

# Data Encryption Standard (DES) & Advanced Encryption Standard (AES)

## Inhalt:

### 1. DATA ENCRYPTION STANDARD

#### 1.1 Geschichtliche Entwicklung

#### 1.2 Funktionsweise

#### 1.3 Schwächen und Stärken

#### 1.4 Anwendungsbereiche

## 1.1 Geschichtliche Entwicklung

- 1973: Ausschreibung zur Vereinheitlichung und Standardisierung eines kryptografischen Algorithmus' durch NBS
- 1974: Vorschlag von IBM (symmetrischer Blockchiffre, auf Lucifer-Algorithmus beruhend)
- 1975: Modifizierung (u.a. Schlüssellänge: 128 → 56 Bits; veränderte S-Boxen) durch NBS, IBM und NSA
- 1976 / 1977: Ernennung / Veröffentlichung des Standards

# Allgemein

- Symmetrischer Blockchiffre
- Blockweise Chiffrierung unter Verwendung des Schlüssels
- 16 Verschlüsselungsrunden mit Verwendung von Substitutionen und Permutationen
- Blocklänge des Klar- und Chiffretext: 64 Bits
- Schlüssellänge: 64 Bits, davon nur 56 Bits nutzbar, da jedes 8. Bit ein Paritätsbit (Prüfbit für Fehlererkennung) →  $2^{56}$  Schlüssel

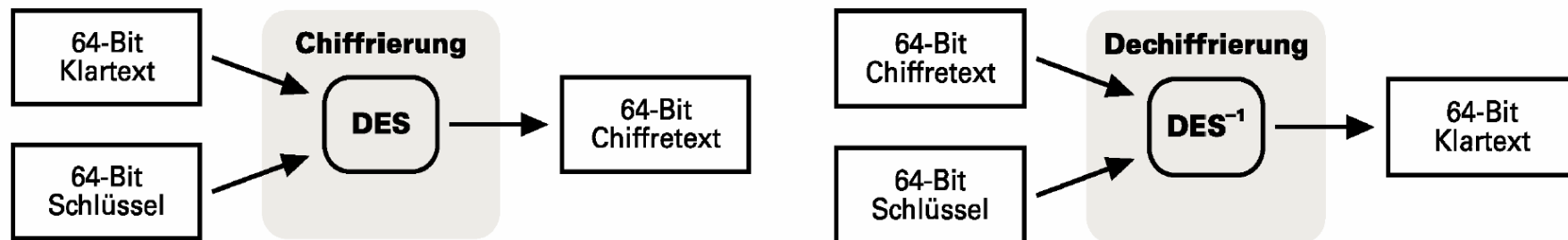


Abb. 1: Chiffrierung und Dechiffrierung beim DES

# Detaillierter Aufbau

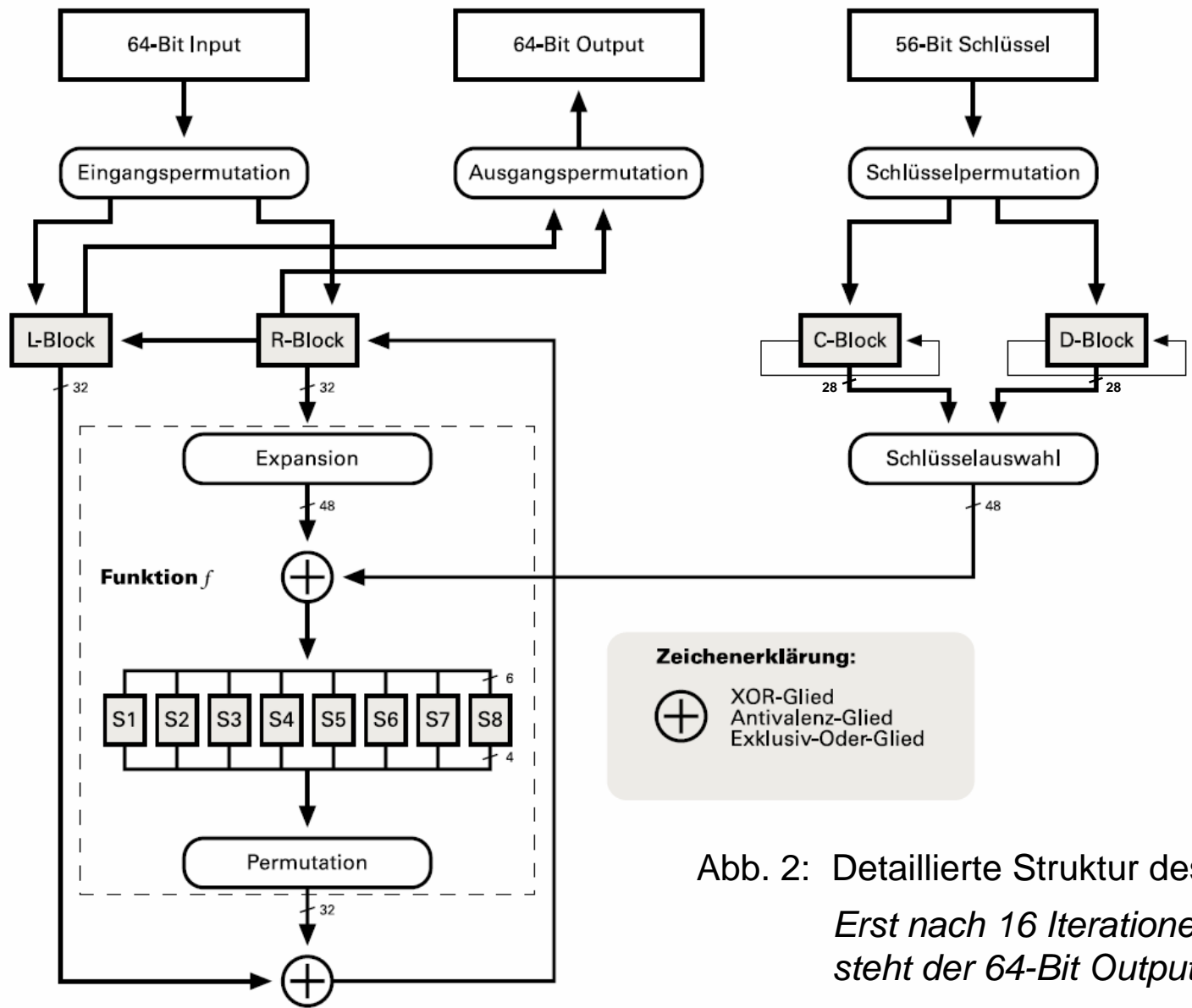


Abb. 2: Detaillierte Struktur des DES  
*Erst nach 16 Iterationen (Runden) steht der 64-Bit Output-Block fest.*

## Eingangsp permutation (*Initial Permutation IP*)

- Modifizierung der Bitpositionen des 64-Bit Eingangsblocks ( $p_1, p_2, \dots, p_{64}$ ) in ( $p_{58}, p_{50}, p_{42}, \dots, p_7$ )
- anschließend Unterteilung in zwei Teilblöcken (je 32-Bit) L ( $p_{58}, p_{50}, \dots, p_8$ ) und R ( $p_{57}, p_{49}, \dots, p_7$ )

58 50 42 34 26 18 10 2 60 52 44 36 28 20 12 4  
62 54 46 38 30 22 14 6 64 56 48 40 32 24 16 8

57 49 41 33 25 17 9 1 59 51 43 35 27 19 11 3  
61 53 45 37 29 21 13 5 63 55 47 39 31 23 15 7

Abb. 3: Bitreihenfolge nach der Eingangsp permutation im L-Block (oben) und im R-Block (unten)

# Schlüsselpermutation (*Permuted Choice 1, PC-1*)

- Unterteilung des Schlüssels in C- und D-Block (je 28 Bits)

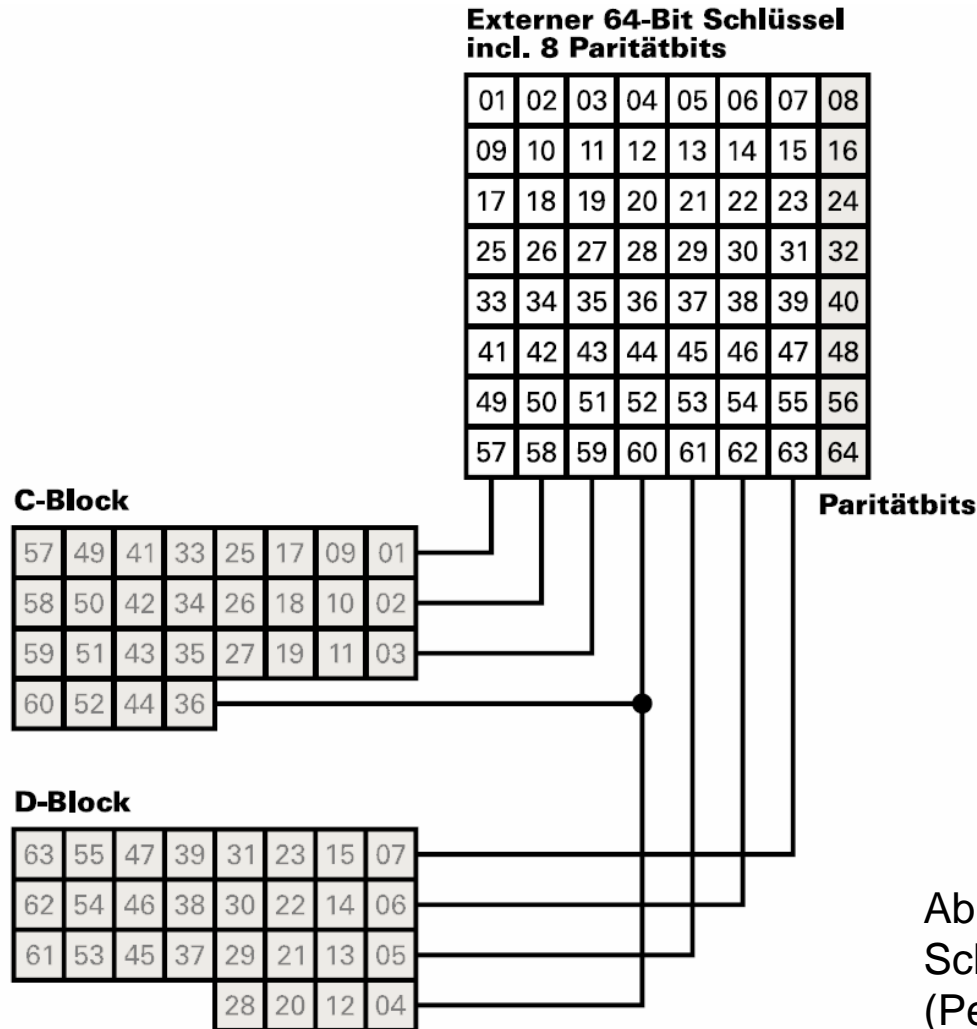


Abb. 4:  
Schlüsselpermutation  
(Permuted Choice 1)

