Chapter 6

Advanced Encryption Standard

Advanced Encryption Standard (AES)

- Also know as Rijndael (Rain-Dal)
  - Rijndael is the name of the algorithm adopted by NIST in 2001 as the Advanced Encryption Standard
- Rijndael was designed by two Belgian Cryptographers – Vincent **Rij**men and Joan **Dae**men
- AES is a family of three algorithms each having a block size of 128-bits and differing in key size i.e. AES-128, AES-192 and AES-256
  - For AES-*n*, *n* is the key size

Advanced Encryption Standard (AES)

- AES is a
  - Symmetric
  - Block cipher
  - Product cipher – number of rounds differs based on the key size
- Each round of AES has four transformations
  - SubBytes
  - ShiftRows
  - MixColumns
  - AddRoundKey
- AES operates in finite field of $GF(2^8)$
  - Uses $x^8 + x^4 + x^3 + x + 1$ as its prime polynomial

Finite Field Arithmetic

- In the Advanced Encryption Standard (AES) all operations are performed on 8-bit bytes
- The arithmetic operations of addition, multiplication, and division are performed over the finite field GF(2^8)
- A field is a set in which we can do addition, subtraction, multiplication, and division without leaving the set
- Division is defined with the following rule:
  - $a / b = a (b^{-1})$
- An example of a finite field (one with a finite number of elements) is the set Z_p consisting of all the integers $\{0, 1, \ldots, p - 1\}$, where p is a prime number and in which arithmetic is carried out modulo p

Finite Field Arithmetic

If one of the operations used in the algorithm is division, then we need to work in arithmetic defined over a field

- Division requires that each nonzero element have a multiplicative inverse

For convenience and for implementation efficiency we would like to work with integers that fit exactly into a given number of bits with no wasted bit patterns

- Integers in the range 0 through $2^n - 1$, which fit into an *n*-bit word

$n = 8$

$[0, 255]$

$[0, 2^n - 1]$

$2^n - 1$

The set of such integers, Z_{2^n}, using modular arithmetic, is not a field

- For example, the integer 2 has no multiplicative inverse in Z_{2^n}, that is, there is no integer *b*, such that $2b \bmod 2^n = 1$

A finite field containing 2^n elements is referred to as $GF(2^n)$

- Every polynomial in $GF(2^n)$ can be represented by an n-bit number

$\boxed{GF(2^n)}$

$\{0, 1, .., n-1\}$ Z_8

Z_{2^n}

$$\cdots \underset{2^3}{\_\_} \quad \underset{2^2}{\overset{0,1}{\_\_}} \quad \underset{2^1}{\overset{0,1}{\_\_}} \quad \underset{2^0}{\overset{0,1}{\_\_}}$$

$1 \; 0 \; 1$

$$2^2 + 0(2^1) + 2^0 = 4 + 1 = 5$$

$$\begin{array}{ccccc} 16 & 8 & 4 & 2 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \end{array}$$

$$16 + 2 + 1 = 19$$

$$\begin{array}{cccccc} 32 & 16 & 8 & & & \\ 1 & 1 & 0 & 1 & 1 & 0 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \end{array}$$

$$32 + 16 + 0 + 4 + 2 + 0 = 54$$

$$\alpha x^7 + x^6 + x^5 + x^4 + \underset{2}{x^3} + \underset{0}{x^2} + \underset{0}{x+1}$$

$GF(2^8)$:

$0 \longrightarrow 0 \longrightarrow 0$

$1 \longrightarrow 01 \longrightarrow 1$

$2 \longrightarrow 10 \longrightarrow x$

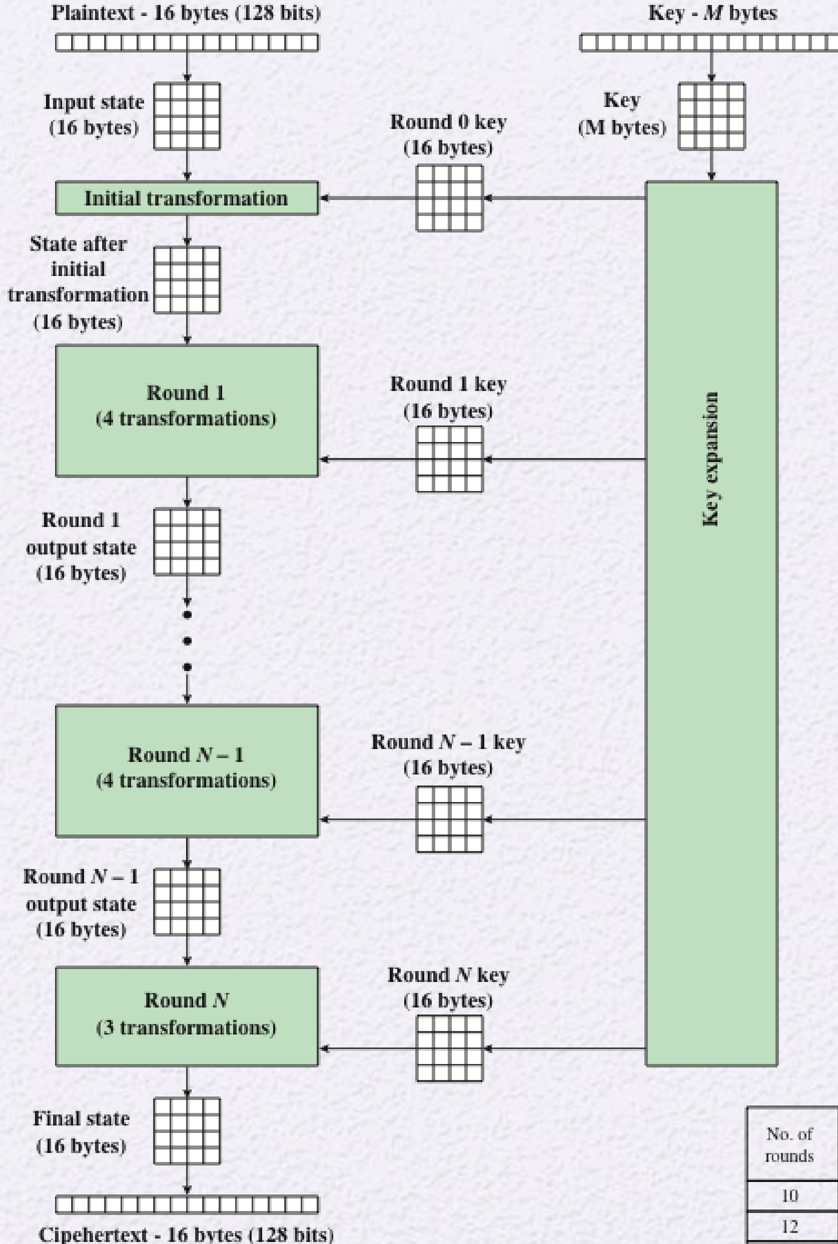
$3 \longrightarrow 11 \longrightarrow x+1$

