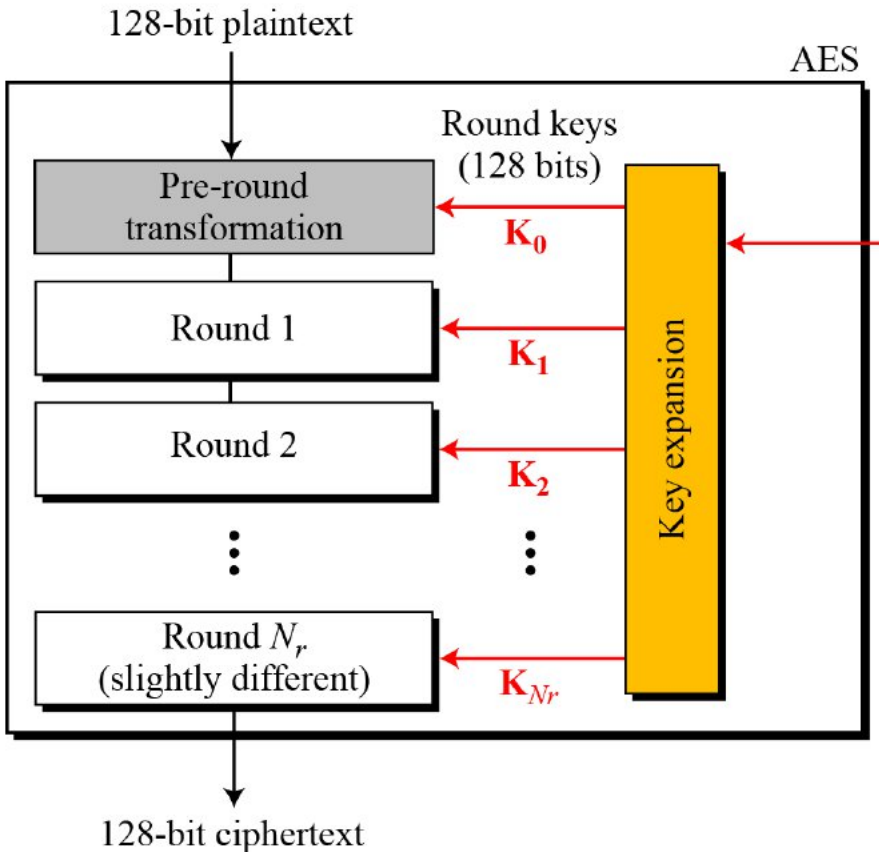
- AES-128: 10 rounds
- AES-192: 12 rounds
- AES-256: 14 rounds
- Each round increases security and complexity.

- Round Keys created by the Key expansion algorithm are always 128 bits , the same size as the plaintext or ciphertext
- The number of round keys generated by the key expansion algorithm is always one more than the number of rounds
- Number of Round Keys=N_r+1
- Round Keys are $K_0, K_1, K_2, \ldots\ldots\ldots\ldots\ldots K_{Nr}$

How AES Works (Encryption Process)

- Key Expansion: Generates round keys from the cipher key.
- Initial Round: AddRoundKey (XOR with the key).
- Main Rounds (9/11/13 Rounds):
  - SubBytes (byte substitution using S-box).
  - ShiftRows (row-wise shifting).
  - MixColumns (column mixing for diffusion).
  - AddRoundKey.
- Final Round: Same as main rounds but without MixColumns.