## Bitverschiebung und Schlüsselauswahl (*Permuted Choice 2, PC-2*)

- Zyklische Linksverschiebung der Register C und D um jeweils 1 oder 2 Bit (insgesamt 28 Schiebeoperationen)

<b>Rundenindex:</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
<b>Bitanzahl:</b>	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Abb. 5: Anzahl der Bitverschiebung bei der Chiffrierung (*PC-2*)

- Reduzierung des 56-Bit-Schlüssels auf 48-Bit durch *PC-2*

# Detaillierter Aufbau

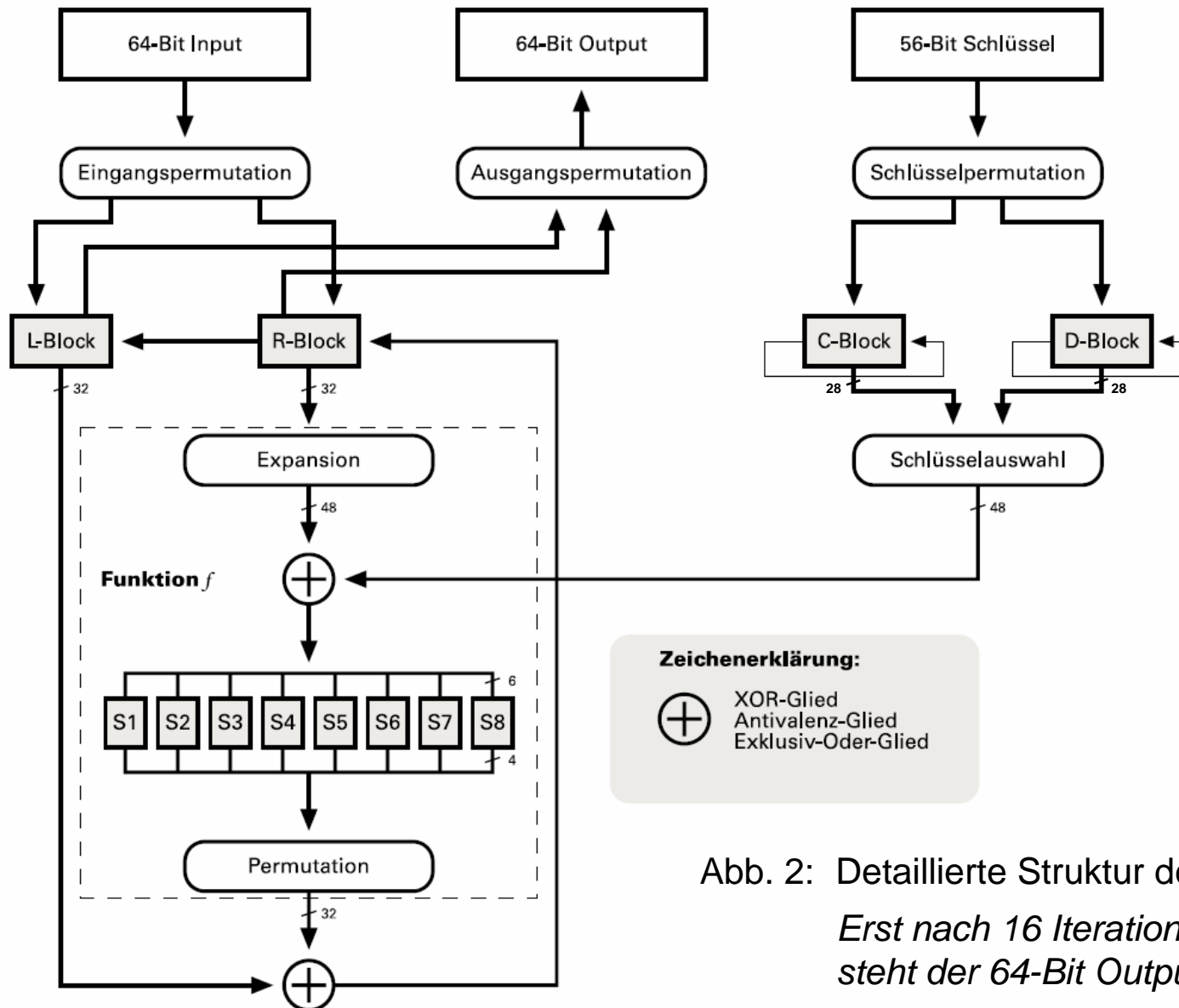


Abb. 2: Detaillierte Struktur des DES  
*Erst nach 16 Iterationen (Runden) steht der 64-Bit Output-Block fest.*

# Die Funktion f des DES

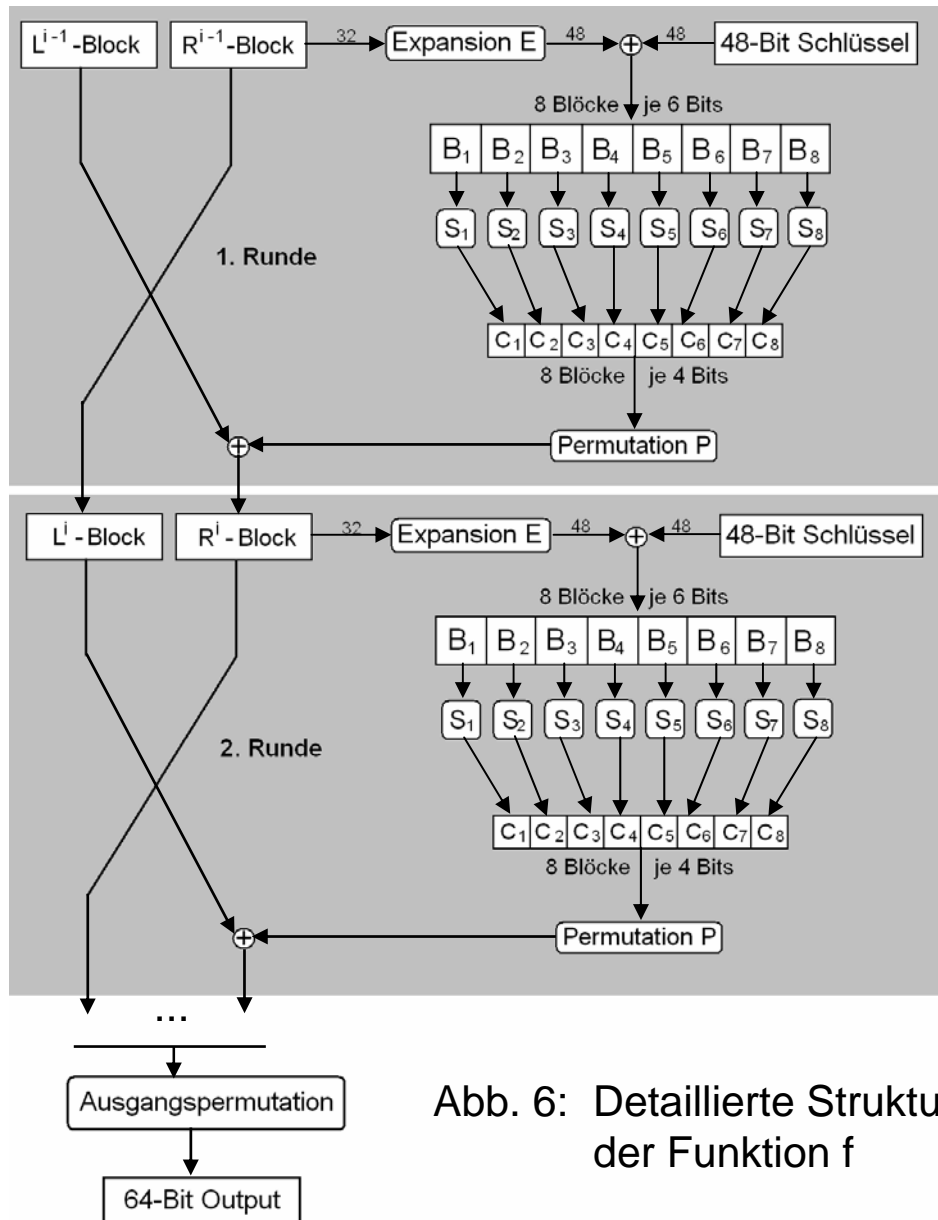


Abb. 6: Detaillierte Struktur der Funktion  $f$

1. Expansion E
  2.  $E(R) \oplus 48\text{-Bit Schlüssel}$   
Ergebnis:  $B = B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8$
  3. S-Boxen (4x16-Matrix)  
 $S_j: \{0,1\}^6 \rightarrow \{0,1\}^4$   
 $S_j(B_j) = C_j \quad (1 \leq j \leq 8)$
  4.  $C = C_1 C_2 C_3 C_4 C_5 C_6 C_7 C_8$   
(8 x 4-Bit)
- $\rightarrow P(C) = f(R \oplus 48\text{-Bit Schlüssel})$
5. Nach 16 Runden:  
64-Bit Output =  $IP^{-1}(L^{16}R^{16})$