$4 \longrightarrow 100 \longrightarrow x^2$

$5 \longrightarrow 101 \longrightarrow x^2+1$

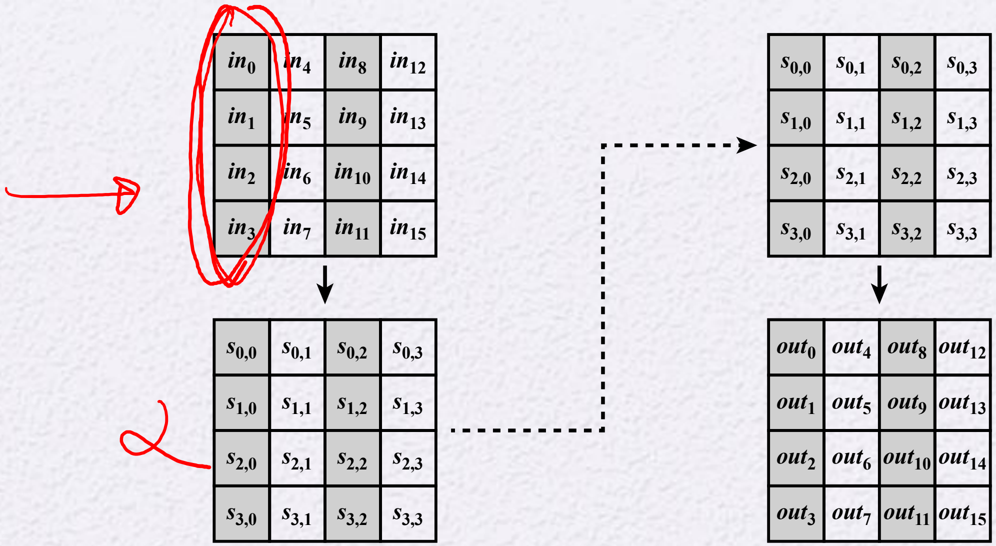
$\vdots \qquad\qquad\qquad \vdots$

$255 \longrightarrow 11111111 \longrightarrow x^7 + x^6 + x^5 + \cdots + 1$

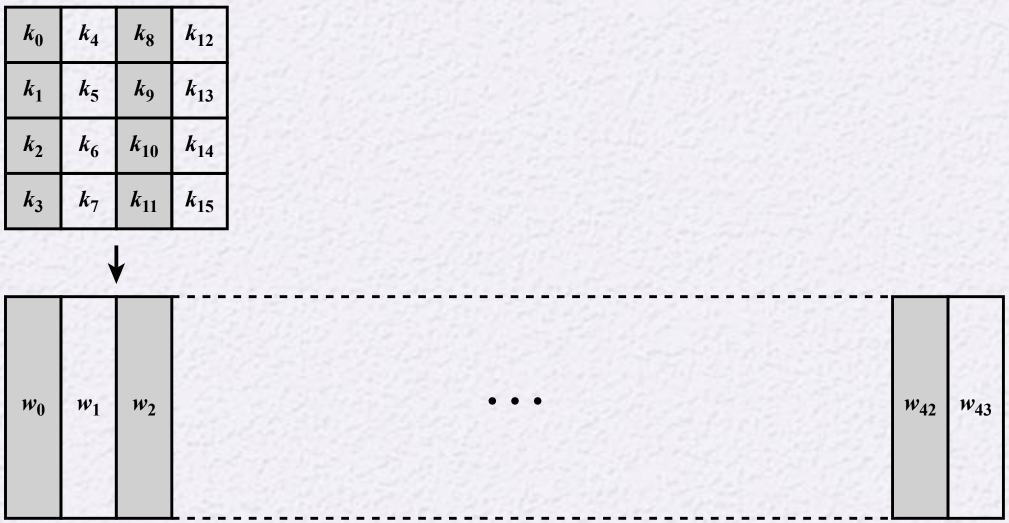


No. of rounds	Key Length (bytes)
10	16
12	24
14	32

Figure 6.1 AES Encryption Process



(a) Input, state array, and output



(b) Key and expanded key

Figure 6.2 AES Data Structures

Table 6.1
AES Parameters

	AES - 128	AES - 192	AES - 256
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

