6장. 관용 암호 방식: 알고리즘

박 종 혁

(jhpark1@snut.ackr)

http://www.parkjonghyuk.net

❑ 목 차

1. 3중 DES
2. BLOWFISH
3. RC5
4. 개선된 대칭 블록 암호의 특징
5. RC4 스트림 암호
6. IDEA

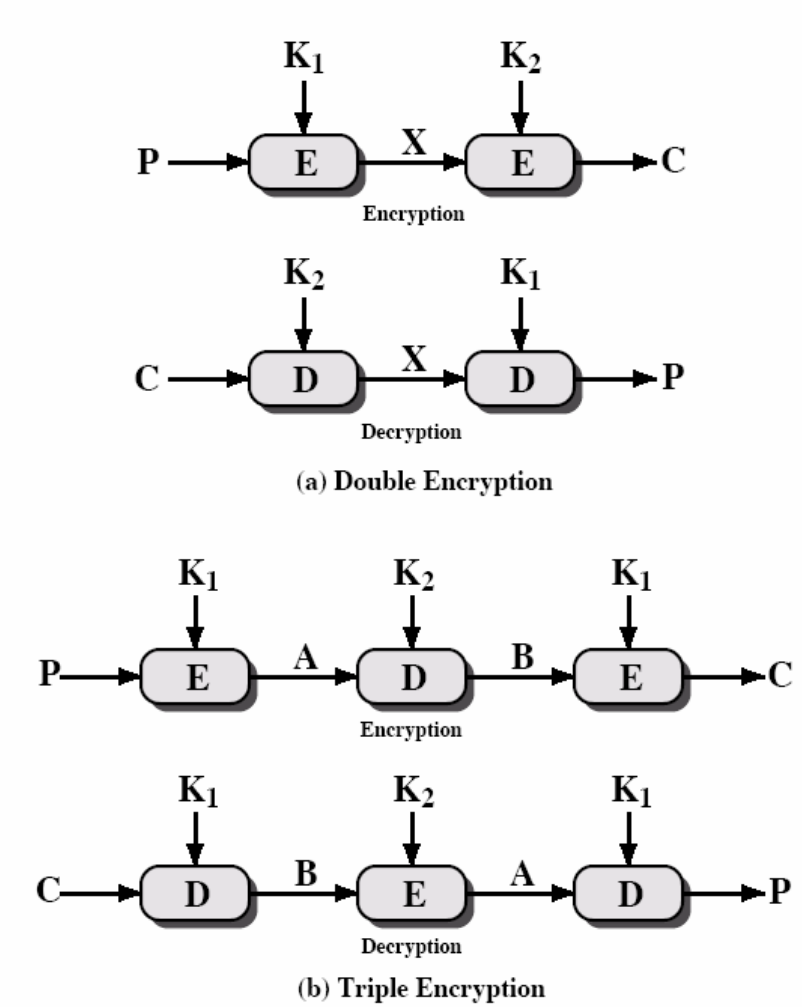
3중 DES

- 2중 DES
  - 암호화

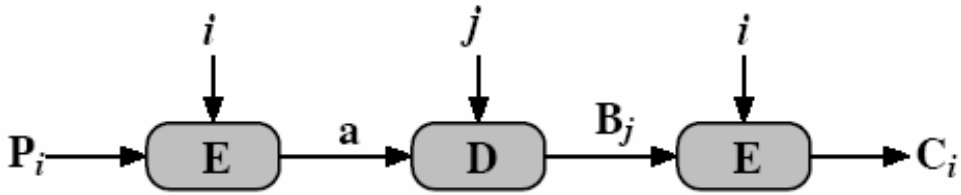
$$C = E_{K_2}[E_{K_1}[P]]$$

  - 복호화

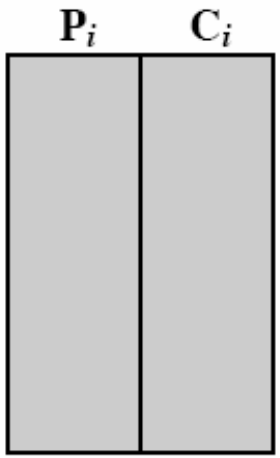
$$P = D_{K_1}[D_{K_2}[C]]$$



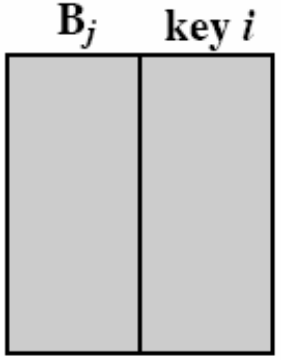
- 중간 결과에 의한 공격
- 2키 3중 DES
- 3키 3중 DES



(a) Two-key Triple Encryption with Candidate Pair of Keys



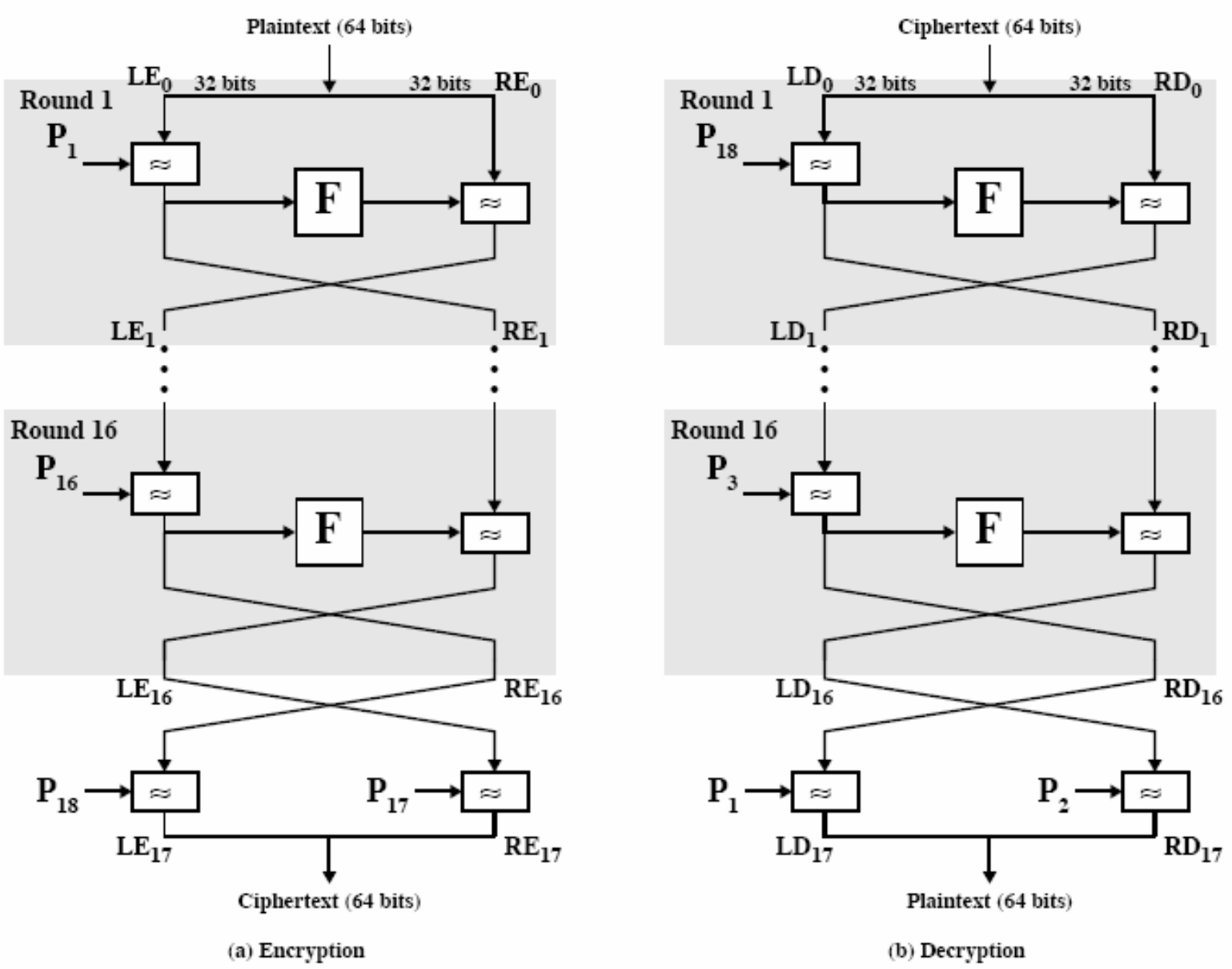
(b) Table of n known plaintext-ciphertext pairs, sorted on P