# 1. Expansionspermutation

- 32 Bits = 8 Eingabeblock à 4 Bits
- Expansion der 32 Bits des R-Blocks auf 48 Bits (16 redundante Bits)  
→ 8 Ausgabeblöcke à 6 Bits

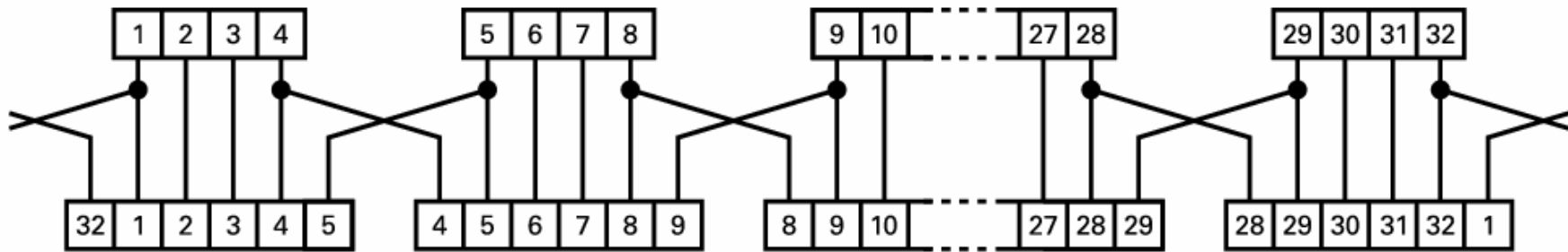


Abb. 7: Expansionpermutation

$$R = (a_1, a_2, \dots, a_{32})$$

$$E(R) = (a_{32}, a_1, a_2, a_3, a_4, a_5, a_4, a_5, a_6, \dots, a_{31}, a_{32}, a_1)$$

- anschließend:  $E(R) \oplus$  48-Bitschlüssel
- Ergebnis:  $B = B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8$  (Eingabe-Inputs für S-Boxen)

## 2. S-Boxen

- 48 Bits = 8 Eingabeblock à 6 Bits ( $B_1B_2B_3B_4B_5B_6B_7B_8$ )
- Eingabe/Ausgabe nicht linear, da 6-Bit  $\rightarrow$  4-Bit

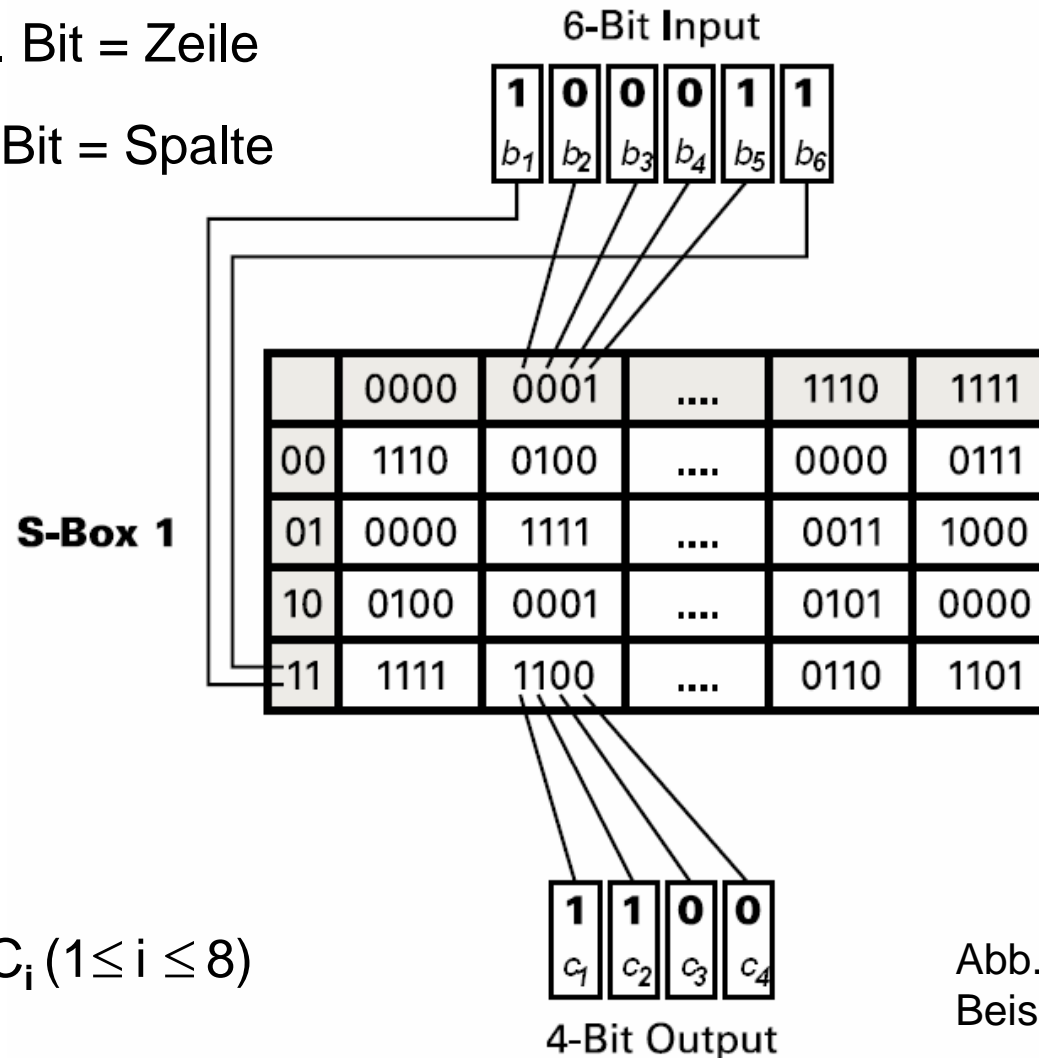
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
<b>S-Box 1</b>	<b>0</b>	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	<b>1</b>	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	<b>2</b>	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	<b>3</b>	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
<b>S-Box 2</b>	<b>0</b>	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	<b>1</b>	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	<b>2</b>	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	<b>3</b>	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
<b>S-Box 3</b>	<b>0</b>	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	<b>1</b>	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	<b>2</b>	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	<b>3</b>	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

Abb. 8:  
S-Boxen 1-3

### 3./4. Beispiel an der S-Box 1

6-Bit Input  $B_1$ :

- 1. und 6. Bit = Zeile
- 2. bis 5. Bit = Spalte



- $S_i(B_i) = C_i (1 \leq i \leq 8)$

Abb. 9:  
Beispiel an der S-Box 1

## 5. Permutation $P$ und Ausgangspermutation $IP^{-1}$

- Ergebnis der 8 S-Boxen: 32-Bit  $(p_1, p_2, a_3, \dots, p_{32})$
- Permutation  $P$ : wiederholungsfreie Abbildung von 32 Bits auf 32 Bits  
→ jedes Bit hat in der nächsten Runde einen neuen Nachbarn

16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

Abb. 10: Bitreihenfolge nach der Permutation  $P$

- Nach 16 Runden:  $L^{16}$ -Block (32-Bit) +  $R^{16}$ -Block (32-Bit)  
64-Bit Output =  $IP^{-1} (L^{16}R^{16})$

40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25

Abb. 11: Bitreihenfolge nach der Ausgangspermutation  $IP^{-1}$

# Entschlüsselung

- Umgekehrte Reihenfolge der 16 Rundenschlüssel
- Zyklische Rechtsverschiebung der Register C und D durch  $PC-2^{-1}$

<b>Rundenindex:</b>	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
<b>Bitanzahl:</b>	0	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Abb. 12: Anzahl der Bitverschiebung bei der Chiffrierung ( $PC-2^{-1}$ )

# Anwendungsbereiche

- EC-Karten (PIN-Generierung), PAY-TV,...