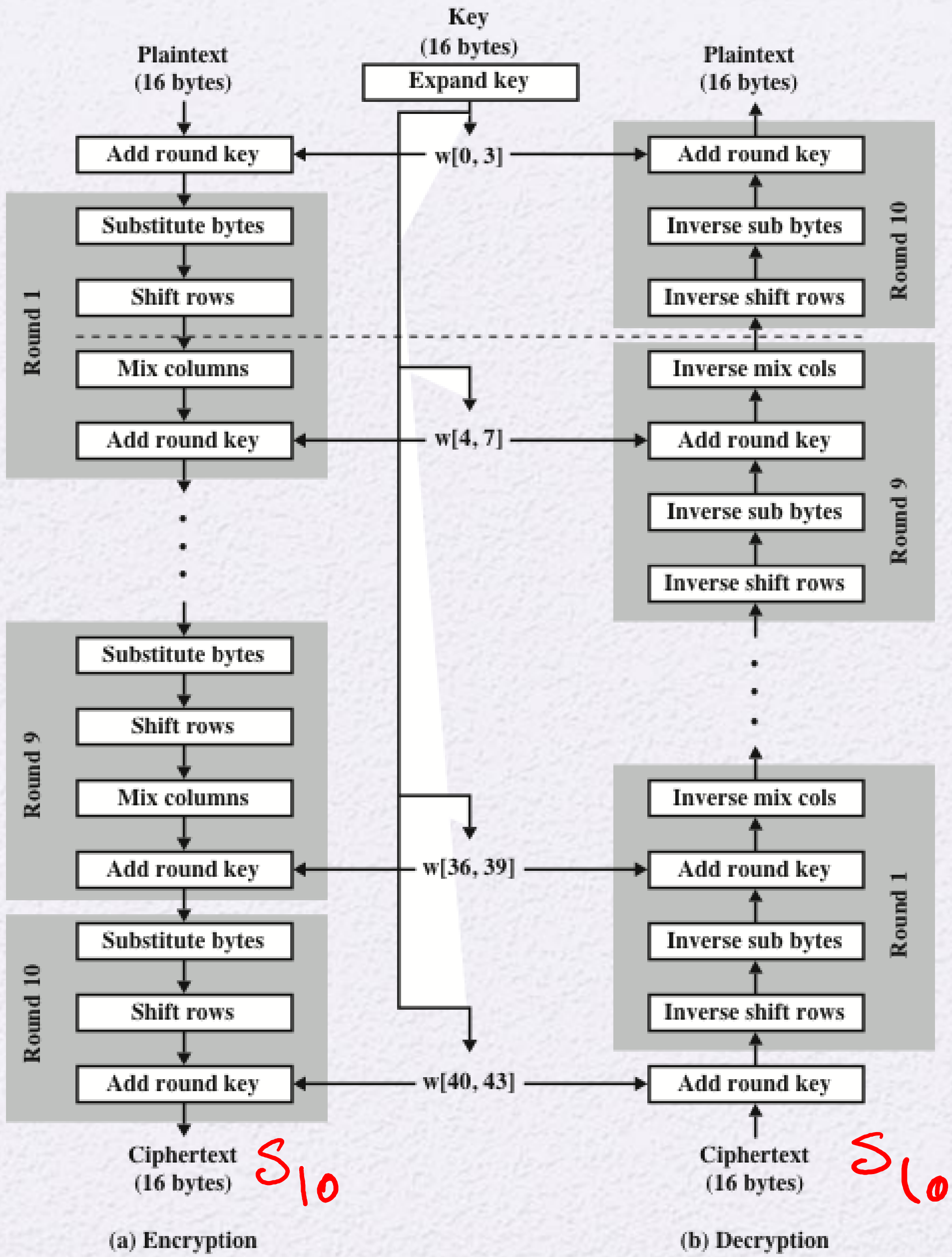


Figure 6.3 AES Encryption and Decryption

Detailed Structure

- Processes the entire data block as a single matrix during each round using substitutions and permutation
- The key that is provided as input is expanded into an array of forty-four 32-bit words, *w[i]*

Four different stages are used:

- Substitute bytes – uses an S-box to perform a byte-by-byte substitution of the block
- ShiftRows – a simple permutation
- MixColumns – a substitution that makes use of arithmetic over GF(2^8)
- AddRoundKey – a simple bitwise XOR of the current block with a portion of the expanded key

- The cipher begins and ends with an AddRoundKey stage
- Can view the cipher as alternating operations of XOR encryption (AddRoundKey) of a block, followed by scrambling of the block (the other three stages), followed by XOR encryption, and so on
- Each stage is easily reversible
- The decryption algorithm makes use of the expanded key in reverse order, however the decryption algorithm is not identical to the encryption algorithm
- State is the same for both encryption and decryption
- Final round of both encryption and decryption consists of only three stages

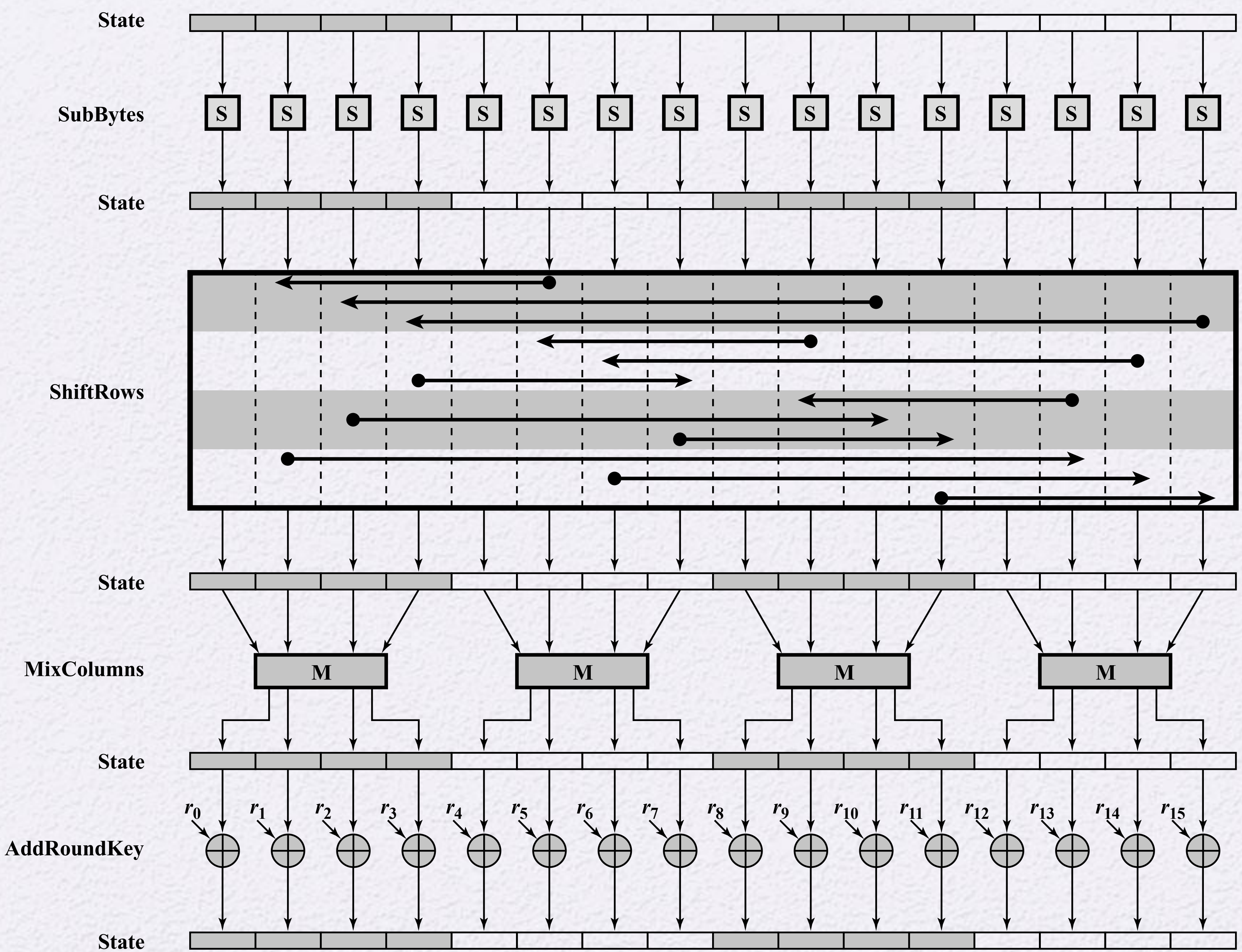
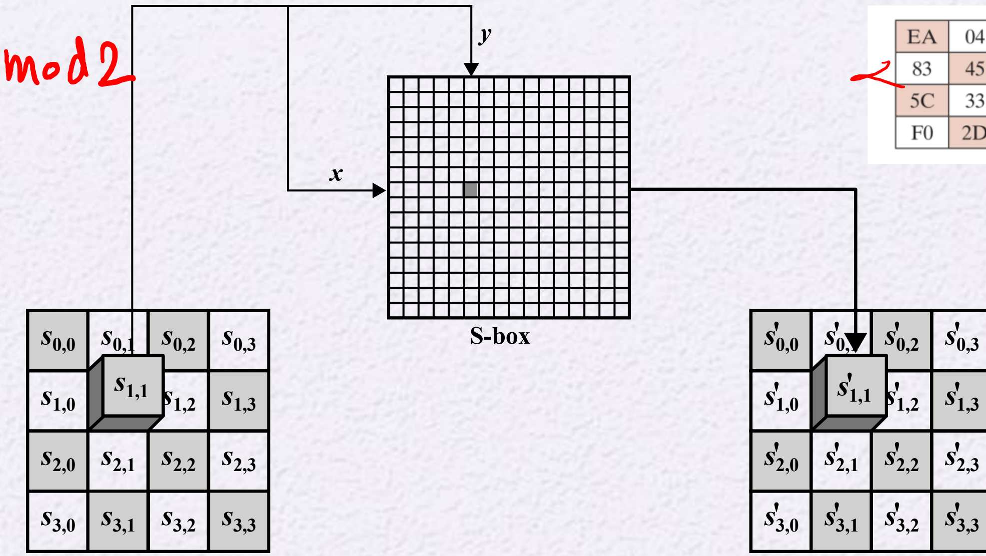


