cifrari a flusso → generatori di sequenze di bit

↓

le sequenze generate non devono essere distinguibili da sequenze casuali

↓

il testo cifrato non dovrebbe evidenziare proprietà statistiche rilevanti

se testo in output distinguibile da testo casuale → c'è un problema

I test si fanno direttamente sulla codifica di 0 i.e sullo stream in output del cifrario

Limitazione fondamentale: testi "vicini" sono cifrati a partire da chiave e posizione in messaggi vicini

$M = \mathbb{Z}_2^n$ $\quad d_H(\bar{x}, \bar{y}) = |\{i \mid x_i \neq y_i\}|$ distanza di Hamming.

Notiamo che d_H è una distanza.

1) $d_H(\bar{x}, \bar{y}) = 0 \iff \forall i : x_i = y_i \iff \bar{x} = \bar{y}$

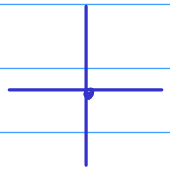
2) $d_H(\bar{x}, \bar{y}) = d_H(\bar{y}, \bar{x})$

3) $d_H(\bar{x}, z) \leq d(\bar{x}, \bar{y}) + d(\bar{y}, \bar{z})$

$$d(\bar{x}, \bar{z}) = |\{i \mid x_i \neq z_i\}| = |\{i \mid x_i \neq z_i \ \& \ y_i = x_i\}| + |\{i \mid x_i \neq z_i \ \& \ y_i \neq x_i\}|$$

$$\leq |\{i \mid y_i \neq z_i\}| + |\{i \mid x_i \neq y_i\}| =$$

$$= d(\bar{x}, \bar{y}) + d(\bar{y}, \bar{z})$$

$n = 2$ la $B_1(\underline{0}) =$ 

Se $E(\bar{m}, \bar{k})$ stream cipher

$d(\bar{m}_1, \bar{m}_2) = t \Rightarrow d(\bar{c}_1, \bar{c}_2) = t$ ove $\bar{c}_i = E(\bar{m}_i, \bar{k})$

$$\bar{c}_1 = \bar{s} \oplus \bar{m}_1$$

$$\bar{c}_2 = \bar{s} \oplus \bar{m}_2$$

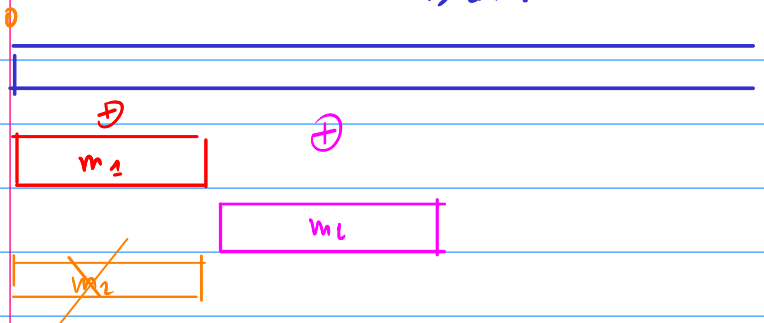
$$d(\bar{c}_1, \bar{c}_2) = |\{i \mid c_{1,i} \neq c_{2,i}\}| =$$

$$= |\{i \mid c_{1,i} \oplus c_{2,i} = 1\}| =$$

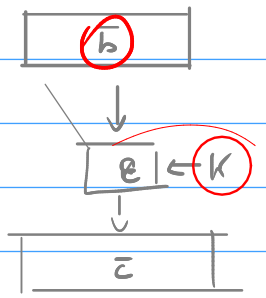
$$= w(\bar{c}_1 \oplus \bar{c}_2) = w(\bar{s} \oplus \bar{m}_1 \oplus \bar{s} \oplus \bar{m}_2)$$

ove $w(\bar{x}) = |\{i \mid x_i \neq 0\}|$ $= w(\bar{m}_1 \oplus \bar{m}_2) = d(\bar{m}_1, \bar{m}_2)$

RACCOMANDAZIONE: Evitare di codificare messaggi differenti con uno stream cipher nel medesimo stato!



Block cipher



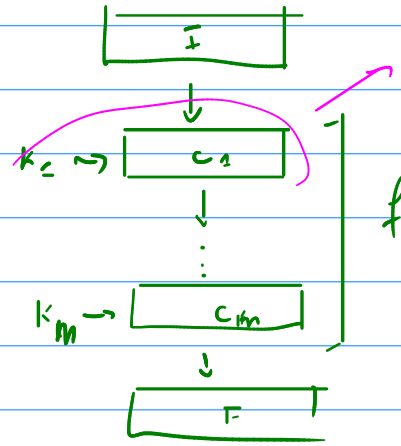
reti di permutazione/sostituzione

↓

stato interno che dipende sia da k che da b

ARCHITETTURE ITERATIVE.