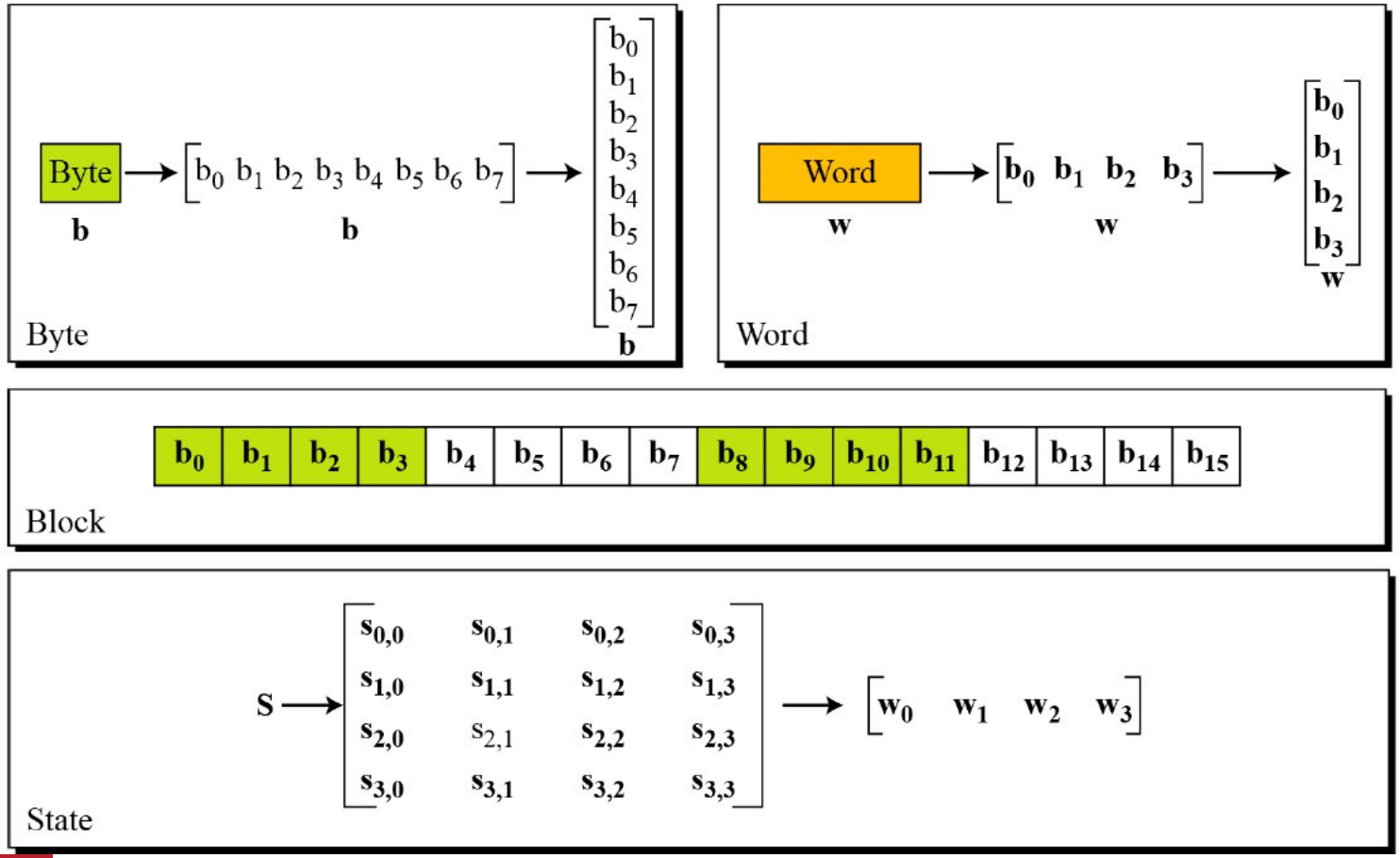
AES: General design structure



Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

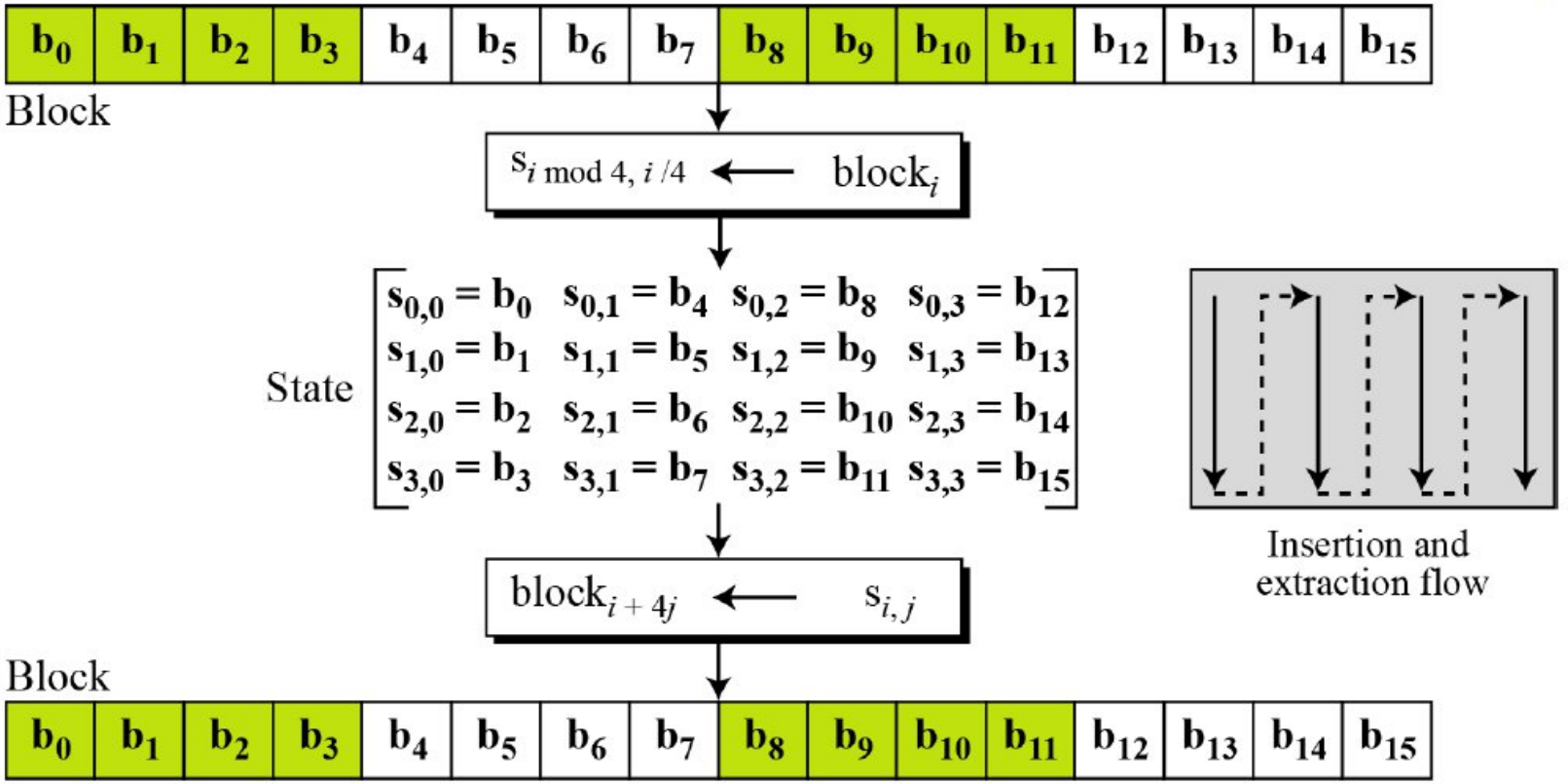
Data sizes in AES



- Byte- 8 bits
- Word- 32 bits
- Block – 128 bits
- State- 16 bytes

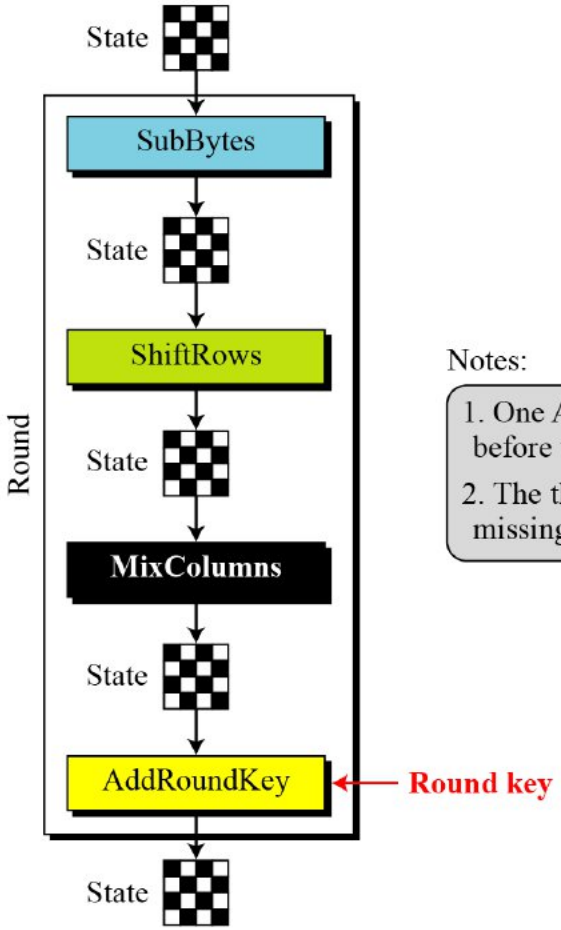
Data transformation - bit, block, state