Figure 6.4 AES Encryption Round

XOR → ⊕ + mod2

EA	04	65	85		→	87	F2	4D	97
83	45	5D	96			EC	6E	4C	90
5C	33	98	B0			4A	C3	46	E7
F0	2D	AD	C5			8C	D8	95	A6



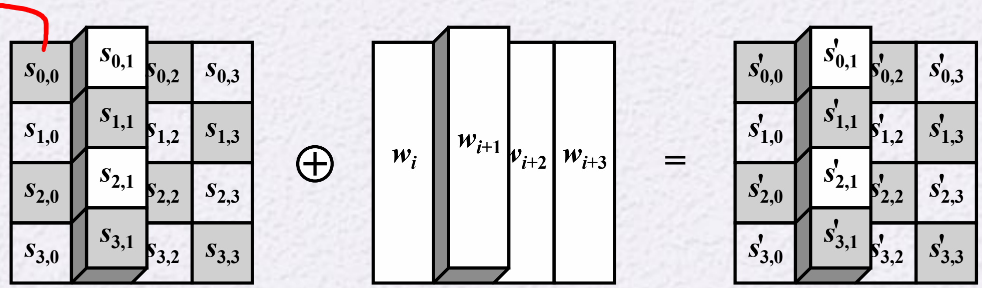
(a) Substitute byte transformation

Ex.

234   131

EA ⊕ 83 key

11101010
⊕ 10000011
01101001



(b) Add round key Transformation

Figure 6.5 AES Byte-Level Operations

Table 6.2

x \ 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
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

(a) S-box

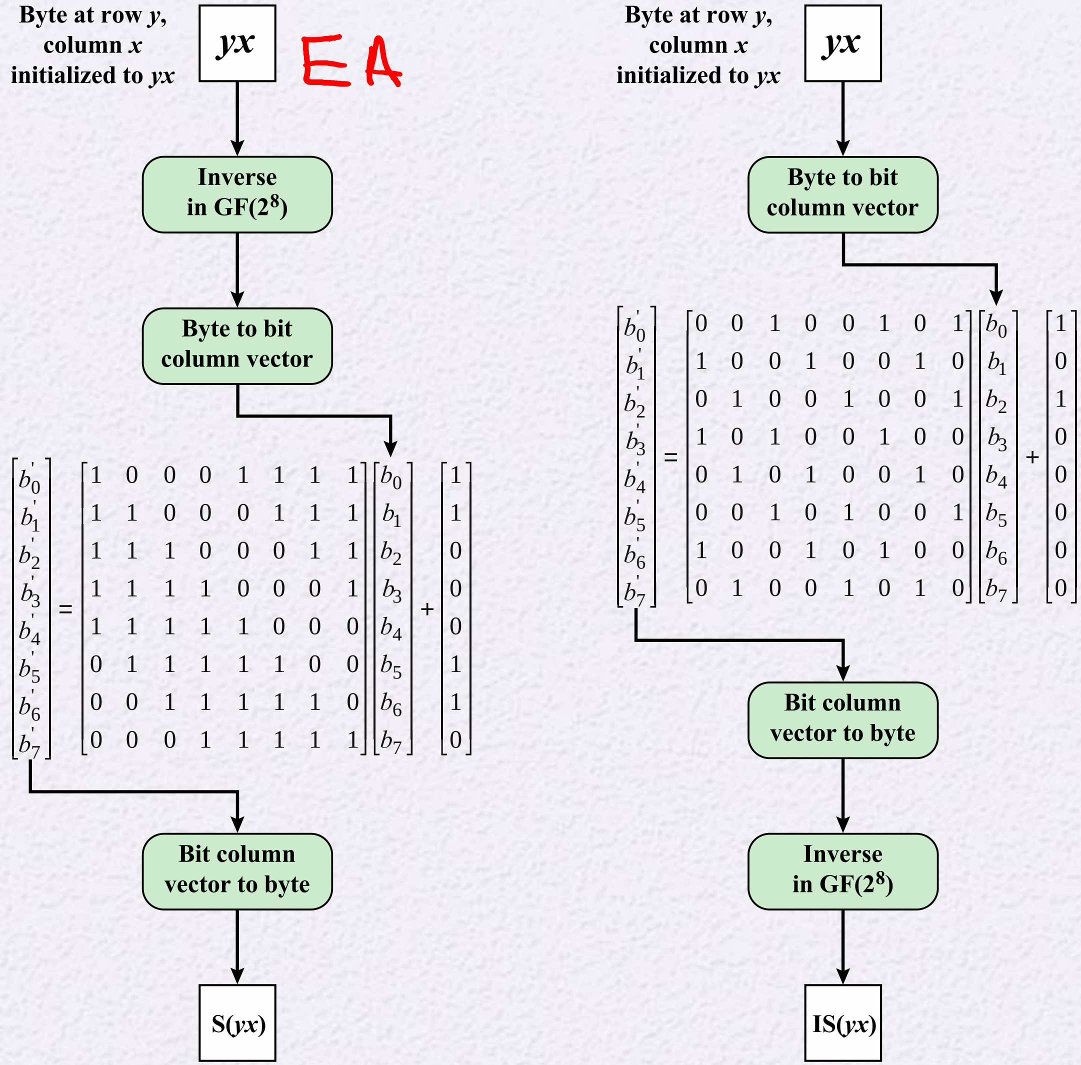
(Table can be found on page 155 in textbook)