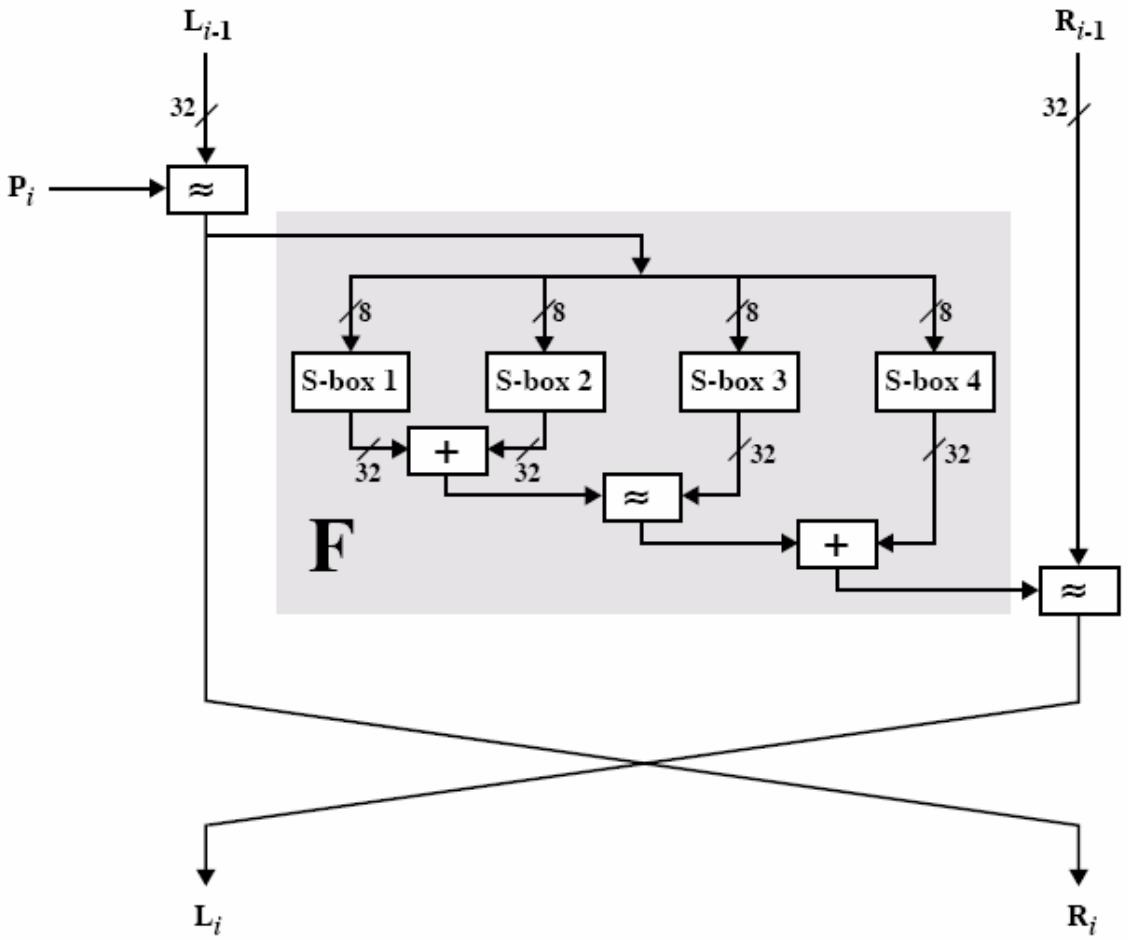


(c) Table of intermediate values and candidate keys

BLOWFISH

- ❖ 빠른 속도
  - ➢ **32**비트 마이크로 프로세스에서 **1** 바이트 당 **18**클럭 사이클의 속도로 암호화
- ❖ 간결성
  - ➢ **5K** 이내의 메모리에서는 실행 될 수 있음
- ❖ 단순성
  - ➢ 간단한 구조는 구현이 쉽고 알고리즘의 강도 결정이 용이
- ❖ 가변성
  - ➢ 키의 길이는 가변적이며 **448**비트 만큼 길어 질 수 있음
- ❖ 두개의 기본 연산
  - ➢ 덧셈 : 2^{32}를 법으로 수행되는 단어의 덧셈 연산 +
  - ➢ 비트 **XOR** 연산


Plaintext (64 bits)
LE0
32 bits
32 bits
RE0
Round 1
P1
F
LE1
RE1
Round 16
P16
LE16
RE16
P18
P17
LE17
RE17
Ciphertext (64 bits)
(a) Encryption
Ciphertext (64 bits)
LD0
32 bits
32 bits
RD0
Round 1
P18
F
LD1
RD1
Round 16
P3
LD16
RD16
P1
P2
LD17
RD17
Plaintext (64 bits)
(b) Decryption


L_{i-1}
R_{i-1}
32
32
P_i
≈
8
8
8
8
S-box 1
S-box 2
S-box 3
S-box 4
32
32
32
32
+
≈
+
≈
F
L_i
R_i

❑ 펜티엄상의 블록 암호 속도 비교

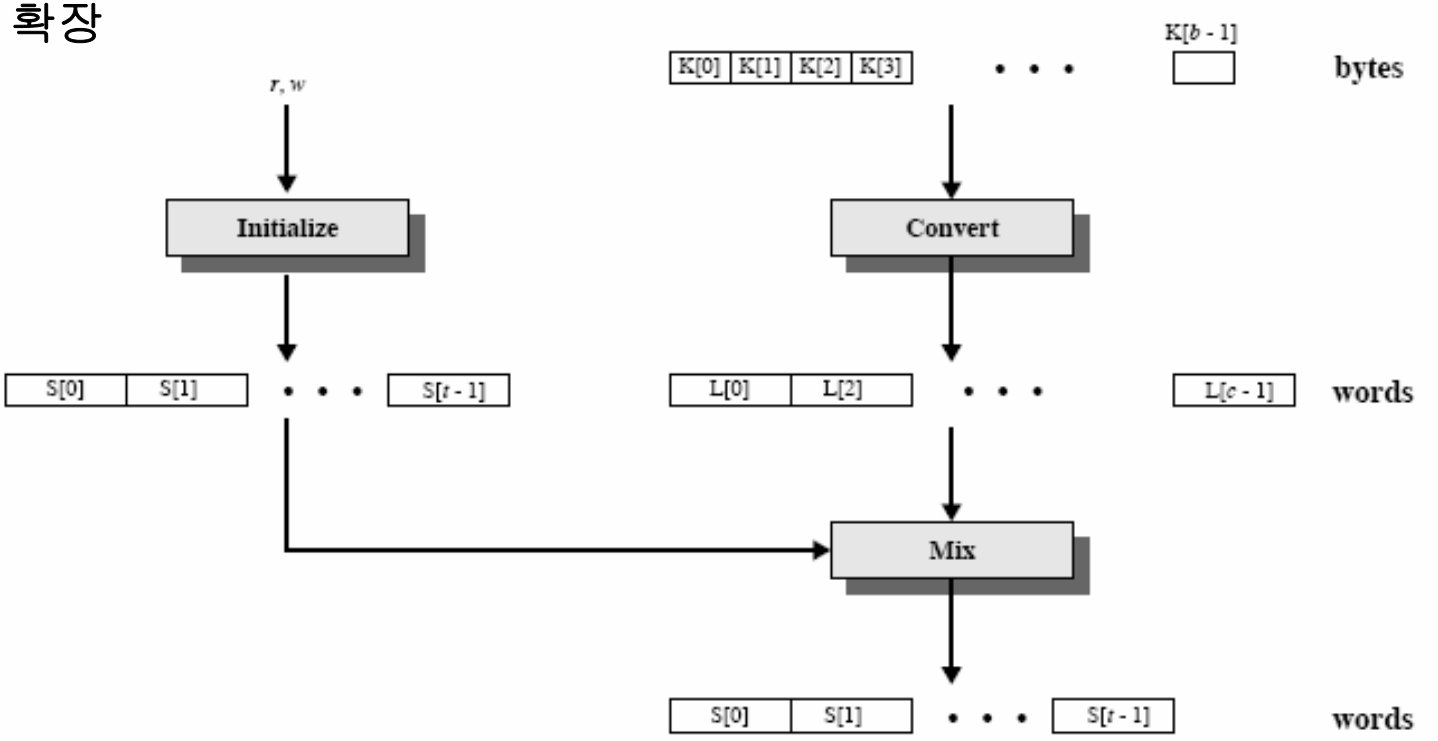
알고리즘	Clock cycles per round	# of rounds	#of clock cycles per byte encrypted
Blowfish	9	16	18
RC5	12	16	23
DES	18	16	45
IDEA	50	8	50
3중 DES	18	48	108