At the end of cipher, the bytes in data block are extracted in the same way



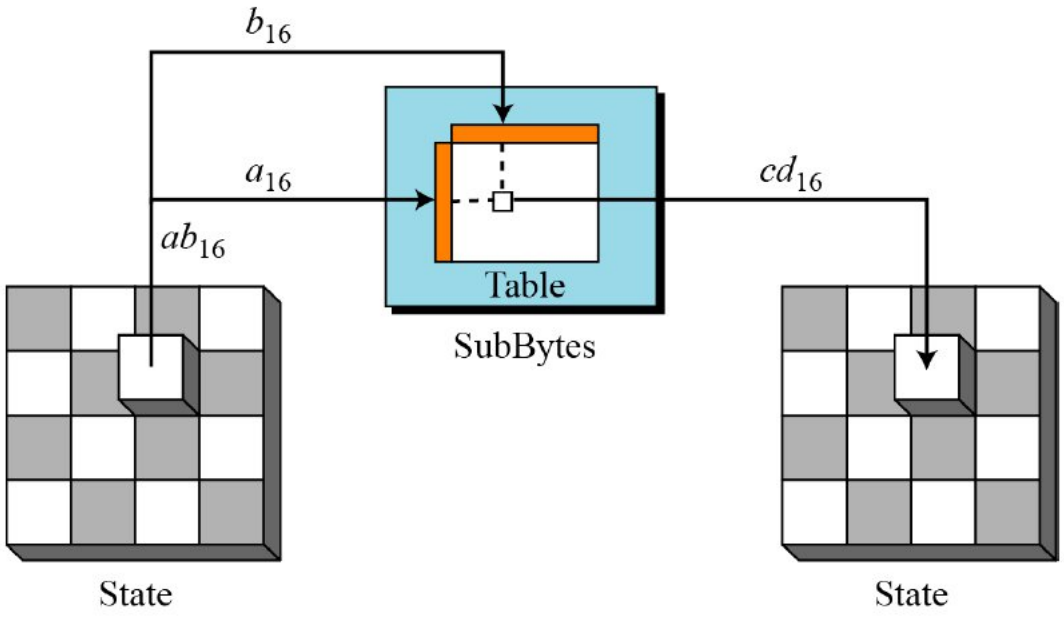
Structure of AES round

- All transformations are invertible
- *The pre-round section uses only one transformation(AddRoundKey)*
- *•In the last round, MixColumns transformation is missing*



