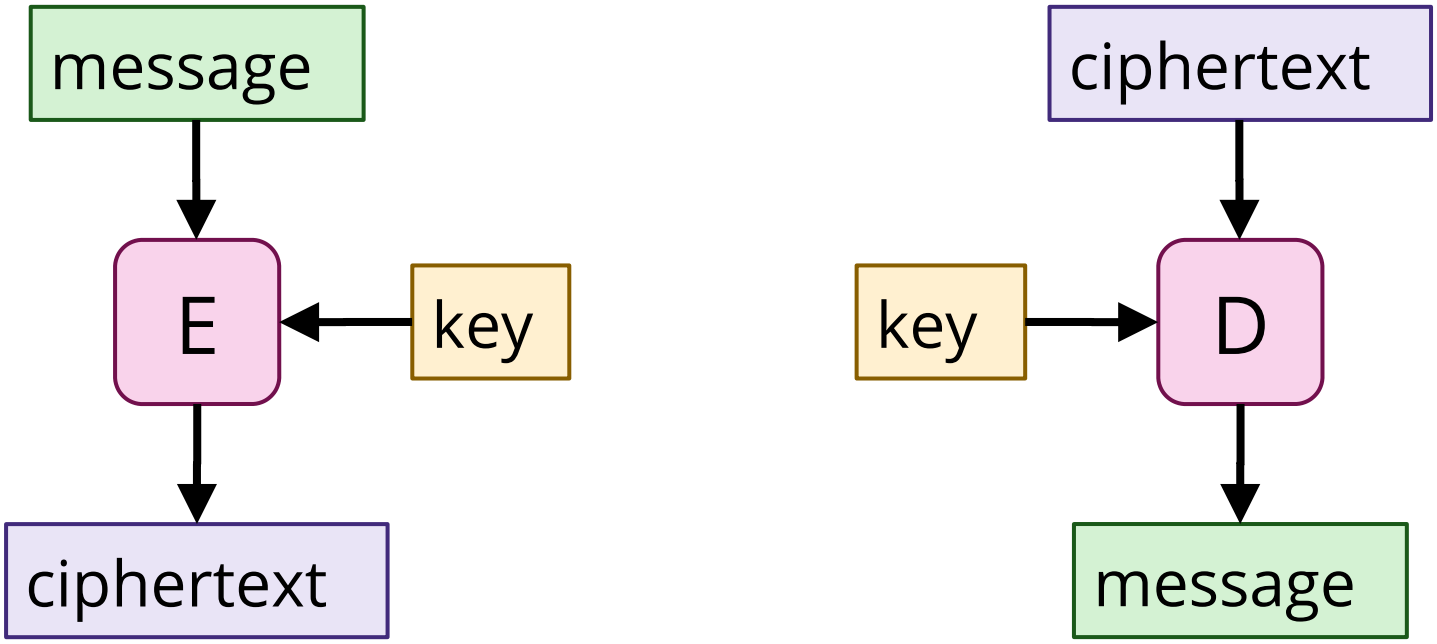
# Block Ciphers

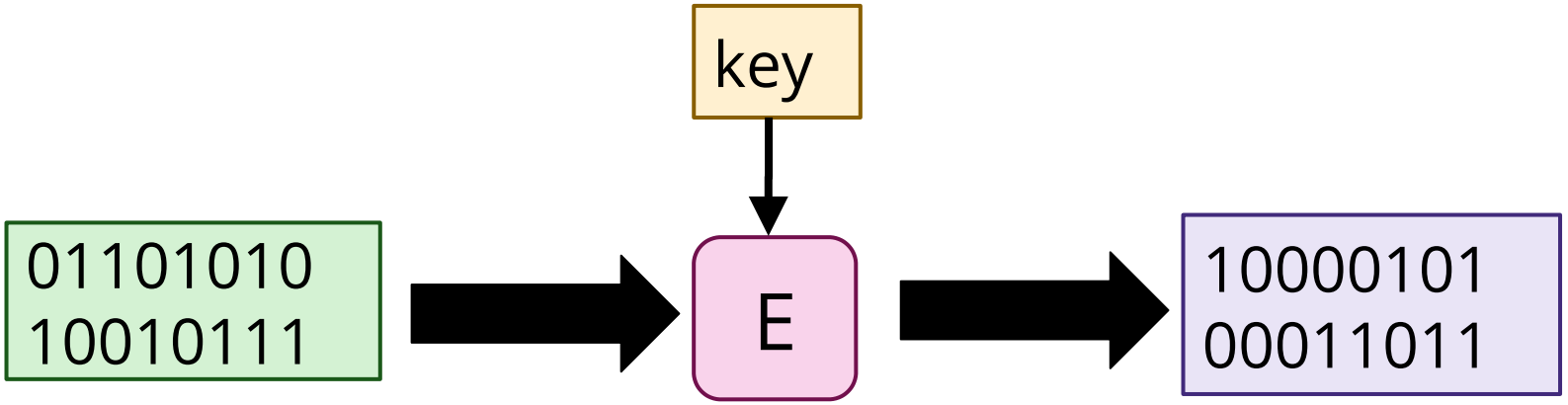
# The story so far…

Block ciphers are a type of **symmetric cryptography**



# The story so far…

Block ciphers operate on **blocks** of bits (e.g., 16 bits at a time)



# The story so far…

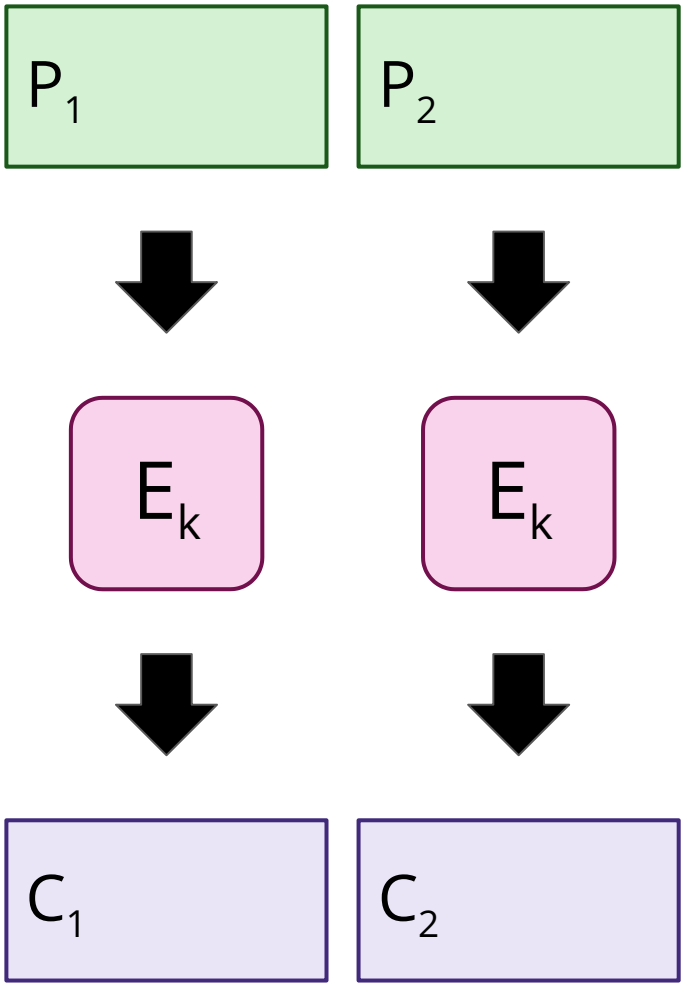
How we use a block cipher – the **mode of operation** – can be as important as its internals.