Table 6.2

		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

(b) Inverse S-box

(Table can be found on page 155 in textbook)



$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

(a) Calculation of byte at row y, column x of S-box

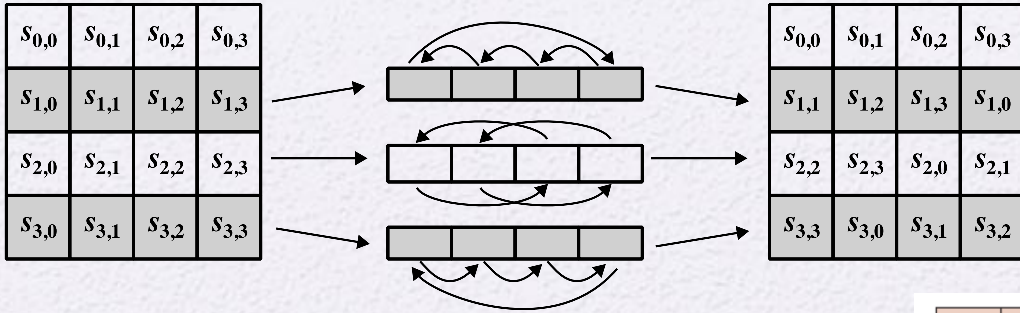
$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

(a) Calculation of byte at row y, column x of IS-box

Figure 6.6 Construction of S-Box and IS-Box

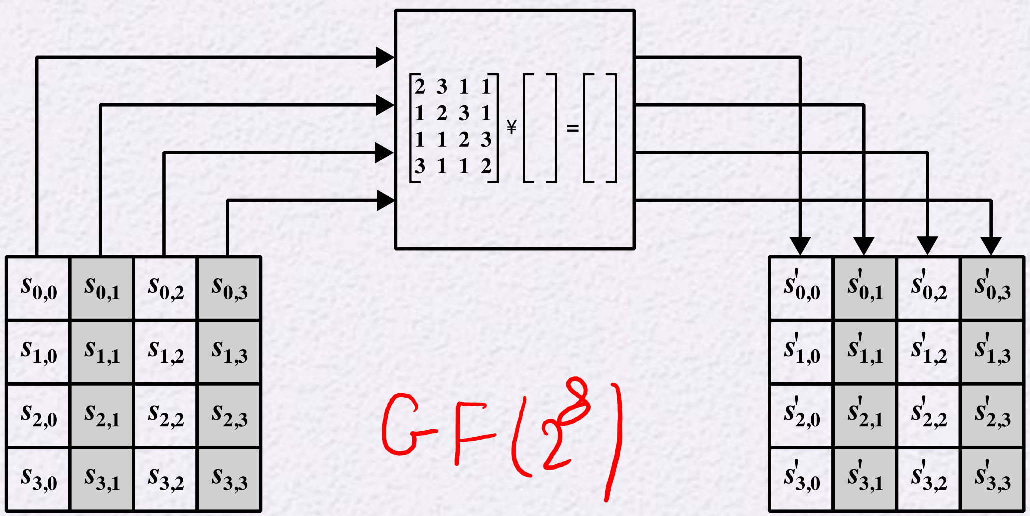
S-Box Rationale

- The S-box is designed to be resistant to known cryptanalytic attacks
- The Rijndael developers looked for a design that has a low correlation between input bits and output bits and the property that the output is not a linear mathematical function of the input
- The nonlinearity is due to the use of the multiplicative inverse



(a) Shift row transformation

87	F2	4D	97		87	F2	4D	97
EC	6E	4C	90	→	6E	4C	90	EC
4A	C3	46	E7		46	E7	4A	C3
8C	D8	95	A6		A6	8C	D8	95



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

(b) Mix column transformation

Figure 6.7 AES Row and Column Operations

Shift Row Rationale

- More substantial than it may first appear
- The State, as well as the cipher input and output, is treated as an array of four 4-byte columns
- On encryption, the first 4 bytes of the plaintext are copied to the first column of State, and so on
- The round key is applied to State column by column
  - Thus, a row shift moves an individual byte from one column to another, which is a linear distance of a multiple of 4 bytes
- Transformation ensures that the 4 bytes of one column are spread out to four different columns

Mix Columns Rationale

- Coefficients of a matrix based on a linear code with maximal distance between code words ensures a good mixing among the bytes of each column
- The mix column transformation combined with the shift row transformation ensures that after a few rounds **all output bits depend on all input bits**

AddRoundKey Transformation

- The 128 bits of State are bitwise XORed with the 128 bits of the round key
- Operation is viewed as a columnwise operation between the 4 bytes of a State column and one word of the round key
  - Can also be viewed as a byte-level operation

Rationale:

Is as simple as possible and affects every bit of State

The complexity of the round key expansion plus the complexity of the other stages of AES ensure security