SubBytes- involves 16 independent byte-to-byte transformations

- *State is treated as 4X4 matrix of bytes*
- *Transformation is done one byte at a time*
- *The content of each byte is changed but the arrangement of the bytes in the matrix remains the same*
- *Each byte is transformed independently*



SubBytes

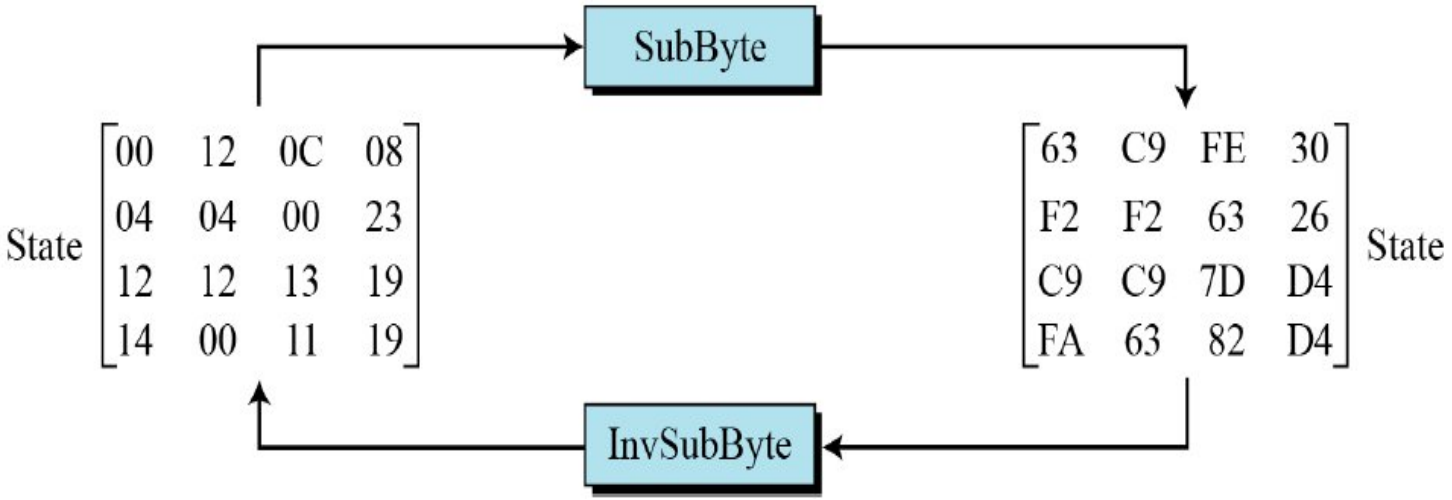
- The Substitution Table is used (S Box) for SubBytes transformation
- Provides confusion effect
- Two bytes $5A_{16}$ and $5B_{16}$ which differ only in one bit are transformed to BE_{16} and 39_{16} which differ in four bits

Table 7.1 *SubBytes transformation table*

	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

Table 7.1 *SubBytes transformation table (continued)*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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	CB	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



SubBytes and InverseSubBytes

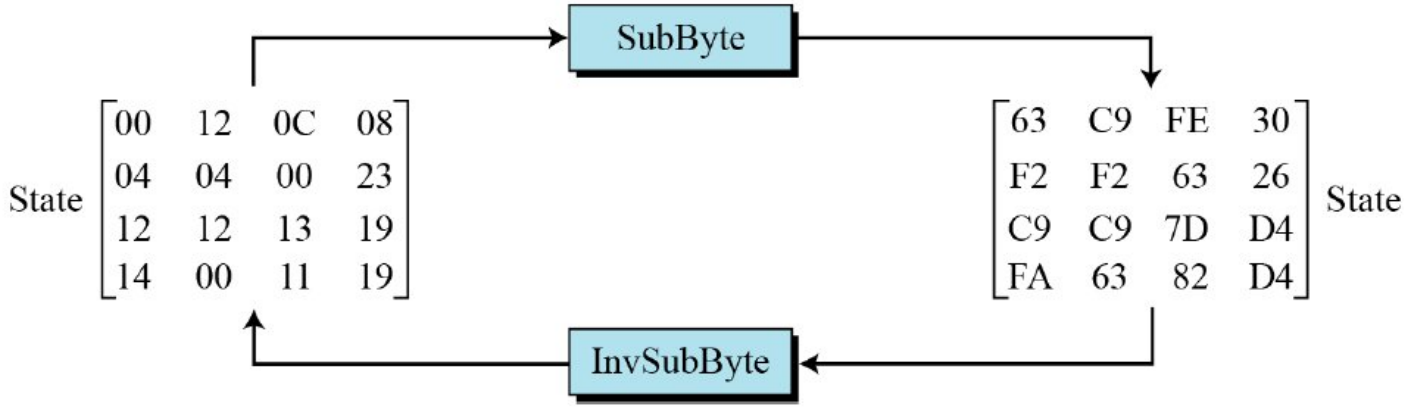
Example 7.2

Table 7.2 *InvSubBytes transformation table*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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

Table 7.2 *InvSubBytes transformation table*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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