This relates to how we apply a block cipher to messages, which will typically be longer than a single block.

**Modes of Operation** we have covered:

1. ECB mode
2. CBC mode
3. CTR mode

# The story so far: ECB mode



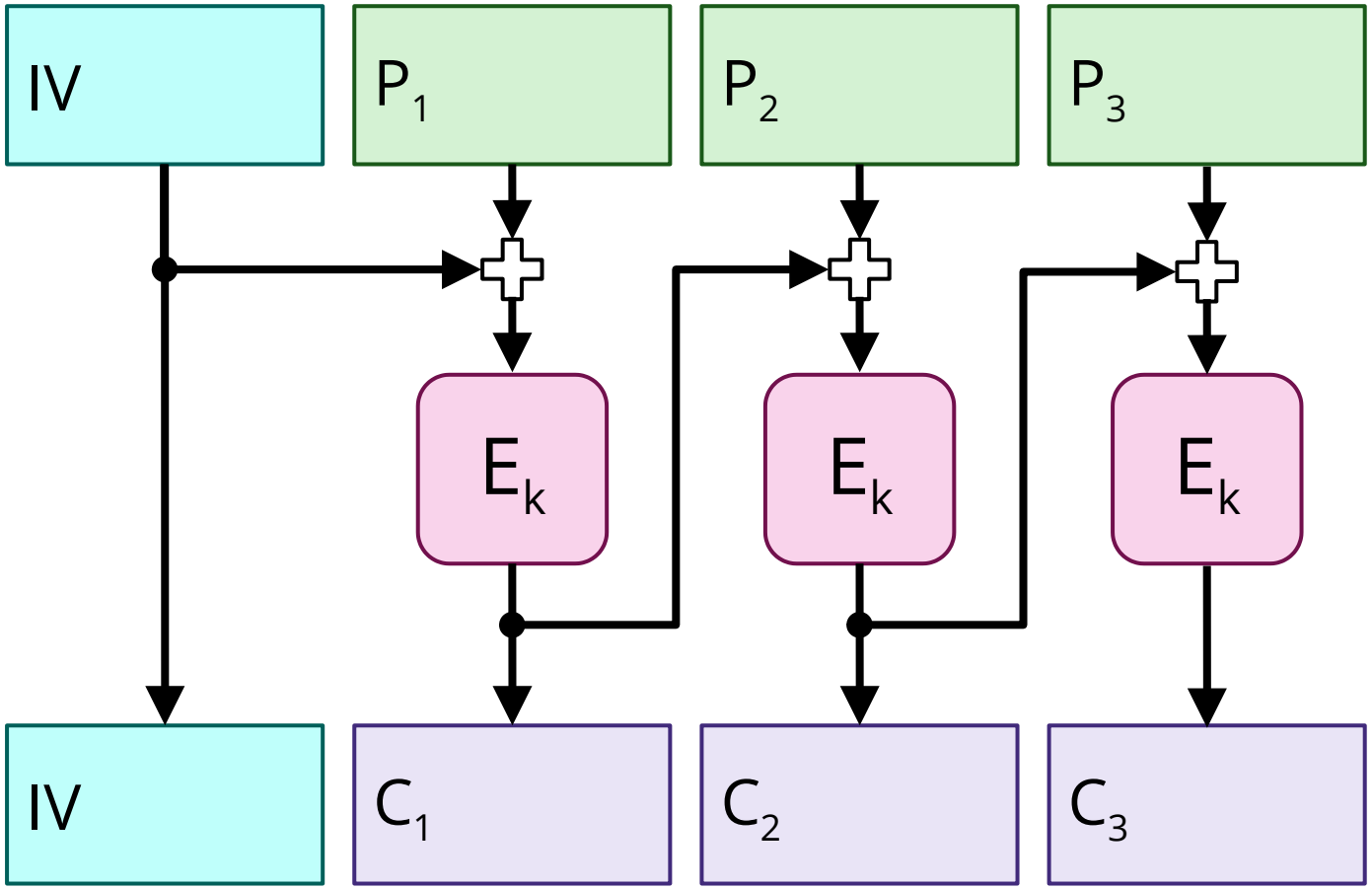

Using a block cipher like this is called ECB mode (electronic code book).

formula: $\mathbf{C_i = E_k(P_i)}$.