RC5

- 하드웨어 및 소프트웨어 적합성
  - 마이크로프로세서에서 일반적으로 사용되는 기본 연산만 사용
- 빠른 속도
  - 단어 지향적, 기본 연산은 한번에 데이터 단어 전체를 처리
- 다른 단어 길이 프로세서에서의 적응성
  - 단어 당 비트수는 **RC5**의 매개 변수
- 반복수의 가변성
- 가변의 길이의 키
- 단순성
- 낮은 메모리 요구량
- 높은 보안성
- 데이터 의존적인 순환 이동

매개 변수	정의	허용 값
w	단어의 비트수 RC5는 2 단어 블록씩 암호화함.	16, 32, 64
r	반복 횟수	0, 1, ..., 255
b	비밀키 K 내의 8비트 바이트(옥텟)수	0, 1, ..., 255

❖ 키 확장



❑ RC5 암호화 복호화

❖ 연산

- ➢ 덧셈 : +로 표시되는 단어의 덧셈은 2^w를 법으로 수행, 역 연산은 뺄셈으로 –로 표시되며 2^w를 법으로 수행
- ➢ 비트 **XOR** 연산 : ⊕로 표시됨
- ➢ 좌측 순환 이동 : **x**<<<**y**로 표시됨, 역 연산인 단어 **x**를 **y**비트 우측 순환 이동하는 연산은 **x**>>>**y**로 표시