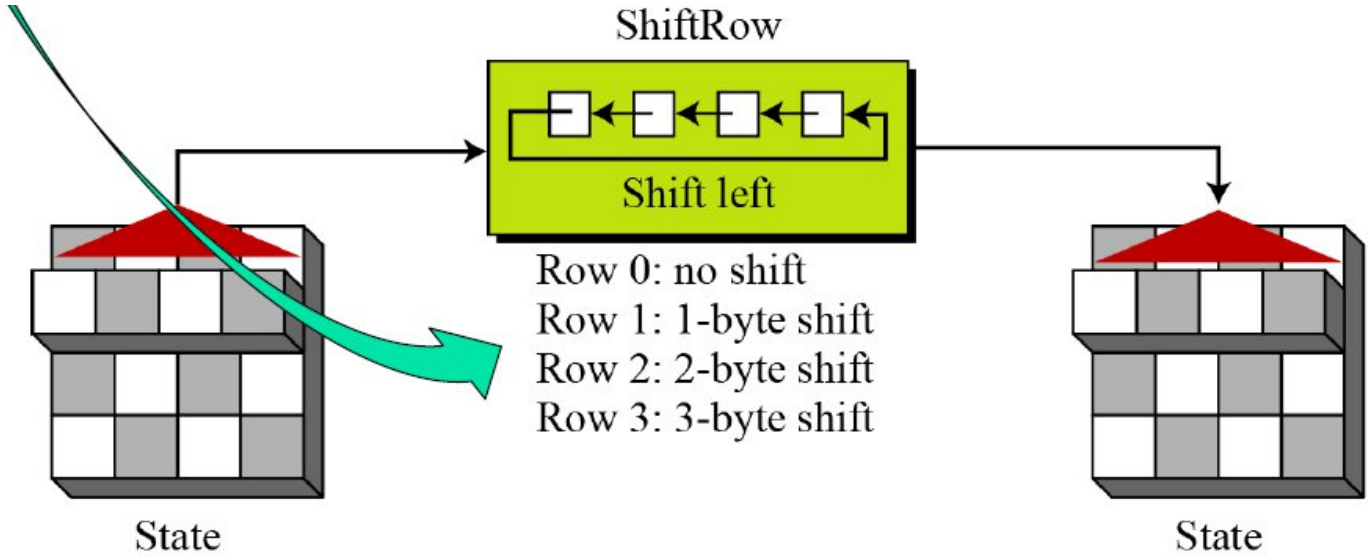
ShiftRows

- *Another transformation found in a round is shifting, which permutes the bytes.*
- *In DES, Permutation is done at the bit level, Shifting transformation in AES is done at the byte level*
- *The order of the bits in the byte is not changed*

ShiftRows

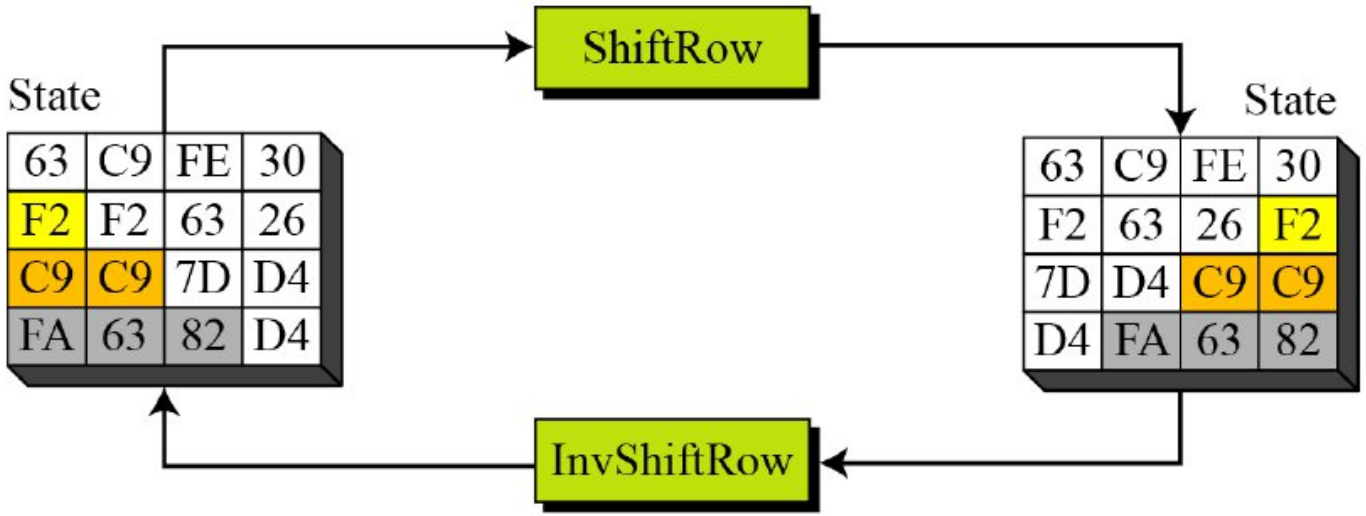
- In the encryption, the transformation is called ShiftRows.
- Shifting is to the left
- The no of shifts depends on the row number(0,1,2,3) of the state matrix
- Shift row transformation operates one row at a time

ShiftRows


ShiftRow
Shift left
Row 0: no shift
Row 1: 1-byte shift
Row 2: 2-byte shift
Row 3: 3-byte shift
State
State

