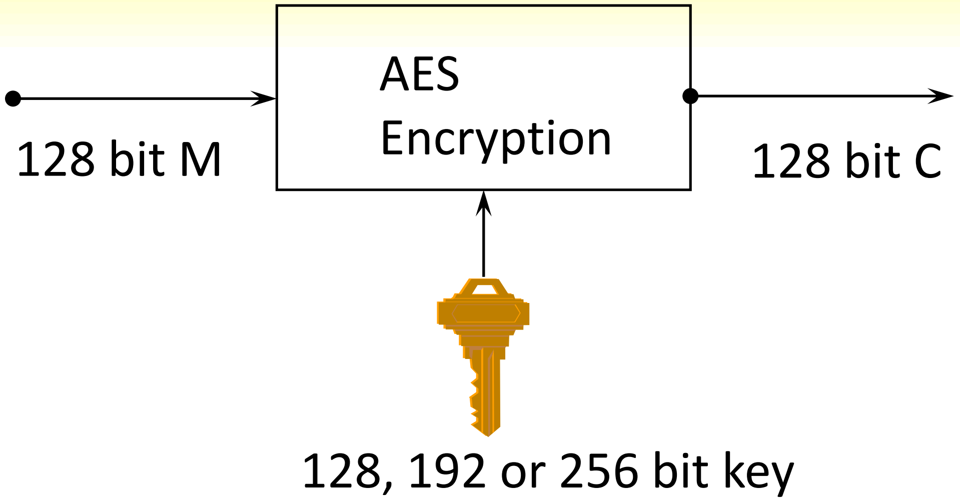
AES

McAlpin
CECS-UCF

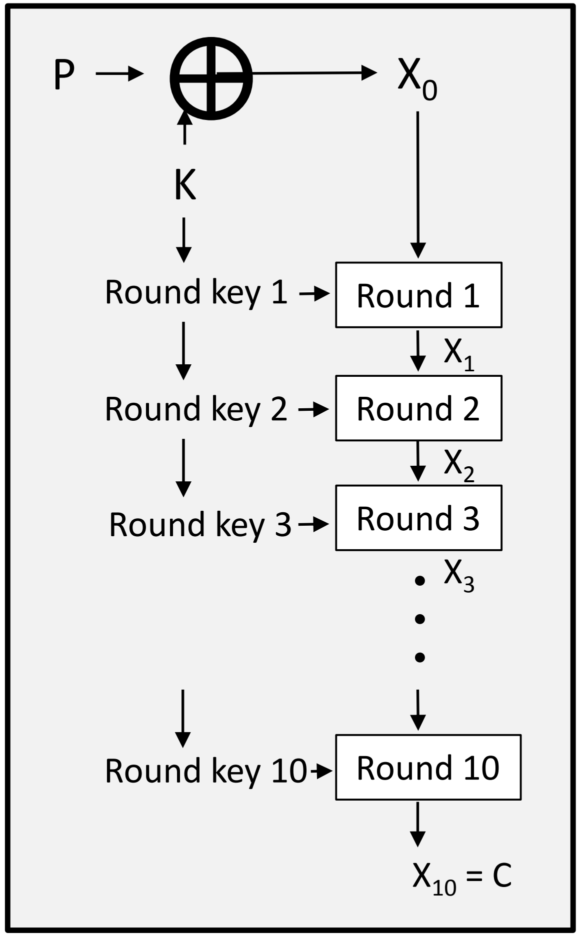
AES Background



- **Advanced Encryption Standard (AES)**
  - Published in 2001, standardized in 2002.
  - AES is based on **Rijndael** cipher structure (not Feistel)
    - ***Rijndael structure uses advanced mathematics (group, ring, and field theory) which we will not cover***
  - **Key size:** 128, 192 or 256 bits
  - **Block size:** 128 bits
  - Rounds (10) – similar to DES

AES Round Structure

- The 128-bit version of AES uses 10 rounds to encrypt each block of the input plaintext
- Each round performs an invertible transformation on a 128-bit array, arranged as a **4-byte by 4-byte square array** called the **state**.
- The initial state X_0 is the **XOR** of the plaintext **P** with the key **K**: $X_0 = P \oplus K$.
- Round i (i = 1, …, 10) receives state X_{i-1} as input and produces state X_i.
- The ciphertext C (for the block) is the output of the final round: $C = X_{10}$.