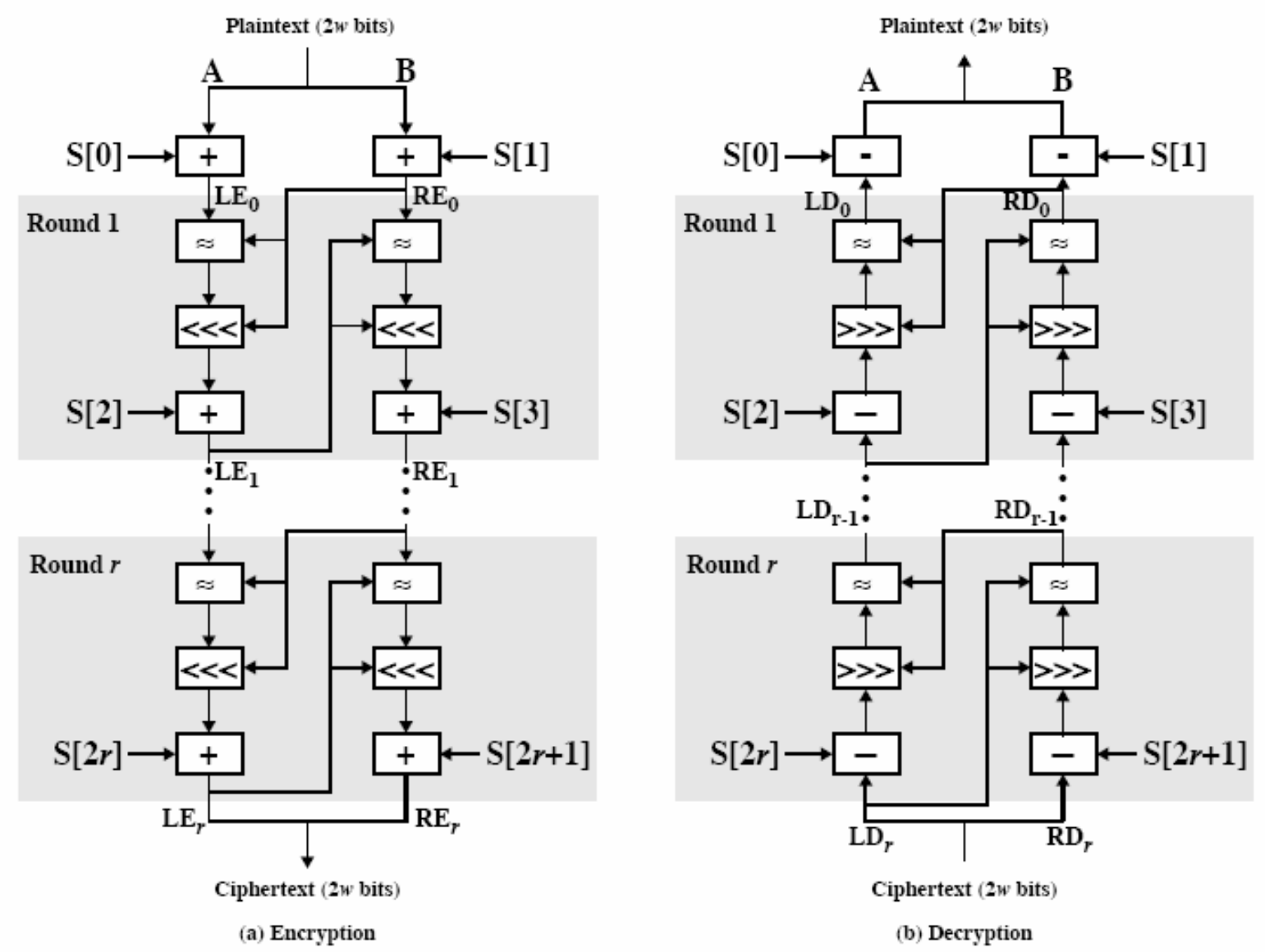
❑ RC5 모드

❖ RC5 블록 암호 모드

❖ RC5-CBC 모드

❖ RC5-CBC-Pad 모드

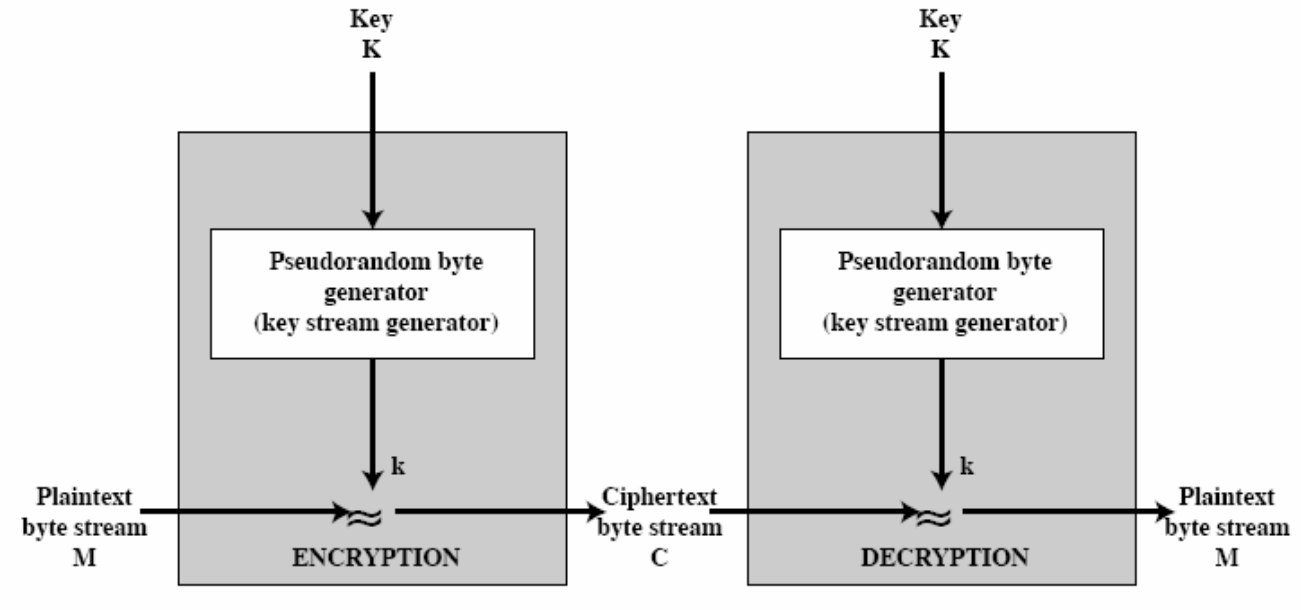
❖ RC5-CTS 모드


Plaintext (2w bits)
A
B
S[0]
S[1]
+
LE0
RE0
Round 1
≈
<<<
S[2]
S[3]
LE1
RE1
Round r
S[2r]
S[2r+1]
LEr
REr
Ciphertext (2w bits)
(a) Encryption
Plaintext (2w bits)
-
LD0
RD0
>>>
–
LDr-1
RDr-1
LDr
RDr
Ciphertext (2w bits)
(b) Decryption

개선된 대칭 블록 암호의 특징

- 가변의 키 길이
  - 키 길이가 길수록 전사적 키 탐색에 걸리는 시간 증가
- 혼합 연산자
  - 산술 및 부울 연산자중 하나 이상을 사용할 경우 암호 해독이 복잡해짐
  - 분배 및 결합 법칙을 따르지 않으면 암호 해독은 더욱 어려워짐
- 데이터 의존 회전 이동
  - 충분한 횟수의 반복 과정이 수행될 경우 우수한 혼돈과 확산 효과를 제공할 수 있음
- 키 의존 회전 이동
- 긴 키 스케줄 알고리즘
- 가변적 F
- 가변적 평문/암호문 블록 길이
- 가변적 반복 횟수
- 매 반복 시 양 데이터 절반의 연산

RC4 스트림 암호



❑ 펜티엄 2 기반의 대칭 암호의 속도 비교

Cipher	Key Length	Speed(Mbps)
DES	56	9
3DES	128	3
RC2	Variable	0.9
RC4	Variable	45

IDEA : International Data Encryption Algorithm

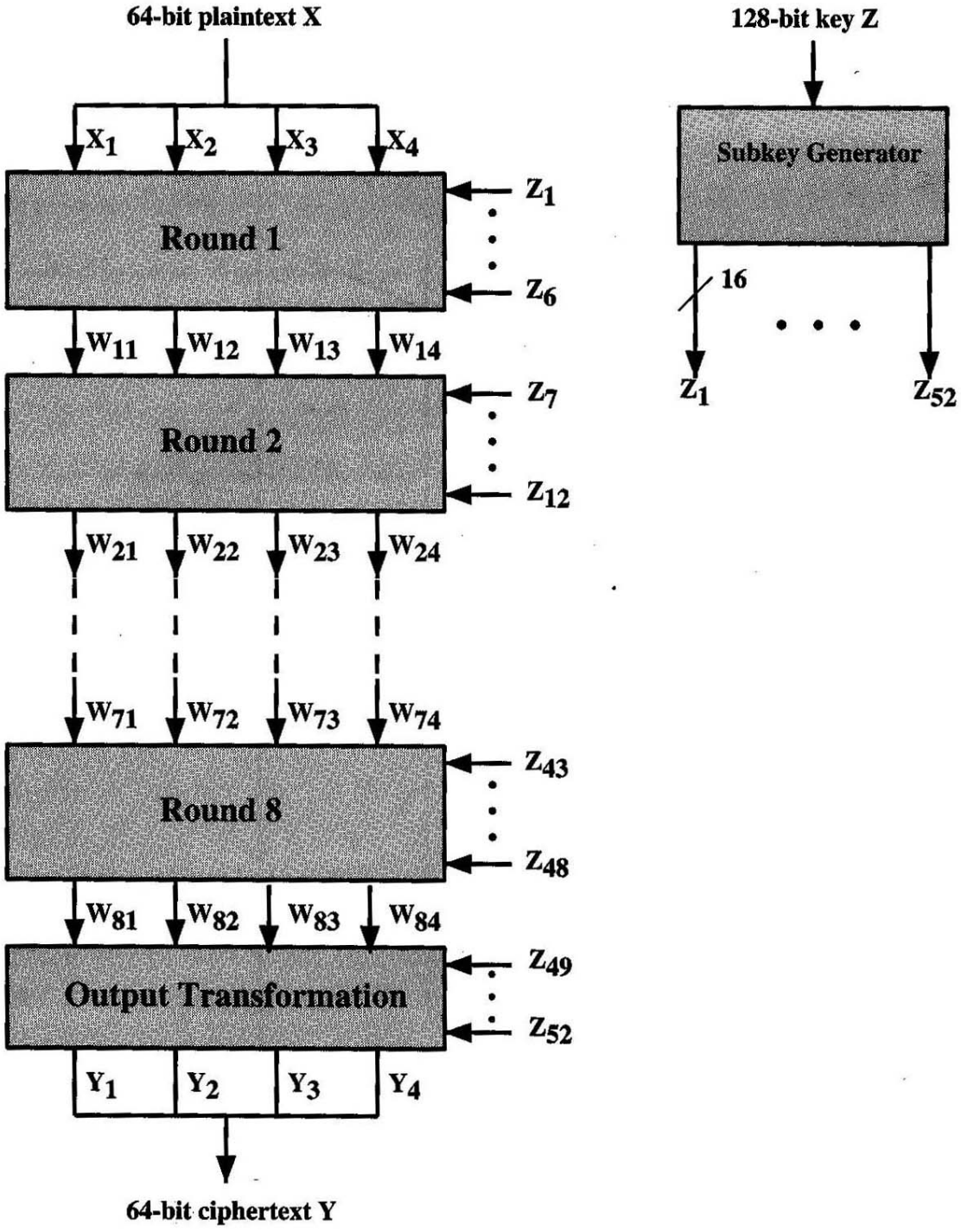
- 스위스 연방 기술 연구소의 Xueja Lai와 James Massey에 의해 1990년에 개발