InvShiftRows

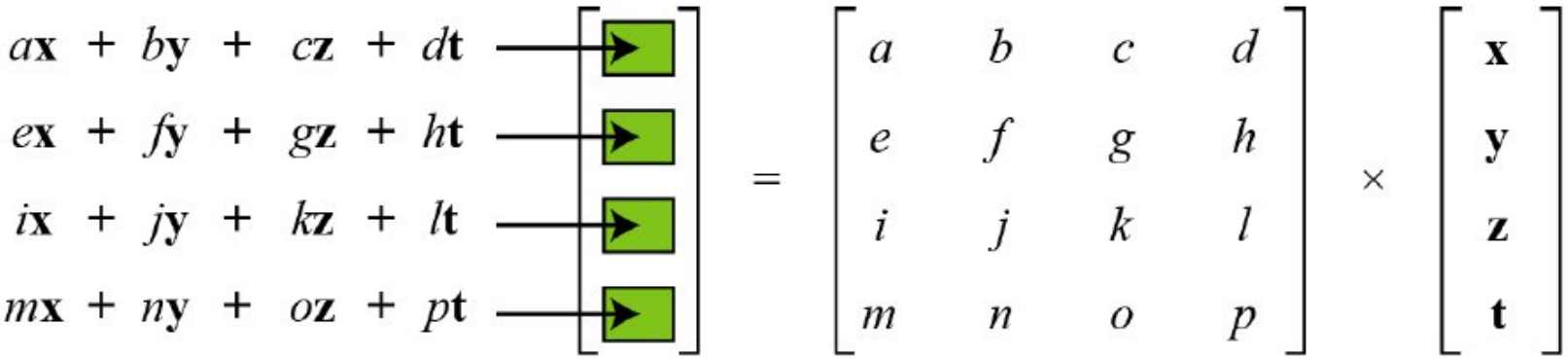
- the shifting is to the right.
- The number of shifts is the same as the row number (0,1,2 and 3) of the state matrix
- ShiftRows and InvShiftRows transformations are inverses of each other



Mixing

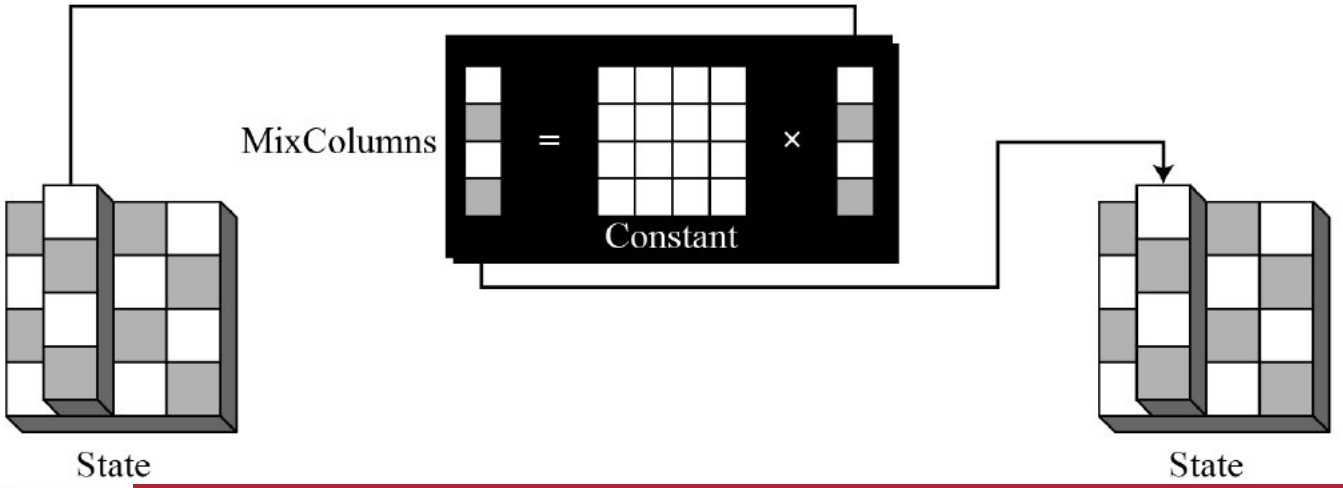
- interbyte transformation that changes the bits inside a byte, based on the bits inside the neighboring bytes.
- Takes 4 bytes at a time, combining them to recreate four new bytes
- Each new byte is different, even if all 4 bytes are the same
- Multiplies each byte with a different constant and mixes them

Mixing


ax + by + cz + dt
ex + fy + gz + ht
ix + jy + kz + lt
mx + ny + oz + pt
=
a b c d
e f g h
i j k l
m n o p
×
x
y
z
t