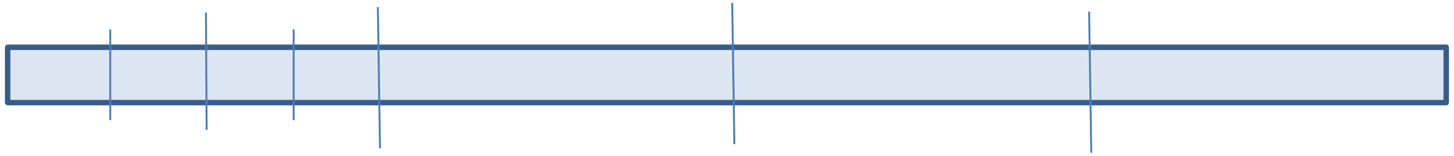


AES Round Processing

- *In each round*, the state undergoes:
  - **SubBytes step:**
    - byte *substitution*: same S-box used on ***every*** element of the state (8 bits each)
  - **ShiftRows step:**
    - shift rows: *permutation* of the bytes in each row
  - **MixColumns step:**
    - *mix values in each column* using matrix multiplication
    - basically, applies a Hill Cipher to each column
  - **AddRoundkey step:**
    - XOR the state with the *round key* derived from the 128-bit encryption key

State Representation of 128-bit Block

128 bits = 16 bytes of 8 bits each – interpreted in column major order



$(b_{0,0} | b_{1,0} | b_{2,0} | b_{3,0} | b_{0,1} | b_{1,1} | b_{2,1} | b_{3,1} | b_{0,2} | b_{1,2} | b_{2,2} | b_{3,2} | b_{0,3} | b_{1,3} | b_{2,3} | b_{3,3})$