- DES를 대체하기 위해 제안된 관용 암호 알고리즘 중 하나

- 가장 성공적인 DES의 대체 알고리즘
  - ❖대부분의 암호 공격으로부터 안전
  - ❖E-mail 암호를 위한 PGP에 포함

- 설계원리
  - ❖64비트 블록의 데이터 입력
  - ❖128비트의 키를 사용
  - ❖블록 암호

IDEA 암호화


64-bit plaintext X
128-bit key Z
Subkey Generator
16
Z_1
Z_{52}
X_1 X_2 X_3 X_4
Round 1
Z_1
Z_6
W_{11} W_{12} W_{13} W_{14}
Round 2
Z_7
Z_{12}
W_{21} W_{22} W_{23} W_{24}
W_{71} W_{72} W_{73} W_{74}
Round 8
Z_{43}
Z_{48}
W_{81} W_{82} W_{83} W_{84}
Output Transformation
Z_{49}
Z_{52}
Y_1 Y_2 Y_3 Y_4
64-bit ciphertext Y

❑ 암호학적 강도

- ❖ 블록길이
  - ➢ 통계적 분석을 막을 수 있을 만큼 길어야 함
  - ➢ **64비트**의 블록이면 충분
- ❖ 키의 길이
  - ➢ 모든 키의 탐색**(Exhaustive Search)**을 효율적으로 막을 수 있을 만큼 커야 함
  - ➢ **128비트**면 향후에도 안전할 것이라고 여겨짐
- ❖ 혼돈(Confusion)
  - ➢ 목적 **:** 암호문의 통계적 성질이 평문의 통계적 성질에 의존하는지에 대한 결정을 복잡하게 만드는 것
  - ➢ 세가지 연산 **: XOR,** 덧셈**,** 곱셈
- ❖ 확산(Diffusion)
  - ➢ 목적 **:** 각 평문 비트는 모든 암호문 비트에 영향을 끼쳐야 하고**,** 각 키 비트는 모든 암호문 비트에 영향을 주어야 함

❑ 혼돈을 위한 연산

- ❖ 입력 : 16비트
- ❖ 출력 : 16비트
- ❖ 연산
  - ➢ **XOR** 연산
  - ➢ 덧셈연산 : 법을 **2^16**로 하는 덧셈
  - ➢ 곱셈연산 : 법을 **2^16+1**로 하는 곱셈 ⊙

=> 세가지 연산을 조합함으로써 DES보다 암호해독을 더 어렵게 함

❑ 확산을 위한 구조

❖ 곱셈/덧셈 구조에 의해 제공

⊕ : bit by-bit exclusive-OR

⊞ : Addition of integers modulo 2^{16}

❖ ⊙ : Multiplication of integers modulo $2^{16} + 1$

❖ 효율적인 확산을 위해 8번 반복

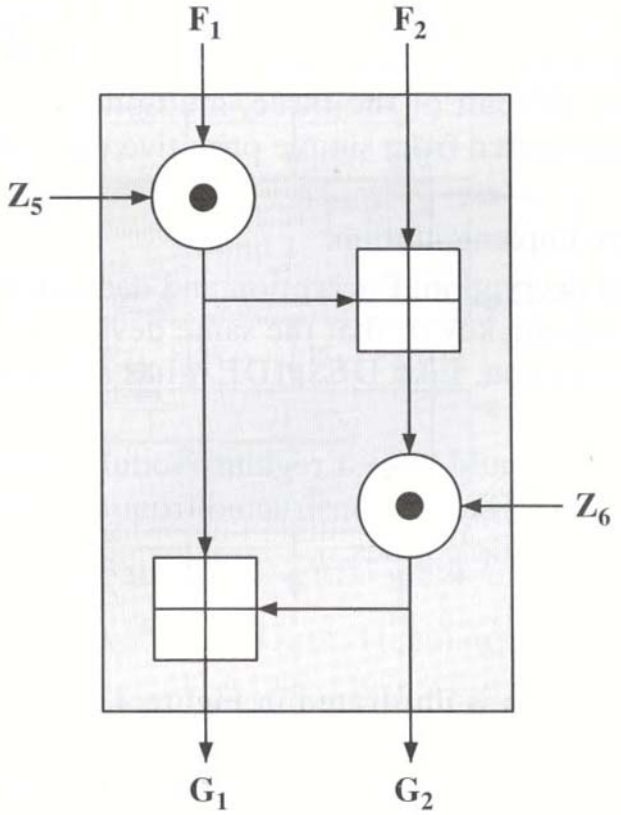


Figure 4.3 Multiplication/Addition (MA) Structure.

❑ 구현상의 고려 사항

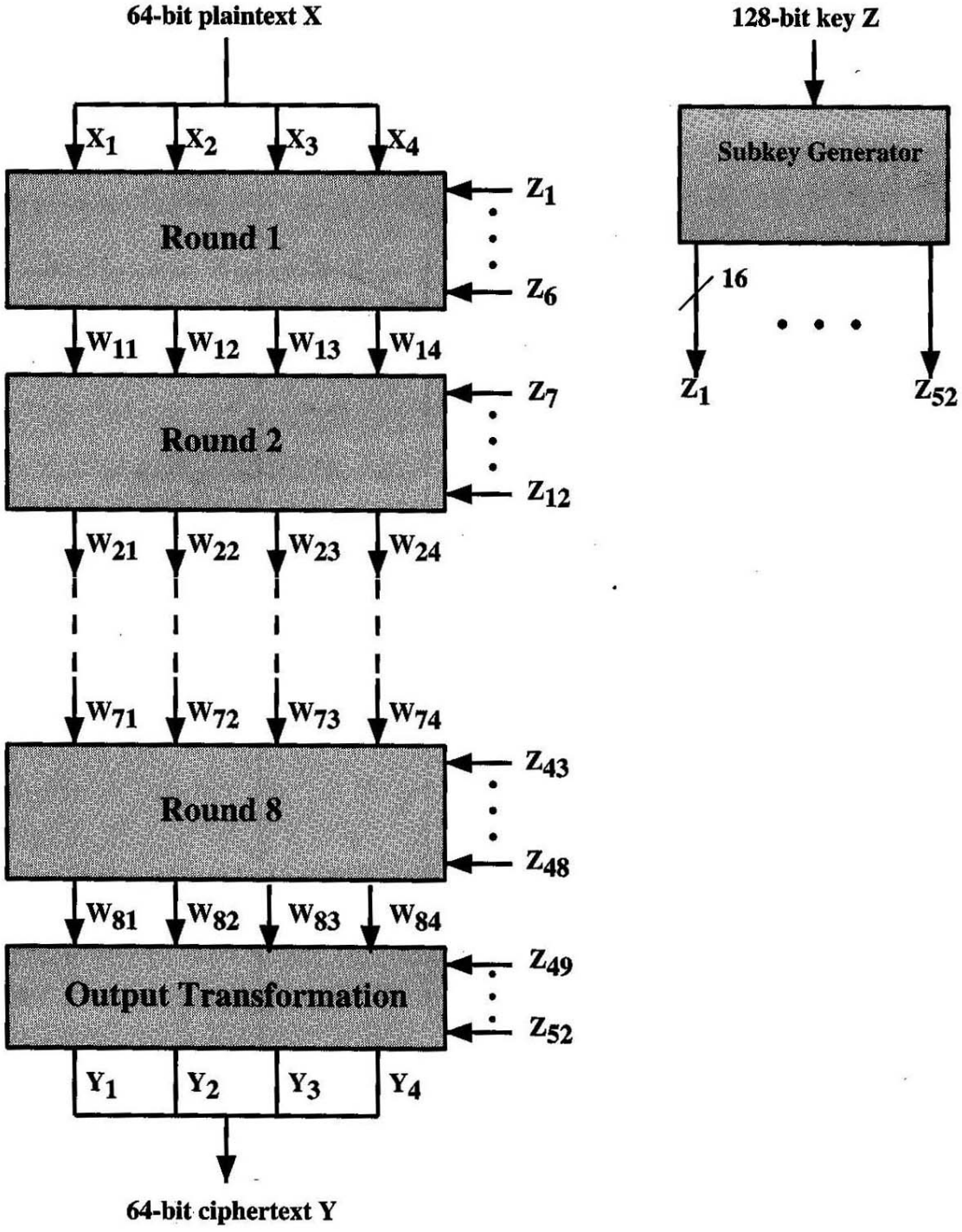
❖ 소프트웨어 구현을 위한 설계 원칙

- ➢ 서브블록의 사용
  - 암호연산은 소프트웨어에 대해 당연히 **8, 16, 32**비트와 같은 서브 블록에서 동작하도록 한다
  - **IDEA**는 **16**비트 서브 블록을 사용
- ➢ 간단한 연산의 사용
  - 덧셈, 자리 이동 등을 사용하여 쉽게 프로그램 되어야 함
  - **IDEA**의 기본 연산은 이 요구사항을 만족