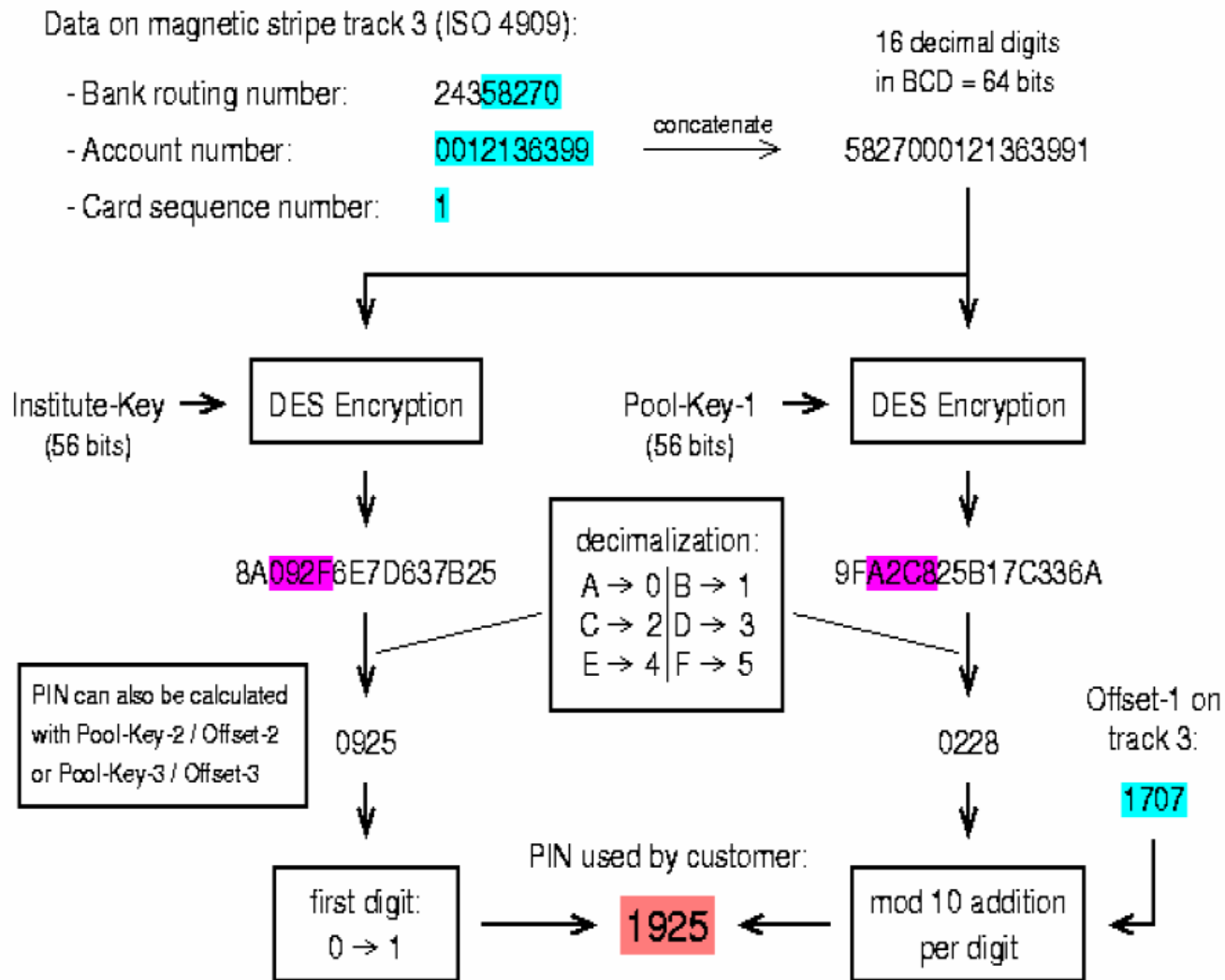


Abb. 13:  
PIN-Generierung

## Vorteile vs. Nachteile

- Vorteile:
    - Geschwindigkeit: schnelle Verarbeitung großer Datenmengen aufgrund von XOR
    - Entschlüsselung: gleicher Aufbau, nur umgekehrte Reihenfolge
  - Nachteile:
    - Schlüssellänge
    - schwache Schlüssel und Rundenschlüssel (*weak keys*)
      - *echt schwache Schlüssel*: nur Nullen/Einsen
      - *6 halbschwache Schlüssel*, so dass gilt:  
$$\text{DES}(\text{DES}(X, K1), K2) = X \quad (K1, K2: \text{Schlüssel}, X: \text{Klartext})$$
      - *48 possibly weak keys*: nehmen 4 verschiedene Werte in 16 Runden an
- 64 schwache Schlüssel von  $2^{56}$  Schlüssel
- Brute-Force-Angriff

## Inhalt:

### 2. ADVANCED ENCRYPTION STANDARD

#### 2.1 Auswahlverfahren

#### 2.2 Arbeitsweise

## Designkriterien

- Symmetrischer Blockalgorithmus  
(Blocklänge, Schlüssellänge: 128, 192 und 256 Bits)
- Sicherheit (Resistenz gegen alle bisher bekannten Attacken)
- Performance und Kosten (schnell, leicht implementierbar)
- Patentfrei und weltweit kostenlos nutzbar
- Spezifische Besonderheiten (flexibel, verschiedene Plattformen  
(z.B. 8-Bit-Prozessoren auf Smart Cards), anpassbar, einfach)

## Zeitlicher Ablauf (1)

### **National Institute of Standards and Technology (NIST):**

- Vorrunde (01/1997 – 07/1998):
  - Ankündigung der Entwicklung eines neuen Verschlüsselungsstandard (benannt als AES)
- Runde 1 (08/1998 – 04/1999):
  - Vorstellung der 15 Kandidaten, der bisherigen Analysen und Vorschlägen
- Runde 2 (08/1999 – 05/2000):
  - 5 Finalisten (MARS, RC6, Rijndael, Serpent, Twofish)
  - Aufruf zur Abgabe von Kommentaren
  - Vorstellung der technischen Analyse

## Zeitlicher Ablauf (2)

- 02. Oktober 2000: Rijndael als Sieger
  - Überlegenheit in der Kombination von Sicherheitsaspekten mit Eigenschaften der Performanz, der Effizienz, der Implementierbarkeit und der Flexibilität
  - Überdurchschnittlich schnell als Hardware- und Softwareimplementierung



Abb. 14:  
Vincent Rijmen  
und Joan Daemen

- 26. November 2001: Verabschiedung des AES' (FIPS PUB 197)

# Allgemein

- Blocklänge, Schlüssellänge: 128, 192, 256 Bits

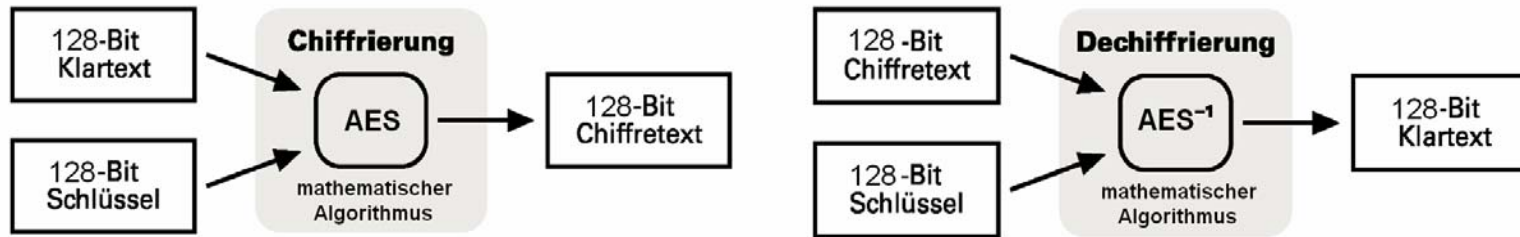


Abb. 15: Chiffrierung und Dechiffrierung beim AES

- Anzahl der Runden von Schlüssellänge (k) und Blockgröße (b) anhängig

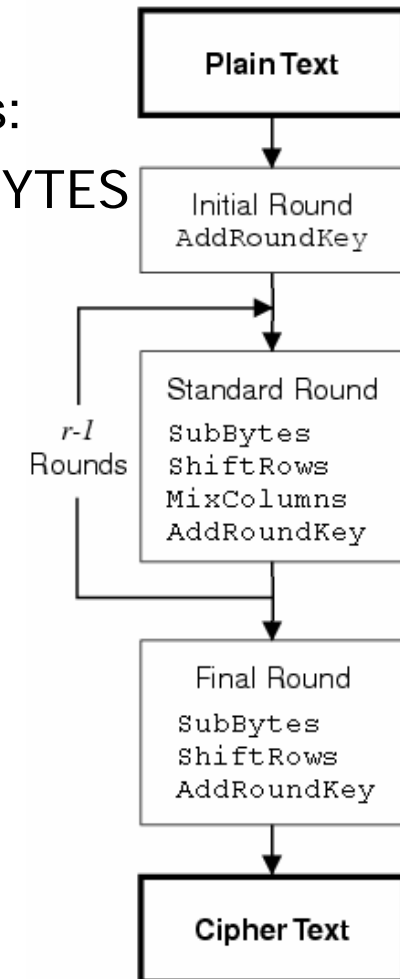
Runden R	b = 128	b = 192	b = 256
k = 128	10	12	14
k = 192	12	12	14
k = 256	14	14	14

Abb. 16: Anzahl der Runden in Abhängigkeit von der Schlüssellänge k und der Blockgröße b

## Allgemeiner Aufbau

Führe **KEYEXPANSION** aus.

- I. Sei ein Klartext  $x$  gegeben. Initialisiere den Zustand als State und führe **ADDRoundKey** aus ( $\text{Roundkey} \oplus \text{State}$ ).
- II. Runde 1 bis  $R-1$ , führe folgendes auf dem Zustand aus:
  - (1) mithilfe der S-Box die Substitutionsoperation **SUBBYTES**
  - (2) eine Permutation **SHIFTRows**
  - (3) eine Operation **MIXColumns**
  - (4) **ADDRoundKey**
- III. Runde  $N$ , führe folgendes auf dem Zustand aus:
  - (1) **SUBBYTES**
  - (2) **SHIFTRows**
  - (3) **ADDRoundKey**
- IV. Definiere den so erhaltenen Zustand als Chiffretext  $y$ .



# I: AddRoundKey

- Sei ein Klartext  $X$  gegeben. Initialisiere den Zustand als State und führe **ADDRoundKey** aus ( $\text{Roundkey} \oplus \text{State}$ ).