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

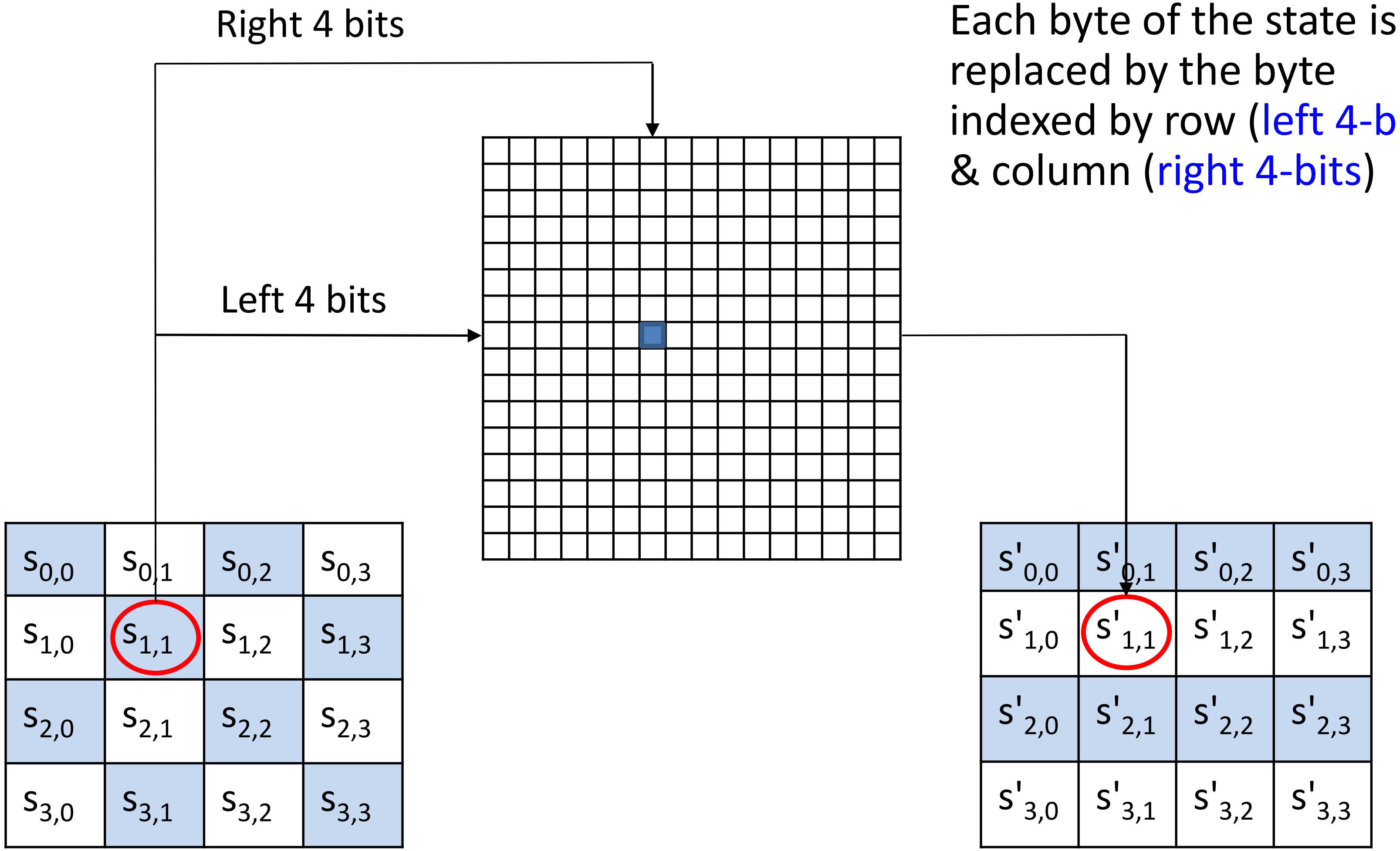
This array is called the **State**

Each group of 8 bits represented as 2 hex characters

SubBytes Step: Byte Substitution

S-boxes created:
Using number/group theory

Each byte of the state is replaced by the byte indexed by row (left 4-bits) & column (right 4-bits)



S-Box (16 x 16)

Example: Byte {95} is replaced by byte in row 9 column 5, which has value {2A}

		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

source: Table 20.2(a)

Sub Bytes – S-Box Encrypt - Example

00	01	02	22
23	24	25	80
81	82	a1	a2
b3	b4	b5	ff

Input – 16 bytes of hex

S-box Encrypt

63	7c	77	93
26	36	3f	cd
0c	13	f1	1a
4b	c6	d2	16

Output – 16 bytes of hex

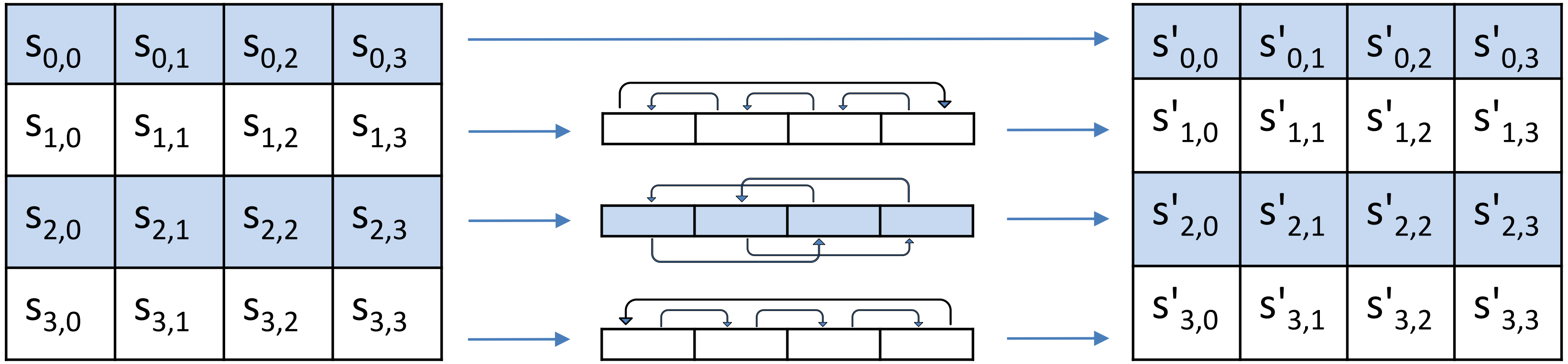
Inverse S-Box

NOTE: This table is used for decryption

x \ y	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
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

source: Table 20.2(b)