MixColumns & Inv MixColumns

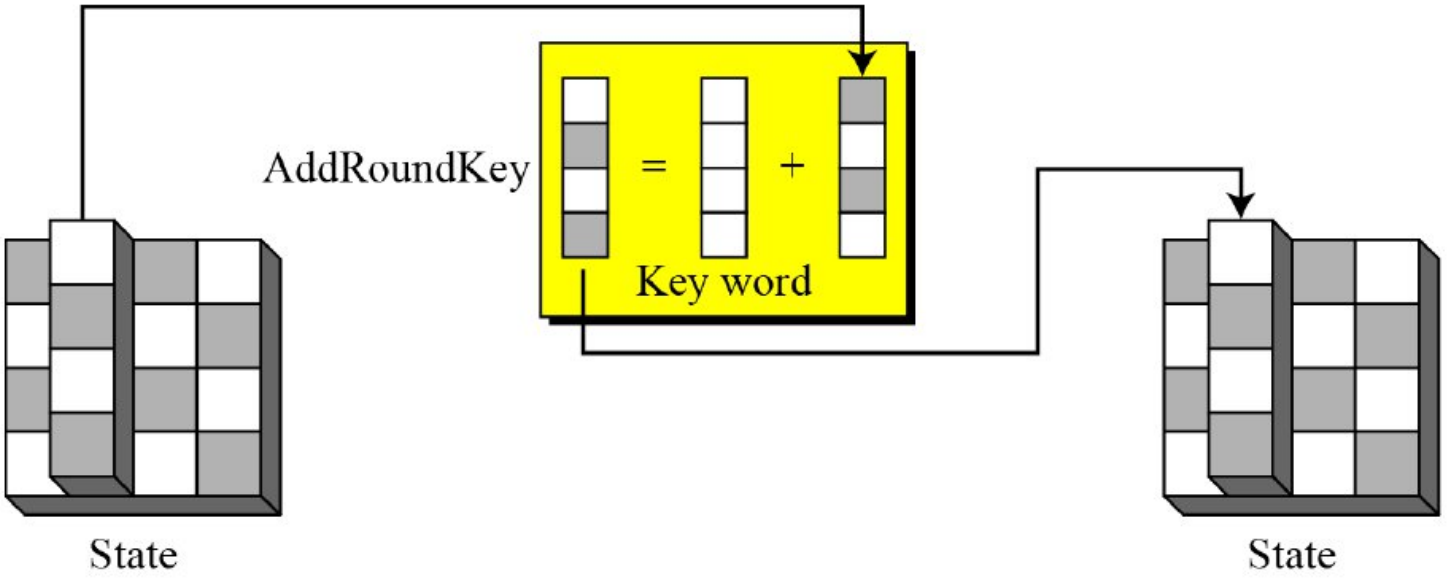
- The MixColumns transformation operates at the column level; it transforms each column of the state to a new column.
- Matrix multiplication of a state column by a constant square matrix
- *The InvMixColumns transformation is basically the same as the MixColumns transformation*



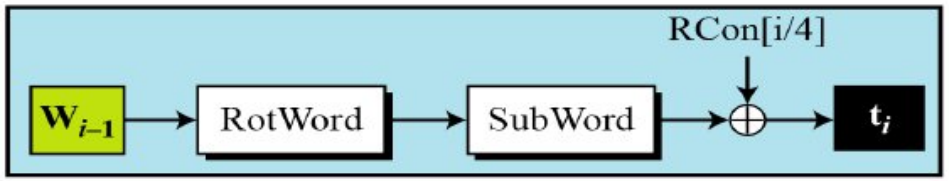
AddRoundKey

- Each Round key is 128 bits long
- Treated as Four 32 bit words
- For adding the key to the state , each word is considered as a column matrix
- AddRoundKey proceeds one column at a time.
- AddRoundKey adds a round key word with each state column matrix;
- The operation in AddRoundKey is matrix addition.
- The AddRoundKey transformation is the inverse of itself.

AddRoundKey



Key expansion



Making of t_i (temporary) words $i = 4\ N_r$

- *RotWord-*
  - *Rotate Word*
  - *Takes a word as an array of 4 bytes*
  - *Shifts each byte to the left with wrapping*
- *SubWord-*
  - *Substitute Word*
  - *Takes each byte in the word and substitute another byte for it*
- *Round Constants- (Generated with some mathematical logic)*
  - *Rcon=4 byte value, Rightmost three bytes are always Zero*

- Each round key in AES depends on the previous round key.
- The dependency, however, is nonlinear because of SubWord transformation.
- The addition of the round constants also guarantees that each round key will be different from the previous one.