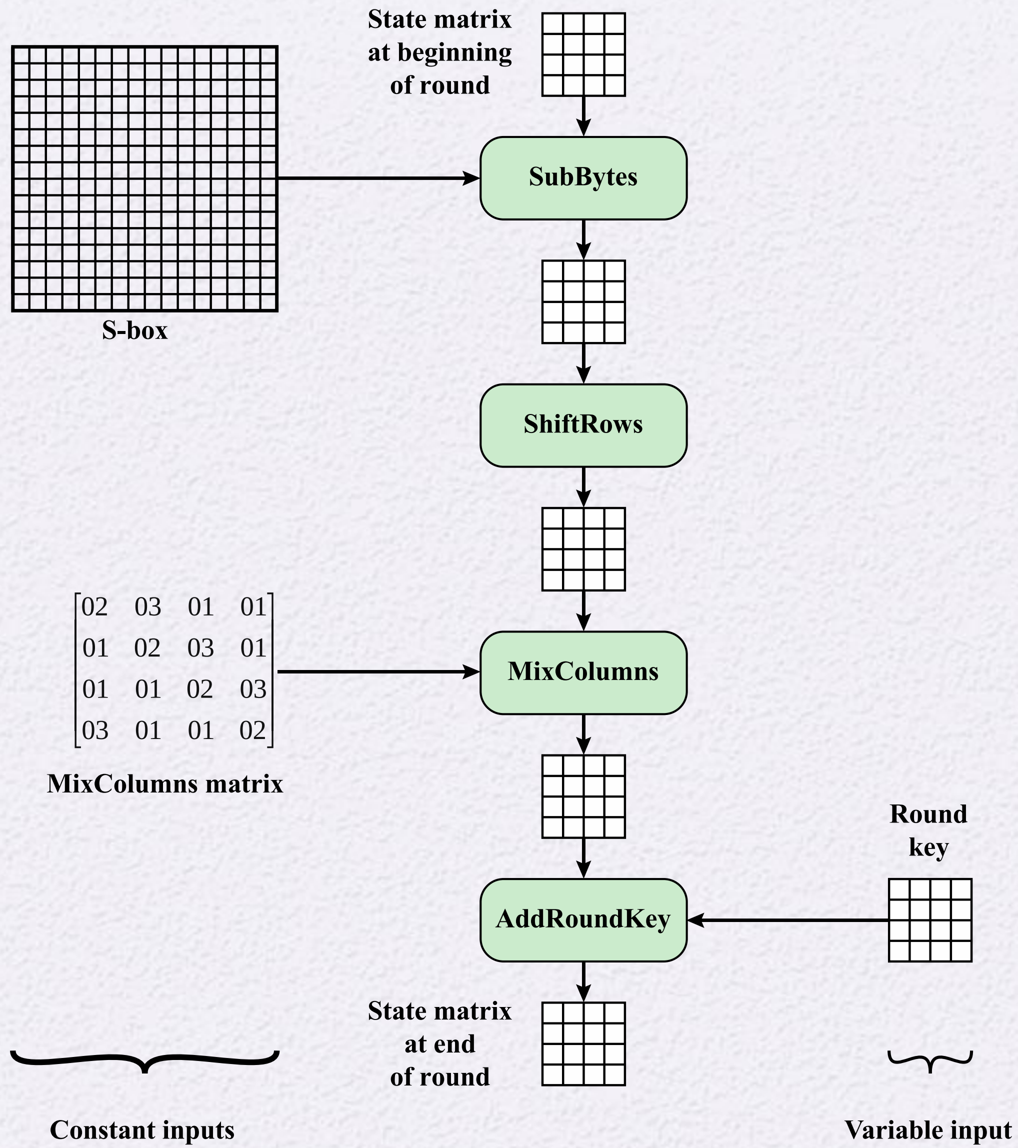


Figure 6.8 Inputs for Single AES Round

AES Key Expansion

- Takes as input a four-word (16 byte) key and produces a linear array of 44 words (176) bytes
  - This is sufficient to provide a four-word round key for the initial AddRoundKey stage and each of the 10 rounds of the cipher

- Key is copied into the first four words of the expanded key
  - The remainder of the expanded key is filled in four words at a time

- Each added word *w*[i] depends on the immediately preceding word, *w[i – 1],* and the word four positions back, w[i – 4]
  - In three out of four cases a simple XOR is used
  - For a word whose position in the *w* array is a multiple of 4~~,~~ a more complex function is used

AES Key Expansion

W_i

$W_0 - W_3$

$W_4 - W_7$

- The key expansion algorithm is as follows:

If i mod 4 = 0 then

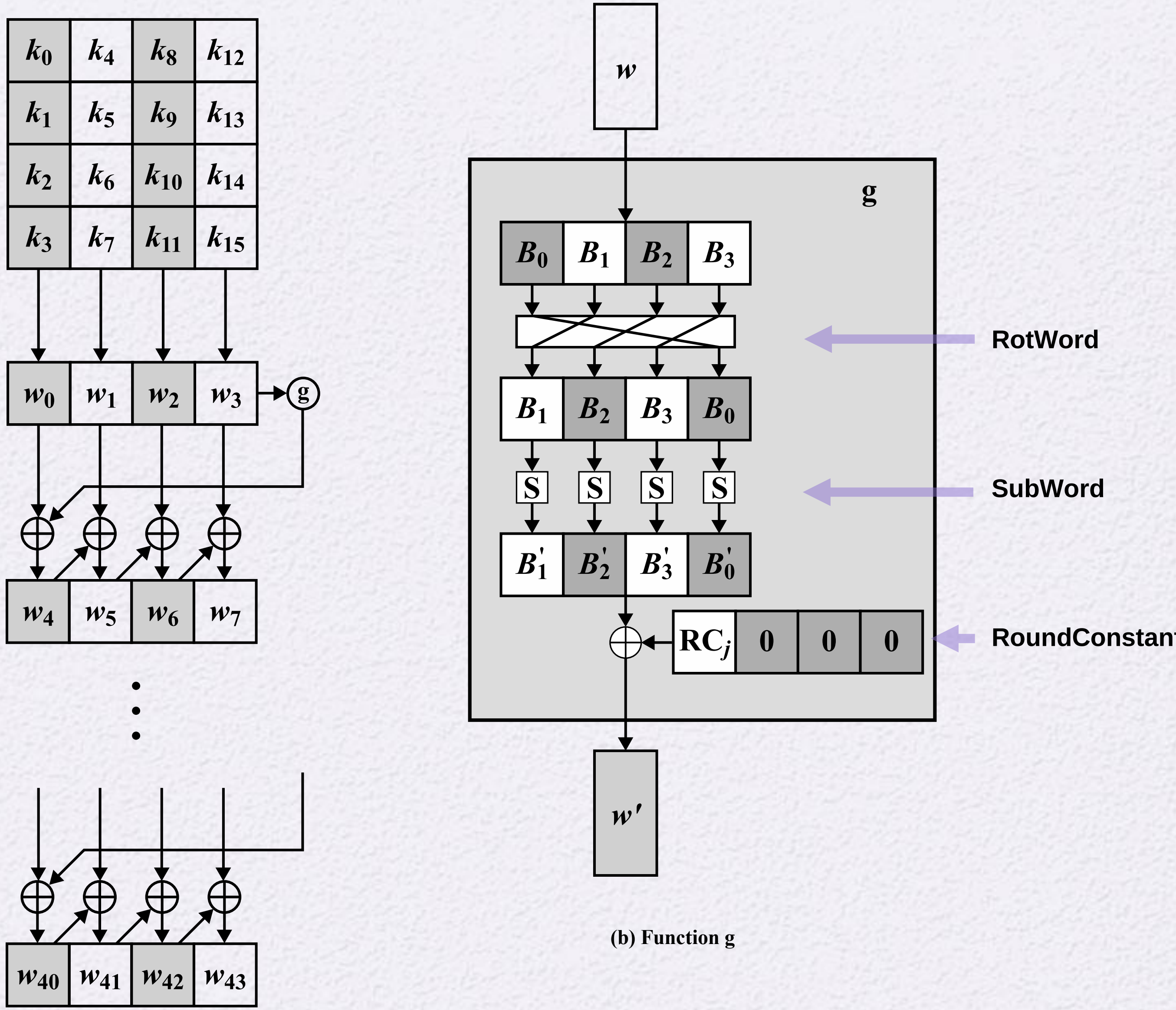
$$\boldsymbol{w[i] = g(w[i\text{-}1]) \oplus w[i\text{-}4]}$$