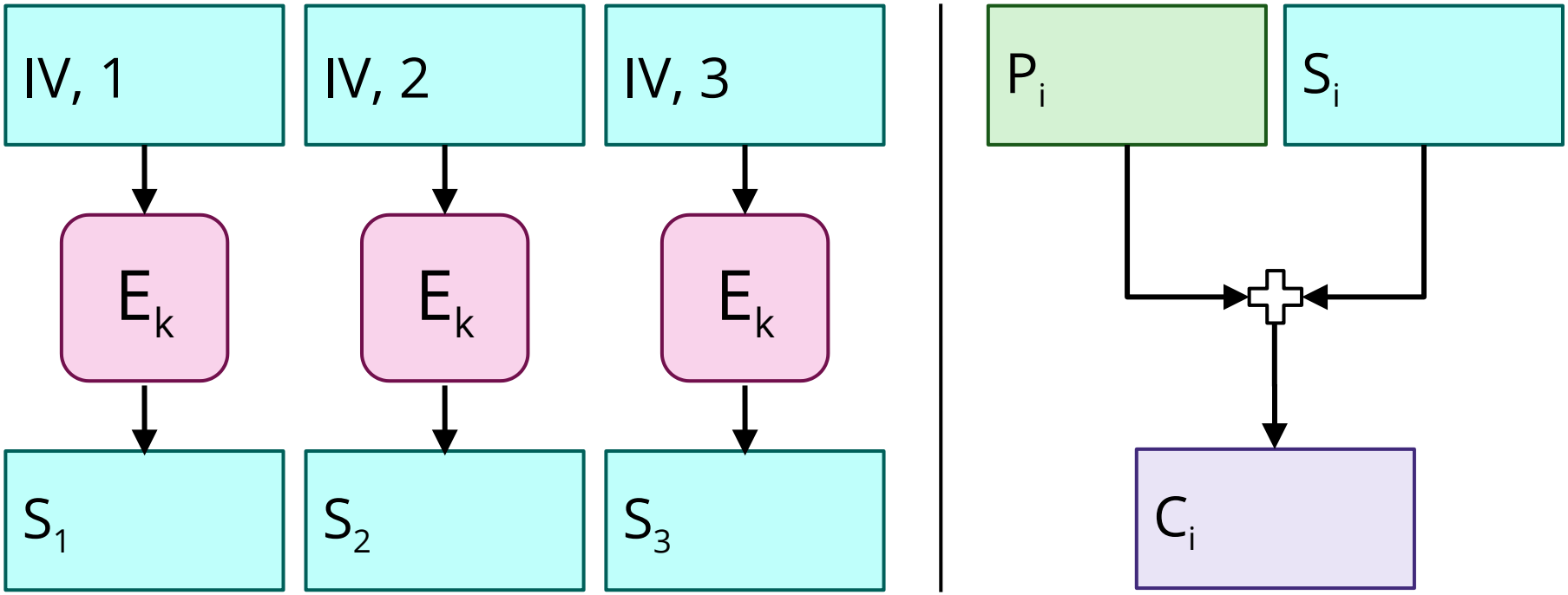
**Never do this!**

# The story so far: CBC mode

CBC = cipher block chaining



# The story so far: CTR mode




counter mode (CTR) - stream cipher

**$S_i = E_k(IV, i)$** **$C_i = P_i + S_i$**

# The story so far…

Focusing on the internals…
Block ciphers commonly have multiple internal ‘rounds’, with each round making use of part of the shared key in some way.

There are two important ‘families’ of block ciphers we have covered.

**1. Feistel networks**
**2. Substitution-permutation networks**

Another ‘family’ are the **Lai-Massey** ciphers.

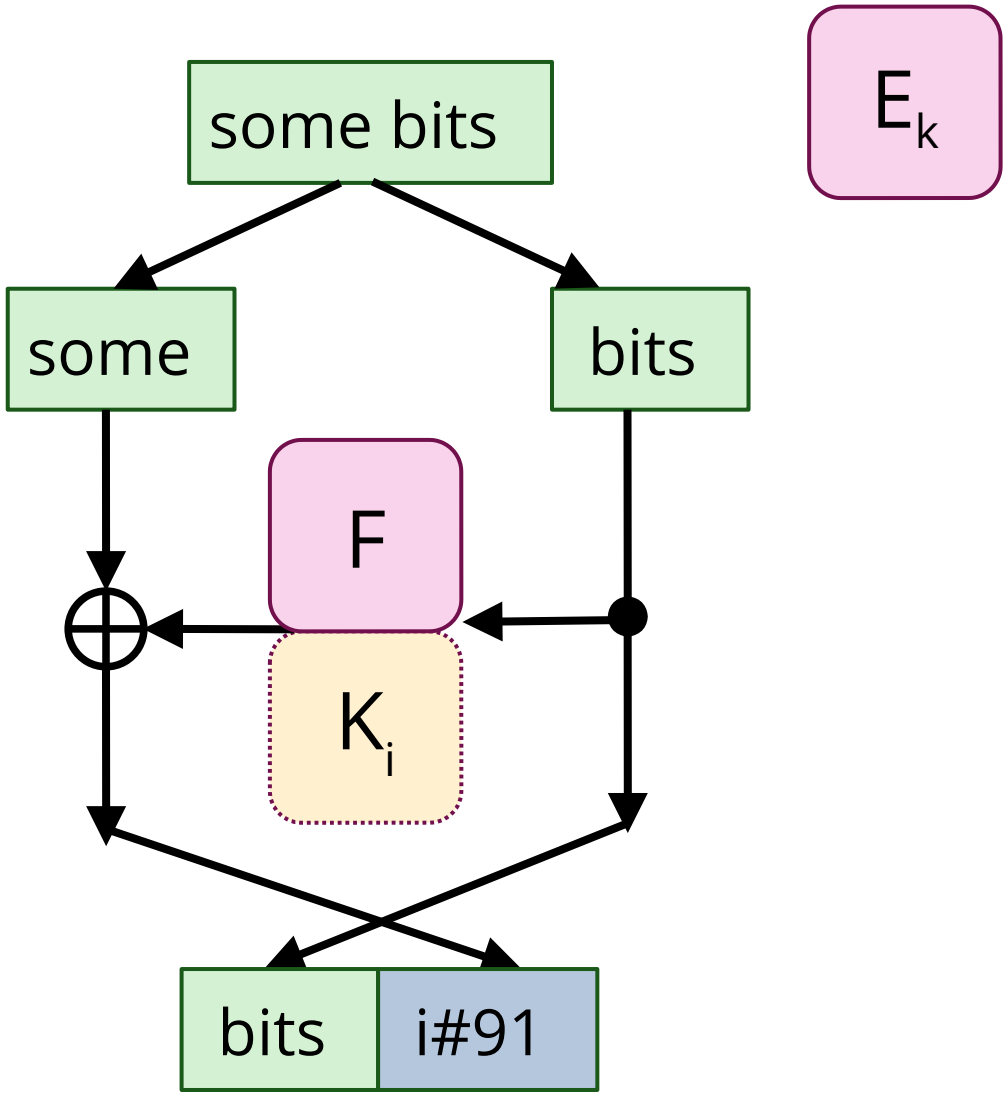
These are not specific ciphers, but general descriptions of a cipher’s approach, with particular functions or internal arrangements left unspecified.