❖ 하드웨어 구현에 대한 설계 원칙

- ➢ 암호화 복호화의 유사성
  - 암호화와 복호화는 키를 사용하는 방법에서만 달라야 함
- ➢ 정규구조
  - **VLSI** 구현을 용이하게 하기 위한 정규적인 모듈 구조를 가져야 함

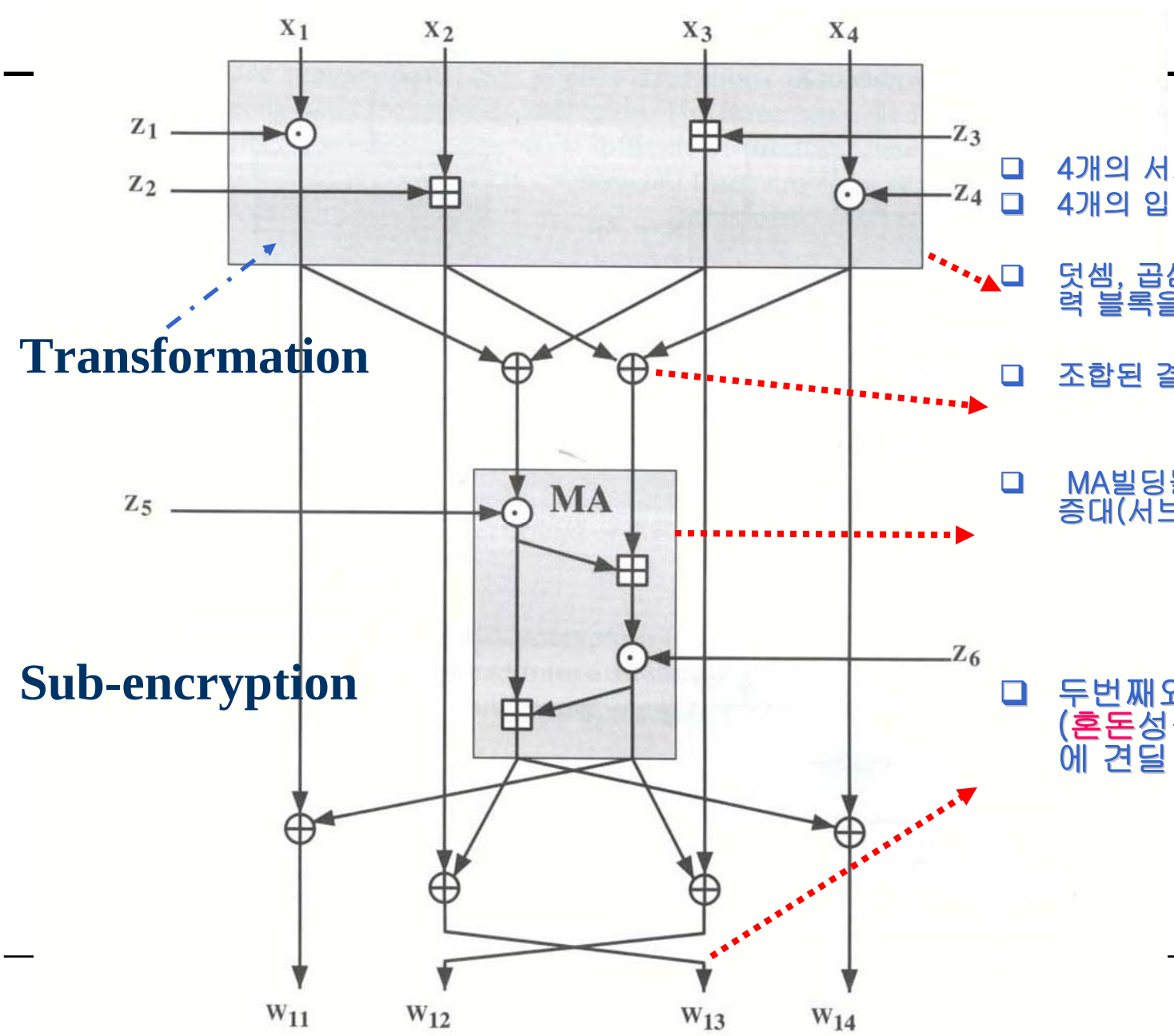
IDEA 암호화


64-bit plaintext X
128-bit key Z
Subkey Generator
16
Z_1
Z_{52}
X_1 X_2 X_3 X_4
Round 1
Z_1
Z_6
W_{11} W_{12} W_{13} W_{14}
Round 2
Z_7
Z_{12}
W_{21} W_{22} W_{23} W_{24}
W_{71} W_{72} W_{73} W_{74}
Round 8
Z_{43}
Z_{48}
W_{81} W_{82} W_{83} W_{84}
Output Transformation
Z_{49}
Z_{52}
Y_1 Y_2 Y_3 Y_4
64-bit ciphertext Y