Klartext  $X = X_0, X_1, \dots, X_{15}$       State = 4 x 4 Bytes Matrix ( $B_{[j,i]}$ )

$$S_{0,0} = X_0, S_{1,0} = X_1, \dots, S_{2,3} = X_{14}, S_{3,3} = X_{15},$$

$$\begin{array}{c}
 \mathbf{S}_{[j,i]} \\
 \left( \begin{array}{cccc}
 \mathbf{s}_{0,0} & \mathbf{s}_{0,1} & \mathbf{s}_{0,2} & \mathbf{s}_{0,3} \\
 \mathbf{s}_{1,0} & \mathbf{s}_{1,1} & \mathbf{s}_{1,2} & \mathbf{s}_{1,3} \\
 \mathbf{s}_{2,0} & \mathbf{s}_{2,1} & \mathbf{s}_{2,2} & \mathbf{s}_{2,3} \\
 \mathbf{s}_{3,0} & \mathbf{s}_{3,1} & \mathbf{s}_{2,3} & \mathbf{s}_{3,3}
 \end{array} \right)
 \end{array}
 \oplus
 \begin{array}{c}
 \mathbf{K}_r[j,i] \\
 \left( \begin{array}{cccc}
 \mathbf{k}_{0,0} & \mathbf{k}_{0,1} & \mathbf{k}_{0,2} & \mathbf{k}_{0,3} \\
 \mathbf{k}_{1,0} & \mathbf{k}_{1,1} & \mathbf{k}_{1,2} & \mathbf{k}_{1,3} \\
 \mathbf{k}_{2,0} & \mathbf{k}_{2,1} & \mathbf{k}_{2,2} & \mathbf{k}_{2,3} \\
 \mathbf{k}_{3,0} & \mathbf{k}_{3,1} & \mathbf{k}_{2,3} & \mathbf{k}_{3,3}
 \end{array} \right)
 \end{array}
 =
 \begin{array}{c}
 \mathbf{S}_{[j,i]} \\
 \left( \begin{array}{cccc}
 \mathbf{s}_{0,0} & \mathbf{s}_{0,1} & \mathbf{s}_{0,2} & \mathbf{s}_{0,3} \\
 \mathbf{s}_{1,0} & \mathbf{s}_{1,1} & \mathbf{s}_{1,2} & \mathbf{s}_{1,3} \\
 \mathbf{s}_{2,0} & \mathbf{s}_{2,1} & \mathbf{s}_{2,2} & \mathbf{s}_{2,3} \\
 \mathbf{s}_{3,0} & \mathbf{s}_{3,1} & \mathbf{s}_{2,3} & \mathbf{s}_{3,3}
 \end{array} \right)
 \end{array}$$

$$\mathbf{S}_{[j,i]} = \mathbf{S}_{[j,i]} \oplus \mathbf{K}_r[j,i]$$

# I: AddRoundKey - Beispiel für Runde 1

$$\begin{array}{c}
 \mathbf{K}_1[j,i] \\
 \left( \begin{array}{cccc}
 \boxed{63} & \boxed{63} & 63 & 63 \\
 63 & 63 & 63 & 63 \\
 63 & 63 & 63 & 63 \\
 63 & 63 & 63 & 63
 \end{array} \right) \oplus \left( \begin{array}{cccc}
 \boxed{D6} & \boxed{D2} & DA & D6 \\
 AA & AF & A6 & AB \\
 74 & 72 & 78 & 76 \\
 FD & FA & F1 & FE
 \end{array} \right) = \left( \begin{array}{cccc}
 \boxed{B5} & \boxed{B1} & B9 & B5 \\
 C9 & CC & C5 & C8 \\
 17 & 11 & 1B & 15 \\
 9E & 99 & 92 & 9D
 \end{array} \right)
 \end{array}$$

$$S_{[j,i]} = S_{[j,i]} \oplus K_r[j,i]$$

$$63 = 01100011$$

$$\oplus D6 = 11010110$$

---


$$B5 = 10110101$$

## II: Schritt 1 (SubBytes $\Pi_S$ )

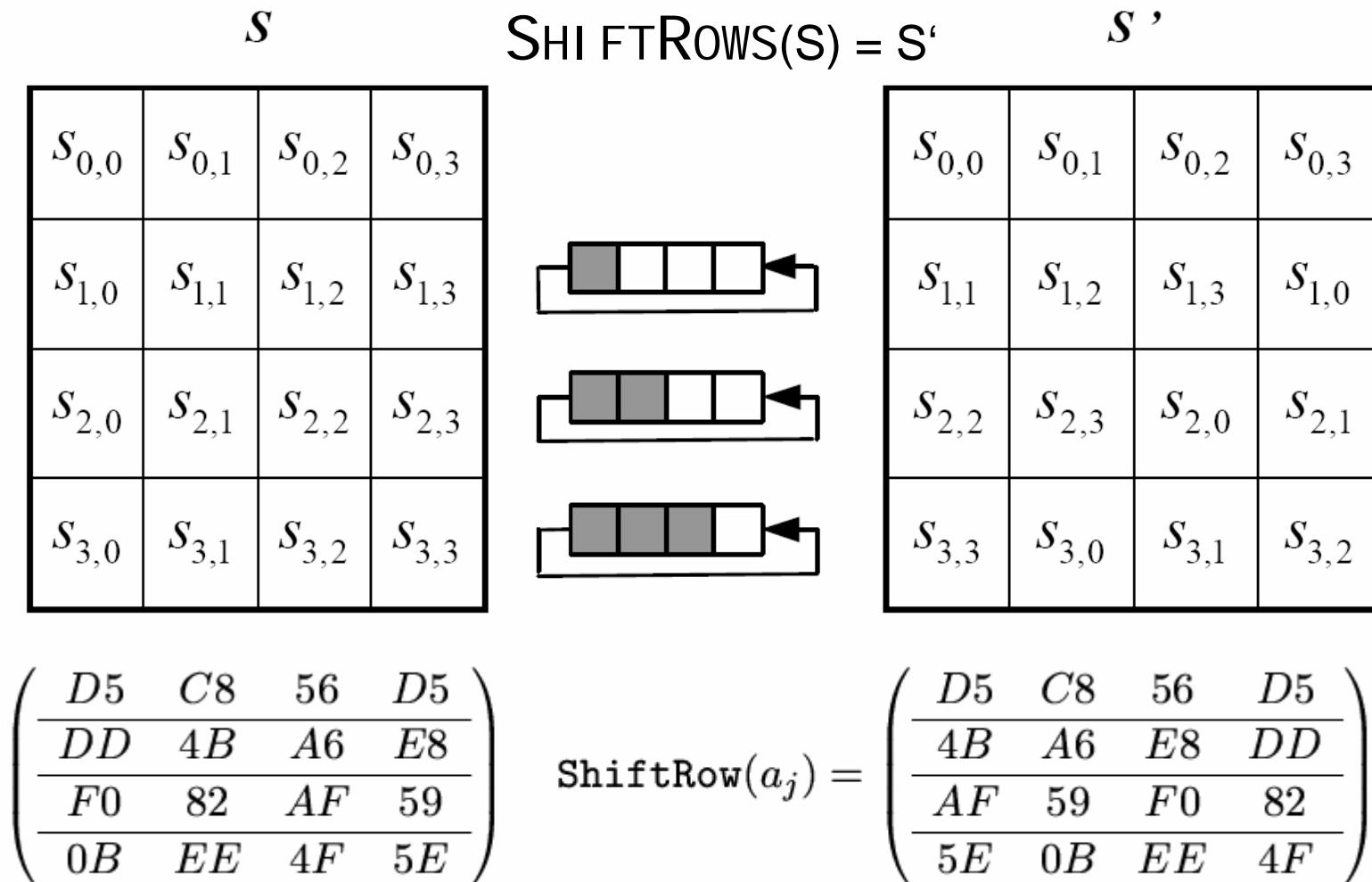
$\Pi_S : \{0,1\}^8 \rightarrow \{0,1\}^8$  (hexadezimale Notation für Bytes) anhand der S-Box  
 1 Byte = 2 aufeinanderfolgende Hexadezimalziffer (XY); X:Zeile, Y: Spalte

Beispiel:  $\Pi_S(1110\ 1101) = \Pi_S(ED) = 55 = 0101\ 0101$

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

## II: Schritt 2 (ShiftRows)

Zyklische Linksverschiebung der  $i$ -ten Zeile um  $i$  Bytes ( $1 \leq i \leq 3$ )



## II: Schritt 3 (MixColumns)

Mathematischer Hintergrund:

Körper  $\mathbf{K}(2^8)$  – 2 Polynome, Potenz 7. Grades

Byte  $a = a_7, a_6, a_5, \dots, a_0$  als Polynom:

$$a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + ax + a_0x^0$$

(1) Addition = Bitweises  $\oplus$

(2) Multiplikation = gewöhnliche Polynomprodukt unter (1)

**modulo**  $m(x) = x^8 + x^4 + x^3 + 1$  (Reduzierung)

$$(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2 \quad \text{Polynom}$$

$$\{01010111\} \oplus \{10000011\} = \{11010100\} \quad \text{Binär}$$

$$\{57\} \oplus \{83\} = \{d4\} \quad \text{Hexadezimal}$$

## II: Schritt 3 (MixColumns)

$\{57\} \bullet \{83\} = \{c1\}$ , da

$$\begin{aligned}(x^6 + x^4 + x^2 + x + 1)(x^7 + x + 1) &= x^{13} + x^{11} + x^9 + x^8 + x^7 + \\ &\quad x^7 + x^5 + x^3 + x^2 + x + \\ &\quad x^6 + x^4 + x^2 + x + 1 \\ &= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1\end{aligned}$$

$$x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \text{ modulo } (x^8 + x^4 + x^3 + x + 1)$$

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

$$= 1100\ 0001$$

$$= C\ 1$$

## II: Schritt 3 (MixColumns)

Division zweier Polynome:

$$\begin{array}{r} (x^{12} + x^9 + x + 1) \\ x^{12} + x^8 + x^7 + x^5 + x^4 \\ \hline x^9 + x^8 + x^7 + x^5 + x^4 + x + 1 \\ x^9 + x^5 + x^4 + x^2 + x \\ \hline x^8 + x^7 + x^2 + 1 \\ x^8 + x^4 + x^3 + x + 1 \\ \hline x^7 + x^4 + x^3 + x^2 + x \end{array} \quad : (x^8 + x^4 + x^3 + x + 1) = x^4 + x + 1$$