# Block ciphers: approaches

**Feistel networks**
for each subkey $K_i$ in 0..N

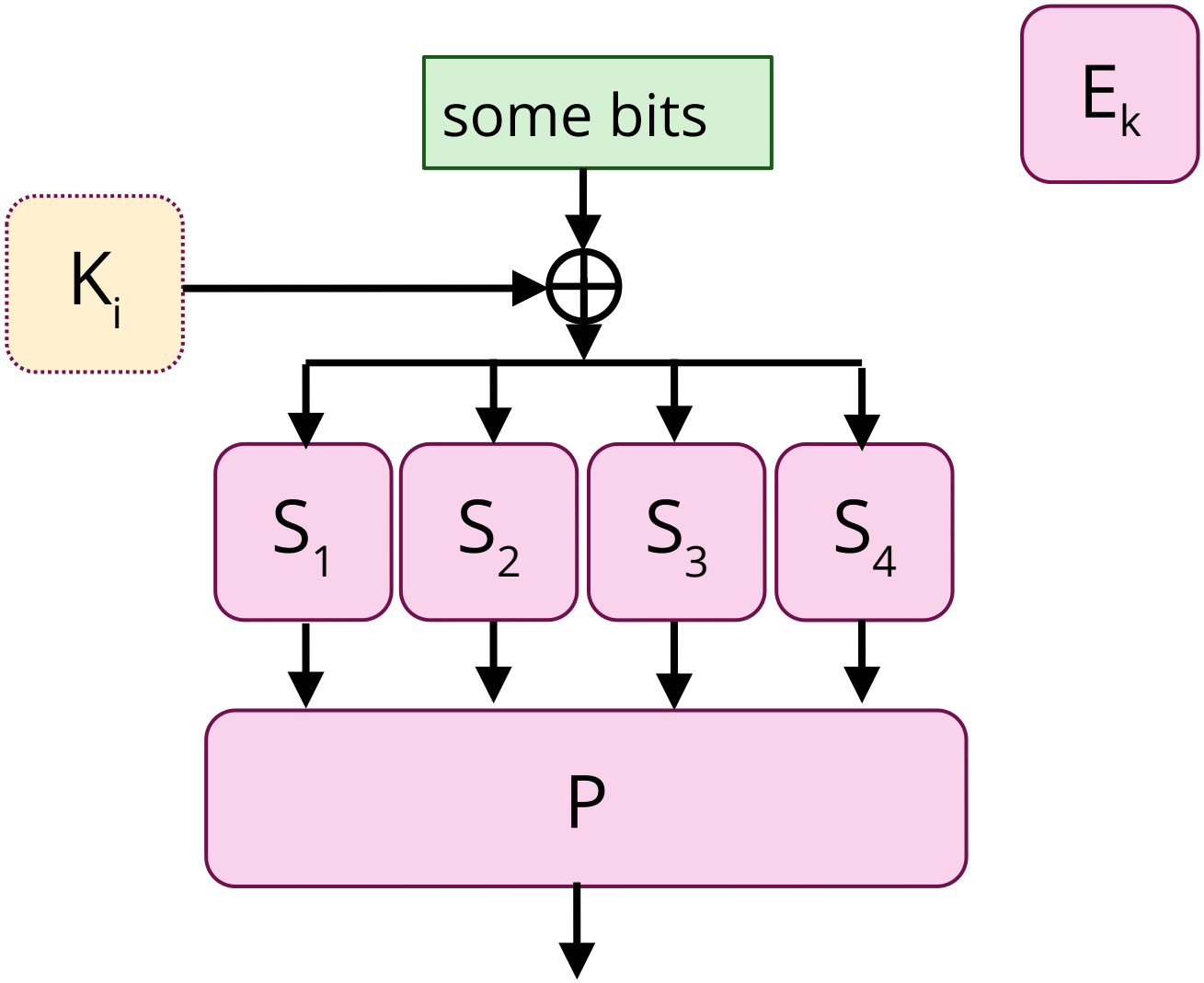
Used in DES, many other modern block ciphers

Easily reversible (saves code/circuits)



# Block ciphers: approaches

**Substitution-permutation networks**




**Substitution:** replace with something else

**Permutation:** shuffle the content around

# Example: DES

**Feistel network**

# The history of DES

(No longer a standard)

DES stands for **Data Encryption Standard**

Developed in the ‘70s based on work by Horst *Feistel*
The version submitted to the US National Bureau of Standards was modified slightly after a consultation with the NSA

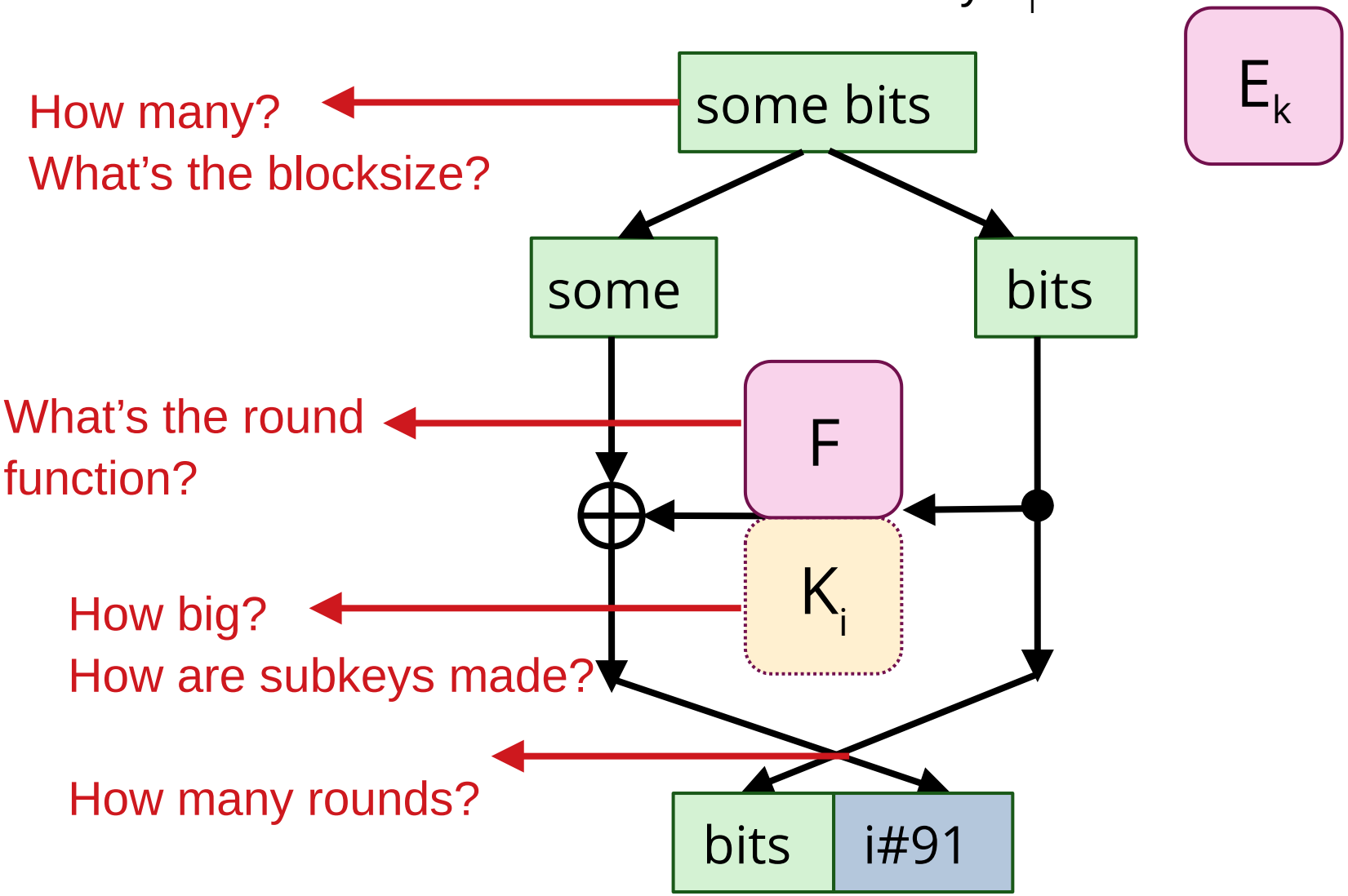
***DES is insecure***

However, it’s an influential example from a modern style of cryptography.