E = encoding si decompone in



fasi di codifica iterate

- le proprietà di sicurezza finali dipendono dalla applicazione delle fasi centrali un determinato numero di volte. (applicarle una volta sola non basta)
- la codifica deve essere invertibile.

→ Obiettivi di sicurezza: diffusione, confusione

rispetto chiave e blocco da cifrare.

CONFUSIONE → il testo in output non deve avere proprietà statistiche evidenti

↓

non deve essere distinguibile da un testo casuale.

vogliamo anche che codifiche di messaggi "vicini" siano indistinguibili da codifiche di messaggi "lontani"

$\bar{c}_1 = \mathcal{E}(k, \bar{m}_1)$

$\bar{c}_2 = \mathcal{E}(k, \bar{m}_2)$

da $d(\bar{c}_1, \bar{c}_2) = 6$

non passa dire nulla su $d(\bar{m}_1, \bar{m}_2)$

IND-CPA

$\underline{0}$ / $\bar{m}_2$

$\underline{0}$ \ $\bar{m}_1$

$w(c_i) = w(\underline{0}) + 3w(\bar{m}_i)$

$15 + 3 = 18$

↓

15 bit = 1

90 bit = 0

Alice conosce $w(\mathcal{E}(\underline{0})) = m$

cifrario a flusso.

prende $\bar{m}_1$ con tutti i bit ad 1 di lunghezza n

$\bar{m}_2$ con 1 bit ad 1 e tutti gli altri a 0

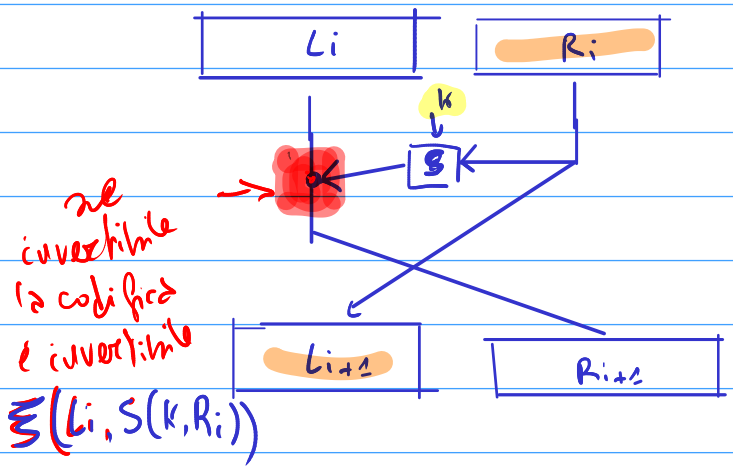
manda a bob → riceve $\bar{c} = m_i \oplus \mathcal{E}(\underline{0})$

$w(\bar{c})$ / $n - m \Rightarrow i = 1$

$m - 1 \leq w(\bar{c}) \leq m + 1 \Rightarrow i = 2$

DIFFUSIONE → ogni bit dell'output dipende da ogni bit della chiave (e del resto cifrato)

Modelli di cifrario a blocchi → rete di Feistel



ne invertibile la codifica è invertibile

$\Xi(L_i, S(k, R_i))$

$$\begin{cases} R_{i+1} = \Xi(L_i, S(k, R_i)) \\ L_{i+1} = R_i \end{cases}$$

Di solito (ma non sempre)

Ξ = XOR

oppure Ξ = somma mod 256

$$R_i = L_{i+1}$$

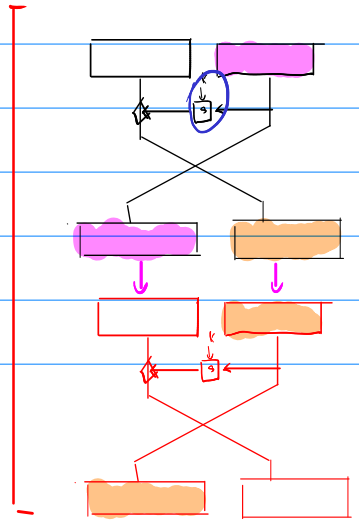
$$L_i = \Xi^{-1}(R_{i+1}, S(k, R_i)) =$$