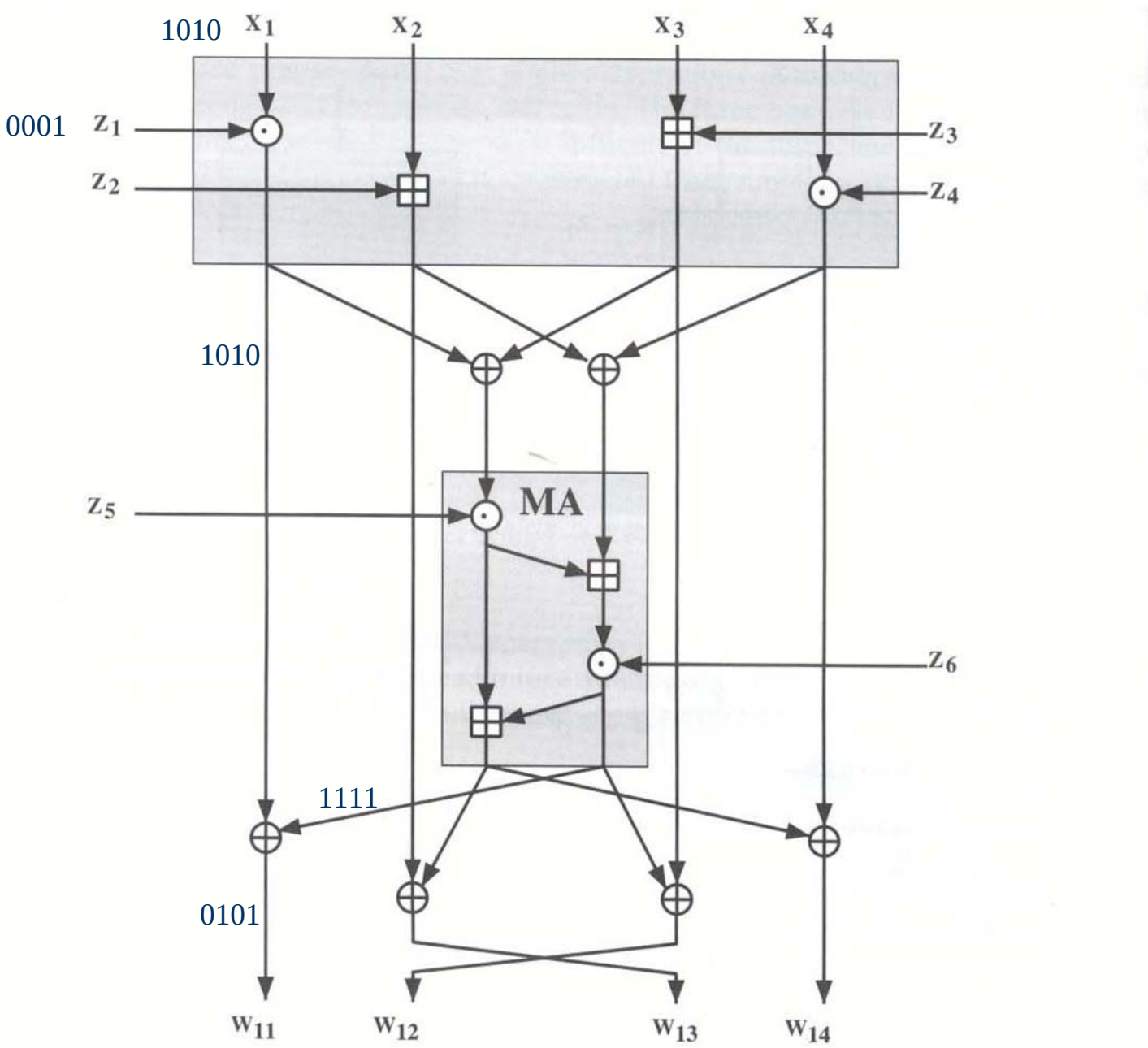
❑ IDEA 암호화

- ❖ 입력
  - ➢ 평문 **64** 비트
  - ➢ 키 **128**비트
- ❖ 반복횟수 : 전체 8라운드
- ❖ 알고리즘
  - ➢ 입력을 **4**개의 **16**비트 서브블록으로 분해
  - ➢ 각 반복은 **4**개의 **16**비트 서브블록들을 처리
  - ➢ 서브키**(52**개**)**
    - 각 라운드에 **6**개의 **16**비트 서브키를 이용
    - **6*8 = 48**
    - 최종 변환은 **4**개의 서브키 사용