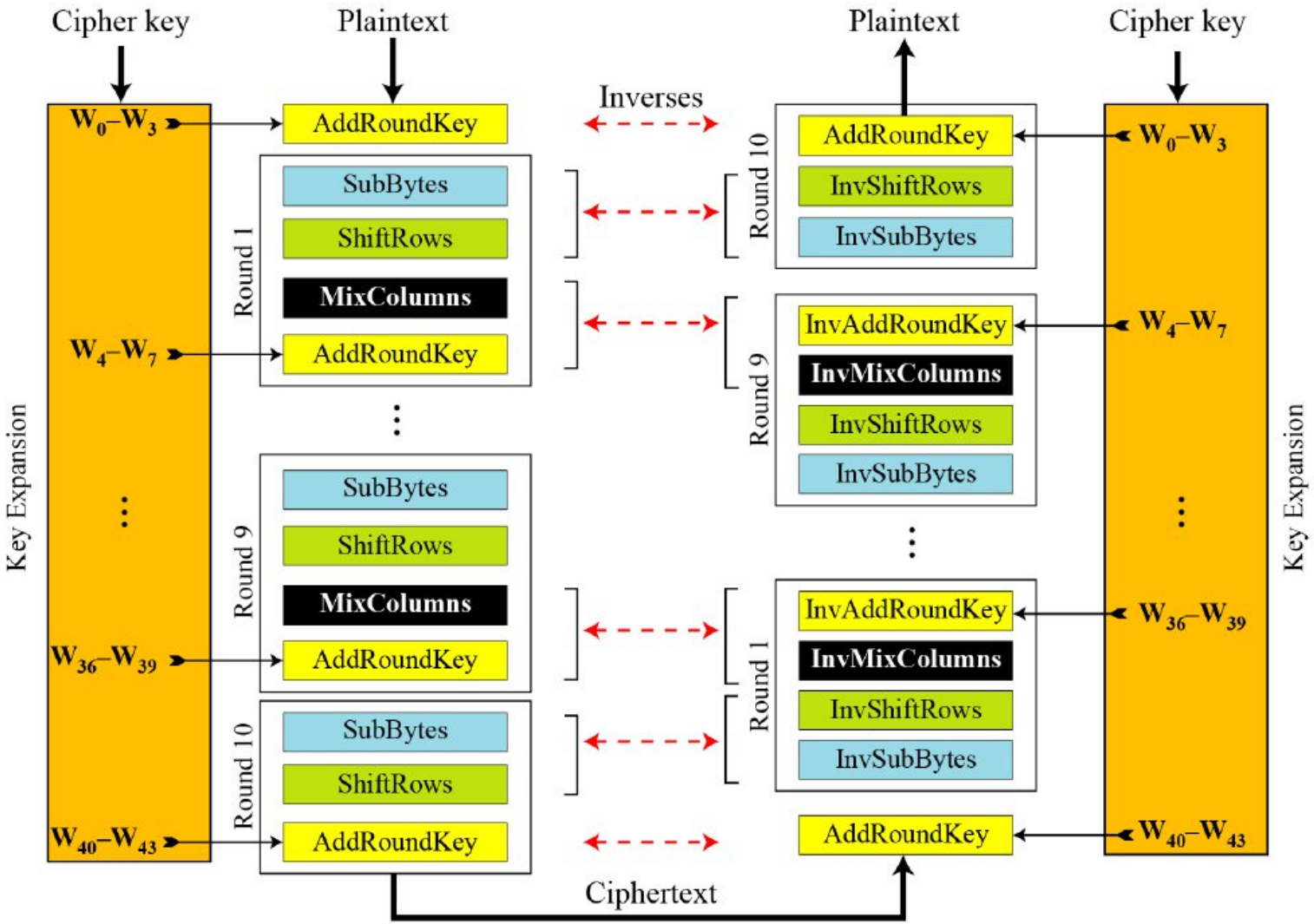
Key expansion analysis

- designed to provide several features that thwart the cryptanalyst.
- Two different Cipher keys, no matter how similar to each other, produce two expansions that differ in atleast a few rounds
- Each bit of cipher key is diffused into several rounds.
- Changing a single bit in the cipher key, will change some bits in several rounds
- No serious weak keys in AES

AES Decryption

- 🔄 Reverse process of encryption:
- Uses :
  - **Inverse SubBytes,**
  - **Inverse ShiftRows, and**
  - **Inverse MixColumns.**
- The **AddRoundKey** step remains the same.

AES process


Cipher key
Plaintext
Plaintext
Cipher key
Inverses
W_0–W_3
AddRoundKey
SubBytes
ShiftRows
MixColumns
AddRoundKey
Round 1
W_4–W_7
SubBytes
ShiftRows
MixColumns
AddRoundKey
Round 9
W_{36}–W_{39}
SubBytes
ShiftRows
AddRoundKey
Round 10
W_{40}–W_{43}
Key Expansion
Round 10
AddRoundKey
InvShiftRows
InvSubBytes
Round 9
InvAddRoundKey
InvMixColumns
InvShiftRows
InvSubBytes
Round 1
InvAddRoundKey
InvMixColumns
InvShiftRows
InvSubBytes
AddRoundKey
W_0–W_3
W_4–W_7
W_{36}–W_{39}
W_{40}–W_{43}
Key Expansion
Ciphertext

Why AES is Secure?

- **Large Key Size** – 128-bit key has 2^{128} possible keys.
- **Strong S-box** – Prevents linear and differential cryptanalysis.
- **No Known Practical Attacks** – Even **brute-force** attacks are infeasible.
- **Used by Governments & Organizations** – Approved for **Top Secret** U.S. data.

Applications of AES

- **Cybersecurity** – Secure communication, SSL/TLS encryption.
- **Banking & Finance** – Secure transactions and ATM encryption.
- **Mobile Security** – Used in encrypted messaging apps like WhatsApp.
- **Disk & File Encryption** – BitLocker, VeraCrypt, and other encryption tools.
- **Government & Military** – Protection of classified information.

Advantages of AES

- **High Security:** Resistant to known cryptographic attacks.
- **Speed:** Efficient in both hardware and software.
- **Flexibility:** Supports various key lengths.
- **Global Standard:** Widely adopted across industries.

Limitations of AES

- **Key Management Challenges:** Security relies heavily on key secrecy.
- **Vulnerability in Poor Implementations:** Side-channel attacks if not properly implemented.
- **Not Suitable for Very Large Data Streams** in its basic form without modes like CTR.

Thank you!!