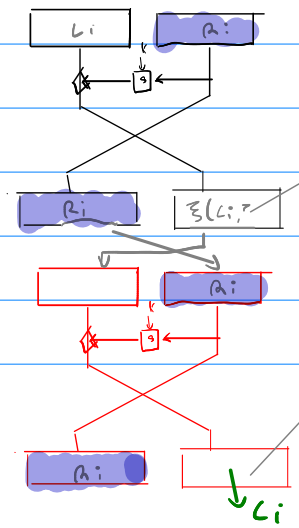
$$= \Xi^{-1}(R_{i+1}, S(k, L_{i+1}))$$

Ξ si deve poter invertire nel I argomento

MA NON SERVE CHE S sia invertibile!

come minimo serve un numero pari di iterazioni





$\Xi(L_i, S(k, R_i))$

$\Xi(\Xi(L_i, S(k, R_i)), S(k, R_i)) = \Xi^2(L_i, S(k, R_i))$

Se $\Sigma^2 = Id \Rightarrow$ otteniamo L_i in output.

questo si verifica in particolare se $\Sigma = \oplus$

↓

In questo caso codifica e decodifica sono eseguite dalla medesima macchina con le medesime chiavi (ma input alterati).

Σ invertibile — S deve essere resistente alle collisioni (ma non necessariamente invertibile)

Feistel, Lucifer, DES

DES → chiavi di 56 bit / blocchi di 64 bit.

1) per ogni blocco in ingresso di messaggi c'è solo che c'è uno e codifica di in per ognuna chiave k

2) 2^{56} possibili chiavi sembravano poche anche nel 1974 Lucifer aveva chiavi di 128 bit.