ShiftRows Step



1st row is unchanged
2nd row does 1 byte circular shift to left
3rd row does 2 byte circular shift to left
4th row does 3 byte circular shift to left

NOTE: same set of shifts every time

MixColumns Step

- **Each column is processed separately**
- Each byte is replaced by a value dependent on all 4 bytes in the column
- Mix columns matrix is part of AES, just like the S-box and Inverse S-box

Mix Columns Matrix *State* *New State*

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

Example: $s'_{1,2} = 1*s_{0,2} + 2*s_{1,2} + 3*s_{2,2} + 1*s_{3,2}$

Mix columns fun… with Galois

- AES security is entirely based on $GF(2^8)$ (see Galois Fields) using irreducible polynomial:
  - $x^8 + x^4 + x^3 + x + 1$ …
- A byte is not interpreted as binary but
  - As a special polynomial of at most degree 7
- 63_x -> 0110 0011
  - $X^6 + X^5 + X + 1$
- $63_x = 99_{10}$
  - $2^6 + 2^5 + 2 + 1$
  - Replace 2 with x to make the polynomial
- $GF(2^8)$
  - 8 coefficients
  - Each with value of {0,1}
- Big idea?
  - Define addition, subtraction & multiplication on the polynomials
    - Have closure
    - Be valid polynomials

Wait? There's a problem?

- Calculate coefficients Mod 2 (similar to XOR)
- $(x^6 + x^5 + x + 1) + (x^3 + x^2 + x)$

$$= (x^6 + x^5 + x^3 + x^2 + 2x + 1)$$

$=$ REDUCE coefficients Mod 2 (Xor)

$$= (x^6 + x^5 + x^3 + x^2 + 1)$$

- Note that subtraction is identical to addition & both are essentially like Xor
- Hard part?
- Multiplication
- $(x + 1)(x^6 + x^5 + x + 1)$

$$= (x^7 + x^6 + x^2 + x) + (x^6 + x^5 + x + 1)$$

$$= x^7 + 2x^6 + x^5 + x^2 + 2x + 1$$

$$= x^7 + x^5 + x^2 + 1$$

Hmmm... About that problem

- Multiplying by x is usually left shifting the bits, in this case 8 of 'em, 1 bit to the left then Xor the shifted bits with the original value

  0110 0011

  <u>1100 0110</u>

  = 1010 0101
- The problem?

  $x(x^7 + x^5 + x^2 + 1)$

  $x^8 + x^6 + x^3 + x$
- … overflow
- So, mod by the AES (GFx^8) factor
- $x^8 + x^6 + x^3 + x$ mod $x^8 + x^4 + x^3 + x + 1$
- Consider $x^8 \Rightarrow x^4 + x^3 + x + 1$
- THEREFORE
  - See x^8?
  - Replace with $x^4 + x^3 + x + 1$

The solution – in example[1]

- Recap, given

  $x^8 + x^6 + x^3 + x$
- Replace x^8 with $x^4 + x^3 + x + 1$ then

  $(x^4 + x^3 + x + 1) + x^6 + x^3 + x$

  $x^6 + x^4 + 1$

- Consider

  1010 0101 multiply by 2

  1 0100 1010

  0100 1010

  <u>0001 1011</u>

  0101 0001
- Same algorithm as shown earlier

The solution – in example[2]