Triple DES (essentially: using DES three times) is more secure, but now also deprecated and considered a weak cipher.

# How does DES work?

**Feistel networks**
for each subkey $K_i$ in 0..N



# Blocksize & Keylength

DES works on blocks of **64 bits –** this is the blocksize.

The key in DES was intended to be (and notionally is) also **64 bits.**

However, the DES key is stored as 8 bytes, each with a **parity bit.**

This means the effective size of the DES key is only **56 bits**

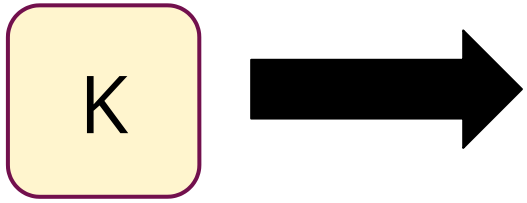
Difference between a $2^{56}$ key and $2^{64}$ key == 256 *times* fewer possible keys.

# Rounds & Subkeys

How many rounds? – **16**

How are subkeys created? *(key schedule)*

Complex process involving splitting the 56 bits into two halves, and selecting 24 bits from each half in every round (subkeys are 48 bits each)




$K_0$

$K_1$

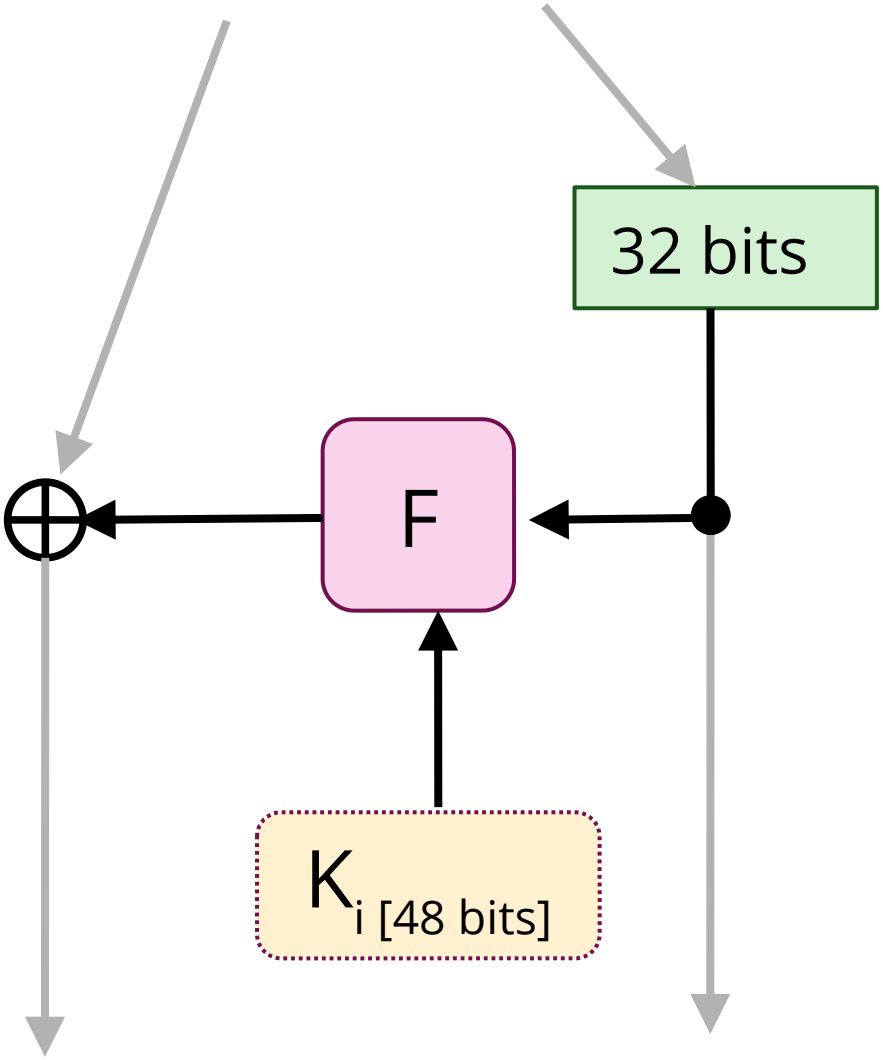
...

$K_{15}$

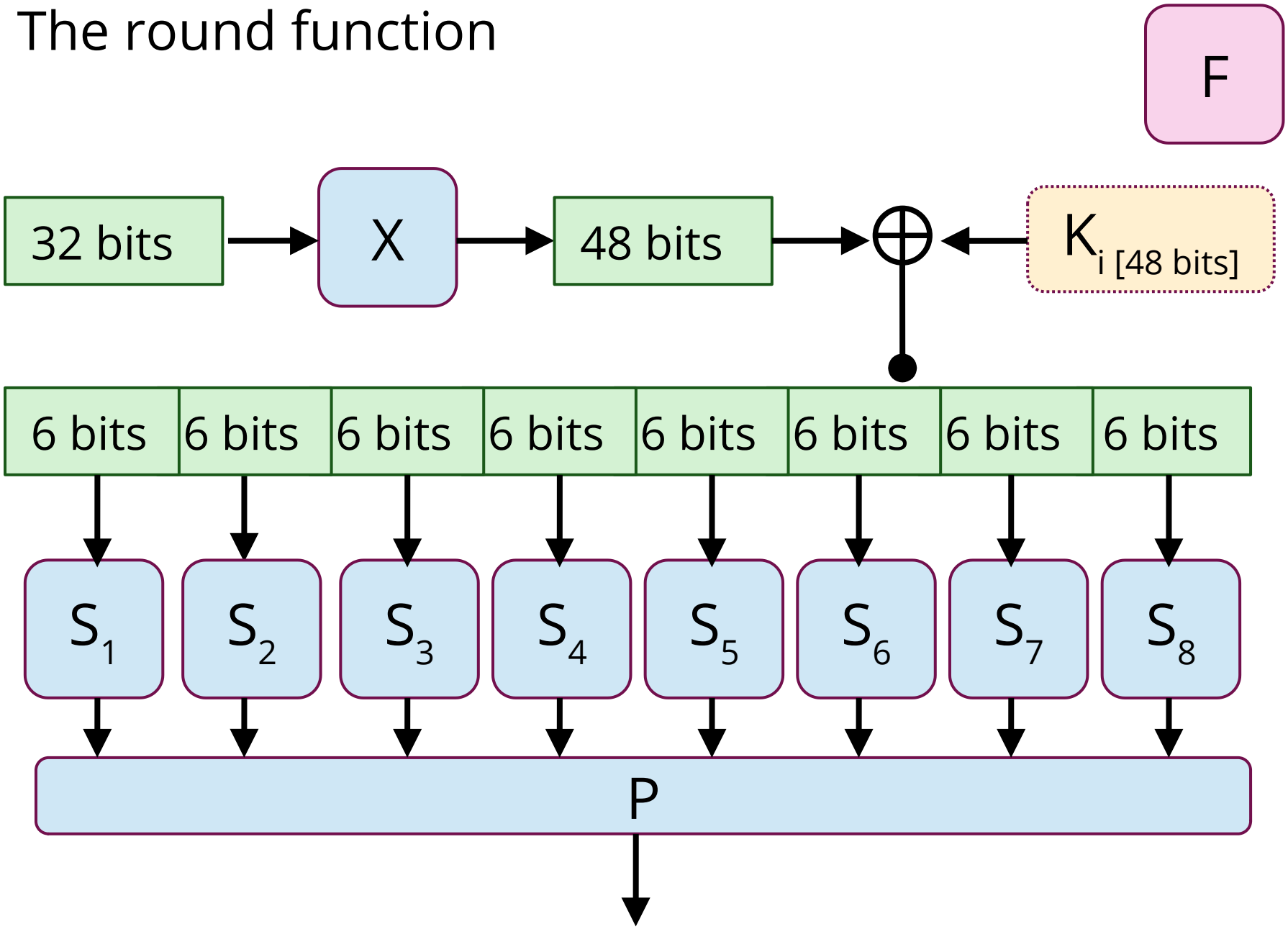
Each bit of the key is used in a subkey for ~14/16 rounds.