$W_4 = g(w_3) + W_0$

else w[i] = w[i-1] ⊕ w[i-4]

$W_5 = W_4 \oplus W_1$

- The **g** function consists of three rounds of transformation
  - RotWord – Circular byte shift
  - SubWord – S-Box substitution
  - RoundConstant –Bitwise XOR with the round constant



(b) Function g

(a) Overall algorithm

Figure 6.9 AES Key Expansion

AES Key Expanded Key Calculation

	#Rounds*	Words per Round	Size of Expanded Key Array (Words)	Size of Initial Key (Words)	Expanded Key Values Derived from Initial Key	Expanded Key Values Calculated
AES-128	10 + 1	4	44	4	w[0-3]	w[4-43]
AES-192	12 + 1	4	52	6	w[0-5]	w[6-51]
AES-256	14 + 1	4	60	8	w[0-7]	w[8-59]

* The extra added round is for the initial transformation

Key Expansion Rationale

- The Rijndael developers designed the expansion key algorithm to be resistant to known cryptanalytic attacks
- Inclusion of a round-dependent round constant eliminates the symmetry between the ways in which round keys are generated in different rounds

The specific criteria that were used are:

- Knowledge of a part of the cipher key or round key does not enable calculation of many other round-key bits
- An invertible transformation
- Speed on a wide range of processors
- Usage of round constants to eliminate symmetries
- Diffusion of cipher key differences into the round keys
- Enough nonlinearity to prohibit the full determination of round key differences from cipher key differences only
- Simplicity of description

$w_{36} = w_{32} + g(w_{35})$

0 3
4 7

Table 6.3 Example Round Key Calculation

Description	Value
i (decimal)	36
temp = w[i - 1]	7F8D292F
RotWord (temp)	8D292F7F
SubWord (RotWord (temp))	5DA515D2
Rcon (9)	1B000000
SubWord (RotWord (temp)) ⊕ Rcon (9)	46A515D2
w[i – 4]	EAD27321
w[i] = w[i – 4] ⊕ SubWord (RotWord (temp)) ⊕ Rcon (9)	AC7766F3

(Table is located on page 166 in the textbook)

Table 6.4

Key Expansion

for

AES Example

Key Words	Auxiliary Function
w0 = 0f 15 71 c9 w1 = 47 d9 e8 59 w2 = 0c b7 ad d6 w3 = af 7f 67 98	RotWord(w3)= 7f 67 98 af = x1 SubWord(x1)= d2 85 46 79 = y1 Rcon(1)= 01 00 00 00 y1 ⊕ Rcon(1)= d3 85 46 79 = z1
w4 = w0 ⊕ z1 = dc 90 37 b0 w5 = w4 ⊕ w1 = 9b 49 df e9 w6 = w5 ⊕ w2 = 97 fe 72 3f w7 = w6 ⊕ w3 = 38 81 15 a7	RotWord(w7)= 81 15 a7 38 = x2 SubWord(x4)= 0c 59 5c 07 = y2 Rcon(2)= 02 00 00 00 y2 ⊕ Rcon(2)= 0e 59 5c 07 = z2
w8 = w4 ⊕ z2 = d2 c9 6b b7 w9 = w8 ⊕ w5 = 49 80 b4 5e w10 = w9 ⊕ w6 = de 7e c6 61 w11 = w10 ⊕ w7 = e6 ff d3 c6	RotWord(w11)= ff d3 c6 e6 = x3 SubWord(x2)= 16 66 b4 8e = y3 Rcon(3)= 04 00 00 00 y3 ⊕ Rcon(3)= 12 66 b4 8e = z3
w12 = w8 ⊕ z3 = c0 af df 39 w13 = w12 ⊕ w9 = 89 2f 6b 67 w14 = w13 ⊕ w10 = 57 51 ad 06 w15 = w14 ⊕ w11 = b1 ae 7e c0	RotWord(w15)= ae 7e c0 b1 = x4 SubWord(x3)= e4 f3 ba c8 = y4 Rcon(4)= 08 00 00 00 y4 ⊕ Rcon(4)= ec f3 ba c8 = 4
w16 = w12 ⊕ z4 = 2c 5c 65 f1 w17 = w16 ⊕ w13 = a5 73 0e 96 w18 = w17 ⊕ w14 = f2 22 a3 90 w19 = w18 ⊕ w15 = 43 8c dd 50	RotWord(w19)= 8c dd 50 43 = x5 SubWord(x4)= 64 c1 53 1a = y5 Rcon(5)= 10 00 00 00 y5 ⊕ Rcon(5)= 74 c1 53 1a = z5
w20 = w16 ⊕ z5 = 58 9d 36 eb w21 = w20 ⊕ w17 = fd ee 38 7d w22 = w21 ⊕ w18 = 0f cc 9b ed w23 = w22 ⊕ w19 = 4c 40 46 bd	RotWord(w23)= 40 46 bd 4c = x6 SubWord(x5)= 09 5a 7a 29 = y6 Rcon(6)= 20 00 00 00 y6 ⊕ Rcon(6)= 29 5a 7a 29 = z6
w24 = w20 ⊕ z6 = 71 c7 4c c2 w25 = w24 ⊕ w21 = 8c 29 74 bf w26 = w25 ⊕ w22 = 83 e5 ef 52 w27 = w26 ⊕ w23 = cf a5 a9 ef	RotWord(w27)= a5 a9 ef cf = x7 SubWord(x6)= 06 d3 df 8a = y7 Rcon(7)= 40 00 00 00 y7 ⊕ Rcon(7)= 46 d3 df 8a = z7
w28 = w24 ⊕ z7 = 37 14 93 48 w29 = w28 ⊕ w25 = bb 3d e7 f7 w30 = w29 ⊕ w26 = 38 d8 08 a5 w31 = w30 ⊕ w27 = f7 7d a1 4a	RotWord(w31)= 7d a1 4a f7 = x8 SubWord(x7)= ff 32 d6 68 = y8 Rcon(8)= 80 00 00 00 y8 ⊕ Rcon(8)= 7f 32 d6 68 = z8
w32 = w28 ⊕ z8 = 48 26 45 20 w33 = w32 ⊕ w29 = f3 1b a2 d7 w34 = w33 ⊕ w30 = cb c3 aa 72 w35 = w34 ⊕ w32 = 3c be 0b 38	RotWord(w35)= be 0b 38 3c = x9 SubWord(x8)= ae 2b 07 eb = y9 Rcon(9)= 1B 00 00 00 y9 ⊕ Rcon(9)= b5 2b 07 eb = z9
w36 = w32 ⊕ z9 = fd 0d 42 cb w37 = w36 ⊕ w33 = 0e 16 e0 1c w38 = w37 ⊕ w34 = c5 d5 4a 6e w39 = w38 ⊕ w35 = f9 6b 41 56	RotWord(w39)= 6b 41 56 f9 = x10 SubWord(x9)= 7f 83 b1 99 = y10 Rcon(10)= 36 00 00 00 y10 ⊕ Rcon(10)= 49 83 b1 99 = z10
w40 = w36 ⊕ z10 = b4 8e f3 52 w41 = w40 ⊕ w37 = ba 98 13 4e w42 = w41 ⊕ w38 = 7f 4d 59 20 w43 = w42 ⊕ w39 = 86 26 18 76	