- Given
- $(x + 1)(x^7 + x^5 + x^2 + 1)$

  $x(x^7 + x^5 + x^2 + 1) + (x^7 + x^5 + x^2 + 1)$

  $x^6 + x^4 + 1 + (x^7 + x^5 + x^2 + 1)$

  $x^7 + x^6 + x^5 + x^4 + x^2$

- Hmmm… 03 x A5 = F4

  $(x + 1)(x^7 + x^5 + x^2 + 1) = x^7 + x^6 + x^5 + x^4 + x^2$

Mixin' some columns…

- Given the ***FIXED*** matrix

02	**03**	**01**	**01**
01	02	03	01
01	01	02	03
03	01	01	02

X

63	**7c**	77	93
36	**3f**	cd	26
f1	**1a**	0c	13
16	**4b**	c6	d2

=

02 x 7c + 03 x 3f + 01 x 1a + 01 x 4b = e8

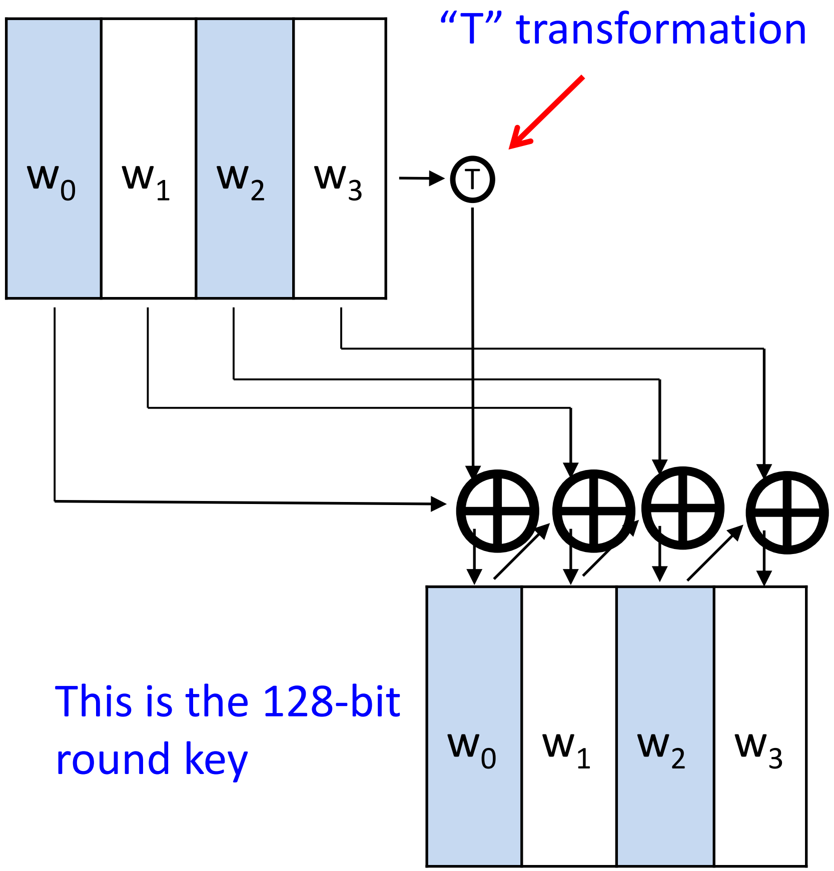
AddRoundKey Step

$$
\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} \oplus \begin{bmatrix} w_i & w_{i+1} & w_{i+2} & w_{i+3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}
$$