Key schedule reversed for decryption

# The round function



# The round function



# Example: AES

**Substitution-permutation network**

# How does AES work?

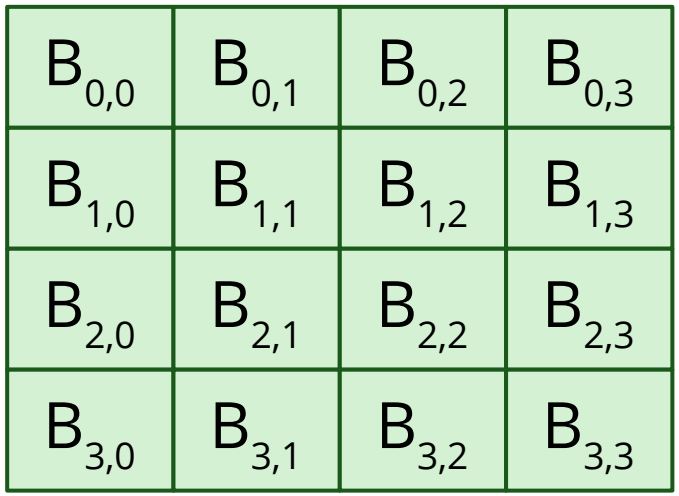
**Substitution-permutation network**

$E_k$

**Blocksize:** 128 bits (16 bytes)

The **number of rounds** varies according to the key length.

| *Keylength* | *Rounds* |
|---|---|
| 128 bits | 10 |
| 192 bits | 12 |
| 256 bits | 14 |

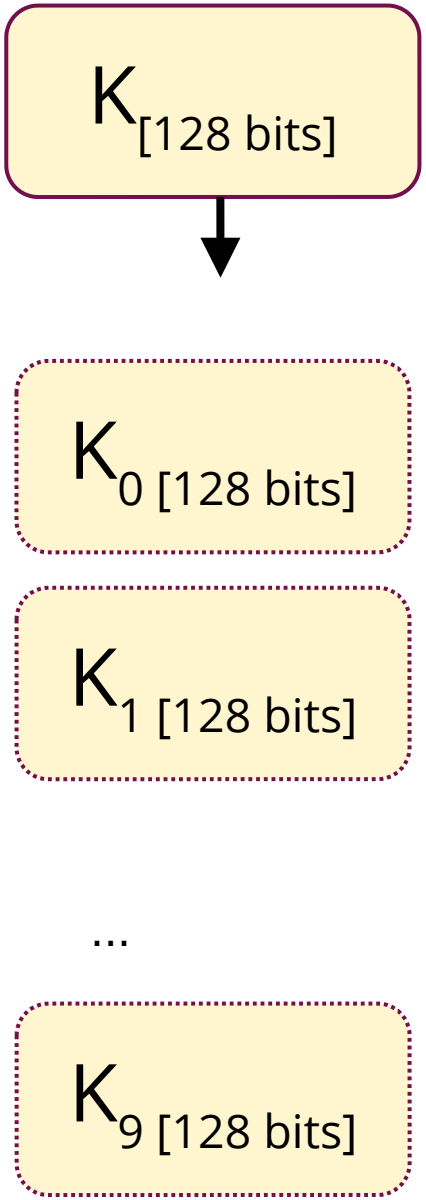


# The key schedule

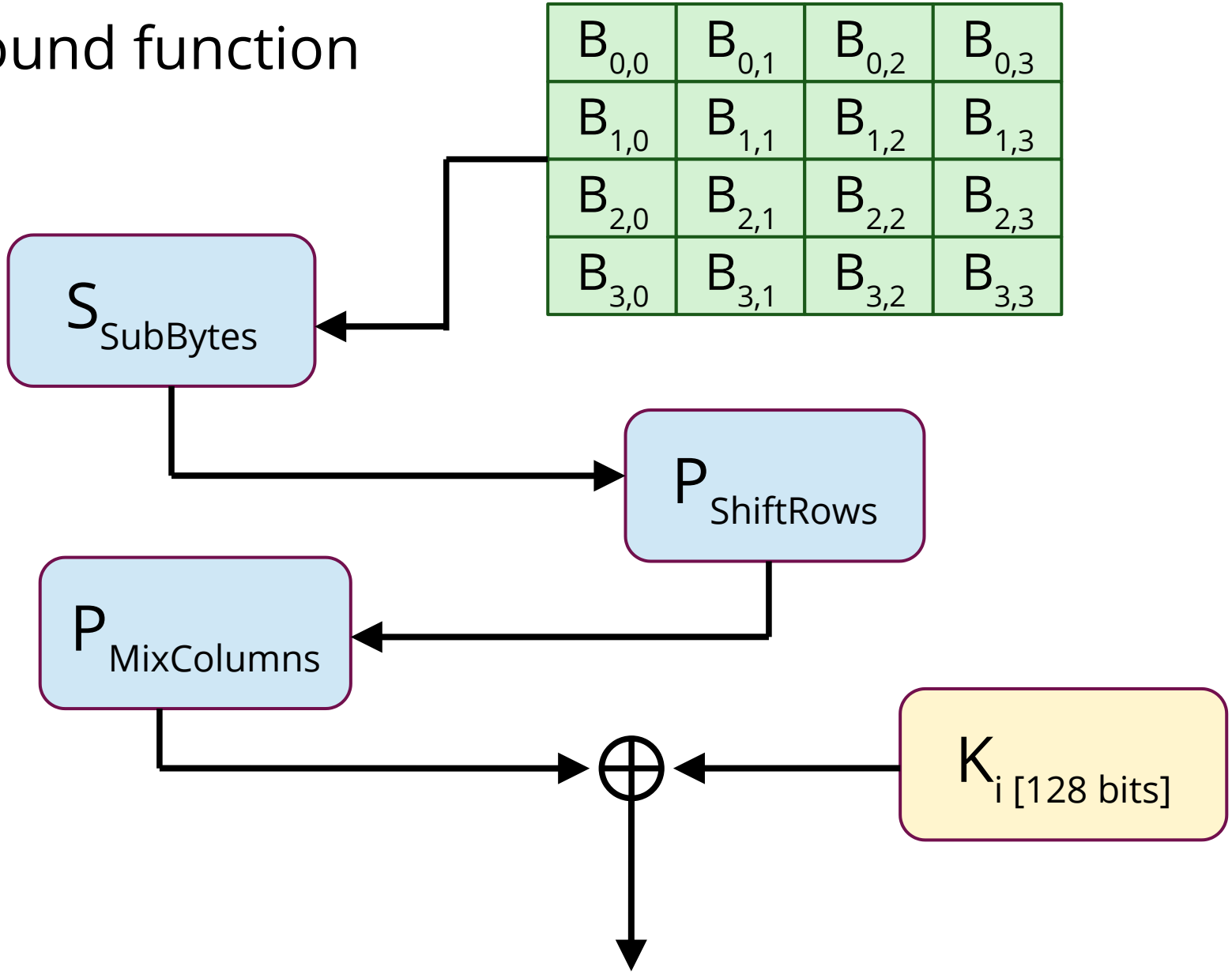
Need **rounds + 1** subkeys.
(Input block is XORed with first subkey)