(Table is located on page 167-168 in the textbook)

Table 6.5

AES EXAMPLE

(Table is located on page 169 in textbook)

Round		Number of Bits that Differ
	0123456789abcdeffedcba9876543210 0023456789abcdeffedcba9876543210	1
0	0e3634aece7225b6f26b174ed92b5588 0f3634aece7225b6f26b174ed92b5588	1
1	657470750fc7ff3fc0e8e8ca4dd02a9c c4a9ad090fc7ff3fc0e8e8ca4dd02a9c	20
2	5c7bb49a6b72349b05a2317ff46d1294 fe2ae569f7ee8bb8c1f5a2bb37ef53d5	58
3	7115262448dc747e5cdac7227da9bd9c ec093dfb7c45343d689017507d485e62	59
4	f867aee8b437a5210c24c1974cffeabc 43efdb697244df808e8d9364ee0ae6f5	61
5	721eb200ba06206dcbd4bce704fa654e 7b28a5d5ed643287e006c099bb375302	68
6	0ad9d85689f9f77bc1c5f71185e5fb14 3bc2d8b6798d8ac4fe36a1d891ac181a	64
7	db18a8ffa16d30d5f88b08d777ba4eaa 9fb8b5452023c70280e5c4bb9e555a4b	67
8	f91b4fbfe934c9bf8f2f85812b084989 20264e1126b219aef7feb3f9b2d6de40	65
9	cca104a13e678500ff59025f3bafaa34 b56a0341b2290ba7dfdfbddcd8578205	61
10	ff0b844a0853bf7c6934ab4364148fb9 612b89398d0600cde116227ce72433f0	58

Table 6.6

Avalanche Effect in AES: Change in Plaintext

(Table is located on page 170 in textbook)

Round		Number of Bits that Differ
	0123456789abcdeffedcba9876543210 0023456789abcdeffedcba9876543210	1
0	0e3634aece7225b6f26b174ed92b5588 0f3634aece7225b6f26b174ed92b5588	1
1	657470750fc7ff3fc0e8e8ca4dd02a9c c4a9ad090fc7ff3fc0e8e8ca4dd02a9c	20
2	5c7bb49a6b72349b05a2317ff46d1294 fe2ae569f7ee8bb8c1f5a2bb37ef53d5	58
3	7115262448dc747e5cdac7227da9bd9c ec093dfb7c45343d689017507d485e62	59
4	f867aee8b437a5210c24c1974cffeabc 43efdb697244df808e8d9364ee0ae6f5	61
5	721eb200ba06206dcbd4bce704fa654e 7b28a5d5ed643287e006c099bb375302	68
6	0ad9d85689f9f77bc1c5f71185e5fb14 3bc2d8b6798d8ac4fe36a1d891ac181a	64
7	db18a8ffa16d30d5f88b08d777ba4eaa 9fb8b5452023c70280e5c4bb9e555a4b	67
8	f91b4fbfe934c9bf8f2f85812b084989 20264e1126b219aef7feb3f9b2d6de40	65
9	cca104a13e678500ff59025f3bafaa34 b56a0341b2290ba7dfdfbddcd8578205	61
10	ff0b844a0853bf7c6934ab4364148fb9 612b89398d0600cde116227ce72433f0	58

Table 6.7

Avalanche Effect in AES: Change in Key

(Table is located on page 171 in textbook)

Round		Number of Bits that Differ
	0123456789abcdeffedcba9876543210 0123456789abcdeffedcba9876543210	0
0	0e3634aece7225b6f26b174ed92b5588 0f3634aece7225b6f26b174ed92b5588	1
1	657470750fc7ff3fc0e8e8ca4dd02a9c c5a9ad090ec7ff3fc1e8e8ca4cd02a9c	22
2	5c7bb49a6b72349b05a2317ff46d1294 90905fa9563356d15f3760f3b8259985	58
3	7115262448dc747e5cdac7227da9bd9c 18aeb7aa794b3b66629448d575c7cebf	67
4	f867aee8b437a5210c24c1974cffeabc f81015f993c978a876ae017cb49e7eec	63
5	721eb200ba06206dcbd4bce704fa654e 5955c91b4e769f3cb4a94768e98d5267	81
6	0ad9d85689f9f77bc1c5f71185e5fb14 dc60a24d137662181e45b8d3726b2920	70
7	db18a8ffa16d30d5f88b08d777ba4eaa fe8343b8f88bef66cab7e977d005a03c	74
8	f91b4fbfe934c9bf8f2f85812b084989 da7dad581d1725c5b72fa0f9d9d1366a	67
9	cca104a13e678500ff59025f3bafaa34 0ccb4c66bbfd912f4b511d72996345e0	59
10	ff0b844a0853bf7c6934ab4364148fb9 fc8923ee501a7d207ab670686839996b	53

Implementation Aspects

- AES can be implemented very efficiently on an 8-bit processor
- AddRoundKey is a bytewise XOR operation
- ShiftRows is a simple byte-shifting operation
- SubBytes operates at the byte level and only requires a table of 256 bytes
- MixColumns requires matrix multiplication in the field $GF(2^8)$, which means that all operations are carried out on bytes

Implementation Aspects

- Can efficiently implement on a 32-bit processor
  - Redefine steps to use 32-bit words
  - Can precompute 4 tables of 256-words
  - Then each column in each round can be computed using 4 table lookups + 4 XORs
  - At a cost of 4Kb to store tables
- Designers believe this very efficient implementation was a key factor in its selection as the AES cipher

Summary

- Present an overview of the general structure of the Advanced Encryption Standard (AES)
- Understand the four transformations used in AES

- Explain the AES key expansion algorithm
- Understand the use of polynomials with coefficients in $GF(2^8)$