(Rest  $x^7 + x^4 + x^3 + x^2 + x$ )

## II: Schritt 3 (MixColumns)

Division zweier Polynome:

$$\begin{array}{r}
 (x^{12} + x^9 + x + 1) \\
 \underline{x^{12} + x^8 + x^7 + x^5 + x^4} \\
 x^9 + x^8 + x^7 + x^5 + x^4 + x + 1 \\
 \underline{x^9 + x^5 + x^4 + x^2 + x} \\
 x^8 + x^7 + x^2 + 1 \\
 \underline{x^8 + x^4 + x^3 + x + 1} \\
 x^7 + x^4 + x^3 + x^2 + x
 \end{array}
 \quad : \quad (x^8 + x^4 + x^3 + x + 1) = x^4 + x + 1$$

(Rest  $x^7 + x^4 + x^3 + x^2 + x$ )

Es gilt also:

$$\text{und} \quad \begin{aligned}
 (x^{12} + x^9 + x + 1) \operatorname{div} (x^8 + x^4 + x^3 + x + 1) &= x^4 + x + 1 \\
 (x^{12} + x^9 + x + 1) \operatorname{mod} (x^8 + x^4 + x^3 + x + 1) &= x^7 + x^4 + x^3 + x^2 + x.
 \end{aligned}$$

## II: Schritt 3 (MixColumns)

Multiplikation zweier Polynome 3. Grades ( $\mathbf{K}(2^8)$ ) :

Problem: Produkt ist kein 4-Byte-Vektor, d.h.  $\text{Grad}(\text{Produkt}) \leq 6$

Lösung: Reduzierung mit modulo  $x^4+1$ , d.h.  $\mathbf{x^i \text{ modulo } x^4+1 = x^{i \bmod 4}}$

$$c(x) = c_3x^3 + c_2x^2 + c_1x + \boxed{c_0} \quad a(x) = a_3x^3 + a_2x^2 + a_1x + \boxed{a_0}$$

$$\boxed{b(x) = c(x) \cdot a(x) \bmod M(x), \quad M(x) = x^4 + 1}$$

$$\boxed{b_0} = \boxed{c_0 \cdot a_0} \oplus c_3 \cdot a_1 \oplus c_2 \cdot a_2 \oplus c_1 \cdot a_3$$

$$b_1 = c_1 \cdot a_0 \oplus c_0 \cdot a_1 \oplus c_3 \cdot a_2 \oplus c_2 \cdot a_3$$

$$b_2 = c_2 \cdot a_0 \oplus c_1 \cdot a_1 \oplus c_0 \cdot a_2 \oplus c_3 \cdot a_3$$

$$b_3 = c_3 \cdot a_0 \oplus c_2 \cdot a_1 \oplus c_1 \cdot a_2 \oplus c_0 \cdot a_3$$

## II: Schritt 3 (MixColumns)

Multiplikation zweier Polynome 3. Grades ( $\mathbf{K}(2^8)$ ) :

Problem: Produkt ist kein 4-Byte-Vektor, d.h.  $\text{Grad}(\text{Produkt}) \leq 6$

Lösung: Reduzierung mit modulo  $x^4+1$ , d.h.  $\mathbf{x^i \text{ modulo } x^4+1 = x^{i \bmod 4}}$

$$c(x) = \boxed{c_3 x^3} + c_2 x^2 + c_1 x + c_0 \quad a(x) = a_3 x^3 + a_2 x^2 + \boxed{a_1 x} + a_0$$

$$\boxed{b(x) = c(x) \cdot a(x) \bmod M(x), \quad M(x) = x^4 + 1}$$

$$\boxed{b_0} = c_0 \cdot a_0 \oplus \boxed{c_3 \cdot a_1} \oplus c_2 \cdot a_2 \oplus c_1 \cdot a_3$$

$$b_1 = c_1 \cdot a_0 \oplus c_0 \cdot a_1 \oplus c_3 \cdot a_2 \oplus c_2 \cdot a_3$$

$$b_2 = c_2 \cdot a_0 \oplus c_1 \cdot a_1 \oplus c_0 \cdot a_2 \oplus c_3 \cdot a_3$$

$$b_3 = c_3 \cdot a_0 \oplus c_2 \cdot a_1 \oplus c_1 \cdot a_2 \oplus c_0 \cdot a_3$$

## II: Schritt 3 (MixColumns)

- Spalte  $A_{0,\dots,3}$  (4 Bytes) = Polynom 3. Grades über  $\mathbb{K}(2^8)$  (Hexadezimal)
- $B'_{0,\dots,3}(x) = A_{0,\dots,3}(x) \cdot c(x)$ , reduziert modulo  $x^4+1$
- MixColumns = Multiplikation der Spalte  $A_{0,\dots,3}$  mit  $c(x)$

$$c(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$$

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} c_0 & c_3 & c_2 & c_1 \\ c_1 & c_0 & c_3 & c_2 \\ c_2 & c_1 & c_0 & c_3 \\ c_3 & c_2 & c_1 & c_0 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

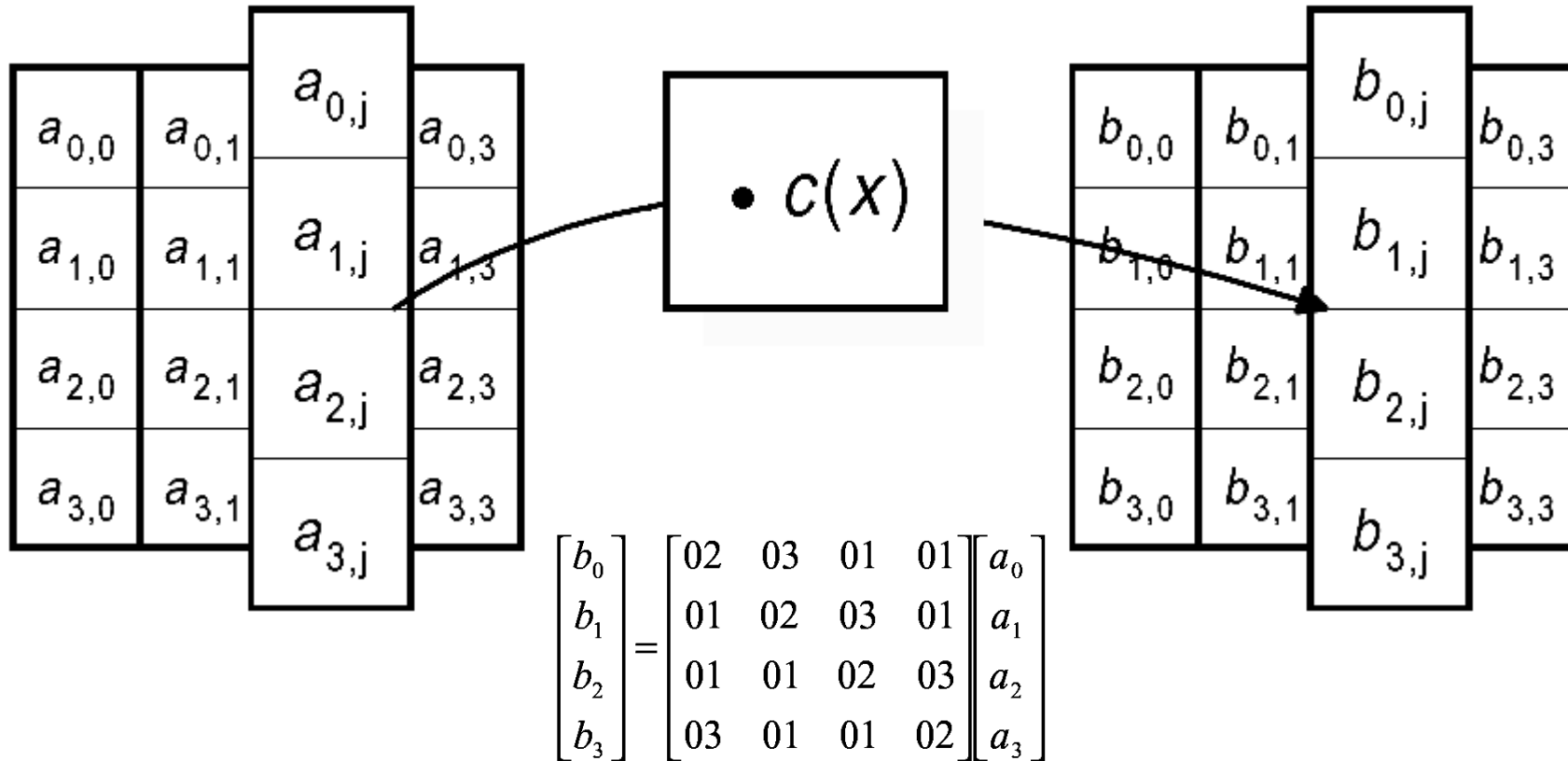
## II: Schritt 3 (MixColumns)