Applies bitshifts and rotation to the original key, and XORs with parts of itself in a predictable fashion.

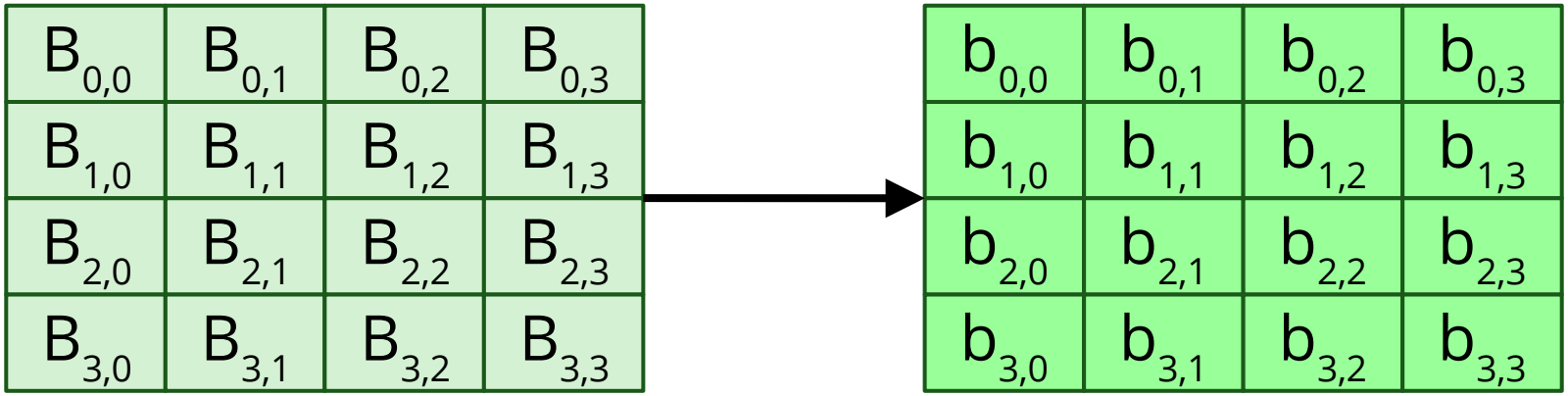
Produces a separate 128-bit subkey for each round.



# The round function



# The SubBytes step



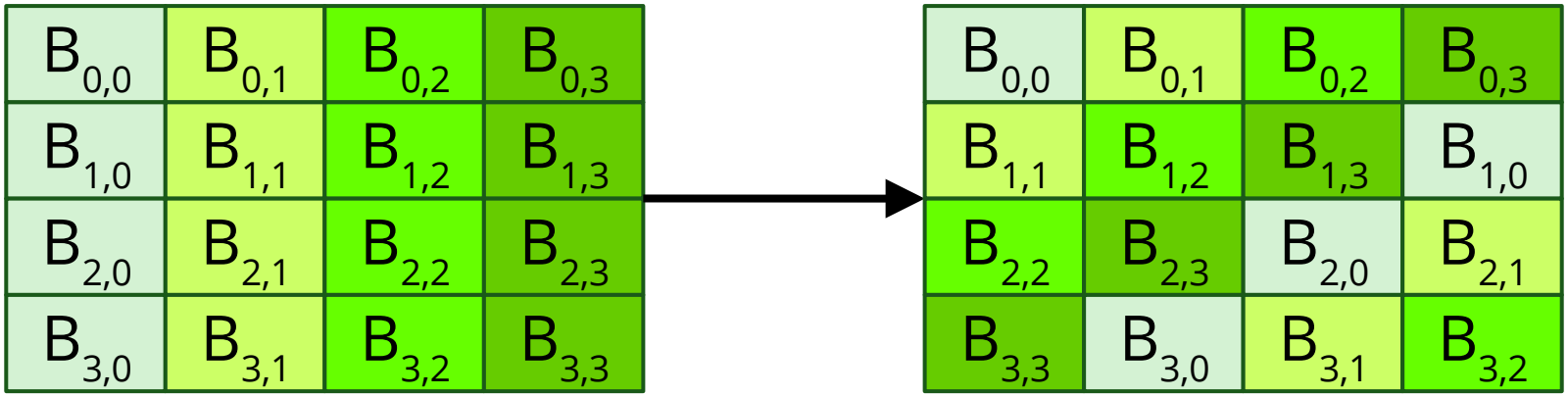

**Substitution** made through a lookup table known as the **Rijndael S-box**.

Each input byte is mapped to its multiplicative inverse, then transformed with an *affine transformation* which XORs together multiple rotations of the byte (<<< 1:4) along with a constant value **01100011**