3) il modo in cui lo standard è presentato

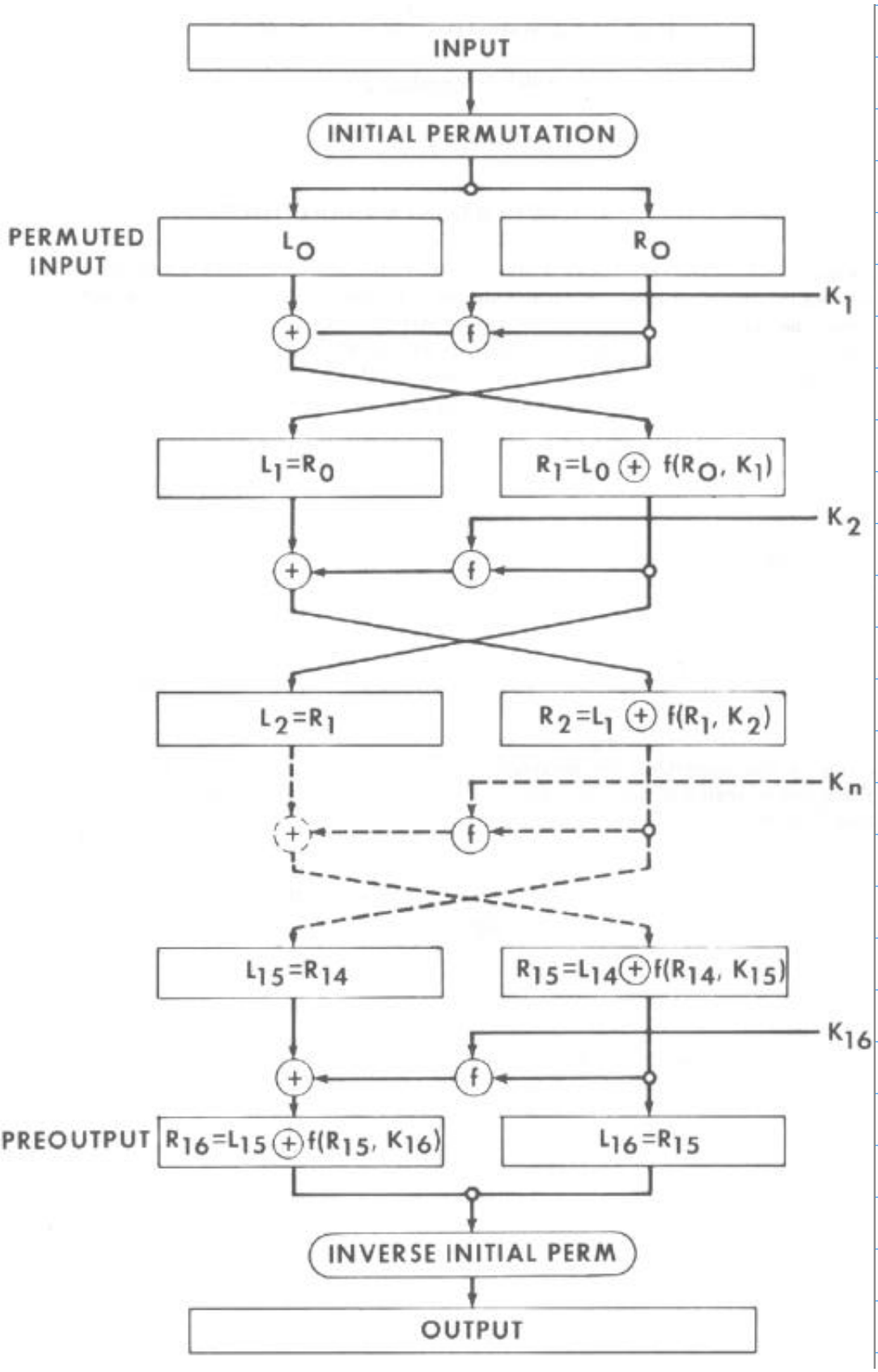
→

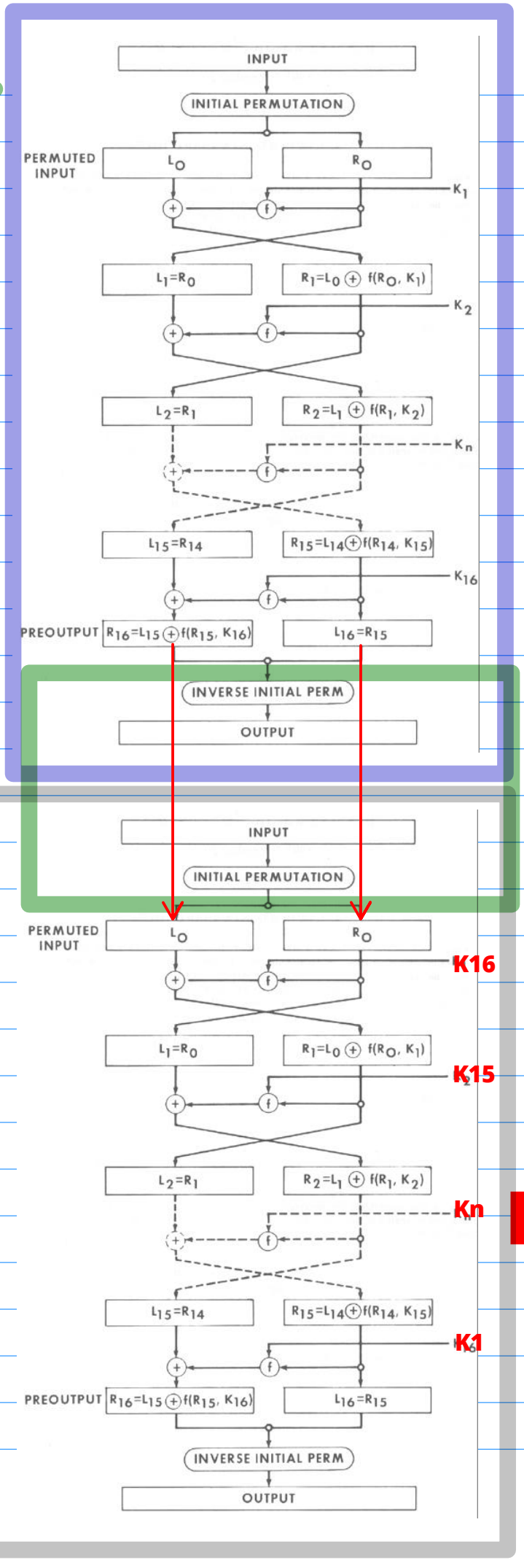
DES → ARCHITETTURA DI FEISTEL. blocchi 32+32
chiave 56 bit

$\Sigma = \oplus$

S → funzione non lineare descritta da tabelle.