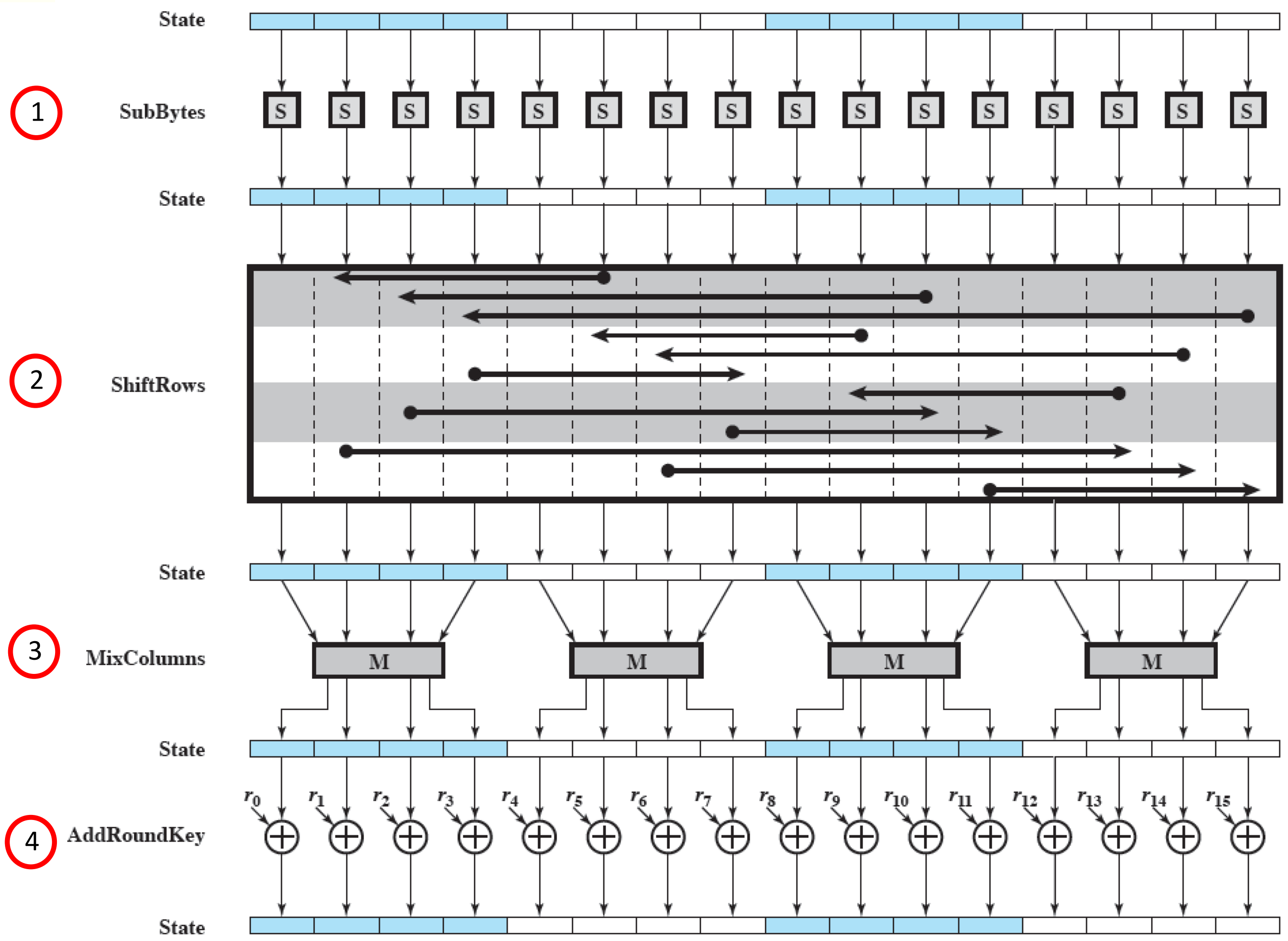
Exclusive-or (XOR) the state with a ***set*** of keys for the round, derived from the 128-bit secret key

AES Key Schedule (Expansion)

- ***The 128-bit key is first divided into four 4-byte "words" (represented as columns in diagram)***
- ***First column of round key is*** computed as XOR of previous round first column and the "T" transformation of the previous round last column.
- "T" transformation involves
  - Cyclical left shifts
  - S-box substitutions
  - XOR first byte with a "round" constant
- ***Remaining columns*** computed in order using XOR of previous round column value and the preceding column in the current round.



AES Encryption Round



source: Fig. 20.4

AES Assembler instructions

- For Intel & AMD:

Instruction	Description
AESENC	Perform one round of an AES encryption flow
AESENCLAST	Perform the last round of an AES encryption flow
AESDEC	Perform one round of an AES decryption flow
AESDECLAST	Perform the last round of an AES decryption flow
AESKEYGENASSIST	Assist in AES round key generation
AESIMC	Assist in AES Inverse Mix Columns
PCLMULQDQ	Carryless multiply (CLMUL)

Resources

- NIST Specification
  - https://www.nist.gov/publications/advanced-encryption-standard-aes
- Intel – AES instructions w. explanation
  - https://software.intel.com/content/www/us/en/develop/articles/intel-advanced-encryption-standard-instructions-aes-ni.html
- Source code library (in C)
  - https://github.com/BrianGladman/aes