- Spalte  $A_{0,\dots,3}$  (4 Bytes) = Polynom 3. Grades über  $\mathbb{K}(2^8)$  (Hexadezimal)
- $B'_{0,\dots,3}(x) = A_{0,\dots,3}(x) \cdot c(x)$ , reduziert modulo  $x^4+1$
- MixColumns = Multiplikation der Spalte  $A_{0,\dots,3}$  mit  $c(x)$

$$c(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$$

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

## II: Schritt 3 (MixColumns)



$$b_0 = \{02\} \cdot a_0 \oplus \{03\} \cdot a_1 \oplus \{01\} \cdot a_2 \oplus \{01\} \cdot a_3$$

$$b_1 = \{01\} \cdot a_0 \oplus \{02\} \cdot a_1 \oplus \{03\} \cdot a_2 \oplus \{01\} \cdot a_3$$

$$b_2 = \{01\} \cdot a_0 \oplus \{01\} \cdot a_1 \oplus \{02\} \cdot a_2 \oplus \{03\} \cdot a_3$$

$$b_3 = \{03\} \cdot a_0 \oplus \{01\} \cdot a_1 \oplus \{01\} \cdot a_2 \oplus \{02\} \cdot a_3$$

## II: Schritt 3 (MixColumns)

Beispiel für MixColumns:

$$\left( \begin{array}{c|c|c|c} D5 & C8 & 56 & D5 \\ \hline 4B & A6 & E8 & DD \\ \hline AF & 59 & F0 & 82 \\ \hline 5E & 0B & EE & 4F \end{array} \right) \text{MixColumn}(a_j) = \left( \begin{array}{c|c|c|c} 9D & 28 & 91 & 00 \\ \hline F7 & 7F & 78 & A6 \\ \hline 39 & C1 & 6C & C6 \\ \hline 3C & AA & 25 & A5 \end{array} \right)$$

$$A(x) = \begin{bmatrix} D5 \\ 4B \\ AF \\ 5E \end{bmatrix} = \begin{bmatrix} 11010101 \\ 01001011 \\ 10101111 \\ 01011110 \end{bmatrix} \equiv \begin{bmatrix} x^7+x^6+x^4+x^2+1 \\ x^6+x^3+x+1 \\ x^7+x^5+x^3+x^2+x+1 \\ x^6+x^4+x^3+x^2+x \end{bmatrix} \quad c(x) = \begin{bmatrix} 10 & 11 & 1 & 1 \\ 1 & 10 & 11 & 1 \\ 1 & 1 & 10 & 11 \\ 11 & 1 & 1 & 10 \end{bmatrix} \equiv \begin{bmatrix} x & x+1 & 1 & 1 \\ 1 & x & x+1 & 1 \\ 1 & 1 & x & x+1 \\ x+1 & 1 & 1 & x \end{bmatrix}$$

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} x & x+1 & 1 & 1 \\ 1 & x & x+1 & 1 \\ 1 & 1 & x & x+1 \\ x+1 & 1 & 1 & x \end{bmatrix} \bullet \begin{bmatrix} x^7+x^6+x^4+x^2+1 \\ x^6+x^3+x+1 \\ x^7+x^5+x^3+x^2+x+1 \\ x^6+x^4+x^3+x^2+x \end{bmatrix}$$

$$\begin{aligned} b_0 &= x(x^7+x^6+x^4+x^2+1) \oplus (x+1)(x^6+x^3+x+1) \oplus (x^7+x^5+x^3+x^2+x+1) \oplus (x^6+x^4+x^3+x^2+x) \\ &= [(x^8+x^7+x^5+x^3+x) \bmod (x^8+x^4+x^3+x+1)] \oplus \dots \\ &= 9D \end{aligned}$$

# KeyExpansion

```
proc KeyExpansion(K,w);  
begin external RotWord; SubWord;  
RCon[1] := 01000000; RCon[2] := 02000000;  
RCon[3] := 04000000; RCon[4] := 08000000;  
RCon[5] := 10000000; RCon[6] := 20000000;  
RCon[7] := 40000000; RCon[8] := 80000000;  
RCon[9] := 1B000000; RCon[10] := 36000000;  
for i := 0 to 3 do  
    w[i] := (K[4i];K[4i+1];K[4i+2];K[4i+3])  
endfor;  
for i := 4 to 43 do  
    temp := w[i-1];  
    if (i mod 4=0)  
        then temp := SubWord(RotWord(temp)) XOR RCon[i/4];  
    w[i] := w[i-4] XOR temp  
endfor;  
return(w[0].....w[43])  
end
```

Abb. 16: Pseudo-Code für KeyExpansion

## KeyExpansion

- *wortorientierte* Konstruktion der Rundenschlüssel (RK) (1 Wort = 4 Bytes)
- 11 Rundenschlüssel à 4 Wörter (d.h. 16 Bytes oder 128 Bits)
- *Expanded Key* = Konkatenation der 11 Rundenschlüssel (44 Wörter)
- $RK[0] = w[0]w[1]w[2]w[3] = \mathbf{K}$  (=  $K[0] \dots K[15]$ , wobei jedes  $K[i]=1$  Byte)
- $w[i] = w[i-1] \text{ XOR } \text{temp}_Z$ , für alle  $i \in \{4, 8, 12, \dots, 40\}$ ,  $Z = i$   
 $\text{temp}_Z = \text{SubWord}(\text{RotWord}(w[i-1])) \text{ XOR } \text{Rcon}[i/4]$
- $RK[1] = \{ \begin{array}{l} w[4] = w[0] \text{ XOR } \text{temp}_4 \\ w[5] = w[1] \text{ XOR } w[4] \\ w[6] = w[2] \text{ XOR } w[5] \\ w[7] = w[3] \text{ XOR } w[6] \end{array} \}$        $RK[2] = \{ \begin{array}{l} w[8] = w[4] \text{ XOR } \text{temp}_8 \\ w[9] = w[5] \text{ XOR } w[8] \\ w[10] = w[6] \text{ XOR } w[9] \\ w[11] = w[7] \text{ XOR } w[10] \end{array} \}$   
    ...  
 $RK[10] = w[40]w[41]w[42]w[43]$

## Beispiel: KeyExpansion

- $\mathbf{K} = 00\ 01\ 02\ 03\ 04\ 05\ 06\ 07\ 08\ 09\ 0A\ 0B\ 0C\ 0D\ 0E\ 0F$
- $RK[0] =$   
 $w[0] = K[0]K[1]K[2]K[3] = 00\ 01\ 02\ 03$   
 $w[1] = K[4]K[5]K[6]K[7] = 04\ 05\ 06\ 07$   
 $w[2] = K[8]K[9]K[10]K[11] = 08\ 09\ 0A\ 0B$   
 $w[3] = K[12]K[13]K[14]K[15] = 0C\ 0D\ 0E\ 0F$

## Beispiel: KeyExpansion

- $\mathbf{K} = 00\ 01\ 02\ 03\ 04\ 05\ 06\ 07\ 08\ 09\ 0A\ 0B\ 0C\ 0D\ 0E\ 0F$
- $RK[0] =$   
 $w[0] = K[0]K[1]K[2]K[3] = 00\ 01\ 02\ 03$   
 $w[1] = K[4]K[5]K[6]K[7] = 04\ 05\ 06\ 07$   
 $w[2] = K[8]K[9]K[10]K[11] = 08\ 09\ 0A\ 0B$   
 $w[3] = K[12]K[13]K[14]K[15] = 0C\ 0D\ 0E\ 0F$
- $temp4 = \text{SubWord}(\text{RotWord}(w[3])) \text{ XOR } Rcon[1] = D6\ AB\ 76\ FE$   
 $\text{RotWord}(w[3]) = 0D\ 0E\ 0F\ 0C$   
 $(\text{SubWord}(\text{RotWord}(w[3])) = D7\ AB\ 76\ FE) \text{ XOR } 01\ 00\ 00\ 00 = D6\ AB\ 76\ FE$