iteration: 15

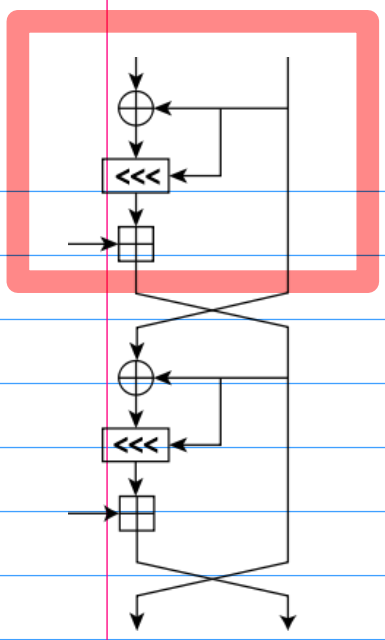




CODIFICA

DECODIFICA

Inserire le chiavi
in ordine inverso



RC5

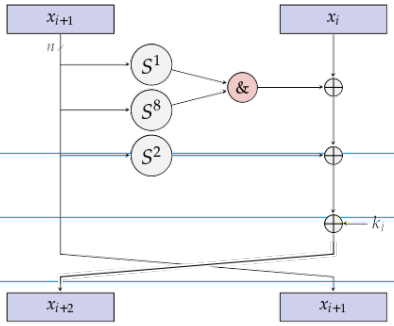


Figure 3.1: Feistel stepping of the SIMON round function.

SIMON

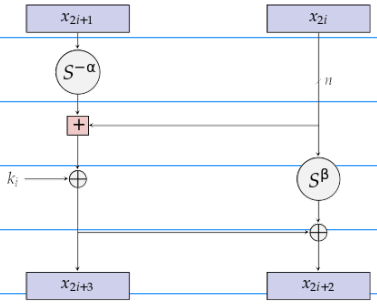
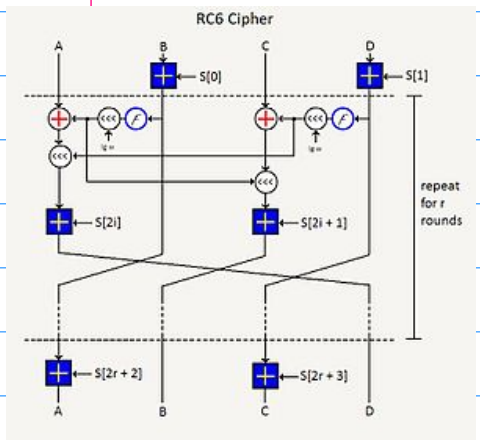
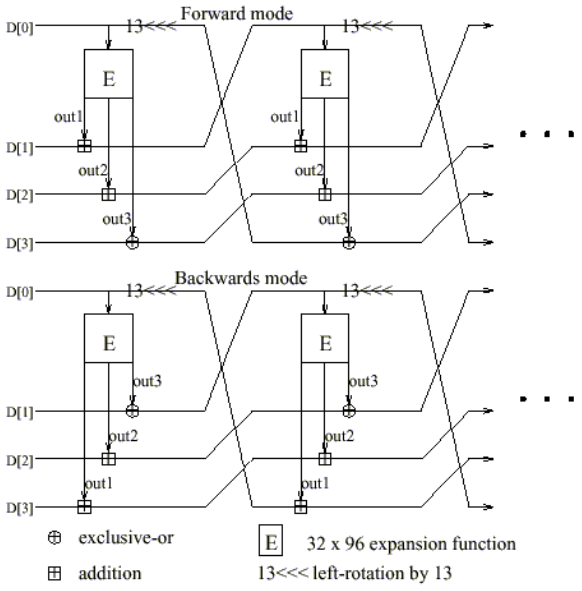


Figure 4.1: SPECK round function; (x_{2i+1}, x_{2i}) denotes the subcipher after i steps of encryption.

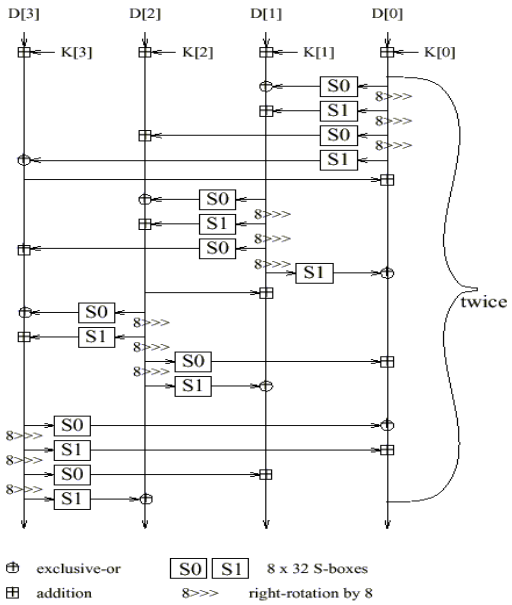
SPECK



RC6



MARS (main keyed transform)



MARS (forward mixing)