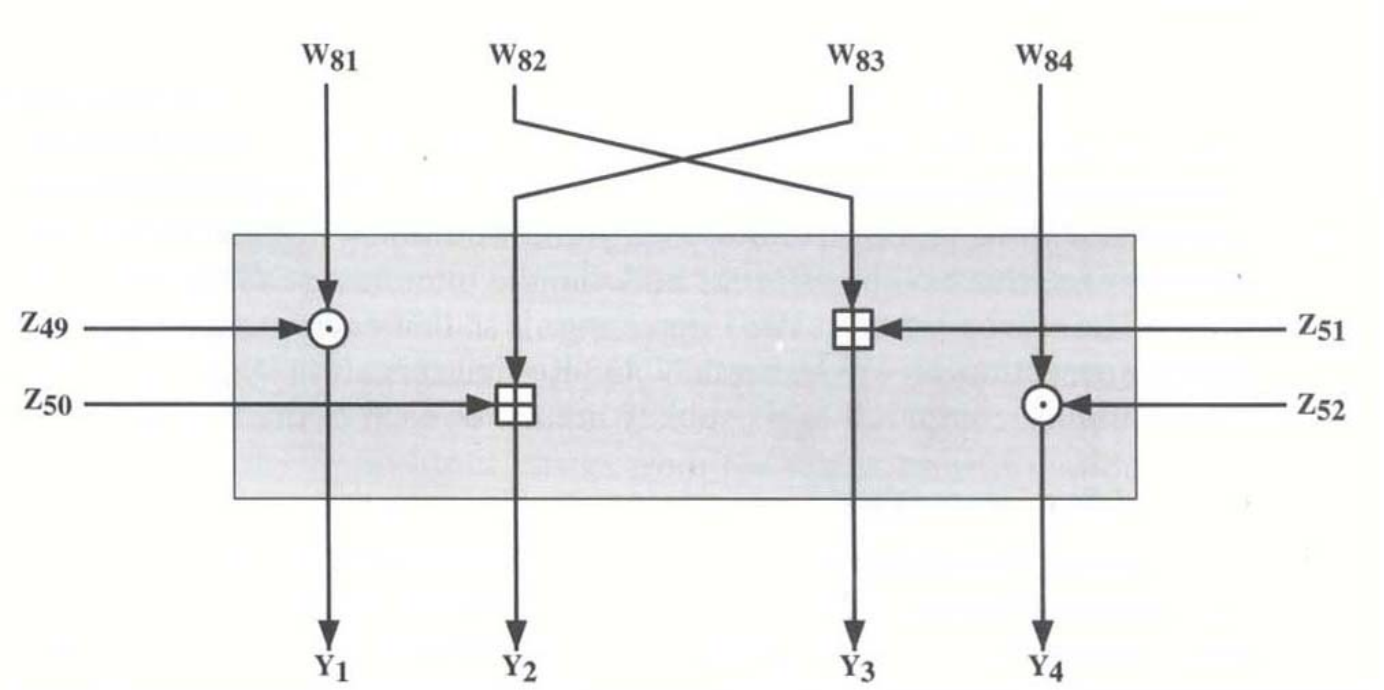
단일 과정



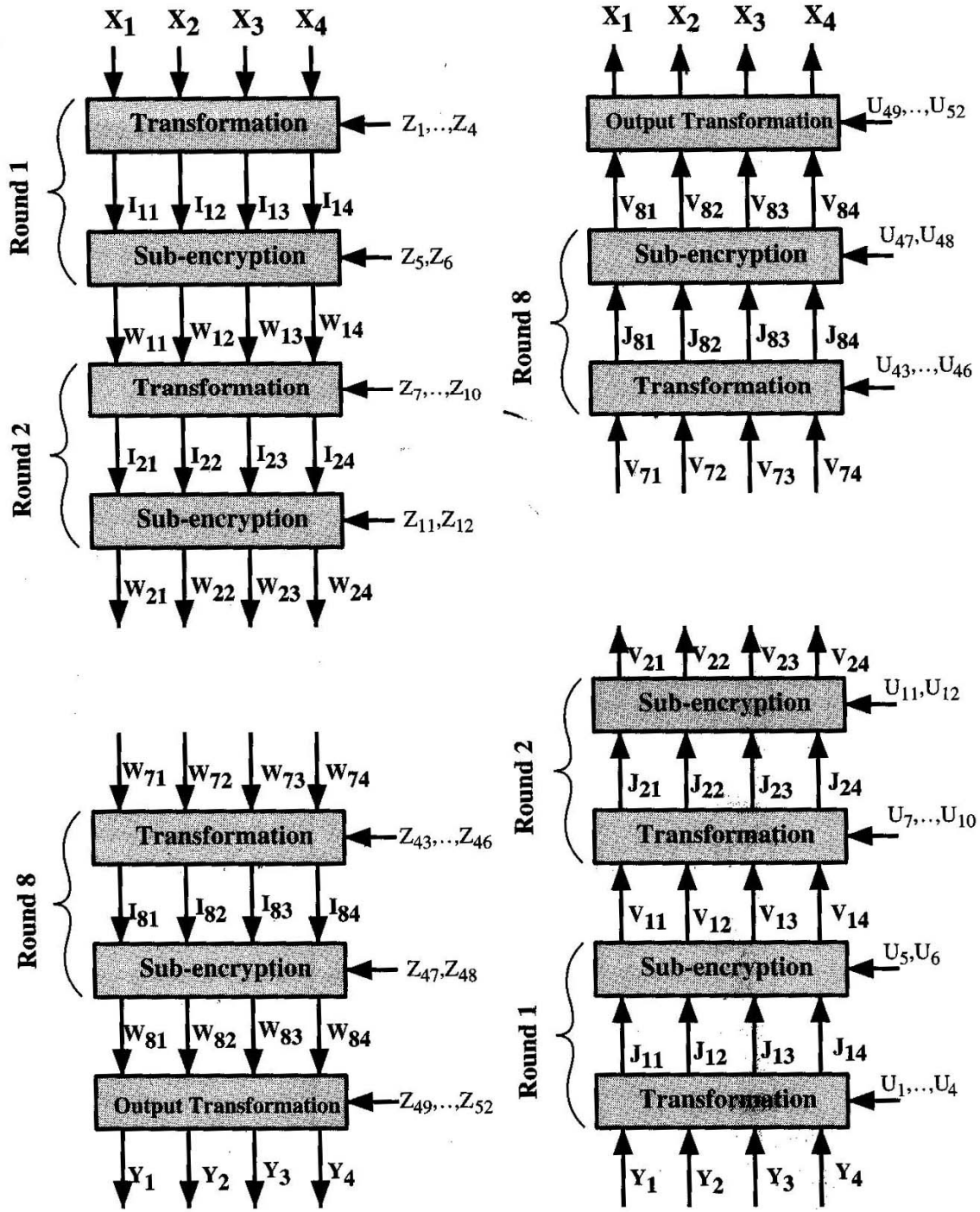
- 4개의 서브키(Z1 ~ Z4)
- 4개의 입력 블록(X1~X4)
- 덧셈, 곱셈연산을 이용해서 키와 입력 블록을 조합(변환과정)
- 조합된 결과를 XOR
- MA빌딩블록은 확산성질의 효과를 증대(서브 암호화 과정)
- 두번째와 세번째의 결과 교환 (혼돈성질의 증가 및 차분해독에 견딜 수 있는 성질)


1010
x_1
x_2
x_3
x_4
0001
z_1
z_2
z_3
z_4
1010
z_5
MA
z_6
1111
0101
w_{11}
w_{12}
w_{13}
w_{14}

❑ 출력 변환 단계



❑ 암/복호화

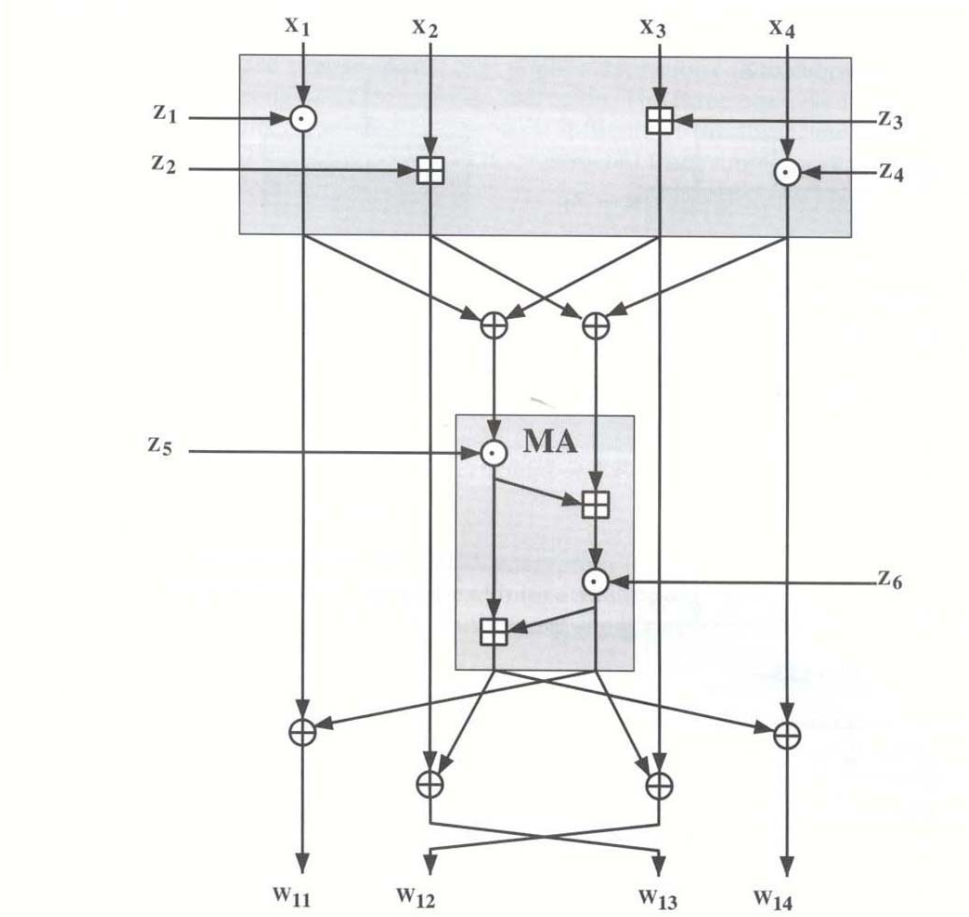

X1 X2 X3 X4
Transformation
Z1,..,Z4
I11 I12 I13 I14
Sub-encryption
Z5,Z6
W11 W12 W13 W14
Transformation
Z7,..,Z10
I21 I22 I23 I24
Sub-encryption
Z11,Z12
W21 W22 W23 W24
Round 1
Round 2
W71 W72 W73 W74
Transformation
Z43,..,Z46
I81 I82 I83 I84
Sub-encryption
Z47,Z48
W81 W82 W83 W84
Output Transformation
Z49,..,Z52
Y1 Y2 Y3 Y4
Round 8
X1 X2 X3 X4
Output Transformation
U49,..,U52
V81 V82 V83 V84
Sub-encryption
U47,U48
J81 J82 J83 J84
Transformation
U43,..,U46
V71 V72 V73 V74
Round 8
V21 V22 V23 V24
Sub-encryption
U11,U12
J21 J22 J23 J24
Transformation
U7,..,U10
V11 V12 V13 V14
Sub-encryption
U5,U6
J11 J12 J13 J14
Transformation
U1,..,U4
Y1 Y2 Y3 Y4
Round 2
Round 1

❑ IDEA 복호화

- ❖ 복호를 위한 서브키 생성
  - ➢ $Z_1^{-1}, -Z_2, -Z_3, Z_4^{-1}$

- ❖ 기존 알고리즘의 연산
  - ➢ **XOR**

  => **XOR** 하면 이전 값 복원
- ❖ 예) 키 : 1111, 연산값: 1001
  - ➢ **XOR: 1111(+)1001(+)1111**
  - ➢ 덧셈: **1111+1001-1111**
  - ➢ 곱셈: **1111*1001*x**
  - ➢ **1111*x mod 2^5-1 = 1**



IDEA의 동작 모드

- DES와 유사한 4가지 동작 모드
- ECB
  - 64비트의 각 평문 블록이 독립적으로 암호화
  - 작은 블록의 데이터 암호화에 유용
- CBC
  - 평문의 다음 64비트와 암호문의 이전 64비트에 대한 XOR
  - 같은 64비트 평문이라도 매번 다른 암호문 생성
- CFB
  - 입력은 한번에 J 비트로 처리
  - J비트 암호문이 다음 암호화에 입력
- OFB(Output feedback)
  - 평문과 XOR 되기 이전의 IDEA 출력이 다음 암호화에 입력
  - 잡음이 있는 채널에서의 스트림 전송에 유효

Thanks

Q & A