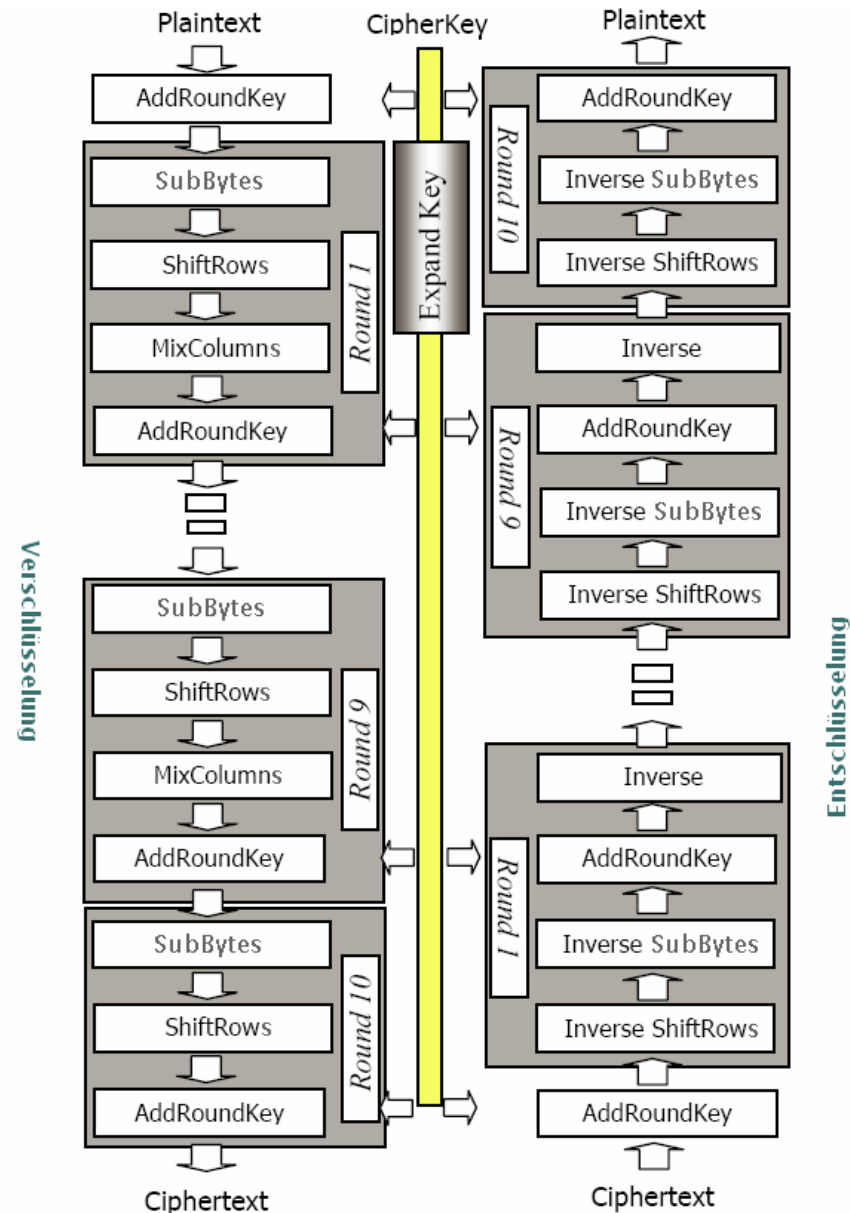
## Beispiel: KeyExpansion

- $\mathbf{K} = 00\ 01\ 02\ 03\ 04\ 05\ 06\ 07\ 08\ 09\ 0A\ 0B\ 0C\ 0D\ 0E\ 0F$
- $\text{RK}[0] =$   
 $w[0] = K[0]K[1]K[2]K[3] = 00\ 01\ 02\ 03$   
 $w[1] = K[4]K[5]K[6]K[7] = 04\ 05\ 06\ 07$   
 $w[2] = K[8]K[9]K[10]K[11] = 08\ 09\ 0A\ 0B$   
 $w[3] = K[12]K[13]K[14]K[15] = 0C\ 0D\ 0E\ 0F$
- $\text{temp4} = \text{SubWord}(\text{RotWord}(w[3])) \text{ XOR } \text{Rcon}[1] = D6\ AB\ 76\ FE$   
 $\text{RotWord}(w[3]) = 0D\ 0E\ 0F\ 0C$   
 $(\text{SubWord}(\text{RotWord}(w[3])) = D7\ AB\ 76\ FE) \text{ XOR } 01\ 00\ 00\ 00 = D6\ AB\ 76\ FE$
- $\text{RK}[1] =$   
 $w[4] = w[0] \text{ XOR } \text{temp4} = \mathbf{D6\ AA\ 74\ FD}$   
 $00 \text{ XOR } D6 = D6$   
 $01 \text{ XOR } AB = AA$   
 $02 \text{ XOR } 76 = 74$   
 $03 \text{ XOR } FE = FD$

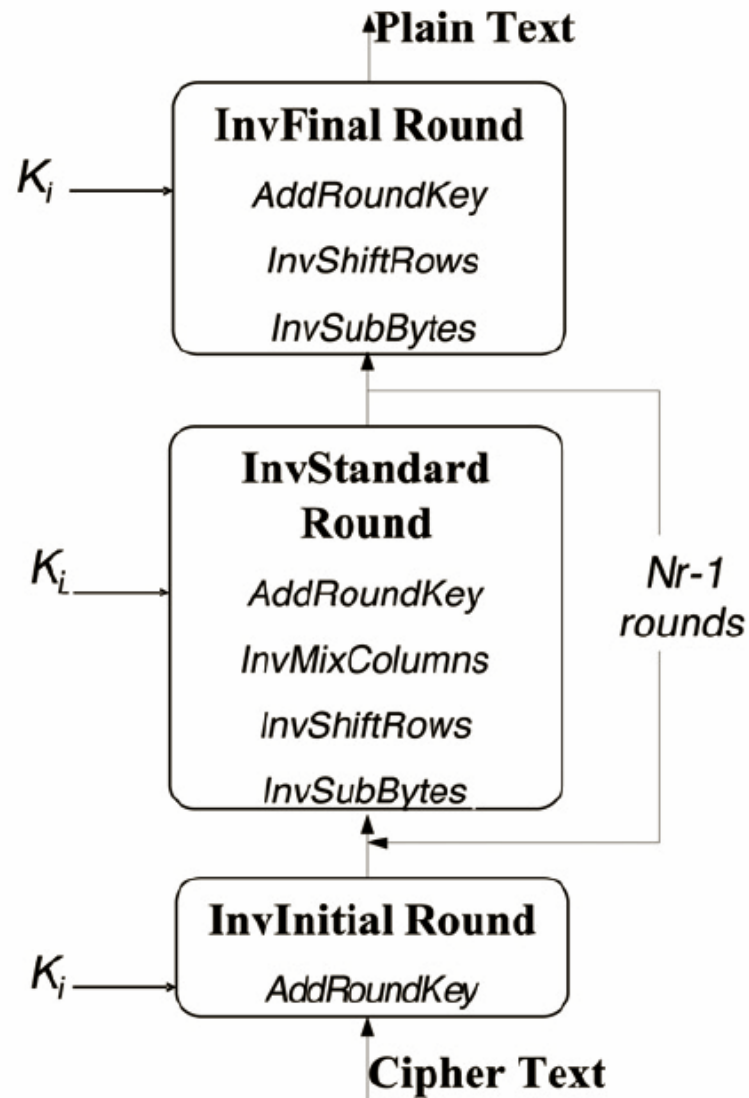
## Beispiel: KeyExpansion

- $\mathbf{K} = 00\ 01\ 02\ 03\ 04\ 05\ 06\ 07\ 08\ 09\ 0A\ 0B\ 0C\ 0D\ 0E\ 0F$
- $\text{RK}[0] =$   
 $w[0] = K[0]K[1]K[2]K[3] = 00\ 01\ 02\ 03$   
 $w[1] = K[4]K[5]K[6]K[7] = 04\ 05\ 06\ 07$   
 $w[2] = K[8]K[9]K[10]K[11] = 08\ 09\ 0A\ 0B$   
 $w[3] = K[12]K[13]K[14]K[15] = 0C\ 0D\ 0E\ 0F$
- $\text{temp4} = \text{SubWord}(\text{RotWord}(w[3])) \text{ XOR } \text{Rcon}[1] = D6\ AB\ 76\ FE$   
 $\text{RotWord}(w[3]) = 0D\ 0E\ 0F\ 0C$   
 $(\text{SubWord}(\text{RotWord}(w[3])) = D7\ AB\ 76\ FE) \text{ XOR } 01\ 00\ 00\ 00 = D6\ AB\ 76\ FE$
- $\text{RK}[1] =$   
 $w[4] = w[0] \text{ XOR } \text{temp4} = \mathbf{D6\ AA\ 74\ FD}$   
 $00 \text{ XOR } D6 = D6$   
 $01 \text{ XOR } AB = AA$   
 $02 \text{ XOR } 76 = 74$   
 $03 \text{ XOR } FE = FD$   
 $w[5] = w[1] \text{ XOR } w[4] = \mathbf{D2\ AF\ 72\ FA}$   
 $w[6] = w[2] \text{ XOR } w[5] = \mathbf{DA\ A6\ 78\ F1}$   
 $w[7] = w[3] \text{ XOR } w[6] = \mathbf{D6\ AB\ 76\ FE}$

# Zusammenfassender Überblick



# Entschlüsselung



- Benötigt inverse Operationen
- InvSubBytes: inverse S-Box

								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		
x	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	5e	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		

- InvShiftRows: zyklische Rechts- statt Linksverschiebung
- InvMixColumns: inverse Matrix

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

## 3. Zusammenfassung

### Data Encryption Standard (IBM)

- Symmetrischer Blockchiffre mit 64-Bit-Block und 56-Bit Schlüssel
- 16 Runden mit Substitution und Permutation
- Entschlüsselung: umgekehrte Reihenfolge der Rundenschlüssel
- Weitverbreitet, jedoch angreifbar durch Brute-Force-Angriffe

## 3. Zusammenfassung

### Data Encryption Standard (IBM)

- Symmetrischer Blockchiffre mit 64-Bit-Block und 56-Bit Schlüssel
- 16 Runden mit Substitution und Permutation
- Entschlüsselung: umgekehrte Reihenfolge der Rundenschlüssel
- Weitverbreitet, jedoch angreifbar durch Brute-Force-Angriffe

### Advanced Encryption Standard (Rijndael)

- Symmetrischer Blockchiffre mit 128/192/256-Bit-Block und Schlüssel
- Rundenanzahl in Abhängigkeit von Block- und Schlüsselgröße
- AddRoundkey, SubBytes, ShiftRows, MixColumns
- Entschlüsselung mit inversen Operationen
- Weltweite Anerkennung und Verwendung, da Auswahlverfahren sehr öffentlich

## 4. Quellenverzeichnis

- Douglas R. Stinson: Cryptography: Theory and Practice. 2nd Edition, Chapman & Hall/CRC 2002
- Homepage of NIST:  
<http://csrc.nist.gov/encryption/aes/>
- <http://parsys.informatik.uni-oldenburg.de/~best/kryptographie/kry4-May21.pdf>
- [http://informatik.uibk.ac.at/~c70236/2003ws/vortraege/abfalterer\\_niederbacher-des-arbeit.pdf](http://informatik.uibk.ac.at/~c70236/2003ws/vortraege/abfalterer_niederbacher-des-arbeit.pdf)
- *Die Abb. 1 – 12 (ausser Abb. 5) sind entnommen aus*  
[http://archiv.tu-chemnitz.de/pub/2002/0059/data/PS\\_Electronic\\_Banking.pdf](http://archiv.tu-chemnitz.de/pub/2002/0059/data/PS_Electronic_Banking.pdf)

# Attacken?

Nicholas Courtois:

“Das Problem, einen einzigen Schlüssel zu finden, kann als System von 8000 quadratischen Gleichungen mit 1600 Unbekannten dargestellt werden.”