# The ShiftRows step

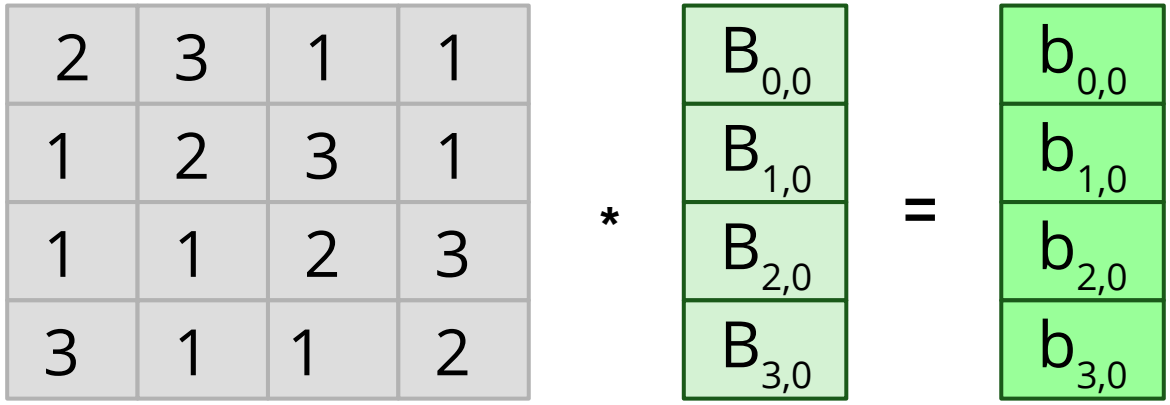
Shift each row
by **row-number** columns

| | | | | |
|---|---|---|---|---|
| 0 | $B_{0,0}$ | $B_{0,1}$ | $B_{0,2}$ | $B_{0,3}$ |
| < 1 | $B_{1,0}$ | $B_{1,1}$ | $B_{1,2}$ | $B_{1,3}$ |
| << 2 | $B_{2,0}$ | $B_{2,1}$ | $B_{2,2}$ | $B_{2,3}$ |
| <<< 3 | $B_{3,0}$ | $B_{3,1}$ | $B_{3,2}$ | $B_{3,3}$ |



# The MixColumns step

Each column is permuted by multiplication with a fixed matrix, in an invertible linear transformation.



# Decrypting AES

Unlike a Feistel network (like DES), AES has to operate differently for decryption.

*InverseSubBytes:* A reversed lookup table (B → A becomes A → B)

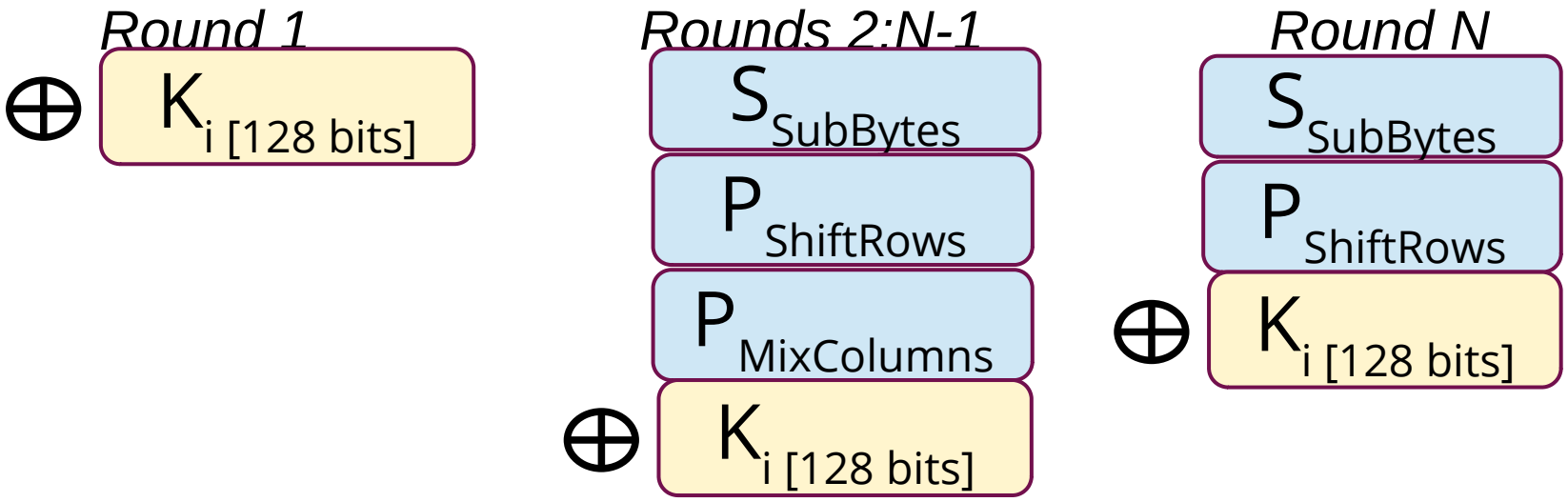
*InverseShiftRows:* Rows shift right instead of left.

*InverseMixColumns*: Use inversion matrix instead

| | | | |
|---|---|---|---|
| 14 | 11 | 13 | 9 |
| 9 | 14 | 11 | 13 |
| 13 | 9 | 14 | 11 |
| 11 | 13 | 9 | 14 |

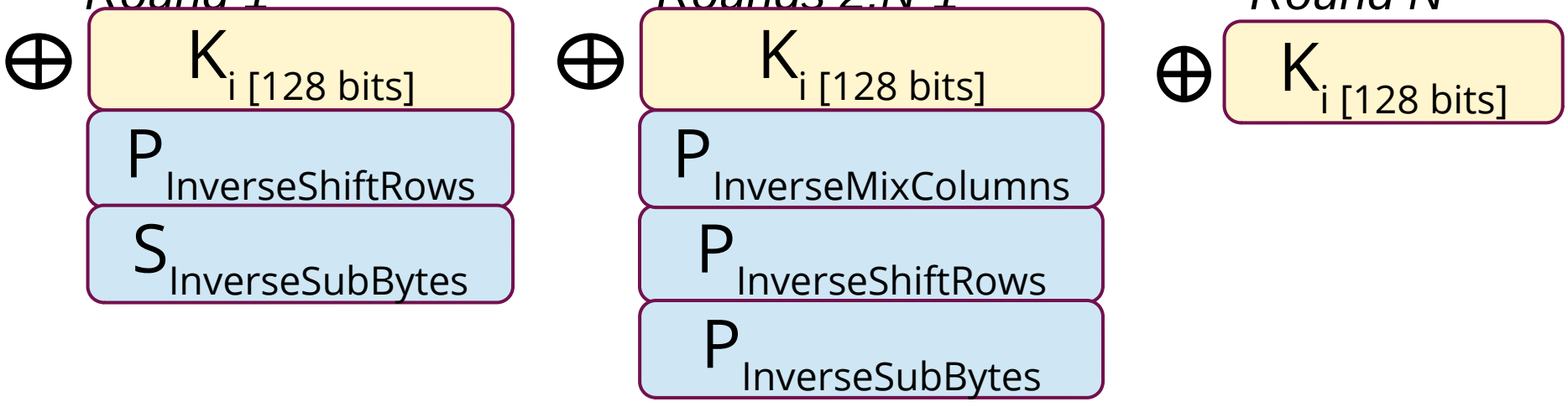
# AES: Encryption

**Encrypt**



# AES: Decryption

**Decrypt**



# Mock Exam

# Treasure Hunt

**Pencil** (<u>not</u> pen)

For making choices


For unmaking choices
(Optional for the confident)